JOMO KENYATTA UNIVERSITYOF AGRICULTURE AND TECHNOLOGY

SCHOOL OF ELECTRICAL, ELECTRONIC AND INFORMATION ENGINEERING

DEPARTMENT OF ELECTRICAL & ELECTRONIC ENGINEERING

FINAL YEAR PROJECT REPORT

PROJECT TITLE: AUTOMATIC POWER OUTAGE NOTIFIER SYSTEM.

Submitted by:

NAIRUKO MERCY – EN271-3878/2015

DAVIES MOMANYI – ENE211-0233/2016

Project Supervisor: MR. LINUS ALOO

A Final Year Project Report submitted to the Department of Electrical and Electronic Engineering in partial fulfillment of the requirements for the award of a Bachelor of Science Degree in Electrical and Electronic Engineering.

DECLARATION

This project report is our original work, except where due acknowledgement is made in the text, and to the best of our knowledge has not been previously submitted to Jomo Kenyatta University of Agriculture and Technology or any other institution for the award of a degree or diploma.

NAME: NAIRUKO MERCY

REG. No. EN271-3878/2015

SIGNATURE…… DATE ……

NAME: DAVIES NYAMWANGE MOMANYI

REG. No. ENE211-0233/2016

SIGNATURE…… DATE ……

TITLE OF PROJECT: AUTOMATIC POWER OUTAGE NOTIFIER SYSTEM

SUPERVISOR CONFIRMATION:

This project report has been submitted to the Department of Electrical and Electronic Engineering, Jomo Kenyatta University of Agriculture and Technology, with my approval as the supervisor:

NAME: MR. LINUS ALOO

SIGNATURE…… DATE ……

Table of Contents

List of Abbreviations

AC	Alternating Current
AI	Artificial Intelligence
ADC	Analogue to Digital Converter
ADT	Android Development Tools
API	Application Interface
CT	Current Transformer
DC	Direct Current
DNS	Domain Name System
GPS	Global Positioning System
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IED	Intelligent Electronic Devices
IR	Infrared
LoRaWAN	Long Range Wide Area Network
LPWAN	Low Power Wide Area Network
OS	Operating System
PHP	Hypertext Preprocessor
RAM	Random Access Memory
RISC	Reduced Instruction Set Computer
SDK	Software Development Kit
SoC	System-on-a-Chip
SCADA	Supervisory Control and Data Acquisition
SMS	Short Message Service
WIFI	Wireless Fidelity
XML	Extensible Markup Language

List of Figures

List of Tables

ABSTRACT

Electricity is and has proven to be the most important innovation in the past 150 years. However, nothing is perfect and sometimes there is a stop or sudden of supply of electricity, power outage. Power Outages can occur due to various reasons mostly a fault in the system, transformer breakdown or even harsh weather. In Kenya, the response time of Kenya power emergency or operations and maintenance team is usually dependent on humans. i.e, the power has to be reported and correct details of the location recorded before a team is assigned to fix the situation and restore power. This delay often leads to various types of loses mainly in monetary value and even loss of lives in some areas where untrained personnel try to fix the problems.

This project improves the response time of Kenya power to breakdowns that have led to power outages by notifying them immediately of the occurrence and providing the location of the breakdown on a map service. The solution is a device that tracks if there's a power on a specific line and updates a distribution company and it's affected consumers through an app and website whenever there is a power outage caused by a fault on the line.

The challenge was solved by coming up with an automatic power outage notifier system that informs the distribution company whenever there's a power outage. The system comprises of; Current Transformers (CTs) which detect when current is flowing and not flowing on the line, whenever it's not flowing a notification is sent to the distribution company about the same through a system that comprises of a Microcontroller, WIFI modules and location sensors that notifies on the exact location where a power outage has occurred.

CHAPTER 1 : INTRODUCTION

1.1 Background Study

Electricity is simply defined as the flow of charge. In the past 200 years it has proven to be the most important innovation in the world. It's also the most important tool of improving the living standards and industrialization of developing nations. In Kenya, most consumers get electricity from distribution and transmission lines connected to the national grid. The distribution of electricity and maintenance of the distribution lines in Kenya is done by Kenya Power and Lighting Company. Unfortunately, these lines often have various breakdowns that ultimately lead to power outages. A power outage is a short- or long-term loss of electricity in a given area or section of distribution lines. According to the world bank the rate of access to electricity in Kenya has risen from 32 % in 2013 to 73.42% in 2017. As a result of this expansion the number of power outages occurring in Kenya has also increased.

For a power outage to be resolved and power restored to consumers, it is first reported to Kenya Power through calls, social media or even an email. Kenya power then inquires on the location of the power outage and meter number. It then assigns a team to go to the location, find the fault and try to resolve it in a short time. This process is long and often leads to a slow response time by the Kenya Power team assigned. The delay caused by the response time of Kenya Power prolongs the power outage and has numerous negative effects not only on both commercial and domestic consumers but also Kenya Power itself. The most impactful effect is loss of revenue for all parties involved.

In this project we improved the response time on power outages by designing and implementing an electronic device that would immediately notifies the distribution company when an outage occurs in a specific location.

1.2 Problem Statement

There is no perfect national grid in the whole world. In all distribution networks power outages are expected. Through various means one can reduce the number of power outages but not prevent

them fully. However, a slow response time to a power outage causing fault prolongs and worsens the effects of power outages. Effects of the power outages include loss of revenue due to a standstill and slowing down of processes in industries, and hindering access to certain needs and services that require electric power to domestic users.

Although there are options of notifying Kenya power about these power outages such as calling their service lines, sending them emails and raising concerns through Facebook or twitter, the response time is still slow and has its own shortcomings. Some of these shortcomings are flooded emails and SMS, busy lines, people being abusive on social media and generally too much human interaction. Also, most people do not have full information on the power lines or cause of the power outage.

Therefore, to improve the response time the shortcomings of the current options should be overcome. This was achieved by a means that: provides immediate notification whenever a power outage occurs, provides information on the location and distribution line affected by the power outage, records and stores power outage time and duration, makes it easier to assign teams and provides real time customer feedback through an app or web service.

1.3 Justification

The system is made up of hardware and software. The hardware part is made up of a microcontroller, current transformer and a Global Positioning System sensor. The software part consists of a website and an app. The electronic device is able to immediately tell if a power outage has occurred on a distribution line. If so, it immediately notifies the power distribution company of the power outage through the website and an app. It also provides the location of the distribution line. Locations affected by the power outage are also illustrated on a map on both platforms. The customers or persons affected by the outage are notified and receive various feedback on the current state of the power outage i.e. how long it is expected to last and whether a team has been assigned.

Various gaps and shortcomings existing in the current options are therefore filled and overcome by this system. First of all, in the current options reporting of a power outage is a tedious and long process. This is because of waiting times due to busy lines. This inadvertently contributes to the slow response time to power outages. In this system the distribution company immediately notifies of the power outage as soon as it happens. Currently it's also hard to get accurate information from

the person reporting the outage. This is due to some locations and areas sharing common names. Some locations also have so many distribution lines hence it takes the Kenya Power team some time before identifying the faulty line. Through the map service and GPS module, the system ensures that accurate information on the location of the power outage is shared to the assigned team resulting in a fast response time

Another advantage is providing the customer with feedback and information on the current status of the power outage. With the current system and options customers are hardly informed on the current status of the power outage. The result is multiple customers calling or emailing about the same power outage which ends up in busy lines for Kenya Power Call Centre and too much complaints or reports on their social media handles. Not only does the system offer a platform through the app that enables Kenya power to inform customers on the current status of the power outage but also helps in addressing concerns raised on specific power outages

1.4 Objectives

1.4.1 Main Objective

To design, implement and test an automatic power outage notifier system.

1.4.2 Specific objectives

1. To come up with an electronic device that detects when a power outage occurs and when power is restored on a distribution line
2. To connect the electronic device wirelessly to a cloud database
3. To create a cloud database that will store and record power outage information provided by the electronic device
4. To display locations affected with power outages through a map service on a website and provide exact locations of power outages
5. To create a mobile application that will enables feedback between consumers and the power distribution company

CHAPTER 2 : LITERATURE REVIEW

2.1 Introduction

A power outage, commonly called a blackout or power failure is the loss of the electrical power network supply to an end user. These are commonly caused by faults at power stations, damage to electric transmission lines, substations or other parts of the distribution system, a short circuit, cascading failure, fuse or circuit breaker operation.

In Kenya most of these cases are reported as no supply since most people do not know the causes. The chart in figure 2.1 shows some of these causes of power outages reported as at 17th May 2021.

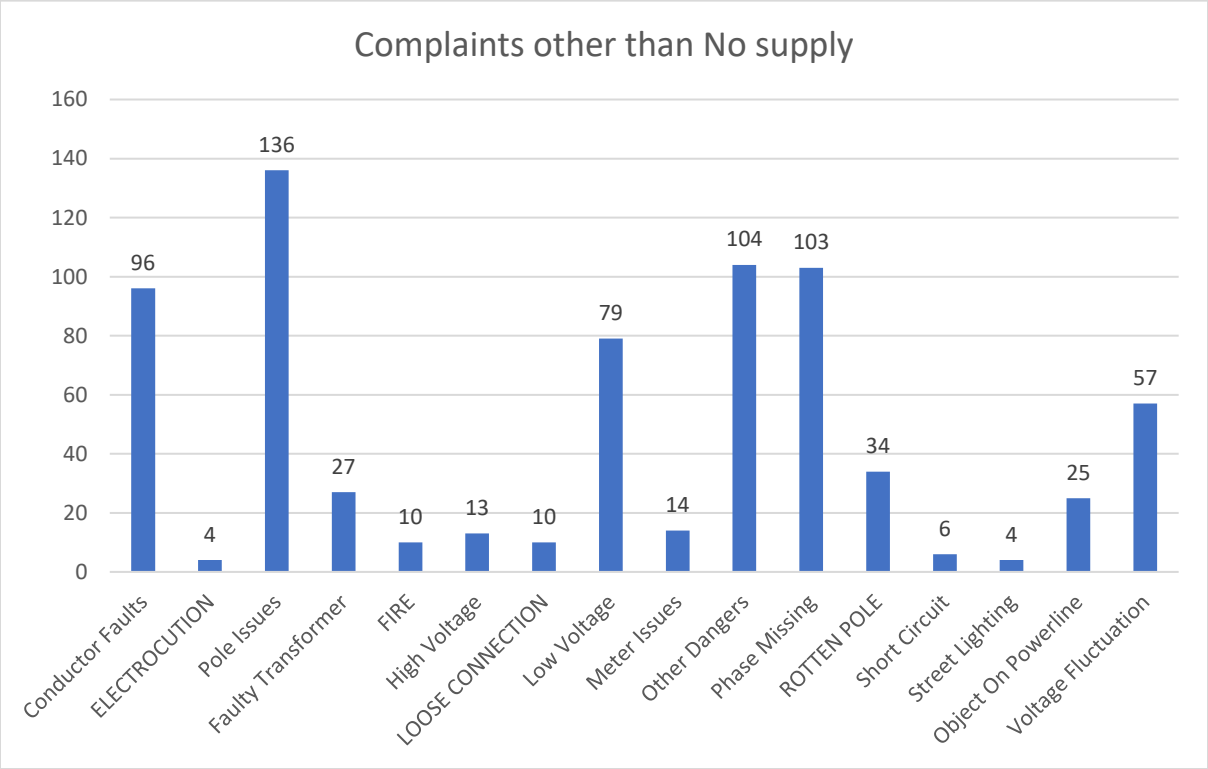


Figure 2.1 Complaints Other Than No Supply Chart

Power outages are particularly critical at sites where the environment and public safety are at risk. Institutions such as hospitals, sewage treatment plants, and mines will usually have backup power sources such as standby generators, which will automatically start up when an outage occurs. Other critical systems, such as telecommunication, are also required to have emergency power. The battery room of a telephone exchange usually has arrays of lead–acid batteries for backup and also a socket for connecting a generator during extended periods of outage.

These cases are mostly reported through social media platforms such as Facebook and Twitter. The causes of these power outages cannot be avoided but the response time can be improved.

2.1.1 Effects of Power Outages

Accessibility to reliable, quality and efficient supply of electricity is regarded as a key conduit for economic growth and development across the world [1]. Power outages therefore are a direct hindrance to the growth and development of Kenya and inadvertently power outages have various grave effects.

The effects of power outages on commercial consumers has been greatly investigated and documented and it can be seen that inadequacies in electrical power sector in Kenya constrains economic growth and development [2]. There are various challenges that affect the power sector in Kenya but particularly singles one as the most critical, power outages. Data from the world bank shows and states that most enterprises in Africa encounter regular power outages and that these power outages result in a negative impact on foreign ownerships of businesses in the continent. In addition, linking power outages with inefficient production, negative effects on firm performance and revenue generation. power outages in Kenya therefore have an adverse effect on the performance of firms.

Not only does power outage have a huge impact on economic growth but also on living standards. This is through the effects it has on domestic users, it lowers their living standards and causes inconveniences within the households [3].

2.2 Existing Systems

2.2.1 Using Social Media to Report Outages

Using social media as a means of reporting outages is the most cost effective and reliable way. Twitter is the most widely used social media platform as shown in fig. 2.2.

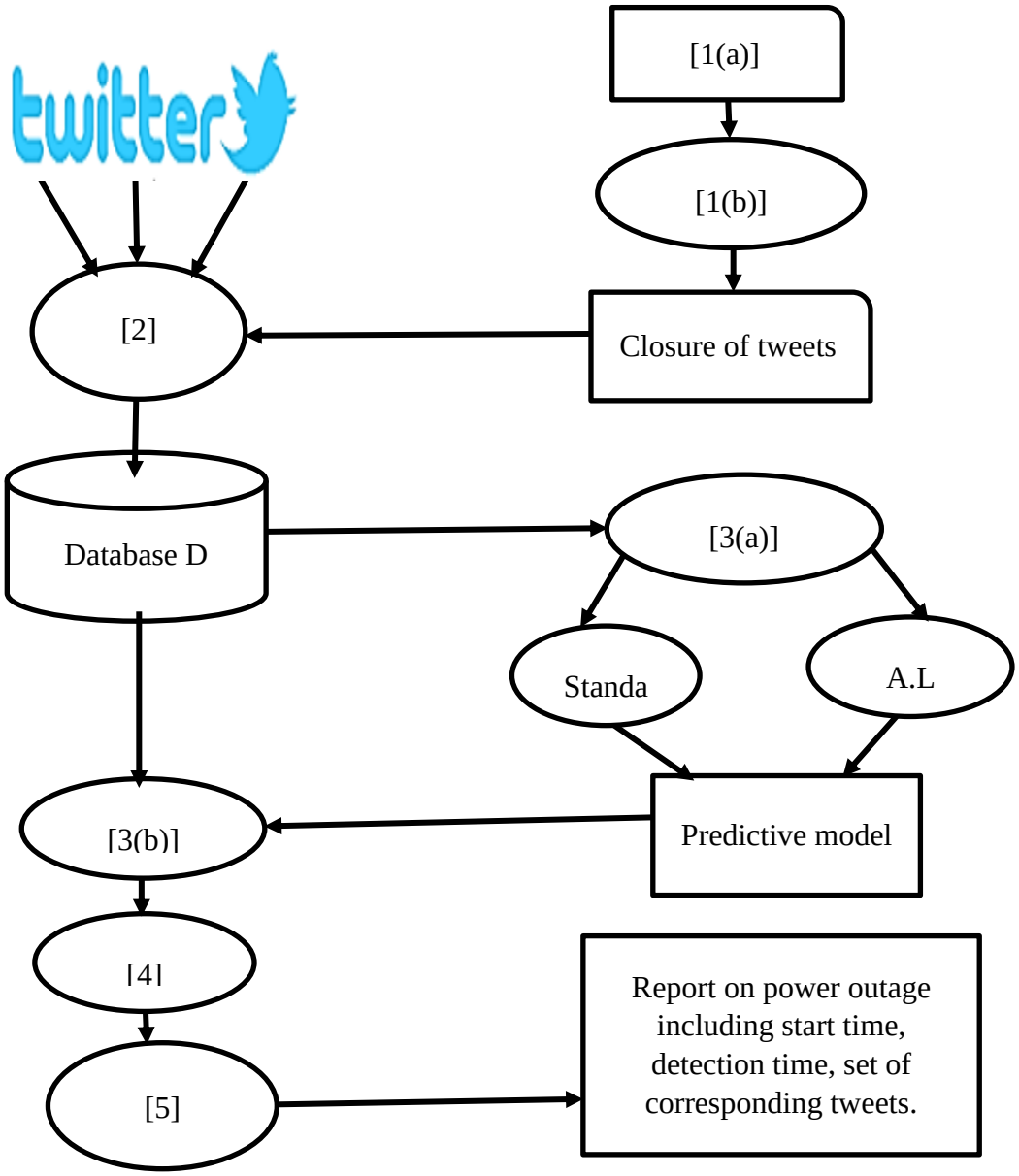


Figure 2.2 Power Outage Detection Using Social Media

From the diagram in Figure 2.2;

1.a) Specify the set of core key concepts K pertaining to power outages and (b) compute its closure C.

2. Collect the tweets from the Twitter stream containing at least one key concept from set C and store them in database D.

3) (a) Build a predictive model that identifies if tweet x from D posted at time t refers to a power outage that occurred around time t in the region where tweet x originated (as determined by the tweet location);

(b) classify the tweets from database D into Class 1specifying those tweets that include their GPS data or home location and that were posted by the individuals

who witnessed real power outages and immediately tweeted about them, and all other Class 0 tweets.

4) Identify the bursts of Class 1 tweets in the stream generated in Step 3 (that refer to currently occurring power outages).

5) Extract the aspects of the power outages for each burst of Class 1 tweets, these aspects pertaining to the possible reasons and the weather conditions of the outage

Social media users are used as sensors to obtain information on power outages. The ubiquity of smart phones and social networks has given rise to an entirely new class of sensor, the human "social sensor". Indeed, any individual with a networked device and a social media account has the potential to become a "social sensor node" [4]. Twitter is amongst the largest social media networks and has an average of 63 million users in the USA alone. During a power outage there is a sudden burst of tweets mentioning the power outage in the regions affected. Hence, collecting the data from these tweets leads to effective power outage detection with specific details such as location or source of fault that would lead to a faster response by the power company. Not only do the tweets help to identify the power outage locations but they also help to identify a possible fault that caused the power outage and the weather condition at the time of the outage [4]. The advantage of this method is that it's cheap as the distribution company requires less capital.

2.2.2 Geo Tagged Power Outages Using Social Media

Other studies continue to suggest but also improve on use of social media to report power outages. Social media data is used to gain insight in quickening the resolution process of a power outage [5]. Similar to the first case study Twitter is used as the source for power outage data. Kenya Power and Lightning System have a database with all meter numbers geo-referenced. Therefore, when a customer tweets about a power outage and includes their meter number the location of the power outage is automatically be geo-referenced and mapped out. This is achieved by use of various Application Interfaces. First, tweepy is used for authentication of consumer keys and access tokens. Restful API (application interface) enables use and access of the twitter data. The twitter stream listeners enable the streaming of data from twitter that meet certain criteria. The designed system filters only relevant tweets with location aspect and power outage report, which are later geocoded and displayed in a map [5]. The advantage of this approach is that the exact locations of power outages are determined.

2.2.3 Using Small Scale SCADA To Detect Outages

There also exist other methods of outage detection which do not depend on customer input. One such method or system is using a small-scale Supervisory Control and Data Acquisition (SCADA) system whose architecture is as shown in figure 2.3.

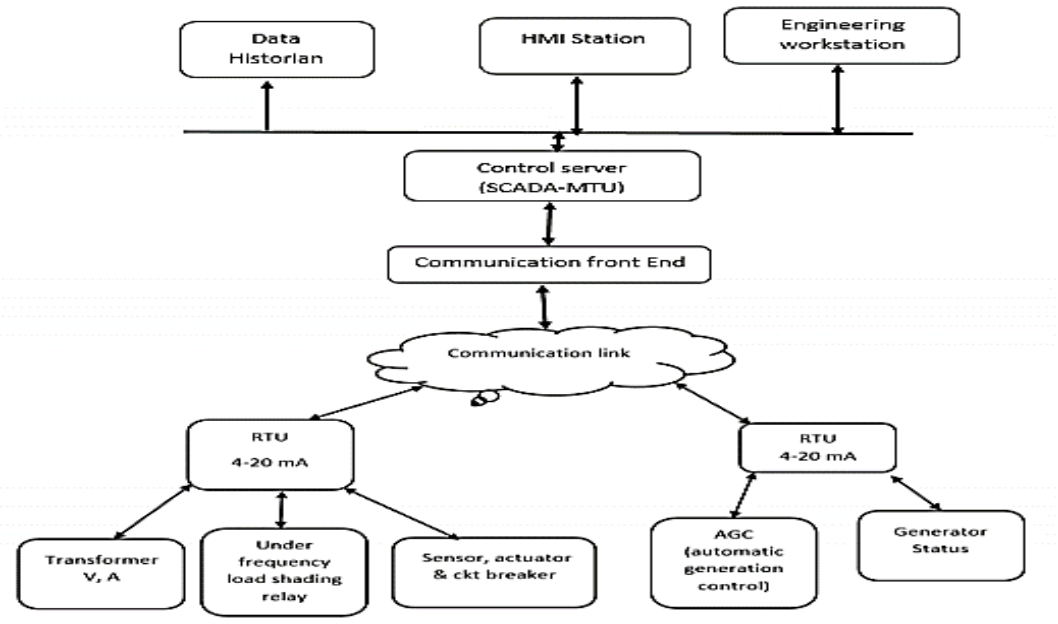


Figure 2.3 Generalized Simple SCADA System

Supervisory control is a term for high scale control of many controllers. Data is also acquired from various sensors and analyzed in real-time. SCADA systems have been and are normally used for monitoring high voltage transmission lines which mainly includes the national grid. However, in recent times some power distribution companies have implemented SCADA systems in the control and monitoring of their distribution lines [6]. Hence small utilities can benefit from SCADA functionality, requirements and technical considerations. Any SCADA implementation must be customized to address a utility's needs. The key aspects of a SCADA system include Real time equipment status, Historical data trending and storage, Remote operation capability, System Security, Communications and ongoing technical support access.

The most important and critical of the requirements is communications capability. It is evident that SCADA systems have been improving safety and reliability and enhancing utility metrics. therefore, SCADA systems have enabled Large scale utilities to have and maintain adaptable, responsive and secure infrastructure by continuously monitoring measurement data. However, small scale utilities have resisted on using SCADA. This is because in the past SCADA systems needed to be regulated with control room settings and dedicated staff. Hence for small scale implementation to be highly adopted in distribution lines they have to be cost effective. The advantages of this system are; Real time monitoring and control of the distribution line systems. This enables the distribution company to make timely decisions like scheduling maintenances that might prevent power outages, very fast detection of power outages and historical data of various parts of a distribution system that may enable fast detection of the fault that caused the power outage.

2.3 Similar Projects

There are some projects that have used and utilized components that are part of the proposed methods. These projects successfully applied and used the components and achieved desirable results.

2.3.1 Automatic Power Meter

Meter reading done by Kenya power is mainly done by clerks. This method of meter reading is very inefficient and most often than not provides inconsistent and incorrect readings. Also, some of the clerks are bribed by the customers to record wrong readings to reduce the bill of the customer. The study therefore proposed a solution for automatic meter reading using an ATmega328p MCU [7]. This entailed an ACS712 current sensor that will be connected onto a power meter coupled to a LoRa gateway through a LPWAN for transmission of data to a cloud server for subsequent upload and analysis. [7] The project aids the development and research of the proposed method in various ways. This is because the backbone of this project is monitoring, collecting and analyzing data on load. Moreover, the author first sends and stores this information in a database remotely.

The key components of this project are:

a) IR arduino sensor - The IR Arduino Sensor is an electronic instrument used to sense certain characteristics of its surroundings either by emitting or detecting Infrared radiation. In a rather creative way, to use this sensor to sense light from the meter reader and using a data logger to convert the data to an energy consumption value.

b) Lora module – It consists of the ATmega328P Microcontroller, a low power CMOS 8-bit microcontroller based on AVR enhanced Reduced Instruction Set Computer (RISC) architecture. Makes up part of the communication hardware that is to enable machine to machine communication.

c) Power and load – contain embedded hardware that is to sense load from the meter in current and voltage.

d) LoRa Infrastructure – Contains Radio gateway that is the transceiver module. Long Range Radio Gateway that sits on a raspberry pi and sends data to the cloud server via internet.

One Outstanding feature that used in this project is LoraWAN. The automatic Meter reading prototype connects to the cloud through a long-range power wide area network via a long-range wide area network Gateway (LoRaWAN).

LoRaWAN is a networking protocol that has been designed to wirelessly connect battery operated devices to the internet through regional, national and worldwide networks. It targets important

internet of things requirements such as bi-directional communication, end to end security and localization services. LoRaWan uses less power compared to other long-range networks such as Wi-Fi and Bluetooth. A sensor using LoRaWan for communication can therefore last several years just on normal standard AAA batteries.

To incorporate LoRa Wan to his project the researcher used RN2483. The R2483 is a transceiver developed by Micro-Chip. It is a Low-Power long range technology. It provides a low power solution for long range wireless data transmission. It runs on LoRa and complies with the LoRaWAN class A protocol specifications. It integrates RF, a baseband controller, command Application Programming Interface (API) processor thus making it a complete long-range solution. This capability allows it to transmit data up to 15km in sub-urban environments and 10km in 38 dense cities.

2.3.2. Arduino Based Wireless Power Meter

The aim of the project was to give a homeowner a wireless non-invasive way of tracking their power usage [8]. The Arduino based wireless power meter is a non-invasive current meter for household power with a MATLAB interface. Current is measured using split core current transformers. This data is then transmitted over an 802.11b connection through the home's wireless router to the base station and MATLAB interface. The project aims to provide a clear picture of a home's current usage, and through this data provide an estimate of power consumption. The project also aims to identify which devices turn on and off by analysis of this current data. The project is a relevant source of insight for developing the proposed method as it contains monitoring of load using a current meter and transmitting the results through a wireless medium. the system requirements are:

- Accurately collect a house's total current consumption safely, and with a relatively fast update rate.
- Transmit the information back to a base station to be represented visually to a user.
- Identify when devices in the household turn on and off based on changes in the current data.

The first and third requirement are almost similar to one objective of the proposed method, only that the proposed method will try to accomplish the requirements on a much larger scale. The

fast update rate and monitoring when there's load or not are key requirements for the proposed method.

Similar to the previous project the chosen micro controller was an Arduino, arduino Duemilanove. The arduino Duemilanove is centered on an Atmega328p. An outstanding feature of this project is its method for current measurement. A split core current transducer was used as the current sensor. The sensor is clamped to the mains line without interrupting power flow. The output of the current transformer is an AC voltage proportional to the AC current flowing in the mains line. The peak voltage is obtained by the following equation (2.1).

$$V = \frac{I * R}{3100} \tag{2.1}$$

Where V is the rms voltage across the burden resistor, I is the rms current enclosed by the current transformer and R is the resistance of the burden resistor. The output range of the sensor is therefore determined by the value of the burden resistance. VAC output of the sensor could not interface with the micro controller due to limits of reverse voltage on the microcontroller's pins. The researcher therefore designed two rectifier circuits. A full wave rectifier using a diode bridge and a half wave rectifier. The diode bridge was paired with a capacitor and resistor to reduce the voltage ripple. However, due to rapid changes in current the rectifier had a forward voltage drop.

The second rectifier, a precision rectifier was therefore chosen. Moreover, it managed to solve the forward voltage drop challenge as the operational amplifier would correct any loss of voltage over the diodes.

For Communication the system used Wishield's open source implementation of UIP. Wishield was selected over other existing communication systems because UIP was ported and released as open source code. This made prototyping of the system faster and allowed the researcher to focus on measuring current and watts used.

For results the researcher states that the project was successful but it did not meet all the design goals. The update rate and measurement accuracy were both satisfactory but the ability for the base station to detect different devices turning on and off was not achieved. Visualization of the data was also a challenge.

2.4 Key Devices

2.4.1 Nodemcu Esp8266

The Nodemcu (Node Microcontroller Unit) as shown in fig. 2.4 is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WIFI), and even a modern operating system and SDK.

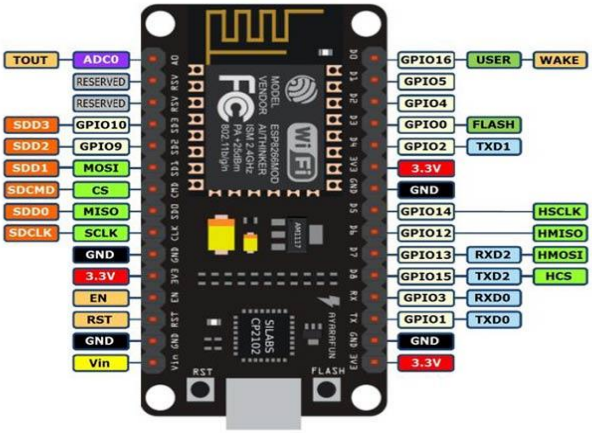


Figure 2.4 Nodemcu ESP8266

2.4.2 Current Transformer

To measure the current and hence load on the lines, a current transformer will be used. The current transformer that will be used is Model SCT-013-030. It is a noninvasive AC current sensor (100A max). It is manufactured by Beijing YaHuadechang Electronic Co.

It also has a burden resistor.

Figure 2.5 Current Transformer

2.4.3 Rectifier

As seen from Arduino Based Wireless Power Meter [8], a precision rectifier should be connected. this is because the VAC output of the sensor cannot directly interface with the microcontroller. A rectifier will therefore ensure that no negative voltage will reach the microcontrollers pins.

2.4.4 GPS Module

For the proposed method to achieve its required objectives the electronic devise should be able to tell its exact location. To do this a GPS module will be required. GPS satellites orbit the earth two times in 24 hours. The satellites transmit a unique signal and orbital data that allow GPS modules to decode the exact location of the satellite. With distance measurements from 3 or 4 satellites the GPS module can determine accurately its location.

The GPS module that will be used is NEO-6M which is as shown in fig 2.6.

Figure 2.6 NE06M GPS module

2.4.5 MySQL

According to Oracle a database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

MySQL is an open-source relational database system. It has various applications and is mostly used for developing web-based software. There are existing libraries that enable Nodemcu to read and write data to a MySQL database wirelessly.

MySQL database can also be accessed and connected to QGIS which is used for mapping.

2.4.6 Geographic Information System

This a system that creates, manages, analyses and maps all types of data. A GIS is used to link data to a map by use of location data. A GIS is a very useful and necessary component of the proposed

method. After a power outage has been detected and its location identified the location should be updated on a map. A GIS will help in creation and updating the map based on power outages detected. There are many GIS softwares but the one which most fits the proposed system is QGIS

QGIS is a GIS application that is free and open source. It therefore is a user-friendly Open-Source Geographic Information System (GIS) licensed under the GNU General Public License. QGIS is an official project of the Open-Source Geospatial Foundation (OSGeo).

Core functions and plugs provide various capabilities. One can visualize, manage, edit, analyze data and compose printable maps.

2.4.7 Geoserver

It is a server based on java that allows one to view and edit geospatial data.it therefore allows for great map creation and data sharing. It is free and open source. QGIS has a Geoserver plugin such that one can analyze data and map it on QGIS and use geoserver to publish the map on a website. Thus, continuous update of a map on a website is possible.

2.4.8 Node.js

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.

Node.js allows the creation of Web servers and networking tools using JavaScript and a collection of "modules" that handle various core functionalities. Modules are provided for file system I/O, networking (DNS, HTTP, TCP, TLS/SSL, or UDP), binary data (buffers), cryptography functions, data streams, and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications.

2.4.9 Android SDK

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample

code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later, and Windows 7 or later. As of March 2015, the SDK is not available on Android itself, but software development is possible by using specialized Android applications.

