TikTok Youth Camp Android Development Group Project Report

Taxi Fare Comparator

Done by:
Mervyn Chiong Jia Rong
Lim Ke En
Apurva Mishra
Yang Dong Han
Wong Chu Yi Cloey

Table of Contents

1.Introduction

In this project, our group aims to create an application with its main function of comparing taxi fares between different riding companies such as ComfortDelgro, Grab and Gojek.

In this report, we will first elaborate on our project inspiration, followed by the technical overview of our application, then product overview and lastly our project management.

1.1 Product Inspiration and Overview

Most taxi services can account their success to one simple fact – effectively reducing our efforts required when commuting. Before the online ride-hailing boom, one would have to physically wait along the roadside to flag down a taxi. Now, we seldom find ourselves doing so as companies have transitioned to using mobile applications. Payment has also been nearly automated. In today’s context, there are numerous rising applications and choices of taxi services. Commuters now have a plethora of ride-hailing companies to choose from but currently, the 3 major companies in Singapore are Grab, Gojek and ComfortDelgro. As people are inclined to opt for the most cost effective option, ride-hailing is now tedious – opening individual applications, re-entering the location and destination, and mentally comparing prices. In some unfortunate cases, by the time commuters have decided on the company to book their rides, prices might have surged.

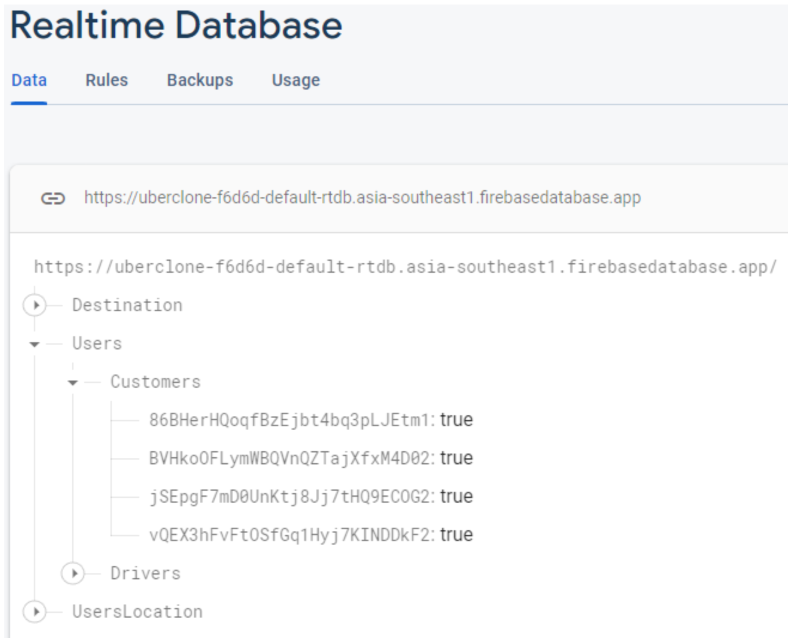
Our application solves this problem. The user only needs to enter the destination once, and is presented the fares from different companies in ascending order, all in just one click. This makes the process of booking rides very effective and efficient. Therefore, helping users save cost on expensive ride-hailing services.

2.Technical Overview

2.1 Framework and Libraries

For our application we utilized 3 main frameworks in Firebase, Geofire and Google Maps.

The main reason for selecting those 3 frameworks was to create an app with further enhancements to be established down the development pipeline. Google Maps is the key feature for our TaxiComparator application which helped us obtain the user's location coordinates. This facilitated us in creating a code which calculates the distance between the starting location and destination. Subsequently, it would be used to run through taxi companies' fare calculation to feedback an estimated price from each company.

