Feed.Me

Version 2.0

by Team 4
Tung Dinh, Lauren Hager, Bryan Hsieh, Margaret Lanphere, Petra Power, Kim Van Wyk

Revision History

Date	Version	Description	Author
1/26/2019	1.0	Added Business Proposal, Software Requirements Specification, Business Team Approval, Domain Diagram	Team 4
1/31/2019	1.1	Added 6 Use Case Descriptions, Use Case Diagram, and 1 Activity Diagram	Team 4
2/12/2019	1.1.2	Week 3 Formal Report Feedback Revisions	Lauren
2/17/2019	1.2	Added 5 additional Activity Diagrams	Team 4
2/24/2019	1.3	Update business proposal, add wireframes	Team 4
3/3/2019	1.4	Added 6 Robustness Diagrams, and 1 Sequence diagram. Made revisions based on stakeholder peer review.	Team 4
3/19/2019	2.0	Added 5 sequence diagrams, 1 class diagram, 1 test case, and implemented revisions from Week 6 submission, implemented Developer Peer Review feedback	Team 4

Table of Contents

1. Business Proposal

1.1 Executive Summary

Feed.me is a mobile application which aims to help its users organize one of the most important aspects of their everyday lives – food. With Feed.me, the users can keep track of the contents of their fridge and pantry, create grocery shopping lists, prepare meal plans, and look up recipes according to their dietary preferences - all in one place.

Mobile phone users increasingly use the applications available on their devices to help them handle a variety of issues in their day-to-day lives. With so many different apps for finding recipes, managing one's nutritional needs, planning meals, and organizing shopping lists, it may be difficult to stay organized and not feel overwhelmed. Feed.me's intention is to enable users to control all these activities in a single application with an intuitive, user-friendly interface.

The application relies on revenue from advertisers, and users who choose to pay to disable the advertisements on their client devices. The application also counts on a prescription model with premium features available to users who pay a monthly fee. There is a wide variety in potential advertisers, from grocery stores to cooking utensil sellers and manufacturers to kitchen appliances makers. The application stands to be very financially successful, and has potential to be a good investment opportunity, as it aims to gather a large user base with a wide variety of needs.

Feed.me has plans to continuously evolve to suit its users' needs and adapt to the newest technology. The plans for future features include smart fridge integration, ordering groceries from online grocery stores, and integration of home voice assistants such as Amazon Alexa.

1.2 Stakeholders

Figure 1.2.1 Table of Stakeholders Roles and Responsibilities.

Stakeholders	Roles/Responsibilities
Business Team of Feed.Me (Group 3)	This team is responsible for creating the business idea and explaining the requirements to the Architecture team. They will oversee the development of the business plan and reviewing the architecture teams work.
End users	The people who acquire and use the finished application. People who want to be able to select recipes based on household ingredients, food preferences, nutrition and generate shopping lists from selected recipes.

UI Designer	Responsible for using the architecture requirements, diagrams, wireframes to create a user interface that is easy to use, appealing and conforms to the use cases.
Marketing Team	This team is responsible for marketing the application to end users to inform people of its use, applications and existence.
Project Manager	Responsible for planning the workflow and development timeframe and keeping development ontrack and running smoothly. Managing the Project to ensure the business plan is properly enacted.
Architecture Team (Group 4)	This team is responsible for developing the architecture that will outline the plan for developing the application. This plan will flesh out all of the details needed to design and create the application with specification, diagrams, graphs, requirements and use cases.
Developers/Testers	Responsible for coding the application and testing its use to ensure that the requirements are met and the user experience is enjoyable and meets the specifications.

1.3 The Problem

There are several problems the Feed.me application is trying to address:

- Many different but related tasks scattered across various, unrelated applications:
  Food planning preparation is an activity which takes up a significant amount of our time. Trying to use different applications, or even non-software-based systems, for the various aspects of meal planning can be time consuming and frustrating.
- Keeping track of ingredients in the home:
  It can be difficult to keep track of everything we have in the pantry, in the fridge, in the freezer, and any other area that we use to store food. When planning a celebration dinner or meals for the week, it's good to know what ingredients are already available, and in what amounts.
- Keeping track of what groceries to buy:
  Some food needs to be bought weekly or twice a week, while other items need to be purchased rarely. For example, buying an extra bag of flour only to discover two more bags behind the canned beans may not be a problem, since a bag of flour can sit around for several of months before hitting their expiration date. However, buying an gallon of

milk when your spouse bought two gallons two days ago may result in spoiled food, needlessly spent money, and a space in the fridge taken by a large milk container.

- Keeping track of one's health and nutritional needs:
  Many people are increasingly concerned with their diets and nutritional intake, and still others have food allergies or dietary preferences that don't allow them to consume certain products. Searching for recipes that contain certain ingredients while avoid others can be time consuming and difficult.

1.4 The Solution

The solution is a multipurpose phone application that will handle recipe search and reverse recipe search, hold the information about the ingredients and products the user has, and keep track of shopping lists for the items the user needs, and help with meal planning and nutrition tracking.

Our goal is to create a simple UI for the users without overwhelming the users when using the application. Any user can use all the features the application offers, or they can use just the parts that are useful to them personally.

Feed.me has many components and functionalities, which is why it will use several third party APIs for recipe search and nutrition and health information retrieval. Additionally, we will use API's for the app to connect with Android and IOS platform to integrate different features.

1.5 Financial Summary

Figure 1.5.1 Cash Outflows

CASH OUTFLOWS	Quarter												Total per Item
	1	2	3	4	5	6	7	8	9	10	11	12	
Salaries	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$168,000	$2,016,000
Rent	$7,500	$7,500	$7,500	$7,500	$7,500	$3,000	$3,000	$3,000	$3,000	$3,000	$3,000	$3,000	$58,500
Infrastructure	$10,000	$100	$100	$100	$100	$100	$100	$100	$100	$100	$100	$100	$11,100
Software	$5,000												$5,000
Utilities	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$1,500	$18,000
Operations	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$10,000	$120,000
Insurance	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$2,500	$30,000
App Store Fees						$50	$25	$25	$25	$25	$25	$25	$200
Business Registration	$19												$19
Patent/Trademark	$5,000												$5,000
Marketing					$300,120	$300,120	$300,120	$300,120	$300,120	$300,120	$300,120	$300,120	$2,400,960
Total per Quarter	$209,519	$189,600	$189,600	$189,600	$489,720	$485,270	$485,245	$485,245	$485,245	$485,245	$485,245	$485,245	$4,664,779

1.5.2 Justification for Outflows

1. We are taking the following assumptions for our project: considerable understanding of requirements, none to modest amount of experience with similar projects, only conform with basic requirements, only conform with basic external interaction, extensive amount of new operational environment, none to minimal new tech, low need to meet schedule, and less than 50k lines of code. Based on this, we will assume **Organic mode.**
2. This is an average Android/iPhone app. After reviewing the chart at https://www.businessinsider.com/how-many-lines-of-code-it-takes-to-run-different-software-2017-2, we estimate this project to have 40k lines of code.
3. Based on Cocomo model, calculations are: Effort = 115 person-months, Time = 15 months, therefore, 7-8 people needed.
4. The average app developer makes $84,000 a year, therefore, 8 people x ($84,000 / 4) = $168,000 for salaries per quarter. Presuming 15 months of development, this will go on for 5 quarters.
5. Average rent for an office for 6-10 people is $2500/month. Presumably, we can downsize after development is complete.
6. Infrastructure includes items such as computers, printer, phones, hardware, and office supplies = $10000. This will mostly be purchased at the start of development and then maintained.
7. Software will be purchased at the start of development. This includes IDEs, other development tools, and productivity tools.
8. Utilities for the office space include heat, water, internet, phone.
9. Operations include security, maintenance, support.
10. Insurance include general business insurance.
11. App store fees include Apple ($99 a year) and Google Play ($25 one time).
12. We will market the app more aggressively near release, and then continue to market at a lesser extent.

Figure 1.5.3 Cash Inflows

CASH INFLOWS	Quarter												Total per Item
	1	2	3	4	5	6	7	8	9	10	11	12	
Advertisements						$16,200	$16,200	$16,200	$16,200	$16,200	$16,200	$16,200	$113,400
App subscriptions						$2,985,000	$2,985,000	$2,985,000	$2,985,000	$2,985,000	$2,985,000	$2,985,000	$20,895,000
Total per Quarter						$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$21,008,400

1.5.4 Justification for Inflows

1. We plan to release the app in our 6th quarter. The free app will have ads, or users can pay a monthly subscription of $1.99 to remove ads.
1. Additional monetization strategy will be implemented as we add Samsung Smart Refrigerator integration, and as we roll out family-sharing account. The subscription fees for these additional services will be determined during the design phase for these new features, according to the market rates at that time. The design phase for the planned new features will start at the end of Quarter 4.
2. We envision that our app will have 20 million users per month, a similar statistic to a similar app called MyFitnessPal (https://www.statista.com/statistics/650748/health-fitness-app-usage-usa/)
3. About 5% of app users spend money on in-app purchases, so we err on the side of caution that 500 thousand users (2.5%) will purchase a monthly app subscription that removes ads.
4. We estimate that 30,000 users will use our app every day, so 30,000 users * 2 minutes * 2 ads per minute = 120,000 banner ads. 120,000 ads * (1.5/100 click rate) = 1,800 clicks. So, revenue through ads will be 1800 clicks * .10 rate per minute = $180 a day

1.5.5 Cost/Profit Analysis

Cost/Profit Summary	Quarter												Total per Item
	1	2	3	4	5	6	7	8	9	10	11	12	
Net Profit						$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$3,001,200	$21,008,400
Net Cost	$209,569	$189,625	$189,625	$189,625	$199,625	$27,125	$27,125	$27,125	$19,125	$19,125	$19,125	$19,125	$1,135,944
Net per Quarter	-$209,569	-$189,625	-$189,625	-$189,625	-$199,625	$2,974,075	$2,974,075	$2,974,075	$2,982,075	$2,982,075	$2,982,075	$2,982,075	$19,872,456

1.6 Risks

Figure 1.6.1 Risks Analysis and Management for FeedMe app

Risk ID	Risk Category	Risk Description	Impact	Frequency	Risk Control
R-1.0	Security	Storing credentials, passwords, login info and user health info	Catastrophic	Remote	**Reduce/Mitigate:** Use shell architecture to abstract access
R-2.0	Testability	Interaction of API's with App may prove more difficult to test due to user preferences in apps from API's	Minor	Remote	**Reduce/Mitigate**: Test with various user preferences from API apps. Get Beta testing

R-3.0	Economy	Overestimating the number of users willing to pay additional fees	Serious	Remote	**Reduce/Mitigate**: Study market and user behavior on similar applications, thoroughly analyze data
R-4.0	Legal Compliance	Regulations of data storage for geographic usage locations	Serious	Occasional	**Avoid**: Hire legal team to stay abreast of current regulations
R-5.0	Competitiveness	Advancement in other technology and apps with greater features	Critical	Remote	**Avoid:** Continue to develop the app, adding features and usability, with focus on ease-of-use and customer satisfaction.
R-6.0	Extensibility	Planned future features, such as the Samsung Smart Fridge integration, will depend on technology available at the time of design and implementation. The codebase may need to change significantly for these new features to work	Serious	Occasional	Reduce/Mitigate: Design and code with the extensibility in mind. Employ the principle of encapsulation for modules that are likely to change.
R-7.0	Reliability	Risk of failure to the server/API during recipe selection, info retrieval	Serous	Remote	**Avoid:** Use match style to provision servers in relation to demand.
R-8.0	API Interaction	Changes/updates to API's may cause errors in app	Minor	Remote	**Reduce/Mitigate**: Monitor API updates for changes
R-9.0	Feasibility	Large scope and diverse functionalities required may make the system hard to develop and make function properly	Catastrophic	Occasional	**Avoid:** Spend enough time on design before starting the development phase

1.7 Conclusion

Currently, this idea is the only app that has integrated the idea of reverse recipe lookup, automated weekly meal planning, integration with other IOT systems, and customized to user's personal preference. Once this app has been released, it will be able to take up big margins of the market within the cooking industry and made simple for users to use. This will help the app gain traction and become a popular mobile app for users to use to organize their shopping and meal planning.

2. Software Requirements Specification

2.1 Functional Requirements

The following functional requirements are categorized by Account Management, Recipe Finder and Bigoven.com Integration, and MyFitnessPal and Recipe Finder Integration. The Account Management section lists attributes of the system that relate to user accounts. The Recipe Finder and Recipe Database Integration section lists how recipes are found and integration with the database that holds our recipes. Finally, the MyFitnessPal and Recipe Finder and Apple HealthKit Integration section details functionality relating to MyFitnessPal and Apple HealthKit, an application that allows user to track their nutrition and fitness goals.

2.1.1 Account Management

Identifier	Requirement
FR-AM-01.01	The system shall provide user profiles to store username, password, and email address.
FR-AM-01.02	Users may be able to create a profile using Facebook, Google, or by filling in the profile fields and saving the information.
FR-AM-02.01	The basic user profile shall use the account email address for account recovery.
FR-AM-02.02	The user shall be able to request username or password if forgotten. Requested username/password sent to account recovery email.
FR-AM-02.03	User shall be able to reset the account password if the account password is forgotten.
FR-AM-03.01	User shall be able to edit user preferences and profile.
FR-AM-03.02	User shall be able to change the account password or the account email address.
FR-AM-04.01	The user shall be able to save partial changes to their user information.
FR-AM-04.02	The user profile shall save automatically upon exiting the app.
FR-AM-05.01	The user shall be able to select their notification preferences for push and email notifications for recommended recipes and shopping reminders.
FR-AM-06.01	The user profile shall have filters for allergies, dietary restrictions, and nutritional lifestyle choices (ie. vegan, pescetarian, gluten-free, dairy-free, paleo, ketogenic etc.).

FR-AM-06.02	User profile filters must be saved and auto populates the filters screen when choosing filters.
FR-AM-06.03	The system shall allow the user to generate filter options based on rating, allergy, lifestyle, personal preference, cuisine style, ingredient (ie. butter vs. olive oil).
FR-AM-07.01	The user profile must contain ability for user to set daily and weekly nutrition requirements. ie. sodium, calories, vitamins, protein, carbs, etc.
FR-AM-08.01	The user shall have the option to create a 4-digit passcode to open the app.

2.1.2 Recipe Finder and Recipe Database Integration

Identifier	Requirement
FR-RF-01.01	The system will connect to an existing recipe database using an available API.
FR-RF-02.01	The user shall be able to add/remove recipes from a list of favorite recipes.
FR-RF-02.02	The user must be able to add/remove their own recipes to their favorites list.
FR-RF-02.03	The app shall save calorie info, nutrition info, along with ingredients lists and cooking instructions to the recipe list.
FR-RF-02.04	The user shall have the ability to download recipes to their device's local storage.
FR-RF-03.01	The user shall be able to save and add/remove items in an ingredients list on the app.
FR-RF-03.02	The user shall be able to add a list of additional current ingredients in the home.
FR-RF-04.01	The user shall be able to search for recipes based on ingredients lists.
FR-RF-04.02	The user must be able to search for recipes that they have no ingredients for.
FR-RF-04.03	The user shall be able to rate recipes after use for like and dislike. Dislike must be able to be categorized as specific recipe or food itself. (For example, this banana bread recipe vs. banana bread as

	a food).
FR-RF-05.01	The user may be able to select a recipe missing a predetermined amount of ingredients, not on their ingredients list.
FR-RF-05.02	The app must be able to recommend recipes based on users list of previous recipe likes and dislikes.
FR-RF-05.03	The app must be able to recommend recipes based on user profile filters.
FR-RF-06.01	App shall make a shopping list of ingredients to create the selected recipe that are not already on the user's saved ingredients list.
FR-RF-06.02	User shall be able to request reminders to purchase shopping list ingredients

2.1.3 MyFitnessPal and Recipe Finder and Apple HealthKit Integration

Identifier	Requirement
FR-MF-01.01	The app shall be able to recommend recipes based on nutrition filters.
FR-MF-01.02	User shall be able to link their account to MyFitnessPal.com.
FR-MF-01.03	User profile nutrition information should update from MyFitnessPal or Apple HealthKit
FR-MF-02.01	The app should update user's MyFitnessPal and Apple HealthKit apps with nutritional info from recipes used.
FR-MF-02.02	User may be able to enter their height, weight, and age.
FR-MF-02.03	User should be able to import their meal nutritional history from the MyFitnessPal.com Diary.
FR-MF-03.01	iOS users may allow integration between Feed.me & their Health app via Apple's HealthKit to automatically import nutritional data.

2.1.4 Weekly Meal Planner

Identifier	Requirement
FR-MP-01.01	User shall be able to create a weekly meal plan based on nutritional requirements and filters.

FR-MP-01.02	The app may be able to see what meals you have had for the week and filter recommendations. (ie. pizza on Monday, don't recommend pizza for Tuesday)
FR-MP-01.03	The user shall have the ability to assign/remove saved recipes to their weekly meal plan.
FR-MP-02.01	The user shall be able to create meal plans up to two weeks in advance.
FR-MP-02.02	The app shall add ingredients to the shopping list that are needed for the weekly meal plan that are absent from the pantry.

2.1.5 User Interface

Identifier	Requirement
FR-US-01.01	An icon must display to show app in processing during lookup or other processes
FR-US-02.01	The GUI must be able to accommodate increased text sizes for accessibility.
FR-US-02.02	The interface must clearly labeled UI elements that are found within 5 seconds of loading the screen.
FR-US-02.03	The application shall provide English language options for text

2.2 Non-Functional Requirements

The Non-functional requirements reflect the attributes of the system. Rather than specific capabilities, these requirements are quality considerations of the system. This section specifies how we plan to design our system in terms of concerns such as performance, usability, and availability.

2.2.1 Usability

Identifier	Requirement
NF-US-01.01	User interface shall conform to Google's design guidelines for Android apps.
NF-US-01.02	User interface shall conform to Human Interface Guidelines for Apple platforms.
NF-US-02.01	The GUI shall allow users to navigate to a destination with 3 or fewer clicks.
NF-US-02.02	The interface must display help messages within less than 1

	second when a user requests, to assist the user to navigate through the app.

2.2.2 Reliability

Identifier	Requirement
NF-RL-01.01	The app defect rate shall be less than 1 failure per 1000 hours of operation.
NF-RI-01.02	No more than 1 per 5,000 recipe searches shall result in a failure requiring the user to restart the app.

2.2.3 Extensibility

Identifier	Requirement
NF-EX-01.01	When the app needs updates, developers shall be able to make changes to less than 8 modules to complete the update.

2.2.4 Performance

Identifier	Requirement
NF-PF-01.01	The app must process 10,000 recipe checks per user requesting a recipe.
NF-PF-02.01	The app must analyze the ingredients correctly 99% when inputted in order to provide a recipe that contains the correct ingredients.
NF-PF-03.01	The system must return a list of recipes from the search within 5 seconds.

2.2.5 Availability

Identifier	Requirement
NF-AV-01.01	The app must be available to users by meeting or exceeding 99% uptime.
NF-AV-01.02	The app shall not be unavailable for more than 1 hour per 1000 hours of operation.
NF-AV-02.01	The app must need less than 20 seconds to restart after a failure 95% of the time.

2.2.6 Security

Identifier	Requirement
NF-SC-01.01	The app shall inform the user through email when their account has been used to access the app from another location within less than 3 seconds.
NF-SC-01.02	At least 99% of intrusions shall be detected within less than 10 seconds.
NF-SC-02.01	The app shall identify all of its clients before allowing them to use its capabilities.
NF-SC-02.02	The app shall communicate with the database over HTTPS.

2.2.7 Portability

Identifier	Requirement
NF-P-01.01	The app must withhold 100% of user data when the user installs the app on different platforms (IOS and Android).
NF-P-02.01	The time to transfer Feed.Me's data to another database shall not exceed more than 2 hours.

