

ORPHANAGE APP

REPORT OF MAJOR PROJECT SUBMITTED FOR PARTIAL FULFILLMENT OF
THE REQUIREMENT FOR THE DEGREE OF MASTER IN COMPUTER
APPLICATION

ABARA KARMAKAR

REG NO:151170510001

ROLL NO: 11701015001

BHASKAR CHAKRABORTY

Reg No: 151170510012

Roll No: 11701015012

TRISHA GHOSH

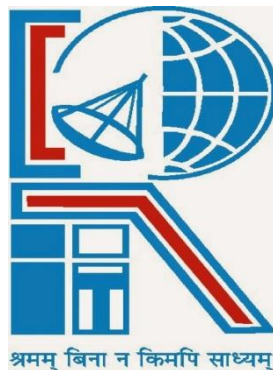
Reg No: 151170510057

Roll No: 11701015056

UNDER THE SUPERVISION OF

MRS SATABDWI SARKAR

Asst. Professor



RCC Institute Of Information Technology

AT

RCC INSTITUTE OF INFORMATION TECHNOLOGY

Affiliated to Maulana Abul Kalam Azad University Of Technology

CANAL SOUTH ROAD, BELIAGHATA, KOLKATA-700015, May 2018



INTRODUCTION

An orphanage is a residential institution devoted to the care of orphans—children whose biological parents are deceased or otherwise unable or unwilling to take care of them. The problem of orphan is acute due to urbanization and industrialization.

The orphanage is also referred to as NGO i: e “NON GOVERNMENTAL ORGANIZATION” .

The orphanage application provides a platform to the NGOs working for the needy people, as well as the kind hearted people having concern for humanity, trying to help the society by their donations in any form both in cash or kind.

When an NGO needs some fund to rise to help the people from a selected location, all they need is the proper donation. What we provide is a room for an NGO/orphanage and a donor to shake hands under a roof.

From this application a child can be sponsored. They can donate books, other stationary items, miscellaneous items like clothing, blankets, bed sheets, shoes etc.

They can donate any amount of cash in online or offline. The bank details are available. Our volunteers are available, anyone can contact with us.

• **OBJECTIVE**

'Little Homes', an NGO management system, is a platform that provides hands to the needy children. We seek a world of hope, love, tolerance, social justice and peace where poverty has been overcome and children lives in dignity and with security.

The vision and mission of 'Little Homes' guides us towards our goal of overcoming poverty, and ensuring a life of dignity and security. We focus on children those who lost their parents to realize their rights, avail opportunities and build a better future for them.

It provides a platform to the people working for the needy people, as well as the people who are trying to help the society by their donations.

When orphanage needs some fund to help the orphan children, all they need is the proper donation. What we provide is a room for orphanage and a donor to shake hands under a roof.

The end users are both the orphanage and a donor. The purpose of this project is:-

- ✓ **Helping needy children to get great support and affection from the people around them.**
- ✓ **To maintain social, ethical and organizational norms.**
- ✓ **Building awareness about the needy people around the world.**

Admin Module:The administrator is the ultimate controller of the application with the highest authority. He/she has the following features:-

1. He/she can organize any event for the orphanage.
2. He/she can see the details of the donor.

User Module: It enables a person to login or sign up if he/she has registered for the Orphanage app. Once logged it has the following features: -

1. he/she can donate anything.
2. He/she can see the details of the orphanage.
3. He/she can access the GPS.
4. He/she can call the volunteers of that nearby orphanage organization.

● **FEASIBILITY STUDY**

A Feasibility study aims to judge the cost required and the value to be attained. This system has been enlightening the TELOS principle of feasibility study.

