Design Document

for

RecipeBuddy

Version 1.0

Prepared by Michael Prevost, Michael Pierce, Taylor Manley, Randeep Singh

PiercePrevostSinghManley

11/14/2021

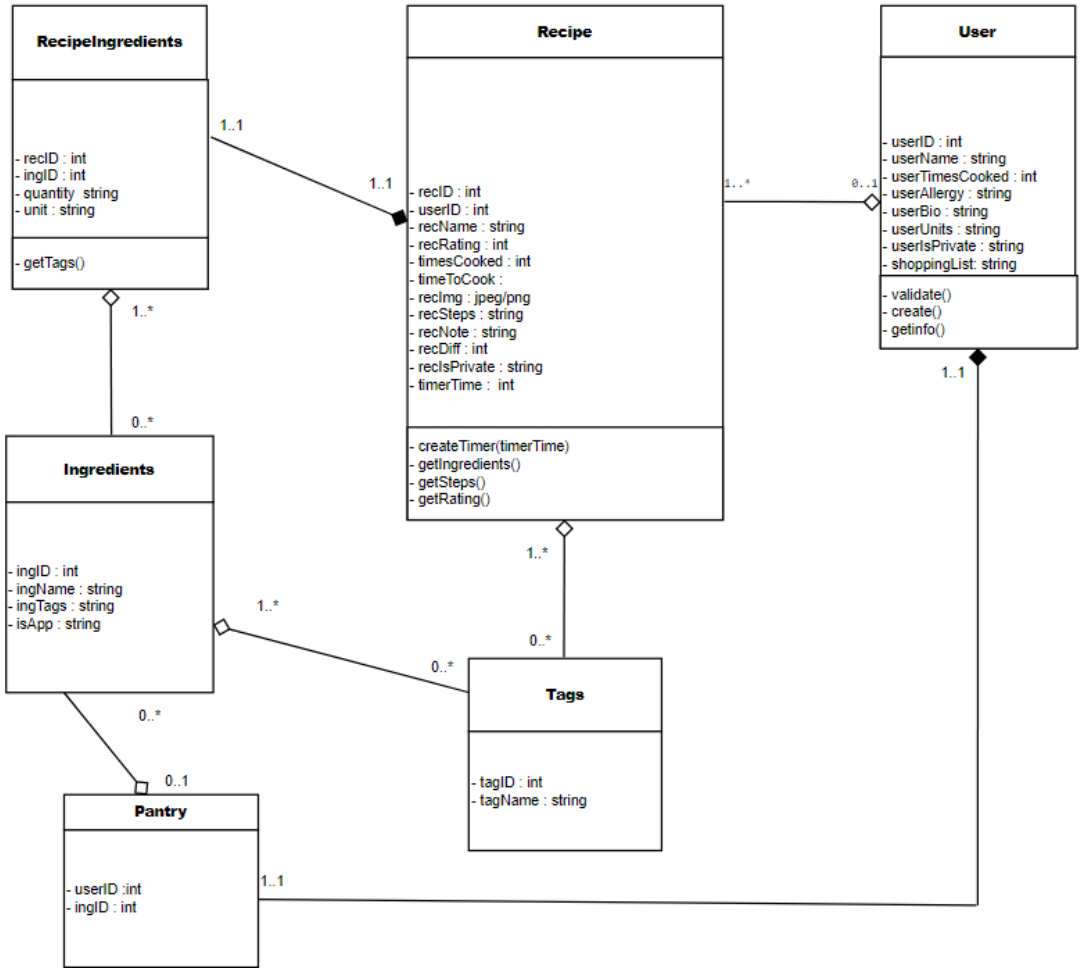

Figure 1: Class Diagram

Figure 1 illustrates the current class diagram of our system. Currently, we have been focusing on 6 main classes: *Recipe, User, Pantry, Ingredients, RecipeIngredients,* and *Tags*, each with their own set of variables and methods. There are mainly aggregation arrows connecting most of the classes, as most of the time can survive without the other, the only exceptions being a user's pantry not existing without a user, and a RecipeIngredients object should not exist if the recipe it is associated with is deleted.

