SOFTWARE PLATFORM FOR CELLULAR SYSTEM
PERFORMANCE EVALUATION & RAW DATA COLLECTION
(MOBILE & WEB APPLICATION)

By
MOHAMMED MAHMOUD MOHAMMED WASFI
INDEX NO.124091

Supervisor
Prof. Sami Sharif

A REPORT SUBMITTED TO
University Of Khartoum
In partial fulfillment for the degree of
B.Sc. (HONS) Electrical and Electronics Engineering
(ELECTRONIC SYSTEMS SOFTWARE ENGINEERING)
Faculty of Engineering
Department of Electrical and Electronic Engineering
October 2017
DECLARATION OF ORGINALITY

I declare this report entitled “Software Platform for Cellular System Performance Evaluation & Raw Data Collection” is my own work except as cited in references. The report has been not accepted for any degree and it is not being submitted currently in candidature for any degree or other reward.

Signature: ____________________
Name: _______________________
Date: ________________________
DEDICATION

To my pillar of strength, my mother.

To my family, friends, and loved ones, thank you for being there in time of need.
ACKNOWLEDGMENT

All thanks and appreciations to my mentor and supervisor Prof. Sami Sharif for his continuous support and guidance. The warmest gratitude to the NTC staff for their continuous support and patience during the past months.

To my colleague, and project partner, who helped by motivating and encouraging me during the past year.
ABSTRACT

The mobile network operators always aspire to provide the best services to their clients, without any loss of data or interferences. Therefore, the ability of the mobile network operators to evaluate the performance of their networks is now more important than ever. One of the methods used to evaluate the performance of the network by collecting and analyzing raw data from the network is the Drive Test method. Although the Drive Test method fulfil the needs of the mobile networks operators, it is considered expensive and time consuming.

In this thesis, a software platform is introduced as an alternative cheaper method that makes the data available at any time. The platform consist of a mobile application for raw data gathering, which is installed on the mobile phones of the clients. The data gathered is then sent to a web application, which is used to review the data and represent it in different forms for ease of read and analysis.

The aims and objectives of this thesis was met, and the results obtained is encouraging. The platform was able to collect useful raw data, which might be used to evaluate the performance of the network. In addition, it was able to display the coverage and the quality of the network using the data acquired.

The software platform was reviewed to state all possible future work, which can make it a suitable replacement for the Drive Test method.
المستخلص

مشغلي شبكات المحمول يطمحون دائماً لتقديم أفضل الخدمات لعملائهم، دون أي فقدان للبيانات أو التداخلات. لذلك، أصبحت قدرة مشغلي شبكات المحمول على تقييم أداء شبكاتهم أكثر أهمية من أي وقت مضى. إحدى الطرق المستخدمة لتقييم أداء الشبكة من خلال جمع وتحليل البيانات الأولية من الشبكة هي الاختبار باستخدام السيارة، على الرغم من أن الاختبار باستخدام السيارة يلبي احتياجات مشغلي شبكات المحمول، إلا أنه يعتبر مكلف ويستغرق وقتاً طويلاً.

في هذه الاطروحة، يتم تقديم منصة برمجيات كوسيلة بديلة رخيصة تجعل البيانات متاحة في أي وقت. تتكون المنصة من تطبيق هاتف محمول لجمع البيانات الأولية، والذي يتم تحميله على الهواتف المحمولة للعملاء. ثم يتم إرسال البيانات الأولية التي تم جمعها إلى تطبيق ويب، والذي يستخدم لمراجعة البيانات وعرضها في أشكال مختلفة لكي تكون سهلة القراءة والتحليل.

وقد تحققت أهداف الاطروحة والنتائج الأولية التي تم الحصول عليها مشجعة. حيث تمكن المنصة من جمع بيانات أولية مفيدة يمكن استخدامها لإختبار أداء الشبكة. وبالإضافة إلى ذلك، كانت المنصة قادرة على عرض تغطية و نوعية الشبكة باستخدام البيانات التي تم الحصول عليها.

تمت مراجعة المنصة لتعرف على جميع الأعمال المستقبلية الممكنة، والتي يمكن أن تجعلها بديلاً مناسباً لطريقة الاختبار باستخدام السيارة.
# TABLE OF CONTENTS

Dedication ................................................................................................................................. iii

Acknowledgement ...................................................................................................................... iv

Abstract ........................................................................................................................................ v

المستخلص ................................................................................................................................... vi

Table of Contents ........................................................................................................................ vii

List of Figures .............................................................................................................................. xi

List of Tables ............................................................................................................................... xii

List of Abbreviation ..................................................................................................................... xiii

CHAPTER 1: Introduction ............................................................................................................ 1

1.1 Overview ................................................................................................................................ 1

1.2 Problem Statement .................................................................................................................. 1

1.3 Objectives ............................................................................................................................... 2

1.4 Thesis Layout ........................................................................................................................ 2

CHAPTER 2: Literature Review .................................................................................................. 3

2.1 Drive Testing: .......................................................................................................................... 3

2.1.1 Drive Testing by MNOs .................................................................................................... 3

2.1.2 Drive Testing by a third party companies .................................................................... 4

2.1.3 General Reasons to Use Drive Test ............................................................................. 4

2.1.4 Data Acquired from Drive Test ..................................................................................... 4

2.1.5 Drive Test Equipment’s ............................................................................................... 6

2.1.6 Drive Test Types .............................................................................................................. 7

2.1.6.1 Network Benchmarking ......................................................................................... 7

2.1.6.2 Optimization and Trouble Shooting ..................................................................... 7
2.1.6.3 Service Quality Monitoring

2.1.7 How it works

2.2 Related Work

2.2.1 GSM Signal Monitoring:

2.2.1.1 INFO Tab:

2.2.1.2 Strength chart tab:

2.2.1.3 Speed chart tab:

2.2.2 Network Signal Info:

2.2.2.1 Mobile Signal Tab:

2.2.2.2 Wi-Fi Signal Tab (WLAN):

CHAPTER 3: Methodology

3.1 Overview

3.2 Technologies Used

3.2.1 Android OS

3.2.2 SQLite Database

3.2.3 .NET Core

3.2.4 ASP .NET Core

3.2.5 RESTful API

3.3 Android Application

3.3.1 Data collection

3.3.1.1 Signal strength

3.3.1.2 Operator’s information

3.3.1.3 Neighboring cells information

3.3.1.4 GPS location
LIST OF FIGURES

Figure 2.1 Drive Test Equipment’s .............................................................. 6
Figure 2.2 Shows a screenshot of GSM Signal Monitoring’s ‘INFO’ tab ................. 9
Figure 2.3 Screenshot of the GSM Signal Monitoring’s ‘CHART’ tab ..................... 10
Figure 2.4 Shows screenshot of GSM Signal Monitoring’s ‘SPEED’ tab ................. 11
Figure 2.5 Screenshot of Network Signal Info’s ‘MOBILE’ tab .......................... 13
Figure 2.6 Screenshot of Network Signal Info’s ‘WLAN’ tab ............................ 14
Figure 3.1 Mobile application operation process ............................................. 16
Figure 3.2 Web application operation process ................................................. 16
Figure 3.3: MVC architecture (Courtesy of Microsoft) ....................................... 27
Figure 3.4 Locations table ............................................................................. 28
Figure 3.5 Signals table .................................................................................. 28
Figure 3.6 RESTful API (Courtesy of Microsoft) ............................................. 31
Figure 4.1 Android application ..................................................................... 34
Figure 4.2 Sample of locations data .............................................................. 35
Figure 4.3 Sample of the signals data ............................................................. 36
Figure 4.4 Google map used to display the data ............................................. 37
Figure 4.5 Information of a point displayed on the google map ......................... 37
Figure 4.6 The variation of the geo. locations accuracy .................................. 38
Figure 4.7 Number of neighboring cells, which the system connected with ........ 39
Figure 4.8 Types of the different networks ..................................................... 39
Figure 4.9 Signal strength of the collected data ............................................. 40
Figure 4.10 Signal strength distribution .......................................................... 41
LIST OF TABLES

Table 3.1 Table showing the mobile OSs market share .......................................................... 17
Table 3.2 The parameters of the locations table .................................................................... 28
Table 3.3 The parameters of the signals table ...................................................................... 29
Table 3.4 Signal quality labeling .......................................................................................... 30
# LIST OF ABBREVIATION

