UNIVERSITY OF HOUSTON – CLEAR LAKE

SENIOR PROJECTS

CINF 4388.01

Software Requirements Specification

Authors
Ogheneovo (*Nova*) ABU
Manuel BERLANGA
Ethan HANN
Thu LE
Mark ODOM

Instructor
Dr. Alfredo PEREZ-DAVILA

September 24, 2019

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to provide a clear outline to the designers, programmers, and testers for the development of the *Reverse Recipe App*. By using this document, the team members working on this project should have a clear understanding of how to proceed in the development of this project.

1.2 Scope

This document contains a complete description of the functionality of the *Reverse Recipe App*. It consists of use cases, functional requirements, and non-functional requirements. Together, these form a complete description of the software.

1.3 System overview

The *Reverse Recipe App* is a smart-phone application for Android that allows the user to perform a reverse recipe search based on their ingredients. The system should allow the user to add and delete items from their ingredients, search recipes based on their ingredients, save/favorite recipes they may frequently use, and keep a shopping list.

2 Definitions

2.1 Terms Defined

ANDROID	a mobile operating system developed by Google; has multiple release versions
THEMEALDB	an open source database of recipes from around the world with a free API
API	application programming interface; an interface between a client and a server that simplifies building client-side software
UC	use case
FR	functional requirement
NF	non-functional requirement

2.2 Traceability

Throughout this document, the use cases, functional, and non-functional requirements are given unique identifiers so that they are traceable and easily referenced.

For use cases, an identifier of UC-## is used where ## corresponds to the ID of that use case. In a similar manner, the functional and non-functional requirements are named where the former has a prefix of FR and the latter has a prefix of NF.

Every use case, functional requirement, and non-functional requirement also has a unique name.

3 Use cases

3.1 UC-01 - Search for Recipes based on Ingredients

Summary	Recipes are searched for based on the ingredients provided by the user.
Rationale	When planning to cook, many users may not know what it is they can cook. The search for recipes function allows the user to quickly get recipes that they can cook without having to manually check a website. Users will also save money by utilizing existing ingredients in their pantry to search for a recipe they can cook.
Users	Any user
Preconditions	A list of ingredients exists and the software has Internet connection
Basic Course of Events	1. The user indicates that the software is to search by their ingredients. 2. The software responds by showing the user their ingredients list. 3. The user indicates which ingredients are to be included in the search. 4. The software searches a web resource based on the selected ingredients.
Alternative Paths	1. In step 3, the user may de-select an item from their ingredients list. In this case, the search function does not include that ingredient.
Postconditions	A list of search results is displayed to the user.

3.2 UC-02 - Save Frequently Used Recipes

Summary	Recipes are saved in the user's favorites.
Rationale	User's may have a recipe they enjoy and do not want to keep searching for. With the save frequently used recipes function, the user can save these recipes for easy access in the future.
Users	Any user
Preconditions	Software is displaying search results from UC-01.
Basic Course of Events	1. The user indicates that a recipe is to be saved to their favorites. 2. The software responds by adding that recipe to their favorites for future use.
Alternative Paths	NA
Postconditions	See UC-05

3.3 UC-03 - Edit Pantry Items

Summary	Ingredients are saved in the user's pantry. The user is able to add or delete ingredients from their pantry.
Rationale	In order to be able to search by ingredients as in UC-01, the user must have a list of ingredients that can be used. This list also serves as a way for the user to see at a glance what ingredients they have.
Users	Any user
Preconditions	NA
Basic Course of Events	1. The user indicates that an ingredient is to be added to their pantry. 2. The software responds by asking for the ingredient name, primary food tag (cold, hot, or room temperature), and an optional tag for the ingredient. 3. The user specifies values for the options in step 2. 4. The software adds the ingredient to the pantry and categorizes it by primary and optional tag.
Alternative Paths	1. In step 2, the user may not enter an optional tag. In this case, the software still adds the ingredient to the pantry but only categorizes it by primary tag. 2. In step 1, the user may indicate that an item is to be deleted from the pantry. In this case, the software responds by removing that ingredient and updating the rest of the pantry to reflect the change.
Postconditions	The software displays the updated pantry with the newly added ingredient.

