DWIT COLLEGE

DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

Institute of Science and Technology

MOBILE APPLICATION TO AUTO CALCULATE TAXI AND MICROBUS FARE

A PROJECT REPORT

Submitted to

Department of Computer Science and Information Technology

DWIT College

In partial fulfillment of the requirements for the Bachelor's Degree in Computer Science and Information Technology

Submitted by

Sandesh Sharma

August, 2016

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

SUPERVISOR’S RECOMENDATION

I hereby recommend that this project prepared under my supervision by SANDESH SHARMA entitled **“MOBILE APPLICATION TO AUTO CALCULATE TAXI AND MICROBUS FARE”** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology be processed for the evaluation.

…………………………………………

Rituraj Lamsal

Lecturer

Deerwalk Institute of Technology

DWIT College

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

LETTER OF APPROVAL

This is to certify that this project prepared by SANDESH SHARMA entitled **“MOBILE APPLICATION TO AUTO CALCULATE TAXI AND MICROBUS FARE”** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

…………………………………… Rituraj Lamsal [Supervisor] Lecturer DWIT College	**………………………………………… Hitesh Karki Chief Academic Officer DWIT College**
…………………………………………….. Jagdish Bhatta [External Examiner] IOST, Tribhuvan University	**…………………………………………….. Sarbin Sayami [Internal Examiner] Assistant Professor IOST, Tribhuvan University**

ACKNOWLEDGEMENT

This project would have been impossible to complete without the guidance and motivation of some of the highly encouraging persons. I would like to take this opportunity to thank Mr. Ritu Raj Lamsal, Faculty of DWIT whose constant supervision not only reduced the complexity of the project but also made the project fun while doing.

I would also like to express my deepest appreciation to Mr. Hitesh Karki, CAO DWIT whose never give up attitude drove me till the end of the project. His inspiration helped me a lot while getting stuck into some bugs which I thought I could never fix.

Furthermore I would also like to acknowledge with much appreciation the crucial role of Mr. Rajiv Manandhar, Engineering Manager of Deerwalk Services. He helped me a lot learning new technologies that is needed to implement this project.

At the end I would like thank all my friends whose feedbacks and suggestions helped a lot to make this project better.

Sandesh Sharma
Roll No. 1811/069

Tribhuvan University

Institute of Science and Technology

STUDENT'S DECLARATION

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

Sandesh Sharma

August 23, 2016

ABSTRACT

This is an android application built to provide the exact fare to the travellers (inside the Kathmandu valley) and (across Nepal). All the implementation part of this application is included in this report. The main objective behind this application is to help facilitate the traveller through the use of technology and give travellers more control over their travel fare. This report also includes different analysis that is done prior to the implementation of the application.

The application was built on android SDK and Android Studio was used as IDE. The user can choose the type of vehicle in which they are travelling on. The fare is calculated based on the type of vehicle and the standard fare provided by the department of transportation. The Fare is calculated on real time just like the meter of taxi. This application will track the location of user using GPS, Wi-Fi Signals and signals from the cell phone tower. This application will use the best current location based on the location from these three mediums. The location of user is tracked every 6 seconds and the distance of the user is calculated based on these consecutive locations. This application uses Google maps so that the user can view the exact path in which they are travelling on.

Keywords : GPS, Google Maps, Standard fare, track location

TABLE OF CONTENTS

LETTER OF APPROVAL i

ACKNOWLEDGEMENT ii

STUDENT'S DECLARATION iii

ABSTRACT iv

LIST OF FIGURES vii

LIST OF TABLES viii

LIST OF ABBREVIATIONS ix

CHAPTER 1: INTRODUCTION 1

1.1 Introduction 1

1.2 Problem Definition 2

1.3 Objective 2

1.4 Scope and Limitation 3

1.5 Outline of Document 4

CHAPTER 2: REQUIREMENT ANALYSIS AND FEASIBILITY STUDY 5

2.1 Related Works 5

2.2 Requirement Analysis 7

2.2.1 Functional requirement: 7

2.2.2 Non functional requirement 8

2.3 Feasibility Analysis 9

2.3.1 Economic feasibility 9

2.3.2 Technical feasibility 9

2.3.3 Schedule feasibility 9

2.3.4 Operational feasibility 10

CHAPTER 3: SYSTEM DESIGN 11

3.1 Methodology 11

3.1.1 Data collection 11

3.1.2 Algorithm 11

3.2 System Design 14
3.2.1 Class diagram 14
3.2.2 Event diagram 16
3.2.3 Sequence diagram 17
CHAPTER 4: IMPLEMENTATION AND TESTING 18
4.1 Implementation 18
4.1.1 Tools used 18
4.2 Testing 20
4.2.1 Unit testing 20
4.2.2 Integration testing 21
4.2.3 System testing 23
CHAPTER 5: MAINTENANCE AND SUPPORT 25
5.1 Maintenance 25
5.1.1 Corrective maintenance 25
5.1.2 Adaptive maintenance 25
5.1.3 Perfective maintenance 25
5.1.4 Preventive maintenance 25
5.2 Support 25
CHAPTER 6: CONCLUSION AND RECOMMENDATION 26
6.1 Conclusion 26
6.2 Recommendation 26
REFERENCES 27
BIBLIOGRAPHY 28
APPENDIX 29

LIST OF FIGURES

Figure 1 Project Block Diagram 4

Figure 2 Use Case Diagram of the application 7

Figure 3 Critical path schedule 10

Figure 4 Class diagram of the application 15

Figure 5 Event diagram of the application.. 16

Figure 6 Sequence diagram of the application... 17

LIST OF TABLES

Table 1 CPM table 9

Table 2 Test case 1 for unit testing 20

Table 3 Test case 2 for unit testing 20

Table 4 Test case 3 for unit testing 21

Table 5 Test case 1 for Integration testing... 21

Table 6 Test case 2 for Integration testing... 22

Table 7 Test case 1 for System testing... 23

Table 8 Test case 2 for System testing... 24

LIST OF ABBREVIATIONS

GPS: Global Positioning System

IDE: Integrated Development Environment

CHAPTER 1: INTRODUCTION

1.1 Introduction

The means of transportation in Kathmandu are microbus, minibus, taxi etc. Most of the people use public vehicle like microbus or minibus while some may prefer taxi. To move around a big city like Kathmandu, the most convenient and fastest way is boarding a microbus. Microbuses run around almost every lane that could be ridden on in the places in and around Kathmandu. The microbuses in Kathmandu Valley are operated and guided by "Yatayat Byawasayi Sangh" and all of the microbuses are listed under it as it is their umbrella organization that decides on the fares and routes of the microbuses.

Besides Microbus, Taxi is also preferred but it is more expensive and people could rarely afford it. There is a taximeter in each taxi which gives the cost of travel. It calculates the fare based on the distance travelled and waiting time of the total journey. The passenger can travel directly using the taximeter or by making the deal with the taxi driver about the fare before the journey begins.

"Mobile application to auto calculate taxi and microbus fare" is an android application which gives the travel fare in public vehicles of Kathmandu Valley. It is applicable to both taxi and microbuses. All smart phones has inbuilt GPS service i.e. the GPS device is embedded inside the mobile phone. Hence this application uses that GPS to track the location of user and calculate the total distance covered and total fare. Google Maps will be used to trace the path of the user. The fare is calculated based on the standard rate provided by the department of transportation management so this application will give exact fare.

1.2 Problem Definition

Travelling these days has been more difficult in terms of financial perspective. The problem is that the travellers don’t know the exact fare to travel from one place to another. This causes the taxi driver and public vehicle to claim more money than the actual. So the ordinary people are compelled to pay more fare as they do not have other alternatives.

