Smart water meter for Decision Support

J I Pitiyage
149223U

Faculty of Information Technology
University of Moratuwa
May 2017

Smart water meter for Decision Support

J I Pitiyage
149223U

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of the Master of Science in Information Technology.

May 2017

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

---------- ----------

Name of the Student Signature of the Student

Date:

Supervised by

---------- ----------

Name of the Supervisor Signature of the Supervisor

Date:

Dedication

This thesis is dedicated to my Mother I K Amarasekara and Wife A H Pitiyage for their endless love, encouragement and support.

Acknowledgements

First and foremost, I would like to offer my sincere gratitude to my research supervisor Mr. B. H. Sudantha, for his guidance, supervision, encouragement and support throughout this study.

I would not know what research is and how to do research if the lecture series of thesis writing and research methodologies is not offered. So I would like to offer my sincere gratitude to the Prof. Asoka S. Karunananda for feeding the knowledge and guidance for doing researches.

I would also like to thank all the lecturers of Faculty of Information Technology – University of Moratuwa, for their guidance and encouragement to get maximum use of knowledge and capabilities. So I would like to thank my batch mate Mr. Ravindra Vithange, Mr.Mihiranga Alwis and Mr.Rekha Sandaruwan, for their support.

Finally, I would like to extend my deepest gratitude to Mr. H. M Chandrapala and my class mate Dr. Rangika Halwathura, for their continuous support given in every possible way to make this project a success.

Abstract

In recent years, the demand for water has increased in households. Consumer's awareness regarding daily water consumption is very low. Traditional water meter unable get daily consumption efficient manner and water consumption has been calculating once in a month. It has been calculated by manually. With the advancement of information technology, over the past years there has not been any attempt to enhance manual meter reading to automate within Sri Lanka.

The main objective of this project is to propose and implement system to identify customer behaviors and pattern of consume water using Smart Water Meter System (SWMS). As developing nations, the technology of employing smart water meters new to the society. Cost savings and improved operational efficiency of meter reading personnel was achievable.

In some cases, existing water meters are fixed in the backyards and difficult in granting access to enter the premises. In such a situation, meter reading personal has to wait until the customer opens the gate. There are cases where the customer has locked and out of the premises and unable to get the meter reading. In such a situation the meter reader calculates the bill using an average monthly reading as the current month consumption. Customer's point of view this calculation will not be feasible when the actual reading exceeds the monthly consumption as charges may vary according to the tariff category they belong. These customers have been instructed to read their meters themselves and inform to a substation after receiving the monthly bill for alteration. This will be an extra cost to the billing as an officer has to be appointed to solve these matters.

The proposed system consists of Smart Water Meter (Developed using The Arduino Uno - microcontroller board, Flow Sensor, GSM Modem SIM900A and Wi-Fi Shield) native Android mobile application and a Web application. Instant meter reading has been saved to memory in the control board and periodical time interval data will be uploading to the cloud system. Customer can be logging to the cloud system and able to view daily consumption and analytical information during the given period. As additional feature meter reader can download meter reading via Wi-Fi to hand held device using Meter Reader mobile application.

With this system, it is expected to facilitate consumer to make better service and save the water as resource and reduce meter reading cost of the National Water Supply and Drainage Board.

Table of Contents

List of Figures

List of Table

Chapter 1

Introduction

1.1 Prolegomena

New technology platform is required to sustainably manage urban water resources in sri lanka. This will be achieved by real-time monitor water consumption at domestic, corporate and city levels. It will enable the implementation of efficient resources and demand management strategies. The proposed project aims to be better understanding consumption pattern of the customer and extracting water meter reading data with less human interaction efficient manner. This chapter presents the background and motivation of the research, hypothesis, objectives, problem statement, approach and the structure of the rest of the thesis.

1.2 Background and Motivation

Digital water meter allows to extracting meter reading electronically with less human interaction. Cost savings and improved operational efficiency was achievable. Mainly cost savings that could be created from improve efficiency of meter reading personnel. Once meter reading data is available, it can be captured and processed like any other signal. Internet, Mobile communication technology and other data communication technology make it possible to bring this signal to mobile phone or hand held device [1]. Currently there is no such system to record daily individual water consumption customers belongs to National Water Board in Sri Lanka. This data will be converting to information to get a better understanding of the consumption patterns of the water consumers.

In some cases, existing water meters are fixed in the backyards and difficult to granting access to enter the premises. In such situation meter reader has to wait until the customer opens the gate. There are cases where the customer locked and out premises. Such situation meter reader unable to get the meter reading. Water board instructed to calculates the bill using an average monthly consumption on special case as such. Customer's point of view this calculation will not be feasible when the actual reading exceeds the monthly consumption. Because next month customers have to pay higher bill value because of charges may be vary according to the tariff. These customers have been instructed to read their meters themselves and inform to a

substation after receiving the monthly bill for alteration. This will be an extra cost to the water board billing department. Because officer has to be assign to solve daily cases similar to this.

1.3 Problem statement

Above section we have discussed introduction of research area of this thesis, background

information of Smart Water Meter. In Chapter 2 we have done comprehensive literature review of digital image watermarking. We have got the evidence during in depth study of research aria, still have unsolved problems. During our study, we found that, currently there is no system to real-time monitor water consumption at domestic, corporate and city levels in Sri Lanka. Some solutions are implemented United State, United Kingdome and China to monitor their water consumption. It is necessary to implement an affordable solution which should be beneficial for the consumer who intends to know their current consumption of water. Therefore, utilizing commonly available equipment's, technologies and infrastructure for a Smart Water Meter System has been employed.

1.4 Hypothesis

We hypothesis that Smart Water Meter and automate data collection is efficient than manual meter reading.

1.5 Objectives

- Design and Develop Digital water meter
- Develop communication module to transfer meter reading using one of; Wi-Fi, GPRS
- Develop application to capture meter reading
- Develop application for customer to view water consumption
- Analyze the consumption pattern and demand management of water
- To estimate household water consumption and pattern
- To analyze water-saving potential for the residential sector

1.6 SWMS approach

The system can access from various devices including a computer connected to the Internet, Smart phones. The output of the system would be available in the printed

form for bill or soft copy versions such as SMS notifications. Smart Water Meter has identification number and it will be used to identify the device uniquely. Device will be register and activated before transmit data. Each and every user has authentication process before use the facilities of the system. This process goes beyond verification of user name and the password, but asks for Customer Billing Account Number when user registered for the first time. Meter Reader can download meter reading data using mobile application via Wi-Fi connection with proper access level.

1.7 Structure of the Thesis

The rest of the thesis is organized as follows. Chapter 2 critically reviews the literature on Smart Water Meter System and the research problems. Chapter 3 is about the technology for developing Smart Water Meter System. Chapter 4 presents new approach to use Smart Water Meter data transferring technique. Chapter 5 is about design of the system.

1.8 Summary

This chapter gave overall picture of the entire project presented in this thesis. As such we described the background and motivation, problem definition, hypothesis, objectives, and a brief overview of the solution. Next chapter presents critical review of literature on Smart Water Meter.

Chapter 2

Developments and Challenges in Smart Water Meter System

2.1 Introduction

Chapter 1 gave a comprehensive description of the overall project described in this thesis. This chapter provides a critical review of the literature in relation to developments and challenges of Water Meter System. For this purpose, the review of the past researches have been presented under three major sections namely, early developments, modern trends and future challenges. At the end, this chapter defines the research problem as the inadequate security in the data transferring techniques, and identifies the Wi-Fi and Android technology that can be used to address the problem.

2.2 Early developments

Among the various implementations of Automated Meter Reading System (AMRS) consist of two major modules. The Automated Meter Reading Module and Communication Module were identified. Communication Module use GSM Network (Global System for Mobile Communications) for data and SMS messaging services. GSM network provides coverage across countries [22]. GSM technology provides SMS service (Short Message Service) and GPRS (General Packet Radio Service) for requesting and retrieving data. GSM network widely used without any technical issues more than several years. It enable with efficient, reliable and secure communication standard [2, p. 66].

The ZigBee technology is effective high-level communication protocols such as create personal area networks, low power consumption and zero traffic. ZigBee communication technology installation not requires special permissions from the authorities (Any one can transmit) and handle multiple channel similar way. It uses unlicensed 2.4 GHz ISM band which is available worldwide. ZigBee has range between 10 m to 2 km and it works well with networks such as Wi-Fi, Ethernet and GPRS and also provides scalable networking solution which is suitable for used in controlling and monitoring application. GSM has a built in transport layer encryption, which is supported by most network providers. [3, p. 4].

WPAN (Wireless Personal Area Network) covers a small geographical area, and it uses IEEE 802.15 standard. It allows devices to provide communication each other. Bluetooth is standard for enabling wireless communication among mobile computers, mobile phones, and portable handheld devices. Wireless local area network (WLAN) allows sharing information between different equipment located in a limited distance. WMAN networks use Worldwide Interoperability for Microwave Access (WiMAX) technology to connect to wide areas (metropolitan width). It is a wireless communications standard designed to transferring rate 30 to 40 Mbps. It can transmit much more data and handle over far greater distances [4, p. 22].

Wi-Fi networks have no physical wired connection between sender and receiver by using radio frequency (RF). RF current is supplied to an antenna, an electromagnetic field is generated. Then It is able to broadcast through space. The keystone of any wireless network is an access point (AP). The primary job of an access point is to broadcast a wireless signal. Computers or mobile device can detect. In order to connect to an access point and join a wireless network, device must be equipped with wireless network adapters [23]. Table 2.1 summarize comparison of technology used for connectivity.

Technology Used	**Cost**	**Feasibility**	**Reliability**	**Coverage**	**Communication Protocol**
GSM	Low	Most Feasible	High	High	Stable
ZigBee	Medium	Small Scale	Low	Low	Least Stable
WiMAX	Medium	Small Scale	Medium	Low	Stable
Wi-Fi	Low	Small Scale	High	Low	Stable

Table 2.1: Consolidated comparison of all the systems

2.3 Modern Trends

The first attempts of Meter Automation or Automated Meter Reading (AMR) allowed utilities to remotely read the consumption [5, p. 1341]. Recent Smart Metering Systems equipped with improved architecture and include smart sensors and more

sophisticated distributed control technology. Table 2.2 summarize comparison of Smart and Traditional Meter.

No.	Features	Smart Meter	Traditional Meter
1	Remote Monitoring	Possible	Not Possible
2	Remotely Disconnection of Supply	Possible	Not Possible
3	Maintenance Cost	Less	High
4	Data Security	Avoid error meter reading	Error in meter readings

Table 2.2: Consolidated comparison of Smart and Traditional Meter

2.4 Future challenges

Global warming and urban migration has an enormous amount of pressure on the resources in many cities. This fast exhaustion of resources such as water calls for faster and firm actions from respective authorities to improve the management of resources. The solution must be moving Traditional Water Meter to Smart Water Meter system (SWM). SWM are electronic devices that record consumption of water in periodic intervals and communicates that information to a server. Domestic users will be able to monitor their water usage.

2.5 Problem Definition

Transferring technique, protocol and future challenges have been discussed so far. As per the literature review, low cost data transferring technique and low power consumption techniques are more relevant. The most of researchers have been used GSM enable technique for send and receive data (Ashna & George, 2013; Prashanthi & Prasad, 2014). The literature review has been identified various unsolved problems including security, efficiency, and reliability of Digital Water Meters. Table 2.3 summaries the achievements and the limitations of the key research discussed in this chapter.

Research	Country	Technology/ Algorithm used	Key benefits	Limitations
Wireless power meter monitoring with power theft detection and intimation system using GSM and ZigBee networks[3]	India	ZigBee, GSM	Automated system of theft detection, saving lot of labor work, time and cost of reading	The proposed system found to be little bit complex as far as distribution network is concerned
GSM Based Automatic Energy Meter Reading System with Instant Billing(Akhila, Murthy, & Ragini, 2013)	India	GSM enabled energy meter, .NET framework and C#	Generate timely bills, better understand energy demand patterns, manage meter failures more efficiently and fraud detection	Data transmission is charged at standard SMS rates
A Low Cost and Low Power Consumption Automatic Water Meter Reading System: Hardware Investigation and Network Design(Nguyen, Huynh, & Nguyen, 2015)	Vietnam	cluster based wireless sensor network, GPRS gateway	Low Cost and Low Power Consumption	The packet loss due to collision, the packet delivery delay

Table 2.3: Summary of literature of Smart Water Meters

After doing comprehensive literature review, some problems need to address on further research. According to Table 2.3, despite many solutions are available for wireless sensor network, they are rather expensive. Therefore, low-cost solution must be feasible.

The limited number of research on Smart Water Meter with Wi-Fi connectivity has been recorded in the literature[6]. More important, android application has been most popular for accessing large system through mobile device. Details of the technology behind the solution will be discussed in Chapter 3.

2.6 Summary

This chapter presented a comprehensive literature review on the Smart Water Meter research and identified the research problem as the inadequate attention to reliability of wireless protocols. Next chapter will discuss the technology to be used for our solution.

Chapter 3

Technology Adopted of SWMS

3.1 Introduction

In the previous chapter, various researches address specific issues were critically reviewed. Advantages, disadvantages and features of existing systems and proposed systems were analyzed and listed. In this chapter, technologies regarding Smart Water Meter System will be described. The methods used to development, testing and implementation will be discussed.

3.2 Technologies Available

3.2.1 Flow sensor

The Flow sensor is devices that detect and measure water flow through pipes. Water flow meter basically works with the flow sensor in order to calculate water flow. This model is produce as model of YF-S201 water flow sensor. The water flow follows through the rotor blade; Rotor will start to rotates. Thus pulses produce an output frequency which is directly proportional to the volumetric flow rate/total flow rate through the meter [9]. Figure 3.1 shows Turbine of Flow Meter block diagram.

.

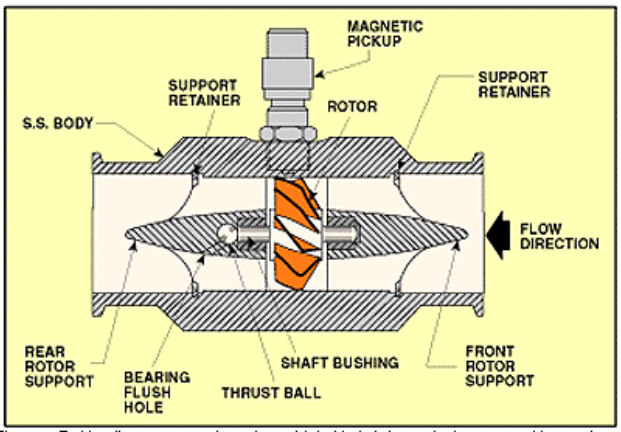

Figure 5: Turbine flowmeter consists of a multiple-bladed, free-spinning, permeable metal rotor housed in a non-magnetic stainless steel body. In operation, the rotating blades generate a frequency signal proportional to the liquid flow rate, which is sensed by the magnetic pickup and transferred to a read-out indicator

Figure 3.1: Turbine of Flow Meter

Table 3.1 illustrate Specifications of water flow sensor and Figure 3.2 shows YF-S201 water flow sensor.

Features	Details
Model:	YF-S201
Sensor Type	Hall effect
Max current draw	15mA @ 5V
Output Type	5V TTL
Working Flow Rate	1 to 30 Liters/Minute
Working Temperature	-25 to +80
Working Humidity	35%-80% RH
Accuracy	±10%
Maximum water pressure	2.0 MPa
Output duty cycle	50% +-10%
Output rise time	0.04us
Output fall time	0.18us
Flow rate pulse	Frequency (Hz) = 7.5 * Flow rate (L/min)
Durability	minimum 300,000 cycles
Cable length	15cm
	1/2" nominal pipe connections, 0.78" outer
Size	2.5" x 1.4" x 1.4"

Table 3.1: Specifications of water flow sensor

Figure 3.2: YF-S201 water flow sensor

3.2.2 Little Math Work

Flow rate can be determined relating to different techniques like change in velocity. The flow rate can vary according to velocity of water. Velocity will be depending on the pressure that force through pipelines. The pipe cross sectional area is known and

remains constant. The average velocity is an indication of the flow rate. The basis relationship for determining the liquid flow rate in such cases is Q=V*A, where Q is flow rate/total flow of water through the pipe. The V is average velocity of the flow and A is the cross sectional area of the pipe (viscosity, density and the friction of the liquid in contact with the pipe also influence the flow rate of water) [24] [25].

Pulse frequency (Hz) = 7.5Q, Q is flow rate in Liters/minute

Flow Rate (Liters /hour) = (Pulse frequency x 60 min) / 7.5Q

In other words:

Sensor Frequency (Hz) = 7.5 * Q (Liters/min)

Liters = Q * time elapsed (seconds) / 60 (seconds/minute)

Liters = (Frequency (Pulses/second) / 7.5) * time elapsed (seconds) / 60

Liters = Pulses / (7.5 * 60)

3.2.3 The Microcontroller board - Arduino Uno

The Uno Microcontroller board consist 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. The Uno can simply connect to a computer using USB cable or an AC-to-DC adapter or battery to get started. It consist of Atmega328 programmable microcontroller [7, p. 1]. Figure 3.3 illustrate major component of Arduino Uno - microcontroller board and Figure 3.4 explain Microcontrollers Pin Plan of the Arduino Uno.

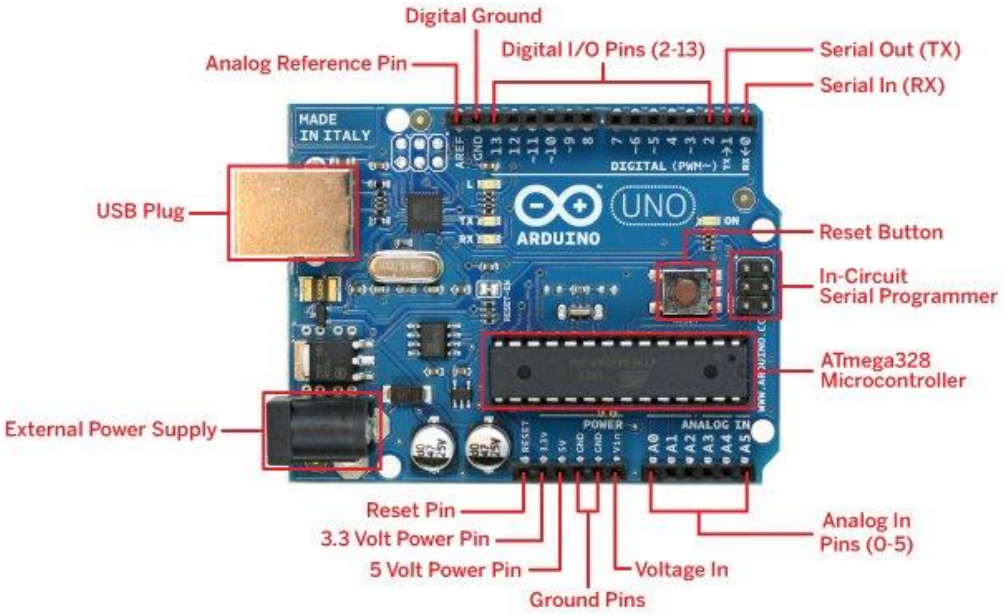

Figure 3.3: Arduino Uno - microcontroller board

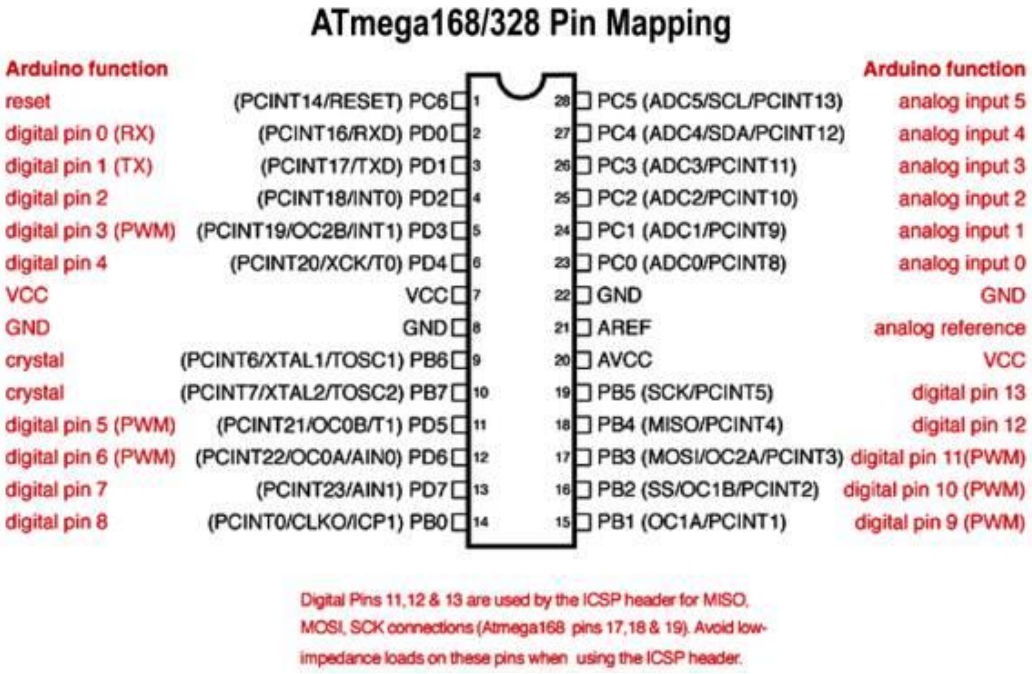

Figure 3.4: Arduino Uno - Microcontrollers Pin Plan

3.2.4 Wi-Fi Module

The ESP8266 has a fully TCP/UDP stack support. It can also be easily configured as a web server. The module accepts commands via a simple serial interface. Arduino device get similar advantage like Wi-Fi Shield when connect ESP8266. The ESP8266 module is an extremely cost effective board than Wi-Fi Shield [27]. ESP8266 Wi-Fi Module shows in Figure 3.5.

