

MODUL PRAKTIKUM

MOBILE PROGRAMMING



**JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIKIBRAHIM
MALANG**

Jl. Gajayana No. 50 Malang (65144)
Telp : 0341-551354, Faks : 0341-572533

PENYUSUN

Supriyono, M. Kom

KATA PENGANTAR

Dengan menyebut asma Allah Yang Maha Pengasih lagi Maha Penyayang. Mari kita panjatkan puji syukur ke hadirat Allah SWT., yang telah melimpahkan rahmat, taufiq serta inayah-Nya sehingga penulis mampu menyelesaikan modul Praktikum Mobile Programming bagai salah satu buku panduan praktikum jurusan Teknik Informatika Universitas Islam Negeri Malang sesuai dengan waktu yang telah ditentukan.

Shalawat serta salam semoga tetap tercurahkan kepada Nabi Muhammad saw. yang membawa cahaya kebenaran, sehingga mengeluarkan umat manusia dari zaman kegelapan ke masa yang terang benerang agama Islam.

Penulis mengakui bahwa buku praktikum Praktikum Mobile Programming ini masih banyak kekurangan, kelemahan, dan masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun guna perbaikan ke depan.

Akhirnya semoga karya ini diterima di sisi Allah SWT. dan semoga mendapatkan balasan yang setimpal dari-Nya. Harapan penulis semoga buku panduan praktikum Praktikum Mobile Programming data ini dapat bermanfaat bagi penyusun khususnya, dan para mahasiswa pada umumnya.

Malang, 8 Juli 2018

Penyusun

DAFTAR ISI

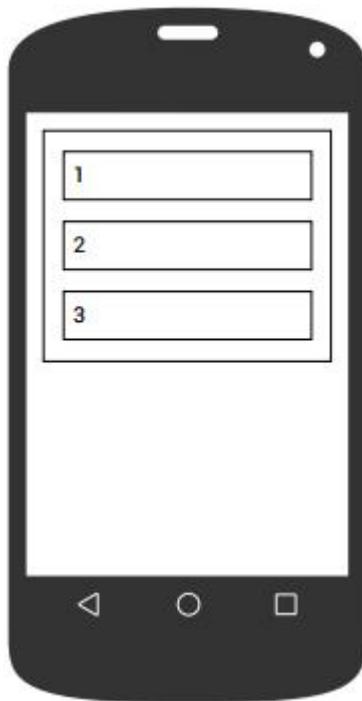
JUDUL	i
PENYUSUN	ii
KATA PENGANTAR	iii
DAFTAR ISI	iv
MODUL 1 LAYOUT	1
MODUL 2 WIDGET	14
MODUL 3 VIEW	19
MODUL 4 MAPS	50
MODUL 5 DATABASE.....	61
MODUL 6 API	77
MODUL 7 Database SQLite.....	84
MODUL 8 Database Real Time Using Firebase	98
MODUL 9 FIREBASE AUTHENTICATION.....	101
MODUL 10 AdMob	133
MODUL 11 PUBLISH APPS.....	198
MODUL 12 iOS.....	148

MODUL 1

LAYOUT

1. LINEAR LAYOUT

Vertical Linear Layout

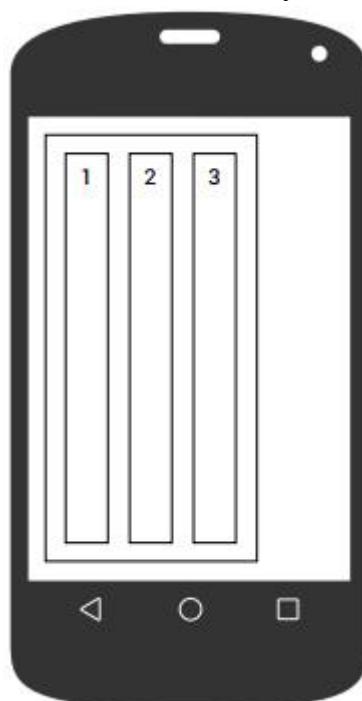


A vertical LinearLayout arranges its children in a column.

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="vertical">  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="1"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="2"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.1 Vertical Linear Layout

Horizontal Linear Layout

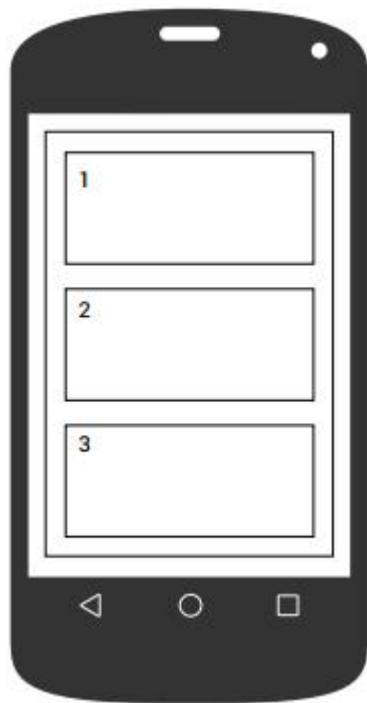


A horizontal LinearLayout arranges its children in a row.

```
<LinearLayout  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:orientation="horizontal">  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="match_parent"  
        android:text="1"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="match_parent"  
        android:text="2"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="match_parent"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.2 Horizontal Linear Layout

Vertical Linear Layout: Equal High

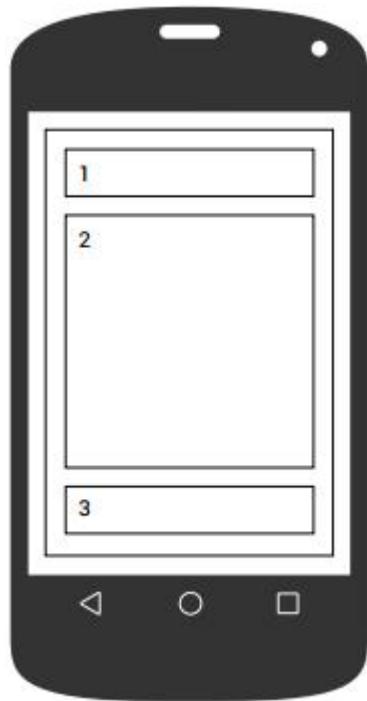


A vertical LinearLayout can give all of its children equal height.

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="1"  
        android:text="1"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="1"  
        android:text="2"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="1"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.3 Vertical Linear Layout: Equal High

Vertical Linear Layout: Leftover High

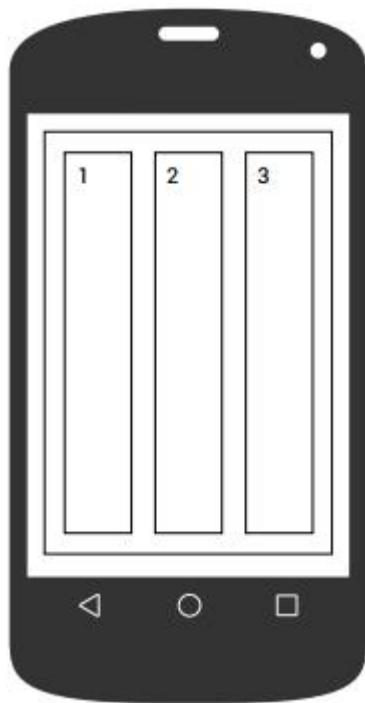


A vertical LinearLayout can give one of its children all the leftover height.

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:layout_weight="0"  
        android:text="1"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="0dp"  
        android:layout_weight="1"  
        android:text="2"/>  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:layout_weight="0"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.4 Vertical Linear Layout: Leftover High

Horizontal Linear Layout: Equal Width

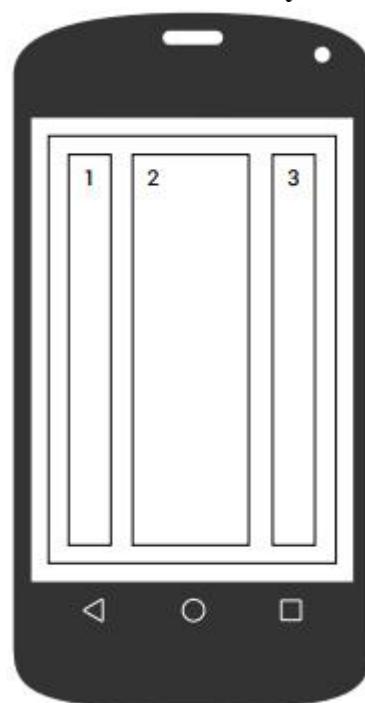


A horizontal LinearLayout can give all of its children equal width.

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="horizontal">  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="1"/>  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="2"/>  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.5 Horizontal Linear Layout: Equal Width

Horizontal Lineae Layout: Leftover Width



A horizontal LinearLayout can give one of its children all the leftover width.

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="horizontal">  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="match_parent"  
        android:layout_weight="0"  
        android:text="1"/>  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="2"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="match_parent"  
        android:layout_weight="0"  
        android:text="3"/>  
</LinearLayout>
```

Figure 1.6 Horizontal Lineae Layout: Leftover Width

RelativeLayout

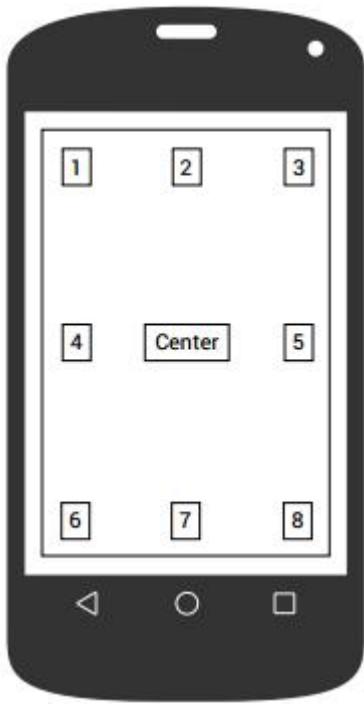


Figure 1.7Relative Layout model 1

A RelativeLayout can position a child relative to the RelativeLayout.

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:text="1"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:text="2"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:text="3"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_alignParentLeft="true"
        android:text="4"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:text="Center"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_alignParentRight="true"
        android:text="5"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:text="6"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:text="7"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:text="8"/>
</RelativeLayout>
```

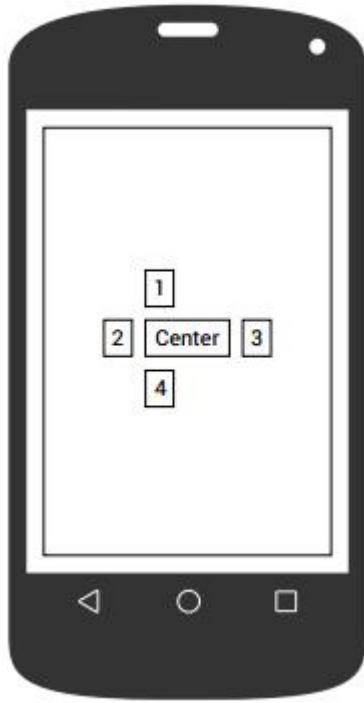


Figure 1.8Relative Layout model 2

```
<RelativeLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <TextView  
        android:id="@+id/center"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_centerInParent="true"  
        android:text="Center"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_toLeftOf="@+id/center"  
        android:layout_alignBottom="@+id/center"  
        android:text="2"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_toRightOf="@+id/center"  
        android:layout_alignBottom="@+id/center"  
        android:text="3"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_above="@+id/center"  
        android:layout_alignLeft="@+id/center"  
        android:text="1"/>  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_below="@+id/center"  
        android:layout_alignLeft="@+id/center"  
        android:text="4"/>  
</RelativeLayout>
```

See the UI Overview and Layouts Guide. The attributes for the children of each type of layout are listed in the LayoutParams classes.

- For LinearLayout, see the LinearLayout Guide, class LinearLayout and its source code, and class LinearLayout.LayoutParams and its source code.
- For RelativeLayout, see the RelativeLayout guide, class RelativeLayout and its source code, and class RelativeLayout.LayoutParams and its source code. The Views in the above images are outlined only for clarity. Additional code would be needed to draw the black borders and the margins between them.

2. INFLATER LAYOUT

LayoutInflater is one of the classes or libraries used to create or convert xml layout files, as new View objects, in the main layout or can be termed layout stack. The Inflater layout is needed when someone wants to build the user interface dynamically (programmatically). The use of LayoutInflater has become a common thing when it wants to develop more complex Applications.

Suppose that in the creation of an application that uses a ListView there are two layouts, the first is the layout in charge of the main layout (eg: activity_main.xml), and has a ListView container, then the layout containing the content (eg item_list.xml), which can load TextView, ImageView and others.

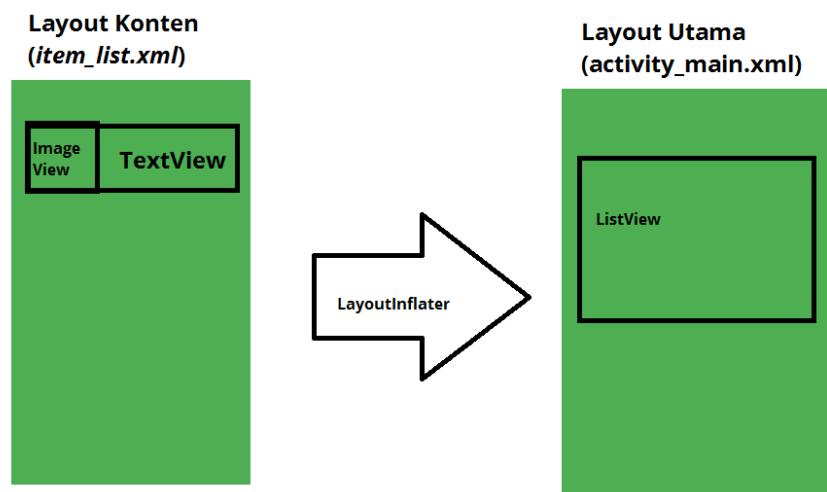


Figure 1.9 InflaterLayout Example

InflaterLayout java source code example:

```
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.RelativeLayout;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        getSupportActionBar().setTitle("Contoh LayoutInflater");
        getSupportActionBar().setSubtitle("Belajar di okedroid.com");

        RelativeLayout relativeLayout =
        (RelativeLayout)findViewById(R.id.activity_main);
        //1 initialisasi layout utama
        View view =
        getLayoutInflater().inflate(R.layout.konten,relativeLayout,false);
        // 2 mengconvert layout konten
        relativeLayout.addView(view);
        //3 method dari object relativelayout untuk menambahkan child view

    }
}
```

3. DIALOG

Alert Dialog is one of the most important and fundamental components of the Android App. Which serves to alert / alert to the user, and receive confirmation of the button action of the user of the Application. Here's how to create it.

1. First we create 1 xml layout file then copy the instruction line below:

main_alert.xml

```
“<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent" android:layout_height="match_parent"
    tools:context=".AlertMain" >
    <Button android:id="@+id/one_button_alert" android:layout_width="200dp"
        android:layout_height="wrap_content" android:layout_alignParentLeft="true"
        android:layout_marginTop="25dp" android:layout_marginLeft="20dp"
        android:text="One Button Alert" />
    <Button android:id="@+id/two_button_alert" android:layout_width="200dp"
        android:layout_height="wrap_content" android:layout_alignParentLeft="true"
        android:layout_below="@+id/one_button_alert" android:layout_marginLeft="20dp"
        android:text="Two Button Alert" />
    <Button android:id="@+id/three_button_alert" android:layout_width="200dp"
        android:layout_height="wrap_content" android:layout_alignParentLeft="true"
        android:layout_below="@+id/two_button_alert" android:layout_marginLeft="20dp"
        android:text="Three Button Alert" />
</RelativeLayout>
```

- Second we create the java file and copy the instruction line below:

```
package com.okedroid.myapplication;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
/** * Created by FATHUR on 12/16/2015. */public class AlertMain extends
Activity {
private Button button1;
private Button button2;
private Button button3;
@Override protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main_alert);
button1 = (Button) findViewById(R.id.one_button_alert);
button2 = (Button) findViewById(R.id.two_button_alert);
button3 = (Button) findViewById(R.id.three_button_alert);
button1.setOnClickListener(new View.OnClickListener() {
@Override public void onClick(View v) {
AlertDialog.Builder builder = new AlertDialog.Builder(
AlertMain.this);
builder.setTitle("Contoh Alert");
builder.setMessage("Alert dengan 1 Action Button ");
builder.setPositiveButton("OK",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,
int which) {
Log.e("info", "OK");
}
});
builder.show();
})
);
button2.setOnClickListener(new View.OnClickListener() {
@Override public void onClick(View v) {
AlertDialog.Builder builder = new AlertDialog.Builder(
AlertMain.this);
builder.setTitle("Contoh Alert");
builder.setMessage("Alert dengan 2 Action Button ");
builder.setNegativeButton("NO",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,
int which) {
Log.e("info", "NO");
}
});
builder.setPositiveButton("YES",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,
int which) {
Log.e("info", "YES");
}
});
builder.show();
})
);
```

```
button3.setOnClickListener(new View.OnClickListener() {
    @Override public void onClick(View v) {
        AlertDialog.Builder builder = new AlertDialog.Builder(
                AlertMain.this);
        builder.setTitle("Contoh Alert");
        builder.setMessage("Alert dengan 3 Action Button ");
        builder.setNegativeButton("NO",
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog,
                            int which) {
                        Log.e("info", "NO");
                    }
                });
        builder.setPositiveButton("YES",
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog,
                            int which) {
                        Log.e("info", "YES");
                    }
                });
        builder.setNeutralButton("BATAL",
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog,
                            int which) {
                        Log.e("info", "BATAL");
                    }
                });
        builder.show();
    }
});
}
}

@Override public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.menu_main, menu);
return true;
}
}
```

4. CONSTRAIN

ConstraintLayout is used to facilitate android developers, in designing user interfaces or interfaces on the Application, flexibly and dynamically, by drag and drop, without having to involve many Nested Multiple Layouts. So we no longer need to build a hierarchical layout, as in RelativeLayout, there is LinearLayout. Some View Content will be tied together and side by side. If you've ever built a UI in xcode, on an iOS device, maybe you're very familiar and familiar, and it does not take long to adapt and adjust to this layout.

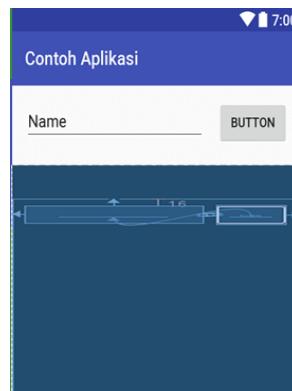


Figure 1.10designing user interfaces

In this section, explained how to build User Interface components such as EditText and Button, using the ConstraintLayout manager layout.

1. First open a project that already exists in Android Studio.
2. In build.gradle (Module App) add library from ConstraintLayout in dependencies section as shown below:

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 25
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "com.okedroid.contohaplikasi"
        minSdkVersion 16
        targetSdkVersion 25
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    testCompile 'junit:junit:4.12'
}
```

The screenshot shows the Android Studio project structure on the left with "build.gradle (Module: app)" selected. The code editor on the right displays the build.gradle file. A black rectangle highlights the "dependencies" block. Inside this block, the line "compile 'com.android.support.constraint:constraint-layout:1.0.2'" is also highlighted with a black rectangle, indicating it is the line being added or modified.

Figure 1.11add library from ConstraintLayout in dependencies

3. Then drag the ConstraintLayout on the left window of the Palette, then select Layouts, and then select ConstraintLayout. As shown below:

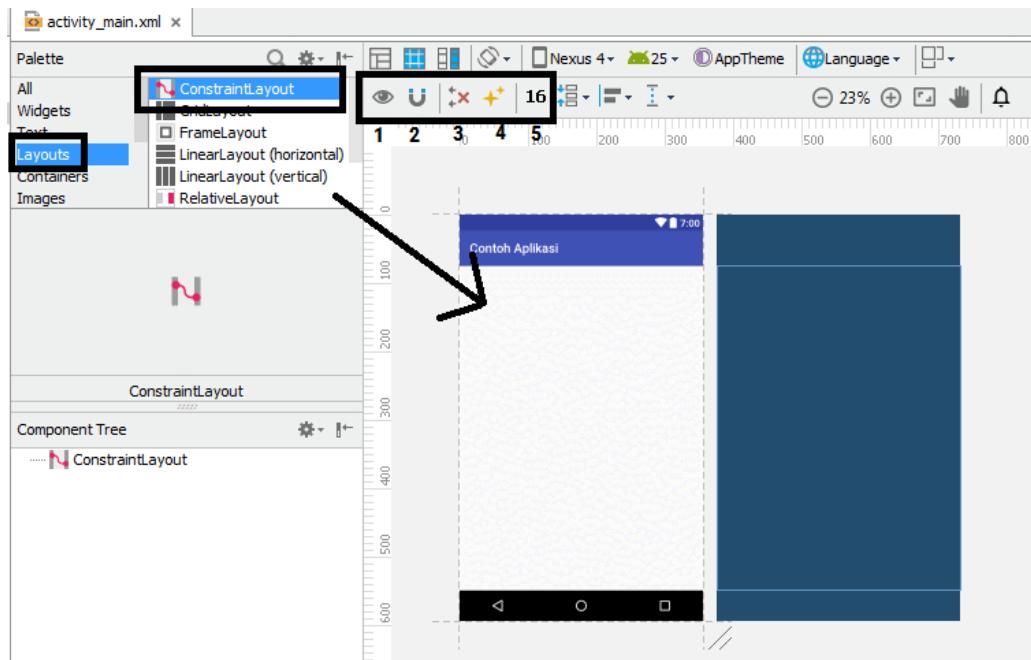


Figure 1.12Palette

There are special icons or toolbars while building User Interface with ConstraintLayout like:

- To turn it on or off, hide the constraint
- To enable or disable, Autoconnect constraint
- To delete the corresponding constraint
- To relate the constraint
- Standard size Margin in ConstraintLayout

Or you can also convert RelativeLayout to ConstraintLayout in the Component Tree window by right-clicking on the mouse, then select Convert RelativeLayout to ConstraintLayout.

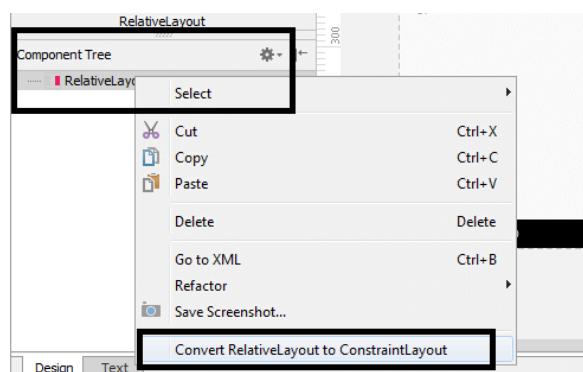


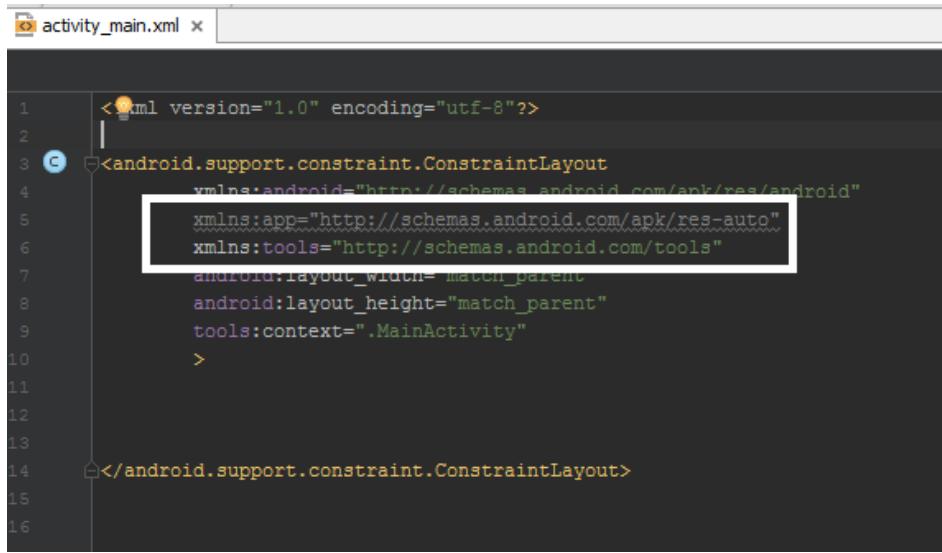
Figure 1.13convert RelativeLayout to ConstraintLayout

4. For the source code xmlnya, you can see in the picture below:

Here we can use a special namespace, such as

```
xmlns: app = "http://schemas.android.com/apk/res-auto" and
xmlns: tools = "http://schemas.android.com/tools"
```

To use the attributes available in the ConstraintLayout library.



```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    >
</android.support.constraint.ConstraintLayout>
```

Figure 1.14 ConstraintLayout library

- Now we will start trying to design the User Interface by drag and drop. We will try to design, using EditText and also Button on ConstraintLayout, and how to be consistent across all screen sizes on Android devices.

The source code will look like this:



```
<EditText
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:inputType="textPersonName"
    android:id="@+id/editText"
    android:ems="10"
    app:layout_constraintTop_toTopOf="parent"
    android:layout_marginLeft="16dp"
    android:layout_marginTop="16dp"
    app:layout_constraintRight_toLeftOf="@+id/button"
    app:layout_constraintLeft_toLeftOf="parent"
    android:id="@+id/editText2" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/button"
    android:layout_marginRight="16dp"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintBaseline_toBaselineOf="@+id/editText"
    app:layout_constraintLeft_toRightOf="@+id/editText2"
    android:layout_marginLeft="16dp" />
</android.support.constraint.ConstraintLayout>
```

Figure 1.15 ConstraintLayout

To understand this, consider the example illustrations:

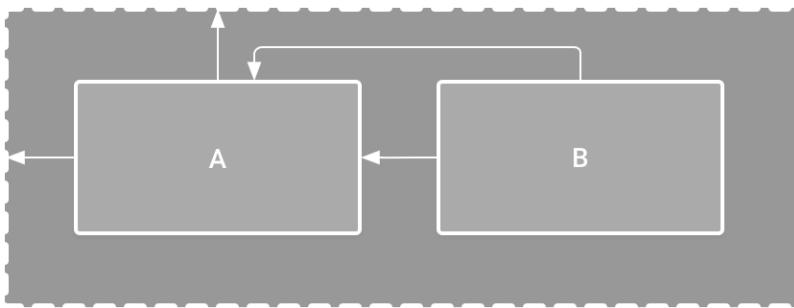


Figure 1.16 illustrations

Information :

View A with Parent Layout has size spacing or margin 16 dp and above

View A with Parent Layout has a spacing or margin size of 16 dp to the left

View B with View A has a spacing or margin of 16 dp to the left

View B is aligned at the top with View A

MODUL 2

WIDGET

1. Widget Concept

- a. Android widget has been little implemented in the activity section.
- b. Basically the widget package is a visualization of the user interface elements used on Android application screen where the programmer can design itself according to the needs of the user / client.
- c. Inside android, widgets are displayed with the View concept, where android apps use widgets as an XML layout in general.

2. The types of android widgets are as follows:

a. TextView

To display text on the screen and optionally can be edited, but by default TextView can not be edited, to be able to edit must use subclass that serves to display the contents.

Example property of TextView is android: textSize to set size, android: textStyle to set whether bold or italic font, android: textColor.



Figure 2.1 TextView

b. EditText

EditText is a TextView customization which becomes a TextView that can configure itself so that it can be edited. Please try one by one and note the resulting XML.



Figure 2.2 EditText

c. Radio Button/ Radio Group

RadioButton is used in conjunction with RadioGroup. Inside a RadioGroup there are several RadioButton, and in one RadioGroup user can only do one check / RadioButton selection.



Figure 2.3RadioButton

d. Image View/ Image Button

ImageView is a widget that displays images like icons.

1. ImageView can load images from various sources (resource or content Providers),
ImageButton is a widget that displays the contains button Images (not text) that
can be pushed or clicked by the user.
2. By default, ImageButton is almost the same as the regular button



Figure 2.4Image View

e. Spinner/ ComboBox

This widget is almost similar to ListView. The difference is that the ListView comes from a single combo or known as a spinner.



Figure 2.5 Spinner

f. Button

Button is a derivative of TextView so applicable in textView also applies to the button.
The most important addition is onClick.

```
<Button  
    android:id="@+id/button"  
    android:layout_width="95dp"  
    android:layout_height="38dp"  
    android:text="KLIK"  
    tools:layout_editor_absoluteX="66dp"  
    tools:layout_editor_absoluteY="41dp" />
```

Figure 2.6 Button

The result (note the use of fill_parent for layout_width attribute):



g. CheckBox

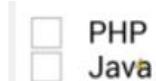
User can select more than one option with checkbox. On the palette, checkbox is in the section from widgets.

CheckBox RadioButton

Try adding two checkboxes then set the id and text attributes through the property window:



So the result,



An example of creating a Login Form

Create Layout for login page, res / layout -> Android XML file. Name login.xml, then type the code.

```
<?xml version="1.0" encoding="utf-8"?>
<!--Login Form-->

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="10dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.proje.vimalki.tutorial.MainActivity"
    android:weightSum="1">

    <!--Email Label-->
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Email"
        android:textSize="12dp"
        android:textColor="#4c4c4c"
        android:textStyle="bold"
        android:id="@+id/textView1"
        android:paddingTop="10dp"
        />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:layout_marginBottom="20dp"
        android:singleLine="true"
        android:id="@+id/editText1"
        android:layout_weight="0.06"
        android:background="#ffffffff"
        android:foregroundTint="#bf2c2c" />

    <!--Password Label-->
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Password"
        android:textSize="12dp"
        android:id="@+id/textView2" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editText2"
        android:layout_marginTop="5dp"
        android:singleLine="true"
        android:password="true"
        android:layout_weight="0.06"
        android:background="#ffffffff"
        android:foregroundTint="#bf2c2c" />
```

```

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:inputType="textPassword"
    android:ems="10"
    android:id="@+id/editText5" />

<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Login"
    android:textSize="15dp"
    android:gravity="center"
    android:layout_marginTop="15dp"
    android:id="@+id/button"
    android:layout_weight="0.02"
    android:onClick="bSapaClick"
    android:background="#b4b4b4" />

<TextView
    android:id="@+id/link_to_register"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="40dp"
    android:layout_marginTop="40dp"
    android:gravity="center"
    android:text="Register, for new user !"
    android:textColor="#0b84aa"
    android:textSize="20dp" />

```

And the results on the display of the above code is

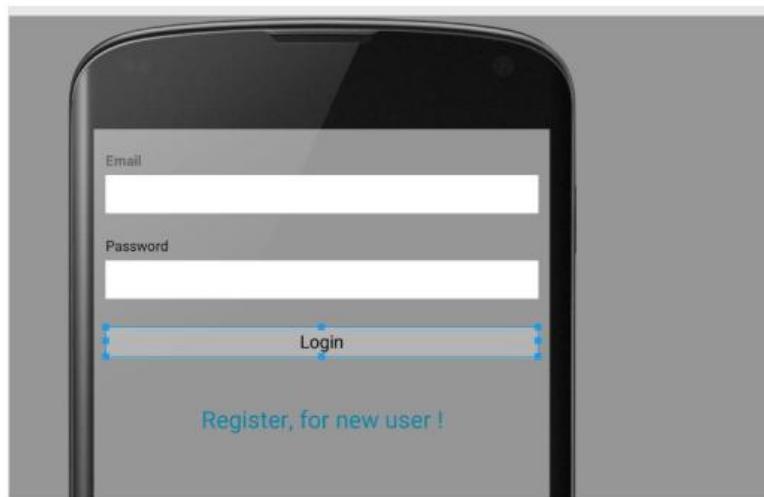


Figure 2.6 Result Display

MODUL 3

VIEW

1. SPLASH SCREEN

Splash Screen is an activity that is set as an initial layer or an opening. Splash Screen only appears in a short time and then move on to the next activity. Splash Screen is used as a means of checking whether the application is ready or not to load the data. Examples of Splash screen quite a lot, such as screen loading, screen welcome, and others. Along with the antenna intent can also be used to send data from activity to other activity.

Create a Splash Screen

1. Create an activity with an empty activity with the name of MainActivity that will be Used as a Splash screen, then edit its xml layout.

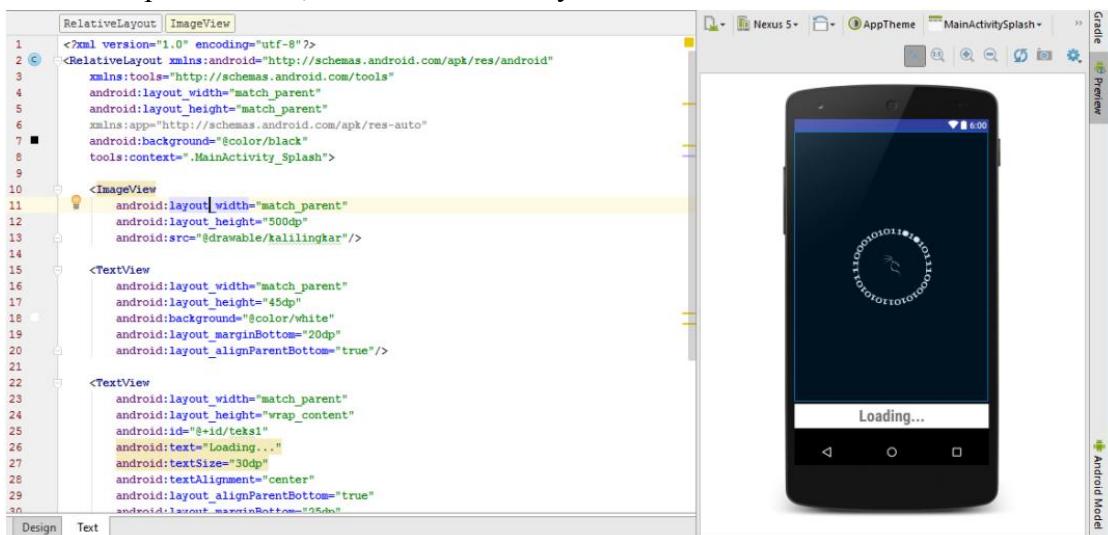


Figure 3.1 Splash Screen

2. In the java code activity_main to be used as Splash Screen, type source code for splash screen and intent, then run observe what happened.

```
public class MainActivity_Splash extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        new Handler().postDelayed(new Runnable() {  
            @Override  
            public void run() {  
                Intent intent = new Intent(MainActivity_Splash.this, LoginScreen.class);  
                startActivity(intent);  
                finish();  
            }  
        }, 2000);  
    }  
}
```

Source code
splash screen

Figure 3.2 Source Code Splash Screen

From the source code above the "(MainActivity_Splash.this, LoginScreen.class)" section it shows that the intent starts from this Activity which is MainActivity_Splash and will end in Activity LoginScreen.

3. After layout and code display for Splash Screen is complete, now create a new Activity with LoginScreen name.

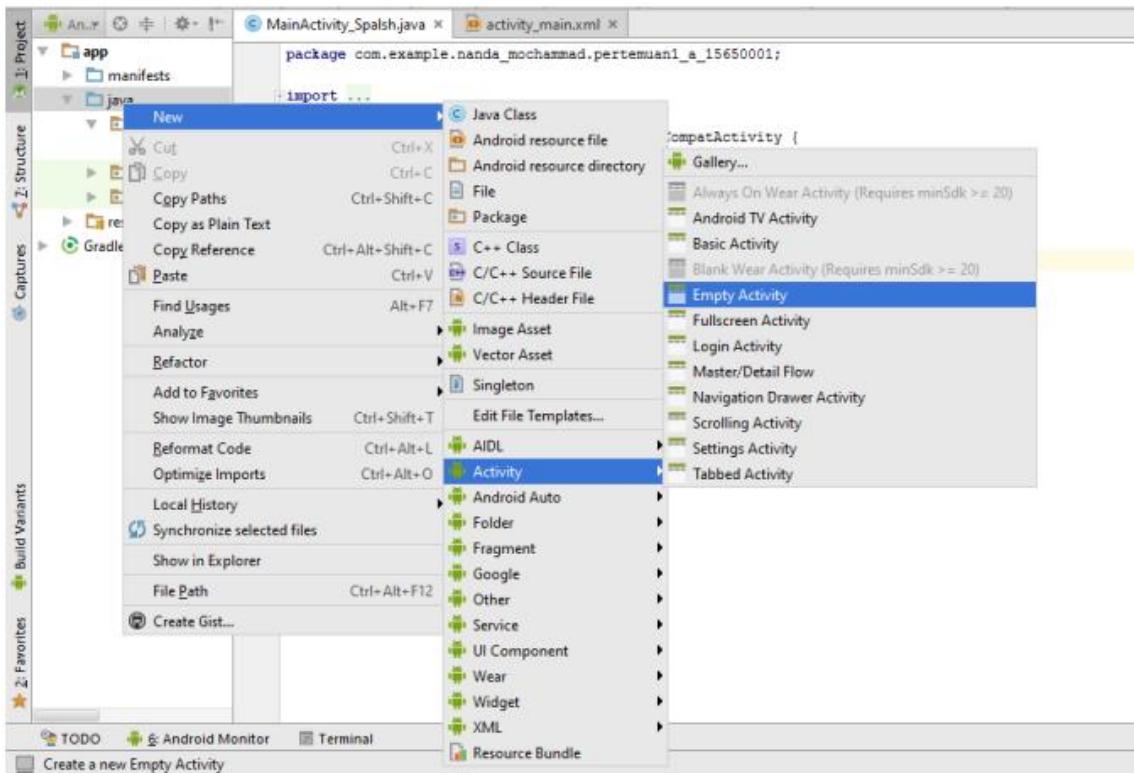


Figure 3.3 Create new empty activity

4. Then edit its xml Layout on Login Screen

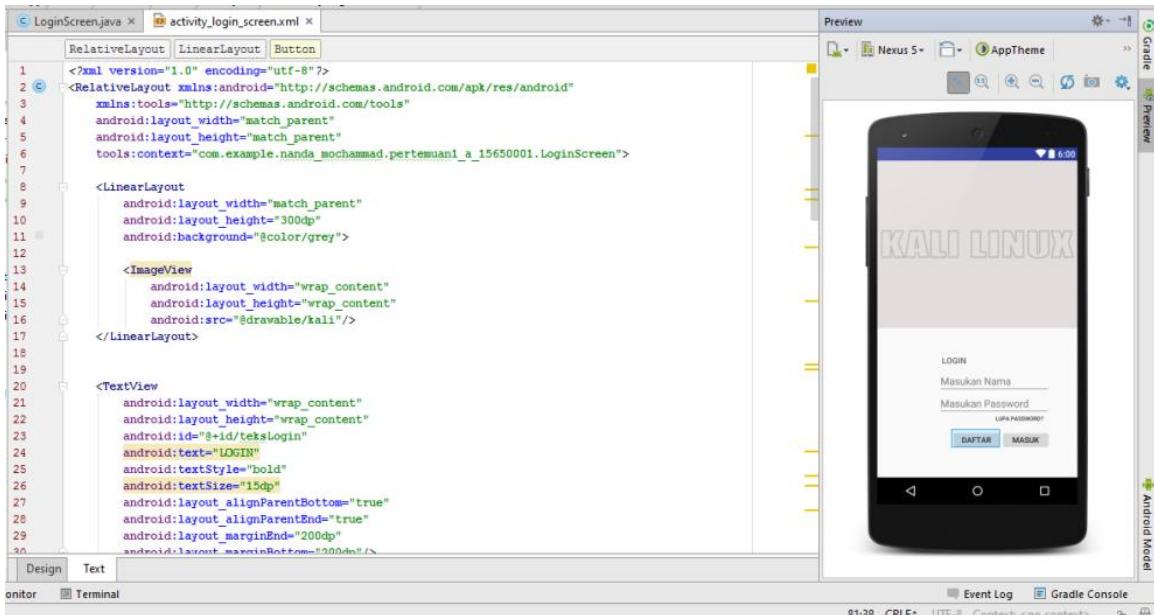


Figure 3.4 xml Layout on Login Screen

```
<EditText  
    android:layout_width="200dp"  
    android:layout_height="wrap_content"  
    android:id="@+id/editLogin"  
    android:hint="Masukan Nama"  
    android:textAlignment="textStart"  
    android:layout_alignParentBottom="true"  
    android:layout_alignParentEnd="true"  
    android:layout_marginEnd="50dp"  
    android:layout_marginBottom="150dp"/>  
  
<EditText  
    android:layout_width="200dp"  
    android:layout_height="wrap_content"  
    android:id="@+id/editLogin1"  
    android:hint="Masukan Password"  
    android:textAlignment="textStart"  
    android:layout_alignParentBottom="true"  
    android:layout_alignParentEnd="true"  
    android:layout_marginEnd="50dp"  
    android:layout_marginBottom="110dp"/>  
  
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="40dp"  
    android:id="@+id/buttonForgot"  
    android:background="#ffcc00"/>
```

Figure 3.5 source code xml on login screen

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="40dp"  
    android:id="@+id/buttonForgot"  
    android:text="Lupa password?"  
    android:textSize="10dp"  
    android:keepScreenOn="true"  
    style="?borderlessButtonStyle"  
    android:layout_alignParentBottom="true"  
    android:layout_marginBottom="85dp"  
    android:layout_alignParentEnd="true"  
    android:layout_marginEnd="50dp"/>  
  
<LinearLayout  
    android:layout_width="wrap_content"  
    android:layout_height="35dp"  
    android:weightSum="2"  
    android:id="@+id/linearLogin"  
    android:layout_alignParentEnd="true"  
    android:layout_alignParentBottom="true"  
    android:layout_marginBottom="50dp"  
    android:layout_marginEnd="50dp">  
    <Button  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_weight="1"  
        android:id="@+id/buttonLogin"  
        android:text="Daftar"/>
```

Figure 3.6 source code xml on login screen

```

        android:layout_alignParentEnd="true"
        android:layout_marginEnd="50dp"/>

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="35dp"
        android:weightSum="2"
        android:id="@+id/linearLogin"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginBottom="50dp"
        android:layout_marginEnd="50dp">
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:id="@+id/buttonLogin"
            android:text="Daftar"/>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:id="@+id/buttonLogin2"
            android:text="masuk"/>
    </LinearLayout>
</RelativeLayout>

```

Figure 3.7 source code xml on login screen

5. After Edit Layout to Login Screen, now type the java code for LoginScreen To send data.

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class LoginScreen extends AppCompatActivity {

    //Deklarasi Object yang sudah dibuat di Layout agar dapat diberi aksi.
    EditText nama, password;
    Button daftar, masuk, lupa;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login_screen);

        //Ini juga deklarasi nya
        nama = (EditText) findViewById(R.id.editLogin);
        password = (EditText) findViewById(R.id.editLogin1);
        daftar = (Button) findViewById(R.id.buttonLogin);
        masuk = (Button) findViewById(R.id.buttonLogin2);
        lupa = (Button) findViewById(R.id.buttonForgot);

        //Kode untuk intent. Dan mengirimkan data ke activity selanjutnya
        masuk.setOnClickListener(v) -> {
            Intent intent = new Intent(LoginScreen.this, firstScreen.class);
            intent.putExtra("Nama ", nama.getText().toString());
            startActivity(intent);
        };
    }
}

```

6. Create one Activity that will be the destination of the intent. Name the Activity FirstScreen. And type java code firstScreen in accordance with the code below. Then Run from the beginning of Activity and observe what happened.

```
import ...  
  
public class firstScreen extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_first_screen);  
  
        //Berikut Kode menerima data dari activity sebelumnya  
        String namaUser = getIntent().getStringExtra("Nama");  
  
        //Cibawat ini kode untuk menampilkan TOAST  
        Toast.makeText(getApplicationContext(), "Selamat Datang" + namaUser, Toast.LENGTH_LONG).show();  
    }  
}
```

To receive data sent from Activity before it can be used .getStringExtra ("DataData"). In this example Toast used to display data that has been obtained from the previous Activity.

2. LISTVIEW

- a. Original Listview = Creating a Listview using a template already provided by Android Studio
- b. Custom List View = Create List View with customization from the developer itself

ORIGINAL LIST VIEW

Adapter is a bridge between AdapterView (sample listview) with data. This adapter provides access to data items and is also responsible for creating a View of each item in the data set.

1) Prepare some XML and Class needed.

Here we need 2 Class in the java folder that is "Detail Activity" and "Main Activity". Then we enter in the res - layout folder and create 3 xml ie "activity_detail.xml", "Activity_main.xml" and "item.xml".

