**GEOGRAPHICAL LOCATION AND IDENTIFICATION OF PLANTS**

*A*

*Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

**B.MANIKANTA(1602-17-737-070)**

**MVS.MADHAV(1602-17-737-097)**

**DVS.MANOJ(1602-17-737-115)**

**T.VIJAYA SREE(1602-17-737-118)**

*Under the guidance of*

**KRISHNA KISHORE**

**Assistant Professor**



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University) Ibrahimbagh, Hyderabad-31**

**2019-2020**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University) Hyderabad-500 031**

**Department of Information Technology**



**DECLARATION BY THE CANDIDATE**

I, B.**MANIKANTA** , **MVS.MADHAV** , **DVS.MANOJ , T.VIJAYA SREE** bearing hall ticket number **1602-17-737-070, 1602-17-737-097, 1602-17-737-115,1602-17-737-118** hereby declare that the project report entitled **“GEOGRAPHICAL LOCATION AND IDENTIFICATION OF PLANTS”** under the guidance of **KRISHNA KISHORE**, Assistant Professor, Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology.**

This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**B.MANIKANTA(1602-17-737-070)**

**MVS.MADHAV(1602-17-737-097)**

**DVS.MANOJ(1602-17-737-115)**

**T.VIJAYA SREE(1602-17-737-118)**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University) Hyderabad-500 031**

**Department of Information Technology**



**BONAFIDE CERTIFICATE**

Thisis to certify that the project entitled “**GEOGRAPHICAL LOCATION AND IDENTIFICATION OF PLANTS”** being submitted by **MANIKANTA, MADHAV , MANOJ , VIJAYA SREE** bearing **1602-17-737-070,1602-17-737-097,1602-17-737-115,1602-17-737-118** in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Information Technology is a record of bonafide work carried out by him/her under my guidance.

**Krishna Kishore Dr. K. Ram Mohan Rao**

**Assistant Professor HOD , IT**

**Internal Guide**

**ACKNOWLEDGEMENT**

The satisfaction that accompanies that the successful completion of the project would not have been possible without the kind support and help of many individuals. We would like to extend my sincere thanks to all of them.

We would like to take the opportunity to express our humble gratitude to Krishna kishore Sir under whom we executed this project.

We would also use this opportunity to thank our senior. We are grateful to his guidance, and constructive suggestions that helped us in the preparation of this project. His constant guidance and willingness to share his vast knowledge made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks. We would like to thank all faculty members and staff of the Department of Information Technology for their generous help in various ways for the completion of this project.

Finally, yet importantly, We would like to express our heartfelt thanks to our HOD Dr. K. Ram Mohan Rao Sir and classmates for their help and wishes for the successful completion of this project.

**ABSTRACT**

Geographical location and identification of plants is an app which helps in the identification of the plant . This is possible either by capturing the picture or uploading the picture and after it will predict the plant by assigning it probability . We can also know the location of the plant where the plant is its geographical location can be known.

|  |  |
| --- | --- |
| **Contents** | **Pages** |
| 1.Introduction  1.1.Application Description  1.2.Scope | 7 |
| 2.Related Work | 8 |
| 3.Proposed Work  Motivation of the proposal  3.1.Use cases  3.2.UI prototypes or screenshots  3.3.Architecture and Technology used  3.4.Design  3.4.1.UML static diagrams(Class)  3.4.2.UML Runtime  diagrams(sequence,flowchart)  3.5.Implementation  3.5.1.Description of Main class  3.5.2.Specific Algorithms/Logics  3.5.3.Github Links/Folder structure  3.6.Testing | 9  9  9  11  11  12  13  25  26  27 |
| 4.Results | 27 |
| 5.Discussion and Future work | 30 |
| 6.References | 30 |

1. **INTRODUCTION**
   1. **Application Description**

Plant identification is a crucial process in the field of naturals products. In the case presented here, it is hard to confuse the two plants when we look at them closely. But in some case even a close inspection may not be enough for an untrained eyes. We will not be able to easily tell the difference between cannabis and kenaf, even if we identify we will not be able to find that difference, Sometimes it is not an easy task.

So it is very important to identify the plant in order to know what are the uses of it , whether it is a poison or whether it is a medicine. All these we will be able to know.

“Plant Identification is key to Conservation”

**1.2 Scope**

We designed this App mainly for the farmers who wants to know the plant name and where they are located.

**2.RELATED WORK**

There are apps in the playstore that have image identification. There are apps that can identify plant . But some apps provide only camera capturing facility, whereas some apps provide only image upload facility .But our app provides both facilities image capturing and image uploading.

Apart from just image identitfication in our app we can also know the location of the plant.

**3.PROPOSED WORK**

**Motivation of the Proposal**

In today’s world there are many people who are very passionate about plants so they want to know what is the name of the plant they can know that by just clicking a picture of that plant. Also if there are people who want to know which plants are available at which locations this can be helpful for farmers as well as for people who are passionate about plants.

So our app helps in knowing what is the name of the plant and its location

**3.1.Use Cases**

1.Login

The user can login with email id and password

2.Uploading image or capturing picture

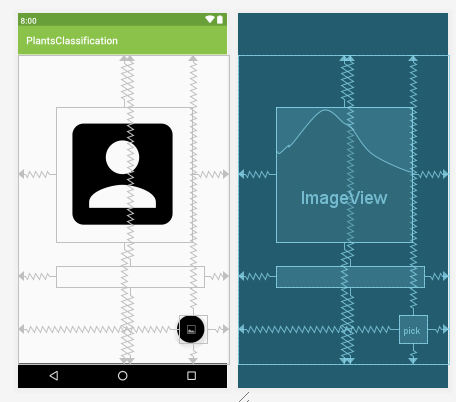
Adding image upload functionality to app and also camera facility.