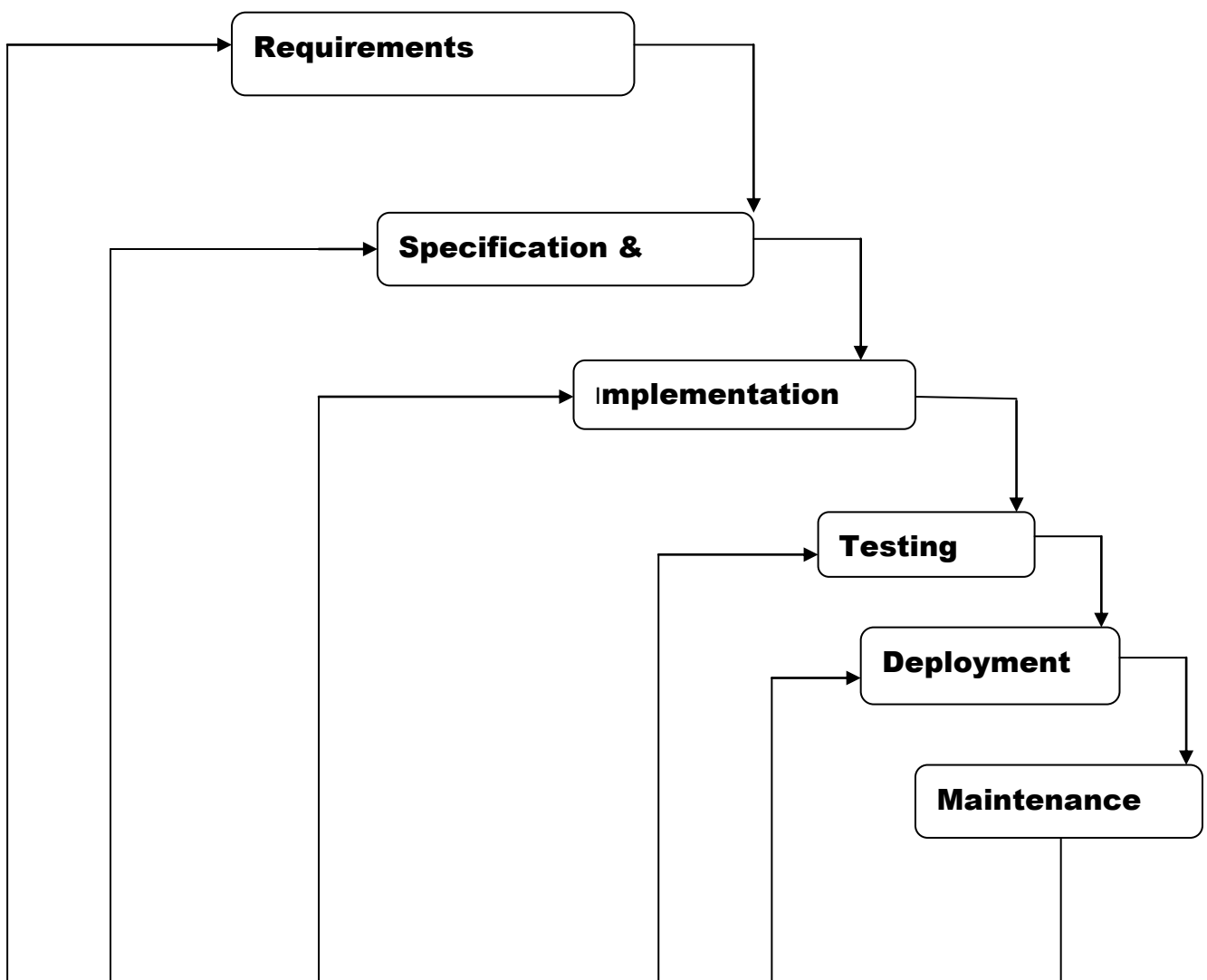
TECHNICAL FEASIBILITY

Before starting that project we thought about the technology we need to use. So we decide to use android studio because with it we could design an effective and beautiful application. Not only that, it takes less time to develop.

SCHEDULE FEASIBILITY

A project will fail if it takes too long to respond. So this is important.

● **SOFTWARE AND HARDWARE REQUIREMENT**

AVAILABLE TECHNOLOGIES:**Database: Google Firebase.****Development Platform: Android Studio.****TOOLS USED:****Editor Used: Android Studio with Java, Google Firebase.****Operating System: Windows 10.****HARDWARE USED:****Processor: Intel Core i5****RAM: 4GB****Hard Disk: 500 GB****SOFTWARE REQUIREMENT SPECIFICATION**

Software Requirement Specification

SRS is a document created by system analyst after the requirements are collected from various stakeholders.

SRS defines how the intended software will interact with hardware, external interfaces, speed of operation, and response time of system, portability of software across various platforms, maintainability, speed of recovery after crashing, security, quality, limitations etc.

The requirement received from client is written in natural language. It is the responsibility of system analyst to document the requirements in technical language so that they can be comprehended and useful by the software development team.

SRS should come up with following features:

User requirement are expressed in natural language.

Technical requirement are expressed in structured language, which is used inside the organization.

- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as unit testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaw or errors. Testing is done so that the client doesn't face any problem during the installation of the software.

- **Deployment of system: Once the functional and non- functional testing is done, the product is deployed in the customer environment or released into the market.**
- **Maintenance: This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. This modification arises either due to change requests initiated by the customer, or defects uncovered during live use of the system. Client is provided with regular maintenance and support for the developed software.**

REVIEW OF LITERATURE:

FIREBASE:

Firestore is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. Firestore evolved from Envolv, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolv provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that weren't chat messages. Developers were using Envolv to sync application data such as game state in real time across their users.

Firestore Auth

Firestore Auth is a service that can authenticate users using only client-side code. It supports social login providers Facebook, GitHub, Twitter and Google (and Google Play Games). Additionally, it includes a user management system whereby developers can enable user authentication with email and password login stored with Firestore.

Firestore Database

Firestore provides a realtime database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firestore's cloud. The company provides client libraries that enable integration with Android, iOS, JavaScript, Java, Objective-C, swift and Node.js applications. The database is also accessible through a REST API and bindings for several JavaScript frameworks such as [AngularJS](#), [React](#), [Ember.js](#) and [Backbone.js](#). The REST API uses the Server-Sent Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the real-time database can secure their data by using the company's server-side-enforced security rules.

Firestore Storage

Firestore Storage provides secure file uploads and downloads for Firestore apps, regardless of network quality. The developer can use it to store images, audio, video, or other user-generated content. Firestore Storage is backed by Google Cloud Storage.

ANDROID STUDIO:

Android Studio is the official [integrated development environment](#) (IDE) for [Google's Android operating system](#), built on [JetBrains' IntelliJ IDEA](#) software and designed specifically for [Android development](#). Android Studio was announced on May 16, 2013 at the [Google I/O](#) conference. It was in early access preview stage starting from version 0.1 in May 2013. The current stable version is 3.1 released in March 2018.