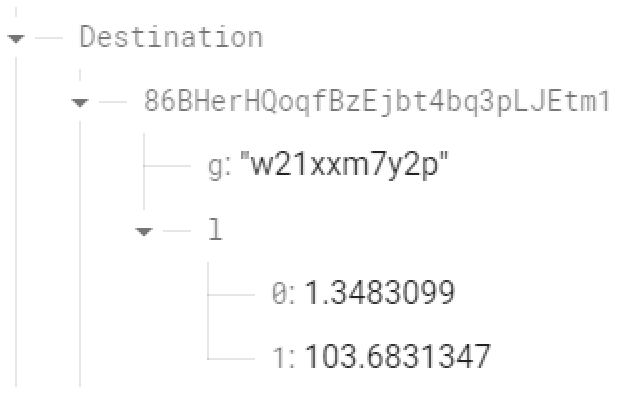


```
https://uberclone-f6d6d-default-rtdb.asia-southeast1.firebasedatabase.app

https://uberclone-f6d6d-default-rtdb.asia-southeast1.firebasedatabase.app/
  Destination
  Users
    Customers
      86BHerHQoqfBzEjbt4bq3pLJEtm1: true
      BVHkoOFLymWBQVnQZTajXfxM4D02: true
      jSEpgF7mD0UnKtj8Jj7tHQ9EC0G2: true
      vQEX3hFvFtOSfGq1Hyj7KINDDkF2: true
    Drivers
  UsersLocation
```

Figure 1: Realtime Database:Customers

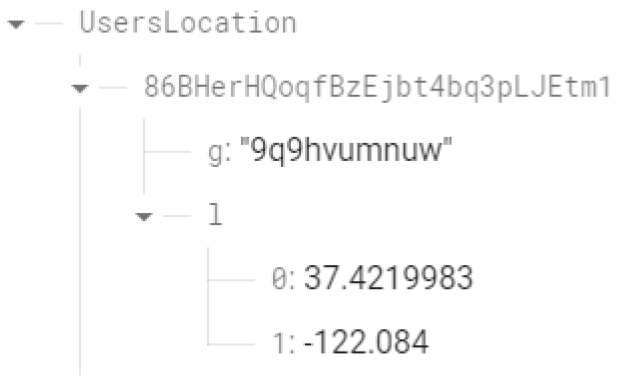
Firebase was initially used to store User's details only. As seen in Fig 1, if we choose to upscale this application, we would have to include an input for Drivers.



```
Destination
  86BHerHQoqfBzEjbt4bq3pLJEtm1
    g: "w21xxm7y2p"
    l
      0: 1.3483099
      1: 103.6831347
```

Figure 2: Realtime Database:Destination

Using Geofire, Figure 2 highlights the Destination location's Latitude and Longitude when users input their location into the search bar.

