Senior Capstone Project Design

Project Name

Household Menu Planner and Ingredient Tracker (HoMePIT)

Team Members

- Erik Anderson
- Jerome Chestnut
- Maxwell Fugette

Design Description

1. Block Diagram

The HoMePIT web application is built upon a well-structured system architecture that ensures efficient functionality and user interactions. HoMePIT comprises three essential sub-components: Local Storage, Network Connection, and the User Interface. These elements collectively enable the application to seamlessly manage user data, interact with external services, and present a user-friendly experience. The "Local Storage" component is responsible for handling data persistence on the user's device. It stores critical information, such as user preferences, ingredient lists, and recipes, locally within the browser. This enables the application to provide a smooth and responsive experience to the user. The "Network Connection" component plays a pivotal role in establishing communication between the application and external services. It ensures data can be synchronized with remote resources, such as Replit and Supabase. This connectivity is crucial for real-time updates, data sharing, and authentication processes.

Beyond the core HoMePIT components, the application interfaces with external services to enhance its capabilities. The "Browser" component acts as a bridge between HoMePIT and the Replit Server. It enables the application to send and receive data from the server, facilitating dynamic content updates and secure data storage. The "Replit Server" component serves as the intermediary between the application and the Supabase Database. It manages data requests, storage, and retrieval, ensuring that user-generated content is securely stored and readily available. The "Supabase Database" component, in turn, offers a robust and scalable data storage solution, housing critical information like user accounts, recipes, and ingredient details. To authenticate users securely, the application relies on the "Google OAuth 2.0" component, which is offered through Supabase. This authentication mechanism ensures that only authorized users can access and interact with the HoMePIT application.

2. Components

HoMePIT will consist of twelve main software components, consisting of the SignIn, Pantry, RecipeBook, MealPlanner, ShopingList, and Settings pages, along with the Ingredient, Recipe, Days, and Groceries classes. In addition, the root file (labeled “HoMePIT” in Figure 2.1) will serve as the router for all components in the project. The “index” component will serve to render the entire project through the router.

As stated, all main pages will be rendered through the HoMePIT router. Beginning with the SignIn page, this is the initial display which is presented to the user upon accessing the site.

After successful sign-in, the first page displayed to the user will be the Pantry. This component defines a list of ingredients from the Ingredient class, of which is implemented by the Pantry. The Pantry will contain a number of methods, such as addIngredient() and removeIngredient(), which will be used to add and remove Ingredient objects to the Pantry’s Ingredients list. The sortIngredients() method will sort the Pantry’s Ingredient list by taking in the desired metric (Requirements 3.1.6.1.2.2), as well as the desired sorting order (ascending or descending), with the outcome being a complete sorted version of the ingredients in the users Pantry. Additionally, when opted for, the displayByTags() method will ensure that the only ingredients displayed on the Pantry page are those specified by the user, categorized into the appropriate tags. All the above methods are visible in Figure 2.1.

Implemented by the Pantry, the Ingredient class will serve to define what constitutes an ingredient in the context of HoMePIT. Every Ingredient object will consist of a name, quantity, macro (calorie, protein, fat, carbohydrate) content, cost, and a myriad of other attributes. The Ingredient class will contain methods which allow the user to set these attributes to the desired values, such as setIngredientName() for the “name: string” attribute or setPurchaseCost() for the “cost: number” attribute. The Ingredient class will also handle ingredient substitutions in the form of a separate Ingredient list labeled “substitution”, which is associated with a single ingredient. The addSubstitution() and removeSubstitution() will serve to add and remove these, respectively. The Ingredient class is visible in Figure 2.2.

Following the Pantry, the next Page accessible to the user will be RecipeBook. Similar to Pantry, this component will handle the initialization of new Recipe objects which will exist as recipes in their own right. The methods addRecipe() and removeRecipe() will do what their names suggest, adding and removing Recipe objects to the RecipeBook. All recipes will be stored in a list of Recipe objects named “Recipes”. The method sortRecipes() will sort this list by taking in the desired metric and sorting order, just as in the sortIngredients() method from the Pantry component. It should be noted that some of the sorting methods applicable to the Ingredients list will not be so for the Recipes list, as is the case with the “Cook Time”, Time since last cooking”, “Leftover freshness time”, “Number of leftovers”, and “Time until leftover expiry” metrics (Requirements 4.1.5.1.2.2.6-10). The user, again, will have the option of displaying recipes by their tags via the displayByTags() method. All the above methods are visible in Figure 2.1.

Implemented by the RecipeBook, the Recipe class will serve to define what constitutes a recipe in the context of HoMePIT. Each Recipe object will be assigned a name, serving count, cook time, and more, along with an Ingredient list assigned to the specific recipe. It is from this Ingredient list that data such as the recipe’s total macro content and cost per serving will be calculated. Additionally, each Recipe object will include the attribute “subRecipe”, which will be defined by a Recipe list to serve as a sub-recipe to the given recipe. As a whole, this class will be

dependent on the Pantry due to its reliance on ingredients making up much of what constitutes a recipe. All attributes and methods of the Recipe class are depicted in Figure 2.2.

Following the RecipeBook, the MealPlanner component will be the next page accessible by the user. Here, the user will see all days of the current month displayed, with each month functioning as a list of Days objects (currentMonth: Days[]), representing the days of the month. The addMeal() and removeMeal() methods will handle the addition and removal of meals, as their name suggests. The defineWeeks() method will function to define the weeks in a given month, taking in a list of Days called "days" for future use in calculating nutrition and cost data. The attributes and methods of MealPlanner are visible in Figure 2.1.

Implemented by the MealPlanner, the Days class will serve to define what constitutes a given day in the month. Each individual day will consist of slots for meals (breakfast, lunch, dinner, etc) in which the user will be able to add recipes as well as meal times. In addition, each day will be identified by its own name via the dayOfWeek attribute, which will be one of the seven days of the week. This class will contain several methods relating to meal planning, such as addRecipeToMeal() and removeRecipeFromMeal, methods which add and remove recipes to a given meal, respectively. Each method takes in a Recipe object and a Recipe list, labeled as "meal". This ensures that the given recipe is added or removed from the intended meal. Notably, this class also includes the useLeftovers() and cookFromScratch() methods, which will determine whether a meal will consist of leftovers or not. Both methods return a boolean value. Lastly, the Days class contains multiple nutrition and cost calculation methods to determine the total nutrient and financial outcomes of a given day, week, or month, as well as a set of methods for doing the same within a custom date range. The user will provide the appropriate start and end dates which will serve as the parameters for these methods. The attributes and methods of the Days class are visible in Figure 2.2.

Following the MealPlanner, the ShoppingList component will be the next page accessible by the user, as well as the last major feature of HoMePIT. The primary purpose of this component will be to list the users various shopping lists in the form of a list of Grocery objects (shoppingLists: Groceries[]). The option will exist to add and remove new shopping lists via the addShoppingList() and removeShoppingList() methods, which are the only two methods contained within this component. The methods and attributes for ShoppingList are depicted in Figure 2.1. The major features associated with this component will be expressed in the associated Groceries class.

As an implementation of the ShoppingList component, the Groceries class will handle all functionality related to generating new shopping lists. The main attributes will consist of a list of ingredients to buy (ingredientsToBuy: Ingredient[]), which is a list of Ingredient objects. This attribute will be initialized by the generateFromPantry() method, which takes the Pantry's Ingredients list as a parameter and returns a new list of ingredients that must be purchased, which will be accomplished by the method observing which ingredients are at risk of going under (or are under) their set threshold quantity. Additionally, the user will have the option of manually adding or removing ingredients from a shopping list via the addToList() and removeFromList() methods, each taking in an Ingredient object. As stated in Requirement 6.2.1.2, there will exist the option to generate a shopping list to satisfy a given date range. This feature will be accomplished through the setDateRange() method, which takes a start and end date as parameters and returns a list of Ingredient objects to satisfy said date range. Lastly, the Groceries class will include the setItemBeingPurchased() method, which as the name suggests sets an item the user

has purchased and stores its data in the Pantry. This class, its attributes and relations are depicted in Figure 2.2.

The final page accessible to the user will be the Settings page. This component serves a fairly simple purpose, that being the toggling of regular shopping trips via the toggleRegularShoppingTrip() method, as well as setting the frequency of grocery trips via the setGroceryTripFrequency() method. The former returns a boolean value that is stored in the regularShoppingTrip attribute, and the latter returns void. In addition to these two methods, the Settings component includes the setPreferredShoppingDay() method, which takes in the desired day as a parameter and returns void. The Settings component, its relations, methods, and attributes are depicted in Figure 2.1.

3. UI and UX

HoMePIT's UI design is produced with the intention of minimizing the complexity of its usage, particularly with respect to reducing nested menus. It is designed with primarily mobile device users in mind, but is intended to be compatible with desktop users as well. Deriving this design from earlier requirements revealed that said requirements had issues with respect to having customized variants of existing pages accessed via other pages' menus. However, with the completion of the UI designing process, these requirements have been addressed and modified.

The first UI element that a user will interact with upon opening HoMePIT will be the sign-in page. This is depicted in Figure 3.1, wherein only the HoMePIT logo and a Google Authentication-based sign-in will be available. Upon clicking the prompt for signing in via Google, the user will be presented with a Google sign-in window, wherein the user may sign in with their Google account. If their sign-in fails, or if they exit the Google sign-in window without signing in, they will be informed that their sign-in failed, as depicted in Figure 3.1. If their sign-in succeeds, the user will be presented with the Pantry page, as depicted in Figure 3.4, with a welcome message already overlain, as depicted in Figure 3.2. After the user clicks the message's close button or clicks outside of the message window, the message will disappear, and the user will be able to start using HoMePIT.