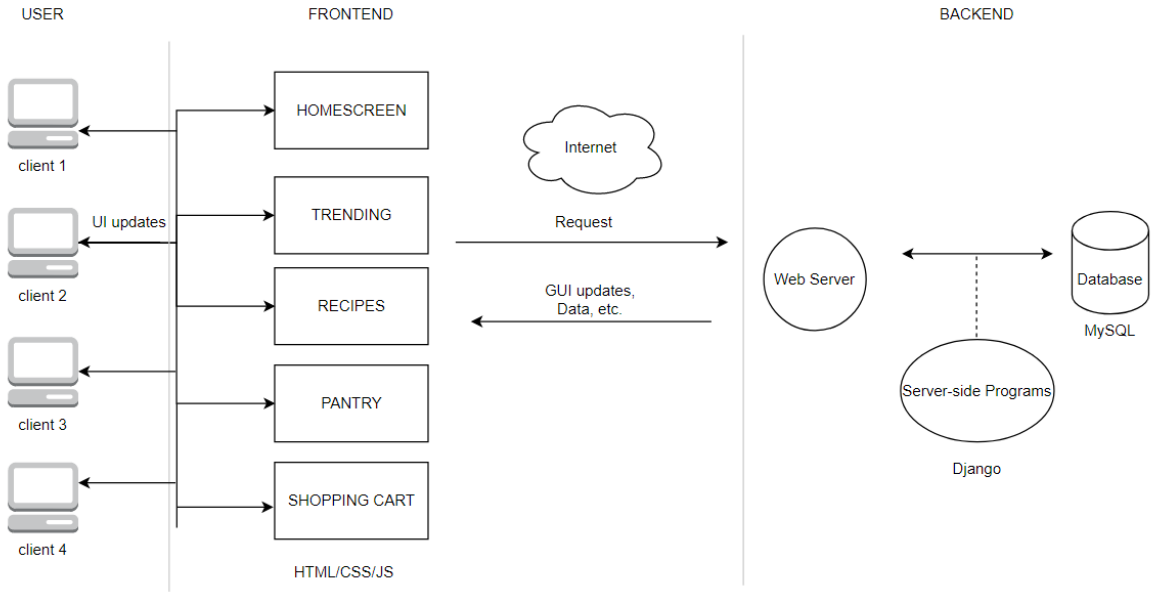

Figure 2: Architectural view of RecipeBuddy

Figure 2 shows the architectural view of RecipeBuddy. Users will be able to access the different functionalities through the front-end of the website including a trending page, personalized profiles, different recipes, and more. Data for all of these functions will be stored in a MySQL database and talk to the website using Django. Django will be what handles all communication and transferring of data between the database and the website.

