

Chef Impastable

Team 9



Carmen Del Mastro

Kendalyn Freuhauf

Jiahui Huang

Mahima Malhotra

Sarah Wagler

Index

Purpose	3
Functional Requirements	3
Non-Functional Requirements	5
Design Outline	7
High Level Overview	7
Sequence of Events Overview	8
Design Issues	10
Functional Issues	10
Nonfunctional Issues	12
Design Details	15
Class Design	15
Descriptions of Classes and the Interactions between Classes	16
Sequence Diagrams	21
State Diagram	26
UI Mockups	27

Purpose

Many people living alone may find preparing food to be an overwhelming ordeal. This is especially true for young adults who have little cooking experience and need help finding recipe ideas. Thus, there is a demand for a tool that makes cooking easiest for those new to cooking.

The purpose of our project is to provide a web application that makes cooking easier by taking what is in the user's pantry and recommending meals accordingly. While other recipe recommendation applications exist, none incorporate a depth of unique features that Chef Impastable does in addition to recipe suggestions. For example, our application will incorporate a simplified meal prep system, where the user can plan out their meals for the week and calculate their nutrition goals. Additionally, there will be an integrated shopping list and a review system.

Functional Requirements

1. Account

As a user:

- a. I would like to be able to sign up for an account.
- b. I would like to be able to login to my account.
- c. I would like to be able to reset my account password.
- d. I would like to be able to logout.

2. Recipe interactions

As a user:

- a. I would like to be able to save recipes.
- b. I would like to be able to organize recipes into customizable folders.
- c. I would like to be able to create recipes.
- d. I would like to be able to view my saved recipes and recipes I have created.

3. Ingredient interactions

As a user:

- a. I would like to be able to add ingredients to the ingredient database.
- b. I would like to be able to pull ingredients from the database when creating a recipe.
- c. I would like to be recommended possible substitute ingredients.

- d. I would like to be able to tag ingredients as vegetarian, vegan, gluten-free, or by other allergens.
4. Profile
 - As a user:
 - a. I would like to be able to view my profile page.
 - b. I would like to be able to edit my profile page.
5. Sharing
 - As a user:
 - a. I would like to be able to share recipes with others.
 - b. I would like to be able to view my shared recipes
 - c. I would like to be able to add/remove friends.
 - d. I would like to be able to view my friend's profiles.
 - e. I would like to be able to see notifications of shared recipes and new friends.
 - f. I would like to add other users to my "household".
 - As a user in a household:
 - g. I want to see which recipes my household has favorited.
 - h. I would like to differentiate between shared fridge items and my own fridge items.
6. Reviewing recipes
 - As a user:
 - a. I would like to be able to review recipes.
 - b. I would like to be able to view a recipe's ratings and reviews.
 - c. I would like to be able to view the recipes I have already rated/reviewed.
7. Recipe details
 - As a user:
 - a. I would like to be able to see the ingredients and nutrition facts of a recipe.
 - b. I would like to be able to see the steps in making the recipe.
8. Fridge and Kitchen
 - As a user:
 - a. I would like to be able to edit my "fridge" (owned food).
 - b. I would like to be able to view my "fridge".
 - c. I would like to be able to edit my "kitchen" (owned equipment).
 - d. I would like to be able to view my "kitchen".
9. Filtering
 - As a user:
 - a. I would like to be able to edit my dietary restrictions/preferences.
 - b. I would like to be able to search for specific food recipes which include ingredients from my fridge.
 - c. I would like to be able to search and filter for recipes by various factors such as cuisine and meal type.
 - d. I would like to be recommended recipes based on my past history of

liked/disliked recipes.

- e. I would like to be recommended recipes based on my fridge and kitchen.
- f. I would like to be able to search by event (birthdays, holidays, etc.).

10. Shopping lists

As a user:

- a. I would like to be able to add ingredients to my shopping list.
- b. I would like to edit and combine ingredients in my shopping list.
- c. I would like to be able to view my shopping list.

11. Meal planning and nutrition goals

As a user:

- a. I would like to be able to plan my meals/snacks for the week by adding them to a weekly calendar.
- b. I would like to view my meals, and their ingredients, planned for the week in a calendar.
- c. I would like to be able to add planned meal ingredients to my grocery list, based upon fridge contents or disregarding fridge contents.
- d. I would like to be able to define my nutrition goals for the day/week.
- e. I would like to be able to view/visualize how my meal plan accomplishes my nutrition goals.

Non-Functional Requirements

1. Performance

As a developer,

- a. I want the application to be responsive to user interaction and run well.
- b. I want the server to be able to support thousands of users.
- c. I want any errors in the client or server to be handled smoothly.

2. Server

As a developer,

- a. I want the server to store information about the user such as user credentials.
- b. I want the server to store information about recipes and ingredients web scraped from the internet.
- c. I want the user to be able to add recipes and ingredients to the database.
- d. (If time allows) I want to prevent users from adding bad data to the database, such as adding random words to the ingredients list.

3. Appearance

As a developer,

- a. I want the application to be intuitive, easy to use, and aesthetically pleasing.

4. Security

As a developer,

- a. I want to protect the user's personal information that is being stored, such as the

username and password.

5. Usability

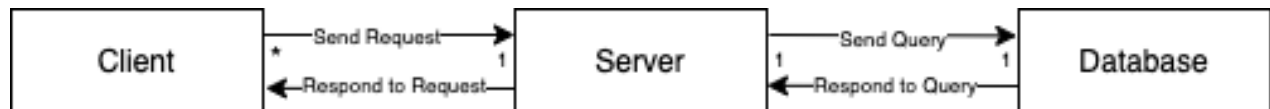
As a developer,

- a. I want the application to have a user interface that is easy to navigate.
- b. I want the application to be available on web browsers.
- c. (If time allows) I want the application to be available on mobile browsers.

