```
package com.test.AndroidApplicationFramewrok;

import android.content.ContentValues;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
import android.widget.ScrollView;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.RadioButton;
import android.widget.ArrayAdapter;

import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;
import android.widget.Toast;

import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserFactory;
import org.xmlpull.v1.XmlPullParserException;

import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;


@SuppressWarnings("ALL")
public class MainActivity extends Activity
{
   String strApplicationName = null;
   String strScreenName = null;
   ArrayList<Fields> fieldsList = null;
   Button btnSave = null;
   Button btnView = null;
   Button btnExit = null;

   SQLiteDatabase db;
   LinearLayout ll = null;

   ListView lstFields = null;

   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
```

```
    super.onCreate(savedInstanceState);

    try
    {
       populateMainView();
    }
    catch(Exception ex)
    {
       ex.printStackTrace();
    }
}

void populateMainView()
{
    ScrollView sv = new ScrollView(this);
    sv.setLeft(10);
    ll = new LinearLayout(this);
    ll.setGravity(ll.TEXT_ALIGNMENT_GRAVITY);
    ll.setOrientation(LinearLayout.VERTICAL);
    sv.addView(ll);

    ll.setDividerPadding(5);

    TextView textViewControl = null;
    EditText editTextControl = null;
    RadioButton radioButtonControl = null;
    Spinner spinnerControl = null;

    //Read the Fields xml file
    readXML();



    //Set Application Title
    textViewControl = new TextView(this);
    textViewControl.setText(strApplicationName);
    textViewControl.setLeft(10);
    textViewControl.setGravity( View.TEXT_ALIGNMENT_CENTER );
    textViewControl.setTextAlignment(View.TEXT_ALIGNMENT_CENTER);
    textViewControl.setTextSize(20);
    ll.addView(textViewControl);



    //Set Screen heading
    textViewControl = new TextView(this);
    textViewControl.setText(strScreenName);
    textViewControl.setLeft(10);
    textViewControl.setTextAlignment(View.TEXT_ALIGNMENT_CENTER);
    textViewControl.setTextSize(15);
    ll.addView(textViewControl);
```

```
LinearLayout fieldLL = null;

//Create all the controls for fields dynamically
if ( fieldsList != null && fieldsList.size() > 0 )
{
   for(Fields currentField : fieldsList)
   {
      fieldLL = new LinearLayout(this);
      fieldLL.setOrientation(LinearLayout.HORIZONTAL);
      fieldLL.setLeft(10);
      if (currentField.fieldType.equalsIgnoreCase("edittext"))
      {
         textViewControl = new TextView(this);
         textViewControl.setText(currentField.fieldName);
         textViewControl.setWidth(200);
         fieldLL.addView(textViewControl);

         editTextControl = new EditText(this);
         editTextControl.setWidth(300);
         editTextControl.setTag(currentField.fieldName.replace(' ','_'));
         fieldLL.addView(editTextControl);

      }
      else if (currentField.fieldType.equalsIgnoreCase("radiobutton"))
      {
         textViewControl = new TextView(this);
         textViewControl.setText(currentField.fieldName);
         textViewControl.setLeft(10);
         textViewControl.setWidth(200);
         fieldLL.addView(textViewControl);

         for(String fieldValue : currentField.fieldValues)
         {
            radioButtonControl = new RadioButton(this);
            radioButtonControl.setText(fieldValue);
            radioButtonControl.setTag(fieldValue.replace(' ','_'));
            fieldLL.addView(radioButtonControl);
         }
      }
      else if (currentField.fieldType.equalsIgnoreCase("spinner"))
      {
         textViewControl = new TextView(this);
         textViewControl.setText(currentField.fieldName);
         textViewControl.setLeft(10);
         textViewControl.setWidth(200);
         fieldLL.addView(textViewControl);

         spinnerControl = new Spinner(this);
         spinnerControl.setTag(currentField.fieldName.replace(' ','_'));
         ArrayAdapter<String> aa = new ArrayAdapter<String>( this,
android.R.layout.simple_spinner_item,
                  currentField.fieldValues);
```

```
                spinnerControl.setAdapter(aa);
                fieldLL.addView(spinnerControl);
            }

            ll.addView(fieldLL);
        }
    }

    btnSave = new Button(this);
    btnSave.setText("Save");
    btnSave.setWidth(100);
    btnSave.setHeight(50);
    btnSave.setPadding(10,10,10,10);

    btnView = new Button(this);
    btnView.setText("View");
    btnView.setWidth(100);
    btnView.setHeight(50);
    btnView.setPadding(10,10,10,10);

    btnExit = new Button(this);
    btnExit.setText("Exit");
    btnExit.setWidth(100);
    btnExit.setHeight(50);
    btnExit.setPadding(10,10,10,10);

    fieldLL = new LinearLayout(this);
    fieldLL.setOrientation(LinearLayout.HORIZONTAL);

    fieldLL.addView(btnSave);
    fieldLL.addView(btnView);
    fieldLL.addView(btnExit);

    fieldLL.setGravity(View.TEXT_ALIGNMENT_CENTER);

    ll.addView(fieldLL);

    this.setContentView(sv);

    //Getting database instance
    db = new MySQLiteOpenHelper(this,
          "dynamic_db_"+strApplicationName.replace('
','_'),null,1,fieldsList).getWritableDatabase();

    btnSave.setOnClickListener(new View.OnClickListener()
    {

        @Override
        public void onClick(View v)
        {

            //Creating a row to be inserted in database with the help of ContentValues.
