NITTE EDUCATION TRUST | **N.M.A.M. INSTITUTE OF TECHNOLOGY**
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
: 08258 - 281039 – 281263, Fax: 08258 – 281265

Report on Mini Project

"ML Companion App"

Course Code: 18CSE42

Course Name: Mobile Application Development

Semester: VI Section : B

Submitted To,

Mrs. Shabari Shedthi B

Asst Prof Gd II,Dept of CSE,NMAMIT

Submitted By:

Name: B Ananthakrishna Rao USN: 4NM18CS031
Name: K Prahlad Bhat USN: 4NM18CS072

Date of submission: 20-06-2021

Signature of Course Instructor

NMAM Institute of Technology

(An Autonomous Institute Affiliated to VTU, Belagavi)

(A unit of NITTE Education Trust)

NITTE – 574110, UDUPI DIST., KARNATAKA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

"ML Companion App"

is a bonafide work carried out

by B Ananthakrishna Rao-

4NM18CS031

K Prahlad Bhat - 4NM18CS072

in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in computer science and engineering prescribed by the Vishvesvaraya Technological University, Belagavi during the year 2019 - 2020

It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in the report.

The mini-project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide Signature of HOD

ACKNOWLEDGEMENT

We believe that our project will be complete only after we thank the people who have contributed to making this project successful.

First and foremost, we express our deep sense of gratitude and indebtedness to our guide Mrs. Shabari Shedthi, Assistant Professor, Department of Computer Science and Engineering, for her guidance, constant encouragement, support, and suggestions for improvement during the course of our project.

We sincerely thank Dr. Jyothi Shetty, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

Our sincere thanks to our beloved principal, Dr. Niranjan N. Chiplunkar for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

Finally, we thank the staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

B Ananthakrishna Rao - 4NM18CS031

K Prahlad Bhat - 4NM18CS072

Table Of Contents

Chapter 1: Introduction

1.1 Scope

The gist of the project is to provide an accessible and reliable means of communication with the ML Model developed under the Mini Project "Stroke Prediction using Logistic Regression"

1.2 Importance

Strokes are becoming dangerously common, thus prevention of such an extensively damaging and traumatic medical emergency is something that cannot be done with extreme accuracy. However, some common lifestyle traits between individuals who have undergone this medical emergency can be used to pinpoint the probability of its occurrence. Hence, providing an interface to the prediction model that can run on possibly any device without any interference is of utmost importance.

1.3 Objective

The main intention of this application is to provide seamless and easy access to the ML Model developed under the Mini Project "Stroke Prediction Using Logistic Regression". In order to reduce the load on the mobile device that may occur during the calculations necessary for prediction, the ML Model has been hosted on the Heroku Cloud Platform and hence the application becomes very lightweight and communicates with this model via HTTP requests in order to process the data provided as inputs.

Chapter 2: Literature Survey

2.1 Technical Background

The approach is to provide a set of APIs to the trained ML Model, which can service the requests that users make in the form of data entered into the form via the application.

2.2 Existing System

Since this is an ad-hoc system designed for the sole intensive purpose of servicing the model which we have developed, there is no existing system to be referenced.

2.3 Proposed System

In order to reduce the load on the mobile device that may occur during the calculations necessary for prediction, the ML Model has been hosted on the Heroku Cloud Platform and hence the application becomes very lightweight and communicates with this model via HTTP requests in order to process the data provided as inputs. Furthermore, the application consists of several other features, namely:

1. News repository of top global headlines related to healthcare.
2. Email and SMS forwarding.
3. List of nearby hospitals based on the current user location via Google Maps.

Firebase has been used to authenticate users into the application and hence our goal of developing a secure, lightweight, and scalable application to pair up with our Machine Learning Model was achieved.

Chapter 3: System Requirements and Specifications

3.1 Functional Requirements

- Only authorized users must be able to use the system.
- A quick response from the system is expected.
- The system should be accessible from anywhere.
- The application should be able to run on most hardware.

