**Lab 1**

**Create an App to navigate from one Activity to another using an intent object and passing data**

Step 1: Create a New Project and Select the Empty Activity.

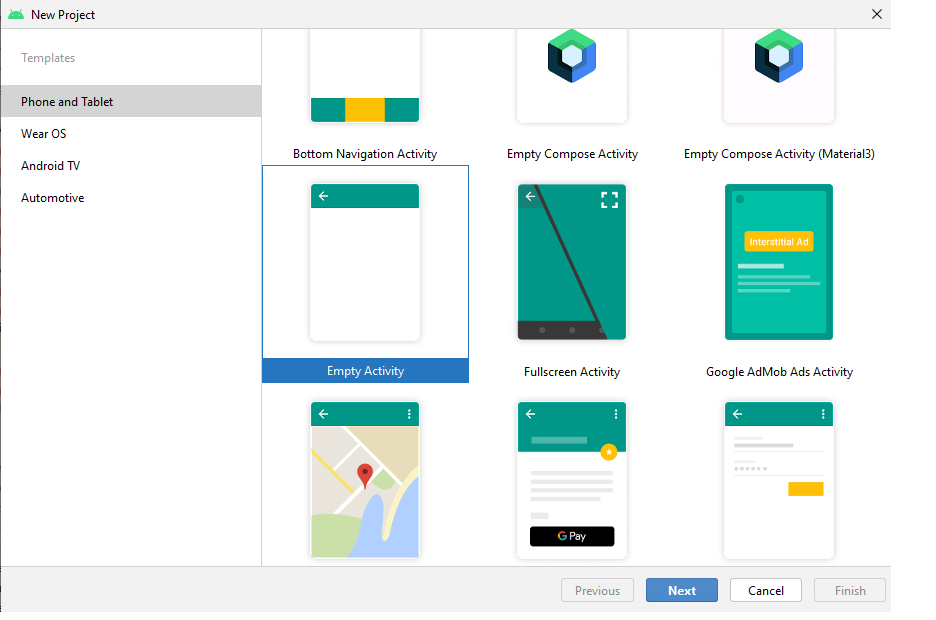


Figure 1 Selecting the Empty Activity

Step2: Name the Project and Select the Language as Java and click on finish.

Note: Make sure you have an active internet connection.

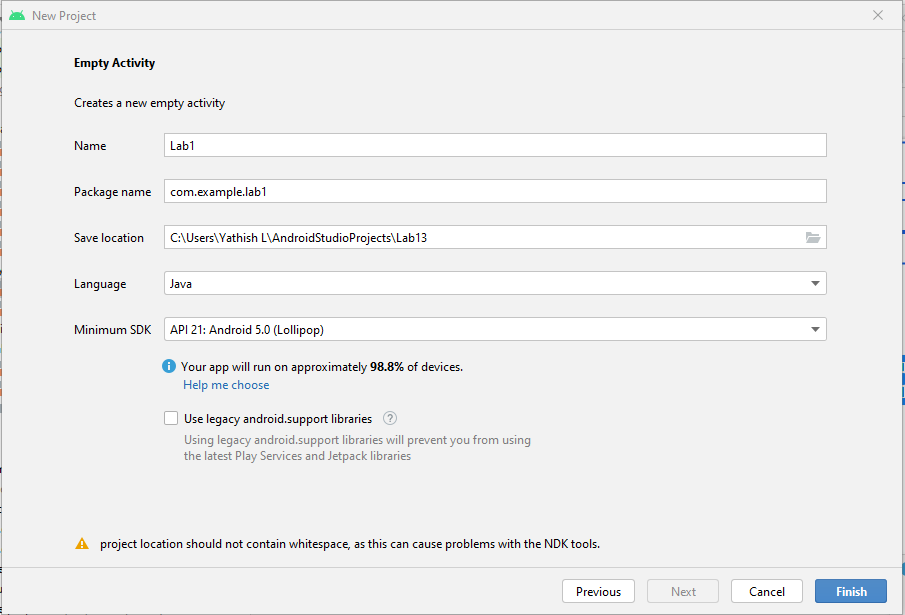
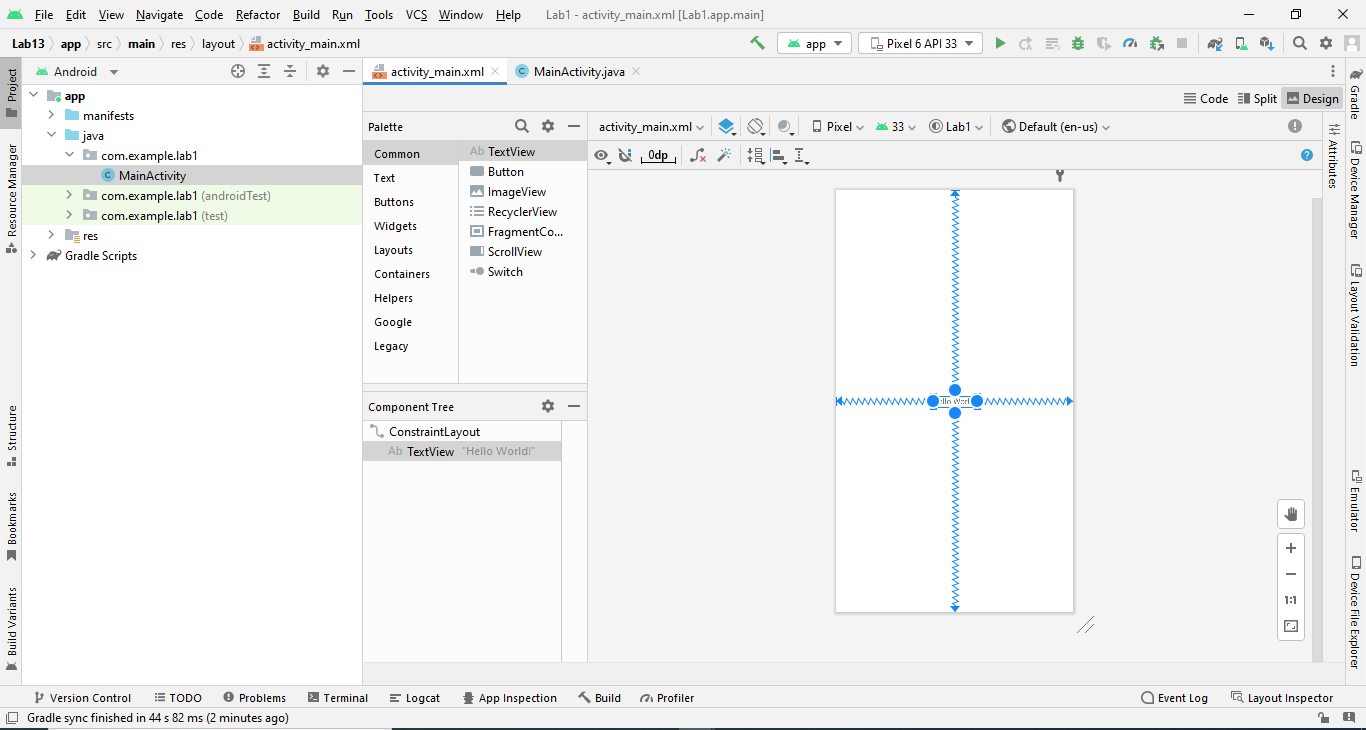


Figure 2 Name the project and select the Programming language.

Note: Please Wait till gradle build is completed before coding or designing.



After the Gradle Build is Completed.



Step3: Add the EditText and Button Components to Activity\_main.xml

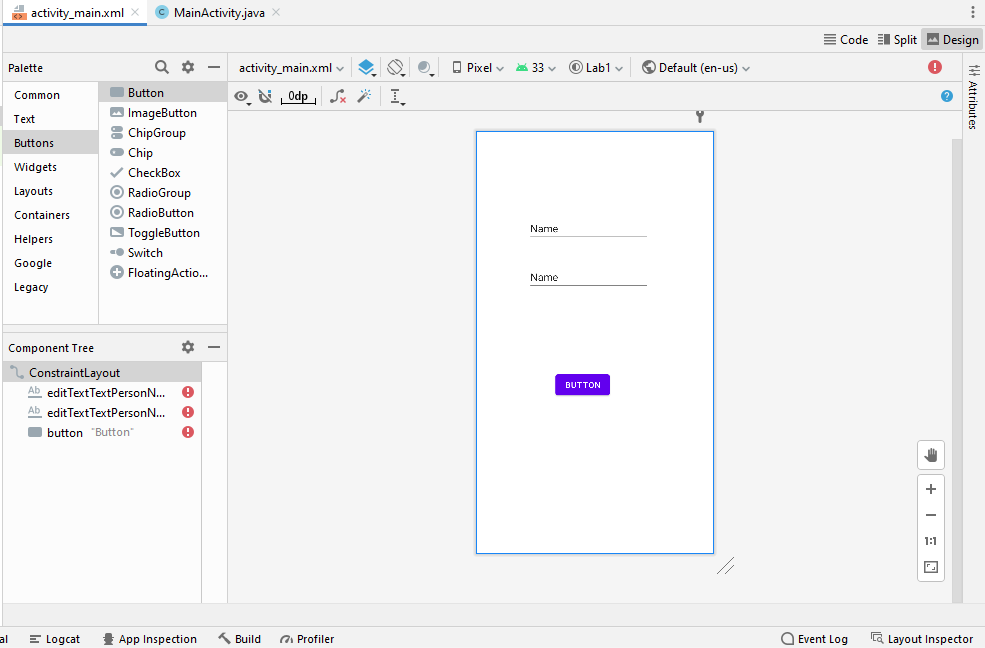
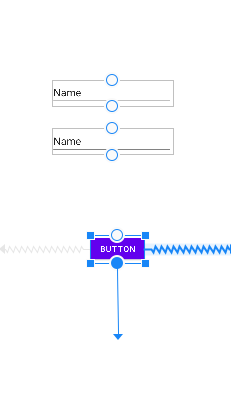
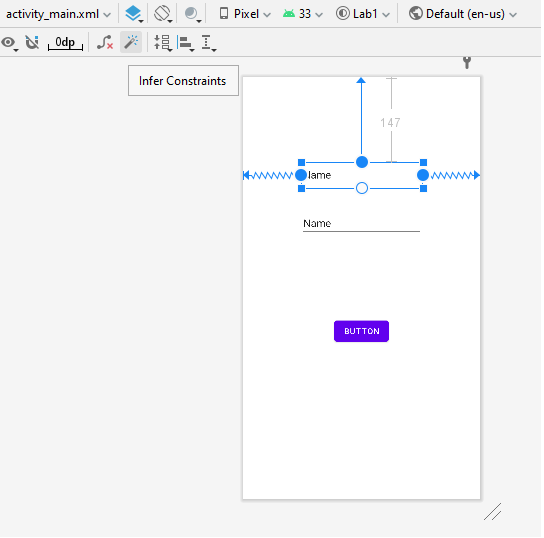
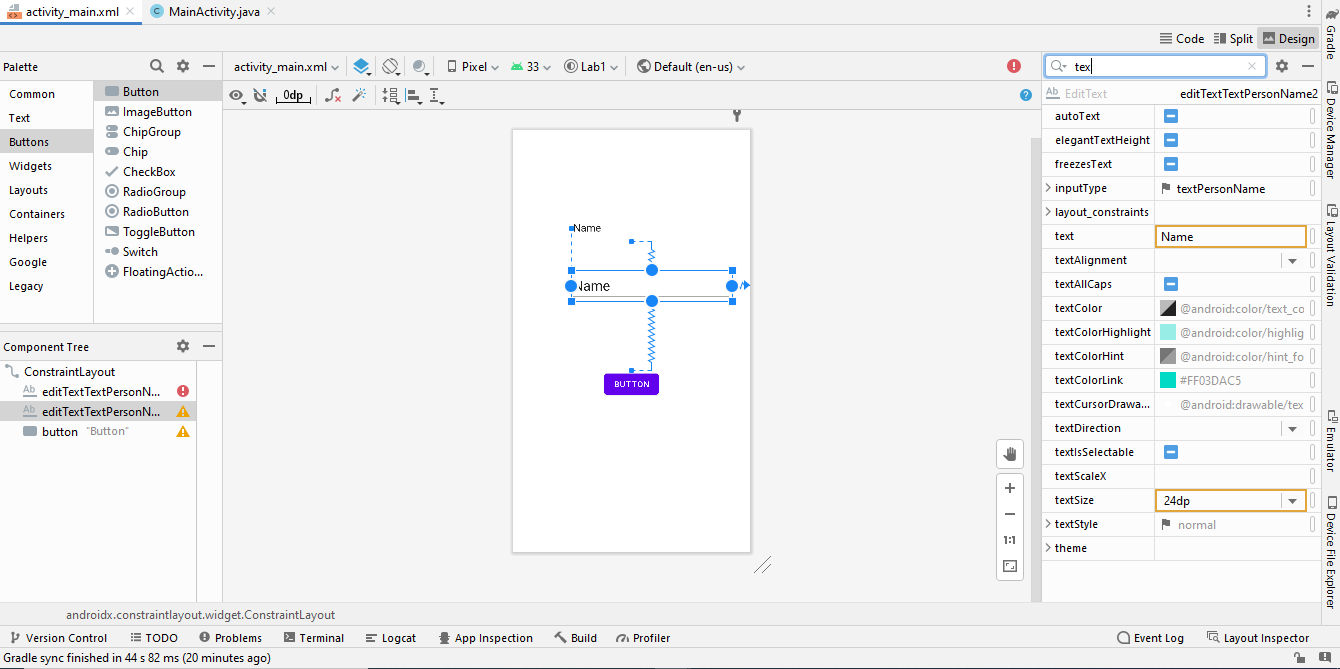


