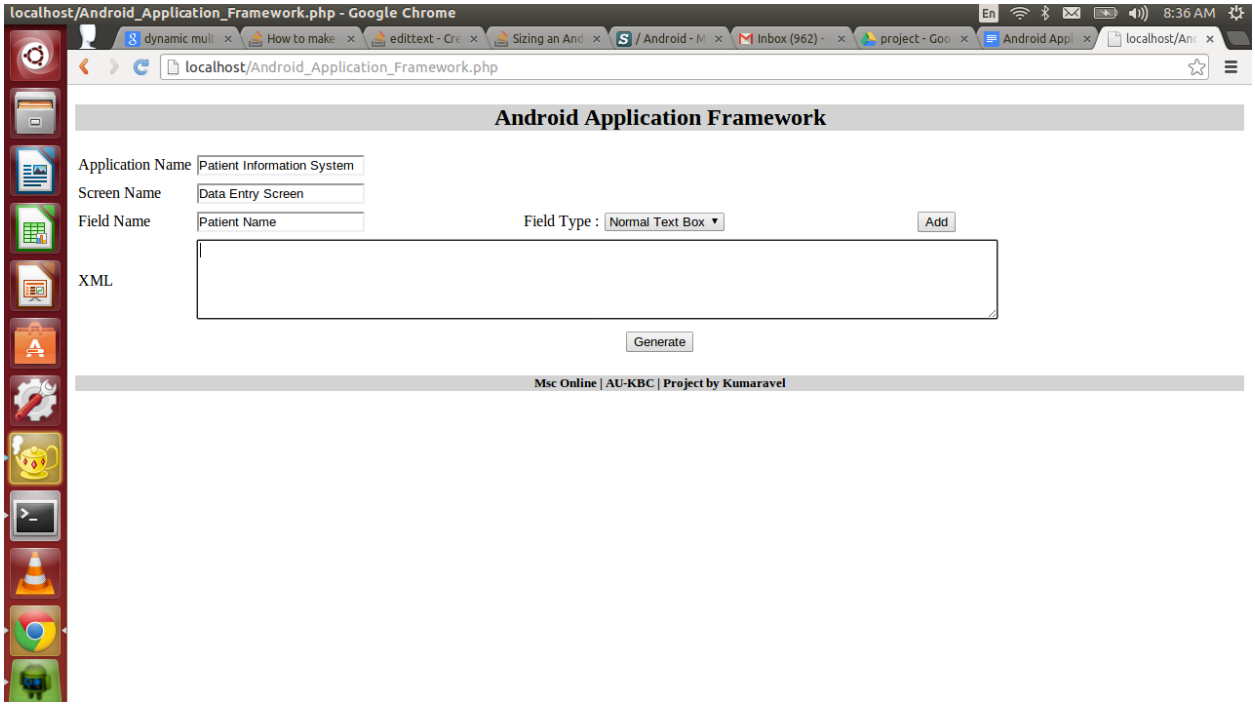
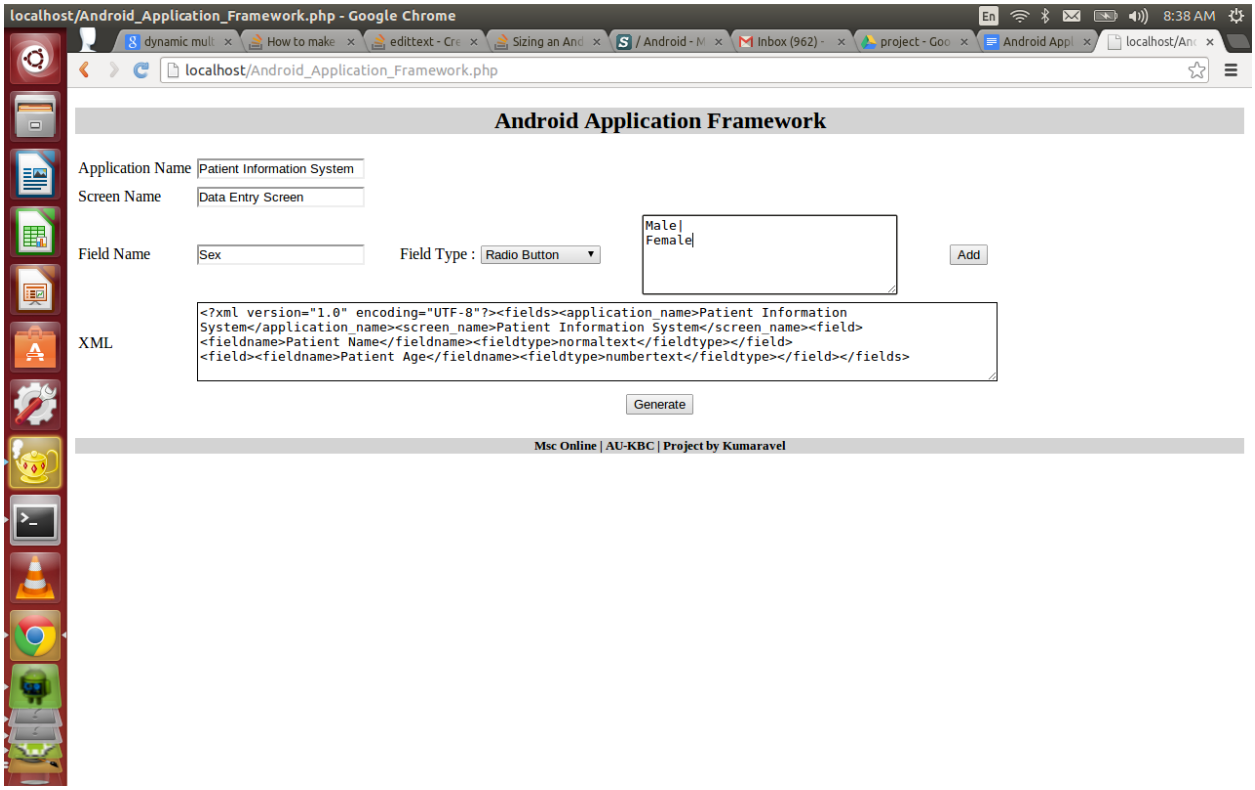
