```java
package com.example.viva;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;

import android.Manifest;
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationManager;
import android.media.AudioManager;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.BatteryManager;
import android.os.Build;
import android.os.Bundle;
import android.os.Looper;
import android.os.Vibrator;
import android.provider.Settings;
import android.speech.RecognizerIntent;
import android.speech.tts.TextToSpeech;
import android.text.format.DateUtils;
import android.view.KeyEvent;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;


import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationCallback;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationResult;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;

import java.io.IOException;
import java.text.DateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;

import java.util.List;
import java.util.Locale;

public class viva extends AppCompatActivity {

    // initializing
    // FusedLocationProviderClient
    // object
    FusedLocationProviderClient mFusedLocationClient;
```

```
    int PERMISSION_ID = 44;

    private TextToSpeech myTTS;


    private TextView chatt;
    private TextView chatu;
    private ImageView mic;
    private AudioManager audioManager;


    private long backPressedTime;
    private Toast backToast;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_viva);

        chatt = findViewById(R.id.chat_viva);
        chatu = findViewById(R.id.chat_user);
        mic = findViewById(R.id.mic);

        mic.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
                promptSpeechInput();
                chatt.setText(" ");
                chatu.setText(" ");
                myTTS.stop();

            }
        });

        mFusedLocationClient =
LocationServices.getFusedLocationProviderClient(this);

        initializeTextToSpeech();


    }

    //start mic with volume down key
    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_VOLUME_DOWN){
            promptSpeechInput();
            chatt.setText(" ");
            chatu.setText(" ");
            myTTS.stop();
            Vibrator vb = (Vibrator)
getSystemService(Context.VIBRATOR_SERVICE);
            vb.vibrate(200);
        }
        return true;
    }
```

```
    /**
     * Showing google speech input dialog
     */
    public void promptSpeechInput() {
        Intent voice = new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        voice.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
        voice.putExtra(RecognizerIntent.EXTRA_LANGUAGE,
Locale.getDefault());
        voice.putExtra(RecognizerIntent.EXTRA_PROMPT, "Speak Now...");
        try {
            startActivityForResult(voice, 1);
        } catch (ActivityNotFoundException a) {
            speak("Sorry!Your device does not support speech input.");
            Toast.makeText(getApplicationContext(),"Sorry!Your device does
not support speech input.",Toast.LENGTH_SHORT).show();
        }
    }


    /**
     * Receiving speech input
     * */

    Context context;
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
        super.onActivityResult(requestCode, resultCode, data);
        context = getApplicationContext();

        audioManager = (AudioManager)
getApplicationContext().getSystemService(Context.AUDIO_SERVICE);

        myTTS.setLanguage(Locale.UK);

        if (requestCode == 1 && resultCode == RESULT_OK && null != data) {

            ArrayList<String> arrayList = data

.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
            chatu.setText(arrayList.get(0));
            String text = arrayList.get(0).toLowerCase();


            if (text.indexOf("time") != -1 ) {
                Date now = new Date();
                String time = DateUtils.formatDateTime(this, now.getTime(),
                        DateUtils.FORMAT_SHOW_TIME);
                speak("Its " + time);
                chatt.setText("Its " + time);
            } else

                if (text.indexOf("date") != -1 ) {
                Calendar calendar = Calendar.getInstance();
                String date =
DateFormat.getDateInstance().format(calendar.getTime());
                speak("Its " + date);
                chatt.setText("Its " + date);
            } else
```

```
            if (text.indexOf("battery") != -1 ) {

                    BatteryManager bm =
(BatteryManager)getSystemService(BATTERY_SERVICE);
                    int percentage =
bm.getIntProperty(BatteryManager.BATTERY_PROPERTY_CAPACITY);
                    IntentFilter ifilter = new
IntentFilter(Intent.ACTION_BATTERY_CHANGED);
                    Intent batteryStatus = context.registerReceiver(null,
ifilter);
                    int status =
batteryStatus.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
                    boolean isCharging = status ==
BatteryManager.BATTERY_STATUS_CHARGING || status ==
BatteryManager.BATTERY_STATUS_FULL;
                    if(isCharging) {
                        speak("Battery is at " + percentage + " percentage