Department of Transportation Management has published a document stating the fare of travelling all over the Nepal including the taxi and public vehicles in Nepal. The fare is allocated based upon the fuel consumption and other factors, [1]. But People these days are less aware of it and pay as much as the driver or the conductor demands. Even the taxi owner outsmarts the normal people and the meter calculates the fare more than the specified. On the other hand it is not possible to take all documents and look at the fare each time a person travels. In a taxi and public vehicles inside Kathmandu the fare is calculated based on the distance in kilometer. So it is impractical to carry the distance measuring instrument while travelling.

Therefore if this problem can be solved in the form of mobile application than it would efficient and effective as mobile devices comes with inbuilt GPS which can measure the distance travelled and we can program the application to calculate the total fare from the total distance travelled.

1.3 Objective

This project has been developed with the following objective:

a) To make the travellers easy to figure out the fare while travelling from one place to another.

b) To make the Traffic Police Department easy to identify the fraud activities taking place in town.

c) To understand the way of solving problem through the use of smart phone and android programming.

1.4 Scope and Limitation

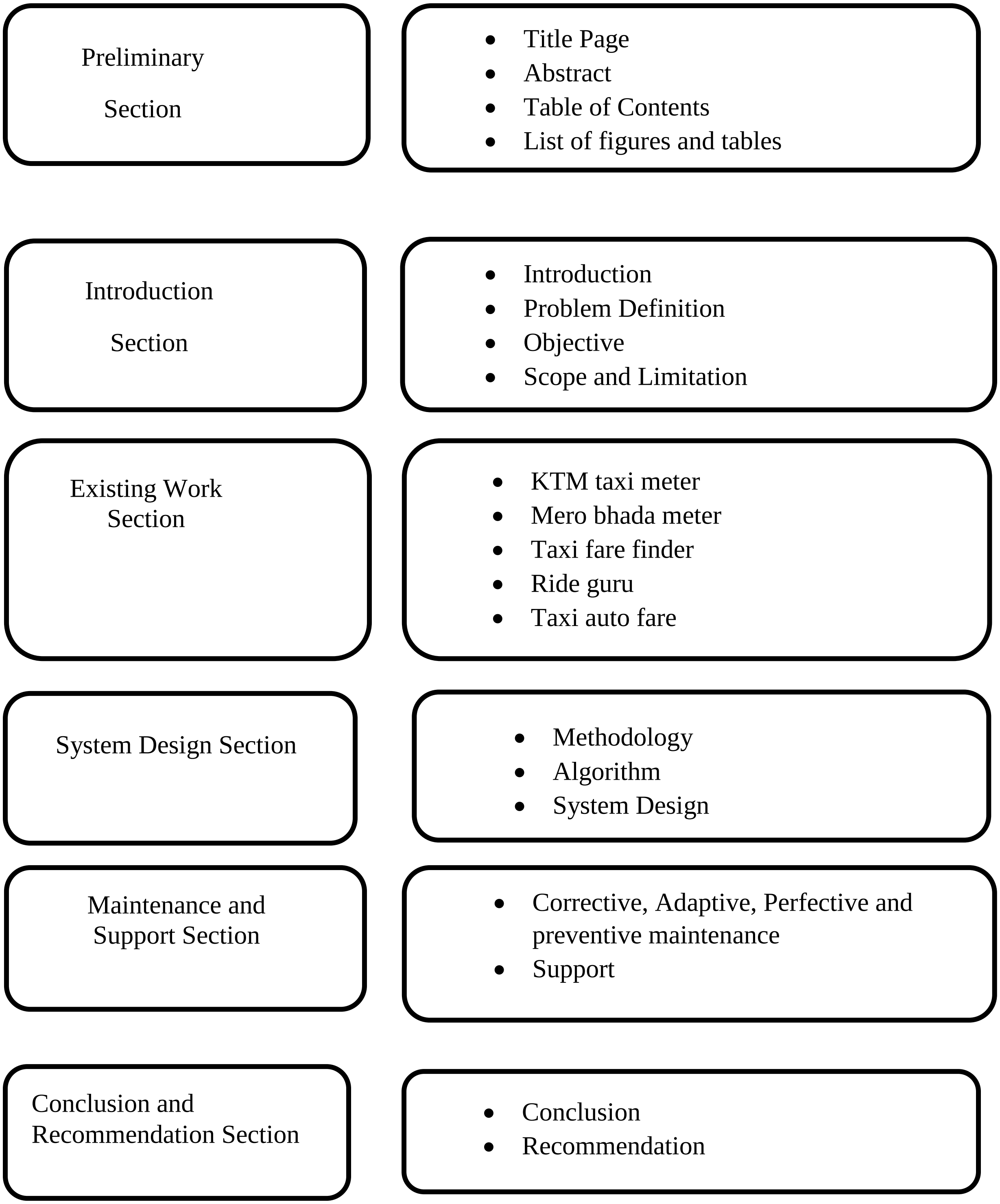
The users with android mobile device can use this application. As the number of android users are nearly triple than that of IOS users, this application covers more no of users.

This application will be very helpful to the travellers inside the Kathmandu valley as they don’t have to rely on the taxi meter which may not give exact fare. So they will only pay the exact amount of fare stated by the government.

The metropolitan traffic division will also be benefitted by this application as they don’t have to check the taxi meter manually. This application also has the feature of reporting police. If the taxi driver does not agree with the fare generated by the application, the traveller can report to the nearest traffic police station. This will help metropolitan traffic division to maintain unnecessary crackdown.

The only limitation of this application is that the Google map requires internet connection to load. Once the map gets loaded, then the application does not require any form of internet connection.

1.5 Outline of Document



Preliminary Section	• Title Page • Abstract • Table of Contents • List of figures and tables
Introduction Section	• Introduction • Problem Definition • Objective • Scope and Limitation
Existing Work Section	• KTM taxi meter • Mero bhada meter • Taxi fare finder • Ride guru • Taxi auto fare
System Design Section	• Methodology • Algorithm • System Design
Maintenance and Support Section	• Corrective, Adaptive, Perfective and preventive maintenance • Support
Conclusion and Recommendation Section	• Conclusion • Recommendation

Figure 1 Project Block Diagram

CHAPTER 2: REQUIREMENT ANALYSIS AND FEASIBILITY STUDY

2.1 Related Works

i) KTM taxi meter:

KTM taxi meter is an android application that gives the travel fare of only taxi. It uses GPS to track the location of the user and calculates the distance based on the coordinates obtained. It uses location manager (a package in Java to track the location of the user), [2]. It is the traditional way of tracking the location and does not give the exact location of the user. This application was developed by the four students of Kathmandu University.

In addition to the taxi fare it also shows the path of the user in which the user is travelling on with the help of Google maps. As the taxi moves the marker on the Google map also moves and the location is updated in a certain interval of time. After the user reaches to the destination, the user has to press the stop button. Than it shows the route summary (total distance, total charge, rate).

This application is only for taxi and does not work for other public vehicles.

ii) Mero bhada meter:

Mero Bhada Meter is an app that lets you track you taxi fare and gives you a chance to save yourself from taxi meter tampering. You can either travel with GPS tracking or you can calculate the fares by selecting the route in the map. Mero Bhada Meter works both online and offline.

It uses Open Street Map(OSM) and the user can download the map and use it offline as well. It was developed by Kathmandu living labs.