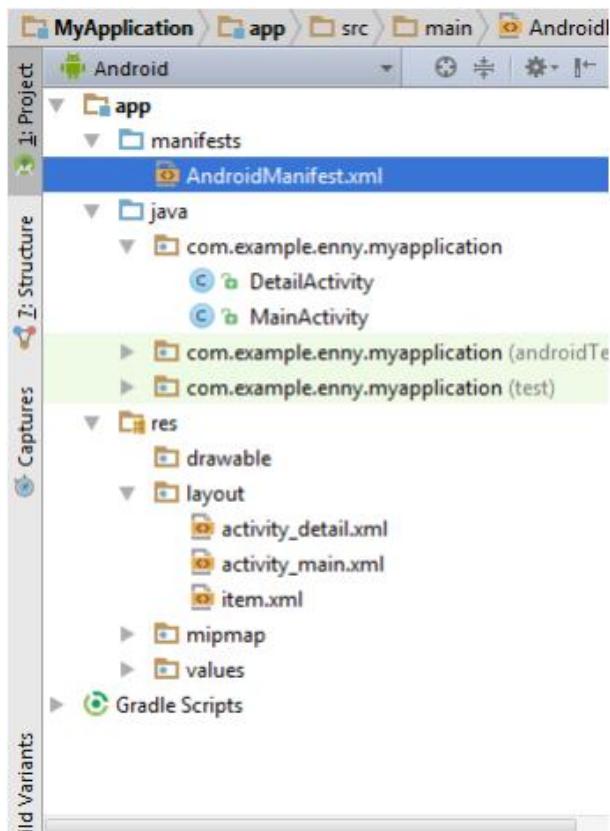


Figure 3.8 Android Manifest

2) Creating ListView

After creating some required attributes, then click the res - layout - activity_main.xml folder, write the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="com.example.enny.myapplication.MainActivity">

    <ListView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/listView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true">

    </ListView>
</RelativeLayout>
```



Figure 3.9 Source code android manifest

- 3) Create an Array then we create MainActivity.java class to create data of fruit names in array and enter the following code:

```
public class MainActivity extends AppCompatActivity {
    Context context;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        context=MainActivity.this;
        final String ArrayBuah[]={ "Apel", "Manggis", "Melon", "Nanas", "Jeruk" };
```

Figure 3.9 source code array on MainActivity.java

- 4) TextView in detail.

Then we go to Activity_Detail.xml and add textView as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="com.example.enny.myapplication.DetailActivity">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/detail"
        android:text="Berhasil"
        android:textStyle="bold"
        android:textSize="35dp" />
</RelativeLayout>
```



Figure 3.10 source code Activity_Detail.xml

5) Creating data.

In DetailActivity.java we fill in the following code:

```
public class DetailActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);

        TextView textViewDetail = (TextView) findViewById(R.id.detail);
        textViewDetail.setText(getIntent().getStringExtra("data"));
    }
}
```

6) Creating Adapters.

In MainActivity.java add the following adapter code:

```
ListView listView = (ListView) findViewById(R.id.listView);
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>
    (context, R.layout.activity_main, ArrayBuah);
listView.setAdapter(arrayAdapter);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Toast.makeText(context, "Anda Memilih: " + ArrayBuah[position], Toast.LENGTH_SHORT).show();
        Intent intent = new Intent(context, DetailActivity.class );
        intent.putExtra("data", ArrayBuah[position]);
        startActivity(intent);
    }
});
BaseAdapter baseAdapter = new BaseAdapter() {
    @Override
    public int getCount() {
        return ArrayBuah.length;
    }
    @Override
    public Object getItem(int position) {
        return null;
    }

    @Override
    public long getItemId(int position) {
        return 0;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View itemCustom = getLayoutInflater().inflate(R.layout.item, null);
        TextView textViewContent = (TextView)itemCustom.findViewById(R.id.textData);
        textViewContent.setText(ArrayBuah[position]);
        return itemCustom;
    }
};
listView.setAdapter(baseAdapter);
}
}
```

Then run. And will display as below:

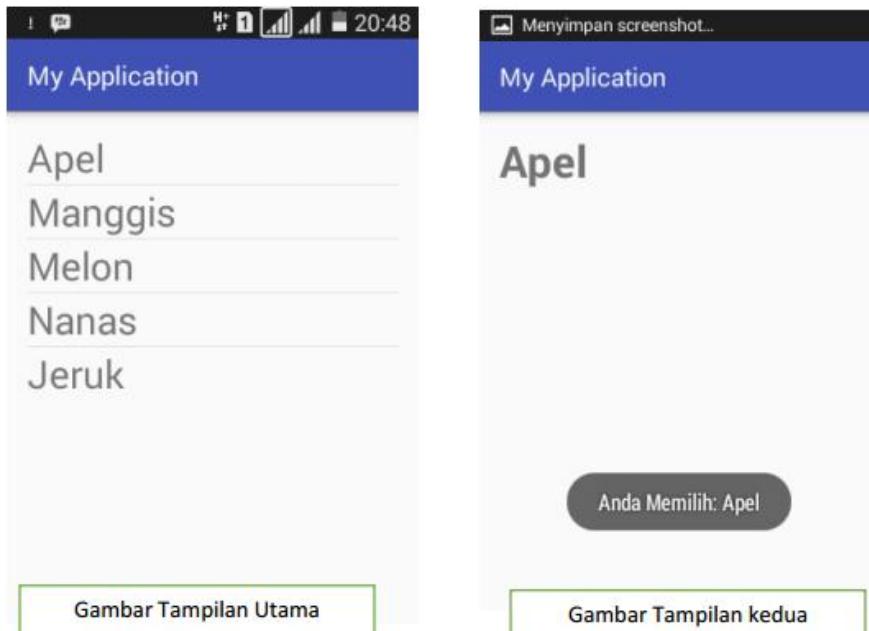


Figure 3.11 Result Display

3. CARDVIEW

CardView function as a wrapper or frame layout that will wrap the layout inside with a card-like design. If the reader sees a list of apps on the Google Play Store, the view used is CardView. The hallmark of CardView design is the presence of rounded corners and the presence of shadows for elevation effects.

Please create a new project on Android Studio with CardViewTest name, or name it as you wish. Open Android Studio, click **File - New Project** fill in Application Name with **CardViewTest** name. In the Activity window please select Empty Activity.

After that, we will add dependencies library from CardView on the project being created. Please open the **build.gradle(Module: app)** file located on the already created project. Then add the code below.

```
compile 'com.android.support:appcompat-v7:21.0.3'  
compile 'com.android.support:cardview-v7:21.0.3'  
compile 'com.android.support:support-v4:21.0.3'
```

After adding the library, please open the **activity_main.xml** file located in the **res / layout** / created project directory. Then add the code as below.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="oc.startup.ra.cardview2.MainActivity">

    <android.support.v7.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/card_view"
        android:layout_width="match_parent"
        android:layout_height="380dp"
        android:layout_margin="8dp">

        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <ImageView
                android:id="@+id/thumbnail"
                android:layout_width="match_parent"
                android:layout_height="250dp"
                android:layout_alignParentTop="true"
                android:scaleType="centerCrop"
                android:src="@drawable/wallpaper" />

            <TextView
                android:id="@+id/title"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_below="@+id/thumbnail"
                android:maxLines="3"
                android:padding="8dp"
                android:text="@string/title"
                android:textColor="#222"
                android:textStyle="bold"
                android:textSize="22dp" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_below="@+id/title"
                android:maxLines="3"
                android:padding="8dp"
                android:text="@string/description"
                android:textColor="#666"
                android:textSize="14dp" />

        </RelativeLayout>
    </android.support.v7.widget.CardView>

```

By adding the source code above, we can already see the results of the CardView display created. However, to complete the display to display good results please change the source code in ImageView and TextView. In ImageView, prepare a picture or photo to be used as wallpaper wrapped by CardView. To add a picture please go to the created project directory and save it in the **drawable** folder. In this example it is located in **CardViewTest \ app \ src \ main \ res \ drawable**. Give the image a name as desired, then change the source code below according to the name of the selected image.

[android:src="@drawable/wallpaper"](#)

To complete the TextView, please open the **res / values / string.xml** directory located on the created project. Then add the code like below.

```

<string name="title">Riswan Abidin</string>
<string name="description">Seorang yang sangat tertarik dengan perkembangan teknologi dan antusias terhadap komunitas dan startup</string>

```

Name = "title" and **name = "description"** are the same on the TextView source code found on the **activity_main.xml** as below.

```
<TextView  
    android:id="@+id/title"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/thumbnail"  
    android:maxLines="3"  
    android:padding="8dp"  
    android:text="@string/title"  
    android:textColor="#222"  
    android:textStyle="bold"  
    android:textSize="22dp" />  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/title"  
    android:maxLines="3"  
    android:padding="8dp"  
    android:text="@string/description"  
    android:textColor="#666"  
    android:textSize="14dp" />
```

Below is an app view when running on an Android emulator.

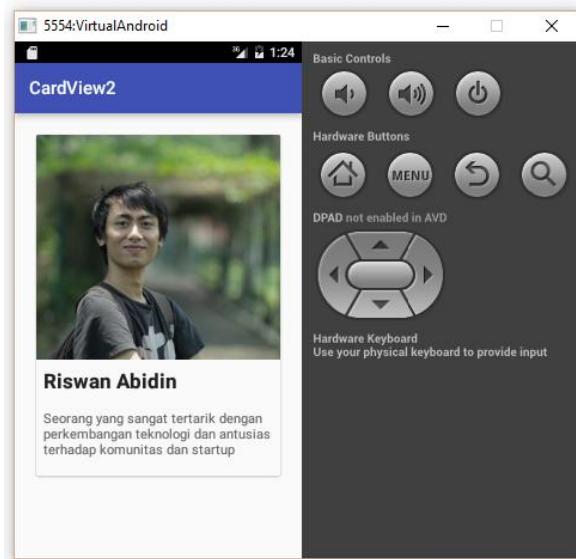


Figure 3.12 Result Display CardView

4. RECYCLERVIEW

Recyclerview is a development of listview android that has better performance and many other advantages. Recyclerview appeared since the emergence of android lollipop (Android 5.0). Recyclerview by default has no divider unlike listview.

1) Create a project

2) Dependencies

After Project is created, now open **Gradle Script > build.gradle (Module: app)** > Add Recyclerview dependency **com.android.support:recyclerview-v7:24.0.0** > then Rebuild Project.

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    testCompile 'junit:junit:4.12'  
    compile 'com.android.support:appcompat-v7:24.0.0'  
    compile 'com.android.support:recyclerview-v7:24.0.0'  
}
```

3) Create a view

Go to activity_main.xml in Res> layout> activity_main.xml to

```
<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <android.support.v7.widget.RecyclerView  
        android:id="@+id/recyclerview"  
        android:layout_width="match_parent"  
        android:layout_height="match_parent" />  
  
</RelativeLayout>
```

Create custom row_layout.xml in Res> Layout> row_layout.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="#ffffffff"
    android:orientation="horizontal"
    android:paddingLeft="10sp"
    android:paddingRight="10sp"
    android:paddingTop="5sp">

    <ImageView
        android:id="@+id/image_view"
        android:layout_width="45sp"
        android:layout_height="45sp"
        android:layout_gravity="center"
        android:src="@android:drawable/ic_lock_idle_low_battery" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="2sp"
        android:orientation="vertical"
        android:paddingLeft="10sp">

        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingBottom="10sp">

            <LinearLayout
                android:id="@+id/linearLayout"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:orientation="vertical">

                <TextView
                    android:id="@+id/txtDate"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:singleLine="true"
                    android:text="Nama"
                    android:textStyle="bold" />

                <TextView
                    android:id="@+id/txtDay"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="status" />

            </LinearLayout>

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentEnd="true"
                android:layout_alignParentRight="true"
                android:layout_alignTop="@+id/linearLayout"
                android:text="Mobile" />
        </RelativeLayout>

        <View
            android:layout_width="match_parent"
            android:layout_height="0.5dp"
            android:background="@android:color/darker_gray"
            android:paddingLeft="20sp" />
    </LinearLayout>
</LinearLayout>
```

4) Create a Setter Getter Class

Create a Class Model or what we call the setter getter, eg name Contact.java.

```
package com.imamfarisi.whatsupcontact;

/*
 * Created by imam-pc on 04/07/2016.
 */
public class Contact {
    private String nama, status, tipePhone;
    private Integer photoPic;

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public String getStatus() {
        return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }

    public String getTipePhone() {
        return tipePhone;
    }

    public void setTipePhone(String tipePhone) {
        this.tipePhone = tipePhone;
    }

    public Integer getPhotoPic() {
        return photoPic;
    }

    public void setPhotoPic(Integer photoPic) {
        this.photoPic = photoPic;
    }
}
```

5) Create a RecycleView Adapter

Then we make a Special Adapter for Recyclerview, for example we named RecyclerviewAdapter.java.

```
package com.imamfarisi.whatsappcontact;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.support.v4.graphics.drawable.RoundedBitmapDrawable;
import android.support.v4.graphics.drawable.RoundedBitmapDrawableFactory;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.List;

/*
 * Created by imam-pc on 04/07/2016.
 */
public class RecyclerviewAdapter extends RecyclerView.Adapter<RecyclerviewAdapter.MyViewHolder> {

    private List<Contact> contactList;
    private Context ctx;

    public class MyViewHolder extends RecyclerView.ViewHolder {
        public TextView txtNama, txtStatus, txtTipePhone;
        public ImageView imgView;

        public MyViewHolder(View view) {
            super(view);
            txtNama = (TextView) view.findViewById(R.id.txtNama);
            txtStatus = (TextView) view.findViewById(R.id.txtStatus);
            txtTipePhone = (TextView) view.findViewById(R.id.txtPhoneType);
            imgView = (ImageView) view.findViewById(R.id.imgView);
        }
    }

    public RecyclerviewAdapter(List<Contact> contactList, Context ctx) {
        this.ctx = ctx;
        this.contactList = contactList;
    }
}
```

```

@Override
public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View itemView = LayoutInflater.from(parent.getContext())
        .inflate(R.layout.row_layout, parent, false);

    return new MyViewHolder(itemView);
}

@Override
public void onBindViewHolder(MyViewHolder holder, int position) {
    Contact contact = contactList.get(position);
    holder.txtName.setText(contact.getName());
    holder.txtStatus.setText(contact.getStatus());
    holder.txtTypePhone.setText(contact.getTypePhone());

    Bitmap srcBitmap = BitmapFactory.decodeResource(ctx.getResources(), contact.getPhotoPic());
    holder.imgView.setImageBitmap(srcBitmap);
    circularImage(holder.imgView, srcBitmap);
}

@Override
public int getItemCount() {
    return contactList.size();
}

private void circularImage(ImageView imageView, Bitmap srcBitmap) {
    Paint paint = new Paint();

    int srcBitmapWidth = srcBitmap.getWidth();
    int srcBitmapHeight = srcBitmap.getHeight();

    int borderWidth = 30;

    // destination bitmap width
    int dstBitmapWidth = Math.min(srcBitmapWidth, srcBitmapHeight) + borderWidth * 2;
    float radius = Math.min(srcBitmapWidth, srcBitmapHeight)/2;

    // Initializing a new bitmap to draw source bitmap, border and shadow
    Bitmap dstBitmap = Bitmap.createBitmap(dstBitmapWidth, dstBitmapWidth, Bitmap.Config.ARGB_8888);

    // Initialize a new canvas
    Canvas canvas = new Canvas(dstBitmap);

    // Draw a solid color to canvas
    canvas.drawColor(Color.WHITE);

    // Draw the source bitmap to destination bitmap by keeping border and shadow spaces
    canvas.drawBitmap(srcBitmap, (dstBitmapWidth - srcBitmapWidth) / 2, (dstBitmapWidth - srcBitmapHeight) / 2, null);

    // Use Paint to draw border
    paint.setStyle(Paint.Style.STROKE);
    paint.setStrokeWidth(borderWidth * 2);
    paint.setColor(Color.WHITE);

    // Draw the border in destination bitmap
    canvas.drawCircle(canvas.getWidth() / 2, canvas.getHeight() / 2, canvas.getWidth() / 2, paint);

    // Use Paint to draw shadow
    paint.setColor(Color.LTGRAY);

    // Draw the shadow on circular bitmap
    canvas.drawCircle(canvas.getWidth() / 2, canvas.getHeight() / 2, canvas.getWidth() / 2, paint);

    RoundedBitmapDrawable roundedBitmapDrawable = RoundedBitmapDrawableFactory.create(ctx.getResources(), dstBitmap);

    // Make the ImageView image to a circular image
    roundedBitmapDrawable.setCircular(true);
    roundedBitmapDrawable.setAntiAlias(true);
    // Set the ImageView image as drawable object
    imageView.setImageDrawable(roundedBitmapDrawable);
}
}

```

6) Create MainActivity.java

Then Last step is to register recyclerview to MainActivity.java.

```
package com.imamfarisi.whatsappcontact;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private List<Contact> contactList = new ArrayList<>();
    private RecyclerView recyclerView;
    private RecyclerviewAdapter mAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        recyclerView = (RecyclerView) findViewById(R.id.recyclerview);
        setData();
    }

    private void setData() {
        Contact dataSpiderman = new Contact("Spiderman", "Homecoming", "MOBILE", R.drawable.spiderman);
        Contact dataThor = new Contact("Thor", "Ragnarok", "MOBILE", R.drawable.thor);
        Contact dataIronMan = new Contact("Ironman", "Bukan Besi Biasa", "MOBILE", R.drawable.ironman);
        Contact dataCaptain = new Contact("Captain America", "Civil War", "MOBILE", R.drawable.captain);
        Contact dataFlash = new Contact("The Flash", "Speed is Everything", "MOBILE", R.drawable.flash);

        contactList.add(dataSpiderman);
        contactList.add(dataThor);
        contactList.add(dataIronMan);
        contactList.add(dataCaptain);
        contactList.add(dataFlash);

        mAdapter = new RecyclerviewAdapter(contactList, this);
        RecyclerView.LayoutManager mLayoutManager = new LinearLayoutManager(getApplicationContext());
        recyclerView.setLayoutManager(mLayoutManager);
        recyclerView.setAdapter(mAdapter);
    }
}
```

Once executed then the result will be like this

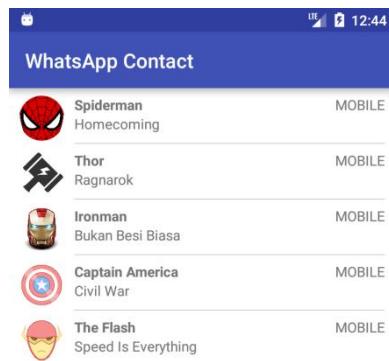


Figure 3.12 Result Display RecycleView

5. NAVIGATION DRAWER

Navigation Drawer View is one of the navigation menu components or so-called sliding menu that serves to wrap and navigate a content inside Activity or Fragment, from Applications. Navigation drawer is now supported by the Material Design Library. Which belong to the Appcompat library (v21). Navigation Drawer View is often used in popular applications such as BBM. To use the user (user) just need to move your thumb or finger right then Navigation Drawer View will appear.

In this android learning tutorial, we will try to create and implement Navigation Drawer View with Android Material Design support.

1) Setup Gradle

Before getting started to by default make sure on the build gradle especially in the dependencies section you settings like this:

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. dependencies {
2.     compile fileTree(dir: 'libs', include: ['*.jar'])
3.     testCompile 'junit:junit:4.12'
4.     compile 'com.android.support:appcompat-v7:23.3.0'
5.     compile 'com.android.support:design:23.3.0'
6. }
```

2) Setup resource values

In the folder values (app / res / values) section you apply the instruction row (codingan) below on each file:

styles.xml

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <resources>
2.
3.     <!-- Base application theme. -->
4.     <style name="AppTheme" parent="Theme.AppCompat.NoActionBar">
5.         <item name="colorPrimary">@color/colorPrimary</item>
6.         <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
7.         <item name="colorAccent">@color/colorAccent</item>
8.         <!-- Customize your theme here. -->
9.     </style>
10.
11. </resources>
```

Karna kita nanti akan membuat toolbar, maka kita set atau pilih thema "Theme.AppCompat.NoActionBar"

strings.xml

DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

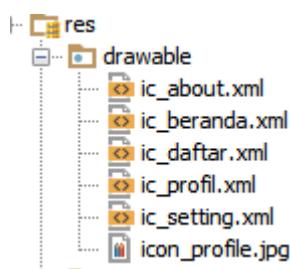
```
1. <resources>
2.   <string name="app_name">"Contoh Navigation Drawer "</string>
3.   <string name="openDrawer">Drawer dibuka</string>
4.   <string name="closeDrawer">Drawer ditutup</string>
5.     <string name="navigation_view_item_1">Beranda</string>
6.     <string name="navigation_view_item_2">Profil</string>
7.     <string name="navigation_view_item_3">Daftar</string>
8.     <string name="navigasi_kategori_2">Sub Item Navigasi</string>
9.     <string name="navigation_view_item_4">Settings</string>
10.    <string name="navigation_view_item_5">About</string>
11.
12.
13. </resources>
```

This file is used to store resource or text data from menu navigation items.

3) Prepare the icon / drawing in drawable

Prepare a .png or vector .xml image and save it in the folder (app / res / drawable).

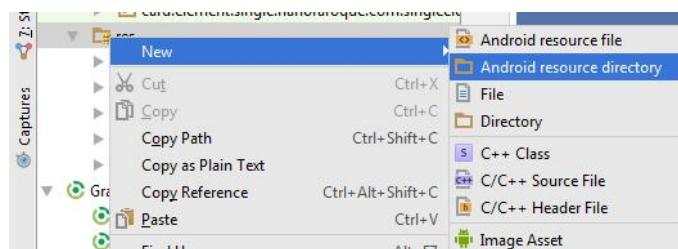
Examples like this:



4) Creating a menu folder

By default (default) if we select Empty Activity, we do not include menu folder in resource, so we will make it manually by:

In the res folder in your Android Studio project structure, right click and select New> select Android Resource Directory as in the picture below:



Select the menu in the resource type and select the OK button.

item_navigasi.xml

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

1. <?xml version="1.0" encoding="utf-8"?>
2. <menu xmlns:android="http://schemas.android.com/apk/res/android">
3.     <group android:checkableBehavior="all">
4.         <item
5.             android:id="@+id/navigation1"
6.             android:checked="false"
7.             android:icon="@drawable/ic_beranda"
8.             android:title="@string/navigation_view_item_1" />
9.         <item
10.            android:id="@+id/navigation2"
11.            android:checked="false"
12.            android:icon="@drawable/ic_profil"
13.            android:title="@string/navigation_view_item_2" />
14.         <item
15.             android:id="@+id/navigation3"
16.             android:checked="false"
17.             android:icon="@drawable/ic_deftar"
18.             android:title="@string/navigation_view_item_3" />
19.
20.
21.     </group>
22.
23.
24.     <item
25.         android:id="@+id/navigasi_kategori_2"
26.         android:title="@string/navigasi_kategori_2">
27.         <menu>
28.             <item
29.                 android:id="@+id/navigation4"
30.                 android:checked="false"
31.                 android:icon="@drawable/ic_setting"
32.                 android:title="@string/navigation_view_item_4" />
33.             <item
34.                 android:id="@+id/navigation5"
35.                 android:checked="false"
36.                 android:icon="@drawable/ic_about"
37.                 android:title="@string/navigation_view_item_5" />
38.
39.         </menu>
40.     </item>
41. </menu>
```

In the file we will define any navigation menu item, which we will display.

- **android: id = "@ + id / navigation1"** = id identification which will be called in the Java Activity file.
- **android: icon = "@ drawable / ic_beranda"** = call the vector image icon file contained in the drawable resource folder we have created.
- **android: title = "@ string / navigation_view_item_1"** = calling the text data file contained in the resource strings.xml folder.

5) Layout

In the layout folder, we will apply the instruction line (codingan) below in the activity_main.xml file, layout_header.xml (header navigation drawer), layout_toolbar.xml (toolbar for Actionbar).

activity_main.xml

DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

```
1. <android.support.v4.widget.DrawerLayout
2.     xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:id="@+id/drawer"
6.     android:layout_width="match_parent"
7.     android:layout_height="match_parent"
8.     android:fitsSystemWindows="true"
9.     tools:context=".MainActivity">
10.
11.    <LinearLayout
12.        android:layout_width="match_parent"
13.        android:layout_height="match_parent"
14.        android:measureWithLargestChild="false"
15.        android:orientation="vertical">
16.
17.        <include
18.            android:id="@+id/toolbar"
19.            layout="@layout/layout_toolbar" />
20.        <TextView
21.            android:layout_width="wrap_content"
22.            android:layout_height="wrap_content"
23.            android:textAppearance="?android:attr/textAppearanceLarge"
24.            android:text="Ini adalah Contoh isi konten dengan widget TextView yang
dibungkus oleh Navigation Drawer"
25.            android:id="@+id/textView" android:layout_gravity="center_horizontal"/>
26.
27.    </LinearLayout>
28.    <android.support.design.widget.NavigationView
29.        android:id="@+id/navigation_view"
30.        android:layout_width="wrap_content"
31.        android:layout_height="match_parent"
32.        android:layout_gravity="start"
33.        app:headerLayout="@layout/layout_header"
34.        app:menu="@menu/item_navigasi" />
35. </android.support.v4.widget.DrawerLayout>
```

layout_header.xml

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

1. <?xml version="1.0" encoding="utf-8"?>
2. <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.         android:layout_width="match_parent"
4.         android:layout_height="190dp"
5.         android:background="#2ecc71"
6.         android:orientation="vertical">
7.     <ImageView
8.         android:layout_width="wrap_content"
9.         android:layout_height="wrap_content"
10.        android:layout_gravity="bottom"
11.        android:layout_marginLeft="100dp"
12.        android:padding="30dp"
13.        android:src="@drawable/icon_profile"
14.        android:layout_alignBottom="@+id/linearLayout"
15.        android:layout_alignParentRight="true"
16.        android:layout_alignParentEnd="true"/>
17.     <LinearLayout
18.         android:layout_width="wrap_content"
19.         android:layout_height="wrap_content"
20.         android:layout_gravity="bottom"
21.         android:orientation="vertical" android:layout_alignParentBottom="true"
22.         android:layout_alignParentLeft="true" android:layout_alignParentStart="true"
23.         android:id="@+id/linearLayout">
24.         <TextView
25.             android:id="@+id/name"
26.             android:layout_width="match_parent"
27.             android:layout_height="wrap_content"
28.             android:layout_marginBottom="5dp"
29.             android:layout_marginLeft="16dp"
30.             android:text="Okedroid.com"
31.             android:textAppearance="@style/TextAppearance.AppCompat.Body1"
32.             android:textStyle="bold" />
33.         <TextView
34.             android:layout_width="match_parent"
35.             android:layout_height="wrap_content"
36.             android:layout_marginBottom="16dp"
37.             android:layout_marginLeft="16dp"
38.             android:text="Blog Android and Teknologi"
39.             android:textColor="#ffff" />
40.     </LinearLayout>
41. </RelativeLayout>
```

layout_toolbar.xml

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.v7.widget.Toolbar
3.     xmlns:android="http://schemas.android.com/apk/res/android"
4.     android:layout_width="match_parent"
5.     android:layout_height="wrap_content"
6.     android:background="@color/colorPrimary"
7.     android:theme="@style/ThemeOverlay.AppCompat">
8. </android.support.v7.widget.Toolbar>
```

6) Activity

In the MainActivity.java file you can apply the instruction line below:

MainActivity.java

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. package com.okedroid.contohnavigationdrawer;
2.
3. import android.os.Bundle;
4. import android.support.design.widget.NavigationView;
5. import android.support.v4.widget.DrawerLayout;
6. import android.support.v7.app.ActionBarDrawerToggle;
7. import android.support.v7.app.AppCompatActivity;
8. import android.support.v7.widget.Toolbar;
9. import android.view.MenuItem;
10. import android.view.View;
11. import android.widget.Toast;
12. public class MainActivity extends AppCompatActivity {
13.     //Mendefinisikan variabel
14.     private Toolbar toolbar;
15.     private NavigationView navigationView;
16.     private DrawerLayout drawerLayout;
17.     @Override
18.     protected void onCreate(Bundle savedInstanceState) {
19.         super.onCreate(savedInstanceState);
20.         setContentView(R.layout.activity_main);
21.         // Menginisiasi Toolbar dan mensetting sebagai actionbar
22.         toolbar = (Toolbar) findViewById(R.id.toolbar);
23.         setSupportActionBar(toolbar);
24.         // Menginisiasi NavigationView
25.         navigationView = (NavigationView) findViewById(R.id.navigation_view);
26.         // Mengatur Navigasi View Item yang akan dipanggil untuk menangani item klik
27.         menu navigasi
28.         navigationView.setNavigationItemSelectedListener(new
29.             NavigationView.OnNavigationItemSelectedListener() {
30.                 // This method will trigger on item click of navigation menu
31.                 @Override
32.                 public boolean onNavigationItemSelected(MenuItem menuItem) {
33.                     // Memeriksa apakah item tersebut dalam keadaan dicek atau tidak,
34.                     if(menuItem.isChecked()) menuItem.setChecked(false);
35.                     else menuItem.setChecked(true);
36.                     // Menutup drawer item klik
37.                     drawerLayout.closeDrawers();
38.                     // Memeriksa untuk melihat item yang akan diklik dan melalukan aksi
39.                     switch (menuItem.getItemId()){
40.                         // pilihan menu item navigasi akan menampilkan pesan toast klik
41.                         // Kalian bisa menggantinya
42.                         // dengan intent activity
43.                         case R.id.navigation1:
44.                             Toast.makeText(getApplicationContext(), "Beranda Telah
Dipilih", Toast.LENGTH_SHORT).show();
45.                             return true;
46.                         case R.id.navigation2:
47.                             Toast.makeText(getApplicationContext(),"Profil Telah
Dipilih",Toast.LENGTH_SHORT).show();
48.                             return true;
49.                     }
50.                 }
51.             });
52.     }
53. }
```

```

46.             case R.id.navigation3:
47.                 Toast.makeText(getApplicationContext(),"Daftar telah
Dipilih",Toast.LENGTH_SHORT).show();
48.                 return true;
49.             case R.id.navigation4:
50.                 Toast.makeText(getApplicationContext(),"Setting telah
dipilih",Toast.LENGTH_SHORT).show();
51.                 return true;
52.             case R.id.navigation5:
53.                 Toast.makeText(getApplicationContext(),"About telah
dipilih",Toast.LENGTH_SHORT).show();
54.                 return true;
55.             default:
56.                 Toast.makeText(getApplicationContext(),"Kesalahan Terjadi
",Toast.LENGTH_SHORT).show();
57.                 return true;
58.             }
59.         }
60.     });
61.     // Menginisiasi Drawer Layout dan ActionBarToggle
62.     drawerLayout = (DrawerLayout) findViewById(R.id.drawer);
63.     ActionBarDrawerToggle actionBarDrawerToggle = new
ActionBarDrawerToggle(this,drawerLayout,toolbar,R.string.openDrawer,
R.string.closeDrawer){
64.         @Override
65.         public void onDrawerClosed(View drawerView) {
66.             // Kode di sini akan merespons setelah drawer menutup disini kita
biarkan kosong
67.             super.onDrawerClosed(drawerView);
68.         }
69.         @Override
70.         public void onDrawerOpened(View drawerView) {
71.             // Kode di sini akan merespons setelah drawer terbuka disini kita
biarkan kosong
72.             super.onDrawerOpened(drawerView);
73.         }
74.     };
75.     //Mensetting actionBarToggle untuk drawer Layout
76.     drawerLayout.setDrawerListener(actionBarDrawerToggle);
77.     //memanggil synstate
78.     actionBarDrawerToggle.syncState();
79. }
80.
81.
82. }

```

7) Run project

After finished all we try to run the application via Android Studio. The result will look more or less like this.

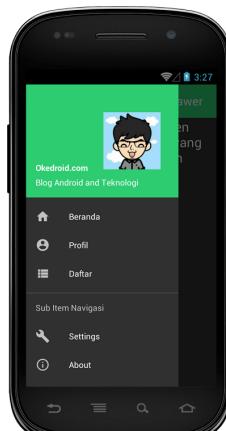


Figure 3.13 Result Display Navigation Drawer

6. TAB MENU

The tab menu is often used to display menus in applications. Many applications use tabbed menus such as the built-in android media player.

- 1) Create a new project named MenuTab
- 2) Then open the folder res => layout => activity_main.xml. Here are 3 components: TabHost, TabWidget and FrameLayout. TabWidget is used to display tab menus that we create. And FrameLayout is used to display the contents of the tab menu. Here's the full code.

```
<TabHost xmlns:android="http://schemas.android.com/apk/res
    android:id="@+id/tabhost"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical" >

        <TabWidget
            android:id="@+id/tabs"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom" />

        <FrameLayout
            android:id="@+id/tabcontent"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_gravity="bottom" />

    </LinearLayout>
</TabHost>
```

- 3) Next go into the src -> MainActivity.java folder to create tab menus. MainActivity under using TabActivity extends which means that the class contained in TabActivity can be downgraded to MainActivity.

```
package com.example.menutab;

import android.os.Bundle;
import android.app.TabActivity;
import android.content.Intent;
import android.widget.TabHost;

@SuppressWarnings("deprecation")
public class MainActivity extends TabActivity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        TabHost tabhost = getTabHost();
        TabHost.TabSpec spec;
        Intent intent;

        intent = new Intent().setClass(this, BerandaActiviti
spec = tabhost.newTabSpec("beranda").setIndicator(
tabhost.addTab(spec); //untuk membuat tabbaru disini

        intent = new Intent().setClass(this, BeritaActivit
spec = tabhost.newTabSpec("berita").setIndicator("
tabhost.addTab(spec);

        intent = new Intent().setClass(this, TemanActivity
spec = tabhost.newTabSpec("teman").setIndicator("T
tabhost.addTab(spec);

    }
}
```

- 4) Then we set up the layout to display in the tab menu.
- 5) Source home.xml. This layout is used to display the home menu.

```
<?xml version="1.0" encoding="utf-8"?>
<DigitalClock xmlns:android="http://schemas.android.com/apk
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:gravity="center_vertical|center"
    android:textSize="50sp" >

</DigitalClock>
```

- 6) Source news.xml. This layout is used to display news menu.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/ap
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >
    <ListView
        android:id="@+id/list"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        />
</LinearLayout>
```

- 7) Source teman.xml. This layout is used to display the friend menu.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/ap
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >
    <ListView
        android:id="@+id/list"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        />
</LinearLayout>
```

- 8) Next create a new class to call the layout "we have created.
- 9) BerandaActivity.java source code.

```
package com.example.menutab;

import android.app.Activity;
import android.os.Bundle;

public class BerandaActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.beranda);
    }
}
```

10) BeritaActivity.java source code.

```
package com.example.menutab;

import android.app.ListActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;

public class BeritaActivity extends ListActivity {
    String [] berita ={"Jadwal Piala Dunia 2014", "Capres
@Override
public void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.berita);

    setListAdapter(new ArrayAdapter<String>(this, andr
}
}
```

11) TemanActivity.java source code.

```
package com.example.menutab;

import android.app.ListActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;

public class TemanActivity extends ListActivity {
    String [] teman ={"Andra", "Dina", "Edo", "Julia"};
@Override
public void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.teman);

    setListAdapter(new ArrayAdapter<String>(this, andr
}
}
```

12) Do not forget to register the activity that we created to Androidmanifest.xml code as follows.

```
<activity android:name="BerandaActivity"></activity>
<activity android:name="BeritaActivity"></activity>
<activity android:name="TemanActivity"></activity>
```

Here are the results:



Figure 3.14Result Display Tab Menu

7. FRAGMENT

Fragment is one component of the interface (user interface) which is a part of Activity, can also be called by the name of Sub-Activity. One Activity can manage multiple fragments. To display results on a user screen. In One Activity too, a fragment can be replaced, added and removed, as needed. Fragment is affected from Activity's lifecycle (lifecycle), because Fragment is part of Activity. Here are the steps to create a fragment:

- 1) Create a new project
- 2) Create a new activity

Here will need 2 pieces of java files and 2 pieces of file layout, for it created a new Activity first. For example I will create an activity file named Fragment1.java, Fragment2.java and file layout fragment_satu.xml, fragment_dua.xml.

- 3) Applying activity source code

Layout

activity_main.xml

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2.         android:layout_width="fill_parent"
3.         android:layout_height="fill_parent" >
4.
5.     <fragment
6.         android:id="@+id/fragment2"
7.         android:name="com.okedroid.aplikasisaya.Fragment2"
8.         android:layout_width="0px"
9.         android:layout_height="match_parent"
10.        android:layout_weight="1"
11.    />
12.
13.    <fragment
14.        android:id="@+id/fragment1"
15.        android:name="com.okedroid.aplikasisaya.Fragment1"
16.        android:layout_width="0px"
17.        android:layout_height="match_parent"
18.        android:layout_weight="1"
19.    />
20.
21. </LinearLayout>
```

fragment_satu.xml

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.         android:layout_width="match_parent"
4.         android:layout_height="match_parent"
5.         android:orientation="vertical"
6.         android:background="#3498db"
7.     >
8.
9.     <TextView
10.        android:id="@+id/textView1"
11.        android:layout_width="wrap_content"
12.        android:layout_height="wrap_content"
13.        android:text="Fragment Pertama"
14.        android:textAppearance="?android:attr/textAppearanceLarge" />
15.
16. </LinearLayout>
```

fragment_dua.xml

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.         android:layout_width="match_parent"
4.         android:layout_height="match_parent"
5.         android:orientation="vertical"
6.         android:background="#2980b9"
7.     >
8.
9.     <TextView
10.        android:id="@+id/textView1"
11.        android:layout_width="wrap_content"
12.        android:layout_height="wrap_content"
13.        android:text="Fragment Kedua"
14.        android:textAppearance="?android:attr/textAppearanceLarge" />
15.
16. </LinearLayout>
```

Java

MainActivity.java

DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

```
1. package com.okedroid.aplikasisaya;
2.
3. import android.os.Bundle;
4. import android.support.v7.app.AppCompatActivity;
5.
6. public class MainActivity extends AppCompatActivity {
7.
8.
9.
10.    // Button variables
11.
12.
13.    @Override
14.    protected void onCreate(Bundle savedInstanceState) {
15.        super.onCreate(savedInstanceState);
16.        setContentView(R.layout.activity_main);
17.
18.    }
19.
20.
21. }
```

Fragment1.java

DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

```
1. package com.okedroid.aplikasisaya;
2.
3.
4. import android.os.Bundle;
5. import android.app.Fragment;
6. import android.view.LayoutInflater;
7. import android.view.View;
8. import android.view.ViewGroup;
9.
10. public class Fragment1 extends Fragment {
11.     @Override
12.     public View onCreateView(LayoutInflater inflater, ViewGroup container,
13.                             Bundle savedInstanceState) {
14.         // TODO Auto-generated method stub
15.         return inflater.inflate(R.layout.fragment_satu,container, false);
16.     }
17.
18. }
```

Fragment2.java

```
DATA HOSTED WITH ❤ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL

1. package com.okedroid.aplikasisaya;
2.
3.
4. import android.os.Bundle;
5. import android.app.Fragment;
6. import android.view.LayoutInflater;
7. import android.view.View;
8. import android.view.ViewGroup;
9.
10. public class Fragment2 extends Fragment {
11.     @Override
12.     public View onCreateView(LayoutInflater inflater, ViewGroup container,
13.         Bundle savedInstanceState) {
14.         // TODO Auto-generated method stub
15.         return inflater.inflate(R.layout.fragment_dua,container, false);
16.     }
17.
18. }
```

4) Run the application

After successfully applied we try to run the application via Android Studio. The result would be something like this:

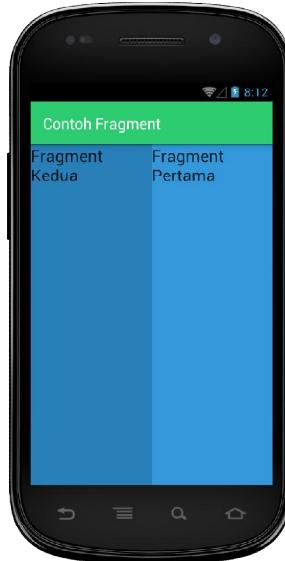


Figure 3.15Result Display Fragment

MODUL 4

MAPS

Google Maps is a google navigation map that helps the user search, view and find the location of the place where you want to go and where you want to go. Google Maps is not only available on the Web, but it is also available in mobile applications especially Android. Well Google provides an API used for Android developers, developing their Navigation Maps app called Google Maps API.

1. CURRENT LOCATION

This section uses one of the features of the Google Maps API, which displays the location area around the user, which is marked with a marker on the Google Maps screen. Here are the steps to display the current location.

