



CORONAVIRUS TRACKING APP

Surajit Baitalik

Table of Contents

Section 1:Introduction:	2
Section 2: Use-Case Model:	3
Section 3: Detailed Use Cases:	4
Section 4: Sequence Diagrams:	9
Section 4.1:User Registration :	9
Section 4.2:Bluetooth Connect and exchange tokens ,Analyze Token:	10
Section 4.3: Track Location and get density:	12
Section 4.4: Add/Update user data:	13
Section 4.5: Get Location disease Info:	14
Section 4.6: Display Warning and Alarm:	15
Section 4.7: Update CDC:	16
Section 5: Class Model in the analysis phase:	17
Section 6:GUI Interface Model:	18
Section 6.1:Registration Screen:	18
Section 6.2:Home Screen:	20
Section 6.3:Update User Screen:	21
Section 6.4:Disease Info screen:	22
Section 6.5:Menu Bar:	23
Section 7:Static Modelling:	24
Section 7.1:GUI Classes :	24
Section 7.2: Entity Classes:	26
Section 7.3:Interactions between all classes:	27
Section 8:State Chart:	28
Section 8.1: Analyze Token :	28
Section 8.2:Location Density:	29
Section 9: Application Architecture:	30
Section 10:Demo:	31

Section 1:Introduction:

To effectively prevent the wide spread of the Corona virus pandemic (COVID-19), tracking the status of the entire population in terms of the disease contraction and personal contacts is one of the essential but challenging methods. The most feasible solution for tracking the disease is to design and install a tracking application on smartphones. Such a virus tracking app (TrackApp for short) would ideally be installed on everyone's phone.

Assumptions (maybe unrealistic):

- All privacy issues have been taken care of. So, this project needs NOT to consider such issues.
- All smartphones installed with TrackApp would turn on Bluetooth and communicate via Bluetooth. Two essential phone facilities should be used: Bluetooth and Location tracking. Each phone carries a colored token marking the status of its user and initialized as below:
- **Red** – infected or test positive;
- **Green** – normal or test negative;
- **Orange** – having symptoms or had close contact with someone infected.

The token changes from green to orange or red according to the specified condition, from red to orange after the user is tested negative, and from orange to green after the user self-quarantined for 2 weeks.

Two phones within a distance of 5 meters or less would inform each other of their users' disease status by sending each other its token and location.

A warning signal is displayed whenever:

- The phone is within a distance of 5 meters of an infected person (a red token is received);
- The phone receives an orange token, or the location of the phone has a density between M and N persons per KM^2 .

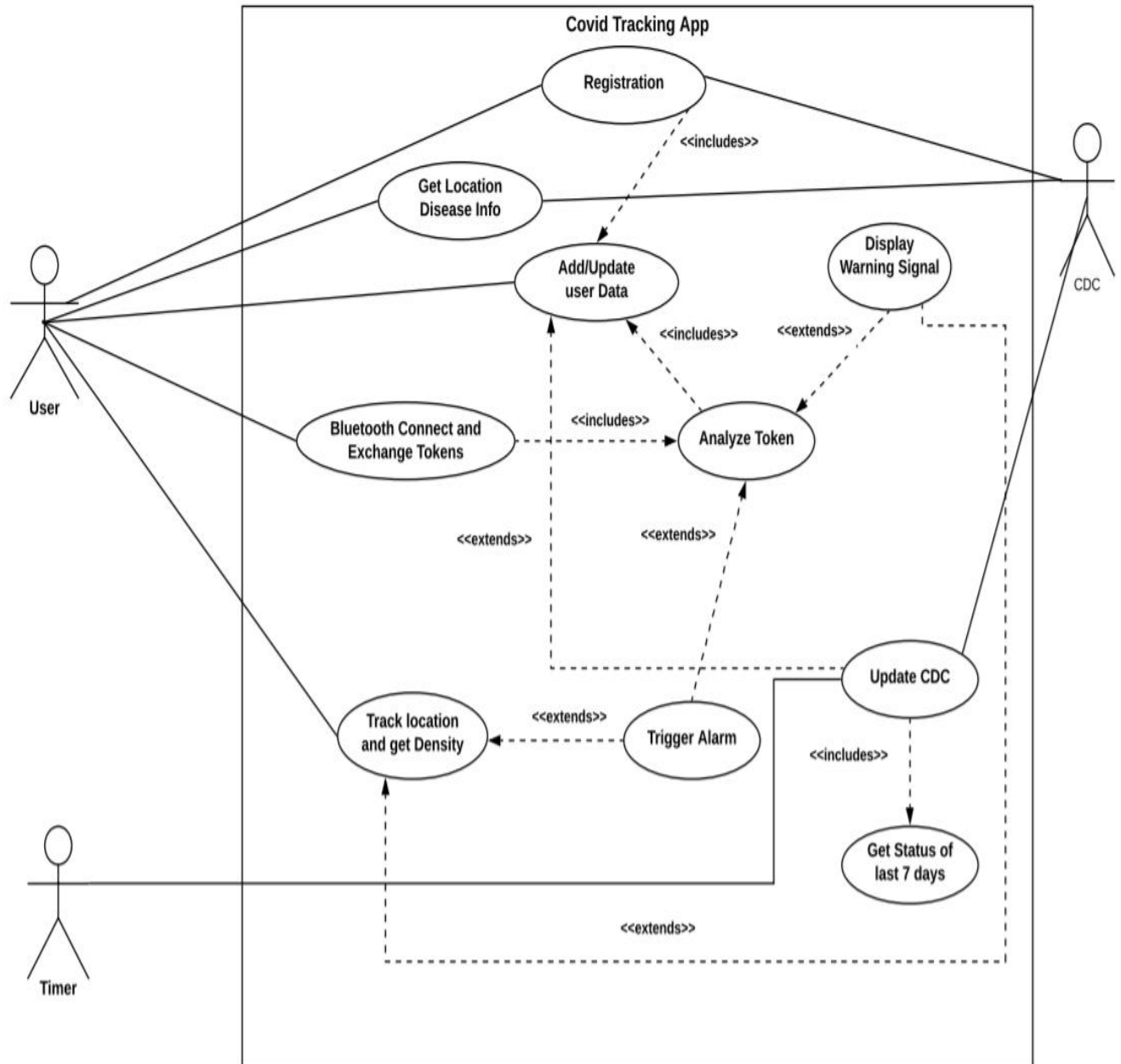
An alarm signal is triggered, and its own token is set to orange whenever:

- The phone is within a distance of 2 meters of an infected person (a red token is received); or the location of the phone has a density higher than N persons per KM^2 .

The entire status including the locations within the past 7 days is sent to CDC periodically every week, and as soon as the token becomes red.

The project must be able to demonstrate the scenarios of all TrackApp's user's 3 statuses, plus the token turning red with transmission of the situation to CDC. TrackApp should allow its user to retrieve relevant disease and location data whenever desired.

Section 2: Use-Case Model:



UML diagram of Use case Model

Section 3: Detailed Use Cases:

Here I have formulated following high-level use cases for this tracking app

1. User Registration
2. Bluetooth Connect and exchange tokens
3. Analyze Token
4. Track Location and get density
5. Add/Update user data
6. Get Location disease Info
7. Display Warning Signal
8. Trigger Alarm
9. Update CDC which includes Get status of last 7 days

Detail description of the use cases:

1. Use case Name: User Registration

Summary : User registers to the CDC via this app and add all required information.

Actors: App user and CDC Server

Precondition: User downloaded the app from app store and installed on his smart phone

Description:

1. open the application.
2. User registration page will appear at first.
3. User enters required information like Name, Contact No., Age, Gender, Test Result
4. Based on the TestResult, the status of the user will change . If positive then status will be red, if negative or the Test Result field is not selected then status will be green, if there is symptom and TestResult is not done then status will be orange.
5. After filling all details click on Register, app will send all information to CDC as well as store it to application database.

Postcondition: Registration is Successful, and user automatically navigated to home screen.

2. Use case Name: Bluetooth Connect and exchange tokens

Summary: Connect with all devices and exchange tokens one by one.

Actors: App user's system or app user

Precondition: System already found a device and listed them in the device list.

Description:

- 1.If system finds a device which is already in a device list and is active . If active and within the range of 5 meters, then it will connect.
- 2.After connecting with a device it will send user's token to that device .
- 3.After sending token it will wait for the token of the receiving device(Synchronous).
- 4.Receiving device will now send a token and sending device will receive it.
- 5.Once received the token , it will send it for analyzing token, which is explained later.
- 6.After analyzing the token the system will update the status if there is a change .
- 7.Now the system will free for exchanging token of another devices.

Postcondition: System will analyze the token.

3. Use case Name: Analyze token

Summary: Once the token is received it needs to analyze.

Actors: System

Precondition: System got token from another device.

Description:

- 1.If the token is green , then won't change your status.
- 2.If the token is orange then show the warning signal on screen .
- 3.If the token is red and distance between two persons is 5 meters ,then show the warning signal.
- 4.If the received token is red and distance between the users is 2m or less, raise an alarm and change the status of the user to orange.
- 5.Every time the user receives the token , it will analyze it and insert to the database if the user is new or it will update the existing user's information.

4. Use case Name: Track Location and get density

Summary: Track the current location and get number of expected population density per KM² every time user changes the location .

Actors: user

Precondition: User changes the location.

Description:

- 1.If the location density of the user is more , suppose it is above the threshold value but below the maximum density value, then show the warning signal ($M < \text{density} < N$).
- 2.If the location density of the of the user is above the maximum value then raise an alarm($\text{density} > N$).
- 3.Location density is retrieved by calling google Heatmap API, which will give us approximate intensity of a surrounding(latitude, longitude).

5. Use case Name: Add/Update user data

Summary: User can update his disease status , quarantine start date ,end date .

Actors: App user

Precondition: User already completed the registration.

Description:

- 1.If the user wants to update his test result as well as quarantine start date and end date.
- 2.Suppose user wants to change his status **symptoms** to **normal** after getting the test result, then he must select Test Result as negative.
- 3.After clicking the update button, user data will be stored in application database as well as on the app's cache.
- 4.If the user puts the test result as positive then it will call the "update CDC " functionality and update the CDC server with all the last 7 days location and close contacts information and update the status of the user as **red** on home screen. User information will be stored on application data base also.
- 5.If the current date is more than quarantine end date and no more update on the test result for las 15 days then the status on the home page will be updated to normal automatically.

6. Use case Name: Get Location disease Info

Summary: User gets the disease info after selecting location which will be provided as a dropdown option for the user.

Actors: App user and CDC Server

Precondition: User already completed the registration.

Description:

1. Click on the disease Info tab.
2. It will fetch the result from CDC server based on location. After navigating to this page, a drop-down spinner will appear on the top of the screen with a set of locations.
3. User selects one location.
4. 3 graphs will appear containing daily deaths, new cases and no of recoveries for last 10 days.

Postcondition: Result will be shown, and user can go back to home screen by clicking on home button on the menu.

7. Use case Name: Display Warning Signal

Summary : The warning icon with particular warning message will be shown after meeting certain criteria.

Actors: System

Precondition: If the user is in high density location or received either orange or red token.

Description:

1. If the location has a density in between M and N users per square KM, then user should be notified with warning message mentioning situation.
2. If a red token is received and the user is 5 meters away from the infected person then warning message will be shown.
3. User receives an orange token, then warning message will be shown.

8. Use case Name: Trigger alarm

Summary : The warning alarm will be triggered if certain conditions are met.

Actors: System

Precondition: If the user is in high density location or received a red token.

Description:

- 1.If the location is dense and density is above a threshold (density >N persons/sq. KM) value then trigger the alarm.
2. If a red token is received and the user is 2 meters away from the infected person, then trigger the alarm.
- 3.UI controller will receive Alarm flag as True from location tracker service and Bluetooth service.

9. Use case Name: Update CDC

Summary :update the location and status of the user for last 7 days to CDC server.

