AQUARIUM MONITORING SYSTEM

BY

THIYRAASH A/L DAVID

A REPORT

SUBMITTED TO

UNIVERSITI TUNKU ABDUL RAHMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

BACHELOR OF INFORMATION TECHNOLOGY (HONS) COMPUTER

ENGINEERING

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

(PERAK CAMPUS)

MAY 2017

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: ____

____

____

Academic Session: _____________

I ____

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_________________________ _________________________

(Author's signature) (Supervisor's signature)

Address:

____

_________________________ _________________________

_________________________ Supervisor's name

Date: _____________________ **Date**: ____________________

AQUARIUM MONITORING SYSTEM

BY

THIYRAASH A/L DAVID

A REPORT

SUBMITTED TO

UNIVERSITI TUNKU ABDUL RAHMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

BACHELOR OF INFORMATION TECHNOLOGY (HONS) COMPUTER

ENGINEERING

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

(PERAK CAMPUS)

MAY 2017

DECLARATION OF ORIGINALITY

I declare that this report entitled "**AQUARIUM MONITORING SYSTEM"** is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree **or** other award.

Signature: ________________________

Name: _________________________

Date: _________________________

ACKNOWLEDGEMENT

First of all, I would like to express my deepest gratitude and thanks to my FYP supervisor Mr. Teoh Shen Khang for his valuable knowledge, experiences and endless patience to guide me to build this IOT based embedded system throughout the whole project lifecycle. The lecture of Mr Teoh Shen Khang about the embedded system in the past time helps me to expose to the knowledge on how an embedded system work. Those knowledge helps me a lot in build this IOT based embedded system. Besides that, I would like to thank my family and friends for their encouragement and motivation throughout this project.

Last but not least, I would also like to thank to FICT for giving this opportunity to do this project as my FYP as I really interested in IOT based topic. I also have the opportunity to learn website development and design from scratch in this project which is an extra asset for me.

ABSTRACT

This project is mainly on building an Aquarium Monitoring System using a single chip computer called Raspberry Pi. The main purpose of this project is to help those who are having difficulties in maintaining their indoor aquariums, especially those who are frequently outstation, thus unable to constantly monitor their aquariums. Through the use of this system, users can monitor and maintain their fish aquarium regularly via internet, using devices such as smartphones and laptops. The major role of this system is to enable users to monitor and maintain their fish aquarium through a server of database, which include tasks such as feeding the fishes on time, checking the water temperature, water level, and changing the water automatically, whenever the turbidity level of the water reaches a pre – determined unsafe point for the fishes. When there are emergencies such as water leakage in aquarium or the drop of the water level of aquarium to below normal levels, the system would trigger an alarm and send a message to notify the user to take any appropriate action. In order to continuously check the aquarium's status, the Raspberry Pi 3 is chosen as central board to collect data from sensors and subsequently uploads the data to the database and to the own host website. The user will then login to the website hosted by Raspberry Pi to check on the status of their aquarium. Besides that, sensors such as water temperature sensor, turbidity sensor, water leaking sensor and water level sensor are also required to make sure that the system functions optimally. Other than that, water pump, water filter and motors for feeding the fishes, are also required.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

IoT	Internet Of Things
IDE	Integration Development Environment
GUI	Graphical User Interface
OS	Operating System
SDLC	Software Development Life Cycle
USB	Universal Serial Bus
DC	Direct Current
LED	Light Emitting Diode
RGB	Red, Green, Blue
GPIO	General Purpose I/O pins
ADC	Analog to Digital Converter
PHP	Hypertext Preprocessor
RTC	Real Time Clock
GPRS	General Packet Radio Service
SoC	System On A Chip

Chapter 1: Introduction

1.1 Project Background

The term "Internet of Things (IoT)" (Huang & Li, 2010; Uckelmann, Harrison, & Michahelles, 2011) has been around for the past few years and is gaining recognition with the breakthrough of advanced wireless technology. Although there are several definitions for the phrase "Internet of Things" the best interpretation would be the integration of the physical world with the virtual world of the Internet. IoT is a network of physical objects that contain embedded technology to communicate and sense or interact with their internal or the external environment. Throughout the years, there have been many products built based on the IoT concept, for example, smartphones, smart watches, laptops, household electronic devices and other monitoring electronic system devices. Among these devices, the aquarium monitoring system is an electronic monitoring system that is grabbing a lot of attention nowadays.

What is an aquarium? An aquarium is a container or an artificial pond in which living aquatic animals or plants are kept. Aquarium is used for fishkeeping purpose for hobby and for indoor and outdoor decoration. Besides that, fishkeeping in aquarium is also related to various cultures, for example it is related to Vastu for Indians and Feng Shui for Chinese. According to Vastu, the aquarium is a perfect combination of harmony and balance. The water symbolizes the flow of life, growth and activities of living things. The motion and sounds of the bubbling water as it moves throughout the fish tank activates and increases the positive energy around the area, thus, bring good fortune, wealth and abundance. Meanwhile, according to Feng Shui principles, placing an aquarium in the home or office is an excellent way to attract auspicious chi into the space, especially for good luck, abundance and prosperity. The aquarium monitoring system was then introduced order to properly maintain the aquarium. Besides that, with the help of Iot devices such as aquarium monitoring system, users can monitor and manage their aquariums and fishes, anytime and from anywhere through the internet.

1.2 Motivation and Problem Statement

Nowadays, daily expenses have increased dramatically and to cope with this, many people go work from early morning until late evening or night. This, has in turn has made them unable to keep their house environment clean, especially for those who own fish aquariums. They might be unable to feed their fish on time or clean the aquarium frequently. If a fish tank is not cleaned frequently, waste products from the fishes would accumulate in the water, thus, making the water cloudy. This would in turn cause the water to turn into an unpleasant greenish and also give of an unpleasant odor to the surroundings. Due to high chemical content in the water and an improper feeding schedule, fishes in the aquarium might be suffocated and starve to death. Hence, an Aquarium Monitoring System is designed to help solve the issues mentioned above, with the features below:

- An Aquarium Monitoring System will constantly transfer the status of the aquarium on the database and users can monitor them through the internet.
- An Aquarium Monitoring System is able to detect the aquarium water temperature, cloudiness of water, water level and any water leakage from out of the aquarium.
- An Aquarium Monitoring System will feed the fishes at appropriate times automatically, based on fixed times or the users could also feed the fishes at a click of a button through the internet.
- An Aquarium Monitoring System will able to make appropriate decisions and perform actions such as changing the aquarium water if the water is too cloudy.
- An Aquarium Monitoring System will also trigger the alarm and alert users if there was any water leakage from the aquarium.

With the advancement in technology, there are many aquarium monitoring systems available in the market nowadays. However, the existing systems are lacking in several functions or features as listed in the following:

- Some aquarium monitoring systems cannot make appropriate decisions.
- Some aquarium monitoring systems are not portable.
- Some aquarium monitoring systems are not IOT - based systems.
- Some aquarium monitoring systems are not user-friendly.
- Some aquarium monitoring systems are high in cost.

1.3 Project Scope

The main aim of this project is to develop a system that is able to monitor the status of the aquarium at all times and enable the user to monitor status of the aquarium by logging on to the database system and perform appropriate actions through the internet itself or let the system itself perform informed decisions and actions. Besides that, system also have to be portable, come at a low cost, uses low power, convenient and efficient. To overcome the issues mentioned above, several measures have been implemented for this project:

- The coverage of this system is limited to small size aquariums, such as 10-30 galloon fish aquarium.
- The boundary of the area used is only within a certain range of the home or office which has a good internet coverage, as the system needs to update the user on the status of the aquarium via the internet.
- The project requires programming skill to handle the Raspberry Pi coding that is Python language, plus some knowledge to configure an own database server in Raspberry pi is also required to complete this project.

1.4 Project Objective

The Objectives of this project are as follow:

- To develop a reliable aquarium monitoring system for users.
- To develop a real time based aquarium monitoring system to send the data about the aquarium to the users on time to time by updating the data to the server.
- To develop an aquarium monitoring system that is able to update the users on the current situation of the aquarium, thus enabling the users to take appropriate actions based on the data received.
- To develop an aquarium monitoring system that is able to make informed decisions and perform certain action on its own during critical situations without the users prompting it.
- To develop an aquarium monitoring system that can be understood and operated by non-technical users easily without the need for any advanced technical knowledge.
- To make it more convenient and easier for users to maintain their aquarium from anywhere around the world.

1.5 Impact, Significance and Contribution

There are many aquarium monitoring systems available for users in the market, but there is still some limitations which can be further improved. A high power consumption by the device is also a cause for concern. Many users prefer to choose products with lower power consumptions compared to products that functions similarly but consume more power. Therefore, when designing this system, use of an appropriate microcontroller is very important.

Besides that, the aquarium monitoring system must also be based on real time. This is because, the main objective of the system to update the user on the status of the aquarium and enable the users to take appropriate actions based on the information received. If the system does not operate on real time, when the aquarium is in any critical situation which needs human interference, such as aquarium water leakage, the system might unable to prompt the user to take appropriate actions on time, to overcome the problem. Thus it is crucial for the system to be based on real-time.

Other than that, the aquarium monitoring system must also be able to make some informed decisions and perform actions that might not need human interference, for example, if system found the water to be too cloudy, it must automatically switch on the motor to suck out the cloudy water and replace with clean water. It also has to feed the fish automatically at appropriate time intervals.

Last but not least, the aquarium monitoring system must be built in such a way that users can understand how to use all the features available in the system easily and clearly. For example, the GUI (graphical user interferences) of the website of the aquarium monitoring system should be built in a convenient and user-friendly way so that non-technical users can use it without any difficulty.

As a conclusion, through the development of this system, users can properly maintain their aquariums even when they are far away from home or office.

Chapter 2: Literature Review

2.1 Literature Review and Critical Remark of Previous Works

Based on the research, there are some of the similar products related to this current project. One of the products which known as "**A Multi-functional Aquarium Equipped with Automated Thermal Control/Fodder-Feeding/water Treatment using a Network Remote Control System**" develop by (Min-Chie Chiu, 2010). After analyzing on their product, found that there have a few strength and weakness on their product.

In the product, PC (personal computer) is used as main controller to manage all the sensors. The sensors were connected to the PC controlling system via different module. The module is the ADC (Analogue to Digital) which convert the analogue signal from the sensors to digital signal so that the PC can read and classify the values. Then, via the VB interface, the client PC can communicate with server-PC to monitor and control the aquarium based on the data from sensors. The **strength** of the product is the used of PC as the controlling system. PC has high processing power and it can manipulate the data obtained from the sensors via modules faster compared to microcontroller which has limited processing power.

However, there are some **weaknesses** exist in this product. At first is the cost of the product. The product consists of a PC as center of the system which control and managed the whole system. It is really expensive to allocate a whole PC just to manage the aquarium system. Second is the portability of the product. The used of PC as center part of system make it difficult to move the product from one place to another. The solution for the cost and portability issue can be solved by using a single board computer called Raspberry pi, to replace the PC to function as main controller of the system to operate the system.

Another system that related to this project is **"The Design and Development of Automatic Fish Feeder System using PIC microcontroller**" by (M. Z. H. Noor, A. K. Hussian, M. F. Saaid, M. S. A. M. Ali, M. Zolkapli,2012). The microcontroller used in this research project is PIC16F886. The system developed combines mechanical and electrical system in controlling fish feeding activity. This device, basically consists of pellet storage, former, stand, DC motor and microcontroller. The pellets controlled by DC motor which located under the pellet storage. A control system was then attached to this device allowing the fish to be fed at the right cycle time as required or predefined by user. Timer was employed in this device to control the motor rotation attached to sphere former, which dispense the pellets into the water. The pellets dispensed into the marking area of the pond based solely on the rotation speed of the motor itself. The controller came with a keypad giving user more option in determining the suitable speed for the motor depends on their cattle. The **strength** of the project is, it able to dispense pellets into the desire area based on the rotation speed of the motor, combined with suitable cycle time. This system replace the traditional way to control the motor speed that is by using resistance strung in rotor circuit or adjust the voltage of electrical machine with used of PWM (Pulse width modulation) that is change speed by changing duty-cycle.

Anyhow, there is some **weakness** in the following system that is the user cannot change the DC motor speed remotely because the microcontroller is not connected to internet. The microcontroller just automatically dispenses pellets on desired area using DC motor at the schedule time. If the user wants to change the DC motor speed or the schedule time, he/she have to manually go to the place where the microcontroller is located and change value by key in the value using the hex keypad. This restricts the user from getting full control of the system. This solve by connect the main microcontroller of the system to internet via low budget Raspberry pi minicomputer with own configured server, where user can connect to the microcontroller through IP address and control the DC motor or schedule time from anywhere through internet.

Besides, there is another research project regarding this project that is **"Automatic Arowana Raiser Controller Using Mobile Application Based on Android"** by (Nurliani Hidayah Ritonga; Agung Nugroho Jati; Rifki Wijaya 2016). This system is called AURORA systems consist of Raspberry pi as microcontroller which is always online and it is connected to sensors like, ultrasonic sensor, temperature sensor and actuator like servo motor to feed, light for aquarium and water pump. The measure value of different sensors will be send to raspberry pi and uploaded to the cloud so that the android based end devices can access the data and take the appropriate actions. The **strength** of this project is the implementation of Internet of things (IoT) concept. Android application is used to monitor and control the sensors and actuator that are connected to the microcontroller through the cloud services. This helps the user to access the information about their aquarium and control them from anywhere in the world through internet easily.

The **weakness** of this project is although they have sensors to examine their aquarium condition and android apps to monitor and control their aquarium's actuator, the system still need human's interference to control the servo motor for feeding, to switch on or off the light for aquarium and to change water pump for the aquarium. The system does not have the ability to make own appropriate decision based on the sensors value of the aquarium. For example, the system should feed the fish on time or change the water of aquarium when the water turn cloudy automatically without human interferences. This is feature is very important because if the user when for long holiday away from their house, it's not practical that the user have to monitor and feed the fish all the time because human can forget.

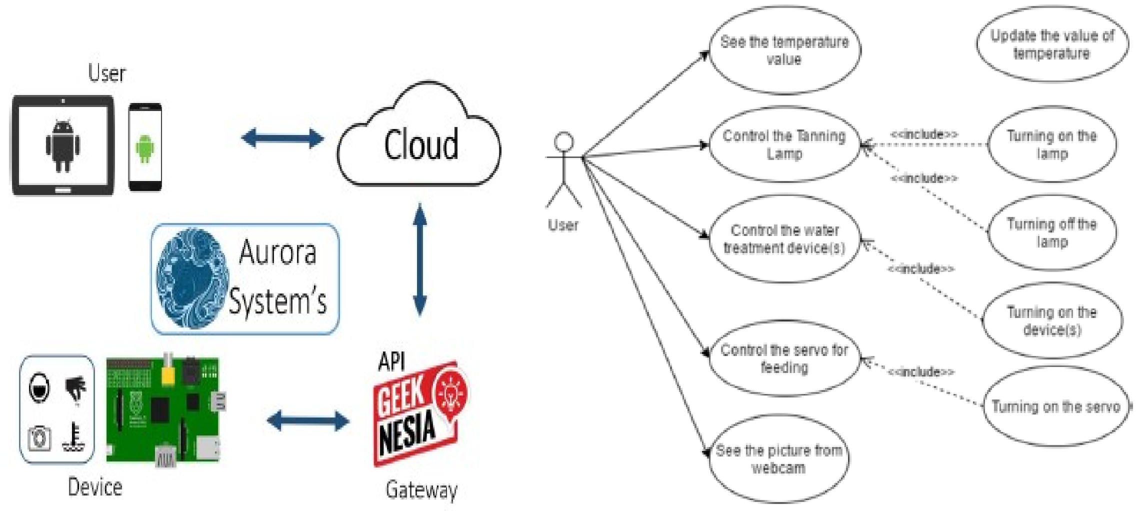


Figure 2-1-F1: Architecture Design and Use Case Design of AURORA

Furthermore, there is another research related to this project, that is "**Smart Water Quality Monitoring System**" by (A.N.Prasad, K. A. Mamun, F. R. Islam, H. Haqva,2015). This system consist of a microcontroller board called Waspmote with external ADC connected to different sensors such as pH, water temperature ,turbidity , conductivity sensors. All the sensors and microcontroller used are from the Libelium World, a leading company from Europe. All the water information will be collect by the Waspmote microcontroller board and store them in two places either sends to Cloud through GSM and or store in SD card. The **strength** of this project is the complexity of the product. This product is very easy to use just by connecting the sensors and connects the microcontroller across the internet; user can easily access the information about the water quality either in aquarium, lakes or pond easily using their end devices. Besides, the product is also very portable because the Waspmote microcontroller board size is small that can exactly fit to our palm hand. User can easily bring the whole system from one place to another.

However, although the product is easy to use, the cost of the product is too high for the user to afford. For example, from its Libelium World Company's official website, the cost for temperature sensor, pH, Turbidity sensor, conductivity sensor without include theirs calibration kit itself almost reach 3375€. If include the Waspmote microcontroller board Plug & Sense! SE-PRO LoRa - 868 which is 415€, the total cost of the whole system is approximately 3790€ that is RM17234 in Malaysia currency. The Europe user might think the cost is affordable but for the Asia users, it is very high and no one will ever spend that much money just to monitor their aquarium. But in my project, the whole project cost will be approximately around 158.90€ that is around RM700 only. It is way more cheaper compared to the Libelium World's Smart Water Monitoring System's product. Although we are using much cheaper products in our project such as Raspberry pi, its performance and durability is approximately same as the commercial products.

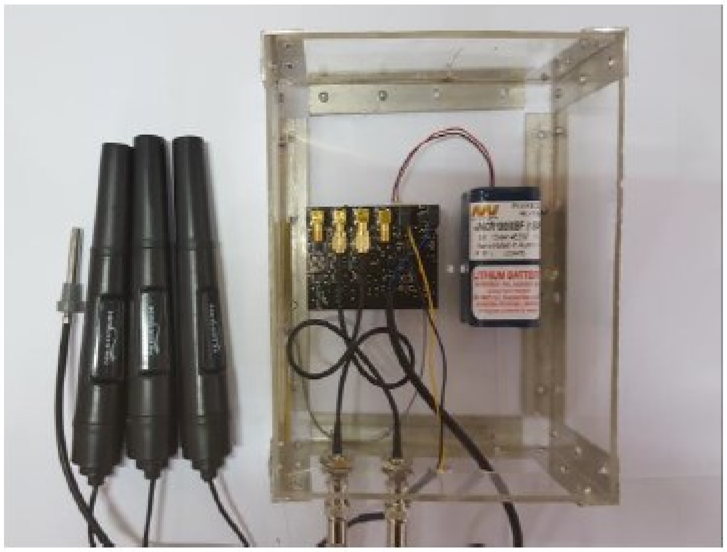


Figure 2-1-F2: Deployment Setup with the sensors and Waspmote microcontroller board.

Another research related to this project is "**An Embedded Fuzzy Decision System for Aquaculture"** develop by (Taotao Xu; Feng Chen,2014). In this project, they develop a Fuzzy Decision System based embedded water quality monitoring platform. In simple, a system that can take appropriate decision based on the data get from the sensors. The **advantage** of this research is the research itself that is a system that can make appropriate decision. This type of system is very important because human can't monitor the aquaculture all the time. Besides, in critical situation, human might not able to make correct decision on correct time due to panic. With this system, when there are some changes in the aquaculture, the system able to monitor them and take an appropriate action for the changes.

The weakness of this project is it only the result of the decision-making are send to the aquaculture farmers via GPRS or messages. The farmers will be not notify about the sensor values for example the temperature value or the pH value of water. Besides that, the user/farmer also not able to see the sensor value on real time for example with a real time graph plotting with time domain through internet because this system is not IoT based system. Real time graph plotting with time domain is very important because if there is something wrong with the aquaculture, for example some fish have died at particular time, the user/farmer able to trace back the graph on that particular time to identify what have go wrong. Although the system able to make correct decision on correct time, they are still a machine. Machine can go wrong. So human interference also is needed to ensure everything working perfectly. In my project, there are two mode on monitoring our aquarium system, one is manual mode which control by the user and another is auto mode which are control by the system itself. A better Aquarium Monitoring System should be develop based on the energy efficiency, portable, reliability, ease of use and low cost with high functionalities.

Chapter 3 System Design

3.1 System Overview

In this session, how the aquarium monitoring system is designed will be discuss in detail by using some diagram and chart such as block diagram, activity diagram, flow charts, and pictures of the system. Raspberry Pi 3 are use as the main controller to monitor and control the sensors and actuators in this aquarium monitoring system. There are about four sensors and six actuator that the Pi needed to control. Below show the block diagram and the picture of this aquarium monitoring system.

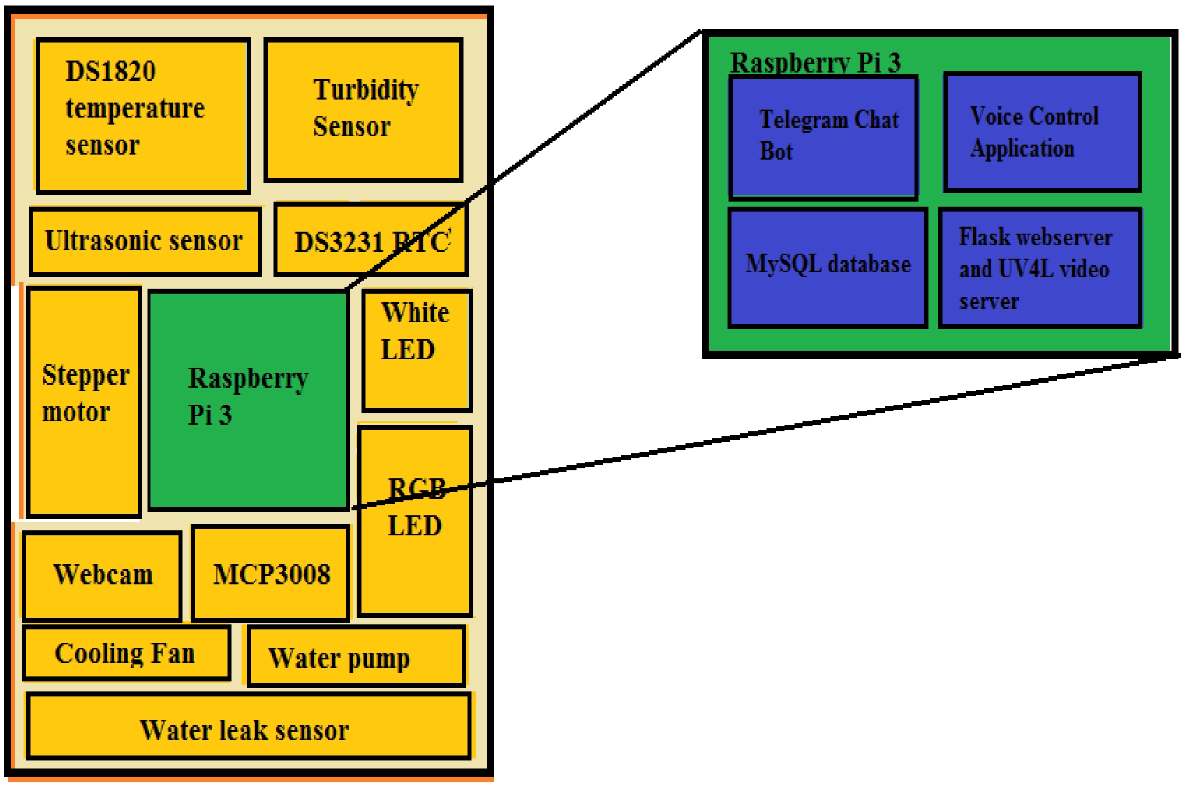


Figure 3-1-F1: Block diagram of Aquarium Monitoring System

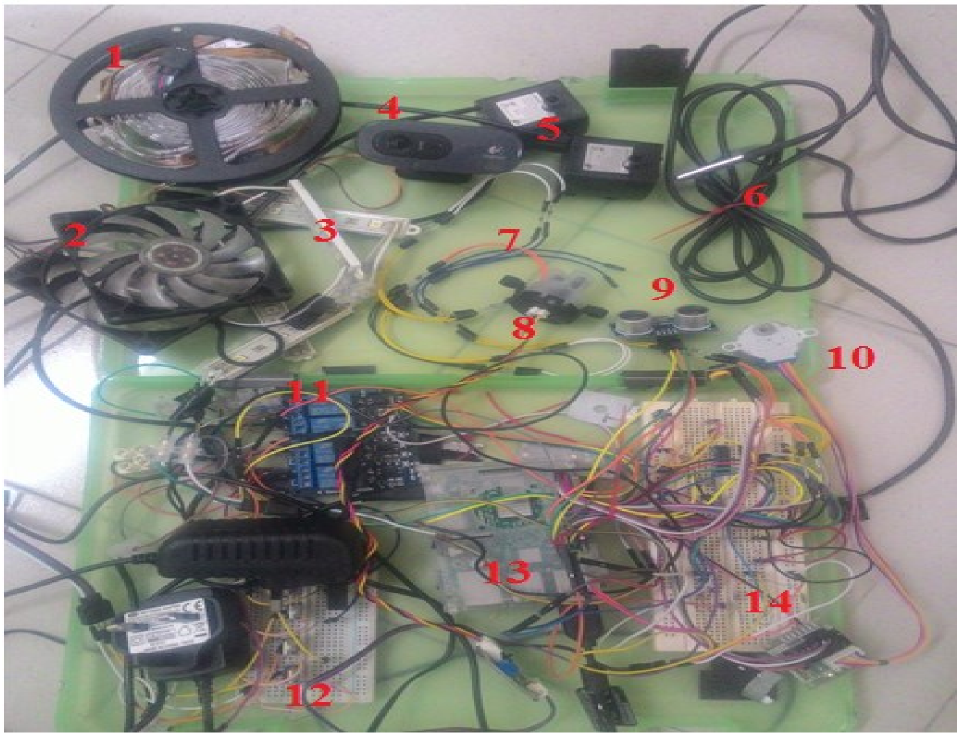


Figure 3-1-F2: Physical connection of the components of system

1. RGB LED Strip
2. Cooling Fan
3. White LED
4. Logitech C270 Webcam
5. Water Pump(x2)
6. DS128B water temperature sensor
7. DIY water leak sensor
8. Turbidity sensor
9. Water level Sensor
10. Fish Feeder (Stepper Motor)
11. 5V 4 channel relay
12. RGB Control circuit
13. Raspberry Pi 3
14. Main circuit for the system

The project's hardware and software components, their specification setup and connection are discussed in detail in Chapter 5 Technologies and Tools Involved.

3.1.1 Algorithm Design

This aquarium monitoring system consist of two modes, one is manual mode and another is auto mode. The manual mode need the user interferences. For example , to feed the fish , the user have to manually send command to Pi using Web, Telegram or Voice Control application. Whereas in automatic mode, the Pi able to make intelligent decision based on the sensor values, for example feed the fish if the time is 9pm, change the water if it's too dirty by getting the turbidity sensor value and so on. So to do all the thing mention above an algorithm is needed. Below shows the main algorithm flowchart and auto mode pseudo code used in this system.

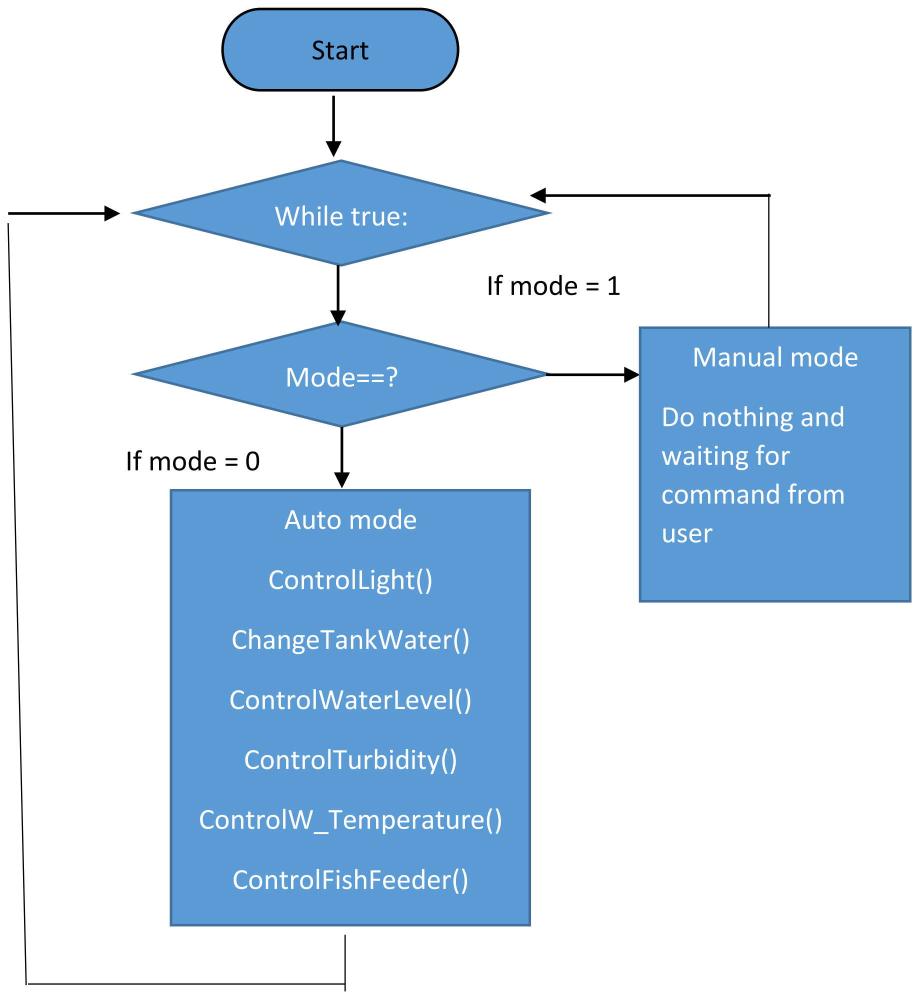


Figure 3-1-F3: Algorithm flowchart of the system for change modes

ControlLight()

```
CurrentTime =getCurrentTime()

If  (CurrentTime > 7.00pm
       and  CurrentTime < 6.30am)
     On the aquarium light (OnLight())

Else
     Off the aquarium light  (OffLight())
```

ChangeTankWater()

```
TankEmpty()
Sleep for half second
TankRefill()
```

ControlWaterLevel()

```
CLevel=getWaterLevel()

If (CLevel <=OptimumWlevel
                  and CLevel > HalfWlevel)
    Off both water pump 1 and 2

ElseIf (CLevel <= HalfWlevel)
       Water level is low
        (On Pump 2 to fill the  tank)

ElseIf (CLevel > OptimumWlevel)
       Water level is too high.
      ( On Pump 1 to reduce the water in tank )
Else ( Out of range)
```

ControlTurbidity()

```
CTur=getCturbidity()

IF (CTur >300)
       Sensor is out of water

ElseIf (CTur >=260 and CTur <300)
      Water is clean

ElseIf (CTur <260)
       Water is dirty
         ChangeTankWater()

Else (SomethingWrong)
```

ControlW_Temperature()

```
W_Temp=getTemp()

If (W_Temp >30) -//30 Celcius
    Temperature to High
   On the Cooling Fan

Elseif (W_Temp < 23) //23 Celcius
    Temperature to Low
   On the Water Heater

Else
     Temperature is in Optimum
```

ControlFishFeeder()

```
CurrentTime =getCurrentTime()

If (CurrentTime == 12am)
      FeedTheFish()

Elseif (CurrentTime == 3am )
      FeedTheFish()

ElseIf (CurrentTime == 9am)
      FeedTheFish()

ElseIf (CurrentTime == 12pm)
      FeedTheFish()
```

	```
	ElseIf (CurrentTime == 9pm)
	      FeedTheFish()

	Else
	   Don't feed the fish
	```

Table 3-1-T1: Automatic mode's pseudo code

The algorithm above were initially run in separated python file. Then we encountered problem in sharing variable among the web server file and the algorithm file. To solve this problem, the algorithm file and server file are combined to become one file where the web server run as main thread and the algorithm function run as child thread to the server. By doing this sharing variable among the two function become much easier.

3.1.2 Control modes of the system

There are three ways that the user can control and monitor this aquarium monitoring system, one through website with own hosted web server in Pi, second using a Telegram chat bot and lastly using Voice Control application. In this section, each control modes will be discussed

<u>Website</u>

Website is the main control mode in this aquarium monitoring system where the user can monitor the sensors of aquarium and control the its actuators such as light, water pump, fish feeder and cooling fan by sending commands through the website. For server side, Flask a python based micro-framework for web development are used as web server. The page skeleton is written in HTML together with CSS and JavaScript for the client side. JavaScript library called FusionChart is used to plot the real time graph and charts on the website to visualize the sensors values of this system for the user. Below shows the activity diagram and flow chart of the website.

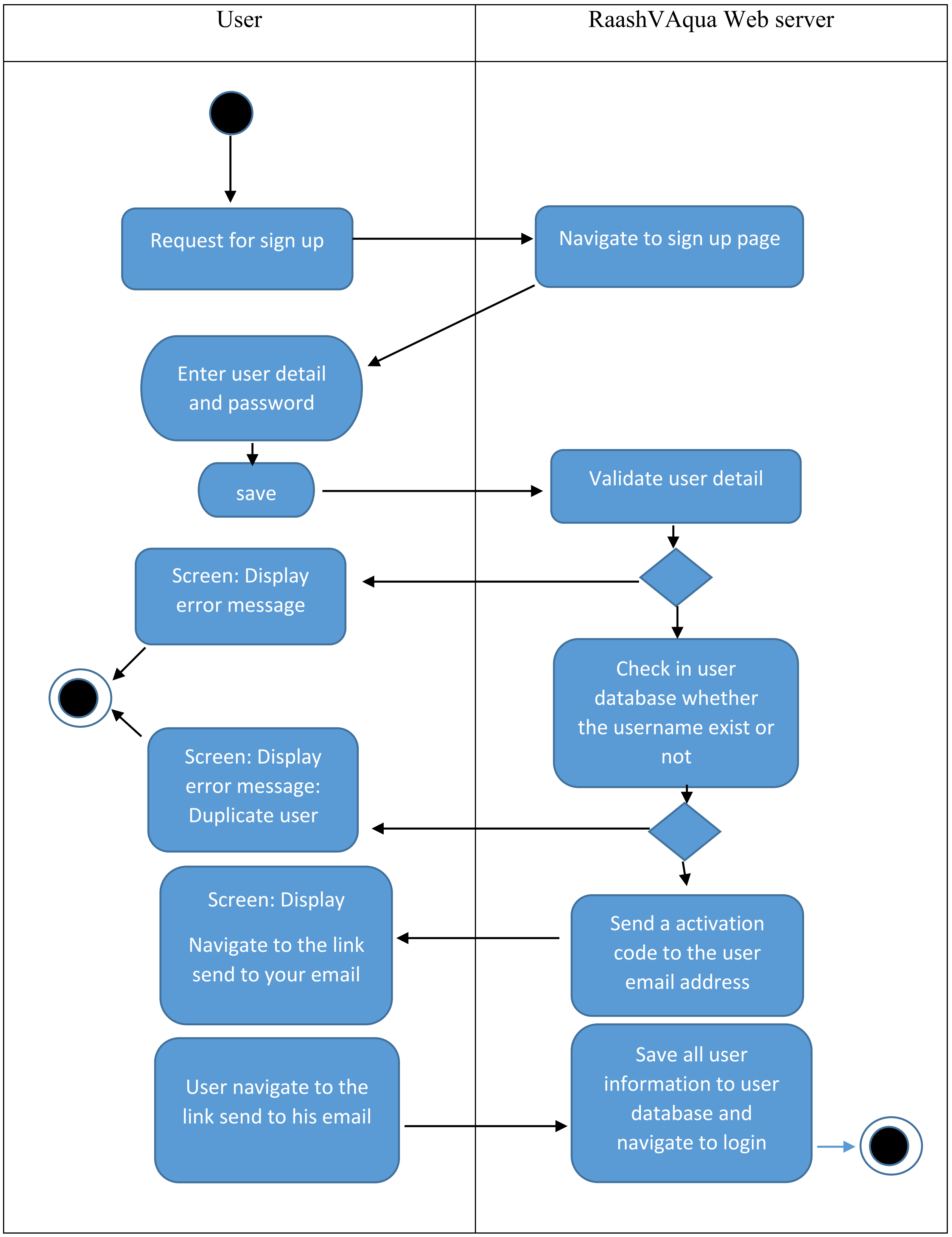

User
RaashVAqua Web server
Request for sign up
Navigate to sign up page
Enter user detail and password
save
Validate user detail
Screen: Display error message
Check in user database whether the username exist or not
Screen: Display error message: Duplicate user
Screen: Display Navigate to the link send to your email
Send a activation code to the user email address
User navigate to the link send to his email
Save all user information to user database and navigate to login

Figure 3-1-2-F1: Activity Diagram for Sign up for website

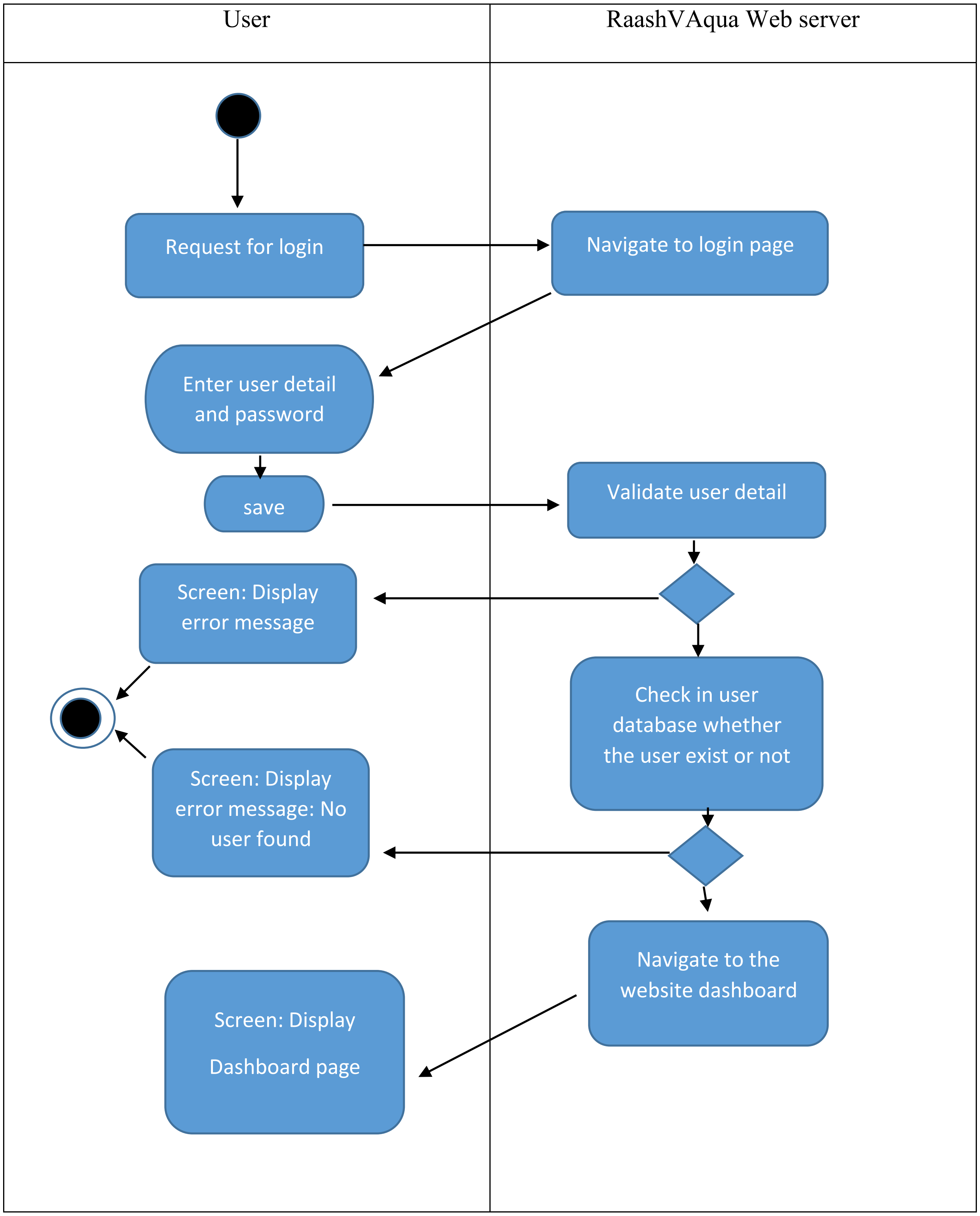

User
RaashVAqua Web server
Request for login
Navigate to login page
Enter user detail and password
save
Validate user detail
Screen: Display error message
Check in user database whether the user exist or not
Screen: Display error message: No user found
Navigate to the website dashboard
Screen: Display Dashboard page