Figure 3.5: ESP8266 Wi-Fi Module

3.3 Web application (User interface and Backend Service)

There are plenty of technologies available to develop proposed Web application for Water Board Billing. ASP.net, C#.net, Java EE and PHP are commonly used to build web application. PHP has been selected for development for proposed application since there are many tools for developments and it is low cost specially in hosting the solution. Main requirement is calculating the bill and usage information. System is ability to cater a large amount of user's access simultaneously. Web application on Apache web server running on a Linux server which is low cost when compares with Windows servers. For the demonstration purposes host the application on a shared server.

3.4 Data model

MySQL is the top choice for developers and open source nature. It is free and can be used by anyone without any license or permission. It is under the General Public License (GNU) and the source code is available under domain. This enables developers to customize the source code according to their needs and modify the database for their requirement. As it is basically a modified version of SQL, a general knowledge of SQL is enough to work efficiently with MySQL.

3.5 Mobile Application

There are few popular mobile application platforms in the world at present including android, iOS, Windows and BlackBerry etc. The main goal of mobile application is to facilitate consumers to view their monthly usage and collect meter reading data to mobile device. Android studio is Integrated Development Toolkit (IDE) for Android application development.

3.6 Secure way of Data Transfer

Ciphertext is proposed method to encrypt electronic data into another form in secure manner. which cannot be easily understood by anyone except authorized parties [28].

3.6.1 Security features on Wi-Fi

Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access II (WPA2) are two security protocols and security certification programs developed by the Wi-Fi Alliance to secure wireless computer networks [10].

3.6.2 Encrypting Method - Symmetric Key Encryption (Private-Key)

A symmetric key, sometimes called private-key, encryption cipher is an algorithm in which the key for encryption is trivially related to the key used for decryption. An analogy of this is a typical mechanical lock. The same key that engages the lock can disengage it. To protect anything valuable behind the lock, the key must be given to each member securely [8, p. 13].

3.7 Cloud computing

Cloud computing is most probably the cost effective method to maintain and upgrade. Traditional desktop software need to pay higher license fee for venders. It is higher cost in terms of finance. The licensing fees for multiple users have been very expensive but cloud available at comparatively cheaper rates. Storing information in the cloud gives almost unlimited storage capacity. Hence, no more need to worry about running out of storage space or increasing current storage space availability. Once register in the cloud, user can access the information from anywhere, when Internet connection available. This convenient feature lets move beyond time zone and geographic location issues.

3.8 Technology Stack

Cloud Computing architecture include of many components, which are loosely coupled. Cloud architecture can broadly divide the in to two parts [11]. Figure 3.6 illustrate major component of Cloud architecture.

- Front End
- Back end

3.8.1 Front End

The front end refers to the client part of cloud computing system. Cloud computing platform is required applications and interfaces (Web Browser).

3.8.2 Back End

The back End refers to the cloud itself. It contains all of the resources essential to provide cloud computing services. It contains huge data storage, virtual machines, security mechanism, services, deployment models and servers etc.

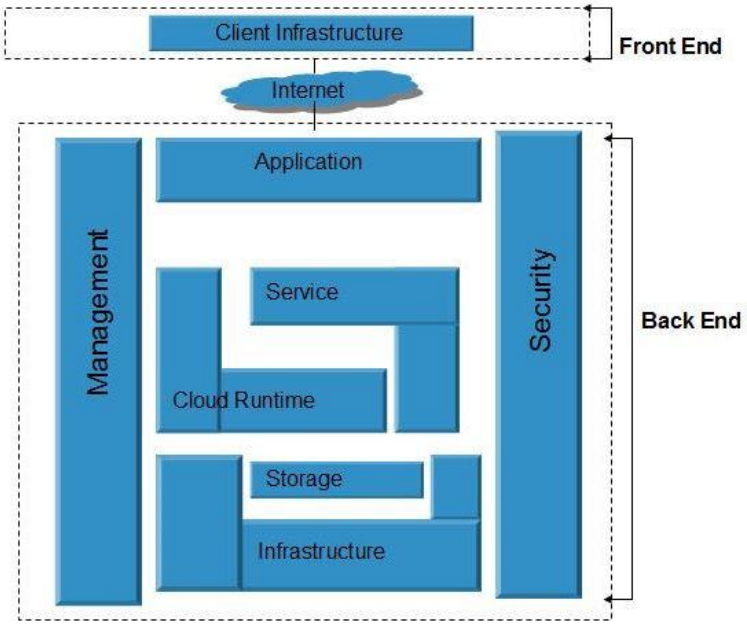

Figure 3.6: Cloud architecture

3.9 Summary

In this chapter, technologies and reasons for use relevant technologies were described. Advantages and disadvantages of technologies briefly discussed. In next chapter, the approach to implement SWMS will be described.

Chapter 4

An approach to SWMS

4.1 Introduction

In the previous chapter, SWMS design principals and technologies have been described. In this chapter, the approach used to solve the mentioned issues terms of inputs, expected outputs, processes available inside the system, additional features and the targeted set of users of the system has been described.

4.2 Hypothesis

Automated data collection using Smart Water Meter is efficient than manual meter reading.

4.3 Users

There are number of uses can be benefited from the Smart Water Meter system in multiple ways. National Water Supply and Drainage Board, Domestic Water Consumers, Private, State Sector Consumers and Meter Readers are can be directly benefited from this solution. Those who are interested to study of Smart Water Meter system can also use for learning purposes.

4.4 Input

The input could be a Water Meter reading data captured by using water meter and Meter reader's comments.

4.5 Output

The output of the system would be currant cost and consumption of water can display by customer. Monthly meter reading text file can download by Meter reader. Decision support information is available for Hourly, Daily, Weekly and Monthly consumption. Water leakage information can be identifying after analyzing individual data.

4.6 Process

Smart Water Meter has identification number and it will be used to identify the device uniquely. Device will be register and activated before transmit data. Each and every user has authentication process before use the facilities of the system. Meter Reader can download meter reading data using mobile application via Wi-Fi connection with

proper authentication. Meter reader upload data to server to calculate individual water bill.

4.7 Features

In connection with the input, output, users and process the system include the following characteristics.

- Mobile Application solution
- Upload Meter Reading to the System
- Download Meter Reading Data using Hand held Device
- Calculate Water Bill
- User friendliness
- Easy accessibility

4.8 Summary

This chapter described overall solution for the research describing in terms of its inputs, outputs, process, users and specific features. Next chapter will describe in detail, extended design of process and what system does.

Chapter 5

Design of SWMS

5.1 Introduction

The previous chapter gave full picture of the entire solution. This chapter describes the design of solution for the process presented in the approach. WE design the solution as a server system with a backend database. Here we describe the top-level architecture of the design by elaborating on the role of each component of the architecture.

5.2 Top Level Architecture of Smart Water Meter

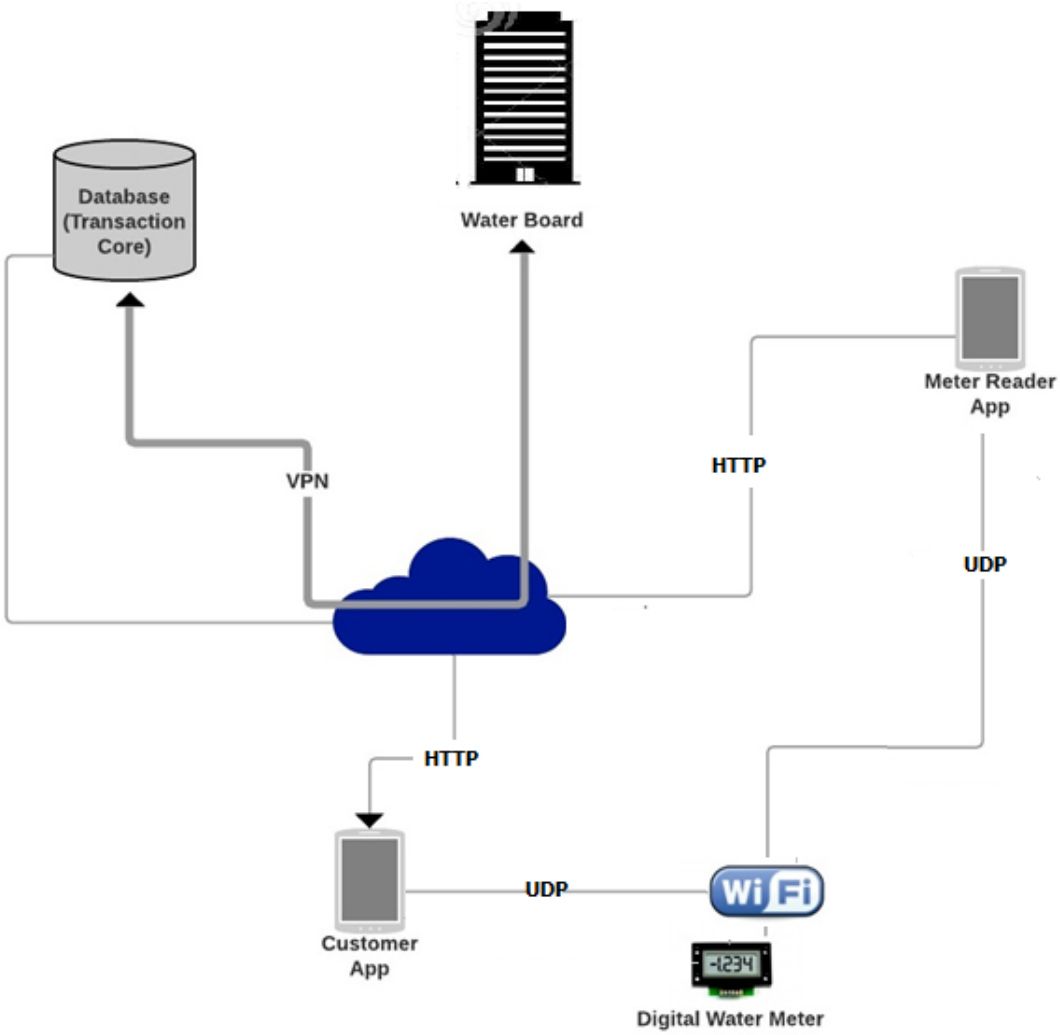

Figure 5.1: Component diagram with input and output

Above diagram Figure 5.1 describes input, output, major modules in system and relationship between each module. Digital Water Meter, Customer Mobile Application, Meter Reader Mobile Application and Water Board Billing system (Transaction Core) are four major modules of the SWMS. Each and every module of the system will be described individually in this chapter. Figure 5.2 illustrate Top Level architecture of SWMS.

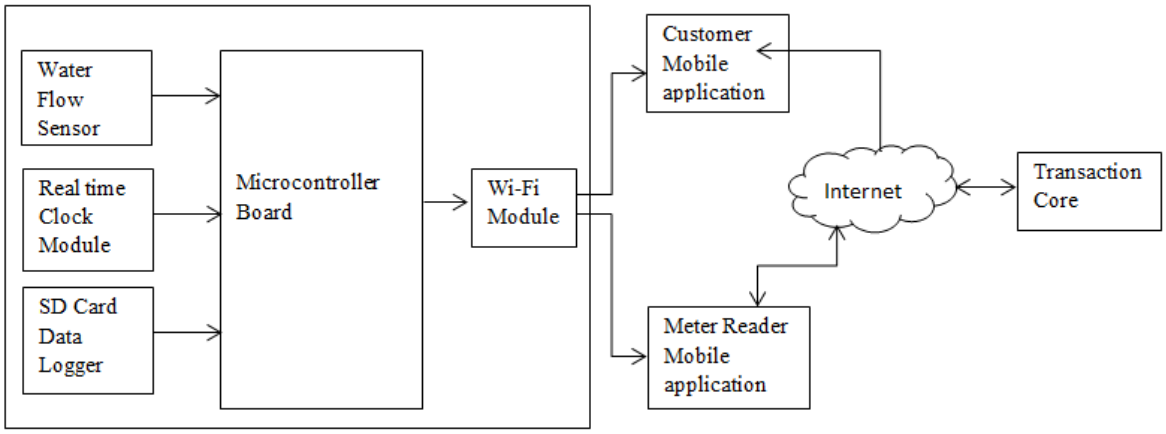

Figure 5.2: Top Level architecture of SWMS

5.3 Digital Water Meter

Digital Water meter has five major components namely Flow Sensor, Arduino Hardware Interface, Real Time Clock Module, SD Card Data Logger Module and Wi-Fi Module.

5.3.1 Flow Sensor

This sensor sits in line with water line and measure how much liquid has moved through it. There is integrated magnetic hall effect sensor and send out an electrical pulse. The hall effect sensor is sealed. It will allow the sensor to stay safe and dry.

5.3.2 Arduino platform

Arduino is an open source electronics platform based on easy to use hardware and software. Developer can send a set of instructions to the microcontroller. All Arduino boards are totally open source, empowering users to build them independently and ultimately adapt them to their particular needs [12].

5.3.3 Arduino Hardware Interface

Arduino / Genuino Uno board consists of ATmega328P microcontroller chip. It has 14 digital input output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack and a reset button. The ATmega328 on the Arduino / Genuino Uno comes preprogrammed with a boot loader that allows uploading new code [13].

5.3.4 Real Time Clock Module

The DS3231 is extremely accurate I²C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal. The device consists of inbuilt battery to maintains accurate time when main power interrupted. The RTC maintains seconds, minutes, hours, day, date, month, and year information. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator [14].

5.3.5 SD Card Data Logger Module

The SD Card Module required to save meter reading data. The measurements read from sensors (Flow sensor and Real Time clock) and log in to text file on SD card. The data required to transferred to mobile device later date for billing. SD card is cheap and massive storage solution. The Arduino development environment support providing a built in library which supports to SD card data logger module.

5.3.6 Wi-Fi Module

ESP8266 Module is a based on ESP8266 system on chip (SoC) with consist of Wi-Fi serial transceiver. This chip implements a full TCP/IP protocol stack. It has great computational power onboard feature. Hosting simple TCP or UDP onboard sever can make a very compact Internet of Things solution [15].

5.4 Customer Mobile Application

This mobile application facilitates customer to view current reading and cost of water usage using smart phone. Through this module user authentication, secure login will be provided. The module offers water consumption, Billing and analytical information.

5.5 Meter Reader Mobile Application

Through this module user authentication, secure login has been provided. The application module offers facilities to transfer meter reading data from water meter to Mobile device. Meter Reader can input his observation when required.

5.6 Water Board Billing web application

The sever works as the central point of the system. Data of each water meter has to record in to sever. Data will be used to identify consumption patterns and change of consumer over the time. This information is useful to improve efficiency operational processes. Management can understand what would be the best time when maintenance should be done on the nation water system. The data is benefited for resource planning demand forecast and demand management.

5.7 Summary

This chapter mainly described the overall architecture and the design of each component with relevant technologies and their interconnections. The next chapter is mainly focused on implementation details of the Smart Water Meter System. It will present some important code segments and related implementation details.

Chapter 6

Implementation of SWMS

6.1 Introduction

In this chapter will describe Smart Water Meter System design in detail according to the design on previous chapter. This chapter will describe how to integrate individual hardware modules to the arduino board and code segment run on each hardware module.

6.2 Overall solution

Digital Water Meter, Two Mobile applications and web application are three main components of the solution. Meter Reader Mobile application facilitate to download data from the water meter. Customer Mobile Application will display current meter reading.

6.3 Software and Hardware used

Arduino / Genuino Uno use open-source software (IDE) to progrmme ATmega328 micro controller chip.

Xamarin Studio is a modern, sophisticated IDE to creating iOS, Mac and Android mobile applications. It includes a rich editor, debugging, native platform integration with iOS, Mac and Android [17]. C# is the mostly commonly used language to create cross-platform applications collaborate with Xamarin Studio.

Separate billing system was developed for water board. PHP use to development of web application. Which is open source scripting language having Object Oriented features. MySQL database use to keep record of customer information, water consumption and charges. WAMP server which consist of apache, MySQL and PHP. This open source stack cost is comparatively low when compared to other technologies such as ASP.NET and Java.

For the demonstration purposes, this billing web application is hosted on shared hosting plan which consist of Apache 2.2 web server, MySQL 5.6 database and PHP 5.6.

6.4 Implantation of UDP Sever

ESP8266 module has programmed to implements UDP Sever. Arduino IDE use to develop and upload UDP Sever. There is two way of upload Sever programme to ESP8266 module.

- Upload code Using USB TTL converter
- Using Arduino Uno Board (Without USB TTL converter)

6.4.1 Upload code Using USB TTL converter

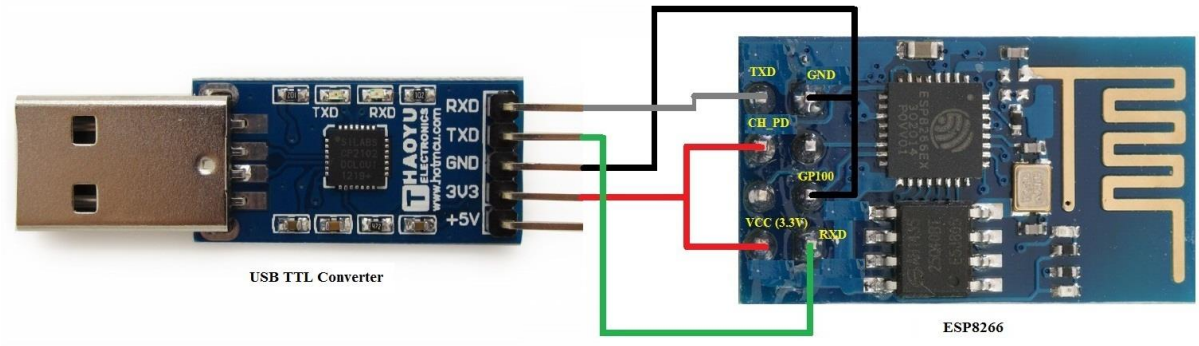

Figure 6.1: USB TTL converter and ESP8266

Figure 6.1 illustrate connectivity of USB TTL converter and ESP8266 module using female connecting wires, steps of configuration and installing relevant libraries in Appendix A.

