Project Report

Team ID	**PNT2022TMID07010**
Project Name	**CONTAINMENT ZONE ALERTING APPLICATION**

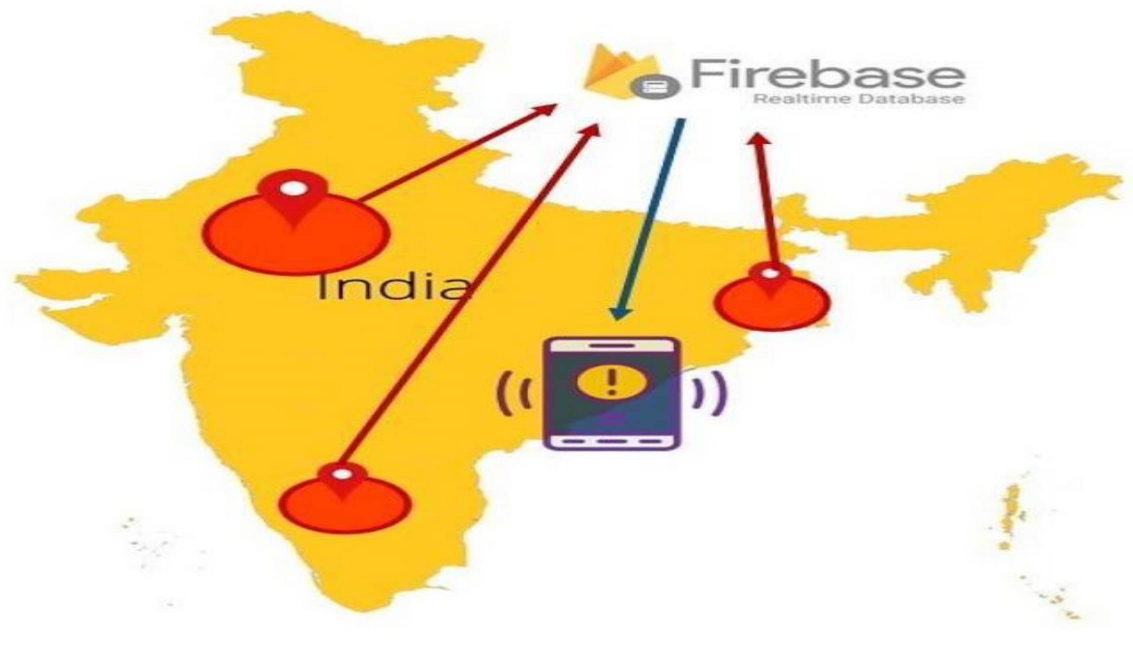
1. INTRODUCTION

1.1 Project Overview:

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 (Bahl et al. 2020). 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. In this paper, we focus on developing a mobile based application to provide information regarding the Covid-19 containment zones in West Bengal. The application further tracks the user's location and provides notification alert if the user has entered a containment zone. The application also provides daily Covid-19 case statistics to the users to keep them updated. The application is developed on Android SDK and uses Firebase Cloud Firestore to store the location data. Android's geofencing client is used to create geofences around the containment zones and notification manager is used to provide notifications. The application also uses RESTful web services to show the Covid-19 cases in West Bengal. We have tested our application with different users in different locations across West Bengal and it works efficiently and is able to attain our target.

Purpose:

The Android application shows the location of the containment zones to the users. It also notifies the user when he or she trespasses the boundary of a containment zone or stays in the containment zones



2. LITERATURE SURVEY:

2.1 Existing problem:

People doesn’t have proper knowledge about containment zones since they do change

daily and hard to keep updated and if they are not updated properly, they will lead to wide spread

of disease.

2.2 References:

PAPER 1:

TITLE: Tracking the Covid zones through geo-fencing technique

AUTHOR NAME: Anto Arockia Rosaline R ,Lalitha R ,Hariharan G ,Lokesh

PUBLICATION YEAR: 2017

DESCRIPTION:

Following the tracking of a suspicious person, the geo-fenced layer is mapped out in the vicinity, and the virtual perimeter is then employed for the subsequent trapping procedure. As soon as the Covid monitoring team updates this geo-fenced layer, the public can view it. The idea of creating a virtual perimeter region is known as geo-fencing. Effective containment zone monitoring is made possible by this virtual perimeter monitoring technology. By utilising an automated system based on wireless infrastructure, it lowers operational costs. Additionally, it promptly alerts the law enforcement to find the offenders. As a result, it facilitates the inspection of containment areas and the monitoring of those who disobey governmental regulations. Users can receive updates from the Covid team on the alert zone. The Covid team has a number of modules for suspect tracking, hotspot fencing, etc. The Covid team must seek a service from the service network provider in the case of suspect tracking, and following authorization, they will offer the coordinates. According to our telecommunication legislation, it is illegal to share data; nonetheless, exchanging personal information without the individual's knowledge via any means is occasionally allowed with governmental approval for investigative purposes.

PAPER 2:

AUTHOR NAME: Geofencing 2.0: Taking Location-based Notifications to the Next Level

PUBLICATION YEAR: 2016

DESCRIPTION:

Sandro Rodriguez Garzon Bersant Deva The basic Android application that served as the prototype Geofencing client was used. This client is primarily responsible for carrying out the geofencing server's ongoing location update strategy. This must be accomplished with little energy consumption because the Geofencing client is located on a mobile device. We made the decision to employ the low energy Geofencing features of the Android operating system to keep an eye on the safety zone. As a result, a safety zone is considered as a single circular geofence with a required exit on the mobile device. However, they discovered that there was occasionally a significant lag time between leaving the safety zone and receiving a notification from the system about the leave. In order to address this issue, a specific amount of the safety zone's radius is decreased. While the safety zone and how it is implemented have a significant impact on overall energy consumption, it is also important to make the right choice when it comes to a placement mechanism. In order to reduce power consumption without compromising the necessary position precision, they used a device-based smart combination of various positioning mechanisms introduced by. By temporarily deactivating placement when a device is not in motion, the Geofencing client also makes use of cutting-edge mobile sensing capabilities integrated into the Android operating system's activity recognition unit. Mobile users who live close to a geo-border fence's will find this to be of particular utility. If the Geofencing server notifies the Geofencing client about a geo-notice, the notification will appear right away.

PAPER 3

TITLE: Development of An Android Application for Viewing Covid19 Containment Zones Alerting.

AUTHOR NAME: India Ranajoy Mallik, Amlan Protim Hazarika, Sudarshana Ghosh Dastidar, Dilip Sing & Rajib Bandyopadhyay

PUBLICATION YEAR: 2019

DESCRIPTION:

The World Health Organization has declared the outbreak of the novel coronavirus, Covid-19 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 hotspots, 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. 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 IMEI number to the online database. To achieve all these functionalities, many tools, and APIs from Google like Firebase and Geofencing API are used in this application. 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.

PAPER 4:

TITLE: Aarogya Setu

AUTHOR NAME: National Informatics Centre, Ministry of Electronics & Information Technology, Government of India

PUBLICATION YEAR: 2014

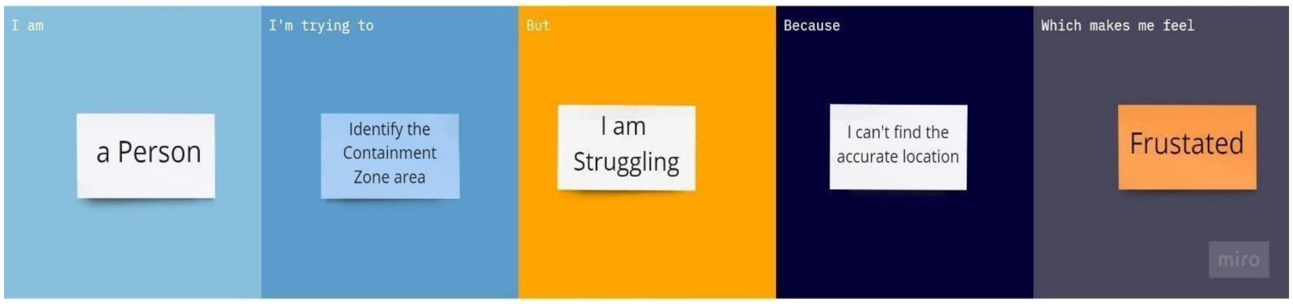
DESCRIPTION:

The most popular containment zone alert application among the options currently in use in India is called Aarogya Setu. The Indian government created a mobile application to link the public with crucial health services. Its primary features include

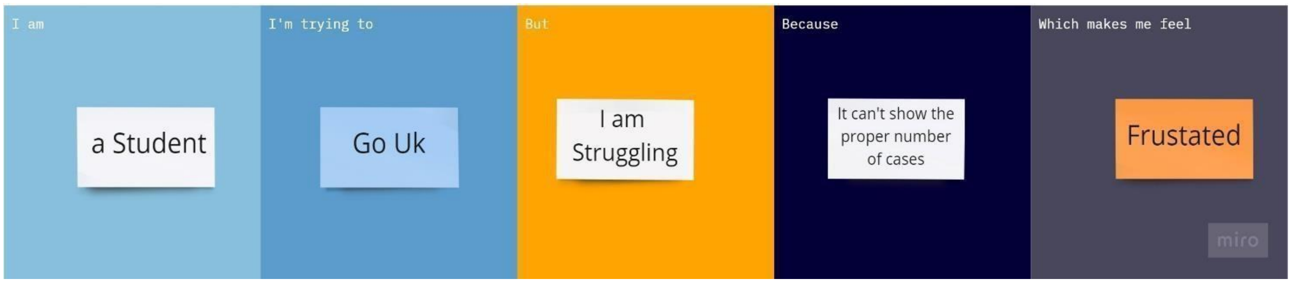
geo-location-based COVID19 data, user risk status, automatic contact tracing using Bluetooth, and much more. The movement of an infected individual is tracked using Bluetooth and GPS technology, and the system notifies the public of the locations the infected person has visited while designating those locations as vulnerable ones. It employs cellular triangulation to determine a person's location in the absence of GPS technology. While Aarogya Setu can track down contacts and notify those who have come into touch with someone who has COVID-19, it also actively keeps track of quarantine or containment zones and alerts users who enter them. The Terms of Use and Privacy Policy must be accepted at the time of registration when installing the application on any Android or iOS mobile device, and ongoing use of the application denotes continued acceptance. Name, age, sex, occupation, phone number, overseas travel within the previous 28–45 days, and whether the user is a smoker are all pieces of information that the app gathers. This data is kept on a server that is under the jurisdiction of the Indian government. It is hashed and sent to the user's mobile application along with a special digital ID (DID). The user is recognised using the DID. In order for the user's mobile phone to exchange information with another device that has the app when it gets within range, the Bluetooth and GPS services must be turned on. Their individual IDs, along with the time and GPS location, are kept on the two phones when two users come into close proximity. The format in which this data is kept is encrypted. Only after a person tests positive is it posted to the government-controlled servers of the app.

