McMaster University SE3A04 - Fall 2014

Project Outline for Software Design II (SE3A04)

Taxi Sharing Application

Last Updated: September 1, 2014



With the advent of mobile operating systems, such as Android, iOS, and BlackBerry, the development of software applications for mobile devices has become increasingly popular. The aim of this project is to specify, design, and implement an application for a mobile device, in particular, one based on the Android platform.

1. Project Outline

Have you ever needed to take a taxi somewhere and were astonished at the cost for the trip? Do you wish that there was a convenient way to create taxi carpools with others going to the same destination in order to minimise the cost? Does the high cost of taking a taxi deter you from using the service?

A local Taxi Company wishes to attract more customers by oﬀering a more convenient way to create taxi carpools as an incentive to use their service. The expectation is that more customers will be inclined to use the taxi service if the cost of a trip can be minimised by sharing the taxi with other customers headed to the same destination. Despite the risk of short term losses, the Taxi Company projects that the convenience of arranging carpools will lead to an increase in customers, which in turn will result in a higher revenue in the long term.

You and your development team have been contracted to specify, design, and implement a mobile Taxi Sharing application that allows taxi customers to arrange taxi carpools to minimise the cost per customer.

The software should meet the following minimum requirements:

1. The application must implement a centralised “Distpatcher” which must store all relevant information about each taxi in the fleet. The “Distpatcher” shall decide how to handle oﬀers and requests for taxi carpools.
2. The application shall have two modes:
   1. In “Request Taxi Carpool” mode, the application must allow for the customer to enter their destination along with some optional search criteria (sort by nearest cab, highest rating first, find specific oﬀeror, etc.) to be sent to the “Dispatcher”. The “Dispatcher” returns a list of potential matches to the customer’s search criteria. When the customer chooses a match, a request is sent to the “Dispatcher” which forwards the request to the oﬀeror of the selected match.
   2. In “Oﬀer Taxi Carpool” mode, the application must allow the customer to scan a code which would theoretically be placed in the taxi and represent a taxi ID. A barcode or a QR code are the obvious and suggested formats, but other options may be viable, provided they are fully explained and a rationale is given for their choice. The application must then allow the customer to enter the relevant information of their oﬀer (destination, taxi ID, maximum number of customers to share with, etc.) to be sent to the “Dispatcher” for processing. Once an oﬀer has been made, the customer waits for requests.
3. Each customer must register with the application by creating a profile that can later be edited or removed. Only registered customers may oﬀer or request a taxi carpool.
4. When a taxi carpool request arrives, the application must present the options to the oﬀering customer, indicating the potential changes to their trip (estimated fare, distance, time, etc.) along with some kind of measure to help them decide whether it is optimal to accept or refuse the request.

1

McMaster University SE3A04 - Fall 2014

1. Upon arrival at the destination, the application must:
   1. Display the fare owed by each customer. You may decide on the business model for calculating the fare provided that it is fully explained and a rationale is given for the choice. (Tip: The taxi company should not be losing money by oﬀering the Taxi Sharing service.)
   2. Present each customer with a rating form to assess the quality of the customers with whom they have shared the ride.
2. The application must encrypt all transmitted messages using the cryptosystem of your choice for confidentiality and integrity purposes.
3. The application must interface with Google Maps or another online mapping service to allow customers to track the taxi route and present potential routes when requesting or oﬀering taxi carpools.

\*Note\* The implementation of the requirements listed above constitutes the minimum requirements for project completeness. Any additional functionality will be rewarded. Each team is required to implement at least one additional innovative feature not mentioned in the outline. Creativity will be generously rewarded.

1. Technology

The Android Software Development Kit (SDK) is natively written in Java. However, there exists the Scripting Layer for Android (SL4A) which allows you to develop your android applications in scripting languages such as Python.

You are free to use Java, Python, or another scripting language supported by SL4A, as long as it meets the specified requirements. However, we are better equipped to provide you technical support with Java.

