Software Version Description

for

Key2Keto

Version 3.0 approved

Prepared by Mike Bauer, Christian Young, Kenya Holland

White Tiger Inc.

April 30, 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Key2Keto	3/19/2021	Initial creation	0.1.0
Sprint 2	4/9/2021	Adding changes made since sprint 1	0.2.0
Final Sprint	4/30/2021	Adding changes since sprint 2	0.3.0

1. Introduction

1.1 System Overview

This software's purpose is to allow the user to monitor and track various aspects of their health based on a few criteria and to promote an overall healthy well being. The software will help give the user options for recipes based on their chosen criteria as well as populate the various food items that would be needed to cook the recipes. Key2Keto's sole purpose is not to be another health tracker but an application that gives the users ways to an overall healthier well being.

1.2 Version Overview

The current version of the software is a continuation of the previous version. This is the final version of the program which includes more features. The added features to this version of the software is the ShoppingView tab. This tab allows the user to view all the ingredients that make up the recipes that the user has chosen for the week. With those ingredients, the user is able to create a PDF file so that they can print it out and take it to a grocery store. The remaining features of the tab are showing the total macros of the week based on the recipes that have been chosen as well as a dropdown with a list of restaurants that the user can view a meal option from the respective restaurant. The restaurants that populate are based on the user's chosen diet type. This version of the software eliminated a couple of bugs when populating recipes for the week. Finally, saving and loading accounts from text files was added. Upon clicking the Log Out button, and the user confirms they want to log out, their account information will be saved to a text file. When inputting valid information into the login screen, their account will be loaded in, and they can continue from where they left off.

1.3 Team Contributions

Mike Bauer: A recipe pane for adding, deleting, and viewing recipes for each day of the week. This pane is broken up to three separate classes and panes to allow for single responsibilities. Added a class to be able to read in a comma separated text file that represents a list of recipes. Also created the main pane of the program that allows for switching between tabs (Dashboard, Recipes, Account, and more). A Dashboard pane that will display to the user their information of their selected recipes, daily goals, water consumption, hours of sleep. Made layer and package view diagrams. A Shopping pane that will display the chosen recipe's ingredients for each day, a restaurant meal option, as well as the total macros for the day. This Shopping pane also gives the user the option to save the ingredient list as a PDF to be printed at a later date. Created and populated a list of meals that can be chosen as restaurant options.

Christian Young: A login pane to allow the user to login with pre-existing account information, alongside a create account button that leads to a separate pane that allows the user to create an account. This account creation pane has checks against invalid input. Also created a "My Account" pane that shows the current variables for the user's account, and allows the user to change those variables to valid alternatives. Upon invalid input, these views will display error messages detailing what was wrong with the user's input. These three views also have junit and styling done on them,

as well as separation of the view from its logic. Also implemented generating a PDF file from a list of recipes, aka a shopping list, and saving and loading account information from text files.

Kenya Holland: Created recipe text files for three different Keto diet styles (Classic Keto 4+:1, Modified Keto 2-4:1, and Light Keto 1-2:1). They all contain 16-17 recipes each. Assisted in reading in those text files. Implemented scene switching mechanisms following the development of Login, Create Account, and Main panes. Made UML diagrams and demonstration video. Created everything in the Tracker package and assisted in implementing it into dashboardView. Created Junit testing for everything in the Tracker package as well as testing for MainView.java. Separated logic from TackerView and DayView classes. Added images for recipes. Restyled the entire program. Made UML diagrams and added tracking features.

1.4 Document Conventions

The conventions used in describing this document were standard conventions using basic fonts. The parts of the project that are described in the document had level or near-level priorities.

1.5 References

This document does not refer to any outside web addresses or other documents.

2. Inventory of Materials

2.1 Documents Released

- App.java
- MainView.java
- SceneSwitcher.java
- ketoIcon.png
- AccountCreationView.java
- AccountView.java
- LoginView.java
- LoginViewLogic.java
- Account.java
- AccountCreationView.java
- AccountCreationViewLogic.java
- AccountFileReader.java
- AccountSaver.java
- DayOfWeekView.java
- DayOfWeekLogic.java
- Ingredient.java
- Recipe.java
- RecipeFileReader.java
- RecipeList.java
- RecipeView.java
- RecipeViewLogic.java
- RecipeDetailView.java

- ModifiedKeto.txt
- ClassicKeto.txt
- LightKeto.txt
- RestaurantMeals.txt
- List of Recipe Images
- RestaurantFileReader.java
- RestaurantMeals.java
- ShoppingList.java
- ShoppingView.java
- ShoppingViewLogic.java
- Dashboard.java
- DashboardView.java
- DashboardViewLogic.java
- ConfirmPopUp.java
- DayView.java
- DayViewLogic.java
- Tracker.java
- TrackerView.java
- TrackerViewLogic.java
- FileReaderInterface.java
- ShoppingListInterface.java
- ViewInterface.java

2.2 Executable Media Released

No executables are included in this software system at this time.