3.2 User Requirements

- Android Device
- Internet Connection(for prediction and email)
- SIM Card for SMS
- Location Permissions for Map Functionality

3.3 Software Requirements(Developers)

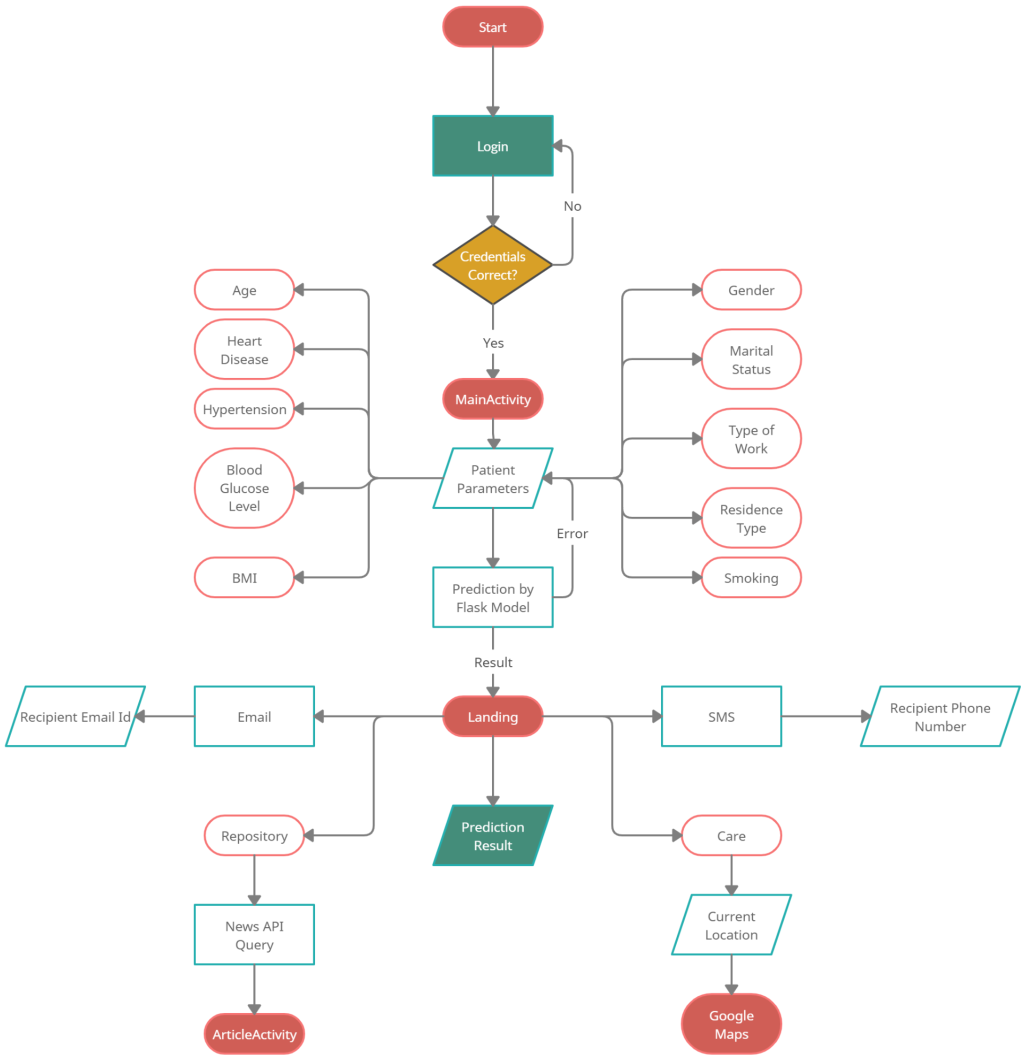
- Major Operating System(Windows/macOS/flavors of Linux)
- Android Studio

3.4 Hardware Requirements(Developers)

- 4 Gb RAM Minimum, 8 Gb RAM Recommended
- 2 Gb of Disk Space Minimum
- x86_64 CPU architecture; 2nd generation Intel Core or newer, or AMD CPU with support for a Windows Hypervisor.

