Team 1: Recipe Planner

Names: Andrew Blake Madison, Dominic Portolese, Hamza Ahmed, Melaku Tarekegn, Talon Martin, Yamlak Takele

14.8.0 - 2/24/22: Professor feedback from sequence diagrams 50

The following is the foundation for the Recipe Planner system.

Project Description

Overview

Recipe Planner is software that allows the user to create and store recipes in the cloud. The user can create an eating plan using these recipes. The software analyzes this plan, and returns to the user information such as what ingredients they will need to purchase and in what quantities, the overall nutritional value for each day along with warnings if any nutrient is not balanced. The software is also equipped with a social networking system that allows users to use their Google login and interact with other users. These interactions include messaging, adding friends, and viewing recipes of other users.

Key Features

The key features this app must have are:

- Available on iOS and Android.
- Ability to view, add, edit, and remove recipes.
- Ability to view, add, edit, and remove ingredients in the recipes.
- Ability to view and edit user profiles.
- Ability to calculate ingredients that are available and needed for future use.
- Ability to notify beforehand when ingredients run out of stock.
- Ability to tell how healthy a recipe is and the healthy percentage of ingredients in the recipe.
- Ability to share recipes with other users through the application.

Market Value

Many families in the US have had the tradition of passing down family recipes to future generations. Families would share or trade recipes with other families and this was viewed as something special. In the modern day, the internet gives people the power to instantly share information with one another and archive information of all kinds. With a simple internet search, one could find a recipe to cook pretty much anything.

The application will serve as a one-stop shop for those looking to try their hand at cooking something new. Our vision is a recipe database where users can search for new recipes and share them with others. Also included in our application are various tools for users to track ingredients as a type of "grocery list" and plan out meals they wish to cook in the future. With these assistive capabilities, social aspects, and quick access to new and delicious recipes from people of all kinds, this app could be seen as a must for anyone who cooks meals professionally or casually.

From a monetary standpoint, the application will cost a small amount of money to purchase. An alternative would be to make the application free, but including advertisements that could be removed with an in-app purchase. Having the app be paid without advertisements is preferred because otherwise advertisements need to be coded into the app.

Scope

This project will hold all important things to fulfill the requirements of stakeholders. Therefore, the size could be as large as we are able to do. Some of the key constraints could be time shortage and limitation of some sources that are related to what we need to accomplish. However, we are going to do this project with well planned strategies to minimize the constraints.

Stakeholders

Project Manager:
Ensure that the project resources are designed in such a manner that objectives are fulfilled promptly.

Developers:
Frontend developer:
Provides the structure, appearance, behavior, and content of everything that appears on the app, and links to the server backend.

Backend developer:
Provides the app's server side functionality, such as writing APIs, creating libraries while working with system components and linking servers to the user.

UX/UI Designer:
Solves user problems and creates an elegant, intuitive design with interfaces.

Salesperson:
Committed for discovering potential clients and effectively advertising the product.

Client Stakeholders:
Client stakeholders for this project are restaurant and catering businesses as well as food product distributors and amateur chefs. To implement the project effectively, the client stakeholders have to provide the requirements. All of the nutrition information comes from the stakeholders. After this, their role is following up and giving frequent feedback.

Objective

The main objective of this project is to help the people get nutritious food when they need it. Therefore, this software could be able to deliver detailed recipes to the user. To do so the cooking recipe app provides information including the ingredients in terms of quality and quantity, for example, organic, inorganic, GMO, non-GMO etc.

Feasibility

The project is very feasible. The software is relatively simple in design, and should not produce an excessive amount of problems in the development process. The software also enjoys the benefit of having to do with eating, making it marketable to a broad range of people.

We may face technical, organizational, operational, economic, support/maintainability, marketability, and schedule issues but all are not beyond the team. We are going to accommodate the issues that could be raised so that it is possible to minimize the effects. Starting from the initial draft until we accomplish the project, team members will apply all efforts to boost the project's feasibility.

Technical

- There should be no real hangups in the technical process due to the simplicity of the design. All parts of this software are known technology, nothing new needs to be invented.

Organizational

- The relative simplicity of the project allows for a small team of people to complete it. This, along with the use of the agile development process, would make it easy to manage the project.

Operational

- The application will require the internet and use some amount of data depending on how often and how much the user uses it. The majority of the data usage will occur during the

first loading of listings and detecting the user's location. The nutritional information will be stored locally to minimize usage.

- Most people have smartphones with internet access, therefore the app won't cost the client much aside from the initial purchase price.

Economic

- The app will cost a small fee to install, then will be free to use indefinitely with no ads. Costs:
  - ☐ Payroll
  - ☐ Renting office space during development
  - ☐ Advertisements
  - ☐ Server maintenance
  - ☐ Updating, adding, or removing features

Support / Maintainability

We do not foresee any potential updates to the application. All functionality will be made available at launch. Some members of the development team will be kept for support, maintenance, future fixes, and to do backups.

Marketability

- Cooking is something that most people need to do on a daily basis. This makes the app very feasible due to the amount of people that may be interested in purchasing it.

- The application is only limited to those with a smartphone, and a means to pay electronically.

Developability

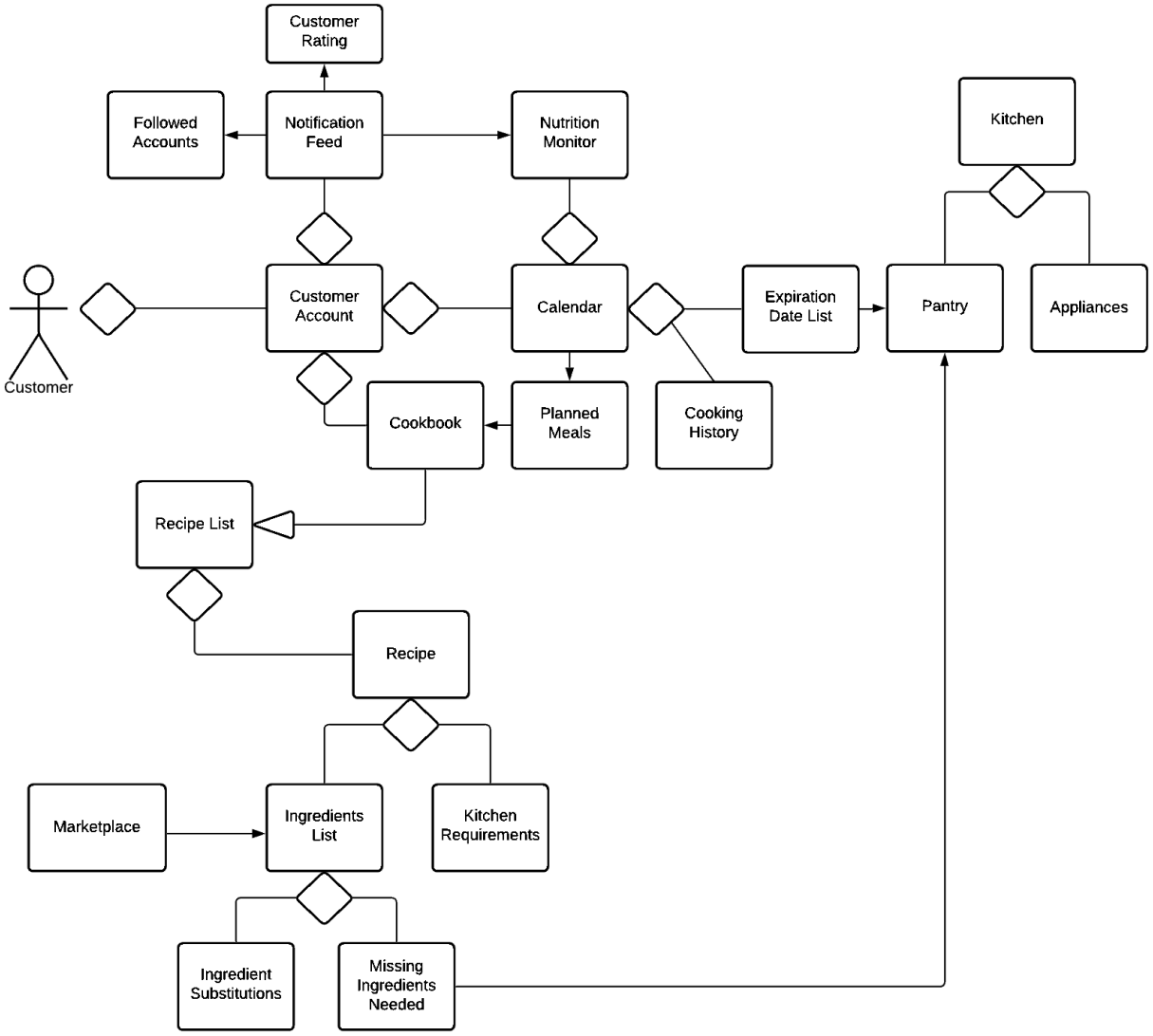
- Adapting the app's functionality in response to client feedback.

- Degree of dedication necessary required to create the app.

Schedule

- With a team of 6-7 developers, the software could be completed within 6 weeks, and the advertising could be arranged in the same time period.

1.0 Domain Model



2.0 User Stories

- As a user, I want to be able to add or remove elements from my recipes in order to get a recipe that has the ingredients I desire.
- As a user, I want to be able to add recipes saved in my cookbook to my planned meals so that I can have a full list of future planned meals in one section.

- As a user, I want to be able to view the calendar to see when my meals are scheduled
- As a user, I want to be able to turn on notifications for my planned meals and to get notified periodically before each meal
- As a user, I want to login to my account to view my saved recipes
- As a user, I want to be able to check which meals are planned for today.
- As a user, I want to substitute missing ingredients in the recipe so that I can have a complete recipe.
- As a user, I want to be able to see recipes that my friends (who also use the app) have made recently, so that I can bond with my friends through our shared interest in exploring new recipes to cook.
- As a user, I want to add the expiration date of the foods I purchase to the expiration date list so that I can use the foods before they expire.
- As a user, I want to know the nutritional content of the food I am eating, so that I can make healthy choices.

3.0 Use Cases

3.1.0 UC-1: Add or Remove

Precondition: User logged in and the app is ready to use.

Actors: User

Trigger: User wants to edit a recipe like removing or adding ingredients.

Basic Flow:

1. User visits the app and chooses the recipe that they need.
2. The system displays recipe details such as the one that the user picked. Example: Caesar Salad has ingredients (Tomato, cheese, chicken, lettuce, white bread, vegetable oil.)
3. User adds one or more new ingredients in the recipe like salt.
4. The system adds and updates the recipe.
5. User removes one or more ingredients from the recipe like tomato.

6. The system removes and updates the recipe.
7. The system records the updated recipe and display.

Alternatives:

1. In step 2: User might want to keep the recipe as they are without adding or removing ingredients.
2. In step 5: Due to a database problem, the system is unable to save the changed Recipe.

3.2.0 UC-2: Substituting ingredients

Precondition: User is logged in to the application.

Actor: User

Trigger: User chooses a recipe that they want to make sure is complete

Basic flow:

1. User selects ‘check ingredients’ for their chosen recipe
2. The system notifies the user of ingredients they are missing
3. User selects each missing ingredient they want to replace
4. The system displays possible substitutions for the selected ingredients
5. User selects the substituted ingredients they want and selects ‘substitute’
6. System adds the substitutions to the recipe

Alternatives:

1. In step 5: the user can enter their own ingredient substitutions instead of the substitutions offered by the recipe

3.3.0 UC-3: Recipes with Friends

Precondition: The user has added their friends to their account’s list of followed accounts within the recipe application.

Actors: One or more Users (who are mutual friends)

Trigger: One or more of a user’s friends successfully cooks a recipe through the application.

Basic Flow:

1. One of a user's friends is viewing a recipe and presses a "Share" button, opening up a pop-up menu.
2. The system displays a list of the user's friends, sorted alphabetically, in the newly opened menu.
3. The user selects one or more of their friends from the list and is asked to confirm their choices.
4. The system adds a notification to the selected friend(s)' notification feeds containing the details of the shared recipe and the user who shared it with them.
5. Upon next opening the application, users who have been given the notification will be prompted to check their notification feed.
6. Each friend is given the option to view, save, and manipulate any shared recipe(s) within their notification feed as they would with any other recipe they encounter through the app.
7. The shared recipe will remain in the friend's notification feed for one week before the notification expires and is automatically removed.

Alternatives:

1. In step 1: The prompted user may decide not to share the recipe with their friends. In this case, the user can simply exit the pop-up menu and Steps 2-5 are ignored.
2. In step 6: Friends who have received a recipe may choose not to interact with the recipe in any way. In this case, the friend may simply dismiss the notification from their feed.

3.4.0 UC-4: Add foods' Expiration Date

Precondition: User is logged in on the app and wants to add the expiration date of foods.

Actor: User

Trigger: User wants to use foods before they expire

Basic Flow:

1. User visits the app's calendar
2. The app displays the expiration date list with other options for the user to choose
3. The user chooses the expiration date list
4. The app displays foods on the expiration date list and the option to add foods
5. The user chooses the add option
6. The app displays options of add by scanning the expiration date on the food or manually
7. The user chooses to add by scanning the expiration date on the food

8. The app opens the camera and starts scanning
9. The user can see the expiration date of the food being added in the list.

Alternatives:

1. In step 6: The user may want to enter the expiration date manually. In that case, step 7 and 8 will be ignored.
2. In step 7: The user may need to allow the app to use the user's device camera if it was not allowed before.
3. In step 8: If the app can't scan the expiration date, the user needs to enter it manually.

3.5.0 UC-5: View Nutritional Content

Precondition: User is logged into the app, and wants to know the nutritional value of a meal.

Actor: User

Trigger: User is concerned about their diet

Basic Flow:

1. User visits the app's calendar
2. The system shows the user the meals on their calendar
3. User chooses a meal they have planned on their schedule
4. The system provides the user with a menu of actions regarding the meal
5. User chooses to view the meal's nutritional content.
6. The system shows the user the nutritional content.

Alternatives:

1. In step 1, the user may select a meal from Cooking History, instead of a meal that is yet to be made.
2. In step 1, the user may select a meal from another user's cookbook, instead of a meal in their own calendar.
3. User can wait for the program to send them a warning when their diet has a potentially unhealthy distribution of nutrition.

3.6.0 UC-6: Register Kitchen

Precondition: User is logged into the app, and wants to register which appliances and ingredients are present in their kitchen so that they can check which recipes they have everything needed to make.

Actor: User

Trigger: User wants to know which recipes they can and can't make with what they have

Basic Flow:

1. User visits the app's Kitchen section
2. User selects option to create a saved kitchen on their account
3. User is prompted to add appliances to this new kitchen or to skip to the next step
4. User is prompted to add ingredients to their pantry, and each ingredient added prompts the user to optionally add the expiration date
5. User selects next option to complete their kitchen creation
6. User is prompted to enter a nickname for their kitchen or finish with the default "kitchen" + "n", where n is number of kitchens saved to the user's account

Alternatives:

1. In step 6, the user may select an option to share the specs of their newly created kitchen to their followers or on social media.