2.3 Software Projects\Assemblies Released

Key2Keto is developed using a Maven project through Eclipse. The GUI aspect utilizes JavaFX. Developers of the software use either OpenJDK through AdoptOpenJDK 11 or Oracle's JavaJDK 11 with JavaFX 13. For testing views, TestFX version 4 is used. When generating PDF files, iText version 7 is used.

2.4 Test Projects\Scripts Released

There currently are no test scripts or mocks for this version of the software outside of the individual java test classes.

3. Completed Work

3.1 Creating a New Account

3.1.1 Description

When the program starts up, a login screen is shown to the user. If the user has not created an account yet, they can click the Create Account button to be brought to the account

creation form. There, they will input a username, password, their first and last name, choose their sex, input their height, weight, and age, and choose which diet type they wish to follow. These diet types consist of Classic, Light, and Modified Keto. Once the form has been completed, and every input is valid, the user may click the button at the bottom of the form to confirm the creation of their account, and to be brought to the main view of the application.

3.1.2 Completed Functionality

1. The user is able to click the Create Account button to be brought to an account creation form
2. The user is able to enter account details, which consist of a username and password
3. The user is then able to enter personal details, which consist of their first and last name, sex, height, weight, age, and diet type
4. The user is finally able to click a button to create an account, and to be brought to the main application

3.1.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
AccountTest.java	2, 3	100	0
LoginViewTest.java	1	100	0
AccountCreationViewTest.java	2, 3, 4	100	0

3.1.4 Related Acceptance Tests

Scenario: Account creation form is shown to the user
Given the application has been started
And the login page is showing
When the user clicks the Create Account button
Then the account creation form shows up

Scenario: User attempts to create an account with invalid inputs
Given the form has been filled
And some or all of the inputs are invalid
When the user clicks the submit button
Then the account will not be created
And the application will not move on to the main view

Scenario: User creates an account with valid inputs
Given the form has been filled

And all of the inputs are valid
When the user clicks the submit button
Then the account will be created
And the application will move on to the main view

3.2 Viewing and Adding Recipes to Meal Planner

3.2.1 Description

When the user clicks on the Recipe tab from the main set of tabs, they will be prompted to select a day of the week which will populate the lower half of the application. From there, if the user has already created an account, their recipes for that selected day will already show. If they have not created an account already, then the left side will be empty for them to populate by selecting which type of recipe they are trying to pick. A dropdown will populate for the user to add to the recipes for that day. When the user selects a recipe from the dropdown and before adding the recipe, the right side will populate the details of the recipe for the user to look over to determine if they want to add it or not.

3.2.2 Completed Functionality

1. The user is able to click on the Recipe tab
2. The user is then able to pick a day of the week
3. The user is then able to pick a recipe category: Breakfast, Entrees, Snacks
4. The user is able to select a recipe from a dropdown list
5. After selecting the recipe from the dropdown, the user is able to view all the details of the recipe
6. The user is then able to either select a new recipe from the dropdown or add the recipe to the recipes of the day by clicking the Add button
7. The user can also delete a recipe from the day's list of recipes and change them

3.2.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
IngredientTest.java	5	100	0
RecipeListTest.java	4,5,6	100	0
RecipeTest.java	4,5,6	100	0
RecipeViewTest.java	1,2,3,4,5,6,7	100	0

3.2.4 Related Acceptance Tests

Scenario: Viewing recipes on the meal planner

Given: the user has logged in and has an account
When: the user clicks on the Recipe tab
And: the user clicks on a day of the week
And: the user clicks on Overview
Then: the user will see the selected recipes for that day

Scenario: Viewing potential recipes to add to the meal planner
Given: the user has clicked the Recipe tab
When: the user clicks on a day of the week
And: the user clicks on either breakfast, entrees, or snacks
Then: the user will see the dropdown box to pick a recipe

Scenario: Adding potential recipes to add to the meal planner
Given: the user click on either breakfast, entrees, or snacks
When: the user clicks on a recipe from the dropdown box
And: the user clicks on the Add button
Then: the user should see the recipe listed for the type chosen (breakfast, entrees, snacks)

Scenario: Seeing the details of the recipe to potentially add to the planner
Given: the user clicked on either breakfast, entrees, or snacks
When: the user clicks on a recipe from the dropdown box
Then: the user should see the recipe's details in the bottom right of the application.

All test classes listed above were tested for these acceptance tests as they all pertain to a part or all of a Recipe's private member variables. These aspects are then shown in the RecipeDetailView and DayOfWeekView classes.

3.3 Edit Account Details

3.3.1 Description

When the main view is showing in the application, the user is able to click a tab at the top of the window titled "My Account". Upon clicking it, the user will see a list of their account details. Here, they may click the edit button to the side of each account detail to change it to a different valid value. If the user changes a detail, the label showing that detail will update to show the new value.

