Recipe Book

Design Document
Version (0.1.0)

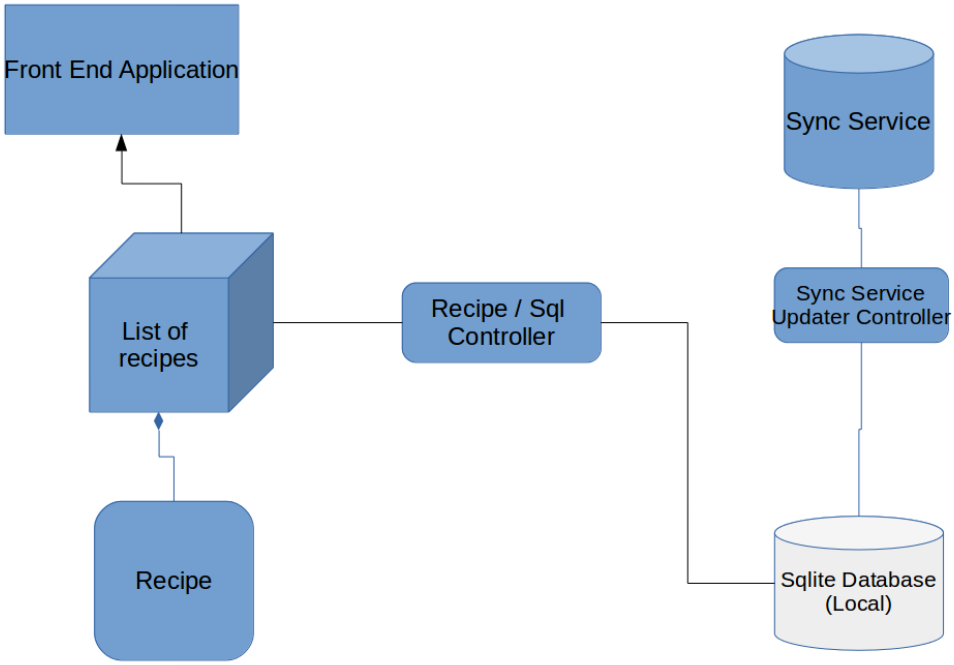
Carter Hay
Mariah Wright

I. Purpose

This Android app will allow for the storage and categorization of recipes through a simple, easy-to-use interface. In many current recipe storage apps, it is difficult for users to store their own recipes, because they are often partnered with some third party recipe website. One problem many people have when cooking for multiple people is adjusting measurements on the original recipe in order to cook a different number of servings. This app will insure it is easy to convert both the number of servings and ingredient measurements without having to leave the app.

This app will be targeted toward those individuals who may have a large collection of hand written or poorly organized recipes. Almost everyone who cooks has an example of a recipe that has been handed down from a family member and hastily written on a note pad. This app will allow users to transfer these types of recipes into one organized digital environment that is as easy to use as a notebook.

II. High Level Design



a) **Front End Application**

The front end application is what the users will see. This will allow the users to read, edit, and create their recipes.

b) **List of recipes**

This is a list object which will act as an abstraction layer between the front end and database controller. The front end will be able to interact with this list and any additions deletions or modifications to this list will also be handled in the database.

c) **Recipe**

This recipe object is what will be stored in the list of recipes. It will allow for the storage of recipe data and will perform conversions on measurements.

d) **Recipe / Sql Controller**

This object will interact directly with the database performing queries.

e) **Sqlite Database**

The Sqlite database is the local database in which recipes are stored on the user's device.

f) **Sync Service Updater Controller**

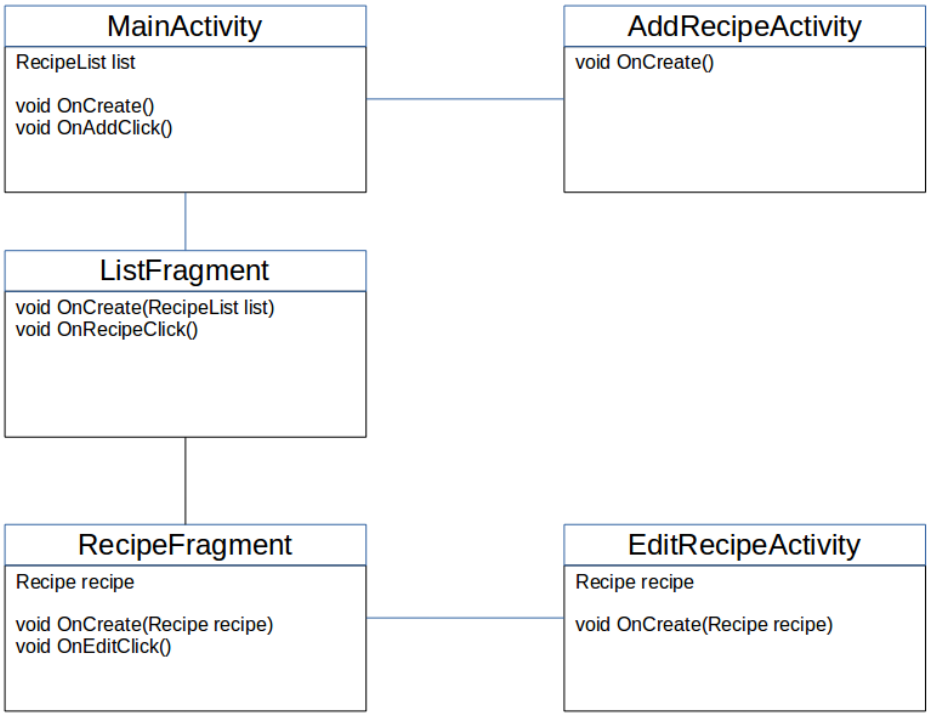
The sync service updater controller will run on a separate thread which will be responsible for keeping the local database and remote sync service in sync. It will track changes between the local database and the remote system.

