

Name	K.SHABEER
REGISTER NO	621421106046
COLLEGE CODE	6214
COURSE IBM	GROUP 5 IOT
NM ID	au621421106046

SMART WATER MANAGEMENT

IoT Based Water Level Indicator Using Ultrasonic Sensor

Today we will be working on a Water level indicator whose data can be monitored through a webpage over a local area network. The water level will be detected by using an ultrasonic distance measurement sensor. We have previously built another IoT based tank water level monitoring system using a float sensor, but in this project, we will use an ultrasonic sensor for detecting the level of water.

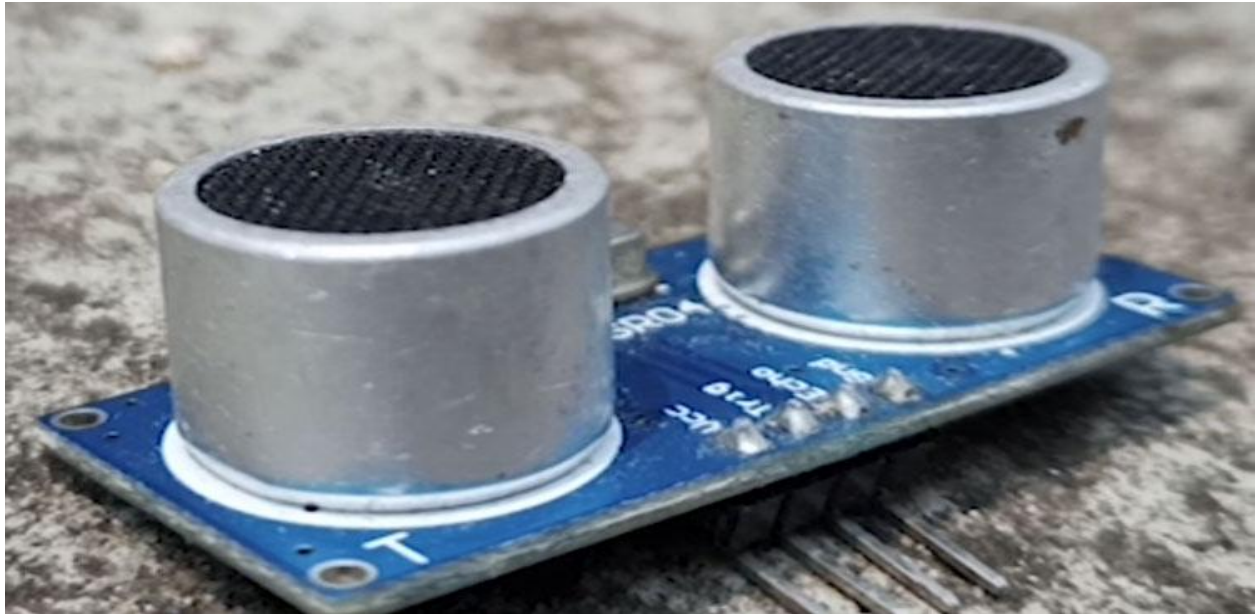


You must have seen several water level indicators but most of them use special float sensors that are not easily available and neither simple to use. But in this tutorial, we will be using an ultrasonic sensor for measuring the water level. You will come to know how further in the article.

For making the project more user friendly, we will be integrating it with a local webserver through which you can monitor the data from any device connected to the same Wi-Fi as your ESP board.

