Water monitoring system

Water Level monitoring system

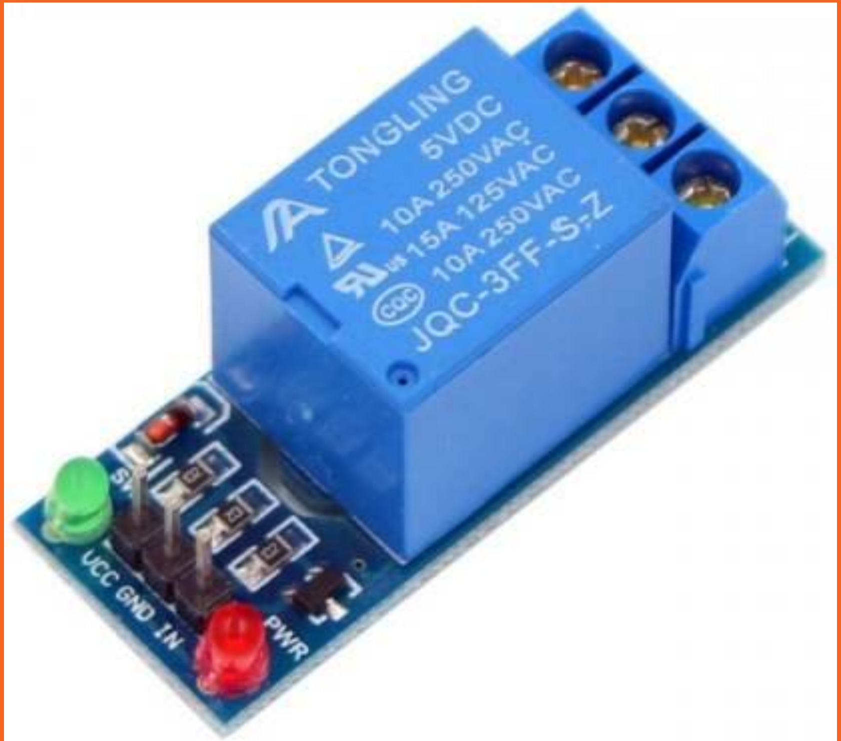
- This system would help us to monitor the level of water in tank or any other storage device.
- The system would tell upto which level the tank has been filled. It would also have an alarm that would alert the user when the tank is fully filled.
- We can control the switch of motor using app.
- We can check temperature ratings around motor

Hardware components.

- Relay Board
- Ultrasonic Sensor
- Buzzer
- Submersible water pump (3-6V)
- Variable resistor
- Voltage regulator
- ESP32

RELAY BOARD

- Relay boards are **computer boards with an array of relays and switches**.
- They have input and output terminals and are designed to control the voltage supply.
- Relay boards provide independently programmable, real-time control for each of several onboard relay channels.

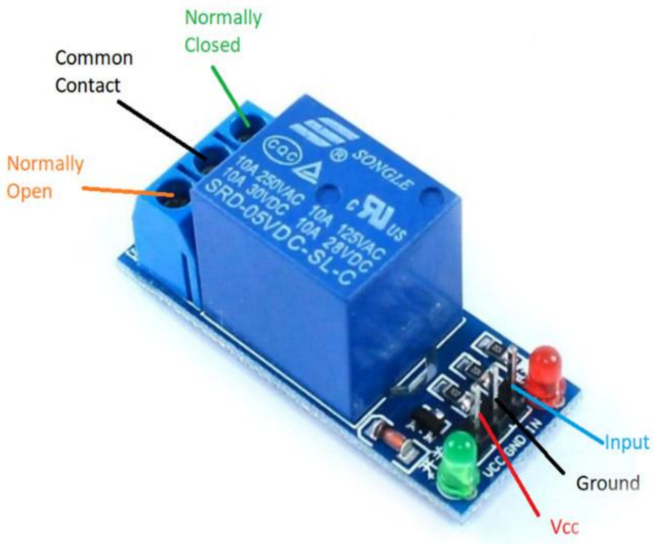


Relay board pins

The relay pins can be categorized two groups: low voltage and high voltage.

1. **The low voltage pins:** are interfaced to ESP32, including three pins:
    a. **GND pin:** connect this pin to **GND** (0V)
    b. **VCC pin:** connect this pin to **VCC** (5V)
    c. **IN pin:** receives the control signal from ESP32

2. **The high voltage pins:** are interfaced to high-voltage device, including three pins (usually in screw terminal):
    a. **NO pin:** is normally open pin. It is used in the normally open mode
    b. **NC pin:** is normally closed pin. It is used in the normally closed mode
    c. **COM pin:** is the common pin. It is used in both normally open mode and normally closed mode



ESP32

is a series of low-cost, low-power system on a chip microcontrollers with integrated Wi-Fi and dual-mode Bluetooth.

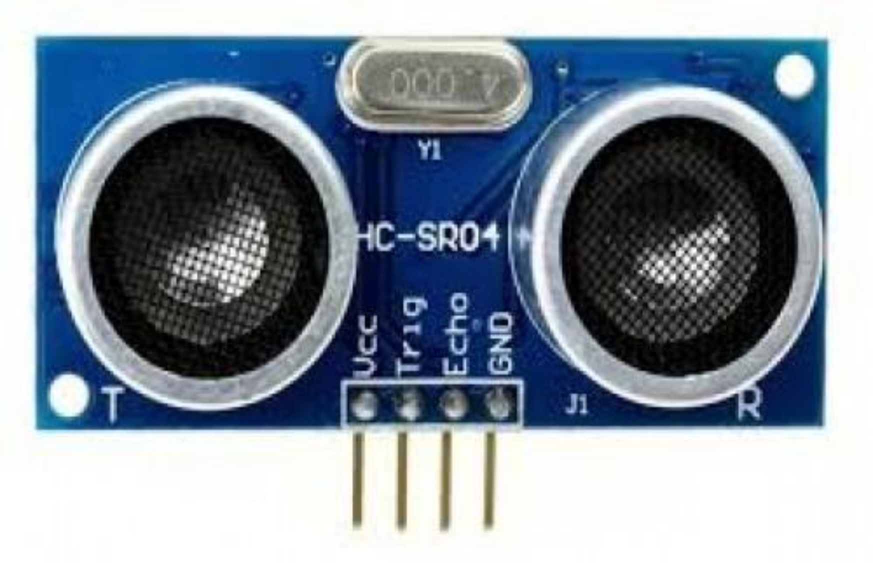
- The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules.

