**Team H - Sobrio Final Report**

Sobrio is an application which connects people who need rides with the designated drivers in their university, organization, or community. The main goal behind the Sobrio app is to prevent drunk driving by providing a simple and safe way to find a ride. Anyone that needs a ride home from the bar, a ride to the movies, or is going to a friend’s residence will be able to use this application to share rides safely between people in their community. The application can also be extended for other purposes, such as carpooling with co-workers to work or catching a ride to essentially anywhere.

Sobrio has two sorts of roles that users can fill, the driver role and and the rider role. The average user of Sobrio would be a rider looking for a driver, although there will also be many drivers which are looking to make money by giving people rides. There need to be separate interfaces set up for a driver or a rider, and those must not intermix. Drivers must be limited to specific options, and users must be limited to separate options. For example, only riders should be able to review their drivers, drivers must not be able to review themselves. The riders would only be limited to viewing the location of their own driver and track them only after the driver has accepted that they be allowed to do so. The driver would be able to see information on the rider to a certain extent, for example their rider’s reviews, their full name, and their phone number to contact them.

Drivers and riders will be able to join groups and communities of people similar to them so that they would hopefully be given rides by people that they are more familiar with. For example, a University of Connecticut student at Storrs would likely join the UConn group on Sobrio, wherein they would interact and be able to offer and receive rides solely from those that are in the UConn group, which is going to be UConn students and possibly staff, for the most part. More precise groups can be made to target specific demographics in a certain community, such as students or staff.

For the most part, our overall goals have not changed much, if at all, since the commencement of this design project, there have been changes over the course of the design project that have affected our progress in reaching our pre-determined goals and have required us to adjust the implementations of large parts of the application.

One such major change occurred over the summer when there was a lack of communication with the Sobrio developers and we had only realized at the beginning of the Fall semester that the Sobrio API and their entire backend system was overhauled and rewritten. The Sobrio developers had moved from using a Parse database over to their own MongoDB system. As a result of this change, we realized we were going to have to rethink and redesign many portions of the work we had already completed on the app, as we had originally made plans to use the Parse Object API. Having this new Sobrio API thrown at us halfway through our design process was very limiting for us and caused us major delays.

Around the middle of September, Sobrio also released a new Android version along with a front end update that changed some of views and image assets within their application, which they then shared with us and asked us to implement into our ongoing design. Because of certain factors, such as the resolution of the images provided and the need to resize the images, and similar factors which resulted in delays, we decided that our implementation within Titanium was not the best and that it would be easier and much more beneficial for us to start from scratch and write new code using the updated Sobrio designs as our new model for development.

**Team H - Sobrio Team Assessment**

**What did you/your team accomplish in this course this semester?**

The goal of our design project was to develop a cross-platform mobile application that used Sobrio’s API for on-demand ridesharing. Our project was divided up into multiple phases which were distributed into the prototypes throughout the semester. Within these phases, we implemented separate specifications that were given to us by the Sobrio developers and the Sobrio founder, Tom Bachant.

The preliminary project goals were to implement the respective GUIs for Sobrio within Titanium, and afterwards to focus on porting and implementing backend functionality using Sobrio’s API. Implementing the classes and objects necessary to communicate with the Sobrio API was also a requirement. The backend functionality of Sobrio would include the ability for users to sign in and sign up through Facebook, the ability to request rides, add credit cards, view ride details, chat within the app, pay for a ride, and manage user accounts via a settings GUI.

Through our team’s process in becoming acquainted with Titanium, our progress on the Sobrio design project this semester began with the implementation of the GUIs. We developed a plan on development and we viewed GUI development as a good way for the team to gain a good understanding of how Titanium SDK functioned as well as best coding practices within Titanium. Our plan was to develop new GUIs at a steady rate, and while we were on track for this throughout our first few prototypes, we quickly found out that Titanium Studio and SDK had some compatibility issues that caused it to be more difficult for some of our team members to assist within the development process, which caused an overall slowdown on developing GUIs within Sobrio.

Fellow team members that were capable of assisting with the development process quickly gathered a basic understanding of Titanium SDK and began implementing the preliminary GUI functionality that was planned within our prototype schedule. One of the first GUIs that was implemented was the Sobrio login screen and the driver/rider chooser screen. Some of our team members split off and began work on the backend while others continues rolling out GUIs. Still, we ran into trouble and delays due to team members that were unable to fix their compatibility issues with Titanium and as a result could not contribute towards code development.