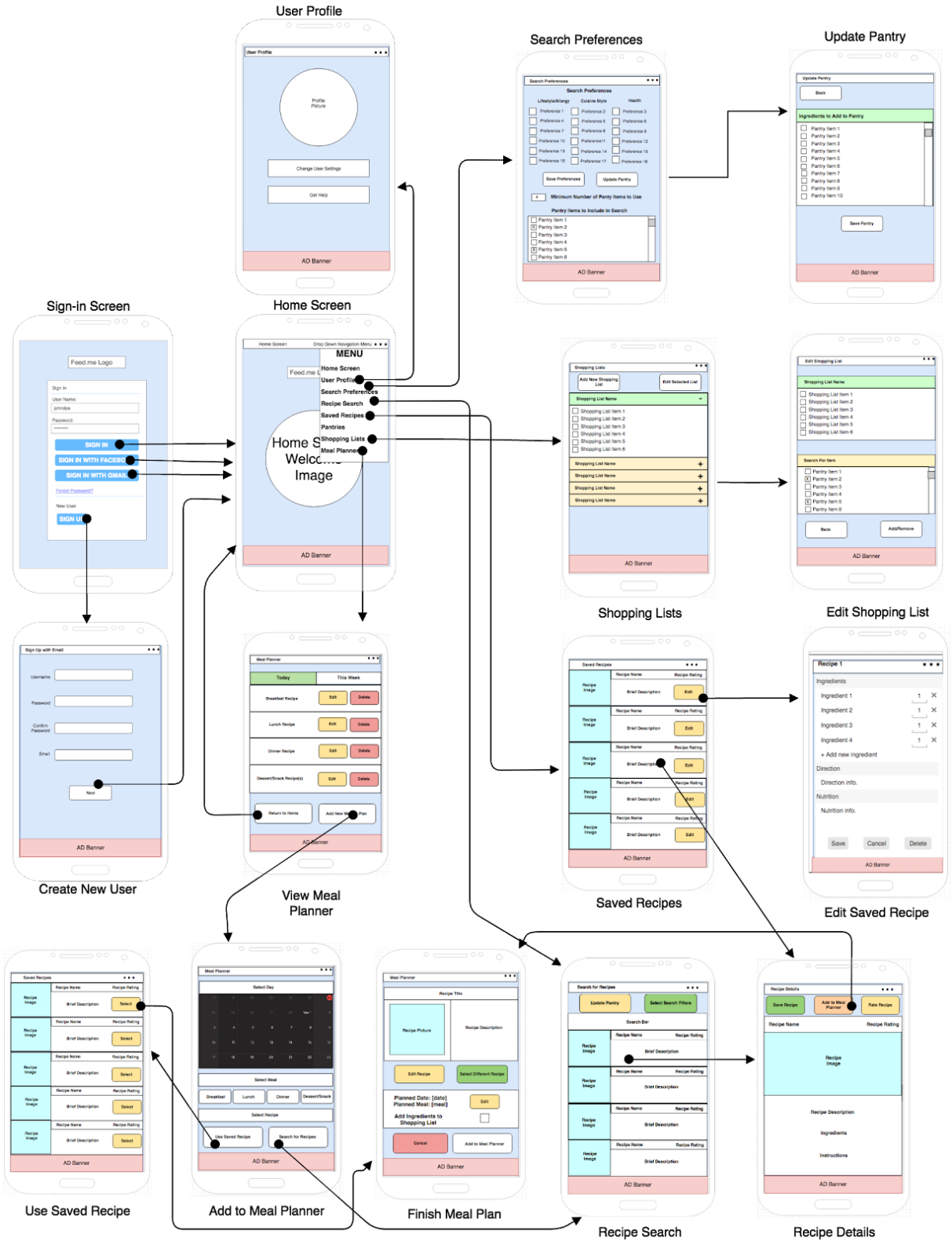
2.3.1 Requirements for Future Optional Features

Identifier	Requirement
NF-FO-01.01	The additional languages French, German, and Spanish may be implemented.
FR-FO-02.01	All recipes and saved list may sync to all other user devices with the same account.
FR-FO-03.01	The user may be able to add items to their ingredients list by scanning a UPC barcode or QR code, via connection to an available UPC or QR Code API
FR-FO-04.01	The user shall be able to link the app to an account with the recipe database provider, if they have one.
FR-FO-05.01	The system may connect with Samsung Smart Refrigerators for the pantry function via Smart Home Cloud API (if accessible).
FR-FO-06.01	The user may be able to get total price approximation of each recipe.

FR-FO-07.01	User may be able to select between individual account and family account.

3. Wireframes

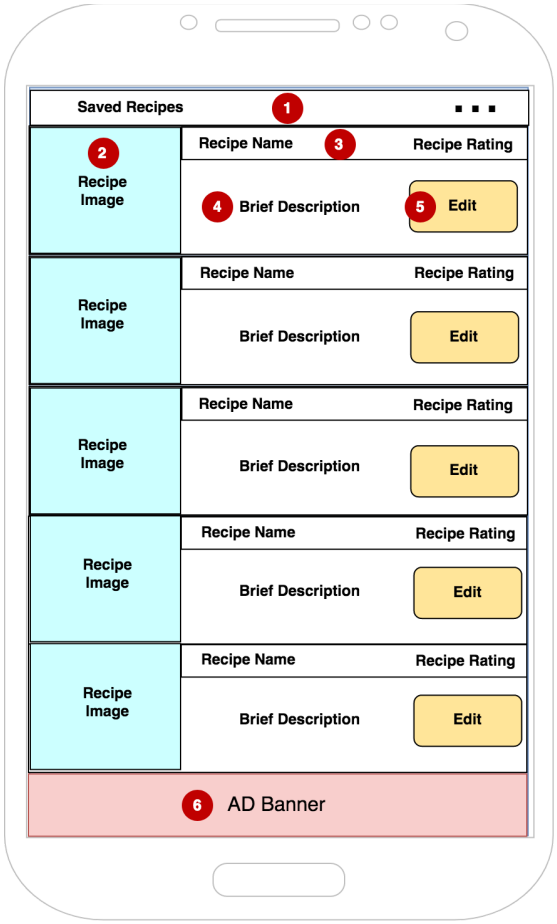
Figure 3.1 Main App Features Storyboard



3.2 Saved Recipes Wireframe

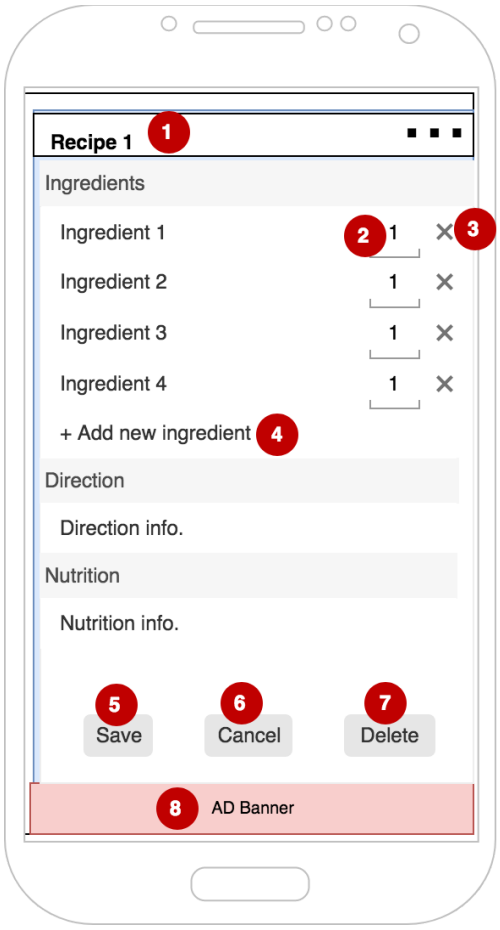
Wireframe ID	Wireframe Feature Description
WF-SA-01	User opens saved recipes
WF-SA-02	Recipe image
WF-SA-03	Recipe name
WF-SA-04	Brief description of a recipe
WF-SA-05	User edits a recipe
WF-SA-06	Ad Banner

Figure 3.2.1 Wireframe Saved Recipe Screen



Wireframe ID	Wireframe Feature Description
WF-RI-01	User edits recipe name
WF-RI-02	User edits the quantity of an ingredient
WF-RI-03	User deletes an ingredient
WF-RI-04	User adds new ingredient
WF-RI-05	User saves changes
WF-RI-06	User cancels editing recipe
WF-RI-07	User removes the recipe from saved lists
WF-RI-08	Ad Banner

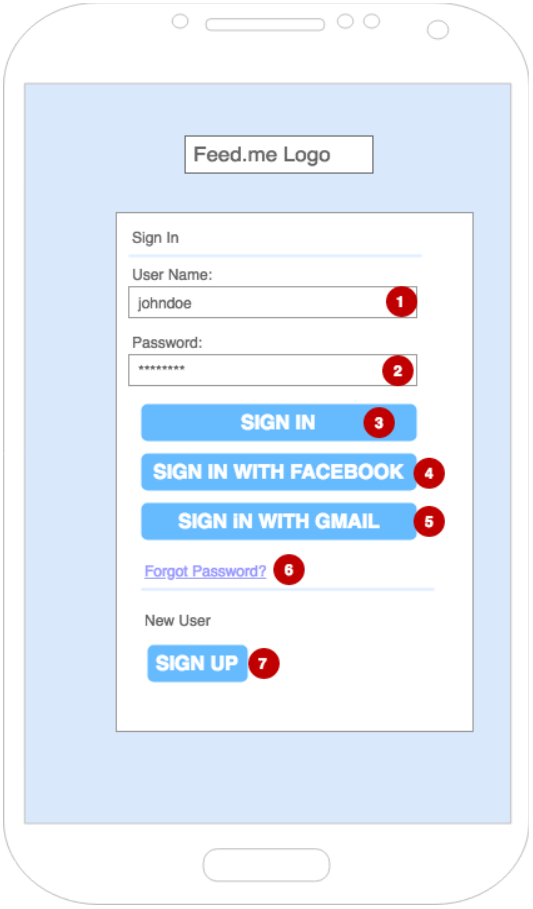
Figure 3.2.2 Wireframe Recipe Ingredient Screen



3.3 Sign In Screen Wireframe

Wireframe ID	Wireframe Feature Description
WF-SI-01	User sign in with username
WF-SI-02	User uses password to sign in
WF-SI-03	Button to sign in after typing in username and password
WF-SI-04	Alternatively, user can sign in with their Facebook credentials
WF-SI-05	Alternatively, user can sign in with their Gmail credentials
WF-SI-06	User can request temporary password to be sent to the email linked to their account
WF-SI-07	New user can sign up for an account

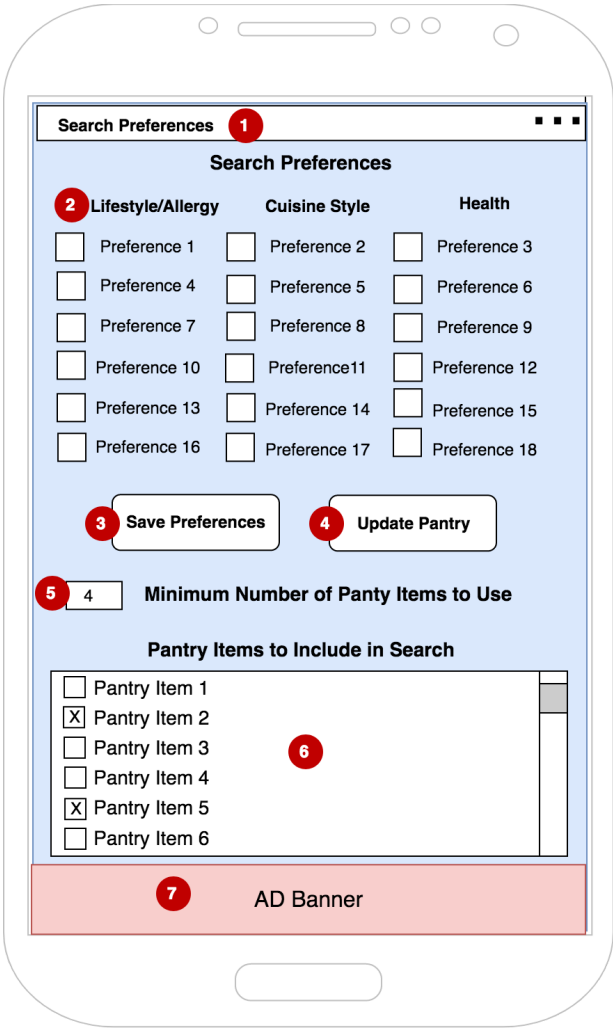
Figure 3.3.1 Wireframe Sign In Screen



3.4 Search Preferences Wireframe

Wireframe ID	Wireframe Feature Description
WF-SP-01	Drop Down - Navigation / Menu Bar
WF-SP-02	Preferences check boxes
WF-SP-03	Save Preferences Button - Saves preferences that are in profile to profile
WF-SP-04	Update Pantry Button - Takes you to the pantry screen to add items
WF-SP-05	Text box to enter min pantry items to include use in search
WF-SP-06	Scrolling list of pantry items with check boxes for items to include
WF-SP-07	Ad Banner

Figure 3.4.1 Search Preferences Wireframe



3.5 Shopping List Wireframes

Figure 3.5.1 Shopping List Wireframe

Wireframe ID	Wireframe Feature Description
WF-SL-01	Drop Down - Navigation / Menu Bar
WF-SL-02	Currently selected shopping list
WF-SL-03	List of existing shopping lists
WF-SL-04	Add New Shopping List Button for creating new shopping list
WF-SL-05	Ad banner
WF-SL-06	Edit the selected shopping list
WF-SL-07	Unselected shopping lists

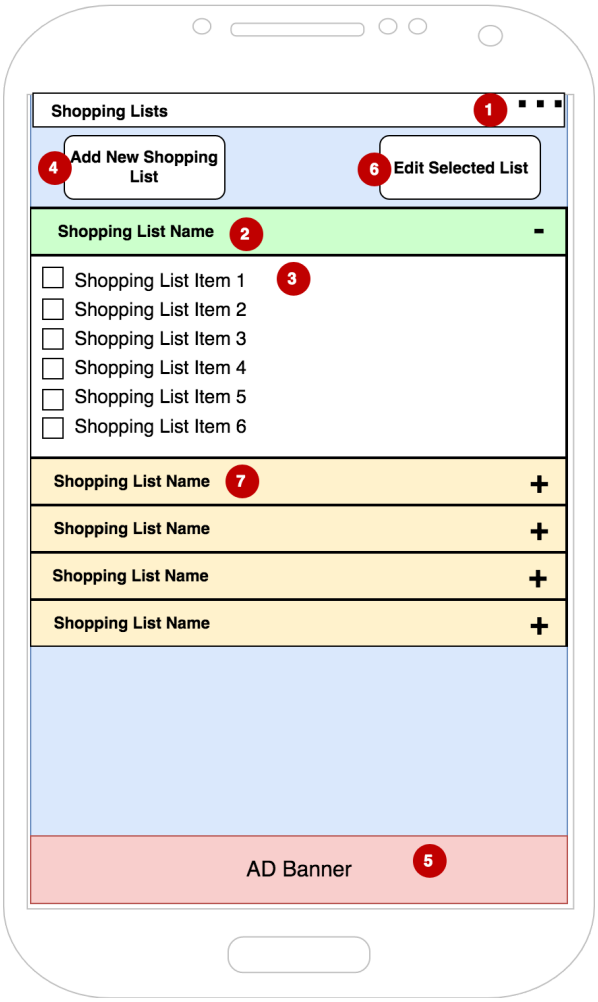
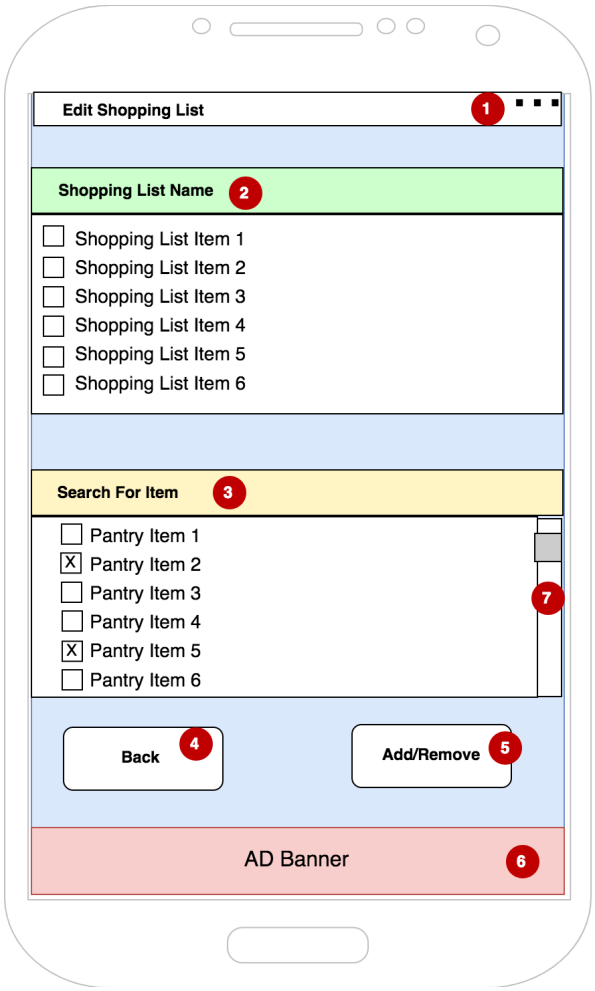


Figure 3.5.2 Edit Shopping List Wireframe

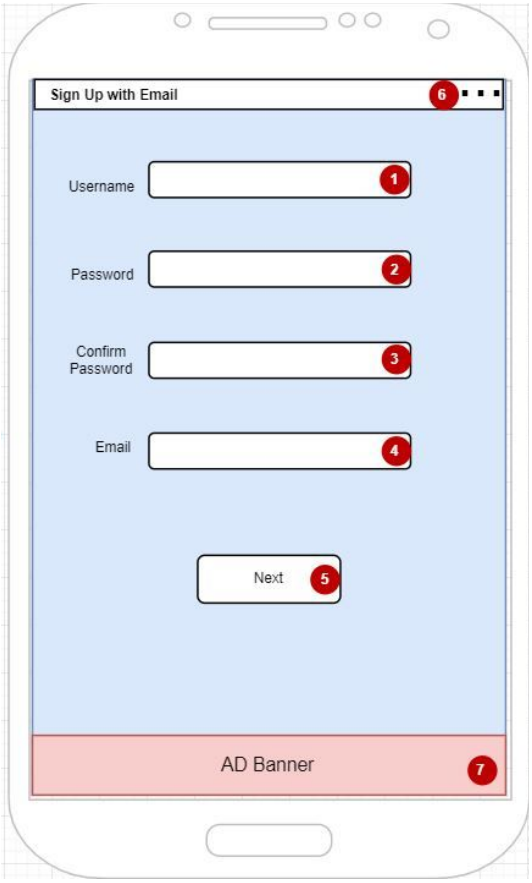
Wireframe ID	Wireframe Feature Description
WF-ESL-01	Drop Down - Navigation / Menu Bar
WF-ESL-02	Displays the items contained on the selected shopping list
WF-ESL-03	Search for new items to add to shopping list
WF-ESL-04	Back button to navigate to shopping list page
WF-ESL-05	Add or remove items from the shopping list or from the search menu
WF-ESL-06	Ad Banner
WF-ESL-07	Scroll bar to scroll through items in search menu



3.6 Create Profile Wireframe

Wireframe ID	Wireframe Feature Description
WF-CP-01	User enters username of choice
WF-CP-02	User enters password of choice
WF-CP-03	User enters password of choice again to confirm
WF-CP-04	User enters unique email of choice
WF-CP-05	Proceed with Information
WF-CP-06	Drop Down - Navigation / Menu Bar
WF-CP-07	Ad Banner