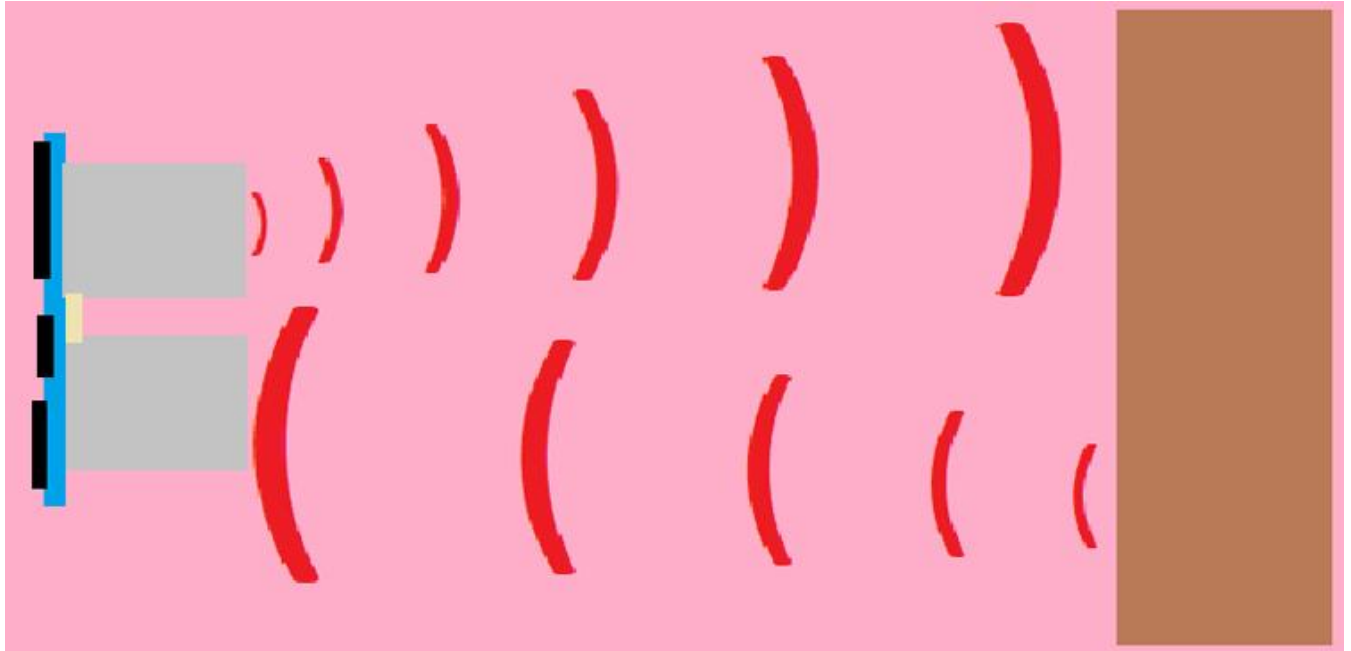
Ultrasonic Sensor Work and Use

Before understanding the working of the project, let us first see how an ultrasonic sensor works. A typical HC-SR04 ultrasonic sensor used in this project is shown below.



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 traveled to and from the target).



To calculate the distance between the sensor and the object, the sensor measures the time it takes between the emissions of the sound by the transmitter to its contact with the receiver. The formula for this calculation is

```
digitalWrite(TRIGGER, LOW);
```

```
long duration = pulseIn(ECHO, HIGH);
```

```
data = (duration/2) / 29.09;
```

So where can we use these sensors? We have previously used these sensors in Contactless Temperature measurement, IoT based Inventory

management, and many other projects, apart from this, it can also be used in Robot navigation, as well as factory automation. Water-level sensing is another good use and can be accomplished by positioning one sensor above a water surface. Another aquatic application is to use these sensors to “see” the bottom of a body of water, traveling through the water, but reflecting off the bottom surface below. We have used the same principle in our Flood Detection and Monitoring system as well.

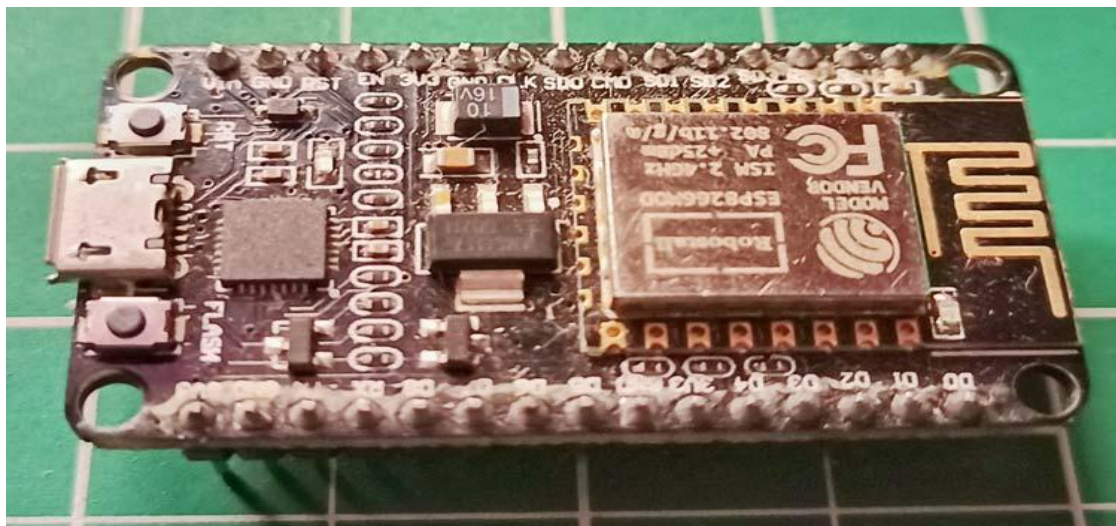
Ultrasonic Sensor for Measuring Water Level

So, now as we know the working of the ultrasonic sensor, it is pretty straightforward to understand the working of the project. We just have to read the values from the sensor and convert it to CMs and after doing all that, we have to publish the data in a local webserver that will be created by our NodeMCU board after getting connected to Wi-Fi.