Following our third prototype of the semester, we had completely accomplished one of our main preliminary goals, which was user login and sign up through Facebook. We used Titanium’s built in support for Facebook authentication in order to accomplish this and we were able to make it so that the user was forwarded to the role choice screen after authenticating. After this goal was accomplished, we realized that we could not effectively proceed on solid implementation of the rest of the goals, such as ride request or ride payment, or settings management, until the Sobrio API has been successfully integrated into our Javascript environment within Titanium. Further delays in implementing the link from the Sobrio API to our Javascript environment were created due to the learning curve that was involved in becoming familiar with the API to such a level where we were able to create the necessary objects and classes within Javascript to make the Sobrio API respond to requests. We ran into a bottleneck specifically when we realized that some HTTP methods that the Sobrio API required were not supported within Titanium SDK, namely the HTTP PUT method. Time was spent researching the solution to the issue, which actually involved modifying the Titanium SDK to add support for the PUT verb, which although was not an ideal solution, it seemed it was the only solution available unless we wanted to completely re-do all the work we had done and restart development in Alloy, which did have support for the method.

By the time we reached our fifth prototype, we had successfully implemented communication with the Sobrio API, which was a key element in our design project that we fundamentally had to complete in order to proceed with anything else. We were satisfied when we reached this state because it meant that no matter what further progress we had ended up making with the GUIs within the Titanium app, the Sobrio developers that would end up taking our code and working with in in the future would have a basic starting point where all Sobrio API communication could be seamlessly made.

By the time we reached our final prototype, we had made significant progress on API integration and GUI implementation. We continued developing GUIs and at our final state, we had developed a GUI for ride request, a GUI to view ride details, and a GUI for managing the user account (Settings GUI). Along with that, the Sobrio API was made fully functional with Titanium. The application is able to function on both iOS and Android environments, although at our final state we were not able to reach a level of completion where all GUIs were implemented and the app was fully functional.

Therefore, although we made a lot of progress, it was not at the level that we had originally planned on making at the beginning of the semester. The failure to strive to reach our level of expectations was a result of multiple factors that caused hindrances in our progress, which will be elaborated upon in the following section.

**What have you/your team learned through your accomplishments listed in**

**response to the previous question?**

There were tons of positive things that we learned with regards to our accomplishments described in the previous question. For a lot of us, this course was quite a memorable one that taught us a lot of lessons that we will likely carry on throughout of lives.

On a purely educational level, we all learned more about how Android and iOS worked internally, as well as app development within those two platforms. We learned how Titanium Studio worked, we gained experience with plugins for Titanium Studio that helped the development process, we became very familiar with the Titanium SDK and developing apps using the Titanium SDK. We learned how to make use of the iOS and Android virtual machines in order to test our beta builds as we developed our app, as well as how to sideload the app onto our smartphones (Android only) so we could do live testing as well as debugging. We learned how to work with the Sobrio API, which was a REST API, and we learned how such APIs are developed and how they are integrated with regards to frontend to backend communication with the database, in this case MongoDB. In essence, we gained lots of experience, became much more familiar with the Javascript language, became much more advanced coders, and we learned how we would go about making a Titanium app and what was involved in developing an app.

One of the more important things we learned through our accomplishments listed in response to the previous question was the fact that working on a project as a group effort can be much more effective than working on a project alone, due to the fact that work can be split up within multiple group members and distributed among the team such that each member only has to accomplish a little bit in order to make the project successful.

Further, we learned that although teamwork and splitting up work was useful in improving efficiency, there was a second aspect to keeping efficiency consistent, and this was the need for documentation. Splitting work up means that team members are only working on and understanding development of a certain aspect of the code, and not following other aspects which other team members have been working on. Therefore, documentation should always be kept up to date, code should be as completely commented and described as possible, and communication regarding code modifications is a must. Also, commit comments on Git should always be specific!

Another thing we learned was how important communication was, both with our teammates as well as with our advisors who were guiding us and giving us support on how to proceed whenever we needed it. We were very lucky to have Tom Bachant, the founder and developer of Sobrio, make himself easily available for us whenever we needed him and easy to communicate with. Whenever our team members had a misunderstanding regarding the Sobrio API or how something within Sobrio worked, Tom was quick to reply with clarification. If we had not had Tom readily available for us, we would never have gotten as far as we did.