Design Outline

High Level Overview

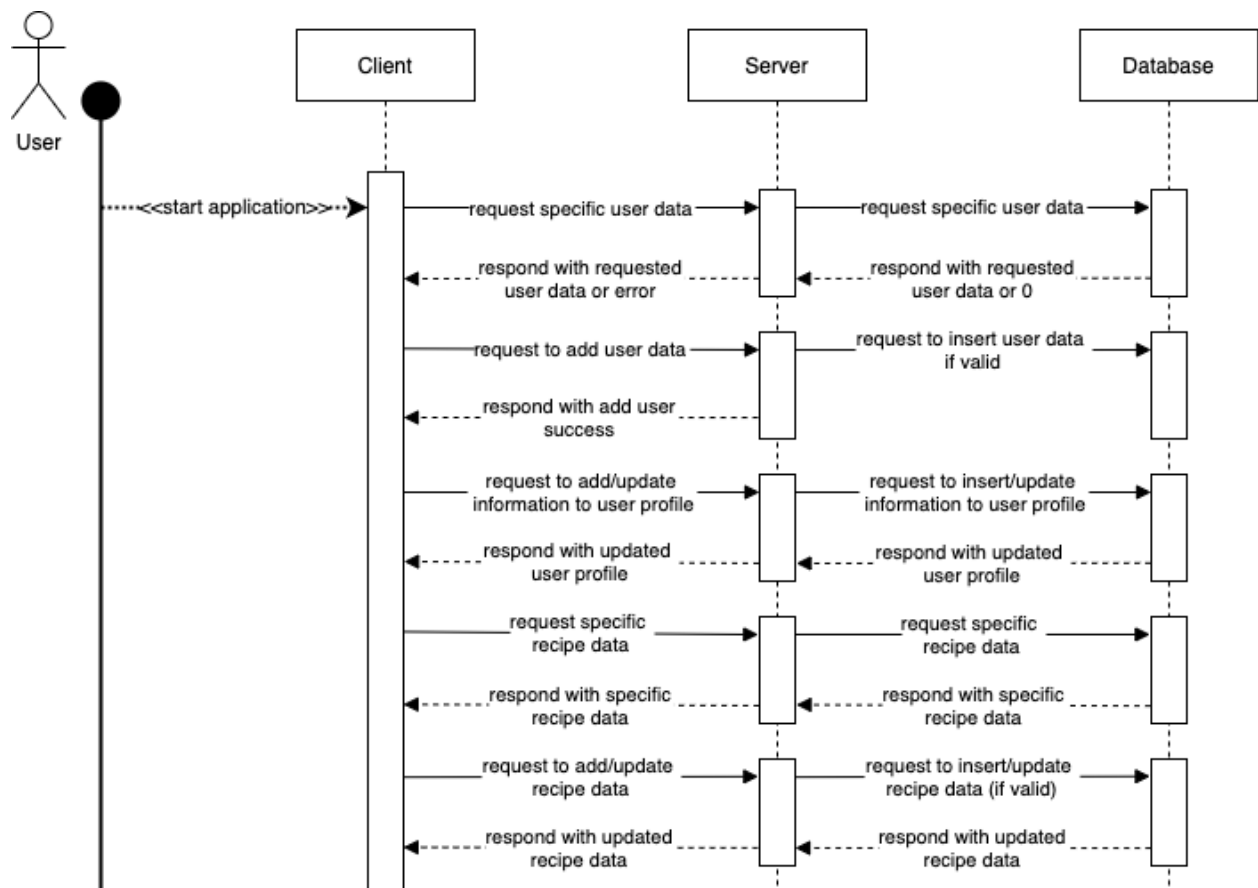
The project will be a web application that allows users to find recipes by name and by basic user inputted details. This application will use a client-server model that can handle multiple clients using a Node.js framework. The server accepts ajax requests from the client, sends a query to the database, and responds back to the client.

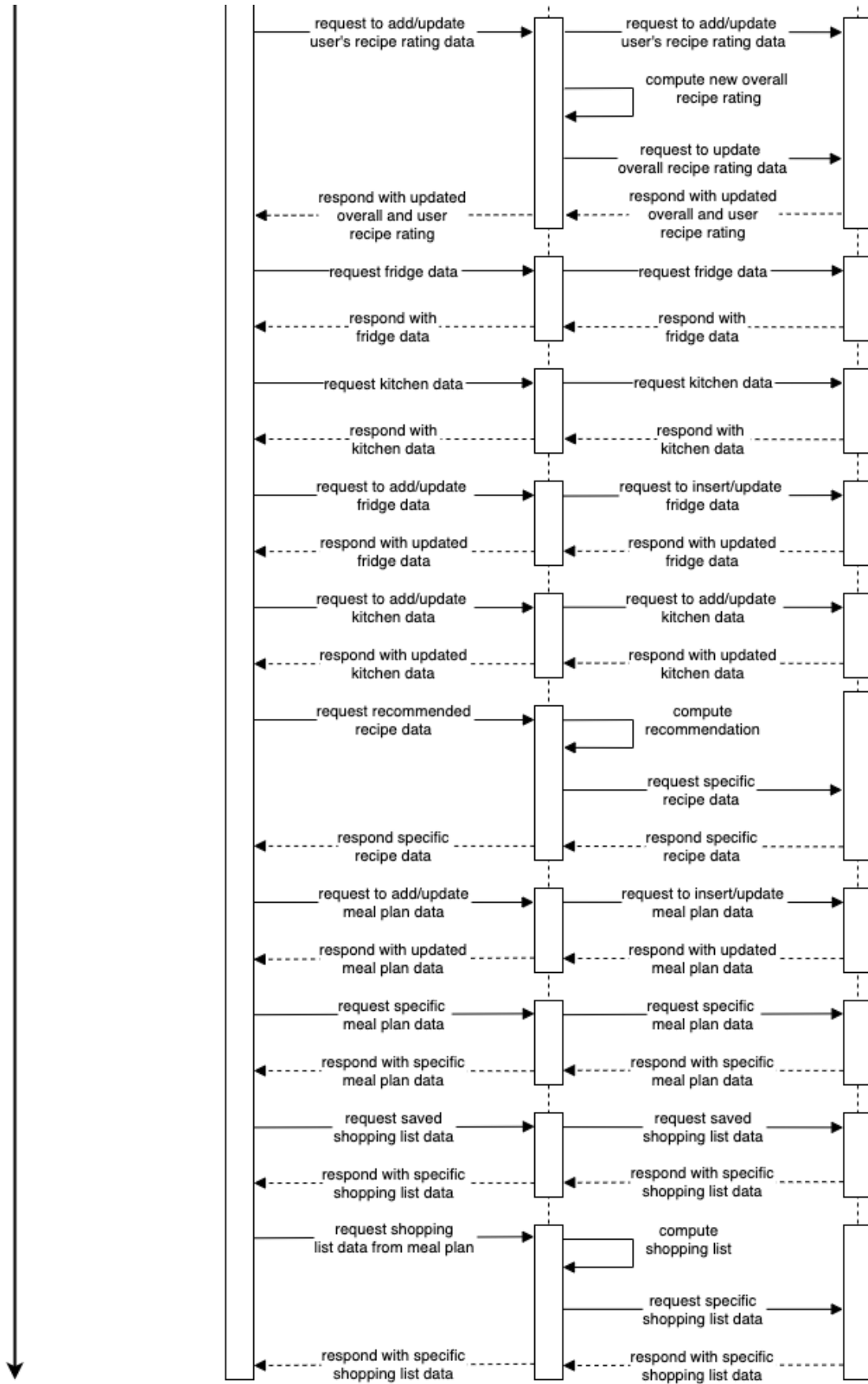


1. Client
 - a. Client acts as an interface for the user to interact with the system
 - b. Client sends ajax requests to the server
2. Server
 - a. Server receives and handles ajax requests from the client
 - b. Server processes requests and sends queries to the database to add/edit the data within.
 - c. Server generates a response and sends it back to a specific client
3. Database
 - a. Database stores data used in the application including user, recipe, and ingredient information
 - b. Database receives and responds to queries from the server