Required Components

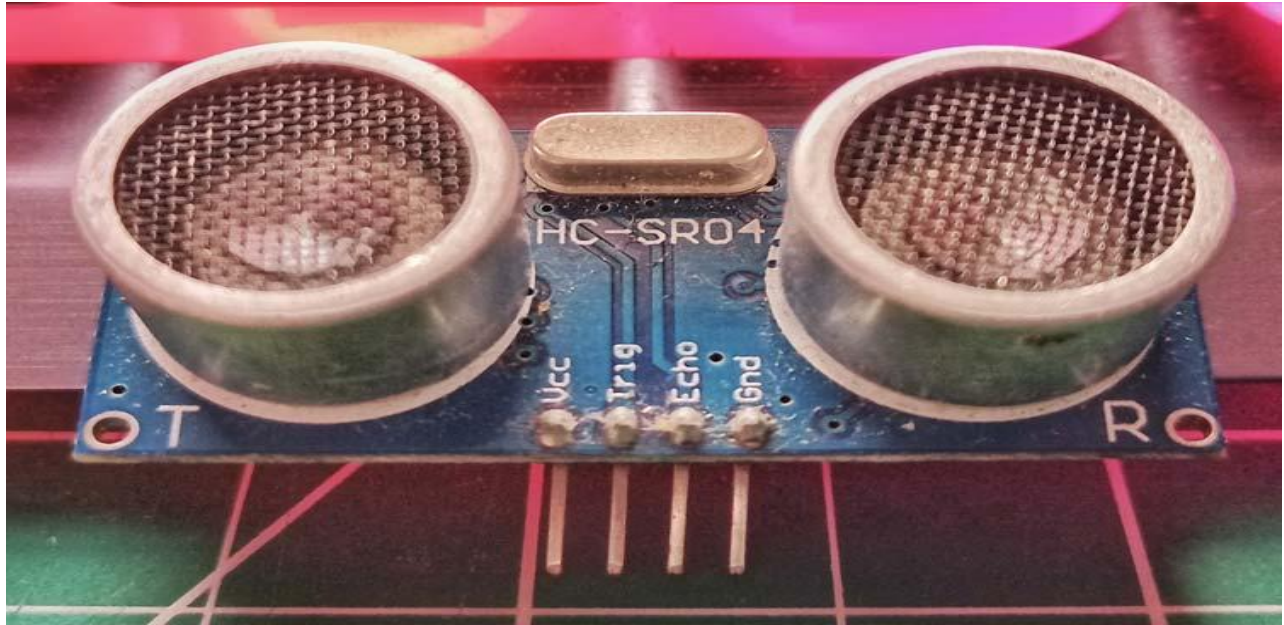
For making this project, we will be needing a few basic components.

ESP8266 NodeMCU board: This will be the heart of our whole project.



HC-SR04 Ultrasonic Sensor: This will be used for sensing the level through ultrasonic sound waves as explained earlier

HC SR04 Ultrasonic Sensor



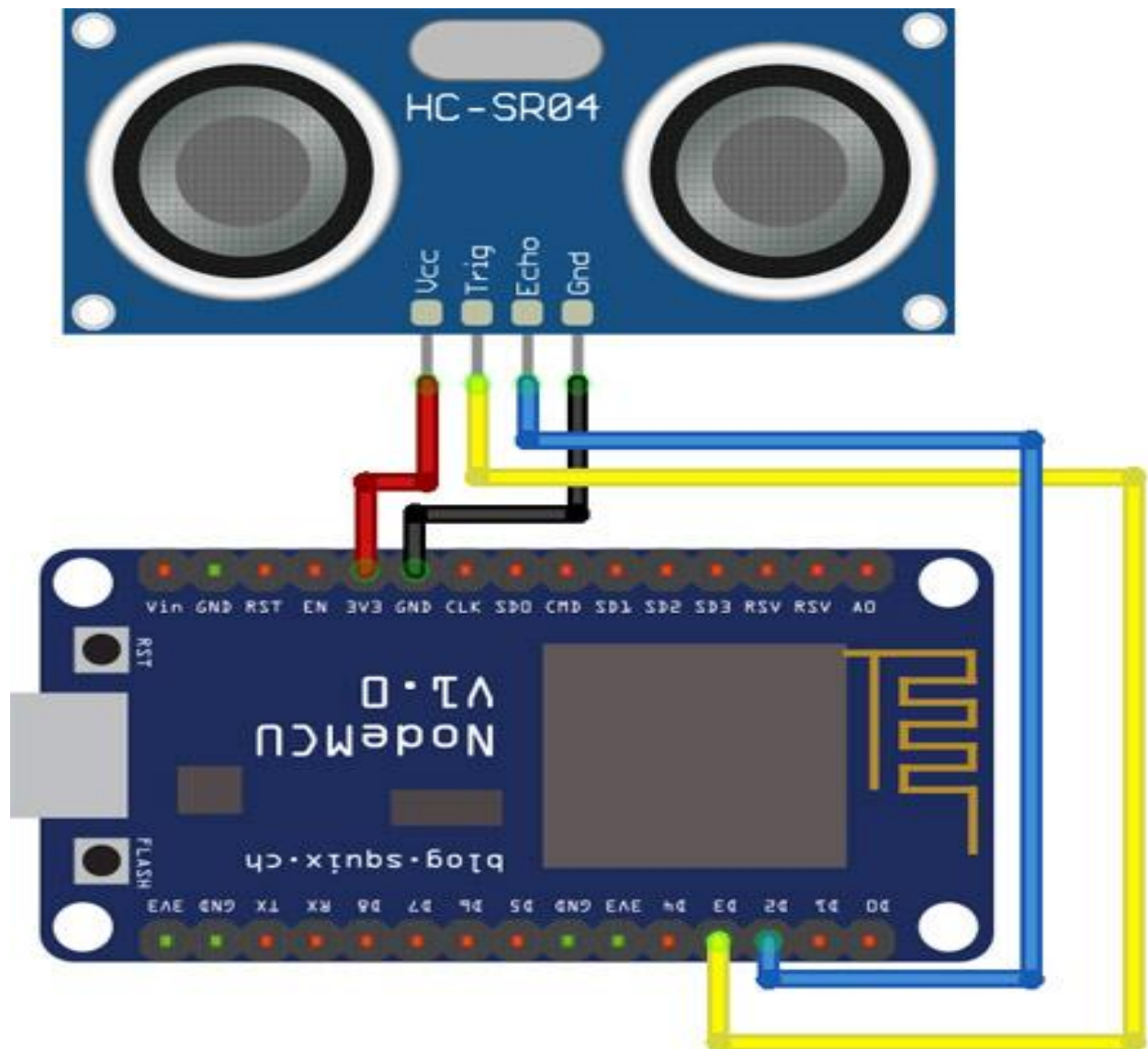
Breadboard: All the connections will be made on the breadboard itself for making it simple.

