COMP5541 - Design Documentation

Amirali Ashraf, Bo Jin, Nicholas LaMothe, Kyle Taylor Lange, Pavlo Ruban

January 2020 - April 2020

1 Introduction

The main purpose of the project is to develop a dietary manager application based on requirements provided by Dr. Nora Houari, further referred as the Client. The original requirements have been modified and extra requirements have been added.

1.1 Purpose

The Design Document is a requirement of the Project initiated by the Client. This design document consists of Architectural Design (AD) and Interface Specifications (IS). While the AD provides a subsystem and module diagram, IS describes each service provided by each module.

1.2 Scope

This document is composed during the second increment of the Project and it builds upon the requirement document delivered at the end of the increment one and it will become the foundation for the implementation phase at a further stage. The software architecture is provided in detail for the development team to implement the design into the actual application. The screenshots from the prototype will be used to provide the Client with examples of the Graphical User Interface (GUI). Class diagrams have been constructed using an online UML tool Lucidchart and this document has been created using an online LaTex editor OverLeaf.

1.3 Definitions and Abbreviations

1.3.1 Definitions

Term	Definition
Food	Any abstract food item, either at its purest form of an ingredient or a composition of ingredients that form a single dish. For example, Pasta and Chicken Pasta are separate Food items. Food is a representation of one serving with corresponding attributes and their quantification.
Food Group	Collection of foods that have similar nutritional properties in common.
Unit	Refers to the units of measurement for foods, i.e. g, ml, cup etc.
Meal	Represents an actual food item that has been scheduled for consumption at a certain time, at a certain location and is quantified by the number of servings. Eventually the meal may or may not be consumed.
Meal Type	A verbal representation of a meal depending on the time of consumption, e.g. breakfast, lunch, dinner etc.
Location	Place where Meal is consumed.
Model View Controller	A software design pattern that consists of three separate modules: model, view and controller. Each module is developed separately and it is responsible for tasks that are disjoint with the tasks of other modules.

1.3.2 Abbreviations

Abbreviation	Term
MVC	Model View Controller
SE	Software Engineering
DAO	Data Access Object
DTO	Data Transfer Object

1.4 Overview

In the following sections the document will contain the Architectural Design, Interface Specification and Detailed Class Diagram. The Architectural Design includes rationale for design, System Topology, Software Architecture, Object Model, Dynamic Model and Functional Model. Interface Specification describes each service provided by modules. Detailed Class Design will add data and functions description on top of Interface Specification.

2 Architecture Design

2.1 Rationale

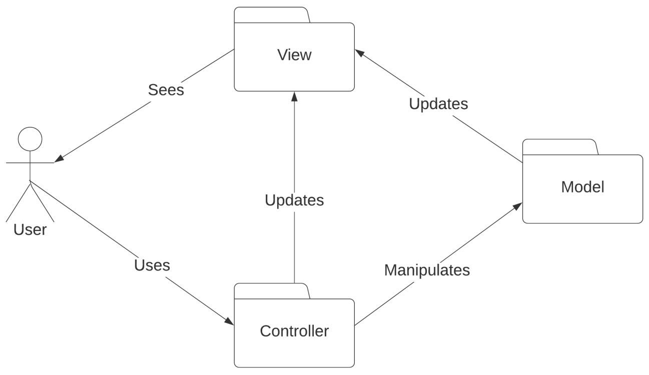
MVC is the architecture that has been chosen for the project for several reasons. First and foremost, it is a requirement imposed by the client and therefore has to be expected. Second, this architecture allows separation of the the project into three independent modules that can be developed separately by different team members. Within MVC architecture it also is easy to achieve high cohesion, when the modules incorporate all functionality that is relevant to it, and low coupling, as the three modules are independent from each other and the communication is achieved through carefully assigned interfaces.

The three modules are responsible for the following tasks. Model is the representation of the data and all the manipulations on it. View is the visual representation of the program and it is realized in the form of GUI. Controller is a link between the two and it serves to track modification in the Model to represent them in the View. Therefore the user has no direct access to the Model. This communication is done via Controller and the changes are automatically updated in the View.

This separation of models allows concurrent development of the modules and this modularity is helpful for debugging and testing.

2.2 Software Architecture Diagram

The diagram below shows the high level MVC diagram and interaction between the actor and the modules. User initiates interaction with the system via the Controller that handles inputs done with a mouse and/or a keyboard. The Controller also provides user input validation. The Controller, based on user's input, manipulates the Model and updates the View. The Model is not directly accessible by the user. The View is realized as GUI and it can be updated by the Controller that accepts user input.



Class Diagram of MVC Architecture

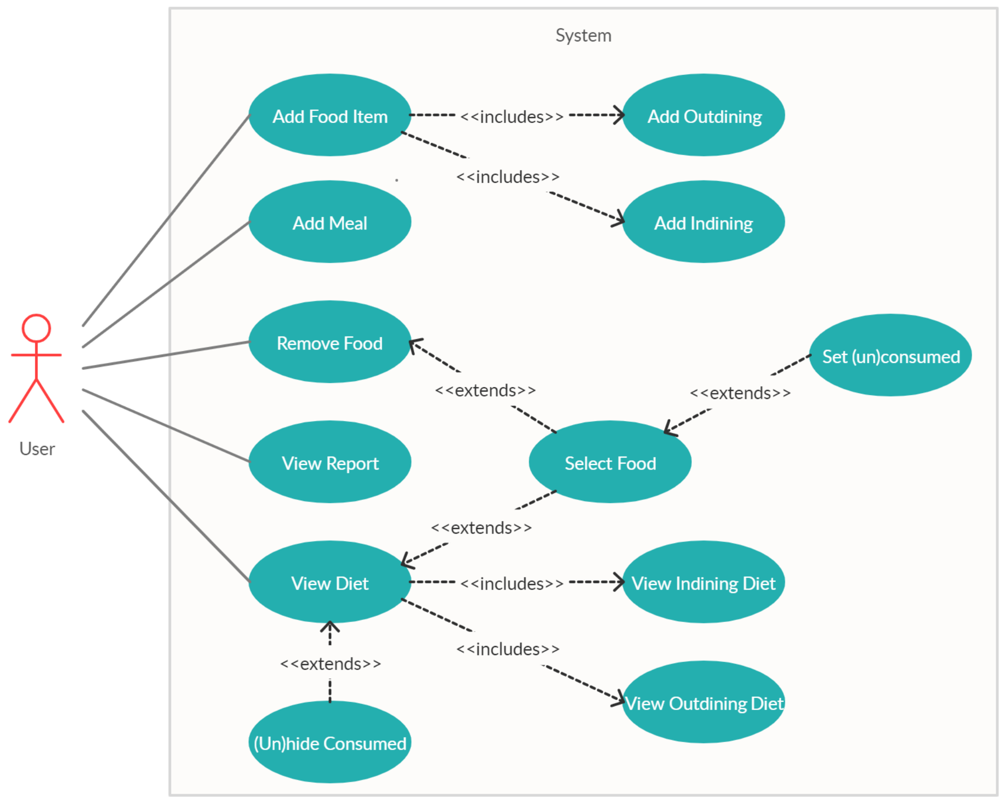
2.3 System Topology

The system is independent of any third-party software. All dependencies will be included in the installation package. The Dietary Application will be a stand alone application and will have to be installed on each machine to run.

