Name : Mujahid Ullah

Subject: Android Development

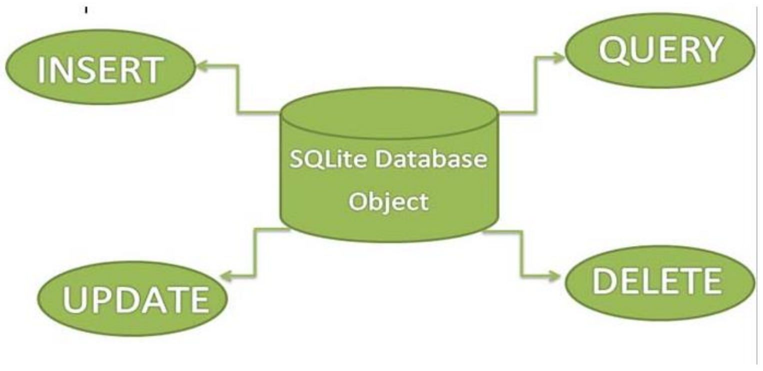
Submitted To Sir Junaid

Date : 28-Dec-20

Assignment No 7

1) Sql Lite Data BaseWith Example:

SQLite is a Structure query base database, open source, light weight, no network access and standalone database. It support embedded relational database features.



Android has built in SQLite database implementation. It is available locally over the device(mobile & tablet) and contain data in text format. It carry light weight data and suitable with many languages. So, it doesn’t required any administration or setup procedure of the database.

Important Note – The database created is saved in a directory: data/data/APP_Name/databases/DATABASE_NAME.

This Android SQLite Database Example will cover Creating Database, Creating Tables, Creating Records, Reading Records, Updating Records and Deleting Records in Android SQLite Database. After going through this post you will be having a complete idea about using SQLite database for your

We will also add some CRUD methods.

- addStudent
- getStudent
- getAlStudent
- updateStudent
- deleteStudent
- delelteAllStudent

All above methods will interact with SQLite database and perform CRUD operations.

Step 1: Create Project

File -> new-> Create Project

Step 2: Past Code In activityMain. XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:layout_margin="20dp"
    android:background="#ffff"
    android:padding="20dp"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/e_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Name"
        android:inputType="text"
        android:textColorHint="#3700B3" />

    <EditText
        android:id="@+id/e_email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress"
        android:textColorHint="#3700B3" />

    <EditText
        android:id="@+id/e_username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
```

```
            android:inputType="text"
            android:textColorHint="#3700B3" />

        <EditText
            android:id="@+id/e_phnumber"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Phone Number"
            android:inputType="number"
            android:textColorHint="#3700B3" />


        <Button
            android:id="@+id/btn_save"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:background="#03D4AC"
            android:text="Save" />

        <Button
            android:id="@+id/viewdata_in_recycler"
            android:layout_margin="20dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:background="#03D4AC"
            android:text="View data(RecycleView)" />

        <Button
            android:id="@+id/viewdataLL"
            android:layout_margin="20dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:background="#03D4AC"
            android:text="View data(ListView)" />
</LinearLayout>
```

Step 4: Main_Act.java code

```
package com.cal.sql_litewithrecyclerviewandlistview;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class MainActivity extends AppCompatActivity {

    EditText e_name, e_email, e_username, e_phnumber;
    Button bt_save,viewdata_in_recycler,viewdatall;
    public static final String DATABASE_NAME = "StudentDatabases.db";
    SQLiteDatabase mDatabase;

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

        mDatabase = openOrCreateDatabase(DATABASE_NAME, MODE_PRIVATE, null);
        createEmployeeTable();


        //FindById (Button and Edittxt)
        e_name = (EditText) findViewById(R.id.e_name);
        e_email = (EditText) findViewById(R.id.e_email);
        e_username = (EditText) findViewById(R.id.e_username);
        e_phnumber = (EditText) findViewById(R.id.e_phnumber);

        bt_save = (Button) findViewById(R.id.btn_save);
         viewdatall= (Button) findViewById(R.id.viewdataLL);
        viewdata_in_recycler= (Button) findViewById(R.id.viewdata_in_recycler);


        viewdata_in_recycler.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent= new Intent(MainActivity.this,
Employee_Recycler_view.class);
                startActivity(intent);
            }
        });

        //Onclick Btn
        bt_save.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                //Get the Enter data
                String name = e_name.getText().toString().trim();
                String email = e_email.getText().toString().trim();
                String username = e_username.getText().toString();
                String phone = e_phnumber.getText().toString();


                if (name.isEmpty() || email.isEmpty() || username.isEmpty() ||
phone.isEmpty()) {

                    Toast.makeText(MainActivity.this, "Fil the form",
Toast.LENGTH_SHORT).show();

                } else {

                    String insertSQL = "INSERT INTO Student \n" +
                            "(Name, Email, UserName, PhoneNo)\n" +
                            "VALUES \n" +
                            "(?, ?, ?, ?);";

                    //using the same method execsql for inserting values
                    //this time it has two parameters
                    //first is the sql string and second is the parameters that is
to be binded with the query
                    mDatabase.execSQL(insertSQL, new String[]{name, email,
username, phone});

                    Toast.makeText(MainActivity.this, "Great! Data Saved",
Toast.LENGTH_SHORT).show();
                }


            }
        });


        viewdatall.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent= new Intent(MainActivity.this,
Employee_Listview_Activity.class);
                startActivity(intent);
            }
        });

    }

    private void createEmployeeTable() {
        mDatabase.execSQL("CREATE TABLE IF NOT EXISTS Student " +
                "(\n" +
                "    id INTEGER NOT NULL CONSTRAINT employees_pk PRIMARY KEY
AUTOINCREMENT,\n" +
                "    Name varchar(200) NOT NULL,\n" +
                "    Email varchar(200) NOT NULL,\n" +
                "    UserName varchar(200) NOT NULL,\n" +
                "    PhoneNo Varchar(200) NOT NULL\n" +
                ");"

        );
    }

}
```

Step 3: Create Model Class

```
package com.cal.sql_litewithrecyclerviewandlistview;

public class Model_Class {

    private int id;
    private String mname;
    private String musername;
    private String memail;
    private String mphno;
    public int getId() {
        return id;
    }

