Software Design Description
for
MBP

Version 2.0
Date: 4/23/24
Department of Math and Computer Science
Biola University

Revisions Page

Overview

The Software Design Description document provides a complete description of the Meal Budgeting System. The document will follow the functionalities identified in the SRS document of the project and will describe how the system will incorporate the ideas that define the system's software requirements. This document will describe how the system will achieve each functionality, feature and requirement as described in the SRS document. Overall, this document provides detailed descriptions of the architecture, design, and implementation of the software system.
In the first section, the document will provide an overview of the software design. Additionally, it contains the Requirements Traceability Matrix which describes the requirements identified in the SRS document of the project. The Requirements Traceability Matrix will provide detail of each requirement including their level of priority and status.
In the second section, the document will describe the System Architectural Design. It will provide detailed descriptions regarding the chosen system architecture for this project and the system interface.
In the third section, the document will provide detailed descriptions of the system components and their design. This includes the user interface, application and database,
In the fourth section, the document will provide a detailed overview of the user interface design with descriptions and visual models of the screens, objects, and actions implemented in the user interface design. This section will provide a wireframe between the screens and describe the error messages, buttons, and navigation bar that will enhance usability.

Target Audience

The target audience of this document is Computer Science students. This document serves as an example for Computer Science students interested in software engineering and project management. The document provides detailed descriptions of how the team will meet the requirements defined in the SRS document in order for the application to perform all of the decided functionalities and contain the decided features.

Students can see the process of our project development as the team works to achieve the goals of our system’s design.

Project Team Members

Rachel Liu, Oscar Navarro, Miguel Oh, Jonathan Wiley

Version Control History

Document Owner: Rachel Liu

Version	**Author**	**Description**	**Date Completed**
1.0	Rachel Liu	Document Creation	2/2/2024
1.0	Oscar Navarro	Requirements Traceability Matrix	2/2/2024
1.0	Oscar Navarro	Figure 1: System Architecture	2/2/2024
1.0	Rachel Liu	Figure 2: User Interface Design	2/2/2024
1.0	Oscar Navarro/ Miguel Oh	Prototype 1	2/2/2024
1.0	Oscar Navarro/ Miguel Oh	Prototype 2	2/2/2024
1.0	Oscar Navarro/ Miguel Oh	Prototype 3	2/2/2024
1.0	Oscar Navarro/ Miguel Oh	Prototype 4	2/2/2024
1.0	Oscar Navarro/ Miguel Oh	Prototype 5	2/2/2024
1.0	Oscar Navarro	Figure 3: Database Model	2/2/2024

1.0	Rachel Liu	Figure 4: Wireframe- Log in	2/2/2024
1.0	Rachel Liu	Figure 5: Wireframe - Navigation Bar	2/2/2024
1.0	Rachel Liu	Figure 6: Meal Selection	2/2/2024
1.0	Rachel Liu	FIgure 7 : Grocery List	2/2/2024
1.0	Rachel Liu	Figure 8: Explore	2/2/2024
1.0	Rachel Liu	Figure 9: My Meals	2/2/2024
1.0	Rachel Liu	Figure 10: My Budget	2/2/2024
1.0	Rachel Liu	Figure 11: My Account	2/2/2024
1.0	Rachel Liu	Figure 12: Navigation Bar	2/2/2024
1.0	Rachel Liu	Figure 13: Account Button	2/2/2024
1.0	Rachel Liu	Figure 14: Login Error	2/2/2024
1.0	Rachel Liu	Figure 15: Create an Account Error	2/2/2024
1.0	Rachel Liu	Figure 16: Meal Selection Error	2/2/2024
1.0	Rachel Liu	Figure 17: View Previous/ Future Meals	2/2/2024
2.0	Rachel Liu	Finalized Requirements Traceability	4/23/24

Table of Contents

1 INTRODUCTION

1.1 Design Overview

The software system design for the Meal Plan Budgeting app consists of the user interface, application and database. The components will work together to provide users with meals for the week according to their selected budget. The user interface will provide a user-friendly interface where users can create an account or log in. Users can select their budget option and select their meals for the week. Users can view their previous meals and upcoming meals as well as explore other meal options to potentially sub out for a current one. Users can view and modify their account, view their grocery list and their grocery spendings over the last five weeks. The application component will present the functionality for the user to do these things. The application will send user input to the database where it will retrieve or store the users information, budget selection, meal selection. The application will retrieve and organize the user's weekly meals as well as the additional meal options according to their budget.

1.2 Requirements Traceability Matrix

Req. ID	Type	Requirement	Source	Status	Priority
MBP-1.0	Required	General App Design	Initial Meeting 01/11/24	Done	Mandatory
MBP- 2.0	Required	Database Consideration	Session #4 01/22/24	Done	Mandatory
UI-1.0	Required	Budget options	Session #4 01/22/24	Done	Mandatory
DB-1.0	Functional	User Account Database	Initial Meeting 01/11/24	Done	High
A -1.0	Functional	Create Account	\ Session #6 02/04/24	Done	High
DB-1.1	Functional	Store Meal selection	\ Session #7 02/06/24	Done	High
DB- 2.0	Functional	Meals entity(Ingredients, Recipes, Calories)	\ Session #4 01/22/24	Done	High
A - 3.0	Functional	Meal options based on budget	Initial Meeting 01/11/24	Done	High
UI-2.3	Functional	Weekly Meal Plan	Initial Meeting 01/11/24	Done	High
UI- 2.1	Functional	Meal Recipe	Initial Meeting 01/11/24	Done	Moderate
A-1.3	Required	Authenticate User	\ Session #2 01/16/24	Done	Moderate

Req. ID	Type	Requirement	Source	Status	Priority
UI-4.0	Functional	More meal options	Initial Meeting 01/11/24	Done	Low
UI-5.0	Functional	User Settings	\ Session #7 02/06/24	Done	Low
A-1.2	Functional	Logout	Session #602/04/24	Done	Low

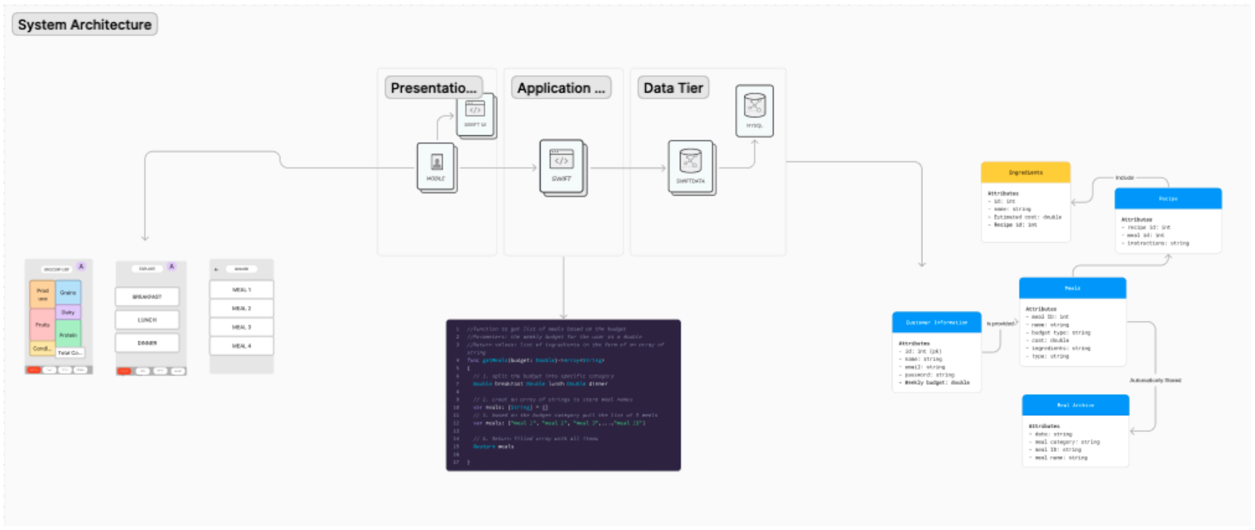
Table 1: Requirements Traceability Matrix

Description:

The Requirements Traceability Matrix displays all of the requirements for the system. Each requirement contains a requirement ID and is categorized by requirement type. The functionality requirement type describes the functions the software is to execute while the Required type indicates that the requirement is mandatory for the successful completion of the requirement. Each requirement is labeled with its level of priority, status in terms of progress and source, indicating the meeting session and date of when the requirement was decided.

2 SYSTEM ARCHITECTURAL DESIGN

2.1 Chosen System Architecture



```
//Function to get list of meals based on the budget
//Parameters: the weekly budget for the user as a double
//Return values: list of ingredients in the form of an array of
string
func getMeals(budget: Double)->Array<String>
{
  // 1. split the budget into specific category
  Double breakfast Double lunch Double dinner

  // 2. creat an array of strings to store meal names
  var meals: [String] = []
  // 3. based on the budget category pull the list of 3 meals
  var meals: ["meal 1", "meal 2", "meal 3",....,"meal 15"]

  // 4. Return filled array with all items
  Return meals

}
```

Figure 1: System Architecture

Description:
The System Architecture Figure portrays the overall system architecture chosen for the project. The Architecture includes the three components of the system- the User Interface, Application, and Database. The multi-tier architecture improves the scalability, maintainability, and overall system flexibility.