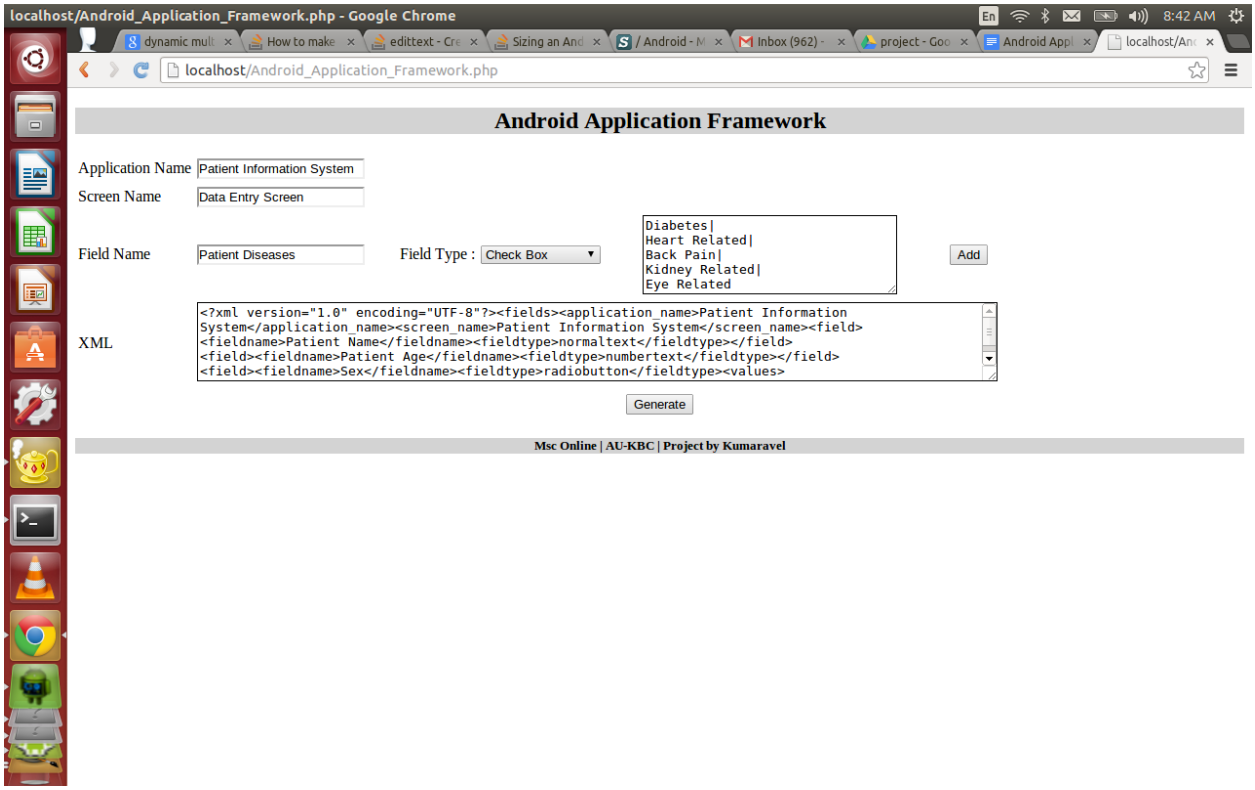
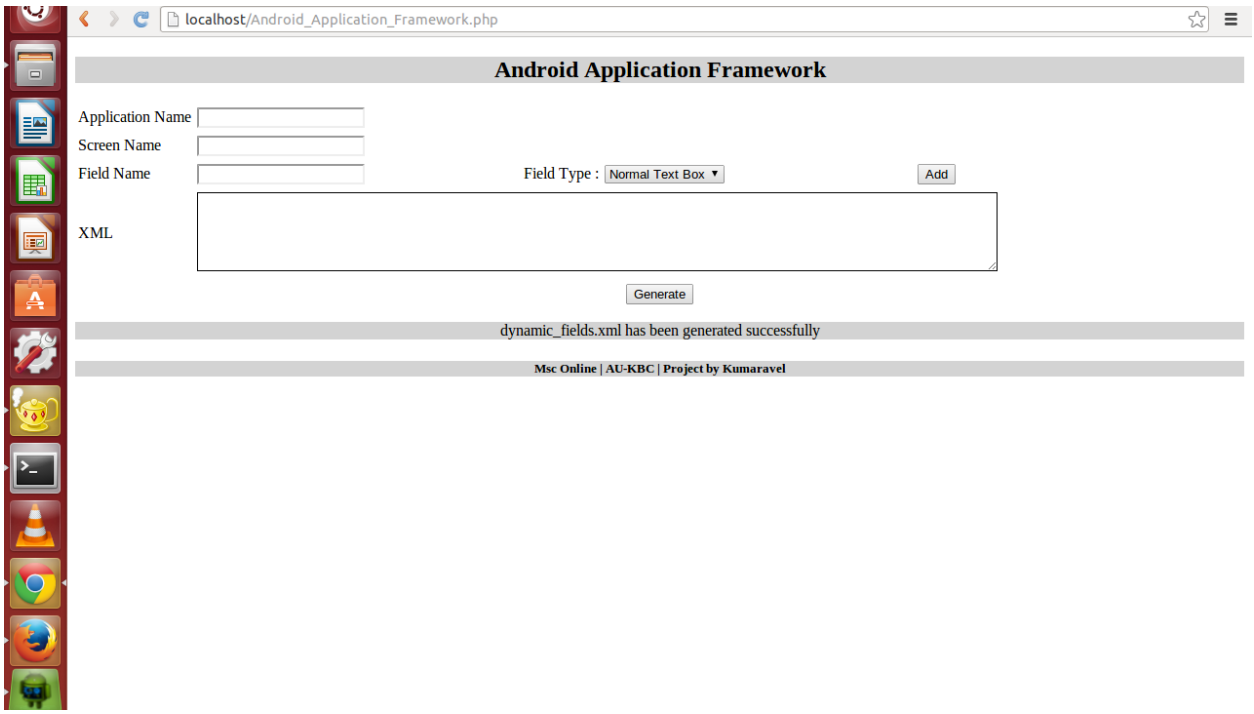

