SEG2105A - Fall 2016
Final Report

Cook Helper

Team: QWERTY123

Team members:
David Delisle-Lalancette: 7822733
Kenneth Lum : 8174815
NamChi Nguyen : 7236760
Raphael Okoh: 7156141
Chibike Echendu: 7381031

Professor:
Miguel Garzon

Submission date:
Dec 7, 2016

Table of Contents

Introduction

The purpose of this report is to give an overview of the process involved in the development of CookHelper; a recipe managing, android mobile application. The development process is demonstrated through a series of deliverables which contain functional requirements, non-functional requirement, use cases, UML class diagrams, UML sequence diagrams, and UML state diagrams pertaining to CookHelper. There are also screenshots of the UI in the document displaying all of the application's functionality.

Scope

The CookHelper application will perform the following tasks:

- Recipes will be stored on the local device.
- The user can view stored recipes in full detail or in list form.
- Stored recipes can be edited or deleted by the user.
- New recipes can be added by the user.
- The user can perform a search, filtering the recipes based on the following fields: recipe name, ingredients, cultural-category, meal-type and maximum preparation time.

Contribution and Corrections

Table 1. Team member contribution

Member	Deliverable 1	Deliverable 2	Deliverable 3	Deliverable 4
David Delisle-Lalancette	75	40	45	45
Kenneth Lum	5	20	25	15
NamChi Nguyen	5	20	25	15
Raphael Okoh	5	0	5	15
Chibike Echendu	10	20	0	10

Table 2. Changes/Correction

Deliverable	Changes/Corrections
1	Removed sorting and added saving and navigating to the functional requirements.
2	● Fixed multiplicities of the RecipeBook singleton class ● Converted the domain model into a system model ● Updated the sequence diagrams to correspond to the system model ● Corrected issues with the state machine diagram
3	Added the missing UI screenshot for deleting a recipe
4	No changes were made

Software Requirements (Deliverable 1)

This section contains functional and non-functional requirements.

Functional Requirements (20)

1. Cook helper shall allow the user to view the list of recipes.
2. Cook helper shall allow the user to view a detailed representation of a recipe.
3. Cook helper shall allow the user to add a recipe to the list.
4. Cook helper shall allow the user to remove a recipe from the list.
5. Cook helper shall allow the user to edit an existing recipe.
6. Cook helper shall allow the user to set the *name* of a recipe while adding or editing a recipe.
7. Cook helper shall allow the user to specify the cultural *category* of a recipe while adding or editing a recipe.
8. Cook helper shall allow the user to specify the meal *type* of a recipe while adding or editing a recipe.
9. Cook helper shall allow the user to specify the preparation time of a recipe while adding or editing a recipe.
10. Cook helper shall allow the user to specify all *ingredients* used in a recipe while adding or editing a recipe.
11. Cook helper shall allow the user to specify all instructions in a recipe while adding or editing a recipe.
12. Cook helper shall allow the user to search for a recipe by *name*.
13. Cook helper shall allow the user to search for a recipe by *category*.
14. Cook helper shall allow the user to search for a recipe by *type*.
15. Cook helper shall allow the user to search for a recipe by *ingredient*.
16. Cook helper shall allow the user to search for a recipe by preparation time, by setting a maximum time.
17. Cook helper shall allow the user to search for a recipe with more than one of the aforementioned filters.
18. Cook helper shall allow the user to access a comprehensive set of easy to follow instructions to use the application.
19. Cook helper shall allow the user to navigate between the main activity, advanced search and the help page through a navigation drawer.
20. Cook helper shall allow the user to permanently save recipes.

Non-Functional Requirements (5)

1. Cook helper shall operate on the Android Platform.
2. Cook helper shall persist recipes using JSON.
3. Cook helper shall be designed according to the Model-View-Controller design pattern.
4. Cook helper shall show the list of recipes in less than 5 seconds.
5. Cook helper shall have no more than 4 activities on the activity stack.

UML Design (Deliverable 2)

This section contains three use-cases (from deliverable 1 but moved to the UML section), a system model, three sequence diagrams and a state machine diagram.

Use-cases

Adding a recipe

Name: Adding a recipe
Actors: User
Goals: To add a recipe to the list of recipes
Preconditions:
Summary: The user will prompt the system to add a recipe, then they will fill out required and optional fields. The recipe will be saved with the rest of the recipes
Related use cases: Editing a recipe

Steps	
User actions	System responses
Clicks “Add Recipe”	Opens a new activity for recipe editing.
User provides the following information: recipe name, cultural category, meal type, a list of relevant ingredients, a list of relevant steps. The user clicks: “Save recipe”	The system validates the information and creates a recipe object and adds it to the list of recipes. The lookup tables of ingredients, categories and types are updated with a reference to the recipe. The list is saved (to local storage).

Postconditions: The newly created recipe is saved in the list.

Searching for a recipe

Name: Advanced search for a recipe
Actors: User
Goals: To find all recipes that correspond to the search query.
Preconditions: The user opens the Advanced Search activity.
Summary: The search system must parse the request and filter the list according to the parsed request. The application will then show the results to the user, which may be one or many recipes, or no recipes at all.
Related use cases:

Steps	
User actions	System responses
From the main activity, the user clicks “Advanced Search”.	Opens the Advanced Search activity.
User enters their query, a combination of name, type, category, and ingredients with binary operators. Click “Search”	The search system parses the query. Looks through the list of recipes and selects those that match the query. Opens a new list-view activity with the result on display. If there are no results, indicate that to the user.

