

# **DEVELOPMENT OF DATA DRIVEN IOT ENABLED WATER QUALITY MONITORING FOR AQUACULTURE**

## **A PROJECT REPORT**

*Submitted in partial fulfillment of the  
requirement for the award of the  
Degree of*

**BACHELOR OF TECHNOLOGY  
IN  
ELECTRONICS AND COMMUNICATION ENGINEERING**

*by*

**PANNALA MANISH REDDY (17BEC1211)  
MEDISETTI ABHAYA HANUMA (17BEC1126)  
DODDA RAVI KUMAR (17BEC1115)**



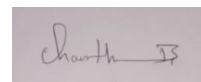
**VIT<sup>®</sup>**  
**Vellore Institute of Technology**  
(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF ELECTRONICS ENGINEERING  
VELLORE INSTITUTE OF TECHNOLOGY  
CHENNAI - 600127**

*May 2021*

## ***CERTIFICATE***

This is to certify that the Project work titled “*Development of data driven IOT enabled water quality monitoring for aquaculture*” that is being submitted by *Pannala Manish Reddy (17BEC1211)*, *Medisetti Abhaya Hanuma (17BEC1126)* and *Dodda Ravi kumar (17BEC1115)* is in partial fulfillment of the requirements for the award of **Bachelor of Technology in Electronics and Communication Engineering**, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.



**Dr. Chanthini Baskar**  
**Guide**

**The Project Report is satisfactory / unsatisfactory**

**Internal Examiner**

**External Examiner**

**Approved by**

**Head of the Department**  
B. Tech. (ECE)

**DEAN**  
School of Electronics Engineering

## ACKNOWLEDGEMENT

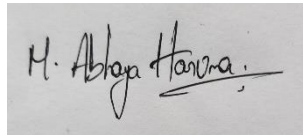
We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Chanthini Baskar**, Professor, School of Electronics and Communication Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the implementation of the project work.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

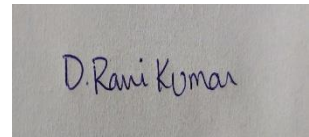
We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.



**Pannala Manish Reddy**  
(17BEC1211)



**Mediseti Abhaya Hanuma**  
(17BEC1126)



**Dodda Ravi Kumar**  
(17BEC1115)

## **ABSTRACT**

India's aquaculture industry has been growing at a rate of 6-7 percent per year. Large-scale agriculture and intensive farming, on the other hand, have resulted in a decline in aquaculture water quality and an increase in the incidence of aquatic animal diseases. As a result, maintaining water quality is essential for effective aquaculture management. Using IoT, the engineered device enables farmers to track the most critical physio-chemical factors of pond water in real time.

The Internet of Things (IoT) is a rapidly evolving technology that is now spreading its wings through all industries. With advancements in computers such as Arduino and Raspberry Pi, technology is reaching the ground level in agriculture and aqua-farming. We designed and executed aquafarm water-quality observation using Arduino UNO, different types of sensors, a webcam and Raspberry Pi in this project. Temperature, pH, and turbidity were used as water quality parameters in this study. Arduino is used to collect data from temperature and pH sensors, while the Raspberry Pi is used to process the data. The Raspberry Pi also performs photo acquisition with the help of a webcam in order to detect the turbidity of the water using image processing techniques. In addition, if these conditions go out of bounds, a fuzzy logic-based action will be taken, and a mail message will be sent to the end user. An internet and android app attached to the ThingSpeak cloud can also be used to track the water condition. The parameter readings are examined to establish the overall estimated status of water along with the necessary measures.

## TABLE OF CONTENTS

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ACKNOWLEDGEMENT</b>	3
	<b>ABSTRACT</b>	4
	<b>LIST OF FIGURES</b>	7
	<b>LIST OF TABLES</b>	8
	<b>LIST OF SYMBOLS AND ABBREVIATIONS</b>	9
<b>1</b>	<b>INTRODUCTION</b>	10
	1.1 Objectives	11
	1.2 Background and Literature Survey	11
	1.3 Need for IoT in Aquaculture	14
	1.4 Organization of the Report	14
<b>2</b>	<b>IOT ENABLED WATER QUALITY MONITORING FOR AQUACULTURE</b>	15
	2.1 Methodology	15
	2.2 Design Approach	16
	2.3 Standards	18
	2.3.1 Arduina as ADC	18
	2.3.2 Raspberry Pi as Processing Unit	19
	2.3.3 Image Processing	19
	2.4 Calibrations	20
	2.5 Constraints, Alternatives and Tradeoffs	21
	2.6 Required Hardware	21
	2.6.1 Raspberry Pi	21
	2.6.2 Arduino UNO	22
	2.6.3 Temperature Sensor	23
	2.6.4 pH Sensor	24
	2.6.5 Stepper Motor	24
	2.7 Technical Specifications	25

	2.8	Summary	26
<b>3</b>	<b>IMPLEMENTATION</b>		27
	3.1	Fuzzy Logic for Decision Making	27
		3.1.1 Architecture	28
		3.1.2 Algorithm	29
	3.2	Image Processing for Turbidity	29
	3.3	SMTP for Mail alerts	31
	3.4	Summary	31
<b>4</b>	<b>COST ANALYSIS</b>		32
<b>5</b>	<b>RESULTS AND DISCUSSIONS</b>		34
<b>6</b>	<b>CONCLUSION AND THE FUTURE WORK</b>		39
	6.1	Conclusion	39
	6.2	Future Work	39
<b>7</b>	<b>APPENDIX</b>		40
<b>8</b>	<b>REFERENCES</b>		49
	<b>BIODATA</b>		50
	<b>VIDEO DEMO</b>		51

## LIST OF FIGURES

<b>FIGURE No.</b>	<b>TITLE</b>	<b>PAGE No.</b>
2.1	Overall Block-diagram	15
2.2	Sensor Data Collection	16
2.3	Turbidity From the Image	17
2.4	Action Controller	17
2.5	Sensor Connections	18
2.6	Hardware Setup	20
2.7	Raspberry Pi	22
2.8	Arduino UNO	23
2.9	Temperature Sensor	23
2.10	pH Sensor	24
2.11	Stepper Motor	25
3.1	Fuzzy-Logic(FL) Possibilities	27
3.2	Fuzzy Architecture	28
3.3	Conventional logic vs Fuzzy logic	29
3.4	Laplacian Kernel	30
3.5	Increasing Turbidity with decreasing Variance	30
5.1	Temperature and pH readings	34
5.2	Graph between Average of pH by Time	34
5.3	Graph between Average of Temperature by Time	35
5.4	Key Influences	36
5.5	Cloud Data Collection	37
5.6	Mail alert received by the user	38

## LIST OF TABLES

<b>TABLE No.</b>	<b>TITLE</b>	<b>PAGE No.</b>
1.1	Literature Survey	12
3.2	Fuzzification Module types	28
4.1	Cost analysis	32



## LIST OF SYMBOLS AND ABBREVIATIONS

### LIST OF SYMBOLS

<b>SYMBOL</b>	<b>TITLE</b>	<b>PAGE NO.</b>
$\lambda$	Wavelength	13
k	Proportional constant	13
C	Celsius (Temperature Measurement)	20

### LIST OF ABBREVIATION

<b>ABBREVIATION</b>	<b>TITLE</b>	<b>PAGE NO.</b>
pH	Potential of Hydrogen	4
DO	Dissolved Oxygen	11
ADC	Analog to Digital Converter	15
FL	Fuzzy Logic	27
SMTP	Simple Mail Transfer Protocol	31

## **CHAPTER 1**

### **INTRODUCTION**

Aquaculture is amongst the most developing sectors in many nations throughout the globe, as the fish and aquatic animals demand is growing on daily basis. According to the United Nations Food and Agriculture Organization or commonly known UNFAO, the world yearly production of fishes and aquatic products is over 12.8 crore tons. Also, the average animal consumption per person is roughly 15%, and there is a growing reliance on fishery services. According to UNFAO, the total annual consumption of fish and aqua products is now 19 to 20 kilograms per person and is expected to be 16.7 kilograms in 2030. Fisheries are all linked with productivity, development and future food supply.

Aquaculture is the cultivation of marine plants and animals via a range of methods, resources, and techniques. This practice is extremely important for economic growth and food security. Continuous observation of pond water's physical, chemical, and biological parameters aids in not only predicting and controlling negative aquaculture environments, but also avoiding environmental degradation and the failure of the development process. Monitoring physical and chemical variables in water such as turbidity, temperature, and pH, is critical for aquaculture systems to function properly.

The rapid evolution of computer technology and smartphone devices for management and monitoring has been seen thanks to advancements in electronics, automated networks, microcontrollers, database technology, and telecommunications. The idea of the Internet of Things (IoT) was born in this age of mobile technologies and computer interconnectivity, and it consists of providing interconnectivity and contact with items. This offers a smart service through the use of the Internet and a sensor network. In layman's words, it's an interdisciplinary study that includes not only humans but also related objects.

The main aim of the project is to make use of fast growing technological service IoT and to develop a system which has the capability of monitoring the water quality parameters in real time and also have the ability to maintain those parametric values in their respective optimal range with the help of certain actions.

## **1.1 Objectives**

- To create a sensor network to monitor the various conditions of aqua farm. (Temperature and pH).
- To collect sensor values using Arduino.
- To use serial communication for transferring parameter values to Raspberry pi.
- To collect a picture of water through webcam connected to raspberry pi.
- To use image processing techniques to find the turbidity from the image.
- To send the parameters value to the ThingSpeak cloud.
- To implement a decision making system with fuzzy logic for action controller.
- To send mail alerts to the user whenever these parameters goes out of bound and whenever a specific action has been taken.