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

    public String getMname() {
        return mname;
    }

    public void setMname(String mname) {
        this.mname = mname;
    }

    public String getMusername() {
        return musername;
    }

    public void setMusername(String musername) {
        this.musername = musername;
    }

    public String getMemail() {
        return memail;
    }

    public void setMemail(String memail) {
        this.memail = memail;
    }

    public String getMphno() {
        return mphno;
    }

    public void setMphno(String mphno) {
        this.mphno = mphno;
    }

    public Model_Class(int id, String mname, String musername, String memail,
String mphno) {
        this.id = id;
        this.mname = mname;
        this.musername = musername;
        this.memail = memail;
        this.mphno = mphno;
    }

    public Model_Class(){}
}
```

Step5: Adapter For ListView

```java
package com.cal.sql_litewithrecyclerviewandlistview;

import android.content.Context;
import android.content.DialogInterface;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AlertDialog;

import java.util.List;

public class EmployeeAdapter_listview extends ArrayAdapter<Model_Class> {

    Context mCtx;
    int listLayoutRes;
    List<Model_Class> employeeList;
    SQLiteDatabase mDatabase;

    public EmployeeAdapter_listview(Context mCtx, int listLayoutRes,
List<Model_Class> employeeList, SQLiteDatabase mDatabase) {
        super(mCtx, listLayoutRes,employeeList);
        this.mCtx = mCtx;
        this.listLayoutRes = listLayoutRes;
        this.employeeList = employeeList;
        this.mDatabase = mDatabase;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull
ViewGroup parent) {

        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(listLayoutRes, null);


        final Model_Class employee = employeeList.get(position);
        TextView textViewName = view.findViewById(R.id.textViewName);
        TextView textViewUsername = view.findViewById(R.id.textViewDepartment);
        TextView textViewEmail = view.findViewById(R.id.textViewSalary);
        TextView textViewphno = view.findViewById(R.id.textViewJoiningDate);


        textViewName.setText(employee.getMname());
        textViewUsername.setText(employee.getMusername());
        textViewEmail.setText(String.valueOf(employee.getMemail()));
        textViewphno.setText(employee.getMphno());
        ImageView buttonDelete = view.findViewById(R.id.buttonDeleteEmployee);
        ImageView buttonEdit = view.findViewById(R.id.buttonEditEmployee);

//adding a clicklistener to button
        buttonEdit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                updateEmployee(employee);
            }
        });

        //the delete operation
        buttonDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);
                builder.setTitle("Are you sure?");
                builder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {
                        String sql = "DELETE FROM Student WHERE id = ?";
                        mDatabase.execSQL(sql, new Integer[]{employee.getId()});
                        reloadEmployeesFromDatabase();
                    }
                });
                builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {

                    }
                });
                AlertDialog dialog = builder.create();
                dialog.show();
            }
        });

        return view;
    }



    private void updateEmployee(final Model_Class employee) {
        final AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);

        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(R.layout.dialog_update_employee, null);
        builder.setView(view);


        final EditText editTextName = view.findViewById(R.id.editTextName);
        final EditText editusername = view.findViewById(R.id.editUsername);
        final EditText editemail = view.findViewById(R.id.editEmail);
        final EditText editphno = view.findViewById(R.id.editTextphno);

        editTextName.setText(employee.getMname());
        editusername.setText(employee.getMusername());
        editemail.setText(employee.getMemail());
        editphno.setText(employee.getMphno());


        final AlertDialog dialog = builder.create();
        dialog.show();

        // CREATE METHOD FOR EDIT THE FORM
        view.findViewById(R.id.buttonUpdateEmployee).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String name = editTextName.getText().toString().trim();
                String username = editusername.getText().toString().trim();
                String email = editemail.getText().toString().trim();
                String phno = editphno.getText().toString().trim();

                if (name.isEmpty()) {
                    editTextName.setError("Name can't be blank");
                    editTextName.requestFocus();
                    return;
                }

                if (username.isEmpty()) {
                    editusername.setError("Username can't be blank");
                    editusername.requestFocus();
                    return;
                }

                String sql = "UPDATE Student \n" +
                        "SET Name = ?, \n" +
                        "Email = ?,\n"+
                        "Username = ?,\n"+
                        "PhoneNO= ? \n" +
                        "WHERE id = ?;\n";

                mDatabase.execSQL(sql, new String[]{name, email,username,phno,
String.valueOf(employee.getId())});
                Toast.makeText(mCtx, "Student Updated",
Toast.LENGTH_SHORT).show();
                reloadEmployeesFromDatabase();

                dialog.dismiss();
            }
        });
    }

    private void reloadEmployeesFromDatabase() {
        Cursor cursorEmployees = mDatabase.rawQuery("SELECT * FROM Student",
null);
        if (cursorEmployees.moveToFirst()) {
            employeeList.clear();
            do {
                employeeList.add(new Model_Class(
                        cursorEmployees.getInt(0),
                        cursorEmployees.getString(1),
                        cursorEmployees.getString(2),
                        cursorEmployees.getString(3),
                        cursorEmployees.getString(4)
                ));
            } while (cursorEmployees.moveToNext());
        }
        cursorEmployees.close();
        notifyDataSetChanged();
    }

}
```

Step6: ListView_Ativity.java code:

```
package com.cal.sql_litewithrecyclerviewandlistview;

import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.widget.ListView;

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

public class Employee_Listview_Activity extends AppCompatActivity {

    List<Model_Class> employeeList;
    SQLiteDatabase mDatabase;
    ListView listViewEmployees;
    EmployeeAdapter_listview adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_employee__listview_);
        listViewEmployees = (ListView) findViewById(R.id.listViewEmployees);
        employeeList = new ArrayList<>();

        //opening the database
        mDatabase = openOrCreateDatabase(MainActivity.DATABASE_NAME, MODE_PRIVATE,
null);

        //this method will display the employees in the list
        showEmployeesFromDatabase();
    }

    private void showEmployeesFromDatabase() {
        //we used rawQuery(sql, selectionargs) for fetching all the employees
        Cursor cursorEmployees = mDatabase.rawQuery("SELECT * FROM Student",
null);

        //if the cursor has some data
        if (cursorEmployees.moveToFirst()) {
            //looping through all the records
            do {
                //pushing each record in the employee list
                employeeList.add(new Model_Class(
                        cursorEmployees.getInt(0),
                        cursorEmployees.getString(1),
                        cursorEmployees.getString(2),
                        cursorEmployees.getString(3),
                        cursorEmployees.getString(4)
                ));
            } while (cursorEmployees.moveToNext());
        }
        //closing the cursor
        cursorEmployees.close();

        //creating the adapter object
        adapter = new EmployeeAdapter_listview(this,
R.layout.list_layout_employee, employeeList, mDatabase);

        //adding the adapter to listview
        listViewEmployees.setAdapter(adapter);
    }