Figure 3.6.1 Create Profile Wireframe



3.7 Meal Planner Wireframes

Figure 3.7.1 Meal Plan View Screen Wireframe

Wireframe ID	Wireframe Feature Description
WF-MPV-01	Drop Down - Navigation / Menu Bar
WF-MPV-02	Toggle button to switch view between today's and the current week's meal plan
WF-MPV-03	List of planned recipes for each meals, or blank spaces if there are no planned recipes for that meal
WF-MPV-04	Edit button to edit the selected recipe for that meal
WF-MPV-05	Delete button to delete the planned recipe for that meal
WF-MPV-06	Button to return to home screen
WF-MPV-07	Button to add a new meal to the meal planner
WF-MPV-08	Ad Banner

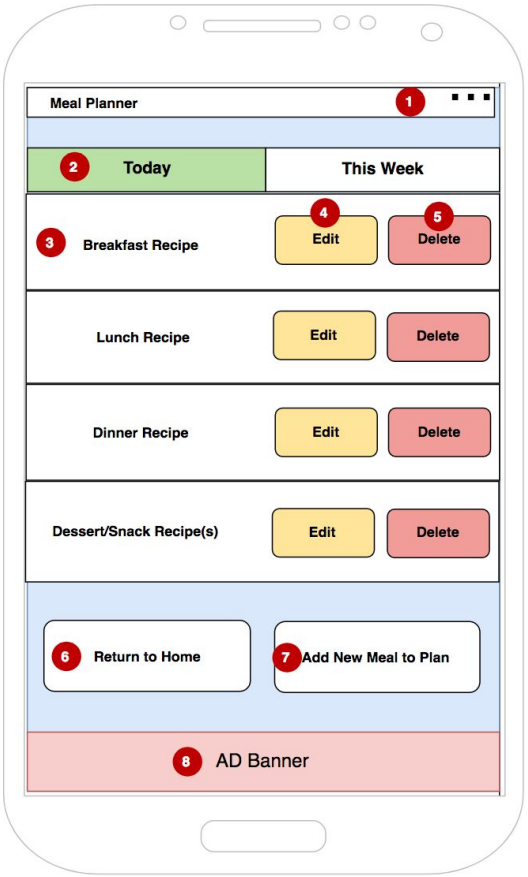


Figure 3.7.2 Add to Meal Plan Screen Wireframe

Wireframe ID	Wireframe Feature Description
WF-MPA-01	Drop Down - Navigation / Menu Bar
WF-MPA-02	Calendar for choosing the day the meal is planned for
WF-MPA-03	Meal selection buttons for breakfast, lunch, dinner, and dessert
WF-MPA-04	Use Saved Recipe button takes user to list of saved recipes to select
WF-MPA-05	Search for Recipes button takes user to recipe search to find a recipe to use
WF-MPA-06	Ad Banner

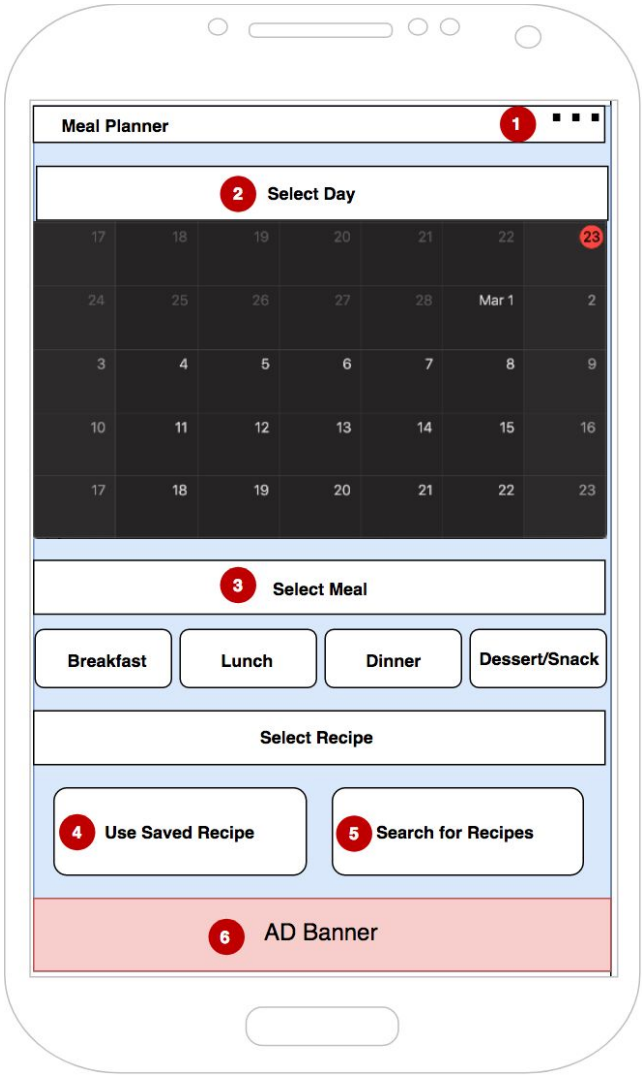


Figure 3.7.3 Finish Meal Plan Screen Wireframe

Wireframe ID	Wireframe Feature Description
WF-MPF-01	Drop Down - Navigation / Menu Bar
WF-MPF-02	Recipe Title Bar
WF-MPF-03	Picture for Recipe
WF-MPF-04	Recipe Description
WF-MPF-05	Button to edit recipe portions, ingredients, etc.
WF-MPF-06	Button to return to recipe search to select different recipe
WF-MPF-07	Meal plan details from previous page
WF-MPF-08	Button to return to edit date or meal time
WF-MPF-09	Checkbox for adding missing ingredients to shopping list
WF-MPF-10	Button to cancel adding the recipe to meal plan and return to home screen
WF-MPF-11	Button to add planned recipe to the meal planner and return to home screen
WF-MPF-12	Ad Banner

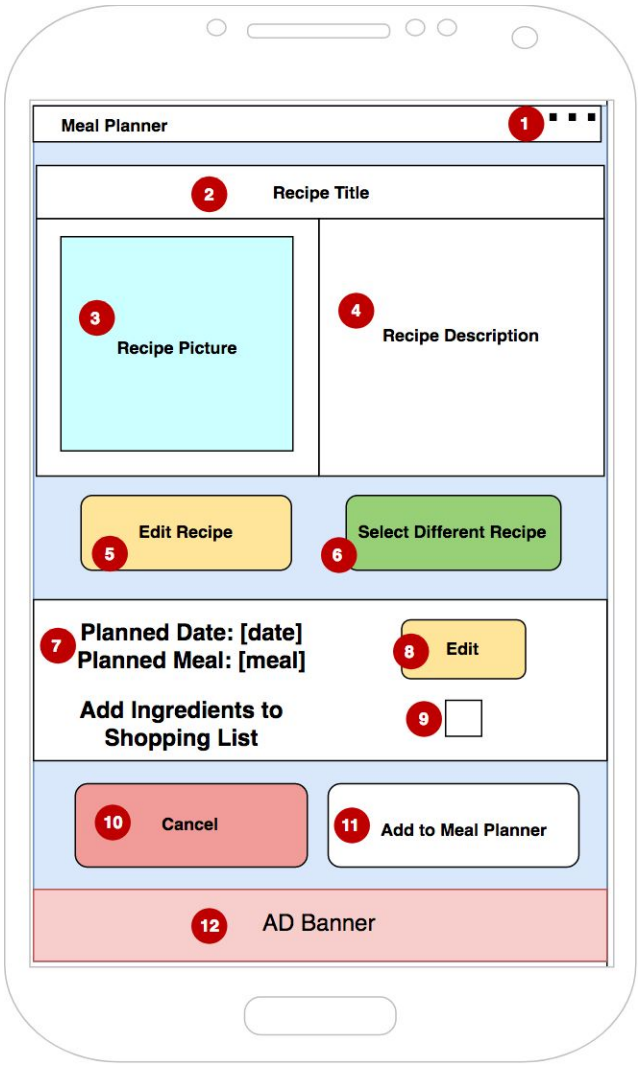
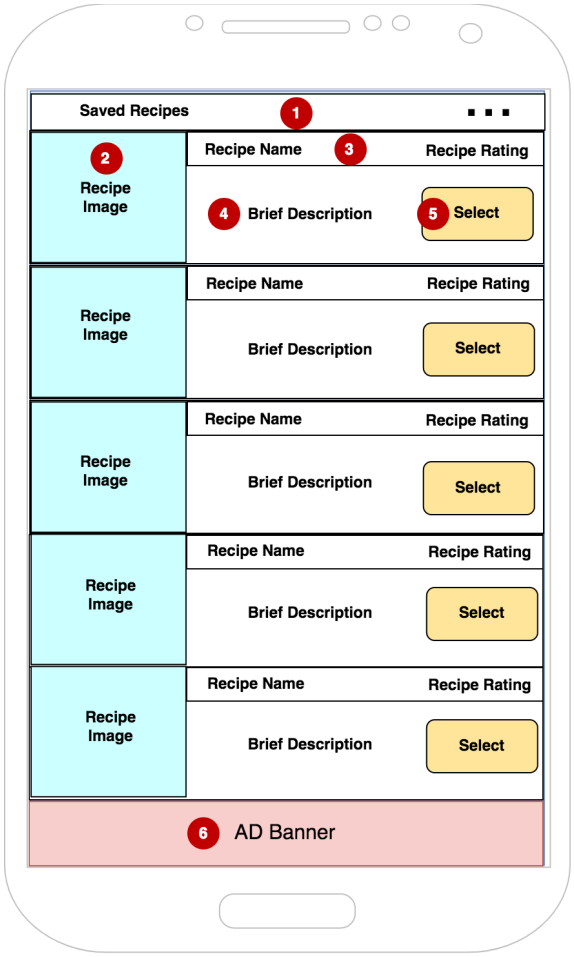

Meal Planner
1
2
Recipe Title
3
Recipe Picture
4
Recipe Description
Edit Recipe
5
6
Select Different Recipe
7
Planned Date: [date]
Planned Meal: [meal]
8
Edit
Add Ingredients to Shopping List
9
10
Cancel
11
Add to Meal Planner
12
AD Banner

Figure 3.7.4 Select Saved Recipe Screen Wireframe

Wireframe ID	Wireframe Feature Description
WF-MPS-01	Drop Down - Navigation / Menu Bar
WF-MPS-02	Image of recipe
WF-MPS-03	Recipe header with recipe name and rating info
WF-MPS-04	Brief recipe description
WF-MPS-05	Button to select recipe for meal planner use
WF-MPS-06	Ad Banner



3.8 Search for Recipe Wireframe

Figure 3.8.1 Search for Recipe Wireframe

Wireframe ID	Wireframe Feature Description
WF-SR-01	Drop Down - Navigation / Menu Bar
WF-SR-02	Button to update ingredients in pantry
WF-SR-03	Button to set search filters for this recipe search
WF-SR-04	Search bar where user can type in their search parameters
WF-SR-05	Image of recipe
WF-SR-06	Recipe header with recipe name and rating info
WF-SR-07	Brief summary of recipe
WF-SR-08	Ad banner

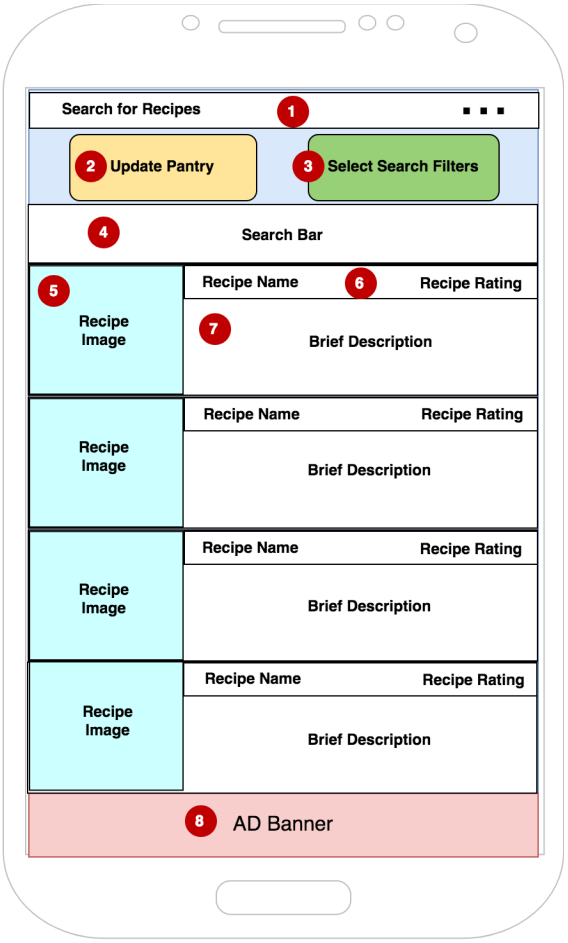
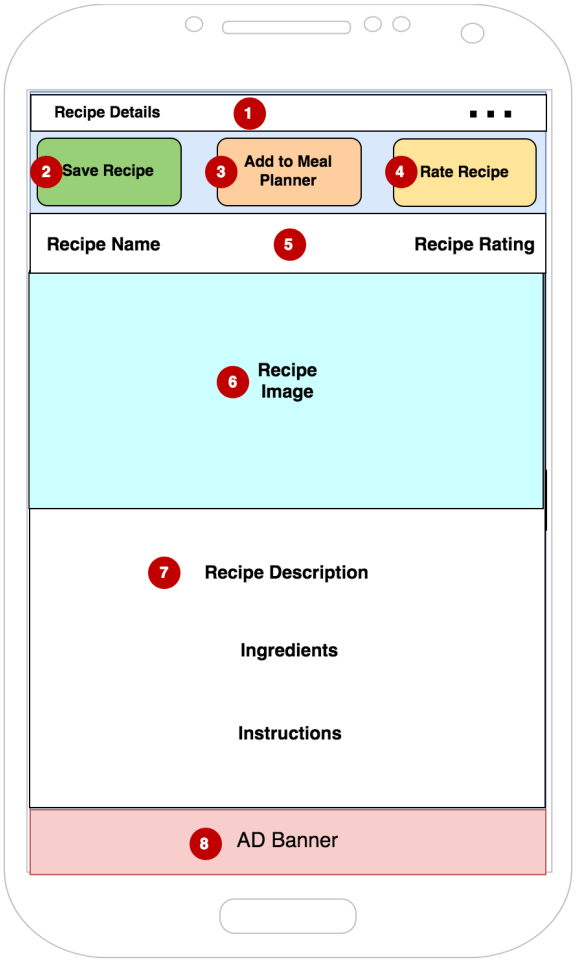


Figure 3.8.2 Recipe Detail Wireframe

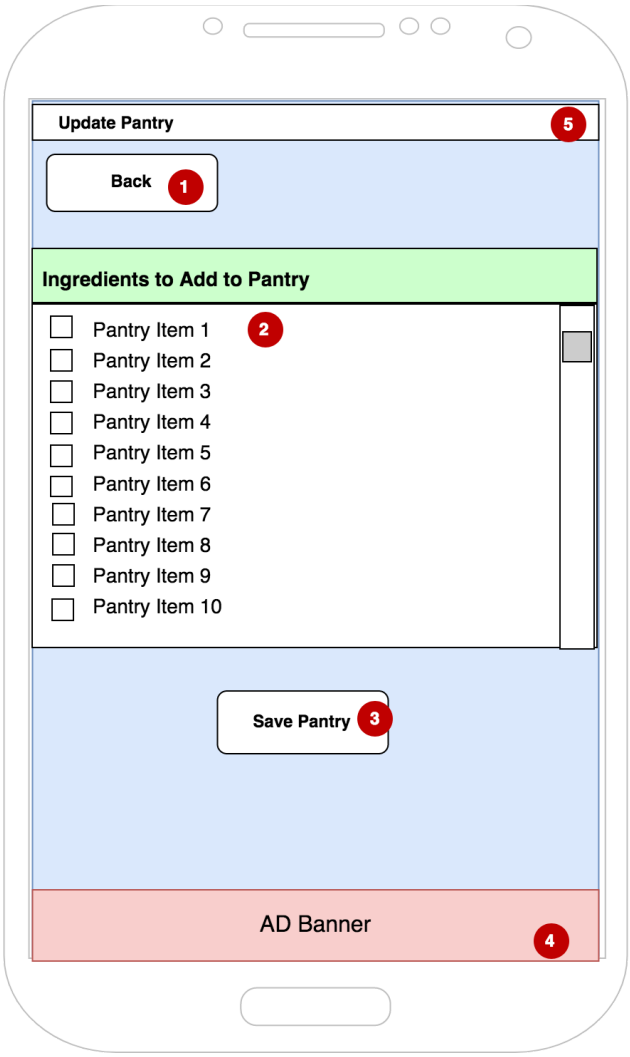
Wireframe ID	Wireframe Feature Description
WF-RD-01	Drop Down - Navigation / Menu Bar
WF-RD-02	Button to save recipe to Saved Recipes List
WF-RD-03	Button to add the recipe to the meal planner
WF-RD-04	Button to add a rating for the recipe
WF-RD-05	Recipe header with name and rating info
WF-RD-06	Image of recipe
WF-RD-07	Description, ingredients list, and instructions for recipe
WF-RD-08	Ad banner



3.9 Update Pantry Wireframe

Wireframe ID	Wireframe Feature Description
WF-UP-01	Back button - Goes back to the search preferences page
WF-UP-02	List of ingredients to add to pantry
WF-UP-03	Save Pantry - Saves the ingredients added to the pantry list
WF-UP-04	Ad Banner
WF-UP-05	Update Pantry Label

