GSoC 2020 : OpenMRS Android Client 2.9.x Project

<table>
<thead>
<tr>
<th>Primary mentor</th>
<th>Fawwaz Yusran</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup mentors</td>
<td>CLIFF GITA Vibhor Chinda</td>
</tr>
<tr>
<td>Assigned to</td>
<td>Rishabh Agarwal</td>
</tr>
</tbody>
</table>

Abstract

The goal of the Android client is to provide an alternative to access a hospital's OpenMRS instance by just using the provider's Android devices. See the full Android client guide for more info.

Considering the services offered by a hospital, an Android application can help doctors, patients, and other staff a lot with its mobility and ease of use, without them having to start the OpenMRS web app on a desktop computer. This will improve the productivity and efficiency of the hospital workflow.

This year, we will focus on two aspects to improve the Android client - **performance** and **UI/UX**.

There are still frequent crashes when using the app, especially if a doctor is using it for a long time with no pause. On crashes, the app will just return the user to the dashboard page, and any unsaved transactions will be lost. Also, the **offline-first** solution needs to be developed, where a provider cannot connect to the internet and needs to work offline temporarily. Data should be synced again when back online.

Next, the current user interface is slow to respond to user inputs and does not look pleasing to the eye, compared to the newer and modern Android apps in the google play store. E.g. 'Undo' buttons (that takes the user back to the previous state if he/she errors) should be available for any operation. So we should enhance the feel of the app so that the user knows that he/she is in control every time.

Finally, seeing the number of components in the web app, the Android app should integrate more of them so that doctors and staff can expect to see the same functionalities in the web app implemented in the client.

An additional request is to migrate the Java source code to Android's now recommended Kotlin, to increase readability and reduce a thousand lines of code present in the GitHub repository.

Objectives

1. Material UI design ([Figma Link](#))
2. Integrate more components of the web app to the client (to be discussed further)
3. Password reset via email
4. Replace Java code with Kotlin
5. Replace native SQL to Room

Future Works that need to be done

1. Migrate the remaining 80% codebase to kotlin.
2. Start converting MVP pattern to MVVM as it follows the new standard guideline brought by Google.
3. Re-design Home screen with a better UI.
4. Add the password-reset feature.
5. Create a diagnosis using REST.
6. Implement the Appointment Scheduling Module.
7. Mark Patient deceased using a non-coded response.

Sample Use Cases

1. A doctor does not have access to a PC, but his Android phone is available to use during office hours.
2. No internet connectivity and a doctor needs to work offline temporarily.

Project Champions

Same people who use the web app: Providers (e.g. doctors, nurses, clinicians)

**Skills Needed**

Android, Java, Room, Kotlin

Progress

Issues that are completed and their status:

Timeline
Week 1

Blog post

Tasks completed:

- **AC-689** - Re-design Settings Activity  [CLOSED]
- **AC-665** - Re-Design Login Activity  [CLOSED]
- **AC-763** - Re-design Splash Screen  [CLOSED]
- **AC-764** - Re-design Contact Us Screen  [CLOSED]
- **AC-767** - Add Roboto Family and update FontUtils to change the app Font  [CLOSED]
- **AC-770** - Migrate ContactUs Screen to Kotlin  [CLOSED]
- **AC-450** - Add an Admissions form  [CLOSED]

Week 2

Blog post

Tasks completed:

- **AC-778** - Tweak ToastUtils for better UI appeal  [CLOSED]
- **AC-779** - Re-design Dialog Layouts  [CLOSED]
- **AC-783** - Migrate Dashboard Screen to Kotlin  [CLOSED]
- **AC-777** - Migrate Introduction Screen to Kotlin  [CLOSED]
- **AC-780** - Migrate Active Visits Screen to Kotlin  [CLOSED]
- **AC-782** - Update Readme.md file for Android Client Repository  [CLOSED]

Week 3

Blog post

Tasks completed:

- **AC-780** - Migrate Active Visits Screen to Kotlin  [CLOSED]
- **AC-776** - Migrate Logs Screen to Kotlin  [CLOSED]
- **AC-785** - Migrate FormEntryPatientList Screen to Kotlin  [CLOSED]
- **AC-787** - Migrate Settings Screen to Kotlin  [CLOSED]

Week 4

Blog post

Tasks completed:

- **RA-1790** - Modify MarkPatientDeadPageController to expect a uuid or integer  [ACCEPTED]
- **RESTWS-777** - Add causeOfDeathNonCoded as the updatable Property in REST  [CLOSED]

Week 5

Blog post

Tasks completed:

- **AC-799** - Mark Patient Deceased Using Coded Response  [CLOSED]
### Week 6

**Blog post**

Tasks completed:

- AC-791 - Re-design Registration Screen **CLOSED**
- AC-766 - Update dependencies and target SDK for the project **CLOSED**
- AC-794 - Notify User to restart app when form list is blank **CLOSED**

### Week 7

**Blog post**

Tasks completed:

- AC-804 - Show allergies stored in server in patient dashboard **CLOSED**

### Week 8

**Blog post**

Tasks completed:

- AC-817 - Create FormEntity and update other entity classes **CLOSED**

### Week 9

**Blog post**

Tasks completed:

- AC-814 - Re-design Patient Dashboard Screen **CLOSED**
- AC-596 - Remove Unused resources **CLOSED**
- AC-805 - Add offline support for allergy module **CLOSED**
- AC-816 - Re-design Admission form Screen **CLOSED**
- AC-824 - Release builds crash on startup, debug builds run fine **CLOSED**
- AC-823 - Overload converter methods in AppDatabaseHelper **CLOSED**

### Week 10

**Blog post**

Tasks completed:

- AC-810 - Delete an existing allergy **CLOSED**
- AC-808 - Migrate Form Admission Screen to Kotlin **CLOSED**
- AC-664 - App restarts on downloading concepts **CLOSED**
- AC-809 - Add a new Allergy based on server configuration **CLOSED**

### Week 11

**Blog post**

Tasks completed:
Week 12

Blog post

Tasks completed:

- AC-836 - Re-design Add provider screen  CLOSED
- AC-829 - App crashes at form admission in dark mode  CLOSED
- AC-840 - Better UI for Visit Dashboard Card Views  CLOSED
- AC-841 - Better UI for Chart Fragment Dashboard Activity  CLOSED

Dev Tracks

- Project Issue Tracker (JIRA) - https://issues.openmrs.org/projects/AC/issues
- GitHub repo - https://github.com/openmrs/openmrs-contrib-android-client

Resources

- OpenMRS Android Client Project 2.8+ - Previous year's GSoC page for this project
  - https://github.com/openmrs/openmrs-contrib-android-client - main repo