Greenhouse system Automation.

Name of the Group Members:

1. Wordh Ul Hasan 1330471642
2. Rahatul Ashakin 1320301042
3. Sabrina Afrin Simi 141110304
4. Wasim Iqbal 1331192042

**Functionalities:**

* For lowering temperature we will be using cooling fans,
* To raise temperature will be using heaters but for the purpose of demonstration will be using Indication bulb
* To decrease humidity level we will be using exhaust fans.
* Increase moisture level using a water pump connected to a water tank.
* Can make all the data available over the internet and make this an IOT project

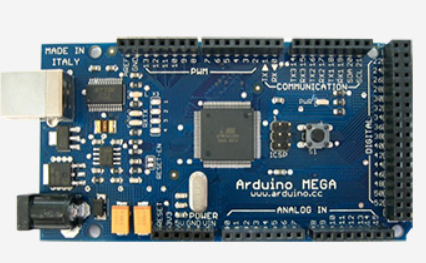
**Overview:**

A complete greenhouse monitoring and controlling system, that is automated, updating each and every detail on the Internet that can be accessed from anywhere. For sensors, it uses a light sensor, temperature sensor, moisture sensor, humidity sensors. So it would be a way more easy to have the automation do all the hard work and maintain the stability of the greenhouse system.

We will constantly monitor and control environment conditions in the greenhouse to ensure it remains at preset temperature, light, moisture and humidity levels.

**Equipments:**

1. Arduino Mega:



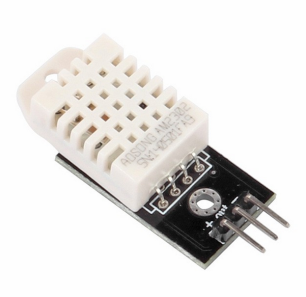
The Arduino Mega is a microcontroller board based on the ATmega1280.It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

1. Exhaust Fan & Cooling Fan



The cooling fans operating voltage is 8 volts.

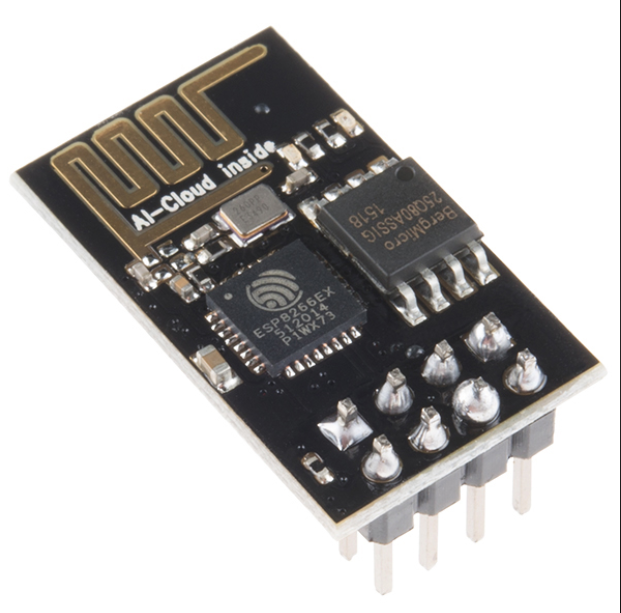
1. Humidity & Temperature Sensor Module (DHT 22)

 Operating voltage is 5 volts

1. Power Supply (LIPO Battery)



1. WIFI Module (ESP826)



[ESP8266](http://espressif.com/en/products/esp8266/) (presently ESP8266EX) is a chip with which manufacturers are making wirelessly networkable micro-controller modules. More specifically, ESP8266 is a system-on-a-chip (SoC) with capabilities for 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2), general-purpose input/output (16 GPIO), Inter-Integrated Circuit (I²C), analog-to-digital conversion (10-bit ADC), Serial Peripheral Interface (SPI), I²S interfaces with DMA (sharing pins with GPIO), UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2), and pulse-width modulation (PWM). It employs a 32-bit RISC CPU based on the Tensilica Xtensa L106 running at 80 MHz (or overclocked to 160 MHz). It has a 64 KB boot ROM, 64 KB instruction RAM and 96 KB data RAM. External flash memory can be accessed through SPI.

