**Mini Project : Class Scheduler Application Using Android**

**AIM:**

To implement a Class Scheduler Application using Android.

**OBJECTIVE:**

* To learn the layouts and controls of UI.
* To understand the working of Android.

**SOFTWARE REQUIRED:**

JDK 1.6.0 or higher, Android Developer’s Kit version 22.0.4 , Eclipse IDE version 4.2.1

**DESCRIPTION:**

The Class Scheduler application allows a trainer to keep track of batches and classes. Once trainer enters details about batches that are to be started, the application displays the ending date and the entire schedule of classes. It is also possible to add, delete and cancel classes.

The following are the major operations in this application.

* List of Batches – Displays the batch details with date of commencing, subject code and name , the course type.
* Adding new batch – Adding batch with new details and timings.
* Updating an existing batch – It helps to alter or update about the class ie. about the new timings or date.
* Deleting an existing batch – After the course for a particular batch is over,the batch’s details can be deleted from the existing schedule of classes and batches.
* List of classes – Displays the list of classes,all subjects and courses being handled.
* Adding a new class without or without changing last date
* Cancel a class and add a new class after the last class
* Update an existing class
* Deleting an existing class

This application on whole is simple to use and it helps to organize the classes and batches. Specific details like course number, name, date of commencing,date of ending,number of hours per week,duration of each class are collected in order to organize,schedule and manage the classes and the batches accordingly . This app efficiently manages and schedules classes and it is also easy to update and delete batches if the particular course has ended .

**PROGRAM:**

**AddBatchActivity.java :**

package com.st.cs;

import java.util.Calendar;

import android.app.Activity;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.TimePickerDialog;

import android.os.Bundle;

import android.view.View;

import android.widget.DatePicker;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.TimePicker;

import android.widget.Toast;

public class AddBatchActivity extends Activity {

private static final int *DATE\_DIALOG* = 1;

private static final int *TIME\_DIALOG* = 2;

private int day, month, year, hours, mins;

private TextView textStartDate, textStartTime;

private EditText editBatchcode,editCourse,editPeriod,editClasses,editClassesPerWeek, editRemarks;

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.*addbatch*);

textStartDate = (TextView) this.findViewById(R.id.*textStartDate*);

textStartTime = (TextView) this.findViewById(R.id.*textStartTime*);

editBatchcode = (EditText) this.findViewById(R.id.*editBatchCode*) ;

editCourse = (EditText) this.findViewById(R.id.*editCourse*) ;

editPeriod = (EditText) this.findViewById(R.id.*editPeriod*) ;

editClasses = (EditText) this.findViewById(R.id.*editClasses*) ;

editClassesPerWeek = (EditText) this.findViewById(R.id.*editClassesPerWeek*) ;

editRemarks = (EditText) this.findViewById(R.id.*editRemarks*) ;

setDateToSysdate();updateDateDisplay();}

private void setDateToSysdate() {

Calendar c = Calendar.*getInstance*();

day = c.get(Calendar.*DAY\_OF\_MONTH*);

month = c.get(Calendar.*MONTH*);

year = c.get(Calendar.*YEAR*);

}

public void addBatch(View v) {

boolean done = Database.*addBatch*(this,

editBatchcode.getText().toString(),

editCourse.getText().toString(),

textStartDate.getText().toString(),

textStartTime.getText().toString(),

editClasses.getText().toString(),

editPeriod.getText().toString(),

editClassesPerWeek.getText().toString(),

editRemarks.getText().toString());

if ( done )

Toast.*makeText*(this,"Added batch successfully!", Toast.*LENGTH\_LONG*).show();

else

Toast.*makeText*(this,"Sorry! Could not add batch!", Toast.*LENGTH\_LONG*).show();}

public void showDatePicker(View v) {~~showDialog~~(*DATE\_DIALOG*);}

public void showTimePicker(View v) {

~~showDialog~~(*TIME\_DIALOG*);}

protected Dialog onCreateDialog(int id) {

super.~~onCreateDialog~~(id);

switch (id) {

case *DATE\_DIALOG*:

return new DatePickerDialog(this, dateSetListener, year, month, day);

case *TIME\_DIALOG*:

return new TimePickerDialog(this, timeSetListener, hours,mins, false);}return null;}

private DatePickerDialog.OnDateSetListener dateSetListener = new DatePickerDialog.OnDateSetListener() {

public void onDateSet(DatePicker view, int pYear, int pMonth, int pDay) {

year = pYear;

month = pMonth;

day = pDay;

updateDateDisplay();}};

private TimePickerDialog.OnTimeSetListener timeSetListener =

new TimePickerDialog.OnTimeSetListener() {

public void onTimeSet(TimePicker arg0, int pHours, int pMins) {

hours = pHours;

mins = pMins;

updateTimeDisplay();}};

private void updateDateDisplay() {

textStartDate.setText(String.*format*("%04d-%02d-%02d", year, month + 1,day));}

private void updateTimeDisplay() {

textStartTime.setText(String.*format*("%02d:%02d", hours,mins));}}

**AddClassActivity.java:**

package com.st.cs;

import java.util.Calendar;

import android.app.Activity;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.TimePickerDialog;

import android.os.Bundle;

import android.view.View;

import android.widget.CheckBox;

import android.widget.DatePicker;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.TimePicker;

import android.widget.Toast;

public class AddClassActivity extends Activity {

private static final int DATE\_DIALOG = 1;

private static final int TIME\_DIALOG = 2;

private int day, month, year, hours, mins;

private TextView textClassDate, textClassTime, textBatchCode;

private EditText editPeriod,editRemarks, editTopics;

private CheckBox chkAdjust;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.addclass);

textBatchCode = (TextView) this.findViewById(R.id.textBatchCode);

textClassDate = (TextView) this.findViewById(R.id.textClassDate);

textClassTime = (TextView) this.findViewById(R.id.textClassTime);

