CO880 TaxiCloud App - UI and Refactoring Review

School of Computing, University of Kent

Norwood, Anthony
an327@kent.ac.uk

August 11, 2014

Introduction

This review will provide an overview of the system in it's current implementation, that being the state it was delivered to us in after the completion of the CO600 TaxiToo project, and the proposed and planned changes we wish to make to it. That project was carried out by Anurag Ajwani and Blake Smith, students at the University of Kent, for their final year project.

This project is being carried out by Anthony Norwood and Christopher Kelly for their CO880 MSc Dissertation Project, and is a continuation of the work carried out by Anurag and Blake. Both projects aim to interface and connect with the TaxiCloud TNG project, itself a continuation of work carried out in a previous year - The TNG project provided a viable and fully functional backend and web interface for the TaxiCloud system.

However, one of the major issues with the TaxiToo project was the failure to successfuly integrate the backend database with the application, and therefore the app is unable to access the data. To test the application, Anurag and Blake developed a stub test server in order to populate the app with some data.

There existed in the TNG project an API developed to be used in conjunction with TaxiToo. However, after some investigation and contact with the developers it has been determined that the API is not functional - where it to be working, we would be able to marry the two systems together almost seamlessly.

UI

This section will review the User Interface at it stands at the beginning of the project, unaltered from the work carried out by Anurag and Blake.

Login

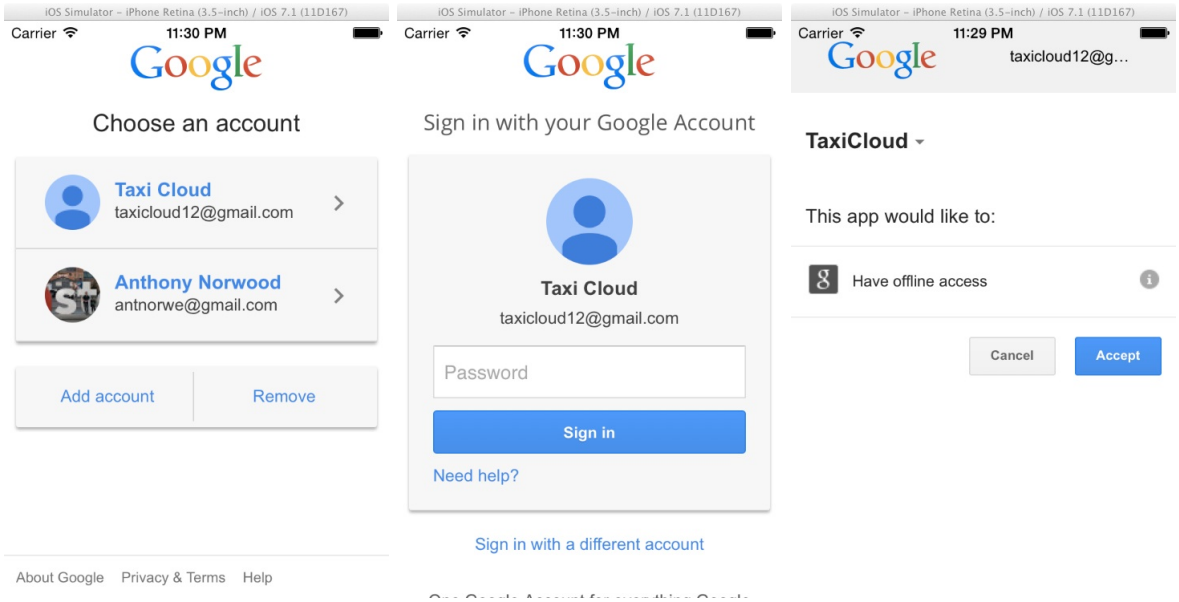

Figure 1: The TaxiToo Login 'Journey'

The login screens are provided by Google as part of the OAuth2.0 class integrated with the application. Each screen shown above is provided as a WebView object and thus is not directly controlled by the application itself - it is simply instantiated by it. There is no opportunity for us to modify, change or otherwise alter these screens as they are centrally provided by Google.

Bookings

Due to the current lack of connectivity to either the TNG database or the test database constructed by the CO600 team, we are unable to critically appraise the way that data is displayed when applied to the constraints of the programming. We have contacted the developers of TaxiToo in order to gain access to their test server and therefore hope to populate the application with some data. Until then, the only insight we have into the booking view is that which is included in the Storyboard. Below are screen shots of the initial booking screen, which also acts as the default landing page for the application, and the 'add booking' screen which at this time does not function as intended.