## **1.2 Background and Literature Survey**

A number of research documents in the literature review focus on how aquatic life affects progress in water quality and how IoT is used to solve the problem. A lot of research is done with IoT to address these types of challenges, since IoT lately reaches the level with its applicability to all type's farmers around the globe. Many studies concentrate on a few kinds of sensors, such as pH, DO and turbidity, etc. and solve them. However, optimal fish production is subject to the great majority of the chemical, physical and biological features of water Effective maintenance of the pond therefore demands the quality of water. Factors like dissolved oxygen (DO), temperature, disturbances, clearness, pH, carbon dioxide, alkalinity, hardness, conductivity, salinity, TDS, synthesized ammonia, nitrite, main productive activity, plankton population, BOD, etc. are determined for water quality.

The following are the few research papers that we have referenced to acquire overall picture of use of latest technologies in aqua cultures and also the image processing techniques that comes in handy while determining the quantities like turbidity whose value can be obtained from the visual information gathered and processed.

Sl.No	Name of article	Name of journal	Parameters concerned
1	Design of water quality monitoring system for aquaculture ponds based on NB-IoT.	Elsevier Aquacultural Engineering 90 (2020) 102088	Temperature PH DO
2	Turbidity detection using Image Processing	ICCCA-IEEE Publication (2016)	Turbidity
3	Development of a cloud-based IoT monitoring system for Fish metabolism and activity in aquaponics	Elsevier Aquacultural Engineering 90 (2020) 102067	PH DO
4	Using a Digital Camera Combined With Fitting Algorithm and T-S Fuzzy Neural Network to Determine the Turbidity in Water	IEEE Publication (2019)	Turbidity
5	Design and operation of a low-cost and compact autonomous buoy system for use in coastal aquaculture and water quality monitoring	Elsevier Aquacultural Engineering 80 (2018) 28–36	Temperature Salinity PH

**Table 1.1 Literature Survey**

Various researchers have published many papers about the usage of latest technological trends like IoT in aquaculture. After going through some of those papers the following key points have been noted:

- Physical, Chemical and Biological Parameters (Temperature, PH and Turbidity) are important for Good Profits and protein rich aqua life.
- Aqua animals' activity or movements can cause a slight increase in the Temperature and those movements of the aqua animals is a sign that they are hungry. Temperature can also change depending on surrounding environment.
- Optimal Temperature range = 26 – 33C

- Optimal PH range = 7.5 – 8.5.
- The Turbidity of the water in aqua culture mainly occurs because of the food that we supply for the fish. The uneaten food particles after some time become the main reason for turbidity.
- Finding turbidity of the water by image processing can be a good alternative method for actual laboratory process because of its cost effectiveness.
- Using decision making algorithm like Fuzzy can help in getting better results when compared to normal conventional methods.

The Temperature, pH and turbidity are the very important factors to monitor the water quality of the pond. The temperature and pH can be easily determined with the help of physical sensors. But for turbidity we have used image processing technique to determine from an image. Turbidity is defined as a decrease in the transparent property of liquid induced by the occurrence of non-soluble materials, and it is a direct indication of water quality. It is also frequently defined sediment-logically as an extent of tiny particle material with a significantly longer deferment time in water-column, separating it from coarser sediment settle in both physically and chemically. Currently, all the turbidity measuring techniques relies completely on the electro-optic approaches, such as visual-turbidimetry, detecting approach that grounded on broadcasted and the reflected light, and a detecting approach build on the ratio of transmission and scattering.

Rayleigh formula for the link amongst intensity of light after dispersion and particulate content per-unit of liquid content:  $I_s = k I_o n^2 / \lambda^4$  where  $I_s$  - scattering intensity of light,  $k$  – proportionality value,  $I_o$  – intensity of the incident light,  $n$  – total no of particles per sample,  $V$  - volume of the sample, and  $\lambda$  – wavelength of incident light. This is the turbidimeter's traditional approach. We'll use similar properties of light but a different approach to find turbidity from an image in this as shown in chapter 3.

### **1.3 Need of IoT in Aquaculture**

Aquaculture is a significant economic activity and a thriving sector in India, with a wide range of resources and potential. The sector's dynamism is exemplified by India's 11-fold rise in fish production over the last six decades. Freshwater aquaculture accounts for more than 90% of

all aquaculture output. New technologies such as the Internet of Things (IoT), which aid computation, communication, and control within devices, will be used to continue the trend and increase output. We can monitor various parameters within the aquafarm, such as temperature, pH, and turbidity, using a smart decision-making algorithm. This will assist farmers in ensuring a higher crop yield along with giving the ability to monitor entire farm or farms with less human interaction.

## **1.4 Organization of the Report**

The overall organization project report is described as follows:

- Chapter 1 contains the introduction, literature survey, the need of IoT in aquaculture and organization of report.
- Chapter 2 contains the methodology, design approach, standards used and constraints of the methodology used in the project.
- Chapter 3 contains the implementation of precision agriculture with server node algorithm and end node algorithm. It also explains the communication between end node and server node.
- Chapter 4 gives the cost involved in the implementation of the project and analyses its economic feasibility.
- Chapter 5 compiles the various results obtained after the project was successfully implemented.
- Chapter 6 concludes the report with discussions about the results obtained and their future implications.

## CHAPTER 2

### IOT ENABLED WATER QUALITY MONITORING FOR AQUACULTURE

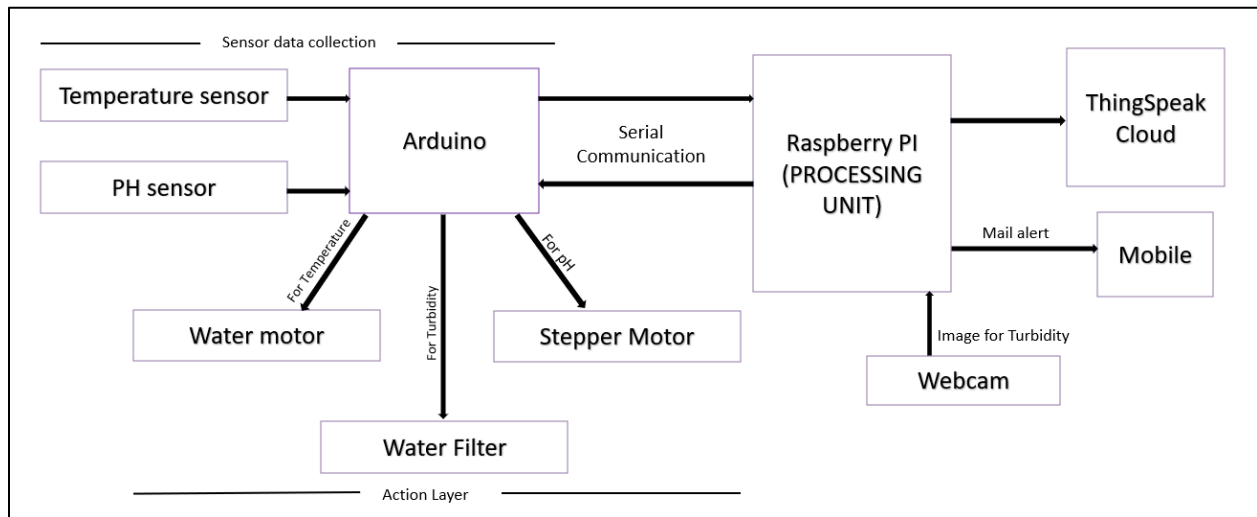
This Chapter describes the methodology, design, standards, calibrations, sampling rate and constraints of IoT enabled water quality monitoring for aquaculture.

#### 2.1 Methodology

The implementation idea of the project is been divided into three categories:

1. Sensor data collection
2. Image processing for turbidity
3. Fuzzy logic for decision making in action control.

We set up a small aqua farm in an aquarium for the project demonstration and connected various sensors for measuring temperature and pH through Arduino. The Arduino functions as an ADC, converting the analogue data from the sensors to digital and sending the same digital data to the Raspberry Pi.



**Figure 2.1 Overall Block Diagram**

The picture of the setup was also captured using a webcam connected to the Raspberry Pi. Turbidity is discovered using image processing techniques. As a result, we can obtain all three

parameters (Temperature, pH and Turbidity). At regular intervals, these values will be transmitted to the ThingSpeak cloud. And this data can be accessed at any time and from any place with internet.

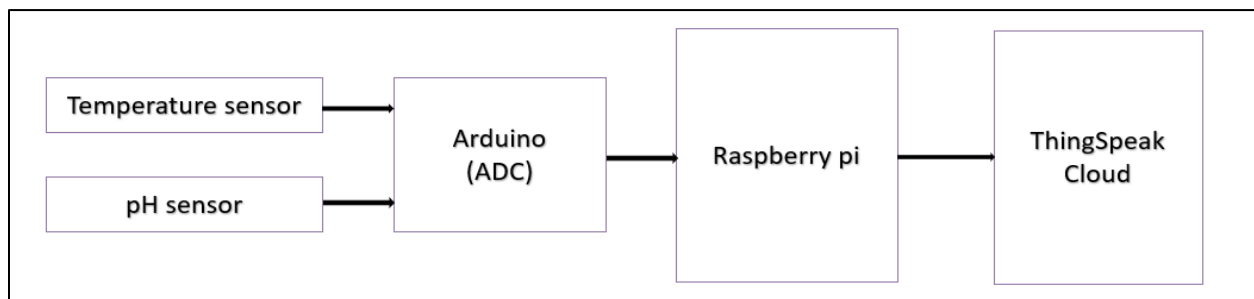
When the parameters cross their threshold limits, the following steps will be taken based on Fuzzy decision making logic, as well as sending an email warning to the end user.