Figure 3.9.1 Update Pantry Wireframe



4. Architecture Design

4.1 Use Case Diagrams & Descriptions

4.1.0 Use Case Diagram

Figure 4.1.0.1 Use Case Packages Diagram

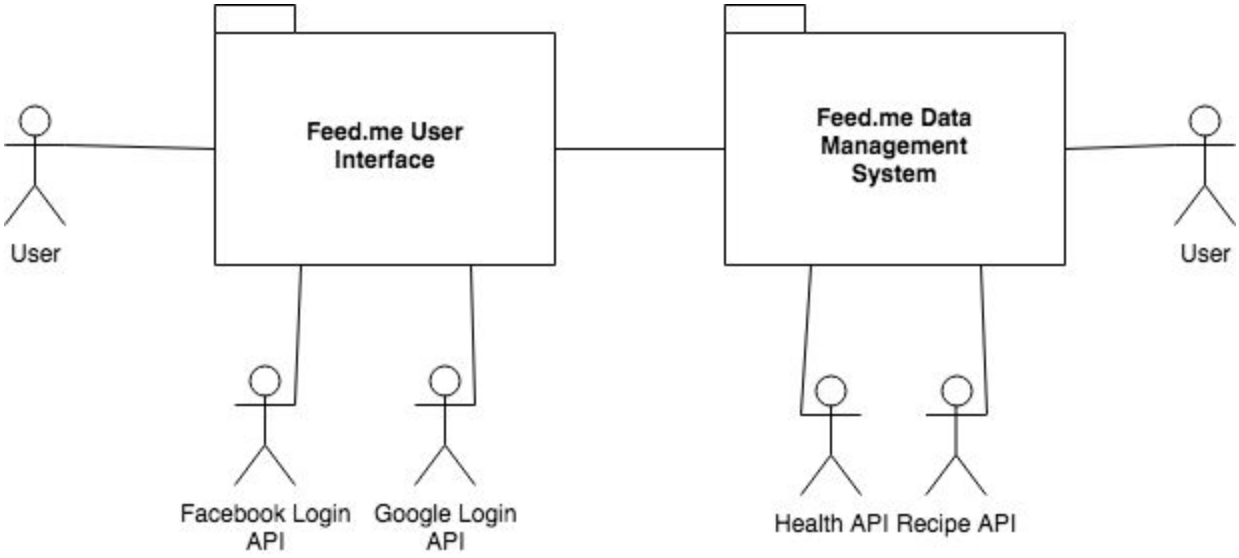
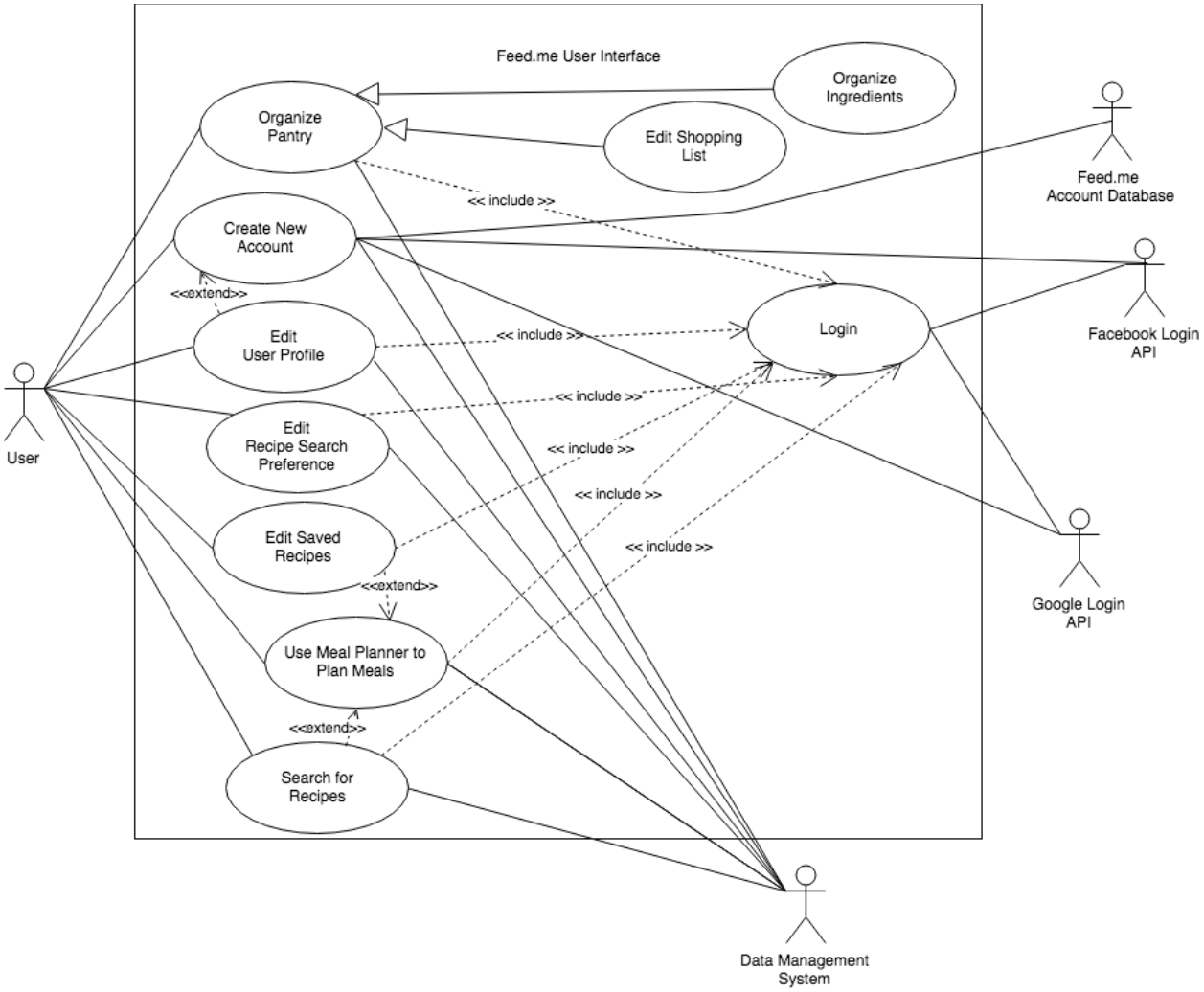
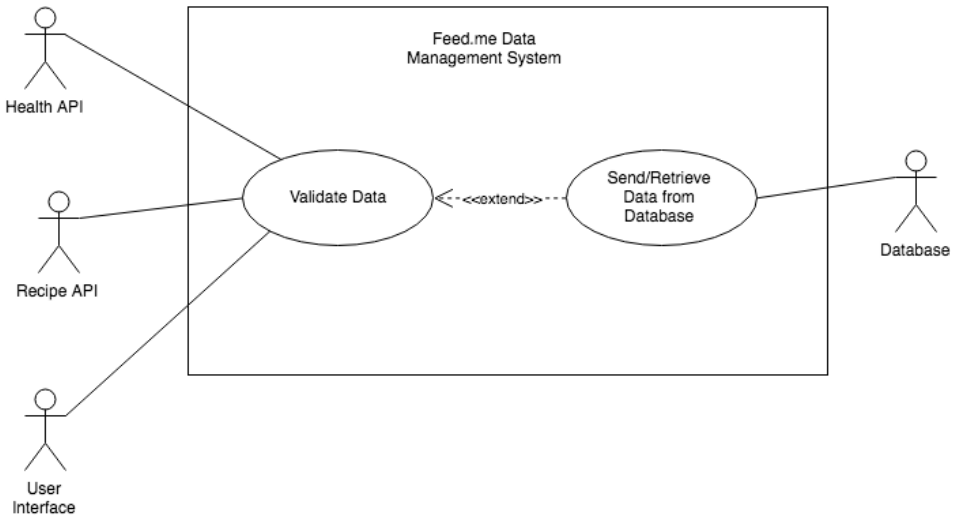


Figure 4.1.0.2 Use Case Diagrams





4.1.1 Create New Account

Name: Create New Account Use Case

Scenario:

1. User clicks Sign Up on Sign-In Screen of Feed.me app.
2. User is taken to Create Method Screen, where user clicks Sign Up With Email.
3. System will display the Collect User Information screen where they enter email, password, and username into the specified fields and clicks Next when finished.
4. App saves User Account information into a temporary session, gives user a message that they have successfully created a profile. User needs to verify the profile by clicking on the link sent via email.
5. Upon email validation, app adds user to the Feed.me Account Database.
6. Give user the option to click edit user preferences or exit. If the user want to edit user preferences, apply Edit User Profile Use Case.

Alternatives:

1. In Step 2, if user clicks Continue with Facebook, app will connect with Facebook API to gather credentials.
2. In Step 2, if user clicks Continue with Google, app will connect with Google API to gather credentials.
3. In Step 3, if APIs are unable to validate user, app displays “Unable to verify information. Please try again.”
4. In Step 4, if user leaves any field blank, app displays an error message at the top of the page that says “Please fill in all required fields.”
5. In Step 4, if the user enters an email address that is already in use, app displays “Sorry, that email address is already in use. Please enter a different email address.”
6. In Step 4, if the user selects a username that is already in use, app displays “Sorry, that username is taken. Please select a different email address.”

Figure 4.1.1.1 Activity Diagram

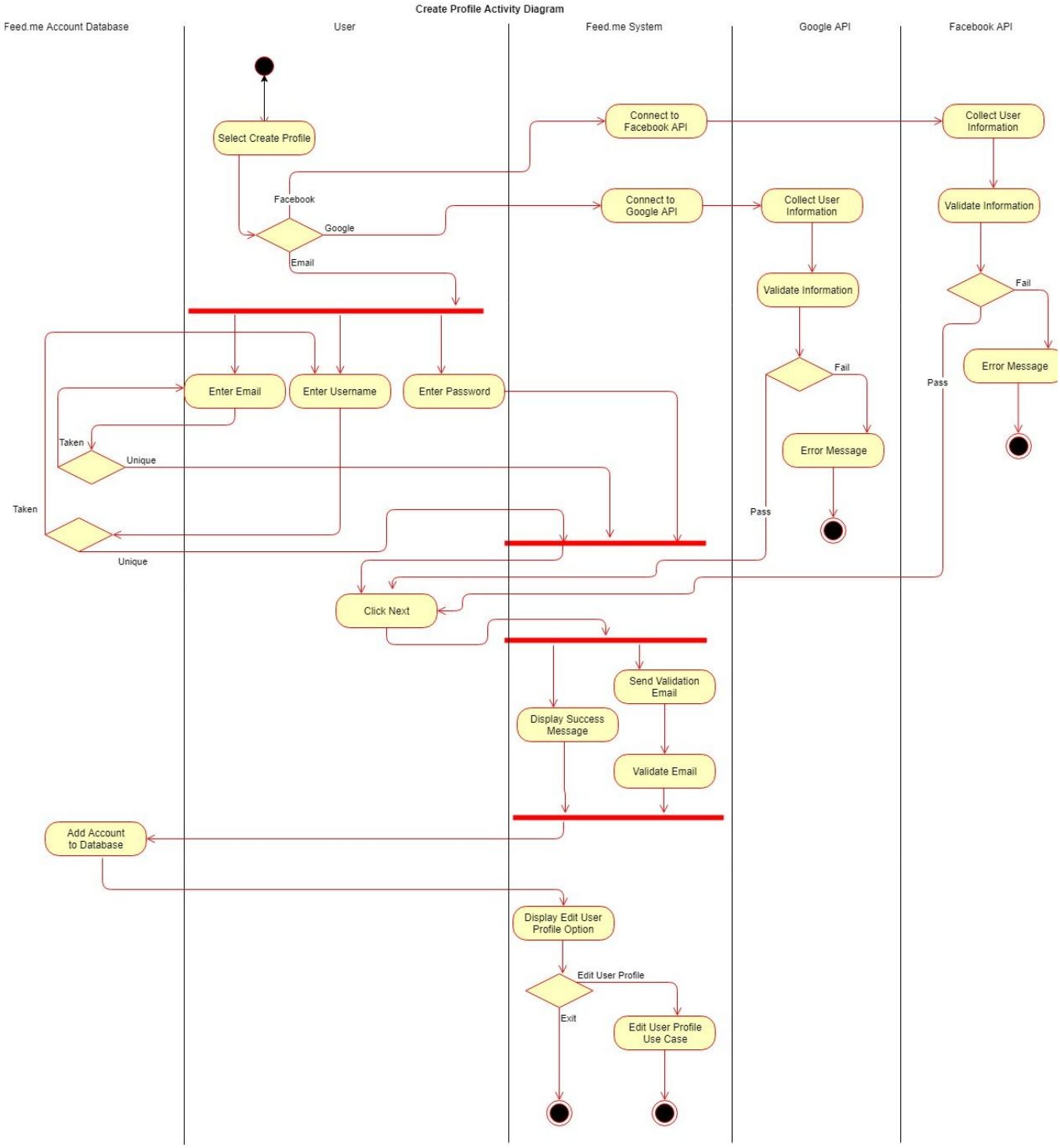


Figure 4.1.1.2 Robustness Diagram

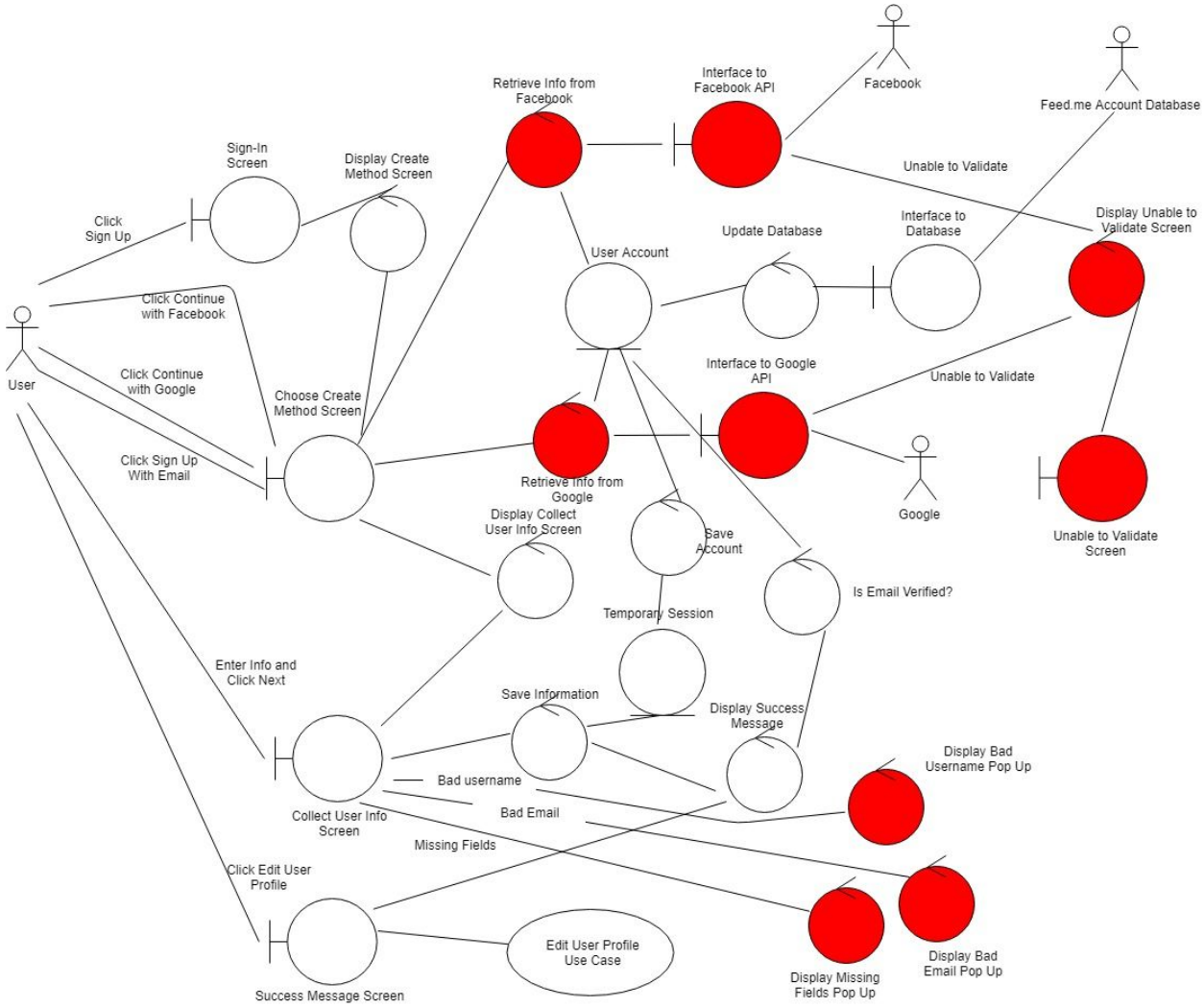
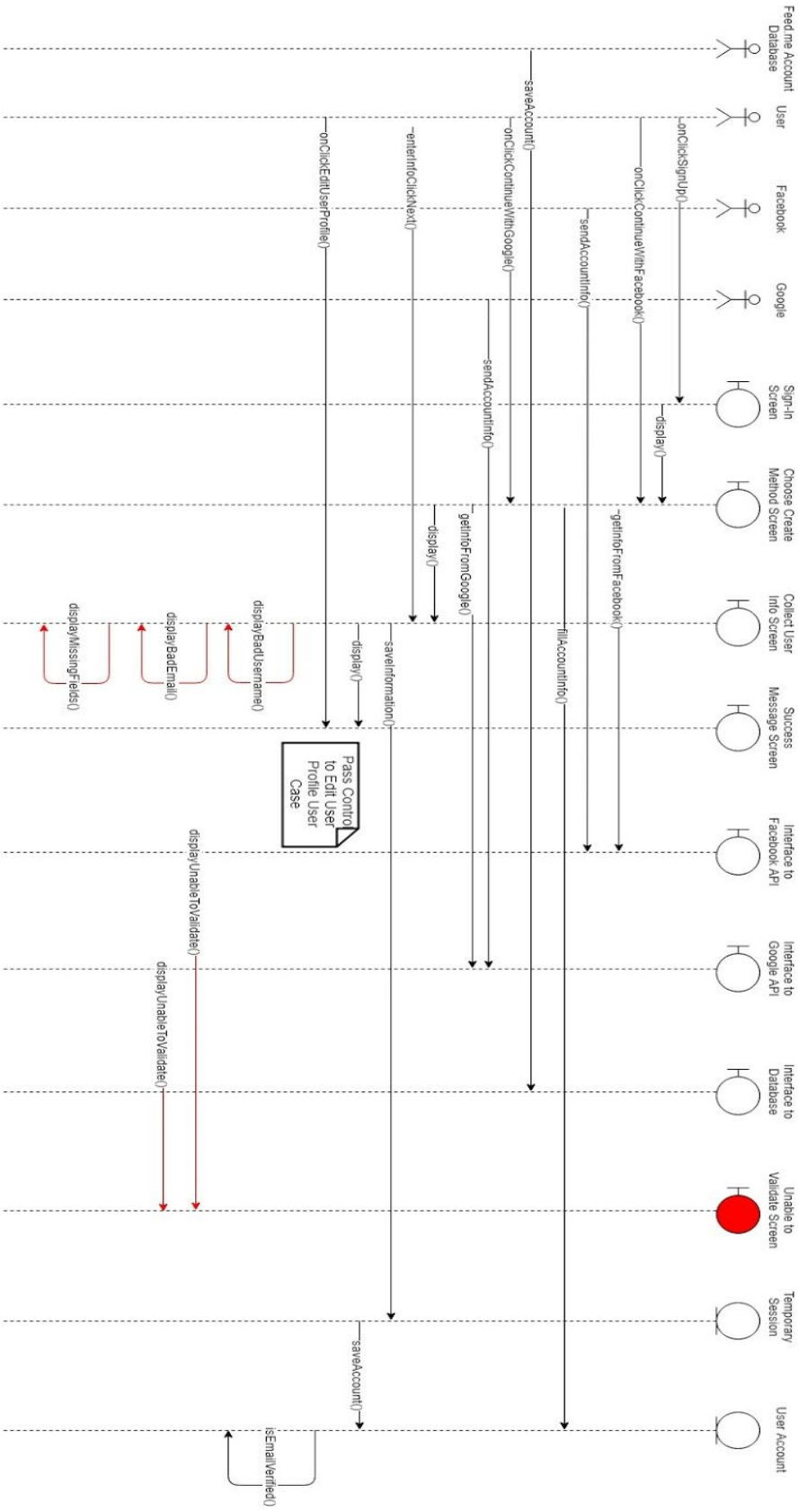


Figure 4.1.1.3 Sequence Diagram



4.1.2 Edit Saved Recipes

Name: Edit Saved Recipes

Scenario:

1. Apply Login use case.
2. User clicks Saved Recipe.
3. Feed.me User Interface proceeds to Saved Recipe screen.
4. User selects a recipe and clicks Edit.
5. Feed.me User Interface automatically loads the recipe ingredient details.
6. User taps on Edit name to change the name of the recipe.
7. User chooses to taps on Edit quantity on ingredients to change the number of servings.
8. User uses Add Ingredient icon to add more ingredients
9. User taps on Delete Ingredient icon to delete the ingredient.
10. User may choose to delete the whole recipe by clicking Delete Recipe button located at the end.
11. User can select Cancel to terminate any changes he/she just made, and return to step 3.
12. After user has done editing the recipe, user taps Save to save any changes he/she just made.

Alternatives:

1. In step 6: If user leaves the Recipe Name field in blank space, the system automatically restores to the last edited recipe name.
2. In step 7: If user enters 0 on Ingredient Quantity field, that ingredient will automatically be deleted.

