**Data Collection App Template Documentation**

By Ahamad Imtiaz Khan

This template is basically a renovation. Not much new things are added with this template. If a programmer is familiar with the previous template it is quiet same. The following things are added with this template to make it more user-friendly. Details about the points will be described later.

1. Some forms are updated. As I am not familiar with all kinds of forms used in all types of app so all forms are not added here. Anyone can copy the code for non available forms with this template easily. Most frequently used forms are available in this template.

2. Sometimes the questionnaire is long. User may want to edit the questionnaire by Section. Previously boundary of sections were defined by question serial number so if any question is added or removed then lots modifications were needed. It has been made easier for programmer.

3. Normally when any user Click Edit option. All DataIDs are shown in a list. An option is added so that normal user can see the IDs which are added within last 5 days. So they can only edit those IDs. If anyone login with "admin" they can see and edit all DataIDs. This option can be disabled in code.

4. For transferring data a separate application was used. That one is merged with this application. So now on user can transfer data from same application.

5. Database copier tool was a separate application. It is also merged with this application.

6. Automatic data transfer/ data backup after regular interval option is introduced in this app.

7. Database encryption is introduced in this application.

**Brief Description**

1.If frmmultiplechoice form's check box has other option then a text box is appeared. Options added to make those text box numerical text box or can use those text boxes to pick date or time. Same options are added to frmmultiplecheckcombotwo.

Hint: Inside loadGuiFrmMultipleChoice look for **if**(**qName**.equalsIgnoreCase(**""**))portion.

A new form named frmcamera is added. Using this form user can capture photos. The Photos will be stored in phone memory inside "icddrbDB/<APP NAME> " folder. The location of the last taken photo of a specific form will be stored in DB and that last photo will be shown when someone open that form.

2. For showing Edit option by Sections Go to MenuScreen.java class and inside loadGui() function Comment the following portion of code

///Edit mode with all questions  
/\*CommonStaticClass.mode = CommonStaticClass.EDITMODE;  
CommonStaticClass.subEditMode = 0;  
Intent i = new Intent();  
i.setClassName(CommonStaticClass.pName, CommonStaticClass.pName  
 + ".EditEntry");  
startActivity(i);\*/

and Uncomment the following portion of code

*//Edit mode with Sub section*CommonStaticClass.*subEditMode* = 1;  
clearEveryThing();  
Intent i = **new** Intent();  
i.setClassName(CommonStaticClass.*pName*, CommonStaticClass.*pName* + **".subedit"**);  
startActivity(i);

If you don't want to show the edit mode my section then do the opposite.

Important: The application searches the boundaries of sections by using keyword "sec" so each section must start with a frmmessage and the variable name must be start with "sec".

3. At present any user can Edit any Entry. In EditEntry.java class inside loadDataToList() function the following code is added so only admin can edit any Entry and other user can see and edit the data of Entries that are created within 5 days.

**if**(mCursor2.moveToFirst() &&   
 mCursor2.getString(mCursor2.getColumnIndex(**"Name"**)).equalsIgnoreCase(**"admin"**))  
{  
 **dID**.add(mCursor.getString((mCursor.getColumnIndex(**"dataid"**))));  
}  
**else**{  
 String sDate1 = **""**;  
   
 SimpleDateFormat df = **new** SimpleDateFormat(**"dd/MM/yyyy"**);  
 sDate1 = mCursor.getString((mCursor.getColumnIndex(**"EntryDate"**)));  
 Calendar c1 = Calendar.*getInstance*();  
 Date startDate = df.parse(sDate1);  
 c1.setTime(startDate);  
   
 **if**((System.*currentTimeMillis*() - c1.getTimeInMillis()) < (24\*60\*60\*1000))  
 {  
 **dID**.add(mCursor.getString((mCursor.getColumnIndex(**"dataid"**))));  
 }  
}

If anyone wants to go back previous state i.e any user can see and edit any entry the Comment the code and just keep the following line.

**dID**.add(mCursor.getString((mCursor.getColumnIndex(**"dataid"**))));

4. The classes for Data transfer is added under "datatransfertool" package. Earlier when the users sent data all entries of the local database were sent to server. But now the data that were not sent earlier will be sent. For this purpose a new column is added in table called "isTransferred". Every table that contains data to be sent must have this field along with other mandatory fields. For updating this field a function is added called updateIsTransferred(String sql). If any entry is inserted or updated then "isTransferred" is set to "0".

5. Database copier is added under "schedulebackup" package. When there is no internet connectivity in tab and the user tries to transfer data data is backed up in external SD card inside icddrbDB/<APP NAME> folder by date. If external SD card is not available then database is copied in phone memory.

6. Every time a user logs in an automatic data transfer/backup activity is initiated. If anyone wants to disable this feature the comment the code stated below in MenuScree.java-> onCreate() function:

Intent alarmIntent = new Intent(this, ScheduleBackup.class);  
alarmIntent.putExtra("dbpath", DatabaseHelper.DB\_PATH);  
alarmIntent.putExtra("dbname", CommonStaticClass.DB);  
alarmIntent.putExtra("dbpass", DatabaseHelper.getpw());  
pendingIntent = PendingIntent.getBroadcast(this, 0, alarmIntent, 0);  
  
startAlarm();

Inside startAlarm() function the interval for auto transfer/backup is set to 24 hours. you can change it.

7. Database encryption is introduced in this application. All the functions regarding database operations are of sqlcipher library. If you want to use encrypted database for your app the Encrypt the database using Another app "EdCryption". If you Encrypt the app using EdCryption app then the encrypted app will be stored inside "Edcryption/Encrypted DBs" folder of phone memory. Copy the DB from this location to Root location of phone memory. Now go to the end of DatabaseHelper.java class of your application. There is a function called getpw() which return empty string "". Set the password you used to encrypt your DB using EdCryption app in return statement.

If you don't want to use encrypted database then keep the return string as it is (empty).

For Running EDCryption App Please keep "dbencryption.sqlite" in the root of your phone memory. The names and passwords of Encrypted Database using Edcryption app is stored inside this app. Please keep this db private as this db is not encrypted!! (what an irony)