

ET 02/85
IT 01/113

LB/DOON/27/2016

Mobile Application for Locate and Contact Nearest Taxies via GPS

LIBRARY
UNIVERSITY OF MORATUWA, SRI LANKA
MORATUWA

Dinesh Priyankara Samarasekara
(129166D)

Dissertation submitted in partial fulfillment of the requirements for the degree
Master of Science in Information Technology

Faculty of Information Technology

University of Moratuwa

Sri Lanka

004 "15"
004 (043)

May 2015

109934



University of Moratuwa



109934

DVD (109933-45)

TH3032

109934

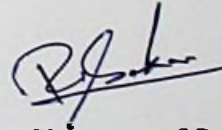
DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name of Student

Dinesh Priyankara Samarasekara

Date: 2015/06/04



Signature of Student

Supervised by

Mr. Saminda Premarathna

Date: 04/06/2015

UOM Verified Signature

Signature of Supervisor

ACKNOWLEDGEMENT

It is my great pleasure to express my affectionate and deeply felt gratitude to Mr, Saminda Premarathne, Senior Lecturer, University of Moratuwa. This dissertation would not have been possible without his guidance, invaluable suggestions and constant inspiration. His patience in reading, correcting and refining this work is commendable.

I am more than grateful to all my lecturers for their knowledge and support shared with me. I would like to thank Mr. Vipula Anandapiya and Mr. Udantha Diyanath Pathirana, Senior Software Engineers at CMS (pvt) Ltd. I express my warm thanks to all the authors who have contributed their time and expertise for their previous studies. And also I would like to thank all the non-academic staff of Mathematical Department of University of Moratuwa for their support and services.

Finally, special thanks go to my loving parents for their dedication, patience and faith shown on me.

ABSTRACT

It is extremely difficult to find a taxi in Colombo when a person needs a taxi. When a person needs a taxi, there are multiple of ways to find a taxi; (i) passenger can reserve cab on the web (ii) passenger can phone for a taxi. Of the above most of the time the easiest and convenient way is phone for a taxi. When the passenger phone for a cab on peak times, the cab services would not be responded or they don't have free taxis around, while more other individual taxi drivers are available around the passenger.

Therefore, the objective of this mobile taxi tracking system is to target those individual taxi drivers through their embedded taxi meter. Solution will be based on GPS navigational data and GPRS communication technology. This GPS taxi meter will be connected to a central web server with a unique taxi id and location. Web server will have a separate account for a taxi associated with contact numbers and rates. When a passenger searches for a taxi, the web server will filter available taxis around the passenger and fetch to the Google map interface on the user mobile. The driver rating system enables user to rate the taxi and the hire which highlight the high rates taxis when others search on map. Application will be based on Android mobile operating system, so it will be simpler and user friendly.

Furthermore, target is to reach all individual taxies as well as taxi companies which handling more than one. While using definitely the application will with their new requirements and also they will get better opportunity to serve their customers. In other hand customers also will gain the benefit of the user friendliness and the new features of the solution while using it for their day today taxi needs.

TABLE OF CONTENTS

	Page
Chapter 1 – Introduction	01
1.1 Introduction	01
1.2 Aims and Objectives of the Project	01
1.3 Proposed Approach	02
1.4 Scope	02
1.5 Current System	02
1.6 Problems/Issues of the current system	03
1.7 Methods of addressing the current problems	04
1.8 Summary	05
Chapter 2 – Current Trends of the GPS Taxi Tracking Applications	06
2.1 Introduction	06
2.2 Existing Systems	06
2.2.1 Open GTS	06
2.2.1 Yellow Taxi	08
2.2.2 Online Cabs – taxi Sri Lanka	09
2.2.3 Find Taxi	10
2.2.4 Call Taxi	11
2.2.5 Easy Taxi – Taxi Cab App	11
2.3 Feature Summary and Comparison	12
2.4 Summary	13
Chapter 3 – Technology Adapted and Approach	14
3.1 Introduction	14
3.2 Communication with GPS/GPRS	14

3.3 Roll of the Web Server	14
3.4 Web API connects clients and the server	15
3.5 Android Application	15
3.6 Summary	15
Chapter 4 –Analysis and Design	16
4.1 Introduction	16
4.2 Software requirement specification	16
4.2.1 Functional requirements	16
4.2.2 Non-Functional Requirements	17
4.3 Application Design	18
4.4 Design Methodology	20
4.5 Proposed Architecture	20
4.6 Why Agile Scrum	24
4.7 Modules of the Solutions	25
4.7.1 User Module	25
4.7.2 Taxi Module	25
4.7.3 Hire Module	25
4.7.4 Ratings Module	25
4.8 Database Design	25
4.9 Summary	26
Chapter 5 – Implementation	27
5.1 Introduction	27
5.2 Software	27
5.2.1 Android SDK	27
5.2.2 PHP 5	27

5.2.3 My Sql 5	27
5.2.4 REST API	28
5.2.5 Meteor JS	28
5.2.6 Mongo DB	28
5.3 Implementation	29
5.3.1 Android Application	29
5.3.2 Data Handling and Synchronization	29
5.3.3 Google Map Integration	30
5.3.4 Back End Application	30
5.3.5 API	30
5.3.6 Use of Simulator instead of Actual Hardware	31
5.3.7 Summary	31
Chapter 6 – Conclusion	32
6.1 Introduction	32
6.2 Evaluation hierarchy	32
6.3 Evaluation of Hardware	32
6.4 Proving the accuracy of the simulator	33
6.5 Evaluation of the backend software	33
6.6 Evaluation of the Application	33
6.7 Evaluation of the User friendliness	34
6.8 Summary	35
Chapter 7 – Evaluation	36
7.1 Introduction	36
7.2 Competiveness	36
7.3 Future works	36

7.4 Conclusion	36
7.4.1 Problem in brief	36
7.4.2 Level of success of the proposed solution	37
7.5 Summary	37
References	38
Appendix A	39

LIST OF FIGURES

	Page
Figure 5.1: Context diagram of the main application	19
Figure 5.2: Level 0 diagram of the main application	21
Figure 5.3: Class Diagram	22
Figure 5.4: Use Case Diagram	23
Figure 5.5: Activity Diagram	24
Figure 5.6: Agile Scrum System	25
Figure 5.7: Database design ER (Entity Relation) Diagram	27
Figure 6.1: Meteor Main Configuration	31
Figure 6.2: User Register Template	32
Figure 6.3: Register Functionality	33
Figure 6.4: Google Map Configuration	35
Figure 7.1: End User Feedback Results	39



ACRONYMS AND DEFINITIONS

GPS	Global positioning system
OS	Operating System
GPRS	General Packet radio service
SMS	Short Message Service
API	Application Processing Interface
REST	Representational state transfer
ER	Entity Relation
SDK	Standard Development Kit
IDE	Integrated Development Environment
RDBMS	Relational Database Management System

CHAPTER 1

INTRODUCTION

1.1 Introduction

Chapter one describes the Problem, Aims and Objectives of the project, proposed approach, and scope of the project. It also describes the current system, problems and issues of the current system and the ways of solving those problems.

Now a day's people use taxies for their day to day works rather than using their own vehicles. Expensiveness and the heavy traffics on the roads are the main reason for this. But we have to wait a long time when we contact a taxi using popular taxi numbers, until they reach to us from their current location. In some cases, when taxi reaches to the customer location, he already left the place. So taxi has to bare the lost. Some cases there are no known taxies in that area when customers contact the, while there are some other unknown taxies waiting for hires.