Until around the end of 2014, the officially-supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) Plugin. As of 2015, Android Studio, is the official IDE; however, developers are free to use others, but Google made it clear that ADT was officially deprecated since the end of 2015 to focus on Android Studio as the official Android IDE. Additionally, developers may use any text editor to edit Java and XML files, then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

Enhancements to Android's SDK go hand-in-hand with the overall Android platform development. The SDK also supports older versions of the Android platform in case developers wish to target their applications at older devices. Development tools are downloadable components, so after one has downloaded the latest version and platform, older platforms and tools can also be downloaded for compatibility testing.

Android applications are packaged in .apk format and stored under /data/app folder on the Android OS (the folder is accessible only to the root user for security reasons). APK package contains dex files (compiled byte code files called Dalvik executables), resource files, etc.

2.4.10 Thunkable

Thunkable is the platform where anyone can build their own mobile apps. Available for iOS and Android. It helps people build native Android and native iOS apps, as well as mobile responsive web apps.

Theoretically, you can build complex apps with Thunkable by dragging and dropping different logical components as if they were blocks. Using these building blocks is reminiscent of the visual programming language known as Scratch, which is largely an educational tool that creates limited logic and helps teach some basic coding.

2.5 Shortcomings of Existing Technologies

2.5.1. Shortcomings of Using Social Media to Report Outages

This has its method has its own shortcomings such as; People in Rural areas do not use social media as much as people in urban areas. If the system depends on bursts or a lot of mentions to detect the power outage then it becomes difficult to detect outages at sparsely populated areas It is slow. It takes time for a power outage to be detected and its location correctly identified. Due to privacy concerns most customers are not willing to share their exact locations on social media. this makes it hard to determine the exact location of a power outage. There's no recorded data of past incidences on the distribution line or switchgear affected by the power outage.

2.5.2 Shortcomings due to Geo Tagged Power Outages Using Social Media.

This method gives has shortcomings such as; Slow response time to power outages. The customer first has to tweet or mention the power outage together with his meter number before it is geocoded. Redundancy. Social media users from a densely populated area or region are likely going to report about the same power outage. The time taken to filter, analyze and geocode tweets reporting the same outage might delay detection of another outage.

2.5.3 Shortcomings of Using Small Scale Scada to Detect Outages

This system is very costly. There are a lot of requirements both in software and hardware for a SCADA based system to be implemented. The costs increase exponentially for a large-scale distribution system. Personnel must also be trained on how to use and interact with the SCADA system.

2.6 Improvements

The proposed system seeks to improve the response time to power outages and have an automatic system that doesn't entirely depend on customer complaints. It seeks to ensure that even before the complains are made, the company gets to know that there's no power at certain areas. This will be done through an automatic power outage notifier system that can automatically sends notification to Kenya Power whenever there's a power outage. The system will comprise of; Current Transformers (CTs) which can detect when current is flowing and not flowing on the line, whenever it's not flowing a notification is sent to Kenya Power about the same through a system that which will comprise of; Microcontroller, WIFI modules and location sensors to notify on the exact location where a power outage has occurred.

The system is also expected to be cheaper compared to systems like SCADA since no personnel will be required to monitor the devices after they have been installed on the line. Also, the components of the system will be cheaper.

CHAPTER 3 : METHODOLOGY

3.1 Block Diagram

Figure 3.1 shows the block diagram of the system i.e. Automatic Power Outage Notifier System.

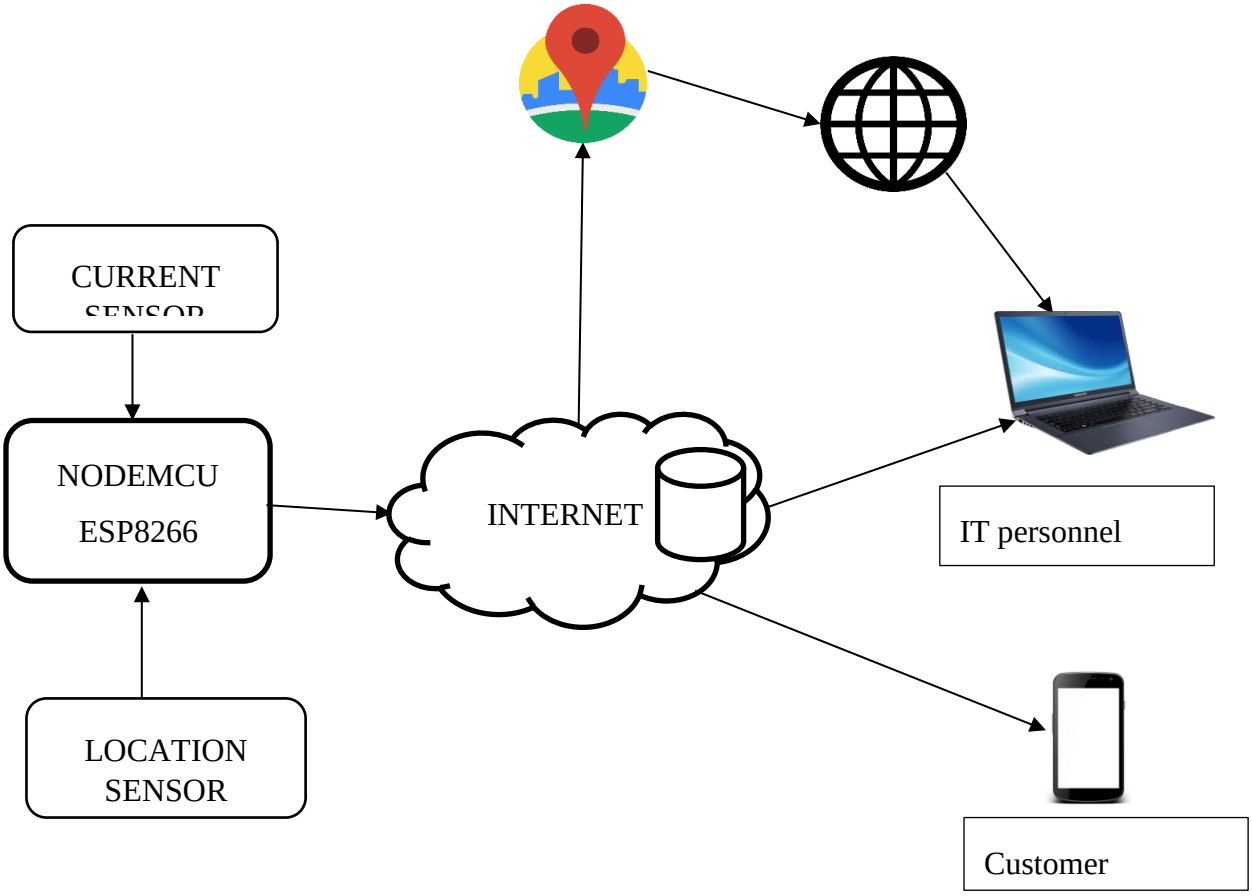


Figure 3.1 Automatic Power Outage Notifier System Block Diagram

3.2 Software Design

During initialization the microcontroller connects to the internet via a WIFI connection. The next step is connecting to the cloud and ensuring that the microcontroller can write data to a remote database.

During initialization the microcontroller connects to the internet via a WIFI connection. The next step is connecting to the cloud and ensuring that the microcontroller can write data to a remote database. This is illustrated in figure 3.2.

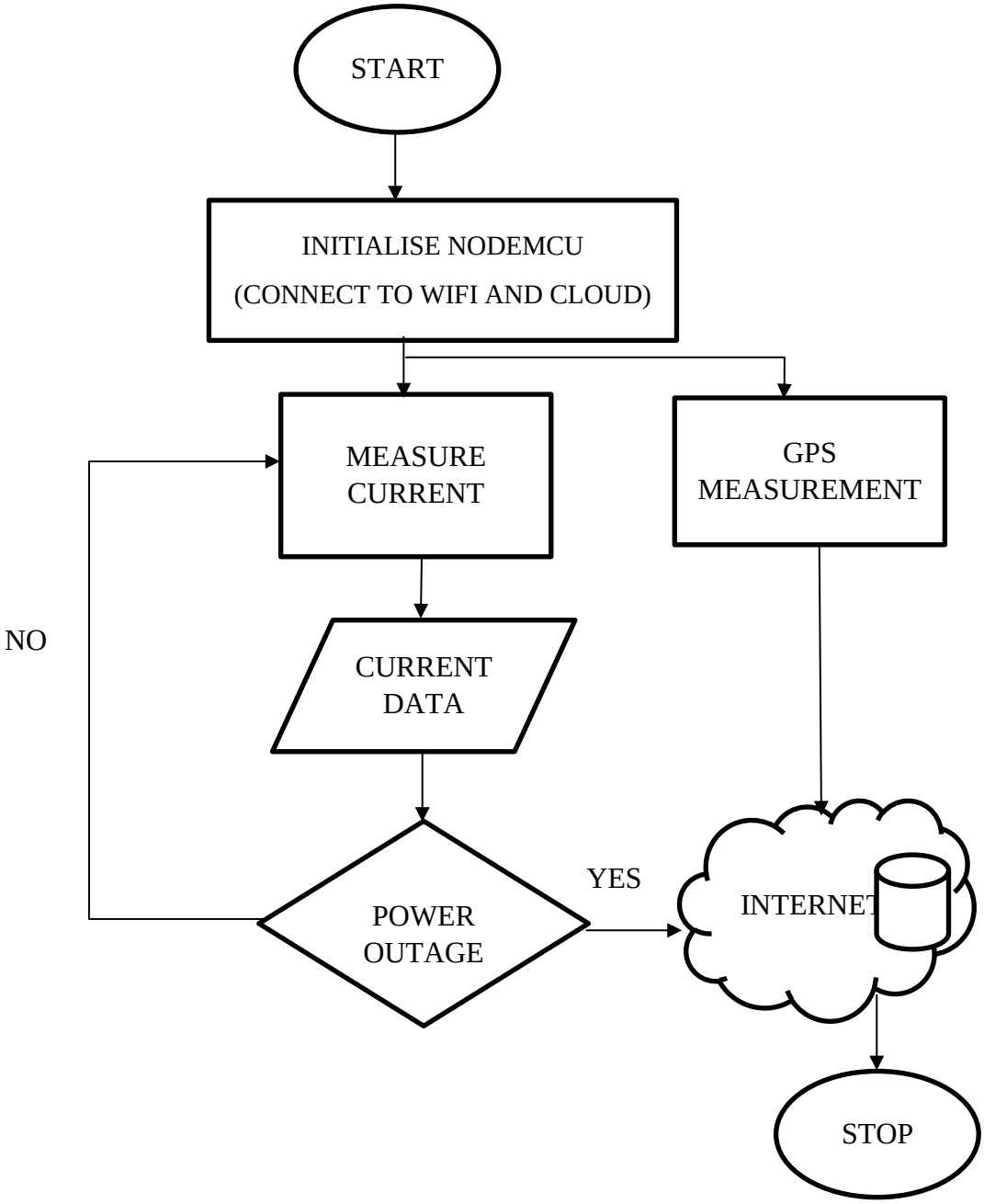


Figure 3.2 Electronic Device Interfacing Flowchart

If the current measurement is below 0 then the microcontroller sends and writes data to the cloud database. This data includes location data obtained from real time measuring of the location by the location sensor

3.2.1 Current Sensor

Using a current transformer AC current through a line is measured. A burden resistor limits the amount of output voltage at both ends of the current transformer. The AC voltage at the outputs is then rectified and converted into digital signals by the ADC port in the Nodemcu.

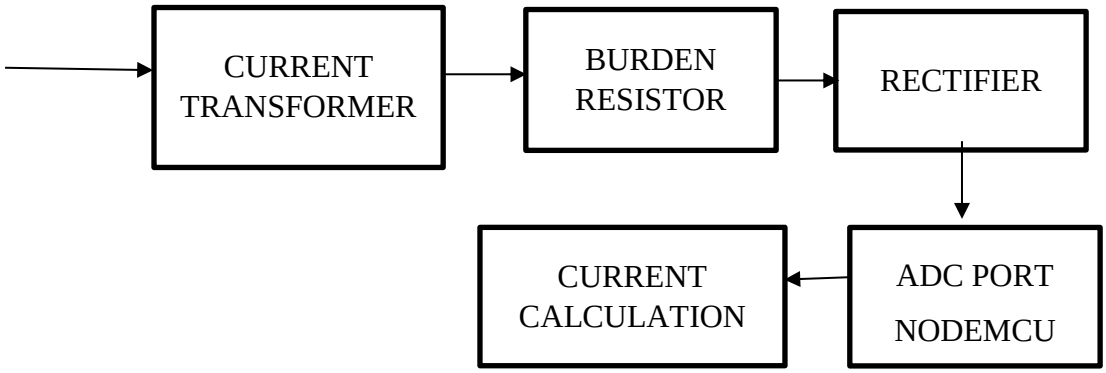


Figure 3.3 Current Detection and Measuring

Sct 013-030 is used to take the current measurements. it has a built-in burden resistor. its maximum output is 1volt for 30 Amperes.

Nodemcu esp8266 has only one analog input pin. This is not sufficient for measuring current in a 3-phase setup which requires at least 3 analog pins. Therefore, an ADS1115 is used. it is a 16-bit analog to digital converter as shown in figure 3.4.

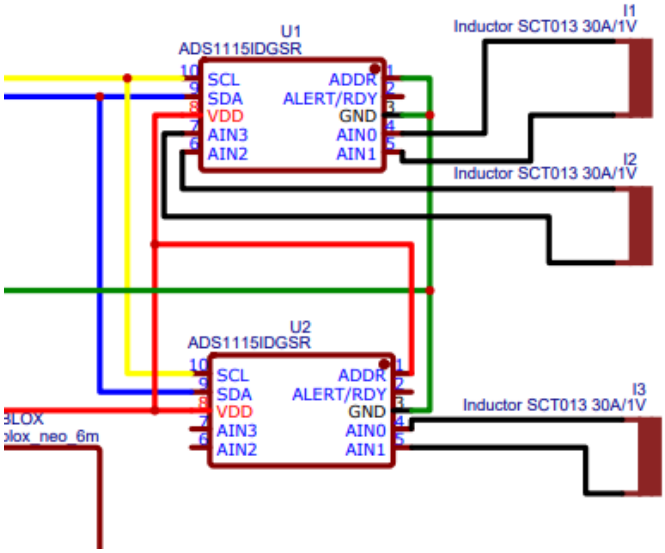


Figure 3.4 Current Detection Circuit

Apart from increasing the number of analog pins the ADS1115 also increases the resolution of digital to analog conversion. This is because the nodemcu's ADC is only 10 bit compared to 16 bit of the ADS1115.

ADS1115 has two methods of measuring input data i.e. voltage level of the current transformers. These two methods are Single Ended method and Differential method.

In single ended inputs measure the voltage between the analog input channel and analog ground. In differential method inputs measure the voltage between two analog input channels i.e (A0&A1 or A2&A3).

However, single ended method can only measure positive voltages and therefore one only gets an effective 15-bit resolution. Not only does the differential mode provide the full 16 bits of resolution but also it offers more immunity from electromagnetic noise.

If differential mode is used two ADS1115 are required because one input uses two input pins and the ADS1115 only has 4 inputs per board.

The Microcontroller uses I2C connections to communicate with the ADS1115. I2C only requires two pins to communicate. These pins are SDA and SCL. The default SDA and SCL pins for a nodemcu esp8266 microcontroller are D1 and D2. None the less they can be allocated to other pins through coding. The ADS11x5 chips have a base 7-bit I2C address of 0x48 (1001000) and a clever addressing scheme that allows four different addresses using just one address pin (named ADDR for Address). To program the address, connect the address pin is connected as follows:

- 0x48 (1001000) ADDR -> GND
- 0x49 (1001001) ADDR -> VDD

In the code current measurement begins by first including the Adafruit ADS1X15 library. Two variables ads and ads2 are then inherit the library properties. To boost small signals, the gain on the ADC can be adjusted. For an input range of +/-1.024V a gain of four is recommended and set. Getting current per phase is done through a function called get current (); The ADC reads the differential input from two pins. This is done in many cycles up to 1000ms. the value measured is compared to a previous measured value until the maximum value is obtained. This is the peak current of the AC current. Since it's AC the peak current

can be obtained from both positive and negative cycles. The ADC has a maximum input levels of 2^{16} . Half of the levels are used to represent negative levels and the other half positive levels. Therefore, the smallest change is represented by equation 3.1

$$\frac{1}{32000} = 0.00003125 \qquad (3.1)$$

This value is called the multiplier and is used to convert the ADC's measurement back to real values. Assuming the maximum value is the peak voltage measured it is multiplied by the inverse of square root of 2 to get the voltage in Rms. In the current transformer used a value of 30A is represented by 1v at the inputs. therefore, the maximum value has to multiplied by this factor. to get current in rms the voltage is multiplied with this factor.

sketch_oct31a

```
float getcurrent()
{
  float voltage;
  float current;
  float sum = 0;
  long Raw=ads.readADC_Differential_0_1();
  long maxRaw=Raw;
  long minRaw=Raw;
  long time_check = millis();
  int counter = 0;

  while (millis() - time_check < 1000)
  {
   Raw=ads.readADC_Differential_0_1();
   maxRaw= maxRaw > Raw ? maxRaw: Raw;
   minRaw= minRaw > Raw ? minRaw: Raw;
    //current /= 1000.0;
delay(5);

  }
  maxRaw= maxRaw > -minRaw ? maxRaw: -minRaw;
  voltage =maxRaw*multiplier/1000;
  float voltageRms= voltage*0.70710678118;
  current=voltageRms*FACTOR;

  return (current);
}
float getcurrent2()
{
  float voltage;
  float current;
```

Figure 3.5 Function getcurrent ()

3.2.2 Location Sensing

The flowchart below (Figure 3.6) illustrates the steps taken to obtain valid location coordinates.

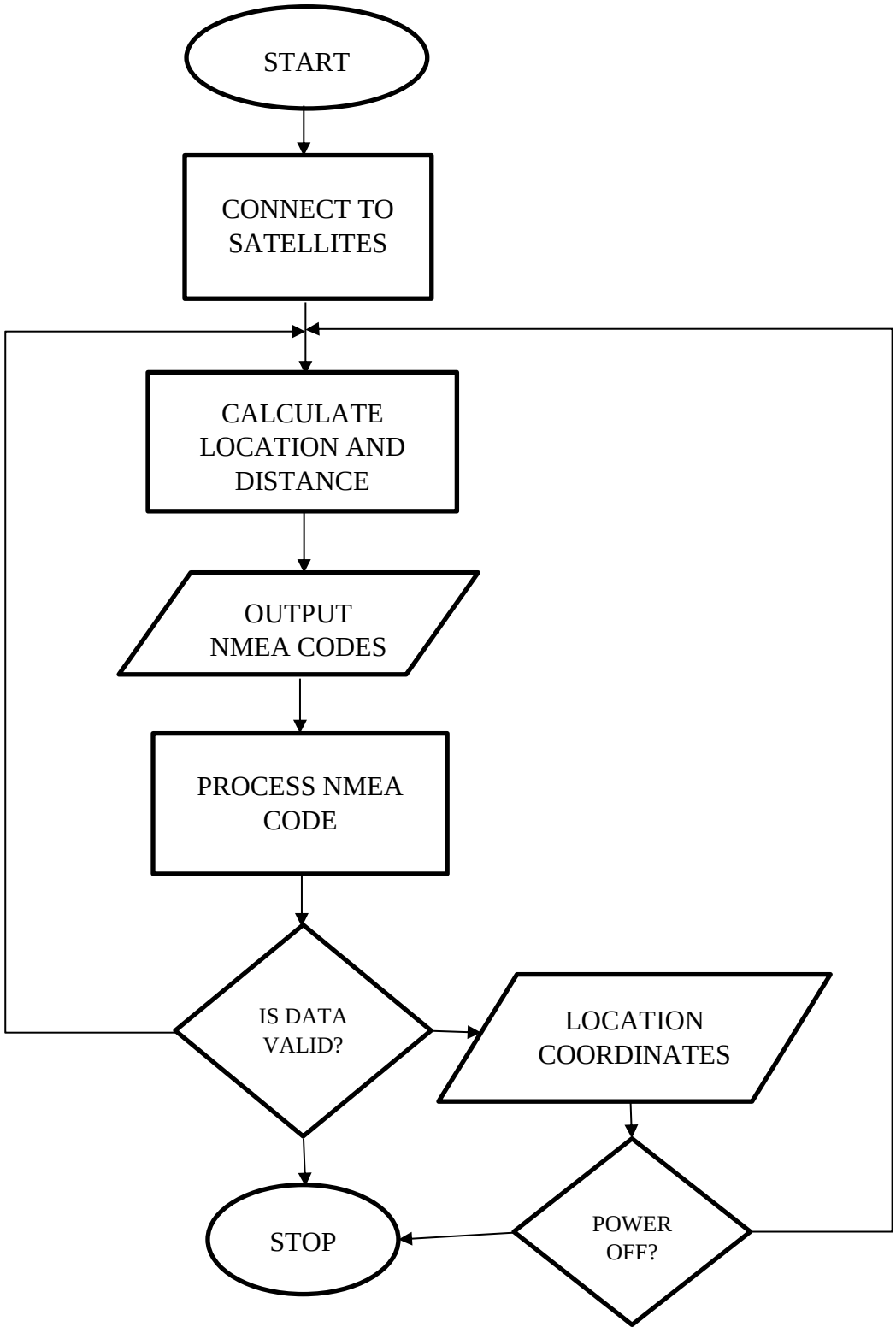


Figure 3.6 Location Sensing Block Diagram

The ublox ne06m is connected to the nodemcu through two digital pins (D5, D6). It uses a serial interface to communicate. The data received from the ublox module is in form of GPS NMEA sentence codes. To convert the codes and interpret them to GPS coordinates, Tinygpsplus library is used. It is a library whose main function is to convert the NMEA data to information such as time, location or distance. The ne06m continuously sends NMEA data to the nodemcu. However, location coordinates are only obtained when the GPS module has a fix and the NMEA data is deemed to be valid. This is shown in figure 3.7.

```
TinyGPSPlus gps;  // The TinyGPS++ object

SoftwareSerial ss(12, 14); // The serial connection to the GPS device
float latitude , longitude;
int year , month , date, hour , minute , second;
String lat_strl , lng_strl;
String lat_str="NotValid";
String lng_str="NotValid";
int pm;
  sketch_oct31a§
  voltage =maxRaw*multiplier/1000;
  float voltageRms= voltage*0.70710678118;
  current=voltageRms*FACTOR;

  return (current);}
void getGps(){


while (ss.available() > 0) //while data is available
    if (gps.encode(ss.read())) //read gps data
    {
      if (gps.location.isValid()) //check whether gps location is valid
      {
        latitude = gps.location.lat();
        lat_str = String(latitude , 6); // latitude location is stored in a string
        longitude = gps.location.lng();
        lng_str = String(longitude , 6); //longitude location is stored in a string
      }
    }
  Serial.println(lat_str);
Serial.println(lng_str);
```

Figure 3.7 TinyGPS library

3.2.3 Connecting Hardware to Cloud Database

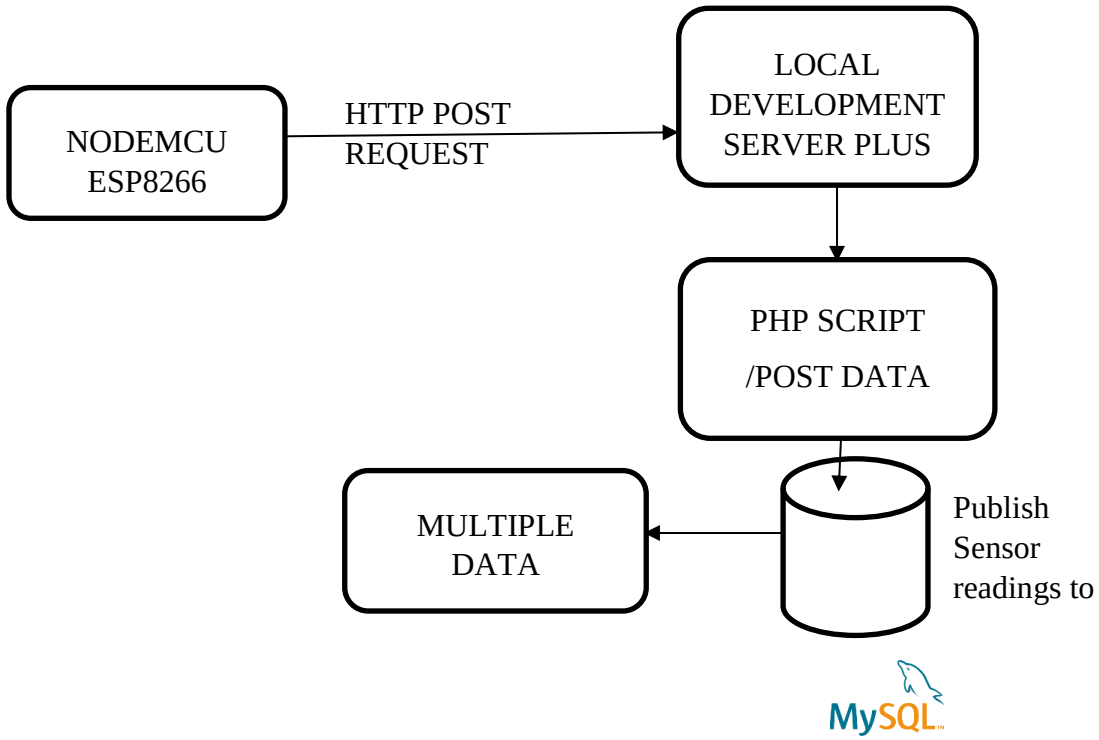


Figure 3.8 Connecting Hardware to Cloud Database

To connect to a database the nodemcu first connects to the internet. This is enabled by its internal WIFI module.

The nodemcu writes data to the MySQL database by making a http post request to a PHP script to insert data into the database as shown in figure 3.9 It makes 3 independent post requests to 3 php scripts. the scripts are dbwrite.php, dbwrite2.php and finally dbwritegps.php

The nodemcu posts a request to dbwrite.php when the power status has changed. i.e. if any of the three phases has turned on or off. It posts requests to dbwrite2.php periodically. This contains the current values in number of all the 3 phases. it also posts a request to dbwritegps.php when the location of the sensor has changed significantly. i.e. more than 0.01 degrees for latitude or longitude.

All of these files update the same database table in MySQL called senloc. The data is inserted into rows depending on the sensors unique id (UID).

Once the data has been logged into the database, it can be accessed by numerous services for processing into different kinds of information.

Figure 3.9 PHP post request function

Additional code on connecting the hardware to the cloud database can be found in Appendix A

3.2.4 MySQL Database

The database is hosted in an online ubuntu droplet provided by Digital Ocean. To connect and easier management of the database phpMyAdmin was also installed in the ubuntu droplet. The main database that facilitates the project is called check_data. Within check data there are 4 important tables Senloc, Sentatus, Locs and User_Table and are related as shown in figure 3.10

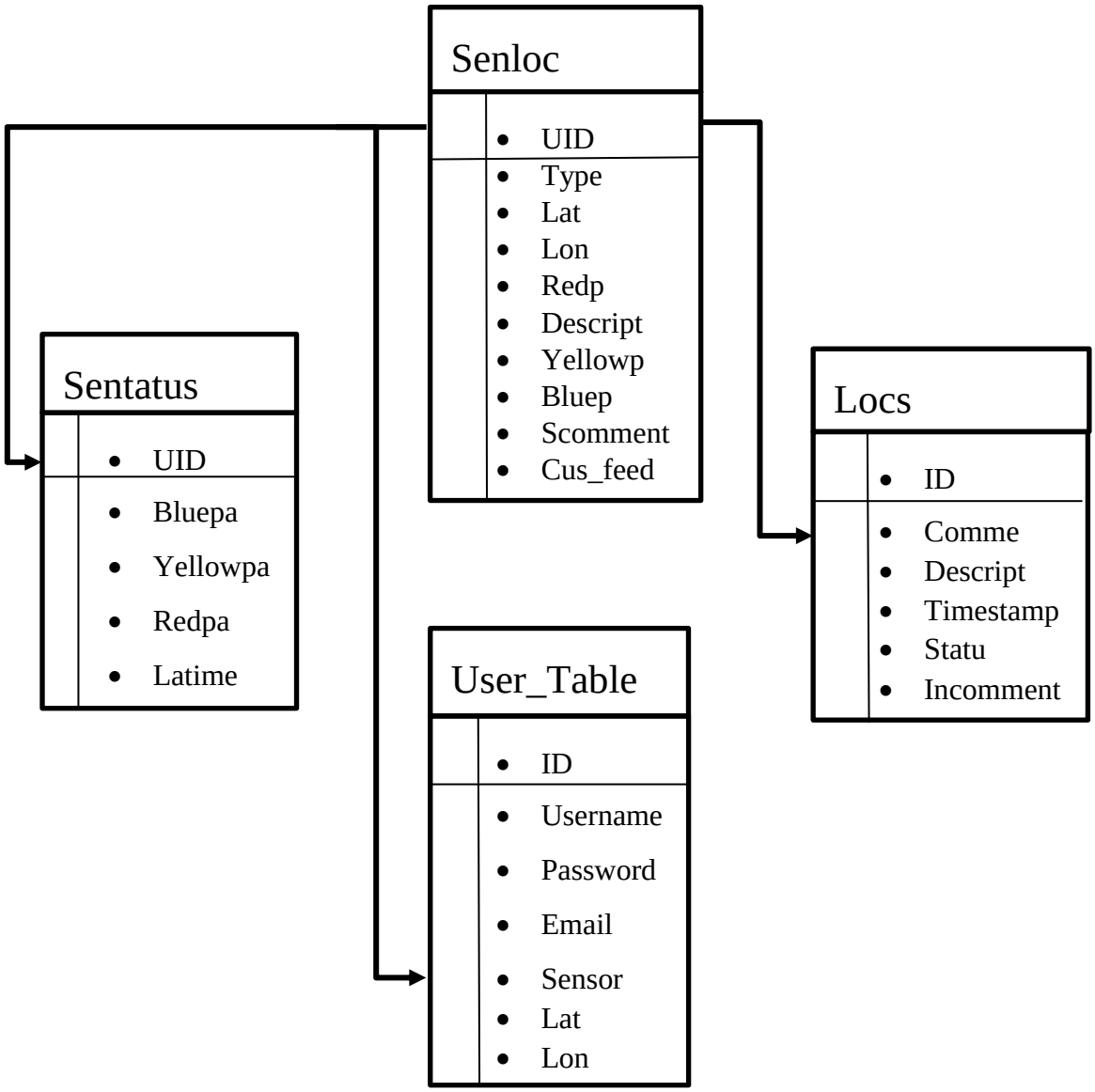


Figure 3.10 Database Relational View

3.2.4.1 Sensloc Table

The main table is called Sensloc. Its primary key is a unique id that is used to differentiate each sensor. Other columns identify the sensors location, power status, internal comment, customer feedback and the description of where the sensor is.

Columns Lat and Lon are used to store the location of the sensor in degrees. Internal comment is used to record information that might be helpful to other staff. It will also be used to tell or record past occurrences of the transformer where the sensor is placed. This helps in predicting issues that might cause a power outage.

Cus_feed is used to broadcast messages to customers who are under a particular sensor. this integrates with the app.

Descript gives a slight idea of where the sensor is located for quick navigation.

Status shows the current state of the sensor. If any phase is off the status is off, otherwise if all phases are on then the status is on. Bluep, Yellowp, Redp indicate the current status of each phase. Type indicates whether the sensor is placed on a Transformer or on a line.

3.2.4.2 Locs table

It records the most recent sensor whose status has changed. ID is an auto increment column that indicates the index of all records. It is also the primary key of the table. Comme is linked to transformer uid from senloc table. Timestamp records the time of insertion of the record and is useful in recording and calculating the elapsed time since the status change. Statu and incomment are linked with status and scomment in senloc respectively. Unlike in senloc where the data is updated directly from http post requests, data in the Locs table is updated through a trigger in senloc table which is shown in figure 3.11.