Figure 3-1-2-F2: Activity Diagram for Login up for website

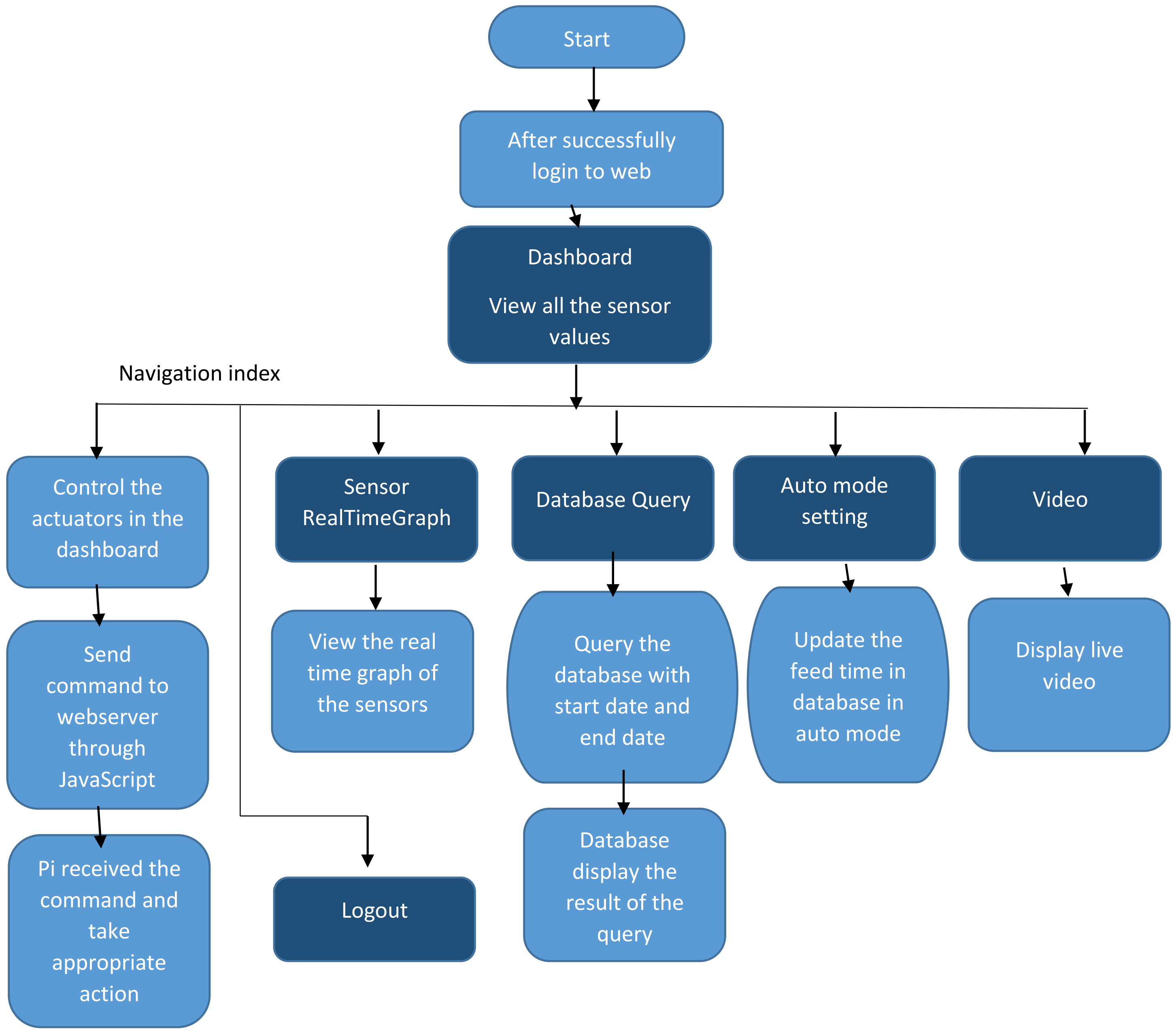

Start
After successfully login to web
Dashboard
View all the sensor values
Navigation index
Control the actuators in the dashboard
Sensor RealTimeGraph
Database Query
Auto mode setting
Video
Send command to webserver through JavaScript
View the real time graph of the sensors
Query the database with start date and end date
Update the feed time in database in auto mode
Display live video
Pi received the command and take appropriate action
Logout
Database display the result of the query

Figure 3-1-2-F3: Overview Flowchart of the website after login

Above figure show the overview flowchart on how the website looks like after the user login into the website. The first page that the user will be navigated is to the dashboard webpage where user can view the sensor values which are in form of graph and charts. The dark blue color box above represent the webpage that can be navigate in this website. Below figure shows the overview how the real website's dashboard looks like,

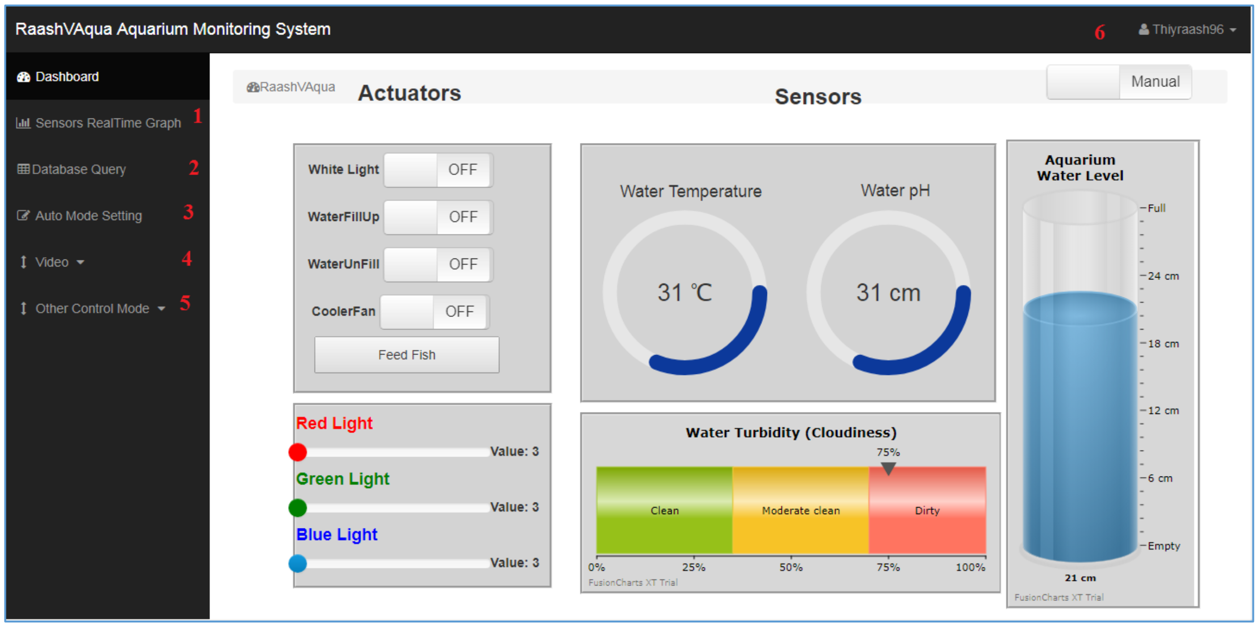


Figure 3-1-2-F4: Overview of real dashboard page of this project's website

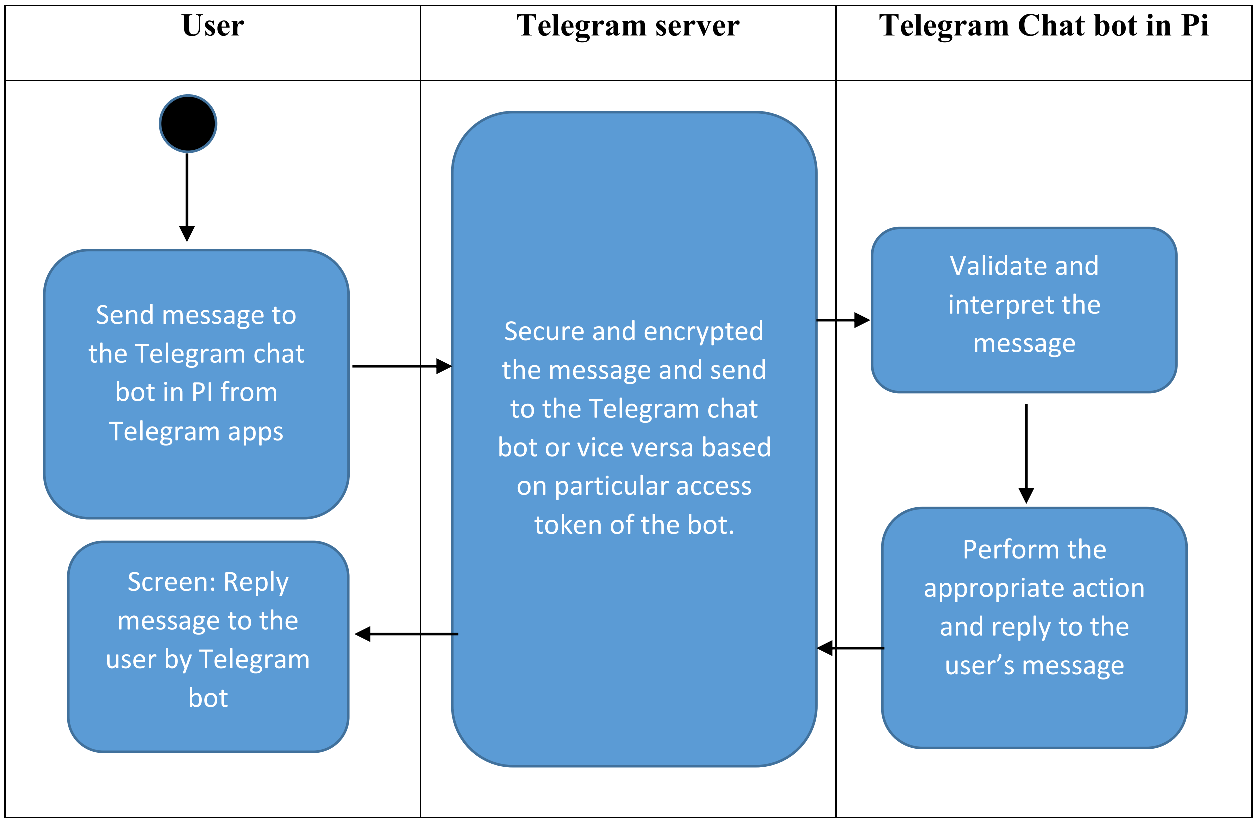
Number	Webpage name	Function
1	Sensors RealTime Graph	Display the real-time sensors values such as water temperature, turbidity and water level as graph
2	Database Query	Query the sensor value on particular date and time by input start datetime and end datetime. Database will displayed value of sensors in form of table to the user.
3	Auto mode Setting	a) View and update the feed time of auto mode b) Enable/Disable certain part of auto mode control
4	Video	View Live Stream of aquarium from USB camera

5	Other Control Mode	a) Display the control commands of Telegram chat bot and Voice control. b) Enable/Disable other mode of control
6	Logout	Logout

Table 3-1-2-T1: Aquarium Monitoring System Webpages

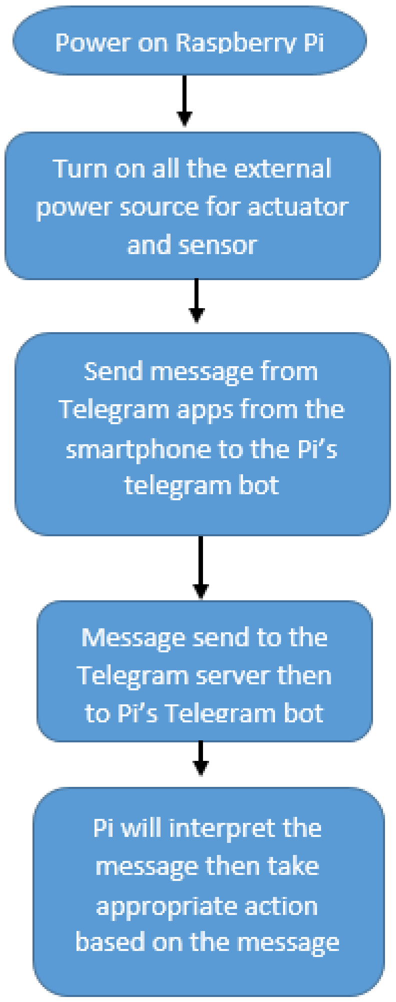
Telegram chat bot

The second way to control the system is by using Telegram chat bot and application. In this project, I host a custom Telegram bot in my Pi under my name "Thiyraash " and I able to communicate with my Pi by sending them messages and commands. The message that I send to my Pi's bot using my Telegram smartphone application will first go to the Telegram Server then to the Telegram bot hosted in my Pi. The Pi will interpret the message and do the appropriate action based on the message. Below shows the activity diagram, flowchart and example of picture of this Telegram chat bot.



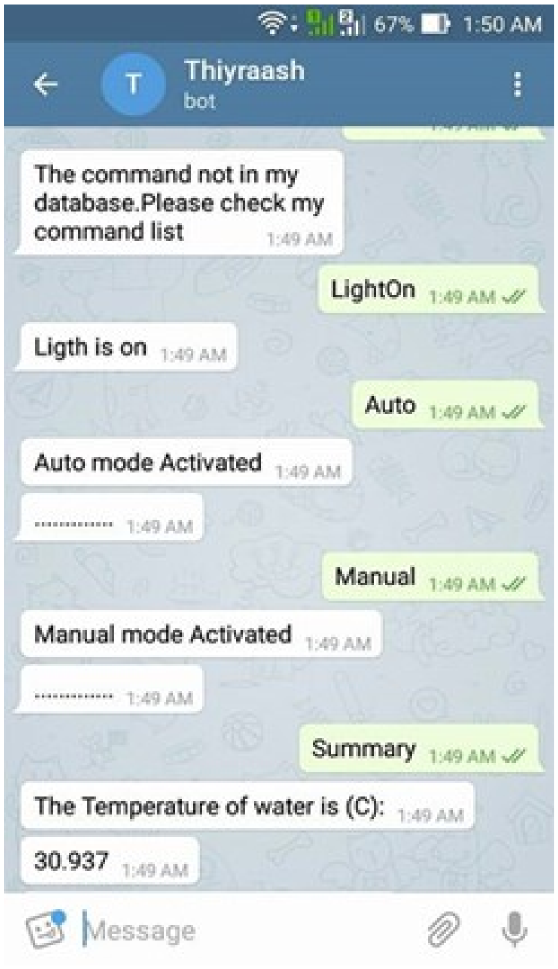
User
Telegram server
Telegram Chat bot in Pi
Send message to the Telegram chat bot in PI from Telegram apps
Secure and encrypted the message and send to the Telegram chat bot or vice versa based on particular access token of the bot.
Validate and interpret the message
Perform the appropriate action and reply to the user's message
Screen: Reply message to the user by Telegram bot

Figure 3-1-2-F5: Activity Diagram of Telegram chat bot mechanism works



Power on Raspberry Pi
Turn on all the external power source for actuator and sensor
Send message from Telegram apps from the smartphone to the Pi's telegram bot
Message send to the Telegram server then to Pi's Telegram bot
Pi will interpret the message then take appropriate action based on the message

Figure 3-1-2-F6: Simple Flowchart how to control the Pi using Telegram



The command not in my database.Please check my command list 1:49 AM

LightOn 1:49 AM

Ligth is on 1:49 AM

Auto 1:49 AM

Auto mode Activated 1:49 AM

.... 1:49 AM

Manual 1:49 AM

Manual mode Activated 1:49 AM

.... 1:49 AM

Summary 1:49 AM

The Temperature of water is (C): 1:49 AM

30.937 1:49 AM

Figure 3-1-2-F7: Example of command send from my smartphone to Pi's Telegram bot

<u>Voice Control Application (Snowboy)</u>

The third way to control this aquarium system by using Voice control application. This voice control application is run based on the Snowboy hotword detection engine service and it is fully offline. The detailed description about the Snowboy and its installation process are discussed in Chapter 5 Technologies and Tools Involved. Below shows the activity diagram of how the Voice control application runs,

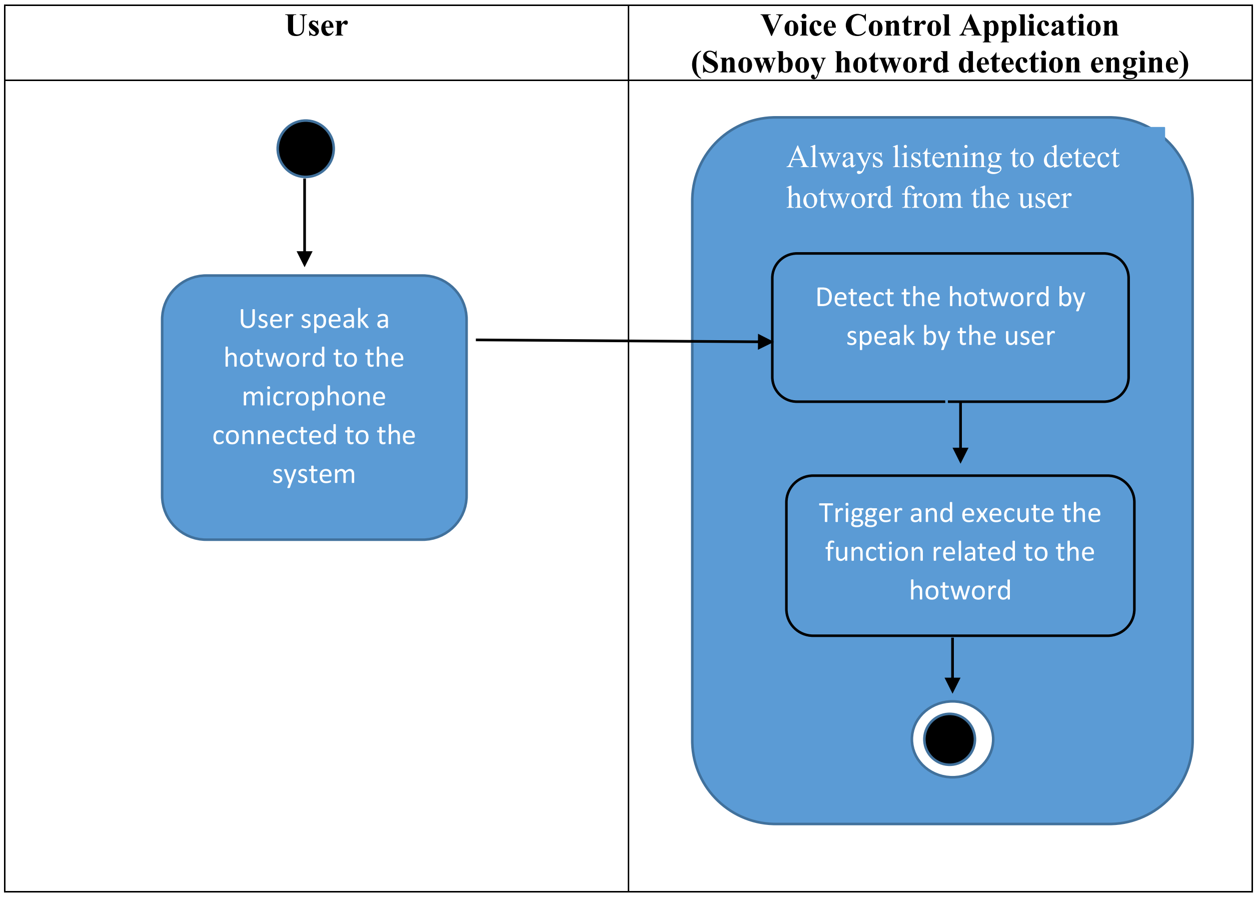

User
Voice Control Application (Snowboy hotword detection engine)
User speak a hotword to the microphone connected to the system
Always listening to detect hotword from the user
Detect the hotword by speak by the user
Trigger and execute the function related to the hotword

Figure 3-1-2-F8: Activity diagram on how the hotword detection engine works

The three control mode of this system that is the website, Telegram chat bot and Voice control application run in an individual file and not as single file. Since all of them running on different files, it is very hard to share variable among them for example, if the user change the system from manual mode to auto mode through website, Telegram chat bot and Voice command application should know that user have change the system mode to auto. However this feature is very difficult to achieve because global variable across files does not work properly. Hence MySQL databases is chosen to solve this problem. By default, MySQL databases is used in this project to store the values of sensors time to time. But now it can be used as communication tools between this three different files.

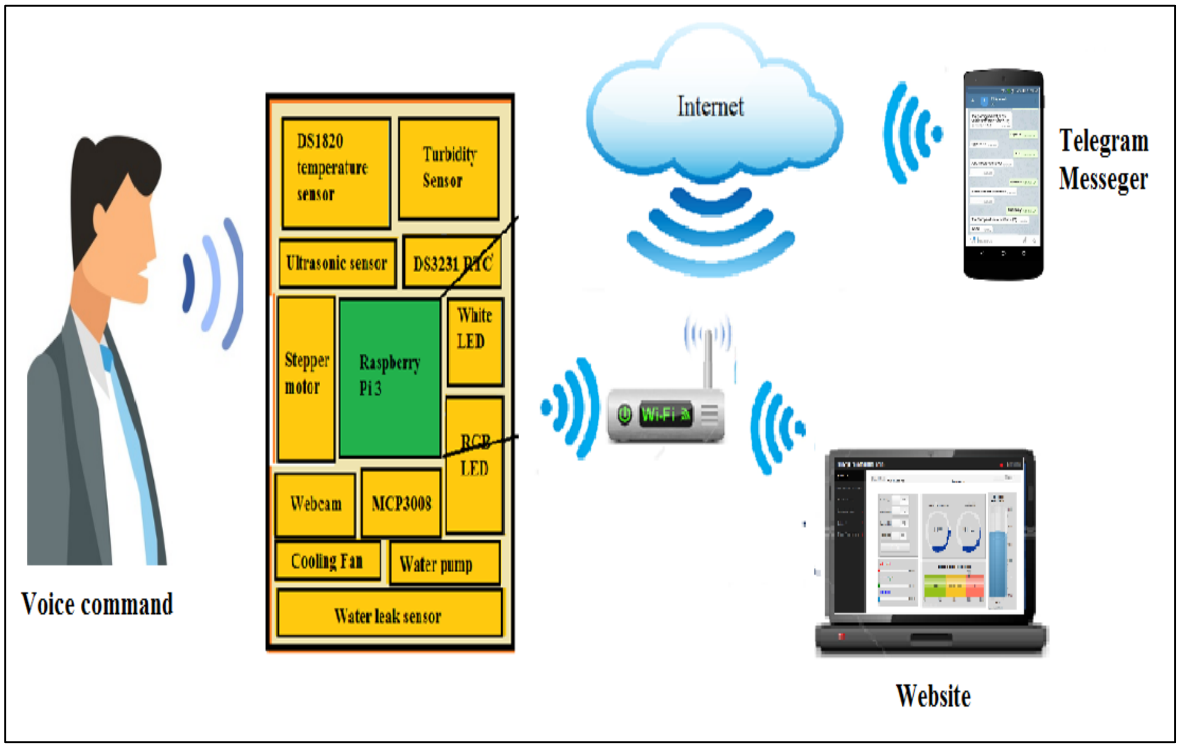


Figure 3-1-2-F9: Overall block diagram of the system with the control modes

Chapter 4: Methodology

4.1 Design Specifications

Selecting a proper design methodology is very essential in order to produce high functionalities system. Before that, it is important to know what **Software Development Life Cycle (SDLC)** is. **Software Development Life Cycle** (SDLC) is essentially a series of steps, or phases that provided a model for the development and lifecycle management of an application or system. The intent of an SDLC process is to help produce a product that is cost-efficient, effective and of high quality. There are many types of models under SDLC, but in this project, I'm using the Prototyping Model to model my project. Prototyping model is developed based on current requirements. The main idea behind Prototyping Model is that in this SDLC model, the user's requirements will be freeze before the design or coding phase can proceed. Temporary prototype is built to understand the customer product requirements and this temporary prototype will keep updating based on the customer's requirement. This model follows such method so that at the end of the project a well function system can be produce with minimal risk of fail.

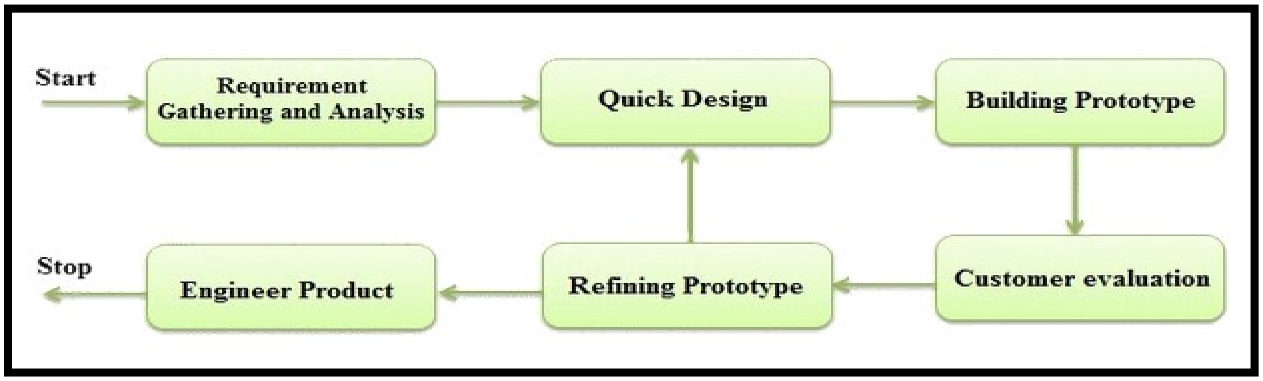


Figure 4-1-F1 Prototyping Model

(Picture taken from http://er.yuvayana.org/sdlc-prototype-model-design-advantages-disadvantages-and-applications/)

Following phases below representing the different phases of Prototyping Model:

- **Requirement gathering and analysis:** In this phase, the fundamental user's requirement is collect and accumulated during this phase. After complete understanding the user demand, the product requirement documents are prepared.
- **Quick Design:** The underlying model of the framework is improvement, for example, essential prerequisite and user interfaces are provided during this phase. In this quick design, the features used in this design won't not work precisely in the same way as the genuine outline. But its overall working environment is more similar to the final development.
- **Building prototypes:** The prototype model takes quick design from the output of the quick design and builds the system design locally. This local prototype gives a look and similar feel as the final product has to be designed.
- **Customer evaluation:** After building the prototype of system, the prototype will be send to user for evaluation to check whether all function of the system works properly as the requirement. The user needs to give feedback to developers so that the developer can further improve the product.
- **Refining product:** In this phase, the user feedback is examined and if there is any issues in the model, the developer will attempt to solve the issue and satisfy the user necessity by loop back to quick design phase.
- **Engineer product:** After all successive feedback and positive review form the user, the actual product is designed and developed.

The main reason I choose Prototype model as the modeling method in this project because of the flexibility of this model, which is the ability to loop back to previous phase if there is any bug or error in the system that need to be resolved. This is very important because we can’t build a perfect system within few tries. There will be bug in the system/product and we should able to solve it before proceed to the Engineer product phase that is last stage of the model. Besides that, the interaction between the developer and user in the customer evaluation phase helps the developer to create better user friendly system or product because the user’s feedback about the system/product is always welcome and examined. Giving importance to user’s requirement in developing a product will eventually bring success for the product. That is why the prototype model are been choose to model this project.

Chapter 5: Technologies and Tools Involved

5.0 Tools

There are two major parts involved in this project that is hardware and software component. Below table shows the overview of hardware components and software used in build this Aquarium Monitoring System.

Hardware component	Software component
Raspberry Pi	Python 2 (IDLE)
MCP3008 ADC Converter	MySQL(Database)
HC-SR04 Ultrasonic sensor	FLASK Web Server
Turbidity sensor	Hypertext Markup Language(HTML)
DS18B20 water temperature sensor	Cascading Style Sheets (CSS)
28BYJ48 Stepper motor and ULN2003 driver	JavaScript
SRD-05VDC-SL-C 4channel 5V Relay	UV4L Streaming Server
DS3231 RTC board	Snowboy (hotword detection engine)
5050 White LED Strip	Telegram Bot Application
3528 SMD RGB LED	---
3.5V~12V DC BRUSHLESS WATER PUMP	---
DIY COOLING FAN	---
LOGITECH HD WEBCAM C270	---

Table 5-T1: Tools

In the following sections, the specification, setup and connection, codes and the unit verification and testing of each components for hardware and software will be discussed. Specification part discuss about the components characteristics. Whereas, setup and connection will discuss about the installation process or physical connection of the components with the main board, Raspberry Pi .In code section, each component's code will be discuss and in unit verification and testing, each components will be test whether it work as expected.

<u>5.0.1 Hardware Components</u>

A) RASPBERRY PI 3 MODEL B

<u>Specification</u>

The development board that used in this project as central controller and processing unit is a single board computer called Raspberry Pi 3 Model B. It's called a low cost PC because an Operating System (OS) is needed by the board in order the board to function. The processor used in Raspberry Pi 3 is Broadcom BCM2837 SoC with 1.2 GHZ 64-bit quad-core ARM Cortex-A53 with 512KB shared L2 cache. Besides that, Raspberry Pi 3 Model B have 40-pin Extended GPIO and have built in WIFI and Bluetooth. Table 2 shows Raspberry Pi 3 Model basic specification.

Architecture	ARMv8-A (64/32-bit)
SoC	Broadcom BCM2837
CPU	1.2 GHz 64-bit quad-core ARM Cortex-A53
Memory (SDRAM)	1 GB
USB 2.0 ports	4
Video outputs	HDMI (rev 1.3), composite video (3.5 mm TRRS jack), MIPI display interface (DSI) for raw LCD panels
On-board storage	MicroSDHC slot, USB Boot Mode
On-board network	10/100 Mbit/s Ethernet, 802.11n wireless, Bluetooth 4.1
Pins	24 - GPIO pins 1 - Serial UARTs (RPi3 only includes mini UART) 2 - SPI bus 1 - I2C bus 2 - 5V power pins 2 - 3.3V power pins 8 - Ground pins
Power ratings	800 mA ,(4.0 W)
Power source	5V 2.5 A via MicroUSB
Size	85.60 mm × 56.5 mm × 17mm

Table 5-0-1-T1 : Specification

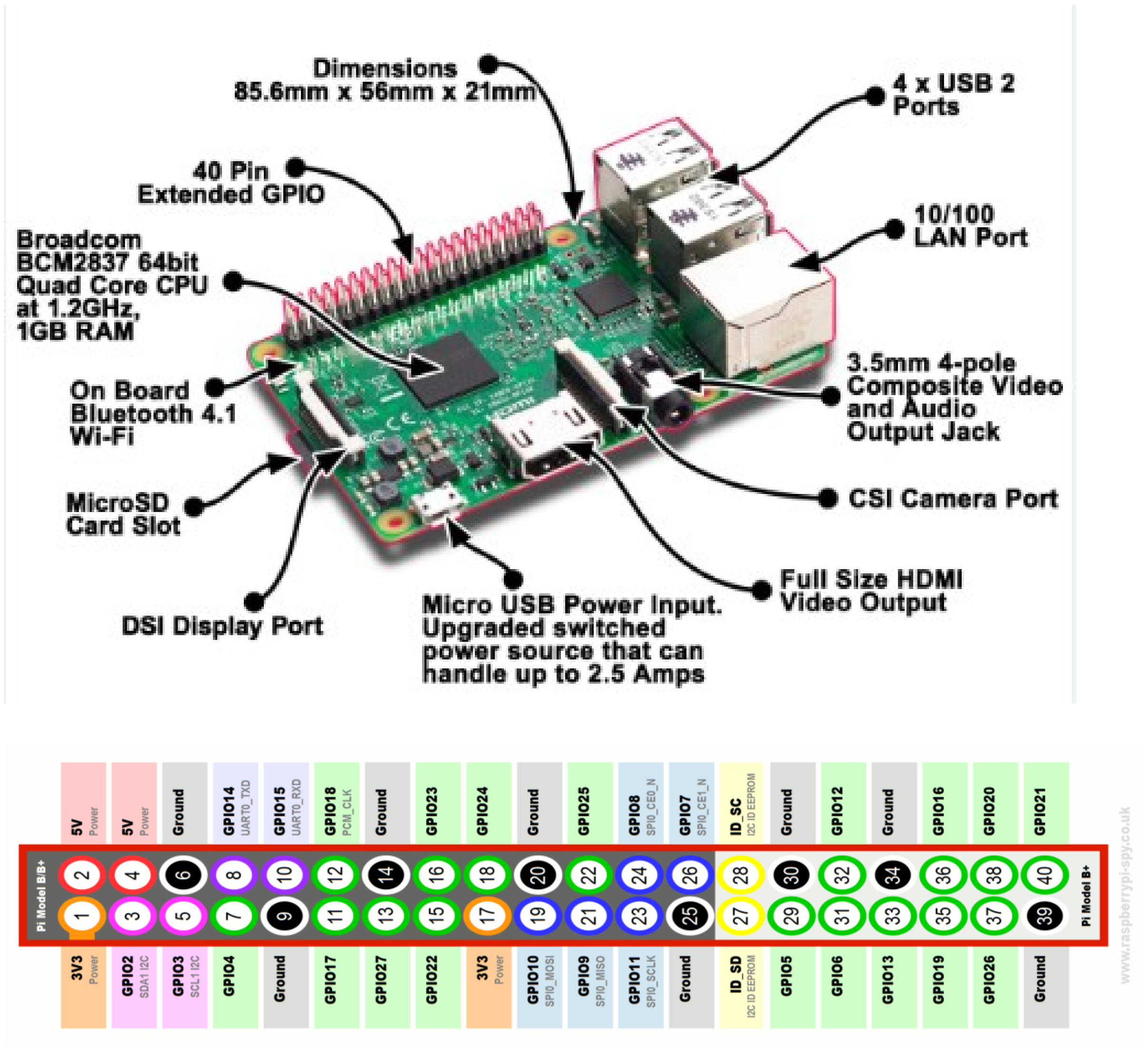


Figure 5-0-1-F1: Raspberry Pi 3 with Pins Specification

<u>Setup and connection</u>

Initially, the Raspberry Pi 3 that out of box does not have any operating system install in it. So, I download the Rasbian OS from Raspberry Pi official website and write the OS image on the Raspberry Pi 3's Micro SD card using Win32 Disk Imager. Then, I load the SD card to raspberry pi and power up it. For initial setup, an external keyboard, mouse and HDMI monitor is required because unlike Arduino, Raspberry pi cannot be control by just plugging a USB to laptop. Raspberry pi need an OS to function. After connect all the I/O to Raspberry pi, you will able to see the Raspbian boot up shown in the monitor. After successfully installed and login into Raspbian. I install VNC the virtual network computing to allow me to remotely control the Pi directly from my laptop with both of them connected under same network. With VNC, I does not need any external keyboard or mouse to control my Pi. Just by install VNC viewer in my laptop, by typing the Raspberry pi's local IP address, I will able to see Raspberry Pi desktop and control it from the VNC viewer from my laptop. Figure 5-0-1-F2 shows the output of the initial setup.

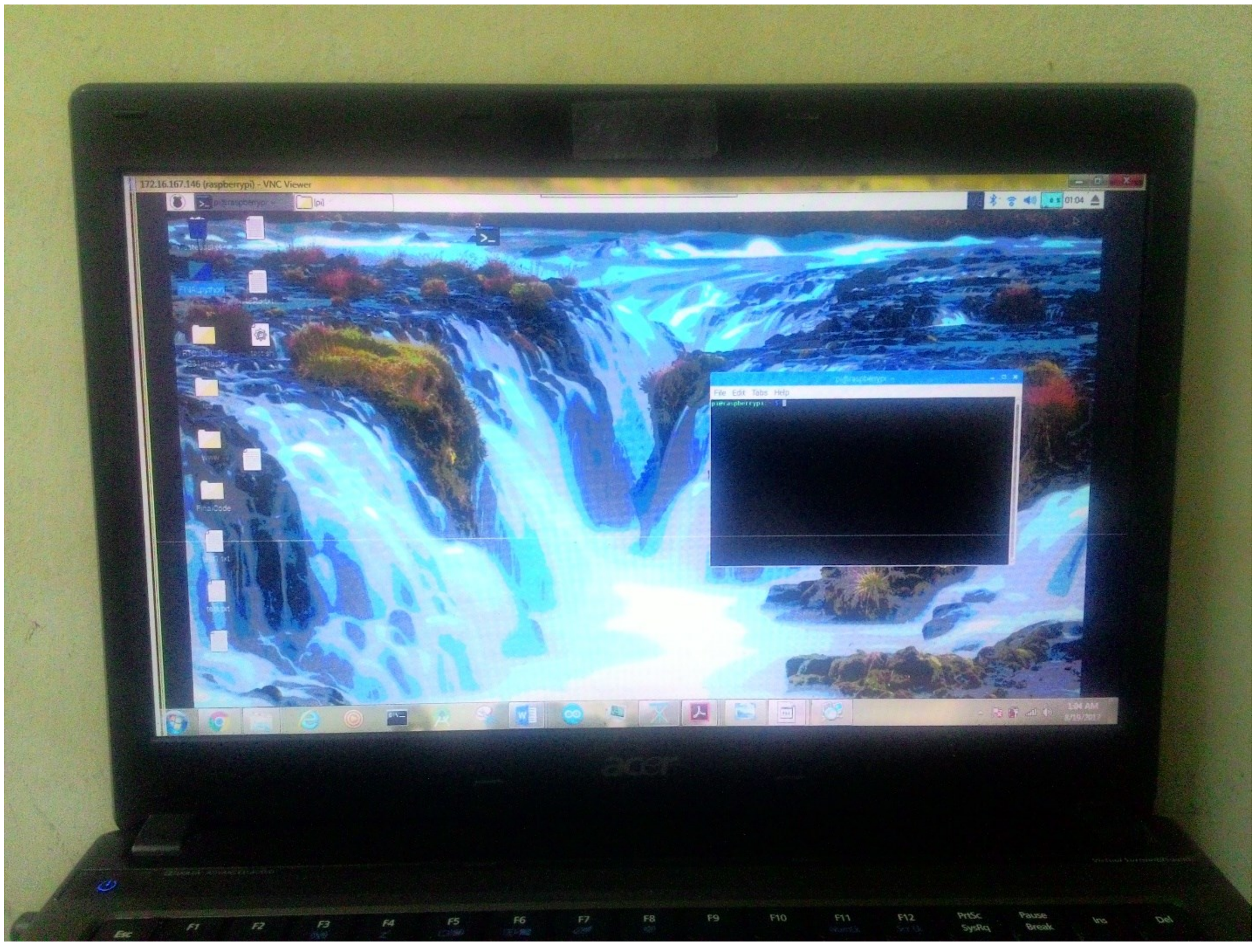


Figure 5-0-1-F2: Output of raspberry desktop at VNC viewer in my laptop

B) MCP3008 ADC CONVERTOR

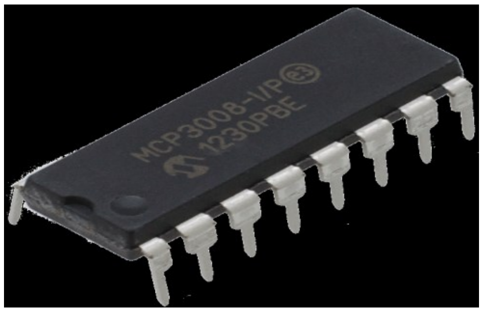


Figure 5-0-1-F3: MCP3008 chip

Although Raspberry Pi 3 Model B have many special capability in term of hardware, it does not have integrated analogue to digital converter GPIO. So an external ADC converter is needed to in order to perform such process. In this project I'm using a MCP3008 8-channel 10bit ADC chip in order to read analogue signal from analogue based sensors and convert them to digital values for the Pi. MCP3008 use **SPI for serial communication**. Table 3 shows the basic specification of MCP3008.

