Final Report for MiddShare: A Mobile Application for Service Exchange and Free & For Sale

Birgitta Cheng, Shannia Fu, and Yanfeng Jin (Tony)

December 15, 2016

Abstract

This project is part of our Computer Science Senior Seminar curriculum. We have chosen to create an Android mobile application to optimize the Service Exchange and Free & For Sale platforms currently on campus. This mobile app has been successfully created to be a fully functioning Android app: MiddShare. Our hope is that we will implement it for campus-wide use in the following semester. This paper explains all the problems in the current platform and how MiddShare provides a simple solution to them.

1 Introduction

The goal of this project is to create a new mobile application that will fix all the current problems with the campus Free & For Sale platform. Currently, there is no mobile platform, and the Free & For Sale group on Facebook has several issues, which we will further address in our problem statement.

Our motivation for this project is three-fold:

- Improve the functionality of the Free & For Sale Facebook group
- Learn how to develop a complicated mobile app
- Learn to build an application with a back-end

This topic is important because Middlebury students usually need to buy and sell items and ask for help within the community. However, the only platform for these activities, the Free & For Sale group on Facebook, is highly inefficient. Therefore, we would like to move the Facebook group to a mobile platform and customize it for the needs of the Middlebury community. By improving and optimizing this platform, we can enable students to buy and sell services/items more efficiently.

Furthermore, most web services, such as E-commerce, music, social media, and maps, have moved towards mobile platforms. Twitter and Amazon are two examples that choose to create mobile applications even when their contents can be accessed through phone browsers. Since the Free & For Sale page is currently on the Facebook platform, we do not have as much freedom to customize it as we would with an individual mobile application. By creating one, we can choose specifically what we would like it to do and how it would serve the students. It is more convenient, especially for people who rely on their mobile phones all day. We should always be in line with the current technology, which now includes mobile platforms.

This project also serves as a great learning opportunity as there has not been a hands-on mobile application development class offered in the Computer Science department yet. This project will allow us to learn how to create a mobile application from start to finish, a useful skill to have when applying for jobs in tech.

2 Problem Statement

This project is focused on solving a problem of supply and demand: more specifically, how do we best optimize the Free & For Sale online forum to suit student needs throughout the school year? What we

currently have in terms of a Free & For Sale platform is a somewhat generic "Group" on Facebook–because this is our only option right now, we make comparisons with only its problems, which are detailed here:

- The platform is not separate from the entity of Facebook. This means that notifications received from the group are nested within other notifications, such as those from "likes" on photos and posts, comments on photos and posts, posts from other groups, and event invitations. In addition, as a Facebook group, Free & For Sale's notifications only show that someone has posted in the group, with no detailed information on what the post is actually about. Users have to go into the Facebook app to see what others are buying or selling. In short, it's extremely easy to miss a relevant notification from Free & for Sale.
- There is currently no way to sort through the posts if need be. The Facebook Free & For Sale group keeps every single request that has been posted, and in order of posting. If we wanted to see the oldest posts, the only way we could do so is to scroll all the way to the end of the requests.
- There are time sensitive posts on the current platform which are kept forever. For example, if someone is selling a ticket for something this coming Saturday, the post will still not be deleted after the show has finished, and will in fact stay there forever.
- The group doesn't have a location, or geofencing, feature. Applications like Yik Yak, for example, only show posts from a specific radius around the user to make the content more relevant. On the other hand, Facebook's Free & For Sale shows posts from all locations.
- People often do not mark their items as sold or bought after the interaction has occurred. This populates the Free & For Sale forum with several items and requests that have been fulfilled and rendered irrelevant. It makes it harder to filter for other users who still have requests and distracts users with posts they cannot fulfill anymore.
- The current platform doesn't regulate prices on items. Many people will post the price of their items as "$999,999,999" to draw attention to their item instead of using an actually reasonable price.

Besides addressing and fixing all of the problems listed above, our mobile app MiddShare further enhances the convenience of the Free & For Sale page. It's a more targeted way to address the problem of the Free & For Sale forum.

Deliverable: Our goal was to create a fully functional Android application that addresses all the problems with the current Free & For Sale platform. We have not only succeeded in creating such an application, but also polished its details so that the application looks professional and is easy to use.

