INDEX

1. **INTRODUCTION**
   1.1 Project Overview
   1.2 Purpose
2. **LITERATURE SURVEY**
   2.1 Existing problem
   2.2 References
   2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   3.1 Empathy Map Canvas
   3.2 Ideation & Brainstorming
   3.3 Proposed Solution
   3.4 Problem Solution fit
4. **REQUIREMENT ANALYSIS**
   4.1 Functional requirement
   4.2 Non-Functional requirements
5. **PROJECT DESIGN**
   5.1 Data Flow Diagrams
   5.2 Solution & Technical Architecture
   5.3 User Stories
6. **PROJECT PLANNING & SCHEDULING**
   6.1 Sprint Planning & Estimation
   6.2 Sprint Delivery Schedule
   6.3 Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   7.1 Feature 1
   7.2 Feature 2
   7.3 Database Schema (if Applicable)
8. **TESTING**
   8.1 Test Cases
   8.2 User Acceptance Testing
9. **RESULTS**
   9.1 Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION** & **FUTURE SCOPE**
12. **APPENDIX**
    Source Code
    GitHub & Project Demo Link

1.INTRODUCTION

1.1 PROJECT OVERVIEW

This application is intended to provide information about containment zones in a particular region by alerting people, through continuous monitoring of an individual's location. Key benefits of the application are monitoring people's activity and alerting them to their safety movements. Currently there are several research works undergoing in the country to prevent Covid-19 cases from rising. Previously our country was importing medical kits like PPE (Personal Protection Kits), mask from outside, but now it has been successful in developing these kits. Along with taking initiatives to fight this disease, our country has also taken steps to make people aware of the disease. The news and media have a great part in creating this awareness by informing the public about the preventive measures that can keep them away from infection. Awareness among the people to carry out all the preventive measures can immensely help to reduce spread of the virus. The country has created containment zones throughout the cities wherever Covid-19 cases have been reported to prevent further spread of the virus. These containment zones have been kept isolated from the outside public to ensure no contamination occurs outside. After more than 2 months of the lockdown, the government has relaxed some of the lockdown rules and has permitted reopening of government offices, bus and other road transportation facilities and shopping markets. People can move inside the city for work and other purposes. But the containment zones are still being kept isolated, and new containment zones are being formed wherever Covid-19 cases have been reported. These zones are highly contagious as droplets with virus coughed out from an unscreened asymptomatic patient can travel up to 8 m (). Though these containment zones are guarded by policemen, still there remains a chance that people might unknowingly step into them. In this situation where people can move in the city, these containment zones pose a risk of infection to these city dwellers. Therefore, informing people about the location of the containment zones can help them bypass and avoid these zones and thereby reduce the chance of community transmission.

1.2 PURPOSE

The project aims at building an application that provides information about the containment zones of a particular region by continuously monitoring an individual's location. The location of the containment zone must be stored in the Database. Alerts are sent using the notification service.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

Name of the paper:

COVID19-Tracker: A shiny app to produce to produce comprehensive data visualization for SARS-CoV-2 epidemic in Spain.

Author:

- Aurelio Tobias-Institute of Environmental Assessment and Water Research | IDAEA · Geosciences
- Cristian Tebe-Bellvitge Biomedical Research Institute
- Joan Valls
- Pau Satorra

Published Online:

April 2020

Topic:

Containment Zone Alerting Application

Theme:

Our theme of the project is to provide information about containment zones in a particular region by alerting people, by continuously monitoring an individual's location. Key benefits of the application are monitoring people's activity and alerting them of their safety movements.

Overall inference:

Data visualization is an important tool for exploring and communicating findings in medical research, and specially in epidemiological surveillance.

Name of the paper:

Mobile Health Apps That Help With COVID-19 Management: Scoping Review

Author:

Tracie Risling and Gunther Eysenbach

Published Online:

Aug 2020

Topic:

COVID-19 Management Application

Theme:

Our theme is to scope the evidence base on apps that were developed in response to COVID-19.

Overall inference:

Mobile health (mHealth) apps have played an important role in mitigating the coronavirus disease (COVID-19) response. However, there is no resource that provides a holistic picture of the available mHealth apps that have been developed to combat this pandemic

This review identifies that the majority of COVID-19 apps were for contact tracing and symptom monitoring. However, these apps are effective only if taken up by the community. The sharing of good practices across different countries can enable governments to learn

2.2 REFERENCES

1. AarogyaSetu(2020) https://play.google.com/store/apps/details?id=nic.goi.aarogyasetu&hl=en
2. Android developer-Locations (2020) https://developer.android.com/reference/android/location/Location#distanceBetween(double,%2520double,%2520double,%2520double,%2520float[])
3. BSafe tracking (2020) https://play.google.com/store/apps/details?id=com.cyberdome.bsafe&hl=en
4. Bahl P, Doolan C, de Silva C, Chughtai AA, Bourouiba L, MacIntyre CR. Airborne or droplet precautions for health workers treating Coronavirus disease 2019? *J Infect Dis.* 2020 doi: 10.1093/infdis/jiaa189. [PMC free article] [PubMed] [CrossRef] [Google Scholar]
5. Bihar Saathi (2020) https://play.google.com/store/apps/details?id=com.ibihar.saathi&hl=en
6. CG Covid-19 ePass (2020) https://play.google.com/store/apps/details?id=com.allsoft.corona&hl=en
7. Chikitsa Setu (2020) https://play.google.com/store/apps/details?id=com.abhitech.chikitsasetu&hl=en
8. Cloud Firestore Data model (2020) https://firebase.google.com/docs/firestore/data-model
9. Cloud Firestore (2020) https://firebase.google.com/docs/Firestore
10. Cloud Firestore SDKs and client libraries (2020) https://firebase.google.com/docs/Firestore/client/libraries

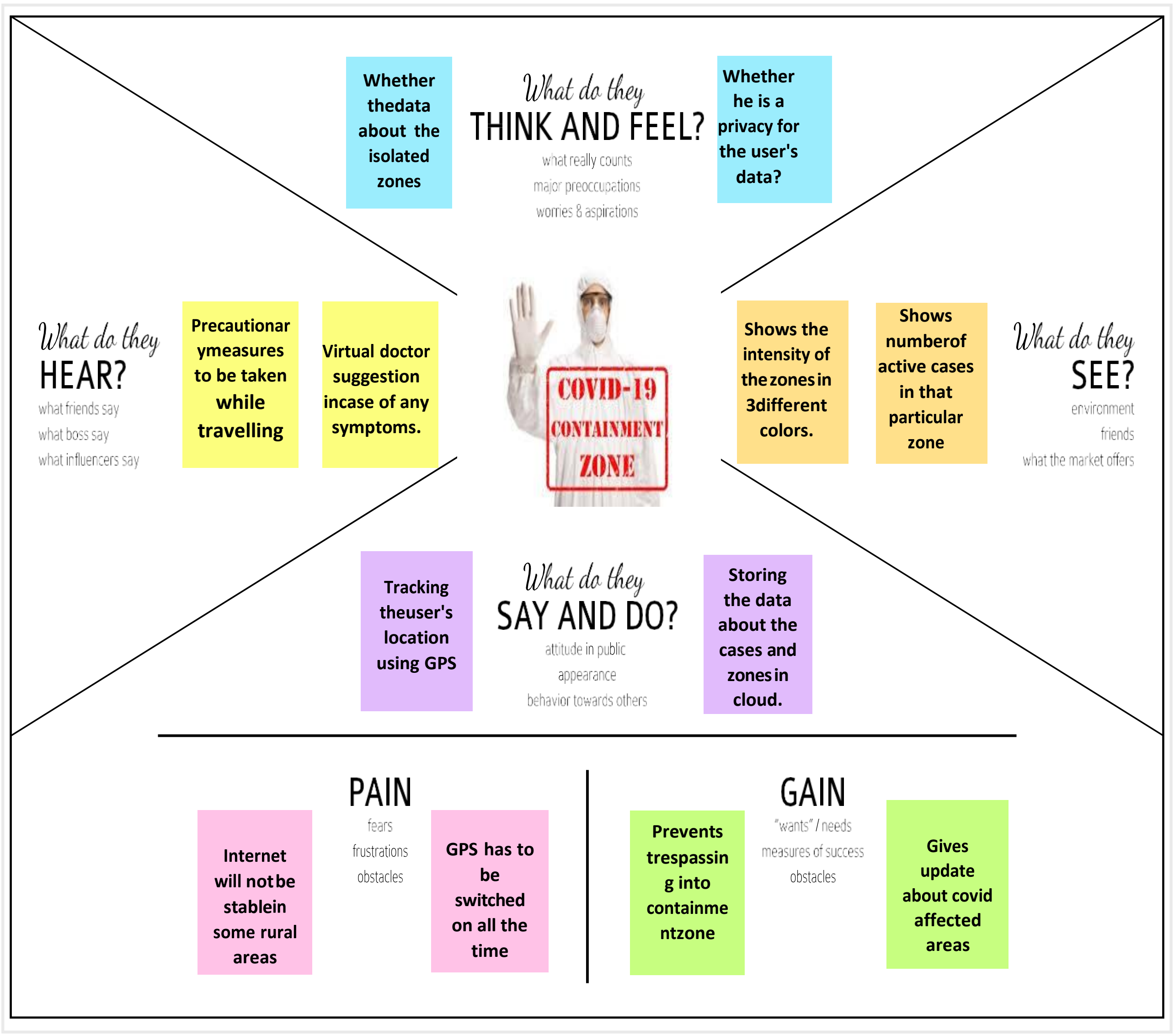
2.3 PROBLEM STATEMENT

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	User	View containment zones	The app works very slow	It requires stable andgood internet	Frustrated

PS-2	User	Explore the application	Confused whether to use the app or not	Not sure about user privacy and security	Frustrated
PS-3	Travellers	View safe routes	The safe route is optimal or not	Safe routes might lead tolong distance travel	Irritating
PS-4	User	View the covid cases	Not able to believe or not	Whether thenews is from verified resources	Irritating

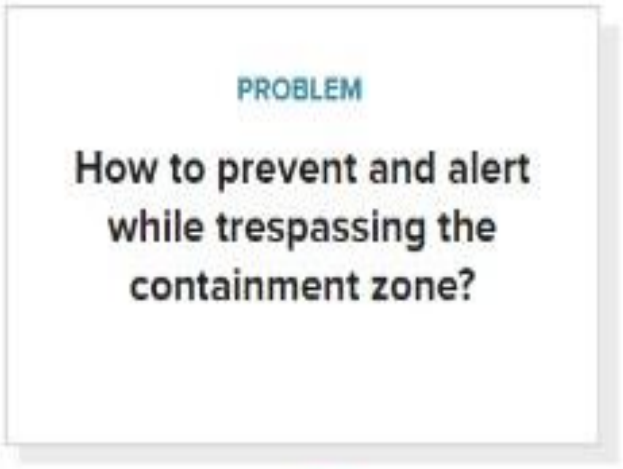
3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP


Whether thedata about the isolated zones
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Whether he is a privacy for the user's data?
What do they HEAR?
what friends say
what boss say
what influencers say
Precautionar ymeasures to be taken while travelling
Virtual doctor suggestion incase of any symptoms.
COVID-19 CONTAINMENT ZONE
Shows the intensity of the zones in 3different colors.
Shows numberof active cases in that particular zone
What do they SEE?
environment
friends
what the market offers
Tracking theuser's location using GPS
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Storing the data about the cases and zones in cloud.
PAIN
fears
frustrations
obstacles
Internet will notbe stablein some rural areas
GPS has to be switched on all the time
GAIN
"wants" / needs
measures of success
obstacles
Prevents trespassin g into containme ntzone
Gives update about covid affected areas

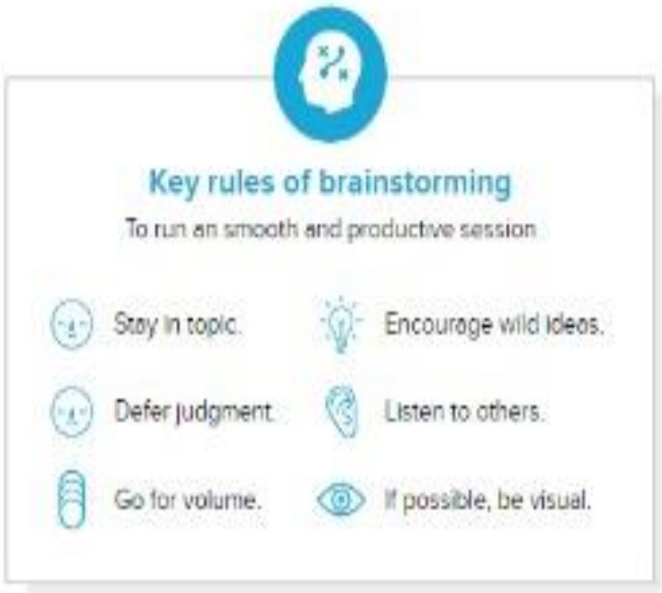
3.2 IDEATION & BRAINSTORMING

PROBLEM STATEMENT



PROBLEM

How to prevent and alert while trespassing the containment zone?



Key rules of brainstorming

To run an smooth and productive session

Stay in topic.

Encourage wild ideas.

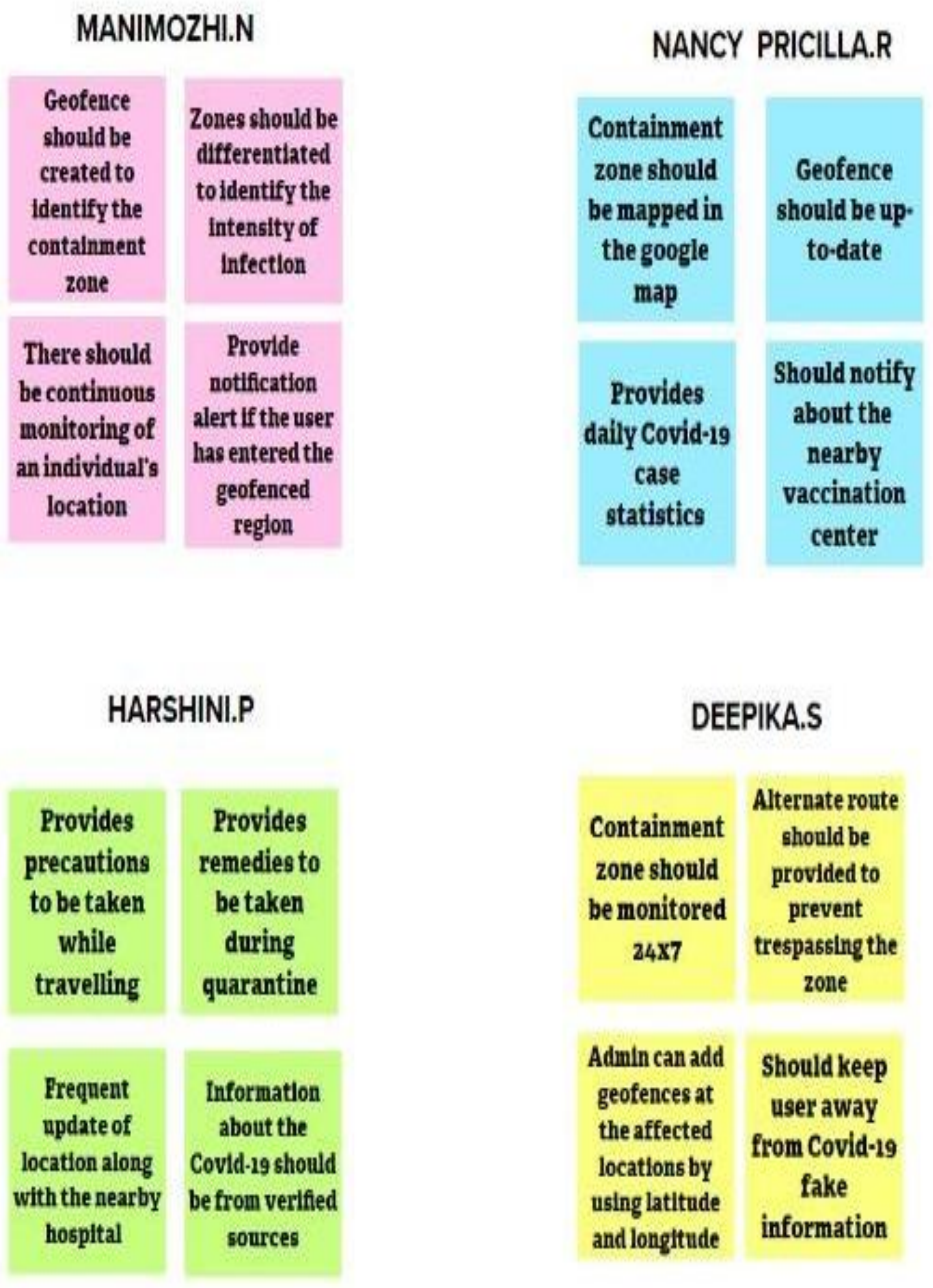
Defer judgment.