ESP32

- ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process.It is a successor to the ESP8266 microcontroller.

HC-SR04 ULTRASONIC SENSOR

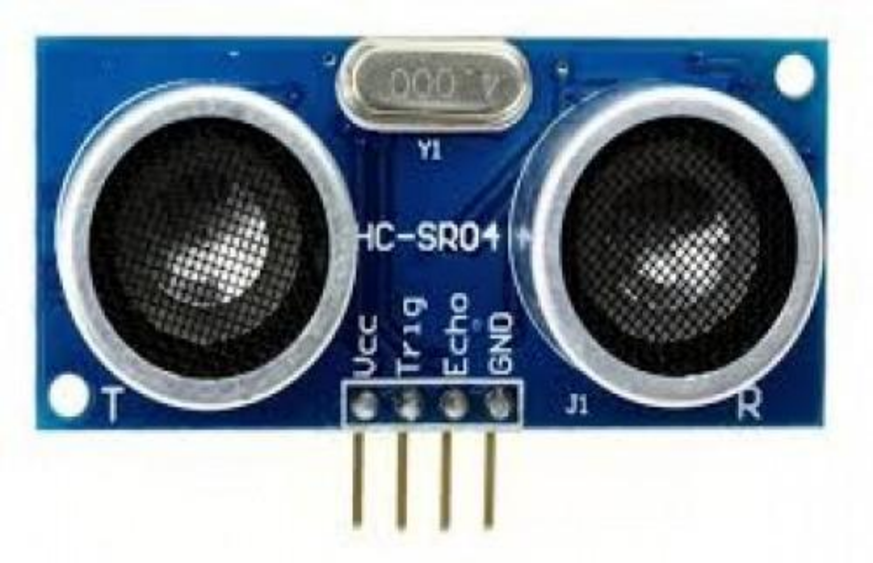
- An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal.
- Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear).
- Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).



HC-SR04 ULTRASONIC SENSOR

- In order to calculate the distance between the sensor and the object, the sensor measures the time it takes between the emission of the sound by the transmitter to its contact with the receiver.

- The formula for this calculation is **D = ½ T x C** (where D is the distance, T is the time, and C is the speed of sound ~ 343 meters/second).



PIEZO BUZZER

A buzzer or beeper is an audio signaling device,[1] which may be mechanical, electromechanical, or piezoelectric (piezo for short).

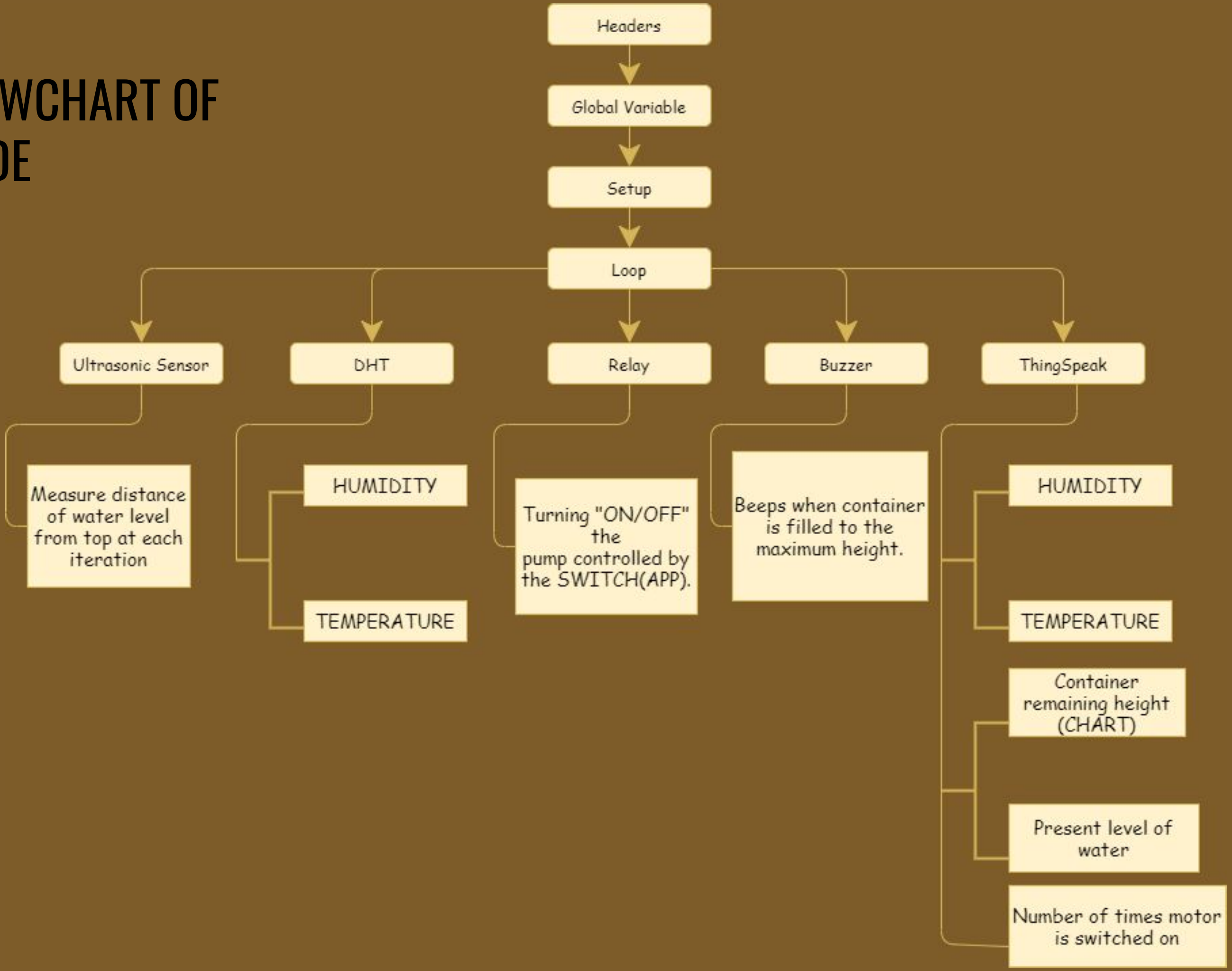
→ Piezo Buzzer is used to generate sound, beep or even melody of a song.