}
```

Step7: Adapter_for_ReyclerView

```
package com.cal.sql_litewithrecyclerviewandlistview;

import android.content.Context;
import android.content.DialogInterface;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.recyclerview.widget.RecyclerView;

import com.google.android.material.snackbar.Snackbar;

import java.util.List;

public class Adapter_Employee_RecyclerView extends
RecyclerView.Adapter<Adapter_Employee_RecyclerView.ProductViewHolder>   {
    int custom_list_item;
    SQLiteDatabase mDatabase;
    //this context we will use to inflate the layout
    private Context mCtx;

    //we are storing all the products in a list
    private List<Model_Class> productList;

    public Adapter_Employee_RecyclerView(Context mCtx, List<Model_Class>
productList,int custom_list_item, SQLiteDatabase mDatabase ) {
        this.custom_list_item = custom_list_item;
        this.mDatabase = mDatabase;
        this.mCtx = mCtx;
        this.productList = productList;
    }

    //getting the context and product list with constructor

    @NonNull
    @Override
    public Adapter_Employee_RecyclerView.ProductViewHolder
onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        //inflating and returning our view holder
        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(R.layout.custom_list_item, null);
        return new ProductViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull
Adapter_Employee_RecyclerView.ProductViewHolder holder, int position) {


        //getting the product of the specified position
        final Model_Class product = productList.get(position);

        //binding the data with the viewholder views
        holder.textViewName.setText(product.getMname());
        holder.textViewUsername.setText(product.getMusername());
        holder.textViewEmail.setText(product.getMemail());
        holder.textViewPhone.setText(product.getMphno());


        holder.editbtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                updateEmployee(product);
            }
        });

        holder.deletebtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(final View view) {
                AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);
                builder.setTitle("Are you sure?");
                builder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {
                        String sql = "DELETE FROM Student WHERE id = ?";
                        mDatabase.execSQL(sql, new Integer[]{product.getId()});
                        Snackbar.make(view, "Deleted" + product.getMname(),
Snackbar.LENGTH_SHORT).show();
                        Toast.makeText(mCtx, "Deleted successfully!",
Toast.LENGTH_SHORT).show();

                        reloadEmployeesFromDatabase(); //Reload List
                    }
                });
                builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {

                    }
                });
                AlertDialog dialog = builder.create();
                dialog.show();
            }
        });


    }

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

    public class ProductViewHolder extends RecyclerView.ViewHolder {

        TextView textViewName, textViewUsername, textViewEmail, textViewPhone;
        ImageView editbtn, deletebtn;

        public ProductViewHolder(@NonNull View itemView) {
            super(itemView);
            textViewName = itemView.findViewById(R.id.textViewName);
            textViewUsername = itemView.findViewById(R.id.textViewUsername);
            textViewEmail = itemView.findViewById(R.id.textViewEmail);
            textViewPhone = itemView.findViewById(R.id.textViewPhone);

            deletebtn = itemView.findViewById(R.id.buttonDeleteStudent);
            editbtn = itemView.findViewById(R.id.buttonEditstudent);

        }
    }



    private void updateEmployee(final Model_Class product) {
        final AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);

        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(R.layout.dialog_update_employee, null);
        builder.setView(view);


        final EditText editTextName = view.findViewById(R.id.editTextName);
        final EditText editUsername = view.findViewById(R.id.editUsername);
        final EditText editemail = view.findViewById(R.id.editEmail);
        final EditText editphno = view.findViewById(R.id.editTextphno);



        editTextName.setText(product.getMname());
        editUsername.setText(product.getMusername());
        editemail.setText(product.getMemail());
        editphno.setText(product.getMphno());

        final AlertDialog dialog = builder.create();
        dialog.show();

        // CREATE METHOD FOR EDIT THE FORM
        view.findViewById(R.id.buttonUpdateEmployee).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String name = editTextName.getText().toString().trim();
                String email = editemail.getText().toString().trim();
                String username = editUsername.getText().toString().trim();
                String phno = editphno.getText().toString().trim();



                if (name.isEmpty()) {
                    editTextName.setError("Name can't be blank");
                    editTextName.requestFocus();
                    return;
                }

                if (username.isEmpty()) {
                    editUsername.setError("Salary can't be blank");
                    editUsername.requestFocus();
                    return;
                }//Name, Email, UserName, PhoneNo

                String sql = "UPDATE Student \n" +
                        "SET Name = ?, \n" +
                        "Email = ?,\n"+
                        "Username = ?,\n"+
                        " PhoneNO= ? \n" +
                        "WHERE id = ?;\n";

                mDatabase.execSQL(sql, new String[]{name, email,username,phno,
String.valueOf(product.getId())});
                Toast.makeText(mCtx, "Student Updated",
Toast.LENGTH_SHORT).show();

                dialog.dismiss();
            }
        });
    }

    void reloadEmployeesFromDatabase(){

        Cursor cursorproduct1 = mDatabase.rawQuery("SELECT * FROM Student", null);
        if (cursorproduct1.moveToFirst()) {
            productList.clear();
            do {
                productList.add(new Model_Class(
                        cursorproduct1.getInt(0),
                        cursorproduct1.getString(1),
                        cursorproduct1.getString(2),
                        cursorproduct1.getString(3),
                        cursorproduct1.getString(4)   ));
            } while (cursorproduct1.moveToNext());
        }
        cursorproduct1.close();
        notifyDataSetChanged();
    }
}
```

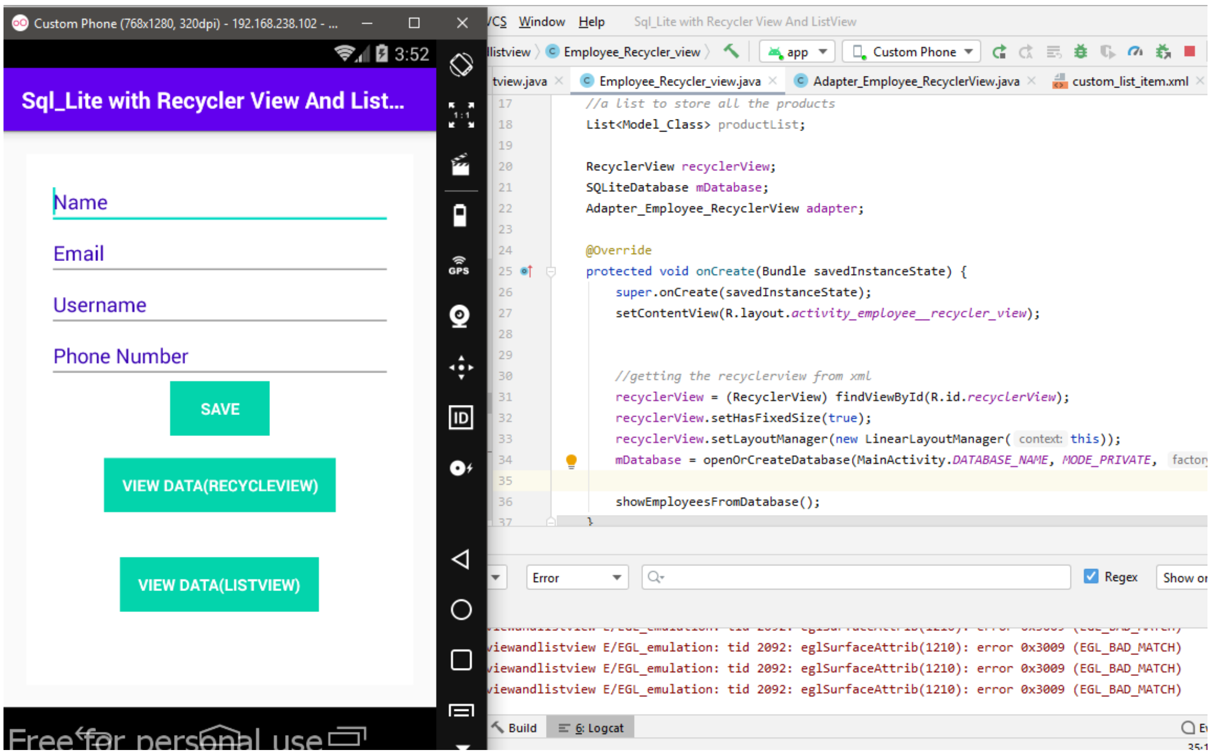
Step 8 :RecycerView Dependency

