Student declaration

I declare that this dissertation entitled, "**IoT Flood Monitoring System in Rwanda**" contains the results of my work and has not been submitted for any other university of Rwanda or any other Institution of Higher Learning. The content of this work is the product of the project carried out since the approval of research activities. All the guidelines, ethics and procedures have been followed during its preparation.

Florentine   MUKAMANA (Reg.No: 219013575)

Signed: …………………..

Date: ……/….. /2021

Bonafide certificate

This is to certify that the project entitled" **IoT FLOOD MONITORING SYSTEM IN RWANDA**" is a record of original work done by **MUKAMANA Florentine** with **Reg number 219013575** in partial fulfillment of the requirement for the award of masters of sciences in Internet of Things in College of Science and Technology, University of Rwanda, Academic year 2018/2019

This work has been submitted under the guidance of **Dr. NYESHEJA MUHIRE Enan** and **Dr. SIBOMANA Louis.**

Main Supervisor:
Dr. NYESHEJA MUHIRE Enan

Signature:

Co-Supervisor:
Dr. SIBOMANA Louis.

Signature:

The Head of Masters and Trainings

Names: **Dr MUKANYIRIGIRA Didacienne**

Signature:

Acknowledgments

Many thanks to my supervisors, Dr. NYESHEJA MUHIRE Enan, Faculty of Computer and Information Sciences/ University of Lay Adventists of Kigali (UNILAK) and Dr. SIBOMANA Louis, School of Engineering/ College of Science and Technology. I thank them for their full support and valuable advice during my master’s dissertation. They made all this possible and showed me the path for my first steps. I acknowledge their time, attention, patience, corrections and direction as well as their wise advice and discussions. Without their help, it would have been impossible to complete the master's journey. I also thank my spouse and my parents for their selfless care and warm encouragement. I would also thank my child who inspired me and patient in my busy time on this program. Finally, I thank my colleagues and friends in Internet of Things for the happy time they have shared with me.

Abstract

Flooding is among the major catastrophes which occur in all places of the world and results in a massive quantity of damage to the environment. The problem occurs during a flash flood where the citizens may not have time to move their important things from the flood area to the safe area. As such, there is a need to develop technology for monitoring and alerting the population about flooding. This project presents a flood monitoring system to collect, receive, and analyze data from two different locations situated in Kigali, Rwanda namely the Mpazi brook and Rwampara brook that contributes to the Nyabugogo flood. The data captured by sensors are published using MQTT (Message Querry Telemetry Transport) Protocol to ThingSpeak. Each location has a flood detector system composed of an ultrasonic sensor for measuring the changes of the water level of the brook, a rain drop sensor to measure rain intensity, and NodeMCU to collect data from sensors. The water level is used as a flood indicator and it can be known remotely. The data collected by these sensors are transferred to ThingSpeak web-based application and send an alerting message to public authorities based on a determined threshold value. The developed flood monitoring system can help public authorities to be informed about the rise of water level and to know whether there is a manifestation of a flood.

Key words

Flood

NodeMCU

Internet of Things (IoT)

MQTT

ThingSpeak

List of Symbols and Acronyms

ACEIoT: African Center of Excellence in Internet of Things.

API: Application protocol Interface

Ao: Analog pin

Do: Digital pin

Ec: Echo

GND: Ground

GPIO: General Purpose Input/ Output

ICT: Information communication and technology

IEEE: Institute of Electrical and Electronics Engineering

IFTTT: If This Then That

IT: Information Technology

I^2C: Inter-Integrated Circuit

LED: Light Emitting Diode

MINEMA: Ministry of Emergency Management

MININFRA: Minister of infrastructure

M2M: Machine to Machine

MQTT: Message Query Telemetry Transport

PWM: Pulse Width Modulation

REMA: Rwanda Environment Management Agency

RMA: Rwanda Meteorological Agency

Rx: Receiver

SPI: Serial Peripheral Interface

Tx: Transmitter

UART: Universal Asynchronous Receiver/Transmitter

Table of Contents

List of tables

List of Figures

Chapter 1

1.1 Introduction

Globally, flooding is one of the problems caused by water resources (river and rainwater) and is associated with pollution, erosion, and unsuitable land use [1], [2]. Technology is continuing to make our life better. IoT can help the development in the world in terms of Information and Communication Technology (ICT) as an important tool for innovation of the economy[3]. Consequently, the damages from flooding are visible. These losses can be reduced if there is a system of monitoring floods using IoT.

IoT is a system enabled by technology where the sensors, actuators, embedded software and hardware are interconnected using network connectivity [3],[4]. IoT is used to make smart cities clean and safe[5], [6]. In this research project, IoT was used in the flood monitoring system.

This system contributes an alert notification of the rise of water to the authorities to inform the citizens around the flooded area to leave before it rises to a dangerous level. This proposed system has been implemented in two different locations.

The first location will be focused on the Mpazi drainage channel which collects rainwater comes from Muhima, Kimisagara, Gitega, and Nyabugogo areas which contribute to Nyabugogo floods. Mpazi channel drains an urban area of around $8km^2$ to Nyabugogo[7]. The second location placed in Rwampara where originates Rwampara brook which takes end in Nyabugogo which is the flooding area. Mpazi originates in Nyamirambo and takes end in Nyabugogo River whereas Rwampara originates in Rwampara in Nyarugenge district.

The main idea of this research is to extend previous work and develop new ideas by monitoring floods and notifying public authorities about the rise of water using ThingSpeak IoT platform. The flood cannot be prevented but the consequences can be reduced by notifying people around the affected area to move to the safe place.

This works aims to develop a flood monitoring system to collect the parameters contributing to the flooding in a specific area by measuring the variations of those parameters and to know the level of the flood, and thus measures can be taken on time.

Specifically, this project has the following main contributions:

- Two locations (Rwampara brook and Mpazi brook) have been taken for effective prediction of incoming flood.
- Data are connected and transferred using NodeMCU esp8266 which has an Ethernet shield inbuilt in it.
- Public authority gets notified about the information of the upcoming floods so that all citizens could be informed about flood through public authorities.
- Results show the channel stats of the live data of every entry in function of time.

This project develops the status of flooding in Kigali City, Rwanda using IoT. The flood monitoring system monitors and detects the input parameters (ultrasonic sensors and rain drop sensors), and sends a warning notification (alert message) of flooding to the public authority using ThingSpeak application and IFTTT (If This Then That). A prototype of a flood monitoring system is developed and tested.

The proposed system facilitates the authorities to know real-time information related to flooding for the efficiency of environment protection. The output assists as an important tool to help decision-makers to combat the effect of flooding in Rwanda as well as other hilly regions around the world.

1.2 Background and motivation

Rwanda is among the countries that are affected by flood disasters because of its topography[8]. Nyabugogo area is the area located in Rwanda/Kigali City where most of the citizens located. It is one of the major flood-prone areas in Rwanda where the water is capable to rise at the highest level almost instantly in a short period. There are many cases of floods that involved Nyabugogo area in Rwanda as reported in [9], [10], [11]. The problem occurs during a flash flood where the citizens have or do not have time to move their important things from the flood area to the safe area. This is a big problem when there is no time taken to move before the flash flood strikes.

So, the public authority came to realize the situation so that the citizens might get a warning before the flash flood. The authorities realize that the people move their important things too late thus they lose a huge amount of properties. Therefore, a flood monitoring system using an ultrasonic sensor, rain drop sensor and IoT system has been used to design and develop a flood monitoring system to support sending email notifications to the public authority to warn and alert them when the sensor senses a fast increasing amount of water level and rain intensity and about the flood almost immediately.

A flood detector system composed of rain drop sensor, ultrasonic sensor and NodeMCU will be placed at two different locations namely Rwampara brook and Mpazi brook which contribute to the Nyabugogo flood. The water level is measured by an ultrasonic sensor which is divided into three levels or states according to the level of the flood of each brook. These states are normal state, medium state and dangerous state. When the water is in a normal state, the green Light Emitting Diode (LED) will be lighted ON and the buzzer that acts as an alarm will be triggered in a low state. When the water is in a medium state, the yellow LED will be lighted ON and the buzzer that acts as an alarm will be triggered in the low state. When the water is in a dangerous state, the red LED will be lighted ON and triggers the buzzer that acts as an alarm to alert the public authority in its ON state. All the readings of the ultrasonic sensor and rain drop sensor are shown as graphics representation in a ThingSpeak application.

This flood monitoring system is composed of hardware components and software programming. The ultrasonic sensor and rain drop sensor act as the input of the system, controller as the output of the system. The nodeMCU ESP8266 acts as the control of inputs and output of the system and it is also used as an interface with the output and connection to ThingSpeak application respectively. The LED and a buzzer act as the output system. About software programming, Arduino software IDE is used for hardware coding. Thus, a flood monitoring system to alert the public authorities should be developed.

The flood detected by these two different locations can be monitored at the same time on the ThingSpeak application. The online users can monitor these two locations and receive the message notification through an IFTTT application. This system has been implemented using Internet of Things (IoT) technology which has the best capability of any information over wireless.

This project aims to design and develop a flood monitoring system by detecting the flood on Mpazi brook and on Rwampara brook that could be monitored on the ThingSpeak application. The monitored system alert and warn the public authorities apart from IFTTT platform via the wireless connection.

The regular natural disaster that strikes Rwanda country especially Nyabugogo area is the motivation of this project. Nyabugogo area is one of the flood areas which causes a lot of losses to the citizens. Hence, this system tends to help the community as well as the country particularly for the people located (who lives) in the area affected by flash flood [12]. This was somehow effective as the considered parameters are taken from two sources that can be contributed by water from two sources and two parameters water level and rain intensity were taken into consideration.

1.3 Problem statement

In Rwanda, we are experiencing the problem of floods due to heavy rain. It occurs almost every year during each period of rain and causes loss of lives, loss of damages and destruction of property as well as destruction for infrastructure[13], [14].

In January 2020, Heavy rains, floods, and landslides have killed more than 60 people and damaged more than 1 000 homes in Rwanda [15]. In February 2020, at least 13 people have died, 2 people have injured and 15 houses have destroyed[16]. On 03rd May 2020, 8 people had died, 5 were injured, more than 100 houses had collapsed and roads were closed[17].In the same month 8th May 2020, Torrential Rain and Floods Leave Over 60 Dead[18].

Flood causes a hitch in transport for pedestrians and motorists and bikes[19]. Rwanda Meteorological Agency (RMA) had earlier predicted heavy rains and strong winds in Rwanda's West, South and East[19]. The difficulty arises from not knowing where the upcoming flood may be and it is a big problem for Rwanda Environment Management Agency (REMA), Ministry of Emergency Management (MINEMA), Minister of infrastructure (MININFRA) and all the government and also for each people located around the flood area.

A study has shown that during each rainy season occur flood damages at Nyabugogo River[20]**.** The urban area is where life is dependent on several services and more occupations like water, transport, energy, education, infrastructure and more occupation. Floods in urban areas can interrupt all of these services and a big impact on the population.

There are huge problems caused by floods in Rwanda, among them there is loss of lives (human and animals), material losses, damages to crops and damages to public facilities (infrastructure) [21].

a b

Figure 1**.** a. People struggled to navigate through a flooded road[21], b. a child who was stuck in the middle of a flooded in Nyabugogo River[16]

Flood causes loss of lives, damages to crops, properties, public facilities, and others. It is not easy to avoid them. This is the reason why there should be a way of monitoring and notifying the community when these disasters happen so that they escape or plan for them. Flood also has a big problem with transport activities as is shown in figure 1a. The difficulty arises from not knowing where and when the floods may occur.

There have been periods of floods that have caused a big impact on economic growth in Rwanda. Consequences resulted in infrastructure damage, fatalities and injuries, landslides, loss and damage to crops, soil erosion and environmental degradation[20].

IoT flood monitoring system can play an important role in reducing flood problems. It delivers information about the rising of river water. IoT flood monitoring system sends warning message notifications to the community using IFTTT web services.

1.4 Study Objectives

1.4.1 General objectives

The general objective of this research project is to monitor floods using IoT technology by notifying the public authority using ThingSpeak application.

1.4.2 Specific objectives

The specific objectives of this research project are listed below.

- To develop a prototype of a flood monitoring system by using the ultrasonic sensor to measure the water level of the river and rain drop sensor to measure rain intensity in the river location
- To test the developed prototype
- To send the value of water level from NodeMCU to the ThinSpeak application.
- To show the current value of the water level by setting LEDs to differentiate which level such as normal, medium or dangerous.
- To generate a computer program for nodeMCU for getting the real-time signal using Arduino IDE software.

1.5 Hypotheses

IoT flood monitoring system can be used to inform the conditions of floods by notifying the public authority using ultrasonic sensors, rain drop sensors, nodeMCU (ESP8266) and Thingspeak over MQTT protocol.

1.6 Study scope

This research specifically focused on IoT flood monitoring system to collect, receive, and analyze data from two (2) different locations situated in Kigali, Rwanda namely Mpazi brook and Rwampara brook that contributes to Nyabugogo flood. In this research project, the data captured by sensors are published using MQTT (Message Querry Telemetry Transport) Protocol to ThingSpeak. Each location has an ultrasonic sensor for measuring the changes of the water level of the brook, rain drop sensor to measure rain intensity, and NodeMCU to collect data from sensors. The water level of each location can be known remotely by transferring the data collected by sensors to ThingSpeak web-based application. IFTTT sends an alerting message to public authorities based on a determined threshold value in the ThingSpeak apps. The flood monitoring system can help public authorities to be informed about the rise of water level and to know whether there is a manifestation of a flood or not.

1.7 Significance of the Study

This project aims to monitor flood monitoring system in two brooks which are Mpazi brook and Rwampara brook that contributes to Nyabugogo flood.

This flood monitoring system is composed of hardware parts and software parts. Hardware parts are composed of two (2) breadboards, two (2) ultrasonic sensors, two (2) rain drop sensors, two (2) NodeMCU, jumper wires, two buzzers, and 10 LEDs. A breadboard is used to connect the equipment for every location. Jumper cables are used to connect the ultrasonic sensors, rain drop sensors, and nodeMCU. LED and buzzer are used to give a warning when the flood level rises. Arduino IDE was used as software to code the entire program.

Ultrasonic sensors detect the water level by measuring the distance between it and the surface of the water. This water level detection is done without any physical contact between the ultrasonic sensor and the water surface.

Ultrasonic sensors use sound reflection's principle to measure the distance. The elapsed time that is required to transmit and receive the reflected ultrasonic wave is multiplied by the rapid propagation of sound in the air to obtain the distance value.

Depending on the dimensions of the prior years for a similar river, the LEDs are set to show the state of the water level. The LED with a green color shows that the water level is still at the normal level.

The LED with a yellow color shows the water level at medium state (level). The LED with a red color shows that the water level has reached the dangerous state which is a flood level. The buzzer is activated when the water level is changed rapidly from a medium state to a dangerous state.

1.8 Organization of the study

This research is composed of six chapters. The first chapter is an introduction composed of background and motivation, problem statement, study objectives, hypotheses, the study scope, significance of the study, organization of the study as well as a conclusion. The second chapter focuses on existing literature by showing the gaps and how we are going to fill the gaps. The third chapter outlines the methods employed in the implementation of our projects. It leads to a clear implementation plan. The fourth chapter is the analysis and design. The fifth chapter is the analysis and discussions of the results. It explains our findings and the graphs of our results. The last chapter presents the conclusion and recommendation.

1.9 Conclusion

This research aims to investigate the status of flooding in Kigali City, Rwanda using IoT. It monitors and detects the input parameters (ultrasonic sensors and rain drop sensors), and sends a warning notification (alert message) of flooding to the public authority using the ThingSpeak application. A prototype of a flood monitoring system is developed and tested. The proposed system is implemented in two different locations namely Mpazi brook and Rwampara brook which contribute to the Nyabugogo flood. This study intends to fill a vacuum by granting to the authorities, information related to flooding for the efficiency of environment protection. The output assists as an important tool to help decision-makers to combat the effect of flooding in Rwanda as well as other hilly regions around the world. Information that relates to the flood monitoring system in Rwanda was covered in this chapter. The problems related to the flood monitoring system were highlighted. The aims of the research, research objectives, and research scope as well as the significance of the study were discussed. The motivation for developing this research had explained.