Jumper wires: As we are using a breadboard, jumper or hookup wires are the way to go for connections.

Interfacing Ultrasonic Sensor with NodeMCU

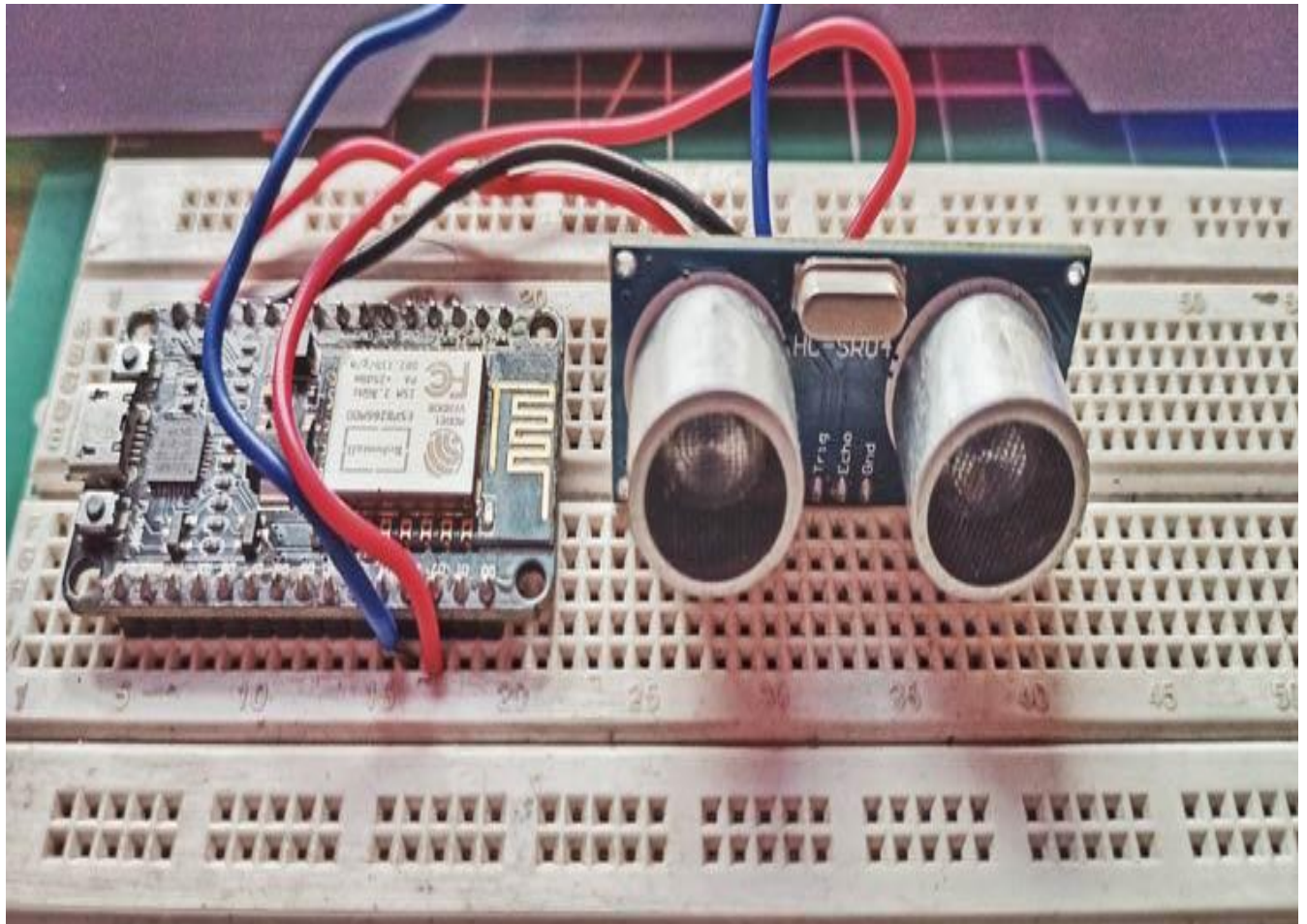
As we are not using many components, the IoT water level indicator circuit diagram used in this project is fairly simple. We just need to connect the HC-SR04 Ultrasonic sensor to our NodeMCU module. The rest of the work will be done via the software part itself.