- When Temperature value is out of bound some food and cool water will be sent to the aqua farm.
- When pH is out of bound then based on the pH-value, basic or acidic materials will be added.
- When turbidity is out of bound then a water filter will be started to clean the water.

## 2.2 Design Approach

The approach we have used to implement the project is being explained here.

The Arduino is interfaced with a water resistant temperature sensor – DS18B20 and pH sensor, which send analogue sensor data. This will be translated to digital data using the Arduino's built-in ADC. This digital information will be transmitted to the Raspberry Pi through serial communication via a USB cable. These sensor values will now be sent to the cloud, where they can be monitored from anywhere in the world with an internet connection. To store the data that we collected from the aqua farm, we used ThingSpeak as a cloud server.

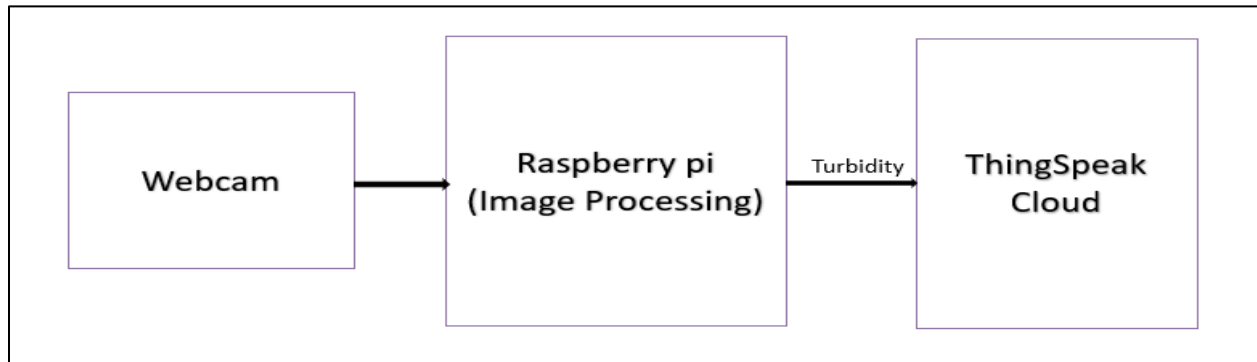


**Figure 2.2 Sensor data collection**

ThingSpeak is an open IoT online data surveillance tool. One can set data to private or public data on ThingSpeak channels, depending on the preference. ThingSpeak takes minimum of 15 seconds to update your readings. The framework for creating IOT projects is great and very easy to use. There are also some systems for tracking your data online, but ThingSpeak offers



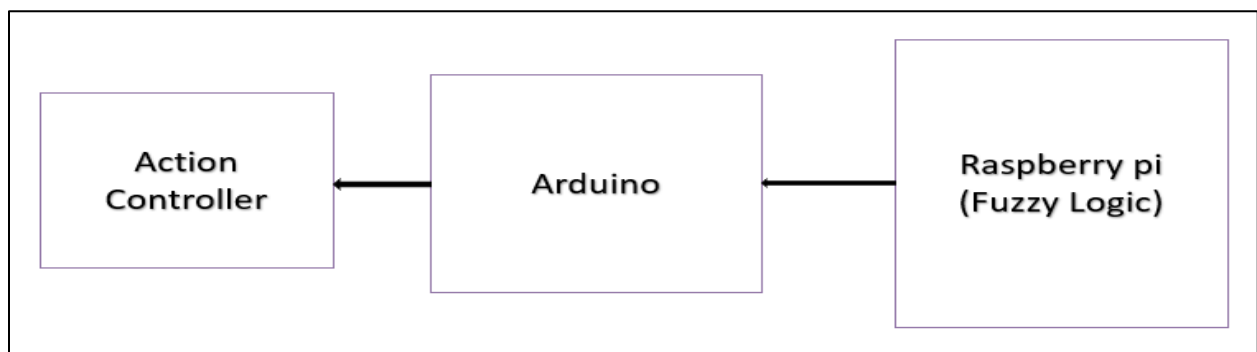
advantages over other platforms, such as very simple configuration and many graph-making choices.



**Figure 2.3 Turbidity from the image**

A webcam has been incorporated with the raspberry pi to take a picture of the rig, which will be analysed using image processing techniques to determine water turbidity. This turbidity is also sent to ThingSpeak, which stores the data and monitors it.

If any of the parameters falls outside of their optimal range, a corresponding action will be taken to put those values back into the optimal range, and an email warning will be sent to the end user via Simple Mail Transfer Protocol (SMTP). Fuzzy logic has been used for decision making in the action part to improve action efficiency and to have human-like thinking capacity when making decisions. The action controller includes actions and equipment such as a water motor to inlet cool water when the temperature is high and a stepper motor to regulate the lid to inlet acidic or basic materials when the pH is abnormal. When the turbidity is high, the filter motor is activated.



**Figure 2.4 Action controller**

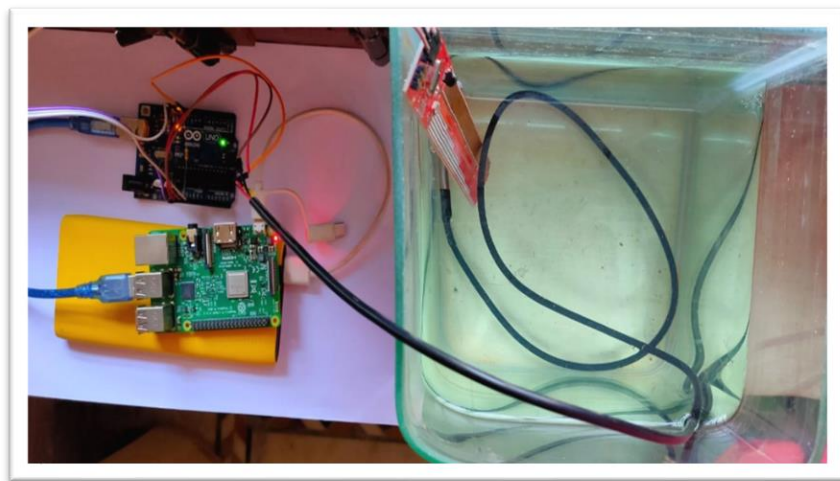
## 2.3 Standards

### 2.3.1 Arduino as ADC

The Arduino can both input and output analogue and digital signals. An analogue signal can have any number of values, as opposed to a digital signal, which only has two: HIGH and LOW. The Arduino includes an analog-to-digital converter for measuring the value of analogue signals (ADC). The ADC converts an analogue voltage to a digital value. The `analogRead` function is used to obtain the value of an analogue signal (pin). This feature transforms the voltage on an analogue input pin to a digital value ranging from 0 to 1023 in relation to the reference value. The default reference voltage is five volts (for five volt Arduinos) or 3.3V (for 3.3volt Arduinos). It has one parameter which is the pin number.

The Atmega328P microcontroller with a 10 bit ADC is used in the Arduino UNO. The IC includes six such 10-bit ADCs (pin 23 to 28). This means that each ADC should count at least  $5V(\text{Maximum logic level})/2^{10} = 4.88\text{mV}$ . It has a maximum of 1024 steps. The Atmel2560IC, which has 16 10bit ADCs, is used in the Arduino Mega. As a result, accuracy remains unchanged. The greater the bit number of an ADC, the greater its accuracy.

The Arduino board is also used to control the water filter and stepper motor. After a specific decision has been taken using fuzzy logic, a signal will be sent to the appropriate action part to neutralise the farm's parameters.



**Figure 2.5 Sensor Connections**

### **2.3.2 Raspberry Pi as Processing Unit**

In our work, the Raspberry Pi serves as the central processing unit. After being converted to digital format by Arduino, all sensor data is delivered through serial communication through a USB connection to Raspberry Pi. In addition, rather than explicitly using the sensor, we are employing image processing techniques to determine turbidity. Much of this will take place on the Raspberry Pi.

Since the data is stored on the cloud server ThingSpeak, there is no need for the Raspberry pi to have higher configuration. So we went with the standard model R pi 3B with 3GB of RAM and 4GB of data (for OS). A camera is built into the controller to capture an image of the setup in order to determine turbidity. The turbidity of the image will be discovered after processing. All three parameters discovered so far will be sent to the ThingSpeak cloud. ThingSpeak is an open IoT forum for online data tracking. In the ThingSpeak channel, you can choose whether to make the data private or public. It also gives you some choices for plotting the graph. From ThingSpeak anyone can monitor parameters from anywhere through internet.

If any of the parameters deviates from their optimum range, a corresponding action based on fuzzy logic decision making will be taken. This decision-making algorithm also runs on the Raspberry Pi and sends data to the Arduino serially. The Arduino then sends the signal to the appropriate action parts once more.

### **2.3.3 Image Processing**

The Raspberry Pi is used for the image processing. The picture is acquired by the connected webcam, utilised to calculate turbidity by the image processing technology.

To calculate the variance, we used the Laplacian kernel. In this case, the variance serves as a score to determine turbidity. The Laplacian is frequently employed for image edge-detection. We simply convolute an image with a Laplacian-kernel and later compute the variation of answer. This variation found is used to calculate the amount of blur in image. If there is a very little variability, then the response is extremely small, which means the image has hardly any edges. As we know, the more fluid an image, the fewer edges. We shall maintain a threshold for the variance such that the turbidity is determined. This limiting value or the threshold acts as a barrier value to keep turbidity neutral.