1. To increase the humidity level of the system we need have water pumping system
2. Bulb to increase the heating level of the system

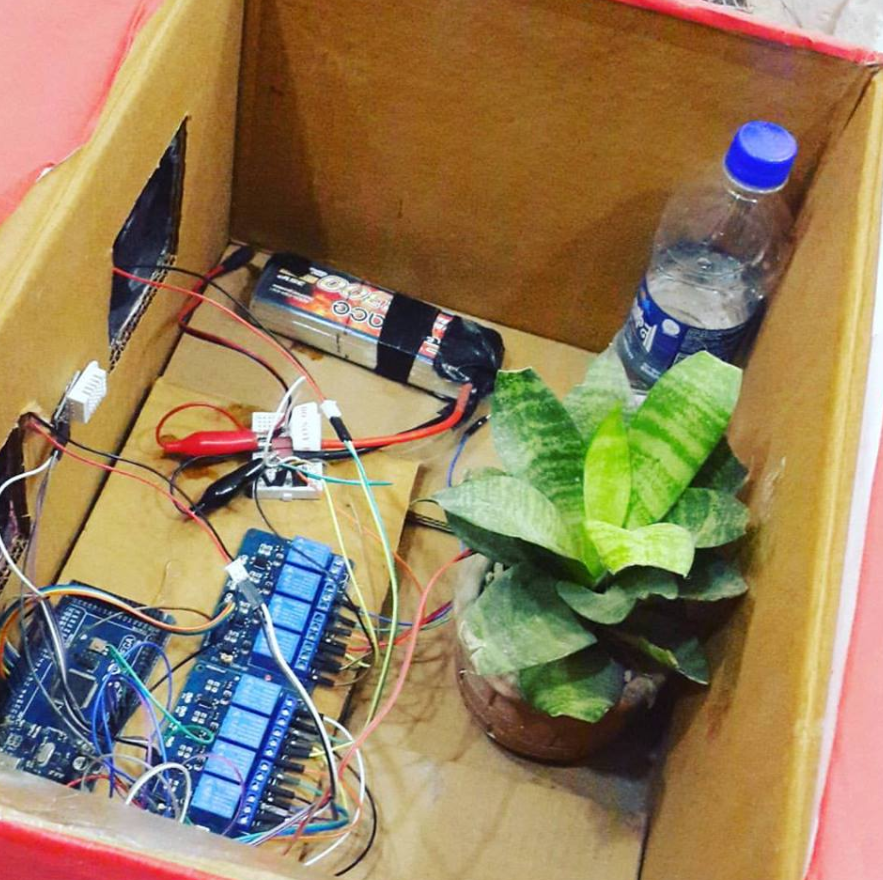


Part 1:

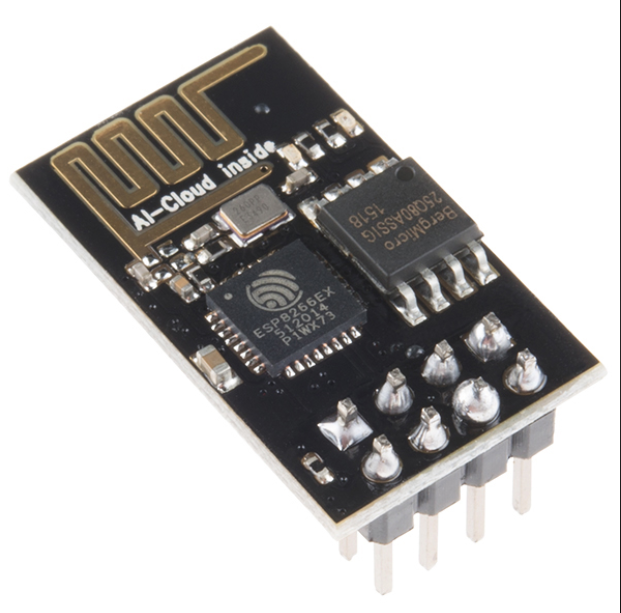
* Interfacing the Input Sensors and Output Devices with the microcontroller
* Programming the micro controller to work get the desired output
* Set up a demo Green House System using CardBoard

Part 2:

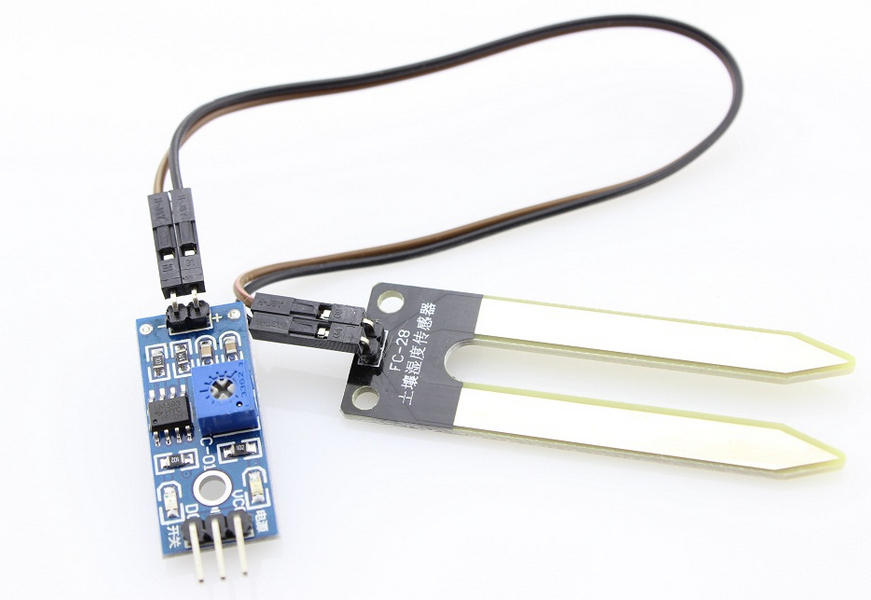
* Storing The Data in a server
* Accessing the data using an App and Show the statistics