Operating Voltage	2.7 to 5.5 (V)	-
Operating Current	550 (mA)	Condition when, VDD = VREF = 5V, DOUT unloaded VDD = VREF = 2.7V, DOUT unloaded
Operating Temperature Range	-40 to +85 (°C)	--
High Level Input Voltage	0.7*V_dd	-
High Level Output Voltage	4.1 (V) Min	-

Table 5-0-1-T2: Specification of MCP3008

Connection of MCP3008 ADC converter with Raspberry Pi 3

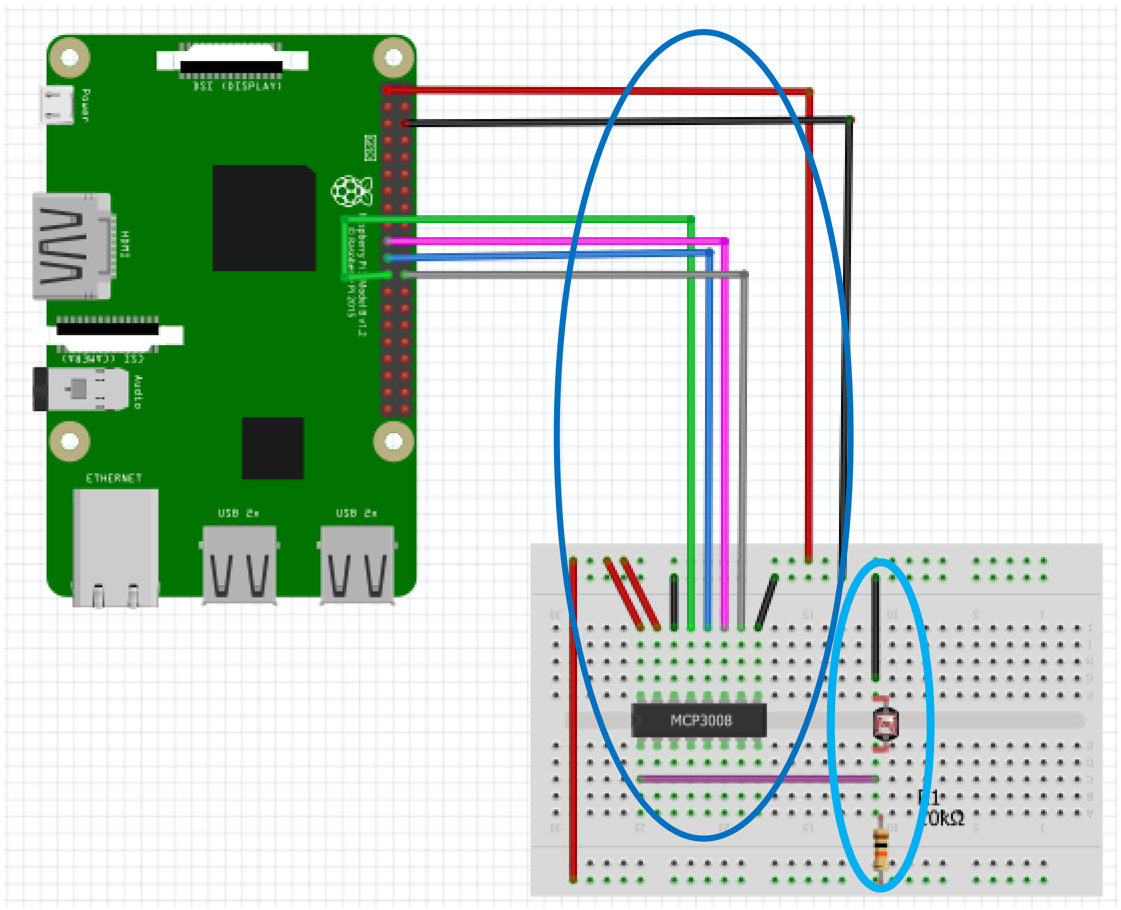


Figure 5-0-1 F4: Connection of MCP3008 ADC Convertor with LDR and Raspberry Pi 3

Figure 5-0-1-F4 shows the connection of MCP3008 ADC Convertor and the Raspberry Pi 3. As discussed in early section Raspberry pi does not have an integrated ADC and that's why an external ADC convertor MCP3008 is needed to read the analog sensors values such as Turbidity sensor or LDR sensor. The MCP3008 is power up using 3.3V from the Pi. According to the MCP3008's datasheet, the power supply to the MCP3008 V_dd should be equal to the voltage to be measure V_ref (V_dd = V_ref). This means that the maximum voltage of the signal V_ref that MCP3008 can read from the analog pin (CH) cannot be higher than the V_dd of itself. Figure and table below shows the MCP3008 pin description, the pin connection of MCP3008 and LDR sensor to Pi and the code used to read the analog sensor.

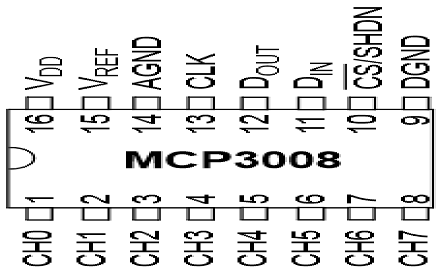
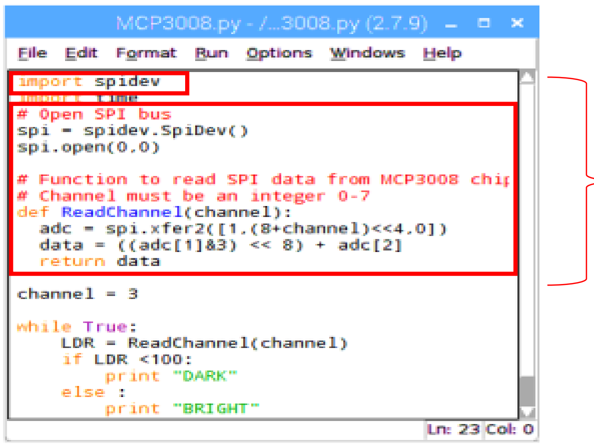


Figure 5-0-1-F5: Pin description of MCP3008

Raspberry Pi 3	**MCP3008**	**LDR sensor**
3.3V (Pin 1)	VDD and VREF (Pin 16 and 15)	Resistor with LDR
Ground (Pin 6)	AGAND and DGND (Pin 14 and 9)	Ground wire (Black wire)
GPI011 (Pin 23)	CLK(Pin 13)	--
GPI09 (Pin 21)	DOUT(Pin 12)	--
GPI010 (Pin 19)	DIN(Pin 11)	--
GPI08 (Pin 24)	CS/SHDN(Pin 10)	--
5V (Pin 2)	--	--
--	CH0 (Pin 1)	LDR data (Blue wire)

Table 5-0-1-T3: Pin description


MCP3008.py - /...3008.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 23 Col: 0

```
import spidev
import time
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0.0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
  adc = spi.xfer2([1.(8+channel)<<4.0])
  data = ((adc[1]&3) << 8) + adc[2]
  return data

channel = 3

while True:
    LDR = ReadChannel(channel)
    if LDR <100:
        print "DARK"
    else :
        print "BRIGHT"
```

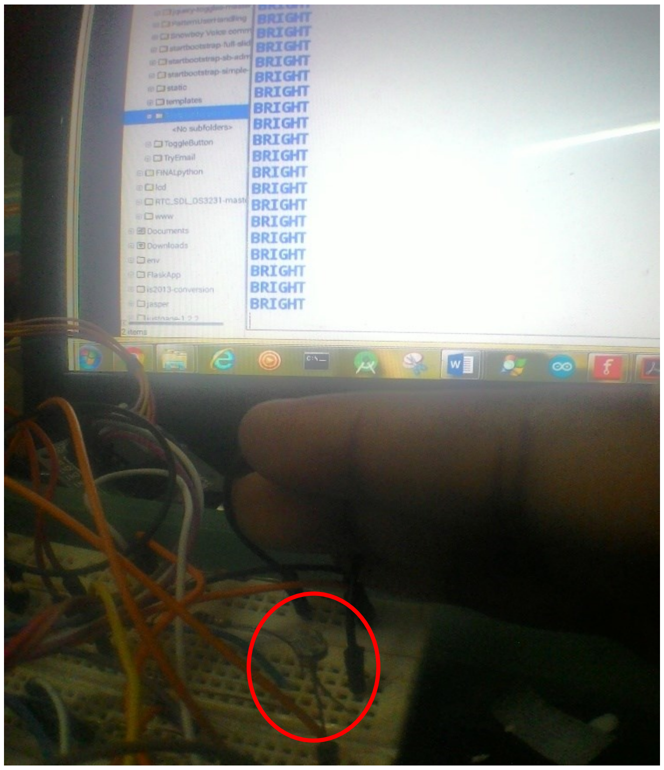
Important block of code for read analog value using MCP3008

Figure 5-0-1-F6: Code to read analog sensor data

In order to test whether the MCP3008 ADC converter is working properly, I have included a light dependent resistor (LDR) with 10k resistor as shows in Figure 5-F2 surrounded by second light blue circle. LDR sensor is used just for testing purpose of MCP3008 and it is not included in Aquarium Monitoring system. Below figure and table shows the test case and result of each case.

No	Test case	Actual result	Result
1	Does not close the LDR sensor with hand	The sensor value read should be more than 100 and the python shell should output "BRIGHT"	The sensor output as same as actual result. (**Figure5-F5**)
2	Close the LDR sensor with hand	The sensor value read should less than 100 and the python shell should output "DARK"	The sensor output as same as actual result. (**Figure5-F6**)

Table 5-0-1-T4: Test case



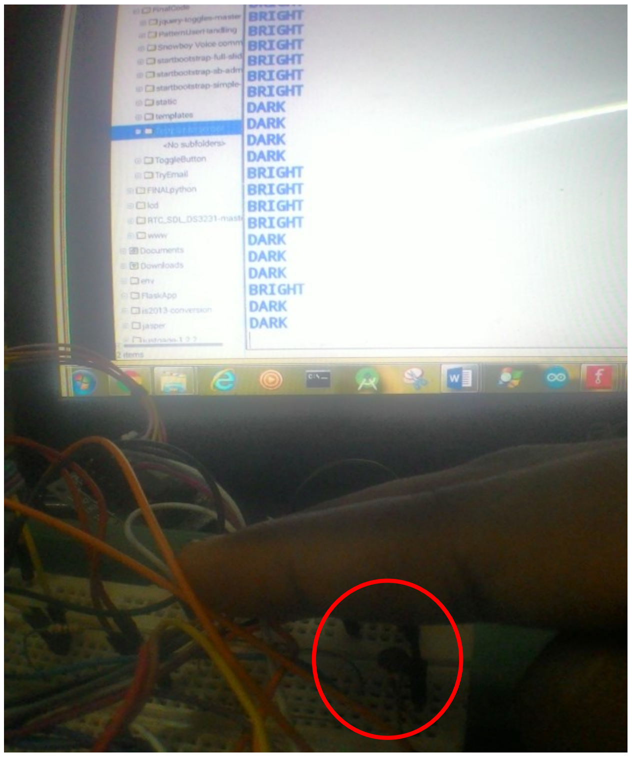


Figure 5-0-1-F7: Did not close sensor

Figure 5-0-1-F8: Close the sensor

C) HC-SR04 (ULTRASONIC RANGING SENSOR)

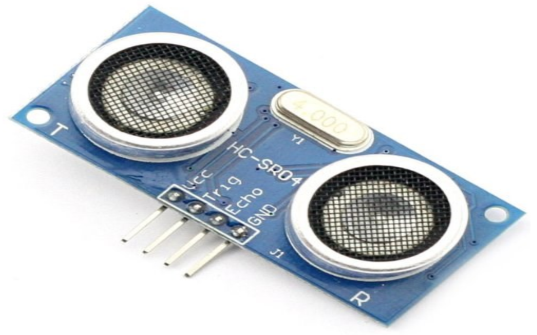


Figure 5-0-1-F9: HC-SR04 ultrasonic sensor

HC-SR04 is ultrasonic ranging sensor used to measure the distance to an object with high accuracy and stable readings. It consists of one ultrasonic transmitter, a receiver and control circuit. The transmitter will emits high frequency sound which bounce off any nearby solid object and the some of the sound will be reflected and detected by the receiver of the sensor. The emit signal and return signal will be proceed by the control circuit in order to calculate the time different between them. Than using some simple formula, the distance between the sensor and reflected object will be calculated. In this project HC-SR04 will be used to measure the water level of aquarium. Table 5 shows the basic specification of HC-SR04.

Operating Voltage	5.0 (V) DC
Working Current	15(mA)
Ultrasonic Frequency	40 kHz
Max Range	4 (m)
Min Range	2 cm
Measuring Angle	15 degree
Trigger Input Signal	10uS TTL pulse

Table 5-0-1-T5: HC-SR04.

Connection of Ultrasonic sensor (HC-SR04) and Raspberry Pi 3

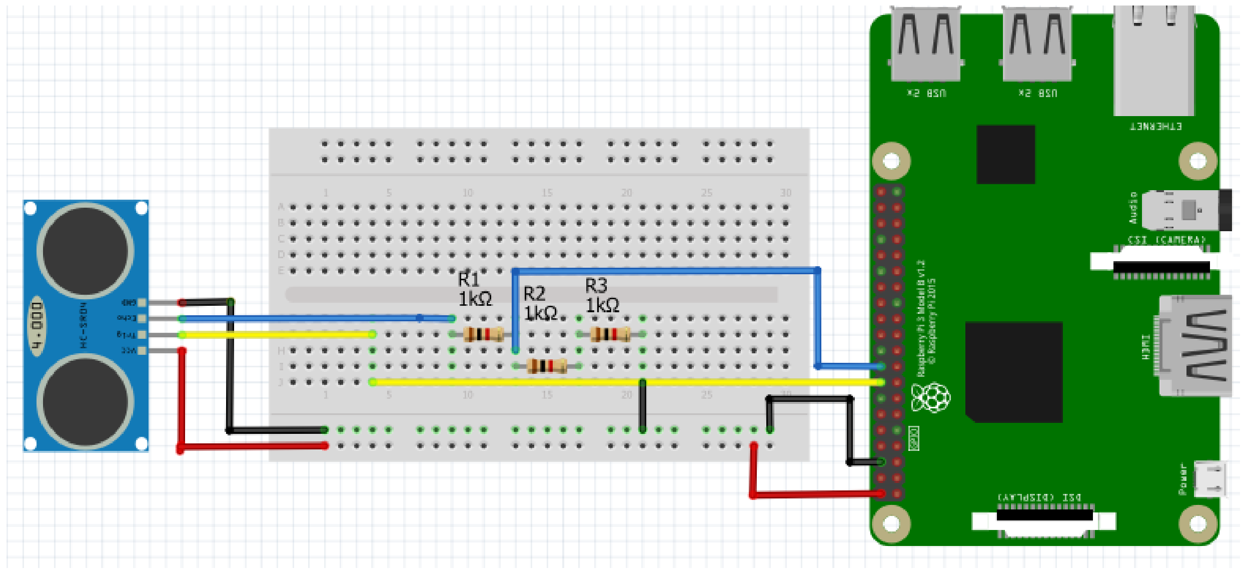


Figure 5-0-1-F10: Connection between HC-SR04 sensor and Raspberry Pi 3

Figure 5-0-1-F10 shows the connection between HC-SR04 sensor and Raspberry Pi 3.There are four pins in following sensor that is **Vcc, Trig, Echo** and **Gnd** pin. The basic principle of how the HC-SR04 works is

1. Trigger for at least 10us high level signal
2. Then the module automatically send eight 40kHz and check whether there is any pulse signal back
3. If any signal is back through high level, the time of high output IO duration is the time from sending ultrasonic to return
4. Lastly the distance is calculate using the formula **(high level time×velocity of sound (340M/S) / 2**

Table below shows the pin connection between HC-SR04 and Raspberry Pi 3

Raspberry Pi 3 pin	Ultrasonic Sensor(HC-SR04) Pins
5v pin (Pin2)	VCC
Ground(Pin 6)	GND
GPI023 (Pin 16)	Trig
GPI024 (Pin 18)	Echo

Table 5-0-1-T6: Pin connection

The Echo output signal of HC-SR04 is rated at 5V. But the Pi's GPI0 is only can accept input voltage up to 3.3V. If we send 5V signal to unprotected 3.3V input port of Pi, this could damage the Pi's GPIO pins. Hence by using voltage divider rule circuit we can reduce the voltage of HC-SR04 from 5V to 3.3V when input to Pi's GPI0. To find the appropriate resistor to output the desired voltage the following equation been used $\boldsymbol{Vout = Vin\ x\ (\frac{R4}{R1+R4})}$ .Two resistor **(R1 and R4 = (R2+R3))** with the value 1kΩ and 2kΩ are use and connected in series with an input voltage (Vin) which need to reduce to output voltage (Vout). Vin will be the ECHO with 5v and Vout will be 3.3V as shown in Figure 5-F7.After successfully reduce the voltage, now the Echo voltage can be safety input to the Pi's GPIO.

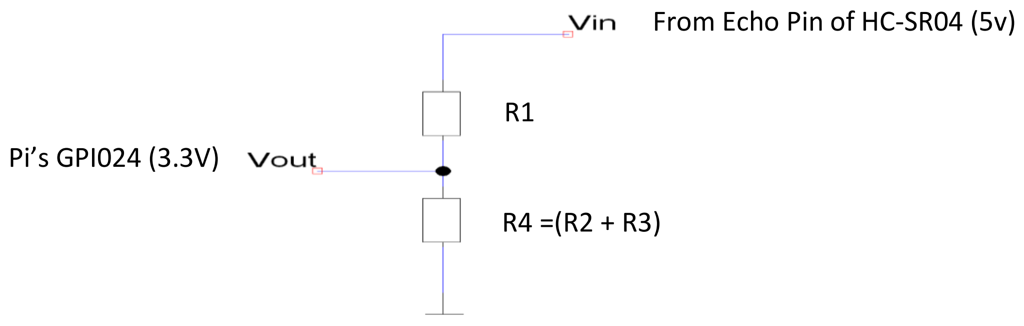
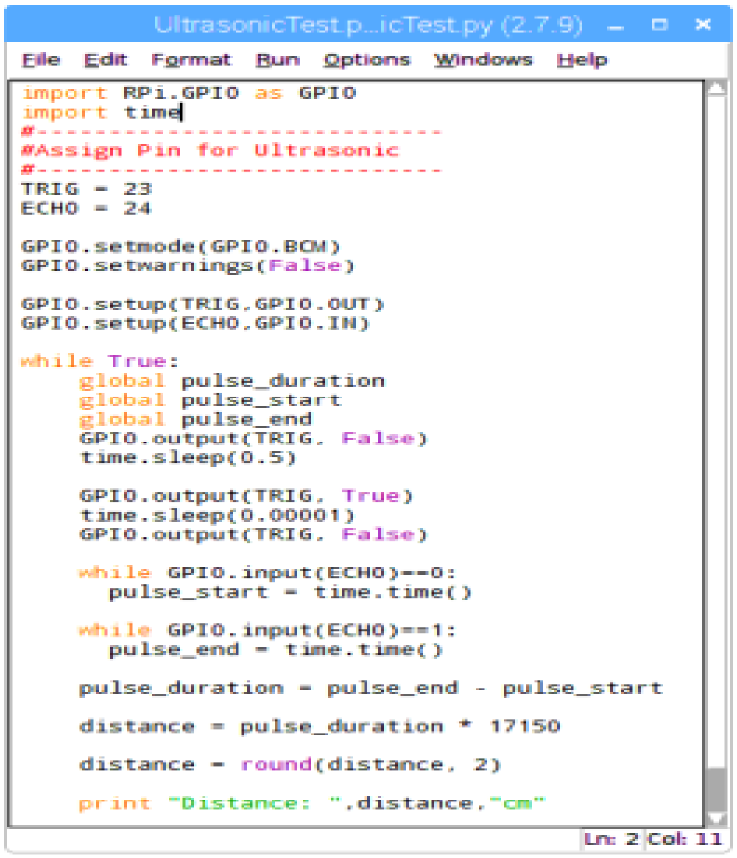


Figure 5-0-1-F11: Resistors connection


UltrasonicTest.p...icTest.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 2 Col: 11

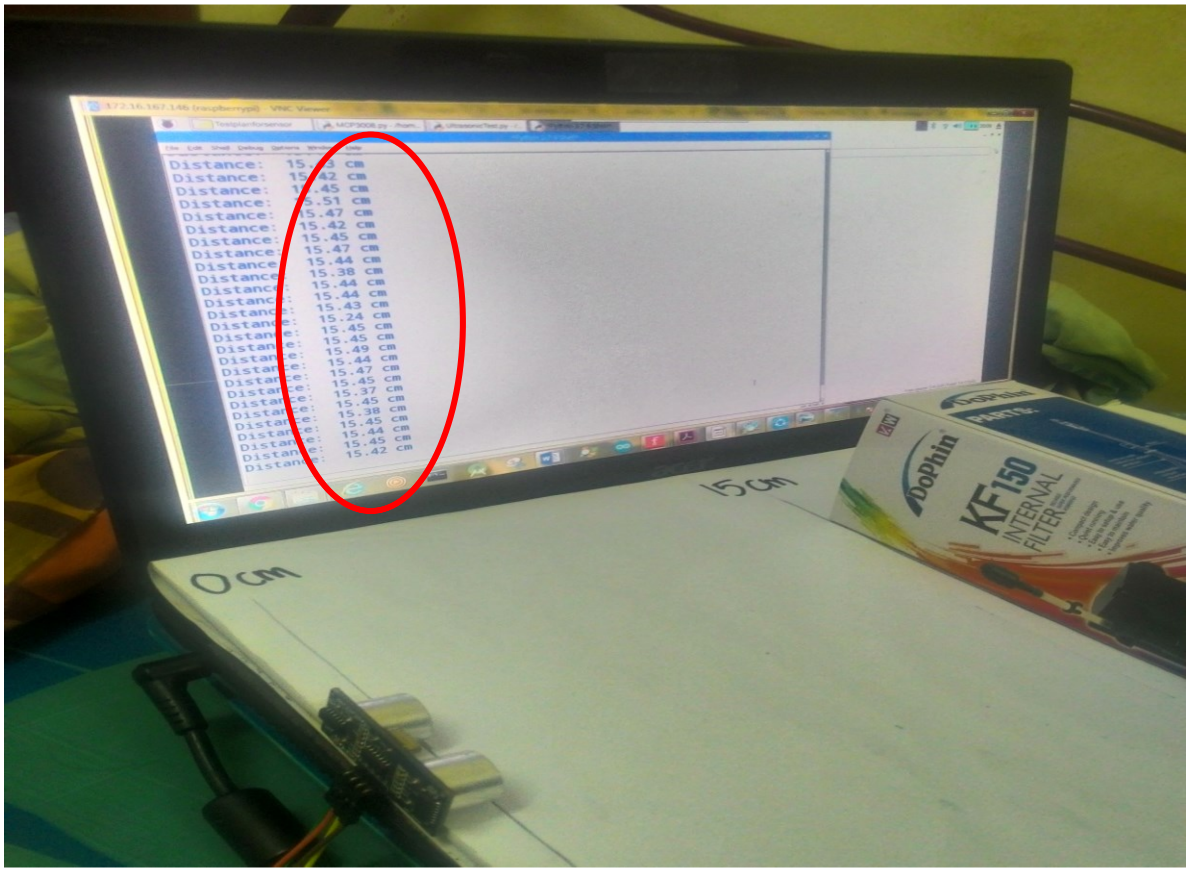
```
import RPi.GPIO as GPIO
import time
#-----------------------------
#Assign Pin for Ultrasonic
#-----------------------------
TRIG = 23
ECHO = 24

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

while True:
    global pulse_duration
    global pulse_start
    global pulse_end
    GPIO.output(TRIG, False)
    time.sleep(0.5)

    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)

    while GPIO.input(ECHO)==0:
      pulse_start = time.time()

    while GPIO.input(ECHO)==1:
      pulse_end = time.time()

    pulse_duration = pulse_end - pulse_start

    distance = pulse_duration * 17150

    distance = round(distance, 2)

    print "Distance: ",distance,"cm"
```

Figure 5-0-1-F12: Code for Ultrasonic Sensor

Several test case has been define in order to test whether the ultrasonic sensor is working as desired. I marked 3 lines on piece of drawing paper with distance 15cm between the first and second line followed by 10cm distance between second and third line. Then I place the ultrasonic sensor at the first line and a box at second line and third line just to test whether the ultrasonic sensor will output the same or approximately same distance value at python shell at VNC viewer in my laptop with the marked distance. Below table shows the test case, actual result and the result obtain from the test.

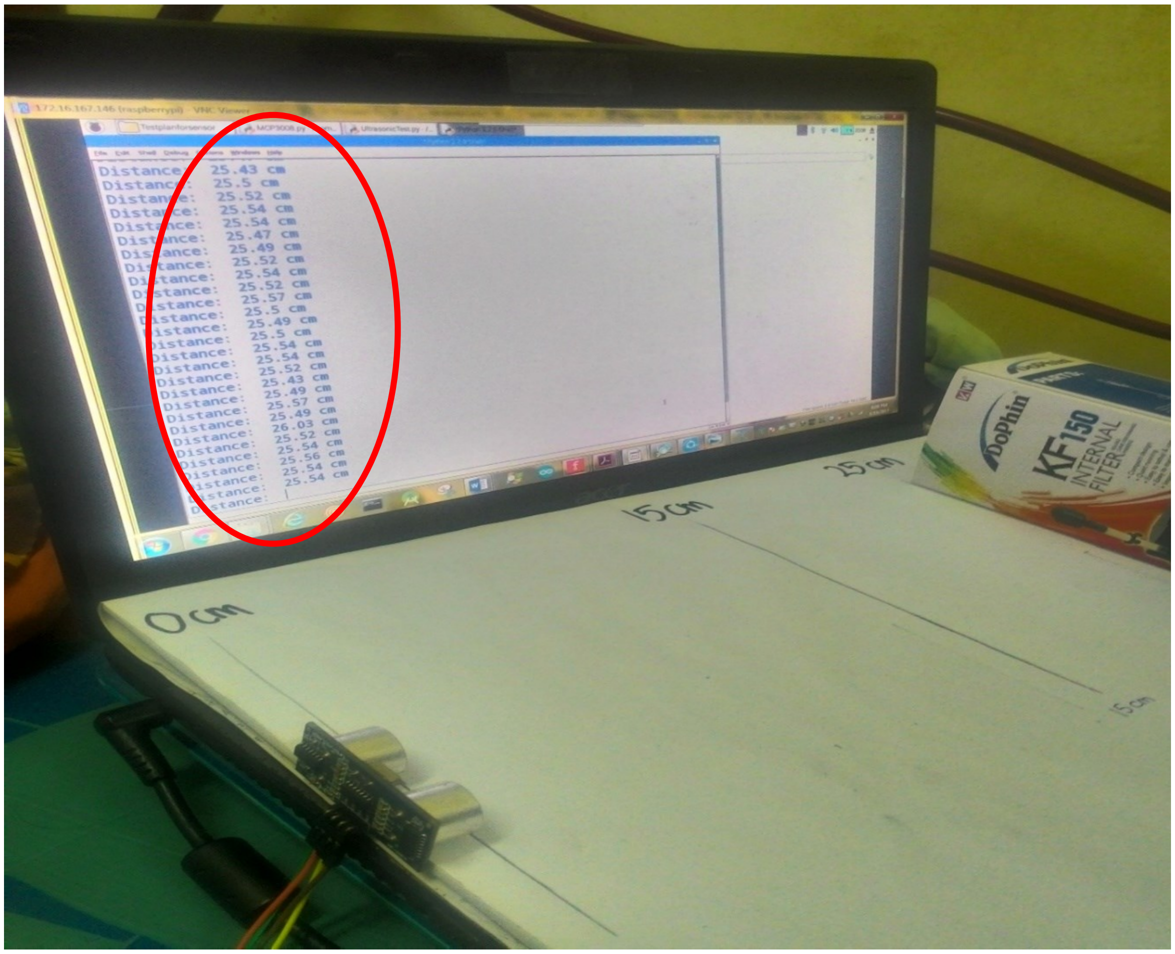
No	Test case	Actual result	Result
1	Place the box at 15cm distance from first line(ultrasonic been placed)	The sensor should output approximately same value as 15cm at python shell on VNC viewer.	The sensor output approximately same value as actual result with percent error of 1.8%. **(Figure 5-F10)**
2	Place the box at 25cm distance from first line(ultrasonic been placed)	The sensor should output approximately same value as 25cm at python shell on VNC viewer.	The sensor output approximately same value as actual result with percent error of 1.8%. **(Figure 5-F11)**

Table 5-0-1-T7: Test plan for HC-SR04



```
Distance:  15.43 cm
Distance:  15.42 cm
Distance:  15.45 cm
Distance:  15.51 cm
Distance:  15.47 cm
Distance:  15.42 cm
Distance:  15.45 cm
Distance:  15.47 cm
Distance:  15.44 cm
Distance:  15.38 cm
Distance:  15.44 cm
Distance:  15.44 cm
Distance:  15.43 cm
Distance:  15.24 cm
Distance:  15.45 cm
Distance:  15.45 cm
Distance:  15.49 cm
Distance:  15.44 cm
Distance:  15.47 cm
Distance:  15.45 cm
Distance:  15.37 cm
Distance:  15.45 cm
Distance:  15.38 cm
Distance:  15.45 cm
Distance:  15.44 cm
Distance:  15.45 cm
Distance:  15.42 cm
Distance:
```

Figure 5-0-1-F13: Box placed at 15cm from ultrasonic sensor



```
Distance:   25.43 cm
Distance:   25.5 cm
Distance:   25.52 cm
Distance:   25.54 cm
Distance:   25.54 cm
Distance:   25.47 cm
Distance:   25.49 cm
Distance:   25.52 cm
Distance:   25.54 cm
Distance:   25.52 cm
Distance:   25.57 cm
Distance:   25.5 cm
Distance:   25.49 cm
Distance:   25.5 cm
Distance:   25.54 cm
Distance:   25.54 cm
Distance:   25.52 cm
Distance:   25.43 cm
Distance:   25.49 cm
Distance:   25.57 cm
Distance:   25.49 cm
Distance:   26.03 cm
Distance:   25.52 cm
Distance:   25.54 cm
Distance:   25.56 cm
Distance:   25.54 cm
Distance:   25.54 cm
Distance:
Distance:
```

Figure 5-0-1-F14: Box placed at 25cm from ultrasonic sensor

In this aquarium monitoring system project, ultrasonic sensor is used to measure the water level of aquarium. In order to measure the water level of aquarium, first we need the height of our aquarium. So, in the web, the user have to key in their aquarium's dimension **(Length x Width x Height) in cm** first before set up other things. In this project I used an aquarium with the height 18.8cm. Below shows simple algorithm on how the HC-SR04 measure the water level of aquarium.

AquariumHeight = 18.8 (Defined by user)

OptimumWaterLevel = 18.8/(100/18) =3.384cm

HalfWaterLevel= 18.8/(100/50)=9.4 cm

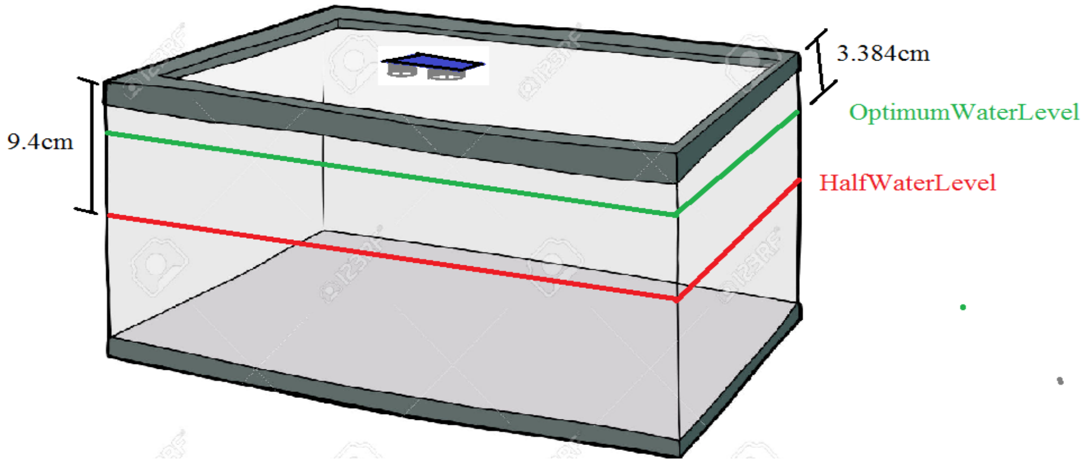


Figure 5-0-1-F15: Illustration of aquarium

If the ultrasonic distance value is about 3.384cm, the Pi will conclude that the 82% of the tank is been fill up and the water level of aquarium is in optimum level. Whereas if the ultrasonic sensor distance value is about 9.4cm, the Pi will conclude that only 50% of the water in the aquarium is left. Anything less than the half water level or more than optimum water level will trigger the water pump to pump in or out the water in order to maintain the water level if the system is in auto mode.

D) TURBIDITY SENSOR

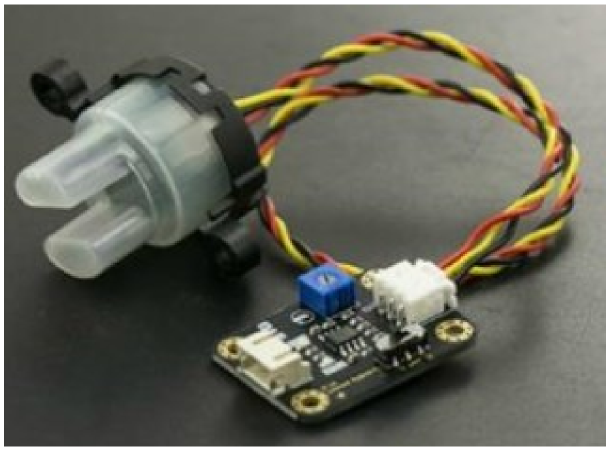


Figure 5-0-1-F16: Turbidity sensor

This turbidity sensor measure the quality of the water based on the water cloudiness. It able to detect the suspended particles in water by measuring the light transmittance and scattering rate. This scattering rate will changes with the amount of total suspended solids (TSS) in water and as the total suspended solids in water increases, the turbidity level also decreases. On other words we can say that turbidity value is inversely proportional to TTS. The turbidity sensor come along with the adapter which provides two output mode analog and digital signal. Table 5-T8 shows the basic specification of Turbidity sensor.

Operating voltage	5V DC
Operating Current	40mA (MAX)
Response Time	<500ms
Output method	• Analog output: 0-4.5V • Digital Output: High/Low level signal (Adjustable using potentiometer)
Operating Temperature	5℃~90℃
Adapter Dimensions	38mm*28mm*10mm

Table 5-0-1-T8: Specification of Turbidity sensor

Connection of Turbidity Sensor with Raspberry Pi 3

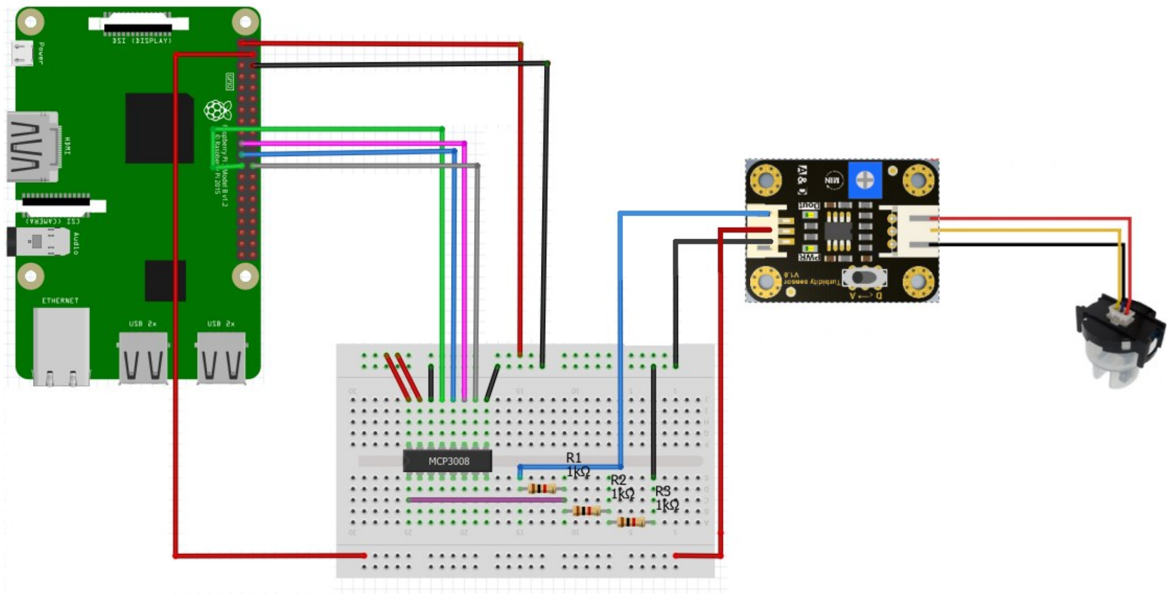


Figure 5-0-1-F17: Connection of Turbidity sensor with MCP3008 ADC converter and the Raspberry Pi 3

Figure 5-0-1-F16 shows the connection of Turbidity sensor the MCP3008 ADC converter and the Raspberry Pi 3. As discussed in earlier section, turbidity sensor is used to measure the cloudiness of water by outputting analog value to its data line (Blue wire) with the operating voltage of 5V. The MCP3008 chip is used to convert the analog signal from turbidity sensor to the Pi since Pi does not have integrated ADC converter. The MCP3008 is power up using 3.3V from the Pi. According to the MCP3008's datasheet, the power supply to the MCP3008 V_dd should be equal to the voltage to be measure V_ref **(V_dd = V_ref).** This means that the maximum voltage of the signal **V_ref** that MCP3008 can read from the analog pin (CH) cannot be higher than the **V_dd** of itself. If **V_ref > V_dd** it might damage the chip. To resolve the problem, I connect the V_dd and V_ref of MCP3008 to Pi's 3.3V and power the turbidity sensor with Pi's 5V pin. Then I feed the turbidity sensor output (Blue wire) into a voltage divider circuit with 3 resistor (Same as Ultrasonic sensor's Echo line setup) which reduce the output voltage from 5V to 3.3Vvoltage the desired voltage of MCP3008. By using this method I able to measure the turbidity of the water without damaging the MCP3008 and the Pi. Figure and table below shows the MCP3008 pin description and the pin connection of MCP3008 and Turbidity sensor to Pi.