Chapter 2 Literature review

This chapter explains the existing literature. It shows the gaps and how it is going to be solved.

2.1 Related works

There is an interest in the research community on the IoT flood monitoring system. J. W. Simatupang et al.[22], have used Arduino and ultrasonic sensor components for detecting floods by sending an alert to the organizations through GSM (Global System for Mobile Communications) and GPRS (General Packet Radio Services) technologies. These two technologies can affect the Arduino functionality because of its low responsiveness to some instructions such as disconnect wiring in Rx (receiver) and Tx (transmitter) each time you burn the program to Arduino [23] and to reduce the functionality of the module [24]. Ethernet shield can be added to the system to be more stable and secure and also by enhancing the reliability of the signal. Hence, this approach does not have live monitoring of parameter variations and its costs are high due to the GSM/GPRS additional charges as well as Ethernet shield price also system responsiveness is low.

Further, the authors in [25] have developed a system using Arduino, ESP8266 as a Wi-Fi (Wireless Fidelity) network and a database to store information from sensors. The Android application has been created to be used by the users who are registered to get information about the current status of data from sensors. The system can be enhanced by using esp8266 as storage and at the same time as Wi-Fi. This system is not effective. Since the user who needs to access the data must be registered and must have the applicable device which can enable him/ her to access all the information. The system is not cost-effective and it considers only one river source alone as the flood contributor.

Moreover, a system using an ultrasonic sensor, water flow sensor, NodeMCU ESP8266, and Blynk application have been implemented [26], [27] to contribute to the reduction of disaster consequences. The rain drop sensor can be added to the system to know if the quantity of rain is raising or not. In [28], a system that predicts several parameters such as water level, weather conditions, and water flow has been proposed by sending the update values of sensors to the mobile application. This system doesn't have an alerting mode in case of an emergency.

Furthermore, the authors in [29] proposed a system that provides the risk alert to the management of an emergency call of a government department in case of an emergency. The temperature sensor, humidity sensor, and level sensor were used as the inputs of the system and the microcontroller ESP8266 was used to collect the data captured by these sensors. The system shows the collected data in case of emergency and it sends the alert message to the citizens. It can be enhanced by helping users to visualize real-time data gathered by sensors.

In addition, A. Diriyana et al. [30] have used Arduino Uno to display data, NodeMCU to manage data, the ultrasonic sensor to detect distance, ThingSpeak platform for displaying real-time data, and the Telegram platform for sending notification. This system works well to generate output information level data and flood early warning information. NodeMCU can be used to manage data and at the same time to store data. This system is not cost-effective and it can be enhanced by reducing its cost using nodeMCU to display and manage data. The rain drop sensor can be added to the system to know the rain intensity in the surrounding area.

The main idea behind this flood monitoring system is to collect the parameters contributing to flooding in a specific area by measuring the variations of those parameters. You can know the level of the flood and measures can be taken at the time.

It should be noted that the work in [22-30] has considered only one location or one river as the flood contributing river.

2.2 Ultrasonic sensor

2.2.1 Background of ultrasonic sensor measurement

An ultrasonic sensor senses the distance from its transceiver to an object and is used to measure the distance between the surfaces of water to the ultrasonic sensor. Ultrasonic sensors transmit sound waves toward a target and will determine its distance by measuring the time it took for the reflected waves to return to the receiver. This sensor is an electronic device that will measure the distance of a target by transmitting ultrasonic sound waves, and then will convert the reflected sound into an electrical signal[31].

The ultrasonic distance sensor uses high frequency sound to determine the distance from a reflected object. Similar to how bats detect obstacles by transmitting high-pitched sounds and listening to echoes. These ultrasonic distance sensors emit a series of supersonic pulses and wait for the echo pulses to be detected. Since the speed of sound is constant in air (340.29m /s), the time elapsed between the transmitted signal and the received signal can be measured and thus the distance to the object can be determined. Ultrasonic distance measurement is based on the speed of sound property. The system transmits several sound waves which propagate through the air. These sound waves are reflected on the objects they impact and return in the form of an echo to the place where they came from. The system detects these reflected sound waves (i.e.echoes). The time between sound wave transmission and echo detection is measured.

The travel time of sound is multiplied by the speed of sound to calculate the total distance traveled by the sound waves. This distance is divided by two to calculate the distance from the object causing the echo. These calculations are shown in Equation 1

$$\mathbf{S} = \frac{\mathbf{v.t}}{\mathbf{2}} \qquad \textbf{Equation (1)}$$

, where S is the distance between the transducer and the detected object, V is the speed of sound, and t is the measured time between sound wave transmission and detection of the echo**.**

Some advantages of the ultrasonic distance sensor are that it is less affected by target materials or color. Even though it does not have a narrow field of view like the laser rangefinder, it is still able to detect objects within a meter. These ultrasonic sensors are designed to withstand external disturbances such as vibration, infrared radiation, ambient noise and EMI radiation. The cost of the ultrasonic range finder depends on the uses of the frequency transducer. High frequency (~ 255 KHz) ultrasonic range finder costs $100-200, but with moderate high frequency (40 KHz) it is cheaper.

Ultrasonic waves are sounds with frequencies above 20 kHz that are not heard by humans. The theory is based on measuring the reflection time of pulses. The ultrasound transducer transmits a wave pulse and receives a reflection signal called echoes as transmitted wave pulse detects an object, the reflected wave, echo wave, is bounced back to the transducer.

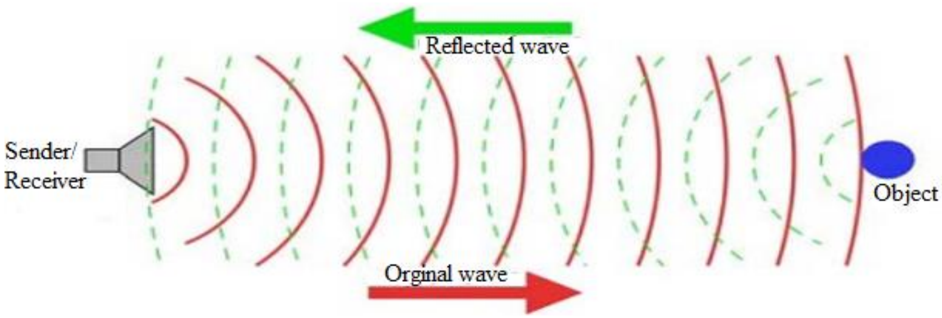

Figure 2: Ultrasonic sensing principle

The ultrasonic sensor is mostly used in distance measurement applications such as level control and also capable of detecting most objects that are metal or non-metal, clear or opaque, liquid. The ultrasonic sensor emits a 40KHz ultrasound through the air and it will bounce back to the module if there an object in front of it[32]. To generate the ultrasound, a trigger pulse burst of 10 μs is given to the trigger pin of the module from the NodeMCU. The module generates 40KHZ ultrasonic burst of 8cycles which is transmitted by transmitting transducer[33]. If any object is detected, the burst returns which is received by receive transducer, sensed by echo line & is used for distance calculation. The Echo pin will output the time in microseconds, and the sound wave traveled is shown in figure 3.

A trigger pulse burst of 10microsecond is given to the trigger pin of the module from the NodeMCU. The module generates a 40KHZ ultrasonic burst of 8cycles which is transmitted by a transmitting transducer. If any object is detected, the burst returns which is received by receive transducer, sensed by echo line & is used for distance calculation.

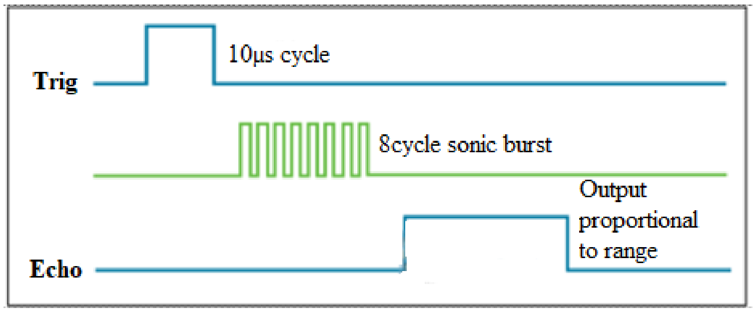

Figure 3: The echo and trig pin

2.2.2 General specifications of ultrasonic sensor

The ultrasonic sensor can measure the distance between 2centimeters and 4meters.

Power input: +5VDC

Output format: the ultrasonic sensor gives the analog output which varies from 2cm to 400cm

2.2.3 Ultrasonic sensor pinout

Ultrasonic sensor has four pins VCC (5v), ground (GND), Trigger (Tr), and Echo (Ec)[34].

Figure 4: Ultrasonic sensor pinout [35].

2.2.4 Wring connections of the ultrasonic sensor to nodeMCU

The table shows the wring connection of the ultrasonic sensor to nodeMCU. The ultrasonic sensor is powered by 5V where nodeMCU is powered by 3.3V. Our NodeMCu Lolin has a pin called VU which is a pin considered a 5V pin. That's why the VCC of the ultrasonic sensor is connected to the VU pin of nodeMCU[36]**.**

Table 1: Wring connections of the ultrasonic sensor to nodeMCU

Ultrasonic sensor	**NodeMCU**
Vcc pin	VU pin
Trig pin	Any digital output pin
Echo pin	Any digital output pin
GND pin	GND pin

2.2.5 Selection of Ultrasonic sensors

There are numerous types of ultrasonic range sensors available with key differences in frequency and power consumption. Ultrasonic sensors with high frequency will have a sharper beam width and can detect obstacles in a longer range. Also, some of the new sensors have similar range detection as previous models but with less power consumption. In this project, an ultrasonic sensor must be able to detect obstacles or objects from 2cm to 400cm. Since the whole system power supply will be taken from the battery supply, less current consumption is crucial and must be able to operate at low voltage. HC-SR04 meets the criteria of this project to detect the obstacles in a short period after the long research was done between the HC-SR04 and others Ultrasonic sensors. In this system, the ultrasonic sensor is used as a flood level identification sensor.

2.3 Rain drop sensor

2.3.1 Background of rain drop sensor

A rain drop sensor is an easy tool for rain detection. It can be used as a switch when a rain drop falls through the raining board and also for measuring rain intensity. This rain drop sensor helps to know which quantity of rain can have an influence or impact on the rise of water level. It is composed of a plaque and an electronic circuit. The plaque is exposed to the rain. An electronic circuit board is responsible for processing the signal that comes from the plaque and exposes it as a digital signal and analog signal. The digital output pin, which shows 1 value when rain is detected, and 0 value when rain is not detected[37]. The Analog Pin (A0) varies from 0 to 1023.

2.3.2 General specifications of rain drop sensor

Power input: +5VDC

Output format: the rain drop sensor gives the digital switching output (0 and 1) and analog output A0.

2.3.2 Rain drop sensor pinout

Rain drop sensor is composed of four pins which are VCC (input supply for +5VDC), GND (Ground), D0 (Digital output 0 and 1 which indicates rain drop detected and no rain drop detected respectively) and A0 (Analog Pin) which varies from 0 to 1023.

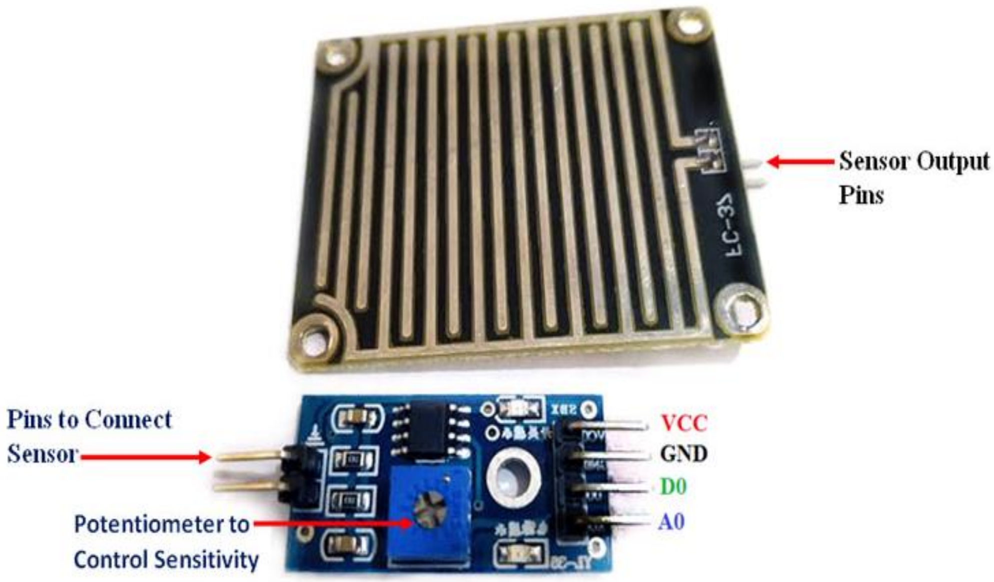

Figure 5: Rain drop sensor pinout[38].

2.3.3 Wring connections of rain drop sensor to nodeMCU

Table 2: Wring connections of rain drop sensor to nodeMCU

NodeMCU	**Rain drop sensor**
VU	**VCC**
GND	**GND**
D0	**D0**
A0	**A0**

2.4 NodeMCU

2.4.1 Introduction to NodeMCU

NodeMCU is like Arduino IDE which has the entire pin-making as NodeMCU boards. Arduino is an open source-based board and is easily applied in various embedded system applications such as industrial applications, disaster mitigation, and environmental management [39]. One of the uses of Arduino and ultrasonic sensors is the measurement of river height and river current speed [40]. Behind the board, we can find the real GPIO PIN which makes it very easy to use with any compiler for programming. NodeMCU can support only one ADC which means it has only one analog pin. The ADC pin is having a 10-bit resolution, which can read 0 to 1024, you can get a value somewhere within this range. The ADC pin reads up to 1V only. Any reading over that, say 1.1V to Vcc of 3.3V is maxed out to 1024. The NodeMCU has an inbuilt voltage divider that has connected to A0 pin, which means any pin connected to this will be corrected automatically we don't need to provide any voltage divider or any other resistor.

Microcontrollers and other Arduino Modules have always been a great choice to incorporate automation into the relevant project. But they have a little disadvantage of not having an inbuilt Wi-Fi capability. We need to add an Ethernet shield or Wi-Fi protocol into these modules to make them well-suited with the internet network. NodeMCU is based on ESP8266 Wi-Fi SoC. Which is a very important Wi-Fi API that allows it to connect with other devices. In addition, it allows access to information from the internet and allows a user to control a device from anywhere or from any part of the world[41]. NodeMCU is a development kit as well as an open-source used in prototyping IoT products and it can always be programmed using Arduino IDE.
It has multiple GPIO pins on the board which allow us to connect the board with other peripherals and are capable of generating PWM, I2C, SPI, and UART serial communications.

The interface of nodeMCU is mainly divided into two parts including both Firmware which runs on the ESP8266 Wi-Fi SoC and Hardware which is based on the ESP-12 module. An open-source gives the ability to edit, modify and rebuilt the existing system and also keep changing the entire interface until you succeed in optimizing the module as per the needed requirements. USB to UART converter is added on the module that helps in converting USB data to UART data which mainly understands the language of serial communication. Instead of the regular USB port, a MicroUSB port is included in the module that connects it with the computer for dual purposes: programming and powering up the board. The board incorporates a status LED that blinks and turns off immediately, giving you the current status of the module if it is running properly when connected with the computer. The ability of the module to establish a flawless Wi-Fi connection between two channels makes it an ideal choice for incorporating it with other embedded devices like Raspberry Pi. In this research, we used nodeMCU Lolin V3 which is the latest version and is based on the ESP8266 Wi-Fi module.