**Figure 2.6 Hardware setup**

## **2.4 Calibrations**

1. Temperature : The optimal temperature range for the aqua farm is 26 to 33C. The temperature sensor DS18B20 has range of -55 to +125C with +0.5C accuracy.
  - If temperature goes below 26C hot water will be added and if temperature goes above 33C hot water will be added based on fuzzy logic
2. pH : The optimal pH range for the aqua farm is 7.5 to 8.5.
  - If pH goes below 7.5 basic material will be added and if it goes above 8.5 acidic material will be added.
3. Turbidity : We are using the variance as score for turbidity. And if variance less than 500 then that is a turbid water.
  - If variance score goes below 500, the motored filter will be turned on to filter out the turbid material.

## 2.5 Constraints, Alternatives and Trade offs

- **Constraint** : Exact turbidity value cannot be found. Since we are using image processing techniques to find the turbidity, we won't be able to tell the exact turbidity value

**Alternative** : We are using variance as a score to measure turbidity. If variance goes below 500 then water is turbid.

**Trade Off** : We are confined to say whether the water is turbid or non-turbid and not the actual turbidity value.

- **Constraint** : Water parameters like Dissolved oxygen and salinity are not measured.

**Alternative** : In order to keep the project cost effective, we reduced the number of parameters.

**Trade Off** : We are confined to 3 important parameters Temperature, pH and turbidity.

- **Project Deployment** :

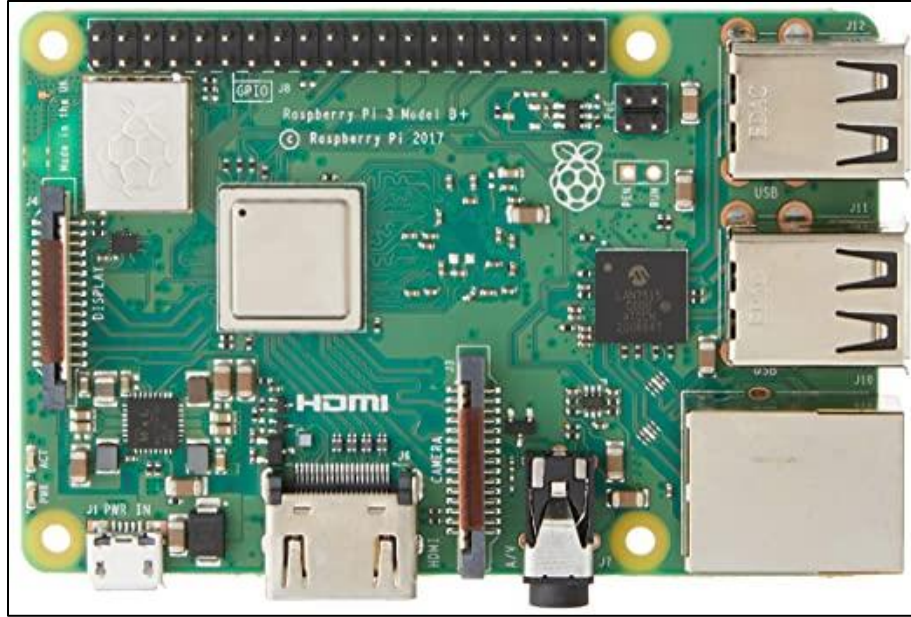
Since this aspect is dependent on the geographical conditions present in the real-time environment, we assumed for our project that the conditions are nearly constant/uniform across the entire aqua farm.

## 2.6 REQUIRED HARDWARE

### 2.6.1 Raspberry pi

In our project, we have used Raspberry Pi3b model with 2gb of Ram. Also, we are supposed to include the OS with the help of 8gb memory card.

Raspberry Pi3b, the core system, is the primary processing unit of this work. The Raspberry Pi3(as shown in figure 2.7) is a small, low-cost computer board that runs Noobs, a Debian-based version of the Linux operating system. It boasts a faster processor and more CPU cores than previous Raspberry Pi models. It already has Wi-Fi and Bluetooth built in. Because it is a little computer, the Raspberry Pi can communicate serially with the Arduino.



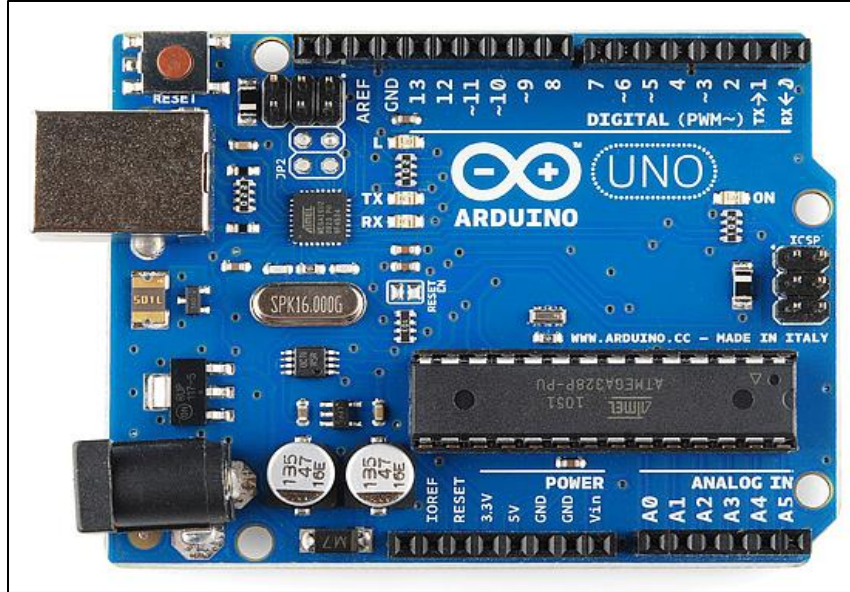
**Figure 2.7 Raspberry Pi**

### **2.6.2 Arduino UNO**

Since the sensors we use cannot be connected directly to the processing unit which is raspberry pi which takes digital input, we need an analog to digital converter and for that purpose we selected Arduino instead of regular ADC because of taking wireless data transmission which is future work of our project which can be achieved easily with Arduino UNO. We utilize Arduino UNO for the acquisition of the sensor data. Arduino Uno (seen Fig.2.8) is our Arduino version which is being used in our work.

Arduino Uno is an ATmega328P-based microcontroller board. It features 6 input analogue and 14 input/output digital pins. The voltage is 5V and the input voltage is suggested at 7-12V. The programming of Arduino IDE is necessary. To be programmed using a standard USB-conversion device, Arduino Uno should be linked to the USB cable computer. It can communicate data using serial connection with your computer and in our case, it is Raspberry Pi.

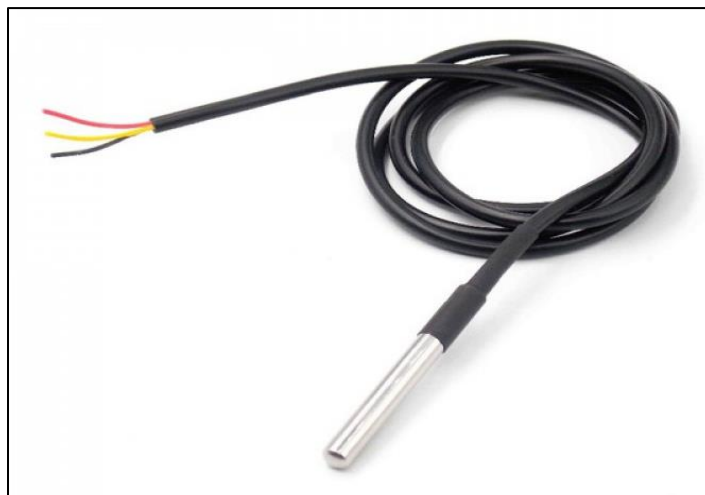
Along with the regular serial communication to transfer the sensor data, we can also use wireless transmission with the help of Bluetooth, Zigbee or any other wireless transmission modules to transmit the data to processing unit Raspberry Pi.



**Figure 2.8 Arduino UNO**

### 2.6.3 Temperature Sensor (DS18B20)

For the acquisition of temperature values from the aqua farm, we have utilized the waterproof DS18B20 Arduino (DFR0198) temperature sensor (as seen in Figure 2.9) . The precision is between  $\pm 0.5^{\circ}\text{C}$  and  $-10^{\circ}\text{C}$  and  $+85^{\circ}\text{C}$ . For this sensor, the upside is only necessary for many sensors to communicate one pin. One Arduino wire library is used for temperature measurement with this sensor.



**Figure 2.9 Temperature Sensor**



#### 2.6.4 pH Sensor (SEN0161)

In this work for collecting the pH values of aqua pond, pH water sensor for Arduino (SEN0161) is utilized (as indicated in Figure 2.10). This pH sensor has an integrated, simple connection and features, and is especially suited for Arduino.

The sensor is to be connected to Arduino using a BNC connection. This sensor has a range of 0-14 ph. It is accurate at 25oC and the working temperature is 0-60°C. It has a pH-accuracy of  $\pm 0.1$ . There are only a few parts in the water of the sensor. When water is pure and one month for highly turbid water, the dependability of the pH sensor can endure for half a year.



**Figure 2.10 pH Sensor**

#### 2.6.5 Stepper Motor

For controlling the lid in the action part for pH and food supply, we have used a regular stepper motor. This moves the lid by rotating in specific degree of angle based on input.