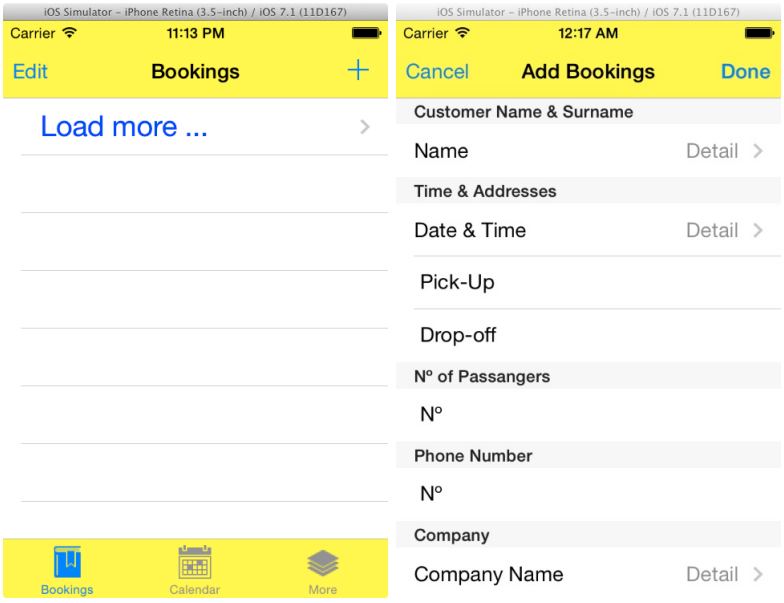

Figure 2: The TaxiToo Booking and Add Booking Screens

Calendar

The calendar view in it's current state is very rigid in it's ability to be manipulated - it uses an external library called TapKu in order to render the calendar view. It is refreshed every time the view is selected, and displays an unscrollable agenda view based on the current system time. While it does include the ability to load different days, the views are locked as starting at midnight - it's possible that where it to be populated by data, the view would load at the earliest booking for that day. However, without the data we cannot confirm or deny this.

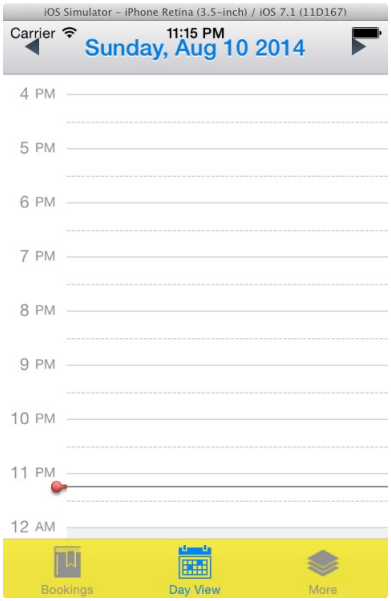

Figure 3: The TaxiToo Calendar Screen

More

The more screen provides all other available functions - links to the currently nonfunctional Driver, Customer and Company views and the logout option. It should be noted that due to the constraints of the Google WebView log in and the pervasive nature of Single Sign On (SSO), the logout button does not in fact log out from Google. Rather, it returns you to the permissions screen as you remain logged in to Google itself. Based on the examination of the OAuth2.0 class, there exists no method to log out from the SSO through the app.

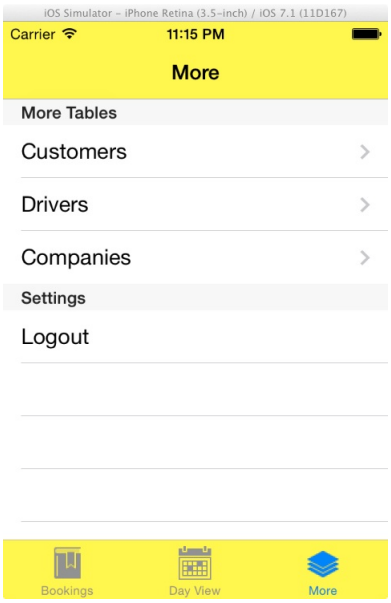

Figure 4: The TaxiToo More Screen

Refactoring

In this section we will discuss the proposed and planned changes that we wish to make to the application. Based on the project requirements the key features will be:

- The ability to view bookings in a list form
- The ability to populate bookings with past or future events
- The ability to view bookings on a calendar

Bookings

Until we can populate the application with data, either from TNG or the stub server, there is no way to accurately determine or test the booking view. It has already been determined that the 'Add Booking' function does not work - when attempting to alter a field, the program will hang and then produce an error in the XCode environment. Further debugging needs to be carried out to trace the exact cause of the issue, however we have surmised at an early stage that it may be as a result of the non-functional database connection. Based on the sample view provided in the Storyboard in XCode, there exists no current opposition to the developed format. However, as soon as we have populated the application with data we intend to conduct a small qualitative survey of

the app design in order to determine whether any changes need to be made.// We would also need to implement the ability to search through both past and future events - currently, the 'load more' option on the booking screen only loads past events and the code only allows for the pre-loading of events for the current day.

Drivers, Companies and Customers

As it stands the Drivers, Companies and Customers views do not function and based on an examination of the code for the views is not completed. The documentation for the TaxiToo project makes reference to the intention to include these features in an earlier iteration, however due to time constraints they removed those features (or at the very least stopped developing them) by the most recent iteration.
It also remains to be seen whether any Driver, Company or Customer data exists in the stub database, and whether the Booking queries require that data to work successfully. We will need to examine the database in order to determine this.

Calendar

The choice of a static agenda view is, to our minds, an unintuitive one. It does not allow the user to look at future or past events outside of the 8 hour slice displayed in the view - thus, if it is possible we would like to alter the view to allow scrolling through time. This would be an extension of the ability to load both past and future events, as the nature of a scrolling calendar requires the bookings to refresh everytime the user 'moves' the calendar rather than storing all possible events locally.

General Code Improvements

Based on our examination of the code, and efforts already undertaken, we are able to remove sections of code or otherwise alter inefficient methods in order to speed up the application and reduce memory usage. For example, based on our method trace during start up it was determined that methods were being called out of order or did not need to be called at all (GetCcompanies being the key example of this). By removing those methods, or otherwise not calling them during the app initialisation, we are able to quantitavely and positively affect the initialisation time.
It was also discovered that when logging out of the application, the access token provided by Google was never revoked and infact was refreshed before presenting the user with the permissions view, creating an obvious risk. This was successfully rectified by altering the method call procedure and ensuring that the relevant classes were properly instantiated.

Revision History

Name	Date	Reason
Anthony Norwood	11/08/14	Document Creation