Chapter - 5

1. Menu

- Menus are a common user interface component in many types of applications.
- It is generally used to provide extra functionally to android application like setting, search, create folder etc.
- I android there are two types of menu available context menu and system menu.
- Context menu related to the android components and system menu is related to menu button of our android mobile.
- Menu items also contain some sub menu items that is known as expanded menu.

1.1 Difference between Context Menu and System Menu

Context Menu	System Menu
That menu shown by component long press event.	Menu showns by pressing menu button of Android Devices.
In provides action that affect the particular components only, like EditText, TextView, Button etc.	It's where you should place actions that have a global impact on the app, such as "search", "compose mail" and "setting".
To Create Option menu OnCreateContextMenu() method is used.	To create Option menu OnCreateOptionsMenu() method is used.
When the menu is selected, the onContextItemSelected() method will called by item.	When the menu is selected, the OnOptionItemSelected() method will called by item.
We need to register Android widget for Context menu using registerForContextMenu() method. for example, registerForContextMenu(editText1)	No need to register Activity for System menu.

1.2 Creating Context (Custom) Menu

- When we long press any component in android at that time this menu is appear.

Step:-

1) Create some menu item in res->menu->.xml file
2) Implement onCreateContextMenu() method of Activity class in .java file
3) Register the android Component for Context Menu items using registerForContextMenu() method.
4) We have onContextItemSelected() method to control the menu item's selection event.

Example:-

Menu>activity_main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/apple" android:title="Apple"></item>
    <item android:id="@+id/banana" android:title="Banana"></item>
    <item android:id="@+id/greapes" android:title="Greapes"></item>
    <item android:id="@+id/orange" android:title="Orange"></item>
    <item android:id="@+id/pineple" android:title="Pineple"></item>
```

```
</menu>
```

Activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <EditText
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:ems="10" >

        <requestFocus />
    </EditText>

    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/editText1"
        android:text="Add Item" />

    <TextView
        android:id="@+id/txtcart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_marginBottom="174dp"
        android:text="Large Text"
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```

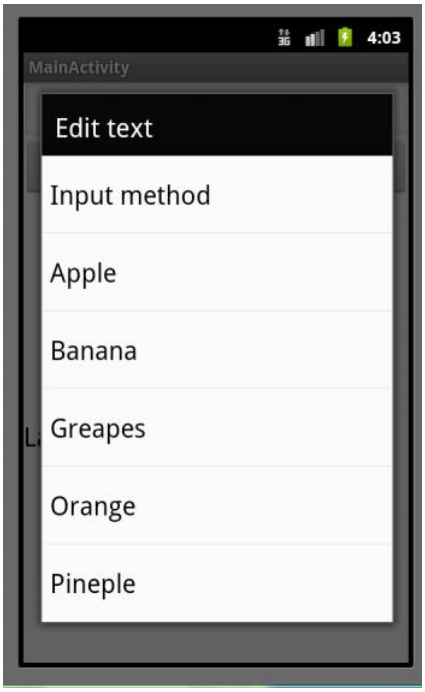
MainActivity.java

```
package com.example.contextmenudemo;

public class MainActivity extends Activity implements OnClickListener {


        EditText ed;

        Button bt;

        TextView cart;

        String str="In Cart :";

  @Override
```

```
public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   ed=(EditText) findViewById(R.id.editText1);
   //register
   registerForContextMenu(ed);
   bt=(Button) findViewById(R.id.button1);
   bt.setOnClickListener(this);
   cart=(TextView) findViewById(R.id.txtcart);
}
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                ContextMenuInfo menuInfo) {
        // TODO Auto-generated method stub
        super.onCreateContextMenu(menu, v, menuInfo);
        //menu connect with xml
        MenuInflater mi=getMenuInflater();
        mi.inflate(R.menu.activity_main, menu);
}
@Override
public boolean onContextItemSelected(MenuItem item) {
        //set value into edittext
        ed.setText(item.getTitle());
        return super.onContextItemSelected(item);
}
        public void onClick(View v) {
                str=str+ed.getText().toString();
                cart.setText(" "+str);
```