editPeriod = (EditText) this.findViewById(R.id.editPeriod) ;

editRemarks = (EditText) this.findViewById(R.id.editRemarks) ;

editTopics = (EditText) this.findViewById(R.id.editTopics) ;

chkAdjust = (CheckBox) this.findViewById(R.id.chkAdjust);

textBatchCode.setText( getIntent().getStringExtra("batchcode"));

setDateToSysdate();

updateDateDisplay();}

private void setDateToSysdate() {

Calendar c = Calendar.getInstance();

day = c.get(Calendar.DAY\_OF\_MONTH);

month = c.get(Calendar.MONTH);

year = c.get(Calendar.YEAR);}

public void addClass(View v) {

boolean done = Database.addClass(this,

textBatchCode.getText().toString(),

textClassDate.getText().toString(),

textClassTime.getText().toString(),

editPeriod.getText().toString(),

editTopics.getText().toString(),

editRemarks.getText().toString(),

chkAdjust.isChecked());

if ( done )

Toast.makeText(this,"Added Class Successfully!", Toast.LENGTH\_LONG).show();

else

Toast.makeText(this,"Sorry! Could not add class!", Toast.LENGTH\_LONG).show();}

public void showDatePicker(View v) {showDialog(DATE\_DIALOG);}

public void showTimePicker(View v) {showDialog(TIME\_DIALOG);}

protected Dialog onCreateDialog(int id) {super.onCreateDialog(id);

switch (id) {

case DATE\_DIALOG:

return new DatePickerDialog(this, dateSetListener, year, month, day);

case TIME\_DIALOG:

return new TimePickerDialog(this, timeSetListener, hours,mins, false);}

return null;}

private DatePickerDialog.OnDateSetListener dateSetListener = new DatePickerDialog.OnDateSetListener() {

public void onDateSet(DatePicker view, int pYear, int pMonth, int pDay) {

year = pYear;

month = pMonth;

day = pDay;

updateDateDisplay();}};

private TimePickerDialog.OnTimeSetListener timeSetListener =

new TimePickerDialog.OnTimeSetListener() {

public void onTimeSet(TimePicker arg0, int pHours, int pMins) {

hours = pHours;

mins = pMins;

updateTimeDisplay();}};

private void updateDateDisplay() {

// Month is 0 based so add 1

textClassDate.setText(String.format("%04d-%02d-%02d", year, month + 1,day));}}

**Batch.java:**

package com.st.cs;

public class Batch {

private String code,course,startdate, enddate,starttime,classes,period,classesperweek,remarks;

public String getCode() {

return code;}

public void setCode(String code) {this.code = code;}

public String getCourse() {return course;}

public void setCourse(String course) {this.course = course;}

public String getStartdate() {return startdate;}

public void setStartdate(String startdate) {this.startdate = startdate;}

public String getEnddate() {return enddate;}

public void setEnddate(String enddate) {this.enddate = enddate;}

public String getStarttime() {return starttime;}

public void setStarttime(String starttime) {this.starttime = starttime;}

public String getClasses() {return classes;}

public void setClasses(String classes) {this.classes = classes;}

public String getPeriod() {return period;}

public void setPeriod(String period) {this.period = period;}

public String getClassesperweek() {return classesperweek;}

public void setClassesperweek(String classesperweek) {this.classesperweek = classesperweek;}

public String getRemarks() {return remarks;}

public void setRemarks(String remarks) {this.remarks = remarks;}}

**BatchesAdapter.java:**

package com.st.cs;

import java.util.ArrayList;

import android.content.Context;

import android.content.Intent;

import android.view.LayoutInflater;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.ViewGroup;

import android.widget.BaseAdapter;

import android.widget.Button;

import android.widget.TextView;

public class BatchesAdapter extends BaseAdapter {

private LayoutInflater inflater;

private ArrayList<Batch> batches;

public BatchesAdapter(Context ctx) {

inflater = LayoutInflater.from(ctx);

batches = Database.getBatches(ctx);}

@Override

public int getCount() {

return batches.size();}

@Override

public Object getItem(int pos) {

return batches.get(pos);}

@Override

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

@Override

public View getView(int position, View convertView, ViewGroup parent) {

if (convertView == null) {

convertView = inflater.inflate(R.layout.batch, null);

Button btnClasses = (Button) convertView.findViewById(R.id.btnClasses);

Button btnUpdate = (Button) convertView.findViewById(R.id.btnUpdate);

Button btnAddClass = (Button) convertView.findViewById(R.id.btnAddClass);

final Batch batch = batches.get(position);

TextView textCode = (TextView) convertView.findViewById(R.id.textCode);

textCode.setText( batch.getCode());

TextView textCourse = (TextView) convertView.findViewById(R.id.textCourse);

textCourse.setText( batch.getCourse());

TextView textStartDate = (TextView) convertView.findViewById(R.id.textStartDate);

textStartDate.setText(batch.getStartdate());

TextView textEndDate = (TextView) convertView.findViewById(R.id.textEndDate);

textEndDate.setText(batch.getEnddate());

btnClasses.setOnClickListener(new OnClickListener() {

public void onClick(View view) {

Context context = view.getContext();

Intent intent = new Intent(context, ListClassesActivity.class);

intent.putExtra("batchcode",batch.getCode());

context.startActivity(intent);}});

btnAddClass.setOnClickListener(new OnClickListener() {

public void onClick(View view) {

Context context = view.getContext();

Intent intent = new Intent(context, AddClassActivity.class);

intent.putExtra("batchcode",batch.getCode());

context.startActivity(intent);}});

btnUpdate.setOnClickListener(new OnClickListener() {

public void onClick(View view) {

Context context = view.getContext();