6.4.2 Upload code Using Arduino Uno Board (Without USB TTL converter)

Arduino Uno Board directly connected to ESP8266 (without USB TTL converter) to upload UDP sever programme. It required to upload blank Arduino code in to Arduino Uno Board. This method is time consuming than previous method (discussed 6.3.1). If blank code will not upload to Arduino Uno Board, end up with error message. Refer Appendix B for detail instruction.

6.4.3 Implementation of Sever programme

ESP8266 has to be enable to upload mode by connecting GP100 pin to GND. While uploading programme to ESP8266 blue led continually blink until end of upload. After upload code, GP100 pin has to be removed from the GND. Defaults IP address

192.168.4.1 and port define as 23. This sever can connect only one client at a time Figure 6.2.

```
#include <ESP8266WiFi.h>
#define MAX_SRV_CLIENTS 1
WiFiServer server(23);
WiFiClient serverClients[MAX_SRV_CLIENTS];
```

Figure 6.2: Create UDP Sever using ESP8266

6.4.4 Universal asynchronous receiver and transmitter (UART)

A universal asynchronous receiver and transmitter (UART) play major role when implementing serial communication. The UART work as an intermediary between serial and parallel interfaces. UART is a bus of eight data lines (some extra control pins include), on the other side there are two serial wires - RX and TX as Figure 6.3 [16].

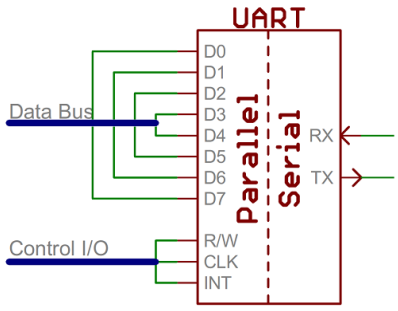

Figure 6.3: Universal asynchronous receiver and transmitter

UART implementation code as follows (Figure 6.4).

```
//check UART for data
if(Serial.available()){
  size_t len = Serial.available();
  uint8_t sbuf[len];
  Serial.readBytes(sbuf, len);
  //push UART data to all connected telnet clients
  for(i = 0; i < MAX_SRV_CLIENTS; i++){
    if (serverClients[i] && serverClients[i].connected()){
      serverClients[i].write(sbuf, len);
      delay(1);
```

Figure 6.4: describe UART code in ESP8266

6.4.5 Interconnection of ESP8266 Module and Arduino Uno Board

Table 6.1 illustrate Pin layout for connecting ESP8266 Module and Arduino Uno Board.

ESP8266 Module	Arduino Uno Board
RX	5
VCC	3.3v
GP100	* GND (at the time programme upload)
Rest	-
GP102	-
CH_PD	3.3v
GND	3.3v
TX	6

Table 6.1: Interconnection of ESP8266 Module and Arduino Uno Board

6.5 Implantation of water flow sensor

Table 6.2 illustrate Pin layout for the flow sensor and Arduino Uno Board.

Flow Sensor	Arduino Uno Board
Red	5v
Yellow	2
Black	GND

Table 6.2: Pin layout of the flow sensor

Table 6.3 illustrate Calibration factor of the flow sensors depend on the diameter of the flow sensor.

Pipe connections (Inch)	Model	Calibration
1/2	YF-S201 Hall Effect Water Flow Sensor	7.5
3/4	YF-S403 Hall Effect Water Flow Sensor	4.0

Table 6.3: Calibration factor of the flow sensors

Data wire of the flow sensor connected to Arduino Uno Board pin 2 and it initiated and calibration factor fine tune as 4.15 in the program, shows in Figure 6.5.

```
byte sensorInterrupt = 0;  // 0 = digital pin 2
byte sensorPin       = 2;

// The hall-effect flow sensor outputs approximately 4.5 pulses per second per
// litre/minute of flow.
float calibrationFactor = 4.15;
```

Figure 6.5: Initialize Calibration Factor of the Flow Sensor

6.6 Implantation of Real Time Clock

The DS3231 RTC module integrates DS3231 Real Time Clock (RTC) IC which has an internal crystal and a switched bank of tuning capacitors. The temperature of the crystal is continuously monitored, and the capacitors are adjusted to maintain a stable frequency. Compared to other RTC solutions, this DS3231 module drifts less than a minute per year, even in extreme temperature ranges. This makes DS3231 module suits very well for time critical applications that cannot be regularly synchronized to an external clock.

6.6.1 Features of Real Time Clock

- Can be connected directly to the microcontroller IO ports
- Two calendars and alarm clock
- Two programmable square-wave outputs
- Real time clock generator for seconds, minutes, hours, day, date, month and year
- Valid until 2100 with leap year compensation
- Battery socket compatible with LIR2032 batteries

6.6.2 Install of DS3231 Real Time Clock Library to Arduino IDE

Arduino IDE need to install DS3231 module library provided by device vender to support Real Time clock illustrate in Appendix A - Figure 4.

6.6.3 Interconnection of DS3231 Real Time Clock Module and Arduino Uno Board

Pin layout of the DS3231 Real Time Clock Module and Arduino Uno Board as follows.

DS3231 Real Time Clock Module	Arduino Uno Board
SCL	SCL
SDA	SDA
VCC	5v
GND	3.3v

Table 6.4: Interconnection of DS3231 RTC Module and Arduino Uno Board

6.6.4 Setting up date and Time to DS3231 Real Time Clock

Date and Time of the module has to be initialize when module use in first time. Code for initialization illustrate in Appendix A - Figure 7. Personal computer mother board also inbuilt similar or even the same CR2032 battery. The RTC battery purpose is keep clock without fail when power failure Figure 6.6 shows component of DS3231.

Figure 6.6: DS3231 Real Time Clock

6.7 Implantation of Micro SD Card Module

The Micro SD (Secure Digital) module use to keep big amount of data as storage. This module enables to plug a Micro SD card to read and write data. Micro SD Cards which operating voltage is 3.3 V. The Module has six pins, two for powering the module, the VCC and the GND pins, and four more pins for the Serial Peripheral Interface (SPI) communication.

6.8 Interconnection of Micro SD Card Module and Arduino Uno Board

Micro SD Card Module	Arduino Uno Board
CS	4
SCK	13
MOSI	11
MISO	12
VCC	5v
GND	GND

Table 6.5: Interconnection of Micro SD Card Module and Arduino Uno Board

6.8.1 Preparation of Data Logger

The SD library allows for reading and writing to SD card. The library supports FAT16 and FAT32 file systems on standard SD cards and SDHC cards. It uses short 8.3 names for files (8 Byte for file name and 3 Byte for extension). The file names passed to the SD library functions can include paths separated by forward-slashes, /, e.g. "directory/filename.txt". Because the working directory is always the root of the SD card, a name refers to the same file whether or not it includes a leading slash (e.g. "/file.txt" is equivalent to "file.txt").

The communication between the microcontroller and the SD card uses Serial Peripheral Interface (SPI), which takes place on digital pins 11, 12, and 13 (Arduino Uno board). Additionally, another pin must be used to select the SD card. This can be the hardware CS pin - pin 4 (Chip Select). Figure 6.7 illustrate initializing SD card before data write.

```
If (ChipSelectPin){
Print ("initialization failed!");}
 Else
{Print ("initialization done.");}
 End if
 If (file not found) {
Prin("SD File Open ERROR !!");}
 End i
```

Figure 6.7: Preparation of Data Logger

6.8.2 Formatting SD card before use

SD card need to format before used to save data. If SD card not properly formatted error massage will be generated. SD card below the 4GB required to format using FAT 16. SD card above the 4GB required to format using FAT 32. Figure 6.8 shows an error massage when SD card not compatible.

```
Initializing SD card...Wiring is correct and a card is present.

Card type: SDHC
Could not find FAT16/FAT32 partition.
Make sure you've formatted the card
```

Figure 6.8: Formatting SD card

6.8.3 Data writing to SD card

pseudo in Figure 6.9 illustrate how data written in to the SD card.

```
 If (File) {
Print ("test.txt:");
       While (File is available)
       Write (File.read);
       File.close;}
Else
 {Print ("error opening test.txt");}
```

Figure 6.9: Data writing to SD card

6.9 Integrated different module to the main system

Individual module has been developed separately and required integrated to the main board to build proposed final product. According to the discussion had in design chapter Flow Sensor, Real Time Clock Module, SD Card Module and Wi-Fi Module required to connect to Arduino Uno Board.

6.9.1 Main programming code use to integrate hardware modules

Real Time Clock and SD Card Modules programme codes are running on Arduino Uno microcontroller board. ESP 8266 Wi-Fi Module program code run on chip on in built.

6.9.2 Integrated complete Water Meter

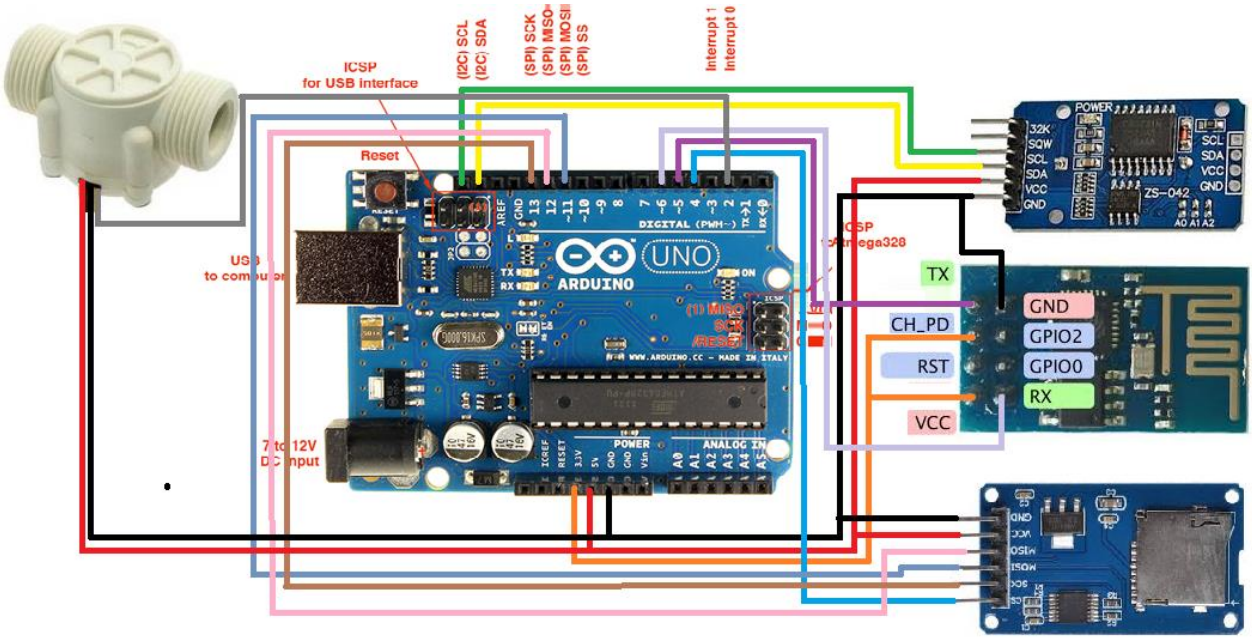

Figure 6.10: Water Meter Full Diagram

6.10 Customer and Meter Reader Mobile Application

Meter reader mobile application used to collect meter reading data on individual water meter through Wi-Fi connectivity. As describe earlier Xamarin supported to develop mobile app development.

6.10.1 WifiManager

```
java.lang.Object
    ↳ android.net.wifi.WifiManager
```

This class provides the primary API for managing all aspects of Wi-Fi connectivity. It can instantiate this class by calling **getSystemService** method [19]. It deals with following list of categories.

- The list of configured networks. The list can be viewed and updated.
- The currently active Wi-Fi network, if any. Connectivity can be established or turn down.
- Results of access point scans, containing enough information to make decisions about what access point to connect be to.

API to manage all aspects of WIFI connectivity refer Appendix C - Figure 1 to illustrate all the available Wi-Fi connection. In order to scan a list of wireless

networks need to register BroadcastReceiver. It will be able to scan a list of wireless networks as in Figure 6.11.

```
{WifiManager.ScanResultsAvailableAction })]

 private class WifiCalBacks : BroadcastReceiver
{
 public override void OnReceive(Context context, Intent
 intent)}
```

Figure 6.11: WifiManager ScanResults

When wifimanager display all the available Wi-Fi connection and user needs to select Wi-Fi connectivity required to connect. System requested password for particular Wi-Fi and user has to enter the password according to Appendix C - Figure 2. Validation of user name and password done as follows in Figure 6.12.

```
Dialog.SetTitle("User Login");

cmdLogin.Click += (s, e) => {
if (txtUserName.Text.Trim().Length == 0)
{
Toast.MakeText(Context, "Please enter the username",
ToastLength.Long).Show();
}
else if (txtPassword.Text.Trim().Length == 0)
{
Toast.MakeText(Context, "Please enter the password",
ToastLength.Long).Show();
```

Figure 6.12: User Login

6.10.2 Wi-Fi Protected Access (WPA)

Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access II (WPA2) are two security protocols and security certification programs developed by the Wi-Fi Alliance to secure wireless computer networks [20].

A pre-shared key (PSK) is a shared secret which was previously shared between the two parties. The key derivation function is typically used to build a shared secret. Such systems always use symmetric key cryptographic algorithms. The term PSK is used in Wi-Fi encryption such as Wired Equivalent Privacy (WEP), Wi-Fi Protected Access (WPA). Those methods are called WPA-PSK and WPA2-PSK. The

Extensible Authentication Protocol (EAP) known as EAP-PSK. In all these cases, both the wireless access points (AP) and all clients share the same key [21] illustrate in Figure 6.13.

```
if (sRes.Capabilities.Contains("WPA"))
{
WPAPasswordEntry wpa = new WPAPasswordEntry();
wpa.res = sRes;
MainActivity.instance.FragmentManager.BeginTransaction()
.Add(wpa,"psk")
.Commit();} };
```

Figure 6.13: Wi-Fi Protected Access

After successful logging application will be display function menu shows in Appendix C - Figure 4. Current Reading Button press system will display Current bill. Read whole button press system will extract data during last month to billing up to date. It will save to meter reader mobile device. Meter reader will upload all the meter reading text file to the water board severs for billing. Turn the device off function can disconnect water supply when customer laps payment last three months. Meter reader has privilege to execute the function when it required to do so.

6.11 Summary

In this chapter, it was explained in detail how the implementation done for SWMS, based on the design which was describe in Chapter 5. In next chapter it will be discussed how this implemented system was evaluated by a selected set of users.

Chapter 7

Evaluation of SWMS

7.1 Introduction

Smart Water Meter System evaluation according to the implementation discussed on previous chapter. This chapter will describe how to evaluate integrated hardware modules, Customer mobile application, Meter reader mobile application and Water Board Billing Application.

7.2 Evaluation Methodology

In order to evaluate the mobile application functionality and performance, randomly selected set of users were employed. They were used system and application as actual production environment. SWMS presented to particular users and instructed them to how to use application as normal working environment. Predefined questionnaire used to get feedback from the users. The system was deployed in a local environment for the purpose of evaluation. SWMS consists of two mobile and a Water Board Billing Application. Hence different set of questions were used to evaluated the system. The following areas were evaluated in the system evaluation.

a) Customer Mobile Application Evaluation
    - Usability (5 questions)
    - System functionality (5 questions)
    - Overall Impression (5 questions)

b) Meter Reader Mobile Application Evaluation
    - Usability (5 questions)
    - System functionality (5 questions)
    - Overall Impression (5 questions)

c) Web Application Evaluation
    - Usability (5 questions)
    - System functionality (5 questions)
    - Overall Impression (5 questions)

For each question on system evaluation sections, following evaluation weight were assigned for the answers (assign weight is in the bracket).

1 - Very poor (2), 2 – Poor (4), 3 – Average (6), 4 - Good (8), 5 - Excellent (10)

The evaluation was design using statistical analytical method. Microsoft Excel used as a tool to analyze data. A critical line was defined and the calculated average point for each question. It had been checked against to the critical line to make a final decision. There are 10 customers and 10 Meter Readers participated survey and weighted ranked mention as follows.

$$\text{Average points for evaluation} = \frac{(\text{Very poor}*2+\text{Poor}*4+\text{Average}*6+\text{Good}*8+\text{Excellent}*10)}{\text{No. of Users}}$$

Critical line of score = 40%

7.3 Evaluation Forms

Please refer Appendix J for evaluation forms.

7.4 Final Evaluation Results

Each and every evaluation forms had been entered to EXCEL sheet. Collected data analysis done under Usability, Functionality and Overall Impression.

7.4.1 Customer Mobile Application

Table 7.1 shows how customers comment on customer Mobile application questioner and average score 75.6. SWMS analysis results of the evaluation on Customer Mobile Application illustrate on Appendix J.

Description	Score
Usability	73.6
Functionality	78.8
Overall Impression	74.4
Average	**75.6**

Table 7.1: Summary of Evaluation on Customer Mobile Application

7.4.2 Meter Reader Mobile Application

Table 7.2 shows how Meter Readers comment on Meter Reader Mobile application questioner and average score 75.9. SWMS analysis results of the evaluation on Customer Mobile Application illustrate on Appendix J.

Description	Score
Usability	74.8
Functionality	81.2
Overall Impression	71.6
Average	**75.9**

Table 7.2: Evaluation of Meter Reader Mobile Application

7.4.3 Water Board Billing Application

Table 7.3 shows how Meter Readers comment on Water Board Billing Application questioner and average score 71.9. SWMS analysis results of the evaluation on Water Board Billing Application illustrate on Appendix L.

Description	Score
Usability	70.8
Functionality	74.8
Overall Impression	70.0
Average	**71.9**

Table 7.3: Evaluation of Meter Reader Mobile Application

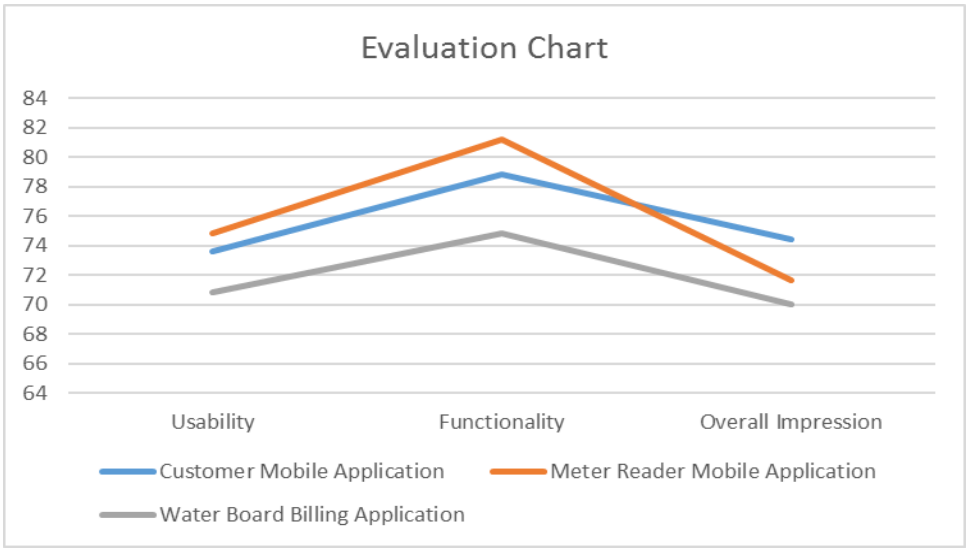

Figure 7.1: Evaluation Chart

According to the evaluation chart in Figure 7.1, all the evaluation points are above the defined critical point (Critical point = 40). Therefore, system had been approached its targeted evaluation points.

7.5 Summary

According to the evaluation results, the system had maintained accepted quality in all aspects of the evaluation of Customer Mobile application, Meter Reader Mobile application and Water Board Billing Web application. In the next chapter, conclusion and further possible enhancements for SWMS will be discussed.

Chapter 8

Conclusion and further work of SWMS

8.1 Introduction

In the above chapter 7, the System had been validated by selected set of users. In this chapter discussed identified enhancements, features and further work.