```
                ed.setText("");

        }

}
```

OUTPUT:



1.3 Creating and Use Handset menu Button (Hardware)

Step:-

1) Create some menu item in res->menu->.xml file
2) Implements onCreateOptionsMenu() method of Activity class to set Option Menu for Activity
3) Implement onOptionsItemSelected() method to handle the menu selection event, we can get the ID of selected menu item using item.getItemId() method.

Example:-

Menu>activity_main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/red" android:title="RED"
android:icon="@android:drawable/btn_star"></item>
    <item android:id="@+id/green" android:title="GREEN"
android:icon="@android:drawable/btn_plus"></item>
    <item android:id="@+id/blue" android:title="BLUE"
android:icon="@android:drawable/btn_minus"></item>
    <item android:id="@+id/black" android:title="BLACK"
android:icon="@android:drawable/btn_radio"></item>
</menu>
```

Activity_main.xml

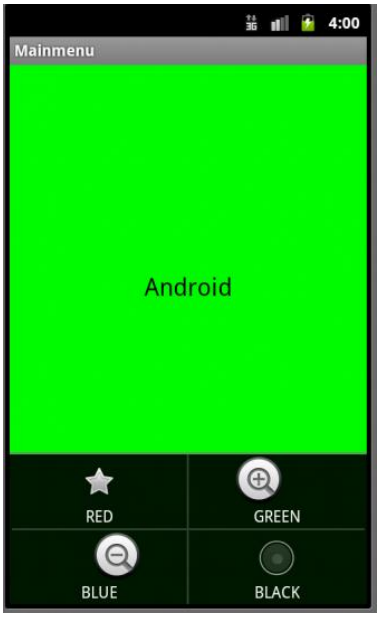
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="183dp"
        android:text="Android"
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```

Mainmenu.java

```
package com.example.systemmenudemo;

public class Mainmenu extends Activity {

        TextView tv;

  @Override

  public void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_mainmenu);

    tv=(TextView) findViewById(R.id.textView1);

  }

  @Override

  public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.activity_mainmenu, menu);

    return true;

  }

  @Override

  public boolean onOptionsItemSelected(MenuItem item) {

        View v=this.getWindow().getDecorView();

        if(item.getItemId()==R.id.red)
```

```
            v.setBackgroundColor(Color.RED);
        else if(item.getItemId()==R.id.green)
            v.setBackgroundColor(Color.GREEN);
        else if(item.getItemId()==R.id.blue)
            v.setBackgroundColor(Color.BLUE);
        else if(item.getItemId()==R.id.black)
            v.setBackgroundColor(Color.BLACK);
        return super.onOptionsItemSelected(item);
    }
}
```

Output:



2. Dialog: Creating and Alerting Dialogs

- A dialog box is a small window that prompt the user and user have to select one of the option from dialog box.
- There are so many type of dialog window available in android.
- In android you can use a dialog box with one, two, or, three buttons or none at all. You can also have a list of selectable items that consists of checkboxes or radio buttons.

2.1 Alert Dialog with OK CANCEL button

Step:-

1) Create a object of AlertDialog.Builder class
2) We need to set the message for Alert dialog for that we have setMessage() method of builder class.
3) If we want that user have to select on of option then we have a setCancelable() method of builder class
4) Now we need to create two button that is OK(positive) button and CANCEL(negative) button for that we have setPositiveButton() and setNegativeButton() method that take two arguments first is caption of button and second is the listener of the button that take care of click event generated by buttons. We have different OnClickListner of DialogInterface class for handle the onClick event.
5) At last create Alerting object and show the dialog box.

