CS3301 P1: Android App Development Report

Matriculation number: 160021429

1. Overview

In this practical, we were asked to design and implement an Android app which works similar to the Uber app. The main aim of this practical is to make our app to communicate with other pre-built components.

1 – 1. Language and Framework

To implement the Android app, I used React Native and expo. The React Native is a JavaScript framework that is used to build a mobile app. And, the expo is a set of tools, libraries and services which let you build native iOS and Android apps by writing JavaScript. Also, by using expo, you could build an apk file easily.

1 – 2. A list of features of the application in the specification

1) User account creation and login.

2) Once the user logs in, the app will display a map of the user's current location and the available drivers within a reasonable distance.

3) Request of trip:

   a. Basic version: send out a request of a trip in a form including the pickup location, the destination, the time for the journey, along with any other useful or desirable information

   b. Advanced version: the user is able to mark the pickup location and the destination by placing the location pins at a desired pickup location and destination

4) Display a list of drivers who are available and reply to the request.

5) Track the driver's location and display in real time

6) When the trip is completed, a summary page will be displayed, including the total cost and the rating

7) History:

a. Present the historical trips in various aspects: distances, fares, drivers, etc

b. Use a visualisation component to present the historical trips in a meaningful way

8) Interactions with users:

a. Use a notification component to notify users when their drivers arrive
b. b. Use a call component to initiate a call to the driver

9. Payment: Use various payment components to support fare payment

Features that I have done: 1, 2, 3-a, 3-b, 4, 5, 6, 8-b

2. Design

According to the lecture notes, there are 4 kinds of components in the Android: 1) Activities, 2) Services, 3) Broadcast Receivers, and 4) Content Providers.

2 – 1. Activities

An activity represents a single screen with a user interface. I used buttons and text inputs to allow the users to register, log in, and proceed the services. Also, my app displays the google map on the screen, and simulate the driver's movement on the screen by using the google map animation functions.

There are 11 Activities in my app: 1) Log in screen, 2) Sign up screen, 3) Menu screen, which allows the user to choose one of the options to proceed the service, 4) MarkScreen, which allows the user to mark the pickup location and the destination by placing the location pins at a desired location, 5) Main screen that shows the available drivers on the map, and allows the user to search the destination by using the search bar on the top of the screen, 6) RequestTripScreen that supports the user to select the driver, and request the journey, 7) CheckRequestScreen, which displays the details of the requested journey: driver name, destination of the journey, estimated journey time, starting time of the journey, and the estimated fare of the trip, 8) LinkScreen that displays all trips that the client requested, and allow the client to start the journey, and make a phone call to the driver, 9) JourneyScreen that tracks the driver's location and display in real time on the google map, 10) SummaryScreen that includes the total cost and rating, and 11) LoadingScreen.

And I used React Stack Navigator to connect activities. The Stack Navigator is an object that uses stack to push the activity instances in it. For example, if the user launched Log in screen and Menu screen, the Stack Navigator will push the Log in screen and Menu screen to the stack. And when the user presses the back button, the Stack Navigator will pop the top activity screen from the stack, and the poped activity will be launched on the screen.

Codes for the Stack Navigator are in the “App.js” file. All other code files are in the “src” directory. In the “src” directory, there are 2 sub-directories: container and component. The container directory contains the source code files about activities, and the component directory contains the files for the custom UI components.

2 – 2. Services

A service is a component that runs in the background to perform long-running operations. For example, fetching data over the network without blocking user interaction with an activity is a service component.

To track the driver, display the available drivers on the map, and get the geolocational information (latitude and longitude) or name of the location, I used 4 kinds of google map APIs: 1) Places API, 2) Geocoding API, 3) Directions API, and 4) Distance Matrix API. And while using these APIs, the data fetching process does not block the user interaction with an activity.