Features

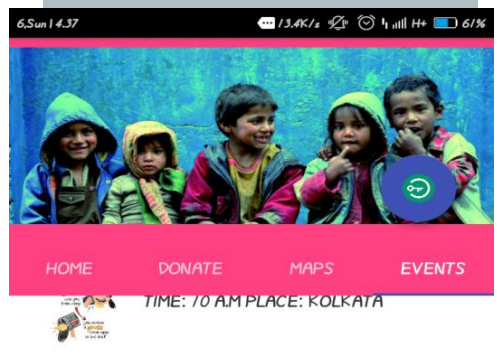
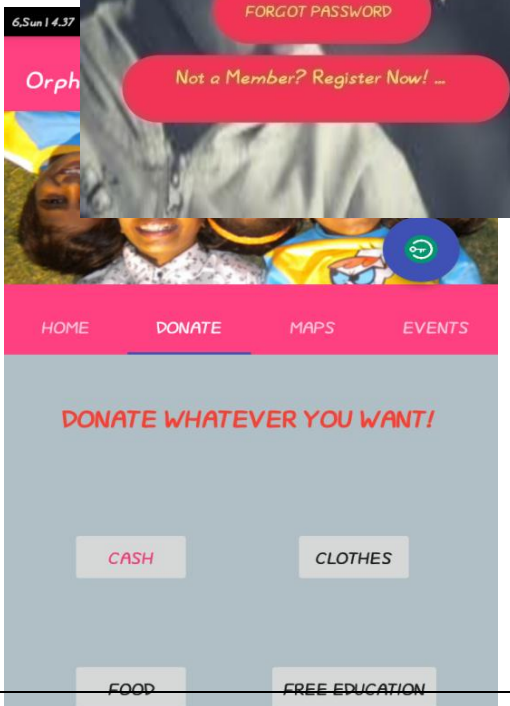
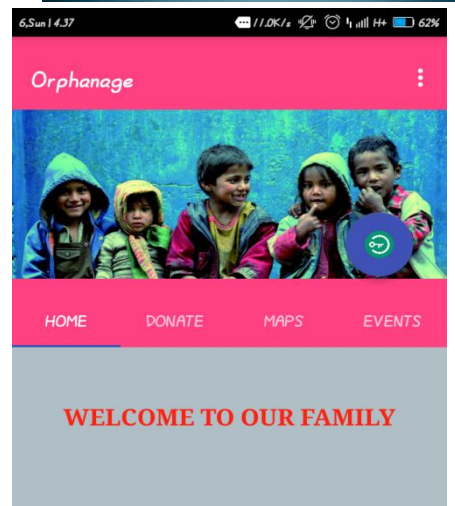
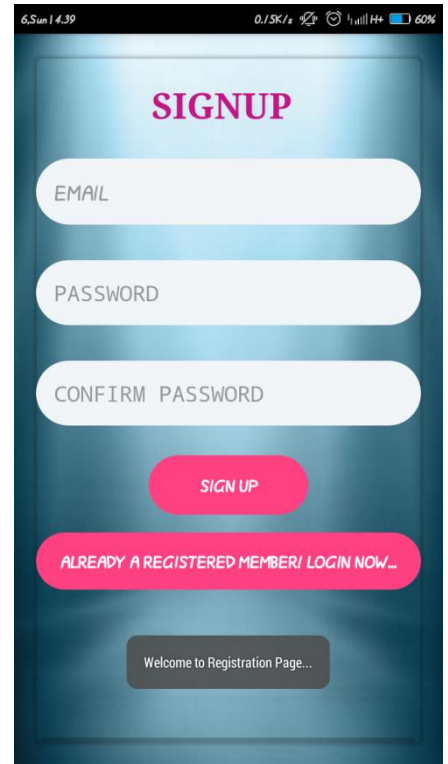
The following features are provided in the current stable version:

- [Gradle](#)-based build support
- Android-specific [refactoring](#) and quick fixes
- [Lint](#) tools to catch performance, usability, version compatibility and other problems
- [ProGuard](#) integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich [layout editor](#) that allows users to drag-and-drop UI components, option to [preview layouts](#) on multiple screen configurations^[6]
- Support for building [Android Wear](#) apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine^[7]
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

XML:

- XML stands for eXtensibleMarkup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive
- XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data.
- XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.
- With XML, data can be available to all kinds of "reading machines" like people, computers, voice machines, news feeds, etc.

SCREENSHOTS



Login Pa



Events

6, Sun 14:38 71.9K/s 71.9K/s 61%

YOUR NAME

YOUR EMAIL

SUBJECT

MESSAGE

Enter your message

SEND MESSAGE

Contact Us

6, Sun 14:38 0.19K/s 61%

Vision

Our vision is to uplift the standard and condition of life for orphans and vulnerable children and to empower young people to have a meaningful life and a better tomorrow.

Mission

Our mission is to work with partners and volunteers to improve the lives of Orphans, Vulnerable Children and young people through a right-based approach.

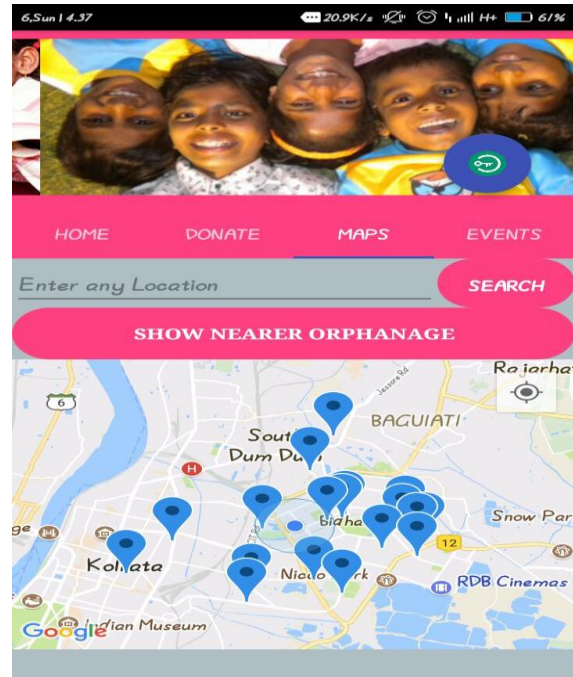
Aims and Objectives

- * The aim of the organization is to prevent children from roaming on the streets, to assist children who are orphaned due to HIV/AIDS, to achieve their goal of education and sustainable family life.*
- * To uplift the quality of life of Orphans and Vulnerable Children (OVC) by creating a home environment.*
- * To empower OVC's of school going age by providing nutrition and educational and skill training.*
- * To protect and promote the wellbeing of Orphans and Vulnerable Children by ensuring that OVC's rights and that of caregivers are protected, respected and full filled*

