Mobile Computing

4005-772-01

Milestone 5

Final Report

Pratik Rasam (psr5529@rit.edu)

Himanshu Kale (hsk5260@rit.edu)

Pankaj Deshmukh (pbd6595@rit.edu)

1. <u>Motivation:</u>

Due to the increased use of smart phones, location based services are becoming more popular. “What’s for Sale” has been inspired from successful neighborhood marketplace apps like EggDrop & YardSale. For a faster search a ‘one touch’ approach & quick registration for searching was required. Especially for frequent movers and students, there is a need for a handy tool to buy or sell goods in locality.

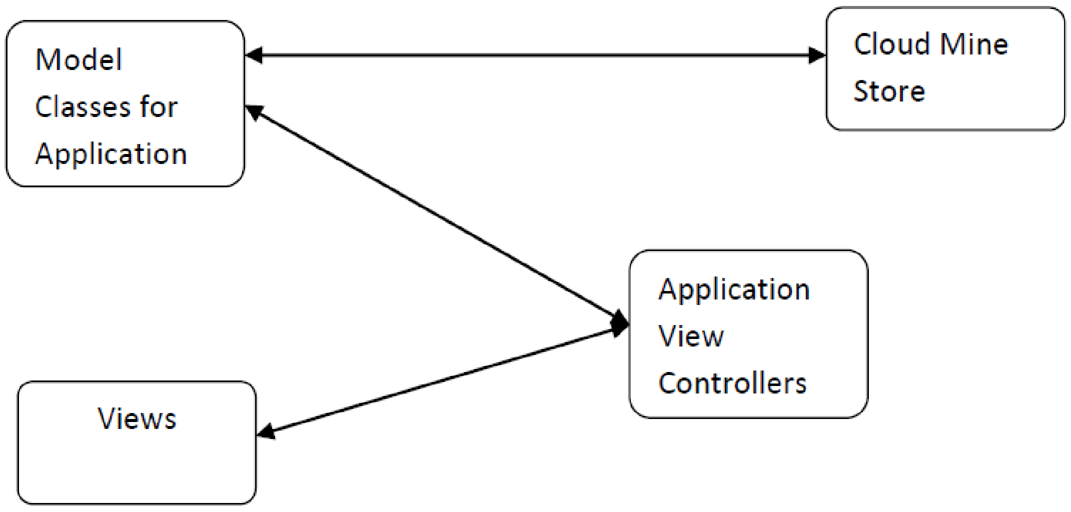
What’s for Sale focuses on all these features and intends to provide users the best deals nearby the user locality.

2. Overview:

"What's For Sale" is a neighborhood marketplace app that enables user to buy and sell items in neighborhood area. The main intention of the app is to create an interface between local buyers and sellers. The app focuses on location-based service which can be beneficial for the users. The user can use this app to buy products which are locally available to the user. In this case, the user becomes the buyer where it can focus on searching for the product of interest and contacting the user. Also, the user can also sell off items. In this case the user becomes the seller where it can post the item to sell by providing appropriate credentials and item details. The app also provides functionality to view the posting on map. The map view enables the user to view the location of the user posting the app. Every user is provided with user account where user must provide authentication information (username and password). The app however does not cover any portion of transactions between buyers and sellers as the app, as mentioned earlier, is just an interface between buyers and sellers.

3. Design & Architecture:

We build the app on iOS platform and the app user interface conforms to Apple standards. Looking at the app requirements, we decided to implement pub-sub architecture as our distributed feature. The backend for this architecture is CloudMine, a backend service, which is capable of handling distributed features we intend to implement. The following shows the design architecture:



1 . Model Classes for Application

Following are the model classes:

1. Session

2. User

3. Listing

Session:

This is the primary class for setting up the user login session . We use a singleton object shared which is global throughout the application . The session class has a current user object which is instantiated to the user who is currently logged in . Thus we get an access to currently logged in users attributes and methods throughout the class.

User:

The user class sets up basic information for the currently logged in user. This information includes fetching user email address and other attributes like location.

Listing :

The listing class stores the basic properties for a listing of a product like product name and product price and the product.

2. Views:

The Views contain all the UI views representing every action in the application.