Listen to others.

Go for volume.

If possible, be visual.

BRAINSTORM



MANIMOZHI.N

- Geofence should be created to identify the containment zone
- Zones should be differentiated to identify the intensity of infection
- There should be continuous monitoring of an individual's location
- Provide notification alert if the user has entered the geofenced region

NANCY PRICILLA.R

- Containment zone should be mapped in the google map
- Geofence should be up-to-date
- Provides daily Covid-19 case statistics
- Should notify about the nearby vaccination center

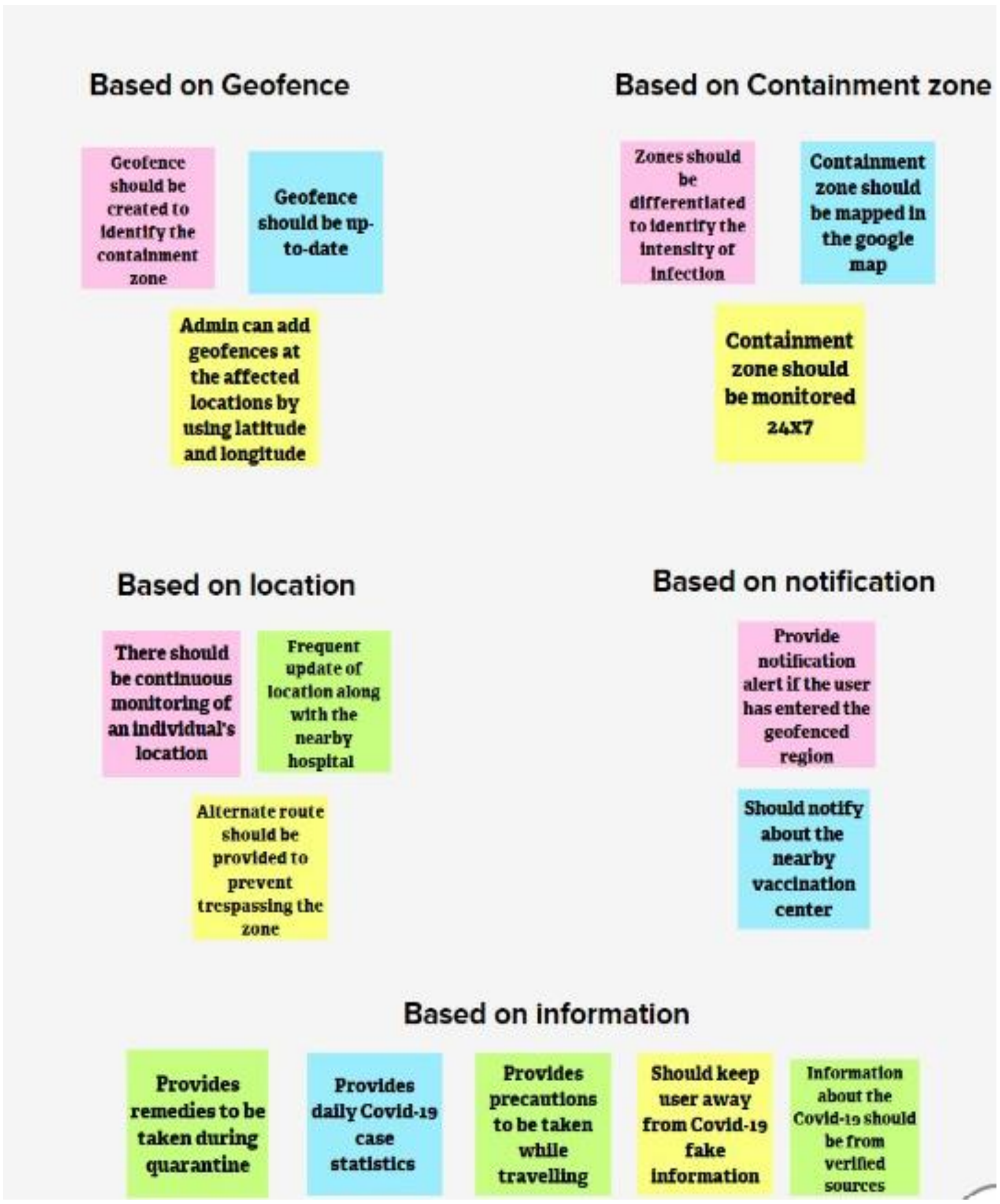
HARSHINI.P

- Provides precautions to be taken while travelling
- Provides remedies to be taken during quarantine
- Frequent update of location along with the nearby hospital
- Information about the Covid-19 should be from verified sources

DEEPIKA.S

- Containment zone should be monitored 24X7
- Alternate route should be provided to prevent trespassing the zone
- Admin can add geofences at the affected locations by using latitude and longitude
- Should keep user away from Covid-19 fake information

GROUP IDEAS



Based on Geofence

Geofence should be created to identify the containment zone

Geofence should be up-to-date

Admin can add geofences at the affected locations by using latitude and longitude

Based on Containment zone

Zones should be differentiated to identify the intensity of infection

Containment zone should be mapped in the google map

Containment zone should be monitored 24X7

Based on location

There should be continuous monitoring of an individual's location

Frequent update of location along with the nearby hospital

Alternate route should be provided to prevent trespassing the zone

Based on notification

Provide notification alert if the user has entered the geofenced region

Should notify about the nearby vaccination center

Based on information

Provides remedies to be taken during quarantine

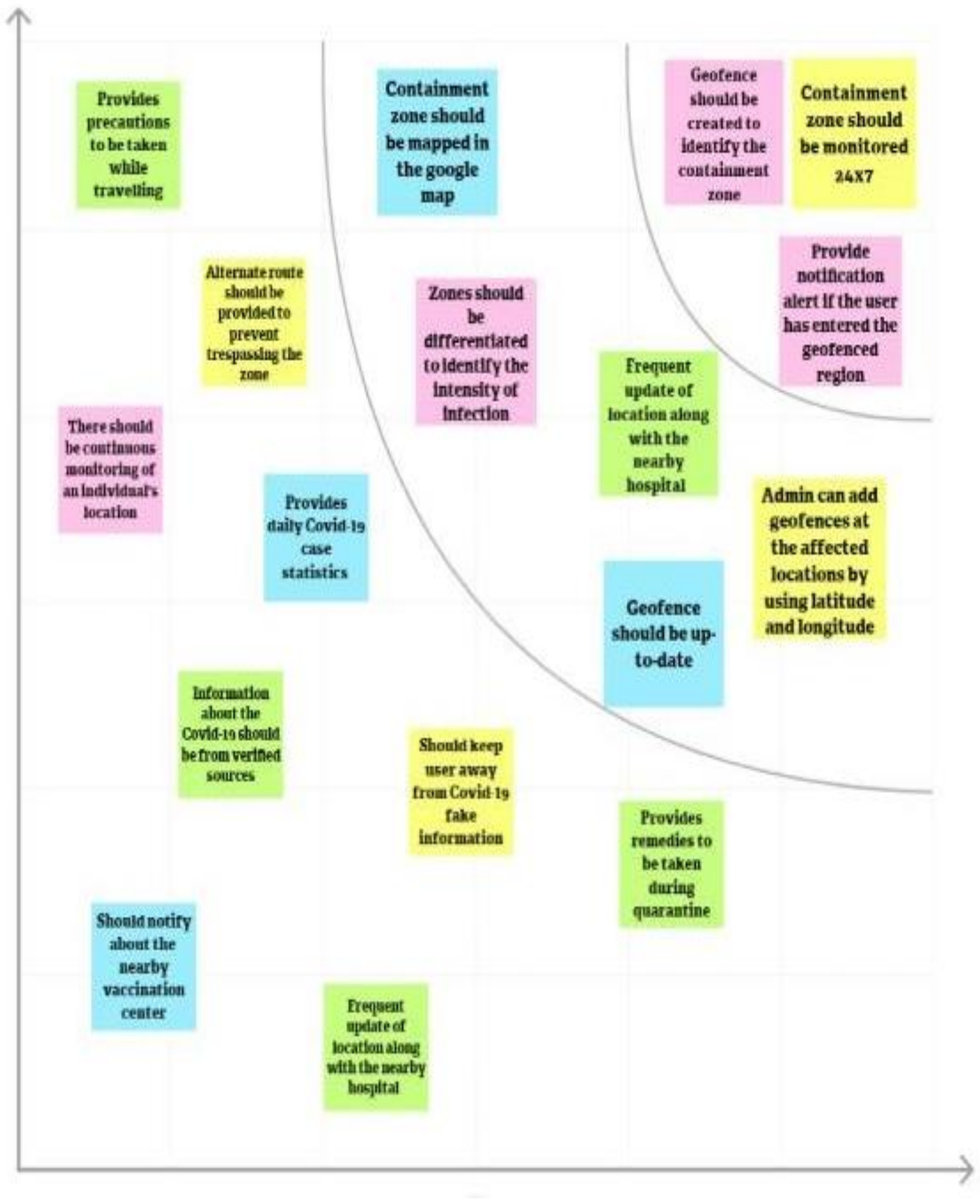
Provides daily Covid-19 case statistics

Provides precautions to be taken while travelling

Should keep user away from Covid-19 fake information

Information about the Covid-19 should be from verified sources

PRIORITIZE

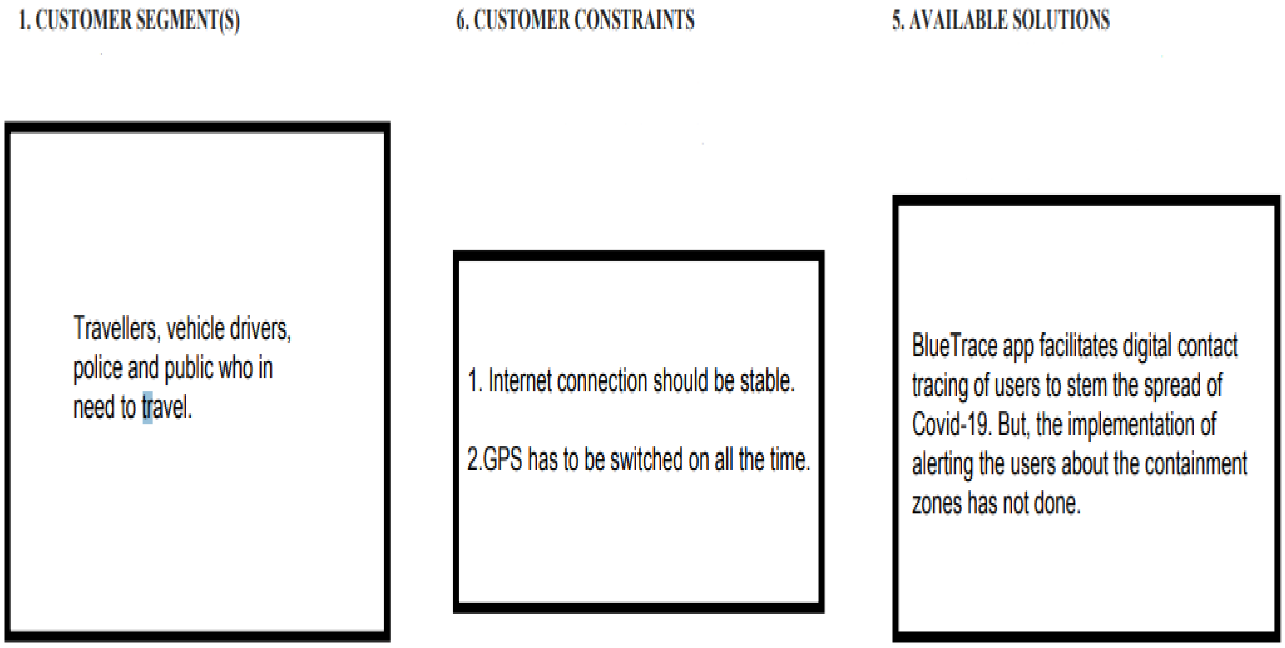

Provides precautions to be taken while travelling
Containment zone should be mapped in the google map
Geofence should be created to identify the containment zone
Containment zone should be monitored 24X7
Alternate route should be provided to prevent trespassing the zone
Zones should be differentiated to identify the intensity of infection
Provide notification alert if the user has entered the geofenced region
Frequent update of location along with the nearby hospital
There should be continuous monitoring of an individual's location
Provides daily Covid-19 case statistics
Admin can add geofences at the affected locations by using latitude and longitude
Geofence should be up-to-date
Information about the Covid-19 should be from verified sources
Should keep user away from Covid-19 fake information
Provides remedies to be taken during quarantine
Should notify about the nearby vaccination center
Frequent update of location along with the nearby hospital

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	To develop an application to notify the users while trespassing the containment zone.
2.	Idea / Solution description	This app provides information about containment zones in a particular region and alerts people, through continuous monitoring of an individual's location.
3.	Novelty / Uniqueness	Containment zones are differentiated using different colours to identify the intensity of infection. Updating containment zone location andcreating geofence within a 100-meter radius.
4.	Social Impact / Customer Satisfaction	Providing precautions to be taken whiletraveling. Frequently updating the location alongwith the nearby hospital.
5.	Business Model (Revenue Model)	Online healthcare services have been incorporated to provide remedies to be taken while quarantining.

6.	Scalability of the Solution	The radius of the geofence can be increased by 150 to 200 meters. Using an IP address to track the user even if the wi-fi or GPS gets turned off.

3.4 PROBLEM SOLUTION FIT



1. CUSTOMER SEGMENT(S)

Travellers, vehicle drivers, police and public who in need to travel.

6. CUSTOMER CONSTRAINTS

1. Internet connection should be stable.

2.GPS has to be switched on all the time.

5. AVAILABLE SOLUTIONS

BlueTrace app facilitates digital contact tracing of users to stem the spread of Covid-19. But, the implementation of alerting the users about the containment zones has not done.

2. JOBS-TO-BE-DONE / PROBLEMS

In order to reduce the spread of Covid-19 while travelling, the safest routes are provided to the user along with the intensity of the zones in 3 different colours

9. PROBLEM ROOT CAUSE

Due to the unknowing of covid affected regions, people are getting affected. This problem is solved by providing information about the containment zones.

7. BEHAVIOUR

Users give feedbacks and reviews to address the problem

3. TRIGGERS

Using the app after hearing its uses like precautionary measures, intensity of containment zones while travelling.

4. EMOTIONS: BEFORE / AFTER

Before: Fears and worries while travelling.

After: No fears or worries because the user will be known about the safe zones and routes while travelling

10. YOUR SOLUTION

This application provides information about containment zones in a particular region and alerts people, through continous monitoring of an individual's location.

