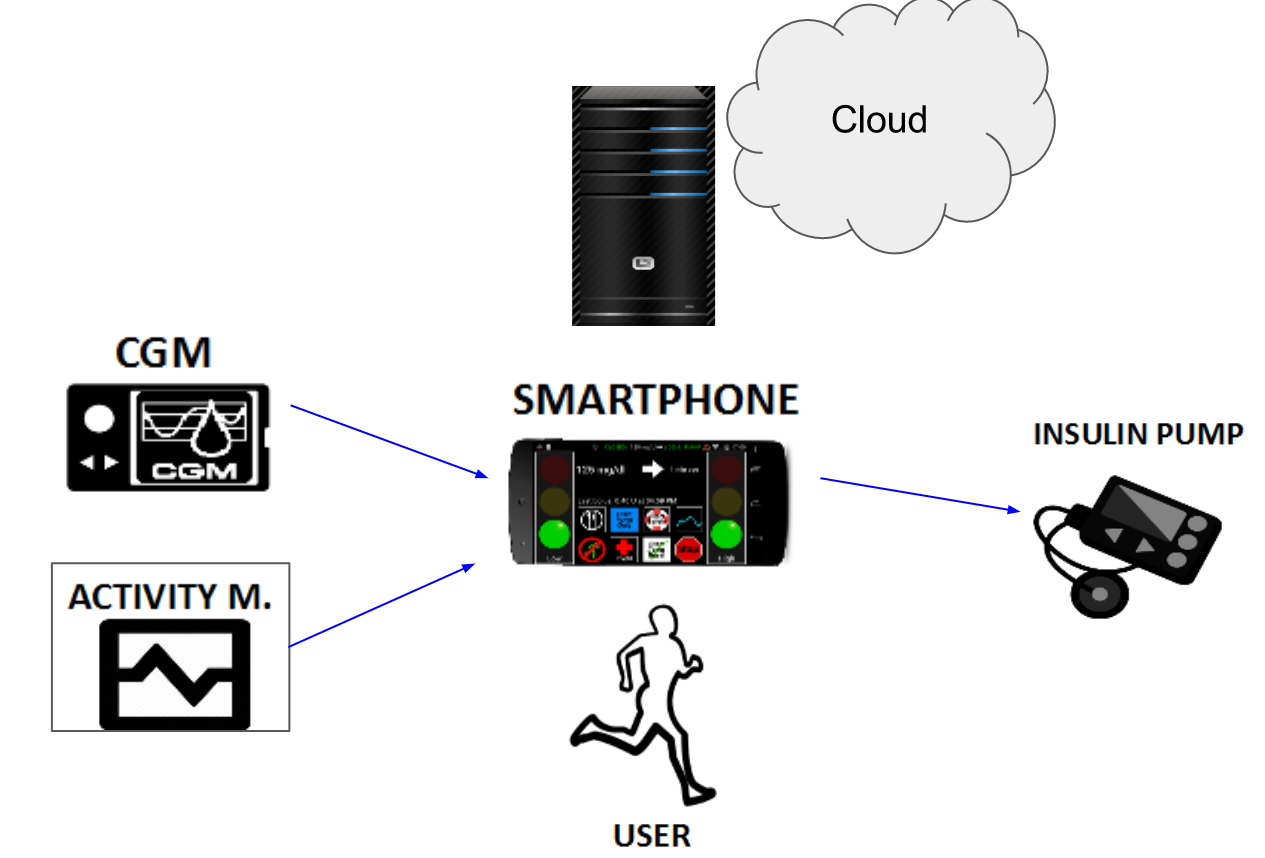
***Artificial Pancreas***

***Overview:***

Artificial Pancreas is basically nothing but the automated insulin delivery system or an autonomous system for glycemic control. So, here we are trying to develop an android application which has following requirements:

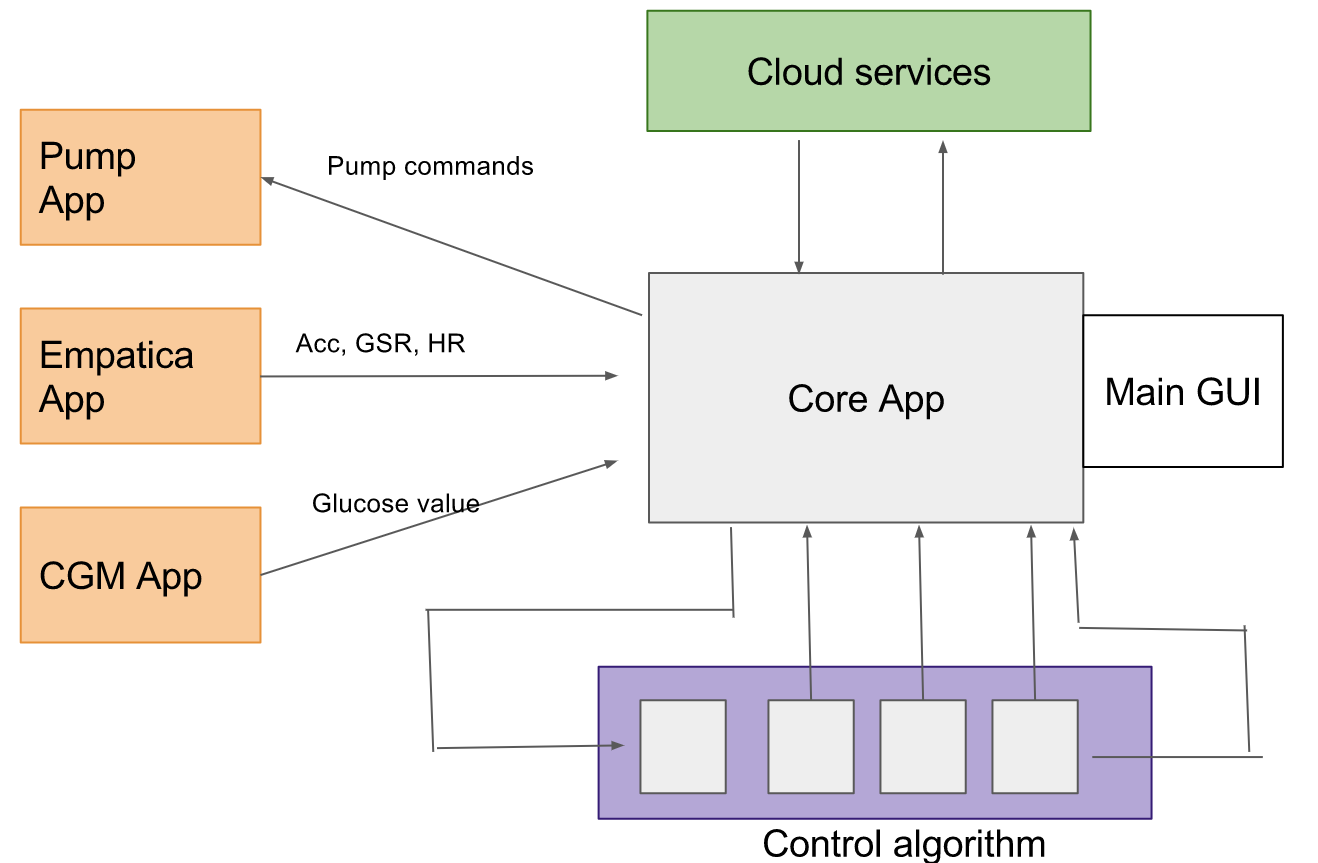
* Android application individual implementation
* Android service implementation
* Android applications interaction:
  + Applications intercommunication
  + Applications integration
  + Shared resources
* GUI user friendly design
* Display
  + CGM, Current person state, Insulin info
  + Coherent screens/activities transition
  + Connection state
  + Alerts
* User management - locally (in the phone) and remotely (in the cloud)

***Architecture:***



***System Requirement:*** Windows 8 or higher, Android Studio 3.0.1

***Android Flow:***



The actual flow for **Content Provider**: Flow can be two way communication.

Application

SQLite Database

Content Provider

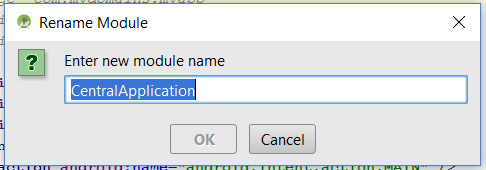
For further information you can access:

https://developer.android.com/guide/topics/providers/content-provider-basics.html

***Three Android Applications:***

**Central Application** (This is our Mock\_CGM Application): I started with this name and now If I change the name of the application then the name has to be changed at all locations i.e. Main Application in Android Manifest and also in the URI’s used in RandomProvider class.

To rename the application you can need to right click on the application and refactor -> rename. The following dialog box will appear. This will change the name and refactor things in the whole application but for different applications such as Main application and Tandem\_Pump application there are permissions given in android manifest.