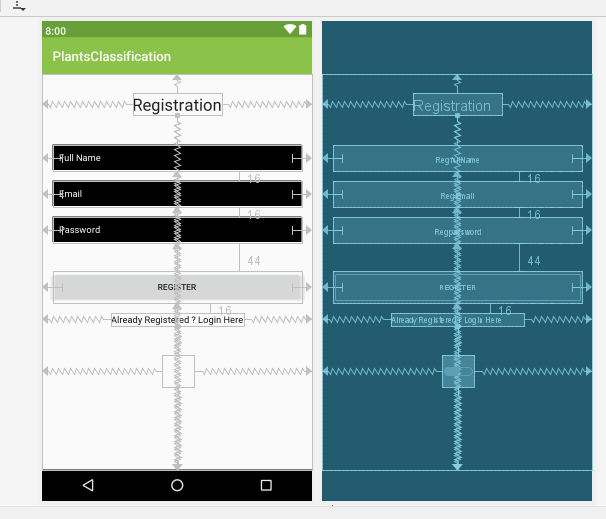
3.  Geographical location

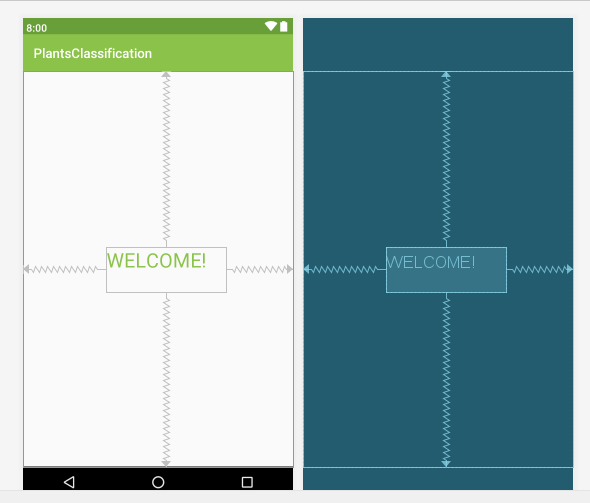
Finding geographical location of plant by using google map API

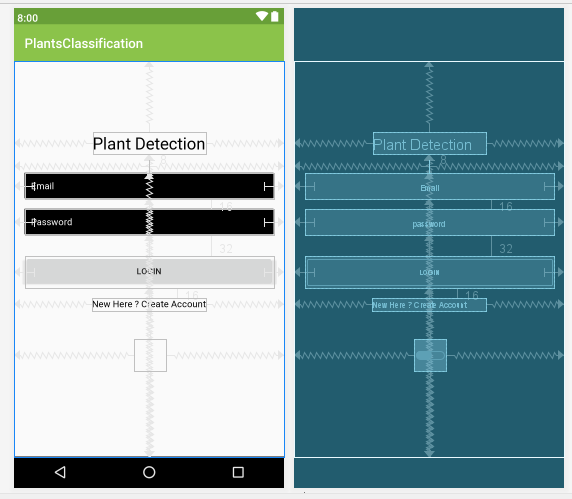
4.Image identification

By using image classification we can identify to which species the plant belongs to,it gives the probability to which species it belongs to

**3.2UI prototypes**



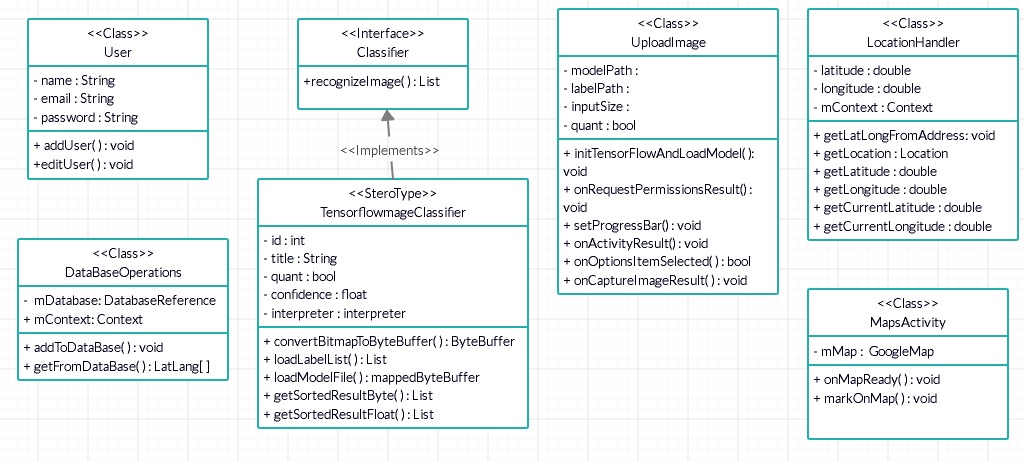




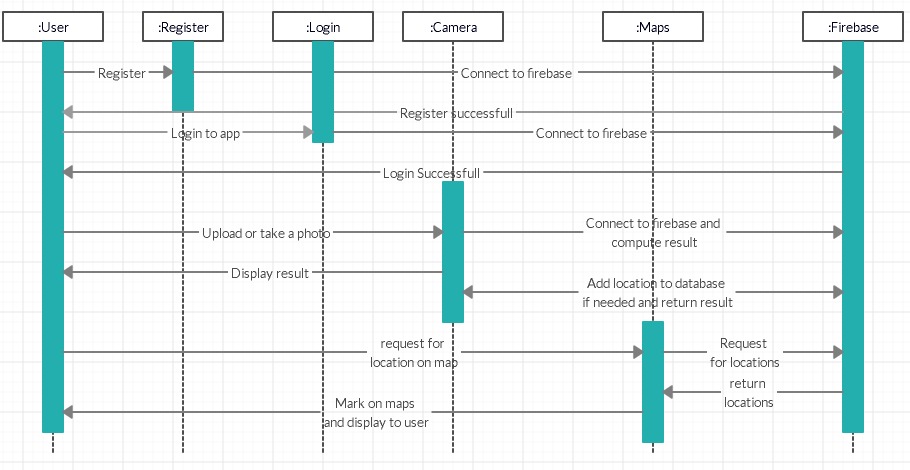
**3. 3. Architecture and Technology Used**

* Android Studio
* ADB Emulator
* Google Map API
* TensorFlow version 1.41
* Firebase (Real Time Database & Authentication)
* Languages used : Java and python