2.4.2 NodeMCU V3 Pinout

NodeMCU V3 comes with several GPIO Pins. Following figure 6 shows the Pinout of the board[41].

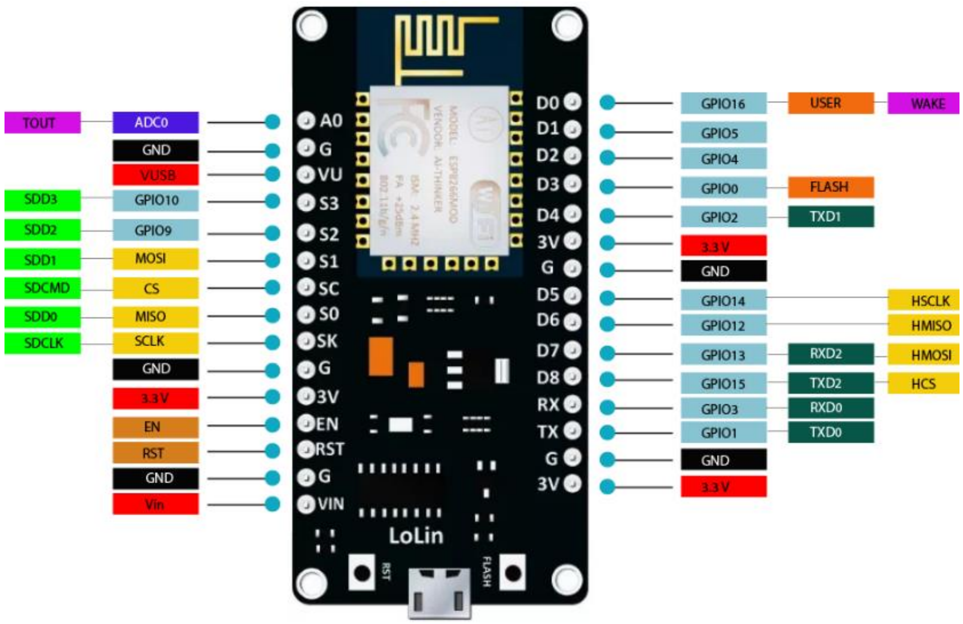

Figure 6: NodeMCU LoLin V3 pinout

There is a big difference between VIN and VU where the former is the regulated voltage that may stand somewhere between 7to12 V while later is the power voltage for USB that must be kept around 5 V.

2.4.3 NodeMCU features

NodeMCU has many features which make it the best board to use[42]. There are listed below.

i) The nodeMCU hardware is like Arduino and it is open-source.

ii) It has Status LED, MicroUSB port, Reset/Flash buttons, and ESP8266 with inbuilt Wi-Fi which is the lowest cost Wi-Fi,

iii) It uses USB to UART converter and has GPIO pins.

iv) It has an advanced API for hardware IO, which can dramatically reduce the redundant work for configuring and manipulating hardware.

v) It is easy to tool for prototyping IoT development kit.

vi) It serves as the IoT best platform for application development at the lowest cost.

vii) It has reset and flash buttons

2.4.4 How nodeMCU works

There must be a micro USB cable that supports a micro USB port that is used to connect the board to the computer. After connecting the board with a computer, the nodeMCU LED will flash.

For programming this module, we use Arduino IDE software. Figure 7 shows the pin configuration to use in Arduino IDE when connecting this board to other components like sensors, LEDs, and so on.

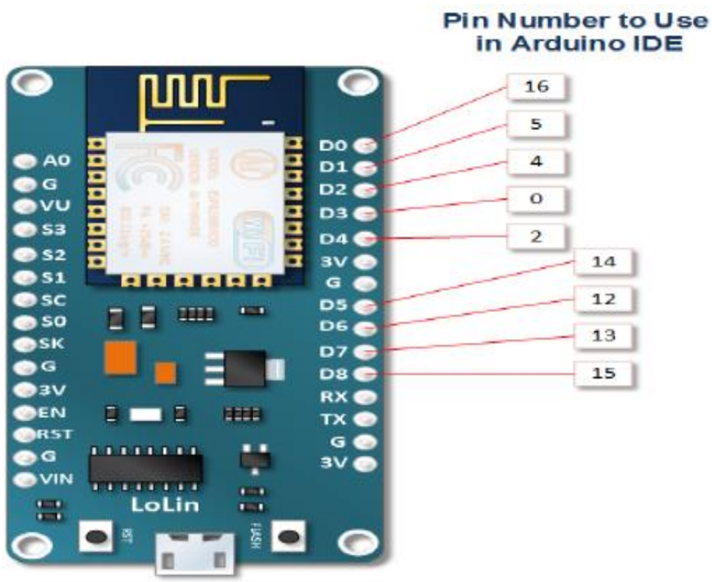

Figure 7: NodeMCU LoLin pin configuratipon

2.4.5 Powering NodeMCU LoLin V3

The nodeMCU LoLin can be powered using three ways. As figure 7 shows, the nodeMCU has five ground (GND) pins and three 3V3 pins.

Powering using USB Power. This powering method shows a perfect choice for uploading programs to the computer.

Powering using 3.3V. This is another great option to power up the module. If you have your off-board regulator, you can generate an instant power source for your development kit.

Powering using VIN. This is a voltage regulator that comes with the ability to support up to 800 mA. It can handle somewhere between 7to12 V. You cannot power the devices operating at 3.3 V, as this regulator unable to generate as low as 3.3V.

2.5 Light Emitting Diode (LED)

A LED is a semiconductor device that emits light when an electrical voltage is applied in the forward direction of the device. When LED anode lead has a more positive voltage, its cathode lead by at least the LED forward voltage drop thus current flows. Electrons can recombine with holes within the device, releasing energy in the form of photons. This effect is called electroluminescence, and the color of the light (corresponding to the energy of the photon) is determined by the energy bandgap of the semiconductor[43].
In this project, the LEDs will be used as indicators at the prototype; thus, we have used green LEDs, blue LEDs, yellow LEDs and red LEDs. Some of them indicate the water level and others indicate the rain intensity.

Figure 8: Light Emitting Diodes

2.6 Buzzer

A buzzer is an audio signaling device that may be mechanical or electronic. It can be used as an alarm, timer, or confirmation of user input[44]. The sound output from the buzzer may be intermittent. As the output is typically at least 75dB, it will provide sufficient sound aid for the user. Thus, we have used a buzzer in our system to give an alarm if the water level changed rapidly and considerably dangerous.

Figure 9: Buzzer

2.7 ThingSpeak and Internet of Things

2.7.1 Introduction to IoT

Internet of things is the network of physical objects or "Things" embedded with electronics, software, sensors, and network connectivity, which enables these objects to connect and exchange data [45]. In IoT information is generated and consumed by machines (M2M) and devices are dedicated to specific tasks [46]. We need IoT to monitor many places at the same time, to do precision and accuracy, to reduce the cost of manpower and to make Data for further analysis and understanding [47].

The term “Internet of Things” (IoT) has been invented by Kevin Ashton in 1999 [48]. It has been in use for several years and continues to be of interest, specifically when it comes to technological progress. Essentially, M2M interaction enables networked devices to exchange data and perform actions without the input or assistance of humans, for instance in remote monitoring [49].

To connect an object to the IoT, several things are needed such as hardware and software[50]. First of all, if one wishes to go beyond simply connecting data from a computer, sensors or actuators are necessary. The type of device that is connected is only limited by the imagination of its creator.

In this case, an object will connect to the cloud through an Internet connection to upload or receive data[51]. Objects to be connected are typically augmented with either sensors or actuators. A sensor is something that tells us about our environment. Actuators are something that you want to control[52].

2.7.2 ThingSpeak IoT

In this project, to connect an object to the IoT, we focus on the ThingSpeak API. The interface provides simple communication capabilities to objects within the IoT environment, as well as interesting additional applications (such as ThingTweet,). Moreover, ThingSpeak allows you to build applications around data collected by sensors. It offers real-time data collection, data processing, and also simple visualizations for its users. Data is stored in channels, which provide the user with a list of features [53]. Each channel allows you to store up to 8 fields of data, using up to 255 alphanumeric characters each[54]. There are also 4 dedicated fields for positional data, consisting of Description, Latitude, Longitude, and Elevation. All incoming data is time and date stamped and receives a sequential ID. Once a channel has been created, data can be published by accessing the ThingSpeak API with a 'write key', a randomly created unique alphanumeric string used for authentication.

Consequently, a 'read key' is used to access channel data in case it is set to keep its data private (the default setting). Channels can also be made public in which case a read key is required. Essentially, 'things' are objects that are given sensors to collect data. Data is sent and received via simple "Hypertext Transfer Protocol" (HTTP) POSTs, much like going to a web page and filling out a form. This communication happens through plaintext, JSON, or XML.

The data is then uploaded to the cloud and from there can be used for a variety of purposes. In turn, data (such as commands or choosing certain options) can be gathered and communicated to the cloud, which in turn sends these messages to the object

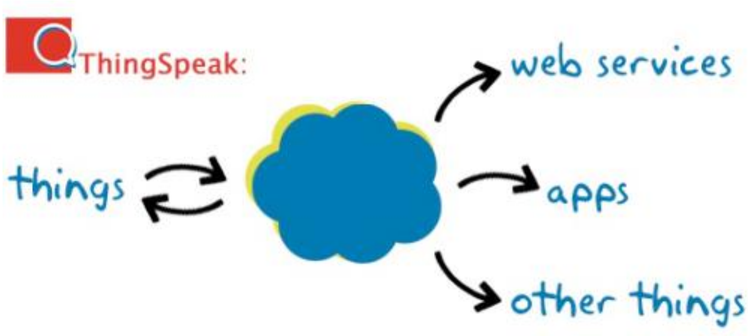

Figure 10: ThingSpeak representing itself as 'cloud' interface

We would want to collect data in ThingSpeak because sensors, or things, sense data and typically act locally. ThingSpeak enables sensors, instruments, and websites to send data to the cloud where it is stored in either a private or a public channel. ThingSpeak stores data in private channels by default, but public channels can be used to share data with others. Once data is in a ThingSpeak channel, you can analyze and visualize it, calculate new data, or interact with social media, web services, and other devices[55].

In this project, ThingSpeak will permit us to collect, analyze and act on collected data. ThingSpeak is an open application platform It receives data from any internet-connected device using message Querry Telemetry Transport (MQTT) [56]. This IoT platform is used to collect data in its respective channel, to analyze your data and to act on your data. It provides storage, analysis, and visualization of data [57]. ThingSpeak is used by following the three (3) steps. The first step is to create a channel and collect data. The next step is to analyze and visualize the online data. The last step is to act on the data [58]. In this work, the sensors have been connected to NodeMCU and the sensor data have been uploaded to ThingSpeak via MQTT protocol by using ThingSpeak channel number, ThingSpeak write API, ThingSpeak channel author [59]. Network SSID and network password have been used to connect NodeMCU to the network and the ThingSpeak channel enables us to monitor the water level and rain intensity.

2.7.3 Key features of ThingSpeak

ThingSpeak allows to aggregate, visualize and analyze live data streams in the cloud. Some of the key features of ThingSpeak include the ability to[60]:

Easily configure devices to send data to ThingSpeak using popular IoT protocols.

Visualize your sensor data in real-time.

Aggregate data on-demand from third-party sources.

Use the power of MATLAB to make sense of your IoT data.

Run your IoT analytics automatically based on events.

Prototype and build IoT systems without setting up servers or developing web software.

Automatically act on your data and communicate using third-party services.

2.8 If This Then That

IFTTT means that if some trigger takes place then action has to perform. IFTTT is a web-based free service used to create applets [61]. In this project, we have created an applet that has one app webhook. For creating this applet, we only need a computer with an internet connection after following the steps [62]. We get an URL that is used to trigger an alerting message whenever a stated trigger condition is met. With this IFTTT we get able to push message notifications [63]. We use the ThingHTTP app to trigger a notification from IFTTT.

Chapter 3. Research Methodology

This chapter explains the methods used in the implementation of our project. A good explanation and understanding of this chapter lead to a clear implementation plan.

3.1 Introduction

This chapter explains the methodology used to develop a flood monitoring system in Rwanda. Generally, the methodology is a chapter that reports details on how this study was conducted. This chapter) discusses the approaches that were employed to collect the data input and decision making to the public.

3.2 Methods

To carry out this work, ultrasonic sensor and rain drop sensors were used as input data of the system. NodeMCU ESP8266 was used as storage of data and as Ethernet shield of the system. The ThingSpeak platform was used to store, visualize and analyze the live data gathered by the sensors. IFTTT platform was also used to send alert notifications to the public authorities.

To effortlessly complete this project, a lot of research work on flood monitoring systems was required. This consisted of activities such as going through reference books, journals, internet resources and components datasheets which played a great role in the success of the project. This flood monitoring system consists of hardware and software parts.

3.2.1 Hardware schematic diagram

A flood monitoring system was constructed and assembled on a breadboard in the workshop. Figure 11 illustrates the hardware connection of the flood monitoring system where two flood detector systems were connected to the same computer shown in figure 12 to be tested and monitored using Arduino IDE as software, ThingSpeak web server, and IFTTT application as an online web service.

Mpazi flood detector and Rwampara flood detector hardware connections were both connected with the same components' breadboard, nodeMCU, ultrasonic sensor, rain drop sensor, buzzer, 5 LEDs, and jumper wires. Mpazi flood detector hardware connection is considered as flood detector system for Mpazi brook location and the Rwampara flood detector hardware connection is considered as flood detector system for Rwampara brook.