Postconditions:

Deleting a recipe

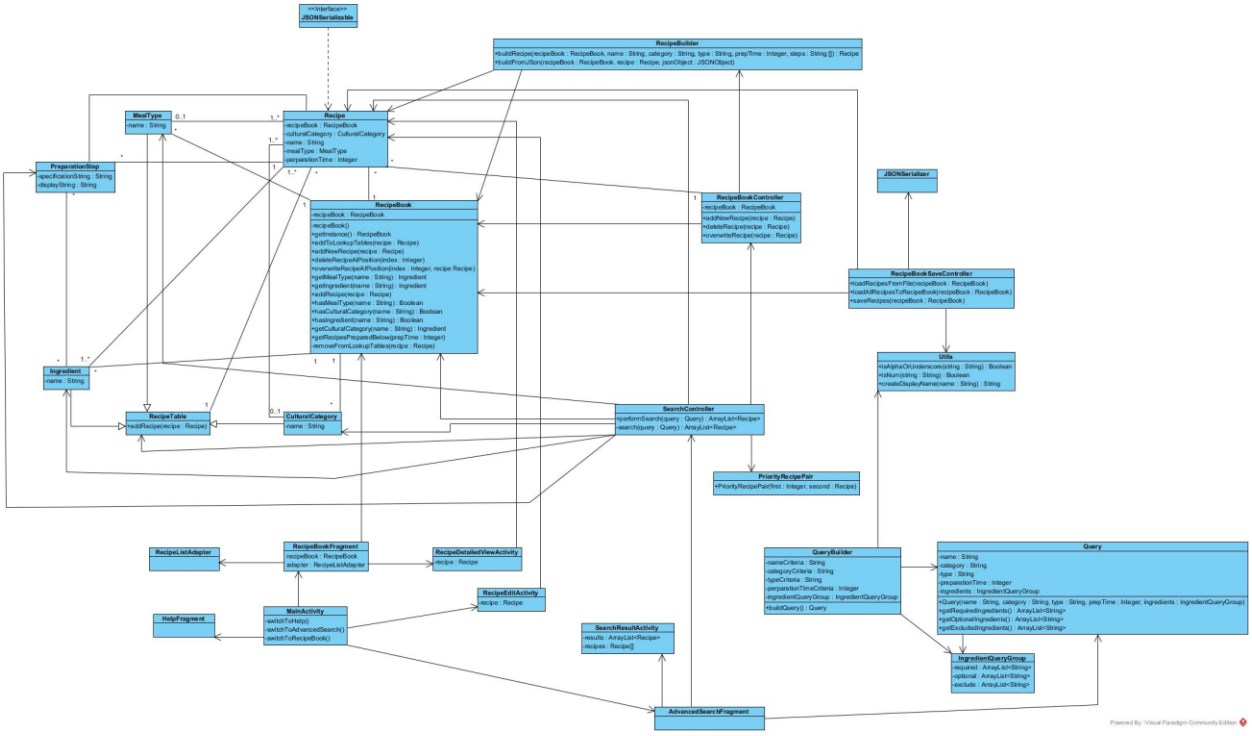
Name: Deleting a recipe
Actors: User
Goals: To remove a recipe from the list of recipes
Preconditions: Displays the list of all recipes
Summary: The user will prompt the system to remove an already saved recipe, the system will ask for a confirmation. The recipe will be removed from the list recipes
Related use cases: Searching for a recipe.

Steps	
User actions	System responses
Long-clicks the recipe to be deleted.	Opens a context menu/action bar offering to delete.
Clicks “Delete Recipe”	Prompts the user for a confirmation.
Clicks “Delete”	Removes the recipe from the list of recipes. Removes all references to that recipe from the lookup tables of ingredient, categories and types.

Postconditions: Displays the list of recipes without the deleted recipe.

