University of Nottingham, School of Computer Science

Mobile Device Programming

COMP3040

Coursework 2: Runner Tracker

Report

Submitted by:

Tong Yew Jun

016927

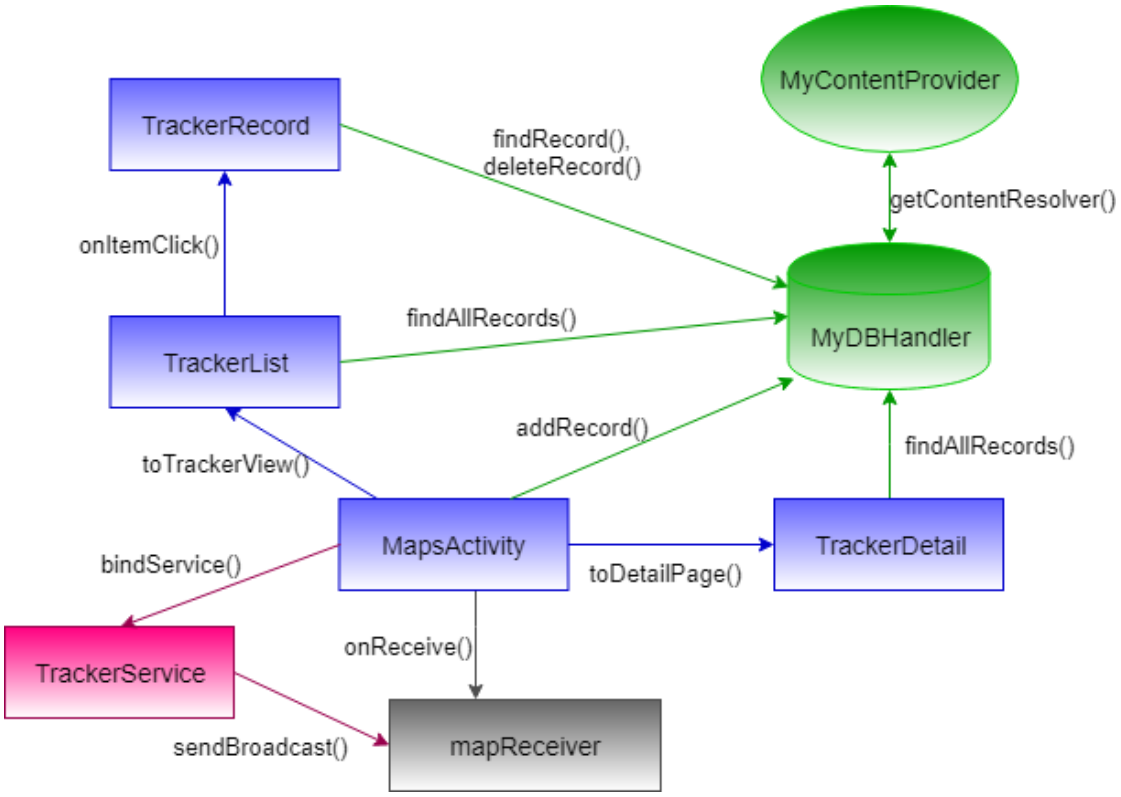
School of Computer Science

khcy6tyj@nottingham.edu.my

Features

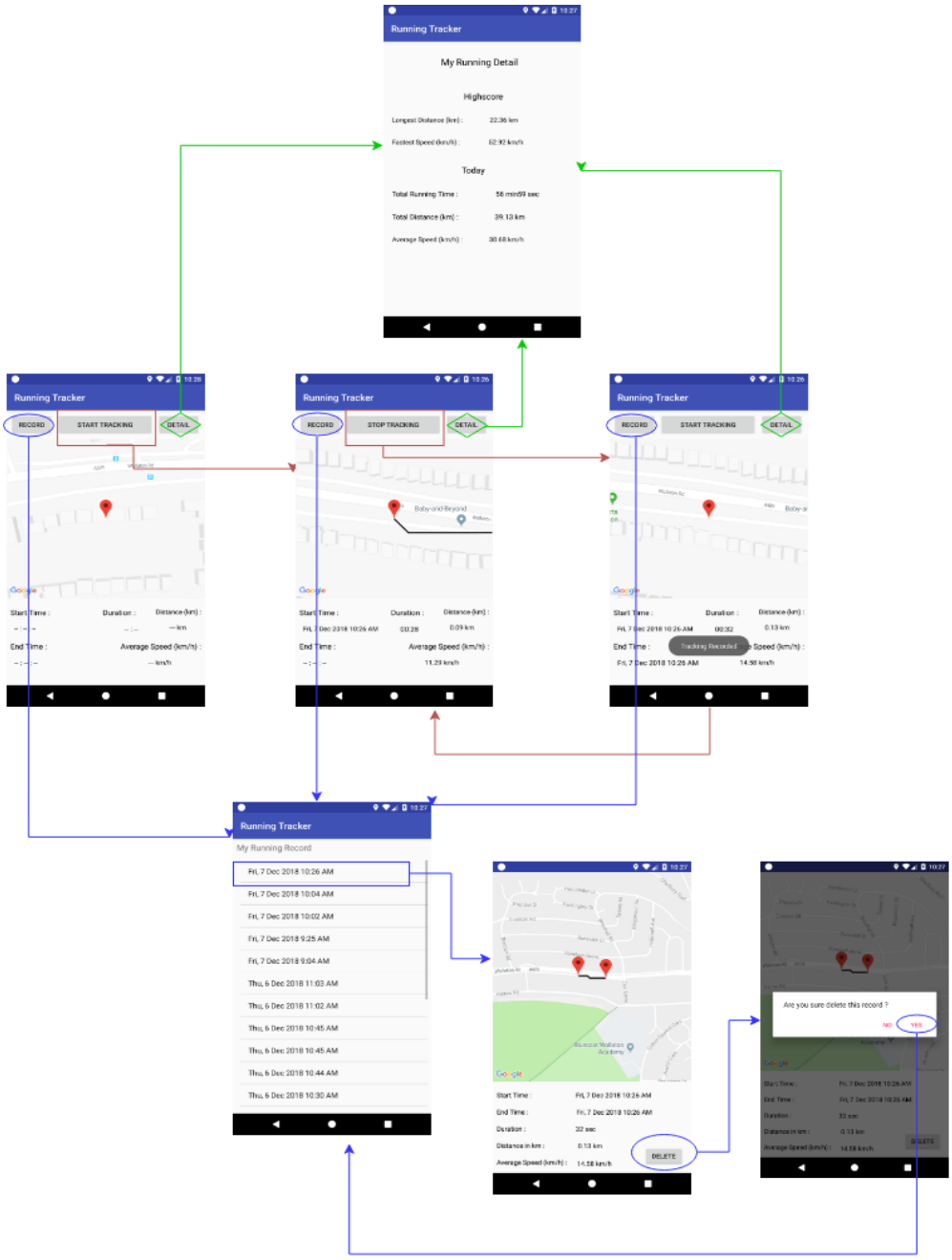
- Keep track of user current location before and after start tracking.
- Display current location on map and running start and end time.
- Record and calculate user running duration, distance and average speed.
- Allow the user to view the running record in a list arranged based on latest record.
- Allow the user to view recorded travelled pathway on map, start and end time, duration, distance, average speed.
- Allow the user to delete the records stored by application individually.
- Allow the user to view the longest distance and fastest speed.
- Allow the user to view today total running time, total distance and average speed.
- App notification activated when app started

App Architecture & Design



The diagram above shows the app architecture and design. The applications contain of activity classes, service, broadcast receiver, content provider and database.