An electromechanical unit is a stepper motor that turns electrical power into mechanical energy. It's a synchronous, brushless electric engine, which can split the entire rotation into a large number of stages. Without the feedback system, the position of the motor may be properly controlled as long as the motor is sizeable for the application. Stepper engines are similar to the



reticence engines flipped. The stepper motor employs magnet function theory to turn the motor shaft precisely when an electric pulse is given. There are eight pole sizes and six poles on the rotor. To move the 24 steps to accomplish one complete rotation, the rotor needs 24 pulses of power. Another way to state that is that for every pulse of electricity received by the engine, the rotor moves precisely 15°.



**Figure 2.11 Stepper Motor**

## **2.7 TECHNICAL SPECIFICATIONS**

The following are the technical specifications and use testcases of all the hardware that we have used in our work.

- Temperature sensor (DS18B20) : The DS18B20 waterproof temperature sensor is reasonably precise and does not need any external components to work. It has a temperature range of -55C to +125C and with precision of +-0.5C. The resolve can be set to 9, 10, 11, or 12 bits. However, at start-up, the resolution is set to 12bit. (i.e. 0.0625C precision).
- pH sensor : The pH sensor is built on an Atmega8 IC with an on-board LDR for light and a Thermistor for temperature (0 to 60C). It has a probe length of 21cm and a measurement scale of 0 – 14pH.

- Arduino UNO : Acts as an ADC for sensor network, for serial communication and controller for the action portion.
- Raspberry pi : Works as central Processing unit.
- Webcam : A webcam with a resolution of 5MP was used to capture the image.
- Stepper Motor : 5Volts stepper motor(28BYJ48) is small, low-cost and high quality geared. It has reduction rate 64:1 with speed of about 15rpm and frequency 100Hz.
- Power Bank : To supply power to all the components.
- Wi-Fi router : To provide internet access to the Raspberry pi.
- Jumper wires and USB cables : For connection purposes.
- Waterproof Motor : For supplying water and to filter turbidity from the water when temperature is abnormal or turbidity is high.

## **2.8 Summary**

This chapter thus addressed the methodology, design approach, requirements applied, technical specifications, and project constraints. Chapter 3 discusses about the complete implementation of Fuzzy Logic and image processing.

## **CHAPTER 3**

### **IMPLEMENTATION**

This chapter describes the implementation of the Fuzzy logic for decision making, Image processing for turbidity and SMTP for mail alerts.

#### **3.1 Fuzzy Logic for decision making**

After collecting and sending the parameters to the cloud, they will be compared to their optimum values. This distinction is made with the aid of fuzzy logic. This Fuzzy logic is only applicable to temperature and pH, not turbidity, since we are using image processing to determine turbidity and therefore do not have the real turbidity value, so we cannot use Fuzzy logic for the turbidity action portion.

Fuzzy Logic is a mode of understanding associated with human like reasoning, also called FL. The solution from FL system is like people's decision-making and contains all of the intermediary options amongst YES or NO digitals.

A typical logic-block can be understood by a computer accepts accurate input and produces certain results like 0 or 1 that are human like YES/NO.

Lotfi Zadeh, the creator of fuzzy logic, found that, unlike machines, A multitude of options amongst YES or NO are necessary for human like decision making, including as –

DEFINITELY YES
POSSIBLY YES
CANNOT SAY
POSSIBLY NO
DEFINITELY NO

**FIGURE 3.1 Fuzzy-Logic(FL) Possibilities**

Fuzzy-logic works at various input levels to reach a defined result. Fuzzy logic has many industrial and practical applications.

- It is capable of controlling machinery and also daily user goods.
- It might not give precise rationale, but it does supply enough understanding.
- It aids in dealing with the complexity present in technology.

### 3.1.1 Architecture

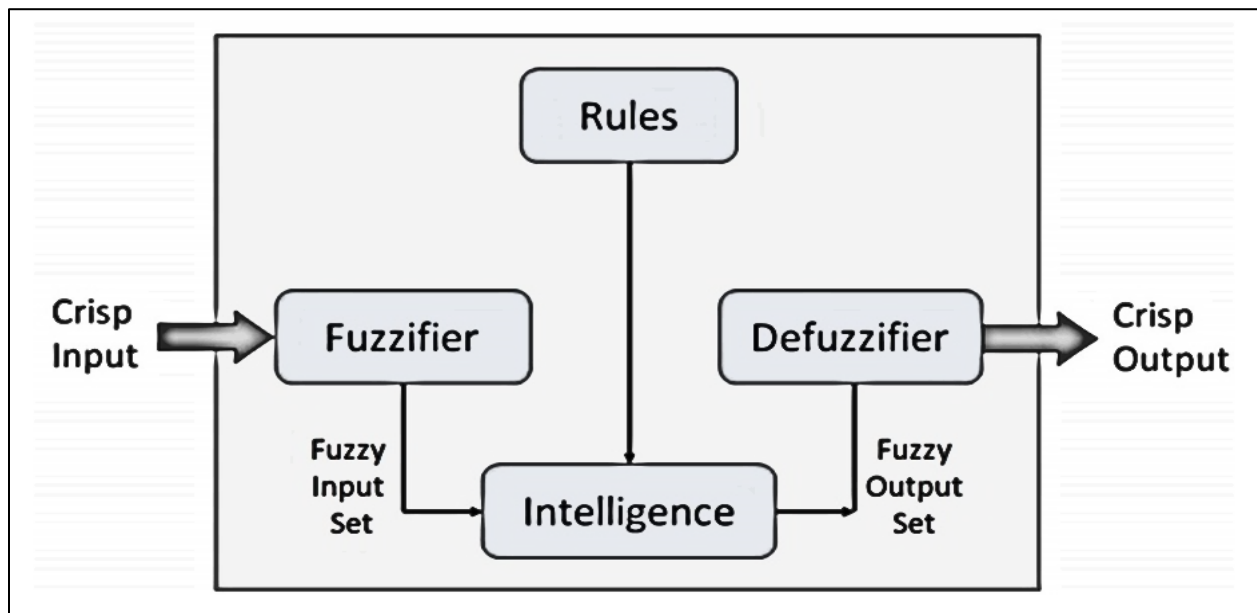
FL architecture is divided into four major sections, as illustrated –

- **Fuzzification Unit** – This converts the platform's sharp numerical sources to fuzzy-sets. It separates the source signal into 5 stages listed below :

<b>LP</b>	“Large-Positive”
<b>MP</b>	“Medium-Positive”
<b>S</b>	“Small”
<b>MN</b>	“Medium-Negative”
<b>LN</b>	“Large-Negative”

**Table 3.2 Fuzzification Module types**

- **Rule Base** – This includes all of the rules and if-then criteria proposed by specialists to regulate the decision making mechanism.
- **Inference-Engine** – This mimic human thinking by using the inference of FL and IF-THEN rules to emulate the human-reasoning processes.
- **Defuzzification Unit** – This transforms the inference engine's fuzzy-set into a crisp value.

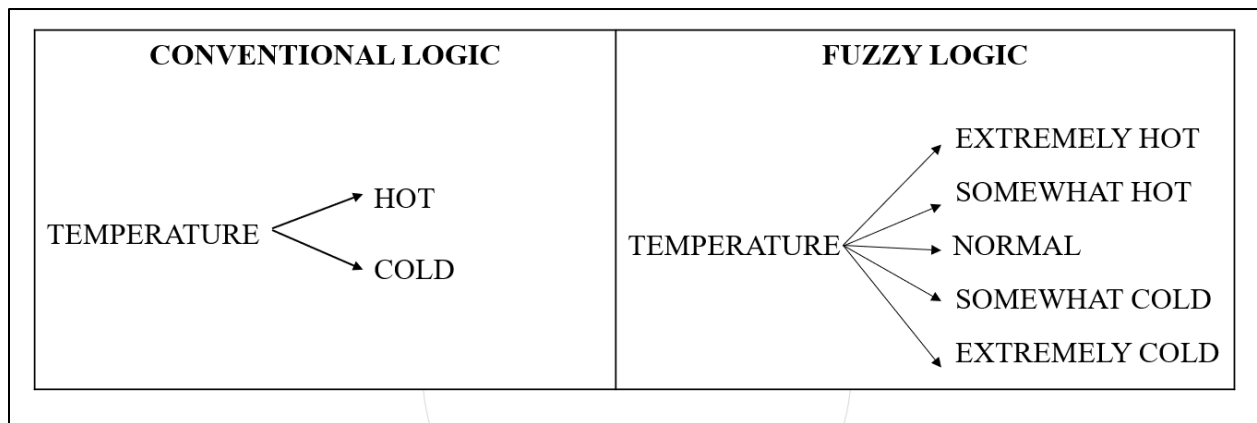


**FIGURE 3.2 Fuzzy Architecture**

### 3.1.2 Algorithm

The following is the algorithm for developing fuzzy logic:

- Start - Identify and specify certain values or phrases as inputs.
- Create the respective membership functions to the defined values.
- Rule Base - Create rules to regulate the decision making mechanism.
- Fuzzifier - With the help of membership functions, recast input data to fuzzy-sets.
- Inference-Engine - Examine with the rules which are predefined.
- Interfacing the outcomes of every rule to get certain outcome.
- Defuzzifier - Translate the final outcomes to normal crisp values.