Figure 4.1.2.1 Activity Diagram

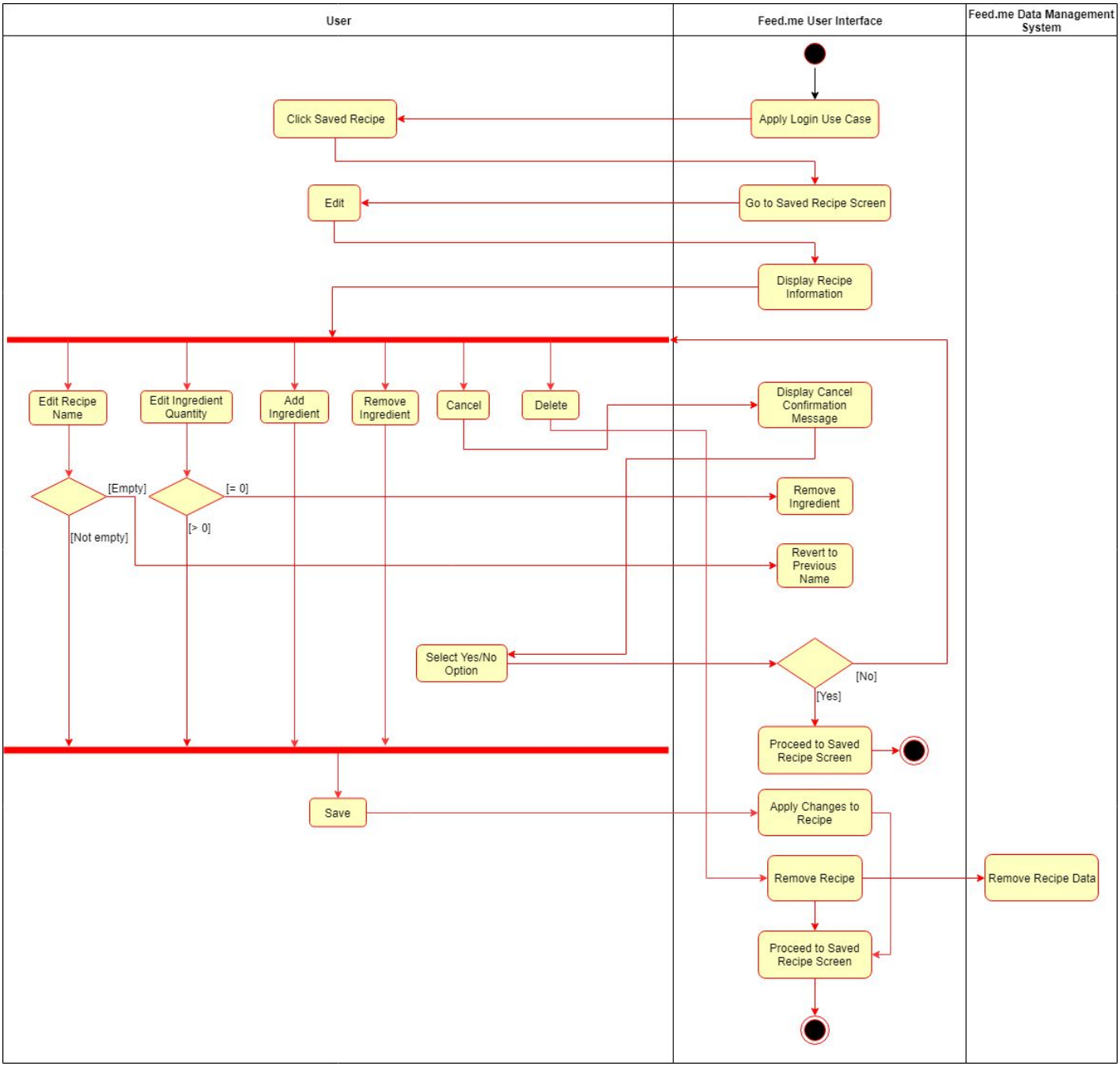


Figure 4.1.2.2 Robustness Diagram

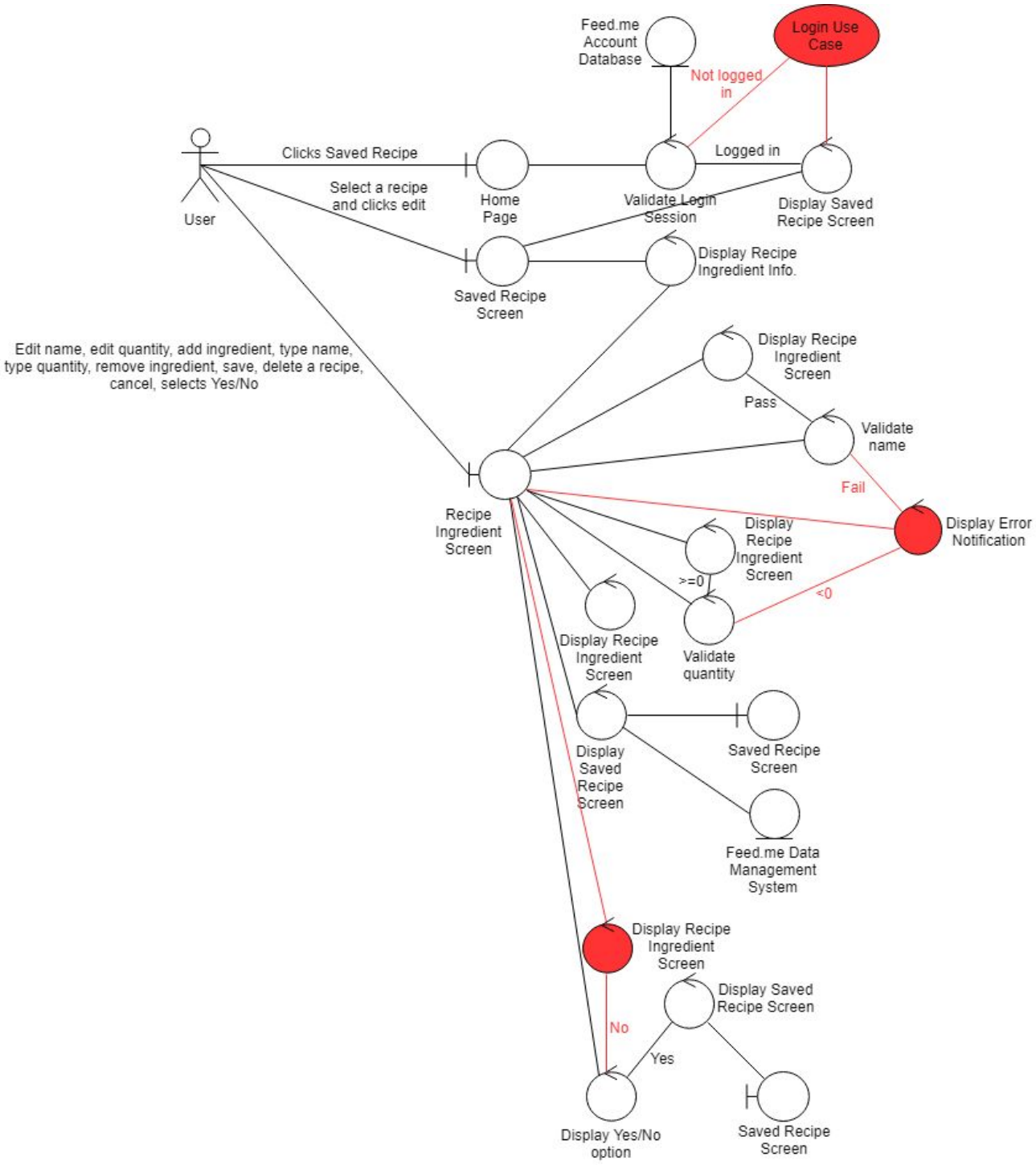
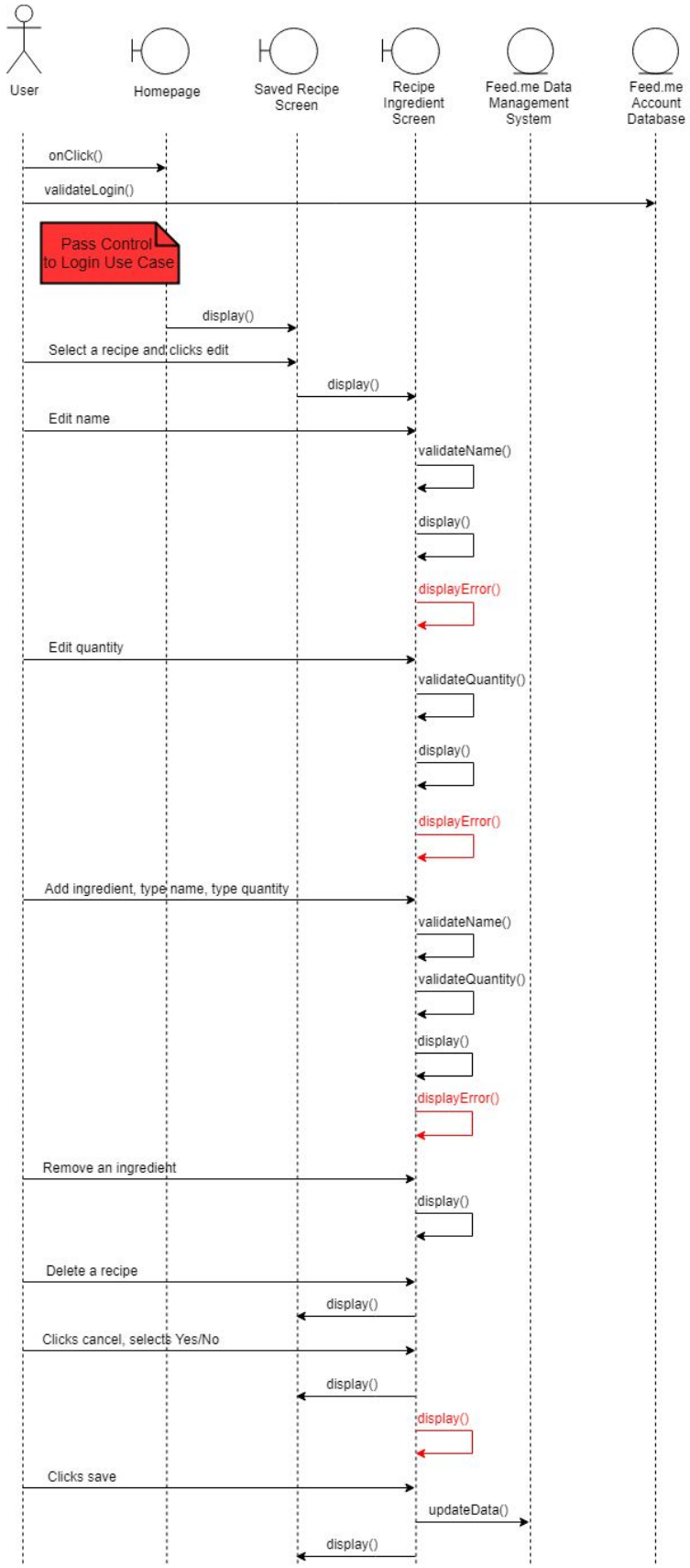


Figure 4.1.2.3 Sequence Diagram



4.1.3 Edit Recipe Search Preferences

Name: Edit Recipe Search Preferences

Scenario:

1. Apply Login use case.
2. User selects to go to Search Preferences screen from drop down menu on Home screen.
3. Saved recipe search preferences automatically preloaded from user profile database and appropriately selected from saved preferences in user profile database by System.
4. System automatically loads pantry ingredients from the pantry list from pantry database and available for selection to bottom of screen.
5. User may choose to select or deselect any of the user search preferences for current searching. Categories include but not limited to: lifestyle preference (ie. vegan, keto), preference for number of pantry ingredients used in search, cuisine style preference, health preferences, etc.
6. User may select to update pantry ingredients list. If not skip to step 8.
7. Apply Organize Pantry use case.
8. User may choose to select or deselect pantry ingredients for preferences to use for current searching.
9. User may select to save appropriate preferences to User Profile (not all are saveable). User preferences that are not available to save in the user profile will remain unsaved and revert to defaults after session ends. If not, skip to step 11.
10. Apply Edit User Profile use case.
11. When user leaves screen selections remain until user logs off.
12. When user logs off unsaved preferences are returned to default status.

Alternatives:

1. In step 3, if user has no preferences saved, there will be no preferences pre-selected.
2. In step 4, if user has no stored pantry ingredients there will be no pantry ingredients preloaded.
3. In step 5, if number of pantry ingredients to use in search exceeds number of pantry ingredients that have been selected to search on (ie. recipe must include 5 pantry ingredients, but only carrots and onions are selected from the list of pantry ingredients as ingredients to use in the search) then system will prompt user to add more pantry ingredients to pantry list or reduce number of ingredients to use, until this is resolved.
4. In step 8, if user has no stored pantry ingredients there will be no list to select or deselect ingredients from. System will display Not Found Message - No ingredients Available - List Empty

Figure 4.1.3.1 Activity Diagram

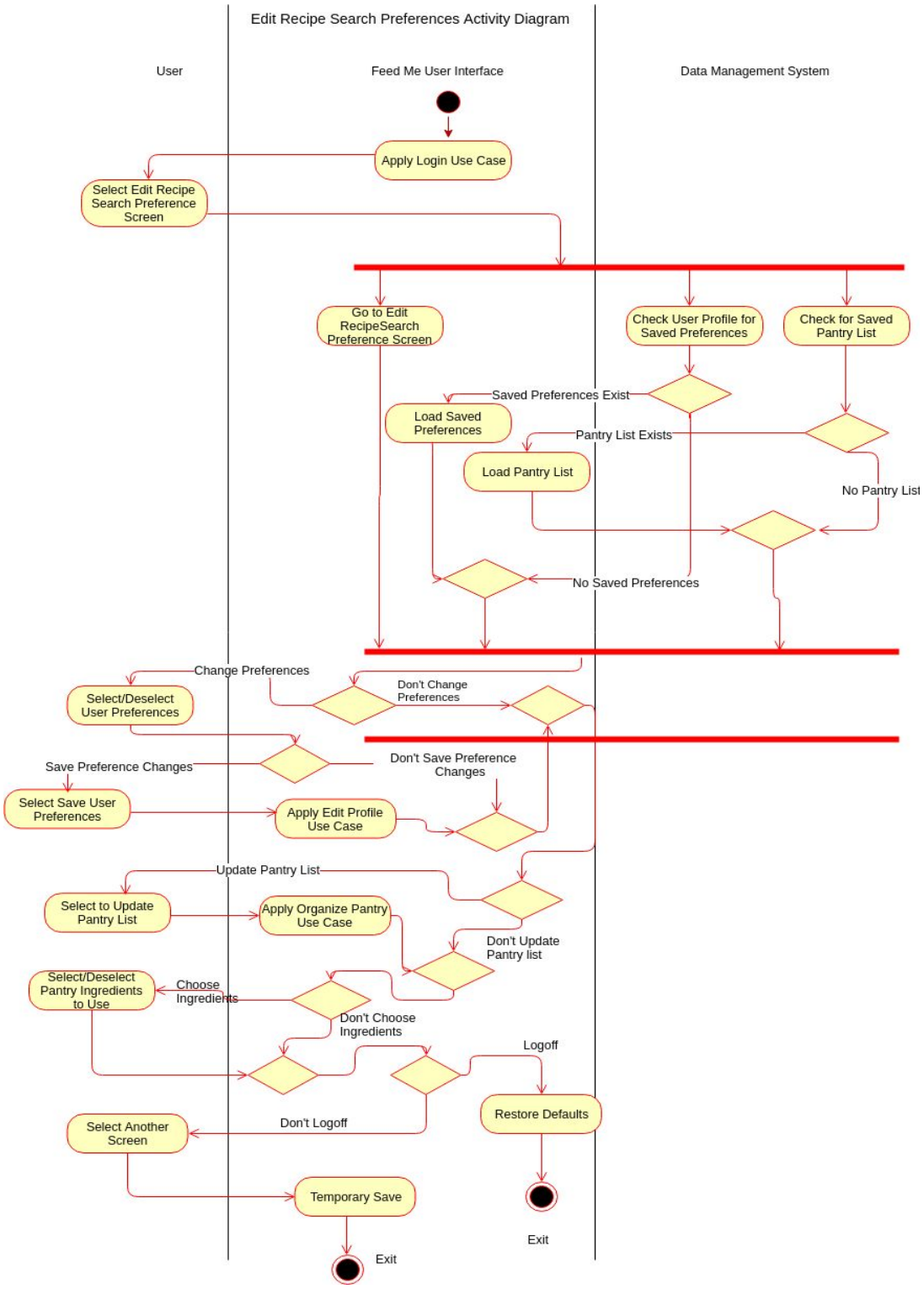


Figure 4.1.3.2 Robustness Diagram

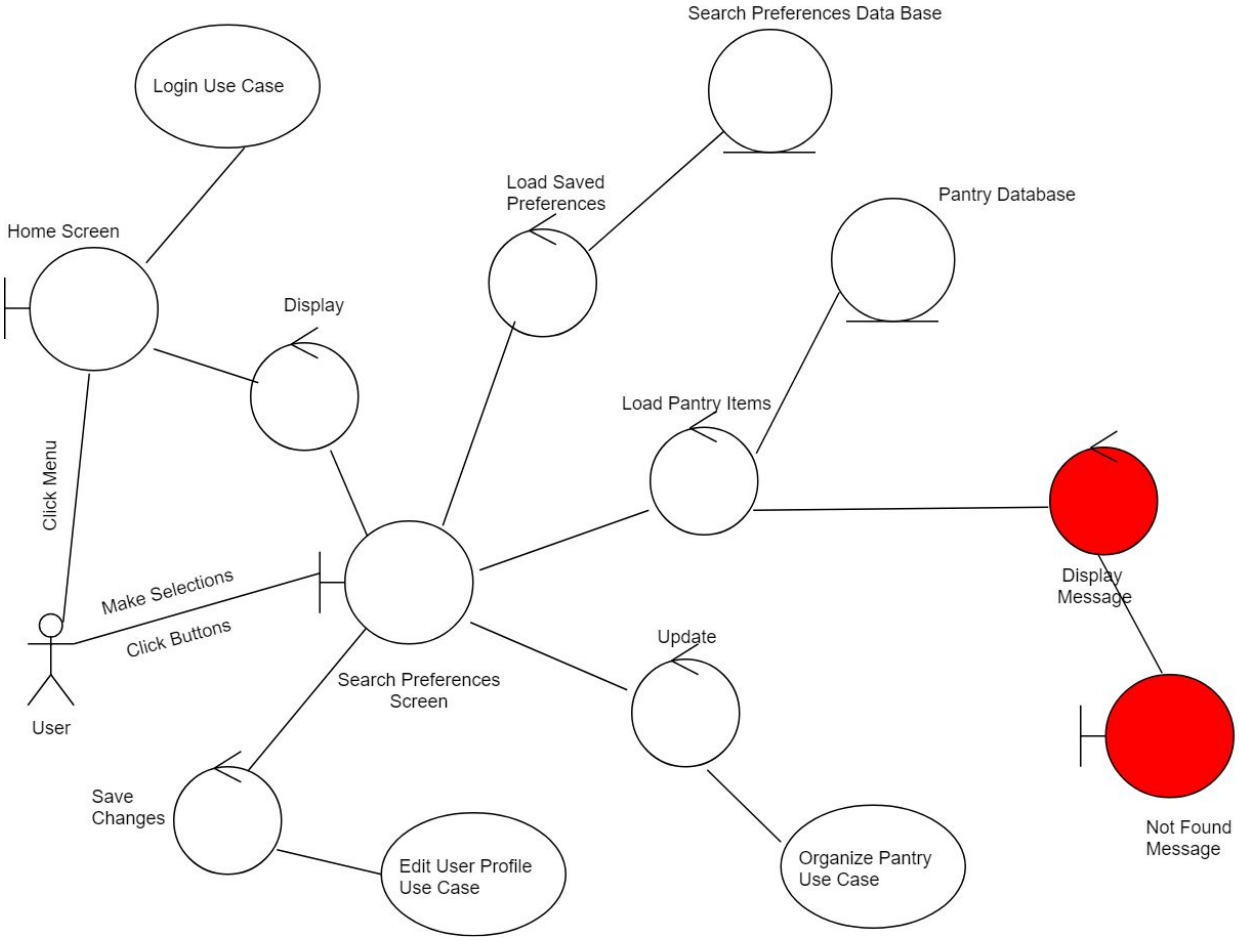
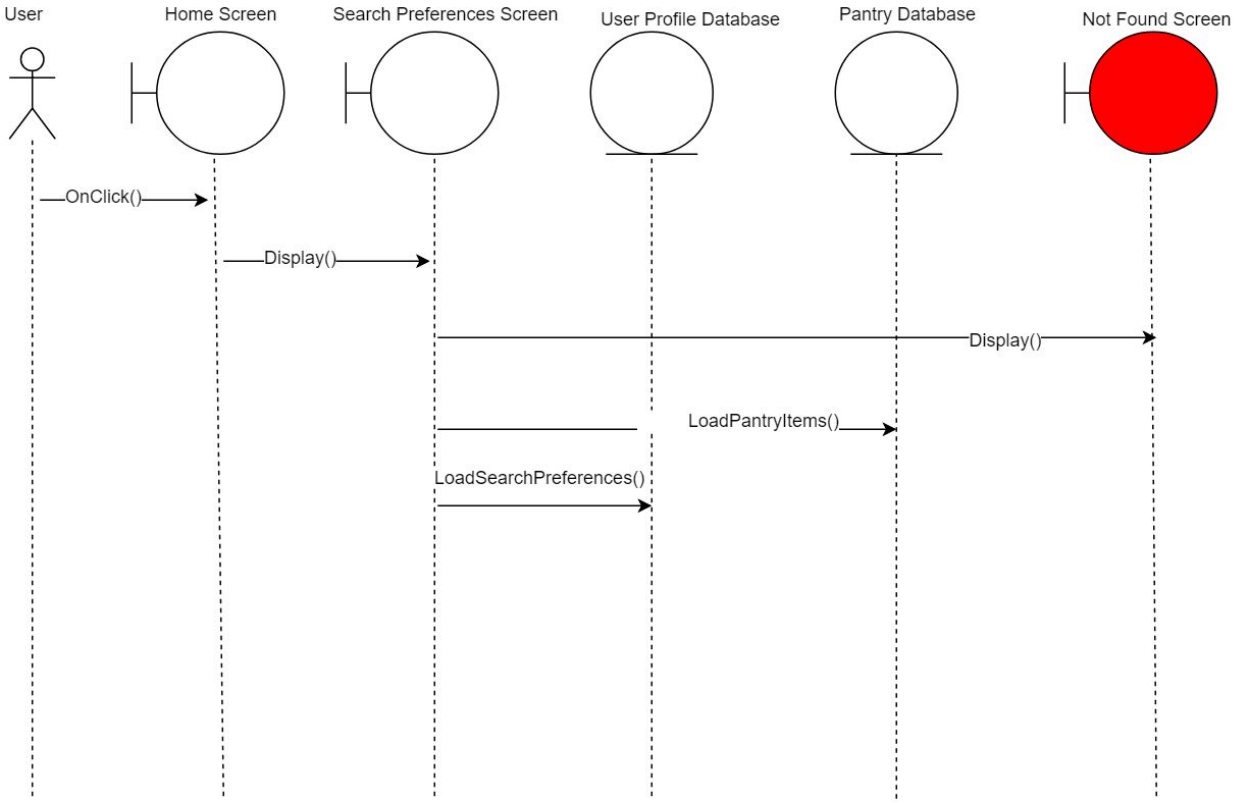