3 Background and Related Work

Middlebury uses a Facebook group called Free & For Sale for students to buy and sell items to anyone on campus. It currently has 2088 members, and gains anywhere from 0-25 new members every week. There is currently one administrator for this Facebook group and he has already graduated from Middlebury College. To access the group, you need to request access and anyone who is already in the group can let you in. As mentioned in the problem statement, there are a number of issues with the current Free & For Sale page that we would like to solve.

To counter these problems, we want to create an application that pushes item-based notifications, gives incentive-based accurate pricing, shows relevant information, and deletes posts after the interaction. We also decided on making this idea into a mobile application so that notifications are more easily accessible as compared to if it were a web application.

We have looked into the novelty of this application idea. and found it to be unique, as there is no existing

mobile platform that is fully functional. Most importantly, because of the collegiate environment that has a strong community base, our application should function better than any Free & For Sale groups within cities or other forums. The initial idea was based on a previous mobile app called Pinch [1] which is a Service Exchange group within Boston. However, we believe our app will extend the initial services and will be built with the framework to be used on college campuses.

4 Methods and Implementation

We successfully created the mobile application using Android Studio and Firebase. Here is a description of the tools we used and the implementation details of our application.

4.1 Software

We used Android Studio to create our application. This is an IDE with many built-in tools specifically created to help developers create Android applications. We used the Facebook SDK to streamline the log in process for our mobile application, which insures that people are not creating multiple accounts unnecessarily. We relied on the Internet and developer tutorials to aid us in the process as Android Studio had a steep learning curve. Fig. 1 shows the Android Studio IDE. It enables us to develop the front-end using XML and the actual functions using Java.

4.2 Back-end

For our back-end, we chose to use Firebase. It is a mobile and web application platform with a real-time database that stores information in JSON. Firebase allows us to store user information in the database and instantly synchronize all the changes with every connected client. For example, when a user adds a new item to sell in our mobile app, Firebase updates the information in the database. When another user's mobile application detects the change in the database, it syncs the change and updates the UI so that the user can view new changes. Fig. 2 shows our Firebase database, which stores a global list of requests, a global list of users, a one-to-many relationship between users and requests, and a one-to-many relationship between requests and comments on them.

4.3 Request list and Sorting

We implemented the request list with the list adapter provided by Firebase UI, which automatically populates the list in our application with the information retrieved from the Firebase database, and updates the list whenever the database is changed. Even though Firebase UI has greatly simplified the synchronization process, it has also posed difficulties for us when it comes to sorting. Firebase's list adapter does not support sorting or filtering on the client's side. Because of its high level of abstraction, we cannot send a specific query to the database to let it sort the results for us before passing the data into the adapter either. In order to implement sorting, we had to find the source code for Firebase's list adapter, and create our own adapter that inherits from Firebase's list adapter. In our custom adapter, we overwrote several of FirebaseListAdapter's original functions in order for it to be able to sort the displayed list based on different criteria.

4.4 Time limit

One of the most important features of our application is the function to set a deadline for a certain request. Taking different circumstances into consideration, we enable users to either select a date and time as the deadline (e.g. if the user needs a ride on Dec. 18th, which makes the request valid only before that date), or to set a certain period of time in which the request is valid (e.g. if the user needs some flour for baking in the next 2 hours). The user's input is then converted into the standard Unix time format and stored together with other information of the request in the Firebase database.

Whenever our application tries to pull the data from the database to populate the global request list, it checks if any of the requests' time limits has been reached. If so, the corresponding item will be deleted from