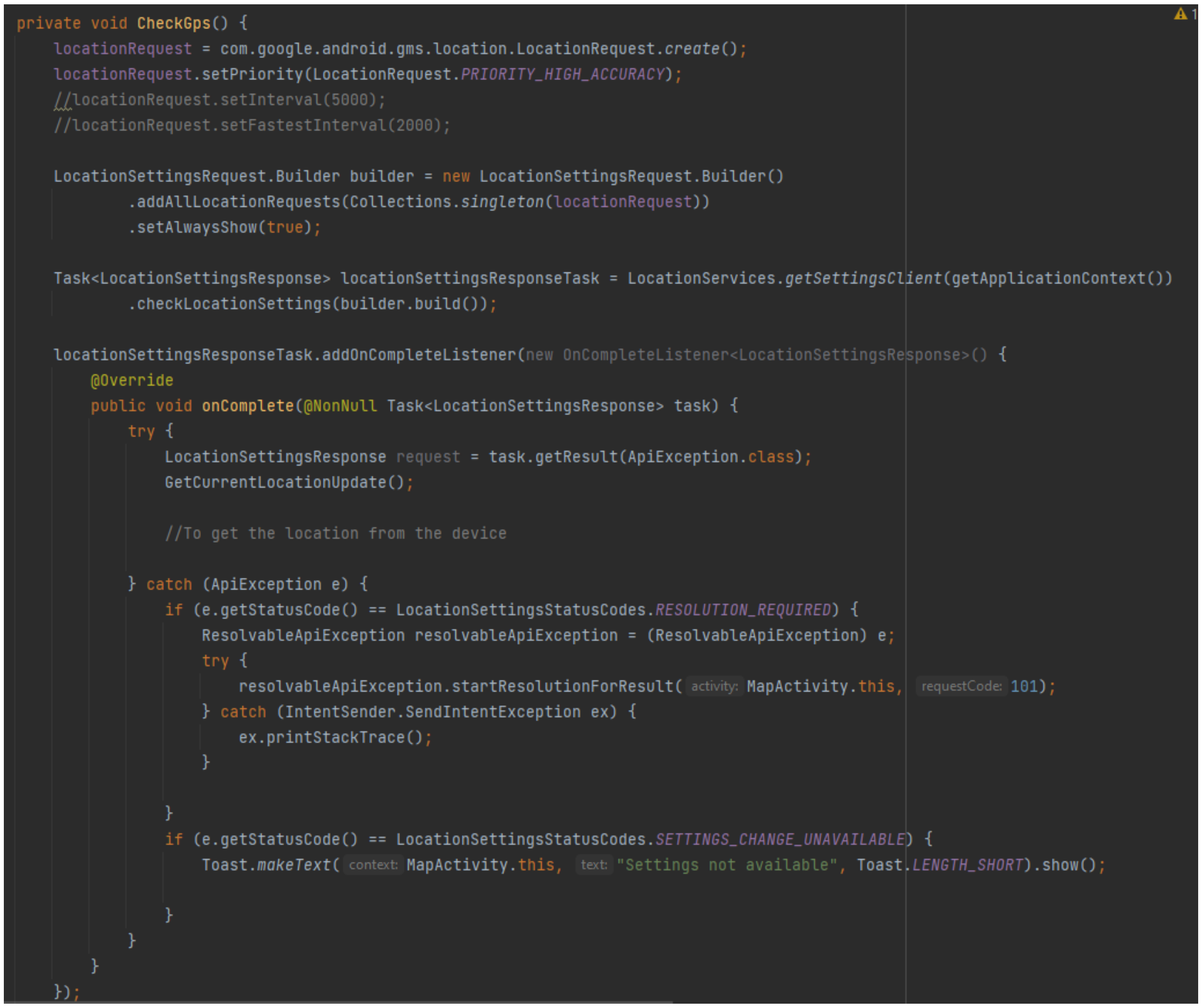


```
UsersLocation
  86BHerHQoqfBzEjbt4bq3pLJEtm1
    g: "9q9hvumnuw"
    l
      0: 37.4219983
      1: -122.084
```

Figure 3: Realtime Database:User's Location

Figure 3 shows the user's Global Positioning System (GPS) location's longitude and latitude. From these 2 results, we can obtain the distance for travel. Geofire would be used together with further enhancements to the application i.e. If we require constant GPS updates, we need to update the database and then recalculate the distance.

2.2 Interesting Code



```
private void CheckGps() {
    locationRequest = com.google.android.gms.location.LocationRequest.create();
    locationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
    //locationRequest.setInterval(5000);
    //locationRequest.setFastestInterval(2000);

    LocationSettingsRequest.Builder builder = new LocationSettingsRequest.Builder()
            .addAllLocationRequests(Collections.singleton(locationRequest))
            .setAlwaysShow(true);

    Task<LocationSettingsResponse> locationSettingsResponseTask = LocationServices.getSettingsClient(getApplicationContext())
            .checkLocationSettings(builder.build());

    locationSettingsResponseTask.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>() {
        @Override
        public void onComplete(@NonNull Task<LocationSettingsResponse> task) {
            try {
                LocationSettingsResponse request = task.getResult(ApiException.class);
                GetCurrentLocationUpdate();

                //To get the location from the device

            } catch (ApiException e) {
                if (e.getStatusCode() == LocationSettingsStatusCodes.RESOLUTION_REQUIRED) {
                    ResolvableApiException resolvableApiException = (ResolvableApiException) e;
                    try {
                        resolvableApiException.startResolutionForResult( activity: MapActivity.this,  requestCode: 101);
                    } catch (IntentSender.SendIntentException ex) {
                        ex.printStackTrace();
                    }

                }
                if (e.getStatusCode() == LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE) {
                    Toast.makeText( context: MapActivity.this,  text: "Settings not available", Toast.LENGTH_SHORT).show();

                }
            }
        }
    });
```

Figure 4: CheckGps()

The CheckGps function is probably the most essential function for the whole application. Firstly, it establishes a connection using LocationRequest and then proceeds to see if location settings are built. Subsequently, it will call upon GetCurrentLocationUpdate(). This function can be repeated multiple times in the above commented code where we set an interval to constantly check the user's location. GetCurrentLocationUpdate() is the function to parse the updated data to Geofire.