```

```
ContentValues cv = new ContentValues();


//Fetching all data for user has entered before clicking save button
for (int i = 0; i < ll.getChildCount(); i++)
{
   View view = ll.getChildAt(i);
   if (view.getClass().equals(LinearLayout.class))
   {
      LinearLayout childLL = (LinearLayout)view;
      getConentValues(childLL,cv);
   }
   else
   {
      getConentValues(view,cv);
   }
}

try
{
   //Inserting newly created row in table in database
   //db.insert(strApplicationName.replace(' ','_'),null, cv);

   StringBuilder sbInsertSql = new StringBuilder();
   sbInsertSql.append("Insert into "+strApplicationName.replace(' ','_') );

   StringBuilder sbKeys = new StringBuilder();
   sbKeys.append(" (");
   for(String keyItem : cv.keySet())
   {
      sbKeys.append(" "+ keyItem + ",");
   }

   sbInsertSql.append( sbKeys.substring(0, sbKeys.length() -1) + " ) values ( ");
   for(String keyItem : cv.keySet())
   {
      sbInsertSql.append("'" + cv.get(keyItem) + "',");
   }

   String finalQuery = sbInsertSql.substring(0,sbInsertSql.length()-1) + " );";
   System.out.println(finalQuery);

   db.execSQL(finalQuery);

}
catch (Exception ex)
{
   ex.printStackTrace();
}

populateDataList();
```

```
        }
    });



    btnView.setOnClickListener(new View.OnClickListener()
    {

        @Override
        public void onClick(View v)
        {
            populateDataList();
        }
    });

    btnExit.setOnClickListener(new View.OnClickListener()
    {

        @Override
        public void onClick(View v)
        {
            MainActivity.this.finish();
        }
    });

}

private void getConentValues(LinearLayout linearLayout, ContentValues contentValues)
{


    for(int controlIndex=0; controlIndex < linearLayout.getChildCount(); controlIndex++)
    {
        View view = linearLayout.getChildAt(controlIndex);

        getConentValues(view,contentValues);
    }

}

private void getConentValues(View view, ContentValues contentValues)
{
    Class viewClass = view.getClass();
    if (viewClass == EditText.class)
    {
        for (Fields currentField : fieldsList)
        {
            if (currentField.fieldType.equals("edittext"))
            {
                if (view.getTag().equals(currentField.fieldName.replace(' ','_')))
                {
                    contentValues.put("COL_"+currentField.fieldName.replace(' ','_'),
```

```
((EditText)view).getText().toString());
                }
            }
        }
    }
    else if (viewClass == RadioButton.class)
    {
        for (Fields currentField : fieldsList)
        {
            if (currentField.fieldType.equals("radiobutton") && !
contentValues.containsKey("COL_"+currentField.fieldName.replace(' ','_')))
            {
                for(String radioButtonName : currentField.fieldValues)
                {
                    if (view.getTag().equals(radioButtonName.replace(' ','_')) &&
((RadioButton)view).isChecked())
                    {
                        contentValues.put("COL_"+currentField.fieldName.replace(' ','_'), "true");
                    }
                    else
                    {
                        contentValues.put("COL_"+currentField.fieldName.replace(' ','_'), "false");
                    }
                    break;
                }
            }

        }
    }
    else if (viewClass == Spinner.class)
    {
        for (Fields currentField : fieldsList)
        {
            if (currentField.fieldType.equals("spinner"))
            {
                if (view.getTag().equals(currentField.fieldName.replace(' ','_')) )
                {
                    int positionIndex = ((Spinner)view).getSelectedItemPosition();
                    contentValues.put("COL_"+currentField.fieldName.replace(' ','_'),
                            currentField.fieldValues.get(positionIndex));
                }

            }

        }
    }
}


private void populateDataList()
{
    try
```

```
    {
        ScrollView sv = new ScrollView(this);
        ll = new LinearLayout(this);
        ll.setOrientation(LinearLayout.VERTICAL);
        sv.addView(ll);


        //Getting an instance of database
        //db = new
MySQLiteOpenHelper(MainActivity.this,strApplicationName,null,1,fieldsList).getWritableDatabas
e();

        StringBuilder sbFieldNames = new StringBuilder();
        for(Fields currentField : fieldsList)
        {
          sbFieldNames.append("COL_"+currentField.fieldName+",");
        }
        String fieldNames = sbFieldNames.substring(0,sbFieldNames.length()-1);

        System.out.println(fieldNames);

        //Fetching data by executing query on our table.
