



Mobile Based Vehicle Service Stations Connecting Application

**A dissertation submitted for the Degree of Master of
Information Technology**

**M.W.M.R. Munasinghe.
University of Colombo School of Computing
2019**



DECLARATION

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: M.W.M.R. Munasinghe

Registration Number: 2016/MIT/045

Index Number: 16550452

Signature

Date:

This is to certify that this thesis is based on the work of Mr. **Mathugama Widanelage Manika Rajitha Munasinghe** under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Prof. N.D. Kodikara

Signature

Date:

ABSTRACT

Sri Lanka's vehicle population rapidly increasing regularly which is around 6.8 million according to Sri Lanka vehicle registration statistics. In order to achieve proper functioning and durability of vehicles which should be maintained and serviced on time. There are number of vehicle service centers around the Sri Lanka to provide the vehicle service facilities such as body washing, oil changing, interior cleaning, waxing etc. However, the main problem or inconvenience is that most of the vehicle service centers are limited to their potential group of customers and they are working as isolated individuals. Although the technologically revolutionary products and services are popular among the Sri Lankan crowd, some of the service centers still do not have their digital presence and they are approaching their customers in conventional way. When considering this from the vehicle owners' or users' side, most of them are busy with their works as well as the smart phones and which related services are much more reliable them to use.

The proposed solution was inspired from the popular mobile based taxi booking application which is providing specific platform to connect with driver and the passenger. This application called "ServoLK" which has two types of mobile applications for both service customer and service center. It provides an opportunity to connect with vehicle users and the individual service centers on a one platform. The application has several features to find the service centers easily such as nearby search and top-rated search as well as review them and find more details about the service centers etc. Service centers also can easily get the service reservations through this application. Therefore, this application introduces specific platform to create their digital identity and fulfills the technological gap between the vehicle user and the service center.

The application supports with Android devices and which mainly uses peer to peer connecting mobile application technology and GPS location services. It was developed by using mobile application development technology called React Native which is robust and highly sophisticated programming language as well as server-side operations were handled by using Google Firebase which is Cloud based NoSQL database. In addition to that number of dependencies were used when compiling the project.

This application was effectively developed and successfully implemented. According to user evaluation feedbacks, "ServoLK" application satisfies the objectives of the project.

ACKNOWLEDGEMENT

I would like to express my deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude I give to University of Colombo School of Computing for offering this master's degree program to individuals like myself whom seeking for knowledge in the IT industry, and also to all the staff members who guide students from the beginning of the program.

Furthermore, I would also like to express my gratitude to Prof. N.D. Kodikara, my supervisor for the invaluable guidance, encouragement and precious support given throughout the journey. More importantly the valuable advice and criticisms drive project more successful.

I also take this opportunity to express a deep sense of gratitude to my closest friends and service centers' owners who supported me to evaluate this project by giving their genuine and valuable feedbacks.

Finally, I express my heartfelt gratitude to my parents, my office colleagues and friend who were with me during the project period by providing advice, guidance and valuable support to make this effort success.

TABLE OF CONTENT

DECLARATION	ii
ABSTRACT	iii
ACKNOWLEDGEMENT	iv
TABLE OF CONTENT	v
LIST OF FIGURES	viii
LIST OF TABLES.....	ix
LIST OF ACRONYMS.....	x
CHAPTER 1. INTRODUCTION	1
1.1 Problem Statement	1
1.2 Motivation.....	1
1.3 Aim and Objectives	1
1.4 Scope of Project.....	2
1.5 Chapter Breakdown.....	3
CHAPTER 2. BACKGROUND	4
2.1 Introduction.....	4
2.2 Proposed Solution.....	4
2.3 Requirements of the Proposed System.....	4
2.3.1 Functional Requirements.....	4
2.3.2 Non-functional Requirements	5
2.4 Reviews of Similar Systems	5
2.4.1 Uber	5
2.4.2 PickMe.....	6
2.5 Implementation Tools and Technologies.....	6
2.5.1 Android Studio.....	6
2.5.2 React Native	6
2.5.3 Google Firebase	7
CHAPTER 3. METHODOLOGY	8

3.1	Introduction.....	8
3.2	Physical and Logical Designs.....	8
3.2.1	Data Design.....	8
3.2.2	Class Diagram.....	9
3.2.3	Use Case Diagram.....	10
3.2.4	Sequence Diagrams.....	11
3.3	Module Structure.....	12
3.4	Important Code Segments.....	12
3.5	User Interface Designs.....	13
3.5.1	Customer App Screens.....	13
3.5.2	Service Station App Screens.....	20
3.6	Tools and Technologies.....	23
3.6.1	Tools.....	23
3.6.2	Technologies.....	23
3.6.3	Deployment Environment.....	23
CHAPTER 4. EVALUATION.....		24
4.1	Introduction.....	24
4.2	Testing Procedure.....	24
4.3	Test Plan and Test Cases.....	24
4.3.1	Register Module.....	25
4.3.2	Login Module.....	25
4.3.3	Search Module.....	26
4.3.4	Reservation Module.....	26
4.4	Lesson Learnt.....	27
4.5	Failures of Achieving the Given Objectives.....	27
CHAPTER 5. CONCLUSION.....		28
5.1	Introduction.....	28
5.2	Problems Encountered.....	28
5.3	Lesson Learnt.....	29
5.4	Future Enhancement.....	29

REFERENCES	30
APPENDIX A – SYSTEM DOCUMENTATION	31
APPENDIX B – DESIGN DOCUMENTATION	33
APPENDIX C – CODE LISTING	41
APPENDIX D – TEST RESULTS	44
GLOSSARY	48
INDEX.....	49

LIST OF FIGURES

Figure 3. 1 : Data Design	8
Figure 3. 2 : Class Diagram.....	9
Figure 3. 3 : Use Case Diagram.....	10
Figure 3. 4 : Sequence Diagram (Login function).....	11
Figure 3. 5 : Sequence Diagram (Reservation function).....	11
Figure 3. 6 : React Native File and Folder Structure.....	12
Figure 3. 7 : Splash Screen (Customer App)	13
Figure 3. 8 : Login Screen (Customer App).....	14
Figure 3. 9 : Registration Screen (Customer App)	15
Figure 3. 10 : Home Screen (Customer App)	16
Figure 3. 11 : Nearby Screen (Customer App)	17
Figure 3. 12 : Detail Screen (Customer App)	18
Figure 3. 13 : Confirmation Screen (Customer App)	19
Figure 3. 14 : Update Profile Screen (Service Station App)	20
Figure 3. 15 : Home Screen (Service Station App)	21
Figure 3. 16 : Reservation Detail Screen (Service Station App)	22
Figure B. 1 : Service Collection.....	36
Figure B. 2 : Rating Sub-Collection	37
Figure B. 3 : Client Collection	37
Figure B. 4 : Reservation Collection	38
Figure B. 5 : Client App Screens	39
Figure B. 6 : Service Station App Screens	40
Figure D. 1 : ServoLK Feedback Form.....	44
Figure D. 2 : Graphical Representation of Feedback (Interface).....	45
Figure D. 3 : Graphical Representation of Feedback (Navigation).....	45
Figure D. 4 : Graphical Representation of Feedback (Functions)	46
Figure D. 5 : Graphical Representation of Feedback (Information).....	46
Figure D. 6 : Graphical Representation of Feedback (Overall).....	47

LIST OF TABLES

Table 4. 1 : Test Case for Register Module	25
Table 4. 2 : Test Case for Login Module	26
Table 4. 3 : Test Case for Search Module	26
Table 4. 4 : Test Case for Reservation Module	27
Table A. 1 : Hardware	31
Table A. 2 : Software.....	31
Table B. 1 : Use Case Description for Login Module	33
Table B. 2 : Use Case Description for Rate Module	33
Table B. 3 : Use Case Description for Reservation Request Module	34
Table B. 4 : Use Case Description for Search Module	34
Table B. 5 : Use Case Description for View History/Records Module	35
Table B. 6 : Use Case Description for Change Status Module	35

LIST OF ACRONYMS

4G – 4th Generation

API – Application Programming Interface

APK – Android Application Package

CPU – Central Processing Unit

EDGE – Enhanced Data Rates for GSM Evolution

GB – Giga Byte

Ghz – Giga Hertz

GPS – Global Positioning System

GPU – Graphics Processing Unit

GSM – Global System for Mobile Communication

HSPA – High Speed Packet Access

iOS – iPhone Operating System

IDE - Integrated Development Environment

JS – JavaScript

Mbps – Mega Bits Per Second

Mhz – Mega Hertz

NoSQL – Not Only Structured Query Language

OS – Operating System

PPI – Pixel Per Inch

RAM – Random Access Memory

SDK – Software Development Kit

URL – Uniform Resource Locator

WiFi – Wireless Fidelity

CHAPTER 1. INTRODUCTION

1.1 Problem Statement

According to the current context in Sri Lanka, vehicle population is very high and it is increasing rapidly. So that vehicle service and maintenance are essential factor in order to make sure proper functioning of them as well as durability. Unfortunately, most of the service stations still follow manual and traditional mechanism to provide their services to the customers. Therefore, customers may have to face lots of inconveniences such as customers may not easily find the service station as they wish, some service station do not exist in the digital platforms therefore which is difficult to connect with them easily and the main problem is there is such a good platform to connect with both individual service stations and individual customers.

Some of the individual service stations they have their own way to address their customer base such as website, Facebook page etc. However, as a whole there is no any specific collaborative and competitive platform to meet with individual services stations and the individual customers who want to get the services from the service stations.

1.2 Motivation

Smart phones are very popular in the modern society. Therefore, people are rapidly engaging with the mobile application-based services because it is very reliable and easy to use without interrupting their day to day activities.

Most of the vehicle service stations which have potential customer base but they do not have much idea about how to increase their customer base and provide better service to their customers by using technology. So that, there is a gap between service stations and using of technology to enhance their business.

Considering all those facts, peer to peer connecting mobile application-based service is much more suitable to fulfill the above-mentioned gaps and provide better service as well.

1.3 Aim and Objectives

The main aim of this project is connecting individual service stations and customers easily through the same mobile application platform which supports to achieve reliable and efficient service to both service stations and customers. The objectives of the project are listed out in follows,

- Customers can be able to find the nearby service stations by using customer's mobile app from the registered service stations.
- Customers can be able to find the top-rated service stations by using customer's mobile app from the registered service stations.
- Customers can be able to give a rating value to service stations.
- Service stations can be able to get the reservation request from the customer's mobile app.

1.4 Scope of Project

This project is carried out as a pilot project which focuses mainly on connecting individual vehicle service stations with the customers easily. There are two main faces in this application as follows.

- **Customer side application** – In customer side application which provides the facility to register customer with adding their necessary information. After completing the registration, application shows that already registered service stations which can be sorted out according to the nearby or top-rated filters. After selecting a particular service station, customer will be able to see the service station profile and reviews and other details. Customer side application includes the rating facility which can be used to give a rate to the particular service station profile and it will be used to calculate average rating to that particular service station.
- **Service Station side application** – In service station side application which provides the facility to register service station with adding their necessary information. After completing the registration, requests of customer side application will be appeared on the screen. Confirmation of the reservation request also can be done through the application and further confirmation can be done by calling the particular customer. After that, the application will be updated with the new information. In addition to that there is an option to show availability or unavailability for customers.

Only the Android versions of the mobile applications will be developed in the initial stage of this project. iOS application and the policy development for the manually validation and evaluation of the users will be focused on the further development stages.

1.5 Chapter Breakdown

Dissertation is the document which contains overall information of the project in chapter wise. This dissertation contains five main chapters followed by References and Appendix. The remaining chapters of the report are briefly described and listed out in below.

Chapter 02: Background

This chapter is written to provide an idea about background of implementing the project. Other than that, chapter provides a review of similar systems, tools and technologies which used in the system implementation.

Chapter 03: Methodology

This chapter provides logical and physical designs of the system. User interface designs and important code snippets using in this project.

Chapter 04: Evaluation

This chapter describes how the system was tested by using various testing methods. It also reported errors and how to overcome those errors and how to modify the system.

Chapter 05: Conclusion

This chapter include the all summarize details of the projects and also include findings and lessons learned during the project with further improvements of the project.

CHAPTER 2. BACKGROUND

2.1 Introduction

This chapter mainly discussed about the proposed system and its features as well as the similar systems which inspired to do this project.