Figure 4.1.3.3 Sequence Diagram



4.1.4 Search for Recipes

Name: Search for Recipes Use Case

Scenario:

1. Apply Login use case.
2. User selects Search for recipe option on the app.
3. If user wants to update Recipe Search Preferences, choose Update Search Preferences. Else skip to step 5.
4. Apply Edit Recipe Search Preferences use case.
5. If user wants to update pantry ingredients - select Update Pantry. Else skip to step 7.
6. Apply Organize Pantry use case.
7. Recipe search automatically loaded with available pantry ingredients selected in Recipe Search Preferences.
8. User can select to use meal planner filters in recipe search.
9. User can select to use user profile health filters in recipe search.
10. User selects for recipe search to begin. System responds to user with a list of recipes corresponding to the ingredients and preferences given by the user.
11. The user may select to add a recipe to their meal plan. If not skip to step 13.
12. Apply the Use Meal Planner use case.
13. User may select to save a recipe to their saved recipe database. If not skip step 15.
14. Apply Edit Saved Recipes use case
15. The system checks the user's pantry and prompts the user if they would like some/all ingredients (not already in pantry) from the selected recipe to be added to one of the users shopping lists. If no, end of scenario.
16. User selects list and ingredients to add to list and System saves selected ingredients to user's shopping list.

Alternatives:

1. In step 3, user can go back to add more ingredients when searching for recipe and perform a recipe search again.
2. In step 3, user enters an ingredient that does exist, the system will ask the user to verify the ingredient entered.
3. In step 4, the system checks with the API to find available recipes. If there are no available recipes, the system will notify the user of no available recipes.

Figure 4.1.4.1 Activity Diagram

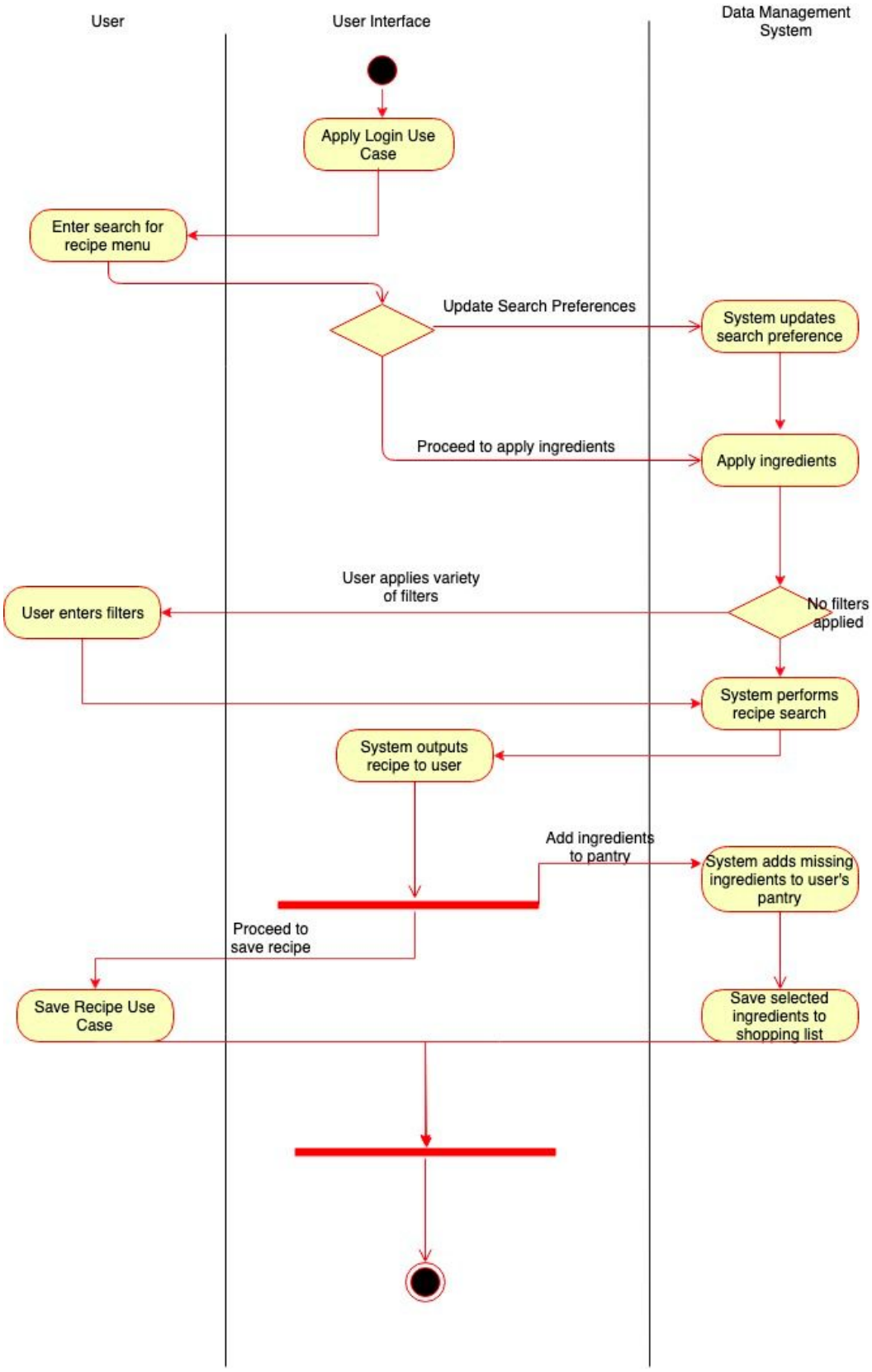


Figure 4.1.4.2 Robustness Diagram

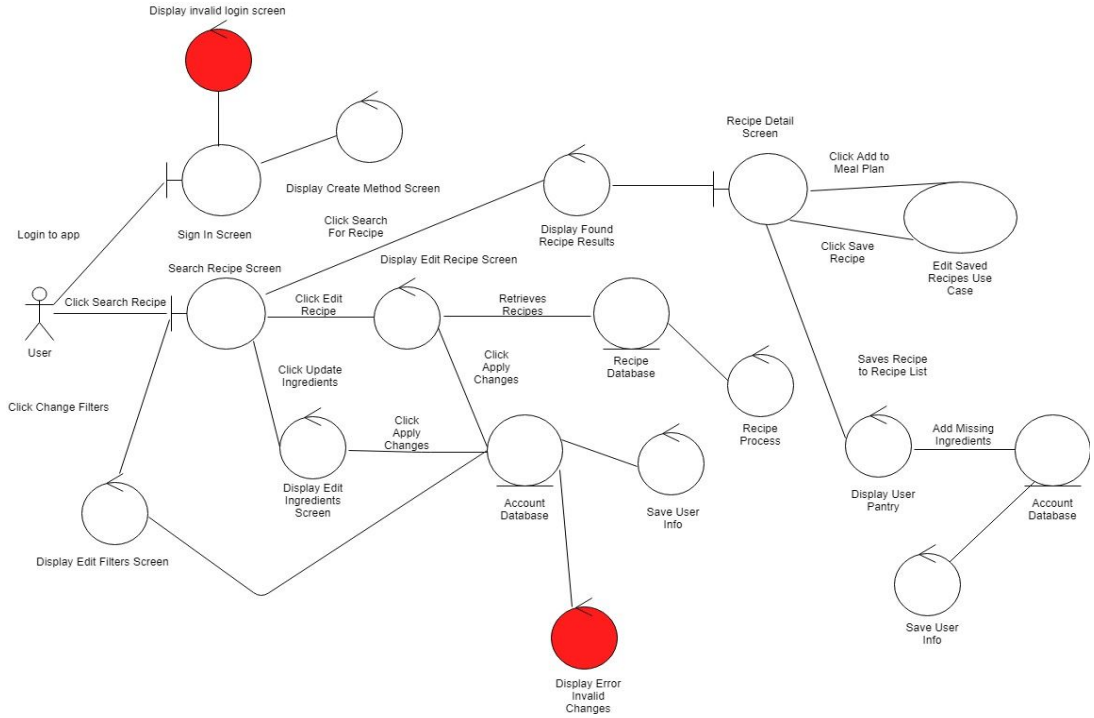
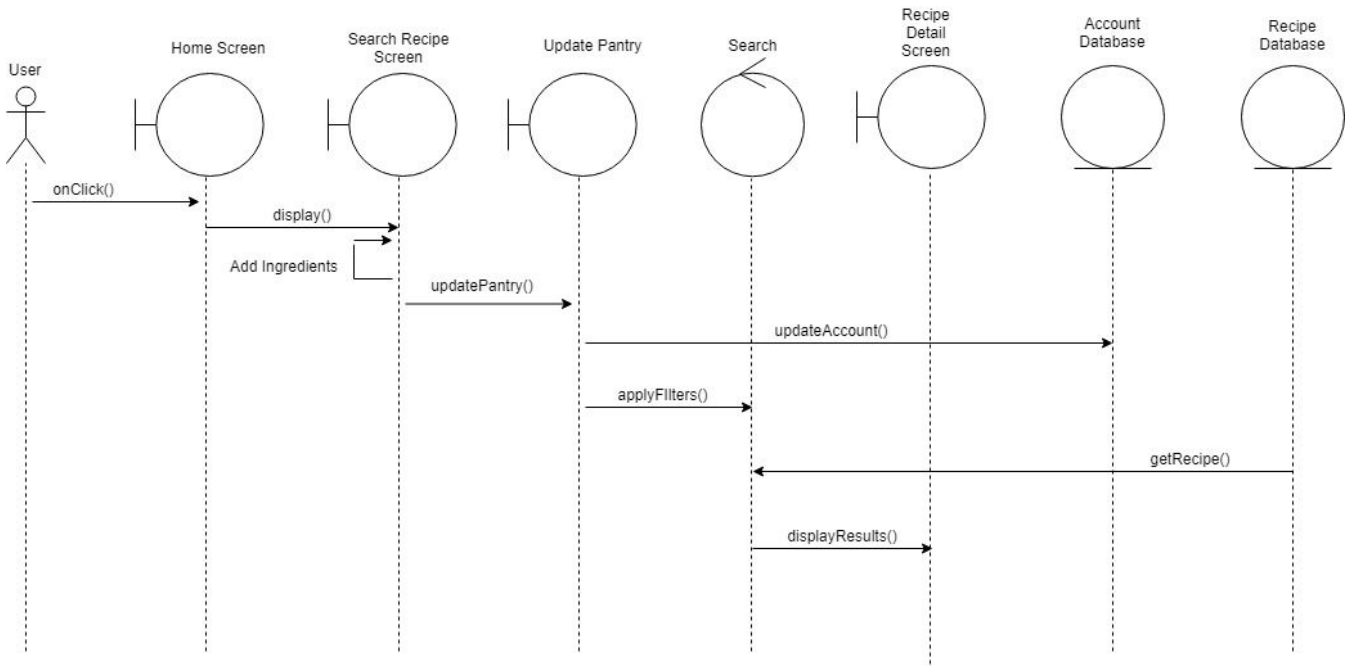


Figure 4.1.4.3 Sequence Diagram



4.1.5 Use Meal Planner Use Case

Name: Use Meal Planner to Plan Meals

Scenario:

1. Apply Login use case.
2. User selects Meal Planner from menu.
3. User selects add new meal to plan to create a new planned meal.
4. User selects the day and the meal time they want to plan on that day (i.e. Breakfast, Lunch, Dinner, Dessert/Snack).
5. User selects search for recipes option to find a recipe for their planned meal. Apply Search for Recipes Use Case to find the recipe.
6. User is shown the planned meal details and selects add to meal planner button.
7. Apply Edit Shopping List Use Case to add ingredients to their shopping list(s).
8. User is returned to home screen.

Alternatives:

1. In step 3, the user has the option to edit a planned meal. Apply Edit Saved Recipes Use Case.
2. In step 3, the user has the option to delete a planned meal. This deletes the planned meal from the database and refreshes screen to remove that meal's details.
3. In step 3, the user has the option to return to home screen.
4. In step 5, the user has the option to select a saved recipe. If the user has saved recipes, they are presented with their list of saved recipes and they select one. Then user proceeds to step 6. If they do not have saved recipes, a pop-up message explains they do not have saved recipes and they're redirected to the Search for Recipes Use Case.
5. In step 6, the user has the option to edit the selected recipe. Apply Edit Saved Recipes Use Case.
6. In step 6, the user has the option to select a different recipe, returning them to the Search for Recipes Use Case.
7. In step 6, the user can edit the selected day and meal for the plan, returning them to the previous screen to reselect day and meal time.
8. In step 6, the user has the option to deselect the add ingredients to shopping list checkbox. Then they are directly returned to the home screen after pressing the add to meal planner button.
9. In step 6, the user has the option to cancel the meal plan in progress, returning them to the home screen.