```
implementation "androidx.recyclerview:recyclerview:1.1.0"
```

Step 9 : RecylcerView_Activity.java Code

```
package com.cal.sql_litewithrecyclerviewandlistview;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.widget.Toast;

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

public class Employee_Recycler_view extends AppCompatActivity {

    //a list to store all the products
    List<Model_Class> productList;

    RecyclerView recyclerView;
    SQLiteDatabase mDatabase;
    Adapter_Employee_RecyclerView adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_employee__recycler_view);
        //getting the recyclerview from xml
        recyclerView = (RecyclerView) findViewById(R.id.recyclerView);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        mDatabase = openOrCreateDatabase(MainActivity.DATABASE_NAME, MODE_PRIVATE,
null);

        showEmployeesFromDatabase();
    }

    private void showEmployeesFromDatabase() {
        //we used rawQuery(sql, selectionargs) for fetching all the employees
        Cursor cursorproduct = mDatabase.rawQuery("SELECT * FROM Student", null);
        List<Model_Class> productList = new ArrayList<>();

        //if the cursor has some data
        if (cursorproduct.moveToFirst()) {
            //looping through all the records
            do {
                //pushing each record in the employee list
                productList.add(new Model_Class(
                        cursorproduct.getInt(0),
                        cursorproduct.getString(1),
                        cursorproduct.getString(2),
                        cursorproduct.getString(3),
                        cursorproduct.getString(4)
                ));
            } while (cursorproduct.moveToNext());
        }
        if (productList.isEmpty()) {
            Toast.makeText(this, "No items Found in database",
Toast.LENGTH_SHORT).show();
        }
        //closing the cursor
        cursorproduct.close();

        //creating the adapter object
        adapter = new Adapter_Employee_RecyclerView(this,
productList,R.layout.custom_list_item, mDatabase);

        //adding the adapter to listview
        recyclerView.setAdapter(adapter);

        adapter.reloadEmployeesFromDatabase();  //this method is in prdouctadapter
    }

}
```

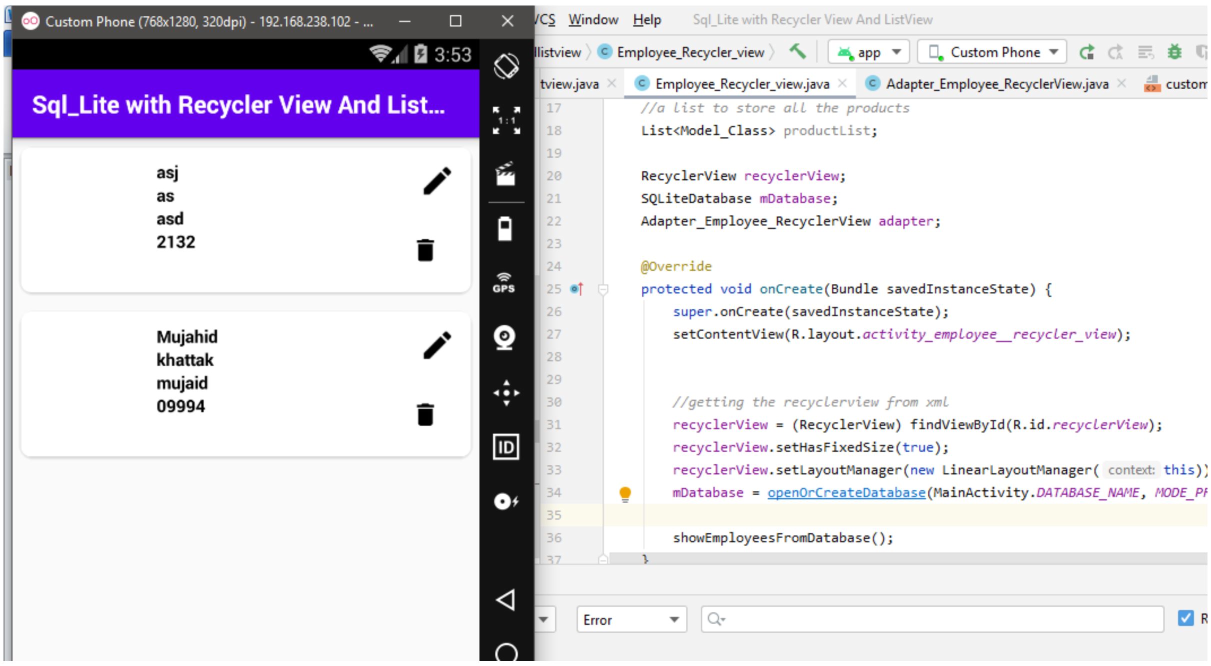
Step 10 :Now Run The The Emulator


Custom Phone (768x1280, 320dpi) - 192.168.238.102 - ...
3:52
Sql_Lite with Recycler View And List...
Name
Email
Username
Phone Number
SAVE
VIEW DATA(RECYCLEVIEW)
VIEW DATA(LISTVIEW)
Free for personal use
VCS Window Help Sql_Lite with Recycler View And ListView
listview Employee_Recycler_view app Custom Phone
tview.java Employee_Recycler_view.java Adapter_Employee_RecyclerView.java custom_list_item.xml
Error Regex Show on
Build 6: Logcat