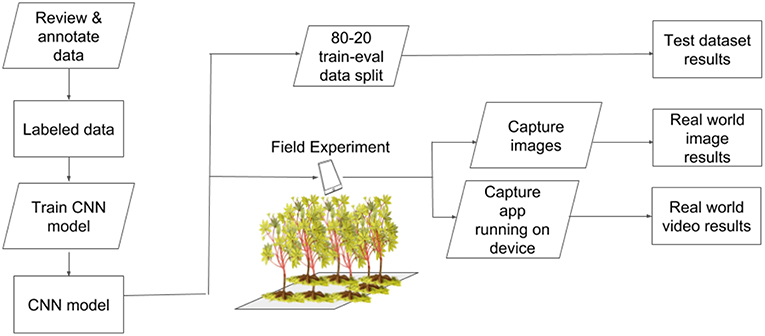
**4.Design**

 **4.1.UML static diagram(Class diagram)**

**4.2.UML Runtime diagrams**

 **1.Sequence Diagram**

**2.Flowchart**

****

**3.5.Implementation**

**3.5.1.Description of Main Class**

**Login.java**

package com.example.plantsclassification;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class Login extends AppCompatActivity {  
  
 EditText mEmail,mPassword;  
 Button mLoginBtn;  
 TextView mCreateBtn;  
 ProgressBar progressBar;  
 FirebaseAuth fAuth;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
  
 mEmail = findViewById(R.id.*Email*);  
 mPassword = findViewById(R.id.*password*);  
 progressBar = findViewById(R.id.*progressBar*);  
 fAuth = FirebaseAuth.*getInstance*();  
 mLoginBtn = findViewById(R.id.*loginBtn*);  
 mCreateBtn = findViewById(R.id.*createText*);  
  
 mLoginBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 String email = mEmail.getText().toString().trim();  
 String password = mPassword.getText().toString().trim();  
  
 if (TextUtils.*isEmpty*(email)) {  
 mEmail.setError("Email is Required.");  
 return;  
 }  
  
 if (TextUtils.*isEmpty*(password)) {  
 mPassword.setError("Password is Required.");  
 return;  
 }  
  
 if (password.length() < 6) {  
 mPassword.setError("Password Must be >= 6 Characters");  
 return;  
 }  
  
 progressBar.setVisibility(View.*VISIBLE*);  
 fAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 if (task.isSuccessful()) {  
 //Toast.makeText(Login.this, "Logged in Successfully", Toast.LENGTH\_SHORT).show();  
 //startActivity(new Intent(Login.this, UploadImage.class));  
 Intent intent = new Intent(Login.this,UploadImage.class);  
 startActivity(intent);  
 } else {  
 Toast.*makeText*(Login.this, "Error ! " + task.getException().getMessage(), Toast.*LENGTH\_SHORT*).show();  
 progressBar.setVisibility(View.*GONE*);  
 }  
 }  
 });  
 }  
 });  
  
 mCreateBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent=new Intent(Login.this,Register.class);  
 startActivity(intent);  
 //Toast.makeText(Login.this,"Register",Toast.LENGTH\_SHORT).show();  
 }  
 });  
  
 }  
}

This Class handles the user login where the user login with his email id and password if he is not registered then he will get an error saying that you are not registered so first register yourself to proceed. If the user enters correct email id and password then he will be taken to the plant identification page where he can identify plants.

**Register.java**

package com.example.plantsclassification;  
  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class Register extends AppCompatActivity {  
  
  
 EditText mFullName,mEmail,mPassword;  
 Button mRegisterBtn;  
 TextView mLoginBtn;  
 FirebaseAuth fAuth;  
 ProgressBar progressBar;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_register*);  
  
 mFullName = findViewById(R.id.*RegfullName*);  
 mEmail = findViewById(R.id.*RegEmail*);  
 mPassword = findViewById(R.id.*Regpassword*);  
 mRegisterBtn = findViewById(R.id.*registerBtn*);  
 mLoginBtn = findViewById(R.id.*createText1*);  
 progressBar = findViewById(R.id.*progressBar1*);  
 fAuth = FirebaseAuth.*getInstance*();  
  
 mRegisterBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String email = mEmail.getText().toString().trim();  
 String password = mPassword.getText().toString().trim();  
  
 if(TextUtils.*isEmpty*(email)){  
 mEmail.setError("Email is Required.");  
  
 }  
  
  
 if(TextUtils.*isEmpty*(password)){  
 mPassword.setError("Password is Required.");  
  
 }  
  
 if(password.length() < 6){  
 mPassword.setError("Password Must be >= 6 Characters");  
 }  
  
 progressBar.setVisibility(View.*VISIBLE*);  
  
 fAuth.createUserWithEmailAndPassword(email,password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
  
 if(task.isSuccessful())  
 {  
 Toast.*makeText*(Register.this,"Registration Successfull",Toast.*LENGTH\_SHORT*).show();  
 Intent intent = new Intent(Register.this,Login.class);  
 startActivity(intent);  
 }  
 else {  
 Toast.*makeText*(Register.this,"Registration UnSuccessfull",Toast.*LENGTH\_SHORT*).show();  
 }  
  
 }  
 });  
  
  
 }  
 });  
  
 mLoginBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(Register.this,Login.class);  
 startActivity(intent);  
 }  
 });  
  
 }  
  
}

In this class when the user clicks on new user sign up then he will be redirected to the register page where he has to enter his name , email id and password after that he will be registered and he will get a response saying you have successfully registered.

**UploadImage.java**

public class UploadImage extends AppCompatActivity {  
  
 private static final String *MODEL\_PATH* = "model.tflite";  
 private static final boolean *QUANT* = true;  
 private static final String *LABEL\_PATH* = "labels.txt";  
 private static final int *INPUT\_SIZE* = 224;  
  
 private Classifier classifier;  
 private Executor executor = Executors.*newSingleThreadExecutor*();  
 TextView tv;  
 ImageView profileImageView;  
 FloatingActionButton pickImage;  
 private static final int *PICK\_IMAGE* = 1;  
 private static final int *CAPTURE\_PHOTO* = 2;  
 private ProgressDialog progressBar;  
 private int progressBarStatus = 0;  
 private Handler progressBarHandler = new Handler();  
 Bitmap thumbnail;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_upload\_image*);  
  
 profileImageView = findViewById(R.id.*profile*);  
 pickImage = findViewById(R.id.*pick*);  
 tv = findViewById(R.id.*tv*);  
  
 pickImage.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 switch (v.getId()) {  
 case R.id.*pick*:  
 new AlertDialog.Builder(UploadImage.this)  
 .setTitle(R.string.*action\_pic*)  
 .setItems(R.array.*uploadImages*, new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
 switch (which) {  
 case 0:  
 Intent photopicker = new Intent(Intent.*ACTION\_PICK*);  
 photopicker.setType("image/\*");  
 startActivityForResult(photopicker, *PICK\_IMAGE*);  
 break;  
  
 case 1:  
 Intent intent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 startActivityForResult(intent, *CAPTURE\_PHOTO*);  
 break;  
 case 2:  
 profileImageView.setImageResource(R.drawable.*ic\_account\_box\_black\_24dp*);  
 break;  
 }  
 }  
 }).show();  
  