localhost/Android_Application_Framework.php - Google Chrome
8:36 AM
localhost/Android_Application_Framework.php
Android Application Framework
Application Name
Patient Information System
Screen Name
Data Entry Screen
Field Name
Patient Name
Field Type :
Normal Text Box
Add
XML
Generate
Msc Online | AU-KBC | Project by Kumaravel


localhost/Android_Application_Framework.php - Google Chrome
8:38 AM
localhost/Android_Application_Framework.php
Android Application Framework
Application Name
Patient Information System
Screen Name
Data Entry Screen
Field Name
Sex
Field Type :
Radio Button
Male|
Female
Add
XML
<?xml version="1.0" encoding="UTF-8"?><fields><application_name>Patient Information System</application_name><screen_name>Patient Information System</screen_name><field>
<fieldname>Patient Name</fieldname><fieldtype>normaltext</fieldtype></field>
<field><fieldname>Patient Age</fieldname><fieldtype>numbertext</fieldtype></field></fields>
Generate
Msc Online | AU-KBC | Project by Kumaravel


localhost/Android_Application_Framework.php - Google Chrome
8:42 AM
localhost/Android_Application_Framework.php
Android Application Framework
Application Name
Patient Information System
Screen Name
Data Entry Screen
Field Name
Patient Diseases
Field Type :
Check Box
Diabetes|
Heart Related|
Back Pain|
Kidney Related|
Eye Related
Add
XML
<?xml version="1.0" encoding="UTF-8"?><fields><application_name>Patient Information System</application_name><screen_name>Patient Information System</screen_name><field><fieldname>Patient Name</fieldname><fieldtype>normaltext</fieldtype></field><field><fieldname>Patient Age</fieldname><fieldtype>numbertext</fieldtype></field><field><fieldname>Sex</fieldname><fieldtype>radiobutton</fieldtype><values>
Generate
Msc Online | AU-KBC | Project by Kumaravel

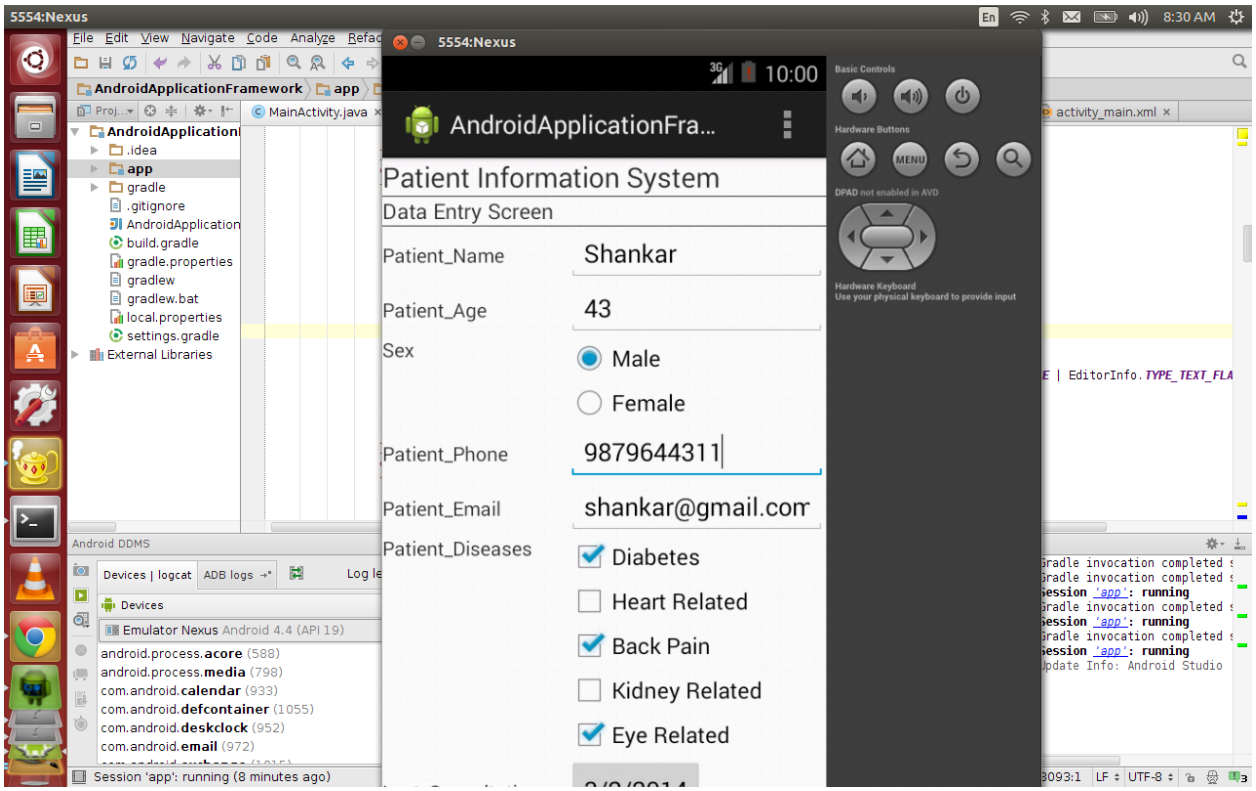

localhost/Android_Application_Framework.php
Android Application Framework
Application Name
Screen Name
Field Name
Field Type :
Normal Text Box
Add
XML
Generate
dynamic_fields.xml has been generated successfully
Msc Online | AU-KBC | Project by Kumaravel

