



Custom Fit Coding, LLC

"The leader in providing custom creative solutions for your business"

Custom Fit Coding, LLC • 123 Anywhere Drive • Radford, Virginia 24142 • customfitcoding@gmail.com

Cooking to Goal Design

A concise guide for application implementation.

Steven Totten
David Ball
Stephanie Kinsella
Drew Salyer
Ryan Spoon

Custom Fit Coding, LLC
04/04/2011

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	<i>Steven Totten David Ball Stephanie Kinsella Drew Salyer Ryan Spoon</i>	Initial version	04/04/2011

Review & Approval

Design Document Approval History

Approving Party	Version Approved	Signature	Date
Steven Totten	1.0	Steven Totten	04/04/2011
D.P. Daugherty			

Design Document Review History

Reviewer	Version Reviewed	Signature	Date
Steven Totten			
Stephanie Kinsella			

Contents

[Revisions](#)

[Review & Approval](#)

[Contents](#)

[1. Introduction](#)

[1.1. System Overview](#)

[1.2. Supporting Materials](#)

[1.3. Definitions and Acronyms](#)

[2. Functional Requirements Overview](#)

[2.1. Guest Features](#)

[2.1.a. User Authentication Screen](#)

[2.1.b. User Registration Screen](#)

[2.2. User Profile](#)

[2.2.a. User Profile Home Screen](#)

[2.3. Meal Planning Features](#)

[2.3.a. Recipe Listing Screen \(Abstract\)](#)

[2.3.b. Browse Recipes Screen](#)

[2.3.c. Recipe Search Results Screen](#)

[2.3.d. Recipe Editor Screen \(Abstract\)](#)

[2.3.e. Create Recipe Screen](#)

[2.3.f. Edit Recipe Screen](#)

[2.3.g. View Recipe Screen](#)

[2.3.h. Delete Recipe Screen](#)

[2.4. Meal Planning Features](#)

[2.4.a. Enroll Recipe to Meal Feature](#)

[2.4.b. Family/Group Assignment to Meal Feature](#)

[2.4.c. Show Weekly Meal Assignments Screen](#)

[2.5. Shopping Features](#)

[2.5.a. Generate Shopping List Feature](#)

[2.5.b. Display Shopping List Screen](#)

[2.5.c. Print Shopping List Feature](#)

[2.6. Nutrition Planning Features](#)

[2.6.a. List Nutrition Plans Screen](#)

[2.6.b. Nutrition Plan Editor Screen \(Abstract\)](#)

[2.6.c. Create Nutrition Plan Screen](#)

[2.6.d. Edit Nutrition Plan Screen](#)

[2.6.e. Test Nutrition Targets Against Planned Meals Feature \(At-A-Glance\)](#)

[2.6.f. Test Active Nutrition Target Against Planned Meals Feature](#)

[2.6.g. Calculate Nutrition Plan Effectiveness \(%\) Feature](#)

3. High Level Design

3.1. Conceptual View (Architectural Context Model)

4. Low Level Design

4.1. Process Model

5. User Interface Design

5.1. Application Control

5.1.1. Screen Navigation Diagram

5.2 Program Screens

1. Introduction

While the requirements specifications document showed “what” your team plans to develop; this design document will illustrate “how” your team will develop your system.

1.1. System Overview

The Cooking to Goal application will have an easy to navigate graphical user interface which will help customers create and compare meals and menus for a weekly period. Users will begin by registering to use the software, if they are new, or logging in if they are current users. They will be able to choose from multiple menu options, allowing them to select a variety of meals while meeting their nutritional goals. A weekly grocery shopping list will be generated based upon their meal selections.

A more detailed statement of scope for this project can be found on page 6 of the Cooking to Goal Software Project Management Plan.

1.2. Supporting Materials

Ball, David. Cooking to Goal Software Project Management Plan. Version 2. Radford, Virginia: Custom Fit Coding, LLC; February 21, 2011.

Spoon, Ryan. Cooking to Goal Software Requirements Specification. Version 1. Radford, Virginia: Custom Fit Coding, LLC; March 21, 2011.

“How to Understand and Use the Nutrition Facts Label”, <http://www.fda.gov/food/labelingnutrition/consumerinformation/ucm078889.htm>

1.3. Definitions and Acronyms

Automation Boundary: Set of actions and functions that a user may perform .

Family Member: Person in the application users family utilizing the functionality of the application or using the application in concert with the user.

Guest: A user of the application that chooses not to register and log in with the application.

Guest Actions: Actions that users not registered with the application may perform.

Ingredient:Single piece of a recipe.

Instruction:Direction set given to create a recipe.

Meal: Recipe or multiple recipes combined.

Meal Plan: Set of meals for a given period of time (i.e. a week)

Nutrition Information: Nutritional values given for each item to include information such as total carbohydrates, fat, sodium, fiber, etc.

Nutrition Plan: Set of goals a user has for their nutritional intake.

Nutrition Plan Editor: Means to edit a users nutritional goals.

Nutrition Plan Effectiveness (%): How effective a given users nutritional goals are.

Recipe: Multiple ingredients combined together.

Recipe Editor: Means to add, modify or delete a recipe from the application.

Shopping List: List of items necessary to be bought in order to cook the chosen recipes.

System Boundary: How users (to include guests and family members) interact with the functions and actions the application is capable of performing.

User: Person utilizing the application who has registered and logged in.

