The University Of Sheffield.

COM4510/COM6510 Software Development for Mobile Devices

Assignment 2018-2019

This assignment is primarily concerned with applying the ideas that are being presented in the module on methods and technologies for Mobile Computing. Some of the basic algorithms needed for it have already been introduced during the module, and can be obtained from MOLE.

Organisation of the Assignment

The assignment will require just one submission and it will be worth 100% of the marks. See MOLE for the exact deadline. All submissions are electronic via MOLE.

Workload and Groups

The whole assignment is intended to account for between 20 and 30 hours of each person's work towards the module as a whole. Experience has demonstrated that it is not really possible for one person to do all the work for the assignment in that time, unless they are an exceptionally competent programmer with previous knowledge of mobile programming. Therefore, the assignment is organised on the basis that people should work on it in **groups**. Groups must be composed of a **maximum of 3 members.** Students are allowed to do the assignment in pairs or on their own.

Please note!
You must register your group at https://tinyurl.com/y9eagx7f

Given a group, you will normally be expected to keep with that group for the whole of the assignment. However, if any problem arises that makes this difficult, you should notify the lecturer **immediately**, so that appropriate action can be taken. See lecture 1 slides on this topic. **If the group splits after week 7**, its members are not allowed to join another group.
It is important that you make clear what contribution each group member has made in your final report. It is required that each person contributes to each stage of the assignment, but it is up to each group to decide how to divide the work up between individuals.

Deadlines

The deadline is **absolutely fixed**. You should therefore plan your work to aim at handing the report in at least a few days before the deadline – do not leave it until the deadline, just in case any minor thing goes wrong and you then find that you are late.
Note that the Computer Science department applies fairly severe penalties for handing coursework in late. If you want to look at the details you can find them on the University's web site.

Material Provided

Lecture notes, lab class examples and websites/books as detailed on the lecture notes.
NOTE: no third party code can be used in the assignment, except the code what has been **explicitly** provided in the lectures. For example you are allowed to reuse the code given in the lecture/lab slides and any library (e.g. EasyImage) used in the lab classes. However you are not

allowed to download any code from the Web or to use any other software that will perform a considerable part of the assignment. **Unauthorised re-use of third party software will be considered plagiarism.** In case of doubt ask the lecturer for permission before using any third party code. In general, we will allow only the use of generic libraries designed to improve the look and feel of any interface. However permission must be requested before use.

1. Scenario

The learning objectives of the assignment are to learn:

- to design and implement an app with a flexible sophisticated layout in Android
- to use separation of concerns (using MVP)
- to cope with multimedia data
- to store data locally using abstractions of databases (Rooms)
- to use the sensors (GPS)
- to use async processes
- to develop as a group
- the principles of implementing client/server communication via mobile

The field chosen is management of photos.
The solution will be composed of the following parts, each covering each of the above learning objectives.

1.1. The problem

Design, build and evaluate an application for the management of personal photos on a mobile.
The app will allow to:

1. Visually browse previews of photos stored in the phone's photo library (gallery) (15% of implementation marks)
2. Showing photo locations on a map using the latitude and longitude stored in the metadata (20% of implementation marks)
3. Inspect the details of a photo (both the full photo in itself and its metadata, i.e. geo-location, EXIF data, description, title, etc.). The user must be able to edit and persist the metadata (20% of implementation marks)
4. Enable taking pictures using the phone's camera and store them in the phone's photo library with the associated metadata (15% of implementation marks)
5. Saving pictures and metadata to local database (15%)
6. Using sensors and in particular locating the user using the phone's location services (15%)

These steps are detailed in the next sections. Examples are provided to help understanding the requirements. You are not required to implement that exact solution. You can use your creativity, as long as the formal requirements are met.

Requirements for the solution are:

- It must be fully functional and robust
- It must work on multiple devices with different screen size, processing power and screen resolution
- It must work at least for Android>6.0 and/or iOS>9.0
- It must be efficient, able to cope with a library of thousands of photos
- The interface must be pleasant to the eye
- The interface must work both in portrait and landscape mode
- The interface must follow the typical design patterns of Android and in particular MVP
- The solution must be of high quality. A simplistic solution (although functional) will not attract many marks.