When presented with the Pantry page, as depicted in the top left screen of Figure 3.4, the user will see several things. At the bottom of the page is the navigation bar, which is shown in an abbreviated form with just its name. The actual appearance of the navigation bar can be seen in Figure 3.3, wherein its five buttons can be observed. Each button takes the user to a different page of the website, with the current page being viewed being visually indicated. The navigation bar is visible on each page, although open modal windows keep the user from interacting with the navigation bar until they've been closed.

Aside from the navigation bar, the user will see a list of ingredients, with new users' lists being empty, as well as the 'search' and 'add ingredient' buttons. If the user is in the process of adding an ingredient to a recipe, setting it as a substitute to another ingredient, or adding an ingredient to a shopping list, the user will respectively see buttons to 'add to recipe', 'add as substitute', and 'add to shopping list'.

When the user clicks the 'add ingredient' button, they will be shown the 'add ingredient' modal, as depicted in the top right screen of Figure 3.4. In this modal, the user will see input fields and dropdown menus that allow them to describe an ingredient with as much detail as they desire. One of these input fields corresponds to ingredient substitutes. If the user clicks to add a substitute ingredient, the modal window will be minimized (the action of which is described next) and the user will be able to select ingredients to add as substitutes, which will be added as substitutes once they click the 'add as substitute' button previously described.. Beyond these methods of adding information, the user will also see three other buttons. The first button is the 'minimize' button, indicated by a '-' symbol in a circle at the top right of the modal. When clicked, this button will minimize the 'add ingredient' modal without discarding already input information. This will also occur if the user clicks outside of the modal window's boundaries, with the lack of information discarding being intended to prevent the user from accidentally undoing their information-entry work with an errant click. The second button is the 'add' button, which adds the described ingredient to the ingredient list, discards any input information, and closes the modal. The third button is the 'close' button, which closes the modal and discards any input information.

After adding an ingredient to the Pantry, the user will see it in their ingredient list, with it being represented in a segment with a check box, the ingredient's name, and a button to expand the ingredient with information. If the user clicks on the check box, the ingredient will be selected, which is used for selecting which ingredients to add to a recipe or as substitutes for other ingredients. If the user clicks on the ingredient's name or quantity, the 'add ingredient' modal will be shown again, but with information from the selected ingredient filled in. If the user clicks the 'expand' button, represented by an upside-down triangle, the user will see that ingredient's quantity, its expiry date if it exists, and a button to delete the ingredient from the list. If the user clicks the 'delete' button, the ingredient and its information will be deleted from the user's Pantry.

When the user clicks the 'search' button, they will be shown the modal depicted in the bottom left screen of Figure 3.4 with input fields and dropdown menus to specify which ingredients they want to view, alongside buttons to apply, reset, or close the search modal. If the user clicks the 'apply' button, any input search parameters will be applied, the modal will close, and the user will see their ingredients that satisfy the search parameters. If the user clicks the 'reset' button, any changed search parameters will be set to their default. If the user clicks the 'close' button, any changed search parameters will be discarded, the modal will be closed, and they will be presented with the ingredient list that they had previously been viewing.

If the user clicks the 'Recipe Book' button on the navigation bar, they will be taken to the Recipe Book page. The Recipe Book page is nearly a copy of the Pantry page, with the exception that any references to ingredients and their information is replaced with references to recipes and their information. This can be seen in Figure 3.5. The main difference in behavior between the Pantry page and the Recipe Book page would be the fact that when selecting recipes, the user will see buttons that allow them to 'add to recipe' and 'add to meal', which will be respectively present when a recipe creation modal or meal planner modal are active.

If the user clicks the 'Meal Planner' button on the navigation bar, they will be taken to the Meal Planner page and presented with a calendar of their planned meals, focused on the closest six weeks, which will be initially empty for a new user. This can be seen in the leftmost screen depicted in Figure 3.6. They will be able to scroll through the calendar to view other weeks, and thus months. If the user clicks on a day, they will be presented with a modal containing a list of meals, as seen in the central screen in Figure 3.6, with each meal having the ability to be assigned a time, recipes to be consumed for that meal, and how many servings the meal will require. If the user wants to add a recipe to one or more meals in a selected day, they simply need to select those meals and minimize the meal planner modal, at which point they may navigate to the Recipe Book and search for recipes to add. Added recipes will show their name and have the option to have a panel with additional information definition drop down. When this panel is deployed, the user may choose to either cook the recipe from scratch or use leftovers. If the user chooses to cook the recipe from scratch, they'll be shown when they need to start cooking the recipe based on their previously specified meal time. They can also input how many servings to prepare and see how many leftovers are predicted to be made, with these fields being filled by default with default information from the recipe, scaled to ensure that enough servings are made to satisfy the number of servings required for the meal. Finally, the user can select substitutions to use in place of given ingredients.

If the user long-clicks a day, clicks a week or month, or clicks the 'custom overview' button, they will be presented with a modal summarizing a variety of mainly-nutritional information for the selected day, week, month, or custom span of time. This can be seen in the

right-most screen in Figure 3.6. As noted in the requirements document, any information that cannot be calculated or is missing information simply will not be shown to the user. The span of time viewed can be modified by adjusting the dates listed as starting and ending dates in the modal.

If the user clicks the ‘Shopping List’ button on the navigation bar and they have neither visited the Shopping List page nor input settings into the Settings page, they will be presented with the leftmost screen depicted in Figure 3.7. In such a case, the user will be implicitly encouraged to enter settings for how often they shop for groceries and on what day they prefer to grocery shop. After they input these settings or opt out of inputting them, they will be presented with the middle screen shown in Figure 3.7. This will likewise occur if they had already input settings or visited the page previously. The user will be presented with a list of shopping lists that satisfy certain date ranges of meals, with the shopping lists being composed of a list of ingredients with the quantity needed to be purchased and cost of that quantity listed. The user has the option to manually add ingredients to shopping lists, which is done via the same action of selecting ingredients on the Pantry page and clicking a button to add them to the selected shopping list. Shopping lists can be confirmed, allowing users to inform HoMePIT that they have obtained any listed quantities of ingredients. A user may also alter the date range covered by a shopping list by interacting with the listed date range that a shopping list satisfies. They may also click the ‘add shopping list button’, indicated with a ‘+’, to manually add a new shopping list satisfying a defined time period via the ‘add shopping list’ modal depicted in the right screen of Figure 3.7.

If the user clicks the ‘Setting’ button on the navigation bar, they will be presented with the settings page shown in Figure 3.8. On this page, the user may alter the settings that they may have chosen to define when first viewing the Shopping List page. They may also choose to sign out of HoMePIT, in which case they will be returned to the Sign In page depicted in Figure 3.1.

4. Database

The database for HoMePIT is fairly extensive as a result of the breadth of data being handled and the extent to which that information is interconnected. This can be seen in Figure 4.1 and Figure 4.2 in the Appendix. The database is designed to adhere to Third Normal Form (3NF), such that data replication and anomalies are minimized while ensuring data integrity. The most critical tables of the database would be Ingredients, Recipes, Meals, and Shopping Lists. These ‘critical’ tables contain information relating to the topics that they’re titled after, such as ingredient or recipe names and nutritional information for each serving of that item. The other tables of the database largely exist to either serve as a junction between these tables’ data or to provide further capability to these tables that would break 3NF if implemented within those tables.

Revised Requirements

1. Sign-in and Sign-out
    1.1. When first opening the website, a user will be presented with a prompt to sign into the website by connecting their Google account.
        1.1.1. On click, the user will be taken to the Google account authentication page
        1.1.2. After signing in, users will be presented with the Pantry page and Navigation Bar
2. Navigation Bar
    2.1. There will be a bar displayed with prompts for changing the currently viewed page
        2.1.1. Clicking a prompt will change the currently viewed page
            2.1.1.1. There will be a prompt for the Pantry page
            2.1.1.2. There will be a prompt for the Recipe Book page
            2.1.1.3. There will be a prompt for the Meal Planner page
            2.1.1.4. There will be a prompt for the Shopping List page
            2.1.1.5. There will be a prompt for the Settings page
        2.1.2. The current page will be visually indicated
        2.1.3. A page that has updates or notifications will be visually indicated
3. Pantry
    3.1. This page will contain a listing of ingredients determined by the user
        3.1.1. The list will initially be empty
        3.1.2. There will be a button to add a new ingredient
            3.1.2.1. On click, a modal for defining an ingredient will appear, with:
                3.1.2.1.1. A label for ingredient name
                    3.1.2.1.1.1. This will be marked as required with an *
                3.1.2.1.2. An input box connected to the ingredient name label
                    3.1.2.1.2.1. There will be no placeholder value
                    3.1.2.1.2.2. It will accept characters, numbers, and punctuation
                3.1.2.1.3. A label for ingredient quantity
                3.1.2.1.4. An input box connected to the ingredient quantity label
                    3.1.2.1.4.1. There will be no placeholder value
                    3.1.2.1.4.2. It will accept numbers
                3.1.2.1.5. A dropdown menu connected to the ingredient quantity input box
                    3.1.2.1.5.1. There will be no placeholder value
                    3.1.2.1.5.2. It will contain imperial masses
                        3.1.2.1.5.2.1. oz, lb
                    3.1.2.1.5.3. It will contain imperial volumes
                        3.1.2.1.5.3.1. gal, qt, pt, cup, ½ cup, ⅓ cup, ¼ cup, fl oz, tbsp, tsp, ½ tsp, ¼ tsp, ⅛ tsp
                    3.1.2.1.5.4. It will contain metric mass
                        3.1.2.1.5.4.1. kg, g
                    3.1.2.1.5.5. It will contain metric volumes

