CONTAINMENT ZONE ALERTING APPLICATION

PROJECT REPORT

Submitted by

Anima U (920419104007)

Jainul Mufafika H (920419104034)

Preethika V (920419104068)

ThangaTamilSelvi S (920419104091)

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE &ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE &ENGINEERING

KAMARAJ COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution - Affiliated to Anna University, Chennai)

K.VELLAKULAM - 625 701 (Near Virudhunagar)

CHAPTER 1

INTRODUCTION

A containment zone alerting application is an application that sends alerts to users when they enter or exit a containment zone. It uses GPS to track the user's location and sends an alert if the user enters or leaves a containment zone. It also allows users to set up alerts for specific containment zones.

1.1 PROJECT OVERVIEW

The World Health Organization has declared the outbreak of the novel coronavirus, COVID19 as pandemic across the world. With its alarming surge of affected cases throughout the world, lockdown and awareness (social distancing, use of masks etc) among people are found to be the only means for restricting the community transmission. In a densely populated country like India, it is very difficult to prevent the community transmission even during lockdown without social awareness and precautionary measures taken by the people. Recently, several containment zones had been identified throughout the country and divided into red, orange and green zones, respectively. The red zones indicate the infection hotspot, orange zones denote some infection and green zones indicate an area with no infection. This paper mainly focuses on development of an Android application which can inform people of the COVID-19 containment zones and prevent trespassing into these zones.

1.2 PURPOSE

Provide information about containment zones in a particular region by alerting people, through continuous monitoring of an individual's location. This Android application updates the locations of the areas in a Google map which are identified to be the containment zones. The application also notifies the users if they have entered a containment zone and uploads the user’s info to the online database. To achieve all these functionalities, many tools and APIs from Google like Firebase and Geofence are used in this app. Therefore, this application can be used as a tool for creating further social awareness about the arising need of precautionary measures to be taken by the people of India.

CHAPTER 2

LITERATURE SURVEY

1. Social Distance Alert System to Control Virus Spread using UWB RTLS incorporate Environments

The author proposed a method to develop a real-time location system (RTLS) based on ultrawideband (UWB) wireless technology that gives the most accurate locations of approximately 10cm using methods like trilateration and TDOA (Time Difference of Arrival). Coordinates of the location can be obtained by installing RTLS in predefined areas which are used to calculate the distance between Mobile UWB Devices (MUD's). An alert triggered by a system to maintain distance if distance between the employees is less than the prescribed social distance can keep the work premises safe and control the spread of coronavirus. This can be a great solution to control the spread of corona virus. This study can be a great solution to control the spread of virus in corporate working environments which are mostly confined in size and indoor in nature.

2. A Detection, Tracking and Alerting System for Covid-19 using Geo-Fencing and Machine Learning

The author proposed a complete Covid-19 Detection, Tracking and Alerting Mobile Application Kit which helps people to defend against Covid-19 spread. This is a first of its kind application that uses Geofencing and Machine learning together to combat the spread of Coronavirus. This app is a threefold app. The first fold is a Detection System for a user to undergo a Symptomatic Quiz based on a Risk Assessment ML Model to detect the presence of Covid in the user's body. The second fold is an efficient Tracking system that uses Geofencing technology to keep track of all the people who come into contact with the user. And the third fold is an Alerting system that sends the alert message to all those people who came into contact with the user if the user is tested as Corona positive. Thus, by using the technology, Geofencing allows to perform contact tracing of potential patients and alerts the possible network of people who might be infected by coronavirus.

3. Android Application based Smart Bus Transportation System for Pandemic Situations

Smart Bus Transportation System was introduced which guides the passengers in booking the bus tickets using the Android Application and it also helps the passengers to keep an update on bus location based on their request. This system also sends alert message few minutes in advance to the passengers before the bus reaches the passengers boarding point. This system also sends the precautionary instruction priory the passengers that have to be followed while traveling in the bus. In order to provide additional safety to the passengers the temperature of the passengers is monitored and intimated to the bus in change before they are permitted into the bus.

4. Social Distancing Inspection to Mitigate COVID-19 Using K-Nearest Neighbour

In this paper, a model is recommended where the total number of people presenting the frame is detected using the YOLO object detection algorithm, and distance between each individual is measured Using K-Nearest Neighbour. If the distance between any two individual less than 6 feet or 2 meters then a red bounding box pops around them indicating that they are violating the rule of social distancing. This model is implemented on Raspberry Pi with a buzzer system for alert.

5. Social Distancing and Face Mask Monitoring System Using Deep Learning Based onCOVID-19 Directive Measures

The author proposed a system consisting of data processing, data augmentation, image classification using mobilenetv2 and object detection. The modules are developed using TensorFlow and open-cv python programming to detect faces with masks. If a person wears a mask they will be in a safe zone and the system shows a green box where if the person doesn't wear a mask, then it will be shown in a red box and with the message of alert as well. Social distancing detection will detect that two or more person in a single frame are walking with maintaining social distancing with at least 2 meters of range with each other using the Euclidean distance method, it will work in a Reliable manner with accurate results during this current situation which will easily help to track the person and collect fine if they violate any government directive guidelines so our system, will prevent the spread of the disease. Every Automation process reduces manual inspection to inspect the people which can be used in public places to control the spread of the virus and this prototype could be used in many places like park, hospital, airports, temples, railway station etc. to control this pandemic situation.

6. Application of Face Recognition in Tracing COVID-19 Fever Patients and close Contacts

The author developed a face recognition system to detect patients with fever symptoms and to trace close contacts. A real-time alert is sent to the account manager on a web or mobile app to enable further actions to quarantine the patients and close contacts. The RGB camera is used to detect a face and locate the forehead. The thermal image of the face is used to measure the temperature of the skin in the forehead. A black body is optional to improve the temperature measurement accuracy. After a patient is confirmed, his identification can be recognized using face recognition. By face recognition clustering, all face images of this person in the past given period of time (e.g., 14 days) can be retrieved. Furthermore, close contacts of this patient can also be retrieved from saved frame images or the camera ID and time stamp. The work [2] proposed a similar idea of using face recognition to trace fever patients and close contacts but did not give an algorithm on how to trace them. These retrieved results are displayed in an account console, and a notification is sent to the personnel (account manager) on duty in real time, and safety action can be taken to quarantine the persons, achieving the goals of stopping the virus spreading.

2.1 EXISTING SYSTEM

There are a number of applications that exist to alert people of containment zones. One such application is the Aarogya Setu app, which was developed by the Indian government. The app uses GPS to track the location of the user and sends alerts if the user enters a containment zone. Other apps that provide similar functionality include the Covid-19 India app and the Corona Kavach app.

2.2 REFERENCES

1.https://ieeexplore.ieee.org/document/9711880

2.https://ieeexplore.ieee.org/document/9432254

3.https://ieeexplore.ieee.org/document/9356316

4.https://ieeexplore.ieee.org/document/9388625

2.3 PROBLEM STATEMENT DEFINITION

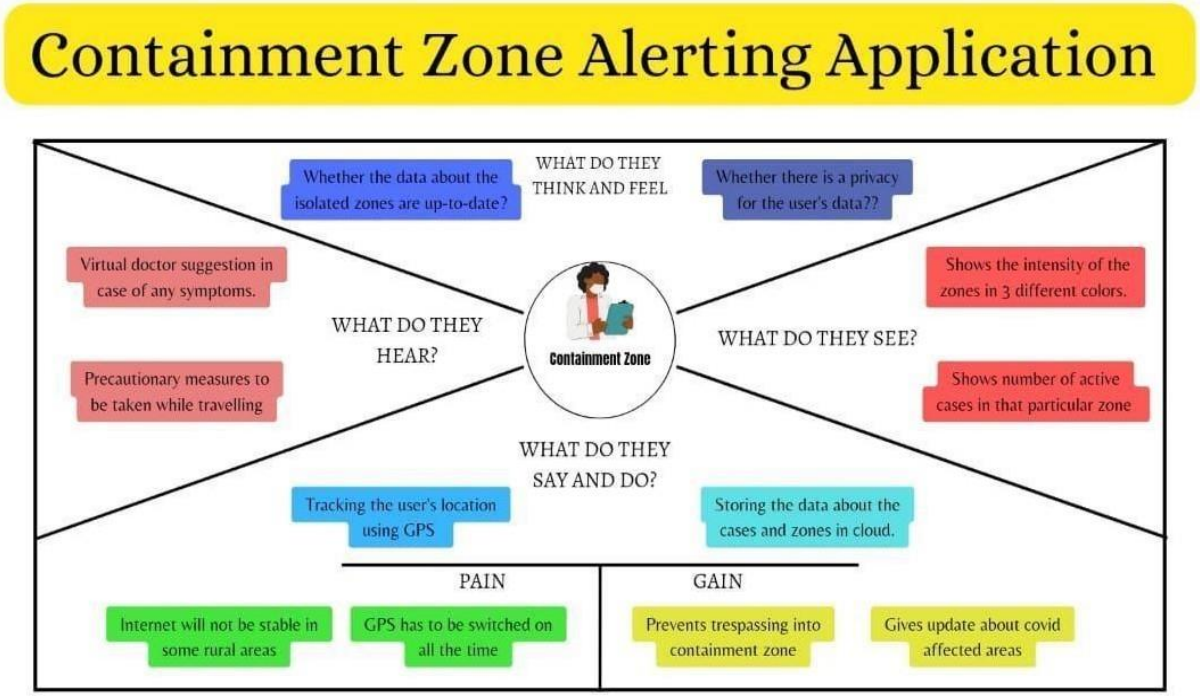
Problem Statement	I am	I'm trying to	But	Because	Which makes me feel
1	Traveler	Evade the containment zones	I accidently step into the containment zones	I don't have any medium to alert when I enterthe containment zones	I'm vulnerable
2	E-commerce delivery person	Not delivering E-commerce products inside containment zones	I don't know where are the containment zones	I don't have any resource to notify me when I enter into a containment zone	I feel fear when delivering because of the feeling of knowing about the containment zones