3.4 UC-04 - Edit Shopping List

Summary	The shopping list is edited by the user.
Rationale	The user may want a way to keep track of needed ingredients for recipes they want to cook. The shopping list provides a way for the user to track what they need while they are out shopping.
Users	Any user
Preconditions	NA
Basic Course of Events	1. The user indicates that an item is to be added to the shopping list. 2. The software responds by displaying the keyboard and allows the user to type what they want. 3. The shopping list is updated to reflect the newly typed item.
Alternative Paths	1. In step 1, the user may select an already added item. In this case, the item selected is struck through to indicate the item has been acquired. It is not deleted from the list. 2. In step 1, the user may indicate they wish to delete an item from the list. In this case, the software deletes that item and updates the list to reflect the change. 3. In step 1, the user may indicate they wish to delete the whole shopping list. In this case, the software responds by asking for confirmation. If the user confirms, the shopping list is deleted and a blank shopping list is created.
Postconditions	The software displays the shopping list.

3.5 UC-05 - View the Saved Recipes

Summary	The view saved recipes function allows the user to view recipes they have saved from UC-02.
Rationale	When the user saves a recipe for later user as in UC-02, they will want a way to review the recipes at a later time. The view saved recipes function will allow the user access to previously saved recipes. This allows recipes to be quickly accessible and save the user time.
Users	Any user
Preconditions	There must be at least one saved recipe.
Basic Course of Events	1. The user indicates that they want to view their saved recipes. 2. The software responds by displaying their saved recipes from UC-02.
Alternative Paths	1. In step 2, the user may indicate to unfavorite a saved recipe. In this case, the recipe is removed from their favorites. 2. In step 2, the user may indicate to see more detail about a recipe. In this case, the software displays the recipe steps and ingredients in more detail.
Postconditions	The software displays/updates the saved recipes.

4 Functional requirements

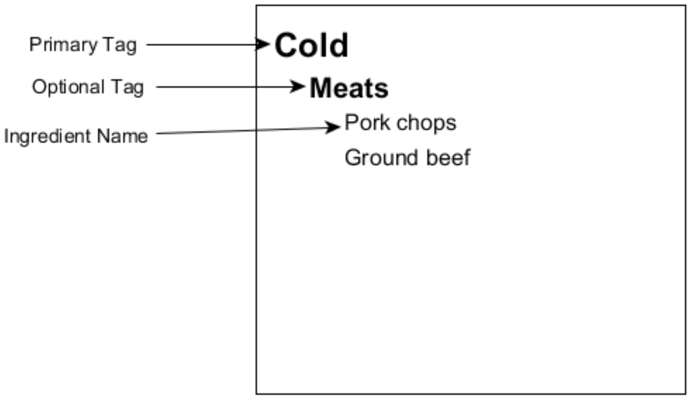
4.1 FR-01: Searching for recipes

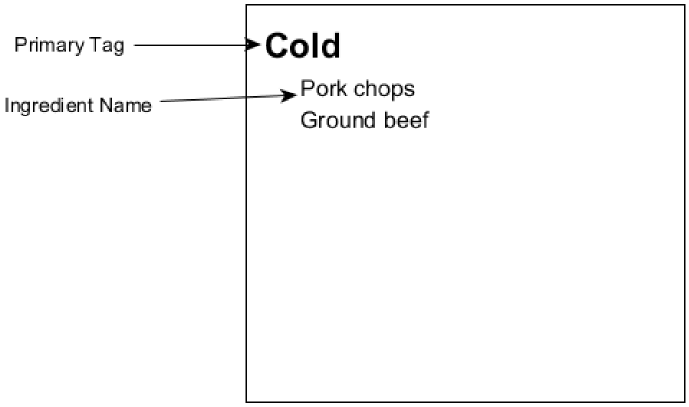
Summary	The search feature must allow searching of recipes by ingredients.
Rationale	This is the system's core feature that users will expect to function correctly.
Requirements	Once the user indicates that a search is to be performed, the system shall initialize the search. This will happen in a 3 step process: 1. The system will first ask the user for the ingredients to be used in the search. 2. This information is then included as a filter when searching. 3. The system will then set up TheMealDB API to begin searching. After initialization, the system will commence the search and be returned with data corresponding to the search parameters. The search function should organize the data returned by prioritizing that data which requires the least amount of time/steps.
Reference(s)	UC-01, NF-01, NF-02, NF-03, Section 2 (Definitions) of this document