 }}  
  
 });  
  
 if (ContextCompat.*checkSelfPermission*(UploadImage.this, Manifest.permission.*CAMERA*) != PackageManager.*PERMISSION\_GRANTED*) {  
 profileImageView.setEnabled(false);  
 ActivityCompat.*requestPermissions*(UploadImage.this, new String[]{Manifest.permission.*CAMERA*, Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*}, 0);  
 } else {  
 profileImageView.setEnabled(true);  
 }  
 initTensorFlowAndLoadModel();  
  
  
 }  
 private void initTensorFlowAndLoadModel() {  
 executor.execute(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 classifier = TensorFlowImageClassifier.*create*(  
 getAssets(),  
 *MODEL\_PATH*,  
 *LABEL\_PATH*,  
 *INPUT\_SIZE*,  
 *QUANT*);  
 } catch (final Exception e) {  
 throw new RuntimeException("Error initializing TensorFlow!", e);  
 }  
 }  
 });  
 }  
 public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {  
 if (requestCode == 0) {  
 if (grantResults.length > 0 && grantResults[0] == PackageManager.*PERMISSION\_GRANTED* && grantResults[1] == PackageManager.*PERMISSION\_GRANTED*) {  
 profileImageView.setEnabled(true);  
 }  
 }  
 }

In this class the user can either upload image or capture image or remove image he has captured .So in this class we are using alert builder which gives the user with choices like pick an image , capture image or upload image based on the option that user selected the action will be taken . If the camera permission is not given the app will not be able to capture the image so it will ask for the permission to allow the app to access the camera .After taking the picture it will initialise the tensorflow model for classification.

**Classifier.java**

public interface Classifier {  
  
 class Recognition {  
 */\*\*  
 \* A unique identifier for what has been recognized. Specific to the class, not the instance of  
 \* the object.  
 \*/* private final String id;  
  
 */\*\*  
 \* Display name for the recognition.  
 \*/* private final String title;  
  
 */\*\*  
 \* Whether or not the model features quantized or float weights.  
 \*/* private final boolean quant;  
  
 */\*\*  
 \* A sortable score for how good the recognition is relative to others. Higher should be better.  
 \*/* private final Float confidence;  
  
 public Recognition(  
 final String id, final String title, final Float confidence, final boolean quant) {  
 this.id = id;  
 this.title = title;  
 this.confidence = confidence;  
 this.quant = quant;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public Float getConfidence() {  
 return confidence;  
 }  
  
 @Override  
 public String toString() {  
 String resultString = "";  
 if (id != null) {  
 resultString += "[" + id + "] ";  
 }  
  
 if (title != null) {  
 resultString += title + " ";  
 }  
  
 if (confidence != null) {  
 resultString += String.*format*("(%.1f%%) ", confidence \* 100.0f);  
 }  
  
 return resultString.trim();  
 }  
 }  
  
  
 List<Recognition> recognizeImage(Bitmap bitmap);  
  
 void close();  
  
  
  
}

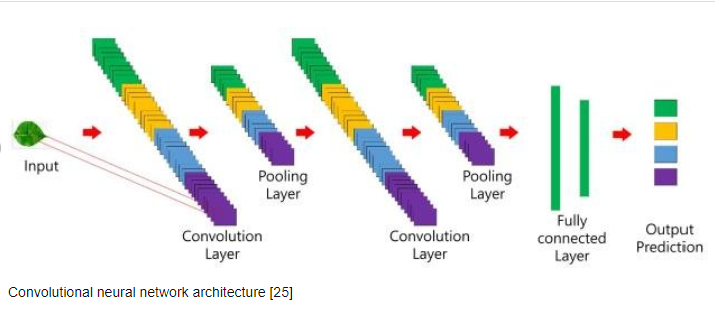
In this class recognize in order to identify the plant we have id which is the unique identifier , its title that is its name which will be taken from labels.txt file and confidence is the probability after the image is predicted ,we will sort the confidence values the higher confidence will be selected and quant is used to check whether the model features are quantized or float weights.

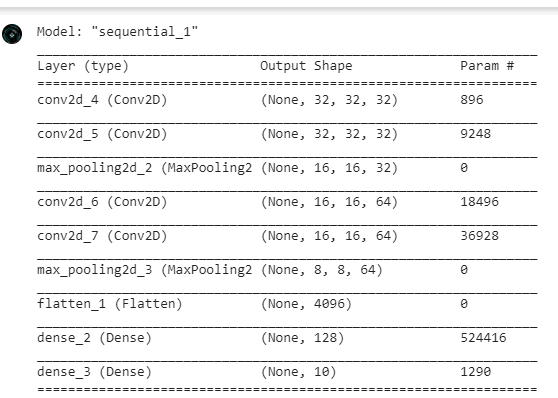
**LocationHandler.java**

public class LocationHandler extends Service implements LocationListener {  
 private static final Location TODO = null;  
  
 private final Context mContext;  
  
  
  
 @Override  
 public void onLocationChanged(Location loc) {  
 }  
  
  
 @Override  
 public void onProviderDisabled(String provider) {  
 }  
  
  
 @Override  
 public void onProviderEnabled(String provider) {  
 }  
  
 boolean checkGPS = false;  
  
  
 boolean checkNetwork = false;  
  
 boolean canGetLocation = false;  
  
 Location loc;  
 double latitude;  
 double longitude;  
  
  
 private static final long MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES = 10;  
  
  
 private static final long MIN\_TIME\_BW\_UPDATES = 1000 \* 60 \* 1;  
 protected LocationManager locationManager;  
  
 @RequiresApi(api = Build.VERSION\_CODES.M)  
 public LocationHandler(Context mContext) {  
 this.mContext = mContext;  
 getLocation();  
 }  
  