Each flood detector system observes the flood level of each brook as well as the rain intensity. HC-SR04 ultrasonic sensor was used to measure the rise of the water level. It emits ultrasound Waves by detecting the distance between the sensor and the obstacle (water). For receiving information about the rain intensity, a rain drop sensor was applied, and its operating was based on sensing the presence of rain on its plaque and it displays the analog values. The nodeMCU was used for transferring data to web applications. The LEDs are indicators of the flood status accordingly. The data were stored and analyzed in ThingSpeak through the MQTT protocol. The ThingSpeak generates a warning message to the authorized users using IFTTT

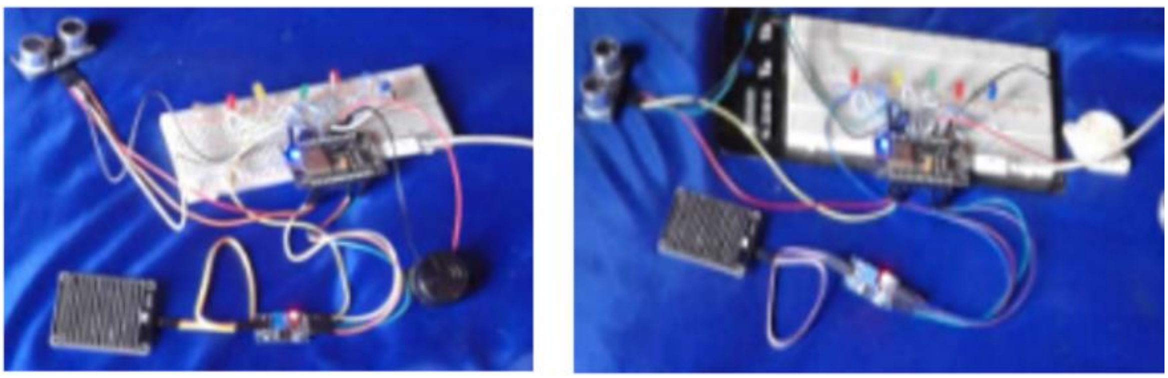

Figure 11: Mpazi and Rwampara flood detector systems hardware connections

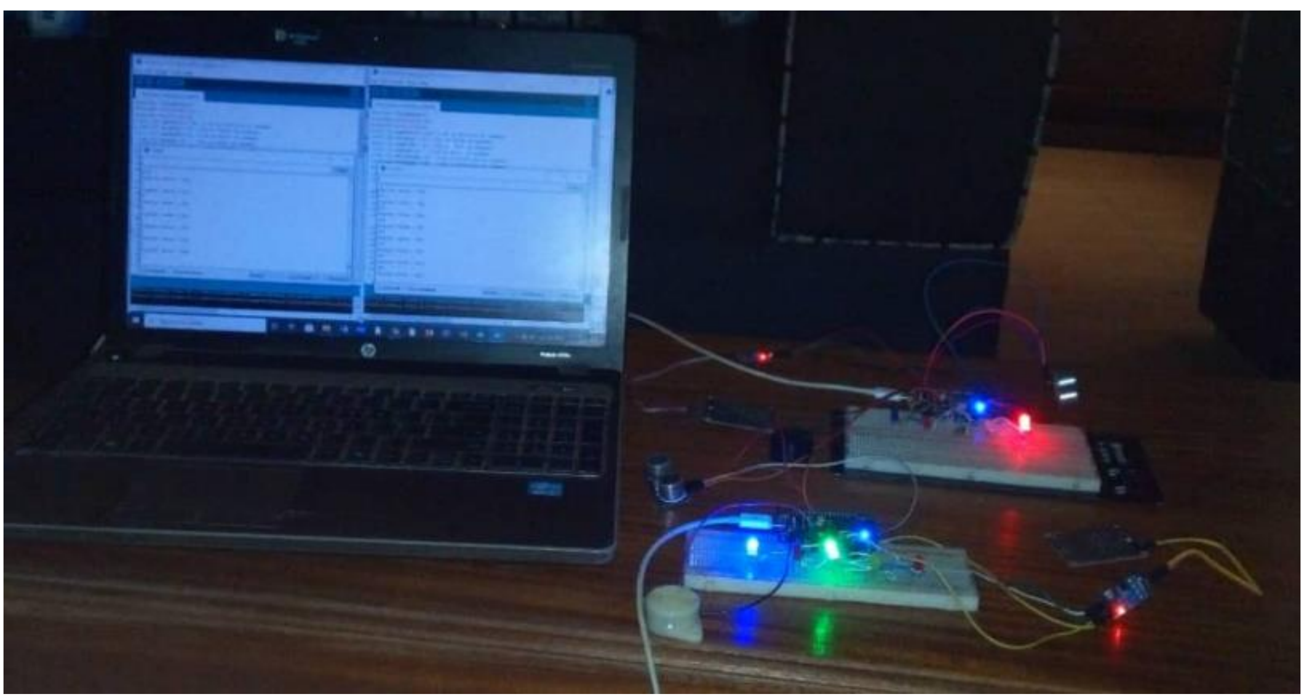

Figure 12: Flood detector systems connected on the same computer

The hardware part consists of Ultrasonic and rain drop sensors**,** nodeMCU esp8266**,** breadboard**,** LEDs and a buzzer. The nodeMCU is the brain of the whole system. It receives the input signals from the sensor. It also controls LEDs and the buzzer. All these processes were done according to the program. The ultrasonic sensor, rain drop sensor, LED, the buzzer is connected to nodeMCU. VCC of ultrasonic sensor and rain drop sensor is connected to VU of nodeMCU.

3.2.2 Software Connection

The **s**oftware part consists of the programming needed for the NodeMCU to perform its task. Algorithms are written to set how the microcontroller works and reacts according to the different scenarios such as reading the input signal from the sensor and flashing of LEDs and activating the buzzer when an alarm happens. For the software application, C programming has been selected as the main software used for the NodeMCU. To achieve these goals, a lot of hard work was required to program the software. In the process of completing the project, tasks like circuit designing, finding components, constructing prototypes and testing the functionality of the prototype were performed, followed by circuit fault diagnoses and troubleshooting.

3.2.3 Software implementation

The software requirement used in this system design is Arduino IDE and the language is done on Embedded C. The Arduino IDE is an integrated development environment designed to program the microcontroller. It is an open platform for some additional add-on boards such as ESP8266, STM32, etc. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation. It is also capable of compiling and uploading programs to the board with a single click [64]. The embedded C is used for supporting the embedded device. In this system, the sensors related to the device are connected to observe the state of flood detection in the system and programmed to notify the public authority immediately.

3.2.4 Integration

The integration between the software and hardware is made by using a nodeMCU with ESP8266. For the IoT part, nodeMCU uploaded the sketch of the coding of ThingSpeak application, Wi-Fi ID and Wi-Fi password. This allows nodeMCU to process the Wi-Fi module to connect the Wi-Fi and then connect to the ThingSpeak apps. With the connection of ThingSpeak, the nodeMCU proceeds the coding which acts as the controller to control the process of the system. The ultrasonic sensor and rain drop sensor act as the input data and process the inputs data in the nodeMCU to identify the current water level categories and the rain intensity and send data to ThingSpeak as graphs.

When the input data from the ultrasonic is matched to the warning level value which is set inside the coding of the nodeMCU, the nodeMCU, processed the part of the warning level process to send the warning level's notification. Meanwhile, for critical value, the system triggers the IFTTT to send an alert notification every 5secs.

Table 3: Condition for each water level

Water level	Height
Normal level	>=200cm
Warning level	100cm<=water level<200cm
Dangerous level	<=100cm

3.3 Working mechanism

The monitoring system includes three levels where the water is measured at the normal level, warning level and dangerous level. Ultrasonic sensors are used to detect the water level and for each level, the depth has been decided where for the condition to be at the normal level, the water must be greater than 200cm for normal level. The warning level is when the water level is between 100cm and 200 cm and the dangerous level is when the water is less than 100cm.

Throughout the three levels, users can indirectly monitor the current water level in the ThingSpeak application. During the safe level, a buzzer that acts as an alarm that is used to flow out any excess water will be OFF. This is because there is no triggering danger yet based on the current water level. However, as water continues to rise to the warning level, the public authorities will start to receive a notification alert on their email to remind them of the current water level. The same method is applied when the water level reaches the dangerous level. The public authorities will once again receive a notification alert. During these two situations, the alarm will finally turn ON. Figure 13 summarizes the working mechanism of the system in a flow chart. It shows that after the initialization of the system, the NodeMCU connects itself to the Wi-Fi and MQTT protocol and starts accepting the readings from the ultrasonic sensors and rain drop sensors. NodeMCU accepts the ultrasonic sensor readings and compares them with the programmed threshold values. The values of the ultrasonic sensor are mapped between 0-400 centimeters [65], [66]. We have divided the ultrasonic sensor readings into three states according to the level of water that can cause a flood or not. The readings are counted from the flood detector system where the ultrasonic sensor is connected to the surface of the water. If the water level is at a normal state, the message "no flood the water level is in normal condition" is sent. If the water level is at a critical state, it sends the message as "be aware of the flood may occur". If the water level is at a dangerous state, the message as “flood alerting alarm” is sent and the buzzer turned on.

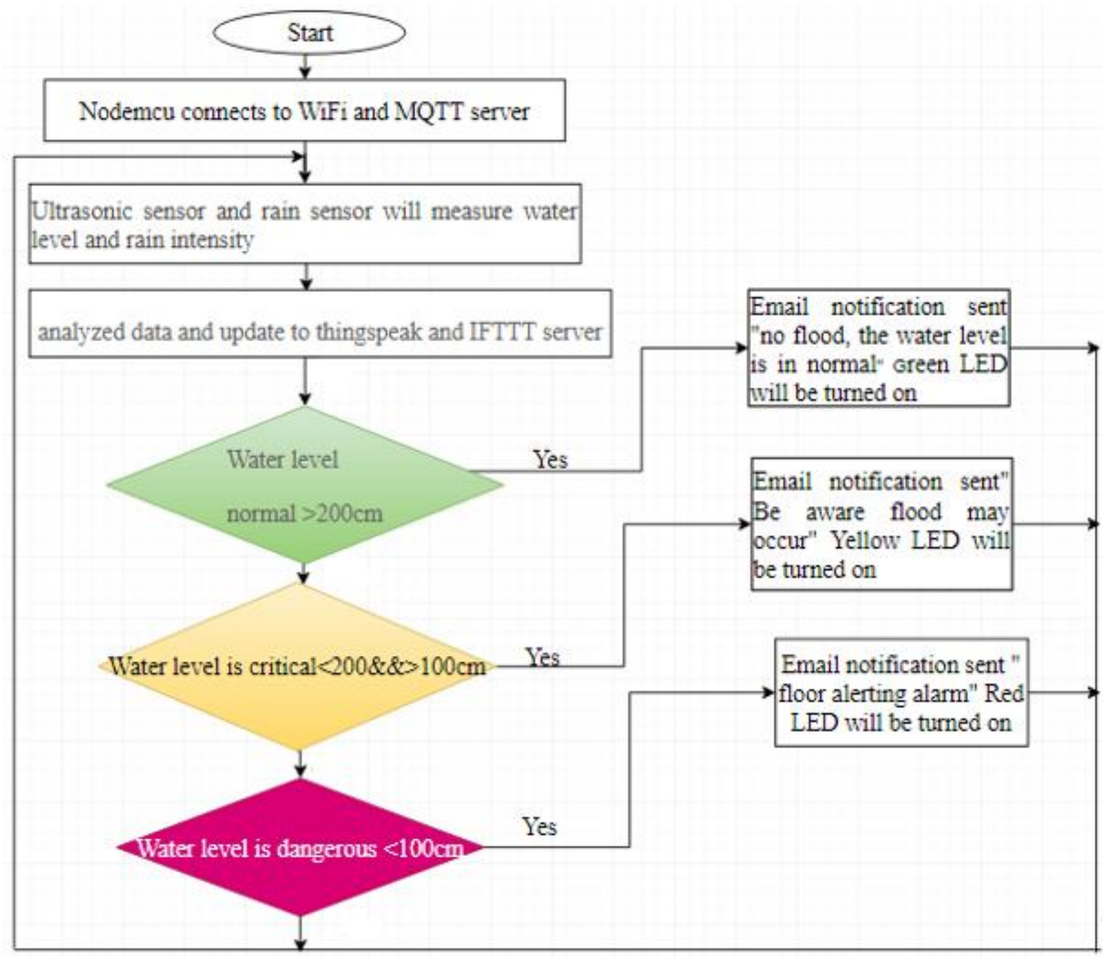

Figure 13: flowchart diagram

The values of the water level of the river and rain intensity around the location of the brook are read through NODEMCU and then it sends these values to ThingSpeak to update the web application with current data. The ThingSpeak will compare the sent water level with the threshold and if the condition meets the threshold value set then the ThingSpeak will automatically trigger an external application IFTTT.

Installing ESP8266 board in Arduino IDE

The installation procedure for adding ESP8266 board in Arduino IDE is very simple with the latest versions of Arduino IDE. Follow these steps

Step1: Open Arduino IDE

After installing the latest version of Arduino IDE, you have to open it from the desktop icon. After opening it, you click on the File tab and then open Preferences. Add additional boards manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json.

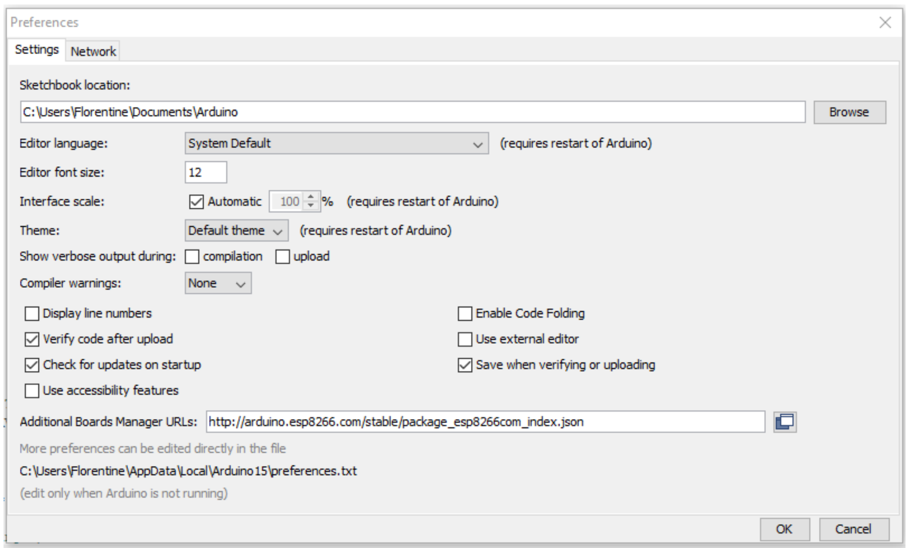

Figure 14: preferences

Step 2: Open board manager

Go to Tools>>Boards>>Boards Manager

Figure 15: board manager

Step 3: Install ESP8266 Board

Search for ESP8266 and click the esp8266 Community option then click install button. Note: If Step 1 URL is not added then it will not find ESP8266 board.

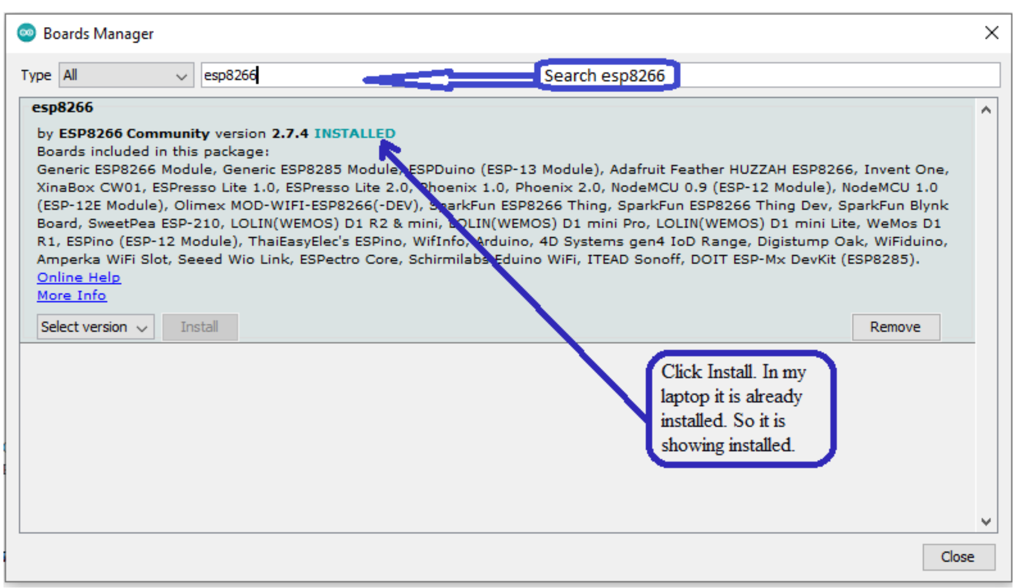

Figure 16: Install ESP8266

Step 4: After clicking install, the installation process is shown at the bottom.