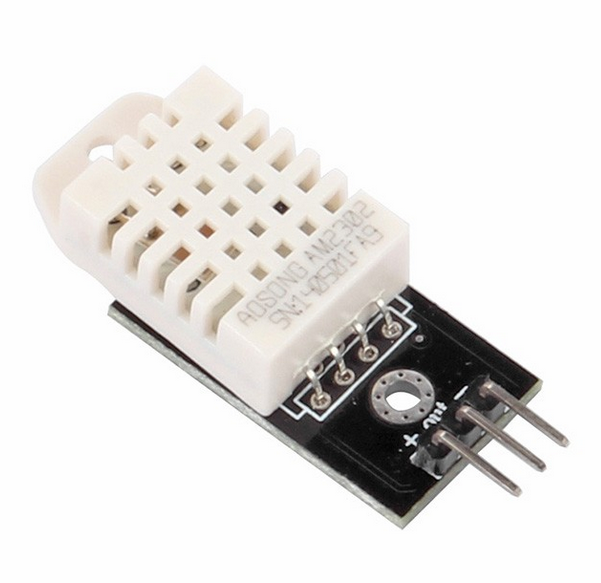
Wifi Module



Soil Moisture Sensor



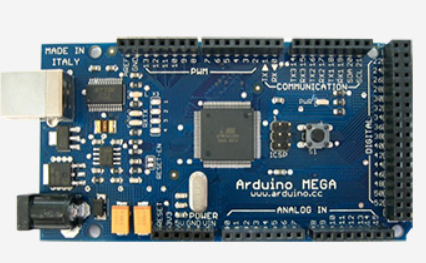
DHT 22



Relay

Relay

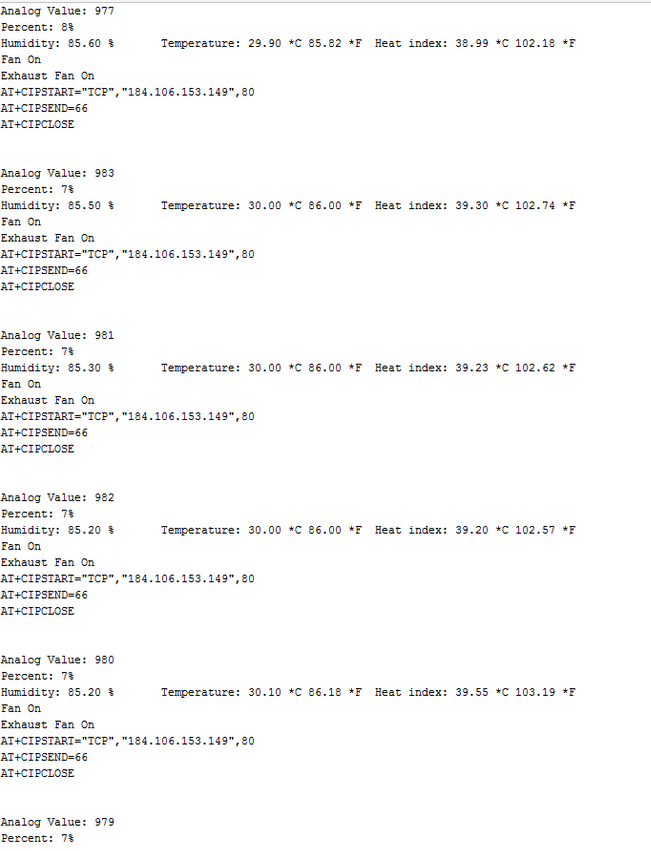