Now the main function of this application is, to generate the random numbers which in future will be the glucose values at the duration of every 5 minutes, but for testing purpose random numbers are generated at every 30 secs.

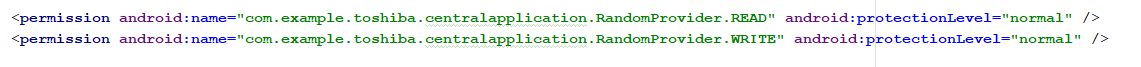
Now when this random numbers are generated they are sent to the main application. To send the values from one application to another, I used the content provider. As soon as you open the project in the android studio, there are four main files for this application:

**MainActivity**: It has the function to generate the random numbers and also retrieves the number generated on this page.

**RandomProvider**: Here the actual content provider is defined.

**MyDBHandler**: Here we define all the sqlite queries and also define the URI’s for tables which will further be passed to RandomProvider.

**Android Manifest**: To give permissions we add some user permissions to read and write the file.



**Main Application** (This is our Core Application):

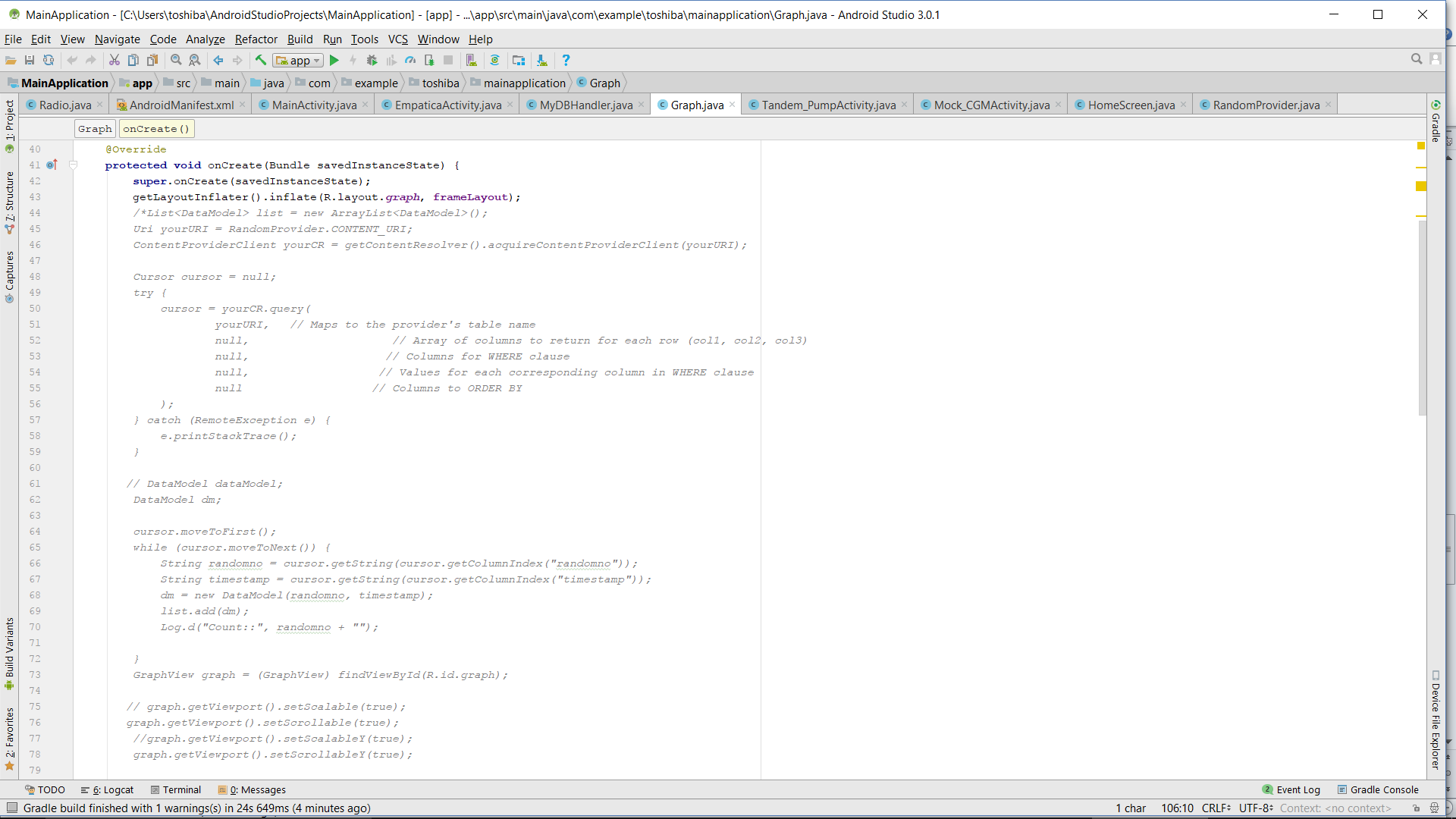
This is the main application which will manage all the other applications. All applications can read and write data from this application. For now we have the main application and Tandem\_Pump but in near future we will also be working on server and Empatica application.