1) Places API

The Places API is used to implement the search bar autocomplete feature. While implementing the MainScreen activity, which is for searching the destination to request the journey, I realized that it might be worth to make the app to have a search bar with the autocomplete feature, so that the user could search the destination easily.

When the user input some characters in the search bar, then the app will use the Google Places API to get the list of places that are matching to the user input, fetch the data, and display the place names on the screen.

2) Geocoding API

The Geocoding API helps the user to get the name of the location by passing the latitude and longitude of the target location. This API is used in the MarkScreen.js file, to get the name of the destination, so that the user could get the name of the place where the user placed the pin at.

3) Directions API

The Directions API is a service that calculates directions between locations using an HTTP request. By using this API, the app could get the polyline that along the way of the journey and the list of latitude/longitude coordinates between the pickup location and the destination. The Journey.js contains the code that uses the Directions API to get the list of coordinates and polyline to draw the polyline along the journey and use the animation to move the driver marker to move on the drawn polyline. This allows the user to track the driver in real time.

4) Distance Matrix API

The Distance Matrix API is a service that provides travel distance and time for a matrix of origins and destinations. The API returns information based on the recommended route between start and end points, as calculated by the Google Maps API, and consists of rows containing duration and distance values for each pair. By using the Distance API, the app gets the estimated journey time and the total distance of the journey. Moreover, by using this distance value, the app will calculate the estimated fare.

2 – 3. Broadcast Receivers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action.

By using the ‘react-native-phone-call' module, my app could allow the user to initiate the phone call to the driver. And this phone call linker uses the Broadcast Receiver component to communicate with the phone call app and allow the user to have a phone call with the driver.

2 – 4. Content Providers

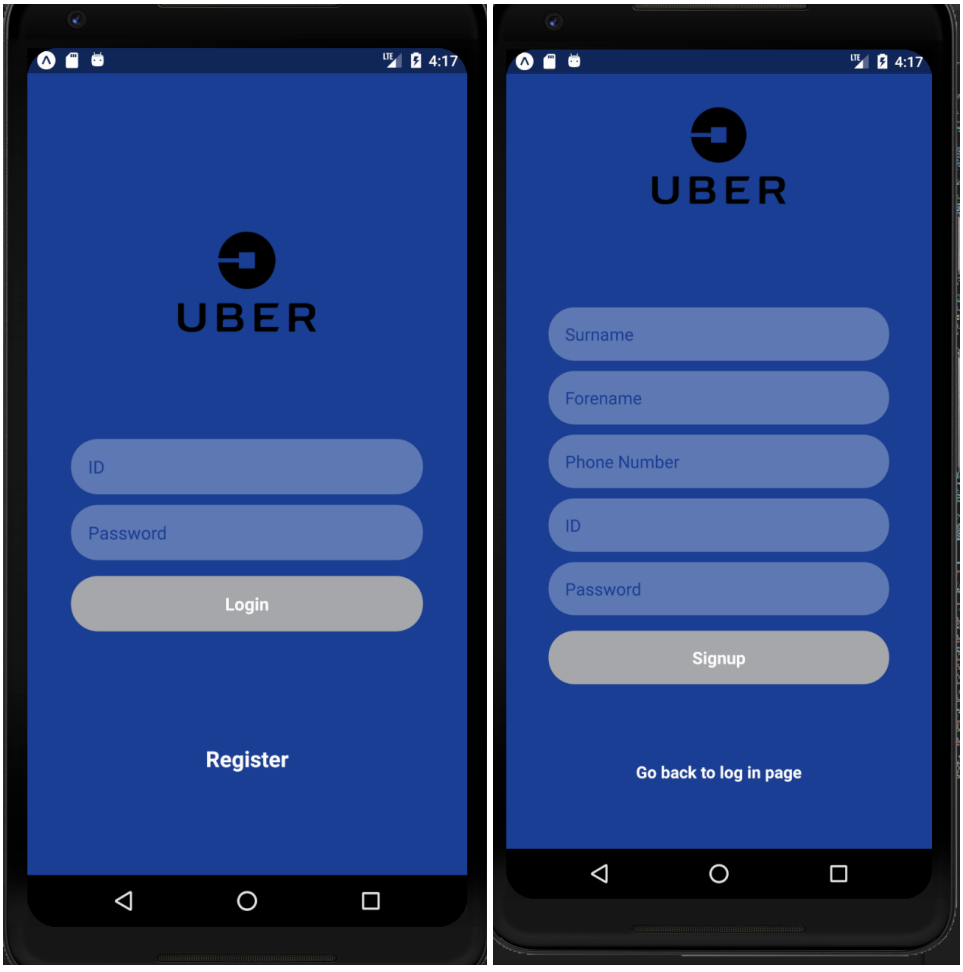
A content provider component supplies data from one application to others on request. The data may be stored in the file system, the database or somewhere else entirely.