3.3.2 Completed Functionality

1. The user is able to click on the My Account tab
2. The user is then able to see their account details
3. Finally, the user can then edit any of their account details to a different valid value

3.3.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed

AccountTest.java	3	100	0
AccountViewTest.java	2, 3	100	0

3.3.4 Related Acceptance Tests

Scenario: User goes to My Account page
Given the application is showing the main view
When the user clicks the My Account tab
Then the user's account details will be displayed

Scenario: User clicks edit button for an account detail
Given the application is showing the My Account view
When the user clicks any of the edit buttons beside their respective values
Then a small popup window will appear that allows the user to input a new value

Scenario: User wants to change an account detail to an invalid value
Given the popup window for changing a value is showing
And the user has input an invalid value
When the user clicks the submit button
Then the value is not changed
And the label displaying that value is not changed

Scenario: User wants to change an account detail to a valid value
Given the popup window for changing a value is showing
And the user has input a valid value
When the user clicks the submit button
Then the value is changed
And the label displaying that value is changed to show the new value

3.4 View Dashboard Details

3.4.1 Description

When the main view is showing in the application, the user is able to click a tab at the top of the window titled "Dashboard". Upon clicking it, the user will see all of the details and information that they have selected through the other tabs. They will be able to see the breakdown of their weekly water intake and hours slept. The water and sleep numbers will be populated from their interaction of the Tracker tab. They will also be able to view the goals that they have set for themselves as well the recipes that they have selected for the day from the Recipe tab.

3.4.2 Completed Functionality

1. The user is able to view the daily goals, selected recipes, line charts of daily water consumption and hours slept.
2. The user will be able to check off the daily goal's checkboxes as they complete them.

3.4.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
DashboardTest.java	2	100	0
DashboardViewTest.java	1,2	100	0

3.4.4 Related Acceptance Tests

Scenario: User goes to Dashboard page
Given the application is showing the main view
When the user clicks the Dashboard tab
Then the user's account overview will be displayed showing any selected recipes, set goals, daily water consumption, and daily hours slept.

Scenario: User view the amount of sleep they had on Monday
Given they have added hours of sleep for Monday from the Tracker Tab
When the user clicks the Dashboard tab
Then the user's Dashboard will display the new amount of sleep hours.

Scenario: User views the amount of water they drank on Monday
Given they have added water for Monday from the Tracker Tab
When the user clicks the Dashboard tab
Then the user's Dashboard will display the new amount of sleep hours.

Scenario: User views the selected goals for the day
Given they have added a goal for Monday from the Tracker Tab
When the user clicks the Dashboard tab
Then the user's Dashboard will display the new goal that they created.

3.5 Tracking sleep, water and adding goals

3.5.1 Description

After the user has logged in or made a new account, they will be brought to the main view. This view contains several tabs. Once the user selects the Tracker tab, they will be brought to another screen where they can enter the total amount of sleep/water they had for each day. They may also add however many goals they want. These goals are daily goals and can consist of anything. When the user enters the amount of sleep they had for example, they must click the save button to store that value. Once the save button is clicked, a pop-up window will show and confirm that everything was saved correctly. All of the

information that is entered in this tab can be seen in the dashboard tab. A list of goals and two graphs for sleep and water intake for the week.

3.5.2 Completed Functionality

1. The user is able to click on the Tracker tab
2. The user is able to select a day of the week
3. The user is able to enter the amount of sleep and/or the amount of water they had for the day
4. The user is able to click save, and see a confirmation that the information they entered was saved.
5. The user is able to add one goal at a time, but as many as they want to have for each day individually.
6. The user is able to click the add button and see a confirmation that the information they entered was saved.
7. All of the information entered in this tab is saved in tracker and can be viewed in the dashboard tab

3.5.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
TrackerTest.java	7	100	0
TrackerViewTest.java	1	100	0
DayViewTest.java	2,3,5	100	0
ConfirmPopUpTest.java	4,6	100	0

3.5.4 Related Acceptance Tests

Scenario: Viewing the Tracker tab
Given: the user has an account
And: the user is logged in and the application is showing the main view
When: the user clicks on the Tracker tab
And: the user clicks on a day of the week
Then: the user will see three textboxes to enter the amount of sleep and water they had for the day as well as what goals they want to set for the day.