Here is the circuit diagram for the connection.



I used a breadboard to connect my ultrasonic sensor with the NodeMCU and then used connecting wires

to make the connection. The testing set-up of mine looks like this below, later we will use longer jumper wires to mount the sensor over a demo water tank.



[Home](#)
[MAIN NAVIGATION](#)
[ARTICLES](#)
[NEWS](#)
[MOUSER ELECTRONICS STORE](#)

IOT PROJECTS

WIRELESS PROTOCOLS

IOT PLATFORMS

RECENT POSTS

Vi Offers IoT lab-as-a-service in Collaboration with C-DOT for IoT Device Interoperability Testing, Certification

Vi Offers IoT lab-as-a-service in Collaboration with C-DOT for IoT Device Interoperability Testing, Certification

Why Malware Attacks Are Increasingly Targeting IoT Devices, OT Infrastructure

Why Malware Attacks Are Increasingly Targeting IoT Devices, OT Infrastructure

Blynk 2.0 IoT Tutorial

Getting Started with Blynk 2.0 using NodeMCU

IIoT Gateways Expanding through OT and IT
How IIoT Gateways Playing An Important Role In Expanding OT and IT

Global Cellular IoT

Why Global Cellular IoT Module Shipments Shrunk in Q2 2023, Explain Analysts

Search

Search

ESP8266 06 Oct, 2020 / 3 Comments

IoT Based Water Level Indicator Using Ultrasonic Sensor

IoT Based Water Level Indicator Using Ultrasonic Sensor and NodeMCU

Today we will be working on a Water level indicator whose data can be monitored through a webpage over a local area network. The water level will be detected by using an ultrasonic distance measurement sensor. We have previously built another IoT based tank water level monitoring system using a float sensor, but in this project, we will use an ultrasonic sensor for detecting the level of water.

You must have seen several water level indicators but most of them use special float sensors that are

not easily available and neither simple to use. But in this tutorial, we will be using an ultrasonic sensor for measuring the water level. You will come to know how further in the article.

For making the project more user friendly, we will be integrating it with a local webserver through which you can monitor the data from any device connected to the same Wi-Fi as your ESP board.

Ultrasonic Sensor Work and Use

Before understanding the working of the project, let us first see how an ultrasonic sensor works. A typical HC-SR04 ultrasonic sensor used in this project is shown below.

Ultrasonic Sensor

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 traveled to and from the target).

Ultrasonic Sensor Working

To calculate the distance between the sensor and the object, the sensor measures the time it takes between the emissions of the sound by the transmitter to its contact with the receiver. The formula for this calculation is-

$D = \frac{1}{2} T \times C$ (where D is the distance, T is the time, and C is the speed of sound ~ 343 meters/second).

Ultrasonic Sensor Distance Measurement

So where can we use these sensors? We have previously used these sensors in Contactless

Temperature measurement, IoT based Inventory management, and many other projects, apart from this, it can also be used in Robot navigation, as well as factory automation. Water-level sensing is another good use and can be accomplished by positioning one sensor above a water surface. Another aquatic application is to use these sensors to “see” the bottom of a body of water, traveling through the water, but reflecting off the bottom surface below. We have used the same principle in our Flood Detection and Monitoring system as well.

Ultrasonic Sensor for Measuring Water Level

So, now as we know the working of the ultrasonic sensor, it is pretty straightforward to understand the working of the project. We just have to read the values from the sensor and convert it to CMs and after doing all that, we have to publish the data in a local webserver that will be created by our NodeMCU board after getting connected to Wi-Fi.

Required Components

For making this project, we will be needing a few basic components.

ESP8266 NodeMCU board: This will be the heart of our whole project.

ESP8266 NodeMCU

HC-SR04 Ultrasonic Sensor: This will be used for sensing the level through ultrasonic sound waves as explained earlier.

HC SR04 Ultrasonic Sensor

Breadboard: All the connections will be made on the breadboard itself for making it simple.

Jumper wires: As we are using a breadboard, jumper or hookup wires are the way to go for connections.

Interfacing Ultrasonic Sensor with NodeMCU

As we are not using many components, the IoT water level indicator circuit diagram used in this project is fairly simple. We just need to connect the HC-SR04 Ultrasonic sensor to our NodeMCU module. The rest of the work will be done via the software part itself.

Here is the circuit diagram for the connection.