Figure 4.1.5.1 Activity Diagram

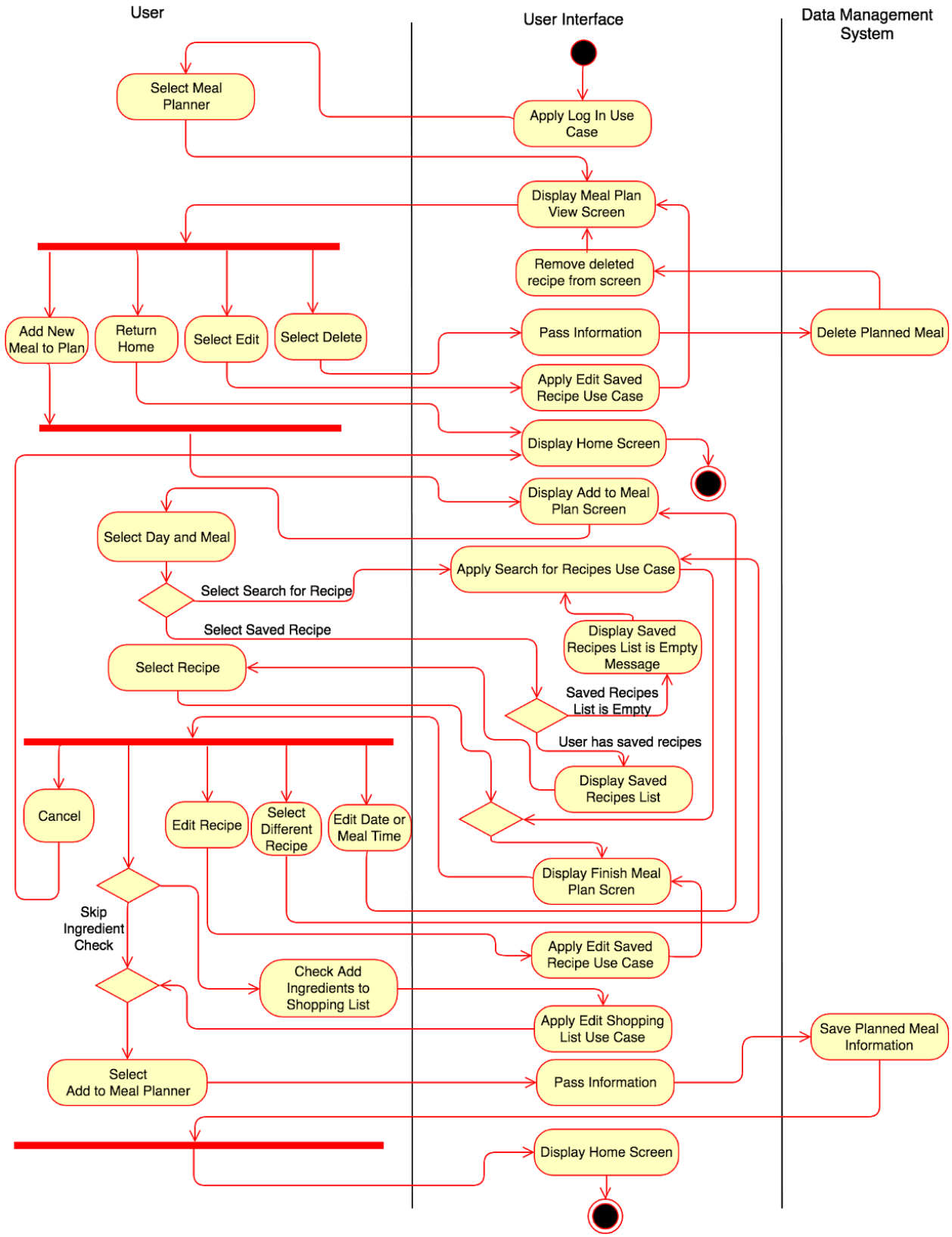


Figure 4.1.5.2 Robustness Diagram

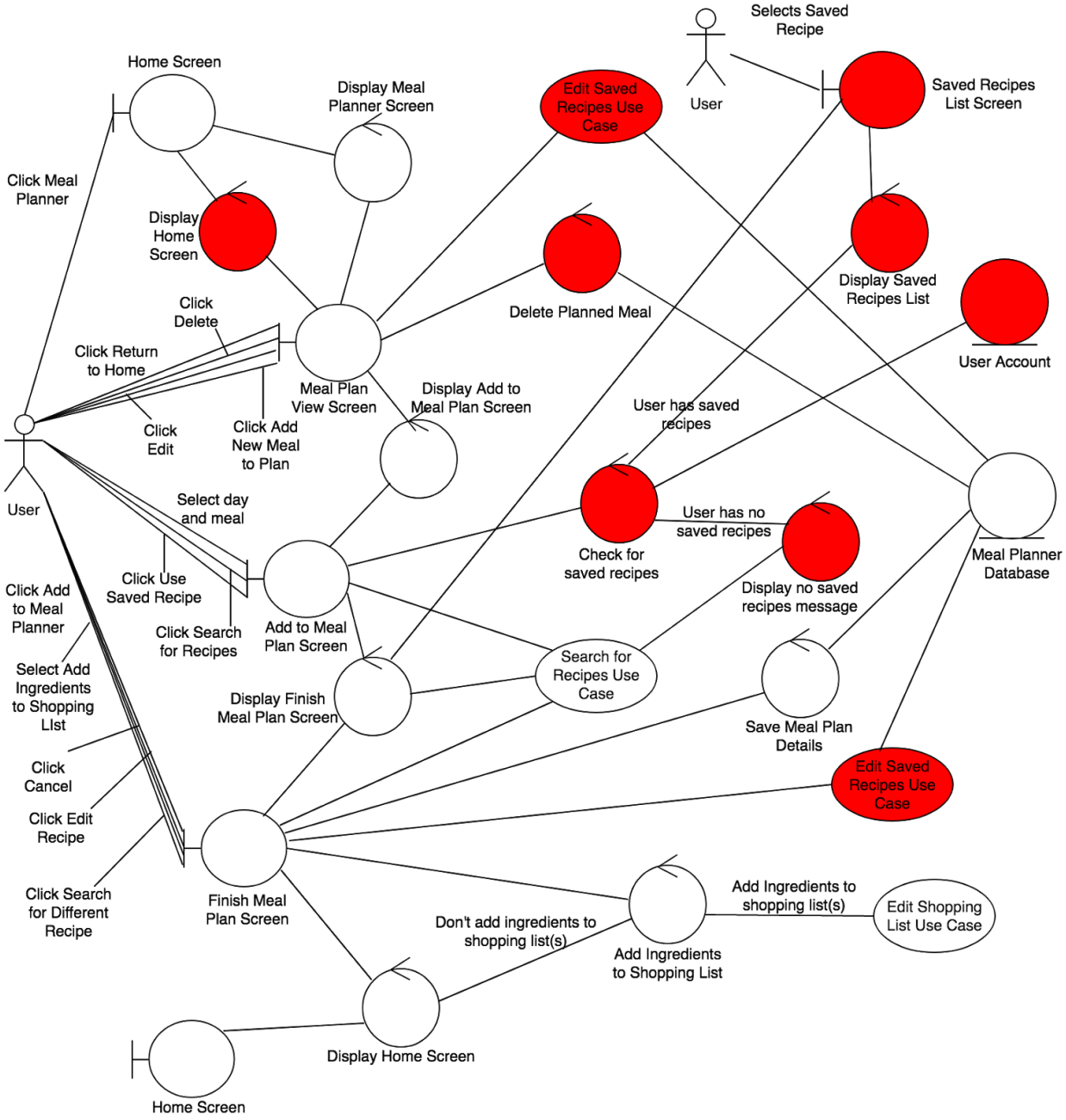
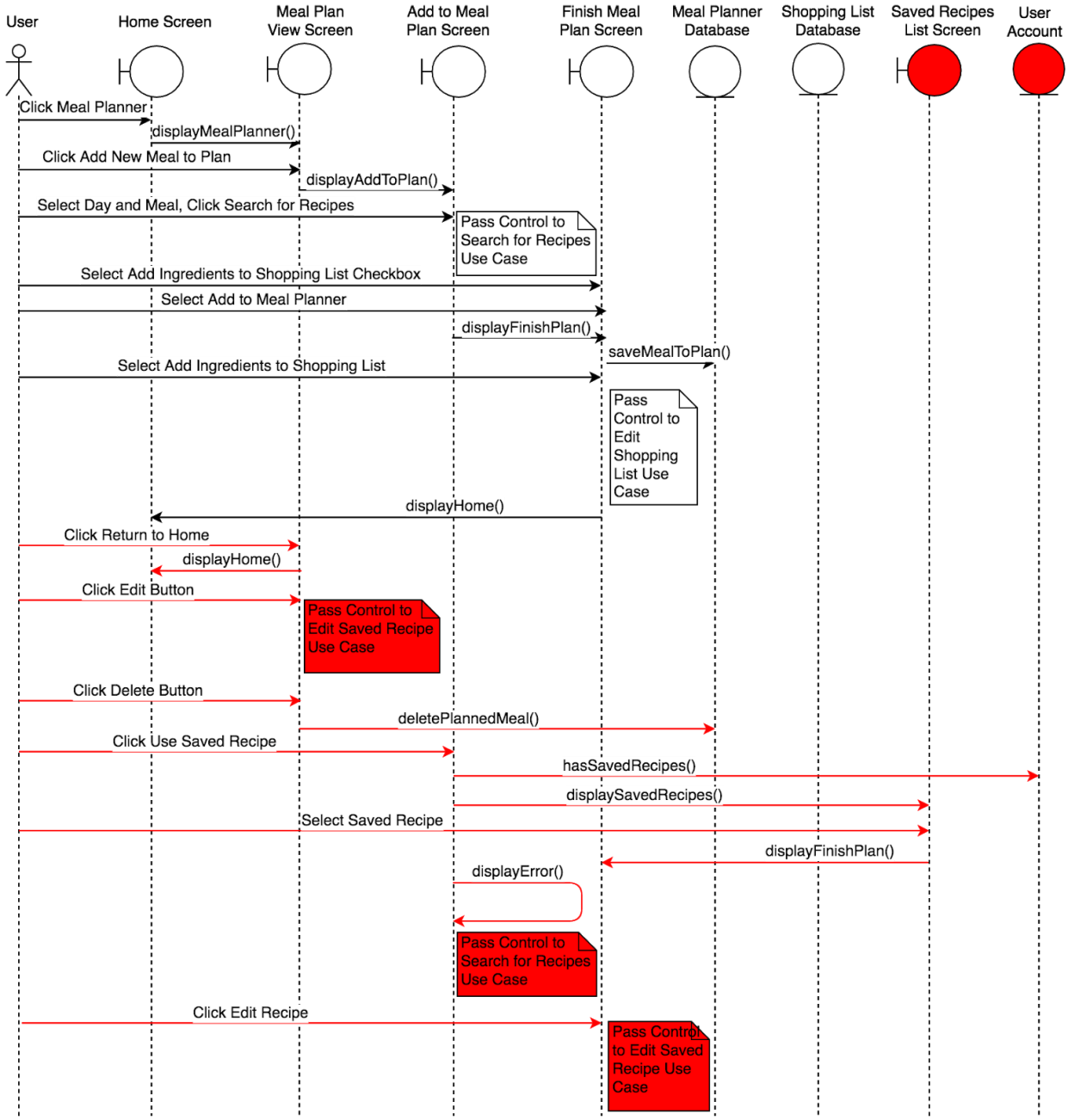


Figure 4.1.5.3 Sequence Diagram



4.1.6 Edit Shopping List

Name: Edit Shopping List Use Case

Scenario:

1. User selects a Shopping List to edit from the Shopping Lists screen by clicking the shopping list name.
2. System displays the full name of the shopping list and the list of Ingredients.
3. User can exit to the Shopping Lists screen and go back to step 2, or continue making changes to the shopping list in step 4.
4. User can select to rename the shopping list and type in a new name in step 5, or continue to step 6.
5. User types in a new name for the Shopping List.
6. User can add an Ingredient by manually typing the ingredient name, or by clicking on ingredient name in the list of most recently used ingredients.
7. User can select an ingredient/s from the shopping list to delete.
8. Every new addition or deletion is saved automatically.
9. User can make additional changes to the shopping list or exit to Shopping Lists screen.

Alternatives:

1. In Step 5, if the user changes the Shopping List name and the new name already exists in the user's list of Shopping Lists, the system displays a message, and the User is asked to type in a new name. If the user doesn't type a unique new name, the Shopping List name will revert to the previous valid name.

Figure 4.1.6.1 Activity diagram

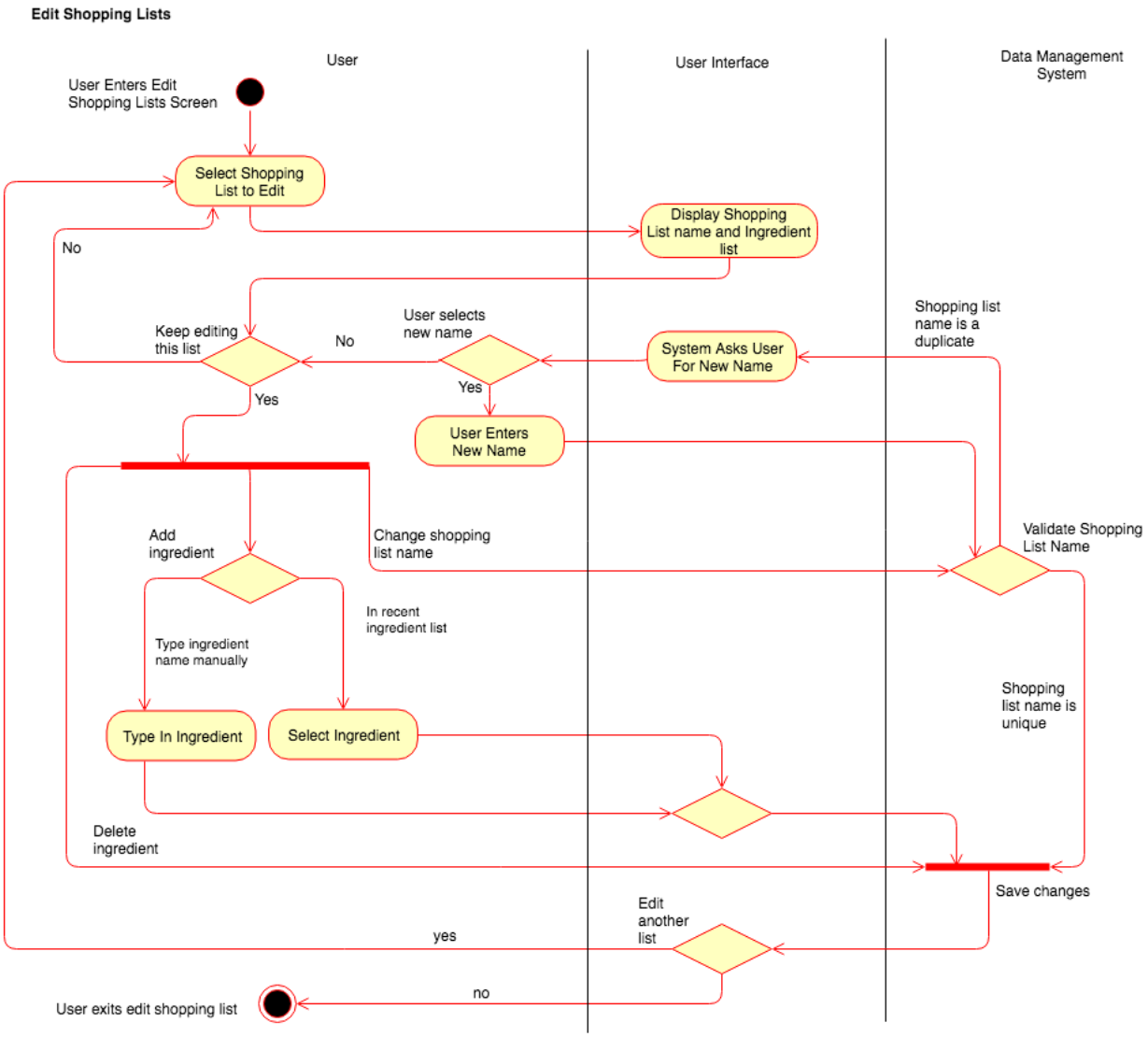


Figure 4.1.6.2 Robustness Diagram

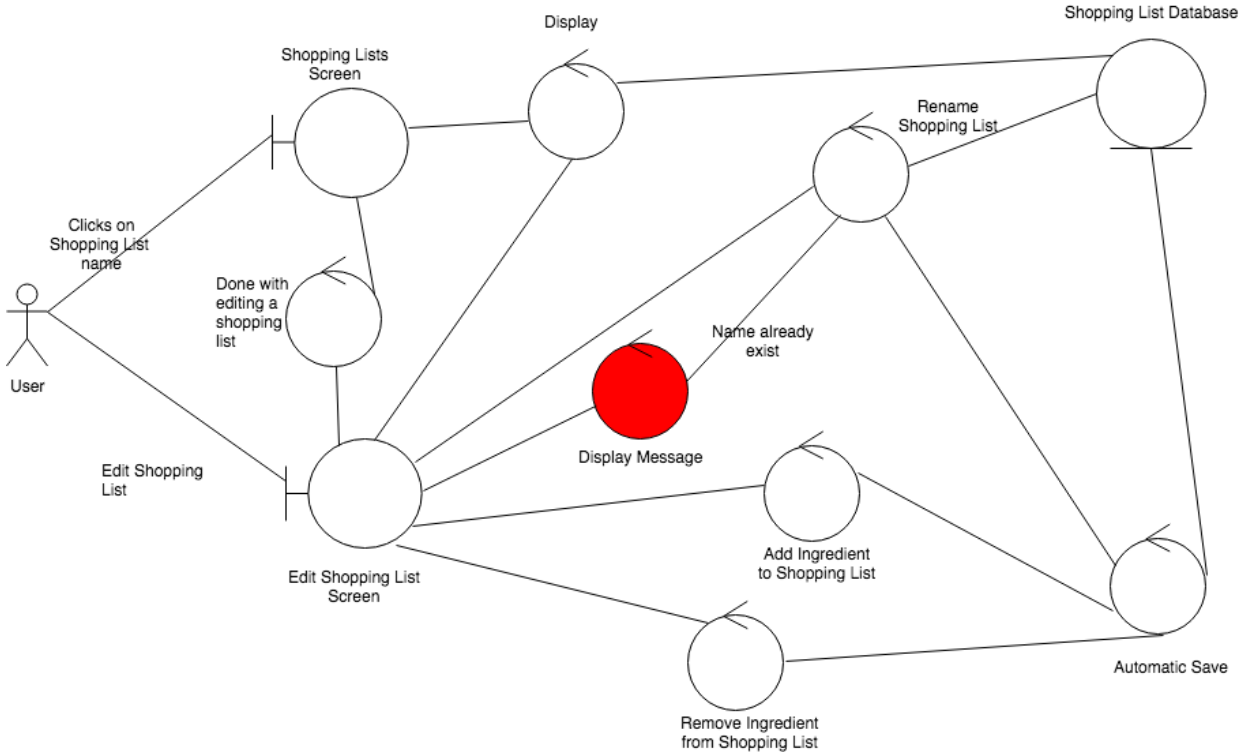
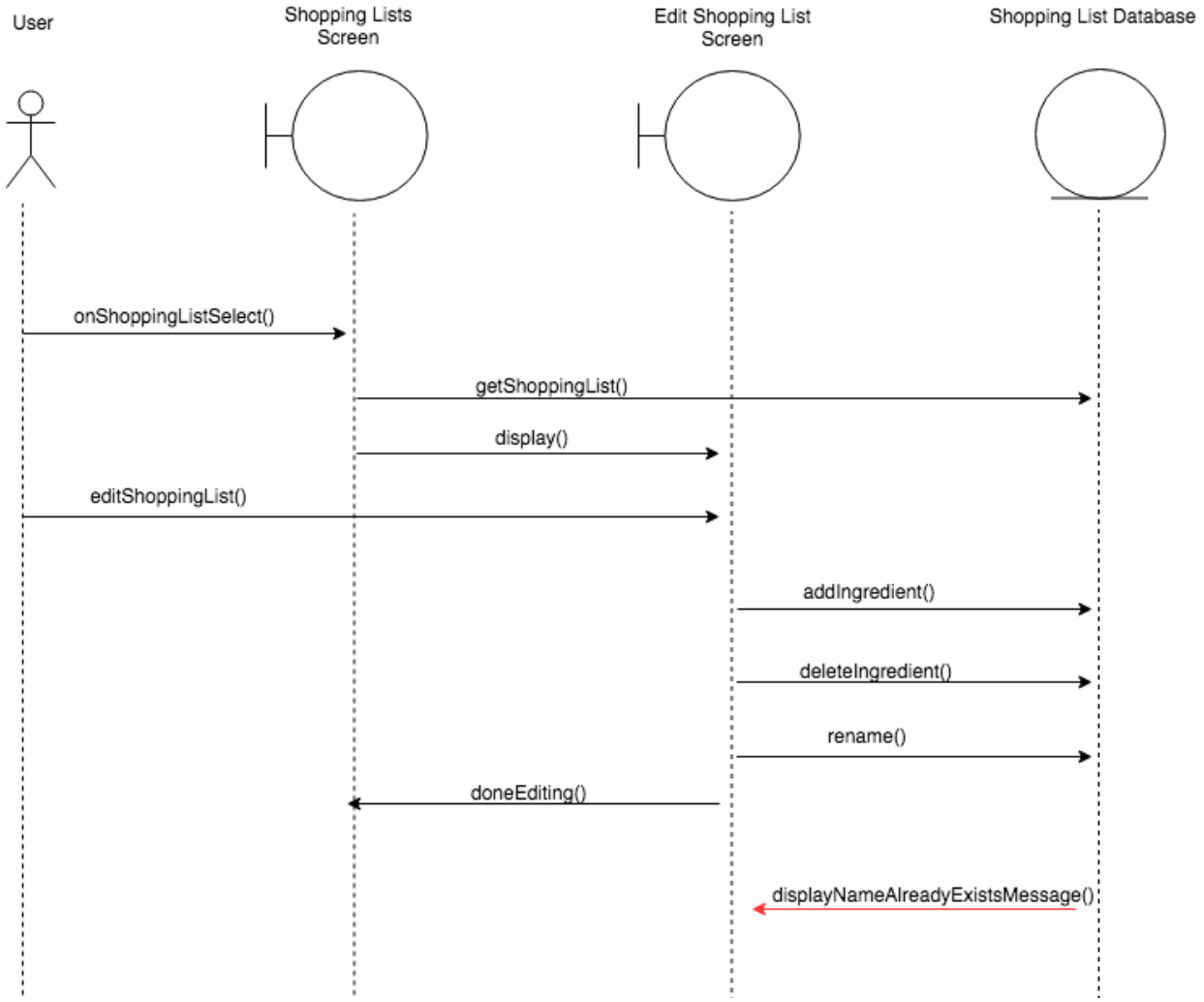
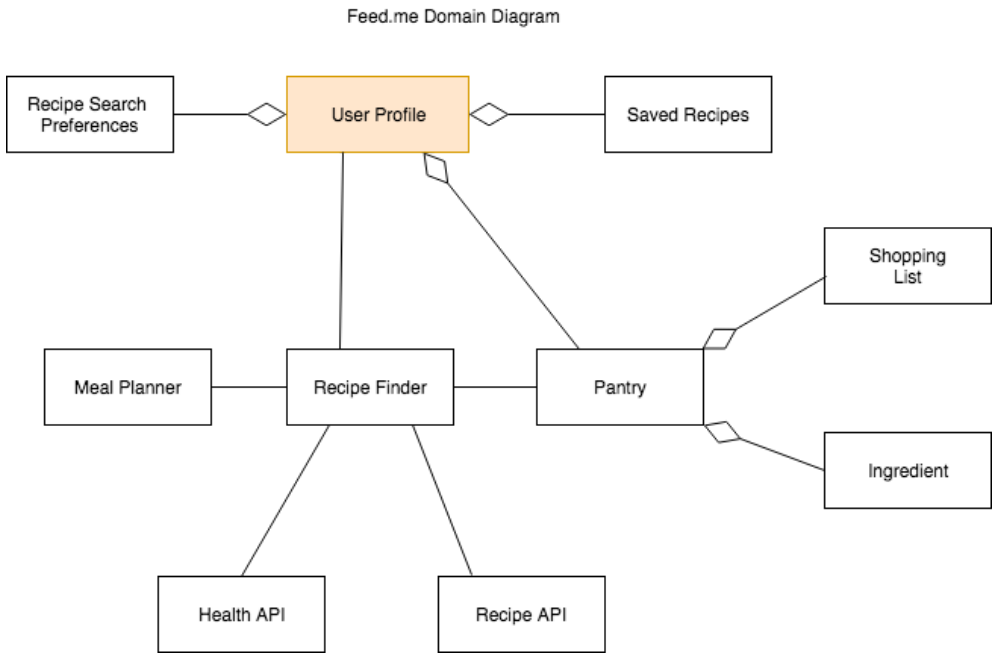