The smart phones are heavily popular among the people since few years ago as well as most of the businesses and revolutionary business models have been invented by using the mobile applications, some of the most popular examples are Uber and PickMe. Their businesses are completely based on the mobile application. The inspiration was come from those applications and this project idea was come out because of some personal experiences as well as some of the similar experiences which faced by other colleagues. The project idea was developed by researching similar projects which is majorly discussed in the section 2.3 (Review of Similar Systems).

2.2 Proposed Solution

The proposed solution is called “ServoLK” which mainly targeted for the mobile platforms. There are two different applications for both customer side and the service station side. Each mobile application has different features and the main feature is both customer side and service station side applications can be interreacted together. It will be helpful to connect individual customers and service stations in a single platform. In case of using technology, these mobile applications use peer to peer connecting and Geo location services.

2.3 Requirements of the proposed system

2.3.1 Functional Requirements

Functional requirements express the required behavior of the system to be built or what the system supposed to do. Below listed are the main functional requirements of the proposed system.

- **Sign in or Sign up to the system** – Both customers and service stations should be able to register with the system and after signing up, they should be able to login with the system.
- **Create and update profile** – Both customer and service station should be able to create and update their profiles.
- **Searching facility** – Customers should be able to search service stations by using nearby filter or top-rated filter.

- **Rating feature** – Customers should be able to rate the service stations by using customer application.
- **Request reservation** – Customers should be able to request a reservation from a particular service station by using the customer application.
- **View reservation history** – Both customers and service stations should be able to see the reservation history and details.
- **Change the status** – Service stations should be able to change their availability or unavailability status by using the service station application.

2.3.2 Non-functional Requirements

Non-functional requirements are the requirements which are not directly affected with the exact basic functions delivered by the system but failing to meet non-functional system requirements may make the whole system unstable. Non-functional requirements are relevant with quality attributes, quality of service requirements and non-behavioral requirements. The non-functional requirements of the system are listed below.

- **User-friendly interfaces** – Both customer side and service station side application should be included simple and user-friendly interfaces.
- **Security features** – Input fields should be validated and database security should be included.

2.4 Reviews of Similar Systems

2.4.1 Uber

URL: <https://www.uber.com/>

Uber Technologies Inc. is a peer-to-peer ridesharing, taxi cab, food delivery, bicycle-sharing, and transportation network company headquartered in San Francisco, California, with operations in 785 metropolitan areas worldwide. Its platforms can be accessed via its websites and mobile apps [2].

Uber has different types of mobile application but mainly it has two mobile applications, one is rider application and other one is customer application. These two types of mobile applications should be needed to connect to provide the services. This application is GPS location base tracking system to identify the rider location and the customer location and all the other calculations and estimations will be handled by the application. User application supports with both Android and iOS platforms [2].

2.4.2 PickMe

URL: <https://pickme.lk/>

PickMe is a Sri Lankan company which introduced the peer to peer connecting taxi service mobile application to the local market. Now PickMe is a vastly growing company in Sri Lanka. It has emerged as the most successful startup within two years and a valuation of Rs. 3 billion and offering 35,000 rides daily [1].

PickMe also follows the same mechanism of Uber with small differentiations. It also has two types of mobile applications which are passenger application and rider application. Both applications should be connected with each other to provide their service. Their service is mainly based on the GPS location tracking service which provides all the necessary data to the application and which generates related other information. The mobile applications are supported with both Android and iOS platforms [1].

2.5 Implementation Tools and Technologies

Mobile computing technologies are changing and enhancing rapidly as well as all the technologies has both negative and positive aspects. So that, it is important to consider all those factors when selecting the tools and technologies. When implementing this project, following technology options are identified as robust and the newest ways of developing mobile applications.

2.5.1 Android Studio

Android studio is the official IDE (Integrated Development Environment) or tool for developing application exclusively for Android platform. It has a strong editor tool for developing creative UI and emulators for different versions to test and simulate sensors without having actual Android devices. It also has a very useful Gradle plugin using which you can create application files (APK) with different configurations. Moreover, it makes exporting and uploading APK on Google Play easy with a single click [7].

2.5.2 React Native

React Native is a JavaScript framework for rendering mobile application in iOS and Android. React Native is a Facebook's JavaScript library for building user interfaces which targets mobile platforms. Now developers can make mobile applications using this JavaScript library which can be shared between platforms that makes it easy to develop in both iOS and Android. Some of the advantages of React Native can be listed as follows [3].

- **Effective for Developers:** While in development cycle when making changes, for web development one can roll out changes by refreshing the browser but in mobile development, it needs to deploy the code-cycle each to implement the changes. It is not necessary to re-compile the entire application to be rebuilt every time it makes changes [3].
- **Cost reduction and Code reuse:** By using React Native, the same code can be used for deployment on iOS as well as on Android. This takes it to huge saving [3].

2.5.3 Google Firebase

Firebase is a platform that makes developing android apps easier. it is owned by Google and is easy to Integrate to the projects which is a back-end service that app can interact with. It has a lot of features such as Real-time database, Firestore, user authentication, powerful security features, file storage and much more. With the Firebase, it does not need to create database schema up front because Firebase is very flexible to make changes to the schema with the progress of the application. As the application evolves over the period of time, it is recommended to build an app with Firebase and change a schema simultaneously based on the requirements. Firebase let the query data from the real-time database which is completely different than traditional SQL queries. Limitation of the Firebase are, it is not supported with the join queries and it supports with the limited number of query types as well as database plan should be upgraded when the project is scaling up [5].

CHAPTER 3. METHODOLOGY

3.1 Introduction

This chapter give an overview of structure of the system which includes logical and physical designs of the system and important business logics using in the system. Under the physical and logical designs, Data design, Class diagram, Use-case diagram, Sequence diagram will be discussed. Some of the important code segments and overview of the project file structure also included here. In addition to that user interfaces designs and tools and technologies that are used in the project will be discussed.

3.2 Physical and Logical Designs

3.2.1 Data Design

Data design section is mainly focused about the designing of the database. This mobile application is based on NoSQL (Not only SQL) database platform which is completely different approach according to the relational database structure. Normally in relational databases which come up with relational data modeling but in here which simply a hierarchical data model as follows [5]. Furthermore, details about the data design are included in Appendix B.

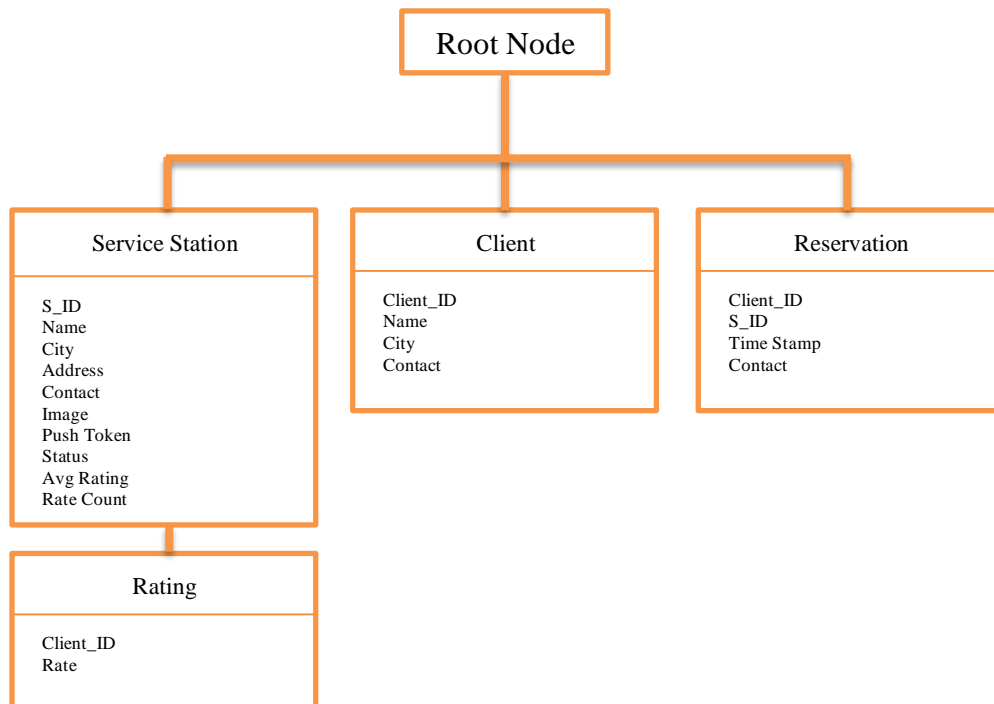


Figure 3. 1 : Data design

3.2.2 Class Diagram

Under the logical designs of the system, class diagram is discussed. There are four main classes which are Client, Service Station and Reservation and Rating. User class is the parent class of both Client and Service Station classes. Reservation class has one-to-many relationship with Client class and Rating class has one-to-many relationship with Service Station class. Figure 3.2 illustrates it in detailed way.

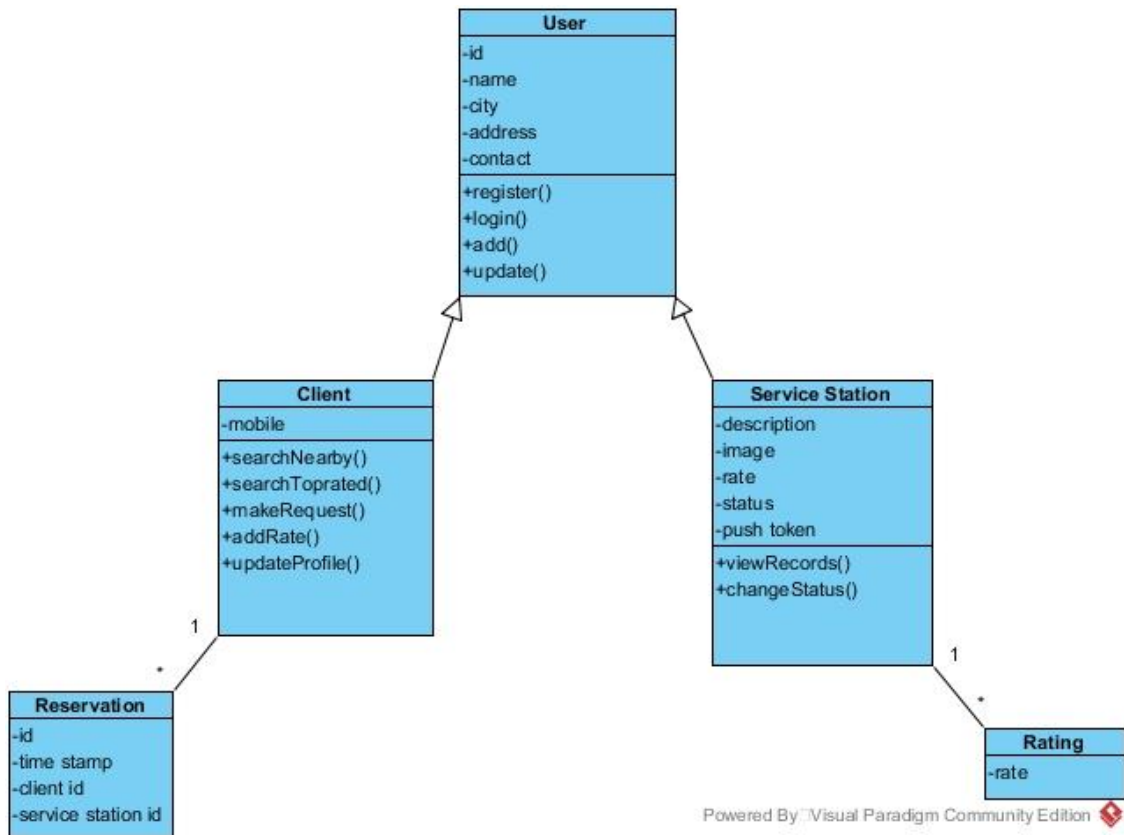


Figure 3. 2 : Class diagram

3.2.3 Use Case Diagram

Use case diagram is also discussed under the logical designs which includes two actors. Those are client and the service station. Figure 3.3 illustrates the corresponding use cases of each actor. Further details about the use cases are included in Appendix B.