3.1.2.1.5.5.1. dl, cl, ml
3.1.2.1.6. A label for ingredient threshold quantity
3.1.2.1.7. An input box connected to the ingredient threshold quantity label
3.1.2.1.7.1. There will be no placeholder value
3.1.2.1.7.2. It will accept numbers
3.1.2.1.8. A dropdown menu connected to the ingredient threshold quantity input box
3.1.2.1.8.1. There will be no placeholder value
3.1.2.1.8.2. It will contain imperial masses
3.1.2.1.8.2.1. oz, lb
3.1.2.1.8.3. It will contain imperial volumes
3.1.2.1.8.3.1. gal, qt, pt, cup, ½ cup, ⅓ cup, ¼ cup, fl oz, tbsp, tsp, ½ tsp, ¼ tsp, ⅛ tsp
3.1.2.1.8.4. It will contain metric mass
3.1.2.1.8.4.1. kg, g
3.1.2.1.8.5. It will contain metric volumes
3.1.2.1.8.5.1. dl, cl, ml
3.1.2.1.9. A label for expiration date
3.1.2.1.10. An input box connected to the expiration date label
3.1.2.1.10.1. There will be no placeholder value
3.1.2.1.10.2. It will accept numbers, in the format of DD-MM-YYYY and DD-MM
3.1.2.1.11. A label for serving size
3.1.2.1.12. An input box connected to the serving size label
3.1.2.1.12.1. There will be no placeholder value
3.1.2.1.12.2. It will accept numbers
3.1.2.1.13. A dropdown menu connected to the serving size input box
3.1.2.1.13.1. There will be no placeholder value
3.1.2.1.13.2. It will contain imperial masses
3.1.2.1.13.2.1. oz, lb
3.1.2.1.13.3. It will contain imperial volumes
3.1.2.1.13.3.1. gal, qt, pt, cup, ½ cup, ⅓ cup, ¼ cup, fl oz, tbsp, tsp, ½ tsp, ¼ tsp, ⅛ tsp
3.1.2.1.13.4. It will contain metric mass
3.1.2.1.13.4.1. kg, g
3.1.2.1.13.5. It will contain metric volumes
3.1.2.1.13.5.1. dl, cl, ml
3.1.2.1.14. A label for calories per serving
3.1.2.1.15. An input box connected to the calories per serving label
3.1.2.1.15.1. There will be no placeholder value
3.1.2.1.15.2. It will accept numbers
3.1.2.1.16. A label for protein per serving
3.1.2.1.17. An input box connected to the protein per serving label
3.1.2.1.17.1. There will be no placeholder value
3.1.2.1.17.2. It will accept numbers

3.1.2.1.18. A label for fat per serving
3.1.2.1.19. An input box connected to the fat per serving label
3.1.2.1.19.1. There will be no placeholder value
3.1.2.1.19.2. It will accept numbers
3.1.2.1.20. A label for carbs per serving
3.1.2.1.21. An input box connected to the carbs per serving label
3.1.2.1.21.1. There will be no placeholder value
3.1.2.1.21.2. It will accept numbers
3.1.2.1.22. A label for purchase serving count
3.1.2.1.23. An input box connected to the purchase serving count label
3.1.2.1.23.1. There will be no placeholder value
3.1.2.1.23.2. It will accept numbers
3.1.2.1.24. A label for purchase cost
3.1.2.1.25. An input box connected to the purchase cost label
3.1.2.1.25.1. There will be no placeholder value
3.1.2.1.25.2. It will accept numbers
3.1.2.1.26. A label for purchase location
3.1.2.1.27. An input box connected to the purchase location label
3.1.2.1.27.1. There will be no placeholder value
3.1.2.1.27.2. It will accept characters, numbers, and punctuation
3.1.2.1.27.3. Multiple locations are separated by commas
3.1.2.1.28. A label for user-defined tags
3.1.2.1.29. An input box connected to the user-defined tags label
3.1.2.1.29.1. There will be no placeholder value
3.1.2.1.29.2. It will accept characters, numbers, and punctuation
3.1.2.1.29.3. Multiple tags are separated by commas
3.1.2.1.30. A label for substitutions
3.1.2.1.31. An input box connected to the substitutions label
3.1.2.1.31.1. On click, the modal will minimize
3.1.2.1.31.1.1. If ingredients are selected and ‘add to substitution’ (3.2) is pressed, the ingredients will be added to the substitution list.
3.1.2.1.31.1.1.1. When ingredients are added to the substitution list for one ingredient, that ingredient will be automatically added to the substitution list of its substitutions

3.1.3. Each ingredient on the list will display their name, with quantity and expiration date being visible in a dropdown panel accessed with a button click
3.1.3.1. The ingredient name can be clicked
3.1.3.1.1. On click, the modal described by 3.1.2.1 will appear
3.1.3.1.1.1. Existing data will show as placeholder data
3.1.3.2. If the ingredient quantity is below the ingredient threshold quantity, there will be a visual indication nearby the shown quantity.

3.1.3.3. Each ingredient on the list will have a “remove” button in the dropdown panel
3.1.3.3.1. On click, the given item will be removed from the pantry
3.1.4. Each ingredient on the list will be paired with a toggle box
3.1.4.1. This toggle box will have a placeholder value of unselected
3.1.4.2. When toggled, an ingredient will be selected
3.1.5. The list will be sorted alphabetically, ascending by default
3.1.6. There will be a button to change the list’s sorting
3.1.6.1. On click, a modal for changing sorting will appear, with:
3.1.6.1.1. A label for metric to sort by
3.1.6.1.2. A dropdown list connected to the sorting metric label
3.1.6.1.2.1. The dropdown list will have a placeholder value of ‘alphabetical’
3.1.6.1.2.1.1. If another option was selected earlier, that option will be the placeholder
3.1.6.1.2.2. The list will contain the following options:
3.1.6.1.2.2.1. Alphabetical
3.1.6.1.2.2.2. Quantity
3.1.6.1.2.2.3. Expiration Date
3.1.6.1.2.2.4. Protein per serving
3.1.6.1.2.2.5. Fat per serving
3.1.6.1.2.2.6. Carbohydrates per serving
3.1.6.1.3. A label for sorting order
3.1.6.1.4. A dropdown list connected to the sorting order label
3.1.6.1.4.1. The dropdown list will have a placeholder value of ‘ascending’
3.1.6.1.4.1.1. If another option was selected earlier, that option will be the placeholder
3.1.6.1.4.2. The list will contain the following options:
3.1.6.1.4.2.1. Ascending
3.1.6.1.4.2.2. Descending
3.1.6.1.5. A label for user tags to be included
3.1.6.1.6. An input box connected to the user-defined tags label
3.1.6.1.6.1. There will be no placeholder value
3.1.6.1.6.2. It will accept characters, numbers, and punctuation
3.1.6.1.6.3. Multiple tags are separated by commas
3.1.6.1.7. A label for user tags to exclude
3.1.6.1.8. An input box connected to the user-defined tags label
3.1.6.1.8.1. There will be no placeholder value
3.1.6.1.8.2. It will accept characters, numbers, and punctuation
3.1.6.1.8.3. Multiple tags are separated by commas
3.1.6.1.9. There will be a button to reset the list’s sorting
3.1.6.1.9.1. On click, the sorting will be reset to alphabetical, ascending
3.2. When the ingredient creation modal is minimized, a button labeled ‘add to substitution’ will be shown.

3.2.1. On click, any selected ingredients will be added to the substitution list of the current ingredient creation modal.

3.3. When a recipe creation modal is maximized or minimized in the Recipe Book, a button labeled ‘add to recipe’ will be shown.

3.3.1. On click, any selected ingredients will be added to the ingredient list of the current recipe creation modal.

3.4. When a shopping list is selected in the Shopping List, a button labeled ‘add to shopping list’ will be shown.

3.4.1. On click, any selected ingredients will be added to the selected shopping list’s ingredient list.

4. Recipe Book

4.1. This page will contain a listing of recipes input by the user

4.1.1. The list will initially be empty

4.1.2. There will be a button to add a new recipe

4.1.2.1. On click, a modal for defining a recipe will appear, with:

4.1.2.1.1. A label for recipe name

4.1.2.1.1.1. This will be marked as required with an *

4.1.2.1.2. An input box connected to the recipe name label

4.1.2.1.2.1. There will be no placeholder value

4.1.2.1.2.2. It will accept numbers, characters, and punctuation

4.1.2.1.3. A label for serving count

4.1.2.1.4. An input box connected to the serving count label

4.1.2.1.4.1. There will be no placeholder value

4.1.2.1.4.2. It will accept leftover serving count

4.1.2.1.5. A label for recipe cook time

4.1.2.1.6. An input box connected to the cook time label

4.1.2.1.6.1. There will be no placeholder value

4.1.2.1.6.2. It will accept numbers and punctuation

4.1.2.1.7. A label for total recipe cook time

4.1.2.1.8. An input box connected to the total recipe cook time label

4.1.2.1.8.1. There will be no placeholder value initially

4.1.2.1.8.2. After ingredients and sub-recipes are selected, a placeholder value will appear that is the sum of the input cook time (4.1.2.1.7) and all sub-recipe cook times