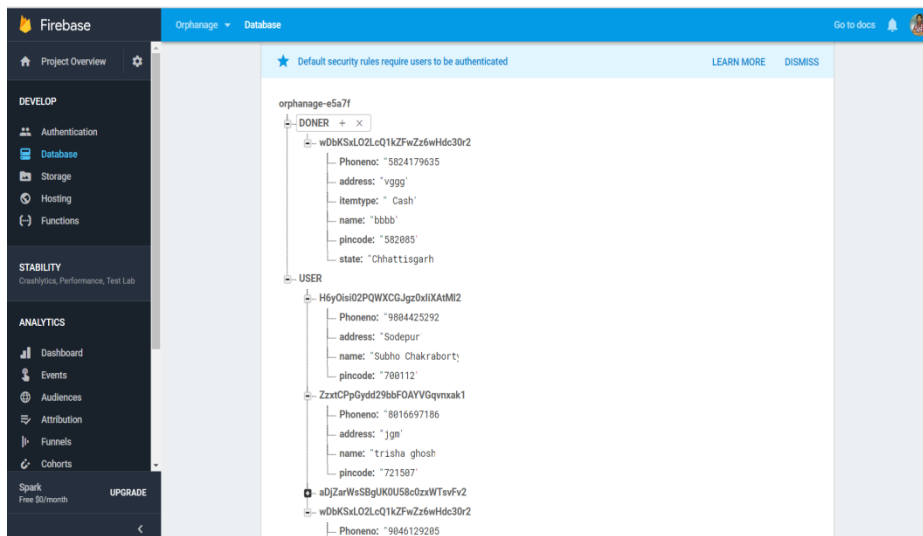
Our Promise

About Us

DONATION FORM



MAP FOR SHOWING NEARER



DATABASE TABLE

SAMPLE CODE:

LOGIN PAGE:


```

Toast.makeText(getApplicationContext(),"Error!",Toast.LENGTH_LONG).show();
        }
    }
});
}
});
}

```

SIGNUP PAGE:

```
packagecom.example.bhaskar.orphanage;
```

```

importandroid.app.ProgressDialog;
importandroid.content.Intent;
importandroid.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.Toast;
importandroid.widget.EditText;

```

```

importcom.google.android.gms.tasks.OnCompleteListener;
importcom.google.android.gms.tasks.Task;
importcom.google.firebase.auth.AuthResult;
importcom.google.firebase.auth.FirebaseAuth;
importcom.google.firebase.auth.FirebaseAuthWeakPasswordException;
importcom.google.firebase.auth.FirebaseUser;
importcom.google.firebase.database.DatabaseReference;
importcom.google.firebase.database.FirebaseDatabase;

```

```

importjava.util.regex.Matcher;
importjava.util.regex.Pattern;

```

```
public class Sign_up extends AppCompatActivity {
```

```

    Button submit,login;
    FirebaseAuthfirebaseAuth;
    DatabaseReferencedatabaseReference;
    EditTexteditTextEmail, editTextPassword, editTextConfirmPassword;
    private ProgressDialogprogressDialog;

```

```

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);

```

```
        // Get References of Views
```

```

        editTextEmail = (EditText) findViewById(R.id.editTextemail);
        editTextPassword = (EditText) findViewById(R.id.editTextpassword);

```

```

editTextConfirmPassword = (EditText) findViewById(R.id.editTextconformpassword);

login = (Button) findViewById(R.id.button_signup_login);
login.setSelected(true);
submit = (Button) findViewById(R.id.button_signup_submit);

progressDialog = new ProgressDialog(this);

    //firebase connection
firebaseAuth = FirebaseAuth.getInstance();
    //if the user is already logged in
if(firebaseAuth.getCurrentUser() !=null){
    // menu1 activity here
finish();
startActivity(new Intent(getApplicationContext(),Sign_upForm.class));
}

databaseReference = FirebaseDatabase.getInstance().getReference();

submit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        // TODO Auto-generated method stub
        String email = editTextEmail.getText().toString().trim();
        String password = editTextPassword.getText().toString().trim();
        String confirmPassword = editTextConfirmPassword.getText().toString().trim();

        // check if any of the fields are vaccant
if (email.equals("") || password.equals("") || confirmPassword.equals("")) {
Toast.makeText(getApplicationContext(), "Field Vacant", Toast.LENGTH_LONG).show();
    return;
}
        Boolean onError = false;
if (!isValidEmail(email)) {
onError = true;
editTextEmail.setError("Invalid Email");
    return;
}
        // check if both password matches
if (!password.equals(confirmPassword)) {
Toast.makeText(getApplicationContext(), "Password does not match", Toast.LENGTH_LONG).show();
    return;
}
if (password.length() > 7) {
    // Save the Data in Database
    //////////////////////////////////////
progressDialog.setMessage("Please Wait... ");
progressDialog.show();
firebaseAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {

```

```

        @Override
public void onComplete(@NonNull Task<AuthResult> task) {
    if (task.isSuccessful()) {
        // saveUserInformation();

        finish();
        Toast.makeText(getApplicationContext(), "Account Successfully Created ",
        Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(Sign_up.this, Sign_upForm.class);
        startActivity(intentHome);
    } else {
        finish();
        Toast.makeText(getApplicationContext(), "Account Not Successfully Created ",
        Toast.LENGTH_LONG).show();

    }
    });
}
else {
    Toast.makeText(getApplicationContext(), "Invalid Password No(No Should be Greater Than 7)",
    Toast.LENGTH_LONG).show();
    return;
}

}
});

login.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        progressDialog.setMessage("Please Wait... ");
        progressDialog.show();
        finish();
        //Toast.makeText(getApplicationContext(), "Account Successfully Created ",
        Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(Sign_up.this, Login.class);
        startActivity(intentHome);

    }
});
}

// validating email id
private boolean isValidEmail(String email) {
    String EMAIL_PATTERN = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@" + "[A-Za-z0-9-]+(\\.[A-Za-z0-9-9]+)*"(\\.[A-Za-z]{2,})$";

    Pattern pattern = Pattern.compile(EMAIL_PATTERN);
    Matcher matcher = pattern.matcher(email);