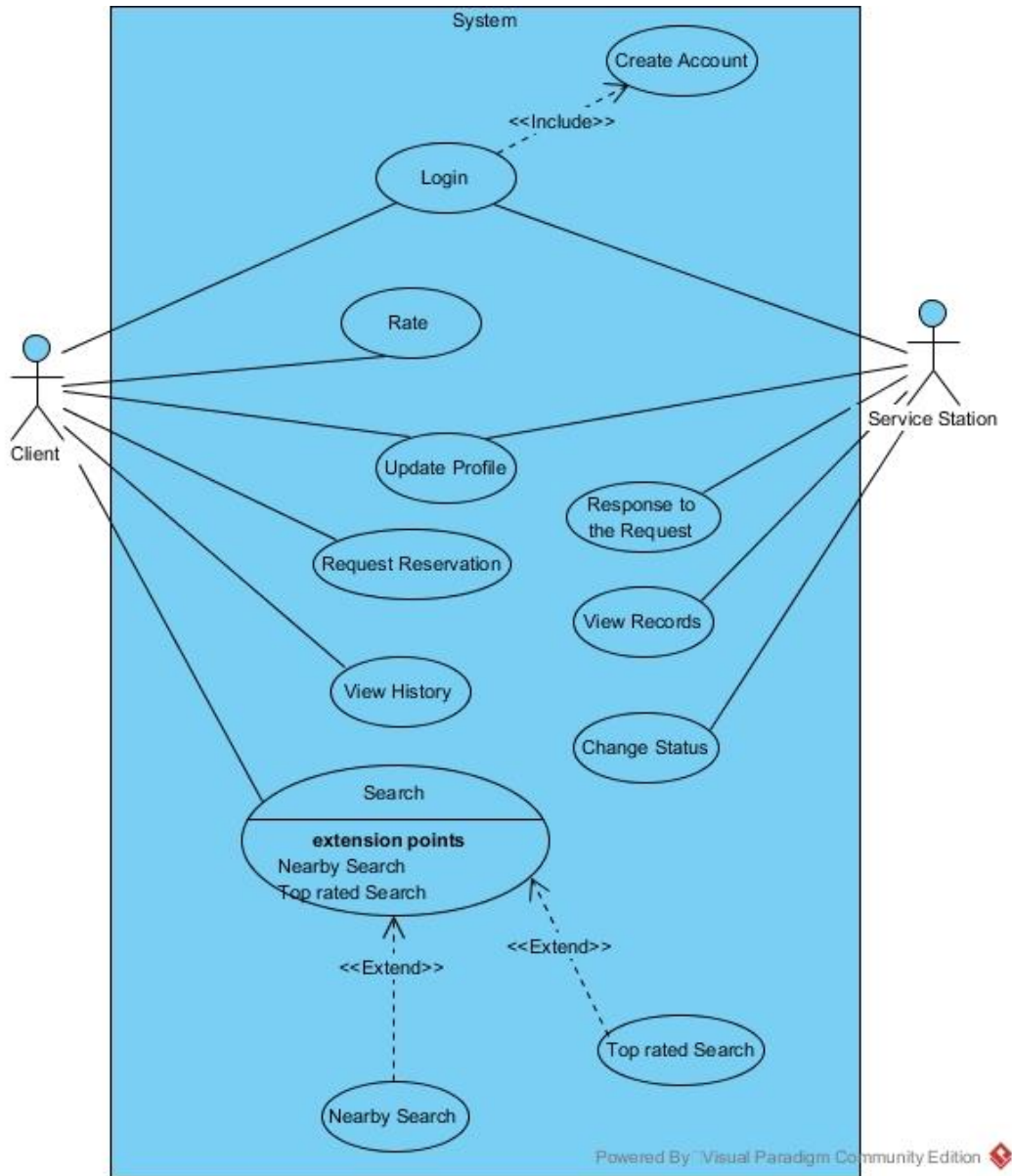


Figure 3. 3 : Use case diagram

3.2.4 Sequence Diagrams

Sequence diagram which indicates the interaction between objects. There are two main sequence diagrams. One is indicated interactions related to the login function and the other one is indicated interactions related to the reservation function. Figure 3.4 and Figure 3.5 illustrates them respectively.

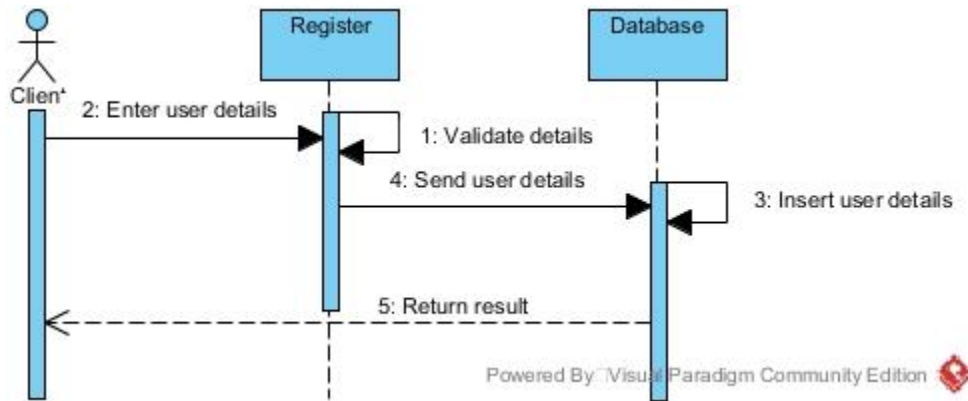


Figure 3. 4 : Sequence diagram (Login function)

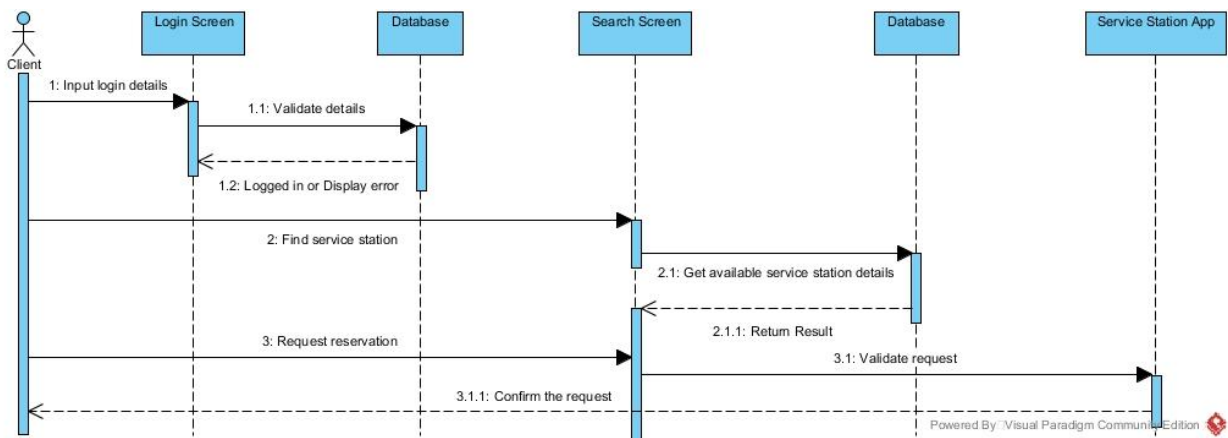


Figure 3. 5 : Sequence diagram (Reservation function)

3.3 Module Structure

React Native is a JavaScript base language. There are some necessary files and folders in the React Native project which is indicated in Figure 3.6. Expo folder consists of all the required expo libraries. Node modules folder consists of all the required Node.js modules. Babelrc file is working as a JavaScript compiler and configurable Transpiler. App.js file which is the main file which renders the code. Package.json file is the configuration file of all the dependencies which use in the project [4].

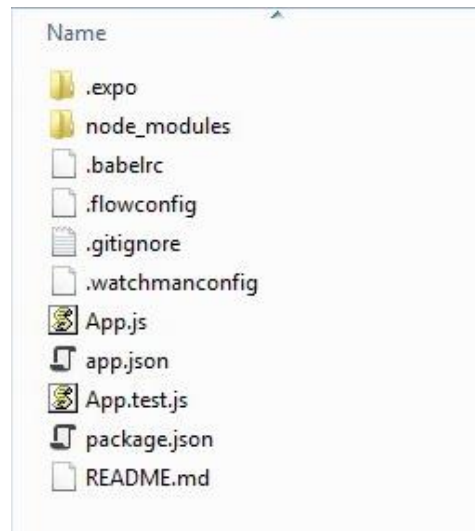


Figure 3. 6 : React Native file and folder structure

3.4 Important Code Segments

This project uses NoSQL cloud database as the server. There are some benefits of using the NoSQL database as well as the drawbacks. One of the major drawback is that there is no way of using join queries in NoSQL databases. In order to join the two collections in NoSQL database, the easiest way of doing is de-normalizing the data. Aggregation function is used in the service station rating function to join with the sub-collection which is explain in the following text base pseudo code [5]. Further details are included in Appendix C.

- 1 → Create a reference for a new rating which should be inside the current service station document.
- 2 → Then, add the new rating value and update the aggregate totals.
- 3 → Compute new number of ratings for the current service station document.
- 4 → Compute new average rating for the current service station document.

- 5 → Finally, commit to Firebase database.

3.5 User Interface Designs

There are two mobile applications in this project. One is called client/customer app and the other one is called service station app. **Customer app** which is designed to the customers who are getting services from the service stations and the **Service station app** is designed to the service stations who are providing services to the customers. Following user interface designs were based to create actual user interfaces. The actual user interfaces of the apps are included in Appendix B.

3.5.1 Customer App Screens

Splash Screen

Splash screen is the welcome screen of customer side application which indicates in Figure 3.7

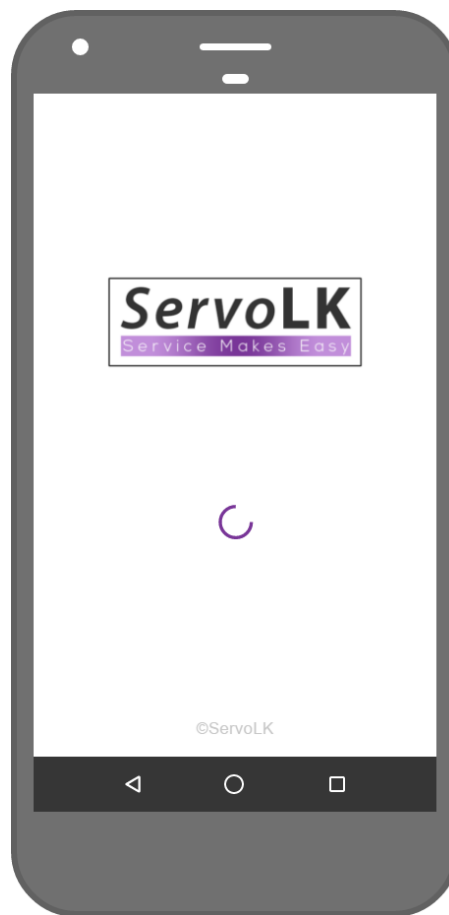


Figure 3. 7 : Splash Screen (Customer App)

Login Screen

Registered users who can login with the system by using the email and password which indicates in Figure 3.8

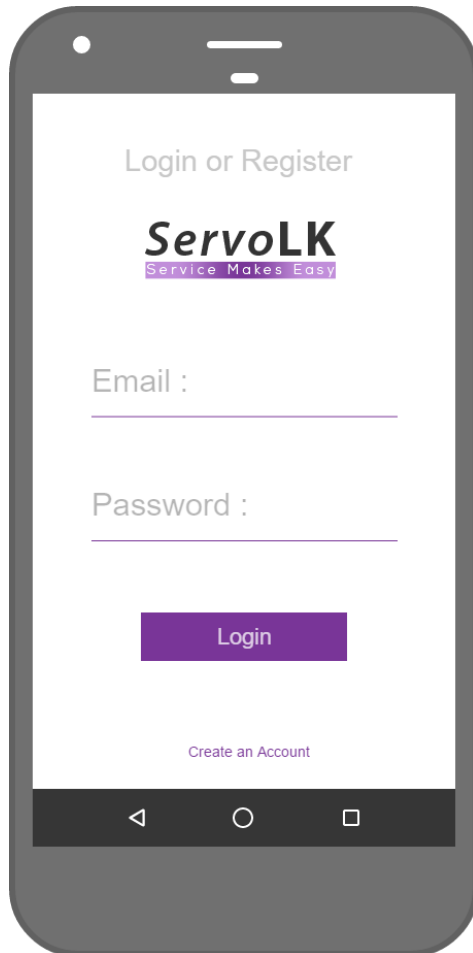
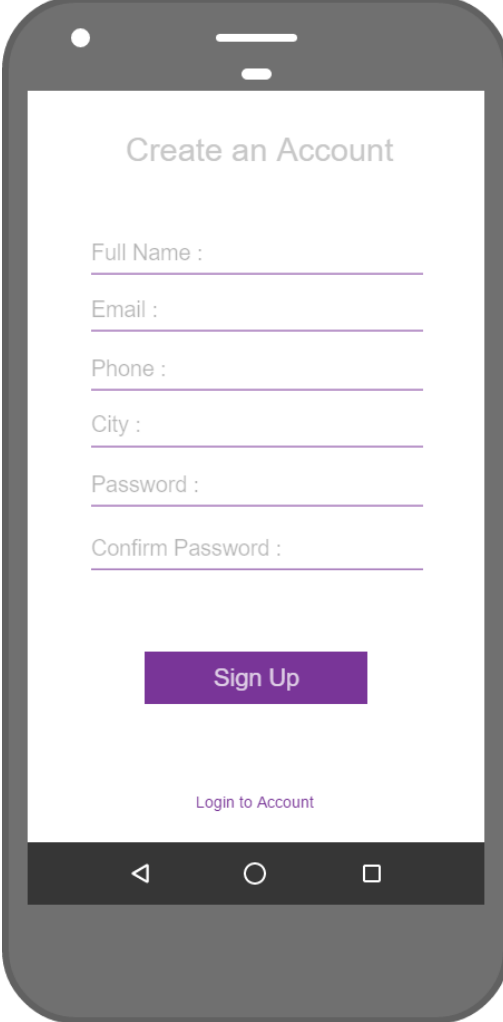