Figure There is No constraint assigned to Edit text or Button, you can do this is in two Ways one using the mapping the Constraint manually or using Infer Constraint

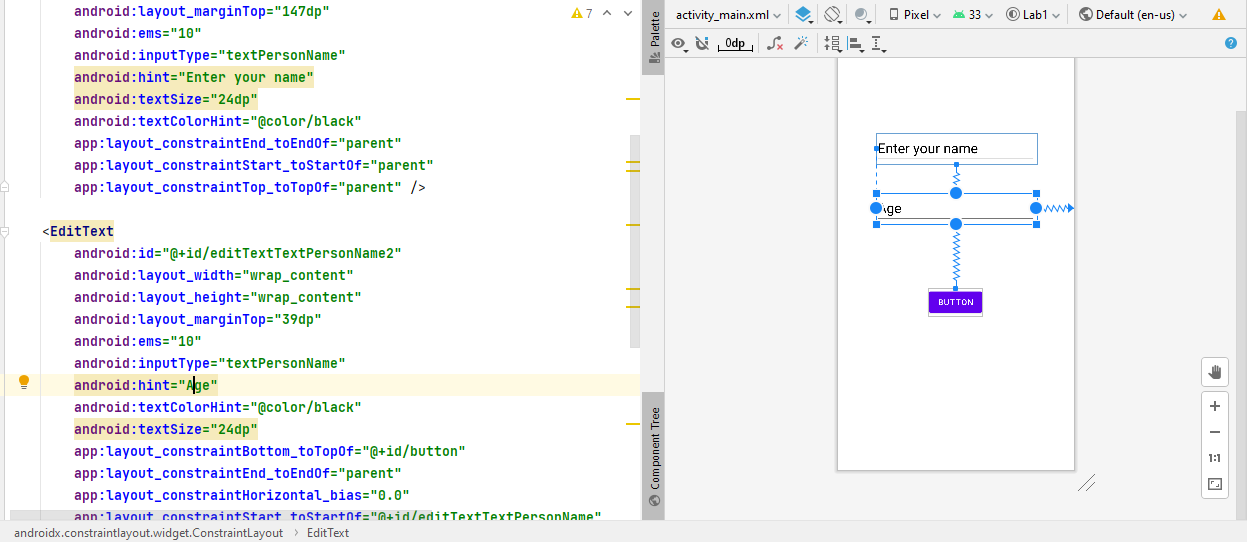


Manual Assigning Infer Constraint

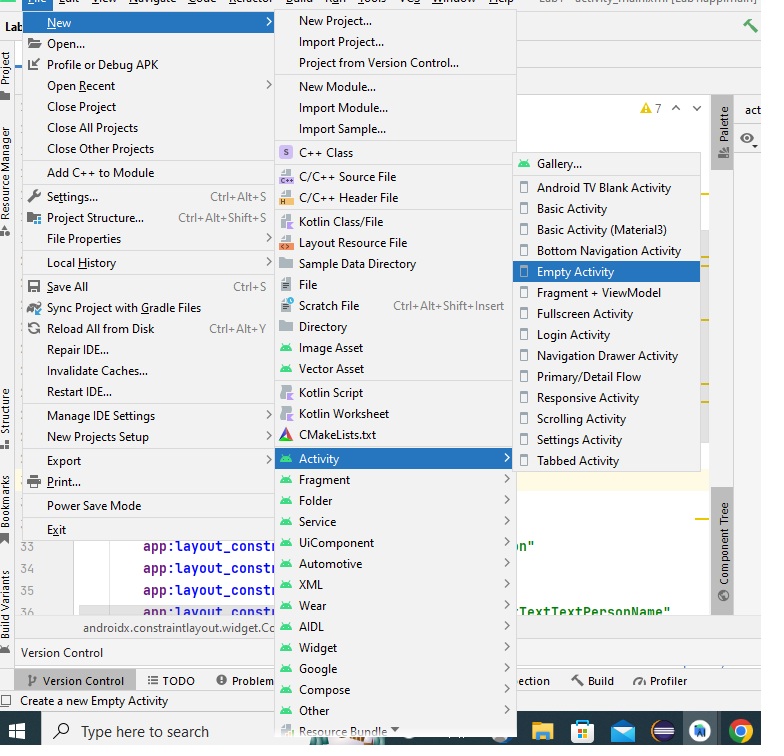
To change the Attributes of a component you can use the Attribute panel or in xml code and use can search the Attributes in the Attributes panel



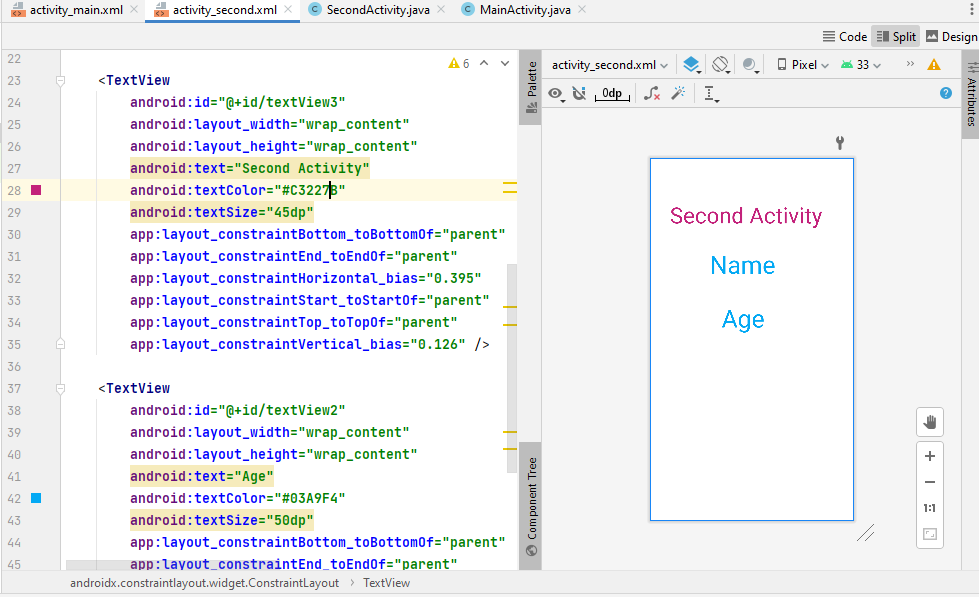
You can also do Change Attributes in the xml File.



Step4: create Another Activity as Second Activity



Please Follow the Above Step and create the Second Activity how you did for Main Activity.



Then You can Start Coding

**XML code of Main Activity**

<**EditText  
 android:id="@+id/editTextTextPersonName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="147dp"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:hint="Enter your name"  
 android:textSize="24dp"  
 android:textColorHint="@color/black"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />  
  
<**EditText  
 android:id="@+id/editTextTextPersonName2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="39dp"  
 android:ems="10"  
 android:inputType="textPersonName"  
 android:hint="Age"  
 android:textColorHint="@color/black"  
 android:textSize="24dp"  
 app:layout\_constraintBottom\_toTopOf="@+id/button"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="@+id/editTextTextPersonName"  
 app:layout\_constraintTop\_toBottomOf="@+id/editTextTextPersonName"  
 app:layout\_constraintVertical\_bias="0.092"** />  
  
<**Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="267dp"  
 android:text="Click"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"** />

**Xml Code for Second Activity**

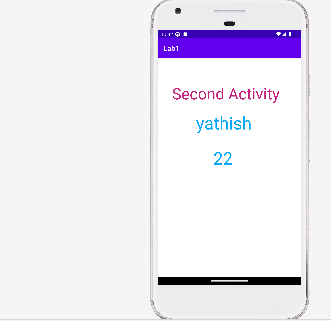
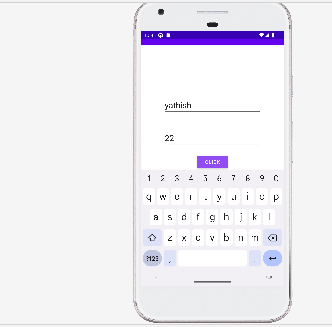
<**TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Name"  
 android:textSize="50dp"  
 android:textColor="#03A9F4"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.436"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.272"** />  
  
<**TextView  
 android:id="@+id/textView3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Second Activity"  
 android:textColor="#C3227B"  
 android:textSize="45dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.395"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.126"** />  
  
<**TextView  
 android:id="@+id/textView2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Age"  
 android:textColor="#03A9F4"  
 android:textSize="50dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.447"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView"  
 app:layout\_constraintVertical\_bias="0.1"** />

**The Java Code of the Main Activity**

@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 Button b = findViewById(R.id.***button***);  
 EditText name=findViewById(R.id.***editTextTextPersonName***);  
 EditText age = findViewById(R.id.***editTextTextPersonName2***);  
 *// it will give you the reference to the view in XML layouts by searching its ID* b.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 */\*These methods will be called by the Android framework when the View to  
 which the listener has been registered  
 is triggered by user interaction with the item in the UI\*/* **public void** onClick(View view) {  
 String Dname=name.getText().toString();  
 String Dage =age.getText().toString();  
 */\* By using the gettext we can get the user input from the activity\*/* Intent i = **new** Intent(MainActivity.**this**,SecondActivity.**class**);  
 *//Intent is the message that is passed between components such as activities,  
 // content providers, broadcast receivers, services etc* i.putExtra(**"name"**,Dname);  
 i.putExtra(**"age"**,Dage);  
 *//Add extended data to the intent.* startActivity(i);  
 *//The startActivity(Intent) method is used to start a new activity,  
 // which will be placed at the top of the activity stack.* }  
 });  
  
}

**The Java Code of the Second Activity**

@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_second***);  
 Intent i = getIntent();  
 *//Return the intent that started this activity* String name=i.getStringExtra(**"name"**);  
 String age = i.getStringExtra(**"age"**);  
 *//this is used to get value which is passed in the Activity* TextView t1=findViewById(R.id.***textView***);  
 TextView t2 = findViewById(R.id.***textView2***);  
 t1.setText(name);  
 t2.setText(age);  
 *//this is used to send the data to the xml layout through java code*}

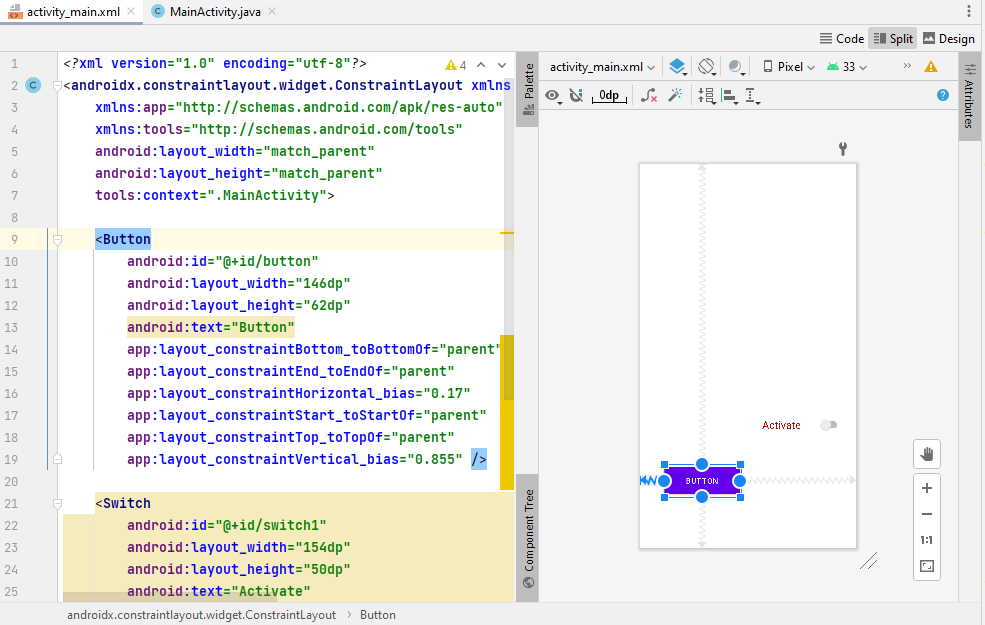
****

**Lab 2**

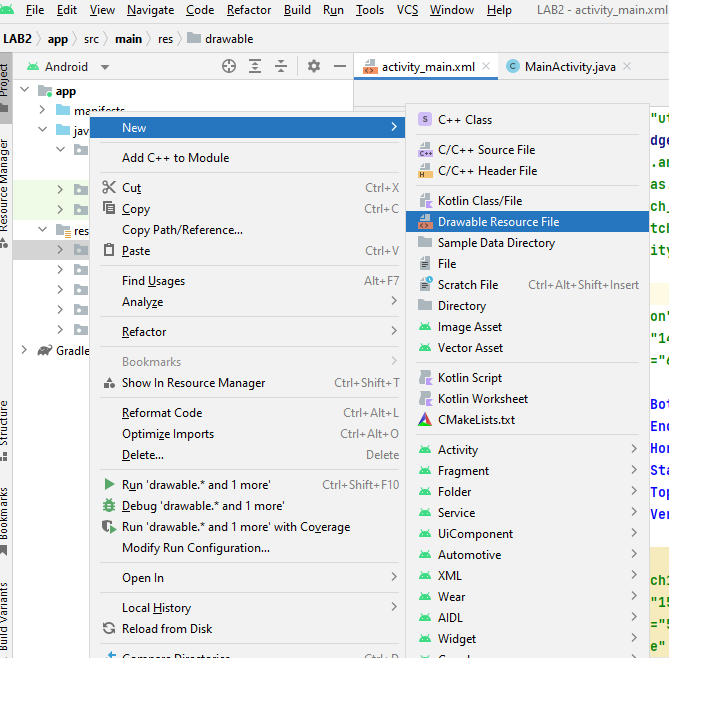
**Devise a mobile app to showcase graphics on button states and add a widget**