Same as KTM taxi meter, this application is also only for taxi and does not work for public vehicles.

iii) Taxi fare finder:

It is a web application that provides the taxi fare of most of the cities of the world. It provides the static taxi fare. The user has to enter the starting place and ending place manually and the application provides the total fare with traffic along with the total distance covered while travelling. Since it is not real time user has check the taxi fare in web before travelling. It also traces the travel path in Google maps.

iv) Ride guru:

It is also a web application that provides the fare same as Taxi Fare Finder does. It provides the fare when the user enters the starting and the ending place. It divides the fare on the following parameters.

a) Initial Charge
b) Metered Fare
c) Total amount the driver earns.

It also provides the taxi reservation features.

v) Taxi auto fare:

It is a web based application where the user has to enter the following parameters:

a) Country
b) City/Airport
c) From
d) To

Then this application displays the total fare along with the distance and Estimated Travel Time. It displays the static taxi fare of most of the cities in the world.

2.2 Requirement Analysis

2.2.1 Functional requirement:

The functional requirement is depicted through use case diagram. The user should be able to search for long route fare, view map and press start/stop journey button. When the user presses the start journey button, then the user should be able to view the current location in the map. Similarly when the user presses the stop journey button than the user should be able to view route summary. Finally the user should be able to report to the police in case of fraud activities.

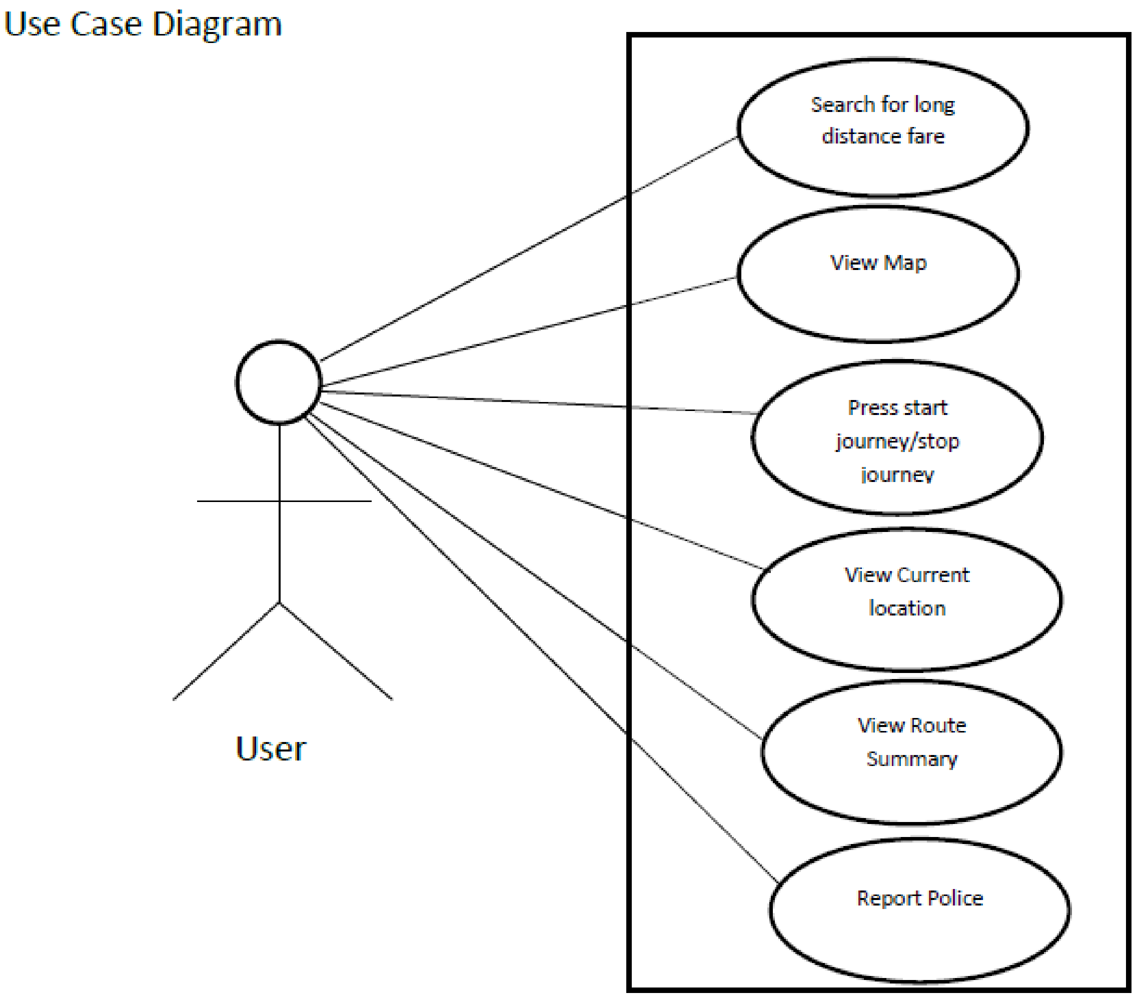


Figure 2 Use Case Diagram of the application

2.2.2 Non-functional requirement

i) Performance

The System is designed to give high performance.

ii) Accuracy

The system gives highly accurate result. The current location of the user tracked by the system is highly accurate as the system uses fused location API of Google instead of the traditional location manager. Fused location API gives the best current location based on the location provided by the GPS, cell phone tower and WIFI signals.

iii) Efficiency

The system is designed so that it consumes less resources of the mobile phone. It does not use unnecessary features so that the battery does not drain too fast. The GPS is used only when necessary. If the user has to view the static fare then it has nothing to do with the GPS. Hence, the GPS remains off at this time. So this application makes the use of resources only when it needed making the system more efficient.

iv) Reliability

Travel fare in the application will be updated as soon as the government updates the fare. Once the Google map gets loaded this application does not require internet connection.

2.3 Feasibility Analysis

2.3.1 Economic feasibility

This application is economically feasible as the GPS and Google play service location API are free of cost. This application is expected to generate more revenue as it will be launched in play store. Hence the investment will be less and the benefit will be more.

2.3.2 Technical feasibility

The available technology in the market will be sufficient to build this application. This application uses GPS and Google play service location API to track the location of users. Google maps will be used to trace the path of the user. This application will run on all android devices. Hence this application is technically feasible.

2.3.3 Schedule feasibility

The project deadline is very reasonable. It will take almost 10 weeks to complete this project. Other information about the schedule is depicted in Figure 3

Table 1 CPM table

Legend	**Activity**	**Duration**	**Precedence**
A	Requirement Analysis	1 Week	
B	Feasibility Study and Data Collection	1 Week	A
C	Coding	3 Week	A,B
D	Testing	2 Weeks	C
E	Debugging	1 Week	C,D
F	Documentation	2 Week	A,B