1) Manifest

First on your Android Studio project in the manifest folder> *AndroidManifest.xml*. Then add the following user permissions to the *AndroidManifest.xml* file.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
```

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <?xml version="1.0" encoding="utf-8"?>
2. <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.     package="com.okedroid.googlemaps">
4.
5.     <!--
6.         The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
7.         Google Maps Android API v2, but you must specify either coarse or fine
8.         location permissions for the 'MyLocation' functionality.
9.     -->
10.    <uses-permission android:name="android.permission.INTERNET" />
11.    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
12.    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
13.
14.    <application
15.        android:allowBackup="true"
16.        android:icon="@mipmap/ic_launcher"
17.        android:label="@string/app_name"
18.        android:supportRtl="true"
19.        android:theme="@style/AppTheme">
20.
21.        <!--
22.            The API key for Google Maps-based APIs is defined as a string resource.
23.            (See the file "res/values/google_maps_api.xml").
24.            Note that the API key is linked to the encryption key used to sign the
25.            APK.
26.            You need a different API key for each encryption key, including the
27.            release key that is used to
28.                sign the APK for publishing.
29.            You can define the keys for the debug and release targets in src/debug/
30.            and src/release/.
31.        -->
32.        <meta-data
33.            android:name="com.google.android.geo.API_KEY"
34.            android:value="@string/google_maps_key"/>
35.
36.        <activity
37.            android:name=".MapsActivity"
38.            android:label="@string/title_activity_maps">
39.            <intent-filter>
40.                <action android:name="android.intent.action.MAIN" />
41.                <category android:name="android.intent.category.LAUNCHER" />
42.            </intent-filter>
43.        </activity>
44.    </application>
```

2) Layout

After that in the file layout *activity_maps.xml*. Make sure the instruction line is like this:

activity_maps.xml

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. <?xml version="1.0" encoding="utf-8"?><!--
2. Copyright (C) 2012 The Android Open Source Project
3. Licensed under the Apache License, Version 2.0 (the "License");
4. you may not use this file except in compliance with the License.
5. You may obtain a copy of the License at
6.     http://www.apache.org/licenses/LICENSE-2.0
7. Unless required by applicable law or agreed to in writing, software
8. distributed under the License is distributed on an "AS IS" BASIS,
9. WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
10. See the License for the specific language governing permissions and
11. limitations under the License.
12. -->
13. <!-- This can go anywhere in your layout (see other demos for some examples). -->
14. <FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
15.             android:layout_width="match_parent"
16.             android:layout_height="match_parent"
17.             android:id="@+id/layout">
18.
19.     <fragment
20.         android:id="@+id/map"
21.         class="com.google.android.gms.maps.SupportMapFragment"
22.         xmlns:android="http://schemas.android.com/apk/res/android"
23.         android:layout_width="match_parent"
24.         android:layout_height="match_parent" />
25. </FrameLayout>
```

3) Activity

In the *MainActivity.java* java file, copy the instruction line below:

MainActivity.java

```
DATA HOSTED WITH ♥ BY PASTEBIN.COM - DOWNLOAD RAW - SEE ORIGINAL
1. package com.okdroid.googlemaps;
2.
3. import android.Manifest;
4. import android.content.Context;
5. import android.content.pm.PackageManager;
6. import android.location.Criteria;
7. import android.location.Location;
8. import android.location.LocationListener;
9. import android.location.LocationManager;
10. import android.location.LocationProvider;
11. import android.support.v4.app.FragmentActivity;
12. import android.os.Bundle;
13. import android.util.Log;
14. import com.google.android.gms.maps.CameraUpdateFactory;
15. import com.google.android.gms.maps.GoogleMap;
16. import com.google.android.gms.maps.OnMapReadyCallback;
17. import com.google.android.gms.maps.SupportMapFragment;
18. import com.google.android.gms.maps.model.BitmapDescriptorFactory;
19. import com.google.android.gms.maps.model.CameraPosition;
20. import com.google.android.gms.maps.model.CircleOptions;
21. import com.google.android.gms.maps.model.LatLng;
22. import com.google.android.gms.maps.model.Marker;
23. import com.google.android.gms.maps.model.MarkerOptions;
```

```
25. public class MapsActivity extends FragmentActivity implements OnMapReadyCallback,
26. LocationListener {
27.
28.     private GoogleMap mMap;
29.     private LocationManager mLocationManager = null;
30.     private String provider = null;
31.     private Marker mCurrentPosition = null;
32.
33.     @Override
34.     protected void onCreate(Bundle savedInstanceState) {
35.         super.onCreate(savedInstanceState);
36.         setContentView(R.layout.activity_maps);
37.         // Obtain the SupportMapFragment and get notified when the map is ready to
38.         // be used.
39.         SupportMapFragment mapFragment = (SupportMapFragment)
40.             getSupportFragmentManager()
41.                 .findFragmentById(R.id.map);
42.         mapFragment.getMapAsync(this);
43.     }
44.
45.     @Override
46.     public void onMapReady(GoogleMap googleMap) {
47.         mMap = googleMap;
48.         mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
49.
50.         if (isProviderAvailable() && (provider != null)) {
51.             locateCurrentPosition();
52.         }
53.
54.         private void locateCurrentPosition() {
55.
56.             int status =
57.                 PackageManager.PERMISSION_GRANTED);
58.             PackageManager.getPackageName());
59.
60.             if (status == PackageManager.PERMISSION_GRANTED) {
61.                 Location location = mLocationManager.getLastKnownLocation(provider);
62.                 updateWithNewLocation(location);
63.                 // mLocationManager.addGpsStatusListener(this);
64.                 long minTime = 5000;// ms
65.                 float minDist = 5.0f;// meter
66.                 mLocationManager.requestLocationUpdates(provider, minTime, minDist,
67.                     this);
68.             }
69.         }
70.     }
```

```

69.     private boolean isProviderAvailable() {
70.         mLocationManager = (LocationManager) getSystemService(
71.             Context.LOCATION_SERVICE);
72.         Criteria criteria = new Criteria();
73.         criteria.setAccuracy(Criteria.ACCURACY_COARSE);
74.         criteria.setAltitudeRequired(false);
75.         criteria.setBearingRequired(false);
76.         criteria.setCostAllowed(true);
77.         criteria.setPowerRequirement(Criteria.POWER_LOW);
78.
79.         provider = mLocationManager.getBestProvider(criteria, true);
80.         if (mLocationManager
81.             .isProviderEnabled(LocationManager.NETWORK_PROVIDER)) {
82.             provider = LocationManager.NETWORK_PROVIDER;
83.
84.             return true;
85.         }
86.
87.         if (mLocationManager.isProviderEnabled(LocationManager.GPS_PROVIDER)) {
88.             provider = LocationManager.GPS_PROVIDER;
89.             return true;
90.         }
91.
92.         if (provider != null) {
93.             return true;
94.         }
95.         return false;
96.     }
97.
98.     private void updateWithNewLocation(Location location) {
99.
100.        if (location != null && provider != null) {
101.            double lng = location.getLongitude();
102.            double lat = location.getLatitude();
103.
104.            addBoundaryToCurrentPosition(lat, lng);
105.
106.            CameraPosition camPosition = new CameraPosition.Builder()
107.                .target(new LatLng(lat, lng)).zoom(10f).build();
108.
109.            if (mMap != null)
110.                mMap.animateCamera(CameraUpdateFactory
111.
112.                    .newCameraPosition(camPosition));
113.        } else {
114.            Log.d("Location error", "Something went wrong");
115.        }
116.    }

```

```

119.     private void addBoundaryToCurrentPosition(double lat, double lang) {
120.
121.         MarkerOptions mMarkerOptions = new MarkerOptions();
122.         mMarkerOptions.position(new LatLng(lat, lang));
123.         mMarkerOptions.icon(BitmapDescriptorFactory
124.             .fromResource(R.drawable.marker));
125.         mMarkerOptions.anchor(0.5f, 0.5f);
126.
127.         CircleOptions mOptions = new CircleOptions()
128.             .center(new LatLng(lat, lang)).radius(10000)
129.             .strokeColor(0x110000FF).strokeWidth(1).fillColor(0x110000FF);
130.         mMap.addCircle(mOptions);
131.         if (mCurrentPosition != null)
132.             mCurrentPosition.remove();
133.         mCurrentPosition = mMap.addMarker(mMarkerOptions);
134.     }
135.
136.
137.     @Override
138.     public void onLocationChanged(Location location) {
139.
140.         updateWithNewLocation(location);
141.     }
142.
143.     @Override
144.     public void onProviderDisabled(String provider) {
145.
146.         updateWithNewLocation(null);
147.     }
148.
149.     @Override
150.     public void onProviderEnabled(String provider) {
151.
152.     }
153.
154.     @Override
155.     public void onStatusChanged(String provider, int status, Bundle extras) {
156.         switch (status) {
157.             case LocationProvider.OUT_OF_SERVICE:
158.                 break;
159.             case LocationProvider.TEMPORARILY_UNAVAILABLE:
160.                 break;
161.             case LocationProvider.AVAILABLE:
162.                 break;
163.         }
164.     }
165. }
```

In the .fromResource (R.drawable.marker) section); if there is error then make sure you have set up marker then save in resource location> drawable> marker.png

4) Run the application

After all is done, let's try to test the Application. By running the Application in Android Studio. The result will look something like this:

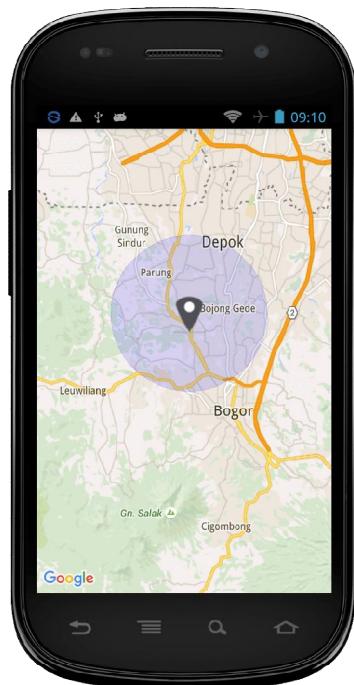


Figure 4.1 Result Display Maps

2. PLACE PICKER

First of all, make sure Google Places API for Android is already enabled on your Developer Console project. The way to go to this page and then go to API> Google Places for Android and then click "Enable API". After that create a new project on Android Studio.

After that go into your build.gradle file and add dependencies as follows:

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:25.1.1'
    compile 'com.google.android.gms:play-services-places:10.0.1'
}
```

After that, we start by making the layout first. This main layout is called *activity_main.xml* and contains only one key and one TextView.

```
activity_main.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="id.web.twoh.placesapitutorial.MainActivity">

    <Button
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/bt_ppicker"
        android:text="Launch Place Picker"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:layout_below="@+id/bt_ppicker"
        android:id="@+id/tv_place_id"
        android:text="Place name..."
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Then we will create the main code in the main activity, open the MainActivity.java class, or other classes that become main activity on your Android Studio project. Then copy-paste the following code:

MainActivity.java

```
package id.web.twoh.placesapitutorial;

import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.common.GooglePlayServicesNotAvailableException;
import com.google.android.gms.common.GooglePlayServicesRepairableException;
import com.google.android.gms.location.places.Place;
import com.google.android.gms.location.places.ui.PlacePicker;

public class MainActivity extends ActionBarActivity {

    private Button btPlacesAPI;
    private TextView tvPlaceAPI;
    // konstanta untuk mendeksi hasil balikan dari place picker
    private int PLACE_PICKER_REQUEST = 1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        tvPlaceAPI = (TextView) findViewById(R.id.tv_place_id);
        btPlacesAPI = (Button)findViewById(R.id.bt_ppicker);
        btPlacesAPI.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                // membuat Intent untuk Place Picker
                PlacePicker.IntentBuilder builder = new PlacePicker.IntentBuilder();
                try {
                    //menjalankan place picker
                    startActivityForResult(builder.build(MainActivity.this), PLACE_PICKER_REQUEST);
                } catch (GooglePlayServicesRepairableException e) {
                    e.printStackTrace();
                } catch (GooglePlayServicesNotAvailableException e) {
                    e.printStackTrace();
                }
            }
        });
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // menangkap hasil balikan dari Place Picker, dan menampilkannya pada TextView
        if (requestCode == PLACE_PICKER_REQUEST) {
            if (resultCode == RESULT_OK) {
                Place place = PlacePicker.getPlace(data, this);
                String toastMsg = String.format(
                        "Place: %s \n" +
                        "Alamat: %s \n" +
                        "Latlng %s \n", place.getName(), place.getAddress(), place.getLatLng());
                tvPlaceAPI.setText(toastMsg);
            }
        }
    }
}
```

Then the last one is to add permissions-permissions required on Android Manifest. The required permissions are more or less the same as when we want to display Maps on Android, such as API Key, GMS Version, Internet permissions, location and so on. How, on Android Studio, open AndroidManifest.xml and add the permissions or the missing sections in your Android Manifest according to the following code:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.web.twoh.placesapitutorial" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVI
    <!--
The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
Google Maps Android API v2, but are recommended.
-->
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="@string/google_maps_key" />
        <activity
            android:name=".MainActivity"
            android:label="TWOHsPlacesAPI" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

```

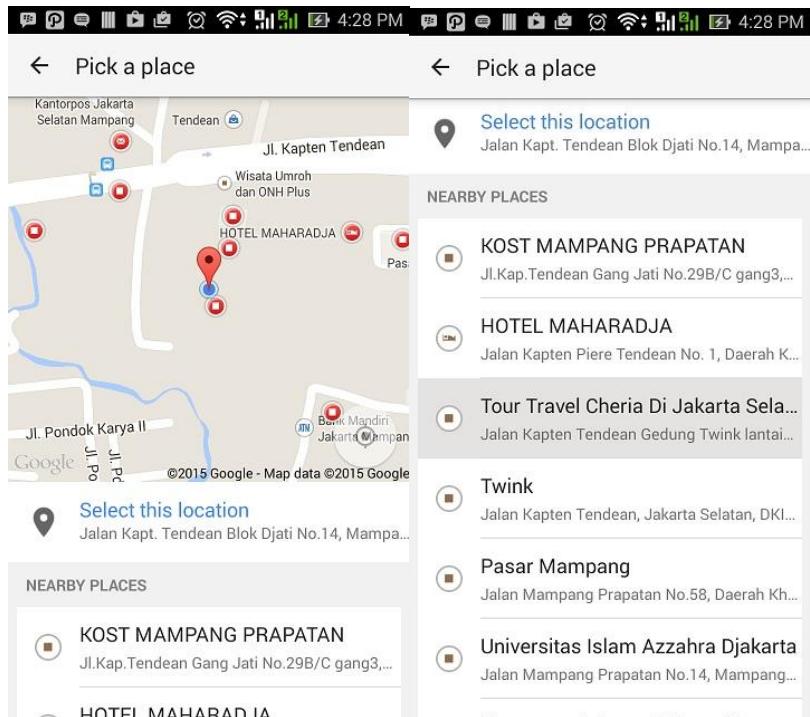


Figure 4.2 Result Display Place Picker

3. SEARCH ADDRESS



Figure 4.3Result Display Search Address

Open your activity_maps.xml file and a LinearLayout as root layout and add EditText and Button inside LinearLayout. EditText is used to type location name to search in map, Button is for search location after typing something in address bar. Give an id to edit text and button and add onClick attribute in button. Following is the complete content of maps XML layout file.

res/layout/activity_maps.xml

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      android:layout_width="match_parent"
3      android:layout_height="wrap_content"
4      android:orientation="vertical">
5
6      <LinearLayout
7          android:layout_width="match_parent"
8          android:layout_height="wrap_content"
9          android:layout_margin="5dp"
10
11         android:orientation="horizontal">
12
13         <EditText
14             android:id="@+id/editText"
15             android:layout_width="wrap_content"
16             android:layout_height="wrap_content"
17             android:layout_weight="4"
18             android:hint="Search Location Here" />
19
20
21         <Button
22             android:id="@+id/search_button"
23             android:layout_width="wrap_content"
24             android:layout_height="wrap_content"
25             android:layout_weight="0.5"
26             android:onClick="onMapSearch"
27             android:text="Search" />
28
29     </LinearLayout>
30
31     <fragment xmlns:android="http://schemas.android.com/apk/res/android"
32         xmlns:tools="http://schemas.android.com/tools"
33         android:id="@+id/map"
34         android:name="com.google.android.gms.maps.SupportMapFragment"
35         android:layout_width="match_parent"
36         android:layout_height="match_parent"
37         tools:context="com.viralandroid.googlemapsandroidapiMapsActivity" />
38
39 </LinearLayout>
```

Now open your java activity file MapsActivity.java and add following lines of code. The complete java code of MapsActivity.java file will look like below.

MainActivity.java

```
package com.viralandroid.googlemapsandroidapi;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.FragmentActivity;
import android.view.View;
import android.widget.EditText;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

import java.io.IOException;
import java.util.List;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    public void onMapSearch(View view) {
        EditText locationSearch = (EditText) findViewById(R.id.editText);
        String location = locationSearch.getText().toString();
        List<Address> addressList = null;

        if (location != null || !location.equals("")) {
            Geocoder geocoder = new Geocoder(this);
            try {
                addressList = geocoder.getFromLocationName(location, 1);

            } catch (IOException e) {
                e.printStackTrace();
            }
            Address address = addressList.get(0);
            LatLng latlng = new LatLng(address.getLatitude(), address.getLongitude());
            mMap.addMarker(new MarkerOptions().position(latlng).title("Marker"));
            mMap.animateCamera(CameraUpdateFactory.newLatLng(latlng));
        }
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        // Add a marker in Sydney and move the camera
        LatLng sydney = new LatLng(27.746974, 85.301582);
        mMap.addMarker(new MarkerOptions().position(sydney).title("Kathmandu, Nepal"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
        if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            return;
        }
        mMap.setMyLocationEnabled(true);
    }
}
```

MODUL 5

DATABASE

Purpose:

Understand how the integration android and database

Material:

Create the database SQLite and do crud process (Insert, Update, delete, and Read)

Database. The use of databases in android is very important in the development android applications. The database used in this material is a MySQL database CRUD, CRU Dis insert, update, delete, perform.

➤ The Advantages Of SQLite:

- Do not require third parties to access the data
- Because these databases are portable, then the application can directly coupled with the applications that are often referred to as an embed
- If you want to copy the database, simply copy the file only
- is suitable in applications which are not connected to the internet applications, both desktop or mobile applications

➤ SQLite Deficiencies:

- Because these applications directly connected on the application without using intermediaries, so that SQLite is used to store data that is a little data or temporary, for example game the only store information score the winner.
- SQLite does not use the User Management, meaning that if get the file database, could be opened without using a username or password
- not all query command can be done on this SQLite
- Example a programmer wants to make a game application the same but on different platforms, such as android, iphone and blackBerry. Then the programmer had to create a database on the each of these platforms

➤ Advantages Of MySql:

- Database has better security, because access should use the username and password
- Has a complete query command
- If you create many applications with different platforms, simply access only one database
- can store data that overwhelmingly and accessed quickly(depending on the query that is used)

➤ Lack Of MySQL:

- Installation more complicated than SQLite
- must have a Server that connects databases with applications
- If you use Android, android should then connect to the server (usually using the internet). Due to access of Android to MySQL through PHP

1. Getting started

To make it easier to see the results of further work, please install

SQLite Manager on Android Studio:

[1. Download SQLite Manager_1.0.0.jar](#)

2. put the SQLite Manager in `dopins` folder (the location of the `/Android/Sdk/tools/lib/monitor-x86_64/plugins/ sqlitemanager_1.0.0.jar`)

3. on Android in the Android Studio, Android Device Monitor open Windows -> Show

View -> Other -> File Explorer

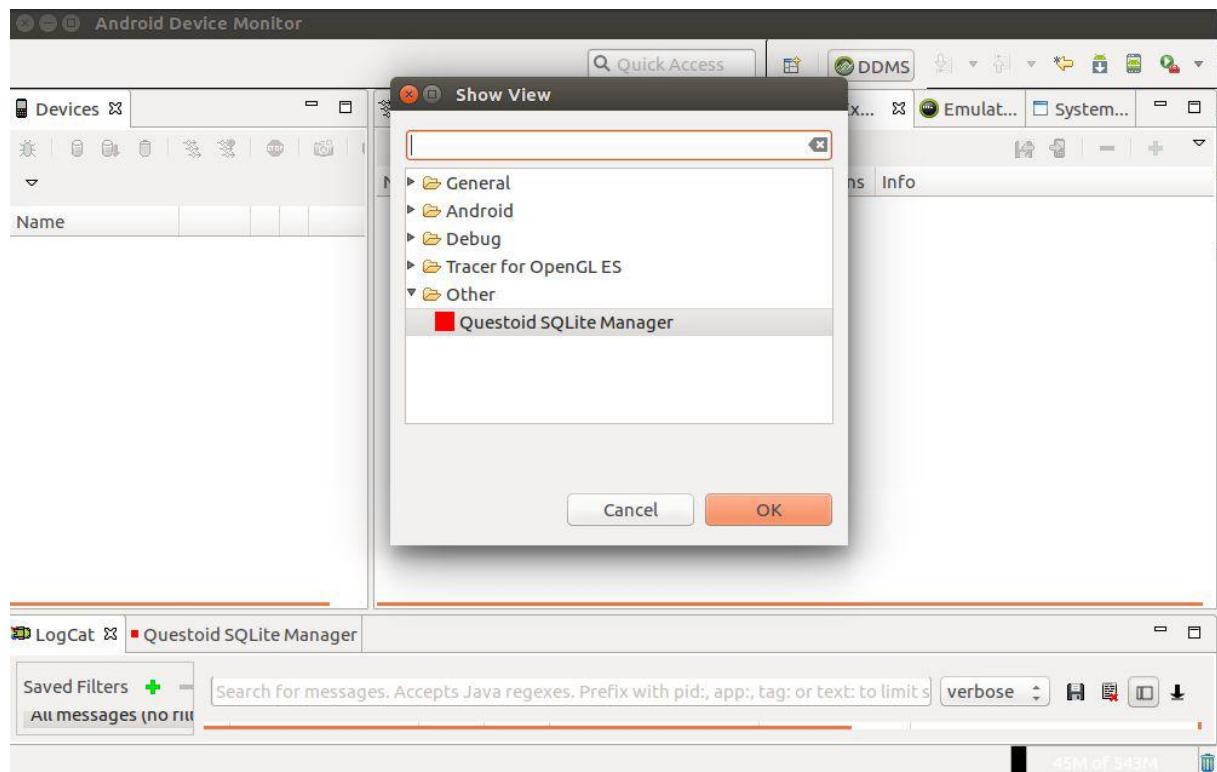


Figure 5.1 Getting started

If emulator has not been executed, the File Explorer will be empty, make sure the emulator is already running if you want to see the display as above.

2. creating a Project CRUD_SQLITE

1. open the Android's Studio, and then file->New->Other->Android->Android Application Project
2. Fill in each Field (for the other field may be made According to the wishes of each):

Application Name :**CRUD_SQLITE**

Package Name : **nama_nim.com**

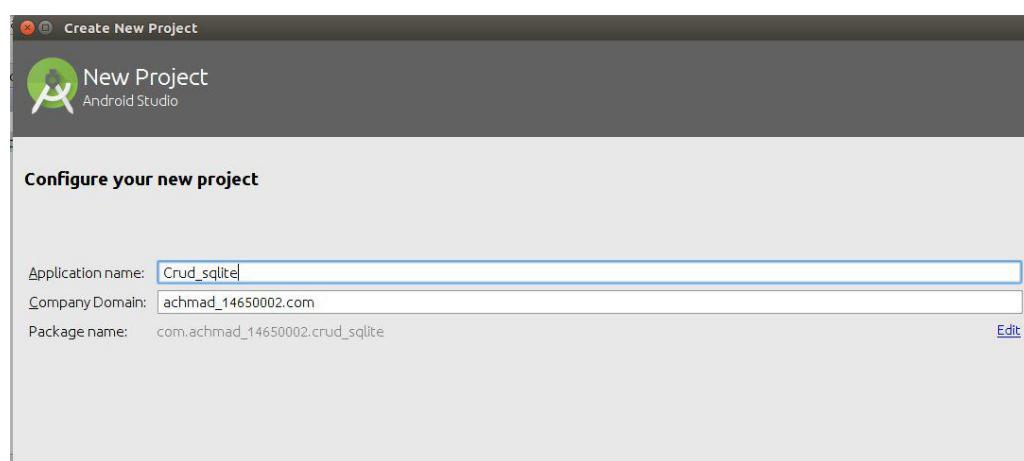


Figure 5.2Create New Project

3. Create a database_mahasiswa. db, tabel_biodata and the trials enteringData

To perform this test, the data used are not using the userinterface, so that the data entered is still using the data entered manual.

1. create a file with the name **SQLiteHelper.java**

2. in the **SQLiteHelper.java**, enter the following source code

```

import android.content.ContentValues;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
public class SQLiteHelper extends SQLiteOpenHelper{
    private static final String nama_database = "database_mahasiswa.db";
    private static final int versi_database = 1;
    private static final String query_buat_tabel_biodata_pemain = "CREATE TABLE IF NOT EXISTS tabel_biodata(id_biodata INTEGER PRIMARY KEY AUTOINCREMENT, nama TEXT, alamat TEXT)";
    private static final String query_hapus_tabel_biodata_pemain = "DROP TABLE IF EXISTS query_buat_tabel_biodata_pemain";
    public SQLiteHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {
        super(context, nama_database, null, versi_database);
    }
    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL(query_buat_tabel_biodata_pemain);
        System.out.println("tabel_biodata sudah dibuat");
    }
    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int versi_lama, int versi_baru) {
        sqLiteDatabase.execSQL(query_hapus_tabel_biodata_pemain);
        onCreate(sqLiteDatabase);
    }
    public void tambah_biodata(String nama, String alamat) {
        SQLiteDatabase database = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put("nama", nama);
        values.put("alamat", alamat);
        database.insert("tabel_biodata", null, values);
        database.close();
    }
}

```

If executed, the result can be found in the File

Explorer->data->data->nama_nim.com->databases->database_pemain.db

4. Method of displaying Data

To display the data, add the method `tampil_semua_biodata()` on a file

```

public ArrayList<HashMap<String, String>> tampil_semua_biodata() {
    // deklarasikan sebuah arraylist yang bisa menampung hashmap
    ArrayList<HashMap<String, String>> arrayListBiodata = new ArrayList<HashMap<String, String>>();
    SQLiteDatabase database = this.getReadableDatabase();
    Cursor cursor = database.rawQuery("SELECT * FROM tabel_biodata", null);
    // kursor langsung diarkan ke posisi paling awal data pada tabel_biodata
    if (cursor.moveToFirst()) {
        do {
            // deklarasikan sebuah hashmap, yang bisa menampung masing-masing field dari tabel_biodata
            HashMap<String, String> hashMapBiodata = new HashMap<String, String>();
            // masukkan masing-masing field dari tabel_biodata ke dalam hashMapBiodata
            // pastikan id_biodata, nama dan alamat sama persis dengan field yang ada pada tabel_biodata
            hashMapBiodata.put("id_biodata", cursor.getString(0));
            hashMapBiodata.put("nama", cursor.getString(1));
            hashMapBiodata.put("alamat", cursor.getString(2));
            // masukkan hashMapBiodata ke dalam arrayListBiodata
            arrayListBiodata.add(hashMapBiodata);
        } while (cursor.moveToNext());
    }
    return arrayListBiodata;
}

```

5. Method Update Data

Add the method update_biodata () on the SQLiteHelper.java

```
public int update_biodata(int id, String nama, String alamat) {
    SQLiteDatabase database = this.getWritableDatabase();
    ContentValues recordBiodata = new ContentValues();
    recordBiodata.put("nama", nama);
    recordBiodata.put("alamat", alamat);
    return database.update("tabel_biodata", recordBiodata, "id_biodata=" + id, null);
}
```

6. Method Hapus Data

Add the method hapus_biodata pada() on the SQLiteHelper.java

```
public void hapus_biodata(int id) {
    SQLiteDatabase database = this.getWritableDatabase();
    database.execSQL("DELETE FROM tabel_biodata WHERE id_biodata=" + id + "");
    database.close();
}
```

7. Method Fetch data based on ID

At this point the user will send the id and sqlite will send back the value of the to the sender only data based on the specified id

```
public HashMap<String, String> tampil_biodata_berdasarkan_id(int id) {
    SQLiteDatabase database = this.getReadableDatabase();
    HashMap<String, String> hashMapBiodata = new HashMap<String, String>();
    Cursor cursor = database.rawQuery("SELECT * FROM tabel_biodata WHERE id_biodata=" + id + "", null);

    if (cursor.moveToFirst()) {
        do {
            hashMapBiodata.put("id_biodata", cursor.getString(0));
            hashMapBiodata.put("nama", cursor.getString(1));
            hashMapBiodata.put("alamat", cursor.getString(2));
        } while (cursor.moveToNext());
    }

    return hashMapBiodata;
}
```

8 . CRUD with User Interface main.xml

On the previous stage has not used the look, so for testing still using the final data, order data is statistic processed, the user used user interface.



Figure 5.3Result Display Crud

```
<LinearLayout xmlns:android='http://schemas.android.com/apk/res/android'  
    xmlns:tools='http://schemas.android.com/tools'  
    android:id="@+id/LinearLayout1"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:layout_gravity="center"  
    android:orientation="vertical"  
    tools:context=".MainActivity" >  
  
    <Button  
        android:id="@+id/buttonTambahBiodata"  
        android:layout_width="186dp"  
        android:layout_height='wrap_content'  
        android:layout_gravity="center"  
        android:text="Tambah Biodata" />  
  
    <HorizontalScrollView  
        android:id="@+id/horizontalScrollView"  
        android:layout_width="match_parent"  
        android:layout_height='wrap_content' >  
  
        <ScrollView  
            android:id="@+id/verticalScrollView"  
            android:layout_width="wrap_content"  
            android:layout_height="match_parent" >  
  
            <TableLayout  
                android:id="@+id/tableBiodata"  
                android:layout_width="match_parent"  
                android:layout_height='wrap_content' >  
                </TableLayout>  
            </ScrollView>  
        </HorizontalScrollView>  
    </LinearLayout>
```

9. MainActivity.java

MainActivity.java like this

```

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import java.util.ArrayList;
import java.util.HashMap;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v4.view.ViewPager.LayoutParams;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity implements OnClickListener {

    SQLiteHelper sqLiteHelper = new SQLiteHelper(this);
    TableLayout tabelBiodata;
    Button buttonTambahBiodata;
    ArrayList<Button> buttonEdit = new ArrayList<>();
    ArrayList<Button> buttonDelete = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tabelBiodata = (TableLayout) findViewById(R.id.tableBiodata);
        buttonTambahBiodata = (Button) findViewById(R.id.buttonTambahBiodata);
        buttonTambahBiodata.setOnClickListener(this);

        TableRow barisTabel = new TableRow(this);
        barisTabel.setBackgroundColor(Color.RED);

        TextView viewHeaderId = new TextView(this);
        TextView viewHeaderNama = new TextView(this);
        TextView viewHeaderAlamat = new TextView(this);
        TextView viewHeaderAction = new TextView(this);

        viewHeaderId.setText("ID");
        viewHeaderNama.setText("Nama");
        viewHeaderAlamat.setText("Alamat");
        viewHeaderAction.setText("Action");

        viewHeaderId.setPadding(5, 1, 5, 1);
        viewHeaderNama.setPadding(5, 1, 5, 1);
        viewHeaderAlamat.setPadding(5, 1, 5, 1);
        viewHeaderAction.setPadding(5, 1, 5, 1);

        barisTabel.addView(viewHeaderId);
        barisTabel.addView(viewHeaderNama);
        barisTabel.addView(viewHeaderAlamat);
        barisTabel.addView(viewHeaderAction);
    }
}

```

```

tableBiodata.addView(barislabel, new TableLayout.LayoutParams(LayoutParams.WRAP_CONTENT, LayoutParams.WRAP_CONTENT));
ArrayList<HashMap<String, String>> array_listBiodata = sqliteHelper.tampil_semua_biodata();
if (arrayListBiodata.size() > 0) {
    for (int i = 0; i < arrayListBiodata.size(); i++) {
        //ambil mapping masing hashmap dari arrayListBiodata
        HashMap<String, String> hashMapRecordBiodata = arrayListBiodata.get(i);
        //JSONObject jsonChildNode = arrayBiodata.getJSONObject(i);
        String name = hashMapRecordBiodata.get("name");
        String alamat = hashMapRecordBiodata.get("alamat");
        String id = hashMapRecordBiodata.get("id_biodata");

        System.out.println("Nama :" + name);
        System.out.println("Alamat :" + alamat);
        System.out.println("ID :" + id);

        barisTabel = new TableRow(this);

        if (i % 2 == 0) {
            barisTabel.setBackgroundColor(Color.LTGRAY);
        }

        TextView viewId = new TextView(this);
        viewId.setText(id);
        viewId.setPadding(5, 1, 5, 1);
        barislabel.addView(viewId);

        TextView viewNama = new TextView(this);
        viewNama.setText(name);
        viewNama.setPadding(5, 1, 5, 1);
        barisTabel.addView(viewNama);

        TextView viewAlamat = new TextView(this);
        viewAlamat.setText(alamat);
        viewAlamat.setPadding(5, 1, 5, 1);
        barisTabel.addView(viewAlamat);

        buttonEdit.add(i, new Button(this));
        buttonEdit.get(i).setId(Integer.parseInt(ic));
        buttonEdit.get(i).setTag("Edit");
        buttonEdit.get(i).setText("Edit");
        buttonEdit.get(i).setOnClickListener(this);
        barisTabel.addView(buttonEdit.get(i));
    }
}

```

```

        tabelBiodata.addView(barisTabel, new TableLayout.LayoutParams(
            LayoutParams.MATCH_PARENT, LayoutParams.MATCH_PARENT));
    }
}
}

@Override
public void onClick(View view) {

    if (view.getId() == R.id.buttonTambahBiodata) {
        // Toast.makeText(MainActivity.this, "Button Tambah Data",
        // Toast.LENGTH_SHORT).show();
        tambahBiodata();
    } else {
        /*
         * Melakukan pengecekan pada data array, agar sesuai dengan index
         * masing-masing button
        */
        for (int i = 0; i < buttonEdit.size(); i++) {

            /* jika yang diklik adalah button edit */
            if (view.getId() == buttonEdit.get(i).getId()
                && view.getTag().toString().trim().equals("Edit")) {
                // Toast.makeText(MainActivity.this, "Edit : " +
                // buttonEdit.get(i).getId(), Toast.LENGTH_SHORT).show();
                int id = buttonEdit.get(i).getId();
                getDataByID(id);

            } /* jika yang diklik adalah button delete */
            else if (view.getId() == buttonDelete.get(i).getId()
                && view.getTag().toString().trim().equals("Delete")) {
                // Toast.makeText(MainActivity.this, "Delete : " +
                // buttonDelete.get(i).getId(), Toast.LENGTH_SHORT).show();
                int id = buttonDelete.get(i).getId();
                deleteBiodata(id);

            }
        }
    }
}

public void deleteBiodata(int id) {
    sqLiteHelper.hapus_biodata(id);
}

```

```

/* restart acrtivity */
finish();
startActivity(getIntent());

}

public void getDataByID(int id) {
    String namaEdit = null, alamatEdit = null;
    HashMap<String, String> hashMapBiodata = sqLiteHelper.tampil_biodata_berdasarkan_id(id);

    for (int i = 0; i < hashMapBiodata.size(); i++) {
        namaEdit = hashMapBiodata.get("nama");
        alamatEdit = hashMapBiodata.get("alamat");
    }

    LinearLayout layoutInput = new LinearLayout(this);
    layoutInput.setOrientation(LinearLayout.VERTICAL);

    // buat id tersembunyi di alertbuilder
    final TextView viewId = new TextView(this);
    viewId.setText(String.valueOf(id));
    viewId.setTextColor(Color.TRANSPARENT);
    layoutInput.addView(viewId);

    final EditText editNama = new EditText(this);
    editNama.setText(namaEdit);
    layoutInput.addView(editNama);

    final EditText editAlamat = new EditText(this);
    editAlamat.setText(alamatEdit);
    layoutInput.addView(editAlamat);

    AlertDialog.Builder builderEditBiodata = new AlertDialog.Builder(this);
    builderEditBiodata.setTitle("Update Biodata");
    builderEditBiodata.setView(layoutInput);
    builderEditBiodata.setPositiveButton("Update", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            String nama = editNama.getText().toString();
            String alamat = editAlamat.getText().toString();

            System.out.println("Nama : " + nama + " Alamat : "
                    + alamat);

            sqLiteHelper.update_biodata(Integer.parseInt(viewId.getText().toString()), editNama.getText().toString(), editAlamat.getText().toString());

            /* restart acrtivity */
            finish();
            startActivity(getIntent());
        }
    });
    builderEditBiodata.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });
    builderEditBiodata.show();
}

public void tambahBiodata() {
    /* layout akan ditampilkan pada AlertDialog */
    LinearLayout layoutInput = new LinearLayout(this);
    layoutInput.setOrientation(LinearLayout.VERTICAL);

    final EditText editNama = new EditText(this);
    editNama.setHint("Nama");
    layoutInput.addView(editNama);

    final EditText editAlamat = new EditText(this);
    editAlamat.setHint("Alamat");
    layoutInput.addView(editAlamat);

    AlertDialog.Builder builderInsertBiodata = new AlertDialog.Builder(this);
    builderInsertBiodata.setTitle("Insert Biodata");
    builderInsertBiodata.setView(layoutInput);
    builderInsertBiodata.setPositiveButton("Insert", new DialogInterface.OnClickListener() {

```

```

@Override
public void onClick(DialogInterface dialog, int which) {
    String nama = editNama.getText().toString();
    String alamat = editAlamat.getText().toString();

    System.out.println("Nama : " + nama + " Alamat : "
        + alamat);

    sqLiteHelper.tambah_biodata(nama, alamat);
    /* restart activity */
    finish();
    startActivity(getIntent());
}
});

builderInsertBiodata.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        dialog.cancel();
    }
});
builderInsertBiodata.show();
}
}

```

Output Program :



Figure 5.4 Output Display Crud

Practical

- try the entire code above to understand every line of code and run it on emulator

Practical Tasks

- Create a new project with the database and the table adjusts to the title of the final project

Practical

- try the entire code above to understand every line of code and run it on emulator

Optional Source Code

```
android.widget.LinearLayout;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity implements OnClickListener
{
SQLiteHelper sqLiteHelper = new SQLiteHelper(this);
TableLayout tabelBiodata;
Button buttonTambahBiodata;
ArrayList<Button>buttonEdit = new ArrayList<Button>();
ArrayList<Button>buttonDelete = new ArrayList<Button>();
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main );
tabelBiodata = (TableLayout) findViewById(R.id.tableBiodata );
buttonTambahBiodata = (Button) findViewById(R.id.buttonTambahBiodata );
buttonTambahBiodata.setOnClickListener(this);
TableRow barisTabel = new TableRow(this);
barisTabel.setBackgroundColor(Color.RED );
TextView viewHeaderId = new TextView(this);
TextView viewHeaderNama = new TextView(this);
TextView viewHeaderAlamat = new TextView(this);
TextView viewHeaderAction = new TextView(this);
viewHeaderId.setText("ID");
viewHeaderNama.setText("Nama");
viewHeaderAlamat.setText("Alamat");
viewHeaderAction.setText("Action");
viewHeaderId.setPadding(5, 1, 5, 1);
viewHeaderNama.setPadding(5, 1, 5, 1);
viewHeaderAlamat.setPadding(5, 1, 5, 1);
viewHeaderAction.setPadding(5, 1, 5, 1);
barisTabel.addView(viewHeaderId);
barisTabel.addView(viewHeaderNama);
barisTabel.addView(viewHeaderAlamat);
barisTabel.addView(viewHeaderAction);
tabelBiodata.addView(barisTabel, new
TableLayout.LayoutParamsLayoutParams.WRAP_CONTENT ,
LayoutParams.WRAP_CONTENT));
ArrayList<HashMap<String, String>> arrayListBiodata =
sqLiteHelper.tampil_semua_biodata();
if (arrayListBiodata.size() >0) {
for (int i = 0; i < arrayListBiodata.size(); i++) {
// ambil masing-masing hashmap dari arrayListBiodata
HashMap<String, String> hashMapRecordBiodata =
arrayListBiodata.get(i);
// JSONObject jsonChildNode = arrayBiodata.getJSONObject(i);
String name = hashMapRecordBiodata.get("nama");
String alamat = hashMapRecordBiodata.get("alamat");
String id = hashMapRecordBiodata.get("id_biodata");
System.out.println("Nama :" + name);
System.out.println("Alamat :" + alamat);
System.out.println("ID :" + id);
barisTabel = new TableRow(this);
if (i % 2 == 0) {
barisTabel.setBackgroundColor(Color.LTGRAY );
}
TextView viewId = new TextView(this);
viewId.setText(id);
viewId.setPadding(5, 1, 5, 1);
barisTabel.addView(viewId);
TextView viewNama = new TextView(this);
viewNama.setText(name);
viewNama.setPadding(5, 1, 5, 1);
barisTabel.addView(viewNama);
TextView viewAlamat = new TextView(this);
```

```

viewAlamat.setText(alamat);
viewAlamat.setPadding(5, 1, 5, 1);
barisTabel.addView(viewAlamat);
buttonEdit.add(i, new Button(this));
buttonEdit.get(i).setId(Integer.parseInt(id));
buttonEdit.get(i).setTag("Edit");
buttonEdit.get(i).setText("Edit");
buttonEdit.get(i).setOnClickListener(this);
barisTabel.addView(buttonEdit.get(i));
buttonDelete.add(i, new Button(this));
buttonDelete.get(i).setId(Integer.parseInt(id));
buttonDelete.get(i).setTag("Delete");
buttonDelete.get(i).setText("Delete");
buttonDelete.get(i).setOnClickListener(this);
barisTabel.addView(buttonDelete.get(i));
tblBiodata.addView(barisTabel, new TableLayout.LayoutParams(
LayoutParams.MATCH_PARENT, LayoutParams.MATCH_PARENT));
}
}
}
@Override
public void onClick(View view) {
if (view.getId() == R.id.buttonTambahBiodata) {
// Toast.makeText(MainActivity.this, "Button Tambah Data",
// Toast.LENGTH_SHORT).show();
tambahBiodata();
} else {
/*
* Melakukan pengecekan pada data array, agar sesuai dengan index
* masing-masing button
*/
for (int i = 0; i <buttonEdit.size(); i++) {
/* jika yang diklik adalah button edit */
if (view.getId() == buttonEdit.get(i).getId() && view.getTag().toString().trim().equals("Edit")) {
// Toast.makeText(MainActivity.this, "Edit : " +
// buttonEdit.get(i).getId(), Toast.LENGTH_SHORT).show();
int id = buttonEdit.get(i).getId();
getDataByID(id);
} /* jika yang diklik adalah button delete */
else if (view.getId() == buttonDelete.get(i).getId() && view.getTag().toString().trim().equals("Delete")) {
// Toast.makeText(MainActivity.this, "Delete : " +
// buttonDelete.get(i).getId(), Toast.LENGTH_SHORT).show();
int id = buttonDelete.get(i).getId();
deleteBiodata(id);
}
}
}
}
}
}

public void deleteBiodata(int id) {
sqLiteHelper.hapus_biodata(id);
/* restart acrtivity */
finish();
startActivity(getIntent());
}

public void getDataByID(int id) {
String namaEdit = null, alamatEdit = null;
HashMap<String, String> hashMapBiodata =
sqLiteHelper.tampil_biodata_berdasarkan_id(id);
for (int i = 0; i < hashMapBiodata.size(); i++) {
namaEdit = hashMapBiodata.get("nama");
alamatEdit = hashMapBiodata.get("alamat");
}
LinearLayout layoutInput = new LinearLayout(this);
layoutInput.setOrientation(LinearLayout.VERTICAL);
// buat id tersembunyi di alertbuilder
final TextView viewId = new TextView(this);

```

```

viewId.setText(String.valueOf ( id));
viewId.setTextColor(Color.TRANSPARENT );
layoutInput.addView(viewId);
final EditText editNama = new EditText(this);
editNama.setText(namaEdit);
layoutInput.addView(editNama);
final EditText editAlamat = new EditText(this);
editAlamat.setText(alamatEdit);
layoutInput.addView(editAlamat);
AlertDialog.Builder builderEditBiodata = new AlertDialog.Builder(this);
builderEditBiodata.setTitle("Update Biodata");
builderEditBiodata.setView(layoutInput);
builderEditBiodata.setPositiveButton("Update",new
DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {
String nama = editNama.getText().toString();
String alamat = editAlamat.getText().toString();
System.out.println("Nama : " + nama + " Alamat : "
+ alamat);
SQLiteHelper.update_biodata(Integer.parseInt (viewId.getText().toString()),
editNama.getText().toString(), editAlamat.getText().toString());
/* restart acrtivity */
finish();
startActivity(getIntent());
}
});
builderEditBiodata.setNegativeButton("Cancel",new
DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {
dialog.cancel();
}
});
builderEditBiodata.show();
}
public void tambahBiodata() {
/* layout akan ditampilkan pada AlertDialog */
LinearLayout layoutInput = new LinearLayout(this);
layoutInput.setOrientation(LinearLayout.VERTICAL );
final EditText editNama = new EditText(this);
editNama.setHint("Nama");
layoutInput.addView(editNama);
final EditText editAlamat = new EditText(this);
editAlamat.setHint("Alamat");
layoutInput.addView(editAlamat);
AlertDialog.Builder builderInsertBiodata = new
AlertDialog.Builder(this);
builderInsertBiodata.setTitle("Insert Biodata");
builderInsertBiodata.setView(layoutInput);
builderInsertBiodata.setPositiveButton("Insert",new
DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {
String nama = editNama.getText().toString();
String alamat = editAlamat.getText().toString();
System.out.println("Nama : " + nama + " Alamat : "
+ alamat);
SQLiteHelper.tambah_biodata(nama, alamat);
/* restart acrtivity */
finish();
startActivity(getIntent());
}
});
builderInsertBiodata.setNegativeButton("Cancel",new
DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {

```

```
dialog.cancel();
}
});
builderInsertBiodata.show();
}
} [
Optional jika bingung membaca kode mainactivity silahkan copas kode
dibawah]
SQLiteHelper.java
package com.achmad_14650002.crud_sqlite;
/**
 * Created by coldwarrior on 29/09/16.
 */
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import java.util.ArrayList;
import java.util.HashMap;
public class SQLiteHelper extends SQLiteOpenHelper{
private static final String nama_database = "database_mahasiswa.db";
private static final int versi_database = 1;
private static final String query_buat_tabel_biodata_pemain = "CREATE TABLE
IF NOT EXISTS tabel_biodata(id_biodata INTEGER PRIMARY KEY AUTOINCREMENT, nama
TEXT, alamat TEXT)";
private static final String query_hapus_tabel_biodata_pemain = "DROP TABLE
IF EXISTS query_buat_tabel_biodata_pemain";
public SQLiteHelper(Context context) {
super(context, nama_database , null, versi_database );
}
@Override
public void onCreate(SQLiteDatabase sqLiteDatabase) {
sqLiteDatabase.execSQL(query_buat_tabel_biodata_pemain );
System.out .println("tabel_biodata sudah dibuat");
}
@Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int versi_lama, int
versi_baru) {
sqLiteDatabase.execSQL(query_hapus_tabel_biodata_pemain );
onCreate(sqLiteDatabase);
}
public void tambah_biodata(String nama, String alamat) {
SQLiteDatabase database = this.getWritableDatabase();
ContentValues values = new ContentValues();
values.put("nama", nama);
values.put("alamat", alamat);
database.insert("tabel_biodata", null, values);
database.close();
}
public ArrayList<HashMap<String, String>> tampil_semua_biodata() {
// deklarasikan sebuah arraylist yang bisa menampung hashmap
ArrayList<HashMap<String, String>> arrayListBiodata = new
ArrayList<HashMap<String, String>>();
SQLiteDatabase database = this.getWritableDatabase();
Cursor cursor = database.rawQuery("SELECT * FROM tabel_biodata", null);
// kursor langsung diarkan ke posisi paling awal data pada tabel_biodata
if (cursor.moveToFirst()) {
do {
// deklarasikan sebuah hashmap, yang bisa menampung
HashMap<String, String> hashMapBiodata = new HashMap<String,
String>();
// masukkan masing-masing field dari tabel_biodata ke dalam
hashMapBiodata
//pastikan id_biodata, nama dan alamat sama persis dengan field
yang ada pada tabel_biodata
hashMapBiodata.put("id_biodata", cursor.getString(0));
hashMapBiodata.put("nama", cursor.getString(1));
arrayListBiodata.add(hashMapBiodata);
}
while (cursor.moveToNext());
}
return arrayListBiodata;
}
```