and Charging");
                        chatt.setText("Battery is at " + percentage + "
percentage and Charging");
                    }
                    else if (status ==
BatteryManager.BATTERY_STATUS_DISCHARGING) {
                        speak("Battery is at " + percentage + " percentage
and Not Charging");
                        chatt.setText("Battery is at " + percentage + "
percentage and Not Charging");
                    }
            } else


            if (text.indexOf("set") != -1) {
                AudioManager am = (AudioManager)
getSystemService(Context.AUDIO_SERVICE);

                if (text.indexOf("vibrate") != -1) {
                    am.setRingerMode(1);
                    chatt.setText("Phone is on Vibrate");
                    speak("Phone is on Vibrate");
                } else if (text.indexOf("ring") != -1) {
                    am.setRingerMode(2);
                    chatt.setText("Phone is on Ring");
                    speak("Phone is on Ring");
                } else {
                    Toast.makeText(getApplicationContext(), "I don't
recognise, Plz try again", Toast.LENGTH_SHORT).show();
                    speak("I don't recognise, Please try again");
                }
            } else

            if (text.indexOf("how") != -1 ) {
                if (text.contains("old are you")) {
                    speak("I'm a new born baby...");
                    chatt.setText("I'm a new born baby");
                } else
                if (text.contains("are you")) {
                    speak("I'm doing great, thanks for asking. Anything I
can help with");
                    chatt.setText("I'm doing great, thanks for asking.
```

```
Anything I can help with");
                } else {
                    Toast.makeText(getApplicationContext(), "I don't
understand, Plz try again", Toast.LENGTH_SHORT).show();
                    speak("I don't recognise, Please try again");
                }
            } else


            if (text.indexOf("what") != -1 ) {
                if (text.contains("your name") ) {
                    speak("My Name is Viva");
                    chatt.setText("My Name is Viva");
                } else
                if (text.contains("meaning of viva") ||
text.contains("meaning of weaver") || text.contains("meaning of vivah") ) {
                    speak("Viva stands for Visually Impaired's Voice
Assistant");
                    chatt.setText("Viva stands for Visually Impaired's
Voice Assistant");
                } else
                if (text.contains("you do") || text.contains("you doing"))
{
                    speak("I am your Personal Assistant, I'm here to help
you");
                    chatt.setText("I am your Personal Assistant");
                } else {
                    Toast.makeText(getApplicationContext(), "I don't
understand, Plz try again", Toast.LENGTH_SHORT).show();
                    speak("I don't recognise, Please try again");
                }
            } else

            if (text.indexOf("on") != -1) {
                if (text.contains("bluetooth")) {
                    Intent bluetooth = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                    startActivityForResult(bluetooth, 1);
                    Toast.makeText(getApplicationContext(),"Bluetooth
Turned ON",Toast.LENGTH_SHORT).show();
                    speak("Bluetooth turned on");
                }
            } else
            if(text.indexOf("off") != -1){
                if(text.contains("bluetooth") ){
                    BluetoothAdapter bluetooth =
BluetoothAdapter.getDefaultAdapter();
                    bluetooth.disable();
                    Toast.makeText(getApplicationContext(),"Bluetooth
Turned OFF",Toast.LENGTH_SHORT).show();
                    speak("Bluetooth turned off");
                }
            } else


            if(text.indexOf("where am i") != -1 || text.indexOf("current
location") != -1){
                    getLastLocation();
                    Toast.makeText(getApplicationContext(), "wait a
sec...", Toast.LENGTH_LONG).show();
            }else
```

```
            if(text.indexOf("volume") != -1  ){
                    if(text.contains("up")){

audioManager.adjustVolume(AudioManager.ADJUST_RAISE,
AudioManager.STREAM_MUSIC);
                        speak("Volume increased");
                        chatt.setText("Volume increased");
                    } else
                    if(text.contains("down")){

audioManager.adjustVolume(AudioManager.ADJUST_LOWER,
AudioManager.STREAM_MUSIC);
                        speak("Volume Decreased");
                        chatt.setText("Volume Decreased");
                    }
            } else

            if(text.indexOf("check network") != -1 ){
                boolean wifiConnected;
                boolean mobileConnected;
                ConnectivityManager connMgr = (ConnectivityManager)
                        getSystemService(Context.CONNECTIVITY_SERVICE);
                NetworkInfo activeInfo = connMgr.getActiveNetworkInfo();