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

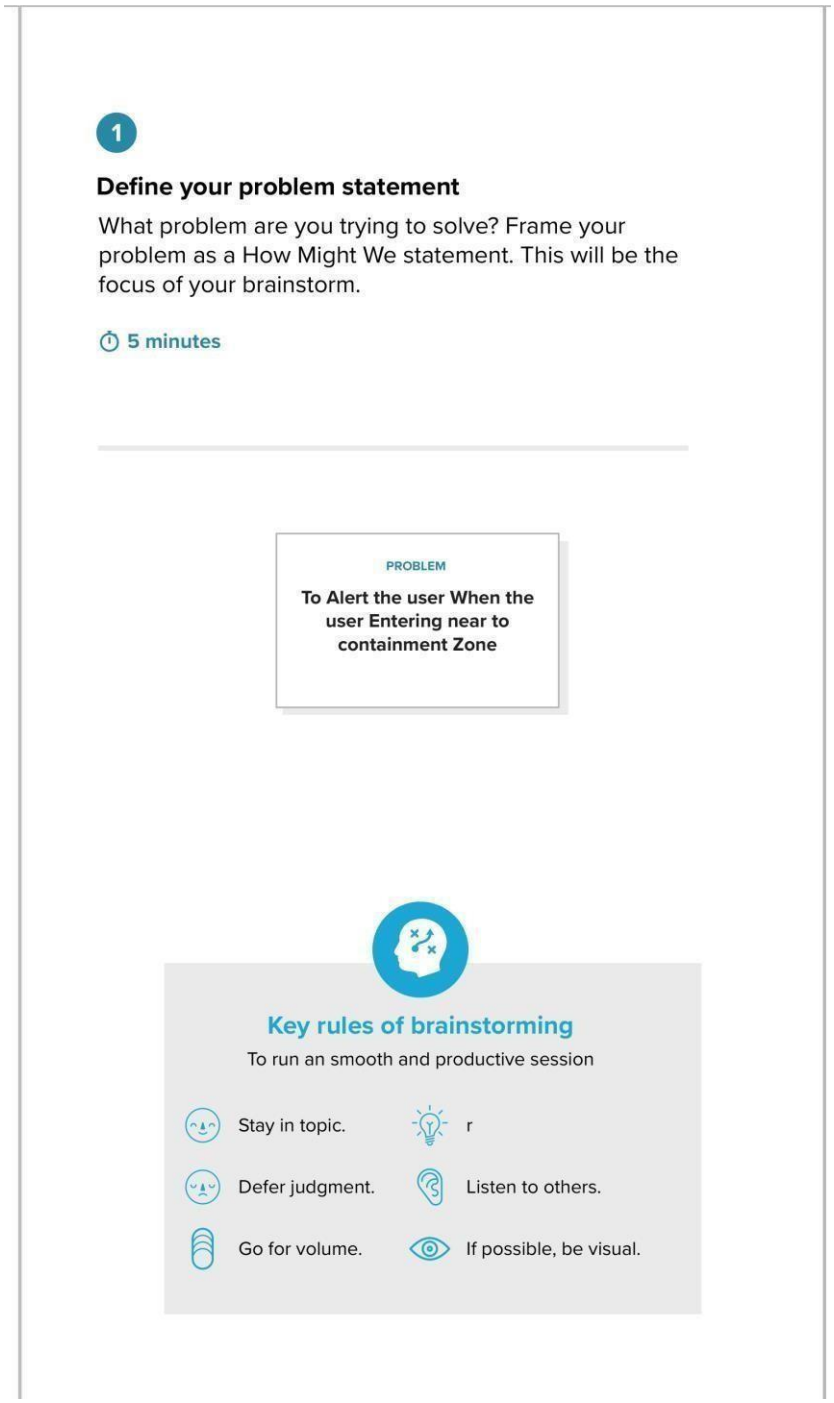
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



3.2 IDEATION & BRAINSTORMING

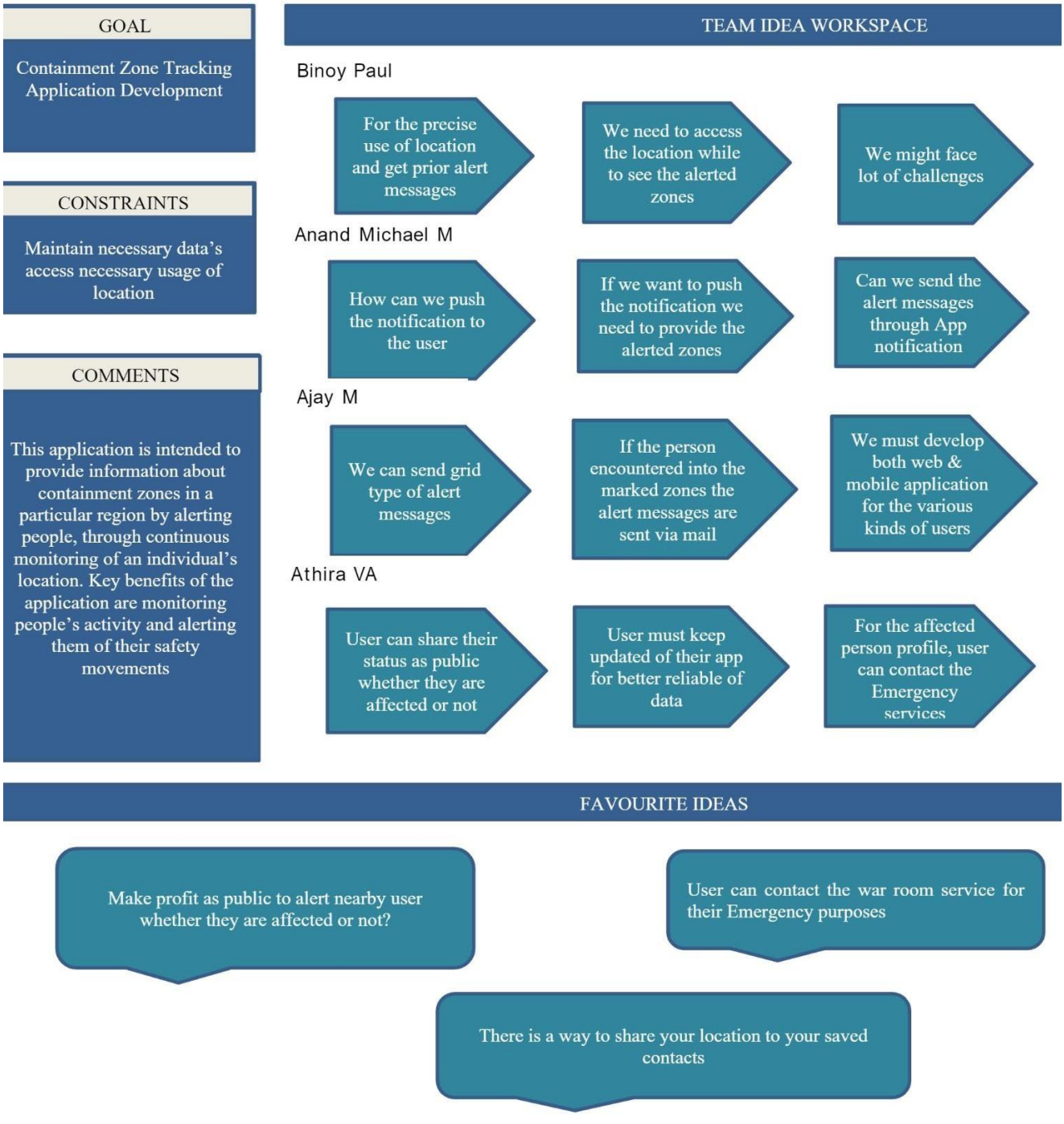
Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room. **STEP – 1: Team Gathering, Collaboration and Select the Problem Statement**