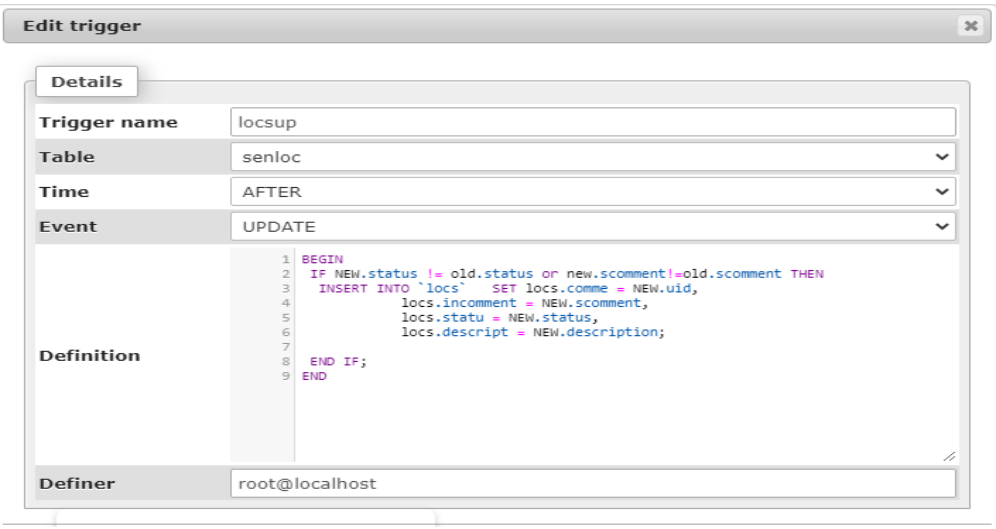


Figure 3.11 Locsup Trigger

3.2.4.3 Sentatus Table

It has 5 columns, uid, redpa, bluepa, yellowpa and latime. The primary key is uid which also identifies the sensor represented in each row. Redpa, bluepa, yellowpa indicate the current Irms of each phase of the sensor represented. Latime is a timestamp and indicates the last time the values in each row were updated. Sensor uids are first inserted through a trigger in senloc table. The trigger is an after-update trigger as shown in figure 3.12 and is used to insert sensors that have not yet been recorded in the sentatus table. From then on, the row is updated periodically through a http post request directly from the sensor.

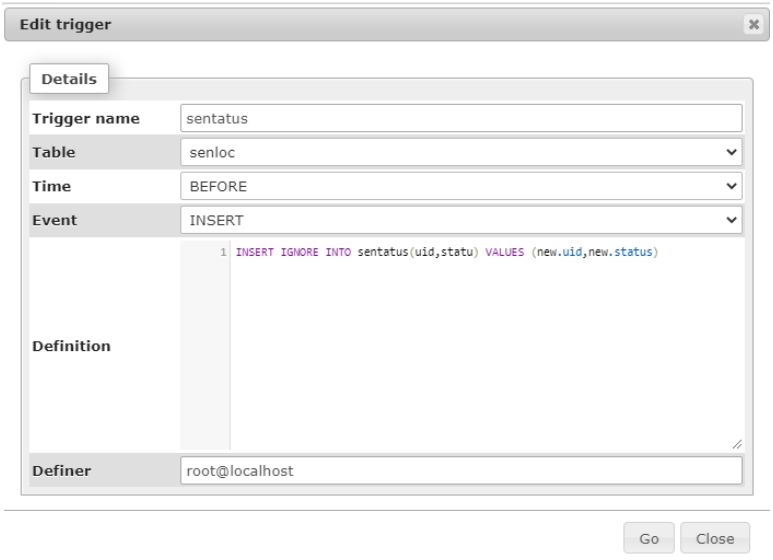


Figure 3.12 Figure 3.12 Sentatus Trigger

3.2.4.4 User-Table

This table contains records of registered users. It has 7 columns. The primary key is ID which is a unique identifier for each user. The other columns include username, email, sensor lat, lon.

Username and email are used to login via the mobile android app. Sensor is the unique ID allocated to the sensor placed on the transformer that serves the user. Lat and Lon are used to record the users home location and enable the administrator to allocate the right sensor to the user's profile.

3.2.5 Map Service on a Website

3.2.5.1 Map.php

The flowchart figure 3.13 illustrates how the power status of all sensors are plotted on a map on an online webpage.

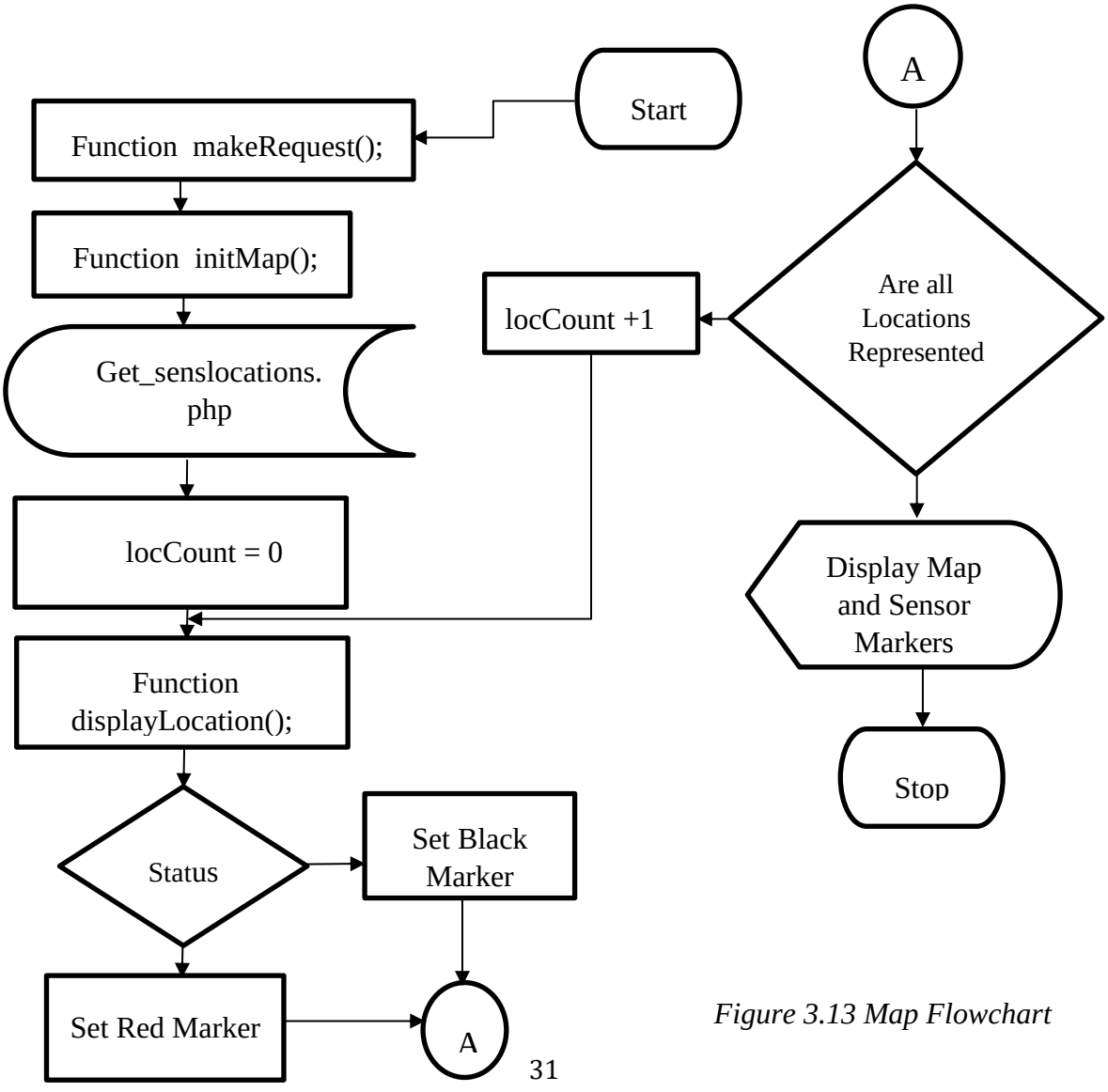


Figure 3.13 Map Flowchart

In the event of a power outage, the systems administrator is required to be informed and alerted of the power outage as soon as possible. To accomplish this the status of each sensor together with their exact location is plotted and represented on a map.

The following flowchart shows the process taken to display the sensor status on a map using google maps API.

Function makeRequest () is used to invoke get_senslocations.php which retrieves the location details of all sensors from the database.

Function initMap() initializes the map. It contains and sets various properties of the map that will be displayed. The functions are shown in figure 3.14

```
makeRequest('get_senslocations.php',function(data){
  var data= JSON.parse(data.responseText);
  for (var i=0;i<data.length;i++){
    displayLocation(data[i]);
  }
});
}

function initMap(){
  var mapOptions={
    zoom:15,
    center: center,
    mapId:'5028ea12062d5f68'
  }
  map=new google.maps.Map(document.getElementById("map"),mapOptions);
 //var marker = new google.maps.Marker({
    //map:map,
    //position:center,
// });
```

Figure 3.14 Function InitMap & MakeRequeSt

Function displayLocation() is used to display each sensor's location and status. It is called in a for loop to ensure that all locations/sensors have been represented and assigned to markers

Additional code on Map.php can be found at Appendix B

3.2.5.2 Index.php

The flowchart below (figure 3.15) illustrates the steps taken to record and upload a new sensor into the system.

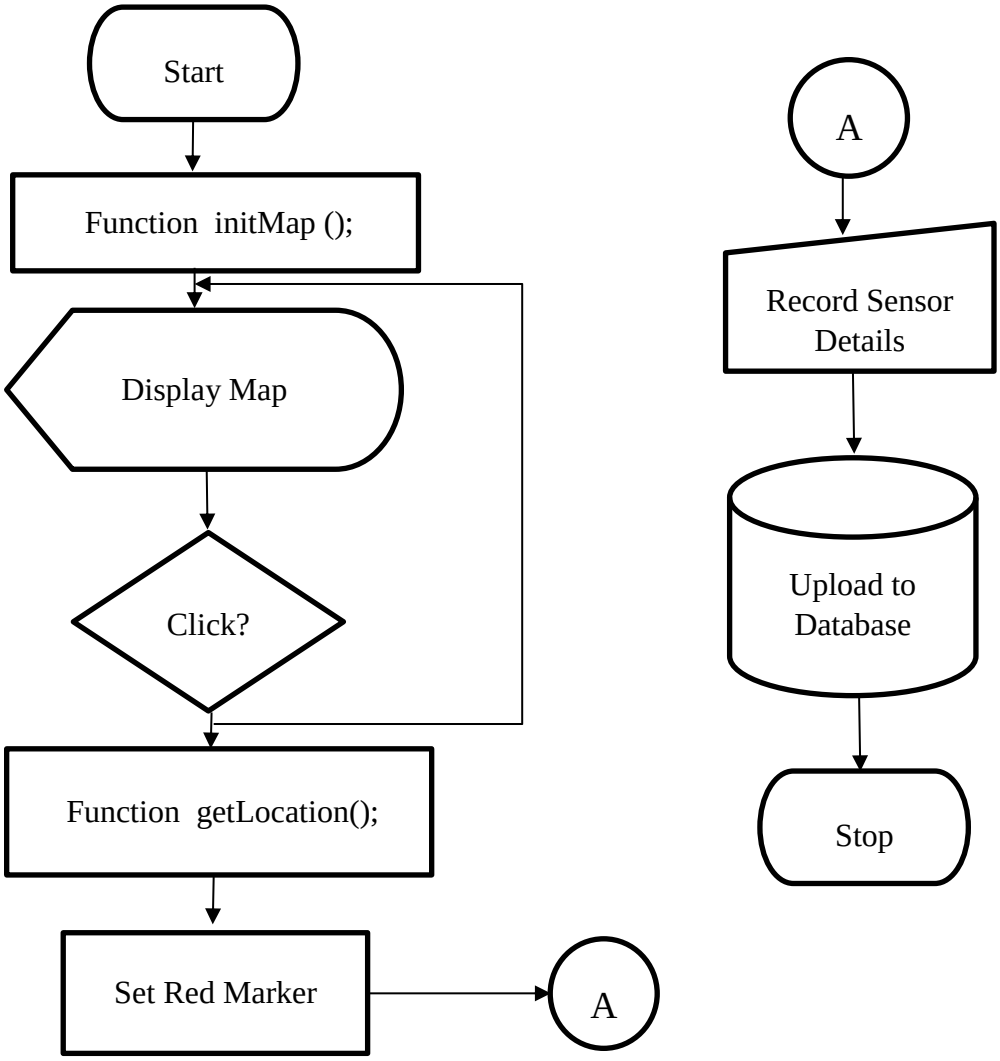


Figure 3.15 Record a New Sensor Flowchart

Its represented as Record New Sensor in the website. The map is used to enable the admin to approximate the location of a sensor while its first being set up. Unlike the power outage map the location marker is only placed after the admin clicks on an area of the map. After the location has been obtained, the admin can now complete the rest of the form. The submit button sends a post request to insert file which uploads the values to the MySQL database.

Additional code on Index.php can be found at Appendix B.

3.2.5.3 Map3.php

The flowchart in the figure below (figure 3.16) shows how a customer is linked and allocated a specific sensor.

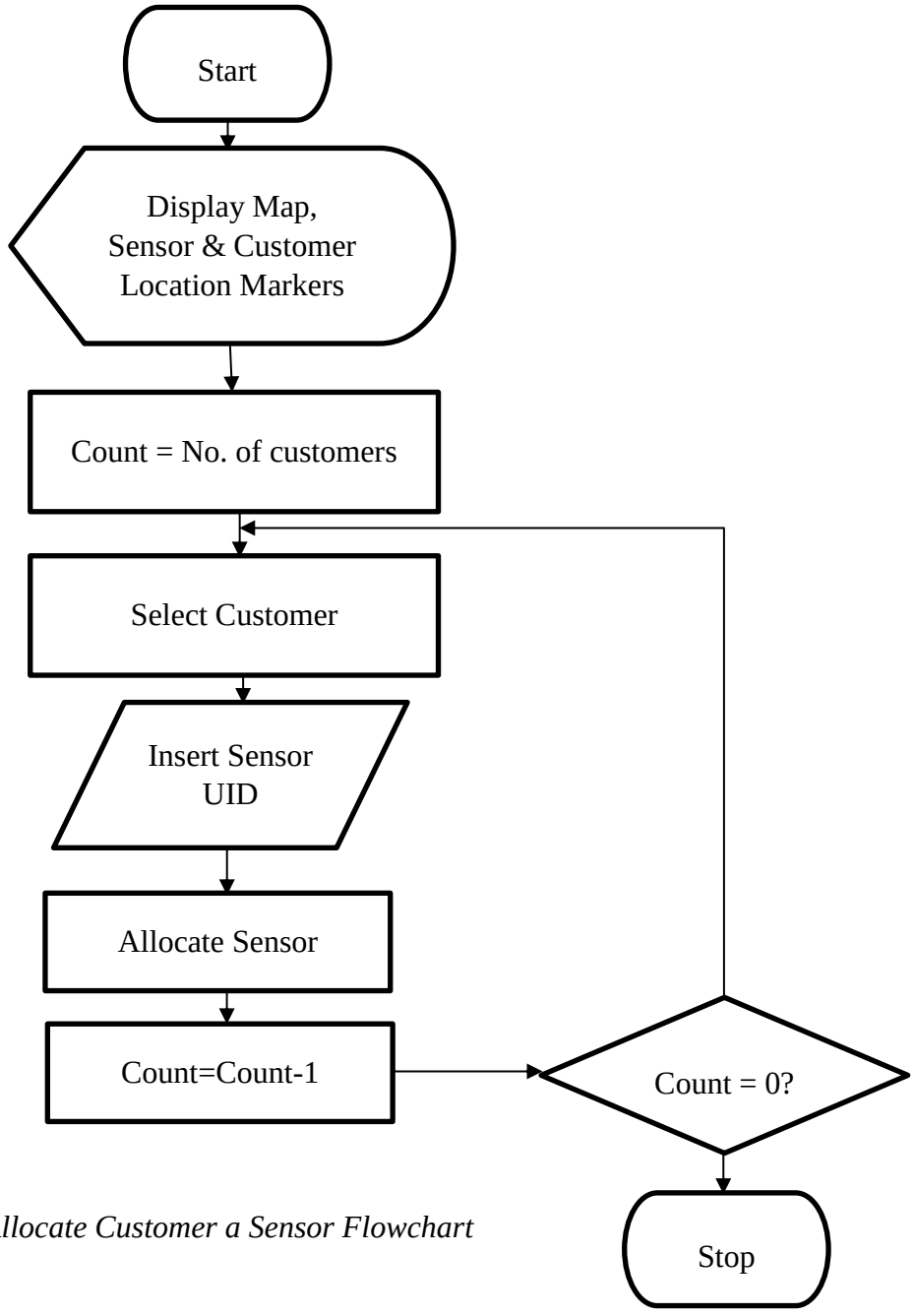


Figure 3.16 Allocate Customer a Sensor Flowchart

It is similar to the power outage map. However, it contains an additional function which displays location of customers who haven't been allocated sensors. The customers home location is displayed with a purple icon.

The markers used to represent customers are also temporary and disappear once the customer has been allocated a valid sensor. The flowchart below represents the steps taken to allocate a customer a valid sensor after loading and representing all markers on the map.

An ajax script shown in figure 3.17 is used to send a post request to allocate.php. This enables allocate.php to update the database without reloading the web page as shown in the figure below.

```
$.ajax(
  {
    type:"POST",
    url:"allocate.php",
    data:"username="+z+"&sensor="+str,
    success: function (msg){
    response(msg,z,str,markerId);
    }
  }
);


}
function response(msg,user,sensor,markerid){
if(msg=="Values Inserted in MySql Database Table")
{
  var marker=clustmarker[markerid]
  marker.html='<div class="infowindow"<strong>' +user+'</strong>'
  +'<br/><p >Allocated Sensor </p>'+'<br/><p>'+sensor+'</div>'

  marker.infowindow.setContent(marker.html);
  count=count-1;
  refreshTab();
  setTimeout(function(){marker.infowindow.close();},3000);
  setTimeout(function(){  marker.setMap(null); // set markers setMap to null to remove it from map
    delete clustmarker[markerid]; // delete marker instance from markers object;
  },5000);
}
```

Figure 3.17 Ajax Script

Additional code on map3.php can be found at Appendix B.

3.2.5.4 Power outage Sensors webpage

The flowchart in figure 3.18 illustrates how data is fetched from the database and represented in tabular form on a web page.

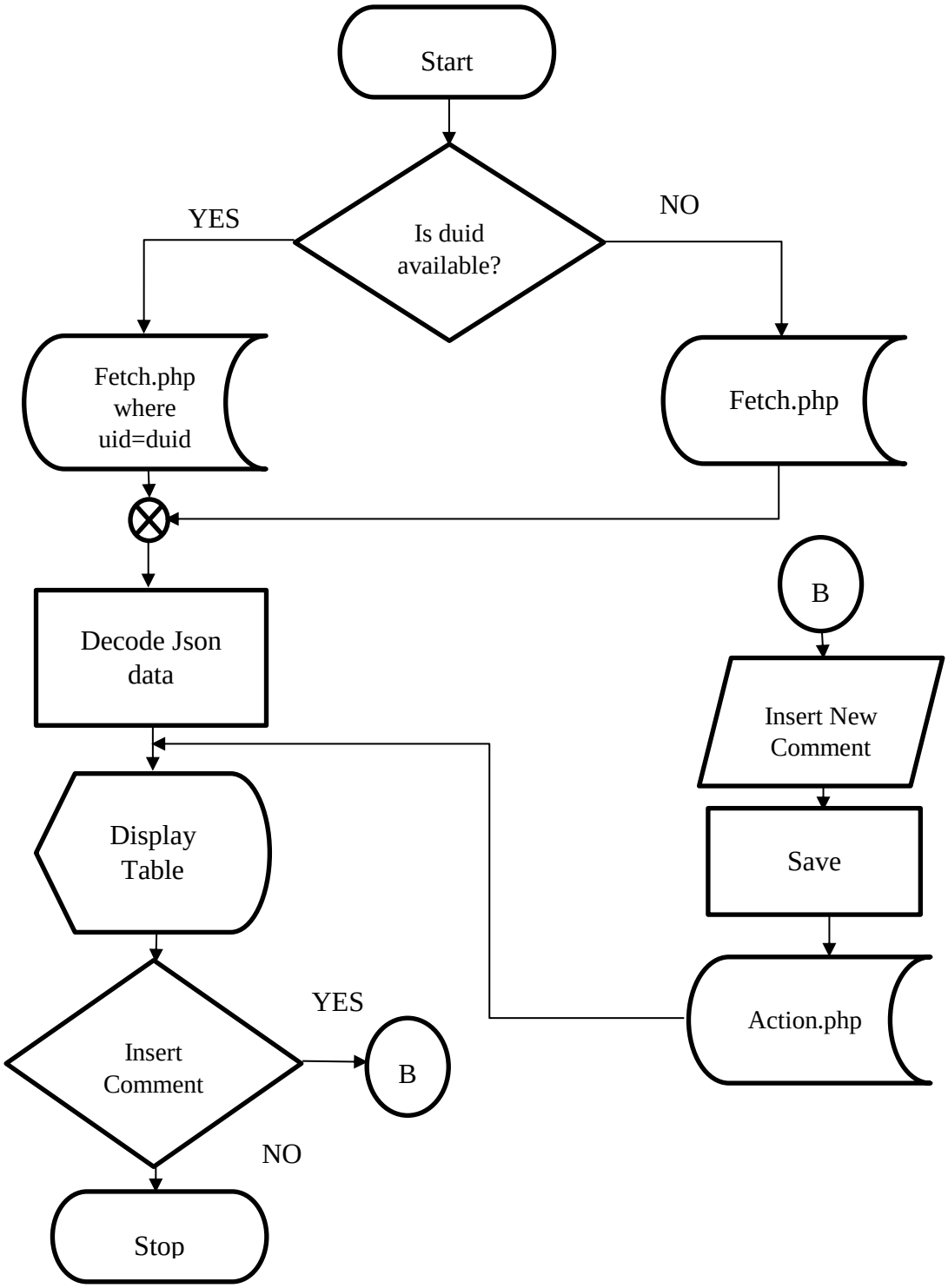


Figure 3.18 Power Outage Sensor Webpage

The google Maps API has methods and inbuilt functions that enable the maps to be interactive.

One such feature is the method infowindow(). It pops up a small window that dispays additional information about the marker. Through this a user can be redirected to two other pages that show extensive information and records regarding the status of the sensor i.e Power Outage Sensors (comment.php), Sensor status (comment2.php), Sensor History (comment3.php)

The webpage can be accessed independently or from an infowindow on the power outage map. When called independently it shows all sensors whose current status is 'OFF' but if its called from an infowindow it only shows the value of the sensor on the infowindow .

Once it's called, fetch.php is invoked which responds with a json encoded array of values from the senloc table. The data is then forwarded to the current page.

it represents the data in form of a table with 7 columns i.e. UID, Status, Redp, Yellowp, Bluep, Incomment and Feedback

A user is able to alter only the incomment and feedback column. These two columns are used to record the reasons for the power outage on each sensor and to provide feeedback to affected customers automatically.

Once the two values have been altered an ajax post request is sent to action.php which, without interrupting or refreshing the current webpage, uploads the values inserted into the Mysql database senloc table.

3.2.5.6 Sensor Status

It is of similar design to the power outage sensor webpage. However, it displays information from a different table, senstatus. It uses fetch2.php to obtain data from the table in the MySQL database

It represents the data in 6 columns i.e. UID, Status, Bluep, Yellowp, Redp and last_connected. Unlike the previous webpage the values of Redp, Yellowp, Bluep are in numeric values. The main aim of this webpage is to check if the electrical device is working correctly and is in constant communication with the database.

3.2.5.7 History

It is used to display faults that have previously happened on the line or transformer that the electronic device is attached to. It uses fetch3.php to fetch data from Locs table.

It can only be called from the power outage maps and only displays values of one sensor.

Comment.php, comment2.php, comment3.php and their corresponding fetch.php files can be found at Appendix C.

3.2.6 Mobile application

We developed an android app using android studio.

In android studio each page is represented by a class called an activity which is also linked to an xml file that determines the visual layout. The android app is named Power Status. It has 4 classes 3 which are activities. The classes are My Singleton, MainActivity, RegisterActivity and AppStartActivity.

3.2.6.1 RegisterActivity

It is linked with RegisterActivity.xml

It is used to record a customer's details which include their username, email and password . once all inputs have been filled then a user can press the register button which invokes a http post request to register.php

Once the post request has been received, first of all the email is checked to be valid. Then if its valid the password is checked if its within 8-40 characters. The username is then cross referenced with other usernames already stored in the user_table. If the username is unique the values are then inserted into the user_table table. This is illustrated in the flowchart in figure 3.19.

After registration the user is automatically redirected to the login page.

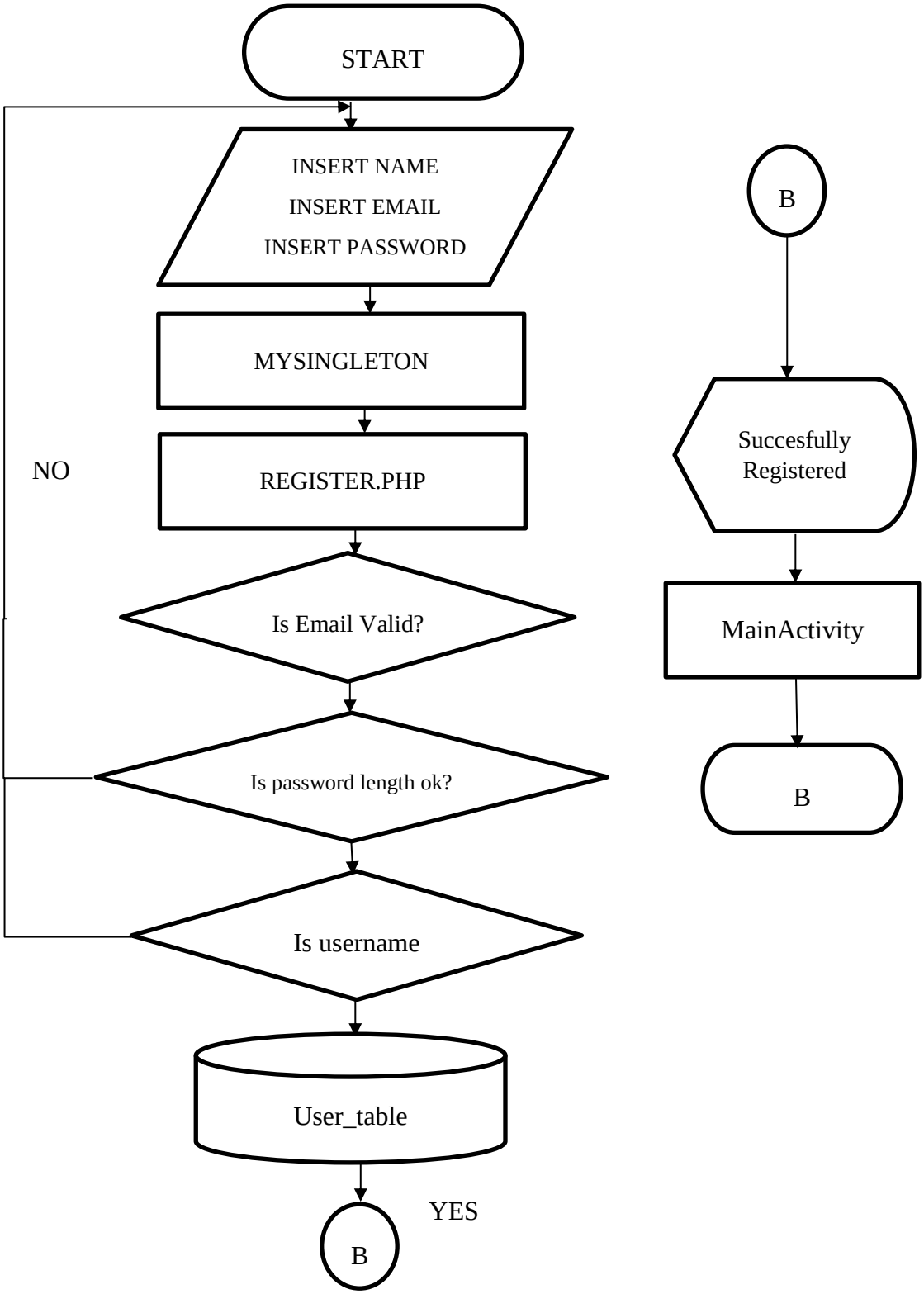


Figure 3.19 Register Activity Flowchart

3.2.6.2 MainActivity

The flowchart below illustrates the steps taken for a customer to log in via the mobile application.

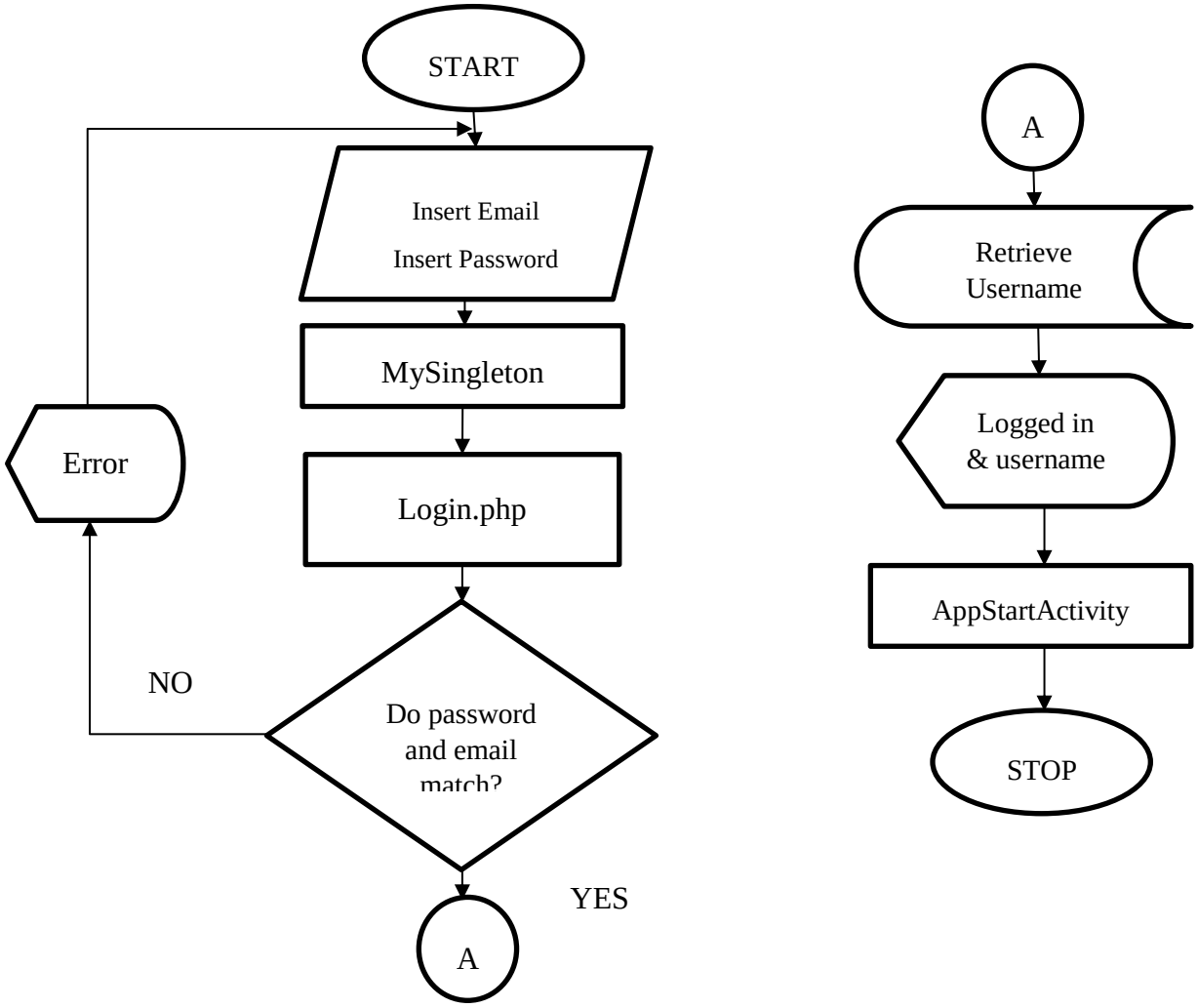


Figure 3.20 MainActivity Flowchart

It is the Login Page. It is linked with mainactivity.xml. it contains 2 input fields email and password. When a user inputs their email and password, through my singleton class a post request is sent to login.php. The two input values are cross referenced with records from the user_table. If the values are correct the php file responds with the username of the user and redirects them to appstartactivity.