1. Piezo Buzzer usually has two pins:

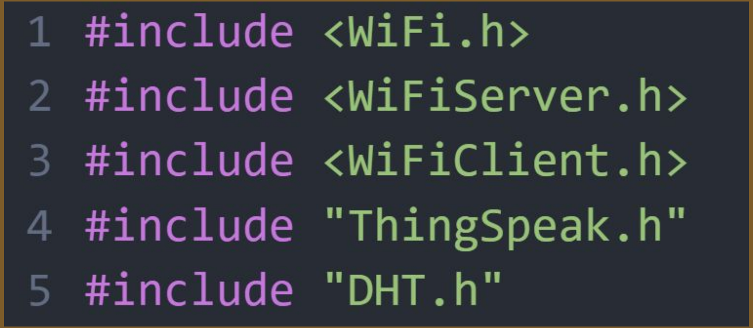
   a. Negative (-) pin: connect this pin to **GND** (0V)

   b. Positive (+) pin: receive the control signal from ESP32

FLOWCHART OF CODE

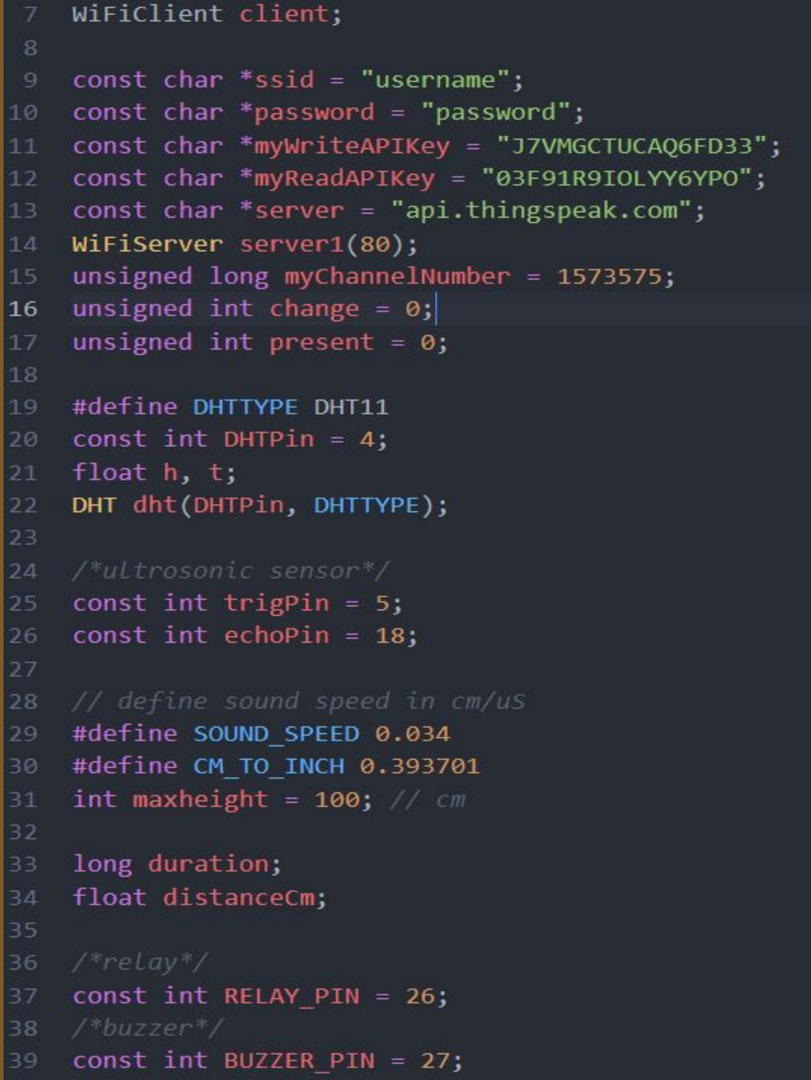

Headers
Global Variable
Setup
Loop
Ultrasonic Sensor
DHT
Relay
Buzzer
ThingSpeak
Measure distance of water level from top at each iteration
HUMIDITY
TEMPERATURE
Turning "ON/OFF" the pump controlled by the SWITCH(APP).
Beeps when container is filled to the maximum height.
HUMIDITY
TEMPERATURE
Container remaining height (CHART)
Present level of water
Number of times motor is switched on

