CSC207 Introduction to Software Design
Winter 2015 – Project Phase III

Logistics

- **Due date:** 9:00pm Tuesday 31 March 2015 (Preliminary interviews (worth 3%) week of 23 March.)
- **Group size:** Four. In this phase of the project you continue to work in your team from Phase II.

Overview

In Phase III of the project you will complete the implementation of the Android application for searching air travel itineraries.

Learning Goals

By the end of this phase, you should have:

- practised dealing with changing software requirements in the course of the software development project,
- worked closely with your teammates to re-evaluate and possibly update your design of a software system,
- produced a working Android application that implements your software design and corresponds to user requirements.

Phase IIIa Interview (week of 23 March 2015)

The purpose of the interview is to demonstrate to your TA that you have a working (although very limited!) Android application. The application must contain the following:

1. Launch and start the main activity.
2. Take some input from the user and use it (in a .java class).
3. Transition from one `Activity` to another `Activity` and carry information between the two `Activity`s in an `Intent`.
4. Use at least one class from your back-end (implemented in Phase II of the project) intelligently.

All team members must be present for the interview and should be prepared to answer questions about the implementation of the Android application. You will be contacted by your TA to schedule the interview.

Feature List for this Phase

Here are the features that you will implement for this Phase of the project. A lot of this functionality you should have already implemented in the previous phase of the project.

- A User (Client or Admin) can launch the flight booking application and login using a username and password, which loads saved data, if it exists. (In our, unrealistic, implementation, you are allowed to simply store usernames and passwords in a file on the device[1].)

[1] Requirements for passwords file:

1. In the PIII directory, commit the plaintext password file, `passwords.txt`, that you will be using for authentication.

- A User (Client or Admin) can search available flights by entering a departure date, and travel origin and destination. A flight info should include: (1) flight number, (2) departure date and time, (3) arrival date and time, (4) airline, (5) origin, (6) destination, (7) cost, and (8) travel time.
  The flight numbers should be unique. We will not deal with time zones, leap years, or any calendar or timing differences, in this project.
- A User (Client or Admin) can search available itineraries by entering a departure date, and travel origin and destination. An itinerary should include, per flight: (1) flight number, (2) departure date and time, (3) arrival date and time, (4) airline, (5) origin, and (6) destination, plus an overall itinerary cost and travel time.
  A valid itinerary contains no cycles, i.e. it does not visit the same place more than once.
- A User (Client or Admin) can display search results sorted by total travel time or by total cost.
- A Client can view personal and billing information stored for that Client.
- An Admin can view personal and billing information stored for any Client.
- An Admin can upload (in a csv file – see format below) personal and billing client information into the system.
- A Client can edit personal and billing information for that Client.
- An Admin can edit personal and billing information for any Client.
- An Admin can upload (in a csv file – see format below) flights information into the system.
- An Admin can edit information for a given flight.
- A User (Client or Admin) can select an itinerary from the displayed list for booking.
- A Client can book an itinerary for that Client.
- An Admin can book an itinerary for any Client.
- A Client can view booked itineraries for that Client.
- An Admin can view booked itineraries for any Client.
- All information stored in the system should persist (be available in the next launch) when the application is not running.

The Admin will use the following data format to upload personal and billing client information in a `csv` file:

```
LastName,FirstNames,Email,Address,CreditCardNumber,ExpiryDate
```

where the expiry date is stored in the format `YYYY-MM-DD`.

The Admin will use the following data format to upload flight information in a `csv` file:

```
Number,DepartureDateTime,ArrivalDateTime,Airline,Origin,Destination,Price,NumSeats
```

where the date and time are stored in the format `YYYY-MM-DD HH:MM`.

Our airlines are not very client friendly — they do not assign seats at the time of booking. The `NumSeats` value in the flights data file specifies the total number of available seats that are available for sale on that flight.

Task I — Software Design

With your teammates, discuss what changes you need to make to your design from Phase II of the project. Create a file `crc.pdf`, following the same format you used in Phase II, and commit this file to the directory `PIII` of your team repository.

In contrast to Phase II of the project, in this Phase your CRC model should also include CRC cards for the front-end classes (the activities).

2. In your app, use the default internal storage location for storing the password file: `this.getApplicationContext().getFilesDir()`

Task II — Implementing the Android Application

When setting up your Android Application Project, select:

- Minimum Required SDK: API 8: Android 2.2 (Froyo)
- Target SDK: API 18: Android 4.3 (Jelly Bean)

The Software Development Process

Your team should meet regularly while working on the project. We have two types of meetings — planning meetings and status meetings.