STEP-2 Brainstorm, Idea Listing and Grouping

Brainstorming, Idea Listing and Grouping

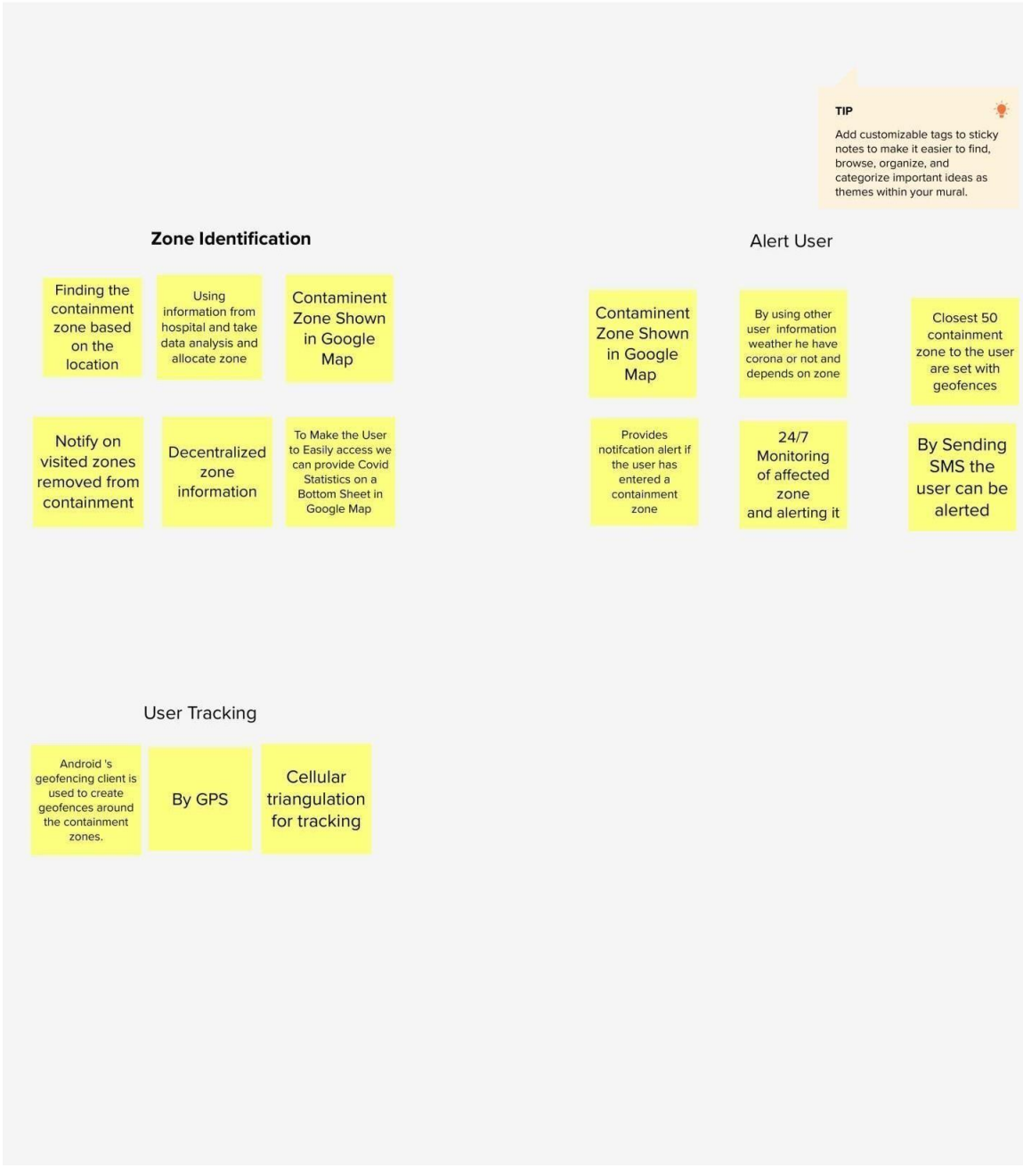


3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



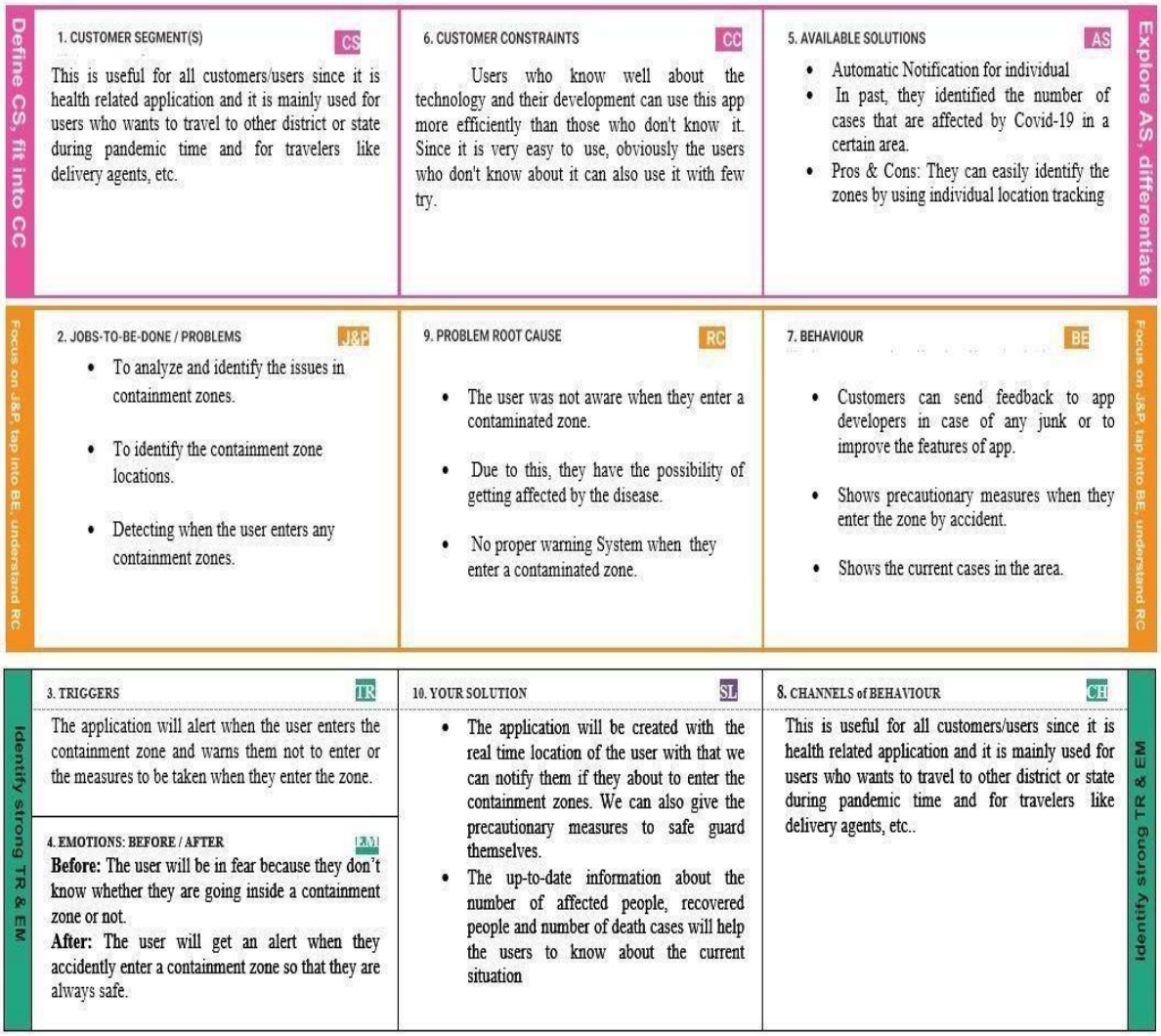
STEP-3 Idea Prioritization

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description

1.	Problem Statement (Problem to be solved)	Development of an android application for viewing the covid containment zones and also the alerting the users not to enter the affected area using cloud and geofencing by sending notification.
2.	Idea / Solution description	To create an easy-to-use android application to alert the user when they enter a Containment Zone. To provide accurate results and alerting at the exact time when they enter the zone. This is done with the help of integration of Google Maps.
3.	Novelty / Uniqueness	• Development of an android application is necessary which can inform people of the Covid-19 containment zones and prevent trespassing into these zones. • Android Application updates the locations of the areas in a Google map which are identified to be the containment zones. • The application also notifies the users if they have entered a containment zone and upload the details of individual in online database.
4.	Social Impact / Customer Satisfaction	The application saves people's life from restricting them entering the Containment zone which saves them from catching the disease. Also shows precautionary measures when they entered the zones.
5.	Business Model (Revenue Model)	Can tie up with people with normal and premium charges. • The data that is derived can be used in Government sectors. • Can tie up the Government and get profit through that.
6.	Scalability of the Solution	The application will be useful for all people from saving their life's from catching the disease by alert

3.4 PROBLEM SOLUTION FIT



CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	It can be registered by valid Email id or Phone number.
FR-2	User Confirmation	Verification code can received by registering email id or phone number.

FR-3	Alert message via Notification	By user access of location while entered in the alerted area the notification are send by GPS tracking system and push the grids through mail id.
FR-4	Show Infected Zones	Marked by Geofencing.
FR-5	Track alternate routes	By Google map API or Google dependencies.

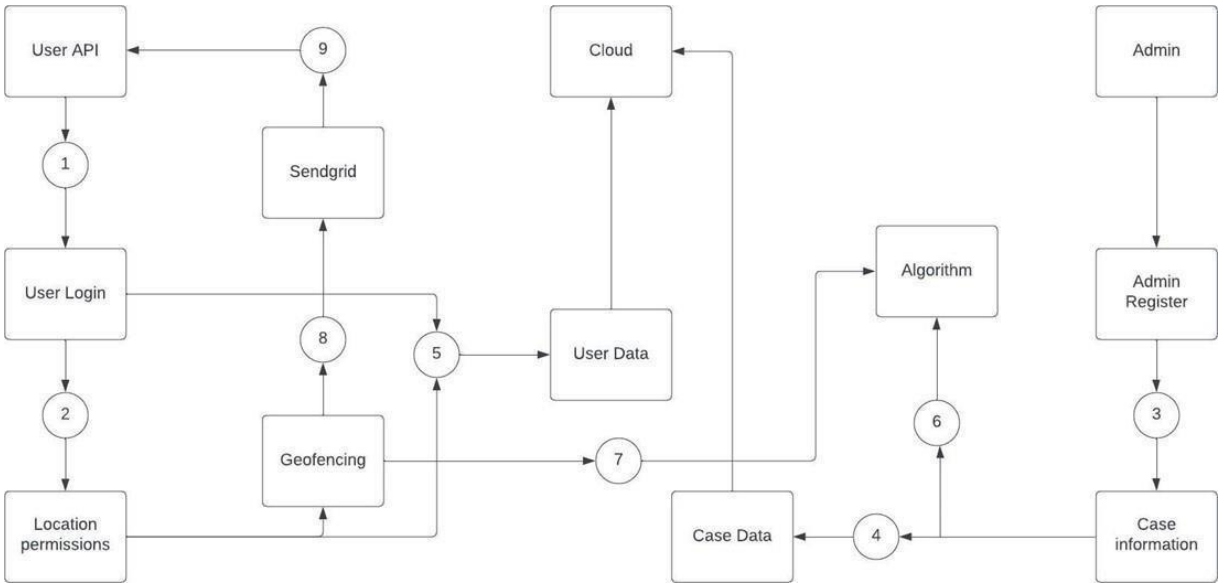
4.2 NON-FUNCTIONAL REQUIREMENT

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	GUI is easier to interact with.
NFR-2	Security	The data collected from the user will be stored securely.
NFR-3	Reliability	The user can trust the results and navigate safely.
NFR-4	Performance	Accurate results can be achieved due to realtime location 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.