Figure 3. 8 : Login Screen (Customer App)

Register Screen

New users can be able to sign up with the system by completing the registration form which indicates in Figure 3.9



The image shows a mobile application registration screen. At the top, the title "Create an Account" is centered. Below the title, there are six input fields, each with a label and a horizontal line for text entry: "Full Name :", "Email :", "Phone :", "City :", "Password :", and "Confirm Password :". Below these fields is a purple rectangular button with the text "Sign Up" in white. At the bottom of the form area, there is a link that says "Login to Account". The entire form is contained within a grey rounded rectangle representing a smartphone screen, with a black navigation bar at the bottom containing three white icons: a back arrow, a circle, and a square.

Figure 3. 9 : Registration Screen (Customer App)

Home Screen

Home screen consists of all the quick access shortcuts to the app features such as nearby search, top-rated search, profile and settings which indicates in Figure 3.10

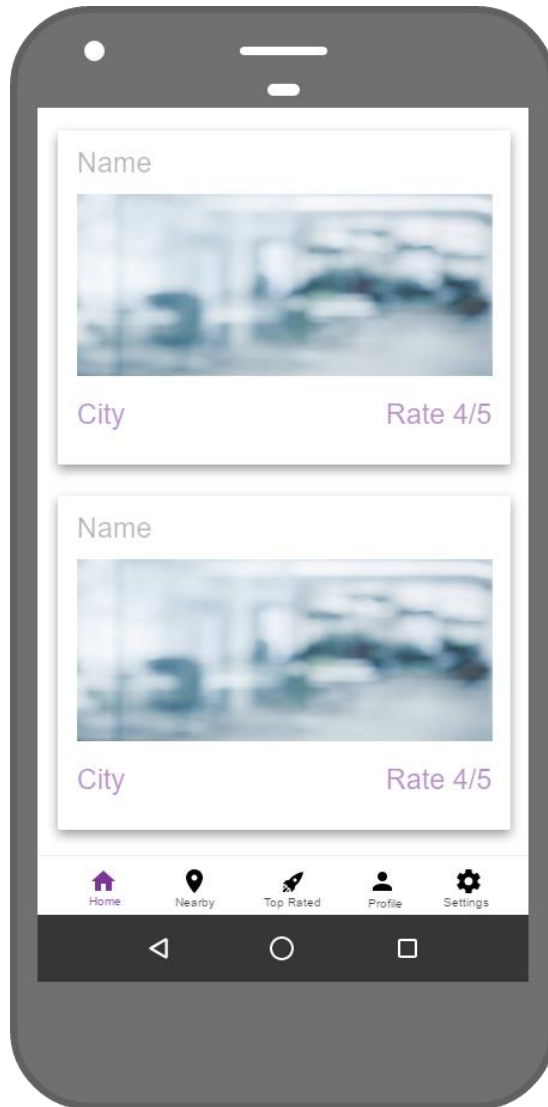


Figure 3. 10 : Home Screen (Customer App)

Nearby Screen

Nearby screen shows the list of nearby service stations which indicates in Figure 3.11

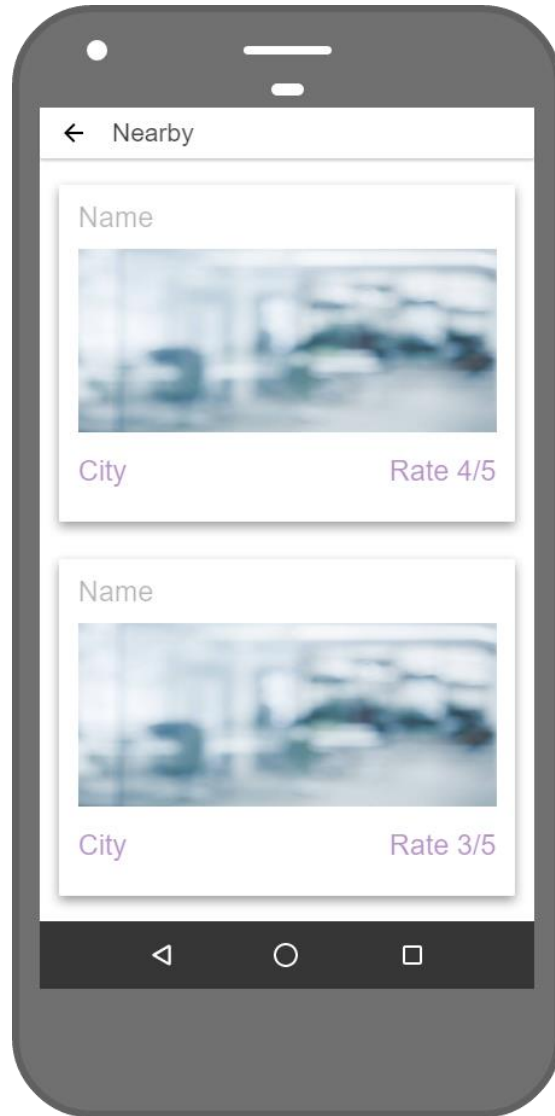


Figure 3. 11 : Nearby Screen (Customer App)

Detail Screen

Detail screen shows the details selected service station such as name, description, map, rating and address. It indicates in Figure 3.12

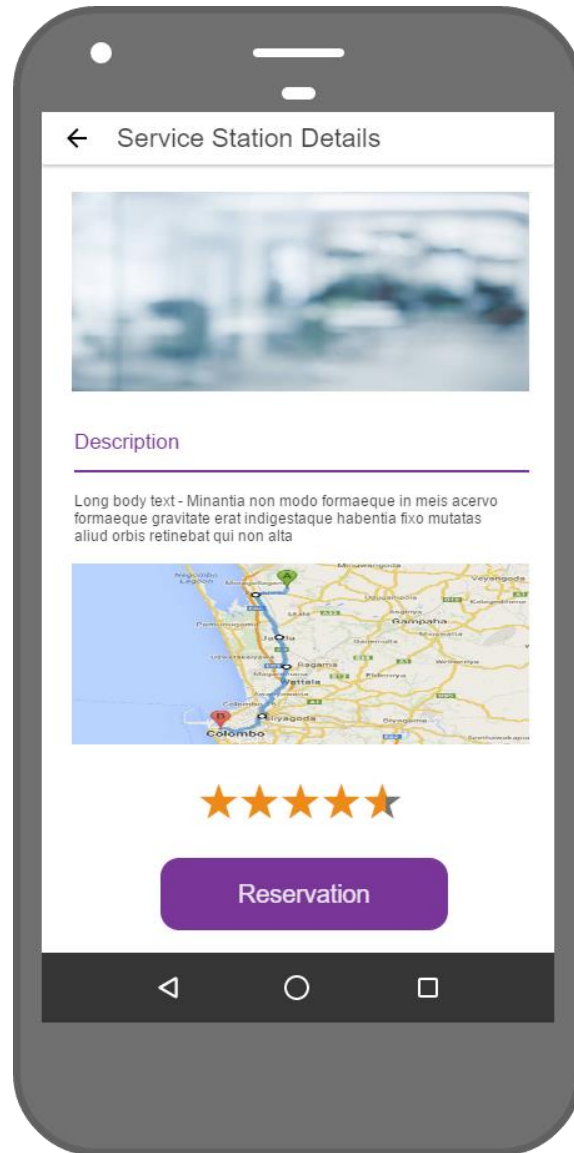


Figure 3. 12 : Detail Screen (Customer App)

Confirmation Screen

After clicking the reservation button which direct to the confirmation screen for service station confirmation which indicates in Figure 3.13

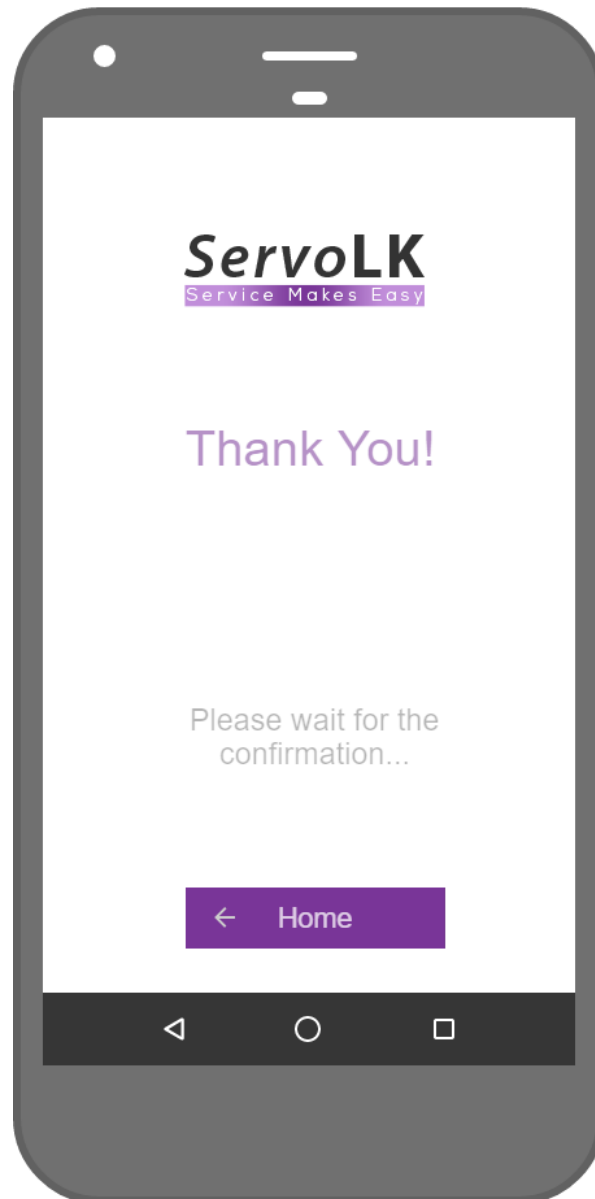


Figure 3. 13 : Confirmation Screen (Customer App)

3.5.2 Service Station App Screens

Service station side application also includes some similar screens to the customer side application but there are some unique screens as well which are listed out in follows.

