Software Requirements and Design Document

For

Group <8>

Version 1.0

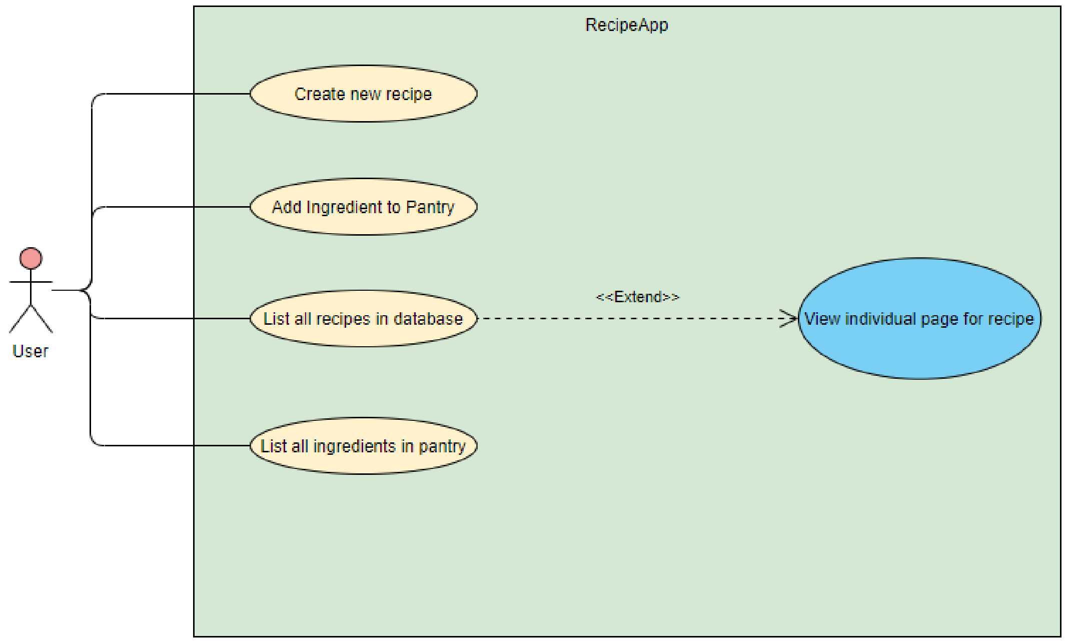
Authors:
Alex J
John F
Justin N
Mason J

1. Overview (5 points)
Our current system primarily runs using HTML on the front end for the user interaction and uses flask on the back end in order to communicate with the databases. There is one overall database with several tables, one which keeps track of the recipes, another to store ingredients associated with each recipeID, the third table stores each step related to a RecipeID. There is an additional table to keep track of the ingredients that the user has on hand, followed by the measurements associated with said ingredient. This database should also keep track of recipes that the user may want to save for later use as well. The system should be quick and easy to use, users should also be able to sign into their account where they can then access recipes they have saved for later use. HTML5 and potentially React as well, will be used to stylize the website on the users end while Flask will primarily be used to ensure database connectivity and communication.

2. Functional Requirements (10 points)
1. System should take in ingredients the user has and return recipes based upon those ingredients (High)
2. System should allow user to submit a recipe to the database (Low)
3. System should allow user to search for recipes by name (high)
4. System should allow user to search for recipes based upon a base meat(for example if the user specifies they would like recipes that use pork chops then only recipes that use pork chops shall be returned) (Medium)
5. System should allow user to list the entire recipe database (High)
6. System should allow user to submit ingredients to the pantry database (High)
7. System Should allow user to view all ingredients that have been entered into the pantry table (High)

3. Non-functional Requirements (10 points)
Performance and versatility. At the current state of development, the system doesn't have proper implementation of security measures and isn't on a large enough scale to worry about efficiency. It is a functionally complete program that should not need the addition of new functions and not require restructuring to give the application a more presentable appearance.