2.3.Problem Statement Definition:

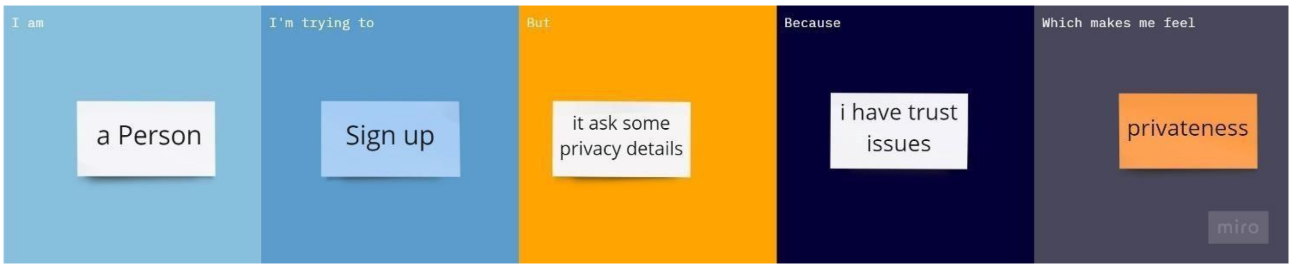
2.2.PROBLEM STATEMENT 1:



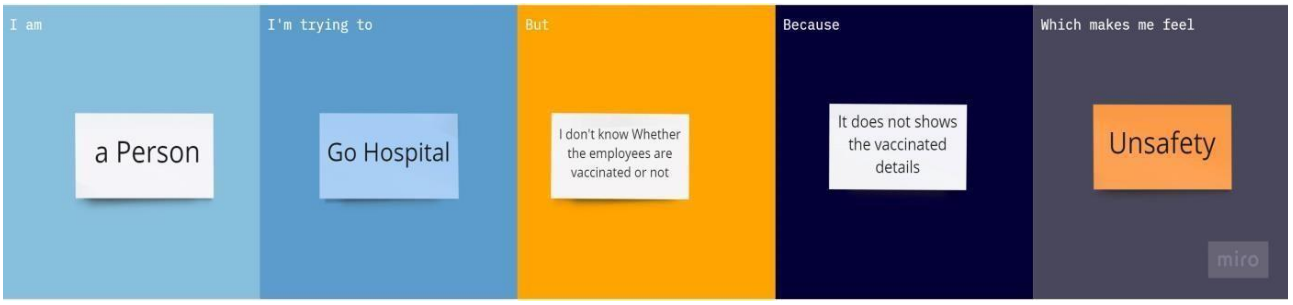
PROBLEM STATEMENT 2:



PROBLEM STATEMENT 3:



PROBLEM STATEMENT 4:


I am
a Person
I'm trying to
Go Hospital
But
I don't know Whether the employees are vaccinated or not
Because
It does not shows the vaccinated details
Which makes me feel
Unsafety
miro

3.IDEATION & PROPOSED SOLUTION

3.1 Proposed Solution Fit

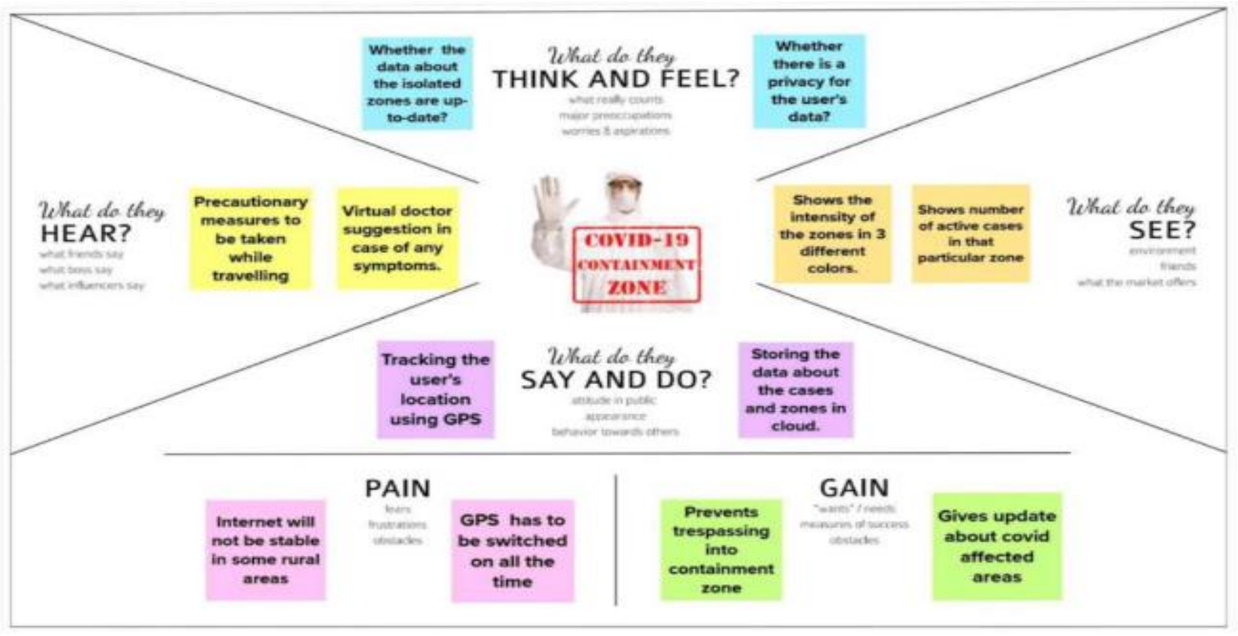
S.NO	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	This application is intended to provide information about containment zones in a particular region by alerting people, through continuous monitoring of an individual's location. Key benefits of the application are monitoring people's activity and alerting them of their safety movements
2.	Idea / Solution description	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. Location of the individual must be stored in the

		Database. Alerts are sent using the notification service.
3.	Novelty / Uniqueness	The uniqueness of containment zone alerting app is it shows the particular area of the district before the 100m,and the user's location history is stored in database and this app provides the precautions measurements ,list of immunity boosters, location of the vaccination providing places . it also gives the lis of the affected and admitted patients and distarchged patients ,percentage of affecting by covid19
4.	Social Impact / Customer Satisfaction	Social Stigma is discrimination against a particular group of people, a place, or a nation in the form of a negative attitude. Public health emergencies (such as COVID-19 pandemic) are stressful situations for people and communities. Fear and anxiety with a lack of knowledge about the disease can lead to social .
5.	Business Model (Revenue Model)	We are going to add personal health tracker in subscription basis .so they can manage their health efficiently.
6.	Scalability of the Solution	In this modern world eventhough the covid pandemic threat is about to end there are high chance of pandemic or endemic .so this application is very useful in that situation and we can use this application in seasonal diseases