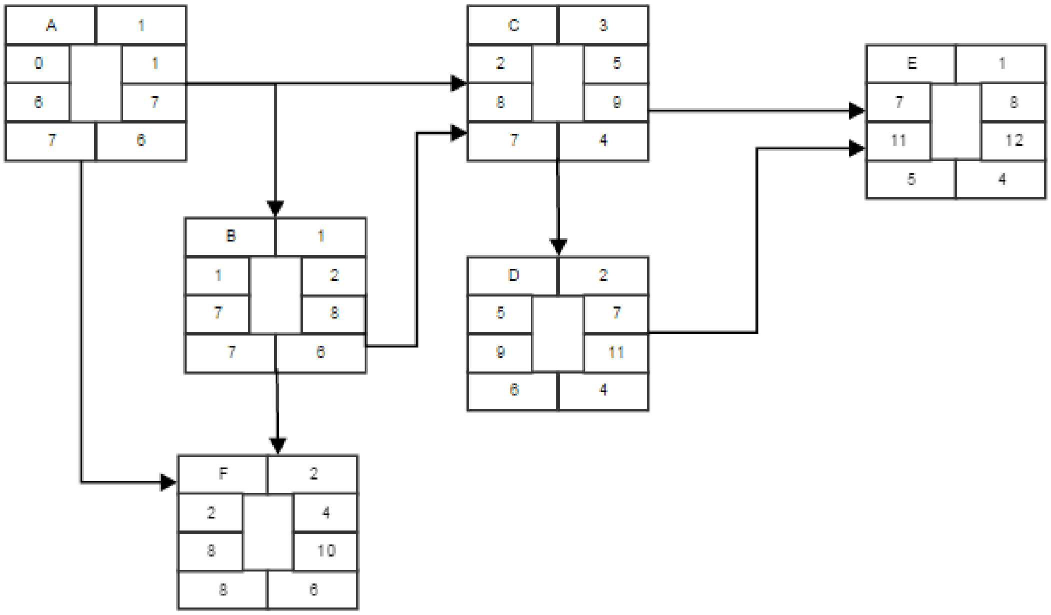


Figure 3 Critical path schedule

The above figure describes the project timeline. The Early finish time of the project will be 8 weeks whereas the late finish time of the project will be 10 weeks. The coding part of the project will take the longest time. Documentation part will go on side by side once Requirement Analysis and Feasibility Study is completed.

2.3.4 Operational Feasibility

User will be able to download this application from playstore for free. Hence more no of users are supposed to download this application.

CHAPTER 3: SYSTEM DESIGN

3.1 Methodology

Waterfall model is used to develop this system. This is because each phase must be completed before starting the next phase and there is a single person building this application. Hence Waterfall model is the best and cost effective choice to build this application.

3.1.1 Data collection

Data Collection is done via the rates published by Transportation Management Department. The fare was last updated on 2072-10-24 and will be updated as soon as the department updates it.

3.1.2 Algorithm

The algorithms used in this project are different for taxi and microbus

<u>Fare Calculation Algorithm for Taxi</u>

```
1)  Start
2)  totaldistance=0, totalfare=13, consdistance,consfare.
3)  Track the location of the user and plot it in google map i.e latlong1.
4)  While( !stop journey).
        i.   Track the new location of the user i.e latlong2.
        ii.  Calculate the distance covered from latlong1 and latlong2 i.e
             consdistance.
        iii. Add the consdistance to totaldistance i.e.
             totaldistance+=consdistance.
        iv.  Compute the consfare as.
                 consfare= (rate/1000)*consdistance
        v.   Add the consfare to total fare
                  totalfare+=consfare.
        vi.  While(!6seconds)
                  1.   Sleep
        vii.  Go to step 4.
5)   End
```

Fare Calculation Algorithm for Microbus

```
1)   Start
2)   totaldistance=0, totalfare=13, consdistance.
3)   Track the location of the user and plot it in google map i.e. Latlong1
4)   While(!stop journey)
        i. Track the new location of the user i.e Latlong2
        ii.  Calculate the distance covered from latlong1 and latlong2 i.e.
           consdistance.
        iii. Add the consdistance to totaldistance  i.e
           totaldistance+=consdistance.
        iv.  if(totaldistance>4km and totaldistance<= 5km)
                  i.   totalfare+=1
        v.else if(totaldistance>5km and totaldistance<=6km)
                  i. totalfare+=1
        vi.  else if(totaldistance>6km and totaldistance<=8km)
                  i. totalfare+=1
        vii. else if(totaldistance>8km and totaldistance<=10km)
                  i. totalfare+=2
        viii. else if(totaldistance>10km and totaldistance<=13km)
                  i.   totalfare+=2
        ix.  else if(totaldistance>13km and totaldistance<=16km)
                  i.   totalfare+=2
        x.else if(totaldistance>16km and totaldistance<=19km)
```

```
                i.   totalfare+=1
          xi.  else if(totaldistance>19km)
                i.   totalfare+=1

           xii. while(!6seconds)

                a. sleep

           xiii go to step 4

      7. End
```

Algorithm to calculate distance from two consecutive Latitudes and Longitudes.

Haversine formula can be used to calculate the great-circle distance between two points – that is, the shortest distance over the earth’s surface, [3].

```
1)   Start
2)   Lat1,Lat2,Lon1,Lon2
3)   Convert lat1,lat2,lat3,lat4 to radians(i.e. initially they are in degree). Let
they be dlat1,dlat2,dlon1,dlon2.
4)   Initialize the value of R = 6371 (i.e. R is the radius of the earth in KM).
5)   Compute the difference of latitude and longitude as.
          i.   dflat = dlat2-dlat1
         ii.   dflon = dlon2-dlon1
6)   Compute a as

          $a = \sin(dflat/2) * \sin(dflat/2) + \cos(dlat1) *\cos(dlat2) + \sin(dflon/2) * \sin(dflon/2)$

7)    Compute c as
          $c = 2 \cdot \operatorname{atan2}( \sqrt{a}, \sqrt{(1-a)} )$
8)   Compute Distance(d) as
          $d = c * R$
9)   Return  d
10)  End
```

Conversion from Degree to Radians.

Radian = Degree * (π /180)

3.2 System Design

3.2.1 Class diagram

Each class in android programming is represented by an activity as shown in Fig. 4. There are altogether six activities. The methods and variables in activities and their interaction with other activities is described below in the class diagram. When the user first enters into the application, main activity gets started. Then main activity starts different activities according to the type of vehicle selected.