**Follow the Step1 of Lab1 Program**

Step 2: Add the button, switch and constraint layout in activity\_main.xml

****

Step 3: Create A drawable file in **res->drawable** folder



Write the code in the Drawable file(shape.xml)

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android=”**[**http://schemas.android.com/apk/res/android**](http://schemas.android.com/apk/res/android)**” android:shape=”rectangle”**>  
 <**gradient android:startColor="#123456" android:type="linear" android:endColor="#456875"**></**gradient**>  
 <**corners android:radius="18dp"**></**corners**>  
</**shape**>

//This code is for design the graphics of the button

//Please change in the theme if Drawable is not working

Repeat the step3 and create a new Drawable file

Write the code in the Drawable file (buttong)

*<?***xml version="1.0" encoding="utf-8"***?>*<**selector xmlns:android="http://schemas.android.com/apk/res/android"**>  
<**item android:state\_pressed="true" android:drawable="@drawable/shape"**></**item**>  
 <**itemandroid:state\_pressed="false"android:drawable="@color/black"/**>  
</**selector**>

//This for the button graphic changed when button is pressed

Xml code

<**Button  
 android:id="@+id/button"  
 android:layout\_width="146dp"  
 android:layout\_height="62dp"  
 android:text="Press"  
 android:background="@drawable/buttong"  
 android:textColor="@color/white"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.162"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.863"** />  
  
<**Switch  
 android:id="@+id/switch1"  
 android:layout\_width="154dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="204dp"  
 android:text="Activate"  
 android:textColor="#AE0C0C"  
 android:textSize="20dp"  
  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.904"  
 app:layout\_constraintStart\_toStartOf="parent"  
  
 app:layout\_constraintVertical\_bias="0.903"** />  
  
<**androidx.constraintlayout.widget.ConstraintLayout  
 android:id="@+id/Layout"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="1dp"  
 android:layout\_marginTop="1dp"  
 android:layout\_marginEnd="1dp"  
 android:layout\_marginBottom="1dp"  
 app:layout\_constraintBottom\_toTopOf="@+id/switch1"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"**>  
  
</**androidx.constraintlayout.widget.ConstraintLayout**>

Java Code

@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 Button button = findViewById(R.id.***button***);  
 Switch switch1=findViewById(R.id.***switch1***);

*// it will give you the reference to the view in XML layouts by searching its ID*  
 ConstraintLayout layout = findViewById(R.id.***Layout***);  
 CalendarView cal = **new** CalendarView(**this**);

// create the calenderview dynamically   
 layout.addView(cal);

//adding the calenderView dyanamically to the constraintLayout  
 switch1.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if**(switch1.isChecked())  
 { //if the switch is checked the calender view is removed

// and button is enabled

button.setEnabled(**true**);  
 layout.removeView(cal);

Toast.*makeText*(MainActivity.**this**,**"Button is Enabled"**,Toast.***LENGTH\_LONG***.show();

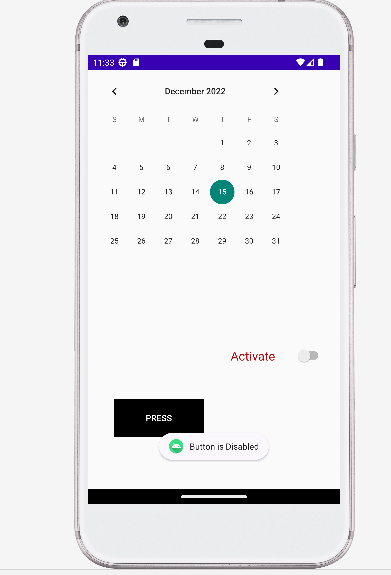
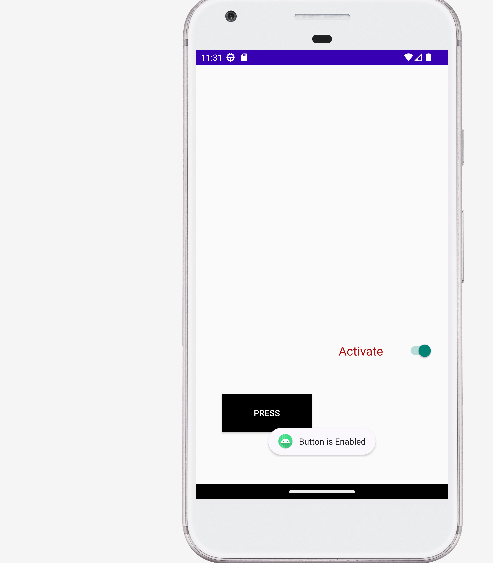
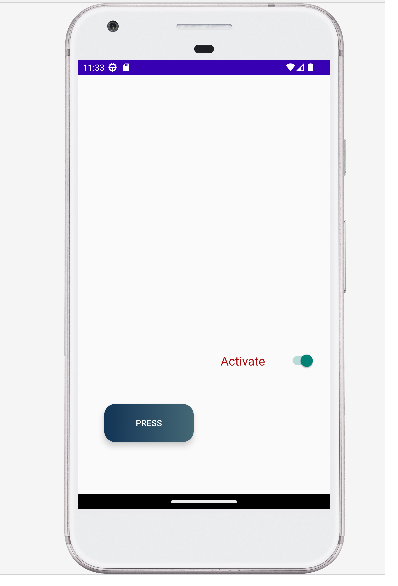
}  
 **else** {//if the switch is unchecked the calender view is removed

// and button is disabled

Toast.*makeText*(MainActivity.**this**,**"Button is Disabled"**,Toast.***LENGTH\_LONG***).show();

button.setEnabled(**false**);  
 layout.addView(cal);  
 }  
 }  
 }  
  
}

Output



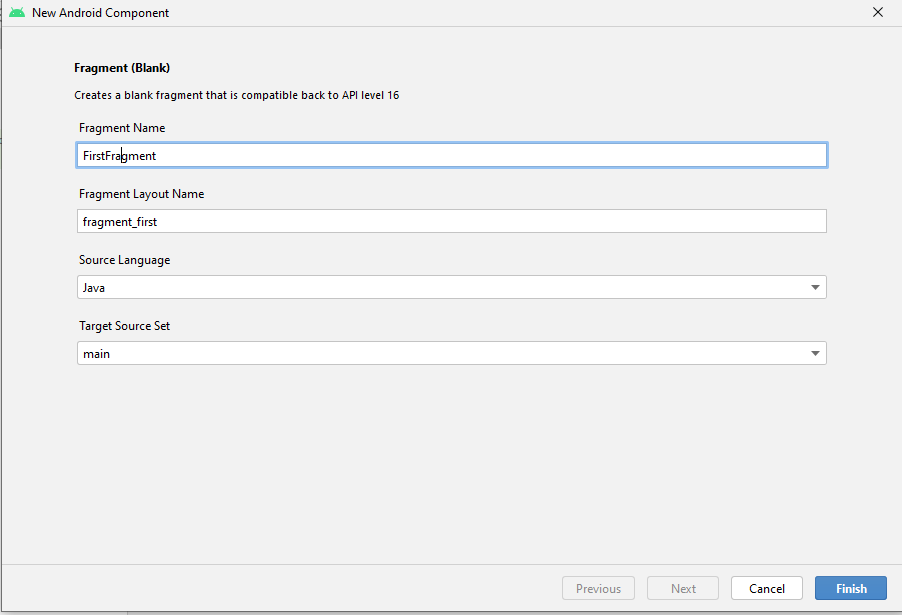
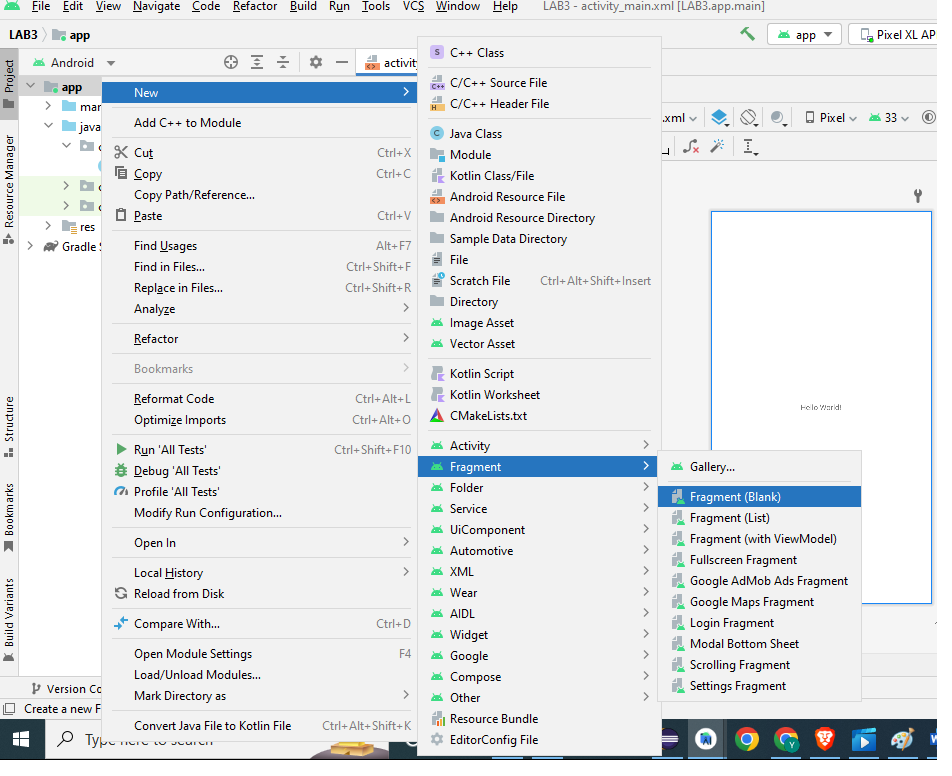
Lab 3

Demonstrate adding and removing fragments at run time

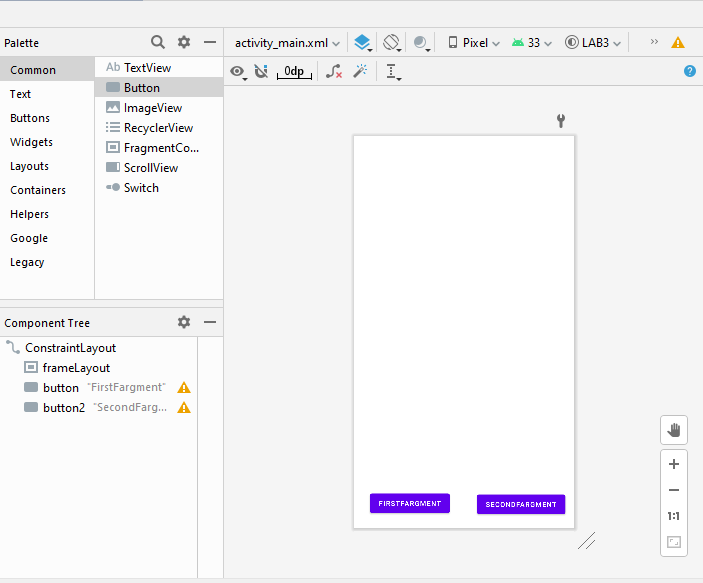
**Follow the Step1 of Lab1 Program**

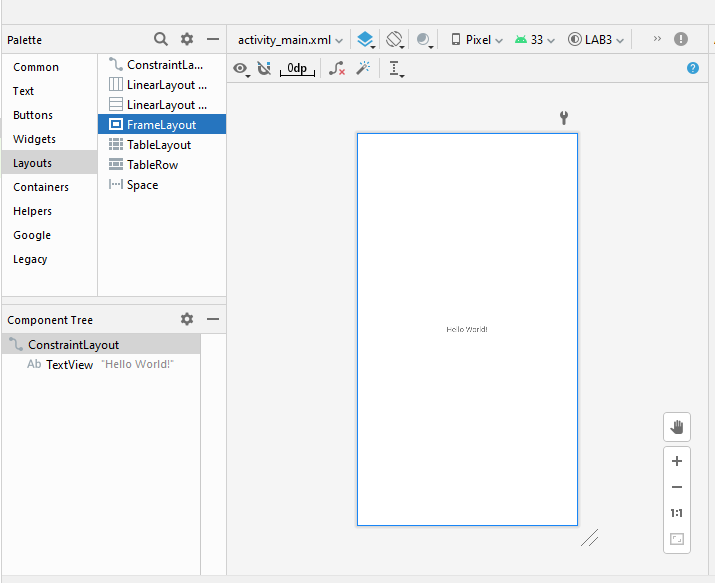
Step2 Create the two Fragment in the project.