3.7.0 UC-7: Login

Precondition: User registered an account and has app opened

Actor: User

Trigger: User selects 'log in'

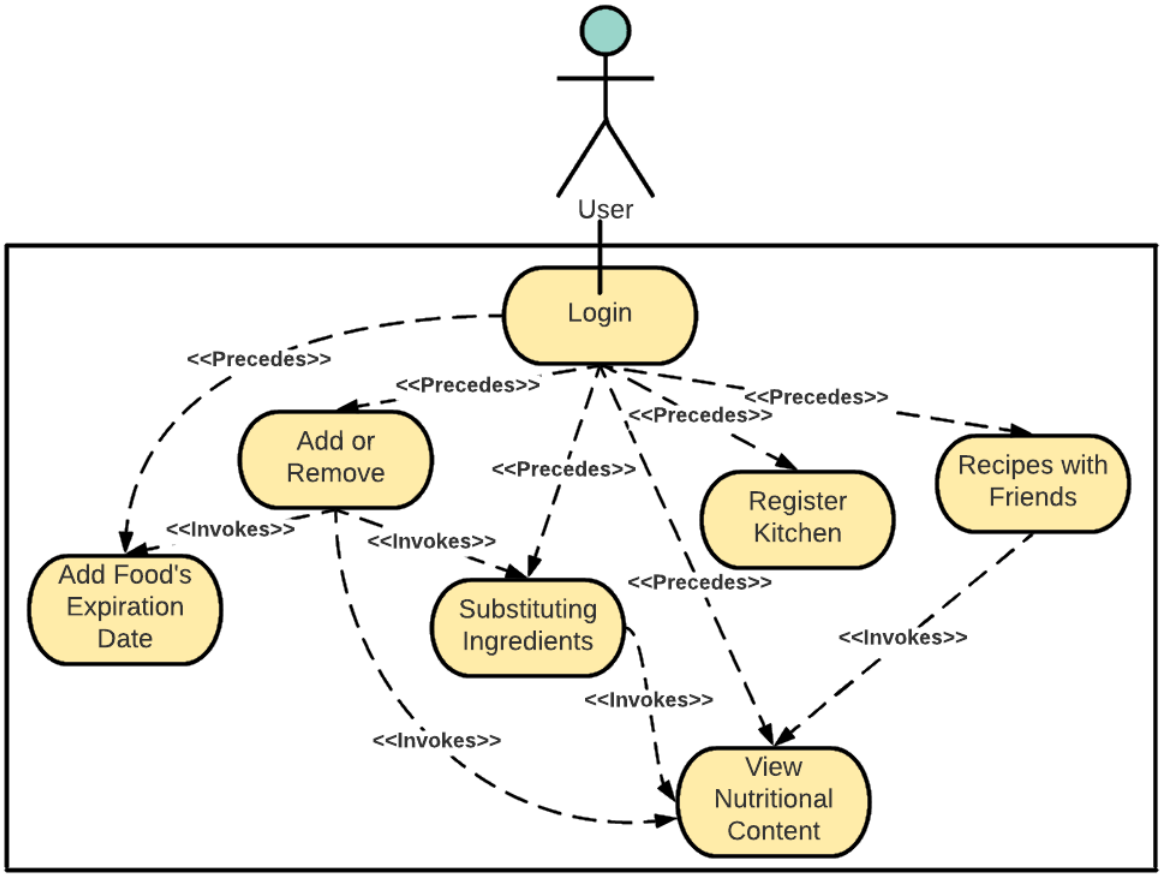
Basic Flow:

1. User enters their username in the first field.
2. User enters their password in the second field.
3. User finalizes the login process with a Submit button.
4. System verifies information entered.
5. System accepts or denies login based on verification result.

Alternatives:

1. In step 1: User can enter their email address instead of their username.

4.0 Use Case Diagram



5.0 Functional Requirements

5.1.0 End User Requirements

Requirement Id	Requirement description
EUR-FR-01	The user should be able to set up an account by using their Google login.
EUR-FR-02	The user should be able to reset their account password.

EUR-FR-03	The user should be able to recover their account information.
EUR-FR-04	The user should be able to create eating plans using recipes.
EUR-FR-05	The user should be able to create profiles for each recipe that they have. Include the number of ingredients served (cheese, tomato, lettuce, and salt), and the number of chlorides.
EUR-FR-06	The user shall be able search and filter for specific recipes. Search for related keywords based on the recipe name or ingredients used.
EUR-FR-07	The user shall be able to add or remove specific recipes. Remove for related keywords based on the recipe name or ingredients used.
EUR-FR-08	The user shall be able to view information from previously used recipes.
EUR-FR-09	The user should be able to add friends to their account.
EUR-FR-10	The user should be able to comment on other people's profiles.
EUR-FR-11	The user should be able to view messages from other profiles in their inbox.
EUR-FR-12	The user should be able to send a message to other profiles.
EUR-FR-13	The user should be able to revise their profile information.

5.2.0 System Requirements

Requirement Id	**Requirement Description**
SR-FR-01	The system should be able to access data and update the data.
SR-FR-02	The system shall process all user data, save it as a User Profile, and store it in the cloud.
SR-FR-03	The system should accept and relay user feedback, as well as save data in a cloud database.

SR-FR-04	The system should be able to access the data of time and location of the user.
SR-FR-05	The system's data should be encrypted.

6.0 Non-Functional Requirements

Usability: The application interface should be user-friendly and easy to use for any smartphone user so that the users continue to use it after initial download.

Reliability: The application should properly execute operations performed by the user and it should work as expected by the user.

Performance: The application should be able to avoid downtime errors, and requests should be processed 5 "9s"(99.999 percent of the time).

Availability: The service should be available as much as possible, limiting the downtime caused by servers being down and maintenance being performed. Maintenance should be planned to limit the amount of users that are affected by the outage.

Security: The system should be accessible through plugging in an account username and password. Passwords should be stored securely.

Maintainability: The application should be maintainable through future software and hardware updates that come with newer cell phones, and servers should be maintainable as more and more user accounts are created.

Interoperability: The application should be available on a multitude of mobile platforms so many users can use it regardless of what type of phone they own.

6.1.0 Usability

Requirement Id	Requirement Description
U-NFR-01	The system should be built with an intuitive GUI.
U-NFR-02	All text should be legible.

6.2.0 Reliability

Requirement Id	Requirement Description
R-NFR-01	Nothing in the user's settings or saved recipes should change without the user's approval.
R-NFR-02	The system shall notify the user in the event that their data is not saved correctly, so they may try an operation again.

6.3.0 Performance

Requirement Id	Requirement Description
P-NFR-01	The system should be able to locate a user's profile within two seconds of searching.

6.4.0 Availability

Requirement Id	Requirement Description
A-NFR-01	The system should be online 99.99% of the time.
A-NFR-02	The system should not be designed to require periodic restarts.

6.5.0 Security

Requirement Id	Requirement Description
S-NFR-01	User's passwords should be required to be sufficiently complex enough to deter hackers from guessing it.
S-NFR-02	Users should be able to make their profiles private, preventing others from seeing their account.

6.6.0 Maintainability

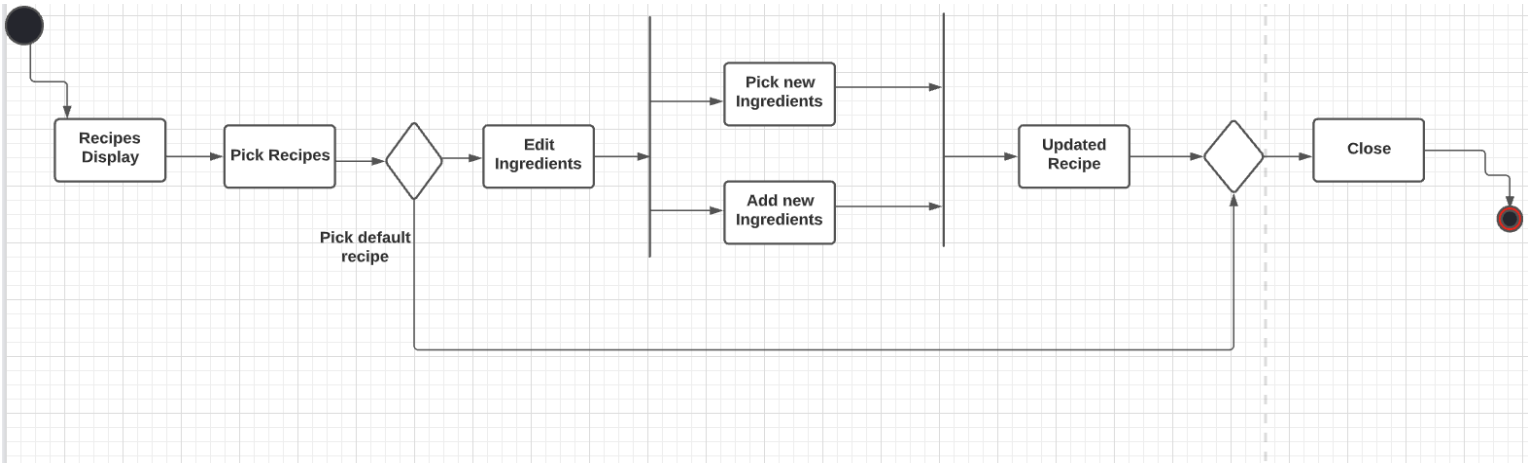
Requirement Id	Requirement Description
M-NFR-01	Code should be frequently refactored in order to make it easier to update and maintain.
M-NFR-02	Application function shall be tested on newer operating system versions prior to an update.

6.7.0 Interoperability

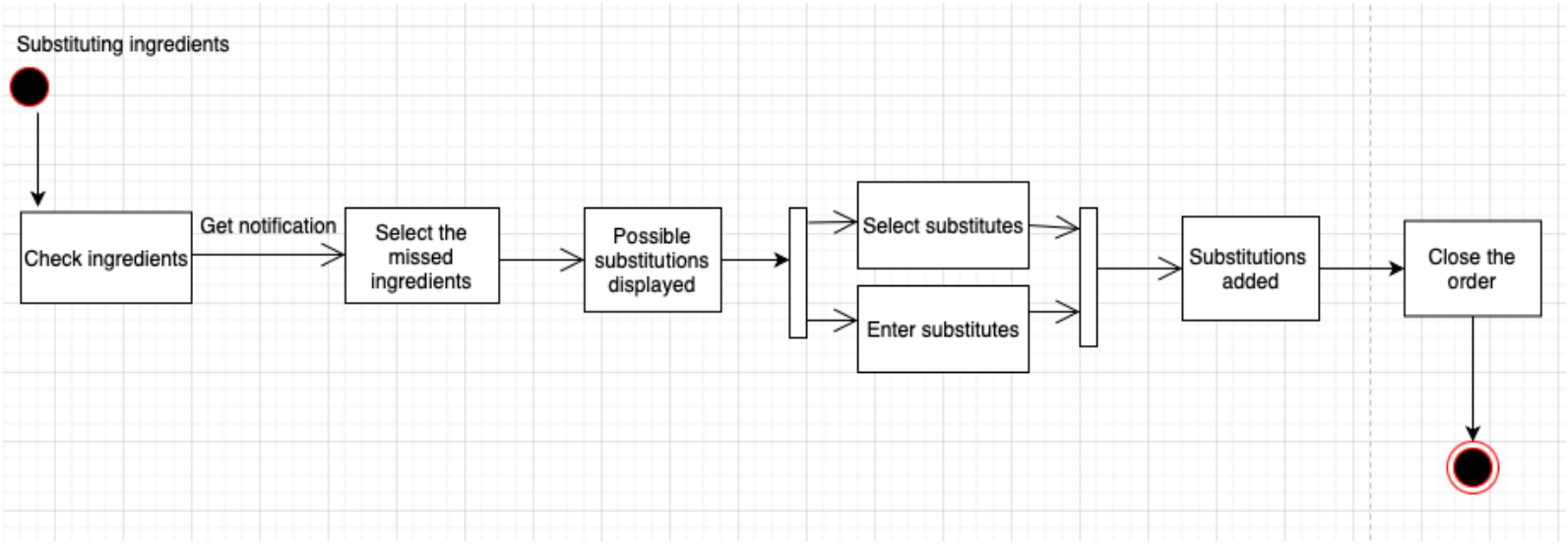
Requirement Id	Requirement Description
I-NFR-01	The application shall be available on Android and Apple mobile devices
I-NFR-02	The application shall be available on all devices with an OS that is no more than 1 year out of date