Chapter 4: System Design

4.1 Data Flow Diagram

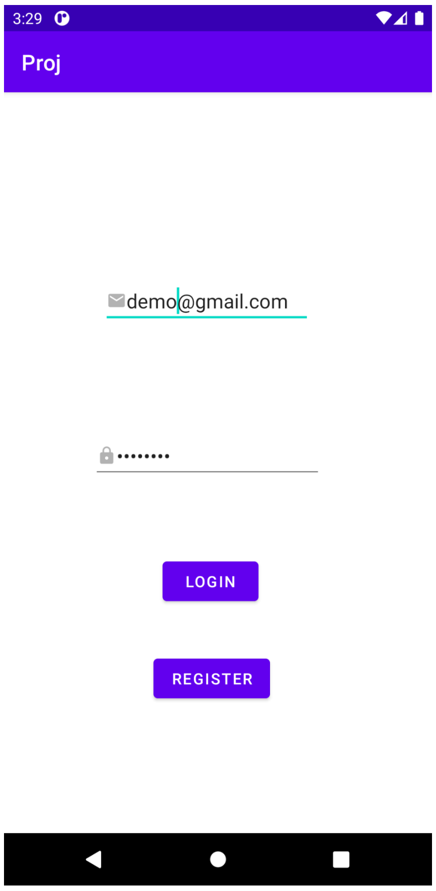


The flow diagram above represents the application control flow. Users are authenticated at the login page, following which they will be prompted to input the patient details at the main activity. Once the user submits these details, the data is processed by the Flask ML Model and the results are displayed on the Landing Page. At the landing page, the users will then have several options :

- Send an Email or an SMS.
- Go to the news repository.

Chapter 5: Implementation

5.1 Login Page



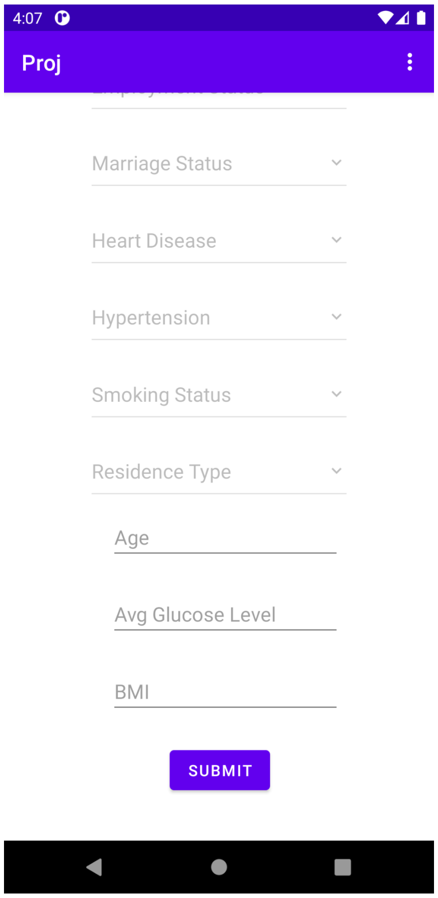
The login page is designed as shown above. New users can register themselves and previously registered users can log in. Firebase authentication ensures a smooth login operation

Implementation:

```
log.setOnClickListener(new View.OnClickListener() {         //Login Button
   @Override
   public void onClick(View v) {

auth.signInWithEmailAndPassword(email.getText().toString(),pw.getText().toS
tring()).addOnCompleteListener(LoginActivity.this, new
OnCompleteListener<AuthResult>() {
```

```
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Log.d("Login","Success");
                    Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
                    startActivity(intent);
                }else{
                    Toast.makeText(LoginActivity.this, "User not found",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
});
sign.setOnClickListener(new View.OnClickListener() {        //Signup Button
    @Override
    public void onClick(View v) {

auth.createUserWithEmailAndPassword(email.getText().toString(),pw.getText()
.toString()).addOnCompleteListener(LoginActivity.this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Log.d("Signup","Success");
                    Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
                    startActivity(intent);
                }
                else{
                    Toast.makeText(LoginActivity.this, "User already
exists", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
});
```