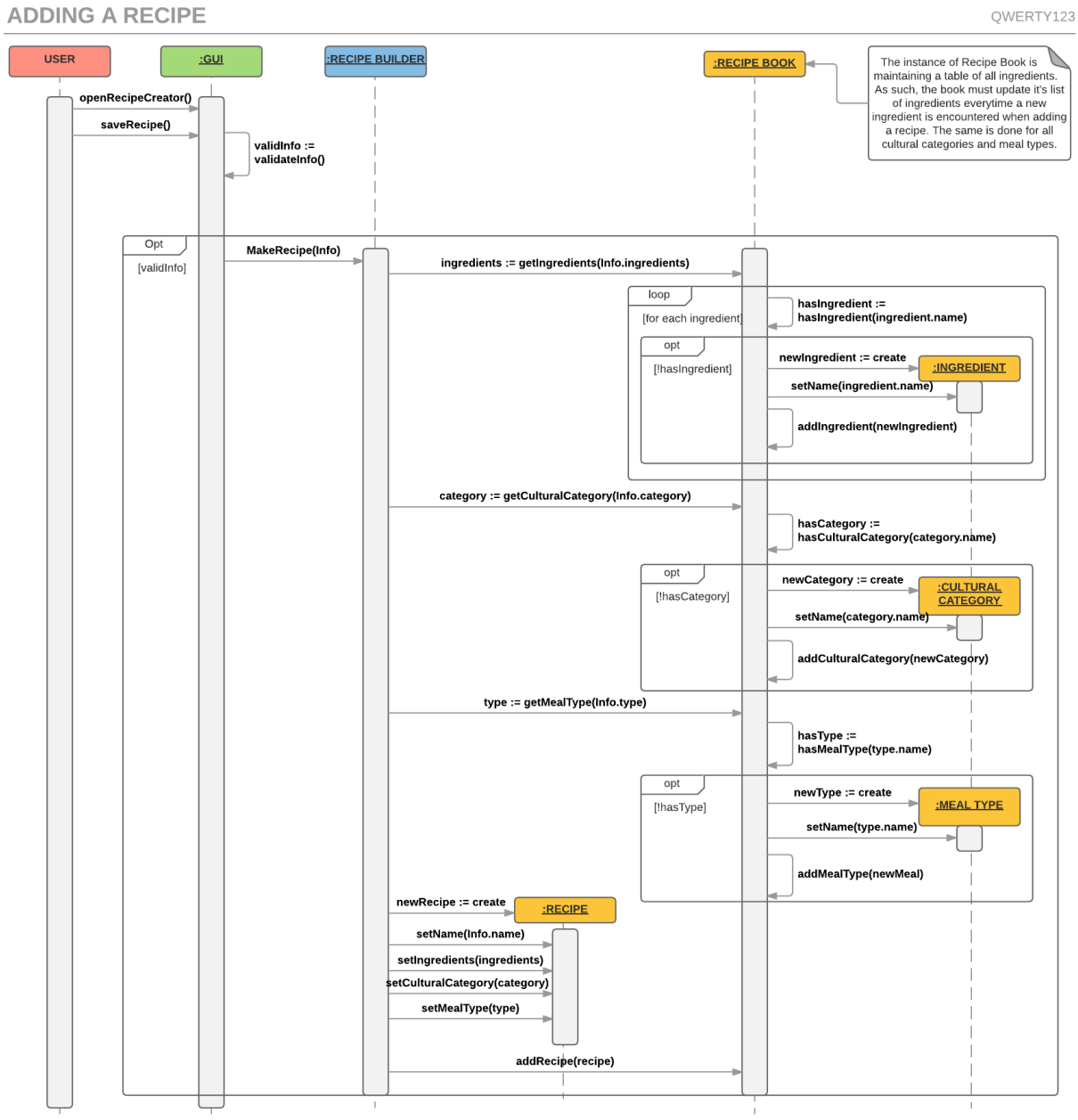
System Model

The domain model has been expanded into a system model which includes the UI and utility classes. The domain model consists of Recipes, Ingredients, Cultural Categories, Meal Types, and Preparation Steps. The Recipe Book is the singleton for this system.