Note: please give valid name to fragment



\*please repeat the step once Again Create the second Fragment(blank)

Step4: Add the Frame Layout and two buttons from the Palette to the main activity



Xml Code for the activity\_main.xml

<**FrameLayout  
 android:id="@+id/frameLayout"  
 android:layout\_width="411dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="1dp"  
 android:layout\_marginEnd="1dp"  
 android:layout\_marginBottom="56dp"  
 app:layout\_constraintBottom\_toTopOf="@+id/button"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0"**>  
  
</**FrameLayout**>  
  
<**Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="22dp"  
 android:text="FirstFargment"  
 android:textSize="13dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.114"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/frameLayout"** />  
  
<**Button  
 android:id="@+id/button2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginEnd="10dp"  
 android:layout\_marginBottom="20dp"  
 android:text="SecondFargment"  
 android:textSize="13dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.89"  
 app:layout\_constraintStart\_toEndOf="@+id/button"** />

Fragment one xml code

<**TextView  
 android:id="@+id/textView2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="FirstFragment"  
 android:textSize="48dp"  
 android:textColor="#FFFFFF"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />

Note Please change the Background of Fragment

**android:background="#83E111"**

Fragment Second xml code

<**TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="SecondFragment"  
 android:textSize="48dp"  
 android:textColor="#FFFFFF"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />

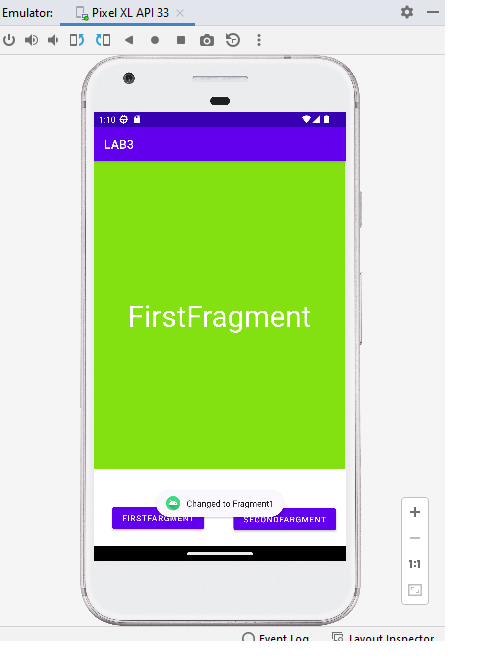
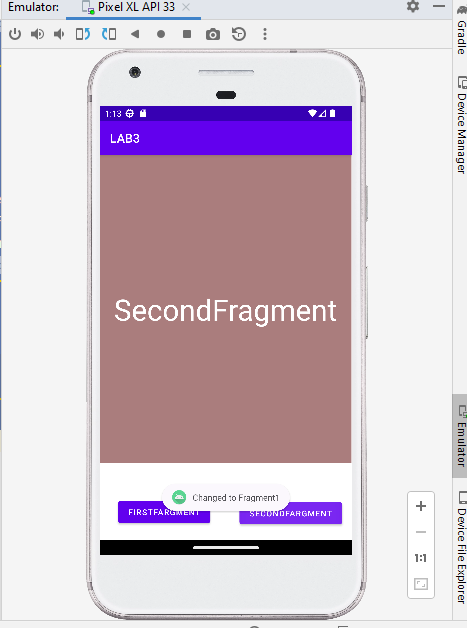
Note Please change the Background of Fragment

**android:background="#AA7D7D"**

Java Code for Main Activity

@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 FirstFragment fragment1=**new** FirstFragment();  
 secondFragment fragment2=**new** secondFragment();  
 FragmentManager managerF = getSupportFragmentManager();  
 *// FragmentManager is the class responsible for performing actions  
 // on your app's fragments,  
 // such as adding, removing, or replacing them* FragmentTransaction Ftransaction = managerF.beginTransaction();  
 *//a FragmentManager can add, remove, replace, and perform other actions with fragments  
 // in response to user interaction. Each set of fragment changes that you  
 // commit is called a transaction* Ftransaction.add(R.id.***frameLayout***,fragment1);  
 *//add the Fragment* Button b1 = findViewById(R.id.***button***);  
 Button b2 = findViewById(R.id.***button2***);  
 b1.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 FragmentManager managerF = getSupportFragmentManager();  
 FragmentTransaction Ftransaction = managerF.beginTransaction();  
 Ftransaction.replace(R.id.***frameLayout***,fragment1);  
 *//replace the Fragment* Toast.*makeText*(MainActivity.**this**,**"Changed to Fragment1"**,Toast.***LENGTH\_LONG***).show();  
 Ftransaction.commit();  
 *//Commit the changes in the Activity Main.* }  
 });  
 b2.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 FragmentManager managerF = getSupportFragmentManager();  
 FragmentTransaction Ftransaction = managerF.beginTransaction();  
 Ftransaction.replace(R.id.***frameLayout***,fragment2);  
 Toast.*makeText*(MainActivity.**this**,**"Changed to Fragment1"**,Toast.***LENGTH\_LONG***).show();  
 Ftransaction.commit();  
 }  
 });  
  
  
  
  
}

Output

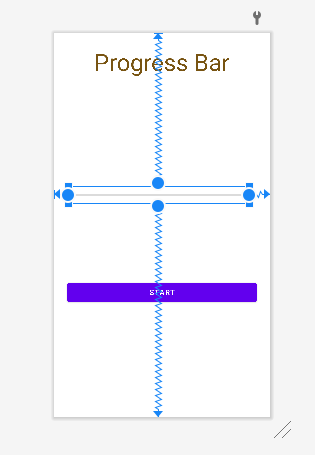


**Lab 4**

**Develop an app to display a progress Bar and show a message with alert dialog.**

Step1 : Create a new Project and Wait for the Gradle build.

Step2 : Add the Progress Bar and Button from the Palette



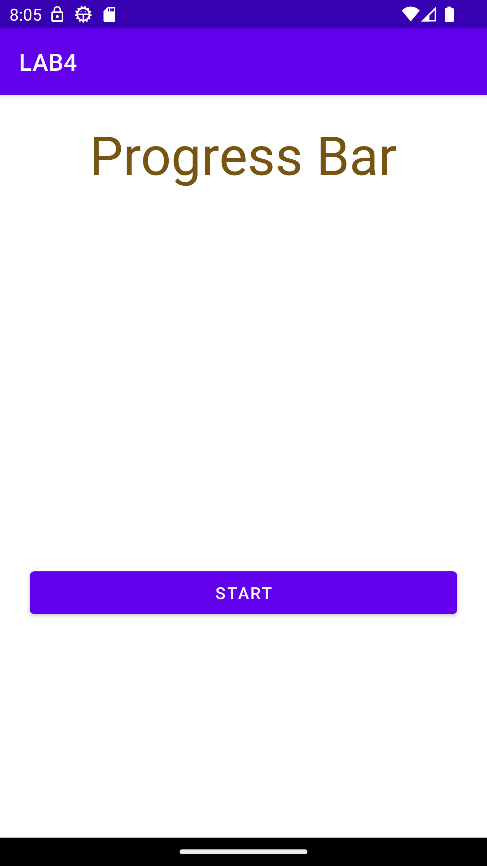
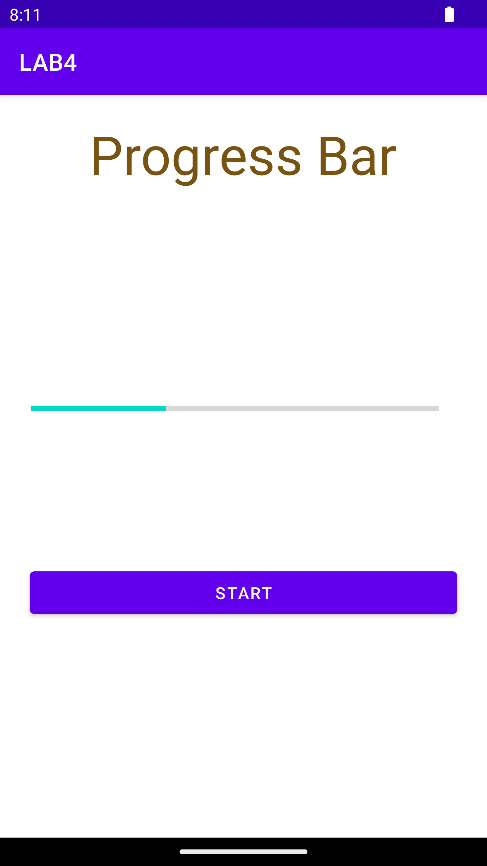
XML Code of Activity\_main.xml

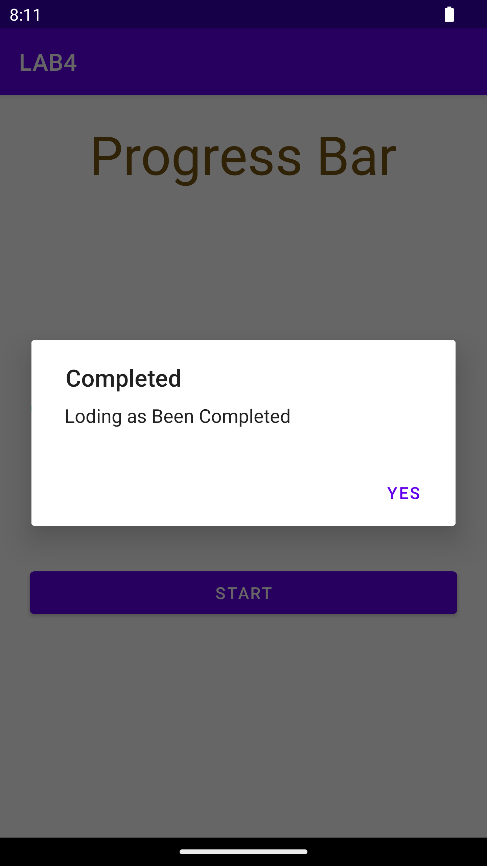
<**Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_margin="25dp"  
 android:layout\_height="wrap\_content"  
 android:text="Start"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.498"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.702"** />  
  
<**ProgressBar  
 android:id="@+id/progressBar2"  
 style="?android:attr/progressBarStyleHorizontal"  
 android:layout\_width="345dp"  
 android:layout\_height="32dp"  
  
  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.392"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.418"** />  
  
<**TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Progress Bar"  
 android:textSize="45dp"  
 android:textAlignment="center"  
 android:textColor="#785412"  
 app:layout\_constraintBottom\_toTopOf="@+id/progressBar2"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.498"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.106"** />

Java Code for MainActivity

**public** Button **button**;  
 ProgressBar **Progress**;  
 CountDownTimer **t**;  
 **float total**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **button**=findViewById(R.id.***button***);  
 **Progress**=findViewById(R.id.***progressBar2***);  
  
 **Progress**.setVisibility(View.***INVISIBLE***);  
 **Progress**.setMax(100);  
 **button**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **Progress**.setVisibility(View.***VISIBLE***);  
 **long** min= 10\*1000;*//second \* milisecond;* **new** CountDownTimer(min,1000)  
 {  
 @Override  
 **public void** onTick(**long** l) {  
 **long** a = min-l;  
 **total**= ((**float**)a/(**float**)min)\*100;  
 Log.*e*(**"Total"**, String.*valueOf*( (a/min)));  
 **Progress**.setProgress((**int**) **total**);  
 }  
  
 @Override  
 **public void** onFinish() {  
 AlertDialog.Builder d = **new** AlertDialog.Builder(MainActivity.**this**);  
 d.setTitle(**" Completed "**).setMessage(**" Loding as Been Completed"**).setPositiveButton(**"Yes"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialogInterface, **int** i) {  
 Toast.*makeText*(MainActivity.**this**,**"DOne"**,Toast.***LENGTH\_LONG***).show();  
 }  
 });  
 AlertDialog dialog = d.create();  
 dialog.show();  
 }  
 }.start();  
*//* }  
 });  
 }

Output



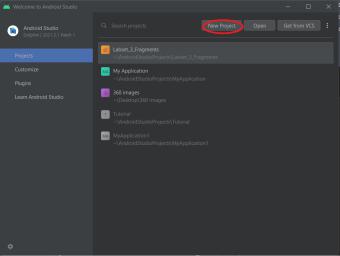


**LAB 5**

**Implements an application that will create a databases with table of user credentials and create a login portal system.**

**Step 1:**

create a new project



* + Under “Phone and Tablet” Templates select “Empty Activity”
  + Project name: ‘Android\_SQLiteHelper’
  + Language: ‘java’
  + Minimum APK: ‘API 21: Android 5.1 (Lollipop)’

**Step 2:**

Paste this xml code in ‘app/src/main/res/layout/activity\_main.xml’ to change the design

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

<androidx.constraintlayout.widget.ConstraintLayout 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"

tools:context=".MainActivity">

<Button

android:id="@+id/button"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginBottom="281dp"

android:text="Login"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent" />

<EditText

android:id="@+id/editTextTextPersonName"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="153dp"

android:ems="10"

android:inputType="textPersonName"

android:hint="User name"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent" />

<EditText

android:id="@+id/editTextTextPassword"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="3dp"

android:layout\_marginTop="50dp"

android:ems="10"

android:hint="Password"

app:layout\_constraintStart\_toStartOf="@+id/editTextTextPersonName" app:layout\_constraintTop\_toBottomOf="@+id/editTextTextPersonName" />

<Button

android:id="@+id/button2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="40dp"

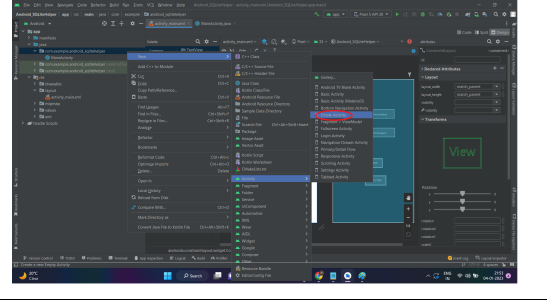
android:text="Register"

app:layout\_constraintStart\_toStartOf="@+id/button"

app:layout\_constraintTop\_toBottomOf="@+id/button" />

</androidx.constraintlayout.widget.ConstraintLayout>

This will be the resulting design

**Step 3:** **Create an Empty Activity called ‘Registration’**

**Step 4:**

Paste this xml code in ‘app/src/main/res/layout/activity\_registration.xml’ to change the design

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

<androidx.constraintlayout.widget.ConstraintLayout 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"

tools:context=".Registration">

<Button

android:id="@+id/button3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="100dp"

android:text="Register"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/editTextTextPassword2" />