3 Structural Design

3.1 Use Cases

The new updated set of Use Cases is presented in his section. These Use Cases can be traced to both the Software Requirement Document delivered with the first increment and additional scenarios from the project requirement for the increment I and II.



Unified Use Case Diagram

In the following section the main Use Cases will be described in more detail.

3.1.1 Add any Indining or Outdining diet to the diet list.

The user is able to add a new Food item and indicate whether this meal is consumed at Location Home (Indining) or elsewhere (Outdining). The action of adding a new Food Item will be performed by the user via USER PANEL.

3.1.2 Delete a food item.

The user will be able to remove a Food item by first searching for the required item and thn performing the action of deleting it. This action will be carried out in User Panel.

3.1.3 View diet with their hierarchy in case of composite element (i.e. Serving, Time, and food Group.

This scenario is performed by viewing the diet shown on two panels: LIST OF CONSUMED FOOD and REPORT PANEL. The user can find all information that pertain to the diet they are interested in, provided they apply the filters to narrow their preferred diet.

3.1.4 Hiding / unhiding consumed diet: where the user has the option to change the current view and to hide/unhide consumed diet (at the root level).

The user can hide or unhide both consumed or not consumed food that will be refected in both the LIST OF CONSUMED FOOD and REPORT PANEL.

3.1.5 Mark a food group as eaten or not eaten

This scenario is automatically considered by the system. When the user modifies their diet for a certain date, the system will present them with a report on what Food Groups have been consumed or not during the day. The system will also allow the user to see the intake of calories and main nutrients.

3.1.6 Set food item as consumed or unconsumed.

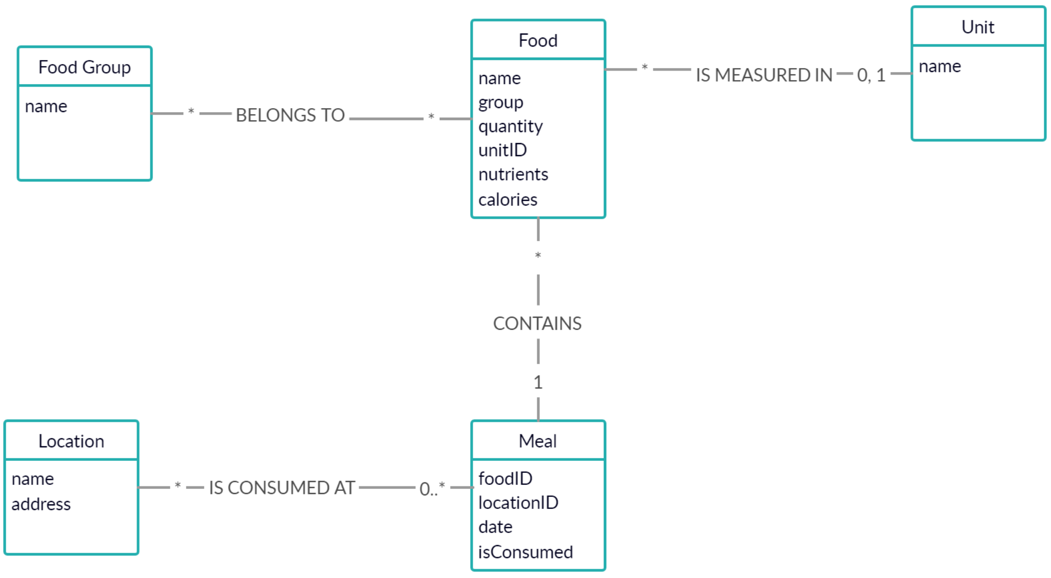
The user will be able to set the food as consumed or not by first viewing the Diet on the LIST OF CONSUMED FOOD. Then the user has to select the Food they wish to modify and finally set it as consumed or not consumed.

3.1.7 View all the consumed food list (in arbitrary order).

The requirement of the increment I that is included in the Use Cases above.

3.2 Domain Model

Based on the Use Cases presented in the previous section main concepts can be extracted to form a Domain Model.