5.2 Main Page



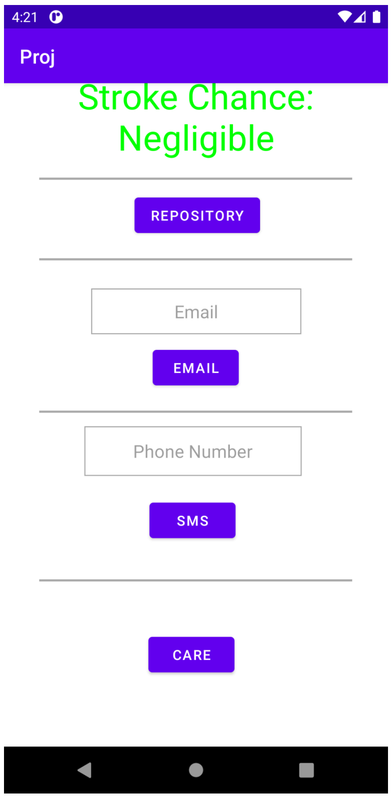
The users can choose particular values from a given plethora of dropdown menus and submit the data. This data is processed and converted into a JSON format which is then forwarded to the Flask Backend using HTTP Requests. The data at the backend undergoes processing and the result is sent back to the application in the form of a decimal value. This value is then displayed as a result on the Landing Page.

Implementation:

```
connect.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
       JSONObject jsonObject = new JSONObject();
```

```
        String gender = e1.getText().toString();

        String married = e2.getText().toString();
        String work = e3.getText().toString();
        String heart = e4.getText().toString();
        String hyper = e5.getText().toString();
        String smoke = e6.getText().toString();
        String age = ed1.getText().toString();
        String glucose = ed2.getText().toString();
        String resid = e7.getText().toString();
        String bmi = ed3.getText().toString();
        try {
                jsonObject.put("gender",gender);        //Put Data into JSON
Format for processing in backend
                jsonObject.put("married",married);
                jsonObject.put("work",work);
                jsonObject.put("heart",heart);
                jsonObject.put("hyper",hyper);
                jsonObject.put("smoke",smoke);
                jsonObject.put("age",age);
                jsonObject.put("glucose",glucose);
                jsonObject.put("bmi",bmi);
                jsonObject.put("resid",resid);
        } catch (JSONException e) {
            e.printStackTrace();
        }

        postRequest(jsonObject.toString(),url);
    }
});
private void postRequest(String message, String URL) {
        RequestBody requestBody = buildRequestBody(message);
        OkHttpClient okHttpClient = new OkHttpClient();
        Request request = new Request
                .Builder()
                .post(requestBody)
                .url(URL)
                .build();
        okHttpClient.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(final Call call, final IOException e) {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {


                        Toast.makeText(MainActivity.this, "Something went
wrong:" + " " + e.getMessage(), Toast.LENGTH_SHORT).show();
                        call.cancel();



                    }
                });

            }

            @Override
            public void onResponse(Call call, final Response response)
throws IOException {             //Received Response from Backend
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        try {
                            String res = response.body().string();
//                              Toast.makeText(MainActivity.this, res,
Toast.LENGTH_LONG).show();

                            Float f = Float.parseFloat(res);
                            f = f*100;
                            int val = Math.round(f);
                            Intent intent = new
Intent(MainActivity.this,Landing.class);        //Proceed to Result Landing
Page
                            intent.putExtra("val",val);
                            Log.d("VAL",f.toString());
                            startActivity(intent);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                });


            }
        });
    }
```