7.0 Activity Diagrams

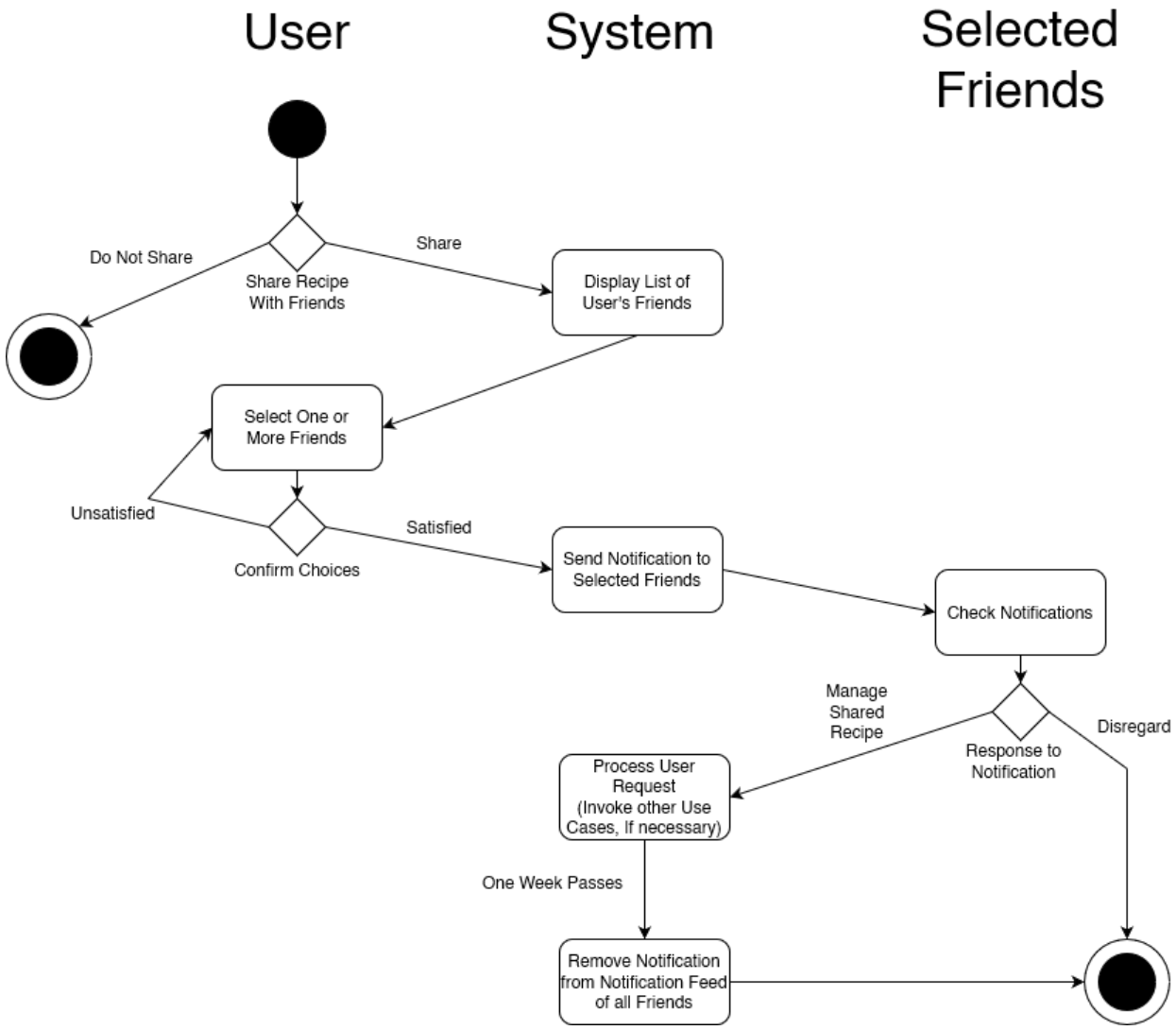
7.1.0 UC-1: Adding Ingredients



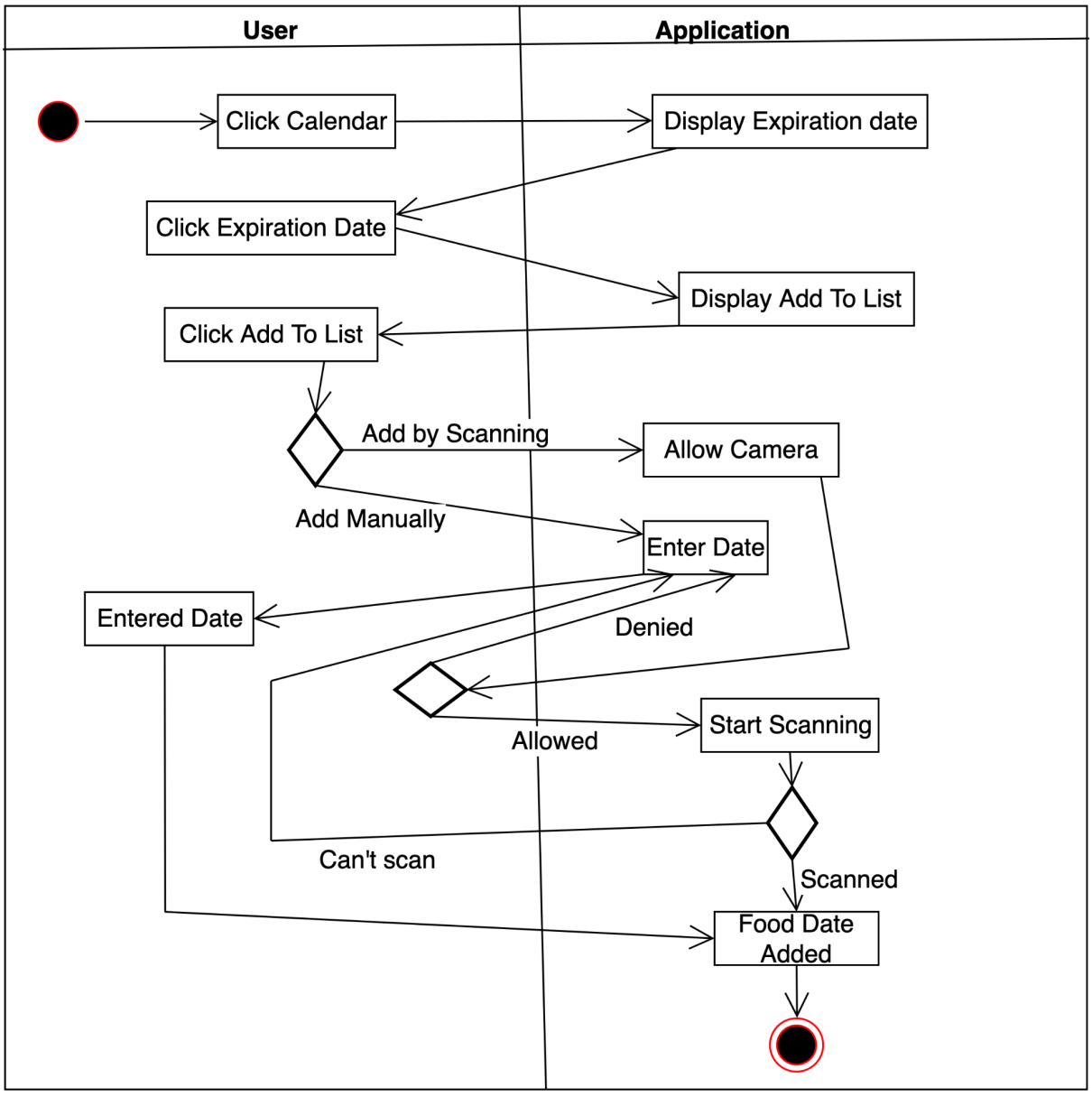
7.2.0 UC-2: Substituting Ingredients



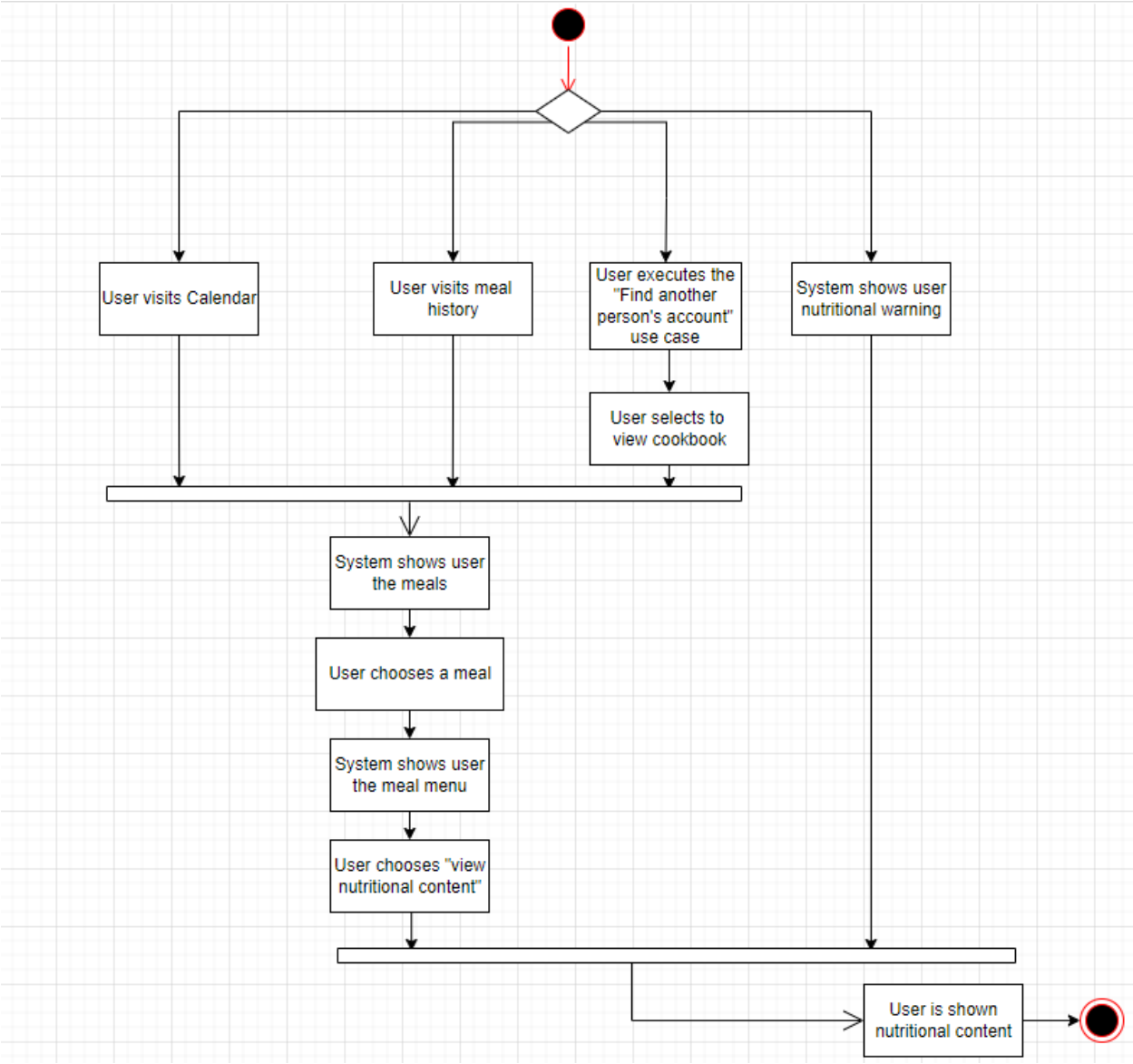
7.3.0 UC-3: Recipes with Friends



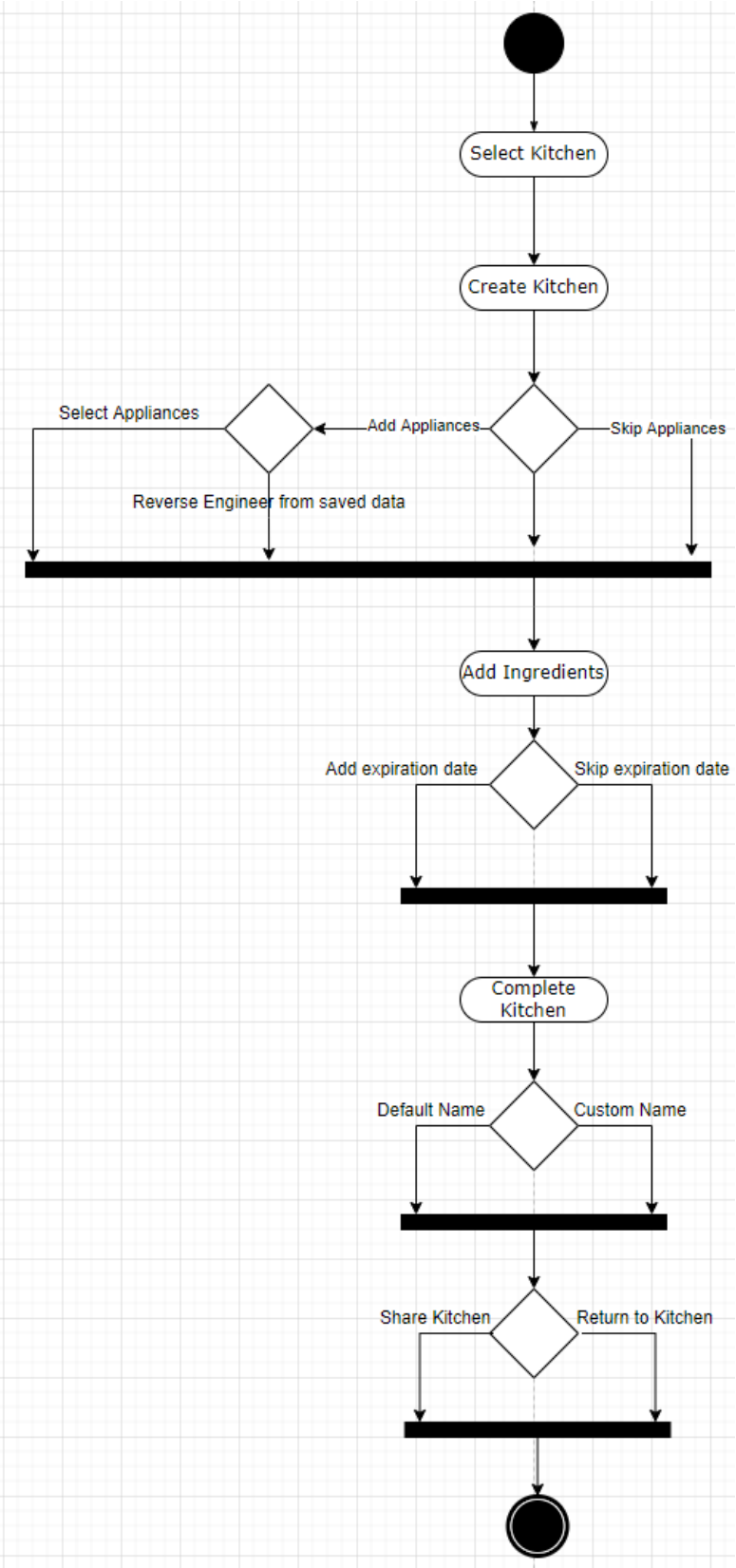
7.4.0 UC-4: Add Food's Expiration Date



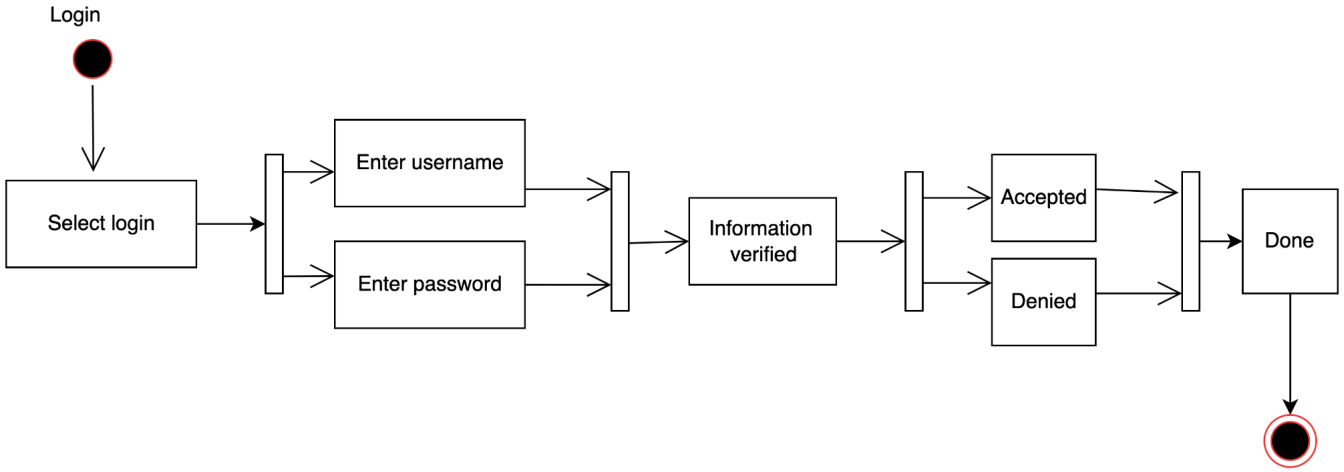
7.5.0 UC-5: View Nutritional Content



7.6.0 UC-6: Register Kitchen

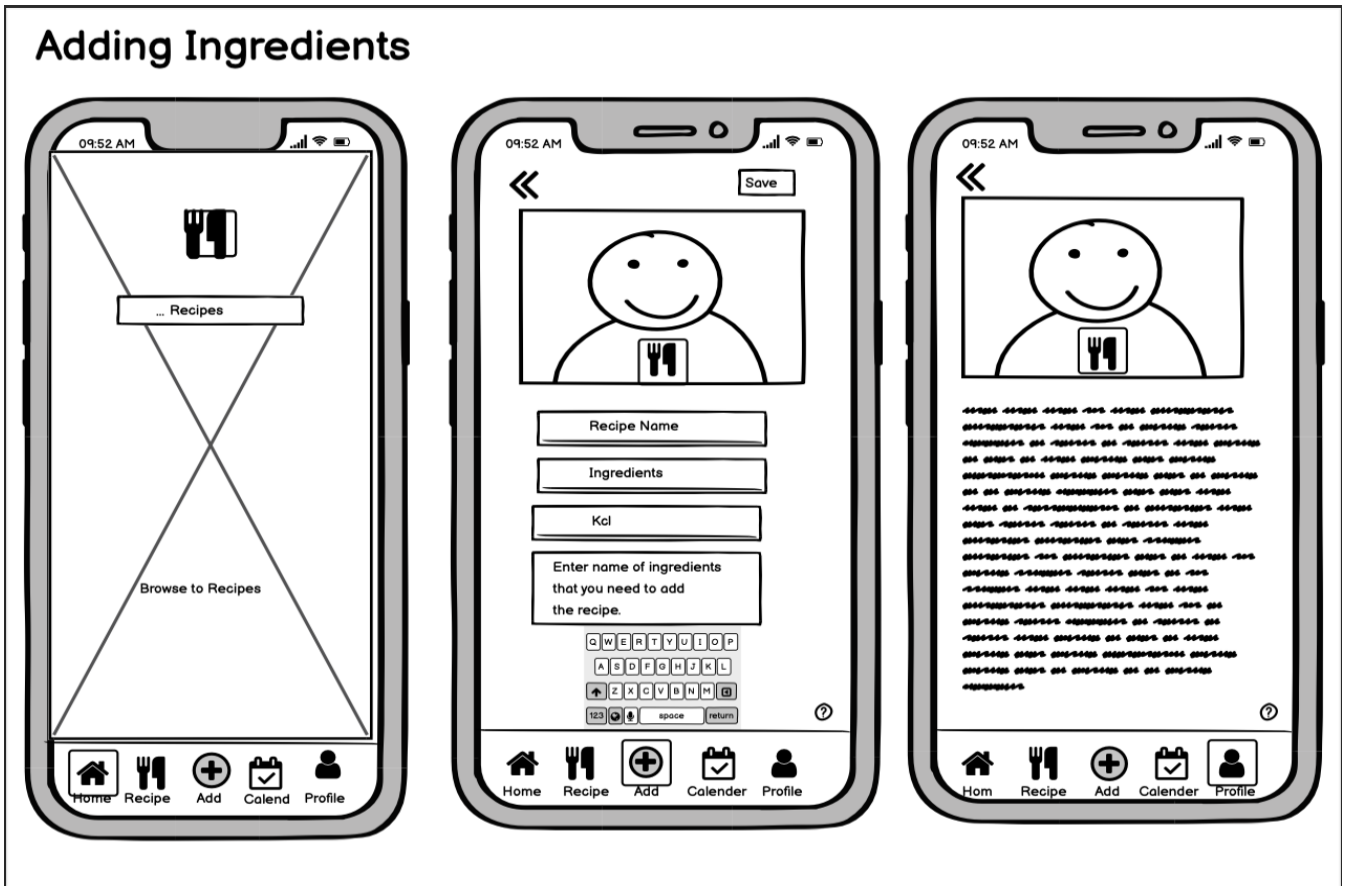


7.7.0 UC-7: Login



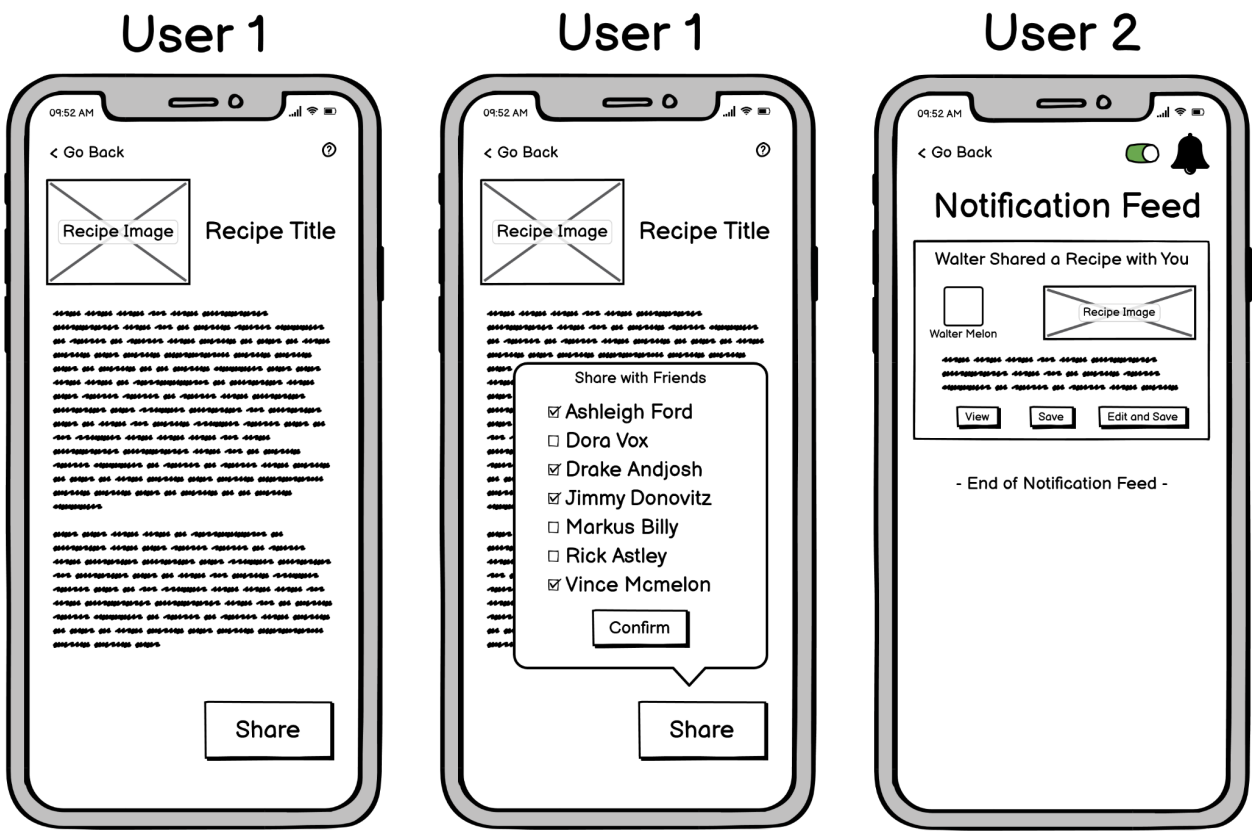
8.0 Wireframes

8.1.0 UC-1: Adding Ingredients

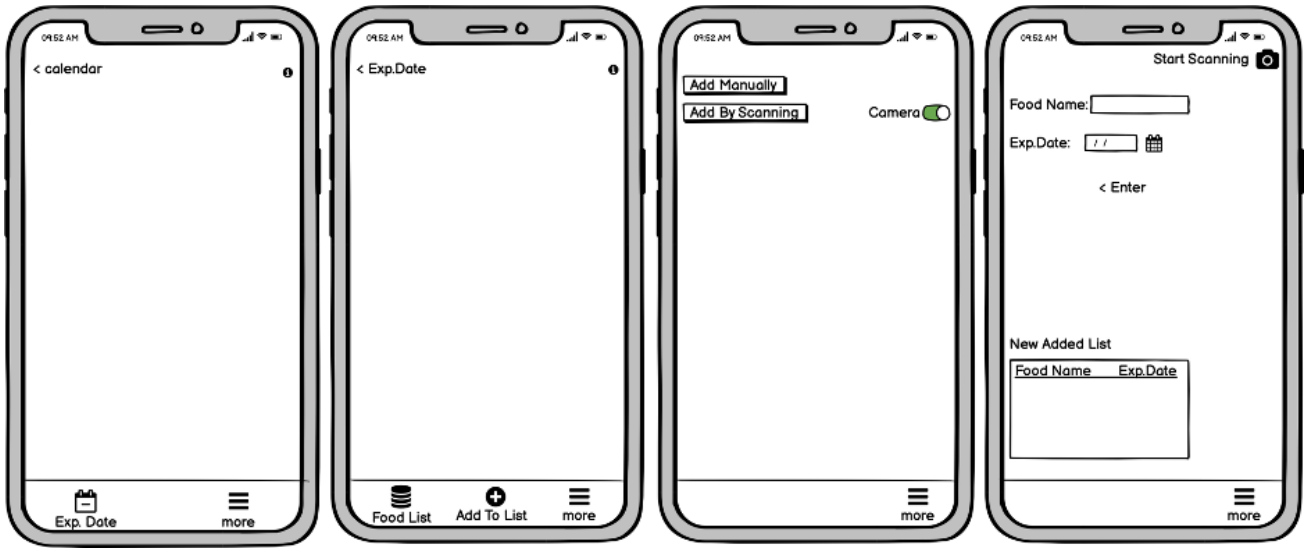