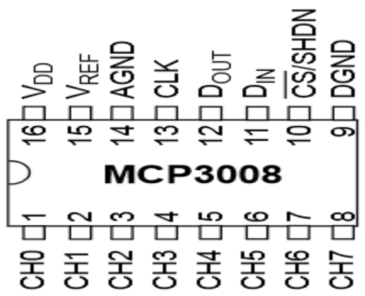
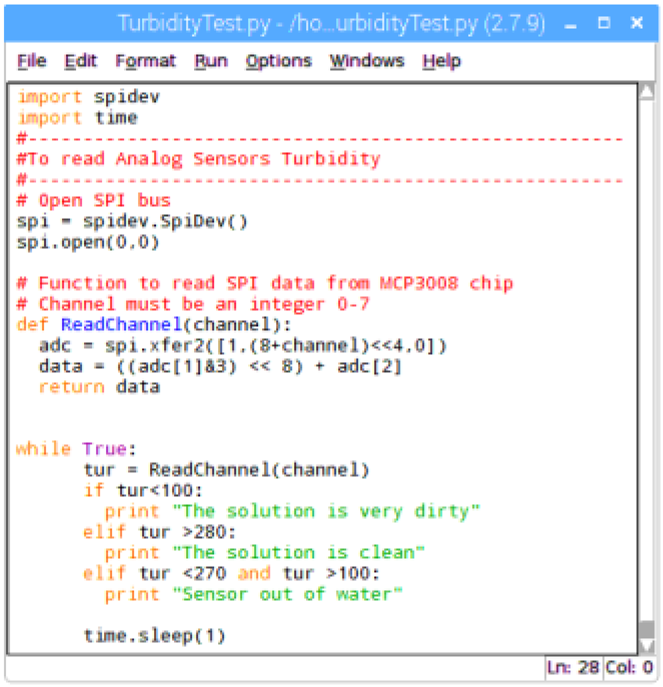


Figure 5-0-1-F18: Pin description of MCP3008

Raspberry Pi 3	MCP3008	Turbidity sensor
3.3V (Pin 1)	VDD and VREF (Pin 16 and 15)	--
Ground (Pin 6)	AGAND and DGND (Pin 14 and 9)	Ground wire (Black wire)
GPI011 (Pin 23)	CLK(Pin 13)	--
GPI09 (Pin 21)	DOUT(Pin 12)	--
GPI010 (Pin 19)	DIN(Pin 11)	--
GPI08 (Pin 24)	CS/SHDN(Pin 10)	--
5V (Pin 2)	--	Power wire (Red wire)
--	CH0 (Pin 1)	Data line (Blue wire)

Table 5-0-1-T9: Pin description


TurbidityTest.py - /ho...urbidityTest.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 28 Col: 0

```
import spidev
import time
#-----------------------------------------------------
#To read Analog Sensors Turbidity
#-----------------------------------------------------
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
  adc = spi.xfer2([1,(8+channel)<<4,0])
  data = ((adc[1]&3) << 8) + adc[2]
  return data


while True:
      tur = ReadChannel(channel)
      if tur<100:
        print "The solution is very dirty"
      elif tur >280:
        print "The solution is clean"
      elif tur <270 and tur >100:
        print "Sensor out of water"

      time.sleep(1)
```

Figure 5-0-1-F19: Code for Turbidity sensor

To test whether turbidity sensor working properly, I prepared two paper cup fill with clean water and dirty water (Black water color) respectively. Then I label them with clean and dirty water and place the turbidity sensor and evaluate whether turbidity sensor able to detect which water is clean and which is not. This can be achieved by simple if else statement. If the turbidity value is less than 100 it considered dirty, if the value bigger than 280 it's considered clean water and if value is in between 100 and 270 considered the sensor is out of water. Below show the test case, the actual result and result obtain from the test.

No	Test case	Actual result	Result
1	Do not place the turbidity sensor value in any solution.	The sensor should output value in between 100 to 270 and should output "Sensor out of water" at the python shell in VNC viewer.	The sensor output approximately same value as actual result and do output the "Sensor out of water" **(Figure 5-F16)**
2	Place the turbidity sensor value in clean solution.	The sensor should output value in bigger than 280 and should output "The solution is clean" at the python shell in VNC viewer.	The sensor output approximately same value as actual result and do output the "The solution is clean" **(Figure 5-F17)**
3	Place the turbidity sensor value in dirty solution.	The sensor should output value in less than 100 and should output "The solution is dirty" at the python shell in VNC viewer.	The sensor output approximately same value as actual result and do output the "The solution is dirty" **(Figure 5-F18)**

Table 5-0-1-T10: Test plan for Turbidity sensor

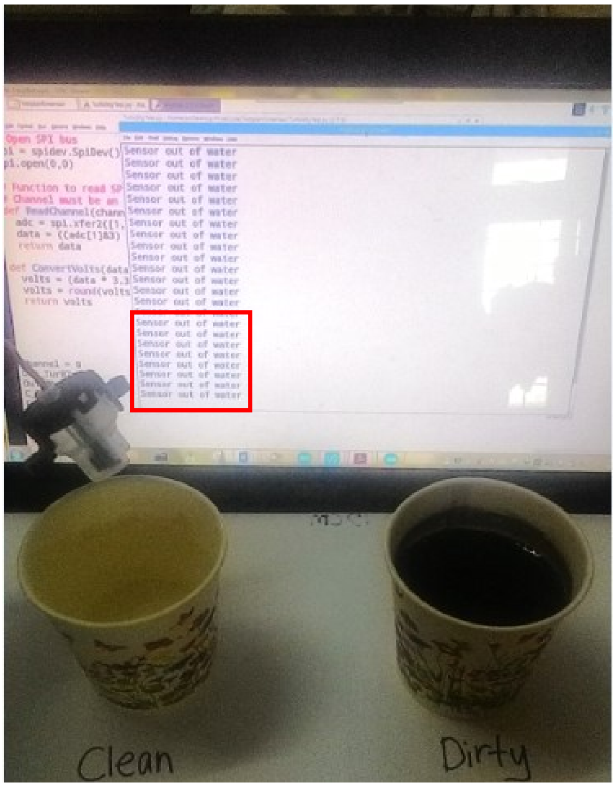
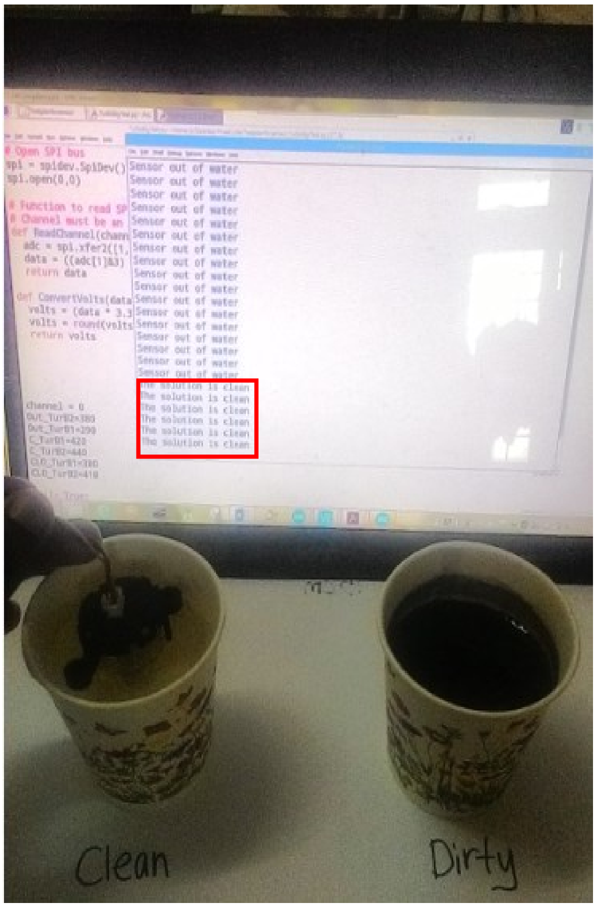


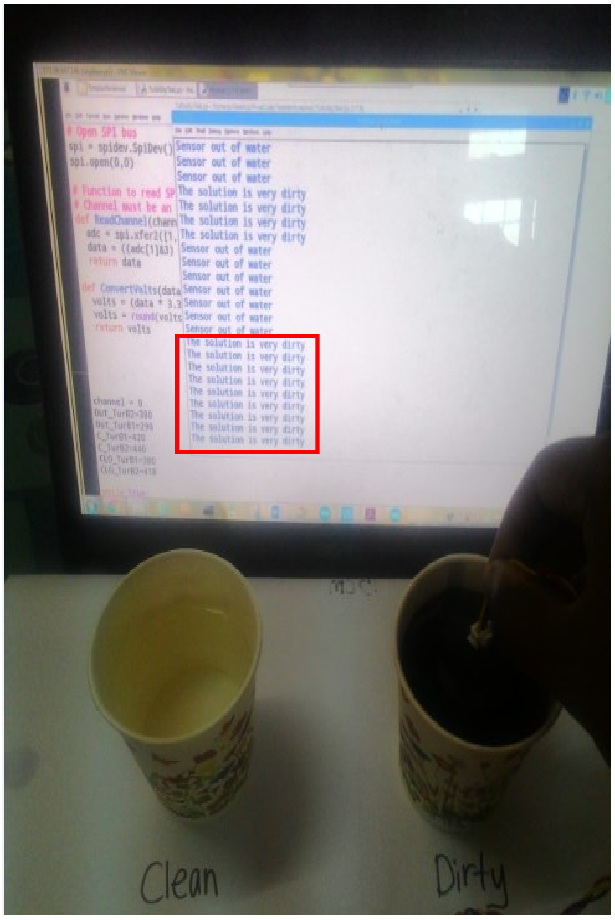
Figure 5-0-1-F20: Sensor out of water



```
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SP
# Channel must be an
def ReadChannel(chann
  adc = spi.xfer2([1,
  data = ((adc[1]&3)
  return data

def ConvertVolts(data
  volts = (data * 3.3
  volts = round(volts
  return volts


channel = 0
Out_Tur02=380
Out_Tur01=290
C_Tur01=420
C_Tur02=440
CLO_Tur01=380
CLO_Tur02=410
```

```
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
The solution is clean
The solution is clean
The solution is clean
The solution is clean
The solution is clean
The solution is clean
```

Figure 5-0-1-F21: The solution is clean



```
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SP
# Channel must be an
def ReadChannel(chann
  adc = spi.xfer2([1,
  data = ((adc[1]&3)
  return data

def ConvertVolts(data
  volts = (data * 3.3
  volts = round(volts
  return volts


channel = 0
Out_Tur02=380
Out_Tur01=290
C_Tur01=420
C_Tur02=440
CLO_Tur01=380
CLO_Tur02=410
```

```
Sensor out of water
Sensor out of water
Sensor out of water
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
Sensor out of water
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
The solution is very dirty
```

Figure 5-0-1-F22: The solution is dirty

In this project, turbidity sensor is used to detect and inform the user whether the aquarium water is dirty or not. It is also important in the automation part of the project where the system will automatic change the aquarium water if it is dirty, without the user interference if the turbidity sensor auto control is been enable by the user through the aquarium's website.

E) DS18B20 (WATERPROOF TEMPERATURE SENSOR)

Figure 5-0-1-F23: DS18B20

DS18B20 is a digital temperature sensor which accurately measure temperature of wet environment just by using a simple 1 –Wire interface. It able to provided 9 to 12 bit temperature reading over a 1 wire interface. Since its digital, there will be no problem in term of signal loss or degradation even over long distance. Table 5-T11 shows the basic specification of DS18B20 digital temperature sensor.

Operating Voltage range	3.0V to 5.5V
Operating temperature range:	-55°C to +125°C (-67F to +257F)
Accuracy over the range of -10°C to +85°C:	±0.5°C
Waterproof	Yes

Table 5-0-1-T11: Specification of DS18B20

<u>Connection of DS18B20 with Raspberry Pi 3</u>

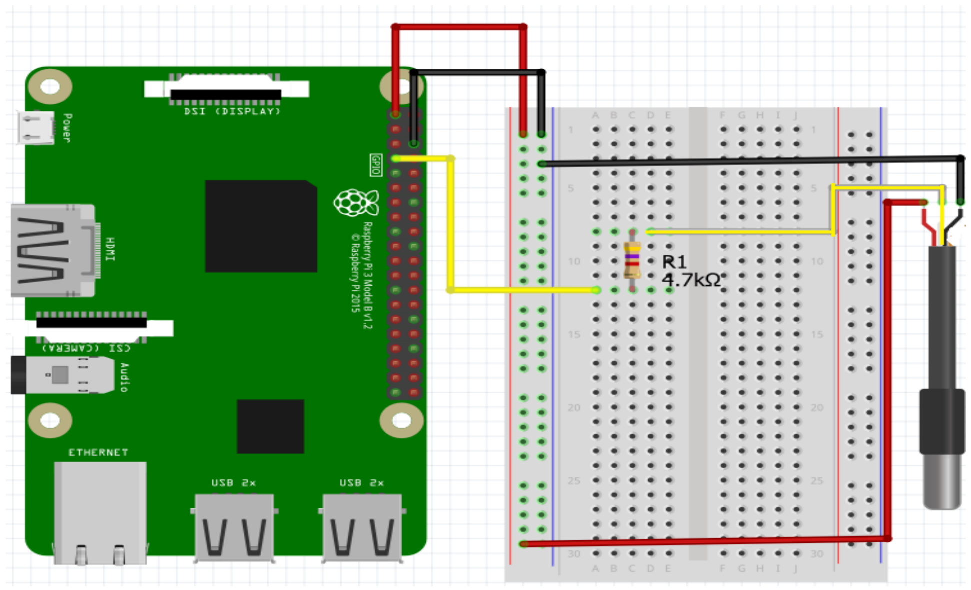
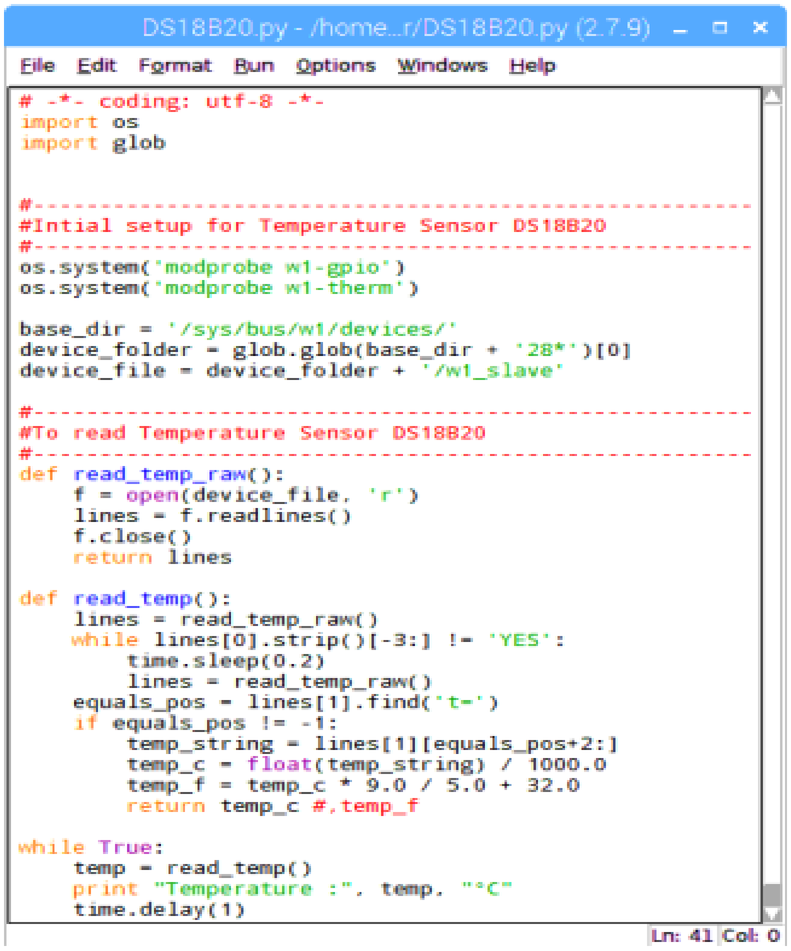


Figure 5-0-1-F24: Connection between 1-Wire DS18B20 sensor and the Raspberry

Figure 5-0-1-F24 shows the connection of 1-Wire DS18B20 waterproof temperature sensor with the Raspberry Pi 3. DS18B20 sensor have 3 pin 1 power line (Red), 1 GND (Black) and 1 DATA line (Yellow). The data line of DS18B20 is connect to a 4.7kΩ before feed into the Pi's GPI04, the dedicate pin for 1-Wire GPIO sensing because the resistor in this setup will act as 'pull-up' for the data line. It is used to ensure the 1-Wire data line is at defined logic level and reduced the electrical noise in case the GPIO pin is left floating. Table below shows the pin connection of DSB1820 with Pi.

Raspberry Pi 3 pin	**DS18B20**
3.3 V (Pin 1)	Power line (Red wire)
Ground (Pin 6)	Ground (Black wire)
GPI04 (Pin 7)-Dedicated pin for 1-Wire sensing	Data line (Yellow)

Table 5-0-1-T12: Pin description


DS18B20.py - /home...r/DS18B20.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 41 Col: 0

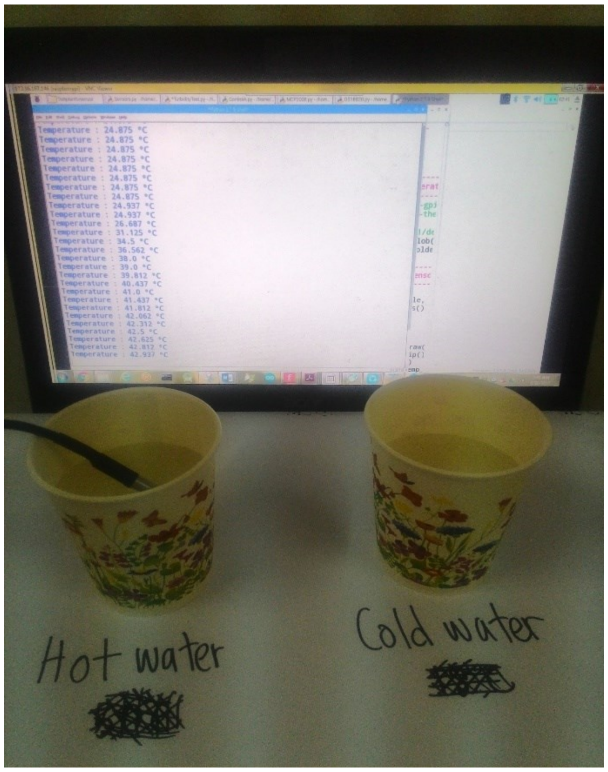
```
# -*- coding: utf-8 -*-
import os
import glob


#-------------------------------------------------------------
#Intial setup for Temperature Sensor DS18B20
#-------------------------------------------------------------
os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

#-------------------------------------------------------------
#To read Temperature Sensor DS18B20
#-------------------------------------------------------------
def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        temp_f = temp_c * 9.0 / 5.0 + 32.0
        return temp_c #,temp_f

while True:
    temp = read_temp()
    print "Temperature :", temp, "°C"
    time.delay(1)
```

Figure 5-0-1-F25: Code for of 1-Wire DS18B20 waterproof temperature sensor

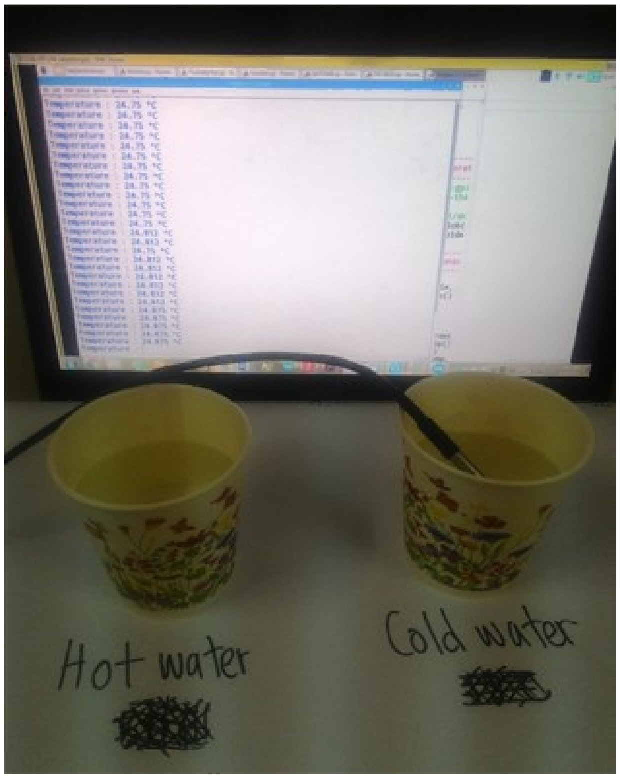
To test whether DS18B20 waterproof temperature sensor working properly, few test case has been defined. Two paper cup are fill with hot water and cold water respectively. The value of the hot water and cold water are measured using a digital thermometer and those values were used as reference to compared with the value obtained from 1-Wire DS18B20 waterproof temperature sensor. Below shows the test case, actual result and the result obtained.

No	Test case	Actual result	Result
1	Place the DS18B20 waterproof temperature sensor into the paper cup fill with hot water.	Digital thermometer display the value of 45 Celsius	The sensor output approximately same value as actual result that is 43 C with percentage error of 4.58%. **(Figure 5-F22)**
2	Place the DS18B20 waterproof temperature sensor into the paper cup fill with cold water.	Digital thermometer display the value of 22 Celsius	The sensor output approximately same value as actual result that is 25 C with percentage error of 4.58%. **(Figure 5-F23)**

Table 5-0-1-T13: Test plan for DS18B20



```
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.937 °C
Temperature : 24.937 °C
Temperature : 26.687 °C
Temperature : 31.125 °C
Temperature : 34.5 °C
Temperature : 36.562 °C
Temperature : 38.0 °C
Temperature : 39.0 °C
Temperature : 39.812 °C
Temperature : 40.437 °C
Temperature : 41.0 °C
Temperature : 41.437 °C
Temperature : 41.812 °C
Temperature : 42.062 °C
Temperature : 42.312 °C
Temperature : 42.5 °C
Temperature : 42.625 °C
Temperature : 42.812 °C
Temperature : 42.937 °C
```



```
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.75 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.75 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.812 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature : 24.875 °C
Temperature
```

Figure 5-0-1-F26: Hot water (45 C) **Figure 5-0-1-F27: Cold water (22 C)**

In this project, DS18B20 is used to measure the water temperature of the aquarium in order to maintain the water temperature for the fish. It is also play important role in the automation part of the project where the system will automatic on the cooling fan if the water temperature is too high for example greater than 31 Celsius.

F) 28BYJ48 STEPPER MOTOR

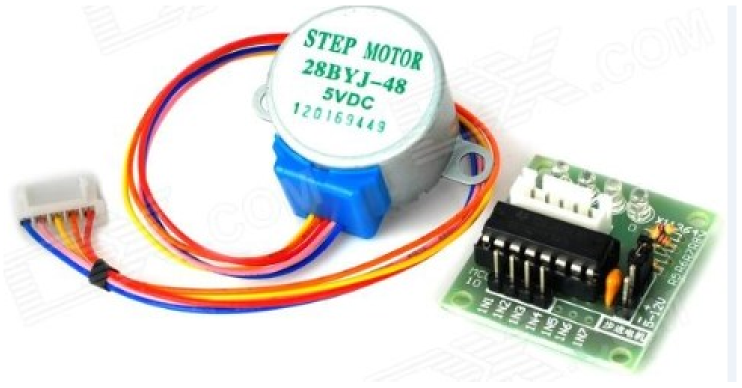


Figure 5-0-1-F28: 28BYJ48 STEPPER MOTOR

The 28BYJ48 Stepper Motor is a small stepper motor which convert electronic signals into mechanical movement each time an incoming pulse is applied to the motor. It is very suitable for large application. There are two commonly used mode for stepper motor, full step and half step. The full step is divided into two types One-phase and Two-phase. In one phase, the stepper motor is operated with only one phase energized at a time. In Two-phase stepper motor is operated with two phase energized at a time. Whereas, in half step mode one phase will energized first follow by two phase. Full step have more torque compared to half step mode. In this project, stepper motor will be used with ULN2003 motor driver in order to feed the fish. Table 5-0-1-T14 shows the basic specification of 28BYJ48 Stepper Motor.

Operating Voltage	5VDC
Number of Phase	4
Speed Variation Ratio	1/64
Stride Angle	5.625° /64
Self-positioning Torque	>34.3Mn.m

Table 5-0-1-T14: Specification of stepper motor

E) SULN2003 (STEPPER MOTOR DRIVER PCB)

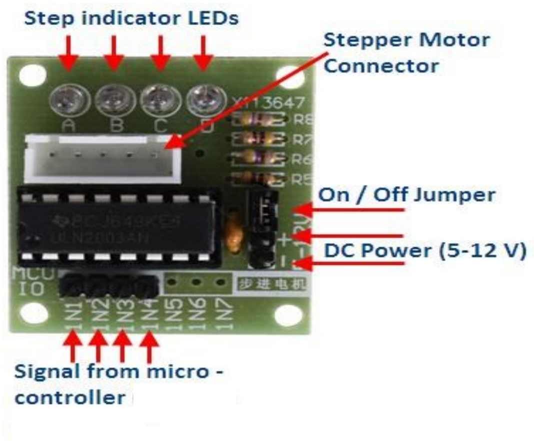


Figure 5-0-1-F29: ULN2003

ULN2003 is motor driver which provided interface between the Raspberry pi and 28BYJ48 stepper motor. The stepper motor cannot be drive directly from the Raspberry pi because their coils are needed to activate in a specific sequence in order to rotate. Hence they need a controller to convert power into the correct sequence of pulses to the motor's various input which a Raspberry Pi pins can't do it efficiently. Besides stepper motor also draws high current compared to the current that Raspberry Pi can output. If you directly connect the motor to Raspberry Pi you will damage the board. ULN2003 stepper motor driver PCB provided 4 input for connection to Raspberry Pi, power supply connection for stepper motor, and 4LEDs to indicate the stepping state. Table 5-T15 shows the basic specification of ULN2003 Stepper Motor driver.

Operating Voltage	5VDC
Phase	4
Step Angle	5.625° (1/64)
Reduction ratio	1/64

Table 5-0-1-T15: Specification of ULN2003

Connection of 28BYJ48 stepper motor with ULN2003 Motor driver and Raspberry Pi 3

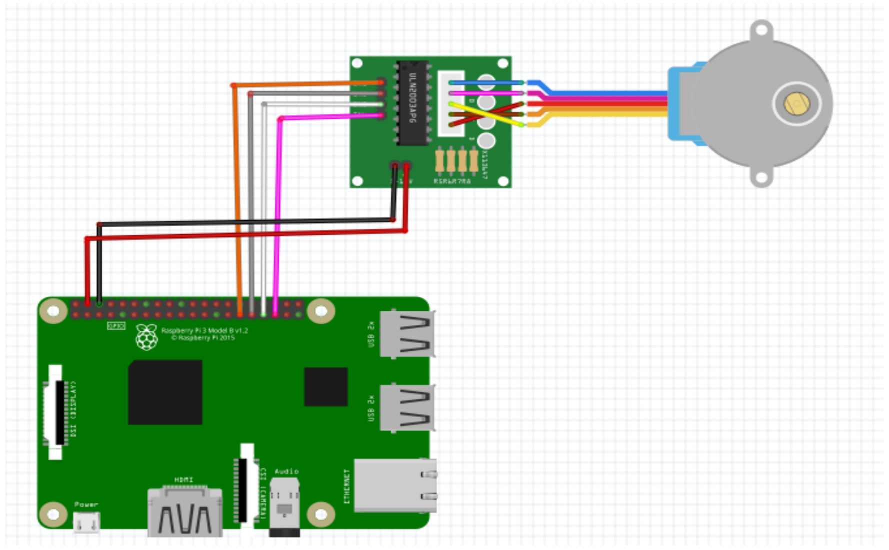


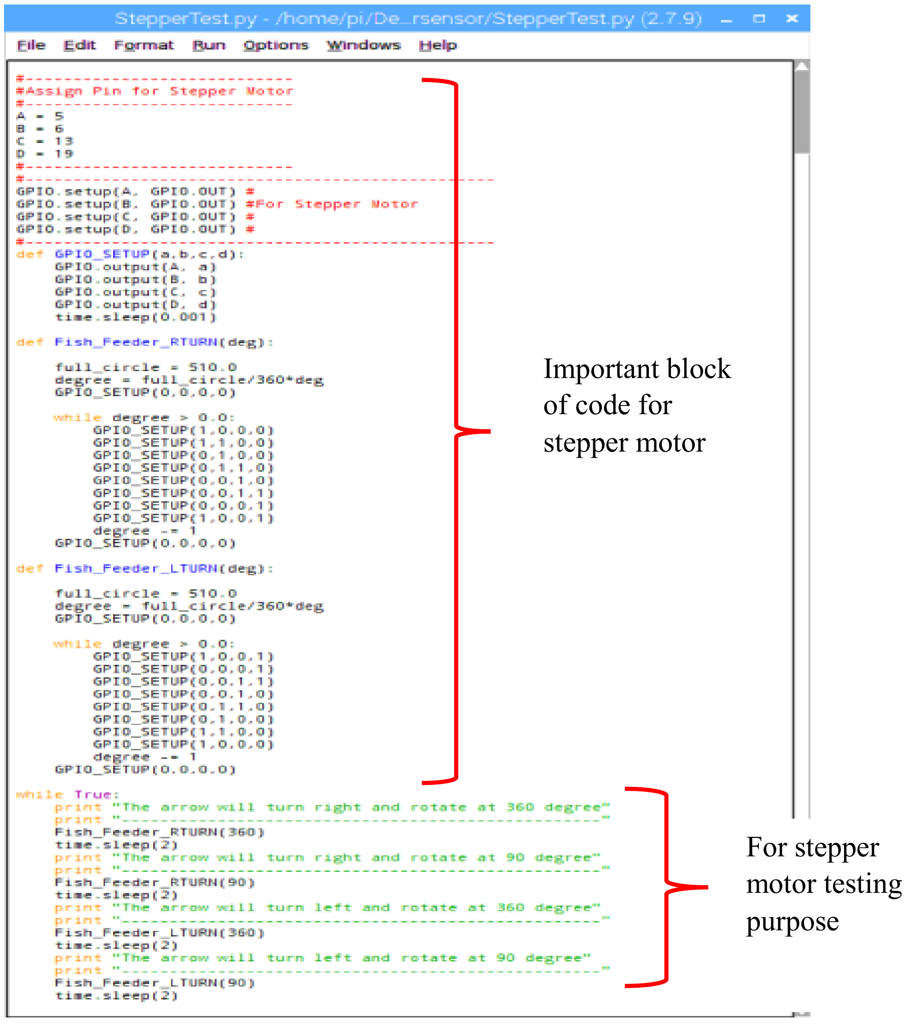
Figure 5-0-1-F30: Connection between 28BYJ48 with ULN2003 Motor driver board and Raspberry Pi 3

Figure 5-0-1-F29 shows the connection between 28BYJ48 with ULN2003 Motor driver board and Raspberry Pi 3. There are 5 wires connected from the stepper motor to the ULN2003 motor driver and 4 wire are connected from Pi to ULN2003 motor driver. As we discuss earlier in section 3.2.1, motor driver is required to drive the stepper motor because Pi's GPI0 is insufficient to energize the different phases of stepper motor. Below shows the pin connection between the Stepper motor, motor driver and the Pi.

Raspberry Pi 3 pin	**ULN2003 Motor driver board**	**28BYJ48 Stepper Motor**
5V (Pin 4)	Power Pin	--
Ground (Pin 6)	Ground Pin	--
GPI019 (Pin 35)	IN4	--
GPI013 (Pin 33)	IN3	--
GPI06 (Pin 31)	IN2	--
GPI05 (Pin 29)	IN3	--

--	OUT1	Blue wire (Coil 4)
--	OUT2	Pink wire (Coil 3)
--	COM	Red (Common)
--	OUT4	Orange (Coil 1)
	OUT3	Yellow (Coil 2)

Table 5-0-1-T16: Pin description


StepperTest.py - /home/pi/De...rsensor/StepperTest.py (2.7.9)
File Edit Format Run Options Windows Help
Important block of code for stepper motor
For stepper motor testing purpose

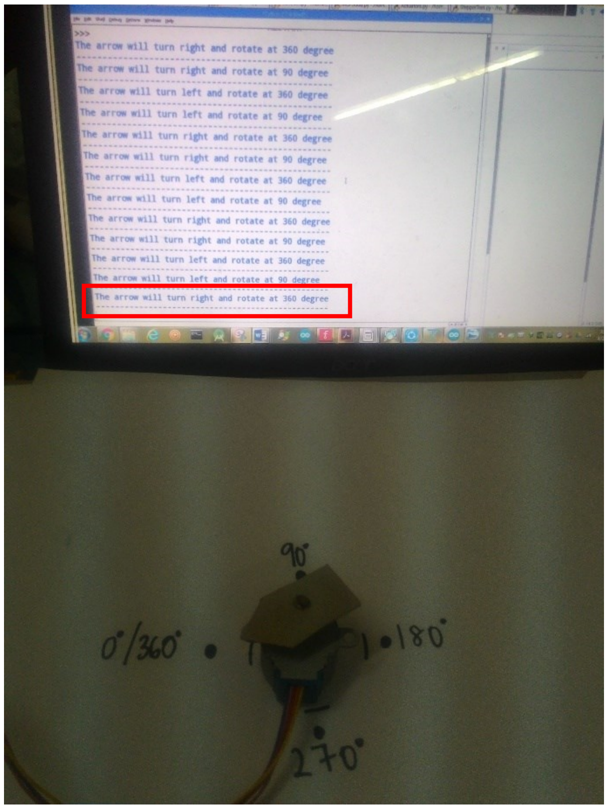
```
#------------------------------
#Assign Pin for Stepper Motor
#------------------------------
A = 5
B = 6
C = 13
D = 19
#------------------------------
#-------------------------------------------------
GPIO.setup(A, GPIO.OUT) #
GPIO.setup(B, GPIO.OUT) #For Stepper Motor
GPIO.setup(C, GPIO.OUT) #
GPIO.setup(D, GPIO.OUT) #
#-------------------------------------------------
def GPIO_SETUP(a,b,c,d):
    GPIO.output(A, a)
    GPIO.output(B, b)
    GPIO.output(C, c)
    GPIO.output(D, d)
    time.sleep(0.001)

def Fish_Feeder_RTURN(deg):

    full_circle = 510.0
    degree = full_circle/360*deg
    GPIO_SETUP(0,0,0,0)

    while degree > 0.0:
        GPIO_SETUP(1,0,0,0)
        GPIO_SETUP(1,1,0,0)
        GPIO_SETUP(0,1,0,0)
        GPIO_SETUP(0,1,1,0)
        GPIO_SETUP(0,0,1,0)
        GPIO_SETUP(0,0,1,1)
        GPIO_SETUP(0,0,0,1)
        GPIO_SETUP(1,0,0,1)
        degree -= 1
    GPIO_SETUP(0,0,0,0)

def Fish_Feeder_LTURN(deg):

    full_circle = 510.0
    degree = full_circle/360*deg
    GPIO_SETUP(0,0,0,0)

    while degree > 0.0:
        GPIO_SETUP(1,0,0,1)
        GPIO_SETUP(0,0,0,1)
        GPIO_SETUP(0,0,1,1)
        GPIO_SETUP(0,0,1,0)
        GPIO_SETUP(0,1,1,0)
        GPIO_SETUP(0,1,0,0)
        GPIO_SETUP(1,1,0,0)
        GPIO_SETUP(1,0,0,0)
        degree -= 1
    GPIO_SETUP(0,0,0,0)

while True:
    print "The arrow will turn right and rotate at 360 degree"
    print "---------------------------------------------------"
    Fish_Feeder_RTURN(360)
    time.sleep(2)
    print "The arrow will turn right and rotate at 90 degree"
    print "---------------------------------------------------"
    Fish_Feeder_RTURN(90)
    time.sleep(2)
    print "The arrow will turn left and rotate at 360 degree"
    print "---------------------------------------------------"
    Fish_Feeder_LTURN(360)
    time.sleep(2)
    print "The arrow will turn left and rotate at 90 degree"
    print "---------------------------------------------------"
    Fish_Feeder_LTURN(90)
    time.sleep(2)
```

Figure 5-0-1-F31: Code for stepper motor

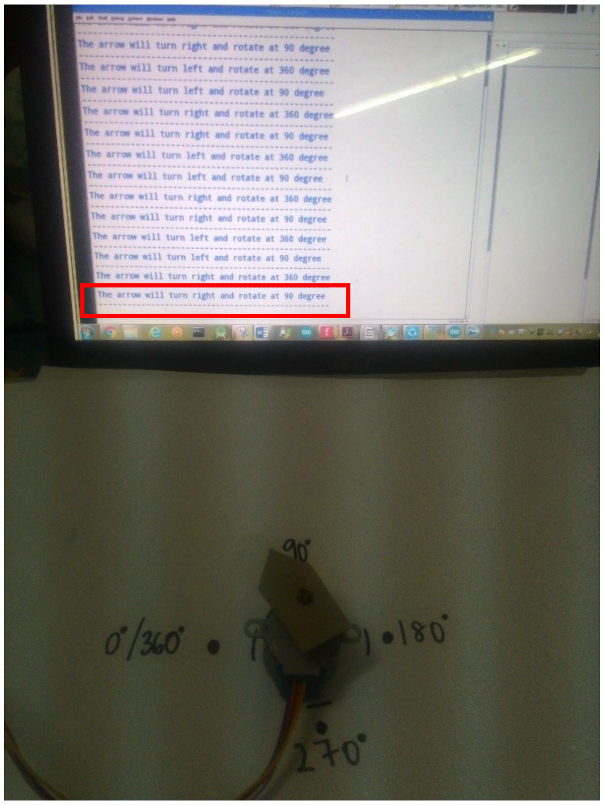
Several test case have been applied to check whether the stepper motor is working properly. I draw the circle with some angles at a piece of drawing paper and place the stepper motor in the middle of the circle. I add a paper arrow on the top rod of the stepper motor to check whether it complete the rotation with the specific angle values. Below table shows the test case, actual result and result obtain from the test for stepper motor.