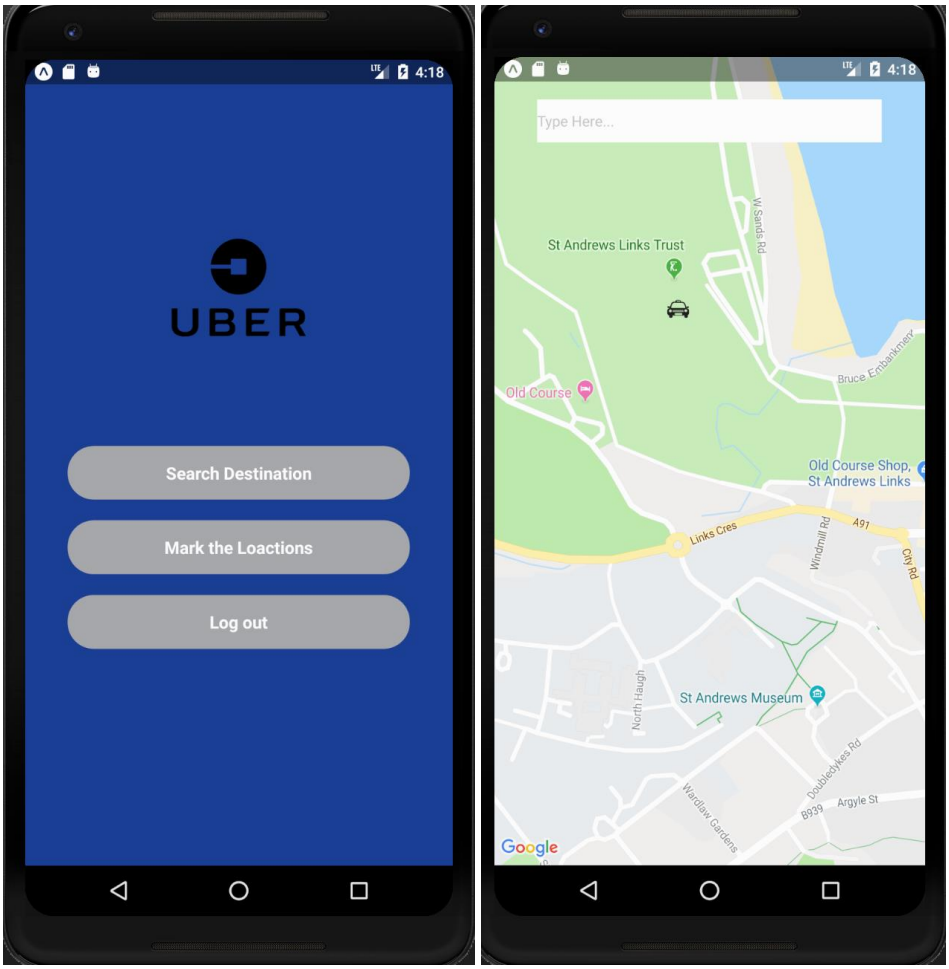
As there is no backend server for this app, it stores all data in the local storage. The app uses the local storage to store the list of registered users, logged user, list of available drivers, and list of journeys.

