



VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Sabin Shrestha

KILOMETER MANAGEMENT:
MOBILE AND DESKTOP APPLICATION

For the Employees of Pasi-Jakulet Oy

Technology and Communication

2017

ACKNOWLEDGEMENTS

First of all, sincere gratitude to our software programming teacher and my thesis supervisor, Dr. Ghodrat Moghadampour for his support, patience and guidance during my studies and in completing this project.

I want to thank my family and friends for their support, encouragement, and motivation.

I am grateful to Pasi-Jakelut Oy for providing a place to work and the office staff for co-operating with the idea of making this project. My sincere gratitude towards VAMK for being such an excellent platform and helping to reach out the goals of students and making it possible.

ABSTRACT

Author	Sabin Shrestha
Title	Kilometer Management: Mobile and Desktop Applications for the Employees of Pasi-Jakelut Oy.
Year	2017
Language	English
Pages	83+2
Name of Supervisor	Dr. Ghodrat Moghadampour

The traditional ways of exchanging information between various business operations is time-consuming and tedious, so the use of new technologies is necessary and growing rapidly. As a result, mobile applications as well as desktop applications, have gained popularity among users.

The idea of this project was to develop a hybrid mobile and desktop application that would offer the employee of Pasi-Jakelut Oy, an easy and efficient way to exchange information. A supervisor can use the desktop application at the office, and the deliverers can use the mobile application.

The desktop application allows the supervisor at the office to log in, manage kilometer details of the deliverers, add new deliverers, edit and delete the deliverers, add new areas, edit and remove areas, send messages as well as edit and delete messages. The mobile application allows the deliverers to log in with the login credentials provided by the supervisor, send kilometer details of their respective areas, view and send messages, update profiles and change passwords.

The desktop application was developed using PHP Desktop GUI Framework, and the mobile application was developed using Ionic 2 Framework. On the server side, PHP was used, and MySQL database was used to store the data.

CONTENTS

ABSTRACT

LIST OF ABBREVIATION	3
1 INTRODUCTION.....	5
1.1 Background.....	5
1.2 Motivation.....	5
1.3 Objectives	6
1.4 Description of Topic	6
2 RELEVANT TECHNOLOGIES	7
2.1 Hybrid Application	7
2.2 Ionic Framework.....	9
2.2.1 Ionic CLI.....	10
2.2.2 Components	10
2.2.3 Native.....	11
2.2.4 Theming	11
2.2.5 Navigation.....	11
2.2.6 Apache Cordova.....	11
2.2.7 Plugins	12
2.3 Angular JS	12
2.4 JavaScript and TypeScript	12
2.5 HTML and HTML5.....	14
2.6 Cascading Style Sheets	14
2.7 Hypertext Pre-processor	15
2.8 MySQL Database.....	15
2.9 JSON.....	16
3 APPLICATION STRUCTURE	17
3.1 Application Development Environment	18
3.2 Application Setup Process	19
4 APPLICATION DESCRIPTION.....	21
4.1 Quality Function Deployment (QFD).....	21
4.2 Use case diagram	23

4.2.1	Deliverer Use Case Diagram	24
4.2.2	Supervisor use case diagram.....	24
4.3	Class Diagram.....	26
4.4	Sequence Diagram.....	27
4.4.1	Supervisor’s Login Sequence Diagram	27
4.4.2	Supervisor add Area Sequence Diagram	28
4.4.3	Supervisor’s Edit Area Sequence Diagram.....	29
4.4.4	Supervisor’s Send Message Sequence Diagram.....	30
4.4.5	Supervisor’s Search Kilometer Details Sequence Diagram.....	31
4.4.6	Add User Sequence Diagram.....	32
4.4.7	Supervisor’s Edit User Details Sequence Diagram	33
4.4.8	Deliverer’s Login Page Sequence Diagram.....	34
4.4.9	Send Kilometer Details Sequence Diagram.....	35
4.4.10	View Supervisor’s Message Sequence Diagram	36
4.4.11	Deliverer’s Send Message Sequence Diagram	37
4.4.12	View Deliverer’s Account Sequence Diagram.....	38
4.4.13	Deliverer’s Update Profile Sequence Diagram.....	39
4.4.14	Deliverer’s Change Password Sequence Diagram.....	40
4.5	Component Diagram.....	41
5	DATABASE.....	42
5.1	Design of the Database:	42
6	GRAPHICAL USER INTERFACE DESIGN	44
6.1	Supervisor’s Graphical User Interface.....	44
6.1.1	Login Page	44
6.1.2	Home Page	45
6.1.3	New Area Page	46
6.1.4	Area List Page.....	47
6.1.5	New Message Page	48
6.1.6	Message List Page.....	49
6.1.7	Search Message Page.....	50
6.1.8	Search Kilometer Page.....	51
6.1.9	User List Page.....	52

6.1.10	Add New User Page.....	53
6.2	Deliverer's Graphical User Interface.....	54
6.2.1	Splash Screen.....	54
6.2.2	Login Page.....	55
6.2.3	Home Page.....	56
6.2.4	Message Page.....	57
6.2.5	Account Page.....	58
6.2.6	Change Password Page.....	59
7	IMPLEMENTATION.....	61
7.1	Supervisor Login.....	61
7.2	Supervisor's Home.....	62
7.3	Add User.....	64
7.4	Area.....	65
7.5	Messages.....	66
7.6	Kilometer Details.....	68
7.7	Deliverer's Login Page.....	68
7.8	Deliverer's Home Page.....	70
7.9	Deliverer's Message Page.....	72
7.10	Account.....	73
7.11	Change Password.....	75
8	TESTING.....	77
9	SUMMARY.....	82
10	CONCLUSIONS.....	83
10.1	Future Tasks.....	83
	REFERENCES.....	84
	APPENDICES	

LIST OF FIGURES AND TABLES

Figure 1: Overall Structure of Ionic Based Hybrid Apps	8
Figure 2: Application Structure /15/	18
Figure 3: Application Development Process	19
Figure 4: Deliverer Use Case Diagram.....	24
Figure 5: Supervisor Use Case Diagram.....	25
Figure 6: Class Diagram for the Mobile Application	26
Figure 7: Supervisor’s Login Sequence Diagram.....	27
Figure 8: Add Area Sequence Diagram.....	28
Figure 9: Edit Area Sequence Diagram	29
Figure 10: Send Message Sequence Diagram.....	30
Figure 11: Search Kilometer Details Sequence Diagram	31
Figure 12: Add New User Sequence Diagram.....	32
Figure 13: Edit User Details Sequence Diagram	33
Figure 14: Deliverer’s Login Sequence Diagram	34
Figure 15: Send Kilometer Details Sequence Diagram	35
Figure 16: View Supervisor’s Message Sequence Diagram.....	36
Figure 17: Deliverer’s Send Message Sequence Diagram.....	37
Figure 18: View Deliverer’s Account Sequence Diagram	38
Figure 19: Deliverer’s Update Profile Sequence Diagram	39
Figure 20: Deliverer’s Change Password Sequence Diagram	40
Figure 21: Component Diagram	41
Figure 22: ER Diagram.....	42
Figure 23: Database Tables.....	43
Figure 24: Supervisor’s Login Page	45
Figure 25: Supervisor’s Home Page	46
Figure 26: Add New Area Page.....	47
Figure 27: Area List Page	48
Figure 28: Add New Message Page.....	49
Figure 29: Message List Page.....	50
Figure 30: Search Message Page	51

Figure 31: Search Kilometer Page	52
Figure 32: User List Page	53
Figure 33: Add New User Page	54
Figure 34: Splash Screen	55
Figure 35: Deliverer's Login Page.....	56
Figure 36: Deliverer's Home Page	57
Figure 37: Deliverer's Message Page and Write Message Page respectively.	58
Figure 38: Deliverer's Account Page.....	59
Figure 39: Change Password Page.....	60
Table 1: Ionic Versions and Corresponding Supporting Native OSs.....	9
Table 2: Quality Function Deployment	22
Table 3: Testing for Mobile Application	77
Table 4: Testing for Desktop Application.....	79

LIST OF ABBREVIATION

OS	Operating System
IT	Information Technology
PHP	Personal Home Page
IDE	Integrated Development Environment
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
SDK	Software Development Kit
UI	User Interface
CLI	Command Line Interface
iOS	iPhone Operating System
API	Application Program Interface
MVW	Model View Whatever
XML	Extensible Markup Language
ES	ECMAScript
DB	Database
DBMS	Database Management System
RDBMS	Relational Database Management System
SQL	Structured Query Language
JSON	JavaScript Object Notation

CRUD	Create Read Update Delete
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ERD	Entity Relationship Diagram
URL	Uniform Resource Locator
VAMK	Vaasan Ammattikorkeakoulu
W3C	World Wide Web Consortium
WWW	World Wide Web

1 INTRODUCTION

The goal of the thesis is to develop kilometer management application for managing employee's kilometer details. This section explains background, motivation, and objectives of this report.

1.1 Background

In the recent years, expansion of mobile applications has been rapid as they are an efficient medium for extracting and exchanging information. This has made the way we communicate more advanced and user-friendly. Use of such technologies, therefore, complements not just the business but also everyday life. Social networking sites, online- shopping, e-banking, finding the desired music, sports information and news is just one application away. They are also being utilized by the companies to increase efficiency. Thus, there is a high demand of efficient, responsive and interactive mobile and web applications.

In the multiple mobile platforms, the mobile applications are developed and targeted to a particular mobile platform called native applications. However, not all the applications require a native functionality. In this case, hybrid mobile application development comes into a role. Hybrid applications have single language/framework base code, and with native wrapper's support, the output is according to desired mobile OS. They are relatively portable, fast and easy to build.

1.2 Motivation

The motivation for the project came as an idea that the IT-based application can contribute positively to the operations of the company. As an advertisement distributor at Pasi-Jakelut Oy (Suomen Suoramainota - SSM), some setbacks, as well as its feasible solution were realized personally. The delivery of advertisements is done twice a week i.e. on Wednesdays and Saturdays in the Ostrobothnia Region. Deliverers are assigned with their respective areas for their work. At the end of every month, if a car is being used for the delivery, the deliverer has to send kilometer details to the office for this is one of the bases on which salary is calculated.

Thus, deliverers have to be filled on a paper list which contains the deliverer's name, social security number, date of delivery, starting and ending address, total kilometer and other information. Most of the time, deliverers do not send kilometer details, and when they do, they use Microsoft Excel sheets where the information is not well managed. It is, therefore, difficult for the employee as well as the employer to manage kilometer details and problems arise while sending it further for the calculation of salary of the deliverers. Hence, this application may ease the burden of the employee.

1.3 Objectives

The primary goal is to develop a hybrid application, that is, a mobile based application to be used by the deliverers and a desktop based application to be used by supervisors at the office. The Ionic 2 frameworks and PHP Desktop GUI Frameworks will be used to develop the applications. The desktop application provides the supervisor with options to add new deliverer/supervisor, add a new area for the deliverer, send a message, check the latest kilometer details and find total kilometers of the deliverers' areas. For the deliverer to utilize the mobile application, the supervisors will provide login details which are registered in the system. The deliverer then can use the application features like sending kilometer details, sending a message and updating his/her personal information like email, phone, and address.

However, this thesis focuses more on details and process of building a hybrid mobile application using Ionic 2 framework where targeted mobile platform is Android.

1.4 Description of Topic

This project provides two different applications where a hybrid application based on Android mobile platform is used to send the kilometer details, and in the other a desktop application is used to track and manage the kilometer details.

2 RELEVANT TECHNOLOGIES

This section gives the description of the technologies that are used during development cycle and application structure.

2.1 Hybrid Application

The hybrid mobile app has the characteristics of native and HTML5 mobile applications. Before approaching hybrid mobile application developing, below is a description for Native and HTML5 mobile developing approach.

A native application is built for a particular platform and distributed from the app store. Various native mobile platforms use their free development tools and programming language. For example, Native iOS platform uses Objective-C, Android platform uses Java, Windows platform uses C#, etc. and they have their integrated development environment (IDE) to develop the applications. They have advanced interactive interfaces and give the best performance. Although they give the best performances and are expensive, they require more time to build as well the time and effort to maintain the application.

The html5 application is also known as a web application and it is stored in a remote server. The application is distributed over the internet through a browser as an interface. The web application uses web technologies, for example, HTML5, CSS, and JavaScript. It is possible to develop advanced and dynamic web applications that can run in the web browsers of desktop or mobile devices. However, the web application does not give the feel and the app store distribution capabilities as the native apps do.

So, the hybrid application is an HTML5/web application which is wrapped inside a "native container" so that it is possible to run on any operating system or devices. Some of the examples of the hybrid applications are Instagram, Twitter, Yelp, UBER, Gmail etc. The main advantages of the hybrid applications are that they take relatively less time to build, the cost is low and they are easy to develop and maintain, especially for the lighter applications than the native apps. Nevertheless, there

might be an unpleasant experience if the application grows bigger and contains more features.

Obviously, there is no complete solution to cope with all the requirements. Each mentioned processes has its own advantages and disadvantages. Native applications give high performance, best looking UIs, and other high-end features but they also require more time and cost while building an app. With the hybrid app, for web developers who are focused on web apps also get the opportunity to make a mobile application. The write once runs anywhere theme can be accomplished partially with the help of Ionic framework and the Apache Cordova platform. Apache Cordova provides excellent support for iOS, Android, Windows phone, etc. /1/

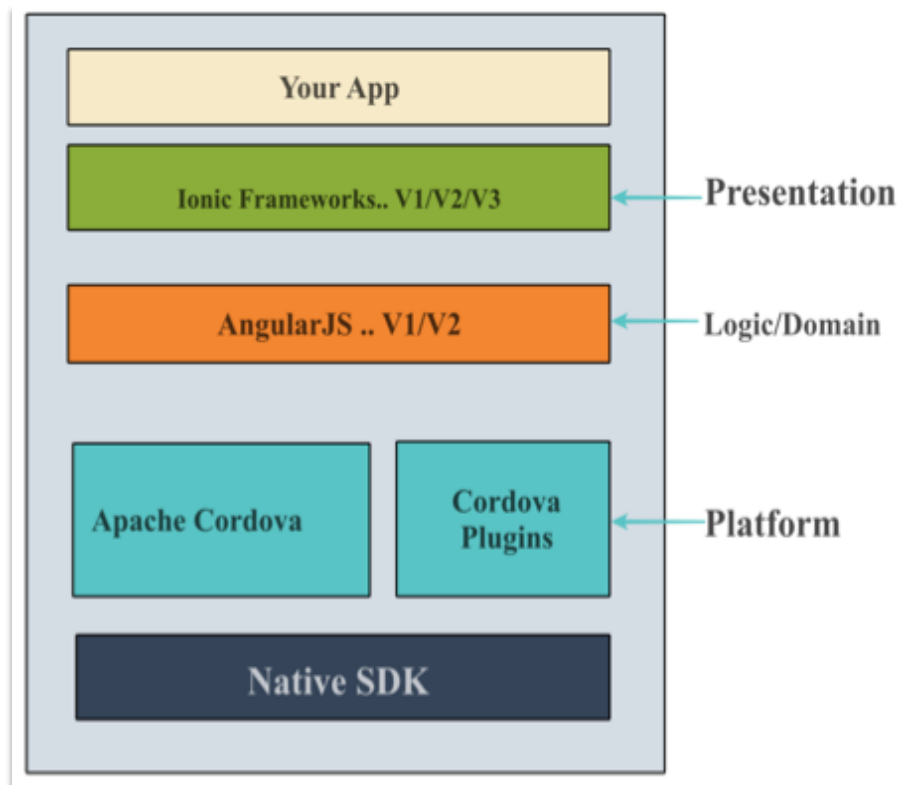


Figure 1: Overall Structure of Ionic Based Hybrid Apps /2/

The presentation layer is built with Ionic Framework, AngularJS is used for the app domain, and Apache Cordova is used for wrapping the native view.

2.2 Ionic Framework

Ionic is an open-source HTML5 SDK hybrid mobile developing framework created in 2013 by Drifty Co. It takes the help of AngularJS and Apache Cordova. Ionic provides a more efficient way to develop a hybrid application using web developing tools like HTML5, CSS and JavaScript. The front end application is mainly focused on the look and feel and UI interaction. Till now, there are three versions main of the ionic framework and Table 1 below shows their basic details.

Table 1: Ionic Versions and Corresponding Supporting Native OSs

Ionic Version	Description	Supporting OSs
Ionic 1	It gives priority to building native/hybrid mobile apps. It uses AngularJS 1.x, and it requires the in-depth knowledge of the framework.	<ul style="list-style-type: none"> • iOS 7+, Android 4.1+, • If Cordova Crosswalk is used, it can be run on older devices.
Ionic 2	Ionic 2 has a lot of core concept of ionic 1 but it is more stable and focused on native/hybrid apps as well as having ability for progressive web apps. The several version 2.0 has been released in 2016. Ionic 2 is built on top of AngularJS 2.x. It is compatible with TypeScript.	<ul style="list-style-type: none"> • iOS 8+, Android 4.4+, • Windows 10 Universal App, • If Cordova Crosswalk is used, the minimum target SDK can be pushed back to 4.1

Ionic 3	This is the Beta version which is published in April 2017 and still on the development process. It supports AngularJS 4.x. It focuses on overall developer experience, CLI speed, and the structure to add new plugins and platforms. The significant changes for example in CLI speed of V2 has been decreased from 150 seconds to around 10 seconds for V3.	<ul style="list-style-type: none"> • iOS 8+, • Android 4.4+, • Windows 10 Universal App, • If Cordova Crosswalk is used, the minimum target SDK can be pushed back to 4.1 • (no other information available till date).
---------	---	--

Ionic is free and open source. It released under a MIT license and developers can use it for their personal and commercial purposes. /3/

2.2.1 Ionic CLI

Ionic has its own command line interface (CLI) tool which provides a lot of useful commands to the developers. CLI is used for installing and updating Ionic applications as well as it has a built-in development server, build and debugging tools. Mac users have Terminal as a command-line-interface and for Windows users, it is a command prompt. /4/

2.2.2 Components

Components are one of the important building blocks of Ionic applications. It allows for creating UI of an application. Components are built with HTML, CSS and, if needed, JavaScript. The Components also take shape on the basis of where the app is running. Ionic has many components such as, popups, cards, checkbox, buttons, alerts, modals, and many more. /5/

2.2.3 Native

Ionic Native uses TypeScript package for Cordova plugins and provides the ability to add native resources into the ionic application. /6/

2.2.4 Theming

Themes provide a style to an app. Ionic has sets of styles and it has two defaults i.e. light theme and dark theme. Moreover, there are platform-specific styles that come from components and the style will vary according to the platform (iOS, Android, Windows, etc.). Ionic also provides CSS Utilities, Responsive Grid, Sass Variables, etc. and these all are used for styles and customizations of an application. /7/

2.2.5 Navigation

Navigation is used to navigate which is one of the main elements of every application. Ionic 1 Framework provides navigation through AngularJS UI Router. Navigation in Ionic 2 Framework application does not use AngularJS UI Router but uses its own navigation concept, which is a stack of pages. A page in Ionic 2 is a normal component seen on the screen of the device and it takes a whole screen so they contain everything that is visible. Ionic 2 manages the stack of such pages of such component in the background. /8/

2.2.6 Apache Cordova

Apache Cordova (updated version of PhoneGap) is a platform used for hybrid application development. It acts as an interpreter which supports and interacts with different device-specific APIs. It combines the features of all the native APIs into single JavaScript API that is accessible by the hybrid application. It simply loads up the JavaScript API when the mobile application is started and the output changes according to the platform of the device.

Inside the web view, hybrid applications are executed in the presence of Cordova. Cordova on top provides the concept of plugins which has the method to interact

with single or multiple native APIs and enables to use native device resources, like camera, notifications, basic file access, etc. /9/

2.2.7 Plugins

Plugins contain two parts. One is JavaScript which is running inside the WebView provides a nice view to the hybrid application and another plugins deal with platform specific native language. Cordova plugin supports at least iOS, Android, Windows and so on. The web view remains the same, and it is the complete screen running in the native container, which is used by Native Operating Systems. This means changes in the native containers occur according to the Operating System (OS). The Cordova libraries exchange information with the native framework of the corresponding OS.