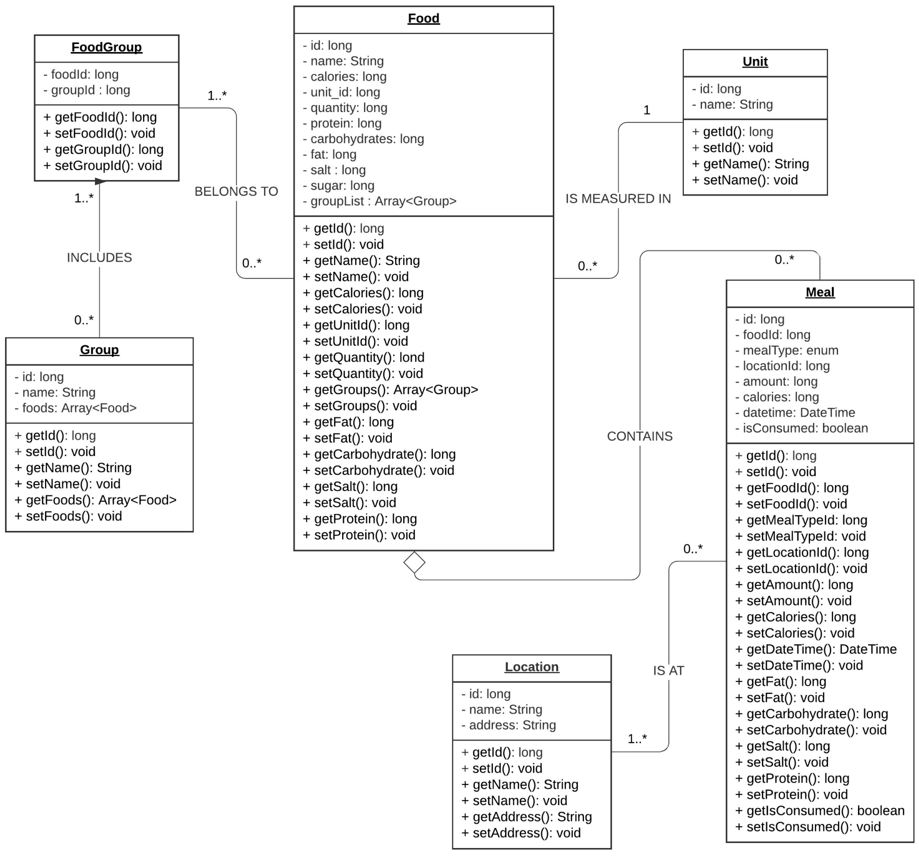


Domain Model

The Domain Model represents a general concept for the future class diagram. At this point the Model represents the requirements covered by the Use Cases. Further development of the Model in the real Class Diagram and an Object Diagram at a specific time instance will be presented further.

3.3 Class Diagram

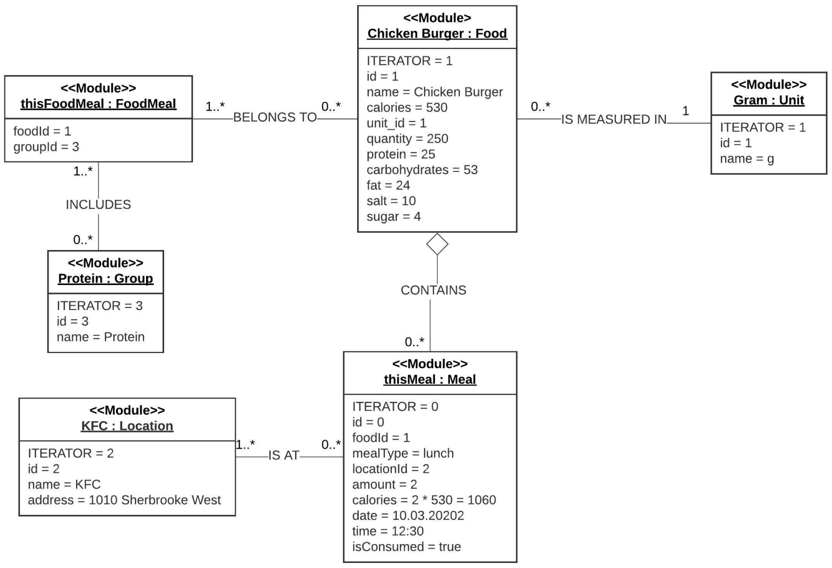
The Class diagram illustrated below is a representation for the Model module of the MVC Architecture and it comprises the logic of the database schema that will be implemented during the increment III. The Class Diagram also includes class descriptions in terms of the attributes, methods and it also contains relations between classes and the verbal explanation.



Class Diagram

3.4 Object Diagram

The following Object Diagram describes a state of the system with one instance of each class in the model. The graph illustrates the objects, attributes and their relationships with other objects.

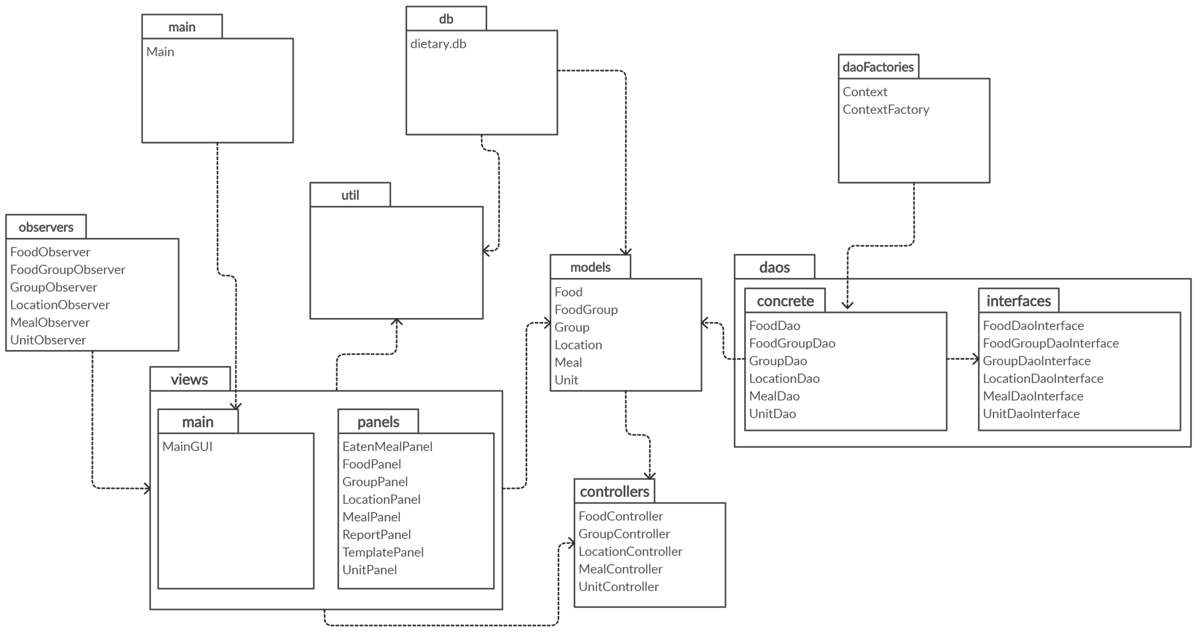


Object Diagram

3.5 Package Diagram

The Package diagram shows packages of of the project with dependencies. It is clear from the graph that the Model depends on the Controller, while the View depends on the Model, which is characteristic to the MVC

Architecture. The dao package with the corresponding classes has been introduced to make a smooth transition to databases in the next increment. The util package provides external libraries that bring valuable features to working with databases and creating a responsive GUI.