```
    //a list to store all the products
    List<Model_Class> productList;

    RecyclerView recyclerView;
    SQLiteDatabase mDatabase;
    Adapter_Employee_RecyclerView adapter;

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


        //getting the recyclerview from xml
        recyclerView = (RecyclerView) findViewById(R.id.recyclerView);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager( context: this));
        mDatabase = openOrCreateDatabase(MainActivity.DATABASE_NAME, MODE_PRIVATE,  factory

        showEmployeesFromDatabase();
    }
```

```
viewandlistview E/EGL_emulation: tid 2092: eglSurfaceAttrib(1210): error 0x3009 (EGL_BAD_MATCH)
viewandlistview E/EGL_emulation: tid 2092: eglSurfaceAttrib(1210): error 0x3009 (EGL_BAD_MATCH)
viewandlistview E/EGL_emulation: tid 2092: eglSurfaceAttrib(1210): error 0x3009 (EGL_BAD_MATCH)
```

When Click On RecylcerView show Button

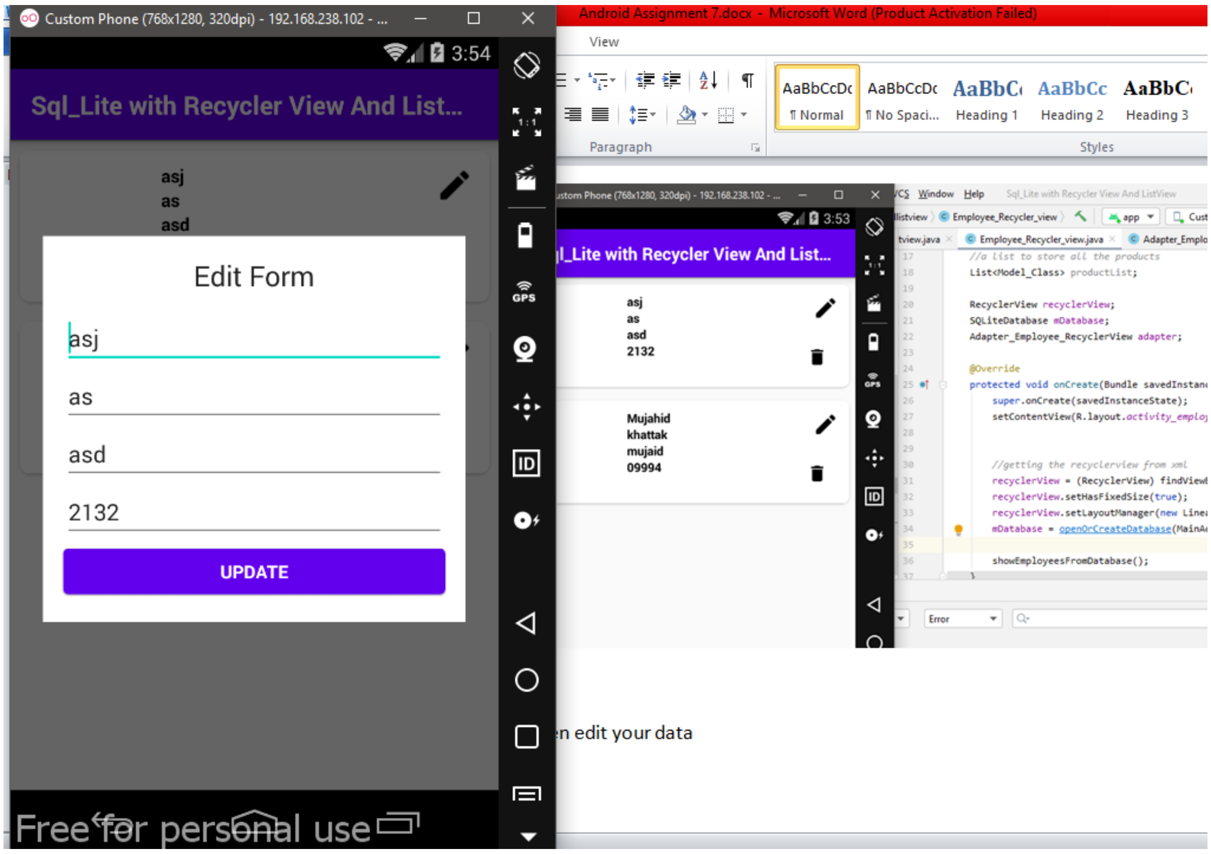

Custom Phone (768x1280, 320dpi) - 192.168.238.102 - ...
3:53
Sql_Lite with Recycler View And List...
VCS Window Help Sql_Lite with Recycler View And ListView
listview Employee_Recycler_view app Custom Phone
tview.java Employee_Recycler_view.java Adapter_Employee_RecyclerView.java custom
Error

asj
as
asd
2132

Mujahid
khattak
mujaid
09994

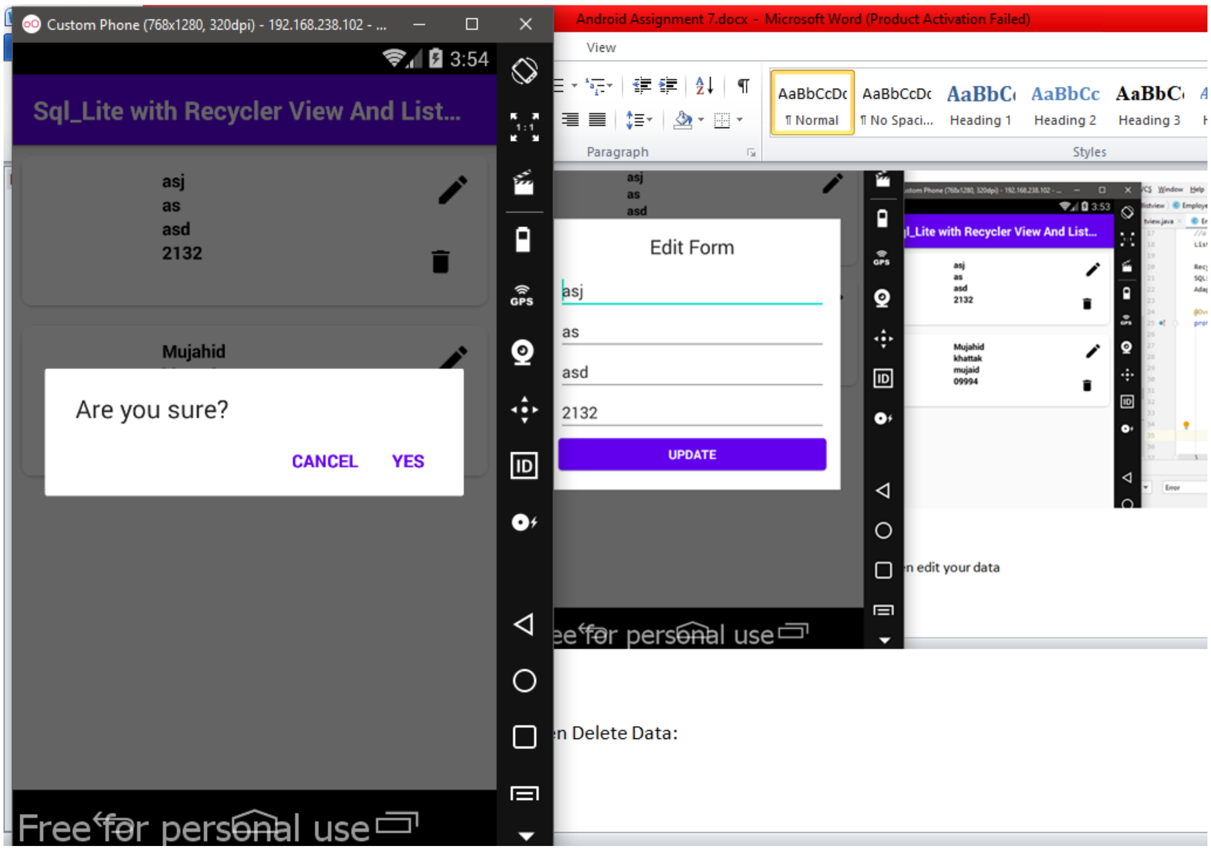
```
    //a list to store all the products
    List<Model_Class> productList;