User Point of View

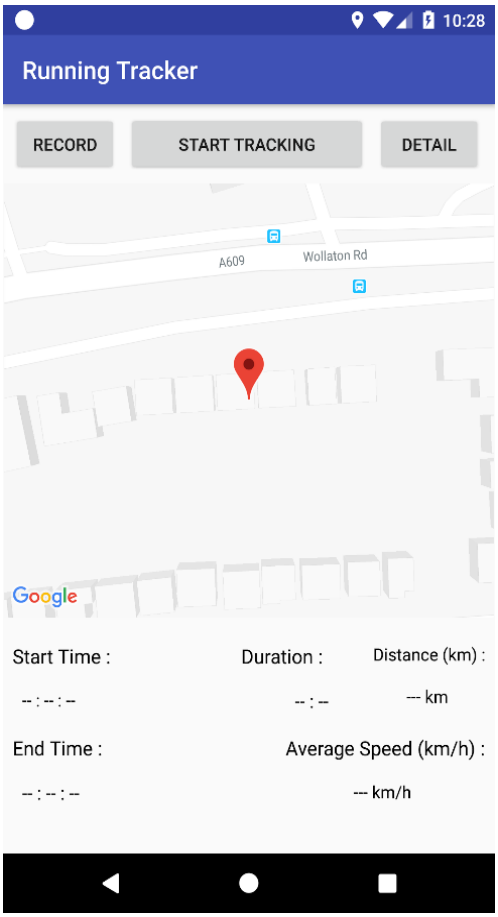


The diagram above shows the user point of view when using the app. There are 3 arrow colours:

1. Red - start and stop of running record
2. Blue - to view the list of record that available in database
3. Green - to view the details of the running record

Activity

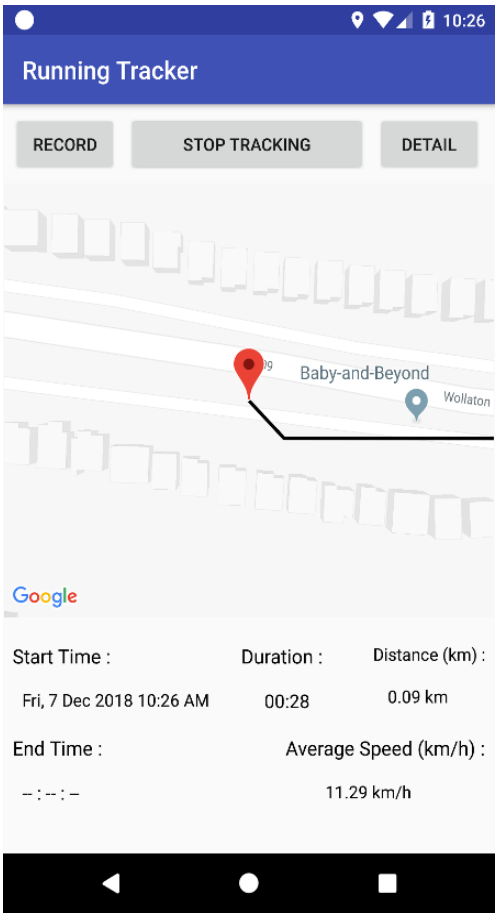
Maps Activity



The maps activity contains of 3 buttons on top, a map on middle and running details on bottom. The first button 'Record' will navigate to a page for viewing all tracking records stored in database. The 'Start Tracking' button will be used when user want to start record his/her movement. The 'Detail' button will shows the details of the running records.

While the google map will shows the user current location with a red marker. The marker will update based on the latitude and longitude of user current location on the map regardless the app is start recording. Running details on bottom will remain empty until user start recording.

‘Start Tracking’ Button Pressed

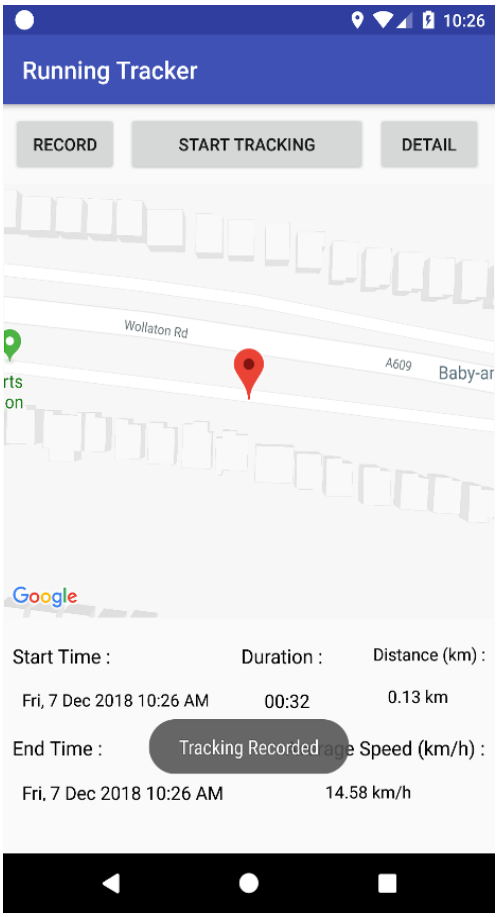


The ‘Start Tracking’ button will be changed to ‘Stop Tracking’ as the recording is started. The google map marker will keep update user current location by getting the latitude and longitude from *MyLocationListener* in *TrackerService* service class.