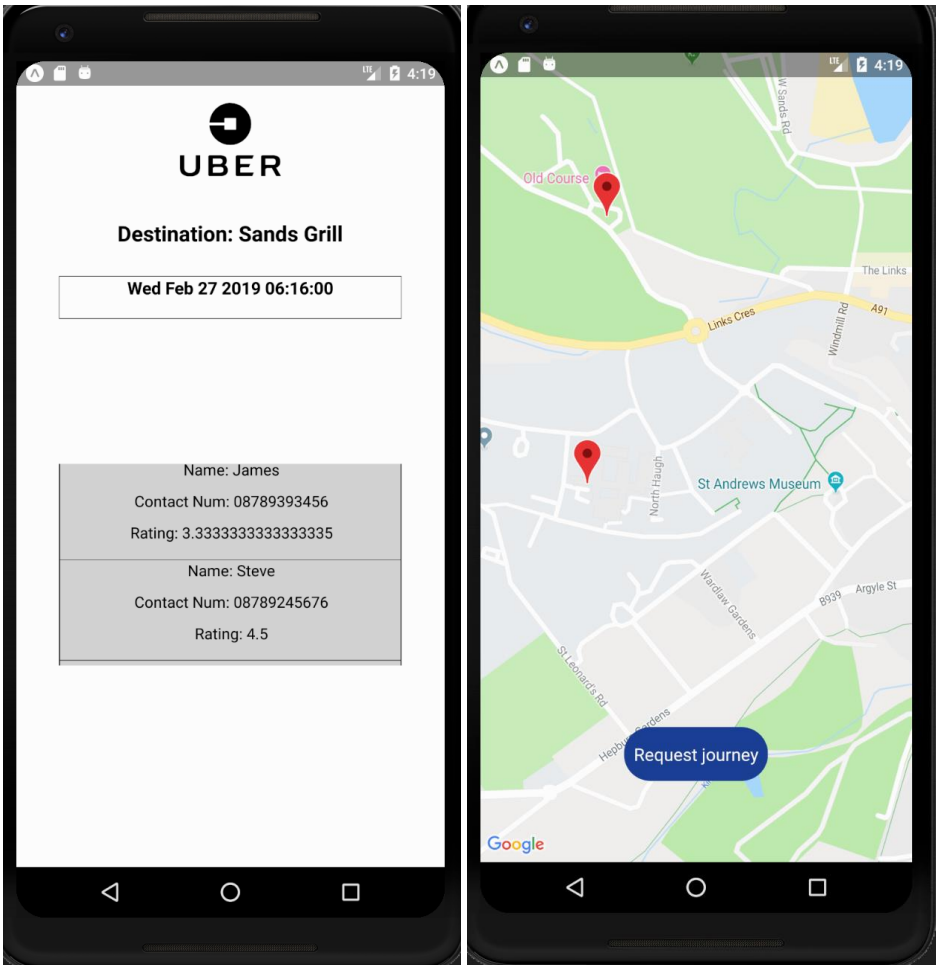
3. Evaluation

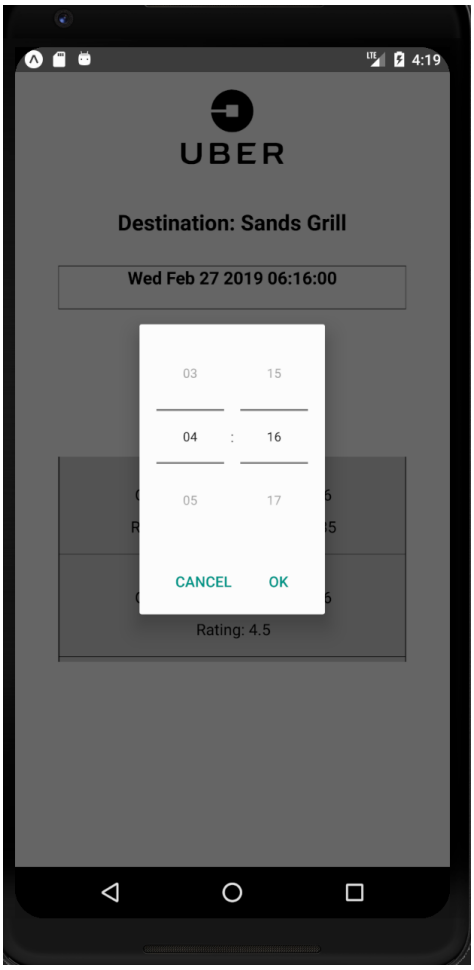
I used Android Emulator to test every little bit of progress each time. By using the console.log() and alert() functions, I checked all procedures and values