The University of British Columbia

L o c a l l y

Project Requirements, Vision, and Scope

Angy Chung

Anna Gudimova

David Jung

Mo Kiani

Andy Lin

Alena Safina

September 28, 2016

Table of Contents

Product Vision Statement

Locally will allow users to view locally grown produce being sold at farmer's markets, check what's in season, and view nearby vendors. The data will come directly from the vendors themselves as they can update their daily produce stock through our application. It differs from other solutions as it is a mobile application which allows convenient, easy, on-the-go checking and updating.

Problem Statement:

The problem of	Not knowing what is in season or where to purchase local produce if/when it's in stock
affects	People who wish to live a healthy lifestyle
the impact of which is	Having more barriers, hassle in finding local, healthy foods
a successful solution would be	A mobile platform where consumers can browse produce at nearby local farmers markets and pick out healthy alternatives

Product Position Statement:

For	Android users
Who	eat healthy/organic foods or sell them,
Our system	is a software mobile application
That	will connect consumers and vendors of locally grown produce
Unlike	other solutions which are mainly web-based
Our product	is a user-friendly mobile platform that is portable and easily available on the go whenever customers or vendors wish to use it

Project Overview

Users: The users of Locally fall into two categories: consumers and vendors.
Consumers*:* Consumers are members of the general public choosing healthy lifestyle and supporting local food producers. Their technology experience may vary; however, our mobile application will not expect a high level of technical expertise. It will have a user-friendly interface and should be straightforward for any smartphone user to navigate. Consumers will

want to use our application to view nearby farmer's markets and their stock and a list of in season produce.

***Vendors**:* Vendors will be farmers/vendors from the Greater Vancouver area selling fresh locally grown produce who operate at farmer's markets. They may be required to have a higher level of technical experience, but only in terms of managing their vendor account and updating their stock and business information. We plan to minimize the technical expectations required of them by providing a simple to use GUI for managing their account information. Vendors will want to use our application to directly reach their customers by providing information about their business and products for sale.

Feature List:

- Search bar for users to search a certain type of produce or vendor name
- List of vendor accounts/profiles displaying information about a particular vendor, including information like market hours, available daily stock, location etc.
- Calendar to display dates, locations, and times of upcoming markets with links to vendor profiles and location of markets on the map
- Map that displays all markets currently open
- List of all produce that is currently in stock (for all vendors) at the time and when a certain produce is clicked on, a list of vendors that sell this produce will be displayed

Constraints:

Our product will run on the Android system and therefore one of the major constraints is that our users and vendors must use Android in order to interact with our application. This means that iOS users will be unable to access our application at the moment but there may be plans for iOS support in the future. We will be using Google Maps API for displaying currently available markets and thus our project will have a dependency on their API. Our product will also be relying on Amazon Web Services so we will be constrained by their policy for free services in the areas of storage, time, and amount of GET and PUT requests.

Scope and Limitations:

Considering our time constraint of eleven to twelve weeks, we have decided to limit the scope of our Android application to include users and vendors in the Vancouver area only and there will be room for expansion in the future if necessary. Given the time constraint, we will not be implementing features that notify the user of nutritional values of the food they are searching, although this may be an additional feature in the future. Another feature that we currently will not be supporting but can be implemented in the future is user accounts to store user preferences and comments on vendors and specific produce.

Assumptions and Dependencies:

We are planning to use AWS' DynamoDB to store information about vendors, markets and items in stock. The reason we decided on DynamoDB was that it is a non-relational database and has an excellent Android SDK. Furthermore, load balancing and growth are handled by Amazon.

Use Cases

Use Case 1: Enter New Stock Item

Identification: The vendor enters a new stock item by selecting it from a list the system displays.

Primary Actor: The Vendor

Stakeholders and Interests:

- The Vendor: The vendor wishes to add an additional stock item for sale for their business in order to inform and attract their customers.
- The Customer: The customer wishes to view what product a vendor currently has in stock or search where/whether a specific produce item is being sold.

Preconditions:

- The vendor must have a registered vendor account/profile and must currently be signed into the system - this will give them access to personal stock and permission to update it

Postconditions:

- The new stock item will be displayed on the vendor's page.
- If a customer searches for that stock item, the vendor's page will be shown as an available result.

Main Success Scenario:

1. The vendor tells the system to enter a new stock item.
2. The system displays the "enter stock item" screen to the vendor. The screen contains a dynamic list of fields
3. The vendor searches for their stock item within a field.

4. The system displays a list of matching stock items to the vendor, using a scatter-gather technique to display the suggestions.
5. The vendor selects their desired item. New empty fields are automatically added if the vendor has completed previous.
6. Steps 3) to 5) may be repeated for additional stock items.
7. The vendor completes entering stock items.
8. The system updates the vendor's stock information.
9. The system displays a request completed message to the vendor.

Extensions and Alternate Flows:

Alternate Flow: Create New Stock Item

1. If at line 5) of Enter New Stock Item, the vendor cannot find their desired stock item, then:
2. The system will prompt the vendor to create a new item.
3. The vendor will enter in the item's information.
4. The vendor completes entering the information.
5. The system creates the new item.
6. The use case restarts at line 5) in Enter New Stock Item.

Open Issues:

- Reconsider how the system should handle finding no matching stock item for the vendor.
- How the system will handle duplicate/erroneous entries
- Should the system also allow the vendor to specify item quantity?

Use Case 2: View Nearby Vendors

Identification: The consumer searches for nearby vendors in their area.
Primary Actor: The Consumer
Stakeholders and Interests:

- The Consumer: The consumer wishes to find a list of nearby vendors in their vicinity
- The Vendor: The vendor wishes to have their name to appear in the search results

Preconditions:

- The system has records of vendors and their locations

Postconditions:

- The consumer has viewed vendors nearby them

Main Success Scenario:

1. The consumer tells the system to find nearby vendors.
2. The system determines the consumer's location.

3. The system displays a list of vendors close to the consumer's location.
4. The consumer selects a vendor to view more information about them.

Extensions and Alternate Flows:

Alternate Flow: Handle Turned Off Location Services

1. If at line 2) of View Nearby Vendors, the consumer does not have their location services turned on, then:
2. The system will prompt the vendor to turn on their location services
3. The consumer turns on their location services
4. The use case restarts at 3) in View Nearby Vendors.

Open Issues:

- How many vendors should be displayed to the user
- What vendor information will be displayed
- How the vendors list should be filtered or displayed (e.g. map, text list, calendar)
- Adding a distance range to the search filter
- Should consumers be able to add favourites (may require consumers to have accounts)

<u>Use Case 3: Search For Stock</u>

Identification: The consumer searches for a specific stock item being sold by nearby vendors.
Primary Actor: The Consumer
Stakeholders and Interests:

- The Consumer: The consumer wishes to find all the locations where they can purchase the item they search for.
- The Vendor: The vendor wishes to have their stock item and name appear in the search results of the customer if they have listed that item in stock.

Preconditions:

- The system has records of vendors, their locations, and the items they have listed as being in stock

Postconditions:

- The consumer has viewed vendors that sell the item they searched for

Main Success Scenario:

1. The consumer tells the system to find a specific item being sold.
2. The system displays a list of vendors that sell that item.
3. The consumer selects a vendor to view more information about them.
4. Steps 1) to 3) can be repeated if the user wishes to search for additional items.

Extensions and Alternate Flows:

Alternate Flow: No Vendors Found

1. If at line 2) of Search for Stock, the system cannot find any vendors selling that particular item, then:
2. The system will alert the consumer of having found no results.
3. The system will suggest other items to search for.
4. The consumer selects a suggestion or searches for a new item.
5. The use case restarts at 2) in Search for Stock.

Open Issues:

- Should we have a “go back” button so that user could return to the search results after they see a vendor’s profile?

Use Case 4: Check the Calendar

Identification: The consumer checks the calendar to see on what dates farmer’s markets will occur
Primary Actor: The Consumer
Stakeholders and Interests:

- The Consumer: The consumer sees the calendar to check the upcoming farmer’s market events, as well as their locations, vendor participants, and produce sold there
- The Vendor: The vendor wants their name and stock items to appear in the farmer’s market event

Preconditions:

- The system has records of farmer’s market events, their dates, locations, vendor participants and their stock items

Postconditions:

- The consumer has viewed farmer’s market dates, locations, vendor participants, and stock items to be sold

Main Success Scenario:

1. The consumer clicks on the calendar icon in the app.
2. The system displays a calendar with Farmer’s Market event links.
3. The consumer clicks on the link for a specific event to see the event time, location, list of vendor participants and items they sell.
4. Step 3) can be repeated to check more dates.

Extensions and Alternate Flows:

Alternate Flow: No Farmer Market events added due to lack of vendors

1. If there are no official vendors ready to enter Farmer Market into the calendar our team will have to create vendor accounts and fill the calendar with primary data

Open Issues:

- How do we identify genuine vendors?
- Should every vendor have the permission to add farmer’s market events to the calendar?
- Should we allow for vendors to delete farmer market events from the calendar? What if those events already have vendor participants assigned?