**Figure 3.3 Conventional logic vs Fuzzy Logic**

### 3.2 Image Processing For Turbidity

Raspberry Pi is used for performing Image processing. The picture is captured by the webcam connected to the R pi, and some image-processing methodologies are used for determine turbidity. The image processing will work as follows:

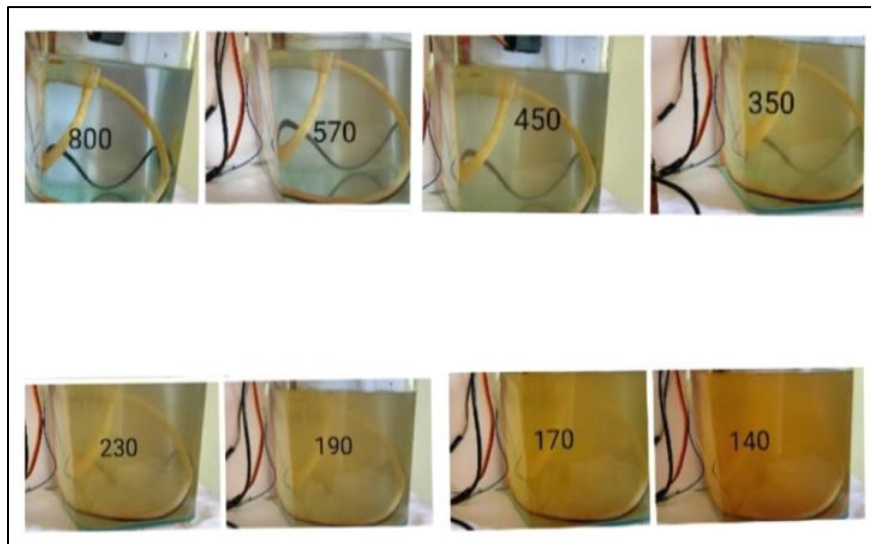
- To calculate the variance, we used the Laplacian-kernel. In this case, the variance serves as a score to determine the turbidity.
- The Laplacian is frequently employed for image edge detection. We just take an image, convolute with the Laplacian-kernel, and later calculate variation of answer.
- The variance thus found is utilized to determine the amount of blur in image.

- If the variation is minimal, the output responses spread will be small, showing that the image has extremely small edges. The higher a picture gets blurry, the less corners it has.
- We will be keeping a threshold for the variance score so that it helps in determining the turbidity.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

**Figure 3.4 Laplacian Kernel**

This way of approach works because of the idea of the Laplacian operator, which is used to calculate second derivative of picture. The Laplacian operator, highlights the regions of an image with rapid intensity changes. The Laplacian is used to detect edges. The premise here is if a picture has a large variation, then there will be wide range of responses that are reflective of a normal, in-focus image. However, if the variance is low, the response distribution is very tiny, implying that the image contains hardly any edges. And per common notion, if there are a smaller number of edges high is the blurriness of picture.



**Figure 3.5 Increasing Turbidity with decreasing variance**

### **3.3 SMTP for Mail Alerts**

A mail warning will be sent to the user if the parameters are out of their optimum range, along with the corresponding action. Simple Mail Transfer Protocol or the SMTP protocol has been used to deliver email alerts. SMTP is the core protocol to provide emailing over TCP system. One can use the protocol to send and receive emails.

SMTP is an application layer protocol used to send and transfer email over the Internet. Its upkeep is the responsibility of the IETF - Internet Engineering Task Force. SMTP has been commonly summarized with an email and client application and consists of 4 major components as described below:

1. Local user or client-end utility known as the MUA - Mail User Agent.
2. Server known as MSA - Mail Submission Agent.
3. MTA - Mail Transfer Agent.
4. MDA - Mail Delivery Agent.

Using the raspberry pi python software to receive email updates or a collection of data is a very useful application. In the python script, all we need is the smtplib library. Python comes in several versions; however the Raspberry Pi works best with the 3.2 and 2.7 versions.

### **3.4 Summary**

This chapter describes the implementation of Fuzzy Logic for decision making, Image Processing for Turbidity, and the Simple Mail Transfer Protocol (SMTP). The cost analysis of the project's various components is given in the following chapter.

## **CHAPTER 4**

### **COST ANALYSIS**

#### **4.1 List of components and their cost**

The detailed cost analysis of each and every component we have used in our project have been listed below in the table 4.1.

**Table 4.1 Cost Analysis**

<b>COMPONENT</b>	<b>COST</b>
Raspberry Pi	Rs 3000
Arduino	Rs 500
Temperature Sensor	Rs 150
pH Sensor	Rs 600
Webcam	Rs 500
Water pump (2 no.s)	Rs 300
Stepper motor	Rs 100
Cables and Jumper wire	Rs 100
<b>TOTAL</b>	<b>Rs 5250</b>



## **4.2 Cost Effectiveness**

To keep the project cost efficient, we did not consider a few water parameters such as dissolved oxygen, salinity for which sensors are very costly. In addition, to minimize sensor costs, we created an image processing technique for determining turbidity, which is one of the selected parameters. As this element depends on the geographical conditions present in the real time environment, for our project we have assumed that the conditions are nearly constant for the entire aqua farm.

If we have to deploy for a very large pond, since we are using Arduino for sensor data collection, we can deploy another Arduino along with sensor network connection and by using the wireless transmission technique to transfer the sensor data to the Raspberry pi. Thus, without using two entire setups, we can install Arduino as nodes and thereby reducing the total cost value irrespective of the pond.

## CHAPTER 5

### RESULTS AND DISCUSSIONS

#### 5.1 Sensor Readings

The temperature and pH sensor data were successfully collected at the cloud server, as shown in the figure below.

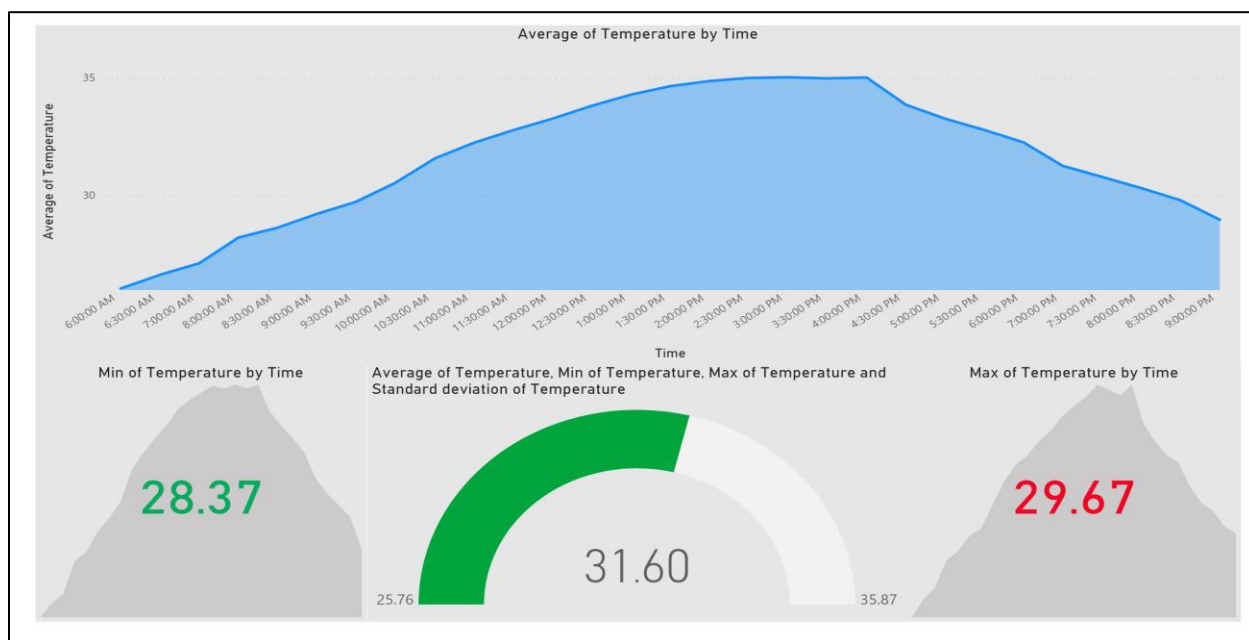
Day	Time	Temperature	pH
25-Mar	10:00	30.79	7.94
26-Mar	10:00	30.89	7.94
27-Mar	10:00	30.64	7.96
28-Mar	10:00	30.46	7.98
29-Mar	10:00	30.51	7.99
30-Mar	10:00	30.54	7.97
31-Mar	10:00	30.34	7.99
1-Apr	10:00	30.48	7.99
2-Apr	10:00	30.26	7.98
3-Apr	10:00	30.57	7.99
4-Apr	10:00	30.49	7.99
5-Apr	10:00	30.57	8.02
6-Apr	10:00	30.46	8.01
7-Apr	10:00	30.13	8
8-Apr	10:00	30.49	8.01
9-Apr	10:00	30.46	8.03
10-Apr	10:00	30.82	8.03
25-Mar	10:30	31.54	8.19
26-Mar	10:30	31.46	8.2
27-Mar	10:30	31.67	8.2
28-Mar	10:30	31.64	8.21
29-Mar	10:30	31.51	8.21
30-Mar	10:30	31.46	8.22
31-Mar	10:30	31.36	8.22
1-Apr	10:30	31.76	8.22
2-Apr	10:30	31.46	8.24
3-Apr	10:30	31.52	8.25
4-Apr	10:30	31.49	8.26
5-Apr	10:30	31.53	8.27