1. Tools

The Android SDK has the tools, sample code, and documentation you will need to create applications. The Android SDK is available at: http://developer.android.com/sdk/index.html.

If you plan to implement your application using Python, you will need the Scripting Layer for Android

(SL4A). SL4A provides interactive interpreters, script editing, and script execution for various scripting languages. SL4A is available at: http://code.google.com/p/android-scripting/.



You are free to use any software design tool, i.e., StarUML, Visio, etc. and any version management tool, i.e., SVN, Google Code, SF.net, etc., as long as it meets the project requirements.

1. Resources

\*Note\* A large portion of this project is to do the required background research on working with the Android platform and information security. Keep in mind that a substantial component of any software project is to solve and/or eliminate the underlying technical diﬃculties. This often means hitting manuals and Google. To start you oﬀ, we provide a narrow selection of references and resources.

2

McMaster University SE3A04 - Fall 2014

1. Android, Java, and Python

There are numerous online resources, including code samples, tutorials, screen-casts, etc. to help you get started with the development of your Android Application.

For those of you wishing to implement your application in Java, a good reference is the following:

1. Sue Smith. Learn Android SDK From Scratch. Tuts+. 2013.

Available: http://code.tutsplus.com/series/learn-android-sdk-from-scratch--mobile-21677

This is a 17 part series of tutorials to help you get started with the Android SDK with a particular focus on programming with Java.

For those of you wishing to script your application in Python, some good references are:

1. Paul Ferrill. Pro Android Python with SL4A. APress. 2011. This book is available free online in PDF form.
2. Paul Barry. Python for Android. Linux Journal. 2011. Available: http://www.linuxjournal.com/article/10940

This is a good beginners tutorial to help install the Android SDK, SL4A, and how to configure things to get them to work. The tutorial is written for Linux but can easily be used for Windows and Mac OS X as well.

For general reference, the Android Developers site is a good place to look:

http://developer.android.com/index.html

1. Information Security

There is a wide variety of resources to help familiarise yourself with information security including encryp-tion/decryption algorithms. We recommend the following textbook to give you a general overview of many of the concepts required to complete this project.

1. Matt Bishop. Computer Security: Art and Science. Addison-Wesley. 2002.

In this book, you will find information about

– Confidentiality Policies: Chapter 5 (pages 123–150)

– Integrity Policies: Chapter 6 (pages 151–167)

– Basic Cryptography: Chapter 9 (pages 217–241)

\*Note\* Please keep in mind that you can find many of these ideas online in the form of tutorials and lecture notes, among others.

1. Deliverables

Every team will submit one team copy of each of the following deliverables at the beginning of lecture on the specified due date with the exception of D4 which will be due during the scheduled time for the project demonstration. Each deliverable must have a list of the contributions of each team member and must be signed and agreed by each team member.

3

McMaster University SE3A04 - Fall 2014

\*Note\* The due dates may be adjusted by the instructor as circumstances dictate.



D1: Software Requirements Specification (10%)

October 6, 2014

D2: High-Level Architectural Design (10%)

November 3, 2014

D3: Detailed Design (10%)

November 24, 2014

D4: Final Code (10%)

December 3, 2014 (approx.)

\*Note\* The project described in each deliverable may be a subset of the project described in the previous deliverable. You may cut features as you progress through the semester. This idea is illustrated in the inverted trapezoid diagram. However, the final implementation must contain at least the minimum project requirements as specified above, plus one additional innovative feature of your own design. Creativity and additional eﬀort will be rewarded, so think big at the beginning and keep your options open as long as possible.

1. Log Books

Each team member is expected to keep a hand-written log book that contains all the notes of the work done during the team meetings and your individual contributions to the project done on your own time. Log books will be randomly requested to evaluate a student’s participation during the course of the project, and may be used as a resource in the event of a dispute within a group. Be sure to bring your log book to all tutorial sessions. Remember that the log book must be a physical book.

4