User Authentication: guests who are existing users will be prompted with the option to log in to the system or exit the application.

User Profile: selections each user may make in order to personalize their goals and meal preferences.

User Profile Actions: actions available for a particular type of person using the system .Users, guests and family members may all have different actions and functions available to them.

User Profile Home: identifies the user in some way, such as by their nutritional goals or by their selected meal plan.

User Registration: the area in which a user can complete a registration form order to gain access to the system.

2. Functional Requirements Overview

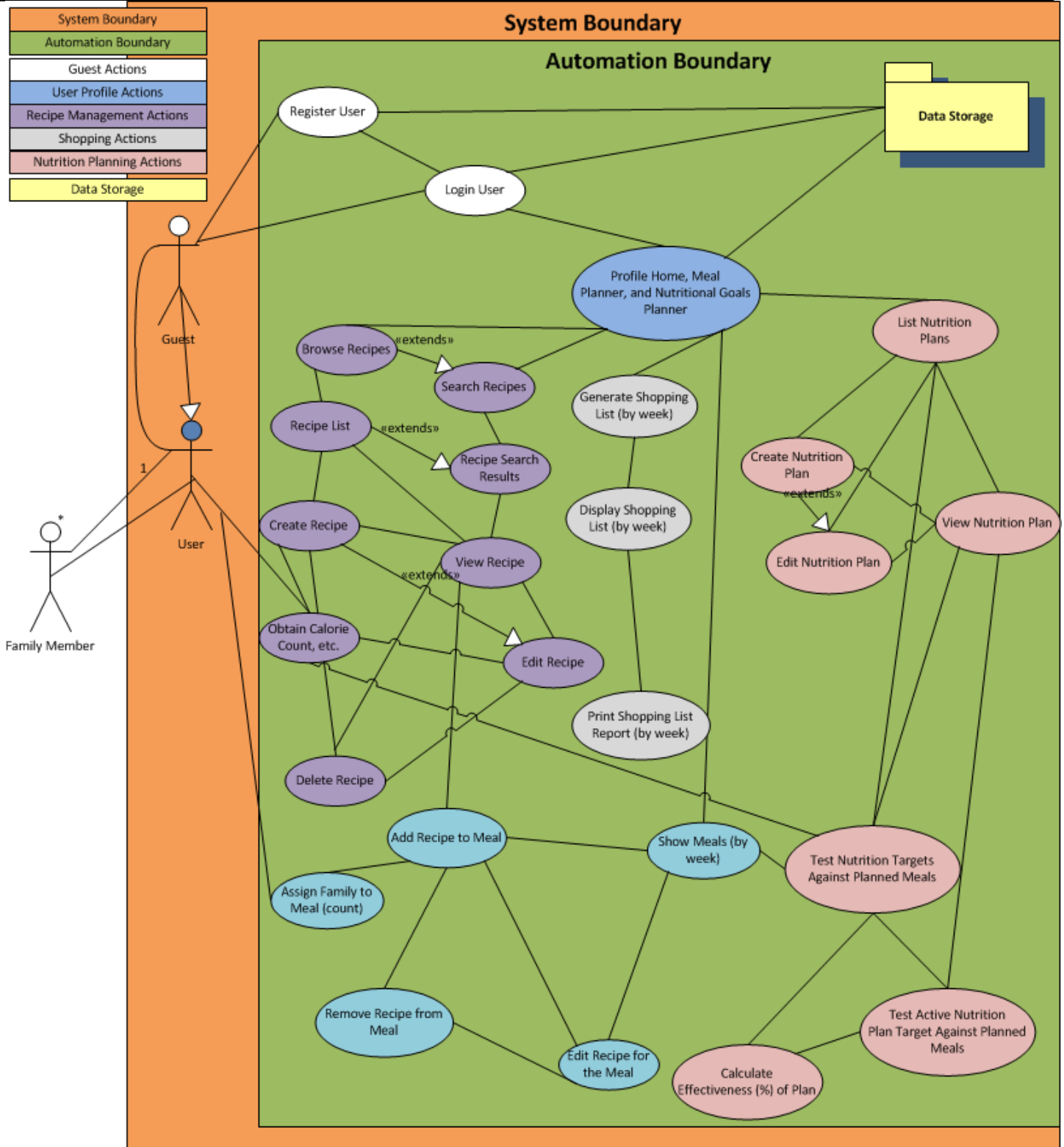


Figure 2.1, Use Case Diagram for Guest and User Actors

This chapter is focused on the baseline functional requirements of the designed information system. Notice that in Figure 2.1 above, there are 2 boundaries inside the graphical universe. The system boundary depicts all the parts of the information system and the automation boundary suggests what type of actions we can program into the system. All the ovals represent actions in the system. Each of the colors corresponds with a section in this chapter. The yellow box represents application data, however implemented. There are two actors managed by the system: Guest and User.

Figure 2.1, Use Case Diagram for Guest and User Actors, indicates the relationship of two distinct actors in the **System Boundary**, who are *Guest* and *User*, and in what ways these actors relate to application system. A *Guest* is transformed to a *User* as a login is completed. By logging in, a Profile Home permits entry to all other program features.