Now for this application we have following java files which play a major role for application:

**MainActivity:** Here it basically creates the view for navigation drawer and also for the home screen.

**MyDBHandler:** Database handler has the same role to perform as in the central application.

Graph: For now I have added the dummy values. Random number generator has very wide range of data and also apart from this, the data is not consistent. So to get the actual graph I have commented the portion as seen below:



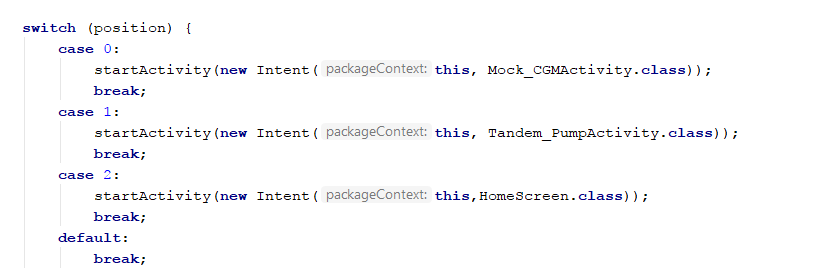
**Tandem\_Pump Activity:** Here for now I have added the alert box so that it can navigate to other application. From here it can navigate to Tandem\_Pump.

**Radio:** I have kept the radio buttons to select either the graph or list. The list shows about the history of random data generated and similarly for graph the data is shown in the line graph form.

**Mock\_CGMActivity:** Here for now I have added the alert box so that it can navigate to other application. From here it can navigate to Mock\_CGMActivity.

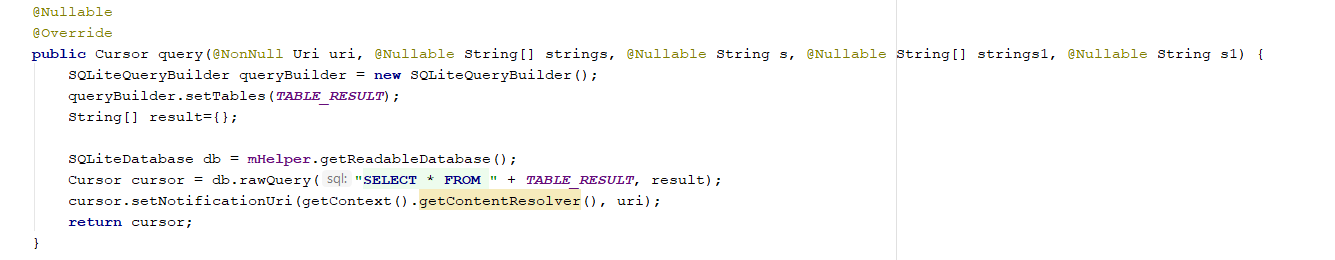
To add the Items in the navigation drawer in MainActivity: Add the list of items that need to be added to this array.



And after that in the MainActivity itself you need to add in the switch case as shown below:

**Tandem\_Pump** (This is our Actual Pump Application):

Random Provider: All the content provider methods and variables are defined and called here. This is the main method of content provider. Here what we do is define a cursor which accesses the database to select the data from the Table\_Result where in all flag values are stored. Now also, we can use this query method for different purpose. Like in main application I have used query method to simply fetch and read the data one by one.



**Android Manifest**: For the user to access the database of other applications we need to give read and write permission of the application which creates and insert the values of data in database.

**Flag:** Here I define one text view so that you can see the flag value. I have also defined the cursor here as the flag value is being read from database. In any case, if you need to access data then you need to define the cursor.