No	Test case	Actual result	Result
1	Place the stepper motor at the middle of circle and run the following line of code **Fish_Feeder_RTURN(360)** to rotate the stepper to 360 degree to the right .	The stepper motor should rotate 360 degree to the right and come back to original position.	The stepper motor do rotate 360 degree to the right and come back to the original position as per actual result. However the stepper motor stop a bit early before reaching the original position. (**Figure 5-F28)**
2	Place the stepper motor at the middle of circle and run the following line of code **Fish_Feeder_RTURN(90)** to rotate the stepper to 90 degree to the right.	The stepper motor should rotate 90 degree to the right and come back to original position.	The stepper motor do rotate 90 degree to the right and come back to the original position as per actual result. However the stepper motor stop a bit early before reaching the original position. (**Figure 5-F29)**
3	Place the stepper motor at the middle of circle and run the following line of code **Fish_Feeder_LTURN(360)** to rotate the stepper to 360 degree to the left.	The stepper motor should rotate 360 degree to the left and come back to original position.	The stepper motor do rotate 360 degree to the left and come back to the original position as per actual result. However the stepper motor stop a bit early before reaching the original position. (**Figure 5-F30)**
4	Place the stepper motor at the middle of circle and run the following line of code **Fish_Feeder_LTURN(90)** to rotate the stepper to 90 degree to the left.	The stepper motor should rotate 90 degree to the left and come back to original position.	The stepper motor do rotate 90 degree to the left and come back to the original position as per actual result. However the stepper motor stop a bit early before reaching the original position. (**Figure 5-F31)**

Table 5-0-1-T17: Test plan for stepper motor

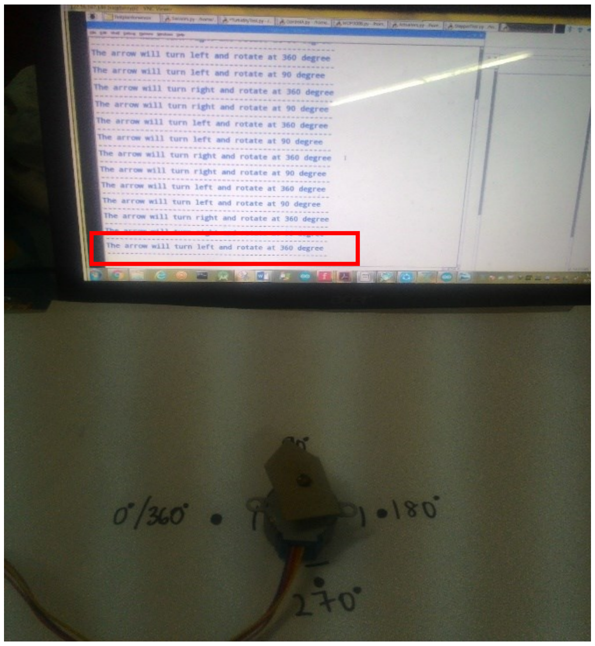


```
>>>
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
```

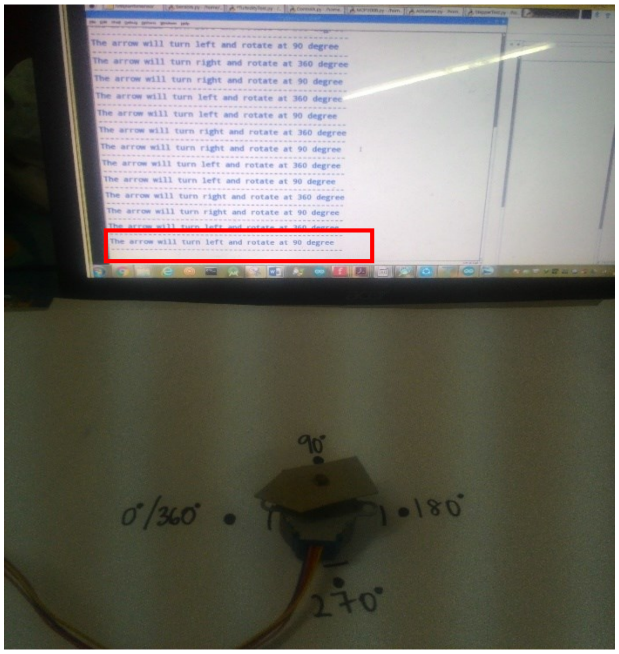


```
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
```

Figure 5-0-1F32: Stepper motor (Right, 360d) Figure 5-0-1F33: Stepper motor (Right, 90d)



```
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
[illegible]
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
```



```
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
The arrow will turn right and rotate at 360 degree
--------------------------------------------------
The arrow will turn right and rotate at 90 degree
--------------------------------------------------
The arrow will turn left and rotate at 360 degree
--------------------------------------------------
The arrow will turn left and rotate at 90 degree
--------------------------------------------------
```

Figure 5-0-1F34: Stepper motor (Left, 360d)   Figure 5-0-1F35: Stepper motor (Left, 90d)

In this project, I using the stepper motor to feed the fish by connecting with a small plastic container. Then I make a small hole at the bottom of the container. If the container rotates, using stepper motor, the fish's dry food will be drop through the hole. Below figure shows the stepper motor with the plastic container,

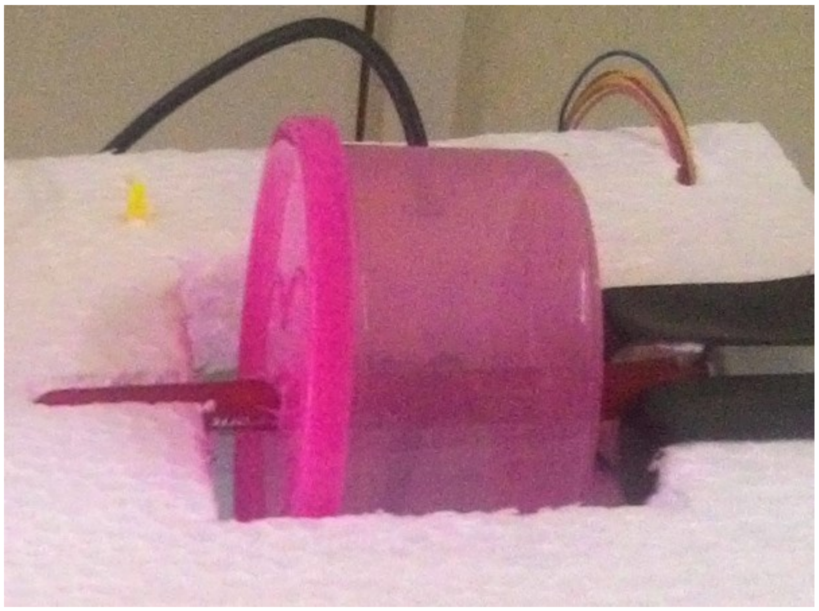


Figure 5-0-1-F36: Prototype of Fish Feeder

F) SRD-05VDC-SL-C (4 CHANNEL 5V RELAY MODULE)

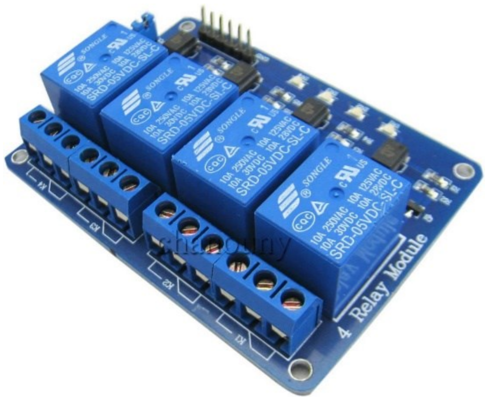


Figure 5-0-1-F37: SRD-05VDC-SL-C 5V

Relay is used to control high voltage (120 to 240V) devices such as fan, lights and some household appliances. Since Raspberry Pi operates at 3.3V, it can't control the high voltage devices directly. Hence with the help of relay, Raspberry Pi can be used to turn on and off high voltage devices. In this project, I'm using 4channel 5V relay to turn on/off two water pump motor, a pair of cooling fan and white LED light for the aquarium. Table below shows the basic specification of SRD-05VDC-SL-C 5V relay.

Maximum Allowable Voltage	250VAC/110VDC
Operation Time	10msec Max.
Max. ON/OFF Switching **Mechanically** **Electrically**	 300 operation/min 30 operation/min
Ambient Temperature	-25°C to +70°C
Operating Humidity	45 to 85% RH

Table 5-0-1-T18: Specification for relay

Connection of SRD-05VDC-SL-C 5V with Raspberry pi 3

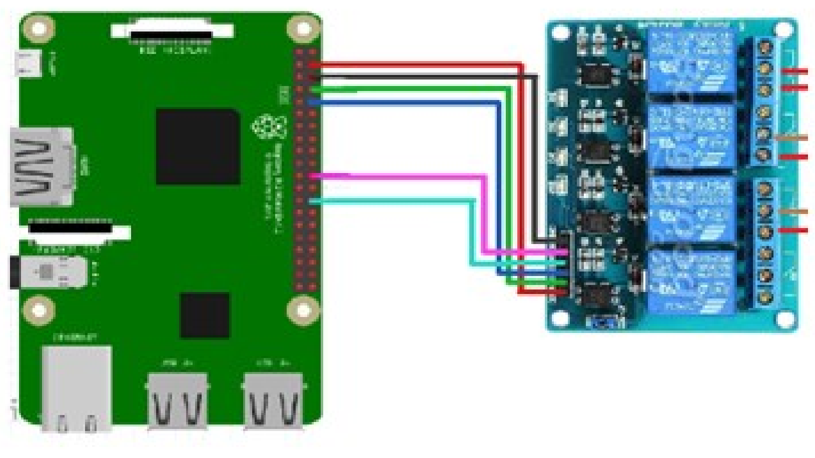
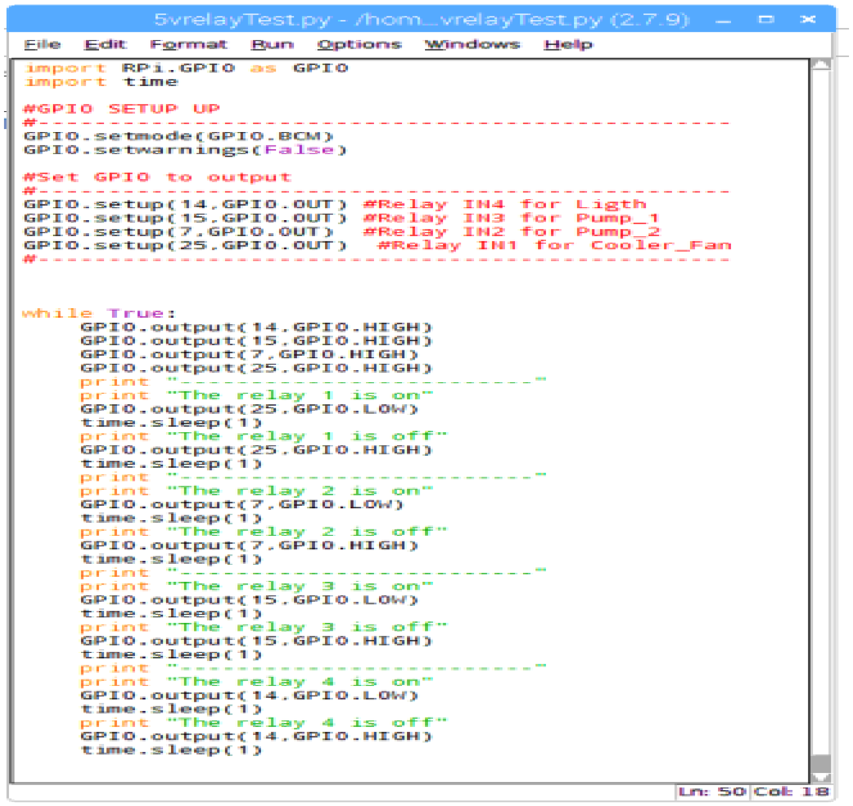


Figure 5-0-1-F38: SRD-05VDC-SL-C 5V with Raspberry pi 3

Figure 5-0-1-F37 shows the connection between the 4 Channel 5V relay module with Pi 3. A single 5V relay does not need any power supply but since 4 channel 5V relay module is used in this project, it need 5V power supply from Pi to light up led indicator at each relay. The LED at relay module act as indicator whether the particular relay receive signal from Pi's GPIO pin or not. Below shows the pin connection between the 4 Channel 5v relay module with Pi 3.

Raspberry Pi 3	4channel 5V relay
5V (Pin 4)	VCC
Ground (Pin 14)	GND
GPI015 (Pin 10)	IN2
GPI07 (Pin 26)	IN3
GPI025 (Pin 22)	IN1

Table 5-0-1-T19: Pin description


5vrelayTest.py - /hom...vrelayTest.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 50 Col: 18

```python
import RPi.GPIO as GPIO
import time

#GPIO SETUP UP
#-----------------------------------------------------------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Set GPIO to output
#-----------------------------------------------------------------
GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligth
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
GPIO.setup(7,GPIO.OUT)  #Relay IN2 for Pump_2
GPIO.setup(25,GPIO.OUT)  #Relay IN1 for Cooler_Fan
#-----------------------------------------------------------------


while True:
    GPIO.output(14,GPIO.HIGH)
    GPIO.output(15,GPIO.HIGH)
    GPIO.output(7,GPIO.HIGH)
    GPIO.output(25,GPIO.HIGH)
    print "---------------------------"
    print "The relay 1 is on"
    GPIO.output(25,GPIO.LOW)
    time.sleep(1)
    print "The relay 1 is off"
    GPIO.output(25,GPIO.HIGH)
    time.sleep(1)
    print "---------------------------"
    print "The relay 2 is on"
    GPIO.output(7,GPIO.LOW)
    time.sleep(1)
    print "The relay 2 is off"
    GPIO.output(7,GPIO.HIGH)
    time.sleep(1)
    print "---------------------------"
    print "The relay 3 is on"
    GPIO.output(15,GPIO.LOW)
    time.sleep(1)
    print "The relay 3 is off"
    GPIO.output(15,GPIO.HIGH)
    time.sleep(1)
    print "---------------------------"
    print "The relay 4 is on"
    GPIO.output(14,GPIO.LOW)
    time.sleep(1)
    print "The relay 4 is off"
    GPIO.output(14,GPIO.HIGH)
    time.sleep(1)
```

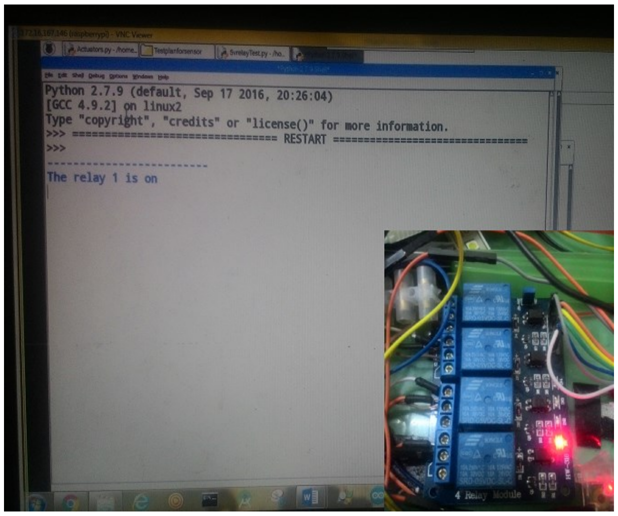
Figure 5-0-1-F39: Code for control the 5V relay

In order test the relay, I have defined several test case to make sure it working properly. I just send a low signal (0) to the relay from the Pi and if the relay received the signal 0 it will on the relay and LED indicator of the particular relay will switch on. If it receive signal high 1, it will off the relay and LED will turn off. Below shows the test case, actual result and result obtained.

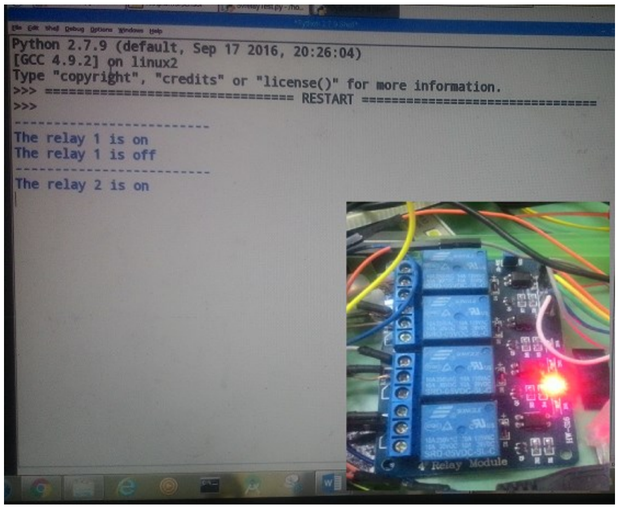
No	**Test case**	**Actual result**	**Result**
1	On the relay by sending a low signal (0) to the first relay (IN1) for 1 second.	IN1 relay is on and its LED indicator is turn on.	IN1 relay is on and its LED indicator is turn on as actual result. **(Figure 5-F36)**
2	On the relay by sending a low signal (0) to the first relay (IN2) for 1 second.	IN2 relay is on and its LED indicator is turn on.	IN2 relay is on and its LED indicator is turn on as actual result. **(Figure 5-F37)**

3	On the relay by sending a low signal (0) to the first relay (IN3) for 1 second.	IN3 relay is on and its LED indicator is turn on.	IN3 relay is on and its LED indicator is turn on as actual result. **(Figure 5-F38)**
4	On the relay by sending a low signal (0) to the first relay (IN4) for 1 second.	IN4 relay is on and its LED indicator is turn on.	IN4 relay is on and its LED indicator is turn on as actual result. **(Figure 5-F39)**

Table 5-0-1-T20: Test plan for relay

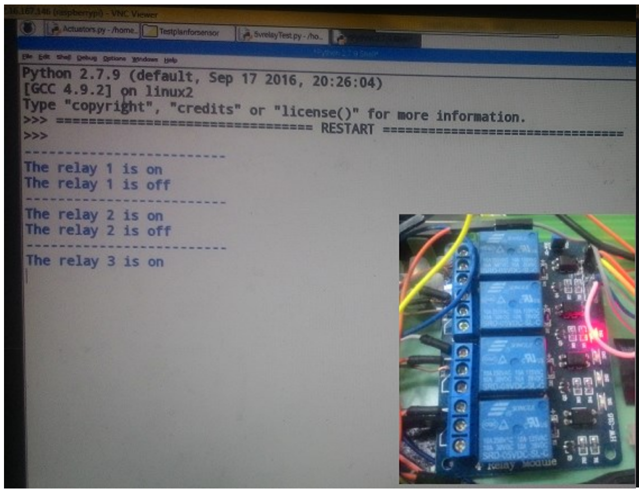


```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
------------------------
The relay 1 is on
```

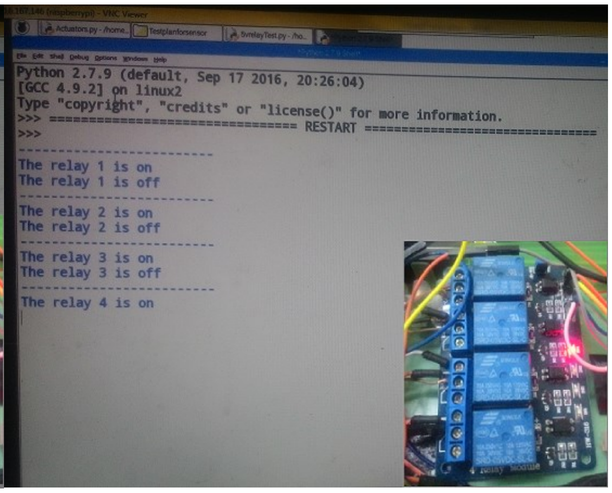


```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
------------------------
The relay 1 is on
The relay 1 is off
------------------------
The relay 2 is on
```

Figure 5-0-1-F40:IN1 relay is turn on **Figure 5-0-1-F41: IN2 relay is turn on**



```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
------------------------
The relay 1 is on
The relay 1 is off
------------------------
The relay 2 is on
The relay 2 is off
------------------------
The relay 3 is on
```



```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
------------------------
The relay 1 is on
The relay 1 is off
------------------------
The relay 2 is on
The relay 2 is off
------------------------
The relay 3 is on
The relay 3 is off
------------------------
The relay 4 is on
```

Figure 5-0-1-F42: IN3 relay is turn on **Figure 5-0-1-F43: IN4 relay is turn on**

G) DS3231 RTC BOARD (REAL TIME CLOCK)

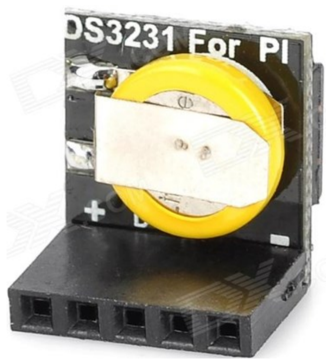


Figure 5-0-1-F44: DS3231

DS3231 is an extremely accurate I2C real time clock which help the Raspberry pi to keep s the time even when the power is off. By default, Raspberry Pi does not have a build in RTC to keep the time, but instead the Pi will try to connect to internet to update its time automatically from global network time protocol (NTP) servers. But if there is no internet connection, the Pi cannot update its time. So, with this low cost DS3231 connect to Pi, the Pi to able to keep the time after it been turn off too. In my project, RTC is very important because there are some automatic steps that the Pi have to do by itself on particular time for example feed the fish at 9pm or turn on the light if its night. Table below shows the basic specification of DS3231.

Operating voltage	3.3 V
Operating Temperature Ranges	Commercial (0°C to +70°C) Industrial (-40°C to +85°C)
Interface	I2C Interface
Accuracy ±3.5ppm	-40°C to +85°C

Table 5-0-1-T21: Specification of RTC

<u>Connection and configuration of DS3231 RTC BOARD with Raspberry Pi 3</u>

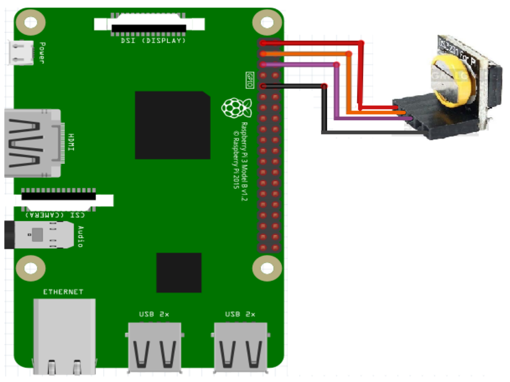


Figure 5-0-1-F45: Connection between the DS3231 RTC board with the Raspberry Pi 3

Figure 5-0-1-F44 shows the connection between the DS3231 RTC board with the Raspberry Pi 3. DS3231 is used to keep the time for the Pi because Pi does not have built in RTC to keep time. Without external RTC, the Pi have to update the time from internet every time it boot up. This might be difficult in the situation where there is no Internet connection for Pi to update the time. To resolve such a problem, the DS3231 RTC is used. Below shows the connection of RTC with the Pi.

Raspberry Pi 3 pin	DS3231 RTC
3.3 V (Pin 1)	VCC
Ground (Pin 9)	GND
GPI02 (Pin 3) – SDA1 I2C	D (SDA)
GPI03 (Pin 5) – SCL1 I2C	C (SCL)

Table 5-0-1-T22: Pin description

Besides the connection, there are some configuration that need to be done in order the DS3231 RTC board to work. Firstly, the /boot/config.txt need to be edit by adding the following line at the bottom of the config.txt file,

dtoverlay = i2c-rtc,ds3231

Secondly edit the hwclock-set file located at /lib/udev/hwclock-set and comment out the following line so that they will be ignore by the system.

if [-e/run/system/system]; then
exit 0
fi
Comment those line at above. It will become as below
#if [-e/run/system/system]; then
#exit 0
#fi
Then reboot the Pi's system and test the RTC with command "**sudo hwclock –r "** the system by right should read the time directly from the RTC. By default the Pi get the current date and time through an NTP server, a time server through internet. By connecting the RTC and make some configuration at shown above, now the system should get time from the RTC board instead from internet. To test whether the Pi is really read the time from RTC and not from the internet, I power off the Pi and disconnect the pi from the global and local network. By disconnect the pi from all network, it cannot find the current time and date from the NTP server. Then I plug in a LCD screen with Pi using HDMI to VGA converter and a locally connected keyboard. I wait for 15 minutes before I power up my Pi. Then I check the current time of the pi. The current time of the Pi is correct and not 15 minutes late. So I can conclude that the DS3231 RTC board is working perfectly.

In this project, RTC is very important especially in the automation part of the project, because the system have to feed the fish and on the light of aquarium at particular time set by the user. So in order to perform such actions, the Pi should know the current time correctly and the external RTC DS3231 can help to keep the time when there is no internet.

H) 5050 WHITE 3 LED STRIP(X4)

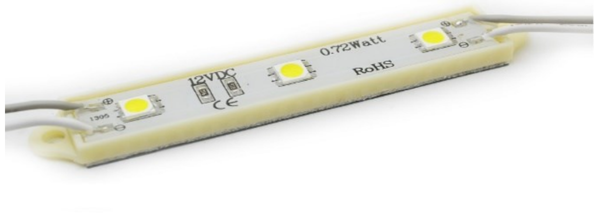


Figure 5-0-1-F46: 5050 WHITE 3 LED

5050 white 3 LED strip is low power consumption bright white LED which are commonly used for interior lighting, showcases and ceilings decoration. In this project, four set of 5050 white 3LED are used to light up the aquarium. Below table shows the basic specification of 5050 white led strip.

Operating Voltage	12V DC
Wattage per Reel	150 LED Count: 36W 300 LED Count: 72W
Operating Temperature	−20 to +50 C
Storage Temperature:	−30 to +80 C
Color	White
Waterproof	Yes

Table 5-0-1-T23: Specification of White LED

Connection of 5050 WHITE LED STRIP with Raspberry Pi 3

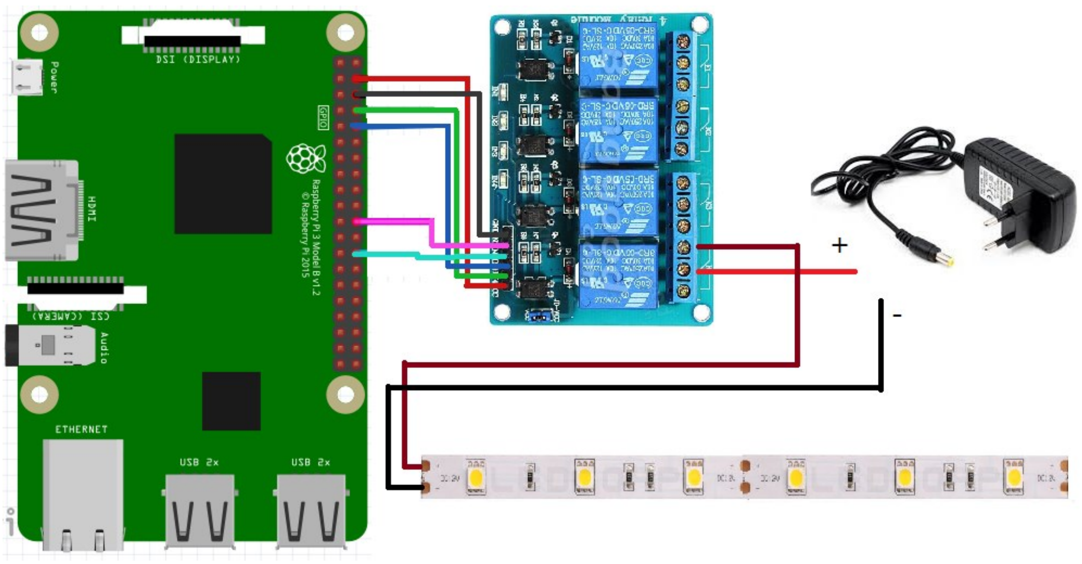
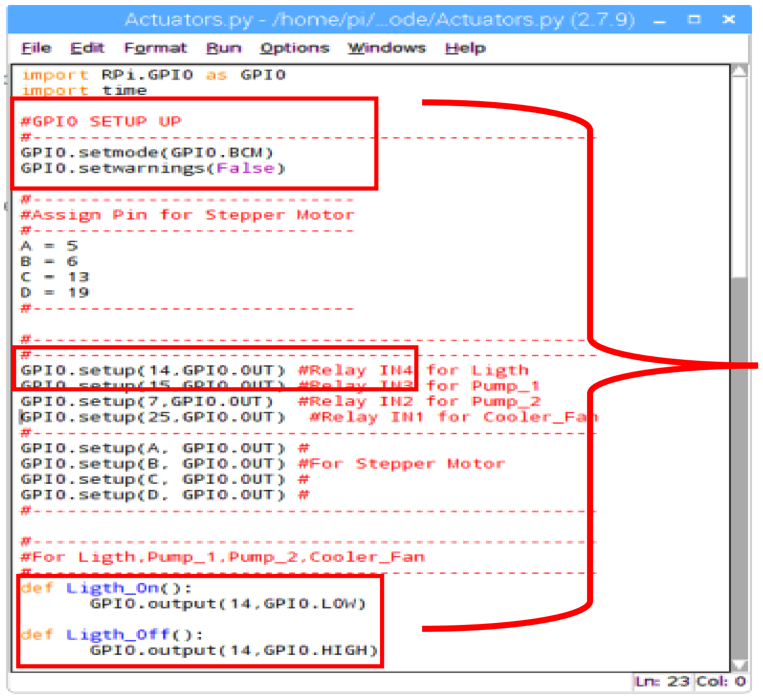


Figure 5-0-1-F47: Connection between 5050 white led strip and Raspberry Pi 3

Figure 5-0-1-F46 shows the connection between 5050 white led strip, 5V relay and the Raspberry pi. The white LED is power up using external power source DC 12V 1A. The power source and the LED's power line is connected at K4 relay. The LED's ground is connect with the ground of external power supply. Connect the 4channel 5V relay board to the power 5V and the ground of the Pi. Then connect the wire from Pi's **GPIO14 (Pin 8)** to the 4 channel relay board's pin of IN4. When the Pi's send LOW to IN4 the relay will connect the power supply to the white LED which will make the light to on and send HIGH to turn of the relay and the light.


Actuators.py - /home/pi/...ode/Actuators.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 23 Col: 0

```
import RPi.GPIO as GPIO
import time

#GPIO SETUP UP
#-----------------------------------------------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#------------------------------
#Assign Pin for Stepper Motor
#------------------------------
A = 5
B = 6
C = 13
D = 19
#------------------------------

#-----------------------------------------------------
#-----------------------------------------------------
GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligth
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
GPIO.setup(7,GPIO.OUT)  #Relay IN2 for Pump_2
GPIO.setup(25,GPIO.OUT)  #Relay IN1 for Cooler_Fan
#-----------------------------------------------------
GPIO.setup(A, GPIO.OUT) #
GPIO.setup(B, GPIO.OUT) #For Stepper Motor
GPIO.setup(C, GPIO.OUT) #
GPIO.setup(D, GPIO.OUT) #
#-----------------------------------------------------

#-----------------------------------------------------
#For Ligth,Pump_1,Pump_2,Cooler_Fan
#-----------------------------------------------------
def Ligth_On():
      GPIO.output(14,GPIO.LOW)

def Ligth_Off():
      GPIO.output(14,GPIO.HIGH)
```

The important block of code for control the White Led on and off.

Figure 5-0-1-F48: Code to control white led using relay and pi

To test the white led whether it work or not, two test case have been defined. Below show the test case, the actual result and the result obtained.

No	Test case	Actual result	Result
1	Turn on the 12V power source for 5050 white LED and send a signal 0 from pi to the relay by calling the function Light_On() at Actuator.py as shown in Figure 5-F44.	The relay should complete the circuit and 5050 white led should turn on.	The 5050 white led do turn on as actual result.
2	Turn on the 12V power source for 5050 white LED and send a signal 1 from pi to the relay by calling the function Light_Off() at Actuator.py as shown in Figure 5-F44.	The relay should cut the circuit and the 5050 white led should turn off.	The 5050 white led do turn off as actual result.

Table 5-0-1-T24: Test plan for White LED

I) 3528 SMD RGB LED

Figure 5-0-1-F49: **3528 SMD RGB**

3528 SMD RGB LED is consist of 3 LEDs Red, Green and Blue. In this project this RGB LED will be used to indicator for the user whenever there is problem with the aquarium such as water leak or the water temperature of aquarium is so high. For example, if the system detect the water leak, the RGB LED will automatically blink continuously to RED color as an alert indicator for the user. Besides, the RGB LED also use as lighting decoration for the aquarium. Below table shows the basic specification of 3528 SMD RGB LED.

Operating Voltage	12 V DC
Operating Power	24W/5M
Operating Current/meter	0.35-0.625A
Working Temperature	-20 to 50°C
Color	RGB(RED,GREEN AND BLUE)
Waterproof	NO

Table 5-0-1-T25: Specification of RGB LED

Connection and configuration of RGB LED strip with Raspberry Pi 3

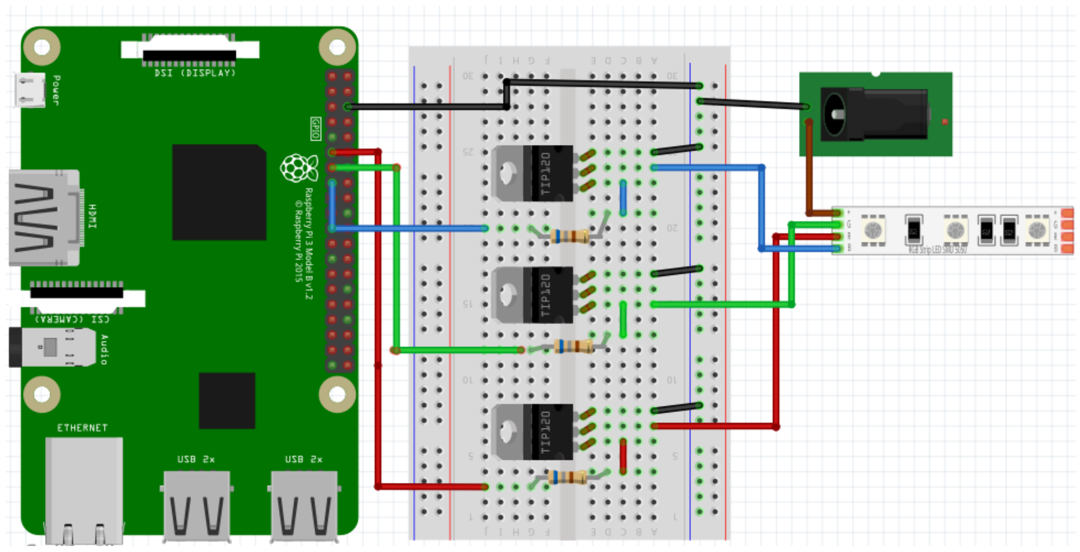


Figure 5-0-1-F50: Connection between RGB LED and Raspberry Pi

Figure 5-0-1-F49 shows the connection between 3528 SMD RGB LED strip with Raspberry Pi 3.The 3528 SMD RGB led strip consist of 4 input pin, R (RED), G (GREEN) , B (BLUE) and V+ the power line. It is power up using an external power source of 12V 2A. In order to control the color of the RGB LED strip, three TIP120 power transistors were used to control each R, G and B of the LED respectively. They will act as switch which will switched on and off the RGB LED fast. A 680Ω resistor is connected between each of the base of the transistors and the Pi's GPIO to limit the base current and protect the Pi's GPIO. The external power source ground should connected to the Pi's ground. Below show the pin connection between Pi and TIP20 transistor.

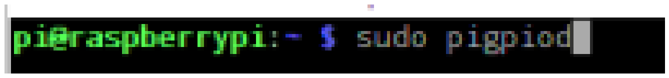
Raspberry Pi 3	**TIP120's base**
GPIO27 (Pin 13)	R (1st transistor from bottom)
GPIO22 (Pin 15)	G (2nd transistor from bottom)
GPIO17 (Pin 11)	B (3rd transistor from bottom)

Table 5-0-1-T26: Pin description

Besides the connection, the Pi needs an important library called **pigpio** in order to control the RGB LEDs. Pigpio is a library which help the Raspberry to position servo, control motor speed, control LED brightness and so on. Download and install the pigpio library to Pi using the following command,

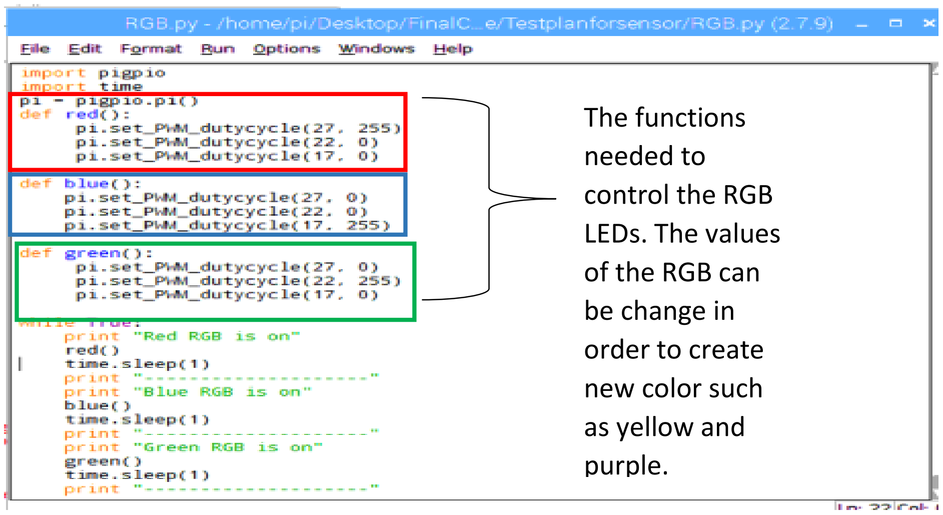
rm pigpio.zip
sudo rm -rf PIGPIO
wget abyz.co.uk/rpi/pigpio/pigpio.zip
unzip pigpio.zip
cd PIGPIO
make
sudo make install

Now we can use the pigpio library to control the RGB light. The following command is need to execute at the Pi terminal in order to start the pigpio library,



```
pi@raspberrypi:~ $ sudo pigpiod
```

(Take note that you have to execute the command above every time you reboot your Pi's system in order to use the library)


RGB.py - /home/pi/Desktop/FinalC...e/Testplanforsensor/RGB.py (2.7.9)
File Edit Format Run Options Windows Help

```
import pigpio
import time
pi = pigpio.pi()
def red():
     pi.set_PWM_dutycycle(27, 255)
     pi.set_PWM_dutycycle(22, 0)
     pi.set_PWM_dutycycle(17, 0)
def blue():
    pi.set_PWM_dutycycle(27, 0)
    pi.set_PWM_dutycycle(22, 0)
    pi.set_PWM_dutycycle(17, 255)
def green():
     pi.set_PWM_dutycycle(27, 0)
     pi.set_PWM_dutycycle(22, 255)
     pi.set_PWM_dutycycle(17, 0)
while True:
    print "Red RGB is on"
    red()
    time.sleep(1)
    print "---------------------"
    print "Blue RGB is on"
    blue()
    time.sleep(1)
    print "---------------------"
    print "Green RGB is on"
    green()
    time.sleep(1)
    print "---------------------"
```

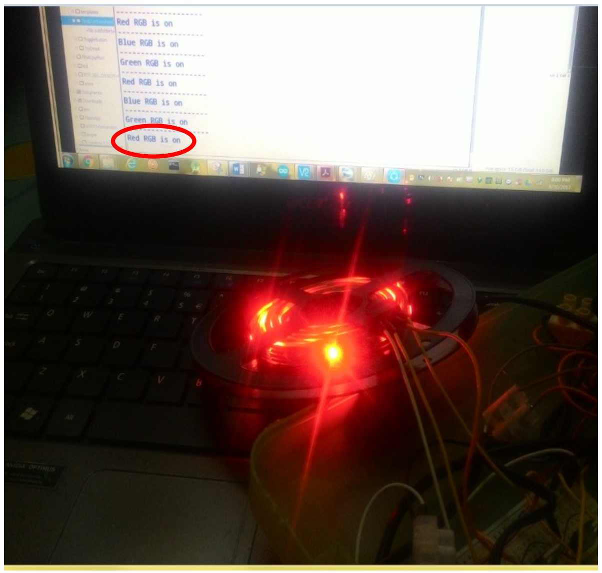
The functions needed to control the RGB LEDs. The values of the RGB can be change in order to create new color such as yellow and purple.

