# COM6510 Software Development for Mobile Devices

# Assignment report

* + 1. **Visually browse previews of photos**
* **Requirements**
  + BrowseActivity does this job. It has a RecylerView and three floating buttons for taking a picture, reading a picture, and filtering pictures.
  + The activity asks permissions in onCreate(), and does not set itself up using the setup() method unless all permissions are granted.
  + In setup, the activity starts an async task to scan the phone for photos. For each image found, the task enters a dummy entry into the DB, with the image’s location and the path where its thumbnail should be built.
  + The activity creates a live data observer from the room database at startup and when a filter is set. Images are added to the adapter using DiffUtil (to speed up performance) in the onChanged callback.
  + The file metadata is read and the thumbnail is actually built when the image is binded to the RecylerView, in an async manner. This enables thumbnails to be shown immediately even if the user suddenly scrolls far below in the list.
  + Gray tiles are shown instead of actual thumbnails if the thumbnail has not been created yet.
  + Tested for thousands of photos. On first app run, the thumbnails can flicker while scrolling until all thumbnails are built.
  + Images are sorted by last modified time, newer images shown first.
* **Design**
  + The design is minimalist. It has a consistent predefined app colors. It is a square grid has with 4 columns in portrait and 8 columns in landscape mode.
  + It has an app bar button to switch to map view, and hidden menu item to reset the app completely, deleting all databases and cache.
  + Layout is flexible and works in any screen size and orientation, since relative constraints are used instead of absolute values in the layout files.
  + Advantage - having square grids allows more thumbnails to be shown in the screen, reducing the need to scroll too much.
  + Advantage – putting the link to map view in the app bar, instead of creating a floating button.
* **Separation of concerns**
  + Separate method for testing and requesting permissions.
  + Separate method for setting up views.
  + Separate file for looking for images on phone.
  + Separate method for updating filters.
  + Separate Util class for common utility code.
* **Special features**
  + Splash screen that is consistent with the app icon.
  + Show gray tiles if thumbnails are not loaded yet.
  + Not running app until permissions are granted. Makes things simpler.
  + Auto hide floating buttons.
* **Additional development**
  + Hidden menu item to reset app completely.
  + Floating button to select a file from phone/cloud.
    1. **Showing pictures on a map**
* **Requirements**
  + MapsActivity does this job. It shows a map with markers indicating the position of pictures with GPS data, and a button for filtering images. It also has a thumbnail strip that can be clicked on to go to a particular marker on the map.
  + Images can be filtered based on their metadata.
  + Current location can be shown by clicking on a button.
* **Design**
  + Design is kept
* **Separation of concerns**
* **Special features**
  + Custom icon for markers.
  + Auto hide floating button.
* **Additional development**
  + Thumbnail strip at bottom
  + Filtering map points.
    1. **Inspecting the details of a photo**
* **Requirements**
  + Shows images with a black background.
  + Metadata can be seen by clicking on the “i” icon.
  + An additional app bar button appears when there is location data available, which shows the location on map.
  + Hidden menu item for editing image metadata.
* **Design**
  + Design is kept minimal, so that maximum screen space is taken by image.
  + No information is shown unless explicitly asked by user, making experience pleasant.
* **Separation of concerns**
* **Special features**
  + Clicking image makes it full screen.
* **Additional development**
  + -
    1. **Taking pictures**

**Requirements**

* Works on both real device and emulator.
* Floating button is hidden if phone does not have a camera.
* GPS data is stored in the image after images are returned by EasyImage.
* If run on emulator, UOS DCS coordinates are provided.
  + 1. **Saving metadata to local database**

**Requirements**

* All image data is stored in a Room database table called “Photo”.
* It has columns for image location, thumbnail location, file last modified timestamp, EXIF fields etc.
* Searching of images is done by setting a dynamic observer using the filter action buttons. Searching is possible by date taken, title, description, artist, phone manufacturer and model.

**Design**

**General organization.**

**Division of Work**

* **1.1.1 –** Mangesh (major)
* **1.1.2 –** Akshay (major), Mangesh (major)
* **1.1.3 –** Wenbo (major)
* **1.1.4 ­–** Mangesh (major)
* **1.1.5 –** Mangesh (major), Akshay (minor)
* **Documentation –** Akshay (major)
* **Screenshots/testing –** Wenbo (major)