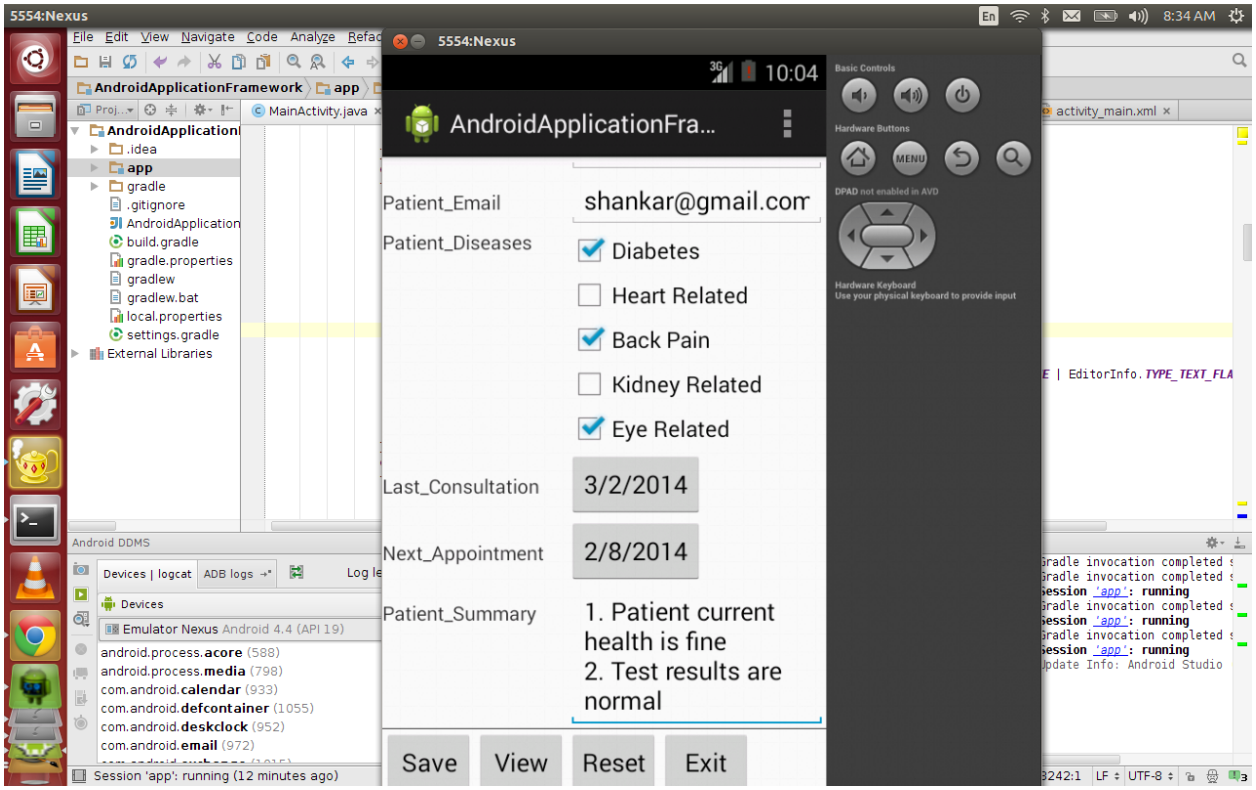
dynamic_fields.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<fields>
	<application_name>Patient Information System</application_name>
	<screen_name>Data Entry Screen</screen_name>
	<field>
		<fieldname>Patient Name</fieldname>
		<fieldtype>normaltext</fieldtype>
	</field>
	<field>
		<fieldname>Patient Age</fieldname>
		<fieldtype>numbertext</fieldtype>
	</field>
	<field>
		<fieldname>Sex</fieldname>
		<fieldtype>radiobutton</fieldtype>
		<values>
			<fieldvalue0>Male</fieldvalue0>
			<fieldvalue1>Female</fieldvalue1>
		</values>
	</field>
	<field>
		<fieldname>Patient Phone</fieldname>
		<fieldtype>phonetext</fieldtype>
	</field>
	<field>
		<fieldname>Patient Email</fieldname>
		<fieldtype>emailtext</fieldtype>
	</field>
	<field>
		<fieldname>Patient Diseases</fieldname>
		<fieldtype>checkbox</fieldtype>
		<values>
			<fieldvalue0>Diabetes</fieldvalue0>
			<fieldvalue1>Heart Related</fieldvalue1>
			<fieldvalue2>Back Pain</fieldvalue2>
			<fieldvalue3>Kidney Related</fieldvalue3>
			<fieldvalue4>Eye Related</fieldvalue4>
		</values>
	</field>
	<field>
		<fieldname>Last Consultation Date</fieldname>
```

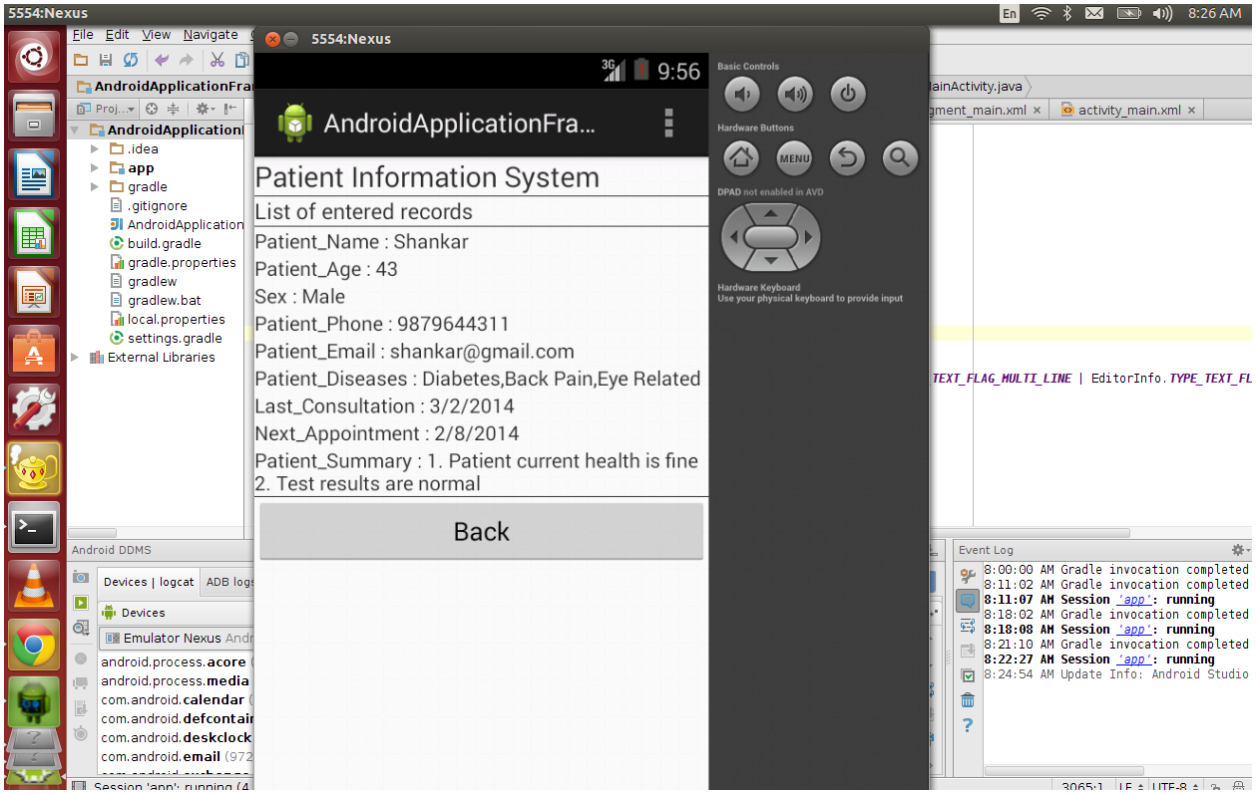
```
            <fieldtype>datepicker</fieldtype>
        </field>
        <field>
            <fieldname>Next Appointment Date</fieldname>
            <fieldtype>datepicker</fieldtype>
        </field>
        <field>
            <fieldname>Patient Summary</fieldname>
            <fieldtype>multilinetext</fieldtype>
        </field>
</fields>
```

Render the dynamic UI, after reading from xml file





After save button, it will show the list of entered records.



Code Reference:-

Android Application Framework - PHP page