g) **Sync Service**

The details of the sync service have yet to be determined, but it will hopefully enable the syncing of recipes between user's devices.

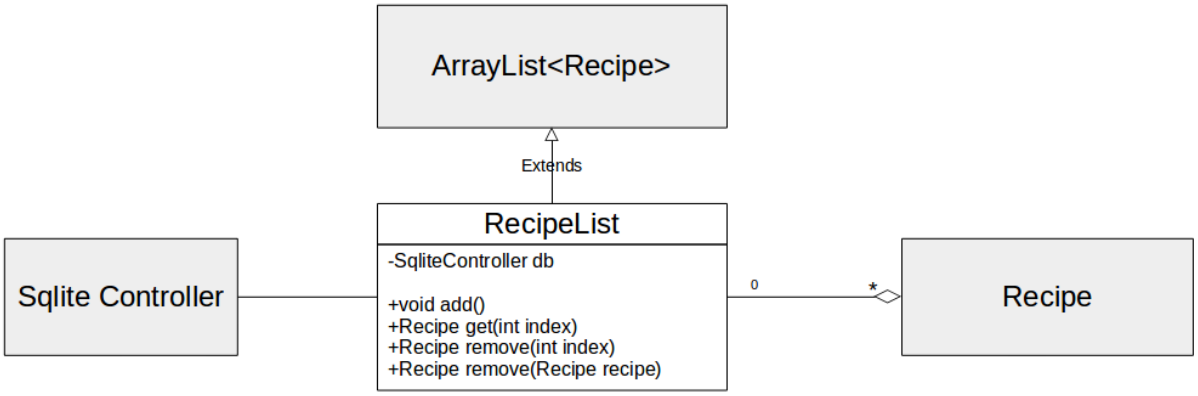
III. Low Level Design

a) **Front End Application**



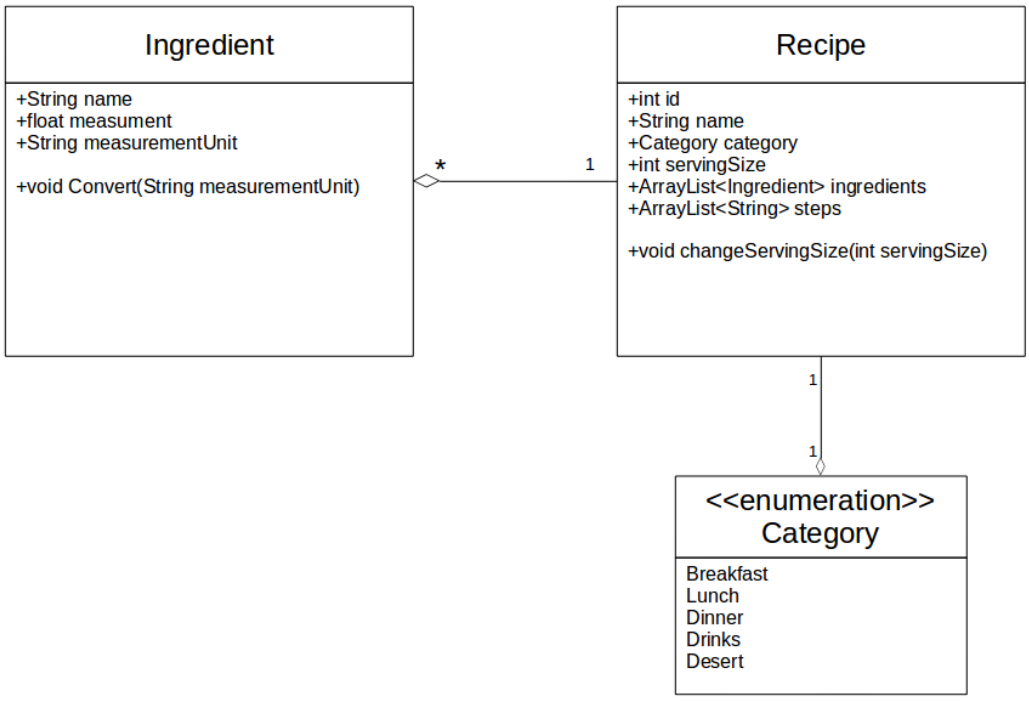
The front end application will consist of a MainActivity, ListFragment, RecipeFragment, AddRecipeActivity, and EditRecipeActivity. The MainActivity will create the ListFragment and RecipeFragment. The ListFragment will be created with a list of recipes passed to it by the MainActivity once a user selects a recipe category. The ListFragment will then allow the user to select a recipe. Once selected, the ListFragment will start the RecipeFragment with the recipe the user selected. If the user chooses to edit a recipe being viewed, the RecipeFragment will pass the current recipe to the EditRecipeActivity and start it. The MainActivity will also contain a button which will enable the user to add a new recipe, this button will start the AddRecipeActivity.

