FoodWatch

Sean Miller; Group Manager
Brandon Zahl; Designer
Mike Silver; User Testing
Kyle Hornberger; Writer
CSE 440; Autumn 2009

Problem and Solution Overview

Food waste is a huge problem in America today. A large component of this problem is the average person's difficulty in keeping track of their food. Proper food purchasing and storage will help mitigate this, and hopefully reduce the amount of food wasted. That's where FoodWatch steps in, by allowing customers to view an inventory of their current food stocks, receive notifications when their food is about to spoil (and should be used), and be given reminders about proper food storage when appropriate.

Task Analysis Review:

1. Who is going to use the system?
   Generally speaking, anybody who wants to keep track of the food they have stored. This will usually be heads of residences or people living on their own. We're targeting the 18-60 year old, English speaking demographic, with the understanding that most users will not be engineers (especially older adults, who have not grown up with computing), and also must be easy to use with one hand, for ease while shopping.

2. What tasks do they perform?
   Currently, most people track purchases and things to do on paper lists. This can include when certain items go bad, but most aren't that organized with their food. As opposed to watching the spoil date in advance, most people simply find out after it has already gone bad. FoodWatch will automate both of these, making them faster and less prone to human error.

3. What tasks are desired?
   After observing people's interactions with food purchase and storage, the key tasks for our app are tracking the current food inventory (along with spoil dates), and recipe search to ease meal planning. Additionally, we feel it's beneficial to include some form of statistic tracking as a motivational factor when using the application.

4. How are the tasks learned?
   All of the tasks involved are fairly basic. Creating and managing lists is a fairly basic skill learned in school. Determining whether food has spoiled or not is a much less accurate skill, and is generally learned through (usually bad) experiences in conjunction with store "sell by" dates. In FoodWatch both are fairly simple – navigating through lists and performing logical

actions on items.

5. Where are the tasks performed?
   Typically, these 'food management' tasks are performed either at home or in the grocery store.

6. What's the relationship between customer and data?
   The data is reflective of reality, but in our implementation will not be created by the customer. In a group-living situation, giving everyone the ability to monitor the food supplies is a good thing, so it must be able to be shared. The capability to be printed and/or compatibility with other data formats is also very desirable.

7. What other tools does the customer have?
   To remind them of their food stocks and expiry dates, people can rely on other people and their collective memory. To determine if food has spoiled, people can check their refrigerator's temperature, or utilize some new packaging (e.g. meat will have a nitrogen sensor).

8. How do customers communicate with each other?
   Currently, shared lists are the typical method of getting food requests or ideas to someone else. A household may have one grocery list that everyone adds to incrimentally, or everyone may gather together when it comes time to clean out the fridge. If someone is already at the store, they can use their phone to call back and see if something is at home/still edible. FoodWatch will remove the need for that last item, and will automate or at the very least make more permanent the first.

9. How often are the tasks performed?
   People eat every day, and go shopping once or twice a week. Refrigerators are usually cleaned every few months, but this definitely varies from residence to residence. FoodWatch will be used every time one goes shopping, and automatic reminders should remove the need for large refrigerator cleaning operations.

10. What are the time constraints on tasks?
    Making a list typically takes a few minutes at maximum. Viewing and interacting with the inventory is the main function of the application, and must be quick. The same goes with looking up recipes, as it's intended to be used in-store.

11. What happens when things go wrong?
    If something goes wrong, some food might spoil. This is not the end of the world, as the customer still has the rest of their tools (sight, smell, temperature) to use to avoid eating spoiled food – and hopefully it will serve as a reminder of how valuable our system is.