2.3 System Interface Description

2.3.1 Presentation Tier

The Presentation Tier is the User Interface that the user will interact with through an Apple device. It is responsible for presenting the meal plans and app features to the users and collecting the users input for things such as budget selection and personal information. The Presentation Tier involves the user experience aspects of the application and focuses on the interaction between the user and the application.

2.3.2 Application Tier

The program will be developed through Swift to create an Apple application, the Meal Budget Plan. The Application Tier contains the core functionality and rules of the application and will interact with the UI and connect to the database for implementation of results. The user input from the Presentation tier is processed in the application tier which then interacts with the data tier for data retrieval and storage. The Application tier, as further described in the components section of the document contains the functionality for retrieving a list of meals based on the users budget, retrieving a list of meals for the week, editing the user settings, providing the recipes for the requested meal, and retrieving a list of additional meal options for the week.

2.3.3 Data Tier

The Data Tier manages the storage and retrieval of the data such as user information, ingredients, meals, and recipes. The Application Tier interacts with the Data Tier through retrieving and storing the information that is essential to the application's operation.

3 DETAILED DESCRIPTION OF COMPONENTS

3.1 User Interface

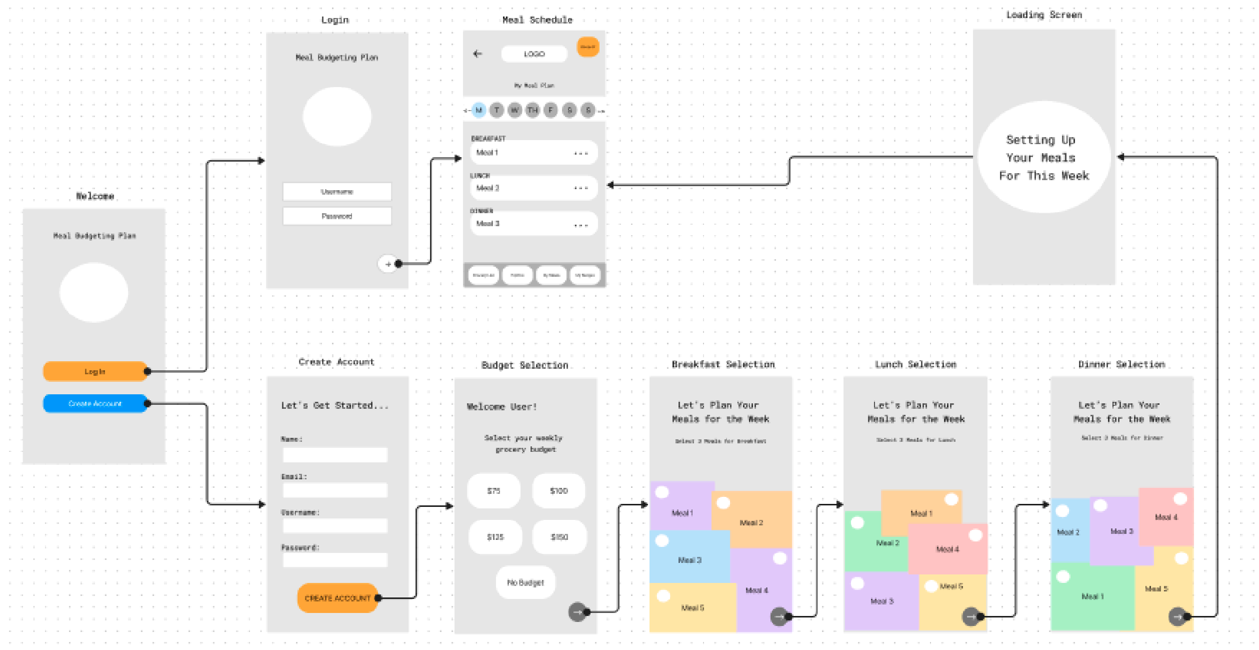


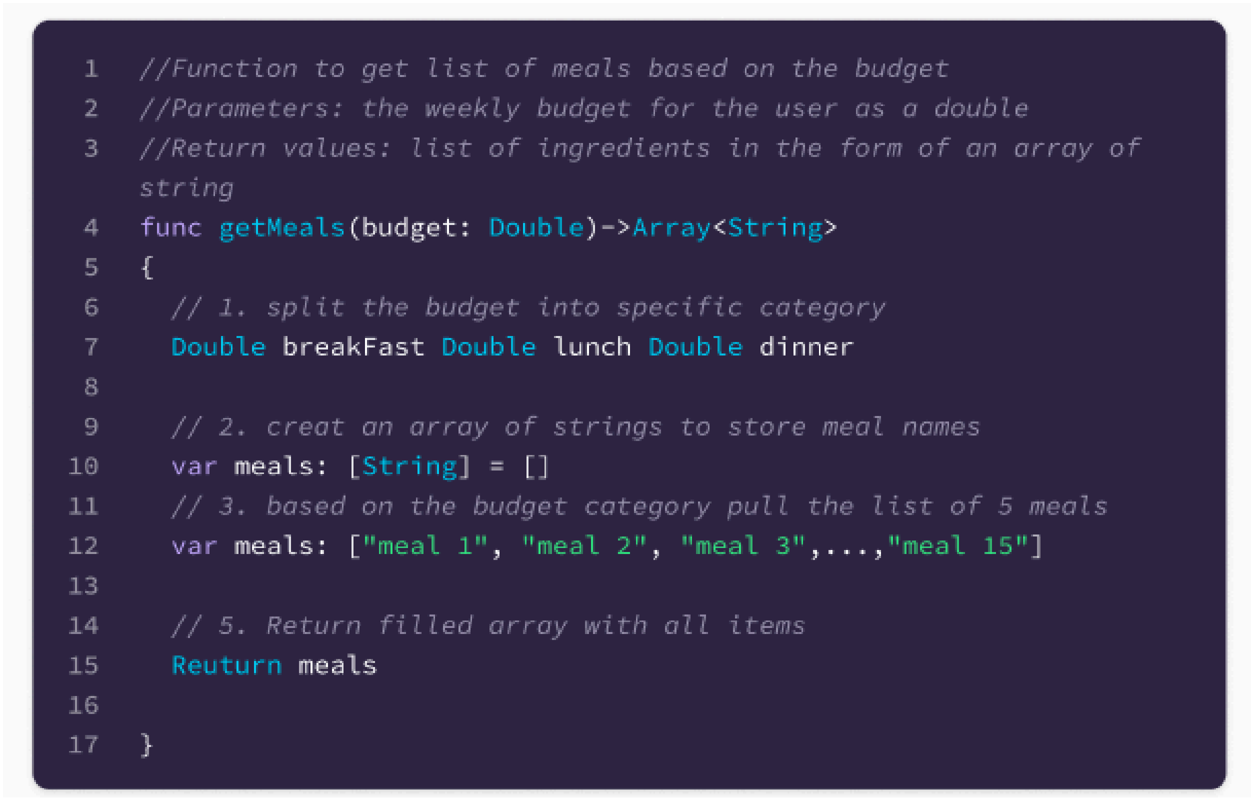
Figure 2: User Interface Design

Description:

The UI is designed to enhance usability. The UI will possess consistent design across the screens and utilize common elements such as a navigation bar and account button amongst the screens to enhance learnability. The UI will possess input controls such as buttons, checkboxes and lists for users with clear labels and purposes. The UI will possess navigational components such as a navigation bar and Account icon amongst every main screen. The UI will also contain informational components such as error messages and help pop ups, creating clear and simple use of the application. The icon buttons on each page are designed to clearly represent the action that follows when the user clicks. Users will be able to easily understand how to access their meals for the week, recipes, edit user settings, view their grocery list and modify meal selection.