Example:-

Activity_main.xml

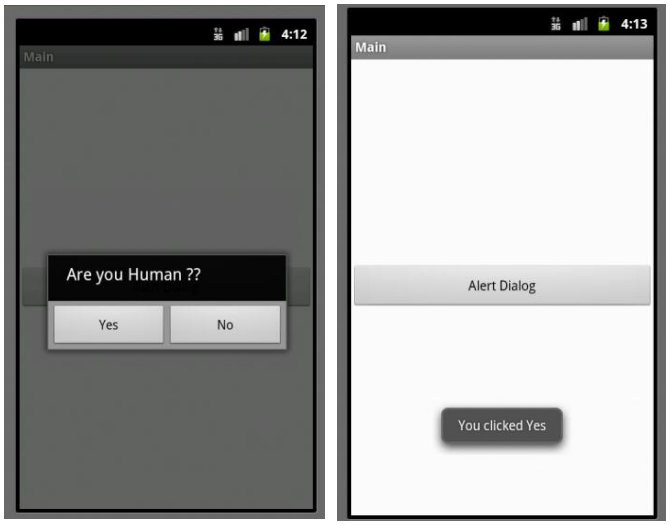
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_centerVertical="true"
        android:text="Alert Dialog" />

</RelativeLayout>
```

Main.java

```
package com.example.alertdialogdemo;

public class Main extends Activity implements OnClickListener{

Button btn;

   @Override

   public void onCreate(Bundle savedInstanceState) {

      super.onCreate(savedInstanceState);

      setContentView(R.layout.activity_main);

      btn=(Button) findViewById(R.id.button1);
```

```
        btn.setOnClickListener(this);
    }
        public void onClick(View v) {
                AlertDialog.Builder build=new AlertDialog.Builder(this);
                build.setMessage("Are you Human ??");
                build.setCancelable(false);
                build.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int which) {
                                Toast.makeText(Main.this, "You clicked Yes",
Toast.LENGTH_SHORT).show();
                        }
                });
                build.setNegativeButton("No", new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int which) {
                                Toast.makeText(Main.this, "You clicked No",
Toast.LENGTH_SHORT).show();
                        }
                });
                AlertDialog alert=build.create();
                alert.show();
        }
}
```

OUTPUT:



2.2 Progress Dialogbox

- This type of dialog shows progress bar, it is used to show some processing like download file, loading application etc.,

Step:-

1) Create a object of ProgressDialog class.
2) Set the message for dialog
3) Set the style for dialog
    a) Display horizontal progress bar
    b) Display round progress bar
4) Set the initial progress of dialog
5) Set the maximum value for ProgressDailog.
6) Show the dialog using show method.

Example :-

Activity_main_p.xml

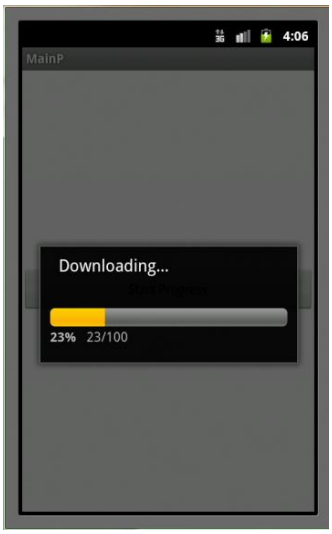
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_centerVertical="true"
        android:text="Start Progress" />

</RelativeLayout>
```

Main_P.java

```
package com.example.progressdialogdemo;

public class MainP extends Activity implements OnClickListener{
   Button start;
         @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main_p);
      start=(Button) findViewById(R.id.button1);
      start.setOnClickListener(this);


   }
         public void onClick(View v) {
                  final ProgressDialog pd=new ProgressDialog(this);
                  pd.setMessage("Downloading...");
                  pd.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
                  pd.setProgress(0);
                  pd.setMax(100);
                  pd.show();


                  final int totaltime=100;
                  Thread t=new Thread(){
                  @Override
                  public void run() {
                           int jumptime=0;
                           while(jumptime<totaltime)
                           {
                                    try{
```

```
                    sleep(200);
                    jumptime+=1;
                    pd.setProgress(jumptime);

                }catch(InterruptedException e)
                {
                    e.printStackTrace();
                }
            }
            super.run();
        }};
        t.start();
    }
}
```

OUTPUT:



2.3 Customized Dialog Box

- When we add some android components to dialog then it's a customized dialog box.
- For customization we need to create some components and add that to our dialog box.