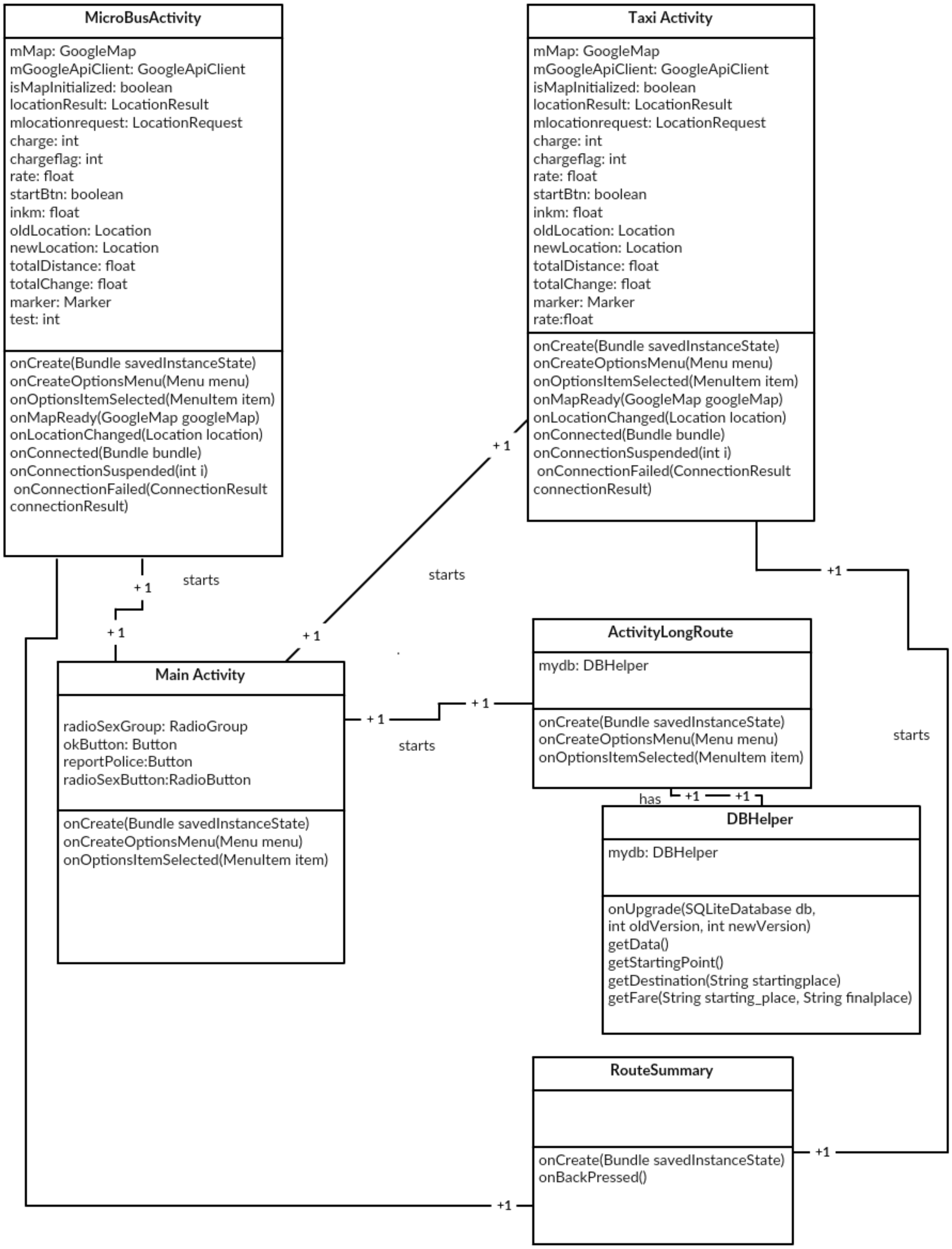

MicroBusActivity
mMap: GoogleMap
mGoogleApiClient: GoogleApiClient
isMapInitialized: boolean
locationResult: LocationResult
mlocationrequest: LocationRequest
charge: int
chargeflag: int
rate: float
startBtn: boolean
inkm: float
oldLocation: Location
newLocation: Location
totalDistance: float
totalChange: float
marker: Marker
test: int
onCreate(Bundle savedInstanceState)
onCreateOptionsMenu(Menu menu)
onOptionsItemSelected(MenuItem item)
onMapReady(GoogleMap googleMap)
onLocationChanged(Location location)
onConnected(Bundle bundle)
onConnectionSuspended(int i)
onConnectionFailed(ConnectionResult connectionResult)
Taxi Activity
mMap: GoogleMap
mGoogleApiClient: GoogleApiClient
isMapInitialized: boolean
locationResult: LocationResult
mlocationrequest: LocationRequest
charge: int
chargeflag: int
rate: float
startBtn: boolean
inkm: float
oldLocation: Location
newLocation: Location
totalDistance: float
totalChange: float
marker: Marker
rate:float
onCreate(Bundle savedInstanceState)
onCreateOptionsMenu(Menu menu)
onOptionsItemSelected(MenuItem item)
onMapReady(GoogleMap googleMap)
onLocationChanged(Location location)
onConnected(Bundle bundle)
onConnectionSuspended(int i)
onConnectionFailed(ConnectionResult connectionResult)
+ 1
+ 1
starts
starts
+ 1
+ 1
+1
Main Activity
radioSexGroup: RadioGroup
okButton: Button
reportPolice:Button
radioSexButton:RadioButton
onCreate(Bundle savedInstanceState)
onCreateOptionsMenu(Menu menu)
onOptionsItemSelected(MenuItem item)
+ 1
+ 1
starts
ActivityLongRoute
mydb: DBHelper
onCreate(Bundle savedInstanceState)
onCreateOptionsMenu(Menu menu)
onOptionsItemSelected(MenuItem item)
starts
has
+1
+1
DBHelper
mydb: DBHelper
onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
getData()
getStartingPoint()
getDestination(String startingplace)
getFare(String starting_place, String finalplace)
RouteSummary
onCreate(Bundle savedInstanceState)
onBackPressed()
+1
+1

Figure 4 Class diagram of the application

3.2.2 Event diagram

The series of event that may occur while the user is navigating through this app is described below. At first the user can choose the type of vehicle. The user can choose either taxi, microbus or long route bus. The user reaches to different pages according to the type of vehicle. The event diagram of the application is shown in Fig 5. The user can press start journey button when the user select taxi or microbus. The fare is calculated than according to the type of vehicle. User has to enter the initial and final destination if the long route bus is selected by the user. Finally the long route fare is displayed to the user.

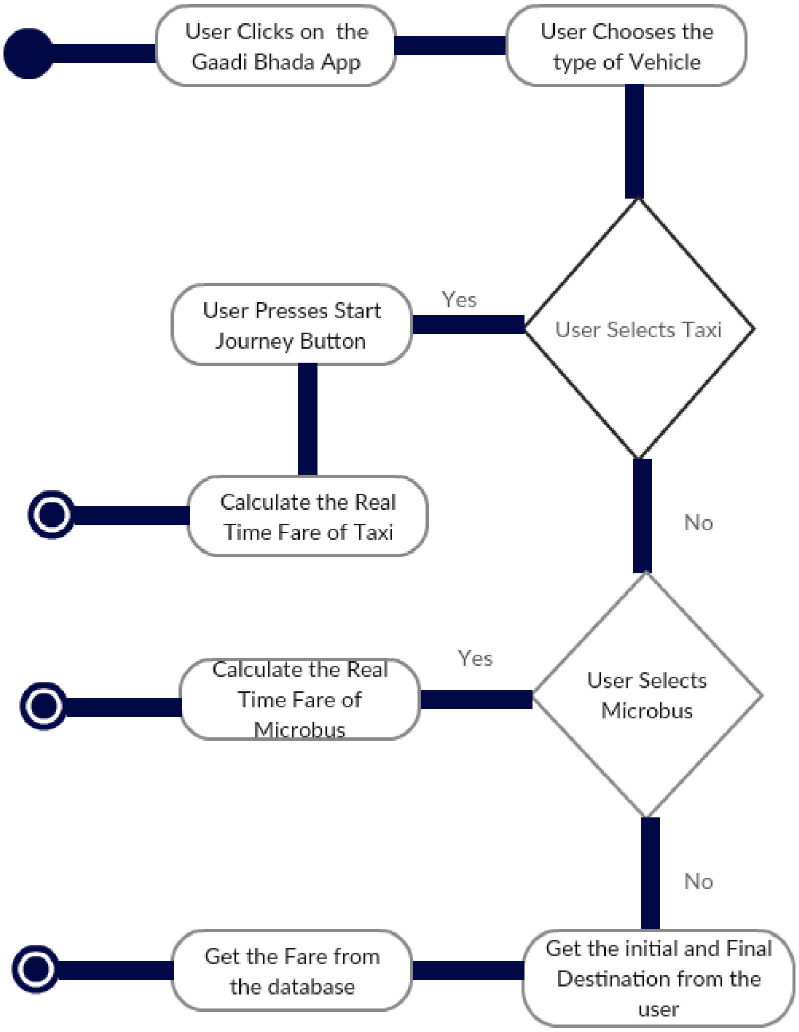

User Clicks on the Gaadi Bhada App
User Chooses the type of Vehicle
User Selects Taxi
Yes
User Presses Start Journey Button
Calculate the Real Time Fare of Taxi
No
User Selects Microbus
Yes
Calculate the Real Time Fare of Microbus
No
Get the initial and Final Destination from the user
Get the Fare from the database

Figure 5 Event diagram of the application

3.2.3 Sequence diagram

The sequence of actions that the user generates is depicted through the sequence diagram as shown in Fig 6. The application than also generates the action responds to the user action. The user may press start journey button for taxi. The application then requests for current location to the GPS. The GPS returns the current location of the user. The application than calculates the real time fare and returns to the user. Similarly the user requests for real time microbus fare and static long route fare and gets the fare accordingly.