```
<!DOCTYPE html>
<head>
	<script type="text/javascript">

		/*
		* Function used to create XML for all the controls and populate in txtResults.
		*/
		function addText()
		{
			//Variables decalaration
			var applicationTitle, screenTitle,
selectedControls,controlName,controlType,controlValue, xmlText;

			//Assign values
			applicationTitle = document.getElementById("txtApplicationName").value;
			screenTitle = document.getElementById("txtScreenName").value;
			selectedControls = document.getElementById("txtXml").value;
			controlName = document.getElementById("controlName").value;
			controlType = document.getElementById("controlType").value;
			controlValue = document.getElementById("controlValue").value;

			//Framing xml
			xmlText =  '<?xml version="1.0" encoding="UTF-8"?><fields>';

			//Application Name
			xmlText = xmlText +
"<application_name>"+applicationTitle+"</application_name>";

			//Screen Name
			xmlText = xmlText +
"<screen_name>"+applicationTitle+"</screen_name>";

			if ( selectedControls.length > 0 )
			{
				xmlText = selectedControls.replace("</fields>","") + "\n" +
getXml(controlName,controlType,controlValue);
			}
			else
			{
```

```
                        xmlText = xmlText +
getXml(controlName,controlType,controlValue);
                }
                xmlText = xmlText + "</fields>";
                document.getElementById("txtXml").innerHTML = xmlText;

                //Clear the content of the control
                document.getElementById("controlName").value = "";
                document.getElementById("controlType").selectedIndex = 0;
                document.getElementById("controlValue").value = "";
        }

        /*
        * Function used to show or hide the controlValue based upon the control type
        * Usually control values parameter is required only for Drop down, Radio Button
and Checkbox
        */
        function visibleControl(sel)
        {
                var controlType = sel.options[sel.selectedIndex].value;
                if ( controlType == "spinner" || controlType == "radiobutton" || controlType
== "checkbox" )
                {
                        document.getElementById("controlValue").style.display = "block";

                }
                else
                {
                        document.getElementById("controlValue").style.display = "none";
                }
        }

        /*
        * Function used to ammend the | symbol when press enter key
        */
        function fKeyDown(e)
        {
                var  kc = window.event ? window.event.keyCode : e.which;
                if (kc == 13)
                {
                        document.getElementById('controlValue').value =
document.getElementById('controlValue').value + "|";
                }
```

```
		}

		/*
		* Function used to get the XML based upon each control add
		*/
		function getXml(controlName, controlType, controlValue)
		{
			var xmlText = "<field><fieldname>" + controlName + "</fieldname>";

			xmlText = xmlText + "<fieldtype>" + controlType +"</fieldtype>";

			if (controlValue.length > 0 )
			{
				var controlValues = controlValue.split("|");
				if (controlValues.length > 0 )
				{
					xmlText = xmlText + "<values>";

					for (var index=0,len=controlValues.length; index<len;
index++)
					{

						xmlText = xmlText + "<fieldvalue"+index+">"+
controlValues[index] +"</fieldvalue"+index+">";
					}
					xmlText = xmlText + "</values>";
				}
			}
			xmlText=xmlText + "</field>";
			return xmlText;
		}


	</script>
</head>
<body>
	<div  style="background-color:lightgrey"><h2 align="center">Android Application
Framework</h2></div>
	<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
		<table>
			<tr>
				<td>Application Name</td>
				<td><input type="text" name="txtApplicationName"
```