8.2.0 UC-2: Substituting Ingredients

8.3.0 UC-3: Recipes with Friends

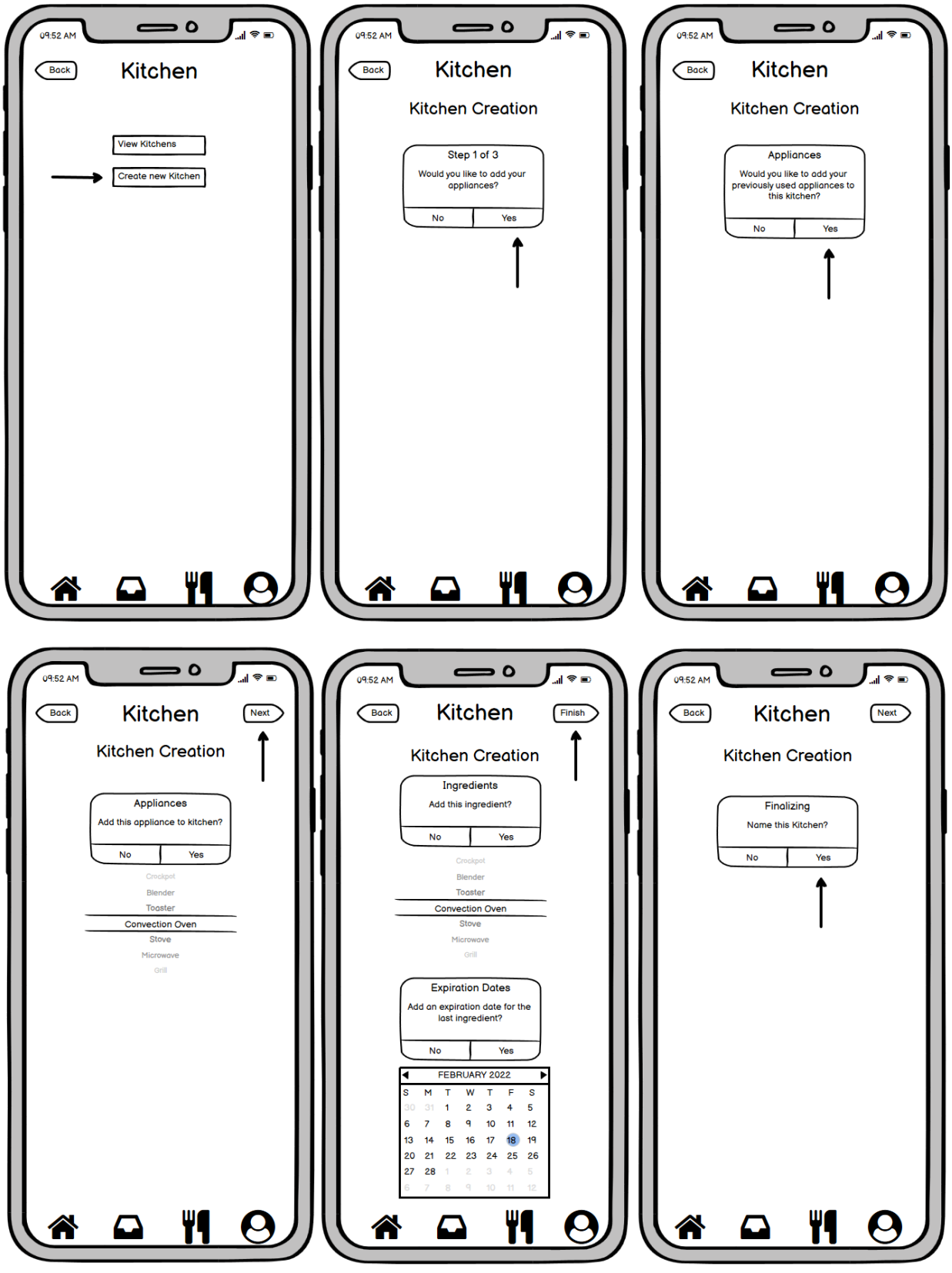


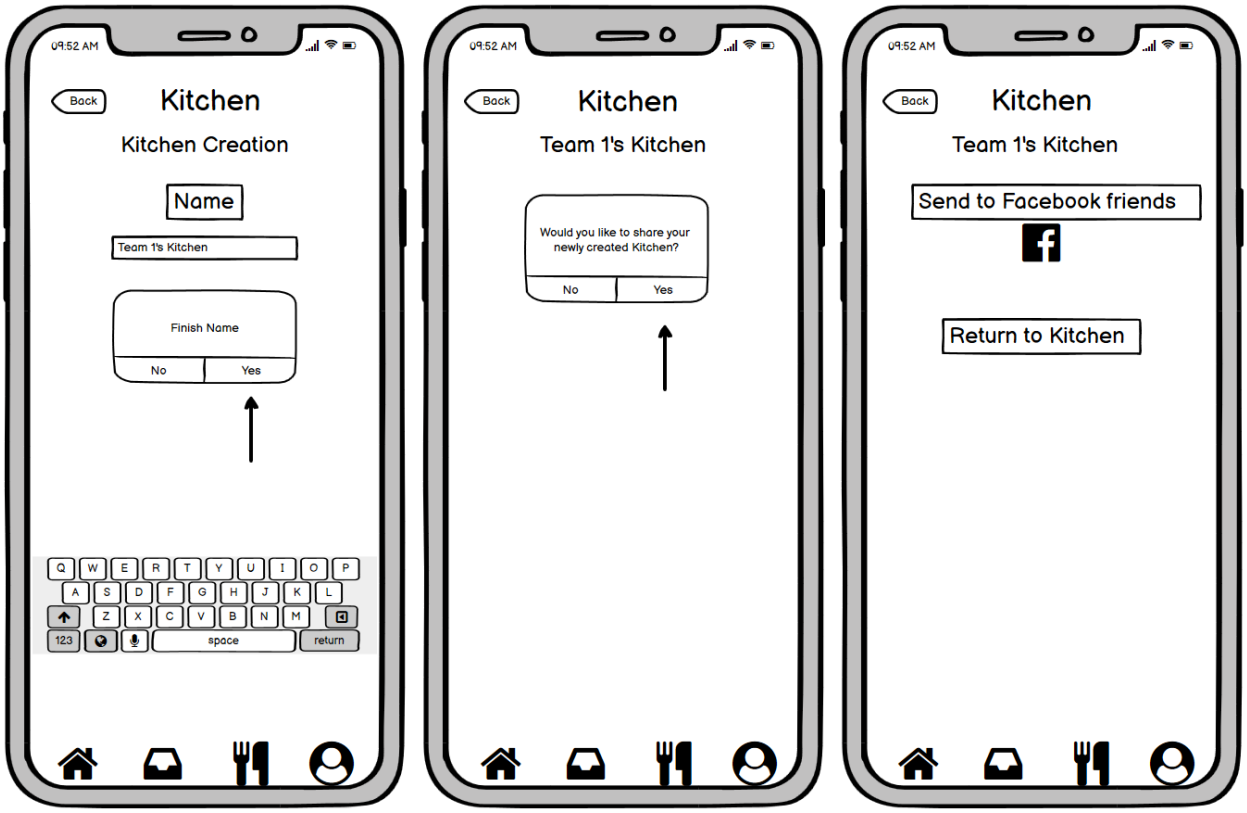
8.4.0 UC-4: Add Food's Expiration Date



8.5.0 UC-5: View Nutritional Content

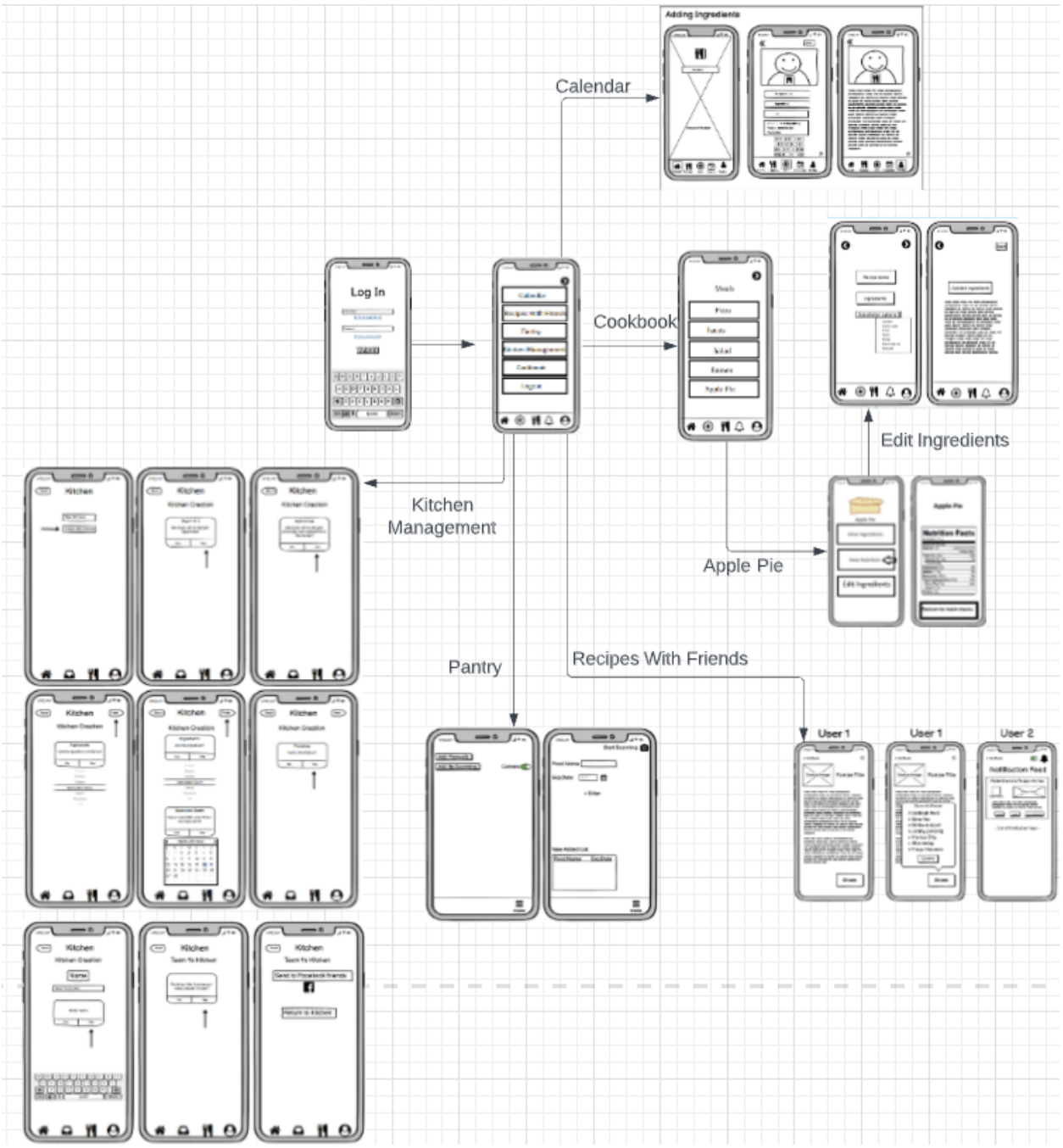
8.6.0 UC-6: Register Kitchen




09:52 AM
Back
Kitchen
Kitchen Creation
Name
Team 1's Kitchen
Finish Name
No
Yes
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
return
09:52 AM
Back
Kitchen
Team 1's Kitchen
Would you like to share your newly created Kitchen?
No
Yes
09:52 AM
Back
Kitchen
Team 1's Kitchen
Send to Facebook friends
Return to Kitchen