Intent intent = new Intent(context, UpdateBatchActivity.class);

intent.putExtra("batchcode",batch.getCode());

context.startActivity(intent);}});}

return convertView;}}

**Class.java:**

package com.st.cs;

public class Class {

private String classno, classId, classDate, classTime, period, topics, remarks, batchCode;

public String getBatchCode() {

return batchCode;}

public void setBatchCode(String batchCode) {

this.batchCode = batchCode;}

public String getClassId() {return classId;}

public void setClassId(String classId) {this.classId = classId;}

public String getClassno() {return classno;}

public void setClassno(String classno) {

this.classno = classno;}

public String getClassDate() {return classDate;}

public void setClassDate(String classDate) {this.classDate = classDate;}

public String getClassTime() {return classTime;}

public void setClassTime(String classTime) {

this.classTime = classTime;}

public String getPeriod() {return period;}

public void setPeriod(String period) {this.period = period;}

public String getTopics() {return topics;}

public void setTopics(String topics) {this.topics = topics;}

public String getRemarks() {

return remarks;}

public void setRemarks(String remarks) {

this.remarks = remarks;}}

**ClassSchedulerActivity.java:**

package com.st.cs;

import android.app.Activity;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

public class ClassSchedulerActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

DBHelper dbhelper = new DBHelper(this);

SQLiteDatabase db = dbhelper.getWritableDatabase();

db.close();}}

**Database.java:**

package com.st.cs;

import java.util.ArrayList;

import java.util.Calendar;

import android.content.ContentValues;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.util.Log;

public class Database {

public static final String BATCHES\_TABLE\_NAME = "batches";

public static final String BATCHES\_ID = "\_id";

public static final String BATCHES\_BATCHCODE = "batchcode";

public static final String BATCHES\_COURSE = "course";

public static final String BATCHES\_STARTDATE = "startdate";

public static final String BATCHES\_STARTTIME = "starttime";

public static final String BATCHES\_CLASSES = "classes";

public static final String BATCHES\_PERIOD = "period";

public static final String BATCHES\_CLASSESPERWEEK = "classesperweek";

public static final String BATCHES\_REMARKS = "remarks";

public static final String CLASSES\_TABLE\_NAME = "classes";

public static final String CLASSES\_CLASSES\_ID = "\_id";

public static final String CLASSES\_BATCHCODE = "batchcode";

public static final String CLASSES\_CLASSDATE = "classdate";

public static final String CLASSES\_CLASSTIME = "classtime";

public static final String CLASSES\_CLASSPERIOD = "period";

public static final String CLASSES\_TOPICS = "topics";

public static final String CLASSES\_REMARKS = "remarks";

public static Batch cursorToBatch(Cursor batches) {

Batch batch = new Batch();

batch.setCode(batches.getString(batches

.getColumnIndex(Database.BATCHES\_BATCHCODE)));

batch.setCourse(batches.getString(batches

.getColumnIndex(Database.BATCHES\_COURSE)));

batch.setStartdate(batches.getString(batches

.getColumnIndex(Database.BATCHES\_STARTDATE)));

batch.setStarttime(batches.getString(batches

.getColumnIndex(Database.BATCHES\_STARTTIME)));

batch.setClasses(batches.getString(batches

.getColumnIndex(Database.BATCHES\_CLASSES)));

batch.setClassesperweek(batches.getString(batches

.getColumnIndex(Database.BATCHES\_CLASSESPERWEEK)));

batch.setPeriod(batches.getString(batches

.getColumnIndex(Database.BATCHES\_PERIOD)));

batch.setRemarks(batches.getString(batches

.getColumnIndex(Database.BATCHES\_REMARKS)));

return batch;}

public static boolean deleteLastClass(SQLiteDatabase db, String batchcode) {

try {

boolean done = false;

Cursor lastClass = db.query(Database.CLASSES\_TABLE\_NAME, null,Database.CLASSES\_BATCHCODE + " = ?",

new String[] { batchcode},

null, null, Database.CLASSES\_CLASSDATE + " desc " , "1");

String classid = null;

if ( lastClass.moveToFirst() ) {

classid = lastClass.getString( lastClass.getColumnIndex( Database.CLASSES\_CLASSES\_ID));

int rows = db.delete(Database.CLASSES\_TABLE\_NAME,Database.CLASSES\_CLASSES\_ID + " = ?",

new String[] { classid} );

done = rows == 1;}

lastClass.close();

return done;}

catch (Exception ex) {

Log.d("Account", "Error in deleteLastClass-->" + ex.getMessage());

return false;}}

public static String calendarToString(Calendar c) {

return String.format("%04d-%02d-%02d", c.get(Calendar.YEAR), c.get(Calendar.MONTH) + 1, c.get(Calendar.DAY\_OF\_MONTH));}

public static boolean cancelClass(Context context, String batchCode,String classid) {

DBHelper dbhelper = null;SQLiteDatabase db = null;

try {dbhelper = new DBHelper(context);

db = dbhelper.getWritableDatabase();

db.beginTransaction();

int rows = db.delete(Database.CLASSES\_TABLE\_NAME,Database.CLASSES\_CLASSES\_ID + " = ?",

new String[] { classid} );

if ( rows == 1)

{if ( addAfterLastClass(db,batchCode))

{db.setTransactionSuccessful();

db.endTransaction();

return true;}}

db.endTransaction();

return false;}

catch (Exception ex) {

Log.d("CS", "Error in cancelClass-->" + ex.getMessage());

return false;}

finally {

if (db != null && db.isOpen()) {

db.close();}}}

public static boolean addAfterLastClass(SQLiteDatabase db, String batchcode) {

try {

boolean done = false;

Batch batch = null;

Cursor lastClass = db.query(Database.CLASSES\_TABLE\_NAME, null,Database.CLASSES\_BATCHCODE + " = ?",

new String[] { batchcode},

null, null, Database.CLASSES\_CLASSDATE + " desc " , "1");

String classdate = null;

if ( lastClass.moveToFirst() ) {

classdate = lastClass.getString(lastClass.getColumnIndex( Database.CLASSES\_CLASSDATE));

batch = getBatch(db,batchcode);