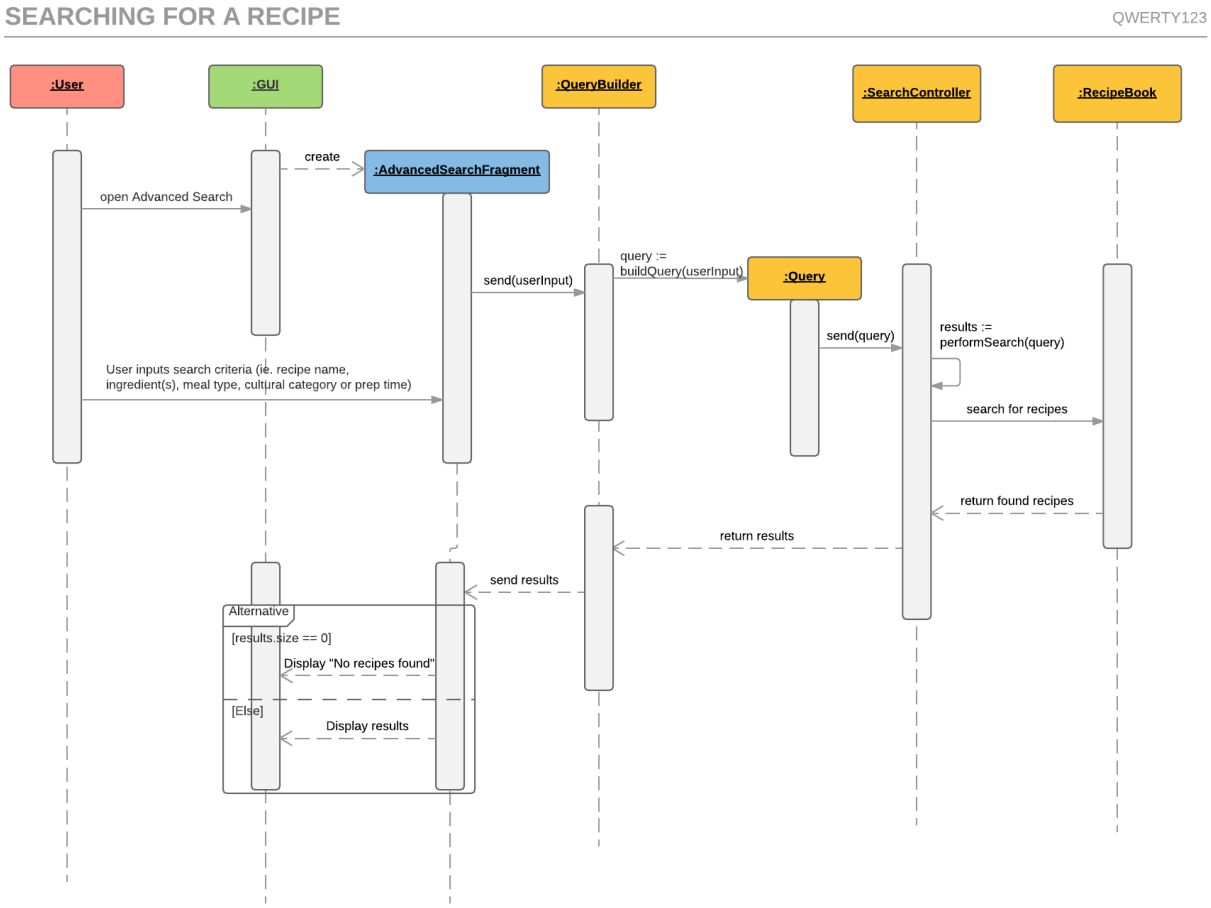
Adding a Recipe Sequence Diagram

This sequence diagram shows the process of adding a recipe to the recipe book, which also involves the updating of the lookup tables (see note).



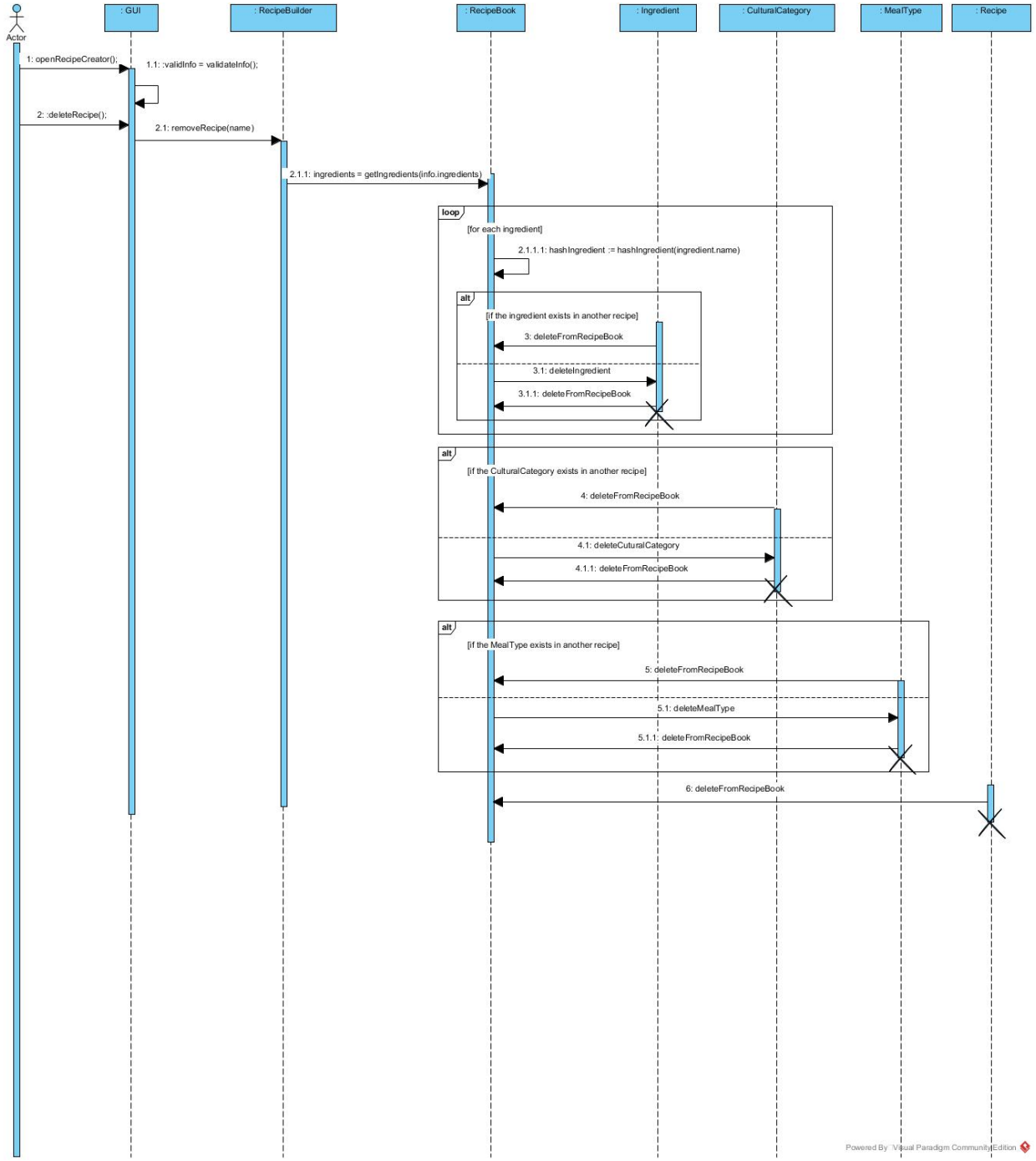
Searching for Recipe Sequence Diagram

This sequence diagram shows the process of performing a search, which involves packaging the user's request in a query object created by the QueryBuilder, which is processed by the SearchController, which retrieves a list of relevant recipes from RecipeBook.