Actors: App user and CDC Server

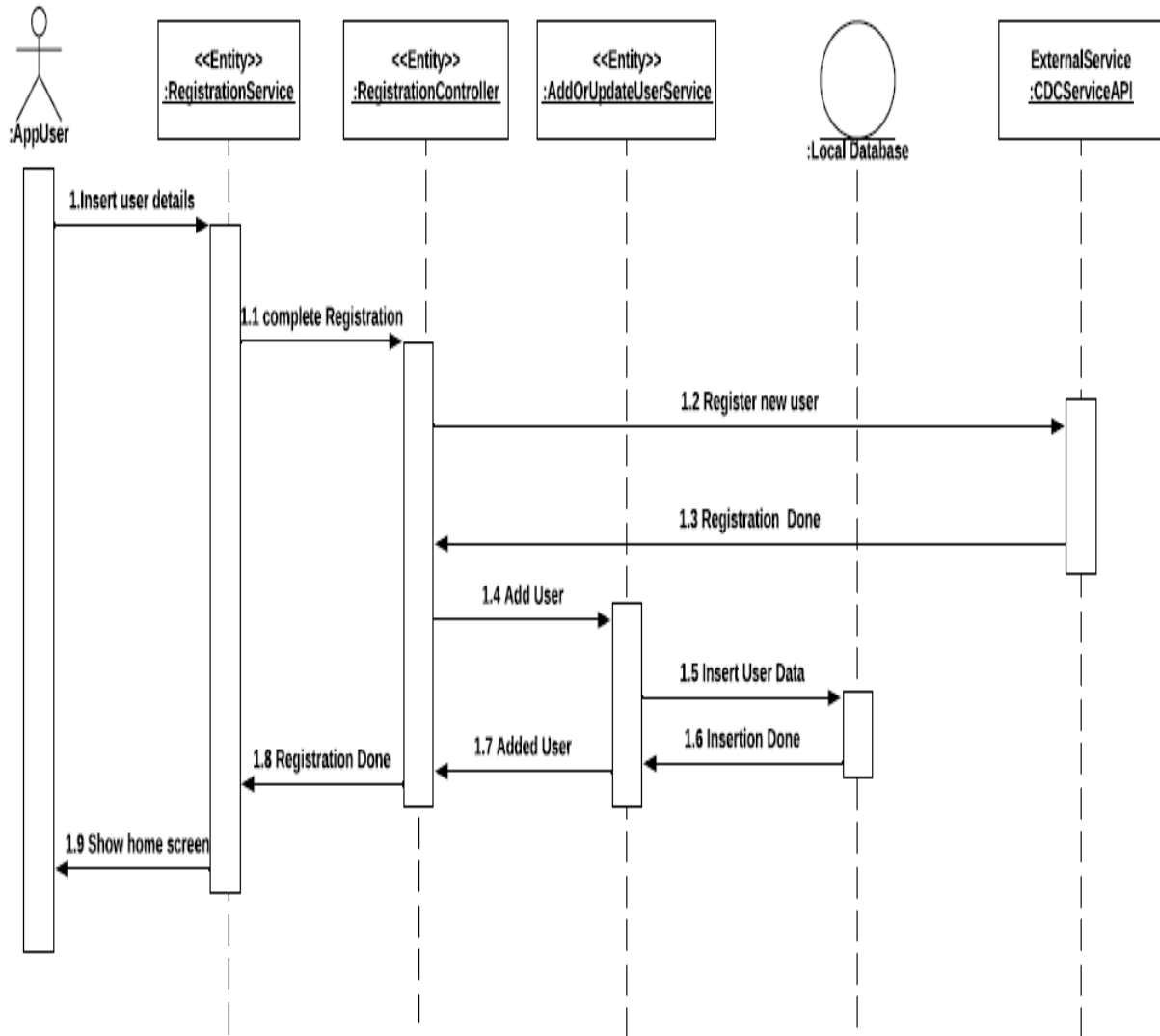
Precondition: No status has been sent for last 7 days or user is tested positive .

Description:

- 1.If the user is tested positive and updated the same status in the app , then this functionality will be called.
- 2.Once it is called , it will get the location and status information of the user for last 7 days and send it to the CDC server via webservice call .
- 3.If the no status has been sent to CDC for last 7 days, then this functionality will be called by timer. The time runs independently on the app.

Section 4: Sequence Diagrams:

Section 4.1: User Registration :



Description:

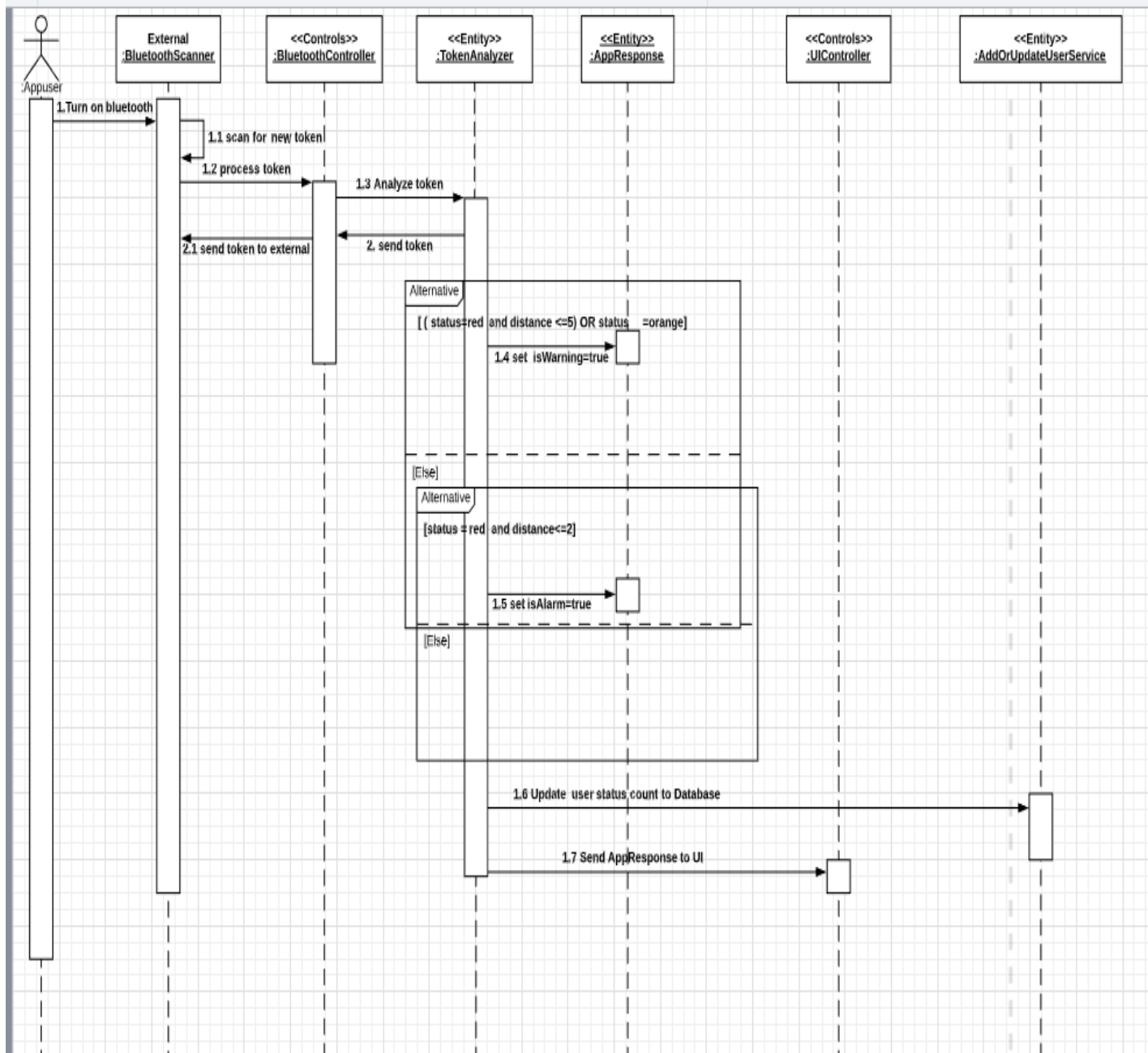
1.Insert user Details will fill the registration form

1.1 Complete Registration function is called

Messages are the same meaning of the functionalities.

Once Registration is done 1.9 Show Screen will navigate the control to home screen.

Section 4.2:Bluetooth Connect and exchange tokens ,Analyze Token:



Description:

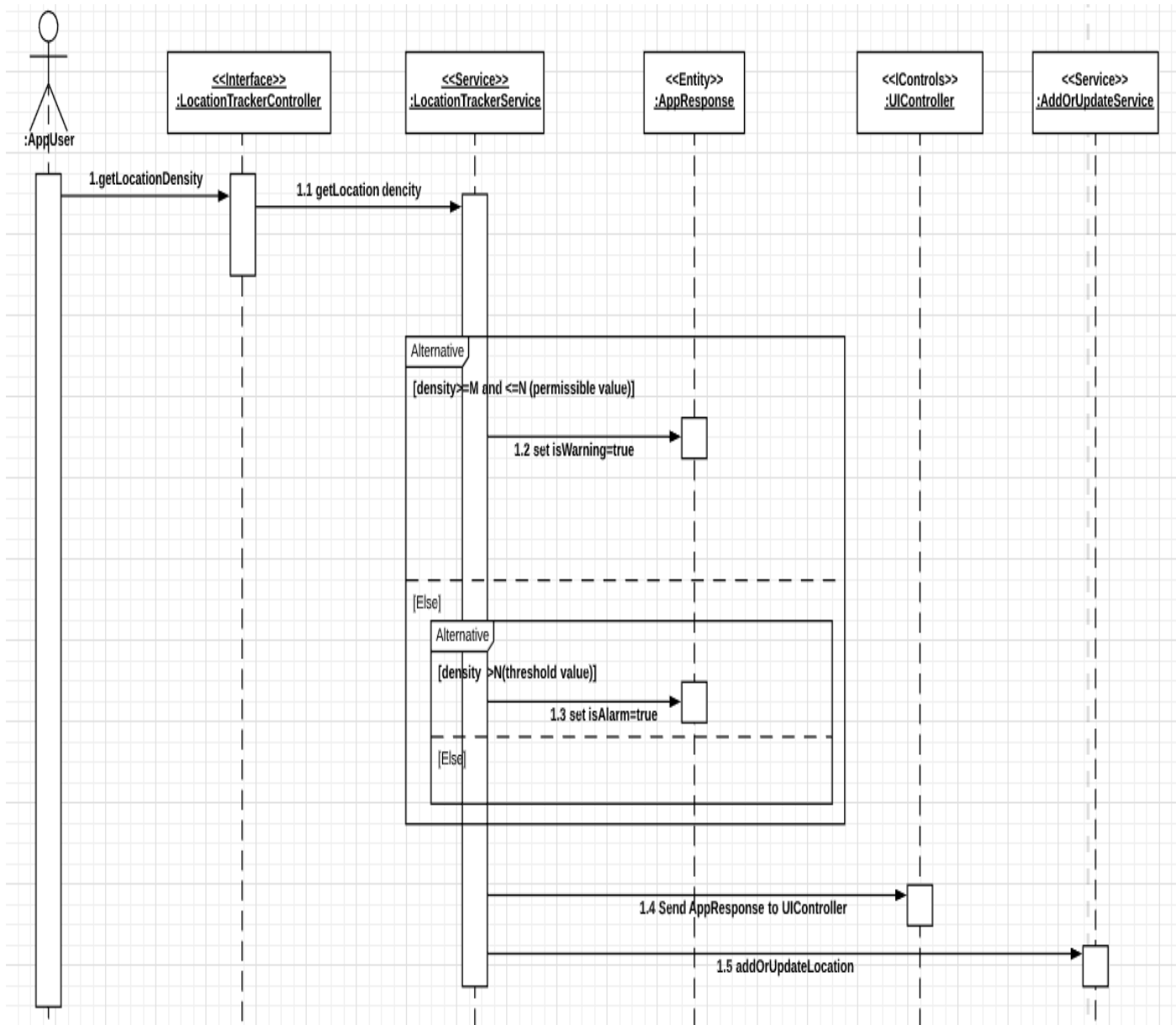
1. Turn the Bluetooth on will on the blue tooth service.

1.1 It will scan for new tokens and then the subsequent tasks will be performed as mentioned in the diagram.

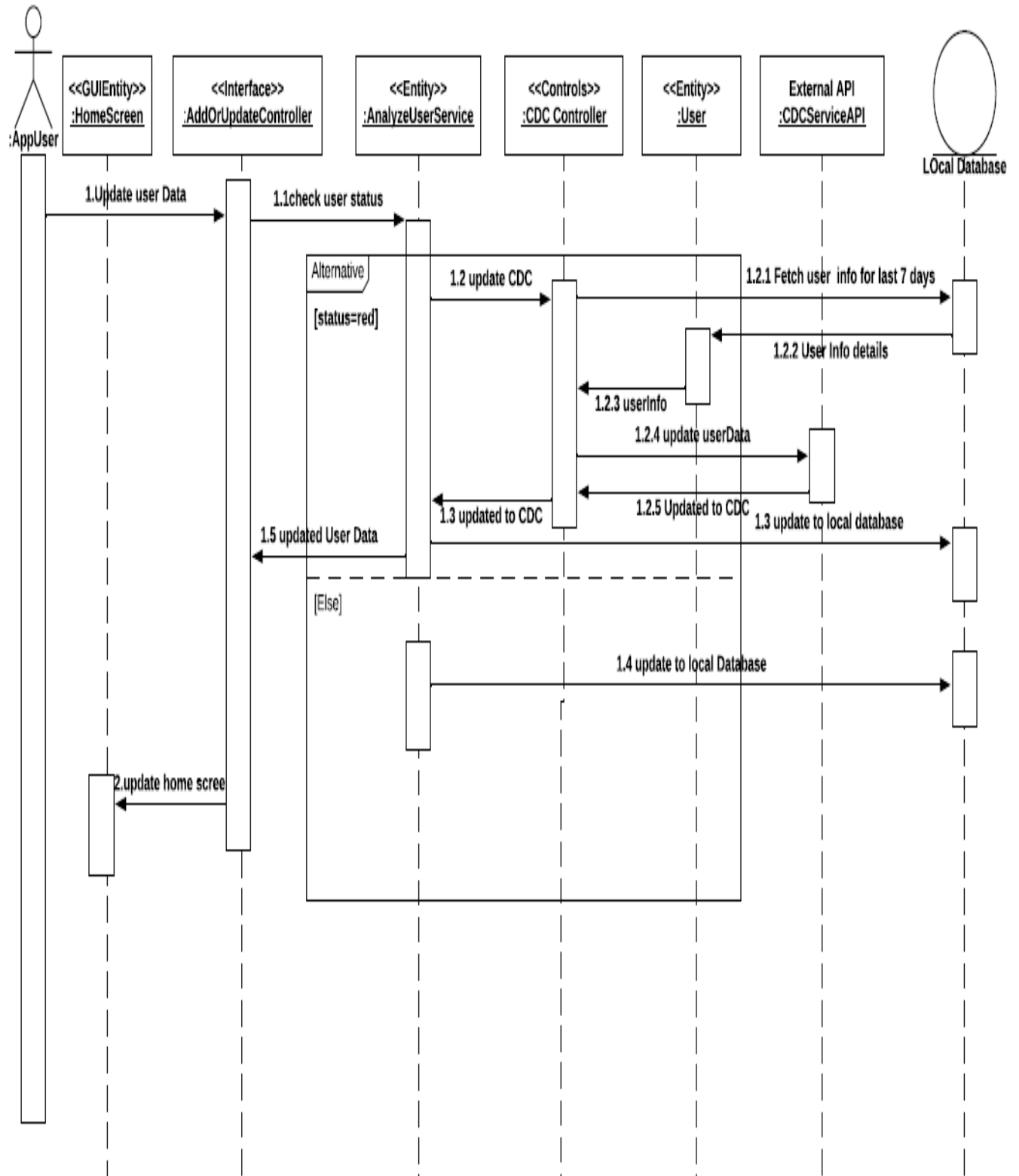
Messages are the same meaning of the functionalities.