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNO</td>
<td>Mobile Network Operator</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>QoE</td>
<td>Quality of Experience</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Application</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunications Service</td>
</tr>
<tr>
<td>LTE</td>
<td>Long-Term Evolution</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>EDGE</td>
<td>Enhanced Data rates for GSM Evolution</td>
</tr>
<tr>
<td>HSPA</td>
<td>High Speed Packet Access</td>
</tr>
<tr>
<td>HSDPA</td>
<td>High-Speed Downlink Packet Access</td>
</tr>
<tr>
<td>HSUPA</td>
<td>High Speed Uplink Packet Access</td>
</tr>
<tr>
<td>MCC</td>
<td>Mobile Country Code</td>
</tr>
<tr>
<td>MNC</td>
<td>Mobile Network Code</td>
</tr>
<tr>
<td>LAC</td>
<td>Location Area Code</td>
</tr>
<tr>
<td>TAC</td>
<td>Tracking Area Code</td>
</tr>
</tbody>
</table>
**RSSI**  Received Signal Strength Indication  

**Wi-Fi**  Wireless Fidelity  

**ASU**  Arbitrary Strength Unit  

**IP**  Internet Protocol  

**SSID**  Service Set Identifier  

**dB**  Decibel  

**WLAN**  Wide Local Area Network  

**API**  Application Programming Interface  

**OS**  Operating System  

**SQL**  Structured Query Language  

**TCP**  Transmission Control Protocol  

**SMS**  Short Messaging Service  

**RAM**  Random Access Memory  

**MVC**  Models Views Controllers  

**HTML**  Hypertext Markup Language
CHAPTER 1: INTRODUCTION

1.1 Overview

In today’s world of personal computers and smart phones, everyone wants to stay connected with the rapid changing world events. Every user expect a high performance cellular network to service all his needs and requests, without any delays, interference, or loss of service at anytime and anywhere.

Mobile network operators have never faced such a challenge of providing the best quality of service with the best quality of experience. In order to achieve this task MNOs have to ensure that their cellular network performance meets the user expectations and needs, and beyond, at all time. MNOs spend a considerable percentage of their yearly budgets in benchmarking, maintaining, optimizing, fixing, and upgrading their cellular network to achieve their goal of providing the best quality of service with the best quality of experience.

MNOs use different methods to test their cellular network performance. One of them is the Drive Test. Although Drive Test fulfil the MNOs need of pointing out the problems in the network and evaluate its performance, the Drive Test cost is very high due to different aspects, from the need of hiring specialized engineers to the cost of the equipment’s used to perform the test. This tradeoff will cause a huge problem to the MNOs, which will reflect on the quality and the performance of their cellular network. This will directly affect the user of the cellular network.

1.2 Problem Statement

The high cost of the Drive Test and the time it takes to conduct it, make it not possible for the MNOs to conduct it more often, therefore the performance of their networks, the QoS, and the QoE might decrease rapidly.

Therefore, a software platform is proposed to replace the Drive Test method, by offering the same output results of the Drive Test with a significant decrease in the cost.
1.3 Objectives

The objective of this project is to provide the MNOs with a software platform that serve some of the Drive Test functionalities as raw data collecting tool with a much cheaper cost by using the mobile phones as the data collecting tool. Since the mobile phones are widely spread and covers a wide geographical area, and the required data can be collected using the phone hardware just by using a software interface; that make the mobile phones a suitable tool. Therefore, the software platform will help the MNOs to lower the costs and save valuable time.

1.4 Thesis Layout

The rest of this thesis is organized as follows:

- **Chapter 2 (Literature Review):** The first part of this chapter is a review of the Drive Test method. It covers the reasons to conduct the test; the types of the data acquired, the test equipment’s, the types of the test, and sums it all by showing how the test is conducted. The second part talks about the related work by taking two mobile apps as an example.

- **Chapter 3 (Methodology):** It talks about the technologies and the detailed steps taken to develop the whole platform. Starting with the design of the android application and its core methods. Then it talks about ASP.NET Core web application highlighting its design and role in the platform. Finally, it illustrate how the RESTful API is used and conclude it all by demonstrating the security measures implemented in the platform.

- **Chapter 4 (Results Analysis and Discussion):** This chapter contains the results of the data collection process shown in its final form as a graphical interface on a google map, after it has been analyzed. The chapter also discuss the variations of the data collected by the platform and the cause of those variations.

- **Chapter 5 (Conclusion and Future Work):** Contains comments about the project and the work done, focusing on what can be done to improve the performance of the platform and suggesting possible future work.
CHAPTER 2: LITERATURE REVIEW

This chapter discusses the basic concepts of standard data collection of network parameters like drive testing techniques and what are their drawbacks, it also covers some similar apps that are used to collect these parameters and their limitations.

There are different methods MNOs can use to measure the performance of their networks such as Drive Testing and RF surveys.

2.1 Drive Testing:

Drive testing is a method of measuring the coverage, capacity, QoS, and QoE levels of a mobile network. It is implemented by deploying vehicles containing mobile radio network air interface measurement equipment and various selected mobile handsets that can measure a broad range of physical and virtual parameters of mobile cellular services in a defined geographic region.

By measuring what a customer is experiencing in any particular location, MNOs can make changes to their networks to enhance coverage and service levels. Drive testing requires a mobile vehicle outfitted with drive-testing measurement equipment as well as multiple popular mobile devices connected to multiple mobile networks. There are two ways MNOs can leverage drive testing to obtain network performance data[1]:

- Drive testing by MNOs; using internal resources.
- Drive testing by a third party companies; hiring specialized drive-testing firms or subscribing to their reports.

2.1.1 Drive Testing by MNOs

Drive testing by MNOs is implemented by using the internal resources (Engineers, vehicles, etc.) of the MNOs to drive-test their services. This method is flexible in the terms of defining the routes and the frequency of the testing, which is a tradeoff between the cost and the budget. The ability of tailoring the test routes and the frequency of the tests allow the MNOs to focus resources on the problem location. MNOs can equip their vehicles with mobile devices connected to competitor networks to get an idea of the competitor network performance.
Drive testing using internal resources also has cons, the primary one being that it is extremely costly to outfit a fleet of vehicles and hire dedicated, skilled engineers to survey locations and collect data. It does not easily enable trend analysis, since in most scenarios geographic locations are only surveyed a couple times a year. Operators also have a limited-time window for collecting the data[1].

2.1.2 Drive Testing by a third party companies

Third party testing is more popular than using internal resources in testing, because it is less expensive and provide better geographical coverage for the MNOs. The test data is more objective and allows the MNOs to highlight their advantages over competitors from a performance perspective.

2.1.3 General Reasons to Use Drive Test

- To check the performance of new deployed network sites in order to meet coverage, capacity, and quality requirements
- Optimization of the network
- Benchmarking of performance
- Trouble shooting
- To verify the performance after an upgrade or reconfiguration of the network

2.1.4 Data Acquired from Drive Test

The information acquired from the drive test include but not limited to:

- Signal strength:
  Refers to the transmitter power output as received by a reference antenna at a distance from the transmitting antenna

- Signal quality:
  It determines how good the signal that you receive during different conditions is.

- Interference:
  The noise or interruptions that affect the received signal due to weather conditions for example.
• Dropped calls:
  Determines the number of calls, which have been cut off before the users finish their conversation.

• Blocked calls:
  A block call acquires when a call cannot be connected.

• Anomalous events:
  Irregular events that happen in the network.

• Call statistics:
  A quantitative information of all the calls of the network[2].

• Service level statistics:
  It records the measured performance of the network against the standard performance.

• Quality of Service information:
  It is a control mechanism for the reliability and the performance of the mobile network.

• Handover information:
  Transfer the data session, connection, or ongoing call from one cell tower to another without interruption or loss of data.

• Neighboring cell information:
  All the previous information for all the neighboring cells registered in the same tower of our cell in use.

• GPS location co-ordinates:
  The latitude and longitude of all the points along the test route.
2.1.5 Drive Test Equipment’s

The basic equipment’s required to perform the drive test include[3] as shown in figure [2.1]:

- A vehicle:
  This can be any kind of an automobile used by the test engineer to follow a predefined route in order to collect data.

- Measurement devices:
  This can be a number of mobile devices. They can be used to measure the network coverage, signal strength, upload speed, download speed, and switching from one cell tower to another.