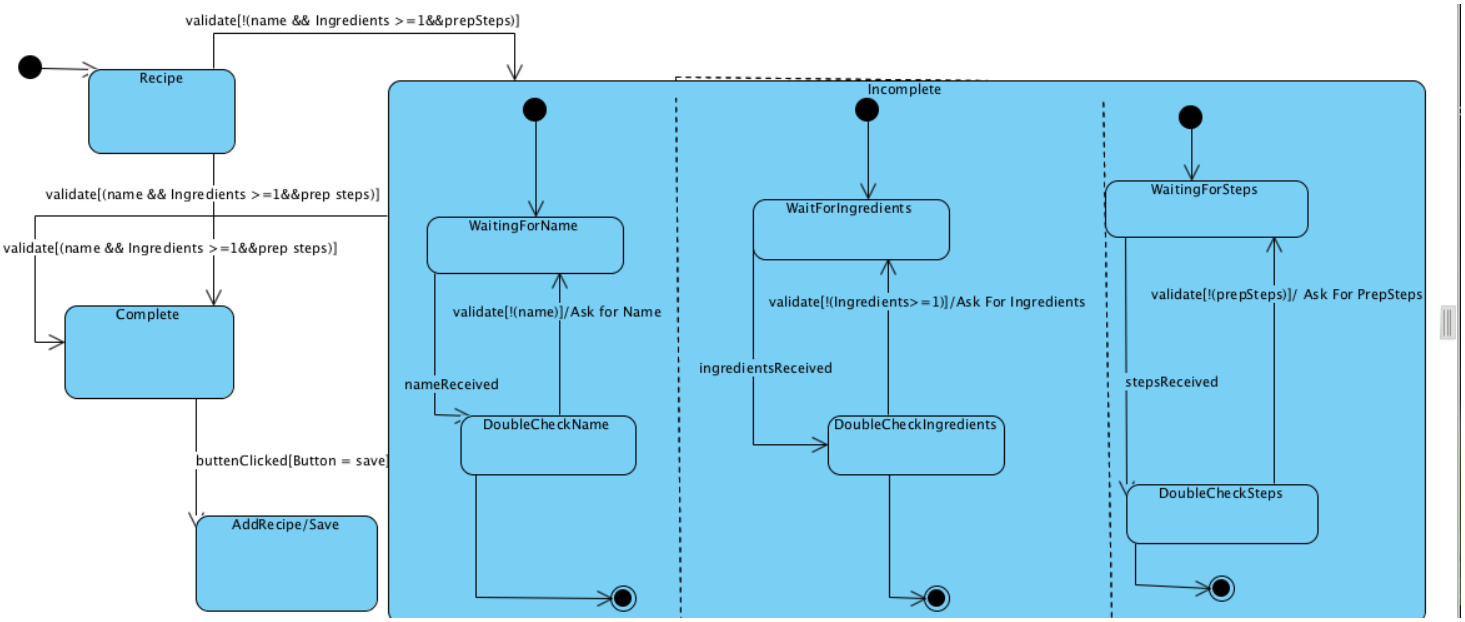
Deleting a Recipe Sequence Diagram

This sequence diagram shows the process of deleting a recipe from the recipe book. The lookup tables are also updated if the recipe was the last one pointed to by an entry in the lookup tables. The entry is removed.



RecipeBuilder State Machine Diagram

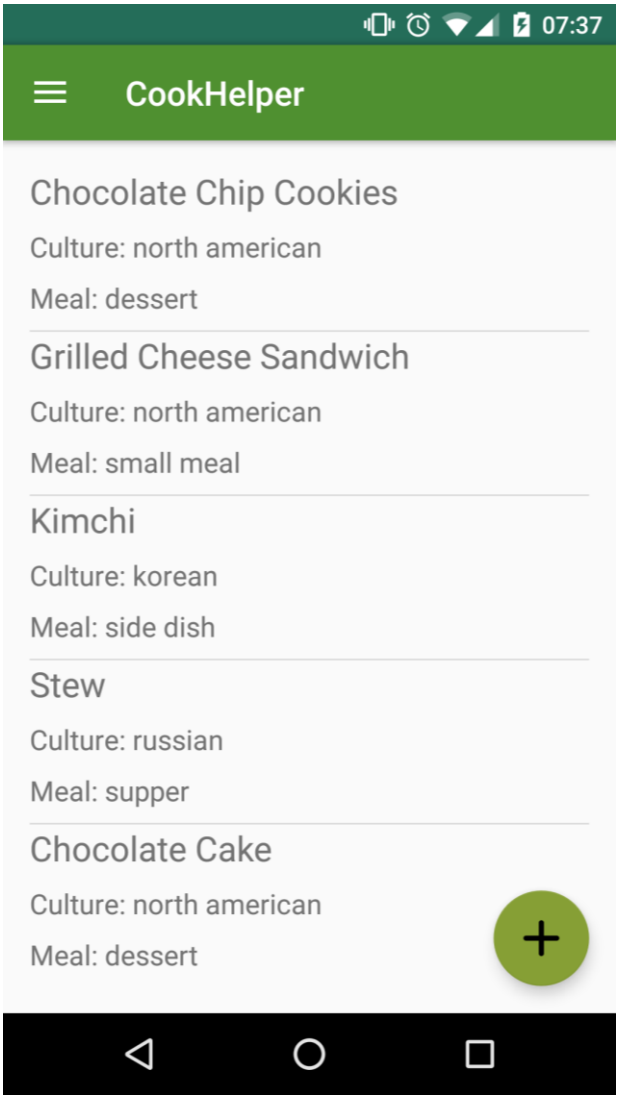
This diagram shows the validation process of the RecipeBuilder class, which is a part of our system model. This class receives the input from the GUI, ensures it is valid, and creates a recipe. It concurrently validates the name, ingredients and preparation steps.