5.3 Landing Page



Once the response from the backend has been received, the user is brought to the landing page, where the result of the prediction is displayed. If the likelihood of the stroke occurring is:

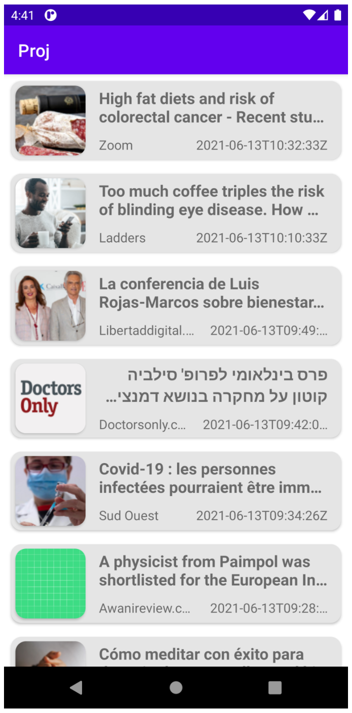
- Less than 40%, green text with negligible as the result is displayed.
- Between 40% to 70%, amber text with percentage value is displayed
- More than 70%, red text with percentage value is displayed.

Implementation:

```
protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_landing);
       Intent intent = getIntent();
       String res;
       int val = intent.getIntExtra("val",0);
       if(val<40)
           res = "Negligible";
```

```
        else
            res = val +"%";
        button = findViewById(R.id.repo);
        mail = findViewById(R.id.mail);
        result = findViewById(R.id.Result);
        result.setText("Stroke Chance: "+res);
        if(val>70)
            result.setTextColor(Color.RED);
        else if(val>40 && val<=70)
            result.setTextColor(Color.rgb(205,214,4));
        else
            result.setTextColor(Color.GREEN);
        mailbutton = findViewById(R.id.mailbutton);
        smsbutton = findViewById(R.id.smsbutton);
        phone = findViewById(R.id.sms);
        carebutton = findViewById(R.id.carebutton);
        body = result.getText().toString();
        if(val>70)
            body+=". Please report to the nearest medical center!";

        button.setOnClickListener(new View.OnClickListener() {
//Launch Articles Page

            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Landing.this,
ArticleActivity.class);
                startActivity(intent);
            }
        });
        mailbutton.setOnClickListener(new View.OnClickListener() {
//Send Email
            @Override
            public void onClick(View v) {
                String sendto = mail.getText().toString();
                Intent intent = new Intent(Intent.ACTION_SEND);
                intent.putExtra(Intent.EXTRA_EMAIL, new String[]{sendto});
                intent.putExtra(Intent.EXTRA_TEXT, body);
                intent.setType("message/rfc822");
                startActivity(Intent.createChooser(intent, "Choose your
email client:"));
            }
        });
        smsbutton.setOnClickListener(new View.OnClickListener() {
//Send SMS
            @Override
            public void onClick(View v) {
                String ph = phone.getText().toString();
                try {
                    Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse("sms:" + ph));
                    intent.putExtra("sms_body", body);
                    startActivity(intent);
                } catch (Exception e) {
                    Toast.makeText(Landing.this, e.getMessage(),
Toast.LENGTH_SHORT).show();
                    e.printStackTrace();
                }
            }
        });
        carebutton.setOnClickListener(new View.OnClickListener() {
//Open Google Maps with Current Coordinates passed on to it
            @Override
            public void onClick(View v) {
//                  Intent intent = new
Intent(getApplicationContext(),MapsActivity2.class);
//                  startActivity(intent);
//
                locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
                Criteria criteria = new Criteria();
                String best = locationManager.getBestProvider(criteria,
true);
                if (ActivityCompat.checkSelfPermission(Landing.this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(Landing.this,
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;
               }
               Location location =
locationManager.getLastKnownLocation(best);

               Uri gmmIntentUri =
Uri.parse("geo:"+location.getLatitude()+","+location.getLatitude()+"?q=hosp
itals");
               Intent mapIntent = new Intent(Intent.ACTION_VIEW,
gmmIntentUri);
               mapIntent.setPackage("com.google.android.apps.maps");
               startActivity(mapIntent);
           }
       });
   }
```