In other hand, in urban areas out of five people, one has a smart phone which supports modern technologies like 3G or 4G and GPS Navigation. Most of those smart phones are on IOS or Android OSs. These smart phones commonly have several types of applications to support user's day to day life. Calculators, image editors, community developments, weather forecasts, sports and fitness and travel supporters are few of them. There are few applications in the market which supports user's travel needs.

1.2 Aims and Objectives of the Project

The aim of this solution to develop an application with better capabilities to address all mentioned issues of the current system. The main objective of the project is to find a solution which a user can find a taxi using his hand hold device, searching the nearest locations while assuring his security. To save the user's time and money also is an objective of the project. In addition to these, the solution will be more user



friendly that can be used by any individual on any age limit without any difficulties. And also it will create a base for more future developments.

1.3 Proposed Approach

Most of the taxies have a digital taxi meter. The solution is to couple a small gadget with a GPS navigator to the Taxi meter. So it will transfer the status of the taxi with the GPS navigator id to a web server which holds all the registered taxi details. The web server will get the taxi id passed through GPRS and collect relevant taxi details which include its contact and ratings details. Then the web server sends those details to the users device through his data connection and shows up a friendly icon of a taxi which has a pop up message box includes all the taxi details and ratings. So the user can tap on that message box to contact the taxi and start his/her journey. This kind of GPS tracking systems are uses for fleets management systems [1] TripLog , [2] Scania Fleet Management to track the locations and roots of the fleets.

1.4 Scope

The solution will only track the nearest taxies through GPS technology and will show up them to user. So the user can view taxi details, rate a given taxi up on his interest. Contact the taxi through voice or text and confirming it will be out of the scope of the solution. For this user can user his/her hand hold mobile device, but it's not the part of the solution.

1.5 Current System

Total population of the western province is 5,821,710. It has three districts and 7major cities and towns [20]. In other words most of the areas of western province are now developed and people are used to use taxies for their day to day works. According to the report of the central bank 2013, usage of the taxies has increased by 30%.

In Sri Lanka there are three main transport methods. Those are general bus service, trains and taxi services. The first two methods are common and only can travel

between two main cities. But the problem is we need to travel more to reach our destination which buses or trains are cannot be used. In this point we need the third transport method. That is the taxi service. This is a private one and can be used for personal destinations.

Almost all towns there is a bunch of taxies around and most the people use those taxies to complete their travel destinations. There are two main current systems of using taxies.

1. Owned by individuals

In this section, there are two main types of taxies.

- a. Taxies with a digital meter
- b. Taxies which doesn't have a meter

2. Owned by private companies

In this section always taxies are coupled with a digital meter.

At the moment in both above cases, people use taxies and pay the fee at the end of the journey. But it's more convenient to both parties to calculate the fee when there is a taxi meter attached.

1.6 Problems/issues of the current system

There is couple of identified problems in the current system which leads to major problems as well as wastages of time and money. These problems can be categorized as follows:

1. Contacting a taxi
2. Selecting a better and secure taxi
3. Paying the taxi fee

The first issue arrives at late hours and most busy times. That is, people have to wait for long time to catch a taxi if all taxies are busy at the main town area. Bust still there can be free taxies around waiting for customers at inner lanes. So the taxi

drivers don't know that there are customers at the main road who searching for them, in other hand customers or the users also don't know that there are free taxies nearby. Also people only know few taxi contact numbers and when those few are busy or occupied, customers have to wait until they find a free taxi or use other way.

Second issue is the most critical one since it could be harm to the life of the customer if he selects a wrong taxi. These kinds of incidences are happened several times and there should be a safer way to use taxies, especially in late hours and long distance travels.

Thirdly both customers and taxi drivers face difficulties when it's come to end of the journey. Some customers are used to say that they usually travels the same distance for less fee and also some taxi drivers used to ask for unreasonable fees. There can be unpleasant scenarios during this kind of occasions and sometimes they'll end with at the nearest police station.

1.7 Methods of addressing the current problems

Currently there's no standard way to solve the identified issues. To overcome the taxi selection issue some taxi companies formed there named taxies. People used to contact these taxies and seem they believe on these. But this solution cannot be considered as a 100% secured way. There is no any solution to track the exact location of the taxi in case of hijacking or noncontact able locations.

Also when using taxies on late hours and selecting taxies for long distance travels both customer and the taxi driver has to face unsecure situations, especially when women's using taxies. There is no finalized way to overcome this issue as well. If there was a solution to track the location of the taxi, at any given time the taxi company or the relations of the customer can be aware of the journey.

The digital taxi meter is a grate solution for calculating the taxi fee. Both customer and the taxi driver can be agreed up on this and it will solve most of the conflicts. But some taxi drivers used to hack the digital meter and insert unreasonable fees.

Customer has to pay more in this kind of situations. If there is a way to avoid this kind of situation, customers can use metered taxies with more secured way.

1.8 Summary

Chapter 01 described the background of the problem and the brief description of the problem. It also described the objectives, proposed solution, scope and the current system with problems it face and solution have and limitations. Finally it described the current methods of solving identified problems.

2.1 Existing System

2.1.1 OpenGPS

OpenGPS is a free and available open source project designed specifically to provide enhanced GPS tracking services for a "fleet" of vehicles. While it was designed for the purpose of use with commercial fleet tracking systems, the software is fully compatible with many other GPS tracking systems as well.

OpenGPS can help support the data collection and storage of GPS tracking and telemetry data from mobile devices, and also provides the following set of features:

- Web-based administration
- GPS tracking device integration
- Customizable web page integration
- Customizable tracking service
- Customizable reports

CHAPTER 2

CURRENT TRENDS OF THE GPS TAXI TRACKING APPLICATIONS

2.1 Introduction

Chapter one described the background and the problem in brief with its current trends with limitations and solutions. Chapter two describes the current trends of the GPS taxi tracking applications. During this description, it also investigates the GPS based mobile taxi tracking and fleet management systems which were developed by several software developments companies and individuals. Yellow taxi [7], [8] Online Cabs, [9] My TAXI, Find Taxi [10], EST Call taxi [11], Easy Taxi [12], 99Taxis [13] are some taxi tracking application listed on Google play android market. These tracking systems have many common and uncommon features. However majority of the people are still using old methods to find their taxi needs.

2.2 Existing Systems

2.2.1 Open GTS

OpenGTS[21] is the first available open source project designed specifically to provide web-based GPS tracking services for a "fleet" of vehicles. While it was designed to fill the needs of an entry-level fleet tracking system, it is also very highly configurable and scalable to larger enterprises as well.

OpenGTS not only supports the data collection and storage of GPS Tracking and Telemetry data from remote devices, but also includes the following rich set of features:

- Web-based authentication
- GPS tracking device independent
- Customizable web-page decorations
- Customizable mapping service
- Customizable reports

- Customizable geofenced areas
- Operating system independent
- Easy localization to languages other than English
- OpenGTS is licensed under the Apache Software License

2.2.1.1 Basic GTS Enterprise

Building upon the features and capabilities of this system, Enterprise version has added many other features to their commercial GTS suite to create a customized GPS tracking/telematics solution to fit your specific requirements.