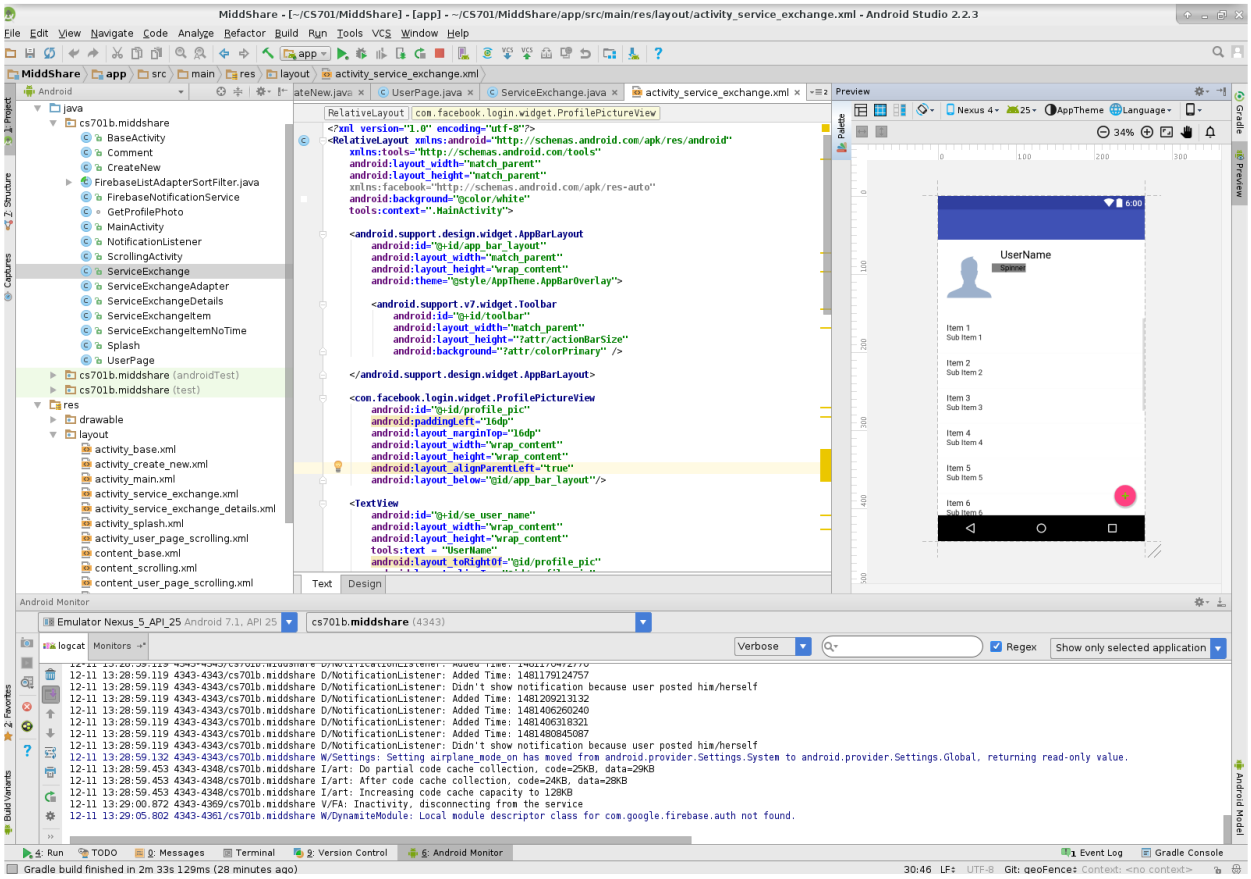

Figure 1: Android Studio interface with XML preview

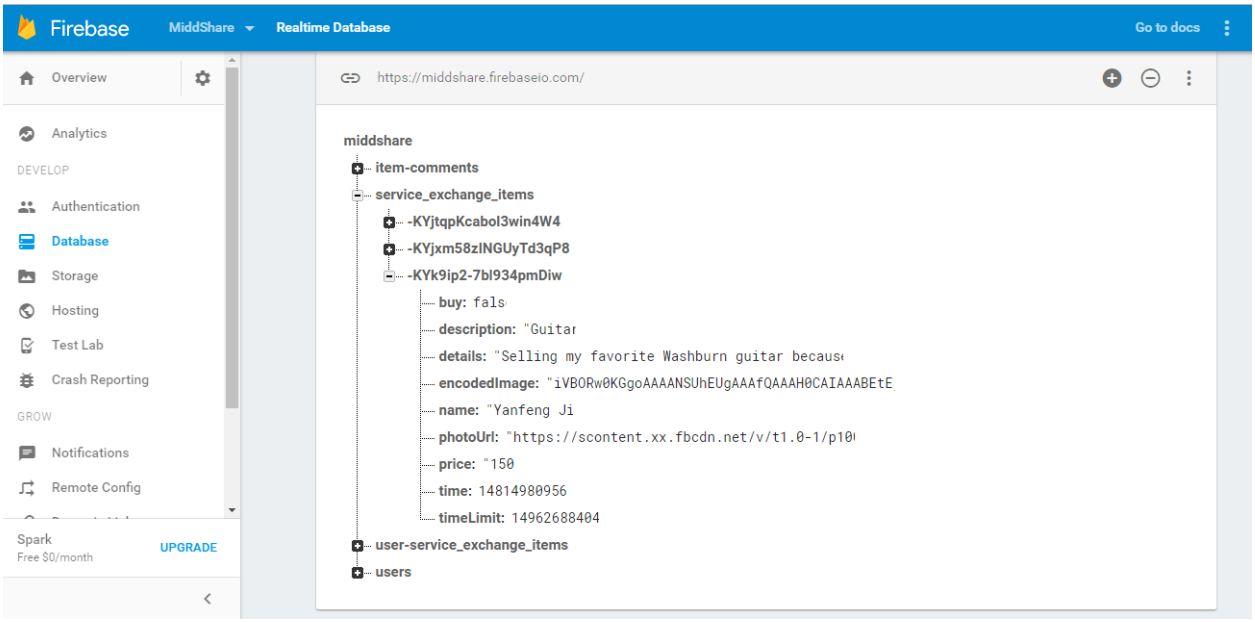

Figure 2: Firebase database

the global list in the database, so that it will no longer be there to distract other users' attention. However, the user who has posted the request can still see that request in his/her user page, which serves as a reference if they want to access their history.

Time limit is important because it enables the system to delete all the outdated requests from the global list in the database so that no irrelevant request is shown to the public. One of the biggest inefficiencies in the Facebook platform is that all requests are kept, and so the list is populated with expired requests.

4.5 UX/UI details

We went further than our original aim to create a functioning app–we paid a great amount of attention to detail when designing the application so that it not only functions well, but also looks professional and is easy to use. Here are some examples of the details that might not be obvious in the screenshots:

- As mentioned earlier, we streamlined the log-in process by enabling users to log in with Facebook. As a result, users don't have to create an account and a profile for our app. We chose Facebook because we hope to replace the Free & For Sale forum, which is used by Facebook users. However, we might also add the option to log-in with Google in the future.
- We made sure to think from the user's perspective and do most of the work for them in the app. For example, when users click on the text field to input the price for the item they're selling/buying, the app automatically fills in the text field with the currency sign in order to make sure that all inputs are in the same format (otherwise, users have to think whether they need to input $10, 10 dollars, or just 10). In addition, we changed the keyboard type so that users only have the option to input digits. By limiting users' options, we save users' time (they don't have to manually switch to the keyboard interface with digits) and enable them to quickly understand what kind of input we expect from them.