Every time user moved from a point to another, it will draw a line that can be generated by using *addPolyline* function by inserting the latitude and longitude location value to indicate the pathway of user travelled.

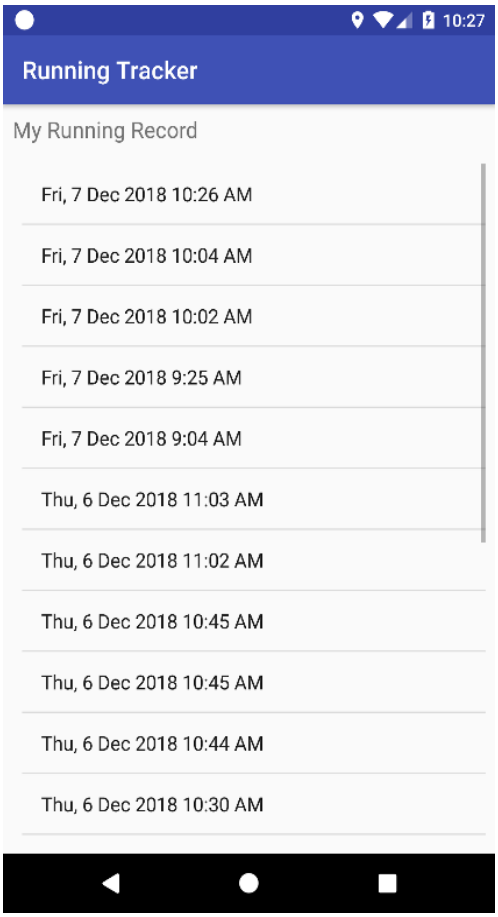
While the start time can be generated by using *Calendar.getInstance().getTime()* function when user pressed ‘Start Tracking’ button. The running duration will be counted by every second using *scheduleAtFixedRate*. The distance will be calculated using *loc1.distanceTo(loc2)*, where the loc1 is the current coordinate and the loc2 is the previous location. The function will return meters for the distance travelled. The average speed can be calculated by dividing distance (km) by time (hour).

‘Stop Tracking’ Button Pressed



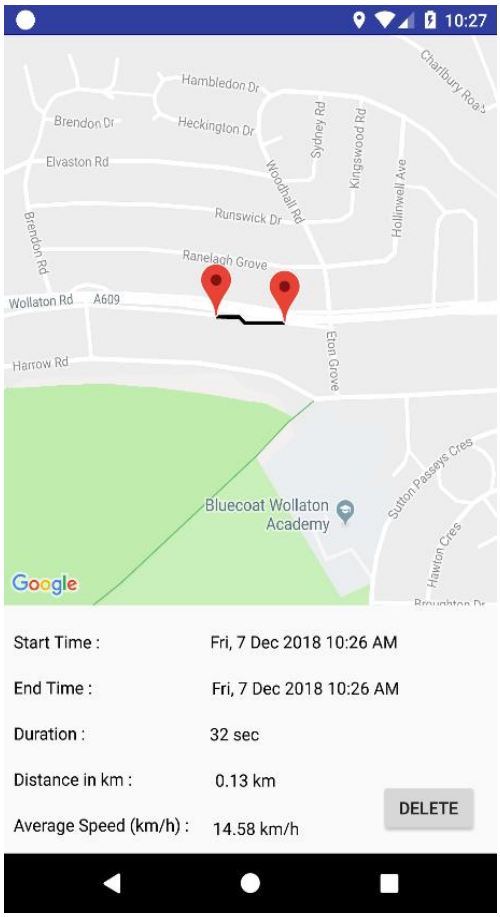
The end time will be recorded by using *Calendar.getInstance().getTime()* function to get the current end time. The tracking recorded data will be inserted into database table including start time, end time, travelled duration, total distance, average speed and travelled coordinates.