- Additional web-interface support (source code included):
- A "Lite" version of the Event Notification Rules-Engine feature
- GTS DB administration utility (binary) with user interface.
- Configuration Assistant utility
- Additional configured reports:
- Additional documentation.
- Auto startup scripts (on reboot) for Fedora and CentOS.
- Support for accessing the GTS database and querying report, mapping, and database information in XML format.
- Outbound SMS gateway support
- Included device communication server
- Device Communication Server support for
- Event Notification Rules Engine (ENRE)
- Additional Remote Tracking Device Support
- State-line border crossing detection
- Support for trailer drop/hook detection
- Support for load temperature monitoring
- Customized client telematics device integration
- Additional commercial mapping provider and reverse-geocoding support
- Secure Geo-Corridor support
- Support for other custom requirements

2.2.1.2 Limitations of the Open GTS

Since it's a large level application, very complex to use for middle level projects. Also bit difficult to install and upgrade, need high level of technical knowledge. Furthermore this needs high level of hardware resources to serve the application.

2.2.2 Yellow Taxi

Yellowtaxi333.com has a mobile application which supports to find nearest taxies around the user. This one supports on detect taxies and calling them. This application is still on testing and they ask people to support them with user ideas. They said they support on find taxi, fees and indicating user's wishes too. They say that user can input hire summary to the software too. But this software supports only for country or region called Sofia. These details were on their Google play account home page and have not mentioned more details on their design and methodology. And there are no any research details which they done over this.

The application allows you:

1. Find out about your fees
2. Indicate your wishes
3. Assess the taxi driver and car
4. To follow the screen of your phone movements of Yellow Taxi to the desired point.
5. For your convenience, the application only located nearest to address you. You can confirm or change it.
6. If you do not know where you're going, you can easily find them with the tools for selecting the address - text search, selection of personal addresses or search by map.
7. Confirm with a button and the system detects fees around you in seconds.
8. In response to the driver, you get full information about his car, model, rated for quality, including the assessment of real users like you and automatic quality indicators taxis.

9. Travel taxi for you: you see at any time the movement of the car to you, along with information on the distance and time of arrival. In case you receive a notice from the driver for the new arrival time or intermittent demand. Upon discontinuation of the application, simply submit a new request.
10. Taxi waiting for you: the moment of arrival, you will receive a notice in 3 minutes free waiting. If you delay, the driver will wait further, but include apparatus. In the absence of a connection after more than five minutes, the car will leave and this will be reflected in your history.
11. Travel and evaluation: while traveling, you can check the itinerary of your trip if you do not know the area in which you are traveling. Upon completion you have the opportunity to give their personal assessment of the trip.
12. Menu of services: access to a wide range of functions - history, addresses, personal profiles, change language and password, exit, help, general conditions and questions.

2.2.3 Online Cabs - Taxi Sri Lanka

This is an android application which supports in Sri Lanka which allows users to find taxies around them using GPS technology. Ecomlanka.com is the company owned this software and their methodology is to connect taxies with taxi drivers through their mobile phones. That means taxi driver also needs to have the driver version of the application and has to maintain status separately, which will be an additional overhead. Normally a taxi driver doesn't like or wont to tap on his mobile phone to change taxi status on each time he starts or ends the hire. If driver forgot to update his status, users will get wrong details on the available cabs. So the solution should address this issue as well.

Application allows to passengers:

1. See the available taxis in your area as they drive around.
2. Tap the Online Cabs button to catch one.
3. Know when your taxi will arrive and see it approaching on the map.
4. Call your taxi driver directly
5. Rate your driver.

6. It's same as you put your hand to a road taxi. But 100% safety and reasonable rates on taxi fare.

Application allows to Taxi Drivers:

1. Receive alerts when you are on shift and get jobs through Online Cabs.
2. Set your status to AVAILABLE or UNAVAILABLE to control when passengers can see you.
3. Get the passenger destination & pickup before you choose to accept or ignore the job

2.2.4 Find Taxi

Find taxi is Taiwan software production on searching on taxies. This software also supports on looking for taxies and hire them. As their Google play page said, they offer easiest way to hire and inform others on that taxi service. But they don't mention their design or methodology. And this software also supports only on Taiwan.

For Passenger:

1. Collect Taxi Company listing in Taiwan, allow you to call taxi with ease
Allow you to call taxi even in off-peak hours or suburbs, provide more choices during the rush hours.
2. Review feature, to encourage good drivers and filter out unqualified services or taxi companies
3. Online Booking feature, sending request via Internet if there's driver around
4. Filter by service item (Accessibility, Pet, Charter Service, Tourist Guide, Airport, English, Japanese, Female Driver, New Car, Micro Moving, No Smoking)

For Driver:

1. Added new Driver feature, allow driver to login and register, to receive booking requests
2. Set queue-up location, allow you to queue for passenger anywhere



3. Allow you to provide additional information to let the passenger understand your specialty and service
4. Set multiple service item (Accessibility, Pet, Charter Service, Tourist Guide, Airport, English, Japanese, Female Driver, New Car, Micro Moving, No Smoking)

2.2.5 Call Taxi

According to software owners, they have many services which seem similar to my solution. Indicate nearest taxies, inform when taxi arrives, taxi rating facilities include on both solutions. But this software solution also supports on few countries and not supports for Sri Lanka. Because of this also is a Google play application they don't mention their methodology or design in details.

Main features of the application:

1. Locating of the passenger position by GPS.
2. All drivers within the passenger area are able to see their bookings.
3. Notifying passengers with details of the vehicle that is picking them up and fare estimates.
4. Notifying passengers with a signal when a Taxi has arrived.
5. Real-time tracking of progress of a passenger vehicle on a map.
6. Rating drivers by the end of a trip.
7. Bonuses for the first and further trips which can be used as payments for trips.
8. Working referral system which provides additional bonuses for invitations of friends to use the application.

2.2.4 Easy Taxi – Taxi Cab App

Easy taxi is software which provides from a taxi company. Users can ask for taxies and then application automatically detects the user location and updates the company CRM application. Then process again link with an old taxi finding methodology and users cannot find their nearest taxies or call them. With this they have to wait until

Taxi Company sends a one. This taxi company also based on India and only supports on few regions on India.

Features:

1. Fast & Easy: Find your taxi easily without wasting time and money.
2. Safe: You can track and see the driver details and the taxi details beforehand.
3. Free: Application is free to install and will be no hidden costs.

2.3 Feature Summary Comparison

According to the specifications of the above previous applications, there are some equations and differences between current solution and those applications. Most of them are running only for limited areas. When look at the technical specifications, all applications are based on the GPS technology to track the location of the taxi. Some has the driver rating facility which also available in the proposed solution.

Proposed solution includes the hire history details and customer satisfaction information through the ratings system embedded to it. Above applications don't show that kind of useful information fluffiness. And also they don't mention the used programming language or the approach which used to develop these applications.

Furthermore they mentioned the GPS technology but there are no any technological declarations with regards to the use of GPS technology over the application and the methods used to communicate between main application and the taxi module.

After carefully looking in to those application features and configuration notes, we can see some disadvantages and the unaddressed areas like complete history recordings and Web server configurations. And also there's no any API used to communicate between the external modules. Finally the proposed solution will assemble all the useful features to one place which currently spread out on several applications and areas.

2.4 Summary

Chapter two described the current trends of the GPS based taxi tracking systems around the world. It also described the features, advantages and the limitations of those systems. Furthermore it describes the ways that those technologies can used to develop the proposed solution.

CHAPTER 03

TECHNOLOGY ADAPTED AND APPROACH

3.1 Introduction

Chapter three described the main technologies that need to implement the proposed solution. During this chapter we investigate the proposed solution. It is based on several technologies such as GPS, web API, Web server and main android application. Mainly the user application will base on android application which runs on user's mobile phone. Other technologies will support for the application and information passing. Mainly GPS will be used to communicate taxi status and location through GPRS and will update the web server. Then web server will issue those details over API call to the main application. And the information such as user ratings will store in the web server database over the API.