 @RequiresApi(api = Build.VERSION\_CODES.M)  
 public Location getLocation() {  
  
 try {  
 locationManager = (LocationManager) mContext.getSystemService(Context.LOCATION\_SERVICE);  
  
 // get GPS status  
 checkGPS = locationManager  
 .isProviderEnabled(LocationManager.GPS\_PROVIDER);  
  
 // get network provider status  
 checkNetwork = locationManager.isProviderEnabled(LocationManager.NETWORK\_PROVIDER);  
  
 if (!checkGPS && !checkNetwork) {  
 Toast.makeText(mContext, "Please Turn on your location", Toast.LENGTH\_SHORT).show();  
 } else {  
 this.canGetLocation = true;  
  
 // if GPS Enabled get lat/long using GPS Services  
 if (checkGPS) {  
  
 if (ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {  
 // TODO: Consider calling  
 // ActivityCompat#requestPermissions  
 // here to request the missing permissions, and then overriding  
 // public void onRequestPermissionsResult(int requestCode, String[] permissions,  
 // int[] grantResults)  
 // to handle the case where the user grants the permission. See the documentation  
 // for ActivityCompat#requestPermissions for more details.  
 }  
 locationManager.requestLocationUpdates(  
 LocationManager.GPS\_PROVIDER,  
 MIN\_TIME\_BW\_UPDATES,  
 MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES, this);  
 if (locationManager != null) {  
 loc = locationManager  
 .getLastKnownLocation(LocationManager.GPS\_PROVIDER);  
 if (loc != null) {  
 latitude = loc.getLatitude();  
 longitude = loc.getLongitude();  
 }  
 }  
  
  
 }  
  
  
 if (checkNetwork) {  
  
  
 if (ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {  
 // TODO: Consider calling  
 // ActivityCompat#requestPermissions  
 // here to request the missing permissions, and then overriding  
 // public void onRequestPermissionsResult(int requestCode, String[] permissions,  
 // int[] grantResults)  
 // to handle the case where the user grants the permission. See the documentation  
 // for ActivityCompat#requestPermissions for more details.  
 }  
 locationManager.requestLocationUpdates(  
 LocationManager.NETWORK\_PROVIDER,  
 MIN\_TIME\_BW\_UPDATES,  
 MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES, this);  
  
 if (locationManager != null) {  
 loc = locationManager  
 .getLastKnownLocation(LocationManager.NETWORK\_PROVIDER);  
  
 }  
  
 if (loc != null) {  
 latitude = loc.getLatitude();  
 longitude = loc.getLongitude();  
 }  
 }  
  
 }  
  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 return loc;  
 }  
  
 public double getCurrentLongitude() {  
 if (loc != null) {  
 longitude = loc.getLongitude();  
 }  
 //Toast.makeText(mContext, "returnig lat "+longitude, Toast.LENGTH\_SHORT).show();  
 return longitude;  
 }  
  
 public double getCurrentLatitude() {  
 if (loc != null) {  
 latitude = loc.getLatitude();  
 }  
 //Toast.makeText(mContext, "returnig lat "+latitude, Toast.LENGTH\_SHORT).show();  
 return latitude;  
 }  
 public double getLatitude() {  
 return latitude;  
 }  
  
 public double getLongitude() {  
 return longitude;  
 }  
 public boolean canGetLocation() {  
 return this.canGetLocation;  
 }  
  
 public void showSettingsAlert() {  
 AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);  
  
  
 alertDialog.setTitle("GPS is not Enabled!");  
  
 alertDialog.setMessage("Do you want to turn on GPS?");  
  
  
 alertDialog.setPositiveButton("Yes", new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int which) {  
 Intent intent = new Intent(Settings.ACTION\_LOCATION\_SOURCE\_SETTINGS);  
 mContext.startActivity(intent);  
 }  
 });  
  
  
 alertDialog.setNegativeButton("No", new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int which) {  
 dialog.cancel();  
 }  
 });  
  
  
 alertDialog.show();  
 }  
  
  
 public void stopListener() {  
 if (locationManager != null) {  
  
 if (ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {  
 // TODO: Consider calling  
 // ActivityCompat#requestPermissions  
 // here to request the missing permissions, and then overriding  
 // public void onRequestPermissionsResult(int requestCode, String[] permissions,  
 // int[] grantResults)  
 // to handle the case where the user grants the permission. See the documentation  
 // for ActivityCompat#requestPermissions for more details.  
 return;  
 }  
 locationManager.removeUpdates(LocationHandler.this);  
 }  
 }  
  
 @Override  
 public void onStatusChanged(String provider, int status, Bundle extras) {  
 }  
  
  
 public void getLatLongFromAddress(final String locationAddress, final Context context) {  
 Geocoder geocoder = new Geocoder(context, Locale.getDefault());  
  
 try {  
 List addressList = geocoder.getFromLocationName(locationAddress, 1);  
 if (addressList != null && addressList.size() > 0) {  
 Address address = (Address) addressList.get(0);  
 this.latitude = address.getLatitude();  
 this.longitude = address.getLongitude();  
 }  
 } catch (IOException e) {  
 }  
 }  
  
 @Nullable  
 @Override  
 public IBinder onBind(Intent intent) {  
 return null;  
 }  
}

In this class we will be able to know the location of the plant by using google map api,so first before viewing it will check whether there is internet connection or not then it will check permissions to use gps location and then it will view the plants location by viewing red color.In this class it has latitude and longitude positions to view in map so there are setting and getter methods to get the latitudes and longitudes position.