of all variables. Moreover, I implemented some callback functions to alert error when the asynchronous functions have some error.

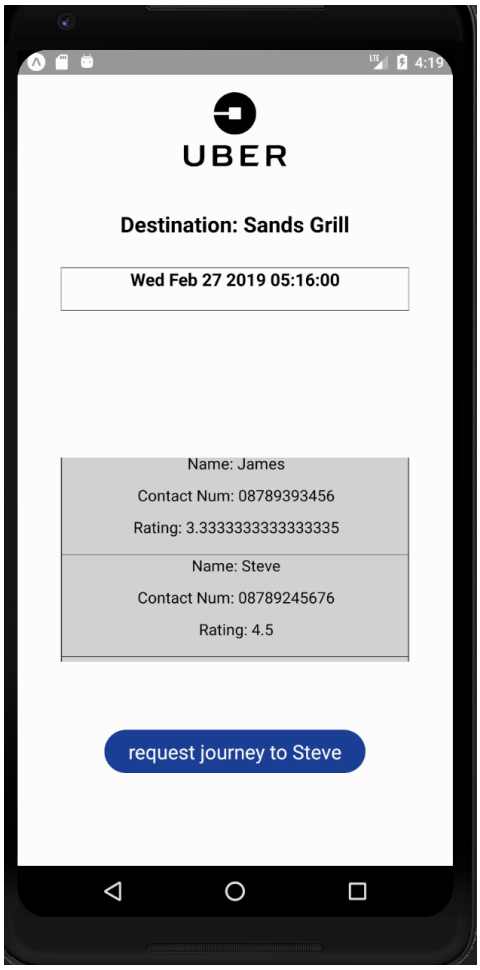
At the end, all features worked properly, and there were no errors in my app.