IoT Water Level indicator Circuit Diagram

I used a breadboard to connect my ultrasonic sensor with the NodeMCU and then used connecting wires to make the connection. The testing set-up of mine looks like this below, later we will use longer jumper wires to mount the sensor over a demo water tank.

Interfacing Ultrasonic Sensor with NodeMCU

Programming NodeMCU to Read Water Level and Display on the Webserver

Now as we have finished making the connections. We will be proceeding to the coding part of the project. Let me clear the algorithm that we will be following while making the code. The complete code can be found at the bottom of this page.

What our Arduino code needs to do is, first of all, connect to the network whose SSID and password are mentioned in the code. After successfully getting, it needs to spit out the Local IP address to the serial monitor, so that we may access the webpage created by it. After doing so, it should start dealing with the ultrasonic sensor and after calculating the distance, it must send the data to the webpage that we have created.

Let us understand it part by part, so first of all, we are including the necessary libraries and the global variables that will be used further in the code. Along with that, we are also providing the SSID and passwords of our Wi-Fi router and initiating the webserver at port number 80.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
int TRIGGER = D3;
int ECHO = D2;
// Replace with your network credentials
const char* ssid = "xxxxx";
const char* password = "xxxxxxx";
ESP8266WebServer server(80); //instantiate server
at port 80 (http port)
String page = "";
int data;
```

Further, in the setup part, we have set the pin mode of the pins as input and output and have stored our page in the page variable using HTML. We have

kept a single line HTML code to make up our webpage. In the setup part itself, we get connected to the Wi-Fi router and print the IP address to the serial monitor.

The HTML page that we have created refreshes itself every three seconds. You may change it in the code itself.

```
void setup(void){
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  delay(1000);
  Serial.begin(115200);
  WiFi.begin(ssid, password); //begin WiFi
  connection
  Serial.println("");
  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");
```

```
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP());
server.on("/", [](){
    page = "<head><meta http-equiv=\"refresh\"
content=\"3\"></head><center><h1>Web based
Water Level monitor</h1><h3>Current water level is
:-</h3> <h4>"+String(data)+"</h4></center>";
    server.send(200, "text/html", page);
});
server.begin();
Serial.println("Web server started!");
}
```

Most of the work is done, now in the loop part, we just calculate the distance from the ultrasonic sensor, and after storing it, we publish the data over our webpage. Our webpage has been made in such a way that, it refreshes every 3 seconds.