- A GPS device:
  Used to locate the latitude, and longitude of the desired location on the test route.

- A laptop or a PC:
  Used to store the collected data from the test along the route for analysis.

- Software analysis program:
  Used to analyze the collected data by the test engineers in real time or later, according to the desired output information.

Figure 2.1 Drive Test Equipment’s[4]
2.1.6 Drive Test Types

Drive test can be categorized into three categories based on the purpose of the output result[5]:

- Network benchmarking
- Optimization and trouble shooting
- Service quality monitoring

2.1.6.1 Network Benchmarking

Collect the data to determine the minimum network coverage, QoS, and QoE, which will be used to determine the minimum network performance required.

2.1.6.2 Optimization and Trouble Shooting

Measurements collected from the test will be analyzed to determine the source of a problem in a network or to determine what part of the network can be upgraded or optimized to increase the performance of the network.

2.1.6.3 Service Quality Monitoring

Measure the network performance at a certain geographical location and compare it to the determine standard to check whether the network is performing as expected or it needs to be optimized.

2.1.7 How it works

The test engineer decides the route of the test, which the vehicle will follow. The equipment is checked and prepared for the test. Using three mobile phones, one to measure the coverage, while the other two phones to collect calls related data (dropped calls, blocked calls, etc...). The three phones are connected with a laptop with a software used for analyzing and displaying the data in real time and at the end of the test. A GPS modem is connected with the laptop to save the geographical locations along the route.

The test starts when the vehicle reaches the start of the route. The data collection process starts by the three mobile phones and sent to the laptop. At the same time, the GPS modem saves
the location of each point along the route and associate each location with the signal collected at that point.

2.2 Related Work

Since the development of mobile devices, the issue of signal strength and network coverage was a constant headache for network providers, the process of covering an area involves building cell towers and constantly maintaining it and keeping up with new technologies.

The quality of service represents an important requirement for the customers and any weak performance might affects the providers’ business state. Meaning that network providers are always in search for any helpful information about the network coverage at anywhere, the process of gathering this data is rather an expensive one and is a headache that they also try to avoid.

There is a note worth mentioning, as any cellphone user can know there is bar indication of the signal power coverage (full bar is very an indication of excellent signal power, 3 bars is fair or good, 1 bar is weak and so on), but the fact is that these bars are subjective across carriers and network providers, for example what might seem like a full bar for MTN (a Sudanese operator) maybe only a good or 3-bar signal for ZAIN (another Sudanese operator). The point is there is no Standard for the bar view so it can’t be used to as a reliable source of information and thus the need for a more quantitative and objective measurement using the decibel gain[6].

In the past, few years there were several apps and approaches to solve this problem, the standard drive test mechanism was discussed earlier, now a more similar approach to our solution is inspected i.e. mobile apps that have similar purpose are discussed next.

2.2.1 GSM Signal Monitoring:

Is a specialized application for Android devices which checks parameters of GSM, UMTS and LTE networks[7].

Its main features are:

- Information about current mobile operator.
- Type and status of mobile connection.
• Uplink and downlink speed of mobile connection.
• Information about current cell.
• Data about strength of base tranceiving station signal.

There are three main tabs in the application main view discussed next.

2.2.1.1 INFO Tab:

The basic idea is that there is background service - a service or thread that runs in the background without the user interference - that is used to obtain the basic readings containing but not limited to the following:

• Name of the current MNO.
• Mobile Country Code (MCC).
• Mobile Network Code (MNC).
• Current network type (GPRS/EDGE/UMTS/HSPA/HSDPA/HSUPA/LTE).
• Speed of incoming and outgoing connection.
• Cell identifier (CID).
• Location Area Code (LAC) — for GSM/UMTS networks.
• Tracking Area Code (TAC) — for LTE networks.
• Received Signal Strength Indication (RSSI).

Then the app uses this data to show overview information about the current state of the network. Figure [2.2] shows a screenshot of the app showing basic overview data.

Figure 2.2 Shows a screenshot of GSM Signal Monitoring’s ‘INFO’ tab
2.2.1.2 **Strength chart tab:**

Strength chart tab displays the change of received signal strength (RSSI). Line colors correspond to the colors of cell IDs in the previous tab. The chart updates every second. Figure [2.3] shows the signal strength chart.

![Figure 2.3 Screen Shot of the GSM Signal Monitoring’s ‘CHART’ tab](image)

2.2.1.3 **Speed chart tab:**

This tab contains charts of upload and download speed. The speed is measured for GSM/UMTS/LTE networks. If Wi-Fi is on the charts will be empty because no data is transmitted via cellular network. Figure [2.4] shows the upload and download speed statistics.
Figure 2.4 Shows screenshot of GSM Signal Monitoring’s ‘SPEED’ tab

The app has other views for preferences, logs showing and data clearing abilities.

2.2.2 Network Signal Info:

Provides detailed information on your currently used network, regardless whether you are using Wi-Fi or a cellular connection, in addition to the mobile signal data this app also provides helpful data about the Wi-Fi signal[8].

The “Mobile Signal” data this app provides includes but not limited to:

- Network operators.
- Sim provider.
• Phone type.
• Network type.
• Signal strength in dBm and ASU.
• Country code of the mobile phone.
• Device ID.
• IP address (internal and external).

The “Wi-Fi Signal” data this app provides includes but not limited to:

• Wi-Fi Name (SSID).
• MAC address.
• Maximum Wi-Fi speed.
• IP address.

The app consists of four tabs, but the two main and most important tabs are the “Mobile Signal” tab and the “Wi-Fi Signal” tab.

2.2.2.1 Mobile Signal Tab:

In this tab, there is a meter-like view to indicate the signal strength; this bar has 14 level ranging from the lowest strength -120dB to the most powerful one -50dB. In addition to that, the tab also presents other basic information about the signal data. Figure [2.5] shows this view as described.
2.2.2.2 Wi-Fi Signal Tab (WLAN):

Like the ‘Mobile Signal’ tab, this tab have bar indication to demonstrate Wi-Fi signal strength, this bar however range from lowest -100dB to most powerful ~ -30dB not like the mobile signal strength. In addition to that, the tab also provides basic information about the Wi-Fi signal. Figure [2.6] shows how this aligns in the app.
Figure 2.6 Screenshot of Network Signal Info’s ‘WLAN’ tab

It is important to note that although numerous apps with the same or similar goal are available, they all have the same model or idea to show the current state of network—mobile of Wi-Fi—and present basic information about the user own device. This is not the aim of this solution, as mentioned earlier the idea is to use user’s base to collect as much data as possible and store this data for more advanced and oriented reports to be used by targeted organizations—like network providers—to enhance the service provided. Meaning that the app is merely a tool to collect data.
CHAPTER 3: METHODOLOGY

3.1 Overview

The system consist of four parts. First, the mobile application, which act as the collecting service, installed on a smartphone, which act as the measurement instrument. The mobile application collect different data parameters such as the geographical location using the GPS, the signal strength, the network type, etc... Then the mobile application stores all the collected data on a local storage before sending it to a central database using a RESTful API.

The web application is the second main part, which is used to display the results after analyzing the raw data.

The third part is the central database, which acts, as the main storage unit for both the mobile application and the web application were all the raw data is being saved and can be accessed directly.

The fourth and final part is the RESTful API, which acts as a communication channel between the mobile application and the database, and between the web application and the database. There is no direct connection between the mobile application and the web application. The only connection between the two applications occur by altering data in the database by using the RESTful API.

In figure [3.1], the mobile application operation process is represented. The mobile phone receives the cell data, GPS data, and other required data. Then by the use of the RESTful API the mobile phone start a connection with the web service and authenticate itself before sending the data to the central database.
In figure [3.2], the web application operation process is represented. A connection is started between the web service and the web application. After authentication, the data is sent from the database to the web application. Then the data is analyzed at the backend of the web service and the final output is represented as a graph on google maps.

Figure 3.1 Mobile application operation process

Figure 3.2 Web application operation process
3.2 Technologies Used

Below is the list of technologies and components used in this solution:

- Android OS
- SQLite and SQL Database
- ASP.Net Core
- RESTful API

3.2.1 Android OS

Android is a mobile operating system developed by Google, based on Linux kernel and designed primarily for touchscreen devices.