For the mobile part of this project, Ionic 2 has been implemented as it has a lot more changes than Ionic 1, for example, organization and structure of the project, tooling and navigation. /10/

2.3 Angular JS

AngularJS is open source powerful, advanced featured JavaScript MVW Framework. It is used for the single page web app which provides structure to the application. It provides structure to the app domain model and finds a way to manage the app logic in a flexible way. This lets developers use HTML as a template language and extend HTML's syntax which is somewhat similar to XML where you can define your own custom elements and attributes. Also, data binding and dependency injection reduce most of the coding issues manually by developers. /11/

2.4 JavaScript and TypeScript

JavaScript or JS is a highly interactive, lightweight programming language and standardized according to ECMAScript language requirement. It is simple to implement but has great features which run on the client side of the web pages and also provides object-oriented capabilities. JavaScript, at the beginning known as LiveScript was later changed to JavaScript by Netscape. As the terms, 'Java' and

'JavaScript' are somewhat similar name, syntax and their documentations but they significantly differ with the design. In 1995, Netscape 2.0 launched JavaScript called LiveScript for the first time. The language has been used in Netscape, Internet Explorer, and other web browsers. /12/

Client-side JavaScript is a commonly used language. The JavaScript codes are written inside HTML codes so that browser understands it easily. Some of the advantages of JavaScript are:

- ❖ Server interaction decreases which means less traffic and burden on the server.
- ❖ Speedy response.
- ❖ Quick and improved interactivity.
- ❖ High level and dynamic user interfaces.

JavaScript has many useful features, but it is hard to maintain and reuse the codes when the application becomes larger. Moreover, JavaScript fails to grasp on some of its main features, like object orientation, multithreading capabilities and support for the networking applications making it difficult to succeed as server-side technology. Hence, TypeScript came on the front to fill some of these gaps.

TypeScript is a superset of JavaScript providing object-oriented features like optional static typing, classes, and interfaces. It is used in JavaScript Framework Angular 2.0 as well as in Ionic 2 Framework. Anders Hejlsberg designed the language at Microsoft. It has core concepts of JavaScript but provides some additional features. Below are some advantages because of which Ionic apps are written in TypeScript: /13/

- ❖ Types System is static and optional. The Type System supports various kinds of values and validates before the program executes them. Other helpful features includes using libraries and frameworks, where they provide APIs information to the developers. As the Type System is optional and it is not necessary to add Types, but in the case of larger and more complex application development, Types seems more helpful and efficient.

- ❖ TypeScript can integrate with a version of JavaScript supported in all browsers. It can downgrade itself and adopts basic features from the ECMAScript5 documentation. It also has access to ES6 and ES7.
- ❖ IntelliSense is defined as a tool for code completion tool, develop into Microsoft Visual Studio. TypeScript provides IntelliSense and gives hints as the coding starts. /14/

2.5 HTML and HTML5

HTML (Hypertext Markup Language) is a building block of web pages and web applications. Hypertext means the communication linked between web pages. It is Markup Language meaning that HTML simply used to 'Mark Up' a text with tags and provides information to the browser to display selected text bold or italic, text alignment, headings, hyperlinks and much more.

Tim Berners-Lee created HTML in 1990. The use of HTML makes it possible to exchange information on the web. HTML is like a skeleton that structures the content of a web page. For the dynamic web development, languages such as CSS and JavaScript provide layout and user interactivity. Server-side language like PHP connects a web page to the database and it can be embedded into HTML documents. /15/

HTML5 is the fifth and newest HTML version which has many more new features than HTML. HTML5 is a hybrid of HTML, CSS, and JavaScript codes which makes developer's work easier and faster. It is for developing hybrid mobile applications, and it supports many browsers. Some of the features of HTML5 are, nav, header, section, footer, email inputs and placeholder.

2.6 Cascading Style Sheets

Cascading Style Sheets (CSS) provides beautiful user interface looks on top of HTML codes. For example, CSS offers colours, fonts, height, width, line, background images and much more. It provides more options for designing layouts, and it runs in all browsers. /16/

2.7 Hypertext Pre-processor

Hypertext Pre-processor (PHP) is an open source server-side scripting language and it can be download from the official PHP website. PHP is mostly used for developing dynamic and interactive web pages. PHP is simple, efficient and flexible. Furthermore, it is easy to understand and secure. It has ".php" file extension which runs on various computers and operating systems like Mac OS X, Windows, and Linux. Below are some functions and features of PHP: /17/

- ❖ PHP helps to create dynamic web pages.
- ❖ PHP is a server-side programming language which means it gain access to the database, e.g. Oracle MySQL, MS-Access, SQLite, and so on.
- ❖ PHP helps to retrieve data.
- ❖ Handling cookies.
- ❖ PHP helps to secure the database information.
- ❖ PHP provide functions, like to add, to update, to insert, to delete in the database.
- ❖ User accessibility.

2.8 MySQL Database

A database is a set of data and information which are well organized and kept systematically so that data is easy to access, update or delete according to the system requirement. With the increase in storage and exchange of information and data increases the databases. Hence, DBMS is used to create and manage those databases. DBMS facilitates its users with an organized way to create, manage and update data. There are many types of DBMS such as RDBMS, NoSQL DBMS, IM-DBMS and CDBMS, for example.

SQL is the standard language for handling the relational database where SQL executes a query against the database and performs various functions like insert, update, delete and retrieve the records. It is also possible to create a new database and table by using the query. RDBMS is based on SQL. Some of the examples of

RDMBS are MySQL, SQL Server, MS Access, Oracle Database, SQLite, and much more. These are used for server side scripting language like Java, PHP, Python, etc.

MySQL is a type of relational database management system (RDBMS). It is fast, efficient in handling a large volume of data and used in small and big organizations. MySQL is free and easily downloaded from its official website <https://www.mysql.com/>.

2.9 JSON

JSON is one of the ways to store and exchange information in a well-organized, efficient way. It exchanges the data between a server and web application. It provides the data that are easy to read and understand. JSON is used in several programming languages such as PHP, Java, AJAX, Python, etc.

JSON contains two main components that are keys and values called key/value pair. String defines the key which are enclosed in quotation marks whereas value has no boundaries. A value has properties such as a string, object, number, double, integer and so on. Key/Value pair has a particular syntax pattern of key, colon and value. Commas separate the pairs. A simple example of JSON is shown below:

```
{ "teacher" : [
    { "name" : "sabin", "school" : "VAMK" },
    { "name" : "shrestha", "school" : "VAMK" } ] }
```

3 APPLICATION STRUCTURE

The project contains two different applications which are installed on two separate devices. An Android application is used only by the deliverers and the desktop application is used only by the office supervisors installed on the Windows computer. Hence, each mobile and desktop application has two different parts, the client side and the server side.

The client side handles the user's interfaces, and it shows the data or result which is requested by the clients or the users. On the other hand, the server-side handles the data exchanging and data storing processes where MySQL database has been implemented.

The users' data are saved inside MySQL database. On the server side, PHP APIs are implemented as they help to exchange data with Android device. The data is exchanged from MySQL database to the server side and to the mobile application in JSON format. The mobile application itself cannot communicate with the server directly, so the use of PHP APIs or any of these kinds of services are obligatory. The desktop application can interact with the database with the implementation of CRUD (Create, Read, Update and Delete) grid. PHP is implemented as a server-side and HTML is used for the client side or front end. The steps and diagrams below shows the descriptions of the interaction between client and server side:

- ❖ Communication starts when client makes a request HTTP POST/GET method to a server.
- ❖ The PHP APIs process the connection through SQL queries to MySQL server.
- ❖ To check errors, PHP APIs echo some JSON data and if the conditions are true, the successful output will be shown on the mobile application.

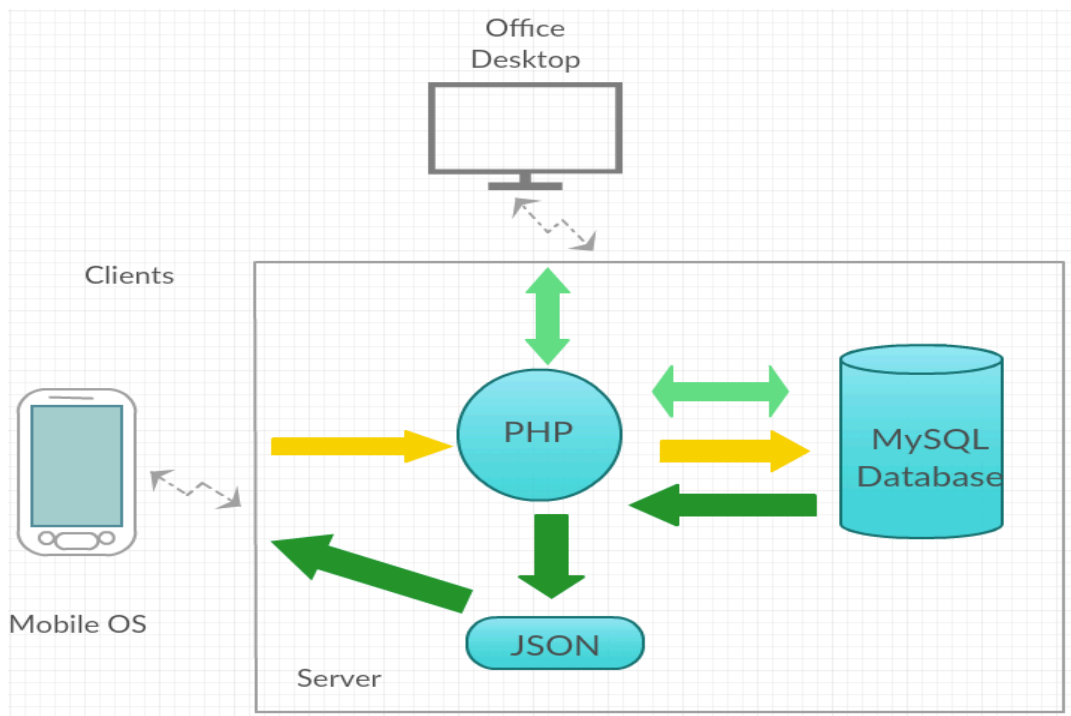


Figure 2: Application Structure

3.1 Application Development Environment

Before starting to build any applications, development environments have to be setup. To obtain the goals of the project, the right tools and technologies are needed. For developing Ionic apps, Node.js latest version needs to be installed which provides the most recent version of Ionic CLI, and it is possible to install Cordova through CLI. The other used tools and technologies are given below:

Hardware

- MacBook Pro Mid 2012 Version 10.12.4.
- One plus 3 Android phone.

Software

Operating Systems: OS X macOS Sierra, Windows 10

IDE: Visual Studio Code, Command Line Tool

- ❖ Ionic 2 frameworks for mobile application, PHP Desktop for desktop application and Inno Setup Installer for creating the .exe executable file.
- ❖ Node.js version 6.10.0

Xampp is used as a local web server. It provides MariaDB, PHPMyAdmin, etc. It can be download from (<https://www.apachefriends.org/download.html>)

3.2 Application Setup Process

The application development process for mobile application has the following phases shown in Figure 3.

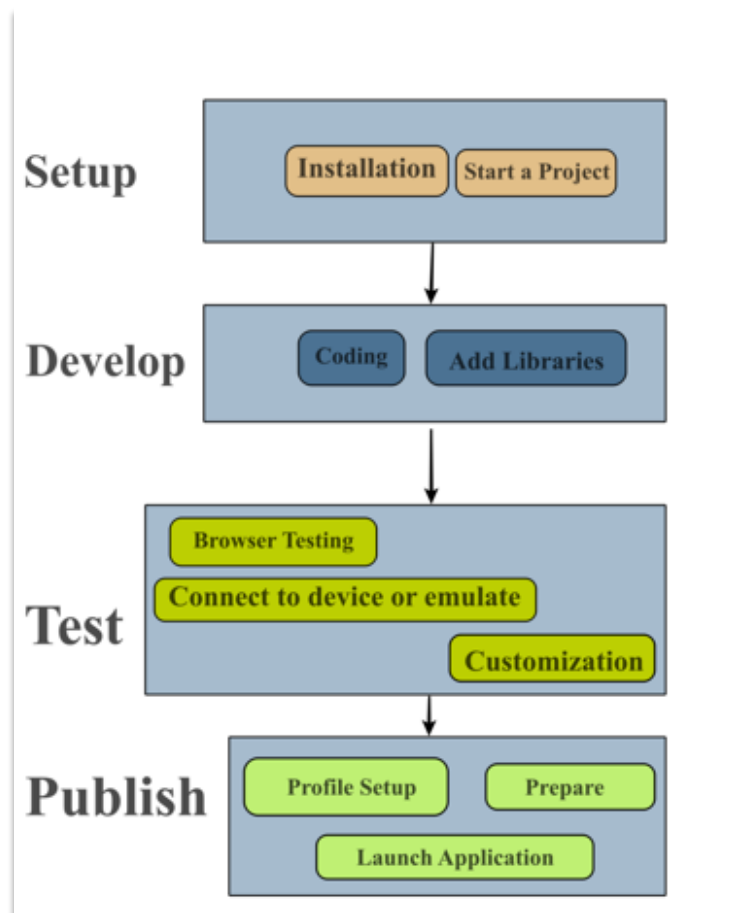


Figure 3: Application Development Process /18/

Setup Phase: In this phase, Ionic CLI and Apache Cordova are installed after installing node.js. The following command is used to install Cordova and Ionic from Terminal where g stands for global.

```
$ sudo npm install -g ionic cordova
```

To install TypeScript:

```
$ npm install -g typescript
```

To create a new project:

```
$ ionic start KilometerManagement --V2
```

After creating a project, add native platform Android or iOS using the following command:

```
$ ionic platform add android  
$ ionic platform add ios
```

The native SDKs should be installed for the native platforms to build. For example, for Android, Android SDK should be installed on the computer.

Develop phase: In this phase, the project is opened using Visual Studio Code editor and the project can be improved for example by adding and writing clean codes, designing user interfaces and adding libraries.

Test Phase: This phase helps the developer whether an application has correct contents or not. With the support of Ionic CLI commands, the test can be done in the browser, built and run or emulated in a device.

Publish Phase: This phase describes review the configuration of the application, the release key and tests in the release mode. Also, for the individual new application developer, it is necessary to create a developer account so that they can publish and sell their applications.

4 APPLICATION DESCRIPTION

In this section, the overall description of the system is explained. Various requirements of the application and their interaction with the system and other entities are also described. The section gives detailed information about using an application and how the application co-operates with the user. The application description provides the information associated with quality function deployment, class diagram, sequence diagram, activity diagram and component diagram of the application. This project has two different applications installed on two separate devices. The kilometer management desktop application for the supervisor at the office and kilometer management mobile application for the deliverers.

The desktop application allows the Supervisors at the office to log in with their login credentials. The supervisor can view the latest kilometer status as well as search for the kilometer details. The supervisor can add, update and delete advertisement delivery areas for the deliverers. The supervisor can send the message to the deliverer as well as view the message information sent by the deliverers. The supervisor can add a new deliverer, which is a registration process for the deliverers as well as act as an admin (if required) and update the information of the existing deliverers. The office user can exit using the logout option.

The mobile application for the deliverer allows the users to login via login credentials given by the office. The deliverer can view home page, message page, and update page. The deliverer can select his/her working areas, select advertisement delivery date, enter total kilometer obtained after delivering the ads in the areas and send it to the office. On the message page, a deliverer can view the messages and information sent by the Supervisor, and he/she can reply as well. On the update page, a deliverer can update his/her email, phone number, address and can change the password. A deliverer can exit using logout button.

4.1 Quality Function Deployment (QFD)

Quality Function Deployment (QFD) is a technique to organize the users' requirements as well as to insure that all the requirements are fulfilled. It helps the project

developers and the team to act on a certain project by providing information according to the level of importance given to certain requirements. There are three types of requirements, and they are Normal requirements, Expected requirements and Exciting requirements. Table 2 provides requirements and information related to this project's Quality Function Deployment.

Table 2: Quality Function Deployment

Normal Requirements (Priority level - 1)
<ul style="list-style-type: none"> • The deliverer must be able to sign in through username and password given by the supervisor at the office. • The deliverer must be able to view and select available tabs (Home, Message and Account). • The deliverer must be able to select working date, time and working area/s. • The deliverer must be able to input total distance obtained while delivering the ads and send it to the supervisor at the office. • The deliverer must be able to view message received from the office inside message page and reply or must be able to send message to the office. • The deliverer must be able to send message to the office using message icon. • The deliverer must be able to change the password. • The deliverer must be able to logout from account page. • The supervisor at the office must be able to sign in with the provided credentials. • The supervisor must be able to see the latest kilometer updates. • The supervisor must be able to view list of deliverers and their detail information. • The supervisor must be able to view list of areas and their information. • The supervisor must be able to add, delete and edit working areas.

- The supervisor must be able to view the total kilometer details.
- The supervisor must be able to view messages send by the deliverer and send messages back to the deliverer.
- The supervisor must be able to update or delete the messages.
- The supervisor must be able to view list of users (deliverer or supervisor) and users' basic information (username, email, name, phone, type)
- The supervisor must be able to add, edit and delete the user.

Expected Requirement (Priority Level - 2)

- The supervisor should be able to see error message if login fails.
- The deliverer should be able to see error message if login fails.
- The mobile and web application should be easy to use and user friendly to the users.

Exciting Requirements (Priority level - 3)

- The deliverer can update his or her information (mobile number, email address and address) inside account tab.
- The supervisor can search deliverers kilometer details from search more option.

4.2 Use case diagram

A use case diagram is a simple illustration of the relation between the user and the system. In this project the actors are the deliverer and the office supervisor.

4.2.1 Deliverer Use Case Diagram

The deliverer use case diagram illustrates the interaction that occurs between the deliverer and the system. The deliverer gets a user name and a password from the supervisor at the office. After logging in, the deliverer can view his/her profile that allows adding date and time of the delivery day, to add working area/s, to enter total kilometers and send to the total kilometers. The deliverer can view the messages sent by the office and reply as well. It is also possible for him/her to update the profile, change the password and log out. Figure 4 shows the deliverer use case diagram:

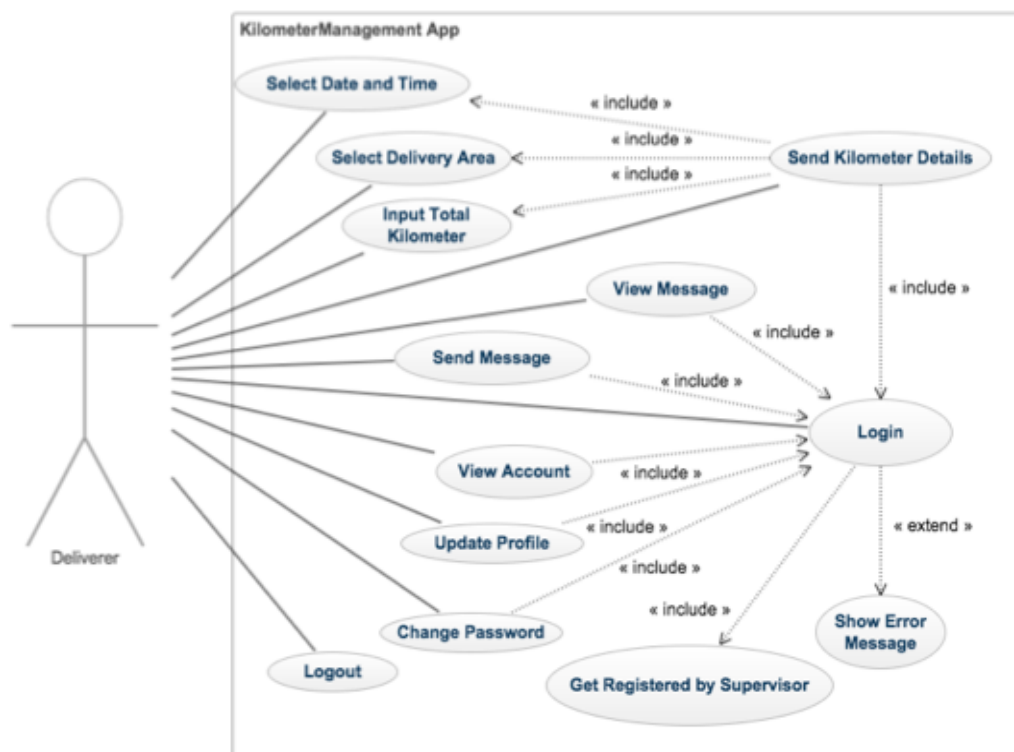


Figure 4: Deliverer Use Case Diagram

4.2.2 Supervisor use case diagram

The supervisor use case diagram illustrates the interaction that occurs between the supervisor at the office and the system. After login, the supervisor at the office can view all the information accessible inside the system. The supervisor can view the latest kilometer details on the dashboard and can view more details by clicking on

the search more option. The supervisor can add a new area, view list of areas, edit or delete areas, send a new message, view a list of messages, edit or delete message details, search kilometer details, add new supervisor/deliverer and view a list of supervisors/deliverers as shown in Figure 5.