8. CHANNELS of BEHAVIOUR

Users can provide suggesstions to add additional features. They can also report in case of any issues.

4 REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	It can be registered by a valid phone number or email id.
FR-2	User Confirmation	A verification code can be received by registering a phonenumber or mail id.
FR-3	Notification Alert	Through phone number or email id if user enters thecontainment zone.
FR-4	Show Containment Zone	It shows different color virtual boundaries.

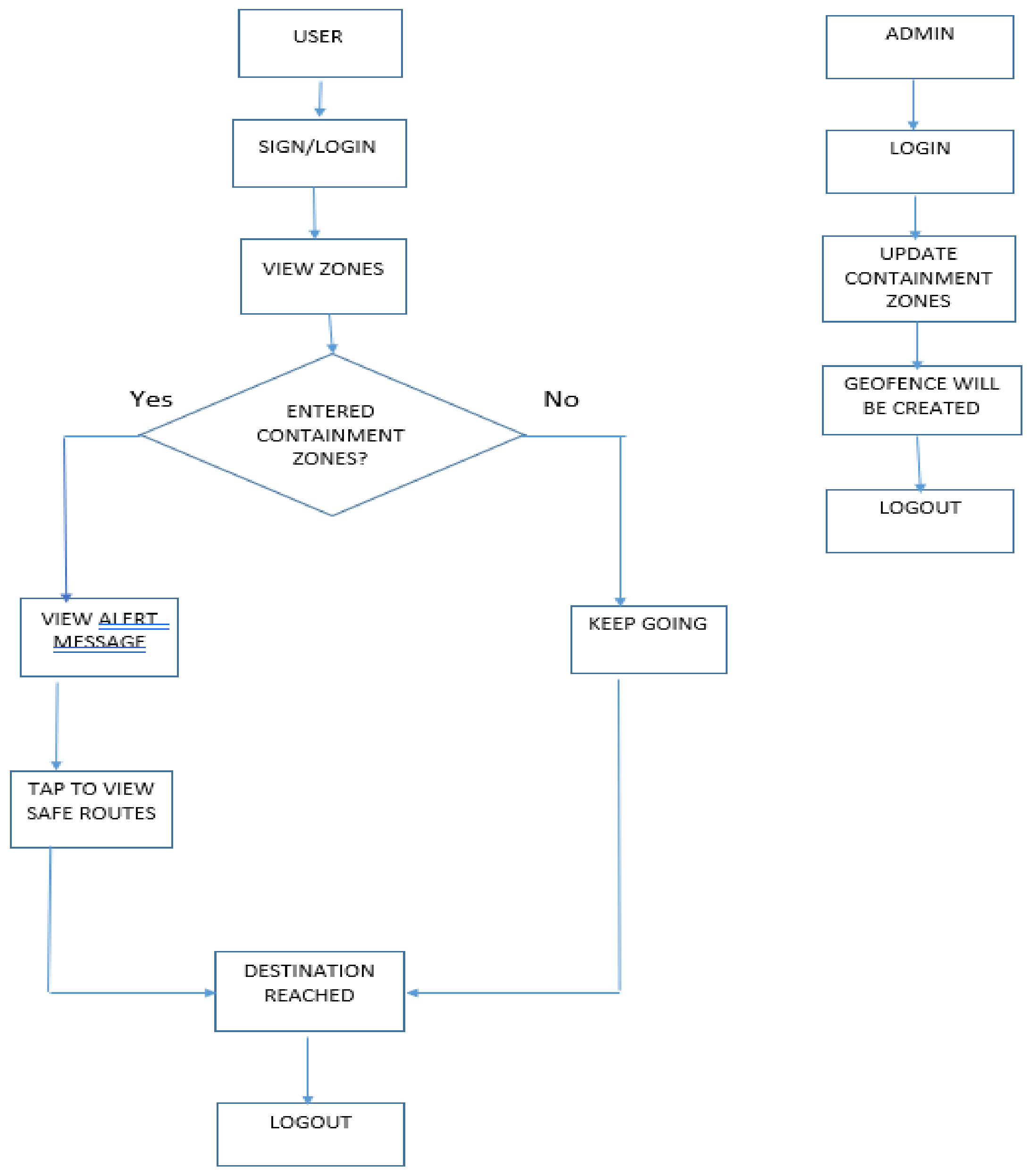
4.2 NON-FUNCTIONAL REQUIREMENT

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Application updates the location of the areas in aGoogle map which are identified to be the containment zones
NFR-2	**Security**	The data collected from the user is stored securely.
NFR-3	**Reliability**	The user can trust the result and navigate safely.
NFR-4	**Performance**	The accurate result can be achieved due to real-timelocation sharing.

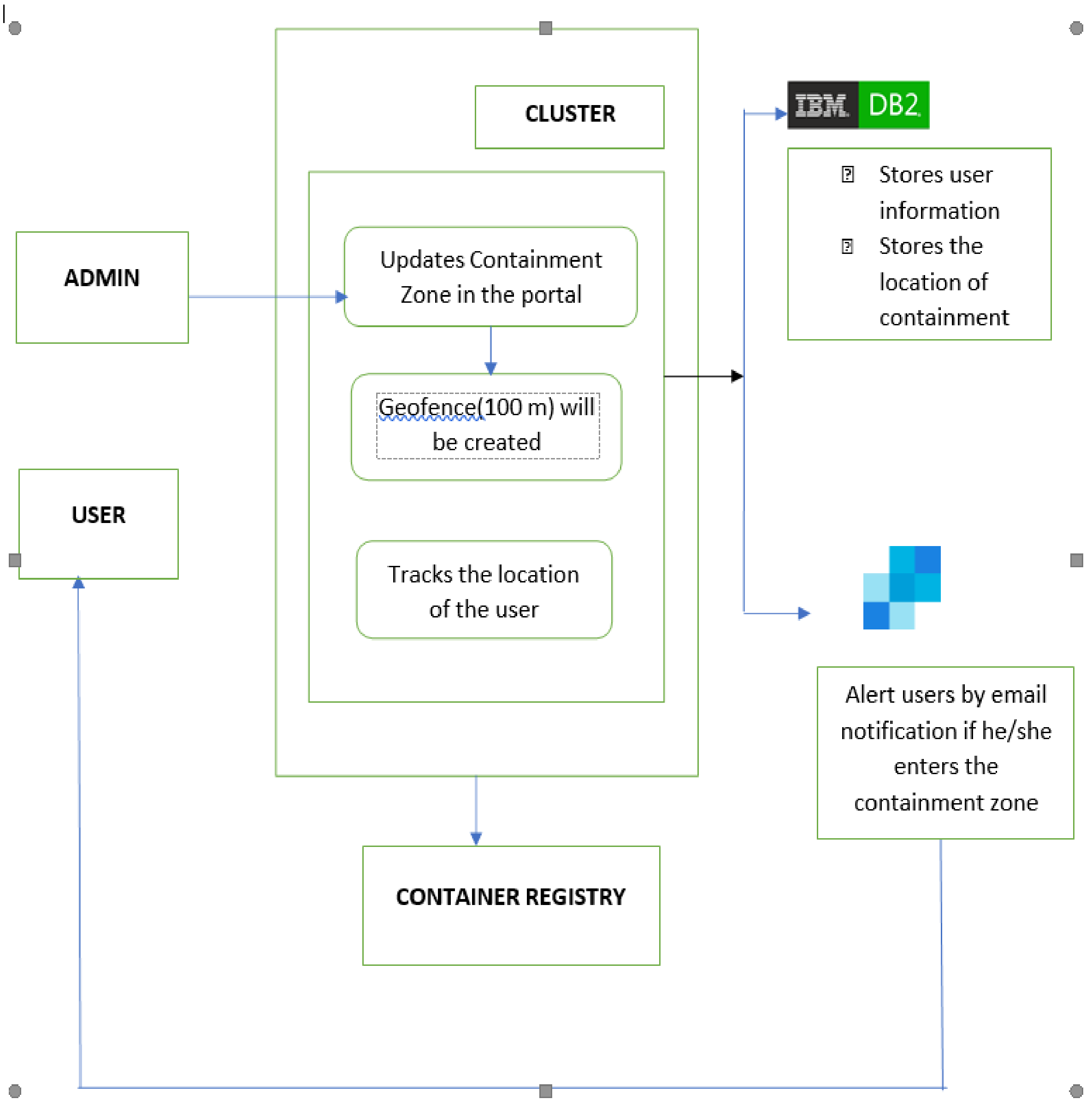
NFR-5	**Availability**	Available if the network bandwidth of the user is of good range.
NFR-6	**Scalability**	The application can be used from anywhere and can also be implemented for both mobile and web apps for the user to interact.

5 .PROJECT DESIGN

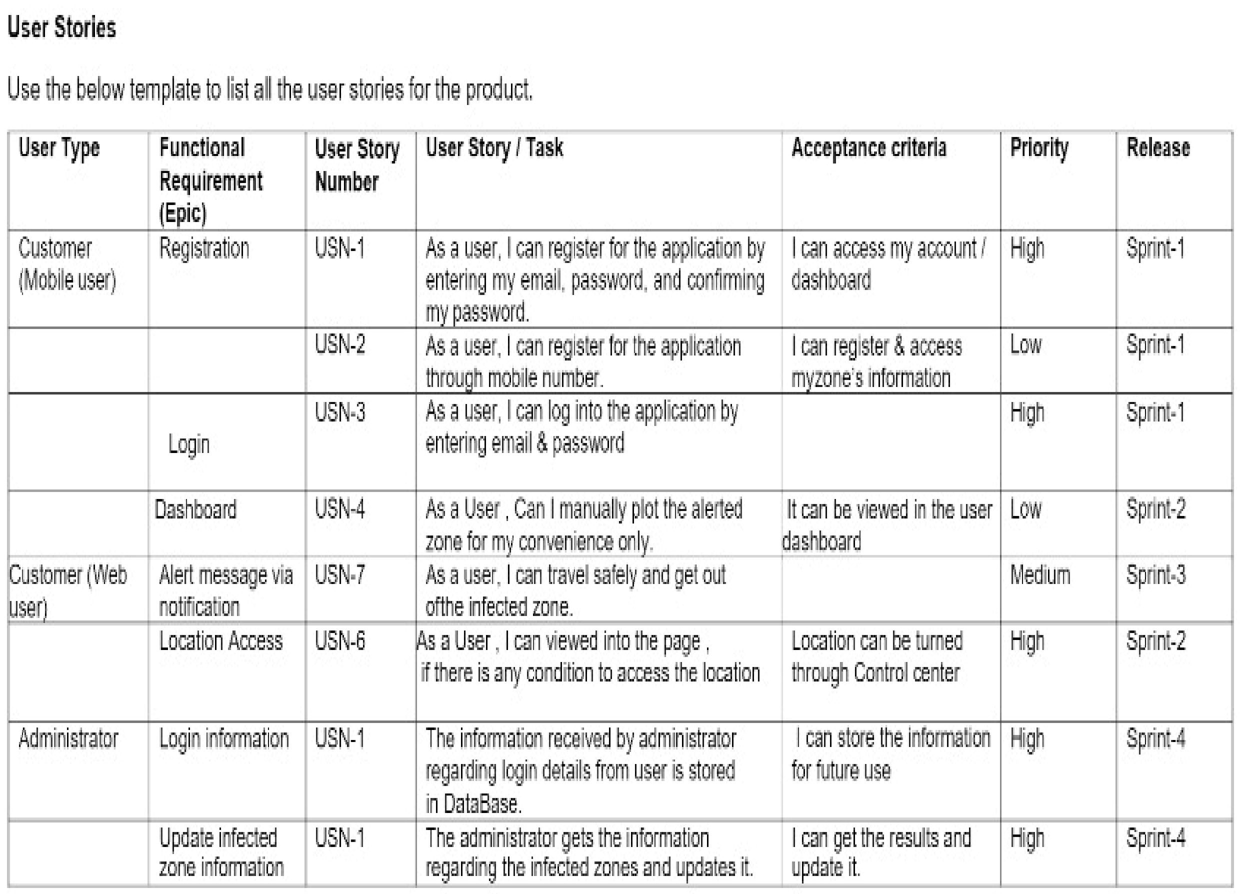
5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE


CLUSTER
ADMIN
Updates Containment Zone in the portal
Geofence(100 m) will be created
USER
Tracks the location of the user
CONTAINER REGISTRY
IBM DB2
Stores user information
Stores the location of containment
Alert users by email notification if he/she enters the containment zone

5.3 USER STORIES

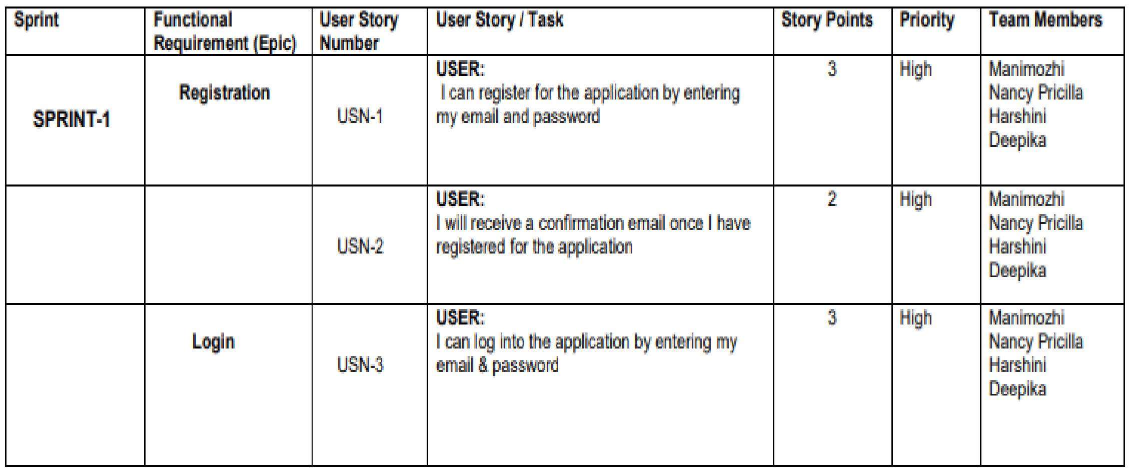

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I can register for the application through mobile number.	I can register & access myzone's information	Low	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-4	As a User , Can I manually plot the alerted zone for my convenience only.	It can be viewed in the user dashboard	Low	Sprint-2
Customer (Web user)	Alert message via notification	USN-7	As a user, I can travel safely and get out ofthe infected zone.		Medium	Sprint-3
	Location Access	USN-6	As a User , I can viewed into the page , if there is any condition to access the location	Location can be turned through Control center	High	Sprint-2
Administrator	Login information	USN-1	The information received by administrator regarding login details from user is stored in DataBase.	I can store the information for future use	High	Sprint-4
	Update infected zone information	USN-1	The administrator gets the information regarding the infected zones and updates it.	I can get the results and update it.	High	Sprint-4

6 .PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION



Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
SPRINT-1	**Registration**	USN-1	**USER:** I can register for the application by entering my email and password	3	High	Manimozhi Nancy Pricilla Harshini Deepika
		USN-2	**USER:** I will receive a confirmation email once I have registered for the application	2	High	Manimozhi Nancy Pricilla Harshini Deepika
	Login	USN-3	**USER:** I can log into the application by entering my email & password	3	High	Manimozhi Nancy Pricilla Harshini Deepika

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
SPRINT-2	**Dashboard**	USN-4	**USER:** I need to give permission to access my location	5	High	Manimozhi Nancy Pricilla Harshini Deepika
		USN-5	**USER:** I can view the map with the containment zones	5	High	Manimozhi Nancy Pricilla Harshini Deepika
	Service	USN-6	**ADMIN:** I need to update the containment zones.	5	High	Manimozhi Nancy Pricilla Harshini Deepika