Step:-

1) Create Linear layout to arrange the components in vertical direction
2) Create some components and add it to the Linear layout

3) Now add these components to linear layout using addView() method
4) Add Linear layout to our Dialog using setView() method.
5) Finally set positive and negative button for Dialog.

Example:-

Activity_cmain.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_centerVertical="true"
        android:text="Customized Dialog" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:text=""
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```

CMainActivity.java

```
package com.example.customisedialogdemo;

public class CMainActivity extends Activity implements OnClickListener{

   Button btn;

   TextView tv;

        @Override

   public void onCreate(Bundle savedInstanceState) {

      super.onCreate(savedInstanceState);

      setContentView(R.layout.activity_cmain);

      btn=(Button) findViewById(R.id.button1);

      btn.setOnClickListener(this);

      tv=(TextView) findViewById(R.id.textView1);
```

```
}
        public void onClick(View v) {
                LinearLayout ll=new LinearLayout(this);
                ll.setOrientation(LinearLayout.VERTICAL);

                final EditText ed=new EditText(this);
                final CheckBox chk1=new CheckBox(this);
                chk1.setText("Easy");
                chk1.setTextColor(Color.WHITE);
                final CheckBox chk2=new CheckBox(this);
                chk2.setText("Hard");
                chk2.setTextColor(Color.WHITE);
                final RatingBar rb=new RatingBar(this);
                rb.setRating(0);
                rb.setNumStars(5);

                ll.addView(ed);
                ll.addView(chk1);
                ll.addView(chk2);
                ll.addView(rb);

                AlertDialog.Builder build=new AlertDialog.Builder(this);
                build.setMessage("Enter Name and Rating");
                build.setCancelable(false);

                build.setView(ll);
                build.setPositiveButton("Submit", new DialogInterface.OnClickListener() {
```

```
                public void onClick(DialogInterface dialog, int which) {
                        String str;
                        str="Name:"+ed.getText().toString();

                        if(chk1.isChecked())
                                str=str+"\n Level:Easy";
                        if(chk2.isChecked())
                                str=str+"\n Level:Hard";
                        str=str+"\n Rating:"+rb.getRating();
                        tv.setText(str);
                }
        });
        AlertDialog alert=build.create();
        alert.show();
    }
}
```

OUTPUT:

3. Toast

- Toast is small message that appear on the scree for some moment
- It is used to display message like error message, when user complete some process etc.

- To create a Toast message we have Toast class and mekeText() function
  Syntax:
  Toast.makeText(context, message,duration).show();
- makeText() : this function will make a text to display it has three arguments
- context : it tells that for which activity this message is for
- message : actual message to display on screen
- duration : duration for the message, there are two duration provided by Toast class Toast.LENGTH_LONG and Toast.LENGTH_SHORT

Example:-

```
Toast.makeText(this, “Your Message Here.”, Toast.LENGTH_LONG).show();
```

4. Basic operation of SQLite Database

4.1 Introduction of SQLite database

- SQLite is at the heart of Android’s database support.
- SQLite is open source SQL database that stores data to a text file on a device.
- Android comes in with built in SQLite database implementation.
- SQLite designed by D. Richard Hipp in the year 2000 for the purpose of no administration required for operating a program.
- Android supports SQLite, which provides a relational database for a mobile device i. e it contains tables (consisting of rows and columns), indexes etc.
- This database was created for embedded environment like Android.
- SQLite does not require a separate server process or system to operate.
- We can perform different query operations in SQLite like Oracle.
- Below are some datatypes supported by SQLite database.