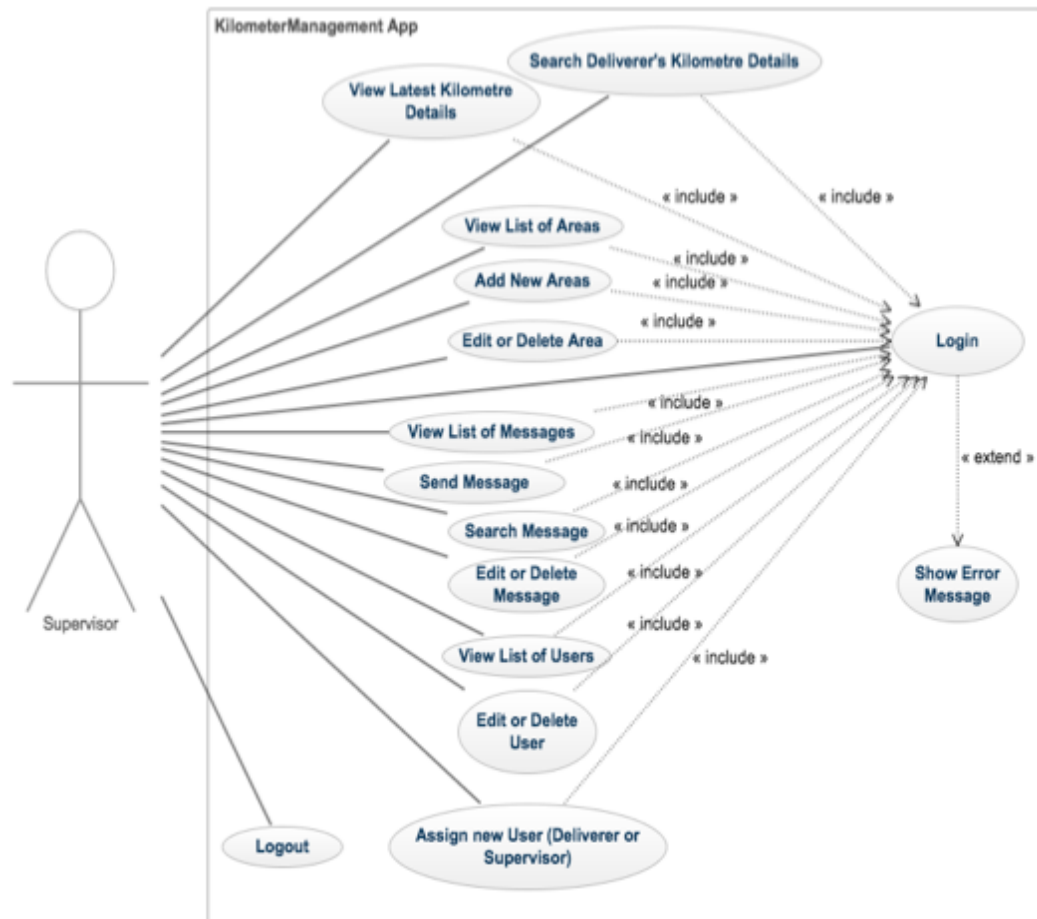


Figure 5: Supervisor Use Case Diagram

4.3 Class Diagram

A class diagram describes a system's classes, attributes, properties, methods and the association among the objects. The class diagram shows the static view of an application. The class diagrams can be mapped directly with OOP languages and used for modelling object-oriented systems. Figure 6 shows the classes that are used in the application.

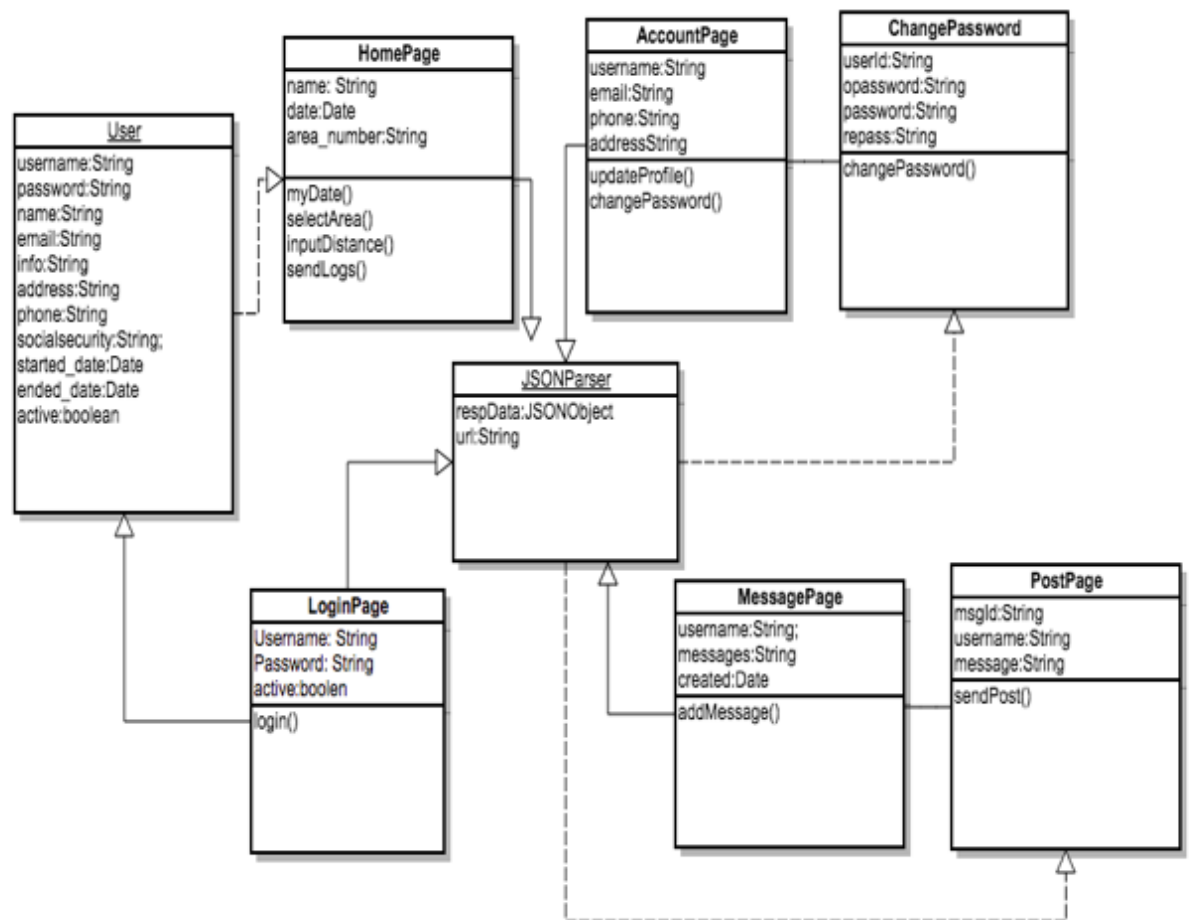


Figure 6: Class Diagram for the Mobile Application

4.4 Sequence Diagram

The sequence diagram is an interactive representation in use case diagrams which describes the various processes used in building the application. It provides information about the interaction between different objects and services inside the system. The project consist of several sequence diagrams which are given below:

4.4.1 Supervisor's Login Sequence Diagram

This sequence diagram shows the steps for logging into the application. As the supervisor starts the application, the application shows the login page. The supervisor enters the username and password. If the entered login credentials are correct and matched in the database, the login process is successful and the home page can be viewed. If the login credentials are not correct, the application shows an error message.

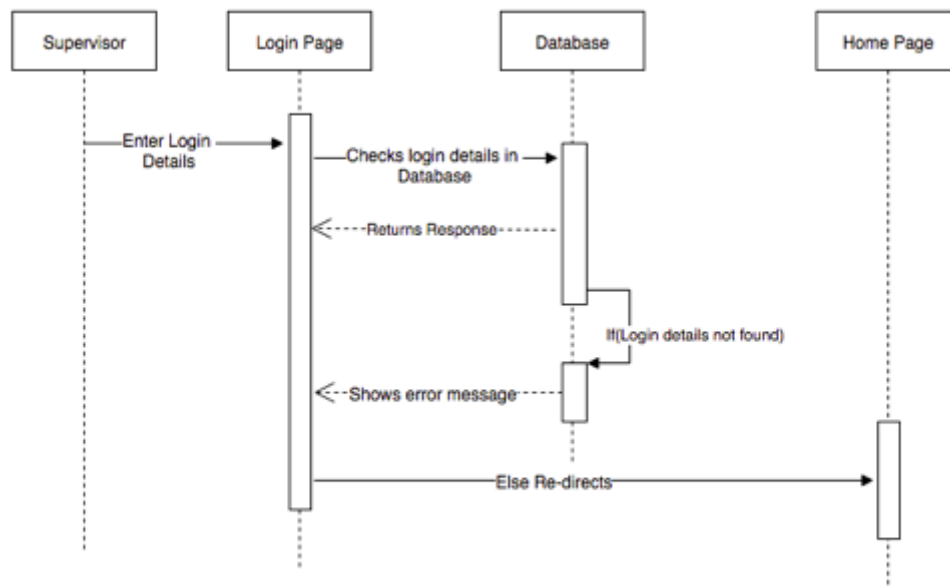


Figure 7: Supervisor's Login Sequence Diagram

4.4.2 Supervisor add Area Sequence Diagram

This sequence diagram shows the various steps for adding a new area. After logging in to the application, the supervisor clicks the add area option, and it redirects to adding a new area page for the supervisor. The supervisor can enter a new area number, select 'Yes' to publish or 'No' to not publish and click on submit button. Clicking on the submit button sends the query to the database, and the entered data is checked. If the data is not correct, the application returns an error message. Otherwise new area is added to the list.

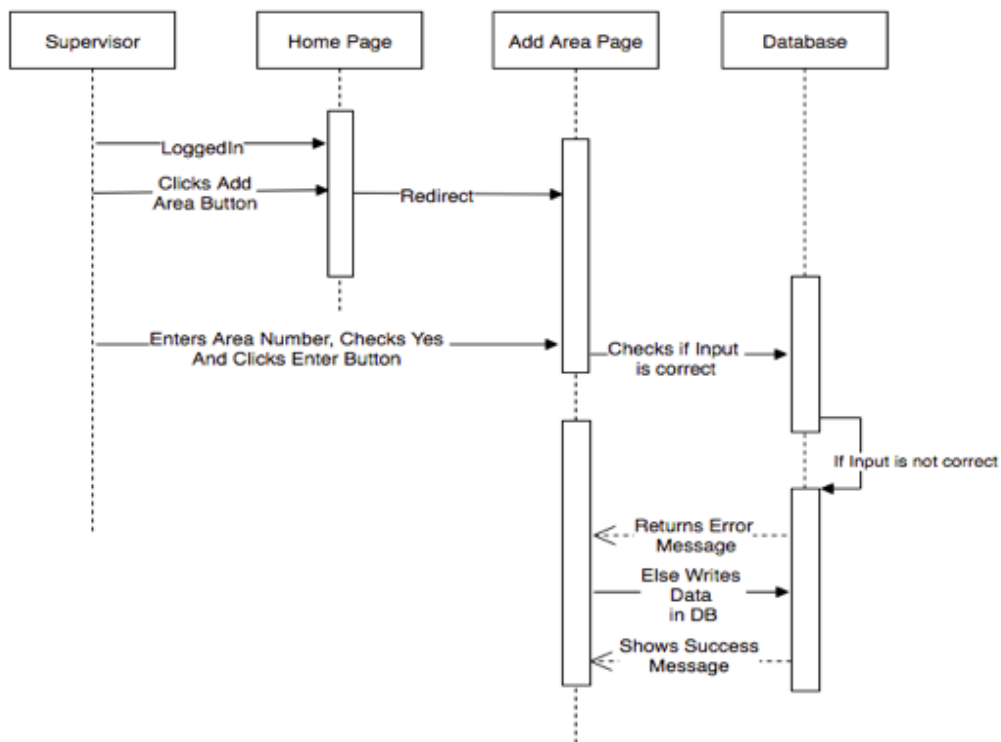


Figure 8: Add Area Sequence Diagram

4.4.3 Supervisor's Edit Area Sequence Diagram

This sequence diagram shows the steps for editing the area number. After login, the application shows the home page to the supervisor where the supervisor can click on the area list option. The application redirects the supervisor to the edit area page. The supervisor can rename the area number or can decide whether to publish to the deliverer's mobile device or not.

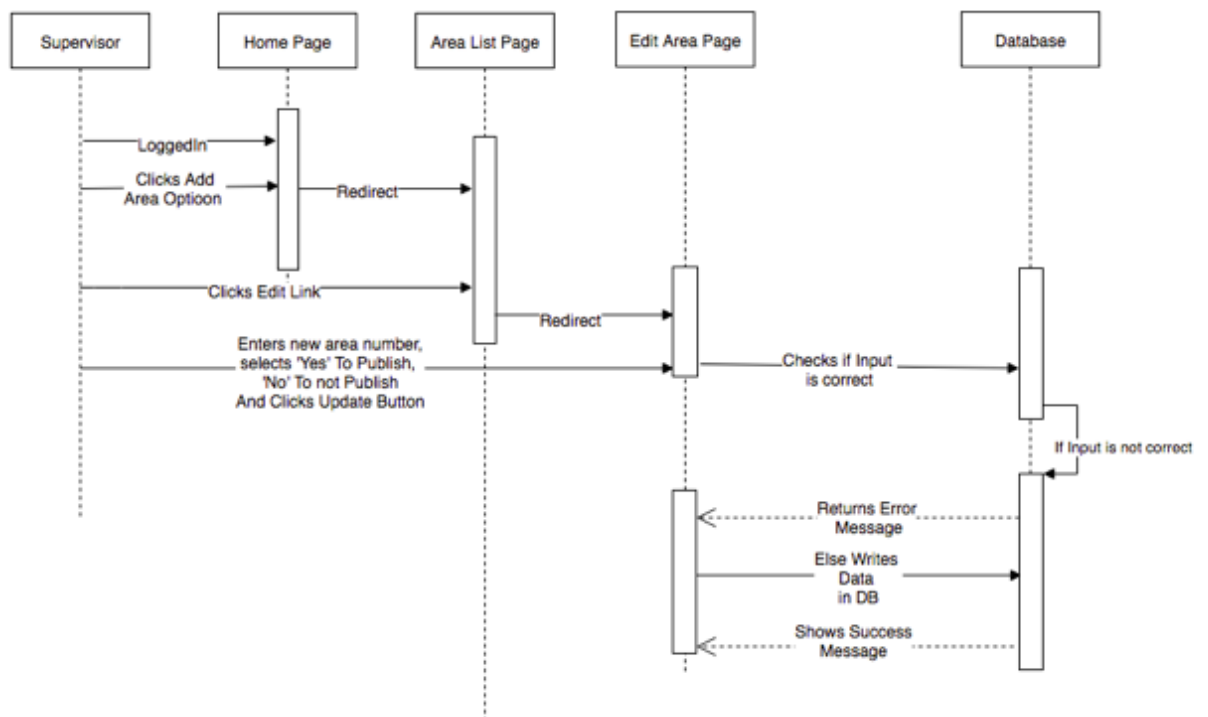


Figure 9: Edit Area Sequence Diagram

4.4.4 Supervisor's Send Message Sequence Diagram

This sequence diagram involves the various steps to sending a message to the deliverer by the supervisor. The supervisor after logging in, views the home page where he/she finds the send message option. The supervisor clicks on that option and the application redirects to 'send a new message' page. The supervisor can select a username, select yes or no to publish, write the message and click on the send button.

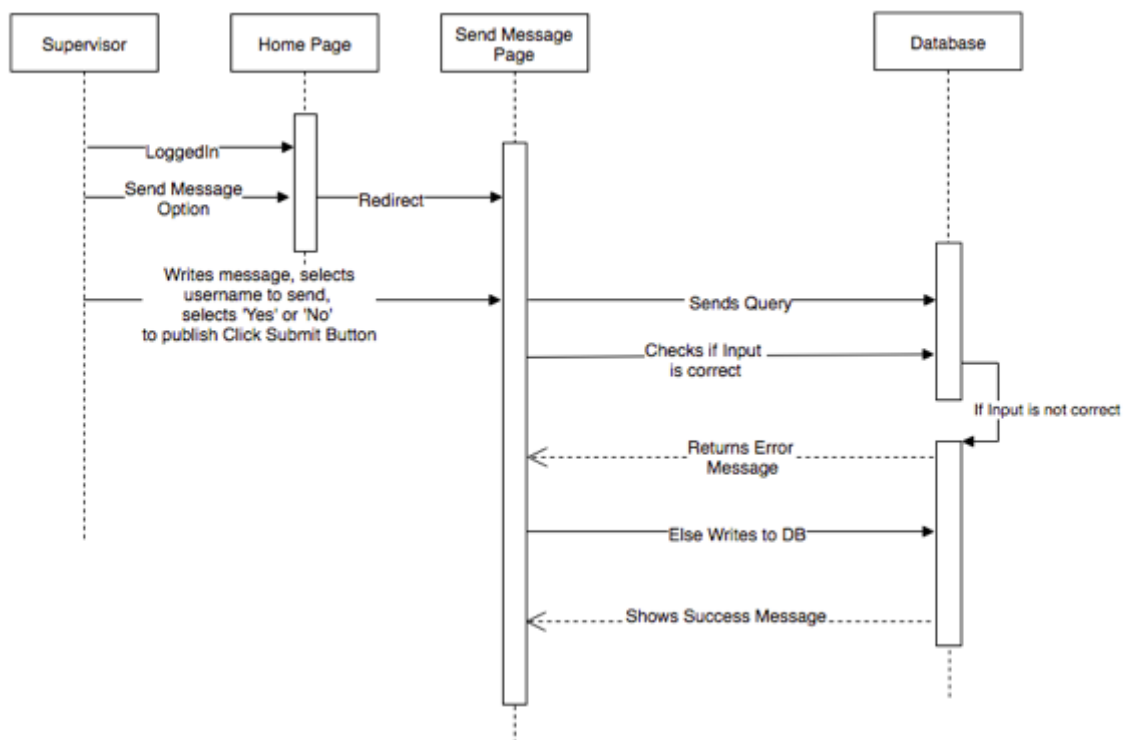


Figure 10: Send Message Sequence Diagram

4.4.5 Supervisor's Search Kilometer Details Sequence Diagram

This sequence diagram shows the steps for searching kilometer details. After the login, the supervisor views the home page where he/she clicks on the kilometer details option, and it redirects the user to the search kilometer details page. The application provides the total kilometers where the supervisor has to select the username, enter the start and end date and click on the submit button. Clicking the submit button sends the query to the database, and it checks if the input is correct or not. If the input is not correct, it shows an error message. Otherwise it checks the data in the database and returns the details.

