COMP5216: 2021 Semester2
Project Report

FindMyGym

Team Name	Project Group 19
Team Members	Jiyue Peng
	Yiwei Wang
	Bohan Zhang
	Yiwei Xu
	Yizhen Xu
Date	25/10/2021

Context

1. Introduction

Many have said that 2020 and 2021 are the most different and difficult years. The COVID-19 dramatically changes everyone's daily life. Masks have been put on, transportation has been restricted and social distance has been ordered to be kept.

Before COVID, there have been reports about how crowded the gyms are during peak hours. The COVID-19 completely changed that(Davalos, 2021). In the most difficult time, the gyms are ordered to shut down. But the gym industry still has the market(ESCHNER, 2021). Even after the restrictions are eased, social distance still needs to be kept, which limits the total number of people in the gym(My Gym Is Reopening. Is It Safe To Work Out There?, 2020). This brings up an opportunity, how about an app that shows all available gyms and requires the user to book a gym session and personal trainer instead of directly walking in(Ang, 2020)?

There are some problems with Sydney's current gym system. To change this situation, we have developed an Android App called FindMyGym. The approaches within the application are shown as below:

1. The gym owners are usually unaware of the current traffic of their gyms if without using any third-party tools. With our app, the gym owners can have a clear understanding of how many people will be in their gym for a particular time. During the COVID period, It can avoid awkward situations for customers being rejected on the spot when walking in as the gym has reached its capacity. If the restriction is tightened, the gym owners can update information about the gym to further limit the capacity of the gym. If all the restrictions are lifted, the gym owner can still have a great visualisation of their gym performance by viewing the number of gym sessions.

2. In general, the availability and the information of personal trainers of Sydney's gyms are only available in physical stores. Also, to know about the one-off entry fee of a particular gym is normally obtained by a phone call. This brings a problem to the casual gymers who are not willing to commit to a membership or constantly travelling and are not familiar with local gyms. With our app, all gymers will be able to view the gym and personal trainers details in one click.

3. Do you feel about the hassle of finding your desired gym? A gymer might also want to switch to another gym with much better reviews than the local ones. Imagine you are a serious gymer arriving in a new town and craving to go to the gym but are unfamiliar with the locals. Our app can help you find the gyms you desire, either by searching their names directly or finding out the closest gym to you. You can view all information about the gym in one single click and be able to save your favourite gym in our application.

2. Implementation

2.1 Summary

The application consists of 6 fragments and the main activity. The flow table of the application can be seen in Figure 2-1.