Once it is installed, open **tools>>boards** and look for ESP8266 to select the board

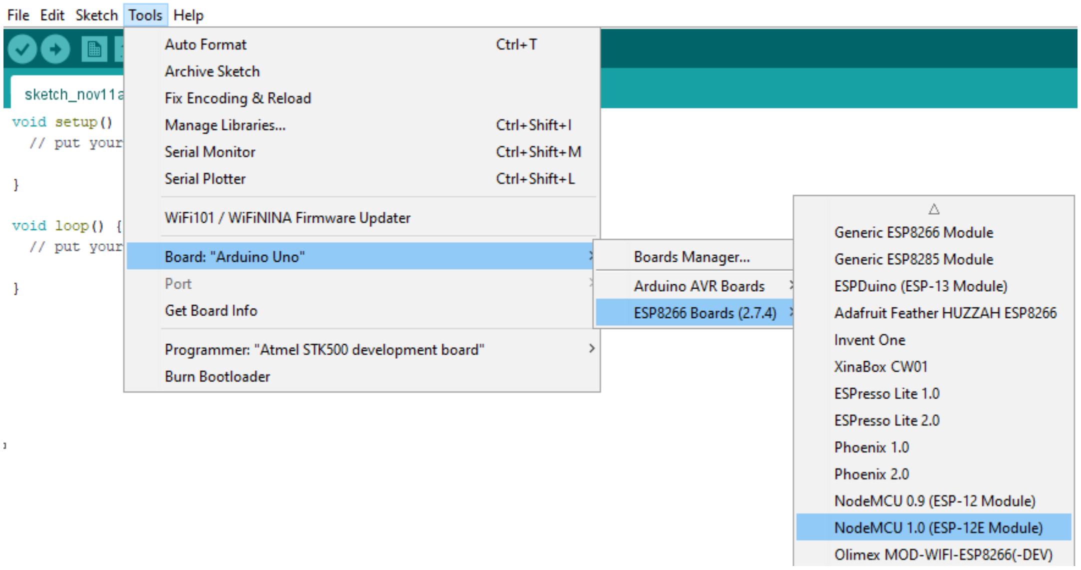

Figure 17: board selection

3.4 Checking installation is working

Step1: Select the type of board we are using in arduino IDE. Open Arduino IDE and click on tools>>Boards then select NODEMCU 1.0 (ESP - 12E Module) as shown in figure 17. Figure 17 shows that NodeMCU is included in board manager of our Arduino IDE.

Step 2: Open LED blink example for Esp8266

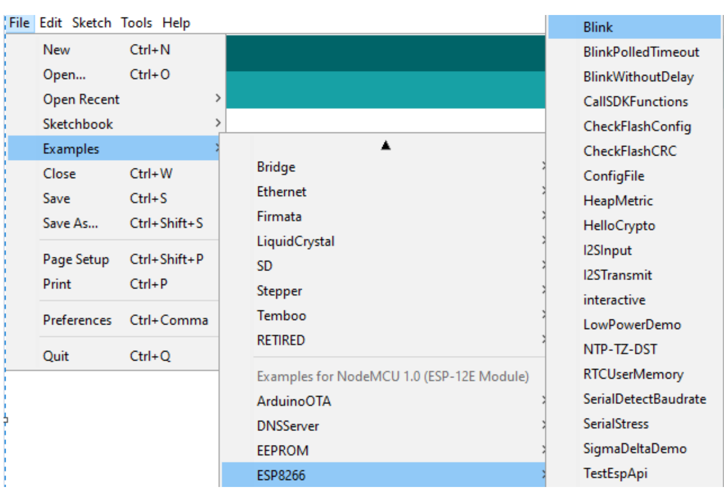

Figure 18: Open blink LED

Step 3: Modify or Write New Program to make GPIO pin LED blinking

Step4: Uploading program to NodeMCU: Select your board's com port. By making sure that drivers are installed on our board. NodeMCU uses CP2102 as USB to Serial converter.

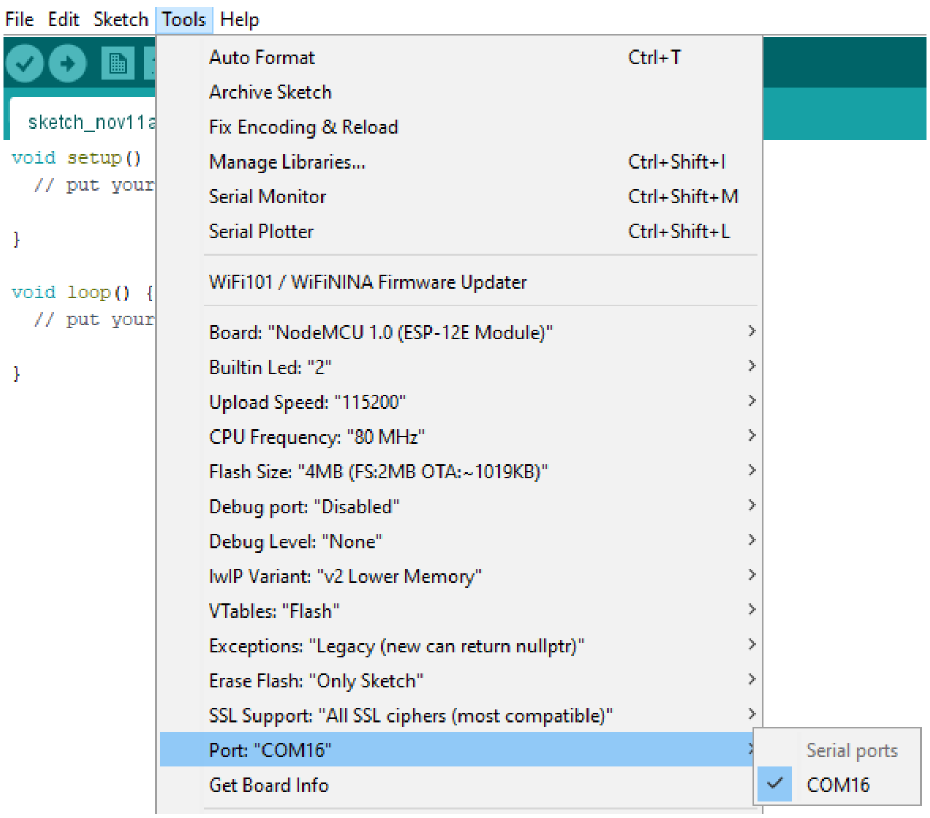

Figure 19: Communication port selection

When you click on upload button, press and hold FLASH button of NodeMCU which is present near USB connection. Once upload is started (blue led blinks at faster rate), you can release it. Pressing of FLASH button is not required if selected board is NodeMCU 1.0.

- Click on tools to select the port we are using.
- Change the Wi-Fi name and password from the following code in table 4.
- Again click on tools to select the port we are using.
- Change the Wi-Fi name and password from the following code in table 4.

Table 4: Connecting NodeMCU to ESP8266Wi-Fi

```
  #include <ESP8266Wi-Fi.h>
  #include <Wi-FiClient.h>
  Wi-FiClient  client;
const char* ssid = "My PC"; //Your Network SSID
const char* password = "Florentine28225"; //Your Network Password
void setup()
{
  Serial.begin(115200);
  Wi-Fi.begin(ssid, password);
  while (Wi-Fi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Wi-Fi connected");
  Serial.println(Wi-Fi.localIP());
}
```

- Click on Upload button to upload the code.
- Power up the board and open the serial monitor from arduino IDE
- After connecting to the Wi-Fi it will show you the IP address.

NodeMCU V3 has inbuilt Wi-Fi which causes it to be used in the Wi-Fi Applications whereas most of the other embedded board uses some external Wi-Fi to process data.

Chapter 4: Analysis and Design

The system analysis describes the system work system in general whereas the system design explains the material requirements for the design of flood detection systems and the design of two locations of flood visualization systems.

4.1 System Analysis

The system analysis is presented in figure 20 where the flood detector system detects water levels and the rain intensity. Furthermore, the flood detector system sends the data of flood level and rain conditions to the flood monitoring information system server using WiFi. Data entering the flood monitoring information system will process the data and visualization is done in real-time. Data can be visualized and accessed by the online user.

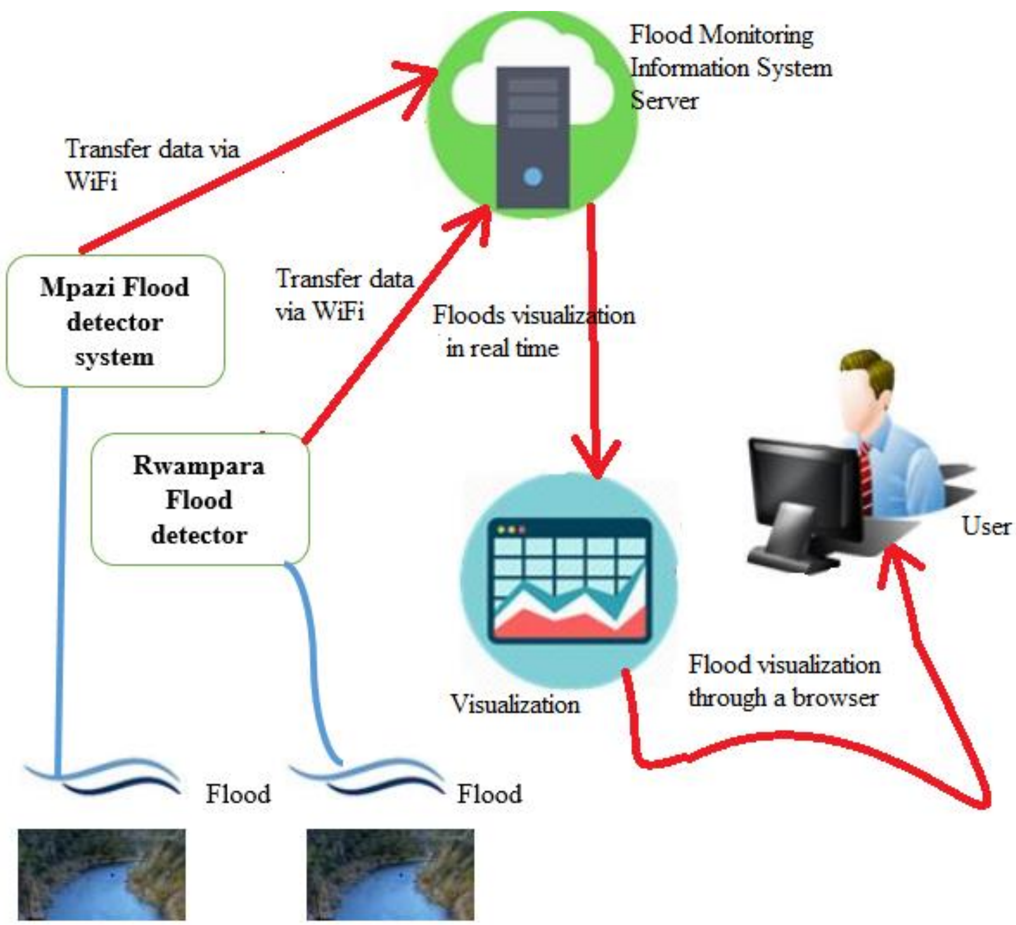

Figure 20: Flood monitoring system analysis

4.2 System Design

The system design in this project explains the material requirements as shown in Figure 21. The prototype designed consists of two parts, namely the design of a flood detection system and the design of a flood monitoring information system. The flood detection system is constructed using sensors as input of the system including ultrasonic sensors to measure water levels and rain drop sensors to determine the rain conditions. The sensors are connected to the nodeMU as a processor. The design of a flood monitoring system will display flood visualization using ThingSpeak application server.

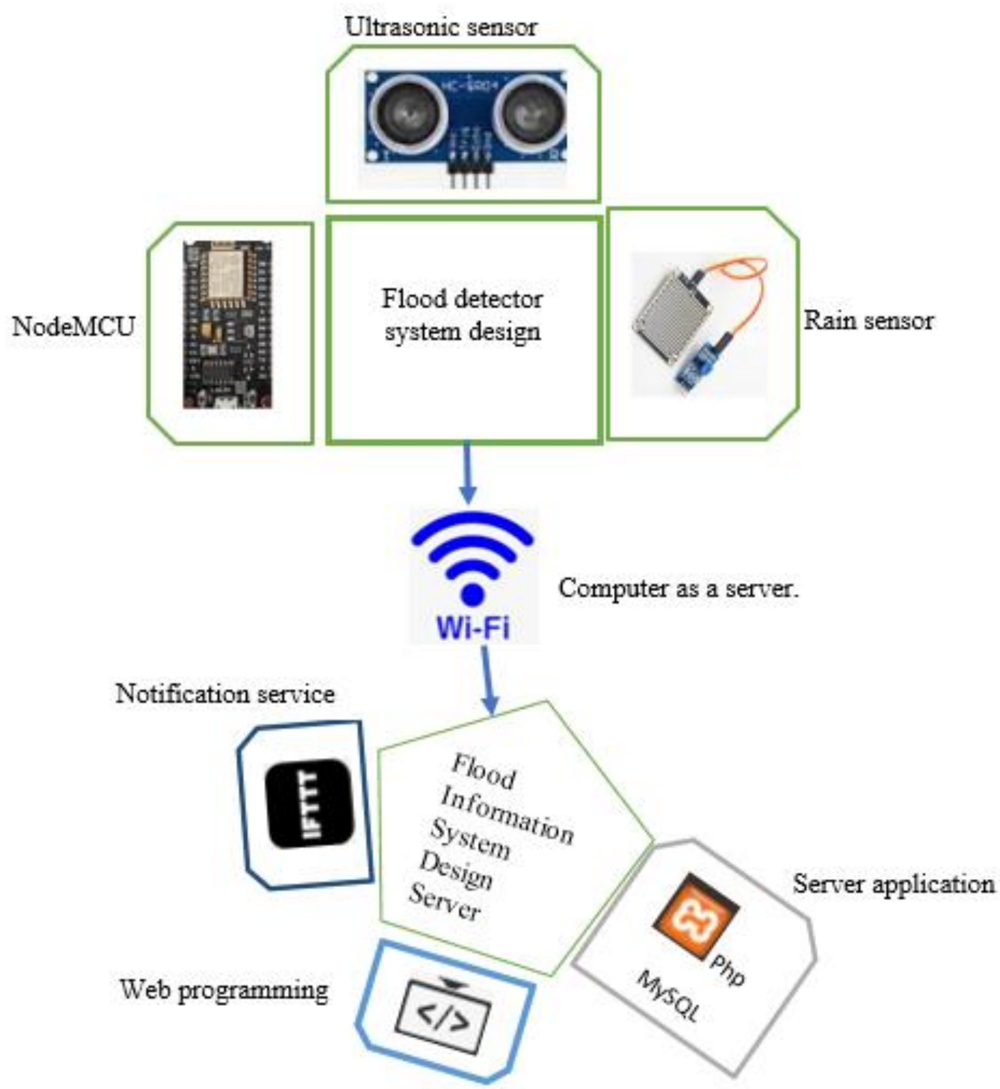

Figure 21: System design

In figure 21, there is a block of software systems, namely flood monitoring information systems, there are several supporting software, namely Xampp and Web programming components. Xampp is a server application consisting of a PHP engine that functions as a PHP programming language processing engine, Apache Web Server as a Web Server application and MySQL is a database management server. The development of the server component system manages the data received from the flood detector system to be stored in a database and can be accessed by users. While the web programming component is a component of web creation and flood visualization.

IoT is being implemented in the design of this project where it is used as a foundation for data transmissions between the detection devices to the ThingSpeak application. Ultrasonic sensor helps to measure the water level and the rain drop sensor is being implemented to know the rain intensity of the brook location. Flood monitoring system helps the citizens to make early preparation after being informed by the public authorities who received alert notification. They are also able to monitor water level and rain intensity at any time of the day. This system is made to alert the authorities about the condition of water level so that users have enough time to make preparation before flood occur. Figure 21 shows the system design of our project where there are two inputs in the system consisting of the ultrasonic sensor and the rain drop sensor used to measure the availability of rain and the data are processed in the NodeMCU controller. Meanwhile, the outputs of the system include the IoT platform which are the ThingSpeak web application and the IFTTT platform. Table 5 summarizes the function of the components used.

Table 5: Function of each components used.

Name	**Function**
Ultrasonic sensor	To measure the distance between the sensor and the water
Rain drop sensor	To measure the rain intensity
NodeMCU	To collect sensor data and connects the system to the internet when Wi-Fi is available
IoT platform	To Support ThingSpeak application and IFTTT application. To display sensor information and notification

Figure 22 shows the Flood data format through the flood detector systems to the flood monitoring information system server.