8.2 Conclusion

The customers, Water Meter Reader and Water Board are mainly benefited by the proposed system. The currently system, Customers are unable to retrieve information on current month water consumption and cost after the last water bill (Unbilled period). Advantage of new system, Customers will be able to get information on their usage at any time (instant). Meter readers are able to collect meter reading data electronically efficient than earlier system. Water Board can easily detect water leaks in customer premises when analysis individual customer data. It will prevent losing money without getting benefit. System will provide information on water consumption every 15 Minutes and it will help to analyze water usage pattern and predict the water demand in future.

Following features of SWMS system manage entire national pipe borne water system efficient manner.

- Capture meter reading
- View water consumption
- Analyze the consumption pattern and demand management of water
- Estimate household water consumption
- Analyze water-saving potential for the residential sector

The main risk has been identified the system require 5v power to operation. System can connect to national electricity grid. when power failure system has to turn to backup power (UPS). The national electricity grid has not been covered in particular areas, solar power can be used as a substitute.

The SWMS system eligible to collect customer water consumption data. This data collection approach would be a new experience to customers and Meter Readers. Therefore, such a set of data could be used to generate new knowledge.

8.3 Further work

Web base payment module can be introducing to pay online when customer view pending bill payment. Automated meter reading data collection technique can introduce through the customer home broadband. Water supply can be disconnected automatically when laps more than three bills. Prepaid water billing system can be introducing according to the customer requirement. Water leaks have been detected by the system and produce SMS alert to the customer.

8.4 Summary

This chapter concludes the thesis by describing the Smart water meter system and how it can further improve with new features.

References

[1] L.-L. Nguyen, H.-T. Huynh, and T.-D. Nguyen, “A Low Cost and Low Power Consumption Automatic Water Meter Reading System: Hardware Investigation and Network Design,” *REV J. Electron. Commun.*, vol. 4, no. 3–4, 2015.

[2] K. Ashna and S. N. George, “GSM based automatic energy meter reading system with instant billing,” in *Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on*, 2013, pp. 65–72.

[3] G. L. Prashanthi and K. V. Prasad, “Wireless power meter monitoring with power theft detection and intimation system using GSM and Zigbee networks,” 2014.

[4] M. D. Oskouei and S. N. Razavi, “A Study on WiMAX: IEEE 802.16 Standard,” *Parameters*, vol. 802, pp. 802–16, 2015.

[5] P. R. Daware and S. S. Patil, “A Review on Intelligent Automatic Meter Reading and E-Billing System using Power Line Communication,” 2013.

[6] D. N. Thanth and A. Borole, “Wi-Fi Based Smart Energy Meter,” *IJRCCT*, vol. 4, no. 4, pp. 282–285, 2015.

[7] M. T. Islam, R. C. Talukder, and others, “Design & Development of a Microcontroller Based Water Flow Control System Using Servo Motor,” East West University, 2016.

[8] G. Kaur and S. Malhotra, “A Hybrid Approach for Data Hiding using Cryptography Schemes.”

[9] Omega,” Liquid Flowmeters”, 21-Nov-1984, [Online]. Available: http://www.omega.com/techref/flowcontrol.html [Accessed: 22-March-2017]

[10] Wi-Fi Protected Access, “security protocols”,16-April-2017, [Online].
https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access.
[Accessed: 22-March-2017]

[11] tutorialspoint, “Cloud Computing Architecture”, 16-April-2017, [Online].
https://www.tutorialspoint.com/cloud_computing/cloud_computing_architecture.htm
[Accessed: 22-March-2017]

[12] Arduino, “Getting Started”, 01-April -2017, [Online].
https://www.arduino.cc/en/Guide/Introduction
[Accessed: 01-April -2017]

[13] Arduino, “Arduino/Genuino UNO”, 01-April -2017, [Online].
https://www.arduino.cc/en/main/arduinoBoardUno
[Accessed: 11- March -2017]

[14] Maxim Integrated, “Extremely Accurate I²C-Integrated RTC/TCXO/Crystal”,
01-April -2017, [Online].
https://www.maximintegrated.com/en/products/digital/real-time-clocks/DS3231.html/
[Accessed: 11- March -2017]

[15] OpenElectronics,“Top 5 Wireless Ways to Communicate with your Controller”,
11- March -2017, [Online].
https://www.open-electronics.org/top-5-wireless-ways-to-communicate-with-your- controller/, [Accessed: 11- March -2017]

[16] Sparkfun,” Serial Communication”, 18- April -2017, [Online].
https://learn.sparkfun.com/tutorials/serial-communication/uarts
[Accessed: 18- April -2017]

[17] xamarin, “Introducing Xamarin Studio”, 21- April -2017, [Online].
https://developer.xamarin.com/guides/cross-platform/xamarin-studio/,
[Accessed: 21- April -2017]

[18] Developers, “WifiManager”, 21- April -2017, [Online].
https://developer.android.com/reference/android/net/wifi/WifiManager.html,
[Accessed: 21- April -2017]

[19] TutorialsPoint,” Android-Wi-Fi”, 21- April -2017, [Online].
https://www.tutorialspoint.com/android/android_wi_fi.htm
[Accessed: 21- April -2017]

[20] Wikipedia,” Wi-Fi Protected Access”, 21- April -2017, [Online].
https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access
[Accessed: 21- April -2017]

[21] Wikipedia, “Pre-shared key”, 23- April -2017, [Online].
https://en.wikipedia.org/wiki/Pre-shared_key
[Accessed: 23- April -2017]

[22] worldtimezone,“GSM World Coverage Map and GSM Country List”,
25- April -2017, [Online].
http://www.worldtimezone.com/gsm.html
[Accessed: 25- April -2017]

[23] webopedia, “Wi-Fi (wireless networking)”, 25- April -2017, [Online].
http://www.webopedia.com/TERM/W/Wi_Fi.html
[Accessed: 25- April -2017]

[24] Wikipedia, “Volumetric flow rate”, 26- April -2017, [Online].
https://en.wikipedia.org/wiki/Volumetric_flow_rate
[Accessed: 26- April -2017]

[25] eprolabs,” Flow Sensor YF-S201”, 26- April -2017, [Online].
https://wiki.eprolabs.com/index.php?title=Flow_Sensor_YF-S201
[Accessed: 26- April -2017]

[26] “Serial-to-WiFi Tutorial using ESP8266”, 26- April -2017, [Online].
http://fab.cba.mit.edu/classes/863.14/tutorials/Programming/serialwifi.html
[Accessed: 26- April -2017]

[27] instructables, “Using the ESP8266 Module”, 26- April -2017, [Online].
http://www.instructables.com/id/Using-the-ESP8266-module/
[Accessed: 26- April -2017]

[28] Wikipedia, “Ciphertext”, 26- April -2017, [Online].
https://en.wikipedia.org/wiki/Ciphertext
[Accessed: 26- April -2017]

Appendixes

Appendix A - Configuration and installing modules

An USB TTL adapter driver has to be installing to the PC before connecting adapter. USB to Serial Com port driver not install Arduino IDE unable to detect port illustrate in Figure 1.

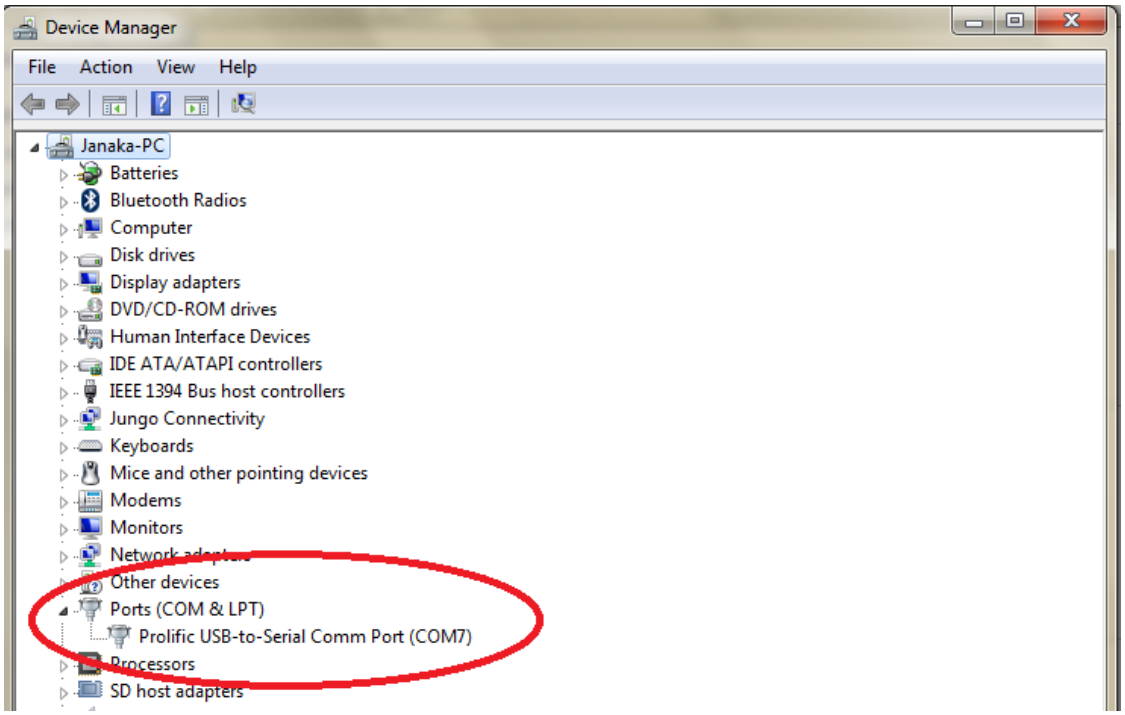

Figure 1: Device Manager

USB TTL converter Connect to the computer to upload programme and comport has to select in Arduino IDE when there was more than one com port activated illustrate Figure 2.

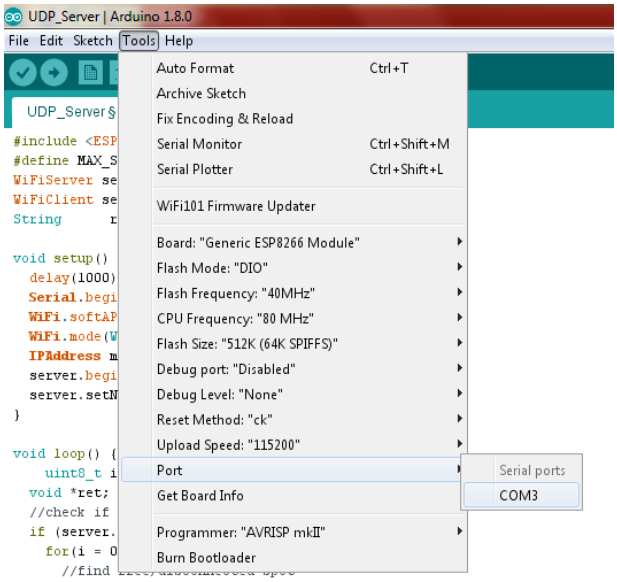

Figure 2: Com Port Selection

When connecting specific modules to Arduino IDE needs to install specific program supplied by device vender Figure 3.

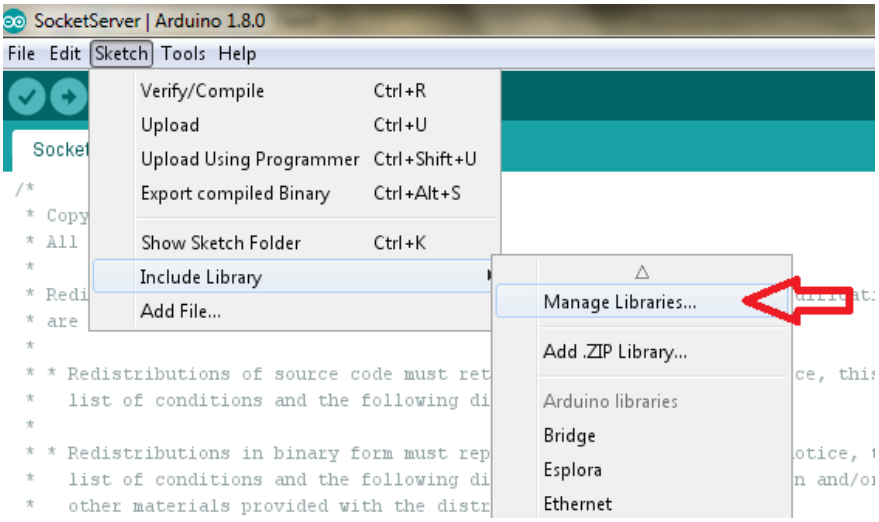

Figure 3: Include Libraries

Install ESP8266 library to Arduino IDE showing Figure 4.

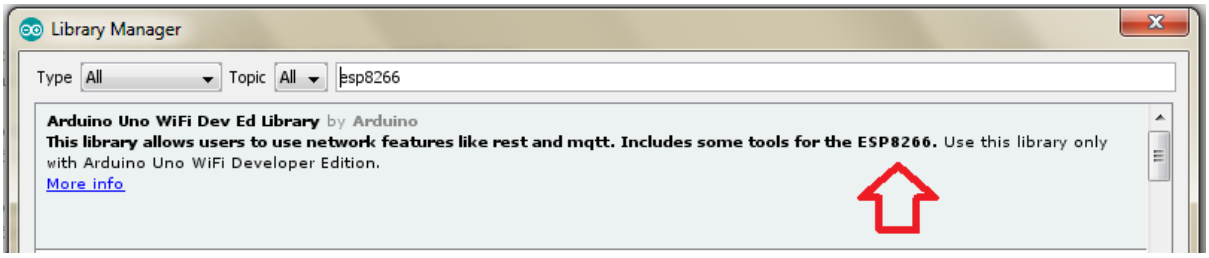

Figure 4: Install ESP8266 library

Before upload Arduino IDE, board has changed to “Generic ESP8266 Module” as in Figure 5.

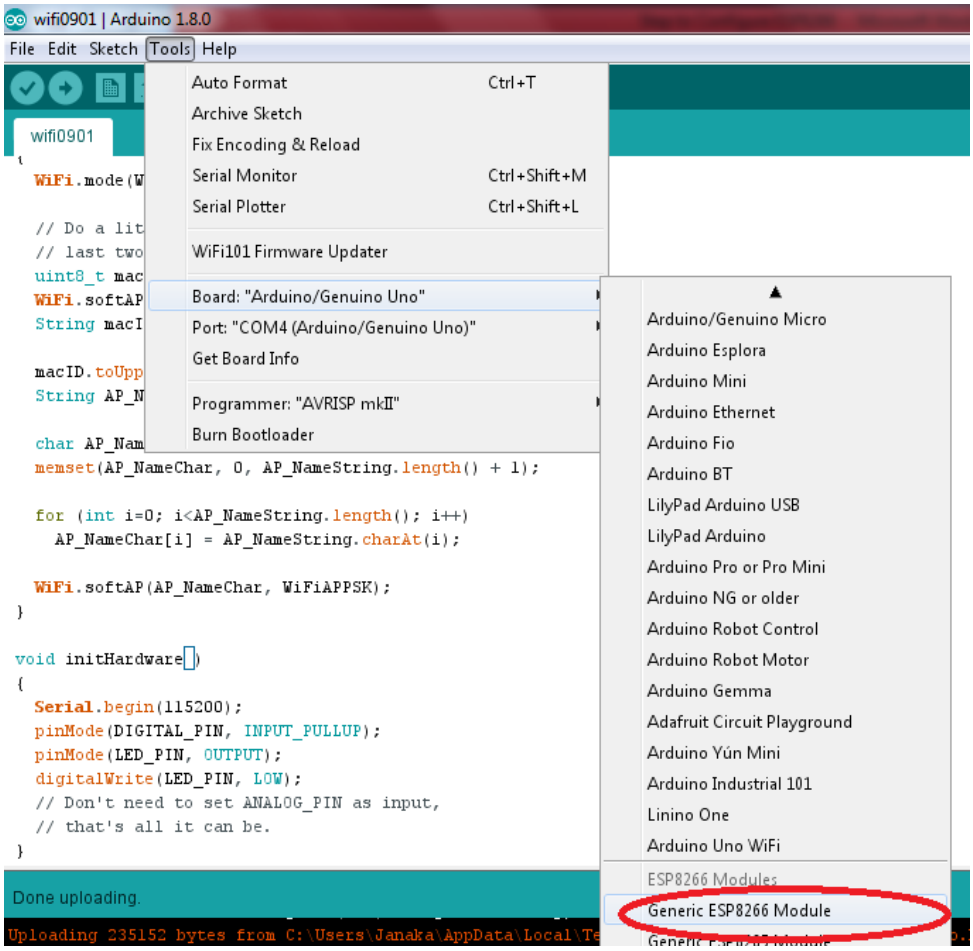

Figure 5: Generic ESP8266 Module Selection

Arduino IDE needs to install DS3231 module library supplied by device vender Figure 6.

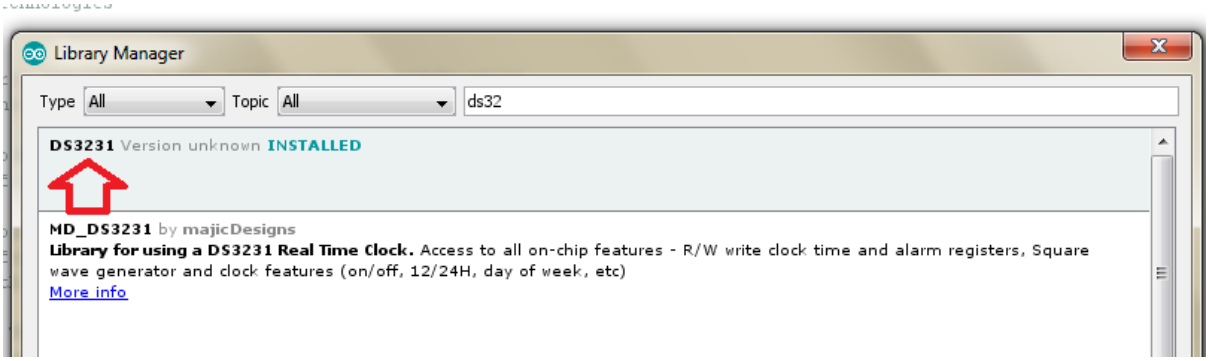

Figure 6: Install DS3231

Date and Time of the module initialization coded as follows Figure 7.