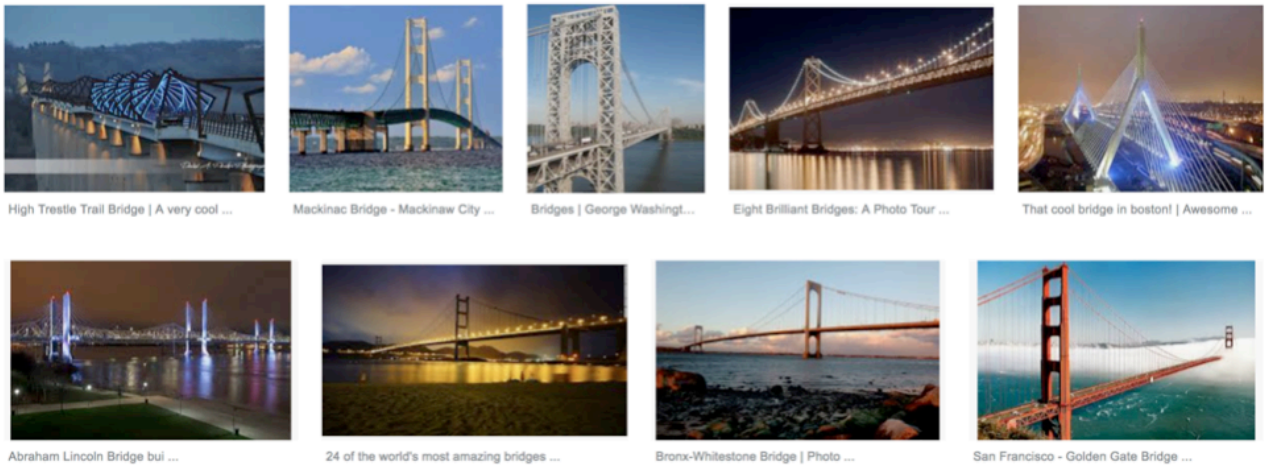
All implementation must be done either in Java (for Android phones). No other languages are allowed (e.g. Cordova, React Native, etc.). If interested in using Kotlin, let the lecturer know in advance.

1.1.1. Visually browse previews of photos

You must create an app that enables a user to visually browse the photos stored in the phone's photo library. In doing so you must design and implement an original program that:

- Allows visualising previews of the pictures
- Allows selecting a picture for further detailed inspection

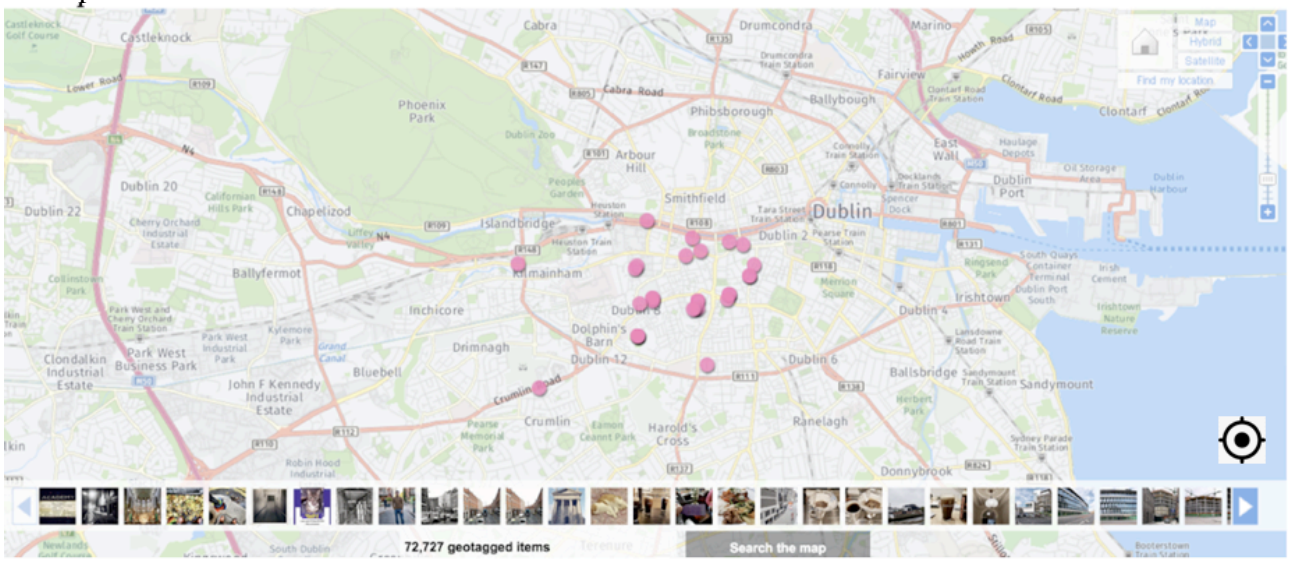
It is important that the interface is efficient and able to cope with a library of thousands of photos. An example of the landscape version of such an interface could be:



As mentioned you are not required to implement this exact solution. You can opt for a different layout (e.g. all pictures may have the same size, etc.).