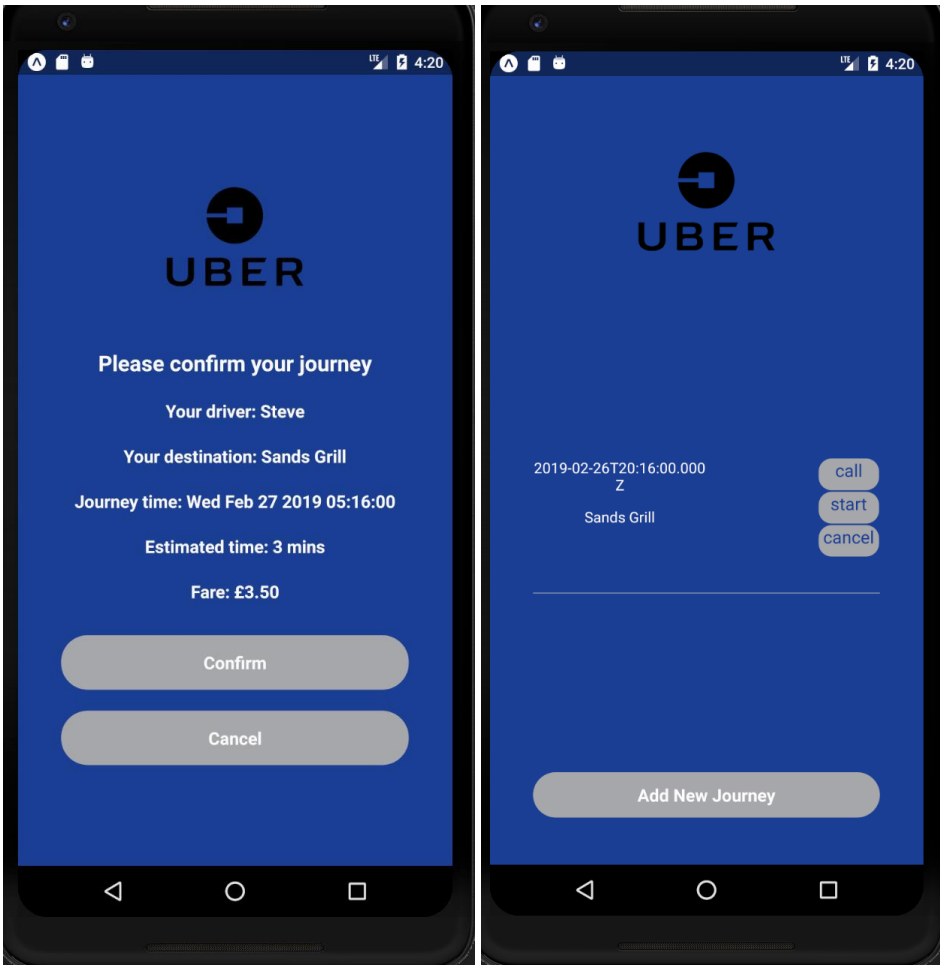


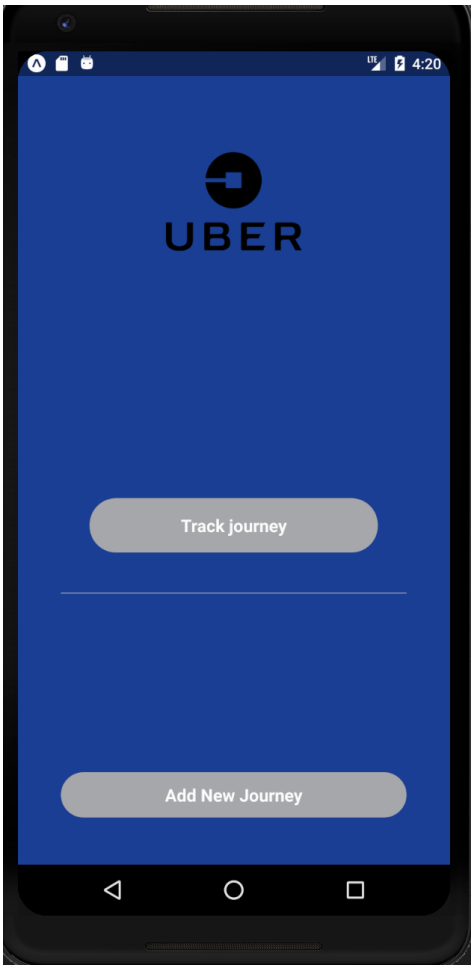

4:18
UBER
Search Destination
Mark the Loactions
Log out
4:18
Type Here...
St Andrews Links Trust
W Sands Rd
Old Course
Bruce Embankment
Old Course Shop, St Andrews Links
Links Cres
A91
Windmill Rd
City Rd
North Haugh
St Andrews Museum
Doubledykes Rd
B939
Argyle St
Wardlaw Gardens
Google