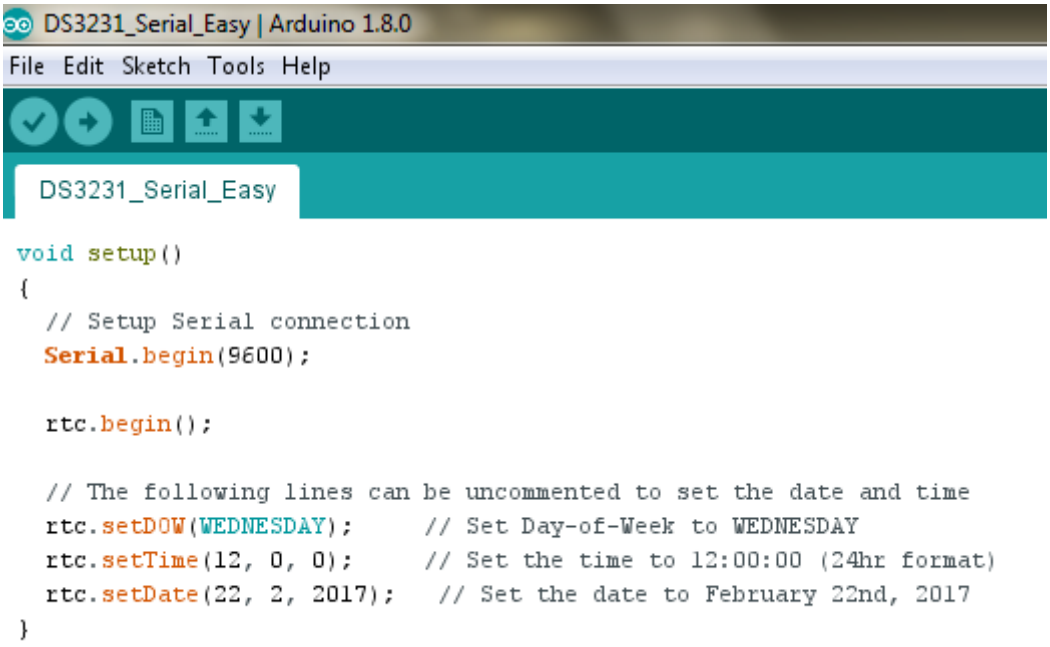

Figure 7: ESP 8266 Date and Time Module initialization

Initializing SD Card

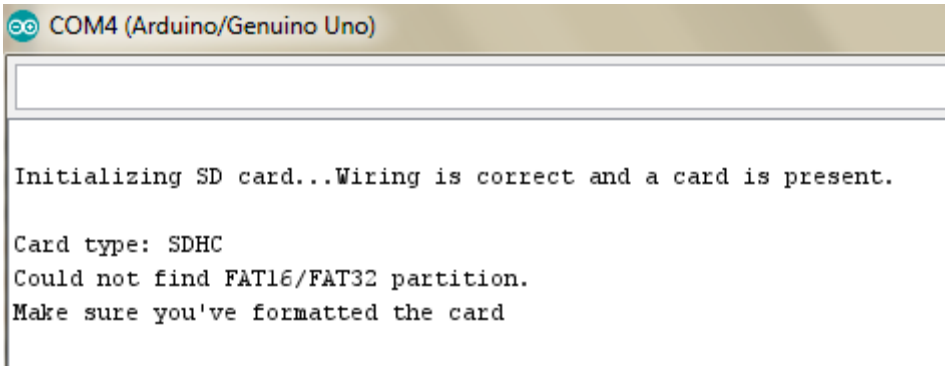

Figure 8: Initializing SD Card

Appendix B – Configuration of ESP8266 (Different Method)