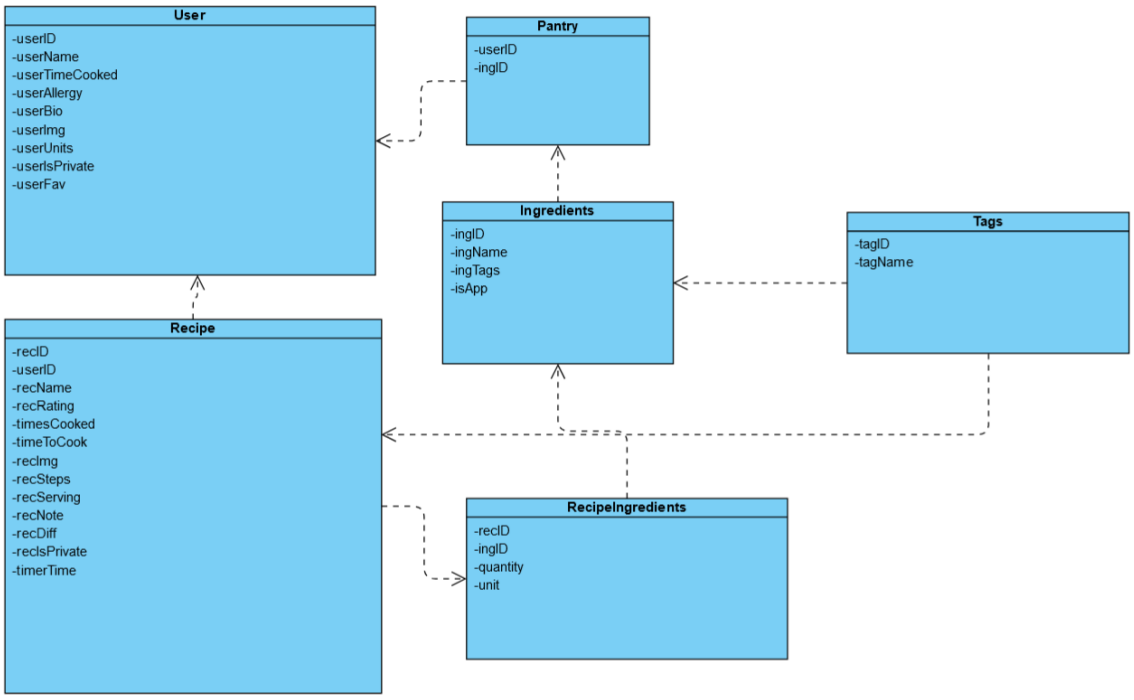

Figure 3: Database Diagram

Figure 3 is our complete database diagram as intended to be laid out. The diagram showcases the information stored within each database and how they interact with each other. The dashed arrows between the different databases showcase dependency. For example, the *RecipeIngredients* database is reliant on the ingID from the *Ingredients* database and the recID from the *Recipe* database. The *User* and *Recipe* databases are the largest databases we will have in terms of information stored. While the *Ingredients* database will have the largest quantity of unique data values stored.

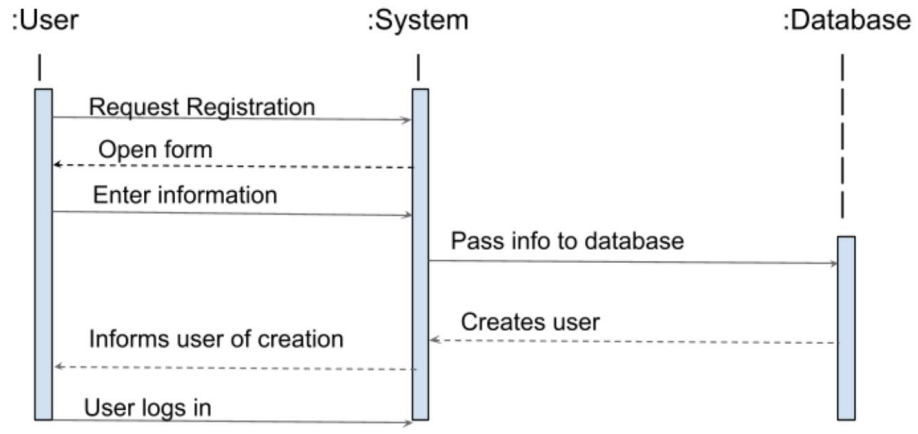

Figure 4: User Registration Sequence Diagram

Figure 4 shows how a user can request the creation of an account using the system. The system responds by opening a form and the user enters all of the information required to make an account. The server sends this off to the database where a user is added to the user database. Once this happens, the server notifies the user and the user can log in to their account.

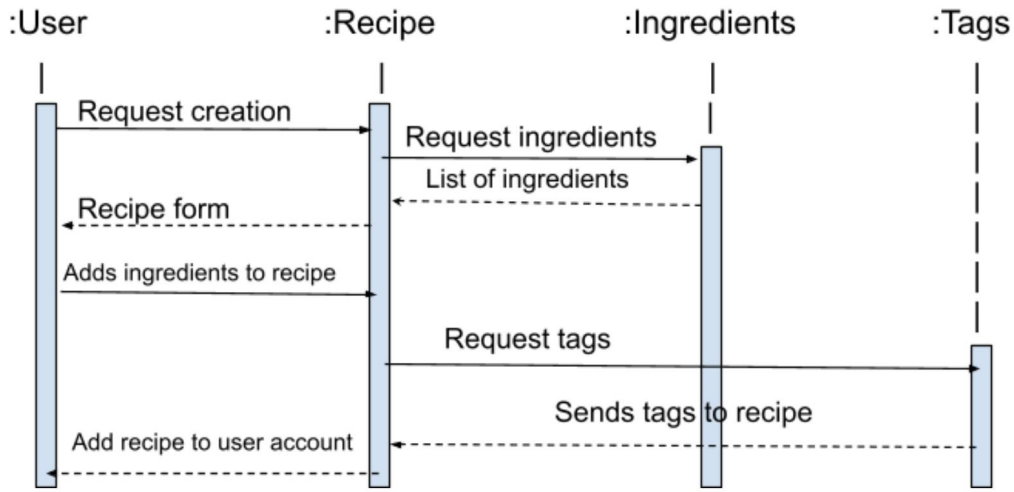

Figure 5: Recipe Creation Sequence Diagram

Figure 5 shows the relationships between the user, recipe, ingredients, tags. Once again, the user requests the Recipe class to request ingredients and tags. Once they are sent from their corresponding classes, they are sent back to the user, and the completed recipe is then added to their account.