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
SPRINT-3	**Service**	USN-7	**ADMIN:** I need to differentiate the containment zones based on the intensity of infection.	3	Medium	Manimozhi Nancy Pricilla Harshini Deepika
		USN-8	**ADMIN:** I need to provide precautionary measures when they travel.	3	Medium	Manimozhi Nancy Pricilla Harshini Deepika
		USN-9	**ADMIN:** I need to provide information about the nearby hospitals	3	Low	Manimozhi Nancy Pricilla Harshini Deepika

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
SPRINT-4	**Service**	USN-10	**ADMIN:** I need to alert the user when they enter the containment zone through email or SMS	5	High	Manimozhi Nancy Pricilla Harshini Deepika
		USN-11	**ADMIN:** I need to provide medical recommendations by collaborating with hospitals.	3	Low	Manimozhi Nancy Pricilla Harshini Deepika
	Data collection	USN-12	**ADMIN:** I need to store user details on the cloud	5	High	Manimozhi Nancy Pricilla Harshini Deepika
		USN-13	**ADMIN:** I need to collect details about covid-19 cases from verified sources	5	High	Manimozhi Nancy Pricilla Harshini Deepika

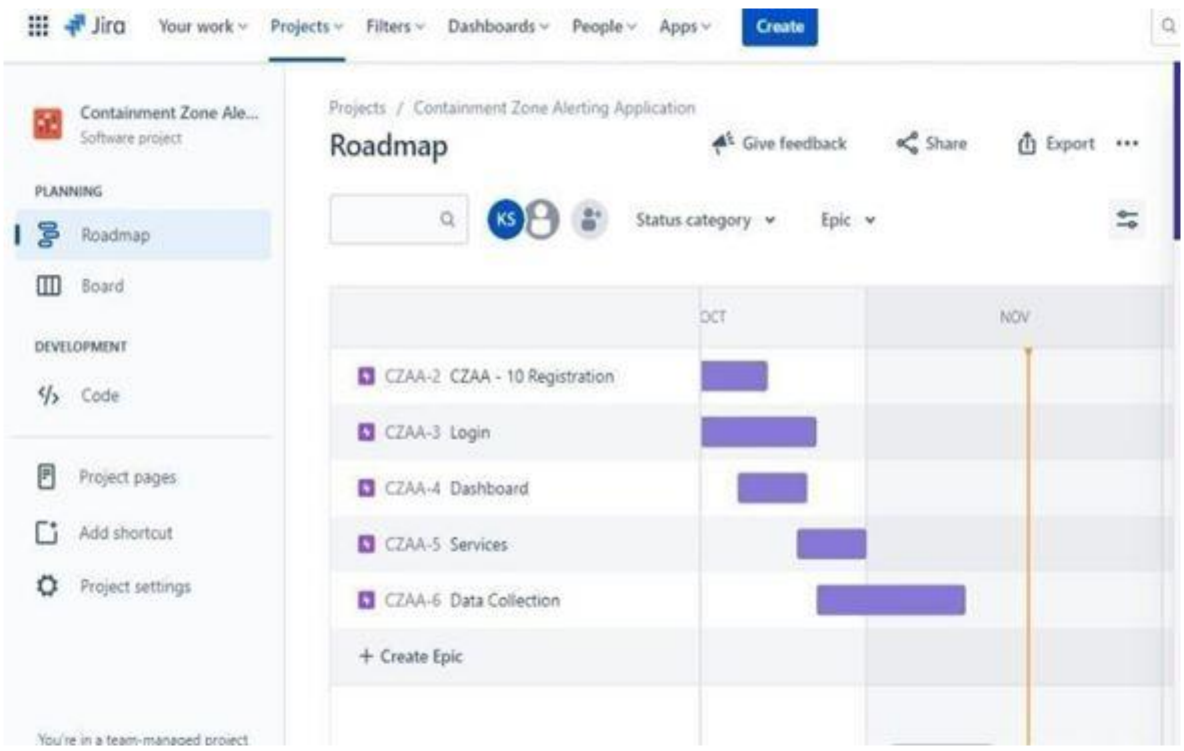
Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.2 Sprint delivery schedule

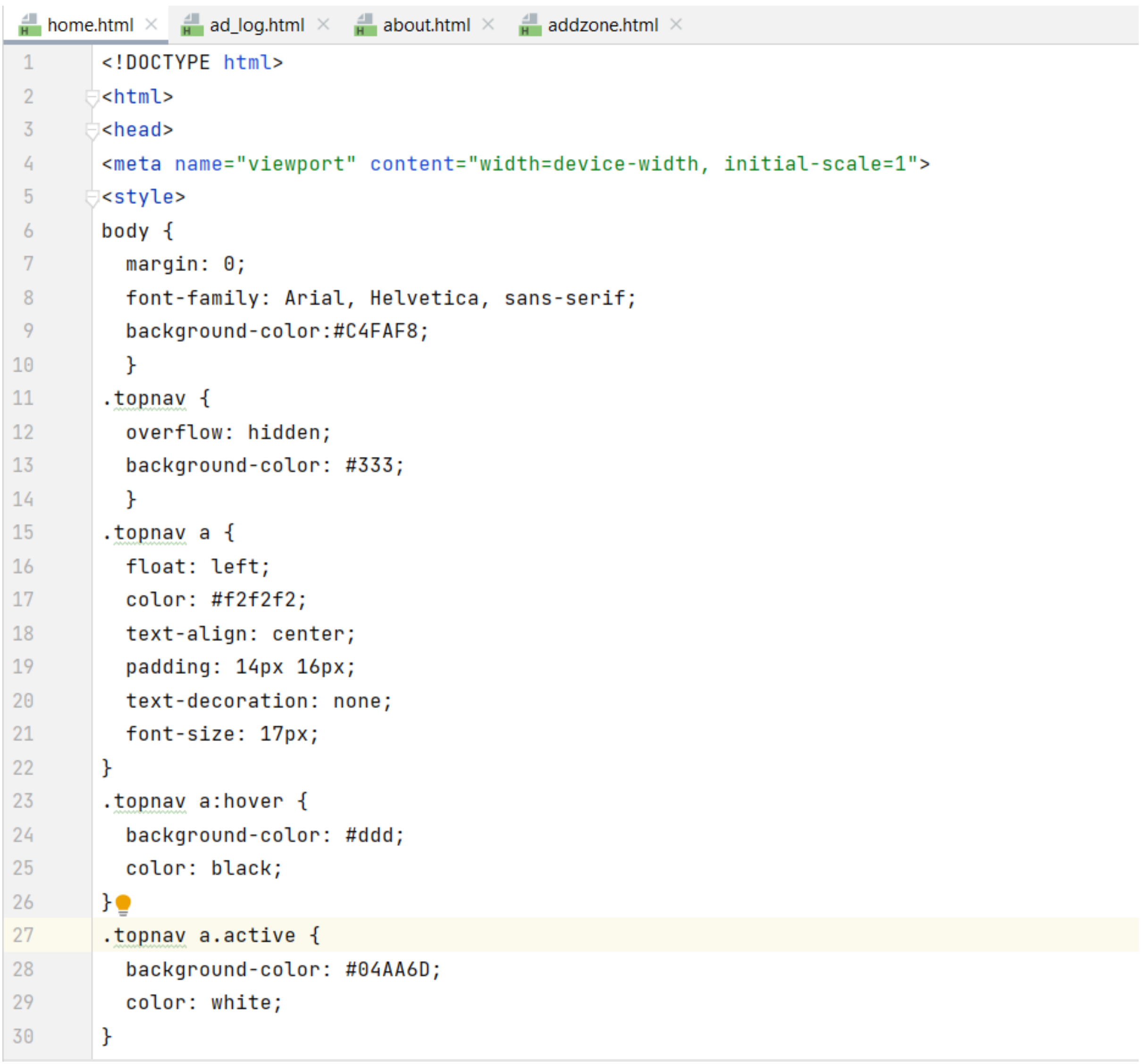
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 REPORT FROM JIRA

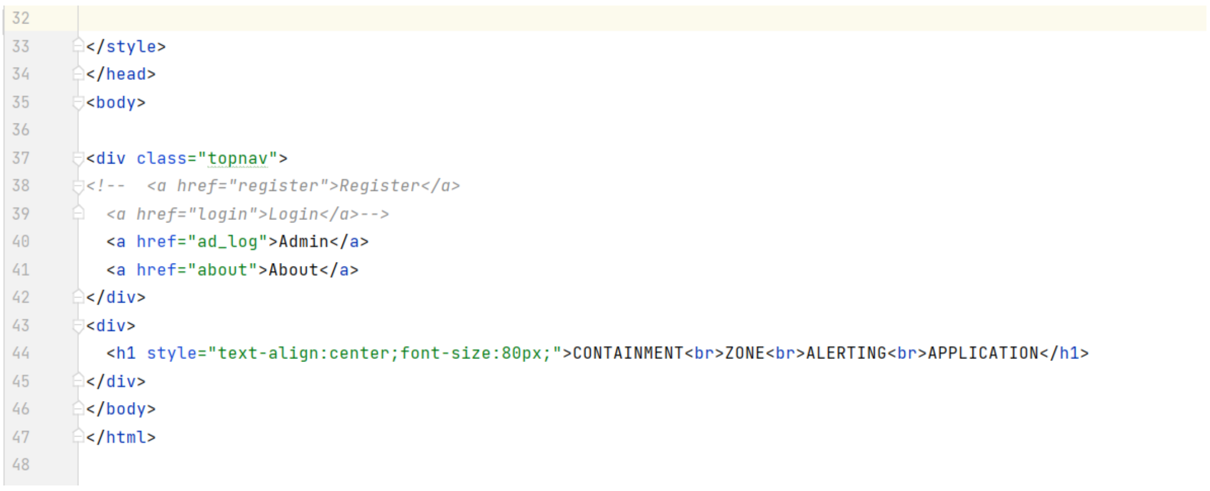


7.Coding and solutioning

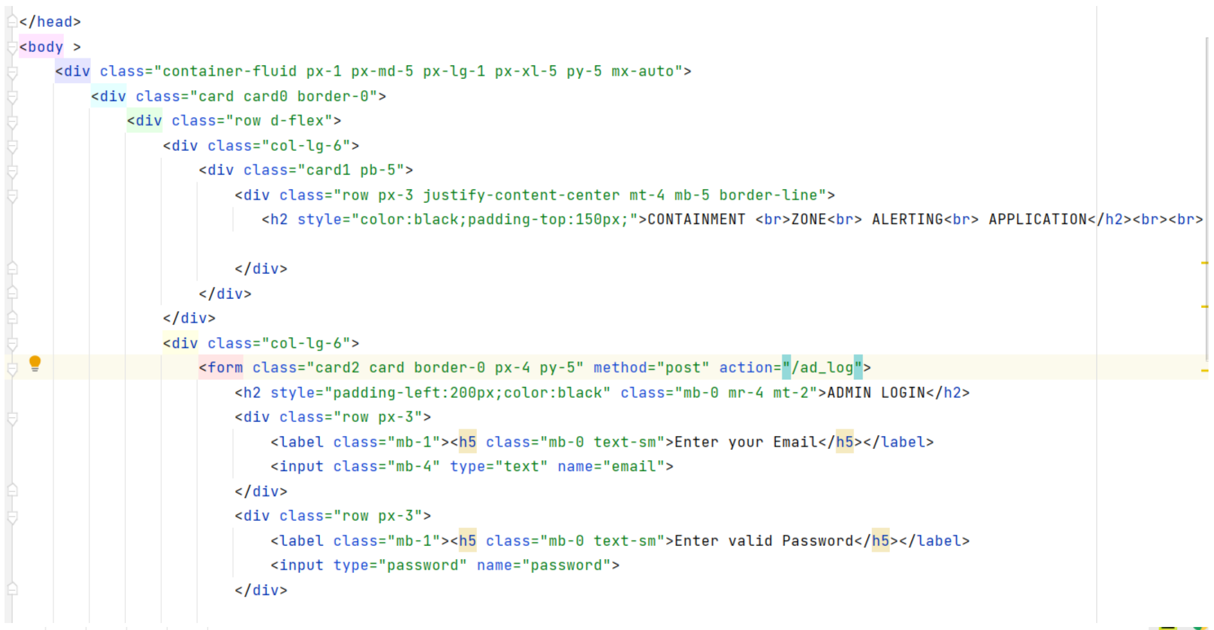
7.1 Feature -1



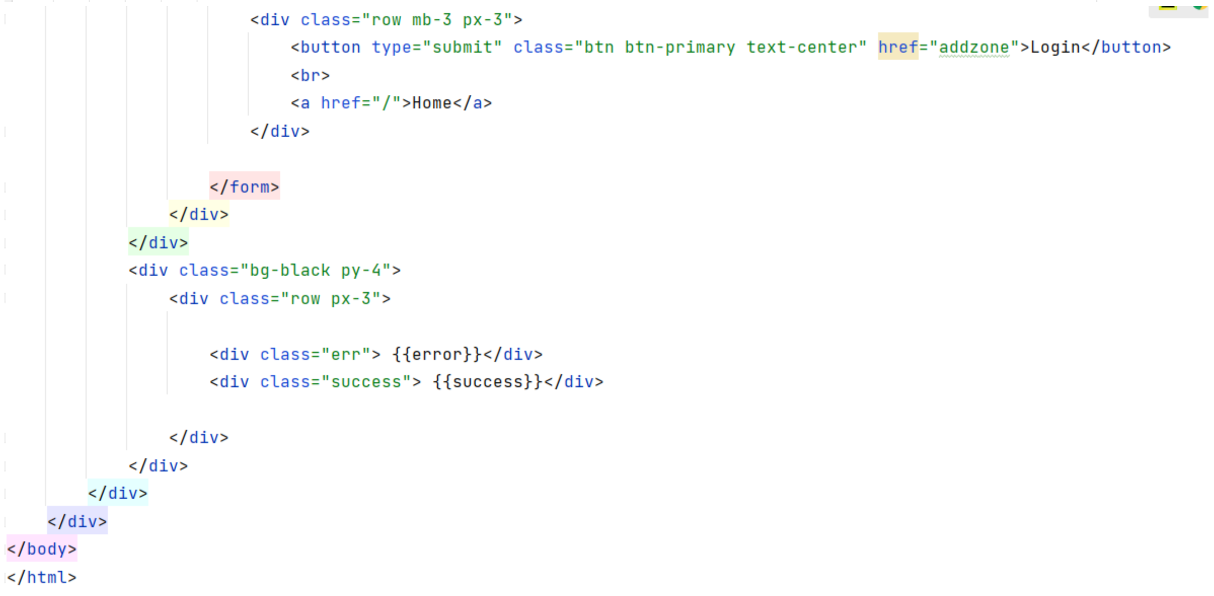
```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
  background-color:#C4FAF8;
  }
.topnav {
  overflow: hidden;
  background-color: #333;
  }
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
.topnav a.active {
  background-color: #04AA6D;
  color: white;
}
```