Sequence of Events Overview

The sequence overview diagram shows the basic interactions between the clients, server, and database. After the user opens the web application, they are prompted to log in. As the user logs in, the client sends a request to the server and the server sends a query to the database then receives data from the database. The server can then use the data to respond to the client with either a successful log in or a failed login. After logging in, there are a variety of features available to the user such as editing profile, fridge, and kitchen data, creating and saving recipes, and creating meal plans and shopping lists which would all include the client sending requests for the server to field as queries to the database. Then the database would respond to the query with the updated or requested data.





Design Issues

Functional Issues

1. Can a user be a part of multiple households?

- Option 1: No, one household should suffice
- Option 2: Yes, but limit the number to 3 households
- Option 3: Yes, and there is no maximum number of households

Choice: Option 3

Justification: Users of our application may want to have more than one household if they live or cook with multiple different groups of people. For example, a college student that lives in an apartment with roommates during the semesters may also live with their parents over break. Therefore, one household may not suffice. Additionally, limiting the user to a few households may also feel restricting. Having no cap on the number of households gives users the freedom to customize how they see fit. It is unlikely for them to create an absurd amount of households as it would be detrimental to their own organization experience.

2. Can the user use the website without an account?

- Option 1: No
- Option 2: Yes

Choice: Option 1

Justification: In our design, all of the user's information is linked to their account, such as the ingredients in their fridge and their nutrition goals. A guest or general account would limit the personalizable features our application will offer, including personalized

nutrition goals and saved folders of recipes. Requiring an account will ultimately be easier to implement as opposed to a system that saves the data based on the user's device.

3. How should we allow users to save recipes?

- Option 1: A single list of saved recipes
- Option 2: A list of folders which each contain a list of save recipes
- Option 3: A list of nested folders

Choice: Option 2

Justification: A list of folders will allow the user to categorize groups of recipes for different events of their choosing. A list of nested folders would overcomplicate the implementation while not adding much utility to the design; nested folders could potentially become disorganized and become counterintuitive to the user. Therefore, a single list of folders will give the user a reasonable amount of customization.

4. What types of nutrition goals should a user be able to have?

- Option 1: Restricted to predefined types
- Option 2: Free input by users

Choice: Option 1

Justification: Restricting the user to predefined types allows the application to calculate how much the user has met their nutrition goals based on the nutrition content of the recipes they have consumed. If the user could freely input types, the values would most likely be inconsistent which would make it harder from an implementation standpoint to compare a recipe's nutrition data to the user's input. Therefore, restricting the user to types helps prevent errors while calculating the user's nutrition goals.

5. How should a user be able to qualify success in meeting a nutrition goal?

- Option 1: Hitting more than or less than their defined goal
- Option 2: Falling between a range of values, with a target in mind
- Option 3: Reach exactly the value they defined

Choice: Option 2

Justification: For most users, hitting an exact number for their nutrition goal, for example 100 g of protein, would likely be too restrictive and could prevent them from using the feature altogether. For those who may be trying to fall underneath a certain value, such as calories, Option 1 doesn't seem to be the most reasonable, as there are also healthy amounts of nutritional categories that a user still may want to consume. Therefore, option 2 allows a user to set their goals for a nutritional value, while also providing a range that they believe will still allow them to attain their goals and achieve success.

Nonfunctional Issues

1. Which web scraping language should we use?

- Option 1: Javascript
- Option 2: Python
- Option 3: C++

Choice: Option 2

Justification: Python contains many libraries to help with efficient web scraping and parsing. There is BeautifulSoup, Scrapy, Selenium, Requests, Urllib3, Lxml, and MechanicalSoup to name a few of the most widely known and used. Both Javascript and C++ have web scraping capabilities, but are not as simple.

2. Which website to web scrape?

- Option 1: allrecipes.com
- Option 2: spoonacular.com (API)
- Option 3: edamam.com (API)

Choice: Option 3

Justification: edamam.com gives us access to nicely formatted recipe and ingredient information without having to pay for multiple requests, like spoonacular.com does.

Since edamam.com is a recipe search API, accessing the data is easy and simple, but with allrecipes.com we would need to do more web scraping to get everything that we need..

3. What frontend framework/language should we use?

- Option 1: React
- Option 2: Vue
- Option 3: Angular

Choice: Option 1

Justification: Some of our team members already have experience with React, and the framework is beginner friendly. React allows developers to build interactive UI with components, and it renders pages quickly due to its use of virtual DOM. Additionally, React is frequently used with a MongoDB database, and a backend composed of Node.js and Next.js. Therefore, React is a good choice because there is plenty of documentation available about this particular stack and about React itself.