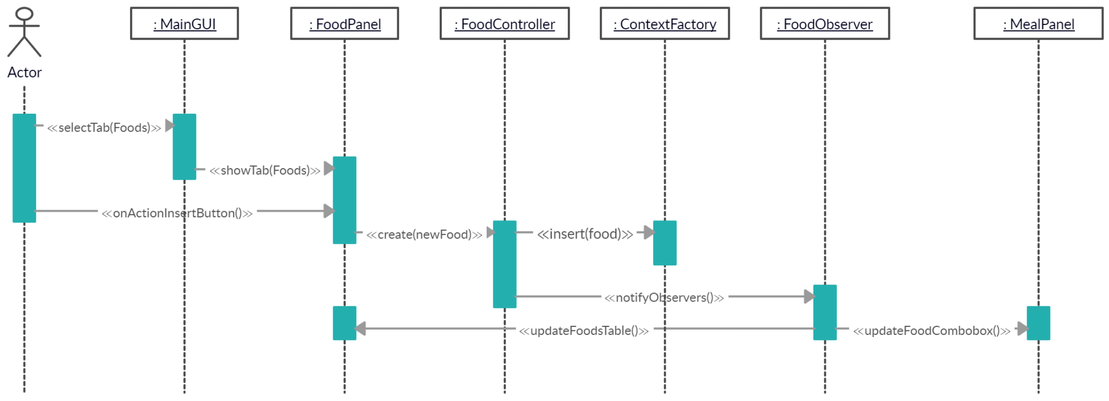


Package Diagram

3.6 Sequence Diagrams

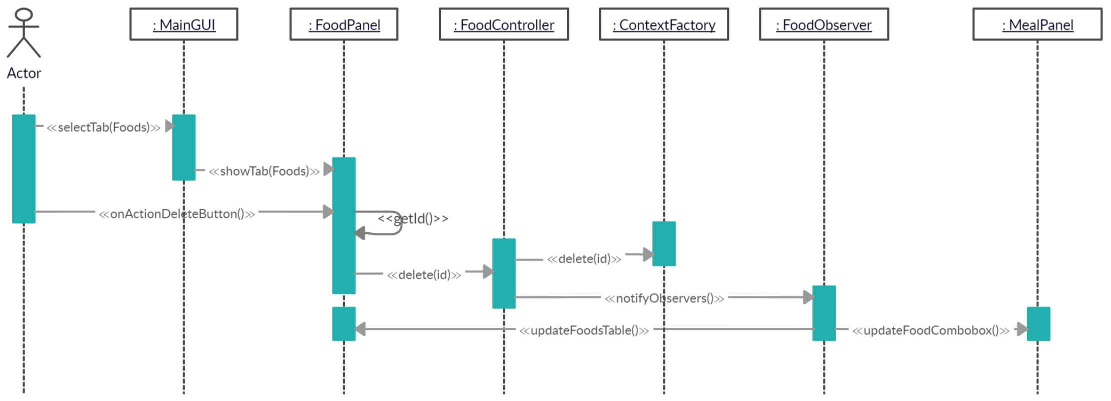
Based on the Use Cases, the following sequence diagrams have been constructed to present a behavioral model of the system.

3.6.1 Add a new Food Item.



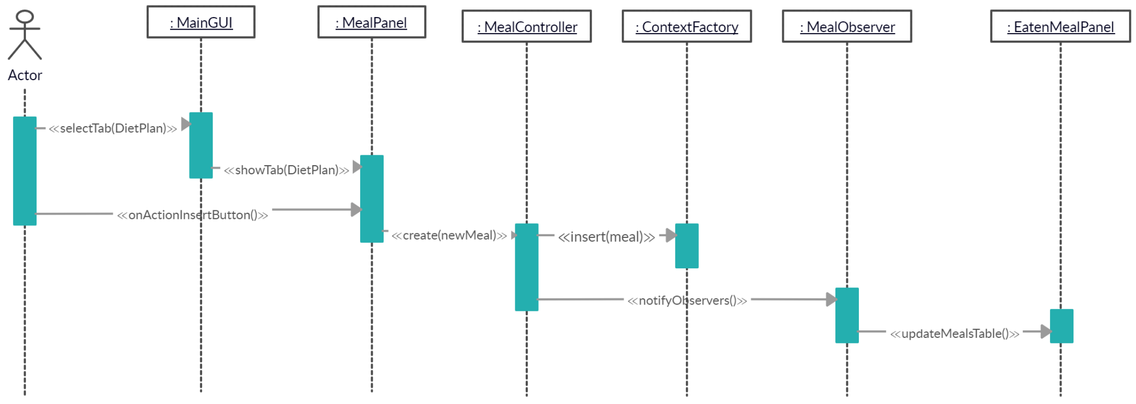
Add New Food Sequence Diagram

3.6.2 Delete Food Item



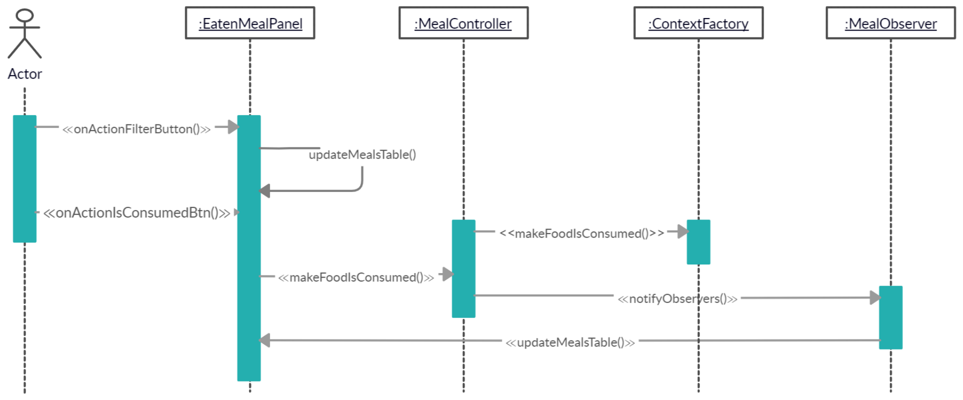
Delete Food Sequence Diagram

3.6.3 Add any Indining or Outdining diet to the diet list



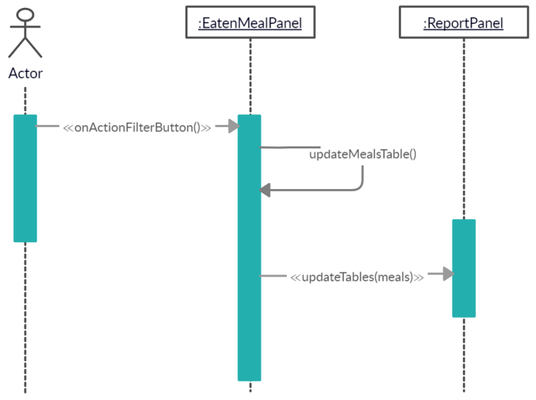
Add Diet Sequence Diagram

3.6.4 Set food item as consumed or unconsumed



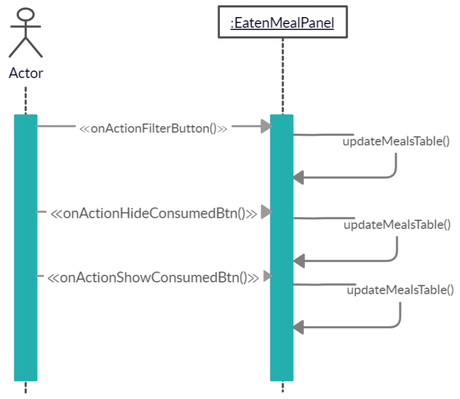
Set Food Consumed or Unconsumed Sequence Diagram