In the following design specification, we outline a potential implementation scenario and the requirements that the design entails. The requirements are divided into six primary functional categories, each with corresponding features. This outline tries to break it down into a few key areas:

- 2.1. Guest Features (*White in Figure 2.1*)
 - 2.1.a. User Authentication Screen
 - 2.1.b. User Registration Screen
- 2.2. User Profile Feature(s) (*Blue in Figure 2.1*)
 - 2.2.a. User Profile Screen
 - 2.2.b. Data Storage Feature
- 2.3. Recipe Management Features (*Purple in Figure 2.1*)
 - 2.3.a. Recipe Listing Screen (Abstract)
 - 2.3.b. Browse Recipes Screen
 - 2.3.c. Recipe Search Results Screen
 - 2.3.d. Recipe Editor Screen (Abstract)
 - 2.3.e. Create Recipe Screen
 - 2.3.f. Edit Recipe Screen
 - 2.3.g. View Recipe Screen
 - 2.3.h. Delete Recipe Screen
- 2.4. Meal Planning Features (*Cyan in Figure 2.1*)
 - 2.4.a. Enroll Recipe to Meal Feature
 - 2.4.b. Family/Group Assignment to Meal Feature
 - 2.4.c. Show Weekly Meal Assignments Screen
- 2.5. Shopping Features (*Gray in Figure 2.1*)
 - 2.5.a. Generate Shopping List Feature
 - 2.5.b. Display Shopping List Screen
 - 2.5.c. Print Shopping List Feature
- 2.6. Nutrition Planning Features (*Pink in Figure 2.1*)
 - 2.6.a. List Nutrition Plans Screen
 - 2.6.b. Nutrition Plan Editor Screen (Abstract)
 - 2.6.c. Create Nutrition Plan Screen
 - 2.6.d. Edit Nutrition Plan Screen
 - 2.6.e. Test Nutrition Targets Against Planned Meals Feature (At-A-Glance)
 - 2.6.f. Test Active Nutrition Plan Target Against Planned Meals Feature
 - 2.6.g. Calculate Nutrition Plan Effectiveness (%) Feature

2.1. Guest Features

In Figure 2.1, Use Case Diagram for Guest and User Actors, the relationship between the Guest actor the Register User and Login User action are shown.

The following two Functional Requirements describe these two actions and how they relate the Guest actor to the system.

2.1.a. User Authentication Screen

2.1.b. User Registration Screen

2.1.a. User Authentication Screen

On the **User Authentication Screen**, guests who are existing users will be prompted with the option to log in to the system or exit the application, which may simply be by having an [X] on the title bar but may include any detail needed to complete the user experience. Guests who want to register may click a link/button to activate the **User Registration Screen**, which is detailed in **Functional Requirement 2.1.b**.

Examining **Figure 2.1, Use Case Diagram for Guest and User Actors**, there exists a relationship between the *Guest* actor the *Register User* and *Login User* actions. You can also see that these two actions on the figure are related to each other, indicating that communication could cross functional boundaries, and subsequently a newly registered user may be logged in as part of a single action/transaction. This is, of course, and always, optional, and would of course, seem to be beneficial.

The figure also shows the transformation of a *Guest* actor to a *User* actor by the act of logging in using this screen.

An example user interface for a **User Authentication Screen** exists in Section {{{REF}}}.

2.1.b. User Registration Screen

When a *Guest* actor on the **User Authentication Screen**, in **Functional Requirement 2.1.a.**, does not have a user name and/or password stored within the system boundaries specified in **Figure 2.1, Use Case Diagram for Guest and User (authenticated use)**, the user may click, activate, etc. some link, button, etc., to activate this **User Registration Screen**.

The **User Registration Screen** is the area a guest can complete a registration form to complete enrollment into the system. Once enrolled, the user may come back to the **User Authentication Screen** any time thereafter to complete a login process.

A user registration may consist of certain details considered important to the application. The minimum amount of information needed in order to proceed into further functional requirements is a User Name and a Password. It would probably pay to hash the visible text in the Password input, if provided on-screen or sent in network-transit, which in our case, will be a text field. If, for example, the user's real name were intended to be important in terms of some report, etc., it would need to be prompted for. {{we should decide on this by the end of this section}}

Examining **Figure 2.1, Use Case Diagram for Guest and User Actors**, there exists a relationship between the *Guest* actor the *Register User* and *Login User* actions. You can also see that these two actions on the figure are related to each other, indicating that communication could cross functional boundaries, and subsequently a newly registered user may be logged in as part of a single action/transaction. This is, of course, and always, optional, and would of course, seem to be beneficial.

An example user interface for a **User Registration Screen** exists in Section {{{REF}}}.

2.2. User Profile

In **Figure 2.1, Use Case Diagram for Guest and User Actors**, the relationship between the *User* actor and the *Profile Home*, *Meal Planner*, and *Nutritional Goals Planner* action are shown.

The following functional requirements describe this action and how the action relates the *User* actor in this system. It also touches on some Data Storage Feature requirements.

2.2.a. User Profile Home Screen

2.2.b. Data Storage Feature

2.2.a. User Profile Home Screen

By logging into system on the **User Authentication Screen** described in **Functional Requirement 2.1.a.**, the authenticated user, hereafter called the user, will have access to most of the remaining application features, by using some process of navigation within the application. The navigation in this design is a clearly-thought, well-documented architecture, though it only represents the work of a single fully-working version of the application. In future versions this profile screen may feature as many items as the client and/or customer might could imagine.