4. What database should we use?

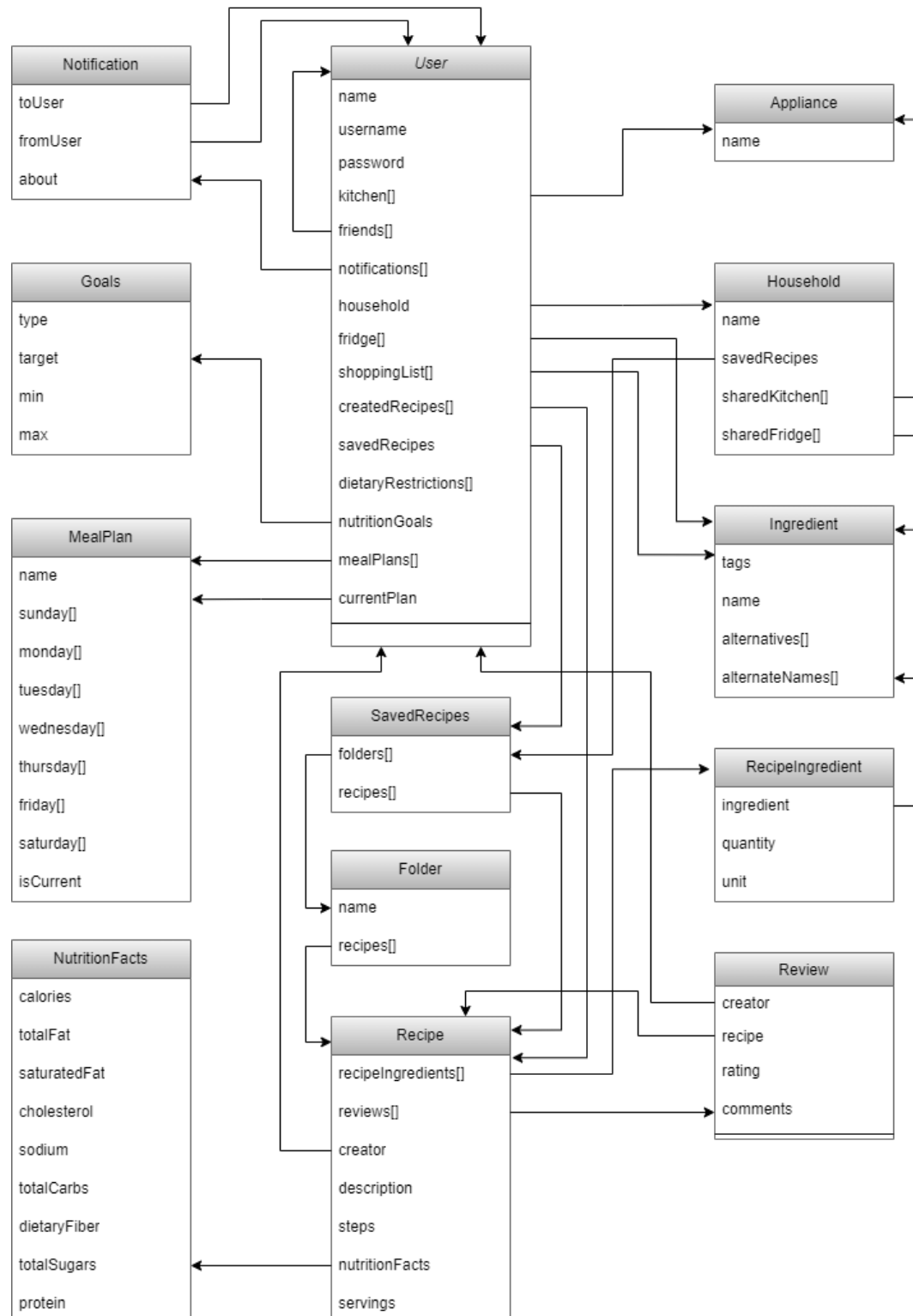
- Option 1: MySQL
- Option 2: Oracle Database
- Option 3: MongoDB

Choice: Option 3

Justification: NoSQL databases allow for greater flexibility than SQL databases due to the fact that information can be stored in different structures, while SQL databases are restricted to defining rows and columns. MongoDB stores information as documents that are JSON formatted, which will allow us to neatly store recipe and ingredient data that we web scrape from the internet. MongoDB is also beginner friendly, as our team does not have much experience with databases. Additionally, MongoDB has a lot of documentation with React, making it a good choice.

Design Details

Class Designin



Descriptions of Classes and the Interactions between Classes

- User
 - The user is created when someone signs up or logs in.
 - In order to sign up, the user will enter their name, username, password, and email.
 - Each user will have a kitchen, where they can enter and see their currently owned appliances.
 - Each user will have a fridge, where they can enter and see their currently owned ingredients.
 - Each user will receive notifications of new friends, shared recipes from friends, etc.
 - Each user can become a part of a household or many households, such as a household of their roommates or family.
 - Each user can create a shopping list, where they can store and see a list of ingredients they need for recipes.
 - Each user can create their own recipes.
 - Each user can save recipes that they like.
 - Each user can state their dietary restrictions, such as gluten-free or dairy-free.
 - Each user can state their nutrition goals, such as consuming higher protein.
 - Each user can create meal plans.
 - Each user can state which of their meal plans is the one they are currently using.
- Recipe
 - Recipes are created through the stored recipes in the database or created by any user.

- Each recipe contains a list of recipe ingredients used in the recipe.
- Each recipe contains a list of reviews of that recipe created by other users.
- Each recipe has a creator or the creator is the database.
- Each recipe has a description of the recipe.
- Each recipe has a list of steps taken to create the recipe.
- Each recipe has nutrition facts.
- Each recipe states the number of servings it creates.
- RecipeIngredient
 - A recipe ingredient is used in a recipe.
 - Each recipe ingredient has its ingredient it is created from.
 - Each recipe ingredient has a quantity that is used in the recipe.
 - Each recipe ingredient has a unit that is used for measurement of the ingredient.
- Ingredient
 - An ingredient is used to specify something used in a recipe ingredient (i.e, will be used in a recipe). They can also be listed as items the user has in its fridge or shopping list.
 - Each ingredient has a name.
 - Each ingredient has tags to specify certain allergens such as containing eggs or gluten.
 - Each ingredient has a list of alternatives that can be used in replace of it in a recipe.
 - Each ingredient has a list of alternative names that may be used in a recipe. An example is “flour” is the same thing as “all-purpose flour”

