**qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm**

|  |
| --- |
| INDUSTRIAL ELECTONICS PROJECT  IOT BASED AUTOMATIC WATER IRRIGATION SYSTEM  SUBMITTED TO SIR RIAZ UN NABI  c |

**IOT BASED AUTOMATIC IRRIGATION SYSTEM**

**COMPONENTS**

1. I2C LCD shield
2. Ultrasonic sensor
3. Soil moisture sensor
4. Pump
5. Nodemcu 0.1
6. Relay
7. Connecting wires
8. Adapters

**WORKING PRINCIPLE**

In this project we have used two sensors ultrasonic to measure the water level in the tank and the soil mositure sensor to determine the moisture content in the soil.The prototype is programmed this way that as the moisture level drops down to 40% the pump is turned on as long as there is enough water in the tank. Since, this is an IOT based project we have used NodeMCU 0.1 ESP8266 embedded controller board through which we are sending the real time data of both the sensors to cloud by using the cloud domain “THINGSPEAK” with a delay of 16 sec.The advantage of using this board is that it directly configures with the server without running any AT COMMANDS.

**PROGRAM:-**

#include <ESP8266WiFi.h>

#include<LiquidCrystal\_I2C.h>

#include<Wire.h>

LiquidCrystal\_I2C lcd(0x27,16,2);

#define trigPin 12//D6 ON NODE MCU

#define echoPin 13 //D7

#define motor 14 //D5

int sensorPin =A0;

int sensorValue = 0;

int percentvalue=0;

float duration;

float distance;

const char\* host = "api.thingspeak.com"; // Your cloud domain

String ApiKey = "PH5LX8XHC300MAER"; // Change this key to your "Write API key"

String path = "/update?key=" + ApiKey +"&field1="; // API Format

const char\* ssid = "ELDFSev"; // Change this to your ssid\_name (WiFi Name)

const char\* pass = "neduet33iacc44&";

void setup() {

pinMode(sensorPin,INPUT);

pinMode(echoPin,INPUT);

pinMode(trigPin,OUTPUT);

pinMode(motor,OUTPUT);

Wire.begin(4,5);

lcd.begin(16,2);

lcd.backlight();

Serial.begin(9600);

WiFi.begin(ssid, pass); // Func to Connect to WiFi

while (WiFi.status() != WL\_CONNECTED) { // Wait for connection

delay(100);

Serial.print(".");

}//

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid); // Display confirmation msg to PC

Serial.print("IP address: ");

Serial.println(WiFi.localIP()); // D

}

void loop() {

sensorValue = analogRead(sensorPin);

Serial.print("at sensor read= ");

Serial.println(sensorValue);

percentvalue=map(sensorValue,1023,0,0,100);

delay(1000);

digitalWrite (trigPin,LOW);

delayMicroseconds (2);

digitalWrite (trigPin, HIGH);

delayMicroseconds (10);

digitalWrite (trigPin, LOW);

duration = pulseIn (echoPin, HIGH);

distance = duration\*0.034/2 ;

delay(1000);

Serial.println("MOISTURELEVEL"+String(percentvalue)+"%");

Serial.println("WATER LEVEL"+String(distance)+"cm");

lcd.clear();

lcd.setCursor(0,0);

lcd.print("MOISTURELEVEL"+String(percentvalue)+"%");

lcd.setCursor(0,1);

lcd.print("WATER LEVEL"+String(distance)+"cm");

delay(3000);

if(distance<=4 && percentvalue<=20)

{digitalWrite(motor,LOW);//AS RELAY OPERATES ON NEGATIVE LOGIC

Serial.print("MOTOR IS TURNED ON");

lcd.setCursor(0,0);

lcd.print("MOTOR IS TURNED ON");

}

else if(percentvalue>=40)

{digitalWrite(motor,HIGH);

Serial.println("THE MOISTURE IS ENOUGH....");