<EditText

android:id="@+id/editTextTextPersonName2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="178dp"

android:ems="10"

android:inputType="textPersonName"

android:hint="Username"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintTop\_toTopOf="parent" />

This will be the resulting design

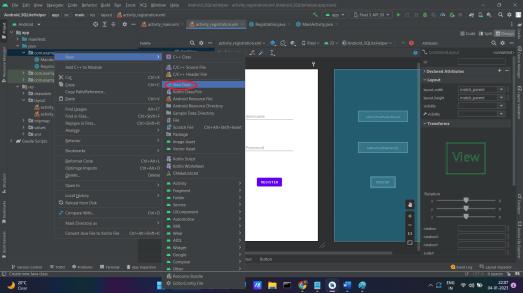


**Step 5**: create a java class called ‘SQLiteHelper’

➔ Right click on ‘app/java/com.example.android\_sqllitehelper’

➔ new -> Java Class

➔ name: ‘SQLiteHelper’



**Step 6:**

Paste the following java code in ‘SQLiteHelper.java’

package com.example.android\_sqlitehelper;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

import android.database.sqlite.SQLiteStatement;

import androidx.annotation.Nullable;

public class SQLiteHelper extends SQLiteOpenHelper {

public SQLiteHelper(@Nullable Context context, @Nullable String name, @Nullable SQLiteDatabase.CursorFactory factory, int version) {

super(context, name, factory, version);

}

public void insertData(String username,String password){

SQLiteDatabase database = this.getWritableDatabase();

String query = "insert into users values(?,?)";

SQLiteStatement statement = database.compileStatement(query);

statement.clearBindings();

statement.bindString(1,username);

statement.bindString(2,password);

statement.executeInsert();

statement.close();

database.close();

}

public void queryData(String sql){

SQLiteDatabase database = getWritableDatabase();

database.execSQL(sql);

}

@Override

public void onCreate(SQLiteDatabase sqLiteDatabase) {

}

@Override

public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

}

}

**Step 7:**

Paste the following code in ‘MainActivity.java’

package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;

import android.database.Cursor;

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 {

private EditText e1,e2;

private Button logButton,registerButton;

private SQLiteHelper helper;

private SQLiteDatabase database;

private String username, password,tempPass;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

e1 = findViewById(R.id.editTextTextPersonName);

e2 = findViewById(R.id.editTextTextPassword);

logButton = findViewById(R.id.button);

registerButton = findViewById(R.id.button2);

helper = new SQLiteHelper(getBaseContext(),"userdb",null,1);

helper.queryData("create table if not exists users(username varchar,password varchar)");

logButton.setOnClickListener(new View.OnClickListener() { @Override

public void onClick(View view) {

username = e1.getText().toString();

password = e2.getText().toString();

try{helper = new SQLiteHelper(getBaseContext(),"userdb",null,1);

final String path =getApplicationContext().getDatabasePath("userdb").getPath(); database = SQLiteDatabase.openOrCreateDatabase(path,null);

Cursor cursor = database.rawQuery("select password from users where username=?",new String[] {username});

if(cursor.getCount() == 0){

Toast.makeText(MainActivity.this, "User not found", Toast.LENGTH\_SHORT).show();

}else{

while(cursor.moveToNext()){

tempPass = cursor.getString(0);

}

if(tempPass.equals(password)){

Toast.makeText(MainActivity.this, "Login Successful", Toast.LENGTH\_SHORT).show(); }else{Toast.makeText(MainActivity.this, "Incorrect Credential", Toast.LENGTH\_SHORT).show();}

}

}catch (Exception e){

System.out.println(e.toString());

}

}

});

registerButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

Intent i = new Intent(MainActivity.this,Registration.class);

startActivity(i);

}

});

}

}

**Step 8:**

Paste the following java code in ‘Registration.java’

package com.example.android\_sqlitehelper;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

public class Registration extends AppCompatActivity {

private EditText e1,e2;

private Button registerButton;

private String username,password;

private static SQLiteHelper helper;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_registration);

e1 = findViewById(R.id.editTextTextPersonName2);

e2 = findViewById(R.id.editTextTextPassword2);

registerButton = findViewById(R.id.button3);

helper = new SQLiteHelper(getBaseContext(),"userdb",null,1);

registerButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

username = e1.getText().toString();

password = e2.getText().toString();

helper.insertData(username,password);

Toast.makeText(Registration.this, "Success", Toast.LENGTH\_SHORT).show();

}

});

}

}

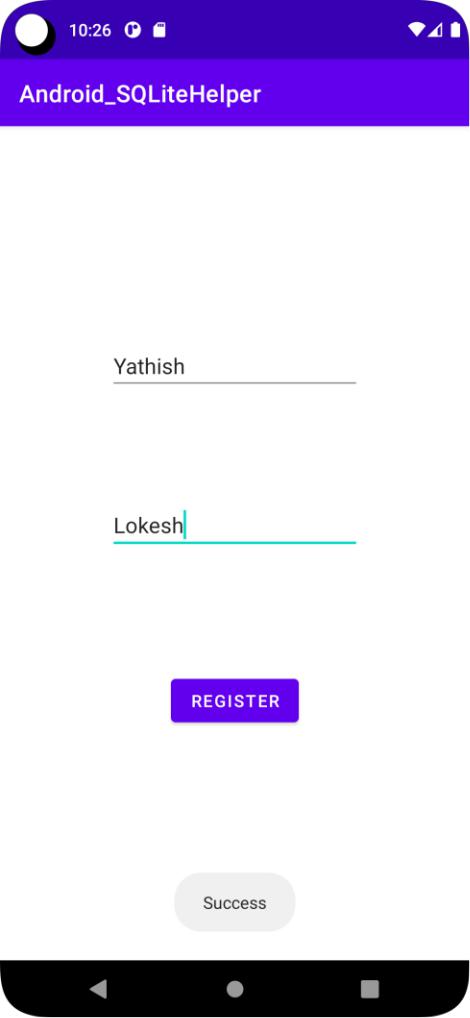
**Step 9:**

Create a Virtual Device and run MainActivity.java

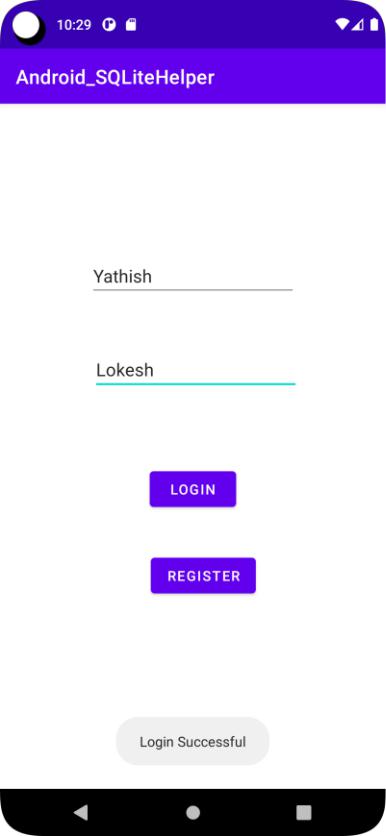
Result:

1 If you click ‘Login’ button without registering a single user or with wrong username, you will get a toast as ‘User not found’

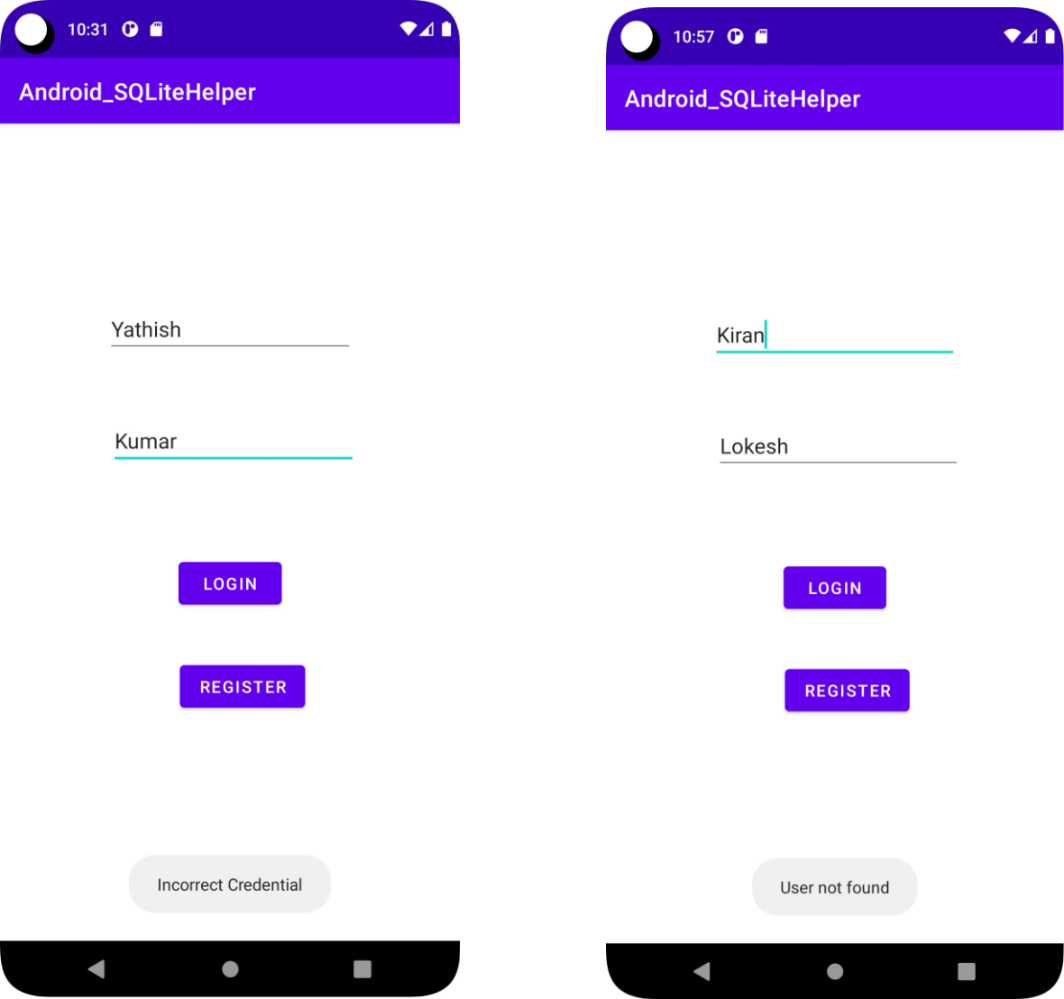
2 Click on ‘Register’ button and register new user



3. Now click on back button and try to login again



Note: For correct credentials it will show ‘Login Successful’ Toast, for wrong password it will show ‘Incorrect Credentials’ and ‘User not found’ will be shown for wrong username as seen below

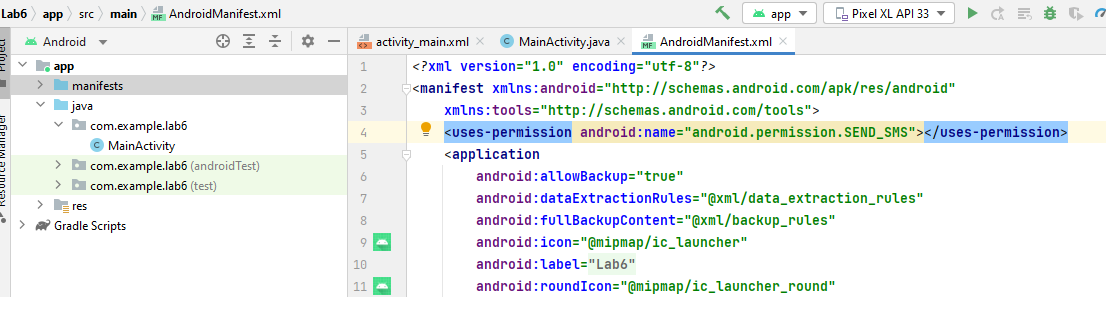


**Lab 6**

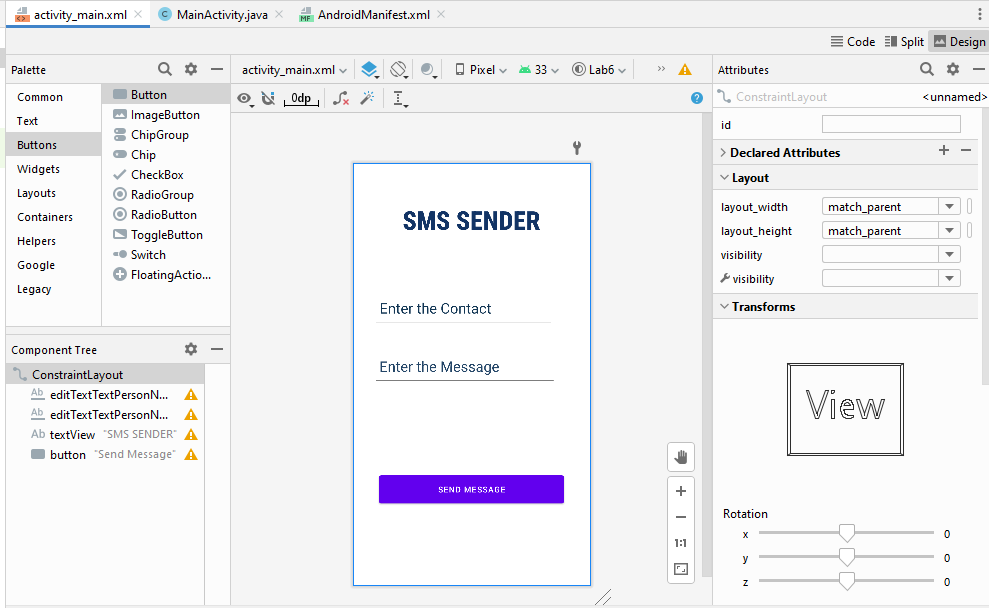
**Develop an application to send SMS to a particular contact from the phonebook**

Follow the Step1 and create the Lab6 Project

Step2 : Wait till gradle Build is successful and open the Manifests file android studio add the send message Permission



Step3:Desgin the activity\_main.xml for sending the message minimum requirement is to get phone number and message two EditText and one button to send Message.