3. Application View Controllers:

The application view controllers act as an interface between the model classes and the views

4. CloudMine:

This is the distributed component of our project which handles all the listings, user objects and media files. CloudMine does not provide image files to be stored within an object, thus we designed a naming scheme to name the files associated with each user and its particular listing. We use this naming scheme to fetch the particular image according to the user and listing requested from CloudMine. We utilize the methods provided by CloudMine API to interact with our object on CloudMine.

Distributed Feature:

We intend to implement **"content based pub-sub"** architecture. As we require the users to view only desired listings, we did not chose topic based pub-sub, also the user does not belong to a specific type, which rules out topic-based pub-sub. To implement this, we used CloudMine backend. CloudMine provided us with centralized server where users can fetch objects. As the users are both buyers and sellers, we require isolating items of interest for each user and must take care that no other user will be able to update any other user’s listings. CloudMine provides time-decoupling due to its centralized nature.

4. Detailed Implementation:

The app implementation is divided into 5 phases for its development lifecycle. The following explains about the implementation for each phase:

Milestone 1:

For milestone 1 we analyzed existing apps similar to our application idea. These were namely EggDrop, YardSale and Smoopa. These apps were analyzed on various parameters which helped us determine the appropriate app to develop.

Milestone 2:

For Milestone 2, we implemented user authentication with CloudMine as the backend. A new user can be registered by entering email address and password following which he will be directed to the main view controller.

2. The User location is also extracted using CLGeolocator , CLLocation and CLLocationManager.

3. We also implemented the “Sell” option where in the user can create a new listing my clicking on the post item. The user enters the listing name, date, price and an image for the product.

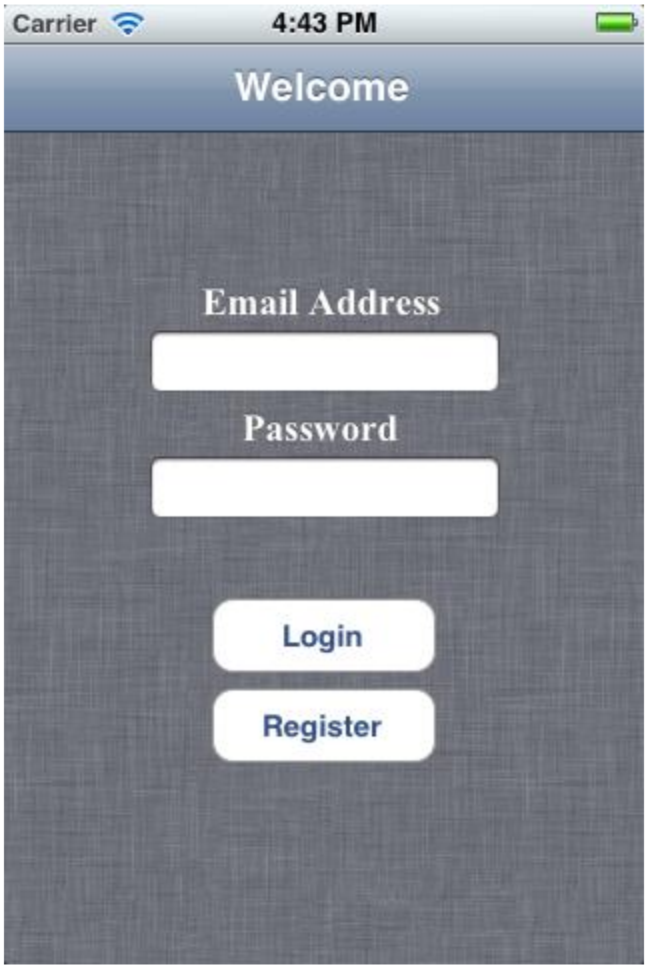
4. For implementing the Camera feature the image can be taken by the user using the UIPickerView and the listing object along with the listing picture are loaded to CloudMine.