Record Page with All Records



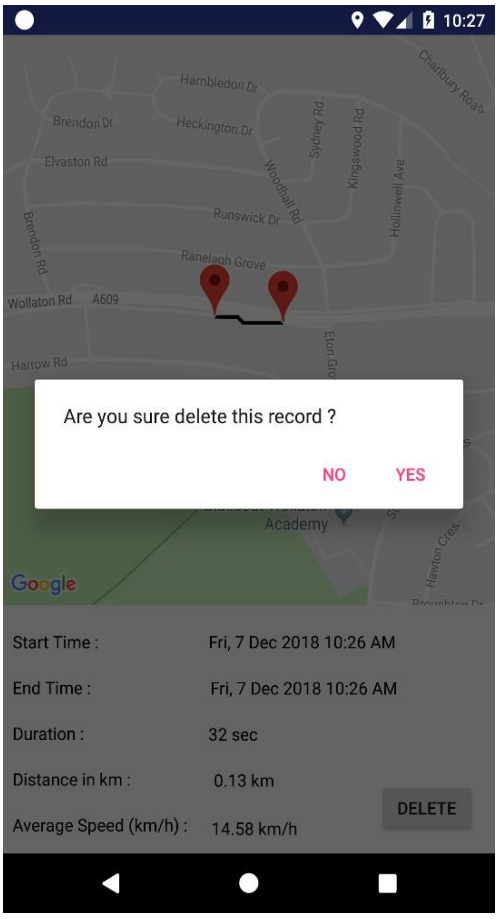
When user pressed 'Record' button in *MapsActivity*, it will shows a list of running record that sorted by latest running start date and time. If there is no running data recorded in database, the list will shows 'No Record Found'. All the start date and time is query from the database by using *findAllRecords()* function in the *MyDBHandler* class.

Record Chosen from List



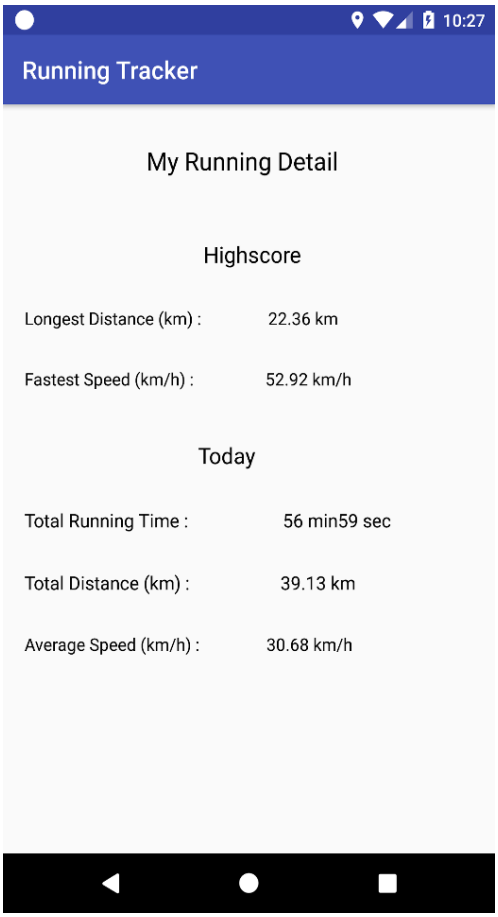
The google map will shows 2 markers which are the start tracking and end tracking location. The line in between the 2 markers is the pathway travelled by user when recording. The bottom of the page will shows the details of the running record including the start time, end time, duration, distance and average speed requested from the database. User can choose to delete the record by selecting the ‘Delete’ button on right bottom of the page.

‘Delete’ Button



A dialog interface will pop out to double confirm deleting the specific record. By selecting ‘No’ the record will remain unchanged on the *TrackerRecord* page. If ‘Yes’ is selected, the specific record will be deleted from the database and it will go back to *TrackerList* that shows list of all record.