**If you/your team had to do the whole project again, what would you do**

**differently or change? Include both the successes (what has gone right so far) and failure**

**(problems).**

If given the chance, our group would make several decisions differently. Many members of our group had difficulties with Titanium, and as a result progress was slowed significantly. Given the opportunity to do things differently a second time, we would likely choose to work with a different language like Java, and develop the app as a native Android application. We believe that working with a more widely used language would have resulted in less time spent trying to fix our development environment, and more time actually developing our application.

Although the project’s goals were originally to use Titanium specifically, a decision early on to rethink our project’s design could have resulted in a more impressive prototype, and a better learning experience for the entire group.

If there was no choice but to still use Titanium, we would make sure we spent more time earlier in the process to sort out our issues with Titanium. It would have been beneficial to speak with someone experienced in Titanium would could have possibly helped us with our technical issues and also given us some advice about how to develop better Titanium apps. If we had been able to get Titanium working with all of our group members, we would have been able to more evenly distribute the workload, and greatly increased the rate of development; unfortunately, since we were never able to resolve our Titanium issues, we only had one or two people who were able to write code without errors at any one time. Through these issues we learned that it is very important to deal with technical issues quickly to minimize their impact on the productivity of the group. We also learned that it is very important to account for possible technical issues when planning a project schedule.

Another change we would make would be to make sure we stayed in closer contact with the Sobrio development team. When the developers updated the front and back ends of the application over the summer, it took some time before our group learned exactly what kind of changes were made. When the developers moved from Parse to MongoDB, it resulted in many of our design documents and prototype plans becoming obsolete. If we had been talking with the developers more regularly we would have known earlier and not had the period of confusion we had at the beginning of the semester and been able to adapt more quickly to the changes

**Team H - Sobrio Developer Documentation**

**Required Technologies:**

**IDE**

* Appcelerator Titanium Studio: <http://www.appcelerator.com/titanium/titanium-studio/>

**SDK**

* Titanium: <http://www.appcelerator.com/titanium/titanium-sdk/>
* Android : <https://developer.android.com/sdk/index.html>
* iOS: <https://developer.apple.com/devcenter/ios/index.action>

**API and API Documentation (Provided by Sobrio Developers)**

* “REST API v1 Documentation”: <https://docs.google.com/document/d/1YwKGcGFfwx3uvLEnSS9amCREB2RZ99jyUs28pAT7yfI>
* API key for communication with Sobrio development server via Sobrio API (Private key provided by Sobrio Developers)
* Sobrio Development server: “<http://apidev.sobrioapp.com>”

**Implementation**

* Replicate functionality of native Sobrio iOS and Android applications
* Images available in “Images” folder in shared Google Drive folder

**Testing**

* Database can be accessed at:

<http://137.117.83.156/rockmongo/index.php?action=login.index&host=0>

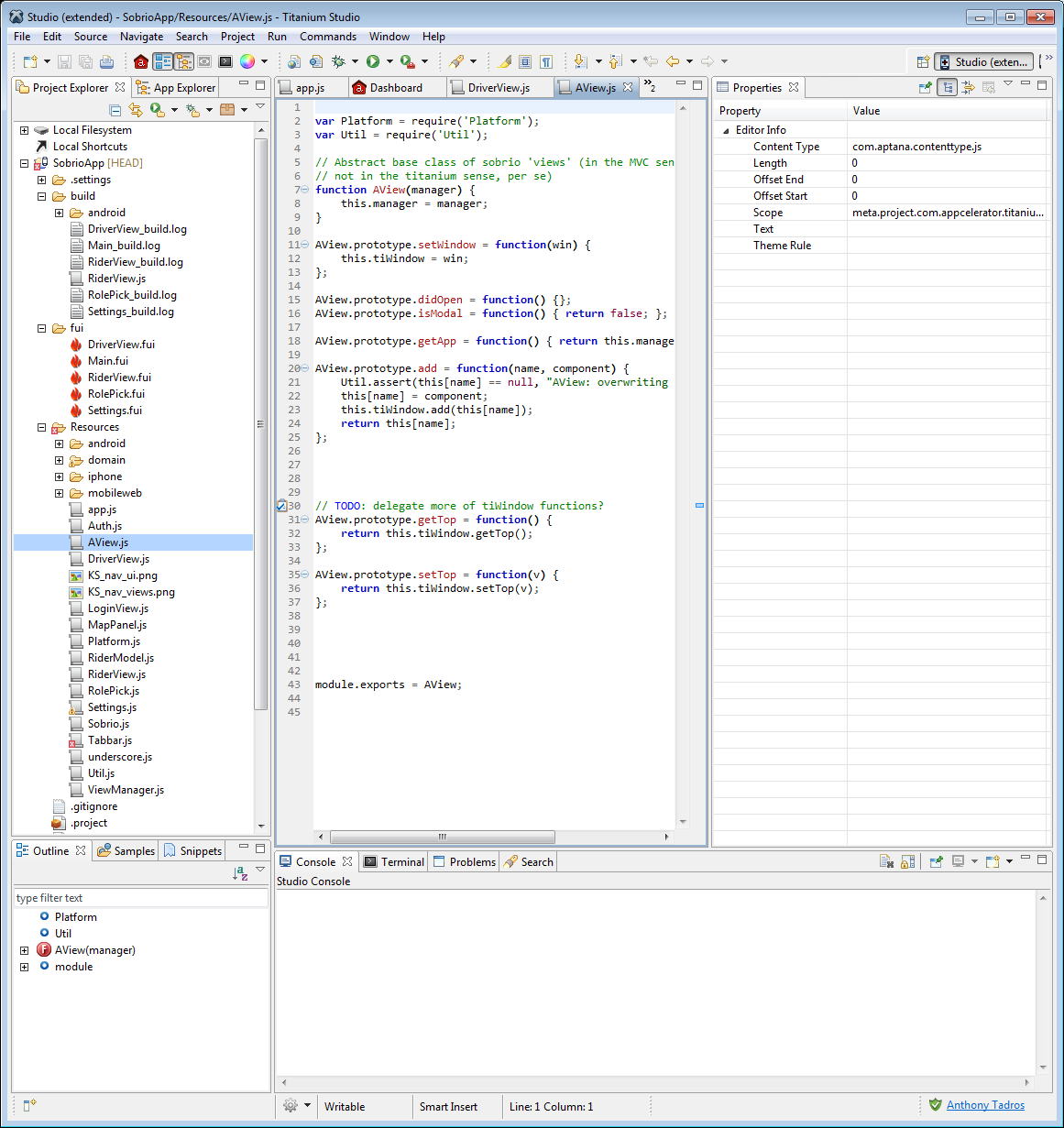
* + Username: admin
  + Password: admin
* Testflight and Google Play alpha testing

**Github**

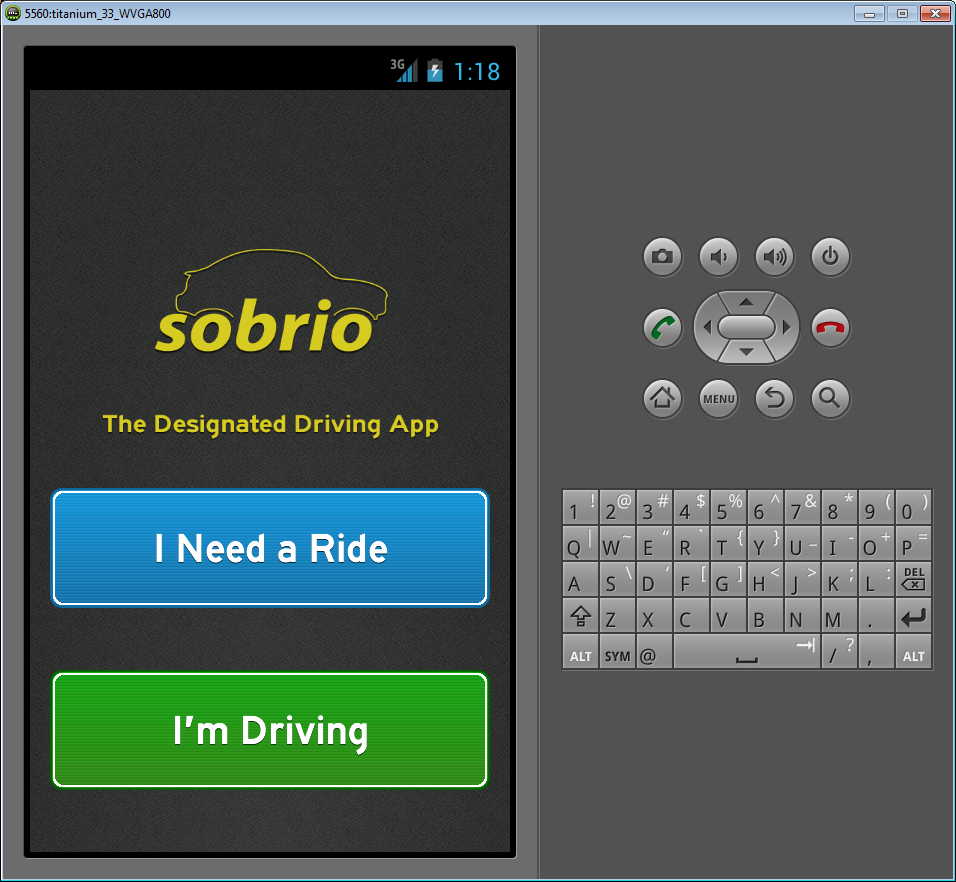
* Repository of source code: <https://github.uconn.edu/aft08003/SobrioApp>