Data Type	Description
NULL	Contain NULL value
INTEGER	Stores integer value
REAL	stores floating point value
TEXT	Stores text value, it also store date and time value.
BLOB	It will store binary data

4.2 SQLite – Query

Create Table Query

- This query will create a table in SQLite database.
- Syntax:

```
CREATE TABLE table_name( column1 datatype PRIMARY KEY,
                         column2 datatype NOT NULL,
```

column3 datatype NOT NULL,
.....
);

- Example:
  CREATE TABLE std_info(no INTEGER PRIMARY KEY, name TEXT);

Drop Table Query

- This query will drop a table in SQLite database.
- Syntax:
  DROP TABLE table_name;
- Example:
  DROP TABLE std_info;

Insert Query

- This query inserts data into table.
- Syntax:
  INSERT INTO table_name(column1,column2,column3,….) VALUES (value1, value2, value3,…);
- Example:
  INSERT INTO std_info(no, name) VALUES (10, 'std1');

Delete Query

- This query delete data from table.
- Syntax:
  DELETE FROM table_name WHERE column_name=value ;
  For delete all record
  DELETE FROM table_name;
- Example:
  DELETE FROM std_info where no=1;

Update Query

- This query update record of table.
- Syntax:
  UPDATE table_name SET column_name=value WHERE column_name=value ;
- Example:
  UPDATE std_info SET name="std2" WHERE where no=1;

Select Query

- This query used to fetch record from database table.
- Syntax:
  1) SELCET * FROM table_name;

2) SELECT column_name FROM table_name;
3) SELECT column_name FROM table_name WHERE column_name=value ;

- Example:
  1) SELCET * FROM std_info;
  2) SELECT no FROM std_info;
  3) SELECT name FROM std_info WHERE no=1;

4.3 SQLite : Open Helper and Create Database

SQLiteOpenHelper class

- Sqlite helper class helps us to manage database creation and version management.
- SQLiteOpenHelper takes care of all database management activities. To use it , override onCreate(), onUpgrade() methods.

Create Database

- To create database use constructor of SQLiteOpenHelper class for example,
  Public SQLHelper(Context context){
  Super(context,"student",null,1);
  }
- Here, student is a database name and 1 is a version for that database.
- Note:- Database actually created when we make an object of this helper class in our Activity class.
- SQLiteOpenHelper class have two method
  1) getwritableDatabase() : when we want to read and write the database then use this method, it will open database in write mode.
  2) getReadableDatabase() : when we want to read the database then use this method, it will open database in readable mode.

Step:-

1) create a new java class and extends the SQLiteOpenHelper class, in that implements methods of class, create constructor and create two string variable for database name and version name.
2) To create a table , in onCreate() method make a query and to run query use execSQL() method.
3) Make a object of MySQLHelper class in our Activity class, remember that when we make a object constructor of that class called automatically so our database will be created and onCreate() method also called automatically so our table is also generated in our database.

Example:-

Activity_mail_sql.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <EditText
```

```
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:ems="10"
        android:hint="En. No." >

        <requestFocus />
    </EditText>

    <EditText
        android:id="@+id/editText2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Name" />

    <EditText
        android:id="@+id/editText3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="22dp"
        android:ems="10"
        android:hint="City" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/button1"
            android:layout_width="77dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.06"
            android:text="Insert" />

        <Button
            android:id="@+id/button2"
            android:layout_width="83dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.02"
            android:text="Delete" />

        <Button
            android:id="@+id/button3"
            android:layout_width="85dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.02"
            android:text="Update" />

    </LinearLayout>

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

</LinearLayout>
```

Mysqlhelper.java

```
package com.example.sqldemo;

public class Mysqlhelper extends SQLiteOpenHelper {

	public Mysqlhelper(Context context) {

		super(context, "student.db", null, 1);