                if (activeInfo != null && activeInfo.isConnected()) {
//connected with either mobile or wifi
                    wifiConnected = activeInfo.getType() ==
ConnectivityManager.TYPE_WIFI;
                    mobileConnected = activeInfo.getType() ==
ConnectivityManager.TYPE_MOBILE;
                    if (wifiConnected) { //wifi connected
                        chatt.setText("Connected with Wifi");
                        speak("Connected with Wifi");
                    } else if (mobileConnected) { //mobile data connected
                        chatt.setText("Connected with Mobile Data
Connection");
                        speak("Connected with Mobile Data Connection");
                    }
                } else { //no internet connection
                    chatt.setText("No internet connection");
                    speak("No internet connection");
                }

            } else


            if (text.indexOf("call") != -1) {
                     String Num = text.replaceAll("[^0-9]", "");
                     Intent c = new Intent(Intent.ACTION_CALL);
                     if (Num.trim().isEmpty()) {
                         Toast.makeText(this, "Plz tell the correct
number", Toast.LENGTH_SHORT).show();
                     } else {
                         c.setData(Uri.parse("tel:" + Num));
                     }
                     if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.CALL_PHONE) != PackageManager.PERMISSION_GRANTED) {
                         Toast.makeText(this, "Please grant the permission
to call", Toast.LENGTH_SHORT).show();
                         speak("Please grant the permission to call");
```

```
                        requestPermission();
                    } else {
                        startActivity(c);
                        speak("Calling "+Num);
                    }
                    chatu.setText("Calling " + Num);
            } else



            if (text.indexOf("thank you") != -1 || text.indexOf("thanks")
!= -1) {
                speak("You're Always Welcome");
                chatt.setText("You're Always Welcome...");
            } else


            if (text.indexOf("see you later") != -1 || text.indexOf("exit")
!= -1) {
                speak("Bye");
                super.onBackPressed();
                Intent i = new Intent(Intent.ACTION_MAIN);
                i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                i.addCategory(Intent.CATEGORY_HOME);
                startActivity(i);
                finish();
                System.exit(0);
            } else {

                Toast.makeText(getApplicationContext(), "I don't
understand, Plz try again", Toast.LENGTH_SHORT).show();
                speak("I don't recognise, Please try again");
            }

        }
    }



    public void initializeTextToSpeech() {
        myTTS = new TextToSpeech(this, i -> {
            if(myTTS.getEngines().size()==0){
                Toast.makeText(viva.this,"Viva engine not on this
device",Toast.LENGTH_LONG).show();
                finish();
            } else {
                myTTS.setLanguage(Locale.UK);

                Calendar c = Calendar.getInstance();
                int timeOfDay = c.get(Calendar.HOUR_OF_DAY);

                if(timeOfDay >= 0 && timeOfDay < 12){
                    speak("Good Morning. Hi, I am Viva, You can speak to me
by clicking on the volume button");
                    chatt.setText("Good Morning");
                }else if(timeOfDay >= 12 && timeOfDay < 16){
                    speak("Good Afternoon. Hi, I am Viva, You can speak to
me by clicking on the volume button");
                    chatt.setText("Good Afternoon");
                }else if(timeOfDay >= 16 && timeOfDay < 24){
                    speak("Good Evening. Hi, I am Viva, You can speak to me
```

```
by clicking on the volume button");
                    chatt.setText("Good Evening");
                }
            }
        });
    }

    private void requestPermission() {
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.CALL_PHONE},1);
    }

    private void speak(String message) {
        if(Build.VERSION.SDK_INT >= 21) {

            myTTS.speak(message, TextToSpeech.QUEUE_FLUSH,null,null);

        } else {
            Toast.makeText(viva.this,"Viva engine not on this
device",Toast.LENGTH_LONG).show();
        }

    }

    @Override
    protected void onDestroy() {
        if (myTTS != null) {
            myTTS.stop();
            myTTS.shutdown();
        }

        super.onDestroy();
    }



    @SuppressLint("MissingPermission")
    private void getLastLocation() {
        // check if permissions are given
        if (checkPermissions()) {

            // check if location is enabled
            if (isLocationEnabled()) {

                // getting last
                // location from
                // FusedLocationClient
                // object

mFusedLocationClient.getLastLocation().addOnCompleteListener(new
OnCompleteListener<Location>() {