**Instructions on compiling:**

1. Git pull latest source code from repository linked above.
2. Open Titanium Studio
3. Via the file menu, click File, then click Import and select “Existing Mobile Project” from the folder named “Titanium”.
4. Click the “Next” button and on the next window, browse to the project directory where the git repository for Sobrio was just pulled to.
5. Click Finish and the project will be imported on Titanium Studio.
6. Click the green “Play” button at the top, or otherwise click on Run >> “Run” via the file menu, or press the hotkey CTRL+F11 in order to execute the code within the Android virtual machine or iOS virtual machine, as can be seen in the below screenshot (Figure 1).
7. After executing the Android VM or iOS VM, the Sobrio application will open and you will be able to test it, as shown in Figure 2 below.



(Figure 1, Titanium Development Environment)



(Figure 2, Sobrio running within Android VM Environment)

. **Team H - Sobrio User Manual**

An initial user's manual to guide the user through the use of your system.

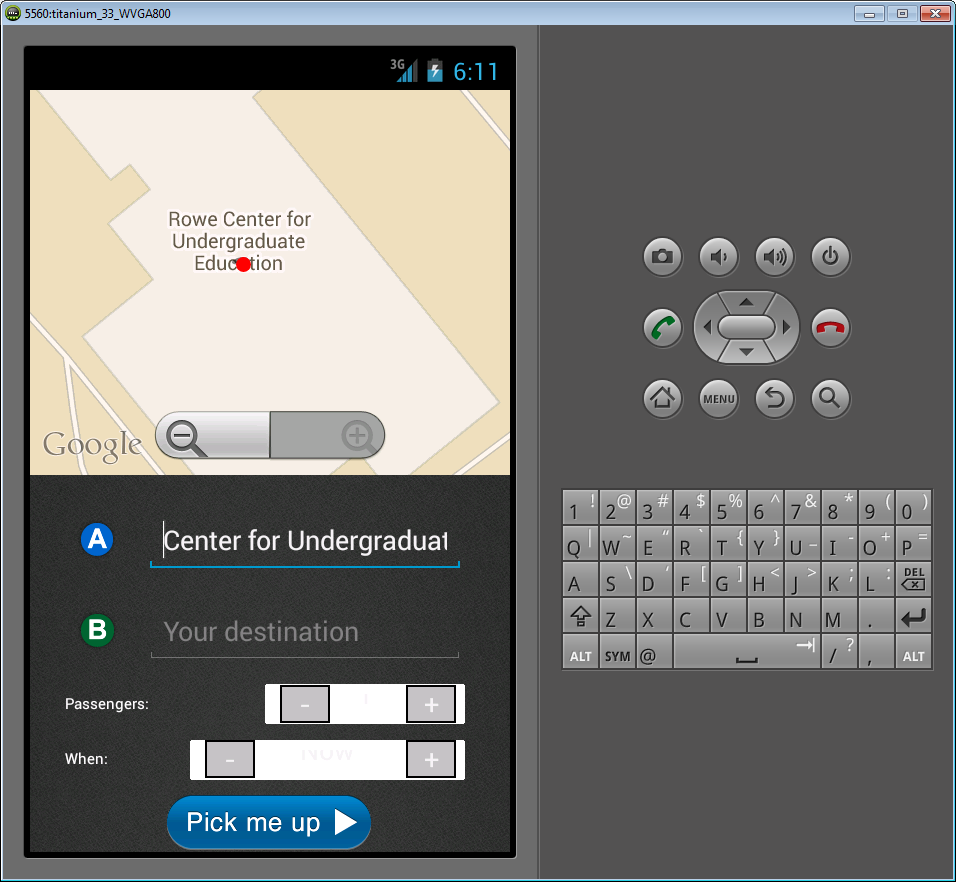
*Although this user manual depicts how the user would typically use the Sobrio app that we developed, please note that the functionality is not fully present within every GUI as of the current date.*