For this upcoming release, it would be nice if the **User Profile Home Screen** identifies with the user in some way. In **Functional Requirement 2.6, Nutrition Planning** is discussed in great detail. A portion of the **User Profile Home Screen** may detail/summarize the **Nutrition Targets** of an activated **Nutrition Plan** in a single aspect, column, area, graphic, etc.

The **Meal Planning** features described in **Functional Requirement 2.4** represents a large component of what we're doing here inside the system boundary, so an area dedicated to a summary of current upcoming meal plans might be a potential candidate view for this set of data. This user interface candidate may take up a column, or reasonable area of space on the **User Profile Home Screen**. An area of this **User Profile Home Screen** could be used to initiate a *Search/Browse for Recipes* action that could be used to initiate the feature detailed in **Functional Requirement 2.3.c., Recipe Search Results Screen**, probably by means of a Search, etc., input text field for searching through the recipes. Somewhere near that, you may also provide a button/link to create new recipes as well; the process of which is outlined in **Functional Requirement 2.3.e.**

In addition to searching for recipes, there is a certain order about Recipes such that Categories and/or Tags may provide a means to organize or otherwise categorize recipes by any conceivable type or other means of qualification, including hot/cold, food/beverage, fast/all-day cooking, seasonal, and at least as many categories as one could imagine. It is totally going to be up to the client and/or the end user to make these distinctions as they view the importance of the qualifiers, either by prefabricated data (provided by the client/vendor) or customized data (provided by the user).

It is also required that we are able to generate shopping lists by the week. Buttons/links/etc. to these features indicated in **Functional Requirements 2.5, Shopping Features**, might be appropriate in an area near the **Browse Recipes** or **Search Recipes** actions on the User Profile Home Screen.

An example user interface for a **User Profile Home Screen** exists in Section {{{REF}}}.

2.2.b. Data Storage Feature

2.3. Meal Planning Features

In **Figure 2.1, Use Case Diagram for Guest and User Actors**, the relationship between the *User's Profile Home*, *Meal Planner*, and *Nutritional Goals Planner* action and the *Browse Recipes*, *Search Recipes*, *Generate Shopping List (by week)*, and *List Nutrition Plans* actions are shown. It also attempts to summarize which requirements depend on each other from the top of the tree to the bottom.

The following functional requirements describe these actions and how they relate the Guest actor to the system through the *Profile Home*, *Meal Planner*, and *List Nutrition Plans* program feature outlined in the use case diagram.

- 2.3.a. Recipe Listing Screen (Abstract)
- 2.3.b. Browse Recipes Screen
- 2.3.c. Recipe Search Results Screen
- 2.3.d. Recipe Editor Screen (Abstract)
- 2.3.e. Create Recipe Screen
- 2.3.f. Edit Recipe Screen
- 2.3.g. View Recipe Screen
- 2.3.h. Delete Recipe Screen

2.3.a. Recipe Listing Screen (Abstract)

For the purposes of this application, consider a recipe. Each recipe contains at least a name, one or more ingredients, and instructions. Given what we know about recipes, we can shorten the detailed knowledge to a condensed set of facts for a short listing.

There could be as many ideal examples of listing as there could precipitate a user need for opening a Recipe. This could include, high rankings/ratings, user comments (perhaps a distant future feature), certain ingredients compared to others or in general, number of steps, etc. They may be able to be sorted or not, and by some field.

In any case, a listing screen here is nothing more than a summary, or short form, of multiple recipes in a concise format. Some of the other facts known, categories, ratings, etc., might be used as filters to limit the criteria displayed. This could be a sidebar with different options in separate lists, or just about any other way you might imagine applying search filters.

Amongst the types of information that can be displayed here, the space on this screen can also be used to activate create a new recipe or edit an existing recipe actions if it is seen fit.

Given our current requirements, we have two examples, which are also **Functional Requirements 2.3.b.-2.3.c., Browse Recipes Screen** and **Recipe Search Results Screen**, that will either directly inherit the same class, interface, etc., or will at a minimum share the concepts detailed here.

some information is {{{{DEFERRED_FOR_FUTURE_ITERATION}}}}

2.3.b. Browse Recipes Screen

This screen inherits the idea considered in **Functional Requirement 2.3.a., Recipe Listing Screen (Abstract)**, and can either directly inherit, implement, extend, etc., the **Recipe Listing Screen** or simply implement the ideas conveyed in it.

This feature can be activated by program features detailed in **Functional Requirements 2.2.a., User Profile Home Screen**.

The **Browse Recipes Screen** is merely a **Recipe Listing Screen (Abstract)** dressed up for Browsing. There could be some tree listing, sets of lists, perhaps in some sidebar, menu, etc., that could display a (un-)/sorted list of Recipe Categories, for which to organize the list of Recipes on the screen and to limit the search by certain filter parameters, as needed.

While the user browses through the recipe list, it may occur to them that they might want to add a new one. So we should be able to add a recipe from some action on this screen.

An example user interface for a **Browse Recipes Screen** exists in Section {{{REF}}}.

2.3.c. Recipe Search Results Screen {{{DEFERRED_FOR_FUTURE_ITERATION}}}

This screen also considers **Functional Requirement 2.3.a., Recipe Listing Screen (Abstract)**, and it can either directly inherit, implement, extend, etc., the **Recipe Listing Screen** or simply implement the ideas conveyed in it.

This feature can be activated by program features detailed in **Functional Requirements 2.2.a., User Profile Home Screen**.