Scenario: User adds the amount of sleep they had on Monday
Given: The user is on the Tracker tab
And: The user has selected Monday
When: The user enters the amount of sleep they had
And: The user has selected the save button under the sleep text field/spinner
Then: The value in the text field is stored in the Tracker class

Scenario: User adds the amount of water they drank on Monday

Given: The user is on the Tracker tab
And: The user has selected Monday
When: The user enters the amount of water they drank in ounces
And: The user has selected the save button under the water text field/spinner
Then: The value in the text field is stored in the Tracker class

Scenario: User adds a goal for Sunday
Given: The user is on the Tracker tab
And: The user has selected Sunday
When: The user enters a goal they wish to achieve
And: The user has selected the add button under the goals text field
Then: The value in the text field is stored in the Tracker class

3.6 Generating Shopping List

3.6.1 Description

Upon clicking the Shopping tab, the user will be presented with several options, one of them allowing the user to save their saved recipes into a shopping list. This will show a popup window where the user may choose a name and a location for the pdf file.

3.6.2 Completed Functionality

1. A pdf file is generated at the specified name and location with an input recipe list.

3.6.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
ShoppingListTest.java	1	100	0

3.6.4 Related Acceptance Tests

Scenario: User wants to print out a shopping list
Given the user has chosen some recipes
And the user has clicked the Shopping tab
And the user wishes to print out their shopping list
When the user specifies they want to save a shopping list
Then a popup window shows asking for a file name and location

Scenario: User is saving a shopping list file
Given the user has specified they want to save their shopping list
And the save popup window is displaying
When the user inputs a specified name and location for the file
Then the file will be saved with the specified name in the specified location

3.7 View a restaurant meal option

3.7.1 Description

The user might not have the time to cook every single day, so they can view what options they have for eating out at restaurants. These options can be viewed in the Shopping tab of the program on the bottom right of the screen. They click the drop down, select a restaurant they want to eat at, and see what food option is available there for them for the specific diet plan that they are following.

3.7.2 Completed Functionality

1. A user is able to view the Shopping tab
2. A user is able to see a dropdown to click
3. A user is able to click the combobox drop down and view a list of restaurants to choose from
4. A user is able to click a restaurant name and view the meal

3.7.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
ShoppingViewTest.java	1,2,3,4	100	0

3.7.4 Related Acceptance Tests

Scenario: User wants to view a restaurant option
Given the user has chosen some recipes
And the user has clicked the Shopping tab
And the user wishes to view a restaurant meal
When the user selects a restaurant from the list
Then a label is created showing the meal option

3.8 View weekly macros and ingredients based off chosen recipes

3.8.1 Description

Once a user has selected all of the recipes they want to have for the week, they will go to the shopping tab to view weekly macros and ingredients. The ingredients for every recipe they choose for each day will be listed on the left side of the screen. They can also choose to save a PDF of their shopping list to view all of the ingredients they will need. The total macros for each day is listed on the top right corner of the screen. This will allow the user to ensure they are on-track with their Keto diet plan.

3.8.2 Completed Functionality

1. A user is able to view the Shopping tab
2. A user is able to view macros for each day
3. A user is able to view ingredients needed for each recipe

3.8.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
ShoppingViewTest.java	1,2,3	100	0

3.8.4 Related Acceptance Tests

Scenario: User wants to view their macros for the week
Given the user has chosen some recipes
And the user has clicked the Shopping tab
When the user selects the Shopping tab
Then 4 columns are shown listing all the ingredients from the chosen recipes

3.9 Logout and log back into the software with a stored user

3.9.1 Description

The user may log out of their account at any time. All of their saved recipes, tracking information, personal information, and etc. will be saved to a .txt file for logging back in. When the user presses log-out, they will be brought back to the login page. They can enter the login information they entered previously and their entire account will be loaded back into the program.

3.9.2 Completed Functionality

1. A .txt file is created when the user selects the log-out button.
2. The information is read in when the user enters the login info for that users .txt file.

3.9.3 Related Unit Test Cases

Class Name	Functionalities Tested	% Passed	% Failed
LoginViewTest.java	1,2	100	0
AccountSaverTest.java	1	100	0
AccountFileReaderTest.java	2	100	0

3.9.4 Related Acceptance Tests

Scenario: User wants to logout and log back into the program
Given the user has created an account
When the user selects the Log out tab
Then the user's account will be saved for later use.

Scenario: User wants to log back into the program
Given the user had created an account previously
When the user enters their username and password on the LoginView

Then the user's account will be logged in.

4. Current Design

4.1 Description and Evaluation of Current Architecture

This majority of the current architecture of the software uses concrete java classes. We have added three interfaces to assist with decreasing dependencies. The program also has static logic classes for all GUI aspects of the software. The interfaces that have been implemented are also mainly for the GUI with two for file reading. The ViewInterface class has methods inside it that all of the views share such as methods for populating the node's children, initializing elements, styling the elements, and a method that houses the setOnAction events that the GUIs will have.

An example of the GUI tabs is the Recipe tab of the main part of the application. There are standard java objects in the RecipeView class such as Recipe and Ingredient. These objects do not need to inherit or extend from each other. The next parts that make up the Recipe tab would be the GUI views which will utilize the ViewInterface. Each one of those views extends the JavaFX object Pane. They extend Pane so that we are able to utilize them in a StackPane object. We have used the Single Responsibility Principle throughout the application's files as well.