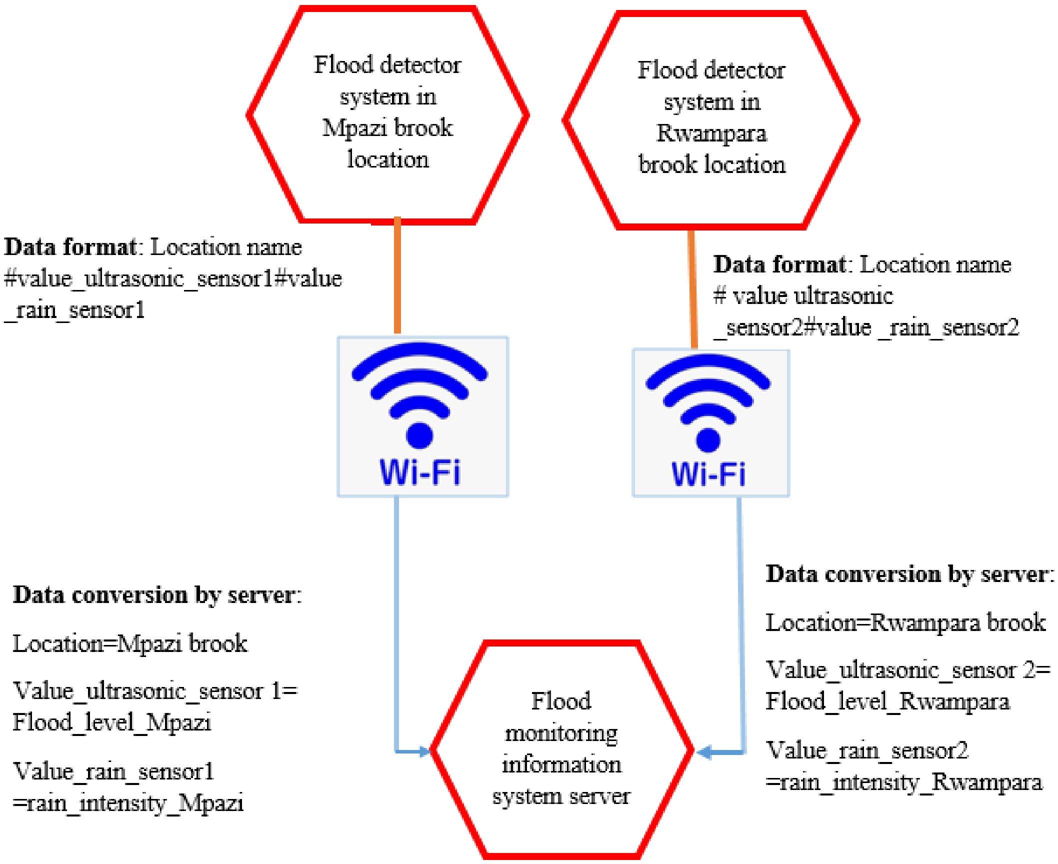

Figure 23: Flood data format through the flood detector system to the flood monitoring information system server.

4.4 System model

The proposed system model can be described in Figure 23. In general, the work system has two flood locations, namely Mpazi flood and Rwampara flood. Each of these locations has a flood detector system. Each flood detector system sends data to the Thingspeak server using MQTT protocol. The ThingSpeak server processes the data to produce a flood visualization that can be accessed by the user. Users access the flood visualization system through a browser.

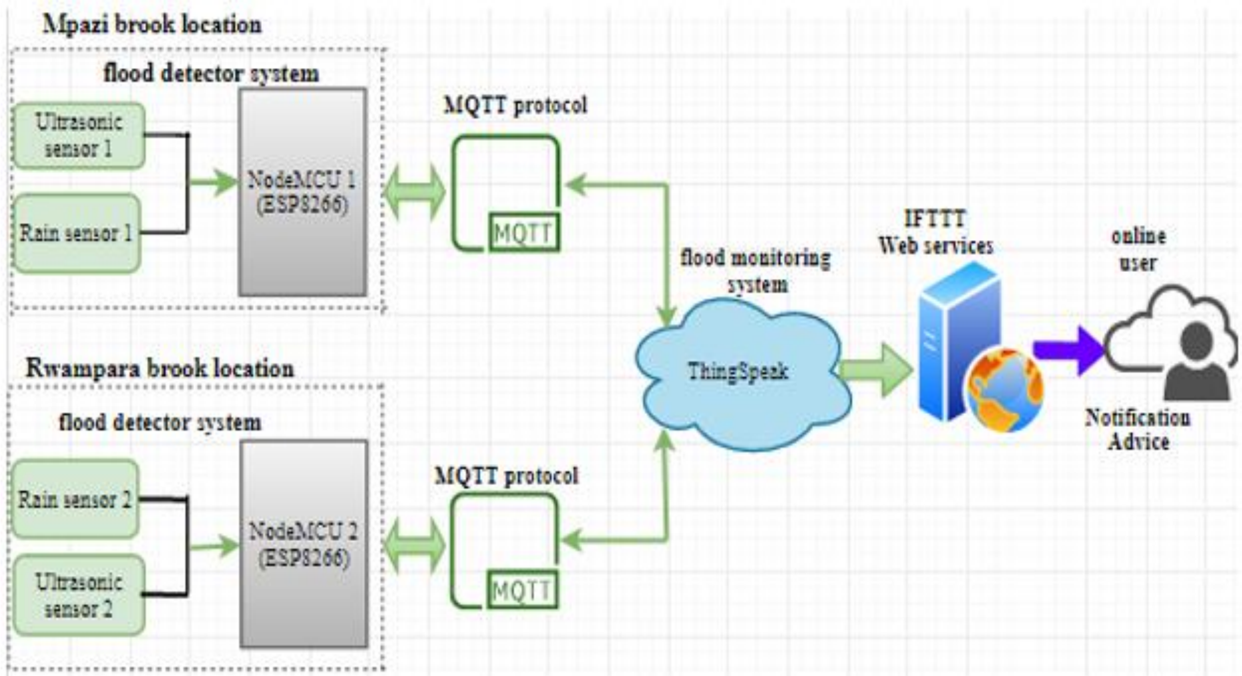

Figure 24: IoT flood monitoring system model

4.5 Verification of the system

The flood detector system composed of ultrasonic sensors, rain sensors, nodeMCU, MQTT protocol, and ThingSpeak has been tested and analyzed for ensuring that the system can give the alert message. The ultrasonic sensor, buzzer, LEDs and nodeMCU has been checked to be sure that they are working properly.

Chapter 5: Results and discussions

This chapter explains our findings and explains the graphs of the obtained results. After implementing the flood monitoring system prototype on the breadboard along with nodeMCU and sensors as shown in the figure. It has been tested at the Rwampara brook and Mpazi brook. The remote correspondence between NodeMCU, ThingSpeak server, and IFTTT has been achieved. The fields of ThingSpeak channel are updated and the triggering based on live data is well done. The notification email is sent to the public authorities through ThingSpeak and IFTTT web services.

Based on the system model, the values of the two flood detectors are read through nodeMCU. Each detector sends its values to ThingSpeak to update the fields with current data. The ThingSpeak visualizes and analyzes the real-time data of Mpazi brook and Rwampara brook.

5.1 Connection Testing and results

The testing of the flood monitoring system in Rwanda is carried out by conducting a connection test which includes sending the data to the Thingspeak web application. The analog data provided by sensors are sent to the ThingSpeak web application as graphics as well as numerical values via NodeMCU. The system was built to automatically sense the water level and rain intensity at the two brooks using the HC-SR04 ultrasonic sensor and rain drop sensor and then send gathered data to the Thingspeak application through NodeMCU. The speed of the data transmission network is influenced by the cellular network signal, the sending time and the Api from the Thingspeak application.

5.2 The obtained results on ThingSpeak website.

The real data gathered by sensors located in two different locations namely Rwampara brook and Mpazi brook are visualized on ThingSpeak application. Figure 24 shows the obtained results and their details.

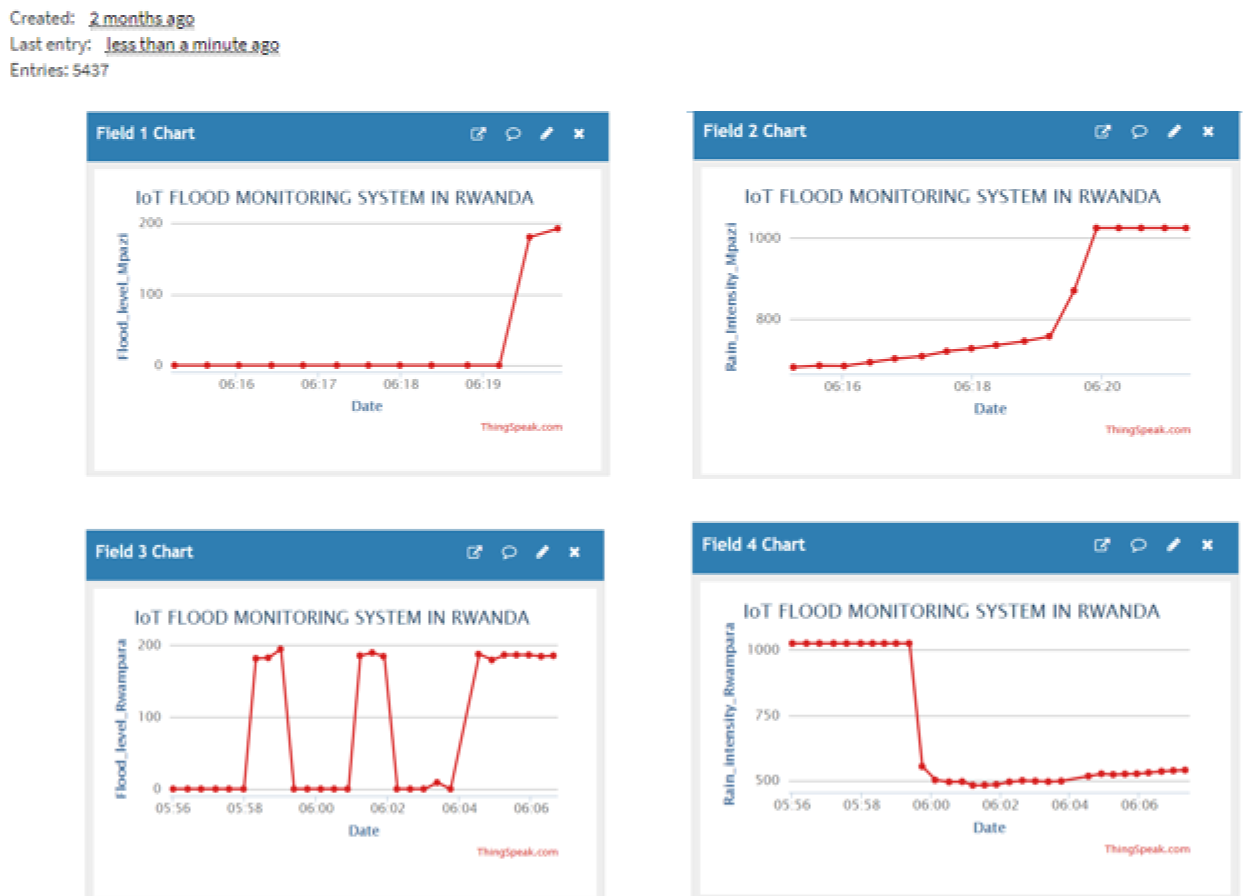

Figure 25: The results obtained on ThingSpeak

The obtained results from ThingSpeak are located in one channel called IoT flood Monitoring system as shown in figure 24. It has four fields. Field 1 and field 2 represents data gathered by flood detector system located in Mpazi brook location. Field 3 and field 4 represent the data captured by flood detector system located in Rwampara brook location.

5.2.1 Data captured by ultrasonic sensor located in Mpazi brook location.

Figure 25 shows the data captured by ultrasonic sensor located in Mpazi flood detector. It measures the flood level of Mpazi brook in centimeters. The y-axis shows the measured values of Ultrasonic sensor. The X-axis shows the time. The Y-axis shows that the distance from the water level of Mpazi brook to the ultrasonic sensors varies from 8cm (the value given by the serial monitor) to 200cm. From 06:15 to 06:19 it was 8cm means that there was a flood at this time because the water level is reaching the ultrasonic sensor. This flood is caused by the heavy rain which was raining at this location shown in figure 26. From 06:19 there is no flood because the distance from ultrasonic sensor to the water level is very high.

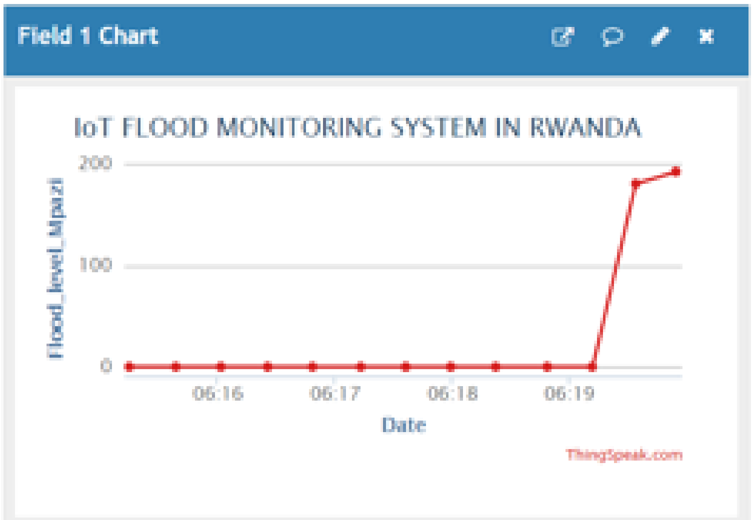

Figure 26: Data captured by ultrasonic sensor located in Mpazi brook location.

5.2.2 Data captured by rain drop sensor located in Mpazi brook location.

Figure 26 shows the data captured by rain drop sensor located in Mpazi flood detector. It measures the rain intensity of Mpazi brook location. The y-axis shows the rain drop sensor readings. The X-axis shows the time. The sensor readings vary from 0 to 1024. The rain drop sensor readings show that from 06:15 to 06:17 there is heavy rain. From 06:17 to 06:19 there is moderate rain. The sensor readings show that there is no rain from 06: 19. Due to these rain drop sensor readings, ultrasonic sensor readings show that there is no flood as shown in figure 25.

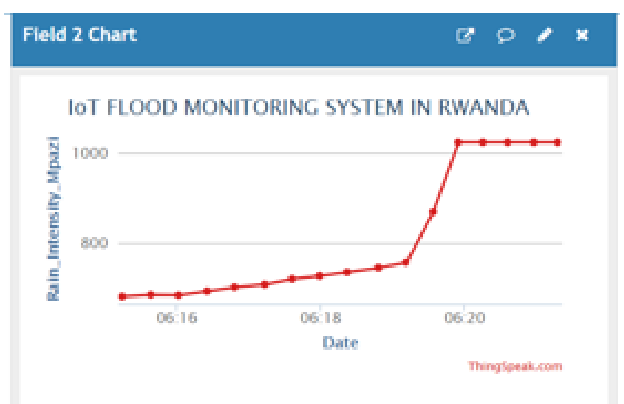

Figure 27: Data captured by rain drop sensor located in Mpazi brook location.

5.2.3 Data captured by ultrasonic sensor located in Rwampara brook location.

Figure 27 shows the data captured by ultrasonic sensor which located in Rwampara flood detector. It measures the flood level of Rwampara brook in centimeters. The y-axis shows the measured values of Ultrasonic sensor. The X-axis shows the time. The Y-axis shows that the distance from water level of Rwampara brook to the ultrasonic sensors varies from 8cm (the value given by serial monitor) to 200cm. The graph shows that before 05:56 there was a flood in Rwampara brook at this time because the water level is reaching the ultrasonic sensor. From 06:04 there is no flood.