Once analyze of token is done , it will send the response to the Home screen through UI Controller.

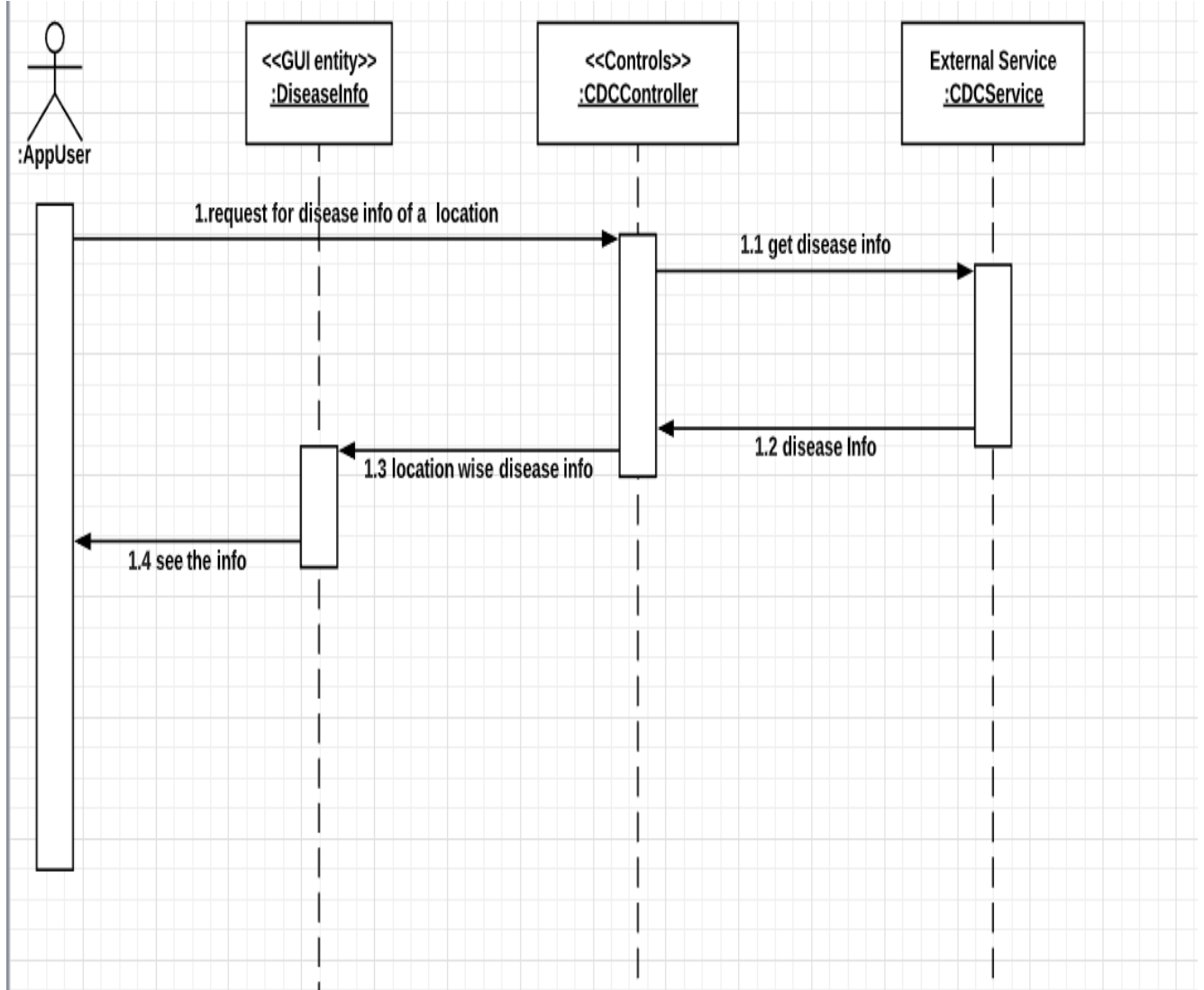
Section 4.3: Track Location and get density:



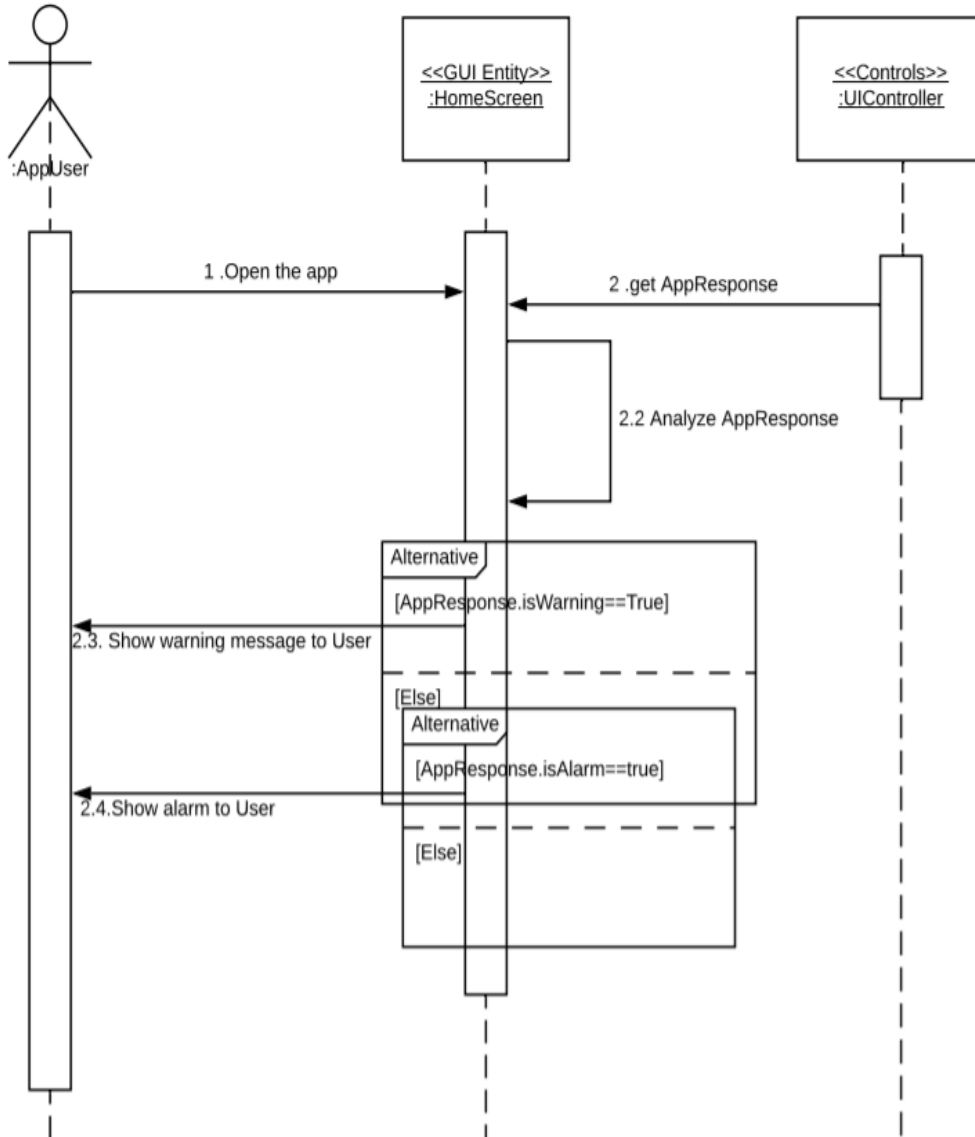
Section 4.4: Add/Update user data:



Section 4.5: Get Location disease Info:



Section 4.6: Display Warning and Alarm:

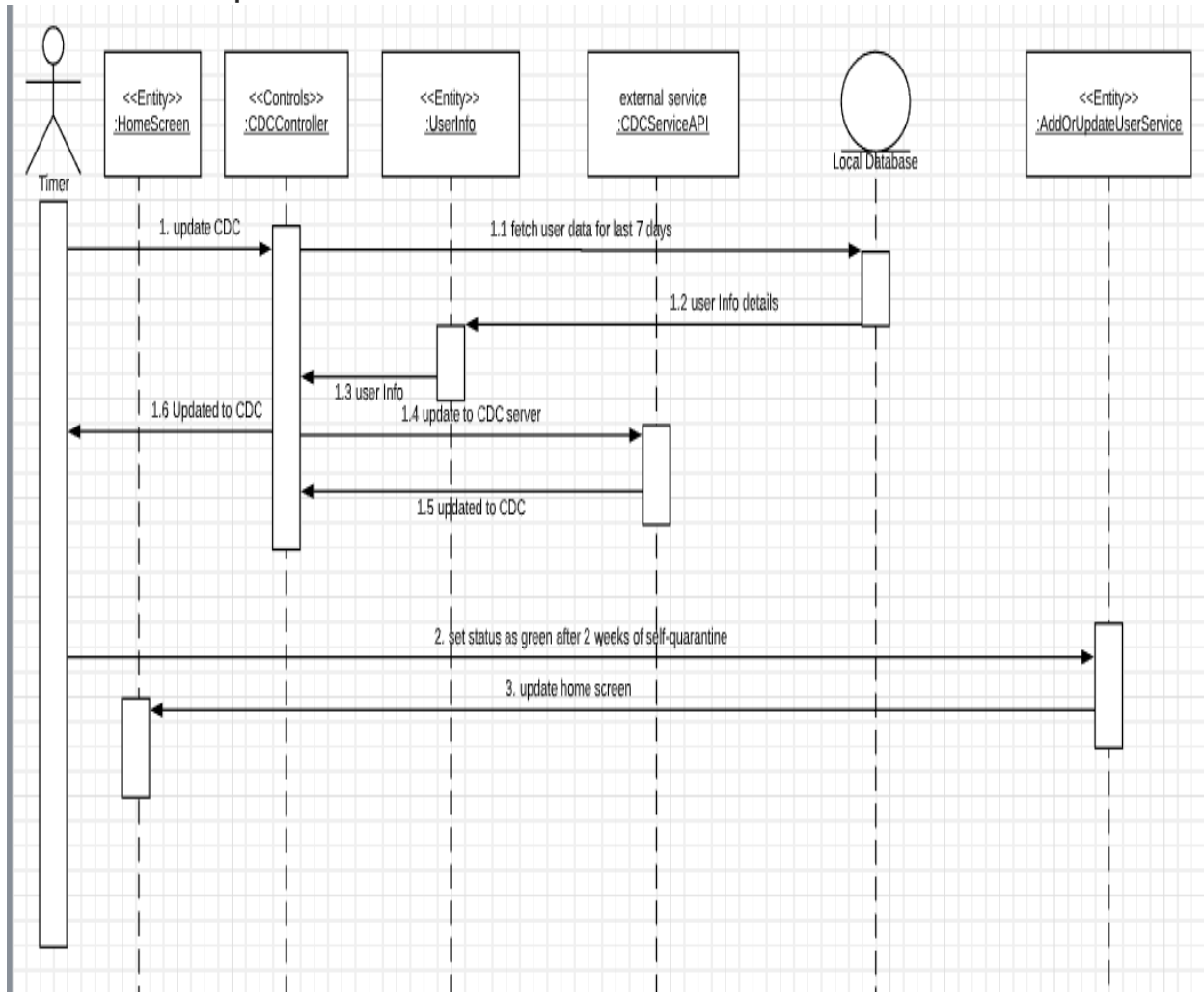


Description:

1. Based on the flag value of `isAlarm` and `isWarning` it will trigger warning as well as alarm service on the device.

Warning messages as well as alarm will be shown on the home screen.

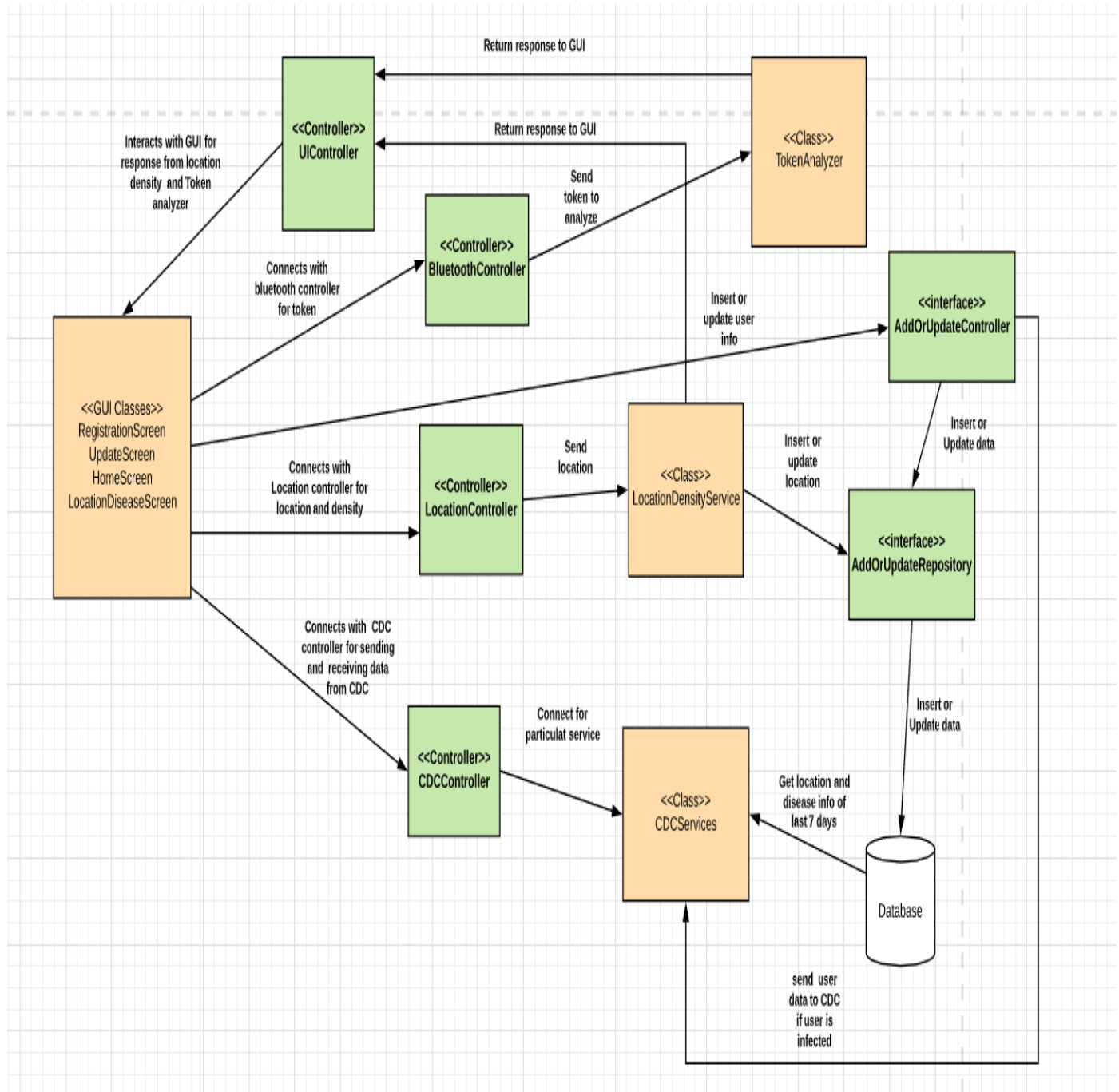
Section 4.7: Update CDC:



Description:

1. Sequencing of the messages are numbered in the diagram, so it will work as per the diagram.

Section 5: Class Model in the analysis phase:



Description:

In the above class diagram it is shown the actual design and architecture of the application.

Here GUI Layer classes are connected with the Bluetooth Controller, Location Controller, CDC Controller, UI Controller which are mainly interfaces to connect with back end.

It has two layer architecture, GUI Layer and Back end layer. Interfaces has implementations in the back end. Bluetooth Controller implement Token analyzer service . Involvement of classes are explained in the static modelling section.

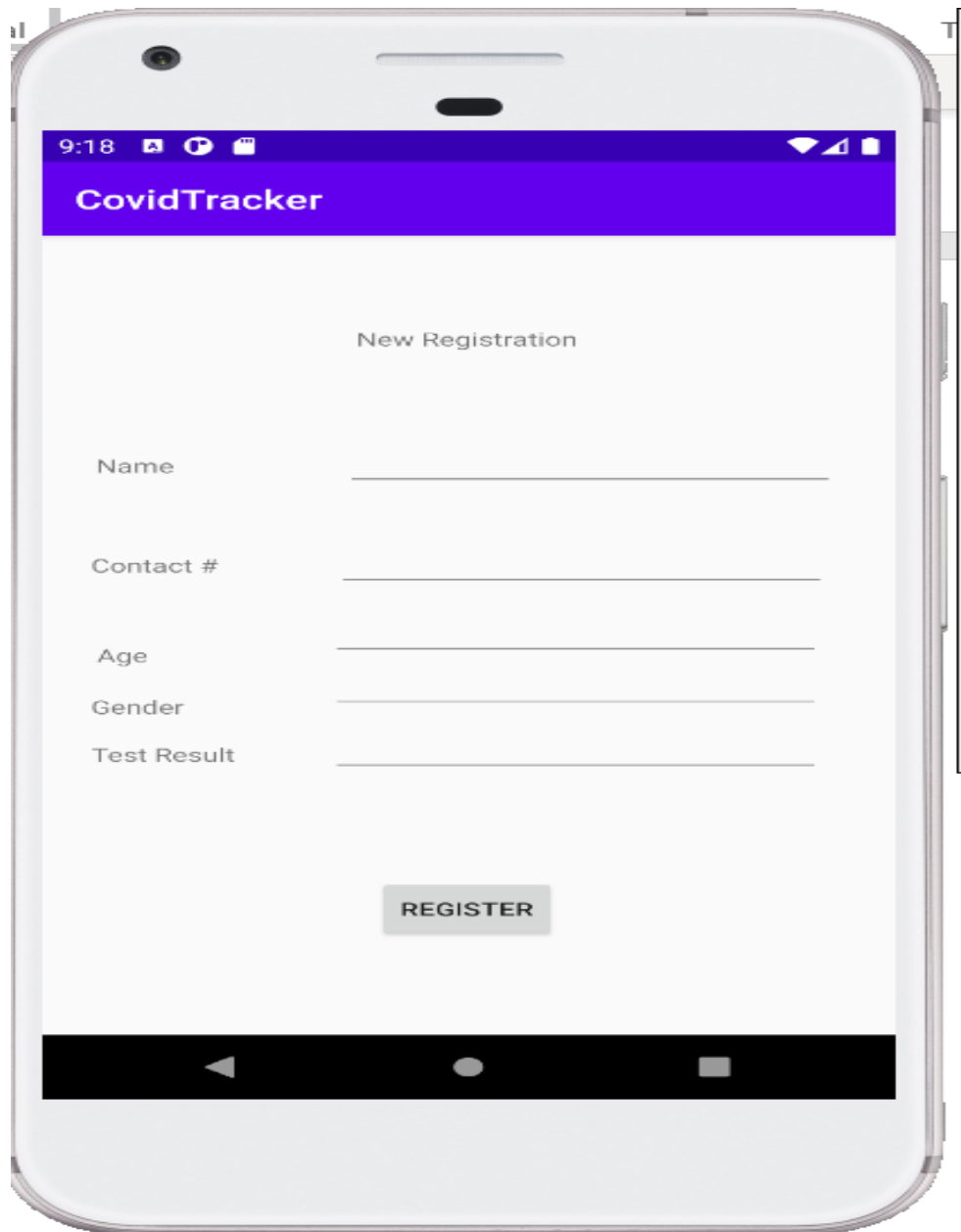
There is only one database wrapper which is AddOrUpdateRepository to add and update info to the data base.

CDCController interface hosted all the functions to send info to CDC as well get disease information from them. It has implementation on CDCService class.

Similar architecture for LocationController also.

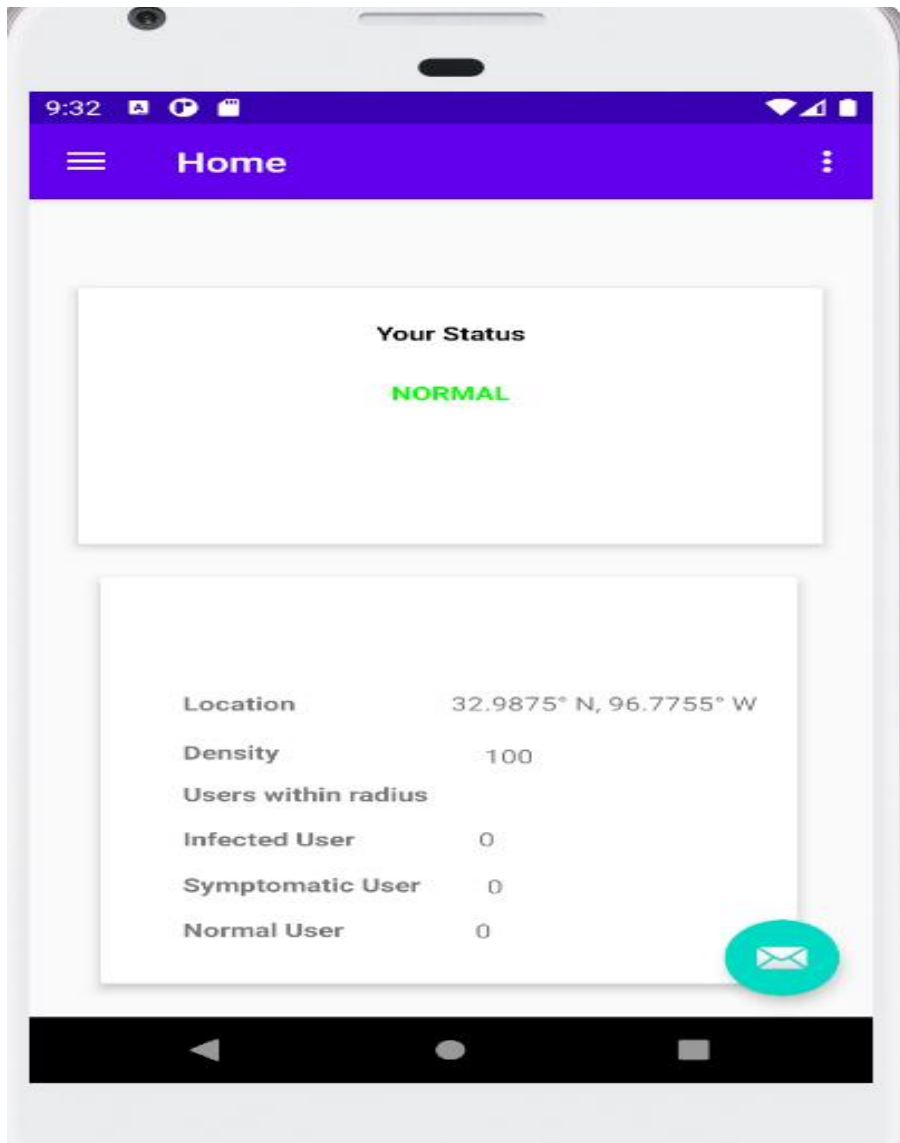
Section 6:GUI Interface Model:

Section 6.1:Registration Screen:



Description: This is the start up page , user must enter their details to register to the app . after clicking **REGISTER** button , user will be registered with CDC and navigated to the Home Screen.

Section 6.2:Home Screen:



Description:

In the home screen it will show your current disease status in the above section.

In the lower part it will show current location, current location density.

Infected User : This will show all the infected user count to whom user was closely contacted. Initially it shows 0, as no infected user has been in close contact.

Symptomatic User: This will show all the symptomatic user count to whom user was closely contacted. Initially it shows 0, as no symptomatic user has been in close contact.

Normal User: This will show all the normal user count (who has no symptoms or covid result is negative) to whom user was closely contacted. Initially it shows 0, as no normal user has been in close contact.

Section 6.3:Update User Screen:

9:45

Update User

Update User

Test Result positive

Quarantine Start Date

Quarantine End Date

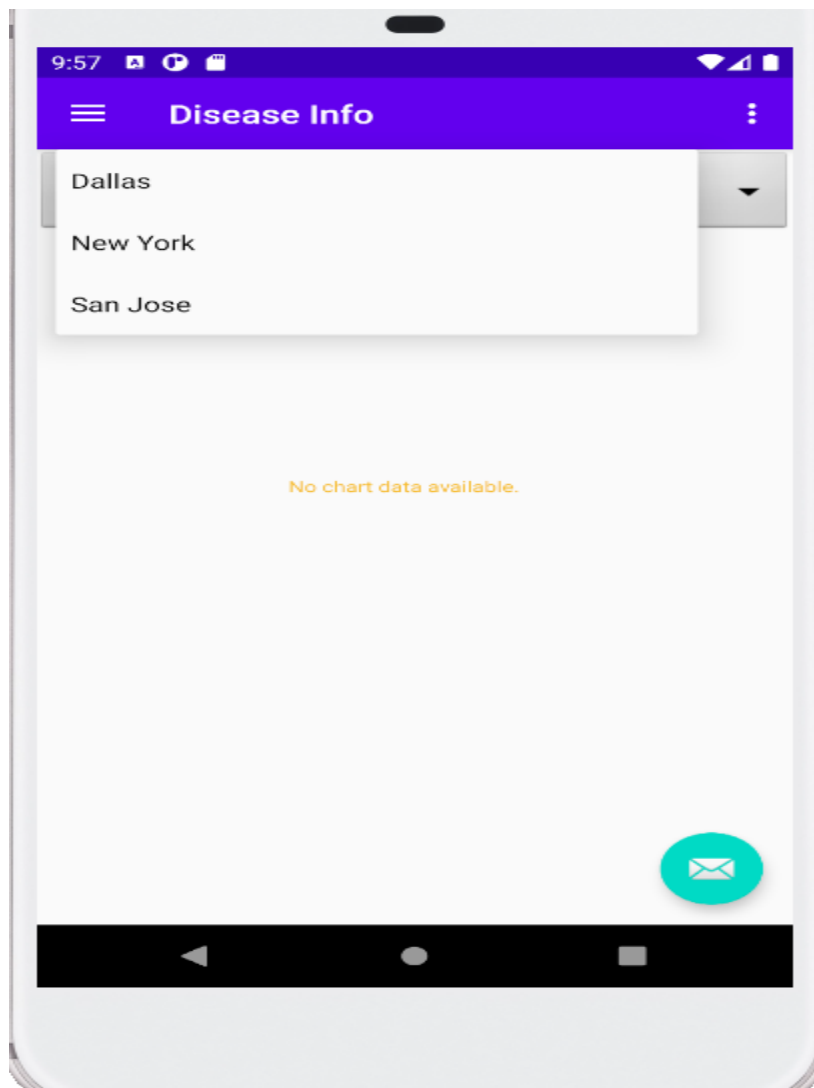
UPDATE

Envelope icon

Description:

User can update their information after getting test result as well as they can update their quarantine start date and end date. If the user selects test result as positive, then the status will be shown as **Infected** in the home page. If they set quarantine start date and end date, then same will be shown below the status in the home screen.

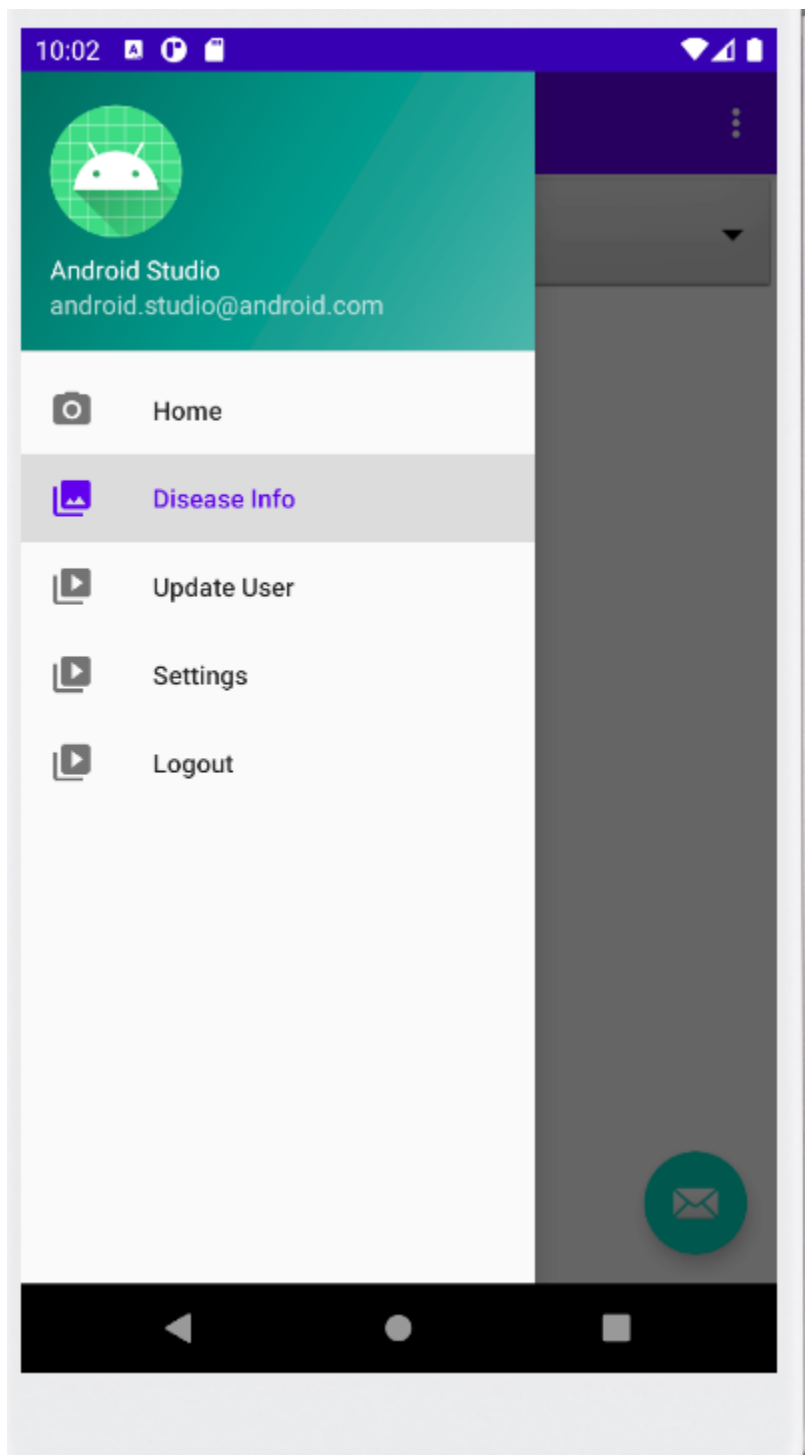
Section 6.4:Disease Info screen:



Description:

If user select a location then it will show Covid information(No. of New Cases, No. of deaths, No. Of recoveries) for past 10 days.

Section 6.5: Menu Bar:

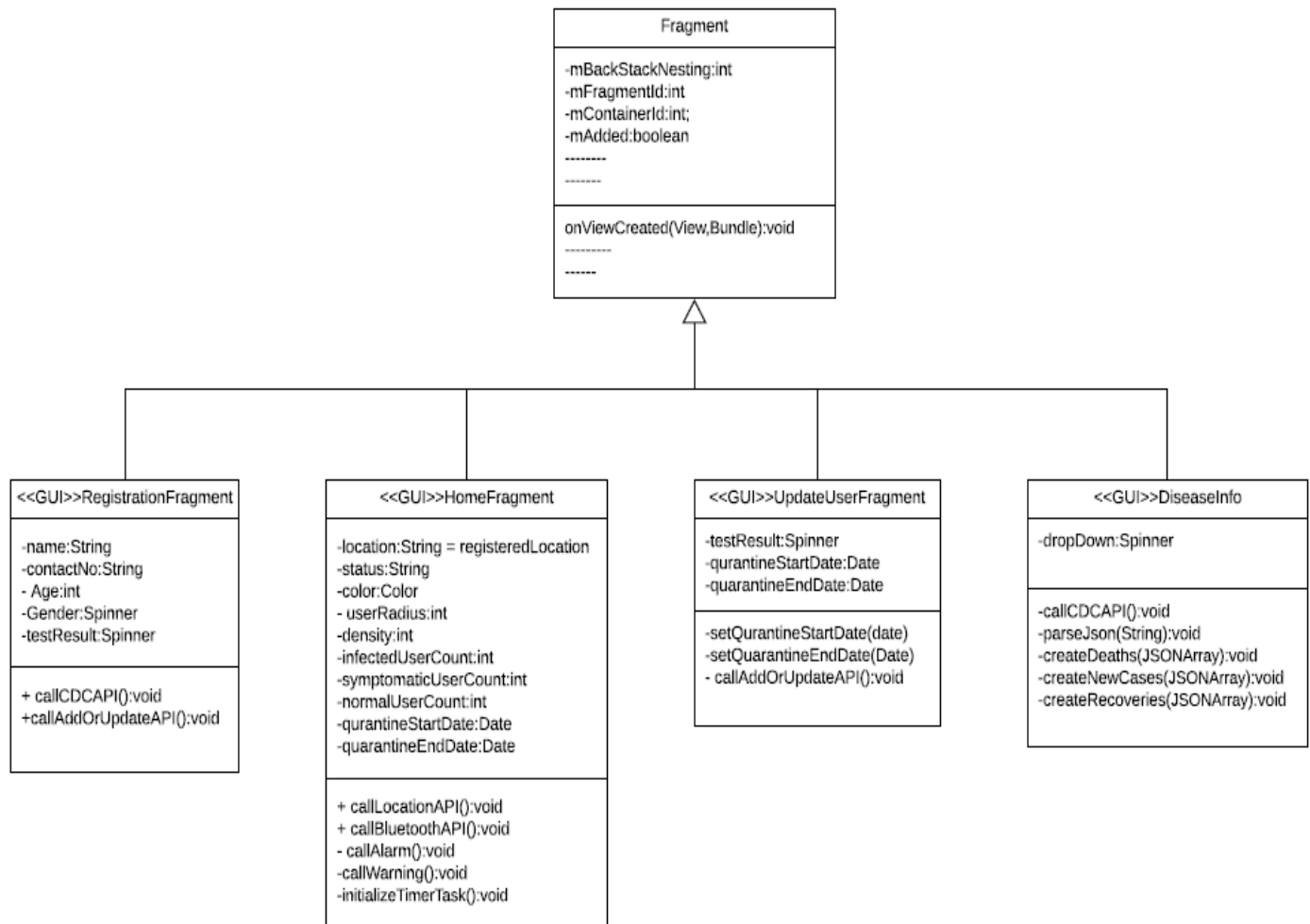


Description:

This will show all options that are provided in this app.

Section 7:Static Modelling:

Section 7.1:GUI Classes :



Description: As right now I have 4 working screens on my android app , so there are 4 Fragment classes for Registration, Home, Update and Disease Info functionalities all these classes extended the functionalities of the Fragment class of Android.

1.**RegistrationFragment**: It contains all the fields of the New Registration screen .When a user first install this app, this page is the start up page to login to the device. It will register the user to the CDC.

2.**HomeFragment**:It includes all the fields and functionalities of Home screen. This class has more responsibilities like calling the location tracker service and Bluetooth service on a fixed interval of time. Based on the status of the incoming token it will change the status also it fetch the location from location service and show it. Based on the severity it will call warning and alarm service.

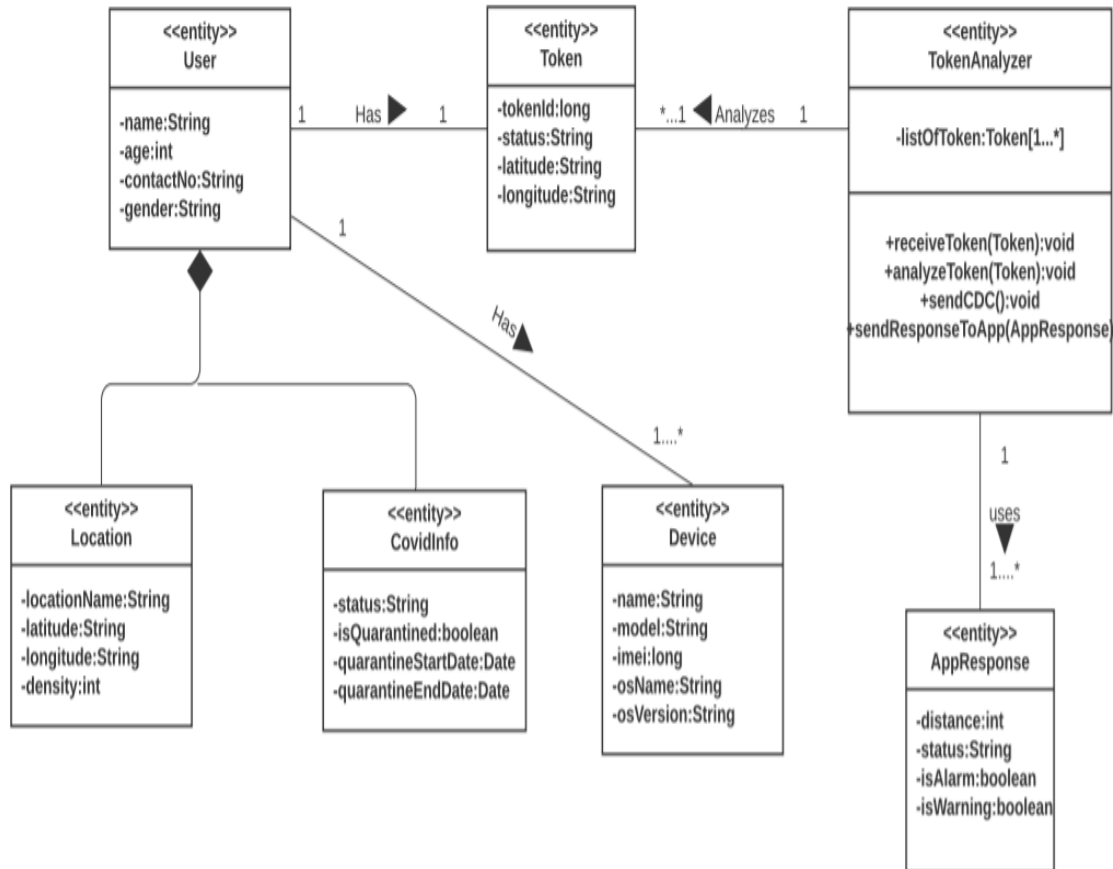
3.**UpdateUserFragment**:It includes all the fields and functionalities of Update User screen like update the status of the user and start quarantine.

4.**DiseaseInfoFragment**:It includes all the fields and functions of disease Info screen. It basically shows all Covid data for last 10 days.

Section 7.2: Entity Classes:

Entity classes

S Baitalik | July 28, 2020



Description:

There are 7 entity classes in my application:

User class is composed of Location and covidInfo. The relationship is composition .

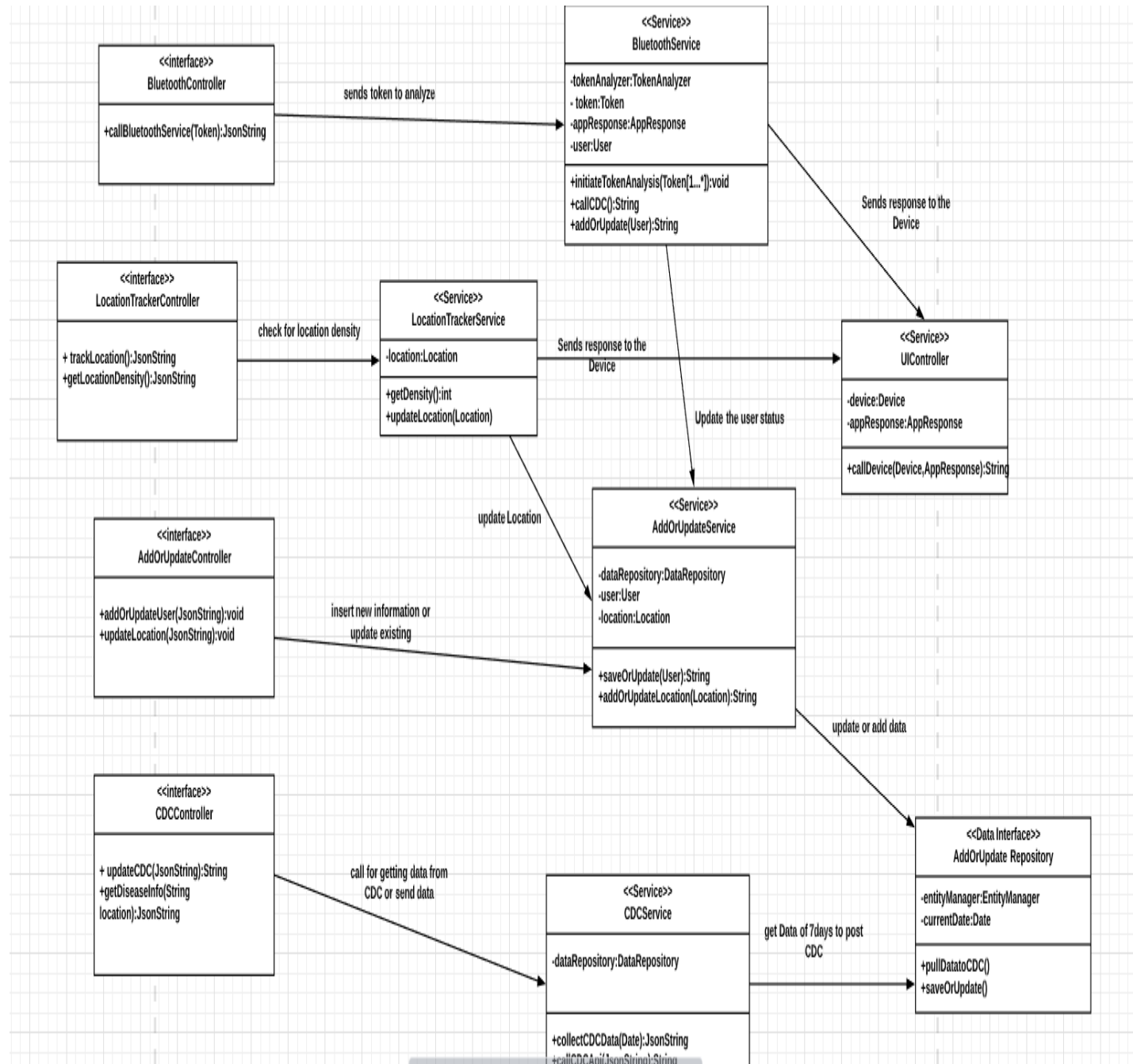
Token class contains few fields which we will communicate with external users .It has one to one relationship with the User as one user can only have one token at a time.

Device class represents the device information, which can be android ,IOS or any other device. One User can log in in multiple devices , so it has many to one relationship with device class.

TokenAnalyzer class analyzes list of Tokens , so it has one to many relationships with the Token class. All methods are mentioned here. It uses AppResponse class to send the response to the device.

AppResponse class has many to one relationship with Token Analyzer class as multiple instances of the AppResponse will be used by one TokenAnalyzer.

Section 7.3:Interactions between all classes:



Description:

The classes which includes interfaces, Services, database connectivity all are mentioned along with the interaction messages

Before explaining the class interactions I'm going to show you my application architecture diagram.

All the controllers are the interfaces with the devices.

Bluetooth controller invoke the Bluetooth service which consists of Analyze token. After analyzing it sends the response to the UI controller which shows the updated data to the device.

Similar for location tracker and addOrUpdate service.

All the message communications between all classes are mentioned in the diagram. Please zoom in the diagram if not visible.

Here is the link:

https://app.lucidchart.com/documents/view/5147bdab-e845-48a7-833a-1322782e5fa7/0_0

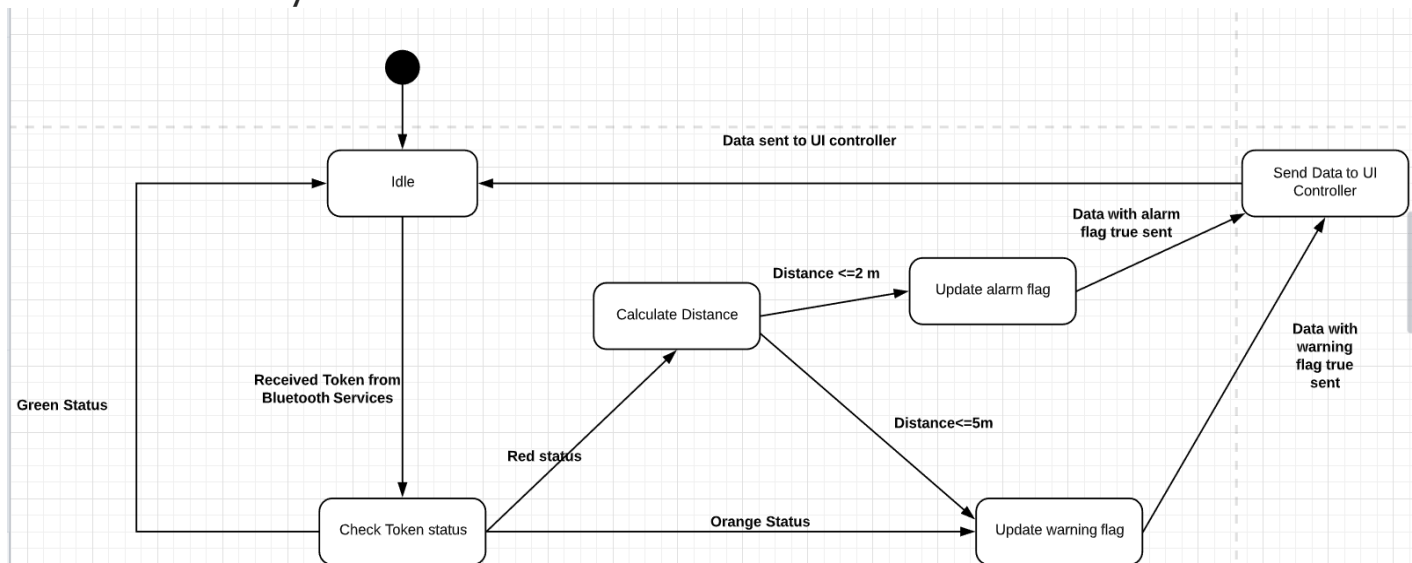
Section 8:State Chart:

There are 2 state dependent functionalities in my design:

1. Analyze Token
2. Location Density

Below I have discussed state charts of the following:

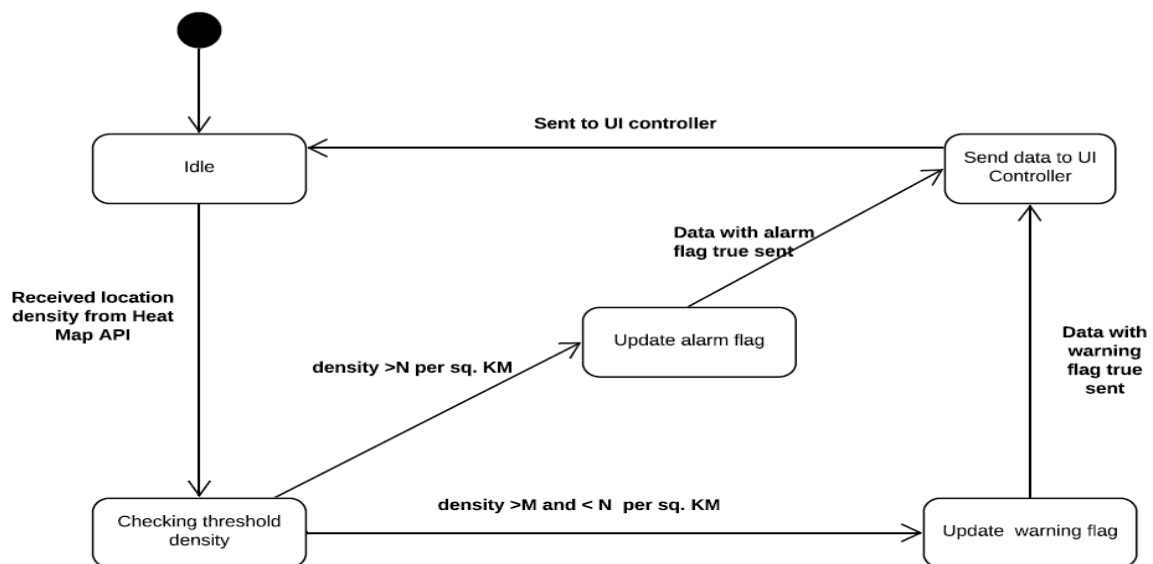
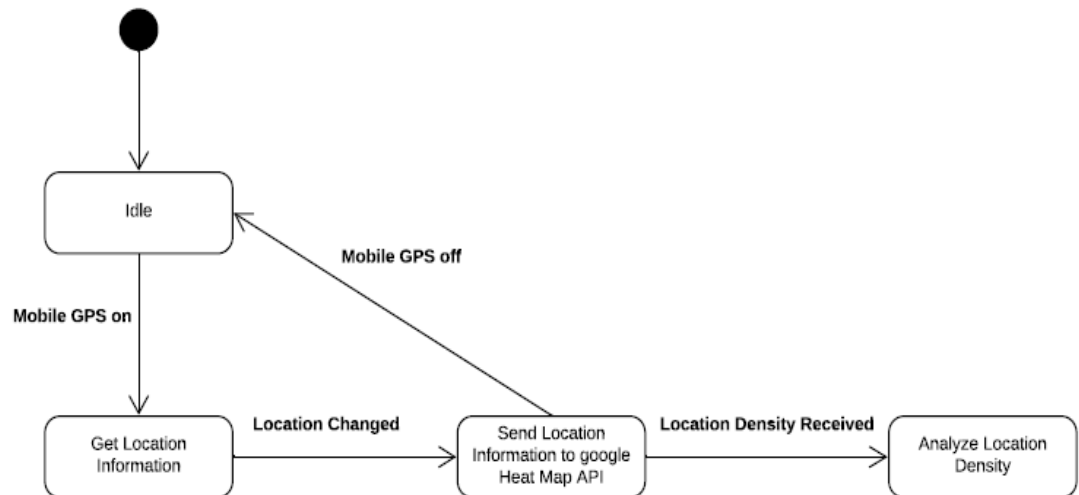
Section 8.1: Analyze Token :



Description:

Token analyzer service will start once it receives token from Bluetooth Services. If the token is Green, then it won't do the analyze. But if it is either red or orange then it will process further as mentioned above.

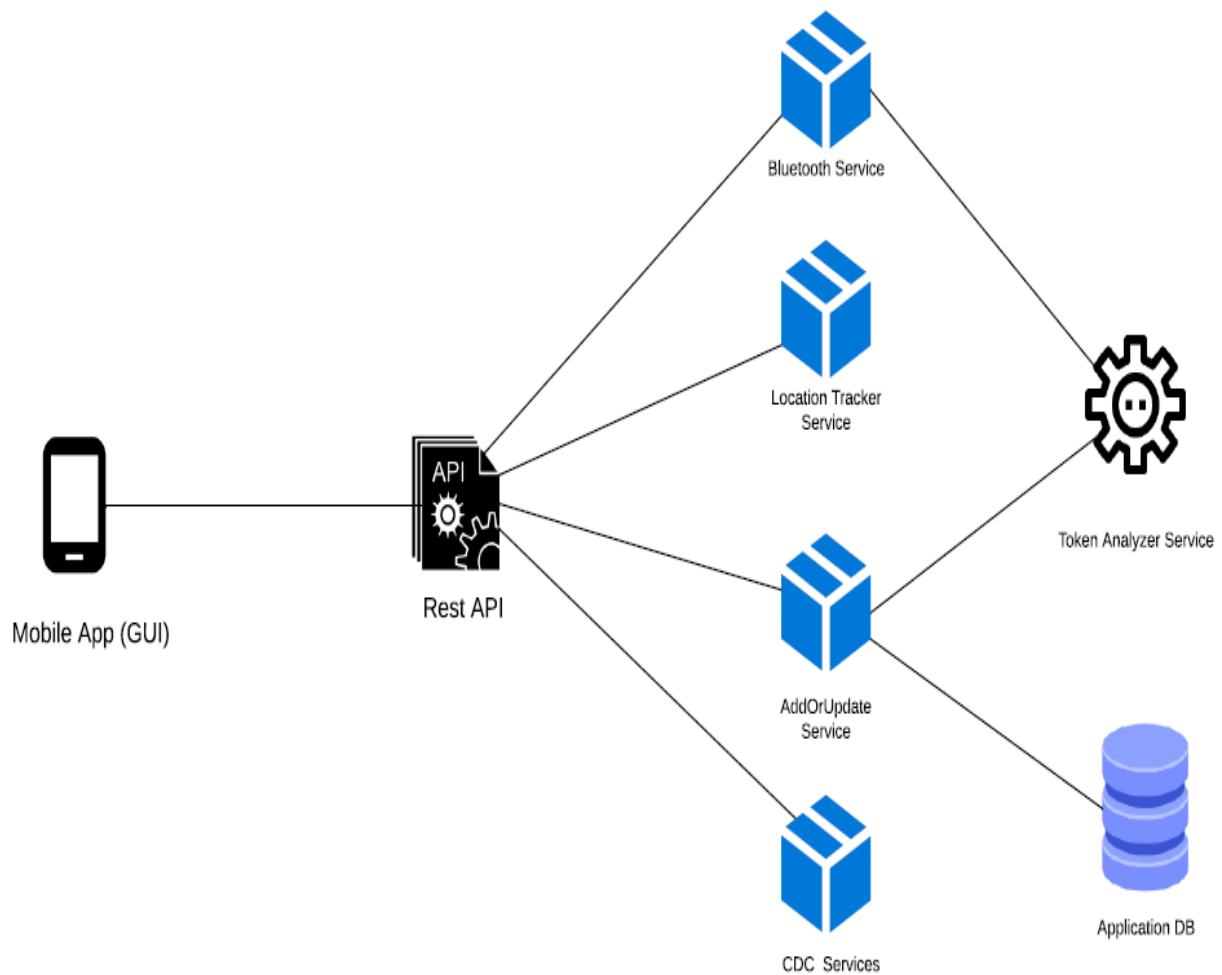
Section 8.2:Location Density:



Description:

Location Density service will start once the GPS is on . If the location is changed then we will send the location information to **Google Heatmap Api** to get the approximate density based on the heat scan result. Once the density is received then we will analyze as mentioned with clear messages in the above.

Section 9: Application Architecture:



Description:

This application is a mobile app not specific to android. We have GUI layer and backend . All are loosely coupled.

Rest API is used to consume the services provided by the application which is independent of programming languages and it communicates through Json .

All the location Tracker and Bluetooth services are invoked by backend on the devices.

User specific information's are stored in the application database as well as device cache.

Response Message: Token analyzer:

```
{
  "token_id": 1,
  "userLatitude": "32.9875° N",
  "userLongitude": "96.7756° W",
  "userdiseaseStatus": "normal",
  "distance": 5,
  "warning": false,
  "alarm": false
}
```

Message format from location tracker Service:

```
{
  "density": "200",
  "location": "32.9875° N, 96.7755° W"
}
```

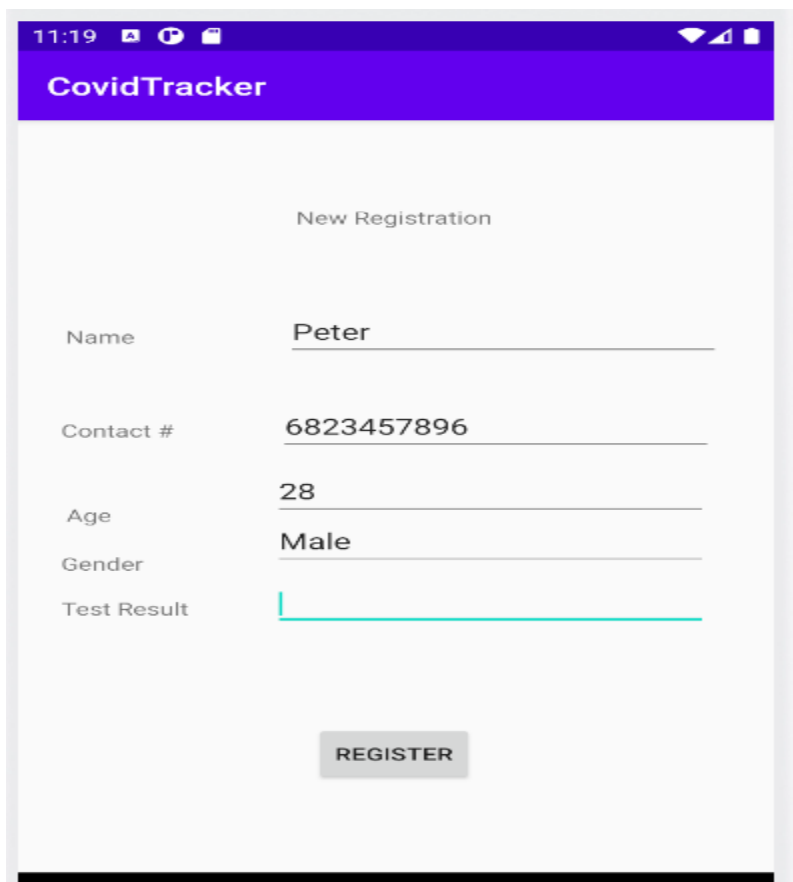
Section 10: Demo:

For looking into my presentation please check the demo recording of 23rd July 2020, I was 2nd last in the sequence:

Here is the URL:

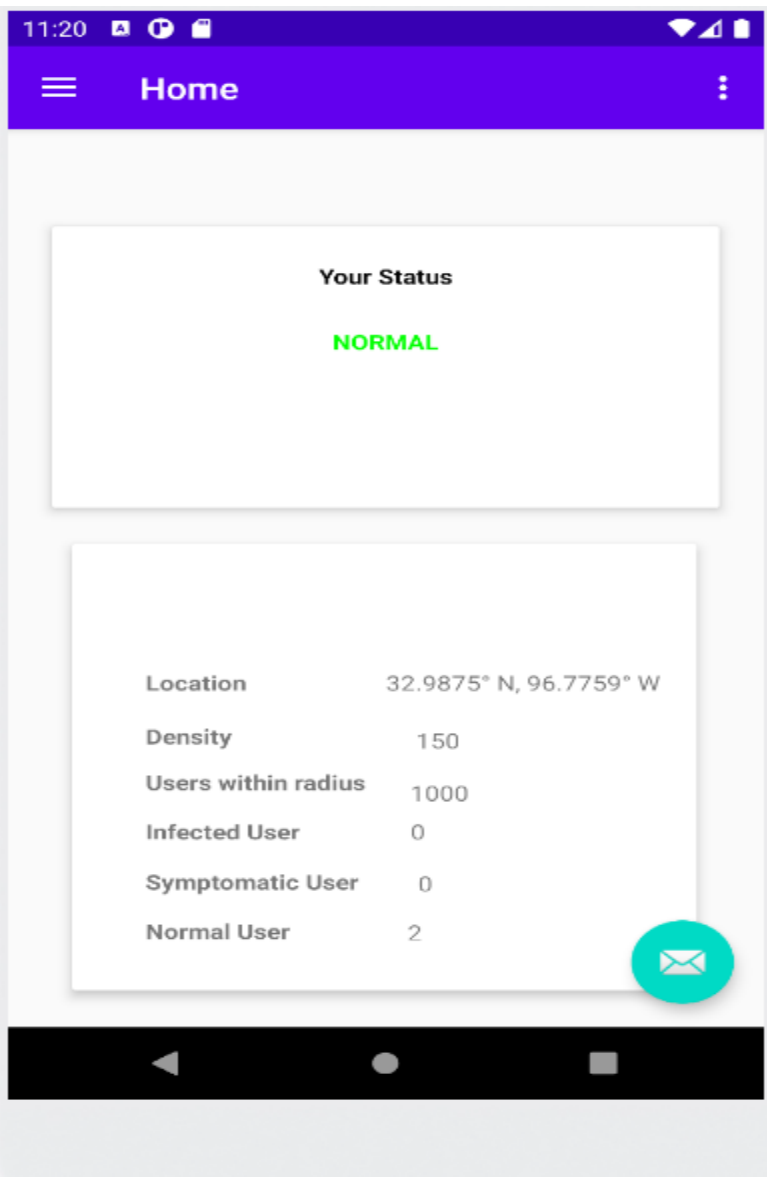
<https://us.bbcollab.com/collab/ui/session/playback>

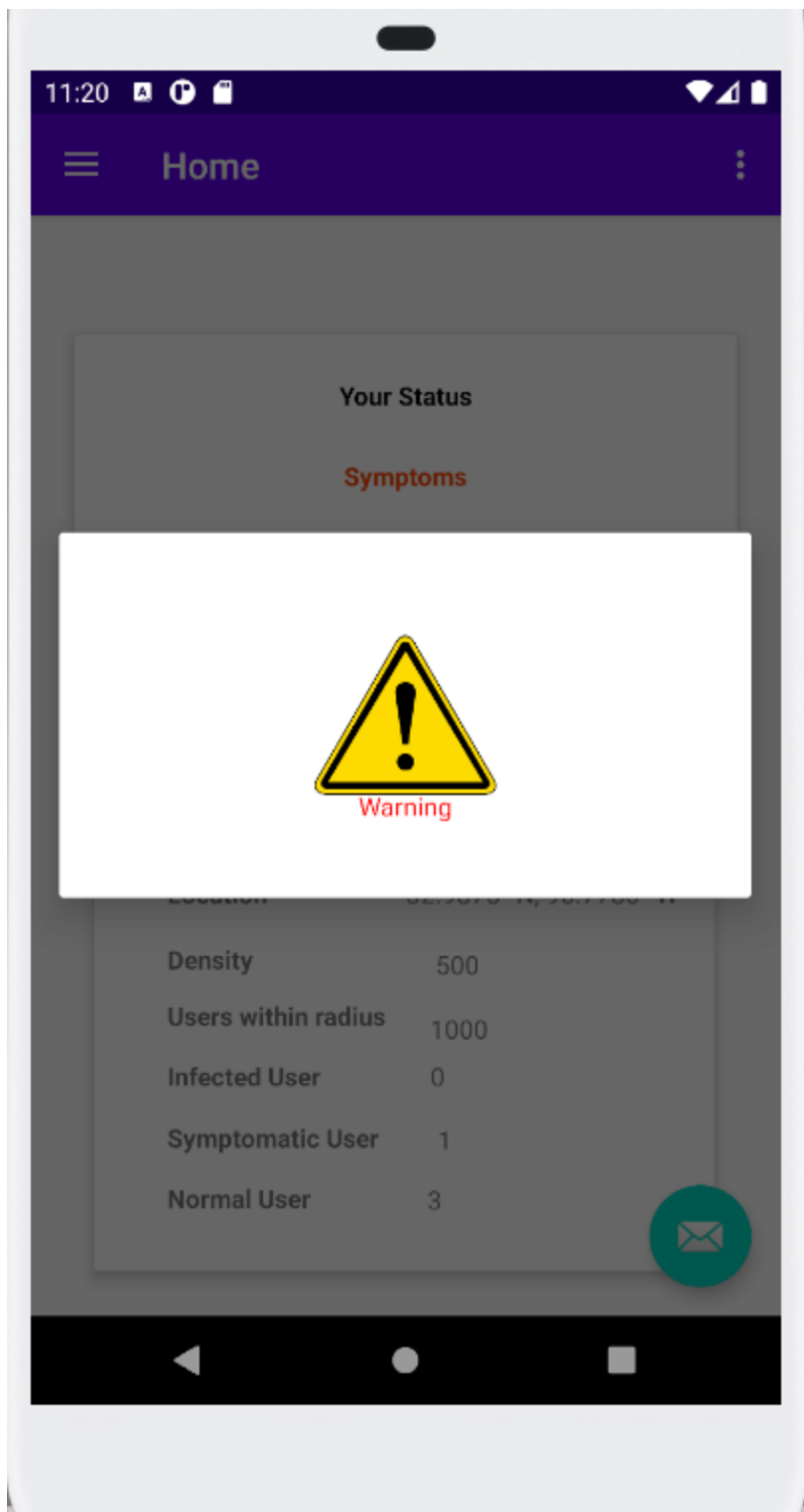
I'm sharing screen shots of my demo:



The screenshot displays the 'CovidTracker' app interface on a mobile device. At the top, a purple header bar contains the app's name. Below this, the title 'New Registration' is centered. The form consists of five labeled input fields: 'Name' (filled with 'Peter'), 'Contact #' (filled with '6823457896'), 'Age' (filled with '28'), 'Gender' (filled with 'Male'), and 'Test Result' (empty). A grey 'REGISTER' button is positioned at the bottom center of the form area. The device's status bar at the very top shows the time as 11:19 and various system icons.

CovidTracker	
New Registration	
Name	Peter
Contact #	6823457896
Age	28
Gender	Male
Test Result	
REGISTER	





11:21



Home



Your Status

Symptoms



Danger

Density	650
Users within radius	1000
Infected User	2
Symptomatic User	3
Normal User	6



11:23



Update User



Update User

Test Result

negative



Quarantine Start Date

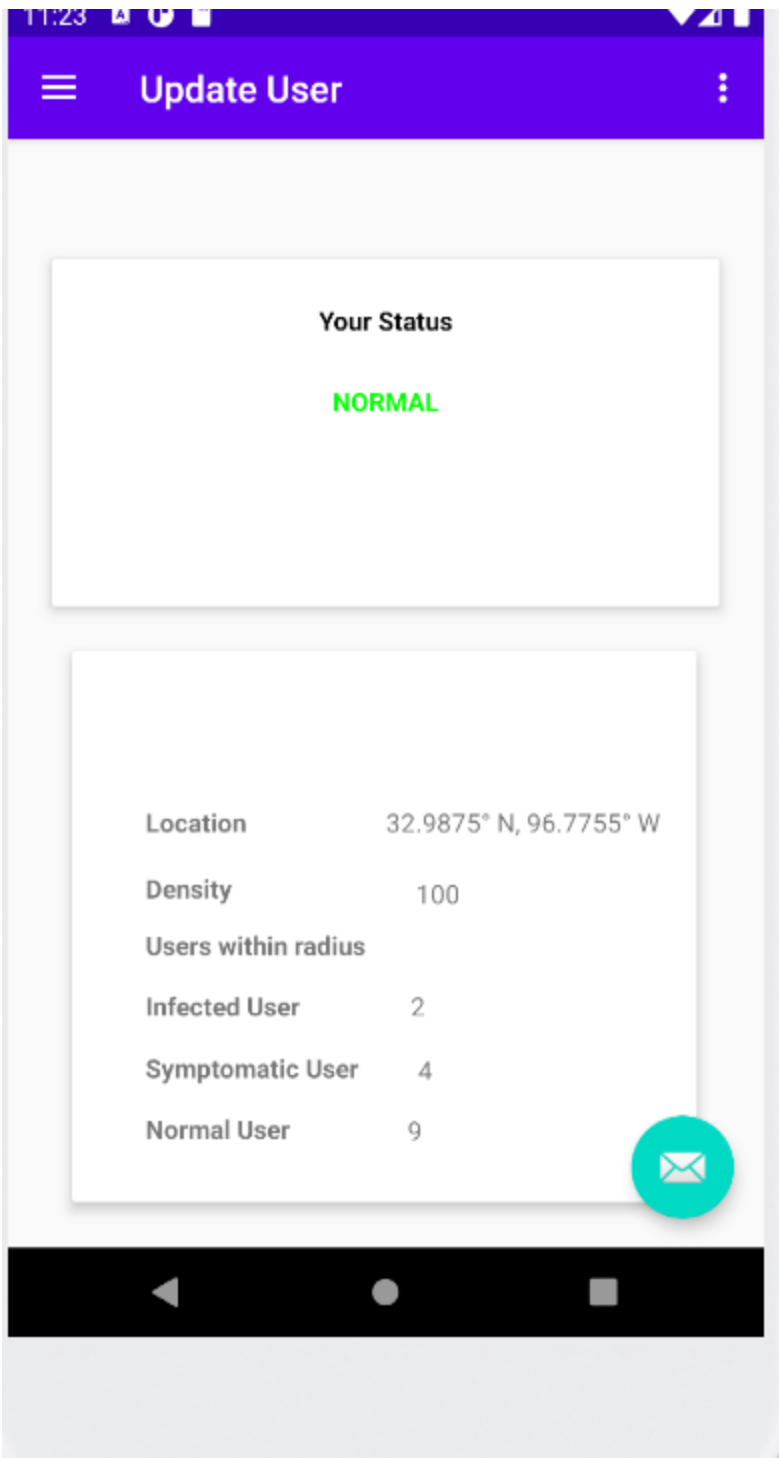
07/27/20

Quarantine End Date

08/18/20

UPDATE





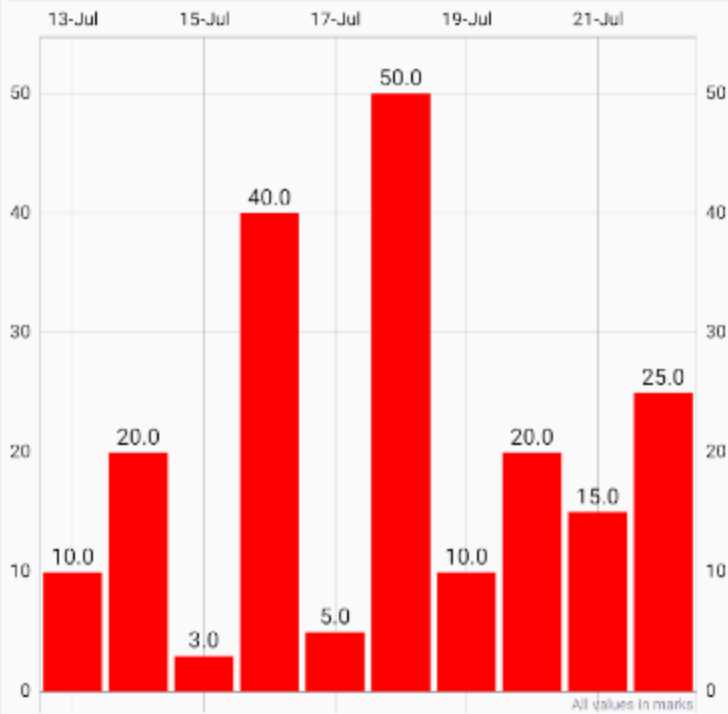
11:23



Disease Info

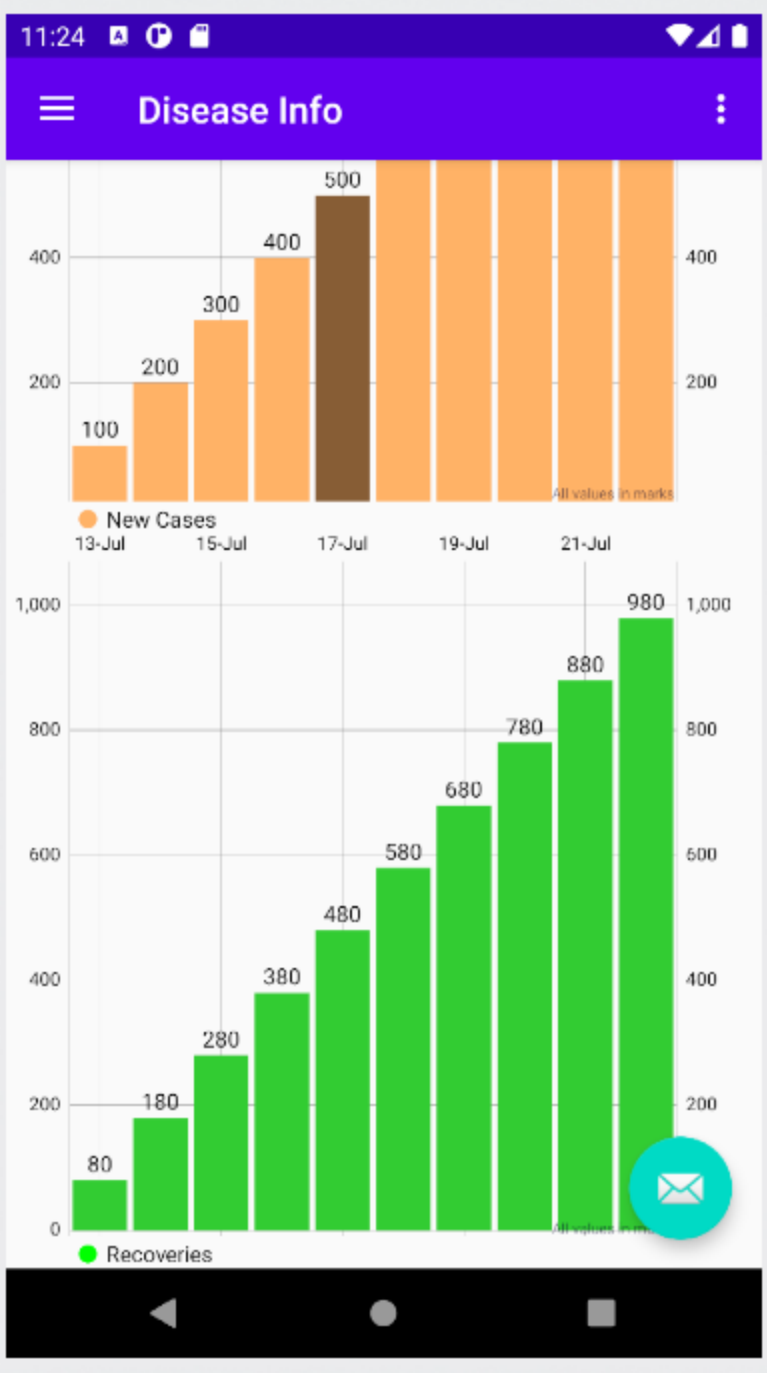


Dallas



● Daily Deaths





11:24



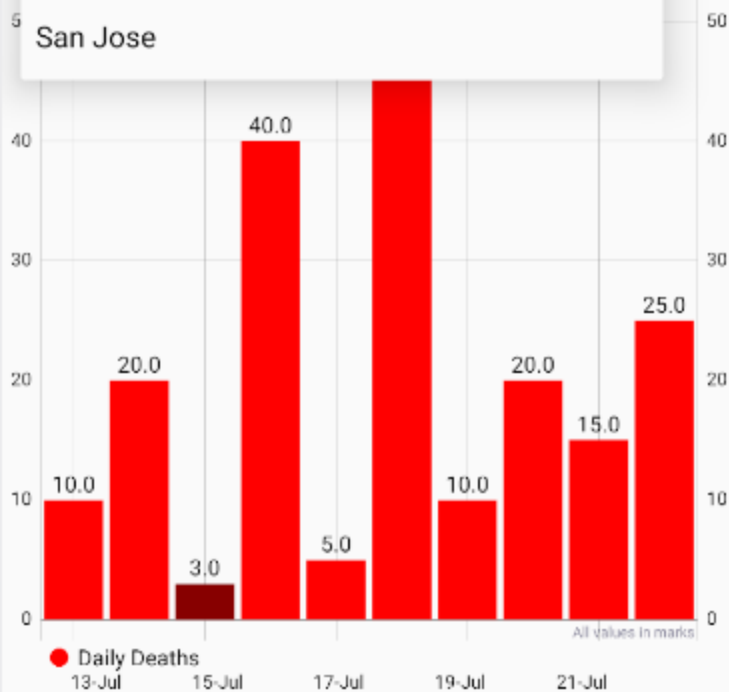
Disease Info



Dallas

New York

San Jose



● Daily Deaths

13-Jul

15-Jul

17-Jul

19-Jul

21-Jul