**3.5.2.Specific Algorithms:**

* Convulation Layer : Convolution operation extracts different features using learned filters
* Pooling Layer: Subsamples the output of a convulational layer.
* ****Fully Connected layer:Takes input from all neurons in the previous layer and performs operation with individual neuron in the current layer to generate output

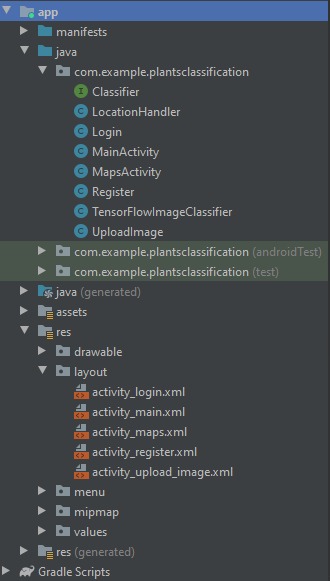
****

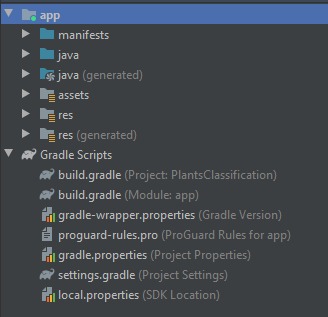
* This is the summary of the model we have used

**3.5.3. Github links and folder structure :**

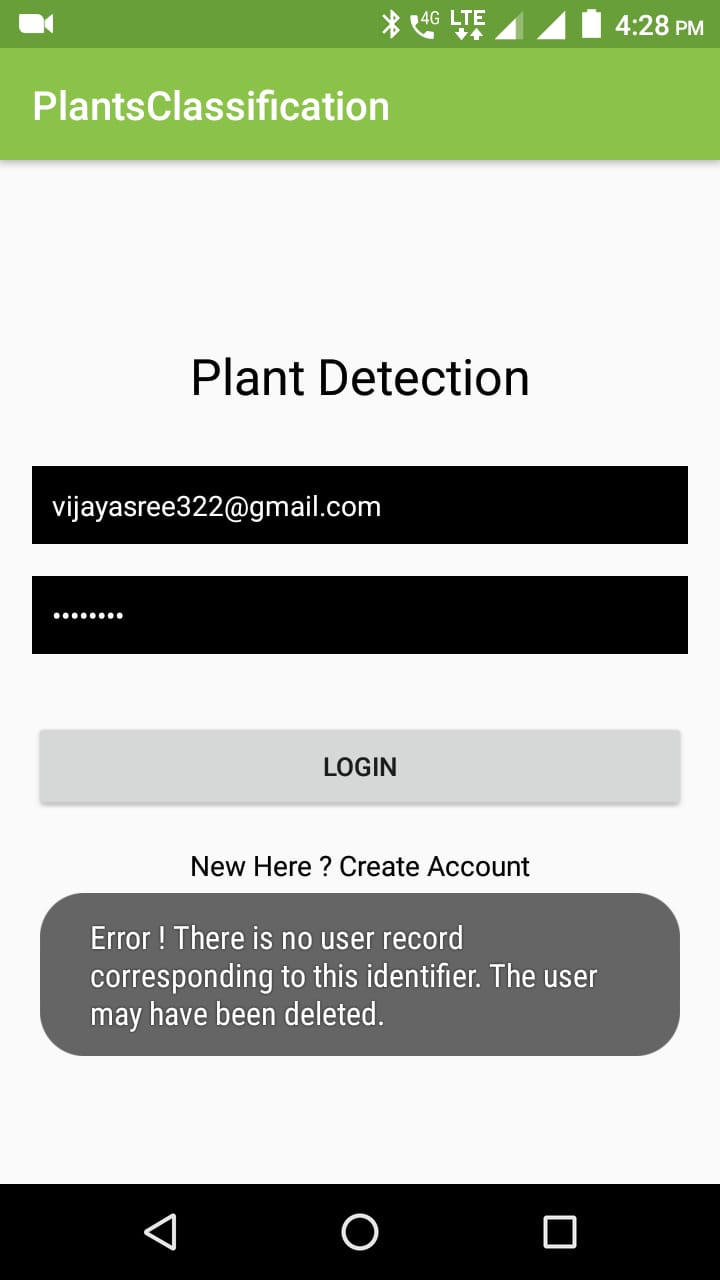
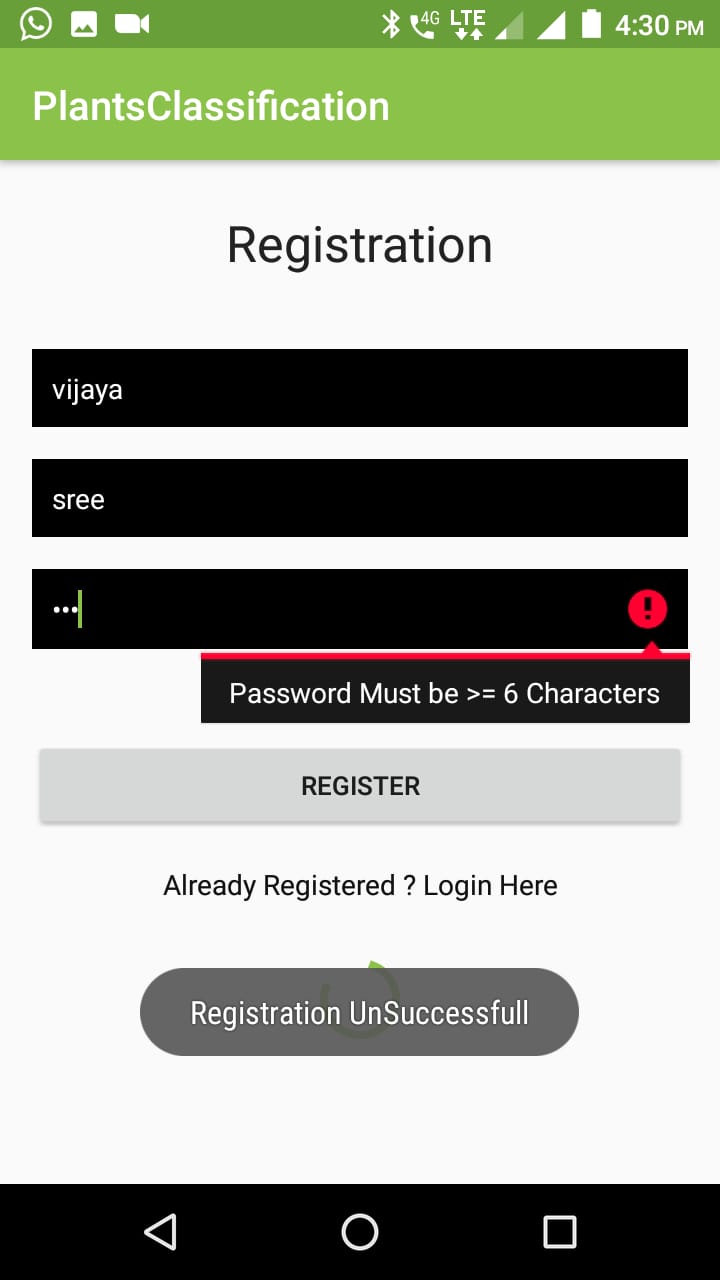
**Github Link:**