```
void loop(void){
    digitalWrite(TRIGGER, LOW);
    delayMicroseconds(2);
```



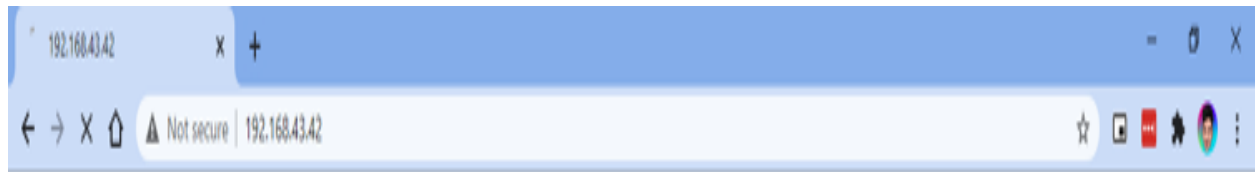
```
digitalWrite(TRIGGER, HIGH);  
delayMicroseconds(10);  
digitalWrite(TRIGGER, LOW);  
long duration = pulseIn(ECHO, HIGH);  
data = (duration/2) / 29.09;  
server.handleClient();  
}
```

This is pretty much about the code, Now just upload the code, and let us see the project in action.

IoT Based Water Level Indicator Testing and Working

This is the webpage that we have created using our NodeMCU board. It contains a heading that is, Web-based Water Level Indicator, below which is printing the live data coming through the HCSR04 Ultrasonic sensor. The water level values are printed in CMs, the less the value, the

less empty the container, and vice versa.

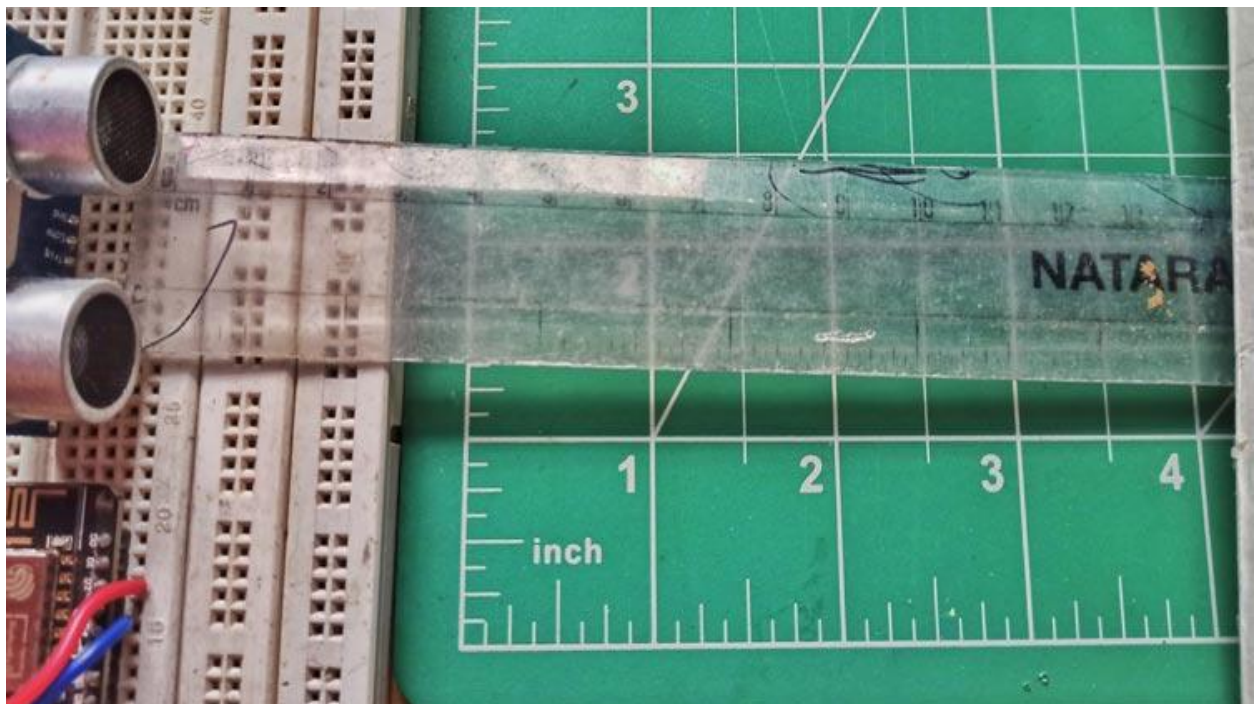


Web based Water Level monitor

Current water level is :-

14

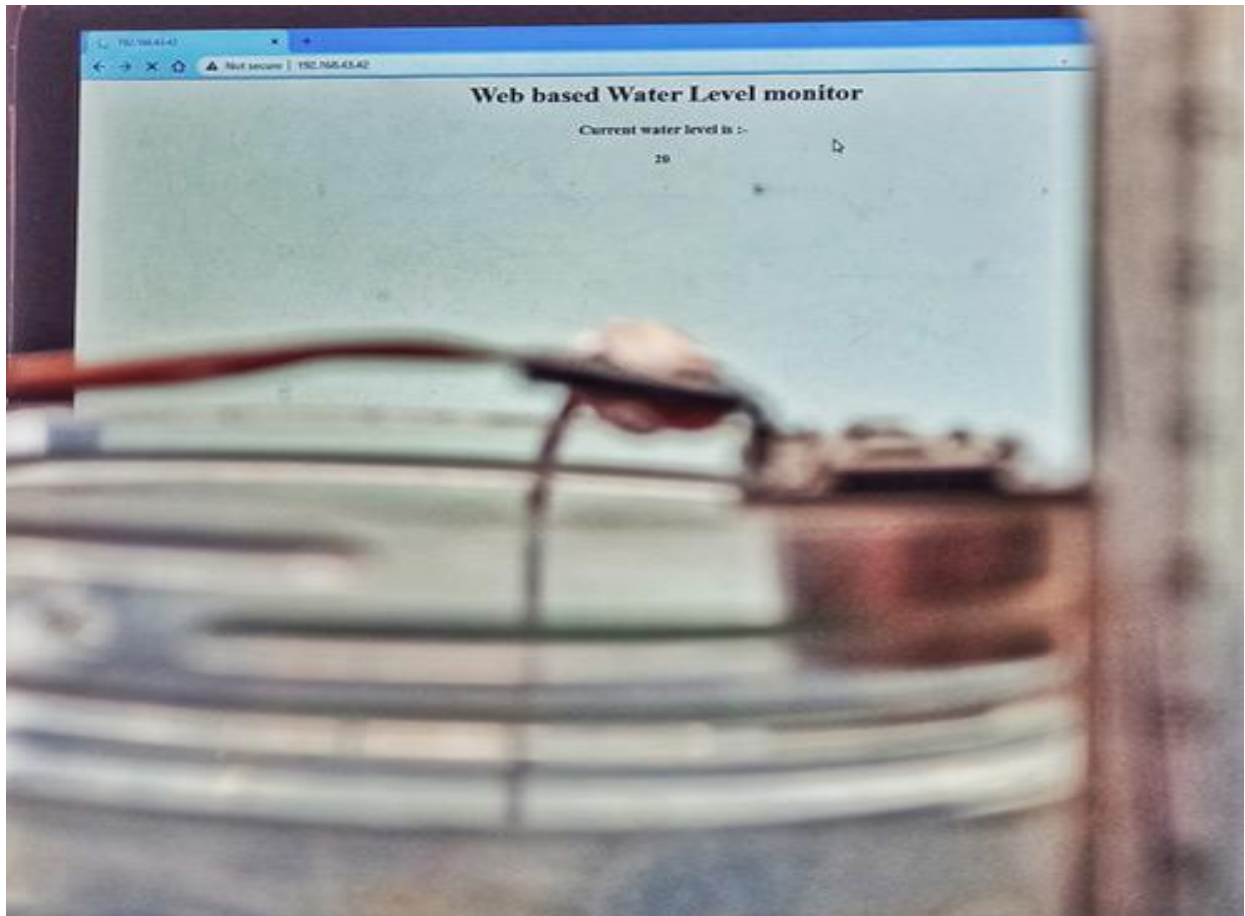
Before directly testing the water level, we tested the distance measurement. It worked like charm as you may see from the above and below images.



After testing this, we set up a water level apparatus for demonstrating the working. We affixed the ultrasonic sensor on top of the jar and also placed a ruler for scale. This way, the monitor will show the data of how empty the jar is in CMs. Here is how the setup looks like

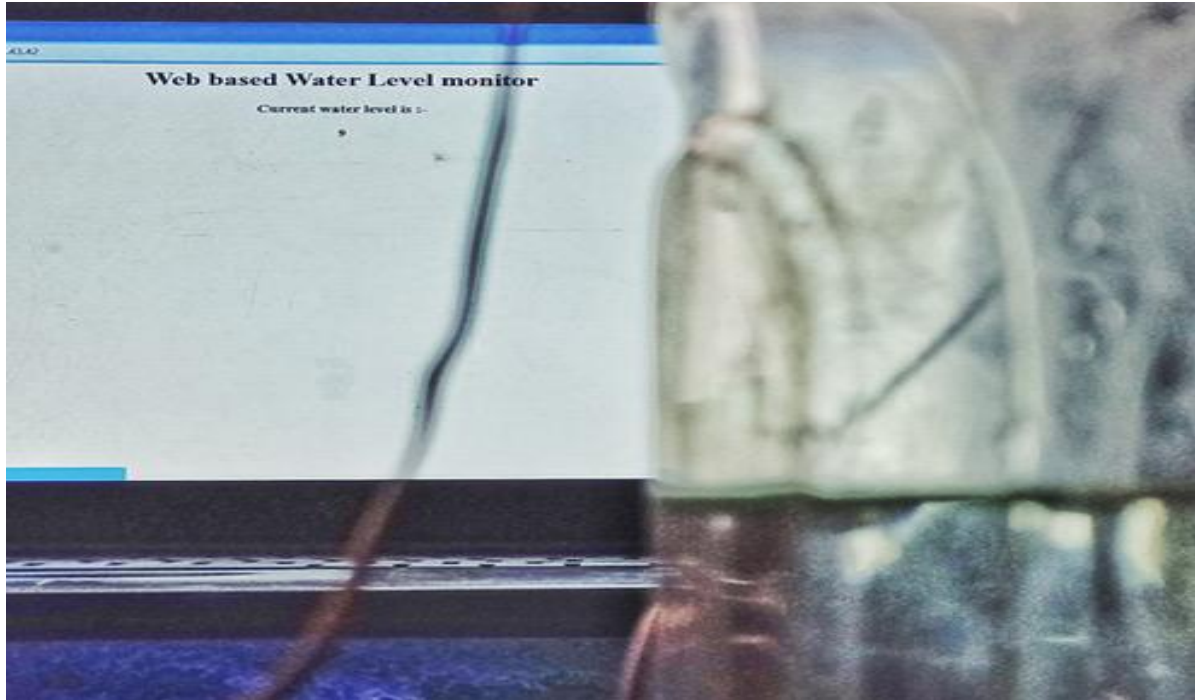


Now here is the working, firstly, we kept the jar empty, so it should display 20 in the server as the jar is 20cm deep and is empty as of now.