4.2 FR-02: Favoriting recipes

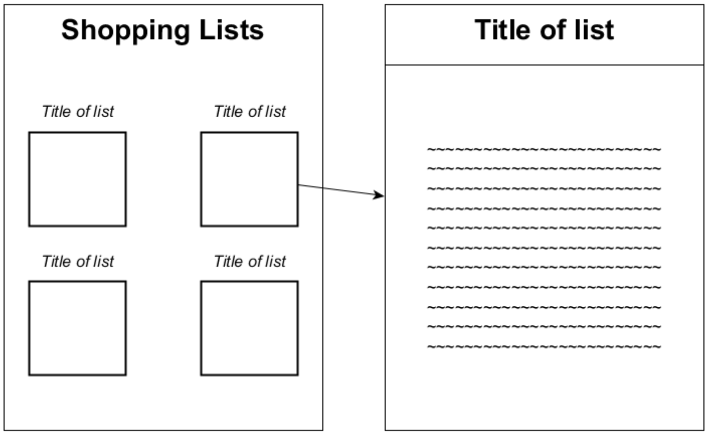
Summary	This feature must allow the user to save their favorite recipes for later use.
Rationale	If a user has a recipe they like, the system must allow them to save it so they can access it quickly in the future.
Requirements	Once a recipe is indicated to be saved, the system shall retrieve the recipe ID obtained from TheMealDB API. This ID will then be saved locally on the users device. The data saved will be organized initially by time, i.e. the most recent favorite will be at the top of the favorites list. However, the user may choose to reorganize this list once they have favored items. (*See FR-05*).
Reference(s)	UC-02, UC-05, FR-05, NF-04, NF-05

4.3 FR-02: Editing the ingredients list

Summary	This feature provides the user with a way to input ingredients into the system.
Rationale	In order for the search feature to function, the user must have a list of ingredients that can be used to filter recipes returned by FR-01.
Requirements	For the initial use of the system, the ingredients list will be empty. In this case, the system is to initialize the list and allow for user input. To add an item to the list, the user first inputs a name and a primary, required tag. The primary tags are as follows: Cold, Hot, and Room Temperature. These tags correspond to the temperature of the ingredient. This feature will avoid the need of multiple lists for pantry items, fridge items, etc. . .. The user may also indicate that they want to add an optional tag for their own organization. The system shall allow this and reference this optional tag (if it exists) for organizational purposes. The items in the ingredients list shall be organized according to Figure 1.  Figure 1: Organizing data **with** optional tag CONTINUED ON PAGE 11 ...

Requirements (cont.)	If the user does not provide an optional tag, then the data should be organized according to Figure 2.  Primary Tag Cold Ingredient Name Pork chops Ground beef Figure 2: Organizing data **without** optional tag Once all input is gathered from the user, the system shall add the new item to the ingredients list provided the input has passed error checking and validity checks. This ingredients list it to be saved by the system on the user's device. The user may wish to delete items from the ingredient list. When this action is indicated by the user, the system shall remove that ingredient from the saved list and update the list shown with any changes. **Error and Validity Checking** • The user should not be allowed to enter any special characters, i.e. ! @ # $ % & * ( ) ˆ / and any other special character available on the standard 65-key smart-phone keyboard. • The user should not be allowed to enter numerical data into any input fields, i.e. 1 2 3 4 5 etc . . .
Reference(s)	UC-03, NF-06, NF-07, NF-08 Figure 1, Figure 2

4.4 FR-04: Editing the shopping list

Summary	This system should allow the user to keep multiple shopping lists.
Rationale	If the user finds a recipe that they want to cook but are missing ingredients, they shall be able to create a list that can be used while shopping to keep track of ingredients needed.
Requirements	The shopping list should function similar to that of a notepad that can be found on most (if not all) computer devices. The user shall have the option to keep multiple lists. When the user creates a new shopping list, they shall have the option of giving it a title. The user then inputs data into the shopping list. This data (and all shopping lists) are saved to the user's device.  Shopping Lists Title of list Title of list Title of list Title of list Title of list Figure 3: Organizing the shopping lists If the user indicates that a list is to be deleted, the system should remove the list from the user's view and delete any data associated with the list. The user should have the option of deleting specific items from within a list. In this case, the system should remove that item from view. **Error and Validity Checking** Because this list is only for the user's reference and no data will be parsed from it, special characters and numerical data should be allowed.
Reference(s)	UC-04, Figure 3