Calendar c = getCalendar(classdate);

int classesperweek = Integer.parseInt( batch.getClassesperweek());

c.add( Calendar.DAY\_OF\_MONTH,1);

int dow = getDayOfWeek(c.get(Calendar.DAY\_OF\_WEEK));

if ( dow == 7 && classesperweek == 6) {

c.add( Calendar.DAY\_OF\_MONTH,1); }

else

if ( dow == 6 && classesperweek == 5)

c.add( Calendar.DAY\_OF\_MONTH,2);

lastClass.close();

ContentValues values = new ContentValues();

values.put(Database.CLASSES\_BATCHCODE, batch.getCode());

values.put(Database.CLASSES\_CLASSDATE, calendarToString(c));

values.put(Database.CLASSES\_CLASSTIME, batch.getStarttime());

values.put(Database.CLASSES\_CLASSPERIOD, batch.getPeriod());

values.put(Database.CLASSES\_REMARKS,"");

values.put(Database.CLASSES\_TOPICS,"");

long rowid = db.insert(Database.CLASSES\_TABLE\_NAME, null,values);

return rowid >= 0;}

else

return false;}

catch (Exception ex) {

Log.d("Account", "Error in deleteLastClass-->" + ex.getMessage());

return false;}}

public static boolean deleteClass(Context context, String classid) {

DBHelper dbhelper = null;

SQLiteDatabase db = null;

try {dbhelper = new DBHelper(context);

db = dbhelper.getWritableDatabase();

int rows = db.delete(Database.CLASSES\_TABLE\_NAME,Database.CLASSES\_CLASSES\_ID + " = ?",

new String[] { classid} );

return rows == 1;}

catch (Exception ex) {

Log.d("CS", "Error in deleteClass-->" + ex.getMessage());

return false;}

finally {

if (db != null && db.isOpen()) {

db.close();}}}

public static boolean addBatch(Context context, String batchcode,

String course, String startdate, String starttime, String classes,

String period, String classesperweek, String remarks) {

DBHelper dbhelper = null;

SQLiteDatabase db = null;

try {dbhelper = new DBHelper(context);

db = dbhelper.getWritableDatabase();

db.beginTransaction();

ContentValues values = new ContentValues();

values.put(Database.BATCHES\_BATCHCODE, batchcode);

values.put(Database.BATCHES\_COURSE, course);

values.put(Database.BATCHES\_STARTDATE, startdate);

values.put(Database.BATCHES\_STARTTIME, starttime);

values.put(Database.BATCHES\_CLASSES, classes);

values.put(Database.BATCHES\_PERIOD, period);

values.put(Database.BATCHES\_CLASSESPERWEEK, classesperweek);

values.put(Database.BATCHES\_REMARKS, remarks);

long rowid = db.insert(Database.BATCHES\_TABLE\_NAME, null, values);

Log.d("CS", "Inserted into BATCHES " + rowid);

addClasses(db, batchcode, startdate, starttime, classes, period,classesperweek);

db.setTransactionSuccessful();

db.endTransaction();

return true;

} catch (Exception ex) {

Log.d("Account", "Error in addTransaction -->" + ex.getMessage());

return false;

} finally {

if (db != null && db.isOpen()) {

db.close();}}}

public static boolean updateBatch(Context context, String batchcode,

String course, String starttime,String period, String remarks) {

DBHelper dbhelper = null;

SQLiteDatabase db = null;

try {dbhelper = new DBHelper(context);

db = dbhelper.getWritableDatabase();

db.beginTransaction();

if ( rows == 1)

db.setTransactionSuccessful();

db.endTransaction();

return true;

} catch (Exception ex) {

Log.d("Account", "Error in deleteBatch-->" + ex.getMessage());

return false;

} finally {

if (db != null && db.isOpen()) {

db.close();}}}

private static int getDayOfWeek(int dow) {

if (dow == 1)

return 7;

else

return dow - 1;}

public static void addClasses(SQLiteDatabase db, String batchcode,

String startdate, String starttime, String classes, String period,

String classesperweek) throws Exception {

try {ContentValues values = new ContentValues();

values.put(Database.CLASSES\_BATCHCODE, batchcode);

values.put(Database.CLASSES\_CLASSTIME, starttime);

values.put(Database.CLASSES\_CLASSPERIOD, period);

values.put(Database.CLASSES\_REMARKS, "");

values.put(Database.CLASSES\_TOPICS, "");

String[] parts = startdate.split("-");

int year = Integer.parseInt(parts[0]);

int month = Integer.parseInt(parts[1]) - 1; // zero based month

int day = Integer.parseInt(parts[2]);

Calendar c = Calendar.getInstance();

c.set(year, month, day);

int noclasses = Integer.parseInt(classes);

int cpw = Integer.parseInt(classesperweek);

int classnumber = 1;

do {

int dow = c.get(Calendar.DAY\_OF\_WEEK);

if (getDayOfWeek(dow) <= cpw) {

values.put(Database.CLASSES\_CLASSDATE, String.format(

"%04d-%02d-%02d", c.get(Calendar.YEAR),

c.get(Calendar.MONTH) + 1,

c.get(Calendar.DAY\_OF\_MONTH)));

long rowid = db.insert(Database.CLASSES\_TABLE\_NAME, null,

values);

Log.d("CS", "Inserted into CLASSES" + rowid);classnumber++;}

c.add(Calendar.DAY\_OF\_MONTH, 1); // increment

} while (classnumber <= noclasses);

} catch (Exception ex) {

Log.d("CS", "Error in addClasses -->" + ex.getMessage());

throw ex;}}

public static ArrayList<Batch> getBatches(Context context) {

DBHelper dbhelper = new DBHelper(context);

SQLiteDatabase db = dbhelper.getReadableDatabase();

Cursor batches = db.query(Database.BATCHES\_TABLE\_NAME, null, null,

null, null, null, null);