Update Profile Screen

Service station side application includes the additional screen to update their profile information which indicates in Figure 3.14

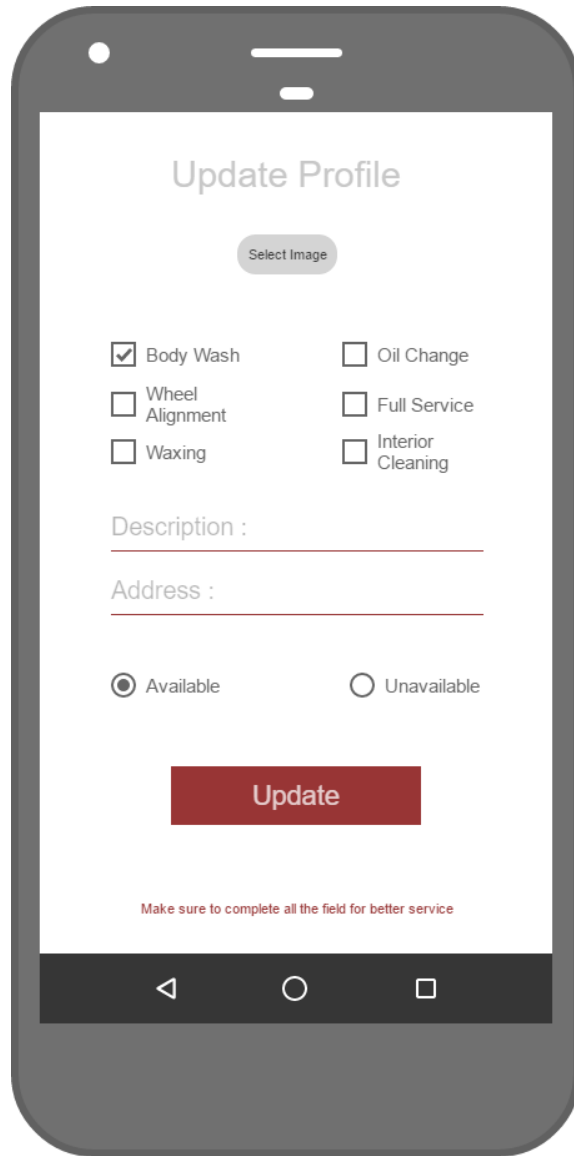


Figure 3. 14 : Update Profile Screen (Service Station App)

Home Screen

Recently updated reservation requests are listed out in the home screen which indicates in Figure 3.15

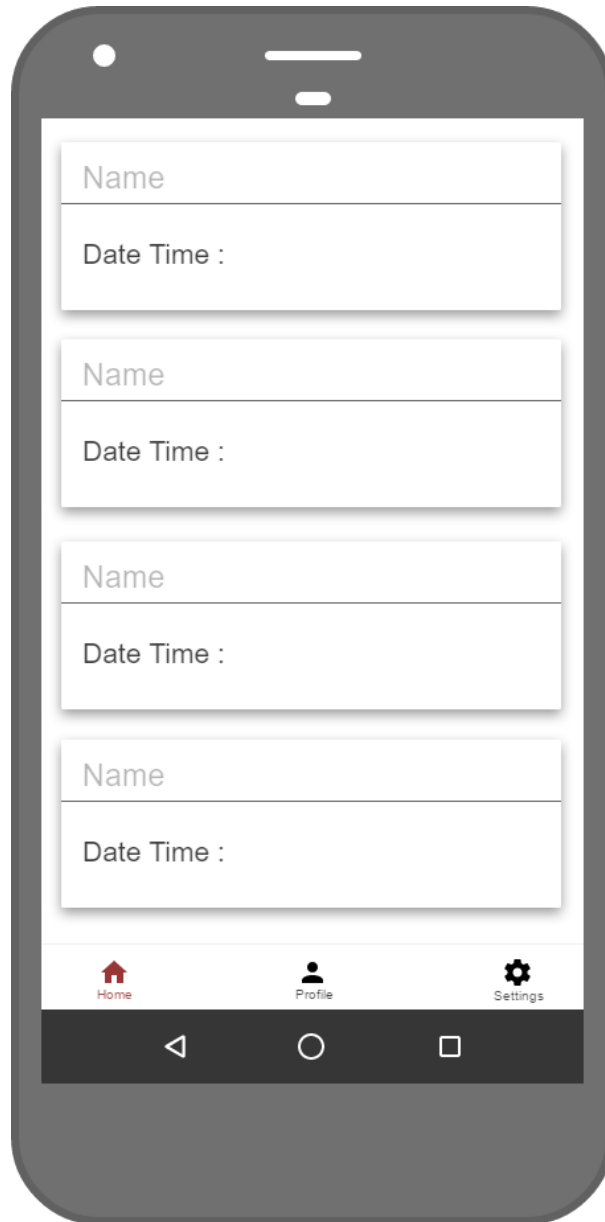


Figure 3. 15 : Home Screen (Service Station App)

Reservation Details Screen

After selecting the particular reservation request which will navigated to the reservation detail screen which shows some additional information about the particular reservation request. It indicates in Figure 3.16

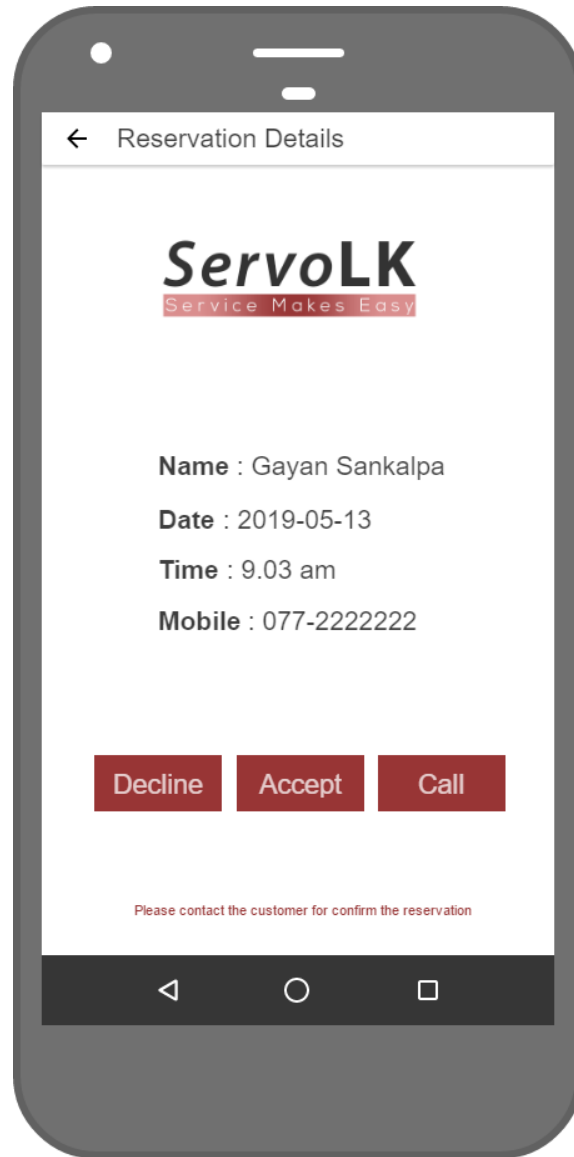


Figure 3. 16 : Reservation Detail Screen (Service Station App)

3.6 Tools and Technologies

3.6.1 Tools

- **Visual Studio Code** – Visual Studio Code (VS Code) was used as the IDE of the project which has so many inbuilt plugins and features for React Native.
- **Android Studio** – Android Studio is used for simulate the project which includes the virtual devices features for run the project virtually [7].
- **Firestore** - Google Firestore cloud data base used as the data base tool which has inbuilt console for manage the database [5].
- **Adobe Photoshop CC** – Adobe Photoshop CC was used as the designing to for app logo and splash screen.
- **Pencil Project** - Pencil Project is an open source GUI prototyping tool which is used to design the app screens.
- **Visual Paradigm 14.1** - Visual Paradigm Community Edition is an UML diagramming tool which is used the design class diagram, use case diagram and sequence diagrams.

3.6.2 Technologies

- **React Native** – React Native was used as the programming language of this project.
- **Expo SDK** – Expo provides the set of tools for React Native.
- **Node.js** – Node.js was used for the package management of the project.
- **Babel** – Babel was used as the JavaScript compiler and configurable transpiler.
- **ECMAScript 6 (ES 6)** - ECMAScript is the latest version of the JavaScript which uses as the base of React Native language.

3.6.3 Deployment Environment

- **Android device** – Recommended requirements for run the application are Android support device with 4.5 inches or higher screen size and Android API level 22 (Lollipop 5) or higher operating system. The device should be enabled with Internet (WiFi/4G) facility and GPS facility to achieve the proper functioning of the mobile application. Further details are included in Appendix A.

CHAPTER 4. EVALUATION

4.1 Introduction

Software testing is an important part of software quality assurance that represents the ultimate analysis of specification, design, and code generation of software product. The testing method is basically combining with Verification and Validation. Validation refers to testing whether the system satisfies the requirements. Verification refers to whether the system implements the specified functions properly. Ultimately it helps to measure whether the project has been satisfied the project objectives or not [6].

4.2 Testing Procedure

Software testing and implementation are repeatable processes as well as software testing is a complimentary process to the implementation. Therefore, both implementation and testing were carried out simultaneously to achieve the project objectives.

The testing procedure of this project is conducted as follows. The basic structural testing was carried out as the “white box” testing which measures “how a program or system does something”. In addition to that “black box” testing techniques were carried out as functional testing which measures the behavioral factors of the particular functions. Mainly, functional testing, unit testing, integration testing and system testing were carried out throughout the project [6].

After completing the implementation process, the system was tested as a whole. Finally, the system was introduced to few people who has some basic knowledge about this process and completed the acceptance test of the project. Test results of the acceptance test and statistical analysis are included in Appendix D.

4.3 Test Plan and Test Cases

Mainly testing is started in the implementation phase. Code is reviewed while development process was carried out. Test plan is used to evaluate the testing process which is used as a guide for overall testing. Test plans were designed in the beginning of the implementation which includes test objectives, schedule, test strategies and especially test cases.

Test cases were designed according to the test plan which contains data, procedure, and expected result and represents which use to system or part of the system run. Each module of the system was tested independently to minimize the complexity of testing process.

Manual testing was used for this project rather than using the automated testing because which is easily manageable. Some of the test cases of basic functions are listed out in below.

4.3.1 Register Module

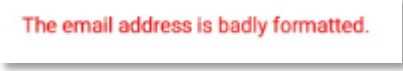

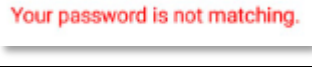

Test Case ID	01		
Tested Component	Register		
Module Name	Register Module		
Test Case	Register Screen		
Expected Output	Appropriate error messages should be displayed for invalid user inputs.		
Test Case Description			
No.	Test Case	Actual Output	Status
01	Incorrect email	Indicates the relevant error message in red color. 	Pass
02	Weak password	Indicates the relevant error message in red color. 	Pass
03	Password mismatching	Indicates the relevant error message in red color. 	Pass
04	When email is already used in the system	Indicates the relevant error message in red color. 	Pass
05	Properly formatted and validated email and password	Validate the email and password and navigate to the home screen.	Pass

Table 4. 1 : Test Case for Register Module

4.3.2 Login Module

Test Case ID	02		
Tested Component	Login		
Module Name	Login Module		
Test Case	Login Screen		
Expected Output	Appropriate error messages should be displayed for invalid user inputs.		
Test Case Description			
No.	Test Case	Actual Output	Status
01	Incorrect email or empty field	Indicates the relevant error message in red color.	Pass


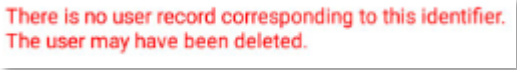
			
02	Invalid user	Indicates the relevant error message in red color. 	Pass
03	Valid email and password.	Validate the email and password and navigate to the home screen.	Pass

Table 4. 2 : Test Case for Login Module

4.3.3 Search Module

Test Case ID	03		
Tested Component	Search		
Module Name	Search Module		
Test Case	Search Screen		
Expected Output	Appropriate error messages should be displayed for invalid user inputs.		
Test Case Description			
No.	Test Case	Actual Output	Status
01	Invalid search	Matching searching item is not found.	Pass
02	Valid search	Validate the searching term and navigate to the search result screen.	Pass

Table 4. 3 : Test Case for Search Module

4.3.4 Reservation Module

Test Case ID	04		
Tested Component	Reservation		
Module Name	Reservation Module		
Test Case	Reservation Screen		
Expected Output	Appropriate app outcomes should be indicated for invalid user behaviors.		
Test Case Description			
No.	Test Case	Actual Output	Status
01	Reservation (Positive)	Reservation request will be sent to the service station app and redirect to the confirmation screen.	Pass

02	Reservation (Negative)	Reservation request will not be sent to the service station app and which is not redirect to the confirmation screen.	Pass
----	---------------------------	---	------

Table 4. 4 : Test Case for Reservation Module

4.4 Lesson Learnt

During the project implementation, many new technologies and dependencies have tried out. React Native is still new to the industry therefore there are so many misconceptions and ideologies behind it. So that, all the possibilities had to be tested out to find out the better way of doing React Native development.

The main misconception is about the React Native is that, pure React Native is much more better than the Expo (React Native Tool) or other third party tools which is correct in some sort of way because pure React Native supports with custom modification but in most of the cases it is much more difficult to configure pure React Native because it has so many dependencies such as React Navigation, Redux etc. Sometimes the project is getting crashed when configuring with other dependencies. Therefore, the lesson learnt is that pure React Native which is good with most of the cases but still it has some bugs and Expo is much more stable version than the pure React Native.