Design Storyboards

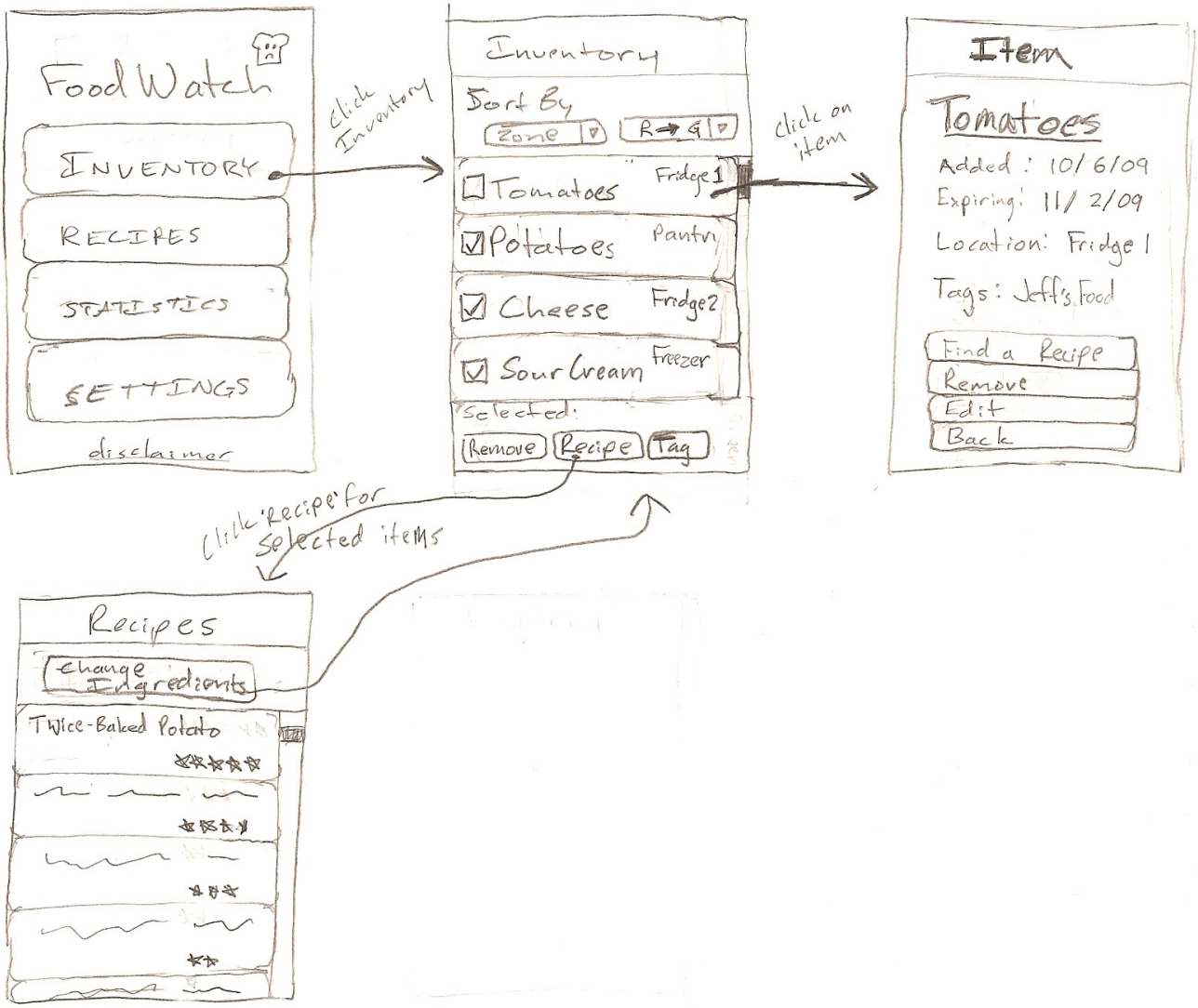

Figure 1

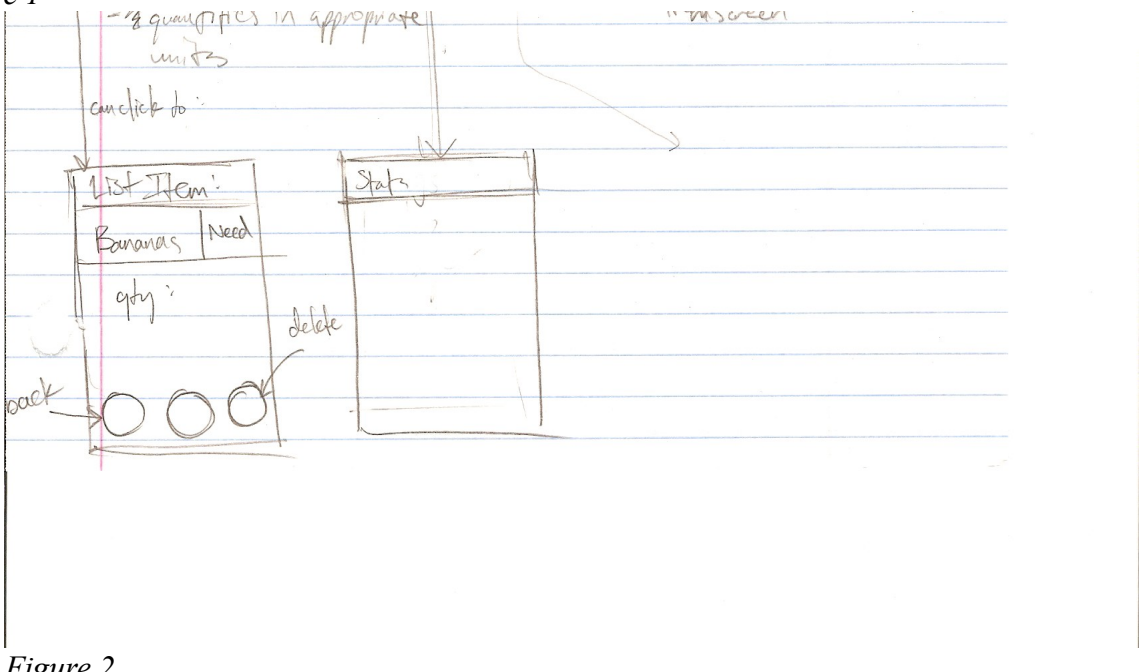

Figure 2

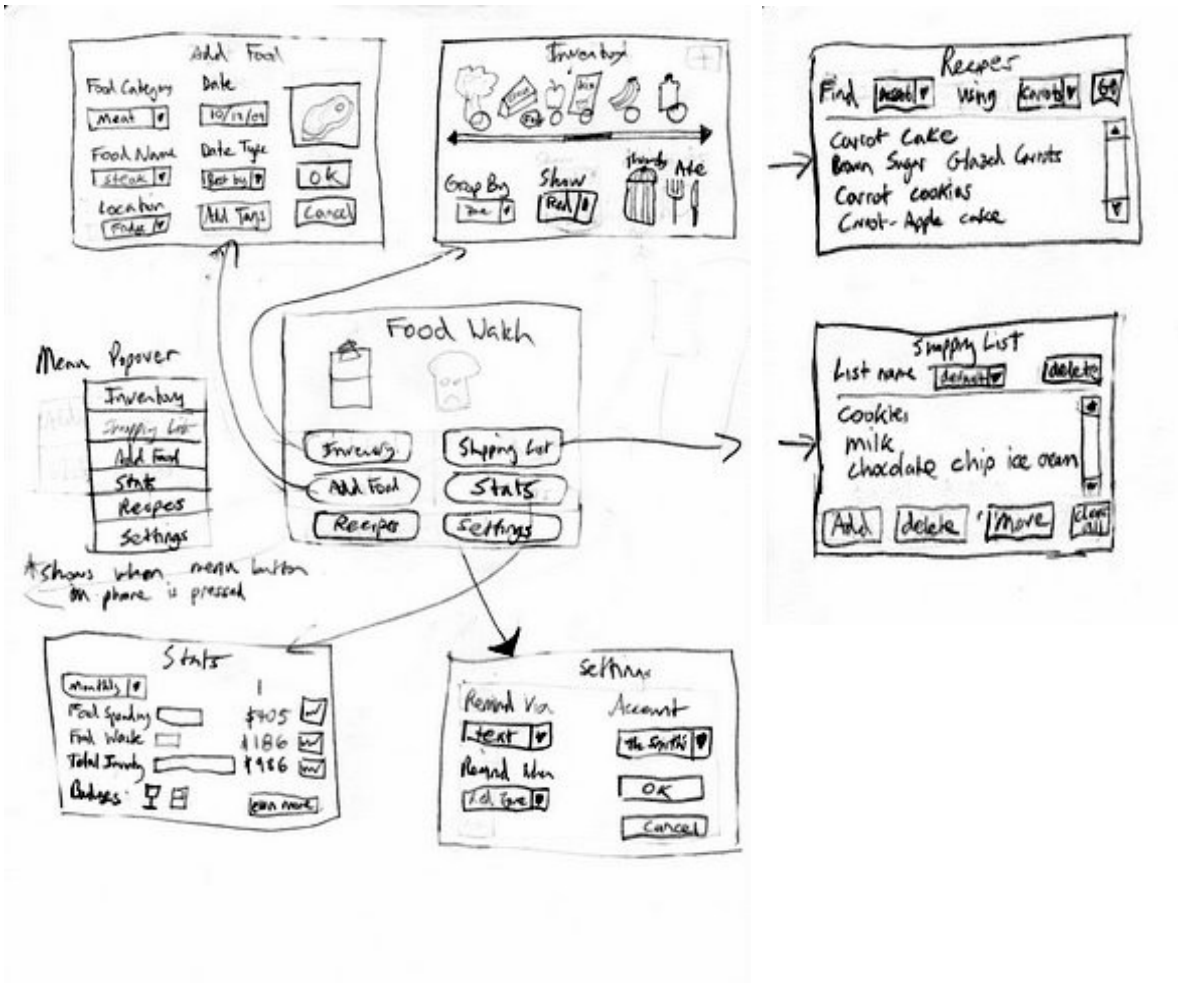

Figure 3

Selected Interface Design

Which Design and Reasoning for Choice

After exploring three different design options, we decided to go with the simplest, most conventional design. The limited screen size of our target mobile device made fancy design elements, such as "cover-flow" to show the food inventory, impractical. Additionally, our application is targeted for a wide audience, including those with little mobile application experience and therefore needs to use only simple design elements. These design elements, such as buttons and scrolling lists, are familiar to all customers and are instantly understood and manipulated by them. Because the UI elements are instantly understood, the customer can quickly navigate through the application and the less time the customer has to spend using the application, the more likely they are to use it on a regular basis.

This idea is the driving force behind many of the other design choices we made. For example, we are going to include a small icon next to each food item in the inventory, giving the customer a quick visual reference for what is in the inventory than reading lines of text. Also, the application will feature a navigation menu that can be pulled up at the touch of a button, allowing the customer to jump anywhere in two clicks.

In summary, after experimenting with different UI concepts and designs, our final result was a simple, conventional interface. This simplicity, combined with other UI enhancements, aims to make the application as efficient and quick to use as possible. After all, most customers are already too busy and do not want another

time burden to deal with. It is our goal to make this application easily fit into their lives so that it can start to be used and start to solve the huge problem of wasted food.

Customer Interface Description

Main Screen (fig. 1)

Functionality

- links to other screens

- condensed into a pop-up menu accessible at any time

Usability

The interface starts at a main screen that presents the customer with buttons linking to the main sections of the program, such as inventory, recipes, or statistics. Instead of having to return the main screen later, there will be a pop-up menu (fig. 3, left side) that is accessible any time while using the program and will have all of the same links.

Inventory Screen (fig. 1)

Functionality

- lists all food, able to be sorted (freshness zone, location, type, etc.)