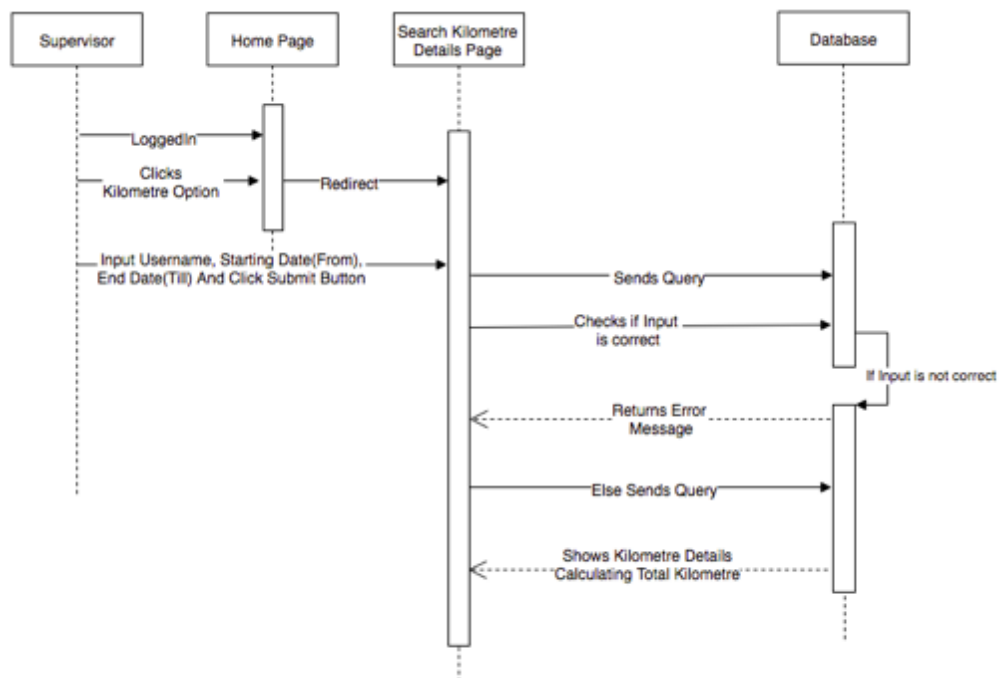


Figure 11: Search Kilometer Details Sequence Diagram

4.4.6 Add User Sequence Diagram

This sequence diagram shows the steps to adding a new user. Users are either deliverers or supervisors. After login to the application, the supervisor views the home page where he/she clicks on add user option and it redirects to the add a new user page. The supervisor can add a new user's details (Name, Address, Phone, Email, Social Security, Username, Password, Job Started, Job Ended, User Type: Supervisor or Deliverer) and click on the submit button. Then the query is sent to the database, and at first, the entered credentials are checked to see if they are correct or not. If not, then application shows an error message related to the correct input otherwise the database writes the new data or user details and returns a success message.

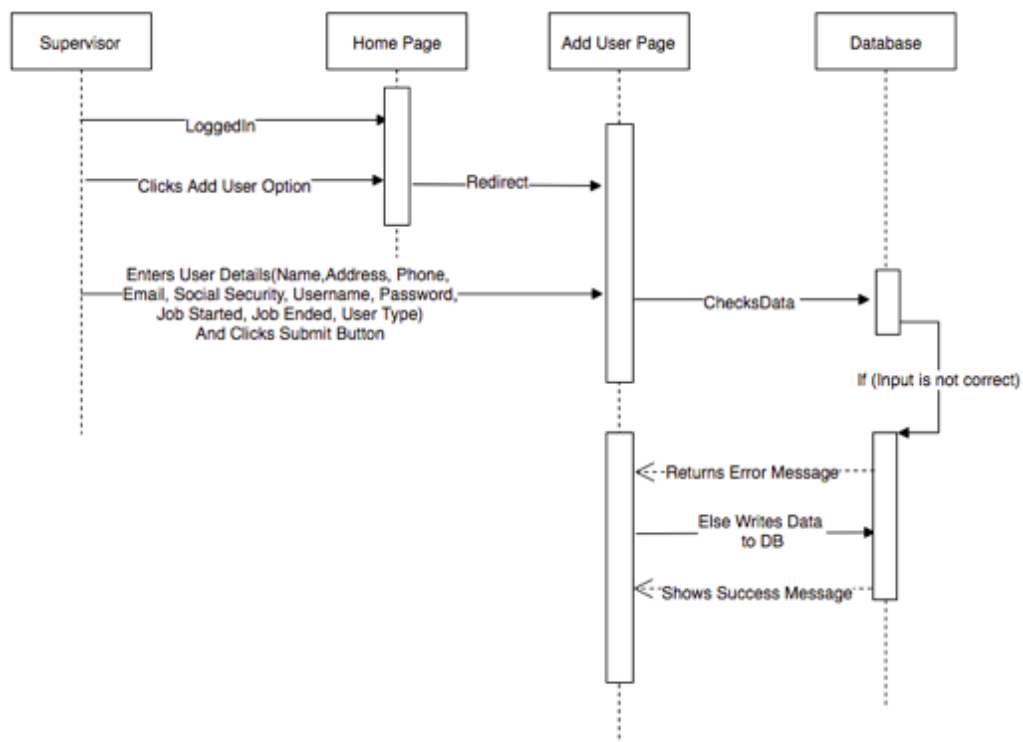


Figure 12: Add New User Sequence Diagram

4.4.7 Supervisor's Edit User Details Sequence Diagram

This sequence diagram shows the various steps related to editing user details. After the login, the supervisor views the home page where he/she clicks on the user list option. On the user list page, the supervisor clicks on the edit link option and it redirects the supervisor to the edit user page. The supervisor can edit any details of the user (Name, Address, Phone, Email, Social Security, Username, Password, Job Started, Job Ended, User Type) and clicks on the submit button. Clicking on the submit button sends a query and checks if the entered details are correct. If the user inputs are not correct, the application shows an error message or it re-writes the user details to the database and returns a successful message.

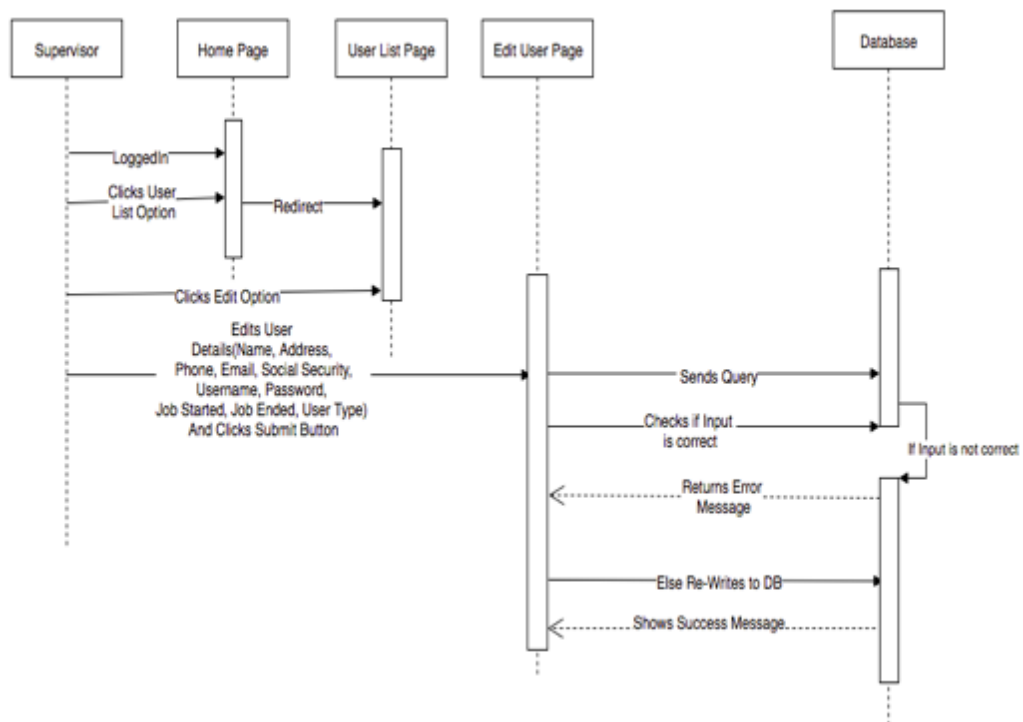


Figure 13: Edit User Details Sequence Diagram

4.4.8 Deliverer's Login Page Sequence Diagram

In this sequence diagram various steps required for the users to login into the application are explained. The deliverer's login details are entered into the application. The login details need to be correct while entering into the application. As the login details are entered, the login button can be tapped; the request goes to Validate User which checks the details in the database. If the data is found in the database, the deliverer can view the home page; if not the applications shows an error message. Figure 14 shows the deliverer's login sequence diagram.

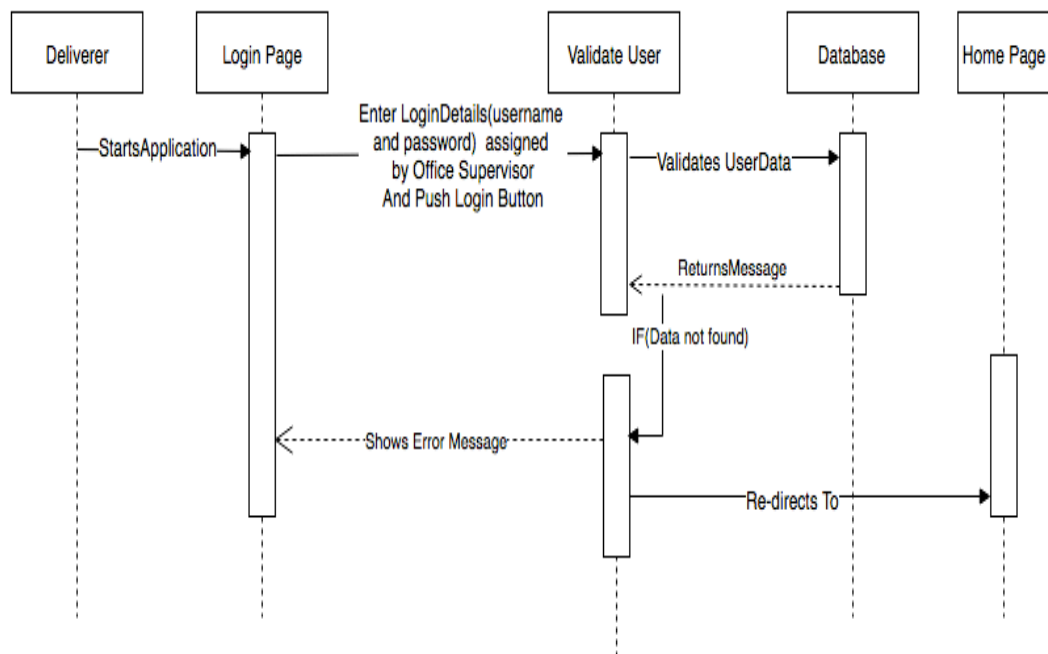


Figure 14: Deliverer's Login Sequence Diagram

4.4.9 Send Kilometer Details Sequence Diagram

This sequence diagram shows how the deliverer can send kilometer details. Once the deliverer is logged in, the deliverer is re-directed to the home page where he/she can select the date, delivery area number, enter total kilometers of the delivered area number and then tap the send button. When the send button is tapped, the query goes to Validation. If the deliverer tries to send kilometer details the without entering kilometer or selecting his/her delivery area number, the application does not allows it. Otherwise, if the correct details are entered, the application saves them to the database as shown in Figure 15.

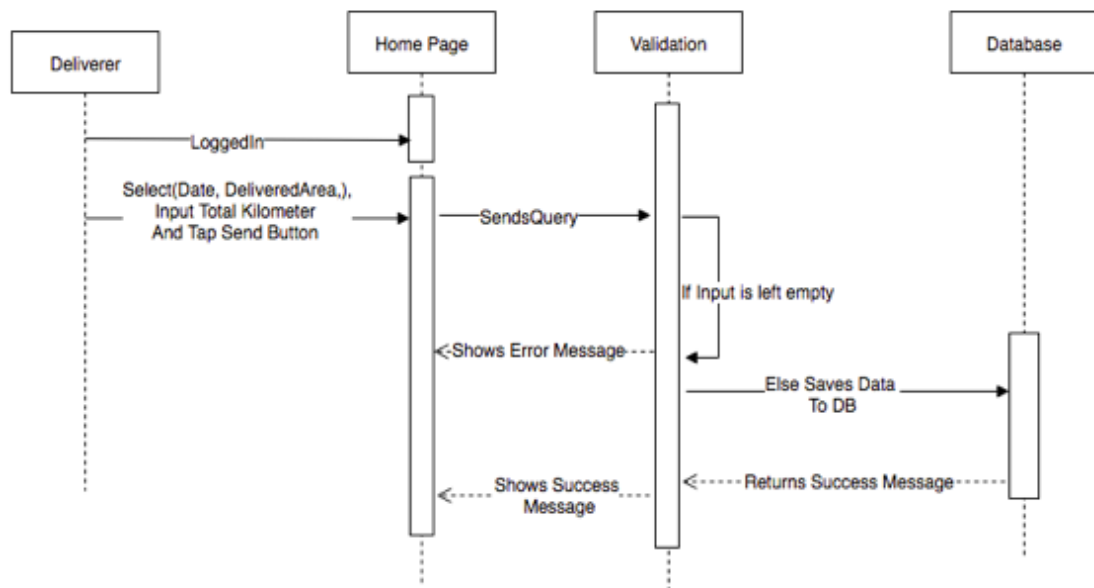


Figure 15: Send Kilometer Details Sequence Diagram

4.4.10 View Supervisor's Message Sequence Diagram

Sequence diagram (Figure 16) shows the steps to be followed by the deliverer to view the message/s sent by the supervisor. As the deliverer is logged in, he/she can see the home page and then can tap the message page. The deliverer can view the message if it has been sent by the supervisor.

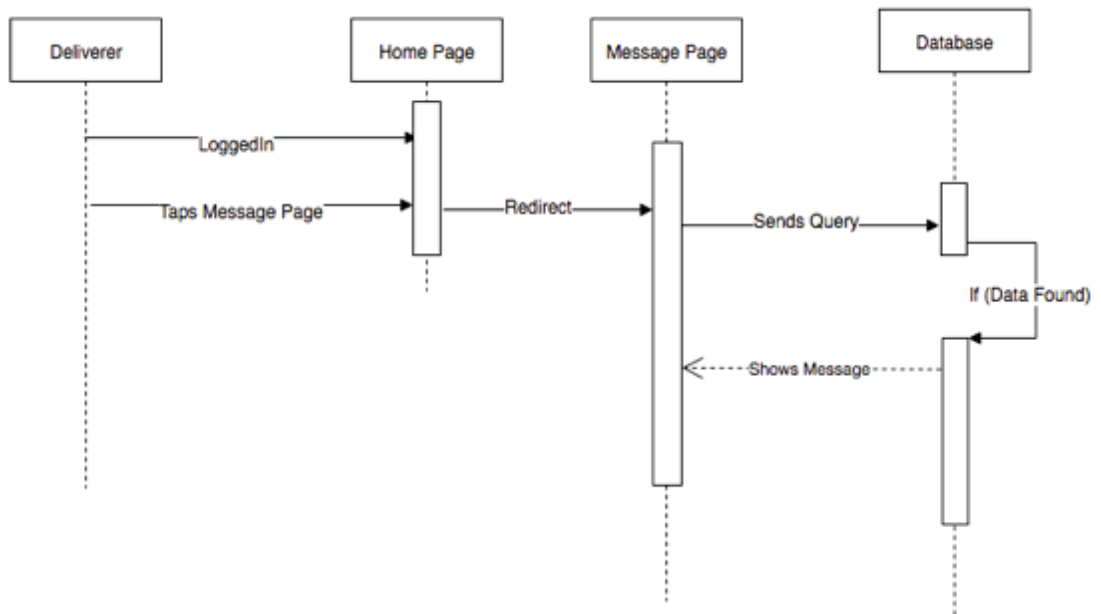


Figure 16: View Supervisor's Message Sequence Diagram

4.4.11 Deliverer's Send Message Sequence Diagram

Sequence diagram (Figure 17) shows the processes of sending a message to the office. The deliverer after login views the home page. The deliverer can tap on the message page, and he/she can find the send icon. When the icon is tapped, a new page appears which is “post message page” or “send message page”. The deliverer can write a message or comments and tap on the send button. After sending the message successfully, it returns to the confirmation message.

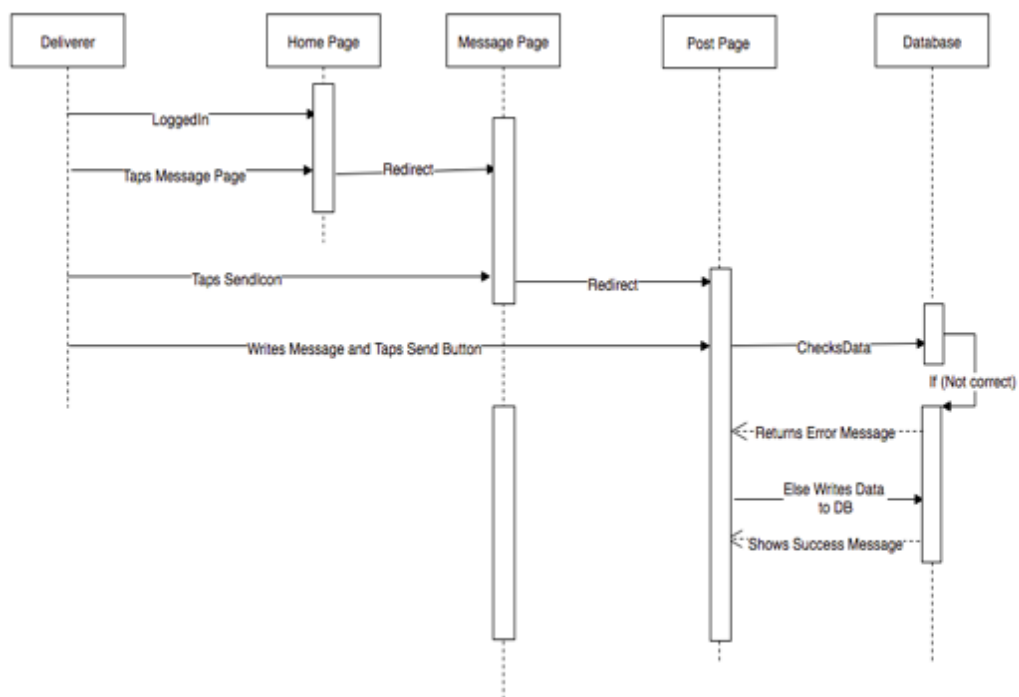


Figure 17: Deliverer's Send Message Sequence Diagram

4.4.12 View Deliverer's Account Sequence Diagram

Sequence diagram (Figure 18) shows the steps to view the deliverer's account page. The deliverer logs in with login details and the application shows the home page. Tapping Account Page shows the deliverer's email, phone number, and address. The page also contains the update profile button to update the profile and changing the password buttons for changing the old password.