1.1.2. Showing pictures on a map

Most pictures will have metadata showing the location where it was taken. Another way to browse the photo gallery will be through visualisation on a map. Clicking on a location with photo, shows the details of the photo.
Example of such an interface is:



The only requirement is to show the location on the map and to show the details of the photo when tapped on. The example above however shows some enhancement that you can implement such as searching for locations or showing a preview of all the images shown.

The interface must provide a button to locate the user's current location on the map.

1.1.3. Inspecting the details of a photo

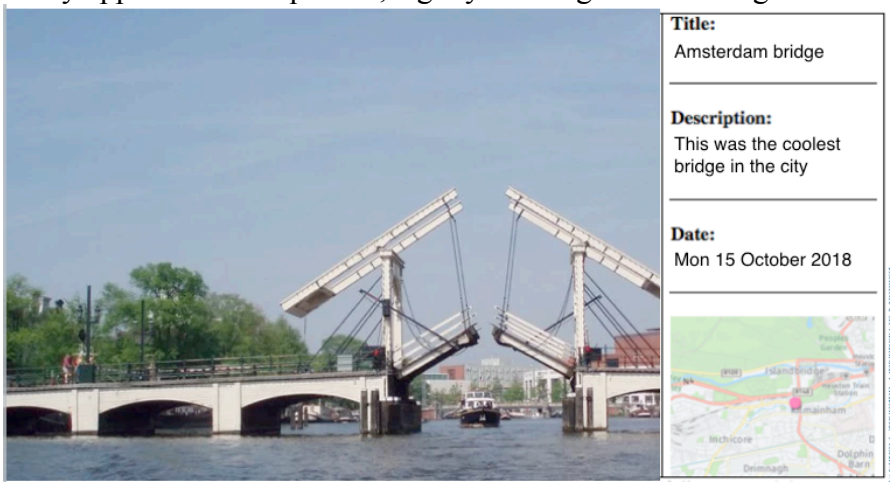
Tapping on a photo in the interface described in 1.1.1 should allow inspecting the details of that

specific photo. While the 1.1.1 interface would show only previews of photos, here the full photo must be shown together with the following metadata:

- A title (if missing, default will be the photo filename)
- A description (if missing: show a hint saying “add a description!”)
- A map showing the location where the photo was taken using latitude and longitude taken from the photo’s Exif data (if missing, do not show the map in the interface)
- Any other interesting metadata (e.g. other Exif data)

An example:

Note that as you are required to show the full photo, the photo should be full screen and the metadata should only appear when requested, e.g. by clicking on a Floating Action Button.



As mentioned this is just an idea and you are free to implement as you deem fit. Make sure that the solution is intuitive for the user.
The metadata must be editable in simple way.

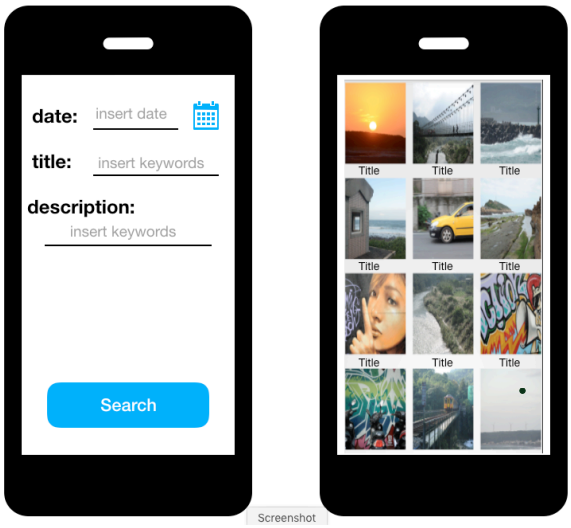
1.1.4. Taking pictures

The app must allow taking pictures using the camera. This functionality must be working both when used on a real device and on the emulator. If the phone model does not have a camera the functionality must not be available to the user. The pictures must be geo-located. If run on an emulator, mock location data must be provided.

1.1.5. Saving metadata to a local database

All the metadata mentioned above (title, date, description, GPS coordinates, pointer to the image

file, etc.) must be saved in a local database implemented using Room (i.e. not directly using SQLite), so that it can be retrieved at a later stage.



The database must allow searching of images based on keywords in title and description AND/OR date of creation. Implement an interface for searching images, e.g. like the one shown above. Note that you must implement an aSync process, as accessing a database on the UI Thread is not allowed.

2.Marking schema

Each part described in the subsections above will carry marks divided as follows:

50% for the quality of the solution, inclusive of separation of concerns, use of async processes, quality of the user interface, etc. as well as compiling and running without an issue

35% for the quality of the documentation

15% for the correctness of results.