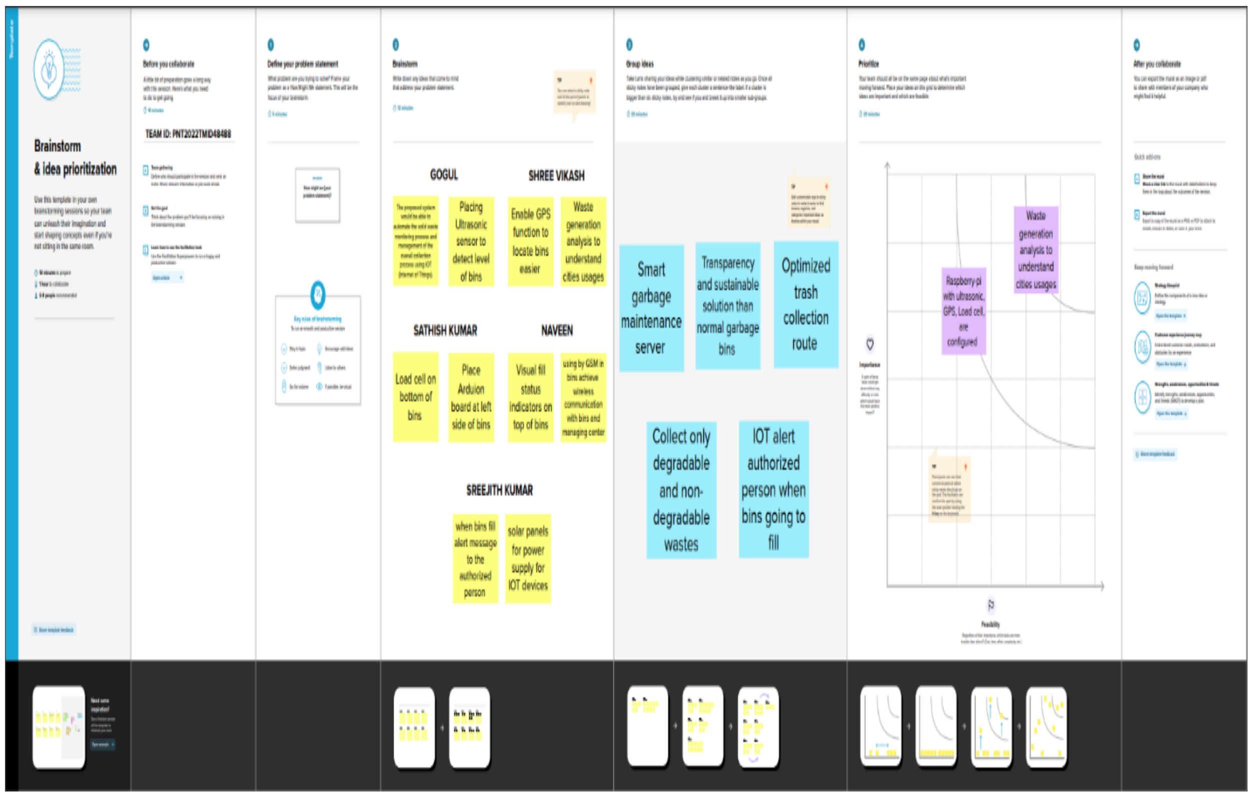
PNT2022TMID07010

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


Whether the data about the isolated zones are up-to-date?
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Whether there is a privacy for the user's data?
What do they HEAR?
what friends say
what boss say
what influencers say
Precautionary measures to be taken while travelling
Virtual doctor suggestion in case of any symptoms.
COVID-19 CONTAINMENT ZONE
Shows the intensity of the zones in 3 different colors.
Shows number of active cases in that particular zone
What do they SEE?
environment
friends
what the market offers
Tracking the user'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 not be stable in some rural areas
GPS has to be switched on all the time
GAIN
"wants" / needs
measures of success
obstacles
Prevents trespassing into containment zone
Gives update about covid affected areas

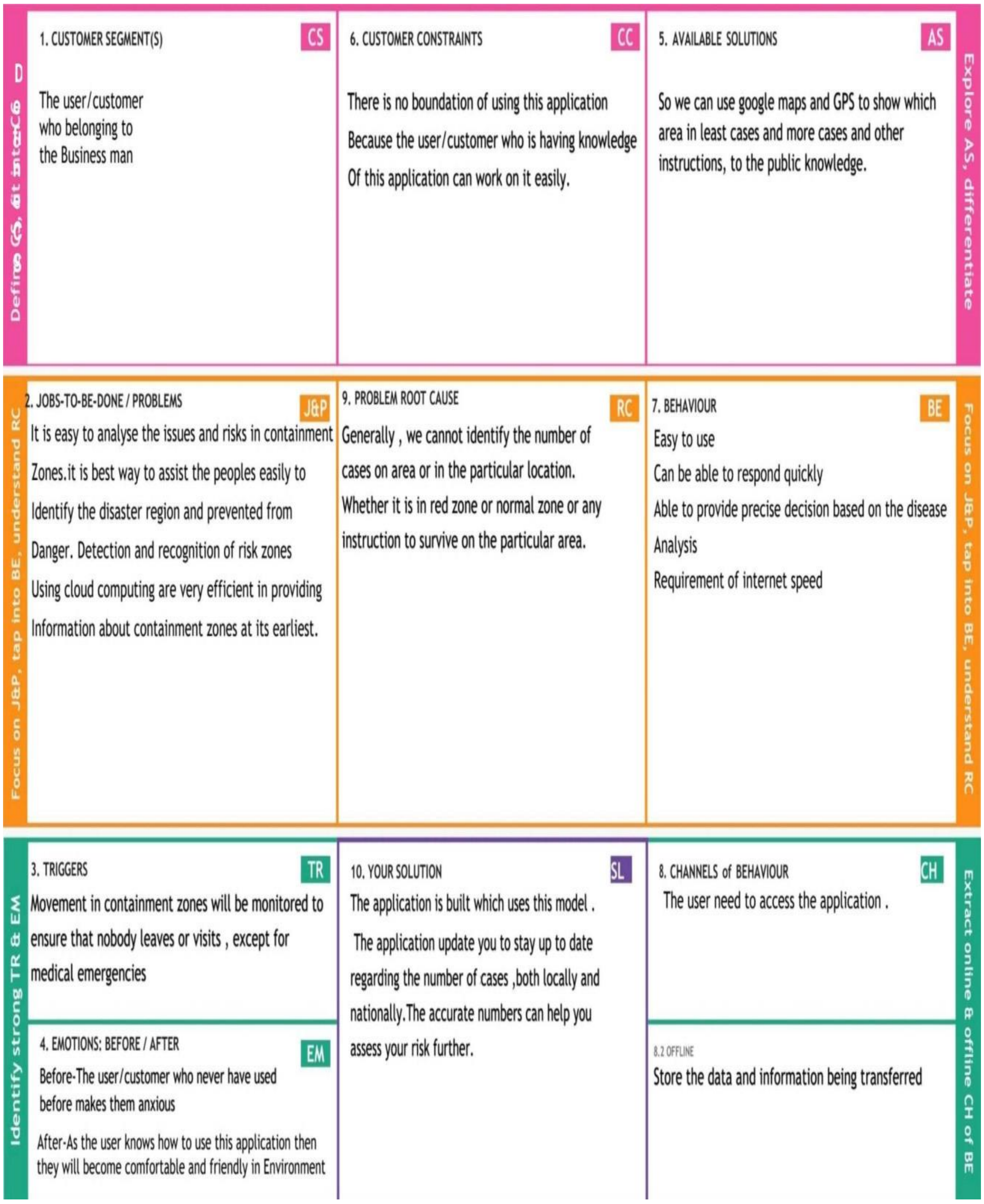
3.3 Ideation & Brainstorming


Brainstorm & idea prioritization
Before you collaborate
TEAM ID: PNT2022TMID48488
Define your problem statement
Brainstorm
GOGUL
SHREE VIKASH
Placing Ultrasonic sensor to detect level of bins
Enable GPS function to locate bins easier
Waste generation analysis to understand cities usages
SATHISH KUMAR
NAVEEN
Load cell on bottom of bins
Place Arduion board at left side of bins
Visual fill status indicators on top of bins
SREEJITH KUMAR
when bins fill alert message to the authorized person
solar panels for power supply for IOT devices
Group ideas
Smart garbage maintenance server
Transparency and sustainable solution than normal garbage bins
Optimized trash collection route
Collect only degradable and non-degradable wastes
IOT alert authorized person when bins going to fill
Prioritize
Raspberry-pi with ultrasonic, GPS, Load cell, are configured
Waste generation analysis to understand cities usages
Importance
Feasibility
After you collaborate

PNT2022TMID07010

3.4 Problem Solution fit

Problem-Solution fit canvas 2.0 **Purpose/ Vision**



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

The user/customer who belonging to the Business man

6. CUSTOMER CONSTRAINTS CC

There is no boundation of using this application Because the user/customer who is having knowledge Of this application can work on it easily.

5. AVAILABLE SOLUTIONS AS

So we can use google maps and GPS to show which area in least cases and more cases and other instructions, to the public knowledge.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

It is easy to analyse the issues and risks in containment Zones.it is best way to assist the peoples easily to Identify the disaster region and prevented from Danger. Detection and recognition of risk zones Using cloud computing are very efficient in providing Information about containment zones at its earliest.

9. PROBLEM ROOT CAUSE RC

Generally , we cannot identify the number of cases on area or in the particular location. Whether it is in red zone or normal zone or any instruction to survive on the particular area.

7. BEHAVIOUR BE

Easy to use

Can be able to respond quickly

Able to provide precise decision based on the disease Analysis

Requirement of internet speed

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Movement in containment zones will be monitored to ensure that nobody leaves or visits , except for medical emergencies

4. EMOTIONS: BEFORE / AFTER EM

Before-The user/customer who never have used before makes them anxious

After-As the user knows how to use this application then they will become comfortable and friendly in Environment

10. YOUR SOLUTION SL

The application is built which uses this model .

The application update you to stay up to date regarding the number of cases ,both locally and nationally.The accurate numbers can help you assess your risk further.

8. CHANNELS of BEHAVIOUR CH

The user need to access the application .

8.2 OFFLINE

Store the data and information being transferred

Extract online & offline CH of BE

4.REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Registration	Registration through Gmail. Registration through mobile number.
FR-2	User Confirmation	Confirmation via Email. Confirmation via OTP.
FR-3	Authentication	It checking the confirmation of the password.
FR-4	Business rule	For subscriber's we give first 3 day's free trail. For unsubscriber's the user needs to watch some advertisement for knowing the zone alert for first 3 day's. **FR No. FR No.**

4.2 Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Providing recommendation link by using customer preference .
NFR-2	**Security**	The software team will issue some strong security code for the user's.
NFR-3	**Reliability**	The database update process must rollback all related updates when any update fails.
NFR-4	**Performance**	The loading speed of the server is quick and fast.