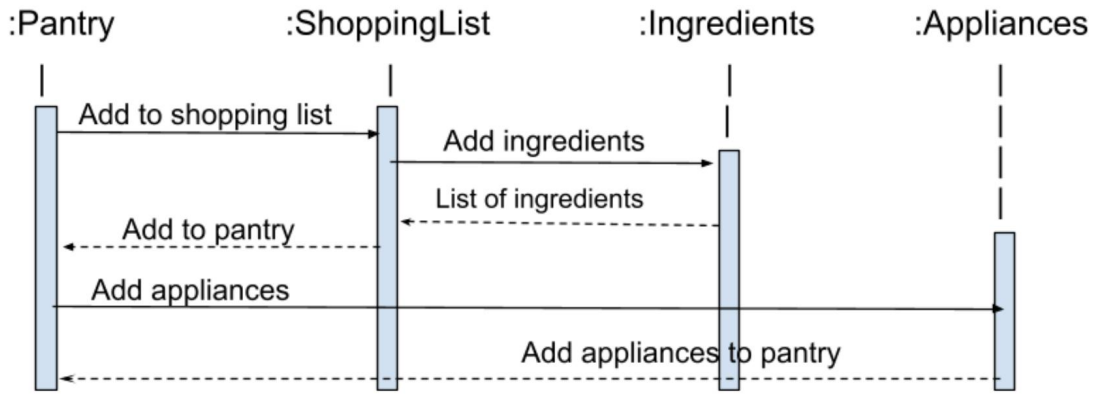

Figure 6: Pantry Sequence Diagram

Figure 6 shows the relationship between pantry, shopping list, ingredients, appliances. The diagram showcases how when the user adds an ingredient to their shopping list, it is also added to the ingredient list. From the ingredients list, a user can also add their pantry or shopping list. The user can also add appliances to the appliances list.

Figure 7: Shopping List Sequence Diagram

Figure 7 shows the relationship between user, ingredients and shopping list. The user will select an ingredient from a list of ingredients, and that ingredient object will get passed to the shopping list class. The shopping list will add the ingredient to the user's shopping list and return the updated list to the user.

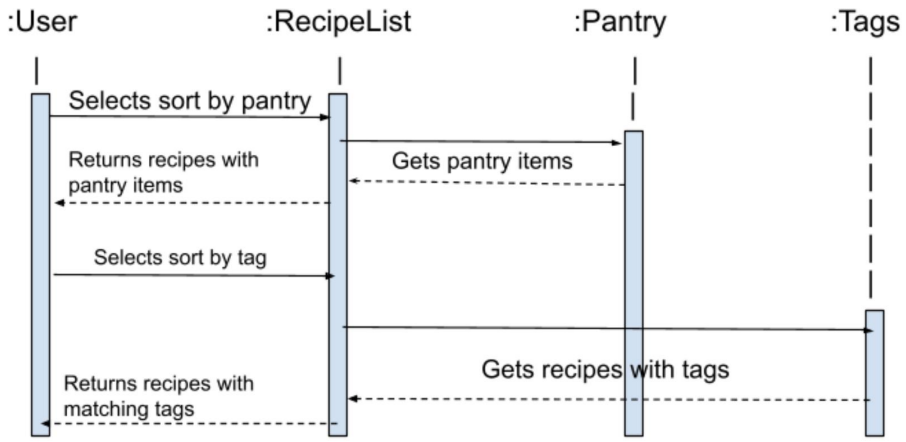

Figure 8: Search Sequence Diagram

Figure 8 shows the relationship between the user, recipelist, pantry and tags classes. The user will have an option to select to sort by pantry, tag or keyword. If the user selects by keyword, the recipelist will return to the user all the recipes containing the string provided. If the user selects pantry or tag, the recipelist will return all the recipes with matching pantry items or tags.