4.1.2.1.8.2.1. If a time has already been entered, it will not be replaced by this automatically calculated time

4.1.2.1.8.3. It will accept numbers and punctuation in the format of HH:MM, 24-hour time

4.1.2.1.8.4. If an invalid time is input, there will be a visual indicator indicating such

4.1.2.1.9. A label for calories, protein, fat, and carbohydrates per serving

4.1.2.1.10. A label connected to the calories, protein, fat, and carbohydrates label that displays said information

4.1.2.1.10.1. Before ingredients and sub-recipes are selected, there will be no placeholder value
4.1.2.1.10.2. After ingredients and sub-recipes are selected, the information to be displayed will be automatically calculated from selected ingredients and sub-recipes
4.1.2.1.10.3. If one or more ingredients is lacking nutritional information, a visual indicator will be displayed to indicate this
4.1.2.1.11. A label for cost per serving
4.1.2.1.12. A label connected to the cost per serving label
4.1.2.1.12.1. Before ingredients and sub-recipes are selected, there will be no placeholder value
4.1.2.1.12.2. After ingredients and sub-recipes are selected, the information to be displayed will be automatically calculated from selected ingredients and sub-recipes
4.1.2.1.12.3. If one or more ingredients is lacking nutritional information, a visual indicator will be displayed to indicate this
4.1.2.1.13. A label for predicted leftover expiration time
4.1.2.1.14. An input box connected to the predicted leftover expiration time label
4.1.2.1.14.1. There will be no placeholder value
4.1.2.1.14.2. It will accept numbers
4.1.2.1.15. A label connected to the predicted leftover expiration time input box displaying ‘days’
4.1.2.1.16. A label for user tags
4.1.2.1.17. An input box connected to the user-defined tags label
4.1.2.1.17.1. The placeholder value will be a list of the unique user-defined tags assigned to its ingredients and sub-recipes
4.1.2.1.17.2. It will accept characters, numbers, and punctuation
4.1.2.1.17.3. Multiple tags are separated by commas
4.1.2.1.18. A label for ingredients and sub-recipes
4.1.2.1.18.1. This will be marked as required with an *
4.1.2.1.19. A list connected to the ingredients label
4.1.2.1.19.1. The list will initially be empty
4.1.2.1.19.2. On click, the modal will minimize
4.1.2.1.19.2.1. If ingredients are selected and ‘add to recipe’ (3.3) is pressed, the ingredients will be added to the ingredient list.
4.1.2.1.19.2.2. If recipes are selected and ‘add to recipe’ (4.2) is pressed, the recipes will be added to the ingredient list
4.1.2.1.19.3. Each ingredient or sub-recipe added to the list will have:
4.1.2.1.19.3.1. A label for ingredient or sub-recipe quantity

4.1.2.1.19.3.2. An input box connected to the ingredient or sub-recipe quantity label
4.1.2.1.19.3.2.1. There will be no placeholder value
4.1.2.1.19.3.2.2. It will accept numbers
4.1.2.1.19.3.3. A dropdown menu connected to the ingredient or sub-recipe quantity input box
4.1.2.1.19.3.3.1. There will be no placeholder value
4.1.2.1.19.3.3.2. It will contain imperial masses
4.1.2.1.19.3.3.2.1. oz, lb
4.1.2.1.19.3.3.3. It will contain imperial volumes
4.1.2.1.19.3.3.3.1. gal, qt, pt, cup, ½ cup, ⅓ cup, ¼ cup, fl oz, tbsp, tsp, ½ tsp, ¼ tsp, ⅛ tsp
4.1.2.1.19.3.3.4. It will contain metric mass
4.1.2.1.19.3.3.4.1. kg, g
4.1.2.1.19.3.3.5. It will contain metric volumes
4.1.2.1.19.3.3.5.1. dl, cl, ml

4.1.3. Each recipe on the list will display their name, as well as cook time, calories per serving, grams of protein per serving, grams of fat per serving, grams of carbohydrates per serving, number of leftovers, time until leftover expiry, time since last creation, and a “remove button” in a dropdown panel opened with a button
4.1.3.1. Name
4.1.3.1.1. The recipe name can be clicked
4.1.3.1.1.1. On click, the recipe creation modal (4.1.2.1) will appear
4.1.3.1.1.1.1. Existing data will show as placeholder data
4.1.3.2. On click of the “remove” button, the given item will be removed from the recipe book

4.1.4. The list will be sorted alphabetically by default

4.1.5. There will be a button to search
4.1.5.1. On click, a modal for searching will appear, with:
4.1.5.1.1. A label for metric to sort by
4.1.5.1.2. A dropdown list connected to the sorting metric label
4.1.5.1.2.1. The dropdown list will have a placeholder value of ‘alphabetical’
4.1.5.1.2.1.1. If another option was selected earlier, that option will be the placeholder
4.1.5.1.2.2. The list will contain the following options:
4.1.5.1.2.2.1. Alphabetical
4.1.5.1.2.2.2. Calories per serving
4.1.5.1.2.2.3. Protein per serving
4.1.5.1.2.2.4. Fat per serving
4.1.5.1.2.2.5. Carbohydrates per serving
4.1.5.1.2.2.6. Cook time
4.1.5.1.2.2.7. Time since last cooking

4.1.5.1.2.2.8. Leftover freshness time
4.1.5.1.2.2.9. Number of leftovers
4.1.5.1.2.2.10. Time until leftover expiry
4.1.5.1.2.3.
4.1.5.1.3. A label for sorting order
4.1.5.1.4. A dropdown list connected to the sorting order label
4.1.5.1.4.1. The dropdown list will have a placeholder value of 'Ascending'
4.1.5.1.4.1.1. If another option was selected earlier, that option will be the placeholder
4.1.5.1.4.2. The list will contain the following options:
4.1.5.1.4.2.1. Ascending
4.1.5.1.4.2.2. Descending
4.1.5.1.5. A label for user tags to include
4.1.5.1.6. An input box connected to the user-defined tags label
4.1.5.1.6.1. There will be no placeholder value
4.1.5.1.6.2. It will accept characters, numbers, and punctuation
4.1.5.1.6.3. Multiple tags are separated by commas
4.1.5.1.7. A label for user tags to exclude
4.1.5.1.8. An input box connected to the user-defined tags label
4.1.5.1.8.1. There will be no placeholder value
4.1.5.1.8.2. It will accept characters, numbers, and punctuation
4.1.5.1.8.3. Multiple tags are separated by commas
4.1.5.1.9. A label for ingredients to include
4.1.5.1.10. An input box connected to the ingredients label
4.1.5.1.10.1. There will be no placeholder value
4.1.5.1.10.2. It will accept characters, numbers, and punctuation
4.1.5.1.10.3. Multiple ingredients are separated by commas
4.1.5.1.11. A label for ingredients to exclude
4.1.5.1.12. An input box connected to the ingredients label
4.1.5.1.12.1. There will be no placeholder value
4.1.5.1.12.2. It will accept characters, numbers, and punctuation
4.1.5.1.12.3. Multiple ingredients are separated by commas
4.1.5.1.13. A button to reset the list's sorting
4.1.5.1.13.1. On click, the sorting will be reset to alphabetical, low-too-high

4.2. When a recipe creation modal is minimized, a button labeled 'add to recipe' will be shown.
4.2.1. On click, any selected recipes will be added to the ingredient list of the current recipe creation modal.

4.3. When a meal creation modal is maximized or minimized in the Meal Planner, a button labeled 'add to meal' will be shown.
4.3.1. On click, any selected recipes will be added to the recipe list of the current meal creation modal.
4.3.2. When selecting recipes, recipes that have available ingredients will be visually indicated by a color. Recipes that have available ingredients

within a week of expiry will be visually indicated by a different color. Recipes that lack one or more ingredients but have substitutions will be indicated by a different color. Recipes that lack one or more ingredients without available substitutions will be visually indicated by another color.

5. Meal Planner

5.1. This page will contain a calendar

5.1.1. The calendar will present all days of the current month

5.1.2. Each day of the month will be clickable

5.1.2.1. On click, a modal will appear with:

5.1.2.1.1. A list

5.1.2.1.1.1. The list will initially be empty

5.1.2.1.1.2. The list will be divided into several sections, each with:

5.1.2.1.1.2.1. A label for either breakfast, brunch, lunch, dinner, dessert, or snack

5.1.2.1.1.2.2. A label for meal time

5.1.2.1.1.2.3. An input box connected to each meal time label

5.1.2.1.1.2.3.1. There will be no placeholder value

5.1.2.1.1.2.3.2. It will accept numbers and punctuation in the format of HH:MM, 24-hour time

5.1.2.1.1.2.3.3. If an invalid time is input, there will be a visual indicator indicating such

5.1.2.1.1.2.4. A label for number of servings required

5.1.2.1.1.2.5. An input box connected to each required servings label

5.1.2.1.1.2.5.1. There will be no placeholder value

5.1.2.1.1.2.5.2. It will accept numbers

5.1.2.1.1.2.6. A label for starting cook time

5.1.2.1.1.2.7. A label connected to each section's starting cook time label

5.1.2.1.1.2.7.1. There will be no placeholder value

5.1.2.1.1.2.7.2. After a section's meal time is entered (5.1.2.1.1.2.3), the label will present the latest time that meal preparation can start based on the input meal time and the longest of the selected non-leftover-using recipes' cook times, taking into account whether sub-recipes are available or need to be prepared