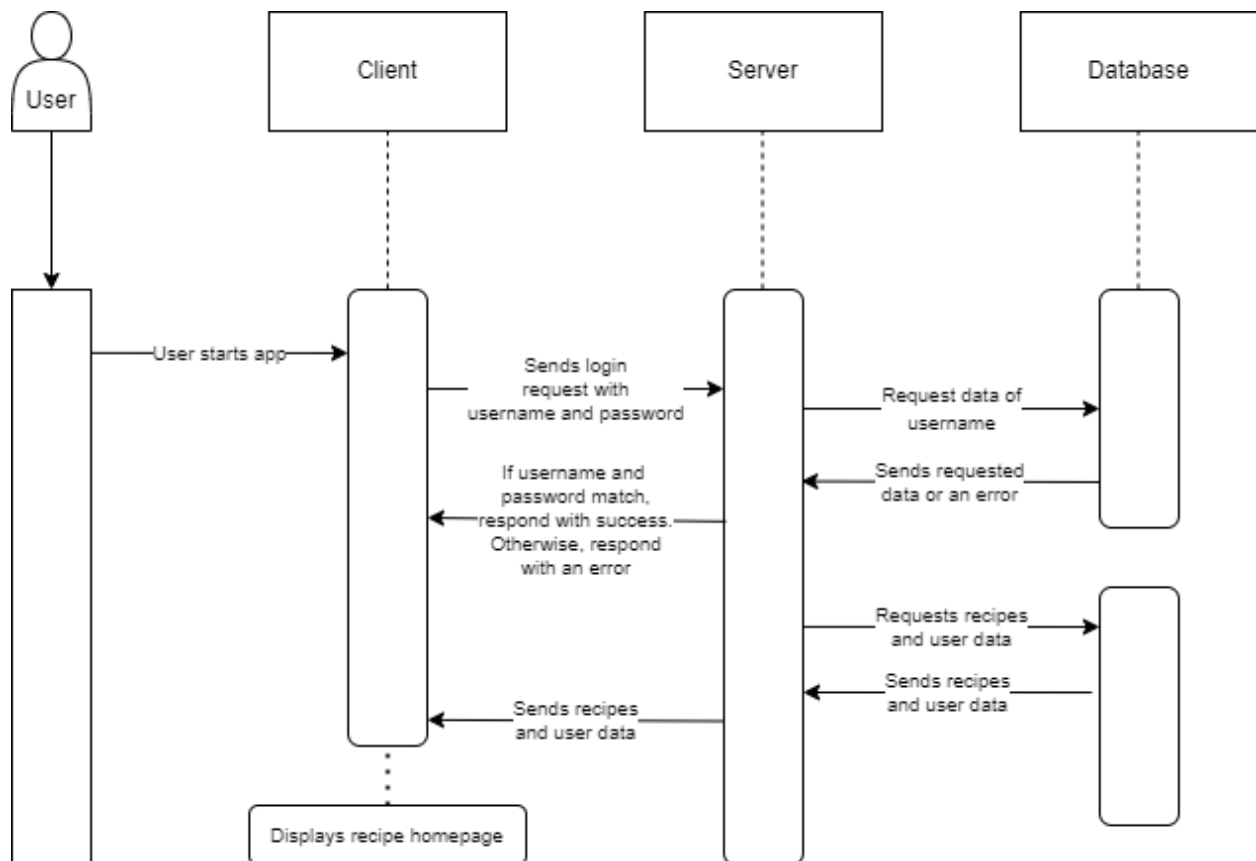
- Review
 - A review is created by a user for a recipe.
 - Each review stores the creator of the review.
 - Each review has a recipe that it was created for.
 - Each review has a rating of the recipe specified by the user who created the review.
 - Each review has comments created by the user who created the review.
- NutritionFacts
 - Nutrition facts store the nutrition details of the recipe.
 - Each nutrition fact of a recipe contains the calories, total fat, saturated fat, cholesterol, sodium, total carbohydrates, dietary fiber, total sugars, and protein that the recipe contains.
- SavedRecipes
 - Saved recipes are the recipes that were saved by a specific user or household.
 - Saved recipes contains a list of folders that can be used by the user to organize all of their saved recipes.
 - Saved recipes contains a list of recipes that were saved by the user but not placed into a folder.
- Folder
 - A folder can be used by a user to organize their saved recipes.
 - Each folder has a name that can be specified by the user who created the folder.
 - Each folder has a list of recipes that the user placed in the folder.
- Notification

- A notification is used to alert a user of events that have occurred from their friends.
- Each notification specifies the user to who the notification is from.
- Each notification specifies the user to whom the notification is to.
- Each notification has an about string to state what the notification is for.
- Household
 - Each user can become a part of one or many households, such as a household of their family members or roommates.
 - Each household has a name.
 - Each household has a list of saved recipes saved by the users that are a part of the household.
 - Each household has a shared fridge among all its members, where the users can specify the ingredients in the house that are shared among them.
 - Each household has a shared kitchen among all its members, where the users can specify the appliances in the house that are shared among them.
- Appliance
 - An appliance is any tool that a user owns, such as an air fryer or mixer.
 - Each appliance has a name.
- Goals
 - Goals are something that a user can specify that they are trying to achieve.
 - Each goal has a type, where the user can specify what they are trying to achieve such as increasing the protein.

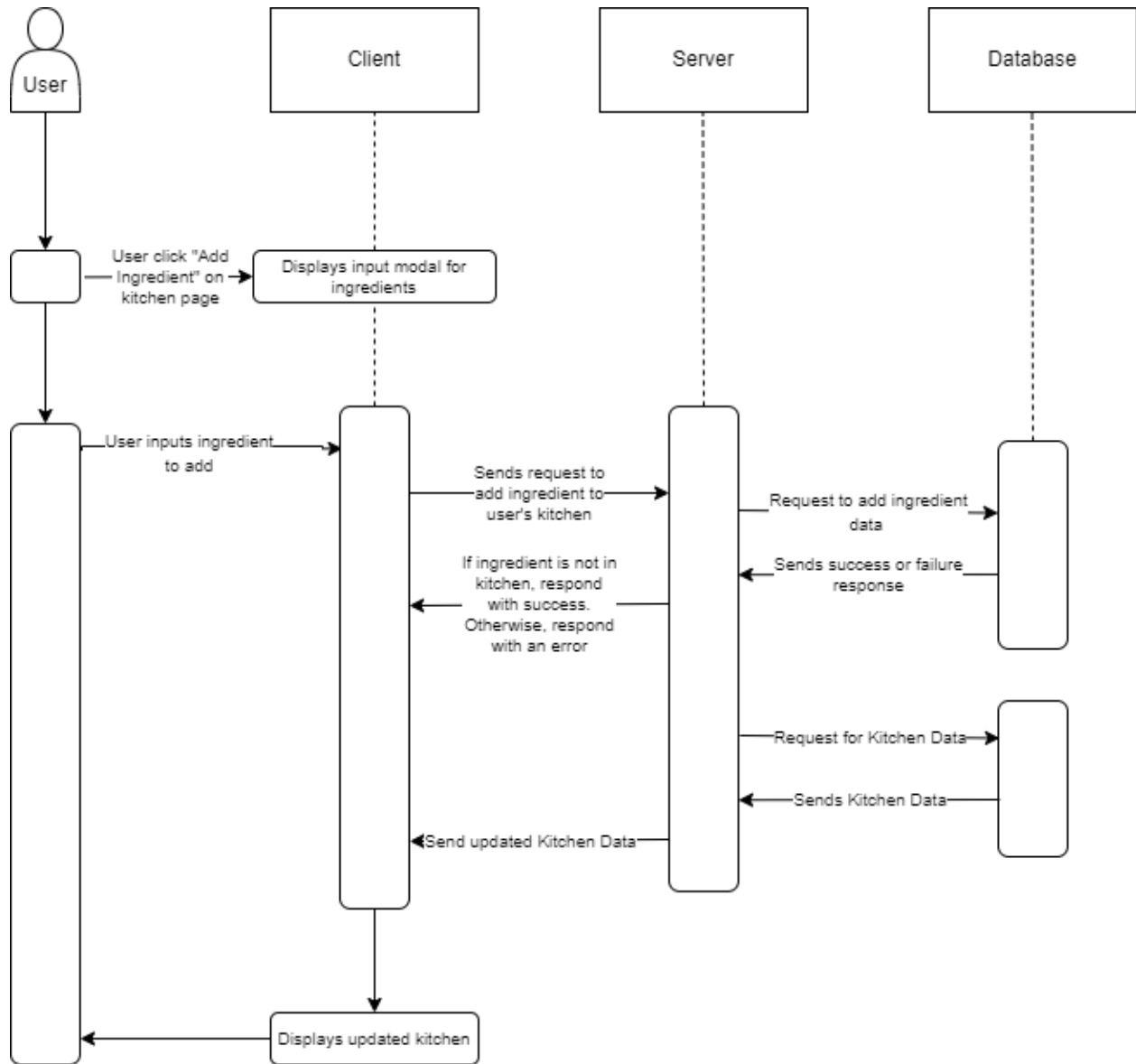
- Each goal has a target, where the user can specify the target they are trying to achieve.
- Each goal has a minimum, where the user can specify the minimum amount of the type they are trying to achieve.
- Each goal has a maximum, where the user can specify the maximum amount of the type they are trying to achieve.
- MealPlan
 - A meal plan is a week calendar that the user can plan the recipes that they plan to make that week.
 - Each meal plan has a name, which the user specifies to distinguish between other meal plans they have created.
 - Each meal plan has a isCurrent field, which is used to specify which meal plan the user is currently using.
 - Each meal plan has a list of recipes for each day of the week. The user can add a recipe that they plan on making to a certain day.