Please note

- the direct consequence of the marking schema is that providing a program returning the correct solution is not enough to get a pass mark. You will need to implement the correct strategies and document/discuss them properly! Quality of documentation and code are very important issues for computer scientists.
- Solutions not working on the departmental computers (e.g. working only on personal computers) will not be considered.

3. Handing in

Your solution must be contained in a self-contained directory called *COM4510 or COM6510* (<***MainDirectory***> in the following) compressed into a zip file submitted through MOLE.
The directory must contain:

1. The source code of the solution (please note that we will both inspect and run the code) contained in the directory **<*MainDirectory*>/code/**.
   All the code must be in the exact format to be run via AndroidStudio
   (a) All code must be **developed in Java**. We must be able to run your solution without problems on a standard lab machine. Please note that the quality of the code carries a relevant portion of marks, so be sure to write it properly.
   (b) All the external libraries must be included in your solution. Maven or Gradle links are acceptable. Libraries are allowed only if previously agreed with the lecturers. No library doing a substantial part of the assignment are allowed.

2. The document documenting your solution as deigned for development. The report must be contained in the directory **<*MainDirectory*>/report/.** An outline and set of requirements for the report are provided in Appendix A.
3. The documentation in the Java files must be of very high quality. Please note that this documentation carries a relevant portion of marks, so be sure to write it properly. More information on guidelines for Java (JavaDoc) see http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html
4. Screenshots of the different app screens showing the implemented functionalities **<*MainDirectory*>/screenshots**
5. The filled self-assessment form.

3.1. How to submit

Everything must be submitted electronically. Nothing is to be handed in at the reception! **Use MOLE**. Store your solution in a .ZIP file that when unzipped will generate the directory organisation described above. As emergency measure (and only in that case!), if any last minute issue should arise in handing in electronically, please send your solution by email to the lecturer (cc to demonstrators) in a self contained .ZIP file.

3.2. Anti-cheat measures

Please note that measures are in place for detecting plagiarism and in general cheating.

4. Queries about this assignment?

Should you have any queries about the assignment, feel free to contact

Fabio Ciravegna: {f.ciravegna}@ shef.ac.uk

5. Appendix A: Report Outline and Marking Schema

It is important that you produce a high quality planning report. Be sure not to leave the report at the very last minute. It is importance that you do not ignore techniques and examples provided to you during the lectures and lab classes. Referring back to them during your planning/implementation stages will give you the base from where to start, as well as a point of comparison/discussion where your ideas differ from what already presented to you (i.e., do not reinvent the wheel – if it has been done before, reuse it and complement it where it lacks functionality).
You may use diagrams to aid your explanation in these sections, but please note that a diagram alone is not acceptable and must be accompanied by an explanation/discussion.

Your report must be organised according to the following outline, which is designed to help you ensuring that all the required information is provided. **Length: maximum 6 pages.**

5.1. Overview of the report

This should give the marker a guide as to what to expect from your report. Please make the marker's work easier by being honest.
Create a description of your application explaining how you have met the requirements, i.e. how

- you have implemented a flexible sophisticated layout
  - describe how the different parts (e.g. Views in Android) compose the app
- the general organisation meets the requirements (i.e. it meets all requirements, some parts are missing, etc.).
- used separation of concerns (using MVP)
- stored data locally (using Rooms)
- used the sensors (GPS) and maps
- used async processes

Also make sure to:

- Discuss any deviation from the requirements
- Highlight any special feature of the solution that increases its quality
- Discuss any development additional to the strictly required by the assignment.

For each part in Sections 1.1.1-1.1.5
Create a subsection in your report for each of these parts. It is very important that they are documented individually by explicitly following the organisation below.

- Describe how the requirements (general and specific to the section) are met (e.g. separation of concerns, scalability to thousands of images, use of async processes, how the views are organised, etc.)
- Discuss the design: does it have advantages/disadvantages over other potential design choices?
- Discuss how you organised the separation of concerns if relevant
- Highlight any special feature of the solution that increases the quality
- Discuss any development additional to the strictly required by the assignment.

Please refer to the self-assessment form for more detailed points to cover in the report.

Division of Work

[No marks – MANDATORY] – Detail which part each member of the group worked on classifying the contribution as major or minor. Be precise. Please state if you feel the marks should be split equally among members.

Library Planned/Used

[No marks – MANDATORY] – Detail which libraries are going to be used (or have used) for the assignment

Bibliography

[no marks] **-** Do not forget to cite any sources and reference within the text where appropriate (e.g., "(...) *we have used the techniques as per [1].")*. Make sure to use a standard format style. The lecture notes must indicate the lecture number (e.g. week 1). Web links are fine.