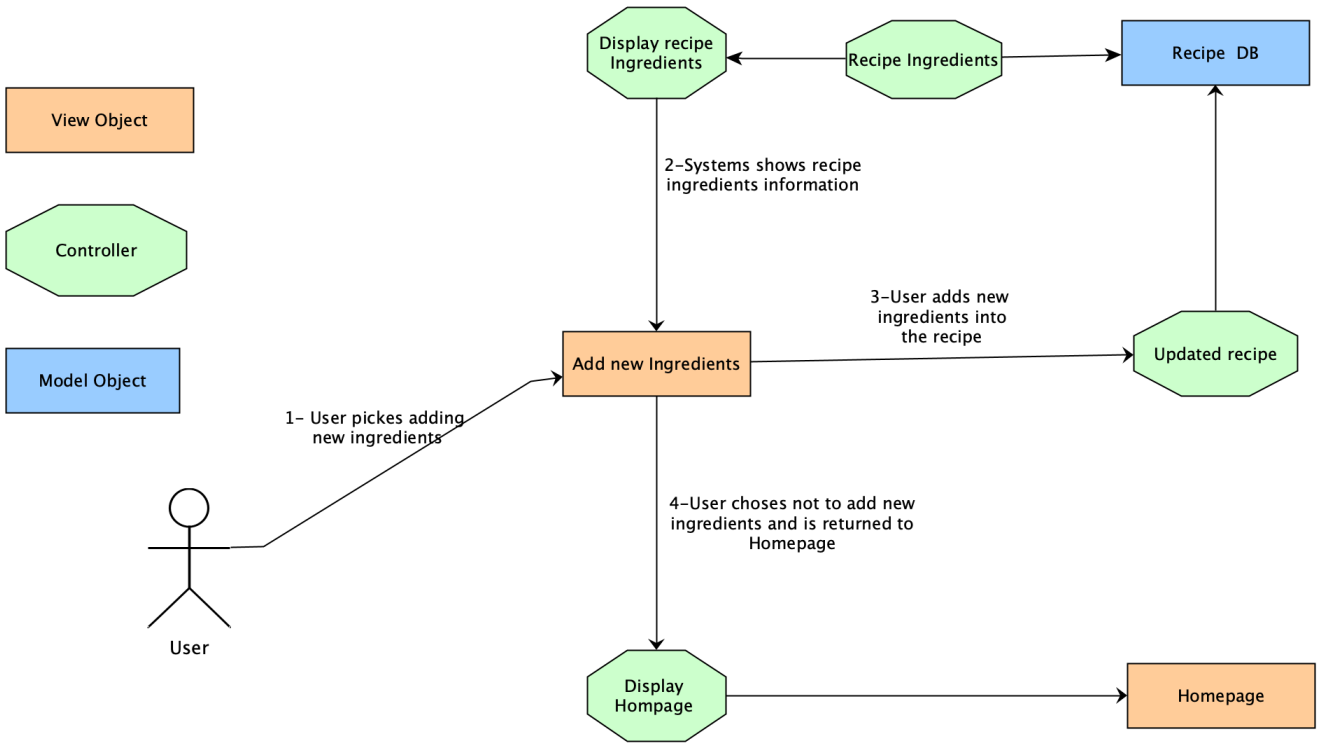
8.7.0 UC-7: Login

8.8.0 Use Case Wireframes Connected

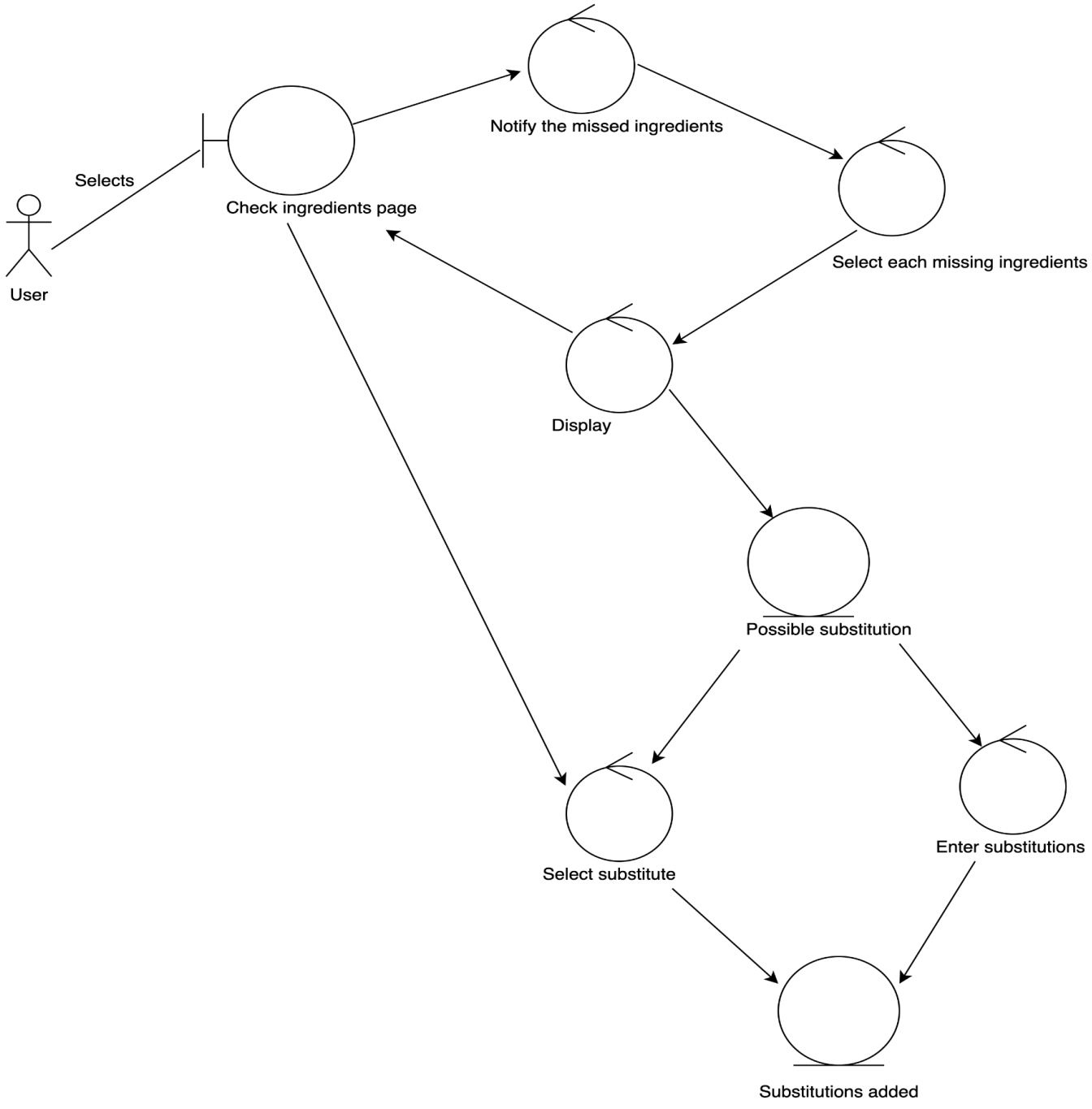


9.0 Robustness Diagrams

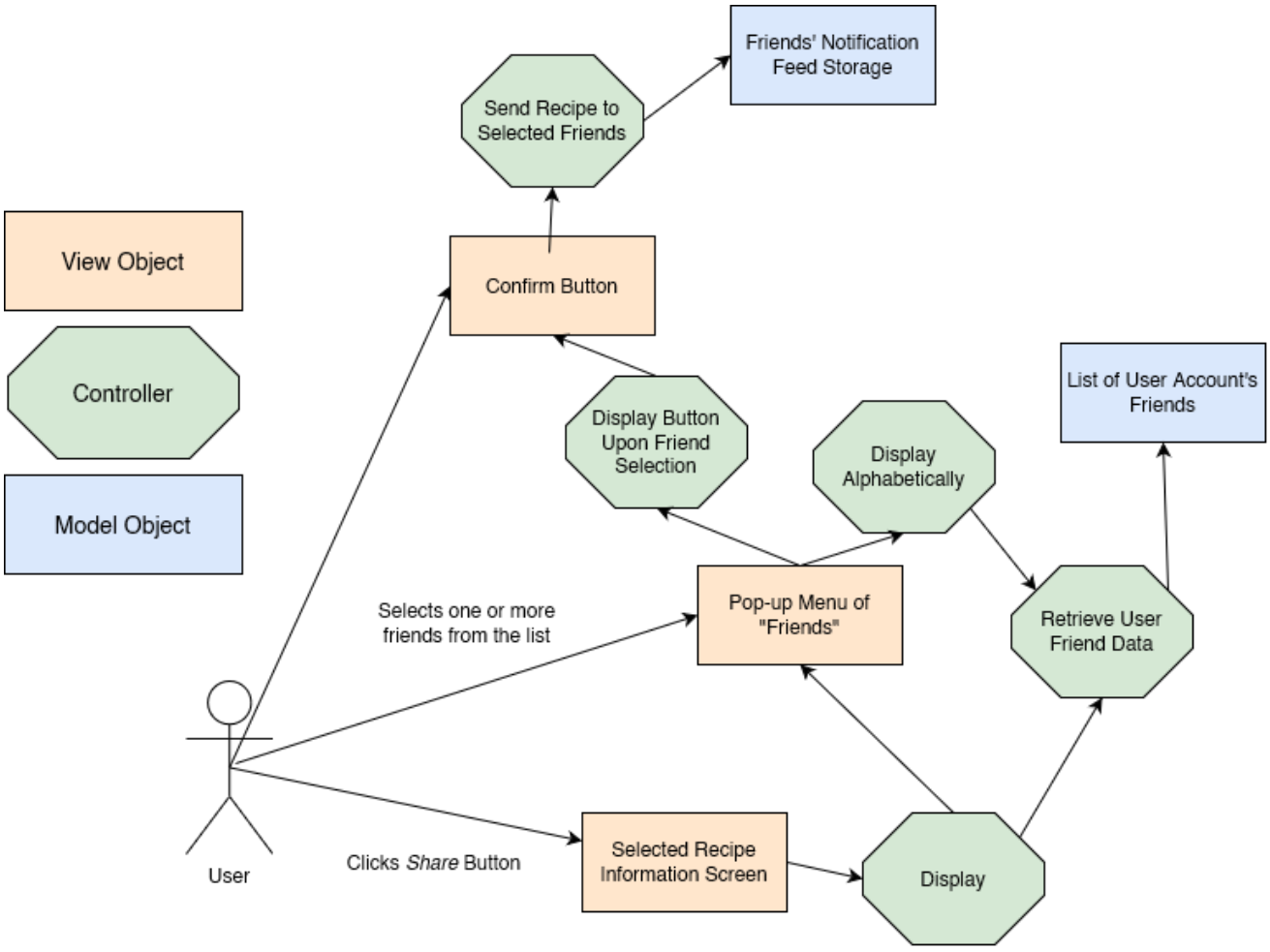
9.1.0 UC-1: Adding Ingredients



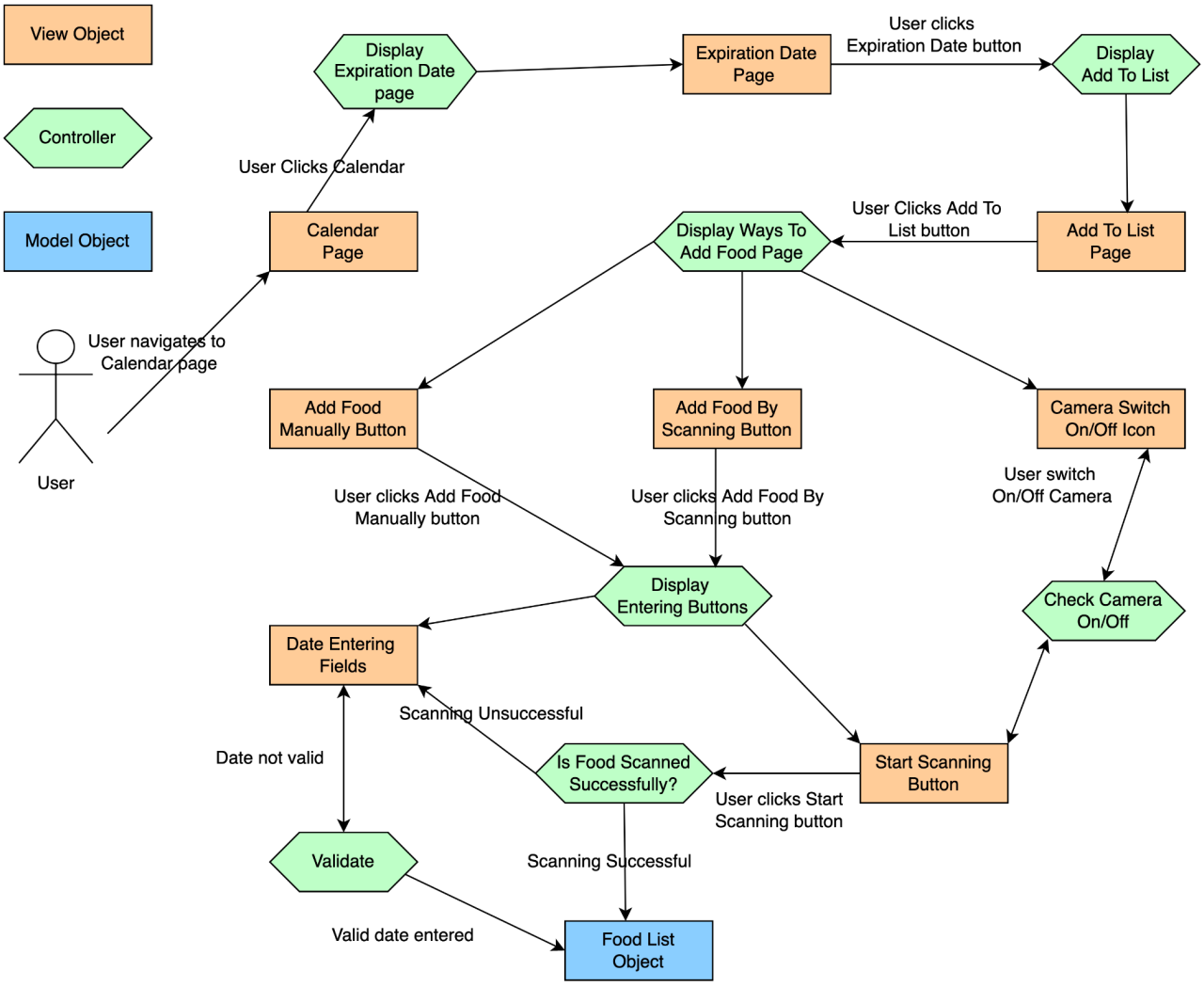
9.2.0 UC- 2: Substituting Ingredients



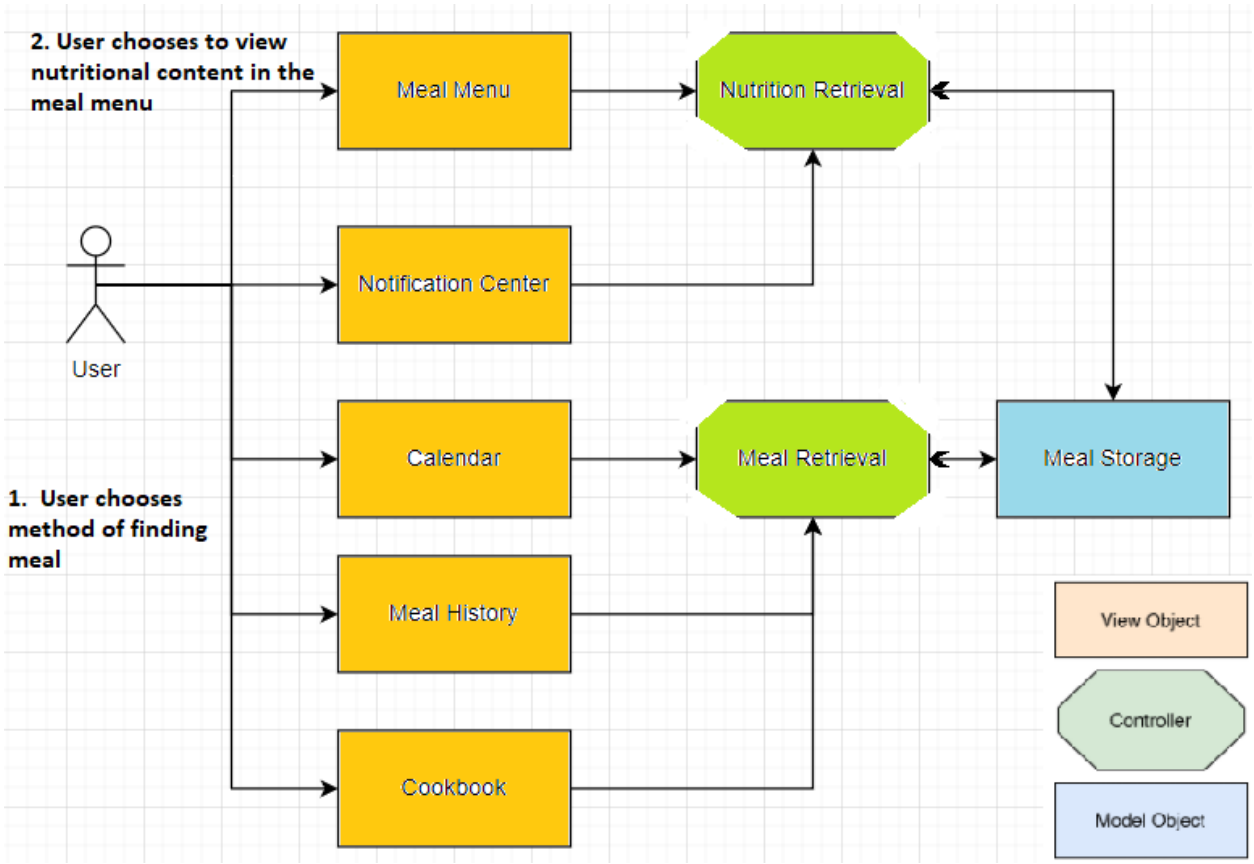
9.3.0 UC- 3: Recipes with Friends



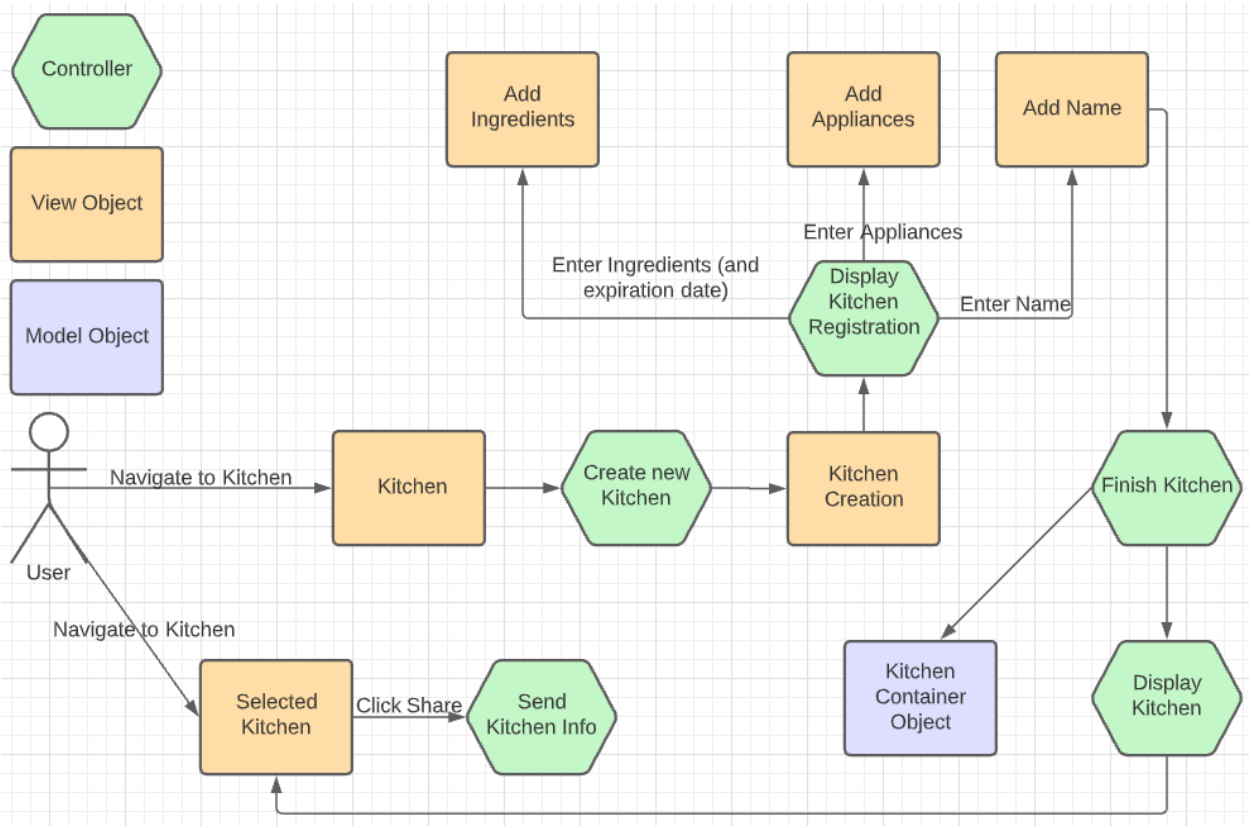
9.4.0 UC- 4: Add Food's Expiration Date



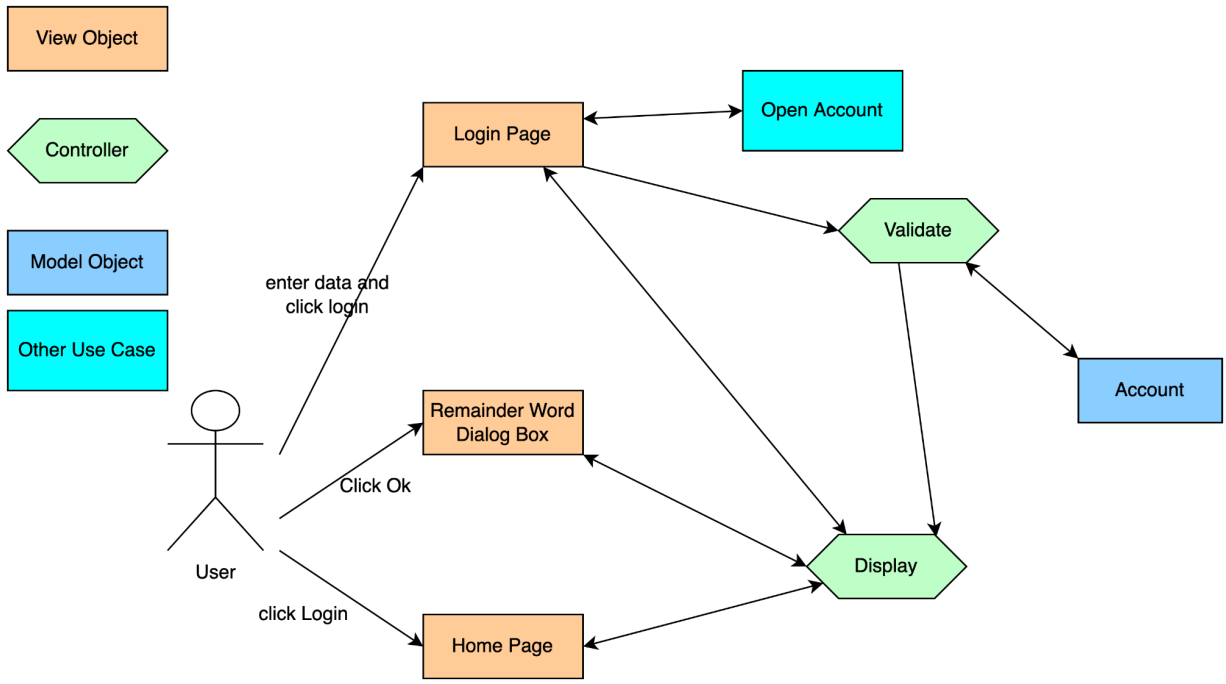
9.5.0 UC- 5: View Nutritional Content



9.6.0 UC- 6: Register Kitchen

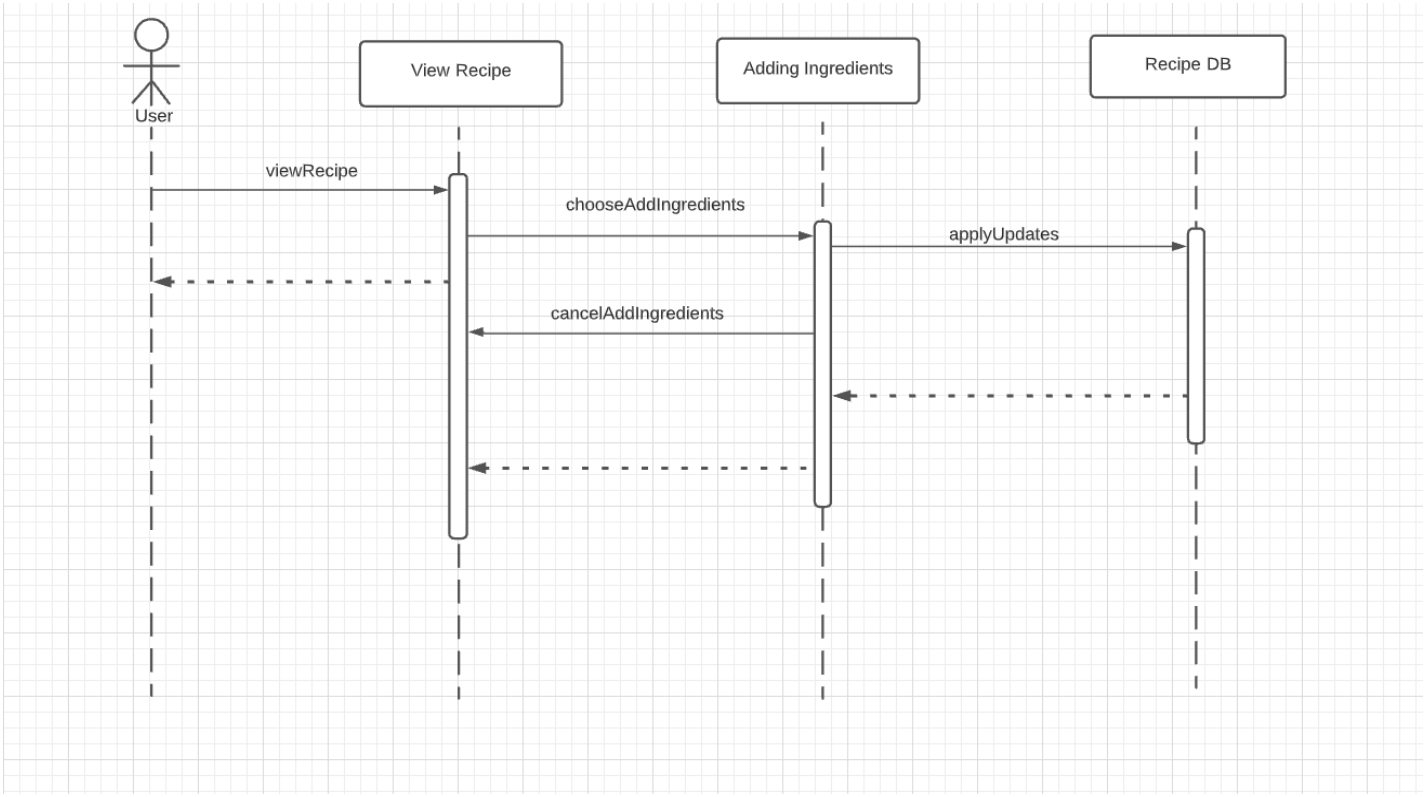


9.7.0 UC- 7: Login



10.0 Sequence Diagrams

10.1.0 UC-1: Adding Ingredients



10.2.0 UC-2: Substituting Ingredients

Basic flow:

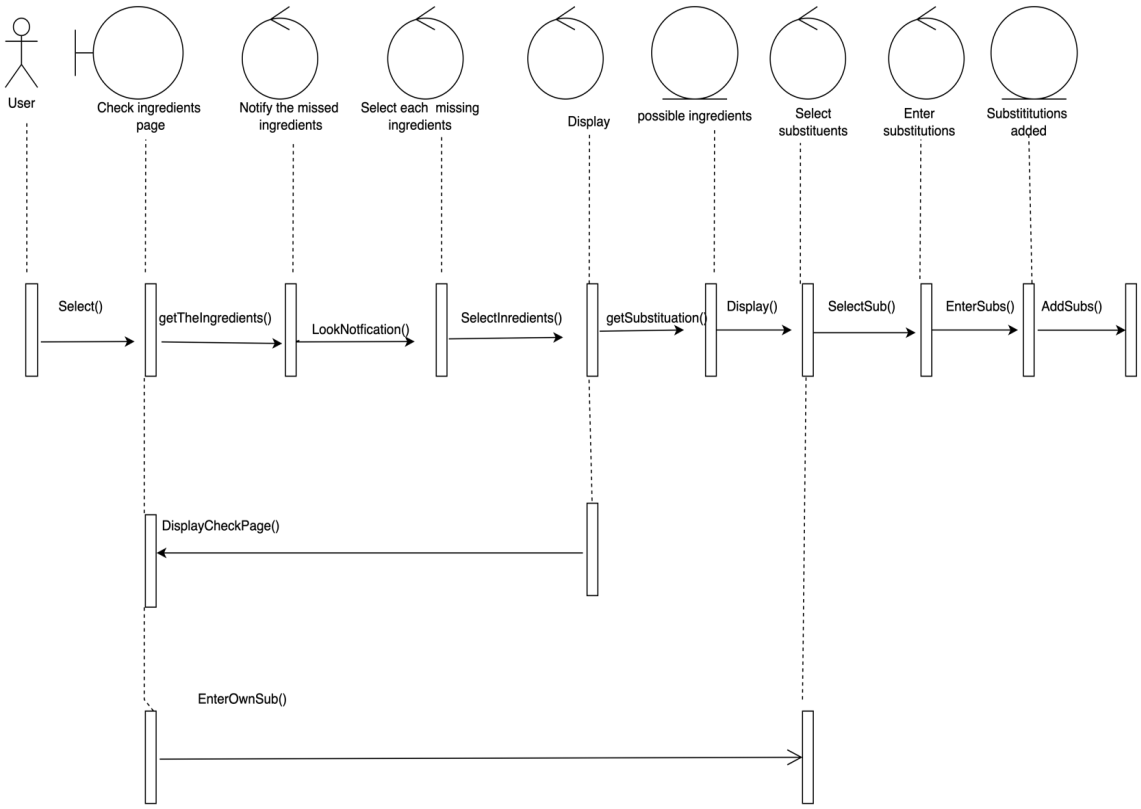
The user selects '**check ingredients**' for their chosen recipe. The system notifies the user of **ingredients they are missing**. User selects each **missing ingredient** they want to replace.

The system displays possible **substitutions** for the selected ingredients. User selects the substituted ingredients they want and selects '**substitute**'.

The system adds the **substitutions** to the recipe

Alternatives:

The user can enter their own **ingredient substitutions** instead of the substitutions offered by the recipe



10.3.0 UC-3: Recipes with Friends

View Object

Model Object

Basic Flow:

One of a **User**'s friends is viewing a recipe and presses a "Share" button, opening up a **Pop-up Menu**.The system displays a list of the **User**'s **Friends**, sorted alphabetically, in the newly opened **Menu**.