**Figure 5.1 Temperature and pH readings**

#### 5.2 Analytics from the Sensor Readings



**Figure 5.2 Graph between Average of pH by Time**

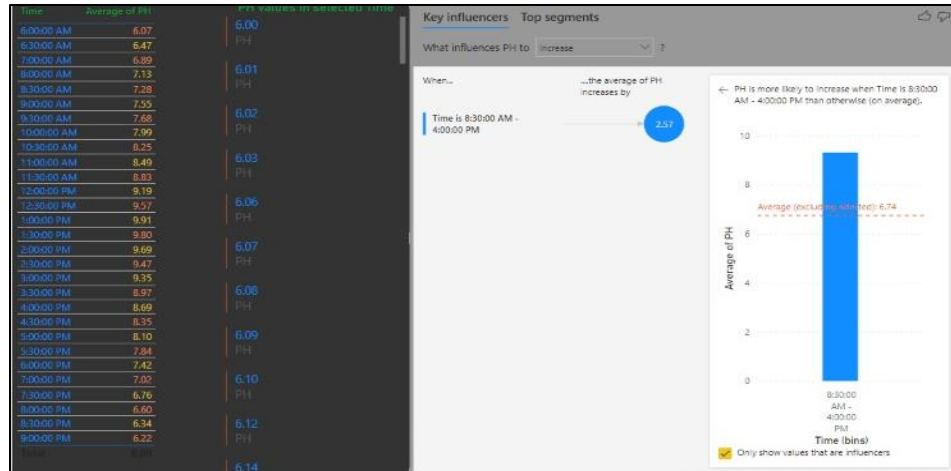


**Figure 5.3 Graph between Average of Temperature by Time**

The above graphs have been plotted by the data we have collected over a time period of 15 days. Following key points have been noted from the data:

- During the period the average pH value is 8.00.
- The average temperature value is 31.60C.
- Both the temperature and pH are maximum at the time of afternoons and minimum at mornings and nights.
- For temperature, special care has to be taken at afternoon between 1:30PM – 4:30PM.
- For pH, special care has to be taken at all times because, at mornings and nights the value is less than the optimal minimum and at afternoons the value is higher than the optimal minimum.

Climate change has the potential to change the data. The information presented above was derived from data obtained between the 25th of March and the 10th of April. Simply put, the points will be applicable only during the summer season and may differ under other circumstances. Furthermore, in a real-life pond, pH and temperature can vary depending on a variety of factors such as aquatic plants and animals, microorganisms and fungi materials present.



(a)

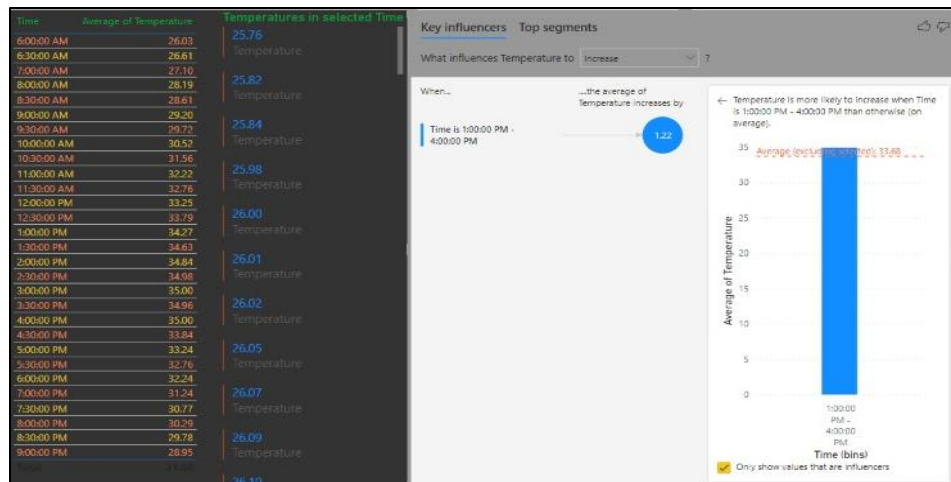


Figure 5.4 (a),(b) Key Influences

Visualizations of the data obtained aided in the discovery of several insights.

Insights into temperature and pH-

- pH, a factor that is important in the project. The recommended temperature range for the finny tribes is 6-8 degrees Celsius. Two bars in the graph show that there is a divergence from ideal circumstances.
- Temperature, another component included in the project, visual insights show that between 1 and 4 p.m., there is a rise in temperature levels.

The Figure 5.5 depicts the data collection process as well as a graphical depiction of the parameter values as fields. The sensor data is sent to the cloud server, in our case ThingSpeak, after passing through the Arduino and Raspberry Pi. We can observe some graphical analyses of

past data on the cloud server. And, thanks to the internet, this cloud server can be accessed and monitored from anywhere. All previous data can be stored and viewed in ThingSpeak. Because ThingSpeak is a free service, there are numerous mobile apps that assist with connecting the sever channel for the purpose of observation.

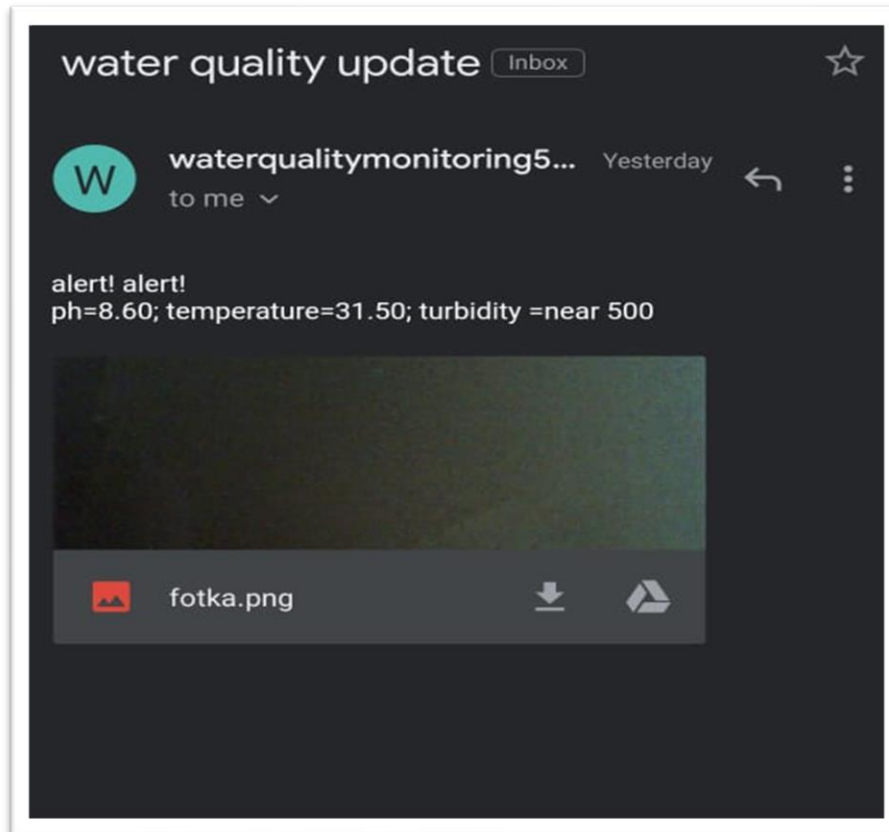
As illustrated in Figure 5.5, we are evaluating four aquafarm fields: temperature, pH, turbidity, and food supply. The temperature and pH sections display the real sensor data from the pond. The turbidity values are actually the variance values, which we use as a turbidity score. And the Food supply field displays the number of times the food has been supplied over a given period of time.



**Figure 5.5 Cloud Data Collection**

### 5.3 Mail Alerts

If the parameters are not in the optimal range, a mail alert will be issued to the user or farmer, along with the ensuing action. The temperature and pH sensor measurements, as well as the turbidity variance value, will be emailed. In addition, a photo of the aquafarm will be given to provide a brief overview of the aquafarm's surroundings.



**Figure 5.6 Mail alert received by user**

As seen in the figure 5.4, the parameters' values, as well as the configuration graphic, will be emailed to the customer. The temperature in the above mail is 31.50°C, which is below the optimum range (26–33°C), but the pH value is 8.60, which is higher than the optimal range (7.5–8.5), and hence the mail warning was received. We are using variance as a score for turbidity. So, in the preceding email, the 500 represents the variance value rather than the true turbidity value. The aim of the email notifications is simply to warn you that the conditions are out of bounds, as subsequent action has already been taken with the assistance of Fuzzy Logic.

## **CHAPTER 6**

### **CONCLUSION AND THE FUTURE WORK**

This chapter brings the thesis to a close on DEVELOPMENT OF DATA DRIVEN IOT ENABLED WATER QUALITY MONITORING FOR AQUACULTURE and suggests additional functionalities for future implantation.

#### **6.1 Conclusion**

In this project, an IoT-based system for measuring and controlling different parameters within an aqua farm was demonstrated. Some image processing techniques were used to extract turbidity from the image. Using Fuzzy Logic, a decision-making mechanism has been developed to take the necessary steps when the aqua farm parameters are out of bounds, as well as to send a mail warning to the recipient. The pond parameters, such as temperature, pH, and turbidity, were uploaded to a cloud server, which can be accessed from anywhere in the world with an internet connection. We were also able to keep this project very cost efficient because we only found a few critical water parameters and used image analysis for turbidity instead of an actual sensor.

#### **6.2 Future work**

We are just able to tell whether the water is turbid or not because we did not use a turbidity filter. Instead, we can pinpoint the exact turbidity value if we construct a dataset of variance and turbidity. We can monitor the water temperature with the aid of a weather forecast/prediction to achieve the best results without wasting resources (power). When manufactured on a wide scale, the circuit would be more cost efficient and robust.