3.2 Communication with GPS Sensor

Taxi meter will be coupled with a GPS sensor which sends taxi meter status and location details over GPRS to the web server. For this we need to input a mobile SIM card to the GPS sensor. This is a one way communication and only the coupled taxi meter sends the signals to the web server.

3.3 Roll of the Web Server

All the application taxi, users and hire details are stored in a web server. This is an Apache/PHP based web application. Taxi GPS sensor and main android application will communicate with this. Taxi registrations and updating will be handling directly from this. Main android application also will communicate with this over an API for taxi details and user ratings.

3.4 How the Web API Connects Clients and the Server

This will be a REST API which supports for data extraction and insertions. Android application will link with this API over API key. When user searching for available taxies, an API call will be fired to the web server and results will be fetched on Google map interface and when user rates the taxi, those ratings data will fire to the web server and taxi ratings will be adjusted.

3.5 Android Application

This is the main application. User can download this application and install on their smart phone. Then basic user details such as name, contact no and location will be saved to the web server database. Then application has an identity on that user, so ratings and hire details also can save based on that. Main features such as fetching available taxies, selecting a taxi, taxi ratings and view history will be displayed on simple and user friendly way. So that all stages of user can access this application without any issue.

3.6 Summary

In this chapter we discussed about the technologies we use to implement the solution. Android, Web Server, API and other main features are described. We will be discussed on Architectural design from next chapter.



CHAPTER 04

ANALYSIS AND DESIGN

4.1 Introduction

Chapter four described the use of technologies for the proposed solution and approach to the new solution implementation. In this chapter we will describe the phases of analysis and design of the solution. There are two main sections. One is the main android application, other one is the web server. For these two main sections, there should be two different designs. System analyzing and design divides to two parts.

1. Application Design
2. Database Design

4.2 Software Requirement Specification

4.2.1 Functional Requirements

Proposed solution should be able to:

1. Find taxies around the users Location.
2. Show selected taxi details on the Map of user's smart phone.
3. Identify the status of the taxies.
4. Display the status of the taxi mentioning whether taxi is available for hires or not.
5. Display the taxies
 - a. Contact details
 - b. Driver name
 - c. Contact numbers.
6. Display the previous ratings belongs to each taxi.
7. Facilitate to feed new ratings on one to five scales.
8. Select a taxi once shown on the map.

9. Tap on the contact number of the taxi details to call.
10. Notify to customer when the taxi reach to his location.
11. Handle admin user account, which controls through a username and password.
12. Allow admin user to log in to the system using his credentials.
13. Allow admin user to change his credentials.
14. Allow admin user to add taxies to the system with.
 - a. Taxi name (Vehicle Number)
 - b. Contact Number
 - c. Driver
15. Allow admin user to add drivers to the system and update driver details when required.
16. Allow admin user to view ad-hoc reports on Hires, Taxies, etc.

4.2.2 Non-Functional Requirements

System should be able to:

1. Access from anywhere through web based protocol like http/https (admin section)
2. View by all users using a common browser with no controls (front end user application).
3. Download easily and install with one or two clicks with no time and no cost.
4. Send requests to server using standard method which use lovers time.
5. Receive and fetch server responses less than 5 seconds.
6. Facilitate at least 5000 taxies and 50000 customer base in the data base.
7. Use free and open source software and flat forms, so that any user can use the application free and fast.

4.3 Application Design

Main application will contain three entities called User, Taxi and the Web server. These three external entities will connect with the application on inputs and outputs. Figure 4.1 shows the main application context diagram.

There you can see three inputs from user to the main application. First one comes when he search taxis around him. Application will automatically get the users location data using his smart mobile phone's GPS sensor. Then it will send those data to the web server through the web API. You can see that data call as an output as 'Location & User data' from the system to the web server. Application will connect with the web server using the GPRS/3G/4G connection of the mobile phone. Then the web server sends those user and the location details to the database and retrieve available taxi details in the given radios. This data set will sends back to the main application as the API response. That will be shown as a input to the main application called 'Available taxi data'. Then the system will fetch this data set on a Google map canvas which user can see and select a high rated taxi. That will be output from the system to the user and it is shown as 'Available taxies'.

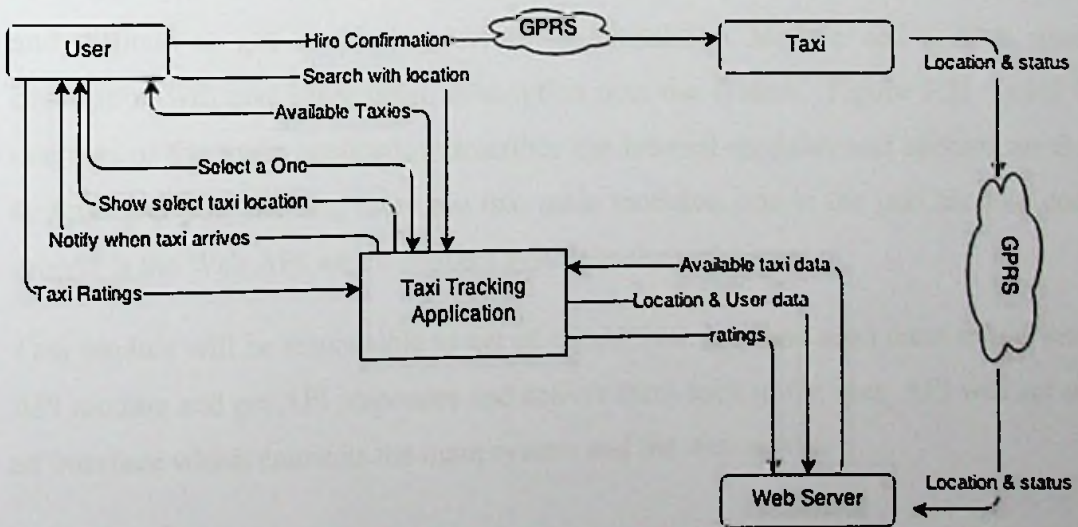


Figure 4.1: Context diagram of the main application

Next input to the system will be the selected taxi data. It's shown as 'Select a one' from user to the system. That means, when user tap over a taxi icon system will sends that taxi details to the web server and retrieve the exact location and rates with contact details of the taxi. If those details are fixing with the user, he will contact the taxi driver over a voice call. This voice call is external to the system and they will discuss the actual availability, conditions, pick up and destinations and will confirm the hire. If they confirm the hire, user can set it as the selected taxi or go for another one. System will show the taxi location and movements when the hire is confirm. This output is shown as 'Show selected taxi location'.

System notifies the user with an alarm after taxi reaching to the pick-up location. This will be done by matching user and taxi GPS data. This is also and output data line, so that is shown as 'notify when taxi arrives'. Then system will be silent for a while, till user finish the hire with selected taxi.

End of the journey, user can rate the taxi and the hire by using the rating system appearing on the screen. These user ratings will be an input to the system and will be sends to the web server. This can be seen as 'User ratings' – input, 'ratings' – output, and will be used to highlight the high rated taxies for future taxi listings.

Above is a high level view of the solution. Only the inputs and outputs can be seen and difficult to get an idea on actual functionalities. Module and section wise description will give more detail description over the system. Figure 5.2: Level 0 diagram of the main application describes the internal modules and section on the system. As you can see, there are two main modules, one is the taxi module and second is the Web API which connect system to the web server.