Screenshots of the UI (Deliverable 3)

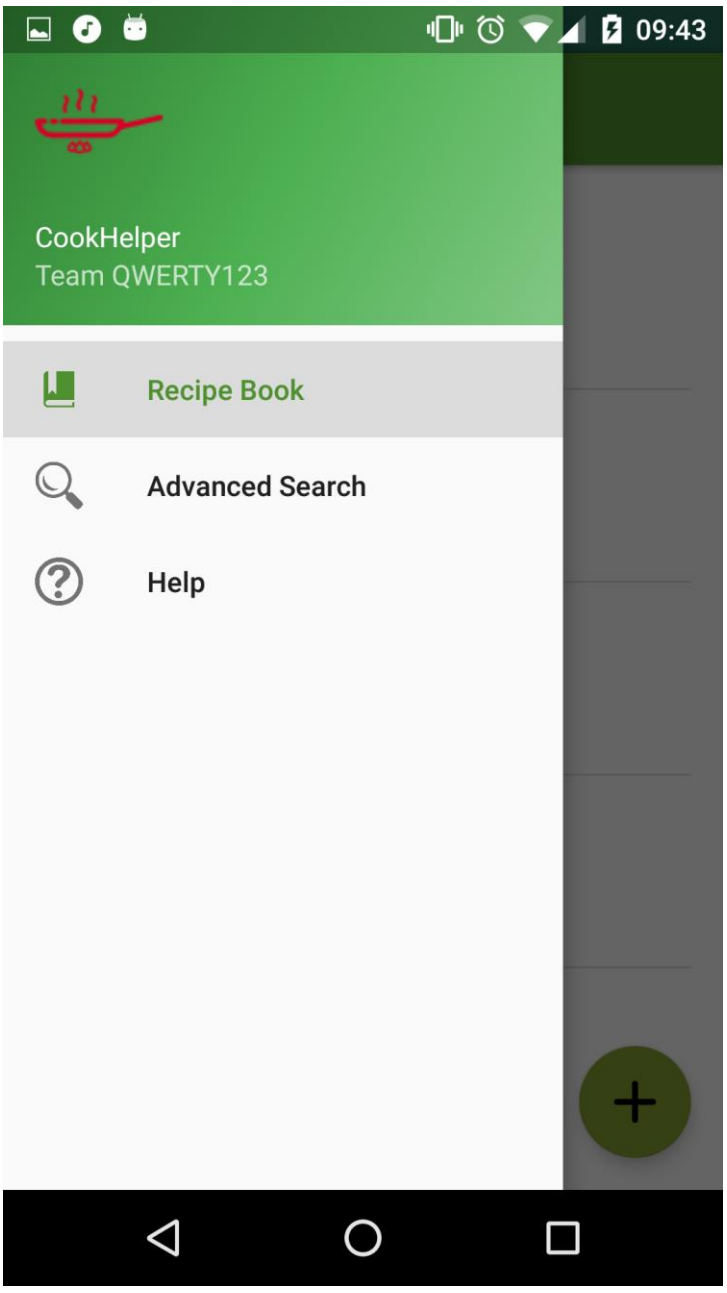
Main activity: Recipe Book

The Recipe Book allows the user to view the list recipes. Clicking one of the recipes will lead the user to another activity for viewing the recipe's details. The floating action button, available from all screens in the navigation drawer, allows the user to add a recipe to the recipe book.



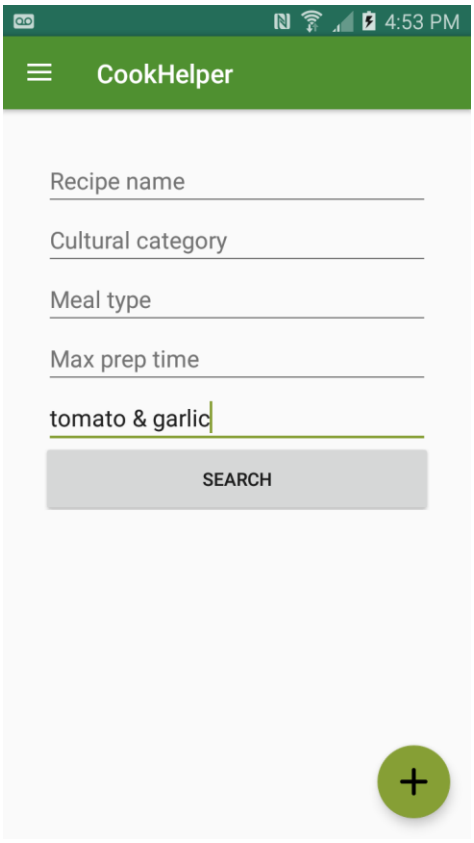
Main activity: Navigation Drawer

The navigation drawer allows the user to change between the Recipe Book, Advanced Search and help screens.