There's also a radio buttons that requests the user if they would like to be logged in even after the app is closed

After a successful logging in, the username is recorded in the internal storage to be used in other pages.

3.2.6.3 AppStartActivity

It is the main page of the app. It is linked with appstartactivity.xml which determines the layout of the page. In this page the user can do three things i.e. Check power status, check on power outage status and finally access and change their location.

The user's username is the main link and is used to carry out the three functions.

Once the page has been created a post request containing the username is immediately sent to checkStatus .php through mysingleton class.

The php script first of all checks if the user has recorded or submitted their location. It then checks if a sensor has been allocated to the user. If the user has been allocated to a sensor the sensors uid is used to retrieve the current status of the sensor from senloc table. The script then echoes a json array containing the power outage status and if the connection has been successful.

If the power outage is 'OFF' the script also responds with the cus_feed field from the sensor table. This is used to give feedback to the customer while the power outage is worked on.

On the page there's a record my location button. When pressed the location of the user is recorded and sent via a post request to rLocate.php. The location is then inserted to Lat and Lon fields of the user_table. The users sensor is recorded as pAllocate which indicates that its pending allocation and helps the admin to know which customers haven't been allocated sensors.

3.2.6.4 MySingleton

It consists of functions and methods that enable the android app to send post requests and receive responses to and from the website.

Code on the mobile application can be found at Appendix D.

3.3 Hardware Design

The major hardware components of the project include three noninvasive current transformers , two Ads1115 Analogue to digital converters , esp82666 nodemcu microcontroller, GPS module, 18650 Li-ion battery , TP4056 battery charger and a solar panel .

3.3.1 SCT013-030 Non-Invasive Current Transformer

It produces an alternating current in its secondary, which is proportional to the AC current in its primary. It has a burden resistor connected across its secondary terminals such that its output is in the form of voltage. For SCT013-030 the maximum current amount that is recommended to measure is 30A and its output will be 1V.

There are three current transformers to measure the current of each phase.

3.3.2 Adafruit ADS115 ADC

It is a 16-bit analogue to digital converter with a programmable gain amplifier. It contains 4 analog inputs. it communicates with the nodemcu or any other microcontroller serially via SCL and SDA ports. It allows precise measurements to be obtained with very little effort.

ADS1115 has two conversion modes: single shot mode and continuous conversion mode. In single shot mode the ADC performs one conversion of the input signal upon request and stores the value to an internal result register. The device then enters a low-power shutdown mode. In continuous mode the ADC automatically begins a conversion of the input signal as soon as the previous conversion is completed

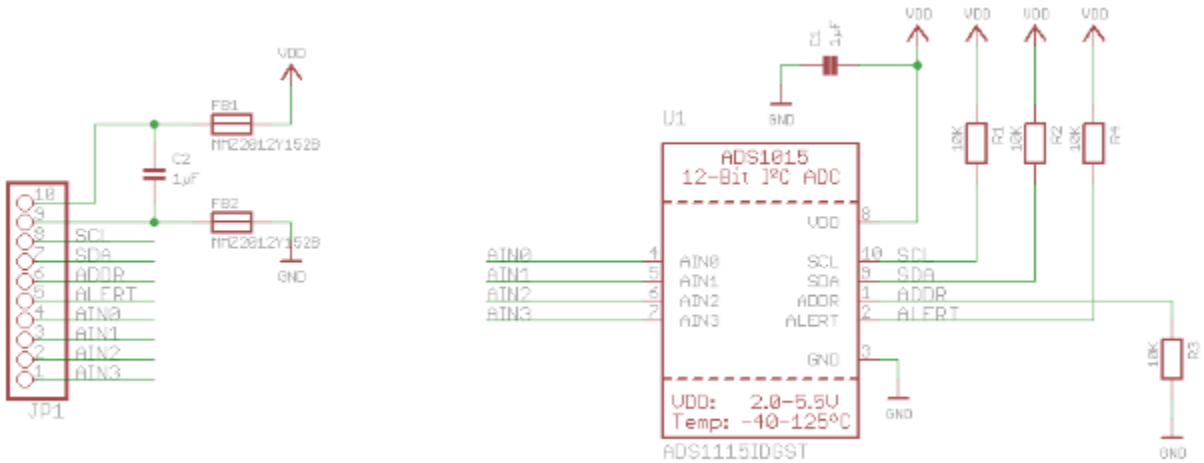


Figure 3.21 ADS1115 Schematic

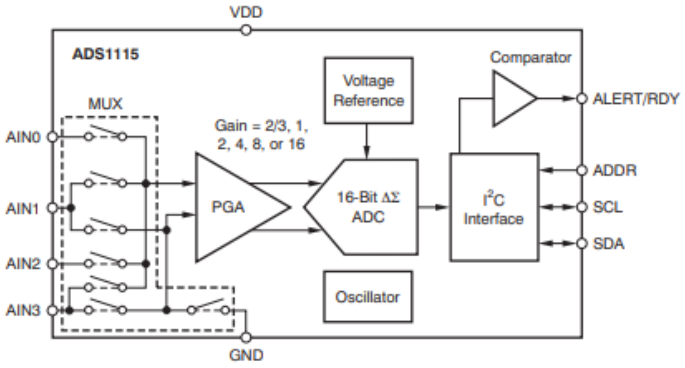


Figure 3.22 Internal Block Diagram

3.3.3 18650 Li-Ion battery

It is a 2600 mAh battery. its voltage ranges from 3.71 to 4.1V. Its maximum pulse discharge current is 2.6A. It is used to power the microcontroller, GPS module and two ADS1115.

Table 3.1 18650 Specifications

S/No.	Parameter	Values
1.	Nominal Voltage	3.6V
2.	Nominal Capacity	2850 mAh
3.	Minimum Discharge Voltage	3V
4.	Maximum Discharge Current	1C
5.	Charging Current	0.5C
6.	Charging Time	3 hrs
7.	Weight	48g

3.3.4 GY-GPS6MV2

It contains u-blox ne06m GPS module with antenna and a built in EEPROM . It has a power range of between 3V to 5V. The EEPROM is for saving configuration data when powered off. It has a default baud rate of 9600bps.

It outputs location, date and time information in form of NMEA codes.

Its specifications are as shown in Table 3.2

Table 3.2 NE06M GPS Module Specifications

S/No.	Parameters	Values		
		Min	**Typical**	**Max**
1	Power supply Voltage	2.7V	3.0V	3.3V
2	Supply Voltage USB	3.0V	3.3V	3.6V
3	Max Supply Current	N/A	N/A	67 mA
4	Input Pin Voltage	N/A	N/A	3.6V
5	DC current through any digital I/O pin	N/A	N/A	10 mA
6	Antenna Gain	N/A	N/A	50dB
7	Operating Temperatures	-40 C	N/A	85 C

3.3.5 TP4056 Battery Charging Module

The TP4056 is a complete constant-current/constant-voltage linear charger for single cell lithium-ion batteries.TP4056 features include current monitor, under voltage lockout, automatic recharge and two status pin to indicate charge termination and the presence of an input voltage

Table 3.3 TP4056 Specifications

S/No.	Parameter	Values
1.	Input Voltage	5V
2.	Charging cut-off voltage	4.2V ± 1%
3.	Maximum charge current	1000mA
4.	Battery over-discharge protection voltage	2.5V
5.	Battery overcurrent protection current	3A

3.3.6 Solar Panel CL-638WP

Used to charge the 18650 battery . It has a maximum voltage 0f 6V. It also has a maximum current of 0.63A. It's maximum power output is 3.8 w.

The module has an IP rating of IP65.

Table 3.4 Solar Panel Specifications

S/No.	Parameter	Values
1.	Brand Name	CCLAMP
2.	Model	CL-638WP
3.	Power	3.8W
4.	Max Voltage	6V
5.	Max Current	0.63A

3.3.7 Nodemcu *ESP8266*

The (Node Microcontroller Unit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. It's specifications are as shown in Table 3.1

The microcontroller has two main functions in this project. The first one is to act as interface between the GPS and ads1115 where both of their output values can be read and interpreted to valuable information. The other main function of the micro controller is uploading the processed and interpreted data on power status and location to an online database.

It is therefore used to communicate with two ADS115 and the GPS module serially. Not only does it carry out various functions on the data it has obtained from the GPS module to obtain the location in degrees but also it carries out calculations n the data retrieved from the ADC's to obtain the rms current of each phase.

Table 3.5 NodeMCU ESP8266 Specifications

S/No.	Parameters	Values
1	Operating Voltage	3.3V
2	Minimum Operating Voltage	2.58V
3	Maximum Operating Voltage	3.6V
4	Digital I/O Pins	11
5	Resolution ADC	10 bit
6	Max DC current I/O Pin	12mA
7	Clock speed	80/160 MHz
8	Length x Width	48mm x 26mm
9	Maximum Output Current	1A

3.3.8 Circuit Diagram

The general circuit diagram of the electrical device is as shown below .

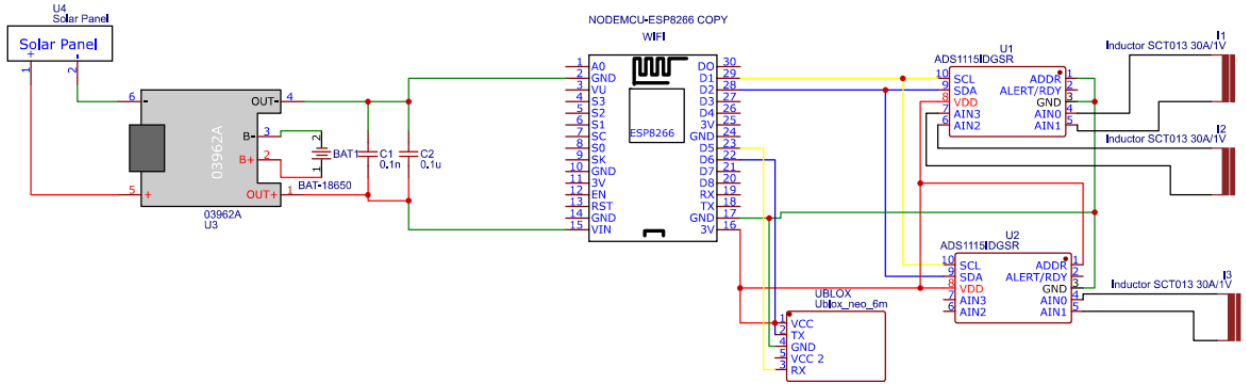


Figure 3.23 Circuit Diagram

The power source is the solar panel. it charges the 18650 Li-ion battery through the tp4056 battery charger module. the maximum current that the tp4056 can charge the battery with is 1A ,

The micro controller is also powered from the tp4056 out terminals. The out terminals are connected to the battery's terminals. The nodemcu micro controller is powered through the Vin port. It is connected to an inbuilt voltage regulator that regulates the voltage to 3.3V.

The micro controller powers the gps through one of its 3.3v pin. It's connected to the rx an tx pin of the gps module through pin d5 and d6 respectively, and effectively creating a serial interface of communication between the two components

The microcontroller also powers the 2 ADC's through its other 3.3v port. It is further connected to the ADC's through pin D1 and D2 which connect to port SCL and SDA respectively. Despite being connected to the same port the microcontroller can differentiate both ADC's through addressing.

Redphase CT is connected to pin A0 and A1 of ADC1.Blue phase CT is connected to pins A2 and A3 of ADC1 whose address is (0x48h)

Yellow phase CT is connected to pin A0 and A1 of ADC2 whose address is (0x49h).

CHAPTER 4 : RESULTS

4.1 To come up with an electronic device that detects when a power outage occurs and when power is restored on a distribution line

We were able to come up with an electronic device that detects when a power outage occurs and when its restored. The device is shown below:

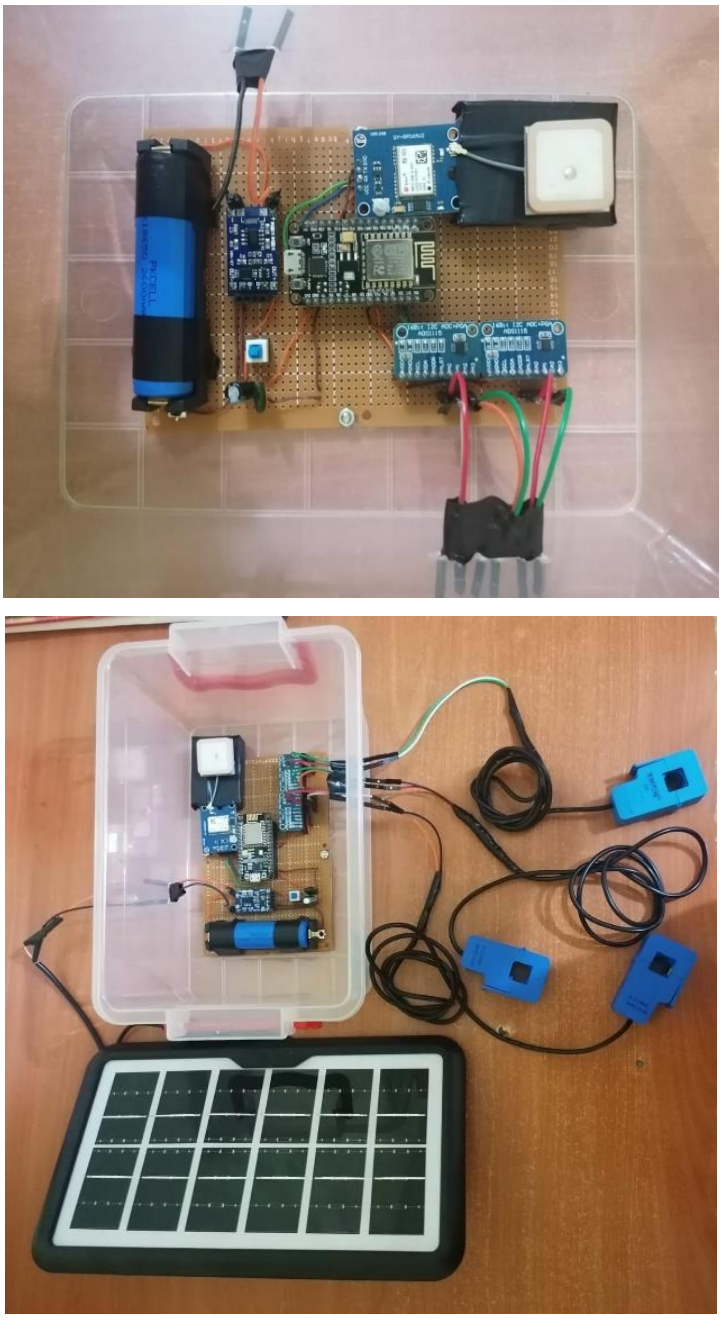


Figure 4.1 Final Electronic Device

4.2 To connect the electronic device wirelessly to a cloud database

These are the results on the serial monitor when the device is connected to the database. The values are successfully updated on the database.

The figure below shows successful connection of the electronic device to the online database .

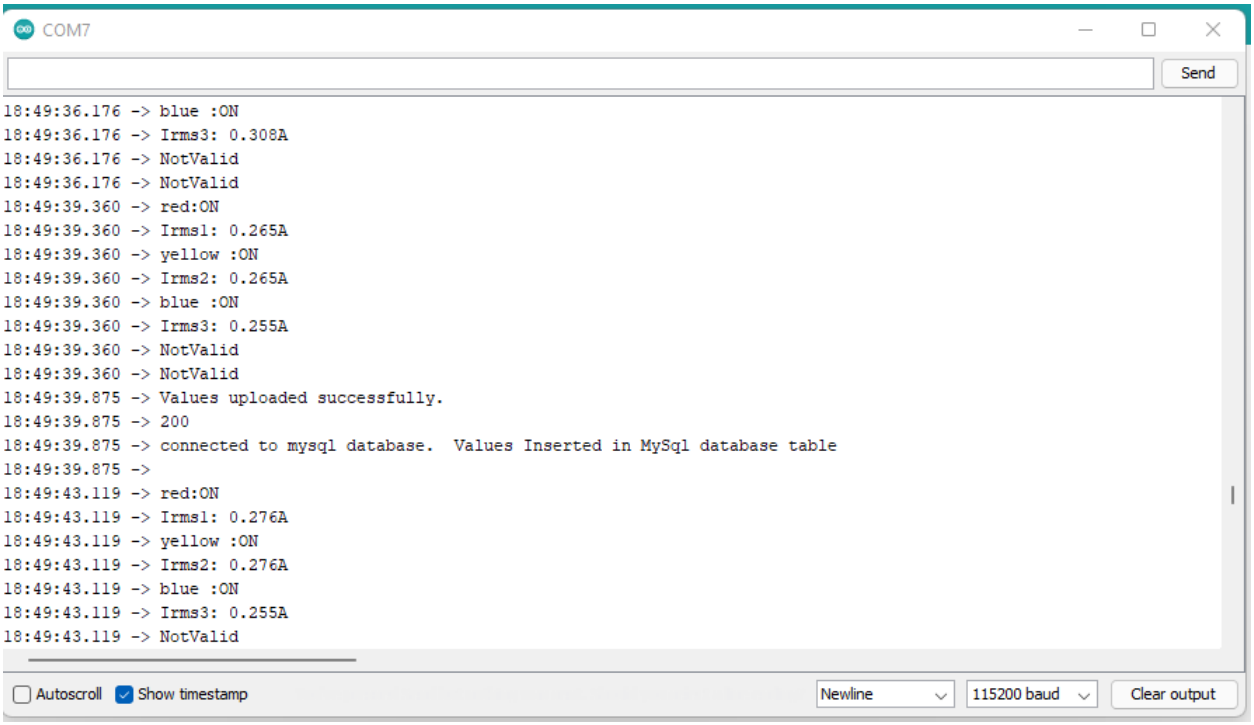


Figure 4.2 Serial Monitor Records

4.3 To create a cloud database that will store and record power outage information provided by the electronic device

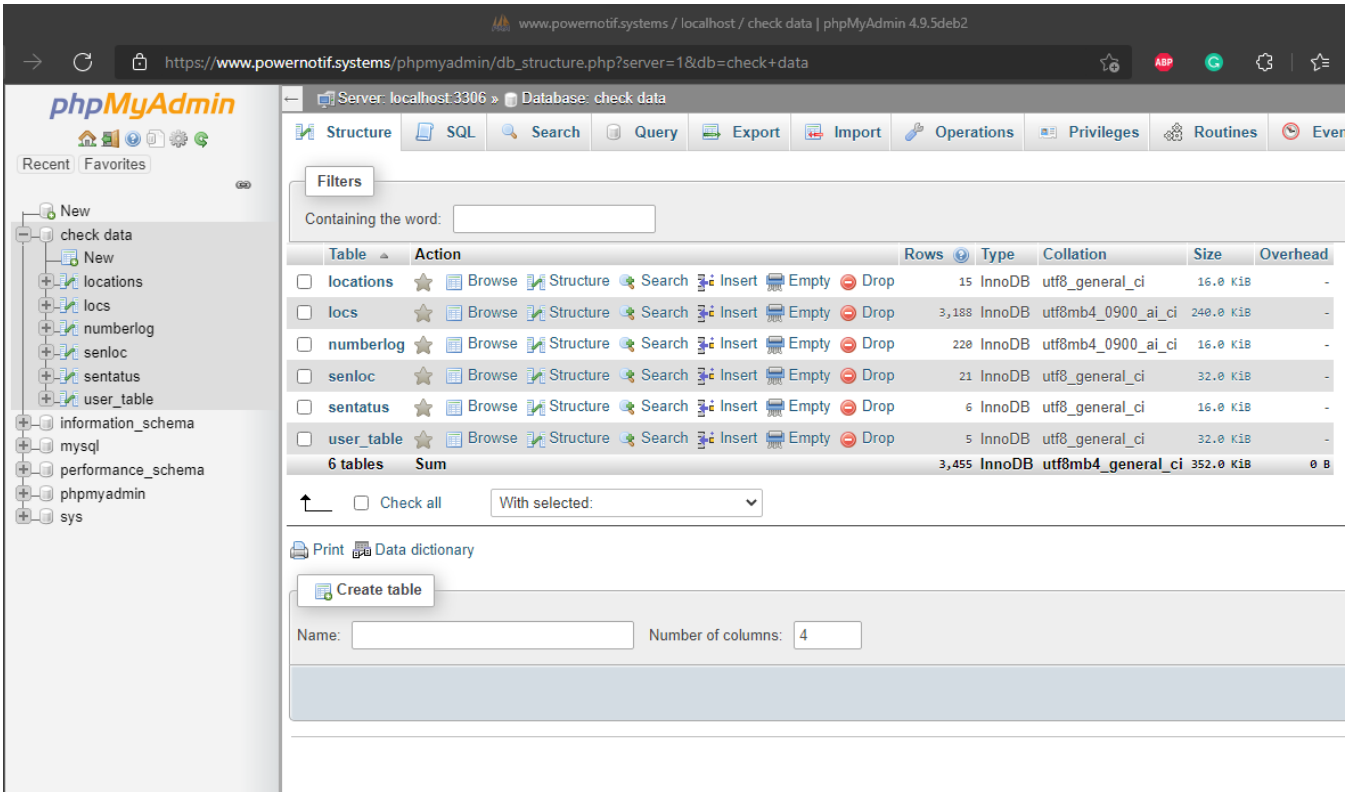


Figure 4.3 MySql Database

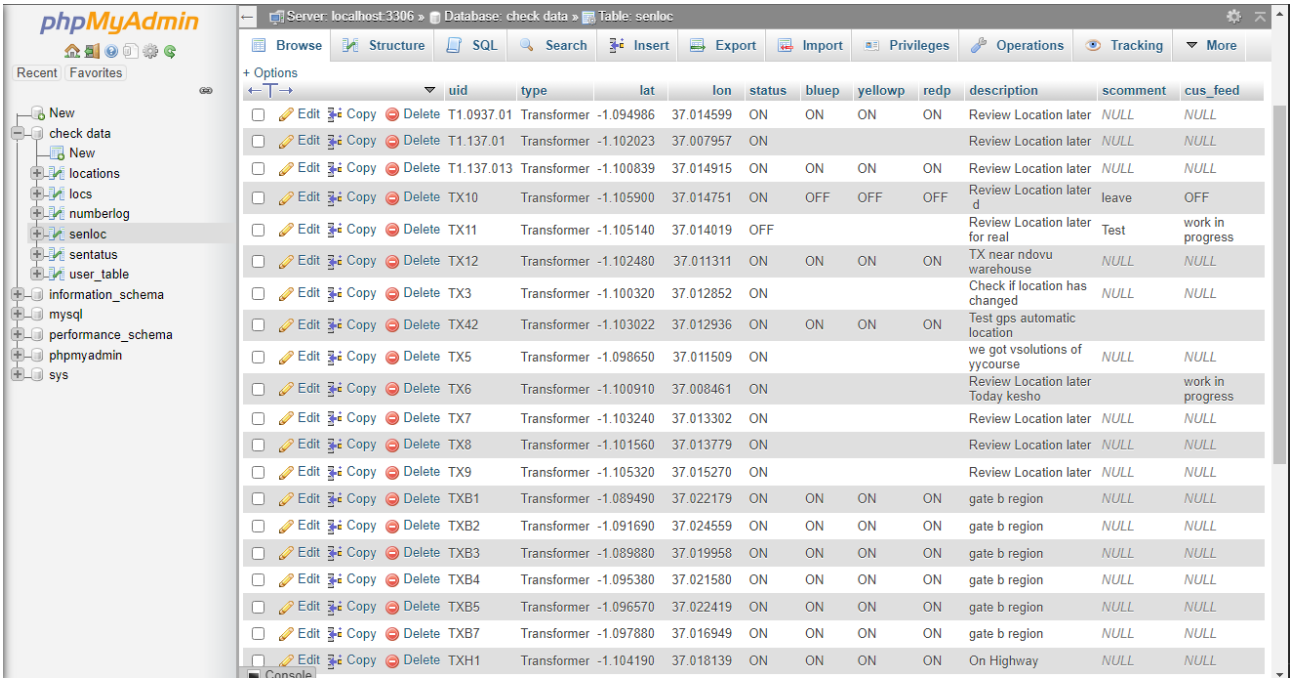


Figure 4.4 Senloc Table

4.3.1 User_Table

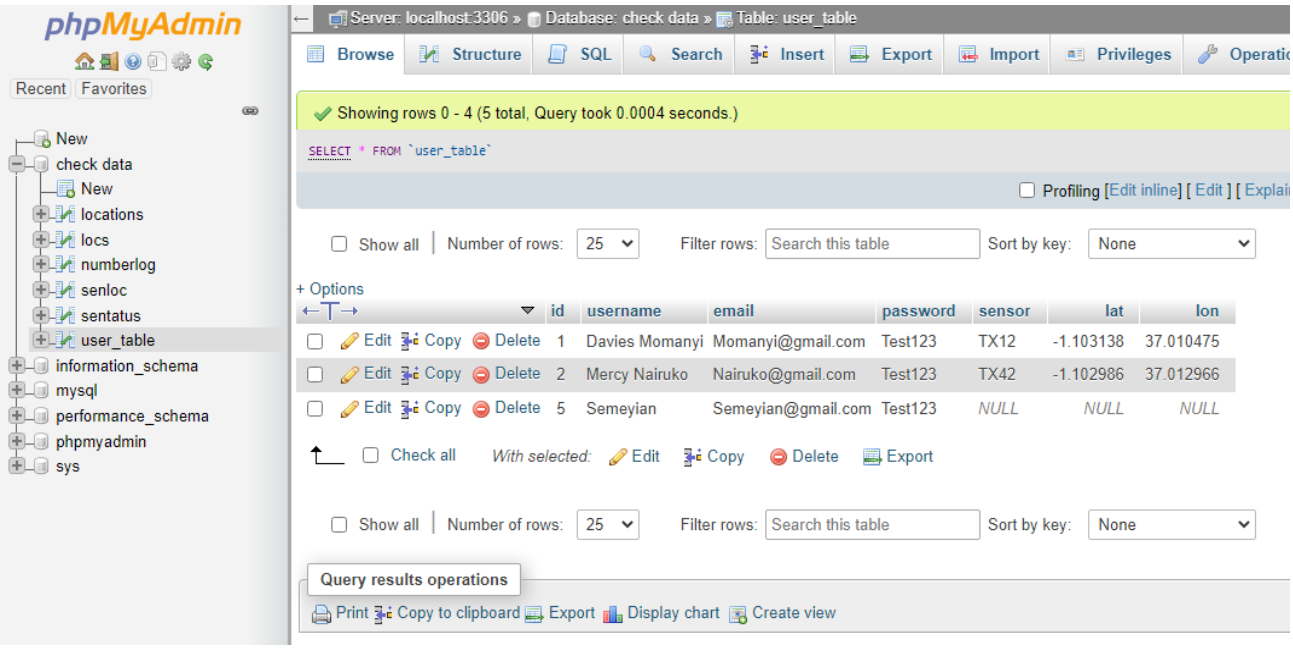


Figure 4.5 User Table

4.3.2 Sentatus table

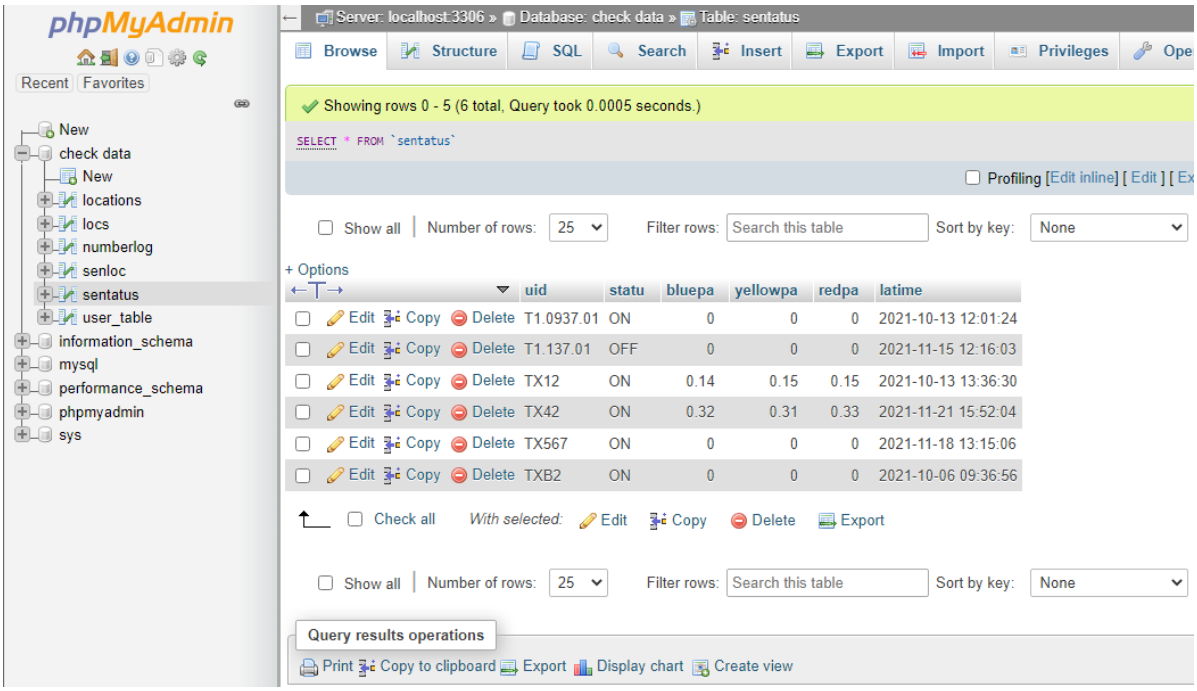


Figure 4.6 Sentatus Table

4.3.3 Locs Table

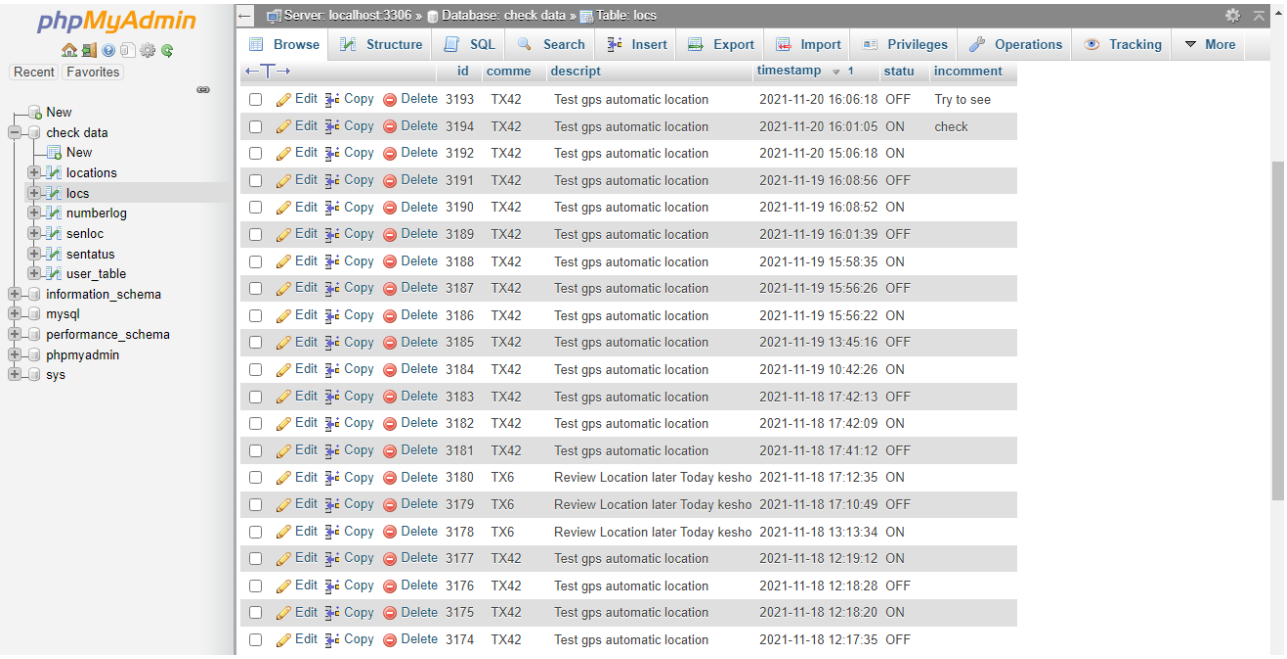


Figure 4.7 Locs Table

4.4 To create a cloud database that will store and record power outage information provided by the electronic device

The website enables one to view locations affected by power outages on a map, one can record a new sensor, can check if the sensor is on and a customer can also be allocated a new sensor.