5.4 Repository


4:41

Proj

High fat diets and risk of colorectal cancer - Recent stu…
Zoom 2021-06-13T10:32:33Z

Too much coffee triples the risk of blinding eye disease. How …
Ladders 2021-06-13T10:10:33Z

La conferencia de Luis Rojas-Marcos sobre bienestar…
Libertaddigital.… 2021-06-13T09:49:…

פרס בינלאומי לפרופ' סילביה קוטון על מחקרה בנושא דמנצי…
Doctorsonly.c… 2021-06-13T09:42:0…

Covid-19 : les personnes infectées pourraient être imm…
Sud Ouest 2021-06-13T09:34:26Z

A physicist from Paimpol was shortlisted for the European In…
Awanireview.c… 2021-06-13T09:28:…

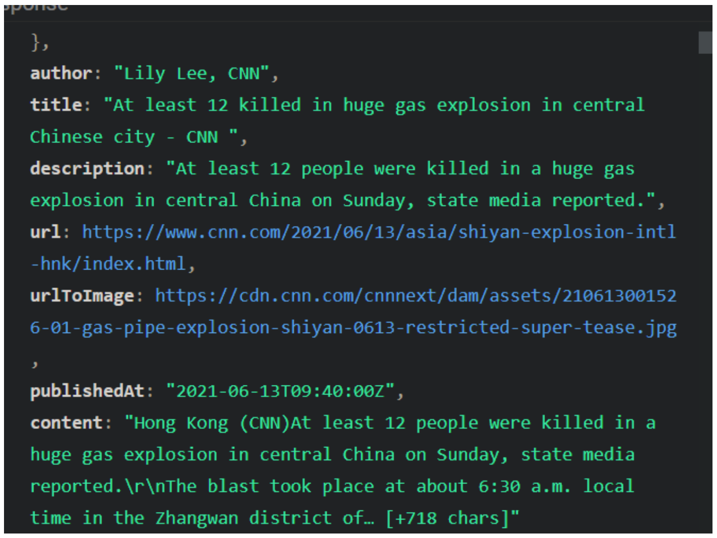
Cómo meditar con éxito para

The user can choose this option to receive the top headlines related to healthcare from around the world. This feature has been implemented using the NewsAPI, which sends news articles in the form of JSON objects from which the required parameters are extracted and data is presented to the user.

Implementation:

```java
private void init() {
   articleList = new ArrayList<>();
   rvNews = findViewById(R.id.rv_movies);
   progressBar = findViewById(R.id.progress_bar);
}
private void setRecyclerView() {
   rvNews.setLayoutManager(new LinearLayoutManager(this));
   rvNews.setAdapter(newsAdapter);
}
private void getNews() {            //Fetch List of Articles from NewsApi
using given Api key
   WebService webService =
WebServiceClient.getClient().create(WebService.class);
```

```
   Call<TopHeadline> call = webService.getTopHeadlines("health",
"cbb15f37c0864af7addb49db6466ff91");
   call.enqueue(new Callback<TopHeadline>() {
       @Override
       public void onResponse(Call<TopHeadline> call, Response<TopHeadline>
response) {
           progressBar.setVisibility(View.GONE);
           Log.d(TAG, "onResponse: " + response.code());
           assert response.body() != null : "Response Body Empty";
           articleList = response.body().getArticleList();
           newsAdapter = new NewsAdapter(ArticleActivity.this,
articleList);           //Load the articles onto recyclerview
           setRecyclerView();
       }

       @Override
       public void onFailure(Call<TopHeadline> call, Throwable t) {
           progressBar.setVisibility(View.GONE);
           Log.d(TAG, "onFailure: " + t.getLocalizedMessage());
       }
   });
}
```