3.6.5 View diet with their hierarchy in case of composite element



View Diet Sequence Diagram

3.6.6 Hiding / unhiding consumed diet



Hiding / unhiding Consumed Diet Sequence Diagram

4 Interface Specification

4.1 System Interface Diagrams

The dietary application utilizes GUI as the only mean of interaction between the user and the system. the user can add, delete or view data via the GUI.

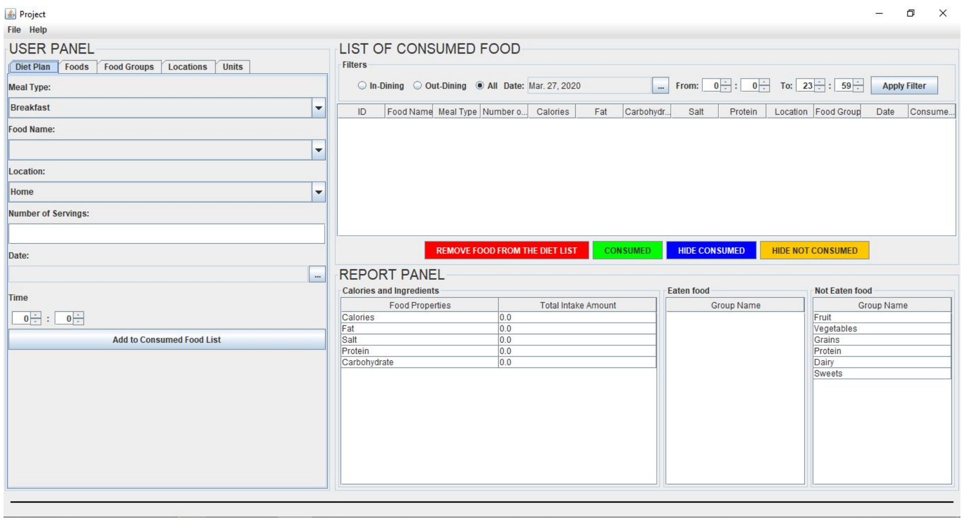
4.1.1 User Interface

When developing the user interface the following software engineering qualities are to be considered:

- Correctness: the UI has to correspond to the software requirements and it should behave in accordance with the specifications
- Friendliness: the UI should be easily understandable, simple to navigate through and use
- Maintainable: the UI should be easy to correct, adapt to a new environment and allow performance improvement

4.1.1.1 Main Window

When the user launches the application they are presented with the Main Window. This window can be divided into three major areas: USER PANEL, LIST OF CONSUMED FOOD and REPORT PANEL. Each area is responsible for specific tasks that are in line with the software requirements.



Meal Type: Breakfast

Food Name:

Location: Home

Number of Servings:

Date:

Time 0 : 0

○ In-Dining ○ Out-Dining [● All] Date: Mar. 27, 2020 From: 0 : 0 To: 23 : 59

ID	Food Name	Meal Type	Number o...	Calories	Fat	Carbohydr...	Salt	Protein	Location	Food Group	Date	Consume...

Calories and Ingredients

Food Properties	Total Intake Amount
Calories	0.0
Fat	0.0
Salt	0.0
Protein	0.0
Carbohydrate	0.0

Eaten food

Group Name

Not Eaten food

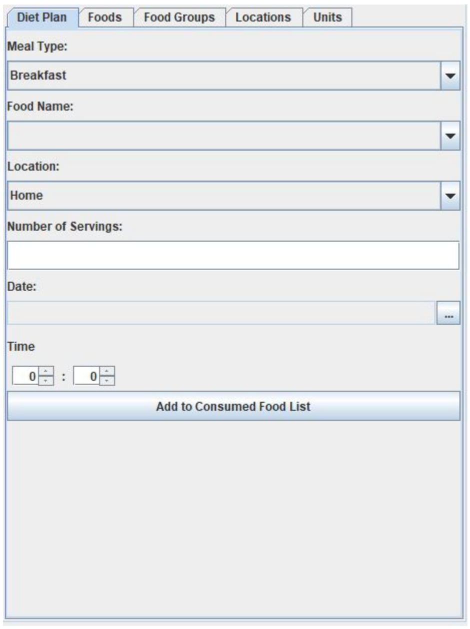
Group Name
Fruit
Vegetables
Grains
Protein
Dairy
Sweets

Main Window GUI

4.1.1.2 USER PANEL

To allow the user to add, modify and delete data the USER PANEL consists of five tabs that the user can navigate through based on what data they intend to add or delete.

- **Diet Plan Tab.** Here the user is presented with a form that they need to fill in to add a meal to the consumed food list.



Meal Type: Breakfast

Food Name:

Location: Home

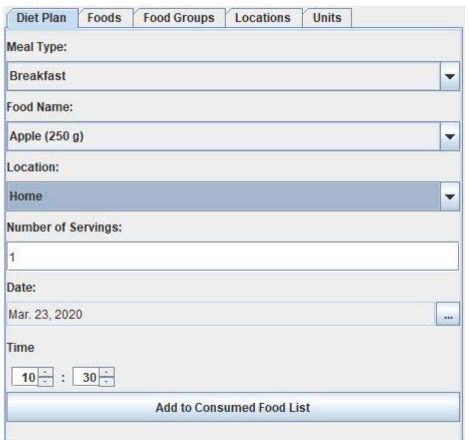
Number of Servings:

Date:

Time 0 : 0

Diet Plan Tab

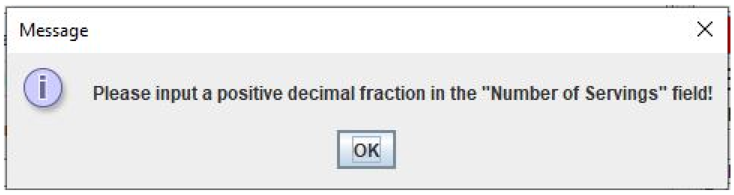
The user has to choose the meal time from the drop-down list of meal types. Next from a drop-down list of foods they choose a food that will be part of the meal. Note that after each food the standard serving size is indicate for the user to consider when they will enter the number of servings of the said food they will be consuming. Following this, the user indicates the location of their meal. Finally, the date and time need to be set. Add to Consumed Food List Button has to be clicked to finalize the action of adding a new meal.