Water Level Indicator Testing

Now we have filled the jar by half so now it should display half the value of 20.



Water Level Indicator Working

For seeing the working in detail, you may see the video we have attached below. Hope you enjoyed the article and learned something useful from it. If you have any questions, you can leave them in the comment section below.

Code

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
int TRIGGER = D3;
int ECHO = D2;
```

```
// Replace with your network credentials
const char* ssid = "Comet";
const char* password = "evilzebra";
ESP8266WebServer server(80); //instantiate server at
port 80 (http port)
String page = "";
int data;
void setup(void){
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  delay(1000);
  Serial.begin(115200);
  WiFi.begin(ssid, password); //begin WiFi connection
  Serial.println("");
  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  server.on("/", [](){
```

```
    page = "<head><meta http-equiv=\"refresh\"  
content=\"3\"></head><center><h1>Web based Water  
Level monitor</h1><h3>Current water level is :-</h3>  
<h4>"+String(data)+"</h4></center>";  
    server.send(200, "text/html", page);  
});  
server.begin();  
Serial.println("Web server started!");  
}  
void loop(void){  
    digitalWrite(TRIGGER, LOW);  
    delayMicroseconds(2);  
    digitalWrite(TRIGGER, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(TRIGGER, LOW);  
    long duration = pulseIn(ECHO, HIGH);  
    data = (duration/2) / 29.09;  
    server.handleClient();  
}
```