Android-based phones are used for the following reasons:

- Android is the most common mobile phones OS in the world. Simply there are more users using android devices than apple ones especially in Sudan. In addition, this leads to a bigger support community that provides answers to almost any problem. Table [3.1] demonstrates the market share distributions. Table [3.1] shows the percentages of mobile operation systems market share [9].
- Android is an open source OS, so it provides easy access to device hardware interface and there are many available libraries related to the solution.

“Open-source software is software that makes the source code freely available, for anyone to see and use.”[10].

Table 3.1 Table showing the mobile OSs market share

<table>
<thead>
<tr>
<th>OS</th>
<th>Percentage %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>73.53%</td>
</tr>
<tr>
<td>iOS</td>
<td>19.37%</td>
</tr>
<tr>
<td>Other</td>
<td>7.1%</td>
</tr>
</tbody>
</table>
3.2.2 SQLite Database

SQLite is a relational database management system contained in C programming library. In contrast to many other database management systems, SQLite is not a client-server database engine. Rather, it is embedded into the end program. It is arguably the most widely deployed database engine[11].

Simply it is a storage method available in android and that has many advantages over other storage methods:

- **Zero-Configuration:**
  SQLite does not need to be "installed" before it is used. There is no "setup" procedure. There is no server process that needs to be started, stopped, or configured. There is no need for an administrator to create a new database instance or assign access permissions to users. SQLite uses no configuration files. Nothing needs to be done to tell the system that SQLite is running. No actions are required to recover after a system crash or power failure. There is nothing to troubleshoot[12].

- **Server less:**
  Most SQL database engines are implemented as a separate server process. Programs that want to access the database communicate with the server using some kind of inter-process communication (typically TCP/IP) to send requests to the server and to receive back results. SQLite does not work this way. With SQLite, the process that wants to access the database reads and writes directly from the database files on disk. There is no intermediary server process[12].

- **Cross-Platform:**
  A database file written on one machine can be copied to and used on a different machine with a different architecture[12].

- **Stable and open source**[12].

Add to that the simplicity and the fact that is implementing the SQLite is relatively easy process, because all of this the SQLite was found to be better suited to the needs of the solution.
3.2.3 .NET Core

.NET Core is a lightweight and modular platform for creating web applications and services that run on Windows, Linux and Mac. It is the cross-platform fork of .NET and it is open source.

3.2.4 ASP .NET Core

ASP .NET Core is a free and open-source web framework, and the next generation of ASP.NET, developed by Microsoft and the community. It is a modular framework that runs on both the full .NET Framework, on Windows, and the cross-platform .NET Core[13].

3.2.5 RESTful API

A RESTful API is an application program interface that uses HTTP requests to GET, PUT, POST and DELETE data. It is based on representational state transfer REST technology, an architectural style and approach to communications often used in web services development.

A RESTful API explicitly takes advantage of HTTP methodologies defined by the RFC 2616 protocol. They use GET to retrieve a resource, PUT to change the state of or update a resource, which can be an object, file or block, POST to create that resource, and DELETE to remove it[14].

3.3 Android Application

First part of the solution is an android app distributed to end users or selected users to perform the first step, which is getting the data.

The app itself performs five following major functions.

3.3.1 Data collection

Here the android platform and the API is used to gather and obtain all the fields concerned with the presented solution. The fields and parameters gathered are:

3.3.1.1 Signal strength:

Indicator of the mobile signal quality, measured in dBm and ASU.
• dBm is an abbreviation for the power ratio in decibels (dB) of the measured power referenced to one mill watt (mW).

• Arbitrary Strength Unit (ASU) is an integer value proportional to the received signal strength measured by the mobile phone.

\[ dBm = 2 \times ASU - 113 \]

ASU is in the range of 0-31 and 99 (for not known or not detectable)[15].

3.3.1.2 Operator’s information:

Fields like CELLID, MCC, MNC and LAC are obtained.

• CELLID is a unique identifier that is used to distinguish each sim card and map it to a specific operator.

• MCC stands for mobile country code and MNC stands for mobile country code. Together they are used to identify a mobile network operator (carrier).

• LAC stands for location area code, which is used to map every sim card to a single area block.

3.3.1.3 Neighboring cells information:

Collect the previous fields and parameters for all neighboring cells to current cell in use. By neighboring, it is meant all the cells that shares the same tower.

Keeping in mind that this particular feature is depended on the device hardware meaning that some vendors allow collecting the neighboring cells data (nexus) while others do not (Samsung).

3.3.1.4 GPS location:

With every read of the fields a GPS (global positioning system) location is associated with it the is used later on to filter the performance of every region.

The GPS feature on android has several modes:

• High accuracy – high battery consumption.

• Fair accuracy - moderate battery consumption.

• Low accuracy – low battery consumption.
While the accuracy of the location is important, a tradeoff is presented with the battery life leading to choosing the fair accuracy mode.

When obtaining the GPS coordinates, two approaches regarding the android API used, present themselves:

- There is the old standard `LocationManager` class to get the GPS data. This method was there from the early versions of android. It simply works by returning the last GPS coordinates obtained by app or system app i.e. this location can be 1 second, 2 hours or few days old and its location accuracy is ranges (8 to few hundred meters) depending on the device vendor and hardware, so while it’s simple and stable it’s not reliable when accurate readings are needed.

- There is also the relatively new `GoogleApiClient` class. It uses google play services - a system app by google user for various purposes like maps, games …etc. - to obtain GPS readings, it is simple, returns more updated readings and with better accuracy. The only drawback of this method that it needs the ‘Google play services’ to be updated to the last version.

Using the second method of `GoogleApiClient` may collect less amounts of data, but it guarantees more reliable set of readings so it is used instead of the first method.

### 3.3.2 Data storage

Android OS offers multiple storage approaches: flat files, SQLite and shared preferences.

- Shared preferences is used to store small amount of data on a scale of app preferences (language, username …etc.) and offers little flexibility in data format so it is discarded from use.

- Flat files are used to store larger amount of data but the modification and deletion process are more complicated than SQLite [16].

- SQLite have the following advantages:
  - Useful when storing large amount of data up to thousands records without drop in performance unlike flat files.
- Stores the data in a structured format, which helps to obtain certain records on certain conditions.
- Allows data extraction in a more smooth way (no need to get the whole file and search what you need like flat files approach)[16].

Taking the previous considerations into account SQLite database is used to store the data. After every read of parameters, all the data is saved into SQLite tables following the pre-defined database schema.

Since the gathering frequency is varying and it can be the case the every read is few minutes from the previous which can lead to very huge bulky database and although the threshold of performance drop is rather far it is inevitable to reach it.

The solution to the previous problem is to delete every record after sending it the centralized database, as we will see shortly.

3.3.3 Data sending to centralized Database

Now that all needed parameters are stored, they are sent to a centralized database for further processing and analysis.

The sending operation can be achieved in multiple ways: SMS and internet.

- SMS approach:
  - Since the data can be rather huge, it is impractical to use SMS since it will cost a lot of account balance (unless a special agreement is performed with the operator).
  - The only advantage of this way is that it is more reliable and is not affected by internet performance (it can be used in rural areas in which the internet performance is poor).

- Internet approach:
  - More practical because of the size of the data.
  - Is entirely dependent on the network state.
  - Can also depend on the user subscription quota and internet service of choice.
Taking the above into consideration the internet approach is found a better fit to use under the current circumstances.

A special API is developed specially for this purpose using the .NET technology and is used to send the collected data to pre-defined centralized database.

A more clarification about this API will be covered in the WEB part of the solution.

3.3.4 Control over gathering service

The proposed idea is to collect the data on a continuous way with a certain time between each read of fields. This certain time will be referred to as gathering frequency.

In order to obtain all the required parameters in a single read some of the phone resources are used like GPS, network usage and data usage. On a single read, the phone will not be affected but on a span of the day, week or month this might raise questions on the phone performance. We take several aspects of the phone resources and examine each one:

- **RAM:**
  - This would have been a concern if the application required to be opened on screen to function, since that is not the case the only need for ram is to perform background operations like: GPS and internet related operations, which compared to foreground operations, do not put stress on RAM.
  - Foreground operations may contain anything require the app to be on the screen foreground and graphical operations (scrolling on a list for example).