    RecyclerView recyclerView;
    SQLiteDatabase mDatabase;
    Adapter_Employee_RecyclerView adapter;

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


        //getting the recyclerview from xml
        recyclerView = (RecyclerView) findViewById(R.id.recyclerView);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager( context: this)
        mDatabase = openOrCreateDatabase(MainActivity.DATABASE_NAME, MODE_PR

        showEmployeesFromDatabase();
    }
```

When edit your data


Custom Phone (768x1280, 320dpi) - 192.168.238.102 - ...
3:54
Sql_Lite with Recycler View And List...
asj
as
asd
Edit Form
asj
as
asd
2132
UPDATE
Free for personal use
Android Assignment 7.docx - Microsoft Word (Product Activation Failed)
View
Paragraph
Styles
¶ Normal
¶ No Spaci...
Heading 1
Heading 2
Heading 3
Mujahid
khattak
mujaid
09994
Employee_Recycler_view.java
Adapter_Emplo
Error
n edit your data

```
//a list to store all the products
List<Model_Class> productList;

RecyclerView recyclerView;
SQLiteDatabase mDatabase;
Adapter_Employee_RecyclerView adapter;

@Override
protected void onCreate(Bundle savedInstan
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_emplo


    //getting the recyclerview from xml
    recyclerView = (RecyclerView) findView
    recyclerView.setHasFixedSize(true);
    recyclerView.setLayoutManager(new Line
    mDatabase = openOrCreateDatabase(MainA