Meal Type: Breakfast
Food Name: Apple (250 g)
Location: Home
Number of Servings: 1
Date: Mar. 23, 2020
Time 10 : 30

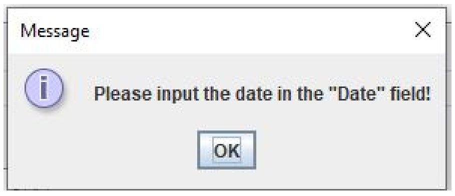
Diet Plan Tab with data filled in

Data entry validation has been put in place to avoid errant data. In such cases the user is notified with a message that describes the invalid input and hints for correct data entry.



Please input a positive decimal fraction in the "Number of Servings" field!

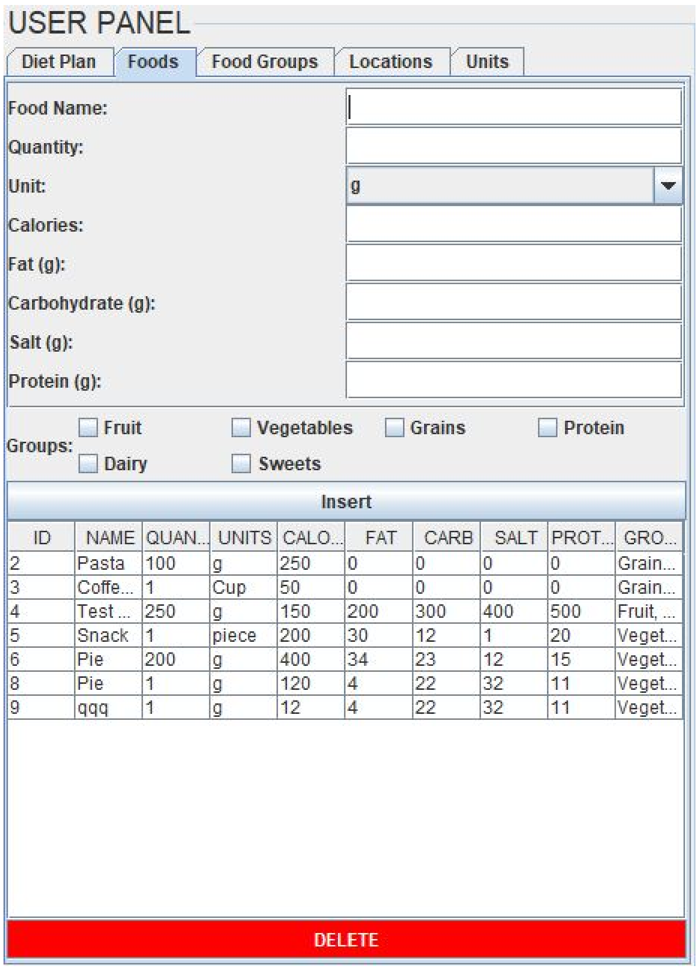
Message notifying user of a wrong serving input



Please input the date in the "Date" field!

Message notifying user of a wrong date input

- **Foods Tab.** This tab allows the user to add new foods and fill in their nutritional facts. To do so the user will have to enter the name of food, the quantity per serving, the amount of calories, fat, carbohydrates, salt and protein. Then the user clicks the Insert Button. To delete a food item the user will have to select the item from the table and click DELETE Button at the bottom.



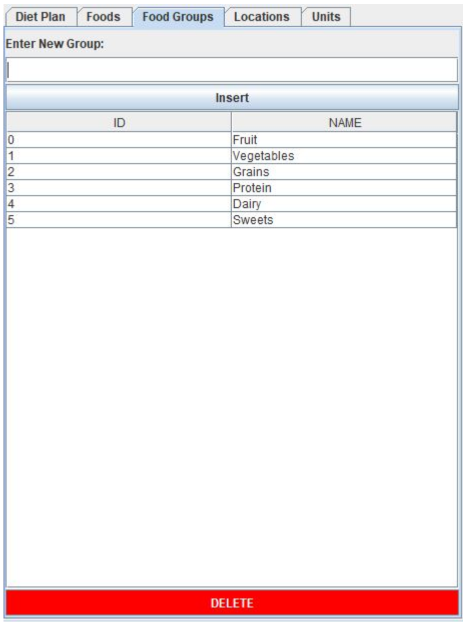
Food Name:	
Quantity:	
Unit:	g
Calories:	
Fat (g):	
Carbohydrate (g):	
Salt (g):	
Protein (g):	

Groups: Fruit, Vegetables, Grains, Protein, Dairy, Sweets

ID	NAME	QUAN...	UNITS	CALO...	FAT	CARB	SALT	PROT...	GRO...
2	Pasta	100	g	250	0	0	0	0	Grain...
3	Coffe...	1	Cup	50	0	0	0	0	Grain...
4	Test ...	250	g	150	200	300	400	500	Fruit, ...
5	Snack	1	piece	200	30	12	1	20	Veget...
6	Pie	200	g	400	34	23	12	15	Veget...
8	Pie	1	g	120	4	22	32	11	Veget...
9	qqq	1	g	12	4	22	32	11	Veget...

Foods Tab

- **Foods Group Tab.** User will be able to add and delete food groups. To add a new food group the user has to fill in the text field at the top and press the Insert Button. To delete a food group the user will have to select the group from the table and click DELETE Button at the bottom.

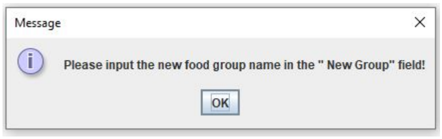


Enter New Group:

ID	NAME
0	Fruit
1	Vegetables
2	Grains
3	Protein
4	Dairy
5	Sweets

Food Group Tab

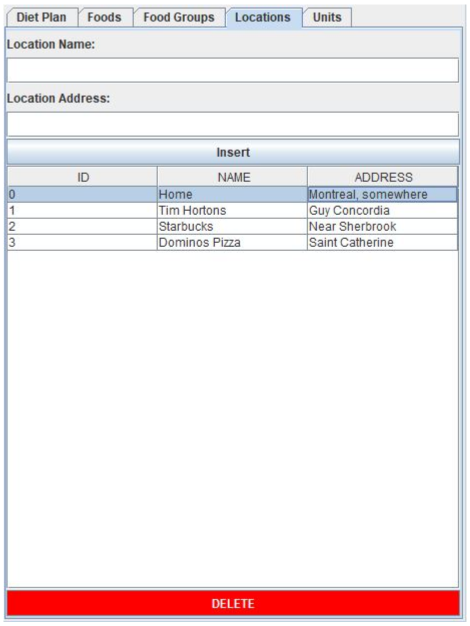
Data entry validation prevents the user from inserting a new group without filling the text field.