The **Recipe Search Results Screen** is simply a **Recipe Listing Screen (Abstract)** suited for Searching. There could be some tree listing, sets of lists, categories of returned results, perhaps in some sidebar, menu, etc., that could display a (un-)/sorted list of Recipe Categories, for which to organize the list of Recipes on the screen and any other parameters which may be used to limit the search by specific parameters (e.g. Tags, Ratings, ..., etc.), as needed.

While the user browses through the recipe search results, it may occur to them that they might want to add a new one. So we should be able to add a recipe from some action on this screen.

An example user interface for a **Recipe Listing Screen** exists in Section {{{REF}}}.

2.3.d. Recipe Editor Screen (Abstract)

The **Recipe Editor** screen is a general prototype for the *Create Recipe* and *Edit Recipe* actions, and it be used for both purposes.

A **Recipe Editor Screen** should permit user input for all the fields of a recipe, which at a minimum should include a name, a list of ingredients, a list of instructions (perhaps at some point in a future version *instruction groups*, as well), and for the purposes of this application, we will need nutrition information for each recipe as well.

If you examine **Figure 2.1, Use Case Diagram for Guest and User Actors**, again, you can see that the *User* actor is called on for to the Obtain Nutrition Information action. That does mean that this is the point we need that information, the nutrition information entered here will be used later in the application to check the effectiveness of a nutrition plan.

Given our current requirements, we have two examples, which are also **Functional Requirements 2.3.e.-2.3.f., Create Recipe Screen** and **Edit Recipe Screen**, that will either directly inherit the same class, interface, etc., or will at a minimum share the concepts detailed here.

2.3.e. Create Recipe Screen

This screen inherits the idea considered in **Functional Requirement 2.3.d., Recipe Editor Screen (Abstract)**,

and can either directly inherit, implement, extend, etc., the **Recipe Editor Screen** or simply implement the ideas conveyed in it.

This feature can be activated by program features detailed in **Functional Requirements 2.2.a., User Profile Home Screen, Functional Requirements 2.3.b., Browse Recipes Screen, and Functional Requirements 2.3.c., Recipe Search Results Screen.**

The **Create Recipe Screen** is simply a **Recipe Editor Screen** that will have empty/initial fields. The user must complete all the required fields on this editor. Once the user saves the Recipe entered here, it should be part of the recipe lists generated by browsing and searching and should be able to be assigned to meals in the feature described in **Functional Requirement 2.4.a., Enroll Recipe to Meal Feature.**

An example user interface for a **Recipe Editor Screen** exists in Section {{{REF}}}.

2.3.f. Edit Recipe Screen

This screen inherits the idea considered in **Functional Requirement 2.3.d., Recipe Editor Screen (Abstract)**, and can either directly inherit, implement, extend, etc., the **Recipe Editor Screen** or simply implement the ideas conveyed in it.

This feature can be activated by program features detailed in **Functional Requirements 2.2.a., User Profile Home Screen, Functional Requirements 2.3.b., Browse Recipes Screen, and Functional Requirements 2.3.c., Recipe Search Results Screen.**

The **Edit Recipe Screen** is simply a **Recipe Editor Screen** that will have pre-populated fields with the recipe's current data provided in the form. The user must complete all the required fields on this editor. Once the user saves the Recipe entered here, it should update the recipe featured in the recipe lists generated by browsing and searching.

An example user interface for a **Recipe Editor Screen** exists in Section {{{REF}}}.

2.3.g. View Recipe Screen

This screen can be similar in layout to any of the recipe editor screens, **Functional Requirements 2.3.d-2.3.f.**, but won't feature any editing properties.

A **View Recipe Screen** should permit display the known fields of a recipe, which at a minimum should include a name, a list of ingredients, a list of instructions (perhaps at some point in a future version *instruction groups*, as well), and for the purposes of this application, we will need nutrition information for each recipe as well.

An example user interface for a **View Recipe Screen** exists in Section {{{REF}}}.

2.3.h. Delete Recipe Screen

This screen can be really simple. It could just be a quick message box, "Are you sure you want to delete this recipe?" which might add, "This process is irreversible. Do you want to continue?" The idea for this screen is simple enough, yet it is a Functional Requirement and corresponds to an action on **Figure 2.1, Use Case Diagram for Guest and User Actors**. The process will delete a recipe and no longer show the recipe in recipe lists and should remove any forthcoming meal assignments containing the recipe.

2.4. Meal Planning Features

In **Figure 2.1, Use Case Diagram for Guest and User Actors**, the relationship between the *User* actor and the *Profile Home, Meal Planner, and Nutritional Goals Planner* action and the relationship between this action and several Meal Planning Actions are indicated.

The following functional requirements describe these relationships.

- 2.4.a. Enroll Recipe to Meal Feature
- 2.4.b. Family/Group Assignment to Meal Feature
- 2.4.c. Show Weekly Meal Assignments Screen

2.4.a. Enroll Recipe to Meal Feature

This application feature will essentially “copy” the template for the recipe to the meal that the user is enrolling. Meals in this application refer to assignments of one or more recipe at a certain time of day during a certain day. So it is correct to say that a meal has one of more recipes, which are copied from their stored definition. Also, the recipe for the meal can be

2.4.b. Family/Group Assignment to Meal Feature

2.4.c. Show Weekly Meal Assignments Screen

2.5. Shopping Features

In **Figure 2.1, Use Case Diagram for Guest and User Actors**, the relationship between the *User* actor and the *Profile Home, Meal Planner, and Nutritional Goals Planner* action and the relationship between this action and several Shopping Actions are indicated.