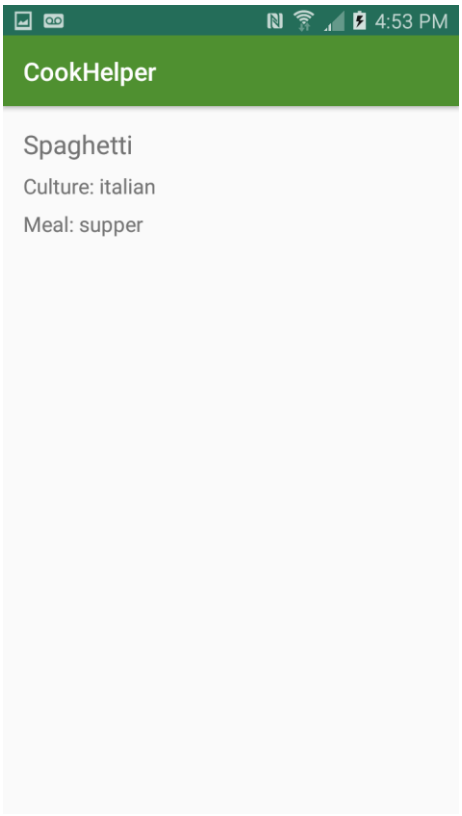


Advanced Search

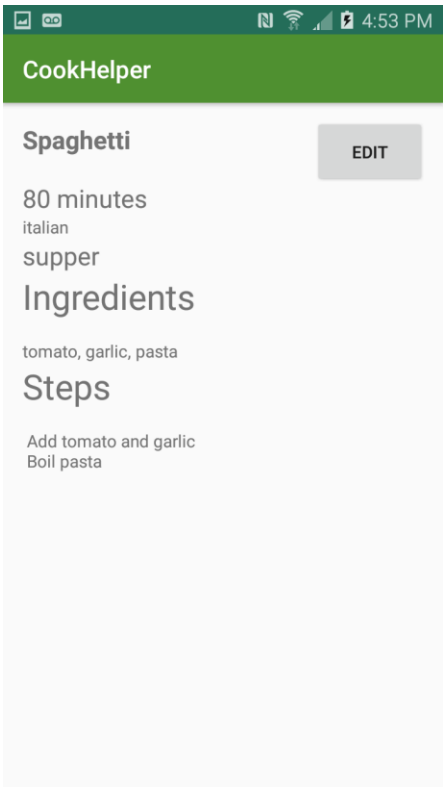
The Advanced Search allows the user to specify some parameters for the search, including a name, the max preparation time, the ingredients, the meal type, and the cultural category.



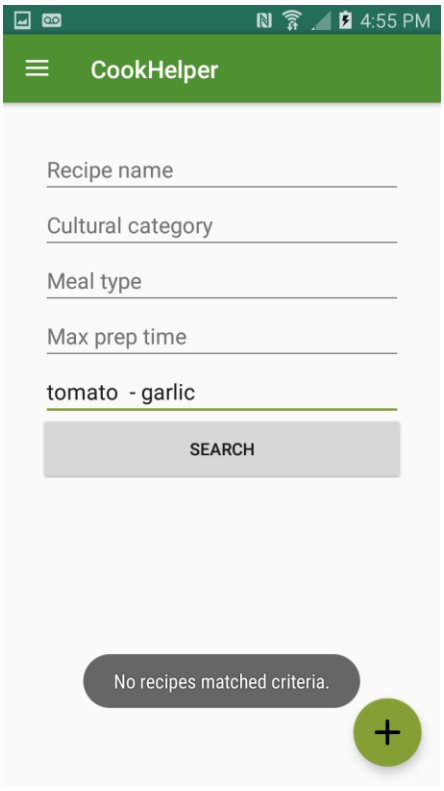
Search criteria (ie. tomato and garlic)