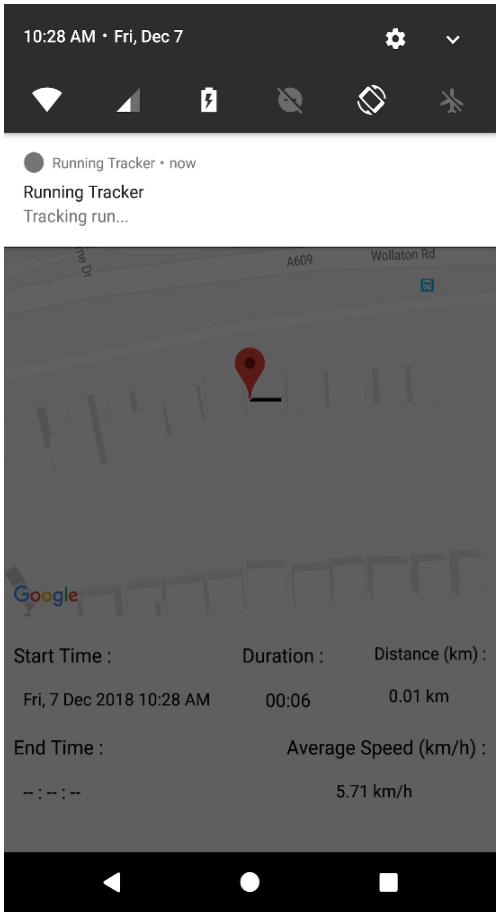
Detail Page



The *TrackerDetail* page will retrieve and calculate the running data recorded in database. The highscore column will shows the longest distance and fastest speed for all running record. The longest distance and fastest speed can be calculated by comparing the distance and speed in the database and get the largest value.

While in the 'Today' column will shows the total running time, total distance and average speed recorded by today. Sorting the data to get today data by using *dateFormat.format(record.getStartdatetime()).equals(dateFormat.format(Calendar.getInstance().getTime()))* to compare the start date. Add up all the distance and time to get today total running time and total distance. Average speed can be calculated by adding all today speed and divided by the number of data added.

App Notification



The app notification will shows when the *MapActivity* service connection is connected and bind in *TrackerService* class. By calling *bindService()*, the *TrackerService* class will build the notification and shows on status bar.

Service

A service class named *TrackerService* will track the user location (latitude & longitude) and send the location through broadcast. The service is started on *MapsActivity' onCreate* that called the *bindService()* function. When the *MapsActivity* is binder to the *TrackerService*, it will creates a notification to show the status bar of the device. The service will allow the app run on background and notification will works as the indicator to shows user this app is running on background. The service will unbind when user quit the app and the location manager will remove the update of the location listener and notification will dismissed.

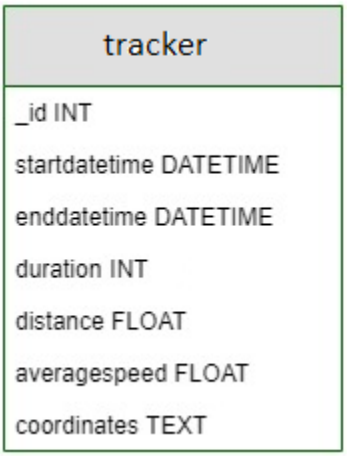
Content Provider

A content provider class named *MyContentProvider* have implemented insert, query and delete function for facilitating the sharing of data with other application in the future. The URI "*com.example.gpsmap.provider.MyContentProvider*" allow other applications access the data stored in the database.

Broadcast Receiver

A broadcast receiver class named *mapReceiver* is implemented in the *MapsActivity* class to receive the broadcasted location from *TrackerService* class. When user's location have changed, the location listener in service class will triggered and send the updated location through broadcast. The *onReceive()* method in *mapReceiver* get the location in the form of intents with the action "*com.example.gpsmap.LOCATION_RECEIVER*" from *TrackerService* and update the marker of the user location on the map.

Database Structure



The database table name called tracker and consist of 7 column which are the id, start date time, end date time, duration, distance, average speed and string containing all latitude and longitude coordinates. Every new row will be added to the table when user finish tracking the run and the row will be deleted when user delete the specific row.