HEADER



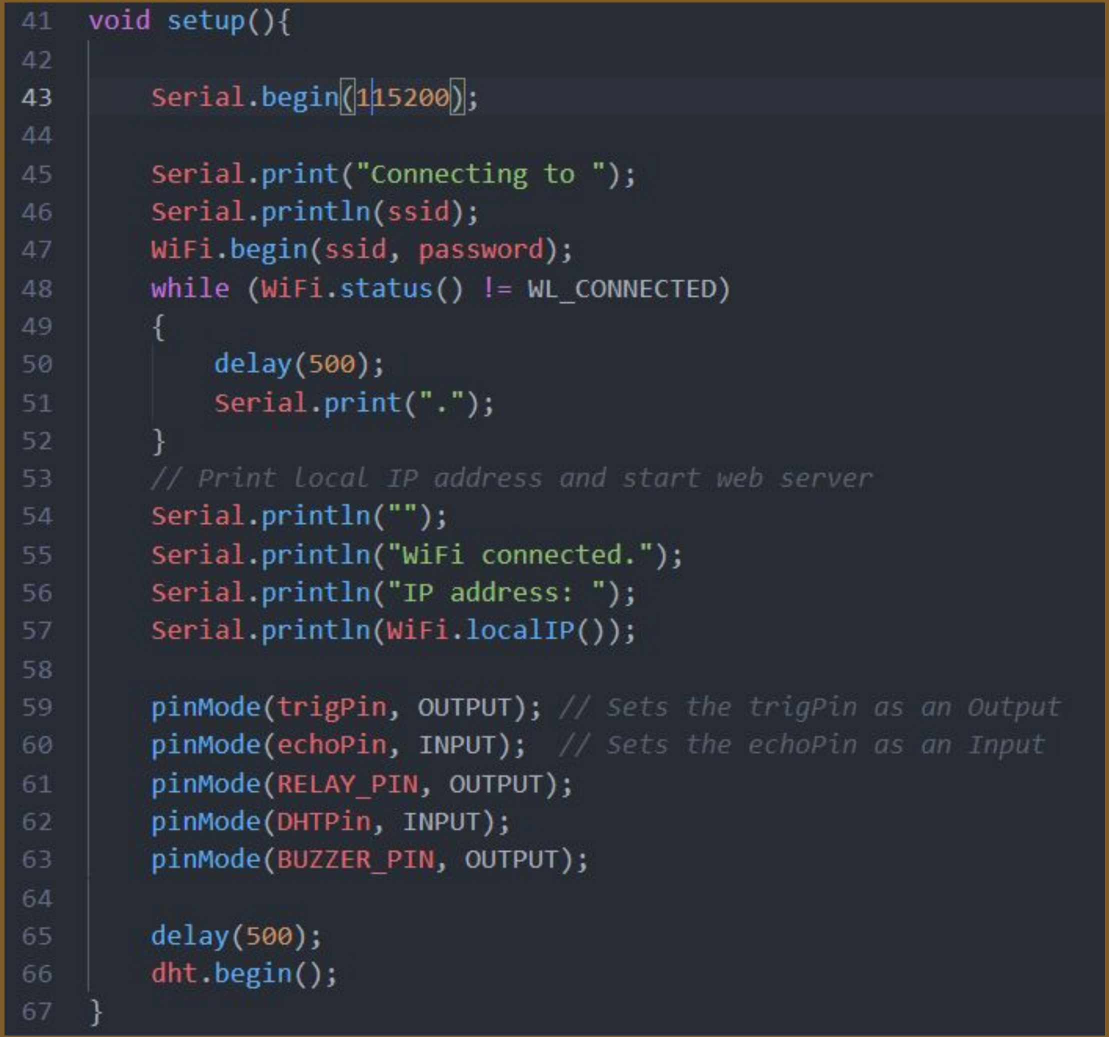
```
#include <WiFi.h>
#include <WiFiServer.h>
#include <WiFiClient.h>
#include "ThingSpeak.h"
#include "DHT.h"
```

GLOBAL DECLARATION



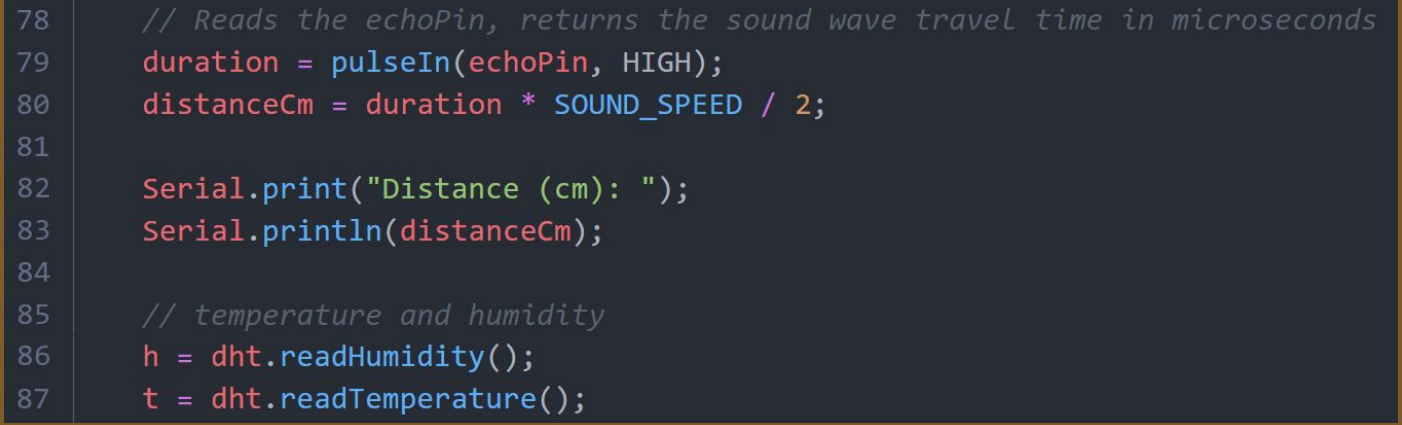
```
WiFiClient client;

const char *ssid = "username";
const char *password = "password";
const char *myWriteAPIKey = "J7VMGCTUCAQ6FD33";
const char *myReadAPIKey = "03F91R9IOLYY6YPO";
const char *server = "api.thingspeak.com";
WiFiServer server1(80);
unsigned long myChannelNumber = 1573575;
unsigned int change = 0;
unsigned int present = 0;

#define DHTTYPE DHT11
const int DHTPin = 4;
float h, t;
DHT dht(DHTPin, DHTTYPE);

/*ultrosonic sensor*/
const int trigPin = 5;
const int echoPin = 18;

// define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
int maxheight = 100; // cm

long duration;
float distanceCm;

/*relay*/
const int RELAY_PIN = 26;
/*buzzer*/
const int BUZZER_PIN = 27;
```