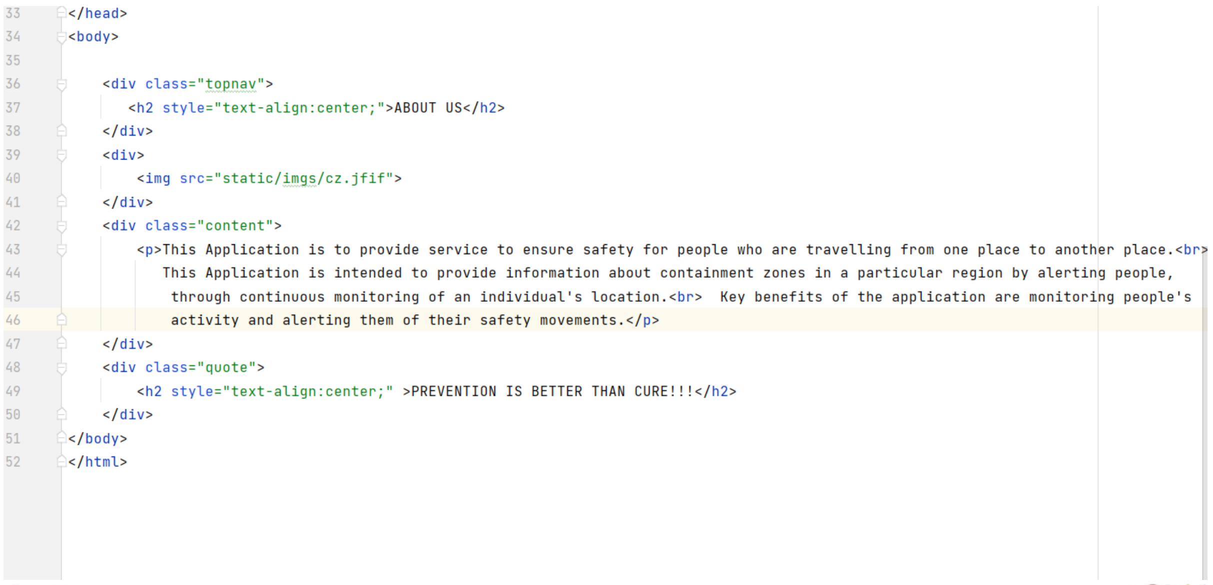
```

</style>
</head>
<body>

<div class="topnav">
<!--  <a href="register">Register</a>
  <a href="login">Login</a>-->
  <a href="ad_log">Admin</a>
  <a href="about">About</a>
</div>
<div>
  <h1 style="text-align:center;font-size:80px;">CONTAINMENT<br>ZONE<br>ALERTING<br>APPLICATION</h1>
</div>
</body>
</html>

```



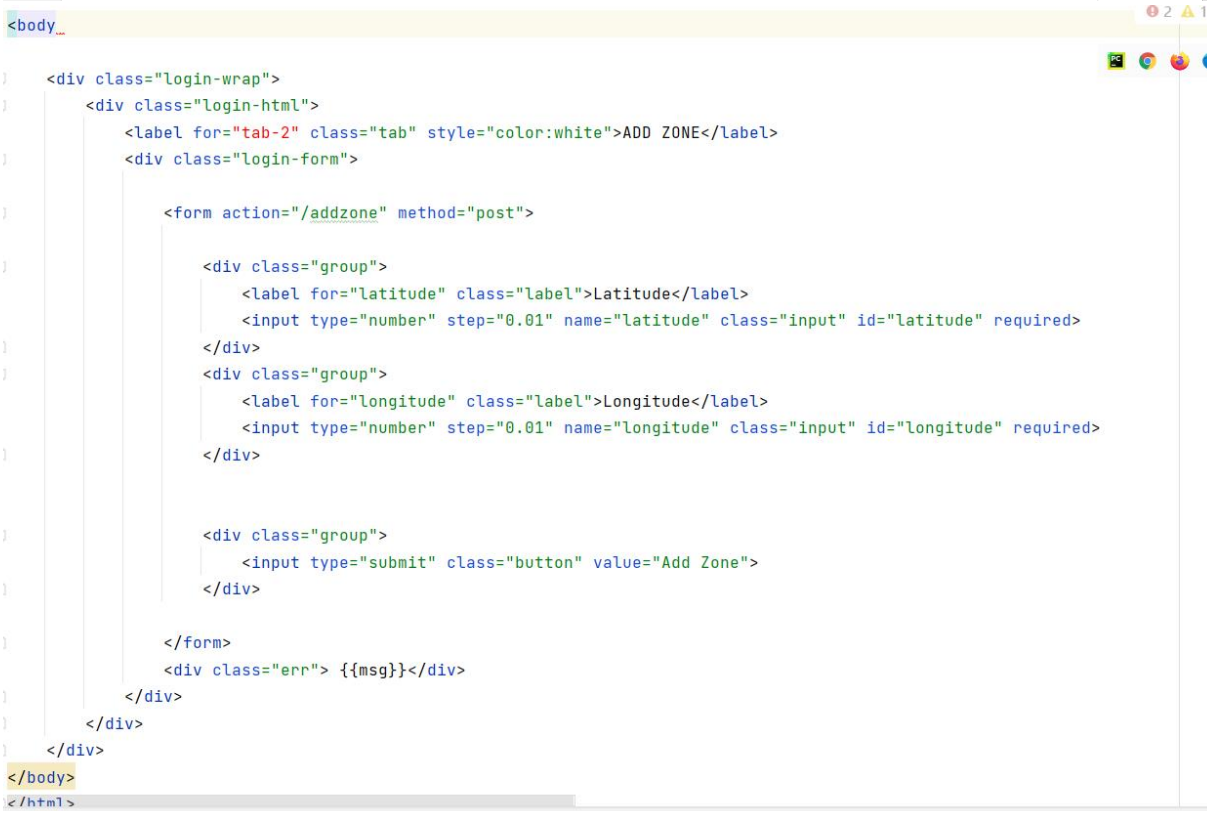
```html
</head>
<body >
    <div class="container-fluid px-1 px-md-5 px-lg-1 px-xl-5 py-5 mx-auto">
        <div class="card card0 border-0">
            <div class="row d-flex">
                <div class="col-lg-6">
                    <div class="card1 pb-5">
                        <div class="row px-3 justify-content-center mt-4 mb-5 border-line">
                          <h2 style="color:black;padding-top:150px;">CONTAINMENT <br>ZONE<br> ALERTING<br> APPLICATION</h2><br><br>

                        </div>
                    </div>
                </div>
                <div class="col-lg-6">
                    <form class="card2 card border-0 px-4 py-5" method="post" action="/ad_log">
                        <h2 style="padding-left:200px;color:black" class="mb-0 mr-4 mt-2">ADMIN LOGIN</h2>
                        <div class="row px-3">
                            <label class="mb-1"><h5 class="mb-0 text-sm">Enter your Email</h5></label>
                            <input class="mb-4" type="text" name="email">
                        </div>
                        <div class="row px-3">
                            <label class="mb-1"><h5 class="mb-0 text-sm">Enter valid Password</h5></label>
                            <input type="password" name="password">
                        </div>
```



```html
                        <div class="row mb-3 px-3">
                            <button type="submit" class="btn btn-primary text-center" href="addzone">Login</button>
                            <br>
                            <a href="/">Home</a>
                        </div>

                    </form>
                </div>
            </div>
            <div class="bg-black py-4">
                <div class="row px-3">

                    <div class="err"> {{error}}</div>
                    <div class="success"> {{success}}</div>

                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

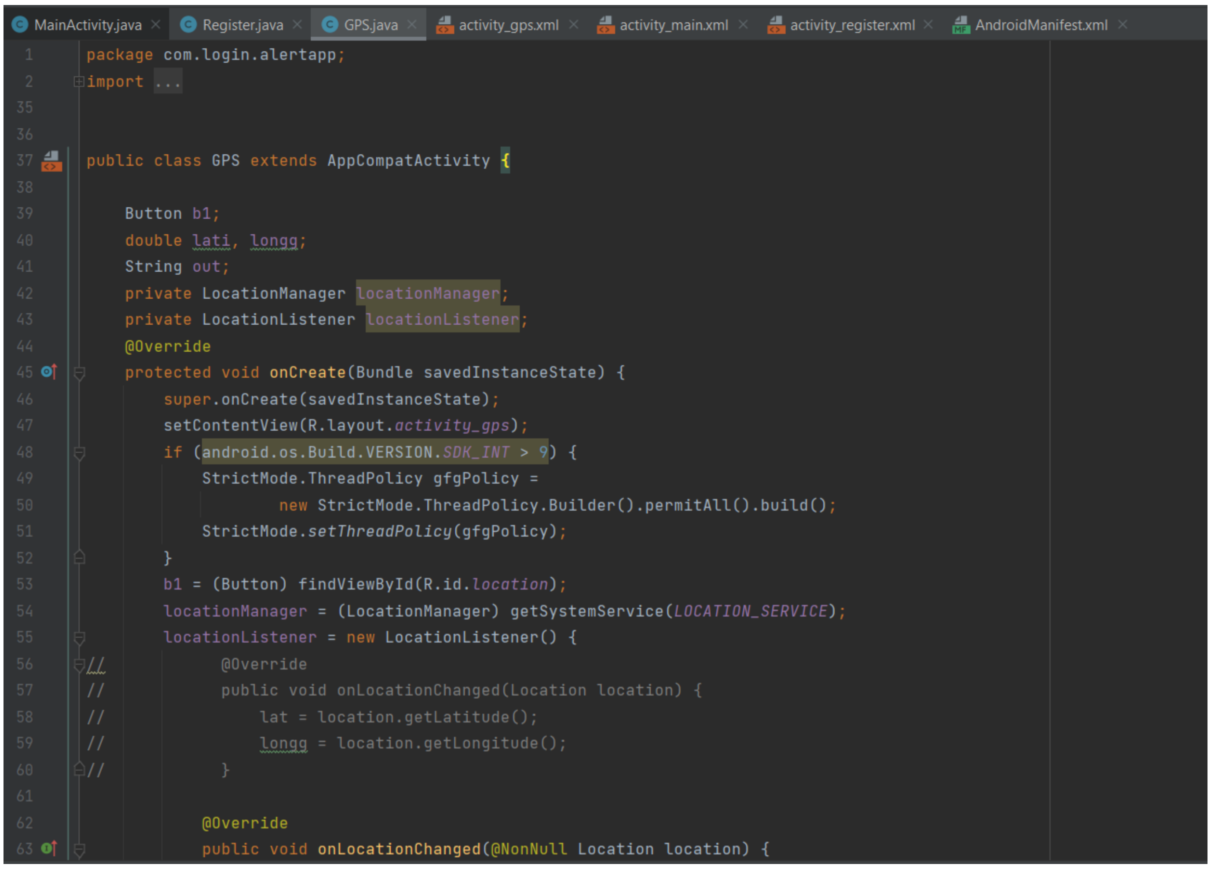


```html
</head>
<body>

    <div class="topnav">
       <h2 style="text-align:center;">ABOUT US</h2>
    </div>
    <div>
        <img src="static/imgs/cz.jfif">
    </div>
    <div class="content">
        <p>This Application is to provide service to ensure safety for people who are travelling from one place to another place.<br>
           This Application is intended to provide information about containment zones in a particular region by alerting people,
            through continuous monitoring of an individual's location.<br>  Key benefits of the application are monitoring people's
            activity and alerting them of their safety movements.</p>
    </div>
    <div class="quote">
        <h2 style="text-align:center;" >PREVENTION IS BETTER THAN CURE!!!</h2>
    </div>
</body>
</html>
```

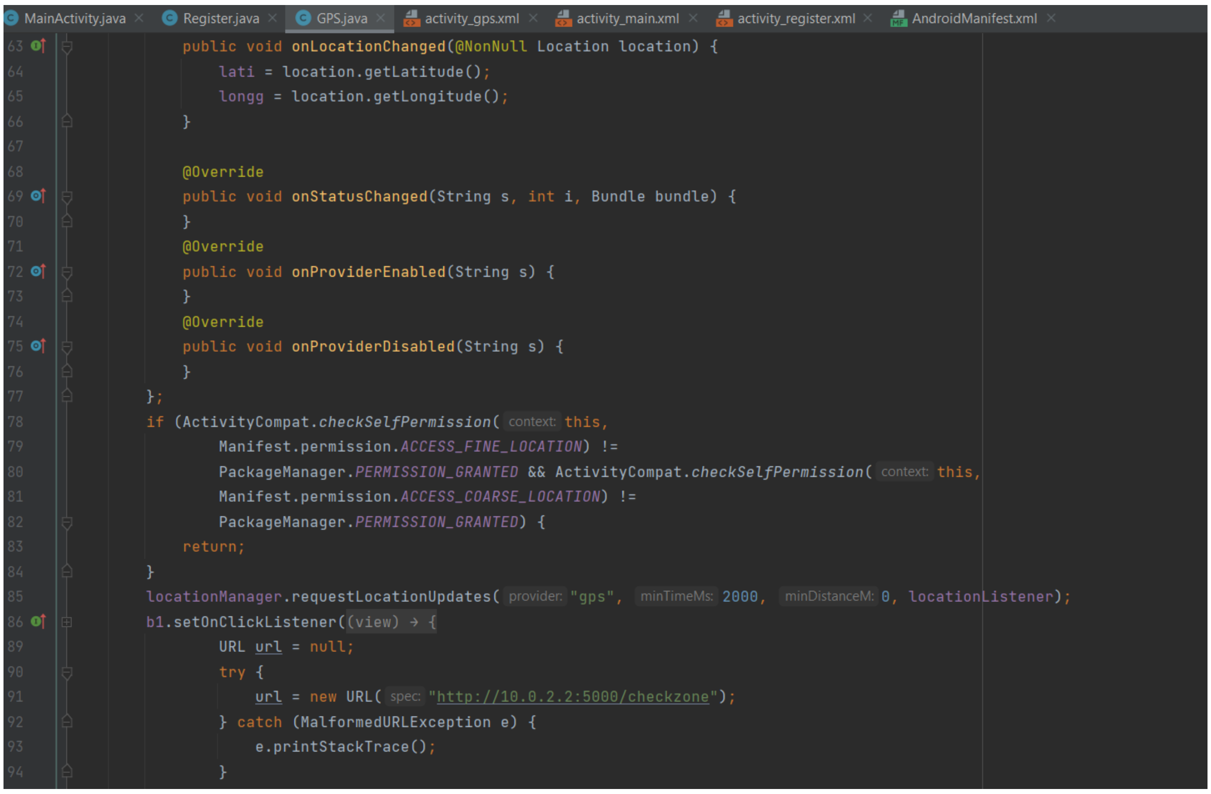


```html
<body

    <div class="login-wrap">
        <div class="login-html">
            <label for="tab-2" class="tab" style="color:white">ADD ZONE</label>
            <div class="login-form">

                <form action="/addzone" method="post">

                    <div class="group">
                        <label for="latitude" class="label">Latitude</label>
                        <input type="number" step="0.01" name="latitude" class="input" id="latitude" required>
                    </div>
                    <div class="group">
                        <label for="longitude" class="label">Longitude</label>
                        <input type="number" step="0.01" name="longitude" class="input" id="longitude" required>
                    </div>


                    <div class="group">
                        <input type="submit" class="button" value="Add Zone">
                    </div>