- **Battery:**
  - The phone battery can easily be consumed if the application is not optimized. Things like GPS, mobile data and data transfer over the internet affect the battery directly.
  - So a moderate approach is used in GPS location as mentioned previously.
  - Data transfer should be minimized with steps like these: the data should not be sent too often and the API optimized to exclude any unnecessarily operations.

Below a google engineer, named **Robert Love** is quoted about why the GPS drains the battery. This is from an answer to question on subject posted on Quora website:
“GPS is expensive because it is a very slow communication channel; you need to communicate with three or four satellites for an extended duration at 50 bits per second. Mobile devices such as Android and the iPhone achieve their battery life largely because they can aggressively and quickly enter into and exit from sleep states. GPS prevents this.

Even with (data-assisted satellite acquisition) A-GPS, using your GPS is a noticeable battery hog. Compounding the cost, most mapping software is processor-intense. A well-designed app can make a significant difference here; Google Maps boasts several optimizations to reduce battery consumption from GPS usage.”[17][18]

The step taken to address the battery issue is as follows:

- Use the fair accuracy mode when the GPS location is obtained.
- Use the minimum frequency value without affecting the goal and statement of the project.

This is done after a series of experiment and trials on the app on a span of a considerable value and after; trying different values of frequencies.

In addition, the frequency depends on the number of users on a specific area and the state of the network in the area.

With all, that being said a definite final value of frequency could not be determined and stated. Instead, the best-suited values is chosen by different admins of different areas.

3.3.5 Data Viewing

The app have a page that hold most useful information and controls, the page has the following components:

- A ‘START’ and ‘STOP’ Controls:
  - Those are menu buttons located at the action bar of the screen, the action bar is the bar located at top of the application holding the app name and main important actions like settings and so on.
  - The ‘START’ button starts the gathering service to work in the background, it’s important to mention that although the service works by itself collecting data it needs to be started by the user and it will keep working until it’s
stopped by the user even if the app is removed from the ‘RAM’ or memory stack. Removing an app from RAM usually just stops the foreground operations of graphical processing which is not the case right now.

- The ‘STOP’ button simply stops the service; the service will no longer collect any data unless it is re-started by the user.
- Another approach could’ve been taken in which the service simply works in the background, with no interaction from the user to start or stop it, as long as possible. The app simply registers new readings, but as discussed earlier the process of acquiring the needed info can consume phone resources like: RAM, battery life and therefore the user is given the liberty to control the collection period.
- Another reason for enabling control of the collection period is simply that there may not be a need to collect at the current time, the user might had enough data about a specific place (home for example), the readings seems to be consistent and the network may be stable so there is no need to acquire new readings.

- The Synchronize icon:

  Also located at the action bar, it’s an icon when pressed initiates the process of uploading the stored readings to a pre-defined centralized database located at a pre-defined server. The uploading process consists of two main parts:

  - **Firstly:** Only the GPS location and its related information are uploaded, meaning that we first upload: the GPS coordinates, Time Stamp of the readings and the accuracy of those readings. All readings are uploaded at on packet or bulk of data. Keeping in mind that the linking between any location readings and its associated signal readings should be maintained.
  
  - **Secondly:** After the location data are uploaded successfully, the signal data is now uploaded. When uploading a single reading it is immediately linked with its already uploaded associated location counterpart.
Main page Data:
- The standard main page of the app is used to show a quick glance of the network state. Only few selected fields are shown and those fields are signal strength, Cell id, LAC, MCC, Operator name.
- Those fields are chosen just to give the user a way to have a quick idea about the current state of network. Also shown the date and time at when those readings are captured so to know how updated this data is.
- The information on this page is updated whenever possible i.e. with every new reading those fields are updated.

3.4 ASP .NET Core Web Application

The web application was designed using the MVC design (architecture) pattern, which will be explained, in section 3.4.1. The web application is used to display the final output data after it has been analyzed in a graphical form or a tabular form.

The web application gets the data from the centralized SQL database. The database receives the raw data from the mobile application by the use of the RESTful API. Then the web application retrieve the data from the database by the use of the RESTful API.

The raw data is analyzed as individual reading points or as a group of data grouped together according to their geographical location and time.

The result of the analysis is then displayed in a graphical interface as points and sectors on a google map interface, while the raw data is displayed in a tabular form for ease of search and read.

3.4.1 MVC Architecture

The Model-View-Controller is a standard architectural pattern that separates the application to three main components: the models, the views, and the controllers.

The client interact only with a controller, which process the requests of the client. Then the controller communicate with the models in order to retrieve the required data from the database by the related model. After retrieving the data, the controller then manipulate it as
required and then send it to the views in order to get the final output view, which is then sent back to the controller. At the end the controller, send these views to the client as a response[19].

- Models:
  Model objects are the parts of the application that implement the logic for the application's data domain. Often, model objects retrieve and store model state in a database.

- Views:
  Views are the components that display the application's user interface. Typically, this user interface is created from the model data.

- Controllers:
  Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render that displays UI. In an MVC application, the view only displays information; the controller handles and responds to user input and interaction.

![MVC Architecture](image)

**Figure 3.3: MVC architecture (Courtesy of Microsoft)**

### 3.4.2 SQL Centralized Database

A relational database schema has been designed with two tables, one for the Locations data as shown in figure [3.2], and one for the Signal data as shown in figure [3.3].
Table 3.2 The parameters of the locations table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>A unique identifier for each new entry</td>
</tr>
<tr>
<td>LocationText</td>
<td>The latitude and longitude of a geographical location read</td>
</tr>
<tr>
<td>Accuracy</td>
<td>The degree of accuracy of a geographical location read</td>
</tr>
<tr>
<td>Date</td>
<td>The date when the data was taken</td>
</tr>
<tr>
<td>Time</td>
<td>The time when the data was taken</td>
</tr>
</tbody>
</table>
Table 3.3 The parameters of the signals table

<table>
<thead>
<tr>
<th>Id</th>
<th>A unique identifier for each new entry.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>The date when the data was taken.</td>
</tr>
<tr>
<td>NetworkType</td>
<td>The type of network whether it is 2G, 3G, or 4G in general.</td>
</tr>
<tr>
<td>Operator</td>
<td>The name of the MNO</td>
</tr>
<tr>
<td>SignalStrength</td>
<td>The power of the received signal by the mobile.</td>
</tr>
<tr>
<td>Registered</td>
<td>The information of the mobile phone sim card.</td>
</tr>
<tr>
<td>LAC</td>
<td>Location area code is a unique identifier for each geographical location.</td>
</tr>
<tr>
<td>MCC</td>
<td>The Mobile Country Code is a unique 3-digit number to identify a country.</td>
</tr>
<tr>
<td>CellInfo</td>
<td>This is the neighboring cell data.</td>
</tr>
</tbody>
</table>

3.4.3 Data Retrieval

The web application retrieve the data from the centralized database using the RESTful API by the GET command.

3.4.4 Data Analysis

Each data reading is classified according to its signal strength. The same goes to a group of data readings located at the same geographical region, using the average signal strength of the group. Then the quality of the signal is determined as shown in table [3.4].
### Table 3.4 Signal quality labeling

<table>
<thead>
<tr>
<th>Signal Quality</th>
<th>Associate Range of Signal Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good</td>
<td>$\geq -90 \text{ dBm}$</td>
</tr>
<tr>
<td>Fair</td>
<td>$-90 \rightarrow -100 \text{ dBm}$</td>
</tr>
<tr>
<td>Bad</td>
<td>$\leq -100 \text{ dBm}$</td>
</tr>
</tbody>
</table>

### 3.4.5 Data Display

The data is displayed in two forms, a tabular view and a map view.

#### 3.4.5.1 Table View

A list of all the readings in a tabular form for ease of access and display. Following the template shown in figure

#### 3.4.5.2 Map View

The data is displayed in two forms:

- Each data reading is displayed as a colored point determined on its signal quality. Good is green, fair is yellow, and bad is red.
- Each group of data readings located at a specified region are displayed in a circular fashion with the center of circle holding the average signal strength of the group. The radius of the circle is determined dynamically according to the need at that time, with the common values: 200 meters, 500 meters, and 1 kilometers.
3.5 Error Handling

Several error handling mechanisms have been implemented to ensure the continuous work of the system:

- Input data validity is checked instantly and returns a warning to the user stating the error if the data is not valid, otherwise the data is accepted and passed to the database.
- User input HTML tags are filtered in order to check whether they contain malicious scripts.