- We studied successful commercial apps in Android in order to learn the best practices in UI design. We added rounded corners for profile photos in the service request list, and adjusted the spacing between UI elements in order for the interface to look clear and professional. We followed some of Google's app designs to add a collapsible title for the user page. We tested the app on different devices with different resolutions to make sure that it looks good on all types of devices.

5 Results and Conclusions

The code of the project can be accessed via: https://github.com/TonyJin001/MiddShare

The UI of our mobile application is shown below in our appendix. It shows the flow of the application from logging in (Fig. 3) to the main page (Fig. 4) to all other functions.

For each of the aforementioned problems, we have addressed it in the following ways (all referenced figures are in section 8:

- We have a sorting function that allows users to sort by categories as seen in Fig. 5. We can sort by newest post, oldest post, user name, or by buying/selling. This allows users to have a variety of options when looking at the requests. One popular one, if a student was interested in making money, is to sort by "buying" so that he/she can try to fulfill those requests and make spare cash. The buying and selling is also color-coded to distinguish between the two.
- Users can comment on other requests to establish contact and correspond to one another if they are able to fulfill a request. Fig. 6 shows a user who is interested in the guitar that another user is selling.
- We have individual user pages so that users have a central location where they can access all their requests. Fig. 7 shows the user page. This is also where a user can delete requests after they have been fulfilled. This does not happen in the Free & For Sale group on Facebook which, as a result, contains many irrelevant posts.