5. We created icon for the app:

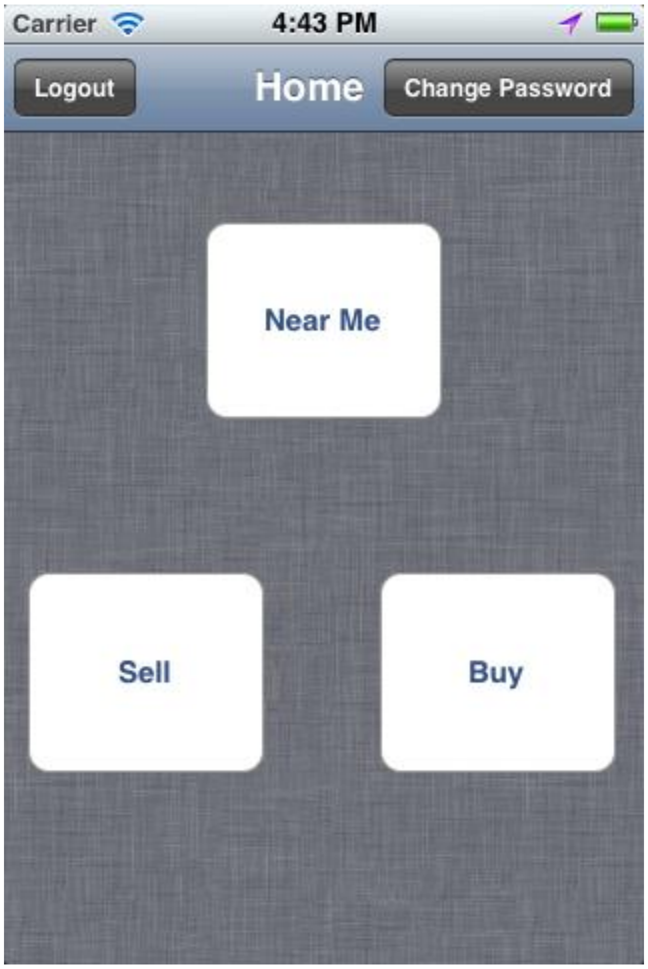
What's for Sale

6. Since the simulator cannot use the camera, we have made the provision that if the camera is not available, the user can pick an image from the image gallery. If no image is selected, item cannot be posted.

7. We started working on the distributed feature for the app.



Screenshot of login view



Screenshot for Home page view

Milestone 3:

For Milestone 3 we achieved the following:

1. On successful login the application loads user specific parameters like user postings.

2. When user clicks on “Near Me” all the listings near by the user will be displayed on the map. Every listing will be denoted as a pin Annotation on the Map. The values of the coordinates the dynamically loaded from Cloudmine from the Listing object. The configuration of the Map is automatically set based on user’s location which we had already implemented in the previous phase. The application logic is well implemented such that the user will not be able to view own listing and view only other user’s listings.

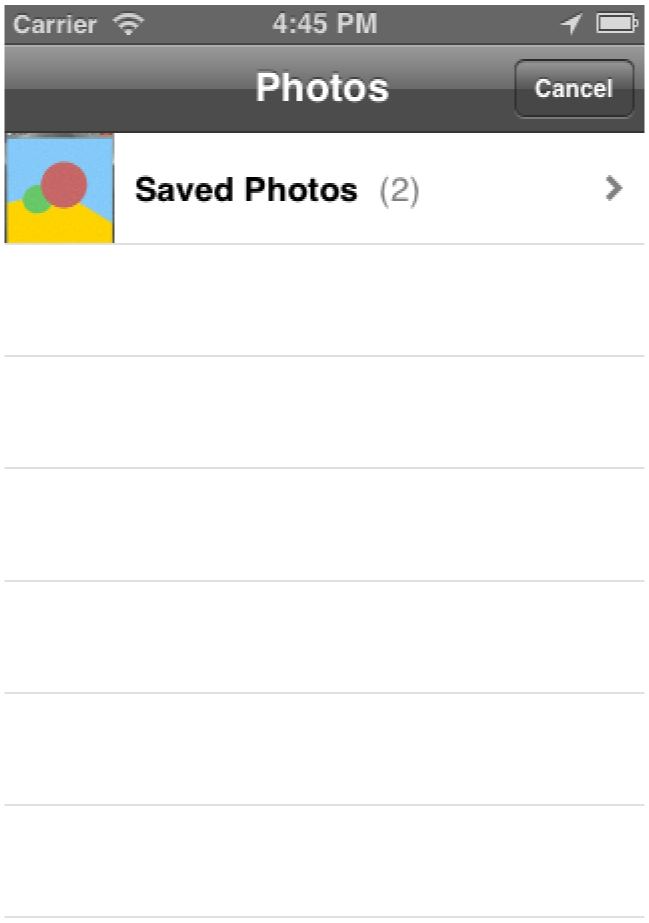
3. The distributed feature of isolation between buyers and sellers is also completely implemented. This could have been done easily using Access Control Lists provided by CloudMine. But due to some unrecognized error, we were not able to use ACL’s. Thus, we decided to filter the objects while fetching them to our app. The approach works well and we achieve complete isolation from other users.