3.6 RESTful API

Both the mobile application and the web application use the RESTful API to connect with each other by either inserting or retrieving data from the database.

The mobile application uses the POST command of the RESTful API to insert the data into the database. While the web application uses the GET command to retrieve the data from the database, the PUT command to edit data in the database, and the DELETE command to remove data from the database.

![RESTful API Diagram](image)

*Figure 3.6 RESTful API (Courtesy of Microsoft)*
3.7 Security

3.7.1 Application

- SQLite Security:

  The database file is located at the path `/data/data/"project package name"/databases`, usually this path is not accessible without rooting the device so this makes it hard to retrieve any data without authorization.

  After android version 4.0, this has changed anyone with a physical access to the phone and a USB cable can gain access to databases. This is dubbed as an intentional feature not an oversight because of performance reasons, SQLite has a very good performance in storing and retrieving data[20].

  Before continuing, an important concept must be briefly discussed.

- Encryption:

  Encryption is the process of encoding a message or information in such a way that only authorized parties can access it. Encryption does not itself prevent interference, but denies the intelligible content to a would-be interceptor. In an encryption scheme, the intended information or message, referred to as plaintext, is encrypted using an encryption algorithm - a cipher - generating cipher text that can only be read if decrypted[11].

  There is an encryption and ciphering process that can be done but discarded for the following reasons:

  - There is a license by SQLite to perform encryption process, but it is not free. There was an approach to perform encryption manually but if someone can hack the phone to obtain the database, they can obtain encryption keys as well.
  - The type of data stored is not quite sensitive; the data is merely a log of readings. In addition, it is not severely important to keep it away especially if that requires buying licenses.
3.7.2 Centralized Database

Security measures have been taken to prevent SQL injection, by preventing SQL statements from being by concatenating strings that involve user input. Instead, a parameterized query is created and user input is used to set parameter values.

SQL injection is a web hacking technique implemented by the placement of malicious code in SQL statements, via web page input.
CHAPTER 4: RESULTS AND DISCUSSION

The data gathering process was concentrated around a specified geographical area for the sake of testing the system. Accordingly, most of the readings were taken within and around the University of Khartoum campus. The data collected by the platform was obtained using the mobile application installed on different mobile phones roaming around the university campus mainly.

4.1 Android Application

The android application, from different mobile phones collected the raw data and stored it in a local database before sending it to the central database. In addition, the android application displayed some of the parameters of the latest data acquired on the main app as shown in figure [4.1].

Figure 4.1 Android application
4.2 Web application

All the data collected by the different mobile phones using the mobile application is available on the web application to be reviewed in different forms.

4.2.1 Locations Data

All the locations data are displayed in a tabular form for ease of read and review. Figure [4.2] shows a sample of the collected locations data.

<table>
<thead>
<tr>
<th>Location Text</th>
<th>Time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.6116417,32.538735</td>
<td>2017-10-09 16:00:19</td>
<td>15.7</td>
</tr>
<tr>
<td>15.610033,32.539364</td>
<td>2017-10-10 10:35:55</td>
<td>20.0</td>
</tr>
<tr>
<td>15.610033,32.539364</td>
<td>2017-10-10 10:37:34</td>
<td>20.0</td>
</tr>
<tr>
<td>15.6113492,32.5388392</td>
<td>2017-10-11 15:39:56</td>
<td>6.068</td>
</tr>
<tr>
<td>15.6099307,32.5365325</td>
<td>2017-10-11 16:15:01</td>
<td>16.687</td>
</tr>
<tr>
<td>15.6031167,32.5397455</td>
<td>2017-10-11 16:19:59</td>
<td>15.17</td>
</tr>
<tr>
<td>15.5955002,32.5395833</td>
<td>2017-10-11 16:21:57</td>
<td>7.585</td>
</tr>
<tr>
<td>15.5933964,32.5414773</td>
<td>2017-10-11 16:22:57</td>
<td>9.102</td>
</tr>
<tr>
<td>15.5674416,32.5464237</td>
<td>2017-10-11 16:23:59</td>
<td>9.102</td>
</tr>
<tr>
<td>15.562999,32.5457798</td>
<td>2017-10-11 16:25:59</td>
<td>19.721</td>
</tr>
<tr>
<td>15.5613144,32.5440158</td>
<td>2017-10-11 16:26:59</td>
<td>12.136</td>
</tr>
<tr>
<td>15.5610904,32.5423301</td>
<td>2017-10-11 16:27:59</td>
<td>19.721</td>
</tr>
<tr>
<td>15.561153,32.542096</td>
<td>2017-10-11 16:28:59</td>
<td>16.687</td>
</tr>
<tr>
<td>15.5610926,32.5424614</td>
<td>2017-10-12 05:20:00</td>
<td>15.17</td>
</tr>
<tr>
<td>15.6104467,32.5367329</td>
<td>2017-10-12 05:39:04</td>
<td>18.204</td>
</tr>
<tr>
<td>15.5386616,32.5756383</td>
<td>2017-10-11 15:31:00</td>
<td>20.0</td>
</tr>
<tr>
<td>15.6089883,32.54256</td>
<td>2017-10-11 10:56:31</td>
<td>13.5</td>
</tr>
<tr>
<td>15.6100433,32.54035</td>
<td>2017-10-11 10:57:41</td>
<td>10.9</td>
</tr>
</tbody>
</table>

Figure 4.2 Sample of locations data
4.2.2 Signals Data

All the signals data are displayed in a tabular form for ease of read and review. Figure [4.3] shows a sample of the collected signals data.

<table>
<thead>
<tr>
<th>Signal Strength</th>
<th>Registered</th>
<th>Lac</th>
<th>mcc</th>
<th>Network Type</th>
<th>Operator</th>
<th>CellInfo</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>-95</td>
<td>true</td>
<td>12004</td>
<td>634</td>
<td>Unknown</td>
<td>Zain SDN</td>
<td>12071</td>
<td>92</td>
</tr>
<tr>
<td>-93</td>
<td>true</td>
<td>12004</td>
<td>634</td>
<td>Unknown</td>
<td>Zain SDN</td>
<td>12071</td>
<td>93</td>
</tr>
<tr>
<td>-91</td>
<td>false</td>
<td>11006</td>
<td>none</td>
<td>GPRS</td>
<td></td>
<td>11151</td>
<td>93</td>
</tr>
<tr>
<td>-97</td>
<td>false</td>
<td>12004</td>
<td>none</td>
<td>GPRS</td>
<td></td>
<td>12072</td>
<td>93</td>
</tr>
<tr>
<td>-99</td>
<td>false</td>
<td>13001</td>
<td>none</td>
<td>GPRS</td>
<td></td>
<td>13432</td>
<td>93</td>
</tr>
<tr>
<td>-101</td>
<td>false</td>
<td>11006</td>
<td>none</td>
<td>GPRS</td>
<td></td>
<td>11021</td>
<td>93</td>
</tr>
<tr>
<td>-97</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265788</td>
<td>94</td>
</tr>
<tr>
<td>-65</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265785</td>
<td>95</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265788</td>
<td>96</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265788</td>
<td>97</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265788</td>
<td>98</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265788</td>
<td>99</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>12005</td>
<td>634</td>
<td>GPRS</td>
<td>Zain SDN</td>
<td>27381</td>
<td>100</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2060</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1266517</td>
<td>101</td>
</tr>
<tr>
<td>-71</td>
<td>true</td>
<td>2060</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1266517</td>
<td>102</td>
</tr>
<tr>
<td>-97</td>
<td>true</td>
<td>2060</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265520</td>
<td>103</td>
</tr>
<tr>
<td>-89</td>
<td>true</td>
<td>2060</td>
<td>634</td>
<td>HSPAP</td>
<td>Zain SDN</td>
<td>1265520</td>
<td>104</td>
</tr>
<tr>
<td>-61</td>
<td>true</td>
<td>2071</td>
<td>634</td>
<td>UMTS</td>
<td>Zain SDN</td>
<td>1265785</td>
<td>105</td>
</tr>
</tbody>
</table>

Figure 4.3 Sample of the signals data

4.2.3 Map

All the locations data are displayed in the form of points on a google map, with the information about the signal strength, location accuracy, and time displayed as shown in figure [4.4] and figure [4.5].
Figure 4.4 Google map used to display the data

Figure 4.5 Information of a point displayed on the google map

4.3 Conclusion

4.3.1 Locations Data

Observed from figure [4.2] the tabular form of the location data and its graphical representation on the google maps in figure [4.4], the accuracy of each data reading -GPS
location- varies in terms of meters from the actual point it was taken from as shown in figure [4.6].