For **planning meetings**, you need to meet twice: once in the beginning of the project and once mid-way through the project phase. During a planning meeting, the team will (a) recap on the current state of the project (if mid-way meeting), (b) decide on a set of tasks the team will accomplish before the next planning meeting, and (c) decide who will perform which tasks.

For the **status meetings**, the team will meet at least once a week, in addition to the planning meetings. During these meetings, each member will report on (a) what (s)he has accomplished since the last meeting, (b) what (s)he plans to accomplish before the next meeting, and (c) if there are any problems/obstacles that prevent him/her from making progress.

To demonstrate the software development process the team followed, you need to **maintain a plain text file** called `meetings.txt`, where the team will record all meeting minutes.[2] *On the day of each meeting*, commit this file into your team repository. The contents of this file must match the state of the rest of your repository!

The end of this project phase

At the end of this project phase, your team should have a working version of an Android application that implements every feature on the above feature list. You should, of course, have Javadoc comments for all your code.

Class `Driver`

One of the tasks in this phase is to complete your back-end implementation. It is entirely up to you to produce the object-oriented design of the application.

In order to test your code, we require that, in addition to your design, you submit a class named `Driver` that implements the methods specified in the `Driver.java` starter code. Notice that class `Driver` is required for testing purposes only! It should not contain any algorithms or do any interesting work. It should merely call appropriate methods in the classes that you designed and implemented. `Driver.java` must be in package `driver`. Read the method descriptions carefully, because some method descriptions have changed since Phase II.

Class `Driver` must be in a package named `driver`.

README

Please include a plain text file **README** in directory PIII of your repository telling your TA where to find your Phase III project. You must tell your TA everything they need to know about your project to help them run your application and navigate your work. If it is necessary to push particular files to the emulator, those instructions should be included here.

[2] See lecture slides for some example meeting minutes.

Task 3 — Team member and self evaluations

Any student who does not submit their evaluations on time will receive a mark of 0 on this phase of the project. The evaluations are due Monday 1 December at 11:59 p.m.

You will be filling out and submitting a peer evaluation activity on CATME. This form will rate all team members, including yourself, on contributing to the team's work (contributing a sufficient amount of work, contributing work of good quality, being on time, helping teammates) and interacting with teammates (showing interest in teammates' ideas and contributions, asking teammates for feedback and using their suggestions to improve, making sure teammates stay informed and understand each other, providing encouragement and enthusiasm to the team).

These are meant to be private: each team member will submit these separately, and you are not required to show each other your forms. In the case of serious disagreement, or if you request it, we will hold a team meeting to discuss the results, but we will never reveal individual ratings.

Marking

All of these items may affect your grade:

- CRC Model
  - The modularity of the design, and the degree to which it is reusable and extensible.
  - The degree to which the design meets the requirements.
  - The use of OO concepts, such as encapsulation and inheritance.
- The appropriate use of files and data structures.
- Functionality and usability of the application:
  - all functions from the feature list implemented
  - stability of application (e.g., it shouldn't crash on invalid input)
  - easy to use application, intuitive navigation
- Javadoc:
  - required for methods and instance and static variables
  - must have a period at the end of every sentence
  - must use @param and @return tags
  - must use good English
- Coding Style:
  - must follow Java naming conventions
  - indentation
  - consistency
  - white space
- Quality of the `README` file:
  - it must take the TA less than 2 minutes to read your `README` file and understand how to run and use your application
- Quality of the software development process:
  - the file `meetings.txt` must be committed according to the schedule
  - the contents of the repository and the state of the code must match the contents of the file `meetings.txt`
- Subversion commit history:
  - participation by all team members
  - frequent commits over an extended period of time
  - appropriate commit logs
- Peer evaluation
  - To view the evaluation criterion, see the CATME online evaluation form (`www.catme.org`).

Checklist

Have you...

- used your new team repository and not your individual repository and not your repository from phase I to submit your work?
- committed `crc.pdf`?
- committed **all** of the necessary Android files? (Everything except for the contents of the bin directory, the gen directory, and hidden files.)
- committed a README file for your TA?
- committed `meetings.txt`?
- verified that your changes were committed using `svn list` and `svn status`?
- before Thursday 2 April at 11:59 p.m.: submitted your team evaluation forms using CATME?