3.2 Application

3.2.1 Retrieve Meal Options Based on Budget



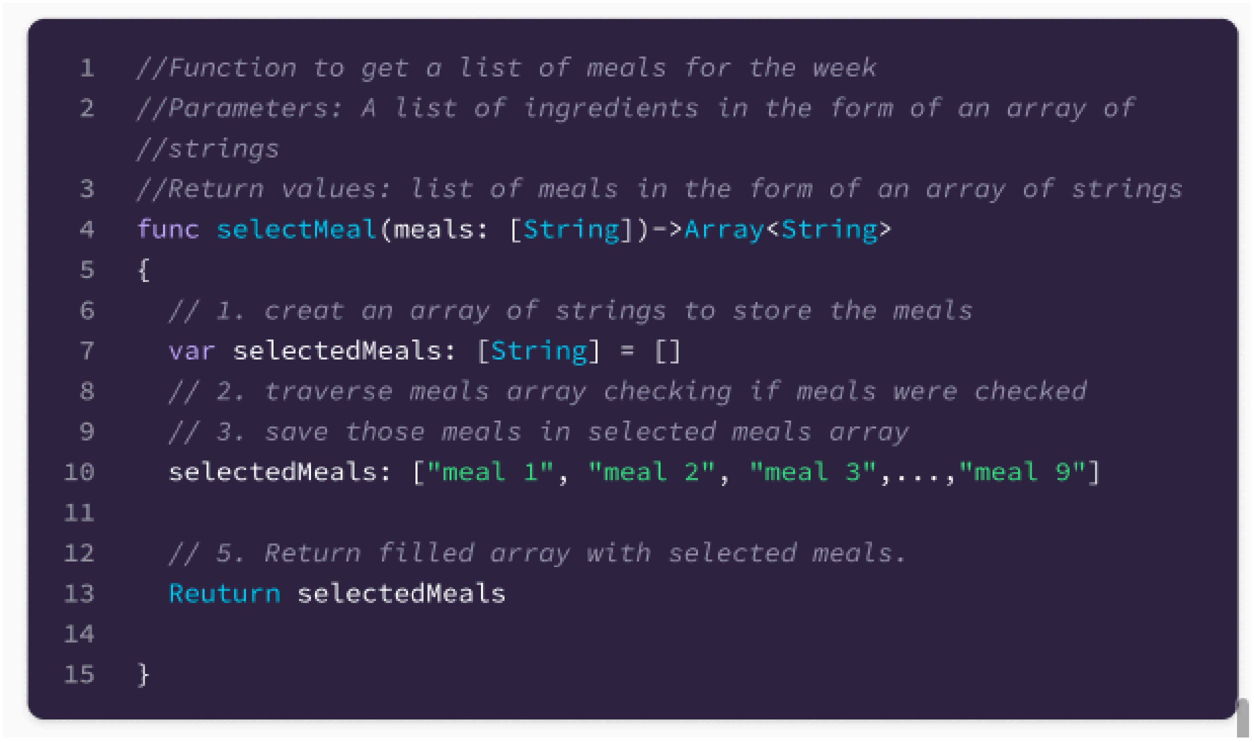
```
//Function to get list of meals based on the budget
//Parameters: the weekly budget for the user as a double
//Return values: list of ingredients in the form of an array of
string
func getMeals(budget: Double)->Array<String>
{
  // 1. split the budget into specific category
  Double breakFast Double lunch Double dinner

  // 2. creat an array of strings to store meal names
  var meals: [String] = []
  // 3. based on the budget category pull the list of 5 meals
  var meals: ["meal 1", "meal 2", "meal 3",...,"meal 15"]

  // 5. Return filled array with all items
  Reuturn meals

}
```

Prototype 1

Description:

The Application Component of the system will include the functionality of retrieving a list of meals from the database based on the user’s selected budget. The function will take the user’s chosen weekly budget and return a list of ingredients which form the meal options for breakfast, lunch and dinner. From this list, user’s will be able to select the meals in which they would like for their week.

3.2.2 Retrieve User Meal Selection



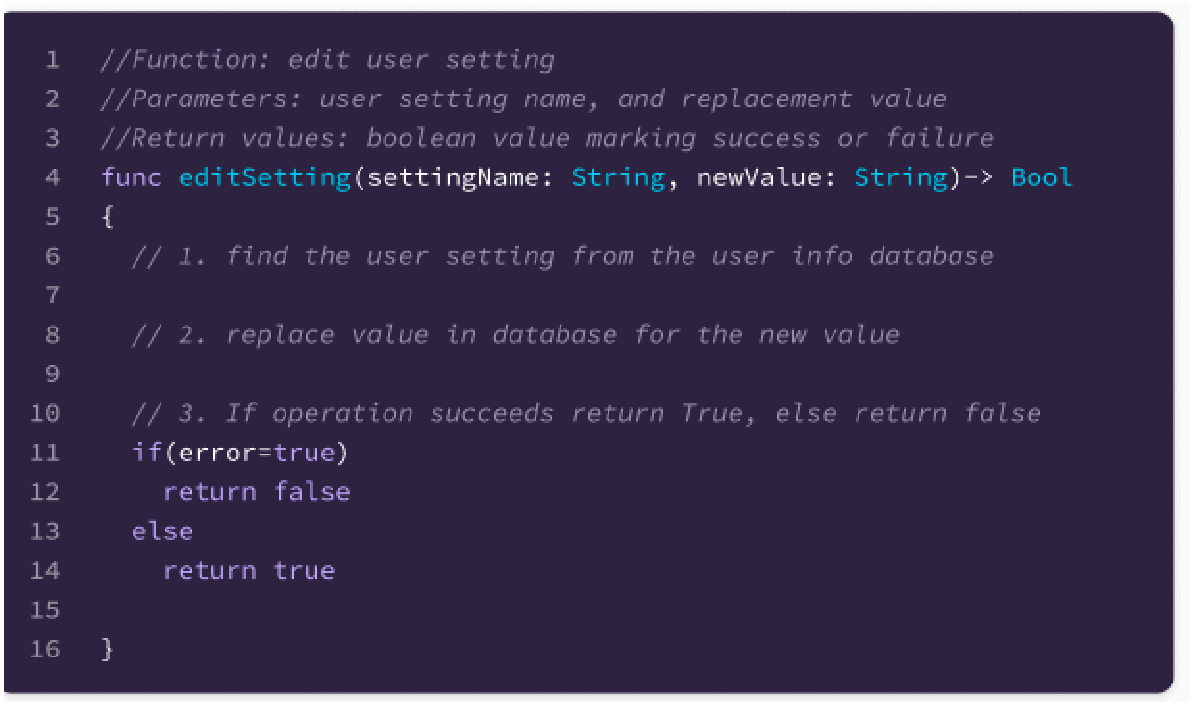
```
//Function to get a list of meals for the week
//Parameters: A list of ingredients in the form of an array of
//strings
//Return values: list of meals in the form of an array of strings
func selectMeal(meals: [String])->Array<String>
{
  // 1. creat an array of strings to store the meals
  var selectedMeals: [String] = []
  // 2. traverse meals array checking if meals were checked
  // 3. save those meals in selected meals array
  selectedMeals: ["meal 1", "meal 2", "meal 3",...,"meal 9"]

  // 5. Return filled array with selected meals.
  Reuturn selectedMeals

}
```

Prototype 2

Description:

The Application component of the system will retrieve a list of meals from the database for the entire week. The function will take a list of ingredients as its parameter and traverse the array of meals to save the ones selected by the user. The function will then return an array containing the selected meals.

3.2.3 Edit Account



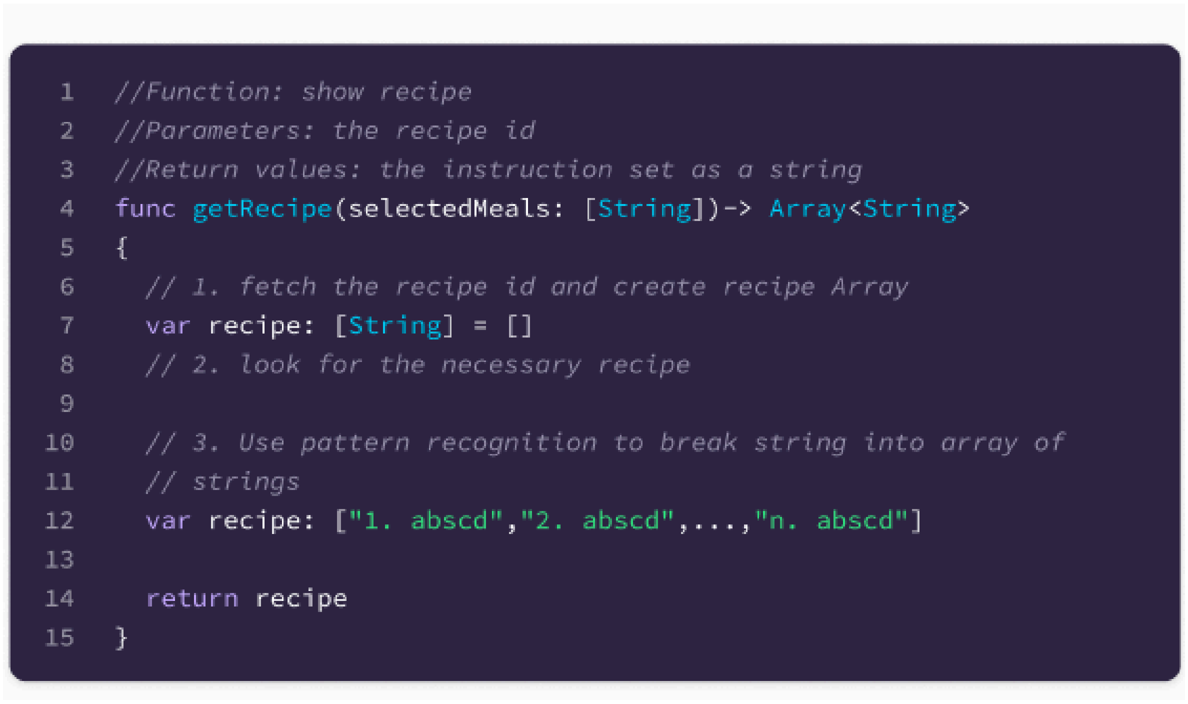
```
//Function: edit user setting
//Parameters: user setting name, and replacement value
//Return values: boolean value marking success or failure
func editSetting(settingName: String, newValue: String)-> Bool
{
  // 1. find the user setting from the user info database

  // 2. replace value in database for the new value

  // 3. If operation succeeds return True, else return false
  if(error=true)
    return false
  else
    return true

}
```

Prototype 3

Description:

The Application component will allow users to modify their setting and account information. The function will find the user's information from the user info database and replace the value in the database with a new value entered by the user. The function will validate successful operations.