Results



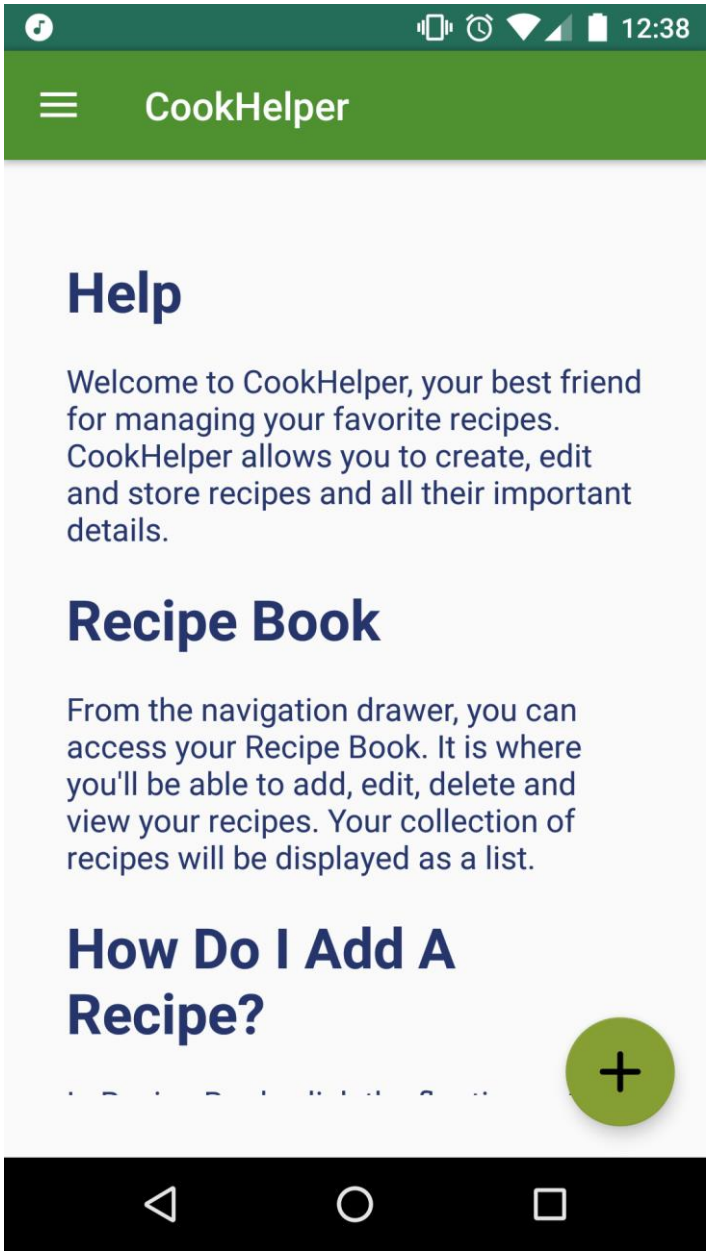
Recipe view



No recipes found (ie. tomato and not garlic)

Help

The Help screen is an introduction and FAQ about the app.

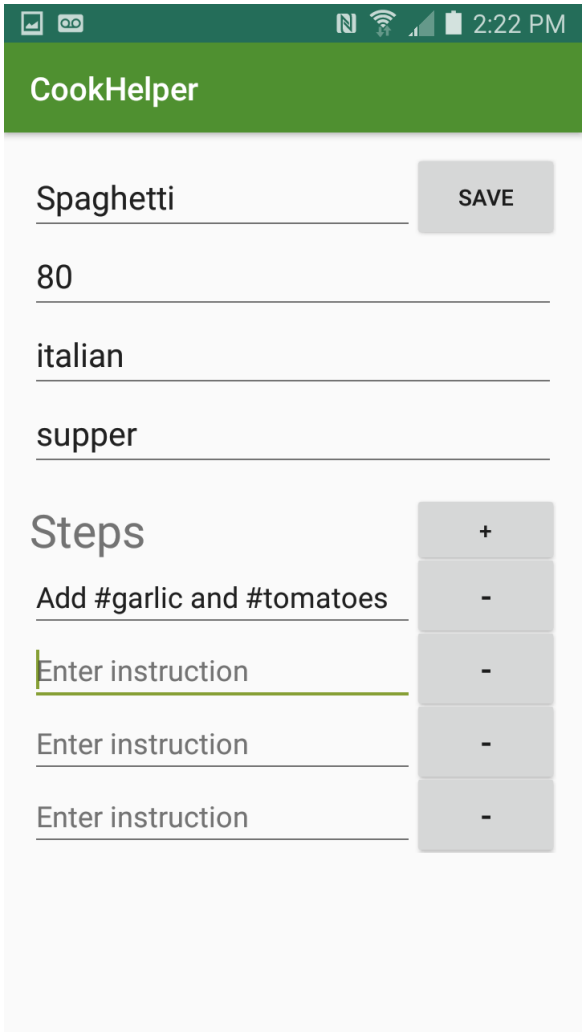


Detailed Recipe View

This activity allows the user to view the details of a recipe, including the name, category, type, preparation time, ingredients and steps.

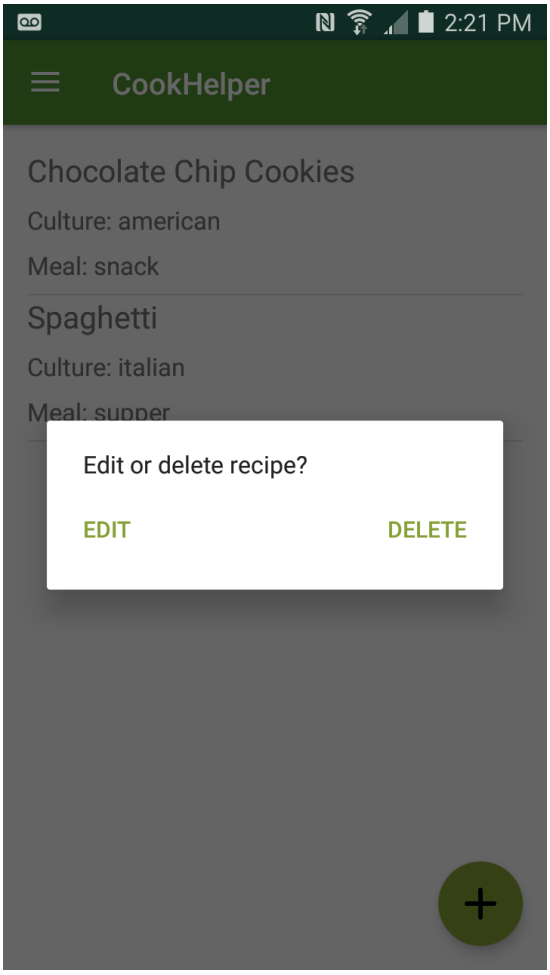
Editing A Recipe

The Edit Recipe activity allows the user to edit an existing recipe or create a new recipe and specify the name, preparation time, category, meal type and add a number of steps which include the ingredients used.



Deleting A Recipe

The user is able to delete a recipe by long-clicking it from the list in Recipe Book.



Note(s)

All images were taken from the internet. All credit goes to their respective creators. They are there simply to replace the icons placed by Android Studio during the generation of the navigation drawer and floating action button.