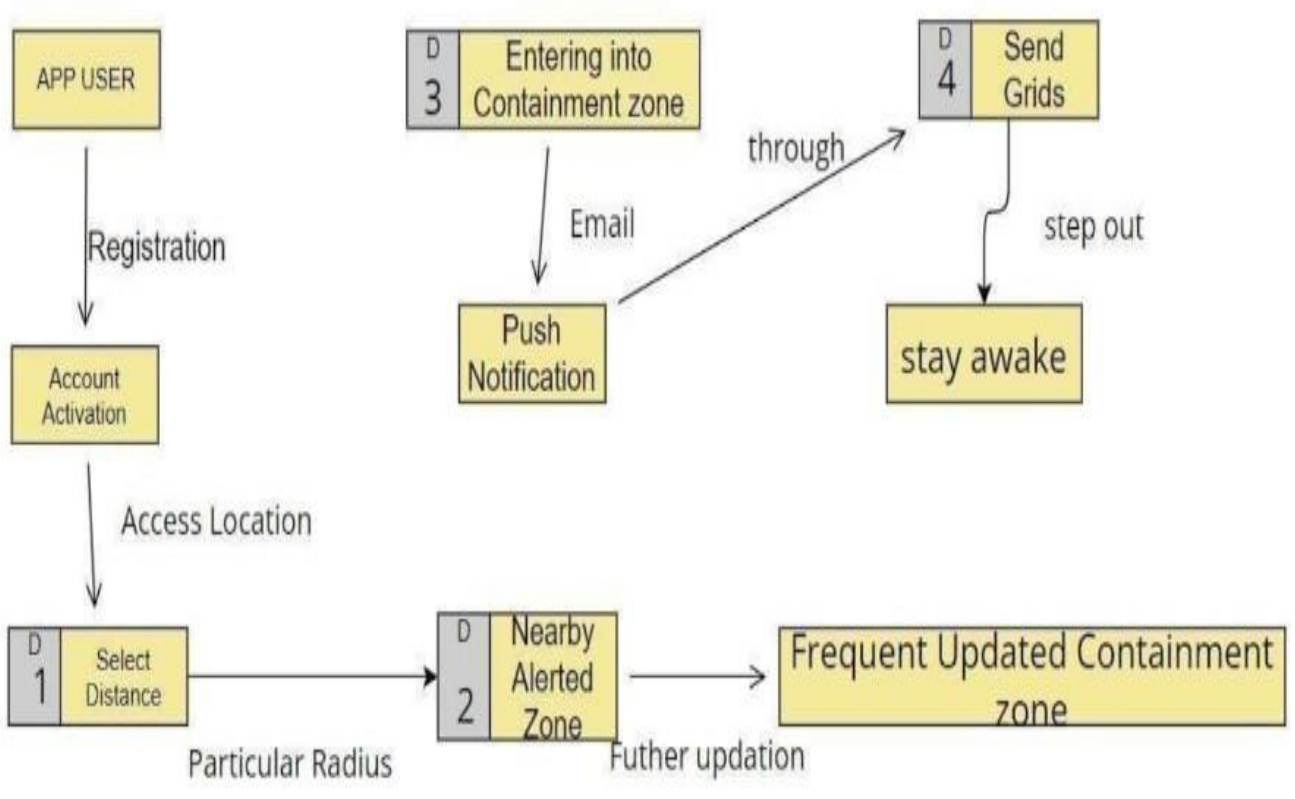
5.PROJECT DESIGN

Data Flow Diagrams

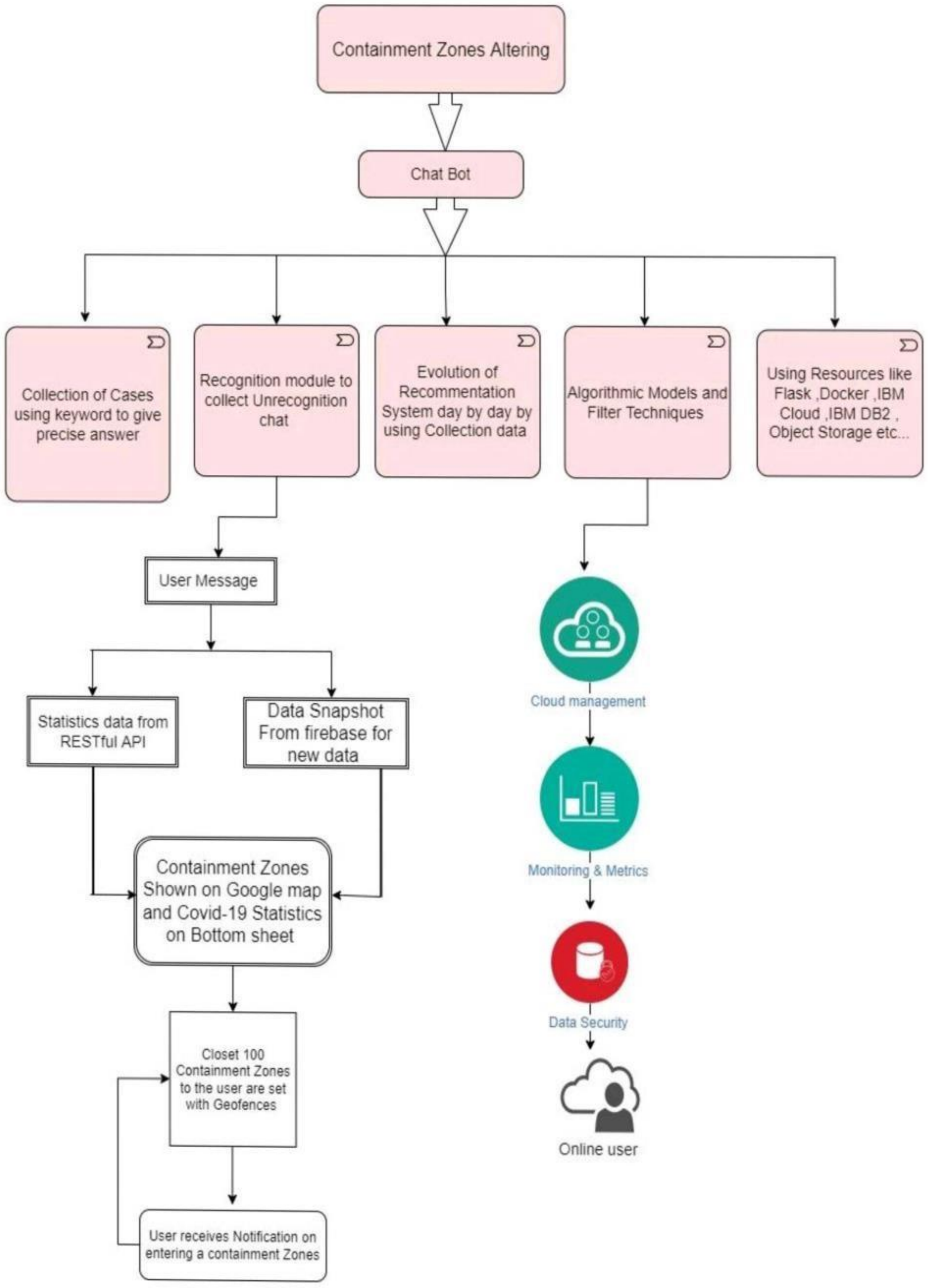
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

5.1 Data flow diagram:



5.2.SOLUTION ARCHITECURE:


Containment Zones Altering
Chat Bot
Collection of Cases using keyword to give precise answer
Recognition module to collect Unrecognition chat
Evolution of Recommentation System day by day by using Collection data
Algorithmic Models and Filter Techniques
Using Resources like Flask ,Docker ,IBM Cloud ,IBM DB2 , Object Storage etc...
User Message
Statistics data from RESTful API
Data Snapshot From firebase for new data
Containment Zones Shown on Google map and Covid-19 Statistics on Bottom sheet
Closet 100 Containment Zones to the user are set with Geofences
User receives Notification on entering a containment Zones
Cloud management
Monitoring & Metrics
Data Security
Online user

TECHNICAL ARCHITECTURE:

S.no	Component	Description	Technology
1.	User Interface	Mobile Application	HTML, CSS, JavaScript.
2.	Application Logic	Logic for a process in the application	Javascript
3.	Database	Data Type, Configurations etc.	Firebase, ibm cloud
4.	Cloud Database	Database Service on Cloud	IBM Cloud
5.	File Storage	File storage requirements	Local Filesystem and IBM cloud
6.	Infrastructure (Server / Cloud)	Application Deployment on Cloud Local Server Configuration	Local and Cloud Foundry

5.2Table-1: Components & Technologies:

Application Characteristics:

S.no	Characteristics	Description	Technology
1.	Open-Source Frameworks	GitHub	Internet hosting service
2.	Security Implementations	Application security: Veracode.	Network automation
3.	Scalable Architecture	It provides the room for expansion more database of smart bins added additionally can be updated.	Cloud storage
4.	Availability	As the system control is connected to web server it is available 24*7 and can be accessed whenever needed.	Server, Appleixe, reple
5.	Performance	Performance is high it uses 5mb caches	Wireless Sensor Network

5.3 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
Login	Registration (web and android)	USN-1	I can register for the application by entering my email and password	I can control my online account and dashboard.	Medium	Sprint-1
Sign Up	Registration (web and android)	USN-2	I will receive a confirmation email once I have registered for the application	I can handle the waste collection.	High	Sprint-1
Services	Dashboard	USN-3	need to give permission to access my location	I can take the shortest path to reach the waste filled route specified.	Medium	Sprint-2
Services	Service	USN-4	I need to differentiate the containment zones	I can collect the trach, pull it to the truck, and send it out.	Medium	Sprint-3
Data collection	Service	USN-5	. I need to alert the user when they enter the containment zone through the notification	All of these processes are under my control.	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Project 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.	19 OCTOBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	18 OCTOBER 2022

Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	18 OCTOBER 2022

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint 1	Registration (web and android)	USN-1	USER: I can register for the application by entering my email and password	3	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
		USN-2	USER: I will receive a confirmation email once I have registered for the application	2	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
	Login (web and android)	USN-3	USER: I can log into the application	3	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team members
Sprint-2	Dashboard	USN-4	USER: need to give permission to access my location	5	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
		USN-5	As a user, I can log into the application by entering email & password	5	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team members
Sprint 3	Service	USN 6	ADMIN: I need to update the containment zones.	5	High	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
		USN 7	ADMIN: I need to differentiate the containment zones based on the intensity of infection.	3	Medium	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya

Sprint 4	Service	USN 8	ADMIN: I need to alert the user when they enter the containment zone through the notification	5	Medium	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
	Data collection	USN 9	ADMIN: I need to store user details on the cloud	5	Medium	Atchaya Jenitha Priyadharshini Priyadharshini Kowsalya
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team members
		USN 10	ADMIN: I need to collect details about covid -19 cases from verified sources	5	Priority	Team members

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	7 Days	25 Oct 2022	31 Oct 2022	20	31 Oct 2022
Sprint-2	20	6 Days	01 Nov 2022	06 Nov 2022	20	06 Nov 2022
Sprint-3	20	5 Days	07 Nov 2022	11 Nov 2022	20	11 Nov 2022
Sprint-4	20	6 Days	12 Nov 2022	17 Nov 2022	20	17 Nov 2022

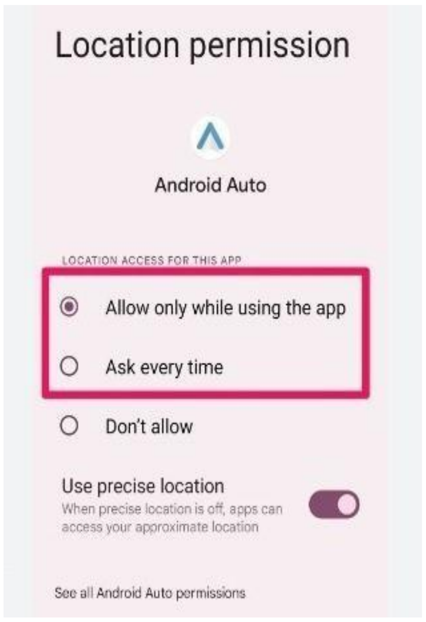
Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint).
Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

6.2. Sprint Delivery Schedule

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)

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

7.CODING & SOLUTIONING



Location permission

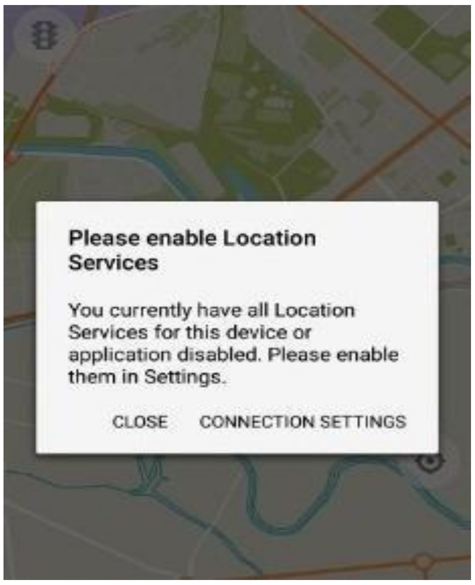
Android Auto

LOCATION ACCESS FOR THIS APP

- [✓ Allow only while using the app]
- Ask every time
- Don't allow

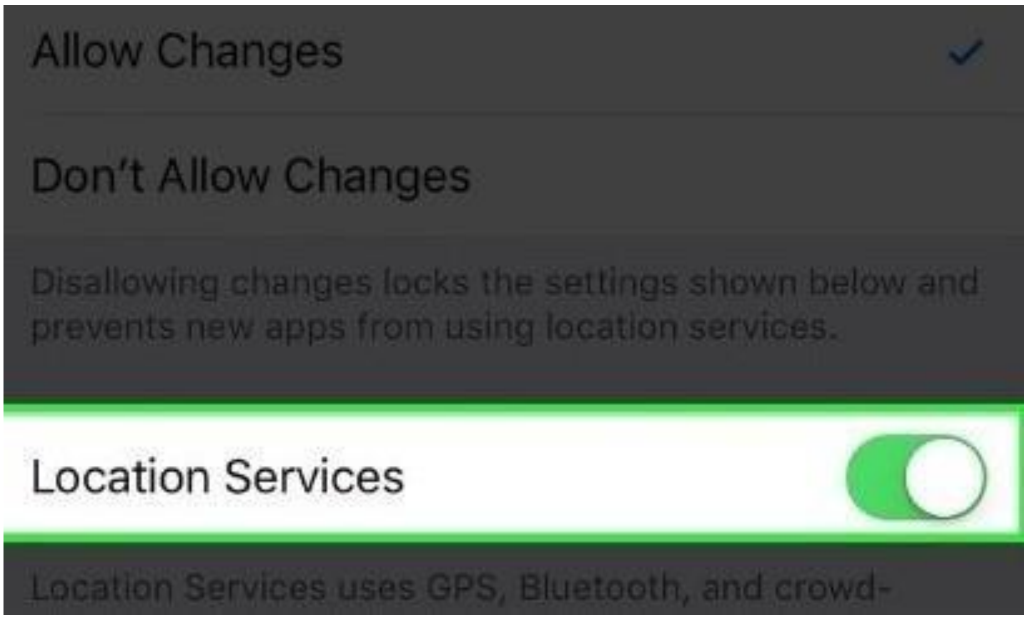
Use precise location
When precise location is off, apps can access your approximate location

See all Android Auto permissions



Please enable Location Services

You currently have all Location Services for this device or application disabled. Please enable them in Settings.



Allow Changes ✓

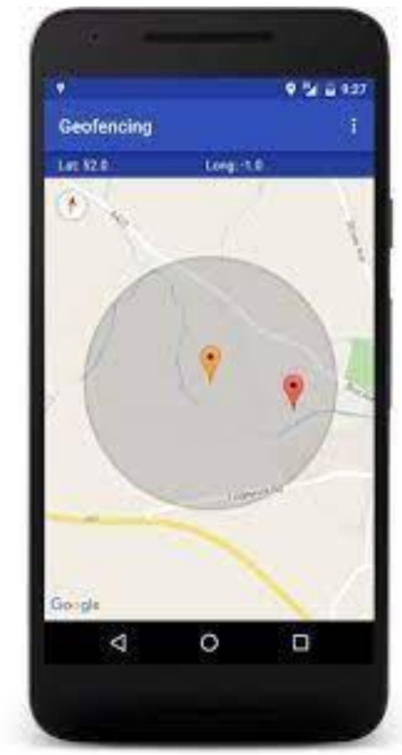
Don't Allow Changes

Disallowing changes locks the settings shown below and prevents new apps from using location services.

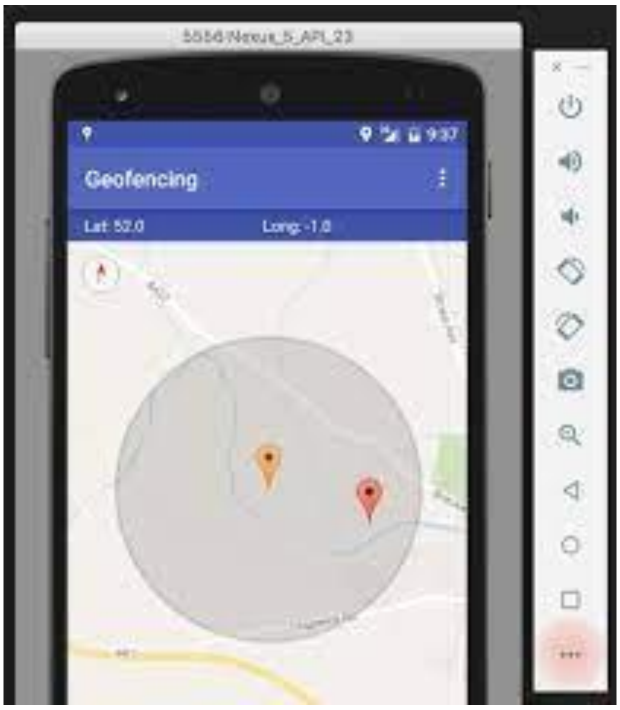
Location Services

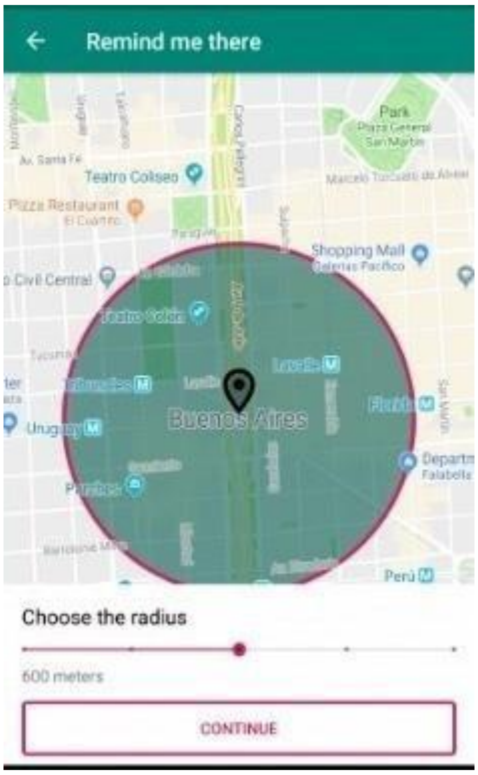
Location Services uses GPS, Bluetooth, and crowd-