Sequence Diagrams

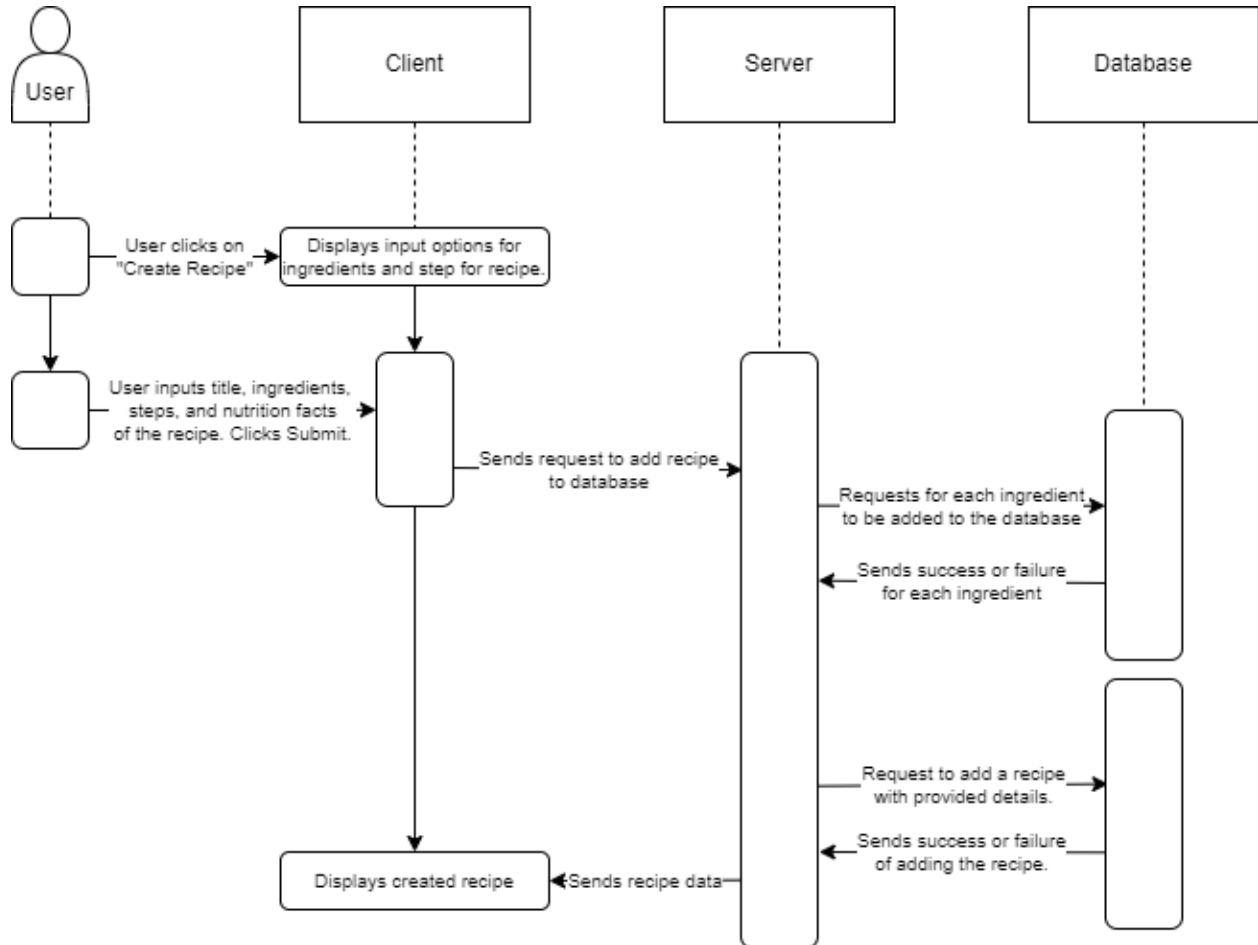
- Sequence of events when a user logs in to their account