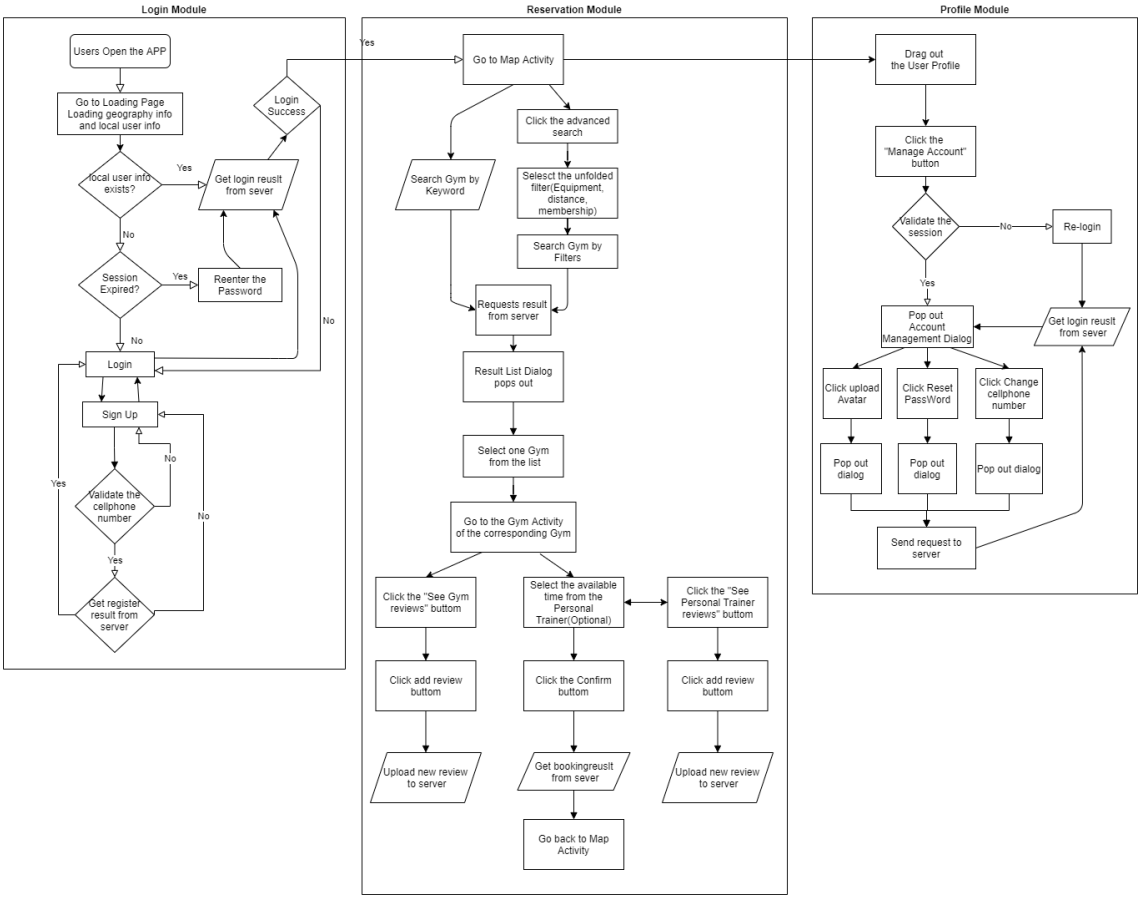


Fig.2-1

2.2 Modules

2.2.1 SignIn

In the SignIn module, we use Firebase to authenticate users. Users are only allowed to register through Google accounts as we believe the target user, which most of them should be under 30, should have at least one Google account. Therefore, it will reduce much time and energy spent on user authentication in the application flow.

In other words, our application's scale actually decides that we'd better not maintain a user database on our own. Using this method to login could provide high security and also could let our group focus more on the feature development.

The login activity can be seen in Fig.2-2 and once logged in successfully, it will pop out a toast to indicate that just like Fig.2-3.

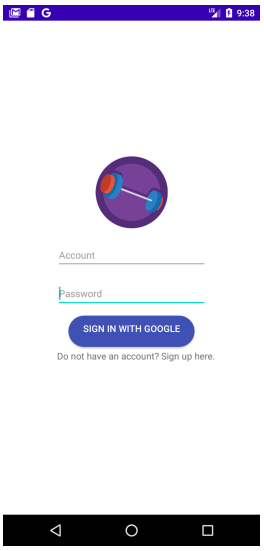


Fig.2-2

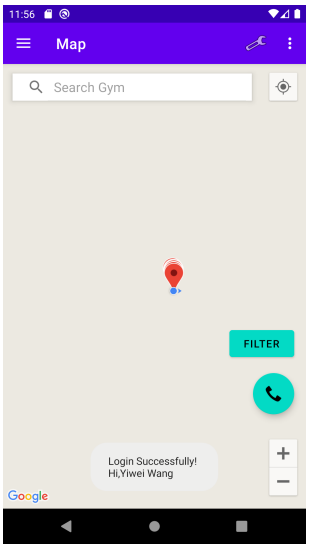


Fig.2-3

2.2.2 Map

The map fragment consists of the content displayed right after the login activity. It will firstly check for permissions to locations. If the location permission is not granted, it would pop up a dialog asking for permissions.

We referenced google map's API to implement the mapview. The real-time location should be read from the user's device. At the same time, the application will read the simple gym data from the firebase. After that, the map will display markers indicating the address of each gym. the user may single click the marker to see the name of the name and the address of it via an info window. The gym info page containing the full set of information can be accessed by long clicking the info window.