5.1.2.1.1.2.8. Each recipe in each section will have:

5.1.2.1.1.2.8.1. A label for recipe name

5.1.2.1.1.2.8.2. A button to remove a recipe

5.1.2.1.1.2.8.2.1. On click, the recipe will be removed from the section

5.1.2.1.1.2.8.3. A radio button with 2 options

5.1.2.1.1.2.8.4. A label connected to the other radio button for for ‘use leftovers’

5.1.2.1.1.2.8.5. A label connected to one radio button for ‘cook from scratch’

5.1.2.1.1.2.8.6. A label connected to the ‘cook from scratch’ label for ‘number of servings to cook’

5.1.2.1.1.2.8.7. An input box connected to each ‘number of servings to cook’ label

5.1.2.1.1.2.8.7.1. The placeholder value will be the number of servings made by an instance of the recipe, as input by the user in the Recipe Book

5.1.2.1.1.2.8.7.2. It will accept numbers

5.1.2.1.1.2.8.7.3. If a user inputs a value below or above the default number of servings for a recipe, there will be a separate visual indicator of each case

5.1.2.1.1.2.8.7.4. If the radio button is toggled to ‘use leftovers’, this input box will become noninteractive

5.1.2.1.1.2.8.8. A label for predicted total number of leftovers’

5.1.2.1.1.2.8.9. An input box connected to the ‘predicted total number of leftovers’ label

5.1.2.1.1.2.8.9.1. The placeholder value will be the number of servings produced according to the following formulae: when ‘cook from scratch’ is selected: ‘number of servings to cook’ - ‘number of servings the meal requires’; when ‘use leftovers’ is selected: ‘number of current leftovers’ - ‘number of servings the meal requires’

5.1.2.1.1.2.8.10. If a recipe is a sub-recipe added by another recipe in the meal: a visual indicator that it is a subrecipe

5.1.2.1.1.2.8.11. If a recipe contains possible substitutions: a list of substitutions connected to that recipe, with:

5.1.2.1.1.2.8.11.1. A list for each ingredient, containing the ingredient that may be substituted and possible substitutes for that ingredient that does not include any ingredients already in the recipe, wherein substitutions with sufficient quantity are indicated visually

5.1.2.1.1.2.8.11.2. A radio button paired with each ingredient and possible substitute. The placeholder value will be for the base ingredient radio button to be enabled

5.1.2.1.1.2.8.12. A label for starting cook time

5.1.2.1.1.2.8.13. A label connected to each recipe's starting cook time label

5.1.2.1.1.2.8.13.1. There will be no placeholder value

5.1.2.1.1.2.8.13.2. After a section's meal time is entered (5.1.2.1.1.2.3), the label will present the latest time that that recipe's preparation can start based on the input meal time

5.1.2.1.2. A label for daily calories

5.1.2.1.3. A label connected to the daily calories label

5.1.2.1.3.1. If daily calories cannot be calculated, no value will be displayed

5.1.2.1.3.2. Otherwise, it will display the total number of calories, assuming one serving of each recipe was consumed

5.1.2.1.4. A label for daily proteins

5.1.2.1.5. A label connected to the daily protein label

5.1.2.1.5.1. If daily proteins cannot be calculated, no value will be displayed

5.1.2.1.5.2. Otherwise, it will display the total grams of protein, assuming one serving of each recipe was consumed

5.1.2.1.6. A label for daily fats
5.1.2.1.7. A label connected to the daily fats label
5.1.2.1.7.1. If daily fats cannot be calculated, no value will be displayed
5.1.2.1.7.2. Otherwise, it will display the total grams of fats, assuming one serving of each recipe was consumed
5.1.2.1.8. A label for daily carbohydrates
5.1.2.1.9. A label connected to the daily carbohydrates label
5.1.2.1.9.1. If daily carbohydrates cannot be calculated, no value will be displayed
5.1.2.1.9.2. Otherwise, it will display the total grams of carbohydrates, assuming one serving of each recipe was consumed
5.1.2.1.10. A label for daily cost
5.1.2.1.11. A label connected to the daily cost label
5.1.2.1.11.1. If daily cost cannot be calculated, no value will be displayed
5.1.2.1.11.2. Otherwise, it will display the total cost

5.1.3. There will be a collection of buttons with similar functionality:
5.1.3.1. Each week of the year will have a button with an internal label containing its numeric value (according to ISO 8601)
5.1.3.2. Each month of the year will have a button with an internal label displaying its name
5.1.3.3. There will be a button to calculate information over a specific date range
5.1.3.4. On clicking any of these buttons, a modal containing that time period's nutritional and financial information will appear. In the case of 5.1.3.3, the information will not appear until the user has entered a time period.
5.1.3.4.1. The modal will:
5.1.3.4.1.1. Contain the following information as necessary in a table:
5.1.3.4.1.1.1. Total calories
5.1.3.4.1.1.2. Weekly calories
5.1.3.4.1.1.3. Daily calories
5.1.3.4.1.1.4. Total protein
5.1.3.4.1.1.5. Weekly protein
5.1.3.4.1.1.6. Daily protein
5.1.3.4.1.1.7. Total fat
5.1.3.4.1.1.8. Weekly fat
5.1.3.4.1.1.9. Daily fat
5.1.3.4.1.1.10. Total carbohydrates
5.1.3.4.1.1.11. Weekly carbohydrates
5.1.3.4.1.1.12. Daily carbohydrates
5.1.3.4.1.1.13. Total cost
5.1.3.4.1.1.14. Weekly cost

5.1.3.4.1.1.15. Daily cost

5.1.3.4.1.2. Not display a value if the value cannot be calculated due to missing information

5.1.3.4.1.3. Otherwise it will display a value, with nutritional information assuming one serving of each recipe in the week was consumed

5.1.3.4.1.4. Have a label for starting date

5.1.3.4.1.5. Have an input box connected to the label for starting date

5.1.3.4.1.5.1. The placeholder value will display the date format of DD-MM-YYYY

5.1.3.4.1.5.2. It will accept numbers and punctuation

5.1.3.4.1.5.3. If an invalid date is input, the user should be alerted of this by a visual indicator

5.1.3.4.1.6. Have a label for ending date

5.1.3.4.1.7. Have an input box connected to the label for ending date

5.1.3.4.1.7.1. The placeholder value will display the date format of DD-MM-YYYY

5.1.3.4.1.7.2. It will accept numbers and punctuation

5.1.3.4.1.8. If an invalid date pair is input, the user should be alerted of this by a visual indicator

6. Shopping List

6.1. On opening the page, if the user has not defined a grocery trip frequency and preferred shopping day, and has not not toggled off regular grocery trips, open a modal with:

6.1.1. A label for grocery trip frequency

6.1.2. An input box connected to the grocery trip frequency label

6.1.2.1. There will be no placeholder value

6.1.2.2. It will accept numbers

6.1.2.3. This data will be pushed to Settings

6.1.3. A label connected to the grocery trip frequency input box saying ‘days’

6.1.4. A label for preferred grocery shopping day

6.1.5. A dropdown menu connected to the preferred grocery shopping day

6.1.5.1. There will be a placeholder value of ‘Friday’

6.1.5.2. It will contain:

6.1.5.2.1. Monday

6.1.5.2.2. Tuesday

6.1.5.2.3. Wednesday

6.1.5.2.4. Thursday

6.1.5.2.5. Friday

6.1.5.2.6. Saturday

6.1.5.2.7. Sunday

6.1.6. If the modal is closed without setting a grocery trip frequency and preferred grocery shopping day, toggle off the regular grocery trips setting (7.1.1).

6.2. This page will contain:
- 6.2.1. A list of shopping lists
  - 6.2.1.1. The list will be divided into shopping lists according to the span of meals any given shopping list should cover
    - 6.2.1.1.1. If a user has defined a grocery trip frequency and preferred shopping day, these sections will be automatically sized and chronologically spaced accordingly
    - 6.2.1.1.2. Otherwise, they will be sized according to the user's manually specified shopping lists (6.2.2)
    - 6.2.1.1.3. Each shopping list will attempt to minimize the number of grocery stores to be visited
  - 6.2.1.2. Each shopping list will have:
    - 6.2.1.2.1. An input box for the first date that it satisfies
      - 6.2.1.2.1.1. The placeholder value will be the first day of the satisfied date range
      - 6.2.1.2.1.2. It will accept dates
      - 6.2.1.2.1.3. If an invalid date is input, the user should be alerted of this by a visual indicator
      - 6.2.1.2.1.4. If a date is changed, adjacent shopping lists should have their date ranges updated to prevent meals from not being covered by a shopping list. All affected shopping lists should update their contents accordingly.
    - 6.2.1.2.2. An input box for the last date that it satisfies
      - 6.2.1.2.2.1. The placeholder value will be the last day of the satisfied date range
      - 6.2.1.2.2.2. It will accept dates
      - 6.2.1.2.2.3. If an invalid date is input, the user should be alerted of this by a visual indicator
      - 6.2.1.2.2.4. If a date is changed, adjacent shopping lists should have their date ranges updated to prevent meals from not being covered by a shopping list. All affected shopping lists should update their contents accordingly.
    - 6.2.1.2.3. A toggle box
      - 6.2.1.2.3.1. This toggle box will have a placeholder value of unselected
      - 6.2.1.2.3.2. When toggled, a shopping list will be selected
        - 6.2.1.2.3.2.1. If a shopping list is selected when a user is in the Pantry page and clicks 'add to shopping list' while one or more shopping lists are selected, the selected ingredients will be added to the selected shopping lists.
    - 6.2.1.2.4. A button to 'close out' a shopping list and confirm that it has been used and its contents have been purchased