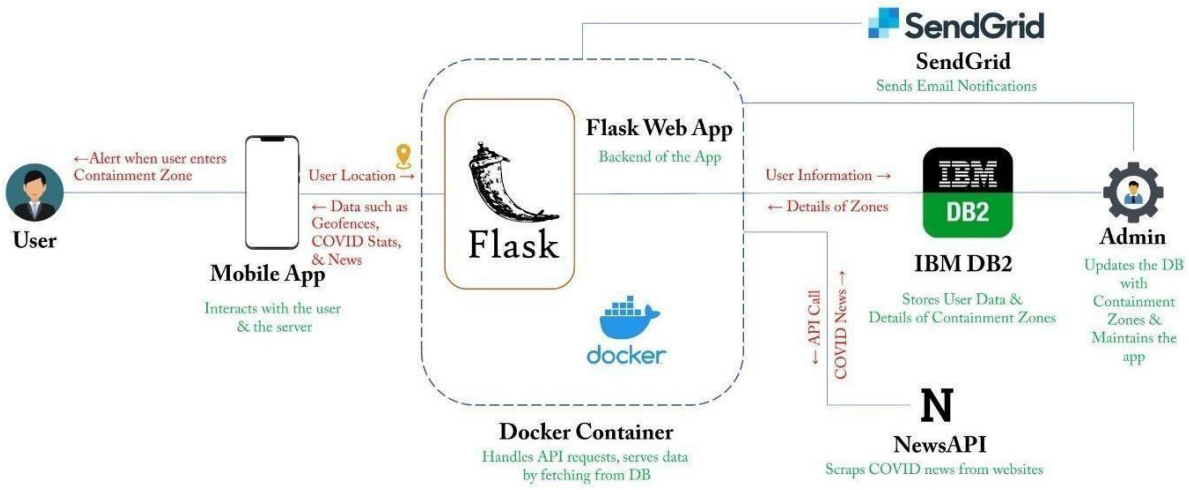
CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

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 will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-4
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Google login	Medium	Sprint-1
		USN-5	As a user, I can register for the application through Twitter	I can register & access the dashboard with Twitter login	Low	Sprint-4
	Login	USN-6	As a user, I can log into the application by entering my email & password	I can access it whenever I want.	High	Sprint-1
	Dashboard	USN-7	As a user, I need to give permission to access my contacts, location and storage	I get access to their services.	High	Sprint-2
		USN-8	As a user, I get access to the dashboard which shows a map with marked zones.	I can see the zone information on the dashboard.	High	Sprint-2
Administrator	Services	USN-9	As an admin, I need to provide valid information	I can get the pandemic	High	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
			about the pandemic out there.	updates out there.		
		USN-10	As an admin, I need to provide medical advice through a chat bot.	I get medical recommendation through a chatbot.	Medium	Sprint-3
		USN-11	As an admin, I need to provide medical recommendations by collaborating with top hospitals.	I get medical instruction through chief doctors.	Low	Sprint-3
		USN-12	As an admin, I need to alert the user when they enter pandemic zones.	I got a notification when I was in the pandemic area.	Medium	Sprint-4

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the technical papers, research publications etc..	16 October 2022
Prepare Empathy Map	Prepare Empathy Map canvas to capture the user pains & gains , prepare list of problem statements.	17 October 2022
Ideation	List the by organizing the Brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	19 October 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty , feasibility of idea, business model, social impact , scalability of solution etc..	18 October 2022
Problem Solution Fit	Prepare problem – solution fit document.	16 October 2022
Solution Architecture	Prepare solution architecture document.	17 October 2022

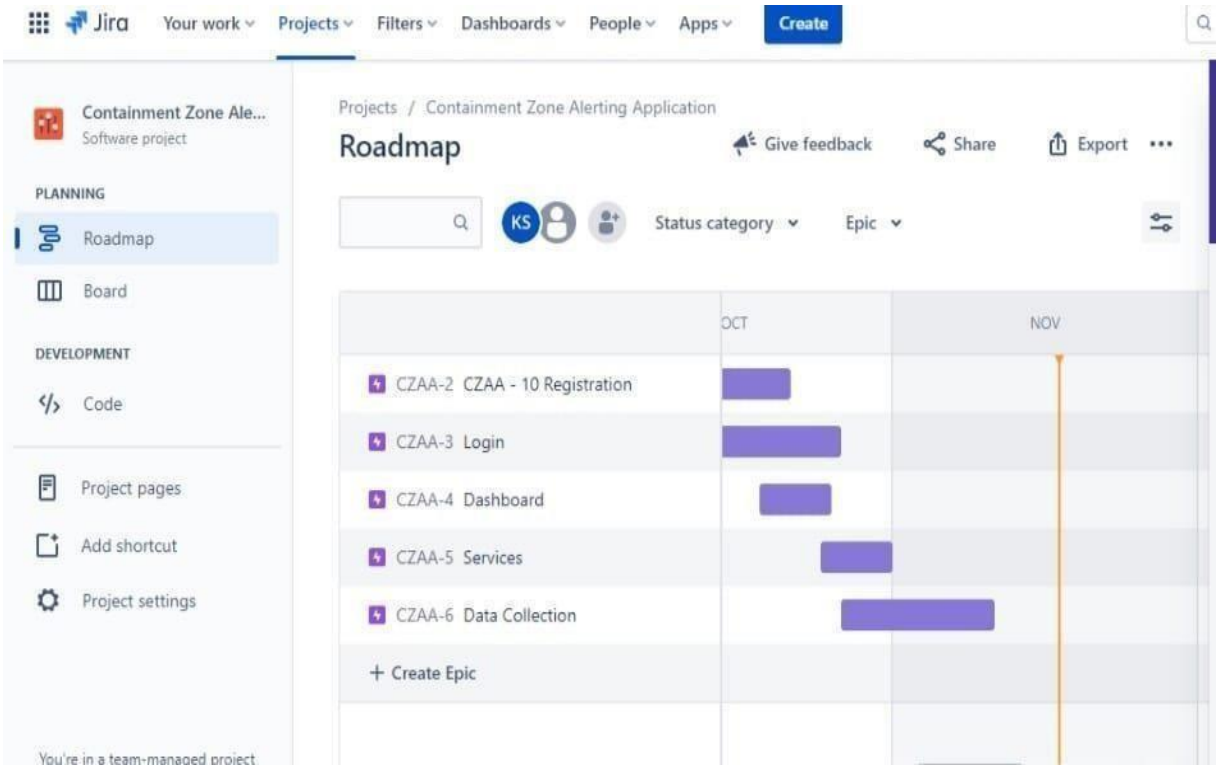
Customer Journey	Prepare the customer Journey maps to understand the user interactions & experiences with the application.	17 October 2022
Functional Requirements	Prepare the functional requirement document.	19 October 2022
Data Flow Diagrams	Draw the data flow diagram and to submit for review.	19 October 2022
Technology Architecture	Prepare the technology architecture diagram.	19 October 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	29 October 2022
Project Development – Delivery Sprint – 1,2,3,4.	Develop & submit the developed code by testing it.	1 November 2022

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, and password and confirming my password.	3	High
Sprint-1		USN-2	As a user, I will receive a confirmation email once I have registered for the application.	2	High
Sprint-4		USN-3	As a user, I can register for the application through Facebook.	2	Low
Sprint-1		USN-4	As a user, I can register for the application through Gmail.	5	Medium
Sprint 4		USN-5	As a user, I can register for the application through Twitter.	2	Low
Sprint-1	Login	USN-6	As a user, I can log into the application by entering my email & password	3	High
Sprint-2	Dashboard	USN-7	As a user, I need to give permission to access My Contacts, Location, and Storage.	5	High

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-2		USN-8	As a user, I get access to the dashboard which shows a map with marked zones	5	High
Sprint-1	Registration	USN-9	As a management, I need to register my hospitals on the site.	2	high
Sprint-1	Login	USN-10	As a management, I need to login into my dashboard with my given hospital id and password.	5	medium
Sprint-2	Dashboard	USN-11	As a management, I need to enter the case information of the patient that visits our hospital.	5	high
Sprint-3		USN-12	As a management, I need to store all the patient information on the cloud	5	high
Sprint-2	Services	USN-13	As an admin, I need to provide valid information about the pandemic out there.	5	high
Sprint-3		USN-14	As an admin, I need to provide medical advice through a chatbot.	5	medium
Sprint-3		USN-15	As an admin, I need to provide medical recommendations by collaborating with top hospitals.	5	low
Sprint-4		USN-16	As an admin, I need to alert the user when they enter pandemic zones.	3	Medium
Sprint-3		USN-17	As an admin, I need to provide preventive measures when they travel through it.	5	high
Sprint 4		USN-18	As an admin, I need to provide special services for premium users by giving services like monitoring health by their smart bands.	3	low
Sprint-4	Data collections	USN-19	As an admin, I need to store all the user information on the cloud	5	Medium

6.3 REPORTS FROM JIRA



CHAPTER 7

CODING & SOLUTIONING

7.1 FEATURE-1

In this page, the admin can register,login,remove zone and add zone.