1. Selection of Uno Board

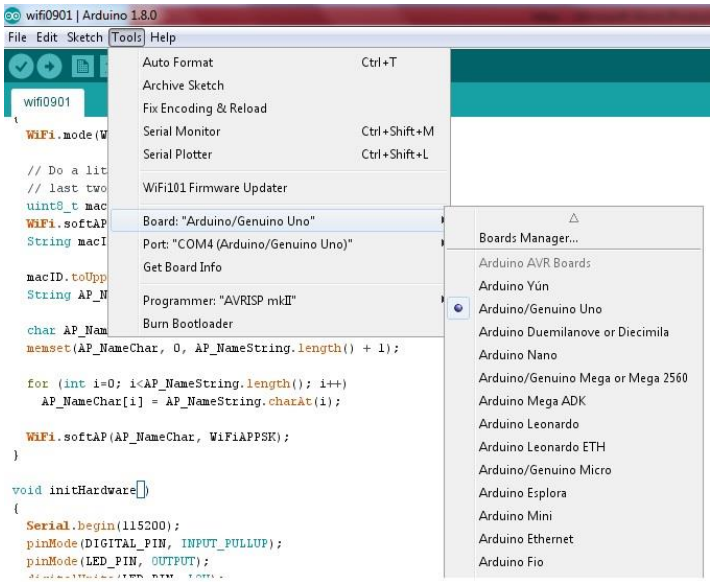

Figure 1: Selection of Board

2. Blank code Uploading to board

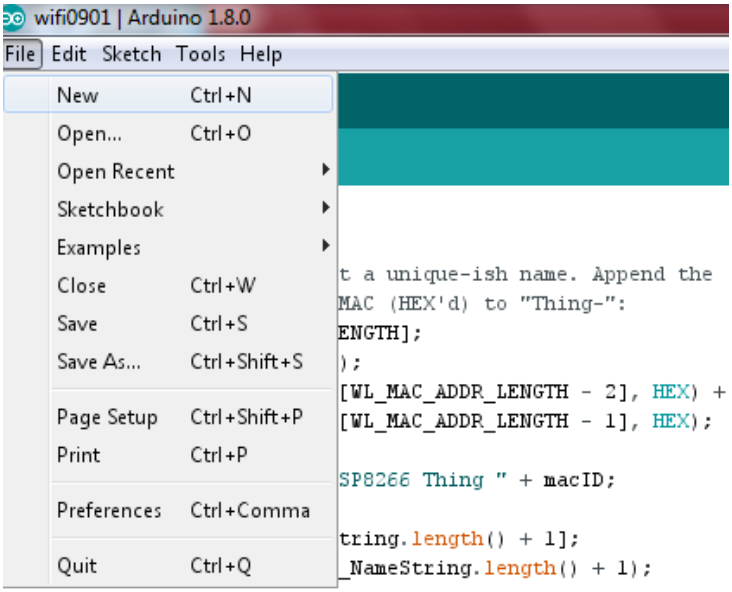

Figure 2: Selection of Blank code

3. Blank code ready for upload

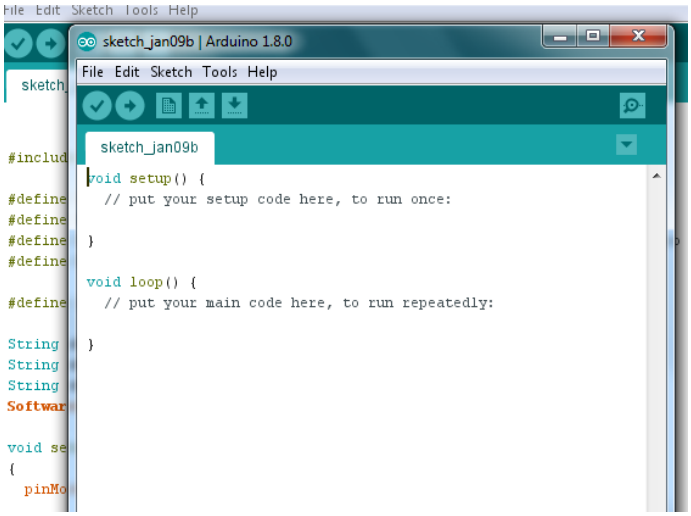

Figure 3: Blank code

4. Selection of Generic ESP8266 Module

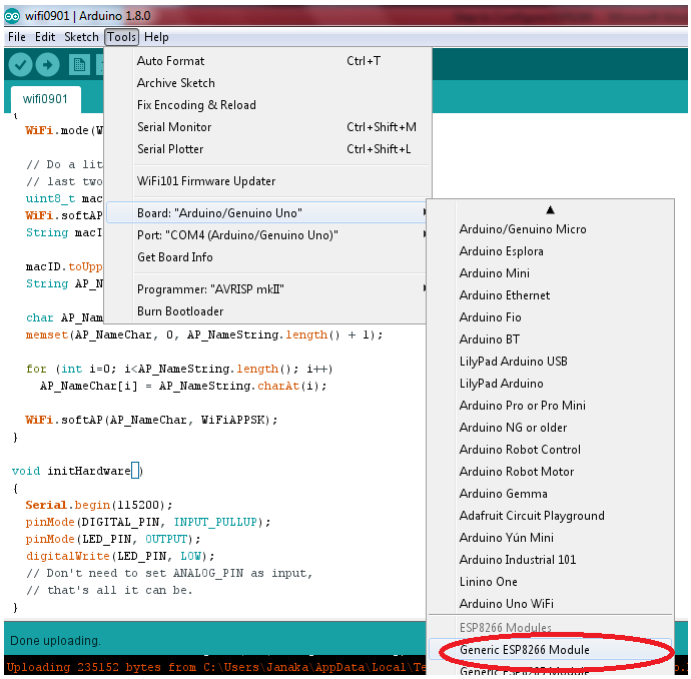

Figure 4: Generic ESP8266 Module Selection

5. Selected Generic ESP8266

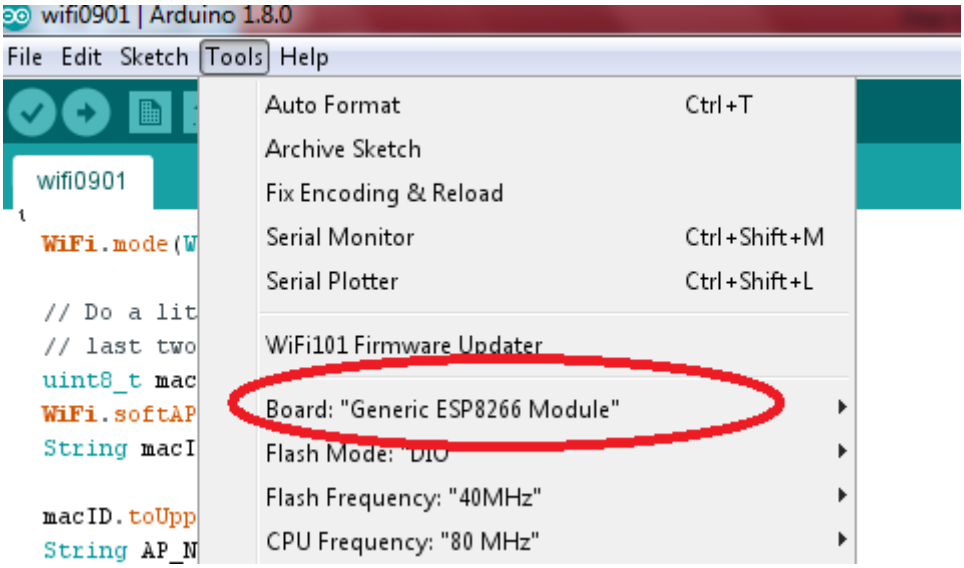

Figure 5: Generic ESP8266 Module Selected

6. Code Compilation

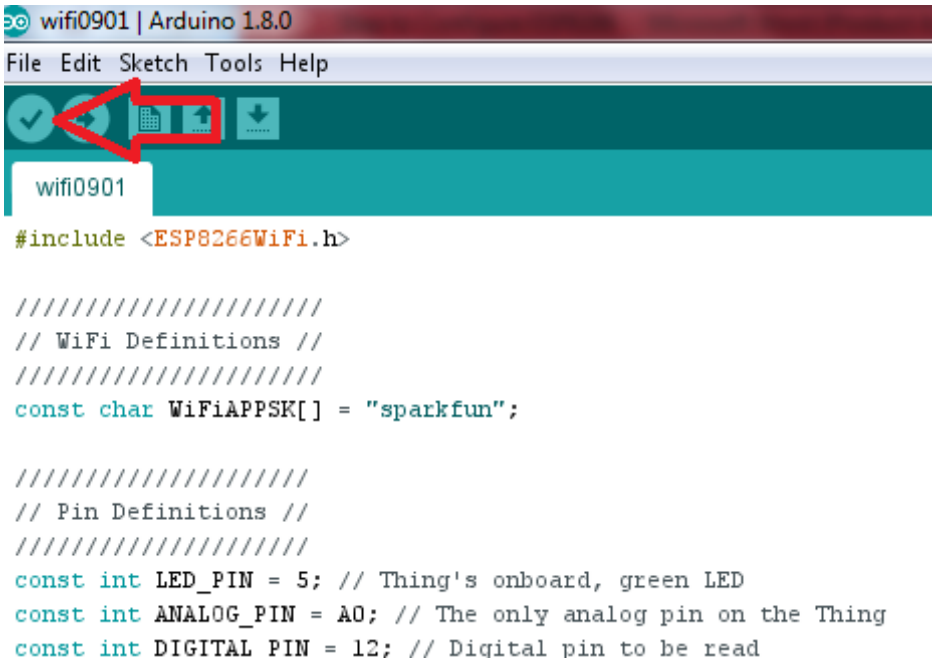

Figure 6: Code Compilation using command Button

7. Upload code in to ESP8266

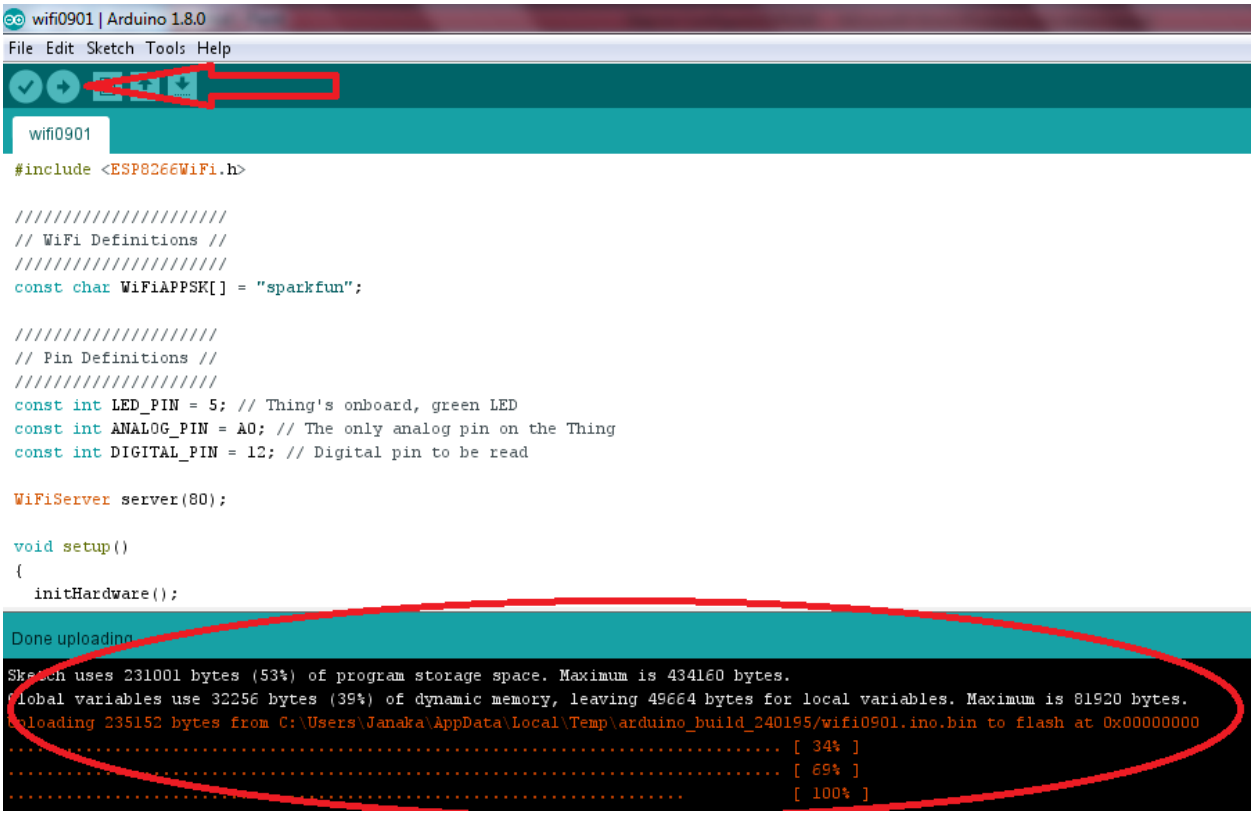

Figure 7: Code uploading to ESP8266 Module

8. Before Connecting to ESP8266 Wi-Fi

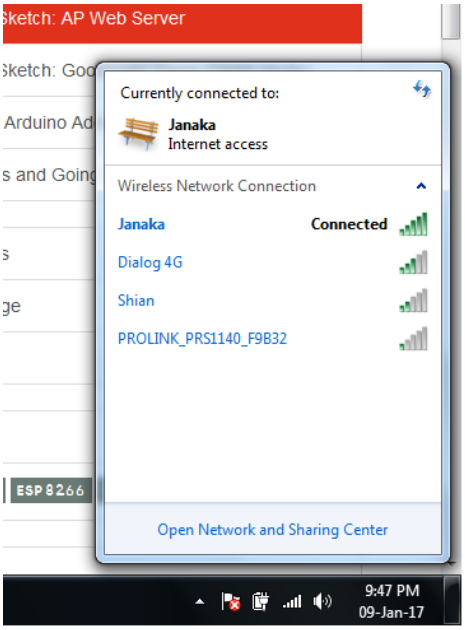

Figure 8: Before Connecting to ESP8266

9. After Connecting ESP8266 Wi-Fi

Figure 9: ESP8266 Wi-Fi connected

10. Check IP configuration

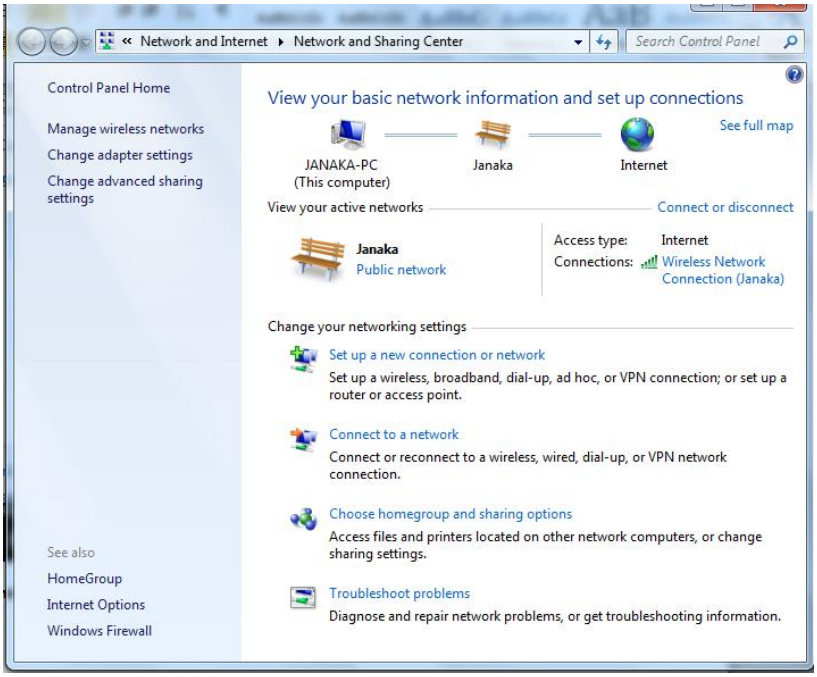

Figure 10: Network Sharing Center

11. Network and Sharing Center Properties

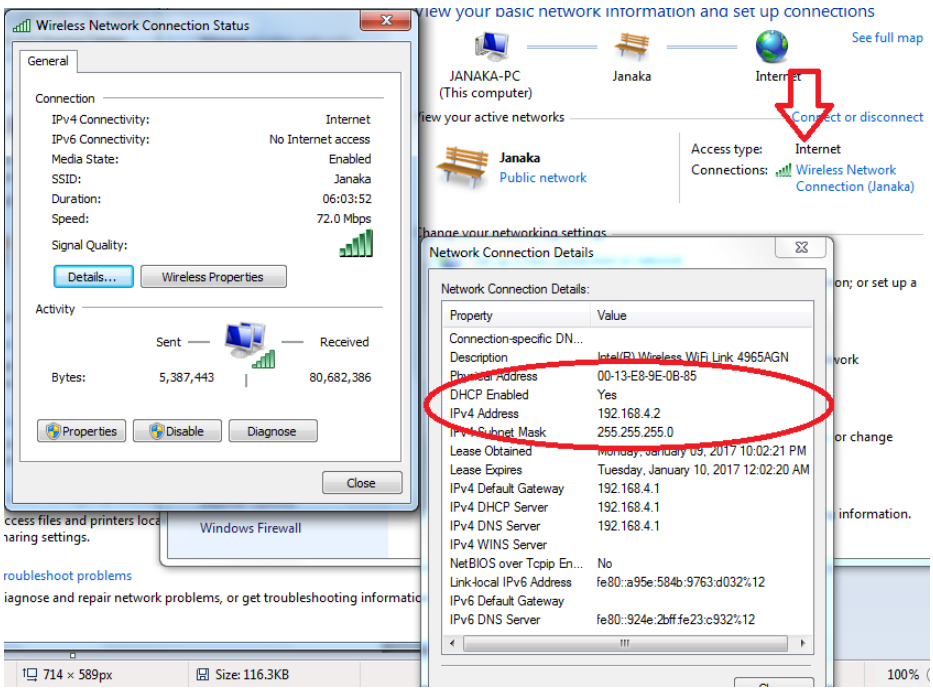

Figure 11: ESP8266 Wi-Fi Connection Details

12. Check Wi-Fi Connection using Command Prompt

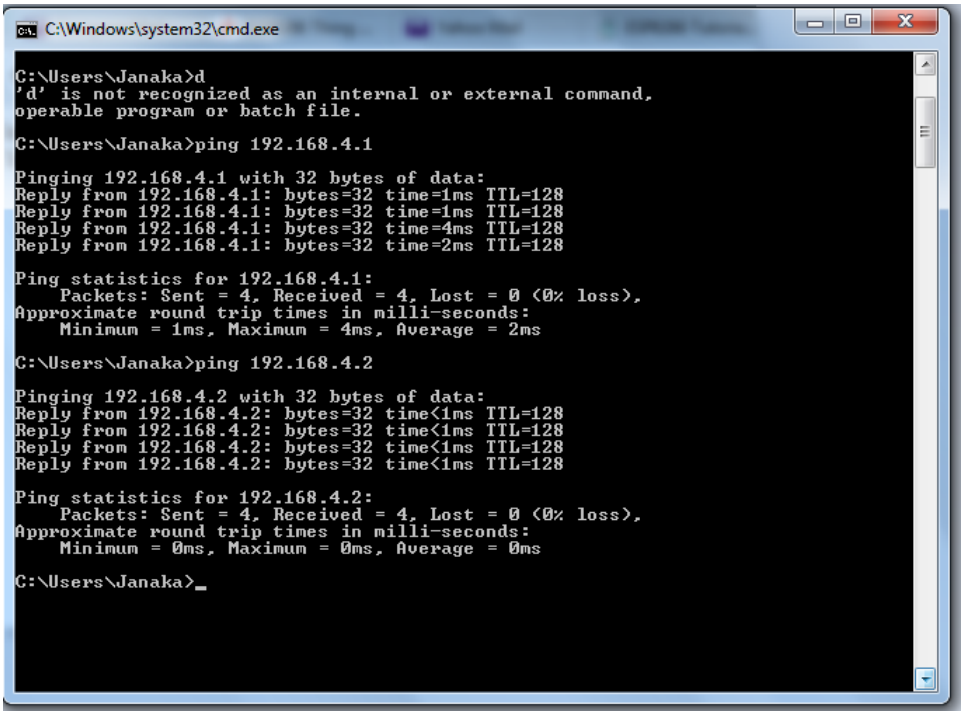

Figure 12: Check connectivity

Appendix C – User interfaces for Customer and Meter Reader Mobile Application

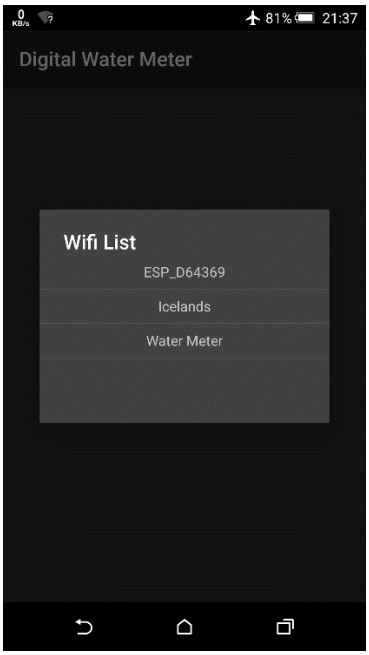

Figure 1: List of Wi-Fi connection

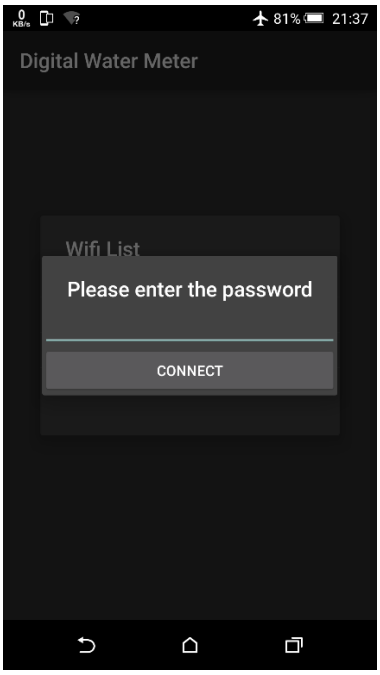

Figure 2: Logging to selected Wi-Fi

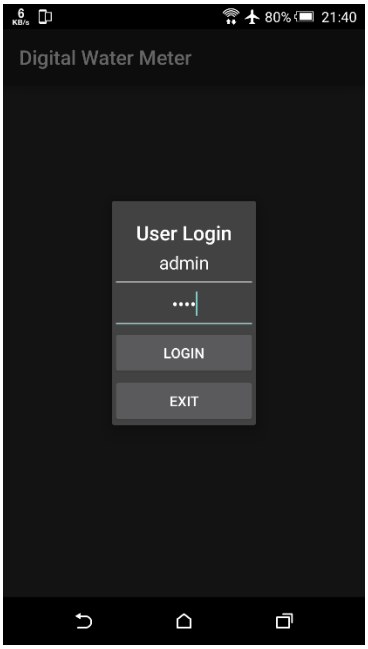

Figure 3: Admin Logging to selected Wi-Fi

Figure 4: Menu of Meter Reader

Appendix D – User interface designs for Water Board Billing Application

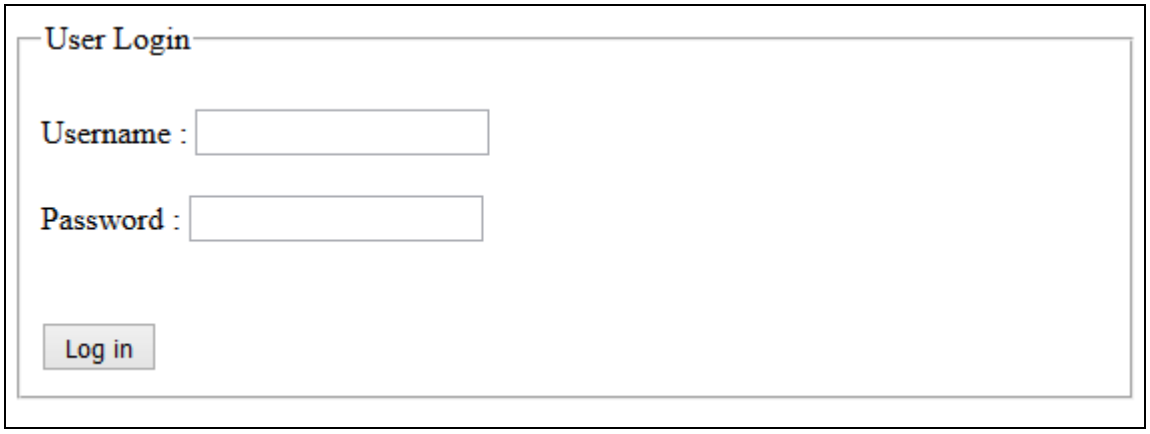

Figure 1: User Logging to the Water Board Billing System

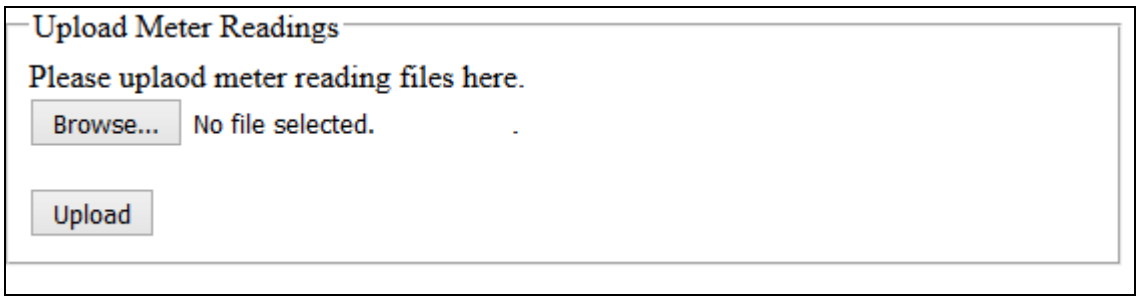

Figure 2: Upload Meter Reading Text File to Server

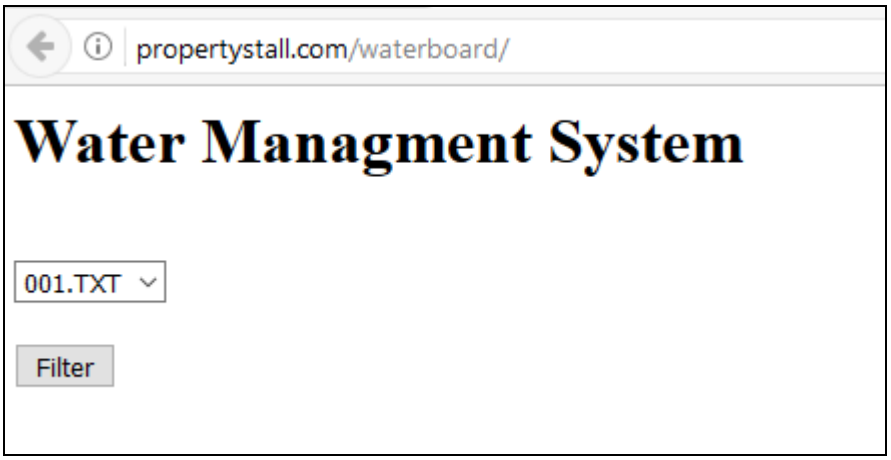

Figure 3: Select Text File to View Water Bill

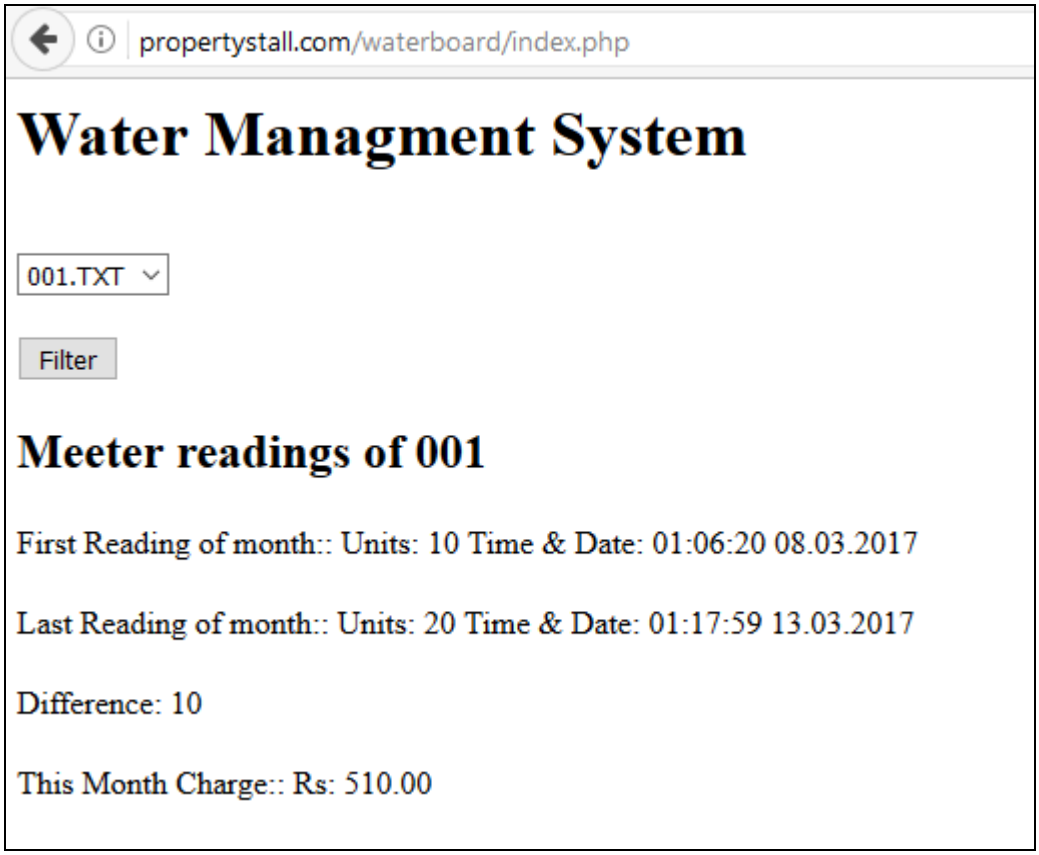

Figure 4: Current Bill for Selected Customer

Appendix E – ER Diagram

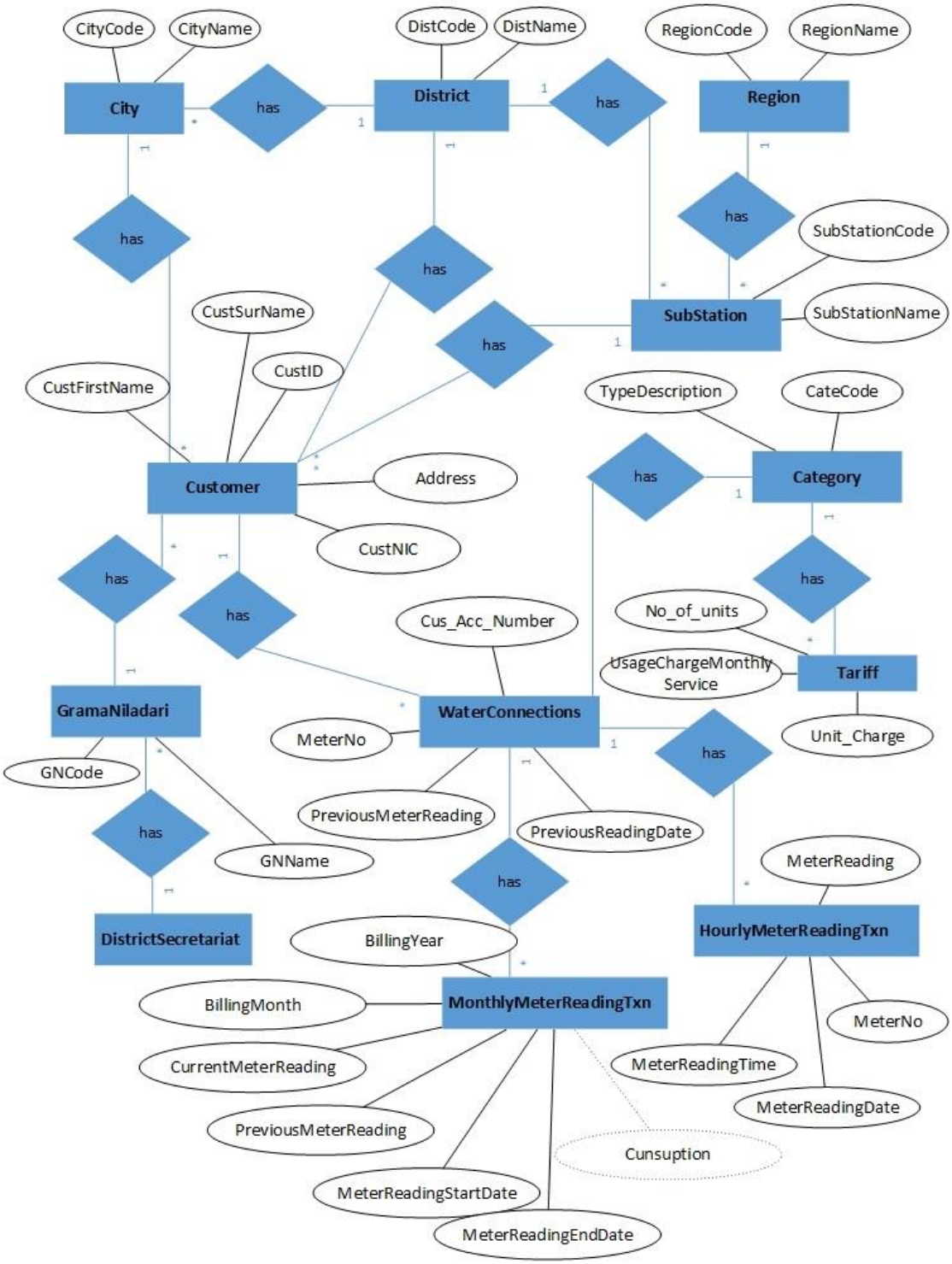

Figure 1: Water Board Billing

Appendix F – Sequence Diagram

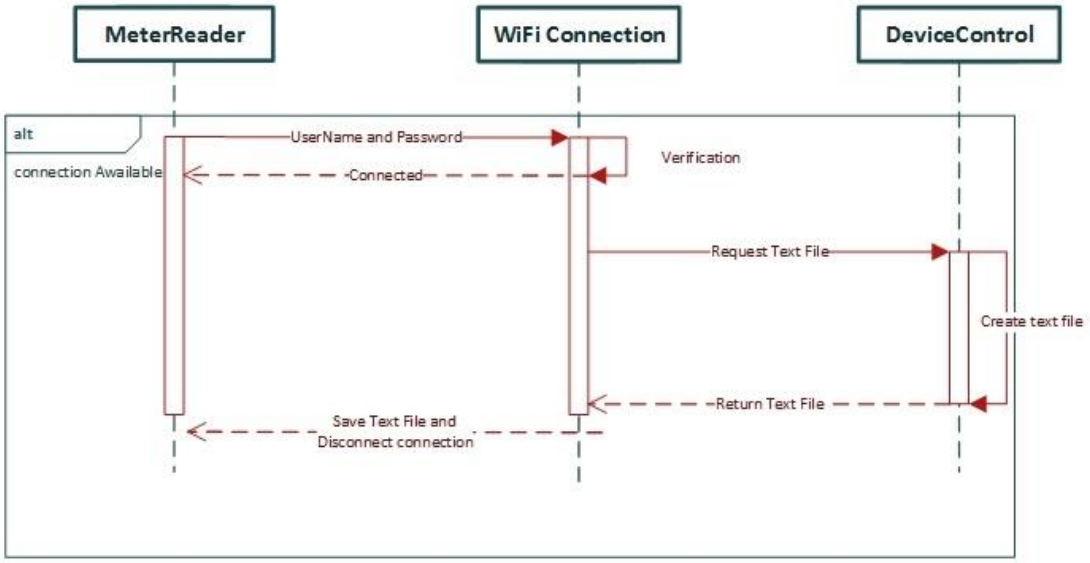

Figure 1: Meter Reader Text File Download

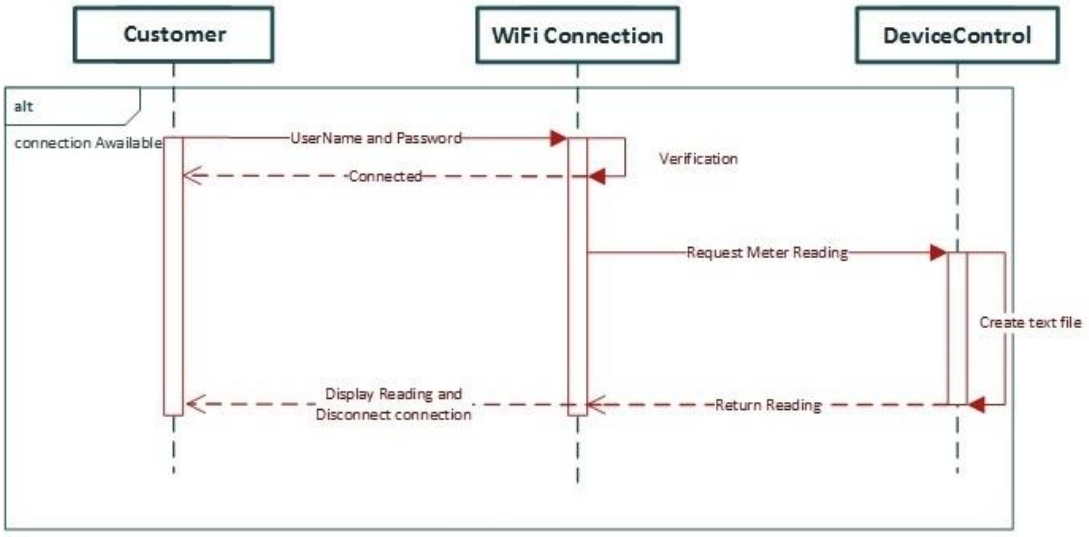

Figure 2: Customer Read Current Meter Reading

Appendix G – Use Case Diagram

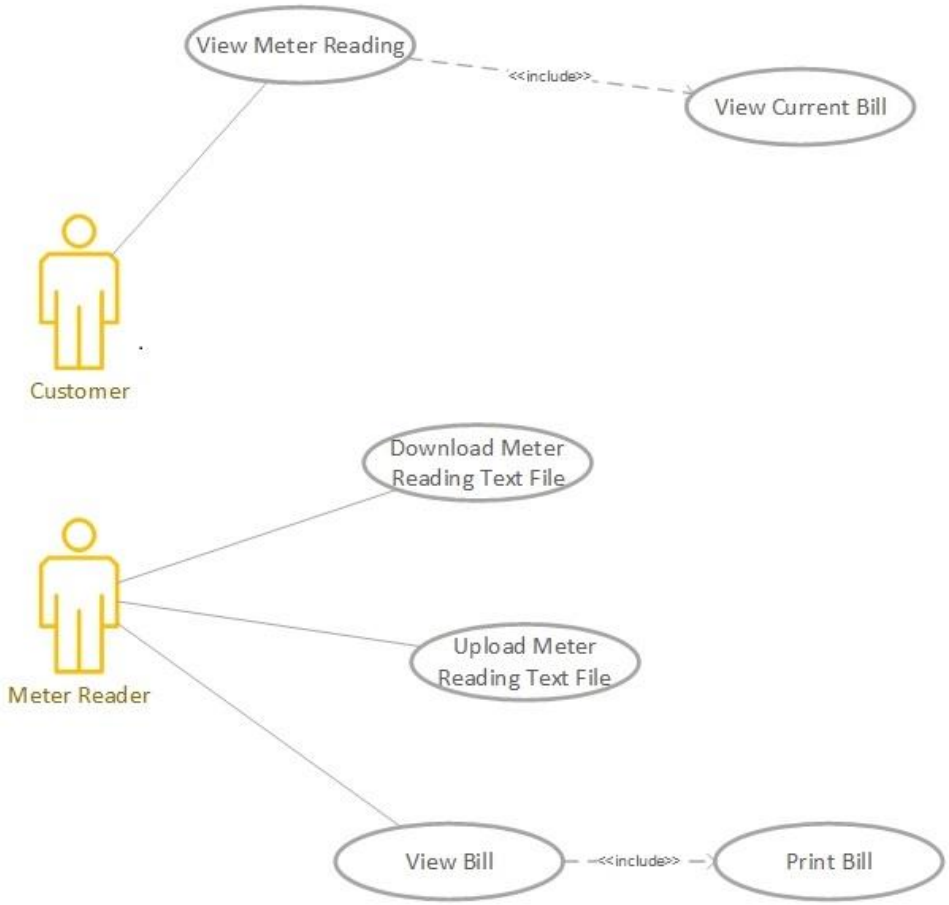

Figure 1: Use Case Diagrams

Appendix H – Flow Chart

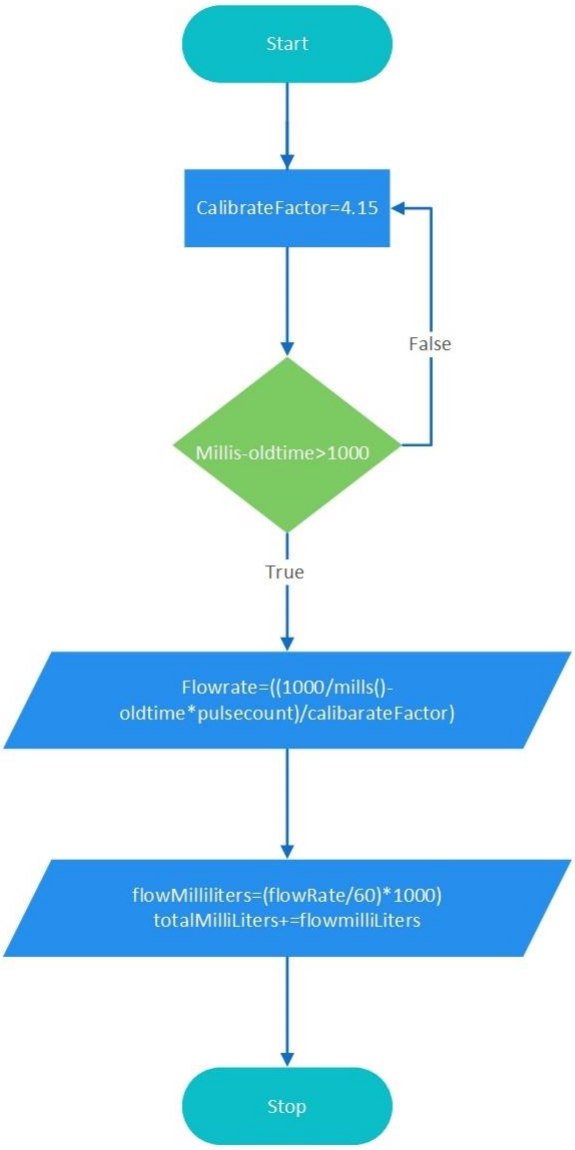

Figure 1: Water Flow Reading

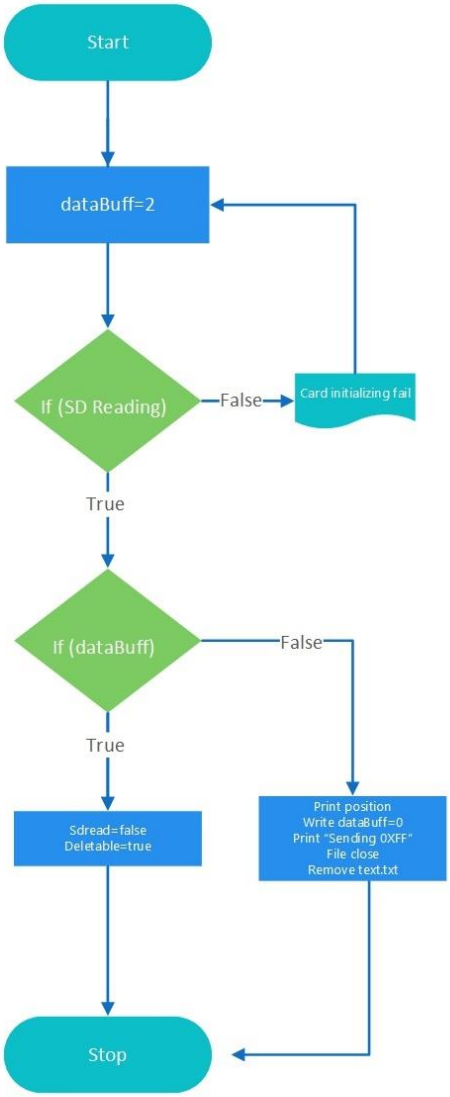

Figure 2: SD Card Reading

Appendix I – System Evaluation Forms

SWMS Evaluation of Usability – Customer Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Can you understand function of the					
2	Easy to learn functions by your self					
3	Time taken to system feedback when execute function					
4	Are you happy with the information provided by the system					
5	Is system give proper feedback and massages to guide you to achieve your					

Table 1: Usability – Customer Mobile Application

SWMS Evaluation of Functionality – Customer Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Does application provide adequate functions to get information on current water consumption and other information you required base on					
2	Does application giving clear expected output for your actions					
3	When comparing with existing systems for the same purpose, how would you rank this application					
4	Are you satisfied with the functionality offered by this					
5	Rank your expectation and system					

Table 2: Functionality – Customer Mobile Application

SWMS Evaluation of Overall Impression – Customer Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Does system provide the final result within your expected time scope with expected quality					
2	How would you categorize system, based on this feature					
3	Look and feel when application use regular basis					
4	Overall satisfaction with this application					
5	Rank your expectation and system functionality					

Table 3: Overall Impression – Customer Mobile Application

SWMS Evaluation of Usability – Meter Reader Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Can you understand function of the system					
2	Easy to learn functions by your self					
3	Time taken to system feedback when execute function					
4	Are you happy with the information provided by the system					
5	Is system give proper feedback and massages to guide you to achieve your goals					

Table 4: Usability – Meter Reader Mobile Application

SWMS Evaluation of Functionality – Meter Reader Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Are you satisfied time taken to copy file, from Water Meter to Mobile device					
2	Does application giving clear expected output for your actions					
3	When comparing with existing systems for the same purpose, how would you rank this application					
4	Are you satisfied with the functionality offered by this application					
5	Rank your expectation and system functionality					

Table 5: Functionality – Meter Reader Mobile Application

SWMS Evaluation of Overall Impression – Meter Reader Mobile Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Does system provide the final result within your expected time scope with expected quality					
2	How would you categorize system based on this feature					
3	Look and feel when application use regular basis					
4	Overall satisfaction with this					
5	Rank your expectation and system functionality					

Table 6: Overall Impression – Meter Reader Mobile Application

SWMS Evaluation of Usability – Water Board Billing Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Can you understand function of the system					
2	Easy to learn functions by your self					
3	Time taken to system feedback when execute function					
4	Are you happy with the information provided by the system					
5	Is system give proper feedback and massages to guide you to achieve your goals					

Table 7: Usability – Water Board Billing Application

SWMS Evaluation of Functionality – Water Board Billing Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	2	3	4	5
		Very poor	Poor	Average	Good	Excellent
1	Does application provide adequate functions to transfer data file and calculate water bill base on your personal experience					
2	Does application giving clear expected output for your actions					
3	When comparing with existing systems for the same purpose, how would you rank this application					
4	Are you satisfied with the functionality offered by this					
5	Rank your expectation and system functionality					

Table 8: Functionality – Water Board Billing Application

SWMS Evaluation of Overall Impression – Water Board Billing Application

1 - Very poor, 2 – Poor, 3 – Average, 4 - Good, 5 – Excellent

#	Evaluation criteria	Ranking				
		1	**2**	**3**	**4**	**5**
		Very poor	**Poor**	**Average**	**Good**	**Excellent**
1	Does system provide the final result within your expected time scope with expected quality					
2	How would you categorize system based on this feature					
3	Look and feel when application use regular basis					
4	Overall satisfaction with this					
5	Rank your expectation and system functionality					

Table 9: Overall Impression – Water Board Billing Application

Appendix J - Data Analysis - Evaluation of Customer Mobile App

Usability Participants – Customer No. of Participants - 10

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Average	Good	Excellent		
1	Can you understand function of the system	0	0	30	16	30	76	
2	Easy to learn functions by your self	0	0	18	32	30	80	
3	Time taken to system feedback when execute function	0	0	24	48	0	72	
4	Are you happy with the information provided by the system	0	4	30	16	20	70	
5	Is system give proper feedback and massages to guide you to achieve your goals	0	0	30	40	0	70	
		Usability					**368**	**73.6**

Table 1: Data Analysis of Usability, Customer Mobile App

Functionality

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Average	Good	Excellent		
1	Does application provide adequate functions to get information on current water consumption and other information you required base on your personal experience	0	0	24	24	30	78	
2	Does application giving clear expected output for your actions	0	0	12	56	10	78	
3	When comparing with existing systems for the same purpose, how would you rank this application	0	0	18	24	40	82	
4	Are you satisfied with the functionality offered by this application	0	0	12	40	30	82	
5	Rank your expectation and system functionality	0	0	36	8	30	74	
		Functionality					**394**	**78.8**