Taxi module will be responsible to get all inputs from user and send them to the web API module and get API responses and deliver them back to the user. API will act as an interface which connects the main system and the web server.

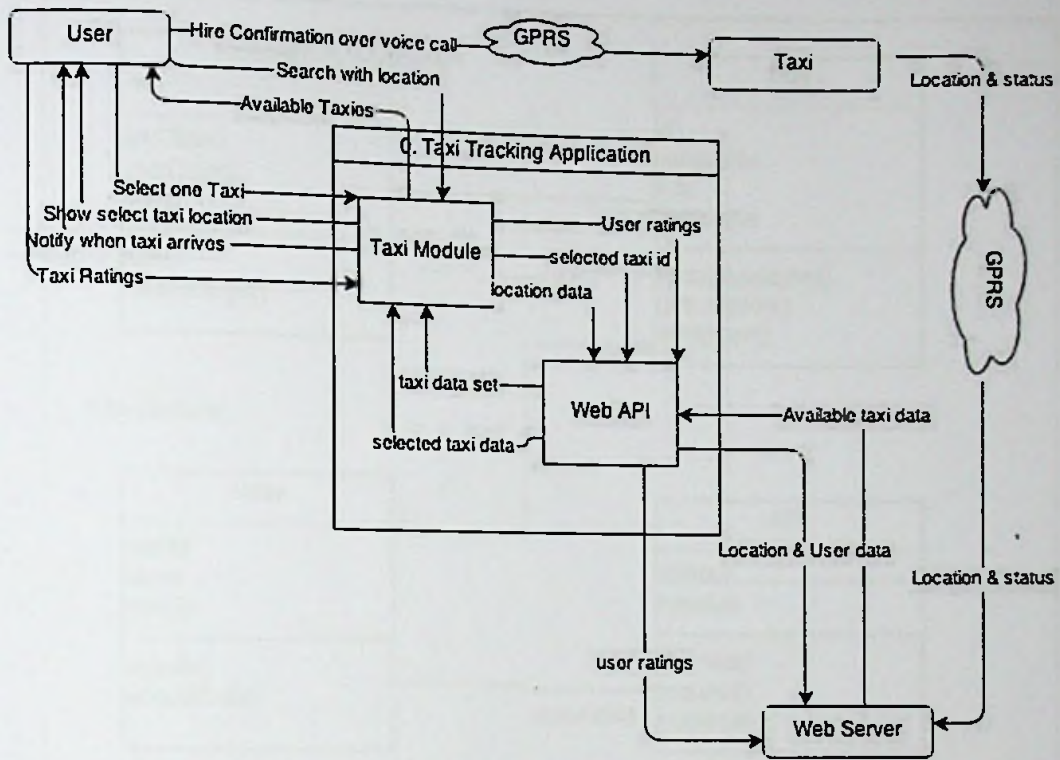


Figure 4.2: Level 0 diagram of the main application

4.4 Design Methodology

The conceptual model is needed to direct or guide the designers to the relevant aspects of the system. The set of procedure provides the designer a systematic and logical set of activities to begin the design task. The evaluation criteria provide an objective measurement of the work done against some established standard or specifications.

4.5 Proposed Architecture

Structure and development plan is based on Figure 4.3: Class Diagram and Figure 4.4 : Use Case Diagram. Currently application has four main classes (Objects). Those are User, Taxi, Ratings and API classes. Here User object will be created for each registered user. When user start the application an object of user class will be generated with all the credentials and will be ready to deal with the application API. All the communications and transaction will be handled by that user object.

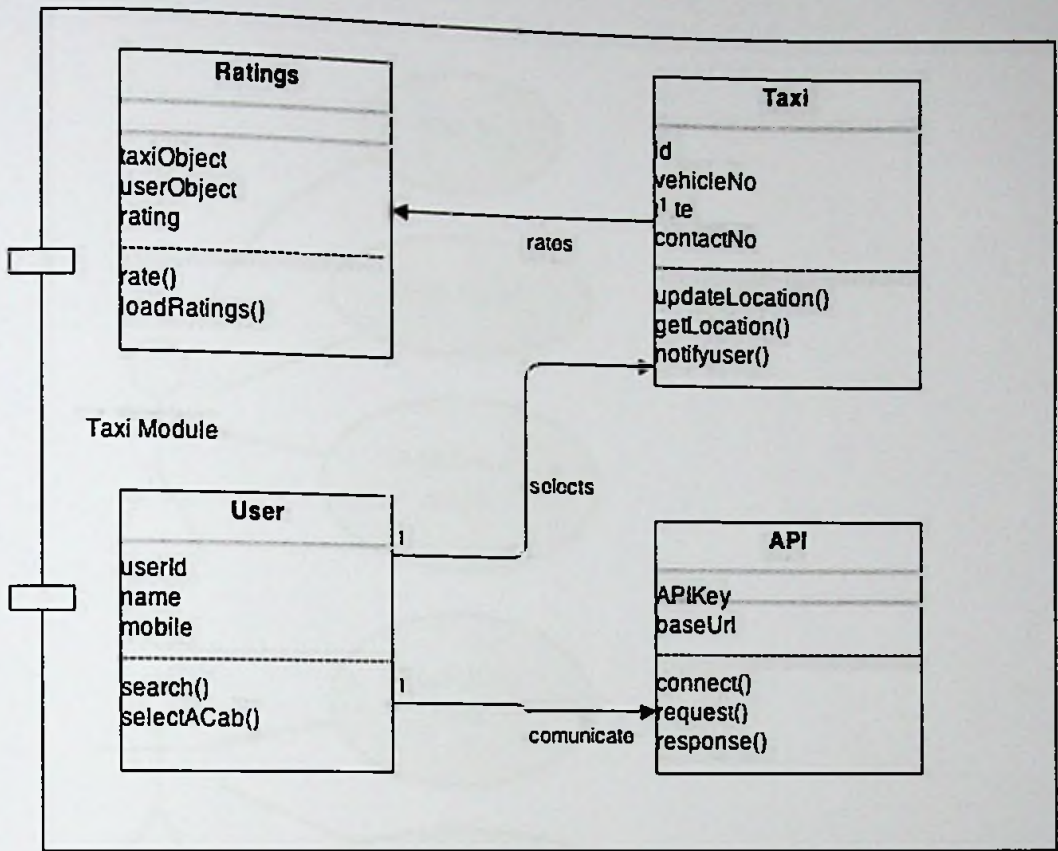


Figure 4.3: Class Diagram

Taxi object will be representing each physical Taxi. When user gets list of available taxis, set of objects will be fetched in to the Google canvas. Thereafter application will be dealing with these taxi objects. When user select a one taxi object, application can directly retrieve details from that taxi object without recalling to the server. Once user confirms the hire selected taxi object will be update with relevant data.

Ratings class will be handing all the user and hire ratings. That means when user loads available taxis, a ratings object also will be coupled in to the taxi object. Same thing will happen, when user rate a given taxi too.