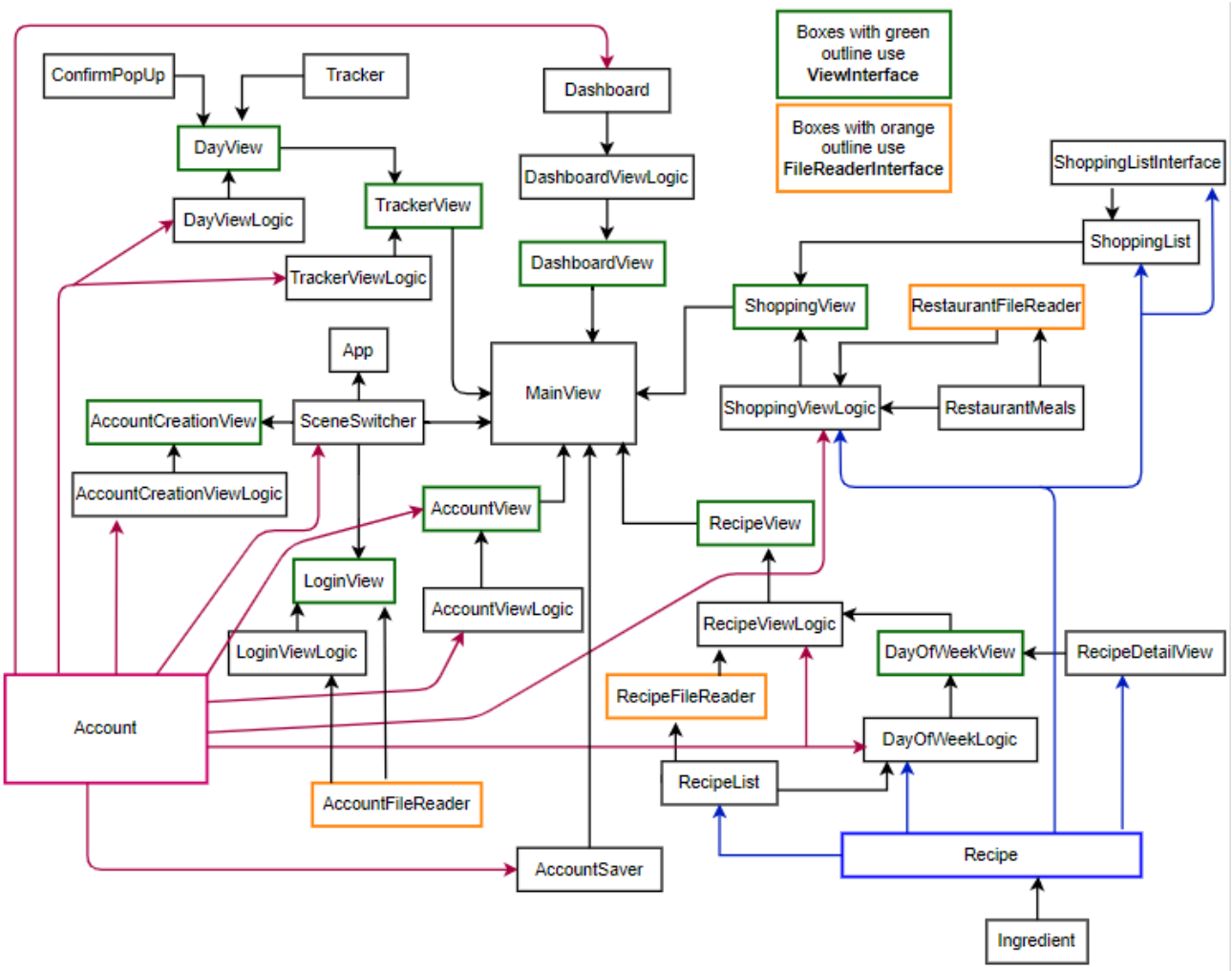
For file reading, the software uses a common FileReaderInterface interface class that has a single method readFile(). This method will be defined differently by each of the classes using it and reads a comma separated text file. These classe are RecipeFileReader, which reads in the text file that populates all the Recipe objects in the program. Another class is the RestaurantFileReader. This class reads a text file that populates the restaurants that can be used for a Restaurant Option in the Shopping tab.

We have added a save and load function for the users' accounts which we use a similar format of reading from a text file. The account reading is done using the AccountFileReader class which will use the same interface that the RecipeFileReader and RestaurantFileReader utilize.

4.2 Major Software Interfaces

There are three interfaces that are implemented in this software: ShoppingListInterface, FileReaderInterface, and ViewInterface. The ShoppingListInterface is simply being used for unit testing. The ViewInterface is used for all the GUI classes with the common methods being part of the interface. Lastly is the FileReaderInterface. This interface has a single function readFile which is used to read and populate the objects that are needed for that specific interaction of the class.