Table 2: Data Analysis of Functionality, Customer Mobile App

Overall Impression

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Average	Good	Excellent		
1	Does system provide the final result within your expected time scope with expected quality	0	0	18	40	20	78	
2	How would you categorize system based on this feature	0	0	12	48	20	80	
3	Look and feel when application use regular basis	0	0	24	48	0	72	
4	Overall satisfaction with this application	0	0	24	48	0	72	
5	Rank your expectation and system functionality	0	0	30	40	0	70	
		Overall Impression					**372**	**74.4**

Table 3: Data Analysis of Overall Impression, Customer Mobile App

Appendix K - Data Analysis - Evaluation of Meter Reader Mobile App

Usability Participants – Meter Reader No. of Participants - 10

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Ave rage	Good	Excel lent		
1	Can you understand function of the system	0	0	12	64	0	76	
2	Easy to learn functions by your self	0	0	6	64	10	80	
3	Time taken to system feedback when execute function	0	0	18	56	0	74	
4	Are you happy with the information provided by the system	0	0	30	32	10	72	
5	Is system give proper feedback and massages to guide you to achieve your goals	0	0	24	48	0	72	
		Usability					**374**	**74.8**

Table 1: Data Analysis of Usability, Meter Reader Mobile App

Functionality

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Ave rage	Good	Exce llent		
1	Are you satisfied time taken to copy file, from Water Meter to Mobile device	0	0	24	32	20	76	
2	Does application giving clear expected output for your actions	0	0	6	72	0	78	
3	When comparing with existing systems for the same purpose, how would you rank this application	0	0	0	24	70	94	
4	Are you satisfied with the functionality offered by this application	0	0	0	64	20	84	
5	Rank your expectation and system functionality	0	0	18	56	0	74	
		Functionality					**406**	**81.2**

Table 2: Data Analysis of Functionality, Meter Reader Mobile App

Overall Impression

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Average	Good	Excellent		
1	Does system provide the final result within your expected time scope with expected quality	0	0	18	56	0	74	
2	How would you categorize system, based on this feature	0	0	12	64	0	76	
3	Look and feel when application use regular basis	0	0	24	40	10	74	
4	Overall satisfaction with this application	0	0	30	40	0	70	
5	Rank your expectation and system functionality	0	0	48	16	0	64	
		Overall Impression					**358**	**71.6**

Table 3: Data Analysis of Overall Impression, Meter Reader Mobile App

Appendix L - Analysis of Evaluation – Water Board Billing App

Usability Participants – Meter Reader No. of Participants - 10

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Ave rage	Good	Exce llent		
1	Can you understand function of the system	0	0	36	32	0	68	
2	Easy to learn functions by your self	0	0	42	24	0	66	
3	Time taken to system feedback when execute function	0	0	6	72	0	78	
4	Are you happy with the information provided by the system	0	0	24	40	10	74	
5	Is system give proper feedback and massages to guide you to achieve your goals	0	0	36	32	0	68	
		Usability					**354**	**70.8**

Table 1: Data Analysis of Usability, Water Board Billing App

Functionality

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Ave rage	Good	Exce llent		
1	Does application provide adequate functions to transfer data file and calculate water bill base on your personal experience	0	0	36	32	0	68	
2	Does application giving clear expected output for your actions	0	0	30	40	0	70	
3	When comparing with existing systems for the same purpose, how would you rank this application	0	0	6	16	70	92	
4	Are you satisfied with the functionality offered by this application	0	0	24	40	10	74	
5	Rank your expectation and system functionality	0	0	30	40	0	70	
		Functionality					**374**	**74.8**

Table 2: Data Analysis of Functionality, Water Board Billing App

Overall Impression

#	Evaluation criteria	Marks offered					Out of 100	%
		2	4	6	8	10		
		Very poor	Poor	Average	Good	Excellent		
1	Does system provide the final result within your expected time scope with expected quality	0	0	24	48	0	72	
2	How would you categorize system based on this feature	0	0	36	32	0	68	
3	Look and feel when application use regular basis	0	0	36	32	0	68	
4	Overall satisfaction with this application	0	0	36	24	10	70	
5	Rank your expectation and system functionality	0	0	24	48	0	72	
		Overall Impression					**350**	**70**

Table 3: Data Analysis of Overall Impression, Water Board Billing App

Appendix M – Source code for Water Meter Device

1. Source Code run on Micro Controller

```
#include <SoftwareSerial.h>
#include <DS3231.h>
#include <SPI.h>
#include <SD.h>

#define WAIT 10
#define ARRAY 64

SoftwareSerial s1(6, 5);
DS3231  rtc(SDA, SCL);

char ch;
char clockData[ARRAY];
char tmp[ARRAY];
char dataBuff[1];

byte statusLed    = 8;
byte sensorInterrupt = 0;  // 0 = digital pin 2
byte sensorPin       = 2;

// litre/minute of flow.
float calibrationFactor = 4.15;
volatile byte pulseCount;
float flowRate;
unsigned int flowMilliLitres;
unsigned long totalMilliLitres;
unsigned long oldTime;

unsigned long waittime = WAIT;
bool sdReading = false;
bool deletable = false;
unsigned long currentPos;
File water;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  s1.begin(9600);
  rtc.begin();

  pinMode(statusLed, OUTPUT);
  digitalWrite(statusLed, LOW);  // We have an active-low LED
attached

  pinMode(sensorPin, INPUT);
```

```
  digitalWrite(sensorPin, HIGH);

  pulseCount        = 0;
  flowRate          = 0.0;
  flowMilliLitres   = 0;
  totalMilliLitres  = 0;
  oldTime           = 0;

  Serial.print("Initializing SD card...");

  if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    return;
  }
       Serial.println("initialization done.");
       water = SD.open("TEST.TXT", FILE_WRITE);

  if (!water) {
    Serial.println("SD File Open ERROR !!");
    return;
  }
       attachInterrupt(sensorInterrupt, pulseCounter,
FALLING);//water flow pin , interupt attach
       memset(clockData, 0x00, ARRAY);//cls variable
       memset(tmp, 0x00, ARRAY);
}

void loop() {
  // put your main code here, to run repeatedly:

  if ((millis() - oldTime) > 1000)   // Only process counters
once per second
  {
    detachInterrupt(sensorInterrupt);
    flowRate = ((1000.0 / (millis() - oldTime)) * pulseCount)
/ calibrationFactor;
    oldTime = millis();
    flowMilliLitres = (flowRate / 60) * 1000;
    totalMilliLitres += flowMilliLitres;
    unsigned int frac;
    frac = (flowRate - int(flowRate)) * 10;
    //Serial.print("  Output Liquid Quantity: ");
// Output separator
    //Serial.print(totalMilliLitres);
    //Serial.println("mL");
    // Reset the pulse counter so we can start incrementing
again
    pulseCount = 0;

    // Enable the interrupt again now that we've finished
sending output
    attachInterrupt(sensorInterrupt, pulseCounter, FALLING);

    if (waittime-- == 0x00) {
      waittime = WAIT;
```

```
      //save data to file, if file is reading then data copy
to tem
      if (!sdReading) {
        water.println(clockData);
        water.flush();
        Serial.println("Data Writtent to SD !!!");
        deletable = false;
      } else {
        strcpy(tmp, clockData);
      }
    }
  }
  memset(clockData, 0x00, ARRAY);
  //copy totml to clocdata
  sprintf(clockData, "%010i ", totalMilliLitres);
  //strcpy(clockData, "00000000 ");
  strcat(clockData, rtc.getTimeStr());
  strcat(clockData, " ");
  strcat(clockData, rtc.getDateStr());

  if (s1.available() > 0) {
    ch = s1.read();

    if (ch == 'd') {
      s1.print(clockData);
    } else if (ch == 'r') {
      sdReading = true;
      currentPos = water.position();
      water.seek(0);
      Serial.println("Read SD");
    } else if (ch == '0'){
      digitalWrite(statusLed, LOW);
    } else if(ch == '1'){
      digitalWrite(statusLed, HIGH);
    } else if(ch == 'c'){
      if(deletable){
        water.close();
        SD.remove("TEST.TXT");
        water = SD.open("TEST.TXT",FILE_WRITE);
        deletable = false;
      }
    }
  }

// Read SD card
  if (sdReading) {
    if(water.read(dataBuff, 1) <= 0 ){
      sdReading = false;
      deletable = true;
    }
    Serial.println(water.position());
    s1.write(dataBuff[0]);

    if(!sdReading){
      Serial.println("Sending 0xFF");
```

```
      s1.write(0xFF);   //notify end  of data
       water.close();
        SD.remove("TEST.TXT");
        water = SD.open("TEST.TXT",FILE_WRITE);//new file
    }
  }

//if data in temp write in to sd card

  if(strlen(tmp) > 0){
    water.println(tmp);
    memset(tmp, 0, ARRAY);
  }
}

void pulseCounter()
{
  // Increment the pulse counter
  pulseCount++;
}
```