ArrayList<Batch> list = new ArrayList<Batch>();

while (batches.moveToNext()) {

Batch batch = Database.cursorToBatch(batches);

String enddate = getEndDate(db, batch.getCode());

batch.setEnddate(enddate);

list.add(batch);

}batches.close();

db.close();

dbhelper.close();

return list;}

public static ArrayList<Class> getClasses(Context context, String batchcode) {

DBHelper dbhelper = new DBHelper(context);

SQLiteDatabase db = dbhelper.getReadableDatabase();

Cursor classes = db.query(Database.CLASSES\_TABLE\_NAME, null,

Database.CLASSES\_BATCHCODE + " = ?",

new String[] { batchcode},

null, null, Database.CLASSES\_CLASSDATE, null);

ArrayList<Class> list = new ArrayList<Class>();

while (classes.moveToNext()) {

Class cls = Database.cursorToClass(classes);

list.add(cls);}

classes.close();

db.close();

dbhelper.close();

return list;}

public static String getEndDate(SQLiteDatabase db, String batchcode) {

Cursor cursor = db.query(Database.CLASSES\_TABLE\_NAME,

new String[] { Database.CLASSES\_CLASSDATE },

Database.CLASSES\_BATCHCODE + "=?", new String[] { batchcode },

null, null, Database.CLASSES\_CLASSDATE + " desc", "1");

cursor.moveToFirst();

String enddate = cursor.getString(cursor.getColumnIndex(Database.CLASSES\_CLASSDATE));

cursor.close();

return enddate;}

public static Batch getBatch (SQLiteDatabase db, String batchcode) {

Cursor batches = db.query(Database.BATCHES\_TABLE\_NAME, null,

Database.BATCHES\_BATCHCODE + " = ?" ,

new String [] { batchcode},

null, null, null, null);

Batch batch;

if ( batches.moveToNext() )

batch = Database.cursorToBatch(batches);

else

batch = null;

batches.close();

return batch;}

public static Batch getBatch(Context context, String batchcode) {

DBHelper dbhelper = new DBHelper(context);

SQLiteDatabase db = dbhelper.getReadableDatabase();

Batch batch = getBatch(db,batchcode);

db.close();

dbhelper.close();

return batch;}

public static Calendar getCalendar(String date) {

String [] parts = date.split("-");

Calendar c = Calendar.getInstance();

c.set( Integer.parseInt( parts[0]), Integer.parseInt( parts[1]) -1, Integer.parseInt( parts[2]));

return c;}

public static Class getClass(Context context, String classid) {

DBHelper dbhelper = new DBHelper(context);

SQLiteDatabase db = dbhelper.getReadableDatabase();

Cursor classes = db.query(Database.CLASSES\_TABLE\_NAME, null,

Database.CLASSES\_CLASSES\_ID + " = ?" ,

new String [] { classid},

null, null, null, null);

Class clas;

if ( classes.moveToNext() )

clas = Database.cursorToClass(classes);

else

clas = null;

classes.close();

db.close();

dbhelper.close();

return class;} }

**DBHelper.java:**

package com.st.cs;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

import android.util.Log;

public class DBHelper extends SQLiteOpenHelper {

public static final int DB\_VERSION = 1;

public static final String DB\_NAME = "cs.db";

public DBHelper(Context ctx) {

super(ctx, DB\_NAME, null, DB\_VERSION);}

public void onCreate(SQLiteDatabase db) {

createTables(db);}

public void onUpgrade(SQLiteDatabase arg0, int arg1, int arg2) {}

public void createTables(SQLiteDatabase database) {

String batches\_table\_sql = "create table " + Database.BATCHES\_TABLE\_NAME + " ( " +

Database.BATCHES\_ID + " integer primary key autoincrement," +

Database.BATCHES\_BATCHCODE + " TEXT," +

Database.BATCHES\_COURSE + " TEXT," +

Database.BATCHES\_STARTDATE + " TEXT," +

Database.BATCHES\_STARTTIME + " TEXT," +

Database.BATCHES\_CLASSES + " integer," +

Database.BATCHES\_PERIOD + " integer," +

Database.BATCHES\_CLASSESPERWEEK + " integer," +

Database.BATCHES\_REMARKS + " TEXT)";

String classes\_table\_sql = "create table " + Database.CLASSES\_TABLE\_NAME + " ( " +

Database.CLASSES\_CLASSES\_ID + " integer primary key autoincrement," +

Database.CLASSES\_BATCHCODE + " TEXT," +

Database.CLASSES\_CLASSDATE + " TEXT," +

Database.CLASSES\_CLASSTIME + " TEXT," +

Database.CLASSES\_CLASSPERIOD + " integer," +

Database.CLASSES\_TOPICS+ " TEXT," +

Database.CLASSES\_REMARKS + " TEXT)";

try {

database.execSQL(batches\_table\_sql);

database.execSQL("insert into batches (batchcode,course, startdate,starttime,classes,period,classesperweek,remarks)"

+ "values ('HB2404','Hibernate','2012-04-24','19:00',6,90,6,'Short course')");

database.execSQL(classes\_table\_sql);

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-24','19:00',90,null,null)");

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-25','19:00',90,null,null)");

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-26','19:00',90,null,null)");

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-27','19:00',90,null,null)");

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-28','19:00',90,null,null)");

database.execSQL("insert into classes (batchcode,classdate,classtime,period,topics,remarks)"

+ "values ('HB2404','2012-04-30','19:00',90,null,'Last class')");

Log.d("CS","Tables created!");}

catch(Exception ex) {

Log.d("CS", "Error in DBHelper.onCreate() : " + ex.getMessage());}}}

**ListBatchesActivity.java:**

package com.st.cs;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.ListView;

public class ListBatchesActivity extends Activity {

ListView listBatches;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.listbatches);}

public void onStart() {

super.onStart();

listBatches = (ListView) this.findViewById(R.id.listBatches);