b) List of recipes



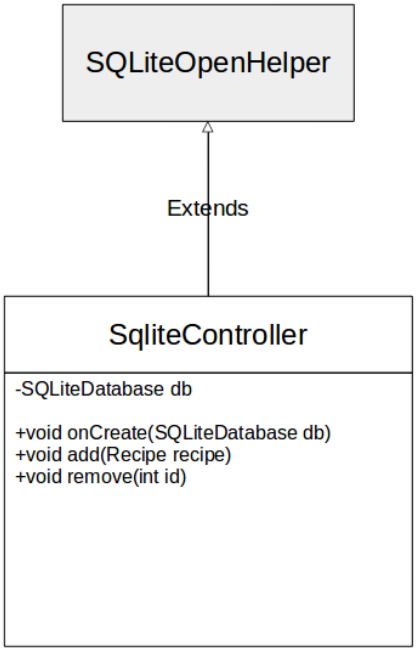
The RecipeList extends java's ArrayList and will be responsible for storing lists of Recipes and abstracting database actions. When the front end application layer adds a recipe to the RecipeList, the RecipeList will first add that recipe to the database using a SqliteController object, and it will then add the recipe to itself. The same is true for removing an object from the list. By extending ArrayList, the RecipeList gains the extra functionality that an ArrayList has, such as being iterable and certain lambda abilities introduced in Java 8.

c) Recipe



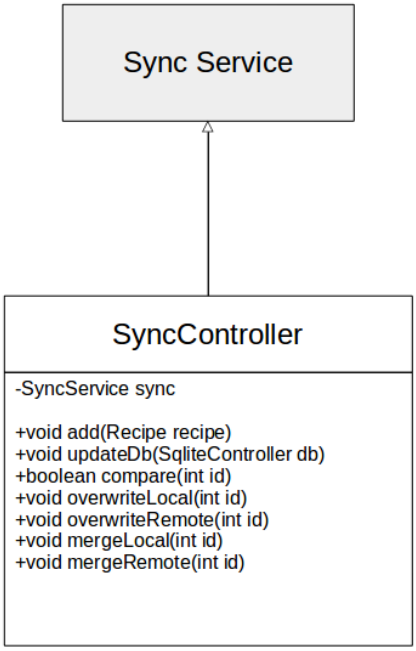
The Recipe is the model which represents a real life recipe. A recipe consists of a name, category, serving size, list of steps, and list of ingredients. The recipe object is also responsible for conversions of ingredient quantities for changes in serving size. The category of a recipe is represented by a category enumeration which contains each meal of the day. Ingredient objects are stored in the Recipe object ingredients ArrayList. Ingredients consist of a name, measurement, and measurement unit. Ingredients also contain a method that allows for conversion to different measurement units. Because Ingredients are contained in an object, we can easily iterate over each object in the list and convert it.

d) Recipe / Sql Controller



The SqliteController is an abstraction of raw sql queries to the application's Sqlite database. It provides a method to add a recipe to the database by accepting a recipe as an argument. It can also remove a recipe from the database with a certain id. The onCreate method creates a new database for the application if one does not already exist.

e) Sync Service Updater Controller



The SyncController is an object that keeps the remote sync service and the local database in sync. By allowing syncing to copy the remote database onto the local database, the application can still be used when off-line, and will use less mobile data, because it does not need to constantly access a remote server. The SyncController will provide methods which allow for the management of the remote and local data, such as adding a recipe to the sync service, updating the local database, comparing a remote and local recipe, overwriting a local recipe, overwriting a remote recipe, and merging local and remote recipes.

IV. Benefits / Assumptions / Risks

a) Benefits

1. By separating each layer of the application, it should be easy to re-implement any part independently.
2. By allowing the recipe list to handle database actions, it is an easy abstraction for any developers who may not be familiar with databases
3. Using simple Add and Remove methods in the database controller should allow a network component to be a drop-in replacement if it the app ever becomes a fully server enabled app.

b) Assumptions

1. The biggest assumption made during the design of this app is that there is an appropriate sync solution that exists. This is the least researched area of this document as it is not required for a viable product.

c) Risks

1. Because every recipe is loaded into the app at its start, users with thousands of recipes will have severe performance issues.
2. Editing a recipe can only be implemented by removing and re-adding a recipe with the current design. If the app were to crash between these two steps, users would lose data.