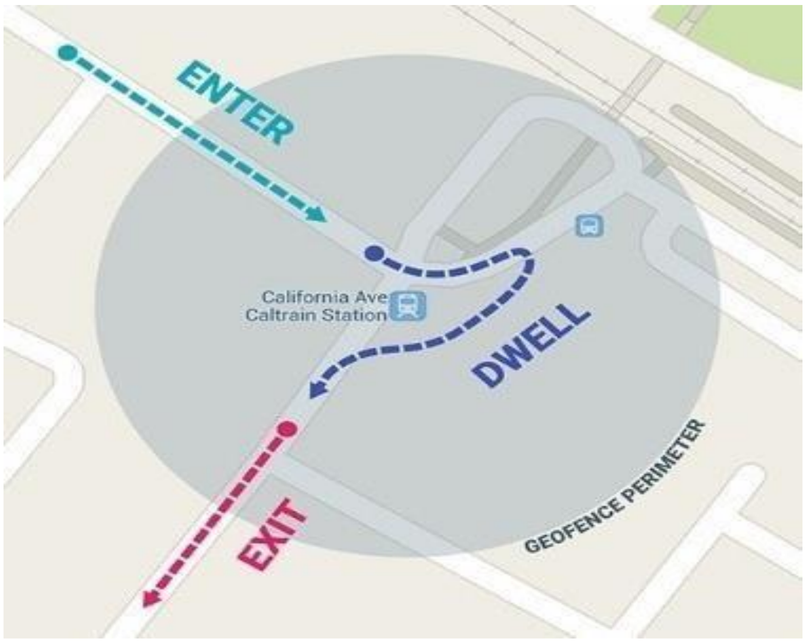
GEOFENCE IN ANDROID APP :



PNT2022TMID07010




Remind me there
Buenos Aires
Choose the radius
600 meters
CONTINUE


ENTER
DWELL
EXIT
GEOFENCE PERIMETER
California Ave Caltrain Station

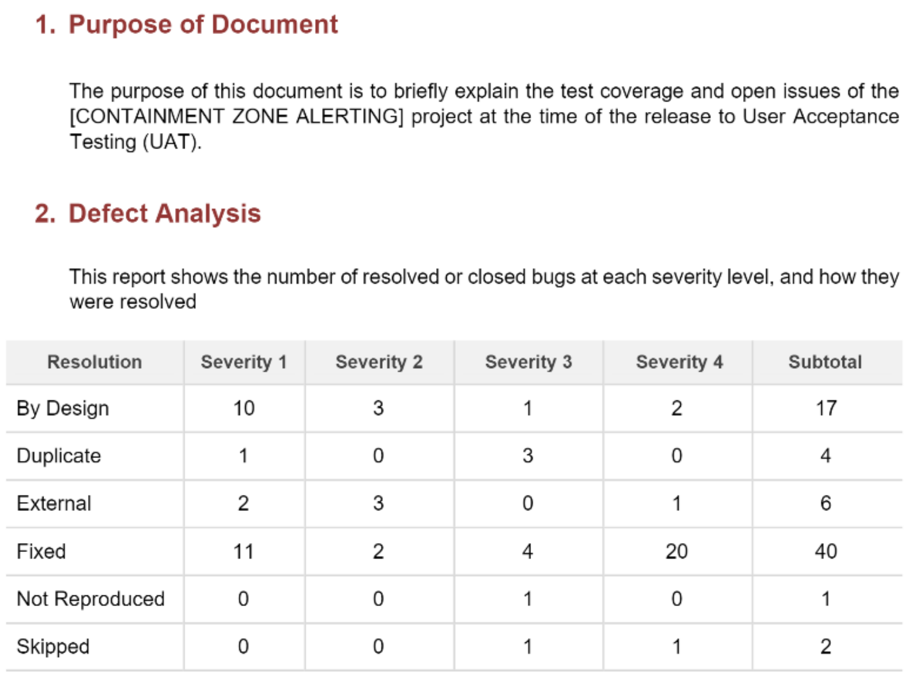
8. Testing

8.1 Testing Performance

Purpose of Test Case

The Page should even work at the worst case scenario and should not produce any kind of error either client or server error. This error includes all the test cases right from the Login Page to the fetch of the details of the containment zone.

8.2 User Acceptance Testing



1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [CONTAINMENT ZONE ALERTING] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	3	1	2	17
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	40
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2

External	2	3	0	1	6
Fixed	11	2	4	20	40
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	13	12	25	78

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

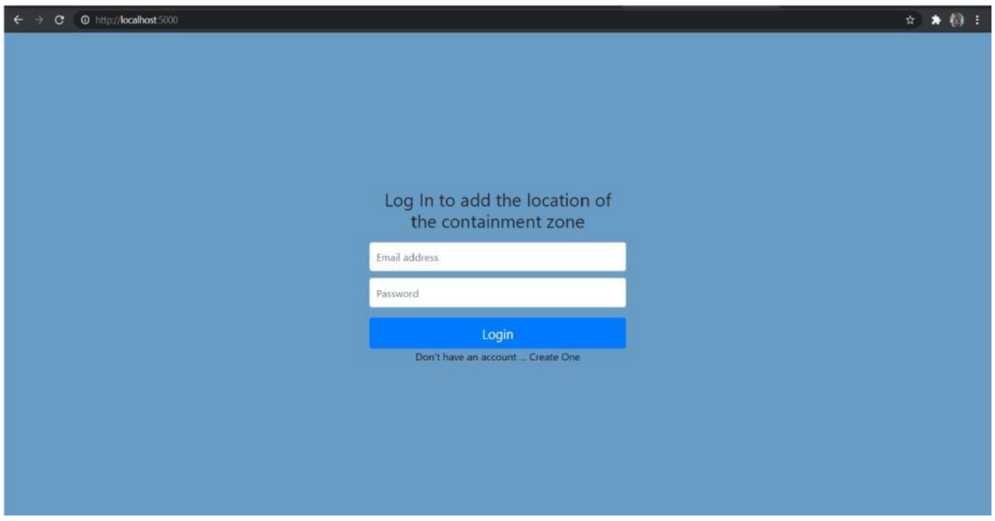
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Security	2	0	0	2

9.RESULTS:

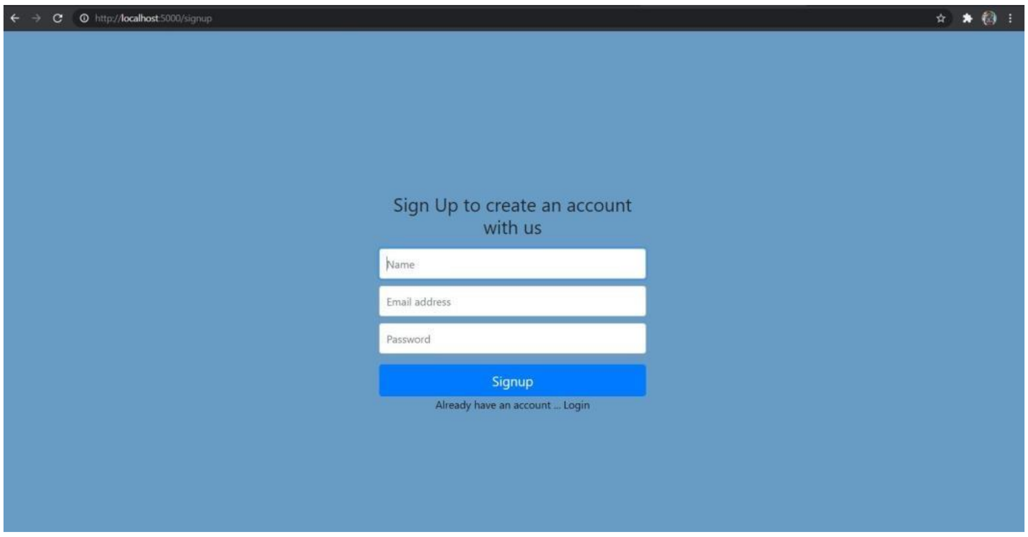
9.1 Performance Testing:

Admin App:

Login Page:


http://localhost:5000
Log In to add the location of the containment zone
Email address
Password
Login
Don't have an account ... Create One

Register page:


http://localhost:5000/signup
Signup
Already have an account ... Login

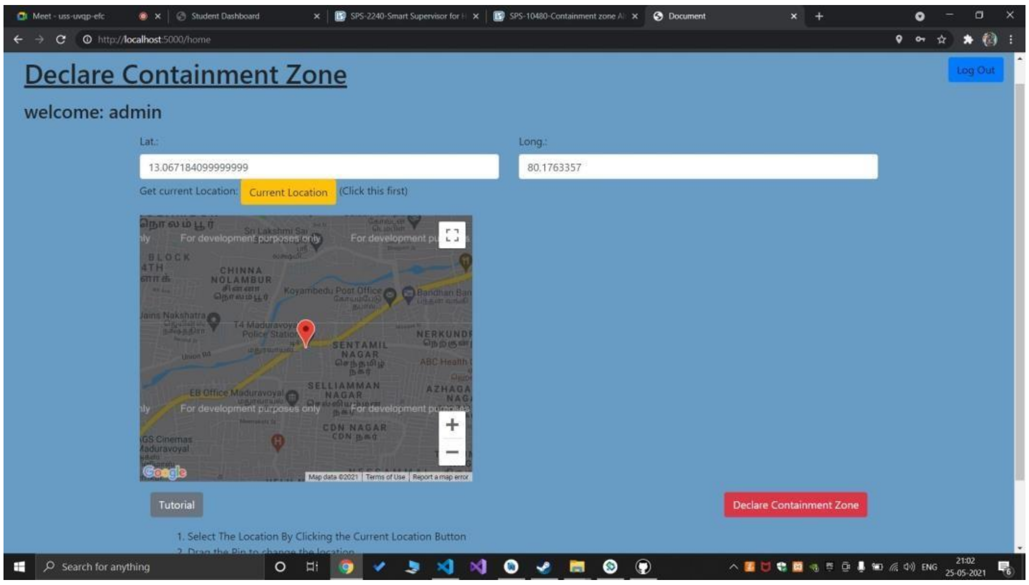
Sign Up to create an account with us

Name

Email address

Password

Home page:



Declare Containment Zone

welcome: admin

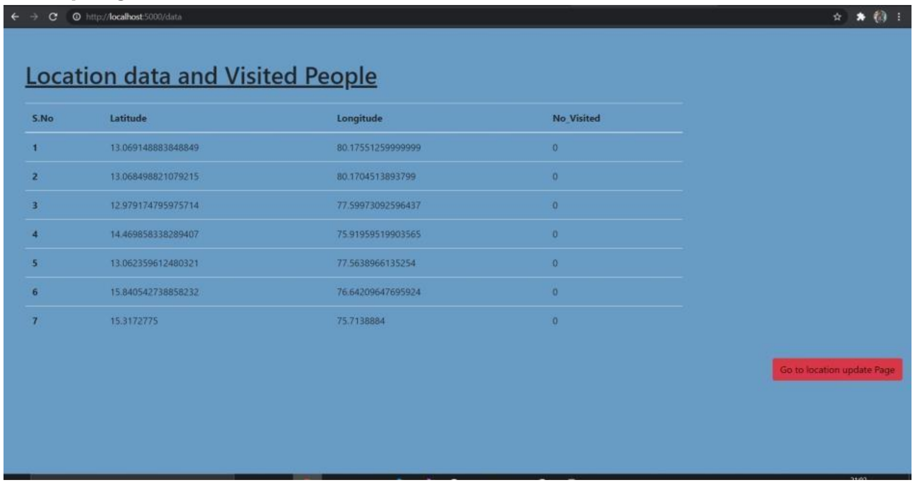
Lat.: 13.067184099999999

Long.: 80.1763357

Get current Location: Current Location (Click this first)

1. Select The Location By Clicking the Current Location Button

Location data page:

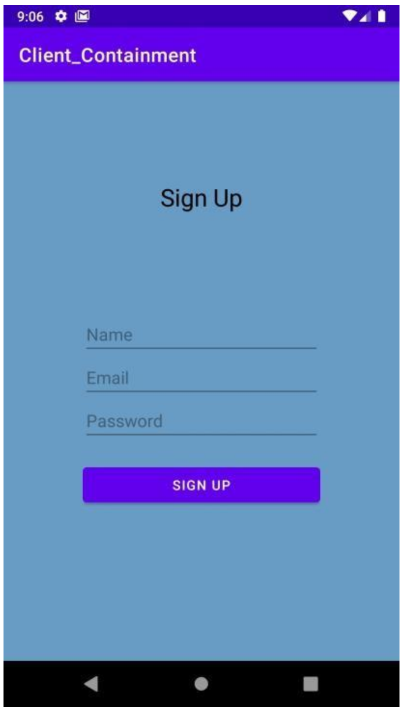


Location data and Visited People

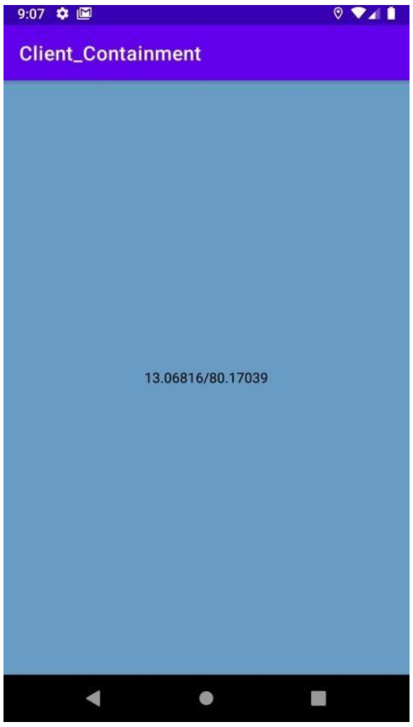
S.No	Latitude	Longitude	No_Visited
1	13.069148883848849	80.17551259999999	0
2	13.068498821079215	80.1704513893799	0
3	12.979174795975714	77.59973092596437	0
4	14.469858338289407	75.91959519903565	0
5	13.062359612480321	77.5638966135254	0
6	15.840542738858232	76.64209647695924	0
7	15.3172775	75.7138884	0

Client Application:

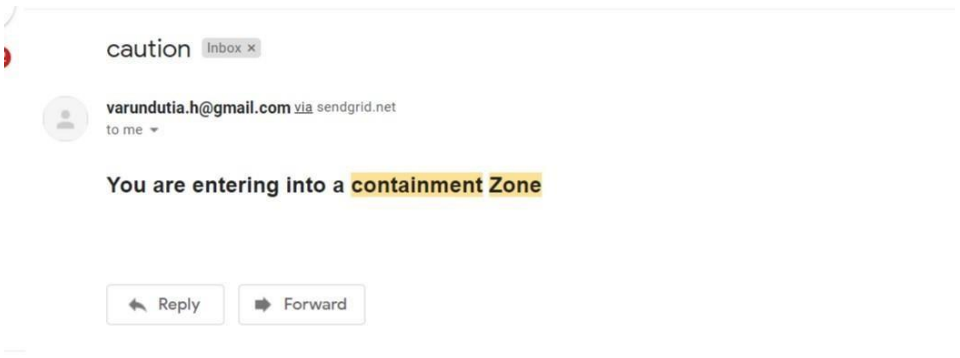
Register screen:



Current Location:



An Email will be sent to the registered mail id if the location is within 100 meters of the locations present in the admin app.



caution Inbox ×

varundutia.h@gmail.com via sendgrid.net
to me

You are entering into a containment Zone

10. ADVANTAGES & DISADVANTAGES

<u>ADVANTAGES:</u>

- People can be alerted before entering containment zone.
- Further spread of virus can be reduced considerably.

<u>DISADVANTAGES:</u>

- Accuracy of application depends on the number of data given to the application.
- Application’s accuracy is directly proportional to the number of data given to the application
- about the infected patients.

11.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.

12.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.