3.Product Overview

3.1 Login Page

We also created a login page to direct users to log in using their registered account. However, if one does not have an account, he/she is able to create an account using the personal email address.

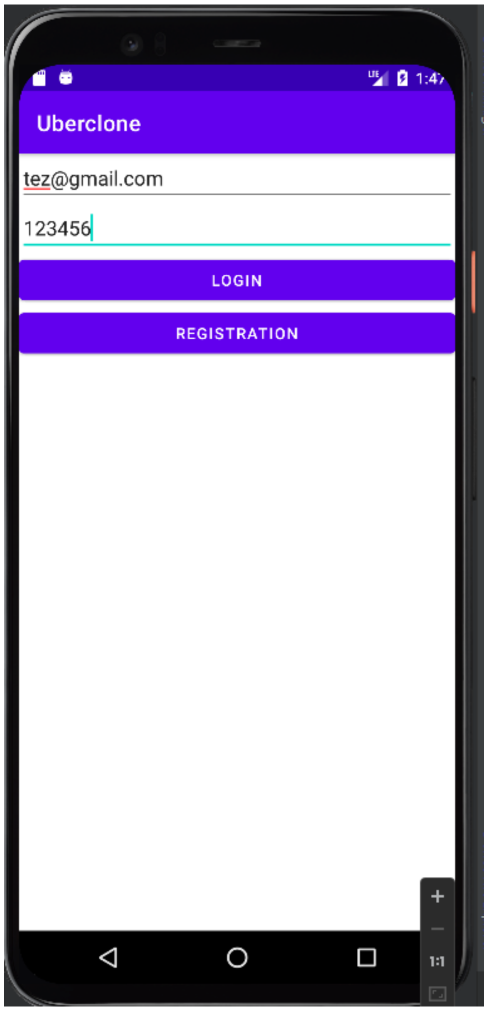


Figure 5: Login page

3.2 Navigation Bar

For ease of usage, we designed a navigation bar where users can toggle between different pages just by sliding the navigation button. From Figure 6, we included pages such as the home page and login page. Using the navigation bar, users can navigate to the login activity and sign in to use the application. In the future, we would like to include more features. On top of the login page, we could include more pages such as real-time updates on the road situation in Singapore to provide users the best user experience and information. However, for simplicity, we are only including the home page and the login page in the navigation bar.

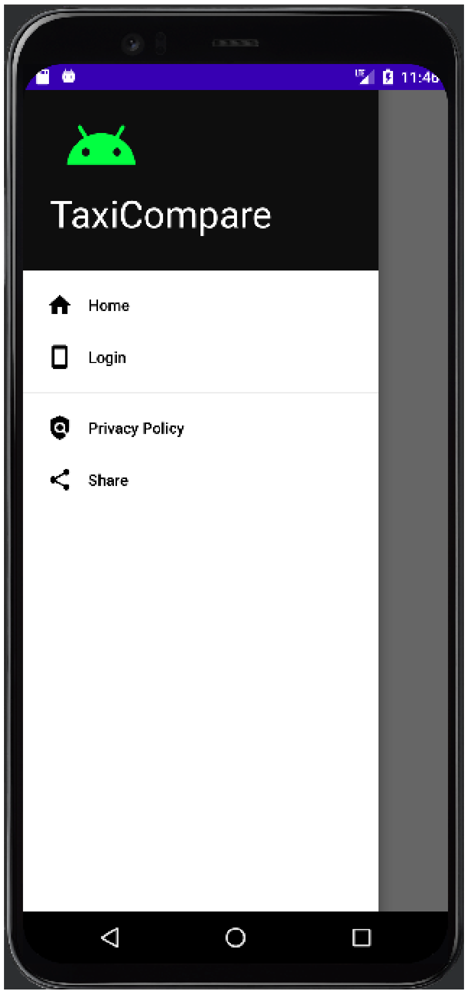


Figure 6: Navigation Bar

3.3 Map Activity Page

This page allows users to obtain their current location and input their destination into the search bar. Next, we created a function on this page to calculate the distance between the 2 locations. The output distance will then be processed into the other functions to calculate the estimated taxi fares for ComfortDelgro, Grab and GoJek each.
The logout button at the top left corner will allow users to log out of the account whenever they want to. We noticed an error was occurring with the logout button. However, due to time constraints we have put it under a //TODO so that future developers are able to recognise the problem easier