3.2.4 Retrieve and Display Recipe



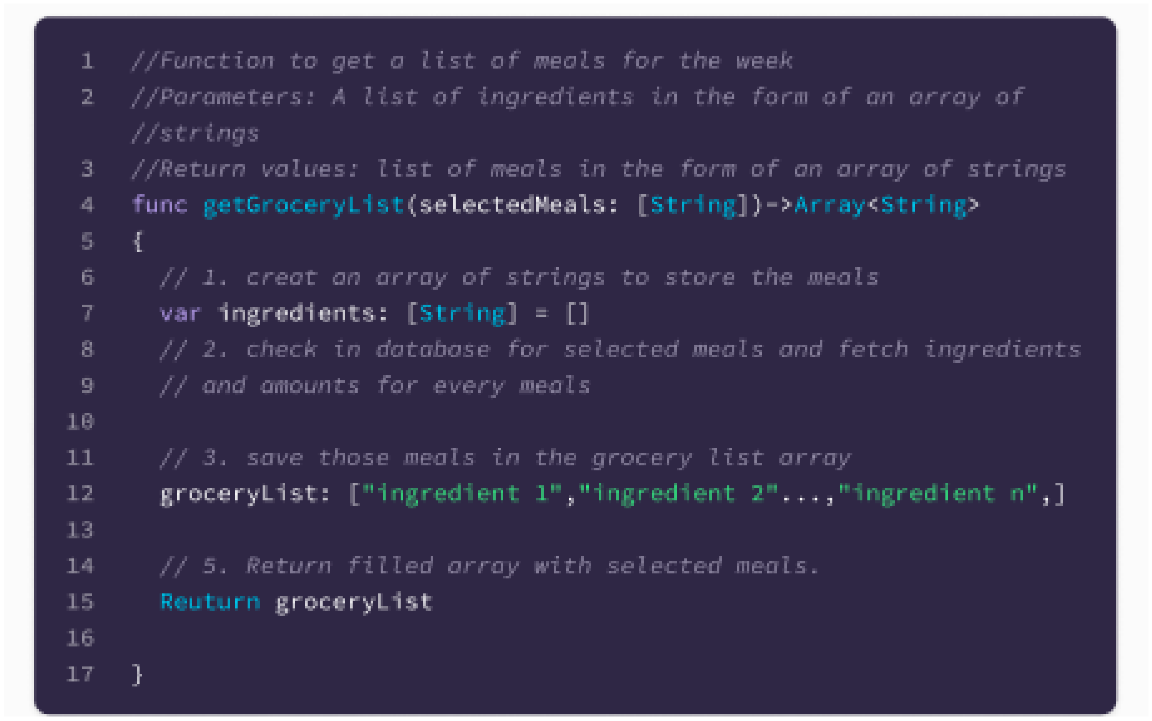
```
//Function: show recipe
//Parameters: the recipe id
//Return values: the instruction set as a string
func getRecipe(selectedMeals: [String])-> Array<String>
{
  // 1. fetch the recipe id and create recipe Array
  var recipe: [String] = []
  // 2. look for the necessary recipe

  // 3. Use pattern recognition to break string into array of
  // strings
  var recipe: ["1. abscd","2. abscd",...,"n. abscd"]

  return recipe
}
```

Prototype 4

Description:

The Application component will retrieve recipe information to display to the user. The function will take the recipe's ID as its parameter and return the instructions for the recipe as a string. It will search for the specific recipe and use pattern recognition to break the recipe string into an array of strings, representing the steps in the recipe.

3.2.5 Retrieve Grocery List



```
//Function to get a list of meals for the week
//Parameters: A list of ingredients in the form of an array of
//strings
//Return values: list of meals in the form of an array of strings
func getGroceryList(selectedMeals: [String])->Array<String>
{
  // 1. creat an array of strings to store the meals
  var ingredients: [String] = []
  // 2. check in database for selected meals and fetch ingredients
  // and amounts for every meals

  // 3. save those meals in the grocery list array
  groceryList: ["ingredient 1","ingredient 2"...,"ingredient n",]

  // 5. Return filled array with selected meals.
  Reuturn groceryList

}
```

Prototype 5

Description:

The Application component will gather the grocery list for the user. It will gather the ingredients of each of the planned meals for the week and compile a list, categorized by grocery type. The list will also contain the average costs, along with the total cost of the groceries.

3.3 Database

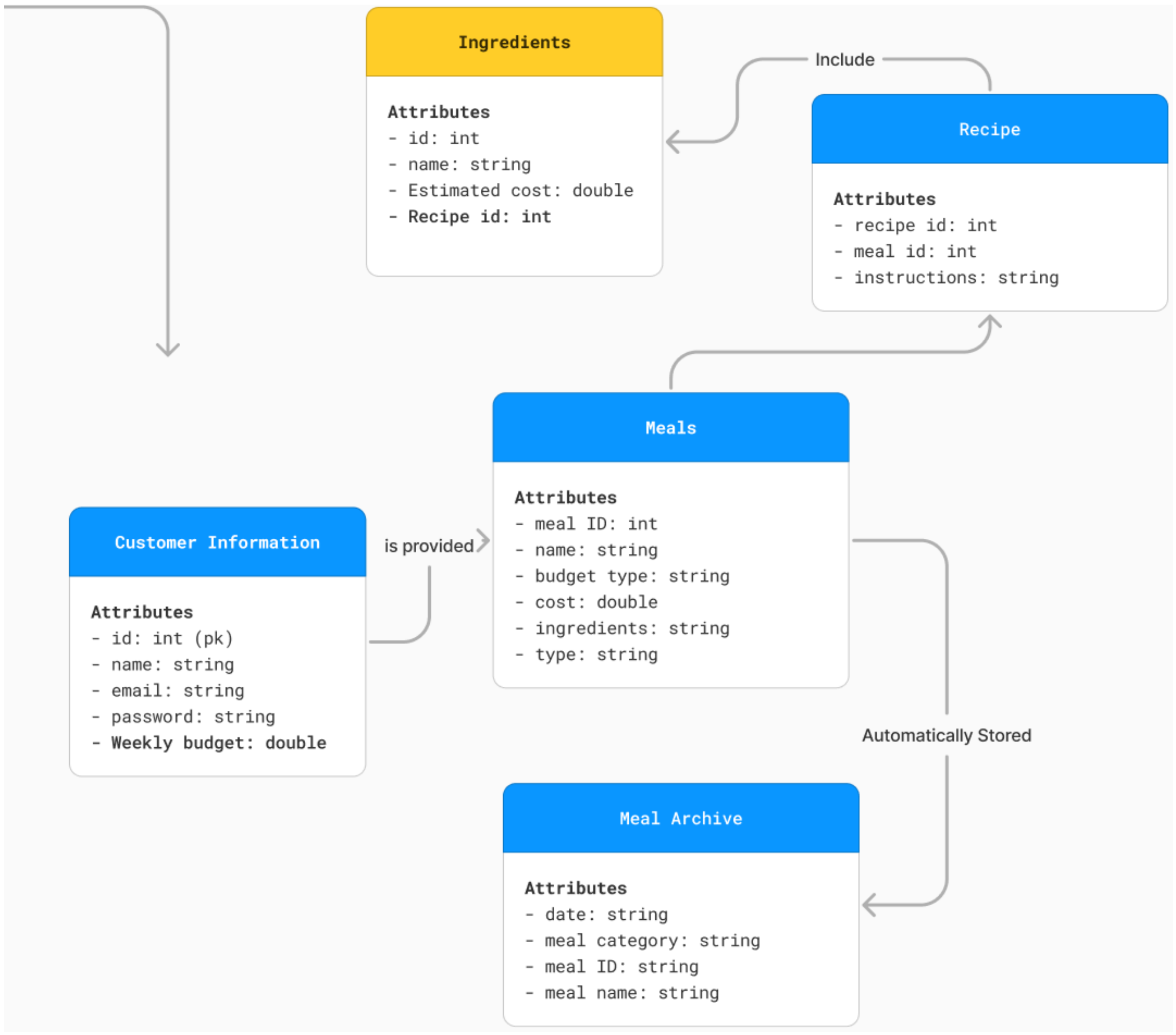


Figure 3: Database Model

Description:

The Database Model displays the database component of the system. The database will possess the user's information such as id, name, email, password and weekly budget. With the user's information stored in the database, user's are able to create accounts and log into existing accounts. Users may also modify the data stored in the database regarding their budget selection and the other attributes. Based on the weekly budget stored in the database, the meals are provided from the meal data. The meal data possesses the cost, budget category for the meal and ingredients. Through this, the application component is able to retrieve the correct meal options based on the user’s budget. The meal data will coordinate with the recipe data where the meal’s instructions are stored. The Recipe data will include the data for the ingredients needed in the recipe.