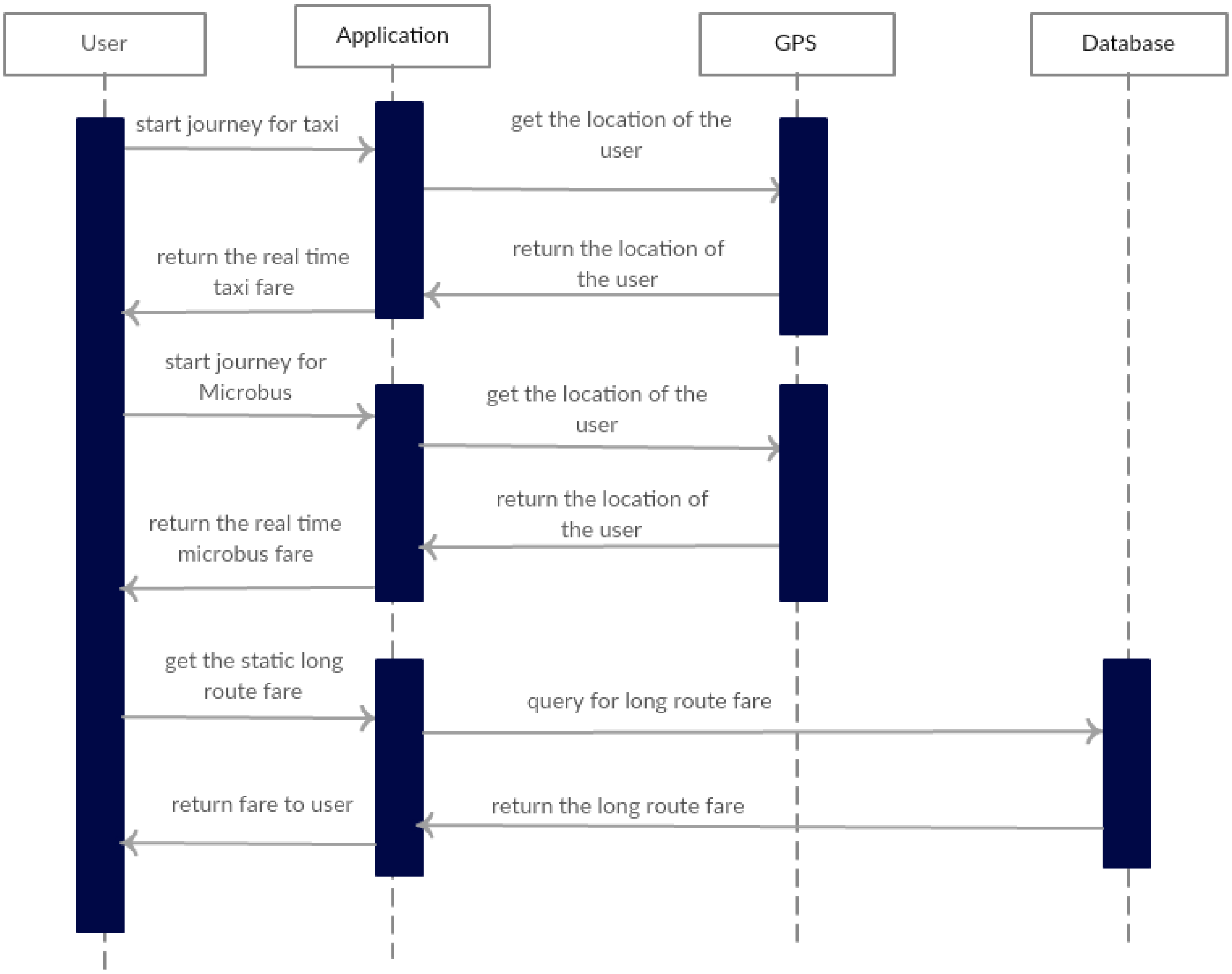

User
Application
GPS
Database
start journey for taxi
get the location of the user
return the location of the user
return the real time taxi fare
start journey for Microbus
get the location of the user
return the location of the user
return the real time microbus fare
get the static long route fare
query for long route fare
return the long route fare
return fare to user

Figure 6 Sequence diagram of the application

CHAPTER 4: IMPLEMENTATION AND TESTING

4.1 Implementation

This project is implemented in java. Since it is a mobile application and require some core device functionality like GPS and phone call, cross platform tools which are used to develop mobile application are not used. The device uses GPS to track the location. The location is tracked every 6 seconds and the distance is calculated based upon the consecutive location.

4.1.1 Tools used

i) Android Studio

Android studio is used as an IDE to develop this application. It provides code editing, debugging, performance tooling, a flexible build system and an instant build/deploy system which allows to help building high quality apps.

ii) Java programming

This application is written in java since android uses java as a only programming language to build its application, [4]. There can be other cross-platform tools but it is not appropriate to make application which requires real time tracking. All the designs are made in xml file provided by android project.

iii) GPS

It is used to track the location of the user. Almost all the smart phone these days comes with a GPS. The user must open a GPS to use this application.

iv) Fused Location API

This application use Google Play Service Location API (i.e. fused location API) provided by Google to track the location of the user. It is the latest location API of Google. Previously there was a location manager i.e. a java package to track the location of the user. The Google

Play services location APIs are preferred over the Android framework location APIs (android.Location) as a way of adding location awareness to your app. If you are currently using the Android framework location APIs, you are strongly encouraged to switch to the Google Play services location APIs as soon as possible, [5].

v) SQLite Storage

The static long route fare is stored in sqlite storage. Since the user has to search for the fare in a location where there may not be internet connection, sqlite storage is preferred over storing the data in web.

4.1.2 Listing of major classes and methods

The main classes of this application are

i) TaxiActivity Class

The main methods inside the taxi activity class are:

i) OnCreate
ii) OnMapReady
iii) OnConnected
iv) OnLocationChanged

ii) MicroBusActivity Class.

The main methods inside the microactivity class are similar to taxi activity class. The only difference is core logic for calculating fare.

iii) ActivityLongRoute Class.

This class gives the fare of long route vechicle. The main method inside this class is:

i) OnCreate

4.2 Testing

4.2.1 Unit testing

Table 2 Test case 1 for unit testing

Test Case No.	1
Title	Track the Location of the user.
Precondition:	The GPS of user's device must be open.
Test Steps	i. Open the application. ii. Click on start journey button. iii. Note the latitude and longitude. iv. Go to the google map and check whether the latitude and longitude match with the latitude and longitude given by the app.
Expected Outcome	Latitude: 27.7123902, Longitude: 85.3425778
Actual Outcome	Latitude: 27.7123902, Longitude: 85.3425778

Table 3 Test case 2 for unit testing

Test Case No	2
Title	Calculate the distance between two places.
PreCondition	The GPS of user's device must be open.
Assumption	The user knows the actual distance between those two places.
Test Steps	i. Open the application. ii. Go the initial place. iii. Open the start journey button. iv. Move from that place to final destination. v. See whether the distance increases or not. vi. Click on the stop journey button once you reach to the final destination. vii. Note the total distance.

	viii. Match the total distance with the actual distance.
Expected Outcome:	Total Distance = 750 m
Actual Outcome	Total Distance = 750 m

Table 4 Test case 3 for unit testing

Test Case No.	3
Title	Plotting the location in google map.
Precondition	The mobile device has internet connection of any form and the GPS of mobile device is on.
Assumption	User knows the place from where the location is tracked and ploted.
Test Steps	i. Open the application. ii. See the marker in the map once the map gets loaded. Iii. Note the position of the marker
Expected Outcome	Marker must be at Latitude: 27.7123902 and Longitude：85.3425778
Actual Outome	Marker was at Location: 27.7123902 and Longitude: 85.3425778