- A user can quickly add a request as seen in Fig. 8. They fill in the title, the price, other additional information, and the optional time limit. They can indicate whether they are buying or selling on top of the page and include an image of the item.
- We have included a time-limit option for people who are adding requests. It allows them to pick either a time limit (such as 2 hours) or a date/time as seen in Fig. 9 and Fig. 10. If this request is not fulfilled by that time, it will be automatically deleted from the global request list. This is to ensure that time-sensitive requests do not populate the request list after the deadline. For example, someone may need Computer Science tutoring for his/her Algorithms final on December 14th. He/She will not need this request after the 14th because he/she will have already taken the final.
- We have added a Geolocation Fence, which allows only Middlebury users to access the application. Therefore, we can ensure trust among the users because it is a community-based platform with social capital. In other words, there is a social cost if you flake out or do not pay.
- We allow for notifications when someone posts requests, and the other users can directly see what the requests are on the notifications. They only have to click on the notifications when they are interested in the requests. On the other hand, notifications for the Free & For Sale group cannot be distinguished from other Facebook notifications. They also do not clearly show what is being sold/bought on the notifications. One example of a notification is in Fig. 11.
- We have added a price-awareness component that will send an alert for any exchange over $5000 (Fig. 12). Instead of placing a strict limit on the maximum price, which might stop people from selling expensive items like cars, we send the users an alert message whenever the price is over our threshold ($5000). Users have the option of changing the price to a reasonable number, or confirming the price (if they are really selling something more expensive than $5000), which prevents our system from sending an alert message on this request again. In this way, we can discourage people from entering unreasonable prices to attract attention, while not affecting the user experience of those who enter accurate prices.

6 Discussion

We have resolved all of the biggest issues of the original Facebook platform and also learned how to create an Android mobile application. It was a successful venture in learning a new skill by ourselves and applying it to a real world problem. It was also an experience in working as a team, which is an important skill in all future careers.

7 Open Problems and Future Work

In our presentation we mentioned a few problems that needed to be fixed by the end of the semester. We are happy to mention that sorting, geolocation, notifications for comments, price alerts, and the option to include an image in the request have all been implemented and mentioned in the sections above.

In the future, after testing the application with a group of students, we hope to publish the app on Google Play. We also hope to collaborate with Middle Endian, Middlebury's CS student club, and develop an iPhone application with the same functionalities. Our aim is to potentially implement the application on Middlebury's campus so that students can have an easier way for exchanging services and items.

References

[1] Browse. Discover. Experience. (n.d.). Retrieved December 14, 2016, from http://pinchapp.com/