- multi-select to perform batch operations: edit tags, deleting, related recipe finding

- view individual items, see item details, tags, expiration date, location

Usability

The most used section will probably be the inventory, where customers will manage the list of food items that are tracked by the application. The main screen of the inventory section is focused around the list of items in the inventory. There are a few ways in which the customer may proceed, depending on the task. One is to sort the inventory; it may be sorted by many of its properties: freshness zone (the default), location, category, tags, etc via a couple of dropdown menus at the top (fig. 1). Another way to proceed is to simply click on an item, which will bring the customer to a full screen detailed view for the selected item. This view has a few links to allow the customer to modify any of the details, delete the item, find a recipe that uses it, or go back to the inventory.

Back on the inventory, each item can be selected or deselected for a batch operation with the checkboxes placed next to each item (fig. 1 & 2). These operations include tagging, deleting, and finding recipes that utilize some (or all, if possible) of the selected items. On deleting items, you will select whether the item spoiled or was eaten. You will also be asked if you would like to add it to a shopping list.

Recipe Screen (fig. 1, 3)

Functionality

- search for recipes using filters like difficulty, category, customer rating, ingredients

- find recipes that use ingredients currently in the inventory

- select a recipe, adjust the servings needed and then automatically add needed ingredients to a shopping list

Usability

Another significant piece of the application is the recipe section. This will display a sorted list of recipes that fit the current filters. The filters may be modified by clicking a link to the filters screen. These filters will be things such as difficulty, category, rating by stars (fig. 1), and ingredients. The ingredients section will bring the customer to the inventory, allowing them to select certain items that they would like to use in a recipe.

Going back to the list of recipes, it can be sorted by certain properties (rating, number of selected ingredients used, etc.) just like the inventory. The customer can click on a recipe to bring it up on the screen in full detail, where the directions and ingredients are all visible. The recipe can be saved and the ingredients that aren't already in the inventory can be added to a shopping list.

Shopping Lists Screen (fig. 3)

Functionality

- add items, delete items, or add a note to an item on a list

- multiple list support: create, view, edit and delete

- automatically created list showing food that recently spoiled or was used up

- move items between lists, clear all items from a list

Usability

The customer can manage these shopping lists in the shopping list section. It shows the names of the shopping lists the customer has created. When selected, the items are listed on the screen and can be modified, deleted, and notes about each can be added. There will also be automatically created lists showing the food that has recently been used up or spoiled, just in case the user would like to be reminded to buy these items again.

Statistics Screen (fig. 3, 4)

Functionality

- shows money spent, value of food wasted and total inventory value over a set time period (week, month, year)

- information can be graphed to show trends and encourage improvement

Usability

Financial and waste history can be checked up on in the statistics section. The statistics screen will list some interesting totals for the amount of money spent on food, amount of food wasted, and total food inventory value. It will have a graph button next to each to show a visual representation of that data (fig. 4). The time period can be switched (weekly, monthly, yearly) via a dropdown menu.

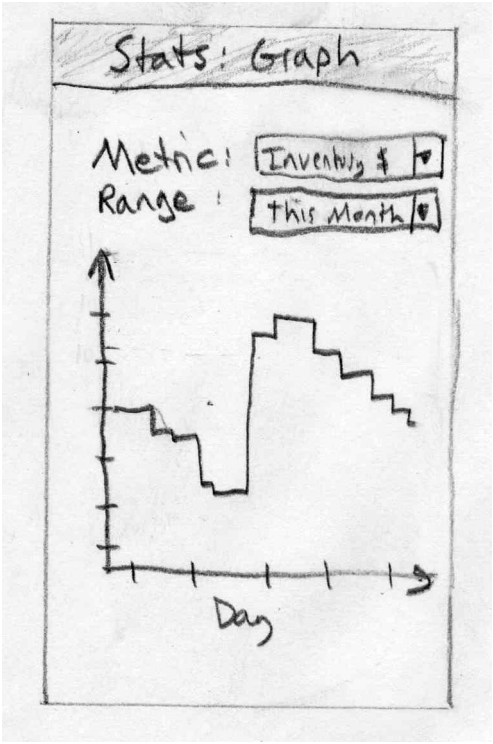

Figure 4

Settings Screen (fig. 3)

Functionality

- edit options such as notification behavior, visual settings, group name, clearing inventory

Usability

The last section of the app is the settings screen, which will be allow the customer to change notification behavior, visual settings, clearing the inventory, etc. through simple and familiar buttons and dropdown menus.