4.5 FR-05: Viewing favorite recipes

Summary	This function allows the viewing of recipes that have been favorited.
Rationale	The user should have a way to view the recipes that have been saved on their device.
Requirements	When the user indicates that they wish to view their saved recipes, the system should retrieve the data that is saved in FR-02. The system shall then display this data to the user. The data will be initially organized by time, i.e. the most recent favorite is at the top of the list. However, the user shall have the option to organize the data their own way. The user may indicate that an item is to be un-favorited. The system shall remove the item from the favorites and update changes to the list. The user may indicate they wish to view more details about a specific recipe. The system shall use the recipe ID that is retrieved from FR-02 to access recipe data associated with the ID. These details are then displayed to the user.
Reference(s)	UC-02, UC-05, FR-02, NF-09, NF-10

5 Non-Functional requirements

In this section, specific measurements are given that the software must meet. The metrics considered are as follows:

- Availability - the amount of time that the system is expected to be operational
- Efficiency - how well does the system utilize resources on the device
- Performance - how long does it take to perform a function within the system; timing constraints.
- Robustness - how well can the system handle error conditions, i.e. invalid user input
- Usability - how easy is the system to use for the intended target base.

5.1 NF-01: Availability of the Search function

Summary	The system should allow searching recipes almost all the time.
Rationale	This is the core functionality of the application and thus should be available.
Requirements	The system should expect 99.9% of uptime on TheMealDB API. This means that the search function of the system should be functional and available for 99.9% of the time.
Reference(s)	UC-01, FR-01

5.2 NF-02: Performance of the Search function

Summary	Searching recipes should be done quickly.
Rationale	If the search is not done quickly, users may not want to use the system.
Requirements	A search for recipes using 20 ingredients on TheMealDB must take 2 seconds or less to perform on an Android phones running Android 5.0 Lollipop.
Reference(s)	UC-01, FR-01

5.3 NF-03: Usability of the Search function

Summary	Searching recipes should be easy and user friendly.
Rationale	If the search function is over complicated and cumbersome, users may not want to use the system.
Requirements	The search must be easy for any normal user to understand and use. The system must aim to be as user friendly as possible.
Reference(s)	UC-01, FR-01

5.4 NF-04: Efficiency of the Save function

Summary	Recipes saved should take up the least amount of space as possible.
Rationale	The system should not take up too many resources on the user's device or their device will quickly run out of memory and storage space.
Requirements	The system must aim to have each recipe saved take up no more than 512KB (kilo-bytes).
Reference(s)	UC-02, FR-02

5.5 NF-05: Performance of the Save function

Summary	Saving recipes should be fairly quick.
Rationale	If the user has to wait too long after saving recipes, they may not want to use this function.
Requirements	The system must aim to have each recipe saved take no more than 1.5 seconds to complete the save.
Reference(s)	UC-02, FR-02

5.6 NF-06: Efficiency of the Edit Ingredients function

Summary	The ingredients list should take up the least amount of space as possible.
Rationale	The system should not take up too much space on the user's device. If this happens, the user may delete the application.
Requirements	The ingredients list on the user's device must be 20MB (mega-bytes) in size or under.
Reference(s)	UC-03, FR-03

5.7 NF-07: Robustness of the Edit Ingredients function

Summary	The system should be able to handle invalid user input.
Rationale	To prevent any issues with the search functionality, the user input to the system should be limited.
Requirements	As stated in FR-03, the system must check and validate user input so that only alpha characters are accepted.
Reference(s)	UC-03, FR-03

5.8 NF-08: Performance of the Edit Ingredients function

Summary	The system should save the ingredients list quickly.
Rationale	If the system takes too long to save ingredients, this may deter users from inputting any ingredients to the list and in turn prevent them from using the search functionality.
Requirements	The list must be saved by the system in under 1.5 seconds.
Reference(s)	UC-03, FR-03

5.9 NF-09: Robustness of the View Saved Recipes function

Summary	This function should be available with or without Internet access.
Rationale	The user may not have access to an Internet connection all the time. The system must still allow the viewing of saved recipes.
Requirements	The system will provide an alternate option if the device does not have Internet connection. The data will be cached when the recipe is saved so it may be accessed without Internet.
Reference(s)	UC-05, FR-05, NF-04

5.10 NF-10: Performance of the View Saved Recipes function

Summary	The system will have different performance constraints depending on the user's connection to Internet.
Rationale	The function to view saved recipes should be quick or the user may not want to use this function.
Requirements	• If the user has Internet access, the system must not take longer than 2 seconds to access saved recipes. • If the user does not have Internet access, the system must take no more than 0.5 seconds.
Reference(s)	UC-05, FR-05, NF-04