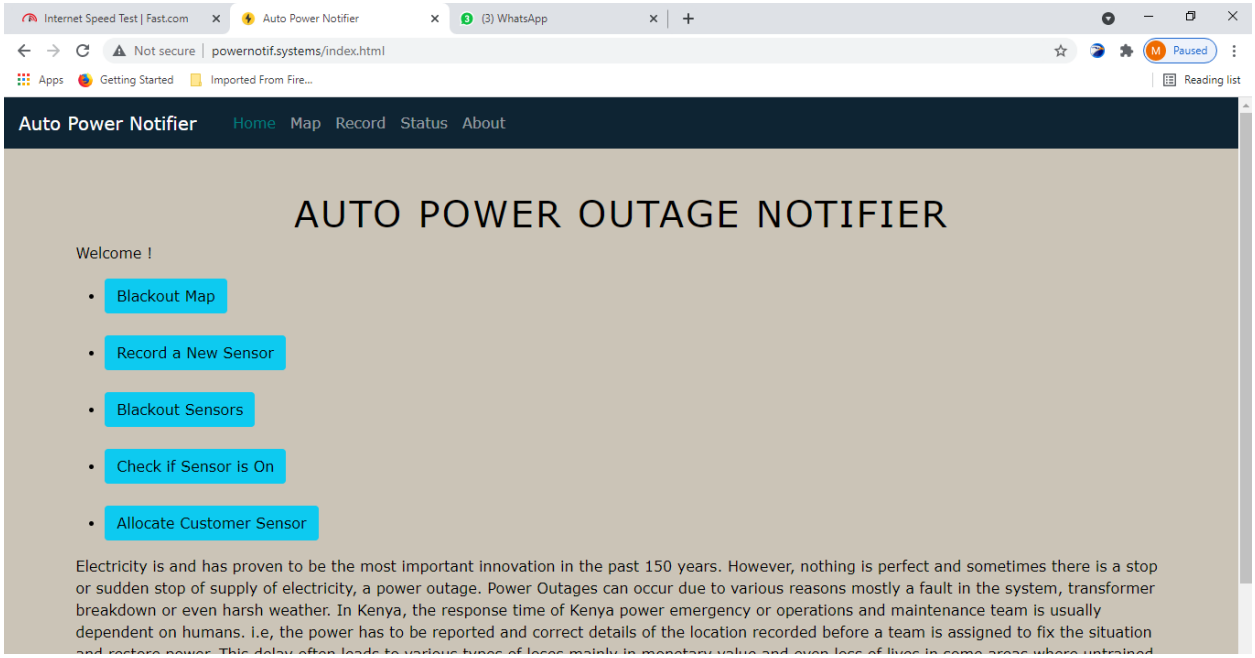
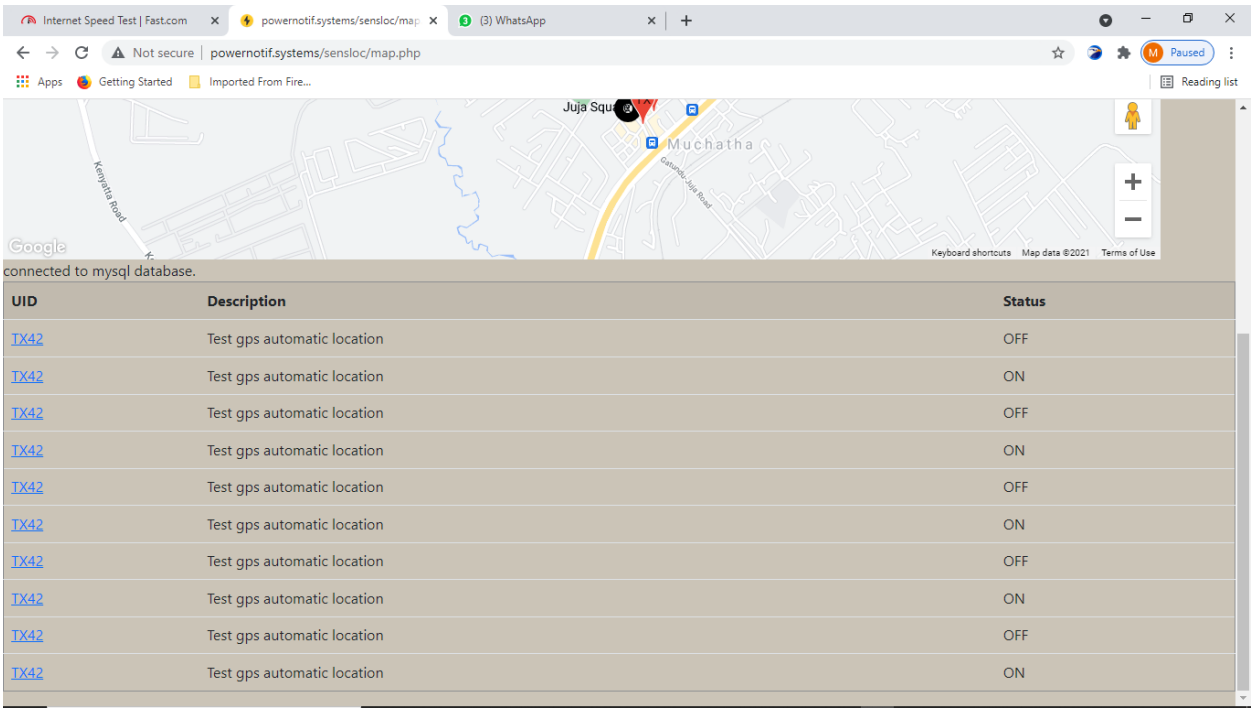


Figure 4.8 Homepage

When we click on the map page or blackout map, a map that shows the location of transformers where the sensors are placed, and the power outages that occur is displayed. The power outages are in black

Figure 4.9 Power Status Map

When a power outage occurs, the status is updated on a database as “off” while when there no power outage its on.



UID	Description	Status
TX42	Test gps automatic location	OFF
TX42	Test gps automatic location	ON
TX42	Test gps automatic location	OFF
TX42	Test gps automatic location	ON
TX42	Test gps automatic location	OFF
TX42	Test gps automatic location	ON
TX42	Test gps automatic location	OFF
TX42	Test gps automatic location	ON
TX42	Test gps automatic location	OFF
TX42	Test gps automatic location	ON

Figure 4.10 Map Immediate Notification Table

The map notification immediate npotification table shows the latest sensors whose status have changed even after the map has been loaded.

On the record page, one can record a new sensor as shown below, that is the transformer, location, status and description on location.

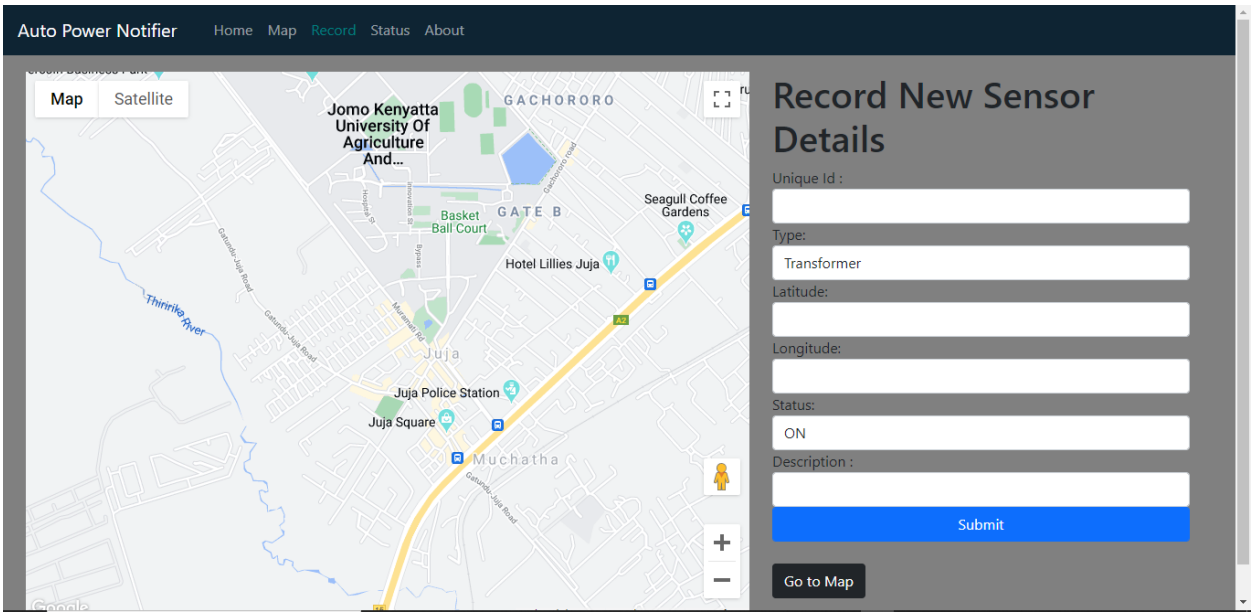


Figure 4.11 Recording a Sensor

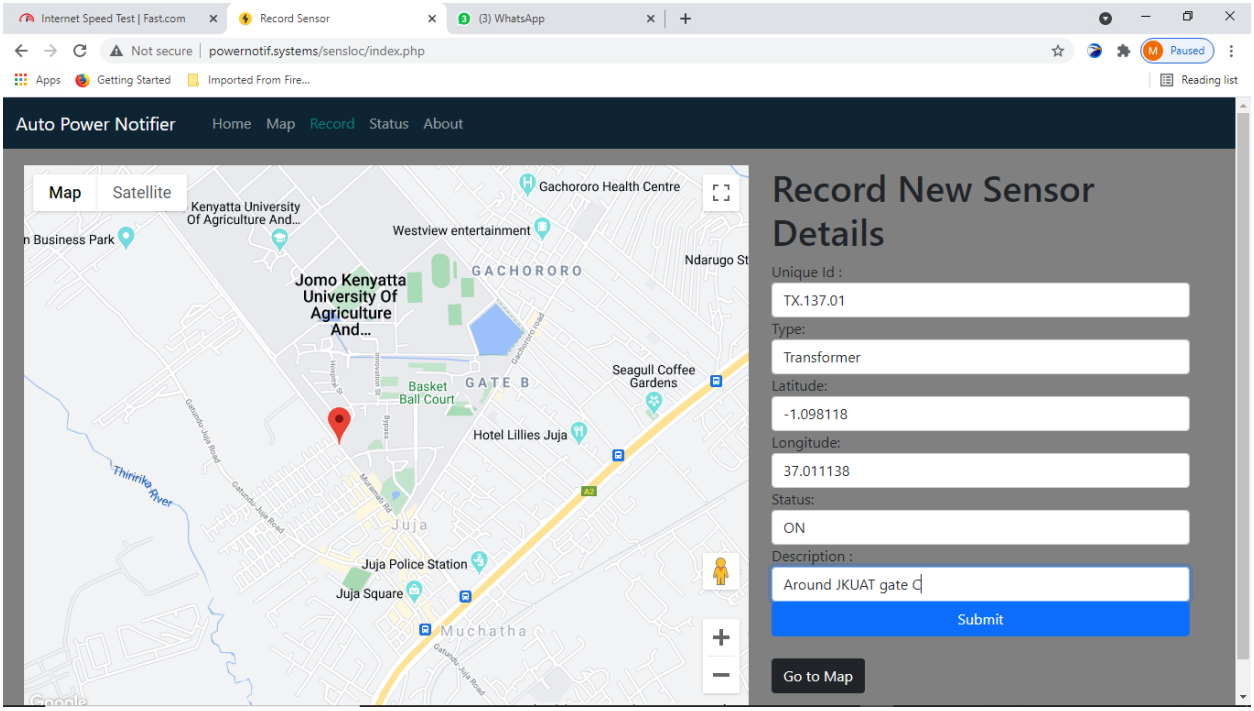


Figure 4.12 Recording a Sensor

The status is shown whether on or off and the feedback can also be given to customers especially when the company is working to fix the problem. Incomment section allows recording of the exact causes of power outage by the staff for future reference.

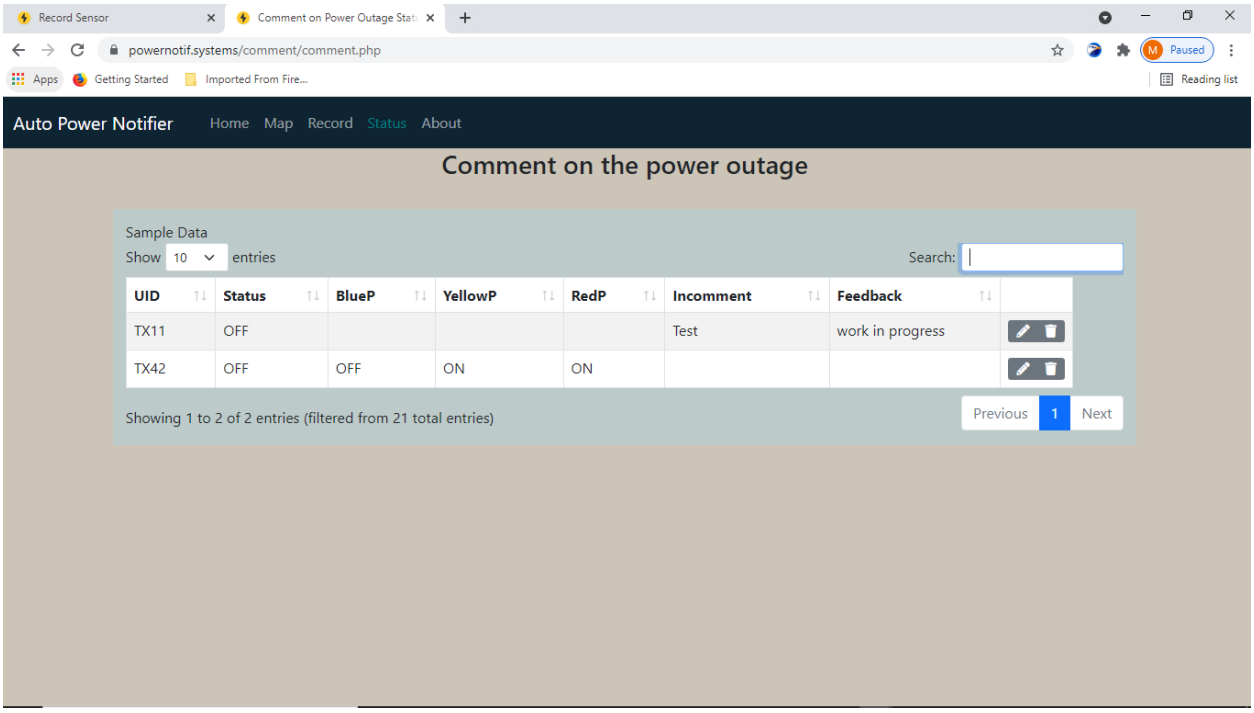


Figure 4.13 Comment Page

The webpage below shows the current sensor reading. It is also used to know the time that has elapsed since a sensor last communicated with the database. If the value is more than a few minutes then the sensor is off or has mal functioned

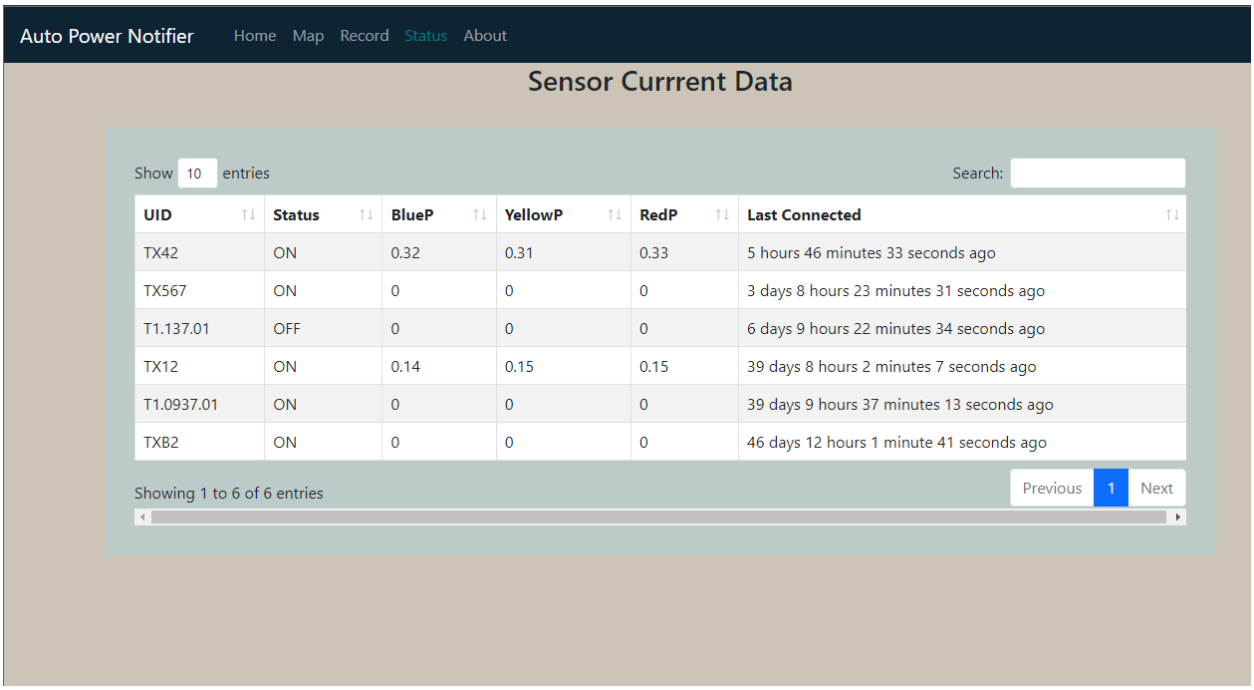


Figure 4.14 Sensor Current Status Page

The about section gives a brief description of the website.

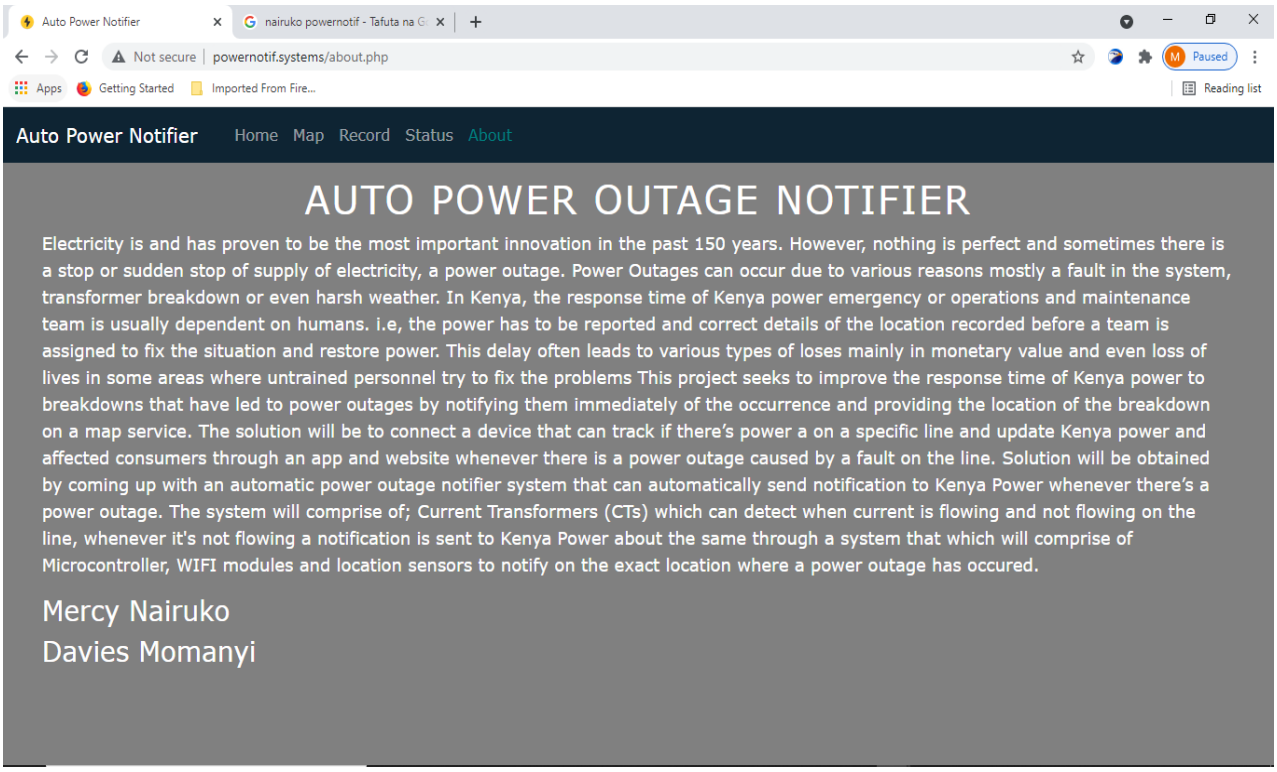


Figure 4.15 About Page

4.5 To create a mobile application that will enables feedback between consumers and the power distribution company

We created an app that enables feedback between customers and the distribution company. The app allows one to register their details as shown below. One can use these details to log in the app any time, once logged in the consumer can see their power status and their current location. The consumer can also request to be allocated a new sensor depending on their location. When the power status is 'OFF' a new field appears which the distribution uses to provide feedback to the customer and assure the customer that the power outage has been detected and is being handled .,

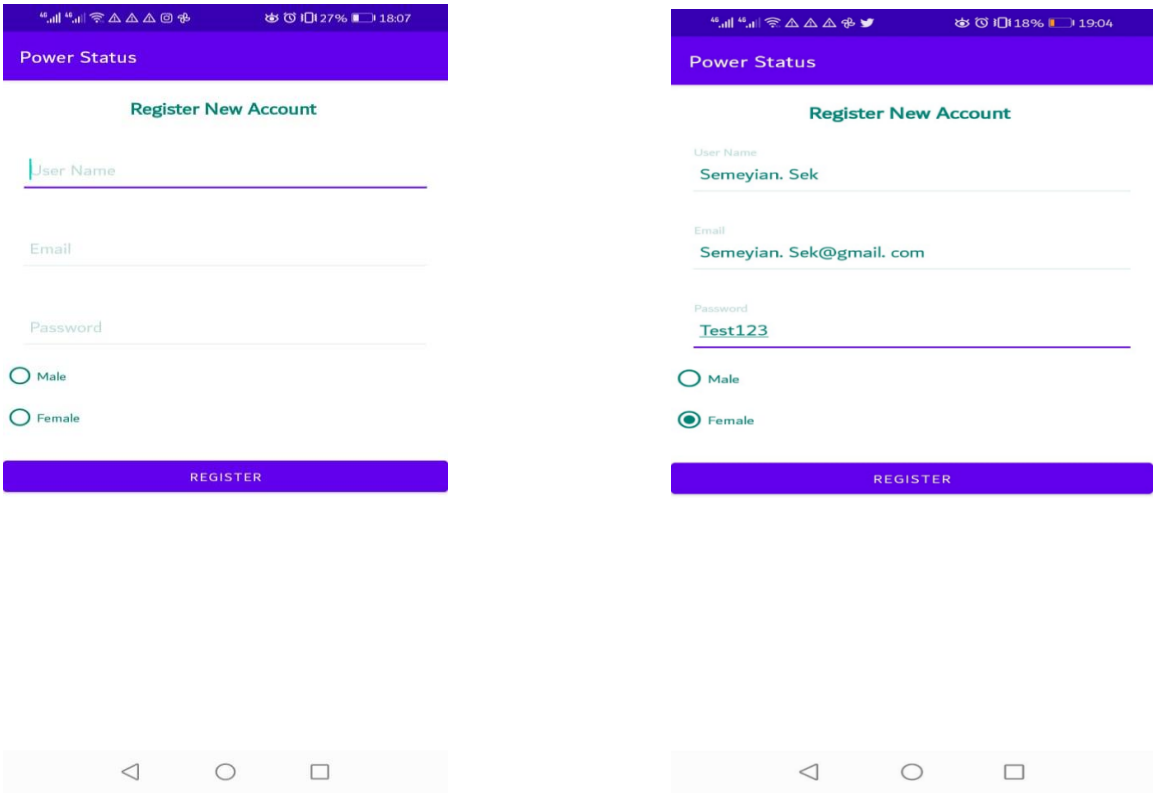


Figure 4.16 New User Registration

After registration the values are immediately and correctly inserted into the user_table.

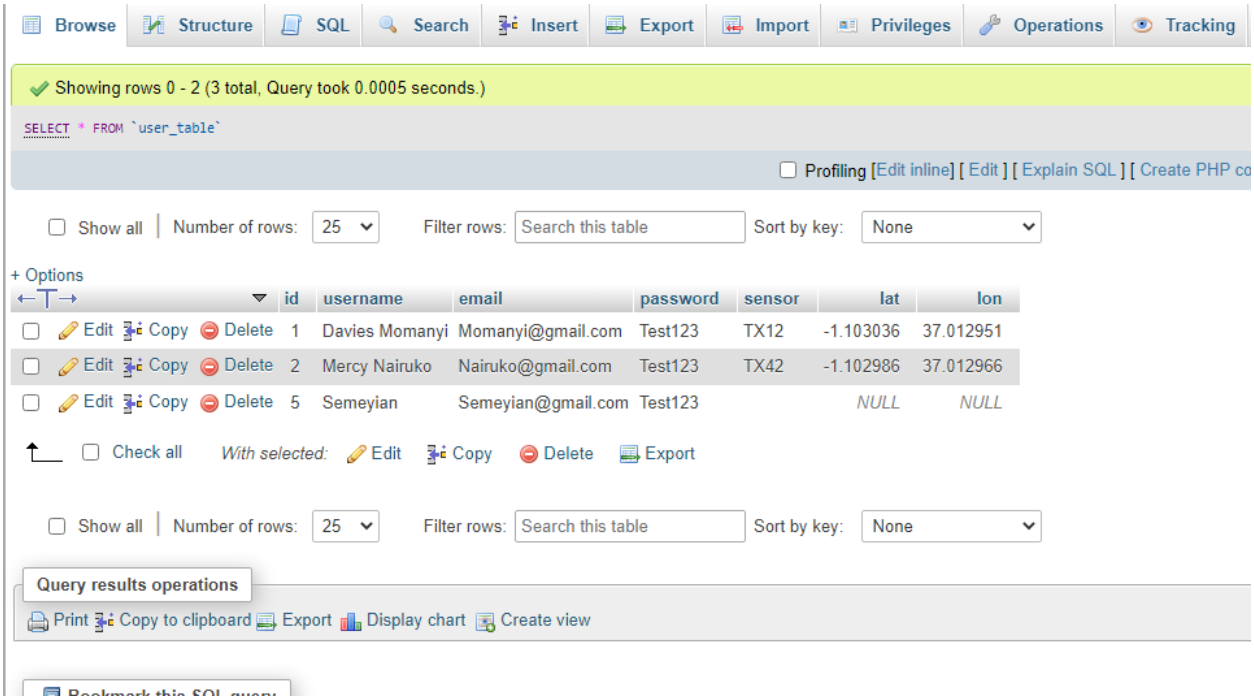


Figure 4.17 New User Database Values

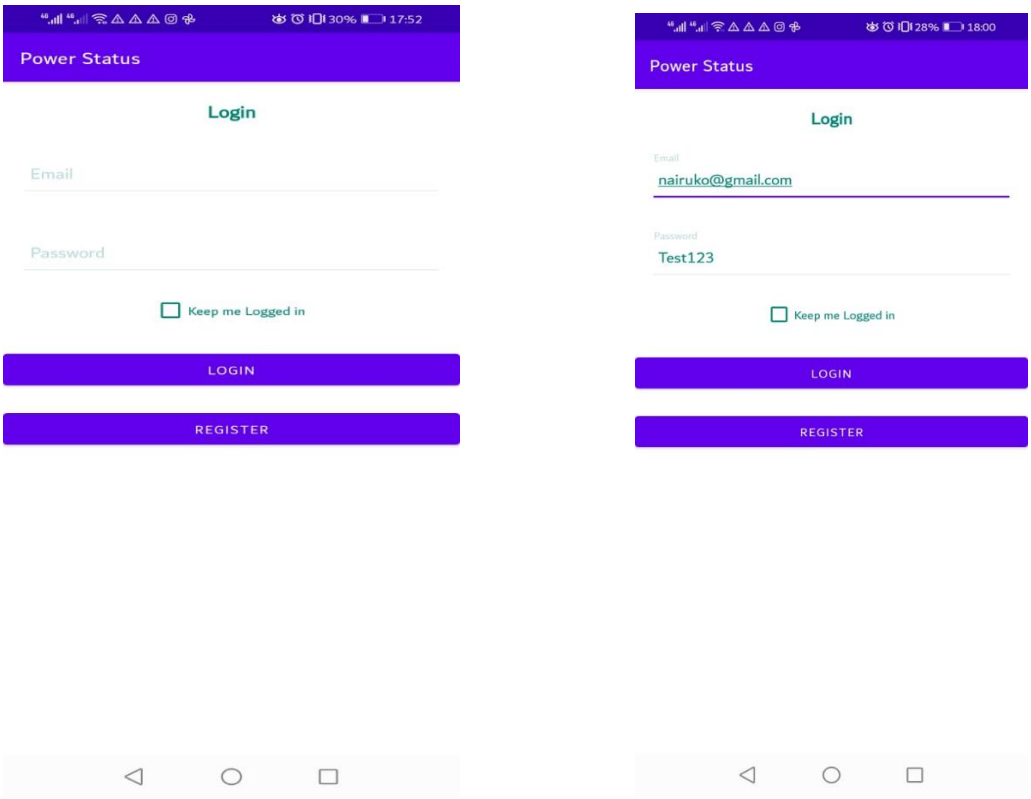


Figure 4.18 Login Page

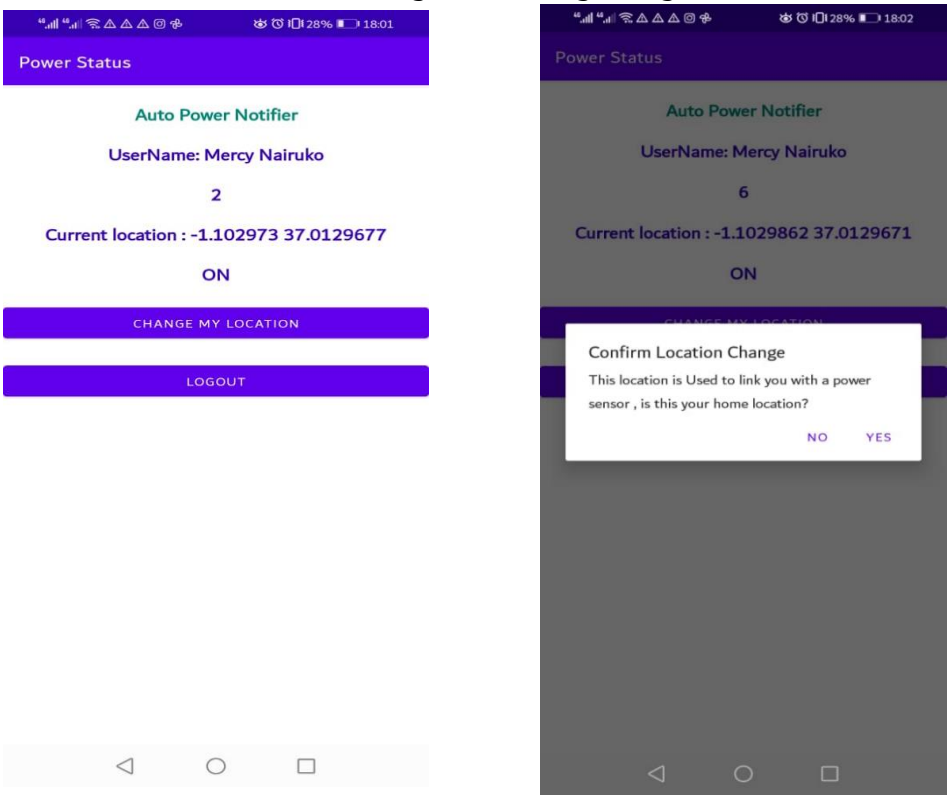


Figure 4.19 Request for Location Change

After a Customer asks for allocation of a sensor , it is shown in the map as shown in Figure 4.18