Message

Please input the new food group name in the " New Group" field!

Message notifying user of an empty Food Group name field

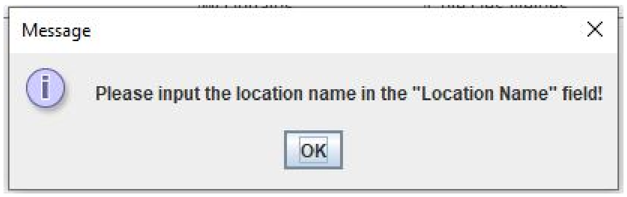
- **Location Tab.** This tab allows adding and deleting locations in which meals can be consumed. Initially Home indicates Indining and it has been set as location 0. Any other location would signify that it is Outdining. To delete a location the user has to select the location from the table and click the DELETE Button at the bottom.



ID	NAME	ADDRESS
0	Home	Montreal, somewhere
1	Tim Hortons	Guy Concordia
2	Starbucks	Near Sherbrook
3	Dominos Pizza	Saint Catherine

Location Tab

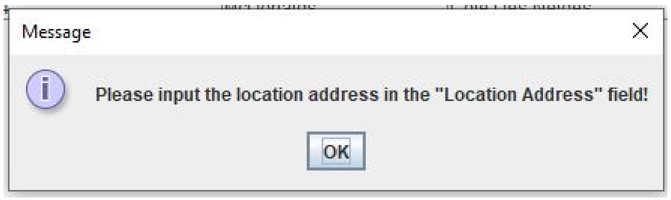
To add a new location the user has to enter the name of the location along with the address and press Insert Button. If the location name text field is empty the user will see a message notifying them of this.


Message

Please input the location name in the "Location Name" field!

Message notifying user of an empty Location name field

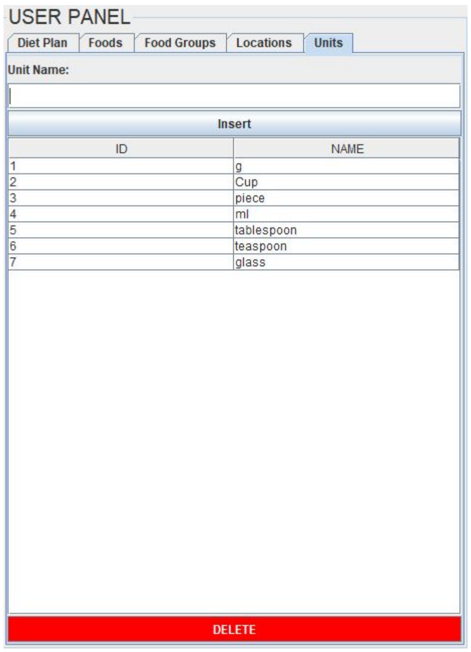
If the location address text field is empty the user will see a message notifying them of this.


Message

Please input the location address in the "Location Address" field!

Message notifying user of an empty Location address field

- **Unit Tab.** This tab allows the user to add a new unit of measurement of food. To do so, the user needs to enter the name of the unit to the text field and press the Insert Button. To delete a unit the user has to select the unit from the table and click the DELETE Button at the bottom.

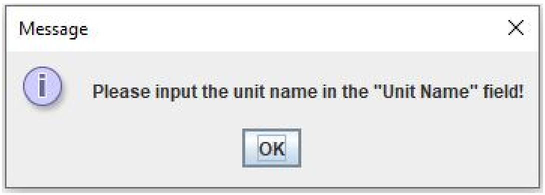


Unit Name:

ID	NAME
1	g
2	Cup
3	piece
4	ml
5	tablespoon
6	teaspoon
7	glass

Unit Tab

If the user leaves the Unit name text field empty a message will appear to notify the user.

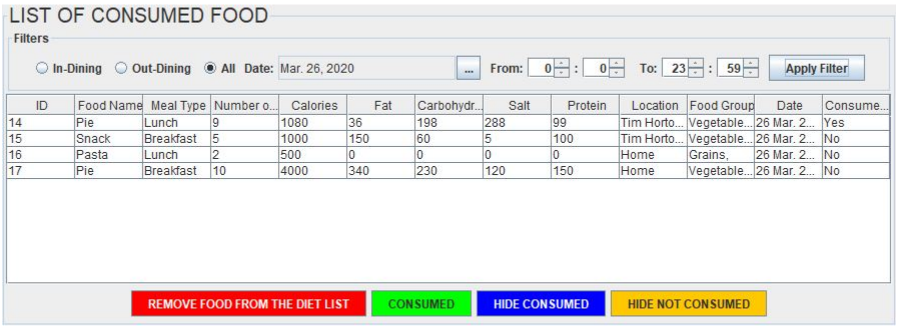


Please input the unit name in the "Unit Name" field!

Message notifying user of an empty Unit name field

4.1.1.3 LIST OF CONSUMED FOOD

This panel contains the information about the meals the user has consumed or is planning to consume in the future. Filters can be applied to show Indining, Outdining or both on a certain date. After setting the appropriate filters the user has to click Apply Filter Button to change the view of the panel. In case of errant data entry or if the user wishes to delete an entry from the Consumed List, they need to select the row in the table with the meal and press REMOVE FROM CONSUMED FOOD LIST Button at the bottom. In addition the user can set a meal as food as consumed or not consumed and hide consumed or not consumed foods from the panel view using the buttons at the bottom of the panel.