BatchesAdapter adapter = new BatchesAdapter(this);

listBatches.setAdapter(adapter);}

public void addBatch(View v) {

Intent intent = new Intent(this, AddBatchActivity.class);

startActivity(intent);}}

**ListClassesActivity.java:**

package com.st.cs;

import java.util.List;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.ImageButton;

import android.widget.TableLayout;

import android.widget.TableLayout.LayoutParams;

import android.widget.TableRow;

import android.widget.TextView;

public class ListClassesActivity extends Activity {

String batchcode;

TableLayout tableClasses;

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.listclasses);

batchcode = getIntent().getStringExtra("batchcode");

tableClasses = (TableLayout) this.findViewById( R.id.tableClasses);}

public void onStart() {

super.onStart();

deleteRowsFromTable();

addRowsToTable(tableClasses,batchcode);}

public void deleteRowsFromTable() {

if ( tableClasses.getChildCount() > 2)

tableClasses.removeViews(2,tableClasses.getChildCount() - 2);}

private void addRowsToTable(TableLayout table, String batchcode) {

List<Class> classes = Database.getClasses(this, batchcode);

TableRow tr = new TableRow(this);

tr.setLayoutParams(new LayoutParams(LayoutParams.MATCH\_PARENT,LayoutParams.WRAP\_CONTENT));

int classno = 1;

for(final Class c : classes) {

TableRow row = (TableRow) LayoutInflater.from(this).inflate(R.layout.classrow, null);

((TextView)row.findViewById(R.id.textNo)).setText( String.valueOf(classno));

((TextView)row.findViewById(R.id.textDate)).setText(c.getClassDate());

((TextView)row.findViewById(R.id.textTime)).setText(c.getClassTime());ImageButton btnUpdate = (ImageButton) row.findViewById(R.id.btnUpdate);

btnUpdate.setOnClickListener( new OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent( ListClassesActivity.this,UpdateClassActivity.class);

intent.putExtra("classid", c.getClassId());

startActivity(intent);}});

table.addView(row);

TableRow line = new TableRow(this);

TextView tv = new TextView(this);

tv.setBackgroundColor(Color.RED);

TableRow.LayoutParams lp = new TableRow.LayoutParams(LayoutParams.MATCH\_PARENT,3);

lp.span = 4;

tv.setLayoutParams(lp);

line.addView(tv);

table.addView(line);classno ++;}}}

**UpdateBatchActivity.java:**

package com.st.cs;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.TimePickerDialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.view.View;

import android.widget.DatePicker;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.TimePicker;

import android.widget.Toast;

public class UpdateBatchActivity extends Activity {

private static final int DATE\_DIALOG = 1;

private static final int TIME\_DIALOG = 2;

private static final int DELETE\_ALERT\_DIALOG = 3;

private int day, month, year, hours, mins;

private TextView textStartDate, textStartTime,textClasses,textClassesPerWeek;

private EditText editBatchcode,editCourse,editPeriod, editRemarks;

@Override

public void onCreate(Bundle savedInstanceState) {super.onCreate(savedInstanceState);

setContentView(R.layout.updatebatch);

textStartDate = (TextView) this.findViewById(R.id.textStartDate);

textStartTime = (TextView) this.findViewById(R.id.textStartTime);

editBatchcode = (EditText) this.findViewById(R.id.editBatchCode) ;

editCourse = (EditText) this.findViewById(R.id.editCourse) ;

editPeriod = (EditText) this.findViewById(R.id.editPeriod) ;

textClasses = (TextView) this.findViewById(R.id.textClasses) ;

textClassesPerWeek = (TextView) this.findViewById(R.id.textClassesPerWeek) ;

editRemarks = (EditText) this.findViewById(R.id.editRemarks) ;

String batchcode = getIntent().getStringExtra("batchcode");

Batch batch = Database.getBatch(this, batchcode);

if ( batch == null)

{}

else{

editBatchcode.setText( batch.getCode());

editCourse.setText( batch.getCourse());

textStartDate.setText( batch.getStartdate());

textStartTime.setText( batch.getStarttime());

editPeriod.setText( batch.getPeriod());

textClasses.setText( batch.getClasses());

textClassesPerWeek.setText( batch.getClassesperweek());

editRemarks.setText( batch.getRemarks());

setDateToStartDate( batch.getStartdate());

setTimeToStartTime( batch.getStarttime());}}

private void setDateToStartDate(String startdate) {

String [] parts = startdate.split("-");

day = Integer.parseInt( parts[2]);

month =Integer.parseInt( parts[1]);

year = Integer.parseInt( parts[0]);}

private void setTimeToStartTime(String starttime) {

String [] parts = starttime.split(":");

hours = Integer.parseInt( parts[0]);

mins =Integer.parseInt( parts[1]);}

public void updateBatch(View v) {

boolean done = Database.updateBatch(this,

editBatchcode.getText().toString(),

editCourse.getText().toString(),

textStartTime.getText().toString(),

editPeriod.getText().toString(),

editRemarks.getText().toString());

if ( done )

Toast.makeText(this,"Updated batch successfully!", Toast.LENGTH\_LONG).show();

else

Toast.makeText(this,"Sorry! Could not update batch!", Toast.LENGTH\_LONG).show();}

public void deleteBatch(View v) {this.showDialog(DELETE\_ALERT\_DIALOG);}

public void showDatePicker(View v) {showDialog(DATE\_DIALOG);}

public void showTimePicker(View v) {showDialog(TIME\_DIALOG);}

protected Dialog onCreateDialog(int id) {super.onCreateDialog(id);

switch (id) {

case DATE\_DIALOG:

return new DatePickerDialog(this, dateSetListener, year, month, day);

case TIME\_DIALOG:

return new TimePickerDialog(this, timeSetListener, hours,mins, false);

case DELETE\_ALERT\_DIALOG:

return getAlertDialog();}

return null;}

private DatePickerDialog.OnDateSetListener dateSetListener = new DatePickerDialog.OnDateSetListener() {