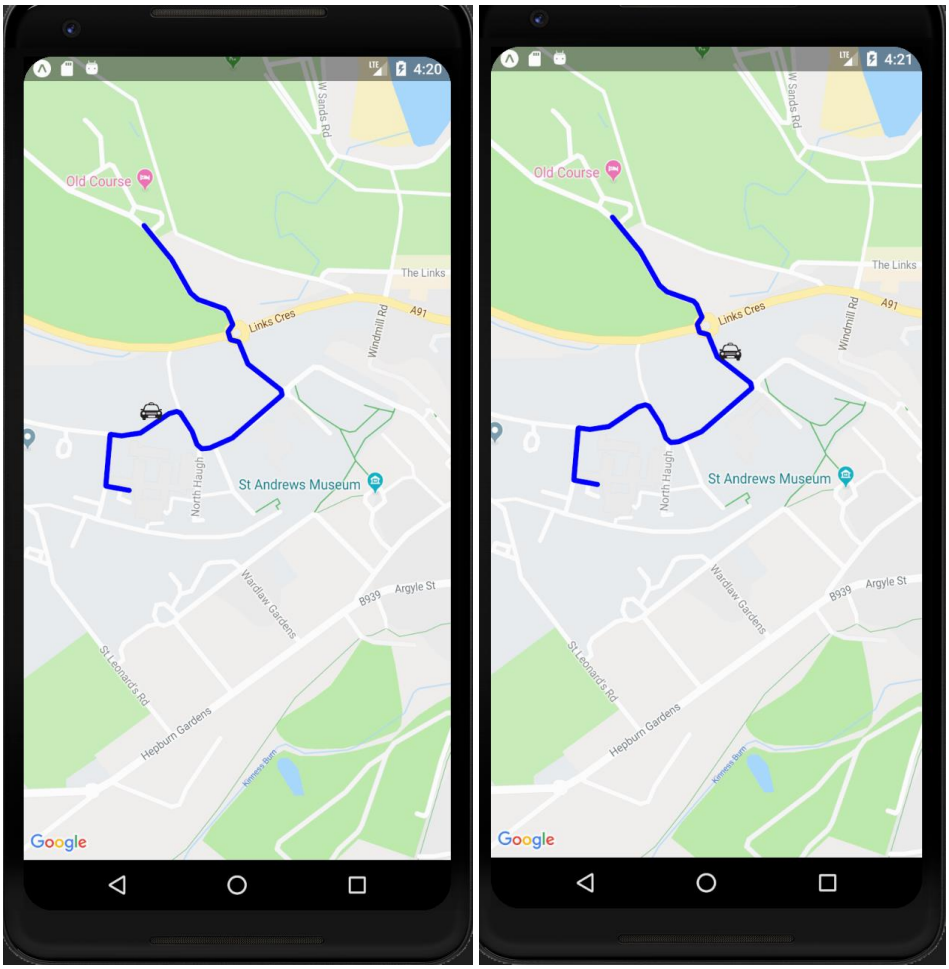

4:19
UBER
Destination: Sands Grill
Wed Feb 27 2019 06:16:00
Name: James
Contact Num: 08789393456
Rating: 3.3333333333333335
Name: Steve
Contact Num: 08789245676
Rating: 4.5
4:19
Old Course
The Links
Links Cres
A91
Windmill Rd
W Sands Rd
North Haugh
St Andrews Museum
Wardlaw Gardens
B939
Argyle St
St Leonards Rd
Request journey
Google