                </form>
                <div class="err"> {{msg}}</div>
            </div>
        </div>
    </div>
</body>
</html>
```

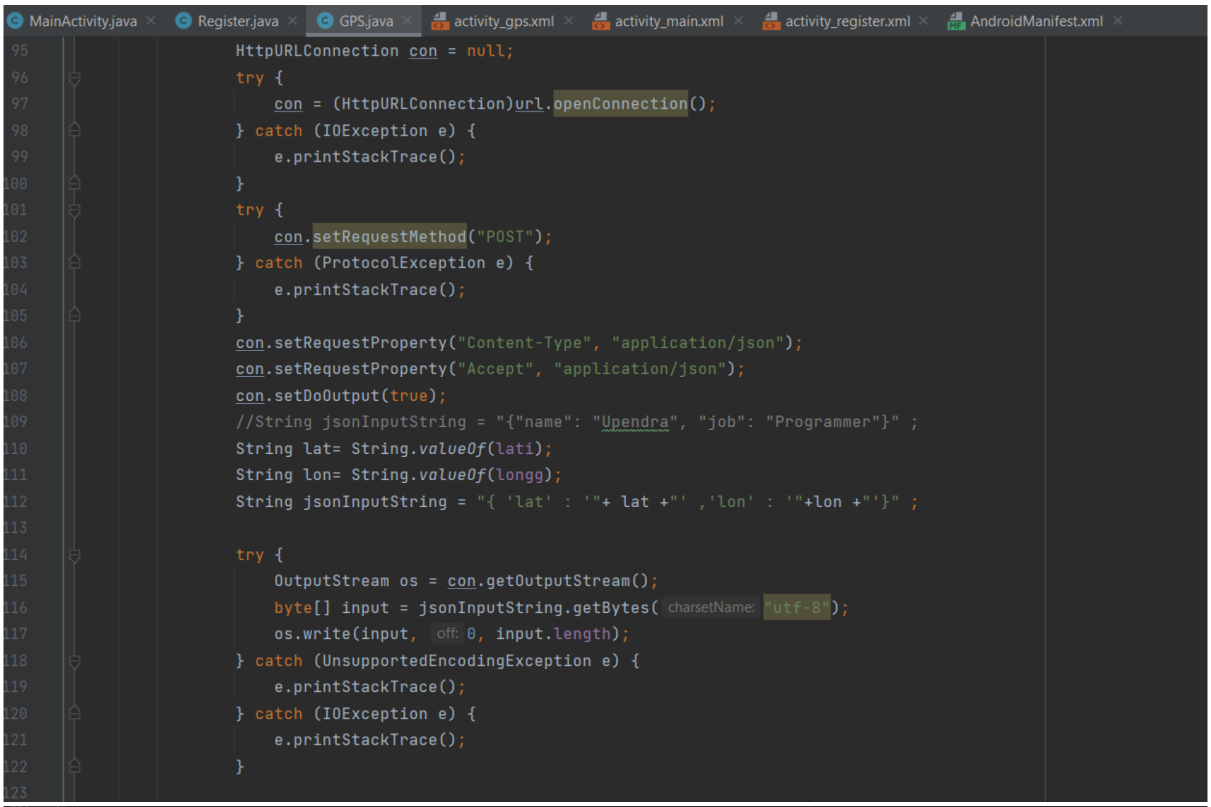
7.2 Feature -2



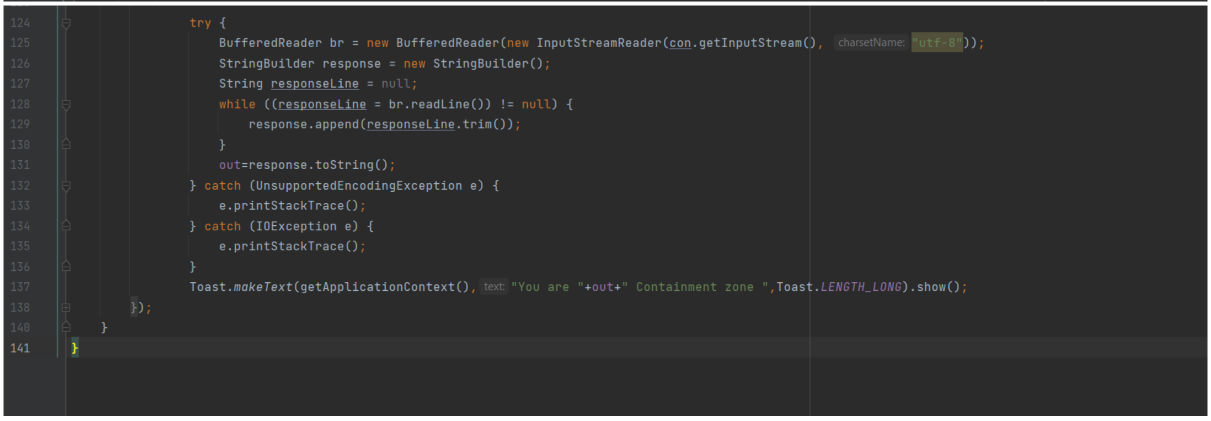
```java
package com.login.alertapp;
import ...


public class GPS extends AppCompatActivity {

    Button b1;
    double lati, longg;
    String out;
    private LocationManager locationManager;
    private LocationListener locationListener;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_gps);
        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(gfgPolicy);
        }
        b1 = (Button) findViewById(R.id.location);
        locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
        locationListener = new LocationListener() {
//            @Override
//            public void onLocationChanged(Location location) {
//                lat = location.getLatitude();
//                longg = location.getLongitude();
//            }

            @Override
            public void onLocationChanged(@NonNull Location location) {
```



```java
            public void onLocationChanged(@NonNull Location location) {
                lati = location.getLatitude();
                longg = location.getLongitude();
            }

            @Override
            public void onStatusChanged(String s, int i, Bundle bundle) {
            }
            @Override
            public void onProviderEnabled(String s) {
            }
            @Override
            public void onProviderDisabled(String s) {
            }
        };
        if (ActivityCompat.checkSelfPermission( context: this,
                Manifest.permission.ACCESS_FINE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission( context: this,
                Manifest.permission.ACCESS_COARSE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED) {
            return;
        }
        locationManager.requestLocationUpdates( provider: "gps",  minTimeMs: 2000,  minDistanceM: 0, locationListener);
        b1.setOnClickListener((view) → {
                URL url = null;
                try {
                    url = new URL( spec: "http://10.0.2.2:5000/checkzone");
                } catch (MalformedURLException e) {
                    e.printStackTrace();
                }
```

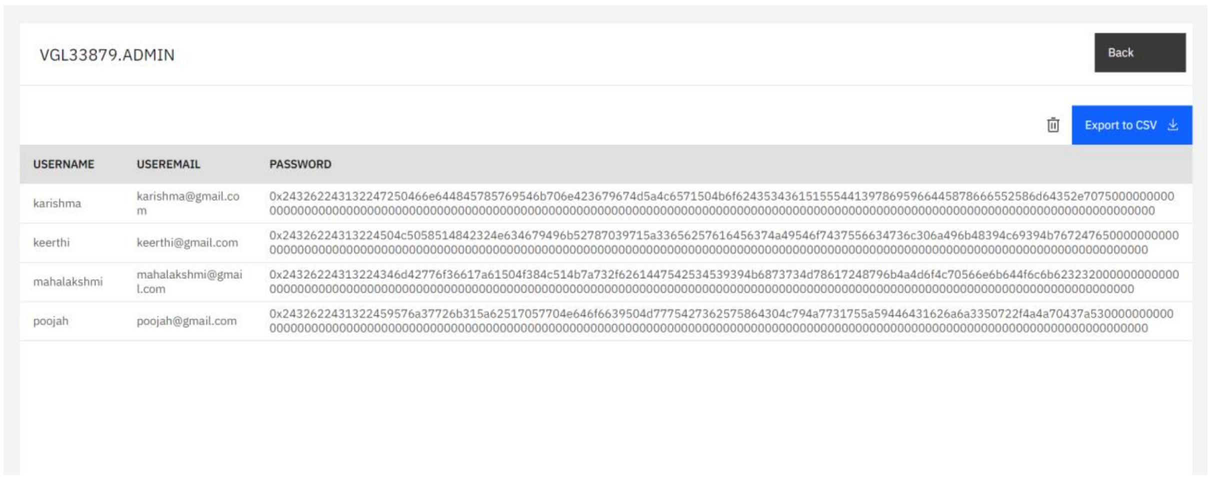


```
            HttpURLConnection con = null;
            try {
                con = (HttpURLConnection)url.openConnection();
            } catch (IOException e) {
                e.printStackTrace();
            }
            try {
                con.setRequestMethod("POST");
            } catch (ProtocolException e) {
                e.printStackTrace();
            }
            con.setRequestProperty("Content-Type", "application/json");
            con.setRequestProperty("Accept", "application/json");
            con.setDoOutput(true);
            //String jsonInputString = "{"name": "Upendra", "job": "Programmer"}" ;
            String lat= String.valueOf(lati);
            String lon= String.valueOf(longg);
            String jsonInputString = "{ 'lat' : '"+ lat +"' ,'lon' : '"+lon +"'}" ;

            try {
                OutputStream os = con.getOutputStream();
                byte[] input = jsonInputString.getBytes( charsetName: "utf-8");
                os.write(input,  off: 0, input.length);
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }

```