```



```
returnmatcher.matches();  
    }  
  
}
```

MAPS:

```
packagecom.example.bhaskar.orphanage;
```

```
importandroid.content.pm.PackageManager;  
importandroid.location.Address;  
importandroid.location.Geocoder;  
importandroid.location.Location;  
importandroid.os.Build;  
importandroid.os.Bundle;  
importandroid.support.annotation.NonNull;  
importandroid.support.annotation.Nullable;  
import android.support.v4.app.ActivityCompat;  
import android.support.v4.app.Fragment;  
import android.support.v4.content.ContextCompat;  
importandroid.util.Log;  
importandroid.view.LayoutInflater;  
importandroid.view.View;  
importandroid.view.ViewGroup;  
importandroid.widget.Button;  
importandroid.widget.EditText;  
importandroid.widget.Toast;  
  
importcom.google.android.gms.common.ConnectionResult;  
  
importcom.google.android.gms.common.api.GoogleApiClient;  
importcom.google.android.gms.location.LocationListener;  
importcom.google.android.gms.location.LocationRequest;  
importcom.google.android.gms.location.LocationServices;  
importcom.google.android.gms.maps.CameraUpdateFactory;  
importcom.google.android.gms.maps.GoogleMap;  
importcom.google.android.gms.maps.OnMapReadyCallback;  
importcom.google.android.gms.maps.SupportMapFragment;  
importcom.google.android.gms.maps.model.BitmapDescriptorFactory;  
importcom.google.android.gms.maps.model.LatLng;  
importcom.google.android.gms.maps.model.Marker;  
importcom.google.android.gms.maps.model.MarkerOptions;  
  
importjava.io.IOException;  
importjava.util.List;  
  
/**  
 * Created by Bhaskar on 13-03-2018.  
 */
```

```
public class Maps extends Fragment implements OnMapReadyCallback,
GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener,
LocationListener,GoogleMap.OnInfoWindowClickListener,View.OnClickListener {
```

```
/**
 * class MyInfoWindowAdapter implements GoogleMap.InfoWindowAdapter {
 * <p>
 * private final View myContentsView;
 * <p>
 * MyInfoWindowAdapter() {
 * myContentsView = getLayoutInflater().inflate(R.layout.content_info, null);
 * }
 *
 * @Override public View getInfoContents(Marker marker) {
 * TextView tvTitle = ((TextView) myContentsView.findViewById(R.id.title));
 * tvTitle.setText(marker.getTitle());
 * TextView tvSnippet = ((TextView) myContentsView.findViewById(R.id.snippet));
 * tvSnippet.setText(marker.getSnippet());
 * <p>
 * return myContentsView;
 * }
 * @Override public View getInfoWindow(Marker marker) {
 * // TODO Auto-generated method stub
 * return null;
 * }
 * <p>
 * }
 */
```

```
@Override
public void onInfoWindowClick(Marker marker) {
```

```
    /*Toast.makeText(getBaseContext(),
        "Info Window clicked@" + marker.getId(),
        Toast.LENGTH_SHORT).show();*/
    // String markerId=marker.getId();
    //Intent intent=new Intent(MapsActivity.this,Submit_activity.class);
    // Bundle bundle=new Bundle();
    // bundle.putString("Sourav",markerId);
    // intent.putExtras(bundle);
    // startActivity(intent);

}
```

```
private GoogleMap mMap;
private GoogleApiClient client;
private LocationRequest locationRequest;
```

```

private Location lastlocation;
private Marker currentLocationmMarker;
public static final int REQUEST_LOCATION_CODE = 99;
int PROXIMITY_RADIUS = 10000;
double latitude, longitude;
Button search, orphan;

@Nullable
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    View rootView = inflater.inflate(R.layout.maps, container, false);
    search = (Button) rootView.findViewById(R.id.B_Search);
    orphan = (Button) rootView.findViewById(R.id.B_orphanage);
    search.setOnClickListener(this);
    orphan.setOnClickListener(this);

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        checkLocationPermission();
    }
    // Obtain the SupportMapFragment and get notified when the map is ready to be used.
    /** SupportMapFragment mapFragment = (SupportMapFragment)
    getSupportFragmentManager()
    .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);*/

    SupportMapFragment mMapFragment = (SupportMapFragment)
    this.getChildFragmentManager().findFragmentById(R.id.map);
    mMapFragment.getMapAsync(this);

    /** MapFragment mapFragment = (MapFragment)
    getFragmentManager().findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);*/
    return rootView;
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
    @NonNull int[] grantResults) {
    switch(requestCode)
    {
        case REQUEST_LOCATION_CODE:
            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED)
            {
                if (ContextCompat.checkSelfPermission(getActivity(),
                    android.Manifest.permission.ACCESS_FINE_LOCATION) !=
                    PackageManager.PERMISSION_GRANTED)
                {

```

```

if(client == null)
    {
    bulidGoogleApiClient();
    }
mMap.setMyLocationEnabled(true);
    }
    }
else
    {
        ////change kr eta k subhp get context e problem ache
        Toast.makeText(getApplicationContext(),"Permission Denied" , Toast.LENGTH_LONG).show();
    }
    return;
}
}

/**
 * Manipulates the map once available.
 * This callback is triggered when the map is ready to be used.
 * This is where we can add markers or lines, add listeners or move the camera. In this case,
 * we just add a marker near Sydney, Australia.
 * If Google Play services is not installed on the device, the user will be prompted to install
 * it inside the SupportMapFragment. This method will only be triggered once the user has
 * installed Google Play services and returned to the app.
 */
@Override
public void onMapReady(GoogleMapgoogleMap) {
mMap = googleMap;

if (ContextCompat.checkSelfPermission(getActivity(),
android.Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
bulidGoogleApiClient();
mMap.setMyLocationEnabled(true);
    //mMap.setInfoWindowAdapter(new MyInfoWindowAdapter());
mMap.setOnInfoWindowClickListener(this);
    }
}

protected synchronized void bulidGoogleApiClient() {
client = new GoogleApiClient.Builder(getActivity()).
addConnectionCallbacks(this).addOnConnectionFailedListener(this).
addApi(LocationServices.API).build();
client.connect();

}

@Override
public void onLocationChanged(Location location) {

latitude = location.getLatitude();