CODING:

Login :

```
<html lang="en" xmlns="http://www.w3.org/1999/html">
<head>
   <!DOCTYPE html>
<meta charset="UTF-8">
   <title>Login</title>
   <link rel="stylesheet" href="../static/adminlogin.css">
</head>
<body>
<video src="../static/video/background.mp4" autoplay loop playsinline muted></video> <div
class="box">
   <form action="/login" method="POST" autocomplete="off">
      <h2>LOGIN</h2>
      <div class="inputBox">
         <input type="text" name="email" id="email" required="required">
         <span>Email</span>
         <i></i>
      </div>
      <small></small>
      <div class="inputBox">
         <input type="password" required="required" name="password" id="password">
<span>Password</span>
         <i></i>
      </div>
      <small></small>
      <div class="links">
         <a href="#">Forgot Password ?</a>
         <a href="{{url_for('adminRegistration')}}"><b>REGISTER</b></a>
      </div>
      <input type="submit" value="Login">
         <h4 class="text-danger" id="pass-warning" style="color: white"></h4>
</form>
</div>
<script>document.getElementById("pass-warning").innerHTML = '{{data}}';</script> <script
src="../static/js/adminlogin.js">
</script>
</body>
</html>
```

Register :

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/html">
<head>
   <meta charset="UTF-8">
   <title>Registration</title>
   <link rel="stylesheet" href="../static/adminreg.css">
   <link rel="stylesheet" href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css"
integrity="sha384-
AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p"
crossorigin="anonymous"/>
</head>
<body>
<video src="../static/video/background.mp4" autoplay loop playsinline muted></video>
<form action="/" method="POST">

   <div class="user">
      <i class="fas fa-user"></i>
      <input type="text" placeholder="Enter your username" name="username" id="username">
<small></small>
   </div>
   <div class="user">
      <i class="fas fa-user"></i>
      <input type="email" placeholder="Enter your mailid" name="email" id="email">
<small></small>
   </div>
   <div class="password">
      <i class="fas fa-lock"></i>
      <input type="password" placeholder="Password" name="password" id="password">
      <br>
      <small></small>
   </div>
   <div class="password">
      <i class="fas fa-lock"></i>
      <input type="text" placeholder="Confirm password" name="confirm_password"
id="confirm_password">
      <br>
      <small></small>
   </div>
   <div class="button">
      <button class="button" type="submit"><b>REGISTER</b></button>
      <button><a href="/login">LOGIN</a></button>
</div>
```

```
   <h4 class="text-danger" id="pass-warning" style="color: white"></h4>
</form>
<script>document.getElementById("pass-warning").innerHTML = '{{data}}';</script>
<script src="../static/js/adminreg.js"></script>
</body>
</html>
```

Containment Zone List :

```
<!DOCTYPE html>
<html lang="en">

<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Zones</title>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous" />
   <style>        body {
padding-top: 30px;            padding-
bottom: 30px;            background-
color: #0C1017;
          color : #F5F4F4;

      }
      a ,td,th {
         color: #F5F4F4;
      }
   </style>
</head>

<body>
   <div class="m-4 container">
      <h1 style="font-family : monospace">Location data and Visited People</h1>
</div>
   <div class="m-4 container">
      <table class="table">
         <thead>
            <tr>
               <th scope="col">S.No</th>
               <th scope="col">Latitude</th>
               <th scope="col">Longitude</th>
               <th scope="col">No_Visited</th>
            </tr>
```

```
        </thead>
        <tbody>

            {%- for row in data %}
            <tr>
                <th scope="row">{{loop.index}}</th>
                <td>{{row['LOCATE_LAT']}}</td>
<td>{{row['LOCATE_LANG']}}</td>
                <td>{{row['VISITED']}}</td>
            {%- endfor %}
        </tbody>
      </table>
   </div>
   <div class="m-3 float-right">
      <button type="button" class="btn btn-danger"><a href={{url_for("home")}}>Go to location
update Page</a></button>
   </div>

</body>

</html>
```

Home/Remove/Add Zone :

```
<!Doctype html>
<html>
   <head>
      <title>Dashboard</title>
      <link rel="stylesheet" href="static/style.css">
      <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap" rel="stylesheet">
<meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous" />
   <style>
```

```
        body {
            padding-top: 30px;            padding-
bottom: 30px;            background-color:
#0C1017;
            color : #F0F6FC;
            font-family: 'Open Sans', sans-serif;
padding: 30px;
        }

        .m-3 float-right {
            background-color : #0C1017;

        }
a {
            color: #F0F6FC;

        }
<!--        design the whole html page to beautify-->


    </style>
    </head>
    <body>
     {% if success %}
    <script>
        alert("Location Uploaded Successfully");
    </script>
     {% elif not success %}
    <script>
        alert("Enter Proper Location data");
    </script>
    {% endif %}
    <div class="logout">
        <button type="button" class="btn btn-primary"><a href={{url_for("logout")}}>Log
Out</a></button>
    </div>
    <div class="logout">
        <h1>Declare Containment Zone</h1>
    </div>

        <h2>Welcome: {{name}}</h2>

    <form method="POST" action="/home">
        <div class="container">
            <div class="form-group row">
                <div class="col-sm-6">
```

```
                <label class="control-label">Lat.:</label>
                <input type="text" class="form-control" id="lat" name="lat" />
</div>
            <div class="col-sm-6">
                <label>Long.:</label>
                <input type="text" class="form-control" id="lon" name="lon" />
            </div>
            <div class="col-sm-6">
                <label>Get current Location:</label>
                <button type="button" class="btn btn-warning" onclick="getLocation()">Current
Location</button>
                <label>(Click this first)</label>
            </div>
        </div>

        <!-- map -->
        <div id="map_disp" style="height: 400px;width: 500px;"></div>
        <div class="m-3 float-right">
            <button type="submit" class="btn btn-danger">Declare Containment Zone</button>
        </div>
        <div class="m-3">
            <button onclick="toggleTips()" type="button" class="btn btnsecondary">Tutorial</button>
            <div id="tips" class="m-3">
                <ol>
                    <li>Select The Location By Clicking the Current Location Button</li>
                    <li>Drag the Pin to change the location</li>
                    <li>Click on Declare Containment Zone to save the location to the database </li>
                </ol>
            </div>
        </div>
        <div class="m-3 float-right">
            <button type="button" class="btn btn-warning"><a href="{{url_for('data')}}">Click Here
To View The
                    Containment Zones and Number of
people visited</a></button>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.min.js"
integrity="sha384-
+YQ4JLhjyBLPDQt//I+STsc9iw4uQqACwlvpslubQzn4u2UU2UFM80nGisd026JF"
crossorigin="anonymous"></script>
    <script src="https://code.jquery.com/jquery-2.2.4.min.js"></script>
    <script
src="https://maps.google.com/maps/api/js?sensor=false&amp;libraries=places"></script>
    <script
```

```
        src="https://rawgit.com/Logicify/jquery-
locationpickerplugin/master/dist/locationpicker.jquery.js"></script>

      <script>
         function getLocation() {
if (navigator.geolocation) {
               navigator.geolocation.getCurrentPosition(showPosition);
            } else {
               alert("No location");
            }
         }
         function showPosition(position) {
$('#map_disp').locationpicker({
               location: {                    latitude:
position.coords.latitude,
                  longitude: position.coords.longitude
               },
               radius: 0,
inputBinding: {
latitudeInput: $('#lat'),
                  longitudeInput: $('#lon'),
               },
               enableAutocomplete: true,
               onchanged: function (currentLocation, radius, isMarkerDropped) {
// Uncomment line below to show alert on each Location Changed event                    //
alert("Location changed. New location (" + currentLocation.latitude + ", " +
currentLocation.longitude + ")");
               }
            });
         }
         function toggleTips() {
            var x = document.getElementById("tips");
if (x.style.display === "none") {
x.style.display = "block";
            } else {
               x.style.display = "none";
            }
         }
      </script>

   </form>
   </body>
</html>
```

7.2 FEATURE-2

The users get alerted from entering the contaminated zone by geofencing the location and sending it as notification.

CODING:

```
# create a flask app
import re import
ibm_db
from flask import Flask, flash, render_template, request, redirect, url_for, session from
sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

app = Flask(_name_, static_url_path='/static', static_folder='static', template_folder='templates')
app.secret_key = 'sus'
conn = ibm_db.pconnect('DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL;S
SLServerCertificate=DigiCertGlobalRootCA.crt;UID=tyb34892;PWD=Qq5GdhZKREQl1Vrc', '', '')