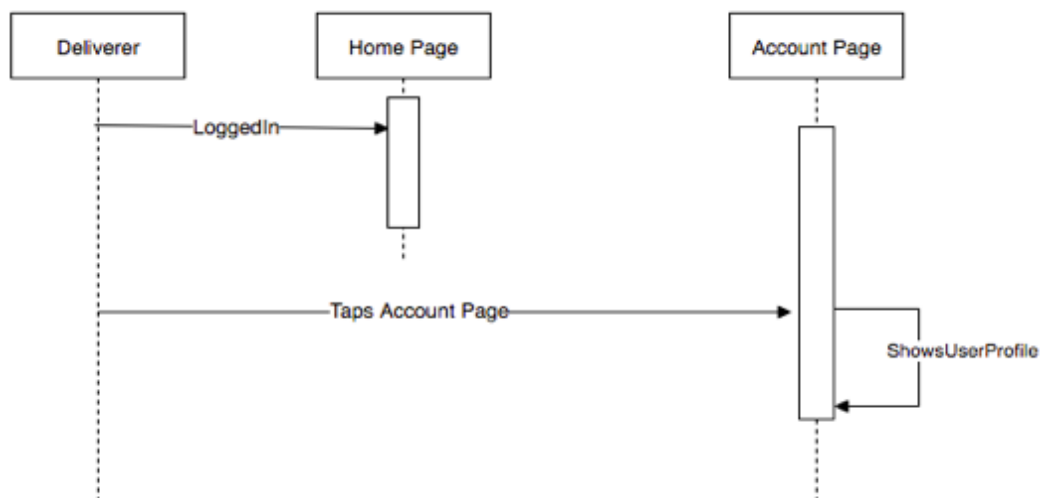


Figure 18: View Deliverer's Account Sequence Diagram

4.4.13 Deliverer's Update Profile Sequence Diagram

This Update Profile sequence diagram shows the steps for updating the deliverer's basic details (email, phone number, and address). After logging in, the application shows home to the deliverer, and he/she can tap the account page. On the account page, the deliverer can edit any of the email, phone number or address or all of them and tap update profile button. Tapping the update profile button sends the query to validation and checks if the input data is correct. If the input data is not correct, the application returns an error message, otherwise the newly changed data is saved in the database and returns the success message. (Figure 19)

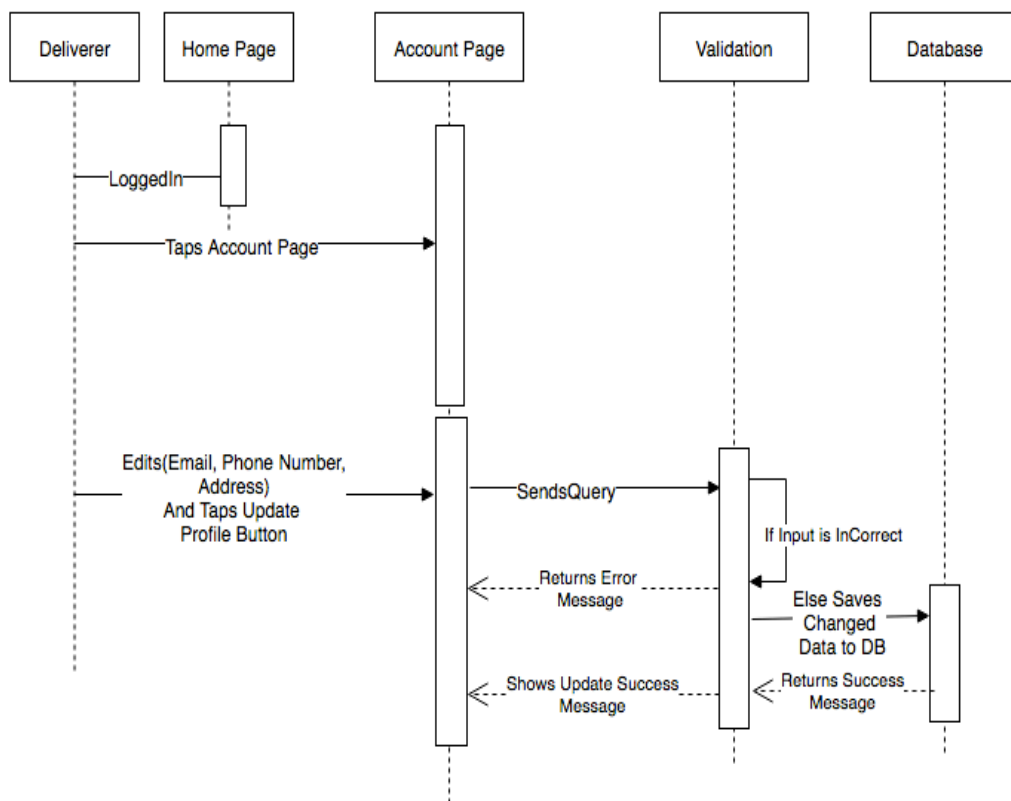


Figure 19: Deliverer's Update Profile Sequence Diagram

4.4.14 Deliverer's Change Password Sequence Diagram

This sequence diagram (Figure 20) shows the steps required for the deliverer to change the password. After login, the application shows the home page to the deliverer. The deliverer can tap account page, tap change password button and enter the old password at first, new password at the second and confirm the new password. Tapping on the change password button in the change password page sends a query to the validation which checks if the given credentials are correct. If the credentials are not correct, the application returns the error message, otherwise the new password is replaced to the old one and after successfully changing the password and the application redirects the change password page to the login page.

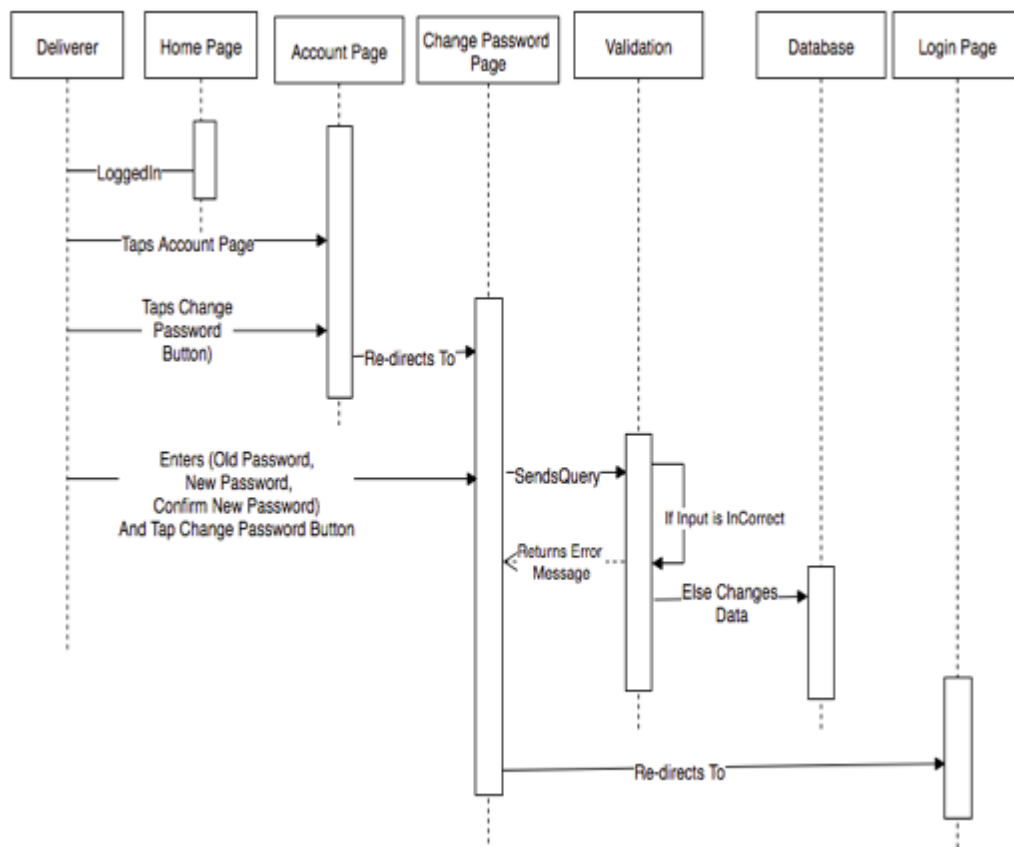


Figure 20: Deliverer's Change Password Sequence Diagram

4.5 Component Diagram

Figure 21 shows the working mechanism of each component on the applications installed on the devices. There are two applications, and each device needs an internet connection to communicate with the service. The desktop application is accessible to the office supervisors' only and the mobile application is only accessible to the deliverers. If the internet is connected, the desktop application communicates with the mobile application with the help of PHP APIs. PHP APIs are also used to connect to the database on the server side.

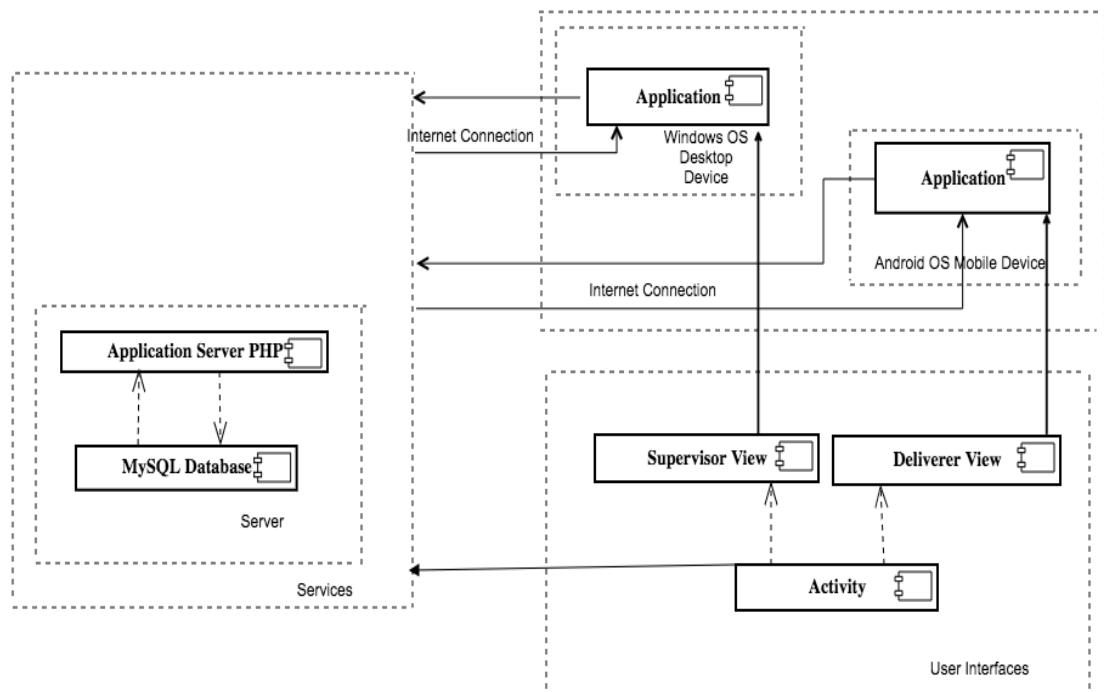


Figure 21: Component Diagram

5 DATABASE

This section is about design and configuration of the applications' database where applications are running by using a remote online database server.

5.1 Design of the Database:

The MySQL database is used to design the database. The database name is e1200655_KilometerManagementApp and is located inside VAMK's remote server and the URL address for the website is (www.mysql.cc.puv.fi).

ER Diagram is shown below:

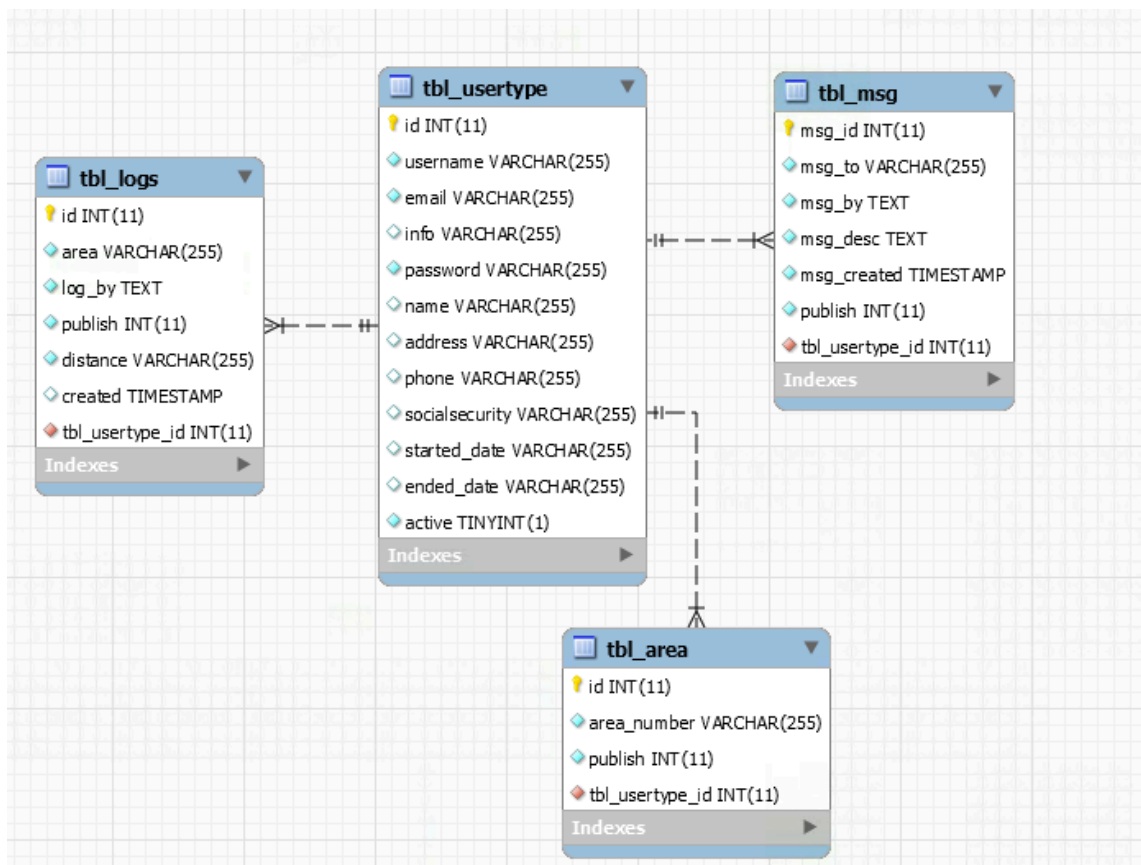


Figure 22: ER Diagram

The ER diagram describes the relationship between the different tables in the database. Figure 22 shows that the application database contains four database tables

which are tbl_usertype user details, tbl_logs for kilometer details, tbl_area for area details and tbl_msg for message details.

The user type table has one to many relationships with the table area because a deliverer has minimum one area, and he/she has one too many kilometer details in each area. Therefore, the usertype has one to many relationships with table logs. The message table can have zero or many message details and the user table has zero to many relationships with tbl_msg.

Figure 23 shows the database tables of the application.

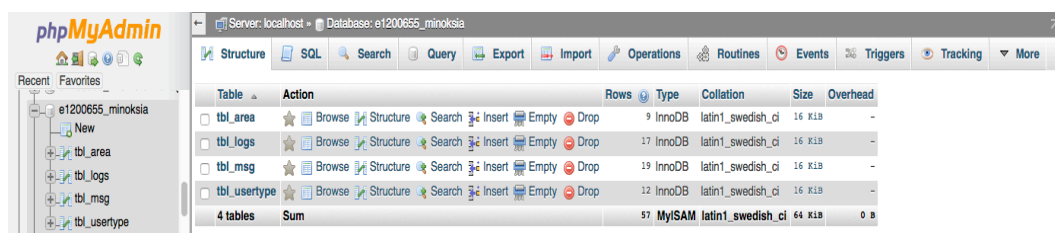


Table	Action	Rows	Type	Collation	Size	Overhead
tbl_area	Browse Structure Search Insert Empty Drop	9	InnoDB	latin1_swedish_ci	16 K1B	-
tbl_logs	Browse Structure Search Insert Empty Drop	17	InnoDB	latin1_swedish_ci	16 K1B	-
tbl_msg	Browse Structure Search Insert Empty Drop	19	InnoDB	latin1_swedish_ci	16 K1B	-
tbl_usertype	Browse Structure Search Insert Empty Drop	12	InnoDB	latin1_swedish_ci	16 K1B	-
4 tables	Sum	57	MyISAM	latin1_swedish_ci	64 K1B	0 B

Figure 23: Database Tables

The PHP APIs are stored inside public_html directory of the server. The PHP files are used to fetch and retrieve data, to write, update, delete to the application to database and vice-versa. The data is exchanged with the help of POST method, and the output value is in the JSON format. The location of the PHP files can be found in (<http://www.cc.puv.fi/~e1200655/minoksia>).

6 GRAPHICAL USER INTERFACE DESIGN

This section provides a description of the user interface designed for the applications. The graphical user interface design is divided into two parts.

6.1 Supervisor's Graphical User Interface

The user interfaces for the desktop application HTML and CSS languages. The HTML scripts are combined with PHP codes and the files are located inside the application's www directory.

6.1.1 Login Page

The login page allows the office supervisor to log in to a home page/dashboard using username and password. The login page contains two input text fields, username and password and one button for login in. The main account or supervisor account has been created while developing the application. So, the main user can use login credentials to login, and he/she can create new users either a new supervisor's account or a new deliverer's account. The login credentials should be entered. The login page shows an error message in case of wrong input or attempts to login without entering the credentials. Figure 24 shows the login page for the supervisors in the office.

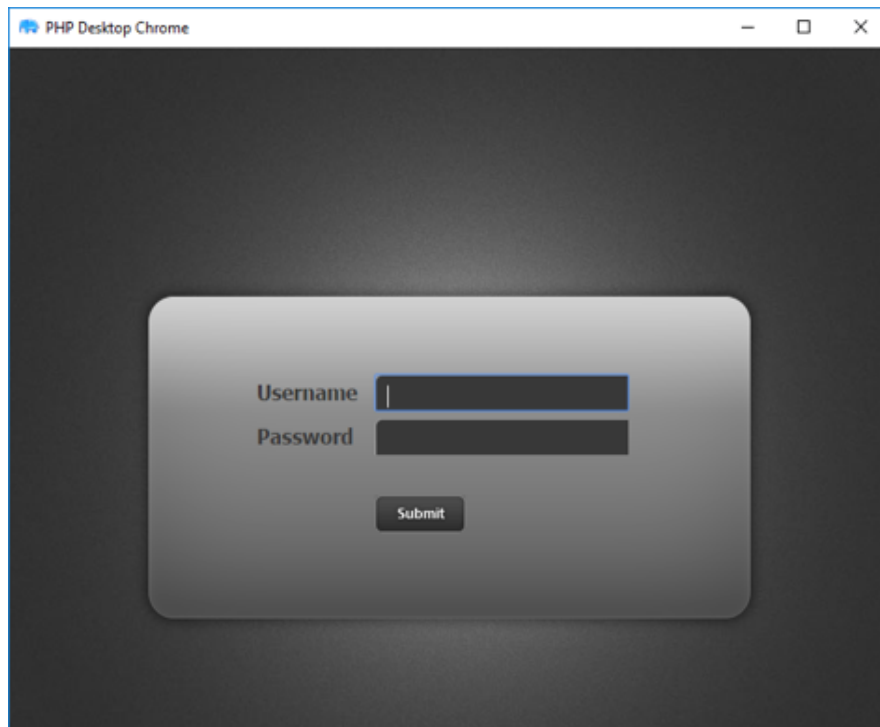


Figure 24: Supervisor's Login Page

6.1.2 Home Page

This is the home page or the main page of the application after a successful login from where the office supervisor can go to other activity pages. On the header section, the page contains a welcome text, the name of the supervisor and a clickable logout option. On the left side of the home page, it contains different clickable options through which the application allows the supervisor to operate different activities. After clicking one of these options, an output can be seen on the main content on the middle of the same page. The dashboard contains the latest kilometer updates table providing the information about id, area number, total kilometer, and send by date and time of sent kilometer details. The area page is to view the list of areas, add a new area page to add a new area, the message page to view a list of messages, new message to send a new message or information to the deliverer; the search message page is to search for messages send to the deliverers and vice-versa. The search kilometer details page is for searching the kilometer details of the deliverer. Figure 25 shows the home page:

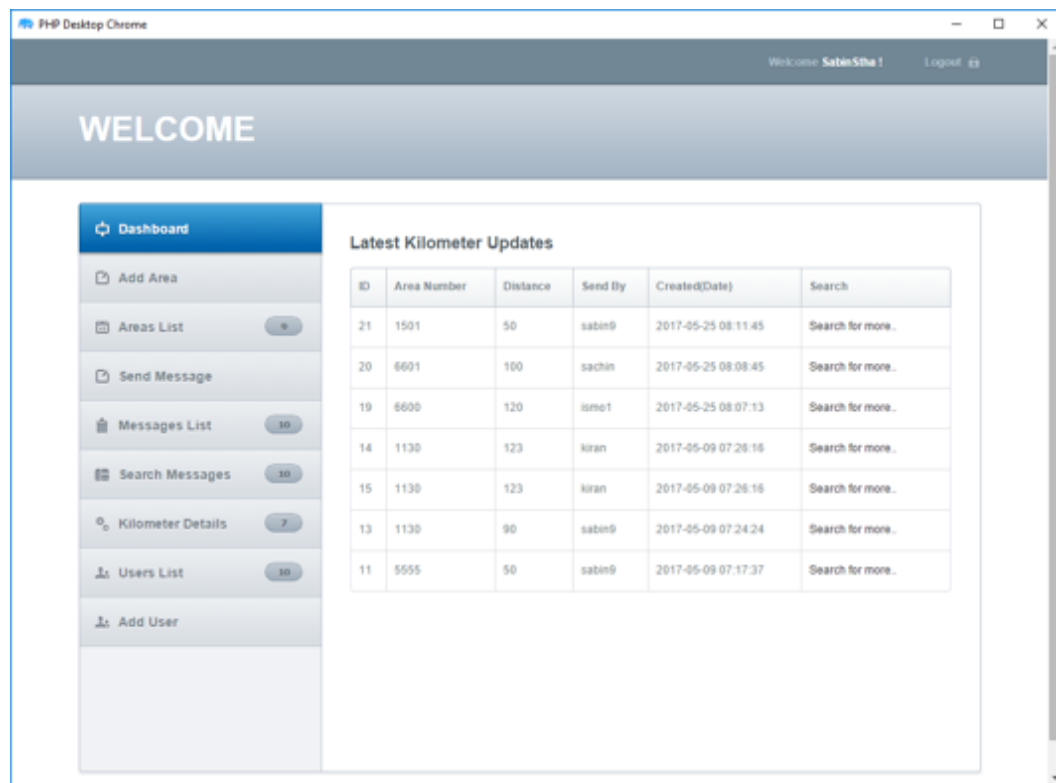


Figure 25: Supervisor's Home Page

6.1.3 New Area Page

This page allows the supervisor to add a new area. When new area option on the home page is clicked, it takes the supervisor to another page where it contains one text field to input the area number, two radio buttons 'YES' and 'NO' and one submit button. Clicking 'YES' and submit makes the respective area number visible as '1' on the area list page while clicking 'NO' and submit makes the area number to be seen as a '0' under the publish column on the area list page. So, those area numbers that have '1' can be seen on the home page of the mobile application where the deliverer can select his/her area number/s. Figure 26 shows the new area page:

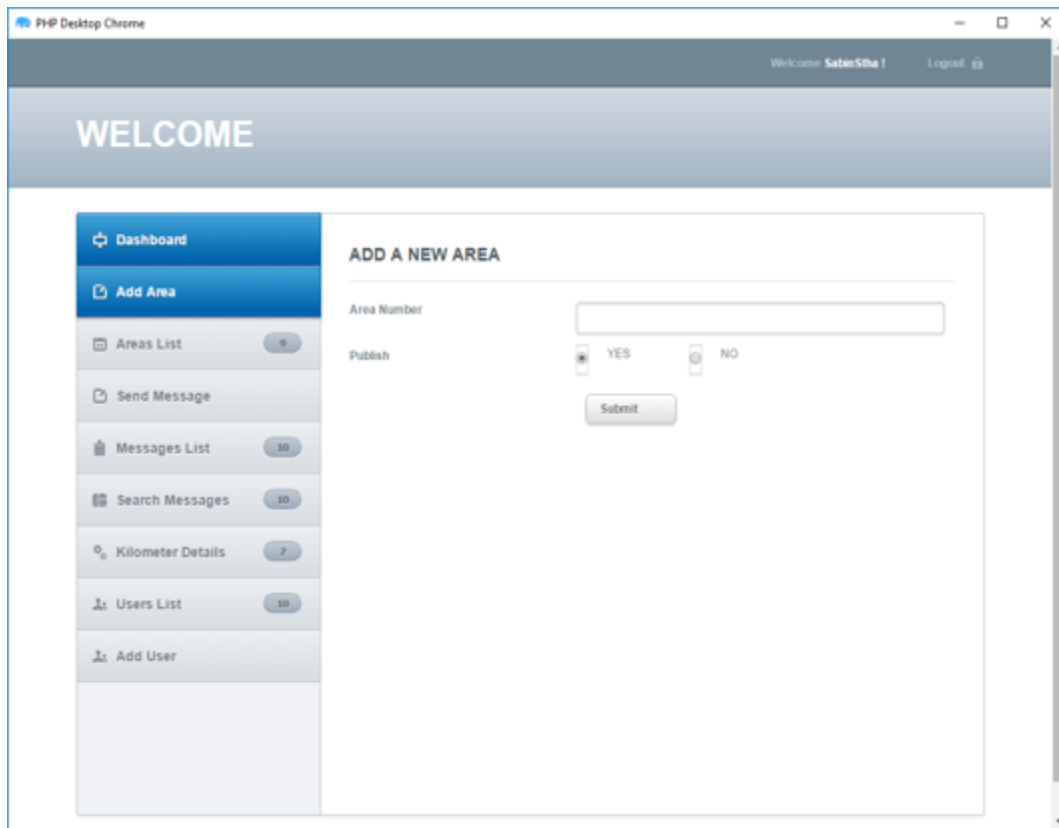


Figure 26: Add New Area Page

6.1.4 Area List Page

This page shows a table with the list of areas and each area is denoted by 4 or 5 digits of numbers called area number. This page provides the information of all the areas available. These are the list of areas which appear on the home page of mobile application of the deliverer. When the office supervisor clicks on the area list page option it shows the information like area id, area number, publish and action. The action column contains two clickable options where the supervisor is allowed to edit or delete the area number. Figure 27 shows the area list page:

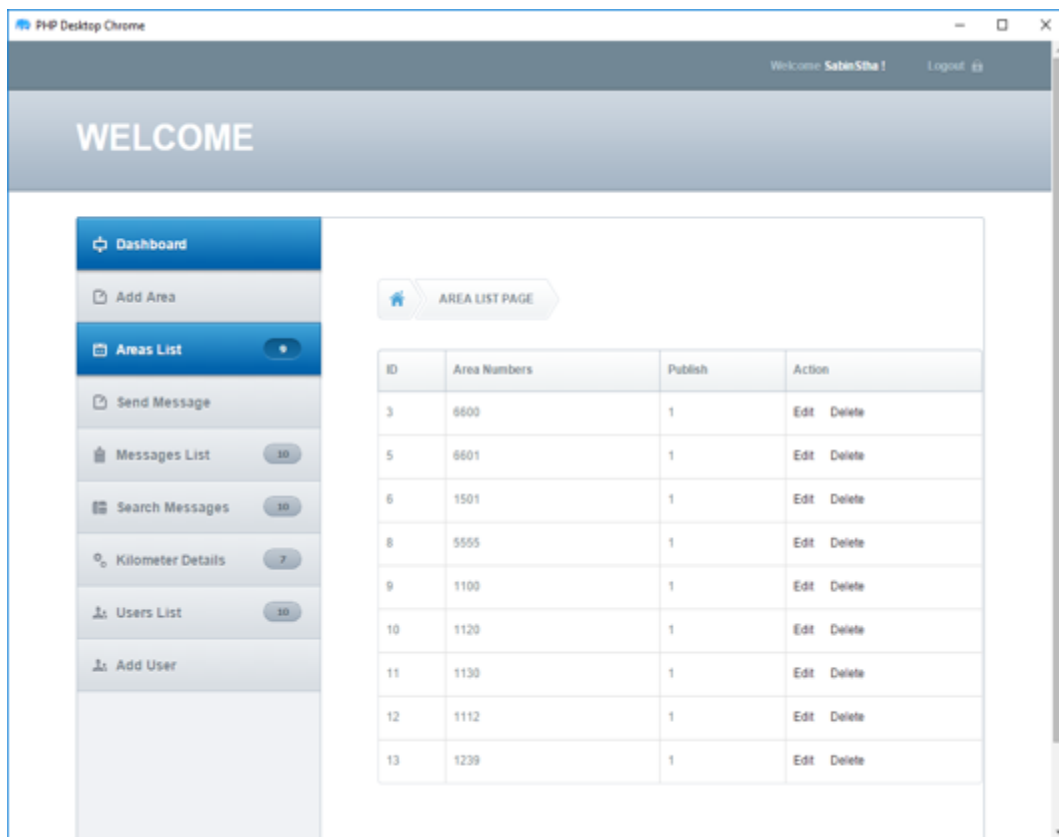


Figure 27: Area List Page

6.1.5 New Message Page

New message page allows the supervisor to write a new message and send it to the deliverer. When the supervisor clicks on the new message option on the home page, it takes the supervisor to another page. The new message page contains a drop-down list where it enables seeing the deliverers' username and selecting each one of them, radio buttons 'YES' and 'NO,' either publish or not to publish the message, a large text box to write the message and a button for submitting the message. Figure 28 shows the new message page.

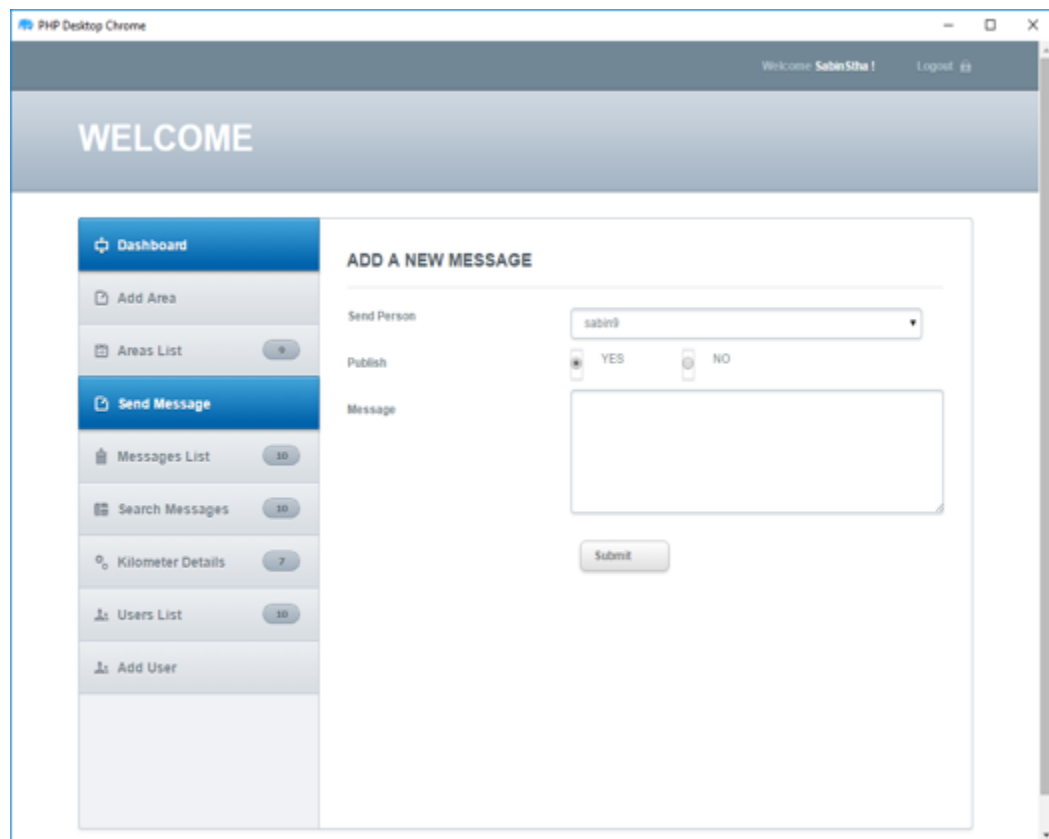


Figure 28: Add New Message Page

6.1.6 Message List Page

This page shows a table with a list of messages which are sent by the supervisor to the deliverers and vice-versa. This page appears when the supervisor clicks on the message list option on the home page. The table provides information like a message id; message send by, message send to, the message or the information's, message created date and time, publish and action. Action column contains two options edit and delete. The edit option is used to update the message list and delete option to delete the message details. Figure 29 represents the message list page.

ID	Send By	Send To	Message	Created	Publish	Action
1	sabin9	admin	Ok.	2017-05-25 11:11:54	0	Edit Delete
2	sachin	admin	This is sachin..	2017-05-25 11:09:03	0	Edit Delete
3	ismo1	admin	I am ismo..	2017-05-25 11:07:50	0	Edit Delete
4	sabin9	admin	Send more money	2017-05-09 17:34:02	0	Edit Delete
5	SabinStha	sabin9	hello testing final..	2017-05-09 15:00:00	1	Edit Delete
6	SabinStha	sabin9	Hi this is a test!!	2017-05-09 10:00:00	1	Edit Delete
7	SabinStha	sachin	hi	2017-05-09 10:00:00	1	Edit Delete

Figure 29: Message List Page

6.1.7 Search Message Page

Search Message page allows the supervisors to search messages according to the username either it is the deliverer or the supervisor. When the search message option is clicked on the home page, it takes the user to another page where there is a text field to input user's name and a clickable button to submit the request. The output is in the tabular form which has message id, whom the message has been sent to, actual message, the message sending person username and the message created date and time. Figure 30 represents the search message page.

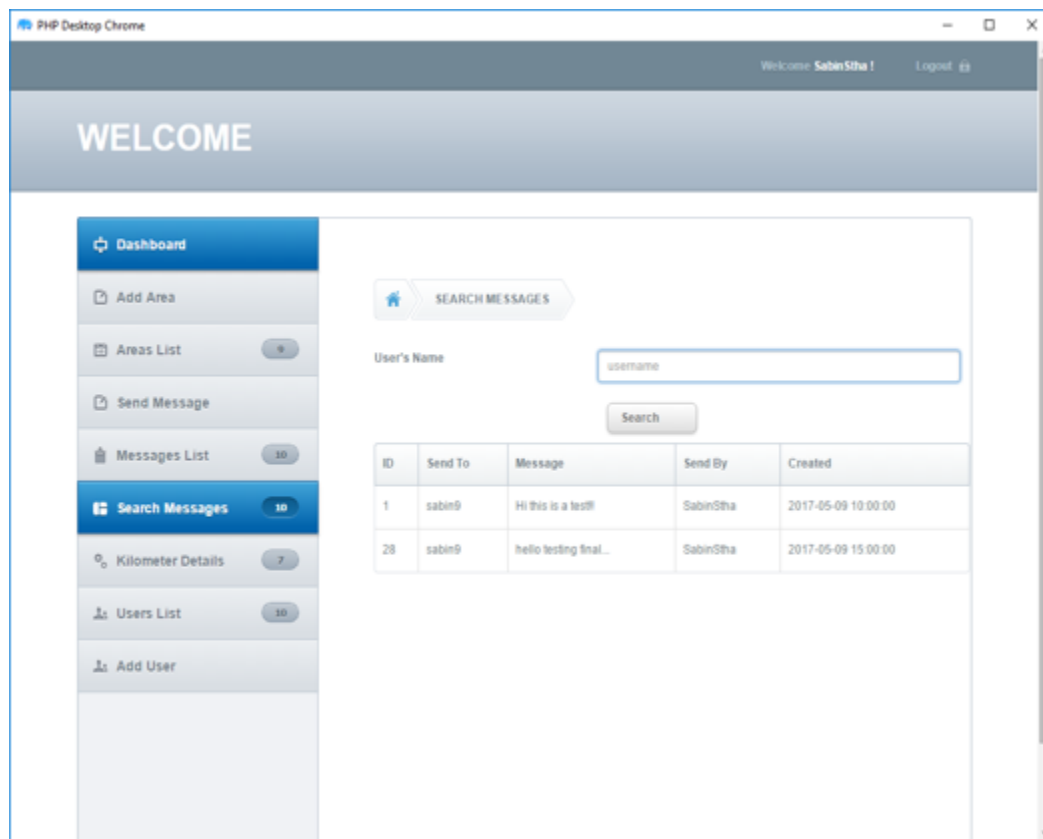


Figure 30: Search Message Page

6.1.8 Search Kilometer Page

Search Kilometer page allows the supervisor to search the kilometer details of a particular deliverer. When the supervisor clicks on the search kilometer option on the home page, it takes to search kilometer details page, and it contains a text field to input the username, two drop-down date selection options and a button to submit the request. The output is shown as a table with the available list of deliverer's kilometer details. The information includes an id, kilometer of every delivery, the deliverer's username, publish and date and time. The important element of this search is to see the total sum of kilometers which can be seen just above the table. Figure 31 represents the search kilometer page.

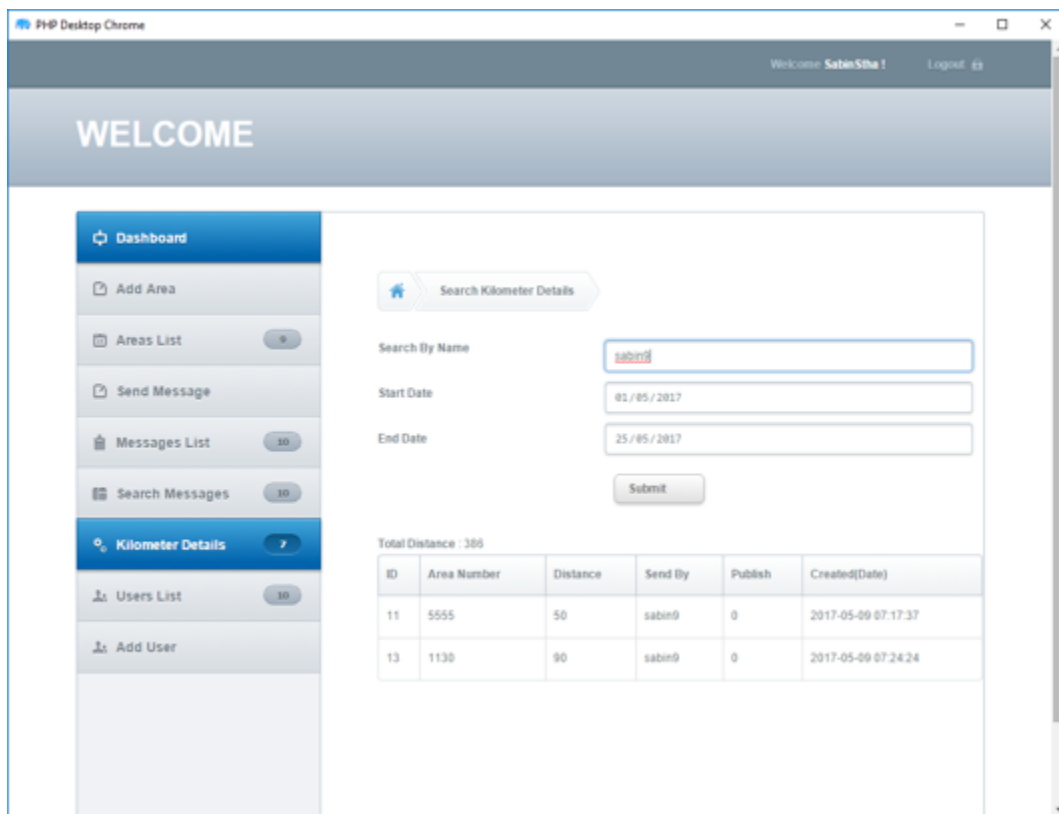


Figure 31: Search Kilometer Page

6.1.9 User List Page

User List page contains a table with a list of deliverers and supervisors working under the Pasi-Jakelut Oy. When a supervisor clicks on the user list option on the home page, it takes to the user list page. This page shows the user's basic information like user id, username, email, name, phone, user's role (deliverer or supervisor) and action. The action column has three clickable options, and they are: edit to update any changes, delete to delete the user's information and more option to view more details about the respective deliverer or supervisor. Figure 32 represents the user list page.