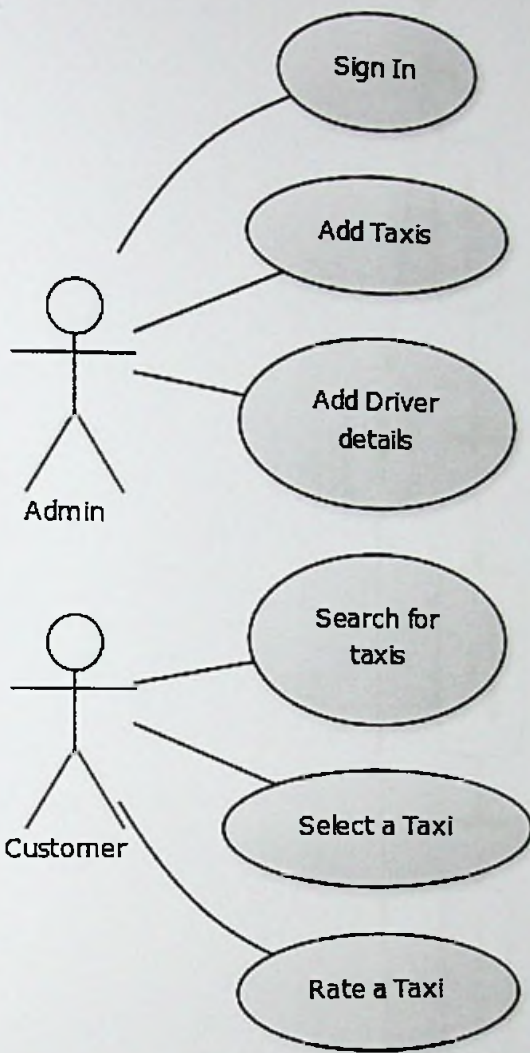


Figure 4.4: Use Case Diagram

The main class will be the API. This is the interface which main application communicates with Web Server. So all the requests and responses will meet the API class object. Creating an object for all requests and response will be a huge memory consuming task. So here I will use the singleton Design Pattern. So one single API object will be handle all the requests and response. There will be more supporting and parent classes to the API class which help to connect through GPRS connection. But here I'm not going to mention those additional classes. Figure 5.5: Activity Diagram shows the flow of selecting and rating a given taxi according to the design.

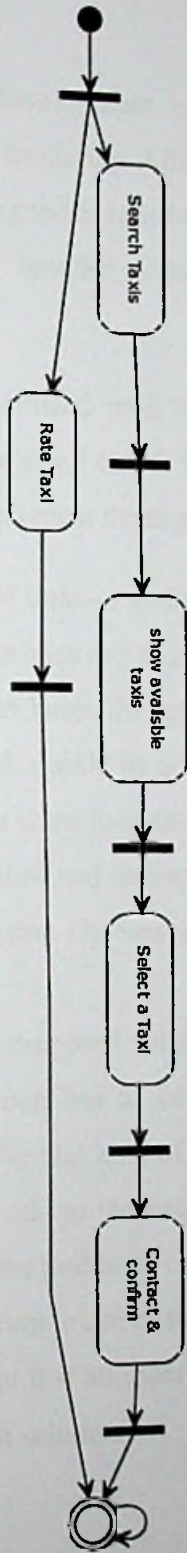


Figure 4.5: Activity Diagram

4.6 Why Agile Scrum

Scrum is the leading agile development methodology, used by Fortune 500 companies around the world. The Scrum Alliance exists to transform the way we tackle complex projects, bringing the Scrum framework and agile principles beyond software development to the broader world of work. It works with below characteristics.

A product owner creates a prioritized wish list called a product backlog. During sprint planning, the team pulls a small chunk from the top of that wish list, a sprint backlog, and decides how to implement those pieces.

The team has a certain amount of time - a sprint which has usually two to four weeks to complete its work, but it meets each day to assess its progress called daily Scrum.

Along the way, the Scrum Master keeps the team focused on its goal.

At the end of the sprint, the work should be potentially shippable: ready to hand to a customer, put on a store shelf, or show to a stakeholder.

The sprint ends with a sprint review and retrospective.

As the next sprint begins, the team chooses another chunk of the product backlog and begins working again.

The plan was to implement the proposed solution with several chunks of time slots since the other works of research has to be performed. When using scrum like methodology, it's easier to handle that kind of time framed schedule. Furthermore it allows adding more features to add to the back log which can implement on future sprints. So the new findings or the problems can easily add to the solution. Although agile has a concept called scrum master, for this solution based on one single developer couldn't use it. Except that all other agile features were highly helpful for the development of the proposed solution.

4.7 Modules of the Solution

There are four modules which drive the total application. Those are:

4.7.1 User Module

All user activities are handles here. Under these user registrations, updates, listings are maintained. There will be one user object throughout the application and singleton design patern will be used as the technology. The main User class object will be used.

4.7.2 Taxi Module

This will handle all taxi related operations. Under this taxi adding, listings and driver details handling will be handled. Most of the times main application and server will use this module to add and update taxi and driver details.

4.7.3 Hire Module

This is the main module which communicate with all other modules. That means all user and taxi modules will link to this for share and save information. When a user select a taxi for a hire this module has to play a major roll to save and handle all the records correctly.

4.7.4 Ratings Module

User can rate a hired taxi after hire and those ratings details will be handled here. During that process adding new ratings and showing the average taxi ratings will be carried out.

4.8 Database Design

In the server side, there will be a database which stores all the taxi, driver, user and ratings data. Figure 5.7: Database design ER (Entity Relation) Diagram will show the structure of the database. As you can see, there are five main entities with relevant attributes which connect each other. Those are Taxi, Driver, User, Hire and

Ratings. When web server receives data through the API, it saves them here and retrieves when required.

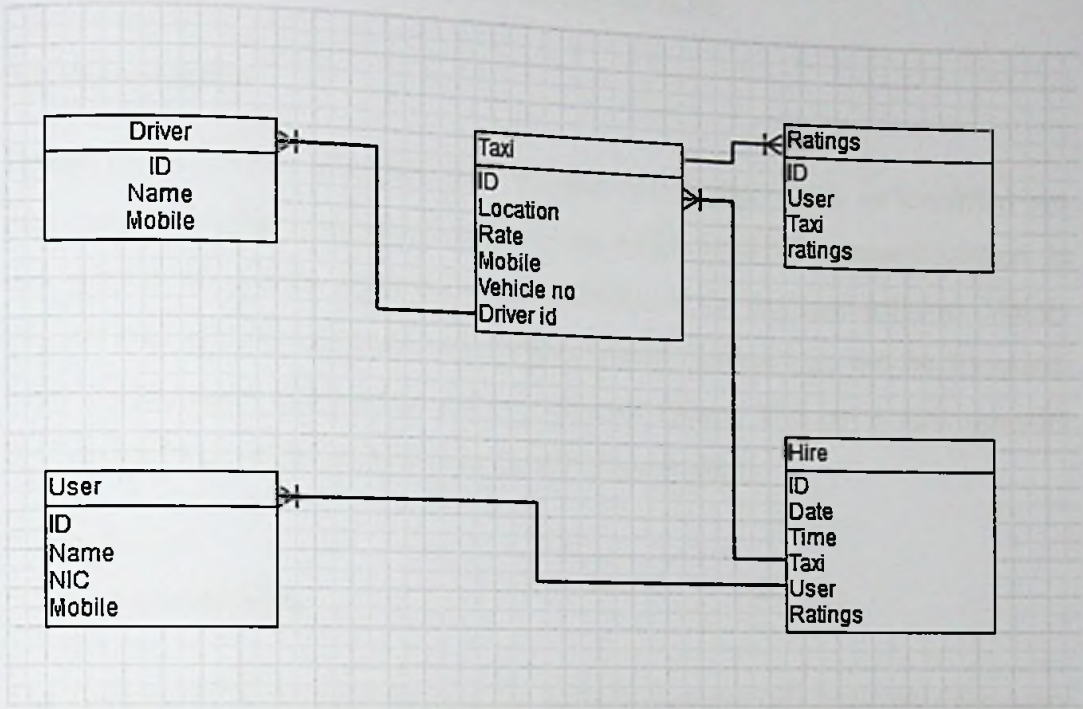


Figure 4.7: Database design ER (Entity Relation) Diagram

All the registered taxi data will contain inside the taxi data table while their hire and ratings will be stored inside Hire and ratings data tables. User table contains all the user related data such as name address location and contact details.