# print("Connected to database", conn)

session = {}


# route for sending email
@app.route('/sendemail') def
sendemail():
id=session["id"]
   sql="select email from users1 WHERE id=?;"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt, 1, id)
ibm_db.execute(stmt)     data =
ibm_db.fetch_assoc(stmt)
   print(data)

   message =
Mail(from_email='anandmic06@gmail.com',to_emails=data['EMAIL'],subject='CONTAINMENT
ZONE ALERT!!!',html_content='<strong>OPPS!<br> YOU HAVE ENTERED A
CONTAMINATED ZONE! <br> PLEASE STAY SAFE.</strong>')     try:
      sg =
SendGridAPIClient('SG.9nbY1OfbSTeYnWdfSEBzXw.e0rYMvlTWmlfcLdwnFVP7hOH7HwSsV
Oxs-GSc_AXMtU')
      response = sg.send(message)
print(response.status_code)
print(response.body)
```

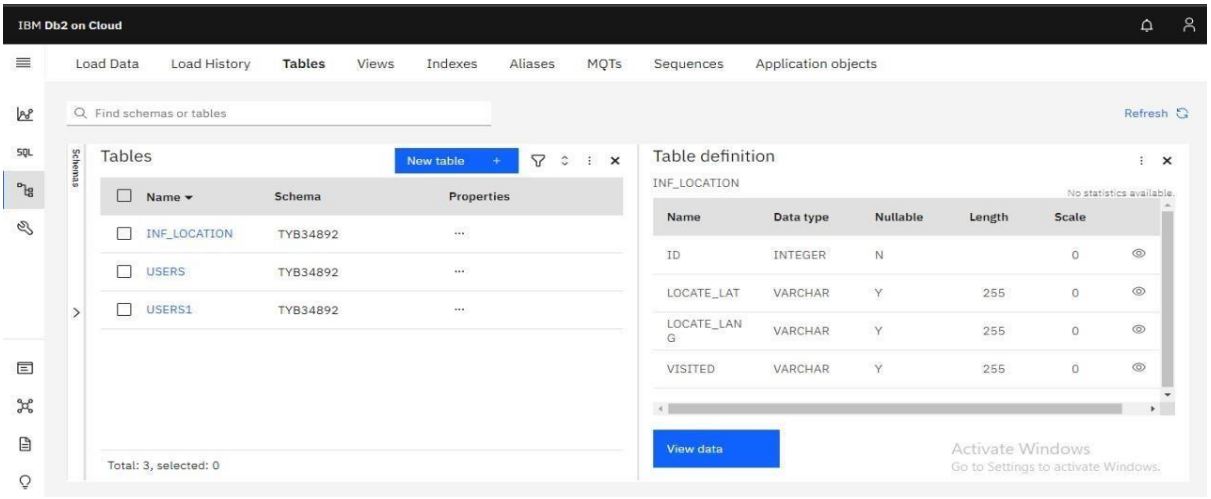
```
print(response.headers)     except
Exception as e:         print(e)

    return render_template('home.html')



# create a route for the home page
@app.route('/', methods=['GET', 'POST'])
def register():     message = ''     if
request.method == 'POST':         # get the
data from the form         name =
request.form['username']         email =
request.form['email']         password =
request.form['password']
        confirm_password = request.form['confirm_password']
        # if nothing is entered in the form         if not name or not email or
not password or not confirm_password:
            message = 'Please fill all the fields!'             return
render_template('register.html', message=message)         # if the
password and confirm password do not match         elif password
!= confirm_password:             message = 'Passwords do not
match!'
            return render_template('register.html', message=message)

        #  password length must be 8 or above         if
len(password) < 8:             message = 'Password must be 8 or
more characters'             return render_template('register.html',
message=message)
        # check if the email is valid         if
re.match(r"[^@]+@[^@]+\.[^@]+", email):
            # insert the data into the database
            # check if email already exists in the database
            sql = "SELECT * FROM users1 WHERE email = '" + email + "'"
            stmt = ibm_db.exec_immediate(conn, sql)
            # print("stmt", stmt)
            result = ibm_db.fetch_assoc(stmt)
            # print("result", result)
if result:
                message = 'The username or email already exists!'
else:
                sql = "INSERT INTO users1 ( username, email, password,type) VALUES (?, ?, ?, ?)"
stmt = ibm_db.prepare(conn, sql)

                ibm_db.bind_param(stmt, 1, name)
ibm_db.bind_param(stmt, 2, email)
```

```
ibm_db.bind_param(stmt, 3, password)
ibm_db.bind_param(stmt, 4, "ssss")                ibm_db.execute(stmt)
              message = 'You have successfully registered!'
return redirect(url_for('login'))         else:
           message = 'The email is invalid!'
    return render_template('register.html', message=message)


@app.route('/login', methods=['GET', 'POST'])
def login():     message = "if
request.method == 'POST':         # get the data
from the form         email =
request.form['email']         password =
request.form['password']        # if nothing is
entered in the form         if not email or not
password:
           message = 'Please fill all the fields!'
           return render_template('login.html', message=message)
       # check if the username and password are valid
       sql = "SELECT * FROM users1 WHERE email = '" + email + "' AND password = '" + password
+ """
       stmt = ibm_db.exec_immediate(conn, sql)
result = ibm_db.fetch_assoc(stmt)
       # print("result", result)
       if result:
           # message = 'You have successfully logged in!'
session['id'] = result['ID']
           session['username'] = result['USERNAME']
           session['email'] = result['EMAIL']             #
print("id ==", session['id'])             return
render_template('home.html', mail=email)         else:
           message = 'The email or password is incorrect!'
return render_template('login.html', message=message)


@app.route('/logout') def
logout():
    session.clear()
    return redirect(url_for('login'))


# create a route for the home page and open only if the user is logged in
@app.route('/home', methods=['GET', 'POST'])
def home():     # print(name)

    if 'id' in session:         if
request.method == 'GET':
```

```
        return render_template('home.html', name=session['username'])
if request.method == "POST":
        # get data            lat =
request.form["lat"]            lon =
request.form["lon"]
        sql = "SELECT * FROM inf_location WHERE locate_lat = ? AND locate_lang = ?;"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt, 1, lat)
        ibm_db.bind_param(stmt, 2, lon)

        ibm_db.execute(stmt)
data = ibm_db.fetch_assoc(stmt)
if (data)!=0:

            flash("OOH!! You Have Entered A Contaminated Zone")
return redirect(url_for('sendemail'))

else:
            flash("You Are in Safe Zone")
return redirect(url_for('home'))

    return render_template('home.html')
else:
    return redirect(url_for('login'))




# create a route for the data page and open only if the user is logged in
@app.route('/data') def data():     if
'id' not in session:        return
redirect(url_for('login'))     else:
    # create a query to fetch the data from the database
sql = "SELECT * FROM inf_location"        stmt =
ibm_db.exec_immediate(conn, sql)
    # print("stmt", stmt)
    # fetch all the data from the database and store it in the result dictionary
result = ibm_db.fetch_assoc(stmt)

    # create a list to store the data
    data = []
    # loop through the result dictionary and append the data to the list
while result:            data.append(result)
        result = ibm_db.fetch_assoc(stmt)
```

```
    # print(data)
    return render_template('data.html', data=data)