The following functional requirements describe these relationships.

- 2.5.a. Generate Shopping List Feature
- 2.5.b. Display Shopping List Feature
- 2.5.c. Print Shopping List Screen

2.5.a. Generate Shopping List Feature

2.5.b. Display Shopping List Screen

2.5.c. Print Shopping List Feature

2.6. Nutrition Planning Features

In **Figure 2.1, Use Case Diagram for Guest and User Actors**, the relationship between the *User* actor and the *Profile Home, Meal Planner, and Nutritional Goals Planner* action and the relationship between this action and several Nutrition Planning Actions are indicated.

The following functional requirements describe these relationships.

- 2.6.a. List Nutrition Plans Screen
- 2.6.b. Nutrition Plan Editor Screen (Abstract)

- 2.6.c. Create Nutrition Plan Screen
- 2.6.d. Edit Nutrition Plan Screen
- 2.6.e. Test Nutrition Targets Against Planned Meals Feature (At-A-Glance)
- 2.6.f. Test Active Nutrition Target Against Planned Meals Feature
- 2.6.g. Calculate Nutrition Plan Effectiveness (%) Feature

2.6.a. List Nutrition Plans Screen

2.6.b. Nutrition Plan Editor Screen (Abstract)

2.6.c. Create Nutrition Plan Screen

2.6.d. Edit Nutrition Plan Screen

2.6.e. Test Nutrition Targets Against Planned Meals Feature (At-A-Glance)

2.6.f. Test Active Nutrition Target Against Planned Meals Feature

2.6.g. Calculate Nutrition Plan Effectiveness (%) Feature

Functional Requirement	Design Artifact
2.1 User Maintenance Screen <i>Guests may register here and recurring users will log in.</i>	See Figure 2.1, <i>Use Case Diagram for Guest and User (authenticated user).</i>
2.2 View Nutritional Goals <i>Users will be prompted to enter their desired nutritional goals for the week.</i>	Figure 2.2 <i><Description></i>
2.3 Meal Planner <i>Users will begin planning their weekly menus here by searching through prepopulated meals. They will be given the option to edit, add and delete meals as they seen fit.</i>	Figure 2.3 <i><Description></i>
2.4 Manage Recipes in the Recipe Box <i>The recipe box will consist of recipes that will help to generate the items contained in the shopping list.</i>	Figure 2.4 <i><Description></i>
2.5 Printable Shopping List <i>Users will be able to generate a shopping list of the ingredients needed to prepare the meals within the weekly menu.</i>	Figure 2.5
2.7 Printable Recipes <i><Description></i>	<i><Figure #><Description></i>

This table can be removed.

Stick to this:

Version 1 Features

The following features are the most important features in terms of the minimum viable product.

Feature	Summary	Priority
User Maintenance Screen	Create, Read, Update, Delete users	High
Recipe Ratings	Allow the user to rate recipes	High
Meal Planner	Quantity calculator	High
Manage Recipe Data in the Recipe Box	Create, Read, Update, Delete recipes	High
Print Shopping List	Generate and print a shopping list	High
View Nutritional Goals	View the users nutritional goals compared to the weekly menu	Medium
Printable Recipes	Generate and print a recipe	Low

To-Do: Review all **Functional Requirements** and mark all features that do not match this table as {{{DEFERRED_FOR_FUTURE_ITERATION}}} on the offending heading. Some features depend on other features in documentation, so be sure not to delete them but do add the tag. --David