SETUP



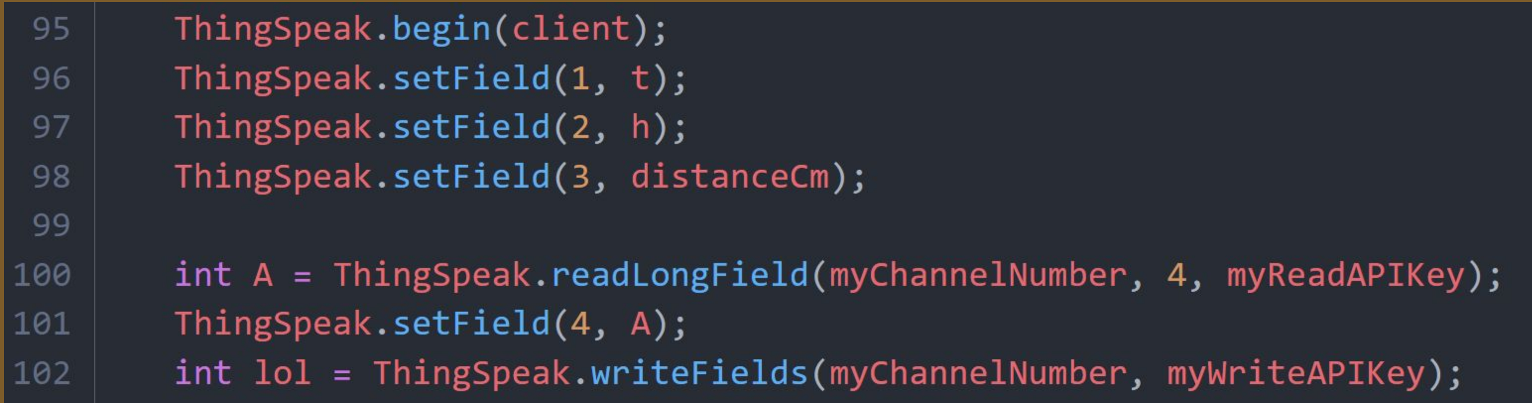
```
void setup(){

    Serial.begin(115200);

    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }
    // Print local IP address and start web server
    Serial.println("");
    Serial.println("WiFi connected.");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());

    pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
    pinMode(echoPin, INPUT);  // Sets the echoPin as an Input
    pinMode(RELAY_PIN, OUTPUT);
    pinMode(DHTPin, INPUT);
    pinMode(BUZZER_PIN, OUTPUT);

    delay(500);
    dht.begin();
}
```

ULTRASONIC CODE



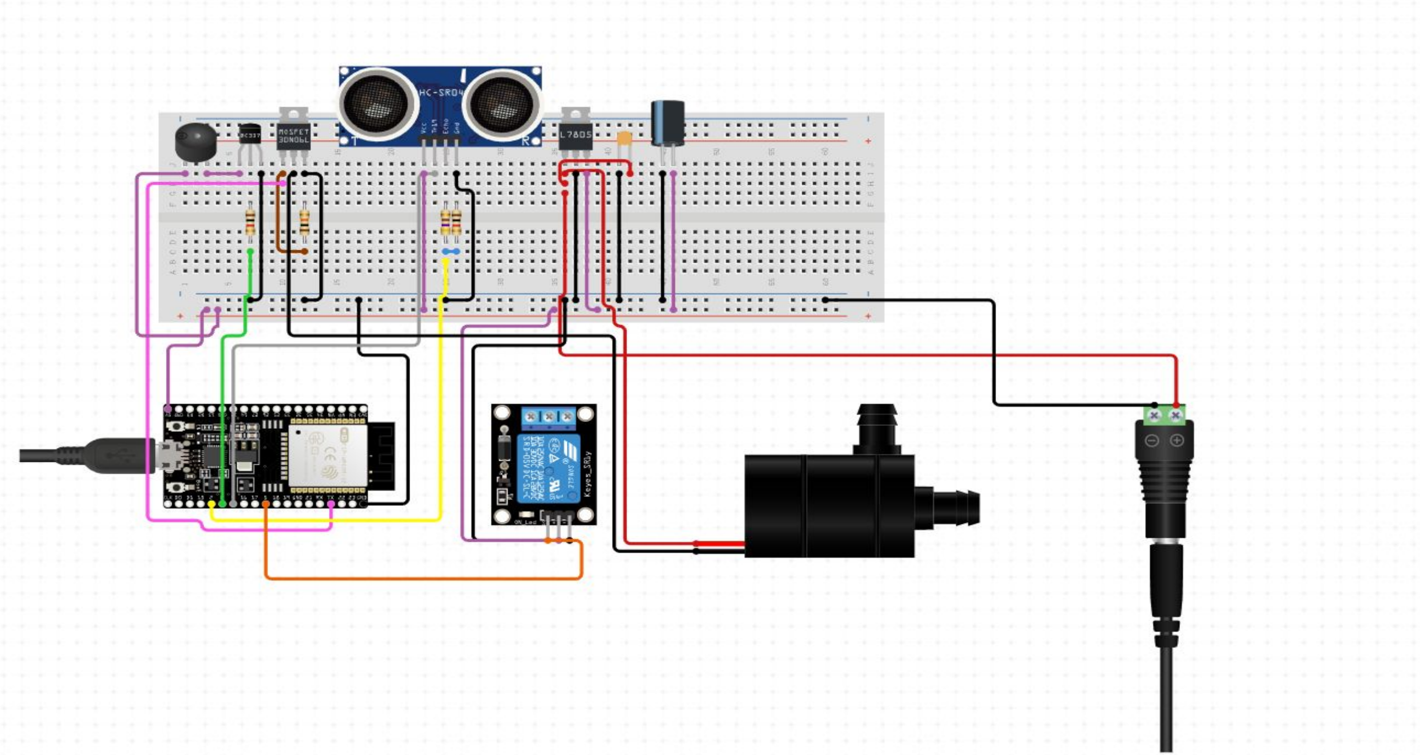
```
    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);
    distanceCm = duration * SOUND_SPEED / 2;

    Serial.print("Distance (cm): ");
    Serial.println(distanceCm);

    // temperature and humidity
    h = dht.readHumidity();
    t = dht.readTemperature();
```