4. Use Case Diagram (10 points)

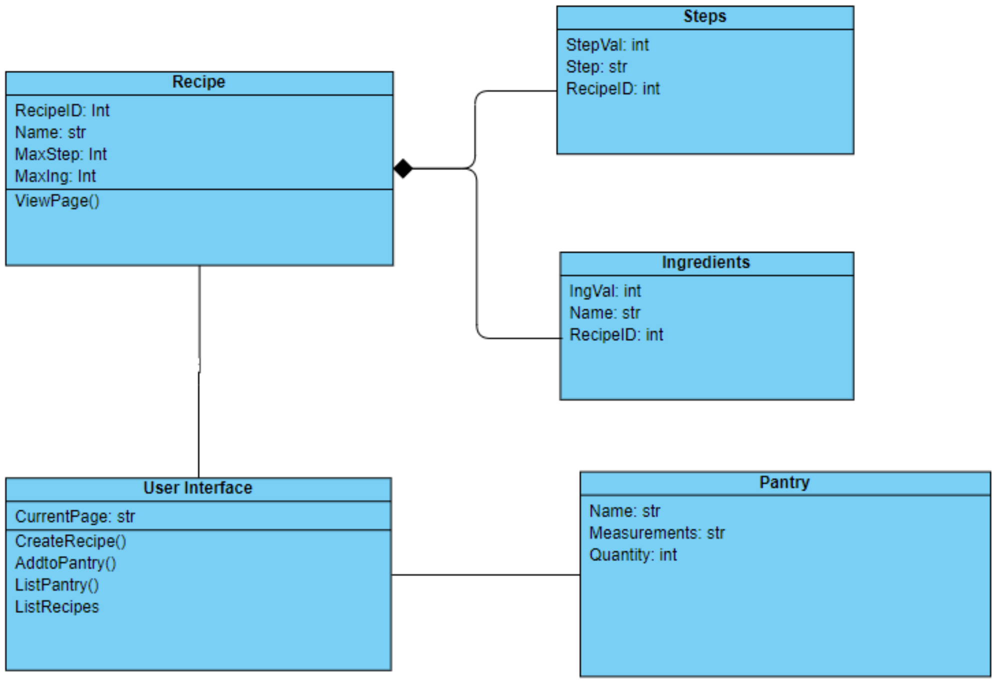

RecipeApp
Create new recipe
Add Ingredient to Pantry
List all recipes in database
List all ingredients in pantry
<<Extend>>
View individual page for recipe
User

Create new recipe: The user can create recipes from scratch by inputting data for the recipe name, the text for the steps of the recipe and the list of ingredients used in the recipe. Individual steps and ingredients are divided by '\n' characters in their respective text boxes while the system tracks the RecipeID across the three tables being used for this use case.

Add ingredients to Pantry: The user has full freedom in adding ingredients to their pantry, tracking the ingredient name, the quantity of said ingredient and the metric by which the quantity is being measured.
List all ingredients in pantry: User is brought to a page that lists all existing ingredients contained in the pantry as well their current quantity.

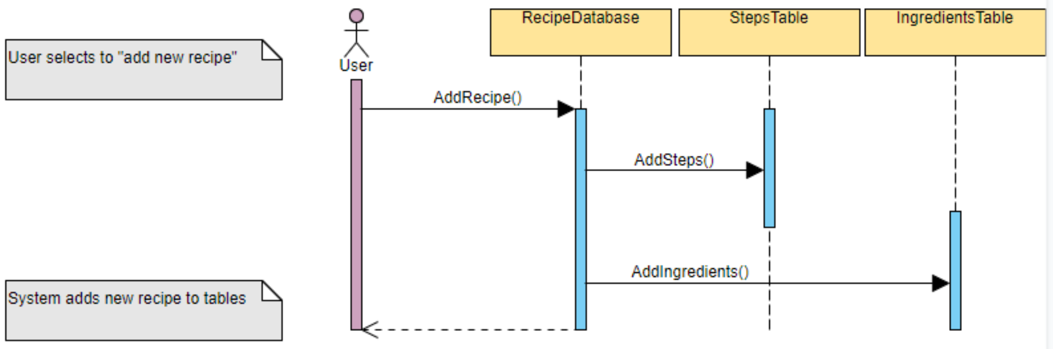
List all recipes in database: User is brought to a page to list all existing recipe names, and will have the option to click said names to extend into the use case for viewing an individual recipe page.
The individual recipe page will list the recipe name as well as an organized view of the steps and ingredients of the recipe given for the use case.