The cause of this variation is due to:

- The difference in the GPS module installed inside each mobile device
- The reflection of the incoming GPS signal from the satellites due to atmospheric reasons, or large objects as buildings.

![Geo. Location Accuracy](image)

**Figure 4.6 The variation of the geo. locations accuracy**

### 4.3.2 Signals Data

The system collected more than a 1,600-signal readings information, which varies in many different parameters due to different reasons.

In figure [4.3], some of the signal data has an operator value and a registered value of “**true**” while the others has empty or “**false**” values. The cause is the first type of data readings is coming from the sim card of each mobile phone, while the second type of data is coming from the neighboring cells. The system was able to receive data and read the information of many neighboring cells with some data redundancy due to using multiple mobile phones as shown in figure [4.7].
Figure 4.7 Number of neighboring cells, which the system was able to connect with

Observed from all the signal readings that the GPRS network type is the most common network type, while the UMTS (3G) comes next as shown in figure [4.8].

Figure 4.8 Types of the different networks
The color of the points on the map view on figure [4.4] represents the signal strength:

- **Green**: Represents a good signal strength from the range of \( x \geq -90, x \equiv signal\ strength. \)
- **Yellow**: Represents a fair signal from the range of \(-100 < x < -80, x \equiv signal\ strength. \)
- **Red**: represents a bad signal from the range of \( x \leq -100, x \equiv signal\ strength. \)

The signal strength varies widely with a high concentration in the range of \(-95.5 \text{ to } -85\), as shown in figure [4.9], while in figure [4.10] the distribution of the variation can be observed. One of the following might cause this variation:

- **Weather conditions**: Like humidity and barometric pressure, they can influence that distance ANY signal can travel.
- **Radio interference**: Like having other signals transferring on the same frequency.
- **Cell tower proximity**
- **Phone software and hardware**: The optimization of the software and the effectiveness of the available antenna.
- **Switching from one cell tower to another**
- **The number of mobile phones connected to the same tower**

![Signal Strength Ranges](image)

**Figure 4.9 Signal strength of the collected data**
Figure 4.10 Signal strength distribution
CHAPTER 5: CONCLUSIONS AND FUTURE WORK

In this chapter, the state of the project, in the light of its aim and objectives, is investigated and possible future work directions are discussed.

5.1 Conclusion

The aim of the project was to help the MNOs to find a suitable cheap and efficient method to collect raw data from their networks, which will be analyzed later in order to evaluate the performance, QoS, and QoE provided by their networks.

A software platform was introduced to solve the problem in hand. First, a mobile application was developed to collect the raw data from the cellular network using a mobile application, which then sends the data to a web application. The web application receives the data and display it in a tabular form for ease of review and access, and in a graphical interface using a google map in the form of colored points representing the signal strength.

The results of the system as a whole was promising, with a huge amount of data collected. Taking in mind that several mobile phones were used to collect the data while their users were moving from one point to another without any extra effort required.

Therefore, the proposed platform met its objectives:

- It collected a huge amount of the required raw data with minimum effort and low cost.
- It is cheap because there is no need to buy any hardware to collect the data; it simply uses the mobile phones available hardware, and there is no need to hire engineers to go out in the field and collect the data; it simply relies on the movement of the user of the phone.
- It is not time consuming and the data is available whenever the MNOs need it.

5.2 Future Work

From the difficulties faced during the developing of the software platform and the results obtained, it is possible to identify the following future work paths:
• Modify the user interface of the mobile application to make it more user friendly and easier to use.
• Increase the type of parameters, which are collected using the mobile application such as the block calls and the drop calls.
• Add more tests to the mobile application to measure the upload and download speed of the network packets.
• Add a region based representation to represent the data graphically on the web application on the google map.
• Add a statistics page on the web application.
• Add a report-generating tool to the web application.
• Increase the security measures implemented in the whole platform.
• Embed the application within the mobile network operators’ sim cards to be installed automatically within the mobile phones of the clients.
REFERENCES


References


APPENDIX A

In this part, an important snippet of a JavaScript code from the web application code of the map view is presented.

```javascript
function myMap() {
    var myCenter = new google.maps.LatLng(15.610736, 32.538506);
    var mapProp = { center: myCenter, zoom: 15, scrollwheel: false, draggable: true, mapTypeId: google.maps.MapTypeId.ROADMAP }; 
    var infoWindow = new google.maps.InfoWindow();
    var map = new google.maps.Map(document.getElementById("googleMap"), mapProp);
    var markers = @Html.Raw(ViewBag.Markers);
    var len = markers.length;

    for (i = 0; i < markers.length; i++) {
        var data = markers[i]
        var myLatlng = new google.maps.LatLng(data.lat, data.lang);
        var marker = new google.maps.Marker(
            { position: myLatlng,
            icon: {
                path: google.maps.SymbolPath.CIRCLE,
                scale: 3,
                strokeColor: data.color,
                fillColor: data.color
            }
            title: data.time
        });

        marker.setMap(map);
        (function (marker, data) {
```

A1
google.maps.event.addListener(marker, "click", function (e) {
    var content = 'Location accuracy: ' + data.accuracy +
                   '<br>' + 'Signal strength: ' + data.strength +
                   '<br>' + 'Operator: ' + data.operator + '<br>' +
                   'Time: ' + data.time

    infoWindow.setContent(content);
    infoWindow.open(map, marker);
});

})(marker, data);
APPENDIX B

In this part, important snippets of the android application code regarding the gathering of the data is presented.

//get network parameters
tel = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
int networkType = tel.getNetworkType();
if(networkType==TelephonyManager.NETWORK_TYPE_UNKNOWN){
    networkType=16;
}
networkTypes.add(0,networkType+" ");
operatorsNames.add(0,tel.getSimOperatorName());

int phoneType = tel.getPhoneType();
if(phoneType==TelephonyManager.PHONE_TYPE_GSM) {

    GsmCellLocation gsmLoc = (GsmCellLocation) tel.getCellLocation();
    if(gsmLoc==null){
        Log.e("gsmLoc", "null");
        cellIds.add(0, "-1");
        Lacs.add(0, "-1");
    } else {
        cellIds.add(0, gsmLoc.getCid() + ");
        Lacs.add(0, gsmLoc.getLac() + ");
        Log.e("gmscid", gsmLoc.getCid() + "");
        Log.e("cidtry", tel.getSimOperator() + " + tel.getSubscriberId());
        Log.e("mccnetwork", tel.getNetworkOperator());
        Log.e("mcceconfig", getResources().getConfiguration().mcc + " + getResources().getConfiguration().mnc);";
        Log.e("mccsim", tel.getSimOperator());
    }
    if(tel.getNetworkOperator().length()<3){
        Mccs.add(0, ");
    } else {
        Mccs.add(0, tel.getNetworkOperator().substring(0, 3));
    }
    servingCellStatus.add(0, "true");
}
else if (phoneType == TelephonyManager.PHONE_TYPE_CDMA) {
    CdmaCellLocation cdmaLoc = (CdmaCellLocation) tel.getCellLocation();
    if(cdmaLoc==null){
        cellIds.add(0, "-1");
        Lacs.add(0, "-1");
    } else {
        cellIds.add(0, cdmaLoc.getBaseStationId() + "");
        Lacs.add(0, cdmaLoc.getNetworkId() + "");
        Log.e("cdmacid", cdmaLoc.getBaseStationId() + "");
        Log.e("cmdalac", cdmaLoc.getNetworkId() + "");
    }
    if(tel.getNetworkOperator().length()<3){
        Mccs.add(0, ");
    } else {

}}
Mccs.add(0, tel.getNetworkOperator().substring(0, 3));
} else if(phoneType ==TelephonyManager.PHONE_TYPE_NONE){
cellIds.add(0,"-1");
Lacs.add(0,"-1");
if(tel.getNetworkOperator().length()<3){
    Mccs.add(0,"-1");
} else {
    Mccs.add(0, tel.getNetworkOperator().substring(0, 3));
}
servingCellStatus.add(0, "true");
} else {
cellIds.add(0,"-1");
Lacs.add(0,"-1");
if(tel.getNetworkOperator().length()<3){
    Mccs.add(0,"-1");
} else {
    Mccs.add(0, tel.getNetworkOperator().substring(0, 3));
}
servingCellStatus.add(0, "true");
}

phone = new myPhoneStateListener(currentLocationMethod,TimeMethod);
tel.listen(phone, PhoneStateListener.LISTEN_SIGNAL_STRENGTHS);