THINGSPEAK

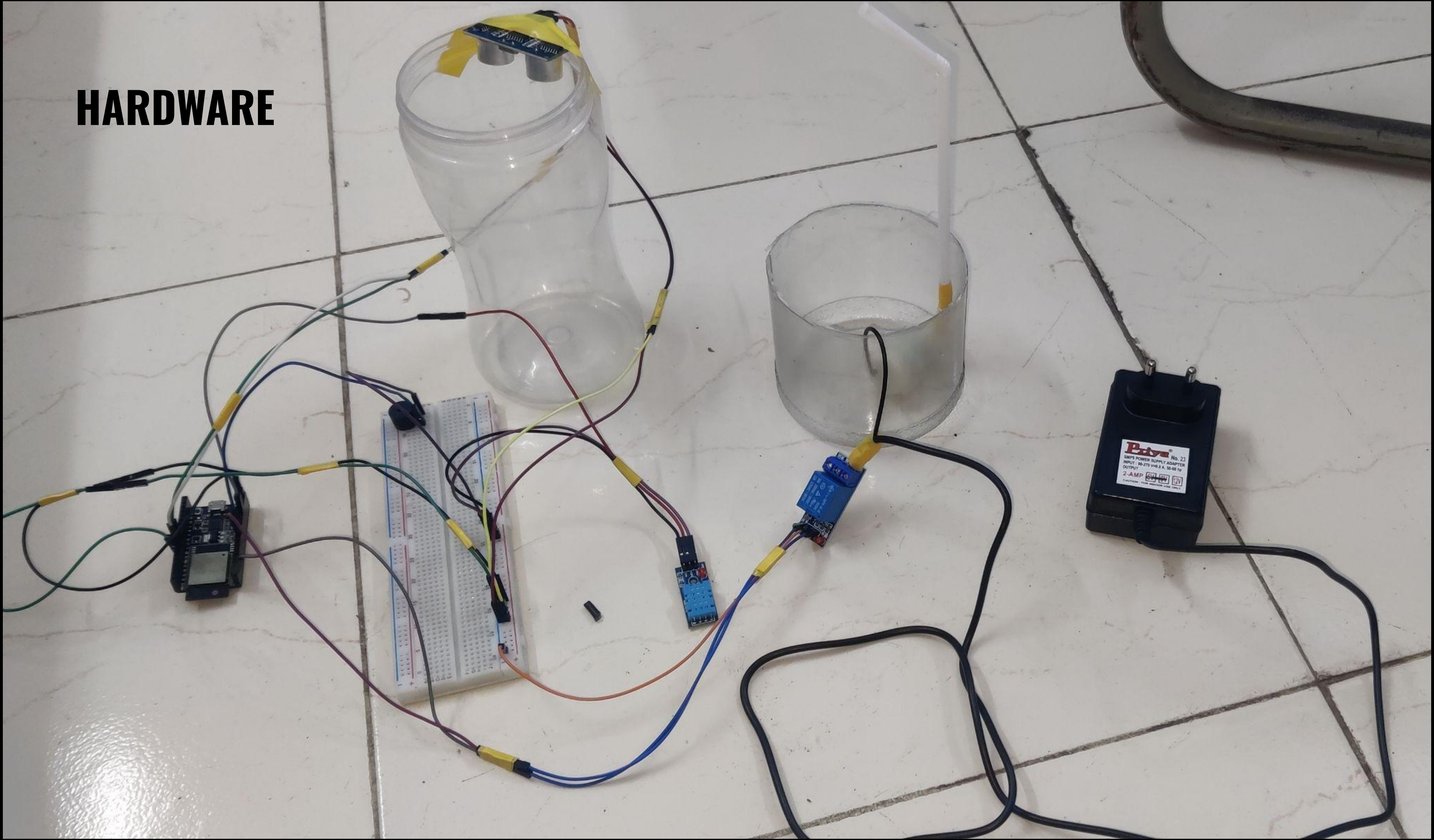


```
    ThingSpeak.begin(client);
    ThingSpeak.setField(1, t);
    ThingSpeak.setField(2, h);
    ThingSpeak.setField(3, distanceCm);

    int A = ThingSpeak.readLongField(myChannelNumber, 4, myReadAPIKey);
    ThingSpeak.setField(4, A);
    int lol = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
```