```

```

longitude = location.getLongitude();
lastlocation = location;
if(currentLocationmMarker != null)
    {
currentLocationmMarker.remove();

    }
Log.d("lat = ", ""+latitude);
LatLnglatLng = new LatLng(location.getLatitude() , location.getLongitude());
MarkerOptionsmarkerOptions = new MarkerOptions();
markerOptions.position(latLng);
markerOptions.title("Current Location");

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_CYAN));
currentLocationmMarker = mMap.addMarker(markerOptions);
mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
mMap.animateCamera(CameraUpdateFactory.zoomBy(10));

if(client != null)
    {
LocationServices.FusedLocationApi.removeLocationUpdates(client,this);
    }
}
public void onClick(View v)
    {
    Object dataTransfer[] = new Object[2];
    GetNearbyPlacesDatagetNearbyPlacesData = new GetNearbyPlacesData();

switch(v.getId())
    {
caseR.id.B_Search:
    EditTexttf_location = (EditText)getActivity().findViewById(R.id.editTFLocation);
    String location = tf_location.getText().toString();
    List<Address>addressList;

if(!location.equals(""))
    {
Geocodergeocoder = new Geocoder(getActivity());

        try {
addressList = geocoder.getFromLocationName(location, 5);

if(addressList != null)
    {
for(int i = 0;i<addressList.size();i++)
    {
LatLnglatLng = new LatLng(addressList.get(i).getLatitude() , addressList.get(i).getLongitude());
MarkerOptionsmarkerOptions = new MarkerOptions();
markerOptions.position(latLng);
markerOptions.title(location);

```

```

mMap.addMarker(markerOptions);
mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
mMap.animateCamera(CameraUpdateFactory.zoomTo(10));
    }
    }
    } catch (IOException e) {
e.printStackTrace();
    }
    }
    break;
    /** case R.id.B_orphanage:
    // mMap.clear();
    String orphanage = "orphanagehome";
    String url = getUrl(latitude, longitude, orphanage);
dataTransfer[0] = mMap;
dataTransfer[1] = url;

getNearbyPlacesData.execute(dataTransfer);
Toast.makeText(getActivity(), "Showing Nearby Orphanage...", Toast.LENGTH_SHORT).show();
    break;
    // Toast.makeText(Maps.this, "Your Message", Toast.LENGTH_LONG).show();

break;*/

caseR.id.B_orphanage:
mMap.clear();
    String school = "orphange";
    String url = getUrl(latitude, longitude, school);
dataTransfer[0] = mMap;
dataTransfer[1] = url;

getNearbyPlacesData.execute(dataTransfer);
Toast.makeText(getActivity(), "Showing Nearby orphanage...", Toast.LENGTH_SHORT).show();
    break;
    /** caseR.id.B_restaurants:
mMap.clear();
    String resturant = "restuarant";
url = getUrl(latitude, longitude, resturant);
dataTransfer[0] = mMap;
dataTransfer[1] = url;

getNearbyPlacesData.execute(dataTransfer);
Toast.makeText(MapsActivity.this, "Showing Nearby Restaurants", Toast.LENGTH_SHORT).show();
    break;
caseR.id.B_to:*/
    }
    }

private String getUrl(double latitude, double longitude, String nearbyPlace)
{

```

```

StringBuildergooglePlaceUrl = new
StringBuilder("https://maps.googleapis.com/maps/api/place/nearbysearch/json?");
googlePlaceUrl.append("location="+latitude+", "+longitude);
googlePlaceUrl.append("&radius="+PROXIMITY_RADIUS);
googlePlaceUrl.append("&type="+nearbyPlace);
googlePlaceUrl.append("&sensor=true");
    googlePlaceUrl.append("&key="+ "AlzaSyB8UAYW84SifM5HSm1zvRkvAVWbnMRImTo");

Log.d("MapsActivity", "url = "+googlePlaceUrl.toString());

    return googlePlaceUrl.toString();

}

@Override
public void onConnected(@Nullable Bundle bundle) {

locationRequest = new LocationRequest();
locationRequest.setInterval(100);
locationRequest.setFastestInterval(1000);
    locationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);

if(ContextCompat.checkSelfPermission(getActivity(),
android.Manifest.permission.ACCESS_FINE_LOCATION ) ==
PackageManager.PERMISSION_GRANTED)
    {
LocationServices.FusedLocationApi.requestLocationUpdates(client, locationRequest, this);
    }

}

publicbooleancheckLocationPermission()
{
if(ContextCompat.checkSelfPermission(getActivity(),
android.Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
)
    {

if (ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),
android.Manifest.permission.ACCESS_FINE_LOCATION))
    {
ActivityCompat.requestPermissions(getActivity(),new String[]
{android.Manifest.permission.ACCESS_FINE_LOCATION },REQUEST_LOCATION_CODE);
    }
    else
    {
ActivityCompat.requestPermissions(getActivity(),new String[]
{android.Manifest.permission.ACCESS_FINE_LOCATION },REQUEST_LOCATION_CODE);
    }
    return false;
}
}

```

```

    }
    else
        return true;
    }

    @Override
    public void onConnectionSuspended(int i) {

    }

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {

    }

    //private GoogleMap mMap;

    /*@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }*/

```

EVENTS:

```

package com.example.bhaskar.orphanage;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ListView;

public class Events extends Fragment {

    ListView listView;
    String[] fruitname = {"PAINTING COMPETITION", "DONATE BLOOD", "DONATE FOR EDUCATION", "ORPHAN SPONSORSHIP PROGRAM", "WINTER CAMPEIGN FOR ORPHANS", "GIFT AN ORPHAN"};
    String[] desc = {"TIME: 10 A.M PLACE: KOLKATA", "TIME: 10 A.M PLACE: KOLKATA", "TIME: 11 A.M PLACE: KOLKATA", "TIME: 09 A.M PLACE: KOLKATA", "TIME: 10 A.M PLACE: JHARGRAM", "TIME: 10 A.M PLACE: KOLKATA"};