```

hashMapBiodata.put("alamat", cursor.getString(2));
// masukkan hashMapBiodata ke dalam arrayListBiodata
arrayListBiodata.add(hashMapBiodata);
} while (cursor.moveToNext());
}
return arrayListBiodata;
}
public int update_biodata(int id, String nama, String alamat) {
SQLiteDatabase database = this.getWritableDatabase();
ContentValues recordBiodata = new ContentValues();
recordBiodata.put("nama", nama);
recordBiodata.put("alamat", alamat);
return database.update("tabel_biodata", recordBiodata, "id_biodata=" +
id, null);
}
public void hapus_biodata(int id) {
SQLiteDatabase database = this.getWritableDatabase();
database.execSQL("DELETE FROM tabel_biodata WHERE id_biodata='" + id +
"'");
database.close();
}
public HashMap<String, String> tampil_biodata_berdasarkan_id(int id) {
SQLiteDatabase database = this.getReadableDatabase();
HashMap<String, String> hashMapBiodata = new HashMap<String, String>();
Cursor cursor = database.rawQuery("SELECT * FROM tabel_biodata WHERE
id_biodata=" + id + "", null);
if (cursor.moveToFirst()) {
do {
hashMapBiodata.put("id_biodata", cursor.getString(0));
hashMapBiodata.put("nama", cursor.getString(1));
hashMapBiodata.put("alamat", cursor.getString(2));
} while (cursor.moveToNext());
}
return hashMapBiodata;
}
}

```

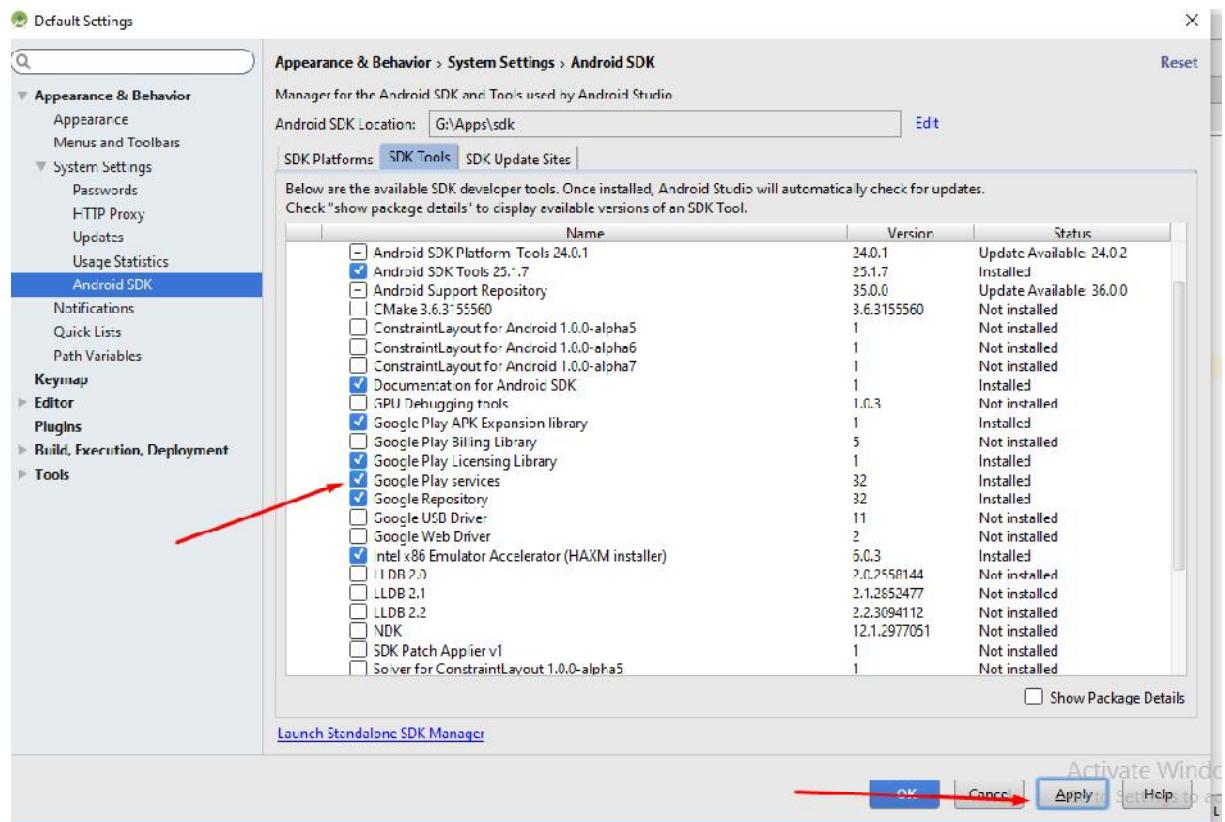
MODUL 6

API

A. Install Google Play Service SDK

The first way is by installing Google Android SDK Service Play in the Studio.

1. Go to the Menu Manager SDK
2. Select Appearance & Behavior < System Setting < Android SDK
3. on the select SDK Android SDK Tools

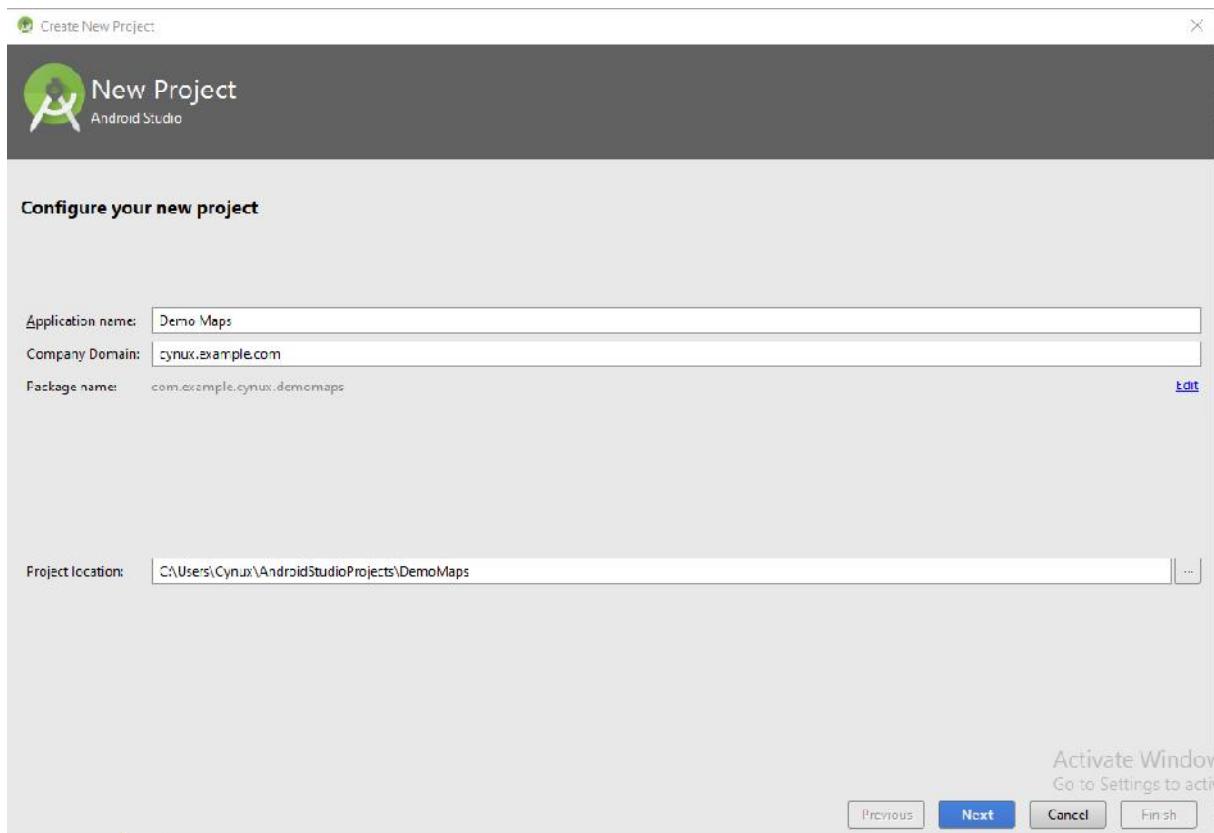


4. Search Google Play Services and checked
5. Click Apply then we will go to the Process Download Google Play Services

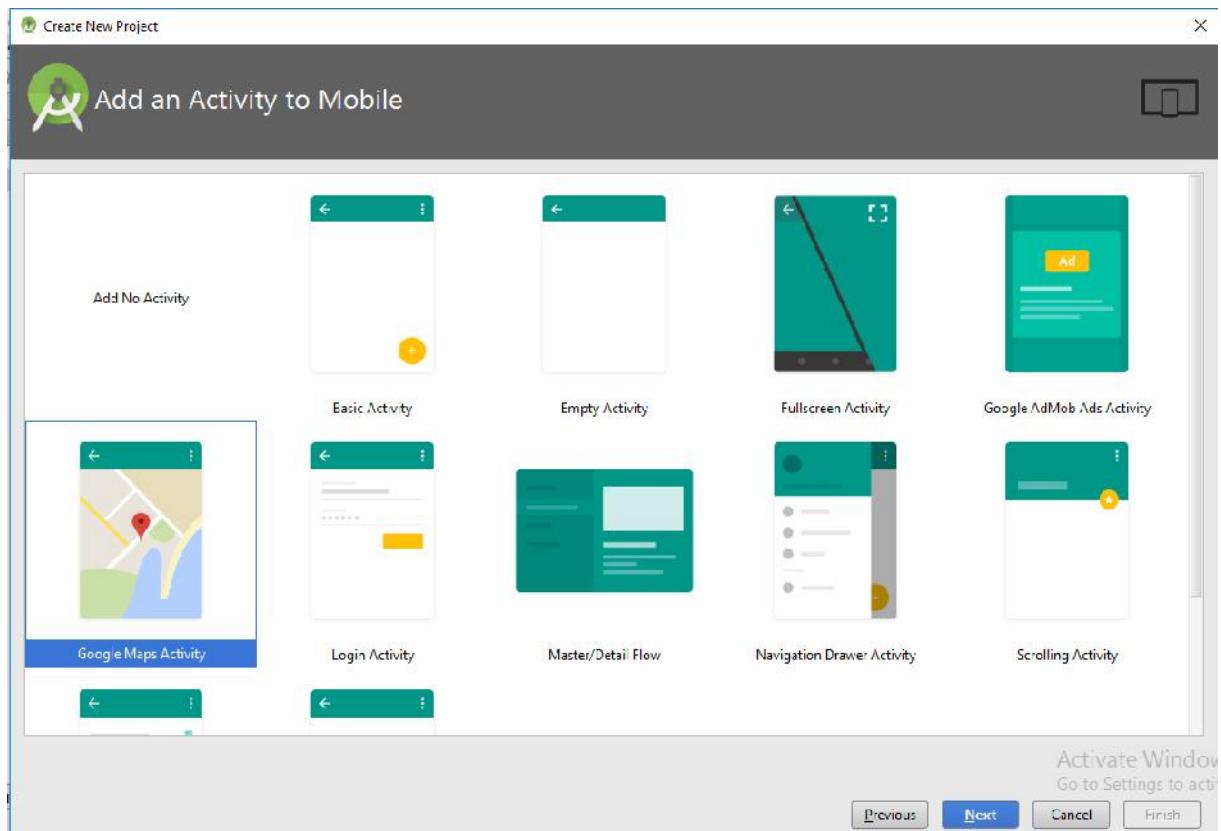
B. Create A Project Of Google Maps

The next step we will create a new application project

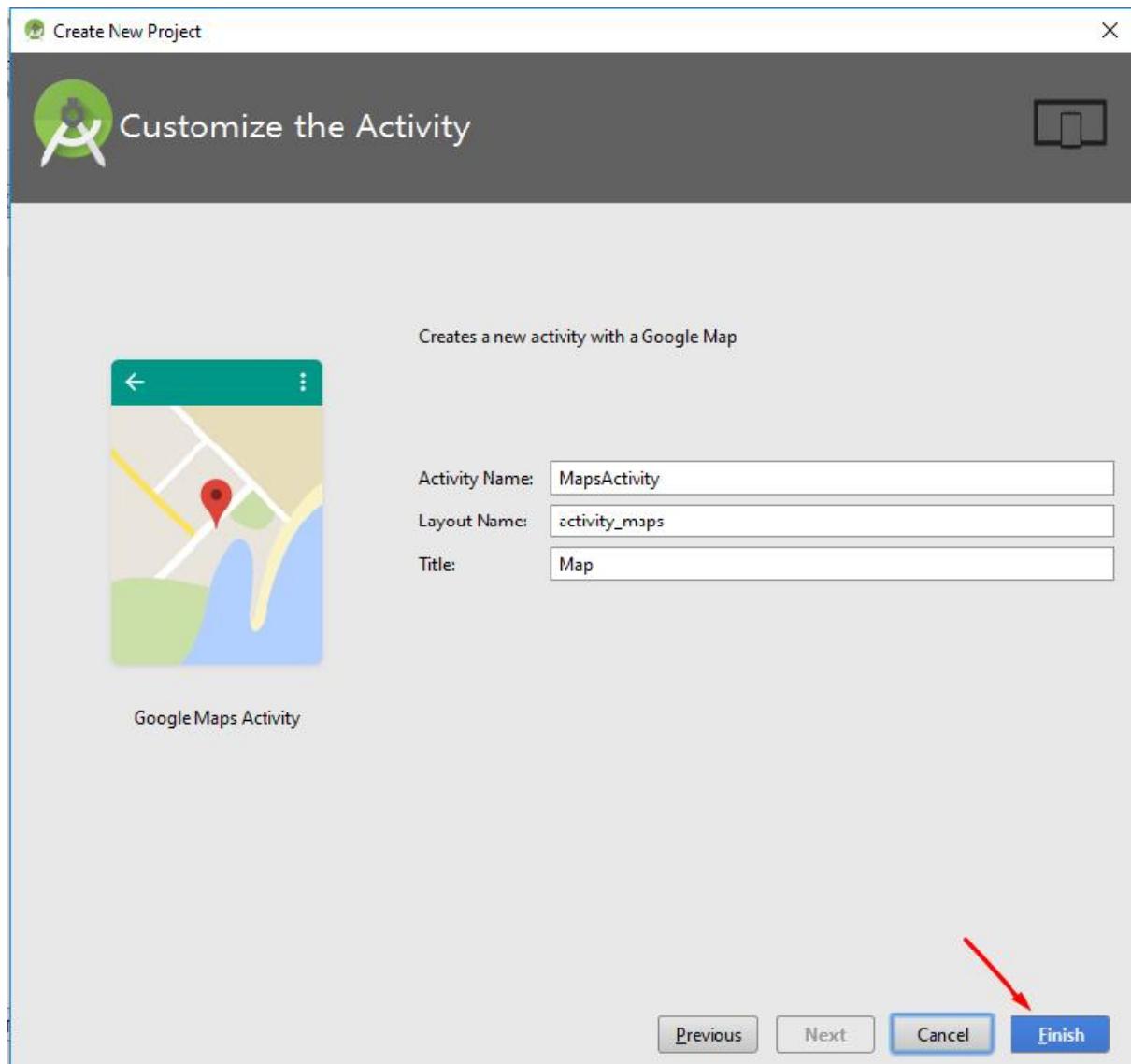
1. New Project, in the Application Name give name the fit you want. Here I will give you the name of "Demo Maps". Then Select Next



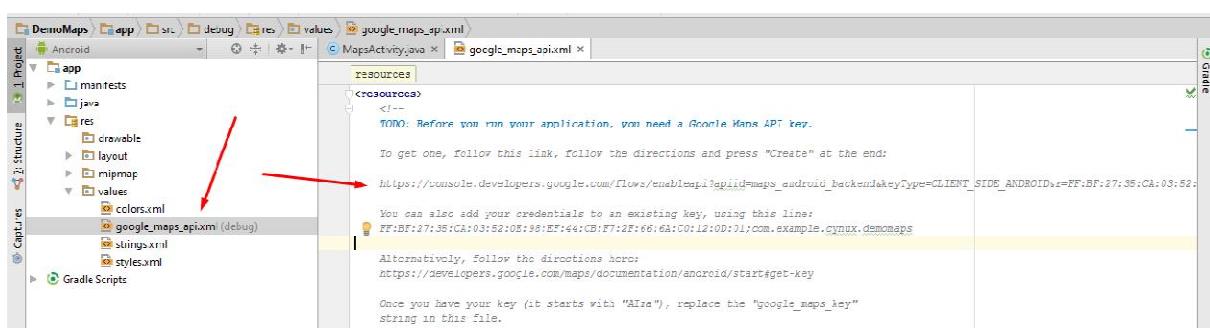
2. Next we will be faced with a screen to select Minimal Android SDK 4.2 Jelly Bean in order to can the device used to JB
3. after click Next we will choose aktivitynya. Usually we use an empty activity, so this time we'll use Google Maps Activity and click Next



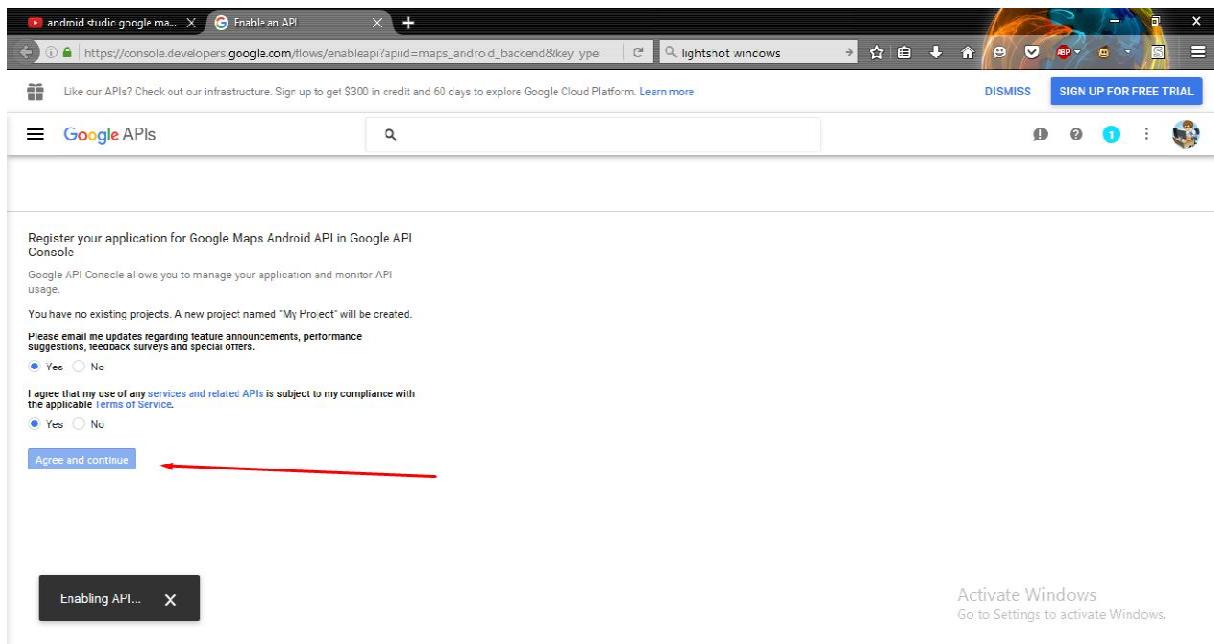
4. Click Next again, and please fill the content of the Activity name and Layout Name. I will keep using the defaults only. And the last step click Finish



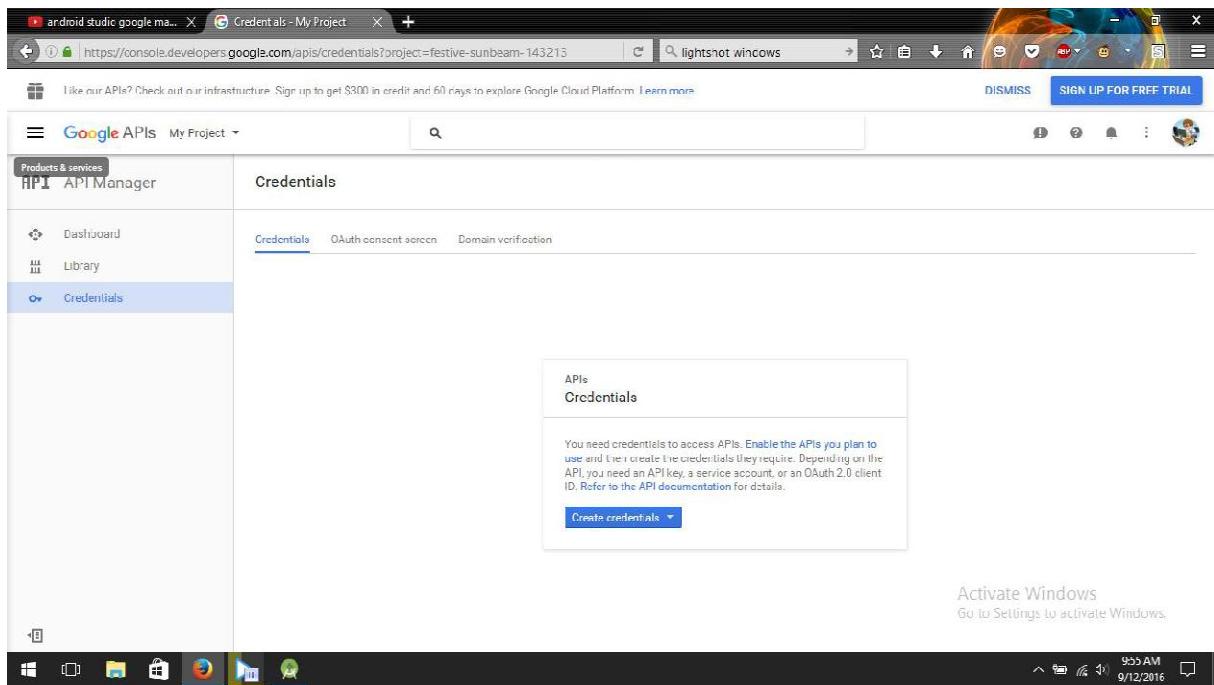
5. An then opens the window of our project. In the tree directory, select the App < res < values < chose google_maps_api.xml. Next, Copy the files on the credentials provided in the google_maps_api.xml paste in your browser.



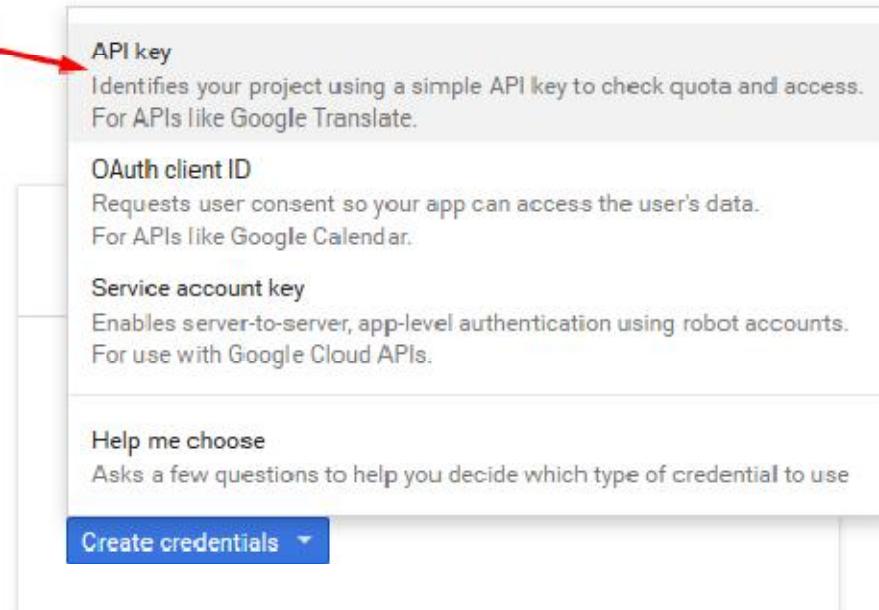
6. Next will appear as below. Simply Accept and Continue



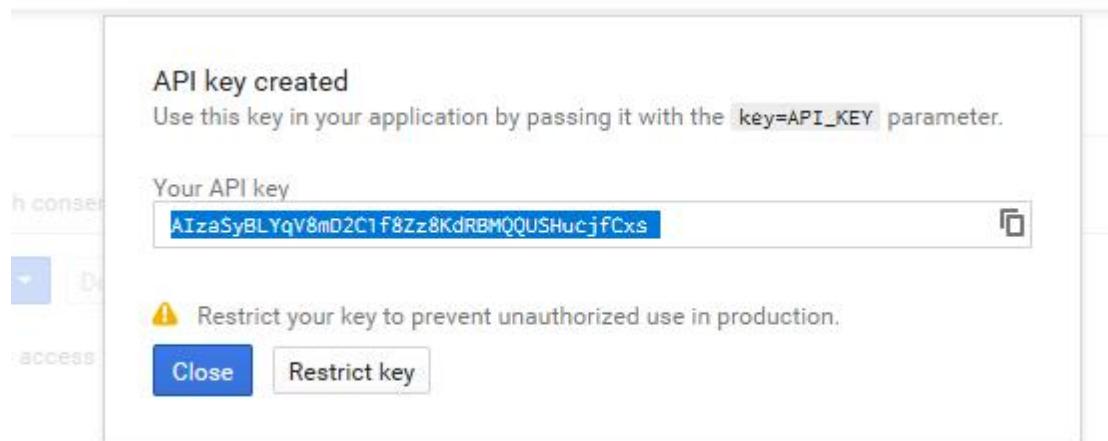
7. Select menu at the bottom left hand corner, next select credentials and then make Create Credentials



8. Then will display options like the image below, click the API Key



- Once we will get the API Key that we will be pastekan our Studio in Android



- Open again to our Android Studio. Open the google_maps_api.xml file, the pastekan file into the "Your Key Here"



After that, we can Run our App. Can using Emulator Android Studio or Genymotion or using our own mobile phones.

MODUL 7 **Database SQLite**

SQLite is a software RDBMS (Relational Database Management System) that supports natively (original) for Android devices. SQLite is a database management system, which has properties of ACID-compliant, which is programmed with C language, and have the size or the size of the memory that is relatively small. Because Sqlite database engine includes embedded (embedded), so Sqlite command that you can use only the standard commands only. And Sqlite only supports data types like INTEGER, DATETIME, NUMERIC, TEXT, and other things.

Line of source code in Sqlite also are public domain developed by d. Richard Hipp. That means you can use Sqlite freely for any purpose, whether commercial or private in nature. SQLite supports all operating system platforms, such as: Windows, Linux, Android, and iOS. Especially on the Android OS, Android apps on Sqlite will converge in a system called Android Runtime. In SQLlite built in available in the android library. So can use it, directly, without the need to wear a piece of software or importing other libraries, while developing Android applications. Can also create a Sqlite database, use the SQL user interface.

On learning android this time, will try to create an application entry for Yourself by using the Android Sqlite database. Here will also try to apply the basic functions of a database that is a CRUD (Create, Read, Update and Delete).

1. Implement The Code

- Table Structure

The structure of the table that will be created will be approximately like this:

Field	Type Data	Key
no	Integer	Primary Key
nama	Text Null	
tgl	Text Null	
jk	Text Null	
alamat	Text Null	

- SQLiteOpenHelper

SQLiteOpenHelper is a subclass, which is used to specify the database name and the version of the database that is being used. Able to apply the method in this class such as: OnCreate (SqliteDatabase), OnUpgrade (SqliteDatabase) and OnOpen (SqliteDatabase).

1. Here is created the file class Activity, named URDataHelper.java. Here will be applied to a method or function, such as: SQLiteOpenHelper from OnCreate (). who will run the database, if the database is not yet available or does not exist.

typing the line below the source code to the class file Activity named URDataHelper.java

```
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

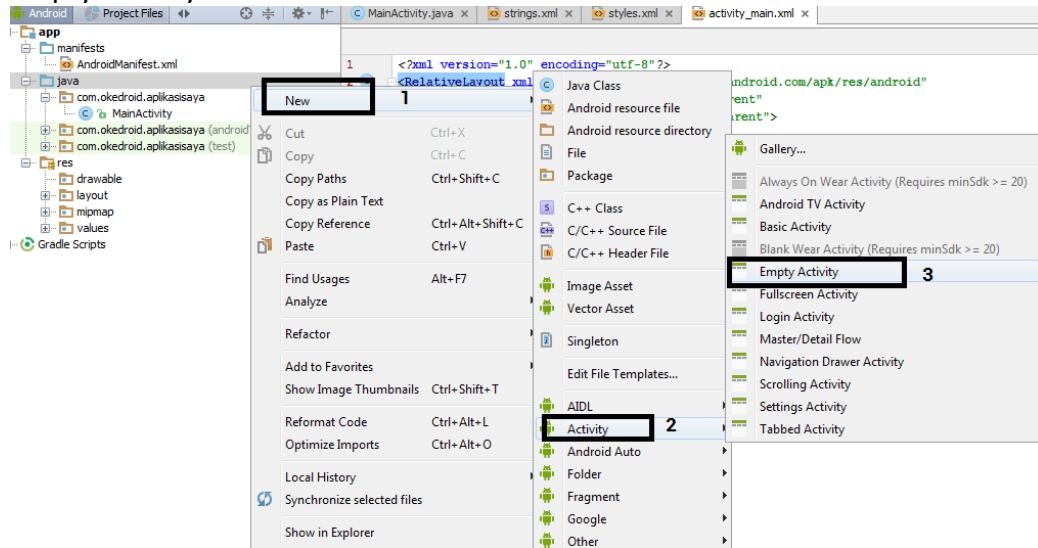
public class UpdateBiodataextends AppCompatActivity {
protected Cursor cursor;
DataHelperdbHelper;
    Button ton1, ton2;
EditTexttext1, text2, text3, text4, text5;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_update_biodata);

dbHelper= new DataHelper(this);
text1 = (EditText) findViewById(R.id.editText1);
text2 = (EditText) findViewById(R.id.editText2);
text3 = (EditText) findViewById(R.id.editText3);
text4 = (EditText) findViewById(R.id.editText4);
text5 = (EditText) findViewById(R.id.editText5);
SQLitedatabasedb = dbHelper.getReadableDatabase();
cursor = db.rawQuery("SELECT * FROM biodata WHERE nama = '" +
getIntent().getStringExtra("nama") + "'",null);
cursor.moveToFirst();
if (cursor.getCount()>0)
{
cursor.moveToPosition(0);
text1.setText(cursor.getString(0).toString());
text2.setText(cursor.getString(1).toString());
text3.setText(cursor.getString(2).toString());
text4.setText(cursor.getString(3).toString());
text5.setText(cursor.getString(4).toString());
}
ton1 = (Button) findViewById(R.id.button1);
ton2 = (Button) findViewById(R.id.button2);
// daftarkan even onClickpadabtnSimpan
ton1.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
SQLitedatabasedb = dbHelper.getWritableDatabase();
db.execSQL("update biodata set nama='"+
text2.getText().toString()+"', tgl='"+
text3.getText().toString()+"', jk='"+
text4.getText().toString()+"', alamat='"+
text5.getText().toString()+"' where no='"+
text1.getText().toString()+"'");
Toast.makeText(getApplicationContext(), "Berhasil", Toast.LENGTH_LONG).show();
MainActivity.ma.RefreshList();
finish();
}
});
ton2.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
finish();
}
});
}
@Override
public booleanonCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
```

2. After that, we'll create a New Activity

Here will be made 3 Activity is 3 file java ,3 file layout, for it will be created a new Activity in advance:

On the folder java or res, can right click Select New, select the Activity, then an Empty Activity.



New Android activity on the contents of the name of the activity or the java file and name the file layout (it's up to you guys). As an example I will create a file with the name of class activity:

Biodata.java,
ViewBiodata.java,
UpdateBiodata.java

file layout

activity_biodata.xml
activity_View_biodata.xml.
activity_update_biodata.xml.

Then you can choose the Finish button.

3. After that when the source code below in the respective file.

MainActivity.java

```
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {
    String[] daftar;
    ListView ListView01;
    Menu menu;
    protected Cursor cursor;
    DataHelper dbcenter;
    public static MainActivity ma;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button btn=(Button) findViewById(R.id.button2);

        btn.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                Intent inte = new Intent(MainActivity.this, BuatBiodata.class);
                startActivity(inte);
            }
        });

        ma = this;
        dbcenter= new DataHelper(this);
        RefreshList();
    }

    public void RefreshList(){
        SQLiteDatabase db = dbcenter.getReadableDatabase();
        cursor = db.rawQuery("SELECT * FROM biodata",null);
        daftar= new String[cursor.getCount()];
        cursor.moveToFirst();
        for (int cc=0; cc <cursor.getCount(); cc++){
            cursor.moveToPosition(cc);
            daftar[cc] = cursor.getString(1).toString();
        }
        ListView01 = (ListView) findViewById(R.id.listView);
        ListView01.setAdapter(new ArrayAdapter(this, android.R.layout.simple_list_item_1,
        daftar));
        ListView01.setSelected(true);
        ListView01.setOnItemClickListener(new OnItemClickListener() {

            public void onItemClick(AdapterView arg0, View arg1, int arg2, long arg3) {
                final String selection = daftar[arg2]; //.getItemAtPosition(arg2).toString();
                final CharSequence[] dialogitem = {"View Biodata", "Update Biodata", "Delete Biodata"};
                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
                builder.setTitle("Option");
                builder.setItems(dialogitem, new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int item) {
                        switch(item) {
                            case 0 :
```

```

        Intent i = new Intent(getApplicationContext(),
LihatBiodata.class);
i.putExtra("nama", selection);
startActivity(i);
break;
case 1 :
        Intent in = new Intent(getApplicationContext(),
UpdateBiodata.class);
in.putExtra("nama", selection);
startActivity(in);
break;
case 2 :
SQLiteDatabase db = dbcenter.getWritableDatabase();
db.execSQL("delete from biodata where nama = '"+selection+"'");
RefreshList();
break;
}
}
builder.create().show();
});
((ArrayAdapter)ListView01.getAdapter()).notifyDataSetChanged();
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
}

```

```

activity_main.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:background="#ecf0f1"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_toRightOf="@+id/button1"
        style="?android:attr/borderlessButtonStyle"
        android:drawableLeft="@drawable/icon_add"
        android:text="Buat Biodata Baru" />

    <ListView
        android:id="@+id/listView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button2" android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true">
    </ListView>

</RelativeLayout>

```

SQLiteDatabase is a basic working class, for database sqlite in Android devices. SQLiteDatabase will run a command SQL directly with the method execSQL (). And will also perform other database management in General, the methods used, such as: (a) Insert, Update() and Delete ().

Biodata.java

```
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class Biodata extends AppCompatActivity {
protected Cursor cursor;
DataHelper dbHelper;
Button ton1, ton2;
EditText text1, text2, text3, text4, text5;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_buat_biodata);

dbHelper= new DataHelper(this);
text1 = (EditText) findViewById(R.id.editText1);
text2 = (EditText) findViewById(R.id.editText2);
text3 = (EditText) findViewById(R.id.editText3);
text4 = (EditText) findViewById(R.id.editText4);
text5 = (EditText) findViewById(R.id.editText5);
ton1 = (Button) findViewById(R.id.button1);
ton2 = (Button) findViewById(R.id.button2);

ton1.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
SQLiteDatabase db = dbHelper.getWritableDatabase();
db.execSQL("insert into biodata(no, nama, tgl, jk, alamat) values('" +
text1.getText().toString() + "','" +
text2.getText().toString() + "','" +
text3.getText().toString() + "','" +
text4.getText().toString() + "','" +
text5.getText().toString() + "');");
Toast.makeText(getApplicationContext(), "success", Toast.LENGTH_LONG).show();
MainActivity.ma.RefreshList();
finish();
}
});
ton2.setOnClickListener(new View.OnClickListener() {

@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
finish();
}
});

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
```

activity_biodata.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
```

```
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context=".BuatBiodata" >

    <EditText
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView1"
        android:layout_below="@+id/textView1" >
        <requestFocus/>
    </EditText>

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:text="Nomor" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText1"
        android:layout_below="@+id/editText1"
        android:layout_marginTop="10dp"
        android:text="Nama" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView2"
        android:layout_below="@+id/textView2" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText2"
        android:layout_below="@+id/editText2"
        android:layout_marginTop="10dp"
        android:text="TanggalLahir" />

    <EditText
        android:id="@+id/editText3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView3"
        android:layout_below="@+id/textView3" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText3"
        android:layout_below="@+id/editText3"
        android:layout_marginTop="10dp"
        android:text="JenisKelamin" />

    <EditText
        android:id="@+id/editText4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView4"
        android:layout_below="@+id/textView4" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText4"
        android:layout_below="@+id/editText4"
        android:layout_marginTop="10dp"
        android:text="Alamat" />
```

```

<EditText
    android:id="@+id/editText5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView5"
    android:layout_below="@+id/textView5" />

<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText5"
    android:layout_alignParentBottom="true"
    android:text="Simpan" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/button1"
    android:layout_alignBottom="@+id/button1"
    android:layout_toRightOf="@+id/textView4"
    android:text="Back" />

</RelativeLayout>

ViewBiodata.java
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class LihatBiodata extends AppCompatActivity {
protected Cursor cursor;
DataHelper dbHelper;
Button ton2;
TextView text1, text2, text3, text4, text5;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_lihat_biodata);

dbHelper= new DataHelper(this);
text1 = (TextView) findViewById(R.id.textView1);
text2 = (TextView) findViewById(R.id.textView2);
text3 = (TextView) findViewById(R.id.textView3);
text4 = (TextView) findViewById(R.id.textView4);
text5 = (TextView) findViewById(R.id.textView5);
SQLiteDatabase db = dbHelper.getReadableDatabase();
cursor = db.rawQuery("SELECT * FROM biodata WHERE nama = '" +
getIntent().getStringExtra("nama") + "'",null);
cursor.moveToFirst();
if (cursor.getCount()>0)
{
cursor.moveToPosition(0);
text1.setText(cursor.getString(0).toString());
text2.setText(cursor.getString(1).toString());
text3.setText(cursor.getString(2).toString());
text4.setText(cursor.getString(3).toString());
text5.setText(cursor.getString(4).toString());
}
ton2 = (Button) findViewById(R.id.button1);
ton2.setOnClickListener(new View.OnClickListener() {

@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
finish();
}
});
}

```

```

        }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}

activity_lihat_biodata.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".LihatBiodata" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_marginRight="104dp"
        android:layout_marginTop="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/textView1"
        android:layout_below="@+id/textView1"
        android:layout_marginTop="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView2"
        android:layout_below="@+id/textView2"
        android:layout_marginTop="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView3"
        android:layout_below="@+id/textView3"
        android:layout_marginTop="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/textView4"
        android:layout_below="@+id/textView4"
        android:layout_marginTop="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/TextView05"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView5"
        android:layout_alignBottom="@+id/textView5"
        android:layout_alignLeft="@+id/TextView03"
        android:text="Alamat" />

```

```

<TextView
    android:id="@+id/TextView03"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/textView4"
    android:layout_alignBottom="@+id/textView4"
    android:layout_alignLeft="@+id/TextView04"
    android:text="JenisKelamin" />

<TextView
    android:id="@+id/TextView04"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/textView4"
    android:layout_alignBottom="@+id/textView3"
    android:layout_alignLeft="@+id/TextView02"
    android:text="TanggalLahir" />

<TextView
    android:id="@+id/TextView02"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/textView2"
    android:layout_alignBottom="@+id/textView2"
    android:layout_alignLeft="@+id/TextView01"
    android:text="Nama" />

<TextView
    android:id="@+id/TextView01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/textView2"
    android:layout_alignParentLeft="true"
    android:text="Nomor" />

<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/TextView05"
    android:layout_below="@+id/TextView05"
    android:layout_marginTop="34dp"
    android:text="Back" />

</RelativeLayout>

UpdateBiodata.java
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class UpdateBiodata extends AppCompatActivity {
protected Cursor cursor;
DataHelper dbHelper;
Button ton1, ton2;
EditText text1, text2, text3, text4, text5;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_update_biodata);

dbHelper= new DataHelper(this);
text1 = (EditText) findViewById(R.id.editText1);
text2 = (EditText) findViewById(R.id.editText2);
text3 = (EditText) findViewById(R.id.editText3);
text4 = (EditText) findViewById(R.id.editText4);
text5 = (EditText) findViewById(R.id.editText5);
SQLiteDatabase db = dbHelper.getReadableDatabase();
}

```

```

cursor = db.rawQuery("SELECT * FROM biodata WHERE nama = '" +
getIntent().getStringExtra("nama") + "'",null);
cursor.moveToFirst();
if (cursor.getCount()>0)
{
    cursor.moveToPosition(0);
    text1.setText(cursor.getString(0).toString());
    text2.setText(cursor.getString(1).toString());
    text3.setText(cursor.getString(2).toString());
    text4.setText(cursor.getString(3).toString());
    text5.setText(cursor.getString(4).toString());
}
ton1 = (Button) findViewById(R.id.button1);
ton2 = (Button) findViewById(R.id.button2);
// daftarkan even onClickpadabtnSimpan
ton1.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
SQLitedatabasedb = dbHelper.getWritableDatabase();
db.execSQL("update biodata set nama='"+
text2.getText().toString() +', tgl=' +
text3.getText().toString()+' , jk='+
text4.getText().toString() +' , alamat=' +
text5.getText().toString() +' where no=' +
text1.getText().toString()+'");
Toast.makeText(getApplicationContext(), "Berhasil", Toast.LENGTH_LONG).show();
MainActivity.ma.RefreshList();
finish();
}
});
ton2.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
finish();
}
});
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
}

```

```

activity_update_biodata.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

```

```
tools:context=".UpdateBiodata" >

<EditText
    android:id="@+id/editText1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView1"
    android:layout_below="@+id/textView1" >

    <requestFocus/>
</EditText>

<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="Nomor" />

<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText1"
    android:layout_below="@+id/editText1"
    android:layout_marginTop="10dp"
    android:text="Nama" />

<EditText
    android:id="@+id/editText2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView2"
    android:layout_below="@+id/textView2" />

<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText2"
    android:layout_below="@+id/editText2"
    android:layout_marginTop="10dp"
    android:text="Tanggal Lahir" />

<EditText
    android:id="@+id/editText3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView3"
    android:layout_below="@+id/textView3" />

<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText3"
    android:layout_below="@+id/editText3"
    android:layout_marginTop="10dp"
    android:text="Jenis Kelamin" />

<EditText
    android:id="@+id/editText4"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView4"
    android:layout_below="@+id/textView4" />

<TextView
    android:id="@+id/textView5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText4"
    android:layout_below="@+id/editText4"
    android:layout_marginTop="10dp"
    android:text="Alamat" />
```

```

<EditText
    android:id="@+id/editText5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/textView5"
    android:layout_below="@+id/textView5" />

<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText5"
    android:layout_alignParentBottom="true"
    android:text="Update" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/button1"
    android:layout_alignBottom="@+id/button1"
    android:layout_toRightOf="@+id/textView4"
    android:text="Back" />

</RelativeLayout>

```

3. Running The Project Application

After completion of all, implementation of the code, can now try to run the application in the Android's Studio.

The result looks like this:



Figure 7.1 Result Display apps

If the user (users) suppressing the + Create a new Entry, it will appear as shown below:

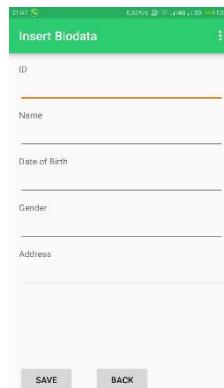


Figure 7.2 Result Display Input Form

If the user (users) suppressing one of the name, it will pop up the options in the form of a Context Menu, which consists of whether the user (users) want to view, update, or remove personal information. As in the picture below:

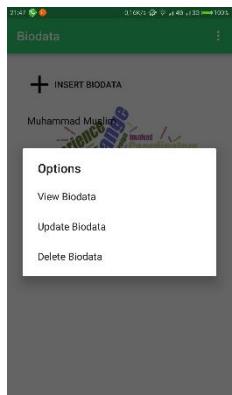


Figure 7.3 Result Display Options

MODUL 8

Database Real Time Using Firebase

Firebase is a cloud service provider's Backend and also as a Service (BaaS), a company based in San Francisco, California. The company makes a product that helps the Software Developer in building Mobile or Web Applications. The founders of this company were Andrew Lee and James Tamplin. Firebase main products have a Realtime Database that is available in the form of API, which lets Developers in storing and synchronizing data across multiple Client. Now the company has in its acquisition by Google in October 2014. Features that have now been developed and in snap in tools. This is Firebase tools which help the work of an Android developer, in developing Android applications, in which case the backend services. Firebase is free, and you can add it on Android applications you guys, on a tutorial that has been provided by the Team's Firebase (read: how to Setup). Firebase also available not only for Android, but it could also be for iOS and Web Applications.