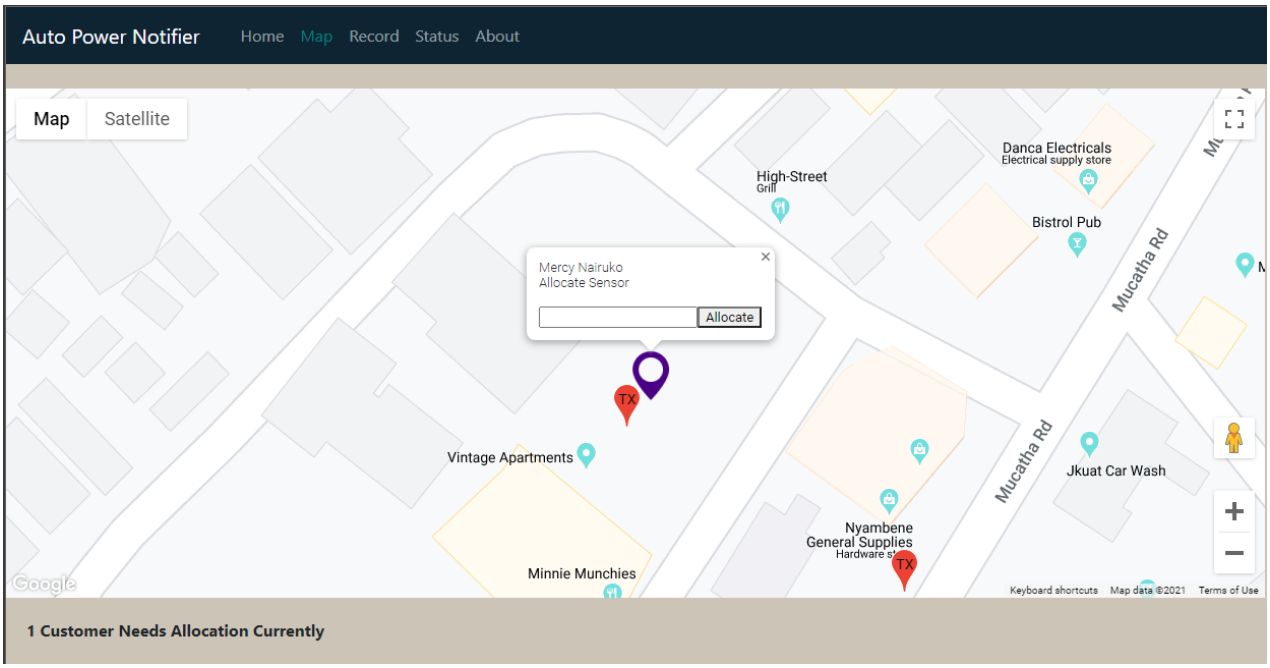


Figure 4.20 Allocating a Sensor

After Allocation the Feedback field on the app changes from pending Allocation to the current powerstatus of customer's home location

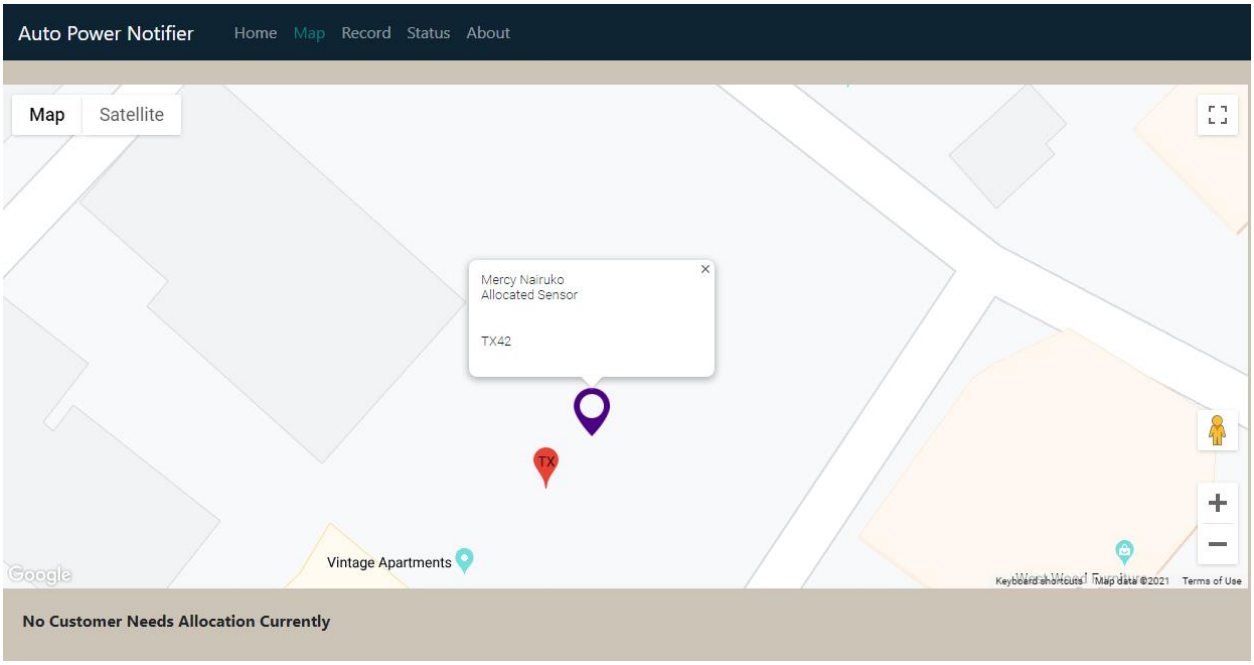


Figure 4.21 After Allocation

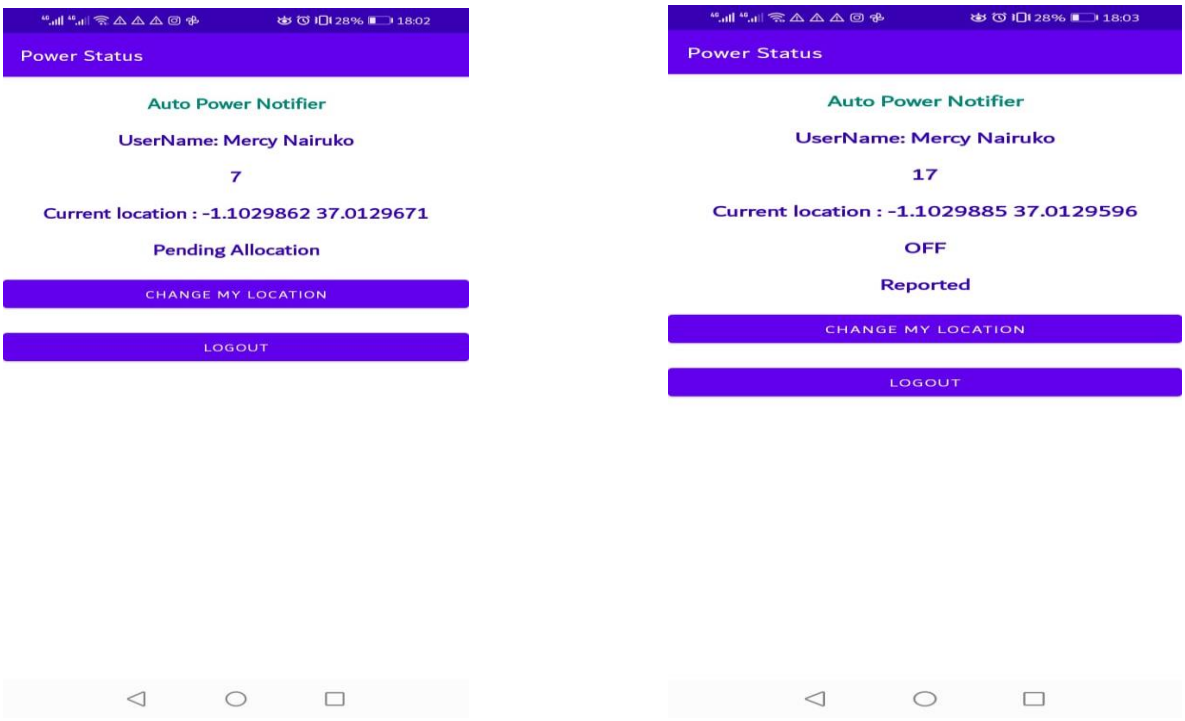


Figure 4.22 After Allocation on APP

Once the status is ON , the customer feedback textview is set to Invisible and does not appear on the page

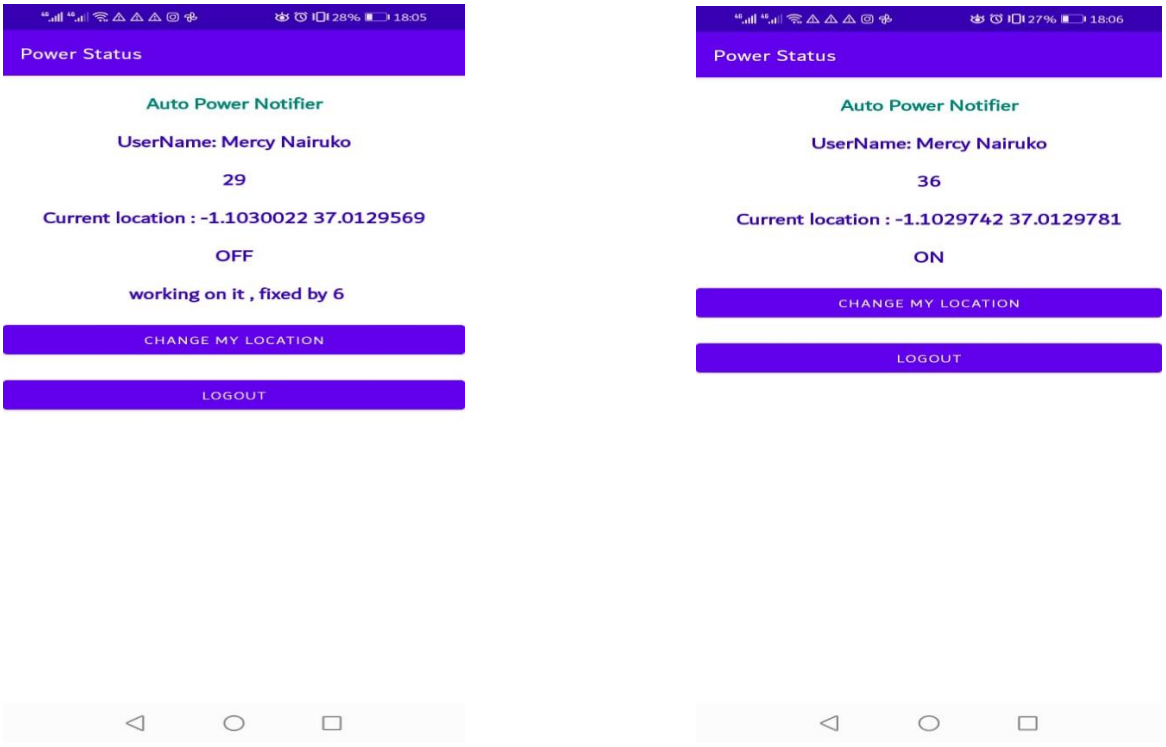


Figure 4.23 Change Of Status

CHAPTER 5 : CONCLUSION

5.1 CONCLUSION

As expected, not only does the system improve the response time to power outages by immediately detecting the location of power outages and displaying them on a map but also it ensures customer calm and satisfaction by enabling quick feedback to the customer.

It would therefore be very useful and should be adopted in power distribution companies in order to avoid the challenges of existing systems such as slow response time and too much human interaction through social media.

5.2 CHALLENGES

1. Testing of GPS can only be done outdoors for it to be able to locate the satellites and give the required location. It also takes time, about ten minutes to obtain a valid location.
2. The device requires an internet connection in order for it to upload the results on the power status to the website and update them on the database while giving customer feedback through the app.

5.3 RECOMMENDATION

1. The electronic device currently uses WI-FI to wirelessly connected to the database. However, Wi-Fi has a limited range which is not more than around 25 meters. It is recommended that LORAWAN is used instead as it has a range of almost 13Km
2. Introduction of projects and documentation from as early as the second year of study in order to make the work easier and more familiar when it comes to the final year project.

PROJECT TIMELINE

Timeline

	2021							
ACTIVITIES	MAY	JUNE	JULY	AUG	SEPT	OCT	NOV	DEC
Project abstract and Submission for approval.								
Proposal writing and review by the supervisor								
Proposal completion and submission								
Proposal presentation								
Project Documentation								
Project Implementation								
Final project presentation								

BUDGET

Budget

S/No.	Component	Quantity	Cost
1.	NodeMCU esp8266	1	Ksh. 1,400.00
2.	SCT001300	3	Ksh. 3,450.00
3.	GPS Module NEO 06M	1	Ksh. 800.00
4.	Capacitors	2	Ksh. 100.00
5.	Solar Panel	1	Ksh. 600.00
6.	ADS1115	2	Ksh 900.00
7.	18650 Battery	1	Ksh. 500.00
8.	Micro Usb Cable	2	Ksh. 300.00
9.	TP4056	1	Ksh. 150.00
10.	Casing	1	Ksh. 150.00
11.	Soldering Lead	1	Ksh. 150.00
12.	Printing	2	Ksh. 600.00
	TOTAL		Ksh. 8950

References

[1] Abotsi, A. (2016). Power Outages and Production Efficiency of Firms in Africa. International of Energy Economics Policy, 6(1), 98-104

[2] Department of Economics, Accounting and Finance, Jomo Kenyatta University of Agriculture and Technology, Kenya, "The Cost of Power Outages on Enterprise Performance in Kenya", 2020.

[3] Department of Economics, University of Navarra. Baker Institute for Public Policy, Rice University. Navarra Center for International Development (NCID), University of Navarra, "The Effect of Blackouts on Households' Electrification Status: evidence from Kenya", 2020.

[4] K. Bauman, A. Tuzhilin and R. Zaczynski, "Virtual Power Outage Detection Using Social Sensors", 2015.

[5] E. Karungu, "Harnessing Social Media Data for Outages Incident Reporting Case Study KPLC", 2019.

[6] J. Formea and J. Gadbury, "Improve power reliability through small-scale SCADA systems", 2014.

[7] S. Keere, "Automatic power meter reading based on Arduino micro-controller unit: case of the Kenya power and Lighting Company"(Thesis) Strathmore University, 2017.

[8] C. McNally, "Arduino Based Wireless Power Meter", 2010.

APPENDIX

Appendix A

Upload

sketch_oct31a

```
  if (count>=10)
  {
WebSerial.println(lng_str+lat_str);
//WebSerial.println("B: "+bluep+"R: "+redp);
    WiFiClient wifiClient;
  HTTPClient http;
  String bluepa= String(currentRMS3);
   String yellowpa= String(currentRMS2);
    String redpa= String(currentRMS1);
  postData= "bluepa="+bluepa+"&yellowpa="+yellowpa+"&redpa="+redpa+"&statusl="+statu+"&uidl="+uid; // sendval = datal
  String serv2 ="http://"+ip+"/dbwrite2.php";
  http.begin(wifiClient ,serv2);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");  //Specify content-type header
  int httpCode = http.POST(postData);   // Send POST request to php file and store server response code in variable named httpCode
  //Serial.println("Values are sendval = " + sendval );
  // if connection eatablished then do this
  if (httpCode == 200) { Serial.println("Values uploaded successfully."); Serial.println(httpCode);
    String webpage = http.getString();  // Get html webpage output and store it in a string
    Serial.println(webpage + "\n");
  } else {
    // if failed to connect then return and restart
    Serial.println(httpCode);
    Serial.println("Failed to upload values. \n");
    http.end();
    return;
  }
 count=0;
    }
  else
    {
      count=count+1;
```

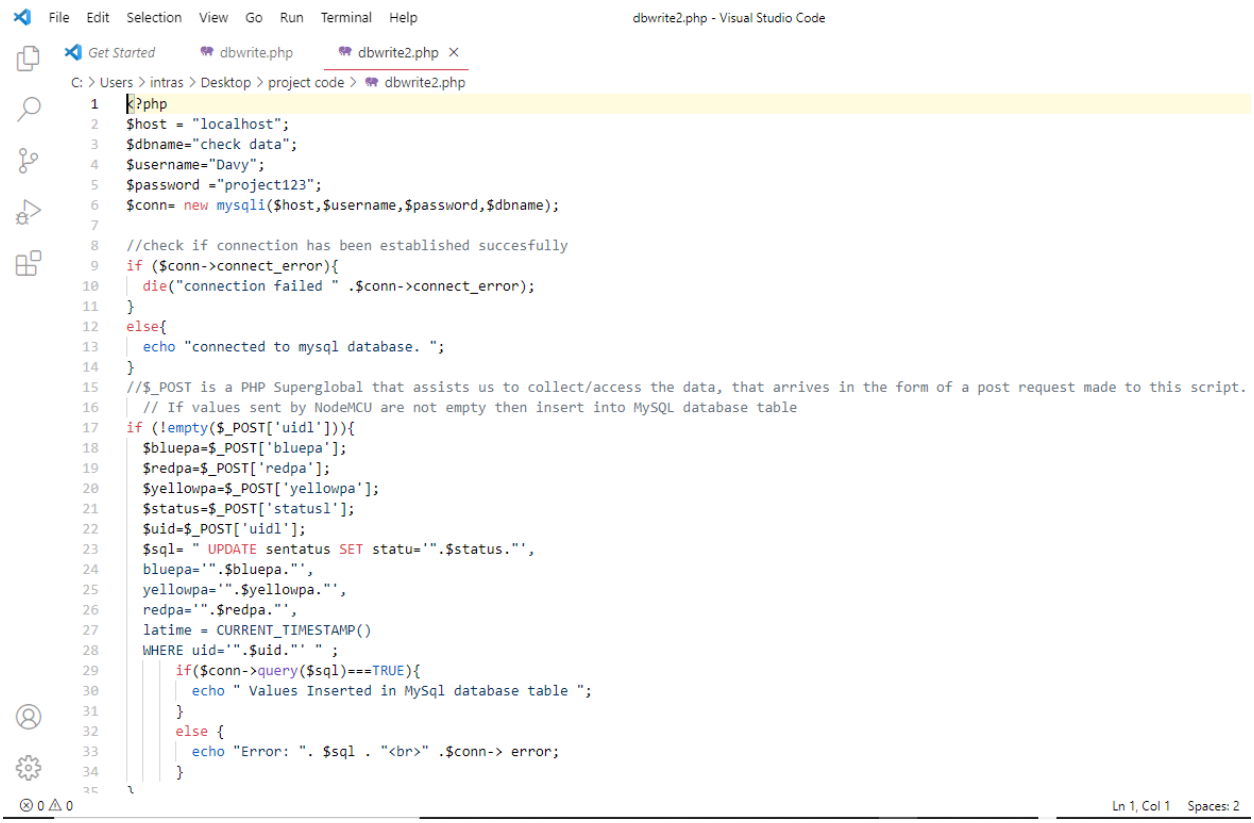
File Edit Selection View Go Run Terminal Help — dbwrite.php - Visual Studio Code

Get Started | dbwrite.php

C: > Users > intras > Desktop > project code > dbwrite.php

```
<?php
$host = "localhost";
$dbname="check data";
$username="Davy";
$password ="project123";
$conn= new mysqli($host,$username,$password,$dbname);

//check if connection has been established succesfully
if ($conn->connect_error){
  die("connection failed " .$conn->connect_error);
}
else{
  echo "connected to mysql database. ";
}
//$_POST is a PHP Superglobal that assists us to collect/access the data, that arrives in the form of a post request made to this script.
  // If values sent by NodeMCU are not empty then insert into MySQL database table
if (!empty($_POST['uidl'])){
  $bluep=$_POST['bluepl'];
  $redp=$_POST['redpl'];
  $yellowp=$_POST['yellowpl'];
  $status=$_POST['statusl'];
  $uid=$_POST['uidl'];
  $sql= " UPDATE senloc SET status='".$status."',
  bluep='".$bluep."',
  yellowp='".$yellowp."',
  redp='".$redp."'
  WHERE uid='".$uid."' " ;
      if($conn->query($sql)===TRUE){
        echo " Values Inserted in MySql database table ";
      }
      else {
        echo "Error: ". $sql . "<br>" .$conn-> error;
      }
}
$conn->close();
```

⊗ 0 ⚠ 0 — Ln 19, Col 25 Spaces: 2


File Edit Selection View Go Run Terminal Help
dbwrite2.php - Visual Studio Code
Get Started
dbwrite.php
dbwrite2.php
C: > Users > intras > Desktop > project code > dbwrite2.php
<?php
$host = "localhost";
$dbname="check data";
$username="Davy";
$password ="project123";
$conn= new mysqli($host,$username,$password,$dbname);
//check if connection has been established succesfully
if ($conn->connect_error){
die("connection failed " .$conn->connect_error);
}
else{
echo "connected to mysql database. ";
}
//$_POST is a PHP Superglobal that assists us to collect/access the data, that arrives in the form of a post request made to this script.
// If values sent by NodeMCU are not empty then insert into MySQL database table
if (!empty($_POST['uidl'])){
$bluepa=$_POST['bluepa'];
$redpa=$_POST['redpa'];
$yellowpa=$_POST['yellowpa'];
$status=$_POST['statusl'];
$uid=$_POST['uidl'];
$sql= " UPDATE sentatus SET statu='".$status."',
bluepa='".$bluepa."',
yellowpa='".$yellowpa."',
redpa='".$redpa."',
latime = CURRENT_TIMESTAMP()
WHERE uid='".$uid."' " ;
if($conn->query($sql)===TRUE){
echo " Values Inserted in MySql database table ";
}
else {
echo "Error: ". $sql . "
" .$conn-> error;
}
⊗ 0 ⚠ 0
Ln 1, Col 1 Spaces: 2

Appendix B

map.php

C: > Users > intras > Desktop > project code > sensloc > map.php

```
<!DOCTYPE html>
<html lang ="en">
<head>
  <meta charset ="utf-8" />
  <style type ="text/css">
    body{ font: normal 14px Verdana;}
    h1{ font-size: 24px;}
    h2{font-size:18px;}
    #sidebar {float:right; width:30px}
    #main{padding-right:15px;}
    .infowindow{width:220px}

  </style>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34M
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script async
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDnD7obafvgMFAlw8bqQb26FmF_yu07uEM&callback=initMap"></script>
  <script type="text/javascript">
  function makeRequest(url,callback){

    var request;
    if (window.XMLHttpRequest){
      request=new XMLHttpRequest();//Chrome , Safari , Firefox
    }
    else{
      request = new ActiveXObject("Microsoft.XMLHTTP");// IE6,IE5
    }
    request.onreadystatechange=function(){
      if (request.readyState==4 && request.status == 200){
        callback(request);
      }
    }
```

map.php

C: > Users > intras > Desktop > project code > sensloc > map.php

```
      }
    }
    request.open("GET",url,true);
    request.send();
  }
    var map;
    var center = {lat: -1.101465, lng:37.011867 };


  function initMap(){
    var mapOptions={
      zoom:15,
      center: center,
      mapId:'5028ea12062d5f68'
    }
    map=new google.maps.Map(document.getElementById("map"),mapOptions);
   //var marker = new google.maps.Marker({
     //map:map,
     //position:center,
  // });
  makeRequest('get_senslocations.php',function(data){
    var data= JSON.parse(data.responseText);
    for (var i=0;i<data.length;i++){
      displayLocation(data[i]);
    }
  });
  }


    function displayLocation(location){
      var content ='<div class="infowindow"<strong>' +location.uid+'</strong>'
      +'<br/><a href="/comment/comment.php?duid='+location.uid+'">Comment</a><br/>' + location.description +'</div>';
      if (parseInt(location.lat)==0){
```

map.php

C: > Users > intras > Desktop > project code > sensloc > map.php

```
67      if (parseInt(location.lat)==0){
68          var geocoder = new google.maps.Geocoder();
69        geocoder.geocode({'address':location.address},function(results,status){
70          if (status==google.maps.GeocoderStatus.OK){
71           var marker = new google.maps.Marker({
72             map:map,
73             position:results[0].geometry.location,
74             title:location.name
75           } );
76
77      google.maps.infowindow.addListener('click',function(){
78          infowindow.setContent(content);
79           infowindow.open(map,marker);
80        });
81          }
82        });
83      }
84     else{
85       var infowindow = new google.maps.InfoWindow({
86         content :'<div class="infoWindow"<strong> Hello There</strong></div>'
87       });
88
89      let position1 ={lat:parseFloat(location.lat),lng:parseFloat(location.lon)};
90      console.log(position1);
91          console.log(location.uid);
92          if (location.status=="OFF"){
93     const marker= new google.maps.Marker({
94           map:map,
95          position:position1,
96
97          icon: {
98         path: google.maps.SymbolPath.CIRCLE,
99         scale: 10,
```

map.php

C: > Users > intras > Desktop > project code > sensloc > map.php

```
98          path: google.maps.SymbolPath.CIRCLE,
99          scale: 10,
100       },
101          label:"TX1",
102         title:location.uid,
103
104       });
105       google.maps.event.addListener(marker, 'click', function() {
106   infowindow.setContent(content);
107   infowindow.open(map,marker);
108   });
109     }
110     else{
111       const marker= new google.maps.Marker({
112             map:map,
113            position:position1,
114            label:"TX",
115           title:location.uid,
116
117         });
118         google.maps.event.addListener(marker, 'click', function() {
119     infowindow.setContent(content);
120     infowindow.open(map,marker);
121     });
122     }
123
124       }
125
126
127     }
128
129
130       </script>
```

map.php

C: > Users > intras > Desktop > project code > sensloc > map.php

```
    }


    </script>
    <script type="text/javascript">
    $(document).ready(function(){
      refreshTable();
    }
  );
  function refreshTable(){
    $('#tableHolder').load('status.php',function(){
      setTimeout(refreshTable,10000);
    });
  }
    </script>
    </head>
    <body >

      <section id="sidebar">
        <div id="directions_panel">

        </div>
      </section>
  <h1>Blackouts are in black</h1>
    <section id="main">
      <div id="map" style="width: 95%; height:500px;"></div>
      </section>

      <div id="tableHolder"></div>
      </body>
      </html>

```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title> Record Sensor </title>
  <style type="text/css">
  a.button{
    -webkit-appearance:button;
    -moz-appearance:button;
    appearance: button;
    text-decoration:none;
    color:white;
    padding: 15px 25px 25px 25px;
    background-color:teal;
  }
  #form{
    padding: 0px 25px 25px 25px;
    float:right;
  }
  #wrapper{
     display:flex;
   }
  </style>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34M
  <script async
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDnD7obafvgMFAlw8bqQb26FmF_yu07uEM&callback=initMap"></script>
  <script type="text/javascript">
  function makeRequest(url,callback){

    var request;
    if (window.XMLHttpRequest){
      request=new XMLHttpRequest();//Chrome , Safari , Firefox
    }
    else{
```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```
    else{
      request = new ActiveXObject("Microsoft.XMLHTTP");// IE6,IE5
    }
    request.onreadystatechange=function(){
      if (request.readyState==4 && request.status == 200){
        callback(request);
      }
    }
    request.open("GET",url,true);
    request.send();
  }
    var map;
    var center = {lat: -1.101465, lng:37.011867 };


function initMap(){
  var mapOptions={
    zoom:15,
    center: center,
    mapId:'5028ea12062d5f68'
  }
  map=new google.maps.Map(document.getElementById("map"),mapOptions);
 //var marker = new google.maps.Marker({
   //map:map,
   //position:center,
// });
 var markers=[];
var getMarkerUniqueId= function(lat, lng) {
            return Math.abs(lat) + '' + Math.abs(lng);
      };
var getLatLng = function(lat, lng) {
                  return new google.maps.LatLng(lat, lng);
              };
```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```
            return new google.maps.LatLng(lat, lng);
        };

        var addMarker = google.maps.event.addListener(map, 'click', function(e) {
            var lat = e.latLng.lat(); // lat of clicked point
            var lng = e.latLng.lng(); // lng of clicked point
            var markerId = getMarkerUniqueId(lat, lng); // an that will be used to cache this marker in markers object.
            var marker = new google.maps.Marker({
                position: getLatLng(lat, lng),
                map: map,
                animation: google.maps.Animation.DROP,
                id: 'marker_' + markerId,
                html: "    <div id='info_"+markerId+"'>\n" +
                "        <table class=\"map1\">\n" +
                "            <tr>\n" +
                "                <td><a>Description:</a></td>\n" +
                "                <td><textarea  id='manual_description' placeholder="+lat+"></textarea></td></tr>\n" +
                "            <tr><td></td><td><input type='button' value='Save' onclick='saveData("+lat+","+lng+")'/></td></tr>\n" +
                "        </table>\n" +
                "    </div>"
            });
            markers[markerId] = marker; // cache marker in markers object
            bindMarkerEvents(marker); // bind right click event to marker
            bindMarkerinfo(marker); // bind infowindow with click event to marker
            var lat1=lat.toFixed(6);
            var lng1=lng.toFixed(6);
            var lat2=lat.toFixed(2);
            var lng2=lng.toFixed(2);
            var markerId2 = getMarkerUniqueId(lat2, lng2);
            document.getElementById('uid').value = "T"+markerId2;

            document.getElementById('lat').value = lat1;
            document.getElementById('lon').value = lng1;
```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```
                    document.getElementById('lat').value = lat1;
                    document.getElementById('lon').value = lng1;
                });
                var bindMarkerinfo = function(marker) {
                google.maps.event.addListener(marker, "click", function (point) {
                    var markerId = getMarkerUniqueId(point.latLng.lat(), point.latLng.lng()); // get marker id by using clicked point's coordinat
                    var marker = markers[markerId]; // find marker
                    infowindow = new google.maps.InfoWindow();
                    infowindow.setContent(marker.html);
                    infowindow.open(map, marker);
                    // removeMarker(marker, markerId); // remove it
                });
            };
            var bindMarkerEvents = function(marker) {
            google.maps.event.addListener(marker, "rightclick", function (point) {
                var markerId = getMarkerUniqueId(point.latLng.lat(), point.latLng.lng()); // get marker id by using clicked point's coordinate
                var marker = markers[markerId]; // find marker
                removeMarker(marker, markerId); // remove it
            });
        };
        var removeMarker = function(marker, markerId) {
        marker.setMap(null); // set markers setMap to null to remove it from map
        delete markers[markerId]; // delete marker instance from markers object
    };



}