2. Source Code run on ESP8266 Wi-Fi Module

```
/* Create a WiFi access point and provide a web server on it.
*/

#include <ESP8266WiFi.h>
#define MAX_SRV_CLIENTS 1

/* Set these to your desired credentials. */
const char *ssid = "WaterMeter";
const char *password = "Password";

WiFiServer server(23);
WiFiClient serverClients[MAX_SRV_CLIENTS];

String      rqData[MAX_SRV_CLIENTS];

/* Just a little test message.  Go to http://192.168.4.1 in a
web browser
 * connected to this access point to see it.
 */


void *process(String);

void setup() {
        delay(1000);
        Serial.begin(9600);

        WiFi.softAP(ssid, password);
        WiFi.mode(WIFI_AP);
```

```
        IPAddress myIP = WiFi.softAPIP();
        server.begin();
        server.setNoDelay(true);

}

void loop() {
  uint8_t i;
  void *ret;
  //check if there are any new clients
  if (server.hasClient()){
    for(i = 0; i < MAX_SRV_CLIENTS; i++){
      //find free/disconnected spot
      if (!serverClients[i] || !serverClients[i].connected()){
        if(serverClients[i]) serverClients[i].stop();
        serverClients[i] = server.available();
        continue;
      }
    }
    //no free/disconnected spot so reject
    WiFiClient serverClient = server.available();
    serverClient.stop();
  }
  //check clients for data
  for(i = 0; i < MAX_SRV_CLIENTS; i++){
    if (serverClients[i] && serverClients[i].connected()){
      if(serverClients[i].available()){

        //equest data throug wifi put index of rqData
        while(serverClients[i].available()){
        rqData[i] += (char)serverClients[i].read();
        }
      }
    }
  }
//processing data in rqData
   for(i = 0; i < MAX_SRV_CLIENTS; i++){
    if(rqData[i].endsWith("\r\n")){
      rqData[i].trim();
      ret = process(rqData[i]);
      //cls rqData
      rqData[i].remove(0,rqData[i].length());

      if(ret != NULL){

      }
    }
   }

   //check UART for data
  if(Serial.available()){
    size_t len = Serial.available();
    uint8_t sbuf[len];
    Serial.readBytes(sbuf, len);
```

```
    //push UART data to all connected telnet clients
    for(i = 0; i < MAX_SRV_CLIENTS; i++){
      if (serverClients[i] && serverClients[i].connected()){
        serverClients[i].write(sbuf, len);
        delay(1);
      }
    }
  }
}

void *process(String cmd){
  if(cmd.equals("enable")){
    Serial.write('1');
  }else if(cmd.equals("disable")){
    Serial.write('0');
  }else if(cmd.equals("disp")){
    Serial.write('d');
  }else if(cmd.equals("read")){
    Serial.write('r');
  }
  return NULL;
}
```

Appendix N – Source code for Customer and Meter Reader Mobile Application

```
using Android.App;
using Android.Widget;
using Android.OS;
using Android.Net.Wifi;
using Android.Content;
using Android.Views;
using Android.Util;
using System.Collections.Generic;
using Android.Net;

namespace WaterApp
{
    [Activity(Label = "Digital Water Meter", MainLauncher =
true, Icon = "@drawable/icon")]
    public class MainActivity : Activity
    {
        private static MainActivity instance;
        internal WifiManager wman;
        protected override void OnCreate(Bundle bundle)
        {
            var wifiMan = GetSystemService(WifiService) as
WifiManager;
            wman = wifiMan;
            base.OnCreate(bundle);

            // Set our view from the "main" layout resource
            // SetContentView (Resource.Layout.Main);

            if (!wifiMan.IsWifiEnabled)
            {
                wifiMan.SetWifiEnabled(true);
            }

            instance = this;
            FragmentManager
                .BeginTransaction()
                .Add(new WifiNetworkListFragment(), "wifi")
                .Commit();

            wifiMan.StartScan();
        }

        private void WifiConpleted()
        {
            var cMan =
GetSystemService(Context.ConnectivityService) as
ConnectivityManager;
            var wifiMan = cMan.ActiveNetworkInfo;

            if (wifiMan.Type == ConnectivityType.Wifi)
```

```
            {
                                        FragmentManager
                    .BeginTransaction()
                    .Add(new LoginFragment(), "login")
                    .Commit();
            }
        }

        private void LoginCompleted()
        {
            SetContentView(Resource.Layout.Control);
        }

        private class LoginFragment : DialogFragment
        {
            public override View OnCreateView(LayoutInflater
inflater, ViewGroup container, Bundle savedInstanceState)
            {
                var view =
inflater.Inflate(Resource.Layout.Login,container);

                var txtUserName =
view.FindViewById<EditText>(Resource.Id.txtUserName);
                var txtPassword =
view.FindViewById<EditText>(Resource.Id.txtPassword);
                var cmdLogin =
view.FindViewById<Button>(Resource.Id.cmdLogin);
                var cmdExit =
view.FindViewById<Button>(Resource.Id.cmdExit);

                Dialog.SetTitle("User Login");

                cmdLogin.Click += (s, e) => {
                    if (txtUserName.Text.Trim().Length == 0)
                    {
                        Toast.MakeText(Context, "Please enter
the username", ToastLength.Long).Show();
                    }
                    else if (txtPassword.Text.Trim().Length ==
0)
                    {
                        Toast.MakeText(Context, "Please enter
the password", ToastLength.Long).Show();
                    }else
                    {
                        if (txtUserName.Text.Trim() == "admin"
&& txtPassword.Text.Trim() == "1234")
                        {

MainActivity.instance.LoginCompleted();
                            Dismiss();
                        }
                        else if (txtUserName.Text.Trim() ==
"janaka" && txtPassword.Text.Trim() == "1999")
                        {
```

```
MainActivity.instance.LoginCompleted();
                            Dismiss();
                        }
                        else
                        {
                            Toast.MakeText(Context, "Invalid
credentials", ToastLength.Long).Show();
                        }
                    }
                };

                cmdExit.Click += (s, e) =>
                {
                    MainActivity.instance.Finish();
                };
                return view;
            }
        }

        private class WifiNetworkListFragment : DialogFragment
        {
            private static WifiNetworkListFragment instance;

            private ListView lv;

            public override View OnCreateView(LayoutInflater
inflater, ViewGroup container, Bundle savedInstanceState)
            {
                var layoutMain = new LinearLayout(Activity);
                var lstWifiList = new ListView(Activity);
                lv = lstWifiList;

                layoutMain.LayoutParameters = new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.MatchParent);
                layoutMain.AddView(lstWifiList, new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
(int)TypedValue.ApplyDimension(TypedValue.DensityDefault, 500,
Resources.DisplayMetrics)));

                instance = this;

                Dialog.SetTitle("Wifi List");

                lstWifiList.ItemClick += (s, e) => {
                    var sRes = e.View.Tag as ScanResult;

                    if (sRes.Capabilities.Contains("WPA"))
                    {
                        WPAPasswordEntry wpa = new
WPAPasswordEntry();
                        wpa.res = sRes;

MainActivity.instance.FragmentManager.BeginTransaction()
```

```
                        .Add(wpa,"psk")
                        .Commit();
                    }
                };
                         return layoutMain;
            }

            private void refresh(IList<ScanResult>
scanResults)
            {

                var wifiListAdapter = new WifiListAdapter();
                wifiListAdapter.ScanResult = scanResults;
                lv.Adapter = wifiListAdapter;
            }

            [BroadcastReceiver(Enabled = true)]
            [IntentFilter(new System.String[] {
WifiManager.ScanResultsAvailableAction })]
            private class WifiCalBacks : BroadcastReceiver
            {
                public override void OnReceive(Context
context, Intent intent)
                {

WifiNetworkListFragment.instance.refresh(MainActivity.instance
.wman.ScanResults);
                }
            }

            private class WifiListAdapter :
BaseAdapter<ScanResult>
            {
                public override ScanResult this[int position]
                {
                    get
                    {
                        return ScanResult[position];
                    }
                }

                public override int Count
                {
                    get
                    {
                        return ScanResult.Count;
                    }
                }

                public IList<ScanResult> ScanResult { get;
internal set; }

                public override long GetItemId(int position)
                {
                    return position;
```

```
                }

                public override View GetView(int position,
View convertView, ViewGroup parent)
                {

                    var llayout =   new
LinearLayout(MainActivity.instance);
                    var txtSSID = new
TextView(MainActivity.instance);
                    var item = ScanResult[position];
                    llayout.Orientation =
Orientation.Vertical;
                    llayout.LayoutParameters = new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.MatchParent);
                    llayout.AddView(txtSSID, new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
(int)TypedValue.ApplyDimension(TypedValue.DensityDefault, 100,
MainActivity.instance.Resources.DisplayMetrics)));

                    txtSSID.Text = item.Ssid.Trim().Length ==
0 ? "Hidden Network" : item.Ssid.Trim();
                    txtSSID.Gravity = GravityFlags.Center;
                    llayout.Tag = item;
                    return llayout;
                }
            }

            private class WPAPasswordEntry : DialogFragment
            {
                internal ScanResult res;
                private static WPAPasswordEntry instance;
                private bool wifiOk = false;
                public override View
OnCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState)
                {
                    var llayout = new
LinearLayout(MainActivity.instance);
                    var txtPSK = new
EditText(MainActivity.instance);
                    var cmdGo = new
Button(MainActivity.instance);

                    llayout.Orientation =
Orientation.Vertical;

                    llayout.LayoutParameters = new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.MatchParent);
                    llayout.AddView(txtPSK, new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.WrapContent));
```

```
                    llayout.AddView(cmdGo, new
ViewGroup.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.WrapContent));

                    cmdGo.Text = "Connect";
                    Dialog.SetTitle("Please enter the
password");
                    instance = this;
                    cmdGo.Click += (s, e) =>
                    {
                        if (txtPSK.Text.Trim().Length == 0)
                        {

Toast.MakeText(MainActivity.instance,"Please enter the PSK .",
ToastLength.Long).Show();
                        }
                        else
                        {
                            var nid =
MainActivity.instance.wman.AddNetwork(

                                new WifiConfiguration()
                                {
                                    Ssid = $"\"{res.Ssid}\"",
                                    PreSharedKey =
$"\"{txtPSK.Text.Trim()}\"",
                                }

                                );


MainActivity.instance.wman.Disconnect();

MainActivity.instance.wman.EnableNetwork(nid, true);

MainActivity.instance.wman.Reconnect();
                            this.Dismiss();

WifiNetworkListFragment.instance.Dismiss();
                            SystemClock.Sleep(2000);

MainActivity.instance.WifiConpleted();
                                wifiOk = true;
                            }
                    };
                    return llayout;
                }

            }
        }

    }
}
```

Appendix O – Source code for Water Board Billing Web Application

Logging Form

```
<?php
session_start();
$Signin=$_POST['Signin'];
if(isset($Signin)){
header("Cache-control: private");
session_register("username");
include('config_main.php');
$myusername=$_POST['username'];
$mypassword=$_POST['password'];
// To protect MySQL injection (more detail about MySQL
injection)
$myusername = stripslashes($myusername);
$mypassword = stripslashes($mypassword);
$myusername = mysql_real_escape_string($myusername);
$mypassword = mysql_real_escape_string($mypassword);
$sql="SELECT * FROM users WHERE username='$myusername' and
password='$mypassword'";
$result=mysql_query($sql);
// Mysql_num_row is counting table row
$count=mysql_num_rows($result);
//If result matched $myusername and $mypassword, table row
must be 1 row
        $a       =      $_REQUEST['username'];
if($count==1){
                        $_SESSION['username']=
$_POST['username'];
                        $abc="ss";
                        session_register("a");
                        session_register("abc");
                        header("location:mkt/sign_success.php");
                        exit;
                }
                {
                echo "Wrong Username or Password";
                exit;
                }
}
mysql_close($conn);
?>

<form method="post" action="sign_insc.php">
<table width="180" border="0" cellpadding="0" cellspacing="0">
  <!--DWLayoutTable-->
  <tr>
    <td height="15" valign="top" align="left"
class="style3">User name: </td>
```

```
  </tr>
  <tr>
    <td height="25" valign="top"> <input type="text"
name="username" size=25 maxlength=50></td>
  </tr>
  <tr>
    <td height="15" valign="top" align="left"><span
class="style3">Password:</span></td>
  </tr>
  <tr>
    <td height="22" valign="top"><input type="password"
name="password" size=25 maxlength=50> </td>
  </tr>
  <tr>
    <td height="20" valign="top"><input type="submit"
name="Signin" value="sign in">      <span class="style3"><a
href="forgot.php" target="_blank">Forgot  password ?</a>
</span></td>
  </tr>
  <tr>
    <td height="15" valign="top" bgcolor="#99FFCC"><div
align="center" class="style5">Sign up here..!!! </div></td>
  </tr>
  <tr>
    <td height="34"> </td>
  </tr>
</table>
</form>
```

Bill Calculation Form

```
<?php
//include_once "dbcon.php";
print "<h1>Water Managment Sysytem</h1><br/>";
print '<form action="index.php" method="post">';
        // Get files
        if ($handle = opendir('data')) {
                print '<select name="file">';
    while (false !== ($entry = readdir($handle))) {
      if ($entry != "." && $entry != "..") {
        print '<option
value="'.$entry.'">'.$entry.'</option>'; }
    }
                print '</select>';
    closedir($handle);
        }
        print '<br/><br/>';
        print   '<input type="submit" value="Filter">';
print '</form>';
if($_POST['file']){
        $meeterid = explode(".", $_POST['file']);
        print '<h2>Meeter readings of '.$meeterid[0].'</h2>';
        $txtfile = "data/".$_POST['file'];
        // Get First line
```

```
        $lines = file($txtfile);//file in to an array
        $first_reding = explode(" ", $lines[0]);
        print 'First Reading of month:: Units:
'.$first_reding[0]. ' Time & Date: '.$first_reding[1].'
'.$first_reding[2];
        print '<br/><br/>';
                $myfile = fopen($txtfile, "r") or die("Unable
to open file!");
        while ($line = fgets($myfile)) {
                $pieces = explode(" ", $line);
                //echo $pieces[0].'-';
                //echo $pieces[1].'<br/>';
        }
        print 'Last Reading of month:: Units: '.$pieces[0]. '
Time & Date: '.$pieces[1].' '.$pieces[2];
        print '<br/><br/>';
        $unitfirst = $first_reding[0];
        $unitlast = $pieces[0];
        print 'Difference: ';
        $diff = $unitlast-$unitfirst;
        print $diff;
        print '<br/><br/>';
        print 'This Month Charge:: ';
        if($diff>0 && $diff<=5){
                print 'Rs: ';
                print number_format(($diff*50)+5, 2);
        }
        if($diff>5 && $diff<=10){
                print 'Rs: ';
                print number_format(($diff*50)+10, 2);
        }
        if($diff>11 && $diff<=15){
                print 'Rs: ';
                print number_format(($diff*50)+15, 2);
        }
        if($diff>16 && $diff<=20){
                print 'Rs: ';
                print number_format(($diff*80)+40, 2);
        }
        if($diff>21 && $diff<=25){
                print 'Rs: ';
                print number_format(($diff*100)+58, 2);
        }
        if($diff>26 && $diff<=30){
                print 'Rs: ';
                print number_format(($diff*200)+88, 2);
        }
        if($diff>31 && $diff<=40){
                print 'Rs: ';
                print number_format(($diff*400)+105, 2);
        }
        if($diff>41 && $diff<=50){
                print 'Rs: ';
                print number_format(($diff*650)+120, 2);
        }
```

```
if($diff>51 && $diff<=75){
        print 'Rs: ';
        print number_format(($diff*1000)+130, 2);
}
if($diff>75){
        print 'Rs: ';
        print number_format(($diff*1600)+140, 2);
}
fclose($myfile);       }
```

Appendix P – Text File of Water Meter Reading

Reading	Time	Date
0000000000	13:00:00	08.03.2017
0000030456	13:15:00	08.03.2017
0000070999	13:30:00	08.03.2017
0000081162	13:45:00	08.03.2017
0000131597	14:00:00	08.03.2017
0000156790	14:15:00	08.03.2017
0000207754	14:30:00	08.03.2017
0000228475	14:45:00	08.03.2017
0000259298	15:00:00	08.03.2017
0000259298	15:15:00	08.03.2017
0000259298	15:30:00	08.03.2017
0000259298	15:45:00	08.03.2017
0000259298	16:00:00	08.03.2017
0000259298	16:15:00	08.03.2017
0000279307	16:30:00	08.03.2017
0000289407	16:45:00	08.03.2017
0000289969	17:00:00	08.03.2017
0000290552	17:15:00	08.03.2017
0000291428	17:30:00	08.03.2017
0000293753	17:45:00	08.03.2017
0000314192	18:00:00	08.03.2017
0000327013	18:15:00	08.03.2017
0000335018	18:30:00	08.03.2017
0000342041	18:45:00	08.03.2017
0000347095	19:00:00	08.03.2017
0000355261	19:15:00	08.03.2017
0000367795	19:30:00	08.03.2017
0000372841	19:45:00	08.03.2017
0000372841	20:00:00	08.03.2017
0000372841	20:15:00	08.03.2017
0000378094	20:30:00	08.03.2017
0000382409	20:45:00	08.03.2017
0000398130	21:00:00	08.03.2017
0000417339	21:15:00	08.03.2017
0000434695	21:30:00	08.03.2017
0000458037	21:45:00	08.03.2017
0000488300	22:00:00	08.03.2017
0000508513	22:15:00	08.03.2017
0000508513	22:30:00	08.03.2017
0000508513	22:45:00	08.03.2017
0000508513	23:00:00	08.03.2017
0000508513	23:15:00	08.03.2017
0000508513	23:30:00	08.03.2017
0000508513	23:45:00	08.03.2017
0000508513	00:00:00	09.03.2017
0000508513	00:15:00	09.03.2017
0000508513	00:30:00	09.03.2017
0000508513	00:45:00	09.03.2017
0000508513	01:00:00	09.03.2017
0000508513	01:15:00	09.03.2017

0000508513	01:30:00	09.03.2017
0000508513	01:45:00	09.03.2017
0000508513	02:00:00	09.03.2017
0000508513	02:15:00	09.03.2017
0000508513	02:30:00	09.03.2017
0000508513	02:45:00	09.03.2017
0000508513	03:00:00	09.03.2017
0000508513	03:15:00	09.03.2017
0000508513	03:30:00	09.03.2017
0000508513	03:45:00	09.03.2017
0000508513	04:00:00	09.03.2017
0000508513	04:15:00	09.03.2017
0000508513	04:30:00	09.03.2017
0000520834	04:45:00	09.03.2017
0000526045	05:00:00	09.03.2017
0000527047	05:15:00	09.03.2017
0000527371	05:30:00	09.03.2017
0000528292	05:45:00	09.03.2017
0000542054	06:00:00	09.03.2017
0000568486	06:15:00	09.03.2017
0000598760	06:30:00	09.03.2017
0000611047	06:45:00	09.03.2017
0000620374	07:00:00	09.03.2017
0000629008	07:15:00	09.03.2017
0000629008	07:30:00	09.03.2017
0000629008	07:45:00	09.03.2017
0000629008	08:00:00	09.03.2017
0000631772	08:15:00	09.03.2017
0000652759	08:30:00	09.03.2017
0000668491	08:45:00	09.03.2017
0000685815	09:00:00	09.03.2017
0000701457	09:15:00	09.03.2017
0000701457	09:30:00	09.03.2017
0000701457	09:45:00	09.03.2017
0000701457	10:00:00	09.03.2017
0000701457	10:15:00	09.03.2017
0000701457	10:30:00	09.03.2017
0000715222	10:45:00	09.03.2017
0000715222	11:00:00	09.03.2017
0000715222	11:15:00	09.03.2017
0000715222	11:30:00	09.03.2017
0000715222	11:45:00	09.03.2017
0000715222	12:00:00	09.03.2017
0000729452	12:15:00	09.03.2017
0000745176	12:30:00	09.03.2017
0000745176	12:45:00	09.03.2017