                    @Override
                    public void onComplete(@NonNull Task<Location> task) {
                        Location location = task.getResult();
                        if (location == null) {
                            requestNewLocationData();
                        } else {

                            getAddress(location.getLatitude(),
location.getLongitude());
                        }
```

```
                    }
                });
            } else {
                Toast.makeText(this, "Please turn on" + " your
location...", Toast.LENGTH_LONG).show();
                Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                startActivity(intent);
            }
        } else {
            // if permissions aren't available,
            // request for permissions
            requestPermissions();
        }
    }

    @SuppressLint("MissingPermission")
    private void requestNewLocationData() {

        // Initializing LocationRequest
        // object with appropriate methods
        LocationRequest mLocationRequest = new LocationRequest();

mLocationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
        mLocationRequest.setInterval(5);
        mLocationRequest.setFastestInterval(0);
        mLocationRequest.setNumUpdates(1);

        // setting LocationRequest
        // on FusedLocationClient
        mFusedLocationClient =
LocationServices.getFusedLocationProviderClient(this);
        mFusedLocationClient.requestLocationUpdates(mLocationRequest,
mLocationCallback, Looper.myLooper());
    }

    private LocationCallback mLocationCallback = new LocationCallback() {

        @Override
        public void onLocationResult(LocationResult locationResult) {
            Location mLastLocation = locationResult.getLastLocation();
            getAddress(mLastLocation.getLatitude(),
mLastLocation.getLongitude());
        }
    };

    // method to check for permissions
    private boolean checkPermissions() {
        return ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) ==
PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED;

        // If we want background location
        // on Android 10.0 and higher,
        // use:
        // ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_BACKGROUND_LOCATION) ==
PackageManager.PERMISSION_GRANTED
```

```
    }

    // method to request for permissions
    private void requestPermissions() {
        ActivityCompat.requestPermissions(this, new String[]{
                Manifest.permission.ACCESS_COARSE_LOCATION,
                Manifest.permission.ACCESS_FINE_LOCATION}, PERMISSION_ID);
    }

    // method to check
    // if location is enabled
    private boolean isLocationEnabled() {
        LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        return
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER) ||
locationManager.isProviderEnabled(LocationManager.NETWORK_PROVIDER);
    }

    // If everything is alright then
    @Override
    public void
    onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);

        if (requestCode == PERMISSION_ID) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                getLastLocation();
            }
        }
    }


    public void getAddress(double lat, double lng) {
        Geocoder geocoder = new Geocoder(getApplicationContext(),
Locale.getDefault());
        try {
            List<Address> addresses = geocoder.getFromLocation(lat, lng,
1);
            Address obj = addresses.get(0);
            String add = obj.getAddressLine(0);
            //add = add + "\n" + obj.getCountryName();
            // add = add + "\n" + obj.getCountryCode();
            // add = add + "\n" + obj.getAdminArea();
            // add = add + "\n" + obj.getPostalCode();
            // add = add + "\n" + obj.getSubAdminArea();
             // add = add + "\n" + obj.getLocality();
             // add = add + "\n" + obj.getSubThoroughfare();

            // Log.v("IGA", "Address" + add);
            //Toast.makeText(this, "Address=>" +
add,Toast.LENGTH_SHORT).show();
            chatt.setText(add);
            speak("My Location is "+add);


            // TennisAppActivity.showDialog(add);
        } catch (IOException e) {
```

```
            // TODO Auto-generated catch block
            e.printStackTrace();
            Toast.makeText(this, e.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    }



    @Override
    public void onBackPressed() {

        if (backPressedTime + 2000> System.currentTimeMillis()){
            backToast.cancel();
            super.onBackPressed();
            Intent i = new Intent(Intent.ACTION_MAIN);
            i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            i.addCategory(Intent.CATEGORY_HOME);
            startActivity(i);
            finish();
            System.exit(0);
        }
        else{
            backToast = Toast.makeText(getBaseContext(),"Press back again
to exit",Toast.LENGTH_SHORT);
            backToast.show();
        }

        backPressedTime = System.currentTimeMillis();
    }



}
```