4.2.2 Integration testing

Table 5 Test case 1 for Integration testing

Test Case No	1
Title	get the static long route fare.
Precondition	The final destination corresponding to the initial destination must be in the database.
Assumption	The fare provided is based upon the last updated fare by the

	transportation management department.
Test Steps	i.Open the application. ii.Choose the vechicle type as static long route fare. iii.Click the OK Button. iv.The page with the heading Long Route Bhada will appear. v. Enter the Starting Place and Final Destination. vi. Click on Get Fare button. vii. Note the fare and total distance.
Expected Outcome	Initial Destination: Kathmandu Final Destination: Gaur Total Fare: Rs. 592
Actual Outcome	Initial Destination: Kathmandu Final Destination: Gaur Total Fare: Rs 592

Table 6 Test case 2 for Integration testing

Test Case No	2
Title	Transition Between the activities of application.
Precondition	All the buttons of the mobile devices should work properly
Assumption	The user has idea about the transition between the activities of the application.
Test Steps	i. Open the application. ii. Choose any of the vehicle type. iii. Click OK button iv. Note which activity gets started. v. Press the back button vi. Choose another vehicle type.

	vii. Click OK button. viii. Press back button. ix. Choose the last Vehicle type. x. Click OK button. xi. Click back button. xii. Click Home button.
Expected Outcome	The transition between the activities should be fast and proper. The back button should work properly and take the user to lastly navigated activity.
Actual Outcome	The transition between the activities was fast and proper. The back button worked properly and took the user to lastly navigated activity

4.2.3 System testing

Table 7 Test case 1 for System testing

Test Case No.	1
Title	Testing of taxi fare
Precondition	The user must book a taxi and go from one place to another.
Assumption	The GPS provides the exact location of the user.
Test Steps	i.Open the application. ii.Choose the vehicle type as taxi. iii.Click OK button iv.Book a taxi and get into it. v.Press the start journey button. vi.See the meter of the taxi and match it with

	fare generated by the application vii.Press the stop journey button once you reach to the destination. ix. View the root summary
Expected Outcome	Total Distance Covered = 2km Total Fare = 86
Actual Outcome	Total Distance Covered = 2.2km Total Fare = 93.2

Table 8 Test case 2 for System testing

Test Case No	2
Title	Test the microbus fare
PreCondition	The user must travel from one place to another using microbus.
Assumption	The GPS provides the exact location of the user.
Test Steps	i. Open the application. ii. Choose the vehicle type as Microbus. iii. Click OK button iv. Get into the microbus. v. Press the start journey button. vi. Press the stop journey button once you reach to the destination. vii. View the root summary
Expected Outcome	Total Distance Covered = 4.5km Total Fare = 15
Actual Outcome	Total Distance Covered = 4.6 km Total Fare = 15

CHAPTER 5: MAINTENANCE AND SUPPORT

5.1 Maintenance

5.1.1 Corrective maintenance

This application will be kept in play store and once the user reports for a bug than the bug will be fixed right away and the application will be updated in play store as soon as the as the bug is fixed.

5.1.2 Adaptive maintenance

The standard fare for taxi, microbus and long route bus is changed constantly. The fare is updated as soon as the transportation management department updates the fare. The application will be than updated in the play store.

5.1.3 Perfective maintenance

The performance of the system will be improved constantly. The performance problem will be identified and fixed. The application will be improved constantly so that it will use minimum resources and gives maximum performance.

5.1.4 Preventive aintenance

The possible errors that can occur in future will be predicted and the system the possible strategy to cope with the future error will be implemented.

5.2 Support

This project will be submitted as a proposal to Metropolitan Traffic Police. It can be further enhanced if Metropolitan Traffic police approves this project. Furthermore, this project will also be submitted to department of transportation management to make this application authorized for all the travel fare across the country.

CHAPTER 6: CONCLUSION AND RECOMMENDATION

6.1 Conclusion

People are overcharged by the taxi and microbus these days. The long route bus also demands more fare than the actual. People just tolerate it and most of them don’t complain about it. Hence this application will be somehow helpful for all the people travelling inside Kathmandu valley and across the country. This application warns the vehicle drivers and conductors and gives more control to the passenger over the fare. It also helps metropolitan traffic police so that they don’t have to check the taxi meter manually.

6.2 Recommendation

This application can be further enhanced to book the taxi or get the ticket of long route bus. Also the fare to be paid to the taxi driver or the bus conductor can be automated and make it pay online through different online payment medium like e-sewa.

REFERENCES

[1] “निर्धारित भाडादर”, Dotm.gov.np, 2016. [Online]. Available: http://www.dotm.gov.np/current-fare/. [Accessed: 05- June- 2016].

[2] 2016. [Online]. Available: http://flipkarma.com/media_dir/main_documents/merged_document.pdf,. [Accessed: 08- June- 2016].

[3] Latitude/Longitude Distance Calculation", Nau.edu, 2016. [Online]. Available: http://www.nau.edu/~cvm/latlongdist.html. [Accessed: 23- June- 2016].

[4] Java Platform Standard Edition 7 Documentation", Docs.oracle.com, 2016. [Online]. Available: http://docs.oracle.com/javase/7/docs/. [Accessed: 12- June- 2016].

[5] Making Your App Location-Aware | Android Developers", Developer.android.com, 2016. [Online]. Available:https://developer.android.com/training/location/index.html. [Accessed: 12- June- 2016].

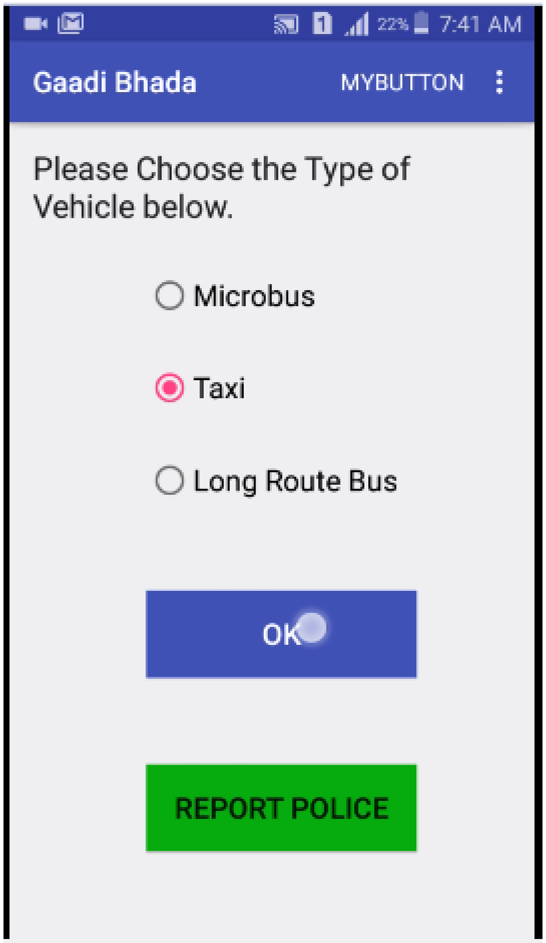
BIBLIOGRAPHY

Dietel Paul, Dietel Harvey,2012, *Java How to Program,* Ninth Edition, PHI Learning Private Limited, New Delhi-110001

Annuzzi Joseph Jr, Darcey Lauren, Conder Shane, 2014, *Introduction to Android Application Development*, Fourth Edition, Addison-Wesley Professional