    showEmployeesFromDatabase();
}
```

When Delete Data:


Custom Phone (768x1280, 320dpi) - 192.168.238.102 - ...
3:54
Sql_Lite with Recycler View And List...
asj
as
asd
2132
Mujahid
Are you sure?
CANCEL
YES
Free for personal use
Android Assignment 7.docx - Microsoft Word (Product Activation Failed)
Edit Form
UPDATE
n edit your data
n Delete Data:

2) Google Map in Android Studio:

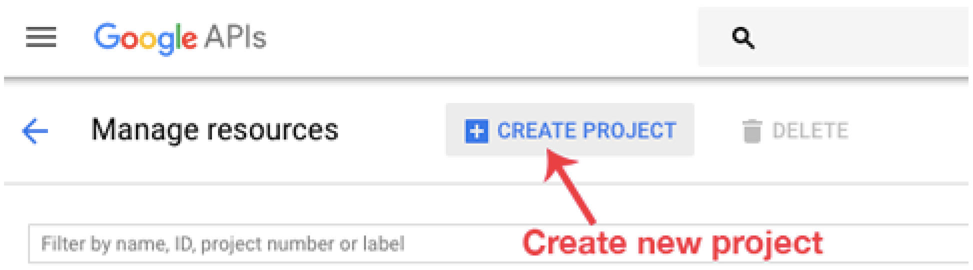
Android allows us to integrate Google Maps in our application. For this Google provides us a library via Google Play Services for using maps. In order to use the Google Maps API, you must register your application on the Google Developer Console and enable the API.

Steps For Getting The Google Maps Api Key:

An API key is needed to access the Google Maps servers. This key is free and you can use it with any of your applications. If you haven't created project, you can follow the below steps to get started:

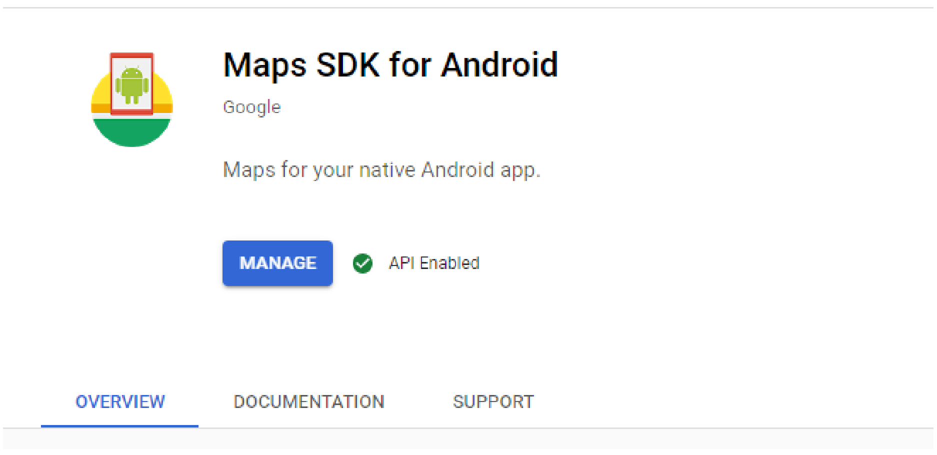
Step 1: Open Google developer console and signin with your gmail account:
https://console.developers.google.com/project

Step 2: Now create new project. You can create new project by clicking on the Create Project button and give name to your project.



Step 3: Now click on APIs & Services and open Dashboard from it.

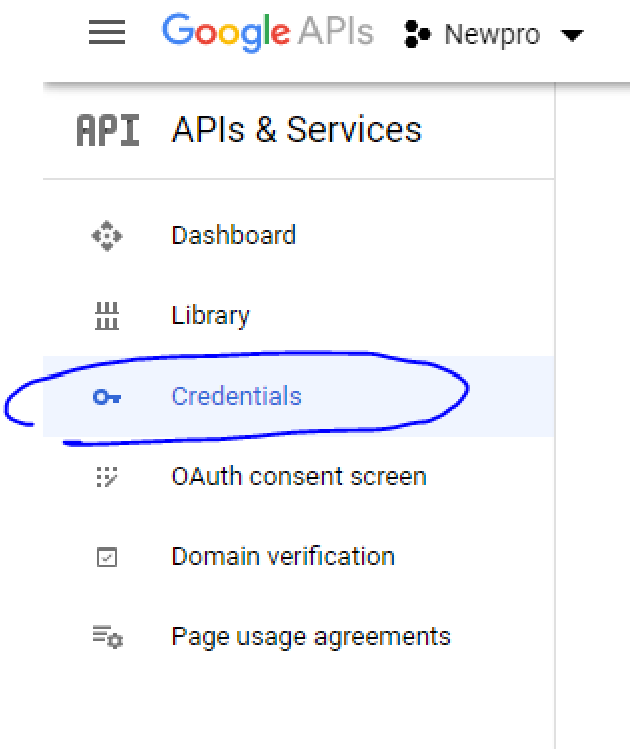
Step 4: In this open Enable APIS AND SERICES.

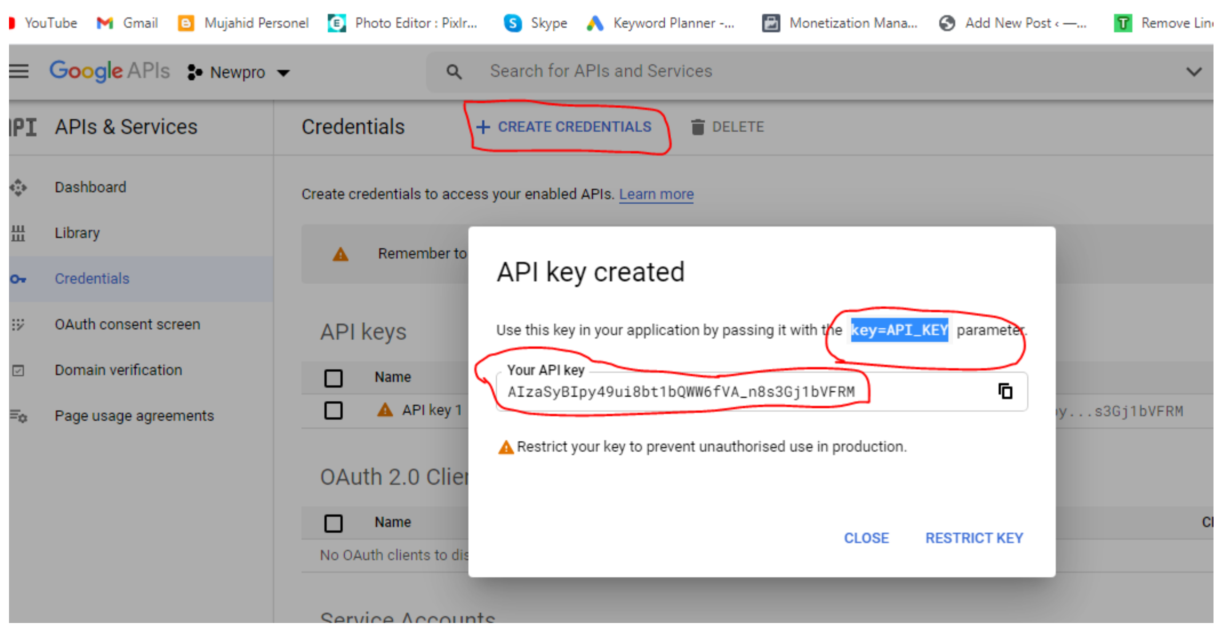


Step 5: Now open Google Map Android API.

Step 6: Now enable the Google Maps Android API.

Step 6: Now go to Credentials





Create credentials to access your enabled APIs. Learn more

API key created

Use this key in your application by passing it with the key=API_KEY parameter.

Your API key
AIzaSyBIpy49ui8bt1bQWW6fVA_n8s3Gj1bVFRM

Restrict your key to prevent unauthorised use in production.

Step 8: Now API your API key will be generated. Copy it and save it somewhere as we will need it when implementing Google Map in our Android project.

API Key: AIzaSyBIpy49ui8bt1bQWW6fVA_n8s3Gj1bVFRM

Google Maps Example To Access User Current Location In Android Studio:

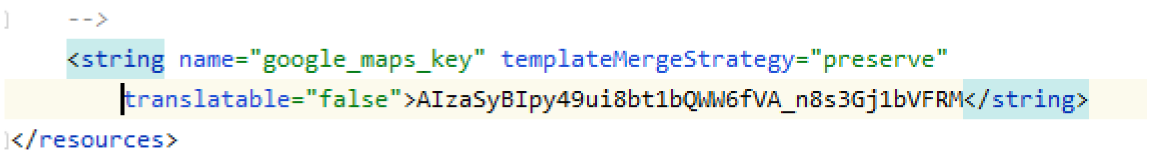
In the below Google Map example we will show user current location in Map. We also example different map types, methods and lots more details required while implementing Map in Android.

Below you can download code, see final output and step by step explanation of example:

Step 1: Create a New Android Project and name it GoogleMaps.

Step 2: Now select Google Maps Activity and then click Next and finish.

Step 3: Now open google_maps_api.xml (debug) in values folder



```
    -->
    <string name="google_maps_key" templateMergeStrategy="preserve"
        translatable="false">AIzaSyBIpy49ui8bt1bQWW6fVA_n8s3Gj1bVFRM</string>
</resources>
```

Step 5: Now open build.gradle and add `implementation 'com.google.android.gms:play-services:8.4.0'`
in dependencies **build.gradle** code

Step 6: Now open activity_maps.xml and add a fragment code in it

Here add a fragment element to the activity's layout file to define a Fragment object. In this element, set the android:name attribute to "com.google.android.gms.maps.MapFragment". This automatically attaches a MapFragment to the activity. The following layout file contains a fragment element:

activity_maps.xml code

```
<fragment android:id="@+id/map"
android:name="com.google.android.gms.maps.SupportMapFragment"
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:map="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MapsActivity"/>
```

Step 6: Now define internet and location permissions in Android Manifest

INTERNET – To determine if we are connected to Internet or not.

ACCESS_FINE_LOCATION – To determine user's location using GPS. It will give us precise location.

AndroidManifest.xml code:

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

Step 7: Now we will code MapsActivity.java file for inserting callbacks in Google Maps:

- -OnMapReadyCallback: This callback is called when the map is ready to be used

```
@Override
public void onMapReady(GoogleMap googleMap) {}
```

- -GoogleApiClient.ConnectionCallbacks: This callback is called whenever device is connected and disconnected and implement onConnected() and onConnectionSuspended() functions.
- -GoogleApiClient.OnConnectionFailedListener: Provides callbacks for scenarios that result in a failed attempt to connect the client to the service. Whenever connection is failed onConnectionFailed() will be called.

