```java
package com.example.tesseractmodule3;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Environment;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.example.tesseractmodule3.View.WritingView;
import com.googlecode.tesseract.android.TessBaseAPI;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Timer;
import java.util.TimerTask;
import java.util.UUID;

public class BHAActivity extends AppCompatActivity {

    public static final String TESS_DATA = "/tessdata";
    private static final String TAG = MainActivity.class.getSimpleName();
    private TessBaseAPI tessBaseAPI;
    private String mCurrentPhotoPath;

    Uri outputFileDir;

    WritingView wv;
    TextView Out;
    Button clr;
    TextView shows;

    private static final int PERMISSION_REQUEST_STORAGE = 1;
    MediaPlayer word,tryagain;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bha);
        if (ContextCompat.checkSelfPermission(BHAActivity.this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED) {
            if
(ActivityCompat.shouldShowRequestPermissionRationale(BHAActivity.this,
Manifest.permission.WRITE_EXTERNAL_STORAGE)) {
                ActivityCompat.requestPermissions(BHAActivity.this,new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE},98);
                ActivityCompat.requestPermissions(BHAActivity.this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, PERMISSION_REQUEST_STORAGE);

            } else {
                ActivityCompat.requestPermissions(BHAActivity.this,new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE},91);
                ActivityCompat.requestPermissions(BHAActivity.this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, PERMISSION_REQUEST_STORAGE);
            }
        }

        Out = findViewById(R.id.output);
        Button result = findViewById(R.id.verify);
        wv = findViewById(R.id.writingview);
        clr = findViewById(R.id.clear);
        shows = findViewById(R.id.show);

        tryagain = MediaPlayer.create(this,R.raw.tryagain);

        word = MediaPlayer.create(this,R.raw.bha);

        checkPermission();
        prepareTessData();

        shows.setText("भ");

        result.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                checkPermission();
                saveimage();

                if(shows.getText().toString().equals("भ") &&
(Out.getText().toString().equals("भ") || Out.getText().toString().equals("क्या") ||
Out.getText().toString().equals("स्म") || Out.getText().toString().equals("भें") ||
Out.getText().toString().equals("मृ") || Out.getText().toString().equals("एँ"))){
```

```
                word.start();
                new Timer().schedule(new TimerTask() {
                    @Override
                    public void run() {
                        // this code will be executed after 4 seconds
                        word.stop();
                        word.reset();
                    }
                }, 3000);

                new Timer().schedule(new TimerTask() {
                    @Override
                    public void run() {
                        // this code will be executed after 4 seconds
                        Intent intent = new Intent(BHAActivity.this,
MAActivity.class);
                        finish();
                        startActivity(intent);
                    }
                }, 4000);
            }
            else {
                tryagain.start();
                new Timer().schedule(new TimerTask() {
                    @Override
                    public void run() {
                        // this code will be executed after 2 seconds
                        tryagain.stop();
                        tryagain.reset();
                    }
                }, 2000);

                new Timer().schedule(new TimerTask() {
                    @Override
                    public void run() {
                        // this code will be executed after 4 seconds
                        Intent intent1 = getIntent();
                        finish();
                        startActivity(intent1);
                    }
                }, 4000);
            }
        }

    });


    clr.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
                checkPermission();
                Intent intent = getIntent();
                finish();
                startActivity(intent);

            }
        });
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        switch (requestCode){
            case PERMISSION_REQUEST_STORAGE: {
                if (grantResults.length>0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED){
                    if(ContextCompat.checkSelfPermission(BHAActivity.this,
Manifest.permission.WRITE_EXTERNAL_STORAGE)==PackageManager.PERMISSION_GRANTED){
                        Toast.makeText(this,"Permission
Granted",Toast.LENGTH_SHORT).show();
                    }else{
                        Toast.makeText(this,"You don't have
Permission!!",Toast.LENGTH_SHORT).show();
                    }
                }
            }
        }
    }

    private void checkPermission() {
        if (ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.READ_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE}, 120);
        }
        if (ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.WRITE_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 121);
        }
    }


    public void saveimage() {

        ActivityCompat.requestPermissions(BHAActivity.this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 100);

        View content = wv;
        content.setDrawingCacheEnabled(true);
        content.setDrawingCacheQuality(View.DRAWING_CACHE_QUALITY_HIGH);
        Bitmap bitmap = content.getDrawingCache();
        String path = Environment.getExternalStorageDirectory().getAbsolutePath();
        File file = new File(path + "/" + "." + UUID.randomUUID().toString()
+".png");
        FileOutputStream ostream;
        try {
            file.createNewFile();
            ostream = new FileOutputStream(file);
            bitmap.compress(Bitmap.CompressFormat.PNG, 100, ostream);
            ostream.flush();
            ostream.close();

        } catch (Exception e) {
            e.printStackTrace();
            Toast.makeText(BHAActivity.this, "Not Saved",
Toast.LENGTH_SHORT).show();
        }

        mCurrentPhotoPath = file.getAbsolutePath();

        startOCR(outputFileDir);

    }

    private void prepareTessData(){
        try{
            File dir = getExternalFilesDir(TESS_DATA);
            if(!dir.exists()){
                if (!dir.mkdir()) {
                    Toast.makeText(getApplicationContext(), "The folder " +
dir.getPath() + "was not created", Toast.LENGTH_SHORT).show();
                }
            }
            String fileList[] = getAssets().list("");
            for(String fileName : fileList){
                String pathToDataFile = dir + "/" + fileName;
                if(!(new File(pathToDataFile)).exists()){
                    InputStream in = getAssets().open(fileName);
                    OutputStream out = new FileOutputStream(pathToDataFile);
                    byte [] buff = new byte[1024];
                    int len ;
                    while(( len = in.read(buff)) > 0){
                        out.write(buff,0,len);
                    }
                    in.close();
                    out.close();
                }
            }
        }catch (IOException e){
            e.printStackTrace();
        }
        catch (NullPointerException e){
            e.printStackTrace();
        }
        catch (Exception e) {
            Log.e(TAG, e.getMessage());
        }
    }


    private void startOCR(Uri imageUri){
        try{

            BitmapFactory.Options options = new BitmapFactory.Options();
            options.inJustDecodeBounds = false;
            options.inSampleSize = 6;
            Bitmap bitmap = BitmapFactory.decodeFile(mCurrentPhotoPath, options);
            String result = this.getText1(bitmap);
            Out.setText(result);
        }catch (Exception e){
            Log.e(TAG, e.getMessage());
        }
    }

    private String getText1(Bitmap bitmap){
        try{
            tessBaseAPI = new TessBaseAPI();
        }catch (Exception e){
            Log.e(TAG, e.getMessage());
        }

        String dataPath = getExternalFilesDir("/").getPath() + "/";
        tessBaseAPI.init(dataPath, "hin",TessBaseAPI.OEM_TESSERACT_ONLY);
        tessBaseAPI.setImage(bitmap);
        String retStr = "No result";

        try{
            retStr = tessBaseAPI.getUTF8Text();
        }catch (Exception e){
            Log.e(TAG, e.getMessage());
        }
        tessBaseAPI.end();
        return retStr;
    }

}
```