APPENDIX

Landing Page



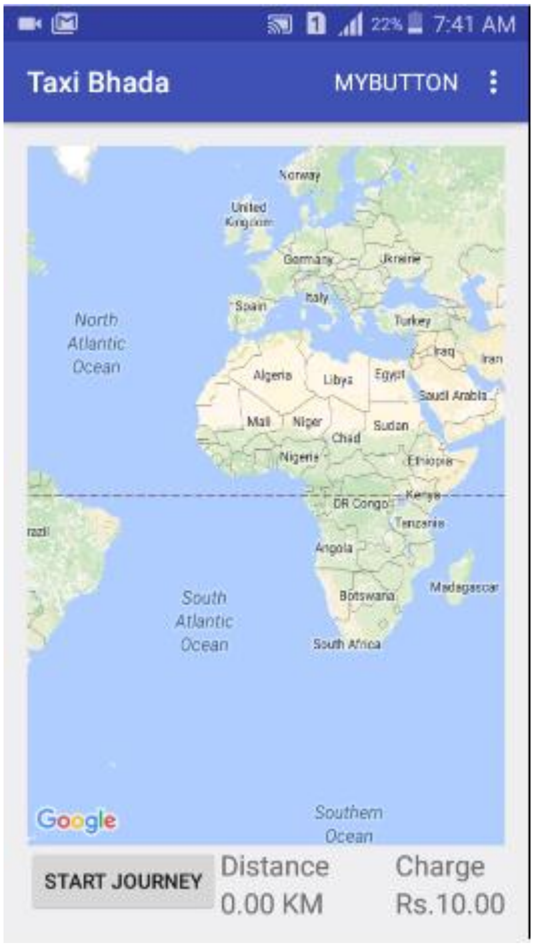
Please Choose the Type of Vehicle below.

○ Microbus

● Taxi

○ Long Route Bus

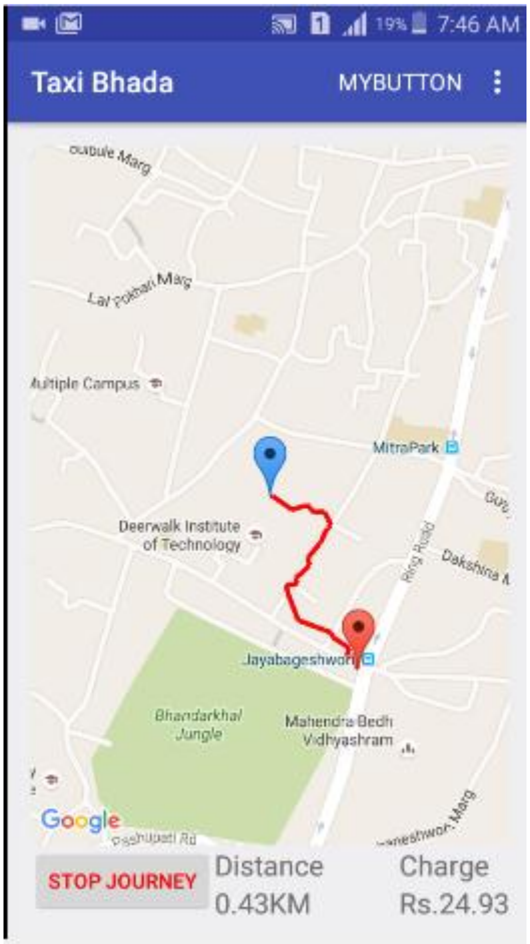
Taxi Bhada Page



Distance 0.00 KM

Charge Rs.10.00

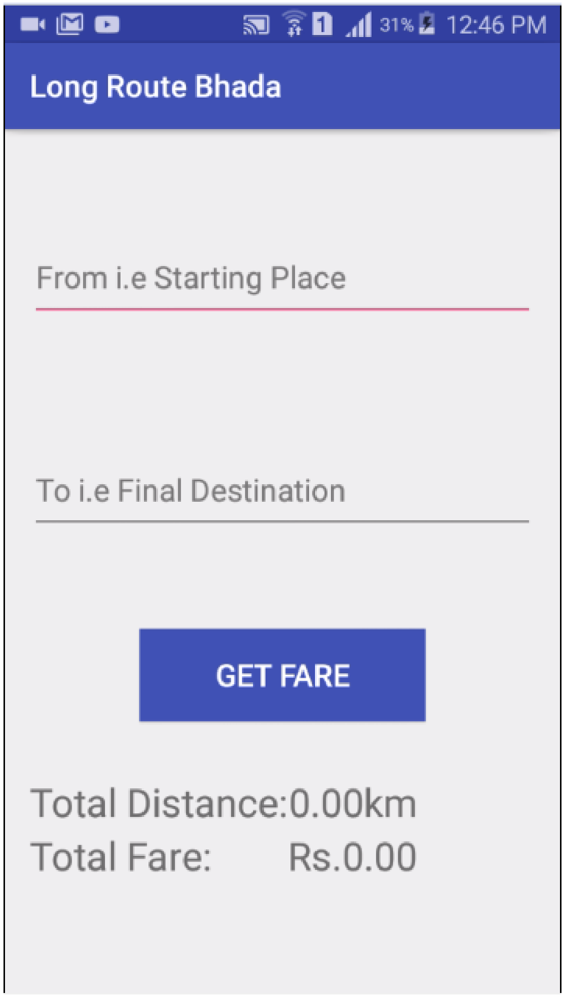
Taxi Bhada Calculation in real time



Distance 0.43KM

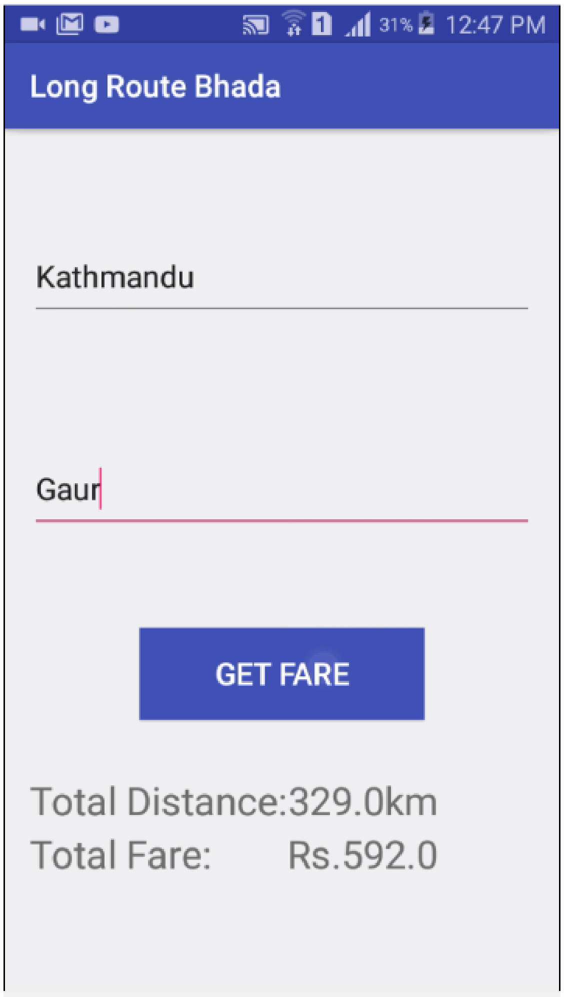
Charge Rs.24.93

Long route Fare Calculation



Total Distance:0.00km
Total Fare: Rs.0.00

Displaying Long Route Fare



Kathmandu

Gaur

Total Distance:329.0km
Total Fare: Rs.592.0