4.5 Failures of Achieving the Given Objectives

Some objectives cannot be achieved because of the technology gaps or difficulties. In section 4.4 (Lesson Learnt) described, React Native is a new technology which needs to configure with some dependencies to work properly. In a mobile app, routing between screens is an essential and main objective. Therefore, React Native which has a dependency called React Navigation which provides all the required features for navigation between screens.

There were some issues which occurred while developing the login function. The routing between screen after the user has been logged in which was not happened the expected and programmed way. The reason of this issue is that the dependency which used to navigate between screen. After complaining and finding out some alternative solutions, finally sorted out that they are having a bug with that particular node module. Therefore, it was a difficulty to achieving the project objectives and alternative solutions had to be found out to fix that.

CHAPTER 5. CONCLUSION

5.1 Introduction

This last chapter discuss about the following topics which are Problem encountered, Lesson learnt and Future enhancement. “ServoLK” is a mobile application which provides the facility to easily find vehicle service stations to the customers. Basically, this app includes two versions for both customer and service station.

The app has included nearby search and top-rated search features which are based on GPS location service and peer to peer connecting mobile technology. Mainly the project was implemented by using the React Native, Google Firebase.

5.2 Problems Encountered

This project idea was come across with some real experience which was faced during the vehicle service experience. Some of the peer to peer connecting services already exist in the market which was inspired to do this project as well. So, in the beginning the project was planned to do with selecting an only one client but after that it was changed to develop as a platform which can be used to connect with both service stations and service customers.

Planning the project and make it as a real application was the biggest challenge because there were not much enough experiences with the mobile technologies and related technologies. It was planned to do by using Android Studio and Java language but when finding out some new technologies, there were some new straight forward and cross platform technologies.

React Native is one of the technology which can be used to develop mobile applications. Although the React Native technology is straight forward, there were so many difficulties occurred while the development. Most common problem of this technology is that there are so many dependencies to configure with React Native and sometimes which led for project crashes. Then the project should be started from the beginning.

The beneficial part of the project and the technology is that React Native is based on JavaScript language. So that, only one code base should be maintained and which is convert to the native Android or iOS modules. In addition to that all the server-side operations were handled by using Firebase which is a cloud-based NoSQL database.

In the beginning there were some confusions of NoSQL database structure and its operations because NoSQL databases violate the relational database concepts, when it is come from relational database background, there were so many new things to learn and some of the relational database concept should be purposely ignored or violated but the advantage part of using the NoSQL cloud based database is that all the server side maintenance will be handled by Firebase itself and pricing is also reasonable than maintaining the own server.

5.3 Lesson Learnt

The significance of the project can be described in three different aspects. Those can be listed out as follows, mainly this project was developed in order to achieve the current issue that having with vehicle users and service stations. The proposed solution is directly addressing the problem and it gives some flexibility to both service customers and service stations which is the main objective of this project.

Another significance of the project is that the technologies which are used to developed this project was added extra challenge because all the technologies had to be learned from the sketch and so many problems were occurred during the development of the project which enhanced the knowledge and experiences about the technologies as well as the time factor made it more challenging.

Finally, after all the significance mentioned above, completing this project is very important to accomplish the high marks for the Master of Information Technology and it is added five credits to the final project. Thus, the project is helpful to enhance the knowledge as well as the academic qualifications.

5.4 Future Enhancement

During the given time period the scope of the project was limited to achieve the main objective which is mentioned in above section. In the initial phase only the two Android apps were developed with the main functions that should be important to run the service. Following developments are listed as further development because of the time limitation.

- iOS apps will be developed as the further development.
- Reservation function will be improved with some addition features.
- Apps will be customized with different devices.

REFERENCES

- [1] PickMe.lk, "Pickme | Cab and Taxi Booking | Fastest, Safest and Smartest - PickMe.lk", 2019. [Online]. Available: <https://pickme.lk/>. [Accessed: 17- Apr- 2019].
- [2] Uber.com, "Uber - Earn Money by Driving or Get a Ride Now", 2019. [Online]. Available: <https://www.uber.com/>. [Accessed: 17- Apr- 2019].
- [3] Facebook.github.io, "Getting Started • React Native", 2019. [Online]. Available: <https://facebook.github.io/react-native/docs/getting-started>. [Accessed: 17- Apr- 2019].
- [4] Docs.expo.io, "Getting to know Expo - Expo Documentation", 2019. [Online]. Available: <https://docs.expo.io/versions/latest/>. [Accessed: 17- Apr- 2019].
- [5] Firebase, "Documentation | Firebase", 2019. [Online]. Available: <https://firebase.google.com/docs/>. [Accessed: 17- Apr- 2019].
- [6] C. Sandler, T. Badgett, G. Myers, The art of software testing. Hoboken, N.J. Wiley, 2013.
- [7] Android Developers, "Documentation", 2019. [Online]. Available: <https://developer.android.com/docs>. [Accessed: 20- Apr- 2019].

APPENDIX A – SYSTEM DOCUMENTATION

This documentation provides the basic guide to setup this application in customer side and the service station side. The basic hardware and software requirements are described below.

Hardware

User should be needed an Android OS running device with following hardware specifications.

Hardware	Recommended Minimum Requirements
CPU	Quad-core 1.4 GHz Cortex-A53
GPU	Adreno 306
Ram	2 GB
Storage	8 GB
Display Resolution	720 x 1280 pixels, 16:9 ratio (~294 ppi density)
Display Size	5.0 inches, 68.9 cm ² (~67.0% screen-to-body ratio)
Connectivity Technologies	GSM, (850/900/1800 MHz), HSPA+, 4G, (850/900/1900/2100 MHz), EDGE, WiFi
GPS	GPS Enabled

Table A. 1 : Hardware

Software

This application currently supports only for the Android devices. Following software specifications should be included to install this application.

Software	Recommended Minimum Requirements
OS	Android 5.1.1 (Lollipop)
API Level	22
SDK Version	22

Table A. 2 : Software

Installation Guideline

First of all, above mentioned minimum hardware and software requirements should available in your device to run this application properly as well as 3Mbps or higher internet connection should be needed to run this application.

- Find the “ServoLk.apk” file from the given source.
- Open the Apk file and install it to your device.
- Turn on your internet connect in the phone and open the installed application.
- First you need to create an account get the service of this application.
- After filling required information, you will be able to create your account.
- Finally, you can log into the application by using registered email and password and get the service from “ServoLK” application.

APPENDIX B – DESIGN DOCUMENTATION

Use Case Diagrams and Descriptions

Use case description for login module

Use Case	Login
Actor	Client (Customer), Service Station
Overview	
	User should be able to login with the system.
Pre-Condition	
	User must be registered with the system.
Flow of Event	
	<ul style="list-style-type: none">• Validate and verify the email and password with database.• Indicate relevant error messages for invalid login.• Redirect to the home screen for valid login.
Post Condition	
	Service station profile information should be updated after the login.

Table B. 1 : Use Case Description for Login Module

Use case description for rate module

Use Case	Rate
Actor	Client (Customer)
Overview	
	Rating values should be added and update the average rating value in the database.
Pre-Condition	
	User must be registered with the system.
Flow of Event	
	<ul style="list-style-type: none">• Validate the user login• Update the rating value with corresponding service station.• Update the average rating value.
Post Condition	

Table B. 2 : Use Case Description for Rate Module

Use case description for request reservation module

Use Case	Request Reservation
Actor	Client (Customer)
Overview	
	Reservation request should be sent to the particular service station.
Pre-Condition	
	User must be registered with the system.
Flow of Event	
	<ul style="list-style-type: none">• Validate the user login• Check the availability of service station.• Send the reservation request to the particular service station.
Post Condition	
	Reservation should be confirmed from the service station side.

Table B. 3 : Use Case Description for Reservation Request Module

Use case description for search module

Use Case	Search
Actor	Client (Customer)
Overview	
	User should be able search service stations by using nearby or top-rated filters.
Pre-Condition	
	<ul style="list-style-type: none">• User must be registered with the system.• GPS should be enabled.
Flow of Event	
	<ul style="list-style-type: none">• Validate the user login• Check the availability of service station.• Display the search result according to the search filter.
Post Condition	

Table B. 4 : Use Case Description for Search Module

Use case description for view history/records module

Use Case	View History/Records
Actor	Client (Customer), Service Station
Overview	
	User should be able see the previous reservation records.
Pre-Condition	
	<ul style="list-style-type: none">• User must be registered with the system.• User should have previous records in the database.
Flow of Event	
	<ul style="list-style-type: none">• Validate the user login.• If the user has previous records, it will be listed out.
Post Condition	

Table B. 5 : Use Case Description for View History/Records Module

Use case description for change status module

Use Case	Change Status
Actor	Service Station
Overview	
	Service station can be able to change availability or unavailability.
Pre-Condition	
	User must be registered with the system.
Flow of Event	
	<ul style="list-style-type: none">• Validate the user login.• Get the current status of service station and change it.
Post Condition	

Table B. 6 : Use Case Description for Change Status Module

Database Design

The database is designed by using Google Firebase console which has inbuilt authentication and storage facility as well. The database has three main collections and sub-collection which are indicated in follows.

Service Collection

The “service” collection has its fields and a sub-collection called “rating” as follows.

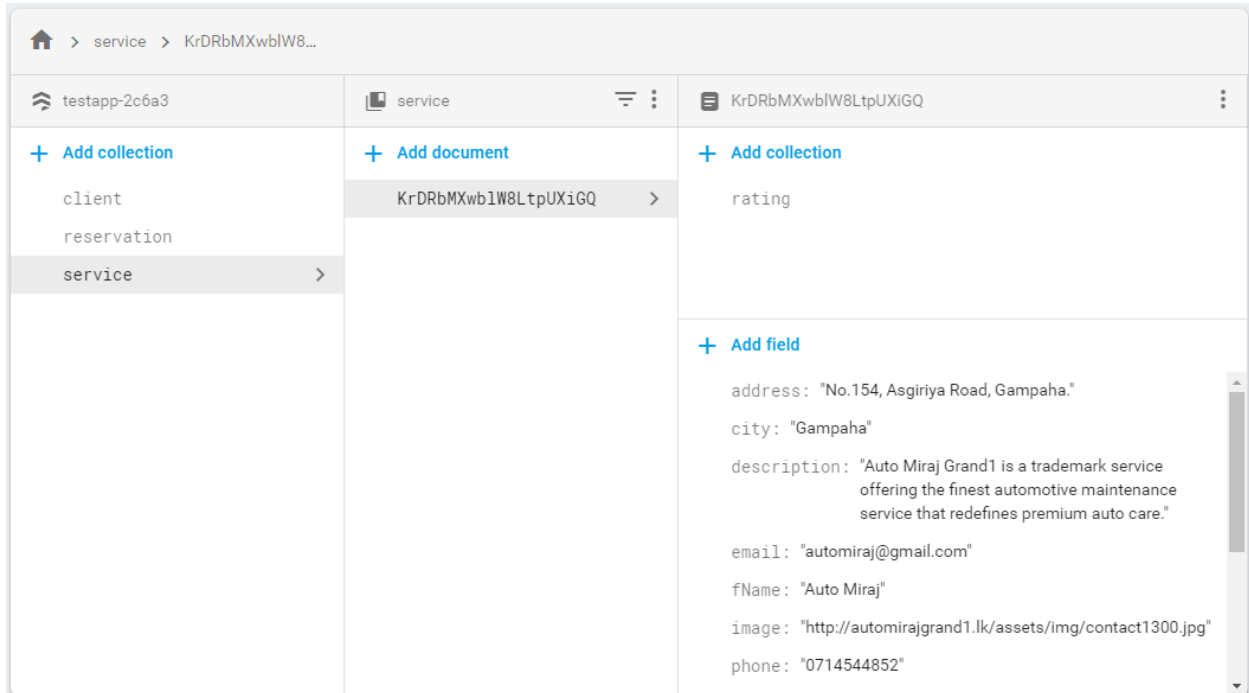


Figure B. 1 : Service Collection

Rating Sub-Collection

Home > service > KrDRbMXwblW8... > rating > DBO8eMk9ipjm...