The screenshot shows a web application interface. At the top, there is a 'WELCOME' header. Below it, a sidebar contains several menu items: 'Dashboard', 'Add Area', 'Areas List' (with a count of 9), 'Send Message', 'Messages List' (with a count of 10), 'Search Messages' (with a count of 10), 'Kilometer Details' (with a count of 7), 'Users List' (with a count of 10, highlighted in blue), and 'Add User'. The main content area is titled 'USER LIST PAGE' and contains a table with the following data:

ID	User Name	Email	Name	Phone	User Roles	Action
1	Suman	admin@mail.com	suman kumar	9843631023	Supervisor	Edit Delete More
3	sabin9	minoksia@yahoo.com	Sabin Shrestha	+355-449890971	Delivener	Edit Delete More
4	SabinStha	raj123@yahoo.com	Sabin Raj	+355-449890970	Supervisor	Edit Delete More
5	sachin	sachin@yahoo.com	Sachin Shrestha	+355-448789870	Delivener	Edit Delete More
6	kiran	kiran@gmail.com	Kiran Maharjan	+355-449880970	Delivener	Edit Delete More
7	hari	hari@yahoo.com	Hari Salovara	+355-449890777	Supervisor	Edit Delete More
8	ppya	pp@yahoo.com	Pragya	+355-449880970	Delivener	Edit Delete More
9	john	john12@gmail.com	David John	+355-449890989	Delivener	Edit Delete More
10	ismo1	ismo@yahoo.com	Isma Lammi	+355-449890945	Delivener	Edit Delete More
11	ira	ira@yahoo.com	Ira Makki	+355-448967799	Delivener	Edit Delete More

Figure 32: User List Page

6.1.10 Add New User Page

Add New User page allows the supervisor to add a new supervisor or a new deliverer. When the supervisor clicks on the new user option on the home page, it takes him/her to the new user page. The page contains a form where there are nine text fields to add the name, address, phone, email, social-security, username, password, job starting date, job ending date, radio buttons to assign the user's role and a button to submit all the details. Figure 33 represents the new user page:

The screenshot shows a web browser window titled 'PHP Desktop Chrome'. The page has a dark header with 'WELCOME' and a user name 'Sabeetha!'. A sidebar on the left contains a list of menu items: Dashboard, Add Area, Areas List (5), Send Message, Messages List (10), Search Messages (10), Kilometer Details (7), Users List (10), and Add User. The main content area is titled 'ADD A NEW USER' and contains a form with the following fields: Name (placeholder: name), Address (placeholder: address), Phone (placeholder: +358-XXXXXXX), Email (placeholder: someone@example.com), Social Security (placeholder: 45678), Enter Username (placeholder: username), Password, Job Started (placeholder: dd/mm/yyyy), Job Ended (placeholder: dd/mm/yyyy), and Type. There are also radio buttons for 'Supervisor' and 'Deliverer' and a 'Submit' button.

Figure 33: Add New User Page

6.2 Deliverer's Graphical User Interface

The deliverers use the mobile application. For the mobile application, the ionic framework has its HTML5 input types and CSS components which give beautiful looking user interfaces. The HTML5 and CSS component codes are inside .html file. The html file represents the user interfaces, located inside "Pages" directory of the application.

6.2.1 Splash Screen

The splash screen is the small visualization of about 3 seconds for the users which provides a welcoming text or animation. When the deliverer taps the application icon, the splash screen starts, and the application takes to the login page. The splash

screen is designed using PhotoScape X where Ionic Framework has specific dimensions for the splash screen. Splash screen's minimum dimension should be 2208x2208 so that it can function well. Figure 34 represents the splash screen.

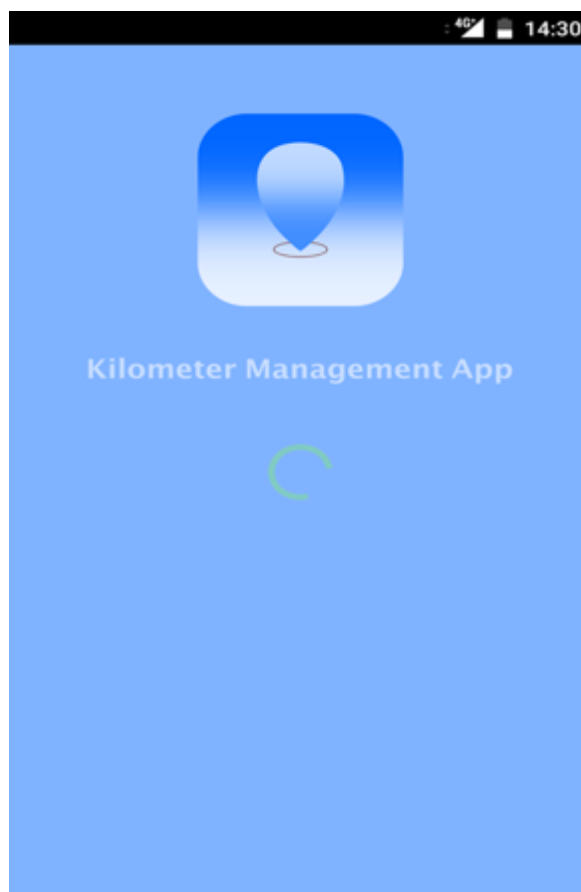


Figure 34: Splash Screen

6.2.2 Login Page

The login page allows the deliverer's login using his/her login credentials. The deliverer should have asked for the username and password at the office. After checking, or if necessary adding, deliverer's information, new username and password are provided so that he/she can use the mobile application. The login page contains two text field to input username and password and a button to submit the login

credentials. There should be a minimum of six characters for the password otherwise the application does not allow to login. If the login fails, the deliverer gets an error message. Figure 35 represents the login page.

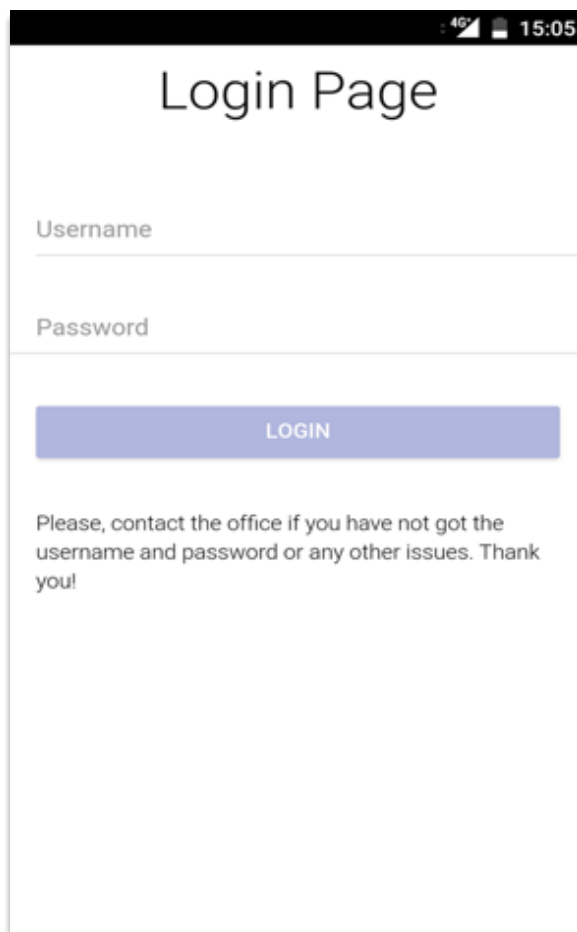
The image shows a mobile application interface for a login page. At the top, there is a black status bar with white icons for 4G signal, battery, and the time 15:05. Below this, the title "Login Page" is centered in a large, black, sans-serif font. Underneath the title, there are two input fields: "Username" and "Password", each with a thin grey border and a light grey placeholder text. Below the password field is a prominent blue button with the word "LOGIN" in white, uppercase letters. At the bottom of the page, there is a block of text in a smaller, black font that reads: "Please, contact the office if you have not got the username and password or any other issues. Thank you!".

Figure 35: Deliverer's Login Page

6.2.3 Home Page

The successful logged in of the deliverer from the login page allows him/her to use the home page. The home page has one text with the name of the deliverer, two drop-down lists, one input label and one button. The application allows deliverer to select advertisement delivery date, to select the delivery area/s, input label to add kilometer and send button to send the kilometer details. The list of areas on the

home page is added by the supervisor from the office. Figure 36 represents the home page:

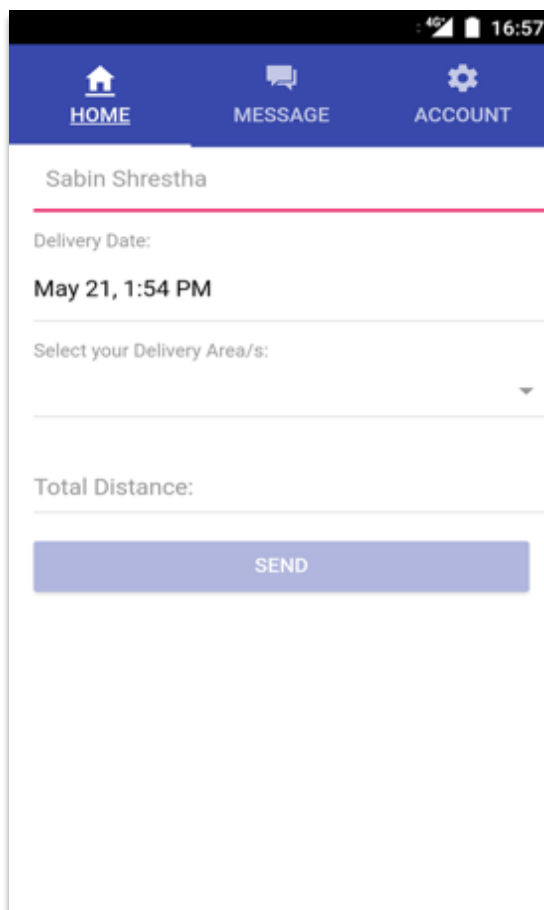


Figure 36: Deliverer's Home Page

6.2.4 Message Page

The successful logged in of the deliverer from the login page allows him/her to use the home page. The home page has one text with the name of the deliverer, two drop-down lists, one input label and one button. The application allows deliverer to select advertisement delivery date, to select the delivery area/s, input label to add kilometer and send button to send the kilometer details. The list of areas on the home page is added by the supervisor from the office. Figure 37 represents the home page:

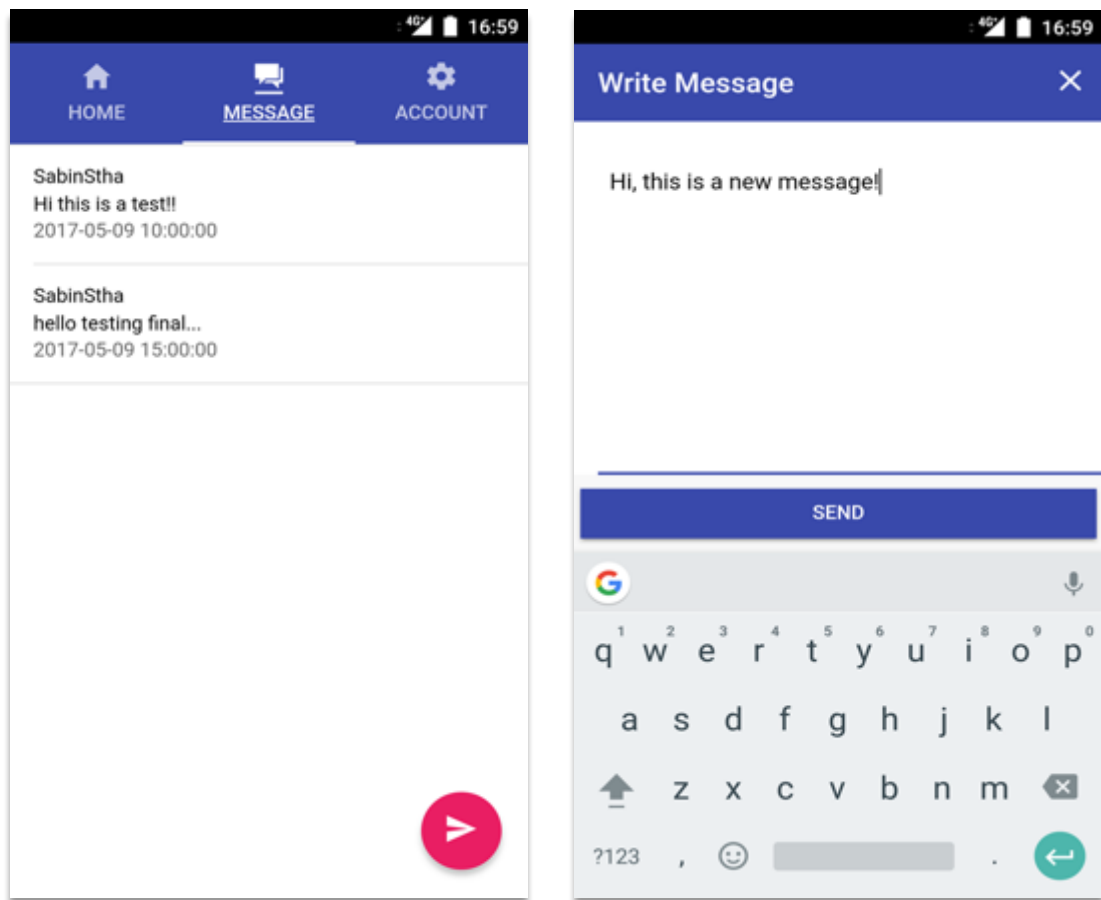


Figure 37: Deliverer’s Message Page and Write Message Page respectively.

6.2.5 Account Page

The account page allows the deliverers to view his/her basic information, username, email, address and phone. This page appears when a deliverer taps the account page. The page contains a text field to show the username, three changeable text fields, and three buttons. If the deliverer wants to change his/her email, address or phone number, the deliverer can click on one of these, edit it and press the update profile button. The update success message appears if the update is successful. After pressing the change password button, it allows the deliverer to change the password and the deliverer can logout by pressing the logout button.

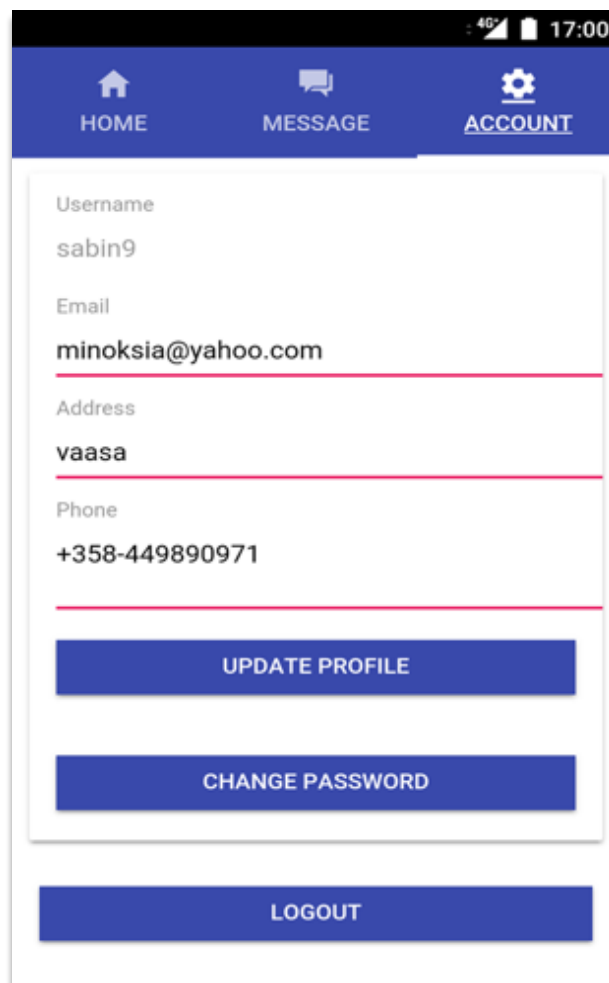


Figure 38: Deliverer's Account Page

6.2.6 Change Password Page

The change password button on the account page allows the deliverer to change the old password and make his/her own password. The change password page contains three text fields where the deliverer needs to input the old password, and the new password twice for the confirmation. After pressing the change password button on the same page, a confirmation message appears, and it automatically logs out. The application logs out so that the deliverer can again login with the new password. Figure 39 shows the change password page.

The image shows a mobile application interface for changing a password. At the top, there is a blue header bar with a white back arrow on the left and the text 'Change Password' in white. Below the header, the page has a white background. There are three text input fields, each with a light gray border and a light gray label above it: 'Old Password', 'New Password', and 'Confirm Password'. Below these fields is a blue button with the text 'CHANGE PASSWORD' in white, all-caps. The top of the screen shows a black status bar with white icons for 4G signal, battery, and the time 17:00.

Figure 39: Change Password Page

7 IMPLEMENTATION

This section provides a description of the implementation of the graphical user interfaces which is built for the application. It shows how the application codes are implemented to achieve the objectives of the project. There are two parts: implementation of mobile application and implementation of the desktop application. TypeScript is used for the backend of the mobile application where PHP APIs and MySQL as a database handles the server. Moreover, PHP codes are used to handle the backend of the desktop application. The following code snippets show the implementation of the applications.