3.3.1. Entities

The following list contains the Entities for the Database:

- Customer Information - Stores information of the User ID, name, e-mail, password, and budget history
  Attributes:
    - CustomerID - Int
    - Name - VarChar (50)
    - Email - VarChar (20)
    - Password - VarChar (30)
    - Weekly Budget - Double

- Meals - Stores Meal ID, name, calories, cost, recipe, and type of ingredient
  Attributes:
    - MealID - Int
    - Meal Name - VarChar (50)
    - Budget Type - String
    - Cost - Double
    - Ingredients - VarChar (30)
    - Type - VarChar (30)

- Meal Archive - Stores date, meal category, meal name, and the Meal ID
  Attributes:
    - MealArchiveID - Int
    - Date - Date
    - Meal Category - VarChar (30)
    - MealID - Int
    - Meal Name - VarChar (30)

- Recipe
  Attributes:
    - RecipeID
    - MealID
    - Instruction - String

- Ingredients - Stores the Food ID, name, calories, and estimated cost
  Attributes:

    - IngredientID - Int
    - IngredientName - VarChar (30)
    - Estimated Cost - Double
    - RecipeID

3.2.3 Business Rules

The following list describes the business rules of the database component:

- Customer requests weekly meal
- Meals gives result to one or many customers
- Meals inquires one or many ingredients from Recipe
- Recipe sends one or many results to Meals
- Meals stores information to one or many Meal Archive
- Meal Archive sends information to one or many Meals
- Recipe includes one to many Ingredients
- Ingredients belong in one to many Recipes

4 USER INTERFACE DESIGN

4.1 Description of the User Interface

As discussed in section 3.1 User Interface, the UI is designed to enhance usability. The UI will possess consistent design across the screens and utilize common elements such as a navigation bar and account button amongst the screens to enhance learnability. The UI will possess input controls such as buttons, checkboxes and lists for users with clear labels and purposes. The UI will possess navigational components such as a navigation bar and Account icon amongst every main screen. The UI will also contain informational components such as error messages and help pop ups, creating clear and simple use of the application. The icon buttons on each page are designed to clearly represent the action that follows when the user clicks. Users will be able to easily understand how to access their meals for the week, recipes, edit user settings, view their grocery list and modify meal selection.

4.1.1 Screen Images

4.1.1.1 Wireframe- Login/ Create an Account

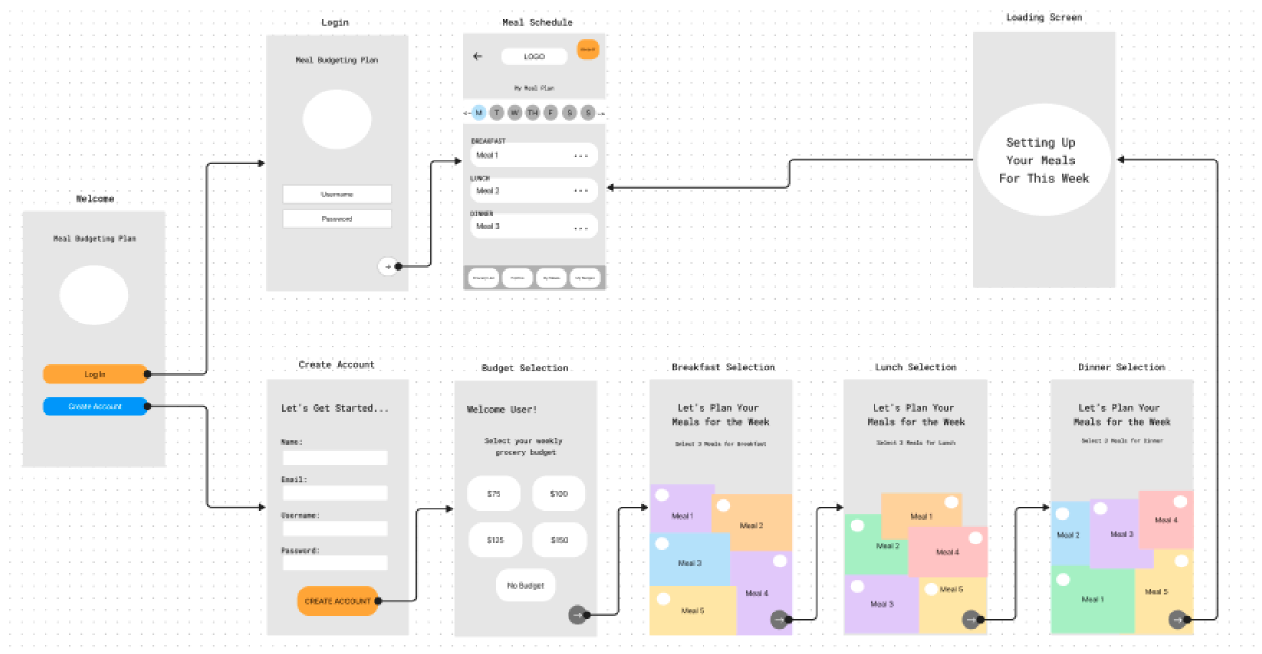


Figure 4: Wireframe- Login

Description:
Users can log in to the system or register in to the system. Users will click on the Login button to log into their account or click the Create an Account button to create their account. When registering in the system, users will be prompted to enter their name, email, and password. Once the personal information is submitted, the user will be prompted to choose from 4 budget options or select no budget. After budget selection, users will press the arrow button to proceed. Then, users will be shown 5 Meals for each breakfast and be prompted to select three. Once three are selected, the user will click the arrow button and continue to do the same for lunch and dinner. Once the dinner selection is complete, users will click on the arrow button and be taken to a loading screen while the meal selection data is retrieved from the database. Once loaded, the user will be taken to the main screen which displays their scheduled meals for the day and the week.
Step-by- Step (Create an Account):

1. If User does not have an account, user will select Create Account button
2. The user enters their name, email, password to create an account
3. User clicks Create Account button
4. The user selects a budget option, clicks the arrow button to continue
5. The user will select 3 meals for breakfast by clicking on the white checkbox space on each image button. User will click arrow button to proceed
6. Users will repeat step 5 for lunch and dinner.
7. The user will then access the main screen of the app and view their week's meal schedule

Step-by- Step (Log in):

1. User will click log in
2. User will enter their email and password and press the arrow button to proceed

4.1.1.2 Wireframe- Navigation Bar

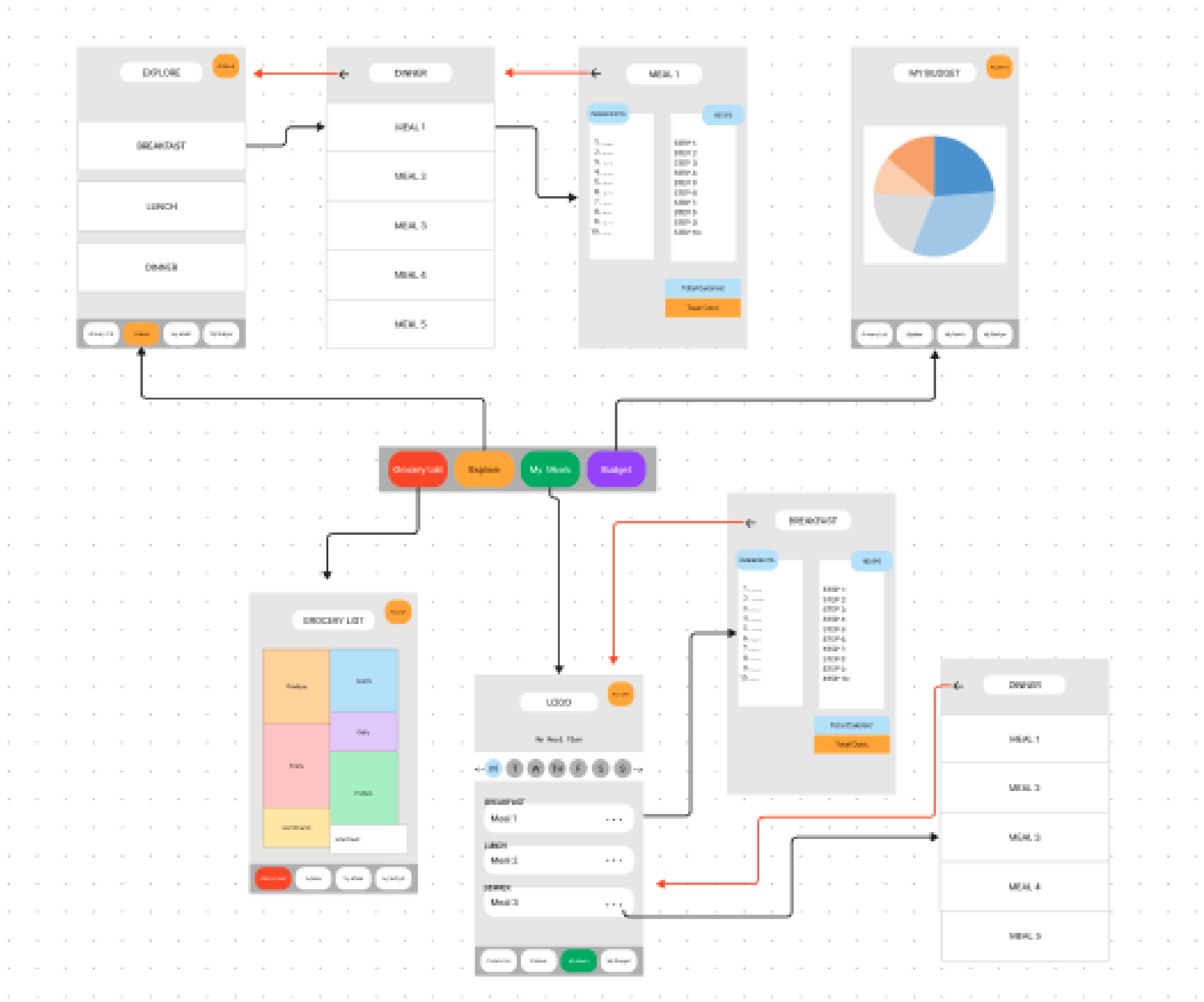


Figure 5: Wireframe- Navigation Bar

Description:

The navigation bar will be displayed on every main screen- grocery list screen, explore screen, my meals screen, budget screen. The buttons will clearly represent their corresponding page. Users will select the screen they wish to go to from the navigation bar. When entering additional pages from the main pages, users can return to the main screen through the back arrow button.