KrDRbMXwblW8LtpUXIGQ	rating	DBO8eMk9ipjmRjN96cLZ
+ Add collection	+ Add document	+ Add collection
rating >	DBO8eMk9ipjmRjN96cLZ >	+ Add field
+ Add field		rateValue: 3.5
address: "No.154, Asgiriya Road, Gampaha."		
city: "Gampaha"		
description: "Auto Miraj Grand1 is a trademark service offering the finest automotive maintenance"		

Figure B. 2 : Rating Sub-Collection

Client Collection

Home > client > AR0dCZ8H1q0d...

testapp-2c6a3	client	AR0dCZ8H1q0ds1cexZU7
+ Add collection	+ Add document	+ Add collection
client >	AR0dCZ8H1q0ds1cexZU7 >	+ Add field
reservation service		city: "Gampaha"
		email: "gayanrandika@gmail.com"
		fName: "Gayan Randika"
		phone: "0716652125"

Figure B. 3 : Client Collection

Reservation Collection

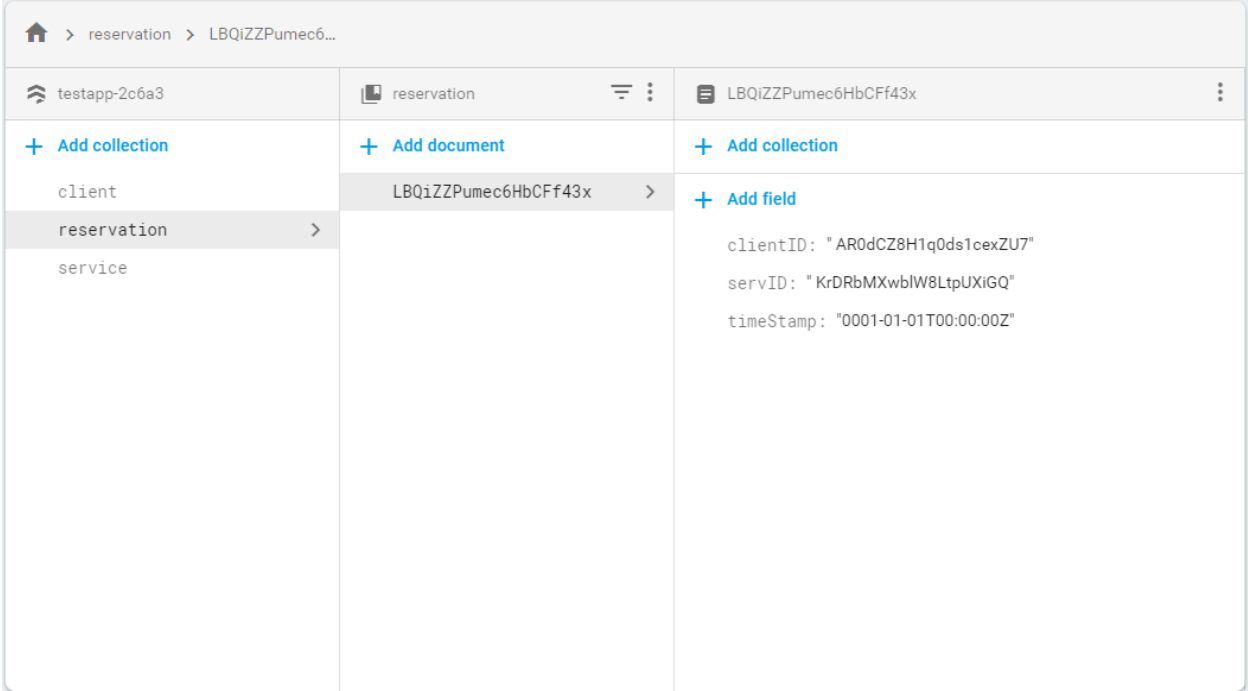
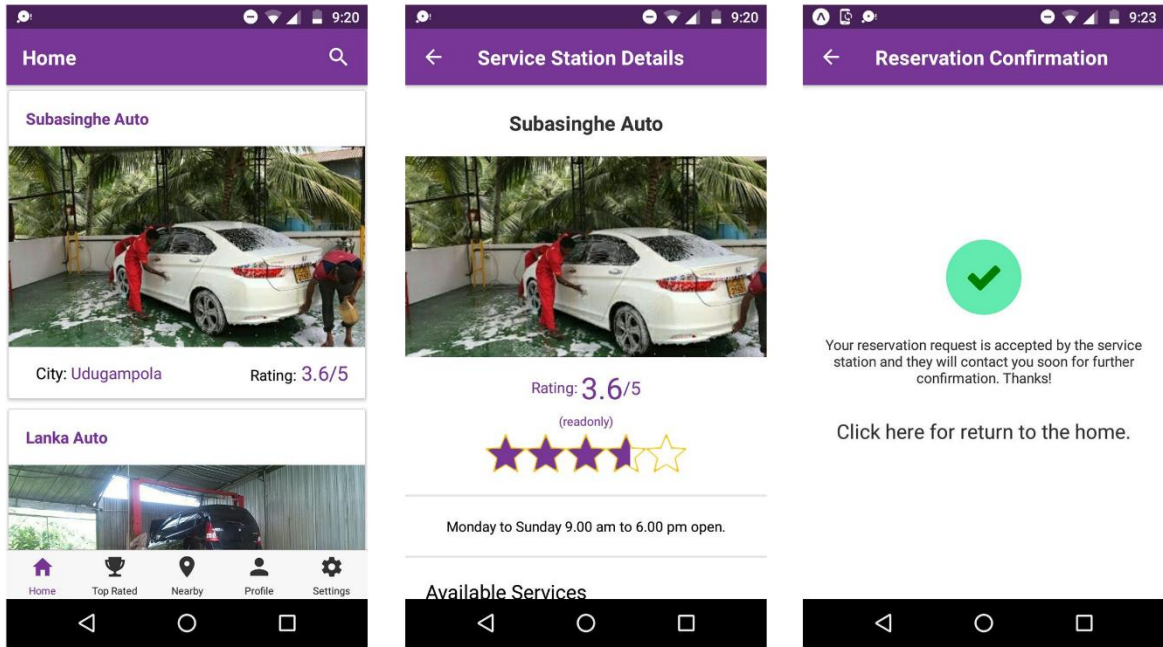


Figure B. 4 : Reservation Collection

“ ServoLK ” Actual User Interfaces

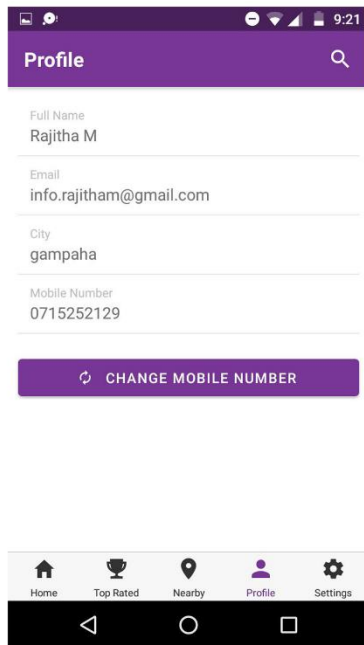
Client/Customer App Screens



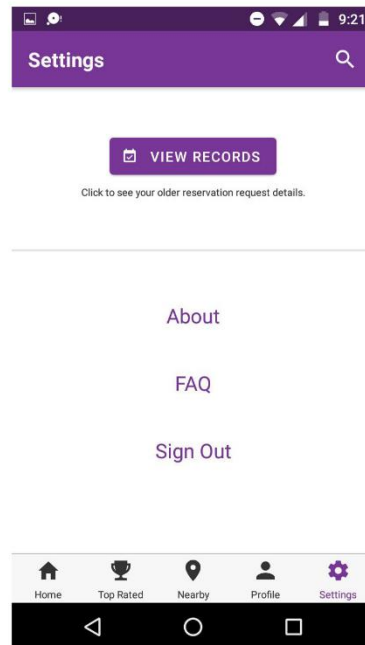
Home Screen

Service Station Details Screen

Confirmation Screen



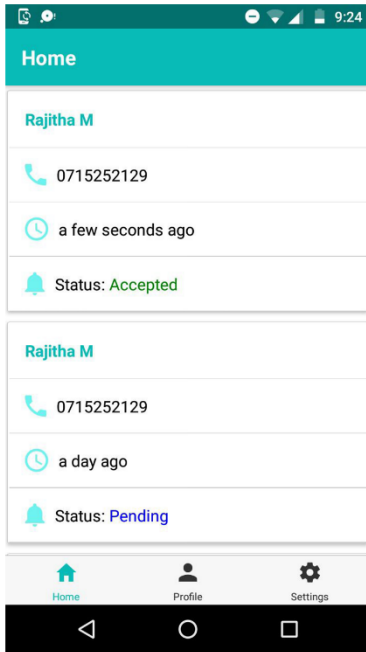
Profile Screen



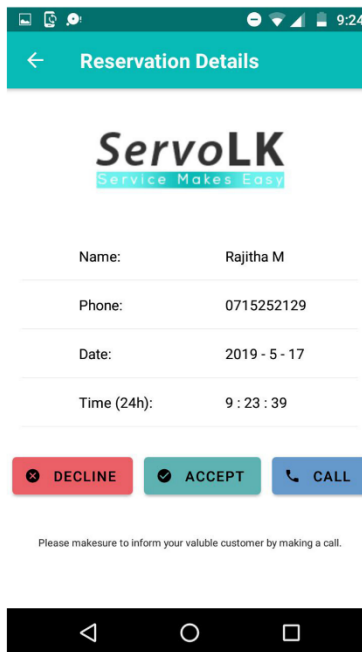
Settings Screen

Figure B. 5 : Client App Screens

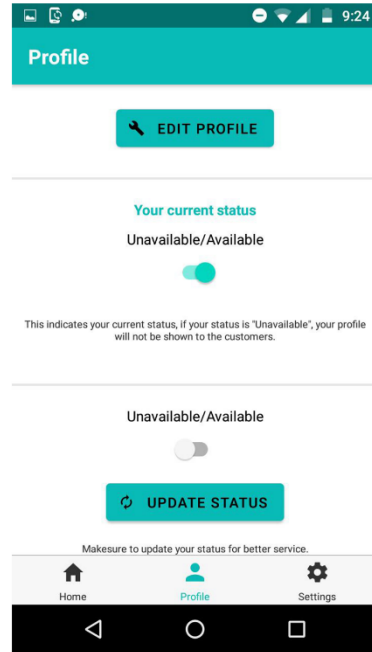
Service Station App Screens



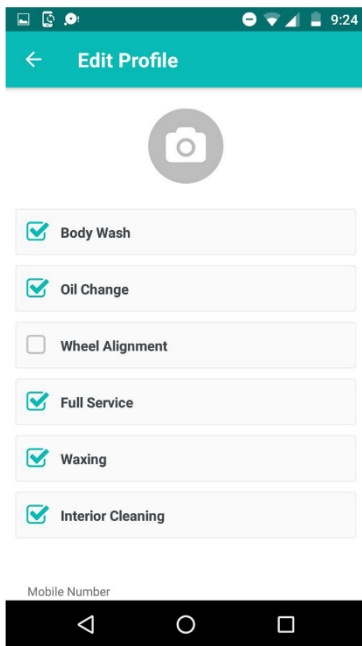
Home Screen



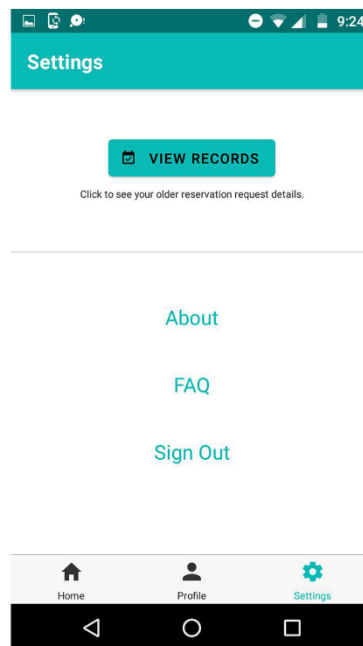
Reservation Details Screen



Profile Screen



Edit Profile Screen



Settings Screen

Figure B. 6 : Service Station App Screens

APPENDIX C – CODE LISTING

This section describes about the main code snippets of this project. Since the codes are very lengthy, only important code snippets are represented in this segment. The main programming language of this project is React Native and data base querying part was done by using Firebase which is cloud based NoSQL database which includes documents and collections rather than the tables of relational database architecture.