The **User** selects one or more of their **Friend**s from the list and is asked to **Confirm** their choices. The system adds a notification to the selected **Friend**(s)' **Notification Feed**s containing the details of the shared recipe and the **User** who shared it with them.

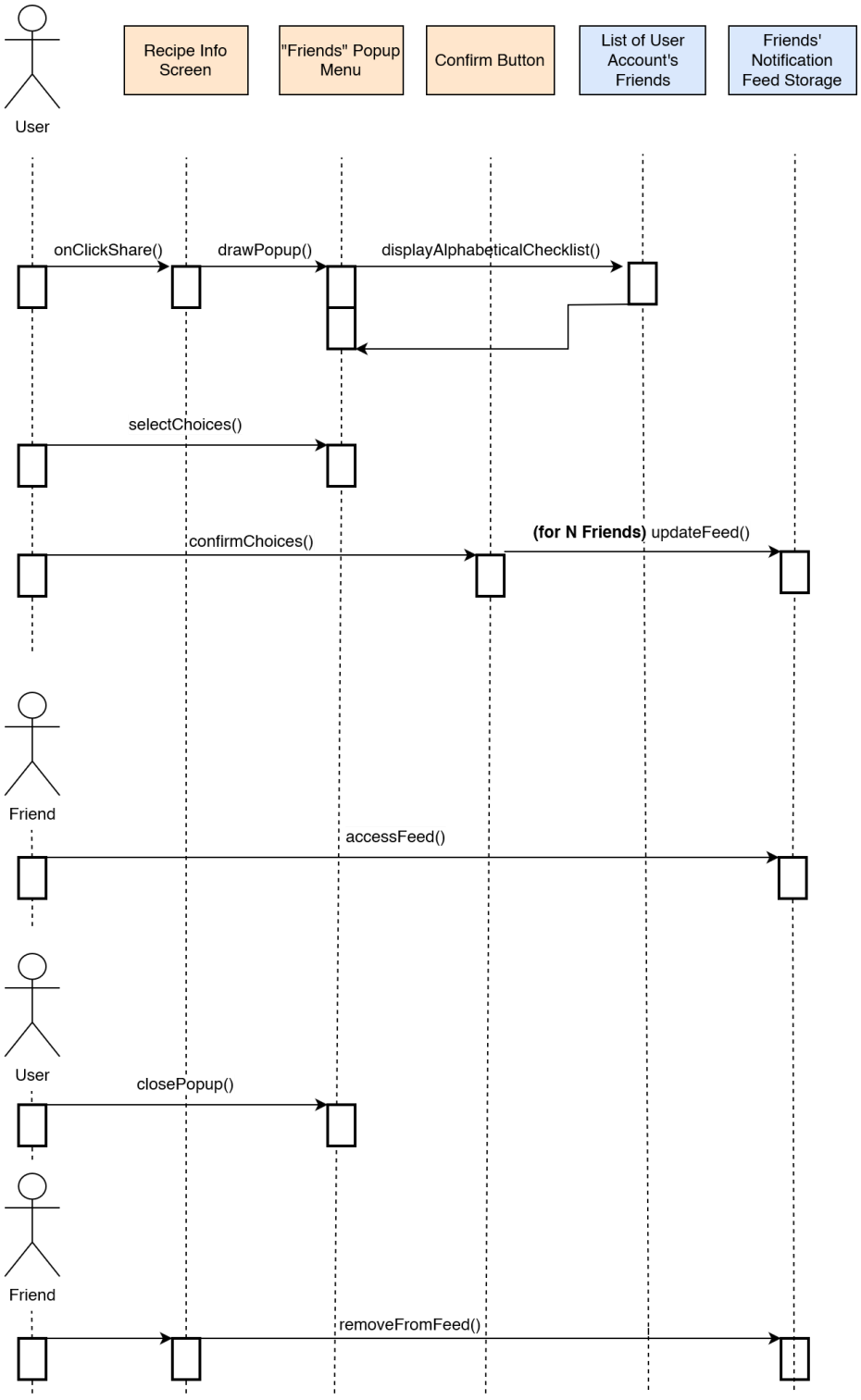
Upon next opening the application, **Friend**s who have been given the notification will be prompted to check their **Notification Feed**. Each **Friend** is given the option to view, save, and manipulate any shared recipe(s) within their **Notification Feed** as they would with any other recipe they encounter through the app. The shared recipe will remain in the **User**'s **Notification Feed** for one week before the notification expires and is automatically removed.

Alternate Course:

The prompted **User** may decide not to share the recipe with their **Friend**s. In this case, the **User** can simply exit the **Pop-up Menu.**

Alternate Course:

Friends who have received a recipe may choose not to interact with the recipe in any way. In this case, the friend may simply dismiss the notification from their **Feed**.



10.4.0 UC-4: Add Food's Expiration Date

View Object

Model Object

Basic Flow

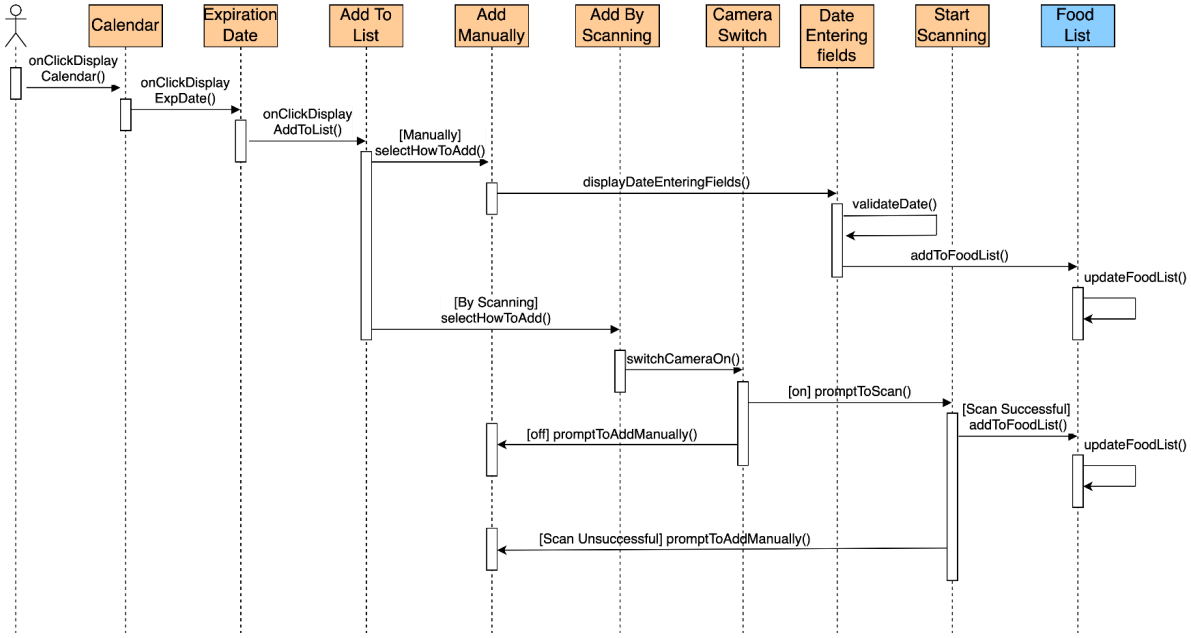
The **user** has purchased some food and is login to add its expiration date in the recipe planner app. After navigating to the calendar page, the user clicks the **Calendar** button and the system displays the **Expiration Date** button. Once the user clicks the expiration date button, the system displays the **Add To List** button and asks the user to choose how to add. The user chooses to add **By Scanning** and is asked to to **Switch Camera**. Once the camera is switched on, the system opens the camera and **Starts To Scan**. Once the food's date is scanned, the system updates the **Food List**.

Alternative Course 1

The user may choose to **Add Manually** on **Add To List** page. In that case the system displays the **Date Entering Fields** and adds the food once the date entered is validated.

Alternative Course 2

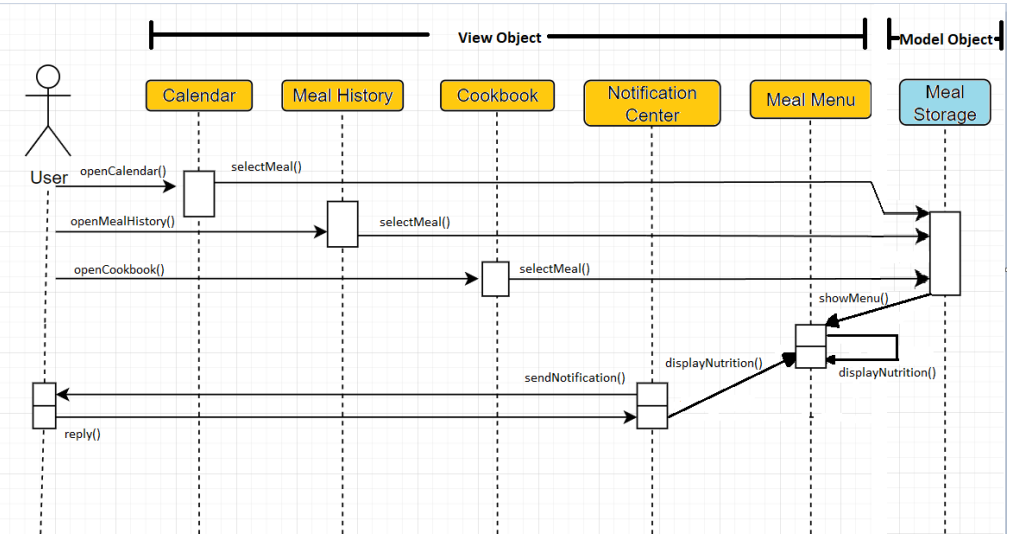
The system also prompts the user to **Add Manually** if the camera is not allowed or the can't scan the food's date.



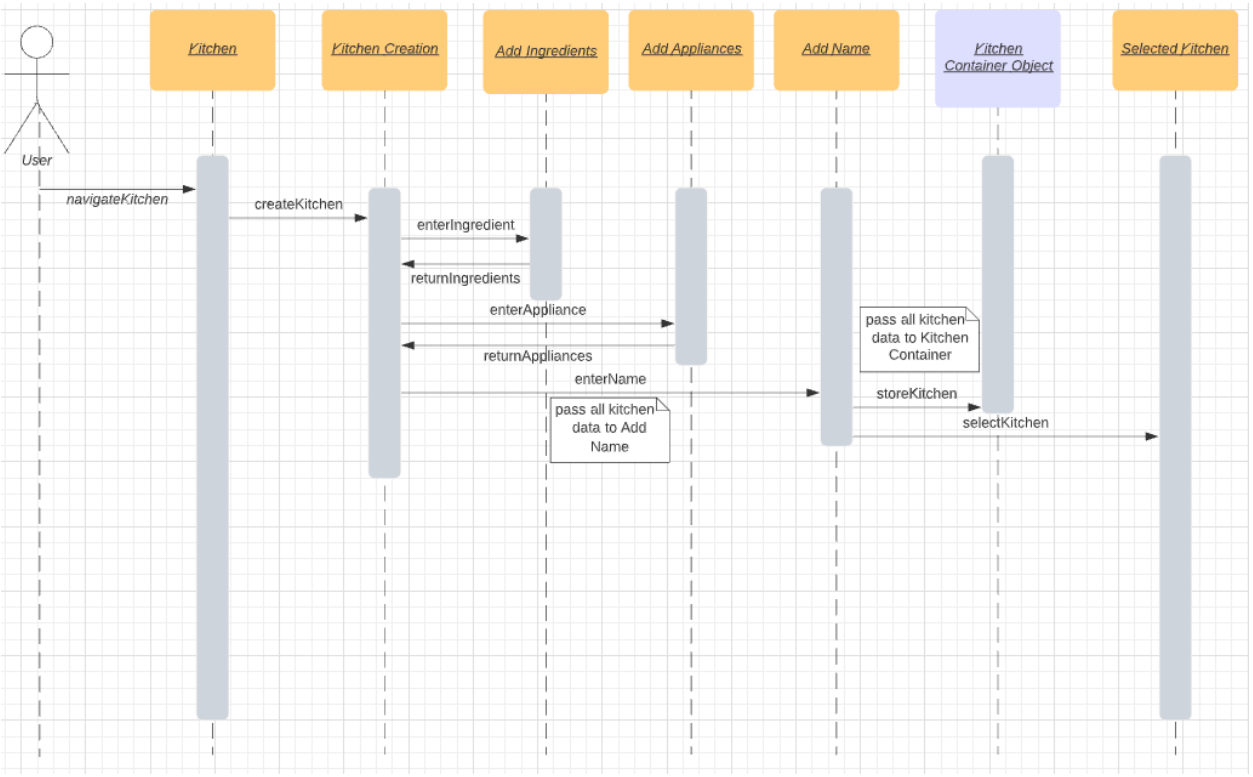
10.5.0 UC-5: View Nutritional Content

Basic Flow

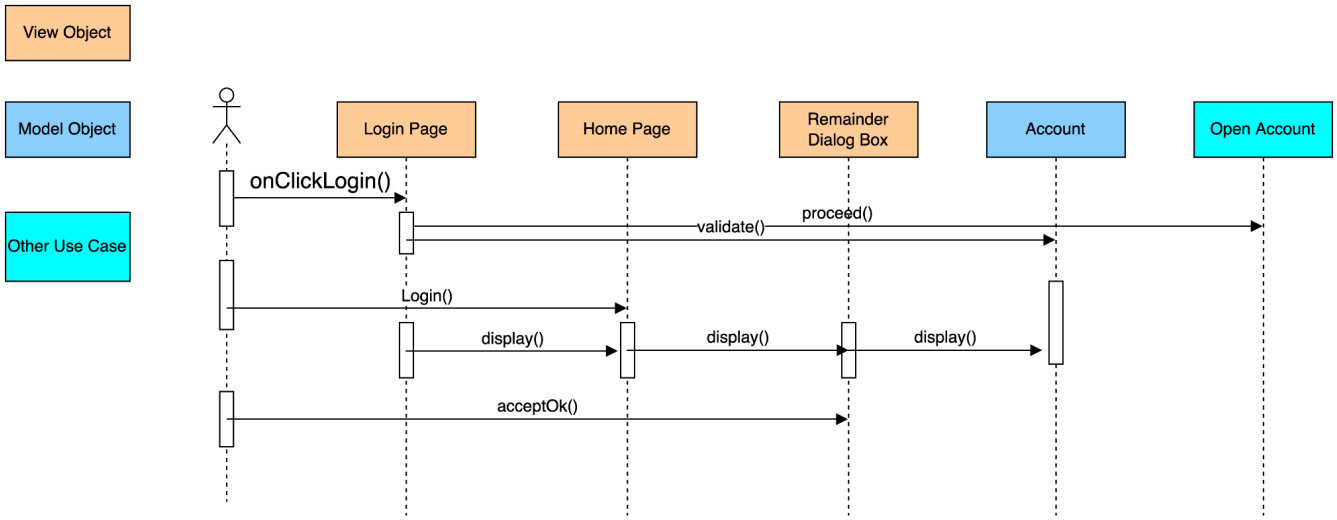
The **User** chooses a means to access a meal, that being either the **Calendar**, the **Meal History**, the **Cookbook**, or they are notified directly from the **Notification Center**. The meal is then retrieved from **Meal Storage** and given to **Meal Menu**. From **Meal Menu**, the **User** selects to view nutritional content, and displayNutrition() displays the nutrition retrieved from **Meal Storage**.