6.2.1.2.4.1. On click, a shopping list will cease being able to be modified.
6.2.1.2.5. Any ingredients with insufficient quantity to satisfy the planned meals in the time covered by the shopping list
6.2.1.2.6. Any ingredients whose quantity would fall below its threshold quantity to satisfy the planned meals in the time covered by the shopping list
6.2.1.2.7. Any ingredients that would expire before they get used for a meal in the time covered by the shopping list
6.2.1.3. Each ingredient in a shopping list will have:
6.2.1.3.1. A label for its name
6.2.1.3.2. An input box for the quantity being purchased
6.2.1.3.2.1. The predefined value will be the quantity needed to satisfy the menu planned for the shopping list's date range
6.2.1.3.2.2. It will accept numbers and punctuation
6.2.1.3.3. A dropdown menu for the quantity unit
6.2.1.3.3.1. The placeholder value will be the predefined ingredient quantity unit
6.2.1.3.3.2. It will contain imperial masses
6.2.1.3.3.2.1. oz, lb
6.2.1.3.3.3. It will contain imperial volumes
6.2.1.3.3.3.1. gal, qt, pt, cup, ½ cup, ⅓ cup, ¼ cup, fl oz, tbsp, tsp, ½ tsp, ¼ tsp, ⅛ tsp
6.2.1.3.3.4. It will contain metric mass
6.2.1.3.3.4.1. kg, g
6.2.1.3.3.5. It will contain metric volumes
6.2.1.3.3.5.1. dl, cl, ml
6.2.1.3.4. A label for ingredient cost
6.2.1.3.5. A button to remove the ingredient from the shopping list
6.2.1.3.5.1. On click, the ingredient paired with the button should be removed from the shopping list
6.2.2. A button to define a new shopping list
6.2.2.1. On click, open a modal with:
6.2.2.1.1. A label for the first day a shopping trip should satisfy
6.2.2.1.2. An input box connected to the first-day-satisfied label
6.2.2.1.2.1. The placeholder value will be the date format of DD-MM-YYYY
6.2.2.1.2.2. It will accept numbers and punctuation
6.2.2.1.2.3. If an invalid date is input, the user should be alerted of this by a visual indicator
6.2.2.1.3. A label for the last day a shopping trip should satisfy
6.2.2.1.4. An input box connected to the last-satisfied-day label
6.2.2.1.4.1. The placeholder value will be the date format of DD-MM-YYYY
6.2.2.1.4.2. It will accept numbers and punctuation

6.2.2.1.4.3. If an invalid date is input, the user should be alerted of this by a visual indicator

6.2.2.2. If the user input data and applied it, shopping lists should be modified to account for the new shopping list's satisfied date range, and the new shopping list should be inserted in its correct chronological position in the list of shopping lists

7. Settings

7.1. This page will contain:

7.1.1. A toggle button for regular grocery trip (automatic shopping list creation)

7.1.1.1. The placeholder value will be 'toggled'

7.1.2. A label for grocery trip frequency

7.1.3. An input box connected to the grocery trip frequency label

7.1.3.1. There will be no placeholder value

7.1.3.2. It will accept numbers

7.1.4. A label for preferred grocery shopping day

7.1.5. A dropdown menu connected to the preferred grocery shopping day

7.1.5.1. There will be a placeholder value of 'Friday'

7.1.5.2. It will contain:

7.1.5.2.1. Monday

7.1.5.2.2. Tuesday

7.1.5.2.3. Wednesday

7.1.5.2.4. Thursday

7.1.5.2.5. Friday

7.1.5.2.6. Saturday

7.1.5.2.7. Sunday

7.1.6. A button to sign out

7.1.6.1. On click, the user will be signed out and presented the sign-in page

8. Misc

8.1. When a ingredient or leftover has an approaching expiration, a modal will appear informing the user of impending expiration

8.1.1. The user will be informed when 7 days away from expiration

8.1.2. The user will be informed when 3 days away from expiration

8.1.3. The user will be informed when 1 days away from expiration

8.1.4. This modal will mention the ingredient or leftover in question

8.1.5. Any ingredient or leftover that will expire within a day of the ingredient that triggers this modal will also be mentioned, but this will not recursively propagate

8.2. The color scheme should be neutral and light

9. Modal

9.1. All modals will have a button to close the modal, discarding any input information

9.2. All modals taking user input will have a button to apply the user input and close the modal

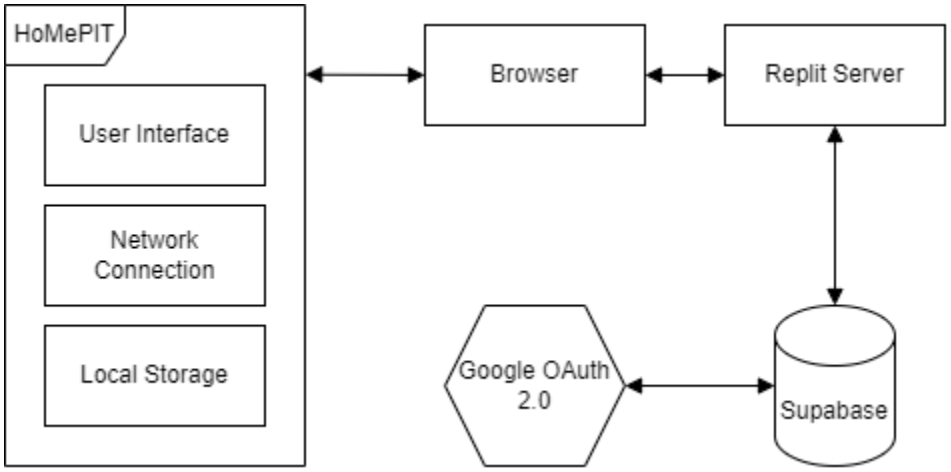
9.3. All modals taking user input will have a button to minimize the modal, retaining any input information

9.4. When a modal is minimized, the page containing the minimized modal will have a button to maximize the modal
9.5. When a modal is minimized, if the original button to open the modal is clicked, the modal will be maximized

Appendix

1. Block Diagram

Figure 1.1: Block Diagram



2. Component Diagram

Figure 2.1: Primary Components

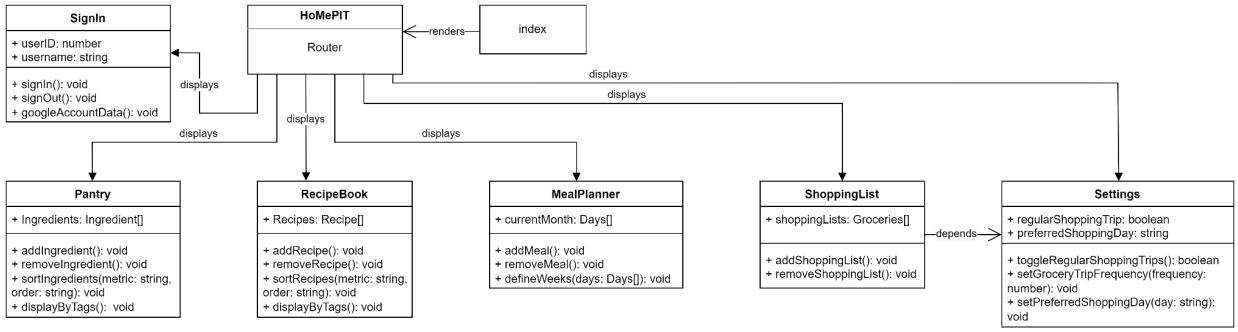
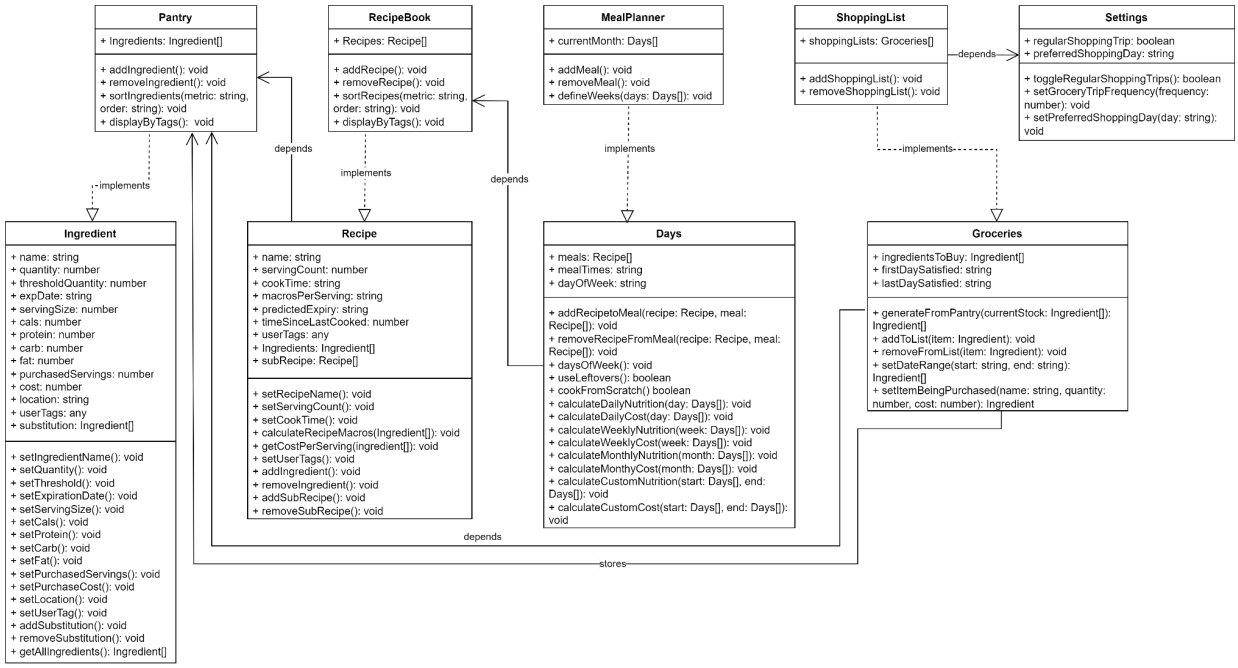