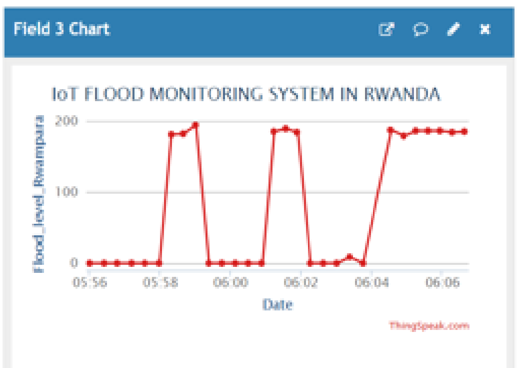

Figure 28: Data captured by ultrasonic sensor located in Rwampara brook location.

5.2.4 Data captured by rain drop sensor located in Rwampara brook location.

Figure 28 shows the data captured by rain drop sensor which located in Rwampara flood detector. It measures the rain intensity of Rwampara brook location. The y-axis shows the rain drop sensor readings. The X-axis shows the time. The sensor readings vary from 0 to 1024. The rain drop sensor readings show that from 05:56 to 06:00 there is heavy rain. From 06:00 there is heavy rain.

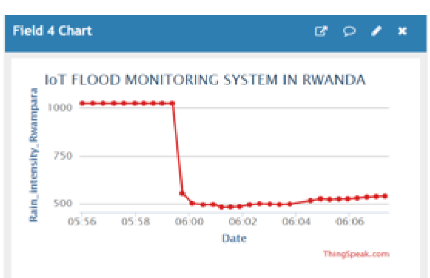

Figure 29: Data captured by rain drop sensor located in Rwampara brook location.

As it is shown by the charts above the flood level varies by the intensity of rain in the same location. When it is raining as it is shown in the field 2 charts, the flood level increases as it is shown in field 1 chart. As it is shown in the field 3 charts and field 4 charts, the flood level can increase without heavy rain. This means that the flood can be caused by the water comes from different surrounding areas as is shown from 05:56 to 06:04.

The flood monitoring system can be also visualized using a web application created by Xampp software by typing localhost/iotfms/ in any browser and the obtained result are shown in figure 29.

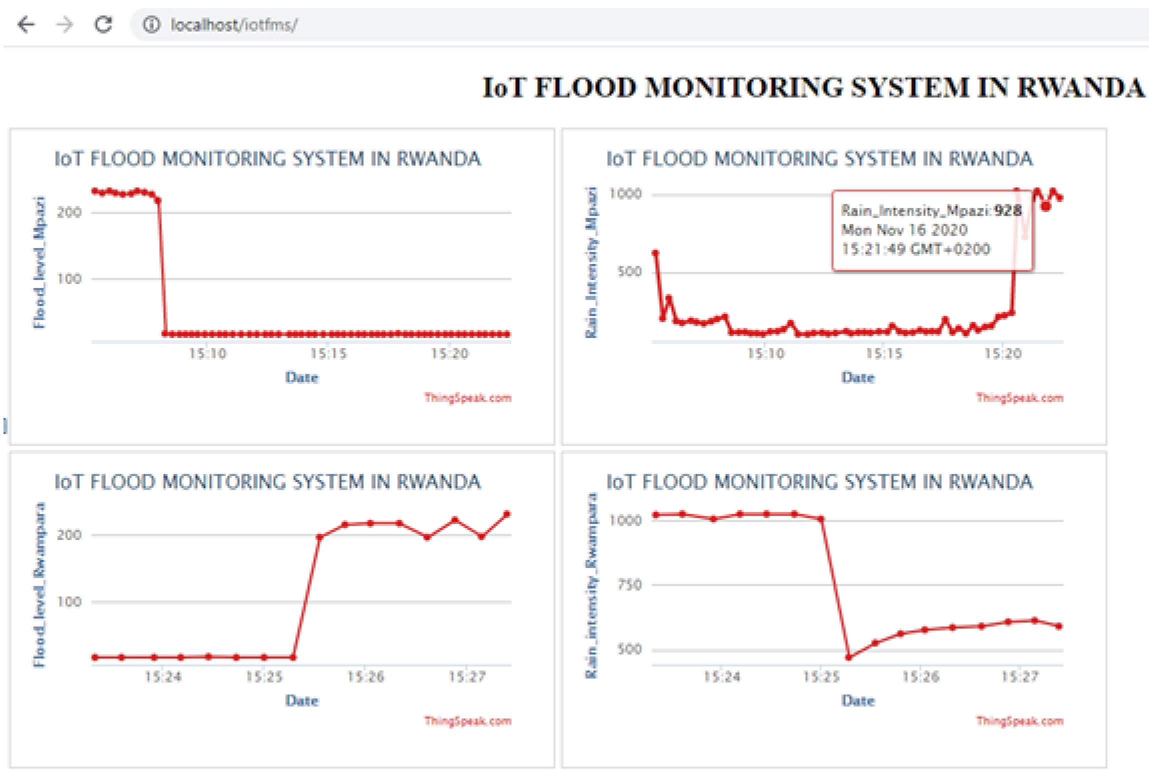

Figure 30: IoT flood monitoring system web application results

The ThingSpeak compares the flood level and rain intensity. If there is a manifestation of a flood, the system sends the alert message to the end-users as shown in figure 30. It alerts the users to be aware of the flood. When the flood has occurred, the message of flood alerting alarm is sent as it is shown in figure 31. The message alerts contain information about the type of alert message and the time of the alerting message. When there is no flood, the message is sent showing that the water level is in a normal state.

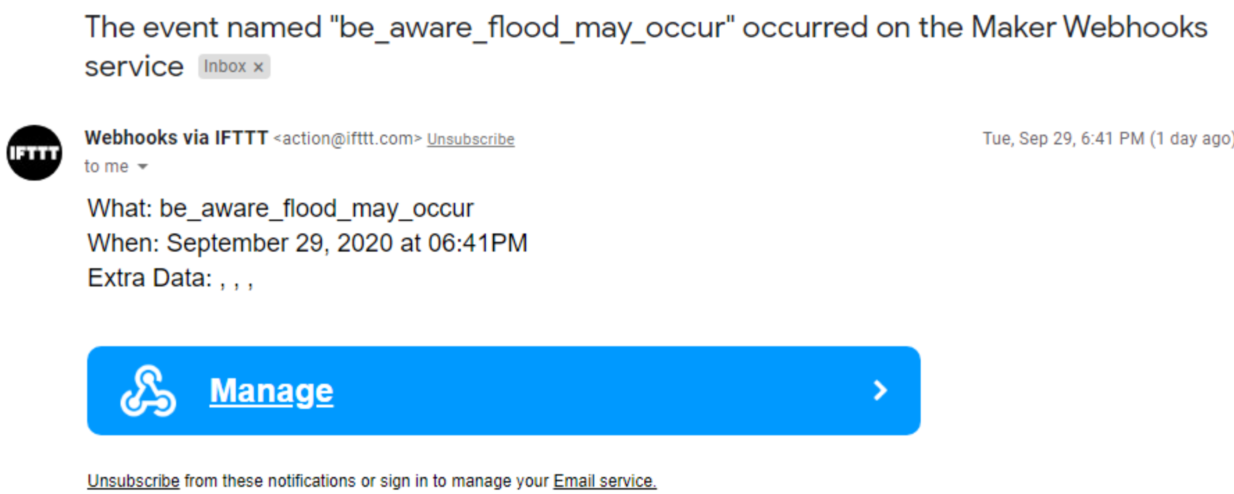

Figure 31: Be aware of flood may occur

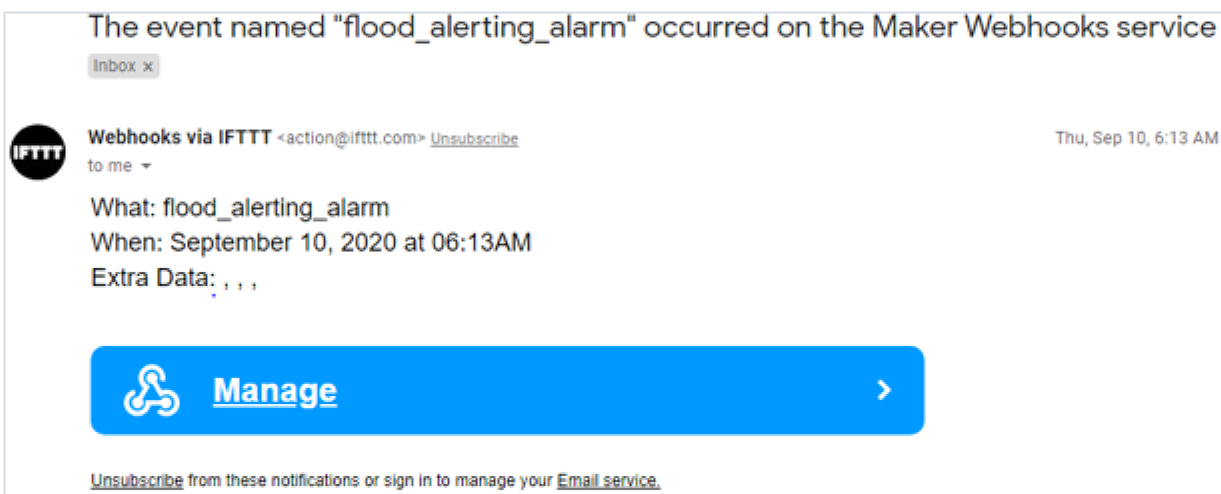

Figure 32: Flood alerting alarm notification

Chapter 6: Conclusion and recommendation

6.1 Conclusion

The flood monitoring system was developed by collecting real-time data using ultrasonic sensor, rain drop sensor and NodeMCU esp8266. The data processed by nodeMCU are transferred to ThingSpeak Flood monitoring system via MQTT protocol. The ThingSpeak server corrects the information sent from the flood detector system in the form of visualization of flood data at two locations in real-time. The system prototype was tested and worked successfully according to the obtained results. The technology solution of this flood monitoring system can contribute to managing floods impact and protects the population as well as to help the organizations in the management of the environment.

The main objective of this study consists of a flood monitoring system in Rwanda using IoT. After examining the features of IoT and its provided services, it was found the best technology to be used by the flood monitoring system with high accuracy. This high accuracy of IoT has been confirmed by the prototype done and the transfer of data given by the developed prototype. The analysis of the received signal allows us to conclude that the public authorities visualize data and receive an email alert from the IFTTT application. Therefore, the hypothesis of our dissertation has been confirmed.

6.2 Recommendation

This system is so open to being improved to provide a better performance, and here are some suggestions to be included in any future work:

Once there is further implementation, cameras and drones can be added to the system for accurate detection and for immediate information about real-time activity like supporting in first aid, or special logistic.

List of References

[1] V. Mukanyandwi, L. Nahayo, E. Hakorimana, A. Gasirabo, and S. Otgon, "Review on Water Resources Management and Key Threats in Rwanda, East Africa," J. Water Secur., vol. 4, no. 0, pp. 1–8, 2018.

[2] "Watersheds, flooding, and pollution," 2019. [Online]. Available: https://www.noaa.gov/education/resource-collections/freshwater/watersheds-flooding-and-pollution. [Accessed: 29-Apr-2020].

[3] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, and H. Jahankhani, Internet of Things Digital Twin Technologies and Smart Cities. 2020.

[4] A. Liotta and G. Fortino, "Internet of Things Technology, Communications and Computing," S. Alessandro, B. Margo, and R. Paul, Eds. Australia, 2019, p. 301.

[5] KPMG, "Smart Cities – Adoption of Future Technologies," no. January, 2020.

[6] E. A. S. Ahmed and Y. Mujtaba, Elbagir, "Internet of things in Smart Environment: Concept, Applications, Challenges, and Future Directions Article Contents," Psjd.Icm.Edu.Pl, vol. 134, no. 1, pp. 1–51, 2019.

[7] I. Nhapi, "Assessment of Water Pollution Levels in the Nyabugogo Catchment, Rwanda," Open Environ. Eng. J., vol. 4, no. 1, pp. 40–53, 2011.

[8] MIDIMAR, "The National Risk Atlas of Rwanda," p. 197, 2015.

[9] L. Nzabonantuma, "Morphodynamic Study of A River to Attenuate Flood Waves, Case Study of Nyabugogo River in Rwanda," vol. 8, no. 2, pp. 78–90, 2015.

[10] F. Plain, K. Cit, J. P. Bizimana, and M. Schilling, "Geo-Information Technology for Infrastructural Flood Risk Analysis in Unplanned Settlements : A Case Study of Informal Settlement Flood Risk in Chapter 6 Geo-Information Technology for Infrastructural Flood Risk Analysis in Unplanned Settlements : A Case ," no. January 2010, 2016.

[11] WGR, "IWRM Programme Rwanda," no. March, pp. 2018–2024, 2018.

[12] D. Loczy, S. Czigany, and E. Pirkhoffer, "Flash Flood Hazards," Stud. Water Manag.

Issues, no. January 2016, 2012.

[13] M. Ivan R., “No floods expected in coming rainy season,” August 01, 2013, Kigali, p. 2, 05-Apr-2013.

[14] REMA, “Republic of Rwanda, Rwanda Environmental Management Authority, Atlas of Rwanda’s changing environment, implications for climate change resilience,” p. 92, 2011.

[15] C. Julie, “Heavy rains, floods and landslides claim at least 60 lives in Rwanda, more than 1 000 homes damaged,” March 11, 2020. [Online]. Available: https://watchers.news/2020/03/11/rwanda-floods-landslides-2020/. [Accessed: 07-Apr-2020].

[16] N. FLOODLIST NEWS IN AFRICA, “Rwanda – at Least 13 Killed in Floods and Landslides,” 4 FEBRUARY, 2020. [Online]. Available: http://floodlist.com/africa/rwanda-floods-landslides-february-2020. [Accessed: 11-Feb-2020].

[17] Floodlist-news, “Rwanda – Heavy Rain Leaves 8 Dead, Homes Destroyed,” 2020. [Online]. Available: http://floodlist.com/africa/rwanda-floods-may-2020. [Accessed: 21-Oct-2020].

[18] Floodlist-news, “Rwanda – Torrential Rain and Floods Leave Over 60 Dead,” 2020. [Online]. Available: http://floodlist.com/africa/rwanda-floods-07-may-2020. [Accessed: 21-Oct-2020].

[19] K. Hudson, “Heavy rains pound Kigali,” January 29, 2020. [Online]. Available: https://www.newtimes.co.rw/news/heavy-rains-pound-kigali. [Accessed: 27-Feb-2020].

[20] L. O. B. A. L. Ec, N. O. Mic, and E. C. Ts, “Summary and key messages.”

[21] M. Emmanuel Come and B. Bertrand, “Rwanda: Heavy Rains - Death Toll Rises to 19,” Kigali, p. 2, 05-Feb-2020.

[22] J. W. Simatupang and F. Naufal, “Flood Early Warning Detection System Prototype Based on IoT Network,” Internetworking Indones. J., vol. 11, no. 1, pp. 17–22, 2019.

[23] J. (Circuits Today), “Interface GSM Module to Arduino-Send and Receive SMS,” 22019. [Online]. Available: https://www.circuitstoday.com/interface-gsm-module-with-arduino. [Accessed: 20-Apr-2020].

[24] U. Manual, “Gprs / Gsm Sim900a Modem With Arduino Compatible,” 2014.

[25] S. Patil, J. Pisal, A. Patil, S. Ingavale, P. Ayarekar, and P. S. Mulla, “A Real Time Solution to Flood Monitoring System using IoT and Wireless Sensor Networks,” pp. 1807–1811, 2019.

[26] M. Syafiq, M. Sabre, S. S. Abdullah, and A. Faruq, “Flood warning and monitoring system utilizing internet of things technology,” vol. 4, no. 3, 2019.