4.1.1.3 Meal Selection

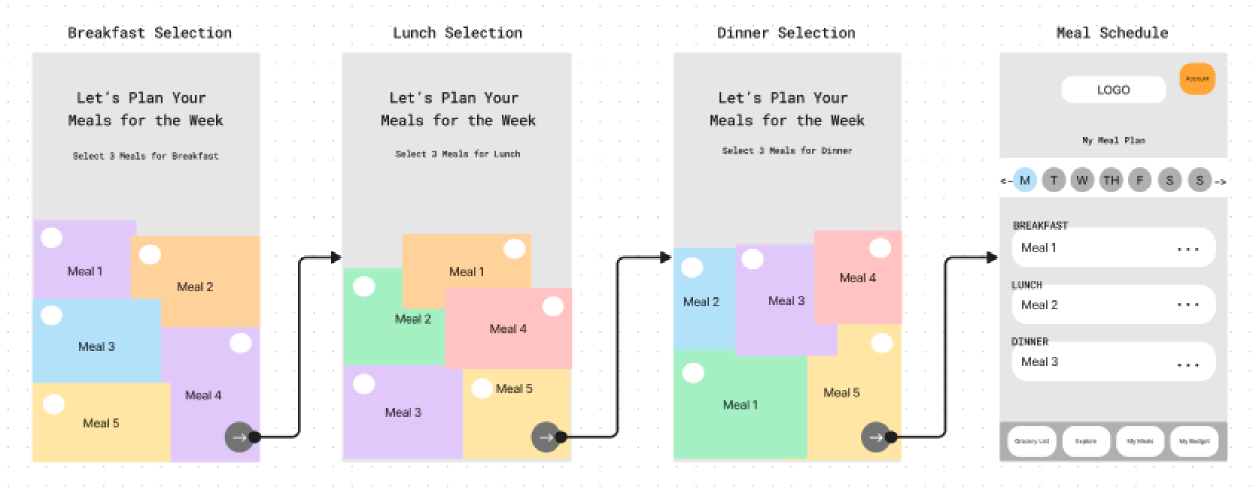


Figure 6: Meal Selection

Description:

When creating an account and on every Sunday, users will be prompted to select their meals for the week. Users will be prompted to select three meals from the list of meals that is based on their budget. Users will click on the checkboxes to select the meal. When selected, the checkbox will turn green and possess a green check mark indicating that it has been selected. Users may deselect the meal by pressing on the check box and the green check mark will disappear. Once three meals have been selected, users will be able to click on the arrow button to continue to the lunch selection and dinner selection. Once complete, the meals for the week will appear on the Meal schedule screen.

4.1.1.4 Grocery List

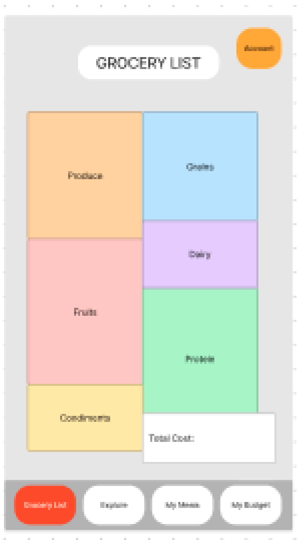


Figure 7: Grocery List

Description:

From the navigation bar, users can click on the grocery list button to view their grocery list for the week. The grocery list is organized by item type to ensure efficient grocery shopping. The grocery list will also display the total amount

4.1.1.5 Explore

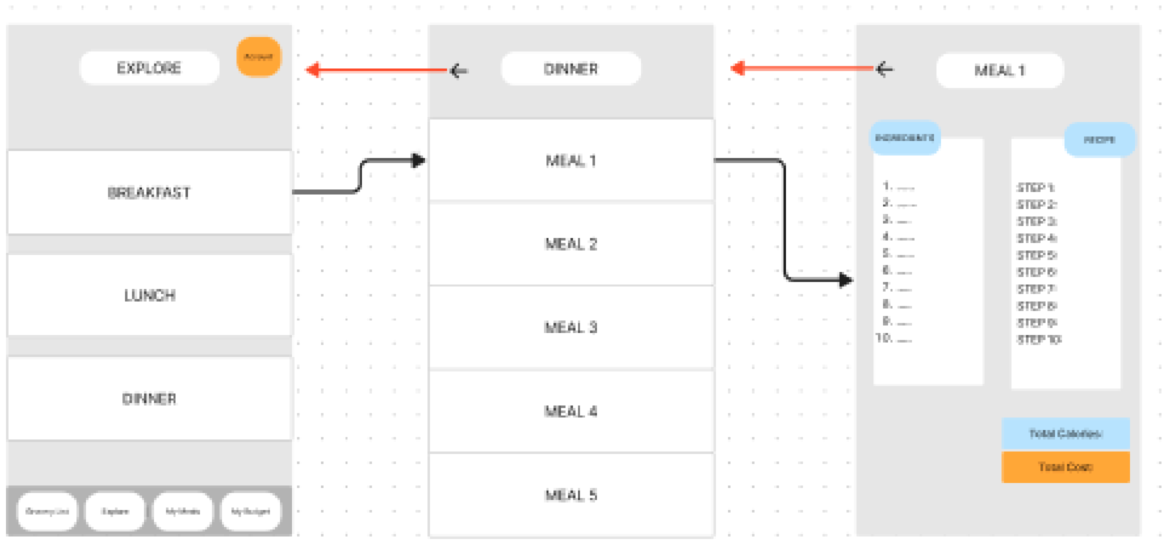


Figure 8: Explore

Description:

From the navigation bar, users can click on the explore button to view other meal options. Once on the explore screen, users can click on which category they would like to view other meals for. After clicking on the category, users will be given a scrolling list of meal options where users can view the recipe, ingredients, total cost and calories of the meal. Users can choose to replace a current meal with the viewed meal if desired by clicking on the "Replace Meal" button. Users can click on the back arrow button to return to the explore page.