Figure 2.2: Main Pages and Secondary Components



3. UI Storyboard

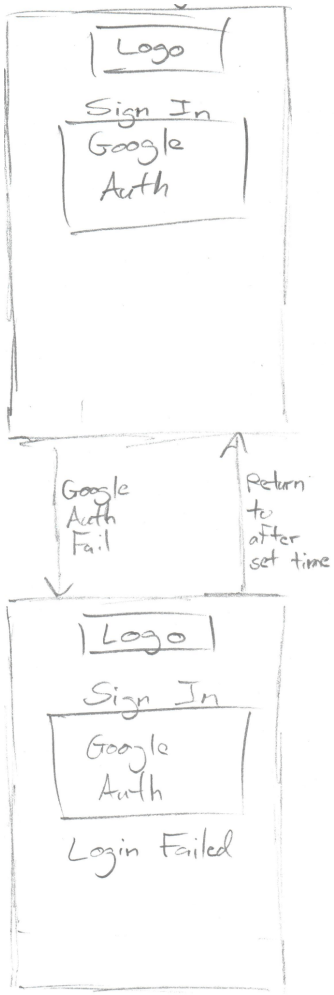


Figure 3.1: Sign in page screens

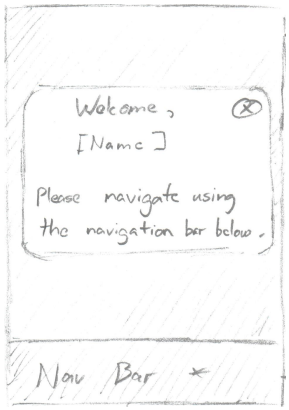


Figure 3.2: Welcome modal

Figure 3.3: Navigation bar

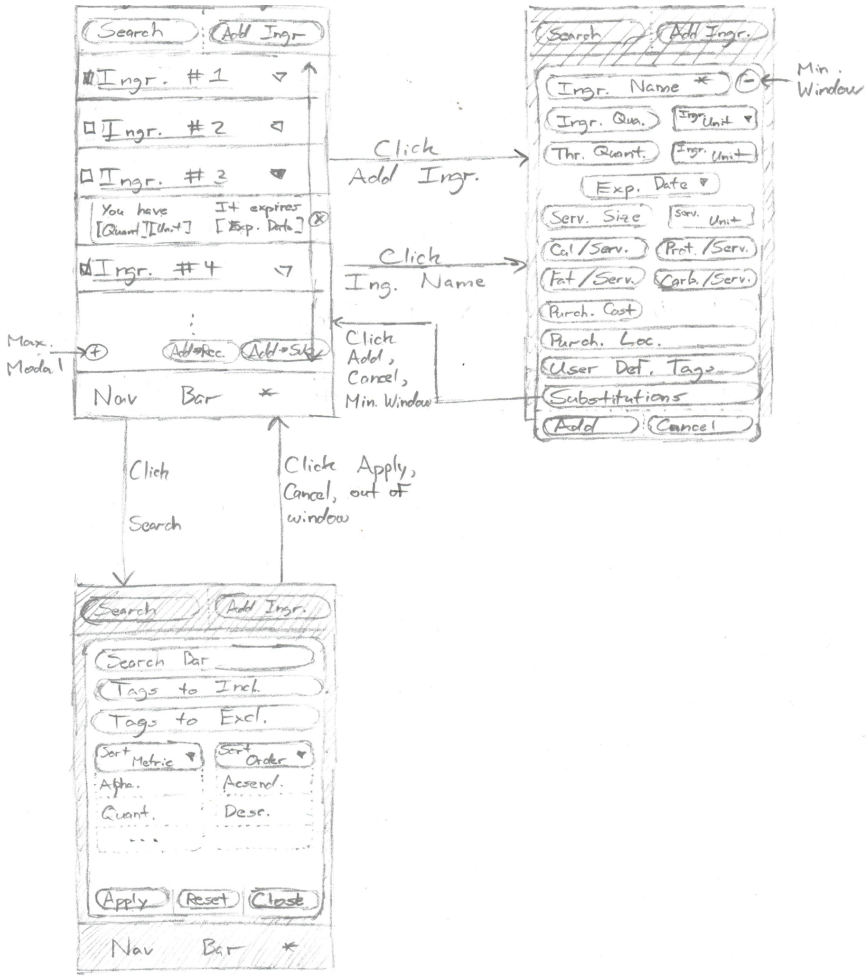


Figure 3.4: Pantry page screens

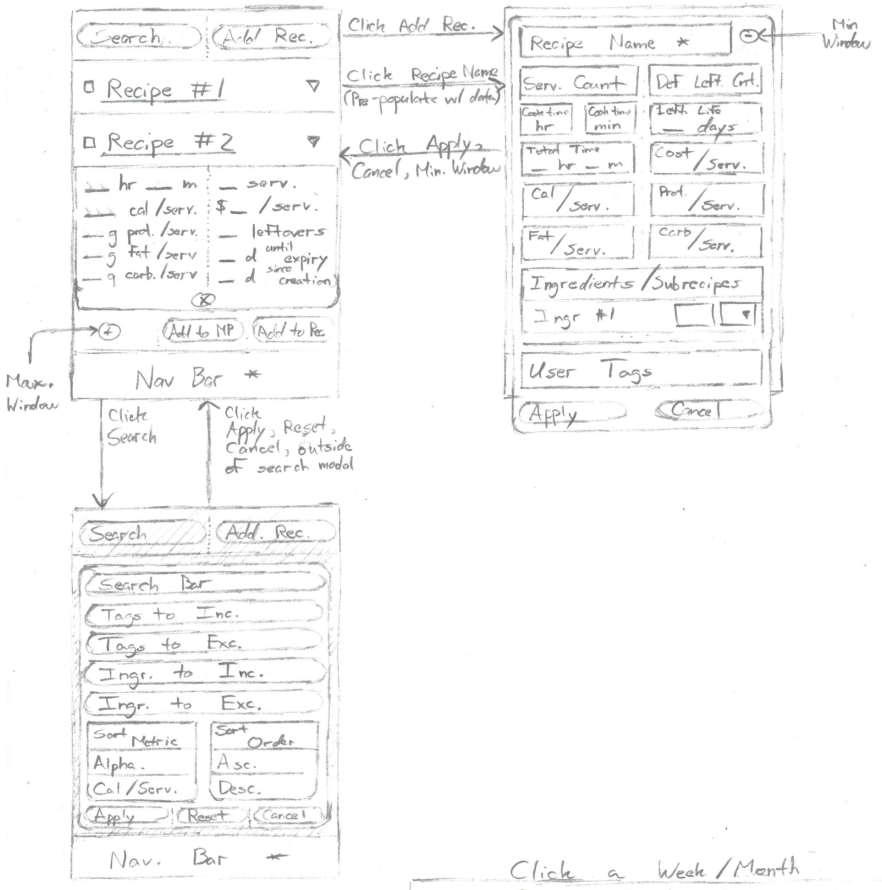


Figure 3.5: Recipe Book page screens

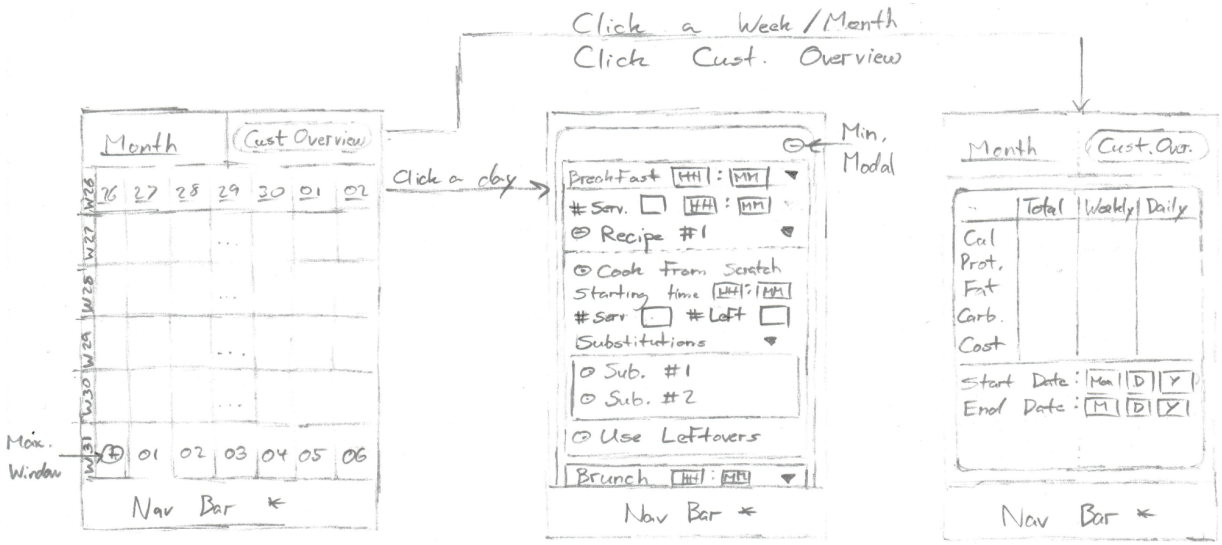


Figure 3.6: Meal Planner page screens