```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```


    </script>
</head>
<body>
  <div id="wrapper">

    <div id="map" style="width: 65%; height:600px;"></div>

  <div id="form">
    <h1 > Record New Sensor Details </h1>
    <form action ="insert.php" method ="post">
      <div class="form-group">
        <label for="uid">Unique Id :</label>
        <input type="text" name="uid" id="uid" class="form-control" required>
      </div>
      <div class="form-group">
        <label for="type">Type:</label>
        <select name="type" id="type" class="form-control">
          <option value = "Transformer">Transformer</option>
          <option value = "Line">Line</option>
        </select>
      </div>
      <div class="form-group">
        <label for="lat">Latitude:</label>
        <input type="number" name="lat" id="lat" step="any" class="form-control">
      </div>
      <div class="form-group">
        <label for="lon">Longitude:</label>
        <input type="number" name="lon" id="lon" step="any" class="form-control">
      </div>
      <div class="form-group" >
```

index.php

C: > Users > intras > Desktop > project code > sensloc > index.php

```
        <label for="lon">Longitude:</label>
        <input type="number" name="lon" id="lon" step="any" class="form-control">
      </div>
      <div class="form-group" >
        <label for="status">Status:</label>
        <select name="status" id="status" class="form-control">
          <option value = "ON">ON</option>
          <option value = "OFF">OFF</option>
        </select>
      </div>
      <div class="form-group">
        <label for="desc">Description :</label>
        <input type="desc" name="desc" id="desc" class="form-control">
      </div>
      <button type="submit" class="btn btn-outline-primary form-control">Submit</button>
    </form>
    <a href="/maps/map.html" class = "button">Allocate later </a>
  </div>
  </div>
</body>
</html>
```

index.php insert.php

C: > Users > intras > Desktop > project code > sensloc > insert.php

```
<!DOCTYPE html>
<html>
<head>
  <title>Confirm Record Sensor </title>
  <style type="text/css">
  a.button{
    -webkit-appearance:button;
    -moz-appearance:button;
    appearance: button;
    text-decoration:none;
    color:white;
    padding: 15px 25px;
    background-color:teal;
  }
  </style>
</head>
<body>
  <div>
    <center>
      <?php
      $host = "localhost";
      $dbname="check data";
      $username="Davy";
      $password ="project123";
      $conn= new mysqli($host,$username,$password,$dbname);
      //checking if connection has been established
      if ($conn->connect_error){
        die("connection failed " .$conn->connect_error);
      }
      else{
        echo "connected to mysql database. ";
      }
      $uid=$_REQUEST['uid'];
```


index.php
insert.php
C: > Users > intras > Desktop > project code > sensloc > insert.php
$uid=$_REQUEST['uid'];
$type=$_REQUEST['type'];
$lat=$_REQUEST['lat'];
$lon=$_REQUEST['lon'];
$status=$_REQUEST['status'];
$desc=$_REQUEST['desc'];
$sql= "INSERT IGNORE INTO senloc (uid,type,lat,lon,status,bluep,redp,yellowp,description)
VALUES ('$uid','$type','$lat','$lon','$status','$status','$status','$status','$desc')";
if(mysqli_query($conn,$sql)){
echo "<h3>data has been successfully stored</h3>";
echo nl2br("\n$uid\n "
. "$lat\n $lon\n $status \n $desc");
}
else{
echo "ERROR SORRY $sql."
.mysqli_error($conn);
}
// To Close the Connection
?>
</br>
Record a Sensor

</body>
</html>

Appendix C

comment.php

C: > Users > intras > Desktop > project code > comment > comment.php

```
<html>
<head>
  <title>Comment on Power Outage Status </title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></script>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />
  <script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
  <script src="https://cdn.datatables.net/1.10.23/js/dataTables.bootstrap4.min.js"></script>
  <link rel="stylesheet" href="https://cdn.datatables.net/1.10.23/css/dataTables.bootstrap4.min.css" />
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
  <script src="https://markcell.github.io/jquery-tabledit/assets/js/tabledit.min.js"></script>
</head>
</body>
<div class="container" style="width: 95%;">
  <h3 align="center">Comment on the power outage </h3>
  <br/>
  <div class="panel panel default">
    <div class="panel-heading">Sample Data</div>
    <div class="panel-body">
      <div class="table-responsive">
        <table id="blacksites" class="table table-bordered table-striped">
          <thead>
            <tr>
              <th>UID</th>
              <th>Status</th>
              <th>BlueP</th>
              <th>YellowP</th>
              <th>RedP</th>
              <th>Incomment</th>
              <th>Feedback</th>

            </tr>
          </thead>
          <tbody></tbody>
```

Explorer (Ctrl+Shift+E)

C: > Users > intras > Desktop > project code > comment > comment.php

```
          <tbody></tbody>
        </table>
      </div>
    </div>
  </div>
</div>
<br/>
<br/>
</body>
</html>
<script type="text/javascript" language="javascript">
var urlstring= window.location.href;
var url=new URL(urlstring);
var duid =url.searchParams.get("duid");
console.log("duid");
var fetch= "fetch.php"
if(duid!=null)
{
  fetch="fetch.php?duid="+duid;
}
console.log(fetch);
$(document).ready(function(){
  var dataTable=$('#blacksites').DataTable(
    {
      "processing":true,
      "serverSide":true,
      "order":[],
      "ajax":{
        url:fetch,
        type:"POST"
      }

    });
```

comment.php

C: > Users > intras > Desktop > project code > comment > comment.php

```
    });
  $('#blacksites').on('draw.dt',function(){
    $('#blacksites').Tabledit({
      url:'action.php',
      dataType:'json',
      columns:{
        identifier:[0,"uid"],
        editable:[[1,"status"],[5,"scomment"],[6,"cus_feed"]]
      },

      restoreButton:false,
      onSuccess:function(data,textStatus,jqXHR)
      {
        if(data.action=='delete')
        {
          $('#'+data.uid).remove();
          $('#blacksites').DataTable().ajax.reload();
        }
      }
    });

  });
  setInterval( function () {

        $('#blacksites').DataTable().ajax.reload( null, false ); // user paging is not reset on reload
  }, 20000 );
});
</script>

```

Explorer (Ctrl+Shift+E)

comment2.php

C: > Users > intras > Desktop > project code > comment > comment2.php

```
<html>
<head>
  <title>Sensor Status  </title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></script>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />
  <script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
  <script src="https://cdn.datatables.net/1.10.23/js/dataTables.bootstrap4.min.js"></script>
  <link rel="stylesheet" href="https://cdn.datatables.net/1.10.23/css/dataTables.bootstrap4.min.css" />
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
  <script src="https://markcell.github.io/jquery-tabledit/assets/js/tabledit.min.js"></script>
</head>
</body>
<div class="container" style="width: 95%;">
  <h3 align="center">Sensor Currrent Data </h3>
  <br/>
  <div class="panel panel default">
    <div class="panel-heading">Sensor Data</div>
    <div class="panel-body">
      <!--<div class="table-responsive">-->
      <div >
        <table id="blacksites" class="table table-bordered table-striped">
          <thead>
            <tr>
              <th>UID</th>
              <th>Status</th>
              <th>BlueP</th>
              <th>YellowP</th>
              <th>RedP</th>
              <th>Last Connected</th>

            </tr>
          </thead>
          <tbody></tbody>
```

comment.php × comment2.php ×

C: > Users > intras > Desktop > project code > comment > comment2.php

```
                <tbody></tbody>
              </table>
            </div>
          </div>
        </div>
      </div>
      <br/>
      <br/>
      </body>
      </html>
      <script type="text/javascript" language="javascript">
      var urlstring= window.location.href;
      var url=new URL(urlstring);
      var duid =url.searchParams.get("duid");
      console.log("duid");
      var fetch= "fetch2.php"
      if(duid!=null)
      {
        fetch="fetch2.php?duid="+duid;
      }
      console.log(fetch);
      $(document).ready(function(){
        var dataTable=$('#blacksites').DataTable(
          {
            "autoWidth": true,
            "processing":true,
            "serverSide":true,
            "order":[],
            "ajax":{
              url:fetch,
              type:"POST"
            }
```

comment.php comment2.php ×

C: > Users > intras > Desktop > project code > comment > comment2.php

```
          });
        $('#blacksites').on('draw.dt',function(){
          $('#blacksites').Tabledit({
            url:'action.php',
            dataType:'json',
            columns:{
              identifier:[0,"uid"]
              ,editable:[]
            },
            editButton:false,
            deleteButton:false,
            restoreButton:false,
            onSuccess:function(data,textStatus,jqXHR)
            {
              if(data.action=='delete')
              {
                $('#'+data.uid).remove();
                $('#blacksites').DataTable().ajax.reload();
              }
            }
          });
        });
        setInterval( function () {
                $('#blacksites').DataTable().ajax.reload( null, false ); // user paging is not reset on reload
        }, 30000 );
      });
      function refreshTable(dataTable){
        dataTable.draw(false,function(){
          setTimeout(refreshTable,10000);
        });
      }
      </script>
```

comment3.php

C: > Users > intras > AppData > Local > Temp > Rar$DIa0.435 > comment3.php

```php
<html>
<head>

  </style>
  <title>Sensor Past History </title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.2/dist/css/bootstrap.min.css" rel="stylesheet">
  <script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
  <script src="https://cdn.datatables.net/1.10.23/js/dataTables.bootstrap4.min.js"></script>
  <link href="https://cdn.datatables.net/1.11.3/css/dataTables.bootstrap5.min.css" rel="stylesheet">
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1

  <script src="https://markcell.github.io/jquery-tabledit/assets/js/tabledit.min.js"></script>
</head>
<body style="background-color:#cbc4b7;">
  <div>
      <nav class="navbar sticky-top navbar-expand-lg navbar-dark " style="background-color: #0e2433;">
    <a class="navbar-brand px-3" href="#">Auto Power Notifier</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#collapsibleNavbar">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="collapsibleNavbar">
      <ul class="navbar-nav" >
        <li class="nav-item">
          <a class="nav-link" href="/index.html">Home</a>
        </li>
        <li class="nav-item ">
          <a class="nav-link" href="/sensloc/map.php">Map</a>
        </li>
        <li class="nav-item ">
          <a class="nav-link" href="/sensloc/index.php">Record</a>
        </li>
```

comment3.php

C: > Users > intras > AppData > Local > Temp > Rar$DIa0.435 > comment3.php

```php
        </li>
        <li class="nav-item active">
          <a class="nav-link" style="color:teal;" href="/comment/comment.php">Status</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/about.php">About</a>
        </li>
      </ul>
    </div>
    </div>
  <div class="container" style="width: 99%;" style="background-color:#cbc4b7;">
    <h3 align="center">Past History </h3>
    <br/>
    <div class="panel panel default" style="background-color:#bccac9;">

      <div class="panel-body">
        <div class="table-responsive">
          <table id="blacksites" class="table table-bordered table-striped">
            <thead>
              <tr>
                <th>UID</th>
                <th>Status</th>
                <th>Incomment</th>
                <th>Time</th>

              </tr>
            </thead>
            <tbody></tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
```

comment3.php

C: > Users > intras > AppData > Local > Temp > Rar$DIa0.435 > comment3.php

```
    </div>
</div>
<br/>
<br/>
</body>
</html>
<script type="text/javascript" language="javascript">
var urlstring= window.location.href;
var url=new URL(urlstring);
var duid =url.searchParams.get("duid");
console.log("duid");
var fetch= "fetch3.php"
if(duid!=null)
{
  fetch="fetch3.php?duid="+duid;
}
console.log(fetch);
$(document).ready(function(){
  var dataTable=$('#blacksites').DataTable(
    {
      "processing":true,
      "serverSide":true,
      "order":[],
      "ajax":{
        url:fetch,
        type:"POST"
      }

    });
  $('#blacksites').on('draw.dt',function(){
    $('#blacksites').Tabledit({
      url:'action.php',
      dataType:'json',
```

comment3.php

C: > Users > intras > AppData > Local > Temp > Rar$DIa0.435 > comment3.php

```
      dataType:'json',
      columns:{
        identifier:[0,"comme"],
        editable:[[1,"status"],[5,"descript"]]
      },
      deleteButton :false,
      restoreButton:false,
      onSuccess:function(data,textStatus,jqXHR)
      {
        if(data.action=='delete')
        {
          $('#'+data.uid).remove();
          $('#blacksites').DataTable().ajax.reload();
        }
      }
    });

  });
  setInterval( function () {

      $('#blacksites').DataTable().ajax.reload( null, false ); // user paging is not reset on reload
  }, 20000 );
});
</script>
```

comment3.php fetch.php

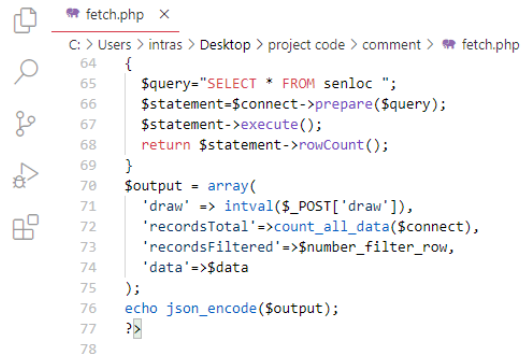
C: > Users > intras > Desktop > project code > comment > fetch.php

```php
<?php
session_start();
include('configdb.php');
$column =array("uid","status","bluep","yellowp","redp","scomment");
$query ="SELECT * FROM senloc WHERE status ='OFF' ";
if (!empty($_GET['duid'])){
  $duid=$_GET['duid'];
  $query ="SELECT * FROM senloc WHERE uid='".$duid."' ";

}

//post search
if($_POST["search"]["value"]){
  $query ="SELECT * FROM senloc ";
  if(isset($_POST["search"]["value"]))
  {
    $st="OFF";
    $query .='
WHERE uid  LIKE "%'.$_POST["search"]["value"].'%"
    OR scomment  LIKE "%'.$_POST["search"]["value"].'%"
    ';
     $query .=' OR status = "%'.$st.'%"';
  }
}
if (isset($_POST["order"]))
{
  $query .='ORDER BY '.$column[$_POST['order']['0']['column']].' '.$_POST['order']['0']['dir'].' ';
}
/*else
{
  $query .= 'ORDER BY uid DESC';

}*/
```

fetch.php

C: > Users > intras > Desktop > project code > comment > fetch.php

```php
}*/
$query1='';
if($_POST["length"])
{
  if($_POST["length"]!=-1)
  {
    $query1 .=  'LIMIT ' . $_POST['start'] . ', ' . $_POST['length'];

  }
}
$statement =$connect->prepare($query);
$statement->execute();
$number_filter_row=$statement->rowCount();
$statement=$connect->prepare($query . $query1);
$statement->execute();
$result=$statement->fetchALL();
$data = array();
foreach($result as $row)
{
  $sub_array=array();
  $sub_array[]=$row['uid'];
  $sub_array[]=$row['status'];
  $sub_array[]=$row['bluep'];
  $sub_array[]=$row['yellowp'];
  $sub_array[]=$row['redp'];
  $sub_array[]=$row['scomment'];
  $sub_array[]=$row['cus_feed'];
  $data[]=$sub_array;

}
function count_all_data($connect)
{
  $query="SELECT * FROM senloc ";
```



Fetch.php

fetch2.php

```php
<?php
//date_default_timezone_set('Africa/Nairobi');
session_start();
include('configdb.php');
$column =array("uid","statu","bluepa","yellowpa","redpa","latime");
$query ="SELECT * FROM sentatus ORDER BY latime DESC ";
if (!empty($_GET['duid'])){
  $duid=$_GET['duid'];
  $query ="SELECT * FROM sentatus WHERE uid='".$duid."' ";

}

//post search
if($_POST["search"]["value"]){
  $query ="SELECT * FROM sentatus ";
  if(isset($_POST["search"]["value"]))
  {
    $st="OFF";
    $query .='
WHERE uid  LIKE "%'.$_POST["search"]["value"].'%"
    OR scomment  LIKE "%'.$_POST["search"]["value"].'%"
    ';
      $query .=' OR statu = "%'.$st.'%"';
  }
}
if (isset($_POST["order"]))
{$query ="SELECT * FROM sentatus ";
  $query .='ORDER BY '.$column[$_POST['order']['0']['column']].' '.$_POST['order']['0']['dir'].' ';
}
/*else
{
  $query .= 'ORDER BY uid DESC';

}*/
```

fetch2.php

```php
}*/
$query1='';
if($_POST["length"])
{
  if($_POST["length"]!=-1)
  {
    $query1 .=  'LIMIT ' . $_POST['start'] . ', ' . $_POST['length'];

  }
```

fetch2.php

```php
$statement =$connect->prepare($query);
$statement->execute();
$number_filter_row=$statement->rowCount();
$statement=$connect->prepare($query . $query1);
$statement->execute();
$result=$statement->fetchALL();
$data = array();
//echo date("h:i:sa");

foreach($result as $row)
{
  $sub_array=array();
  $sub_array[]=$row['uid'];
  $sub_array[]=$row['statu'];
  $sub_array[]=$row['bluepa'];
  $sub_array[]=$row['yellowpa'];
  $sub_array[]=$row['redpa'];

  $row['latime']=strval(time2string(time()-strtotime($row['latime'])).' ago');

  $sub_array[]=$row['latime'];
  $data[]=$sub_array;

}
function count_all_data($connect)
{
  $query="SELECT * FROM sentatus ";
  $statement=$connect->prepare($query);
  $statement->execute();
  return $statement->rowCount();
}
```

```php
$output = array(
  'draw' => intval($_POST['draw']),
  'recordsTotal'=>count_all_data($connect),
  'recordsFiltered'=>$number_filter_row,
  'data'=>$data
);
function time2string($timeline) {
    $periods = array('day' => 86400, 'hour' => 3600, 'minute' => 60, 'second' => 1);
     $ret="";
    foreach($periods AS $name => $seconds){
        $num = floor($timeline / $seconds);
        $timeline -= ($num * $seconds);
    //if ($timeline<1)
        //break;
        if ($num==0)
        continue;
        $ret .= $num.' '.$name.(($num > 1) ? 's' : ' ').' ';
    }

    return trim($ret);
}
echo json_encode($output);
?>
```

fetch3.php

```php
<?php
session_start();
include('configdb.php');
$column =array("comme","statu","descript","timestamp");
$query ="SELECT * FROM locs WHERE statu ='OFF' ORDER BY timestamp DESC ";
if (!empty($_GET['duid'])){
  $duid=$_GET['duid'];
  $query ="SELECT * FROM  locs WHERE comme='".$duid."' AND statu='OFF' AND LENGTH(incomment)>1 ORDER BY timestamp DESC ";

}

//post search
if($_POST["search"]["value"]){
  $query ="SELECT * FROM locs ";
  if(isset($_POST["search"]["value"]))
  {
    $st="OFF";
    $query .='
WHERE comme LIKE "%'.$_POST["search"]["value"].'%"
    OR descript  LIKE "%'.$_POST["search"]["value"].'%"
    ';
      $query .=' OR statu = "%'.$st.'%"';
  }
}
if (isset($_POST["order"]))
{$query ="SELECT * FROM locs ";
  $query .='ORDER BY '.$column[$_POST['order']['0']['column']].' '.$_POST['order']['0']['dir'].' ';
}
/*else
{
  $query .= 'ORDER BY uid DESC';

}*/
$query1='';
if($_POST["length"])
```

fetch3.php

```php
$query1='';
if($_POST["length"])
{
  if($_POST["length"]!=-1)
  {
    $query1 .=  'LIMIT ' . $_POST['start'] . ', ' . $_POST['length'];

  }
}
$statement =$connect->prepare($query);
$statement->execute();
$number_filter_row=$statement->rowCount();
$statement=$connect->prepare($query . $query1);
$statement->execute();
$result=$statement->fetchALL();
$data = array();
//echo date("h:i:sa");

foreach($result as $row)
{
  $sub_array=array();
  $sub_array[]=$row['comme'];
  $sub_array[]=$row['statu'];
  $sub_array[]=$row['incomment'];

  $sub_array[]=$row['timestamp'];
  $data[]=$sub_array;

}
function count_all_data($connect)
{
  $query="SELECT * FROM locs ";
  $statement=$connect->prepare($query);
  $statement->execute();
  return $statement->rowCount();
```

fetch3.php

```php
    $data[]=$sub_array;

  }
  function count_all_data($connect)
  {
    $query="SELECT * FROM locs ";
    $statement=$connect->prepare($query);
    $statement->execute();
    return $statement->rowCount();
  }
  $output = array(
    'draw' => intval($_POST['draw']),
    'recordsTotal'=>count_all_data($connect),
    'recordsFiltered'=>$number_filter_row,
    'data'=>$data
  );
  function time2string($timeline) {
      $periods = array('day' => 86400, 'hour' => 3600, 'minute' => 60, 'second' => 1);
       $ret="";
      foreach($periods AS $name => $seconds){
          $num = floor($timeline / $seconds);
          $timeline -= ($num * $seconds);
      //if ($timeline<1)
          //break;
          if ($num==0)
          continue;
          $ret .= $num.' '.$name.(($num > 1) ? 's' : ' ').' ';
      }

      return trim($ret);
  }
  echo json_encode($output);
  ?>
```

Appendix D

register.php

C: > Users > intras > Desktop > project code > app > register.php

```php
<?php
$host = "localhost";
$dbname = "check data";
$username= "Davy";
$password="project123";

$conn= new mysqli( $host, $username,$password,$dbname);
// check if connection established succesfully

$username=$_POST["username"];
$email=$_POST["email"];
$password=$_POST["password"];
//$username=$_POST["username"];



$isValidEmail=filter_var($email,FILTER_VALIDATE_EMAIL);
if($conn)
{
  if(strlen($password)>40||strlen($password)<6){
echo"Password is too long or Short";

}elseif($isValidEmail===false){
  echo "This email is not valid";

}
else{
  $sqlcheckUser="SELECT * FROM `user_table` WHERE `username` LIKE '".$username."'";
  $usernameQuery=mysqli_query($conn,$sqlcheckUser);
  $sqlcheckEmail="SELECT * FROM `user_table` WHERE `email` LIKE '".$email."'";
  $emailQuery=mysqli_query($conn,$sqlcheckEmail);
  if(mysqli_num_rows($usernameQuery)>0){
    echo "Username Already In Use ";
```

register.php

C: > Users > intras > Desktop > project code > app > register.php

```php
    echo "Username Already In Use ";
  }
  else if(mysqli_num_rows($emailQuery)>0)
  {
    echo "Email already registered with another User";
  }
  else{
    $sql_register="INSERT INTO `user_table` (username,email,password) VALUES ('$username','$email','$password')";
    if(mysqli_query($conn,$sql_register)){
   echo "Successfully Registered";
    }
    else{
      echo "Failed to Register";
    }
  }
}
}
if($conn->connect_error){
  die("Connection failed: ". $conn->connect_error);
  echo "connection error";
}
```

register.php login.php

C: > Users > intras > Desktop > project code > app > login.php

```php
<?php
$host = "localhost";
$dbname = "check data";
$username= "Davy";
$password="project123";

$conn= new mysqli( $host, $username,$password,$dbname);

$email=$_POST["email"];
$password=$_POST["password"];
//$email="Nairuko@gmail.com";
//$password="Test123";
$isValidEmail=filter_var($email,FILTER_VALIDATE_EMAIL);
if($conn){

  if($isValidEmail===false){
    echo "This email is not valid";

  }
  else{
    $sqlcheckEmail="SELECT * FROM `user_table` WHERE `email` LIKE '".$email."'";
    $emailQuery=mysqli_query($conn,$sqlcheckEmail);
    if (mysqli_num_rows($emailQuery)>0)
    {
      $sqlLogin="SELECT * FROM `user_table` WHERE `email` LIKE '".$email."' AND `password` LIKE '".$password."'" ;
      $sqlQuery =mysqli_query($conn,$sqlLogin);
       if (mysqli_num_rows($sqlQuery)>0)
       {
         $userN="username";
         while ($row=$sqlQuery->fetch_assoc()){
        $userN=$row['username'];
         }
         $output = array(
```

login.php

C: > Users > intras > Desktop > project code > app > login.php

```php
         $output = array(
           'respon' => "Login Successfull",
           'userN'=> $userN
         );
         echo json_encode($output);
        //echo "Login Successfull";
       }
      else{
        $output = array(
          'respon' => "Wrong Password",
          'userN'=> " "
        );
        echo json_encode($output);
       //echo "Wrong Password";
      }

    }
    else{
      $output = array(
        'respon' => "This Email is not registered",
        'userN'=> " "
       );
       echo json_encode($output);
      //echo "This Email is not registered";
    }

  }

}
else{
  $output = array(
    'respon' => "Connection Failed",
    'userN'=> " "
```

login.php

C: > Users > intras > Desktop > project code > app > login.php

```
55                  echo json_encode($output);
56                  //echo "This Email is not registered";
57                 }
58
59             }
60
61         }
62         else{
63           $output = array(
64             'respon' => "Connection Failed",
65             'userN'=> " "
66           );
67           echo json_encode($output);
68
69           //echo "Connection Failed";
70         }
71
```

checkStatus.php

C: > Users > intras > Desktop > project code > app > checkStatus.php

```
1   <?php
2   $host = "localhost";
3   $dbname = "check data";
4   $username= "Davy";
5   $password="project123";
6
7   $conn= new mysqli( $host, $username,$password,$dbname);
8   $output=array();
9   $userN=$_POST["username"];
10  //$userN='Davies Momanyi';
11
12  if($conn){
13      $sqlUserCheck="SELECT * FROM `user_table` WHERE `username` LIKE '".$userN."'";
14        $sqlQuery =mysqli_query($conn,$sqlUserCheck);
15        if (mysqli_num_rows($sqlQuery)>0)
16        {
17
18          while ($row=$sqlQuery->fetch_assoc()){
19         $userN=$row['sensor'];
20          }
21          if ($userN!=null&&$userN!="pAllocate")
22          {
23              $sqlStatusCheck="SELECT * FROM `senloc` WHERE `uid` LIKE '".$userN."'";
24                $sqlQuery =mysqli_query($conn,$sqlStatusCheck);
25                if (mysqli_num_rows($sqlQuery)>0)
26                {
27
28                    while ($row=$sqlQuery->fetch_assoc()){
29                        $userN=$row['status'];
30                        $cusfeed=$row['cus_feed'];
31                         }
32                         $output = array(
33                           'status' => $userN,
```


Explorer (Ctrl+Shift+E)
C: > Users > intras > Desktop > project code > app > checkStatus.php
'status' => $userN,
'userN'=> "updated",
'cusfeed'=>$cusfeed,
);
}
else{
$output = array(
'status' => "Wrong Allocation of Sensor",
'userN'=>"updated",
'cusfeed'=>'none'
);
}
}
else if($userN=="pAllocate"){
$output = array(
'status' => "Pending Allocation",
'userN'=> "updated",
'cusfeed'=>'none'
);
}
else{
$output = array(
'status' => "Not yet allocated to sensor",
'userN'=> "updated",
'cusfeed'=>'none'
);
}


checkStatus.php
C: > Users > intras > Desktop > project code > app > checkStatus.php
}
}
else
{
$output = array(
'status' => "Username problem",
'userN'=> "updated",
'cusfeed'=>'none'
);
}
}
else{
$output = array(
'respon' => "Connection failed ",
'userN'=> "updated",
'cusfeed'=>'none'
);
}
echo json_encode($output);

rLocate.php

C: > Users > intras > Desktop > project code > app > rLocate.php

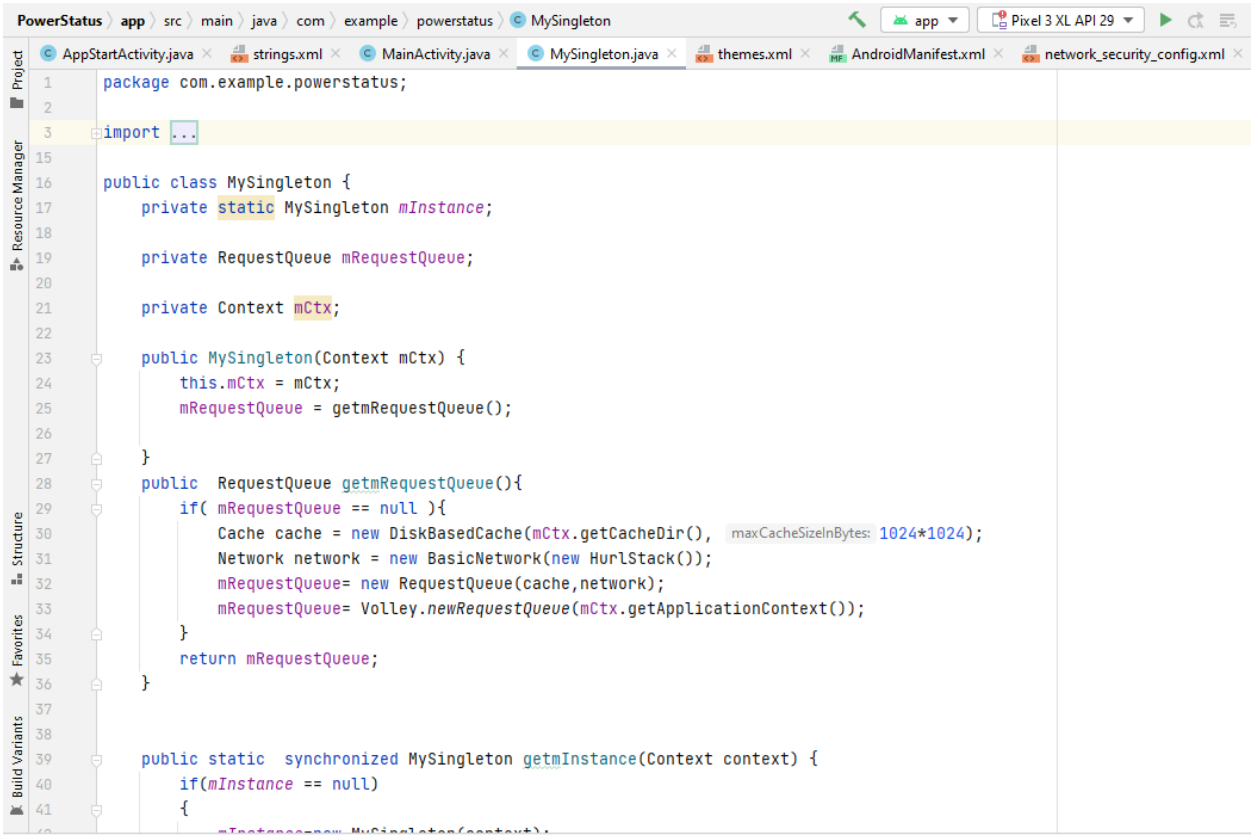
```php
<?php
$host = "localhost";
$dbname = "check data";
$username= "Davy";
$password="project123";

$conn= new mysqli( $host, $username,$password,$dbname);
// check if connection established succesfully

$username=$_POST["username"];
$lat=$_POST["lat"];
$lon=$_POST["lon"];
//$username=$_POST["username"];