4.9 Summary

Application design and the Database design are discussed on this chapter. There overview of the system considered with high level view. Context Diagram, Level 0 diagrams, Use case diagram and Activity diagrams are used to visualize the design of the proposed solution. Implementation details will be discussed from next chapter.

CHAPTER 05

IMPLEMENTATION

5.1. Introduction

Design of the solution discussed from the previous chapter. There we identified two main sections called Application and database which build the proposed solution. In this chapter we discuss about the implementation of the proposed solution. Mainly the software and the hardware and implementation methodology will be discussed. Mobile application will base on android operation system. Web server develops with PHP and HTML. REST will be used for API developments.

5.2. Software

5.2.1 Android SDK

The Android SDK [1] provides you the API libraries and developer tools necessary to build, test, and debug application for Android. ADT Bundle includes the essential Android SDK components and a version of the Eclipse IDE with built-in ADT (Android Developer Tools) to streamline Android application development.

5.2.2 PHP 5

PHP [14] has gained a following among non-technical web designers who need to add interactive aspects to their sites. Offering a gentle learning curve, PHP is an accessible yet powerful language for creating dynamic web pages. As its popularity has grown, PHP's basic feature set has become increasingly more sophisticated.

5.2.3 MySql 5

MySQL [15] is a freely available open source Relational Database Management System (RDBMS) that uses Structured Query Language (SQL). SQL is the most popular language for adding, accessing and managing content in a database. It is most noted for its quick processing, proven reliability, ease and flexibility of use.

5.2.4 REST API

Representational state transfer (REST) [16] is an abstraction of the architecture of the World Wide Web. More precisely, REST is an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed hypermedia system. REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements.

5.2.5 Meteor JS

Meteor, or MeteorJS is an open-source real-time JavaScript web application framework written on top of Node.js. While production-ready and used by a number of high-profile startups, Meteor allows for very rapid prototyping and produces cross-platform (web, Android, iOS) code. It integrates tightly with MongoDB and uses the Distributed Data Protocol and a publish–subscribe pattern to automatically propagate data changes to clients in real-time without requiring the developer to write any synchronization code. On the client, Meteor depends on jQuery and can be used with any JavaScript UI widget library.

5.2.6 Mongo DB

MongoDB is a cross-platform document-oriented database. Classified as a NoSQL database, MongoDB eschews the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas (MongoDB calls the format BSON), making the integration of data in certain types of applications easier and faster. Released under a combination of the GNU General Public License and the Apache License, MongoDB is free and open-source software.

5.3 Implementation

5.3.1 Android Application

There are several modern ways to develop android applications. Most updated way is to use Meteor [17] like javascript framework to develop the application and then compile it to android OS using kordova [18] like compiling tool. While this implementation I also used this newest methodology. Building mobile applications in Meteor is just like writing a browser-based application. In fact, it's easy to write code in Meteor that's designed to run everywhere: in the browser, on Android, and on iOS; all backed by the same Meteor server code running in the cloud; all JavaScript and HTML5. Then all you have to do is convert this JavaScript library via phonegap [19] to compile for the Android OS.

There it's very easy to configure the main application. All you have to do is bind the Events and Event Listeners to the application.

After the configuration, next step is to develop the templates which needs to show to the user. Meteor has a better way for this. it adds the html tags to the beginning and end of the file, and it automatically includes any relevant resources like JavaScript and CSS files.

Templates are used to create a connection between our interface and our JavaScript code. When we place interface elements inside a template, we're then able to reference those elements using our application logic.

To deal with our templates we now need controller actions which do all handling.

5.3.2 DATA Handling and Synchronization

For the mobile application we use Mongo DB as the Data Handler. Every Meteor project comes with its own database. There's no setup or configuration required. As soon as you create a project, the database is also created, and whenever the local server is running, it's database also start to running. This is not a SQL database.

The core feature of the proposed application is the list of taxis. For the taxis table we use a collection with Mongo DB. So we create a collection for taxis.



```
TaxiList = new Mongo.Collection('taxies');
```

After creating the Taxi collection we can use normal database actions like insert, Select with the collection we created. To insert a Taxi we need to supply a JSON object with taxi data to the collection.

Same way we can retrieve data set from the collection using the primary key of the collection.

Finally these data will synchronize with the server application in real time and updates the main database.

5.3.3 Google Map Integration

Information retrieved from the data store should show up on the user's mobile device. For this purpose I used the popular Google map JavaScript Library. It's easier to configure with Meteor framework. Figure 6.4: Google Map Configuration shows the way how it's done.

5.3.4 Back End Applications

Other than the main application functionality, all back end data handling goes with the back end application. There Main Admin login and access control unit handles the user access to the back end side. So only authorized users can log in to the back end manipulate the data. Creating taxis, setting drivers to them and update them are the main functionalities. Other than these there is a set of reports bind to the Admin section which supports to make future decisions.

5.3.5 Connect Server and Client through the API

All the synchronizations and communications are passing through this API. When a request come to the web server, client application sends and request to the API with relevant information. Then the API send that request to the relevant section of the server application depending on the information on it. Then the Server gathered required information calling several database requests and sends back a response through the same API to the client.

5.3.6 Use of Simulator instead of Actual hardware

For the proposed solution, first I suggested a GPS sensor which can couple to the digital taxi meter. But for the development purpose I created another android application which works same as the GPS sensor. This taxi application can work on several status like waiting for hire and occupied. So this saved my time on evaluation process. This application communicate with the web server as same as the GPS sensor does and sends and receives data. Also it proved me the accuracy supplying the correct details for the test results.

5.4 Summary

Chapter 06 describes the details image of implementation. It also gave some example code segments which helps reader to get an idea of the development methods and technology. During that description it described main client application, Google map Integration, Database synchronization, API and the back end application.

CHAPTER 06

EVALUATION

6.1 Introduction

Chapter six described the software implementation in detail with its modules structure. It also gave a brief description of the code base and the data structure of the application with the way data synchronized between client application and the server through an API. During this chapter we discuss on application evaluation process in detail giving a considerable picture of its evaluation hierarchy.

6.2. Evaluation Hierarchy

Evaluation process describes the level of performances and the accessibility of the used hardware, software and the methodology. Every application build so far is met with some kind of issues and malfunctions. The process of testing and evaluation is the section where those issues or successes are identified and rectified. The way it's conducting should be vary according to the application's structure and the used technology.

6.3. Evaluation of hardware

The main client application based on smart phones on AndroidGingerbread (2.3–2.3.7) and higher versions of OS. The minimum Hardware requirements were

1. 1 GHz Processor
2. 3.5 inches Display
3. 512MB RAM

Any smart phone on android OS with this specifications can successfully. The application is tested on all simulations for newer android versions from mentioned above.

6.4. Proving the accuracy of the simulator

Used simulator also an android application which based on the same mentioned OS version. All communications done between the main server and the client application was tested successfully. As an example, simulator has two buttons to simulate taxi status as Idle and on hire. Once the taxi driver hits on idle button, it sends that message to the server through the API. And also when taxi driver clicks on the 'on-hire' button, that message also communicates through to the server. These two main actions were tested during the evaluation process and all results are recorded successfully.

