**CREATE AN APPLICATION TO IMPLEMENT FACE RECOGANATION**

**AIM:**

**TO CREATE AN APPLICATION TO IMPLEMENT FACE RECOGANATION.**

**ALGORITHM:**

**Step 1: File → New Project Provide the application name and Click “Next”**

**Step 2: Select the target android devices, Select the minimum SDK to run the application. Click “Next”.**

**Step 3: Choose the activity for the application (By default choose “Blank Activity). Click “Next”.**

**Step 4: Enter activity name and click "Finish".**

**Step 5: Edit the program.**

**Step 6: Run the application, 2-ways to run the application**

**1. Running through emulator**

**2. Running through mobile device**

**CODE:**

**AndroidManifest.xml :**

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

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

**package="com.atharvakale.facerecognition">**

**<application**

**android:allowBackup="true"**

**android:label="@string/app\_name"**

**android:largeHeap="true"**

**android:roundIcon="@drawable/face\_icon2"**

**android:supportsRtl="true"**

**android:theme="@style/Theme.FaceRecognition.NoActionBar">**

**<activity**

**android:name=".MainActivity"**

**android:label="@string/app\_name"**

**android:theme="@style/Theme.FaceRecognition.NoActionBar"**

**android:exported="true">**

**<intent-filter>**

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

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

**</intent-filter>**

**</activity>**

**</application>**

**<uses-permission android:name="android.permission.CAMERA" />**

**<uses-feature android:name="android.hardware.camera" android:required="false" />**

**</manifest>**

**Activity\_main.xml**

**<?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:id="@+id/coordinatorLayout"**

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

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

**tools:context=".MainActivity">**

**<Button**

**android:id="@+id/button2"**

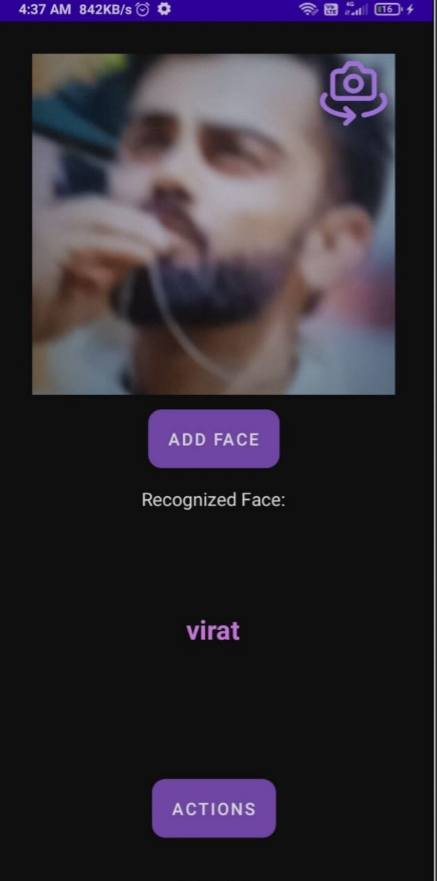
**android:layout\_width="wrap\_content"**

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

**android:background="@drawable/round\_bg"**

**android:text="ACTIONS" app:layout\_constraintBottom\_toBottomOf="parent" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintTop\_toBottomOf="@+id/imageView" app:layout\_constraintVertical\_bias="0.25" /> Main.java package com.atharvakale.facerecognition; import android.Manifest; import android.annotation.SuppressLint; import android.app.Activity; import android.content.Context; import android.content.DialogInterface; import android.content.Intent; import android.content.SharedPreferences; import android.content.pm.PackageManager; import android.content.res.AssetFileDescriptor; import android.graphics.Bitmap; import android.graphics.BitmapFactory; import android.graphics.Canvas; import android.graphics.Color; import android.graphics.ImageFormat; import android.graphics.Matrix; import android.graphics.Paint; import android.graphics.Rect; import android.graphics.RectF; import android.graphics.YuvImage; import android.media.Image; import android.net.Uri; import android.os.Build; import android.os.Bundle; import androidx.annotation.NonNull; import androidx.annotation.RequiresApi; import androidx.appcompat.app.AlertDialog; import androidx.camera.core.CameraSelector; import androidx.camera.core.ImageAnalysis; import androidx.camera.core.ImageProxy; import androidx.camera.core.Preview; import androidx.camera.lifecycle.ProcessCameraProvider; import com.google.android.gms.tasks.OnCompleteListener; import com.google.android.gms.tasks.OnFailureListener; import com.google.android.gms.tasks.OnSuccessListener; import com.google.android.gms.tasks.Task; import com.google.common.util.concurrent.ListenableFuture; import com.google.gson.Gson; import com.google.gson.reflect.TypeToken; import com.google.mlkit.vision.common.InputImage; import com.google.mlkit.vision.face.Face; import com.google.mlkit.vision.face.FaceDetection; import com.google.mlkit.vision.face.FaceDetector; import com.google.mlkit.vision.face.FaceDetectorOptions; import androidx.appcompat.app.AppCompatActivity; import androidx.camera.view.PreviewView; import androidx.core.content.ContextCompat; import androidx.lifecycle.LifecycleOwner; import android.os.ParcelFileDescriptor; import android.text.InputType; import android.util.Pair; import android.util.Size; import android.view.View; import android.widget.Button; import android.widget.EditText; import android.widget.ImageButton; import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast; import org.tensorflow.lite.Interpreter; import java.io.ByteArrayOutputStream; import java.io.FileDescriptor; import java.io.FileInputStream; import java.io.IOException; import java.nio.ByteBuffer; import java.nio.ByteOrder; import java.nio.MappedByteBuffer; import java.nio.ReadOnlyBufferException; import java.nio.channels.FileChannel; import java.util.ArrayList; import java.util.HashMap; import java.util.List; import java.util.Map; import java.util.concurrent.ExecutionException; import java.util.concurrent.Executor; import java.util.concurrent.Executors; public class MainActivity extends AppCompatActivity { FaceDetector detector; private ListenableFuture cameraProviderFuture; PreviewView previewView; ImageView face\_preview; Interpreter tfLite; TextView reco\_name,preview\_info,textAbove\_preview; Button recognize,camera\_switch, actions; ImageButton add\_face; CameraSelector cameraSelector; boolean developerMode=false; float distance= 1.0f; boolean start=true,flipX=false; Context context=MainActivity.this; int cam\_face=CameraSelector.LENS\_FACING\_BACK; //Default Back Camera int[] intValues; int inputSize=112; //Input size for model boolean isModelQuantized=false; float[][] embeedings; float IMAGE\_MEAN = 128.0f; float IMAGE\_STD = 128.0f; int OUTPUT\_SIZE=192; //Output size of model private static int SELECT\_PICTURE = 1; ProcessCameraProvider cameraProvider; private static final int MY\_CAMERA\_REQUEST\_CODE = 100; String modelFile="mobile\_face\_net.tflite"; //model name private HashMap registered = new HashMap<>(); //saved Faces @RequiresApi(api = Build.VERSION\_CODES.M) @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); registered=readFromSP(); //Load saved faces from memory when app starts setContentView(R.layout.activity\_main); face\_preview =findViewById(R.id.imageView); reco\_name =findViewById(R.id.textView); preview\_info =findViewById(R.id.textView2); textAbove\_preview =findViewById(R.id.textAbovePreview); add\_face=findViewById(R.id.imageButton); add\_face.setVisibility(View.INVISIBLE); SharedPreferences sharedPref = getSharedPreferences("Distance",Context.MODE\_PRIVATE); distance = sharedPref.getFloat("distance",1.00f); face\_preview.setVisibility(View.INVISIBLE); recognize=findViewById(R.id.button3); camera\_switch=findViewById(R.id.button5); actions=findViewById(R.id.button2); textAbove\_preview.setText("Recognized Face:"); // preview\_info.setText(" Recognized Face:"); //Camera Permission if (checkSelfPermission(Manifest.permission.CAMERA) != PackageManager.PERMISSION\_GRANTED) { requestPermissions(new String[]{Manifest.permission.CAMERA}, MY\_CAMERA\_REQUEST\_CODE); } //On-screen Action Button actions.setOnClickListener(new View.OnClickListener() { @Override public void onClick(View v) { AlertDialog.Builder builder = new AlertDialog.Builder(context); builder.setTitle("Select Action:"); // add a checkbox list String[] names= {"View Recognition List","Update Recognition List","Save Recognitions","Load Recognitions","Clear All Recognitions","Import Photo (Beta)","Hyperparameters","Developer Mode"}; builder.setItems(names, new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { switch (which) { case 0: displaynameListview(); break; case 1: updatenameListview(); break; case 2: insertToSP(registered,0); //mode: 0:save all, 1:clear all, 2:update all break; case 3: registered.putAll(readFromSP()); break; case 4: clearnameList(); break; case 5: loadphoto(); break; case 6: testHyperparameter(); break; case 7: developerMode(); break; } } }); builder.setPositiveButton("OK", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { } }); builder.setNegativeButton("Cancel", null); // create and show the alert dialog AlertDialog dialog = builder.create(); dialog.show(); } }); //On-screen switch to toggle between Cameras. camera\_switch.setOnClickListener(new View.OnClickListener() { @Override public void onClick(View v) { if (cam\_face==CameraSelector.LENS\_FACING\_BACK) { cam\_face = CameraSelector.LENS\_FACING\_FRONT; flipX=true; } else { cam\_face = CameraSelector.LENS\_FACING\_BACK; flipX=false; } cameraProvider.unbindAll(); cameraBind(); } }); add\_face.setOnClickListener((new View.OnClickListener() { @Override public void onClick(View v) { addFace(); } })); recognize.setOnClickListener(new View.OnClickListener() { @Override public void onClick(View v) { if(recognize.getText().toString().equals("Recognize")) { start=true; textAbove\_preview.setText("Recognized Face:"); recognize.setText("Add Face"); add\_face.setVisibility(View.INVISIBLE); reco\_name.setVisibility(View.VISIBLE); face\_preview.setVisibility(View.INVISIBLE); preview\_info.setText(""); //preview\_info.setVisibility(View.INVISIBLE); } else { textAbove\_preview.setText("Face Preview: "); recognize.setText("Recognize"); add\_face.setVisibility(View.VISIBLE); reco\_name.setVisibility(View.INVISIBLE); face\_preview.setVisibility(View.VISIBLE); preview\_info.setText("1.Bring Face in view of Camera.\n\n2.Your Face preview will appear here.\n\n3.Click Add button to save face."); } } }); //Load model try { tfLite=new Interpreter(loadModelFile(MainActivity.this,modelFile)); } catch (IOException e) { e.printStackTrace(); } //Initialize Face Detector FaceDetectorOptions highAccuracyOpts = new FaceDetectorOptions.Builder() .setPerformanceMode(FaceDetectorOptions.PERFORMANCE\_MODE\_ACCURATE) .build(); detector = FaceDetection.getClient(highAccuracyOpts); cameraBind(); } private void testHyperparameter() { AlertDialog.Builder builder = new AlertDialog.Builder(context); builder.setTitle("Select Hyperparameter:"); // add a checkbox list String[] names= {"Maximum Nearest Neighbour Distance"}; builder.setItems(names, new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { switch (which) { case 0: // Toast.makeText(context, "Clicked", Toast.LENGTH\_SHORT).show(); hyperparameters(); break; } } }); builder.setPositiveButton("OK", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { } }); builder.setNegativeButton("Cancel", null); // create and show the alert dialog AlertDialog dialog = builder.create(); dialog.show(); } private void developerMode() { if (developerMode) { developerMode = false; Toast.makeText(context, "Developer Mode OFF", Toast.LENGTH\_SHORT).show(); } else { developerMode = true; Toast.makeText(context, "Developer Mode ON", Toast.LENGTH\_SHORT).show(); } } private void addFace() { { start=false; AlertDialog.Builder builder = new AlertDialog.Builder(context); builder.setTitle("Enter Name"); // Set up the input final EditText input = new EditText(context); input.setInputType(InputType.TYPE\_CLASS\_TEXT ); builder.setView(input); // Set up the buttons builder.setPositiveButton("ADD", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { //Toast.makeText(context, input.getText().toString(), Toast.LENGTH\_SHORT).show(); //Create and Initialize new object with Face embeddings and Name. SimilarityClassifier.Recognition result = new SimilarityClassifier.Recognition( "0", "", -1f); result.setExtra(embeedings); registered.put( input.getText().toString(),result); start=true; } }); builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { start=true; dialog.cancel(); } }); builder.show(); } } private void clearnameList() { AlertDialog.Builder builder =new AlertDialog.Builder(context); builder.setTitle("Do you want to delete all Recognitions?"); builder.setPositiveButton("Delete All", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { registered.clear(); Toast.makeText(context, "Recognitions Cleared", Toast.LENGTH\_SHORT).show(); } }); insertToSP(registered,1); builder.setNegativeButton("Cancel",null); AlertDialog dialog = builder.create(); dialog.show(); } private void updatenameListview() { AlertDialog.Builder builder = new AlertDialog.Builder(context); if(registered.isEmpty()) { builder.setTitle("No Faces Added!!"); builder.setPositiveButton("OK",null); } else{ builder.setTitle("Select Recognition to delete:"); // add a checkbox list String[] names= new String[registered.size()]; boolean[] checkedItems = new boolean[registered.size()]; int i=0; for (Map.Entry entry : registered.entrySet()) { //System.out.println("NAME"+entry.getKey()); names[i]=entry.getKey(); checkedItems[i]=false; i=i+1; } builder.setMultiChoiceItems(names, checkedItems, new DialogInterface.OnMultiChoiceClickListener() { @Override public void onClick(DialogInterface dialog, int which, boolean isChecked) { // user checked or unchecked a box //Toast.makeText(MainActivity.this, names[which], Toast.LENGTH\_SHORT).show(); checkedItems[which]=isChecked; } }); builder.setPositiveButton("OK", new DialogInterface.OnClickListener() { @Override public void onClick(DialogInterface dialog, int which) { // System.out.println("status:"+ Arrays.toString(checkedItems)); for(int i=0;i**

**OUTPUT:**

****

**RESULT:**

**THE PROGRAM IS SUCCESSFULLY CREATED AND EXECUTED SUCCESSFULLY.**