if__name__== '_main_':
app.run(host='0.0.0.0', port=5000, debug=True)
```

7.3 DATABASE SCHEMA



CHAPTER 8

TESTING

8.1 TEST CASES

1. Login button click with wrong credentials entered.
2. Signup with already registered mail ID.
3. Signup with wrong form data entered.
4. Entering home page with logged out session.
5. Clicking home page buttons with logged out session.
6. Invalid data entered in change password page and requested for change in password.

8.2 USER ACCEPTANCE TESTING

S.NO	TEST CASE	REQUIRED OUTPUT	RESULT OUTPUT	STATUS
1	Login button click with wrong credentials	Wrong credentials entered notification	Wrong credentials entered notification	ACCEPTED
2	Signup with already registered mail ID.	Email already registered notification	Email already registered notification	ACCEPTED
3	Signup with wrong form data entered.	Wrong credentials entered notification	Wrong credentials entered notification	ACCEPTED
4	Entering home page with logged out session.	Take user to login page	Take user to login page	ACCEPTED
5	Clicking home page buttons with logged out session.	Take user to login page	Take user to login page	ACCEPTED
6	Invalid data entered in change password page and requested for change in password.	Wrong form data entered notification	Wrong form data entered notification	ACCEPTED

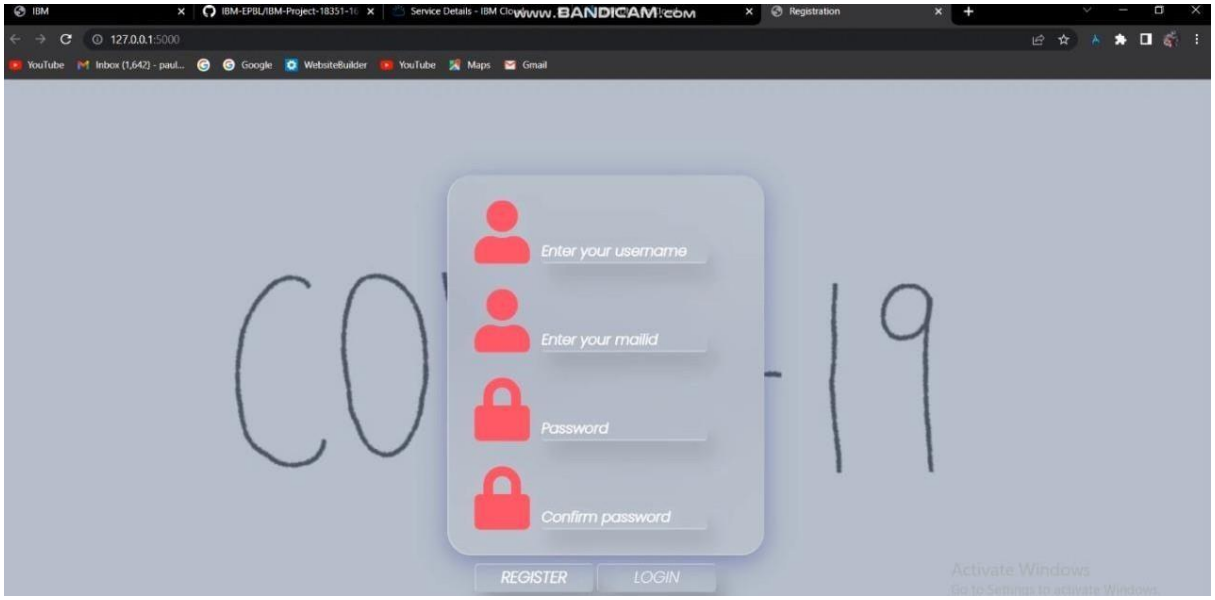
CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

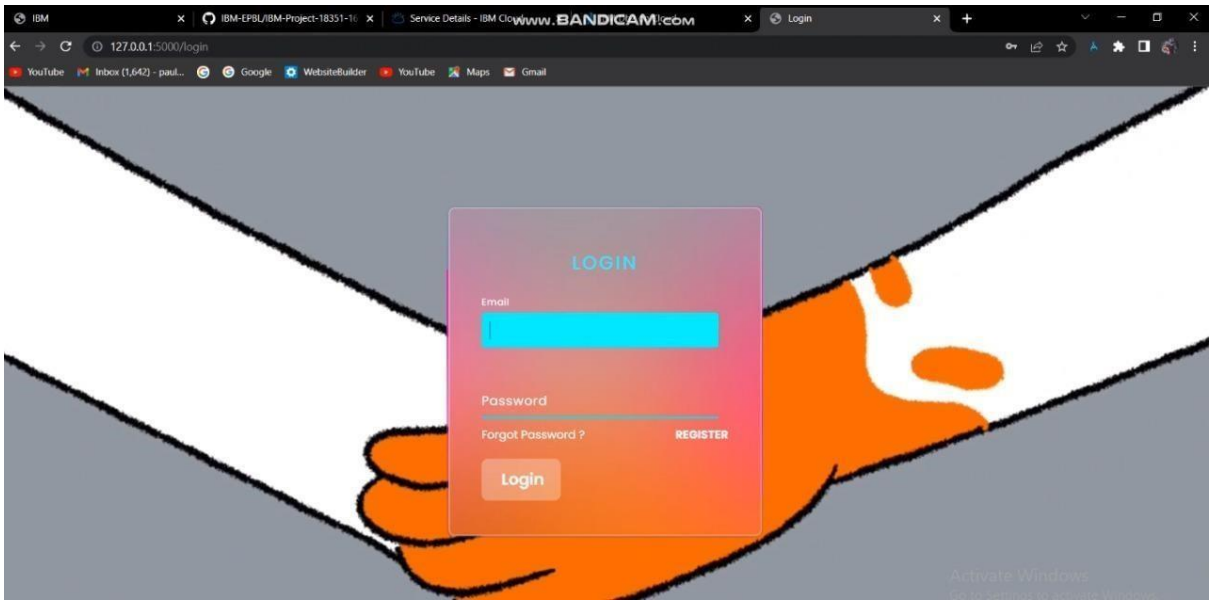
This app service monitors the location and provide information about the contaminated zones near a particular user and send notification to the user. It displays the contaminated zone area by geofencing the particular location.

Registration Form

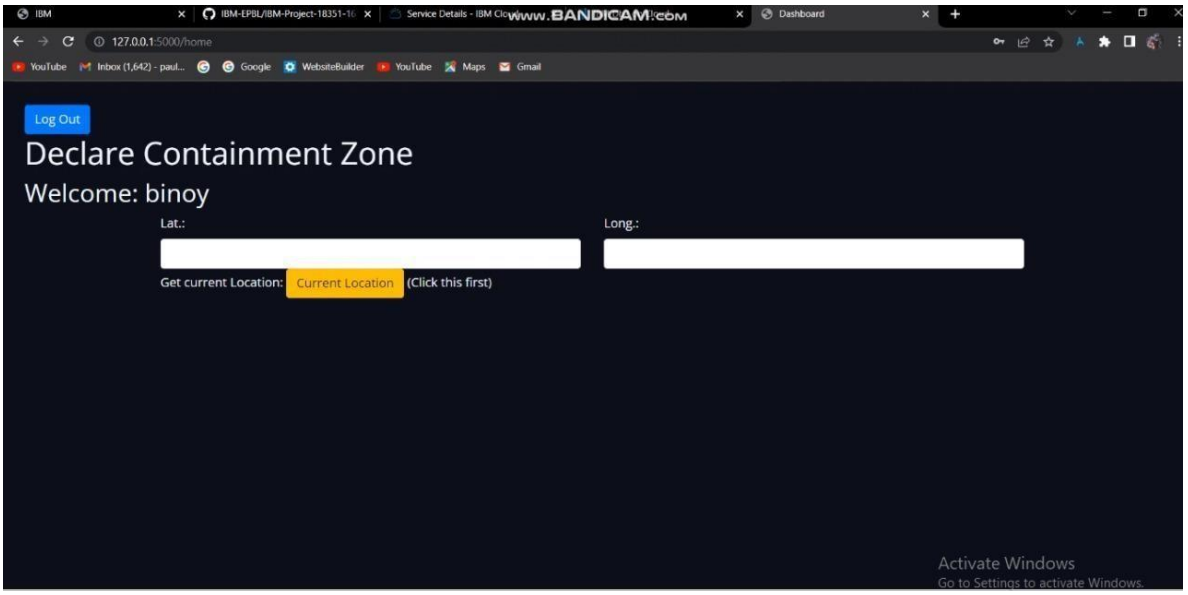


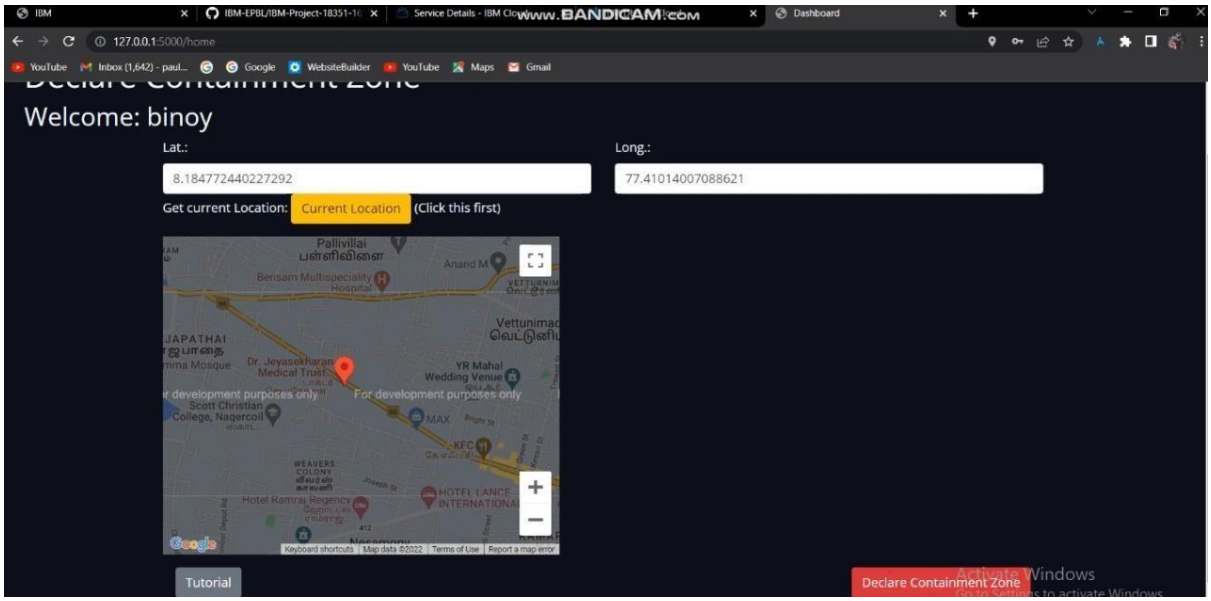
SS

Login Form

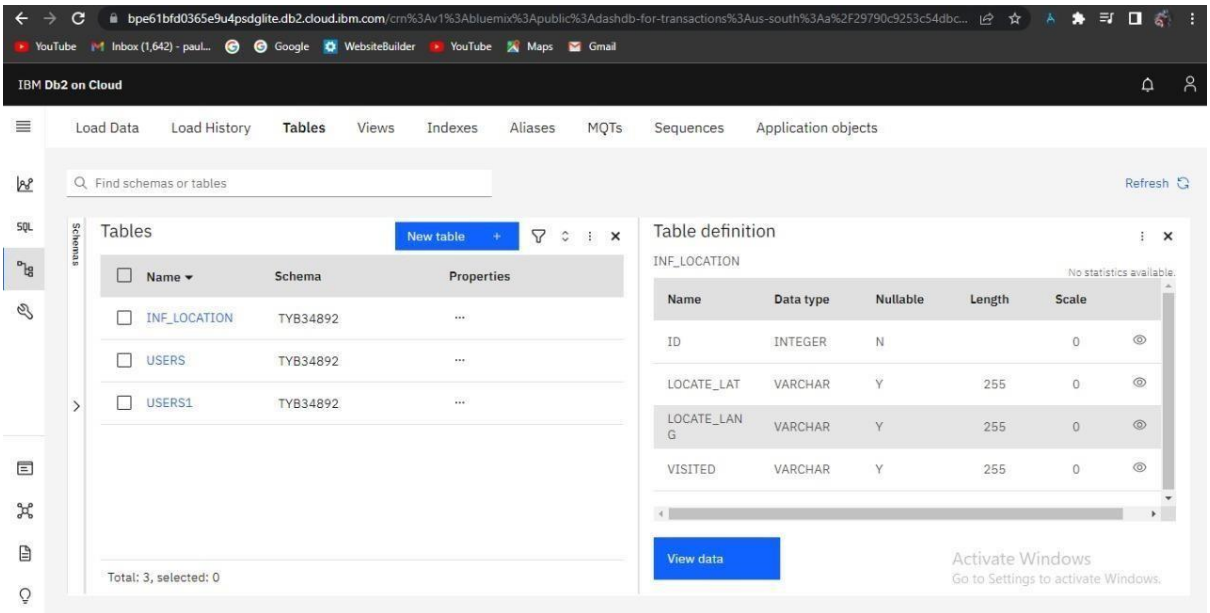

LOGIN
Email
Password
Forgot Password ?
REGISTER
Login

Home Page


Log Out
Declare Containment Zone
Welcome: binoy
Lat.:
Long.:
Get current Location:
Current Location
(Click this first)



IBM Database



CHAPTER 10

ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES & DISADVANTAGES

The main advantage of containment zone alerting applications is that they can help to prevent the spread of diseases by alerting people to areas where there is a risk of infection. However, there are also some disadvantages to these, including the potential for false alarms and the possibility that people may ignore the warnings. 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 are monitoring people's activity and alerting them of their safety movements.

CHAPTER 11

CONCLUSION

We proposed a framework for identifying the contaminated zone areas and store it in database for future use. Then using the database, information is provided to the user about contaminated zone areas and alerting them by sending notification and geofencing the location. From the above information, it can be concluded that the Containment zone Alerting Application, in which we have successfully developed is a web application that sends alerts to users when they enter or exit a containment zone. It uses GPS to track the user's location and sends an alert if the user enters or leaves a containment zone. It also allows users to set up alerts for specific containment zones. It has successfully demonstrated. In this project, we alert users about the containment zone area by that they are aware and realize of high containment zone area.

CHAPTER 12

FUTURE SCOPE

The application provides an efficient way of showing the identified COVID-19 containment zones to the users in a Google map. With the alarming increase of COVID-19 affected cases throughout the world, it can be employed as a tool for creating further social awareness among the people. This application further tracks the user's location and checks whether it is present in the list of identified containment zones. It sends separate notification alerts to the user on entering and exiting the containment areas. The developed android application further extracts the IMEI Number of the trespasser in the containment zones which can be useful to the local police to track and identify people who are frequently trespassing the containment zones. Thereby this application identifies the containment zones and highlights the need for taking further precautionary measures for combating COVID- 19. The application has been tested in various locations and has been found to yield accurate results. The application can be further used for many purposes like maritime and forest safety to prevent users from entering restricted areas.

CHAPTER 13

APPENDIX

The Containment zone alerting application is a web application that sends alerts to users when they are in close proximity to a containment zone. The app uses the user's location to determine if they are in close proximity to a containment zone, and if so, sends an alert to the user. The app also allows users to view a map of containment zones in their area, and provides information on how to avoid contracting the virus.

HOME.HTML