**activity\_main:** Here the actual xml code is written for the android gui.

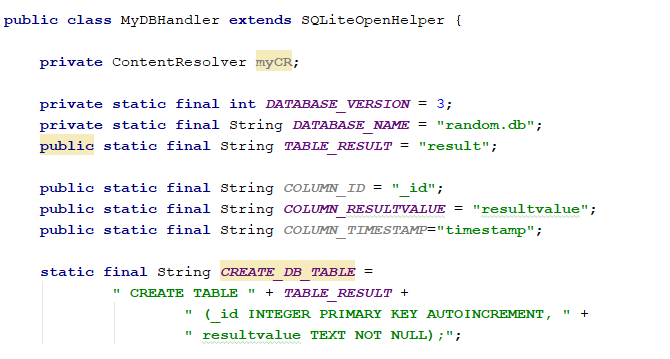
**Lastly I would like to give an overview about the steps when created new application to be integrated to Main Application:**

You can simply follow the steps for Tandem\_Pump application or else you can see below:

* **Create the project.**

Create a new project in the Android Studio as a blank activity.

* **Create Database Classes**



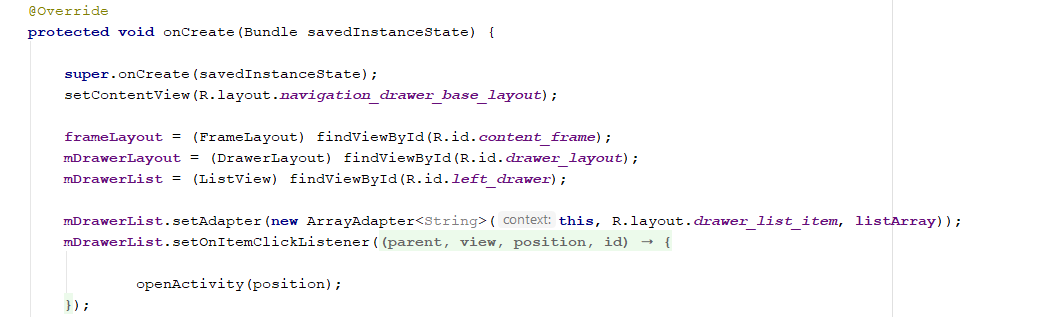
Once you have defined the database handler class, you need to define all the variables in here as shown above.

* **Resources and Layouts:**



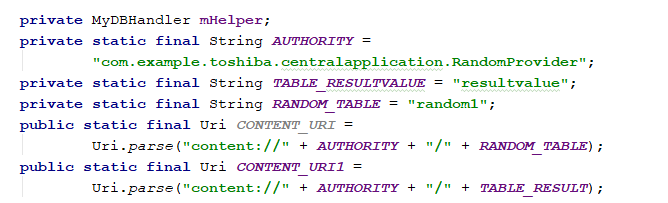
* **Create MainActivity which extends the Activity:**

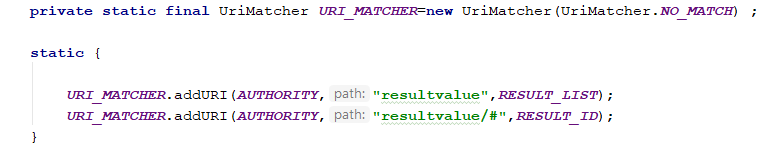
The below is an example what we can do in the main activity. It is the example of navigation drawer. Once you have defined the resource and layout in xml then you need to bind it with java, so you have to define all the text views, buttons and any other widgets that you have added in the resource files in MainActivity.



* **Modify Content Provider:**

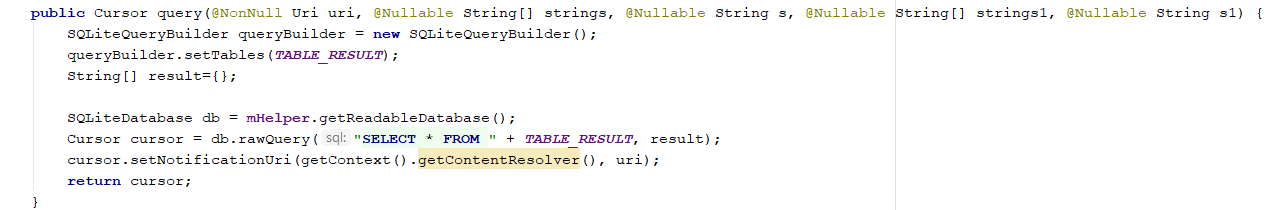
Create Content Provider: Create a Content Provider class. Define the URI and Authority in the random provider class. The Authority and URI should be provided for the application where the content provider class has been defined. There will be only one content provider class in the application which will create the database, table and insert the values in SQLite database.