Figure 4.1.6.3 Sequence Diagram

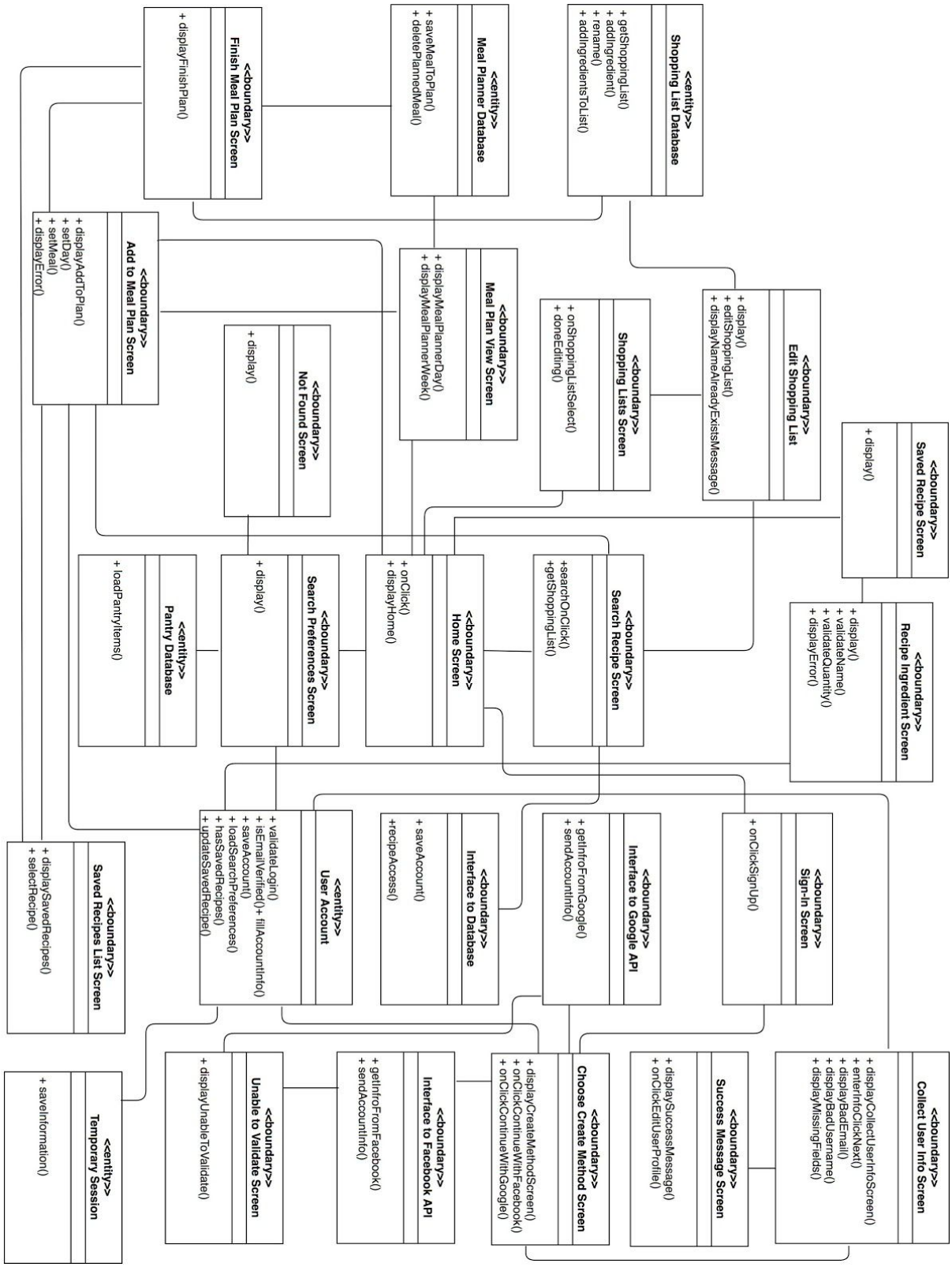


4.2 Domain Modeling

Figure 4.2.1 Domain Diagram



4.3 Class Diagram



5. Release

5.1 Test Cases

5.1.1 Editing Shopping List Test Case

Test Case #: 1.0	**Test Case Name:** Edit Shopping List Name Test Case	**Page:** 1 of 1
System: Feed.me	**Subsystem:** None	
Designed by: Petra Power, Lauren Hager	**Design Date:** March 12, 2019	
Executed by:	**Execution Date:**	

Short Description: Tests editing Shopping List properties

Pre-Conditions:

1. User has successfully created an account
2. User has logged into their account
3. Use has 2 existing lists called "One" and "Two"

Step	Action	Expected System Response	Pass/Fail	Comment
0	Use clicks Shopping Lists from Dropdown menu.	The system displays Shopping List screen.		
1	User clicks on Shopping List to edit, and clicks "Edit Selected List"	The system displays a new page with the full name of the shopping list and the list of ingredients.		
2	User selects to rename Shopping List by double clicking on the shopping list title.	System displays keyboard so user can type		
3	User types in "Shopping List 1" - not currently used by user	System accepts the new name and displays the shopping list with "Shopping List 1" as name		
4	User clicks the X	The system displays a list of all		

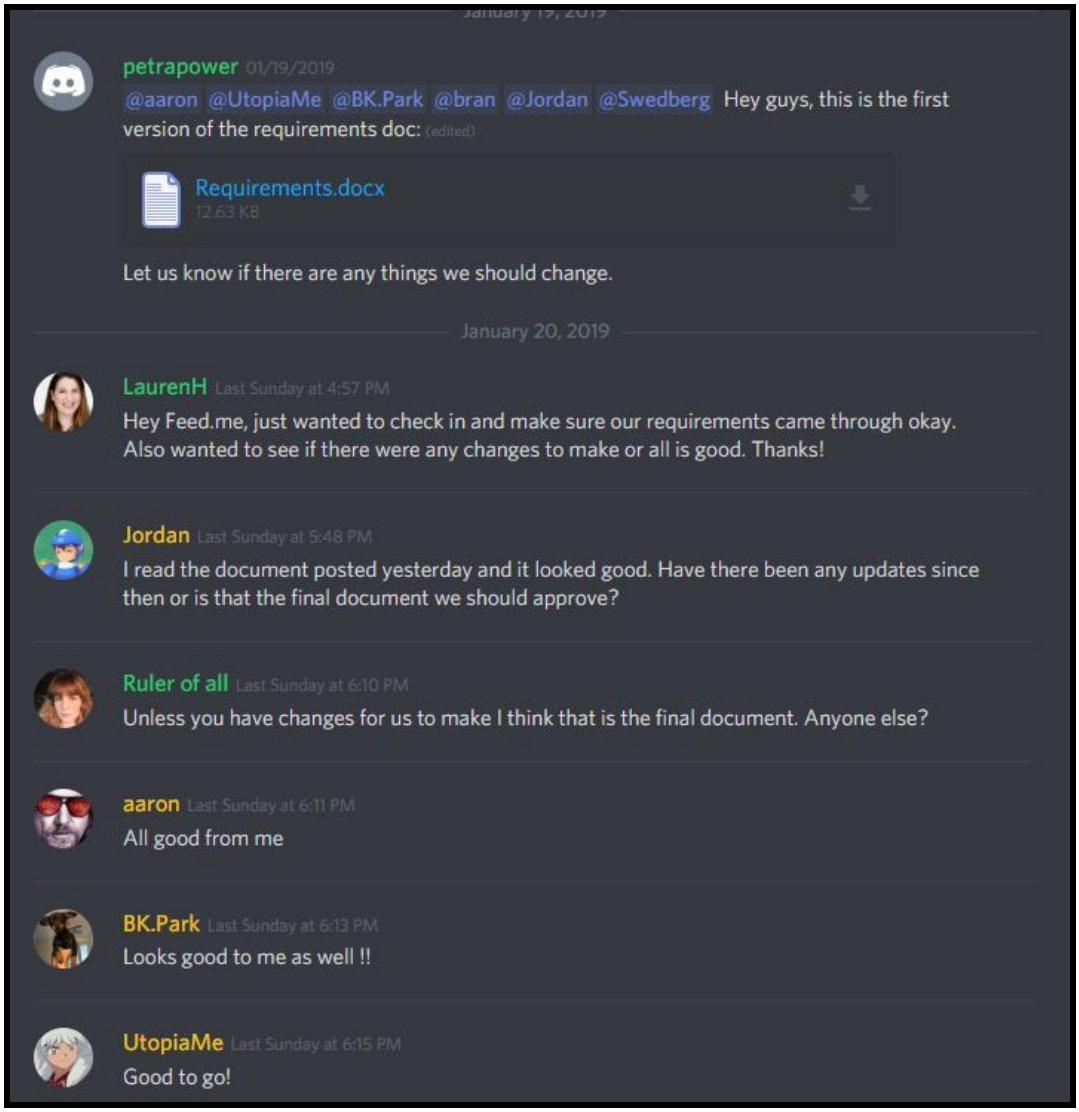
		the shopping lists		
5	Steps 1-2 but with a different shopping list	System displays keyboard so user can type		
6	User types in "Shopping List 1"	System displays a message and asks user to reenter name. The name does not change and displays as previous name		
7	User selects to rename Shopping List and enters "Shopping List 2" - not currently used by user	System accepts the new name and displays the shopping list with "Shopping List 2" as name		

Post-Conditions:

1. The user will have a Shopping List named "Shopping List 1"
2. The user will have a Shopping List named "Shopping List 2"

6. Appendix

6.1 Business Team SRS Sign-Off via Discord Communication



6.2 Stakeholder Peer Review by Team 3

The following is a summary of the observations made by the Team 3 business team on the proposal presented by Team 4. Much of the feedback here is on parts that need improvement because the parts that were done well need no comments or action to be taken.

6.2.1 Business Proposal

6.2.1.1 Budget

We found the estimated budget to be a sensible approximation of what the budget would turn out to be in implementation. However, there were several issues associated with different line item costs and cost projections.

First, app store fees are incurred in cash outflow before the product has been launched. Second, salary cost disappears after Q5, which suggests that there would be no employees working at the company after product launch.

Next, the product subscriber adoption rate does not seem realistic from out point of view. Ramping up to 500K subscribers in the first quarter, then having that number remain static over the next six quarters does not seem likely.

There was also an issue with the marketing budget. A general rule of thumb for market budgeting is 10% of estimated annual revenue. Using this metric, marketing costs should be approximately $1.2MM in the first year.

Finally, we would recommend that the model for monetization be augmented. We believe that subscriber numbers would be maximized if the application was offered as a free app with the core recipe functionality and ads, and a “pro” version of the app that is not only ad free, but offers all of the additional functionality outlined in the proposal.

6.2.1.2 Risks

The risk analysis was also very good with the exception of several elements. First, some risks have no description or elaboration. We realize that since this was an informal submission, most likely this has already been accounted for.

Secondly, competitiveness seems like a major risk, not a minor risk. Considering that there is already an app (MyFitnessPal) that dominates market-share and has serious corporate backing, any new functionality that proves profitable to Feed.me would most likely be emulated by MyFitnessPal, who already has a strong foothold in the marketplace.

6.2.2 Software Requirements Specifications

We found the software requirements to be well described and adequate to our needs. There were no issues as far as we could tell.

6.2.3 Wireframes

The wireframes are overall detailed enough to tell what each screen's purpose is, and most of the navigation between the screens is depicted. The high-level storyboard shows transitions between each screen clearly. Many of the wireframes are acceptable as-is, however there are some issues that should be addressed.

Some minor consistency issues are that the storyboard has two screens named "Create New User" and "Home Screen" which are referred to as "Create Profile" and "Main Menu" elsewhere. Another consistency issue is that "User Profile" and "Home Screen" are depicted in the storyboard but do not have individual wireframe descriptions.

The Search Preferences Wireframe has an "Update Pantry" button (WF-SP-04) which says in the description that it takes you to a pantry screen, but no such wireframe exists. Also, this navigation is not represented in the storyboard. Similarly, the Meal Planner wireframe has a "Search for Recipes" button (WF-MP-05) whose description says that it that takes user to a recipe search which has no related wireframe or representation in the storyboard. The drop down navigation menu in the Home Screen has entries for "Recipe Search" and "Pantries" which I suspect are these same screens, but they just haven't been designed yet.

The shopping list wireframe needs more elaboration. What happens when you select a shopping list? Does it go to another screen displaying the list's contents? Where can the contents of a list be edited? Also, this wireframe appears to be a dead end, meaning there is no way to navigate out of it without using the phone's back button.

6.2.4 Conclusion

The proposed design document is mostly acceptable with the condition that the issues identified in this review are addressed. Team 3 is happy to continue the relationship between our teams and we look forward to continued development of the Feed.me platform.

6.3 Developer Peer Review by Team 12

6.3.1 Outcome

Bid with Conditions

6.3.2 Summary Report

The overall report is well devised with some minor changes that we have suggested below.

The Software Requirements Specification has a nice, simple design with some added color which allows for easy navigation. In specific to the requirements, we found quite a bit of vagueness throughout the entirety of the SRS. Furthermore, we feel that it might be useful to have a glossary table to eliminate confusion on certain terms used in the SRS. Additionally, there are a few requirements that may need to be re-categorized as they do not seem to relate to their current categories.

The Storyboard is well designed and thought out. The flow of the application is clear and seems to be easy to use. While, the application seems to be easy to use, it seems to take the design of a website rather than a mobile application. Additionally, there are couple places where wording may need to be fixed for grammatical purposes, however, overall it is a great job.

The Architecture Design contains much information regarding your application and it is robust enough to show the specific details. Still, it seems that it is not complete and minor errors need to be fixed. There are parts, we found, which require clarification.

The Test Case is covering most of the functionalities that the system can offer. However, we mentioned some of the missing possible cases and the parts that require clarification in the test case.

Overall documentation seem thorough and just needs to be completed by the due date. Considering incomplete document, we as a developers will accept your project with clarifications and changes that we mention.

6.3.3 Software Requirements Specification

- General:
    - ○ Thorough, and fairly well-categorized.
    - ○ Glossary for terms. As developers, we aren't necessarily going to read the business proposal, so the terminology needs to be well-defined for us.
    - ○ Every requirement needs to be verifiable, i.e. testable.
    - ○ Order the SRS by priority [shall, should, may].
    - ○ Numbering system for requirement categories does not correspond to requirement identifiers, difficult to determine which category a requirement is.
    - ○ Empty page 17.
- Section 2.1.2

    - FR-RF-02.04: This is subjective, remove the word “favorite”.
    - FR-RF-03.01: “Pantry staples”. What is meant by this?
- Section 2.1.5
    - FR-US-02.02: “interface must be easy to use”, more like non-functional requirement, also not clear how to test this.
    - FR-US-02.03: not exactly functional, maybe phrase it like “shall provide English language options for text”.
- Section 2.2.1
    - NF-US-02.01: “simple enough”, “quickly”, “user errors”, how do you test these?
        - “The GUI shall facilitate users to complete any given task in under 5 seconds”.
        - “The GUI shall allow users to navigate to a target page with 3 or fewer clicks”.
    - NF-US-02.01: “easy to learn and navigate”, “simple to understand”.
- Section 2.2.3
    - NF-EX-01.01: What is input data? In what case would this happen, additionally, why would the input data change?
- Section 2.2.4
    - NF-PF-01.01: This may need to be better explained.
- Section 2.2.5
    - NF-AV-02.01: Mentions MTTR (Minimum Time To Restart?), what exactly does that stand for? Glossary would be key.
- Section 2.2.6
    - NF-SC-01.01: Wording and functionality unclear.
        - “not allow” - what does it mean to be allowed?
        - “unauthorized” - who is authorized?
        - “accidental” - how does someone accidentally access the app?
        - “unintended” - how does someone unintentionally access the app?
        - “legitimate users” - non-bots?
        - Maybe something like “the app shall detect when an account is being accessed by someone other than the account-holder.”
- Section 2.2.7, neither req. seems to correspond to portability (which I think is more for hardware)
    - NF-P-01.01: seems more like availability.
    - NF-P-02.01: seems more like reliability.
    - NF-P-02.01 should be split into two different requirements (currently there are two “shall”s for a single requirement).
- Section 2.3.1, requirements should be put in the proper categories even if they’re optional. “May” keyword sufficiently indicates that they are optional.
    - NF-FO-01.01: non-functional requirement designation not consistent with FR-US-02.03 functional requirement designation.

6.3.4 Storyboard

- 3.1 Mian app feature wireframe looks great and organized.
- 3.2.2 saved recipes storyboard looks great, but the arrows are hard to follow, maybe re-organize the diagram so arrows are easier to follow

- 3.4.1, The pantry items include in search, items is look very small to select, maybe make it look bigger
- WR 3.2 Saved Recipes Wireframe
    - “Accesses” is not the correct term to use. “Open” works better. Additionally, “accesses” is not being used correctly, grammatically speaking.
- 3.7.1 missing label for recipe picture between WF-MP 13 and 14.
- 3.7.2 AD Banner doesn’t seem necessary enough to have.
- Most of icon and text is a bit small

6.3.5 Architecture Design

- Use Case Diagram
    - Figure 4.1.0.1. should not have actors.
- Use Case Description
    - 4.1.1 Create New Account, may be separate Facebook and Google sign up? One as alternative.
        - Sequence diagram passes control to other use case, however it is not mentioned in the use case description.
- Activity Diagram
    - On 4.1.1.1 Activity Diagram, there should be only 1 arrow coming out of each action, yet the “Enter Password” action has 2 arrow direction coming out. Maybe needs a fork or decision block.
- Robustness Diagram
    - Figure 4.1.1.2. when saving information entered by user for creating an account, user require to verify their email account. During the temporary session, where are the information stored? Don’t you need entity?
    - On 4.1.1.2 Robustness Diagram, when bad email, missing fields, bad email, should the screen display the error and go back to collect user info screen rather than a new screen
    - Figure 4.1.2.2. shows “Recipe Ingredient Screen” boundary object 6 times
        - not sure this is valid, but good readability
    - Figure 4.1.2.2. does not show “LogIn” use case
    - Robustness diagram for the use case name “Edit Recipe Search Preference”
        - Require figure number
        - does not show “LogIn” use case
        - save change controller with ingredients list entity? not specified
        - update with pantry ingredients list entity? not specified
    - 4.1 4.2 the arrows in robust diagram are disconnected from the entities and objects, you should make sure the arrows are connected for syntax and better readability.
- Sequence Diagram
    - No sequence diagram provided for the use case “Edit Saved Recipes”
    - 4.1.6.2. Sequence Diagram, shouldn't it display shopping lists screen after edit shopping list screen, once the user finished their editing?
- Class Diagram
    - Not completed, cannot tell because of missing sequence diagram(s).

    - Otherwise, looks great!

6.3.6 Test Case

- What happens when we click ad banner? it is not included in the test case. Might not be relevant to the test case, just wondering.
- preconditions did not mention initial list, so assuming list 1 and 2 already exists without creating one? It would be nice to see initial list and if the test case is only for editing existing list or should user click “Add New Shopping List” before editing selected list.
- Might be better to have a test case for search for non-existing recipes.
- For the editing test case, Might be helpful to show the steps beforehand. For example, how user get to the shopping list edit.
- preconditions doesn’t mention that there is a shopping list created by the user.
- what if user doesn’t want the the name provided from the system once the user failed to provide the unused name at the first time.