Figure 8.1 Logo Firebase

There are many features in the Firebase in the Tools help in building the backend service, as well as help in terms of the analysis of the target user (users) to monetize(Admob). and also build up a user base in terms of promoting applications (Adwords).

Here are a few features that are in the Firebase:

Develop :

- Cloud Messaging
- Authentication
- Realtime Database
- Storage
- Hosting
- Test Lab
- Crash Reporting

Grow :

- Notification
- Remote Config
- App Indexing
- Dynamic Links
- Invites
- Adwords

Earn:

- Admob

More information if you would like to see the features on offer at Firebase:

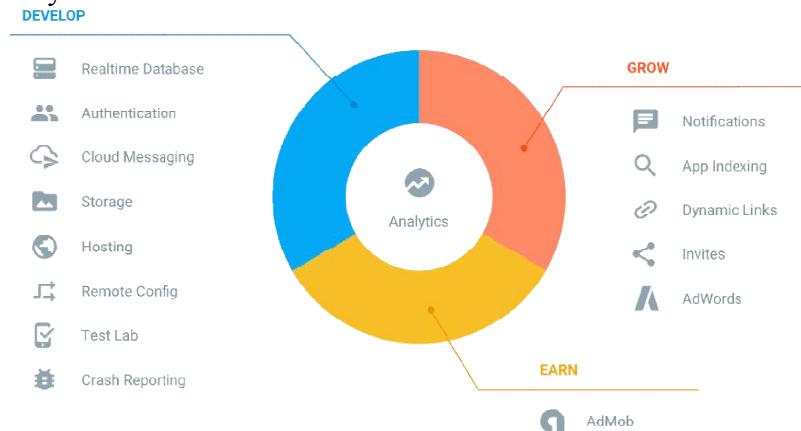


Figure 8.2 Some of the features that are on Firebase (by Google Firebase)

If you want to use the latest version of Android's Studio. Can see these Tools on the Main Menu.

Main Menu: Tools > Firebase

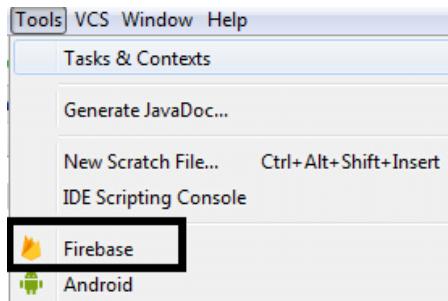


Figure 8.3 Tools Firebase Android Studio

Then on the Tabs right Assistants, can be seen the features provided.

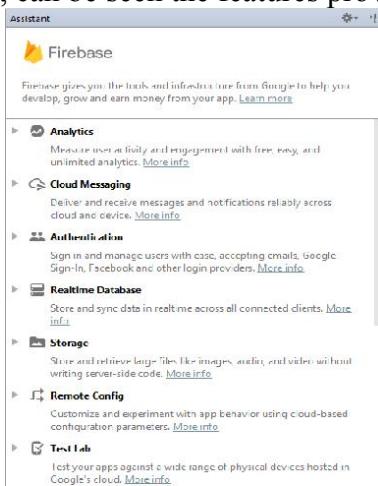


Figure 8.4 Tab Assistants

Well now you already know right what is a Firebase, this all depends on the needs of the application are you guys develop. If the application wants to require a Backend Service as a Service (BaaS) and also data analysis users (users). could use the Tools Firebase.

Link :

<https://firebase.google.com/features/>

MODUL 9

FIREBASE AUTHENTICATION

Let's dive right into creating an android application to learn more about Firebase authentication. The steps are described in 6 detailed yet simple parts :

Note : Make sure you have the latest version of Google Play Services and Google repository installed in your Android SDK manager as shown below.

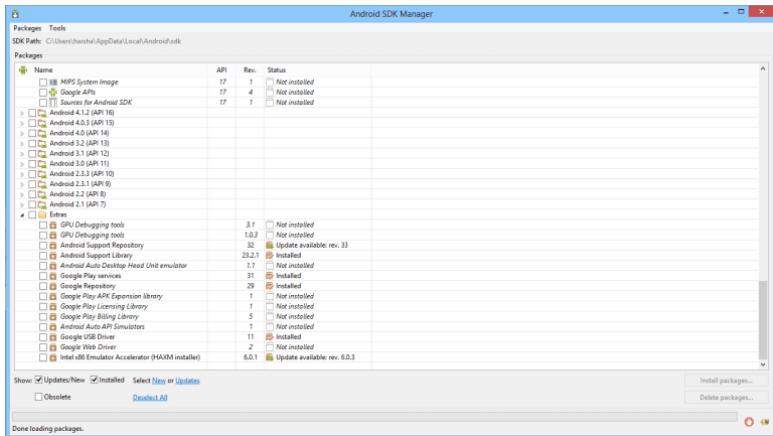


Figure 9.1 Android SDK

Part I : Includes firebase account creation and getting your firebase reference

1. The first thing you need to do to get started with Firebase is sign up for a free account. You can use your Google account to sign in to your firebase account.

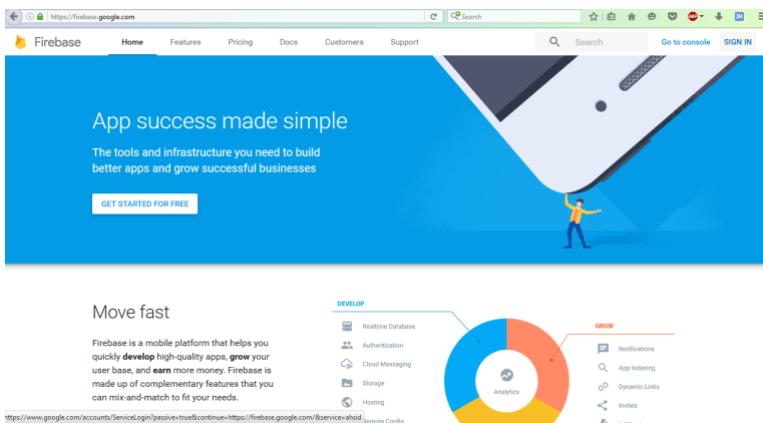


Figure 9.2 Create Account Firebase

2. After clicking on ‘Sign In with Google’ you will be asked to allow the Firebase to view your Email and your basic profile. Click on Allow button to continue.

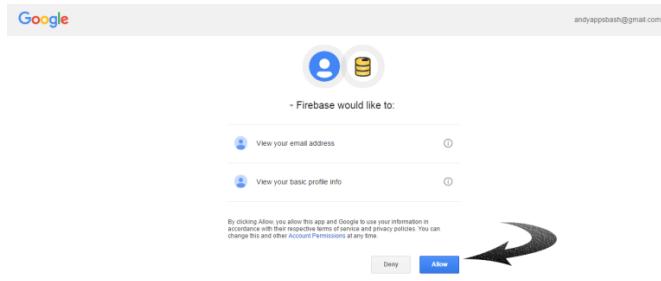


Figure 9.3 Sign In

3. Once you are signed in Click on **Go to your Console** if you are not automatically redirected to your firebase console. Click on **Create New Project** button in your firebase console.

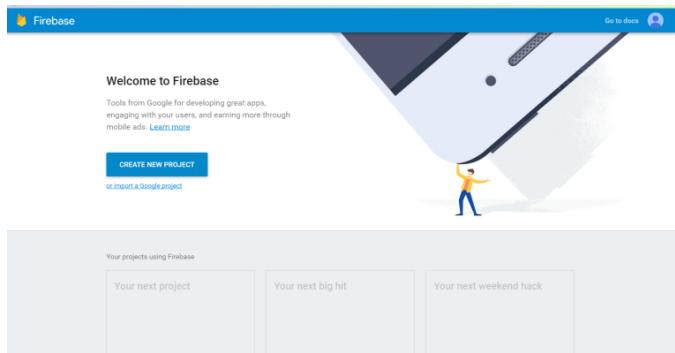


Figure 9.4 Create New Project

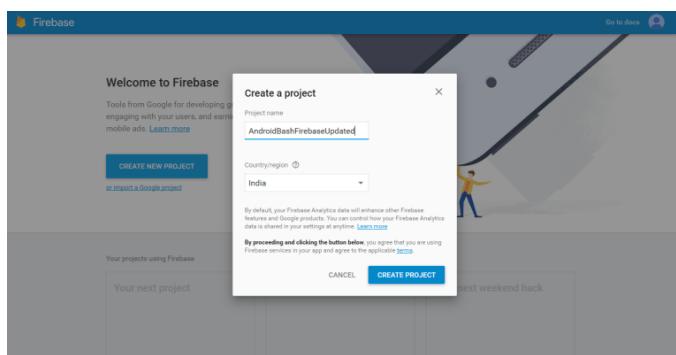


Figure 9.5 Add Project Package

4. Now you can see an overview of your new Firebase project. Now click on **Add Firebase to your Android App** as shown below.

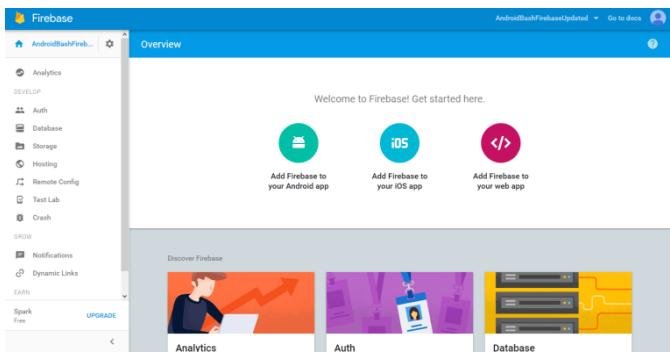


Figure 9.6 Get SHA-1

5. Now you need to enter your App details and also get SHA-1 finger print and paste it in the field as shown below.

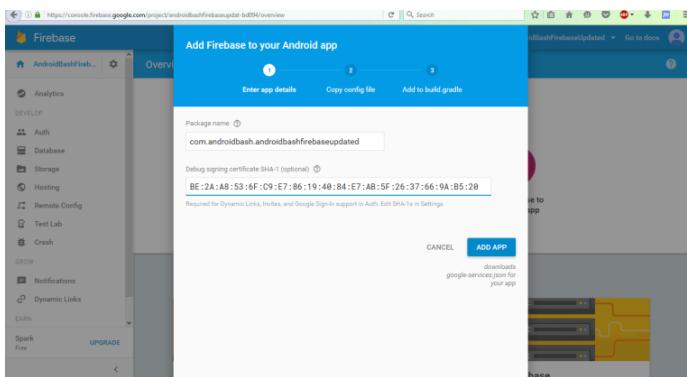


Figure 9.7 Add Key SHA-1

Click on **Add App** button. Upon clicking it, a file named google-services.json gets downloaded to your machine.

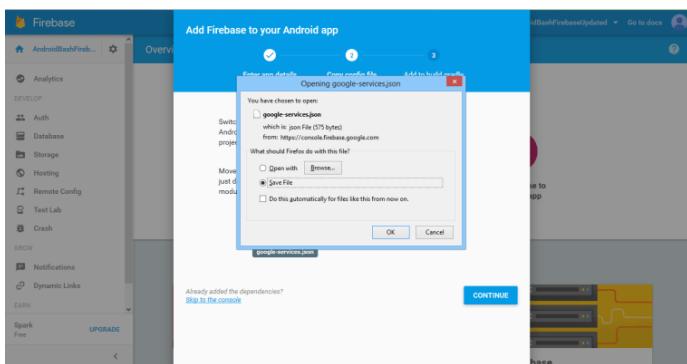


Figure 9.8 Download Json

6. Copy the google-services.json file and paste in the **app** folder of your Android application.

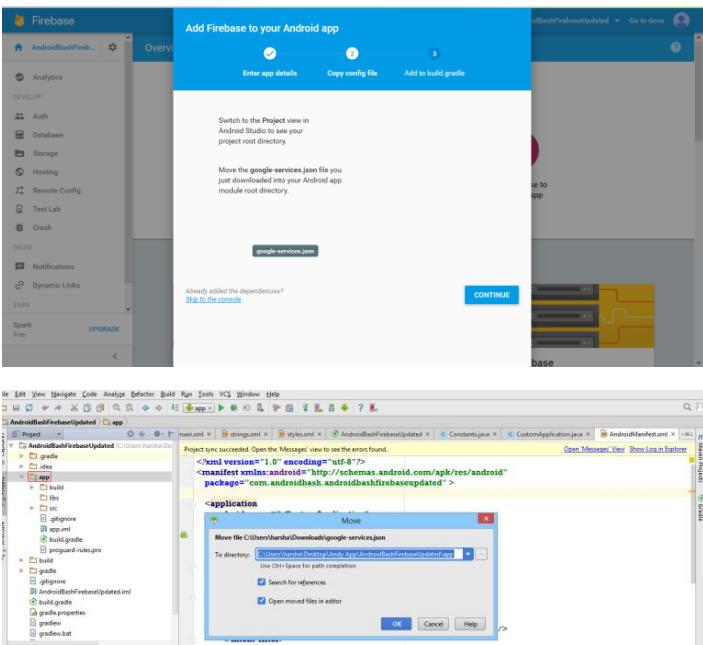


Figure 9.9 Include File Json To Project

Note: If you forget to copy and paste google-services.json file you will get the following error when you build your app.



7. Open the **project level build.gradle** file and add **classpath 'com.google.gms:google-services:3.0.0'** in the dependencies section. Your **project level build.gradle** file will look like below.

```
buildscript {
    repositories {
        jcenter()
        mavenLocal()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:1.3.0'
        classpath 'com.google.gms:google-services:3.0.0'
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
allprojects {
    repositories {
        jcenter()
        maven { url 'https://maven.fabric.io/public' }
        mavenLocal()
    }
}
```

8. Now add **apply plugin: 'com.google.gms.google-services'** at the bottom of your **App level build.gradle** file below dependencies section. Click on sync now or re-build the android application.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.1.1'
```

```

        compile 'com.android.support:design:23.1.1'
        compile 'com.google.firebaseio:firebase-auth:9.2.0'
        compile 'com.firebaseio.firebaseio-client-android:2.5.2'
        compile 'com.facebook.android:facebook-android-sdk:4.9.0'
        compile 'com.google.android.gms:play-services-auth:9.2.0'
    }
apply plugin: 'com.google.gms.google-services'

```

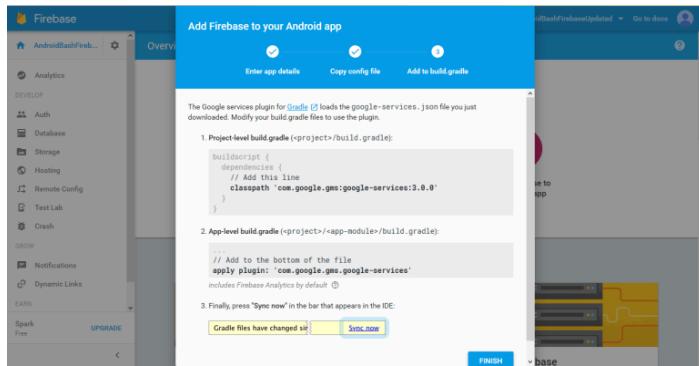


Figure 9.10 Firebase Plugin

- The following section in the Firebase console shows the overview of your project. Click on **Database** option in the left navigation menu.

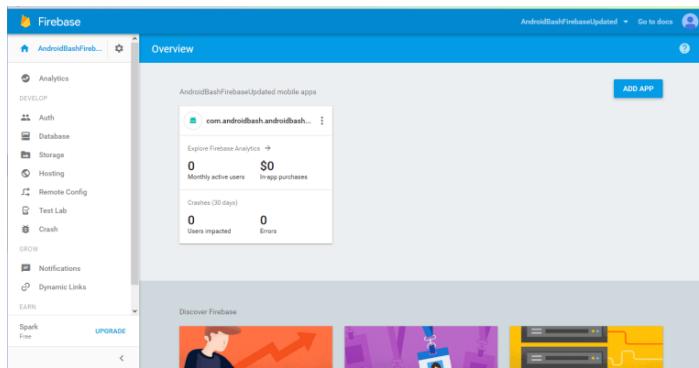


Figure 9.11 Firebase Project Console

- Initially as there will be no users, the database will be empty and it looks as shown in the following image.

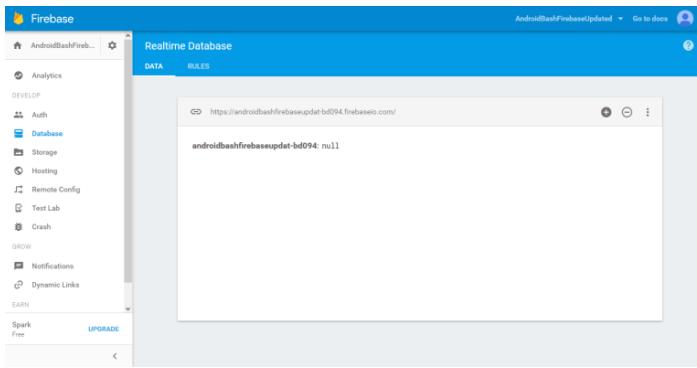


Figure 9.12 Firebase Database Console

11. Go to **Rules** tab of **Database** section and modify the rules as shown below. It gives read and write permissions to a specific node (`users->uid`) of your database.

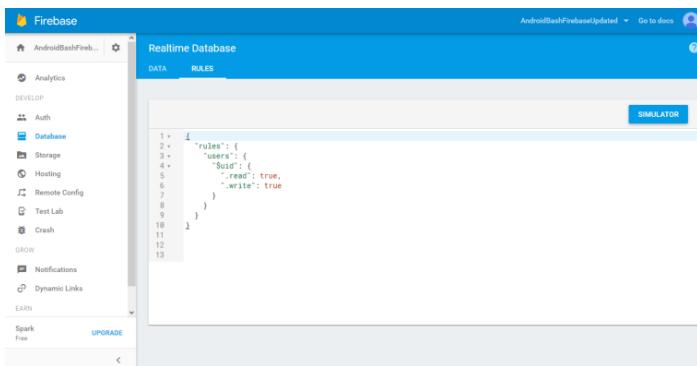


Figure 9.13 Firebase Rules

12. Now as we need to implement login functionality in our android application we need to Enable **Email and Password authentication** under Authentication Tab of your Firebase console. Also you should Enable **Facebook Login** under Authentication Tab of your Firebase console. The steps to enable Facebook sign-in method is described in the Part III of this tutorial. You can re-visit the Authentication tab later to enable Facebook Login.

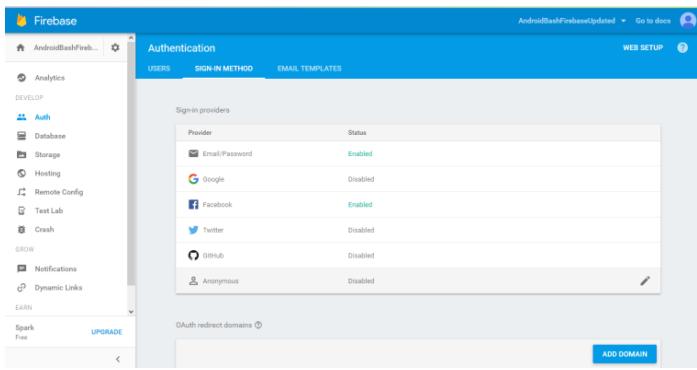


Figure 9.14 Firebase Authentication

Now we are done with the First part and let's move on to the next part.

Part II : Includes steps to create a new Android application and set up Firebase in it.

1. Create a new Android Project using Android Studio. Give it a name and select the Minimum SDK on which your app will run on. I chose API 16 : Android 4.1 (JELLY_BEAN).

2.. When you are prompted to add an activity to your application choose Blank Activity and click on next button.

3. In the next step click on Finish and in few seconds your application should be loaded in Android Studio.

4. Open **build.gradle(Module:App)** file of your application and add compile 'com.firebaseio.firebaseio-client-android:2.5.2' and compile 'com.google.firebaseio.firebaseio-auth:9.2.0' inside dependencies section of build.gradle file.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:design:23.1.1'
    compile 'com.google.firebaseio.firebaseio-auth:9.2.0'
    compile 'com.firebaseio.firebaseio-client-android:2.5.2'
    compile 'com.facebook.android:facebook-android-sdk:4.9.0'
    compile 'com.google.android.gms:play-services-auth:9.2.0'
}apply plugin: 'com.google.gms.google-services'
```

5. Now add the packagingOptions directive to your **build.gradle(Module : App)** file as shown below. This is how your **build.gradle** (Module : App) will look.

```
apply plugin: 'com.android.application'
android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"
    defaultConfig {
        applicationId "com.androidbash.androidbashfirebaseupdated"
        minSdkVersion 16
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
            'proguard-rules.pro'
        }
    }
    packagingOptions {
        exclude 'META-INF/LICENSE'
        exclude 'META-INF/LICENSE-FIREBASE.txt'
        exclude 'META-INF/NOTICE'
    }
}
repositories {
    mavenCentral()
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:design:23.1.1'
```

```

compile 'com.google.firebaseio:firebase-auth:9.2.0'
compile 'com.firebaseio.firebaseio-client-android:2.5.2'
compile 'com.facebook.android:facebook-android-sdk:4.9.0'
compile 'com.google.android.gms:play-services-auth:9.2.0'
}
apply plugin: 'com.google.gms.google-services'

```

6. The Firebase library requires the android.permission.INTERNET permission to operate. Your app will not work unless you add this permission to your AndroidManifest.xml file:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Now the Second Part is done and we can move on to the next part.

Part III : Includes Facebook App Creation and getting your Facebook App Id

1. To get started with Facebook authentication, you need to first visit [Facebook developers page](#) and create a new Facebook application. Click the **Add a New App** button in the top right menu of that page and select Android as your platform.

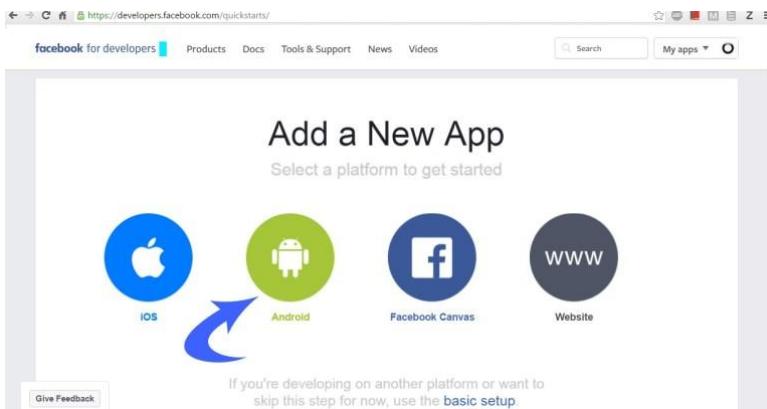


Figure 9.15 Add Android Apps

2. Then choose an App ID (Any name) and click **Create New Facebook App ID** button. Select your app's category and click **Create App ID** button.

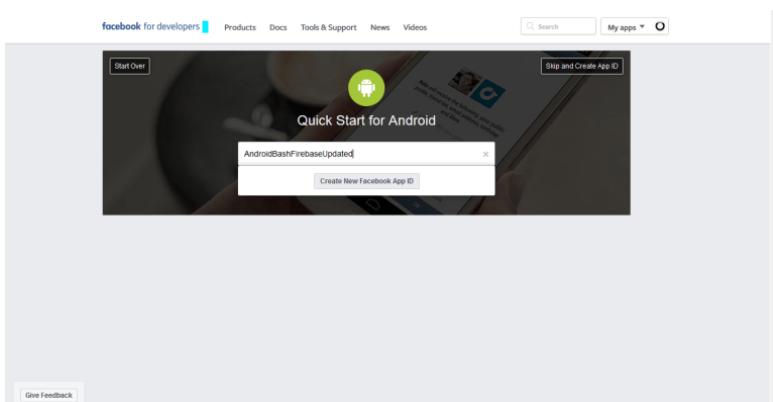


Figure 9.16 Create App ID

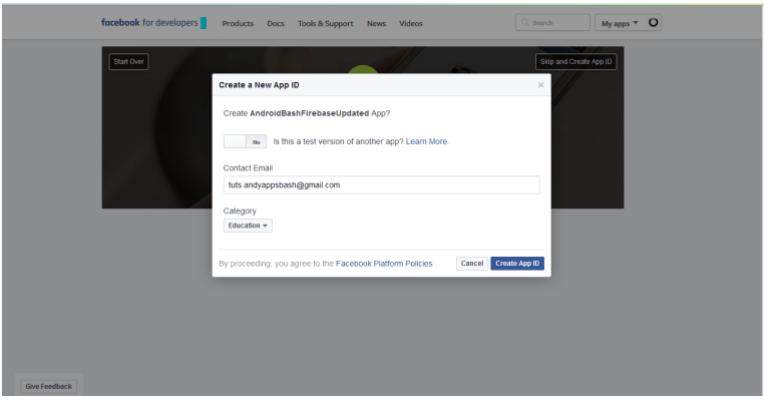


Figure 9.17 Add App ID

3. Now in your Android application project you need to open the **build.gradle** file (Module : app) and add the Maven Central Repository just before dependencies section.

```
repositories {
    mavenCentral()
}
```

4. Add **compile 'com.facebook.android:facebook-android-sdk:4.9.0'** to the dependencies section of your **build.gradle** file (Module : app) and Build your project.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.3.0'
    compile 'com.android.support:design:23.3.0'
    compile 'com.facebook.android:facebook-android-sdk:4.9.0'
}
```

5. Now open **strings.xml** file of your Firebase Android application project and add a new string with the name **facebook_app_id** containing the value of your Facebook App Id. You will get this Id in the **Add SDK** section of quickstart page which is shown as soon as you create a new Facebook App.

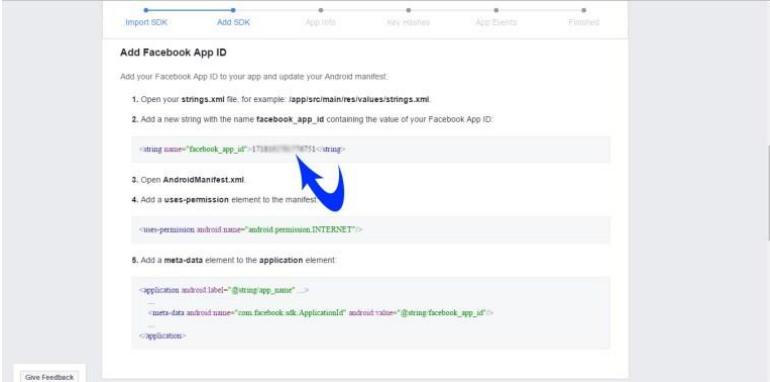


Figure 9.18 Add SDK

6. Now in the **AndroidManifest.xml** of your Android application project you will have to add Internet Permission and also add the following code under **application tag**.

```
<meta-data android:name="com.facebook.sdk.ApplicationId"
    android:value="@string/facebook_app_id"/>

<activity android:name="com.facebook.FacebookActivity"
```

```

        android:configChanges="keyboard|keyboardHidden|screenLayout|screen
        size|orientation"
        android:theme="@android:style/Theme.Translucent.NoTitleBar"
        android:label="@string/app_name" />

```

- 7.** In the **App Info ->Tell us about your Android project** section of the quickstart page fill out the Package Name and Default Activity Class Name of your Firebase Android application as shown below. Click on Next and Click on Use this package name if you are prompted.

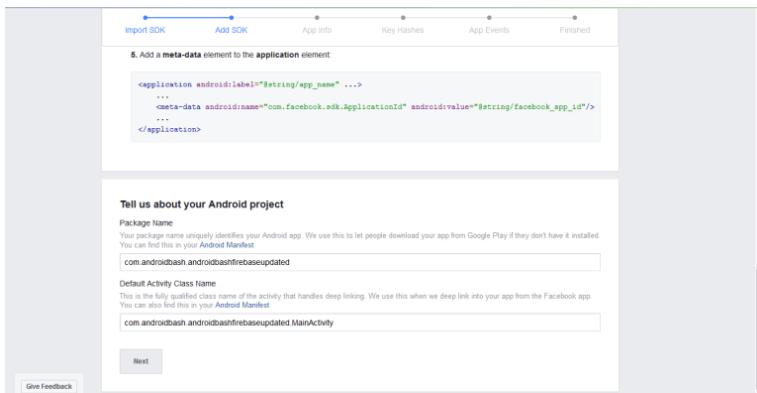


Figure 9.19 App Info

- 8.** Now we have to add the **Development Key Hashes** for our Android application. When your app is ready to be published, it is important to add your Release Key Hashes. For now, let us just add the Development Key Hashes. To get this Development Key Hash we need to install openssl in our system. If you don't have openssl, install it from here: [Windows](#), [MAC](#).

Download the zip and extract the contents to “C:\Users\YOURUSERNAME\” folder. Now go to Command Prompt in your system. To generate a development key hash,

On Mac, run the following command:

```
keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore |  
openssl sha1 -binary | openssl base64
```

On Windows, run this command:

```
keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore |  
"C:\Users\yourusername\openssl\bin\openssl.exe" sha1 -binary |  
"C:\Users\yourusername\openssl\bin\openssl.exe" base64
```

Note : In the command modify the path to openssl.exe if required.

When prompted for a password you need to type “android” as the password. This command will generate a 28-character key hash unique to your development environment. Copy and paste it into the field as shown below and click on Next button.

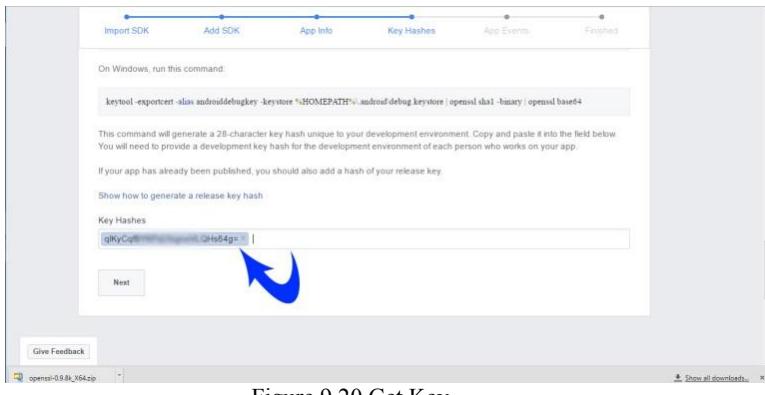


Figure 9.20 Get Key

9. Now click on the link to **Skip to Developer Dashboard** at the end of the quickstart page.

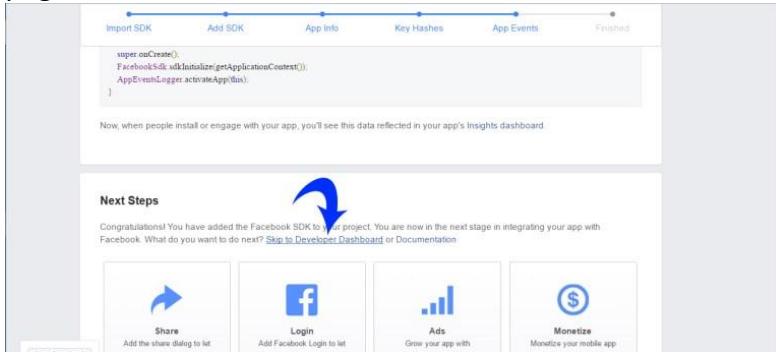


Figure 9.21 Developer Dashboard

10. Now in the **Facebook Developer Dashboard** you will get to know the **App ID** and **App Secret** as shown below.

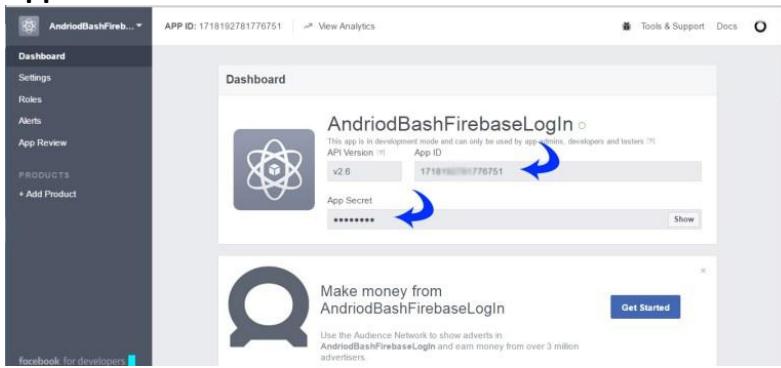


Figure 9.22 Facebook Developer Dashboard

11. Copy and Paste both of them in the **Auth->Sign-In methods->Facebook** tab of your **Firebase console** respectively for App ID and App Secret fields as shown below. Before clicking Save button, copy the OAuth redirect URI.

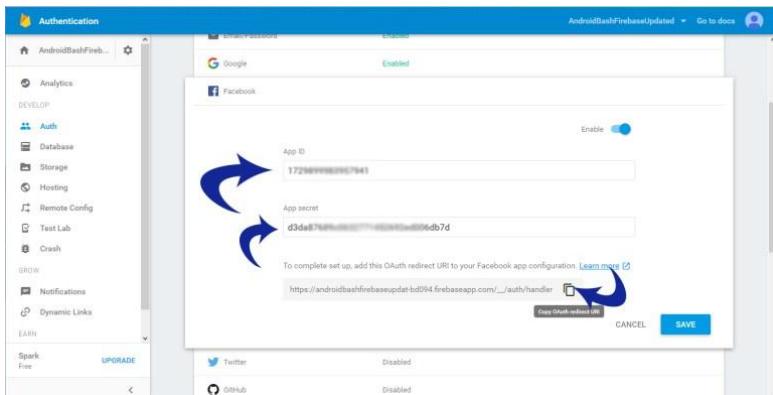


Figure 9.23 Auth Firebase Console

12. You need to paste this OAuth redirect URI in the Products section of your Facebook developers dashboard. Go to facebook developers dashboard and Click on **Add Products** and select Facebook Login. Once you add Facebook Login under Products you can paste the OAuth redirect URI in the **Valid OAuth redirect URIs** section of Facebook Login Tab as shown in the image below.

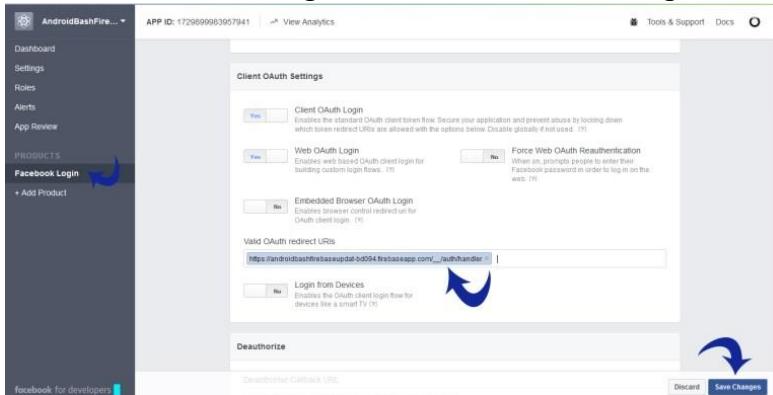


Figure 9.24 Add Product

13. Go to **App Review** section of the Facebook developers page and make your App available to public by clicking on the switch as shown below.

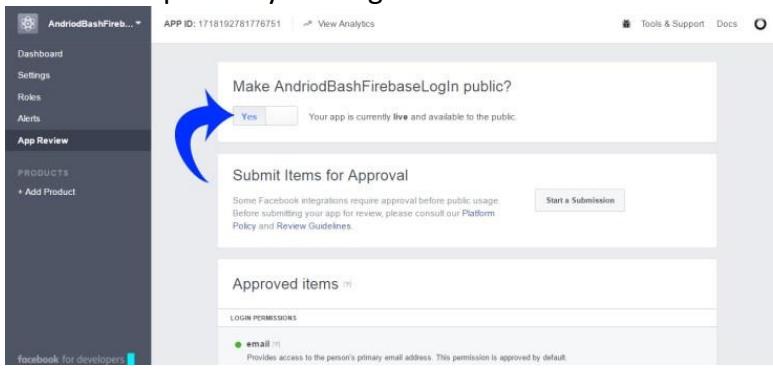


Figure 9.25 App Review

Part IV : Includes modifications to build.gradle file and AndroidManifest.xml file

1. At this point of time, the **build.gradle** (Module:app) file should look like below.

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"
```

```

defaultConfig {
    applicationId "com.androidbash.androidbashfirebaseupdated"
    minSdkVersion 16
    targetSdkVersion 23
    versionCode 1
    versionName "1.0"
}
buildTypes {
    release {
        minifyEnabled false
        proguardFiles getDefaultProguardFile('proguard-android.txt'),
'proguard-rules.pro'
    }
}
packagingOptions {
    exclude 'META-INF/LICENSE'
    exclude 'META-INF/LICENSE-FIREBASE.txt'
    exclude 'META-INF/NOTICE'
}
repositories {
    mavenCentral()
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:design:23.1.1'
    compile 'com.google.firebaseio:firebase-auth:9.2.0'
    compile 'com.firebaseio:firebase-client-android:2.5.2'
    compile 'com.facebook.android:facebook-android-sdk:4.9.0'
    compile 'com.google.android.gms:play-services-auth:9.2.0'
}

apply plugin: 'com.google.gms.google-services'

```

2. After adding Internet permission, meta-data and FacebookActivity to the manifest : AndroidManifest.xml looks like this.

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.androidbash.androidbashfirebaseupdated" >
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:name=".CustomApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <meta-data android:name="com.facebook.sdk.ApplicationId"
        android:value="@string/facebook_app_id"/>

        <activity android:name="com.facebook.FacebookActivity"
            android:configChanges="keyboard|keyboardHidden|screenLayout|screen
Size|orientation"
            android:theme="@android:style/Theme.Translucent.NoTitleBar"
            android:label="@string/app_name" />

        <activity
            android:name=".LoginActivity"
            android:label="AndroidDashFirebase Log In"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />

```

```

        </intent-filter>
    </activity>

    <activity
        android:name=".SignUpActivity"
        android:label="Sign Up Activity"
        android:theme="@style/AppTheme.NoActionBar" />

    <activity android:name=".MainActivity"
        android:label="AndroidBash Home"
        android:theme="@style/AppTheme.NoActionBar"/>

</application>

</manifest>

```

Part V : Includes modifications to the existing XML layout files

- After adding the Facebook Log In Button, the **activity_login.xml** file of your application has the following code in it.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".LoginActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="@color/colorAccent"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="50dp"
        android:gravity="center_horizontal">

        <ProgressBar
            android:id="@+id/progress_bar_login"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
            android:layout_marginBottom="8dp"
            android:visibility="gone" />

        <LinearLayout
            android:id="@+id/login_details"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <TextView
                android:id="@+id/text_view_email_id"

```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="15dp"
        android:fontFamily="sans-serif-medium"
        android:text="Enter Your Email"
        android:textColor="#000000"
        android:textSize="18sp"
        android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_email_id"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:inputType="textEmailAddress" />

<TextView
    android:id="@+id/text_view_password"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="15dp"
    android:fontFamily="sans-serif-medium"
    android:text="Enter Your Password"
    android:textColor="#000000"
    android:textSize="18sp"
    android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_password"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/text_view_password"
    android:layout_marginLeft="15dp"
    android:inputType="textPassword" />

<Space
    android:layout_width="1dp"
    android:layout_height="20dp" />

<Button
    android:id="@+id/button_sign_in"
    style="?android:textAppearanceSmall"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#3333cc"
    android:onClick="onLoginClicked"
    android:padding="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:text="Log In"
    android:textStyle="bold"
    android:textColor="@color/colorText"/>

<Space
    android:layout_width="1dp"
    android:layout_height="35dp" />

<!--<Button-->
<!--android:id="@+id/button_facebook_sign_in"-->
<!--style="?android:textAppearanceSmall"-->
<!--android:layout_width="fill_parent"-->
<!--android:layout_height="fill_parent"-->
<!--android:background="@color/colorPrimaryDark"-->

```

```

<!--android:onClick="onFacebookLogInClicked"-->
<!--android:padding="10dp"-->
<!--android:layout_marginLeft="10dp"-->
<!--android:layout_marginRight="10dp"-->
<!--android:text="Login with Facebook"-->
<!--android:textStyle="bold"-->
<!--android:textColor="@color/colorText"/>-->

<com.facebook.login.widget.LoginButton
    android:id="@+id/button_facebook_login"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_centerInParent="true"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:textStyle="bold"
    />

<Space
    android:layout_width="1dp"
    android:layout_height="10dp" />

<Button
    android:id="@+id/button_sign_up"
    style="?android:textAppearanceSmall"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#0066ff"
    android:onClick="onSignUpClicked"
    android:padding="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:text="New User? Please Sign Up"
    android:textStyle="bold"
    android:textColor="@color/colorText"/>

</LinearLayout>

</RelativeLayout>

</RelativeLayout>

```

2. The acivity_main.xml includes an imageview to hold the facebook profile picture of the user who has signed in using Facebook. The acivity_main.xml has the following code in it.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/backone"
    android:fitsSystemWindows="true"
    tools:context=".MainActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr actionBarSize"

```

```

        android:background="@color/colorAccent"
        app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="50dp"
        android:gravity="center">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="50dp"
            android:orientation="vertical">
            <TextView
                android:id="@+id/text_view_name"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:enabled="true"
                android:gravity="center_horizontal"
                android:textColor="@color/secondary_text"
                android:textSize="20sp" />
            <TextView
                android:id="@+id/text_view_welcome"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:enabled="true"
                android:gravity="center"
                android:textColor="@color/secondary_text"
                android:textSize="24sp" />

            <Space
                android:layout_width="1dp"
                android:layout_height="10dp" />

            <ImageView
                android:layout_gravity="center"
                android:id="@+id/profile_picture"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"/>

            <Button
                android:id="@+id/button_change"
                android:layout_width="match_parent"
                android:layout_height="40dp"
                android:layout_marginLeft="0dp"
                android:layout_marginRight="4dp"
                android:layout_marginTop="14dp"
                android:background="@color/colorPrimary"
                android:paddingLeft="8dp"
                android:paddingRight="8dp"
                android:text="Change Welcome Text"
                android:textColor="@color/colorText"
                android:textSize="18sp" />

            <Space
                android:layout_width="1dp"
                android:layout_height="10dp" />

            <Button
                android:id="@+id/button_revert"
                android:layout_width="match_parent"
                android:layout_height="40dp"
                android:layout_marginLeft="1dp"
                android:layout_marginRight="4dp"

```

```

        android:layout_marginTop="14dp"
        android:background="@color/colorPrimary"
        android:paddingLeft="8dp"
        android:paddingRight="8dp"
        android:text="Revert Text"
        android:textColor="@color/colorText"
        android:textSize="18sp" />
    </LinearLayout>
</RelativeLayout>
</RelativeLayout>

```

3. The activity_sign_up.xml has the following code in it. This activity helps the user to Sign Up using Email,Name, Phonenumber and Password.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".SignUpActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="@color/colorAccent"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="50dp"
        android:gravity="center_horizontal">

        <ProgressBar
            android:id="@+id/progress_bar_sign_up"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentBottom="true"
            android:layout_centerHorizontal="true"
            android:layout_marginBottom="8dp"
            android:visibility="gone" />

        <LinearLayout
            android:id="@+id/email_sign_up_form"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/activity_horizontal_margin"
            android:orientation="vertical">

            <TextView
                android:id="@+id/text_view_username"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="15dp"
                android:layout_marginTop="15dp"

```

```

        android:fontFamily="sans-serif-medium"
        android:text="Enter Your User Name"
        android:textColor="#000000"
        android:textSize="18sp"
        android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_username"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:inputType="textPersonName" />

<TextView
    android:id="@+id/text_view_phone_number"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="15dp"
    android:fontFamily="sans-serif-medium"
    android:text="Enter Your Phone Number"
    android:textColor="#000000"
    android:textSize="18sp"
    android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_phone_number"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:inputType="number" />

<TextView
    android:id="@+id/text_view_new_email"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="15dp"
    android:fontFamily="sans-serif-medium"
    android:text="Enter Your Mail Id"
    android:textColor="#000000"
    android:textSize="18sp"
    android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_new_email"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:inputType="textEmailAddress" />

<TextView
    android:id="@+id/text_view_new_password"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="15dp"
    android:fontFamily="sans-serif-medium"
    android:text="Enter Your Password"
    android:textColor="#000000"
    android:textSize="18sp"
    android:textStyle="bold|italic" />

<EditText
    android:id="@+id/edit_text_new_password"

```

```

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="15dp"
        android:inputType="textPassword" />

    <Space
        android:layout_width="1dp"
        android:layout_height="30dp" />

    <Button
        android:id="@+id/button_user_sign_up"
        style="?android:textAppearanceSmall"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:onClick="onSignUpClicked"
        android:background="@color/colorPrimary"
        android:padding="10dp"
        android:text="Sign Up Now"
        android:textStyle="bold"
        android:textColor="@color/colorText"/>

    </LinearLayout>
</RelativeLayout>

</RelativeLayout>

```

Part VI : Includes modifications to the existing Java Class files (Activities)

1. The class **CustomApplication.java** has the following code in it.

```

import com.facebook.FacebookSdk;
import com.facebook.appevents.AppEventsLogger;
import com.firebaseio.client.Firebase;

public class CustomApplication extends android.app.Application {

    @Override
    public void onCreate() {
        super.onCreate();
        Firebase.setAndroidContext(this);

        FacebookSdk.sdkInitialize(getApplicationContext());
        AppEventsLogger.activateApp(this);
    }
}

```

2. The class **User.java** has the following code in it.

```

public class User {

    private String id;
    private String name;
    private String phoneNumber;
    private String email;
    private String password;

    public User() {
    }

    public User(String id, String name, String phoneNumber, String email,
String password) {
        this.id = id;
        this.name = name;
    }
}

```

```

        this.phoneNumber = phoneNumber;
        this.email = email;
        this.password = password;
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}

```

3. The class **SignUpActivity.java has the following code in it.**

```

import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import com.firebaseio.client.Firebase;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;

```

```

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class SignUpActivity extends AppCompatActivity {

    private static final String TAG = "AndroidBash";
    //Add YOUR Firebase Reference URL instead of the following URL
    private Firebase mRef = new Firebase("https://androidbash.firebaseioupdate-
bd094.firebaseio.com/");
    private User user;
    private EditText name;
    private EditText phoneNumber;
    private EditText email;
    private EditText password;
    private FirebaseAuth mAuth;
    private ProgressDialog mProgressDialog;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        mAuth = FirebaseAuth.getInstance();

    }

    @Override
    protected void onStart() {
        super.onStart();
        name = (EditText) findViewById(R.id.edit_text_username);
        phoneNumber = (EditText) findViewById(R.id.edit_text_phone_number);
        email = (EditText) findViewById(R.id.edit_text_new_email);
        password = (EditText) findViewById(R.id.edit_text_new_password);

    }

    @Override
    public void onStop() {
        super.onStop();
    }

    //This method sets up a new User by fetching the user entered details.
    protected void setUpUser() {
        user = new User();
        user.setName(name.getText().toString());
        user.setPhoneNumber(phoneNumber.getText().toString());
        user.setEmail(email.getText().toString());
        user.setPassword(password.getText().toString());
    }

    public void onSignUpClicked(View view) {
        createNewAccount(email.getText().toString(),
password.getText().toString());
        showProgressDialog();
    }

    private void createNewAccount(String email, String password) {
        Log.d(TAG, "createNewAccount:" + email);
        if (!validateForm()) {
            return;
        }
        //This method sets up a new User by fetching the user entered details.
        setUpUser();
        //This method takes in an email address and password,
        validates them and then creates a new user
    }
}

```

```

        // with the createUserWithEmailAndPassword method.
        // If the new account was created, the user is also signed in, and the
        AuthStateListener runs the onAuthStateChanged callback.
        // In the callback, you can use the getCurrentUser method to get the
        user's account data.

        mAuth.createUserWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, new
        OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {

                Log.d(TAG, "createUserWithEmailAndPassword:onComplete:" +
task.isSuccessful());
                hideProgressDialog();

                // If sign in fails, display a message to the user. If
sign in succeeds
                // the auth state listener will be notified and logic
to handle the
                // signed in user can be handled in the listener.
                if (!task.isSuccessful()) {
                    Toast.makeText(SignUpActivity.this,
"Authentication failed.",
                        Toast.LENGTH_SHORT).show();
                } else {
                    onAuthenticationSuccess(task.getResult().getUser())
;
                }
            }
        });

        private void onAuthenticationSuccess(FirebaseUser mUser) {
            // Write new user
            saveNewUser(mUser.getUid(), user.getName(), user.getPhoneNumber(),
user.getEmail(), user.getPassword());
            signOut();
            // Go to LoginActivity
            startActivity(new Intent(SignUpActivity.this, LoginActivity.class));
            finish();
        }

        private void saveNewUser(String userId, String name, String phone, String
email, String password) {
            User user = new User(userId, name, phone, email, password);

            mRef.child("users").child(userId).setValue(user);
        }

        private void signOut() {
            mAuth.signOut();
        }
        // This method, validates email address and password
        private boolean validateForm() {
            boolean valid = true;

            String userEmail = email.getText().toString();
            if (TextUtils.isEmpty(userEmail)) {
                email.setError("Required.");
                valid = false;
            } else {
                email.setError(null);
            }

```

```

        String userPassword = password.getText().toString();
        if (TextUtils.isEmpty(userPassword)) {
            password.setError("Required.");
            valid = false;
        } else {
            password.setError(null);
        }

        return valid;
    }

    public void showProgressDialog() {
        if (mProgressDialog == null) {
            mProgressDialog = new ProgressDialog(this);
            mProgressDialog.setMessage(getString(R.string.loading));
            mProgressDialog.setIndeterminate(true);
        }

        mProgressDialog.show();
    }

    public void hideProgressDialog() {
        if (mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
        }
    }
}

```

4. The class **MainActivity.java** has an ImageView to display the Facebook profile picture of the user who has logged-in to the application. The profile picture will be loaded into the ImageView once the Asynchronous task executes. If user chooses classic Email and Password method then this imageview will not be loaded with an image.

```

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.firebaseio.client.DataSnapshot;
import com.firebaseio.client.Firebase;
import com.firebaseio.client.FirebaseError;
import com.firebaseio.client.ValueEventListener;
import com.google.firebase.auth.FirebaseAuth;

import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = "AndroidBash";
    private Firebase myFirebaseRef;

```

```

private FirebaseAuth mAuth;
private TextView name;
private TextView welcomeText;
private Button changeButton;
private Button revertButton;
// To hold Facebook profile picture
private ImageView profilePicture;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);
    //Creates a reference for your Firebase database
    //Add YOUR Firebase Reference URL instead of the following URL
    myFirebaseRef = new Firebase("https://androidbash.firebaseioupdate-
bd094.firebaseio.com/users/");
    mAuth = FirebaseAuth.getInstance();
}

@Override
protected void onStart() {
    super.onStart();
    name = (TextView) findViewById(R.id.text_view_name);
    welcomeText = (TextView) findViewById(R.id.text_view_welcome);
    changeButton = (Button) findViewById(R.id.button_change);
    revertButton = (Button) findViewById(R.id.button_revert);
    profilePicture = (ImageView) findViewById(R.id.profile_picture);
    //Get the uid for the currently logged in User from intent data passed
    to this activity
    String uid = getIntent().getExtras().getString("user_id");
    //Get the imageUrl for the currently logged in User from intent data
    passed to this activity
    String imageUrl =
    getIntent().getExtras().getString("profile_picture");

    new ImageLoadTask(imageUrl, profilePicture).execute();

    //Referring to the name of the User who has logged in currently and
    adding a valueChangeListener
    myFirebaseRef.child(uid).child("name").addValueEventListener(new
    ValueEventListener() {
        //onDataChange is called every time the name of the User changes
        in your Firebase Database
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            //Inside onDataChange we can get the data as an Object from
            the dataSnapshot
            //getValue returns an Object. We can specify the type by
            passing the type expected as a parameter
            String data = dataSnapshot.getValue(String.class);
            name.setText("Hello " + data + ", ");
        }

        //onCancelled is called in case of any error
        @Override
        public void onCancelled(FirebaseError firebaseError) {
            Toast.makeText(getApplicationContext(), "" +
            firebaseError.getMessage(), Toast.LENGTH_LONG).show();
        }
    });

    //A firebase reference to the welcomeText can be created in following
    ways :
    // You can use this :

```

```

        //Firebase myAnotherFirebaseRefForWelcomeText=new
        Firebase("https://androidbash.firebaseioupdat-
        bd094.firebaseio.com/welcomeText");*/
        //OR as shown below
        myFirebaseRef.child("welcomeText").addValueEventListener(new
        ValueEventListener() {
            //onDataChange is called every time the data changes in your
        Firebase Database
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                //Inside onDataChange we can get the data as an Object from
        the dataSnapshot
                //getValue returns an Object. We can specify the type by
        passing the type expected as a parameter
                String data = dataSnapshot.getValue(String.class);
                welcomeText.setText(data);
            }

            //onCancelled is called in case of any error
            @Override
            public void onCancelled(FirebaseError firebaseError) {
                Toast.makeText(getApplicationContext(), "" +
        firebaseError.getMessage(), Toast.LENGTH_LONG).show();
            }
        });

        //onClicking changeButton the value of the welcomeText in the Firebase
        database gets changed
        changeButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                myFirebaseRef.child("welcomeText").setValue("Android App
        Development @ AndroidBash");
            }
        });

        //onClicking revertButton the value of the welcomeText in the Firebase
        database gets changed
        revertButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                myFirebaseRef.child("welcomeText").setValue("Welcome to
        Learning @ AndroidBash");
            }
        });
    }

    @Override
    public void onStop() {
        super.onStop();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        if (id == R.id.action_logout) {

```

```

        mAuth.signOut();
        finish();
    }

    return super.onOptionsItemSelected(item);
}

public class ImageLoadTask extends AsyncTask<Void, Void, Bitmap> {

    private String url;
    private ImageView imageView;

    public ImageLoadTask(String url, ImageView imageView) {
        this.url = url;
        this.imageView = imageView;
    }

    @Override
    protected Bitmap doInBackground(Void... params) {
        try {
            URL urlConnection = new URL(url);
            HttpURLConnection connection = (HttpURLConnection)
urlConnection
                .openConnection();
            connection.setDoInput(true);
            connection.connect();
            InputStream input = connection.getInputStream();
            Bitmap myBitmap = BitmapFactory.decodeStream(input);
            return myBitmap;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }

    @Override
    protected void onPostExecute(Bitmap result) {
        super.onPostExecute(result);
        imageView.setImageBitmap(result);
    }
}
}

```

5. Create a new Java class file under src folder and name it as **LoginActivity.java**. Add the following code in it.

```

import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import com.facebook.AccessToken;
import com.facebook.CallbackManager;
import com.facebook.FacebookCallback;
import com.facebook.FacebookException;
import com.facebook.FacebookSdk;
import com.facebook.login.LoginResult;
import com.facebook.login.widget.LoginButton;
import com.firebaseio.client.Firebase;
import com.google.android.gms.tasks.OnCompleteListener;

```

```

import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FacebookAuthProvider;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class LoginActivity extends AppCompatActivity {

    private static final String TAG = "AndroidBash";
    public User user;
    private EditText email;
    private EditText password;
    private FirebaseAuth mAuth;
    private FirebaseAuth.AuthStateListener mAuthListener;
    private ProgressDialog mProgressDialog;
    //Add YOUR Firebase Reference URL instead of the following URL
    Firebase mRef=new Firebase("https://androidbash.firebaseioupdate-
bd094.firebaseio.com/users/");

    //FaceBook callbackManager
    private CallbackManager callbackManager;
    //

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        mAuth = FirebaseAuth.getInstance();

        FirebaseUser mUser = mAuth.getCurrentUser();
        if (mUser != null) {
            // User is signed in
            Intent intent = new Intent(getApplicationContext(),
            MainActivity.class);
            String uid = mAuth.getCurrentUser().getUid();
            String image=mAuth.getCurrentUser().getPhotoUrl().toString();
            intent.putExtra("user_id", uid);
            if(image!=null || image!=""){
                intent.putExtra("profile_picture",image);
            }
            startActivity(intent);
            finish();
            Log.d(TAG, "onAuthStateChanged:signed_in:" + mUser.getUid());
        }

        mAuthListener = new FirebaseAuth.AuthStateListener() {
            @Override
            public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth)
            {
                FirebaseUser mUser = firebaseAuth.getCurrentUser();
                if (mUser != null) {
                    // User is signed in
                    Log.d(TAG, "onAuthStateChanged:signed_in:" +
mUser.getUid());
                } else {
                    // User is signed out
                    Log.d(TAG, "onAuthStateChanged:signed_out");
                }
            }
        };
        //FaceBook
        FacebookSdk.sdkInitialize(getApplicationContext());
        callbackManager = CallbackManager.Factory.create();
    }
}

```

```

        LoginButton loginButton = (LoginButton)
findViewById(R.id.button_facebook_login);
        loginButton.setReadPermissions("email", "public_profile");
        loginButton.registerCallback(callbackManager, new
FacebookCallback<LoginResult>() {
            @Override
            public void onSuccess(LoginResult loginResult) {
                Log.d(TAG, "facebook:onSuccess:" + loginResult);
                signInWithFacebook(loginResult.getAccessToken());
            }

            @Override
            public void onCancel() {
                Log.d(TAG, "facebook:onCancel");
            }

            @Override
            public void onError(FacebookException error) {
                Log.d(TAG, "facebook:onError", error);
            }
        });
    //
}

@Override
protected void onStart() {
    super.onStart();
    email = (EditText) findViewById(R.id.edit_text_email_id);
    password = (EditText) findViewById(R.id.edit_text_password);
    mAuth.addAuthStateListener(mAuthListener);
}

@Override
public void onStop() {
    super.onStop();
    if (mAuthListener != null) {
        mAuth.removeAuthStateListener(mAuthListener);
    }
}

//FaceBook
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
    super.onActivityResult(requestCode, resultCode, data);
    callbackManager.onActivityResult(requestCode, resultCode, data);
}
//

protected void setUpUser() {
    user = new User();
    user.setEmail(email.getText().toString());
    user.setPassword(password.getText().toString());
}

public void onSignUpClicked(View view) {
    Intent intent = new Intent(this, SignUpActivity.class);
    startActivity(intent);
}

public void onLoginClicked(View view) {
    setUpUser();
    signIn(email.getText().toString(), password.getText().toString());
}

private void signIn(String email, String password) {

```

```

        Log.d(TAG, "signIn:" + email);
        if (!validateForm()) {
            return;
        }

        showProgressDialog();

        mAuth.signInWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, new
        OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                Log.d(TAG, "signInWithEmail:onComplete:" +
                    task.isSuccessful());

                // If sign in fails, display a message to the user. If
                sign in succeeds
                // the auth state listener will be notified and logic
                to handle the
                // signed in user can be handled in the listener.
                if (!task.isSuccessful()) {
                    Log.w(TAG, "signInWithEmail",
                        task.getException());
                    Toast.makeText(LoginActivity.this, "Authentication
                    failed.",
                        Toast.LENGTH_SHORT).show();
                } else {
                    Intent intent = new
                    Intent(getApplicationContext(), MainActivity.class);
                    String uid = mAuth.getCurrentUser().getUid();
                    intent.putExtra("user_id", uid);
                    startActivity(intent);
                    finish();
                }
            }
        });
    }

    private boolean validateForm() {
        boolean valid = true;

        String userEmail = email.getText().toString();
        if (TextUtils.isEmpty(userEmail)) {
            email.setError("Required.");
            valid = false;
        } else {
            email.setError(null);
        }

        String userPassword = password.getText().toString();
        if (TextUtils.isEmpty(userPassword)) {
            password.setError("Required.");
            valid = false;
        } else {
            password.setError(null);
        }

        return valid;
    }

    private void signInWithFacebook(AccessToken token) {
        Log.d(TAG, "signInWithFacebook:" + token);

        showProgressDialog();
    }
}

```

```

        AuthCredential credential =
FacebookAuthProvider.getCredential(token.getToken());
        mAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    Log.d(TAG, "signInWithCredential:onComplete:" +
task.isSuccessful());

                    // If sign in fails, display a message to the user. If
sign in succeeds
                    // the auth state listener will be notified and logic
to handle the
                    // signed in user can be handled in the listener.
                    if (!task.isSuccessful()) {
                        Log.w(TAG, "signInWithCredential",
task.getException());
                        Toast.makeText(LoginActivity.this, "Authentication
failed.",
                            Toast.LENGTH_SHORT).show();
                    } else{
                        String uid=task.getResult().getUser().getUid();
                        String
name=task.getResult().getUser().getDisplayName();
                        String
email=task.getResult().getUser().getEmail();
                        String
image=task.getResult().getUser().getPhotoUrl().toString();

                        //Create a new User and Save it in Firebase
database
                        User user = new User(uid,name,null,email,null);

                        mRef.child(uid).setValue(user);

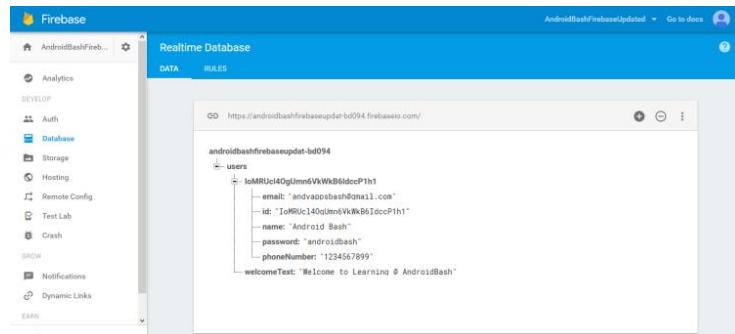
                        Intent intent = new
Intent(getApplicationContext(), MainActivity.class);
                        intent.putExtra("user_id",uid);
                        intent.putExtra("profile_picture",image);
                        startActivity(intent);
                        finish();
                    }
                    hideProgressDialog();
                }
            });
    }

    public void showProgressDialog() {
        if (mProgressDialog == null) {
            mProgressDialog = new ProgressDialog(this);
            mProgressDialog.setMessage(getString(R.string.loading));
            mProgressDialog.setIndeterminate(true);
        }
        mProgressDialog.show();
    }

    public void hideProgressDialog() {
        if (mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
        }
    }
}

```

The following image shows the Firebase database once a User signs up on your application. Every user who signs up will be saved under the users node with unique uid.



A screenshot of the Firebase Realtime Database console. The left sidebar shows various project settings like Analytics, Storage, and Remote Config. The main area is titled 'Realtime Database' with tabs for 'DATA' and 'RULES'. Below that is a URL: 'https://androidbashfirebasedataupdatebd094.firebaseio.com/'. The data structure under 'users' shows a single user node with a unique ID ('JmMRUc14OgUmnenVKKb6ldccP1h1') containing fields: email ('androidbash@hotmail.com'), id ('JmMRUc14OgUmnenVKKb6ldccP1h1'), name ('Android Bash'), password ('androidbash'), and phoneNumber ('1234567899'). A welcomeText field is also present with the value 'Welcome to Learning @ AndroidBash'.

Figure 9.26 Firebase Rules

That's it!. If you have followed all the steps you can now Run your application and test how it works. The following image shows the series of activities, when the application runs on an Emulator. Facebook name as well as Facebook profile picture are loaded into MainActivity once the user is logged-in.

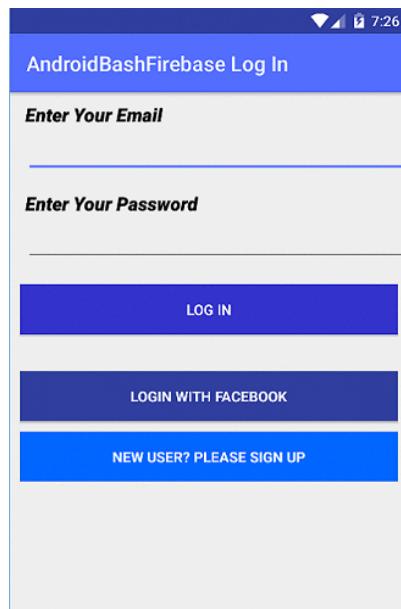


Figure 9.27 Result Display Apps

MODUL 10

AdMob

[AdMob](#) is a multi platform mobile ad network that allows you to monetize your android app. By integrating AdMob you can start earning right away. It is very useful particularly when you are publishing a free app and want to earn some money from it. Integrating AdMob is such an easy task that it takes not more than 5mins. In this article we'll build a simple app with two screen to show the different types of ads that AdMob supports.

1. Create Project.

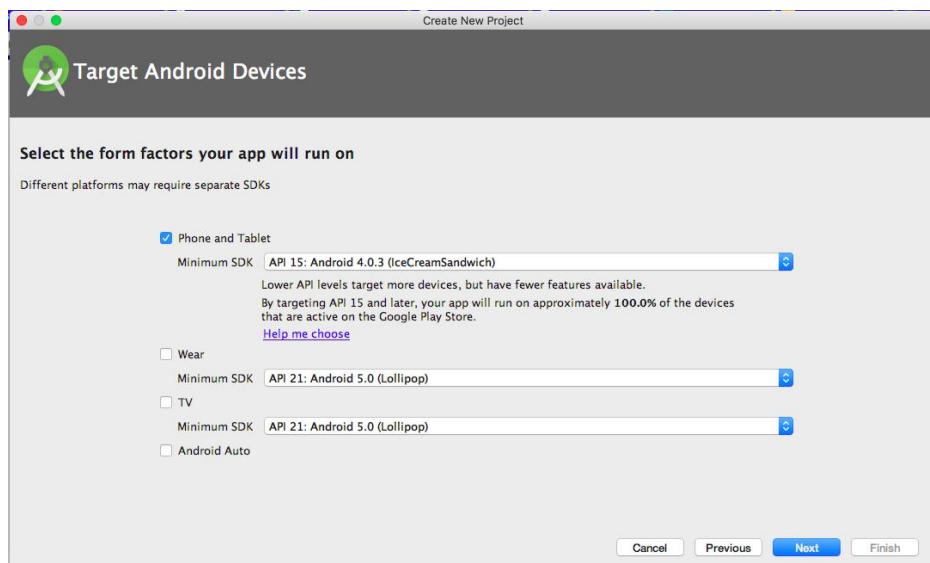
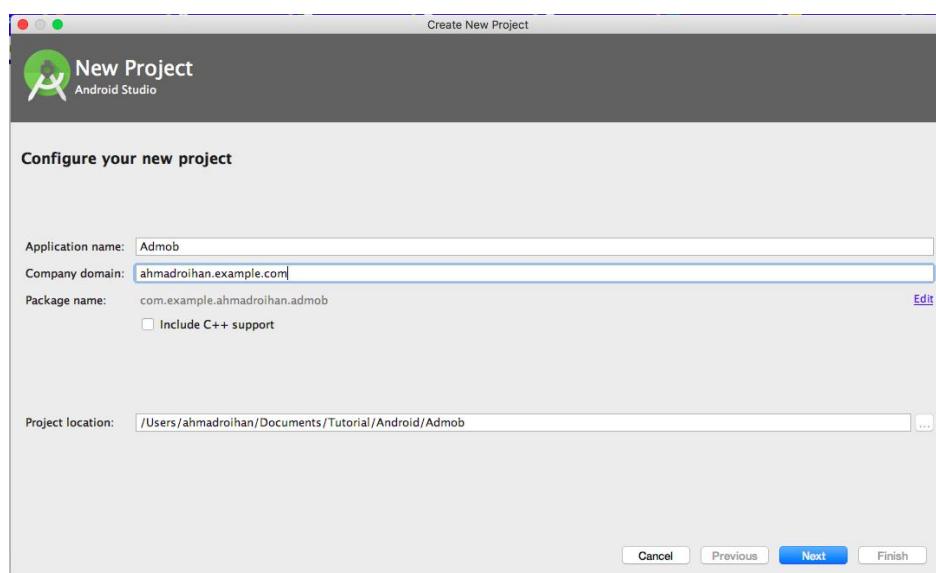


Figure 10.1 Create Project

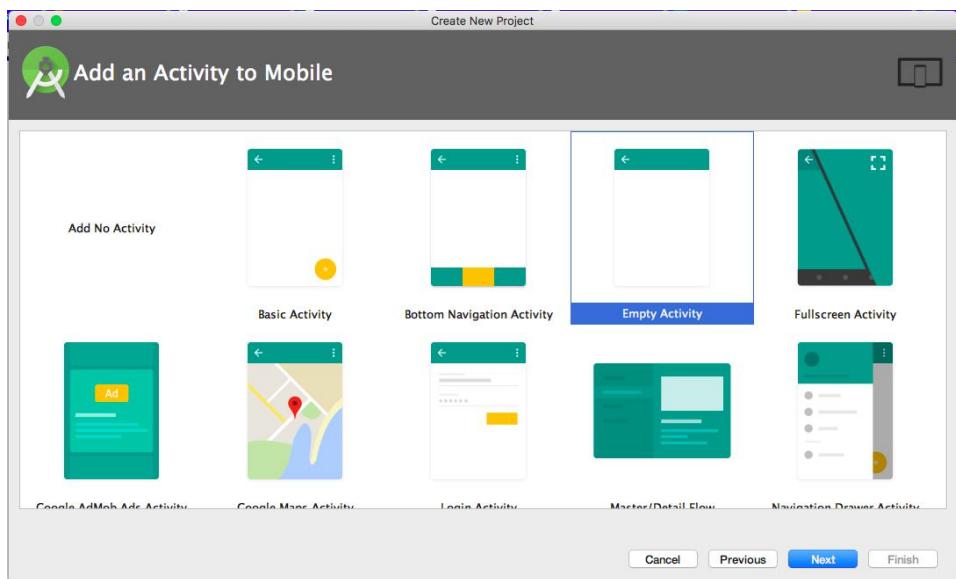


Figure 10.2 Add Activity

2. Add the following line and embed it on **build.gradle**.
compile 'com.google.android.gms:play-services-ads:8.3.0'

The screenshot shows the 'Build.gradle' file in the Android Studio IDE. At the top, a yellow bar indicates 'Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.' On the right, there is a 'Sync Now' button. The code itself defines an application with various configurations and dependencies. The 'dependencies' block includes several entries, notably 'compile 'com.google.android.gms:play-services-ads:8.3.0'', which is highlighted with a yellow background.

```
1 apply plugin: 'com.android.application'
2
3 android {
4     compileSdkVersion 26
5     buildToolsVersion "26.0.0"
6     defaultConfig {
7         applicationId "com.example.ahmadroihan.admob"
8         minSdkVersion 15
9         targetSdkVersion 26
10        versionCode 1
11        versionName "1.0"
12        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
13    }
14    buildTypes {
15        release {
16            minifyEnabled false
17            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
18        }
19    }
20 }
21
22 dependencies {
23     compile fileTree(dir: 'libs', include: ['*.jar'])
24     androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
25         exclude group: 'com.android.support', module: 'support-annotations'
26     })
27     compile 'com.android.support:appcompat-v7:26.+'
28     compile 'com.android.support.constraint:constraint-layout:1.0.2'
29     compile 'com.google.android.gms:play-services-ads:8.3.0'
30     testCompile 'junit:junit:4.12'
31 }
32 }
```

Figure 10.3 Build Gradle

3. Adding Ad Unit ID, if you do not have ID please register to admob.com to get ID in **string.xml**.

The screenshot shows the 'string.xml' file in the Android Studio IDE. It contains XML code defining string resources. One of the entries, '<string name="banner_ad_unit_id">ca-app-pub-5133451899564854/2435512922</string>', is highlighted with a yellow background.

```
1 <resources>
2     <string name="app_name">Admob</string>
3     <string name ="banner_ad_unit_id">ca-app-pub-5133451899564854/2435512922</string>
4 </resources>
```

Figure 10.4 Source Code String.xml

4. Then add the following command line to **activity_main.xml**.

The screenshot shows the Android Studio code editor with the XML file 'Main.xml' open. The code defines a vertical relative layout containing a text view and an ad view.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:ads="http://schemas.android.com/apk/res-auto"
    tools:context="com.example.ahmadroihan.admob.MainActivity">

    <TextView
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Roihan Ganteng !"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="119dp" />

    <com.google.android.gms.ads.AdView
        android:id="@+id/adView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_alignParentBottom="true"
        ads:adSize="BANNER"
        ads:adUnitId="@string/banner_ad_unit_id">
        </com.google.android.gms.ads.AdView>
    </RelativeLayout>
```

Figure 10.5 Source Code Activity Main.xml

5. You can add the following command line to the import section.

```
import com.google.android.gms.ads.AdRequest;
import com.google.android.gms.ads.AdView;
```

6. Then you bundle the following command line under SetContentView

```
AdView mAdView = (AdView) findViewById(R.id.adView);
AdRequest adRequest = new AdRequest.Builder().build();
mAdView.loadAd(adRequest);
```

References :

<http://www.okedroid.com/2015/11/cara-memasang-iklan-admob-pada-aplikasi-android-di-android-studio.html>

<https://teknorial.com/tutorial-memasang-admob-di-aplikasi-android/>

<https://www.androidhive.info/2016/02/android-how-to-integrate-google-admob-in-your-app/>

MODUL 11

PUBLISH APPS

1. Creating a Signed APK

When developing on the Android operating system, you use the *Android application package* (APK) format to distribute apps. Android requires that APKs are digitally **signed** with a certificate before they can be installed. The certificate is used to identify the author of the app. More information about app signing can be found here. A signed APK can be generated manually from command line or in Android Studio. In Android Studio, select **Build\Generate Signed APK**.

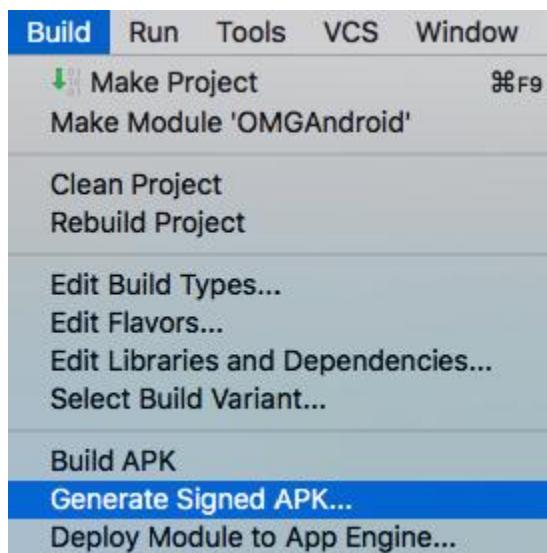


Figure 11.1 Generate Signed APK

Your app only has one module, so select **Next**.



Figure 11.2 Module

With your app module chosen, you need to tell Android Studio how to sign the APK. You'll use a key store, which will stay private to you. Android Studio will guide you through creating a new key store; start by selecting **Create new**.

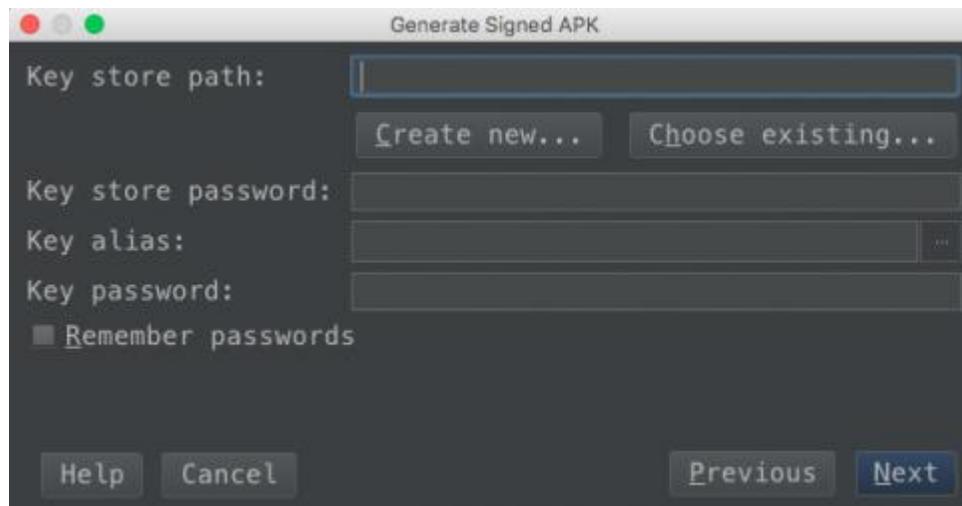


Figure 11.3 Key Store Part

Choose a filename and location on your system, set and confirm a secure password, and fill in the rest of the information about yourself:

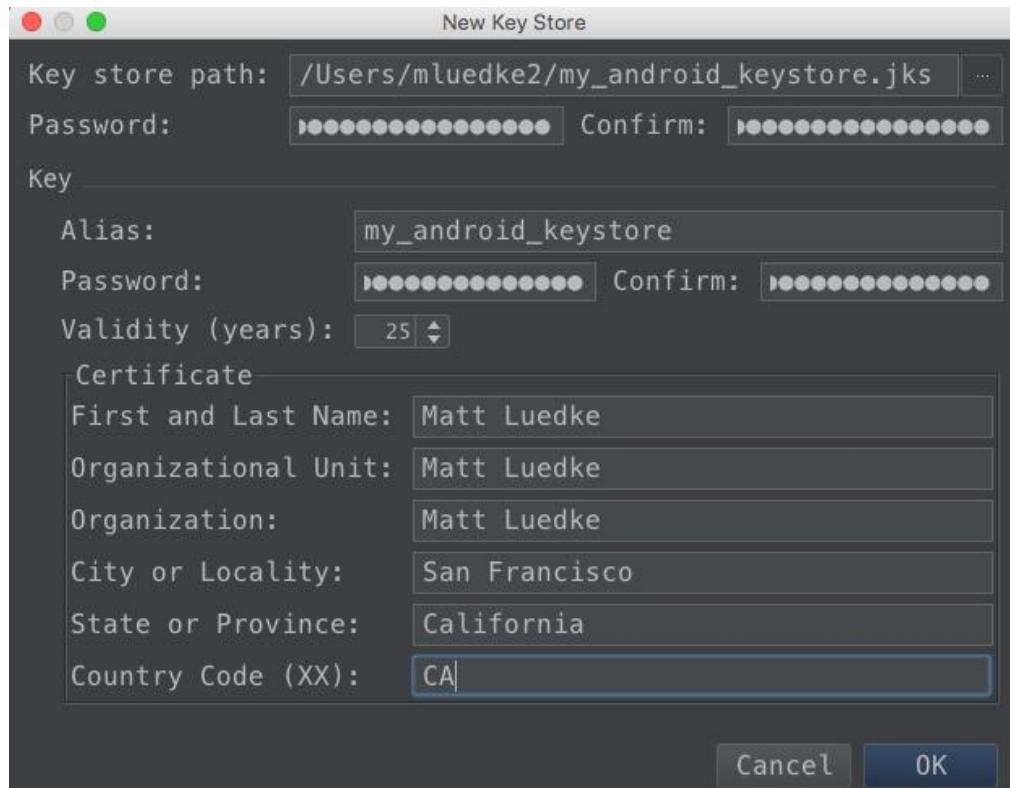


Figure 11.4 Add Key Store Part

Click **OK**, and Android Studio will create your key store. You'll need that file, with its password info, to make any updates to your app, so don't lose it!

With your new key store, Android Studio returns you to the dialog box you saw before. Now the key store information is pre-filled, so just click **Next**.

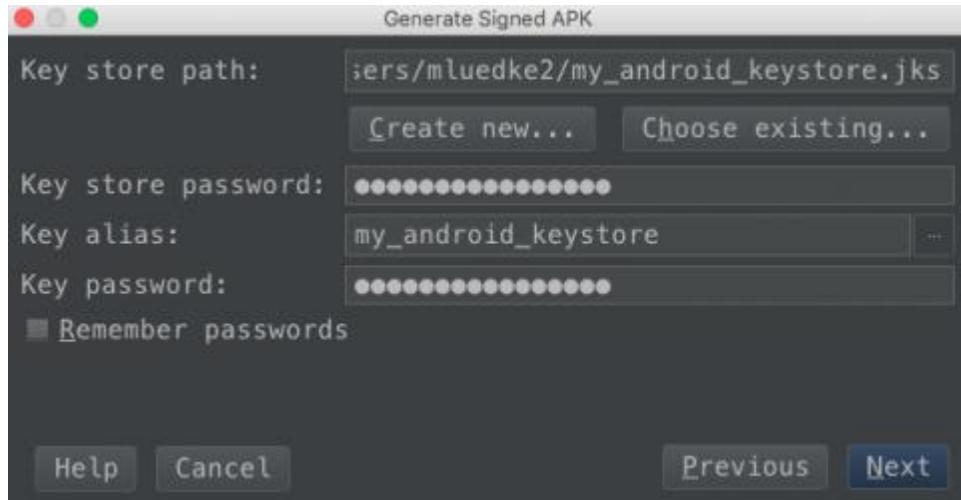


Figure 11.5 Key store Information

Choose a destination folder for your signed APK, specify your Build Type as **release**, and click **Finish**.



Figure 11.6 Destination Folder Apk

When the packaging process is complete, Android Studio will notify you that your APK is ready, and let you open it in Explorer.

2. The Google Play Developer Console

To submit an app to Google Play, you first need a Google Account. Create one on <https://accounts.google.com/> (or sign in, if you already have an account) before going further. When you are signed into your Google Account, navigate to <https://play.google.com/apps/publish/> (the Google Play Developer Console). You should see a screen like this:

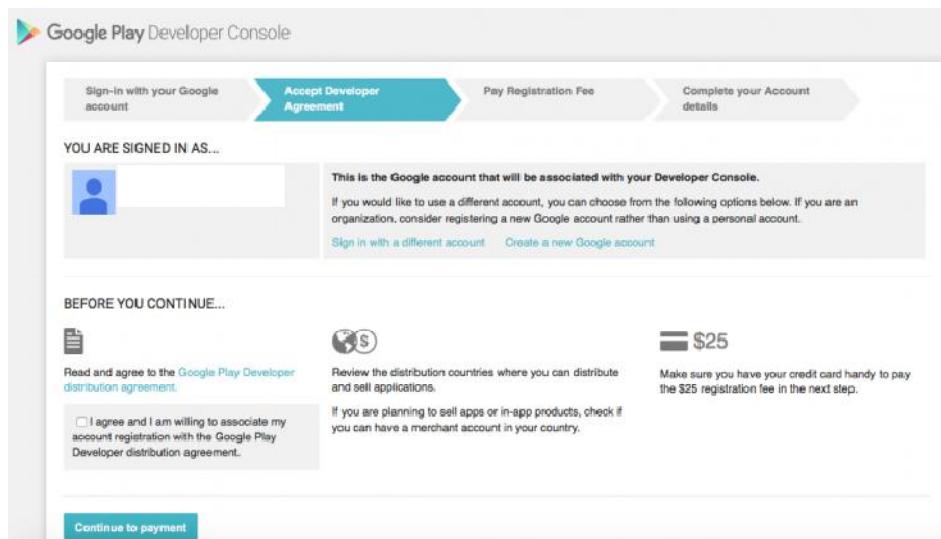


Figure 11.7 Submit apps

Agree to the Google Play Developer distribution agreement, pay the one-time \$25 Google Developer Registration Fee, then complete your Developer Profile on this screen:

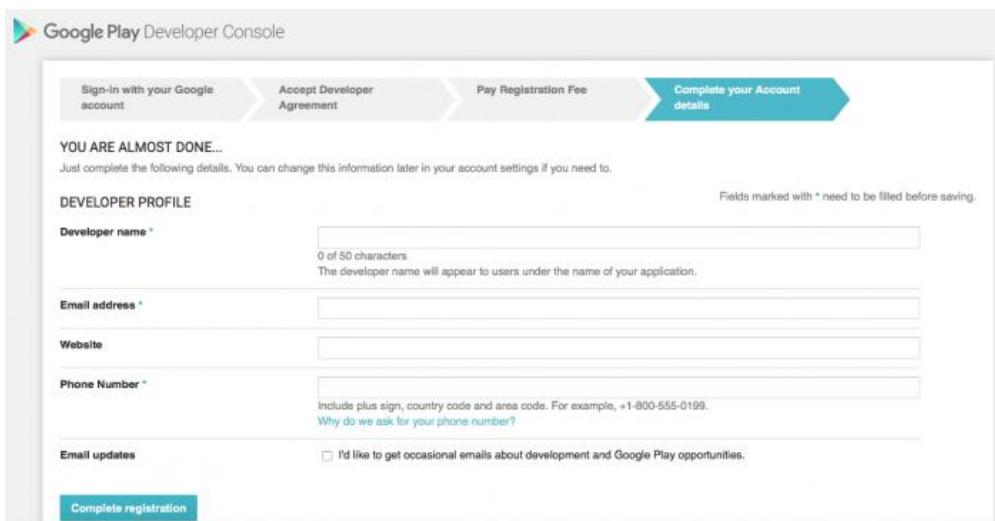


Figure 11.8 Registration

With your developer account now registered, your developer console has more options. Click **Publish an Android App on Google Play** to continue.

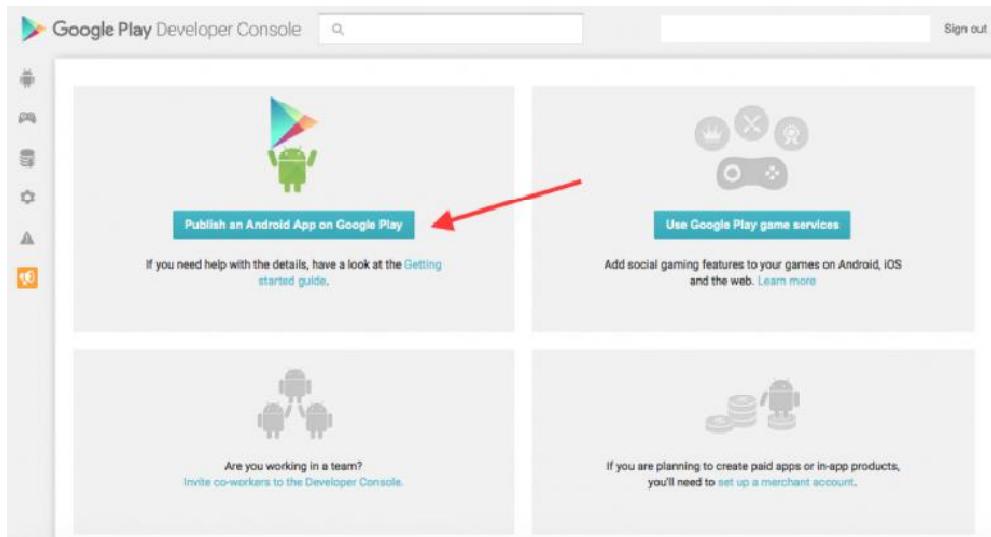


Figure 11.9 Publish Apps

Since this is your first application, you're given a dialog box to specify the app name and a choice on how to begin. Type in your app name and choose **Upload APK**.

The dialog box has a title 'ADD NEW APPLICATION'. It contains the following fields:

- Default language ***: A dropdown menu set to 'English (United States) – en-US'.
- Title ***: An input field containing 'Bookmaster General' with the note '18 of 30 characters' below it. A red arrow points to the placeholder text 'What would you like to start with?'.

What would you like to start with?
- Buttons at the bottom**: 'Upload APK' (highlighted in red), 'Prepare Store Listing', and 'Cancel'.

Figure 11.10 Upload Apk

This creates a draft store listing for your app, currently containing nothing except the app title. To begin, click **Upload your first APK to Production**.

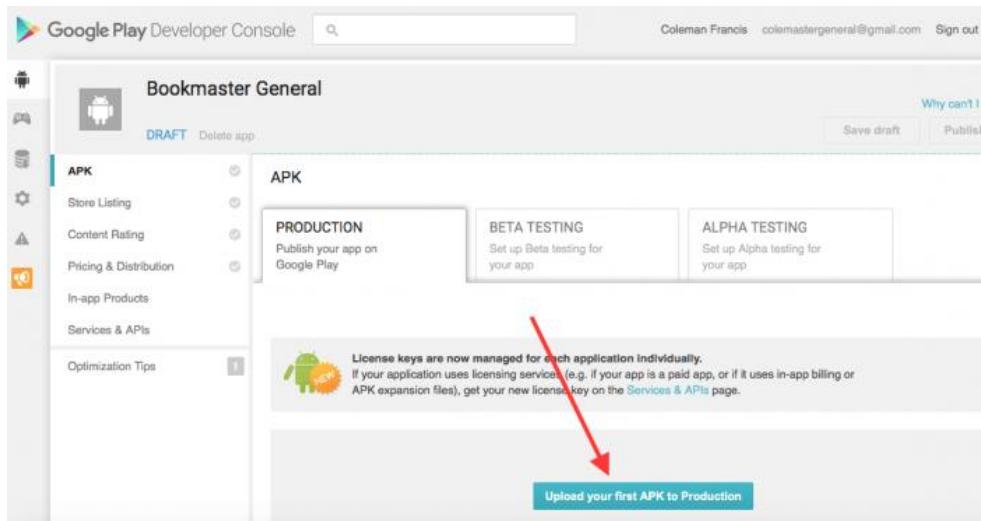


Figure 11.11 Upload Apk Product

Choose the APK file you created earlier, and upload it. When the upload is complete, you will see the app listing updated like this:

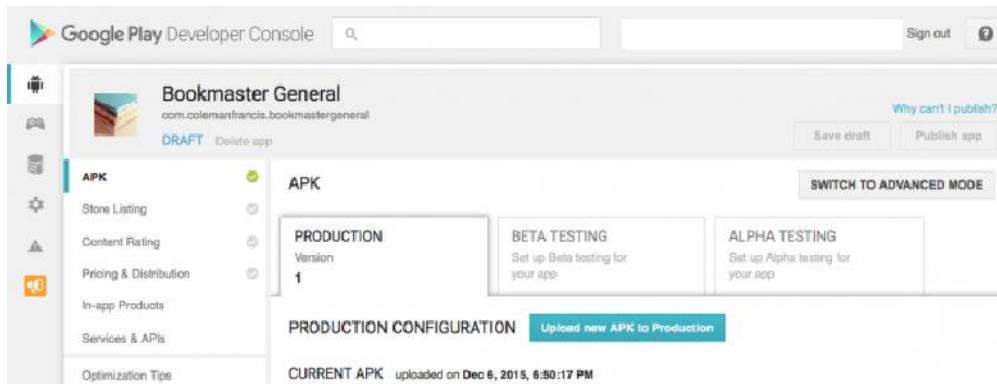


Figure 11.12 Configuration

Notice that the check mark next to **APK** is green, indicating that your APK is ready!

Time to move on to the remaining check marks...

3. Completing the Store Listing

Bookmaster General is a pretty simple app, so the rest of the process will be easy. Click on the **Store Listing** check mark in the menu on the left page and fill in the **Short Description** and **Full Description** fields with information about the app:

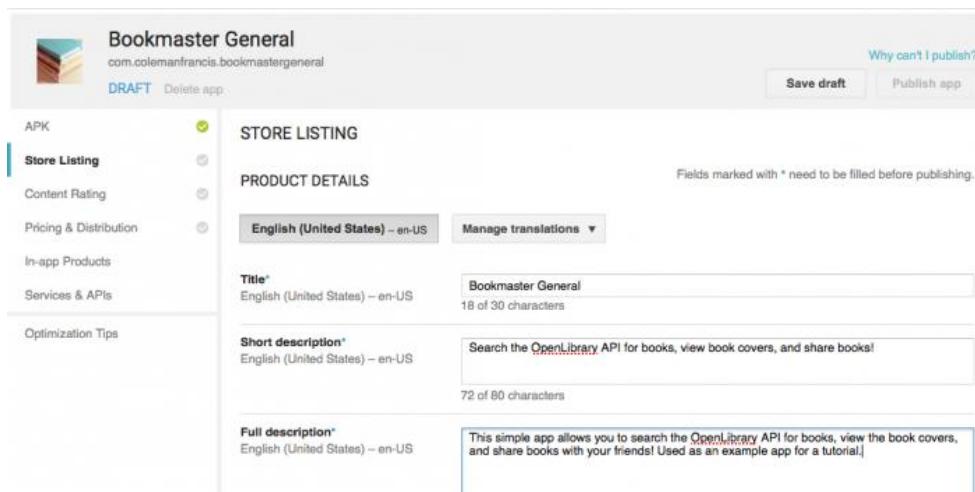


Figure 11.13 Book General

Note: For your own app, you will want to create as engaging a description as possible, so start thinking about this early in the development process!

Next, upload at least two screenshots of the app. If you have the app on an Android device, you can simply take screenshots straight from the device.

If you prefer to use an emulator for the screenshots, you can easily make these in Android Studio. Build and run the app, then navigate to the part of the app where you'd like the screenshot.

With the **Android Monitor** tab opened in Android Studio, click the camera icon on the far left:

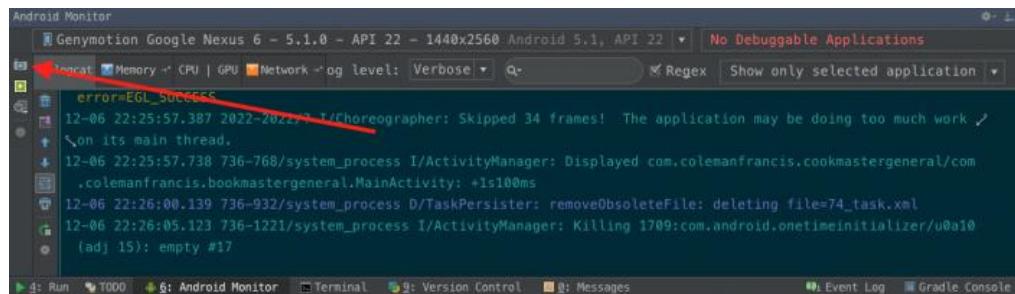
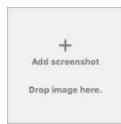


Figure 11.14 Android Monitor

Android Studio will present you with the captured image from the emulator. Click Save and select the file location on your system.

Drag these images onto your store listing where you see the following prompt:



In addition to screenshots, you need to upload a 512×512 high-resolution version of the app icon and a 1024×500 “feature graphic” to display at the top of the page.

Complete the store listing by specifying an **Application Type**, **Category**, **Content rating**, and **Contact Email**. Other fields are optional.

Scroll back to the top of the store listing and click **Save draft**. Since the listing is now complete, the **Store Listing** check mark is now green. We’re making progress!

Click on the **Content Rating** check mark to see a description of the upcoming content rating questionnaire.

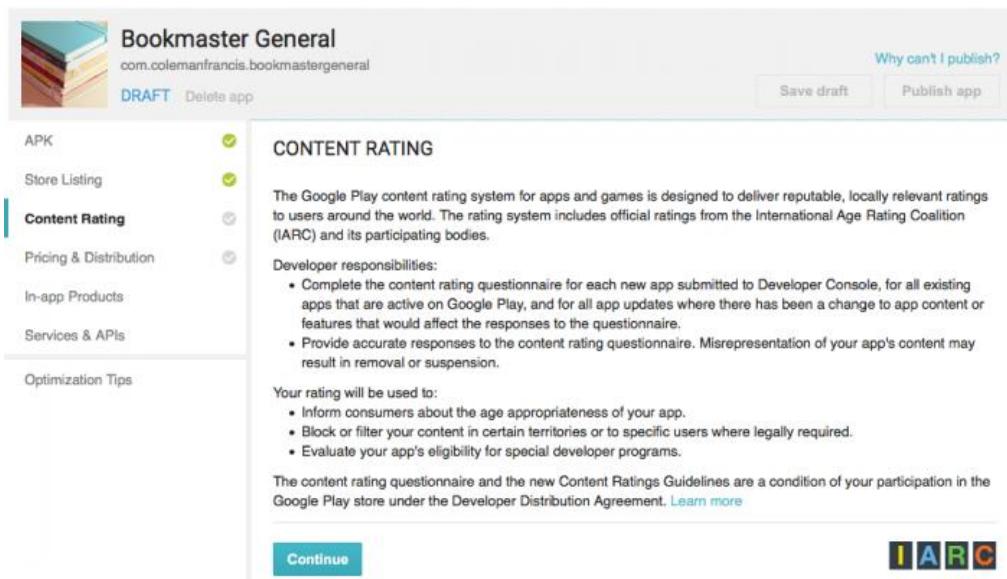


Figure 11.15 Content Rating

This questionnaire asks a straightforward series of questions about the content of the app. For those using our sample app, Bookmaster General is in the **Reference, News, or Educational** category; answer **No** to all the content questions. If you’re using your own app, give the appropriate answers.

Once you are finished, click **Save questionnaire**, then **Calculate rating**. You will see the ratings for the various locales where your app may be on sale:

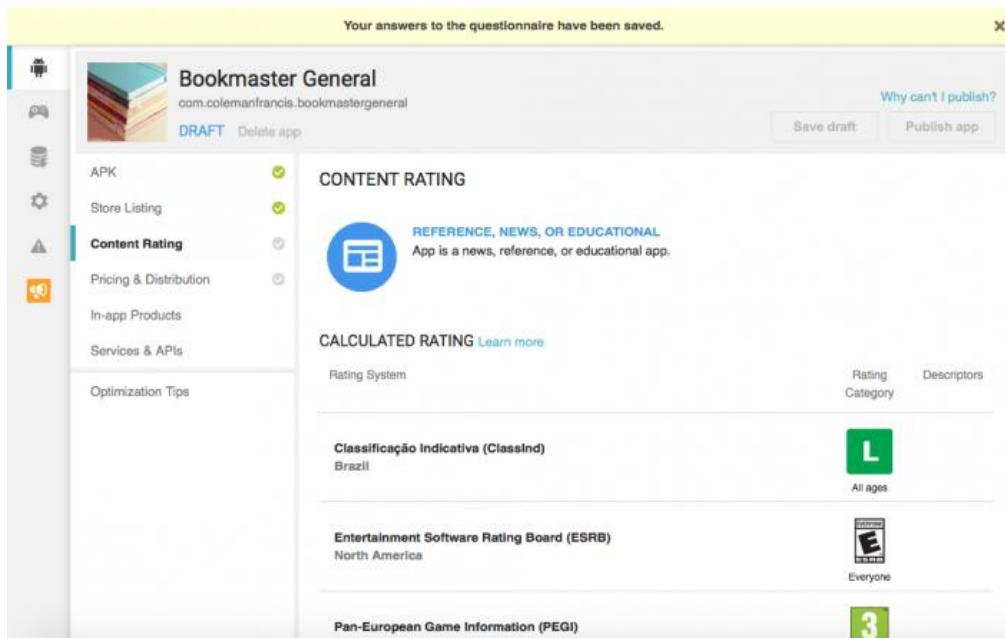


Figure 11.16 Pricing & Distribution

Scroll to the bottom of the list and click **Apply rating**. The **Content Rating** check mark is now green.

Select the **Pricing & Distribution** check mark for the final step: setting the price of your app, and in which countries it will be available. For this example, the app will be free and available universally.

The **Free** pricing option is selected by default, so keep that choice. Check the box next to **SELECT ALL COUNTRIES**.

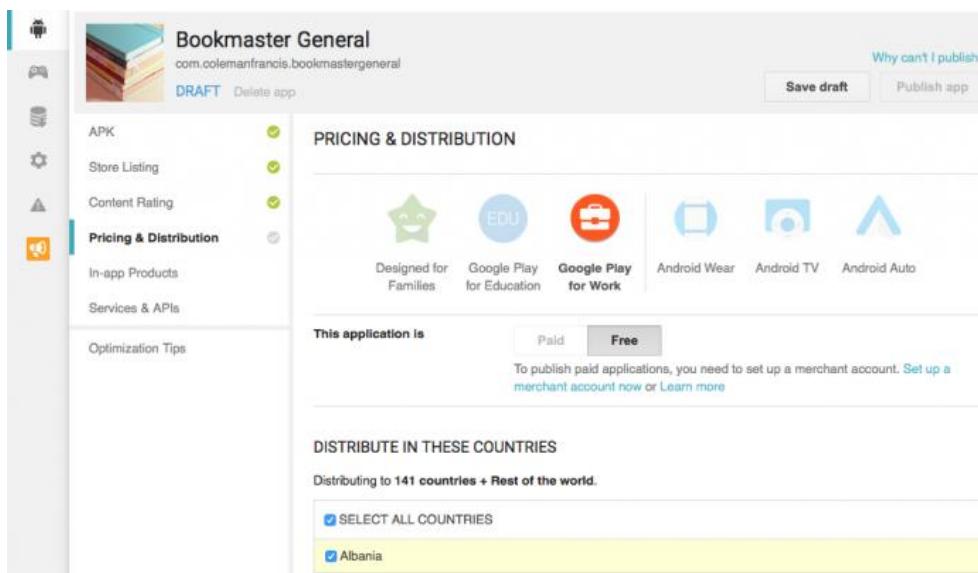
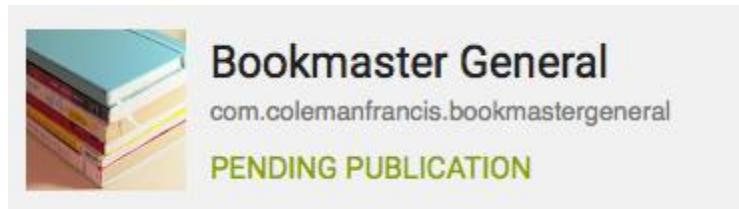


Figure 11.17 Countries

Scroll down past the list of countries, and you will find several more distribution options. Check the boxes by **Content guidelines** and **US export laws** to indicate your app's compliance with relevant rules, then scroll to the top of the page and click **Save draft**.

With that, the final check box is green, and the **Publish app** button in the top-right of the console is now active! When you're satisfied with your app and store listing, go ahead and click it.

Your app listing status now updates to **Pending Publication**, like this:



It generally takes at least a few hours for your app to be available in Google Play.



Figure 11.18 Google Play

Congratulations on your first app in Google Play!

Reference:

<https://developer.android.com/reference/>

<https://www.raywenderlich.com/122114/android-app-distribution-tutorial-zero-google-play-store>

MODUL 12

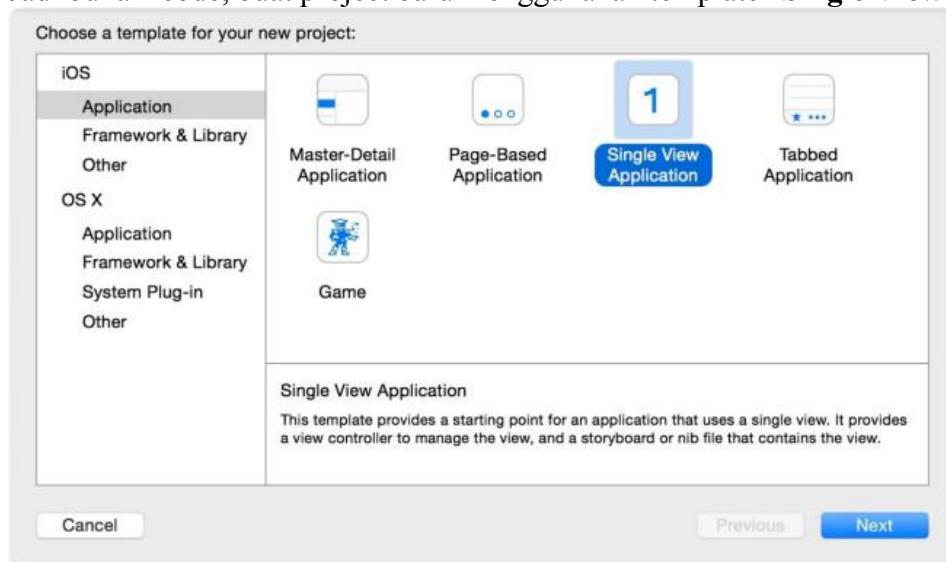
iOS

Membuat Aplikasi Tabel Sederhana di iOS

- ❖ Membuat Project SimpleTable

Aplikasi ini memang simple, dimana kita hanya akan menampilkan daftar dari nama restaurant.

Jadi buka Xcode, buat project baru menggunakan template “Single View Application”.



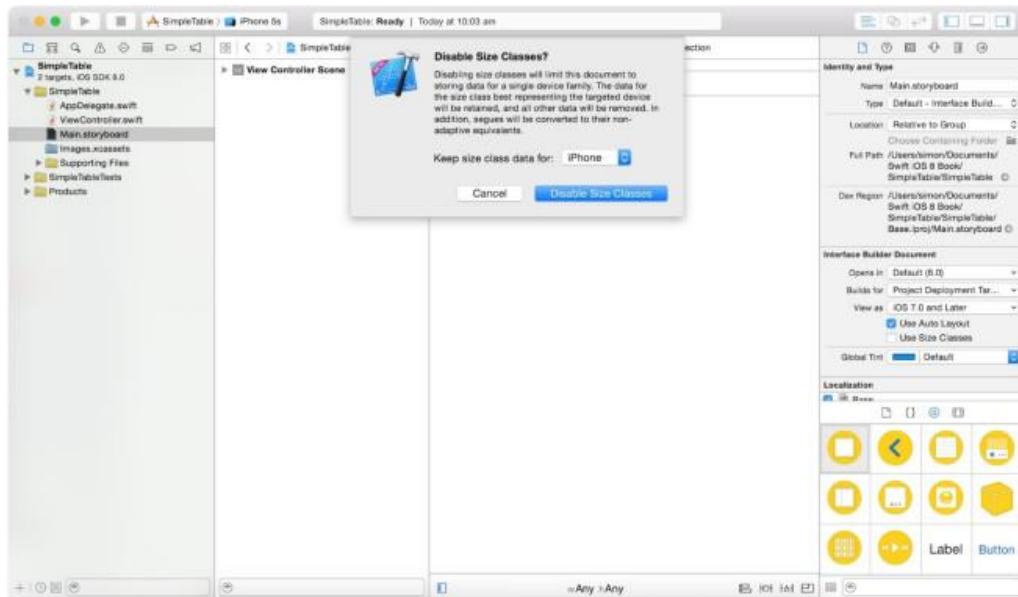
Klik Next untuk melanjutkan. Sekali lagi, isi semua form untuk keperluan project.

- **Product Name : SimpleTable** – Merupakan nama dari aplikasi.
- **Organization Name : AppCoda** – Merupakan nama dari organisasi dari project.
- **Organization Identifier : com.appcode** – Merupakan nama domain kita, soal ini bisa kita ganti sesuai dengan nama domain yang kita inginkan seperti “edu.self”.
- **Bundle Identifier : com.appcoda.SimpleTable** – Merupakan *identifier* unik dari aplikasi yang nantinya kita gunakan untuk submit aplikasi. Kita tidak perlu mengisi form ini, karena *Xcode* akan otomatis membuatnya untuk project kita.
- **Language : Swift** – *Xcode* 6 mendukung pengembangan menggunakan bahasa pemrograman Objective-C dan Swift. Nah untuk tutorial ini, kita hanya akan menggunakan Swift untuk pengembangan aplikasi iOS.
- **Device: iPhone** – Memilih “iPhone” untuk device project ini.
- **User Core Data: [jangan di checklist]** – Jangan menceklis pilihan ini, karena kita tidak membutuhkan Core Data untuk project simple ini. Kita akan menggunakan Core Data di pembahasan lain.

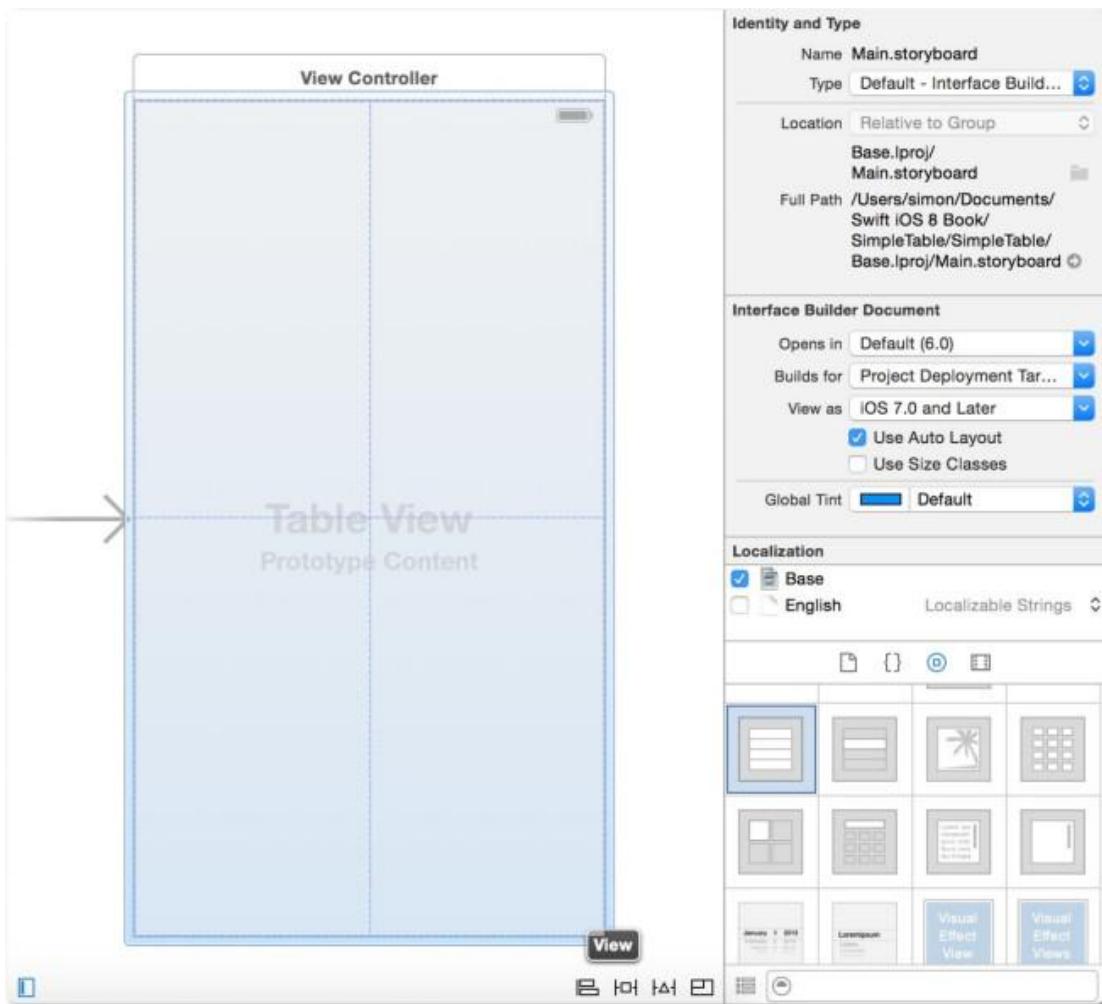
Klik Next untuk melanjutkan, *Xcode* akan bertanya dimana kita akan menyimpan project “Hello World”. Pilih folder manapun yang Anda ingikan sebagai lokasi penyimpanan project. Lalu klik continue.

Membuat Design dengan Storyboard

Pertama kali, kita akan membuat user interface dan menambahkan table view. Pilih “Main.storyboard” untuk mengganti tampilan Storyboard.

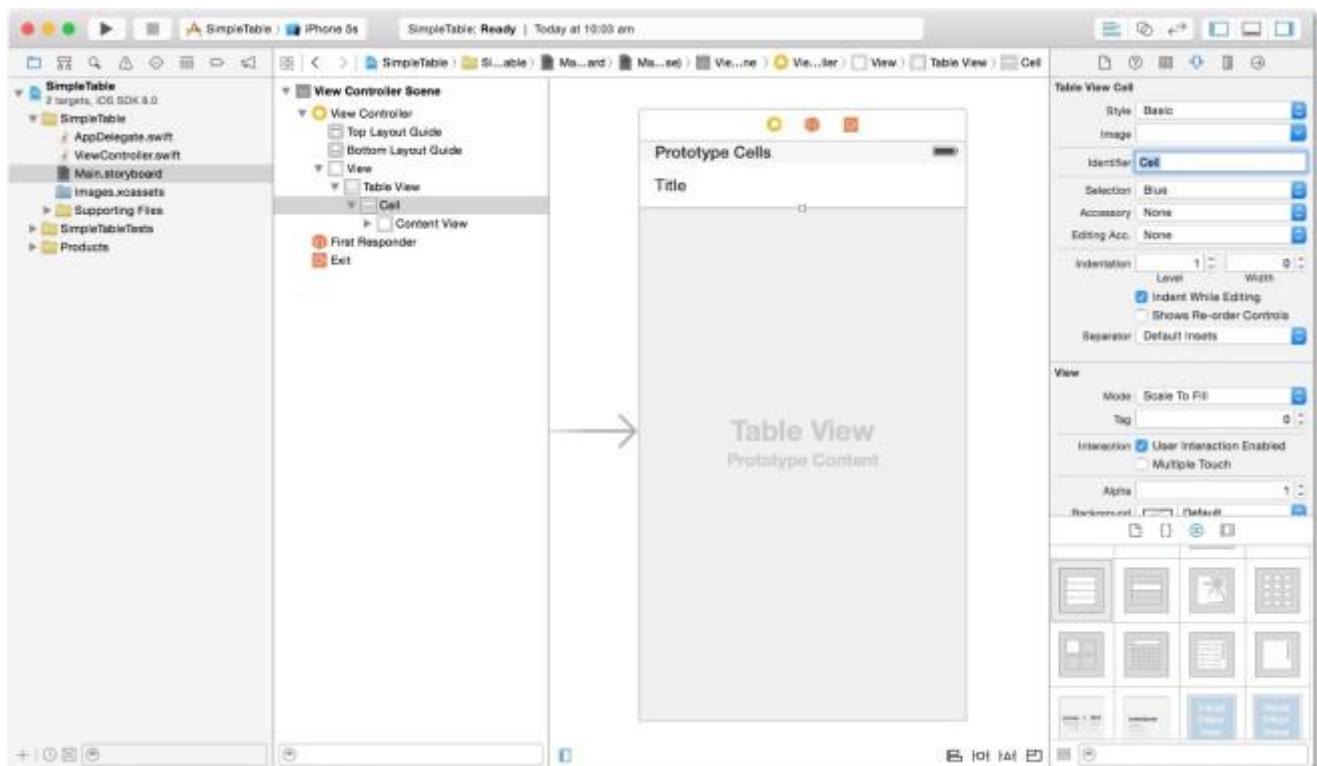


Sekali lagi kita tidak akan menggunakan Size Classes di project ini seperti pembahasan sebelumnya. Di File Inspector, jangan centang “Use Size Classes” yang ada di bawah Interface Builder Document. Apabila File Inspector dalam kondisi tidak terlihat, kita bisa menampilkannya dengan memilih **View>Utility>Show File Inspector**. Ketika kita men-disable size classes, Xcode akan memberitahu untuk memiliki target device yang akan digunakan. Nah untuk project ini pilih iPhone dan klik “Disable Size Classes” untuk mengkonfirmasinya. Maka setelah itu view controller akan terlihat seperti iPhone. Selanjutnya mari kita tambahkan object table view ke dalam view. Di Object Library, pilih object “Table View” dan pindahkan object tersebut ke view seperti gambar dibawah ini.

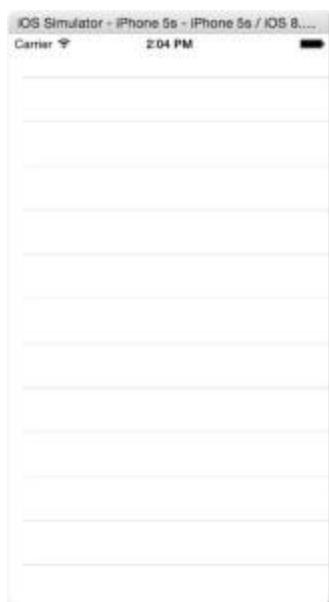


Selanjutnya pilih table view. Di dalam Attributes Inspector (jika tidak menemukan di Xcode Anda, pilih **View>Utilities>Show Attributes Inspector**), ganti nomor dari **Prototype Cell** dari 0 menjadi 1. Prototype Cell akan kelihatan di table view, dimana Prototype Cell ini memperbolehkan kita untuk mendesign layout dengan mudah dari table view cell kita. Didalamnya juga terdapat standar dari cell styles seperti basic, right detail, left detail dan juga subtitle. Di project ini kita menggunakan basic style.

Pilih cell dan buka Attributes Inspector. Ganti cell style menjadi **Basic**. Style ini sudah cukup bagus untuk di tampilan pada sebuah cell dengan text dan juga gambar. Selanjutnya, kondisikan identifier menjadi **Cell**, hal ini berfungsi untuk mengidentifikasi prototype cell. Kita akan menggunakannya dalam code.



❖ Jalankan Aplikasi Sebelum Memulai Coding



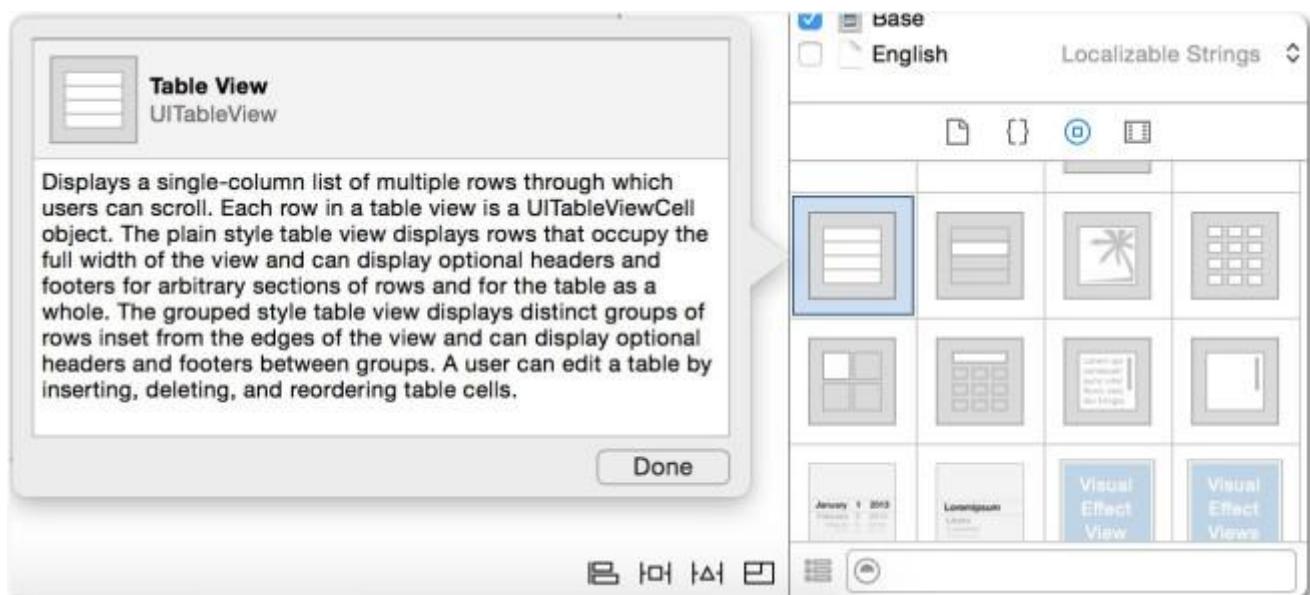
Sebelum kita memulai langkah selanjutnya, coba jalankan aplikasi menggunakan Simulator. Klik tombol “Run” untuk mem-build aplikasi dan mengetesnya. Harusnya sih tampilan aplikasi terlihat seperti gambar disamping.

Mudah bukan, nah kita telah membuat table view di aplikasi kita. Walaupun tidak menampilkan data apapun. Nantinya juga kita akan menulis beberapa code untuk mengisi data tabel.

❖ UITableView dan Protocols

Seperti yang telah disebutkan sebelumnya dimana kita harus mengerti dasar class dari iOS SDK. Class yang telah terorganisasi yang dinamakan dengan **frameworks**. Framework UIKit merupakan satu dan banyak framework yang digunakan untuk membuat aplikasi iOS.

Class ini nantinya menyediakan cara untuk membangun dan mengatur tampilan dari aplikasi kita. Semua objek yang ada di Object Library di storyboard telah disediakan oleh framework. Objek tombol yang kita gunakan pertama kali contohnya, Object Table View yang kita kerjakan sekarang ini juga merupakan framework. Istilah Table View yang kita gunakan merupakan class yang ada di UITableView. Kita bisa meng-klik item apapun yang ada di Object Library untuk menampilkan pop-over untuk menampilkan nama class.



Nah sekarang kita sudah punya gambaran tentang hubungan antara Tabel View dengan UITableView. Selanjutnya kita akan menulis beberapa code untuk menambahkan data pada tabel. Pilih file *ViewController.swift* yang ada di project navigator untuk membuka file kedalam editor pane. Tambahkan “UITableViewDataSource, UITableViewDelegate” setelah “UIViewController” untuk mengadopsi protocolnya.

Setelah kita menambahkan code setelah UIViewController, Xcode akan mendeteksi sebuah error. Xcode akan menampilkan tanda pemberitahuan berwarna merah ketika terjadi masalah. Klik tanda tersebut di sebalik kiri dari editor dan Xcode akan menandai baris dimana terjadi masalah di baris code tersebut. Pesan ini sangat membantu memberitahu masalah tapi tentu tidak akan memberikan kita solusi secara gamblang.



```
// ViewController.swift
// SimpleTable
//
// Created by Simon Ng on 7/8/14.
// Copyright (c) 2014 AppCoda. All rights reserved.
//

import UIKit

class ViewController: UIViewController, UITableViewDataSource, UITableViewDelegate {
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

The screenshot shows the Xcode interface with the file `ViewController.swift` open. A red error highlight is present over the line `override func didReceiveMemoryWarning()`, indicating that the type 'ViewController' does not conform to the protocol 'UITableViewDataSource'. This is a common mistake when starting to implement UITableView.

Jadi apa arti “**ViewController does not conform to protocol UITableViewDataSource”?** **UITableViewDelegates** dan **UITableViewDataSource** dianggap sebagai protocols di bahasa pemrograman Swift. Untuk menampilkan data di table view, kita harus mengkonfirmasi pemasang dari kebutuhan protocol di object (class ViewController), telah diimplementasikan untuk semua method.

Selanjutnya kita akan memerlukan dua method dari protocol UITableViewDataSource.

```
func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    // Return the number of rows in the section.
    return restaurantNames.count
}

func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) ->
    UITableViewCell {
    let cellIdentifier = "Cell"
    let cell = tableView.dequeueReusableCellWithIdentifier(cellIdentifier, forIndexPath:
        indexPath) as UITableViewCell
    // Configure the cell...
    cell.textLabel.text = restaurantNames[indexPath.row]
    return cell
}
```

Method pertama yang digunakan adalah untuk memberitahukan table view terkait jumlah baris. Kita hanya perlu memanggil method *count* untuk mendapatkan angkat dari item yang ada di array *restaurantNames*.

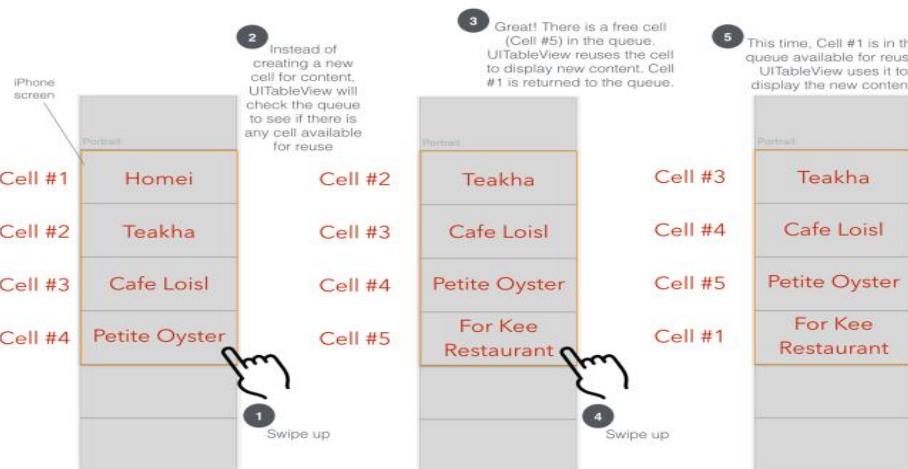
Method kedua yang akan kita panggil setiap saat adalah untuk menampilkan baris tabel. Menggunakan object **indexPath**, kita bisa mendapatkan baris sebelumnya (*indexPath.row*). Jadi apa yang kita lakukan adalah mendapatkan item yang telah terindex dari array *restaurantNames* dan menempatkannya di label text untuk ditampilkan.

Ok, tapi apa maksud dari *dequeueReusableCellWithIdentifier* di code bagian baris kedua? Method *dequeueReusableCellWithIdentifier* digunakan untuk menempatkan kembali *table cell* dari *queue* ke *identifier cell* yang spesifik.

Kita tentu ingin aplikasi *table-based* kita menjadi lebih cepat dan responsive ketika menangani ribuan baris data. Jika kita menyediakan sebuah cell baru untuk setiap baris data maka aplikasi kita akan menggunakan banyak memori sehingga mengakibatkan performa aplikasi menjadi lamban ketika user men-*scroll table view*. Perlu diingat bawah setiap cell membutuhkan biaya performat, khususnya ketika disediakan dalam waktu yang dekat.

Layar iPhone itu terbatas, jadi aplikasi kita akan menampilkan 1000 records, kemungkinan dilayar hanya akan tampil 10 record paling banyak. Jadi kenapa aplikasi kita tidak menyediakan 1000 cell table view dari pada menyediakan 10 tabel cell lalu menggunakannya kembali?

Hal ini dikarenakan akan menghemat banyak memory dan membuat table view terlihat lebih efisien. Dengan alasan performa itulah kenapa kita harus membuat table view yang dapat digunakan kembali untuk aplikasi ini.



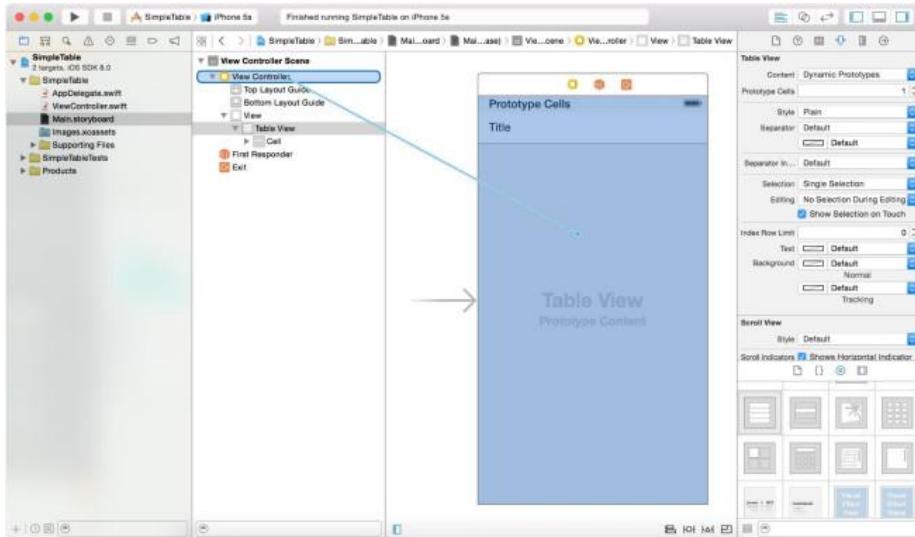
Sekarang coba tekan tombol “Run” untuk melakukan testing aplikasi. Dan oopss, aplikasi masih menampilkan table view kosong seperti gambar sebelumnya.

Menghubungkan DataSource dengan Delegate

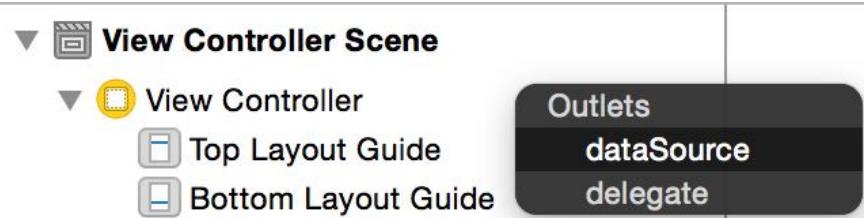
Semenjak class ViewController telah di adopsi oleh protocol UITableViewDelegate dan UITableViewDataSource, object UITableView yang ada di storyboard tidak memiliki

peranan lagi. Kita telah memberi tahu object UITableView bahwa ViewControoler adalah object delegate untuk data source.

Kembali lagi ke storyboard, pilih table view. Tekan dan tahan tombol key, klik table view dan pindahkan ke View Controller yang ada di Document Outline seperti gambar dibawah:

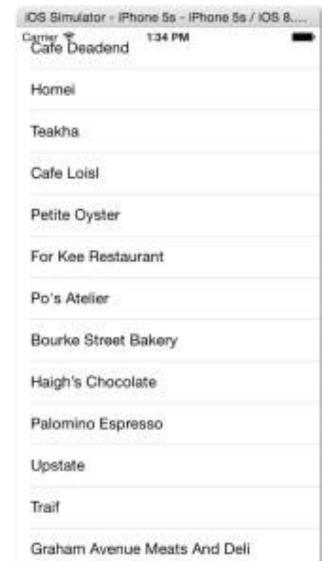


Lepas kembali tombolnya, maka akan muncul pop-over dan pilih “dataSource”. Ulangi langkah diatas dan buat koneksi dengan “delegate”.



Nah untuk memastikan keduanya telah terhubung, kita bisa meng-klik table view kembali. Lalu klik icon Connection Inspector yang ada di area Utility untuk membuka koneksi yang sudah ada sebelumnya. Alternatif lain adalah dengan meng-klik kanan tabel untuk memperlihatkan koneksinya.





❖ Uji Coba Aplikasi

Akhirnya aplikasi pertama kita telah siap untuk di testing. Dengan menekan tombol “Run” maka Simulator akan men-load aplikasi. Harusnya sih aplikasi kita telah menampilkan daftar nama restaurant seperti gambar disamping.

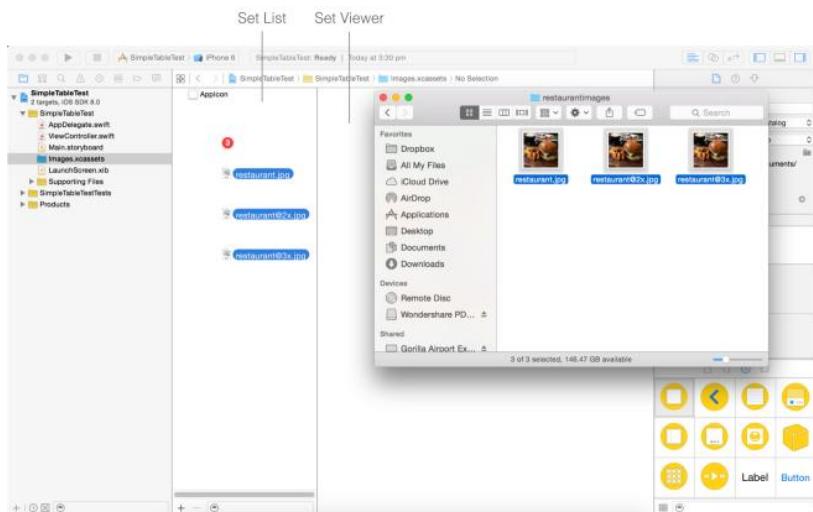
Menambahkan Gambar ke dalam Table View

Hanya menampilkan sebuah tabel dengan teks tentu terlihat kaku, nah bagaimana jika kita menambahkan gambar ke setiap barisnya?

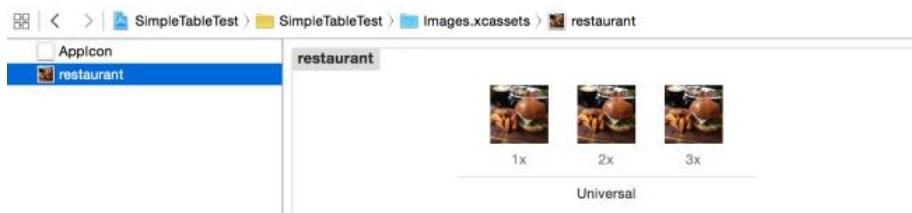
UITableView membuat hal ini menjadi mudah, kita hanya perlu menambahkan satu baris code untuk menambahkan gambar ke dalam baris yang ada di table view.

Langkah pertama coba download contoh gambarnya disini: . Ekstrak file zip nya yang berisikan tiga gambar. Setiap gambar memiliki gambar yang sama tapi dengan resolusi yang berbeda. Ketika kita mengembangka aplikasi iOS, sangat direkomendasikan kita mempersiapkan 3 versi gambar. Gambar pertama dengan ukuran #3x suffix yang digunakan untuk iPhone 6 Plus. Gambar kedua dengan ukuran @2x suffix yang digunakan untuk 4/4s/5/4s/6, dan gambar terakhir tanpa @suffix yang digunakan untuk device non Retina.

Project Xcode telah menyediakan catalog untuk contoh gambar ini (images.xcassets) untuk mengatir tipe dari gambar yang ada di project kita. Untuk meggunakan gambar coba unzip file, pilih images.xcassets untuk membuka assets catalog. Pindahkan tiga gambar yang ada di folder hasil ekstrak dan letakkan di folder list/viewer.



Xcode akan otomatis mendeteksi gambar yang bisa di tampilkan di Retina atau tidak. Satu gambar telah tersedia folder images set.



Sekarang edit file *ViewController.swift* dan tambahkan satu barus code ke method *tableView(_:cellForRowAtIndexPath:)*. Letakkan code dibawah ini sebelum “return cell” :

```
cell.imageView.image = UIImage(named: "restaurant")
```

Class *UIImage* disediakan oleh framwork *UIKit* dimana kita bisa membuat gambar dari file. Ini juga support terhadpa beberapa format gambar seperti *PNG*, *GIF*, dan *JPEG*. Nah sekarang kita sudah bisa melakukan testing kembali dan tekan tombol “Run”. Harusnya aplikasi akan menampilkan gambar seperti gambar disamping.

Menghilangkan Status Bar

Awalnya di iOS 7, view controller ditampilkan dalam bentuk full screen. Konten dari table view sekarang dilengkapi dengan status bar, dimana ini terlihat tidak begitu bagus untuk tampilan aplikasi. Nah kita perlu menghilangkannya, untuk itu kita mengatur tampilan dari status bar di per view controller basis. Jika kita tidak ingin menampilkan status bar di view controller, maka tambahkan beberapa baris code dibawah ini:

```
overridefuncprefersStatusBarHidden() ->Bool {  
    returntrue  
}
```

Tambahkan code diatas pada file *ViewController.swift* dan lakukan testing aplikasi kembali. Sepharusnya sekarang aplikasi telah menampilkan konten dengan mode fullscreen tanpa adanya status bar.

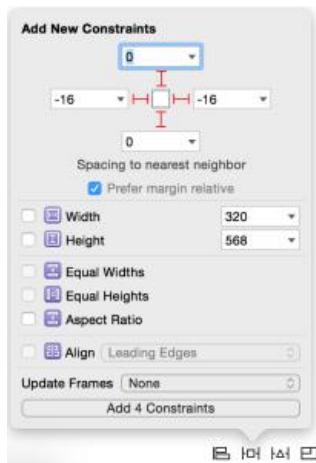
Memabahkan Auto Layout Constraints untuk UITableView

Sejauh ini aplikasi kita sudah terlihat lumayan ciamik, tapi aplikasi kita juga harus bisa responsive di iPhone 5/5s. Sudahkah Anda mencoba testing apikasi menggunakan device iPhone 4 dan melakukan rotasi ke mode landscape ketika menjalankan aplikasi?

Nah mungkin aplikasi kita nantinya akan terlihat seperti gambar dibawah ini.



Table view terlihat tidak tampil secara utuh di layar 3.5-inch, dan juga terjadi ketika kita merotasi device dalam mode landscape. Seperti pembahasan di postingan sebelumnya, kita bisa menggunakan auto layout untuk membuat aplikasi ini terlihat responsive di device manapun



Buka storyboard dan pilih table view, di menu Auto Layout klik tombol Pin untuk membuka menu Pin Tool. Pilih setiap garis merah yang ada di pop-up, setelah itu klik tombol “Add 4 Constrain” untuk menambahkan constrainnya. Disini kita menyediakan 4 spaceing constrain untuk setiap sisi dari UITableView. Nah sekarang kita sudah melakukan testing aplikasi kembali, aplikasi akan tampil lebih ciamik dan responsive tentunya.

❖ Memanggil Konten Web menggunakan UIWebView

Class UIWebView bisa memanggil konten web secara local atau remote. Jika kita memasukkan file html ke dalam aplikasi, kita juga bisa menggunakan class load web page. Biasanya kita harus membuat object NSURL, karena dengan object NSURLRequest maka kita bisa menggunakan method *loadRequest* dari UIWebView untuk merequest konten dari sebuah website. Berikut contoh potongan codenya:

```
if let url = NSURL(string:  
"http://www.wirasetiawan29.wordpress.com") {  
let request = NSURLRequest(URL: url)  
webView.loadRequest(request)  
}
```

Pada code diatas menginstruksikan web view untuk memanggil konten website secara remote. Seperti yang telah disebutkan sebelumnya bawah kita juga bisa membaca halaman website local yang disimpan didalam aplikais. Seperti file

HTMLS, kita bisa memanggilnya dengan menggunakan object NSURL melalui code berikut:

```
let url = NSURL(fileURLWithPath: "about.html")
```

UIWebView akan memanggil konten web secara local, dan ini tetap akan bekerja walupun tidak adanya koneksi internet.

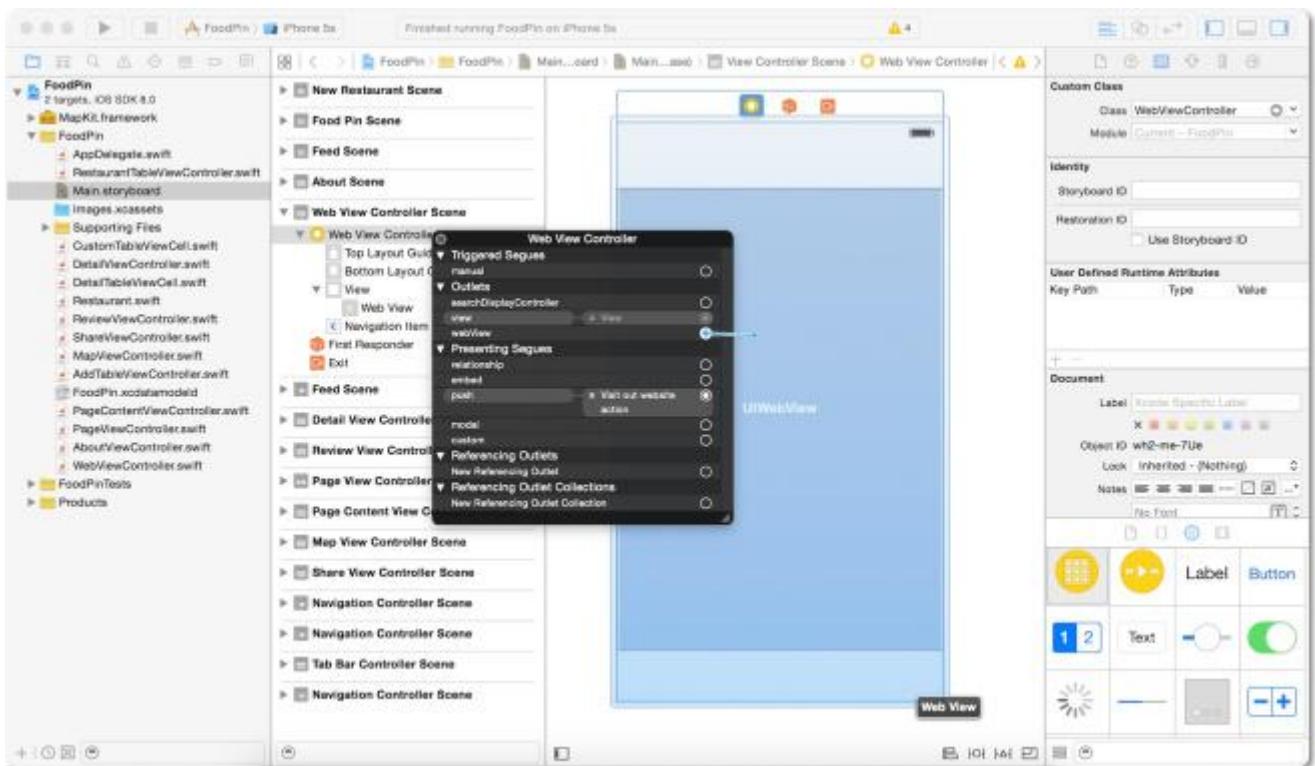
Dengan mengertinya tentang class UIWebView, mari kita lanjutkan implementasi WebViewController. Pertama buat variabel outlet untuk web view di class WebViewController :

```
@IBOutlet weak var webView:UIWebView!
```

Pada method *viewDidLoad*, tambahkan code dibawah ini untuk memanggil website. Kita bebas mengganti URL dari websitenya:

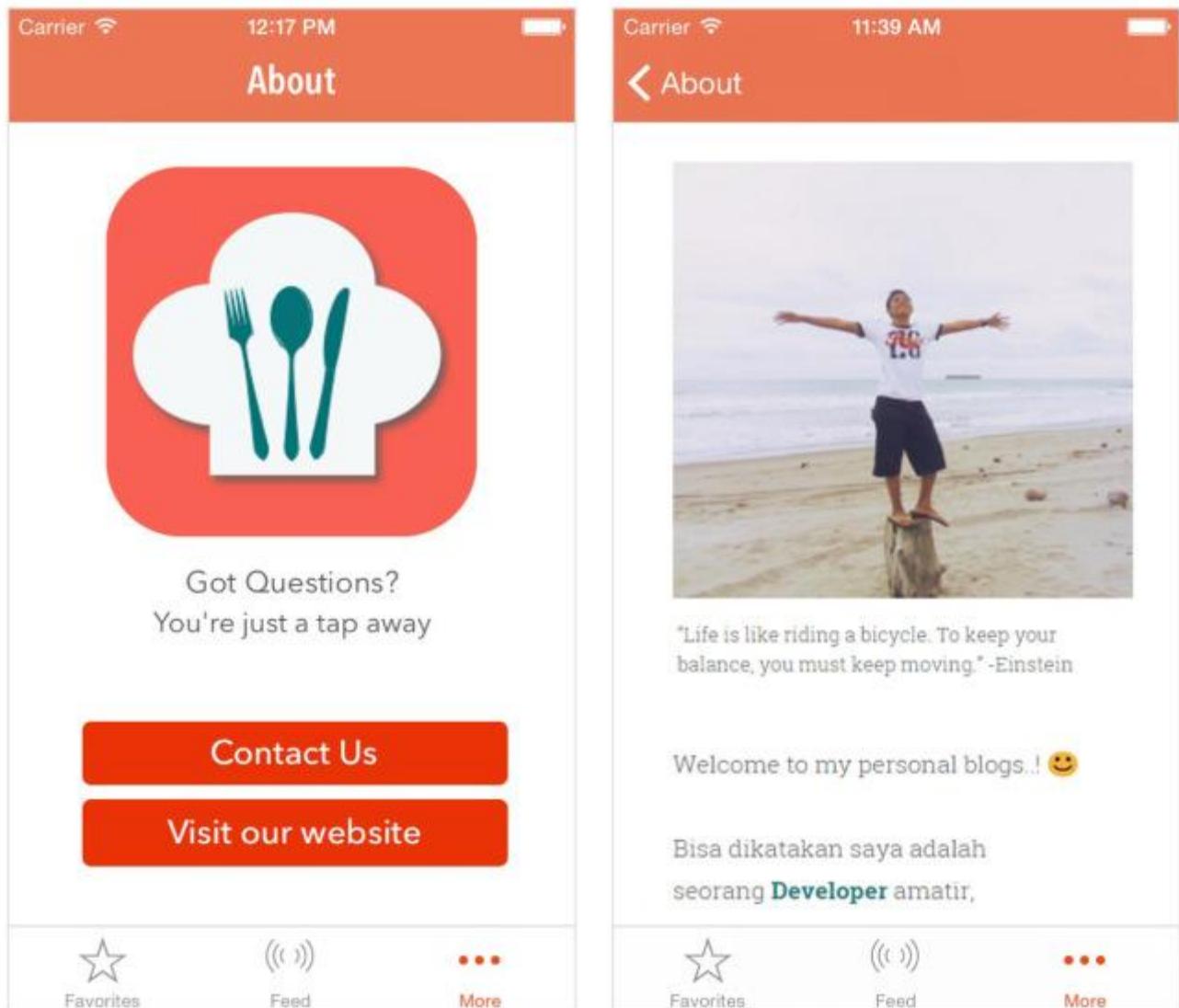
```
if let url = NSURL(string:  
"http://www.wirasetiawan29.wordpress.com/about") {  
let request = NSURLRequest(URL: url)  
webView.loadRequest(request)  
}
```

Terakhir, kembali ke Interface Builder dan buat koneksi dengan outlet webView. Klik kanan Web View Controller di Document Outline. Hubungkan outlet webView dengan object UIWebView di view.



Membuat koneksi dengan outlet web view

Sekarang kita sudah siap untuk menjalankan aplikasi dan melakukan testing. Pilih tab More dan klik tombol “Visit our Website”. Website akan ditampilkan pada halaman web view. Seperti yang terlihat digambar dibawah ini:



Halaman About dan halaman web view ketika mengunjungi website

❖ Menggunakan MFMailComposeViewController

Sekarang kita akan mengimplementasikan tombol Contact Us. Ketika user menekannya, aplikasi akan menampilkan tampilan email dan kita bisa membuat email dengan aplikasi tersebut.

Seperti yang telah disebutkan sebelumnya, kita akan menggunakan class MFMailComposeViewController untuk membuat tampilan email. Hampir sama dengan UIWebView, class MFMailComposeViewController ini menghandle fitur

email yang ada di aplikasi. Yang kita butuhkan adalah menambahkan framework MessageUI, dan membuat contoh MFMailComposesViewController untuk memanggil method *presentViewController*. Berikut contoh codenya:

Class itu bertanggung jawab terhadap method *canSendMail* yang berfungsi untuk mengecek jika device bersedia untuk mengirimkan email. Kita harus memanggil method ini sebelum percobaan membuat tampilan membuat email.

MailComposeDelegate menggunakan object delegate untuk menghilangkan tampilan mail. Object delegate harus menggunakan protocol MFMailComposeViewControllerDelegate dan mengimplementasikan method *mailComposeController(_:didFinishWithResult:eror:)*. Method ini dipanggil ketika user menakan tombol Send/Cancel pada tampilan Mail

❖ Mengimplementasikan tombol Contact Us

Sekarang kita akan mengimplementasikan tombol Contact Us. Pertama, tambahkan statement berikut diawal file AboutViewController.swift:

```
import MessageUI
```

Lalu tambahkan method action di AboutViewController:

Code diatas sebenarnya hampir sama dengan yang dibawah dipostingan sebelumnya, kita hanya menambahkan dua baris code untuk memodifikasi warna navigation bar dan status bar style.

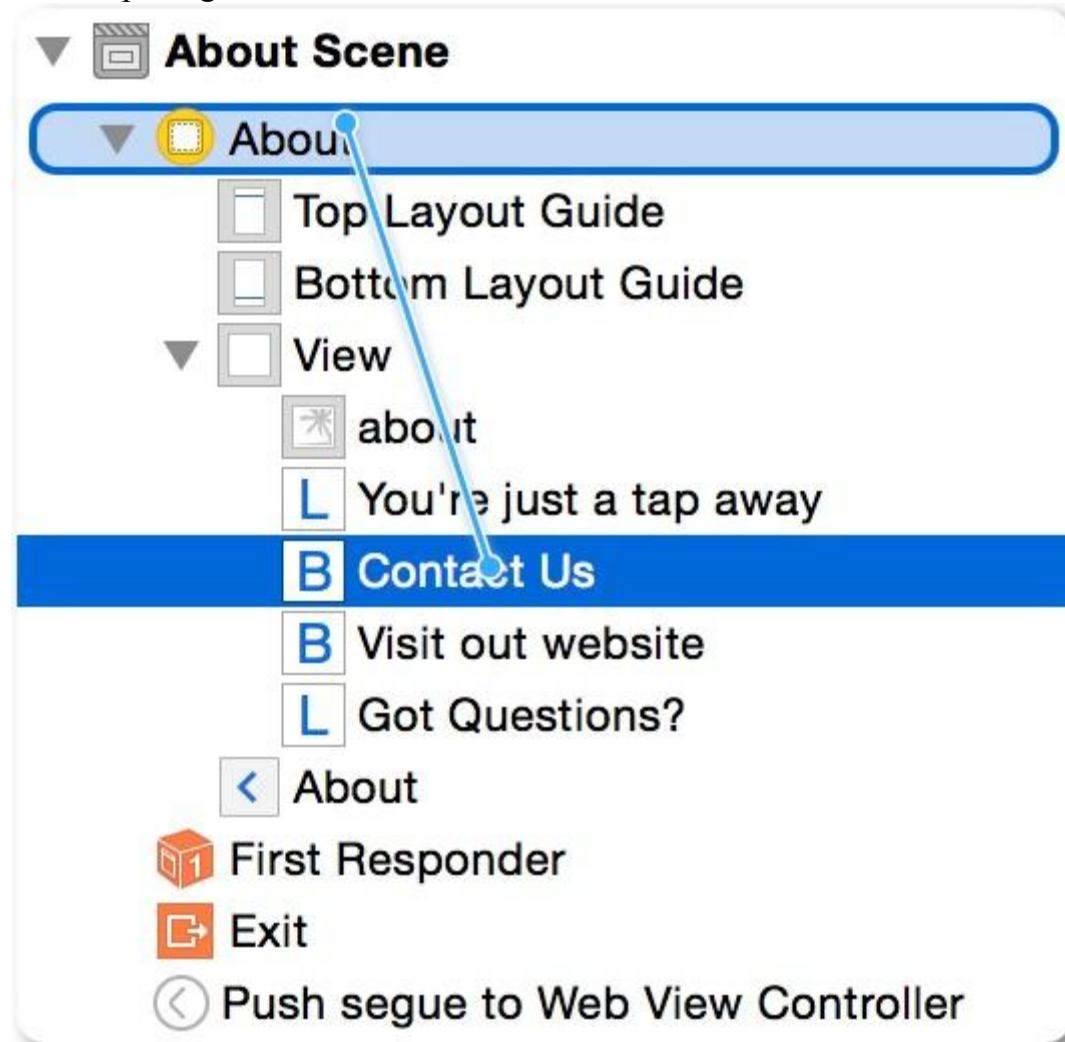
Class ini ditandai sebagai object delegate dari mail compose view controller. Jadi nantinya akan membutuhkan protocol MFMailComposeViewControllerDelegate dan UINavigationControllerDelegate. Jadi ganti deklrasi class tersebut menjadi:

```
class AboutViewController: UIVi
```

Lalu implementasikan method yang dibutuhkan seperti code berikut:

Seperti yang telah dibahas sebelumnya, method ini ditetapkan sebagai protocol `MFMailComposeViewControllerDelegate` dan akan otomatis dipanggil ketika tampilannya ditutup (user membantalkan operasinya melalui tombol cancel).

Sebelum method `sendMail` digunakan, kita harus membuat koneksi dengan tombol Contact Us. Ergi storyboard, tekan dan tahan tombol contro dan pindahkan dari tombol Contact Us ke About View Controller di Document Outline. Seperti yang terlihat pada gambar dibawah ini:



Membuat koneksi dengan action `sendEmail`

Nah itu dia, kita sudah bisa menjalankan aplikasi, dan seharunya aplikasi sudah bisa menampilkan halaman mail ketika kita menekan tombol Contact Us. Seperti yang terlihat digambar dibawah ini.



Tampilan Membuat Email

Well mungkin cukup sekian pembahasan kali ini, oya untuk contoh code bisa didownload

disini: <https://www.dropbox.com/s/nu77onxoj60aqvj/FoodPinWebEmail.zip?dl=0>

Semoga bermanfaat ya Guys, See Yaa ...

Iklan

❖ Implementasi MAP Kit Framework di iOS – FoodPin Apps

30 NOVEMBER, 2015 ~ WIRA SETIAWAN

MAP Kit Framework merupakan fitur yang bertanggung jawab untuk menampilkan maps, navigasi, menambahkan keterangan untuk lokasi yang

spesifik, menambahkan overlays pada maps yang sudah ada, dsb. Dengan framework ini kita bisa menanamkan fungsi tampilan map kedalam aplikasi kita secara full tanpa coding.



Nah pada pembahasan kali ini, kita akan membahas tentang framework ini, mulai dari:

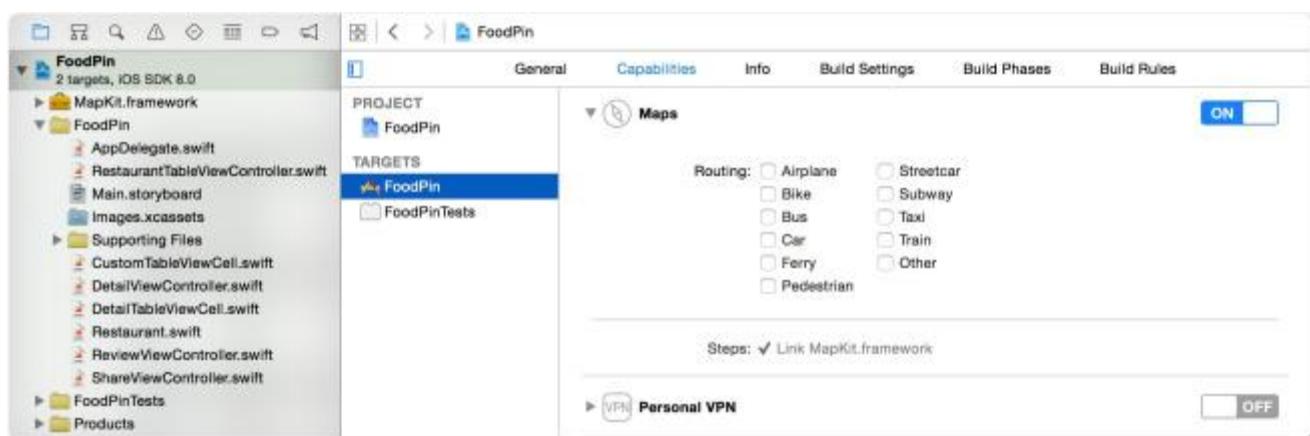
- Bagaimana menanamkan sebuah map pada layout
- Bagaimana menerjemahkan alamat menjadi kordinat menggunakan Geocoder
- Bagaimana menambahkan sebuah pin untuk keterangan pada map
- Bagaimana memodifikasi keterangan pada map

Untuk memperjelas pembahasan mengenai Map Kit Framework, kita akan menambahkan fitur map di halaman detail view dari aplikasi FoodPin. Setelah mengimplementasikannya, aplikasi akan membawa kita ke sebuah map yang diberi tanda dari lokasi restaurant yang kita pilih dengan sebuah anda pin.

Keren kan? Let's get started.

❖ Menggunakan Map Kit Framework

Biasanya Map Kit Framework tidak di bundle dalam project Xcode, jadi untuk menggunakannya kita harus mengaktifkan fitur Maps. Di Project Navigator, pilih project FoodPin dan kemudian pilih target FoodPin. Kita bisa mengaktifkan fitur Maps di bagian bawah Capabilities dengan cara menghidupkan tombol On pada fitur Maps seperti yang terlihat pada gambar dibawah ini. Dengan diaktifkannya pilihan Maps, Xcode akan otomatis menambahkan MapKit Framework pada project kita.



Enable Maps di project Xcode

❖ Menambahkan Tampilan Maps pada Aplikasi

Yang kita lakukan selanjutnya adalah menambahkan icon maps untuk field alamat pada halaman Detail View. Ketika user menekan icon map, aplikasi akan berpindah ke map view controller dan menampilkan lokasi dari restaurant.

Jadi buka Main.storyboard dan pindahkan sebuah tombol dari Object Library ke cell tabel dari Detail View Controller. Beri nama tombolnya dengan Map, jika ingin membuatnya lebih cantik kita bisa mengganti warna background dan fontnya.

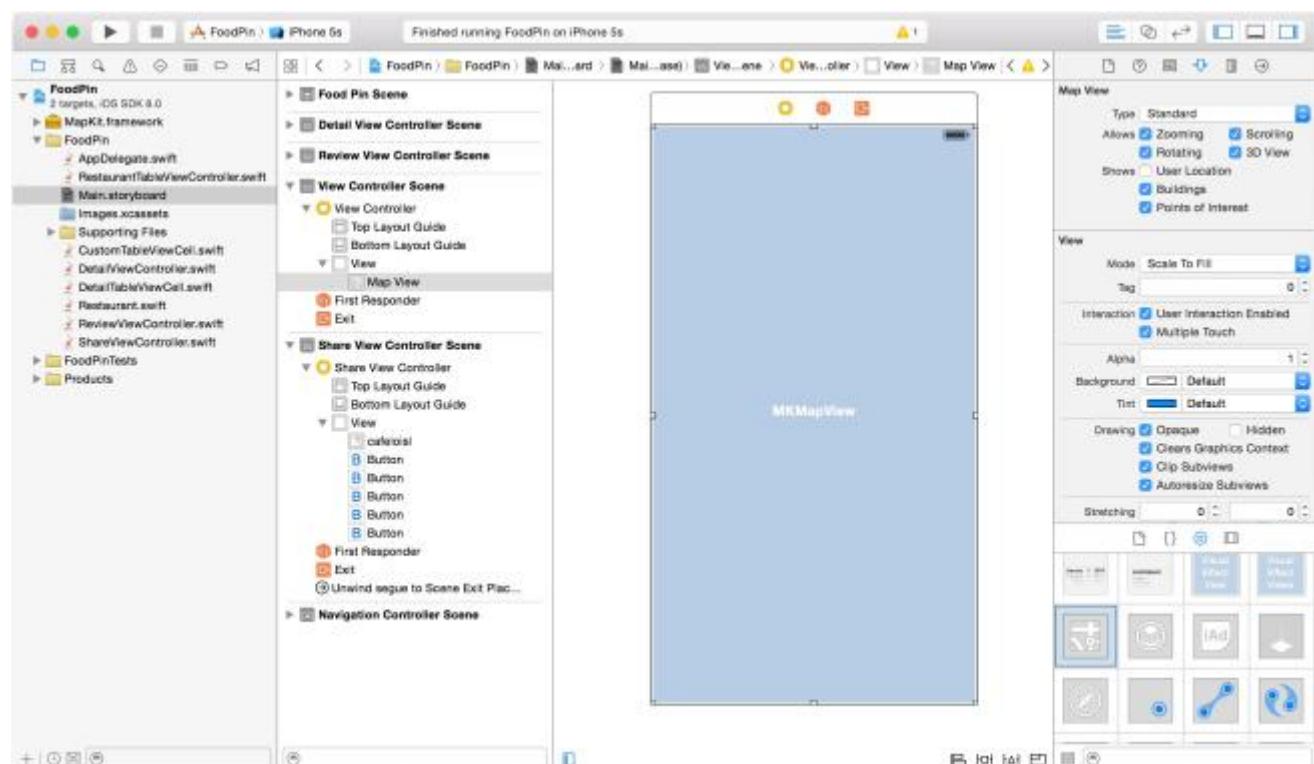
ya adalah menambahkan icon maps untuk field alamat pada halaman Detail View. Ketika user menekan icon map, aplikasi akan berpindah ke map view controller dan menampilkan lokasi dari restaurant.

Jadi buka Main.storyboard dan pindahkan sebuah tombol dari Object Library ke cell tabel dari Detail View Controller. Beri nama tombolnya dengan Map, jika ingin membuatnya lebih cantik kita bisa mengganti warna background dan fontnya.



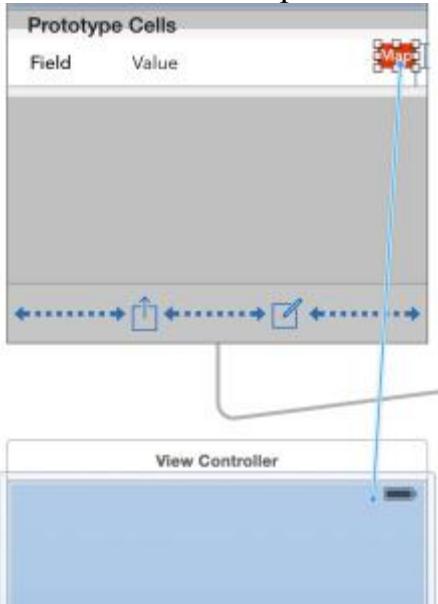
Menambahkan tombol map pada prototype cell

Untuk membuat tampilan map, tambahkan view controller lainnya dari Object Library ke storyboard, dan diikuti dengan penambahan object MapKit View ke View Controller.



Menambahkan tampilan map

Sekarang tahan tombol control dan pindahkan dari tombol Map ke map view controller yang baru untuk membuat segue baru. Pilih Push sebagai tipe dari seguenya, nah setelah segue dibuat, isi identifiernya dengan **showMap** yang ada di bawah Attribute Inspector.



Menghubungkan tombol map dengan map view controller

Jika kita meng-compile dan menjalankan aplikasi, kita akan melihat dua tombol map yang ada di Detail View. Tekan salah satu dari map maka akan menampilkan fungsi map. Inilah kelebihan dari Map Kit Framework, yaitu tanpa menggunakan code apapun kita sudah bisa menanamkan map pada aplikasi kita.

Jika kita melihat kembali ke Attribute Inspector, kita harus mengatur beberapa pilihan seperti memperbesar dan scrolling untuk memodifikasi fitur maps. Kita juga bisa mengganti tipe map dari Standard menjadi Satellite atau Hybrid. Pilihan lainnya menggunakan "User Location" untuk menampilkan lokasi terakhir dari user. Tapi kita belum selesai sampai tahap ini, karena aplikasi baru bisa menampilkan maps secara full. Padahal yang diinginkan adalah map yang menampilkan pin point dari lokasi restaurant yang dipilih.

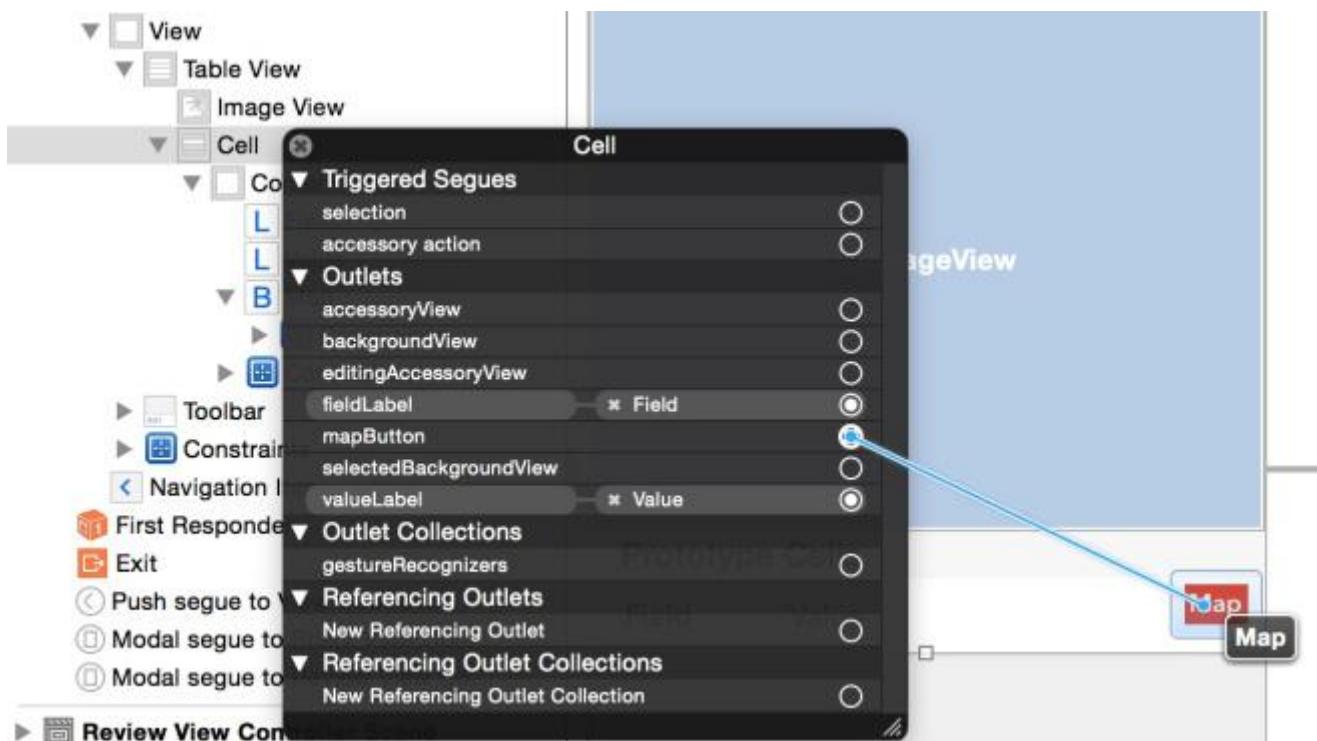
❖ Menghapus Tombol Map yang Duplikat

Sebelum kita melanjutkan implementasi dari map, kita harus menghapus salah satu tombol map yang ada di halaman detail view.

Pertama deklarasikan variabel outlet baru untuk tombol map di file DetailTableViewCell.swift:

```
@IBOutlet var mapButton: UIButton!
```

Pergi storyboard, lalu klik kanan cell yang ada di Document Outline untuk membuat koneksi antara tombol map di storyboard dengan variabel outlet mapButton.



Membuat koneksi antara tombol map dengan variabel outlet

Selain field location, tombol map dan field lainnya harus di hilangkan. Buka file DetailViewController.swift dan ganti method `tableView(_:cellForRowAtIndexPath:)` dengan code dibawah ini:

```
func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) ->
```

```
    UITableViewCell {
```

```
        let cell = tableView.dequeueReusableCellWithIdentifier("Cell", forIndexPath: indexPath)
```

```
        as DetailTableViewCell
```

```

cell.backgroundColor = UIColor.clearColor()

// Configure the cell...

cell.mapButton.hidden = true

switch indexPath.row {

case 0:

    cell.fieldLabel.text = "Name"

    cell.valueLabel.text = restaurant.name

case 1:

    cell.fieldLabel.text = "Type"

    cell.valueLabel.text = restaurant.type

case 2:

    cell.fieldLabel.text = "Location"

    cell.valueLabel.text = restaurant.location

    cell.mapButton.hidden = false

case 3:

    cell.fieldLabel.text = "Been here"

    cell.valueLabel.text = (restaurant.isVisited) ? "Yes, I've been here before" : "No"

default:

    cell.fieldLabel.text = ""

    cell.valueLabel.text = ""

}

return cell
}

```

[view rawTableView6.swift](#) hosted with [GitHub](#)

Pada code diatas, pasangkan *hidden* property dari mapButton menjadi *true* untuk semua field kecuali field location. Ya done, coba compile dan jalankan aplikasi

untuk melakukan testing. Seharusnya tombol duplikat yang tadi tampil sudah tidak ada lagi.

❖ Mengkonversi Alamat menjadi Kordinat menggunakan Geocoder

Untuk memberi tanda sebuah lokasi di map, kita tidak bisa hanya menggunakan alamat aslinya. Map Kit tidak akan bekerja dengan menggunakan data alamat asli, karena map tersebut membutuhkan kordinat berupa latitute dan longitude untuk menemukan titik yang tepat dari globe.

Framework tersebut bertanggung jawab terhadap object Geocoder dimana nantinya developer dapat mengkonversi sebuah alamat dan bangunan menjadi kordinat global, proses ini biasanya dinamakan dengan geocoding. Dan sebaliknya, kita bisa menggunakan Geocoder untuk mengkonversi latitude dan longitude kembali menjadi sebuah alamat, proses ini dinamakan dengan reverse geocoding.

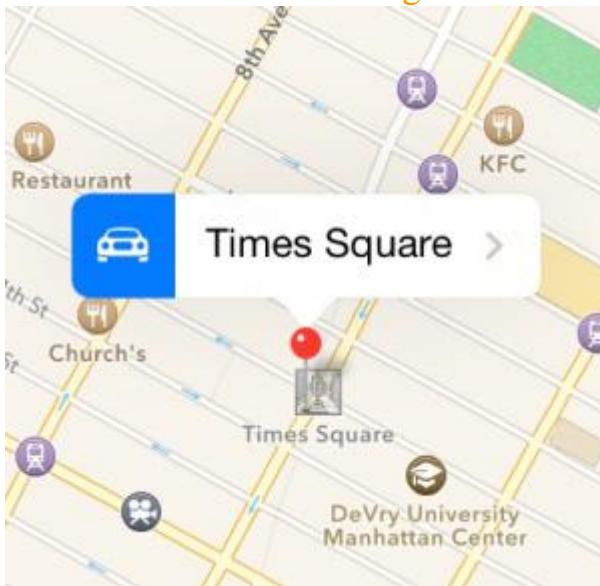
Untuk memulai sebuah reques geocoding menggunakan class CLGeocoder, semua yang kita butuhkan adalah membuat instansi dari CLGeocoder, dengan memanggil method *geocodeAddressString* sebagai parameter alamatnya. Berikut contohnya:

```
let geoCoder = CLGeocoder()  
geoCoder.geocodeAddressString("524 Ct St, Brooklyn, NY  
11231", completionHandler: { placemarks, error in  
    // Process the placemark  
})
```

Itu tidak menunjukkan fotmat dari string alamat, method mengirimkan data lokasi yang spesifik ke server geocoding secara asynchronously. Alamat kemudian diuraikan dan mengembalikannya dalam bentuk array dari object placemark. Dengan object placemark yang merupakan instansi dari class *CLPlacemark*, kita bisa dengan mudah mendapatkan kordinat global dari alamat menggunakan code dibawah ini:

```
let coordinate = placemark.location.coordinate
```

❖ Menambahkan keterangan kedalam Map



Keterangan

Sekarang kita telah mengerti ide dasar dari *Geocoder* dan tahu bagaimana membuat kordinat global dari alamat, sekarang kita akan membuat sebuah penanda kordinat pada map. Untuk membuatnya, MAP Kit framwework menyediakan keterangan sebagai penanda dari lokasi yang spesifik tersebut.

Pin keterangan yang kita lihat di aplikasi Map adalah salah satu contoh keterangannya. Bagi seorang developer, kita harus mengetahui bahwa keterangan tersebut sebenarnya terdiri dari dua object:

- Object Keterangan – yang merupakan data yang ditampilkan seperti nama dari alamat.
- Object Tampilan – yang merupakan object untuk merepresentasikan gambar visual dari keterangan. Gambar pin sebagai contohnya. Jika kita ingin menampilkan keterangan dari lokasi yang kita tandai, kita harus membuat gambar visual dari keterangan sendiri.

Map Kit framework memiliki standar object dan view keterangan sendiri. Jadi kita tidak perlu membuatnya lagi, kecuali ingin memodifikasi gambarnya.

Umumnya standar dari keterangan di map, adalah seperti code berikut:

```
let annotation = MKPointAnnotation()  
annotation.title = "Times Square"  
annotation.coordinate = placemark.location.coordinate  
self.mapView.showAnnotations([annotation], animated:  
true)  
self.mapView.selectAnnotation(annotation, animated:  
true)
```

Class *MKPointAnnotation* merupakan class standar untuk mengimplementasikan protocol MKAnnotation. Class itu berisikan judul, sub judul untuk menampilkan sebuah callout bubble yang sangat simple. Dengan menspesifikasikan object keterangan, kita bisa memanggil method *showAnnotation* dari object mapView untuk di letakkan di pin dari map. Method tersebut cukup pintar untuk mencari denah yang tepat untuk keterangan tersebut.

Biasanya callout bubble tidak akan tampil di map sampai user menekan tombol pin. Jika kita ingin menampilkannya tanpa harus melalui interaksi dari user, kita harus menggunakan method *selectAnnotation*.

❖ Mendefinisikan Lokasi di Aplikasi FoodPin

Setelah memperkenalkan pengetahuan dasar dari annotations/keterangan dan geocoding, marik kita kembali lagi ke project FoodPin. Pergi ke Main.storyboard, lalu kita buat class MapViewController untuk view controller yang telah dibuat.

Di Project Navigator, klik kanan folder FoodPin dan pilih “New File ..”. Lalu buat class baru menggunakan class Cocoa Touch dan beri nama classnya dengan **MapViewController**. Pastikan subclassnya adalah UIViewController dan simpan file tersebut.

Ketika itu telah selesai, tambahkan statement import untuk bisa menggunakan MAP Kit Framework.

```
import MapKit
```

Selanjutnya deklarasikan variabel outlet untuk map view dan variabel lainnya untuk restaurant yang dipilih:

```
@IBOutlet var mapView:MKMapView!  
var restaurant:Restaurant!
```

Variabel outlet digunakan untuk membuat sebuah koneksi dengan map view yang ada di storyboard. Buka Interface Builder dan pilih map view controller, dibawah Identity Inspector, isi custom classnya dengan MapViewController.

Kemudian buat koneksi antara variabel outlet dengan MKMapView.



Membuat koneksi antara MKMapView dengan variabel outlet MapView

Untuk menambahkan keterangan pada map, perbarui method *viewDidLoad* dengan code dibawah ini:

```
override func viewDidLoad() {  
    super.viewDidLoad()  
  
    // Convert address to coordinate and annotate it on map  
  
    let geoCoder = CLGeocoder()  
  
    geoCoder.geocodeAddressString(restaurant.location, completionHandler: { placemarks,  
        error in  
  
        if error != nil {  
  
            println(error)  
        }  
    })  
}
```

```

        return

    }

    if placemarks !=nil&& placemarks.count>0 {

        let placemark = placemarks[0] as CLPlacemark

        // Add Annotation

        let annotation =MKPointAnnotation()

            annotation.title=self.restaurant.name

            annotation.subtitle=self.restaurant.type

            annotation.coordinate= placemark.location.coordinate

        self.mapView.showAnnotations([annotation], animated: true)

        self.mapView.selectAnnotation(annotation, animated: true)

    }

})

}

```

[view raw](#)[viewDidLoad2.swift](#) hosted with [GitHub](#)

Pertama kita konversikan alamat dari restaurant yang dipilih menjadi kordinat menggunakan Geocoder. Di banyak kasus, array dari alamat harus berisikan satu inputan, jadi kita hanya memilih element pertama dari array.

Ada satu hal lagi yang ketinggalan sebelum kita melakukan testing aplikasi, yaitu kita belum mengirimkan informasi restaurant yang terpilih ke map view controller. Pada file DetailViewController, tambahkan method *prepaterForSegue* berikut:

```

override func prepareForSegue(segue: UIStoryboardSegue,
sender: AnyObject!) {
if segue.identifier == "showMap" {
let destinationController =
segue.destinationViewController as MapViewController
destinationController.restaurant = restaurant
}

```

}

}

Ok selanjutnya mari kita lakukan testing, dan buka halaman Map di Detail View, dan seharunya aplikasi kita telah menampilkan halaman map seperti gambar dibawah ini.



Keterangan di MAP

❖ Menambahkan Gambar pada Keterangan Map

Nah akan lebih bagus lagi jika aplikasi kita bisa menampilkan gambar dari restaurant di penanda map yang ada di halaman detail view bukan?

Untuk menambahkan thumbnail atau gambar di keterangan, kita harus memodifikasi tampilan keterangan tersebut. Jadi setiap kali map view akan menampilkan keterangan, maka map view akan memanggil method `mapView(_:viewForAnnotation:)` dari object delegate:

```
optional func mapView(_ mapView: MKMapView!,  
viewForAnnotation annotation: MKAnnotation!) ->  
MKAnnotationView!
```

Diawal kita tidak mengimplementasikan method ini, karena kita hanya menggunakan tampilan keterangan map bawaan. Nah disini kita ingin mengimplementasikan method tersebut gambar dari restaurant.

Buka file MapViewController.swift dan implementasikan MKMapViewDelegate berikut:

```
class MapViewController: UIViewController,  
MKMapViewDelegate
```

Pada method `viewDidLoad`, tambahkan code dibawah ini untuk mendefinisikan delegate dari mapView:

```
mapView.delegate = self;
```

Selanjutnya implementasikan

method `mapView(_:viewForAnnotation:)` menggunakan code dibawah ini:

```
func mapView(mapView: MKMapView!, viewForAnnotation annotation: MKAnnotation!) ->  
MKAnnotationView! {  
    let identifier = "MyPin"  
  
    if annotation.isKindOfClass(MKUserLocation) {  
  
        return nil  
  
    }  
  
    // Reuse the annotation if possible
```

```
var annotationView = mapView.dequeueReusableCellWithIdentifier(identifier)

if annotationView ==nil {

    annotationView =MKAnnotationView(annotation: annotation, reuseIdentifier:
identifier)

    annotationView.canShowCallout=true

}

let leftIconView =UIImageView(frame: CGRectMake(0, 0, 53, 53))

leftIconView.image=UIImage(named: restaurant.image)

annotationView.leftCalloutAccessoryView= leftIconView

return annotationView

}
```

[view rawmapView.swift](#) hosted with [GitHub](#)

Nah kita akhirnya selesai, selanjutnya tekan tombol Run dan jalankan aplikasi. Pilih salah satu restaurant dan arahkan ke detail view yang menampilkan map. Seharsunya tampilan aplikasi akan seperti gambar dibawah ini.



Keterangan dengan gambar

Well mungkin cukup sekian pembahasan kali ini, oya untuk contoh code bisa di download disini:

<https://www.dropbox.com/s/767b4u3n2qv2enj/FoodPinMap.zip?dl=0>