4.3 Evaluation of Classes



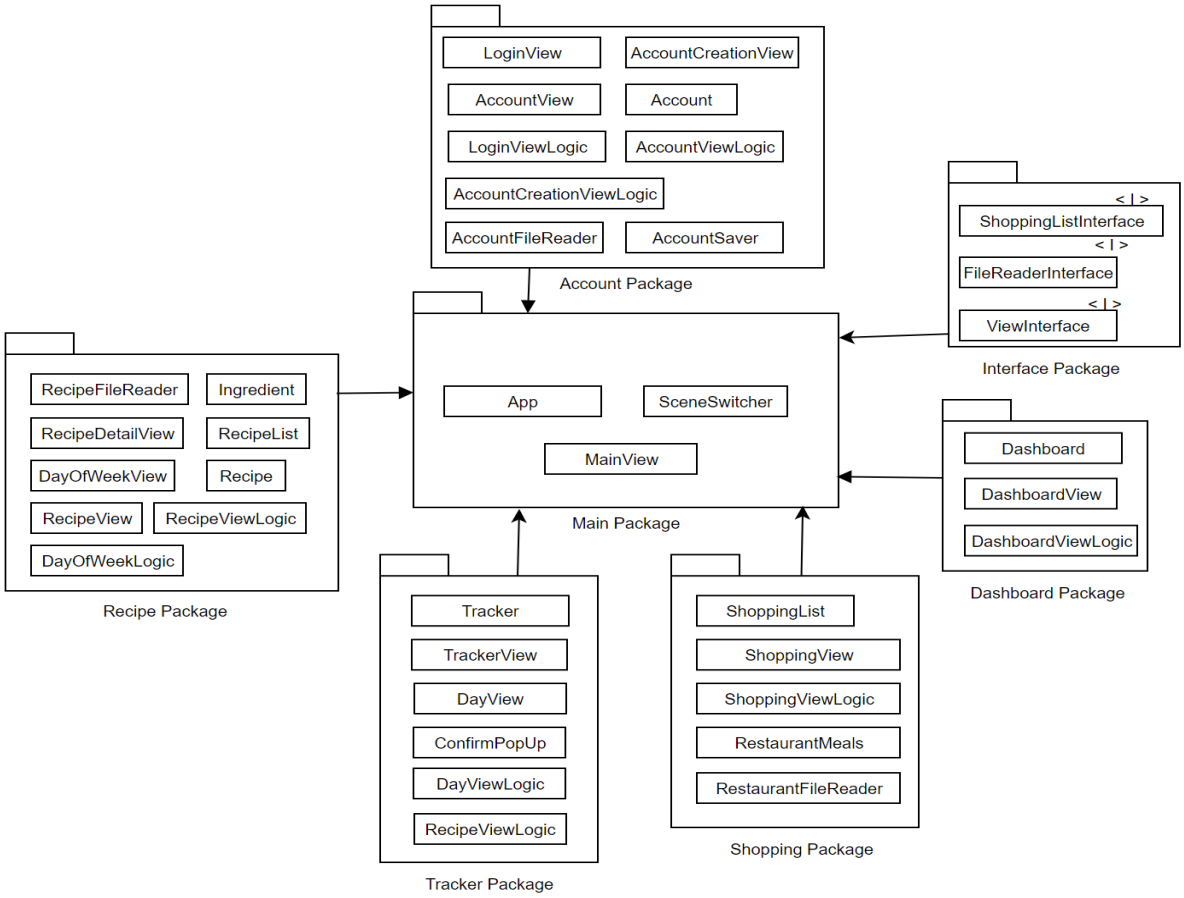
Link to all UML diagrams with variables and functions will be in Appendix B

MainView: This class refers to the scene that you see directly after logging in or creating a new account. This scene contains the Key2Keto title as well as the tabs: Dashboard, Account, Shopping, Recipes, and Logout.
SceneSwitcher: This class's sole purpose is to assist in switching scenes. It creates the necessary layout panes and allows for classes to use the setScene command.
LoginView: The first scene that shows when the program is run. It prompts the user for a login or, if the user does not already have an account, they have the option to create a new account. This login will determine what information needs to be loaded into the program. Such as diet type, and later on, the users saved meals and water/sleep tracking.
LoginViewLogic:This class uses static methods used for the logic for the LoginView. This acts as a controller for the class.
AccountCreationView: Handles new user info. Prompts user for username, new password, name, age, weight, height, sex, and diet type. This info will be stored by Account class as a part of a users save file and must be used to login later on.
AccountCreationViewLogic:

AccountView: This scene regurgitates the info that was entered in the AccountCreationView by the user. Any information can be edited. A user can change their password, their preferred diet, weight, and so on.
AccountViewLogic: This class uses static methods used for the logic for the AccountView. This acts as a controller for the class.
Account: This class stores the information that was entered in the AccountCreationView class. Only consists of getters and setters of variables.
DashboardView: The class shows all the information that the user has placed on the account. This class uses the Account, Recipe, and Tracker classes to populate chosen recipes, number of hours slept, amount of water consumed, and goals set for each day of the week.
DashboardViewLogic:This class uses static methods used for the logic for the DashboardView. This acts as a controller for the class.
Dashboard: This class stores the information that is entered in the DashboardView class.
RecipeView: This is the scene that shows when a user selects the recipe tab. It mainly stores a tree of other scenes and classes. Once the recipe tab is selected, the user will see all 7 days of the week, which refers to the DayOfWeekView class.
RecipeViewLogic: This class uses static methods used for the logic for the RecipeView. This acts as a controller for the class.
DayOfWeekView: This scene handles meal planning. It shows the planned meals for the selected day and stores the planned meals for all 7 days. Also refers to RecipeDetailView on the bottom right corner when a recipe is selected from the drop down.
DayOfWeekLogic:This class uses static methods used for the logic for the DayOfWeekView. This acts as a controller for the class.
RecipeDetailView: Neatly formats the recipes that are stored in RecipeList class.
RecipeFileReader: Reads in the text file that contains comma separated recipes. Recipes include the title, keto ratio, calories, fats in grams, protein in grams, carbs in grams, ingredients, and instructions.
RecipeList: Stores all of the recipes that were read in from a single text file. Separating the information into appropriate variables.
Recipe: Stores a single recipe
Ingredient: Stores a single ingredient and the amount needed. Eg: ¼ cup flour
Tracker: Stores hoursOfSleep, waterIntake, and list of added goals.
TrackerView: Creates Monday-Sunday tab buttons and creates a DayView for each tab.
TrackerViewLogic:This class uses static methods used for the logic for the TrackerView. This acts as a controller for the class.
DayView: Displays 2 spinners and a textfield, along with 3 buttons to save information. Information entered by the user is sent to the tracker for storage.
DayViewLogic:This class uses static methods used for the logic for the DayView. This acts as a controller for the class.
ConfirmPopUp: meant to let the user know that the information they entered was registered, after a button is pushed.
ShoppingList: Takes in a list of ingredients, writes to a pdf document, and allows the user to select a file name and location.
ShoppingListForTests: Basically ShoppingList, but without the save dialog. Used to avoid errors with ShoppingListTest.
ShoppingListInterface: Both ShoppingList and ShoppingListForTests implement this, as they are very similar with only one minor difference.
ShoppingView: Displays the weekly macros based on the user's selected recipes as well as all the ingredients from those recipes which can be saved to a PDF for the user to print out. Also shows a ComboBox which is populated with restaurants that have a meal they can choose based on the user's diet type.

ShoppingViewLogic:This class uses static methods used for the logic for the ShoppingView. This acts as a controller for the class.