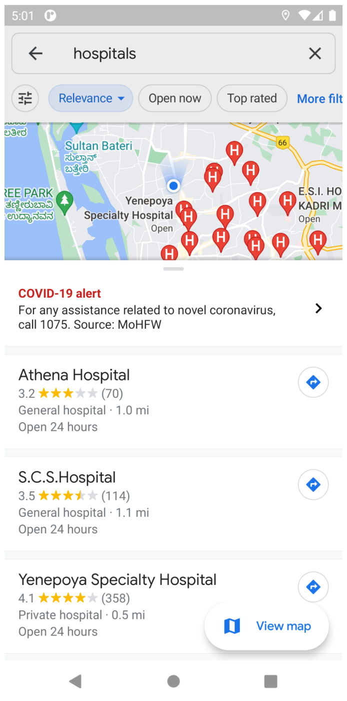
```
},
author: "Lily Lee, CNN",
title: "At least 12 killed in huge gas explosion in central
Chinese city - CNN ",
description: "At least 12 people were killed in a huge gas
explosion in central China on Sunday, state media reported.",
url: https://www.cnn.com/2021/06/13/asia/shiyan-explosion-intl
-hnk/index.html,
urlToImage: https://cdn.cnn.com/cnnnext/dam/assets/21061300152
6-01-gas-pipe-explosion-shiyan-0613-restricted-super-tease.jpg
,
publishedAt: "2021-06-13T09:40:00Z",
content: "Hong Kong (CNN)At least 12 people were killed in a
huge gas explosion in central China on Sunday, state media
reported.\r\nThe blast took place at about 6:30 a.m. local
time in the Zhangwan district of… [+718 chars]"
```

A typical NewsAPI response is shown above. From this, we extract the title, description, image URL, publishedAt, and display these details to the user.

5.5 Email & SMS

The user can enter an appropriate email id or a valid phone number to forward the prediction details. The system default applications are used to satisfy this operation.

5.6 Care



COVID-19 alert
For any assistance related to novel coronavirus, call 1075. Source: MoHFW

Athena Hospital
3.2 (70)
General hospital · 1.0 mi
Open 24 hours

S.C.S.Hospital
3.5 (114)
General hospital · 1.1 mi
Open 24 hours

Yenepoya Specialty Hospital
4.1 (358)
Private hospital · 0.5 mi
Open 24 hours

The user can use this feature to view hospitals that are close to their vicinity. Location permissions must be granted and based on the current location, a list of hospitals will be given to the user.
The hospitals will be listed, as shown above, Google Maps is used for this purpose.

5.7 Backend

The backbone of this application is the ML Model. This model has been loaded onto a Flask application and hosted on the Heroku Cloud Platform so as to reduce the processing load that can be generated by this application. Hence, any device is capable of running this application provided it has access to the Internet.

Chapter 6: Conclusion

Thus, we have been able to design and implement a mobile frontend for our Machine Learning Model along with several other features thus making it a flexible, scalable, multifunctional application that can be run on most devices.

Chapter 7: References

1. Google Android Documentation: https://developer.android.com/docs
2. Flask Documentation: https://flask.palletsprojects.com/en/2.0.x/
3. Better Material Spinner: https://github.com/Lesilva/BetterSpinner
4. NewsAPI Documentation: https://newsapi.org/docs
5. Firebase Documentation: https://firebase.google.com/docs
6. OkHTTP Reference: https://square.github.io/okhttp/
7. Glide: https://github.com/bumptech/glide
8. Heroku Python Guide: https://devcenter.heroku.com/articles/getting-started-with-python
9. Heroku Documentation: https://devcenter.heroku.com/categories/reference
10. RecyclerView Reference: https://guides.codepath.com/android/using-the-recyclerview
11. Stroke Prediction Dataset: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset
12. Retrofit: https://square.github.io/retrofit/
13. Gunicorn: https://docs.gunicorn.org/en/stable/configure.html