```
<!Doctype html>
<html>
   <head>
      <title>Dashboard</title>
      <link rel="stylesheet" href="static/style.css">
      <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap" rel="stylesheet">
<meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous" />
   <style>

      body {
         padding-top: 30px;          padding-
bottom: 30px;          background-color:
#0C1017;
         color : #F0F6FC;
         font-family: 'Open Sans', sans-serif;
padding: 30px;
      }

      .m-3 float-right {
         background-color : #0C1017;

      }
       a
{
```

```
        color: #F0F6FC;

    }
<!--    design the whole html page to beautify-->


  </style>
  </head>
  <body>
   {% if success %}
  <script>
    alert("Location Uploaded Successfully");
  </script>
   {% elif not success %}
  <script>
    alert("Enter Proper Location data");
  </script>
  {% endif %}
  <div class="logout">
    <button type="button" class="btn btn-primary"><a href={{url_for("logout")}}>Log
Out</a></button>
  </div>
  <div class="logout">
    <h1>Declare Containment Zone</h1>
  </div>

    <h2>Welcome: {{name}}</h2>

  <form method="POST" action="/home">
    <div class="container">
      <div class="form-group row">
        <div class="col-sm-6">
          <label class="control-label">Lat.:</label>
          <input type="text" class="form-control" id="lat" name="lat" />
</div>
        <div class="col-sm-6">

          <label>Long.:</label>
          <input type="text" class="form-control" id="lon" name="lon" />
        </div>
        <div class="col-sm-6">
          <label>Get current Location:</label>
          <button type="button" class="btn btn-warning" onclick="getLocation()">Current
Location</button>
          <label>(Click this first)</label>
        </div>
```

```
        </div>

        <!-- map -->
        <div id="map_disp" style="height: 400px;width: 500px;"></div>
        <div class="m-3 float-right">
          <button type="submit" class="btn btn-danger">Check Containment Zone</button>
        </div>
        <div class="m-3">
          <button onclick="toggleTips()" type="button" class="btn btnsecondary">Tutorial</button>
          <div id="tips" class="m-3">
            <h6>
              Type Your Langitude and Longitude<br><br>
              Click on Check Containment Zone to know the Status
            </h6>
          </div>
        </div>
        <div class="m-3 float-right">
          <button type="button" class="btn btn-warning"><a href="{{url_for('data')}}">Click Here
To View The
              Containment Zones and Number of
people visited</a></button>
        </div>
      </div>

      <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.min.js"
integrity="sha384-
+YQ4JLhjyBLPDQt//I+STsc9iw4uQqACwlvpslubQzn4u2UU2UFM80nGisd026JF"
crossorigin="anonymous"></script>
      <script src="https://code.jquery.com/jquery-2.2.4.min.js"></script>
      <script
src="https://maps.google.com/maps/api/js?sensor=false&amp;libraries=places"></script>
      <script
        src="https://rawgit.com/Logicify/jquery-
locationpickerplugin/master/dist/locationpicker.jquery.js"></script>

      <script>          function
getLocation() {              if
(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(showPosition);
          } else {
            alert("No location");
          }
        }
        function showPosition(position) {
$('#map_disp').locationpicker({
            location: {                    latitude:
position.coords.latitude,
```

```
                longitude: position.coords.longitude
            }, radius: 0,
inputBinding: {
latitudeInput: $('#lat'),
                longitudeInput: $('#lon'),
            },
            enableAutocomplete: true,
            onchanged: function (currentLocation, radius, isMarkerDropped) {
                // Uncomment line below to show alert on each Location Changed event
// alert("Location changed. New location (" + currentLocation.latitude + ", " +
currentLocation.longitude + ")");
            }
        });
    }
    function toggleTips() {
        var x = document.getElementById("tips");
if (x.style.display === "none") {
x.style.display = "block";
        } else {
            x.style.display = "none";
        }
    }
  </script>

</form>
</body> </html>
```

LOGIN.HTML :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Login</title>
    <link rel="stylesheet" href="static/style.css">
    <style>          body {
background-image:
url('https://cdn.dribbble.com/users/27417/screenshots/11017927/media/2a0706f846edace3c4e81e0af
4e63807.gif');
        background-repeat: no-repeat;          background-
attachment: fixed;
        background-size: 100% 100%;

    }
    </style>
  </head>
```

```
   <body>
   <h1>Login</h1>
   <p>{{message}}</p>
   <div class="form_div">
      <form action="/login" method="POST">
         <input type="text" name="email" placeholder="Enter your email"><br>
         <input type="password" name="password" placeholder="Enter your password"><br>
         <input type="submit" value="Login">
</form>
<!--       create a sign up button for register-->
      <form action="{{url_for('register')}}" method="GET">
         <p>To create a new account</p>
         <input type="submit" value="Sign Up">
      </form>
   </div>
   </body>
```

REGISTER.HTML :

```
<!DOCTYPE html>
<html>
   <head>
      <title>Registration</title>
      <link rel="stylesheet" href="static/style.css">
      <style>
body {
          background-image: url('https://wallpaperaccess.com/full/1198250.gif');
background-repeat: no-repeat;            background-attachment: fixed;
          background-size: cover;
         }
         </style>
   </head>
   <body>
   <h1>Registration</h1>
   <p>{{message}}</p>
   <div class="form_div">
      <form action="/" method="POST">
         <input type="text" name="username" placeholder="Enter your username"><br>
         <input type="text" name="email" placeholder="Enter your email"><br>
         <input type="password" name="password" placeholder="Enter your password"><br>
<input type="password" name="confirm_password" placeholder="Confirm your password"><br>
         <input type="submit" value="Register">
      </form>
<!--       create a login field for existing user-->
      <form action="{{url_for('login')}}" method="GET">
```

```
        <p>Already have an account?</p>
        <input type="submit" value="Login">
      </form>
   </div>
   </body>
</html>
```

GITHUB ACCOUNT : https://github.com/IBM-EPBL/IBM-Project-18351-1659683556