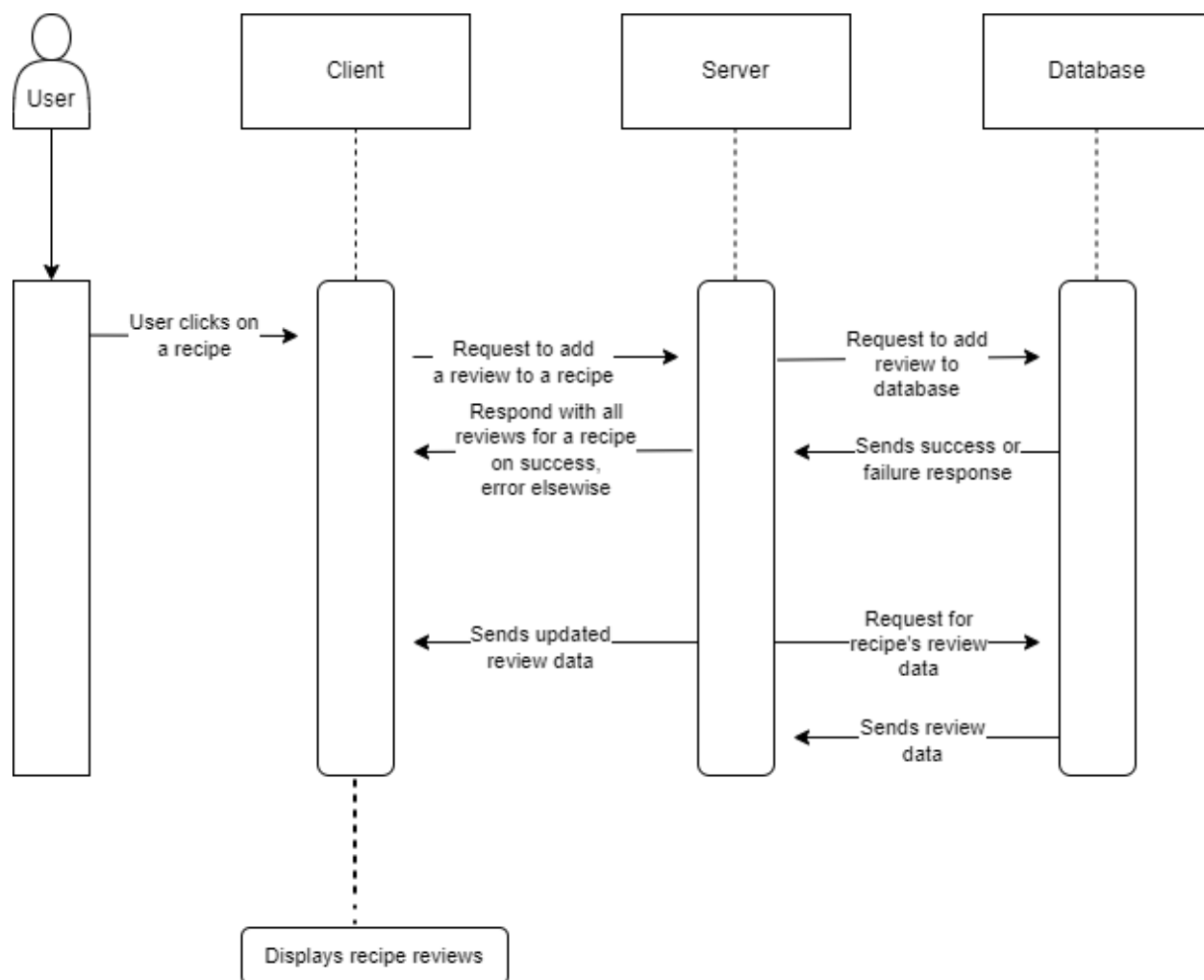
- Sequence of events when a user adds an ingredient to their kitchen



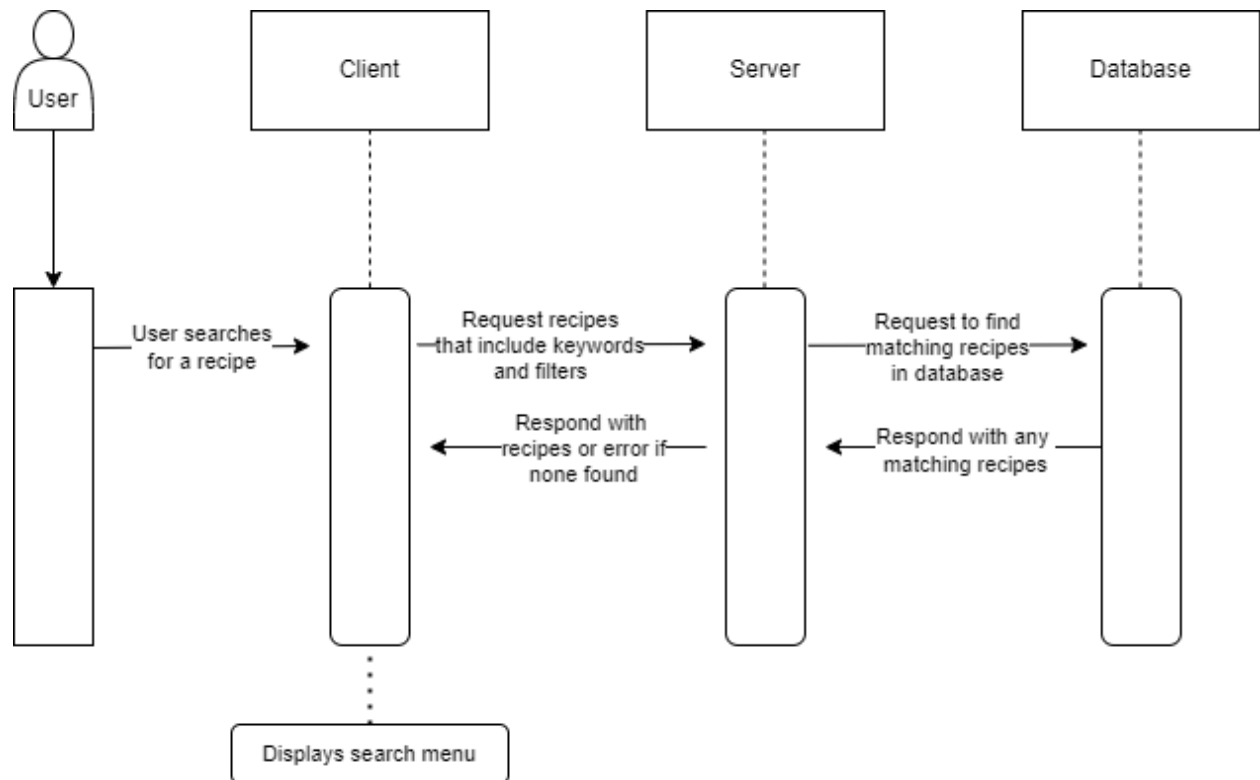
- Sequence of events when a user creates a recipe



