**CRM App Documentation**

Created by Daniel Garcia on 10/Aug/2017

**Overview:**

The app’s main purpose is to allow a user to keep track of their connections all in one place. The app keeps track of the user’s collaborators, showing their contact info, recent activity, and other pertinent information. The app also allows the user to quickly add new collaborators, record interactions with them, and update any information regarding a particular collaborator.

**Important Server Connection Information:**

There are 6 classes whose use should be noted;

KardiaFetcher: This class uses the Kardia API to get information from the server

KardiaProvider: This class uses SQL commands to get information from the app’s local database

CRMContract: This keeps all the constants used in the local database (which are all kept in SQL tables)

CRMOpenHelper: This simply creates the local database and all its tables

PostJson: This class uses OKHttp to connect to the Kardia server and takes care of uploading any new information

PatchJson: Much of this class is similar to PostJson, except it updates or patches existing information rather than posting new things

**Main Features:**

Profile Display: This activity displays a collaborator’s profile picture, contact info, and timeline (which includes any recent activity including interactions, prayer requests, etc.).

New Interactions: After any phone call made or email sent from the app, the user is asked whether or not to record that interaction. An interaction can also be manually added. Interactions keep track of the type of interaction (phone call, in-person, update, etc.), date and time of interaction, subject, and any relevant notes. Optionally, a follow-up can be scheduled (along with a brief note) that will create a notification for the user to remember to follow up on their interaction with the collaborator.

Engagement Tracks: The Engagement Activity lists all current collaborators actively in an Engagement Track, along with a brief comment. Clicking on a track leads to a detailed view that shows the track, the step within that track, a brief comment, and a longer description. The user has the option of moving the collaborator forward to the next step, or simply to edit and save the comment and description.

Person Search: The Action Bar handles the basic Search Bar while a third-party library called Material Dialogs helps create dialog to show the results of the search and should be customized to each activity. The search functionality on the Main Activity allows an existing person in the Kardia server to be added as the current user’s collaborator. The Engagement Activity’s search functionality allows any collaborator to be placed in a new Engagement Track.

**Future To-Dos:**

The various activities (especially the Profile Activity) tends to take some time to load, so it would be good to add a progress bar or a loading bar while an activity is gathering all its information.

Once multiple users start adding new people to Kardia, it’ll be necessary to check for duplicates, to prevent anyone from being added twice.

On the web version of the New Interactions feature, a user can specify what contact was used along with the type of contact (so a user can specify which phone number was called or which email address used). However, this information does not seem to get displayed or used anywhere else, so it wasn’t included on the app. If so desired, that can be added to the Interaction Activity. The XML file for it already has a spinner there, but it is never used anywhere.

A future possibility for Engagement Tracks is to add requirements necessary for someone to advance to the next step in the track. This would most likely simply consist of multiple check boxes in the detailed track activity.

A decision needs to be made in how the EngagementActivity starting collaborators on new Engagement Tracks wants to be implemented. Currently the search bar could be used to search for a user on the server and then start a new Engagement from a dialog. Another option might be to use the floating action button for this purpose and instead use the search for completed tracks or something else entirely. The EngagementTrack branch which is in progress and untested was made with the intention of using the search bar.

The app has some UI changes that need to be made. Firstly, the app lacks UI consistency because we did not have time to properly theme and design good layouts. We (tparr & Galhonne) think the app would look very good adhering to the Android Material guidelines set by Google. Secondly, we have also neglected to test the app on different screen size and will likely have visual bugs.

Testing for all API levels. We have only been testing features on API 19 & 25 both Emulator and with physical devices.

We would like to include some way of invalidating data stored in the database frequently because the app needs the newest data. We do not have anything that checks to update the database unless it the first time a user has logged in.

**Issues/Bugs:**

When dealing with notifications, an issue was found where the delete and query functions in the Kardia Provider were not properly working. The “selection args” parameter is supposed to be substituted in where there is a question mark (“?”) in the previous argument (which in SQL becomes the WHERE command). However, the substitution is not happening, so the statement reads something like “DELETE FROM Notifications WHERE partnerID = ?”. A simple workaround was done where the “selection args” parameter simply got placed directly, rather than being substituted into the question mark. The problem with this solution is that it is significantly less secure, since the “selection args” parameter makes sure no injection statements can be done (Greg knows more about this).

A smaller bug is that when the user advances a collaborator to the next step in an engagement track, a “Network Issues” message pops up, even though the step is advanced successfully. The problem is with the server and it failing to properly name the step completed that was pushed to the server. (Greg has been informed of this server issue).

At inconsistent times, the server will drop our connection especially when performing longer requests. Although this may be a bug in the app we think it more likely to be caused by the Server since interns partially wrote its code.