/*$username="Mercy Nairuko";
$lat="-1.1032171";
$lon="37.0104711";*/




if($conn)
{
  $sql= " UPDATE user_table SET
  lat='".$lat."',
  lon='".$lon."',
  sensor='pAllocate'
  WHERE username='".$username."' " ;
      if($conn->query($sql)===TRUE){
        echo "Successfully Registered";
      }
      else {
        echo "Error: ". $sql . "<br>" .$conn-> error;
      }
```

rLocate.php

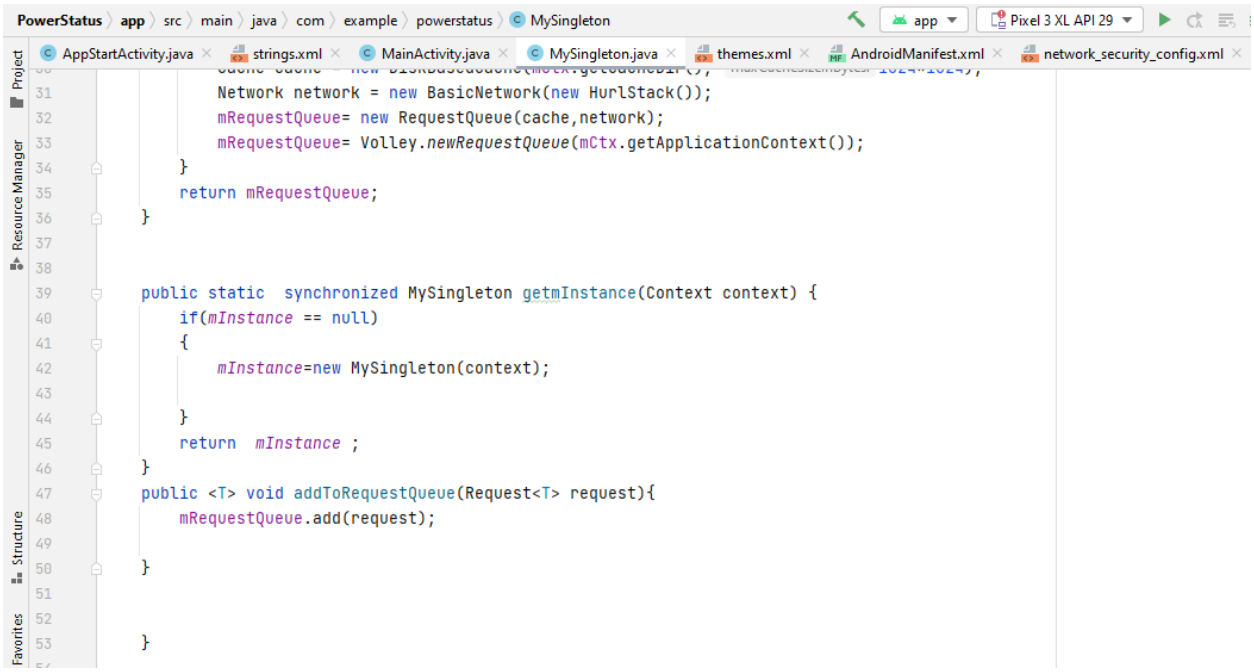
C: > Users > intras > Desktop > project code > app > rLocate.php

```php
{
  $sql= " UPDATE user_table SET
  lat='".$lat."',
  lon='".$lon."',
  sensor='pAllocate'
  WHERE username='".$username."' " ;
      if($conn->query($sql)===TRUE){
        echo "Successfully Registered";
      }
      else {
        echo "Error: ". $sql . "<br>" .$conn-> error;
      }
}
if($conn->connect_error){
  die("Connection failed: ". $conn->connect_error);
  echo "connection error";
}
```

MySingleton.java

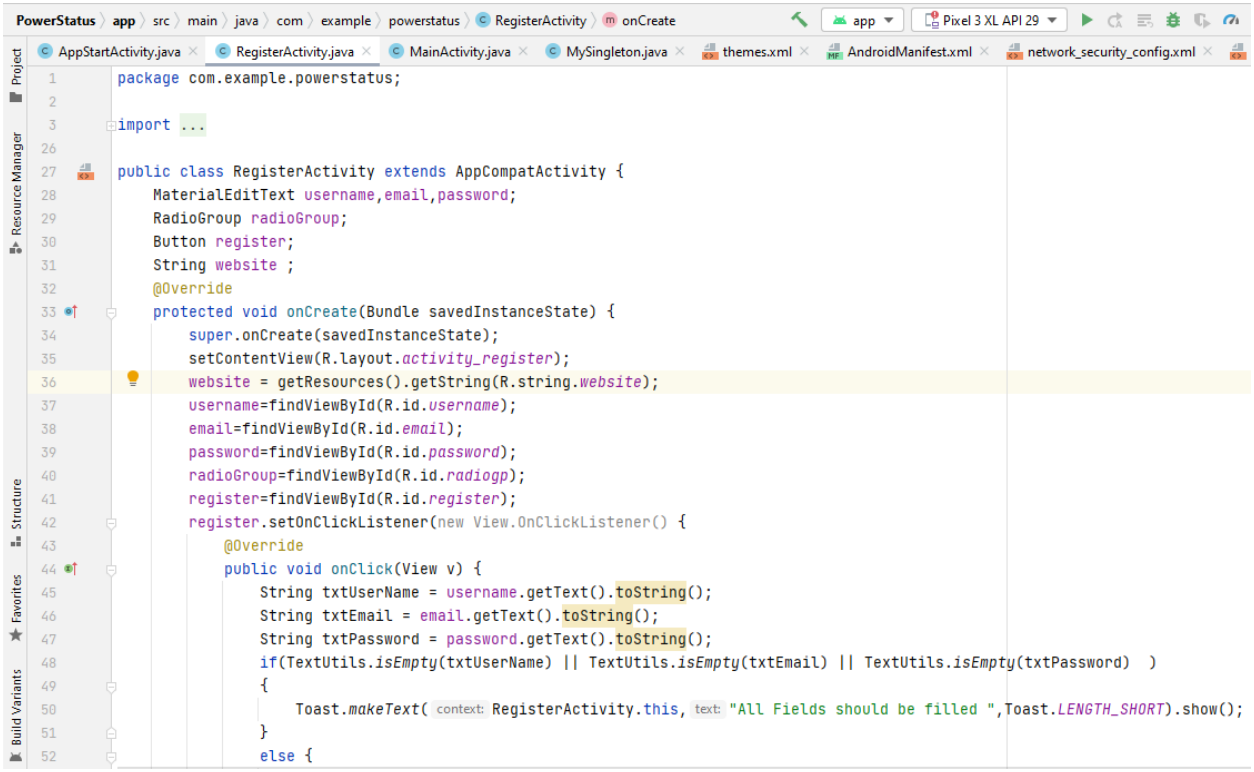


```
package com.example.powerstatus;

import ...

public class MySingleton {
    private static MySingleton mInstance;

    private RequestQueue mRequestQueue;

    private Context mCtx;

    public MySingleton(Context mCtx) {
        this.mCtx = mCtx;
        mRequestQueue = getmRequestQueue();

    }
    public  RequestQueue getmRequestQueue(){
        if( mRequestQueue == null ){
            Cache cache = new DiskBasedCache(mCtx.getCacheDir(), maxCacheSizeInBytes: 1024*1024);
            Network network = new BasicNetwork(new HurlStack());
            mRequestQueue= new RequestQueue(cache,network);
            mRequestQueue= Volley.newRequestQueue(mCtx.getApplicationContext());
        }
        return mRequestQueue;
    }


    public static  synchronized MySingleton getmInstance(Context context) {
        if(mInstance == null)
        {
```

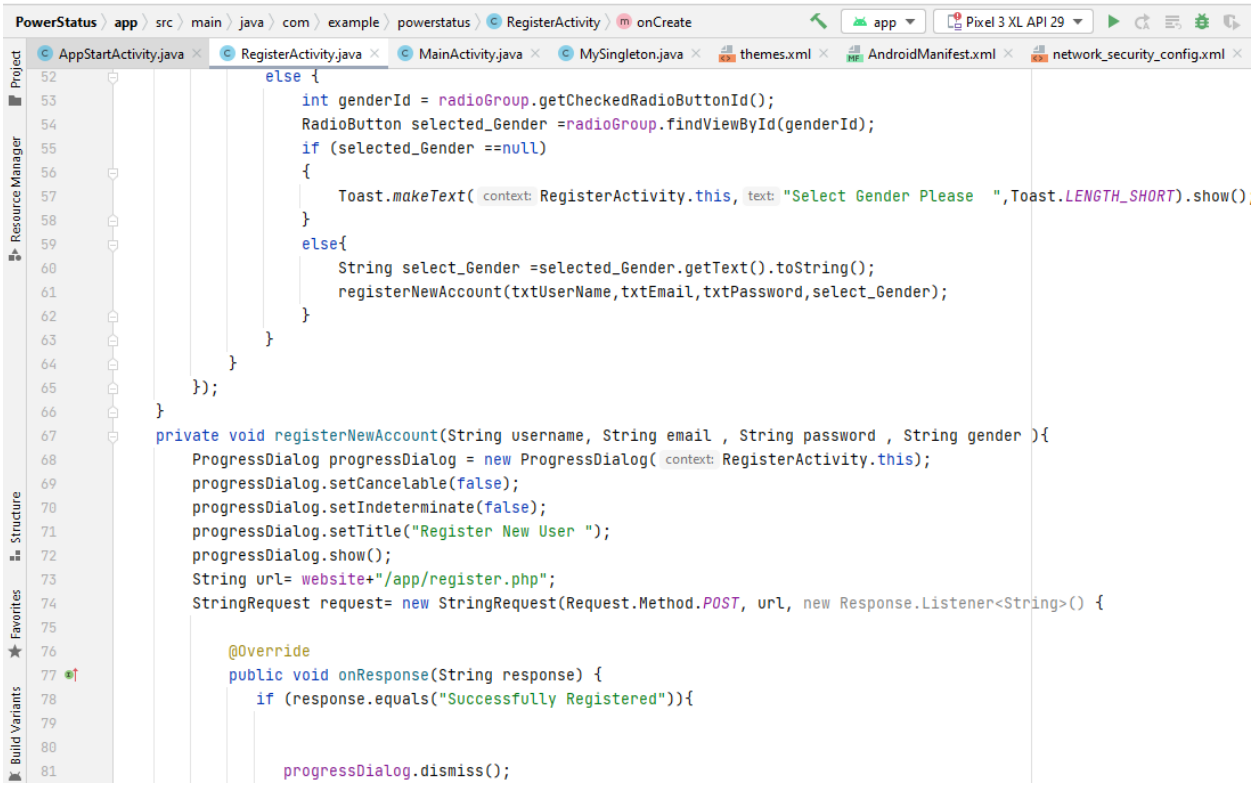


```
            Network network = new BasicNetwork(new HurlStack());
            mRequestQueue= new RequestQueue(cache,network);
            mRequestQueue= Volley.newRequestQueue(mCtx.getApplicationContext());
        }
        return mRequestQueue;
    }


    public static  synchronized MySingleton getmInstance(Context context) {
        if(mInstance == null)
        {
            mInstance=new MySingleton(context);

        }
        return  mInstance ;
    }
    public <T> void addToRequestQueue(Request<T> request){
        mRequestQueue.add(request);

    }


    }
```

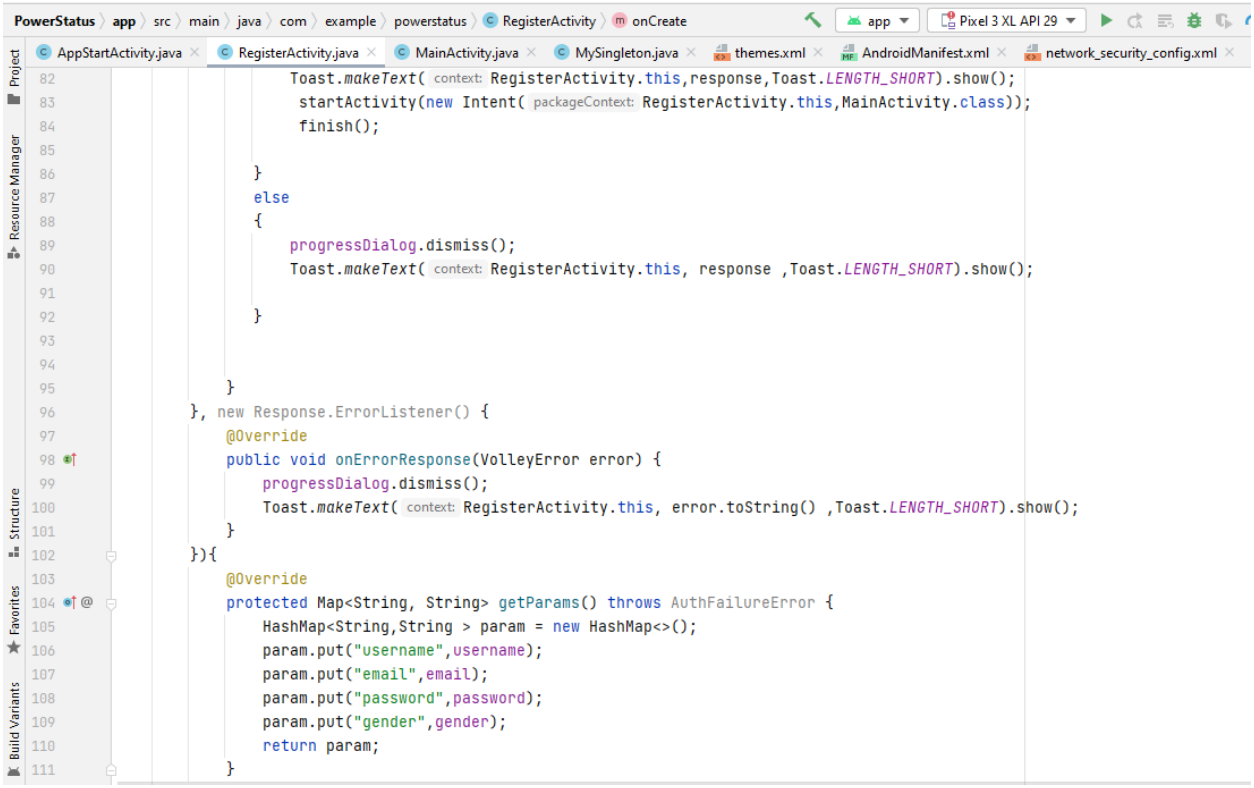
RegisterActivity,java



```
package com.example.powerstatus;

import ...

public class RegisterActivity extends AppCompatActivity {
    MaterialEditText username,email,password;
    RadioGroup radioGroup;
    Button register;
    String website ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        website = getResources().getString(R.string.website);
        username=findViewById(R.id.username);
        email=findViewById(R.id.email);
        password=findViewById(R.id.password);
        radioGroup=findViewById(R.id.radiogp);
        register=findViewById(R.id.register);
        register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String txtUserName = username.getText().toString();
                String txtEmail = email.getText().toString();
                String txtPassword = password.getText().toString();
                if(TextUtils.isEmpty(txtUserName) || TextUtils.isEmpty(txtEmail) || TextUtils.isEmpty(txtPassword)  )
                {
                    Toast.makeText( context: RegisterActivity.this, text: "All Fields should be filled ",Toast.LENGTH_SHORT).show();
                }
                else {
```

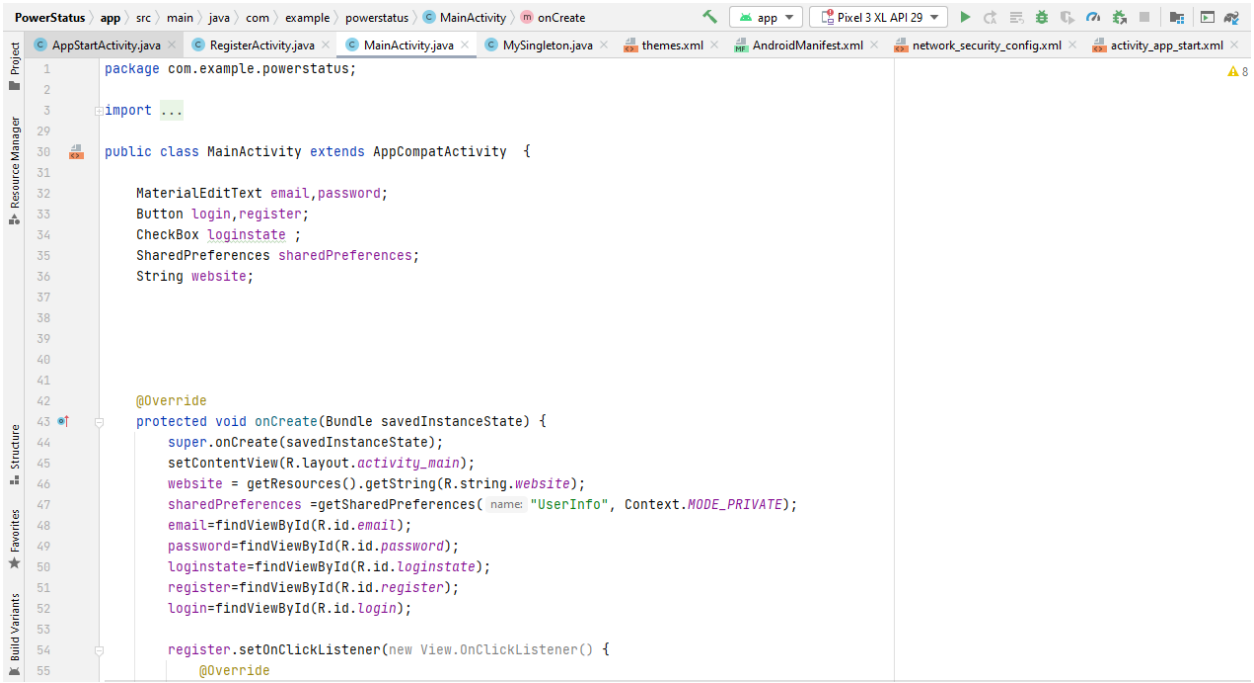


```
                else {
                    int genderId = radioGroup.getCheckedRadioButtonId();
                    RadioButton selected_Gender =radioGroup.findViewById(genderId);
                    if (selected_Gender ==null)
                    {
                        Toast.makeText( context: RegisterActivity.this, text: "Select Gender Please  ",Toast.LENGTH_SHORT).show()
                    }
                    else{
                        String select_Gender =selected_Gender.getText().toString();
                        registerNewAccount(txtUserName,txtEmail,txtPassword,select_Gender);
                    }
                }
            }
        });
    }
    private void registerNewAccount(String username, String email , String password , String gender ){
        ProgressDialog progressDialog = new ProgressDialog( context: RegisterActivity.this);
        progressDialog.setCancelable(false);
        progressDialog.setIndeterminate(false);
        progressDialog.setTitle("Register New User ");
        progressDialog.show();
        String url= website+"/app/register.php";
        StringRequest request= new StringRequest(Request.Method.POST, url, new Response.Listener<String>() {

            @Override
            public void onResponse(String response) {
               if (response.equals("Successfully Registered")){


                  progressDialog.dismiss();
```

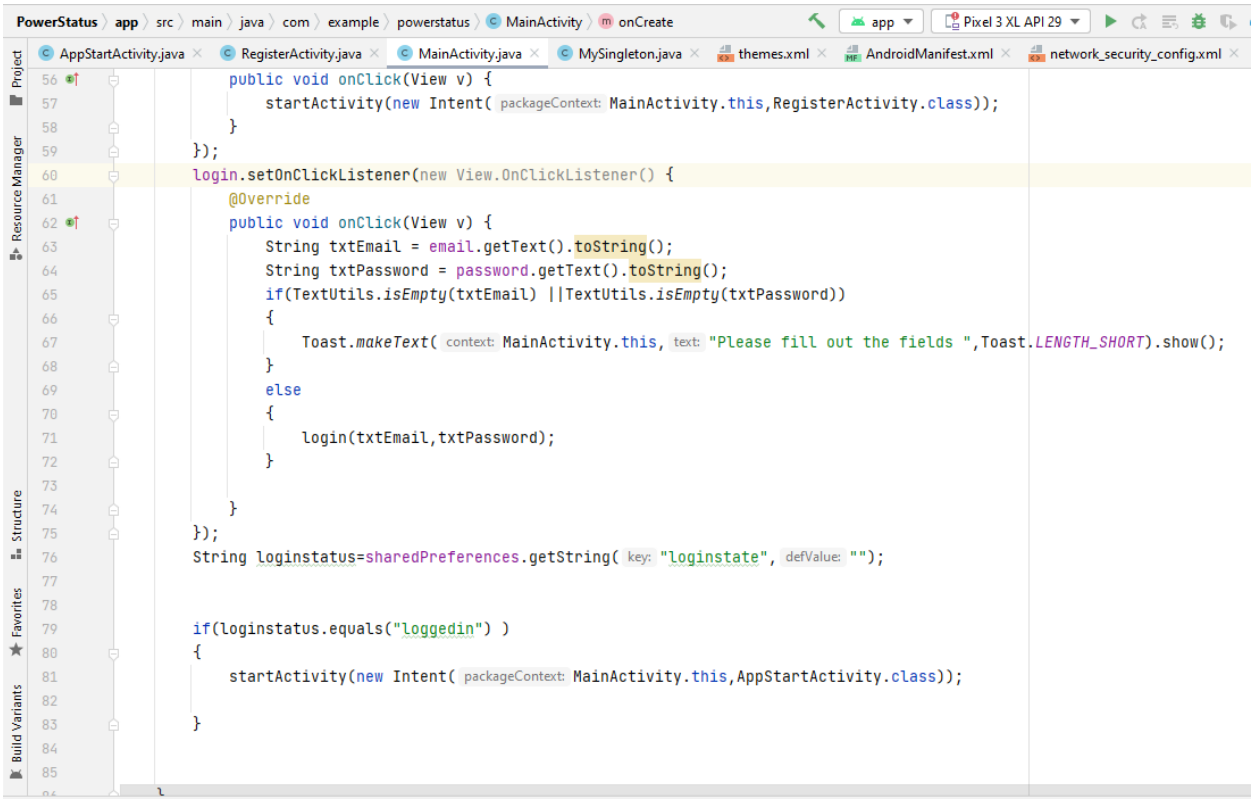


```
                    Toast.makeText( context: RegisterActivity.this,response,Toast.LENGTH_SHORT).show();
                     startActivity(new Intent( packageContext: RegisterActivity.this,MainActivity.class));
                     finish();

                }
                else
                {
                    progressDialog.dismiss();
                    Toast.makeText( context: RegisterActivity.this, response ,Toast.LENGTH_SHORT).show();

                }


            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                progressDialog.dismiss();
                Toast.makeText( context: RegisterActivity.this, error.toString() ,Toast.LENGTH_SHORT).show();
            }
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                HashMap<String,String > param = new HashMap<>();
                param.put("username",username);
                param.put("email",email);
                param.put("password",password);
                param.put("gender",gender);
                return param;
            }
```

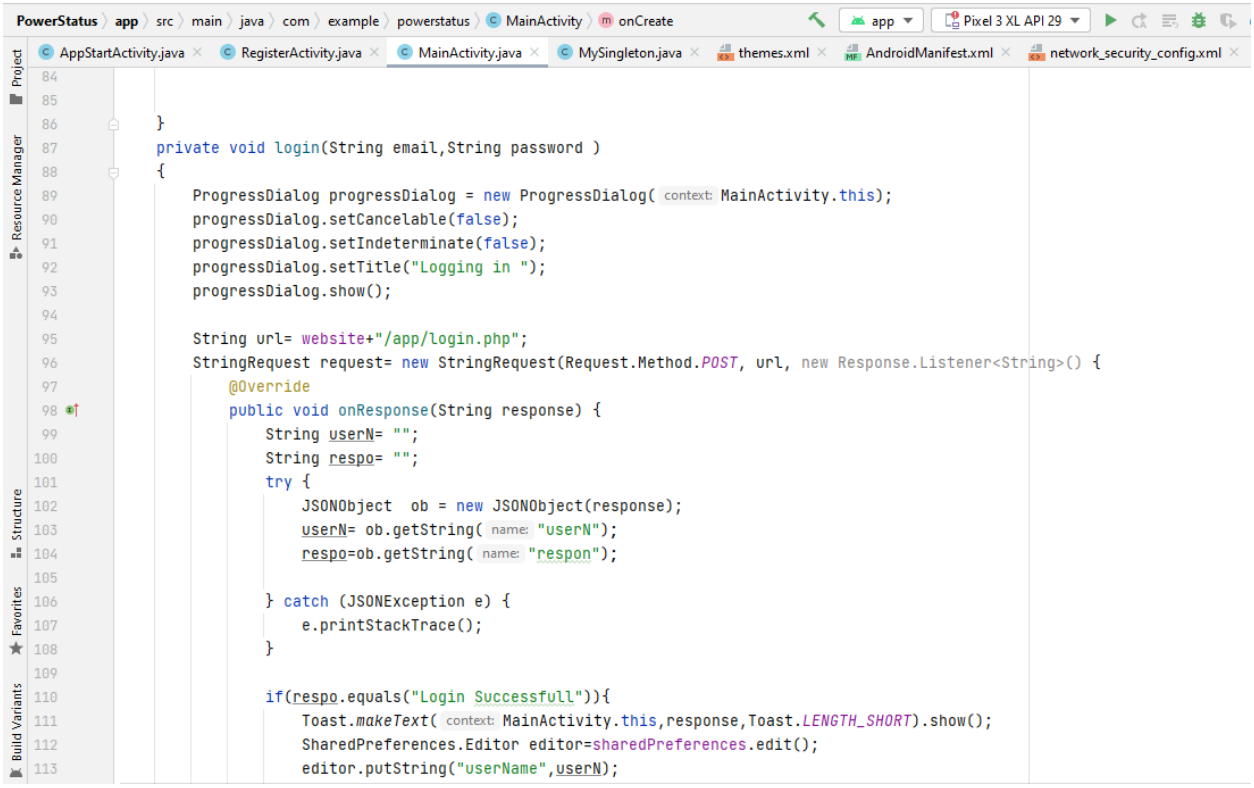
MainActivity.java



```
package com.example.powerstatus;

import ...

public class MainActivity extends AppCompatActivity  {

    MaterialEditText email,password;
    Button login,register;
    CheckBox loginstate ;
    SharedPreferences sharedPreferences;
    String website;




    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        website = getResources().getString(R.string.website);
        sharedPreferences =getSharedPreferences( name: "UserInfo", Context.MODE_PRIVATE);
        email=findViewById(R.id.email);
        password=findViewById(R.id.password);
        loginstate=findViewById(R.id.loginstate);
        register=findViewById(R.id.register);
        login=findViewById(R.id.login);

        register.setOnClickListener(new View.OnClickListener() {
            @Override
```

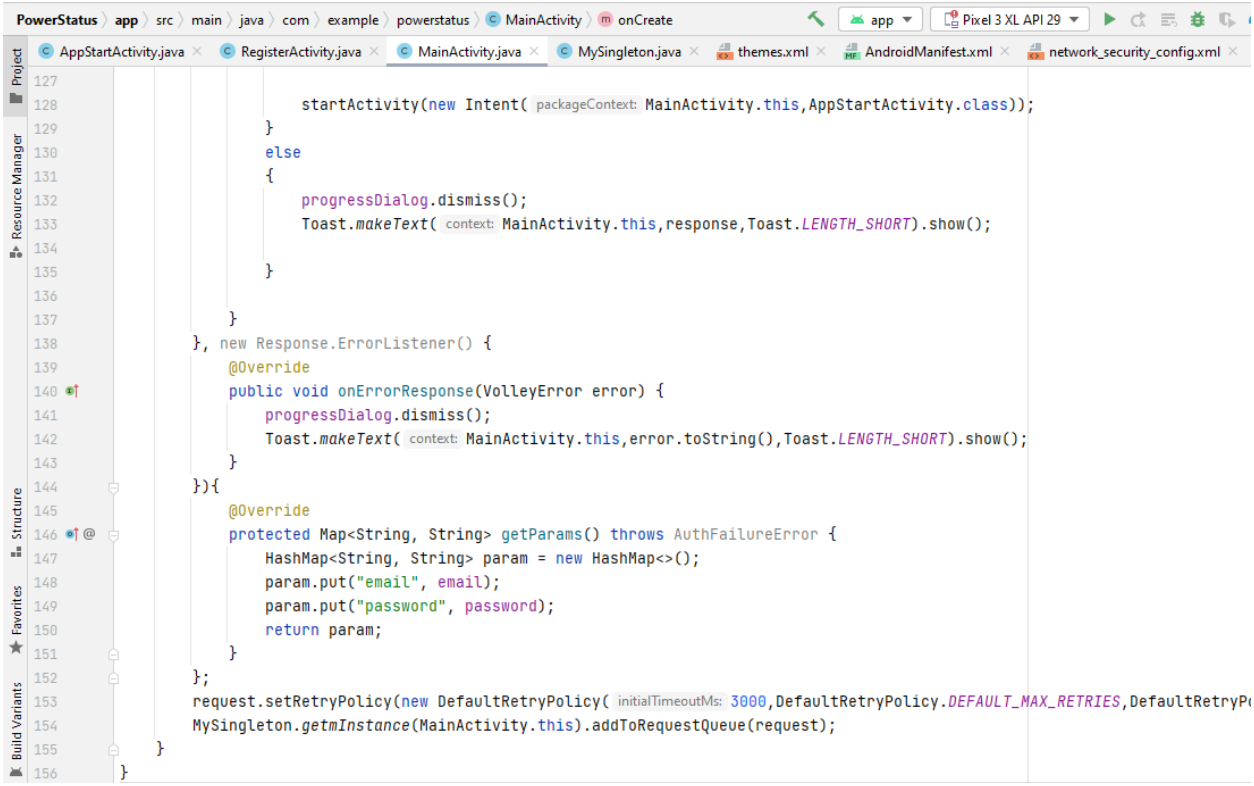


```
        public void onClick(View v) {
            startActivity(new Intent( packageContext: MainActivity.this,RegisterActivity.class));
        }
    });
    login.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String txtEmail = email.getText().toString();
            String txtPassword = password.getText().toString();
            if(TextUtils.isEmpty(txtEmail) ||TextUtils.isEmpty(txtPassword))
            {
                Toast.makeText( context: MainActivity.this, text: "Please fill out the fields ",Toast.LENGTH_SHORT).show();
            }
            else
            {
                login(txtEmail,txtPassword);
            }

        }
    });
    String loginstatus=sharedPreferences.getString( key: "loginstate", defValue: "");


    if(loginstatus.equals("loggedin") )
    {
        startActivity(new Intent( packageContext: MainActivity.this,AppStartActivity.class));

    }
```



```
}
private void login(String email,String password )
{
    ProgressDialog progressDialog = new ProgressDialog( context: MainActivity.this);
    progressDialog.setCancelable(false);
    progressDialog.setIndeterminate(false);
    progressDialog.setTitle("Logging in ");
    progressDialog.show();

    String url= website+"/app/login.php";
    StringRequest request= new StringRequest(Request.Method.POST, url, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            String userN= "";
            String respo= "";
            try {
                JSONObject  ob = new JSONObject(response);
                userN= ob.getString( name: "userN");
                respo=ob.getString( name: "respon");

            } catch (JSONException e) {
                e.printStackTrace();
            }

            if(respo.equals("Login Successfull")){
                Toast.makeText( context: MainActivity.this,response,Toast.LENGTH_SHORT).show();
                SharedPreferences.Editor editor=sharedPreferences.edit();
                editor.putString("userName",userN);
```


```
                if(respo.equals("Login Successfull")){
                    Toast.makeText( context: MainActivity.this,response,Toast.LENGTH_SHORT).show();
                    SharedPreferences.Editor editor=sharedPreferences.edit();
                    editor.putString("userName",userN);
                    editor.putString("pStatus"," ");
                    editor.putString("cStatus"," ");
                    editor.apply();

                    if(loginstate.isChecked()) {

                        editor.putString("loginstate","loggedin");
                    }
                    else
                        {
                            editor.putString("loginstate","loggedout");
                        }
                    editor.apply();

                    startActivity(new Intent( packageContext: MainActivity.this,AppStartActivity.class));
                }
                else
                {
                    progressDialog.dismiss();
                    Toast.makeText( context: MainActivity.this,response,Toast.LENGTH_SHORT).show();

                }

            }
```




```
                    startActivity(new Intent( packageContext: MainActivity.this,AppStartActivity.class));
                }
                else
                {
                    progressDialog.dismiss();
                    Toast.makeText( context: MainActivity.this,response,Toast.LENGTH_SHORT).show();

                }

            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                progressDialog.dismiss();
                Toast.makeText( context: MainActivity.this,error.toString(),Toast.LENGTH_SHORT).show();
            }
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                HashMap<String, String> param = new HashMap<>();
                param.put("email", email);
                param.put("password", password);
                return param;
            }
        };
        request.setRetryPolicy(new DefaultRetryPolicy( initialTimeoutMs: 3000,DefaultRetryPolicy.DEFAULT_MAX_RETRIES,DefaultRetryP
        MySingleton.getmInstance(MainActivity.this).addToRequestQueue(request);
    }
}
```