<< UML GOES HERE>>
Figure 2.2

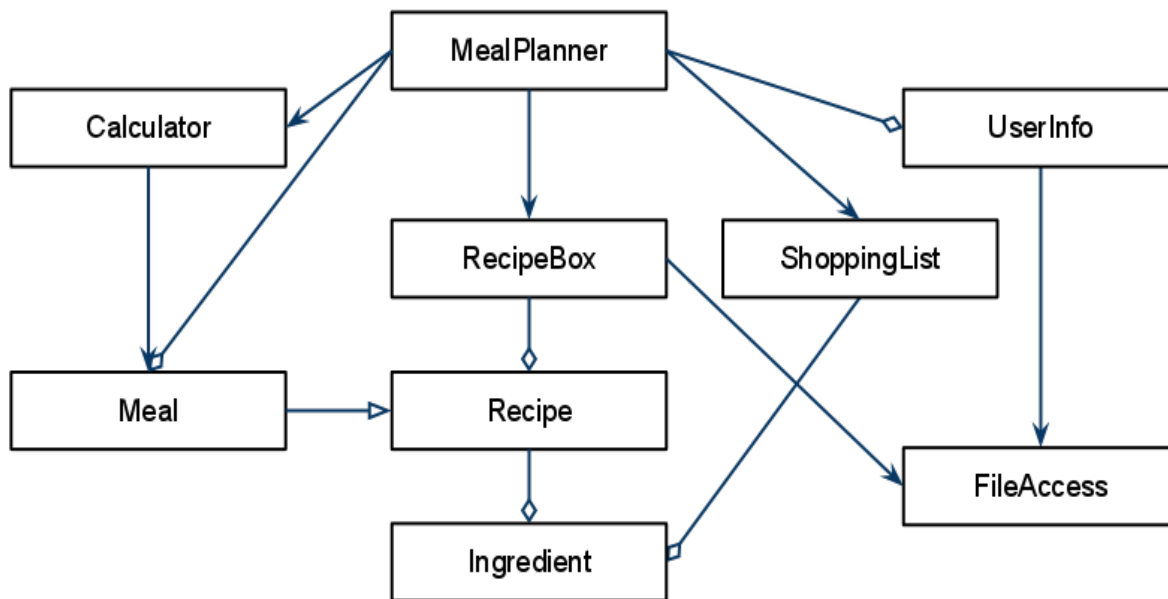
<< UML GOES HERE>>
Figure 2.3

<< UML GOES HERE>>
Figure 2.4

3. High Level Design

3.1. Conceptual View (Architectural Context Model)

This section should include a context diagram showing a high-level relationship between your system and all associated subsystems (ie. Database, internet, security, etc).



4. Low Level Design

4.1. Process Model

Create a process model that shows the flow of data between the processes within your system as well as the flow of data as it is transferred between your system and other external systems.

Provide an activity diagram for major functionality.

5. User Interface Design

This section provides user interface design descriptions that directly support construction of user interface screens.

5.1. Application Control

Detail the common behavior that all screens will have. Common look and feel details such as menus, popup menus, toolbars, status bar, title bars, etc.

Provide a screen navigation diagram for main screens

5.1.1. Screen Navigation Diagram

GUI Layout Flow
for Cooking To Goal Version 1.0

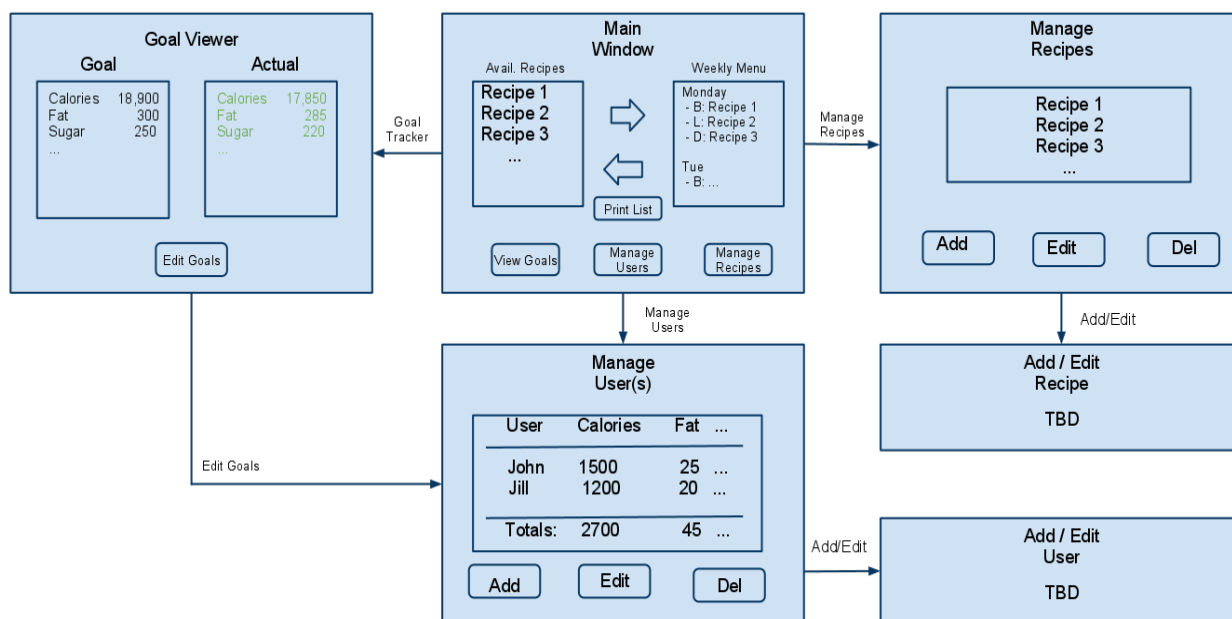


Figure 5.1

5.2 Program Screens

5.2.1 The Meal Planner screen (Figure 5.2.1) is the main window of the application. From here, the user may select from predefined recipes to add to their weekly meal plan or choose to add their own recipes to the system. As meals are added to the weekly meal plan the application will calculate the necessary serving sizes to stay within the designated nutritional goals.