```



```
Integer[]
imgid={R.drawable.donate,R.drawable.donation,R.drawable.help,R.drawable.help2,R.drawable.save,
R.drawable.orphan};
```

```
@Override
```

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    View rootView = inflater.inflate(R.layout.events, container, false);
```

```
lst=(ListView)rootView.findViewById(R.id.listview);
CustomListviewcustomListview=new CustomListview(getActivity(),fruitname,desc,imgid);
lst.setAdapter(customListview);
```

```
    return rootView;
}
```

```
}
```

DONATE:

```
public class Donate1 extends Fragment {
```

```
    Button cash,clothes,food,education;
```

```
@Override
```

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    View rootView = inflater.inflate(R.layout.donate, container, false);
```

```
cash = (Button) rootView.findViewById(R.id.button_donate_cash);
clothes =(Button)rootView.findViewById(R.id.button_donate_clothes);
food = (Button) rootView.findViewById(R.id.button_donate_food);
education=(Button)rootView.findViewById(R.id.button_donate_education);
cash.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(getActivity(), "Please Fill Up the Form ", Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(getActivity(),FormForDonate.class);
        startActivity(intentHome);
    }
});
```

```
clothes.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(getActivity(), "Please Fill Up the Form ", Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(getActivity(),FormForDonate.class);
        startActivity(intentHome);
    }
});
```

```

food.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(getActivity(), "Please Fill Up the Form ", Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(getActivity(),FormForDonate.class);
        startActivity(intentHome);
    }
});

```

```

education.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(getActivity(), "Please Fill Up the Form ", Toast.LENGTH_LONG).show();
        Intent intentHome = new Intent(getActivity(),FormForDonate.class);
        startActivity(intentHome);
    }
});
return rootView;
}
}

```

CASHFORM:

```

public class CashForm extends AppCompatActivity {

    CardForm cardForm;
    Button buy;
    AlertDialog.Builder alertBuilder;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cash_form);
        cardForm = findViewById(R.id.card_form1);
        buy = findViewById(R.id.button_pay);
        cardForm.cardRequired(true)
        .expirationRequired(true)
        .cvvRequired(true)
        .postalCodeRequired(true)
        .mobileNumberRequired(true)
        .mobileNumberExplanation("SMS is required in this number")
        .setup(CashForm.this);
        cardForm.getCvvEditText().setInputType(InputType.TYPE_CLASS_NUMBER |
        InputType.TYPE_NUMBER_VARIATION_PASSWORD);
        buy.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (cardForm.isValid()){
                    alertBuilder = new AlertDialog.Builder(CashForm.this);
                    alertBuilder.setTitle("Confirm before purchase ");

```

```

alertBuilder.setMessage("Card number: " + cardForm.getCardNumber() + "\n" +
    "Card expiry date: " + cardForm.getExpirationDateEditText().getText().toString() +
"\n" +
    "Card CVV: " + cardForm.getCvv() + "\n" +
    "Postal code: " + cardForm.getPostalCode() + "\n" +
    "Phone number: " + cardForm.getMobileNumber());
alertBuilder.setPositiveButton("Confirm", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        dialogInterface.dismiss();
        Toast.makeText(CashForm.this,"Thank you for PAYMENT",Toast.LENGTH_SHORT).show();
    }
});
alertBuilder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        dialogInterface.dismiss();
    }
});
AlertDialog alertDialog = alertBuilder.create();
alertDialog.show();

    } else {
        Toast.makeText(CashForm.this,"Please complete the form",Toast.LENGTH_LONG).show();
    }
    });
}

}
}

```

CONTACT US:

```
packagecom.example.bhaskar.orphanage;
```

```

importandroid.app.ProgressDialog;
importandroid.content.Intent;
importandroid.content.pm.PackageManager;
importandroid.net.Uri;
importandroid.support.design.widget.FloatingActionButton;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.text.TextUtils;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.EditText;

```

```

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class ContactActivity extends AppCompatActivity {

    private ProgressDialog progressDialog;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contact);

        FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab_calling);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent callIntent = new Intent(Intent.ACTION_CALL);
                callIntent.setData(Uri.parse("tel:9046129205"));
                if (ActivityCompat.checkSelfPermission(getApplicationContext(),
                    android.Manifest.permission.CALL_PHONE) != 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;
                }
                startActivity(callIntent);
            }
        });

        progressDialog = new ProgressDialog(this);

        final EditText your_name = (EditText) findViewById(R.id.your_name);
        final EditText your_email = (EditText) findViewById(R.id.your_email);
        final EditText your_subject = (EditText) findViewById(R.id.your_subject);
        final EditText your_message = (EditText) findViewById(R.id.your_message);

        Button email = (Button) findViewById(R.id.post_message);
        email.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                String name = your_name.getText().toString();
                String email = your_email.getText().toString();
            }
        });
    }
}

```

```
        String subject = your_subject.getText().toString();
        String message = your_message.getText().toString();

        if (TextUtils.isEmpty(name)){
            your_name.setError("Enter Your Name");
            your_name.requestFocus();
            return;
        }
        Boolean onError = false;
        if (!isValidEmail(email)) {
            onError = true;
            your_email.setError("Invalid Email");
            return;
        }

        if (TextUtils.isEmpty(subject)){
            your_subject.setError("Enter Your Subject");
            your_subject.requestFocus();
            return;
        }

        if (TextUtils.isEmpty(message)){
            your_message.setError("Enter Your Message");
            your_message.requestFocus();
            return;
        }

        Intent sendEmail = new Intent(android.content.Intent.ACTION_SEND);

        progressDialog.setMessage("Please Wait... ");
        progressDialog.show();

        /* Fill it with Data */
        sendEmail.setType("plain/text");
        sendEmail.putExtra(android.content.Intent.EXTRA_EMAIL, new
        String[]{"bhaskarchakraborty95@gmail.com"});
        sendEmail.putExtra(android.content.Intent.EXTRA_SUBJECT, subject);
        sendEmail.putExtra(android.content.Intent.EXTRA_TEXT,
            "name:"+name+"\n"+"Email ID:"+email+"\n"+"Message:"+'\n'+message);

        /* Send it off to the Activity-Chooser */
        startActivity(Intent.createChooser(sendEmail, "Send mail..."));

    }
    });
}
```