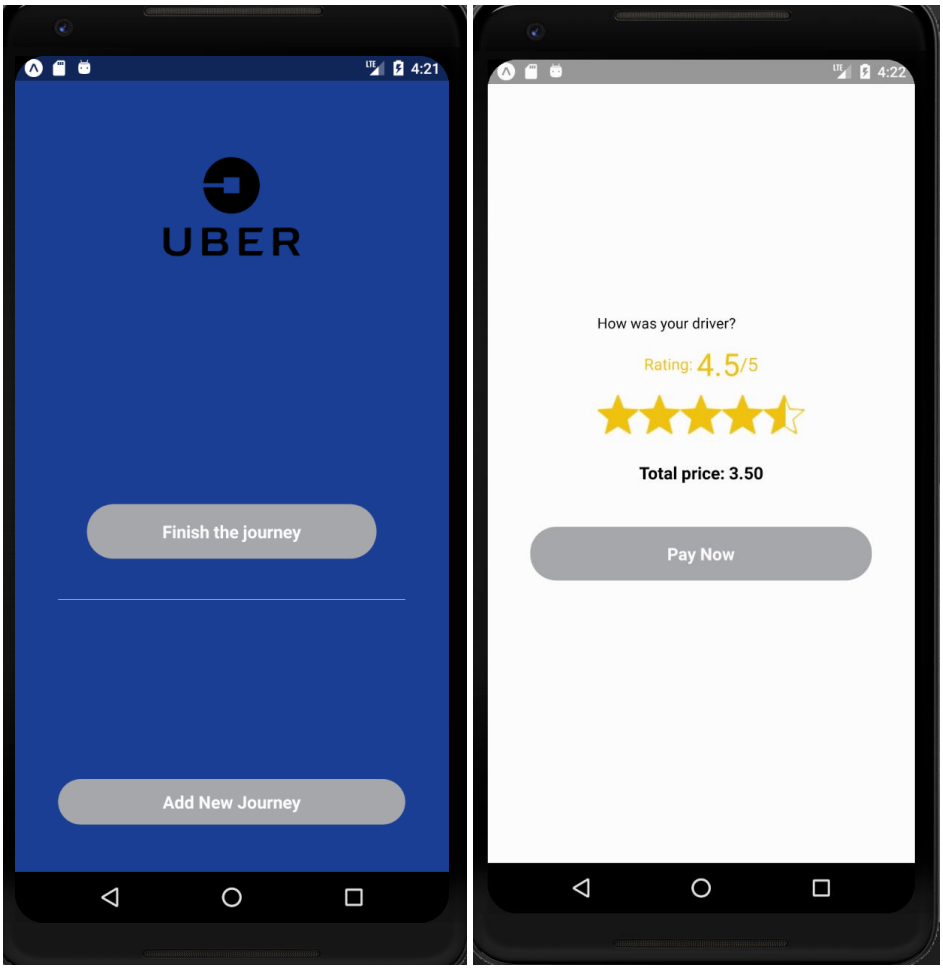

4:19
UBER
Destination: Sands Grill
Wed Feb 27 2019 06:16:00
03
15
04
:
16
05
17
CANCEL
OK
Rating: 4.5


4:19
UBER
Destination: Sands Grill
Wed Feb 27 2019 05:16:00
Name: James
Contact Num: 08789393456
Rating: 3.3333333333333335
Name: Steve
Contact Num: 08789245676
Rating: 4.5
request journey to Steve


4:20
UBER
Please confirm your journey
Your driver: Steve
Your destination: Sands Grill
Journey time: Wed Feb 27 2019 05:16:00
Estimated time: 3 mins
Fare: £3.50
Confirm
Cancel
4:20
UBER
2019-02-26T20:16:00.000
Z
Sands Grill
call
start
cancel
Add New Journey


4:20
UBER
Track journey
Add New Journey


4:20
Old Course
W Sands Rd
The Links
Links Cres
Windmill Rd
A91
North Haugh
St Andrews Museum
Wardlaw Gardens
B939
Argyle St
St Leonard's Rd
Hepburn Gardens
Kinness Burn
Google
4:21
Old Course
W Sands Rd
The Links
Links Cres
Windmill Rd
A91
North Haugh
St Andrews Museum
Wardlaw Gardens
B939
Argyle St
St Leonard's Rd
Hepburn Gardens
Kinness Burn
Google



4. Conclusion

To be honest, this practical was really helpful for me to understand the concepts of the Android Components. I had a lot of research to understand the concept of Activity, Service, Content Provider, and Broadcast Receiver, and had a lot of googling to learn the way to use and link different components to improve the app's functionality.