Water Level monitoring system circuit diagram

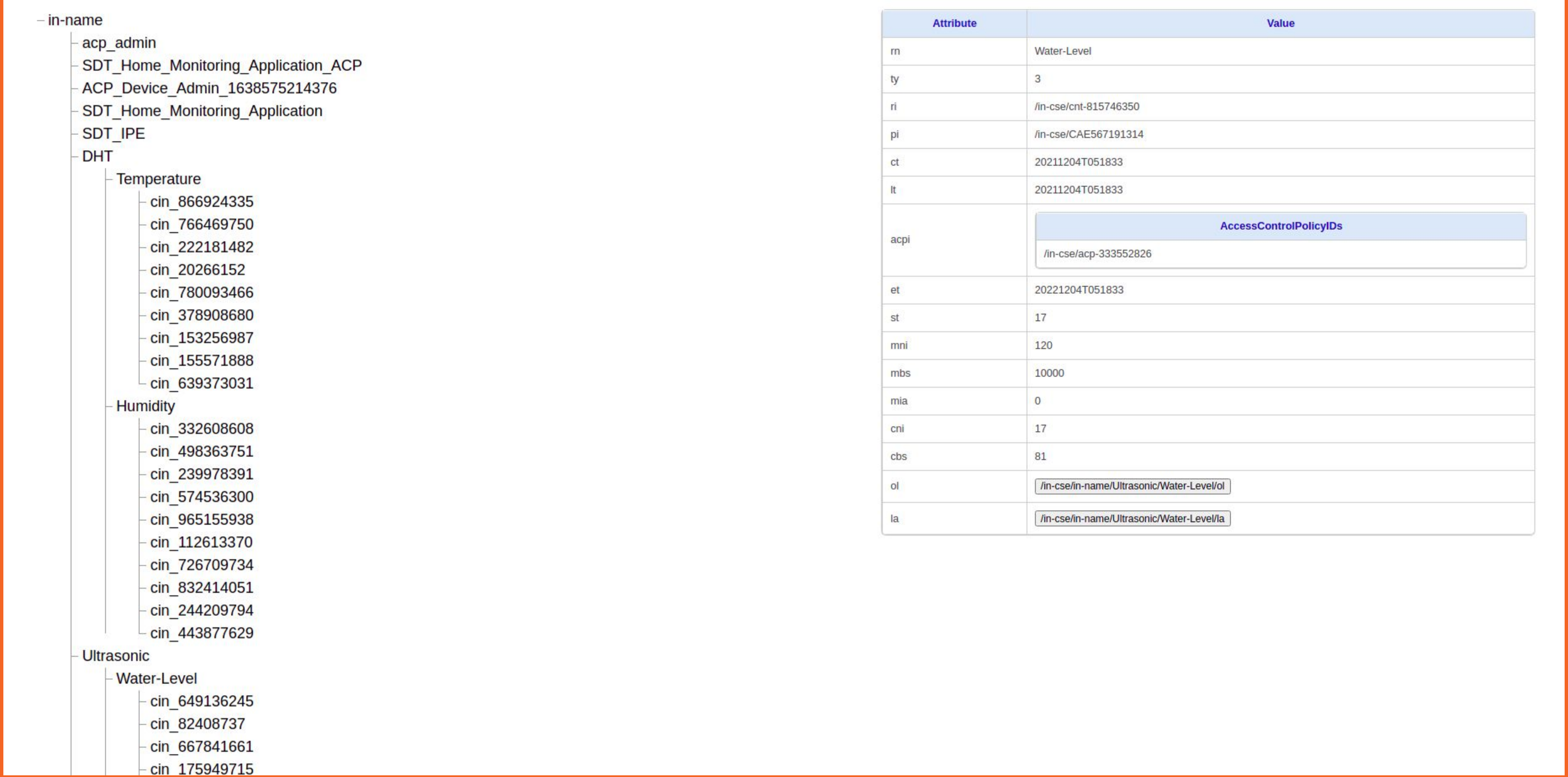


HARDWARE


Priya
No. 23
SMPS POWER SUPPLY ADAPTER
INPUT : 90-270 V~0.5 A, 50-60 hz
OUTPUT
2-AMP 5V 9V 12V
CAUTION : FOR INDOOR USE ONLY