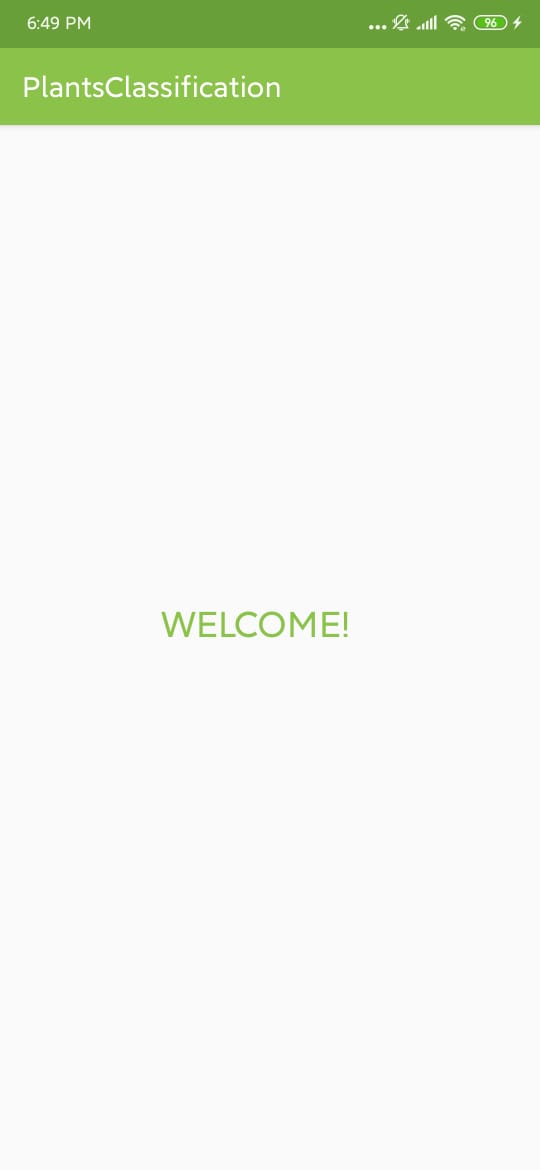
<https://github.com/manojdvs7/ThemeBasedProject>