public void onDateSet(DatePicker view, int pYear, int pMonth, int pDay) {

year = pYear;

month = pMonth;

day = pDay;

updateDateDisplay();}};

private TimePickerDialog.OnTimeSetListener timeSetListener =

new TimePickerDialog.OnTimeSetListener() {

@Override

public void onTimeSet(TimePicker arg0, int pHours, int pMins) {

hours = pHours;

mins = pMins;

updateTimeDisplay();}};

public void onClick(DialogInterface dialog, int id) {

dialog.cancel();}});

return builder.create();}}

**UpdateClassActivity.java:**

package com.st.cs;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.TimePickerDialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.view.View;

import android.widget.DatePicker;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.TimePicker;

import android.widget.Toast;

public class UpdateClassActivity extends Activity {

private static final int TIME\_DIALOG = 1;

private static final int CANCEL\_ALERT\_DIALOG = 2;

private static final int DELETE\_ALERT\_DIALOG = 3;

private int day, month, year, hours, mins;

private TextView textClassDate, textClassTime, textBatchCode;

private EditText editPeriod,editRemarks, editTopics;

private String classid;

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.updateclass);

textClassDate = (TextView) this.findViewById(R.id.textClassDate);

textClassTime = (TextView) this.findViewById(R.id.textClassTime);

textBatchCode = (TextView) this.findViewById(R.id.textBatchCode);

editPeriod = (EditText) this.findViewById(R.id.editPeriod) ;

editTopics = (EditText) this.findViewById(R.id.editTopics) ;

editRemarks = (EditText) this.findViewById(R.id.editRemarks) ;

classid = getIntent().getStringExtra("classid");

Class clas = Database.getClass(this, classid);

private void setTimeToStartTime(String starttime) {

String [] parts = starttime.split(":");

hours = Integer.parseInt( parts[0]);

mins =Integer.parseInt( parts[1]);}

public void updateClass(View v) {

boolean done = Database.updateClass(this,

classid,

textClassTime.getText().toString(),

editPeriod.getText().toString(),

editTopics.getText().toString(),

editRemarks.getText().toString());

public void deleteClass(View v) {

this.showDialog(DELETE\_ALERT\_DIALOG);}

protected Dialog onCreateDialog(int id) {

super.onCreateDialog(id);

switch (id) {

case TIME\_DIALOG:

return new TimePickerDialog(this, timeSetListener, hours,mins, false);

case CANCEL\_ALERT\_DIALOG:

return getCancelAlertDialog();

case DELETE\_ALERT\_DIALOG:

return getDeleteAlertDialog();}

return null;}

private TimePickerDialog.OnTimeSetListener timeSetListener =

new TimePickerDialog.OnTimeSetListener() {public void onTimeSet(TimePicker arg0, int pHours, int pMins) {

hours = pHours;

mins = pMins;

updateTimeDisplay();}};

private void updateTimeDisplay() {

textClassTime.setText(String.format("%02d:%02d", hours,mins));}

public Dialog getDeleteAlertDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setMessage("Do you want to delete current class?")

.setCancelable(false)

.setPositiveButton("Yes",

new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

boolean done = Database.deleteClass(UpdateClassActivity.this, classid);

if ( done ) {

Toast.makeText(UpdateClassActivity.this,"Deleted Class Successfully!", Toast.LENGTH\_LONG).show();

UpdateClassActivity.this.finish();}

else

Toast.makeText(UpdateClassActivity.this,"Sorry! Could not delete class!", Toast.LENGTH\_LONG).show();}})

.setNegativeButton("No", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

dialog.cancel();}});

return builder.create();}

public Dialog getCancelAlertDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setMessage("Do you want to delete current class and add another class?")

.setCancelable(false)

.setPositiveButton("Yes",new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

boolean done = Database.cancelClass(UpdateClassActivity.this, textBatchCode.getText().toString(), classid);

if ( done ) {

Toast.makeText(UpdateClassActivity.this,"Cancelled current class and added new class successfully!", Toast.LENGTH\_LONG).show();

UpdateClassActivity.this.finish();}

else

Toast.makeText(UpdateClassActivity.this,"Sorry! Could not cancel class!", Toast.LENGTH\_LONG).show();}})

.setNegativeButton("No", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

dialog.cancel();}});return builder.create();}}

**AndroidManifest.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<manifest xmlns:android=*"http://schemas.android.com/apk/res/android"*

package=*"com.st.cs"*

android:versionCode=*"1"*

android:versionName=*"1.0"* >

<uses-sdk android:minSdkVersion=*"8"* />

<application

android:icon=*"@drawable/ic\_launcher"*

android:label=*"Class Scheduler"* >

<activity

android:label=*"List Of Batches"*

android:name=*".ListBatchesActivity"* >

<intent-filter >

<action android:name=*"android.intent.action.MAIN"* />

<category android:name=*"android.intent.category.LAUNCHER"* />

</intent-filter>

</activity>

<activity android:label=*"Add Batch"* android:name=*".AddBatchActivity"* />

<activity android:label=*"List Classes"* android:name=*".ListClassesActivity"* />

<activity android:label=*"Add Class"* android:name=*".AddClassActivity"* />

<activity android:label=*"Update Batch"* android:name=*".UpdateBatchActivity"* />

<activity android:label=*"Update Class"* android:name=*".UpdateClassActivity"* />

</application>

</manifest>

**AddBatch.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<ScrollView xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"* >

<TableLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"*

android:stretchColumns=*"1"* >

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Code"* />

<EditText

android:id=*"@+id/editBatchCode"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<requestFocus /></EditText></TableRow><TableRow ><TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Course"* />

<EditText

android:id=*"@+id/editCourse"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