6.5. Evaluation of the back end software

The back end application is acted as the server application. So all administrative functionalities done on this side of the solution. This application needed most security than other sections of the solution. So this was tested for security, accessibility and the load handling. All the test processes were completed successfully with no issues.

6.6. Evaluation of the Application

This should be the interface where end user interact with the proposed solution. So there should be considerable user friendliness throughout the application. User will consider the load time, user friendliness and the accuracy of the data provided. So the test cases should be structured properly to cover all these sections accordingly. To evaluate client application, I created types of evaluation structure.

1. Loading Time Measurements
2. Accuracy Measurements
3. User Friendliness
4. Device Compatibility

All the test processes were successful for the load handling and the accuracy of the data. The way of conducting the evaluation process for the user friendliness will describe under next section in details.

6.7. Evaluation of the User Friendliness

End user should give his/her feedback for a successful evaluation process. Here also I used 52 end users to use and give their feedbacks. 48 of them used the system which was hosted on a beta version and gave their feedbacks. The questionnaire which was given to them was included 5 multiple choice questions which rated their answers to 1 to 5 scale. For a samples of the issued questionnaire, please look in to the Appendix A. the aggregation of the test results are shown on the Figure 7.1: End User Feedback Results.

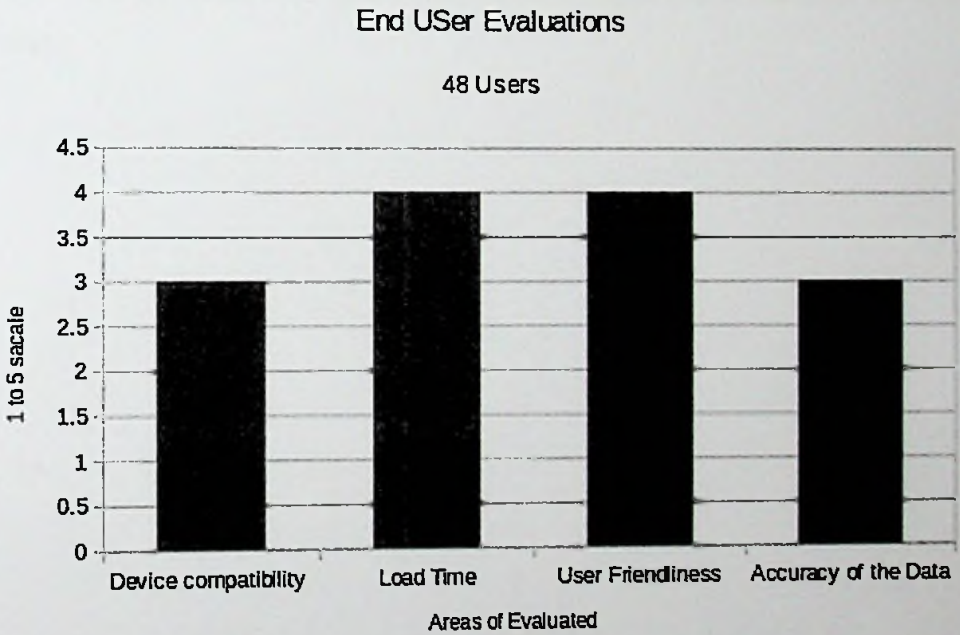


Figure 6.1: End User Feedback Results - Chart

ID	Evaluation Area	Results	%
01	Device Compatibility		70
02	Load Time		85
03	User friendliness		85
04	Data Accuracy		70

Figure 6.1: End User Feedback Results - Table

6.8. Summary

Chapter seven described the evaluation criteria of the proposed solution. It also gave a broader image of the evaluation of all the areas of the application. Finally illustrated the test results for hardware, software applications and received from the end users.

CHAPTER 07

CONCLUSION

7.1. Introduction

Chapter seven described on the evaluation process and its test results. It also describes the areas of tested and results gained from various sources. This chapter concludes the whole proposed solution with the caused problem, and the methodology. Furthermore this chapter will describes the competitiveness of the solution among similar applications and future developments can be made to improve the user experience.

7.2. Competitiveness

Though many applications can be found to find taxies around the passenger in many countries, no such application related to this study has been reported in Sri Lanka. There are many reasons for this such as; (i) difficult to locate (ii) difficult to use and (iii) unfairness. The main purpose of this study is therefore to build an application with more user friendly features such as; (i) select high rated taxi (ii) negotiate directly with the taxi drivers and (iii) see the movement of the selected taxi.

7.3. Future Works

In future, this application can be extended to have more facilities like check hire details using a personalized user accounts, keep default taxis and known taxis for personal use while tracking their routes. Hire rates calculation and more other related functionalities and reports also can be implemented in future versions.

7.4. Conclusion

7.4.1. Problem in Brief

In Sri Lanka, most of the people using private taxies for their day to day travel needs. For this they keep several popular taxi contact numbers in their mobile phone contact data. But when they need a taxi in peak hours or outstations, those remembered taxi companies will be busy or they don't have taxies around that area.

7.4.2. Level of Success of the Proposed Solution

This is a problem which most people face these days. I was able to address the problem by giving an acceptable and user friendly way. This application will give the ability to find taxis near the passenger and make a direct communication channel between taxi driver and passenger. Similarly, this application will have the facility to rate the taxi after the hire and these ratings will save in a web server for other user's reference.

On the other hand taxi drivers will gain more profits, get more hires without even seeing by the passengers. This solution can be enhanced and can be introduced to popular taxi companies.

7.5. Summary

Chapter eight described the conclusion of the proposed solution. It described the problem in brief, success level of the proposed solution and also competitiveness of the solution at market. Finally it described the future works that can be done to improve the proposed solution.

REFERENCE

- [1] TripLog - GPS Mileage Tracker
- [2] Scania Fleet Management
- [3] <http://developer.android.com/about/index.html>
- [4] <http://android.com>
- [5] <http://www.evogps.com>
- [6] <http://www.gpstracking.lk/what-is-sunlanka-gps-tracking/>
- [7] Yellow taxi, www.yellowtaximobile.com
- [8] Online Cabs - Taxi Sri Lanka, <http://www.ecomlanka.com>,
- [9] <https://play.google.com/store/apps/details?id=org.sleepnova.android.taxi>
- [10] Find Taxi, <https://play.google.com/store/apps/details?id=est.taxi.android>
- [11] EST Call taxi, <https://play.google.com/store/apps/details?id=br.com.easytaxi>
- [12] Easy Taxi, <https://play.google.com/store/apps/details?id=br.com.easytaxi>
- [13] 99Taxis, <https://play.google.com/store/apps/details?id=com.taxis99>
- [14] <http://shop.oreilly.com/product/9780596005603.do>
- [15] www.mysql.com
- [16] <http://searchsoa.techtarget.com/definition/REST>
- [17] <https://www.meteor.com/>
- [18] <http://app.kodowa.com/playground>
- [19] <http://phonegap.com/>
- [20] <http://www.wpc.gov.lk/>
- [21] <http://opengts.sourceforge.net/>

APPENDIX A

Traxi Taxi Application Customer Evaluation Form

Your Name :

Province :

City:

When did you recently use a taxi service:

Few days ago Last Week Last Month Last Six Months

After using the Traxi Taxi Application, how do you feel on below areas?

No	Section	Poor	Not Bad	Fair	Better	Excellent
01	Loading Time Measurements	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
02	Accuracy Measurements	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
03	User Friendliness	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
04	Device Compatibility	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>