The users can search for their desired gym's name on the search bar and the map will take them to the gym's location. They can also click the filter button to filter gyms shown on the map. The filter function currently has two features, closest gym and favourite gym. The closest gym filter will calculate the distance between the user's current location and the gyms' locations to retrieve the closest gym and direct the camera to that location. The favourite gym function does a similar thing, which retrieves the user's favourite gym list and shows only the favourite gym on the map.

Regarding ease of use, we ensure that in the map activity, the user has the right to not grant location permission. If the user chooses not to grant permission, the app can still be used,

but functionality will be limited. The app will no longer show the user's current location and the closest gym functionality will be limited.

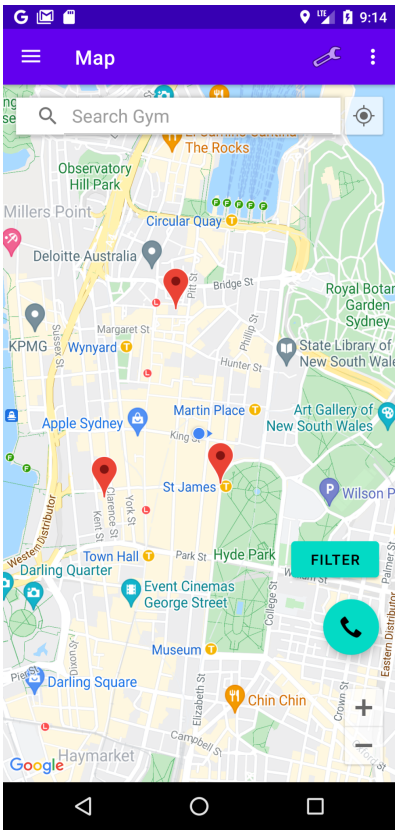


Fig.2-4

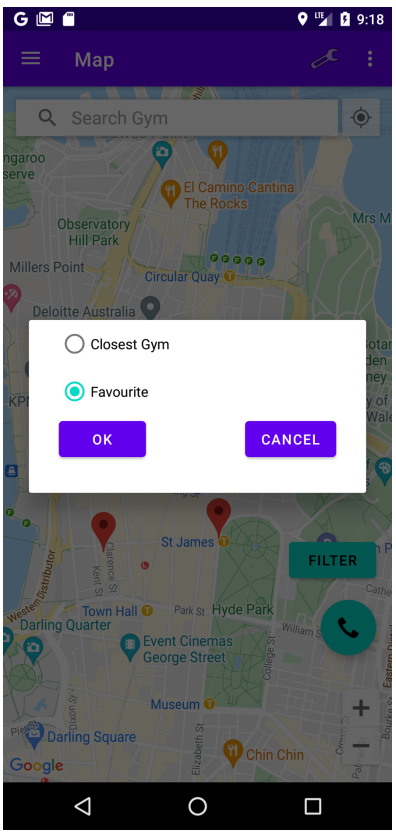


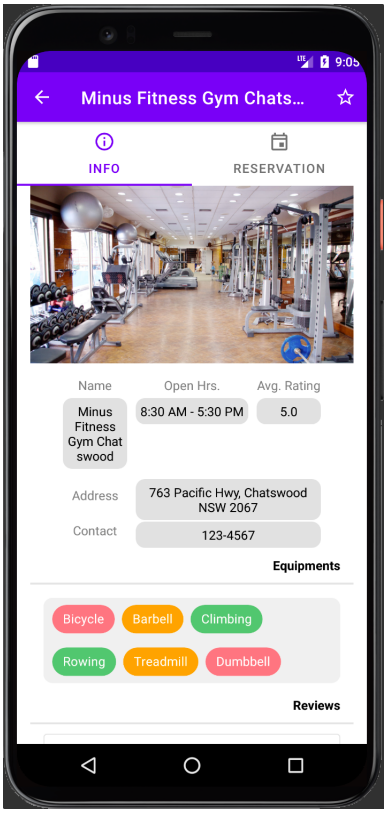
Fig.2-5

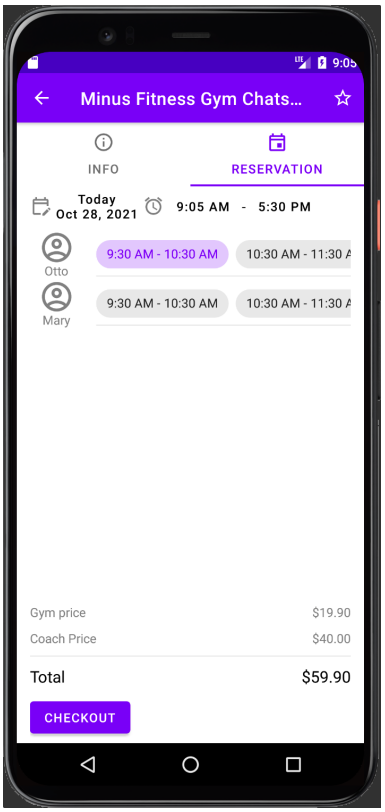
2.2.3 Gym

The Gym fragment is the most important part of this application and it contains our core features. After the user enters the GymActivity, there are two tabs at the top. The info tab contains the gym name, business hours, address and the contact method. There is an attractive feature which could display the equipment in this gym. For the user who has a specific requirement such as using what equipment to improve a part of the muscle, they could directly scan the equipment list on the page and look for the one they want to use.

Besides checking the information for a gym, the user could switch to the reservation tab to make a reservation process. In response to the current epidemic situation, the user could process an order in reservation and select a period in which the trainer is available in our application. Also, the user could book a gym without a trainer. The system will calculate the cost automatically and show it at the bottom of this page. After the user clicks the checkout button, the system will upload the data of the reservation to the database, and the booking information will show on the schedule page, the payment information will show in the wallet. This process can reduce people's contact with each other, users can complete a reservation without having to contact gyms and coaches, reducing the probability of