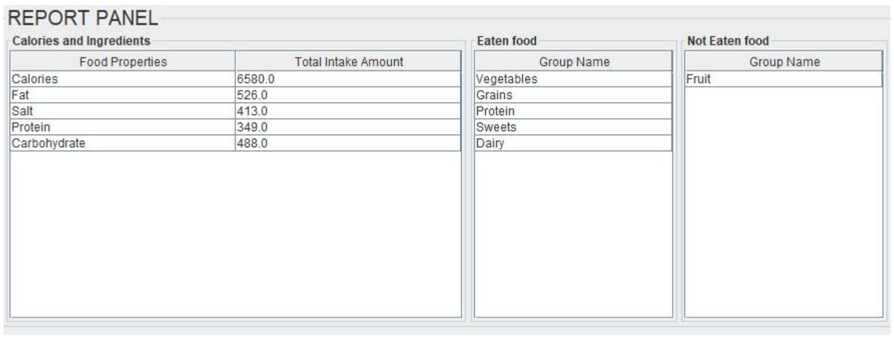
○ In-Dining ○ Out-Dining [✓ All] Date: Mar. 26, 2020 From: 0 : 0 To: 23 : 59

ID	Food Name	Meal Type	Number o...	Calories	Fat	Carbohydr...	Salt	Protein	Location	Food Group	Date	Consume...
14	Pie	Lunch	9	1080	36	198	288	99	Tim Horto...	Vegetable...	26 Mar. 2...	Yes
15	Snack	Breakfast	5	1000	150	60	5	100	Tim Horto...	Vegetable...	26 Mar. 2...	No
16	Pasta	Lunch	2	500	0	0	0	0	Home	Grains,	26 Mar. 2...	No
17	Pie	Breakfast	10	4000	340	230	120	150	Home	Vegetable...	26 Mar. 2...	No

LIST OF CONSUMED FOOD

4.1.1.4 REPORT PANEL

This panel allows the user to view the summary of the diet specified by the filters in the Consumed Food Panel. The user is presented with the total intake of calories and nutrients. In addition, the panel shows the food groups that have been consumed as well as those that have not. This panel is automatically updated when the user makes changes to their diet or applies filters to specify the diet they are interested in viewing.



REPORT PANEL

Calories and Ingredients

Food Properties	Total Intake Amount
Calories	6580.0
Fat	526.0
Salt	413.0
Protein	349.0
Carbohydrate	488.0

Eaten food

Group Name
Vegetables
Grains
Protein
Sweets
Dairy

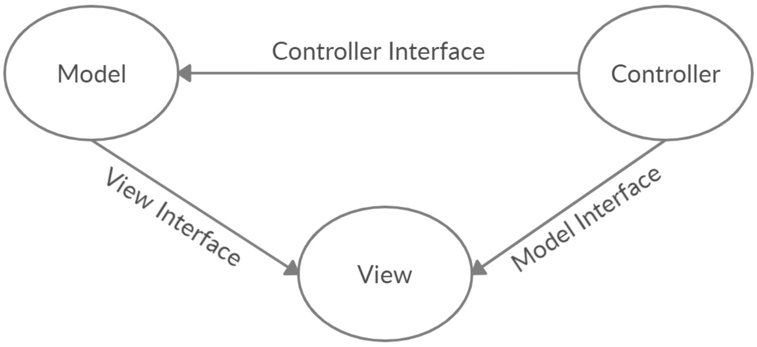
Not Eaten food

Group Name
Fruit

REPORT PANEL

4.2 Module Interface Diagrams

The three components of the MVC model use interfaces to interact with each other. In this section these interfaces will be presented in more detail.



MVC Interfaces

4.2.1 View Interface

This interface provides communication between Model and View. As soon as a change occurs in the Model, this is shown in the View.

4.2.1.1 Main Window

The Main Window is the only window that the user interacts with. It comprises of three major areas: Data Entry Panel, Consumed Food Panel and Report Panel.

4.2.1.2 Data Entry Panel

Data Entry Panel is a tabbed pane that allows the user easy navigation through the data, insertion of new data and deletion of data.

- MealPanel(): creates initial Diet Plan Panel.
- FoodPanel(): creates initial Food Panel.
- GroupPanel(): creates initial Group Panel.
- LocationPanel(): creates initial Location Panel.
- UnitPanel(): creates initial Unit Panel.

4.2.1.3 Consumed Food Panel

This panel contains meals that can be filtered based on the user. EatenMealPanel() method creates the initial table based on arbitrary filter options.

4.2.1.4 Report Panel

This panel presents summarized information based on the Consumed Food Panel and the filter that has been applied. ReportPanel() method creates the initial table based on arbitrary filter options.

4.2.2 Model Interface

The following interfaces have been established to allow communication between controller and model.

4.2.2.1 FoodDaoInterface

This interface allows manipulating Food objects. The following methods have been used in the interface:

- insert(): insert a new Food item.
- all(): return all Food objects.
- deleteAll(): delete all Food objects.
- delete(): delete the Food object with a given ID.
- findById(): return a Food object with a given ID.
- findByName(): return a Food object with a given name.

4.2.2.2 FoodGroupDaoInterface

This interface allows manipulating intermediary FoodGroup objects. The following methods have been used in the interface:

- insert(): insert a new Food-Group item.
- getGroupsOfOneFood(): returns the groups that a given Food belongs to.
- deleteAll(): delete all the Food Group objects.

4.2.2.3 GroupDaoInterface

This interface allows manipulating Food Group objects. The following methods have been used in the interface:

- insert(): insert a new Food Group item.
- all(): return all Food Group objects.
- deleteAll(): delete all Food Group objects.
- delete(): delete the Food Group object with a given ID.
- findById(): return a Food Group object with a given ID.
- findByName(): return a Food Group object with a given name.

4.2.2.4 LocationDoaInterface

This interface allows manipulating Location objects. The following methods have been used in the interface:

- insert(): insert a new Location item.
- all(): return all Location objects.
- deleteAll(): delete all Location objects.
- delete(): delete the Location object with a given ID.
- findById(): return a Location object with a given ID.
- findByName(): return a Location object with a given name.
- findByAddress(): return a Location object with a given address.

4.2.2.5 MealDaoInterface

This interface allows manipulating Meal objects. The following methods have been used in the interface:

- insert(): insert a new Meal item.
- all(): return all Meal objects.
- deleteAll(): delete all Meal objects.
- delete(): delete the Meal object with a given ID.
- findById(): return a Meal object with a given ID.
- findMealsByDate(): return a Meal object with a given date.
- findInRange(): return Meal objects within a given date range.

4.2.2.6 UnitDaoInterface

This interface allows manipulating Unit objects. The following methods have been used in the interface:

- insert(): insert a new Unit item.
- all(): return all Unit objects.
- deleteAll(): delete all Unit objects.
- delete(): delete the Unit object with a given ID.
- findById(): return a Unit object with a given ID.
- findByName(): return a Unit object with a given name.

4.2.3 Controller Interface

The Controller Interface validates the user input and if it is acceptable it calls changes to the model. Otherwise the user is shown a notification with a message telling them why their input cannot be processed.