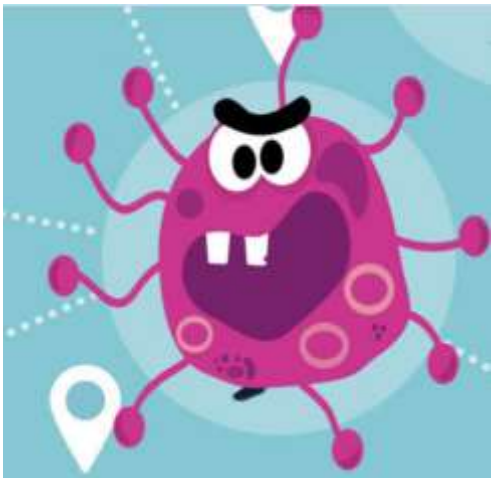
# Ramapo College of New Jersey

Senior Project

CMPS 450

Spring 2021



# Covid-19 Management Assistant

For the small businesses.

Submitted by: Bishal Thapa

Submitted to: Professor Dr. Victor Miller

**Table of Contents:** **Page Number**

Table of Contents: Page Number

## Section 5 ----------Conclusions:

### 1.1 Introduction:

This paper serves as a documentation for Android application (Covid Management Assistant) which helps to manage the problems created by the global pandemic Covid-19 within a small business where employees' size is approximately 10-100. This application is focused on small businesses as the structure of this application only supports small business and is not as effective if large number of employees are present. This application plays two main roles in helping an organization for fighting with Covid-19. Firstly, it helps to track and understand the transmission of covid by tracking user location, asking for employee's input and alerting employees based on the calculation. Secondly it helps for better management by providing important resources like self-assessment features, helpful resources link, and portal to share covid-results or insurance information.

### 1.2 Project Inspiration:

I found few applications on the android and apple store which were made to track and notify the users regarding covid transmission, but most of those applications were focused on large sample of population. For example, states like New Jersey have a smartphone application that tracks the covid transmission and notifies users. The problems of these kinds of large-scale applications were only few percent of the population used this application and even less percent of the population reported the results correctly, which made the application highly ineffective. So, to keep the target population sample small, target audience of this application is based on small businesses. Inside a small business population sample is relatively low, and if administration convinces the employees to strictly use the application, it might be available to get maximum coverage. As workplace is a major area where one can contract with other covid positive people, focusing on small business employees made sense to me.

### 1.3 Platform:

This is a GUI mobile application that can be run on android operating system.

### 1.4 Target Audience:

The main target audience of this application are employees of a small business. Since there needs to be administrative access to manage and review the covid results and insurance files, users have one of two privileges: member or administrator. Since the target audience is small, one default administration login credential is created so that they can access the functions of the application not available to the users with member status.

### 1.5 Project Features:

The project has following features:

1) User authentication (login, registration and forgotten password recovery)
2) Covid related statistics of the organization and current health condition of the user.
3) Room or Location based groupings which helps to identify employees in close contact with Covid positive employee. The application provides the employees to choose such four areas where they spend most of the time working.
4) Self-assessment of the symptoms so that better understand their current health condition and take further steps.
5) Portal to upload Covid results and insurance information for the employees and portal to review or download user submitted files for the administration.
6) Resources and links for the employees that helps to make informed decision regarding the Covid virus.
7) Ability to update and view the profile for the user.
8) Location tracking feature on the background that helps to capture user location coordinates on regular interval.

### 1.6 Summary:

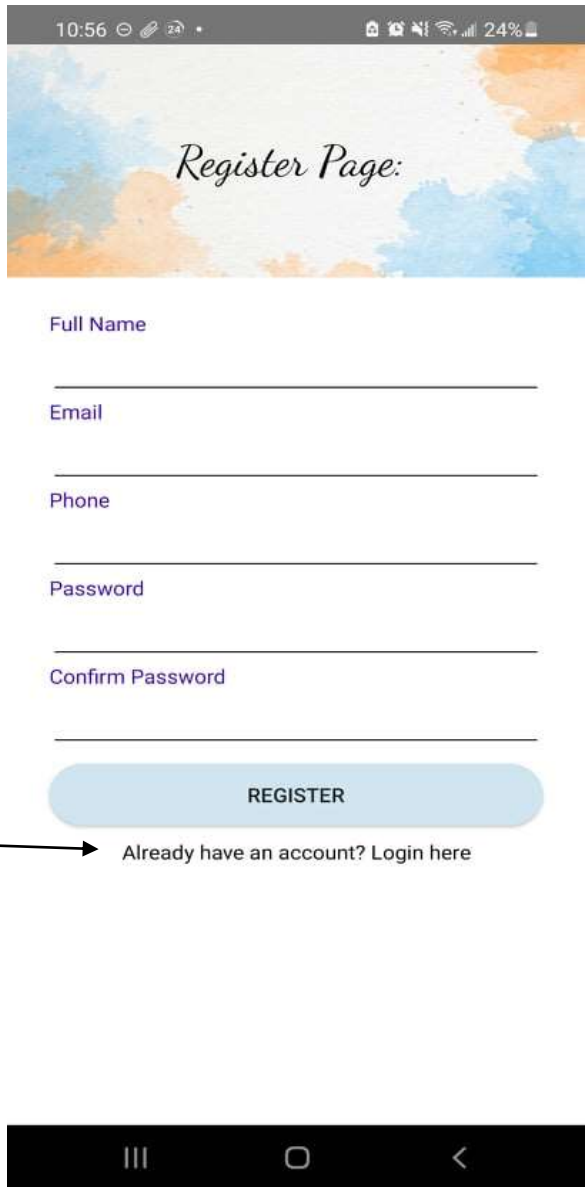
There were few limitations on the starting phase of the application development. Initially location tracking was a big question since most of the users do not feel comfortable being tracked. Secondly, the accuracy of the location tracking was also a big concern because, if the location were not accurate by less than or equal to six meters, than it would not serve the purpose of the application. The application was developed with focus on these two areas from the beginning, but the users should not be fully depended on its effectivity and rather should take it as a helping tool.

### 2.1 Installation Instructions:

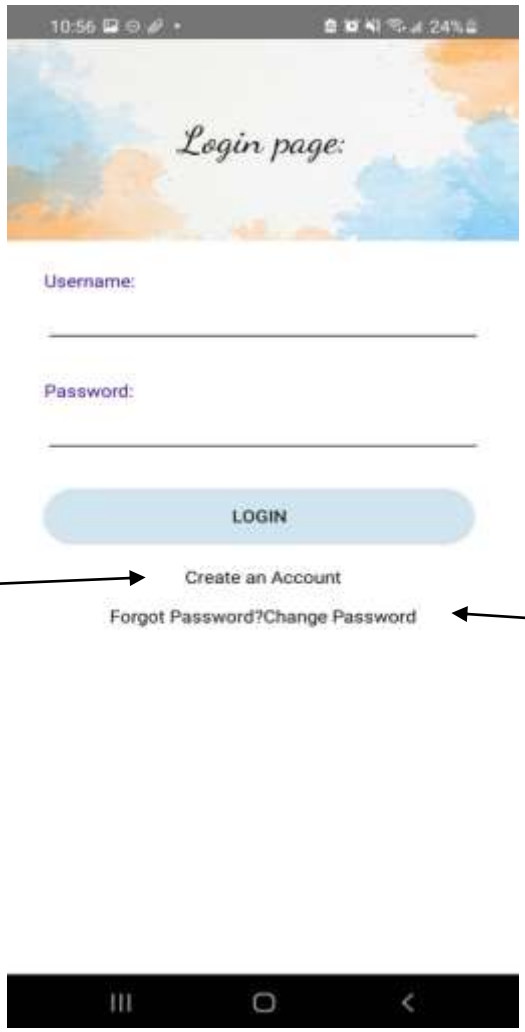
1) Download the file app-debug.apk from the link here.
2) Transfer the file you downloaded to android device or android emulator where you want to run the application.
3) Open CovidManager.apk file on the device with package installer or any other application installer application.
4) When the package installer shows the popup “For your security your phone is not allowed to install unknown apps from this source” click on SETTINGS.
5) Inside the settings toggle the “Allow from the source” button.
6) Go back to package installer and click on “INSTALL”.
7) Allow any permissions as needed.
8) Search for CovidManager application on your device applications list and run it.

### 3.1 User Registration and Login:

The user is directed towards the registration page whenever the user opens the application for the first time. To register for the application user needs to fill up the basic contact information and password that matched the requirement of the specific fields. Once the user signs up, they can use the same credential for login purposes after that.




Go to login page if the user is registered already.




### 3.2 Modifying Covid Status and reading notifications:

The main page or the home page of the application allows the user to do three things. Firstly, it allows the user to modify the current health condition. Users can toggle the covid status to positive or negative based on their health condition. Secondly, Users can see the notification below which mentions if there was a possible contact with covid positive users within their workplace. If so, it also shows the date when the system detected so. The third part of the home page allows the users to know more about the covid condition of the organization or the small business they work for. The line chart present in the screen allows the users to know the number of covid positive employees within their workplace. The x-axis is labeled with the date for each day of the last week. Y-axis is labeled the

positive count system detected for each day. This also helps the employers to better understand the overall condition and to take necessary steps.

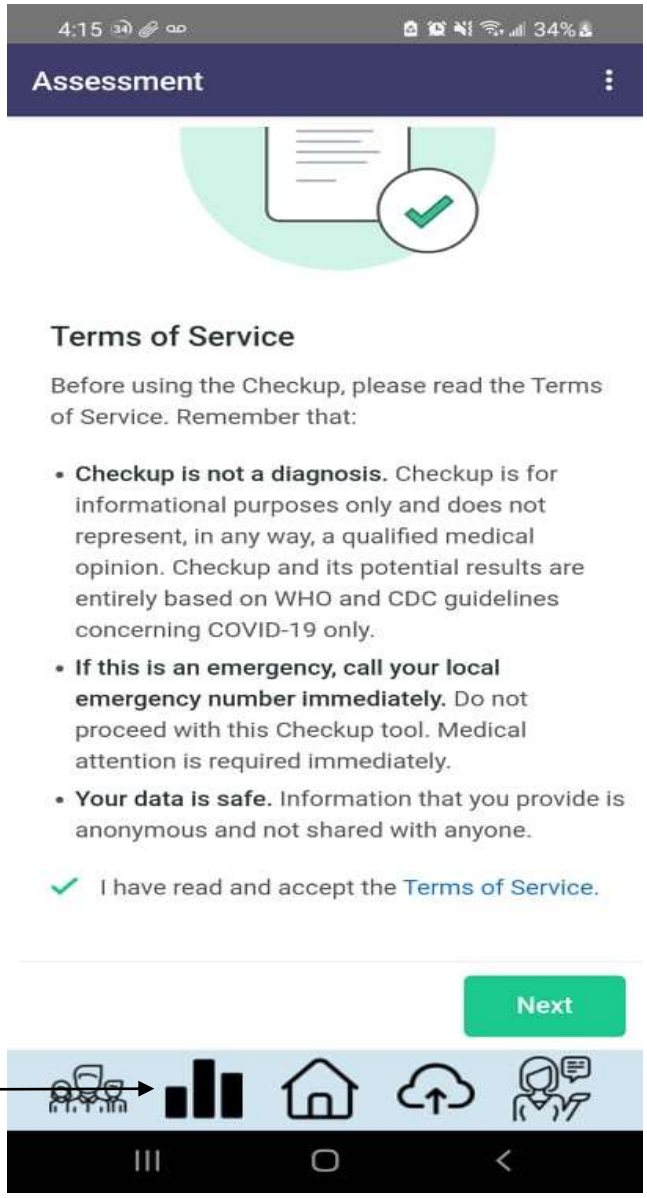

Notification if there was any possible risk detected and date risk was detected.

Toggle button for covid status

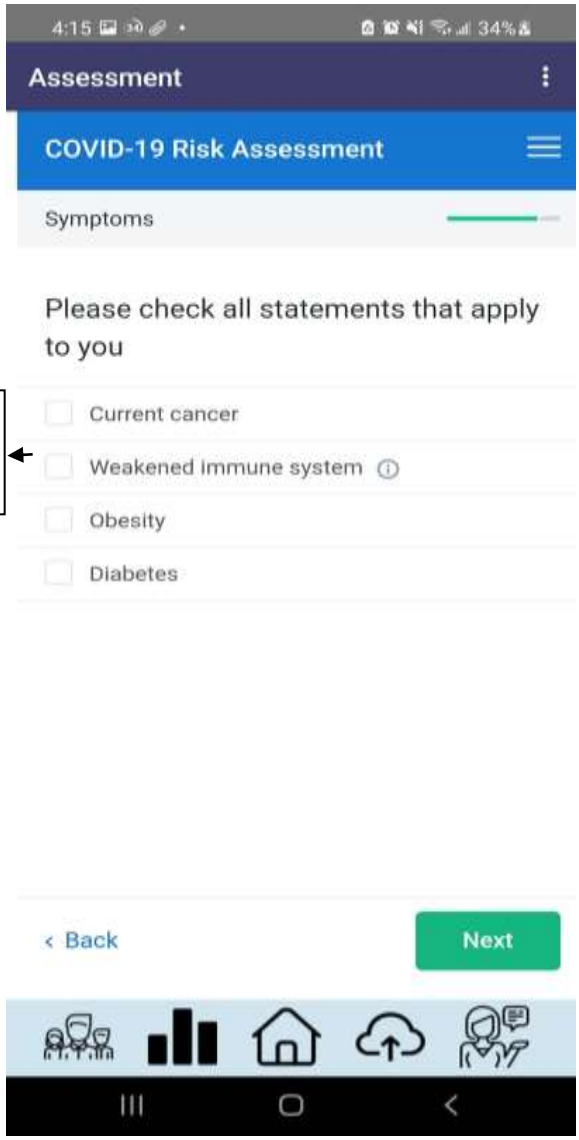
Line Chart with daily positive count.

### 3.3 Assessment of the symptoms:

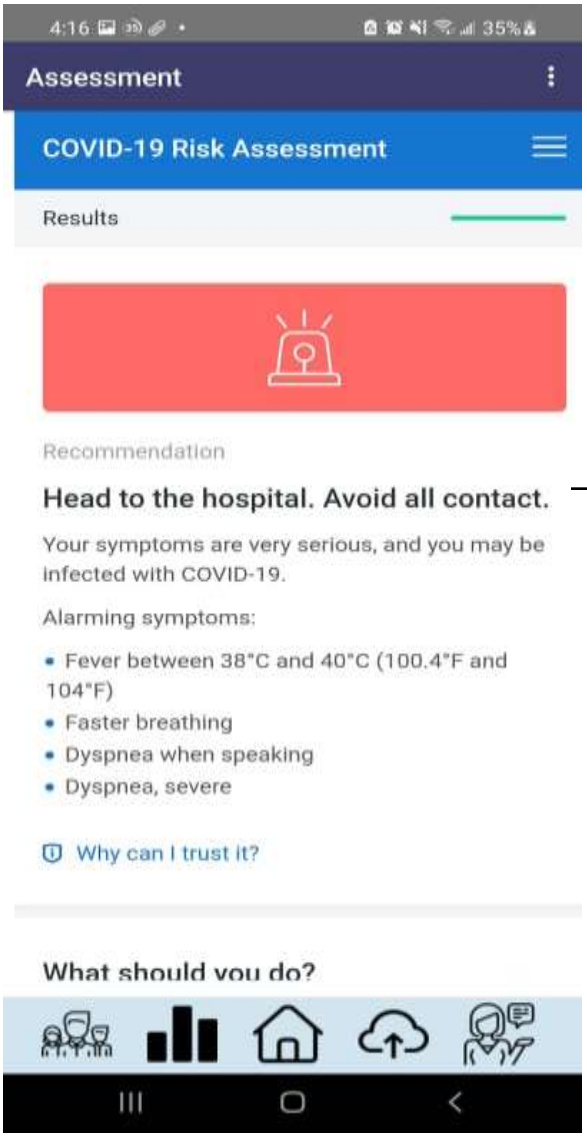
To assess the symptoms and get suggestions based on the users can go to Assessment activity. User can provide the inputs based on their health condition and they will receive a feedback at the end of the survey.




Assessment Page



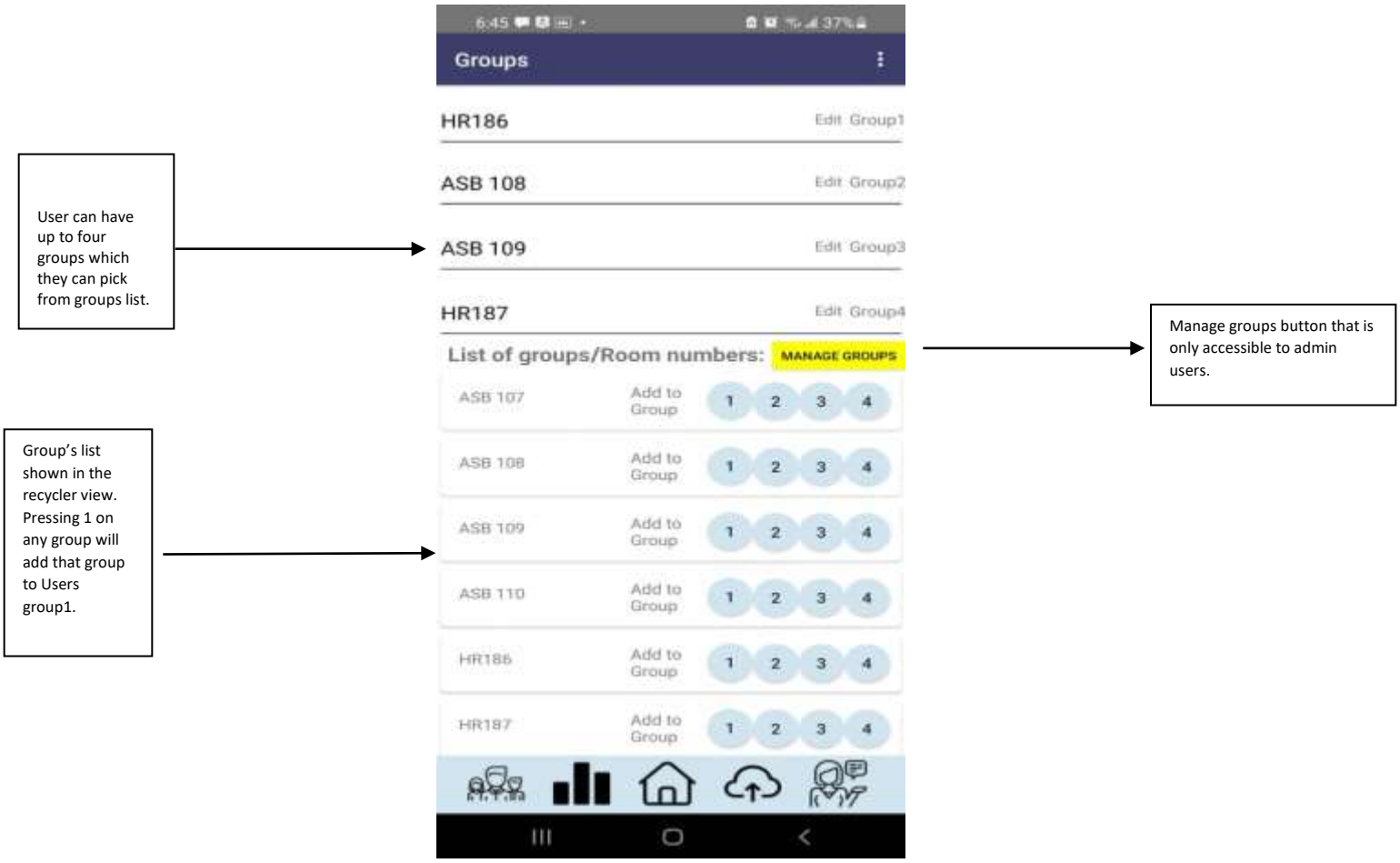

User input for self-assessment.




Suggestions and output based on user symptoms.

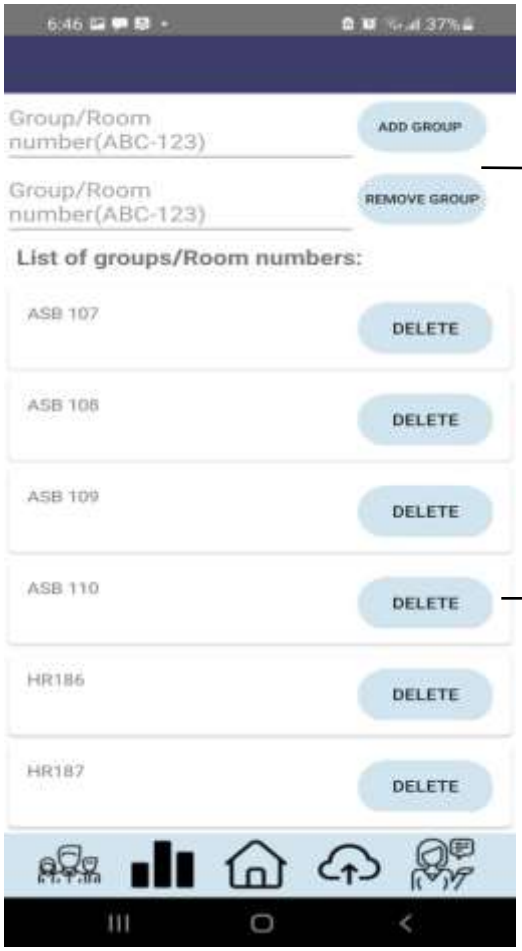
### 3.4 Joining and changing groups.

Users can choose the groups based on the location they spend their time at their workplace. If someone who stays in the same room or same location gets tested positive and reports it to the application, all the other employees get notified. Users can choose up to four different rooms or groups. They can also view all the rooms that are added by the administration. All the groups that a specific user belongs to are matched to notify possible Covid-19 contact whenever a new user tests positive. This calculation is done using cloud trigger functions without using users' resources.



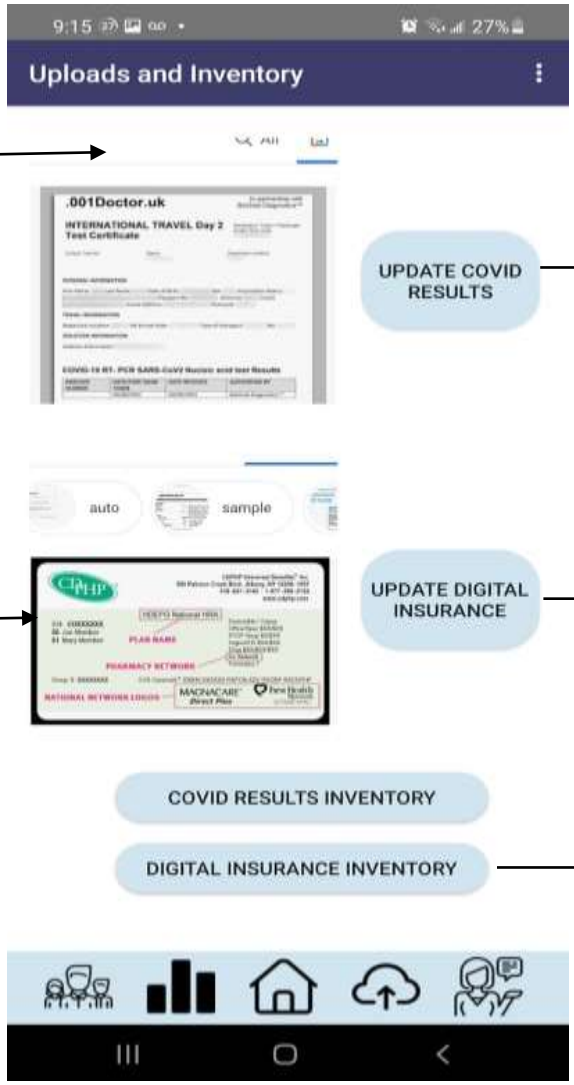

### 3.5 Groups management for the administration.

User with admin privilege can select or create a list of groups that employees or other users can join. To do so admin users need to go to groups management activity by clicking at the "manage groups" button present at the Groups activity. This button is only visible to admin users. After going to groups management activity Admin user can delete groups or add new group. There is also a recycler view which displays all the groups. From the recycler view admin can choose to delete a particular group.



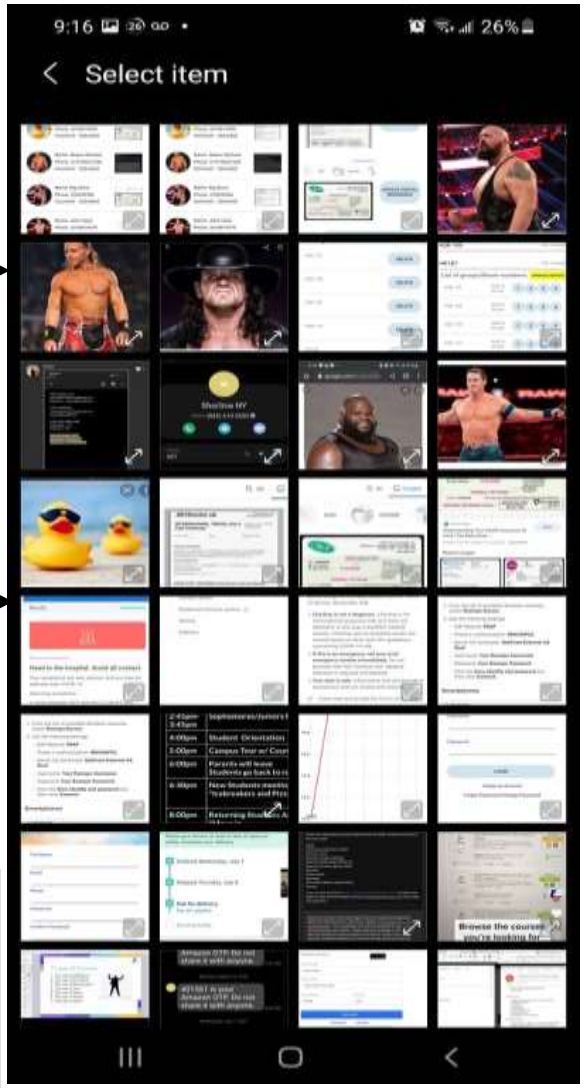

### 3.6 Uploads Activity

Users or the employees of the organization have access to the portal from where they can upload two kinds of files: test results and digital form of their insurance card. They can do so by clicking on the image icon next to “Upload test results” or “Upload digital insurance” on the Uploads management page.




Portal to upload covid test results and insurance card.

Preview of the files uploaded.




Clicking on upload or update button opens the gallery from where users can select the file.

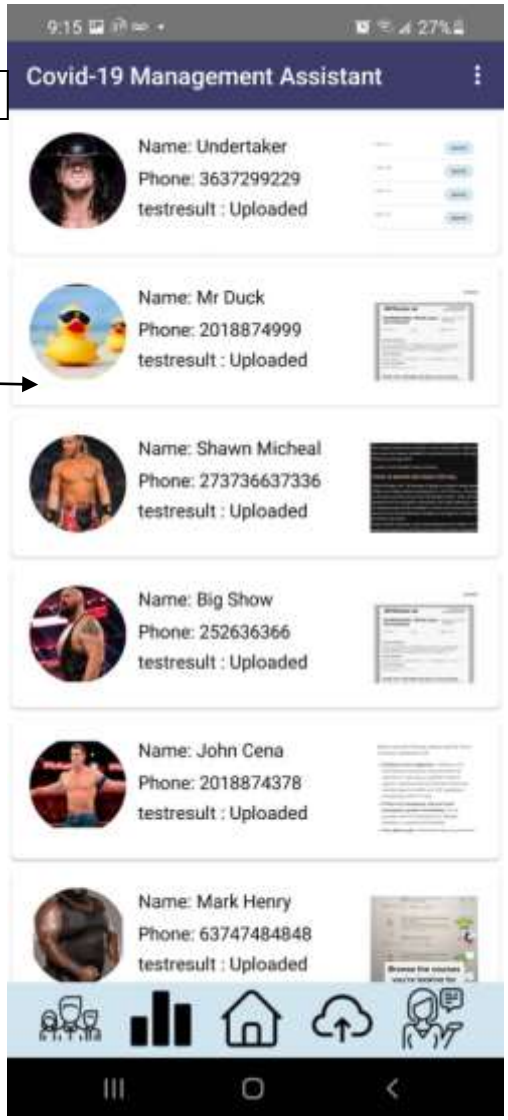
Only admin users have these buttons visible in Uploads Activity. Clicking these buttons will take the admin user to specific file inventory.

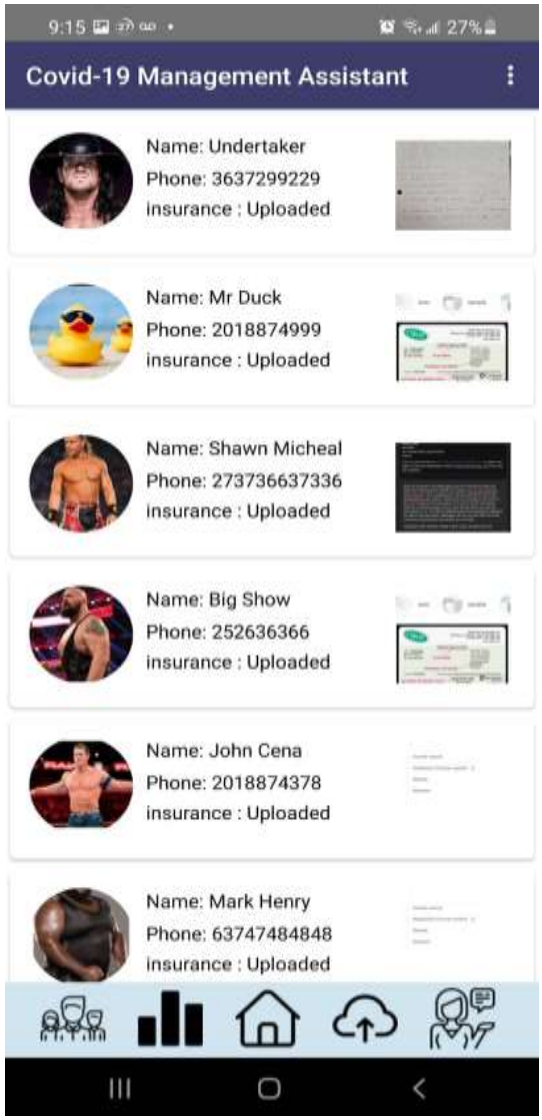
### 3.7 Uploads Inventory for the administration:

The administration can view the file uploaded by all the users. To do so they can click “Go to insurance inventory” or “Go to test results inventory”. This option is only available to admin users as visibility is hidden for other users. Admin user can view the uploads of all the users along with the basic information like name and contact number. Admin user can also download the file uploaded by other users. All the files are stored using firebase storage.

Inventory for test result.

User profile and status of file upload.




Inventory for Insurance cards.

Admin users are also able to preview uploads of specific users and then download the file as needed. They can click on specific user from the recycler view to expand the file preview.


### 3.8 Resources:

Resource page contains the resources like YouTube video and other useful websites from government agencies like WHO (World Health Organization), CDC (Center for Disease Control) and NIMH (National Institute of Mental Health). Users get the basic information regarding the Covid-19 from the video chosen and if they need specific information regarding vaccination sites and mental health resources, they can visit the websites.

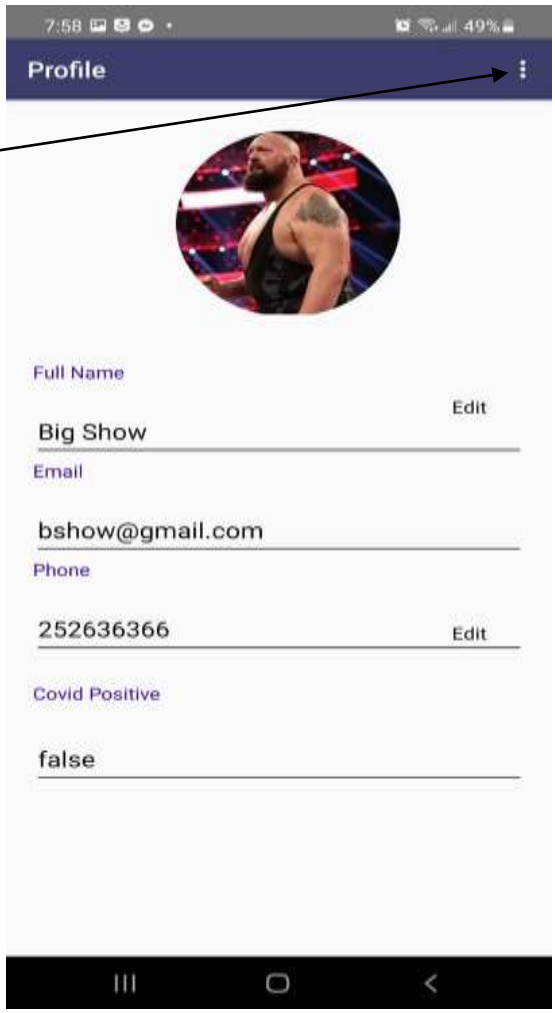

Resources like YouTube videos and important websites link are stored here.

This website is loaded whenever user clicks the CDC button at resources page.

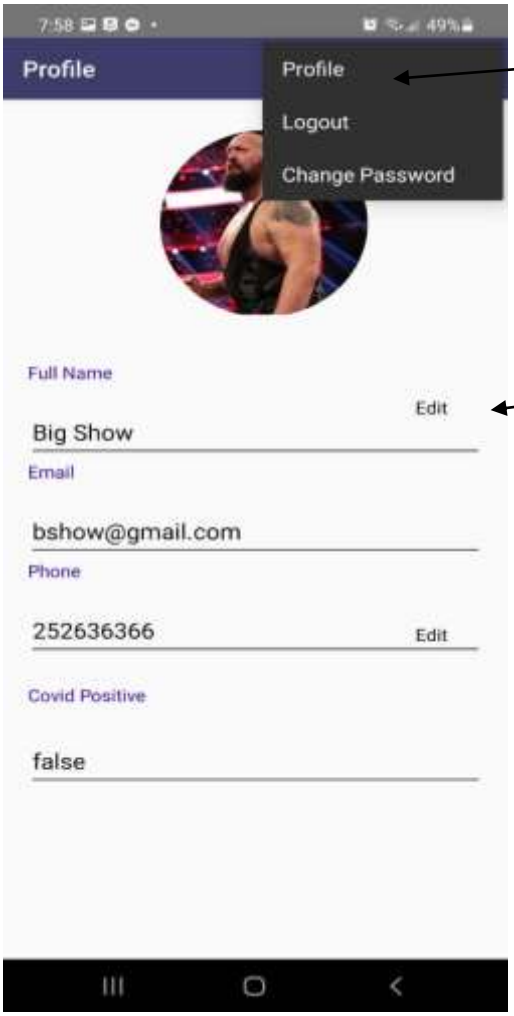
### 3.9 Profile Page

User can also view their profile and make important changes if needed. They can change their contact number and their name if needed. User also can change their displayed profile picture. To reach to their profile users can use toolbar. Toolbar is displayed in all the activities so the users can reach to their profile from any part of their application once they are logged in.

Users can click here to get to the profile page from anywhere.




Users can click on here to get to the profile.

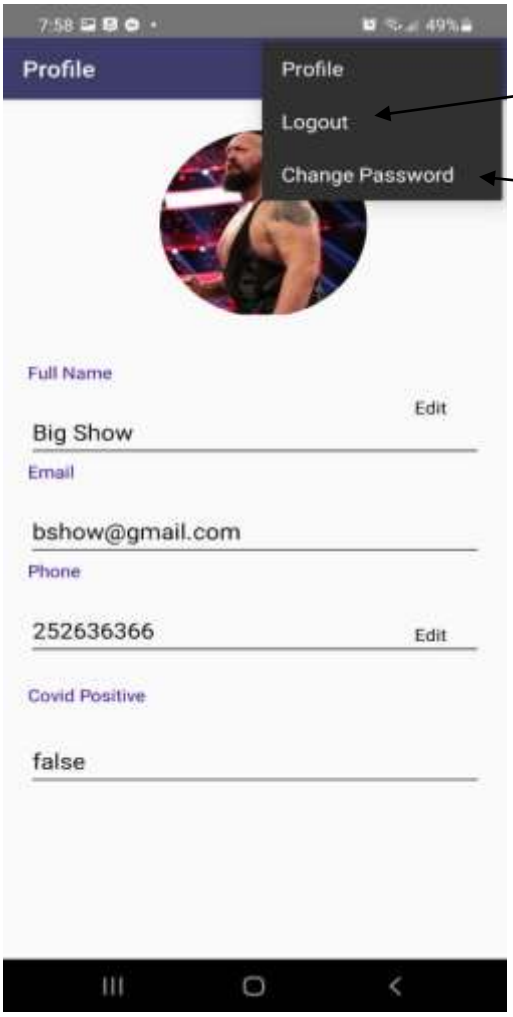



Users can change name and contact number from here.

### 3.10 Changing password and Logging Out.

Users can change password and logout from the application from the toolbar too. Password change is directed to forgot password page which sends the link to reset their password in saved email account. When user's logout they are redirected to Login page.

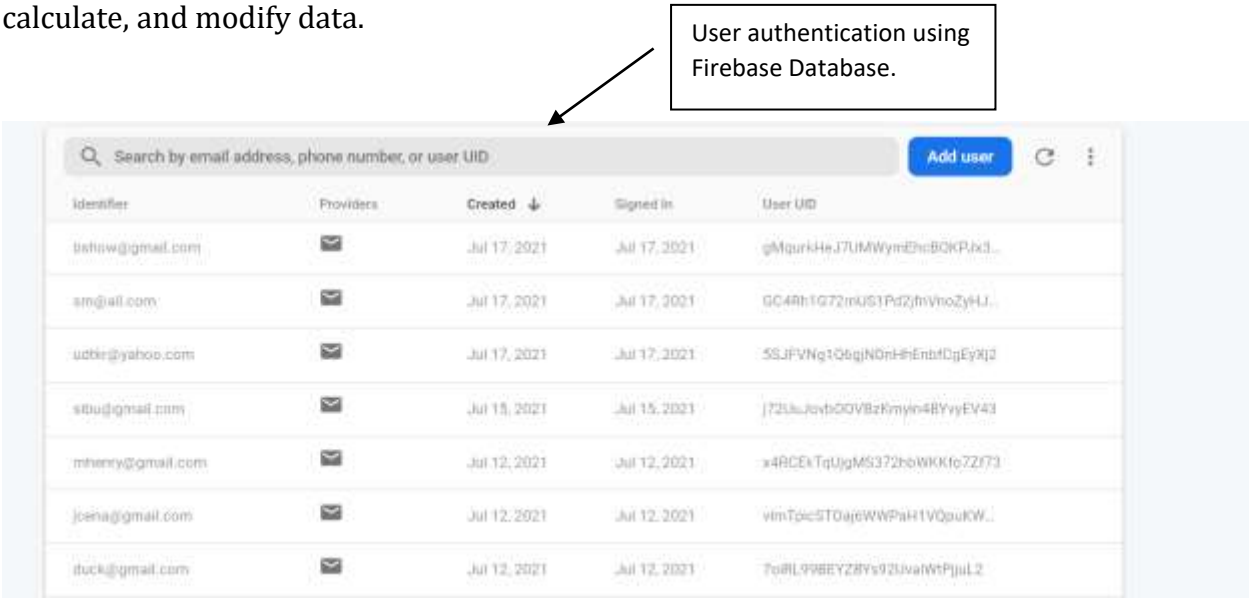
Clears user credentials and takes to login page.




Takes to forgot password page.

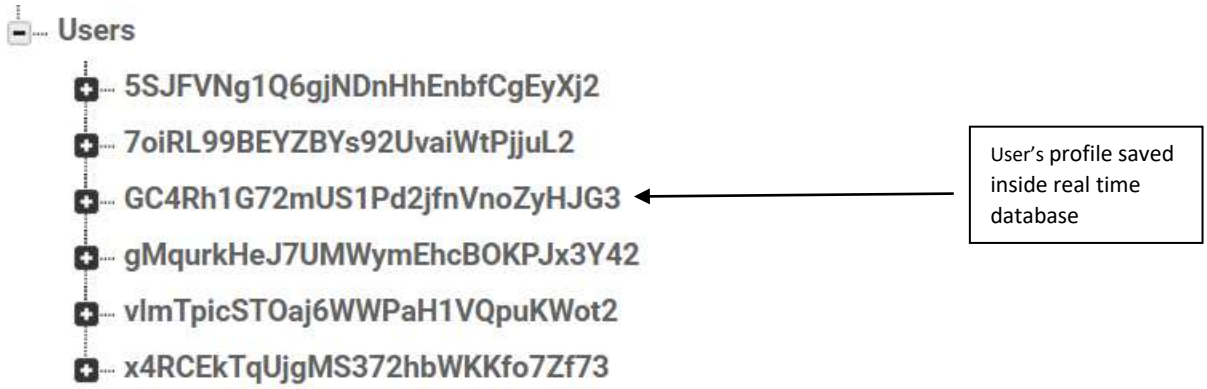
### 4.1 Design Summary:

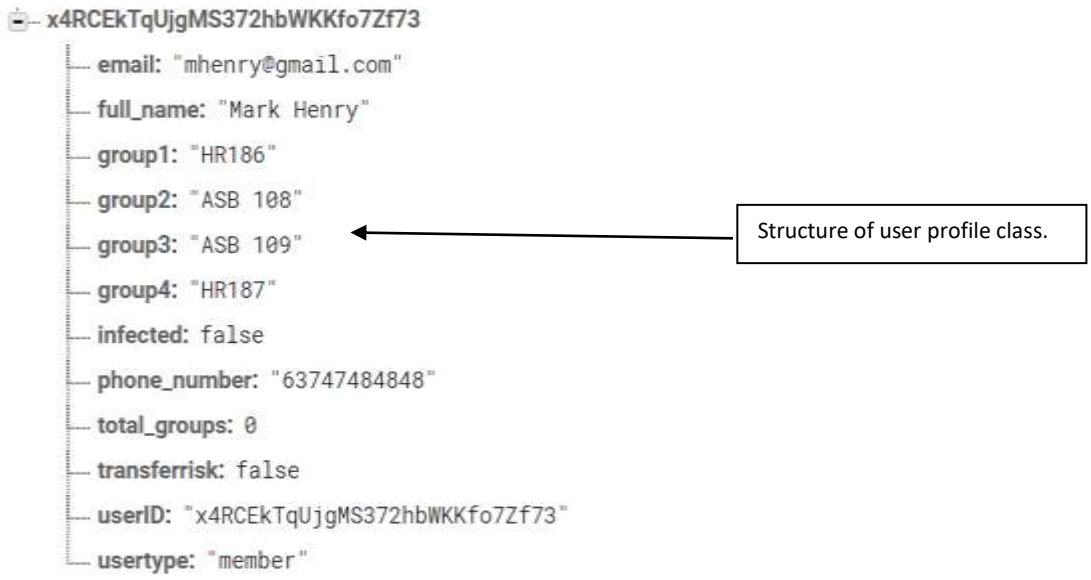
Firebase plays a major role in the design of this application. Firebase is used in areas like user authentication, database and works as a serverless backend to constantly monitor, calculate, and modify data.



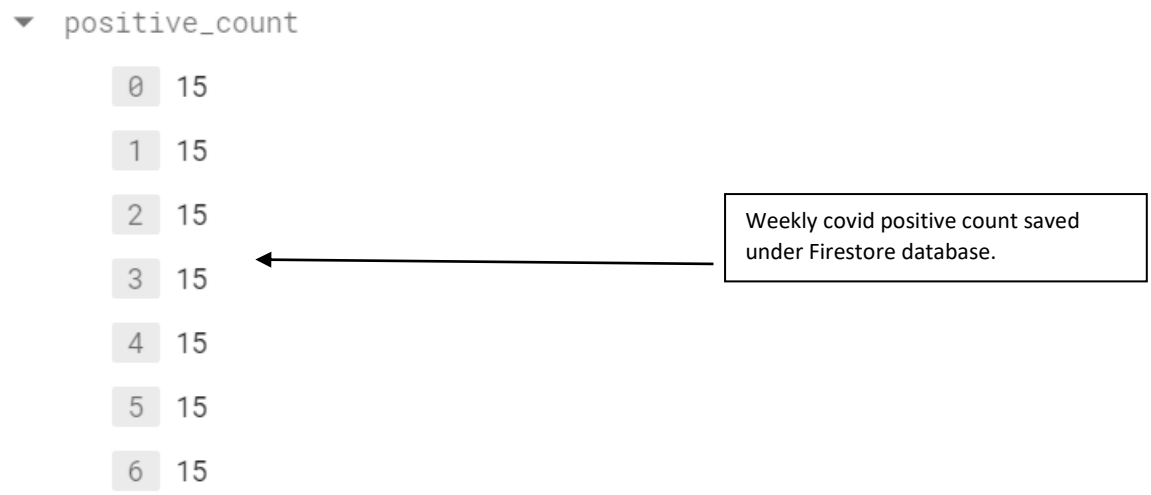

Firebase cloud functions are implemented to monitor any changes in the user data so that calculations can be made according to the change in data.

The data of the application is stores in Firebase Database in two different forms. Firstly, the data pertaining to user profile is stored in Realtime database.



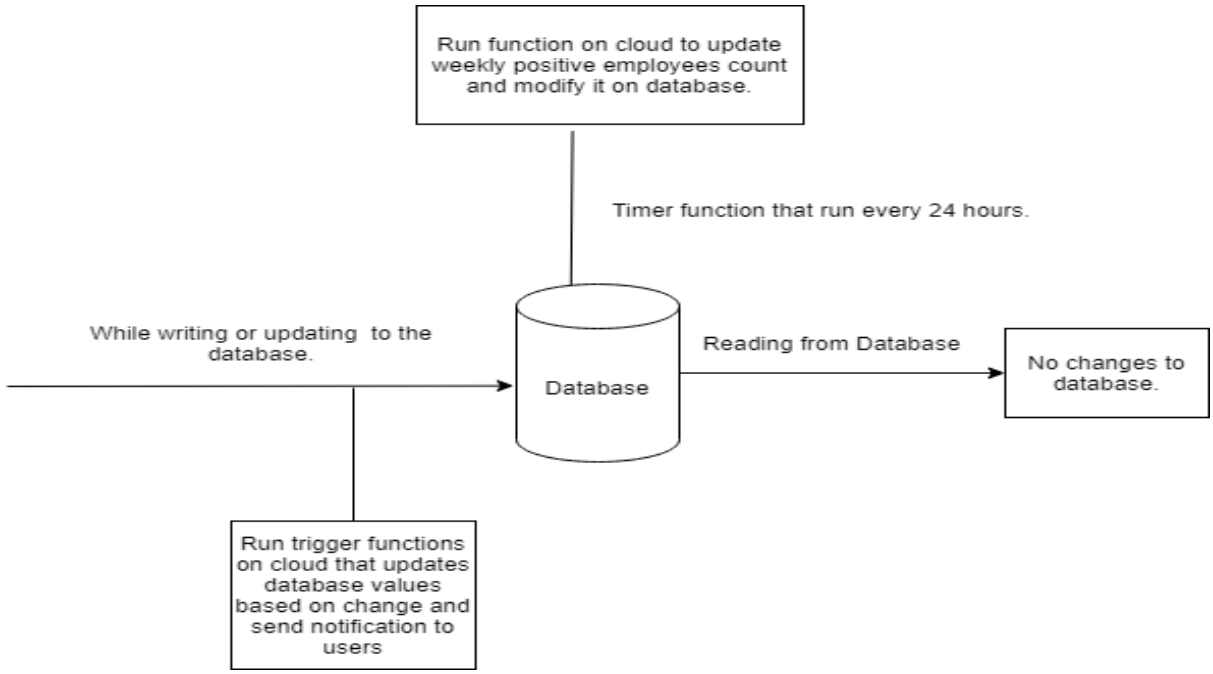



Firestore database is also used to store locations array, array list for weekly positive count etc. which require complex data types like geodata objects which cannot be stored on real time database

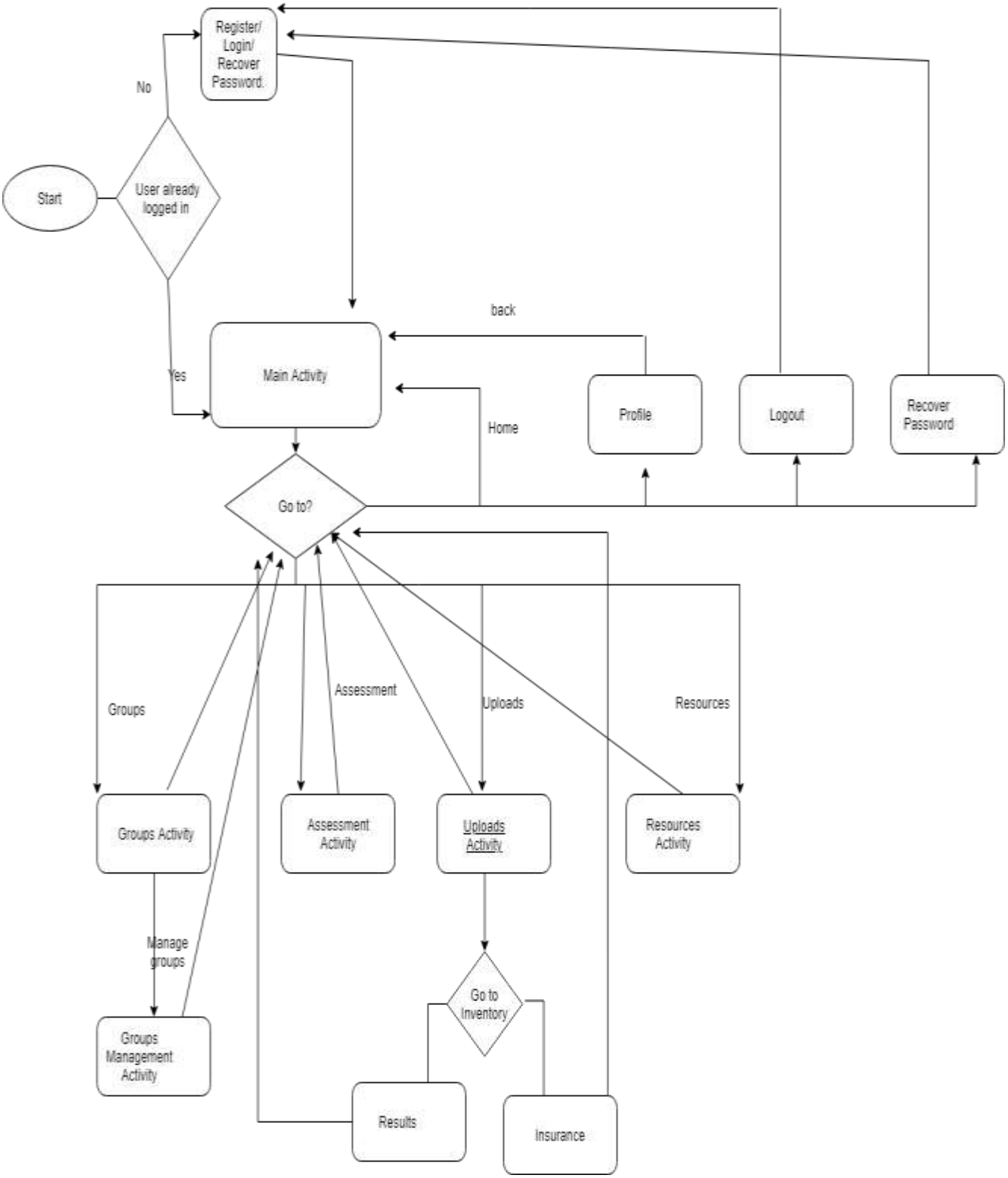



Firebase cloud functions are utilized to form a serverless backend system. Whenever a data is changed from the user side, cloud functions detect them constantly and perform any actions as required. To detect change in database, firebase cloud functions utilize trigger functions that are triggered every time a value of the database is created, deleted, or modified. For example, every time the user changes their covid status in the frontend of the application, that change in reflected in the firebase Realtime database. Cloud functions detect the change using trigger functions and detect the users who were in close contact and alert them if necessary. Cloud functions also utilize timer functions that after a specified amount of time. For example, the backend system maintains the weekly covid

positive employees counts by reading the number of positive employees every 24 hours and maintaining the array list for each week. The below chart explains how the database react to read and write conditions as well as trigger functions.



## 4.2 Flowchart describing the flow of activity screens for frontend:

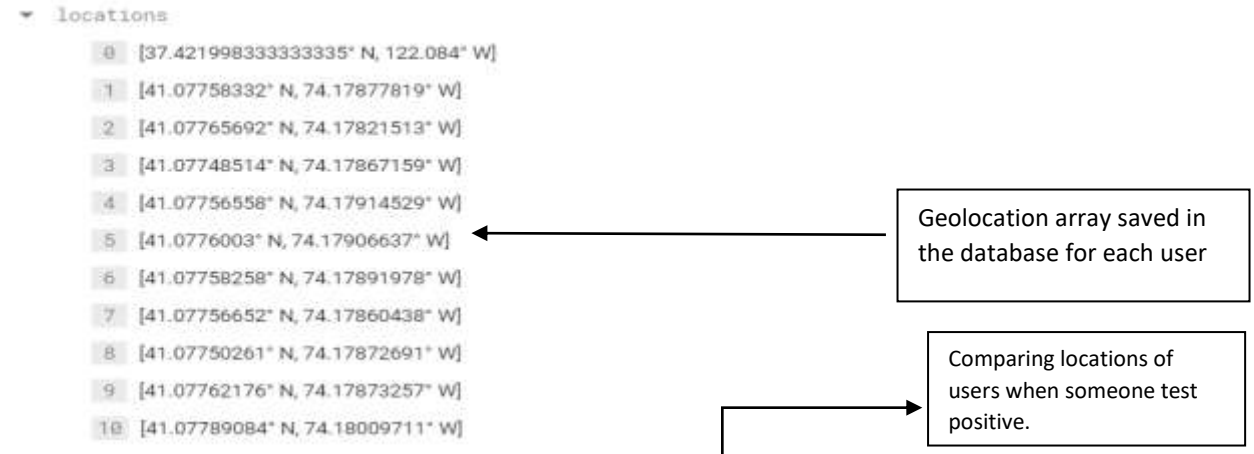


### 4.3 Background Location Tracking:

This application relies heavily on the location tracking feature. To find if the two users were in a same location in the work environment it saves the geo coordinates of the user every thirty minutes. The tracking is strictly based on 30 minutes timer, and it is not affected by user motion.

The background location tracking process is initialized when the user opens the main activity of the application for the first time. After that application runs on the background and consistently sends the geo-location coordinates to the database regardless of the user activity if the application is open.

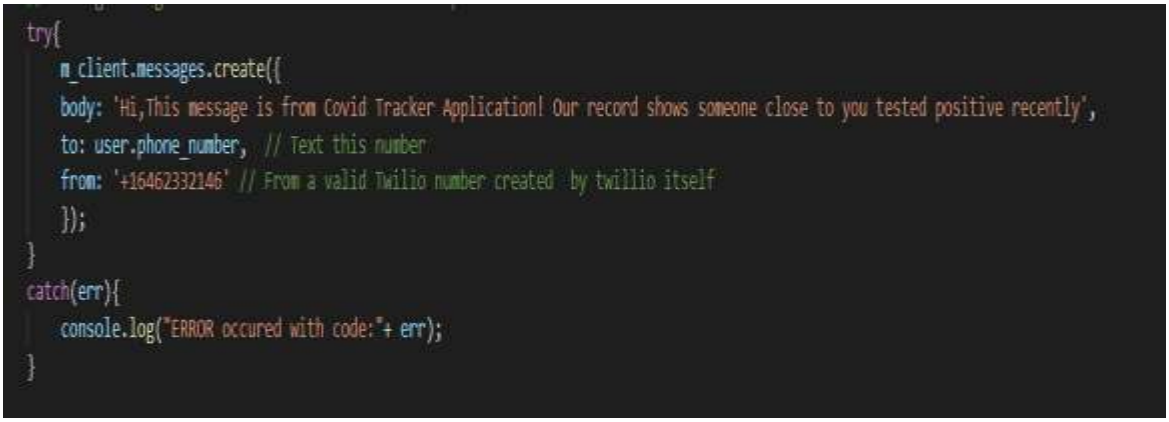
The location coordinates on the database are saved as “geo-point” data types. It contains latitude and longitude for a specific location. An array of such geo-point objects for each user which represents all the locations user spend their time at their workplace. Arrays of each user are matched with each other to see if two users were in close contact with each other.




```
if(testpoint._latitude===positivepoint._latitude  && testpoint._longitude===positivepoint._longitude){

    //getting new data time object for current date
    var datetime = new Date();

    //storing latest date for the user who received alert message.
    admin.firestore().collection("latest-dates").doc(childuserID).set({
        timestamp: datetime
    });

    //Updating their transfer risk to true as there was a close contact.
    usersref.child(childuserID).update({transferrisk:true});

    //sending message notification to user with saved phone number
    try{
        m_client.messages.create({
        body: 'Hi,This message is from Covid Tracker Application! Our record shows someone close to you tested positive recently',
        to: user.phone_number,  // Text this number
        from: '+16462332146' // From a valid Twilio number created  by twillio itself
        });
    }
    catch(err){
        console.log("ERROR occured with code:"+ err);
    }

}
```

### 4.4 SMS Notification:

Whenever a possible transmission is detected, SMS notification is sent to the user. To do so Twilio API has been imported. Twilio API takes the receivers contact number and message to be sent to the user. The default contact number user provides during signup process is used for this purpose. This is done every time a match is found in user groups or location co-ordinate of the covid positive user with another random user within organization.



```
try{
    m_client.messages.create({
    body: 'Hi,This message is from Covid Tracker Application! Our record shows someone close to you tested positive recently',
    to: user.phone_number,  // Text this number
    from: '+16462332146' // From a valid Twilio number created  by twillio itself
    });
}
catch(err){
    console.log("ERROR occured with code:"+ err);
}
```

### 4.5 Use of external APIS and resources:

This application utilizes few external resources and APIs for the activities which helped to application to function more smoothly. Here are the few libraries and resources used for the application:

- **M-Chart library:**

  Weekly covid count graph plays an important role for displaying the status of the organization by displaying the data visually. In order to achieve this functionality in the application, M-chart library is used.

- **“Infermedica” Self-Assessment tool:**

  For the self-assessment page, a link to the webpage designed by Infermedica has been used. It is displayed on top of a web view which takes the URL of the website. It also allows users to switch between different languages for self-assessment feature.

- **Glide Library**

  Glide library has been used throughout the application to display images in different shapes. It supports displaying images in circular shapes for profile picture and even zooms in when using it as a preview using image bitmap for users. Another important feature glide library has is it uses caching feature which makes sure the most recently used images are stored in memory so that it does not have to make request from database. This makes sure user experience is smooth when switching between the activities.

- **Android-YouTube-player library**

  Android-YouTube library is used to display the YouTube videos to the user under the resources page. This library uses Iframe player API which allows playing YouTube video on android app without having application installed on the phone. It is also highly customizable.

### 4.6 Classes used:

- **Assessment Activity Class**
  Assessment Activity deals with handling the self-assessment tool for the users to get feedback regarding their health condition. It uses a WebView where it displays 'Infermedica' webpage.

- **Background Location Tracker Class**
  Background Location Tracker class is used to collect the user's location in regular intervals. For this, background location tracker class is triggered by the main activity class. Once it is triggered there is a timer function setup which runs the location tracker to capture the location and update to database every 30 minutes. For this function to perform its task properly, users need to grant the location access of the device for the application.

- **ForgotpasswordActivity Class**
  ForgotpasswordActivity class is used to help the user recover or change the current password. The user email is taken as an input and then it uses firebase Auth instance to send the email link to the user from where they can change their passwords.

- **Groups Activity Class**
  Groups Activity Class is used to change and manage the user groupings based on the location users stay at their workplace. All the users in the application have upto four groups. Initially all the groups are set as null. Users can pick a valid group from the list, can replace their current group, or view the list of all the valid user groups in a recycler view.

- **Groups Adapter Class**
  Groups Adapter is used as an adapter for groups manager class's recycler view. Each groups adapter consists of a group name and a button using which admin users can delete a group. When a group is deleted, the change is also reflected on the database.

- **Group selector Adapter Class**
  Group's selector Adapter is used as an adapter for the groups Activity class recycler view. Each adapter consists of group name and 4 buttons [1,2,3,4]. Each numerical button will move the displayed groups to the users group holder position. For instance, clicking button '2' on group "Room112" will set the users second group as "Room112".

- **Groups manager Activity Class**
  Groups manager Activity helps the admin user to create new groupings based on the employee's location in the work area. They can also view all the groups that have been created and delete the groups directly from the recycler view.

- **Launcher Activity Class**
  Launcher Activity Class helps to check if the instance of Firebase user is present or not. If there is a valid firebase user instance present user is taken directly to the main activity or else user will be directed to login page.

- **Login Activity Class**
  Login Activity Class helps the users to login to the application. The authorization of the application is handled by firebase. If the login credentials are valid user is directed to login page.

- **Main Activity Class**
  Main Activity Class is used as homepage for the application. Initially when the user opens the app for the first-time, user is asked for all the necessary permissions required to use the application. Main Activity houses stores the basic information like covid chart and user's status. Main Activity uses the current user's Firebase Authorization ID to pull up data from the database and displays it according to the need. This class also handles the change in user's covid status and initializes the background location tracking.

- **Profile Activity Class**
  Profile Activity class helps the users to view their profile and make changes to their profile. Users can view their current profile picture as well as update it. Users can also view their name, contact, email, and address. Users are allowed to change contact, address, and name but not their email.

- **Register Activity Class**

  Register Activity class allows the users to create a new account to get access to login page. This class checks if the users has entered all the required inputs and if that is the case new firebase authorization account is created. User's profile is stored to the Firebase Realtime database.

- **Resources Activity Class**
  Resources Activity class deals with playing a YouTube video and redirects users to different activities based on the images or buttons user choose. The resources are

related to the information regarding Covid and making important decisions to fight against Covid.

- **Resources Webpage Activity Class**
  This class allows a specific website to be displayed on the Web View. The URL of the website is passed as a bundle to the intent from previous activity which is used by this class to display on the webpage.

- **Toolbar Activity Class**
  Toolbar Activity Class is used to display toolbars for all other activities. Other activities extend this class and use it to login, change password on get to the profile from any activity.

- **Upload Activity Class**
  Upload Activity Class manages the user uploads. It involves allowing user to select the file from the gallery and saving the file to the firebase storage under user id so that it can be retrieved easily. Uploads Activity also gets the image from the database to display preview of the file to the user.

- **Uploads Management Activity Class**
  Uploads Management Activity Class deals with showing the admin user all the file uploads that have been made by the user. In order to do so it uses a recycler view and User Info Adapter class as recycler view adapter. Each adapter holds user info and file upload status. Admin users can also view the uploaded file by zooming and there is a download file button that allows the administration to download the file into their device if needed.

- **User Groups Class**
  User groups class is used to hold the information regarding the user groups. The get and set method for this class makes easier to store groups to database and retrieve from the database.

- **User Info Adapter Class**
  User Info Adapter is an adapter class that is used to display user information and file upload status inside the recycler view used at Upload Management Activity class. It uses on click listener to allow displaying individual files for the user.

- **Users Class**
  Users class is used to store user related information like name, address, contact and user groups. It consists of get and set methods which makes it easier to change User profile values and storing to the database.

**Challenges Faced:**

When I started this application, I did not have proper understanding of how backend and front end worked together for mobile applications. As I had worked with web applications before, I was expecting to design the application using rest APIs with a backend server. But as I learned more, I had problems creating a backend server due to lack of experience and could not come up with proper solution. This is when I came up with Firebase cloud functions. Firebase authorization and database were already on my list to use for the application. Since the Firebase functions helped me to monitor and modify data with the custom functions I could write, it seemed like a perfect solution for me.

One of the main problems I faced when using Fire Store cloud functions was testing and deploying functions to the cloud. Since I had no knowledge to test cloud functions without deploying them to the server, I had to deploy it to the server every time I needed to make a change. This was a hectic process because I had to deploy the whole code a lot of times to test the code which took 3-5 minutes every time I needed to deploy.

Apart from that I was troubled a lot by the asynchronous nature of JavaScript. When designing cloud functions for the first time, the outputs I received were confusing because of the asynchronous code I wrote. This took away a lot my time while testing and debugging. I was able to solve this problem using the 'async and await' syntax and fixing various other mistakes I had made.

One of the other challenges I faced was running background task on specific time. According to the new android standard, Android operating system did not allow running timed processes on the background. This caused the location tracking (60 times per hour when testing to not record it properly). I was unaware of such limitation and every time I was recording a location, time frames were not consistent. As I became aware of this problem, I made the app to record location co-ordinates every half hour which helped the process to run approximately 2 times an hour even though the timing might not be consistent.

## Conclusions:

This application was a great learning experience for me, and I have better understanding of how backend and frontend interact and work together. Prior to this project I was unaware of backend and frontend frameworks, and I had hard time to figure out right kind of language and frameworks to use for any mobile or web applications. After the completion of this project, I am more aware about the options available for mobile as well as software developers. At the end of this project, I am happy that I decided to use various tools introduced by the Google Firebase. They made most of the tasks easy and reliable. Although I did not make use of rest APIs for this project, I was constantly researching about it, so I developed a better understanding about it.

Few other tools I learned because of this project are GIT and android studio. I made use of GIT to store the versions of code and learned concept like branching. This was important for me as it helps greatly in professional work environment. I also got an experience to use external libraries for this application.

If I were to do this project again, I would choose cross platform like React-Native so that I would not have to write the code twice. Although I feel like I have done a considerate amount of work in making this application scalable, I would like to work on this more and deploy this to the android app store.

**References Used:**

Assessment page link - *https://covid19.infermedica.com/en/#0-99990*

https://stackoverflow.com/questions/5947775/android-locationmanager-requestlocationupdates

https://stackoverflow.com/questions/45294288/creating-charts-using-mpandroidchart

Recycler view-

https://www.youtube.com/watch?v=9tD6uCOdvmY&list=PLirRGafa75rSMDp5bORq_eHjMLKqJ2EYO

https://www.youtube.com/watch?v=sZ8D1-hNeWo

https://stackoverflow.com/questions/6217378/place-cursor-at-the-end-of-text-in-edittext

How to reset forgot password: https://www.youtube.com/watch?v=UMNeeMSUZl0

https://stackoverflow.com/questions/6925156/how-to-avoid-a-toast-if-theres-one-toast-already-being-shown

https://stackoverflow.com/questions/14876273/simple-example-for-intent-and-bundle

Menu bar - https://www.youtube.com/watch?v=oh4YOj9VkVE