infection. At the same time, the gym can set an upper limit for the number of people in the data, so that a large number of gatherings will not occur in the gym within a certain period of time.





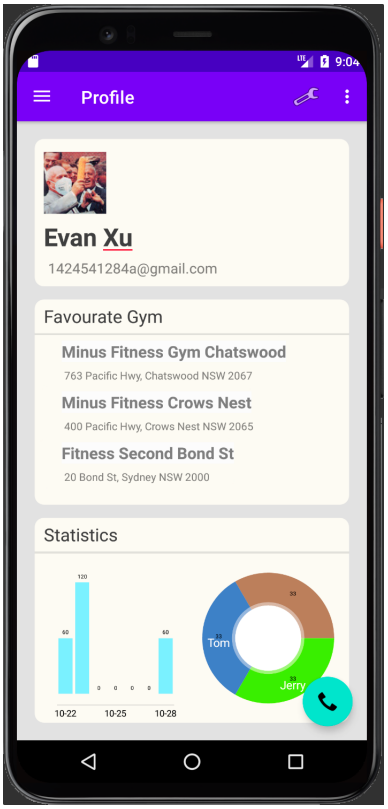


Fig.2-6 Fig.2-7 Fig.2-8

2.2.4 Profile

The profile fragment is a rather simple component compared to other fragments. This fragment consists of 3 different components, showing user information, favourite gyms and statistics. See in Fig.2-8.

The favourite gyms component is a RecyclerView which can be clicked to jump to the desired gyms. We are also using an observer to find out if the user data changes dynamically. If the user adds a gym to their favourite list, this component will add it dynamically instead of fetching it again from the database.

The Statistics part uses a third-party package named MPAndroidChart. We are calculating exercise time into a bar chart and the frequency of visiting recent trainers into a pie chart through retrieving reservations from the past.

2.2.5 Schedule

The Schedule fragment has two tabs which are Booking and History. The future reservation will display in the Booking tab, and the reservation which has passed will display in the History tab. The item in that list will contain the gym name, trainer name and the start time. First, we design an XML file to set the layout of the item. The item contains three textView for reservation information, and one image view to store the gym's preview image. Then we created an adapter to accept those items. In the "Schedule" activity, we also use the adapter to accept different tabs. The most different part is when we classify the "Booking" and "History" lists, we need to compare the calendar for now and the reservation time. Because of that, the Booking tab will show the upcoming reservation, and the History will show the passed reservation for the user.