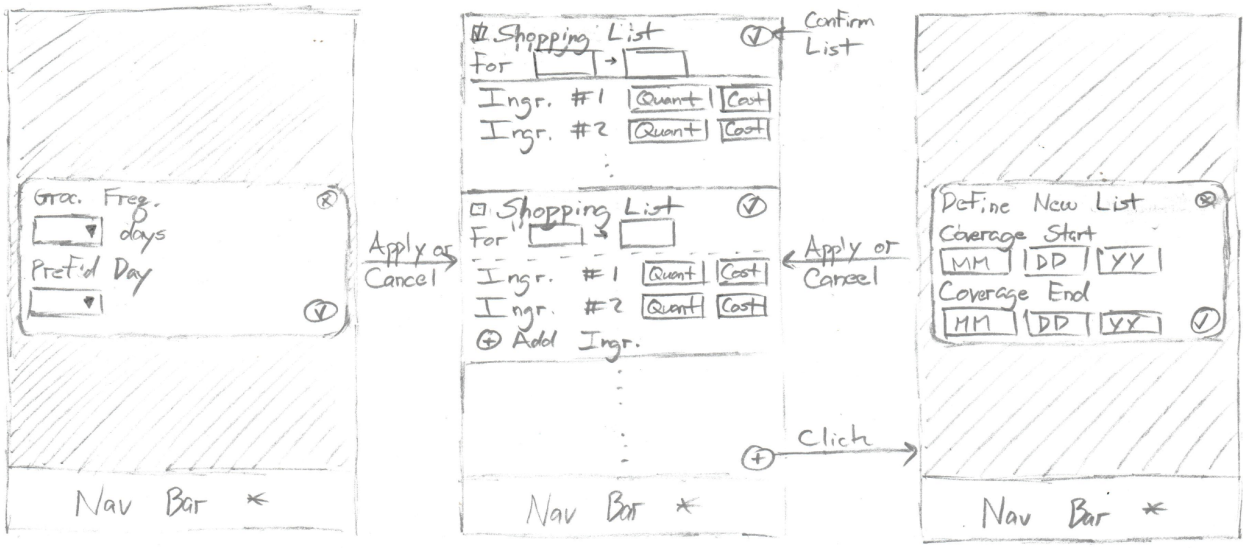


Figure 3.7: Shopping List page screens

Figure 3.8: Settings page screen

4. Database Entity Relationship Diagram

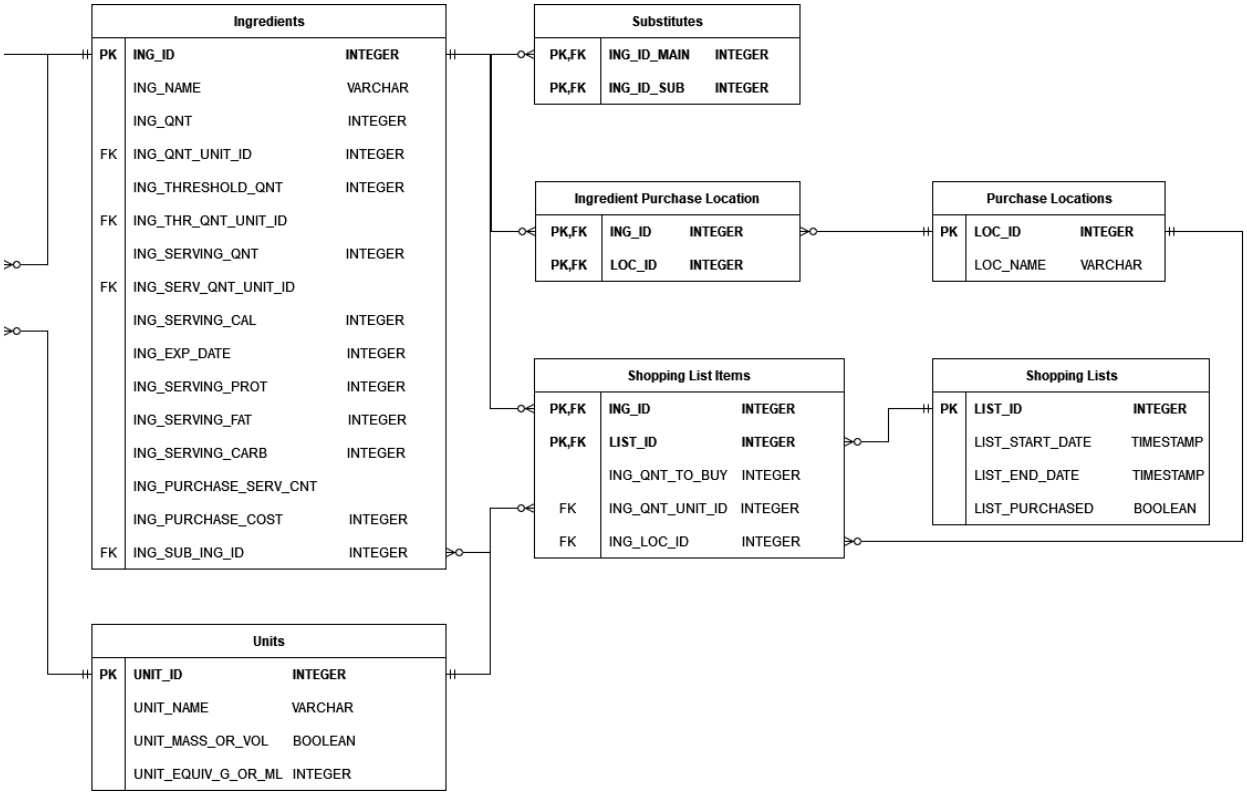


Figure 4.1: The half of the ERD that contains the Ingredients and Shopping List tables, as well as supporting tables.

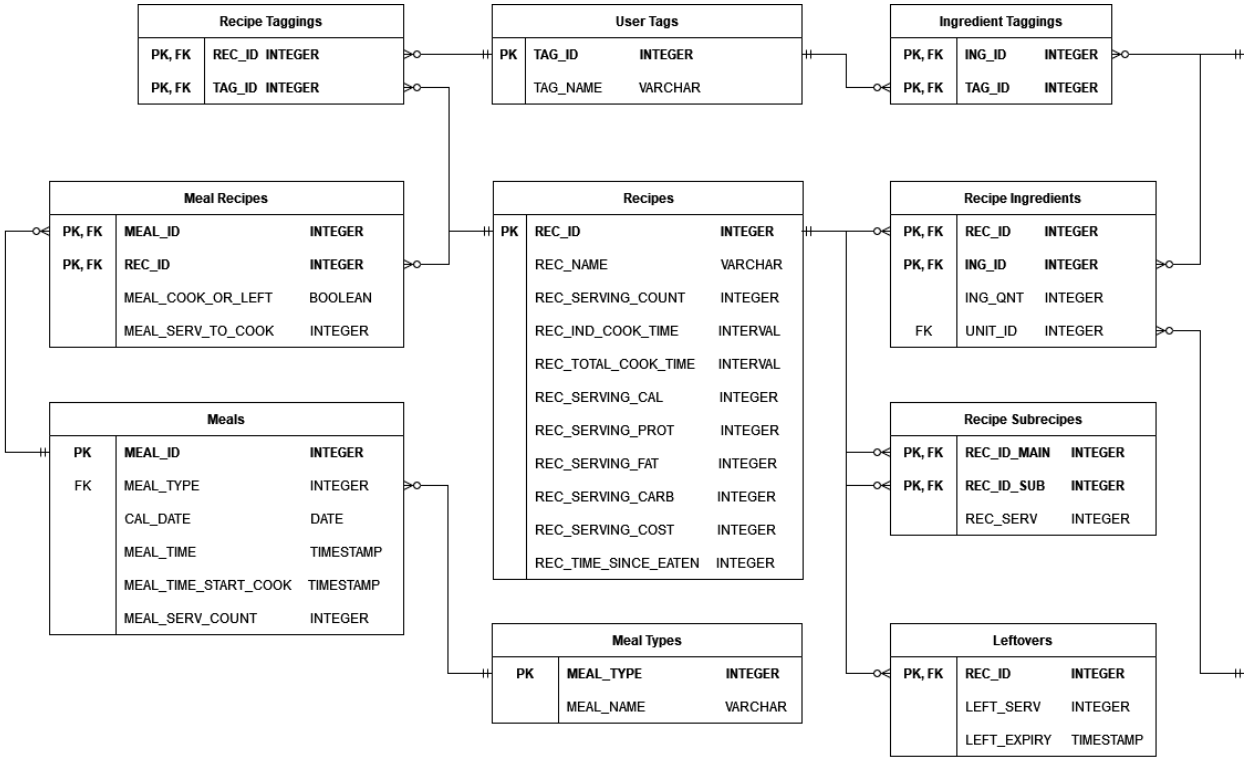


Figure 4.2: The half of the ERD that contains the Recipes and Meals tables, as well as supporting tables.