## APPENDIX

### Arduino Coding :

```
#include <DallasTemperature.h>
#include <OneWire.h>
#include<SoftwareSerial.h>
#include<stdio.h>
#define ONE_WIRE_BUS 10
#include <Servo.h>

Servo servo;

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
float Celcius=0;
void yield() { }
#define t 4
int cool=13;
int servoPin = A1;
#define txp 12
#define rxp 11
SoftwareSerial Serial1(rxp,txp);
char dataString[50] = {0};
int a =0;
int i = 0;

void setup(){
  Serial.begin(9600);
  Serial1.begin(9600);
  sensors.begin();
  pinMode(cool, OUTPUT);
  servo.attach(servoPin);
}
```



```

void loop(){
  String data = Serial.readStringUntil('\n');
  if(data[0]=='1'){
    servo.write(30);
    delay(2000);
    for(i = 0; i < 150; i++)
    {
      servo.write(i);
      delay(20);
    }
    delay(2000);
    for(i = 150; i >= 0; i--)
    {
      servo.write(i);
      delay(20);
    }
  }
  if(data[2]=='1'){
    digitalWrite(cool,HIGH);
    delay(float(data[4])*2000);
  }
  if(data[2]=='2'){
    digitalWrite(cool,HIGH);
    delay(float(data[4])*2000);
  }
  if(data[2]=='0'){
    digitalWrite(cool,LOW);
    delay(float(data[4])*2000);
  }
  if(data[3]=='1'){

```

```

    digitalWrite(cool,HIGH);
}
if(data[3]=='0'){
    digitalWrite(cool,LOW);
}
if(data[1]=='1'){
    servo.write(0);
    delay(2000);
    for(i = 0; i < 150; i++)
    {
        servo.write(i);
        delay(20);
    }
    delay(2000);
    for(i = 150; i >= 0; i--)
    {
        servo.write(i);
        delay(20);
    }
}
if(data[1]=='2'){
    servo.write(0);
    delay(2000);
    for(i = 0; i < 150; i++)
    {
        servo.write(i);
        delay(20);
    }
    delay(2000);
    for(i = 150; i >= 0; i--)
    {

```

```

        servo.write(i);
        delay(20);
    }
}
String s = Serial1.readStringUntil(',');
sensors.requestTemperatures();
Celcius=sensors.getTempCByIndex(0);

if(s[2]=='P'){
    Serial.print(s+",");
    Serial.print(Celcius);
}
delay(100);
}

```

### **Raspberry pi Coding :**

```

import serial
import re
import time
import urllib.request
from datetime import datetime
import datetime
import cv2
import array as arr
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

```

```

#camera
cam_port = 0

# time for food
now = datetime.datetime.now()
last=now
f=0
lastph=lastpl=now

#cloud
URL='https://api.thingspeak.com/update?api_key='
KEY='9364W5QHLOBHRIZY'
#email
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
from email.mime.multipart import MIMEMultipart
fromaddr = "waterqualitymonitoring5@gmail.com"
toaddr = "waterqualitymonitoring5@gmail.com"
import smtplib
server = smtplib.SMTP('smtp.gmail.com:587')
server.ehlo()
server.starttls()
server.login("waterqualitymonitoring5@gmail.com", "S@ndhya03")

#serial communication receiving
ser = serial.Serial('/dev/ttyACM0',9600)
send="100\n"
ser.write(send.encode('utf-8'))
print(send)

```

```

#s = ser.readline()
a=[0,0,0,0]
#loop
while True:

    while ser.in_waiting<=0:
        c=0
    if(ser.in_waiting > 0):
        s = ser.readline()
        if len(s)>13:
            print(s)
            n=0
            x=str(s)
            tur= 250
            a=re.findall(r"[-+]?[d*]\.d+|\d+", str(s))
            #food conditions
            f=0
            now=datetime.datetime.now()
            if((now-last).seconds>21600):
                last=now
                f=1
            #ph conditions
            if(float(a[0])>8):
                if((now-lastph).seconds>20):
                    n=1
                    lastph=now
            elif float(a[0])<6:
                if((now-lastpl).seconds>10):
                    n=2
                    lastpl=now
            else:

```

```

        n=0
#temp conditions
    if(float(a[1])>30):
        p=1
    elif(float(a[1])<25):
        p=2
    else:
        p=0
#Fuzzy logic implementation
    quality = ctrl.Antecedent(np.arange(33, 51, 1), 'quality')
    temp = ctrl.Consequent(np.arange(0, 101, 1), 'temp')

    quality.automf(3)

    temp['low'] = fuzz.trimf(temp.universe, [0, 0, 50])
    temp['medium'] = fuzz.trimf(temp.universe, [0, 50, 100])
    temp['high'] = fuzz.trimf(temp.universe, [50, 100, 100])

#rules
    rule1 = ctrl.Rule(quality['poor'], temp['low'])
    rule2 = ctrl.Rule(quality['average'], temp['medium'])
    rule3 = ctrl.Rule(quality['good'], temp['high'])

    tipping_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])
    tipping = ctrl.ControlSystemSimulation(tipping_ctrl)

    tipping.input['quality'] = float(a[1])
    tipping.compute()
    p1 = tipping.output['temp']
#turbidity
    cap = cv2.VideoCapture(cam_port)

```

```

cap.read()
img_name = "fotka.png"
ret, img = cap.read()
cv2.imwrite(img_name, img)
tur=cv2.Laplacian(img, cv2.CV_64F).var()
cap.release()
print(tur)
if(tur<400):
    z=1
else:
    z=0

```

#sending data to arduino

```

send=str(f)+str(n)+str(p)+str(z)+str(p1)+"\n"
ser.write(send.encode('utf-8'))
time.sleep(0.5)
print(send)

```

#cloud

```

HEADER='&field1={}'.format(a[1])
HEADER1='&field2={}'.format(a[0])
HEADER2='&field3={}'.format(tur)
HEADER3='&field4={}'.format(f)
NEW_URL = URI+KEY+HEADER+HEADER1+HEADER2+HEADER3
data=urllib.request.urlopen(NEW_URL)

```

#mail alerts

```

if(float(a[0])>8 or float(a[0])<6 or float(a[1])>30 or float(a[1])<23) or tur<400:
    msg = MIMEMultipart()
    msg['From'] = fromaddr
    msg['To'] = toaddr

```

```
msg['Subject'] = "water quality update"
body="alert! alert! \nph="+a[0]+"; temperature="+a[1]+"; turbidity =" +str(tur)
msg.attach(MIMEText(body, 'plain'))
part = MIMEBase('application', "octet-stream")
part.set_payload(open("fotka.png", "rb").read())
encoders.encode_base64(part)
part.add_header('Content-Disposition', 'attachment; filename="fotka.png"')
msg.attach(part)
text = msg.as_string()
server.sendmail(fromaddr,toaddr,text)
```



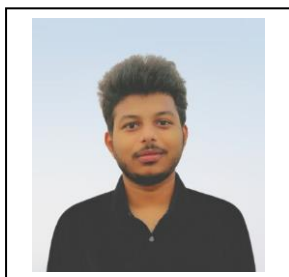
## REFERENCES

- [1] Kiruthika, S.U., Raja, S.K.S., Jaichandran, R., 2017. IoT based automation of fish farming. *Journal of Advanced Research in Dynamical and Control Systems*. 9. Pp. 50-57.
- [2] Durga, S.B., Nirosha, K., Priyanka, P., Dhanalaxmi, B., 2017. GSM based Fish Monitoring System Using IOT, *International Journal of Mechanical Engineering and Technology* 8(7), pp. 1094–1101.
- [3] Salim, T.I., Alam, H.S., Pratama, R.P., Anto, I.A.F., Munandar, A., 2017. Portable and online water quality monitoring system using wireless sensor network. 2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT). IEEE 1–7.
- [4] Francis, E. I., Olowoleni O.J., Ibhaze, A.E., Oni, O., 2017. IoT Enabled Real-Time Fishpond Management System.
- [5] Nam, H., An, S., Kim, C.H., Park, S.H., Kim, Y.W., Lim, S.H., 2015. Remote monitoring system based on ocean sensor networks for offshore aquaculture. *Oceans*. IEEE 1–7.

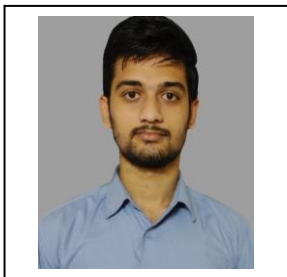
## BIODATA



Name : Pannala Manish Reddy  
Mobile Number : 7397434968  
E-mail : [pannalamanish.reddy2017@vitstudent.ac.in](mailto:pannalamanish.reddy2017@vitstudent.ac.in)  
Permanent Address : Hyderabad, Telangana.



Name : Dodda Ravi Kumar  
Mobile Number : 7397465882  
E-mail : [doddaravi.kumar2017@vitstudent.ac.in](mailto:doddaravi.kumar2017@vitstudent.ac.in)  
Permanent Address : Vijayawada, Andhra Pradesh.



Name : Medisetti Abhaya Hanuma  
Mobile Number : 7397372312  
E-mail : [abhaya.hanuma2017@vitstudent.ac.in](mailto:abhaya.hanuma2017@vitstudent.ac.in)  
Permanent Address : Vijayawada, Andhra Pradesh.

## VIDEO DEMO

The following is the link of the video demo of our project.

- Link : <https://youtu.be/obJv6llsj5M>