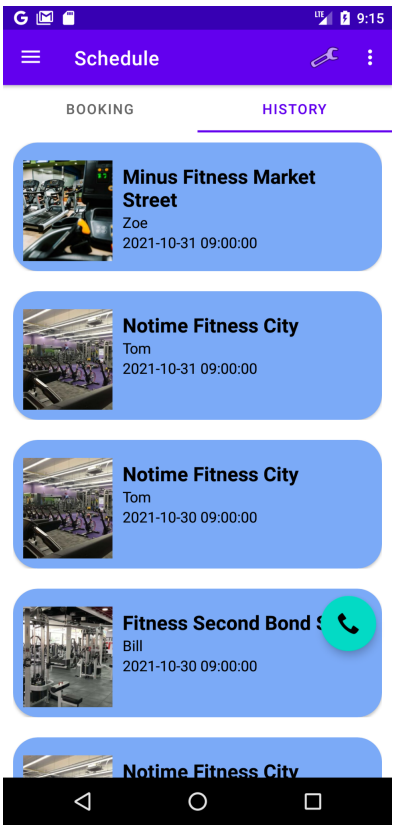


Fig-2.9

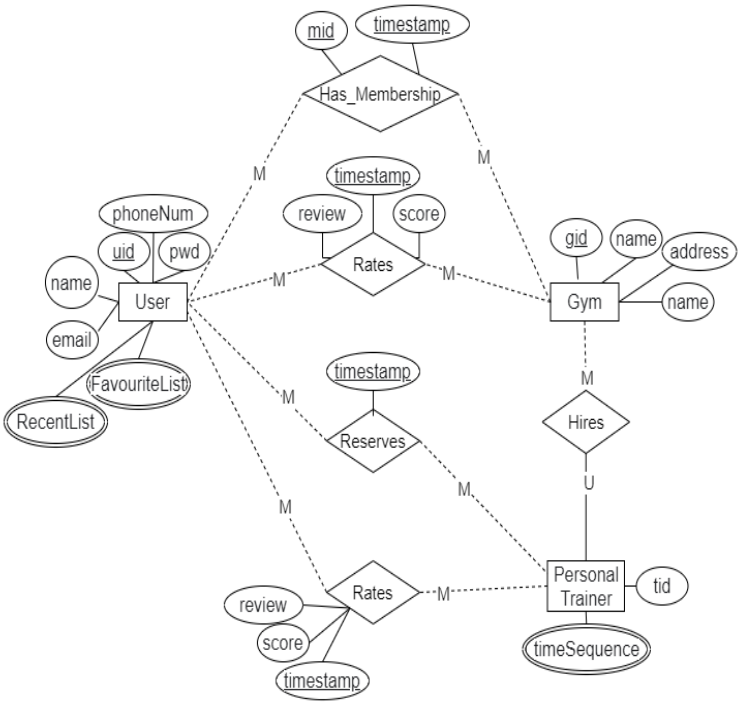


Fig.2-10

2.3 Backend

We use the Firestore database to store our user data and Firebase Storage to store our images. The data is divided into several tables so that we should not download all of the data every time. We can see the E-R diagram in Fig.2-10.

We met a small problem during development as we chose Firebase real-time database mistakenly. Since real-time databases are designed especially for instant messages, we are going through a hard time redesigning our data models and trying to find better query ways.

Now when a certain fragment is doing a query, we register a simple callback function to handle the request. Such simplicity has actually reduced much time in the whole query process and proved to be an effective way.

2.4 Optimization

As we know, it's important to consider the energy/bandwidth consumption during Android development. In this case, Several optimization techniques have been taken to reduce bandwidth and energy consumptions in our application. These techniques mainly include the following.

2.4.1 Online Databases

We divide all the information into several different tables and use firebase storage to get tables in need. This will be much better than just downloading everything every time we try to fetch that data. Another thing worth being mentioned is that we are using simple callbacks with Google Firebase Storage to get data whenever we want. The size of our database is 670kb. By only pulling the tables in need, we can save about 80% of the data consumed, which is about 530kb each time the app runs.