5. Class Diagram and/or Sequence Diagrams (15 points)

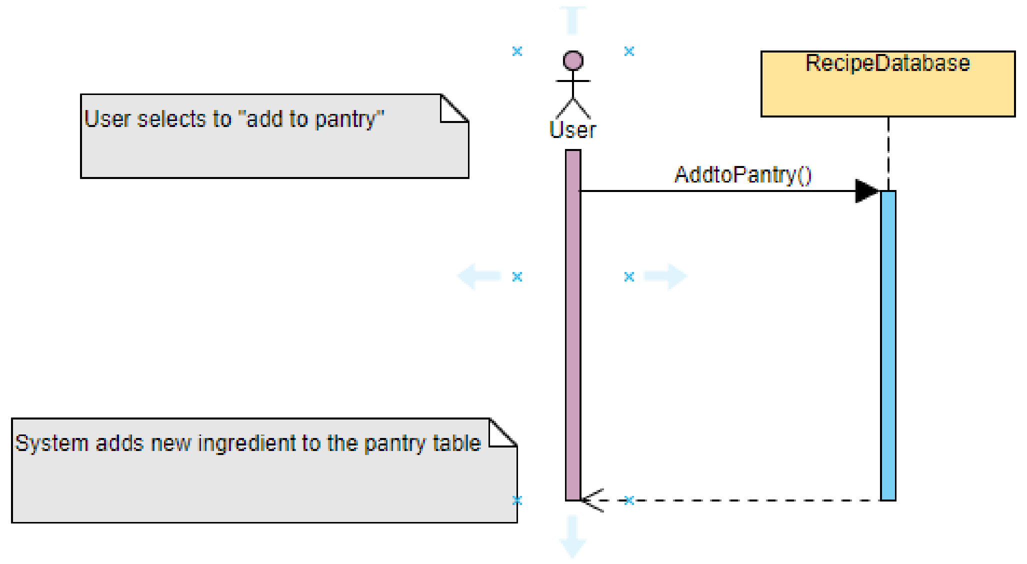


Sequence Diagrams:

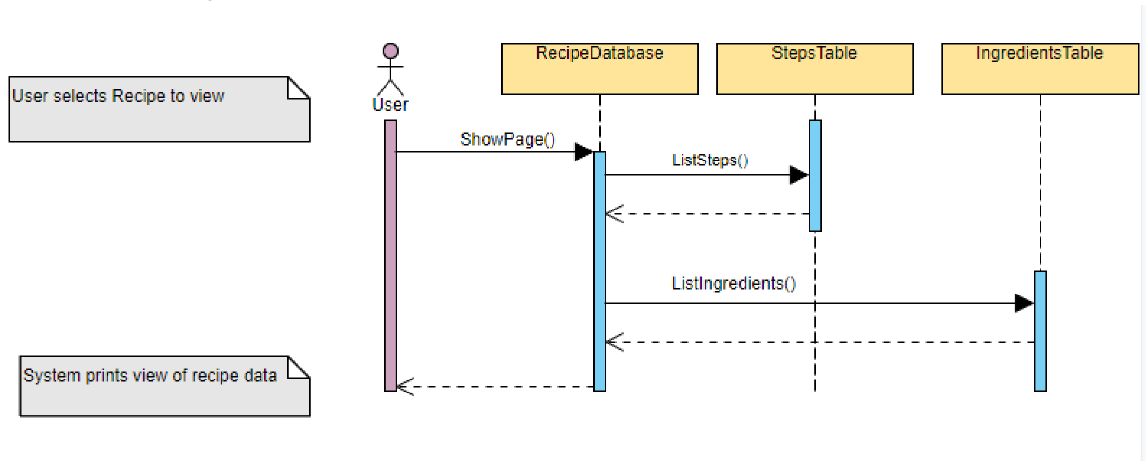
Create a Recipe:



Add to Pantry:


User selects to "add to pantry"
User
RecipeDatabase
AddtoPantry()
System adds new ingredient to the pantry table

View Recipe Page:



6. Operating Environment (5 points)

As of now the operating system runs on HTML5 for the front end of the website which the user interacts with and Flask ver. 2.0.1 is used on the back end in order to communicate with and between the databases.

7. Assumptions and Dependencies (5 points)

A potential dependency of the system revolves around the use of the API MyCookbook.io,which will be used to populate the database with recipes for the user to search through. This dependency comes into play when searching for recipes, so on the off chance that the API is not correctly functioning this could leave our system defective. An Assumed factor taken into consideration is when the user inputs their own recipes there is always the possibility that the user will try to break the system by inputting wrong data. So in order to handle this, fields that require measurements will be restricted to numbers. Users will be restricted to a scroll bar of ingredients to choose from in order to prevent users inputting false input. In regards to instructions, the user will have to input these themselves due to the fact that recipe instructions cannot be so easily limited as ingredients