```
id="txtApplicationName"/></td>
                        </tr>
                        <tr>
                                <td>Screen Name</td><td><input type="text"
name="txtScreenName" id="txtScreenName"/></td>
                        </tr>
                        <tr border><td>Field Name</td><td><input type="text"
name="controlName" id="controlName"></td>
                        <td>
                                Field Type :
                                <select name="controlType" id="controlType"
onchange="visibleControl(this);">
                                  <option value="normaltext">Normal Text Box</option>
                                  <option value="numbertext">Number</option>
                                  <option value="radiobutton">Radio Button</option>
                                  <option value="datepicker">Date Picker</option>
                                  <option value="spinner">Drop Down</option>
                                  <option value="checkbox">Check Box</option>

                                  <option value="phonetext">Phone</option>
                                  <option value="multilinetext">Multiline Text</option>
                                  <option value="passwordtext">Password Text</option>
                                  <option value="emailtext">Email</option>
                                  <option value="uritext">Web URL</option>
                                </select>
                        </td>
                        <td>
                                <textarea name="controlValue" id="controlValue" rows="5"
cols="30" style="display:none" onKeyDown=javascript:fKeyDown(event);></textarea>
                        </td>
                        <td><input type="button" onclick="addText();" name="btnAdd"
value="Add" width="100"/></td>
                        </tr>
                        <tr>
                                <td>XML </td><td colspan="6" align="left"><textarea
name="txtXml" id="txtXml" rows="5" cols="100"></textarea>
                                </td>
                        </tr>
                        <tr></tr>

                        </table>
                        <div align="center"><input type="submit" name="submit"
value="Generate" align="center"></div>
```

```
                        <p style="background-color:lightgrey" align="center">
                        <?php
                                if(isset($_POST['submit']))
                                {
                                        $txtXml  = $_POST['txtXml'];
                                        file_put_contents("dynamic_fields.xml", $txtXml);
                                        echo "dynamic_fields.xml has been generated
successfully";
                                }
                        ?>
                        </p>
                </form>
                <div  style="background-color:lightgrey"><h5 align="center">Msc Online |
AU-KBC | Project by Kumaravel</h2></div>
        </body>
</html>
```

Main Activity.java

```
package foss.projects.androidapplicationframework;

import android.app.ActionBar;
import android.app.Dialog;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.app.Activity;
import android.text.InputType;
import android.view.Gravity;
import android.view.Menu;
import android.view.View;
import android.view.ViewGroup;
import android.view.inputmethod.EditorInfo;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.DatePicker;
import android.widget.ListView;
import android.widget.RadioGroup;
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.app.DatePickerDialog;
import android.widget.ArrayAdapter;

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

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;
import java.util.Calendar;


@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;
   Button btnReset = null;
   Button btnDatePicker = null;

   SQLiteDatabase db;
   String tableName;
   ScrollView sv = null;
   LinearLayout parentLinearLayout = null;
   LinearLayout checkBoxLayout = null;

   ListView lstFields = null;
   int year,month,day,mYear, mMonth, mDay;

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

```
    {
        super.onCreate(savedInstanceState);

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

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

        parentLinearLayout.setDividerPadding(5);

        TextView textViewControl = null;
        EditText editTextControl = null;
        RadioGroup radioGroup = null;
        RadioButton radioButtonControl = null;
        Spinner spinnerControl = null;
        CheckBox checkBox = null;

        //Read the Fields xml file
        readXML();

        LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(
                ViewGroup.LayoutParams.FILL_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
        params.gravity = Gravity.CENTER;


        //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);
        textViewControl.setLayoutParams(params);
        parentLinearLayout.addView(textViewControl);

        View line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

        //Set Table Name
        tableName = strApplicationName.replace(' ','_');

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

        line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

        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("normaltext"))
              {
```

```
                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);
                fieldLL.addView(editTextControl);

            }
            else if (currentField.fieldType.equalsIgnoreCase("emailtext"))
            {
                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(' ', '_'));

editTextControl.setInputType(InputType.TYPE_TEXT_VARIATION_EMAIL_ADDRESS);
                fieldLL.addView(editTextControl);

            }
            else if (currentField.fieldType.equalsIgnoreCase("phonetext"))
            {
                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);
                editTextControl.setInputType(InputType.TYPE_CLASS_PHONE);
                fieldLL.addView(editTextControl);
            }
            else if (currentField.fieldType.equalsIgnoreCase("numbertext"))
            {
                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);
                editTextControl.setInputType(InputType.TYPE_CLASS_NUMBER);
                fieldLL.addView(editTextControl);
            }
            else if (currentField.fieldType.equalsIgnoreCase("uritext"))
            {
                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);
                editTextControl.setInputType(InputType.TYPE_TEXT_VARIATION_URI);
                fieldLL.addView(editTextControl);
            }
            else if (currentField.fieldType.equalsIgnoreCase("multilinetext"))
            {
                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);
                editTextControl.setSingleLine(false);

                editTextControl.setImeOptions(EditorInfo.IME_FLAG_NO_EXTRACT_UI);
                editTextControl.setFocusableInTouchMode(true);
                editTextControl.setInputType(EditorInfo.TYPE_CLASS_TEXT |
EditorInfo.TYPE_TEXT_FLAG_MULTI_LINE | EditorInfo.TYPE_TEXT_FLAG_IME_MULTI_LINE);
                editTextControl.setMaxLines(Integer.MAX_VALUE);
                editTextControl.setHorizontallyScrolling(false);
                editTextControl.setTransformationMethod(null);
                fieldLL.addView(editTextControl);
            }
```

```
else if (currentField.fieldType.equalsIgnoreCase("passwordtext"))
{
   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);
   editTextControl.setInputType(InputType.TYPE_MASK_VARIATION);
   fieldLL.addView(editTextControl);
}
else if (currentField.fieldType.equalsIgnoreCase("datepicker"))
{
   textViewControl = new TextView(this);
   textViewControl.setText(currentField.fieldName);
   textViewControl.setWidth(200);
   fieldLL.addView(textViewControl);

   btnDatePicker = new Button(this);


   btnDatePicker.setTag(currentField.fieldName.replace(' ', '_'));
   fieldLL.addView(btnDatePicker);

   final Calendar c = Calendar.getInstance();
   mYear = c.get(Calendar.YEAR);
   mMonth = c.get(Calendar.MONTH) + 1;
   mDay = c.get(Calendar.DAY_OF_MONTH);

   btnDatePicker.setText(mDay+"/"+mMonth+"/"+mYear);


   // Set ClickListener on btnDatePicker
   btnDatePicker.setOnClickListener(new View.OnClickListener() {

      public void onClick(View v) {
         // Show the DatePickerDialog
         showDialog(0);
      }
   });
}
```