2.4.2 Image thumbnails

Most images are cached locally. If the application fails to read images, it will check the wifi condition and try to fetch them from Google Firebase Storage. Under the cellular network, the application will take bandwidth saving methods, which basically means downloading a 50% quality image. high-resolution images will be downloaded only when the wifi is on. Thus, we can actually save lots of bandwidth from downloading some real high-resolution pictures. Statistically, we are saving about 50K per image downloaded under the cellular network.

2.4.3 Observers

We use observers to see whether users change their data locally, which saves much bandwidth as there is no need to update to servers and fetch them again. It can be also considered as an energy-saving strategy as the download will take place less often than. A rough estimation upon profiler is that we can save about 10% of battery compared to observers not implemented.

2.5 Effectiveness of GUI

In our application, we have made sure our GUI provides as much information to the users and at the same time, to be effective. Below are a few implementations we have done.

2.5.1 Consistent style of elements

In terms of consistency, our implementation is to apply the same style to all adjacent elements. This makes them have the same colour and shape. If the users browse through all the tabs in our app, they will only see a consistent format without any special appearance.

2.5.2 No useless button

One of the examples under this implementation is the “reset” button in map activity. “reset” button is used to reset the markers shown on the map after the filtering function makes some of the markers disappear. Instead of having the “reset” button to be shown on the map all the time, we have made the “reset” only visible after the filter has been applied. This makes the “reset” button useful as it would not be a button floating around even if it doesn't have any functionality at that stage.

2.5.3 Easy navigation

Navigation is an important part of the usage of an app. We believe that the customer should not be confused about how to reach a certain page at any time. Users may click on the info window in map activity to reach the gym info page. They can go back to the map by clicking the back icon.

The sidebar can be clicked in map, profile, schedule and wallet pages and it includes access to these four pages. An example of the sidebar in our application can be found in Fig.2-11.

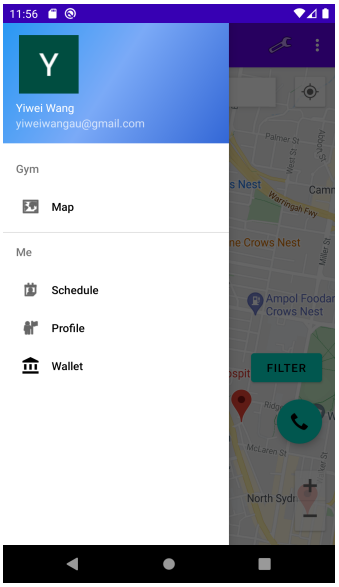


Fig.2-11

2.6 Third-Party Resources

We implement some third-party resources during development including the following libs.
For customized dialog:
https://github.com/F0RIS/sweet-alert-dialog
For floating action button in main activity:
https://github.com/algolia/voice-overlay-android/
For shimmery text in loading activity:
https://github.com/RomainPiel/Shimmer-android
For transformation of login activity:
https://github.com/roynx98/transition-button-android
For scrollable list in membership fragment:
https://github.com/yarolegovich/DiscreteScrollView
For credit cards in credit card fragment:
https://github.com/vinaygaba/CreditCardView
For supporting gif avatar:
https://github.com/koral--/android-gif-drawable
For image loading:
https://github.com/bumptech/glide
For date and time picker in gym activity:
https://github.com/wdullaer/MaterialDateTimePicker
For charts in Statistics:
https://github.com/PhilJay/MPAndroidChart

3. Challenges and Setbacks

There are several things we proposed in the project proposal but failed to achieve at last. This section will briefly discuss these problems.

In the login fragment, we choose to implement a simple login method as users can only log in through a Google account instead of implementing our own login method. This can be somewhat inconvenient to some of the users. We cancel the sign-up page as we make Google manage our account all the time.

As for the map fragment, things basically function as we expected but there is a simple problem that we have to register all the gym information into the database instead of reading gyms into databases from Google Maps as we find it can be expensive to buy such Google APIs. Though we can add this in the future versions of the application. The original idea also includes filtering gym equipment, membership as well, but that was obsolete due to difficulty and time concerns.