Scenarios

FoodWatch is fairly basic in function, so our core tasks haven't changed much from our Contextual Inquiry, we focused instead on exploring different ways to make accomplishing those tasks easier with our storyboards (different ways of representing the inventory and use of screen space).

Task 1: proper food storage

When the system sees food in an improper place for storage, or if it sees a way for the customer to store your food longer, it sends a notification to your phone. Since we're basing this app off of the g1, it appears as a built-in notification on the phone. The customer selects the notification, which opens up FoodWatch at the item screen in question. A dialog box informs the customer that their food (steak, for example) should be moved someplace better (to the freezer so it will last longer). If they are attempting to thaw the steak for dinner tonight, they can manually ignore the warning and it will not appear again. Otherwise, they hit the 'food moved' button and the dialog disappears. In the case of multiple items being poorly placed, it cycles through them all to prompt the user for action. When done, the application returns to the updated inventory.

Task 2: notification of food spoiling

To make sure that steak will still be good for a few days (after deciding to order takeout last night instead), the customer can check its status in their inventory. To get a broad overview of their food, all they have to do is navigate to the inventory one of two ways. The first is through the main screen's button to move directly to the inventory, and the second is via the pop-up menu that goes to any of the major screens of our app no matter what screen is currently active. Once in the inventory, an overview of all the food being stored is shown, along with color-coded 'freshness zones' (red, yellow, green) to give a quick visual indicator of what is still good and what should be used fast. If the customer wants a specific expiration date, they can select the item to go to the item's detail screen and view it there.

Task 3: planning shopping around inventory and recipes

One of the key things we noted in preventing food waste was purchasing food in appropriate quantities, and not overbuying food. While shopping, our application doubly aids this. First, the customer can quickly move between the inventory and a shopping list with the pop-up menu to check that they don't already have something on the list. Secondly, they can select any (reasonable) number of items in the inventory and go straight to a list of recipes containing those ingredients. With this, using food back at home is encouraged over buying lots of new food to make a meal.

a. How did we decide on our scenarios?

We had a lot of ideas when starting out about neat things to add to the application, but these three tasks directly address our core problems: food ends up spoiling because people either buy too much and can't use it up, or they forget about it until it's too late. We addressed these with simple reminders and an easily accessible view of all the food you own, along with relevant information. Recipe listing is an added incentive to use up food you've already purchased, and an easy idea to

couple with checking your inventory.

b. New and interesting techniques?

To get a wide variety of ideas, we took our storyboarding in three very different directions to look at different possibilities. In the storyboards above, we focused on different ways of representing the inventory, namely as a wheel (fig. 2) for easier navigation with a thumb, and as a scrollable list with icons to drag to actions (fig. 3) for easier visual recognition of individual items. These are definitely interesting ideas, but we settled on using a basic list as it's something more people will recognize and be comfortable interacting with, especially since we want to include batch operations on items.

c. What was difficult?

It was difficult at first to decide how we wanted to organize the information visually, and so we went three very different directions in the storyboarding process. The result of this, however, was a lot of contrast between how the interfaces would look and work, and we made a pretty solid decision based on that.

d. What worked well?

Exploring three very different avenues in design worked very well in the end, as we were able to look at our problems and easily decide which critical pieces of which design would address them in the best way. Referencing our task analysis and our contextual inquiries while looking at the designs made it very clear that simple and quickly navigable designs were the easiest, while also maximizing visual information.