8 Appendix

Figure 3: Log In Page using Facebook

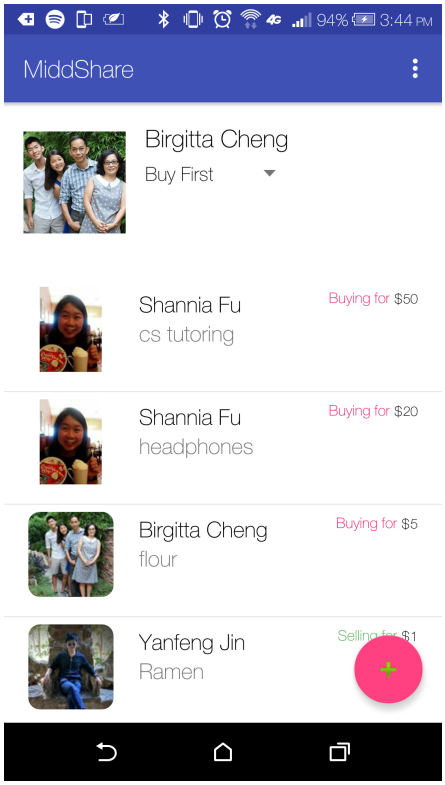

Figure 4: Main page that shows all requests

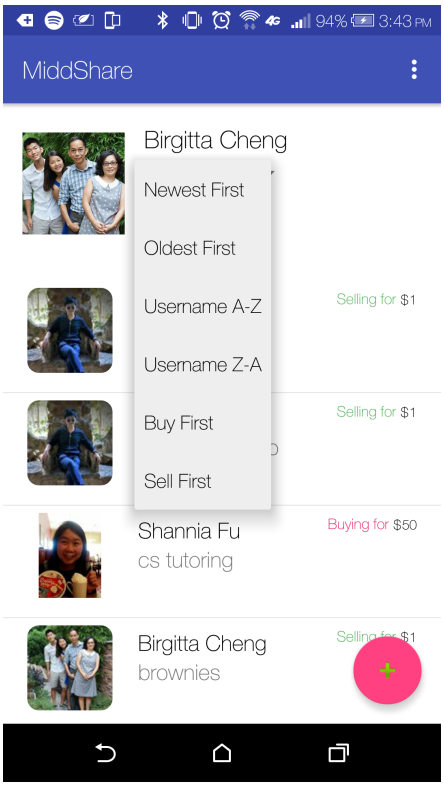

Figure 5: All the sorting filters for the requests

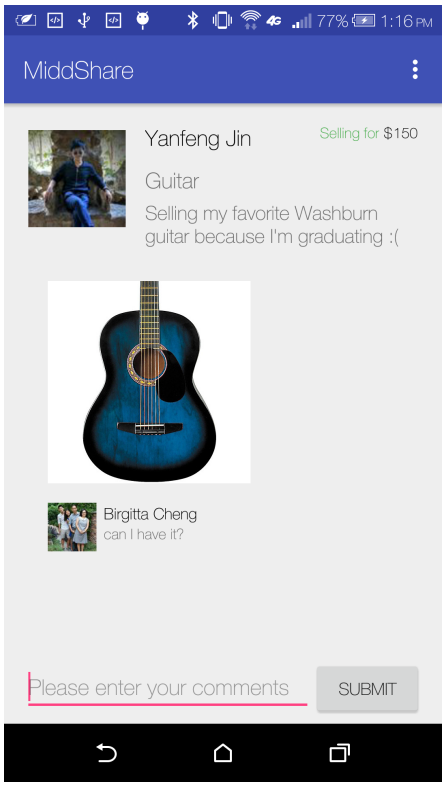

Figure 6: Request with photo and comment

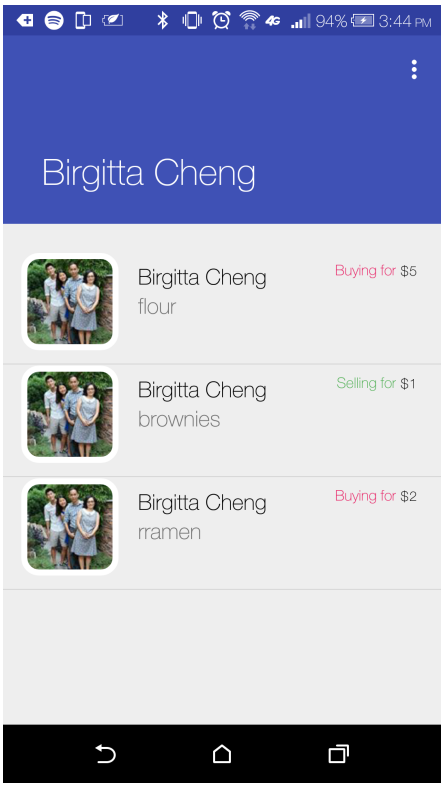

Figure 7: The user page for accessing your own posts