The biggest challenge that we met is the backend. Like we mentioned earlier in the report, we spent much time discussing backend data models. There are still slight problems within the application at the moment. Original ideas are to handle all data issues in the viewModels but most of us are not familiar with the design so we have to change the method. Then we mistakenly use a real-time database and that costs us loads of time to fix and change. The challenge teaches us a great lesson that we should spend more time discussing the data models before implementing the front end.

There are also slight problems with the user interfaces. Current UI cannot be described as satisfying. Since none of the team members is familiar with the UI. It becomes hard to implement a comprehensive UI in time. But the UI part is rather simple as we can easily handle these problems in future versions.

In the logic of the application, we actually decide not to implement the gym manager part of the application. It takes much more time and we believe if the application comes to market, it will need some marketing strategies to persuade the gym managers, which is not realistic for us.

Above are the most challenging parts in the development process, as most of us are inexperienced Android developers, we are struggling a lot these days. But we manage to learn a lot about teamwork and somewhat a little design pattern in it.

Before we deliver our project, we have set a few questions for the tester or user to rate our design. Unfortunately, we didn’t find enough users to test the application and fill our questionnaire. Thus, we can’t collect suggestions to improve and reflect our design. So we decided to put this in our next step plan.

4. Next Steps

The application definitely has many places to be improved or modified.

First is that we choose to use Google accounts only to do the login check. The main reason here is that we believe it's natural for our target users. who we believe should be mostly under their 30s, to have at least one Google account. So that instead of implementing a login method on our own, using Google account login will save much more time spent on the application.

Secondly, our application at this stage is mainly focusing on the usage of direct users. In order to create a healthy ecosystem, we are planning to release a separate platform for the businesses. The businesses are able to list their gyms onto our application and the information of each gym will be reviewed by the development team to make sure all the details are legit. They will be able to list promotions they have or to choose to advertise via us. We can integrate every gym's payment system into our application to streamline the sign-up process. By cooperating with businesses, we would be able to show more gyms in our app and provide the users with a better overview of what they desire. The users would be able to view their membership remaining days, tier and benefits that come with it. In a more advanced approach, our application could view users' annual spend on the gym under their authorization and suggest an alternative solution to reduce gym costs by suggesting gyms near their regularly visited gyms with similar or higher reviews or suggesting taking one of the promotions available at the moment. The user will also be able to track their annual workout history and trends. The application can give suggestions regarding workout frequency or ways to maintain a healthy lifestyle. The end goal of our application is to provide a transparent gym selection process and have our application fit into every users' gym workout.

Thirdly, as per the previous section where the suggestions couldn't be collected due to limited user count. A new review area will be created within the application and give users a certain initiative to give us comments and suggestions. Hopefully, we would be able to get more awesome advice to improve our application even further.

These features are hopefully to be implemented in the near future updates of the application.

5. References

Ang, C., 2020. *Fitness apps grew by nearly 50% during the first half of 2020, study finds*. [online] World Economic Forum. Available at: <https://www.weforum.org/agenda/2020/09/fitness-apps-gym-health-downloads/> [Accessed 27 October 2021].

Davalos, J., 2021. *How Covid-19 Has Permanently Changed the Fitness Industry*. [online] Bloomberg.com. Available at: <https://www.bloomberg.com/news/articles/2021-01-19/fitness-industry-may-never-return-to-its-old-ways-after-covid-19> [Accessed 27 October 2021].

ESCHNER, K., 2020. *COVID-19 has changed how people exercise, but that doesn't mean gyms are going away*. [online] Fortune. Available at: <https://fortune.com/2020/06/11/coronavirus-gyms-workouts-fitness-apps-reopening/> [Accessed 26 October 2021].

Npr.org. 2020. *My Gym Is Reopening. Is It Safe To Work Out There?*. [online] Available at: <https://www.npr.org/sections/health-shots/2020/07/05/884927215/my-gym-is-reopening-is-it-safe-to-work-out-there> [Accessed 29 October 2021].