Figure 5-0-1-F51: Code required to control RGB LEDs

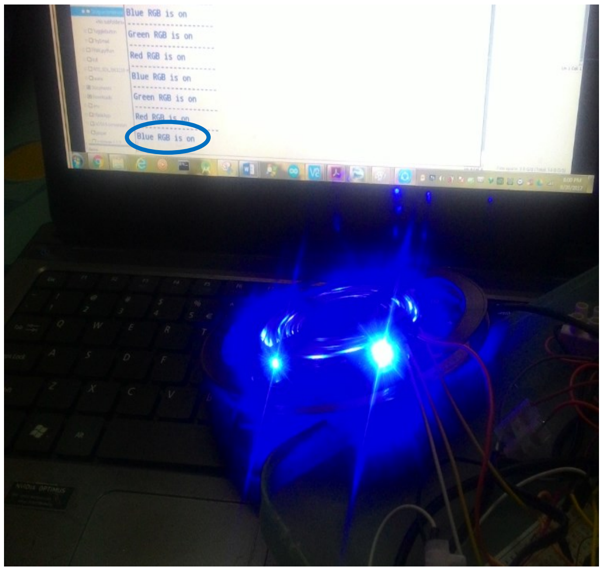
Test plan with three test case has been defined in order to check whether the RGB is working perfectly. Below shows the test cases, actual result and result obtain from the test plan.

No	Test case	Actual result	Result
1	Turn on the power source for RGB LEDs and run the following function called red() at RGB.py as shown in Figure 5-F46	The RGB LED should turn red color.	The RGB LED do turn red color. **(Figure 5-F47)**
2	Turn on the power source for RGB LEDs and run the following function called blue() at RGB.py as shown in Figure 5-F46.	The RGB LED should turn blue color.	The RGB LED do turn blue color. **(Figure 5-F48)**
3	Turn on the power source for RGB LEDs and run the following function called green() at RGB.py as shown in Figure 5-F46	The RGB LED should turn green color.	The RGB LED do turn green color. **(Figure 5-F49**)

Table 5-0-1-T27: Test plan for RGB LED

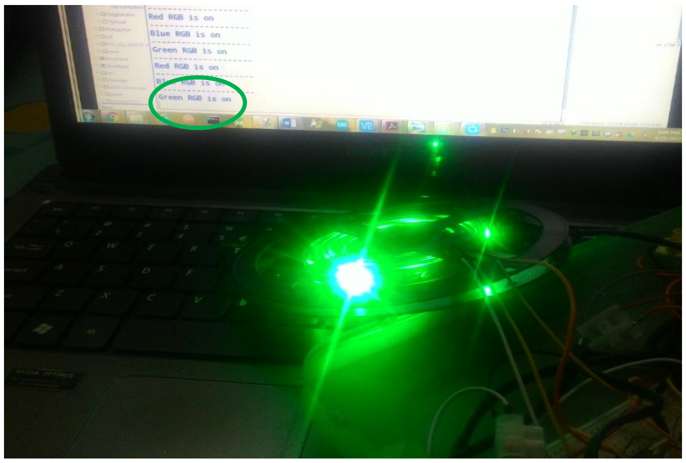


```
Red RGB is on
Blue RGB is on
Green RGB is on
Red RGB is on
Blue RGB is on
Green RGB is on
Red RGB is on
```



```
Blue RGB is on
Green RGB is on
Red RGB is on
Blue RGB is on
Green RGB is on
Red RGB is on
Blue RGB is on
```

Figure 5-0-1-F52: RED Light **Figure 5-0-1-F53: BLUE Light**



```
Red RGB is on
Blue RGB is on
Green RGB is on
Red RGB is on
Blue RGB is on
Green RGB is on
```

Figure 5-0-1-F54: GREEN Light

In this project RGB LED is used as indicator for the user about the aquarium condition. For an example, if there any water leak detected or the water temperature is too high, the RGB led will turn to Red as alert indicator for the user. Besides, the RGB also act as lighting decoration for the aquarium where the user can control what color that they want to color the aquarium from the website. The user can control the brightness of each RGB LEDs to form different color such as yellow or purple through the slider available at website.

J) 3.5V~12V DC BRUSHLESS WATER PUMP (x2)

Figure 5-0-1-F55: **WATER PUMP**

In this project two DC Brushless water pump is used to pump in and pump out the water to and from the aquarium. Below shows the basic specification of 3.5V~12V DC Brushless water pump,

Operating Voltage	3.5V to 12V
Operating Current	50mA-500mA
Maximum volume water pump per hour	350L
Continuously running life span	> 20000 hours

Table 5-0-1-T28: Specification of water pump

K) DIY COOLING FAN

Figure 5-0-1-F56: Cooling Fan

In this project two DC motor cooler fan was used in order to reduce the water temperature of aquarium using evaporative cooling method. The DC motor cooler fan was taken from laptop cooler fan because the commercial cooling fan price for aquarium is very high. Table below shows the basic specification of the DC motor,

Operating Voltage	5V to 12 V
Operating Current	50mA-1A
Dimension	8cm X 8cm

Table 5-0-1-T29: Specification of cooling fan

Connection of Water Pump and Cooling Fan with Raspberry Pi 3

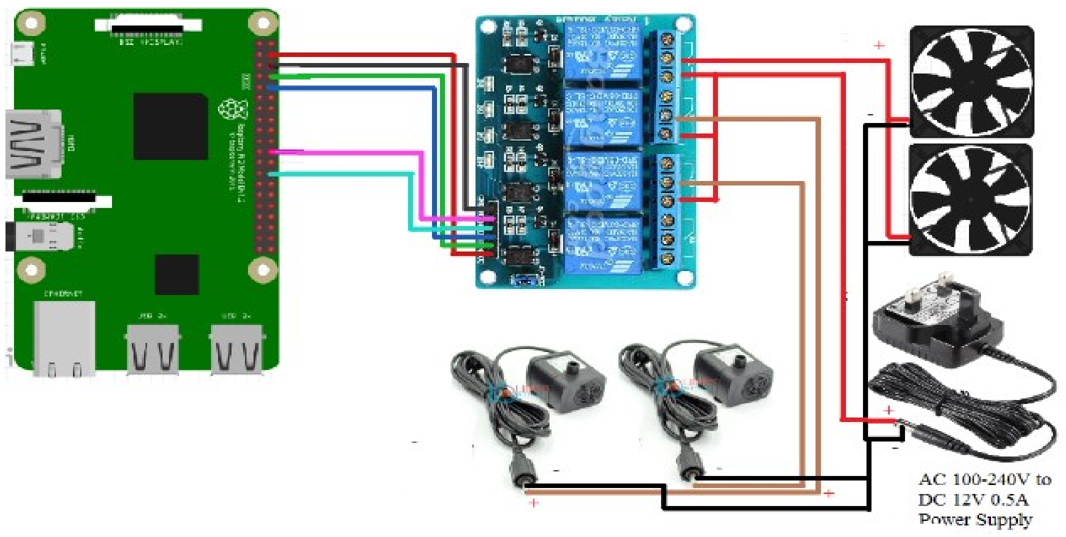
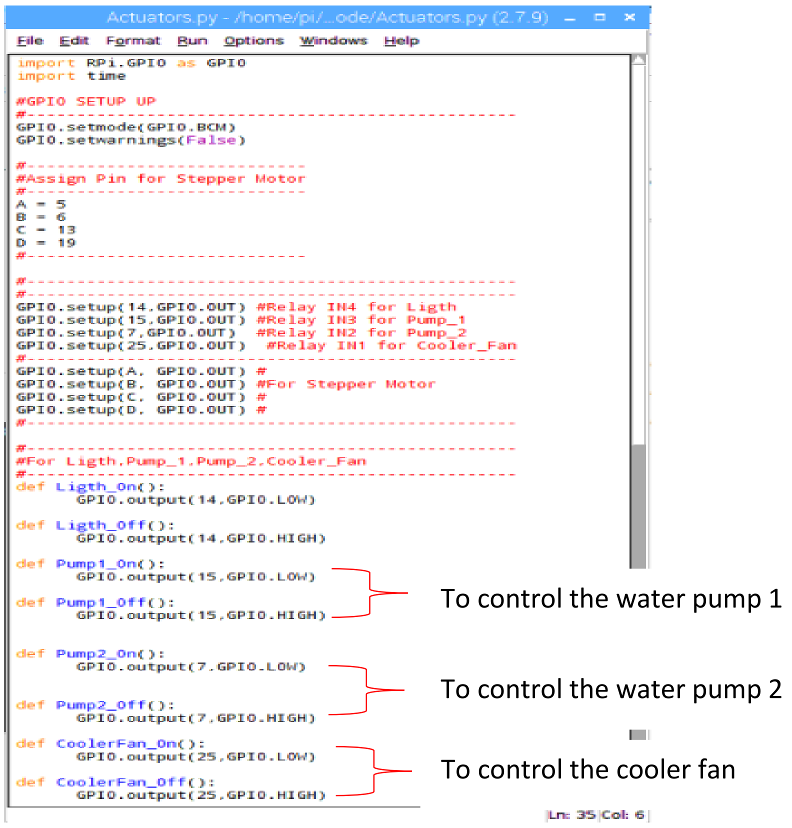


Figure 5-0-1-F57: Connection between Actuators

Figure 5-0-1-F56 shows the connection of two water pump, one cooling fan with the 4channel relay and Raspberry Pi 3. The cooling fan and the water pump were connect with same setup as shown in previous page. But the cooling fan will used the K1 relay and the water pump one and two will used K2 and K3 relay respectively. Below shows the pin connection of 4channel relay board with Pi,

Raspberry Pi 3	4channel 5V relay	Component control
5V (Pin 4)	VCC	--
Ground (Pin 14)	GND	--
GPI015 (Pin 10)	IN2	Pump 1
GPI07 (Pin 26)	IN3	Pump 2
GPI025 (Pin 22)	IN1	Cooling Fan

Table 5-0-1-T30: Pin description


Actuators.py - /home/pi/...ode/Actuators.py (2.7.9)
File Edit Format Run Options Windows Help

```python
import RPi.GPIO as GPIO
import time

#GPIO SETUP UP
#-----------------------------------------------------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#----------------------------------
#Assign Pin for Stepper Motor
#----------------------------------
A = 5
B = 6
C = 13
D = 19
#----------------------------------

#-----------------------------------------------------------
#-----------------------------------------------------------
GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligth
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
GPIO.setup(7,GPIO.OUT)  #Relay IN2 for Pump_2
GPIO.setup(25,GPIO.OUT)  #Relay IN1 for Cooler_Fan
#-----------------------------------------------------------
GPIO.setup(A, GPIO.OUT) #
GPIO.setup(B, GPIO.OUT) #For Stepper Motor
GPIO.setup(C, GPIO.OUT) #
GPIO.setup(D, GPIO.OUT) #
#-----------------------------------------------------------

#-----------------------------------------------------------
#For Ligth,Pump_1,Pump_2,Cooler_Fan
#-----------------------------------------------------------
def Ligth_On():
        GPIO.output(14,GPIO.LOW)

def Ligth_Off():
        GPIO.output(14,GPIO.HIGH)

def Pump1_On():
        GPIO.output(15,GPIO.LOW)

def Pump1_Off():
        GPIO.output(15,GPIO.HIGH)


def Pump2_On():
        GPIO.output(7,GPIO.LOW)


def Pump2_Off():
        GPIO.output(7,GPIO.HIGH)

def CoolerFan_On():
        GPIO.output(25,GPIO.LOW)

def CoolerFan_Off():
        GPIO.output(25,GPIO.HIGH)
```

To control the water pump 1
To control the water pump 2
To control the cooler fan
Ln: 35 Col: 6

Figure 5-0-1-F58: Code to test water pump and cooling fan

To test water pump and cooling fan is working or not, several test case have been defined. Below show the test case, the actual result and the result obtained.

No	Test case	Actual result	Result
1	Turn on the power supply for the water pump and call the function Pump1_On() at Actuators.py as shown in Figure 5-F53. .	The relay should complete the circuit and water pump 1 should turn on.	The water pump 1 do turn on as actual result.
2	Turn on the power supply for the water pump and call the function Pump1_Off() at Actuators.py as shown in Figure 5-F53.	The relay should cut the circuit and water pump 1 should turn off.	The water pump 1 do turn off as actual result.

3	Turn on the power supply for the water pump and call the function Pump2_On() at Actuators.py as shown in Figure 5-F53.	The relay should complete the circuit and water pump 2 should turn on.	The water pump 2 do turn on as actual result.
4	Turn on the power supply for the water pump and call the function Pump2_Off() at Actuators.py as shown in Figure 5-F53.	The relay should cut the circuit and water pump 2 should turn off	The water pump 2 do turn off as actual result.
5	Turn on the power supply for the cooler fan and call the function CoolerFan_On() at Actuators.py as shown in Figure 5-F53.	The relay should complete the circuit and cooler fan should turn on.	The cooler fan do turn on as actual result.
6	Turn on the power supply for the water pump and call the function CoolerFan_Off() at Actuators.py as shown in Figure 5-F53.	The relay should cut the circuit and cooler fan should turn off	The cooler fan do turn off as actual result.

Table 5-0-1-T31: Test plan for actuators

In this project, two water pump in used in order to pump in and pump out water from the aquarium. One pump will be placed inside the aquarium to pump out the water, and another pump will be placed at the new water source and its pipe will be placed into the aquarium to pump in water to aquarium. The cooling fan is used to reduce the water temperature of the aquarium using a method called evaporative cooling method. Evaporative cooling method is concept which move the heat from the surface of the water by blowing air across the surface. By blowing air on surface of water the evaporation rate (Transition from liquid to gas) will increase. The water will release more energy as the rate transition from liquid to vapor increases. The cooling fan used in this project consist of two DC motor that will blow the air to the water surface of aquarium thus reducing its temperature. This evaporative cooling method is a very slow process, thus the temperature reduction rate also reduce in slower rate.

L) SIMPLE WATER LEAK SENSOR WITH RASPBERRY PI 3

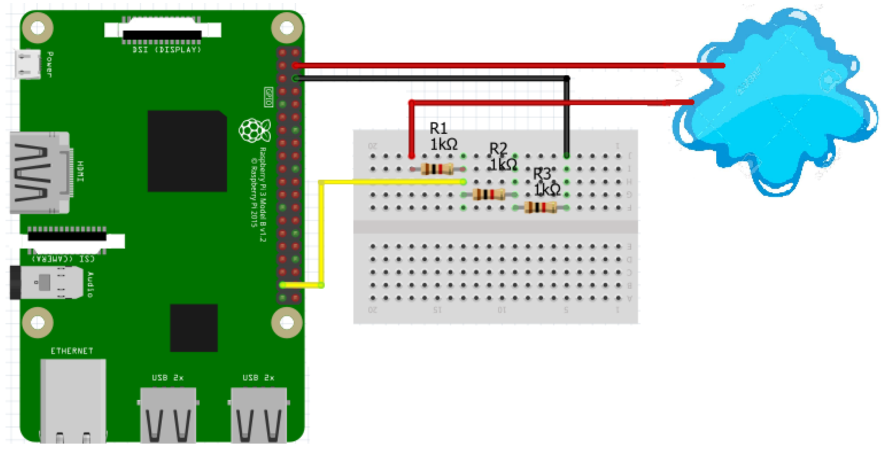
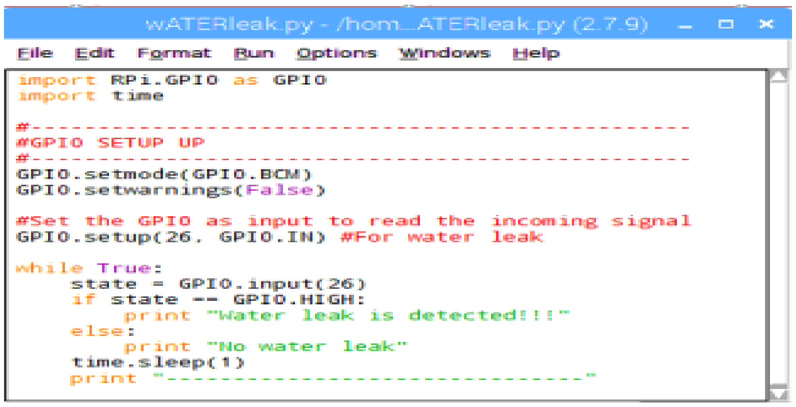


Figure 5-0-1F59: Simple water leak setup

Figure 5-0-1-F58 shows a simple water leak detector using Raspberry pi. We just setup two wires with 1 wire is connected to 5V pin of Pi and another is connected to the Pi's **GPIO26 (Pin 37)** as input. The 5V are used instead of 3.3V power supply because with higher current, the water can conduct electricity easily. But the Pi's GPI0 can only withstand input voltage of 3.3V. Hence, the 5V wire is fed into a voltage divider rule circuit as shown in Ultrasonic and Turbidity section with 3 resistor. If the circuit is completed and the Pi's GPIO read HIGH it indicate that there is water leak. If the circuit is not completed, Pi's GPIO will be remain LOW indicate that here is no water leak.


wATERleak.py - /hom...ATERleak.py (2.7.9)
File Edit Format Run Options Windows Help

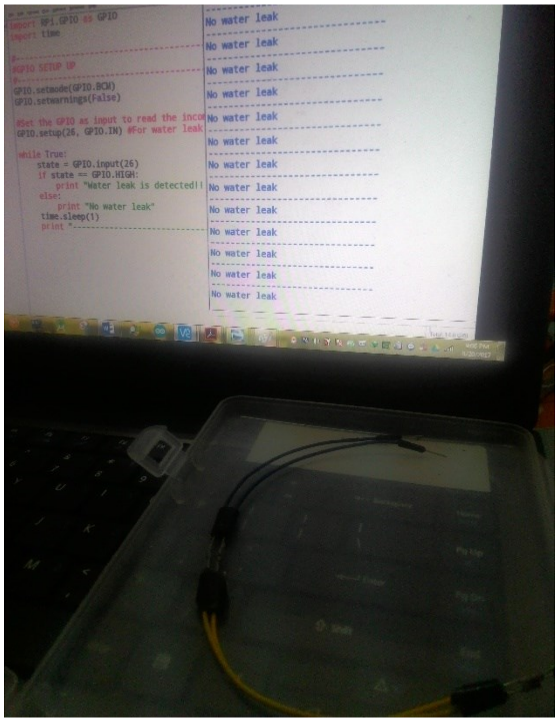
```
import RPi.GPIO as GPIO
import time

#---------------------------------------------------------
#GPIO SETUP UP
#---------------------------------------------------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Set the GPIO as input to read the incoming signal
GPIO.setup(26, GPIO.IN) #For water leak

while True:
    state = GPIO.input(26)
    if state == GPIO.HIGH:
        print "Water leak is detected!!!"
    else:
        print "No water leak"
    time.sleep(1)
    print "--------------------------------"
```

Figure 5-0-1-F60: Code to test water leak sensor

To test the DIY water leak sensor working properly or not, test plan with two test case has been define. I run the code for the water leak sensor and pour some water on the water leak sensor to check whether it can detect the water and output the alert message or not. Below shows the test cases, actual result and result obtained from the test.

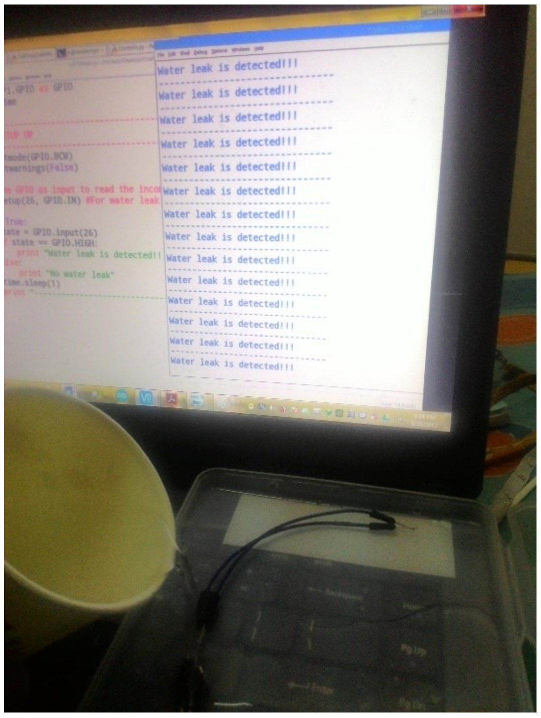
No	Test case	Actual result	Result
1	Do not pour water on the water leak sensor.	The water leak sensor circuit does not completed and the GPIO26 will receive low signal and output the message "No water leak"	No water leak message is display on python shell **(Figure 5-F56)**
2	Pour water on the water leak sensor.	The water leak sensor circuit will completed and the GPIO26 will receive high signal and output the message "Water leak detected"	Water leak detected message display on the python shell. (**Figure 5-F57**)

Table 5-0-1-T32: Test plan for water leak sensor



```
import RPi.GPIO as GPIO
import time

#-----------------------------------------
#GPIO SETUP UP
#-----------------------------------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Set the GPIO as input to read the incom
GPIO.setup(26, GPIO.IN) #For water leak

while True:
    state = GPIO.input(26)
    if state == GPIO.HIGH:
        print "Water leak is detected!!
    else:
        print "No water leak"
    time.sleep(1)
    print "-----------------------------------
```

```
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
--------------------------------
No water leak
```



```
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
--------------------------------
Water leak is detected!!!
```

Figure 5-0-1-F-61: No water leak **Figure 5-0-1-F-62: Water leak detected**

In this project the pair of many jumper wires are used as the water detect sensor which are place around the aquarium. Below shows the picture of the jumper wire used as water leak sensor in the project,

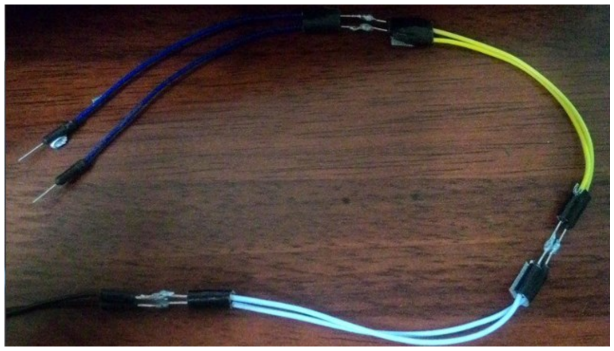


Figure 5-0-1-F-63: Prototype of water leak

L) LOGITECH HD WEBCAM C270

Figure 5-0-1-F64: **C270**

In this project I'm using Logitech C270 web camera to live stream the video of the aquarium to the website in order to check the aquarium condition remotely. Besides I also will be using the embedded microphone in C270 in order to control the system using the Voice Control apllication

Below shows the camera specification of C270,

Connection Type	Corded USB
Focus Type	Fixed
Field of View (FOV)	60°
Optical Resolution (True)	1280 x 960 1.2MP
Video Capture (4:3 SD)	320x240, 640x480, 800x600
Video Capture (16:9 W)	360p, 480p, 720p,

Table 5-0-1-T33: Specification C270

<u>Connection and configuration of C270 Logitech Camera with Raspberry Pi 3</u>

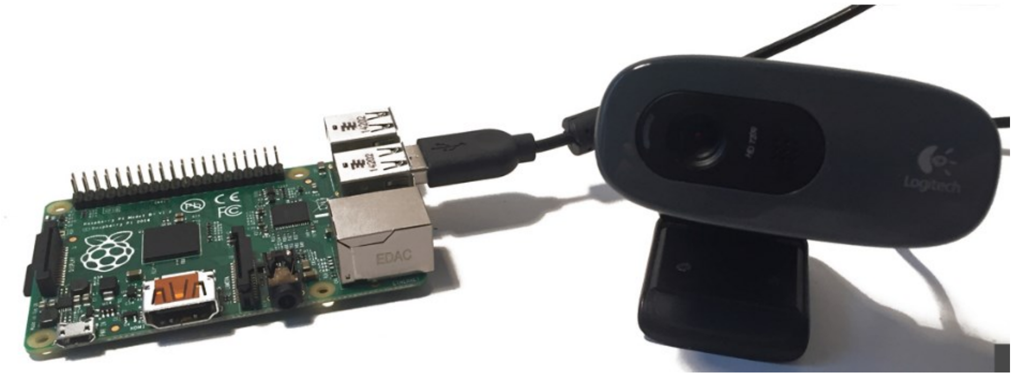
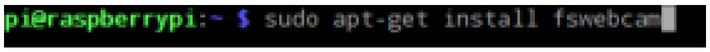


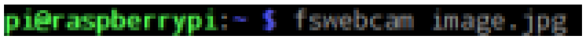
Figure 5-0-1-F-65: Connection between C270 and Raspberry Pi

Figure 5-0-1-F64 shows the how the C270 Logitech camera are connect to the Raspberry Pi 3. Besides the connection, a library is need to be download and install in Pi in order to test the USB camera. First download and install fswebcam library using the following command in Pi's terminal,



```
pi@raspberrypi:~ $ sudo apt-get install fswebcam
```

Once finish installed, now picture can be captured using the C270 USB camera. Just enter fswebcam followed by a file name then the image will be captured. For example,



```
pi@raspberrypi:~ $ fswebcam image.jpg
```

Below shows the picture captured by the C270 USB camera when the above command been executed.

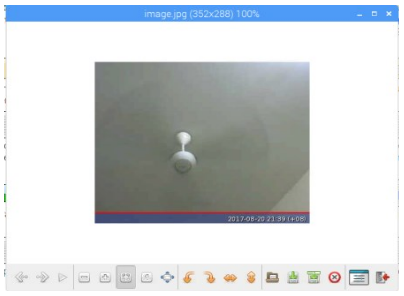


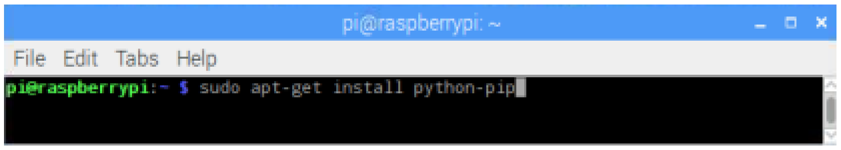
Figure 5-0-1-66: image.jpg

So now we can conclude that the C270 USB webcam working perfectly because it able to capture picture using the fswebcam library. In this project, camera are used to stream the video of the aquarium to the website in order the user can monitor their aquarium remotely. To stream live video to website the fswebcam library cannot be used .This is because fswebcam are only used to capture picture with USB cameras. We need to host a video server in the Pi in order to live stream the video to the local website. The UV4L Streaming Server has been used in this project in order to do the live stream to the local website. The C270 USB webcam also used as microphone to listen to the voice command by the user and control the system based on the command using an offline neural network voice recognition application called Snowboy. The installation and setup process of the UV4L server and Snowboy application in Pi will be discuss in the following software components section.

5.0.2 Software Components

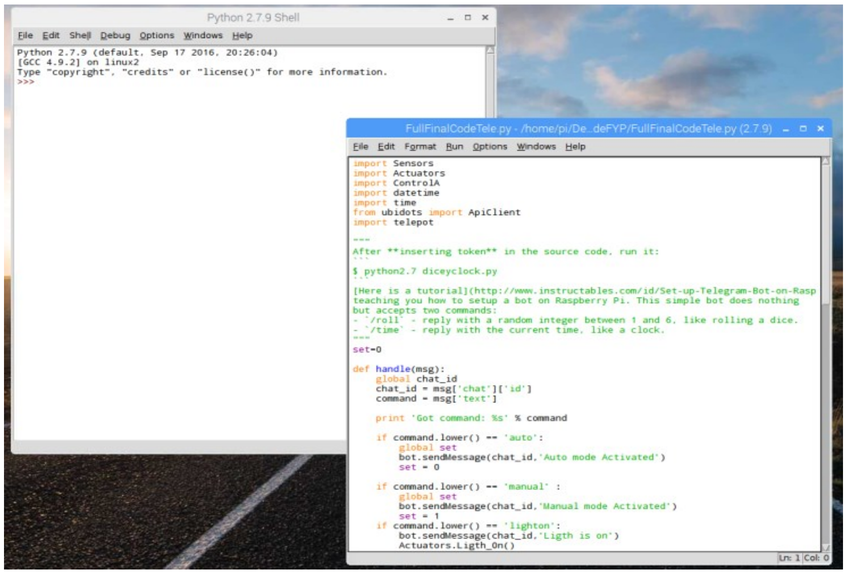
A) Python 2 (IDE)

In this project, Raspberry Pi 3 is used as central controller to read the sensor values, control the actuators such as water pump, light and received command from Web ,Telegram mobile application and Voice Control application. The main programming language used in this project is Python and it is debug using the Python 2 shell. Initially, the Python 2 packages is not available together with Raspbian, the operating system of Raspberry Pi 3. We have to install the python separately in order to use it. To get the Python 2 ide, execute the following command below at Pi's terminal,



```
pi@raspberrypi:~ $ sudo apt-get install python-pip
```

Below show the interface of Python 2 shell,


Python 2.7.9 Shell
File Edit Shell Debug Options Windows Help
FullFinalCodeTele.py - /home/pi/De..deFYP/FullFinalCodeTele.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>>
```

```
import Sensors
import Actuators
import ControlA
import datetime
import time
from ubidots import ApiClient
import telepot

"""
After **inserting token** in the source code, run it:
```
$ python2.7 diceyclock.py
```
[Here is a tutorial](http://www.instructables.com/id/Set-up-Telegram-Bot-on-Rasp
teaching you how to setup a bot on Raspberry Pi. This simple bot does nothing
but accepts two commands:
- `/roll` - reply with a random integer between 1 and 6, like rolling a dice.
- `/time` - reply with the current time, like a clock.
"""
set=0

def handle(msg):
    global chat_id
    chat_id = msg['chat']['id']
    command = msg['text']

    print 'Got command: %s' % command

    if command.lower() == 'auto':
        global set
        bot.sendMessage(chat_id,'Auto mode Activated')
        set = 0

    if command.lower() == 'manual' :
        global set
        bot.sendMessage(chat_id,'Manual mode Activated')
        set = 1
    if command.lower() == 'lighton':
        bot.sendMessage(chat_id,'Ligth is on')
        Actuators.Ligth_On()
```

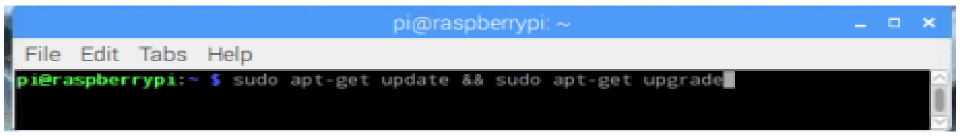
Figure 5-0-2-F1: Python 2

B) MySQL Database

Figure 5-0-2-F2: Logo of MySQL

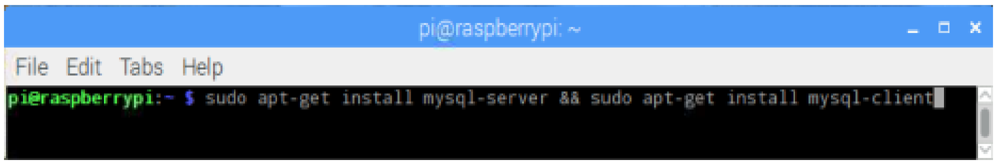
MySQL is an open source RDBMS developed by the Oracle Corporation which written in C and C++ programming language. It is a very popular database in the industry mainly because it is fast, secure and easy to use. Large scale website like Google, Facebook and even YouTube are using MySQL as database. Besides that, is it also known as fundamental components of LAMP open source web application software stack where LAMP stands for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is selected as databases in this project mainly because is it free to download, and easy to use. The steps on how to install MySQL database in Pi will be shown below.

1st step: Execute the following command in terminal, to ensure updates and upgrades are installed



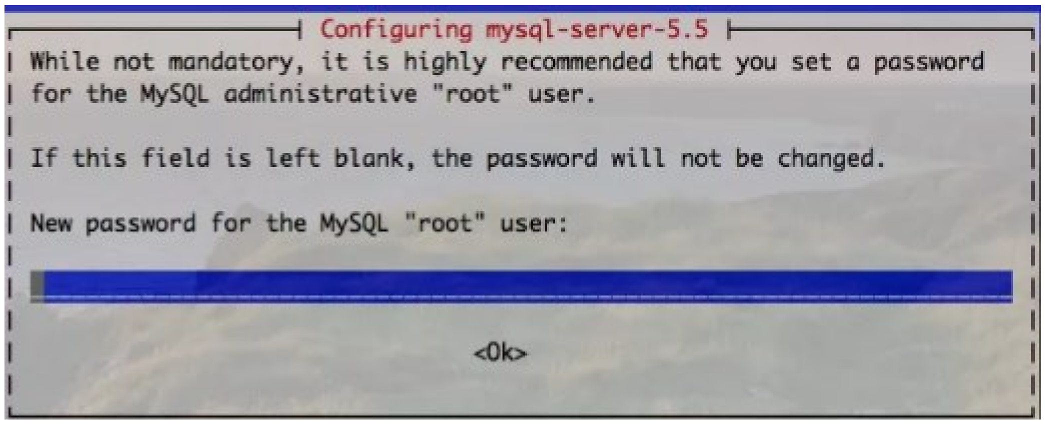
```
pi@raspberrypi:~ $ sudo apt-get update && sudo apt-get upgrade
```

2nd step: Once all updates finished, run the following commands to install MySQL server and client to Pi.



```
pi@raspberrypi:~ $ sudo apt-get install mysql-server && sudo apt-get install mysql-client
```

3rd **step:** After run the above command, the installation will pop up GUI message title configuring mysql-server-5-5 asking for set the password for MySQL. Set the password as root, the default password. After a while, the installation process will finish. Below shows the figure of the GUI message.



```
Configuring mysql-server-5.5
While not mandatory, it is highly recommended that you set a password
for the MySQL administrative "root" user.

If this field is left blank, the password will not be changed.