Rating Function

This function was created by using the aggregate function because in NoSQL databases there is no way of using join queries therefore de-normalizing techniques should be used to create that function. In Google Firebase there is a way of using aggregate function to get sub collection value to the main collection which is described in below.

```
function addRating(restaurantRef, rating) {
  // Create a reference for a new rating, for use inside the transaction
  var ratingRef = restaurantRef.collection('ratings').doc();

  // In a transaction, add the new rating and update the aggregate totals
  return db.runTransaction(transaction => {
    return transaction.get(restaurantRef).then(res => {
      if (!res.exists) {
        throw "Document does not exist!";
      }

      // Compute new number of ratings
      var newNumRatings = res.data().numRatings + 1;

      // Compute new average rating
      var oldRatingTotal = res.data().avgRating * res.data().numRatings;
      var newAvgRating = (oldRatingTotal + rating) / newNumRatings;

      // Commit to Firestore
      transaction.update(restaurantRef, {
        numRatings: newNumRatings,
        avgRating: newAvgRating
      });
      transaction.set(ratingRef, { rating: rating });
    });
  });
};
```

Location Identification Function

In this project there is a function to identify the user's current location and get that current location value to generate some dynamic queries. React Native which has a method to get user's permission and get the current location value to further computation which describes in follows.

```
componentWillMount() {
  if (Platform.OS === 'android' && !Constants.isDevice) {
    this.setState({
      errorMessage: 'Oops, this will not work on Sketch in an Android emulator. Try
it on your device!',
    });
  } else {
    this._getLocationAsync();
  }
}

_getLocationAsync = async () => {
  let { status } = await Permissions.askAsync(Permissions.LOCATION);
  if (status !== 'granted') {
    this.setState({
      errorMessage: 'Permission to access location was denied',
    });
  }

  let location = await Location.getCurrentPositionAsync({});
  this.setState({ location });
};
```

Reservation Function

In reservation function there is a method to get permission for push notification and send it to the service station application. To send the push notification first the particular user's push token should be generated and stored into the database that function will be described in below.

```
import { Permissions, Notifications } from 'expo';

const PUSH_ENDPOINT = 'https://your-server.com/users/push-token';

async function registerForPushNotificationsAsync() {
  const { status: existingStatus } = await Permissions.getAsync(
    Permissions.NOTIFICATIONS
  );
  let finalStatus = existingStatus;

  // only ask if permissions have not already been determined, because
  // iOS won't necessarily prompt the user a second time.
  if (existingStatus !== 'granted') {
    // Android remote notification permissions are granted during the app
    // install, so this will only ask on iOS
    const { status } = await Permissions.askAsync(Permissions.NOTIFICATIONS);
    finalStatus = status;
  }

  // Stop here if the user did not grant permissions
  if (finalStatus !== 'granted') {
    return;
  }

  // Get the token that uniquely identifies this device
  let token = await Notifications.getExpoPushTokenAsync();

  // POST the token to your backend server from where you can retrieve it to send push notifications.
  return fetch(PUSH_ENDPOINT, {
    method: 'POST',
    headers: {
      Accept: 'application/json',
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({
      token: {
        value: token,
      },
      user: {
        username: 'Brent',
      },
    })),
  });
}
```

APPENDIX D – TEST RESULTS

Acceptance Test Result

After completing the project implementation and basic testing, this app was introduced to selected people for conduct the acceptance test. There were six individual people, three of them are having their own service stations and other three people are vehicle users. They have given some time period to test and use the app and after the given time period, feedback form was sent to them for getting their feedback. The feedback form and the result will be shown in below.

Feedback Form URL: <https://forms.gle/iBxRUzpgetQ1iE9Y8>

ServoLK Feedback Form

Please add your genuine feedback to improve the service.

* Required

Which application are you using? *

Service Station

Client

Interfaces are simple and understandable? *

Strongly Agree

Agree

Neutral

Disagree

Strongly Disagree

Easily navigate through the app? *

Strongly Agree

Agree

Neutral

Disagree

Strongly Disagree

Functions are more reliable? *

Strongly Agree

Agree

Neutral

Disagree

Strongly Disagree

App provides sufficient information? *

Strongly Agree

Agree

Neutral

Disagree

Strongly Disagree

Overall impression about the app and service? *

Very Good

Good

Fair

Poor

Very Poor

Any other comments?

Your answer

Thanks for providing your valuable feedback

ServoLK 2019

SUBMIT

Never submit passwords through Google Forms.

Figure D. 1 : ServoLK Feedback Form

Feedback Result

Interfaces are simple and understandable?

6 responses

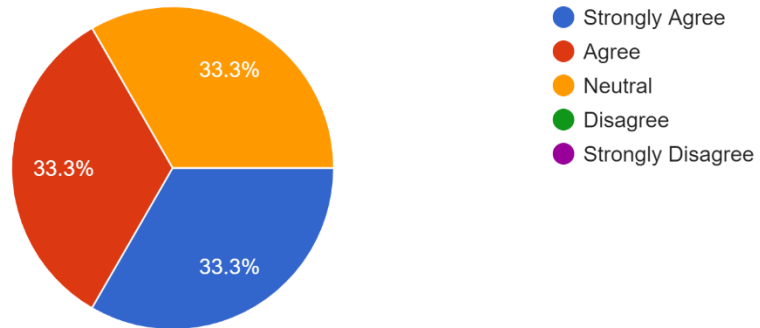


Figure D. 2 : Graphical Representation of Feedback (Interface)

Easily navigate through the app?

6 responses

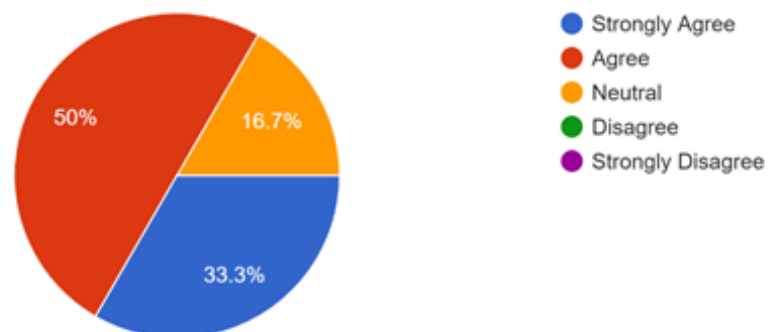


Figure D. 3 : Graphical Representation of Feedback (Navigation)

Functions are more reliable?

6 responses

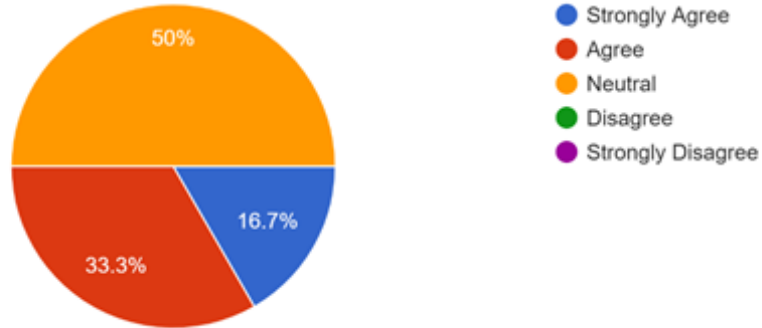


Figure D. 4 : Graphical Representation of Feedback (Functions)

App provides sufficient information?

6 responses

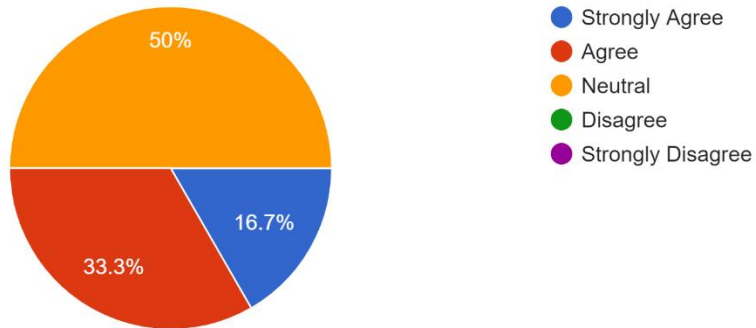


Figure D. 5 : Graphical Representation of Feedback (Information)

Overall impression about the app and service?

6 responses

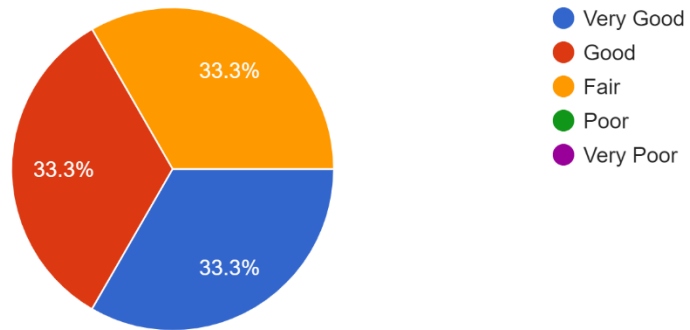


Figure D. 6 : Graphical Representation of Feedback (Overall)

The final feedback received from the survey was significantly positive. Some suggestions were taken and hope to consider them at the further enhancement stages.

GLOSSARY

Babel.js - Babel or Babel.js is a free and open-source JavaScript compiler and configurable transpiler used in web development.

Cloud - The cloud is a term referring to accessing computer, information technology (IT), and software applications through a network connection.

Compiler - A compiler is a computer program that translates computer code written in one programming language into another programming language.

ECMAScript - ECMAScript (or ES) is a scripting-language specification standardized by Ecma International in ECMA-262 and ISO/IEC 16262. It was created to standardize JavaScript.

Expo - A free and open source toolchain built around React Native to help you build native iOS and Android apps using JavaScript and React.

Firestore - Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.

GPS - It is a global navigation satellite system that provides geolocation and time information to a GPS receiver anywhere on or near the Earth.

Library - A library is a collection of precompiled routines that a program can use.

Node.js - Node.js is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code outside of a browser.

NoSQL - A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.

React Native - React Native is an open-source mobile application framework created by Facebook.

Time Stamp - A timestamp is a sequence of characters or encoded information identifying when a certain event occurred, usually giving date and time of day.

Transpiler - It is a type of compiler that takes the source code of a program written in a programming language as its input and produces the equivalent source code in the same or a different programming language.

TypeScript - TypeScript is an open-source programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript.

INDEX

A	
Android.....	v, xi, 3, 5, 6, 7, 23, 28, 30, 31, 48
app	2, 7, 13, 16, 23, 26, 27, 28, 42
B	
back-end	7
C	
code snippets.....	3, 39
collections.....	12, 36, 39
connecting.....	1, 2, 4, 6, 28
Customer...vi, ix, 2, 13, 14, 15, 16, 17, 18, 19, 33, 34, 35	
Customers	2, 4, 5
D	
database .iii, 5, 7, 8, 12, 23, 28, 29, 33, 35, 36, 39, 41, 48	
dependencies.....	iii, 12, 27, 28
Descriptions.....	vii, 33
design.....	vii, ix, 8, 23, 24, 33
diagram	ix, 8, 9, 10, 11, 23
F	
Firebase.....	iii, v, 7, 23, 28, 29, 30, 36, 39, 48
framework.....	6, 48
Functional.....	v, 4
G	
GPS.....	xi, 2, 5, 6, 28, 31, 34, 48
Graphical Representation.....	ix, 43, 44, 45
L	
library	6, 48
M	
maintenance.....	1, 29
mobile.....	1, 2, 3, 4, 5, 6, 7, 8, 13, 27, 28, 48
Mobile computing.....	6
N	
Non-functional.....	v, 5
NoSQL.....	iii, xi, 8, 12, 28, 29, 39, 48
P	
peer to peer	1, 2, 4, 6, 28
platforms.....	1, 4, 5, 6
project.....	iii, iv, 1, 2, 3, 4, 6, 8, 12, 13, 23, 24, 27, 28, 29, 39, 40, 42
R	
React Native .iii, v, ix, 6, 7, 12, 23, 27, 28, 30, 39, 40, 48	
Requests.....	2
Reservationvi, vii, viii, ix, x, 9, 11, 22, 26, 29, 34, 38, 41	
S	
screen 2, 13, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 31, 33	
Security.....	5
service....ix, 1, 2, 4, 5, 6, 7, 10, 13, 17, 18, 19, 26, 28, 29, 31, 32, 33, 34, 35, 36, 41, 42	
service station1, 2, 4, 5, 10, 13, 18, 19, 26, 28, 31, 33, 34, 35, 41	
smart phones.....	4
system...v, vii, 3, 4, 5, 8, 9, 14, 15, 23, 24, 25, 31, 33, 34, 35, 48	
System	xi, 2
T	
technologies.....	3, 6, 8, 27, 28, 29
testing	3, 6, 24, 30, 42
U	
User interface.....	3