</EditText>

</TableRow>

<TableRow >

<TextView

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Start Date"* />

<LinearLayout

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<ImageButton

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"showDatePicker"*

android:src=*"@drawable/datepicker"* >

</ImageButton>

<TextView

android:id=*"@+id/textStartDate"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:paddingRight=*"10dp"*

android:text=*"2012-04-24"* >

</TextView>

</LinearLayout>

</TableRow>

<TableRow >

<TextView

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Start Time"* />

<LinearLayout

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<ImageButton

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"showTimePicker"*

android:src=*"@drawable/datepicker"* >

</ImageButton>

<TextView

android:id=*"@+id/textStartTime"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:paddingRight=*"10dp"*

android:text=*"19:00"* >

</TextView></LinearLayout></TableRow>

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"No. Classes"* />

<EditText

android:id=*"@+id/editClasses"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:inputType=*"number"* >

</EditText>

</TableRow>

<TableRow >

<TextView

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Class Date"* />

<LinearLayout

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<ImageButton

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"showDatePicker"*

android:src=*"@drawable/datepicker"* >

</ImageButton>

<TextView

android:id=*"@+id/textClassDate"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:paddingRight=*"10dp"*

android:text=*"2012-04-24"* >

</TextView>

</LinearLayout>

</TableRow>

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Topics"* />

<EditText

android:id=*"@+id/editTopics"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"* >

</EditText>

</TableRow>

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Remarks"* />

<EditText

android:id=*"@+id/editRemarks"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

</EditText>

</TableRow>

<TableRow android:layout\_span=*"2"*>

<CheckBox

android:id=*"@+id/chkAdjust"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Adjust Last Class?"* >

</CheckBox>

</TableRow>

<TableRow android:layout\_span=*"2"* android:gravity=*"center"*>

<Button

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"addClass"*

android:text=*"Add Class"* >

</Button>

</TableRow>

</TableLayout>

**Batch.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"* >

<LinearLayout android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*>

<TextView

android:id=*"@+id/textCode"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"code"*

android:textSize=*"16sp"* />

<TextView

android:id=*"@+id/textCourse"*

android:gravity=*"right"*

android:layout\_weight=*"1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:textSize=*"16sp"*

android:text=*"Course"*/>

</LinearLayout>

<LinearLayout android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*>

<TextView

android:id=*"@+id/textStartDate"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:textSize=*"16sp"*

android:text=*"stdate"* />

<TextView

android:id=*"@+id/textEndDate"*

android:gravity=*"right"*

android:layout\_weight=*"1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:textSize=*"16sp"*

android:text=*"EndDate"*/>

</LinearLayout>

**Classrow.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<TableRow xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*>

<TextView

android:id=*"@+id/textNo"*

android:layout\_width=*"80dp"*

android:layout\_height=*"wrap\_content"*

android:gravity=*"center"*

android:text=*"no"* />

<TextView

android:id=*"@+id/textDate"*

android:layout\_width=*"80dp"*

android:layout\_height=*"wrap\_content"*

android:gravity=*"center"*

android:text=*"Date"* />

<TextView

android:id=*"@+id/textTime"*

android:layout\_width=*"80dp"*

android:layout\_height=*"wrap\_content"*

android:gravity=*"center"*

android:text=*"Time"* />

<ImageButton

android:id=*"@+id/btnUpdate"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"30dp"*

android:layout\_gravity=*"center"*

android:adjustViewBounds=*"true"*

android:src=*"@drawable/edit"*

android:textSize=*"12sp"* />

</TableRow>

**Listbatches.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"* >

<ListView

android:id=*"@+id/listBatches"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"* >

</ListView>

<Button

android:id=*"@+id/butAddBatch"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"addBatch"*

android:layout\_gravity=*"center\_horizontal"*

android:text=*"Add New Batch"* />

</LinearLayout>

**Listclasses.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<ScrollView xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*>

<TableLayout

android:id=*"@+id/tableClasses"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:stretchColumns=*"\*"*>

<TableRow android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*>

<TextView

android:layout\_width=*"80dp"*

android:gravity=*"center"*

android:layout\_height=*"wrap\_content"*

android:text=*"Class No."* />

<TextView

android:layout\_width=*"80dp"*

android:layout\_height=*"wrap\_content"*

android:gravity=*"center"*

android:text=*"Class Date"* />

</ScrollView>

**Main.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"* >

<TextView

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"@string/hello"* />

</LinearLayout>

**Updatebatch.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<ScrollView xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"* >

<TableLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"*

android:stretchColumns=*"1"* >

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Code"* />

<EditText

android:id=*"@+id/editBatchCode"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<requestFocus />

</EditText>

</TableRow>

<TableRow >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Course"* />

<EditText

android:id=*"@+id/editCourse"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

</EditText>

</TableRow>

<TableRow >

<TextView

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Start Date"* />

<TableRow>

<LinearLayout

android:layout\_span=*"2"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<Button

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"updateBatch"*

android:text=*"Update"* >

</Button>

<Button

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"deleteBatch"*

android:text=*"Delete"* >

</Button>

</LinearLayout>

</TableRow>

</TableLayout>

</ScrollView>

**Updateclass.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<ScrollView xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"vertical"* >

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"* >

<TextView

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Batch Code "* />

<TextView

android:id=*"@+id/textBatchCode"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Hib2404"*

android:textSize=*"20sp"* >

</TextView>

</LinearLayout><ImageButton

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"showTimePicker"*

android:src=*"@drawable/datepicker"* >

</ImageButton>

<TextView

android:id=*"@+id/textClassTime"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:paddingRight=*"10dp"*

android:text=*"00:00"*

android:textSize=*"20sp"* >

</TextView>

</LinearLayout>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"* >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"100dp"*

android:layout\_height=*"wrap\_content"*

android:text=*"Period (Min)"* />

<EditText

android:id=*"@+id/editPeriod"*

android:layout\_width=*"50dp"*

android:layout\_height=*"wrap\_content"*

android:inputType=*"number"* >

</EditText></LinearLayout>

</LinearLayout>

</ScrollView>

**OUTPUT:**