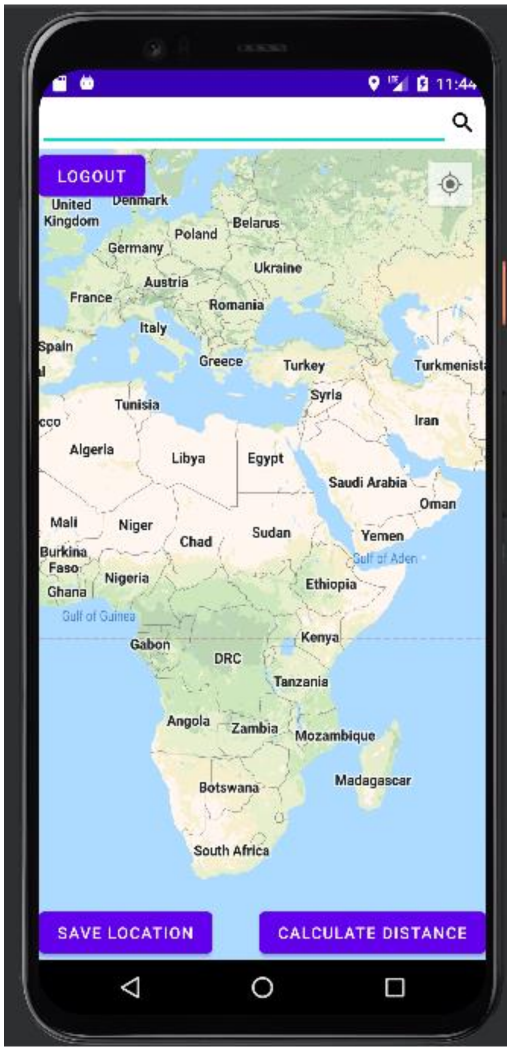


Figure 7: Map Page

3.4 Calculating fare

The fares are calculated using functions written for each company. We obtained the fare rates from ComfortDelgro, Grab and Gojek receptive websites. For simplicity, we standardized the type of rides for each company – Metered Fare Taxi (ComfortDelgro), GrabTaxi (Grab) and Go-Taxi (Gojek). However, there are some limitations to our fare calculation functions. Grab has an additional "per km rate" which we are unable to calculate as we do not have access to the data for the duration of the rides. It is unannounced whether Gojek's fare system is inclusive of the "per km rate", therefore we excluded it. Furthermore, we did not include the surge prices and additional fares incurred during a ride such as the Electronic Road Pricing. However, with future enhancements to this application, we aim to tap on the API of Land Transport Authority to gather real-time traffic conditions to account for peak hour and surcharge pricing. These limitations would result in the discrepancies in accuracy of the calculated fare.

4.Highlights

A large challenge we faced was integrating different parts of the codes we wrote together into one main branch. We are all new to working remotely on an application and had to use GitHub for the first time. Therefore, the learning curve was steep because we had to learn new Git Commands and create new branches to work on individually. This took up a significant portion of our time working on this project because without this skill, we will not be able to make progress and pull from each other's codes. Additionally, there were more layers and steps to the debugging process than a conventional program. As there were many aspects to an application, from front end to back end, it was hard to identify what the specific issue was.

4.1 Test Case [post login]

Input: click the gps location button, and type” permanente creek”