**FEATURES:**

**I Need a Ride**

This option allows users to indicate where they currently are and where they would like a ride to. In addition, they can indicate how many people are in their party. This option is shown in Figure 4.

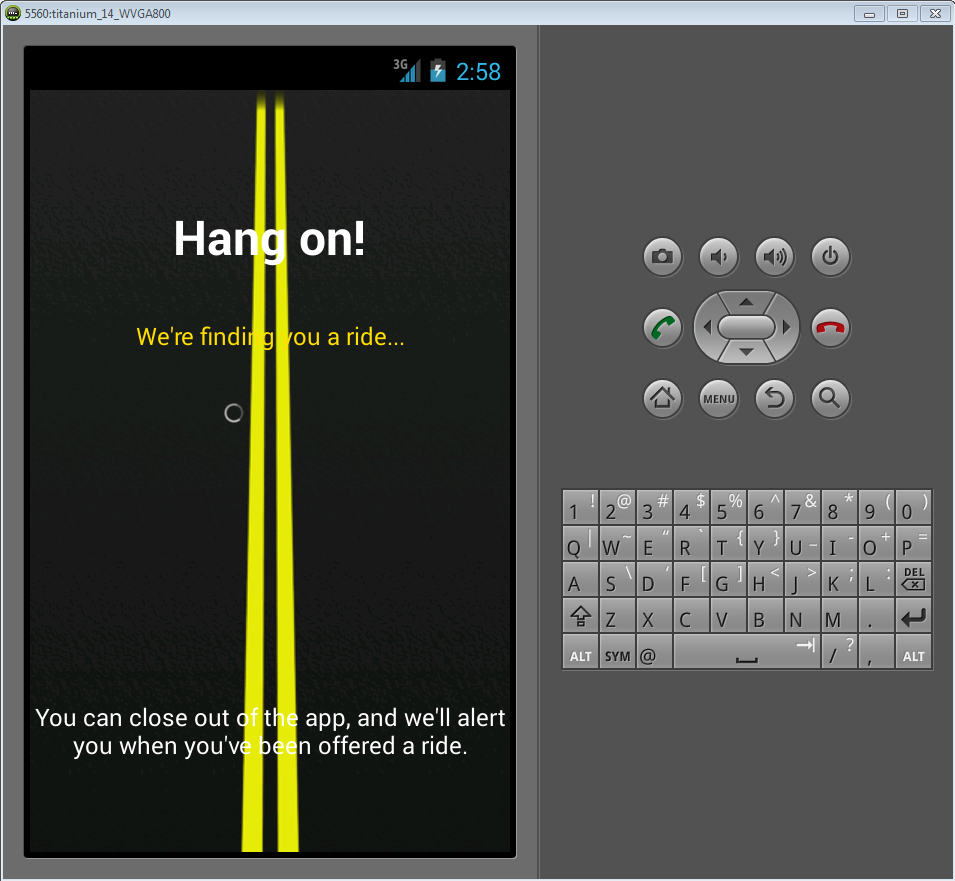
**Ride Request Screen**



This screen has text fields for the user’s current location their desired destination. If the phone has GPS capability Google Maps embedded in the view will automatically detect their current location. The user can then input the expected number of passengers and when they want to be picked up.