```
            try {
                BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream(),  charsetName: "utf-8"));
                StringBuilder response = new StringBuilder();
                String responseLine = null;
                while ((responseLine = br.readLine()) != null) {
                    response.append(responseLine.trim());
                }
                out=response.toString();
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
            Toast.makeText(getApplicationContext(),  text: "You are "+out+" Containment zone ",Toast.LENGTH_LONG).show();
        });
    }
}
```

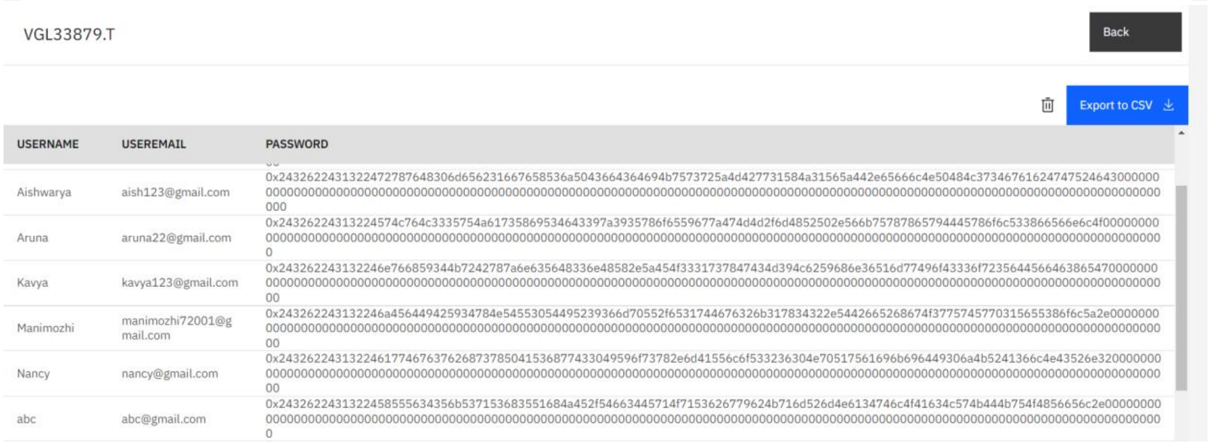
7.3 Database schema


VGL33879.ADMIN

Back

Export to CSV

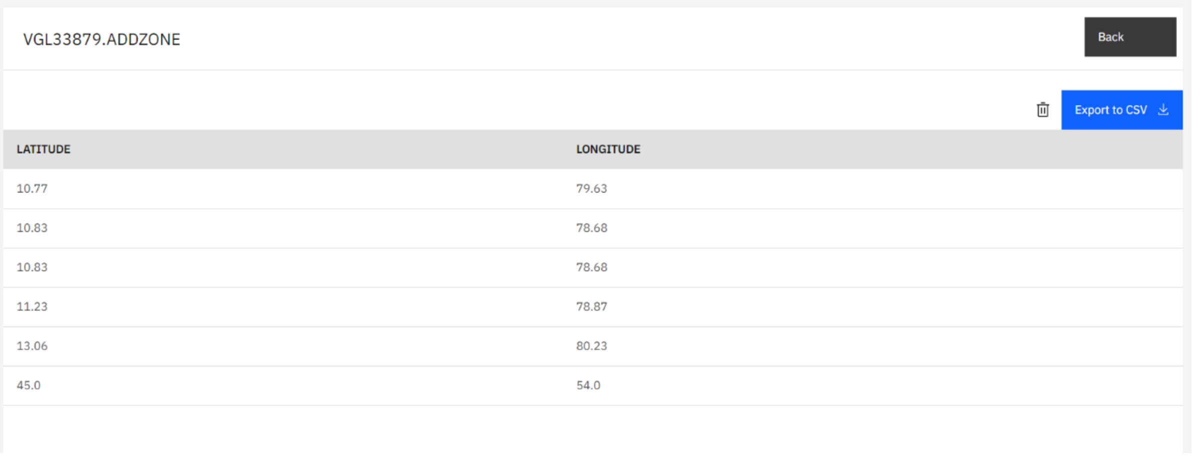
USERNAME	USEREMAIL	PASSWORD
karishma	karishma@gmail.com	0x243262243132247250466e644845785769546b706e423679674d5a4c6571504b6f62435343615155544139786959664458786665552586d64352e7075000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
keerthi	keerthi@gmail.com	0x24326224313224504c5058514842324e634679496b52787039715a3365625761645637 4a49546f7437556634736c306a496b48394c69394b7672476500000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
mahalakshmi	mahalakshmi@gmail.com	0x24326224313224346d42776f36617a61504f384c514b7a732f626144754253453939 4b6873734d78617248796b4a4d6f4c70566e6b644f6c6b62323200000000000000 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
poojah	poojah@gmail.com	0x243262243132245957 6a37726b315a62517057704e646f6639504d7775427362575864304c794a7731755a59446431626a6a3350722f4a4a70437a530000000000000 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000


VGL33879.T

Back

Export to CSV

USERNAME	USEREMAIL	PASSWORD
Aishwarya	aish123@gmail.com	0x2432622431322472787648306d656231667658536a5043664364694b7573725a4d427731584a31565a442e65666c4e50484c373467616247475246430000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 000
Aruna	aruna22@gmail.com	0x24326224313224574c764c3335754a61735869534643397a3935786f6559677a474d4d2f6d4852502e566b7578786579444 5786f6c533866566e6c4f00000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 0
Kavya	kavya123@gmail.com	0x243262243132246e766859344b7242787a6e635648336e48582e5a454f3331737847434d394c6259686e36516d77496f43336f72356445664638654700000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00
Manimozhi	manimozhi72001@gmail.com	0x243262243132246a456449425934784e54553054495239366d70552f6531744676326b317834322e5442665268674f3775745770315655386f6c5a2e0000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00
Nancy	nancy@gmail.com	0x2432622431322461774676376268737850415368774330495 96f73782e6d41556c6f533236304e70517561696b696449306a4b5241366c4e43526e320000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00
abc	abc@gmail.com	0x24326224313224585556343 56b537153683551684a452f54663445714f7153626779624b716d526d4e6134746c4f41634c574b444b754f4856656c2e00000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 0


VGL33879.ADDZONE

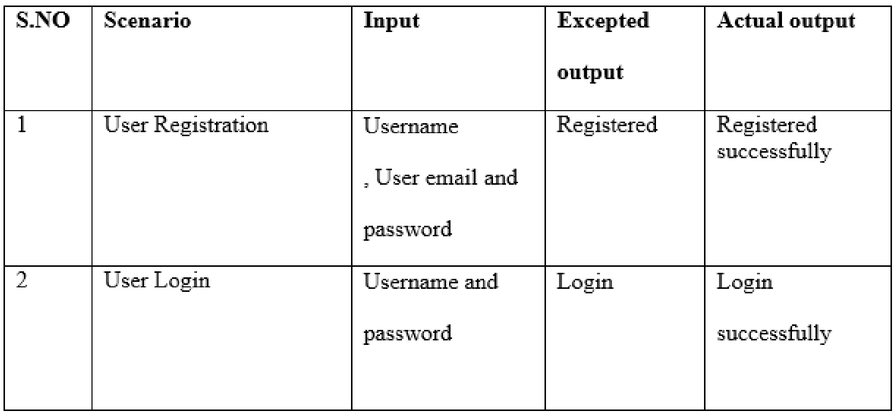
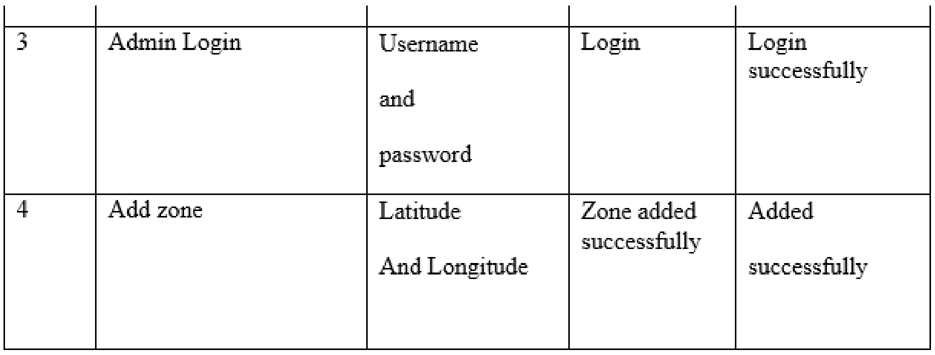
Back

Export to CSV

LATITUDE	LONGITUDE
10.77	79.63
10.83	78.68
10.83	78.68
11.23	78.87
13.06	80.23
45.0	54.0

8.Testing

8.1 Testcases

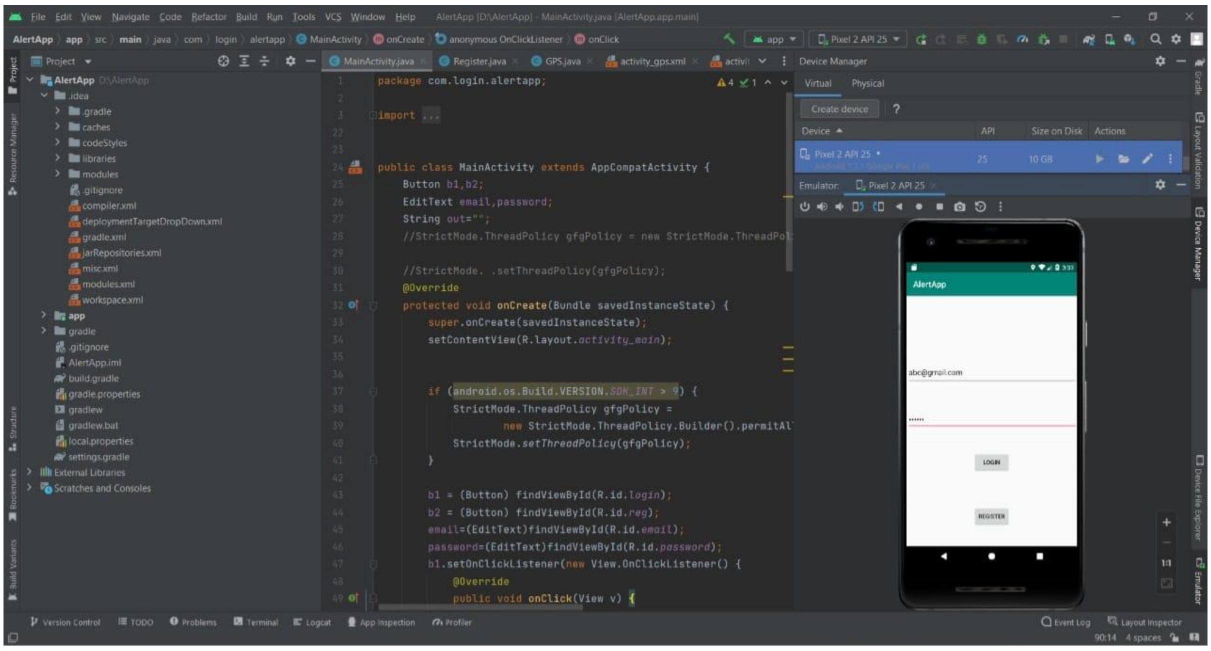
S.NO	Scenario	Input	Excepted output	Actual output
1	User Registration	Username , User email and password	Registered	Registered successfully
2	User Login	Username and password	Login	Login successfully
3	Admin Login	Username and password	Login	Login successfully
4	Add zone	Latitude And Longitude	Zone added successfully	Added successfully

8.2 USER ACCEPTANCE TESTING

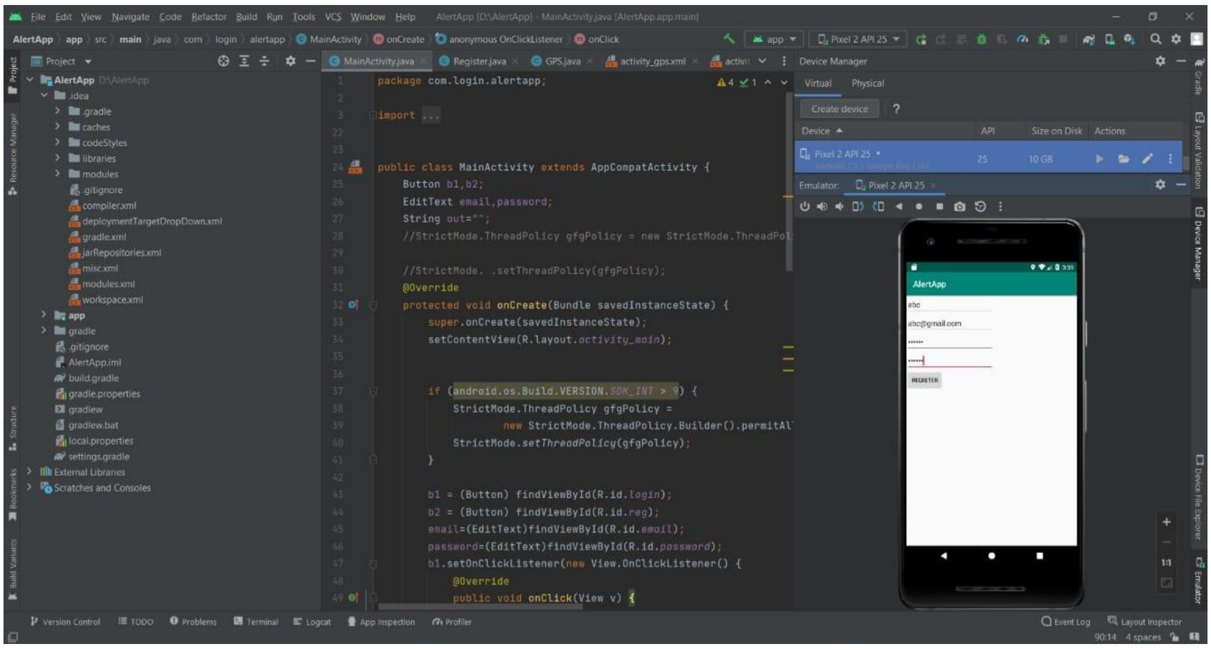
This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programme. It could entail the application's U.I., performance, usability,and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

9.RESULT

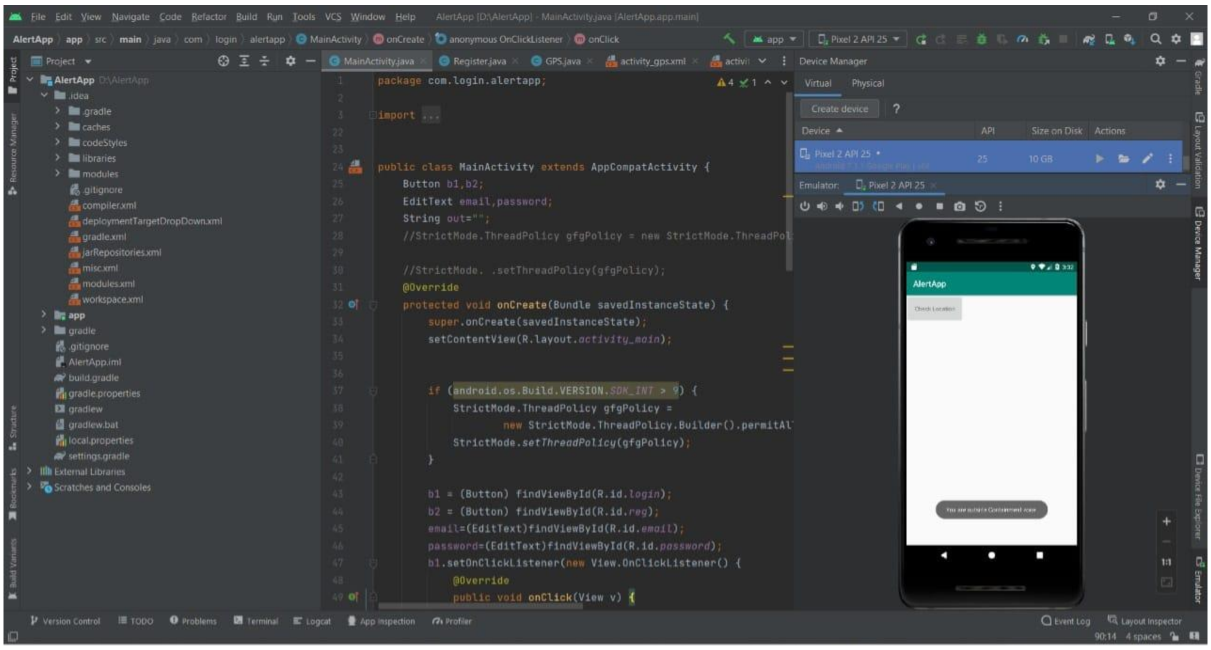
9.1 PERFORMANCE METRIX



```
package com.login.alertapp;

import ...


public class MainActivity extends AppCompatActivity {
    Button b1,b2;
    EditText email,password;
    String out="";
    //StrictMode.ThreadPolicy gfgPolicy = new StrictMode.ThreadPol

    //StrictMode. .setThreadPolicy(gfgPolicy);
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new StrictMode.ThreadPolicy.Builder().permitAl
            StrictMode.setThreadPolicy(gfgPolicy);
        }

        b1 = (Button) findViewById(R.id.login);
        b2 = (Button) findViewById(R.id.reg);
        email=(EditText)findViewById(R.id.email);
        password=(EditText)findViewById(R.id.password);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```



```
package com.login.alertapp;

import ...


public class MainActivity extends AppCompatActivity {
    Button b1,b2;
    EditText email,password;
    String out="";
    //StrictMode.ThreadPolicy gfgPolicy = new StrictMode.ThreadPol

    //StrictMode. .setThreadPolicy(gfgPolicy);
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new StrictMode.ThreadPolicy.Builder().permitAl
            StrictMode.setThreadPolicy(gfgPolicy);
        }

        b1 = (Button) findViewById(R.id.login);
        b2 = (Button) findViewById(R.id.reg);
        email=(EditText)findViewById(R.id.email);
        password=(EditText)findViewById(R.id.password);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```



```java
package com.login.alertapp;

import ...


public class MainActivity extends AppCompatActivity {
    Button b1,b2;
    EditText email,password;
    String out="";
    //StrictMode.ThreadPolicy gfgPolicy = new StrictMode.ThreadPol

    //StrictMode. .setThreadPolicy(gfgPolicy);
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new StrictMode.ThreadPolicy.Builder().permitAl
            StrictMode.setThreadPolicy(gfgPolicy);
        }

        b1 = (Button) findViewById(R.id.login);
        b2 = (Button) findViewById(R.id.reg);
        email=(EditText)findViewById(R.id.email);
        password=(EditText)findViewById(R.id.password);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

10.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

1. SOME SUCCESS IN BREAKING CHAINS OF INFECTION

Scientists haven’t given up on contact tracing apps, despite being skeptical of their overall effectiveness. Countries like South Korea have shown that this technology may be effective at containing the virus – albeit at the cost of user privacy, as we’ll see in the cons section.

Germany and Ireland have some of the highest app adoption rates in the world, although it’s still not enough to be fully effective. Nevertheless, spokespeople say that every life saved as a result of using a contact tracing app is a success in their book.

2. FASTER MEANS OF ALERTING PEOPLE

Exposure notifications are delivered more effectively if the process is automatic. It makes sense – rather than having to call each person individually, an app may alert everyone that’s been in close proximity to an infected person. Provided each party uses the app, that is.

It’s also better than relying solely on people’s memory of where they’ve been and who they came in contact with. Naturally, digital contact tracing and doing things “the old fashioned way” aren’t mutually exclusive. Not everyone will have the app installed on their phone – or even own a smartphone, for that matter.

As such, health organizations still have to rely on old techniques to prevent further outbreaks. At the very least, the apps have somewhat eased the job of on-the-field contact tracers (see South Korea’s Immediate Response Teams).

3. PEACE OF MIND

During times of turmoil, anything that can offer some solace is a welcome sight. Knowing that there are millions of people out there who are doing something to stop the spread of Covid-19 can be a calming thought. Obviously, people shouldn't let contact tracing apps become psychological crutches. Nor should it deter them from following basic anti-Covid measures like wearing a mask in public.

DISADVANTAGES

1. PRIVACY ISSUES

As mentioned before, South Korea was one of the biggest successes for digital contact tracing. At the same time, it was one of the worst offenders on the privacy front (short of China and similar iron-fisted approaches). You can get a good idea about the app's intrusiveness from the fact that it managed to expose cheating partners, among other embarrassing details about infected people's lives.

Of course, South Korea's app is built with location tracking in mind. Less privacy-invasive solutions exist, such as those that use Bluetooth or ultrasonic technology.

Now, disagreements exist even among Bluetooth proponents, mostly regarding the centralisation vs. decentralisation of data. For the most part, though, academics agree that moving away from location-based contact tracing apps is essential for user privacy.

2. SECURITY FLAWS

Months after the start of the pandemic, many contact tracing apps are still a work in progress. The reason? Many governments and app developers rushed out their solutions in a move some call "do-something-itis."

A previous version of the UK NHS's app which was rife with security issues was abandoned in favor of a new one based on the framework built by Google and Apple. A framework that isn't as airtight as the two tech giants initially claimed, it seems. At least these issues are being patched out as they're discovered. App users are advised to keep everything up-to-date, to avoid any unpleasant surprises.

3. EXCLUSIONARY

Another problem with keeping things digital is that not everybody has the means of using the apps. China's Alipay Health Code app has sparked debate on social media due to a video of a senior citizen losing his cool for being unable to show his QR code.

If you aren't familiar, China's app uses a QR color-coding system to provide access to public services. Only green code users benefit from relative freedom of movement, while yellow and red users face restrictions and isolation orders.

Elsewhere, in India, people from poor areas can't even afford smartphones to run Aarogya Setu. The government-produced app was initially mandatory for everyone. After weeks of public backlash, it was eventually decided that only employees and containment zone inhabitants would need to use the app. The app's status has been pretty volatile, so conflicting reports may exist.

In the end – and despite some scattered successes – the public view of digital contact tracing isn't too rosy. Concerns that apps may be used for mass surveillance after the pandemic aren't helping their case, either.

11) CONCLUSION AND FUTURE SCOPE

CONCLUSION:

This application is intended to provide information about containment zones in a particular region by alerting people, through continuous monitoring of an individuals location. Key benefits of the application are monitoring peoples activity and alerting them to their safety movements

FUTURE SCOPE:

Although we tried to cover almost all of the aspects during our developmental phase, however we were forced to leave some aspects because of lack of time as well as monetary and other reasons. Just like in the field of software development where there are always some shortcomings and room for improvement our application can be enhanced further: -

1) The application can include various government organization to help act faster.

2) The dataset obtained from the application can be used for predictive analysis to determine prone areas and include special method for tackling the problem in those areas.

3) Emergency signal in case of network failure and internet connection loss.

4) Tackling victim's movements.

5) Improved Google positioning system's precision.

6) The client part of application can be integrated in a single intelligent device.

For analysis purpose, we could use machine learning (ML) algorithms as well as data mining applications. There is a sub branch of machine learning known as time series analysis (TSA), which could be used to predict and analyze the data obtained through this application. Time series analysis is used to predict crop production as well as sales in different quarter.