Serial.println("MOTOR IS TURNED OFF");

lcd.clear();

lcd.setCursor(0,0);

lcd.print("THE MOISTURE IS ENOUGH....");

lcd.setCursor(0,1);

lcd.print("MOTOR IS TURNED OFF");}

else if(distance>11)

{digitalWrite(motor,HIGH);

Serial.println("NOT ENOUGH WATER IN TANK");

lcd.clear();

lcd.setCursor(0,0);

lcd.print("NOT ENOUGH WATER");

}

WiFiClient client; // Create object of Class WiFiClient

const int httpPort = 80; // Port No. 80

if (!client.connect(host, httpPort)) { // If Couldn't connect

Serial.println("connection failed"); // Display error msg to PC

return;

}

client.print(String("GET ") + path + String(percentvalue)+"&field2="+String(distance)+ " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"Connection: keep-alive\r\n\r\n");

}

**NODEMCU 0.1**

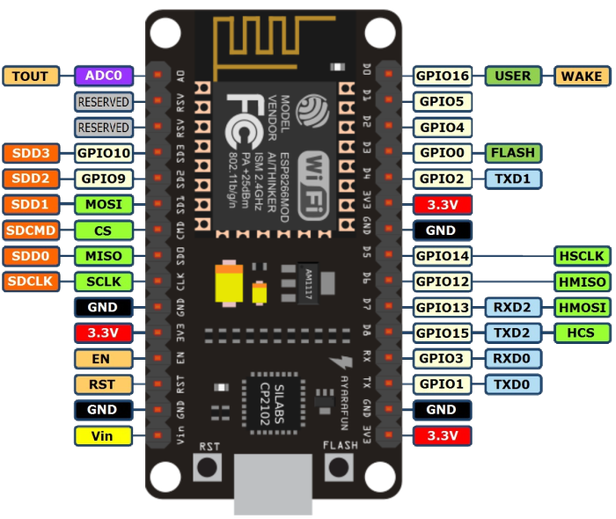
NodeMCU is an open source [LUA](https://www.lua.org/start.html) based firmware developed for ESP8266 wifi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board.

NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a low-cost [Wi-Fi](https://en.wikipedia.org/wiki/Wi-Fi) chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer [ESP8266 WiFi Module](http://www.electronicwings.com/sensors-modules/esp8266-wifi-module).

NodeMCU provides access to the [GPIO](https://en.wikipedia.org/wiki/General-purpose_input/output) (General Purpose Input/Output) and a pin mapping table is part of the API documentation.

|  |  |
| --- | --- |
| **I/O index** | **ESP8266 pin** |
| 0 [\*] | GPIO16 |
| 1 | GPIO5 |
| 2 | GPIO4 |
| 3 | GPIO0 |
| 4 | GPIO2 |
| 5 | GPIO14 |
| 6 | GPIO12 |
| 7 | GPIO13 |
| 8 | GPIO15 |
| 9 | GPIO3 |
| 10 | GPIO1 |
| 11 | GPIO9 |
| 12 | GPIO10 |

[\*] D0 (GPIO16) can only be used for GPIO read/write. It does not support open-drain/interrupt/PWM/I²C or 1-Wire.



# **Where to store data from sensors – ThingSpeak**

Today we take a look at well-known service called [ThingSpeak](https://iotblog.cz/go/thingspeak). ThingSpeak is cooperating with MathWorks – company best known for creating MATLAB software. Project launched in 2010. They even provide older version of their service [as opensource](https://github.com/iobridge/thingspeak).

### What does it offer?

ThingSpeak provides API for storing data (strings, decimals, integers,…) from your sensors and other IoT devices connected to the Internet. You can use REST API over HTTPS or MQTT protocol. Every user can create “unlimited” number of *channels* which represent one particular device. Every channel allows to store 8 different values as *fields*. Entire message from device cannot exceed 3000 bytes – which is more than enough for most use cases.