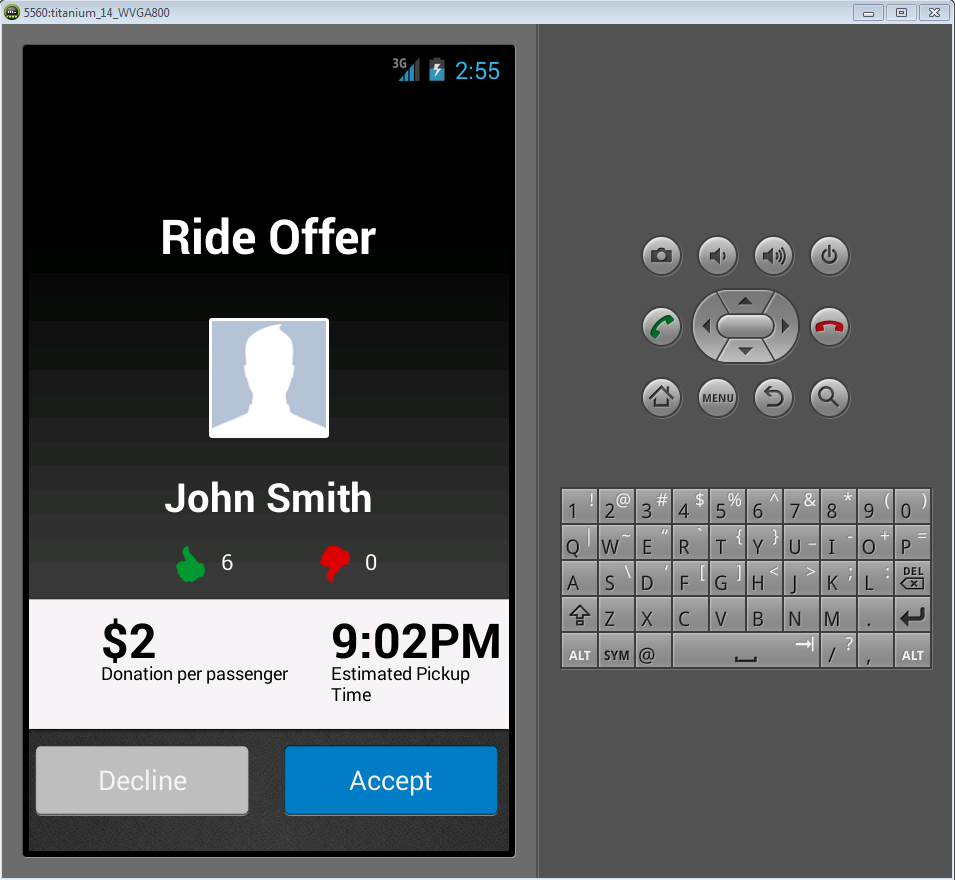
Clicking the pick me up button will move the user to the wait screen.

**Wait Screen**



This screen remains until the user gets a Ride Offer from a Driver

**Ride Offer**



Once the user is offered a ride, they can choose to accept it or decline it based on the rating of the Driver, the charge, or another reason. The Rider may access the Drivers profile through this screen. If the Rider accepts, then the Driver is notified and will go pick up the Rider.

**Profile Screen**

Clicking on any user portrait will display a summary of that user’s profile, including thumbs-up, thumbs-down, number of rides given, received, and canceled, and any ratings from other users.

A user can view their own profile by selecting their own portrait in the menu.

**I’m Driving**

This option allows users to indicate that they are willing to share rides. They are able to select how far they are willing to drive and share this information on Facebook for their friends to see.

**NOTE:** If the user is not a registered Driver with Sobrio, clicking I’m Driving will send them to the registration page on Sobrio’s website where they can submit an application to become a driver.

**Connect to a Group**

This option allows users to connect to a group to receive and share rides from. Groups may be public or private. The group interface is shown in Figure 5.

**Rider and Driver Ratings**

This option allows users to view the ratings of the potential rider or driver they are going to be riding with before they accept the ride. This helps to ensure that the community is kept safe and aware. Users are also encouraged to access this after completing a ride in order to rate the rider or driver that they were with. The interface to view rating is shown in Figure 6 and the interface to rate a driver or rider is shown in Figure 7

**Settings**

The settings menu is accessed from the menu button on Android, or the setting tab in iOS, from this menu the user can edit any of their account information.