Activity\_main.xml code

<**EditText  
 android:id="@+id/EMessage"  
 android:layout\_width="316dp"  
 android:layout\_height="65dp"  
 android:layout\_marginTop="192dp"  
 android:ems="10"  
 android:hint="Enter the Message"  
 android:inputType="textPersonName"  
 android:paddingLeft="10dp"  
 android:textColorHint="#234567"  
 android:textSize="25dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.375"  
 app:layout\_constraintStart\_toStartOf="parent"** />  
  
<**EditText  
 android:id="@+id/Contact"  
 android:layout\_width="311dp"  
 android:layout\_height="65dp"  
 android:ems="10"  
 android:hint="Enter the Contact"  
 android:inputType="textPersonName"  
 android:paddingLeft="10dp"  
 android:textColorHint="#123654"  
 android:textSize="25dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.36"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.329"** />  
  
<**TextView  
 android:id="@+id/Title"  
 android:layout\_width="match\_parent"  
 android:textAlignment="center"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="65dp"  
 android:layout\_marginBottom="91dp"  
 android:fontFamily="sans-serif-condensed-medium"  
 android:text="SMS SENDER"  
 android:textColor="#123465"  
 android:textSize="45dp"  
 android:textStyle="bold"  
  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.261"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="1.0"** />  
  
<**Button  
 android:id="@+id/Send"  
 android:layout\_width="match\_parent"  
 android:layout\_margin="45dp"  
 android:layout\_height="60dp"  
 android:text="Send Message"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.448"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.844"** />

Java Code for Main Activity

Note Please Add contacts before Running the Program in the AVD

EditText **Contact** ,**Emessage**;  
Button **send**;  
@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **send** = findViewById(R.id.***Send***);  
 **Contact**=findViewById(R.id.***Contact***);  
 **Emessage**=findViewById(R.id.***EMessage***);  
 **Contact**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Intent i = **new** Intent(Intent.***ACTION\_GET\_CONTENT***);

i.setType(ContactsContract.CommonDataKinds.Phone.***CONTENT\_ITEM\_TYPE***);  
 *// this is to Enable the intent to Access the Contacts present in the Android Contact  
 // and send the Contact details to app* startActivityForResult(i,1);  
 }  
 });  
 **send**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **try** {  
 SmsManager smg = SmsManager.*getDefault*();

//This is used get the smsManager from the android Device

smg.sendTextMessage(**Contact**.getText().toString(), **null**, **Emessage**.getText().toString(), **null**, **null**);

//This is line will use smsManager to send mail from the app.

Toast.*makeText*(MainActivity.**this**, **"Message Send"**, Toast.***LENGTH\_LONG***).show();  
 }**catch** (Exception e)  
 { Log.*e*(**"Message"**,e.toString());  
 Toast.*makeText*(MainActivity.**this**, e.toString(), Toast.***LENGTH\_LONG***).show();  
 }  
 }  
 });  
}  
  
@Override  
**protected void** onActivityResult(**int** requestCode, **int** resultCode, @Nullable Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 **if**(requestCode==1)  
 {  
 **if**(resultCode==***RESULT\_OK***)  
 {  
 **try**{  
  
 Uri contactData = data.getData();

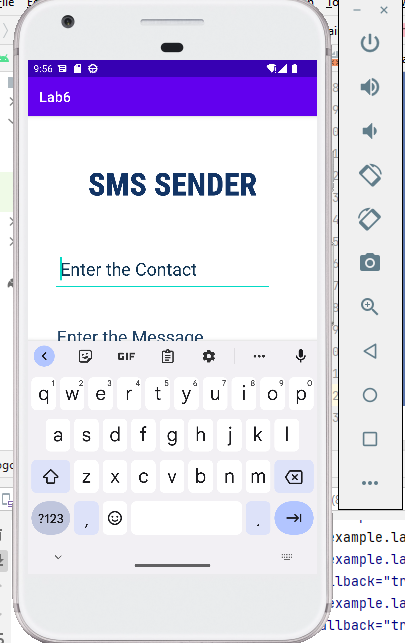
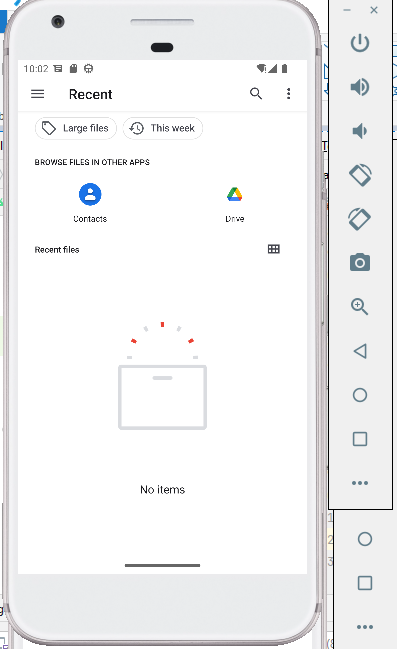
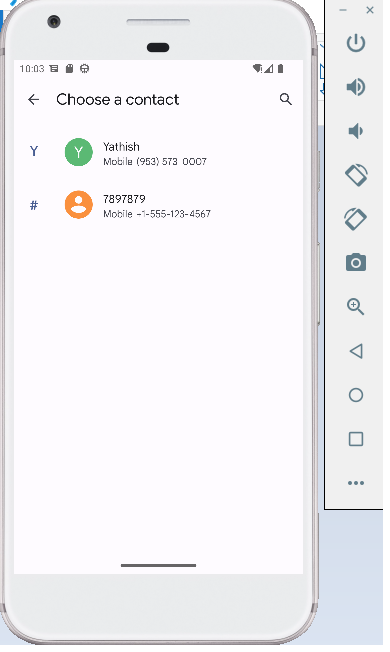
//Accessing the data which is send by the intent Activity  
 Cursor = managedQuery(contactData,**null**,**null**,**null**,**null**);

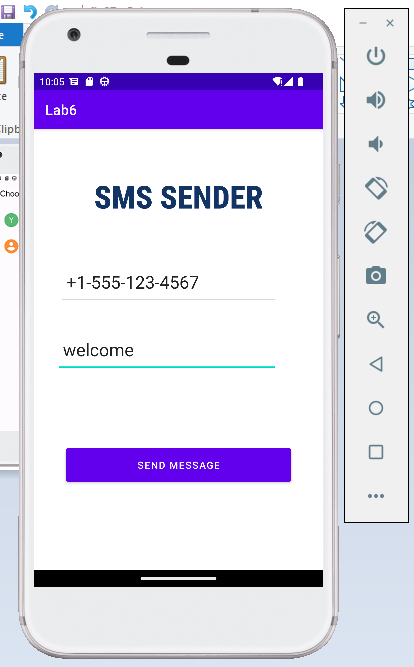
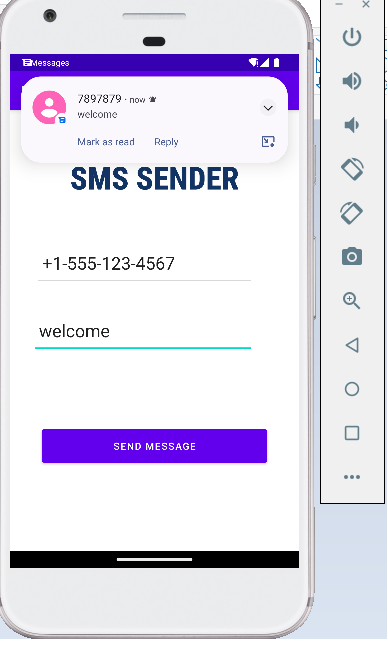
//Use to traversal the data which is got from activity  
 cursor.moveToFirst();

//Accessing the First Data in Got data  
 String number=**"Contact Numbere"**;  
  
  
  
 **int** xz=cursor.getColumnIndex(**"data1"**);

//Getting the Index where Phone Number Exists  
 number=cursor.getString(xz);

//Getting the phone number from the Contact Data;  
  
 **Contact**.setText(number);  
  
 }  
 **catch** (Exception e )  
 {  
 **Contact**.setText(e.toString());  
 }  
 }  
 }  
}

Output

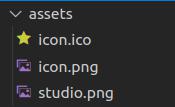


**LAB 7**

**7. Create a login page to authenticate a user using PWA with Manifest file**

**Step 1:** Place all the icons in the assets directory

**Note:** Icons must be square and greater than or equal to 144x144px



**Step 2:** Create the manifest file

manifest.json

{

"name": "User Authentication",

"short\_name": "Login",

"display": "standalone",

"start\_url": "/",

"id": "/",

"Orientation": "portrait\_primary",

"description": "Login using this application",

"icons": [

{

"src": "assets/icon.png",

"type": "image/png",

"sizes": "512x512"

},

{

"src": "assets/studio.png",

"type": "image/png",

"sizes": "128x128"

}

],

"background\_color": "#0000FF",

"theme\_color": "#0000CC"

}

**Step 3:** Create the authentication page in index.html

<!DOCTYPE html>

<html>

<head>

<title>Login</title>

// linking the manifest file

<link rel="manifest" href="manifest.json" /> <link rel="stylesheet" href="index.css" />

<link rel="icon" type="image/x-icon" href="assets/icon.ico" />

</head>

<body>

<center>

<h2>Login to the Application</h2>

<table>

<form id="form">

<tr>

<th>Username:</th>

<td>

<input id="username" type="text" required /> </td>

</tr>

<tr>

<th>Password:</th>

<td>

<input id="password" type="password" required /> </td>

</tr>

<tr>

<td colspan=2 align="center">

<input type="submit" id="submit" value="Login" /> </td>

</tr>

</form>

</table>

</center>

<script>

* storing username and password in local storage localStorage.setItem("username", "qwerty"); localStorage.setItem("password", "123456");

const login = (e) => {

e.preventDefault();

* getting the username and password from form inputs var un = document.getElementById("username").value; var ps = document.getElementById("password").value;

// getting the username and password from local storage var username = localStorage.getItem("username");

var password = localStorage.getItem("password");

if(username == un && password == ps) { alert("Login successful!");

} else {

alert("Login failed!");

}

}

* running the ‘login’ function on form submit event document.getElementById("form").addEventListener("submit",

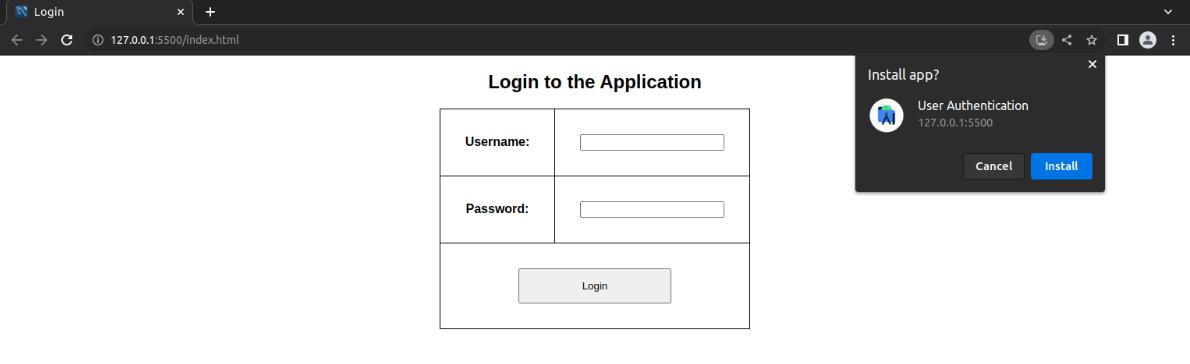
login);

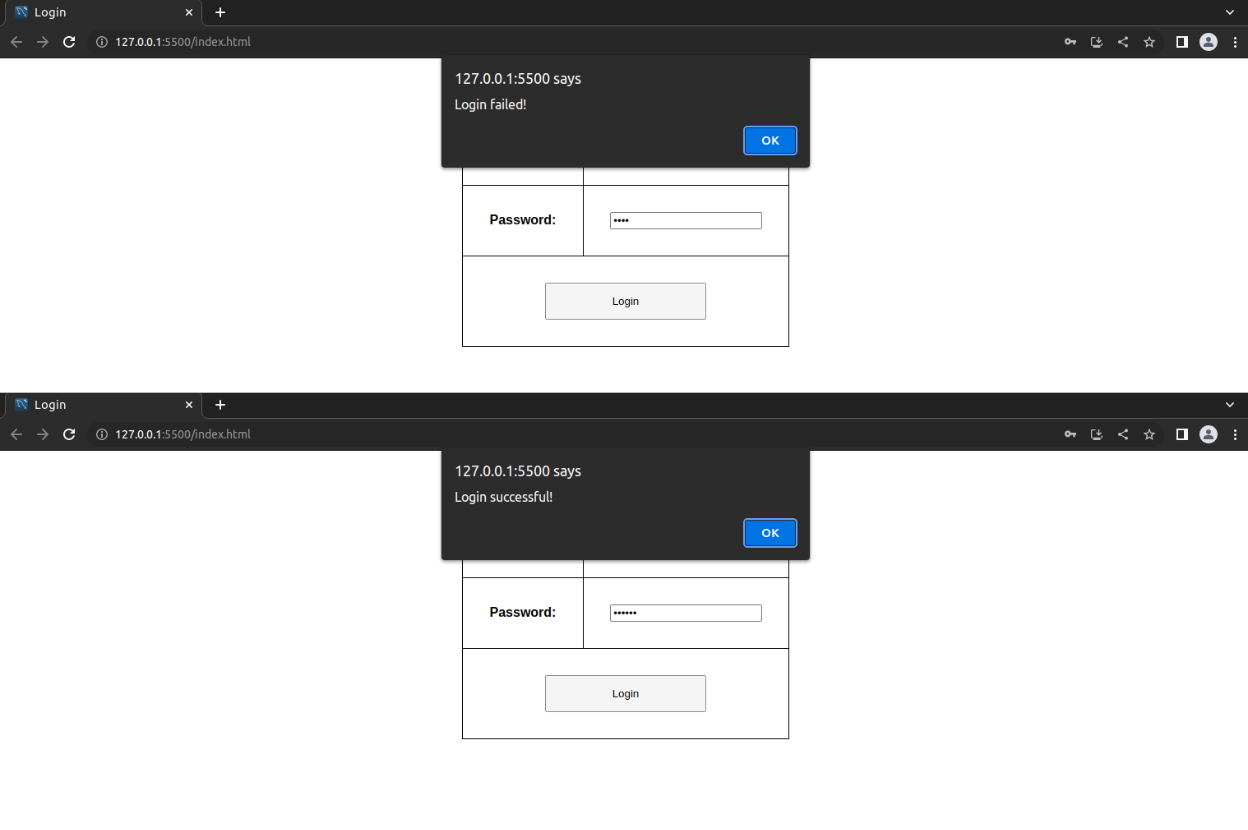
</script>

</body>

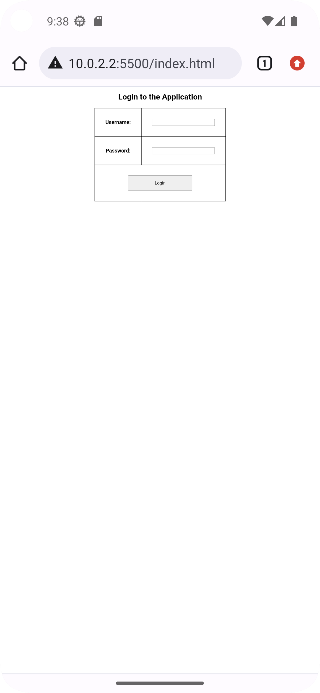
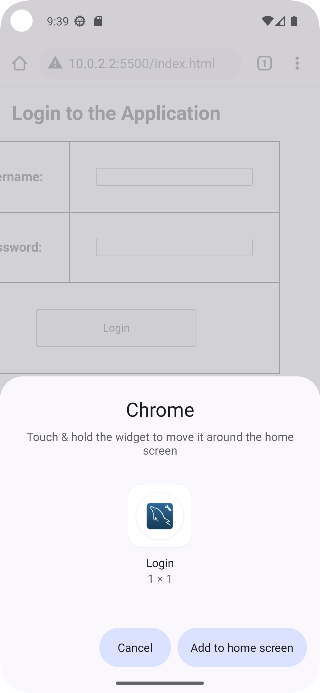
</html>

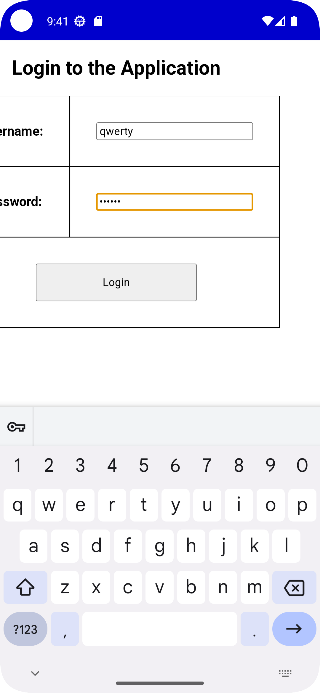
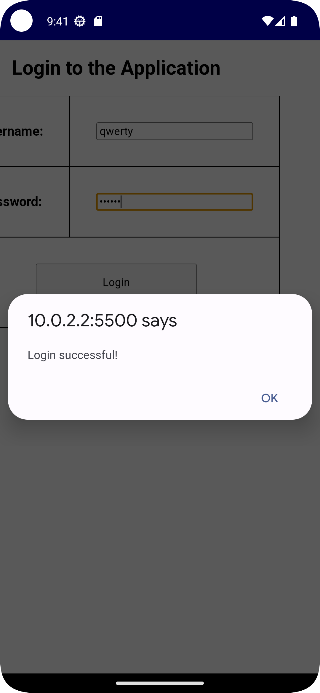
**Output-**

****



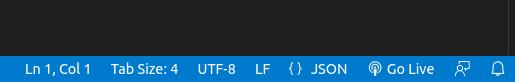
Output





**Running the program-**

Run the live server in VS Code by pressing the Go Live button



To view the output on the Android Studio Emulator-

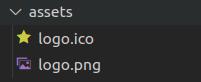
1. Run the AVD
2. Open the AVD settings
3. Search for proxy settings
4. Get the ip address of the system using ifconfig and set it as proxy with port 5500
5. Open chrome in the emulator and enter the ip address of the system as url

**LAB 8**

**8. Build a simple web page using PWA by adding a Service Worker**

**Step 1:** Place all the icons in the assets directory

**Note:** Icons must be square and greater than or equal to 144x144px



**Step 2:** Create the manifest file

manifest.json

{

"name": "Music Academy",

"short\_name": "Registration",

"display": "standalone",

"start\_url": "/",

"id": "/",

"Orientation": "portrait\_primary",

"description": "Register to the Music Academy",

"icons": [

{

"src": "assets/logo.png",

"type": "image/png",

"sizes": "512x512"

}

],

"background\_color": "#00AA00",

"theme\_color": "#00CC00"

}

**Step 3:** Create the service worker file

sw.js

self.addEventListener("install", (e) => { console.log("installed");

})

self.addEventListener("activate", (e) => { console.log("activated");

})

self.addEventListener("fetch", (e) => { console.log("fetched");

})

**Step 3:** Create any simple page in index.html

<!DOCTYPE html>

<html>

<head>

<title>Register</title>

// linking the manifest file

<link rel="manifest" href="manifest.json" /> <link rel="stylesheet" href="index.css" />

<link rel="icon" type="image/x-icon" href="assets/logo.ico" />

</head>

<body>

<center>

<h2>Register to the the Music Academy</h2> <table>

<form id="form">

<tr>

<th>USN:</th>

<td>

<input id="usn" type="text"

pattern="1RV[0-9]{2}[A-Z]{2}[0-9]{3}" required />

</td>

</tr>

<tr>

<th>Name:</th>

<td>

<input id="name" type="text" required /> </td>

</tr>

<tr>

<th>Place:</th>

<td>

<input id="place" type="text" required /> </td>

</tr>

<tr>

<th>No of days:</th>

<td>

<input id="nod" type="number" min="4" max="14"

required />

</td>

</tr>

<tr>

<td colspan=2 align="center">

<input type="submit" id="submit"

value="REGISTER" />

</td>

</tr>

</form>

</table>

</center>

<script>

* checking if service workers are supported by the browser if('serviceWorker' in navigator) {
  + registering the service worker

navigator.serviceWorker.register("sw.js")

.then((e) => {

console.log("registered");

});

} else {

console.log("browser does not support service worker");

}

const register = (e) => {

e.preventDefault();

alert("Registered!");

}

document.getElementById("form").addEventListener("submit",

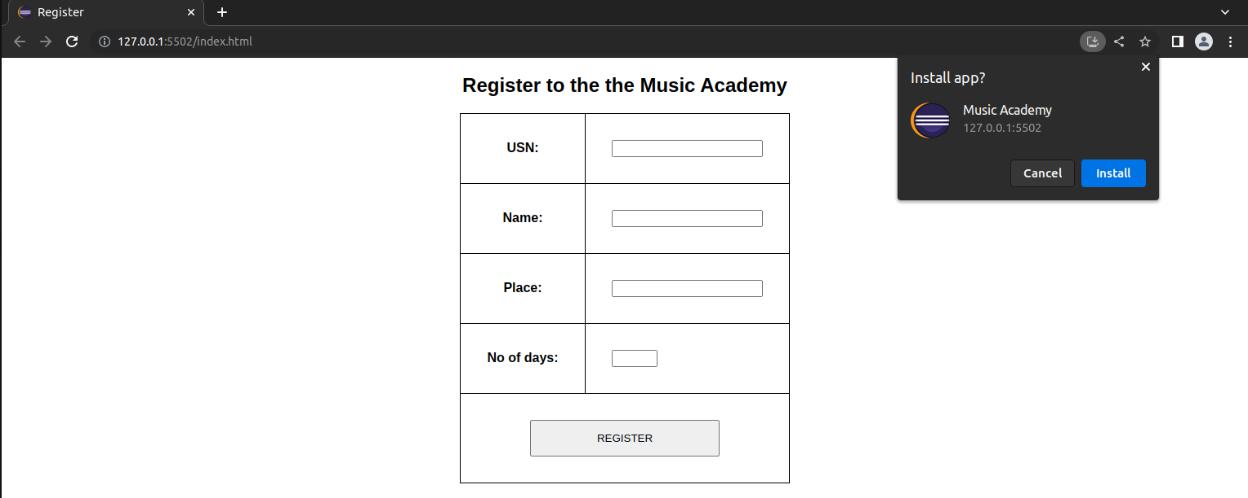
register);

</script>

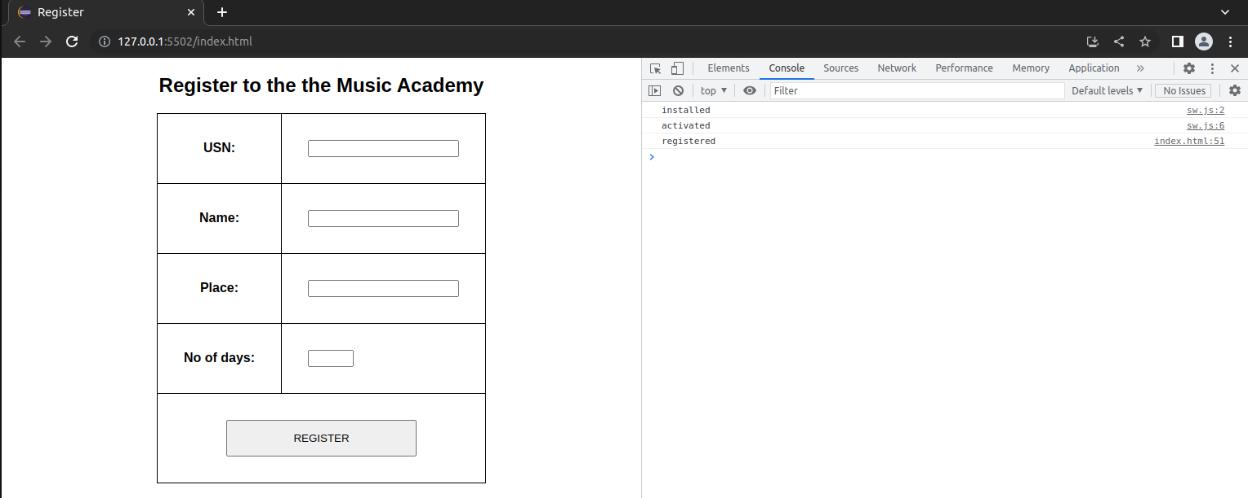
</body>

</html>

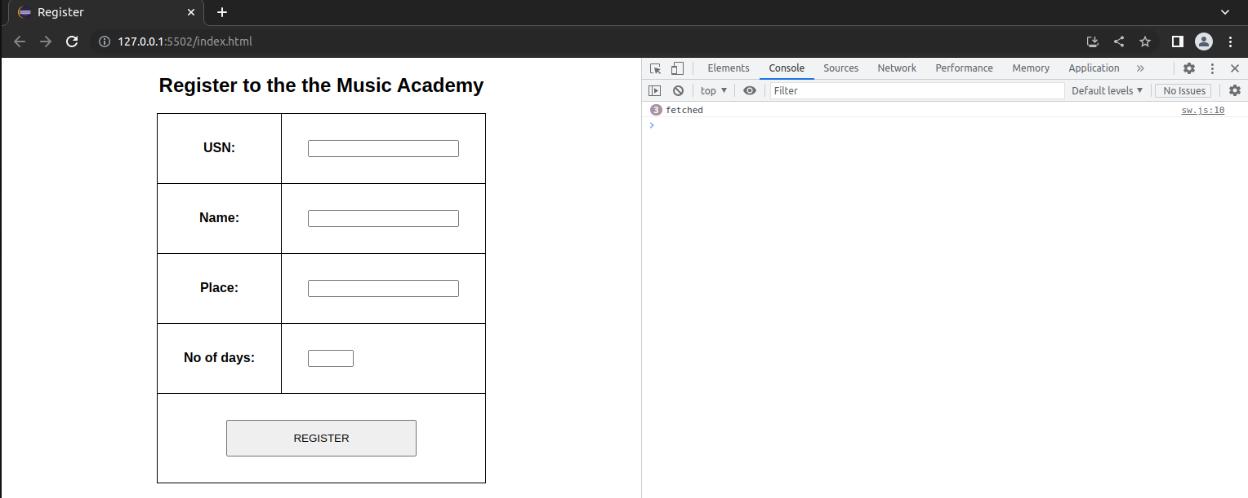
**Output-**

****

On loading for the first time- registered, installed, activated must appear



On reload- fetched must appear



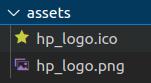
**LAB9**

1. **Devise a PWA to fetch the Github Details for a particular person using the Fetch**