12.APPENDIX

SOUCE CODE

```
from flask import Flask, render_template, request, redirect,
url_for,session,jsonify
import json as j
import ibm_db
import bcrypt
from functools import partial
import pyproj
from shapely.ops import transform
from shapely.geometry import Point

proj_wgs84 = pyproj.Proj('+proj=longlat +datum=WGS84')

try:
    conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=125f9f61-9715-46f9-9399-
c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30426;SECU
RITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID=v
gl33879;PWD=qXLGPqTBwwNxG6bR",'','')
    print(conn)
    print("connection successfull")
except:
    print("Error in connection, sqlstate = ")
    errorState = ibm_db.conn_error()
    print(errorState)
app = Flask(__name__)
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'

#HOME
@app.route("/", methods=['GET'])
def home():
    #if 'email' not in session:
        #return redirect(url_for('login'))
    return render_template('home.html', name='Home')

@app.route("/")

#USER REGISTER
@app.route("/register", methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        stemp = str(request.data)
        stemp = stemp[2:len(stemp) - 1]
        stemp = stemp.replace("'", '"')
        data = j.loads(stemp)
        name=data['name']
        email = data['email']
        password = data['pass']
        cpassword = data['cpass']

        if not email or not name or not password or not cpassword:
            return 'Please fill all fields'
        if password != cpassword:
            return 'The password is not same'
        else:
            hash = bcrypt.hashpw(password.encode('utf-8'),
bcrypt.gensalt())

        query = "SELECT * FROM T WHERE useremail=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)

        if not isUser:
            insert_sql = "INSERT INTO T(USERNAME, USEREMAIL, PASSWORD)
VALUES (?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, name)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, hash)
            ibm_db.execute(prep_stmt)
            return "You can login"
        else:
            return 'Invalid Credentials'

    return render_template('register.html')


#USER LOGIN
@app.route("/login", methods=['GET', 'POST'])
def login():

    if request.method == 'POST':
        stemp=str(request.data)
        print(stemp)
        stemp=stemp[2:len(stemp)-1]
        stemp=stemp.replace("'",'"')
        data=j.loads(stemp)
        email = data['email']
        password = data['pass']
        print(email , password)
        if not email or not password:
            return 'PLEASE FILL ALL FIELDS'
        query = "SELECT * FROM T WHERE useremail=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)
        print(isUser, password)

        if not isUser:
            return "Invalid Username"
            #return render_template('login.html', error='INVALID USERNAME
OR PASSWORD')
        # return render_template('login.html',error=isUser['PASSWORD'])

        isPasswordMatch = False
        temp=str(isUser['PASSWORD'])
        temp=temp.replace("\x00",'')
        temp=temp[2:len(temp)-1]
        temp = temp.encode("utf-8")
        check = bcrypt.hashpw(password.encode('utf-8'),temp)
        #print(check==temp[:len(check)])
        if check==temp[:len(check)]:
            isPasswordMatch=True
        #isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),
isUser['PASSWORD'])

        if not isPasswordMatch:
            return "Invalid Credentials"
            #return render_template('login.html', error='Invalid
Credentials')

        session['email'] = isUser['USEREMAIL']
        return "valid"
    return render_template('login.html', name='Home')


#ADMIN REGISTER
@app.route("/ad_reg", methods=['GET', 'POST'])
def ad_reg():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        cpassword = request.form['cpassword']

        if not email or not name or not password or not cpassword:
            return render_template('ad_reg.html', error='Please fill all
fields')
        if password != cpassword:
            return render_template('ad_reg.html', error='The password is
not same')
        else:
            hash = bcrypt.hashpw(password.encode('utf-8'),
bcrypt.gensalt())
            print(type(hash))
            print(hash)
            #encpass=hash.decode()
            #print(encpass)
            #print(type(encpass))


        query = "SELECT * FROM ADMIN WHERE useremail=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)

        if not isUser:
            insert_sql = "INSERT INTO ADMIN(USERNAME, USEREMAIL, PASSWORD)
VALUES (?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, name)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, hash)
            ibm_db.execute(prep_stmt)
            return render_template('ad_reg.html', success="You can login")
        else:
            return render_template('ad_reg.html', error='Invalid
Credentials')

    return render_template('ad_reg.html')



#ADMIN LOGIN
```

```
@app.route("/ad_log", methods=['GET', 'POST'])
def ad_log():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        if not email or not password:
            return render_template('ad_log.html', error='PLEASE FILL ALL
FIELDS')

        query = "SELECT * FROM ADMIN WHERE useremail=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)

        ibm_db.execute(stmt)
        isAdmin = ibm_db.fetch_assoc(stmt)
        print(isAdmin, password)

        if not isAdmin:
            return render_template('ad_log.html', error='INVALID USERNAME
OR PASSWORD')
        # return render_template('login.html',error=isUser['PASSWORD'])

        isPasswordMatch = False
        temp=str(isAdmin['PASSWORD'])
        temp=temp.replace("\x00",'')
        temp=temp[2:len(temp)-1]
        print(temp)
        temp = temp.encode("utf-8")
        print(temp,type(temp))
        check = bcrypt.hashpw(password.encode('utf-8'),temp)
        #print(check==temp[:len(check)])
        if check==temp[:len(check)]:
            isPasswordMatch=True
        #isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),
isUser['PASSWORD'])

        if not isPasswordMatch:
            return render_template('ad_log.html', error='Invalid
Credentials')

        session['email'] = isAdmin['USEREMAIL']
        return redirect(url_for('addzone'))

    return render_template('ad_log.html', name='Home')


#ADD ZONE
@app.route('/addzone', methods=['GET', 'POST'])
def addzone():

    if request.method=='POST':
        latitude=request.form['latitude']
        longitude = request.form['longitude']

        if not latitude or not longitude:
            return render_template('addzone.html', msg='Please fill all
fields')
        sql = "INSERT INTO ADDZONE(LATITUDE,LONGITUDE) VALUES(?,?)"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, latitude)
        ibm_db.bind_param(stmt, 2, longitude)
        ibm_db.execute(stmt)
        return render_template('addzone.html',msg='zone added sucessfully')
    else:
        return render_template('addzone.html')

#Check ZONE
@app.route('/checkzone', methods=['GET', 'POST'])
def checkzone():
    if request.method == 'POST':
        stemp = str(request.data)
        stemp = stemp[2:len(stemp) - 1]
        stemp = stemp.replace("'", '"')
        data = j.loads(stemp)
        ulat=float(data['lat'])
        ulon=float(data['lon'])
        query = "SELECT * FROM ADDZONE"
        stmt = ibm_db.exec_immediate(conn,query)
        tuple = ibm_db.fetch_tuple(stmt)
        anslat=[]
        anslon=[]
        alat=[]
        alon=[]
        while tuple != False:
            lat=tuple[0]
            lon=tuple[1]
            km = 10
            # Azimuthal equidistant projection
            aeqd_proj = '+proj=aeqd +lat_0={lat} +lon_0={lon} +x_0=0
+y_0=0'
            project = partial(
                pyproj.transform,
                pyproj.Proj(aeqd_proj.format(lat=lat, lon=lon)),
                proj_wgs84)
            buf = Point(0, 0).buffer(km * 1000)  # distance in metres
            b = transform(project, buf).exterior.coords[:]
            for i in b:
                alat.append(i[0])
                alon.append(i[1])
            anslon.append([min(alat),max(alat)])
            anslat.append([min(alon),max(alon)])
            tuple = ibm_db.fetch_tuple(stmt)
        print(anslat,anslon)
        print(ulat,ulon)
        for i in range(len(anslat)):

#print(ulat,anslat[i][0],ulat,anslat[i][1],ulat,anslon[i][0],ulat,anslon[i]
[1])

if((ulat>=anslat[i][0])and(ulat<=anslat[i][1])and(ulon>=anslon[i][0])and(ul
on<=anslon[i][1])):
                return "inside"

        return 'outside'
    else:
        return render_template('addzone.html')

@app.route('/about')
def about():
    return render_template("about.html")
```

```python
@app.route('/logout')
def logout():
    session.pop('email', None)
    return redirect(url_for('login'))


if __name__ == "__main__":
    app.run(debug=True)
```

```java
package com.login.alertapp;

import androidx.appcompat.app.AppCompatActivity;


import android.os.Bundle;
import android.content.Intent;
import android.os.StrictMode;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.ProtocolException;
import java.net.URL;


public class MainActivity extends AppCompatActivity {
    Button b1,b2;
    EditText email,password;
    String out="";
    //StrictMode.ThreadPolicy gfgPolicy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();

    //StrictMode. .setThreadPolicy(gfgPolicy);
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new
StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(gfgPolicy);
        }

        b1 = (Button) findViewById(R.id.login);
        b2 = (Button) findViewById(R.id.reg);
        email=(EditText)findViewById(R.id.email);
        password=(EditText)findViewById(R.id.password);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    //Toast.makeText(getApplicationContext(), "Hi",
Toast.LENGTH_LONG).show();
                    URL  url = new URL("http://10.0.2.2:5000/login");
                    HttpURLConnection con =
(HttpURLConnection)url.openConnection();
                    con.setRequestMethod("POST");
                    con.setRequestProperty("Content-Type",
"application/json");
                    con.setRequestProperty("Accept", "application/json");
                    con.setDoOutput(true);
                    //String jsonInputString = "{"name": "Upendra", "job":
"Programmer"}" ;
                    String eml=email.getText().toString();
                    String pass=password.getText().toString();
                    String jsonInputString = "{ 'email' : '"+ eml +"' ,
'pass': '"+ pass +"' }" ;
                    OutputStream os = con.getOutputStream();
                    byte[] input = jsonInputString.getBytes("utf-8");
                    os.write(input, 0, input.length);
                    BufferedReader br = new BufferedReader(new
InputStreamReader(con.getInputStream(), "utf-8"));
                    StringBuilder response = new StringBuilder();
                    String responseLine = null;
                    while ((responseLine = br.readLine()) != null) {
                        response.append(responseLine.trim());
                    }
                    out = response.toString();
                } catch (ProtocolException e) {
                    e.printStackTrace();
                } catch (MalformedURLException e) {
                    e.printStackTrace();
                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                }
                //Toast.makeText(getApplicationContext(),"check"+ out,
Toast.LENGTH_LONG).show();
                if (out.equalsIgnoreCase("valid")) {
                    Intent in= new
Intent(getApplicationContext(),GPS.class);
                    startActivity(in);
                    Toast.makeText(getApplicationContext(),"Find Your
Location",Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(getApplicationContext(), out,
Toast.LENGTH_LONG).show();
                }


            }
        });

        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent in = new Intent(getApplicationContext(),
Register.class);
                startActivity(in);
            }
```

```java
        });
    }

package com.login.alertapp;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.os.StrictMode;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.ProtocolException;
import java.net.URL;

public class Register extends AppCompatActivity {

    Button b1;
    EditText username,email,password,cpassword;
    String out;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new
StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(gfgPolicy);
        }
        username=(EditText)findViewById(R.id.regun);
        email=(EditText)findViewById(R.id.emailid);
        password=(EditText)findViewById(R.id.password);
        cpassword=(EditText)findViewById(R.id.passcp);
        b1 = (Button) findViewById(R.id.reg);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                URL url = null;
                try {
                    url = new URL("http://10.0.2.2:5000/register");
                } catch (MalformedURLException e) {
                    e.printStackTrace();
                }
                HttpURLConnection con = null;
                try {
                    con = (HttpURLConnection) url.openConnection();
                } catch (IOException e) {
                    e.printStackTrace();
                }
                try {
                    con.setRequestMethod("POST");
                } catch (ProtocolException e) {
                    e.printStackTrace();
                }
                con.setRequestProperty("Content-Type", "application/json");
                con.setRequestProperty("Accept", "application/json");
                con.setDoOutput(true);
                //String jsonInputString = "{"name": "Upendra", "job":
"Programmer"}" ;
                String n = username.getText().toString();
                String eml = email.getText().toString();
                String pass = password.getText().toString();
                String cpass = cpassword.getText().toString();
                String jsonInputString = "{ 'name' : '" + n + "' ,'email' :
'" + eml + "' , 'pass': '" + pass + "' , 'cpass': '" + cpass + "'}";

                try {
                    OutputStream os = con.getOutputStream();
                    byte[] input = jsonInputString.getBytes("utf-8");
                    os.write(input, 0, input.length);
                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                }

                try {
                    BufferedReader br = new BufferedReader(new
InputStreamReader(con.getInputStream(), "utf-8"));
                    StringBuilder response = new StringBuilder();
                    String responseLine = null;
                    while ((responseLine = br.readLine()) != null) {
                        response.append(responseLine.trim());
                    }
                    out = response.toString();
                } catch (IOException e) {
                    e.printStackTrace();
                }

                if (out == "You can login") {
                    Intent in = new Intent(getApplicationContext(),
MainActivity.class);
                    startActivity(in);
                    Toast.makeText(getApplicationContext(), out,
Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(getApplicationContext(), out,
Toast.LENGTH_LONG).show();
                }
            }
        });
    }
}


package com.login.alertapp;
import android.Manifest;
import android.annotation.TargetApi;
import android.content.DialogInterface;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.ProtocolException;
import java.net.URL;


import java.util.ArrayList;

import static android.Manifest.permission.ACCESS_COARSE_LOCATION;
import static android.Manifest.permission.ACCESS_FINE_LOCATION;


public class GPS extends AppCompatActivity {

    Button b1;
    double lati, longg;
    String out;
    private LocationManager locationManager;
    private LocationListener locationListener;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_gps);
        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy gfgPolicy =
                    new
StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(gfgPolicy);
        }
        b1 = (Button) findViewById(R.id.location);
        locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
        locationListener = new LocationListener() {
//            @Override
//            public void onLocationChanged(Location location) {
//                lat = location.getLatitude();
//                longg = location.getLongitude();
//            }

            @Override
            public void onLocationChanged(@NonNull Location location) {
                lati = location.getLatitude();
                longg = location.getLongitude();
            }
```

```java
            @Override
            public void onStatusChanged(String s, int i, Bundle bundle) {
            }
            @Override
            public void onProviderEnabled(String s) {
            }
            @Override
            public void onProviderDisabled(String s) {
            }
        };
        if (ActivityCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_FINE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_COARSE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED) {
            return;
        }
        locationManager.requestLocationUpdates("gps", 2000, 0,
locationListener);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                URL url = null;
                try {
                    url = new URL("http://10.0.2.2:5000/checkzone");
                } catch (MalformedURLException e) {
                    e.printStackTrace();
                }
                HttpURLConnection con = null;
                try {
                    con = (HttpURLConnection)url.openConnection();
                } catch (IOException e) {
                    e.printStackTrace();
                }
                try {
                    con.setRequestMethod("POST");
                } catch (ProtocolException e) {
                    e.printStackTrace();
                }
                con.setRequestProperty("Content-Type", "application/json");
                con.setRequestProperty("Accept", "application/json");
                con.setDoOutput(true);
                //String jsonInputString = "{"name": "Upendra", "job":
"Programmer"}" ;
                String lat= String.valueOf(lati);
                String lon= String.valueOf(longg);
                String jsonInputString = "{ 'lat' : '"+ lat +"' ,'lon' :
'"+lon +"'}" ;

                try {
                    OutputStream os = con.getOutputStream();
                    byte[] input = jsonInputString.getBytes("utf-8");
                    os.write(input, 0, input.length);
                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                }
                try {
                    BufferedReader br = new BufferedReader(new
InputStreamReader(con.getInputStream(), "utf-8"));
                    StringBuilder response = new StringBuilder();
                    String responseLine = null;
                    while ((responseLine = br.readLine()) != null) {
                        response.append(responseLine.trim());
                    }
                    out=response.toString();
                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                }
                Toast.makeText(getApplicationContext(),"You are "+out+"
Containment zone ",Toast.LENGTH_LONG).show();
            }
        });
    }
```

DEMO LINK

https://www.kapwing.com/videos/63790a7044ac05041b260a8b