OM2M
OM2M CSE Resource Tree

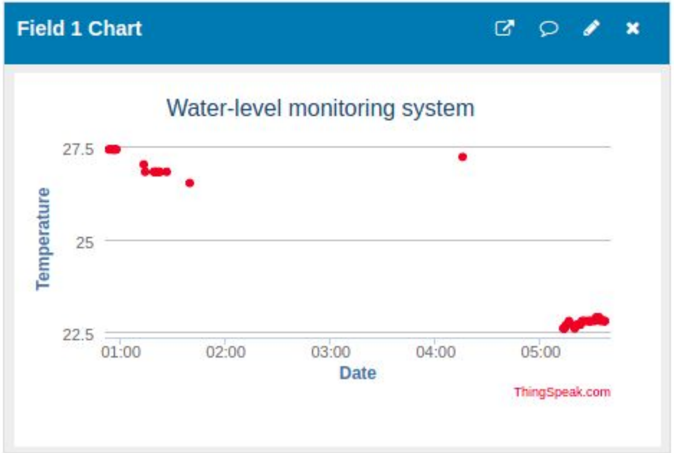
OM2M

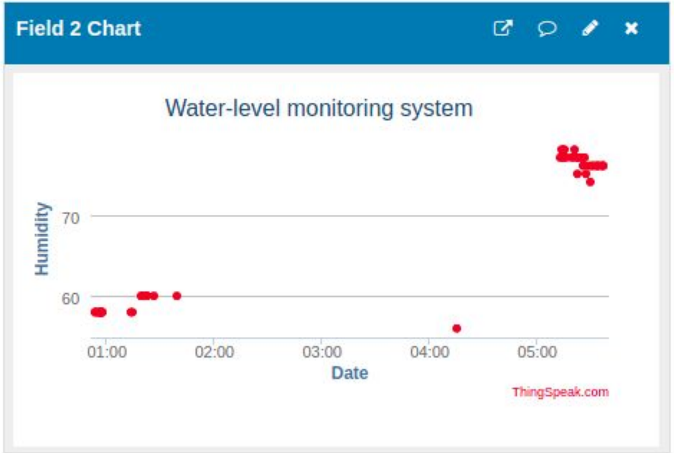


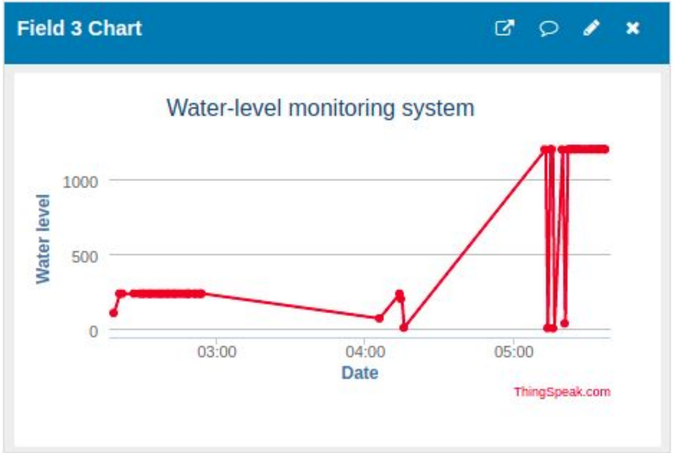
- in-name
  - acp_admin
  - SDT_Home_Monitoring_Application_ACP
  - ACP_Device_Admin_1638575214376
  - SDT_Home_Monitoring_Application
  - SDT_IPE
  - DHT
    - Temperature
      - cin_866924335
      - cin_766469750
      - cin_222181482
      - cin_20266152
      - cin_780093466
      - cin_378908680
      - cin_153256987
      - cin_155571888
      - cin_639373031
    - Humidity
      - cin_332608608
      - cin_498363751
      - cin_239978391
      - cin_574536300
      - cin_965155938
      - cin_112613370
      - cin_726709734
      - cin_832414051
      - cin_244209794
      - cin_443877629
  - Ultrasonic
    - Water-Level
      - cin_649136245
      - cin_82408737
      - cin_667841661
      - cin_175949715

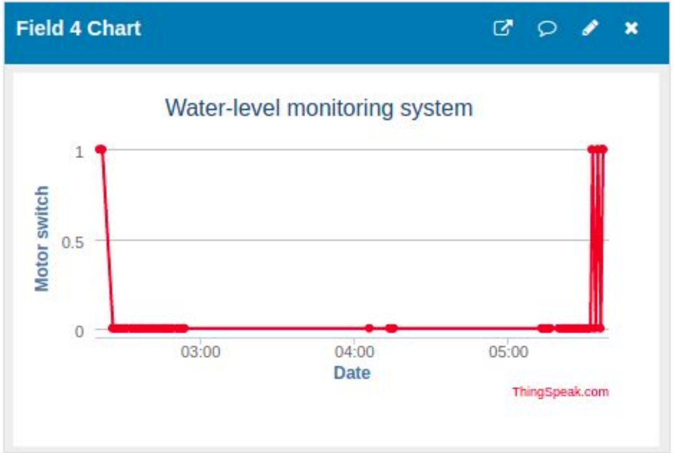
Attribute	Value
rn	Water-Level
ty	3
ri	/in-cse/cnt-815746350
pi	/in-cse/CAE567191314
ct	20211204T051833
lt	20211204T051833
acpi	AccessControlPolicyIDs /in-cse/acp-333552826
et	20221204T051833
st	17
mni	120
mbs	10000
mia	0
cni	17
cbs	81
ol	/in-cse/in-name/Ultrasonic/Water-Level/ol
la	/in-cse/in-name/Ultrasonic/Water-Level/la

THINGSPEAK


Field 1 Chart
Water-level monitoring system
27.5
25
22.5
Temperature
01:00
02:00
03:00
04:00
05:00
Date
ThingSpeak.com


Field 2 Chart
Water-level monitoring system
70
60
Humidity
01:00
02:00
03:00
04:00
05:00
Date
ThingSpeak.com

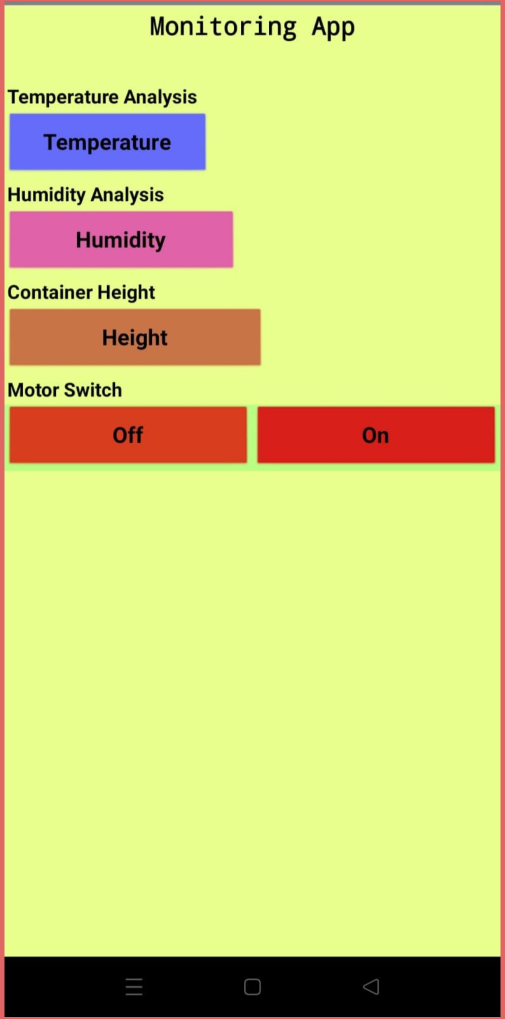

Field 3 Chart
Water-level monitoring system
1000
500
0
Water level
03:00
04:00
05:00
Date
ThingSpeak.com


Field 4 Chart
Water-level monitoring system
1
0.5
0
Motor switch
03:00
04:00
05:00
Date
ThingSpeak.com

CONTROL DEVICE

- **Temperature**
  Temperature analysis graph(Temp. vs Time)
- **Humidity**
  Humidity analysis graph(Hum. vs time)
- **Height**
  Height from the top of container to the water-level ,with both Chart and Widget.
- **Motor Switch**
  To control motor switch

App uses data from thingspeak, reads the value of switch field to power on/off the water pump.



TEAM - GHAJNI

Abhishek Sharma (2020101050)	:	Software part + M2M
Samarth Pandey (2020101048)	:	Hardware part
Vaibhav Agarwal(2020101041)	:	Assistance in software (MIT APP)
Parshwa bhadra(2020101001)	:	ThingSpeak