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

               radioGroup = new RadioGroup(this);
               for(String fieldValue : currentField.fieldValues)
               {
                  radioButtonControl = new RadioButton(this);
                  radioButtonControl.setText(fieldValue);
                  radioButtonControl.setTag(fieldValue.replace(' ','_'));
                  radioGroup.addView(radioButtonControl);
               }
               fieldLL.addView(radioGroup);
            }
            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);
               ArrayAdapter<String> aa = new ArrayAdapter<String>( this,
android.R.layout.simple_spinner_item,
                     currentField.fieldValues);
               spinnerControl.setAdapter(aa);
               fieldLL.addView(spinnerControl);
            }
            else if (currentField.fieldType.equalsIgnoreCase("checkbox"))
            {

               textViewControl = new TextView(this);
               textViewControl.setText(currentField.fieldName);
               textViewControl.setLeft(10);
               textViewControl.setWidth(200);
               fieldLL.addView(textViewControl);
```

```
            checkBoxLayout = new LinearLayout(this);
            checkBoxLayout.setOrientation(LinearLayout.VERTICAL);
            checkBoxLayout.setTag("checkbox");



            for(String fieldValue : currentField.fieldValues)
            {
                checkBox = new CheckBox(this);
                checkBox.setTag(fieldValue);
                checkBox.setText(fieldValue);
                checkBoxLayout.addView(checkBox);
            }

            fieldLL.addView(checkBoxLayout);
        }

        parentLinearLayout.addView(fieldLL);
    }
}

btnSave = new Button(this);
btnSave.setText("Save");
btnSave.setWidth(100);
btnSave.setHeight(30);
btnSave.setPadding(10, 10, 10, 10);

btnView = new Button(this);
btnView.setText("View");
btnView.setWidth(100);
btnView.setHeight(30);
btnView.setPadding(10, 10, 10, 10);

btnReset = new Button(this);
btnReset.setText("Reset");
btnReset.setWidth(100);
btnReset.setHeight(30);
btnReset.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);

        line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

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

        fieldLL.setGravity(View.TEXT_ALIGNMENT_CENTER);

        parentLinearLayout.addView(fieldLL);

        line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

        this.setContentView(sv);

        //Getting database instance
        db = new MySQLiteOpenHelper(this,
              "dynamic_db_"+tableName,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 < parentLinearLayout.getChildCount(); i++)
{
   View view = parentLinearLayout.getChildAt(i);
   if (view.getClass().equals(LinearLayout.class))
   {
      LinearLayout childLL = (LinearLayout)view;
      getConentValues(childLL,cv);
   }
   else
   {
      getConentValues(view,cv);
   }
}

try
{
   //Check whether table exist or not
   if (!isTableExists(db,tableName))
   {
      createTable(db,tableName);
   }

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

   StringBuilder sbInsertSql = new StringBuilder();
   sbInsertSql.append("Insert into "+tableName );

   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) + " );";
```

```
            db.execSQL(finalQuery);

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

        populateDataList();

    }
});



btnView.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        populateDataList();
    }
});

btnReset.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        deleteDatabse(db,tableName);
    }
});

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("normaltext") ||
currentField.fieldType.equals("emailtext")
                        || currentField.fieldType.equals("phonetext") ||
currentField.fieldType.equals("numbertext")
                        || currentField.fieldType.equals("uritext") ||
currentField.fieldType.equals("multilinetext")
                        || currentField.fieldType.equals("passwordtext")
                        )
                {
                    if (view.getTag() != null && view.getTag().equals(currentField.fieldName))
                    {
                        contentValues.put("COL_"+currentField.fieldName,
((EditText)view).getText().toString());
                        break;
                    }
                }
            }
        }
        else if (viewClass == Button.class)
        {
            for (Fields currentField : fieldsList)
```

```
        {
            if (currentField.fieldType.equals("datepicker"))
            {
                if (view.getTag() != null && view.getTag().equals(currentField.fieldName))
                {
                    contentValues.put("COL_"+currentField.fieldName,
((Button)view).getText().toString());
                    break;
                }
            }
        }
        else if (viewClass == RadioGroup.class)
        {
            int radioGroupChildCount = ((RadioGroup)view).getChildCount();
            for( int radioButtonIndex =0; radioButtonIndex < radioGroupChildCount;
radioButtonIndex++)
            {
                RadioButton childRadioButton = (RadioButton)
((RadioGroup)view).getChildAt(radioButtonIndex);
                for (Fields currentField : fieldsList)
                {
                    if (currentField.fieldType.equals("radiobutton") &&
!contentValues.containsKey("COL_"+currentField.fieldName))
                    {
                        for(String radioButtonName : currentField.fieldValues)
                        {
                            if ( childRadioButton.getTag() != null )
                            {
                                if ( childRadioButton.getTag().equals(radioButtonName) &&
childRadioButton.isChecked())
                                {
                                    contentValues.put("COL_"+currentField.fieldName, radioButtonName);
                                    break;
                                }
                            }