3. When the user clicks on “Buy”, the user can search for any item in neighborhood. The user can view the details of the listing on a table view. The user also has an option of locating the listings on Map where all filtered listings are positioned on the map.

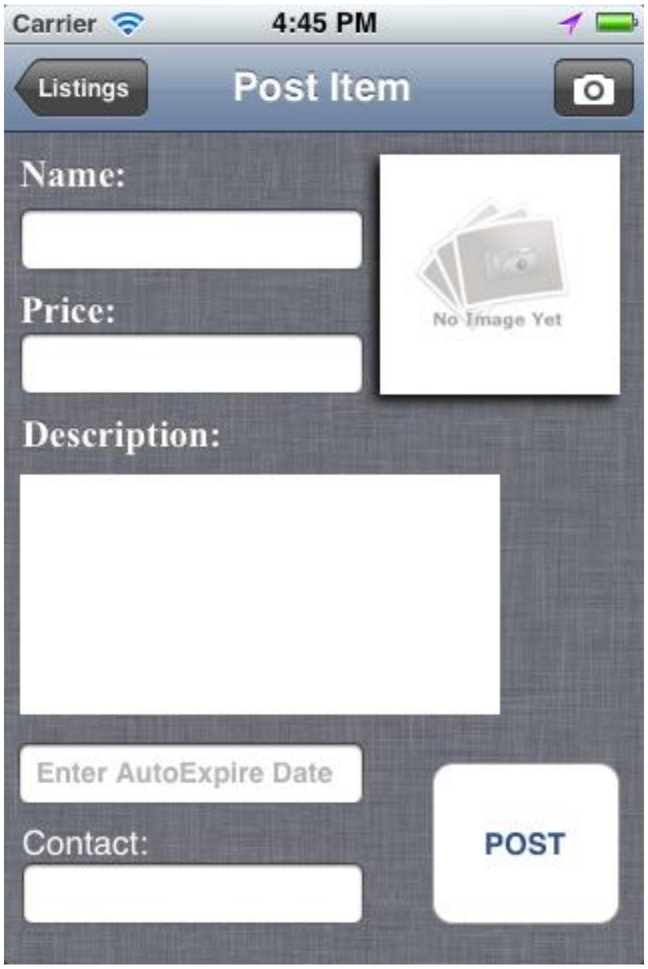
4. Also, we implemented the feature of dynamically loading images for particular listing from Cloud mine. Care has been taken so that the application does not continuously poll the Cloudmine server and consume bandwidth.

5. A user can also update its listings and post the listings to Cloudmine.

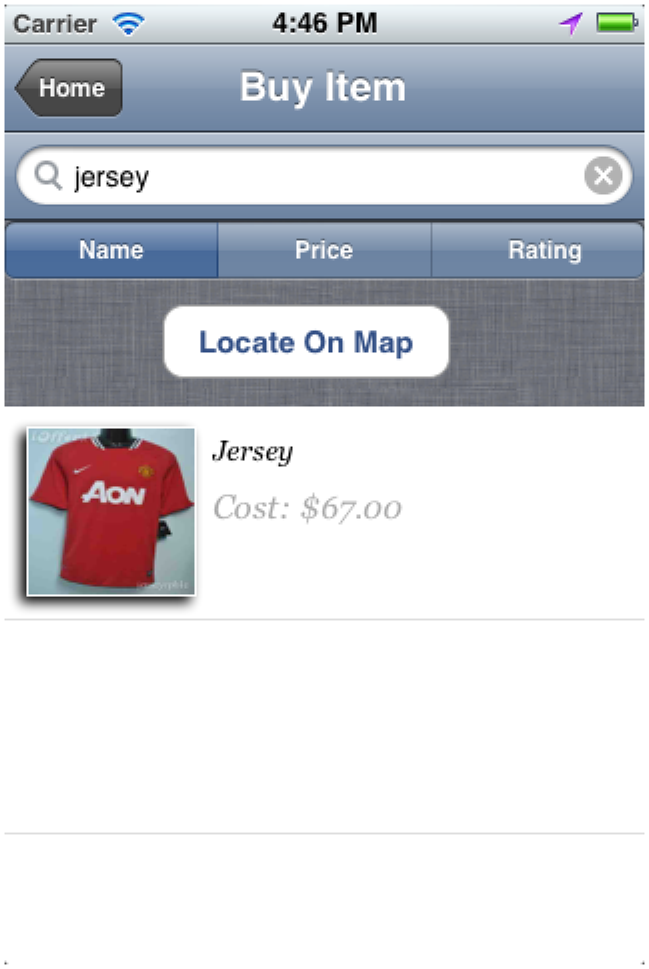
Following are screenshots for implemented features for Milestone 3:



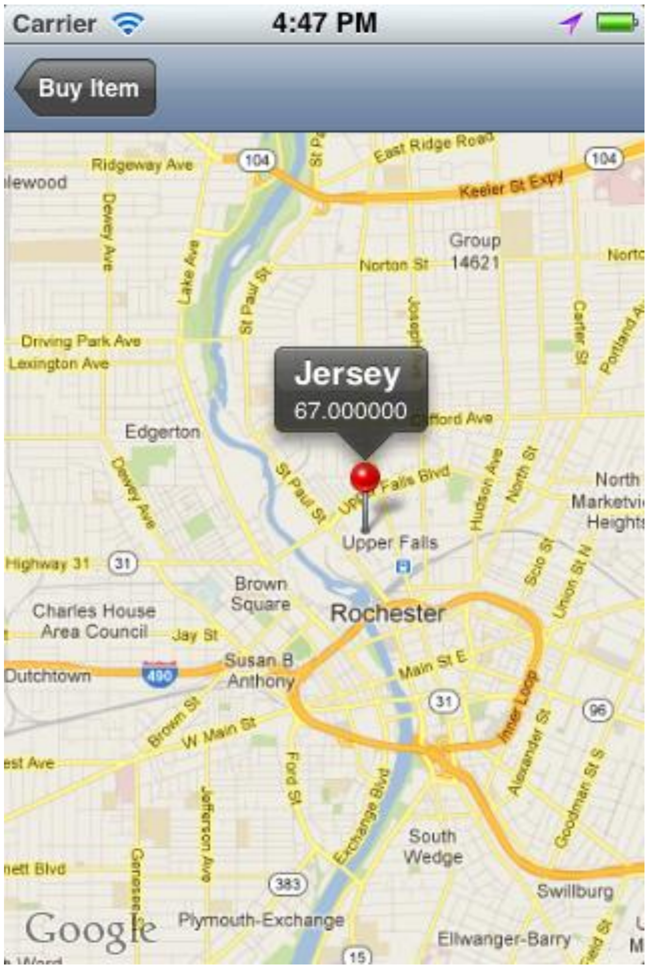
Screenshot to select photo from photo library