```
@Override
public void onConnectionFailed(ConnectionResult connectionResult) {
}
```

- -LocationListener: This callback have function onLocationChanged() that will be called whenever there is change in location of device.

```
@Override

public void onLocationChanged(Location location) {}
```

- -onMapReady(): This function is called when the map is ready to be used.
- -buildGoogleApiClient(): This method is used to initialize the Google Play Services.
- -addConnectionCallbacks(): You need to call registers a listener to receive connection events from this GoogleApiClient.
- -addOnConnectionFailedListener(): This methods adds a listener to register to receive connection failed events from this GoogleApiClient.
- -GoogleApiClient.Builder: Builder is used to help construct the GoogleApiClient object and addApi () specify which Apis are requested by your app.
- -mGoogleApiClient.connect(): A client must be connected before executing any operation.
- -Zoom Controls: The Maps API provides built-in zoom controls that appear in the bottom right hand corner of the map. These can be enabled by calling:
- -Zoom Controls: The Maps API provides built-in zoom controls that appear in the bottom right hand corner of the map. These can be enabled by calling:
- mMap.getUiSettings().setZoomControlsEnabled(true);
- -Zoom Gestures:
- ZoomIn: Double tap to increase the zoom level by 1.
- Zoom Out: Two finger tap to decrease the zoom level by 1.
- mMap.getUiSettings().setZoomGesturesEnabled(true);
- -Compass: You can set compass by calling below method:
- mMap.getUiSettings().setCompassEnabled(true);
- -Changing the Map Type:
- The Android Maps API provides normal, satellite, terrain and hybrid map types to help you out:
- mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
- mMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
- mMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
- mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
- MAP_TYPE_NORMAL : Represents a typical road map with street names and labels.

- MAP_TYPE_SATELLITE: Represents a Satellite View Area without street names and labels.MAP_TYPE_TERRAIN: Topographic data. The map includes colors, contour lines and labels, and perspective shading. Some roads and labels are also visible.
- MAP_TYPE_HYBRID : Combines a satellite View Area and Normal mode displaying satellite images of an area with all labels.
- Map_TYPE_NONE : No tiles. It is similar to a normal map, but doesn't display any labels or coloration for the type of environment in an area.
- Add the following inside setUpMap() just below the setMyLocationEnabled() call:
- The location of the user is updated at the regular intervals. We have used FusedLocationProvider. We have used requestLocationUpdates() method to get regular updates about a device's location. Do this in the onConnected() callback provided by Google API Client, which is called when the client is ready.
- LocationRequest mLocationRequest is used to get quality of service for location updates from the FusedLocationProviderApi using requestLocationUpdates.

All Code Of MainActivity

```
public class MainActivity  extends FragmentActivity implements
OnMapReadyCallback,
        LocationListener,GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener {

    private GoogleMap mMap;
    Location mLastLocation;
    Marker mCurrLocationMarker;
    GoogleApiClient mGoogleApiClient;
    LocationRequest mLocationRequest;

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




        // Obtain the SupportMapFragment and get notified when the map is
ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);



    }
```

```java
    @Override
    public void onConnected(@Nullable Bundle bundle) {

        mLocationRequest = new LocationRequest();
        mLocationRequest.setInterval(1000);
        mLocationRequest.setFastestInterval(1000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURA
CY);
        if (ContextCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_FINE_LOCATION)
                == PackageManager.PERMISSION_GRANTED) {

LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest,this);
        }

    }

    @Override
    public void onConnectionSuspended(int i) {

    }

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult
connectionResult) {

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {

        mMap = googleMap;

        if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
                    Manifest.permission.ACCESS_FINE_LOCATION)
                    == PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                mMap.setMyLocationEnabled(true);
            }
        }
        else {
            buildGoogleApiClient();
            mMap.setMyLocationEnabled(true);
        }


        EditText locationSearch = (EditText) findViewById(R.id.editText);
        String location = locationSearch.getText().toString();
        List<Address> addressList = null;

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

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

            Address address = addressList.get(0);
            LatLng latLng = new LatLng(address.getLatitude(),
address.getLongitude());
            mMap.addMarker(new
MarkerOptions().position(latLng).title(location));

            //move map camera
            mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
            mMap.animateCamera(CameraUpdateFactory.zoomTo(6));




        }

    }
    protected synchronized void buildGoogleApiClient() {
        mGoogleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API).build();
        mGoogleApiClient.connect();
    }



    @Override
    public void onLocationChanged(@NonNull Location location) {

        mLastLocation = location;
        if (mCurrLocationMarker != null) {
            mCurrLocationMarker.remove();
        }
//        Place current location marker
        LatLng latLng = new LatLng(location.getLatitude(),
location.getLongitude());
        MarkerOptions markerOptions = new MarkerOptions();
        markerOptions.position(latLng);
        markerOptions.title("Current Position");

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFa
ctory.HUE_GREEN));
        mCurrLocationMarker = mMap.addMarker(markerOptions);

//        move map camera
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.animateCamera(CameraUpdateFactory.zoomTo(4));

//        stop location updates
        if (mGoogleApiClient != null) {

LocationServices.FusedLocationApi.removeLocationUpdates(mGoogleApiClient,
this);
        }

    }
    public void searchLocation(View view) {


        EditText locationSearch = (EditText) findViewById(R.id.editText);
        String location = locationSearch.getText().toString();
        List<Address> addressList = null;

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

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

        }

    }
}



//On Connected Method-----------------------------------------------
@Override
public void onConnected(Bundle bundle) {
    mLocationRequest = new LocationRequest();
    mLocationRequest.setInterval(1000);
    mLocationRequest.setFastestInterval(1000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURA
CY);
    if (ContextCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION)
            == PackageManager.PERMISSION_GRANTED) {

LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
                mLocationRequest, this);
    }
}
```

Whenever user’s location is changed. For that Google has predefined function onLocationChanged that will be called as soon as user’s location change. Here we are getting the coordinates of current location using getLatitude() and getLongitude() and we are also adding Marker.