        Cursor cursor = db.query("dynamic_db_"+strApplicationName.replace(' ','_'), new String[]
{fieldNames}, null, null, null, null, null);

        //Checking wether cursor pointing to first record or not ?
        if(!cursor.isAfterLast())
           cursor.moveToFirst();

        int fieldsCount = fieldsList.size();
        TextView textViewControl = null;
        //Navigating through all records and storing each row in a new candidate object and then
adding it to ArrayList
        do
        {
           for(int fieldIndex = 0; fieldIndex < fieldsCount; fieldIndex++)
           {
              textViewControl = new TextView(this);
              textViewControl.setText(cursor.getString(fieldIndex));
           }
           //candidates.add(new
Candidate(c.getString(0),c.getString(1),c.getString(2),c.getString(3)));
           cursor.moveToNext();
        }
        while(!cursor.isAfterLast());

        cursor.close();


        Button btnBack = new Button(this);
        btnBack.setText("Back");
        ll.addView(ll);
```

```
        btnBack.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                populateMainView();

            }
        });

        this.setContentView(sv);
    }
    catch (Exception ex)
    {
        ex.printStackTrace();
    }
}

@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;
}

private void readXML()
{
    XmlPullParserFactory pullParserFactory;
    try
    {
        pullParserFactory = XmlPullParserFactory.newInstance();
        XmlPullParser parser = pullParserFactory.newPullParser();

        InputStream in_s = getApplicationContext().getAssets().open("dynamic_fields.xml");
        parser.setFeature(XmlPullParser.FEATURE_PROCESS_NAMESPACES, false);
        parser.setInput(in_s, null);

        parseXML(parser);

    }
    catch (XmlPullParserException e)
    {
        e.printStackTrace();
    }
    catch (IOException e)
    {
        e.printStackTrace();
    }
}


private void parseXML(XmlPullParser parser) throws XmlPullParserException,IOException
```

```
{

   int eventType = parser.getEventType();
   Fields currentField = null;

   while (eventType != XmlPullParser.END_DOCUMENT){
      String name = null;
      switch (eventType){
         case XmlPullParser.START_DOCUMENT:
            fieldsList = new ArrayList();
            break;
         case XmlPullParser.START_TAG:
            name = parser.getName();
            if (name.equalsIgnoreCase("application_name"))
            {
               this.strApplicationName = parser.nextText();
            }
            else if (name.equalsIgnoreCase("screen_name"))
            {
               this.strScreenName = parser.nextText();
            }
            else if (name.equalsIgnoreCase("field"))
            {
               currentField = new Fields();
            }
            else if (currentField != null)
            {
               if (name.equalsIgnoreCase("fieldname"))
               {
                  currentField.fieldName = parser.nextText();
               }
               else if (name.equalsIgnoreCase("fieldtype"))
               {
                  currentField.fieldType = parser.nextText();
               }
               else if (name.equalsIgnoreCase("values"))
               {
                  currentField.fieldValues = new ArrayList<String>();
               }
               else if (name.startsWith("fieldvalue"))
               {
                  currentField.fieldValues.add(parser.nextText());
               }

            }
            break;
         case XmlPullParser.END_TAG:
            name = parser.getName();
            if (name.equalsIgnoreCase("field") && currentField != null)
            {
               fieldsList.add(currentField);
            }
```

```
            }

            eventType = parser.next();
        }


    }


}

class Fields
{

    public String fieldName;
    public String fieldType;
    public ArrayList<String> fieldValues;

}

class MySQLiteOpenHelper extends SQLiteOpenHelper
{
    ArrayList<Fields> fieldsList = null;
    String strApplicationName = null;

    public MySQLiteOpenHelper(Context context, String applicationName,
                              CursorFactory factory, int version,
                              ArrayList<Fields> parametersList)
    {

        super(context, applicationName, null, version);
        fieldsList = parametersList;
        strApplicationName = applicationName;
    }

    @Override
    public void onCreate(SQLiteDatabase db)
    {
        StringBuilder sbSql = new StringBuilder();
        sbSql.append("CREATE TABLE "+strApplicationName.replace(' ','_')+"( COL_ID INTEGER
PRIMARY KEY AUTOINCREMENT,");

        for(Fields currentField : fieldsList)
        {
            sbSql.append("COL_"+currentField.fieldName.replace(' ','_') + " TEXT,");
        }
        System.out.println(sbSql.toString());
        db.execSQL(sbSql.substring(0,sbSql.length() - 1) +")");

    }
```

```
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // TODO Auto-generated method stub

    }

}
```