	}

	@Override

	public void onCreate(SQLiteDatabase db) {

		String q="create table studentinfo(ENNO integer(5), name varchar(20), city
varchar(20));";

		db.execSQL(q);

	}


	@Override

	public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

		// TODO Auto-generated method stub

	}

}
```

MainSql.java

```
package com.example.sqldemo;

public class MainSql extends Activity implements OnClickListener{


	EditText no,name,city;

	Button ins,del,update;

	TextView tv;

	Mysqlhelper helper;

	SQLiteDatabase database;


	@Override
```

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main_sql);
    no=(EditText) findViewById(R.id.editText1);
   name=(EditText) findViewById(R.id.editText2);
  city=(EditText) findViewById(R.id.editText3);
    ins=(Button) findViewById(R.id.button1);
    ins.setOnClickListener(this);
    del=(Button) findViewById(R.id.button2);
    del.setOnClickListener(this);
    update=(Button) findViewById(R.id.button3);
    update.setOnClickListener(this);
     tv=(TextView) findViewById(R.id.textView1);

    helper=new Mysqlhelper(this);
    database=helper. getWritableDatabase();
     //Display Data
    Cursor c=database.rawQuery("select * from studentinfo",null);
                    if(c!=null)
                    {
                            String result="ENNO\t\tName\t\tCity\n";
                            int noindex=c.getColumnIndex("ENNO");
                            int nameindex=c.getColumnIndex("name");
                            int cityindex=c.getColumnIndex("city");
                            c.moveToFirst();
                            do{
                                    String no=c.getString(noindex);
                                    result=result+no+"\t\t\t\t";
```

```
                String name=c.getString(nameindex);
                result=result+name+"\t\t\t";
                String city=c.getString(cityindex);
                result=result+city+"\n";
            }while(c.moveToNext());
            tv.setText(result);
        }
        else
        {
            Toast.makeText(this, "No data Available", Toast.LENGTH_SHORT).show();
        }

}
    public void onClick(View v) {

        if(v.getId()==R.id.button1)
        {
            String sno=no.getText().toString();
            Integer ino=Integer.parseInt(sno);
        String sname=name.getText().toString();
        String scity=city.getText().toString();
        String query="insert into studentinfo values("+ino+",'"+sname+"','"+scity+"');";
        database.execSQL(query);
        Toast.makeText(this, "inserted Successfully", Toast.LENGTH_SHORT).show();
        }
        else if(v.getId()==R.id.button2)
        {
            String sno=no.getText().toString();
```

```
                Integer ino=Integer.parseInt(sno);
                String query="delete from studentinfo where ENNO="+ino+";";
                database.execSQL(query);
                Toast.makeText(this, "Deleted Successfully", Toast.LENGTH_SHORT).show();
            }
            else if(v.getId()==R.id.button3)
            {
                String sno=no.getText().toString();
                Integer ino=Integer.parseInt(sno);
                String sname=name.getText().toString();
                String scity=city.getText().toString();
                String query="update studentinfo set name='"+sname+"' where
ENNO="+ino+";";
                database.execSQL(query);
                Toast.makeText(this, "Updated Successfully", Toast.LENGTH_SHORT).show();
            }
             Cursor c=database.rawQuery("select * from studentinfo",null);
                if(c!=null)
                {
                    String result="ENNO\t\tName\t\tCity\n";
                    int noindex=c.getColumnIndex("ENNO");
                    int nameindex=c.getColumnIndex("name");
                    int cityindex=c.getColumnIndex("city");
                    c.moveToFirst();
                    do{
                        String no=c.getString(noindex);
                        result=result+no+"\t\t\t\t";
                        String name=c.getString(nameindex);
                        result=result+name+"\t\t\t";
```

```
                    String city=c.getString(cityindex);
                    result=result+city+"\n";
                }while(c.moveToNext());
                tv.setText(result);
            }
        no.setText("");
        name.setText("");
        city.setText("");

    }
}
```

OUTPUT: