Safe Back School App

Smart Tracing for School Reopening

VE441
Instructed by Pradeep Ray and Fethi Rabhi

Team:
Ruoxin Geng
Bingcheng Hu
Daiyang Li
Pengqi Lu
Yuanjie Tao

Part 1. Requirement Analysis

1. Business Problem Recap

We are all in our "time of cholera". The only difference is that in our world, cholera is replaced with a much more fierce disease, COVID-19. The current pandemic has caused a series of serious problems to all humankinds, including life threats, economic crashes and most significantly, stoppage in almost every industry.

As students, what makes us concern most is likely the stoppage in schools, especially the education after this summer. Shall we go back to campus for traditional face-to–face mode? Or shall we keep online? A hybrid mode (some courses in person and some online) has been considered broadly, especially after the announcement from the United States Immigrants and Customs Enforcement that "nonimmigrant F-1 and M-1 students attending schools operating entirely online may not take a full online course load and remain in the United States."

The primary concern is "is that really safe to go back to classrooms?" Maybe not a single person can be a hundred percent confident. Thus, the second question to ask becomes "how can we lower the risks as much as possible?" To answer this question, our team decide to develop the "Safe Back School App".

2. Features and User Stories

The value of our app is serving as a good platform to meet our customers' requirements. The general requirements are composed of demands from two categories of users. Students and stuff require a platform to record daily health status and get notified when they have a high potential risk to get infected. Administrators require a platform to easily monitor everyone's health status and find all close contact of an infected person efficiently.

Since the advantage of our app is based on the Bluetooth tracing technology, we mainly focus on designing features for our major customer, students and stuff. The general requirements for students and stuff can be split into complex features. Those features can be classified into three functions which are Bluetooth tracing function, data collection function and general function.

2.1 User story about Bluetooth tracing function

Requirement: Bluetooth tracing function

Feature: Keep Bluetooth on to automatically upload close contact information
As a: processor
So that: I can store the close contact information

I want to: collect address of other detected Bluetooth devices

---

Feature: Push notification when user is at high risk to be infected
As a: student
So that: I can get early detection and isolation and help control epidemic
I want to: get notified
GIVEN that my phone is on
WHEN anyone in my close contact database is diagnosed
THEN I can get a popup notification

2.2 User story about data collection function

Requirement: Data collection function

Feature: Daily Reminder

As a: student
So that: I won't forget to fulfill the health report
I want to: set the app to notify me at a certain time
GIVEN that I am on the setting page
WHEN I set the notification to be pushed at 9:00 a.m.
THEN I can get a popup notification to fulfill the report
WHEN it's 9:00 a.m.

Feature: Simplify health data filling process

As a: student
So that: I won't feel tired about the repetitive work
I want to: set the app to notify me at a certain time
GIVEN that I am on the setting page
WHEN I set the notification to be pushed at 9:00 a.m.
THEN I can get a popup notification to fulfill the report
WHEN it's 9:00 a.m.

Feature: Visualize health status

As a: student
So that: I can show it easily and take it as a proof to enter classrooms or canteens
I want to: have a visualized window to show my health status
GIVEN that I am on home page

WHEN I have a high risk to be infected
THEN I can see a red window with a crying face
WHEN I don't have a high risk to be infected
THEN I can see a green window with a smile face

2.3 User story about general function

Requirement: users want an easy way to sign-up and get the latest update information about Covid-19

Feature: Sign-up and Login

As a: student
So that: my account can be directly bound with university
I want to: use my campus email account to sign-up and login
GIVEN that I am at login and sign-up page
WHEN the university and email account I enter match
THEN I can see my account verified and bound with my university

---

Feature: Get access to brief information about Covid-19 situation in designated countries

As a: student
So that: I can know how the epidemic develops without searching.
I want to: see a brief information like number of infected people in selected countries
GIVEN that I am at home page
WHEN I set the country as China at setting page
THEN I can see number of infected, recovered and death people in China

---

Feature: Browse detailed information about Covid-19

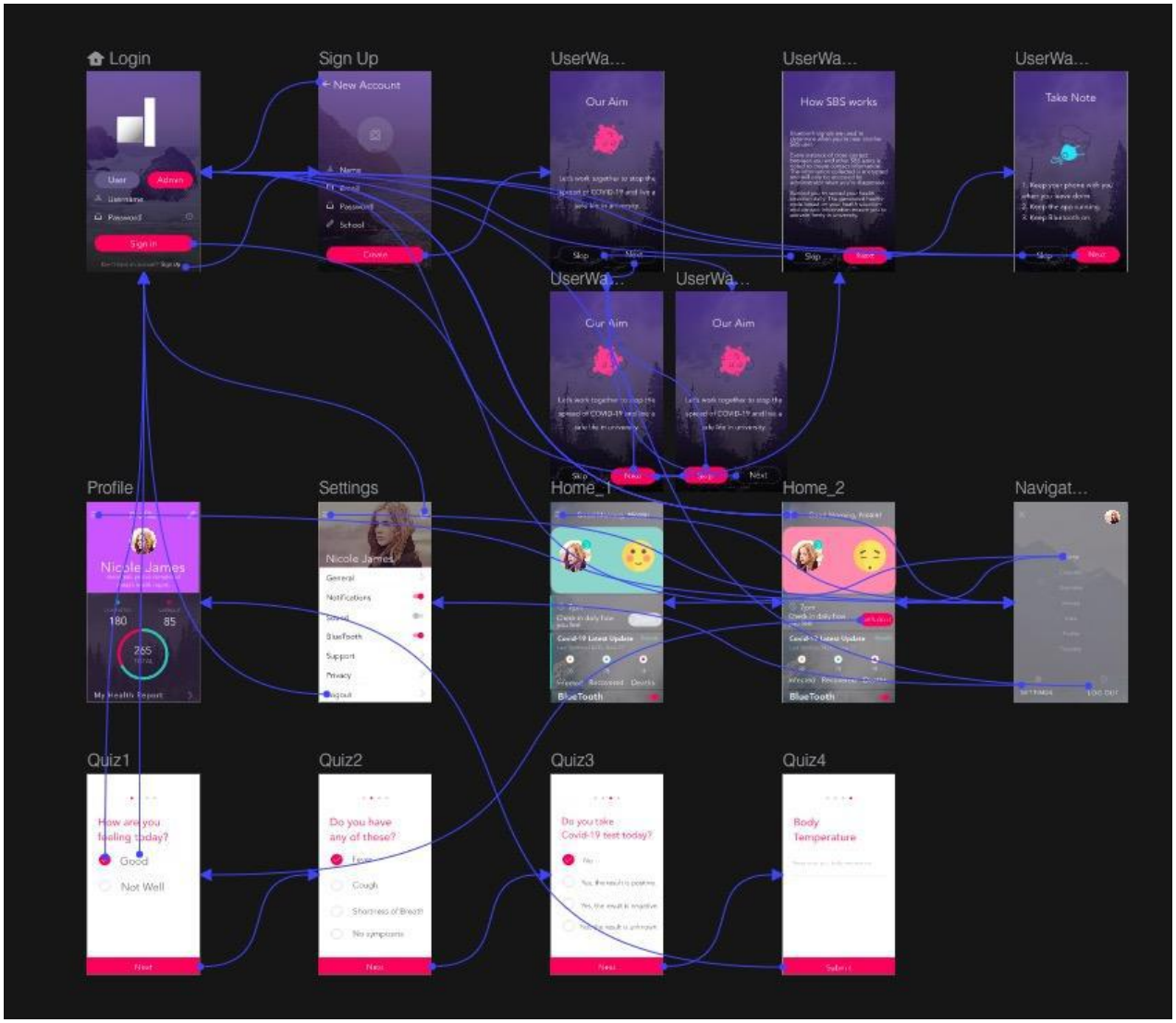
As a: student
So that: I can know more about Covid-19 situations all over the world
I want to: browse detailed information
GIVEN that I am at home page
WHEN I click 'more info' button
THEN I can see detailed Covid-19 situations

---

3 Interface Design

3.1 Original Interface Prototype with Invision Studio

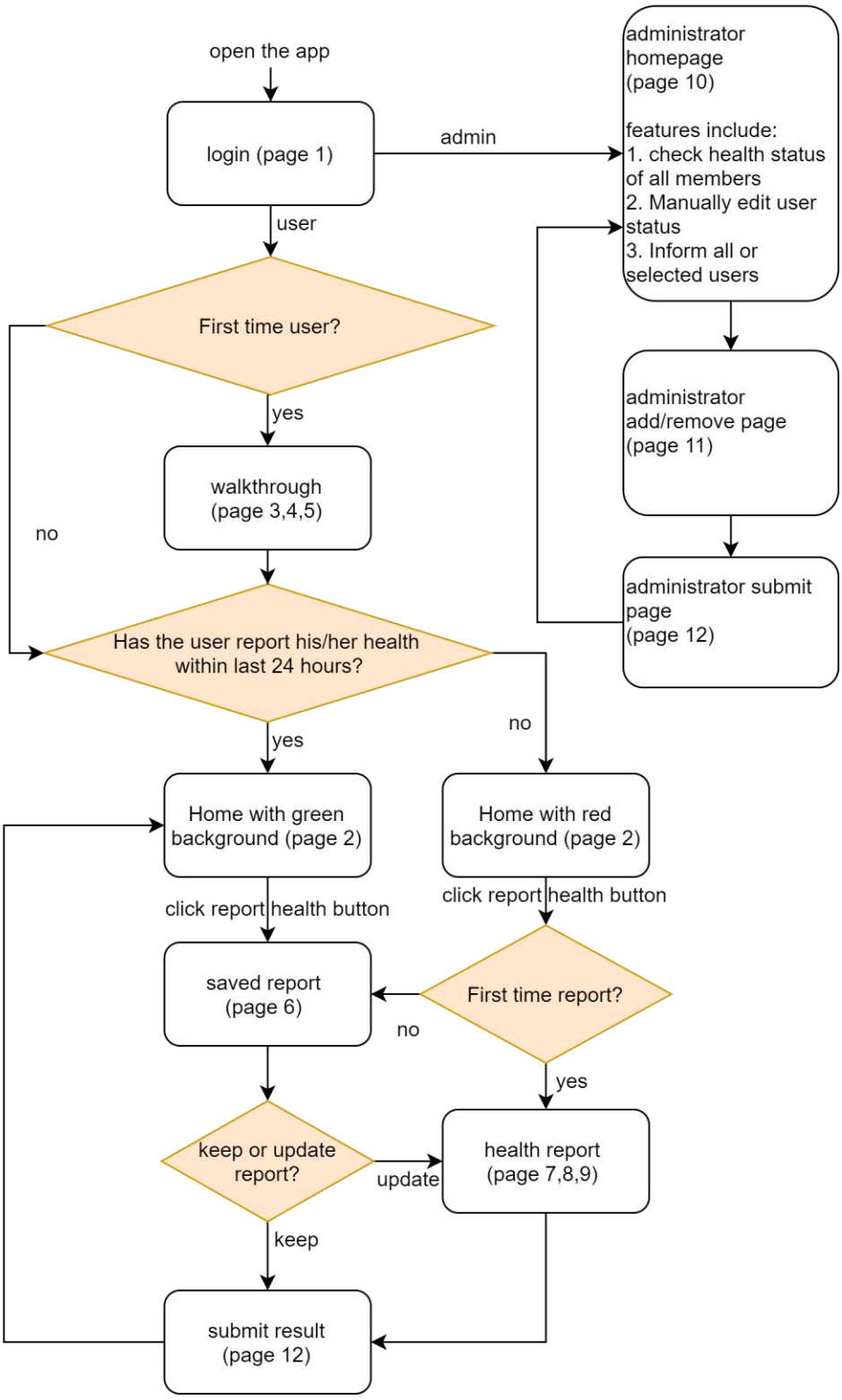
We first designed the prototype of the User Interface (UI) using Invision Studio. We used it because it allows us to quickly prototype our ideas without the need to write any actual codes. As described on its website, Invision studio combines design, prototyping, and collaboration into one harmonious workflow. Below is an overview of our initial design.

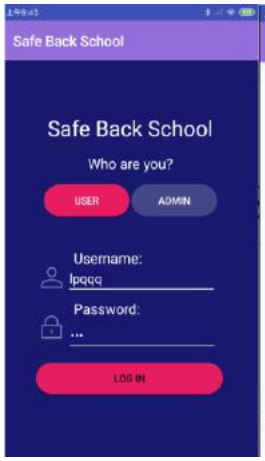


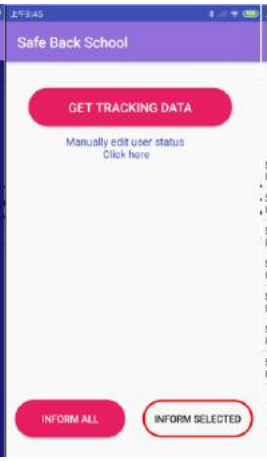
3.2 Interface Implementation in Android Studio

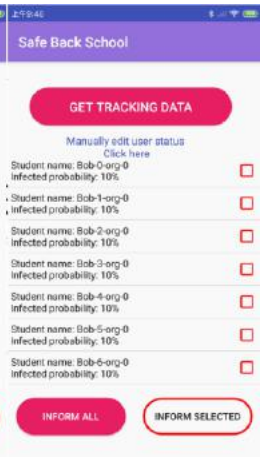
After prototyping UI in Invision Studio, we started to build the actual UI in Android Studio since we are making a native Android application. Below is a

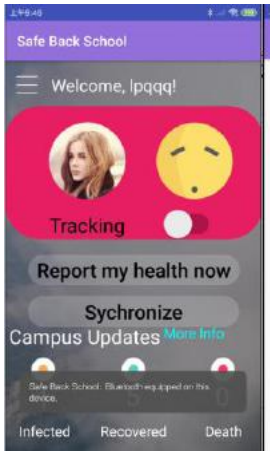
flowchart showing transition logics between different UIs (UIs with page numbers on top-right corners are shown in the next page).
(All page numbers here are noted from left to right, up to down)

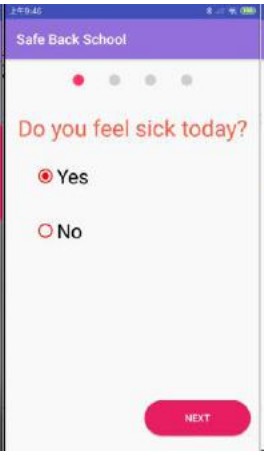


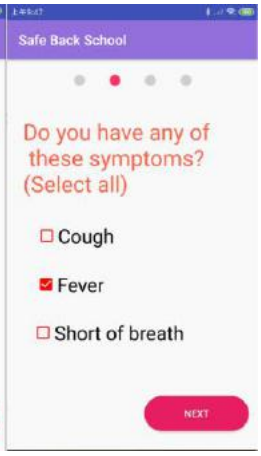

Safe Back School
Safe Back School
Who are you?
USER
ADMIN
Username:
lpqqq
Password:
...
LOG IN

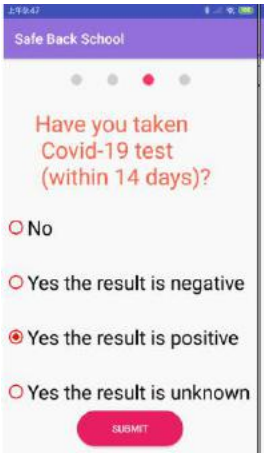

Safe Back School
GET TRACKING DATA
Manually edit user status
Click here
INFORM ALL
INFORM SELECTED

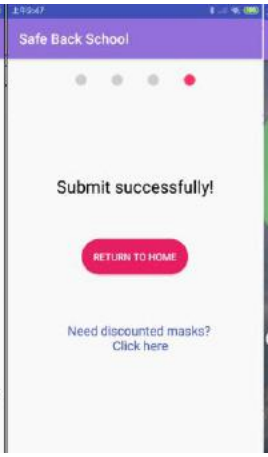

Safe Back School
GET TRACKING DATA
Manually edit user status
Click here
Student name: Bob-0-org-0
Infected probability: 10%
Student name: Bob-1-org-0
Infected probability: 10%
Student name: Bob-2-org-0
Infected probability: 10%
Student name: Bob-3-org-0
Infected probability: 10%
Student name: Bob-4-org-0
Infected probability: 10%
Student name: Bob-5-org-0
Infected probability: 10%
Student name: Bob-6-org-0
Infected probability: 10%
INFORM ALL
INFORM SELECTED

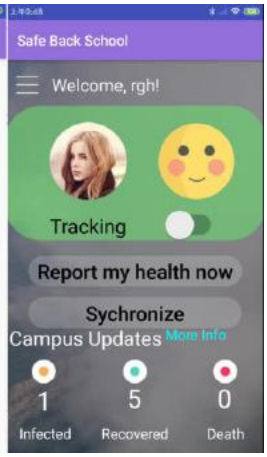

Safe Back School
Welcome, lpqqq!
Tracking
Report my health now
Sychronize
Campus Updates More Info
Infected
Recovered
Death

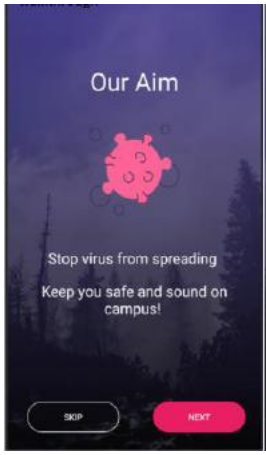

Safe Back School
Do you feel sick today?
Yes
No
NEXT


Safe Back School
Do you have any of these symptoms? (Select all)
Cough
Fever
Short of breath
NEXT

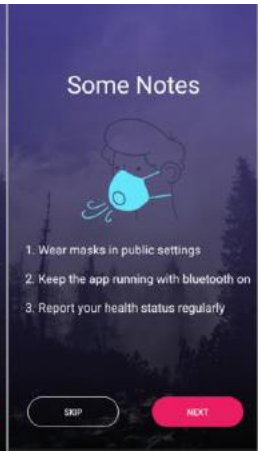

Safe Back School
Have you taken Covid-19 test (within 14 days)?
No
Yes the result is negative
Yes the result is positive
Yes the result is unknown
SUBMIT


Safe Back School
Submit successfully!
RETURN TO HOME
Need discounted masks?
Click here

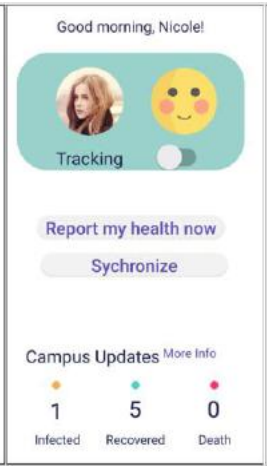

Safe Back School
Welcome, rgh!
Tracking
Report my health now
Sychronize
Campus Updates More Info
1
5
0
Infected
Recovered
Death


Our Aim
Stop virus from spreading
Keep you safe and sound on campus!
SKIP
NEXT


How SBS Works
Bluetooth signals are used to determine when you're near another SBS user.
Every instance of close contact between you and other SBS users is noted to create contact information. The information collected is encrypted and will only be accessed by administrator when you're diagnosed.
Remind you to record your health situation daily. The generated health code based on your health situation and contact information ensure you to activate freely in university.
SKIP
NEXT


Some Notes
1. Wear masks in public settings
2. Keep the app running with bluetooth on
3. Report your health status regularly
SKIP
NEXT


Safe Back School
Who are you?
USER
ADMIN
Username:
Password:
LOG IN


Good morning, Nicole!
Tracking
Report my health now
Sychronize
Campus Updates More Info
1
5
0
Infected
Recovered
Death

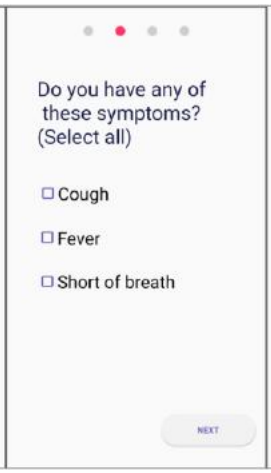

Our Aim
Stop virus from spreading
Keep you safe and sound on campus!
SKIP
NEXT

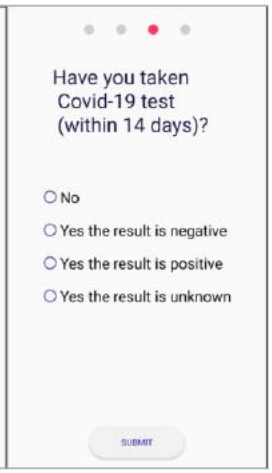

How SBS Works
Bluetooth signals are used to determine when you're near another SBS user.
Every instance of close contact between you and other SBS users is noted to create contact information. The information collected is encrypted and will only be accessed by administrator when you're diagnosed.
Remind you to record your health situation daily. The generated health code based on your health situation and contact information ensure you to activate freely in university.
SKIP
NEXT

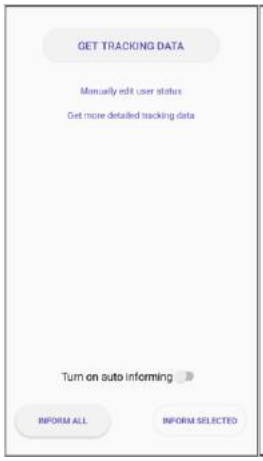

Some Notes
1. Wear masks in public settings
2. Keep the app running with bluetooth on
3. Report your health status regularly
SKIP
NEXT


Update your health?
TextView
SKIP
UPDATE


Do you feel sick today?
Yes
No
NEXT


Do you have any of these symptoms? (Select all)
Cough
Fever
Short of breath
NEXT


Have you taken Covid-19 test (within 14 days)?
No
Yes the result is negative
Yes the result is positive
Yes the result is unknown
SUBMIT


GET TRACKING DATA
Manually edit user status
Get more detailed tracking data
Turn on auto informing
INFORM ALL
INFORM SELECTED


Add/remove from infection list
choose an option below
ADD
REMOVE
Type the username here
SUBMIT


Submit successfully!
RETURN TO HOME
Need discounted masks?
Click here

Part 2. Design and Implementation Details

1. Final software architecture

In this section, the software architecture will be introduced. To better explain the software architecture, the C4 model was used to visualizing software architecture, which is an "abstraction-first" approach to illustrate software architecture.

First, the system context diagram is to be introduced. the system context diagram is shown in figure 1. According to this diagram, the big picture of the system can be seen, and it can help the developer to step back to review the whole system. There are two types of users in the proposed system: The Administrator and the university students. In this part, procedures like register and sign-in will be ignored, and the analysis will concentrate on the relationship between the system and the users.

As can be found in **Figure 1**, the university students will send the Bluetooth UUIDs around them to the server, and the server will send the list of students who could be infected, with the possibility, to the Administrator. And only after the Administrator tells the server which students should be informed, then those students will receive the information about their health state. Because we need to gather the Bluetooth UUIDs around the user, the application cannot be implemented with web APP, because the web APP cannot get the Bluetooth information of the hardware, android phone.

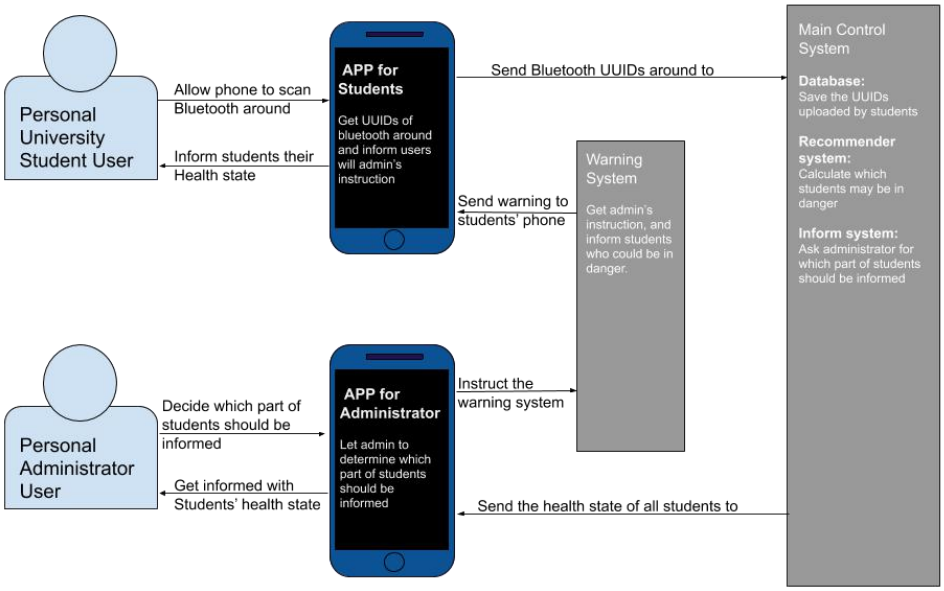


Figure 1. The system context diagram for software architecture.

Other third-party components such as web browser will be used. A link on the home page of the app will direct the user to the website that can inform them of the latest trends of coronavirus in that university's region. We will also add links to the webpage of local CDC or university guidance on preventing Covid-19.

Second, the container diagram is going to be explained **(Figure 2)**. It shows the high-level shape of the software architecture, and by looking into it the responsibilities of every component can be understood. There are 5 most significant containers in the container diagram: The API server, the warning system, the Database, the recommender system, and the Inform system.

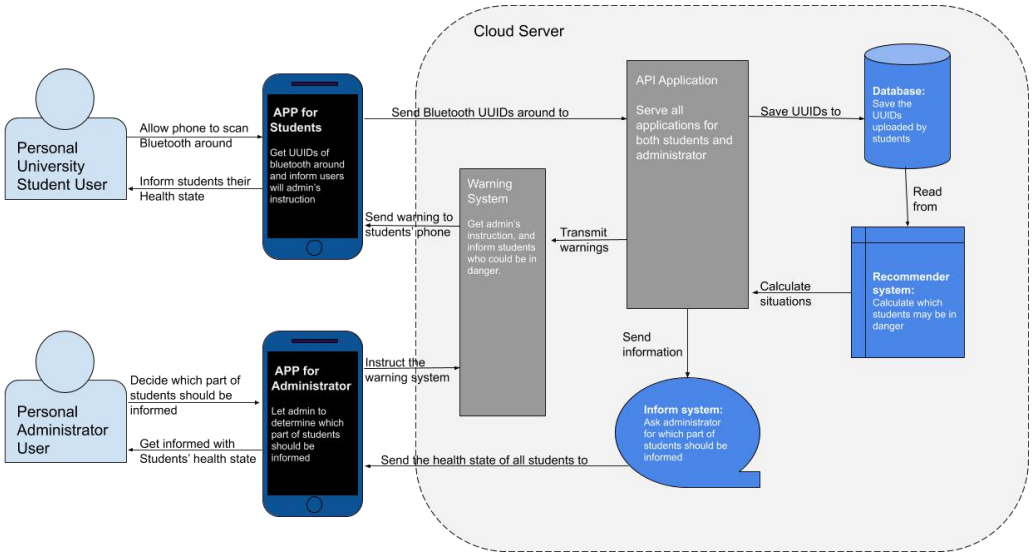


Figure 2. The container diagram for software architecture.

The API server is the bond of the Android APP and the backend; the warning system can inform the administrator which students could be infected and ask for the permission of notify; the Database saves all the UUIDs that the user collected; the recommender system use the data in the database to calculate who could be infected, and the Inform system can inform the students who could be infected.

Most importantly, let's look at the Recommender system. In this system, pipeline and filter are used as the design pattern. The data is delivered from the database to the filter, the recommender system, then whether the user has been in the same place with a possibly infected student will be calculated. If so, according to the time, distance (which can be estimated by the strength of the Signal), the probability of that student being affected can be estimated.

Third, the component diagram is shown in **Figure 3**, which illustrates the responsibilities of each components. Here Flask is chosen as the API server, and apart from the API component, there are three main parts of components: one is for sign-in, one is for Bluetooth information, and one is for warning. The sign-in

controller is connected with the security components, and it can help users to register one account and change the password if needed. The Bluetooth controller can save the Bluetooth UUIDs to the database and serve the infection information to the API server, and the warning system can inform the students who passed the students' selection components with the administrator's instruction.

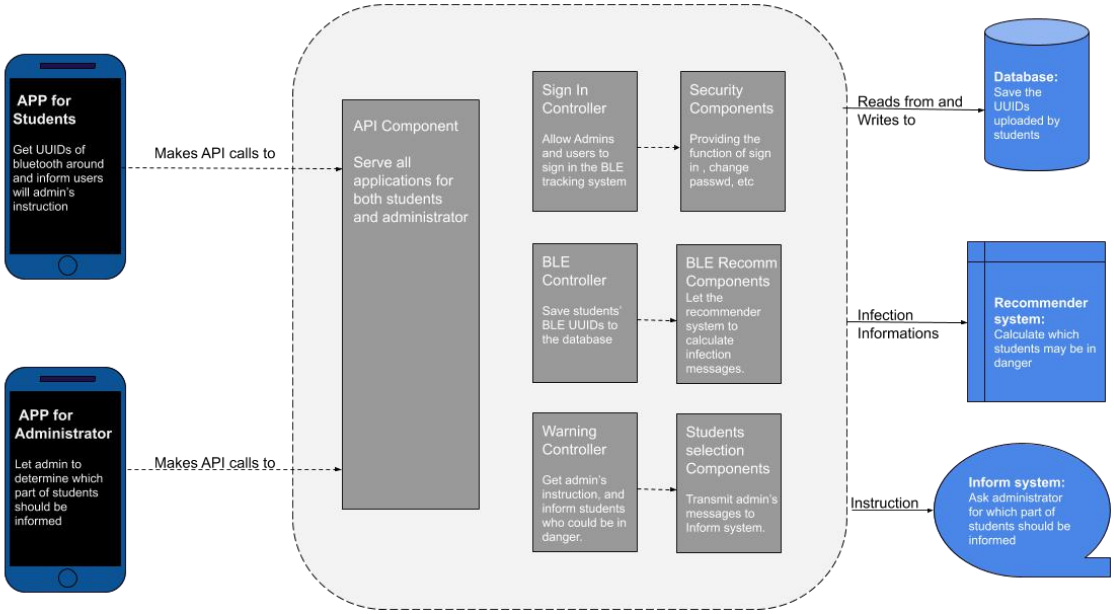


Figure 3. The component diagram for software architecture.

There are many advantages to the separation of components. Because these components are parallel can will not affect each other, the whole system will not break when one error happens in one component. And the API component can control all the three controllers, so when one controller is broken, the API server can weak it up soon. Besides, the performance of those controllers is not the same, so break them apart and implement them separately can help to improve the overall performance by allowing the significant part, the Bluetooth controller, to use more resources. Finally, because it's hard to come up with a perfect recommender system, the Recommender system needs to be updated many times. Because the recommender system is not embedded inside any components, it can be updated easily

2. Sequence and Interaction Diagram

The app is divided into two parts - one for the general users and the other for the administrators. The following part of the report contains the complete use case for both cohorts.

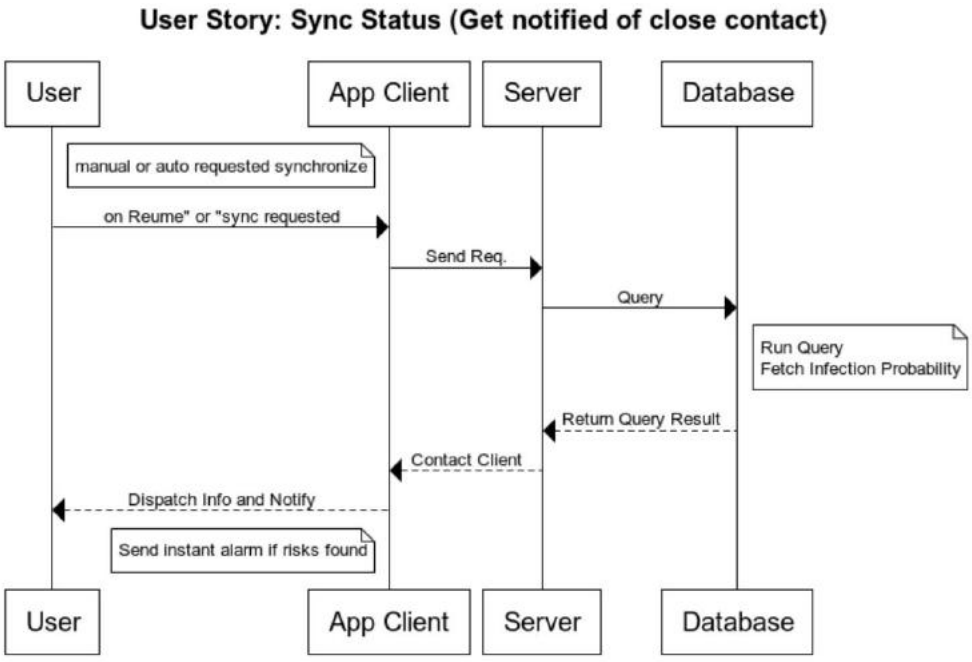
For the users, it is required that the app can achieve:

- Notify the user if they have been exposed to any close contact with someone postive in Covid-19 test. (U1)
- Remind and enable user to upload their latest health status. (U2)
- Periodically detect other mobile devices with the same app installed when the bluetooth is on. (U3)

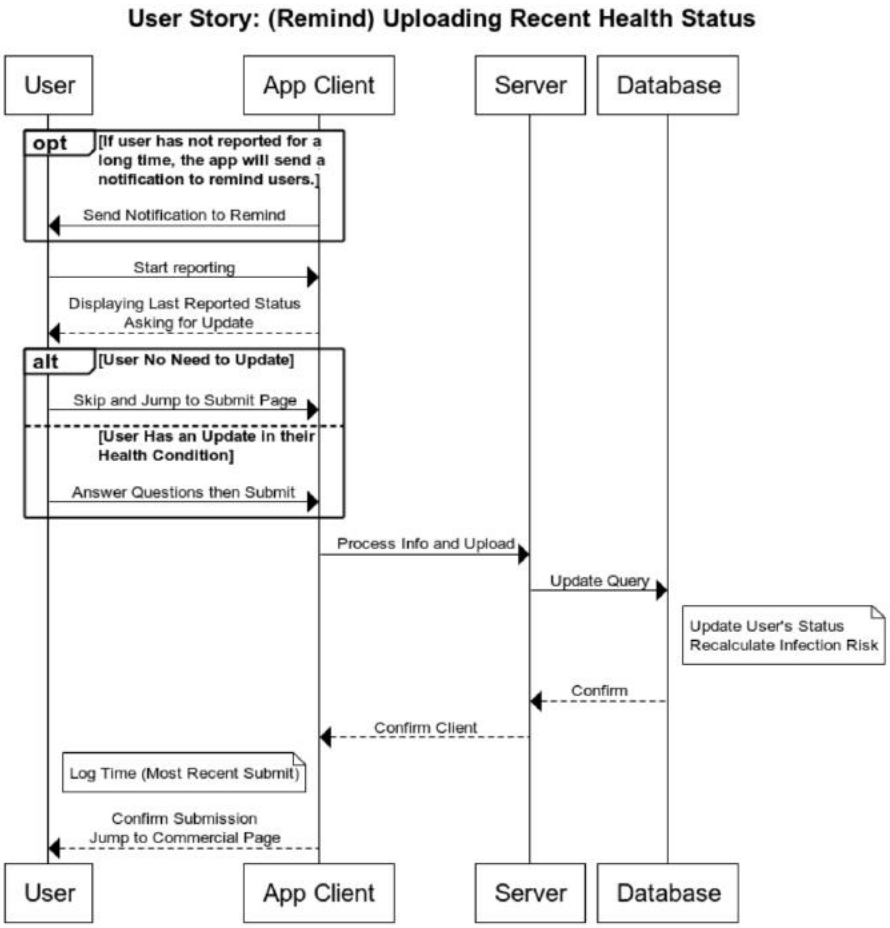
For the administrators, the app is expected to handle the following tasks:

- Get a list of currently infected and potentially infected students and notify them of the risk of necessary. (A1)
- Synchronize the database with official government / hospital database (simulated one in project demo) to ensure greater authenticity. (May be varied based on actual third party implementation.)
- Manually update (add / remove people in and out of) the list of infection to make instant change and send alarm immediately. (A2)

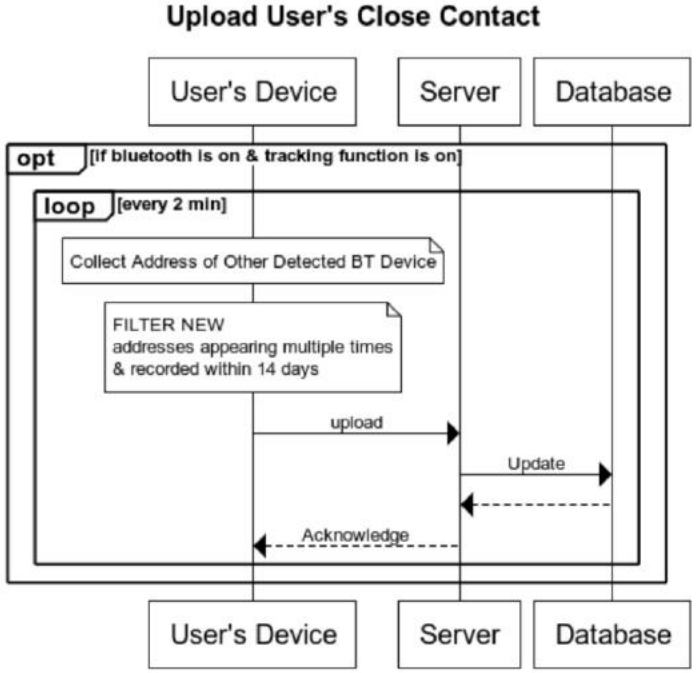
U1:



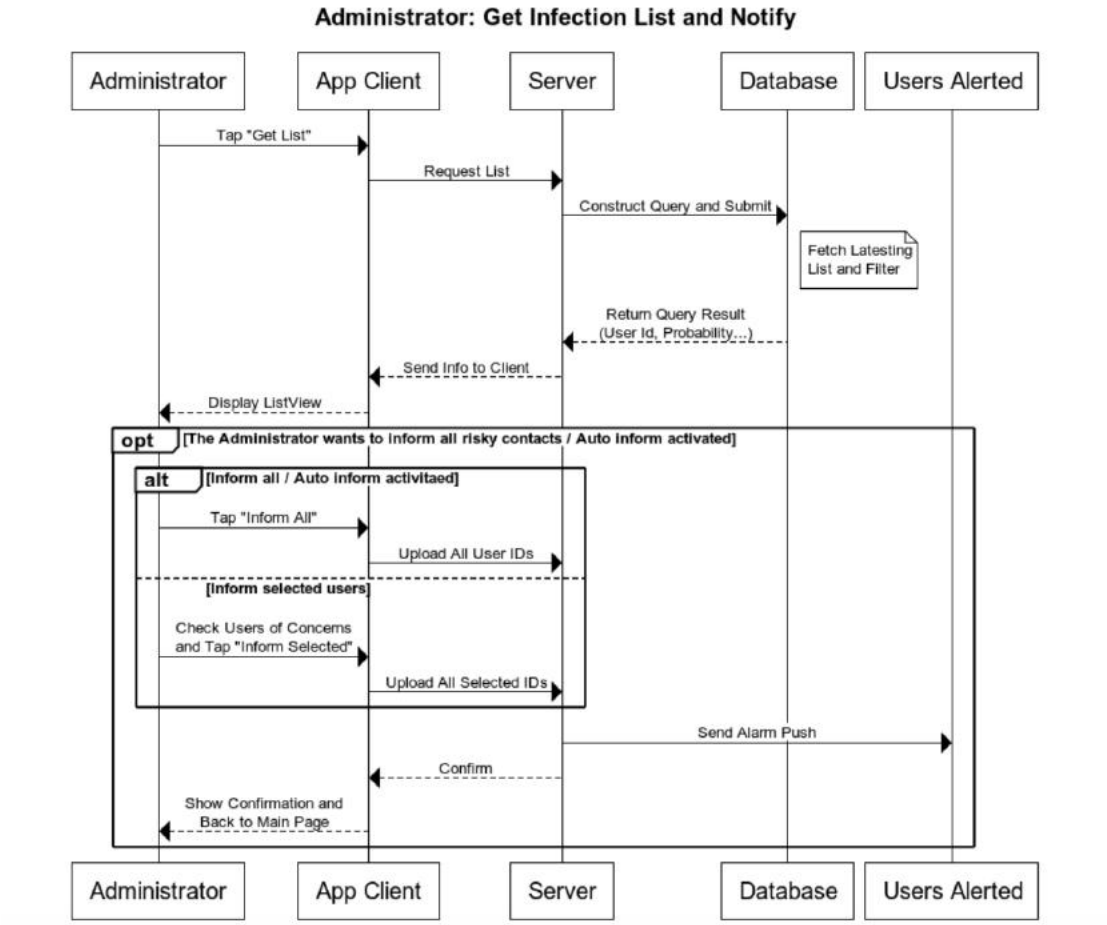
U2:


User Story: (Remind) Uploading Recent Health Status
User
App Client
Server
Database
opt
[If user has not reported for a long time, the app will send a notification to remind users.]
Send Notification to Remind
Start reporting
Displaying Last Reported Status
Asking for Update
alt
[User No Need to Update]
Skip and Jump to Submit Page
[User Has an Update in their Health Condition]
Answer Questions then Submit
Process Info and Upload
Update Query
Update User's Status
Recalculate Infection Risk
Confirm
Confirm Client
Log Time (Most Recent Submit)
Confirm Submission
Jump to Commercial Page
User
App Client
Server
Database

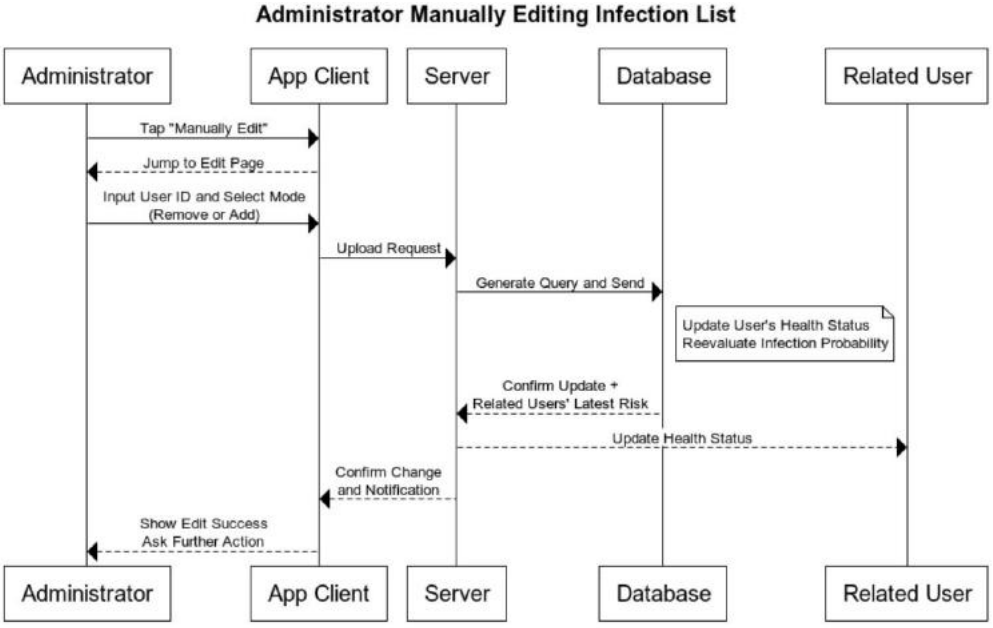
U3:


Upload User's Close Contact
User's Device
Server
Database
opt
[if bluetooth is on & tracking function is on]
loop
[every 2 min]
Collect Address of Other Detected BT Device
FILTER NEW
addresses appearing multiple times
& recorded within 14 days
upload
Update
Acknowledge
User's Device
Server
Database

A1:


Administrator: Get Infection List and Notify
Administrator
App Client
Server
Database
Users Alerted
Tap "Get List"
Request List
Construct Query and Submit
Fetch Latesting List and Filter
Return Query Result (User Id, Probability,..)
Send Info to Client
Display ListView
opt
[The Administrator wants to inform all risky contacts / Auto inform activated]
alt
[Inform all / Auto inform activitaed]
Tap "Inform All"
Upload All User IDs
[Inform selected users]
Check Users of Concerns and Tap "Inform Selected"
Upload All Selected IDs
Send Alarm Push
Confirm
Show Confirmation and Back to Main Page
Administrator
App Client
Server
Database
Users Alerted

A2:


Administrator Manually Editing Infection List
Administrator
App Client
Server
Database
Related User
Tap "Manually Edit"
Jump to Edit Page
Input User ID and Select Mode (Remove or Add)
Upload Request
Generate Query and Send
Update User's Health Status Reevaluate Infection Probability
Confirm Update + Related Users' Latest Risk
Update Health Status
Confirm Change and Notification
Show Edit Success Ask Further Action
Administrator
App Client
Server
Database
Related User

3. Key technologies used in our implementation

As summarized in **Figure 4**, we used Android Studio to develop our frontend, Flask to develop the backend server and MySQL to develop the DataBase. Okhttp3 is used by the client side to send http requests to the server. A recommender system written in python will help the server to decide which info to be collected from the database.

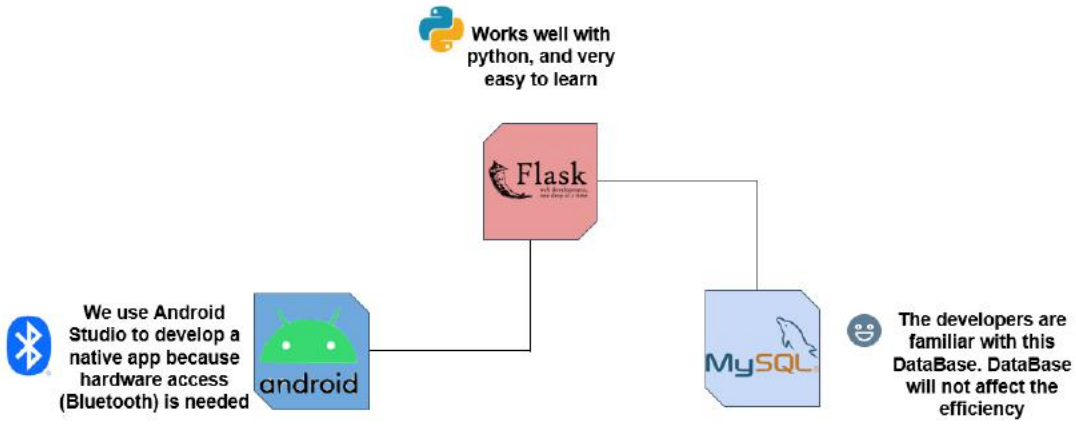


Figure 4. Key technologies used in front/back end of our APP

In Android Studio, java is used to define the internal logic for each activity (UI pages seen by the user) and XML is used to design the UI for each page (see Part 1 for UI design screenshots). We chose to use Android Studio because it's one of the most widely used software for Android APP development and is user friendly. Also, building a native app allows us to use Bluetooth for user tracking, which is an important part in our app design. The user info, user health status, Bluetooth tracking results and infection probability are sent/received using okhttp3 HTTP client.

Flask is a micro web framework written is Python. We use it to write our server which receives requests sent from the frontend. With the help of Flask, we can easily handle http requests such as GET, POST and PUT. Once it collects the data needed, the server will communicate with the DataBase to put/get further info according to different requests.

The recommender system is another essential part of our backend. It's an intelligent AI system that will utilize the Bluetooth tracking data and users' health status from the DataBase to decide the infection probability of a specific user. A person's health status is composed of the person's own symptoms and the health status of his or her contacted person **(Figure 5)**. Personal health symptoms include fever, cough, and other symptoms. The influence of the surrounding people that this

person has touched is the most complicated part of the system, and this part is assisted by Bluetooth scanning.

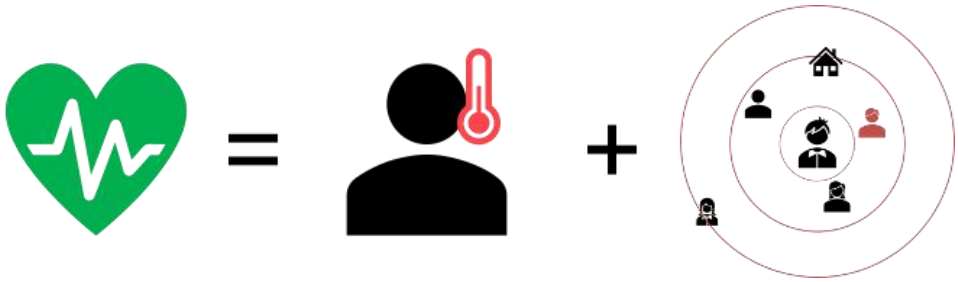


Figure 5. health status is composed of the symptoms and the health status of contacted person

The detection of possible infected persons in the surrounding area consists of two parts: the user directly scans the Bluetooth of another person, and two users scan the same public Bluetooth **(Figure 6).** If the time between the user and the potentially infected person exceeds the threshold, the user's probability of infection will increase. The existence of public Bluetooth can also help schools obtain the location information of students, so as to avoid cross-infection in public more effectively. Besides, the location information is anonymous, so service providers cannot know this information.

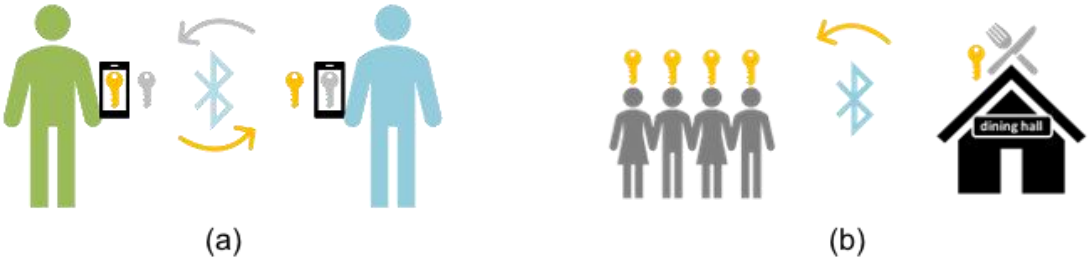


Figure 6. the user directly scans the Bluetooth of another person (a), and two users scan the same public Bluetooth (b).

MyWebSQL is used to manage MySQL DataBase system over the web. All the user related data are securely saved in our database. Currently we have 6 tables in our database which includes Admins, Organizations (in case multiple schools are using our app), infectionDetector (records infection probability of each user), scannedBLE (Bluetooth addresses scanned by each user's mobile phone), userBLE (Bluetooth address of the user's mobile phone) and users **(Figure 7). Figure 8** is a screenshot of Bluetooth addresses scanning info. **Figure 9** is a screenshot of users and their corresponding infection probabilities (testing data, not the real data).

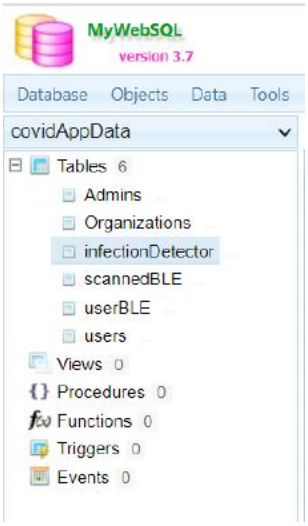


Figure 7. Tables in the Database

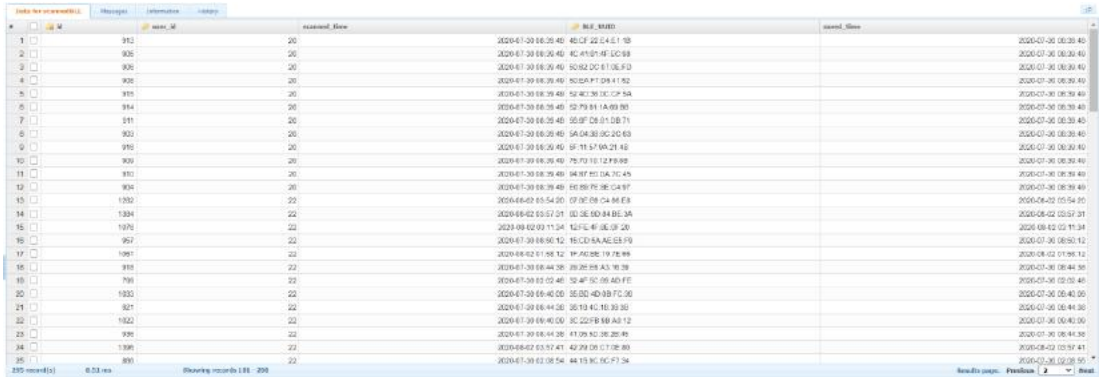
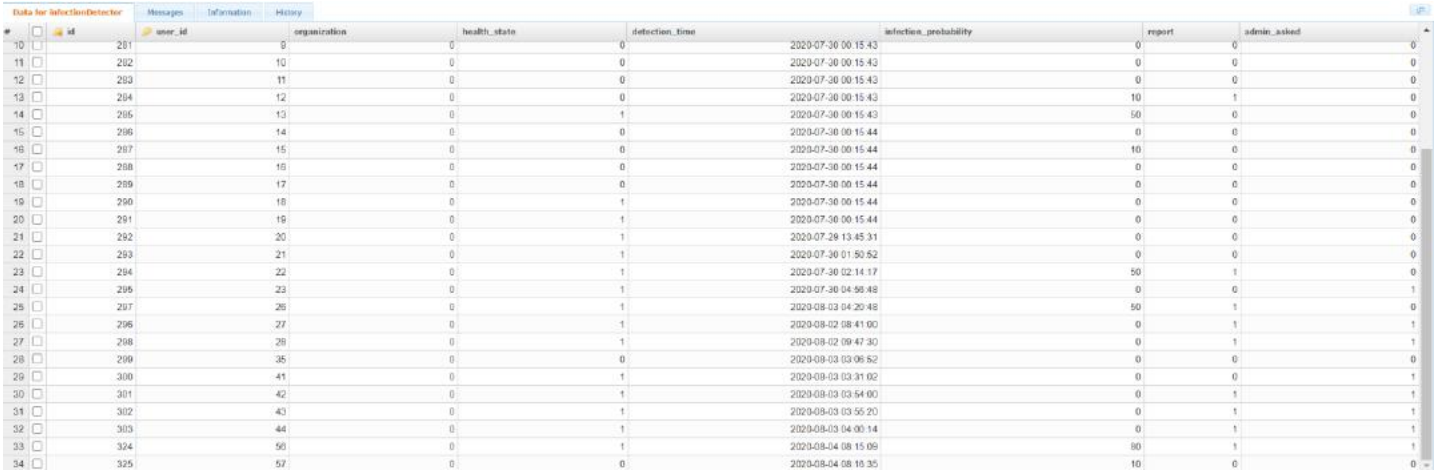


Figure 8. Sample Bluetooth scanning info



	id	user_id	organization	health_state	detection_time	infection_probability	report	admin_asked
10	281	9	0	0	2020-07-30 00:15:43	0	0	0
11	282	10	0	0	2020-07-30 00:15:43	0	0	0
12	283	11	0	0	2020-07-30 00:15:43	0	0	0
13	284	12	0	0	2020-07-30 00:15:43	10	1	0
14	285	13	0	1	2020-07-30 00:15:43	50	0	0
15	286	14	0	0	2020-07-30 00:15:44	0	0	0
16	287	15	0	0	2020-07-30 00:15:44	10	0	0
17	288	16	0	0	2020-07-30 00:15:44	0	0	0
18	289	17	0	0	2020-07-30 00:15:44	0	0	0
19	290	18	0	1	2020-07-30 00:15:44	0	0	0
20	291	19	0	1	2020-07-30 00:15:44	0	0	0
21	292	20	0	1	2020-07-29 13:45:31	0	0	0
22	293	21	0	1	2020-07-30 01:50:52	0	0	0
23	294	22	0	1	2020-07-30 02:14:17	50	1	0
24	295	23	0	1	2020-07-30 04:56:48	0	0	1
25	297	26	0	1	2020-08-03 04:20:48	50	1	0
26	296	27	0	1	2020-08-02 08:41:00	0	1	1
27	298	28	0	1	2020-08-02 09:47:30	0	1	1
28	299	35	0	0	2020-08-03 03:06:52	0	0	0
29	300	41	0	1	2020-08-03 03:31:02	0	0	1
30	301	42	0	1	2020-08-03 03:54:00	0	1	1
31	302	43	0	1	2020-08-03 03:55:20	0	1	1
32	303	44	0	1	2020-08-03 04:00:14	0	1	1
33	324	56	0	1	2020-08-04 08:15:09	80	1	1
34	325	57	0	0	2020-08-04 08:16:35	10	0	0

Figure 9. Sample infection probability data

4. **A summary of the key benefits and achievements**

Our core idea is to develop the app that is user friendly and offer users one more choice except lock-down policy under the epidemic situation. We select down customers to university since it's relatively easy for promotion and management as a start-up. The key achievements of our app can be summarized into four parts.

Firstly, Bluetooth contact tracking technology is implemented as the main function to prevent outbreak of coronavirus in university. The reason why we need to achieve contact tracing function is that it can save lives by helping us diagnose patients earlier, improving the likelihood they'll be cured and reducing the chance they'll spread the virus to others. As long as the user's Bluetooth is on, the app can record information of nearby device and contact duration. Contact tracing based on Bluetooth technology can replace the traditional contact tracking based on manual screening and paperwork because it's a simpler, quicker and more accurate way.

Secondly, the app offers school administrators a great platform to collect and visualize all students' health status. Students' health data are collected and updated automatically to the database. We design a clear user interface for administrator to know who is infected and the potential infected students. The infection probability calculated based on user's symptoms and contact time with infected person is also offered to administrator to make the decisions.

Thirdly, smart alert is also an important feature for our users. Once the user has contacted with an infected person, the app would automatically send the warning message to the user. This can help user to have a clear understanding and judgement about the surrounding environment.

Lastly, the design of questionnaire in daily health report aims to avoid the tedious and repetitive work. The form memorized the latest information user entered so that users don't need to waste time on filling the same content. They just need to modify the content when they get new symptoms. Such design matches our core idea and increases user viscosity to some extent.

Part 3. Team Organization and Appraisal

1. Responsibilities:

Daiyang Li:
- Initiate and schedule group meetings
- Design the logic of client App
- Optimize Bluetooth module and its performance

Bingcheng Hu:
- Design the architecture of the server side.
- Implement the recommender system, the database construction and the bridge between them.
- Build the cloud server with TenCent Cloud.

Yuanjie Tao:
- Design questionnaires and make customer interviews.
- Business and marketing analysis.
- Implement a basic connection between frontend and backend.

Ruoxin Geng:
- Design the user interface for the app using Sketch
- Add interactions between different pages and generate workflow chart using InVision Studio
- Make a showcase video for the app

Pengqi Lu:
- Implement the UI using Android Studio on the client side.
- Implement the APP logic transitions on the client side
- Set up http communication on the client side

2. Difficulties and Challange:

- That trans-pacific team must work against time difference
- Steep learning curve for members with limited prior knowledge about app dev.
- Reduced efficiency in communication with mentors in remote mode
- In order to achieve high-efficiency SQL query, various different queries were tested, and finally the fastest query method was found.
- In order to implement an instant recommendation system, the background needs to be refreshed efficiently. Finally reached the frequency of refreshing every minute. When you continue to increase the frequency, the server will respond slowly to the app.
- Because the server is in Shanghai and the developers are in the United States, the server delay is relatively long.
- The message from the server may take a long time to transmit to the client side (since the server is in China, and we are testing the client in the US).
- The user interface may not be compatible with mobile phones with different screen size

3. Overall appraisal:

Despite the aforementioned challanges and difficulties, in this course we managed to achieve:

- Clearly define a business problem
- Justify why our app helps solve the problem
- Implement an MVP of the app through joint effort
- An understanding of agile method in development

4. What I think could be done differently:

- Meet with business mentors more frequently to constantly make mild adjustments
- Communiate within the team in a more unified way to avoid overlap of work done.
- Work more closely with key partners and take their need into account in the app.
- We may use non-relational databases. Because our system does not require high real-time performance, using MySQL results in slower refresh speed.
- Will use a larger bandwidth server, because the current server bandwidth is only 1M per second, so that the user experience is not excellent.
- To make the UI design compatible with different mobile phones, we need to avoid hard-coded layout sizes and may try available layout width qualifiers.

Part 4. Code Inspection

https://github.com/boldwings/VE441_Undecided