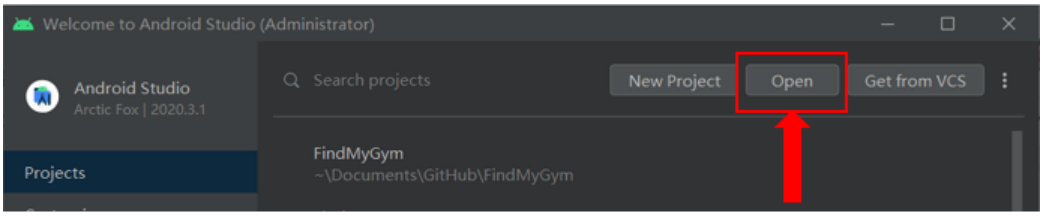
6. Appendix

1. Development Environment:

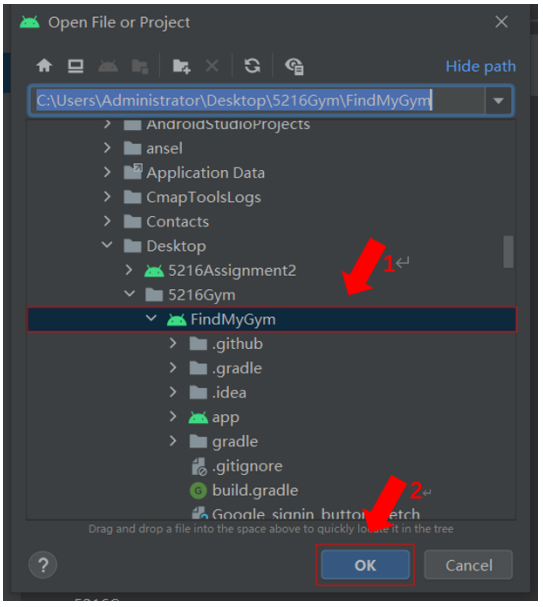
   Android Studio: 2020.3.1

   Version control: Github

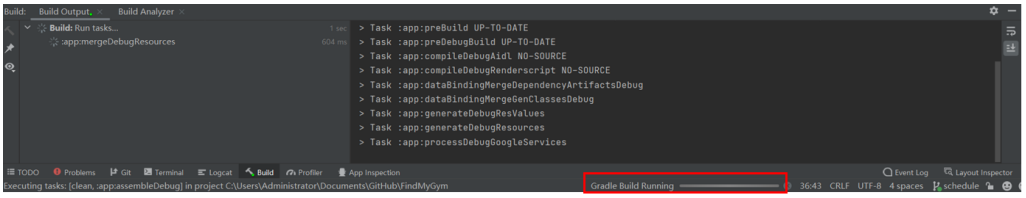
2. Unzipped the project
3. Open project in Android Studio



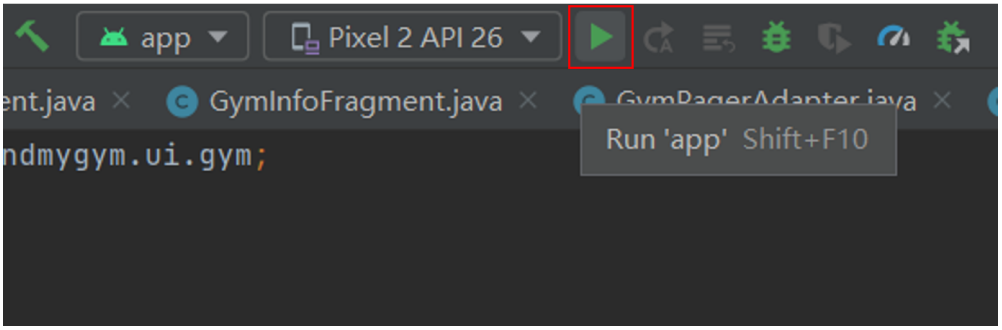
4. Select the project file and press “Ok”



5. Wait for the Android Studio to build the project automatically.



6. Run the project in Android Virtual Devices


app
Pixel 2 API 26
ent.java
GymInfoFragment.java
GymPagerAdapter.java
ndmygym.ui.gym;
Run 'app' Shift+F10