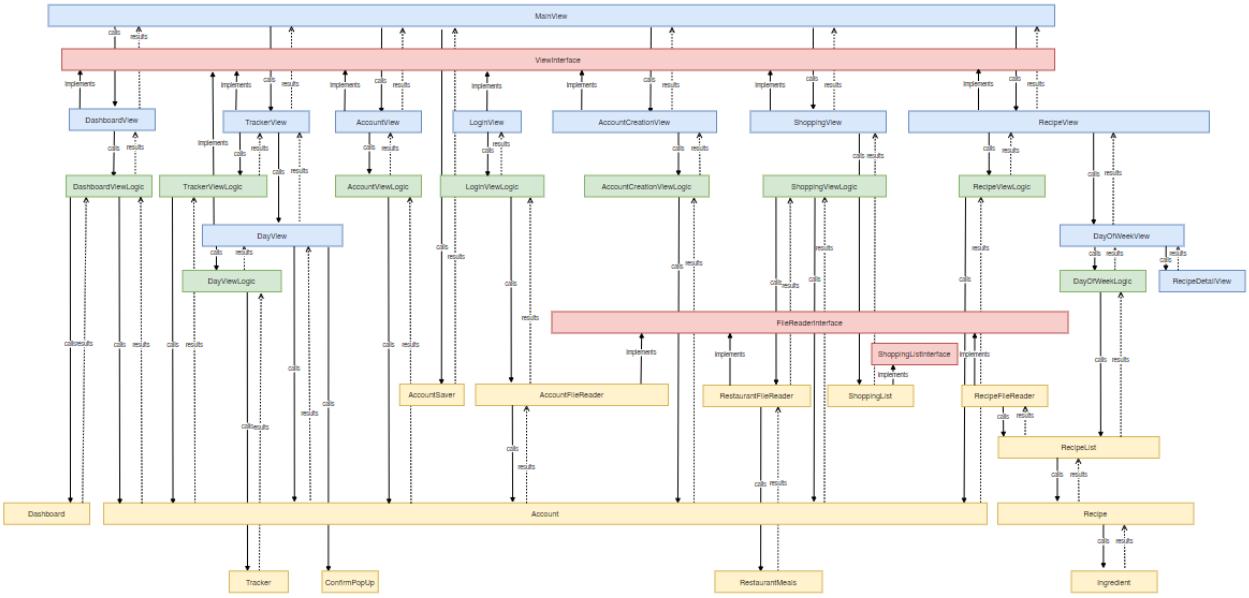
4.4 Current Software Components



Package	Responsibility
Main	Contains Scene and Switcher that allows for a place for all views to be shown and switched
Recipe	Contains all classes and scenes that have to do with displaying, storing, or reading recipes.
Account	Contains all classes that have to do with login, account creation, or the display of a users information
Dashboard	Contains scenes that display a user's overview.
Shopping	Contains classes that pertain to the shopping section of the program, like shopping lists and instructions for navigating grocery stores and restaurant menus.

Tracker	Contains classes pertaining to tracking the user's water and sleep, and their daily goals.
Interfaces	Contains interface classes utilized by other concrete classes.

4.5 Current Software Layers



Class/Classes	Layer	Responsibility
MainView	1	UI class. Contains all scenes and allows for smooth transition between them.
ViewInterface	2	Interface for all View classes
RecipeView, AccountCreationView, AccountView, TrackerView, DashboardView, ShoppingView, LoginView	3	UI classes. This layer expands on individual scenes.
DashboardViewLogic, TrackerViewLogic, AccountViewLogic, LogicViewLogic, ShoppingViewLogic, RecipeViewLogic	4	Controller classes. Contain all information that does not pertain to components that are physically visible.

DayOfWeekView, RecipeDetailView, DayView	5	UI classes. Views that are contained within the second layer's views that show more precise information.
DayViewLogic, DayOfWeekLogic	6	Controller classes. Contain all information that does not pertain to components that are physically visible.
FileReaderInterface, ShoppingListInterface	7 - 8	Interfaces. FileReaderInterface helps those classes that involve reading from files. ShoppingListInterface only pertains to the ShoppingList Class
AccountSaver, AccountFileReader, RestaurantFileReader, ShoppingList, RecipeFileReader	9	Entity classes. AccountSaver connects to MainView. FileReader interface is implemented in AccountFileReader, RestaurantFileReader, and RecipeFileReader. ShoppingListInterface is implemented in ShoppingList.
RecipeList	10	Entity classes. Boundary between RecipeFileReader and Recipe. Contains all recipes that are read in from RecipeFileReader.
Dashboard, Account, Recipe, Tracker, ConfirmPopUp, RestaurantMeals, Ingredient	11 - 12	Entity classes. Concrete classes.

4.6 Incorporating Design Patterns

The Single Responsibility Principle is utilized in this software design. In this iteration, there are three interfaces, each with different responsibilities, a good implementation of the Interface Separation Principle. Regarding Liskov, the various views in our program are treated as a Pane in various situations.

4.7 Data Schema

The data schema that this iteration of the software utilizes is a text file. These files are a comma separated list that has been predetermined what part of the recipe it is associated with. There are currently three separate files that have 15 recipes minimum. The data used for reading in the meals from a file and for reading in a stored user account is done through a comma separated text file as well.

4.8 Violations to Design Principles

There is no Dependency Inversion used throughout the project. Interfaces that are used are implemented through the principle of Interface Separation. If our MainView contained an ArrayList of view interfaces, there could be easier for expansion. With an arraylist of View interfaces there could also be Dependency Injection utilized throughout the project.

5. Remaining Work

5.1 Remaining Software Features

- All features that have been planned to be implemented are implemented.

5.2 Known Issues and Bugs

- When loading an account, the Tracker view sleep and water, and the DayOfWeekView chosen recipes do not show the current values that were present at Logout.

Appendix A: Glossary

GUI - Graphical User Interface
PDF - Portable Document Format

Appendix B: GitHub and Other Information

Current features being used are the basic features for GitHub

GitHub Link -
https://github.com/cen3032-group-projects/semester-projects-cen3032-mw9-30-10-45-group-2

Link to UML diagrams - (Shared with beddy@uwf.edu)
https://docs.google.com/document/d/1pbYqAwuWsaWSUKa_PCXJng7r5BtmwE2vEyWzxmaWjCM/edit

Sprint 1 Demonstration Video -
https://www.youtube.com/watch?v=8NijZGmVBSI

Sprint 2 Demonstration Video -
https://youtu.be/TUhj62QCPCw

Sprint 3 Demonstration Video -
https://youtu.be/71bzBVCr0Nw

Trello Board -
Invite link- https://trello.com/invite/b/T2tLPWif/9cfe42b9ccbc20ba35d578f18ae04739/key2keto

Link - https://trello.com/b/T2tLPWif/key2keto