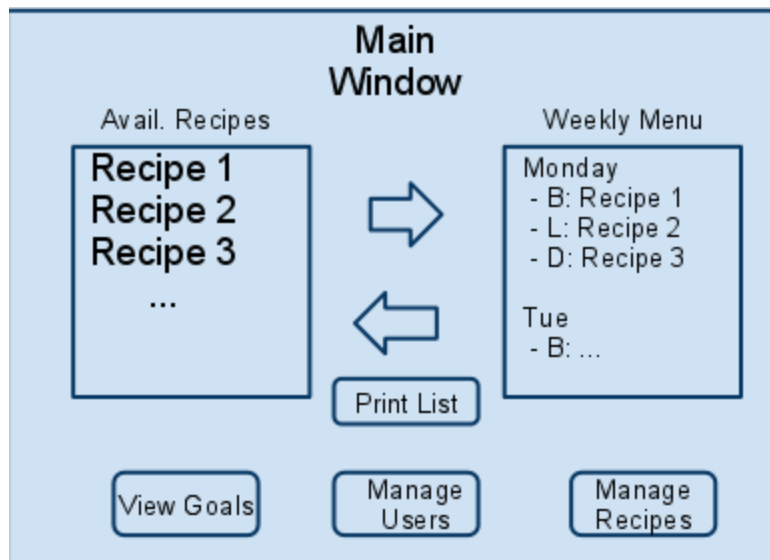


Figure 5.2.1

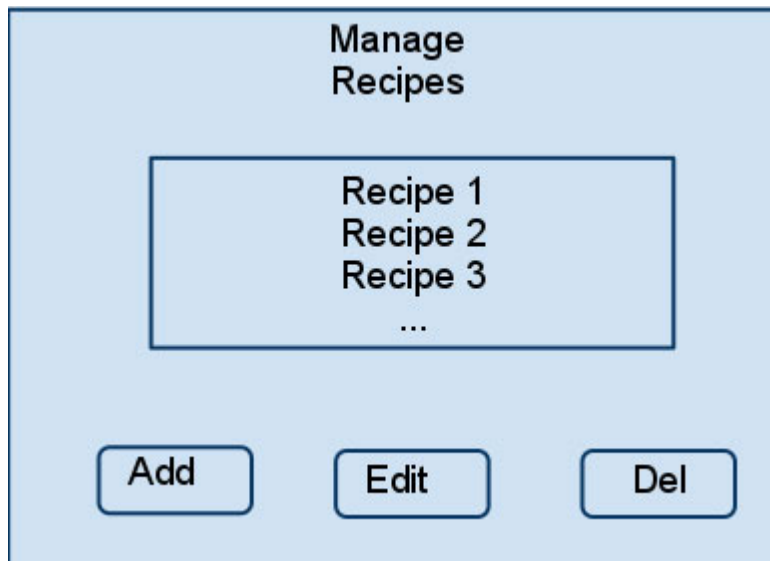
5.2.2 The User Maintenance screen (Figure 5.2.2) is the container for user details such as name, age, nutritional goals, etc. Users may add, edit, or delete their profiles from within this feature. Upon the first initial launch of the program, if a user does not wish to enter their information immediately, a default profile is loaded so that the user may explore the system immediately. The user may claim the default profile at any time, as well as add additional profiles for family members or guests who also have nutritional goals.

The 'Manage User(s)' screen displays a table with user information and three action buttons at the bottom: 'Add', 'Edit', and 'Del'.

User	Calories	Fat	...
John	1500	25	...
Jill	1200	20	...
Totals:	2700	45	...

Figure 5.2.2

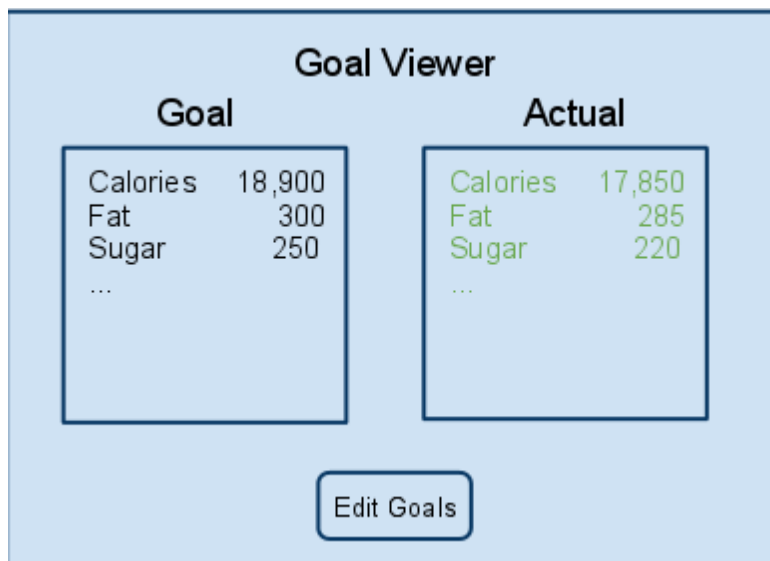
5.2.3 The Manage Recipes screen (Figure 5.2.3) allows users to enter their own recipes to the meal selection menu as well as edit and delete recipes from the meal list. Recipes will include a name, the nutritional information for the prepared recipe, ingredients, cooking instructions, serving size, and a user rating.



The 'Manage Recipes' screen features a light blue background. At the top, the title 'Manage Recipes' is centered. Below it, a rectangular box contains a list of recipes: 'Recipe 1', 'Recipe 2', 'Recipe 3', and an ellipsis '...'. At the bottom of the screen, there are three rounded rectangular buttons labeled 'Add', 'Edit', and 'Del'.

Figure 5.2.3

5.2.4 The Goal Viewer screen (Figure 5.2.4) provides the ability to view the combined nutritional goals of all of the users in the system, side-by-side with the scaled nutritional totals of their weekly meal selections.



The 'Goal Viewer' screen has a light blue background. The title 'Goal Viewer' is centered at the top. Below the title, there are two side-by-side rectangular boxes. The left box is titled 'Goal' and contains the text: 'Calories 18,900', 'Fat 300', 'Sugar 250', and an ellipsis '...'. The right box is titled 'Actual' and contains the text: 'Calories 17,850', 'Fat 285', 'Sugar 220', and an ellipsis '...'. The text in the 'Actual' box is green. At the bottom center of the screen, there is a rounded rectangular button labeled 'Edit Goals'.

Figure 5.2.4