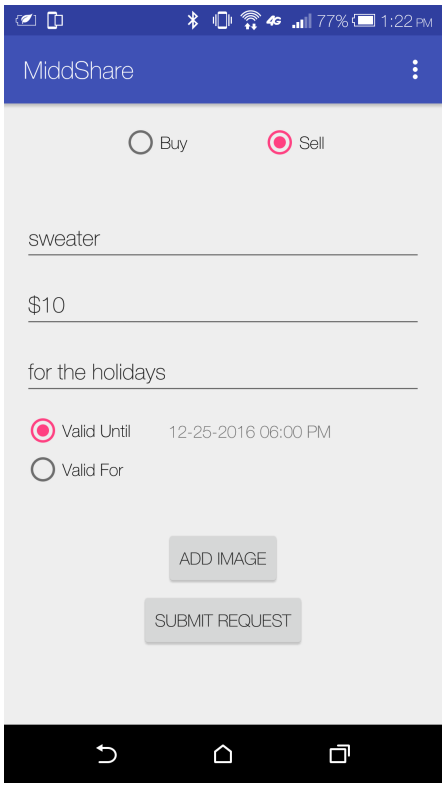

Figure 8: Request with image and time limit functions

Figure 9: Date selection for the time limit

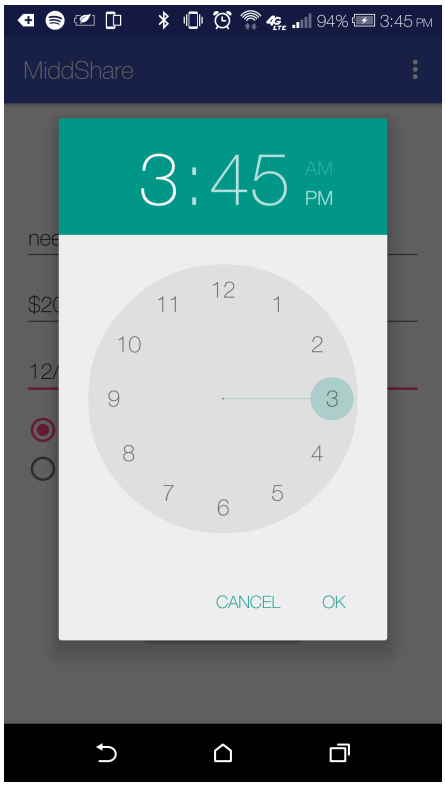

Figure 10: Time selection for the time limit

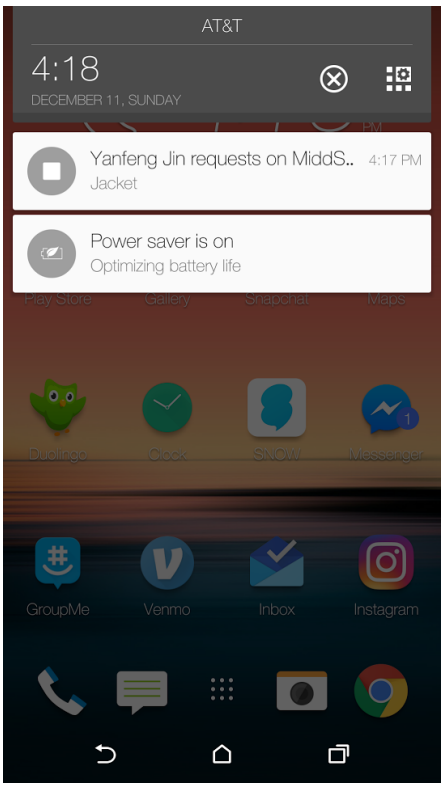

Figure 11: Notification that someone has posted

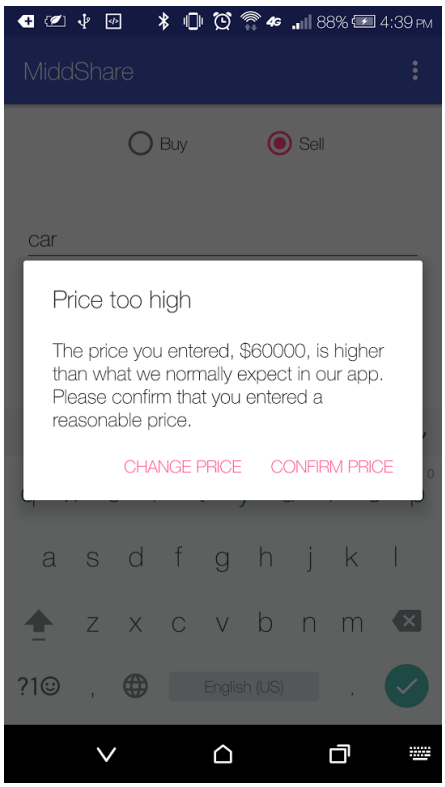

Figure 12: Alert that the input price is too high