New password for the MySQL "root" user:


                              <Ok>
```

Figure 5-0-2-F3: GUI message

Once the installation process completed, we can use MySQL to create database and store data. In this aquarium monitoring system, database called RV_Aquarium is created which contain four tables ActuatorLog, Global, SensorLog and user table. Database is needed in this project in order to keep track of sensor value every 10 minutes, the feed time, and light control time and the to store the date time. Besides, it is also use as the communication tools between the Flask web server, Telegram application and Voice Control application which are located at different files in the Pi because some of the variables are need to share among them. Below table shows the database tables created for this aquarium monitoring system and their function.

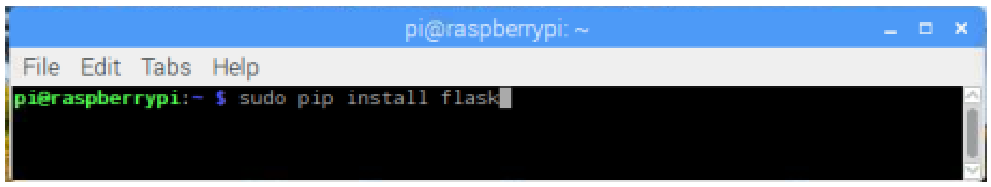
Database table name	Function
ActuatorLog	The function of this tables is to store feed time and light control time for the auto mode. For the auto mode, the system have to feed the fish and on and off the light by its own for certain time set by the user. So this table will be storing those data.
SensorLog	The function of this tables is to store the sensor value to the database every 10 minutes, such as temperature value, water level value and turbidity value. These are for reference purpose for user so that user can trace back sensor values on particular date and time.
user	The function of this table is store the user info that log in into the website. This table store username, email, and their password. The password are hashed before storing onto database for safety purpose.
Global	The function of this table is to help share data between the three control mode website, Telegram and Voice Control. Global variable cannot be used as to share data in between different files. Hence this table are create to act as communication tool between the three controls modes of the system.

Table 5-0-2-T1: RV_Aquarium database's tables

C) FLASK WEB DEVELOPMENT (Server Side of website)

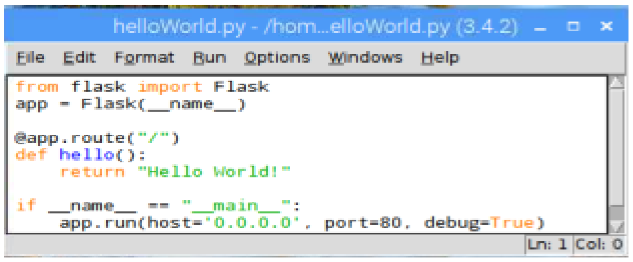
Figure 5-0-2-F4: Flask web framework logo

One of the control mode in this aquarium monitoring system is website, therefore a web server is needed in order to run the aquarium monitoring system. Flask is chosen as webserver in this project because it is easy and flexible to use. Flask basically is a micro web application framework which contain certain tools and libraries that help to build web application. It is written in Python programming language and it is based on Werkzeug and Jinja2 template engine. The primary function of the Flask is to provide tools and libraries for the web developments. But it do include a web server to be used for testing and developments. It is very light, simple and flexible web framework compare to other Python based web server such as Django and Pyramid. Below shows how to install the flask in pi.

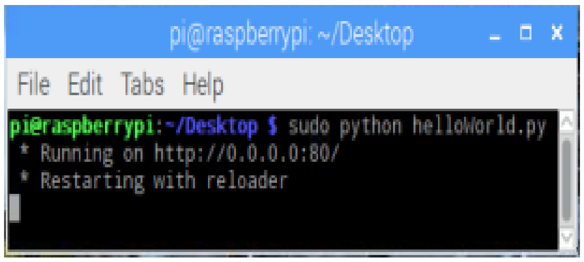


```
pi@raspberrypi:~ $ sudo pip install flask
```

To create a simple flask web server, the following test code helloWorld.py are written and run at the Pi’s terminal. The web server will automatic run at the port 80 whenever the helloWorld.py is run. Then open a browser and navigate to Pi's local ip address followed by the port number 80 for example 192.16.167.146:80 to show the output of the web.

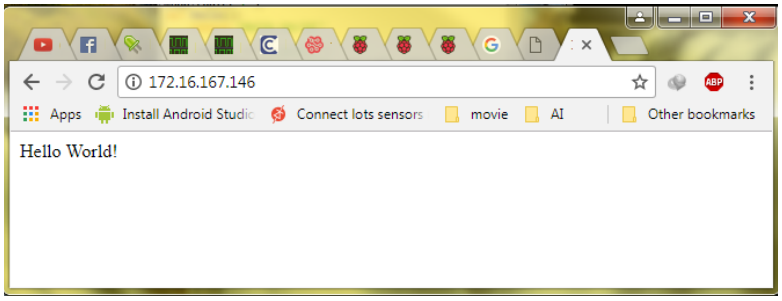

helloWorld.py - /hom...elloWorld.py (3.4.2)
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```



```
pi@raspberrypi:~/Desktop $ sudo python helloWorld.py
 * Running on http://0.0.0.0:80/
 * Restarting with reloader
```

Figure 5-0-2-F5: helloWorld.py server **Figure 5-0-2-F6: Run the server**



Hello World!

Figure 5-0-2-F7: The output of server at local IP: 80 at browser

D) HTML, CSS, JAVASCRIPT (Client side Programming Language for website)

i) HTML (Hypertext Markup Language)

Figure 5-0-2-F8: HTML

HTML stands for Hypertext Markup Language which simply means it is a language for describing the web-pages using ordinary text. It is also known as the skeleton of a webpage which provide structure for the webpages. The web browser such as Chrome and Mozilla are used to read the HTML documents and display them. Below shows the basic structure of HTML document,

```
<!DOCTYPE html> → Define this document to be HTML5
<html>            → Root element of an HTML page
   <head>   ⎫      Contain meta information of documents
   </head>  ⎭
    <body>  ⎫      Contain the page contain of the web
    </body> ⎭
</html>
```

In this project website are used to control the aquarium monitoring system hence HTML5 are used and it considered as backbone of the webpage. HTML itself is not enough in order to create a good looking and functional website. HTML are used along with CSS and JavaScript to create a user friendly and functional website. The CSS and JavaScript will be discussed in the next sections.

ii) CSS (Cascading Style Sheet)

Figure 5-0-2F9: CSS

CSS stands for Cascading Style Sheet which is a language which defined the style of a web page including its colors, design and layouts. It is also known as the skin for webpage which makes the webpage looks more beautiful. Since in this project a website is hosted, CSS is very important to arrange the elements of HTML and make the webpages more beautiful and attractive for the users. Bootstrap CSS is used in this project to design the style of the webpage.

Figure 5-0-2-F10: Bootstrap logo

iii) JAVASCRIPT (JS)

Figure 5-0-2-F11: JS

JavaScript also known as JS is a programming language which enable the dynamic changes on the webpage for example, updating content, real time graph plotting, and displaying images and so on. It is one of the three core components of web technologies follow by HTML and CSS. A static webpage is considered boring and not effective. With JavaScript we can make the webpage more effective and productive. Even navigate to new webpage after a button is pressed is the worked of the JavaScript. In this project, JSON (JavaScript Object Notation) is used to control the aquarium monitoring system by interchange the data in form of text between the client side and the server side Flask. Besides that, real time charts and graph are plotted on the website using a JavaScript library called is FusionCharts. FusionCharts is an open source JavaScript based third party library which help to plot graphs and charts in this project.

Figure 5-0-2F12: FusionCharts logo

E) UV4L Streaming Server

UV4L Streaming Server where UV4L stands for user space Video4Linux is a collection of device driver and an API for supporting real-time video capture on Linux system. It have a customizable web UI which can be used for encrypted live bidirectional data, audio and video streaming and conferencing over the web. Below steps shows how to install the UV4L library in Pi,

1st: Run the following command in Pi's terminal, to install the uv4l package in Pi

```
$ sudo apt-get install uv4l uv4l-server uv4l-uvc uv4l-server uv4l-webrtc uv4l-xmpp-bridge
```

2nd: Open the config file and change a line driver = uvc and save the file

```
$ sudo vim /etc/uv4l/uv4l-raspicam.conf
```

Change driver = uvc

Now restart your system and after restart , open you browser and navigate to your IP local address follows by the port number 8090 , for example in my case 172.16.167.146:8090 .Now you can see the webpage of UV4L Streaming Server. Just navigate to the http://172.16.167.146:8090/stream now you can see the live stream video using your usb camera. Below shows the webpage of UV4L Streaming Server.

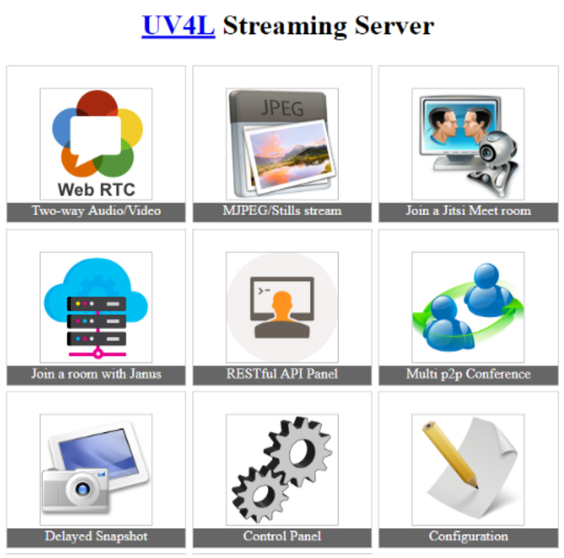


Figure 5-0-2-F13: UV4L

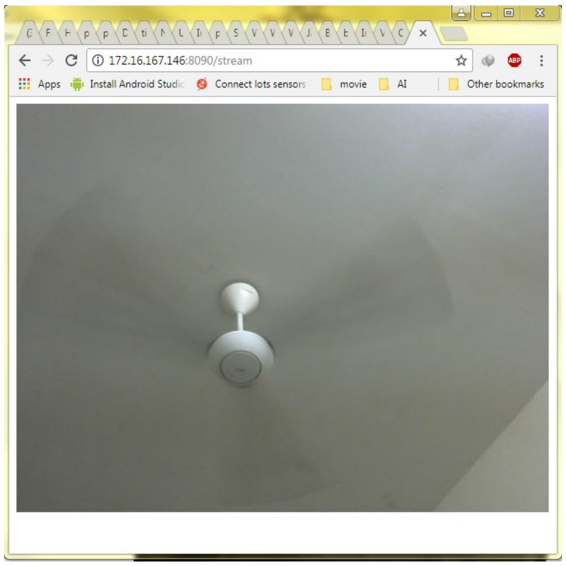


Figure 5-0-2-F14: Output of http://172.16.167.146:8090/stream

In this aquarium monitoring system, the live stream is used in order the user can see and monitor the aquarium although they are not around to take care their aquarium. For example, if they went for vacation, they still can view what is going on to their aquarium through the live stream feature of this system.

G) SNOWBOY-HOTWORD DETECTION ENGINE

Figure 5-0-2-F15: Snowboy logo

Snowboy is a highly customizable hotword detection engine that is embedded real time and always listen even when it is offline (without internet). A hotword means is a keyword that a computer continuously listens for as a signal to trigger other actions. Unlike Google's OK Google hotword detection engine which using ASR (Automatic Speech Recognition) that needs to use internet to detect the hotword , Snowboy is it is powered by deep neural networks that can even works in offline mode to.

To install the Snowboy library, run the following command,

```
$ sudo npm install --save snowboy

$sudo apt-get install swig3.0 python-pyaudio python3-pyaudio sox

$pip install pyaudio

$ sudo apt-get install libatlas-base-dev
```

Below shows steps how to create a particular trained model of hotword,

1st step: Pick a hotword, for example “Raash”

2nd step: Record the hotword 3 times using your machine’s microphone

3rd step: Then submit the audio files through Snowboy’s RESTful API Calls which the submitted audio file will used to train the model and the trained model file with extension .pmdl will be return. Then use the file to do some action based on the hotword.

In this project, Snowboy is used to control the aquarium actuators such as light, RGB light, Stepper motor to feed the fish, water pump and cooling fan when a certain hotwords is detected. There is approximately 15 hotwords that been trained in order to control different actuators in this the aquarium monitoring system.

G) Telegram Bot API

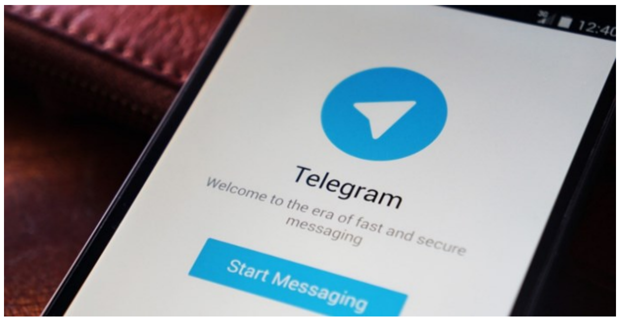


Figure 5-0-2-F16: Telegram app

Bots are third party application that run inside Telegram. User can send the bot some messages, commands and the bot will reply based on the fixed question-answer module. Below shows the steps how to install Telegram Bot API,

1st step: Download Telegram app in your mobile or system

2nd step: Then start BotFather and created a new bot

3rd step: Configure the name of the new bot and obtain the access token from BotFather

4rd step: Install teleport using the command **$ sudo pip install teleport**

5th step: Create a python code based on the example from Telegram bot official website and paste the token access in the python code. Run the python code and send the message from the Telegram massagers. The bot will reply to your message based on what you have set at the python code.

In this project I ‘m using Telegram bots API to send the message and command from my smartphone to the Pi to control the actuators of aquarium monitoring system such as light and feeder .The message that I send from Telegram app will pass through the Telegram server then to the custom Telegram bot that I installed in the Raspberry pi. So, this application is only works when the Raspberry Pi is online that is connected to internet.

H) REMOT3.IT

Figure 5-0-2-F17:remot3.it logo

The remot3.it which previously known as Weaver is a service which help the user to connect to their Pi easily and securely from any mobile apps and browser across the internet. By default, website that host by the Pi can only be access by devices which are in same local network with the Pi. One of the most common way to make Pi's website global by doing some port forwarding at the router where the Pi are connected. But in this project, we have no access to the router in order to do port forward to make the website global. Hence the third party software called remot3.it is used instead of using port forwarding for Network Address Translation (NAT). Below shows the steps on how to install and use the remot3.it service.

1st step: Sign up for a free remot3.it account at **https://www.remot3.it**

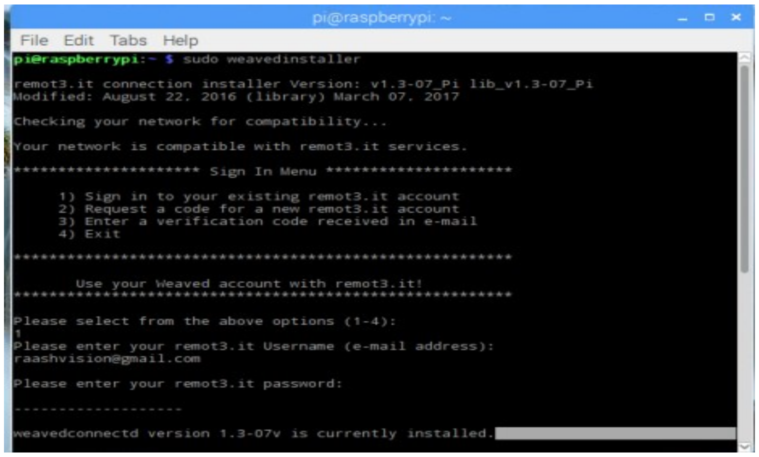
2nd step: After sign up, run the following command to update your pi and install remot3.it package.

```
$ sudo apt-get update

$ sudo apt-get install weavedconnectd
```

3rd step: Then once completed update and install the remot3.it package, set-up the remot3.it using the following command.

```
$ sudo weavedinstaller
```

4rd step: Next, a sign up menu will appear in your terminal right after you run the above command as shown in below Figure 5-F74. Enter option 1 to sign up for the existing account. Then enter your username and password of remot3.it account.



pi@raspberrypi: ~

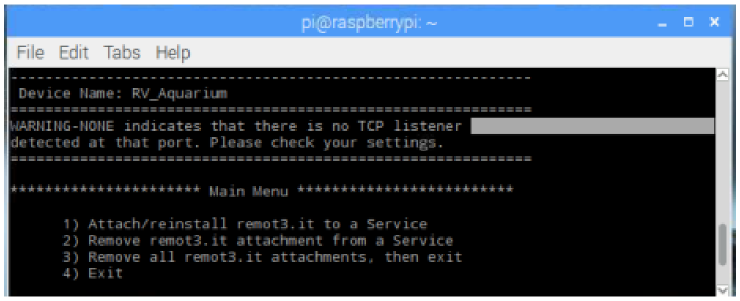
File Edit Tabs Help

```
pi@raspberrypi:~ $ sudo weavedinstaller

remot3.it connection installer Version: v1.3-07_Pi lib_v1.3-07_Pi
Modified: August 22, 2016 (library) March 07, 2017

Checking your network for compatibility...

Your network is compatible with remot3.it services.

********************* Sign In Menu *********************

     1) Sign in to your existing remot3.it account
     2) Request a code for a new remot3.it account
     3) Enter a verification code received in e-mail
     4) Exit

********************************************************

       Use your Weaved account with remot3.it!
********************************************************

Please select from the above options (1-4):
1
Please enter your remot3.it Username (e-mail address):
raashvision@gmail.com

Please enter your remot3.it password:

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

weavedconnectd version 1.3-07v is currently installed.
```

Figure 5-0-2-F18: Sign up menu

5th step: Then you will be asked to give name for you Pi. In my project I name it RV_Aquarium

6th step: Next select the menu item 1 to Attach/reinstall remot3.it to a service.



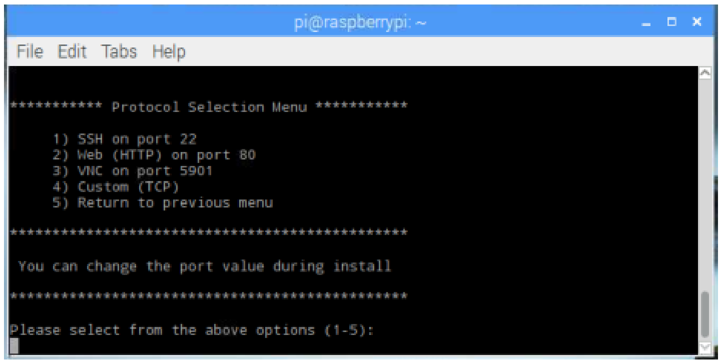
pi@raspberrypi: ~

File Edit Tabs Help

```
-----------------------------------------------------------
 Device Name: RV_Aquarium
===========================================================
WARNING-NONE indicates that there is no TCP listener
detected at that port. Please check your settings.
===========================================================

********************* Main Menu ************************

      1) Attach/reinstall remot3.it to a Service
      2) Remove remot3.it attachment from a Service
      3) Remove all remot3.it attachments, then exit
      4) Exit
```

Figure 5-0-2-F19: Main Menu

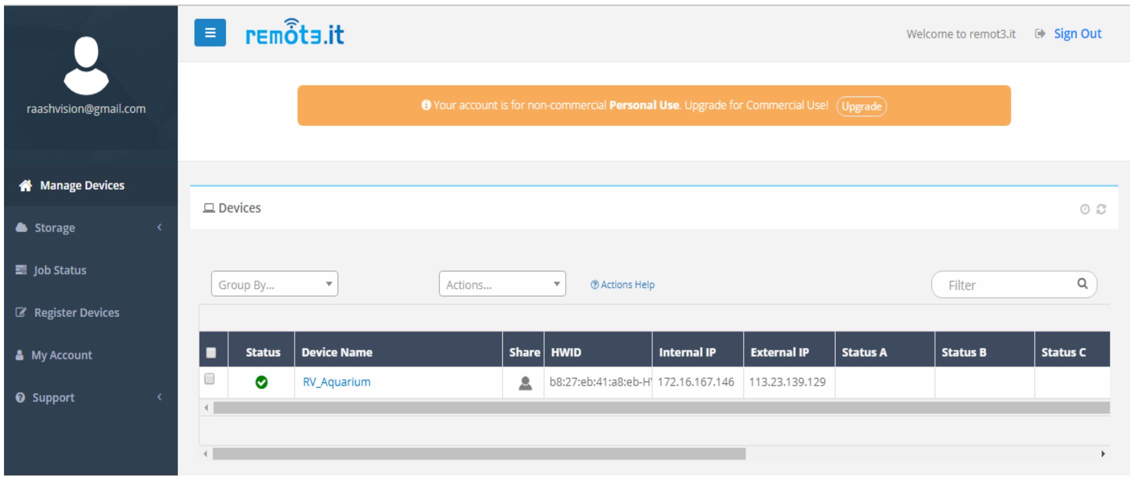
7th step: Then you will see the Protocol Selection Menu as below. Since we want to make the website in this project which hosted at port number 80 to access globally across the internet, number 2 Web(HTTP) on port 80 will be selected. Once selected, exit by enter number 5.



```
*********** Protocol Selection Menu ***********

     1) SSH on port 22
     2) Web (HTTP) on port 80
     3) VNC on port 5901
     4) Custom (TCP)
     5) Return to previous menu