///get Signal strength
class myPhoneStateListener extends PhoneStateListener {
    Location currentLocationMethod;
    String TimeMethod;

    myPhoneStateListener(Location currentLocationMethod,String TimeMethod){
        this.currentLocationMethod = currentLocationMethod;
        this.TimeMethod = TimeMethod;
    }

    @Override
    public void onSignalStrengthsChanged(SignalStrength signalStrength) {
        super.onSignalStrengthsChanged(signalStrength);
        try {
            int Signal = signalStrength.getGsmSignalStrength();
            tel.listen(phone, PhoneStateListener.LISTEN_NONE);
            Signal = -113 + 2 * Signal;
            Log.d("signalstrengthvalue", Signal + "");
            signalStrengths.add(0, Signal + "");
            sendBroadcastMessage(Signal+"",cellIds.get(0),Lacs.get(0),Mccs.get(0),operatorsNames.get(0),TimeMethod);
            saveLocationAndSignal(currentLocationMethod, TimeMethod);
        } catch (Exception e){
            e.printStackTrace();
        }
    }
}

///get Neighboring Cells Info:
TelephonyManager telephonyManager = (TelephonyManager) getBaseContext().getSystemService(Context.TELEPHONY_SERVICE);
GsmCellLocation cl = (GsmCellLocation) telephonyManager.getCellLocation();

List<NeighboringCellInfo> networks = telephonyManager.getNeighboringCellInfo();
Log.d("neighbor cells size:", networks.size() + "");
for (int i = 0; i < networks.size(); i++) {
    NeighboringCellInfo nci = (NeighboringCellInfo) networks.get(i);
        int rss = nci.getRssi();
    if(rss>0) {
            rss = -113 + 2 * rss;
    }
    signalStrengths.add(rss+"");
    cellIds.add(nci.getCid()+"");
    Lacs.add(nci.getLac()+"");
    Mccs.add("none");
    servingCellStatus.add("false");
    Log.d("neighbor", rss + "," + nci.getCid() + "," + nci.getLac());
    Log.d("unknowCid",nci.UNKNOWN_CID+" "+nci.UNKNOWN_RSSI);
    int networkType = nci.getNetworkType();
    if(networkType==TelephonyManager.NETWORK_TYPE_UNKNOWN){
        networkType=16;
    }
    networkTypes.add(networkType+"");
    operatorsNames.add("none");
APPENDIX C

In this part, snippets of the mobile application regarding the upload of data is presented.

```java
// Upload locations Logs
String urlDestroySession = "http://192.168.43.106:53533/api/locationsapi";
JSONArray allReads = new JSONArray();
for(int i=0;i<locationsValues.size();i++) {

    JSONObject jsonParam = new JSONObject();
    try {
        jsonParam.put("locationText", locationsValues.get(i));
        jsonParam.put("time", locationsTimes.get(i));
        jsonParam.put("accuracy", locationsAccuracy.get(i));
    } catch (JSONException e) {
        e.printStackTrace();
    }
    allReads.put(jsonParam);
}
Log.e("all_locs_reads", allReads.toString());

JsonArrayRequest jsonObjectLoginRequest = new JsonArrayRequest(Request.Method.POST, urlDestroySession, allReads, new Response.Listener<JSONArray>() {
    @Override
    public void onResponse(JSONArray response) {
        Log.e("logs_response", response.toString());
        try {
            for(int i=0;i<response.length();i++) {
                int locationIdServer = response.getJSONObject(i).getInt("id");
                Log.e("location_id_server", locationIdServer++);
                datasource.updateLocationIdServer(locationsIds.get(i), locationIdServer++);
                datasource.updateLocationsLogsSent(locationsIds.get(i));
            }
            uploadSignalssRest();
        } catch (Exception e) {
            e.printStackTrace();
            Log.d("upload_errore", e.getMessage());
            if(progressDialog.isShowing()) {
                progressDialog.dismiss();
            }
        }
        Log.e("end_of_upload", "yes");
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e("error_res", error.toString());
        error.printStackTrace();
    }
});
```
if (progressDialog.isShowing()) {
    progressDialog.dismiss();
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    HashMap<String, String> headers = new HashMap<String, String>();
    return headers;
}

jsonObjectLoginRequest.setRetryPolicy(new DefaultRetryPolicy(0, DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
                DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
mainQueue.add(jsonObjectLoginRequest);

///Upload Signals Logs
String urlDestroySession = "http://192.168.43.106:53533/api/signalsapi";

JSONArray allReads = new JSONArray();
for (int i=0;i<signalsLocationServerIds.size();i++) {

    JSONObject jsonParam = new JSONObject();
    try {
        jsonParam.put("idlocation", Integer.parseInt(signalsLocationServerIds.get(i)));
        jsonParam.put("signalstrength", signalsSignalStrengths.get(i));
        jsonParam.put("cellinfo", signlasCellIds.get(i));
        jsonParam.put("registered", signalsRegistered.get(i));
        jsonParam.put("lac", signlasLacs.get(i));
        jsonParam.put("mcc", signlasMccs.get(i));
        if (!signalsOperatorsNames.get(i).equals("none")) {
            jsonParam.put("operator", signalsOperatorsNames.get(i));
        }
    }
    int networkType = Integer.parseInt(signalsNetworkTypes.get(i));
    String networkTypeStr = "";
    switch (networkType) {
        case 1:
            networkTypeStr = "GPRS";
            break;
        case 2:
            networkTypeStr = "EDGE";
            break;
        case 3:
            networkTypeStr = "UMTS";
            break;
        case 4:
            networkTypeStr = "CDMA";
            break;
        case 5:
            networkTypeStr = "EVD0_0";
            break;
        case 6:
            networkTypeStr = "EVD0_A";
            break;
        case 8:
            networkTypeStr = "HSDPA";
            break;
        case 9:
            networkTypeStr = "HSuPA";
            break;
    }

    allReads.put(jsonParam);
case 10:
    networkTypeStr = "HSPA";
    break;
case 11:
    networkTypeStr = "IDEN";
    break;
case 12:
    networkTypeStr = "EVD0_B";
    break;
case 13:
    networkTypeStr = "LTE";
    break;
case 14:
    networkTypeStr = "EHRPD";
    break;
case 15:
    networkTypeStr = "HSPAP";
    break;
case 16:
    networkTypeStr = "UnKnown";
    break;
default:
    networkTypeStr = "other";
    break;
}
jsonParam.put("networktype", networkTypeStr);
} catch (JSONException e) {
    e.printStackTrace();
}
allReads.put(jsonParam);
}

Log.e("signals_read", allReads.toString());

JsonArrayRequest jsonObjectLoginRequest = new JsonArrayRequest(Request.Method.POST, urlDestroySession, allReads, new Response.Listener<JSONArray>() {
    @Override
    public void onResponse(JSONArray response) {
        Log.e("signals_response", response.toString());
        try {
            for (int i = 0; i < signalsLocationsIds.size(); i++) {
                datasource.updateSignalsLogsSent(signalsLocationsIds.get(i));
            }
        } catch (Exception e) {
            e.printStackTrace();
            if (progressDialog.isShowing())
                progressDialog.dismiss();
        }
    }

    @Override
    public void onErrorResponse(VolleyError error) {
        Log.d("upload_error", error.getMessage());
    }

    Log.e("end_of_upload_signals", "yes");


`new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e("error_res", error.toString());
        error.printStackTrace();

        if (progressDialog.isShowing()) {
            progressDialog.dismiss();
        }
    }
}

@Override
public Map<String, String> getHeaders() throws AuthFailureError {
    HashMap<String, String> headers = new HashMap<String, String>();
    return headers;
}

};

jsonObjectLoginRequest.setRetryPolicy(new DefaultRetryPolicy(0, DefaultRetryPolicy.DEFAULT_MAX_RETRIES, DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
mainQueue.add(jsonObjectLoginRequest);