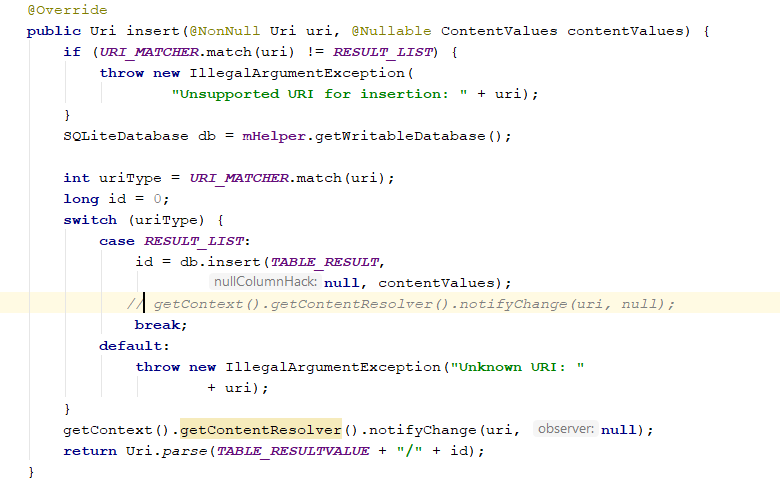
Once you have defined all the variables now you have to add the URI and Authority in the method 

The above are the table names which we have defined in the MYDBHandler, so that the content provider knows exactly which table content has to be accessed.

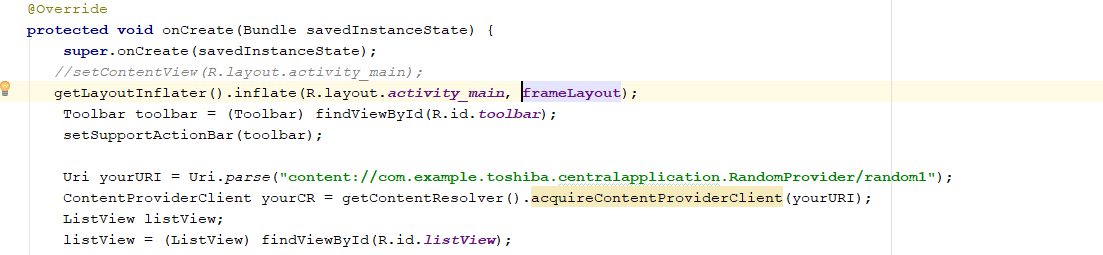
Once you have all your variables and access to table now you need to create a method to get all values from the database.

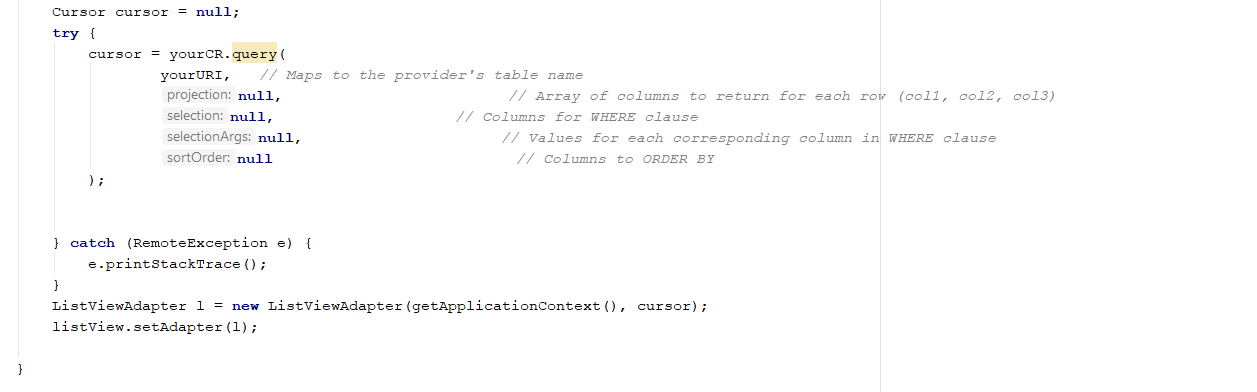


If you have to retrieve data then this method is enough but if you want to insert the data in the table or delete it or update it then you have to define the content in the inbuilt methods insert(), update(), delete(). For inserting into the database we need to define as follows:



Now, last if you want to create a new class and use some values from database then you need to define a cursor and access the database from that cursor. To do so,





***Future Scope:***

In near future we are planning to build the GUI of the main application. Once we get the Dexcom g5 API we can work on integrating the application with main application and also the functionalities Also, we still need to work on Empatica and server application.