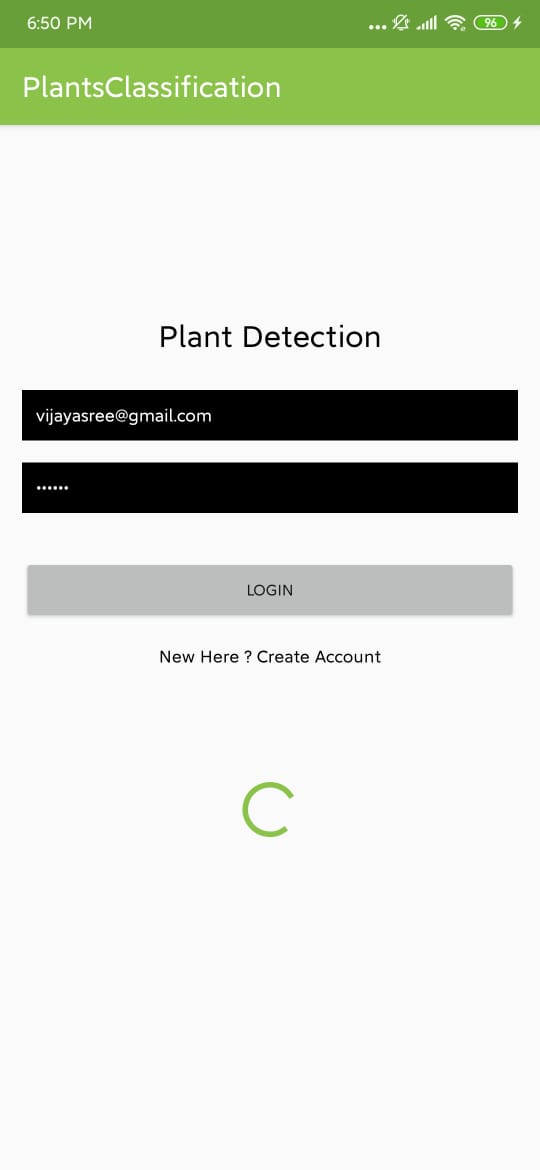
**Folder Structure**

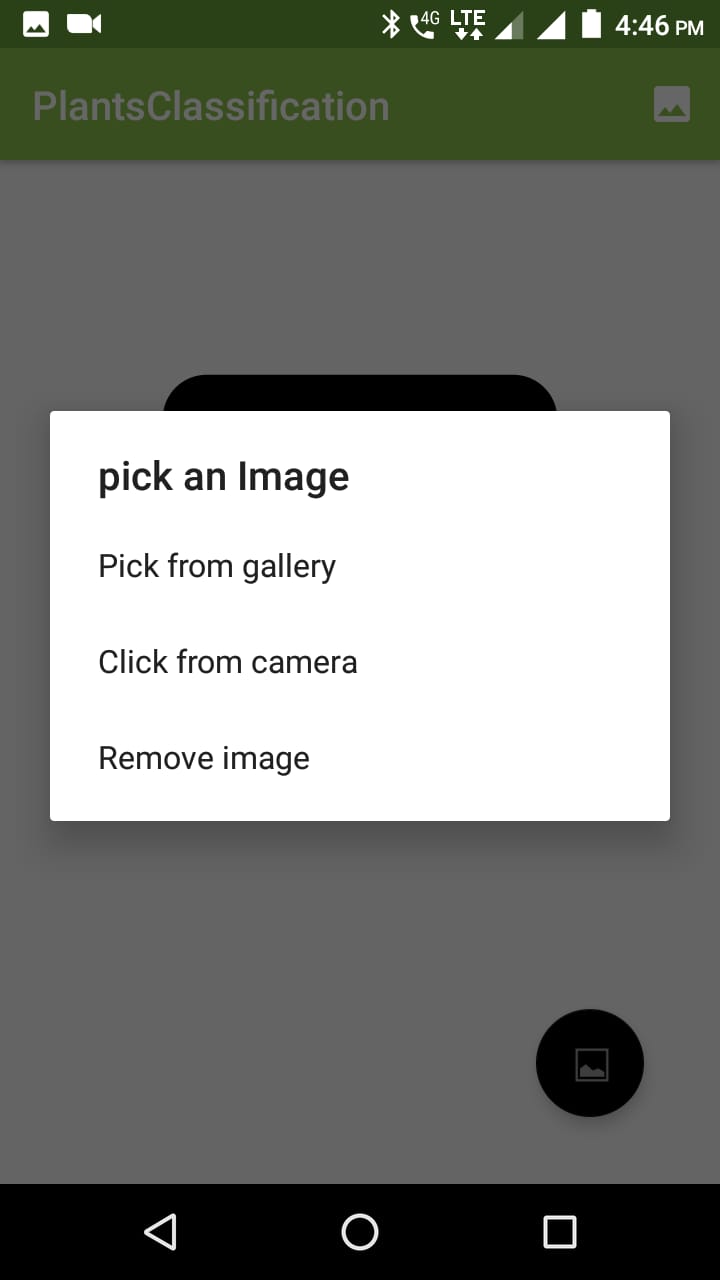
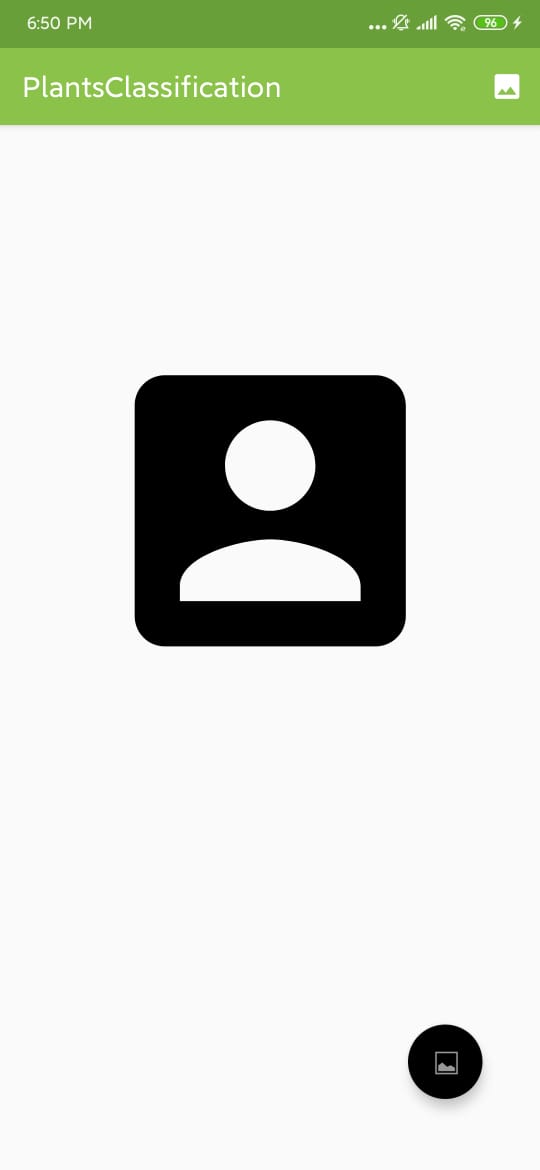


**3.6.Testing**

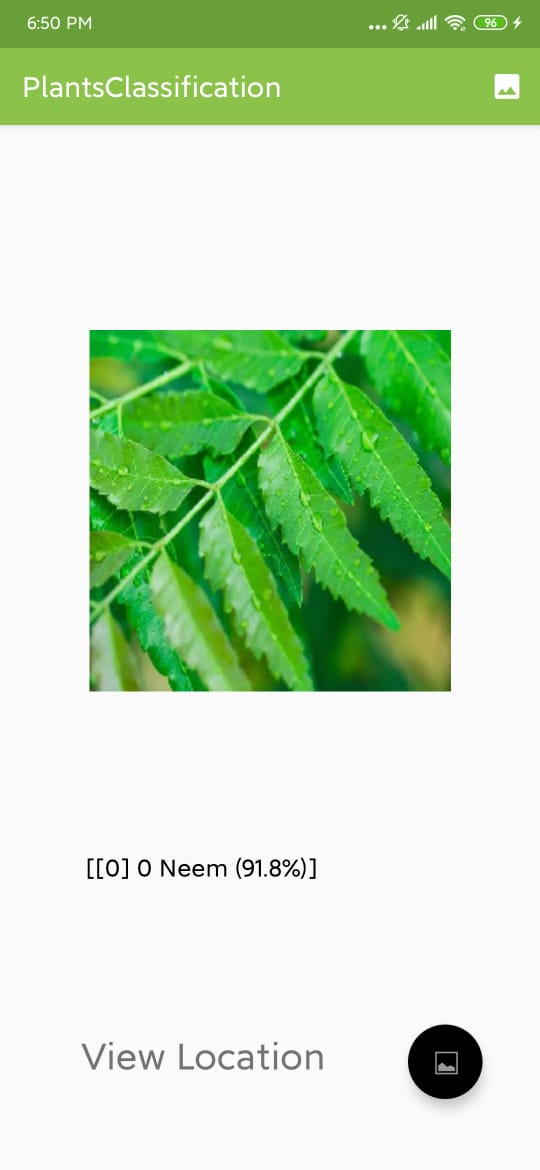
**1.**If user logs in with wrong password 2.If user enters less than 6 characters for password

 **4.RESULTS**











**5.Discussion and Future Works**

* The ability to filter recognized species by genus or family.
* Search facility for the location of plant.

**6.References**

**Firebase Authentication**

<https://firebase.google.com/docs/auth/android/firebaseui>

**Firebase Real time Database**

<https://firebase.google.com/docs/database/android/start>

**Udemy Course: A complete guide on tensorFlow**

<https://www.udemy.com/course/tensorflow-2/>

**Error queries**

<https://stackoverflow.com/questions>

<https://www.codementor.io/collections/learn-android-development-online-bwba0mlle>