[27] M. S. Mohd Sabre, S. S. Abdullah, and A. Faruq, “Flood Warning and Monitoring System Utilizing Internet of Things Technology,” Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control, vol. 4, no. 4, pp. 287–296, 2019, doi: 10.22219/kinetik.v4i4.898.

[28] G. Singh, N. Bisht, P. Bisht, and P. Singh, “Iot Based Flood Monitoring and Alerting System with Weather Forecasting,” Int. J. Innov. Technol. Explor. Eng., vol. 9, no. 6, pp. 559–563, 2020.

[29] N. V. S. S. Varma, E. E. Preethi, M. R. Kumar, and R. K. Tenali, “Internet of things based smart flood monitoring & detecting system,” Int. J. Recent Technol. Eng., vol. 7, no. 6, pp. 1335–1337, 2019.

[30] A. Diriyana et al., “Water Level Monitoring and Flood Early Warning Using Microcontroller With IoT Based Ultrasonic Sensor,” J. Tek. Inform. C.I.T, vol. 11, no. 1, pp. 22–28, 2019.

[31] Roderick Burnett, “Understanding How Ultrasonic Sensors Work,” 24-03-2020, 2020. [Online]. Available: https://www.maxbotix.com/articles/how-ultrasonic-sensors-work.htm. [Accessed: 26-Oct-2020].

[32] U. Sensor and D. Measurement, “alselectro Arduino – UltraSonic Sensor for Distance Measurement,” pp. 1–11, 2017.

[33] R. Reeve, “Ultrasonic sensing,” Ultrasonics, vol. 3, no. 4, p. 217, 1965.

[34] Arbi Abdul Jabbaar, “Ultrasonic Sensor HC-SR04 with Arduino Tutorial,” 2019.

[35] E. Udo, E. Isong, “Flood Detection System Using Wireless Sensor Network,” vol. 5, no. 2,

pp. 386–389, 2015.

[36] E. Projects, “Ultrasonic sensor interfaced with nodemcu esp8266 12e for distance measurement,” April 10, 2019, 2019. [Online]. Available: https://www.engineersgarage.com/esp8266/distance-measurement-with-hcsr04-and-nodemcu/. [Accessed: 23-Oct-2020].

[37] “Rain Drop Size.pdf.” .

[38] D. Electronics, “Rain drop Sensor Module,” 20 December 2019, 2019. [Online]. Available: https://components101.com/sensors/rain-drop-sensor-module. [Accessed: 15-Oct-2020].

[39] T. H. Nasution, E. C. Siagian, K. Tanjung, and Soeharwinto, “Design of river height and speed monitoring system by using Arduino,” IOP Conf. Ser. Mater. Sci. Eng., vol. 308, no. 1, pp. 1–8, 2018.

[40] D. Satria, Zulfan, Munawir, and T. Hidayat, “Implementation of wireless sensor network (WSN) on garbage transport warning information system using GSM module,” J. Phys. Conf. Ser., vol. 1175, no. 1, 2019.

[41] A. Al Dahoud and M. Fezari, “NodeMCU V3 For Fast IoT Application Development,” Notes, no. October, p. 5, 2018.

[42] A. Aqeel, “Introduction to NodeMCU V3,” October 11, 2018, 2018. [Online]. Available: https://www.theengineeringprojects.com/2018/10/introduction-to-nodemcu-v3.html. [Accessed: 23-Oct-2020].

[43] V. kumar peddinti, “Light Emitting Diodes (LEDs) ELE 432 Assignment # 3 Vijay Kumar Peddinti,” p. 200, 2018.

[44] A. Dean and M. O. Agyeman, “A study of the advances in IoT security,” ACM Int. Conf. Proceeding Ser., no. April 2019, 2018.

[45] GSMA, “Understanding the Internet of Things (IoT),” Gsma Connect. Living, no. July, p. 15, 2014.

[46] K. K. Patel, S. M. Patel, and P. G. Scholar, “Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future

Challenges," Int. J. Eng. Sci. Comput., vol. 6, no. 5, pp. 1–10, 2016.

[47] S. K. Datta and C. Bonnet, "Internet of Things and M2M Communications as Enablers of Smart City Initiatives," Proc. - NGMAST 2015 9th Int. Conf. Next Gener. Mob. Appl. Serv. Technol., no. September, pp. 393–398, 2016.

[48] Hewlett Packard, "The Internet of Things: Today and Tomorrow," Hewlett Packard, pp. 6–8, 2017.

[49] J. Thirumaran, "Internet of Things (IoT) and Machine-to-Machine (M2M) communications," pp. 5–10, 2015.

[50] K. J. Singh and D. S. Kapoor, "Create Your Own Internet of Things: A survey of IoT platforms.," IEEE Consum. Electron. Mag., vol. 6, no. 2, pp. 57–68, 2017.

[51] C. Levallois, "Definition of IoT and connected objects," 2017.

[52] T. K. Aseri, "Unit II : Sensors and Actuators," no. 01, pp. 1–10, 2018.

[53] B. Vogel, Y. Dong, B. Emruli, P. Davidsson, and R. Spalazzese, "What is an open IoT platform? Insights from a systematic mapping study," Futur. Internet, vol. 12, no. 4, pp. 1–19, 2020.

[54] BILAL KARAASLAN, "ThingSpeak Channels," 15/7/2013, 2013. [Online]. Available: https://community.thingspeak.com/tutorials/thingspeak-channels/. [Accessed: 19-Oct-2020].

[55] Mathworks, "Why would you want to collect data in ThingSpeak?" [Online]. Available: https://thingspeak.com/pages/how_to#act. [Accessed: 11-Nov-2019].

[56] M. A. Gómez Maureira, D. Oldenhof, and L. Teernstra, "ThingSpeak – an API and Web Service for the Internet of Things," World Wide Web, 2014.

[57] N. A, S. G, and M. K, "ThingSpeak Reviews & Product Details," IoT Anakytics Software, 2019. [Online]. Available: https://www.g2.com/products/thingspeak/reviews#survey-response-2735498. [Accessed: 15-Aug-2020].

[58] C. the Destroyer, "Three IoT Platforms for makers," 20th June 2017, 2017. [Online]. Available: https://www.sparkfun.com/news/2413. [Accessed: 17-Aug-2020].

[59] M. R. Thakur, “NodeMCU ESP8266 Communication Methods and Protocols - Programming with Arduino IDE,” 2018.

[60] “Learn More About ThingSpeak.” [Online]. Available: https://thingspeak.com/pages/learn_more?s_tid=Education. [Accessed: 11-Nov-2019].

[61] X. Mi, Y. Zhang, F. Qian, and X. Wang, “An empirical characterization of IFTTT: Ecosystem, usage, and performance,” Proc. ACM SIGCOMM Internet Meas. Conf. IMC, vol. Part F1319, no. December, pp. 398–404, 2017.

[62] S. Life and A. Instruction, Quick Guide of Using IFTTT to Control Smart Devices. 2017.

[63] K. Sundar Srinivas, S. Baig, T. Navya Sri, Y. Praneeth, and V. Naga Lakshmi, “Flood Monitoring System Using Thingspeak Web Server and Ifttt,” Int. J. Eng. Appl. Sci. Technol., vol. 5, no. 1, pp. 639–643, 2020, doi: 10.33564/ijeast.2020.v05i01.113.

[64] M. Fezari and A. Al Dahoud, “Integrated Development Environment ‘ IDE ’ For Arduino,” ResearchGate, no. October, pp. 1–12, 2018.

[65] Indoware, “Ultrasonic Ranging Module HC - SR04,” Datasheet, pp. 1–4, 2013.

[66] Elec Freaks, “HC SR04 Ultrasonic Ranging Sensor Module,” 2014.

APPENDICES

Appendix 1: Flood detector system code for Mpazi brook

```
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
const int analogPin = A0; // D0 or GPIO1616 of nodemcu
const int trigPin1 = 5;  //D1 Or GPIO5 of nodemcu
const int echoPin1 = 4;  //D2 Or GPIO4 of nodemcu
const int redled= 0;    //D3 or GPIO0 of nodemcu
const int yellowled = 2;   //D4 or GPIO2 pin of nodemcu
const int greenled = 14;   //D5 0r GPIO14pin of nodemcu
const int redled2 = 12;    //D6 or GPIO12 of nodemcu
const int blueled = 13;   //D7 or GPIO13 pin of nodemcu
const int BUZZER = 15;  //D8 or GPIO15 pin of nodemcu

unsigned long ch_no = 1100464;//Replace with Thingspeak Channel number
const char * write_api = "YXDDF5HHG46QD11D";//Replace with Thingspeak write API
char auth[] = "a119887011415506";
const char* ssid = "Mypc"; //Your Network SSID
const char* password = "florentine"; //Your Network Password

unsigned long startMillis;
unsigned long currentMillis;
const unsigned long period = 1000;
WiFiClient  client;
long duration1;
int Flood_level_Mpazi;
int Rain_intensity_Mpazi;
void setup()
{
  pinMode(trigPin1, OUTPUT);
```

```
  pinMode(echoPin1, INPUT);
  pinMode(analogPin, OUTPUT);
  pinMode(blueled, OUTPUT);
  digitalWrite(redled2, LOW);
  pinMode(redled, OUTPUT);
  pinMode(yellowled, OUTPUT);
  pinMode(greenled, OUTPUT);
  digitalWrite(redled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(yellowled, LOW);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
  startMillis = millis();  //initial start time
}
void loop()
{
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);
  duration1 = pulseIn(echoPin1, HIGH);
  Flood_level_Mpazi = duration1 * 0.034 / 2;
```

```
 Serial.print("Flood_level_Mpazi=");
 Serial.println(Flood_level_Mpazi);
 if (Flood_level_Mpazi <= 100)
 {
   digitalWrite(redled, HIGH);
   tone(BUZZER, 300);
   digitalWrite(greenled, LOW);
   digitalWrite(yellowled, LOW);
   delay(1000);
   noTone(BUZZER);
}
 else  if (Flood_level_Mpazi <= 200)
 {
   digitalWrite(redled, LOW);
   digitalWrite(yellowled, HIGH);
   digitalWrite(greenled, LOW);
 }
 else
 {
   digitalWrite(greenled, HIGH);
   digitalWrite(yellowled, LOW);
   digitalWrite(redled, LOW);
 }
 int Rain_intensity_Mpazi =analogRead(A0);
 Serial.print("Rain_intensity_Mpazi =");
 Serial.println(Rain_intensity_Mpazi );
if(Rain_intensity_Mpazi<= 300)
 {
   Serial.println("Heavy Rain");
     digitalWrite(blueled, LOW);
   digitalWrite(redled2, HIGH);
```

```
  }
  else if(Rain_intensity_Mpazi<= 500)
  {
      Serial.println("Moderate Rain");
     digitalWrite(blueled, LOW);
    digitalWrite(redled2, HIGH);
  }
else
  {
    Serial.println("No Rain");
    digitalWrite(blueled, HIGH);
    digitalWrite(redled2, LOW);

    delay(1000);
  }
  currentMillis = millis();
  if (currentMillis - startMillis >= period)
  {
    ThingSpeak.setField(1, Flood_level_Mpazi);
    ThingSpeak.setField(2, Rain_intensity_Mpazi);
    ThingSpeak.writeFields(ch_no, write_api);
    startMillis = currentMillis;
  }
}
```

Appendix 2: Flood detector system code for Rwampara brook

```
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
const int analogPin = A0; // D0 or GPIO1616 of nodemcu
const int trigPin1 = 5;  //D1 Or GPIO5 of nodemcu
const int echoPin1 = 4;  //D2 Or GPIO4 of nodemcu
const int redled= 0;    //D3 or GPIO0 of nodemcu
const int yellowled = 2;   //D4 or GPIO2 pin of nodemcu
const int greenled = 14;   //D5 0r GPIO14pin of nodemcu
const int redled2 = 12;    //D6 or GPIO12 of nodemcu
const int blueled = 13;   //D7 or GPIO13 pin of nodemcu
const int BUZZER = 15;  //D8 or GPIO15 pin of nodemcu
unsigned long ch_no = 1100464;//Replace with Thingspeak Channel number
const char * write_api = "YXDDF5HHG46QD11D";//Replace with Thingspeak write API
char auth[] = "a119887011415506";
const char* ssid = "Mypc"; //Your Network SSID
const char* password = "florentine"; //Your Network Password

unsigned long startMillis;
unsigned long currentMillis;
const unsigned long period = 1000;
WiFiClient  client;
long duration1;
int cm;
int Flood_level_Rwampara;
int Rain_intensity_Rwampara;
void setup()
{
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
```

```
  pinMode(analogPin, OUTPUT);
  pinMode(blueled, OUTPUT);
  digitalWrite(redled2, LOW);
  pinMode(redled, OUTPUT);
  pinMode(yellowled, OUTPUT);
  pinMode(greenled, OUTPUT);
  digitalWrite(redled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(yellowled, LOW);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
  startMillis = millis();  //initial start time
}
void loop()
{
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);
  duration1 = pulseIn(echoPin1, HIGH);
  Flood_level_Rwampara = duration1 * 0.034 / 2;
  Serial.print("Flood_level_Rwampara =");
```

```
 Serial.println(Flood_level_Rwampara);
 //Serial.print(cm);
 if (Flood_level_Rwampara <= 100)
 {
   digitalWrite(redled, HIGH);
   tone(BUZZER, 300);
   digitalWrite(greenled, LOW);
   digitalWrite(yellowled, LOW);
   delay(1000);
   noTone(BUZZER);
}
 else  if (Flood_level_Rwampara <= 200)
 {
   digitalWrite(redled, LOW);
   digitalWrite(yellowled, HIGH);
   digitalWrite(greenled, LOW);
 }
 else
 {
   digitalWrite(greenled, HIGH);
   digitalWrite(yellowled, LOW);
   digitalWrite(redled, LOW);
 }
  int Rain_intensity_Rwampara =analogRead(A0);
 Serial.print("Rain_intensity_Rwampara =");
 Serial.println(Rain_intensity_Rwampara );
If (Rain_intensity_Rwampara<= 300)
 {
  // Serial.println("Heavy Rain");
     digitalWrite(blueled, LOW);
   digitalWrite(redled2, HIGH);
```

```
 }
 else if(Rain_intensity_Rwampara<= 500)
 {
     //Serial.println("Moderate Rain");
    digitalWrite(blueled, LOW);
  digitalWrite(redled2, HIGH);
 }
else
 {
  // Serial.println("No Rain");
  digitalWrite(blueled, HIGH);
  digitalWrite(redled2, LOW);
  delay(1000);
 }
 currentMillis = millis();
 if (currentMillis - startMillis >= period)
 {
  ThingSpeak.setField(3, Flood_level_Rwampara);
  ThingSpeak.setField(4, Rain_intensity_Rwampara);
  ThingSpeak.writeFields(ch_no, write_api);
  startMillis = currentMillis;
 }
```

Appendix3: Xampp codes for monitoring information using localhost

```
<!DOCTYPE html>
<html>
<body>
<center><h2>IoT FLOOD MONITORING SYSTEM IN RWANDA</h2></center>
<h3>
<iframe width="450" height="260" style="border: 1px solid #cccccc;" src="https://thingspeak.com/channels/1100464/charts/1?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=60&type=line"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;" src="https://thingspeak.com/channels/1100464/charts/2?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;" src="https://thingspeak.com/channels/1100464/charts/3?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=60&type=line"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;" src="https://thingspeak.com/channels/1100464/charts/4?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;" src="https://thingspeak.com/apps/matlab_visualizations/359734"></iframe>

</h3>
<center>
<h1>
 <p><button onclick="myFunction()">Start or Stop Alarm</button></p>
<div id="myDIV">Alarm is OFF!</div>
```

```
<script>
function myFunction() {
  var x = document.getElementById("myDIV");
  if (x.innerHTML === "Alarm is OFF!") {
    x.innerHTML = "Alarm is ON!";
  } else {
    x.innerHTML = "Alarm is OFF!";
  }
}
</script></h1></center>
</body>
</html>a
```