***********************************************

 You can change the port value during install

***********************************************

Please select from the above options (1-5):
```

Figure 5-0-2-F20: Protocol Selection Menu

8th step: Now go to the link https://www.remot3.it2.0k and login to your account. You will able to see your Pi in the Manage Device. Click on the "Device Name" to connect. Now you will get an IP at external IP tab. Copy and paste in to any link and it will navigate to your website at port 80 of your pi.



Your account is for non-commercial **Personal Use**. Upgrade for Commercial Use! Upgrade

Status	Device Name	Share	HWID	Internal IP	External IP	Status A	Status B	Status C
✓	RV_Aquarium		b8:27:eb:41:a8:eb-H	172.16.167.146	113.23.139.129			

Figure 5-0-2-F21: At Manage Device of my remot3.it account

5.1 System Integration

In previous section 5.0, all the hardware and software components are explained in detail with their connection with the Raspberry Pi separately. To integrate the hardware and software components to become a full system, all the parts are combined and the code file need to be run. In order to use the Telegram chat bot, the Pi are connected to internet. Below figure shows the block diagram of how the final product will be.

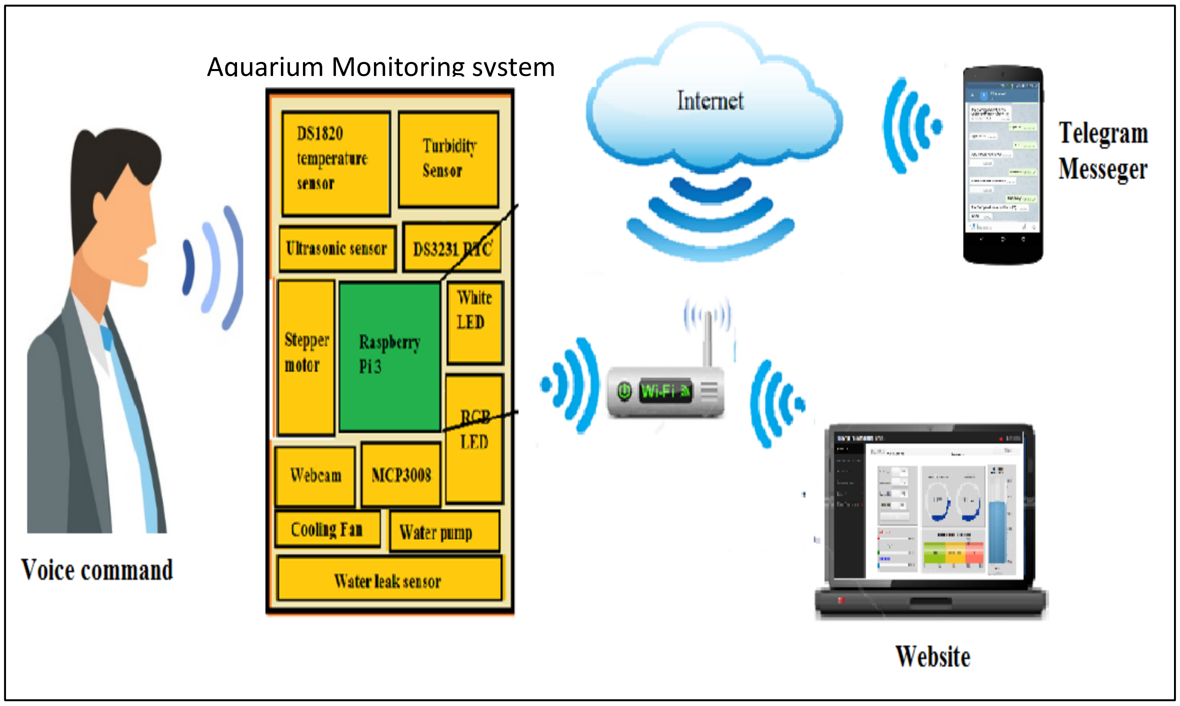


Figure 5-1-F1

The website will be the main control modes of this aquarium monitoring system between the user and the system follow by the Telegram chat bot and the Voice control application. The website is called main control mode of this system because, the website have ability to disable or enable the other two control mode. In this project, 7 file are need in order the aquarium monitoring system to work properly. It consist of four core files, follow by a web server file, Telegram bot file and Voice control file. The web server code consist of two part one is the server part and another is the system control part. Server part run as main thread whereas the system control part is run as child thread for the server part. Below shows the file that used in this project and their function. All the files are written in python language.

File name	Function
Actuators.py	Initialize the actuators configuration and contain function to use them, for example, Fish Feeder, Light control and so on.
ActuatorsLogSQL.py	Read the data such as Feed time and light control time from database and use them.
Sensors.py	Initialize the sensors configuration and contain function to use them, for example temperature sensor, turbidity sensor and water level sensor.
ControlA.py	Important file for auto mode system. Contain functions on how to control the system's actuators based on the sensors value and threshold value set by the user.
WebServer.py	This is the web server file of this project. The whole project's web page the server side code contain in this file. There are approximately 30 python functions inside this webserver file. This file also contain the main algorithm of this system that run as child thread for it. Refer to chapter 3 for more info about the algorithm.
Telegram.py	This file contain the Telegram bot file. Once run, it will keep waiting for a message from the user. All the commands and its reply are already set.
VoiceCommand.sh (inside contain python code)	This is a bash script file. In this script contain python file of this voice control application. This python file can only be execute at terminal along with its pmdl (refer to snowboy at section 5.0). So by making them as bash script it will be easy to execute.

Table 5-1-T1: Files used in this project

Chapter 6: Final Product

After the hardware and software part are integrated, the final product has successfully developed. Following figures shows the aquarium monitoring system overall picture and the website that are developed in this project.

6.1 Aquarium Monitoring system Hardware

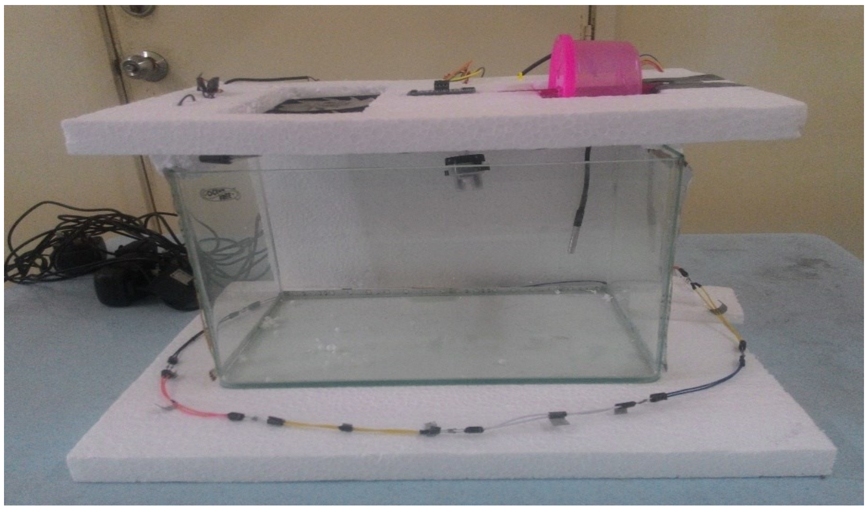


Figure 6-1-F1: Aquarium Monitoring system front view

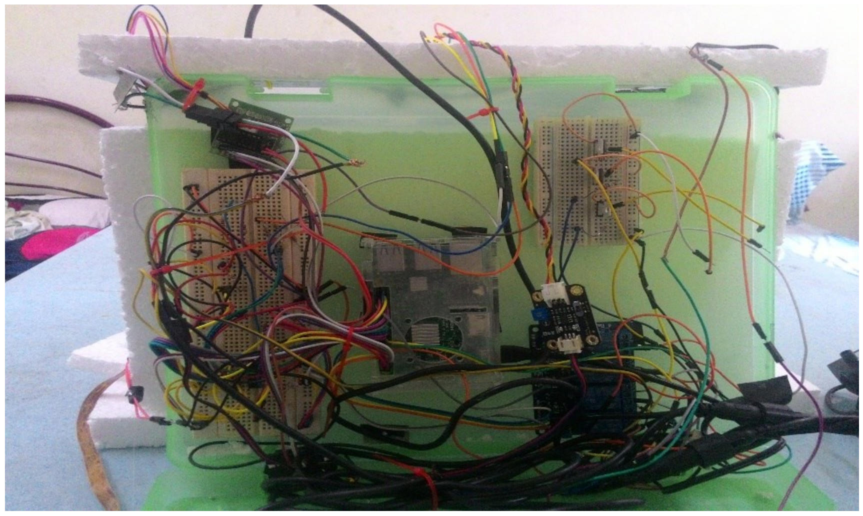


Figure 6-1-F2: Aquarium Monitoring system back view

6.2 Website

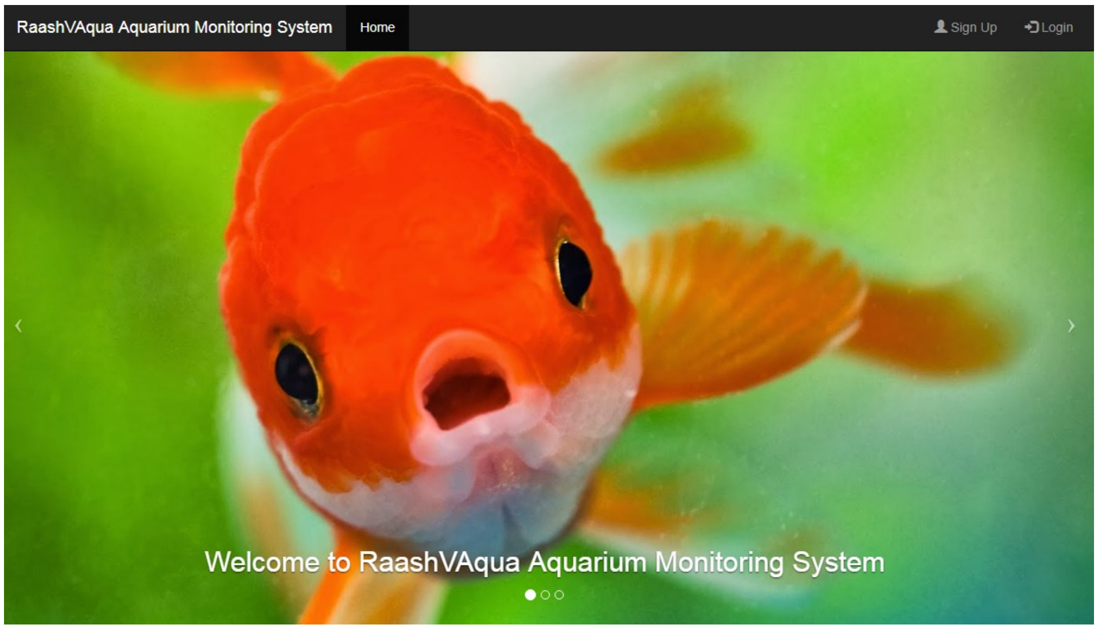

RaashVAqua Aquarium Monitoring System
Home
Sign Up
Login
Welcome to RaashVAqua Aquarium Monitoring System

Figure 6-2-F1: Home page

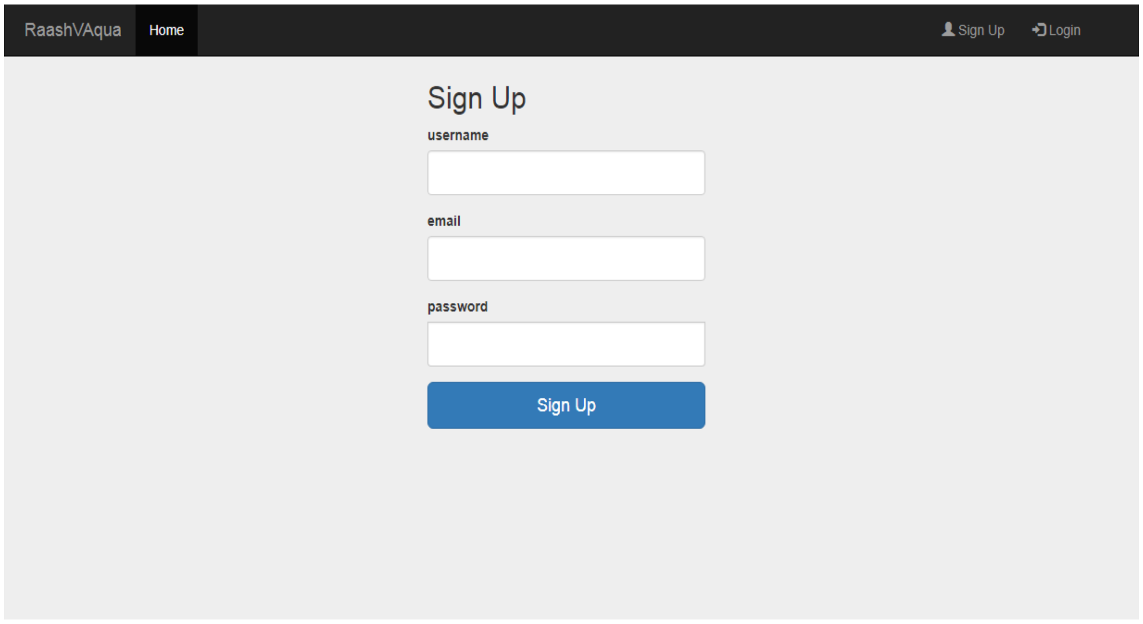


Figure 6-2-F2: Sign up page

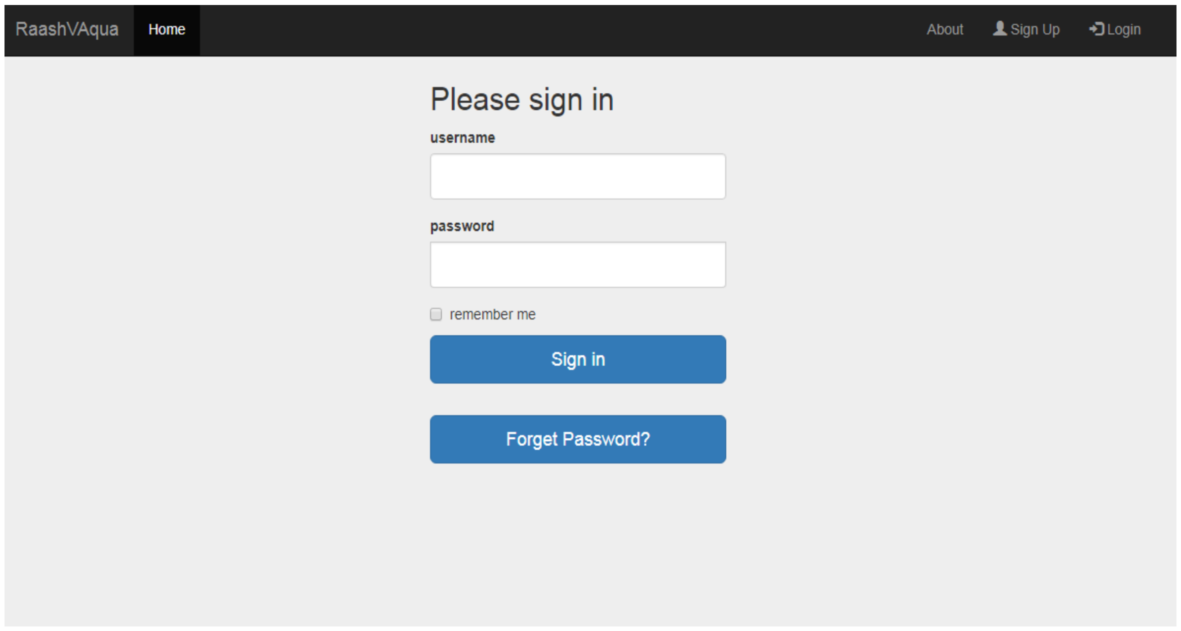


Figure 6-2-F3: Login page

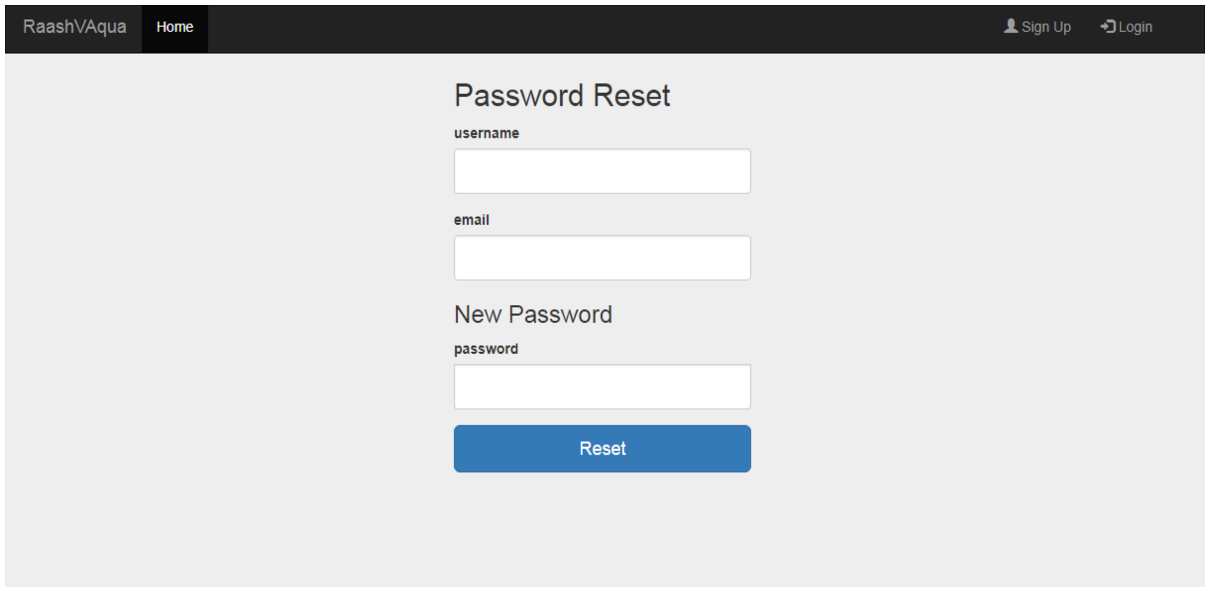


Figure 6-2-F4: Reset Password page

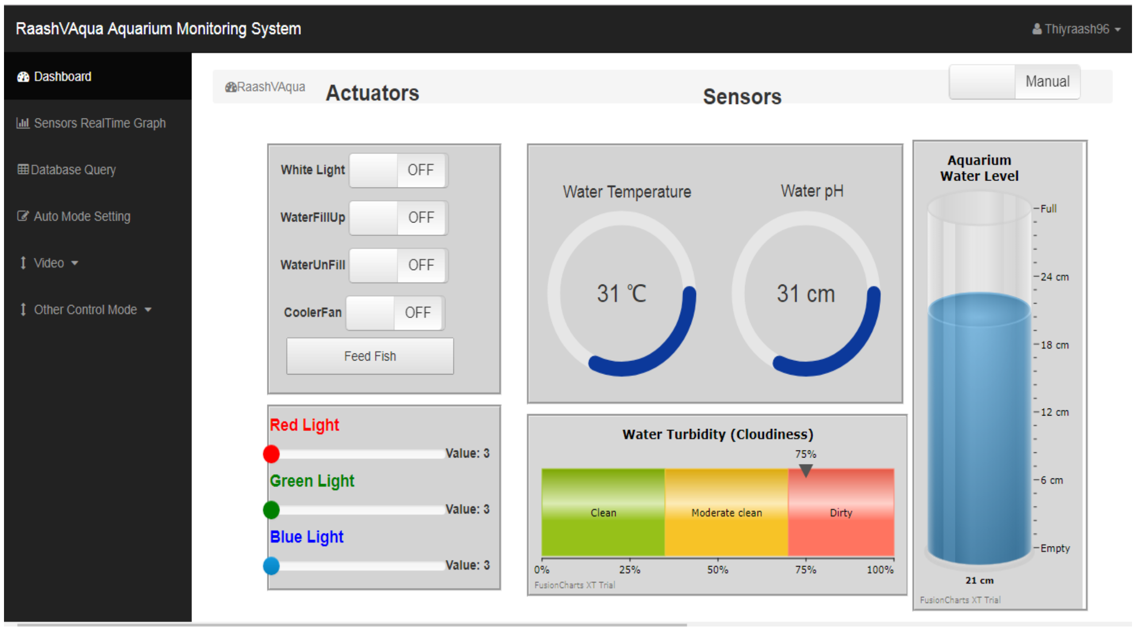


Figure 6-2-F5: Dashboard page

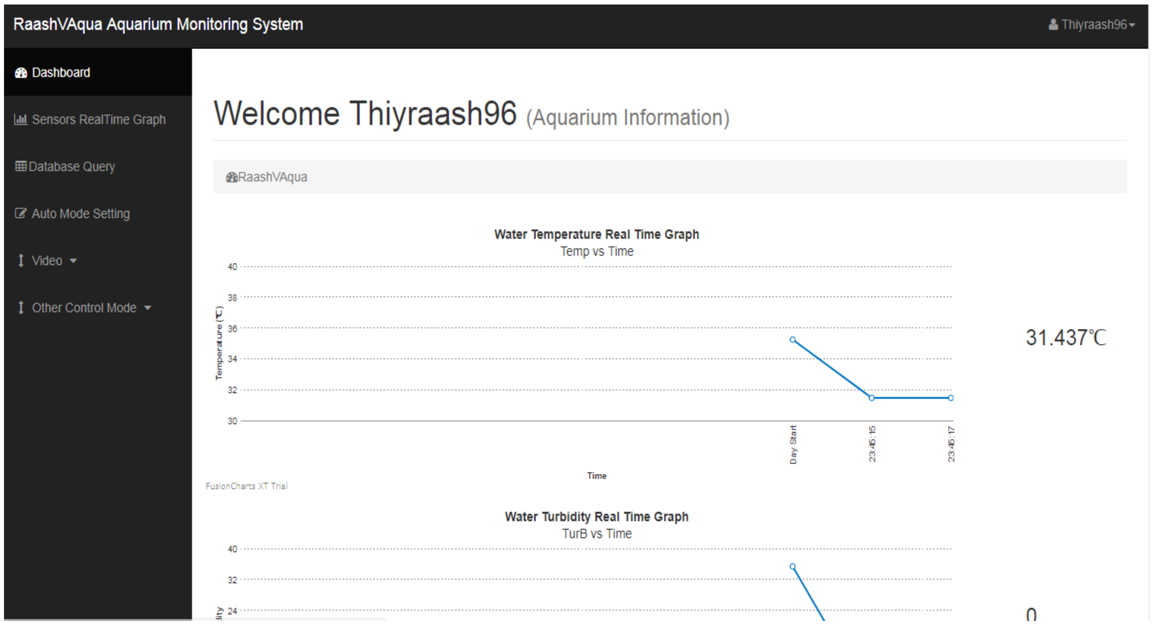
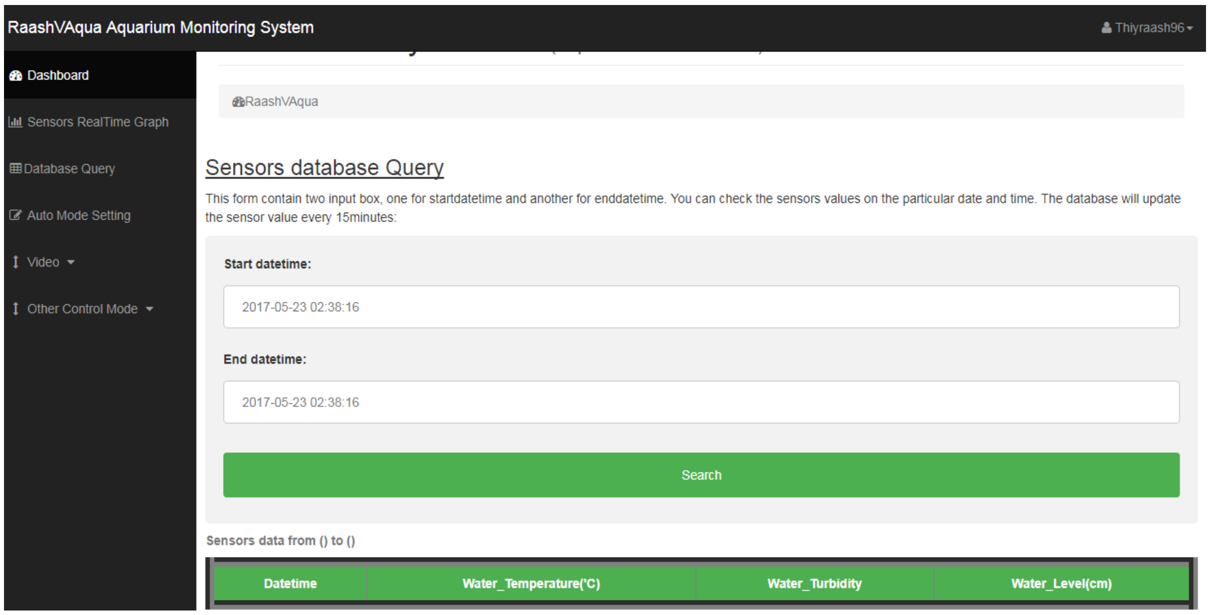


Figure 6-2-F6: Sensors RealTime Graph page



Sensors database Query

This form contain two input box, one for startdatetime and another for enddatetime. You can check the sensors values on the particular date and time. The database will update the sensor value every 15minutes:

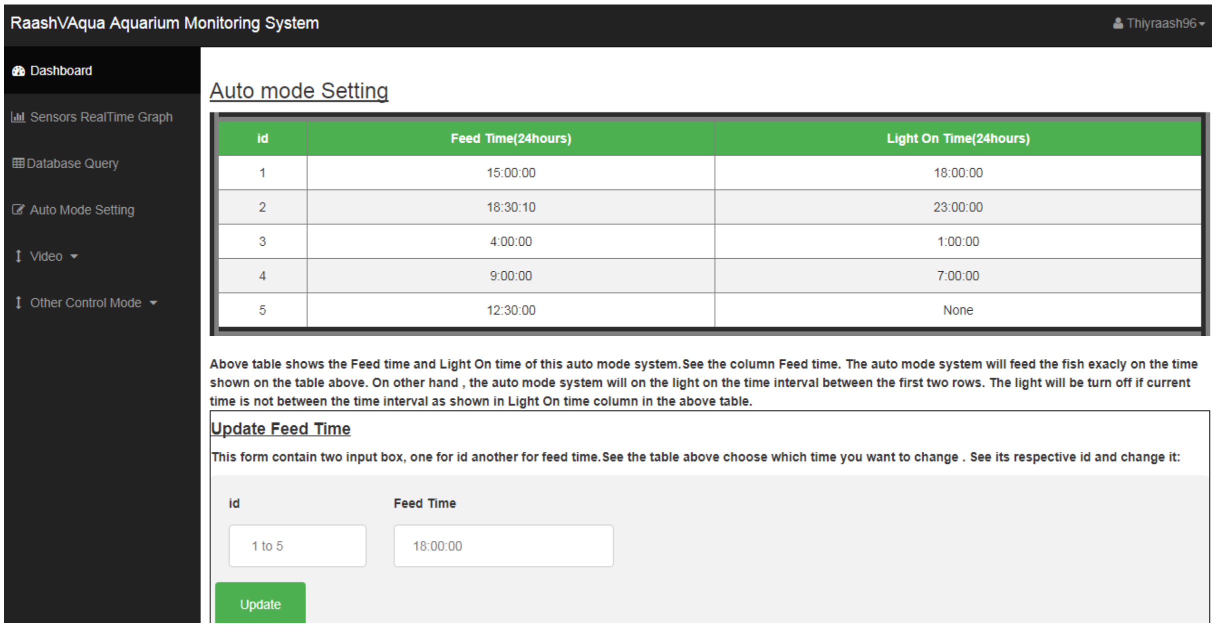
Start datetime: 2017-05-23 02:38:16

End datetime: 2017-05-23 02:38:16

Sensors data from () to ()

Datetime	Water_Temperature('C)	Water_Turbidity	Water_Level(cm)

Figure 6-2-F7: Database Query page



Auto mode Setting

id	Feed Time(24hours)	Light On Time(24hours)
1	15:00:00	18:00:00
2	18:30:10	23:00:00
3	4:00:00	1:00:00
4	9:00:00	7:00:00
5	12:30:00	None

Above table shows the Feed time and Light On time of this auto mode system.See the column Feed time. The auto mode system will feed the fish exacly on the time shown on the table above. On other hand , the auto mode system will on the light on the time interval between the first two rows. The light will be turn off if current time is not between the time interval as shown in Light On time column in the above table.

Update Feed Time

This form contain two input box, one for id another for feed time.See the table above choose which time you want to change . See its respective id and change it:

id: 1 to 5

Feed Time: 18:00:00

Figure 6-2-F8: Auto Mode Setting page

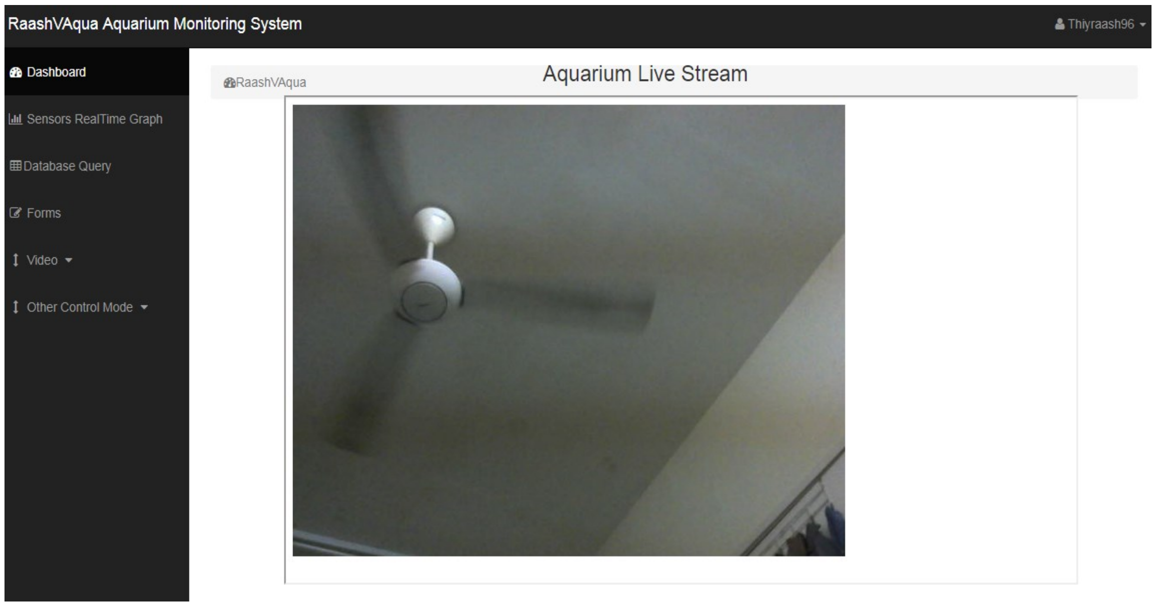
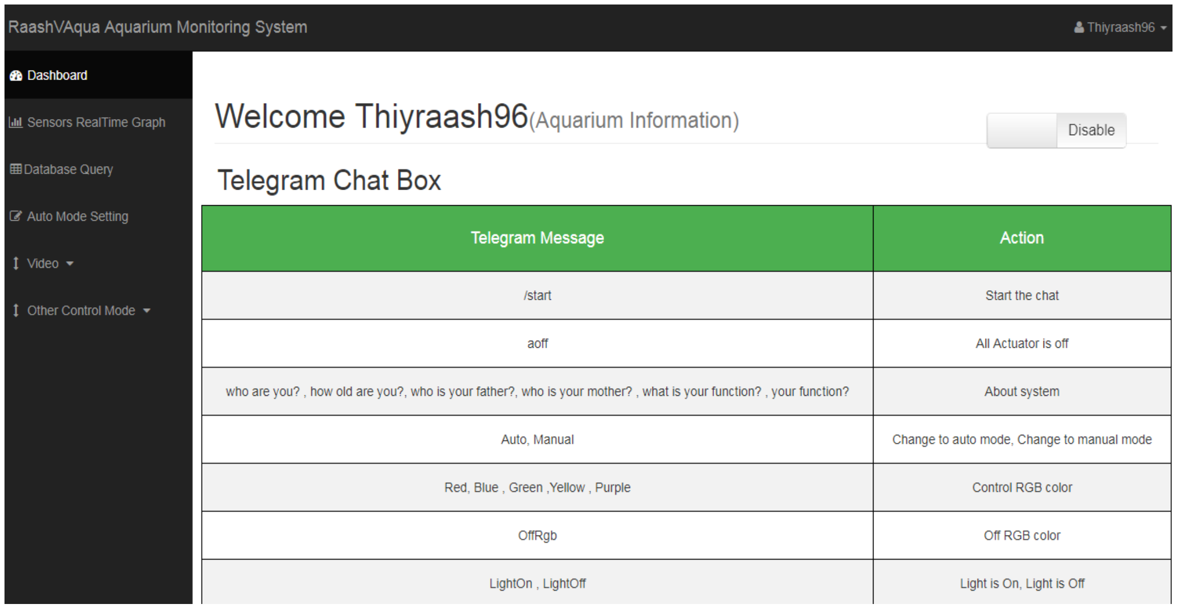


Figure 6-2-F9: Live Stream video page

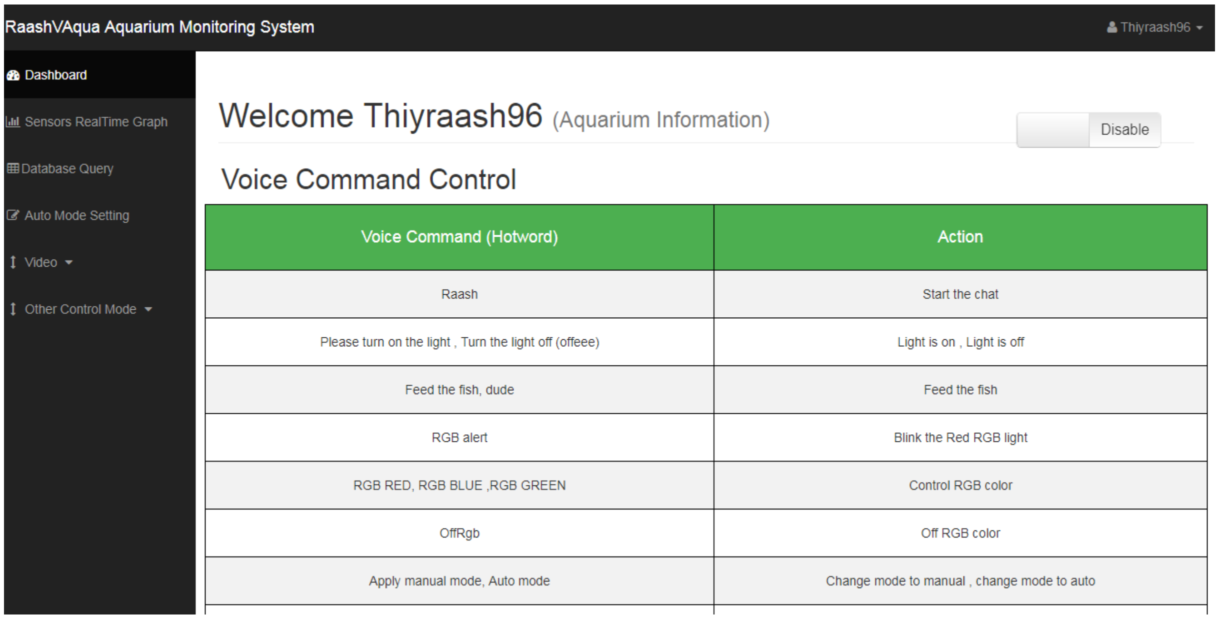


Welcome Thiyraash96(Aquarium Information)

Telegram Chat Box

Telegram Message	Action
/start	Start the chat
aoff	All Actuator is off
who are you? , how old are you?, who is your father?, who is your mother? , what is your function? , your function?	About system
Auto, Manual	Change to auto mode, Change to manual mode
Red, Blue , Green ,Yellow , Purple	Control RGB color
OffRgb	Off RGB color
LightOn , LightOff	Light is On, Light is Off

Figure 6-2-F10: Telegram Chat Bot page



Welcome Thiyraash96 (Aquarium Information)

Voice Command Control

Voice Command (Hotword)	Action
Raash	Start the chat
Please turn on the light , Turn the light off (offeee)	Light is on , Light is off
Feed the fish, dude	Feed the fish
RGB alert	Blink the Red RGB light
RGB RED, RGB BLUE ,RGB GREEN	Control RGB color
OffRgb	Off RGB color
Apply manual mode, Auto mode	Change mode to manual , change mode to auto

Figure 6-2-F11: Voice Command Control page

Chapter 7: Final Testing and Result Discussion

7.1 Website Testing

7.1.1 Result and discussion

In order to ensure that the website works properly, test plan with several test case has been defined. Since the website consist of few pages, unit test for every pages are needed in order to check all the feature in a page is working perfectly.

7.1.1 Unit Test 1: Home page

No	Event	Expected Result	Actual Result
1	Press the sign up button	Navigate to the sign up page	Same as expected result
2	Press the login button	Navigate to the login page	Same as expected result
3	Press the home button	Redirect to the home page	Same as expected result

Table 7-1-1-T1: Home Page Test Plan

7.1.1 Unit Test 2: Sign up Page

The first user register will automatically become the admin of the system. If another user try to register an account in the system, their activation code will be send to admin's email address and not theirs. The new user can only register if the admin allowed.

No	Event	Value	Expected Result	Actual Result
1	User enter information required to register account for access the website. Then click on sign up (First user)	Username:kokoi Email:kokoi@gmail.com Password :12345678	System will load for a while, then a message will display saying a link with an activation code have been seen to user's email. Navigate to the link to complete the sign up process.	Same as expected result
2	User navigate to the link send by the website to complete the sign up. (First user)	--	Navigate to the login page with successful message display at top of the page.	Same as expected result

3	User enter information required to register account for access the website. Then click on sign up (Second user)	Username:kokoi1 Email:kokoi1@gmail.com Password :12345678	System will load for a while, then a message will display saying a link with an activation code have been seen to the admin's email. Navigate to the link to complete the sign up process.	Same as expected result
4	User attempts to register an account with registered username	Username:kokoi Email:kokoi@gmail.com Password :12345678	Error message will pop up on top of the login page "Error: Username of password has been taken. Try another" and will redirect to the sign up page again.	Same as expected result

Table 7-1-1-T2: Sign Up Page Test Plan

7.1.1 Unit Test 3: Login Page

No	Event	Value	Expected Result	Actual Result
1	User enter information required to login into the website. Then click on sign in.	Username:kokoi Password :12345678	System will load for a while then Navigate to the dashboard page.	Same as expected result
2	User press the forgot password	--	Navigate to reset password page	Same as expected result
3	User enter invalid username or password	Username:kokoi1 Password :12345678	Error message will pop on top of login page displaying "Error: Usename or password incorrect. Please try again" and redirect to the login system.	Same as expected result
4	User left the username or the password blank	Username: Password :	Error message will pop up at username and password column displaying "Please Fill up this field"	Same as expected result

Table 7-1-1-T3: Login Page Test Plan

7.1.1 Unit Test 4: Reset Password Page

During the reset process, the reset link will be send to the user's email address and not to the admin's email address. Only for the sign up process the new user's activation code will send to admin's email address.

No	Event	Value	Expected Result	Actual Result
1	User enter information required and new password to reset his account's old password.	Username:kokoi Email: kokoi@gmail. Password :123456789	System will load for a while then a message will display saying a link with an activation code have been seen to the user's email. Navigate to the link to complete the reset password process.	Same as expected result
2	User navigate to the link send by the website to complete the reset password process.	--	Navigate to the login page with successful message display at top of the page.	Same as expected result
3	User enter invalid username or password during reset password	Username:kokoi2 Password :12345678	Error message will pop on top of reset page displaying "Error: Username not found in database. Please sign up first" and redirect to the reset password page.	Same as expected result
4	User left the username or the password blank	Username: Email : Password :	Error message will pop up at username and password column displaying "Please Fill up this field"	Same as expected result

Table 7-1-1-T4: Reset Password Page Test Plan

7.1.1 Unit Test 5: Dashboard Page

No	Event	Value	Expected Result	Actual Result
1	Toggle the mode button from manual to auto	State = auto	JavaScript read the state of the toggle button and send to the server side. Since state is auto, server side activate the auto mode.	Same as expected result
	Toggle the mode button from auto to manual	State= manual	JavaScript read the state of the toggle button and send to the server side. Since state is manual, server side will activate the manual mode.	Same as expected result
2	Toggle the White Light button from off to on.	State = on	JavaScript read the state of the toggle button and send to the server side. Since state is on, server side will switch on the LED of the aquarium using relay.	Same as expected result
	Toggle the White Light button from on to off.	State = off	JavaScript read the state of the toggle button and send to the server side. Since state is of, server side will switch off the LED of the aquarium using relay.	Same as expected result
3	Toggle the WaterFillUp button from off to on.	State = on	JavaScript read the state of the toggle button and send to the server side. Since state is on, server side will switch on the water pump to fill the water into aquarium.	Same as expected result
	Toggle the White WaterFillUp from on to off.	State = off	JavaScript read the state of the toggle button and send to the server side. Since state is off, server side will switch off the water pump to stop fill the water into aquarium.	Same as expected result

4	Toggle the WaterUnFill button from off to on.	State = on	JavaScript read the state of the toggle button and send to the server side. Since state is on, server side will switch on the water pump to unfill the water from aquarium.	Same as expected result
	Toggle the White WaterUnFill from on to off.	State = off	JavaScript read the state of the toggle button and send to the server side. Since state is off, server side will switch off the water pump to stop unfill the water from aquarium.	Same as expected result
5	Toggle the CoolerFan button from off to on.	State = on	JavaScript read the state of the toggle button and send to the server side. Since state is on, server side will switch on the cooling fan of aquarium.	Same as expected result
	Toggle the White CoolerFan from on to off.	State = off	JavaScript read the state of the toggle button and send to the server side. Since state is off, server side will switch off the cooling fan of aquarium.	Same as expected result
6	Click the Feed Fish button	State = on	JavaScript read the state of the button and send to the server side. Since state is on, server side will execute the function to feed the fish.	Same as expected result but sometime the feeder does not work.
7	Drag the slider of RGB Light to certain position.	State= value of slider.	JavaScript read the state of the button and send to the server side. The server side will read the value of the state and control the brightness of RGB LED based on the value	Same as expected but if user change the position of RGB LED slider fast, the whole website will hang.

8	Pour some warm water into the aquarium	--	JavaScript read the temperature value from the server side every 2second and update the chart of the Temperature sensor. Hence the temperature chart value in the dashboard will increase	Same as expected result
9	Pump out some water from aquarium	--	JavaScript read the temperature value from the server side every 2second and update the chart of the Water level sensor. Hence the water level value in the chart should reduce	Same as expected result
10	Pour some dirty water into the aquarium	--	JavaScript read the temperature value from the server side every 2second and update the chart of the Turbidity sensor. Hence the turbidity needle at the chart should point to dirty section.	Same as expected result

Table 7-1-1-T5: Dashboard Page Test Plan

7.1.1 Unit Test 6: Sensors RealTime Graph Page

No	Event	Value	Expected Result	Actual Result
1	Pour some warm water into the aquarium	--	JavaScript read the temperature value from the server side every 2second and update the real-time graph of the Temperature sensor. Hence the temperature value in real-time graph will increase	Same as expected result
2	Pump out some water from aquarium	--	JavaScript read the temperature value from the server side every 2second and update the real-time graph of the Water level sensor. Hence the water level value in the real time should reduce	Same as expected result
3	Pour some dirty water into the aquarium	--	JavaScript read the temperature value from the server side every 2second and update the real-time graph of the Turbidity sensor. Hence the turbidity value in real-time graph will reduce..	Same as expected result

Table 7-1-1-T6: Sensors RealTime Graph Page Test Plan

7.1.1 Unit Test 7: Database Query Page

No	Event	Value	Expected Result	Actual Result
1	Enter the correct format of start datetime and end datetime into the input of start and end.	Start datetime: 2017-05-16 02:41:30 End datetime : 2017-05-16 16:04:28	The page will be refresh and the table will be filled with sensors value in between the start datetime and end datetime from database	Same as expected result
2	Enter the wrong format of start datetime and end datetime into the input of start and end.	Start datetime: 2017/05/16 02:41:30 End datetime : 2017/05/16 16:04:28	The page will be refresh but with empty table	Same as expected result
3	Left the start datetime and end datetime input empty	Start datetime: End datetime :	The page will be refresh with all the sensors value from database will be display in the table.	Same as expected result

Table 7-1-1-T7: Database Query Page Test Plan

7.1.1 Unit Test 8: Auto Mode Setting Page

No	Event	Value	Expected Result	Actual Result
1	Enter the correct format of id and Feed Time	Id = 5 (1-5) Feed Time = 18:00:00	The page will be refresh and the Feed Time column in the table at the particular id will update to 18:00:00	Same as expected result
2	Enter the wrong format of id and Feed Time	Id = 10 (1-5) Feed Time = 18/00/00	The page will be refresh but with the Feed Time column wont update	Same as expected result
3	Left the id and Feed Time input empty	Id = Feed Time =	The page will be refresh with no changes on the table.	Same as expected result
4	Toggle the Light_Control button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the light control function in the auto mode.	Same as expected result
	Toggle the Light_Control button from enable to disable	State= disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the light control function in the auto mode	Same as expected result
5	Toggle the Feeder_Control button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the fish feeder function in the auto mode.	Same as expected result
	Toggle the Feeder_Control button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will	Same as expected result

			disable the fish feeder function in the auto mode	
6	Toggle the WaterLeakAlarm button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the water leak alarm function in the auto mode.	Same as expected result
	Toggle the WaterLeakAlarm button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the water leak alarm function in the auto mode	Same as expected result
7	Toggle the Turbidity_Control button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the Turbidity control function in the auto mode.	Same as expected result
	Toggle the Turbidity_Control button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the Turbidity control function in the auto mode	Same as expected result
8	Toggle the TemperatureControl button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the Temperature control function in the auto mode.	Same as expected result
	Toggle the TemperatureContr	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will	Same as expected result

	ol button from enable to disable		disable the Temperature control function in the auto mode	
9	Toggle the WaterLevelControl button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the Water Level control function in the auto mode.	Same as expected result
	Toggle the WaterLevelControl button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the Water Level control function in the auto mode	Same as expected result

Table 7-1-1-T8: Auto Mode Setting Page

<u>7.1.1 Unit Test 9: Live Stream video Page</u>

No	**Event**	**Value**	**Expected Result**	**Actual Result**
1	Navigate to the page	--	The page will be load for a while than the live stream video should be running on the page. Able to see the aquarium in real-time.	Same as expected result

Table 7-1-1-T9: Live Stream video Page

7.1.1 Unit Test 10: Telegram Chat Bot Page

No	Event	Value	Expected Result	Actual Result
1	Toggle the Telegram's enable/disable button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the Telegram chat bot control function	Same as expected result
	Toggle the Telegram's enable/disable button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the Telegram chat bot control function	Same as expected result

Table 7-1-1-T10: Telegram Chat Bot Page

7.1.1 Unit Test 11: Voice Command Control Page

No	Event	Value	Expected Result	Actual Result
1	Toggle the Voice Command Control enable/disable button from disable to enable	State = enable	JavaScript read the state of the toggle button and send to the server side. Since state is enable, server side will enable the Voice command control function.	Same as expected result
	Toggle the Voice Command Control enable/disable button from enable to disable	State = disable	JavaScript read the state of the toggle button and send to the server side. Since state is disable, server side will disable the Voice command control function.	Same as expected result

Table 7-1-1-T11: Voice Command Control Page

7.2 Telegram Chat Bot Testing

7.2.1 Result and discussion

In order to test the Telegram chat bot is working properly, test plan with several test case have been defined and execute. Below shows test case and the results of the test.

No	Event	Value	Expected Result	Actual Result
1	Send /start message to Telegram chat bot in Pi	Message = /start	The Pi bot received the message and reply the message with "Welcome to RaashVAqua Aquarium Monitoring System. How I can help you?"	Same as expected result
2	Send who are you? message to Telegram chat bot in Pi	Message = who are you?	The Pi bot received the message and reply the message with "I am a bot, programmed in the Raspberry Pi"	Same as expected result
3	Send how old are you? message to Telegram chat bot in Pi	Message = how old are you?	The Pi bot received the message and reply the message with "My age is %d days and %d seconds"	Same as expected result
4	Send who is your father? message to Telegram chat bot in Pi	Message = who is your father?	The Pi bot received the message and reply the message with "My father and mother is Thiyraash s/o David, a Computer Engineering student from Universiti Tunku Abdul Rahman"	Same as expected result
5	Send what is your function? message to Telegram chat bot in Pi	Message = what is your function?	The Pi bot received the message and reply the message with "My function is to monitor the Aquarium such as its temperature, turbidity,water level and inform them to the client	Same as expected result

6	Send auto message to Telegram chat bot in Pi	Message = auto	• The Pi bot received the message and change the aquarium mode to auto. • Reply the message with "Auto mode activated"	Same as expected result
7	Send manual message to Telegram chat bot in Pi	Message = manual	• The Pi bot received the message and change the aquarium mode to manual. • Reply the message with "Manual mode activated"	Same as expected result
8	Send RGB control message to Telegram chat bot in Pi	Message = red/green/blue/yellow/purple	• The Pi bot received the message and change the aquarium RGB color based on the message received.	Same as expected result
9	Send actuator off message to Telegram chat bot in Pi	Message = aoff	• The Pi bot received the message and switch off all the actuator of the aquarium • Reply the message with "Actuator is off"	Same as expected result
10	Send light on message to Telegram chat bot in Pi	Message =Lighton	• The Pi bot received the message and switch on the WHITE led of aquarium • Reply the message with "Light is on"	Same as expected result
11	Send light on message to Telegram chat bot in Pi	Message =Lightoff	• The Pi bot received the message and switch off the WHITE led of aquarium	Same as expected result

			• Reply the message with "Light is off"	
12	Send fan on message to Telegram chat bot in Pi	Message =fanon	• The Pi bot received the message and switch on the cooling fan of aquarium • Reply the message with "Cooling fan is on"	Same as expected result
13	Send fan off message to Telegram chat bot in Pi	Message =fanoff	• The Pi bot received the message and switch off the cooling fan of aquarium • Reply the message with "Cooling fan is off"	Same as expected result
14	Send pump 1 state message to Telegram chat bot in Pi	Message =pump1on/pump2 off	• The Pi bot received the message and switch on/off the pump 1 of aquarium • Reply the message with "Pump 1 is on/off"	Same as expected result
15	Send pump 2 state message to Telegram chat bot in Pi	Message =pump2on/pump2 off	• The Pi bot received the message and switch on/off the pump 2 of aquarium • Reply the message with "Pump 2 is on/off"	Same as expected result
16	Send feed message to Telegram chat bot in Pi	Message =Feed	• The Pi bot received the message and feed the fishes in aquarium • Reply the message with "Feeding"	Same as expected result
17	Send message regarding turbidity to	Message = tur/turbidity	• The Pi bot received the message and obtained	Same as expected result

	Telegram chat bot in Pi		the current turbidity sensor value • Then send the turbidity value back to user Reply the message with "Sensor is out of water" or "Water is dirty " or "Water is clean" or "Opaque object detected between the sensor"	
18	Send message regarding temperature to Telegram chat bot in Pi	Message = temp/temperature	• The Pi bot received the message and obtained the current temperature sensor value • Then send the temperature value back to user by reply the message with "The temperature of water is %s"	Same as expected result
19	Send message regarding water level of sensor values to Telegram chat bot in Pi	Message = level	• The Pi bot received the message and obtained the current water level value • Then send the water level value back to user by reply the message with "The water level of water is %s"	Same as expected result
20	Send message regarding summary of sensor values to Telegram chat bot in Pi	Message = summary	• The Pi bot received the message and obtained the current sensor values • Then send the those value back to user by reply the message with	Same as expected result

			"The temperature of water is %s" and "The water level of water is %s" and "The turbidity of water is "	

Table 7-2-1-T1: Telegram Chat bot Test Plan

7.3 Voice Command Control Testing

7.3.1 Result and discussion

In order to test the whether Voice command control is working properly, test plan with several test case have been defined and execute. Below shows test case and the results of the test.

No	Event	Value	Expected Result	Actual Result
1	Speak the hotword call 'Raash'	hotword = Raash	The system will catch the hotword and trigger a BEEP sound. This indicate it is listening.	Same as expected result
2	Speak the hotword call 'Please turn on the light'	hotword = Please turn on the light	The system will catch the hotword and execute the light on function. The aquarium's White LED will be switch on.	Same as expected result
3	Speak the hotword call 'Turn the light off'	hotword = Turn the light off	The system will catch the hotword and execute the light off function. The aquarium's White LED will be switch on.	Same as expected result
4	Speak the hotword call 'Feed the fish ,dude'	hotword = Feed the fish, dude	The system will catch the hotword and execute feeding function. The aquarium's fish feeder start to feed the fish	Same as expected result
5	Speak the hotword call 'RGB alert'	hotword = RGB alert	The system will catch the hotword and execute RGB alert fucntion. The aquarium's RED LED will start to blink	Same as expected result

6	Speak the hotword call ‘OFFRgb’	hotword= OFFRGB	The system will catch the hotword and execute Off RGB fucntion. The aquarium’s RGB light will be switch off.	Same as expected result
7	Speak the hotword call ‘Apply manual mode’	hotword= Apply manual mode	The system will catch the hotword and change the mode of the system to manual.	Same as expected result
8	Speak the hotword call ‘Auto mode’	hotword= Auto mode	The system will catch the hotword and change the mode of the system to auto.	Same as expected result
9	Speak the hotword call ‘Empty the water’	hotword= Empty the water	The system will catch the hotword and execute empty tank function. The aquarium’s water will be suck out using water pump.	Same as expected result
	Speak the hotword call ‘Change the water’	hotword= Change the water	The system will catch the hotword and execute Change water function. The aquarium’s water will be replace with new water.	Same as expected result

Table 7-3-1-T1: Voice Command Control Test Plan

Chapter 8: Conclusion and Discussion

In this section, the objectives and contribution of this aquarium monitoring system had achieved will be discussed. Besides that, the major problems encountered during the developments of this system will be shared in the session. Lastly the limitations and future enhancement of the developed system will be discussed as well.

8.1 Project Review

In conclusion, an IOT based aquarium monitoring system will be developed in this project. The main goal of this system is to help aquarium owner to make the aquarium maintaining and monitoring more convenient and easier. The objectives of this Aquarium Monitoring system or in short I called it RaashVAqua system is to develop reliable, real-time based aquarium monitoring system where is able to update the users about current situation of their aquarium. Besides that, the system also able to make informed decisions and perform certain tack by its own without the user interferences. The most importantly the aquarium monitoring system should be user-friendly and even can be operated by the non-technical person. All the objectives mentioned above are achieved by this developed aquarium monitoring system.

I have encountered few problem during the development of this project especially on developing the website for this system. The first problem is choosing a server side web framework for the system. Since this is the first time I'm involved in website design and development, I have very little knowledge on server side web framework and the client side programming. So even choosing the suitable web framework for my system itself took a lot of the time. But somehow I managed to fix with web framework called Flask. Another problem I encountered is sharing of data in between different mode of control in aquarium monitoring system. There are three control mode of this system website, Telegram chat bot and Voice control applivation and three of them running in separate files. I need to share some dynamic value in between those files but it's very difficult to achieve because all of them in separate files. But after some research I able to solve those problem just by using MySQL database as medium to share data in between them.

8.2 Limitation

Every system have their pros and cons so as this developed aquarium monitoring system. The limitation of this system is lack of monitoring sensors. Sensors are used in this system to monitor the water condition and the fish environment so that they are in optimum level where the fish can live safely. In this system there are only, four sensors used, water level sensor, water leak sensor, water temperature sensor and turbidity sensor. This sensors are working perfectly under normal conditions but if there are some worse case scenarios where there are some other problem in the fish tank such as imbalance pH or salinity condition in the fish tank, this current developed aquarium monitoring system cannot detect the changes because it does not have the appropriate sensor for it. Imbalance pH is one of the common problem that the aquarium user are facing nowadays. Besides another limitation is with sensor that detect the water cloudiness that is turbidity sensor. Turbidity sensor working perfectly to detect whether water is clean or not but is not very sensitive to the cloudiness of water. One of the reason is because turbidity sensor is not waterproof sensor. The water need to be very dirty in order the sensor assume that the water is dirty.

8.2 Future Enhancement

Currently this system have less effective sensors and actuators. So in future many more advance sensor such as pH, salinity, oxygen level sensor and actuators should be added to the system in order to make the system more advance in monitoring the aquarium. Furthermore, in future the aquarium monitoring system should equipped with some artificial intelligent such as detecting odd pattern of sensors value for certain time or adding computer vision to detect and understand the behavior of one type of fish in the aquarium. Adding a mobile application to the system to enable the user to control them is also one of the future enhancement of this system.

References

1. Chiu, M.C., 2010. A Multi-functional Aquarium Equipped with Automatic Thermal Control/Fodder-Feeding/water Treatment using a Network Remote Control System. *Information Technology Journal*, *9*(7), pp.1458-1466.

2. Noor, M.Z.H., Hussian, A.K., Saaid, M.F., Ali, M.S.A.M. and Zolkapli, M., 2012, July. The design and development of automatic fish feeder system using PIC microcontroller. In *Control and System Graduate Research Colloquium (ICSGRC), 2012 IEEE* (pp. 343-347). IEEE.

3. Vijayakumar, N. and Ramya, R., 2015, March. The real time monitoring of water quality in IoT environment. In *Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on*(pp. 1-5). IEEE

4. Prasad, A.N., Mamun, K.A., Islam, F.R. and Haqva, H., 2015, December. Smart water quality monitoring system. In *2015 2nd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)* (pp. 1-6). IEEE.

5. Perumal, T., Sulaiman, M.N. and Leong, C.Y., 2015, October. Internet of Things (IoT) enabled water monitoring system. In *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)* (pp. 86-87). IEEE.

6. Nurliani Hidayah Ritonga; Agung Nugroho Jati; Rifki Wijaya, 2016, May. Automatic Arowana Raiser Controller Using Mobile Application Based on Android. In *2016 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)* (pp.86-87). IEEE.

7. Taotao Xu , Feng Chen , 2014 August. An Embedded Fuzzy Decision System for Aquaculture. In *2014 2014 IEEE Workshop on Electronics, Computer and Applications. (pp.351-353).* IEEE Conference Publications

8. Merriam – Webster (2016) Definition of Aquarium . Available from: http://www.merriam-webster.com/dictionary/aquarium (Accessed : 21 August 2016)

9. SDLC Prototype Model : Design, advantages, disadvantages and application
Available at: http://er.yuvayana.org/sdlc-prototype-model-design-advantages-disadvantages-and-applications/ (Accessed: 10 August 2016)

10. systems development life cycle (SDLC) Available at : http://searchsoftwarequality.techtarget.com/definition/systems-development-life-cycle (Accessed: 10 August 2016)

11. SmartWater (2015) Available at : http://www.libelium.com/products/waspmote/ (Accessed : 17 July 2016)

12. Divine Vastu Tips for Fish Aquarium:(A Good Remedial Measure for Vastu Defect) Available at: http://theindianeye.net/west-coast-news/divine-vastu-tips-for-fish-aquariuma-good-remedial-measure-for-vastu-defect (Accessed : 21 August 2016)

13. Feng Shui Lucky Number of Fish in a Tank (2016) Available at: http://feng-shui.lovetoknow.com/Feng_Shui_Lucky_Number_of_Fish_in_a_Tank (Accessed : 21 August 2016)

14. Analog Turbidity Sensor Available from : http://www.myduino.com/index.php?route=product/product&product_id=869&search=turbidity+sensor (Accessed : 18 August 2016)

15. Waterproof DS18B20 Kit Available from : http://www.myduino.com/index.php?route=product/product&product_id=496&search=DS18B20 (Accessed : 18 August 2016)

16. DC12V 4.8W Mini Brushless Submersible Water Pump for Fish Tank Available at : http://www.lelong.com.my/dc12v-4-8w-mini-brushless-submersible-water-pump-fish-tank-thestarbuy-174271482-2017-02-Sale-P.htm (Accessed : 18 August 2016)

17. Internet of Things Available at : http://www.gartner.com/it-glossary/internet-of-things/ (Accessed : 18 August 2016)

18. David Ordnung , *Raspberry Pi &RGB LED-Strips* 2015. [online] Available at : http://dordnung.de/raspberrypi-ledstrip/ (Accessed : 31 January 2017)

19. Instructable.com (2015). *Real-time Graphing With the Raspberry Pi.* [online] Available at : http://www.instructables.com/id/Streaming-Data-Visualization-Plotly-Raspberry-Pi/ (Accessed : 1 February 2017)

20. Raspberry Pi Forum (2013). *PWM for servos, motors, and LEDs plus plus* . [online] Available at : *https://www.raspberrypi.org/forums/viewtopic.php?t=43579&p=368070* (Accessed: 1 February 2017)

21. Hackester.com (2017). *Telegram Bot with Raspberry Pi* [online] Available at : https://www.hackster.io/Salman_faris_vp/telegram-bot-with-raspberry-pi-f373da (Accessed : 5 February 2017)

22. Christian Cawley (2014). *Host Your Own Website On Your Raspberry Pi [online]* Available at : http://www.makeuseof.com/tag/host-website-raspberry-pi/ (Accessed : 5 February 2017)

23. Adafruit.com (2013). *MCP3008* [online] Available at : https://learn.adafruit.com/raspberry-pi-analog-to-digital-converters/mcp3008 (Accessed : 7 February 2017)

24. Serving Raspberry Pi with Flask [online] Available at : https://mattrichardson.com/Raspberry-Pi-Flask