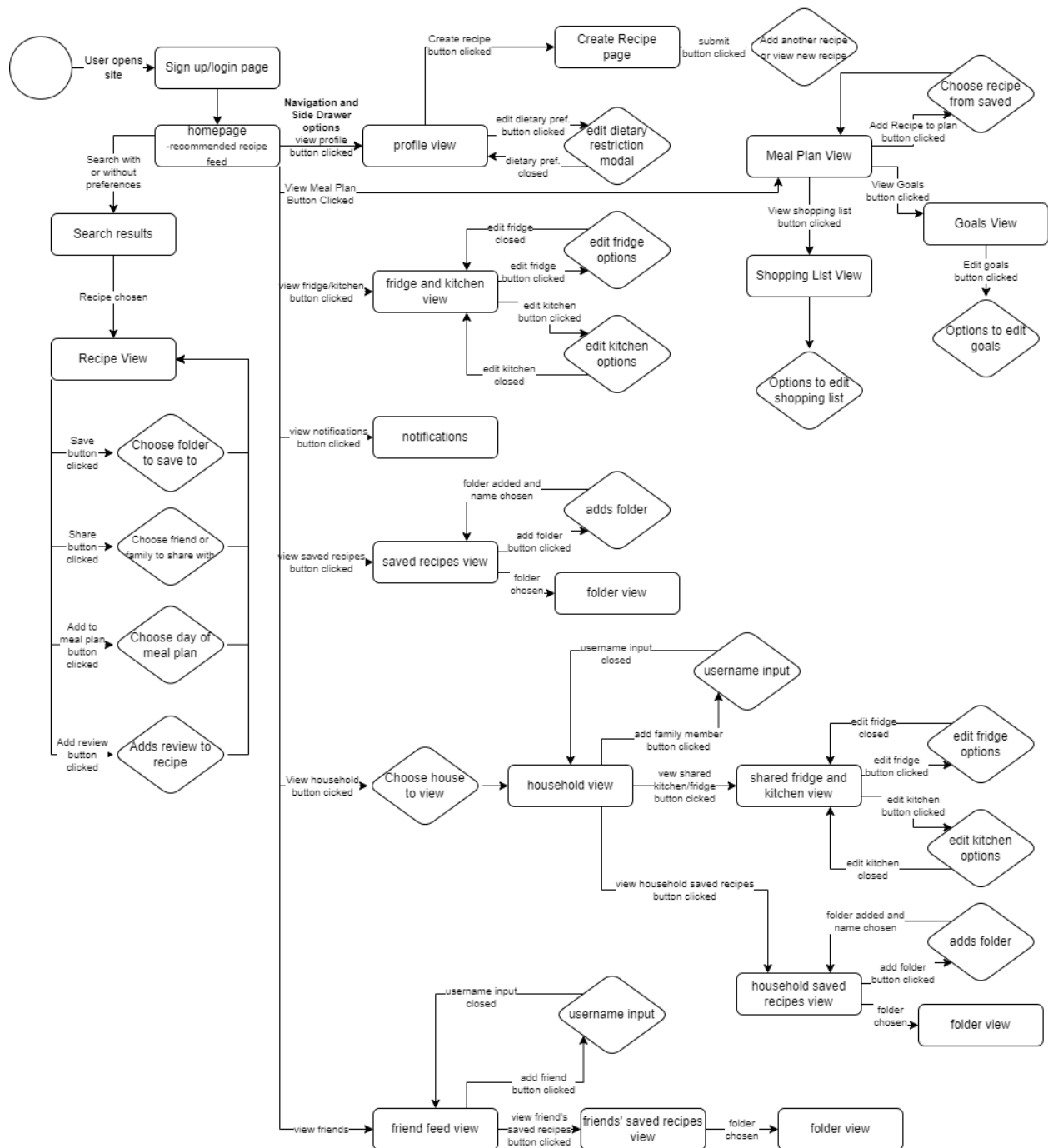
- Sequence of events when a user reviews a recipe



- Sequence of events when a user searches for a recipe

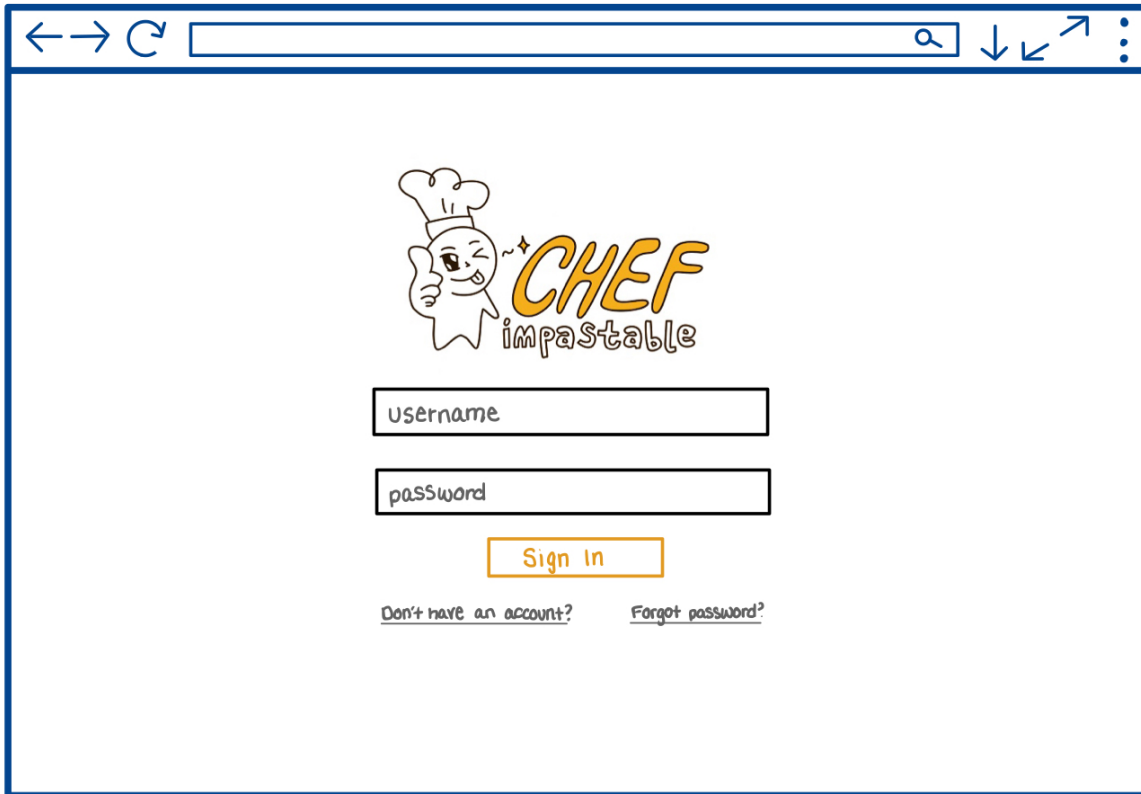


State Diagram




UI Mockups

- User login page



A user login page mockup for 'CHEF impastable'. The page is enclosed in a blue border representing a browser window. At the top, there is a navigation bar with back, forward, and refresh icons, a search bar, and zoom controls. The main content area features a logo of a smiling chef character with the text 'CHEF impastable' in a stylized font. Below the logo are two input fields: 'Username' and 'password'. A yellow 'Sign in' button is positioned below the password field. At the bottom, there are two links: 'Don't have an account?' and 'Forgot password?'.

← → ↻ 🔍 ⏴ ⏵ ↗ ⋮

 **CHEF**
impastable

[Don't have an account?](#) [Forgot password?](#)

- Homepage with navigation bar and side drawer