7.1 Supervisor Login

The supervisor input values to variables are assigned if login form is submitted. The values are checked in the database to see whether they exist or not. If the supervisor's login credentials are matched with the values in the database, the session is created, and the index page or home page is displayed.

```

if(isset($_POST['login'])){
    $username = $_POST['username'];
    $password = md5($_POST['password']);
    if(empty($username) || empty($password)){
        $msg = "Form Field Empty !!!";
    } else {
        $query = mysql_query("select * from tbl_us-
        ertype where username='$username' and pass-
        word='$password' and active=1");
        $query2 = mysql_num_rows($que-
        ryif($query2>0){$_SESSION['loggedIn'] =
        $username;
            echo "<script>>window.location='in-
            dex.php'</script>";
        } else {
            $msg = "User Does Not Exist !!!";
        }
    }
}

```

Code Snippet 1: Login Page

7.2 Supervisor's Home

After a successful login, the home page can be seen. The header function is used to handle the raw HTTP header. Before sending any actual data to a client, the header() function must be called. The isset checks the variable is set and not null.

```
if(!isset($_SESSION['loggedIn'])){
    header('Location: login.php');
```

Code Snippet 2: Header function

The home page has several pages. The switch statement has been assigned to select each of multiple pages. It compares the value of the variables and compares with each case in the structure. If the case is matched with code, the selected page is executed. The break is used to avoid going into the next case.

```
if(isset($_GET['page'])){
    switch($_GET['page']){
        case "area":
            include('includes/area.php');
            break;
        case "add_area":
            include('includes/add_area.php');
            break;
        case "edit_area":
            include('includes/edit_area.php');
            break;
        case "delete_area":
            include('includes/del_area.php');
            break;
        case "mssg":
            include('includes/mssg.php');
            break;
        case "add_msg":
            include('includes/add_msg.php');
            break;
```

```

    case "edit_msg":
    include('includes/edit_msg.php');
    break;
    case "delete_msg":
    include('includes/del_msg.php');
    break;
    case "search_msg":
    include('includes/search_msg.php');
    break;
    case "user":
    include('includes/user.php');
    break;
    case "show_logs":
    include('includes/show_logs.php');
    break;
    case "add_user":
    include('includes/add_user.php');
    break;
    case "edit_user":
    include('includes/edit_user.php');
    break;
    case "view_user":
    include('includes/show_user.php');
    break;
    case "delete_user":
    include('includes/del_user.php');
    break;}
}

```

Code Snippet 3: Sidebar Menu

The Query below is used to display the latest kilometer details sent by the deliverer.

```

$query = mysql_query("select * from tbl_logs");
    $index = 0;
while($result = mysql_fetch_array($query)){

```

```
$index++;
```

Code Snippet 4: Query method to get the kilometer details

7.3 Add User

The PHP file is used to execute all the form data that comes from the clients for processing. The isset checks the variable are set and not null, the HTTP POST method is used to fetch the form data.

```
if(isset($_POST['add_user'])){
    $username = $_POST['username'];
    $name = $_POST['name'];
    $address = $_POST['address'];
    $phone = $_POST['phone'];
    $email = $_POST['emailval'];
    $socialsecurity = $_POST['socialsecurity'];
    $password = md5($_POST['password']);
    $started_date = $_POST['started_date'];
    $ended_date = $_POST['ended_date'];
    $active = $_POST['active'];
```

Code Snippet 5: Variable Declaration

The following code checks if the minimum user inputs are entered or not.

```
if(empty($username) || empty($password) ||
empty($name) || empty($email)){
    $msg = "Username, Passwords, Name and Email
are Required!!!";
```

Code Snippet 6: Check Inputs

When the user inputs are entered on the add user page, clicking submit button sends those form inputs to the database with the following code.

```
$query=mysql_query("insert into tbl_us-
ertype(username, password, active, name, ad-
dress, phone, socialsecurity, started_date,
ended_date, email) val-
ues('$username', '$password', '$ac-
```

```

tive', '$name', '$address', '$phone', '$so-
cialsecuri-
ty', '$started_date', '$ended_date', '$email')"
);

$query = mysql_affected_rows($connect);
if($query >=1){
    $msg = "Successfully User is Added.";
} else {
    $msg = mysql_error();
}
}

```

Code Snippet 7: Query method to add a new user

7.4 Area

When the supervisor clicks on the area list page, it shows the available list of area numbers. With the help of the following query, it retrieves the data from the tbl_area table of the database.

```

$query = mysql_query("select * from tbl_area");
while
($result = mysql_fetch_array($query)){

```

Code Snippet 8: Query method to get the area number list

The following code shows the implementation of updating area to the database when supervisor clicks on the edit link on the area list page.

```

if(isset($_POST['upd_area'])){
    $id = $_POST['id'];
    $areaname = $_POST['areaname'];
    $publish = $_POST['publish'];
    $query = mysql_query("update tbl_area
set area_number='$areaname', publish=
'$publish' where id='$id'");
if($query){
    $msg = "Successfully Area Updated.";

```

```

} else {
    $msg = mysql_error();
}
}

```

Code Snippet 9: Update/Edit area

The add area option when clicked shows a form to add an area number and publish options. The HTTP POST method takes the inputs entered on the add new area page, and with the help of the following query, the new area is inserted if there is no error. The new area can be seen on the area list page.

```

if(isset($_POST['add_area'])){
    $areanumber = $_POST['areanumber'];
    $publish = $_POST['publish'];
    if(empty($areanumber )){
        $msg = "Area Number is Required!!!";
    } else {
        $query = mysql_query("insert into tbl_area
        (area_number, publish) vaues('$are-
        anumber', '$publish')");
    }
    if($query){
        $msg = "Successfully Area Added.";
    } else {
        $msg = mysql_error();
    }
}
}

```

Code Snippet 10: Add a new Area

7.5 Messages

When the supervisor clicks on the send message page, the following codes are executed and the data are inserted into the database in table named as tbl_msg.

```

if(isset($_POST['add_msg'])){
    $msg_user = $_POST['user_by'];
    $msg_post = $_SESSION['loggedIn'];
}

```



```

    $msg_desc = $_POST['editor1'];
    $publish = $_POST['publish'];
    $msg_created = date("Y-m-d H:i:s");
$query = mysql_query("insert into
tbl_msg(msg_to,msg_by,msg_desc,msg_created, pub-
lish) values('$msg_us-
er','$msg_post','$msg_desc','$msg_created','$pub-
lish')");
if($query){
    $msg = "Successfully Message Added.";
} else {
    $msg = mysql_error();
}
}

```

Code Snippet 11: Variable Declaration and add message

When the supervisor clicks on the message page, it displays a list of messages. The implementation of the code behind this display is the following query which selects the list of messages available in the database table named tbl_msg.

```

$query = mysql_query("select * from tbl_msg");
    $index = 0;
while($result = mysql_fetch_array($query)){
    $index++;
}

```

Code Snippet 12: View Messages

The edit link on the message page uses the following codes to edit the user's messages. When all the required variables are set correctly, data are updated into the database, table named as tbl_msg. The id of the table message

```

if(isset($_POST['edit_msg'])){
    $id = $_POST['id'];
    $msg_user = $_POST['user_by'];
    $msg_post = $_SESSION['loggedIn'];
    $publish = $_POST['publish'];
    $msg_created = date("Y-m-d H:i:s");
}

```

```

$query = mysql_query("update tbl_msg set msg_to =
'$msg_user', msg_created = '$msg_created', pub-
lish = '$publish' where msg_id = '$id'");
if($query){
    $msg = "Successfully Message Up-
dated.";
} else {
    $msg = mysql_error();
}
}

```

Code Snippet 13: Variable declaration and update/edit message

7.6 Kilometer Details

The below code shows the form data or variables that are set on the PHP file via HTTP POST method. When the supervisor adds the necessary parameters to the form and presses submit, the query below is executed and show the data.

```

if(isset($_POST['search_log'])){
    $search_by_name = $_POST['search_by_name'];
    $started_date = $_POST['started_date'];
    $ended_date = $_POST['ended_date'];
    $searchque = "SELECT * from tbl_logs where
created between '$started_date' and
'$ended_date' and log_by LIKE
'%" . $search_by_name . "%'";
    $querySearch = mysql_query($searchque);
    $usernum = mysql_num_rows($querySearch);
}

```

Code Snippet 14: Search Kilometer Details

7.7 Deliverer's Login Page

The login page contains the declaration of user data and imports. The Storage and Local Storage help to store key/value pairs and JSON objects. The variables are initialized in the constructor method and also injecting providers like navigation controller for navigating the components or the pages and Http make the request and returns JSON response. The login form or form is basically to get the user info.

```

export class LoginPage {
  loginForm;
  auth;
  http:Http;
  username: string;
  password: string;
  storage = new Storage(LocalStorage);
  constructor(public nav:NavController,form:FormBuilder, http:Http) {
    this.http = http;
    this.loginForm = form.group({
      username: [ "",Validators.required]],
      password:[ "",Validators.required]
    });
  }
}

```

Code Snippet 15: Initializing the Login Variables

The login method consists of a URL link to exchange data. URL includes the PHP codes. It communicates with the database and http.post method writes the user's login values. The subscribe method includes success and return functions respectively and handles multiple values. The success function collects the data in the form of JSON Array from the backend and pushed it to next page or tab.

```

login() {
let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp/mobile_login.php';
  this.http.post(url,this.loginForm.value).subscribe(data => {
    let respData =JSON.parse(data._body);
if(respData.logged){
  this.nav.push(TabsPage);
  console.log(respData,"respData")
  this.storage.set('name',respData.name);
} else {
  let errorMessage = "Enter Correct Credentials";

```

```

        let alert = this.util.doAlert("Error", errorMessage, "Ok");
        this.nav.present(alert);
    }
});

```

Code Snippet 16: Login Background Task and Post Execute

7.8 Deliverer's Home Page

After the deliverer's logs in, the application shows the home page where it contains the deliverer's name, date list, area list and input field. In the constructor, different providers are injected and methods are initialized. The getItem method returns, the username and name.

```

constructor(public navController: NavController, public util: UtilProvider, form: FormBuilder,
private params: NavParams, public events: Events,
public viewCtrl: ViewController, http: Http) {
    this.http = http;
    this.username = localStorage.getItem('userName');
    this.name = localStorage.getItem('name');
    console.log(localStorage.getItem('userName'), "username");
}

```

Code Snippet 17: Checks the user details

When the username is checked on the local storage, the URL link exchanges the data with the database. The success function in the subscribe method receives the data from the backend. The deliverer can select the date and the delivery area from the list on the home page and input the total kilometer.

```

let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp /mobile_area.php';
this.http.post(url, {}).subscribe(data => {
    let respData = JSON.parse(data._body);
    for(let data of respData) {
        this.items.push({ value: data.area_number,
            text: data.area_number, checked: false });
    }
});

```

```

        }
      },error => {
        let errorMessage = "error";
      });
    this.createForm = form.group({
      myDate: ["", Validators.required],
      area: ["", Validators.required],
      distance: ["", Validators.required],
      username:[this.username],
      name:[this.name],
    });
  }
}

```

Code Snippet 18: Add Kilometre Details

This section provides the description of sending kilometer details. When the deliverer's input details are correct, and he/she taps send button, the post method pushes those details to the database through the URL.

```

sendLogs(){
let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp/addlogs.php';
console.log('form log', this.createForm.value);
  this.http.post(url,this.createForm.value).subscribe(data => {
    let respData =data._body;
if(respData == "success"){
  let successMessage = "Kilometere Details Saved Successfully";
  let alert = this.util.doAlert('Success',successMessage,"Ok");
  this.navController.present(alert);
  console.log("succcess !!");
} else {
  let alert = this.util.doAlert("Error",respData,"Ok");
}
}

```

```
});
```

Code Snippet 19: Method to send kilometer details

7.9 Deliverer's Message Page

When the message tab is tapped, the application shows the message page and the available list of messages as well as sends an icon to send the message. At first a user's data is called and checked when the message tab is pressed once. If the data is available in the local storage, http request and post method write the data on the database via URL link, which returns the data in JSON format.

```

    this.tab = this.navController.parent;
    this.http = http;
    let username = localStorage.getItem('userName');
let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp /mobile_messages.php';
    this.http.post(url, { username: username
    }).subscribe(data => {
        let respData = JSON.parse(data._body);
        console.log(respData, "respData");
        this.messages = respData;
    });
}
addMessage() {
    this.navController.push(PostPage)
}
}

```

Code Snippet 20: View Supervisor's Message

After tapping the send icon on the message page, the application redirects to send message page or post page. The user details available on the local storage are called and checked. If the user details are correct, the http post method sends the request and writes the data to the database via an URL link and returns a successful message.

```

sendPost() {
    let username = localStorage.getItem('userName');
    let obj = {content: this.postContent,
              username:username};
let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp /mobile_post.php';
    this.http.post(url,obj).subscribe(data => {
        let respData =data._body
        if(respData == "success"){
            let successMessage = "Message Send Successfully";
            let alert = this.util.doAlert('Success',successMessage,"Ok");
            this.navController.present(alert);
            this.reset();
                this.dismiss();
        }
    }, error => {
        let errorMessage = "Error";
    });
}
reset() {
    this.postContent = "";
}

```

Code Snippet 21: Send Message

7.10 Account

When the account page shows the deliverer's basic profile, it provides an option to update the profile and the change password. The `userId` is called and checked. If the data is found in the local storage, the `http post` method sends a request to the database and returns the user's information.

```

let userId = localStorage.getItem('userId');
let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp /mobile_userinfo.php';

```

```

        this.http.post(url, {userId :userId}).sub-
        scribe(data => {
            let respData =JSON.parse(data._body);
            console.log("response  data", respData);
        if(respData.response){
            this.user.username = respData.username;
            this.user.email = respData.email;
            this.user.phone = respData.phone;
            this.user.address = respData.address;
        }
    }
}

```

Code Snippet 22: View Deliverer's Information

In the account page, a deliverer taps on the update profile button after the necessary changes are made, the `userId` of the deliverer is called and checked on the local storage. When the id is matched, the http post request method sends a request to the database and returns the updated data in the JSON format and shows update success message.

```

updateProfile() {
    let url ='http://www.cc.puv.fi/~e1200655/Kilome-
    treManagementApp /mobile_updateuserinfo.php';
    let userId = localStorage.getItem('userId');
    this.http.post(url, {
        email: this.user['email'],
        phone: this.user['phone'],
        address: this.user['address'],
        userId:userId}).subscribe(data => {
        let respData =data._body;
        if(respData == "success"){
            let successMessage = "Your Profile is
            updated";
            let alert = this.util.doAlert('Suc-
            cess', successMessage, "Ok");
            this.navController.present(alert);
        }
    });
}

```



```

        console.log("success !!");
    } else {
        let alert = this.util.doAlert("Error", respData, "Ok");
    }
}

```

Code Snippet 23: Update Profile

7.11 Change Password

Tapping change password button on the account page redirects to the change password page where necessary updates are made. Changing password is successful when the `userId` is found on the local storage, the request is further pushed with the help of `http post` method, and the data is re-written on the database. As the password is changed, the application redirects to the login page with the use of navigation controller.

```

ChangePassword() {
let userId = localStorage.getItem('userId');
let { password, repass, opassword } =
    this.createForm.value;
    if (password !== repass) {
        let alert = this.util.doAlert("Error",
            "Password doesn't matched", "Ok");
        this.nav.present(alert);
    } else {
        console.log(userId, "userId")

        let obj = { userId: userId, password: password,
            oldpassword: opassword };

let url = 'http://www.cc.puv.fi/~e1200655/KilometreManagementApp /mobile_forgotpass.php';
        this.http.post(url, obj).subscribe(data => {
            let respData = data._body;
            if (respData == "success") {
                let successMessage = "Your password is updated";
            }
        });
    }
}

```

```
        let alert = this.util.doAlert('Success', successMessage, "Ok");
        this.nav.present(alert);
        localStorage.removeItem('userId');
        this.nav.push(LoginPage);
    } else {
        let errorMessage = "Couldn't Update";
        let alert = this.util.doAlert('Success', errorMessage, "Ok");
        this.nav.present(alert);
    }
});
```

Code Snippet 24: Change Password

8 TESTING

The process of testing is carried out so that it is easy to find the errors or the problems of the applications. Software Testing is the validation and certification process where the software fulfills the requirements of the users and project specifications. This project has two applications, and they are tested on two different devices. The mobile application is tested on One plus Three Android mobile phone and the Desktop application is tested in MacBook Pro Windows 10 environment. Table 3 consists of various tests performed during the development of this project.

Table 3: Testing for Mobile Application

No	Test Performed For	Test Descriptions	Test Result	Corrections and Results
1.	Application Icon	Application was installed and launched.	The application icon did not appear on the device but appeared the default icon. The customized app icon should appear.	The icon file was saved directly from the PhotoScape x to the resources folder with correct dimension i.e. 192x192. Result was ok.
2.	Slash Screen	The application started.	The splash screen turned black and redirected to login page. The Customized splash screen should appear.	The splash screen file was again saved on the resources folder with correct dimensions i.e. 2208x2208. Result was ok

No	Test Performed For	Test Descriptions	Test Results	Corrections and Results
3.	Login page	Checked text fields and buttons after installation.	The login page should contain text fields for username and password and login button.	Ok
4.	Validation on login page	Tried to log in without entering login credentials or entering wrong credentials.	The login page displayed error message.	Ok
5.	Validation on login page	Entered the correct login credentials and tried to log in.	The login redirected to homepage.	Ok
6.	Date selection on the homepage	Tried to select the date of delivery.	The DateTime picker appeared.	Ok
7.	Delivery area selection on the homepage	Tried to select the area from the area list.	The area list form appeared and it was possible to select the area.	Ok
8.	Sending kilometer details	The area was not selected and tried to click the send button.	The home page shows error. Without entering the total kilometer, it was not possible to send the details.	Ok

No	Test Performed For	Test Descriptions	Test Results	Corrections and Results
9.	Update profile on the account page	The text fields were left empty and try to click the update profile button.	The update page shows error message.	Ok
10.	Change Password	The text fields were left empty and tried to change the password. The old password was entered wrong and new passwords were entered.	The change password page shows error message.	Ok

Table 4: Testing for Desktop Application

No	Test Performed For	Test Descriptions	Test Results	Corrections and Results
1.	Login Page	The login credentials were left empty and another time entered incorrect login data.	The login page displayed error message.	Ok

2.	Login Page	The login credentials were entered correctly and clicked the login button.	The login page redirected to home page.	Ok
No	Test Performed For	Test Descriptions	Test Results	Corrections and Results
3.	Side menu on the home page	The side menu has different options to be selected. Each pages were clicked.	Each pages when clicked, redirected to their respective new page.	Ok
4.	Searching kilometer details	The kilometer option was clicked and redirected to the requested page. Username and search by date fields were left.	The application did not show the total kilometer.	Ok
5.	Searching kilometer details	Username and search by date fields were entered.	The application showed total kilometer and the kilometer details of each date sent by the deliverer.	Ok

6.	Adding new area	On the add area page, the text field was left empty and tried to add a new area.	The application did not add a new area.	Ok
7.	Adding a new user	On the add user page, tried to add a new user without entering all the required data.	The application did not create a new user.	Ok

9 SUMMARY

The project aimed to develop a mobile and a desktop application where they provide an easy and efficient way to keep the records of kilometer details for the employees of Pasi-Jakelut Oy. The mobile application enables the deliverers to send kilometer details which include the date of delivery, delivered area, and total distance or kilometers. The other features are the deliverer can view the message and send the message, update his/her profile and change the password. The desktop application allows the supervisor to add a user, update user details, check the kilometer details of each deliverer, send message to the deliverer and add area or update area for the deliverers.

The main objectives and goals have been achieved. The mobile and desktop application were tested. The mobile application was tested on Android mobile device and the desktop application was tested on MacBook Pro windows environment.

10 CONCLUSIONS

The application provided an efficient way to manage the users' work details. The process of developing the applications and their successful tests which fulfilled the necessary requirements of the user have created more learning opportunities and boosted confidence.

This project became a medium to work with Hybrid Mobile Application Development Framework I.e. Ionic Framework and PHPDesktop GUI Framework, to use the Command Line Interface, and to get the knowledge of other development technologies such as JavaScript, TypeScript, AngularJS, PHP, JSON, and MySQL.

The development of this project using frameworks made it easier but learning completely new languages such as TypeScript and AngularJS for Ionic Framework was challenging. The process of debugging, creating or running the application through CLI became easier and faster than it was at the beginning.

This project has given the opportunity to learn new programming languages and the techniques to handle the issues which appeared during the development of this project.

10.1 Future Tasks

Although this project has achieved its objectives and goals, there are still some functionalities which can be included in the future so that the application becomes more user-friendly and after maximum use experience. The future task of implementing a map function on the homepage which shows the total kilometers and other details automatically rather than manual input is one functionality that could still be studied in the future.

REFERENCES

- /1/ Overall hybrid application structure. Accessed 25.03.2017
<https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/>
- /2/ Hybrid Application Development. Accessed 26.03.2017
<https://www.codeproject.com/Articles/892677/Developing-Hybrid-Mobile-Apps-with-Phonegap-Angula>
- /3/ Ionic Framework. Accessed 26.03.2017
<https://ionicframework.com/>
- /4/ Ionic CLI. Accessed 27.03.2017
<https://ionicframework.com/docs/cli/>
- /5/ Ionic Components. Accessed 27.03.2017
<https://ionicframework.com/docs/components/>
- /6/ Theming. Accessed 27.03.2017
<https://ionicframework.com/docs/theming/theming-your-app/>
- /7/ Navigation. Accessed 27.03.2017
<https://ionicframework.com/docs/api/navigation/NavController/>
- /8/ Cordova. Accessed 28.03.2017
<https://cordova.apache.org/docs/en/latest/guide/overview/index.html>
- /9/ Plugins. Accessed 28.03.2017
<https://market.ionic.io/plugins>
- /10/ AngularJS and Ionic Relation. Accessed 28.03.2017
<https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/>
- /11/ AngularJS Overview. Accessed 28.03.2017
<https://www.tutorialspoint.com/angularjs/>
- /12/ JavaScript Tutorial. Accessed 29.03.2017
<https://www.tutorialspoint.com/javascript/>
- /13/ TypeScript Tutorial. Accessed 01.04.2017
https://www.tutorialspoint.com/typescript/typescript_overview.html
- /14/ TypeScript Features. Accessed 04.04.2017

<https://ionicframework.com/docs/resources/typescript/>

- /15/ HTML Overview. Accessed 05.04.2017
<https://www.upwork.com/hiring/development/html-vs-xhtml-vs-html5/>
- /16/ CSS Overview. Accessed 06.04.2017
<https://www.w3schools.com/css/>
- /17/ PHP. Accessed 04.04.2017
<https://www.tutorialspoint.com/php/index.htm>
- /18/ Application Development Process. Accessed 08.04.2017
<https://developer.android.com/studio/workflow.html>