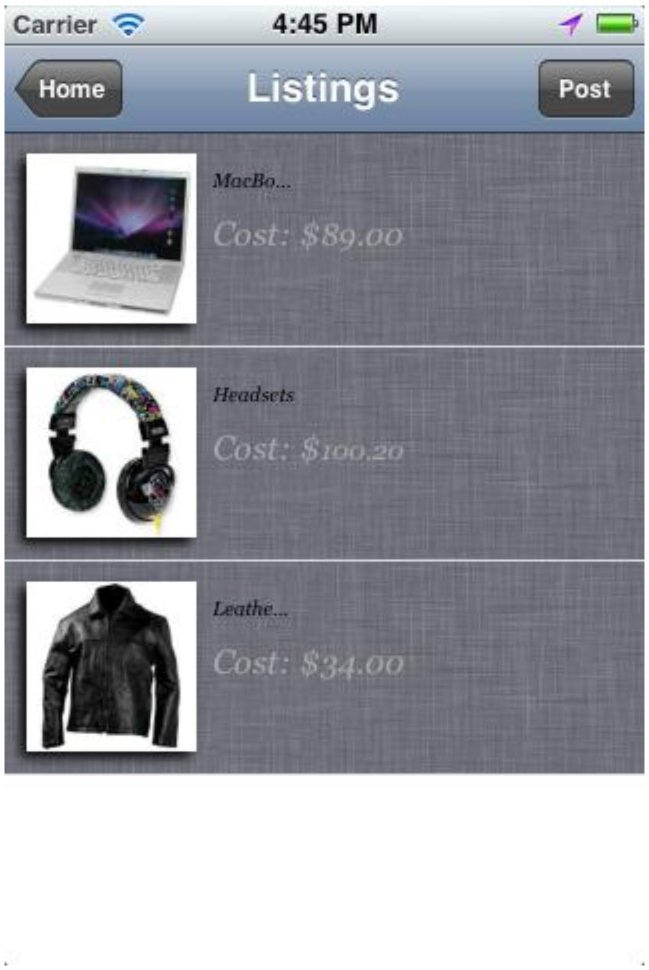
Screenshot for Posting Item



Screenshot for Buy item view



Screenshot for Map View



Screenshot for listing view

Milestone 4:

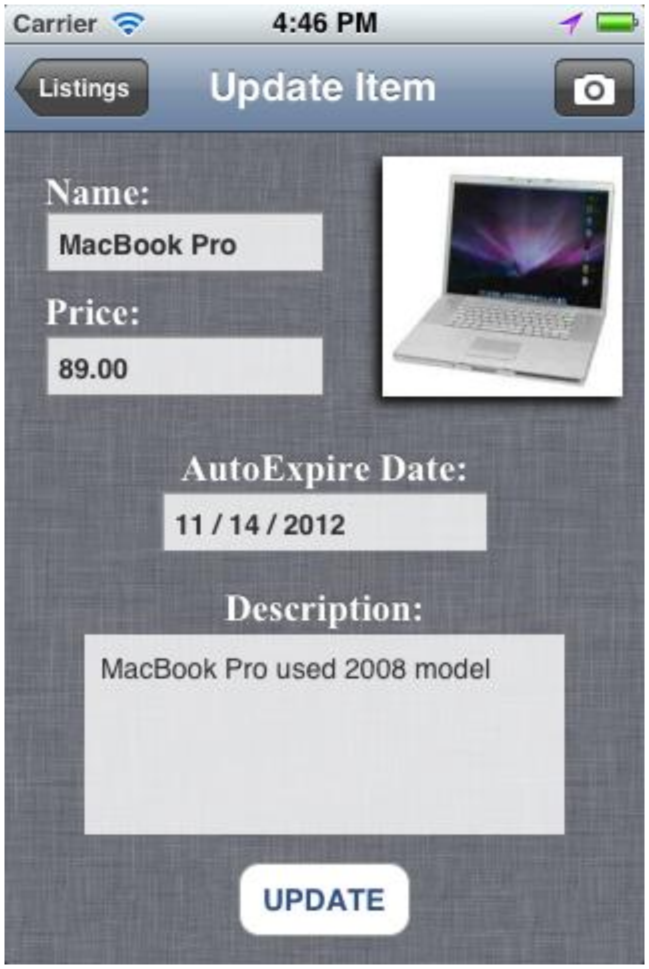
For Milestone 4 we addressed and achieved the following goals:

1. Provisioning a user to contact the seller. We provided a control to call up the seller from our app.

2. A user can remove a listing which will be also removed from CloudMine.

3. Improving the user interface.

Following are screenshots for Milestone 4;