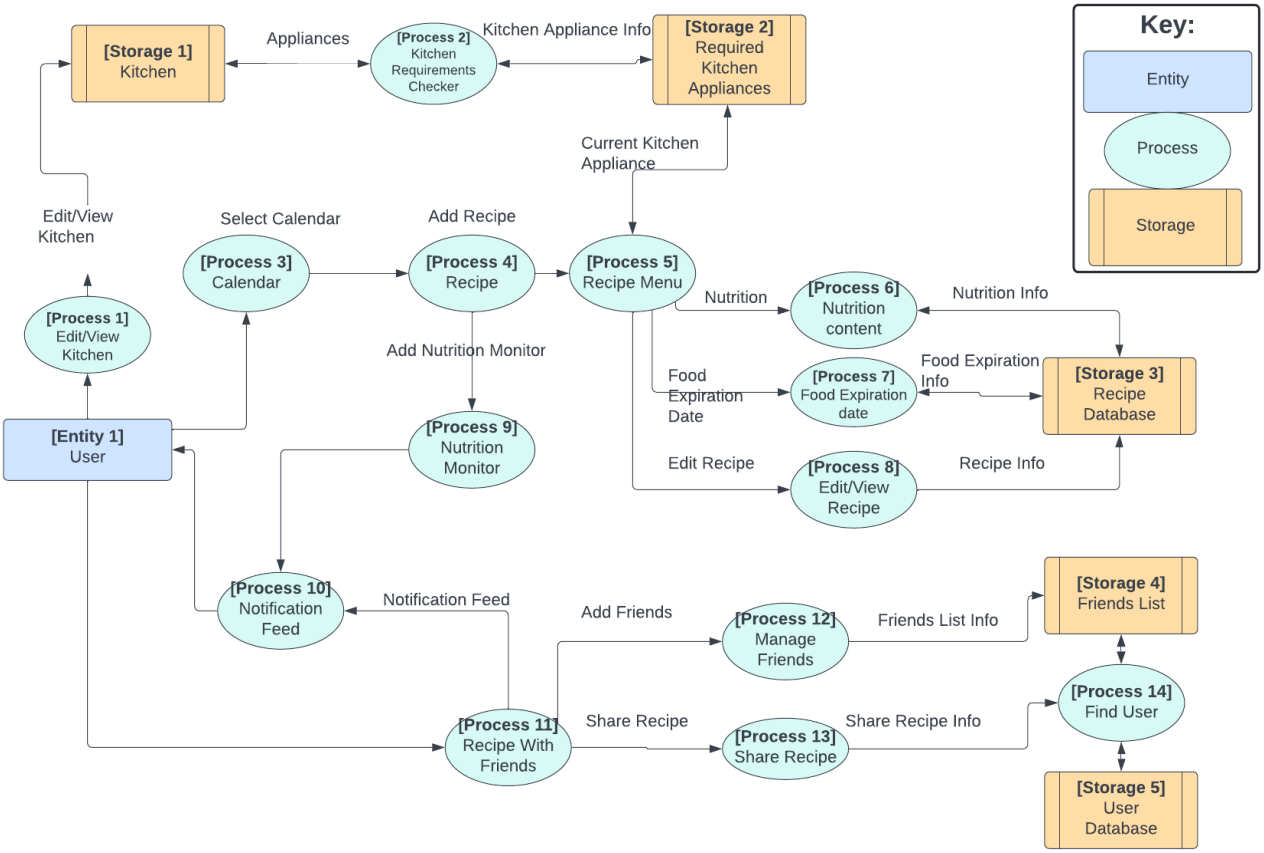
10.6.0 UC-6: Register Kitchen



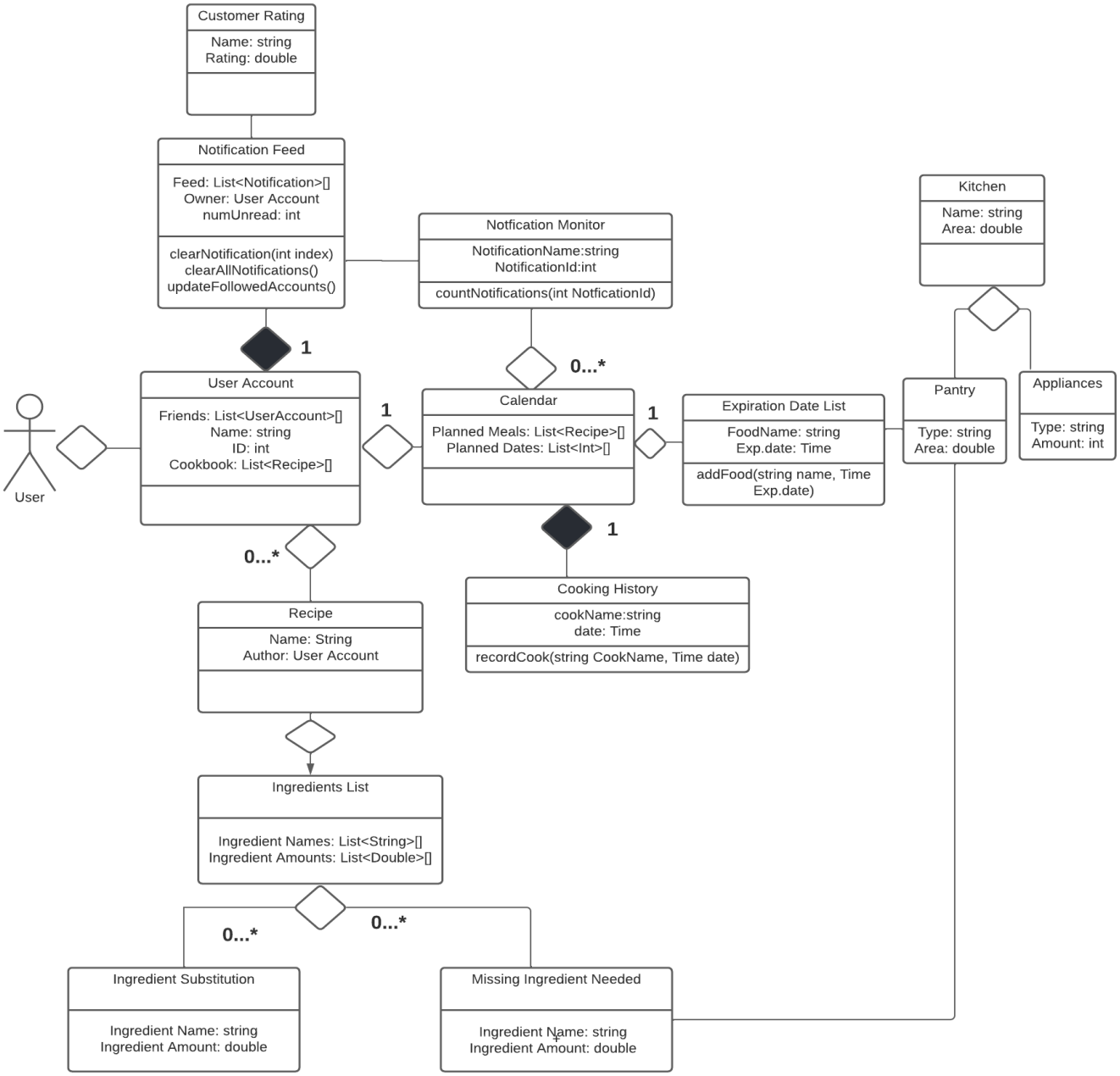
10.7.0 UC-7: Login



11.0 Data Flow Diagram



12.0 Class Diagram



13.0 Resources

What is agile? - what is Scrum? - agile FAQ's. Cprime. (2021, December 20). Retrieved January 13, 2022, from https://www.cprime.com/resources/what-is-agile-what-is-scrum/

Precision eating™ platform. Suggestic. (n.d.). Retrieved January 13, 2022, from https://suggestic.com/platform.html?utm_term=recipe+creator+software&utm_campaign=S%7C%2BSuggestic%2BPlatform%2B-%2BUS%26CA&utm_source=adwords&utm_medium=ppc&hsa_acc=4758559608&hsa_cam=10740362146&hsa_grp=106283372535&hsa_ad=454004165719&hsa_src=g&hsa_tgt=kwd-340516898126&hsa_kw=recipe+creator+software&hsa_mt=p&hsa_net=adwords&hsa_ver=3&gclid=Cj0KCQiAuP-OBhDqARIsAD4XHpegnlQUUwHQ41DpMGSOCPXORM5F-2_dpD5Y5w6g9JOTqv4Ky6nHgYwaArzGEALw_wcB

14.0 Revision Log

14.1.0 - 1/16/22: Professor feedback from Initial Report. (Week 2).

"The application will use the same amount of data as the recipe app uses" - This sounds like there are 2 different applications here - Are you referring to front end and backend, or something else?
Revision made: Adjusted the sentence and removed the “recipe app” phrase to clear confusion.
Location of Revision: Page 5 under operational, first bullet point.

In Marketability, you have "The application is only limited to those with a smartphone"; In the operational section, there is a reference to PCs as well. Will the app run on both?
Revisions made: Removed the references to personal computers to keep things consistent and keep the project scope manageable.
Location of Revisions: Page 3 under Key Features - first bullet point. Page 4 under Operational section - second bullet point.

In Support / Maintainability, you have "The development team will be kept on for a limited time after launch to address any early bug reports". Backend servers and services need ongoing maintenance - who's going to do backups, reboot servers, etc. ?
Revision made: Removed “limited time” phrase and added more specific details to the sentence.
Location of Revision: page 6 under support/maintainability, third sentence.

14.2.0 - 1/23/22: Professor feedback from domain model, user stories, and use case submission. (Week 3).

"As a user, I want to be able to add recipes saved in my cookbook to my planned meals to view a full list of future meals in the planned meals section, view the calendar which adds every planned meal to its planned date, and to get notified when a planned meal is scheduled soon or if I am missing anything to make it."

is actually 3 of them - add, view, get notified.
Revision made: Split this user story into 3 distinct user stories (add, view, get notified)
Location of Revision: Page 7-8, user stories 2-4.

This one - "As a user, I want to login to my account and find out the recipe that I need to have for a day. I check the ingredients that are appropriate for me, and if the ingredients are not available I will find out the substitutions for the missed ingredients."

this is also several user stories -'login to my account so I can see my recipes', 'view recipes for a date' and 'find out about missing ingredients'.

This is a fair amount of overlap with the user story just above.
Revision made: split this user story into 3 distinct user stories (check today's meals, find missing ingredients, view saved recipes)
Location of Revision: Page 8, user stories 5-7.

use case 1-
the steps should be about actions - 'add' is an action, 'wants to add' is not.
Revision made: Changed wording of user actions to be more concrete regarding which actions are performed.
Location of Revision: Page 8, under UC-1. 3rd and 5th items in the list of steps.

use case 2-
the preconditions aren't quite right - these are generally about the state of the application and what actions have been taken prior to the start of the use case. The user isn't going to know that they are missing an ingredient if they haven't logged in yet, so 'logged in' is better as a precondition than the first step in a separate use case. 'Chooses a recipe' might be a better place to start.
Revision made: Changed the precondition for this UC to have the user be logged in instead of it being a step. The trigger for this case was changed to the user selecting a recipe from the cookbook. Alternative steps changed.
Location of Revision: Page 9, under UC-2. Steps 1-4, Precondition, and Trigger for this use case were modified.

use case 3-

step 1 is better done as a definite action by the user - maybe something like 'user selects the "share recipe" option, then step 2 might be "system shows list of friends", then step 3 "user selects friends from list", etc.
Revision made: Expanded Step 1 into three steps as requested, making the first part of the process more of a definite action (“selects one or more friends”). Changed the first alternate slightly to adjust for this.
Location of Revision: Page 9-10, under UC-3. 1st, 2nd, and 3rd items in the list of steps and 1st item in alternates.

use case 5-
Are steps 5 and 6 required for this use case? Must the user exit? The use case can end without the application ending.
Revision made: Removed steps taken to exit the program. Added steps to more clearly display the process of getting to the nutritional information.
Location of Revision: Page 11, UC -5 under basic flow section.

Other Revision: Added UC-7 “Login” while working on the use case diagram.
Location of Revision: Page 12, below UC-6.

14.3.0 - 2/04/22: Professor feedback from use case diagram, functional, and nonfunctional requirements. (Week 4).

I understand the motivation behind putting the second actor on there, but as they are of the same type of actor, only 1 instance of a given type should be on there.
Revision made: Removed the second actor from the use case diagram.
Location of Revision: Page 13, under Use Case Diagram.

SR-FR-03 - is a user data set different than a user profile?
Revision made: Changed “user data set” to “user profile” to keep terminology more consistent and ease confusion.
Location of Revision: Page 14, under SR-FR-02 (formerly SR-FR-03, see below).

SR-FR-02 - I don't understand what this means. Is this derived from a use case or user story? (it doesn't always need to be, but if it is, the reason is a bit easier to follow). What's the measure of similarity?
Revision made: Removed this requirement because it was too ambiguous and not essential to the system. Renamed SR-FR 03-06 to SR-FR-02, SR-FR-03, etc. to account for this removal.
Location of Revision: Pages 14-15 under System Requirements.

14.4.0 - 2/11/22: Professor feedback from activity diagrams. (Week 5).

uc5: steps 2,4, and 6 are actually system responses, so writing them that way ('The system shows the meals on user calendar') would make that clearer.

The activity diagram has a fork (a parallel 'and', which requires all of the forked activities to happen, not something that matches what's in the use case description). It sounds like the basic flow + alternatives 1 and 2 are actually choices, and only one of them needs to be done. A few decisions (e.g., if the user picked their own calendar, else if the user picked their cooking history...) would work better. Picking another user's cookbook involves more complexity (showing a list of users, picking one, selecting their cookbook, etc.) that's not accounted for here. While useful, you might just handle it by referring to a separate use case that just invoke as a choice, but not write up the details for.
Revision made: Changed the wording of some steps to emphasize that the action is being taken by the system. Changed the first fork in the UC-5 activity diagram into a decision diamond. Simplified the social networking meal option in the UC-5 diagram to utilize another use case.

Location of Revision: Page 10, UC-5 section. Page 21, UC-5 activity diagram.

use case 6
Alternative #1 is complex enough and different enough that it really belongs in its own use case, or omit it from this version.
Revision made: Removed this alternative due to its complexity.
Location of Revision: Page 12, under UC-6 Alternatives.

14.5.0 - 2/18/22: Professor feedback from wireframes. (Week 6).

One of the things wireframes and thinking about the the Application UI navigation that comes (or should come) with it is that helps in showing how use cases flow (or should flow) from one to the other. One of the things to think about is where the user can go next after a use case is achieved; there should never be a dead end at the final screen / page / tab. Use cases 5 and 6 appear to have such dead ends, while earlier ones have a navigation menu at the bottom.
Revisions made: UC-5 wireframe has been adjusted to include an option to return to main menu. UC-6 wireframe has a navigation menu added at the bottom.
Location of Revisions:
Wireframe 5 and 6, Page 26-27.

14.6.0 - 2/24/22: Professor feedback from robustness diagrams. (Week 7).

use case 1 -
The 'display recipe ingredients' controller needs a data source for the recipe; maybe a request to a model object, and a returned recipe to hand to a view object.

Revision made: Another controller has been added that connects the display recipe ingredients to the recipe database.
Location of Revision: Page 31, UC-1

use case 2
Your 'Display' controller doesn't connect to a view object to display anything. A 'possible substitutes' view object might fit in well.
Revision made: ‘Display’ controller connected to the view object
Location of Revision: Page 32, UC-2

use case 3
The controllers connected to the 'list of user's friends' are for display, not retrieval; adding another controller for that purpose could feed both of the display controllers.
Revision made: Added another controller that connects to both relevant display controllers and the friend data model object.
Location of Revision: Page 33, under UC-3.

use case 5
the arrows to Meal Storage should be bidirectional.
Revision made: made the arrows to meal storage bidirectional
Location of Revision: Page 35, UC-5

14.7.0 - 2/24/22: Professor feedback from sequence diagrams. (Week 8).

use case 3
controllers don't belong as you have them on here along the top row; the list of user friends could either be a view object on its own or just be a list of data on an existing view object.
Revision made: Removed the Controllers and made the required methods part of view objects where relevant.
Location of Revision: Page 40, under UC-3.

use case 5:
controllers don't belong as you have them on here along the top row; you might have a Meal model object to represent a single meal, and a method on it for getting its nutrition info. 'Meal retrieval' and 'Nutrition retrieval' sound like methods.
Revision made: Changed the sequence diagram to represent the methods as methods instead of controllers.
Location of Revision: Page 41, Sequence diagram UC-5.

Other Revision: Added Table of Contents and Numbering system for the entire document.
Location of Revision: The entire document.

14.8.0 - 2/24/22: Professor feedback from sequence diagrams. (Week9).

You have quite a few edges in your data flow diagram that are unlabeled; for each of these unlabeled edges, please do one of 2 things:

1) label it with the data flow you think exists between its endpoints
Or 2) delete the edge.

Revision made: The data flow diagram has been changed by labeling the items that are present in the diagram.
Location of Revision: Page 43

Feedback From Group 2 Developer Peer Review

Add wireframes of all use cases connected

Revision made: We added a wire frame of all use cases connected based on the feedback we get received from group 2:

Location of Revision: page 30