13 APPENDIX

Source Code

```
# Project : CONTAINMENT ZONE ALERTING APPLICATION
# Team ID : PNT2022TMID07010

APP.PY
from logging import error from flask import *
from jinja2.utils import select_autoescape import bcrypt
from flask_mysqldb import MySQL
```

```
import json
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail
 # initialization
app = Flask(__name__)
 # config
app.secret_key                                                                =
"\x19Ts\xbe\xe7\x8c_\r\x12Q\x14\x13>q\xb7'WTH0\x9f\xe4\xec\xb1"
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_DB'] = 'zone2'
mysql = MySQL(app)
 # functions
 def send_mail(email):
    print(email)
    message = Mail(from_email='varundutia.h@gmail.com',
to_emails=email,
subject='caution',
  plain_text_content='Please Stay Safe',
html_content='<h2>You are entering into a containment Zone</h2>')
     try:
        sg = SendGridAPIClient(
'SG.7BJDtQDlS8unH0r5_TufVQ.Ykpcz19QcqgcNwYZC3a0mNRPhGksG117YURqOTa
2HL')        response = sg.send(message)
print(response.status.code)
print(response.body)
print(response.headers)
except Exception as e:
print(e)
def create_bcrypt_hash(password):     # convert the string to bytes
    password_bytes = password.encode()
    # generate a salt
    salt = bcrypt.gensalt(14)     # calculate a hash as bytes

    password_hash_bytes = bcrypt.hashpw(password_bytes, salt)

    # decode bytes to a string

    password_hash_str = password_hash_bytes.decode()     return password_hash_str


def verify_password(password, hash_from_database):

    password_bytes = password.encode()    hash_bytes = hash_from_database.encode()


    # this will automatically retrieve the salt from the hash,

    # then combine it with the password (parameter 1)    # and then hash that, and compare it to the user's hash   does_match = bcrypt.checkpw(password_bytes, hash_bytes)


    return does_match


# Api's


@app.route("/", methods=["GET", "POST"]) def login():     if(request.method == "POST"):


        # get the data from the form        password = request.form['password']        email = request.form['email']
        # initialize the cursor
        signup_cursor = mysql.connection.cursor()


        # check whether user already exists        user_result = signup_cursor.execute(
            "SELECT * FROM USERS WHERE user_email=%s", [email]
        )


        if(user_result > 0):
            data          =          signup_cursor.fetchone()          data_password          =
data[3]          if(verify_password(password, data_password)):
                signup_cursor.close()          session['id']          =
data[0]          session['name']          =          data[1]          session['email']          =
data[2]          return redirect(url_for("home"))          else:
                return render_template('login.html', error=1)          else:
            return          render_template('login.html',          error=2)     return
render_template('login.html', error=3)


@app.route("/signup", methods=["POST", "GET"])
```

```
def verify_password(password, hash_from_database):
password_bytes = password.encode()
hash_bytes = hash_from_database.encode()
# this will automatically retrieve the salt from the hash,
# then combine it with the password (parameter 1)
# and then hash that, and compare it to the user's hash
does_match = bcrypt.checkpw(password_bytes, hash_bytes)
return does_match
# Api's
@app.route("/", methods=["GET", "POST"])
def login():
if(request.method == "POST"):
# get the data from the form
password = request.form['password']
email = request.form['email']
# initialize the cursor
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS WHERE user_email=%s", [email]
)
if(user_result > 0):
data = signup_cursor.fetchone()
data_password = data[3]
if(verify_password(password, data_password)):
signup_cursor.close()
session['id'] = data[0]
session['name'] = data[1]
session['email'] = data[2]
return redirect(url_for("home"))
else:
return render_template('login.html', error=1)
else:
return render_template('login.html', error=2)
return render_template('login.html', error=3)
@app.route("/signup", methods=["POST", "GET"])

def create_bcrypt_hash(password):
# convert the string to bytes
password_bytes = password.encode()
# generate a salt
salt = bcrypt.gensalt(14)
# calculate a hash as bytes
password_hash_bytes = bcrypt.hashpw(password_bytes, salt)
# decode bytes to a string
password_hash_str = password_hash_bytes.decode()
return password_hash_str
PNT2022TMID07010
def verify_password(password, hash_from_database):
password_bytes = password.encode()
hash_bytes = hash_from_database.encode()
# this will automatically retrieve the salt from the hash,
# then combine it with the password (parameter 1)
# and then hash that, and compare it to the user's hash
does_match = bcrypt.checkpw(password_bytes, hash_bytes)
return does_match
# Api's
@app.route("/", methods=["GET", "POST"])
def login():
if(request.method == "POST"):
# get the data from the form
password = request.form['password']
email = request.form['email']
# initialize the cursor
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS WHERE user_email=%s", [email]
)
if(user_result > 0):
data = signup_cursor.fetchone()
data_password = data[3]
if(verify_password(password, data_password)):
signup_cursor.close()
session['id'] = data[0]
session['name'] = data[1]
session['email'] = data[2]
return redirect(url_for("home"))
else:
return render_template('login.html', error=1)
else:
return render_template('login.html', error=2)
return render_template('login.html', error=3)
@app.route("/signup", methods=["POST", "GET"])
def signup():
if(request.method == "POST"):

    # get the data from the form
name = request.form['name']
email = request.form['email']
    password = request.form['password']

    # hash the password
    pw_hash = create_bcrypt_hash(password)

    # initialize the cursor
    signup_cursor = mysql.connection.cursor()

    # check whether user already exists
user_result = signup_cursor.execute(
      "SELECT * FROM USERS WHERE user_email=%s", [email]
    )
    if(user_result > 0):
      signup_cursor.close()
```

```
            return render_template('signup.html', error=True)
else:
            # execute the query
signup_cursor.execute(
                'INSERT              INTO              USERS(user_name,user_email,user_password,user_type)
VALUES(%s,%s,%s,%s)', (
                    name, email, str(pw_hash), "2"
                )
            )

            mysql.connection.commit()
signup_cursor.close()
            return redirect(url_for('login'))

    return render_template('signup.html', error=False)

@app.route("/home", methods=["POST", "GET"])
def home():
if(session['id'] == None):
        return redirect(url_for('login'))
def upload():
if(request.method == "POST"):
# get the data from the form
name = request.json['name']
email = request.json['email']
password = request.json['password']
# hash the password
pw_hash = create_bcrypt_hash(password)
# initialize the cursor
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS WHERE user_email=%s", [email]
)
if(user_result > 0):
signup_cursor.close()
return {'status': 'failure'}
else:
# execute the query
signup_cursor.execute(
'INSERT INTO USERS(user_name,user_email,user_password,user_type)
VALUES(%s,%s,%s,%s)', (
name, email, str(pw_hash), "1"
)
)
mysql.connection.commit()
id_result = signup_cursor.execute(
'SELECT user_id FROM USERS WHERE user_email = %s', [email]
)
if(id_result > 0):
id = signup_cursor.fetchone()
return {"id": id[0]}
```

```
signup_cursor.close()
return {"status": "failure"}
@app.route("/get_all_users")
def getusers():
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS"

   if(request.method == "POST"):
      # get data
      lat = request.form["lat"]
      lon = request.form["lon"]
vis = 0
  if(lat == "" or lon == ""):
return render_template('home.html', name=session['name'], email=session['email'],
id=session['id'], success=0)
# create a location cursor
location_cursor = mysql.connection.cursor()
# Execute the query
location_cursor.execute(
'INSERT INTO LOCATION(location_lat,location_long,location_visited) VALUES(%s,%s,%s)', (
lat, lon, vis
)
)
mysql.connection.commit()
location_cursor.close()
return render_template('home.html', name=session['name'], email=session['email'],
id=session['id'], success=True)
return render_template('home.html', name=session['name'], email=session['email'],
id=session['id'])

@app.route("/logout")
def logout():
# remove the username from the session if it is there
session['id'] = None
session['name'] = None
session['email'] = None
return redirect(url_for('login'))
@app.route("/data")
def data():
if(session['id'] == None):
return redirect(url_for('login'))
location_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = location_cursor.execute(
"SELECT * FROM LOCATION"
)
if(user_result == 0):
return render_template("data.html", responses=0)
else:
res = location_cursor.fetchall()
```

```
print(res)
return render_template("data.html", responses=res)
@app.route("/android_sign_up", methods=["POST"])
def upload():
if(request.method == "POST"):
# get the data from the form
name = request.json['name']
email = request.json['email']
password = request.json['password']
# hash the password
pw_hash = create_bcrypt_hash(password)
# initialize the cursor
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS WHERE user_email=%s", [email]
)
if(user_result > 0):
signup_cursor.close()
return {'status': 'failure'}
else:
# execute the query
signup_cursor.execute(
'INSERT INTO USERS(user_name,user_email,user_password,user_type)
VALUES(%s,%s,%s,%s)', (
name, email, str(pw_hash), "1"
)
)
mysql.connection.commit()
id_result = signup_cursor.execute(
'SELECT user_id FROM USERS WHERE user_email = %s', [email]
)
if(id_result > 0):
id = signup_cursor.fetchone()
return {"id": id[0]}
signup_cursor.close()
return {"status": "failure"}
@app.route("/get_all_users")
def getusers():
signup_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = signup_cursor.execute(
"SELECT * FROM USERS"
PNT2022TMID07010
)
if(user_result > 0):
rv = signup_cursor.fetchall()
row_headers = [x[0] for x in signup_cursor.description]
json_data = []
for result in rv:
json_data.append(dict(zip(row_headers, result)))
return json.dumps(json_data)
```

```
@app.route("/post_user_location_data", methods=["POST"])
def post_user_location():
if(request.method == "POST"):
# get the data from the form
lat = request.json['lat']
lon = request.json['long']
id = request.json['id']
ts = request.json['timestamp']
# initialize the cursor
user_location_cursor = mysql.connection.cursor()
# execute the query
user_location_cursor.execute(
'INSERT INTO USER_LOCATION(location_lat,location_long,user_id,timestamp)
VALUES(%s,%s,%s,%s)', (
lat, lon, id, ts
)
)
mysql.connection.commit()
return {"response": "success"}
@app.route("/location_data")
def location_data():
location_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = location_cursor.execute(
"SELECT * FROM LOCATION"
)
if(user_result != 0):
res = location_cursor.fetchall()
print(res)
row_headers = [x[0] for x in location_cursor.description]
json_data = []
PNT2022TMID07010
for result in res:
json_data.append(dict(zip(row_headers, result)))
return json.dumps(json_data)
else:
return {"response": "failure"}
@app.route("/send_trigger", methods=["POST"])
def send_trigger():
if(request.method == "POST"):
# get the data from the form
email = request.json['email']
location_id = request.json['id']
location_cursor = mysql.connection.cursor()
# check whether user already exists
user_result = location_cursor.execute(
"SELECT location_visited FROM LOCATION WHERE location_id=%s", [
location_id]
)
if(user_result == 0):
return {"response": "failure"}
else:
```

```
res = location_cursor.fetchone()
print(res[0])
visited = res[0]
visited = visited+1
location_cursor.execute(
"UPDATE LOCATION SET location_visited = %s WHERE location_id=%s",
(visited, location_id)
)
mysql.connection.commit()
send_mail(email)
return {"response": "success"}
# main
if __name__ == "__main__":
app.run(host='0.0.0.0', port=5000)
```

DATA.HTML

```
<!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: #699cc5;
}
a {
color: black;
}
</style>
</head>
<body>
<div class="m-4 container">
<h1><u>Location data and Visited People</u></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 responses %}
<tr>
<th scope="row">{{loop.index}}</th>
<td>{{row[1]}}</td>
<td>{{row[2]}}</td>
<td>{{row[3]}}</td>
</tr>
{%- 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.HTML

```
<!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>Document</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: #699cc5;
     }

     a {
        color: black;
     }
  </style> </head>

<body>
```

```
  {% if success == True %}
  <script>
    alert("Location Uploaded Successfully");
  </script>
  {% elif success == 0 %}
  <script>
    alert("Enter Proper Location data");
  </script>
  {% endif %}
  <div class="m-3 float-right">
    <button type="button" class="btn btn-primary"><a href={{url_for("logout")}}>Log
Out</a></button>
  </div>
  <div class="container m-3">
    <h1><u>Declare Containment Zone</u></h1>
  </div>
  <div class="container m-3">
    <h3>welcome: {{name}}</h3>
  </div>
  <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"           cros
sorigin="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>
</body>
</html>
```

GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-2867-1658484923