                        }
                    }

                }
            }
        }
```

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

                }

            }
        }
        else if (viewClass == LinearLayout.class && view.getTag() != null  &&
view.getTag().equals("checkbox"))
        {
            for (Fields currentField : fieldsList)
            {
                if ( !contentValues.containsKey("COL_"+currentField.fieldName))
                {
                    int checkBoxChildCount = checkBoxLayout.getChildCount();
                    StringBuilder sbCheckBoxValues = new StringBuilder();
                    for(int checkBoxIndex = 0; checkBoxIndex< checkBoxChildCount;
checkBoxIndex++)
                    {
                        CheckBox currentCheckBox =
(CheckBox)checkBoxLayout.getChildAt(checkBoxIndex);
                        if ( currentCheckBox.isChecked() )
                        {
                            sbCheckBoxValues.append(currentCheckBox.getTag());
                            sbCheckBoxValues.append(",");
                        }
                    }

contentValues.put("COL_"+currentField.fieldName,sbCheckBoxValues.substring(0,sbCheckBox
Values.length()-1));
                    break;
```

```
            }

        }
    }
}


private void populateDataList()
{
    try
    {
        sv = new ScrollView(this);
        parentLinearLayout = new LinearLayout(this);
        parentLinearLayout.setOrientation(LinearLayout.VERTICAL);
        parentLinearLayout.setGravity(parentLinearLayout.TEXT_ALIGNMENT_GRAVITY);
        sv.addView(parentLinearLayout);

        LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(
              ViewGroup.LayoutParams.FILL_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
        params.gravity = Gravity.CENTER;

        //Set Application Title
        TextView 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);
        textViewControl.setLayoutParams(params);
        parentLinearLayout.addView(textViewControl);

        View line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

        //Set Screen heading
        textViewControl = new TextView(this);
        textViewControl.setText("List of entered records");
        textViewControl.setLeft(10);
        textViewControl.setTextAlignment(View.TEXT_ALIGNMENT_CENTER);
```

```
        textViewControl.setTextSize(15);
        textViewControl.setLayoutParams(params);
        parentLinearLayout.addView(textViewControl);

        line = new View(this);
        line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
        line.setBackgroundColor(Color.rgb(51, 51, 51));
        parentLinearLayout.addView(line);

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


        //Check whether table exist or not
        if (isTableExists(db,tableName))
        {
           StringBuilder sbFieldNames = new StringBuilder();
           for(Fields currentField : fieldsList)
           {
              sbFieldNames.append("COL_"+currentField.fieldName+",");
           }
           String fieldNames = sbFieldNames.substring(0,sbFieldNames.length()-1);

           //Fetching data by executing query on our table.
           Cursor cursor = db.query(tableName, 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();

              //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(fieldsList.get(fieldIndex).fieldName + " : "+
cursor.getString(fieldIndex));
                    parentLinearLayout.addView(textViewControl);
                }
                line = new View(this);
                line.setLayoutParams(new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT, 1));
                line.setBackgroundColor(Color.rgb(51, 51, 51));
                parentLinearLayout.addView(line);

                cursor.moveToNext();
            }
            while(!cursor.isAfterLast());

        }
        else
        {
            textViewControl = new TextView(this);
            textViewControl.setText("No records exist in database");
            parentLinearLayout.addView(textViewControl);
        }
        cursor.close();
    }
    else
    {
        textViewControl = new TextView(this);
        textViewControl.setText("Table is not created yet");
        parentLinearLayout.addView(textViewControl);
    }
    Button btnBack = new Button(this);
    btnBack.setText("Back");
    parentLinearLayout.addView(btnBack);

    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();
               currentField.fieldName = currentField.fieldName.replace(' ', '_');
            }
            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();
        }


    }

    private void createTable(SQLiteDatabase db, String tableName)
    {
        StringBuilder sbSql = new StringBuilder();
        sbSql.append("CREATE TABLE "+tableName+" ( COL_ID INTEGER PRIMARY KEY
AUTOINCREMENT,");

        for(Fields currentField : fieldsList)
        {
            sbSql.append("COL_"+currentField.fieldName + " TEXT,");
        }

        db.execSQL(sbSql.substring(0,sbSql.length() - 1) +")");
    }

    private boolean isTableExists(SQLiteDatabase db, String tableName)
    {
        Cursor cursor = db.rawQuery("Select count(*) from sqlite_master where type = ? AND
name = ?",
                new String[] {"table", tableName});
        if (!cursor.moveToFirst())
        {
            return false;
        }
        int count = cursor.getInt(0);
        cursor.close();
```

```
        return count > 0;
    }

    public void deleteDatabse(SQLiteDatabase db, String tableName)
    {
        db.delete(tableName, "1", new String[] {});
    }

    // Register  DatePickerDialog listener
    private DatePickerDialog.OnDateSetListener mDateSetListener =
            new DatePickerDialog.OnDateSetListener() {
                // the callback received when the user "sets" the Date in the DatePickerDialog
                public void onDateSet(DatePicker view, int yearSelected,
                                      int monthOfYear, int dayOfMonth) {
                    year = yearSelected;
                    month = monthOfYear + 1;
                    day = dayOfMonth;
                    // Set the Selected Date in Select date Button
                    btnDatePicker.setText(day+"/"+month+"/"+year);
                }
            };


    // Method automatically gets Called when you call showDialog()  method
    protected Dialog onCreateDialog(int parameter) {
        // create a new DatePickerDialog with values you want to show
        return new DatePickerDialog(this,
                mDateSetListener,
                mYear, mMonth, mDay);

    }



}

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 ");
      sbSql.append(strApplicationName.replace(' ','_'));
      sbSql.append("( COL_ID INTEGER PRIMARY KEY AUTOINCREMENT,");

      for(Fields currentField : fieldsList)
      {
         sbSql.append("COL_");
         sbSql.append(currentField.fieldName );
         sbSql.append(" TEXT,");
      }
      db.execSQL(sbSql.substring(0,sbSql.length() - 1) +")");

   }

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

   }
```

```
}
```