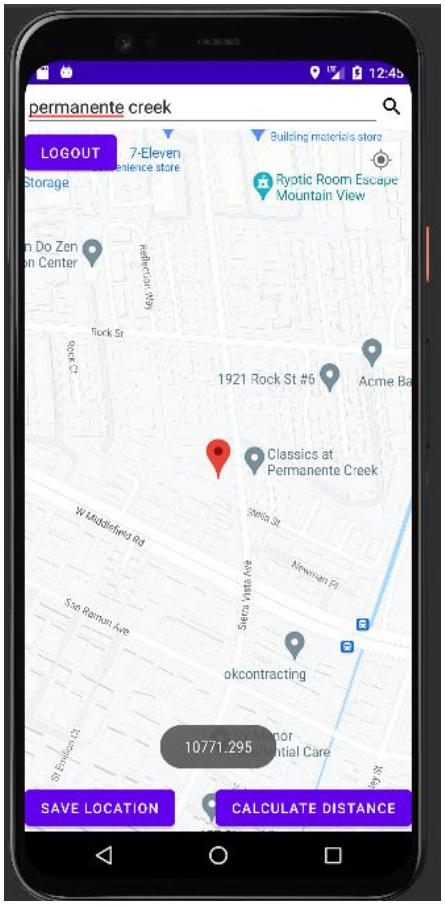
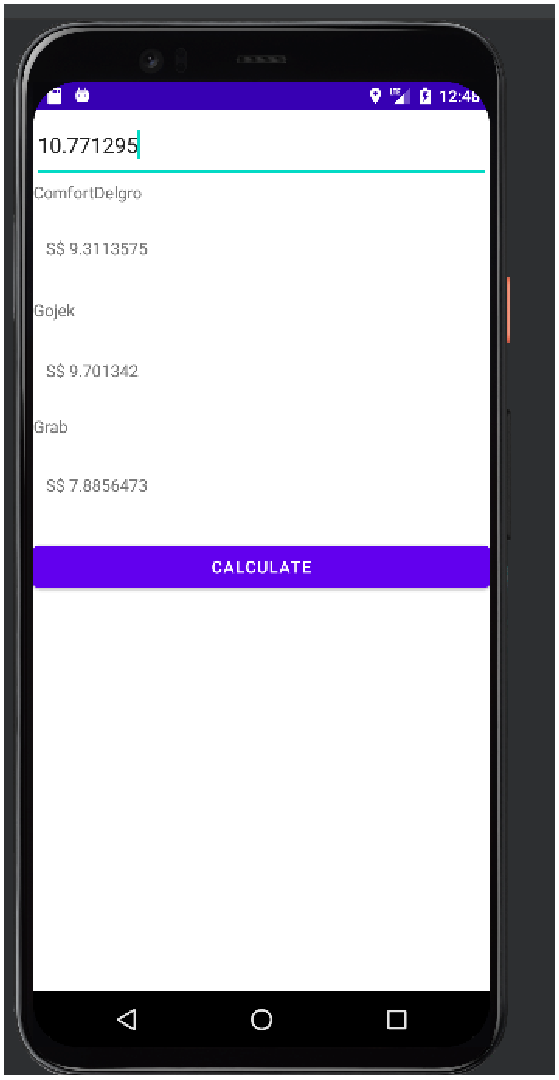


Figure 8: Distance Measuring

Input distance into the Calculation:10.771295


10.771295

ComfortDelgro

S$ 9.3113575

Gojek

S$ 9.701342

Grab

S$ 7.8856473

Figure 9: Calculation Output

5.Project Management

5.1 Work allocation overview

Member	Role
Mervyn	Leader, Backend Development
Lim Ke En	Lead designer,Front end development
Apurva Mishra	Designer,Front end development
Wong Chu Yi Cloey	Backend Development
Yang Dong Han	

5.2 Role Distribution

Overall, everyone contributed into the theory crafting on how the application should turn out. Everyone in the group has little to no experience in developing an application, however, that did not stop us from challenging ourselves.

As each of us are interested in different portions of the application, we allowed each of us to choose which part we wanted to explore. For instance, as some of us are more interested or have better skills in backend development, he/she will initiate to take part and complete that segment of the project. While the others, who are interested in the front-end development, will then complement them by focusing on the design or the UI interface of the application.

6. Conclusion

In conclusion, our group tried to build an application that can calculate the taxi fare just by the location given by the user. However, we acknowledge that our project is still far from being a completed and perfect application. There were numerous hiccups in our application such as being unable to generate the exact distance between locations, this would definitely impact the accuracy of our result. Furthermore, our application for now is still not user-friendly, many of the design and UI interface needs to be improved. Being aware of the problems and incompleteness of our project, we believe that in future, with greater knowledge and resources, this application may be helpful and useful to many others in the world.