**API**

**Step 1:** Place all the icons in the assets directory

**Note:** Icons must be square and greater than or equal to 144x144px



**Step 2:** Create the manifest file

manifest.json

{

"name": "Fetch Github Details",

"short\_name": "API Fetch",

"display": "standalone",

"start\_url": "/",

"id": "/",

"Orientation": "portrait\_primary",

"description": "Get user details from Github using it's API",

"icons": [

{

"src": "assets/hp\_logo.png",

"type": "image/png",

"sizes": "256x256"

}

],

"background color": "#AA0000",

"theme\_color": "#CC0000"

}

**Step 3:** Create the service worker file

sw.js

self.addEventListener("install", (e) => { console.log("installed");

})

self.addEventListener("activate", (e) => { console.log("activated");

})

self.addEventListener("fetch", (e) => { console.log("fetched");

// fetching user data from the github api

fetch("https://api.github.com/users/your\_github\_username")

.then((e) => {

return e.json();

})

.then((json) => {

console.log(json);

})

.catch((e) => {

console.log(e);

});

})

**Step 4:** Create a html page to register the service worker index.html

<!DOCTYPE html>

<html>

<head>

<title>Register</title>

// linking the manifest file

<link rel="manifest" href="manifest.json" /> <link rel="stylesheet" href="index.css" />

<link rel="icon" type="image/x-icon" href="assets/hp\_logo.ico" />

</head>

<body>

<center>

<h2>Fetch From Github API</h2>

</center>

<script>

* checking if service workers are supported by the browser if('serviceWorker' in navigator) {
  + registering the service worker

navigator.serviceWorker.register("sw.js")

.then((e) => {

console.log("registered");

});

} else {

console.log("browser does not support service worker");

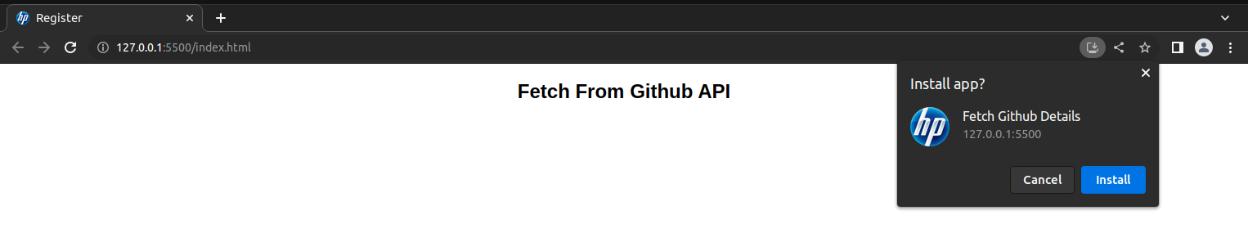
}

</script>

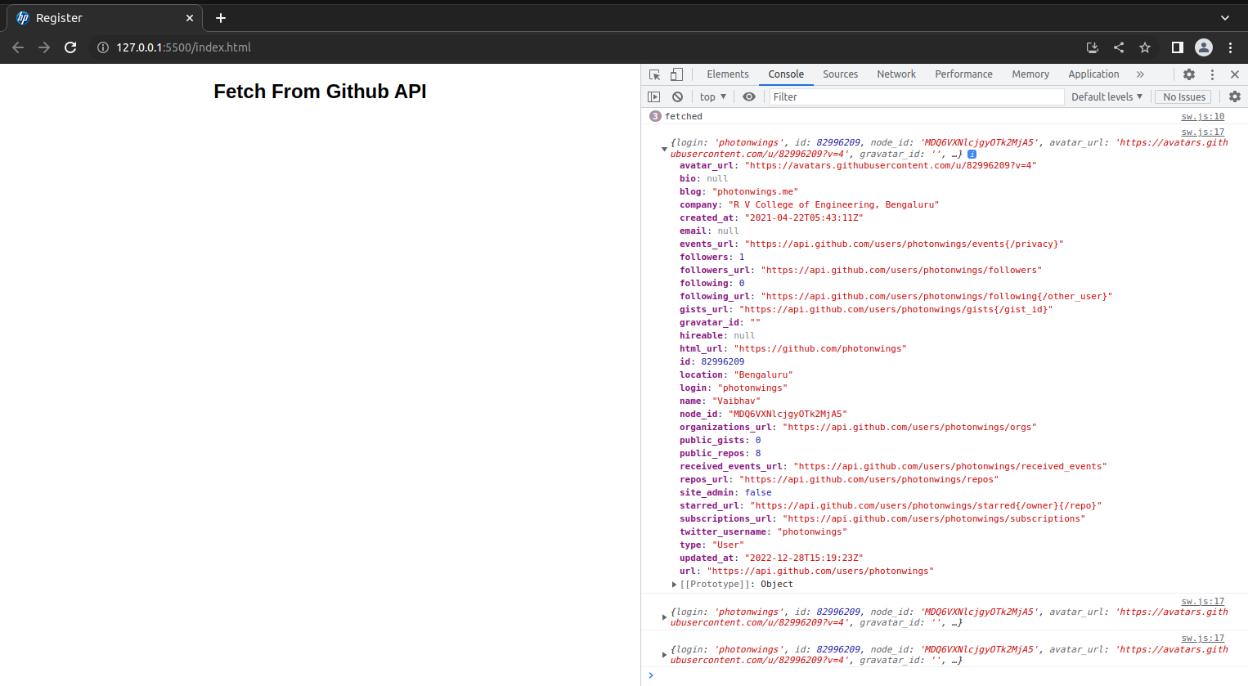
</body>

</html>

**Output- (make sure to unregister the service worker from the previous programs in the applications tab of the inspect menu)**

****

On reload-

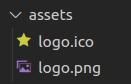


**LAB 10**

**10. Build an application to do a stock display using PWA using a raw JSON file**

**Step 1:** Place all the icons in the assets directory

**Note:** Icons must be square and greater than or equal to 144x144px



**Step 2:** Create the manifest file

manifest.json

{

"name": "Stock Display",

"short\_name": "Stock-Display",

"display": "standalone",

"start\_url": "/",

"id": "/",

"description": "Display stock from json file",

"icons": [

{

"src": "assets/logo.png",

"type": "image/png",

"sizes": "512x512",

"purpose": "maskable"

},

{

"src": "assets/logo.png",

"type": "image/png",

"sizes": "512x512",

"purpose": "any"

}

],

"background\_color": "#00AA00",

"theme\_color": "#00CC00"

}

**Step 3:** Create the service worker file

sw.js

self.addEventListener("install", (e) => { console.log("installed");

* creating a cache storage and adding all the files required for the web page to run even when it’s offline

caches.open("stock").then((cache) => { cache.add("/"); cache.add("./assets/logo.ico"); cache.add("./assets/logo.png"); cache.add("./data.json"); cache.add("./index.html"); cache.add("./manifest.json"); cache.add("./sw.js");

})

.catch((err) => { console.log(err);

});

})

self.addEventListener("activate", (e) => { console.log("activated");

})

self.addEventListener("fetch", (e) => { console.log("fetched");

* intercepting every fetch request and responding from the cache when it’s offline

e.respondWith(

* + checking if requested data is present in the cache

caches.match(e.request)

.then((res) => {

* respond from cache when fetch function doesn’t return any response(i.e, the system is offline)

return res || fetch(e.request);

})

.catch((err) => { console.log(err);

})

);

})

**Step 4:** Create a json file and fill some stock market data

Sample data - https://gist.github.com/tanveery/4ac939d2ad27954da4c8db13e10ef7bd data.json

[

{

"company": "3M",

"description": "3M, based in Minnesota, may be best known for its Scotch tape and Post-It Notes, but it also produces sand paper, adhesives, medical products, computer screen filters, food safety items, stationery products and many products used in automotive, marine, and aircraft industries.",

"initial\_price": 44.28,

"price\_2002": 56.27,

"price\_2007": 95.85,

"symbol": "MMM"

},

{

"company": "Amazon.com",

"description": "Amazon.com, Inc. is an online retailer in North America and internationally. The company serves consumers through its retail Web sites and focuses on selection, price, and convenience. It also offers programs that enable sellers to sell their products on its Web sites, and their own branded Web sites. In addition, the company serves developer customers through Amazon Web Services, which provides access to technology infrastructure that developers can use to enable virtually various type of business. Further, it manufactures and sells the Kindle e-reader. Founded in 1994 and headquartered in Seattle, Washington.",

"initial\_price": 89.38,

"price\_2002": 17.01,

"price\_2007": 93.43,

"symbol": "AMZN"

},

.

.

.

.

.

.

.

]

**Step 5:** Create a html page to register the service worker and display the stock market data index.html

<!DOCTYPE html>

<html>

<head>

<title>Stock Display</title>

<link rel="manifest" href="manifest.json"> <style>

table{

border-collapse: collapse;

}

td, th {

padding: 1em;

border: 1px solid black;

}

</style>

</head>

<body>

<center>

<h2>Stock Market Data Display</h2>

<div id="stock"></div>

</center>

<script>

if('serviceWorker' in navigator) {

navigator.serviceWorker.register("sw.js")

.then((e) => {

if(e.active === null){

console.log("registered");

}

});

} else {

console.log("browser does not support service worker");

}

* data from https://gist.github.com/tanveery/4ac939d2ad27954da4c8db13e10ef7bd

fetch("data.json")

.then((res) => {

return res.json();

})

.then((stock) => {

* creating the table for displaying the stock data var table = "<table>";

table +=

"<tr><th>Company</th><th>Description</th><th>Initial Price</th><th>Price in 2007</th><th>Symbol</th></tr>";

for(item of stock) {

table +=

"<tr><td>"+item.company+"</td><td>"+item.description+"</td><td>"+item.init ial\_price+"</td><td>"+item.price\_2007+"</td><td>"+item.symbol+"</td></tr>"

;

}

table += "</table>";

// inserting the table into html

document.getElementById("stock").innerHTML = table;

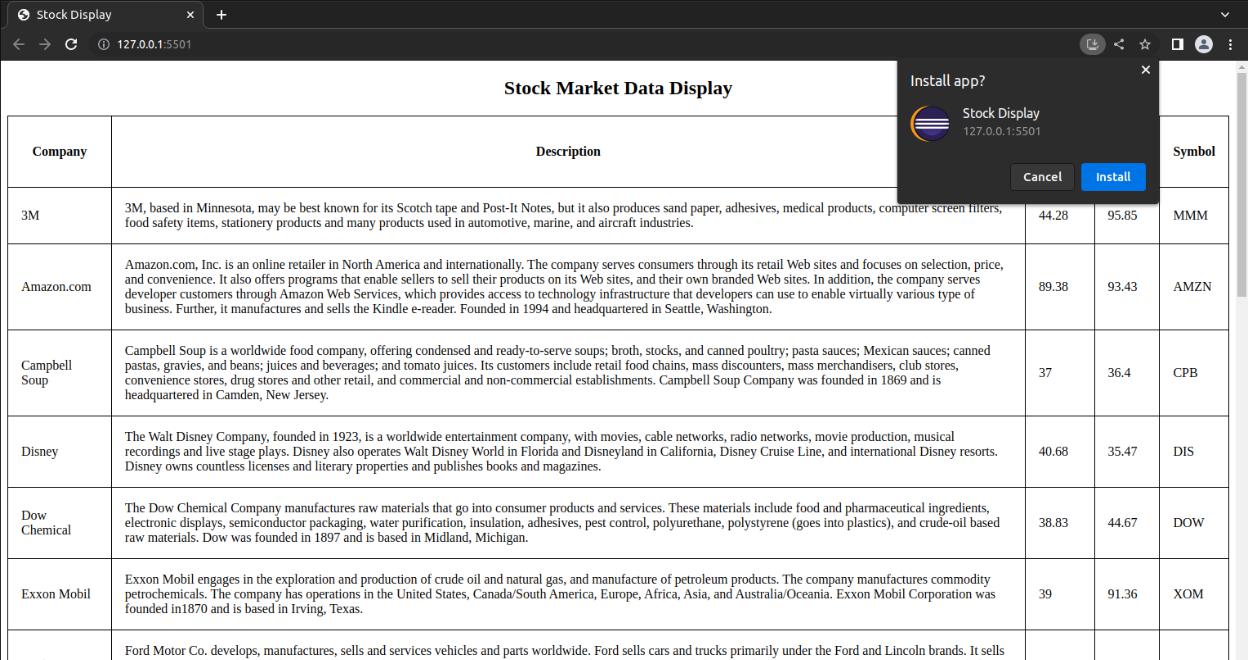
})

</script>

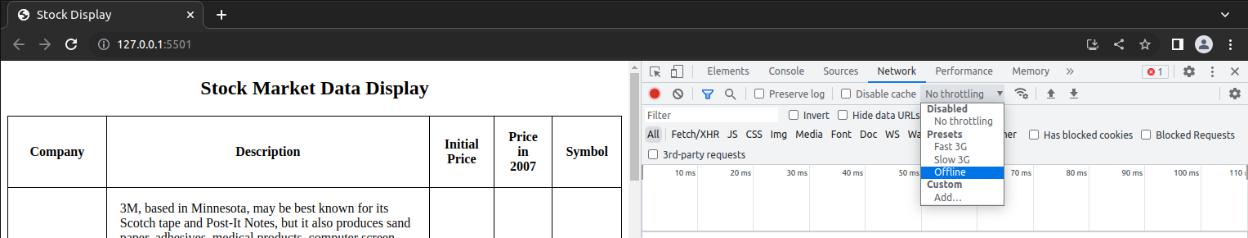
</body>

</html>

**Output- (make sure to unregister the service worker from the previous programs in the applications tab of the inspect menu)**

****

Change the network mode to offline



The website must work even when it’s offline