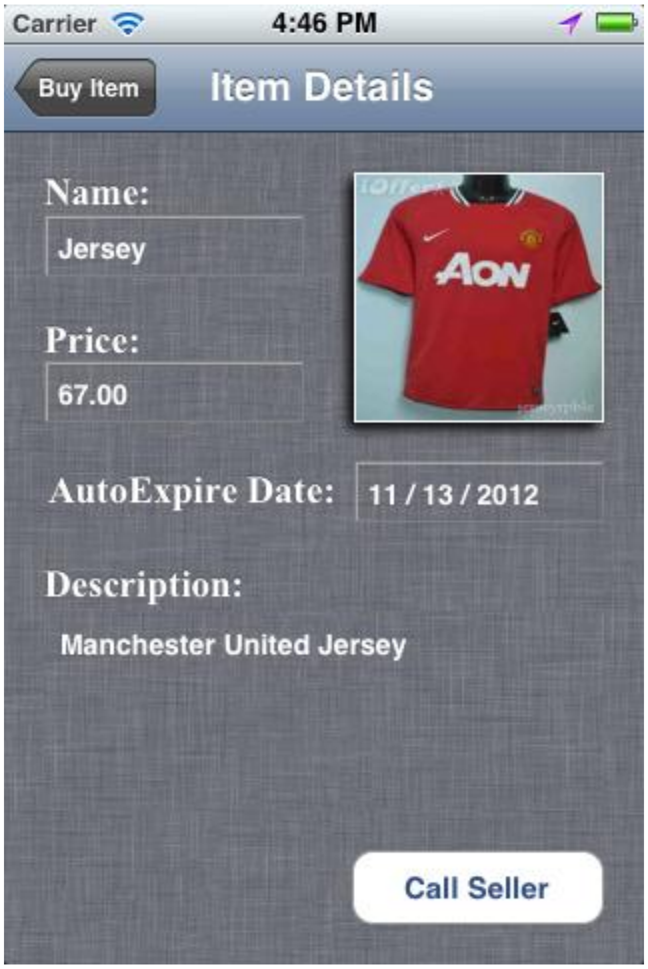
Name:
MacBook Pro

Price:
89.00

AutoExpire Date:
11 / 14 / 2012

Description:
MacBook Pro used 2008 model

Screenshot for updating item



Name:
Jersey

Price:
67.00

AutoExpire Date: 11 / 13 / 2012

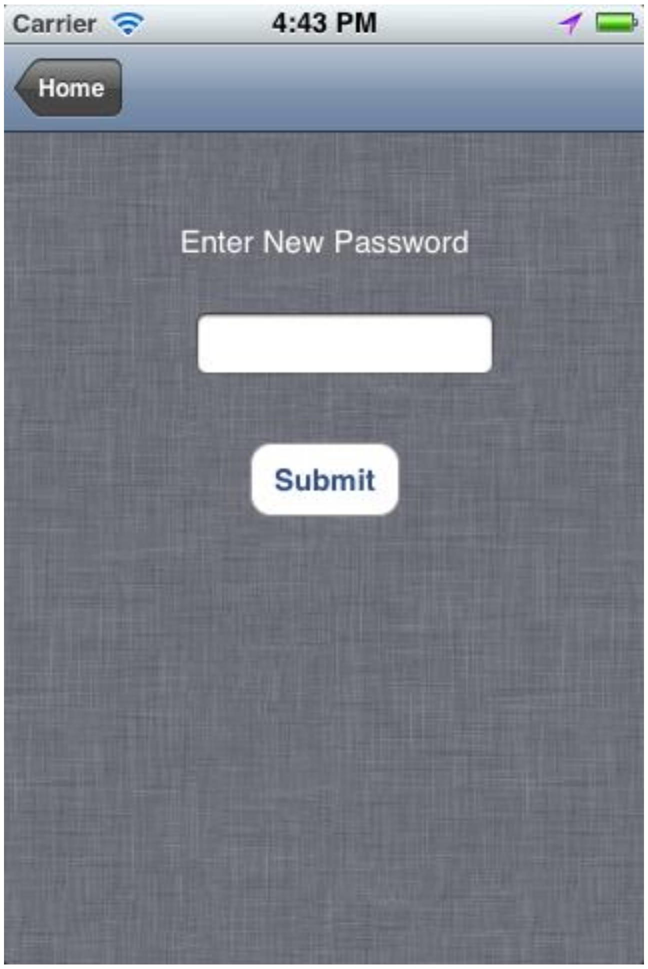
Description:
Manchester United Jersey

Screenshot for detailed view

Milestone 5:

Milestone 5 was patching up phase for our app where we managed to document the code and fix minor issue. One of the important fixes:

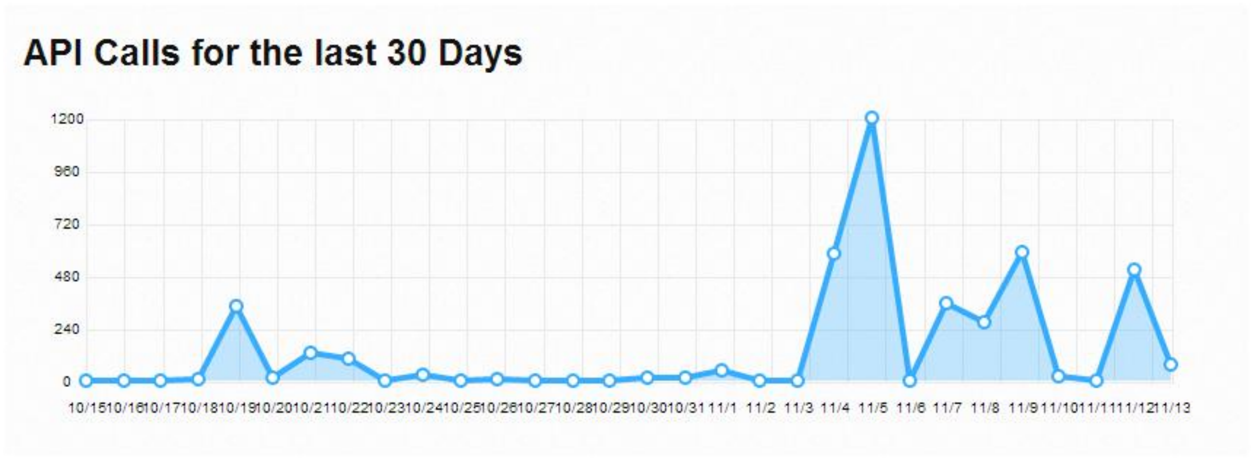
1. Fixing the location of the user if not set by the simulator. The issue was that simulator would set the location of the user at (0,0), this led the map to view show middle of the pacific ocean. This is due to default location set by the simulator when it is not simulating the location. Thus, we provided a default case where the location of the user is set to Rochester, NY if not simulated by the X-Code simulator.
2. Validation testing and fixing minor bugs.
3. Documenting the project and commenting the source code.
4. Analytics using CloudMine’s Analytics feature.



Screenshot for setting new password

5. <u>Analysis:</u>

The core of the analysis is based on the frequency of the API calls made to CloudMine. This would determine the performance of our system. The following is the number of API calls made within last 30 days to CloudMine using CloudMine's analytics feature for our app. We do not consider the entire set to be conforming to our analysis as majority of the calls were for testing purposes.



As seen from the figure above, the calls are too frequent than optimal calls expected in the later part of the timeline. This is due to non-optimal implementation of calls to API. Our initial implementation fetched images of the desired listings on each table view appear. We soon understood the issue and we prefetched the images in array and loaded updated list on each new post of loading the view.

6. Future Work:

The app has working core features and successful implementation of distributed feature. This app can be improved providing features like WishList and user ratings.

WishList:

WishList feature can store user requests and can provide notifications to the user when particular item is available. This can be a handy feature where the user can put items of interest which are currently not available. The app would be responsible to provide notifications to the user about the availability of the item. The design of our app would allow this modification to be implemented.

User Ratings:

The trustworthiness of a listing on the app can be tested using “User Ratings”. User Ratings would be an additional parameter to the user where other users can rate the user. This rating will be based on the quality of the posting. This parameter can be a metric of trustworthiness of the listing posted by the user. There can be catch for such feature. The user can use other users to improve its rating and thus gain enough trust points to sell fake products. Thus, there is a need for an algorithm that would take care of such cases. The design of user class needs to change for implementation of this feature.

<u>Prefetching the data:</u>

The performance of the app can be increased by prefetching the data. The prefetched data can be used to load the listings to the user in negligible time. The problem with this technique is the unnecessary polling to be introduced. This would increase the frequency of API calls to CloudMine server. Thus, we can search for a way to optimize this issue and improve the performance of the app.

7. Conclusion:

Motivation acquired from need of location based marketplace has led us to successfully develop “What’s For Sale”. The app has been designed and implemented in a manner to incorporate further modifications.