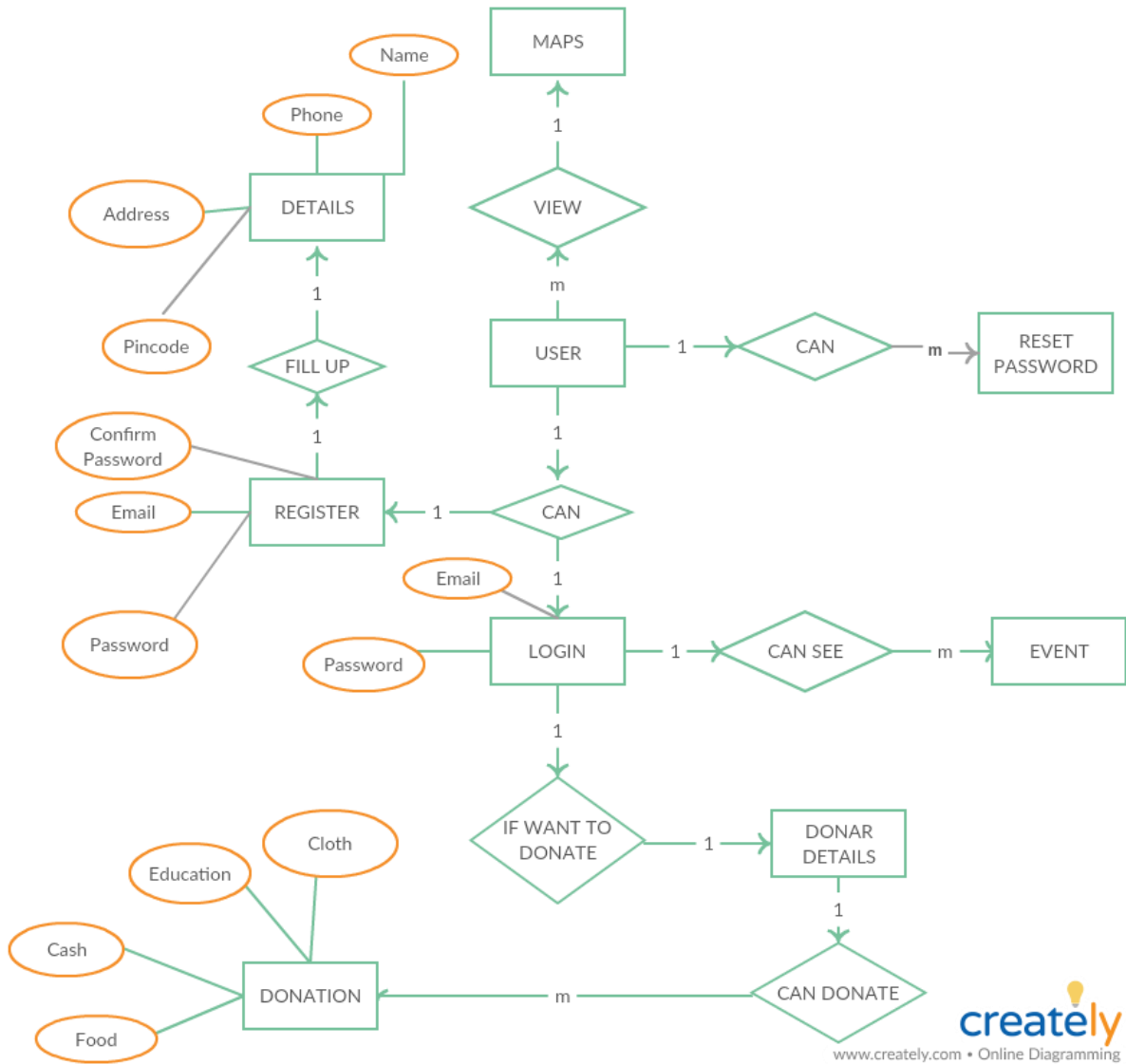
```
// validating email id

private boolean isValidEmail(String email) {
    String EMAIL_PATTERN = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@"
        + "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)*(\\.[A-Za-z]{2,})$";

    Pattern pattern = Pattern.compile(EMAIL_PATTERN);
    Matcher matcher = pattern.matcher(email);
    return matcher.matches();
}
}
```

DESIGN:

ENTITY RELATIONSHIP DIAGRAM



• **TESTING:**

- **The system is easy to use and is friendly user interface**
- **Keep tracks of donor and donation details.**
- **Registration and Login operations work perfectly.**

- If user forget their password then they can click on forget password to recover their password and can reset their password.
- User can see nearer orphanage in the maps.
- User can see events happening in the orphanages.
- User can donate anything through offline and online.

• **FUTURE SCOPE:**

“Little homes” is a project that is totally based on orphanage.

We have tried to give all the features that we possibly give.

But we add more features like old age home, rescue a life, and help acid attack women, etc in the future like we can keep track of donation details that are being paid through offline mode.

It's time to think faster as the world is changing very fast. So we thought we need to develop a website so that who doesn't have smart phones can also access it.

• **CONCLUSION:**

“Little Homes” provide shoulder to every living persons around us, who need our support. We are trying to provide best platform to the NGO's. The future scope of this project is to implement it as website for ease of access to end user and also to the user who doesn't have smart phones.

The main thing that we provide is to give platform to thousand of NGO'S, so that people can easily get information about NGO'S all over.

• **REFERENCE**

For completing this project, certain reference work has been used-

1. www.stackoverflow.com
2. Videos on www.youtube.com
3. [Github.com](https://github.com)
4. [Wikipedia](https://en.wikipedia.org)

THANK YOU