4.1.1.6 My Meals/ View Recipe/ Modify Meal Selection

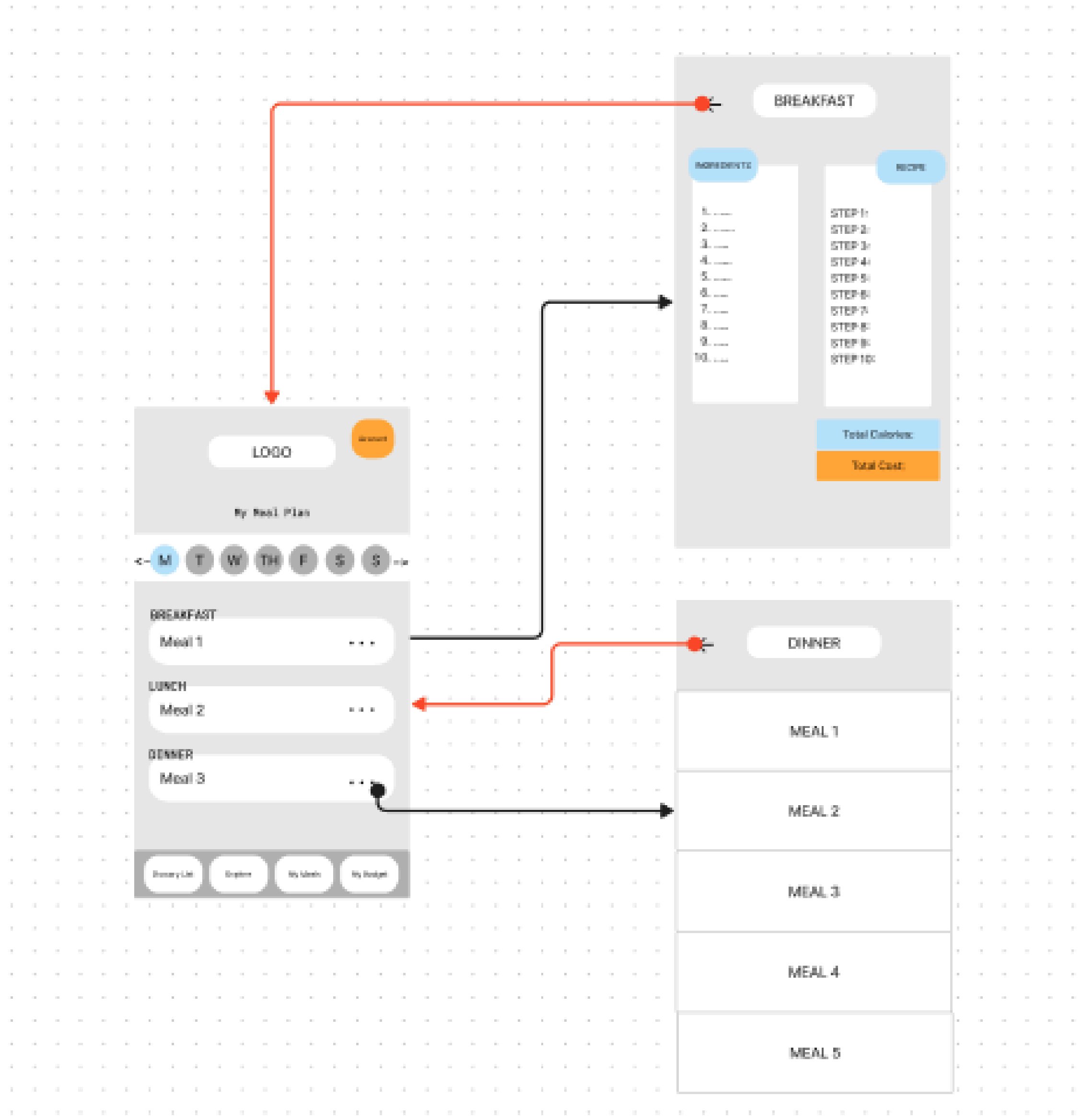


Figure 9 : My Meals

Description:

From the My meals page, which is the default page but can also be accessed from the navigation bar, users can view the meals scheduled for their day and for the other days in the current and previous week. When viewing the meal, users can click on the image button representing that meal and see the ingredients, recipe, cost and calories of the meal. Users can click on the back arrow to return to the My Meals screen. Users can view other meal options by clicking on the “...” next to the meal names. By clicking there, users will be shown a list of other meal options where they can view the recipes and replace a current meal with the meal option. The back arrow will allow users to return to the previous page and back to the My Meals screen

4.1.1.7 My Budget

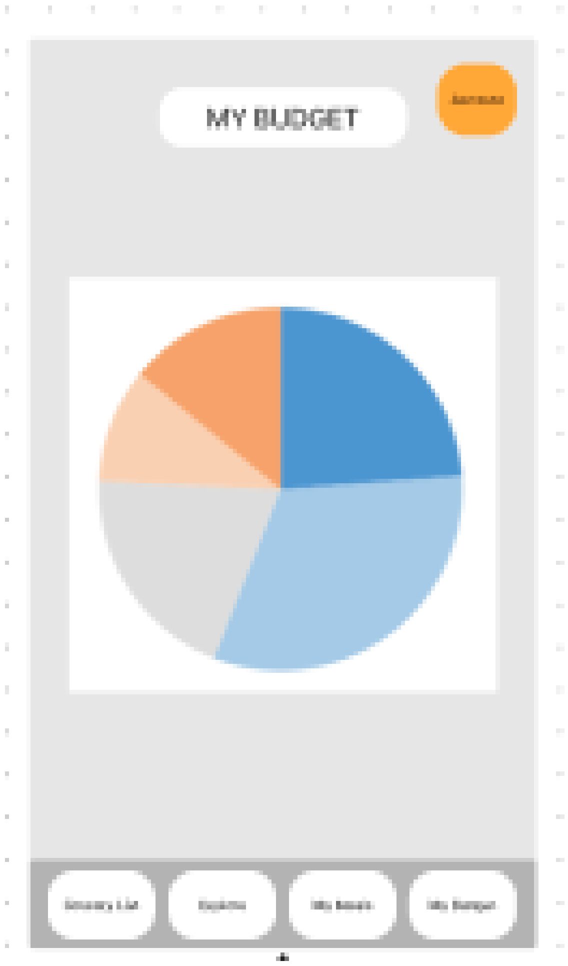


Figure 10: My Budget

Description:

From the navigation bar, users can view their weekly grocery spendings by clicking on the “My Budget” button. On this page, users can see their grocery total cost for the last 5 weeks. Once a week has passed, the application will update the pie chart and add the current week’s spendings. The pie chart will be color coded, distinguishing between weeks and allowing the users to easily understand the data.

4.1.1.8 My Account

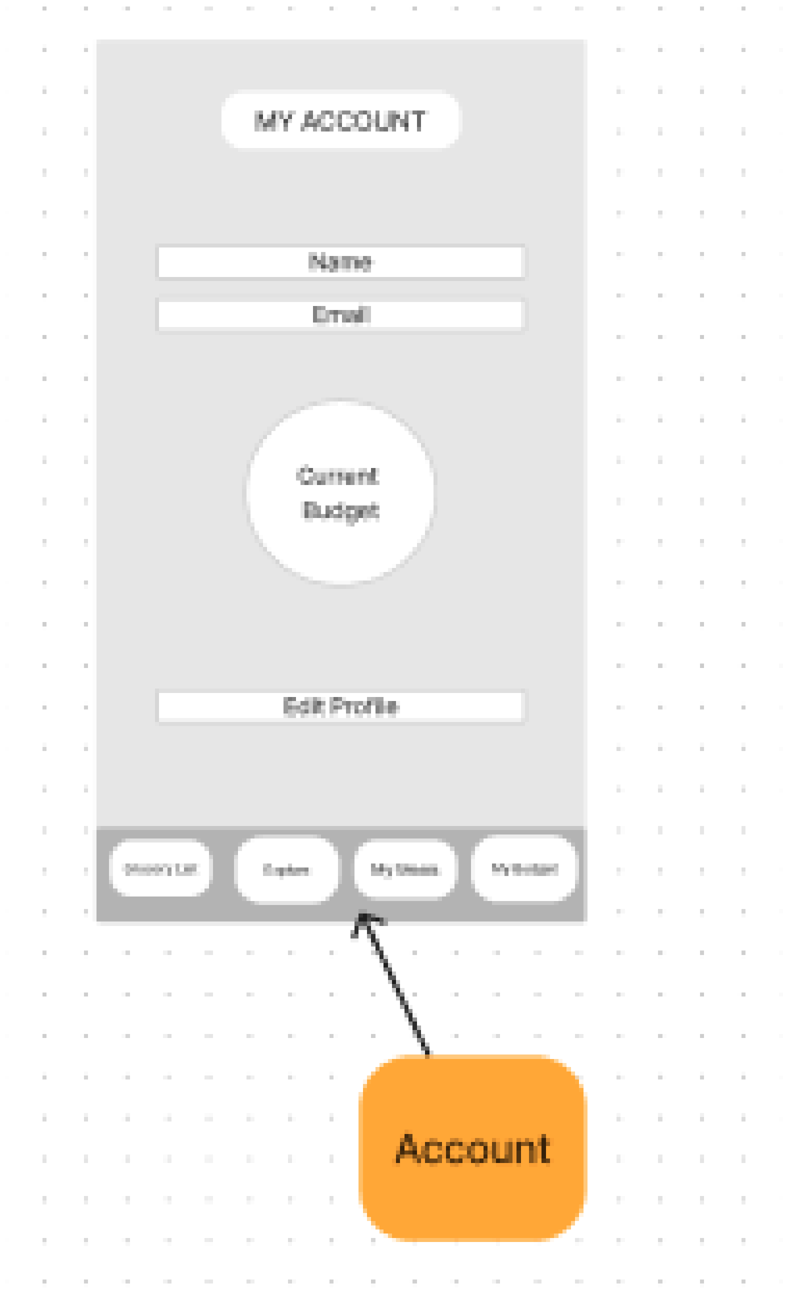


Figure 11: My Account

Description:
Every main page will possess an account button in the top right corner. Users can click on this button to view their account information and current button. Users will also be able to edit their profile and budget preferences.

4.1.2 Objects and Actions

4.1.2.1 Navigation Bar

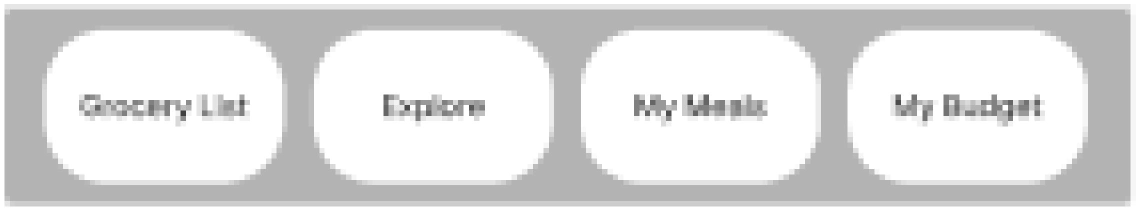


Figure 12: Navigation Bar

Description:

The Navigation Bar will appear on every main page - Grocery List page, Explore page, My Meals page, My Budget Page. Users can click the button to go to the corresponding page. The Grocery List button will lead to the Grocery List page where users can view their grocery list for the week. The explore button will lead to categories of other meal options which users can view and optionally select the meal. The My Meals button will lead to the daily meals that are planned. The My Budget page will lead to a diagram presenting the user's grocery spendings over the last 5 weeks.

4.1.2.2 Account Button

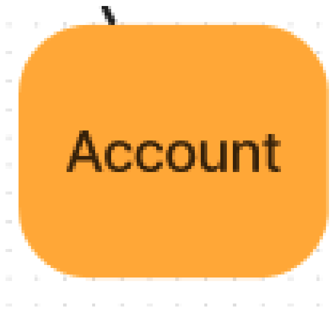
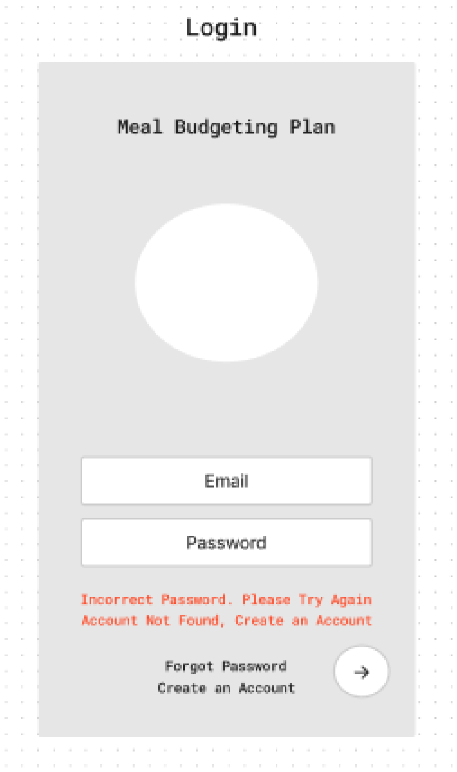


Figure 13: Account Button

Description:
The Navigation Bar will appear on every main page - Grocery List page, Explore page, My Meals page, My Budget Page. Users can click the button to view their account information and budget selection. Users can also edit their preferences and information on this page.

4.1.2.3 Login- Error Messages



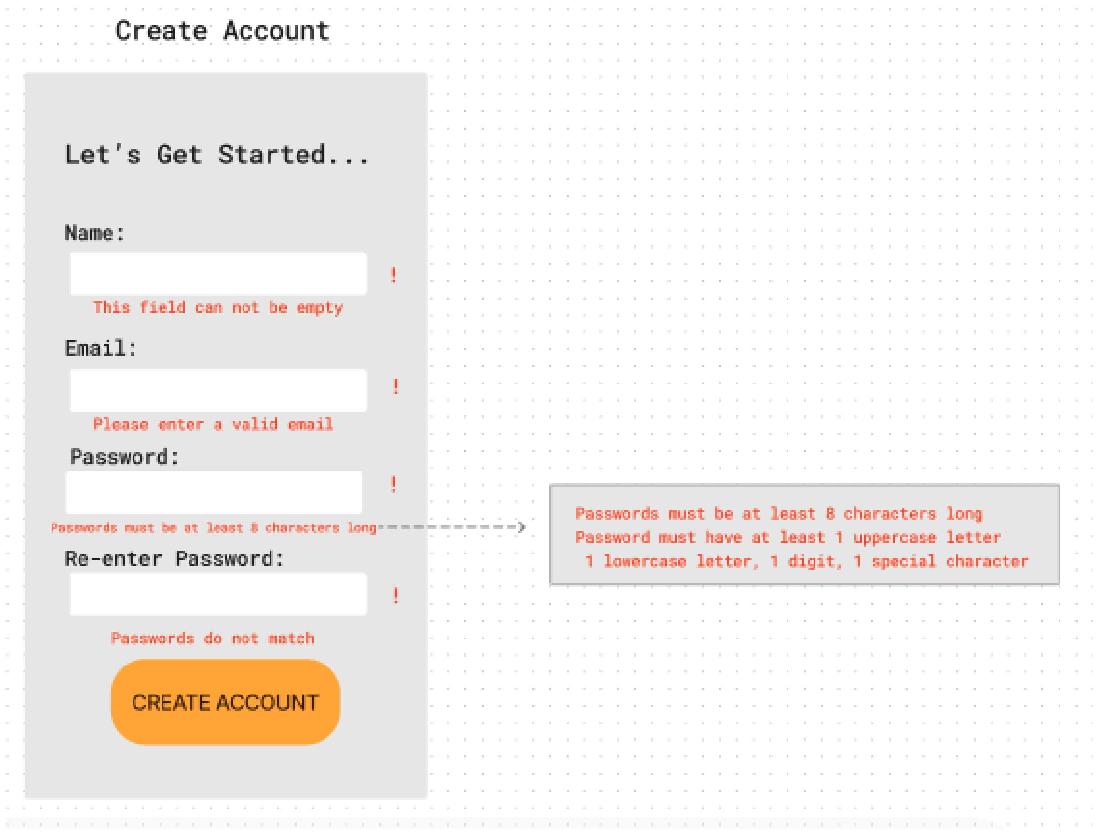
Incorrect Password. Please Try Again
Account Not Found, Create an Account

Figure 14: Login Error

Description:

Users will be given error messages if a password is entered incorrectly. Additionally, if an email is entered that is not found in the system, the UI will display a message notifying the user that the account was not found and prompt the user to create an account.

4.1.2.4 Create an Account - Error Messages



This field can not be empty

Please enter a valid email

Passwords must be at least 8 characters long

Passwords must be at least 8 characters long
Password must have at least 1 uppercase letter
1 lowercase letter, 1 digit, 1 special character

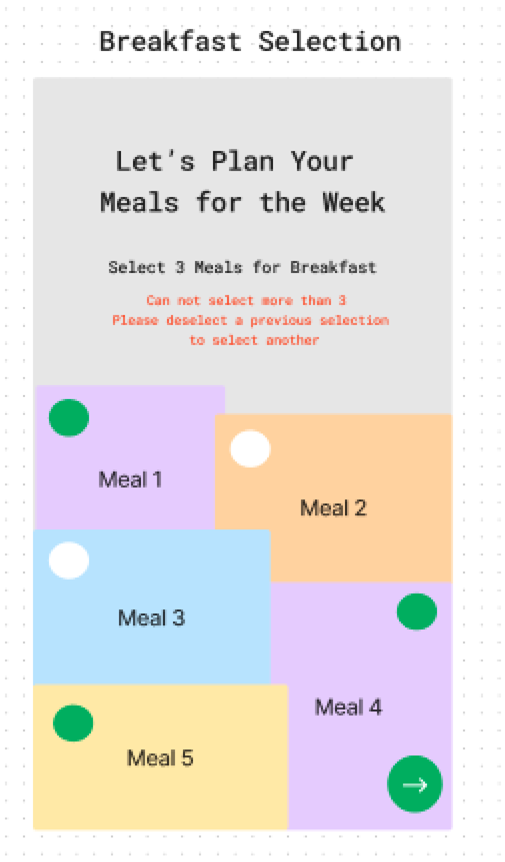
Passwords do not match

Figure 15: Create an Account Error

Description:

When creating an account, users will be given error messages if any of the fields are empty. Additionally, they will be given an error message if the email entered is invalid. Users will be given an error message if the password does not meet the security requirements. Users will be given an error message if the password they entered the second time does not match the first entry.

4.1.2.5 Meal Selection - Error Messages


Breakfast Selection

Let's Plan Your Meals for the Week

Select 3 Meals for Breakfast

Can not select more than 3
Please deselect a previous selection
to select another

Figure 16: Meal Selection Error

Description:

Users will be given an error message if they attempt to select more than 3 meals. Users will be prompted to deselect a currently selected meal in order to select another. Additionally, users will not be able to proceed by clicking on the arrow button until 3 meals are selected. Once three meals are selected, the arrow will become vibrant in color, indicating that the user can proceed.

4.1.2.6 View Previous and Future Meals

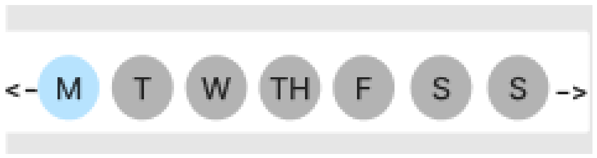


Figure 17: View Previous/Future Meals

Description:

Users can view previous and the upcoming meals for the week. They can scroll to the week they would like to view by clicking on the arrows. The arrow to the left will scroll to previous weeks. The arrow to the right will scroll to the current week if the user is viewing previous weeks. Users can select a specific day of the week to view by clicking on the button that represents that day.

5 ADDITIONAL MATERIAL

3.1 Definitions and Acronyms

- ➢ MBP - Meal Budget Plan
- ➢ SQL - Standard Query Language
- ➢ iOS - iPhone Operating System
- ➢ UI - User Interface

3.2 References

i. Apple HIG:
https://developer.apple.com/design/human-interface-guidelines/designing-for-ios
Modern database management 13th Edition by Jeffrey A. Hoffer et al.