Arduino Mega



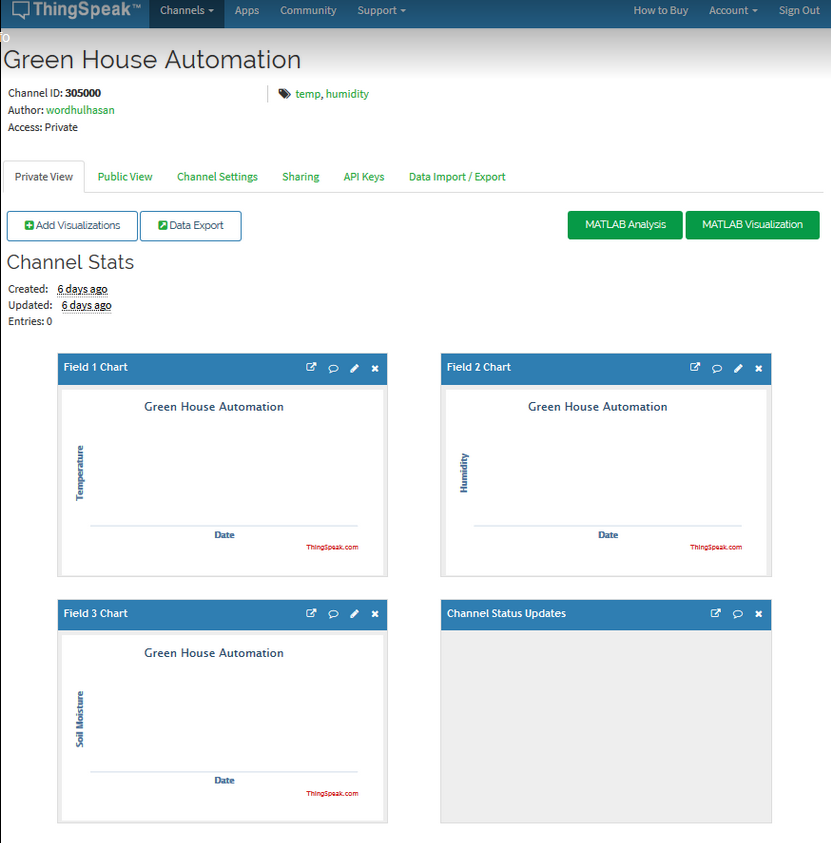


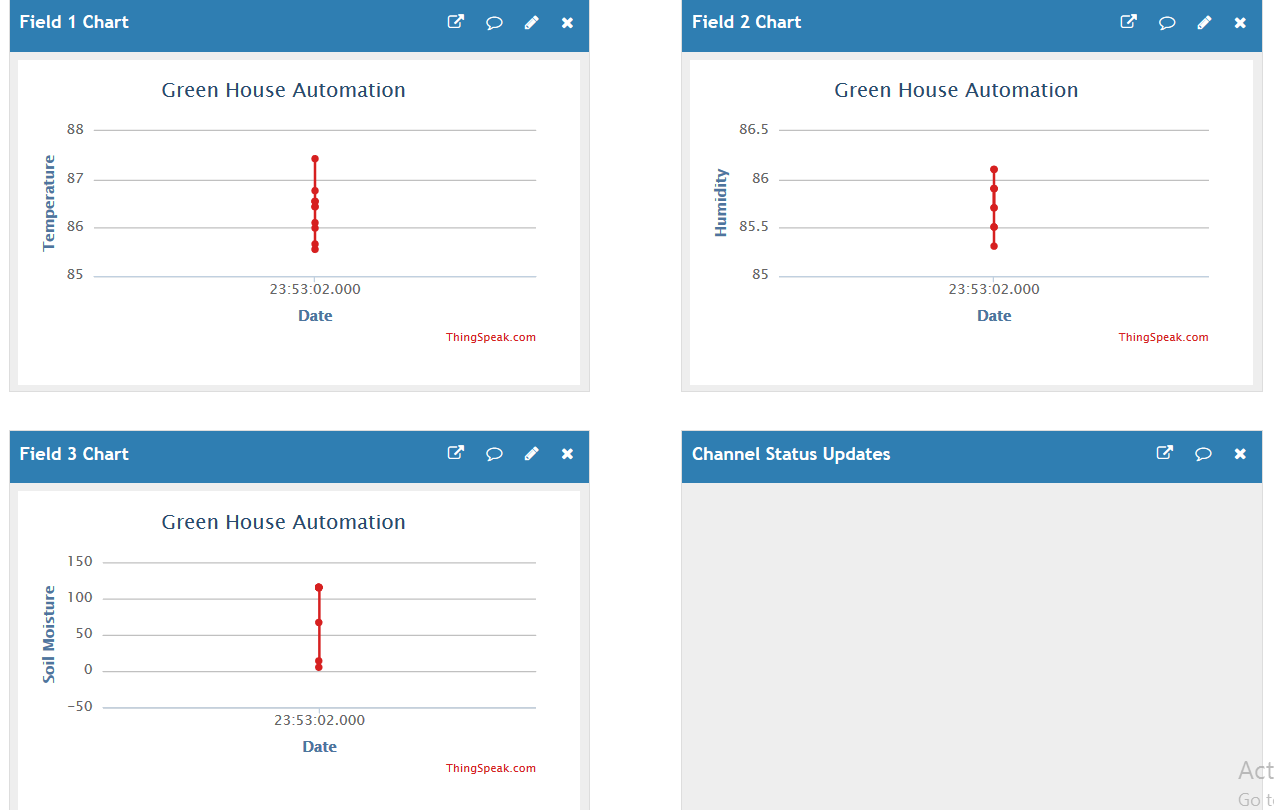
Water Pump Motor

**Output of the Serial Monitor:**

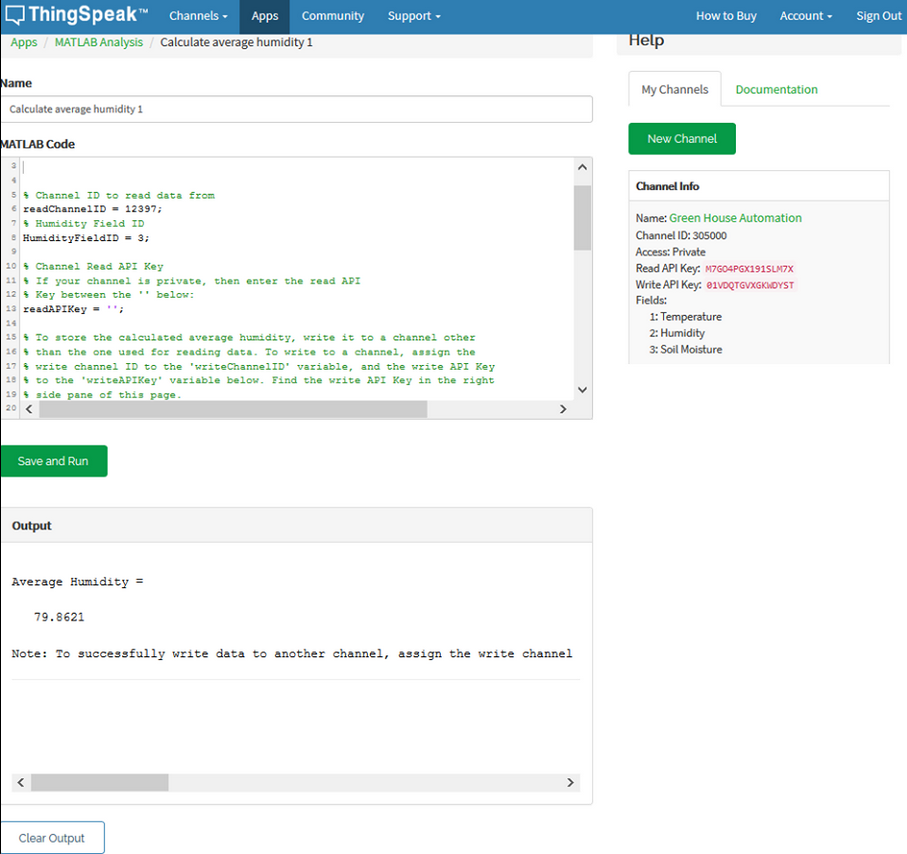
****

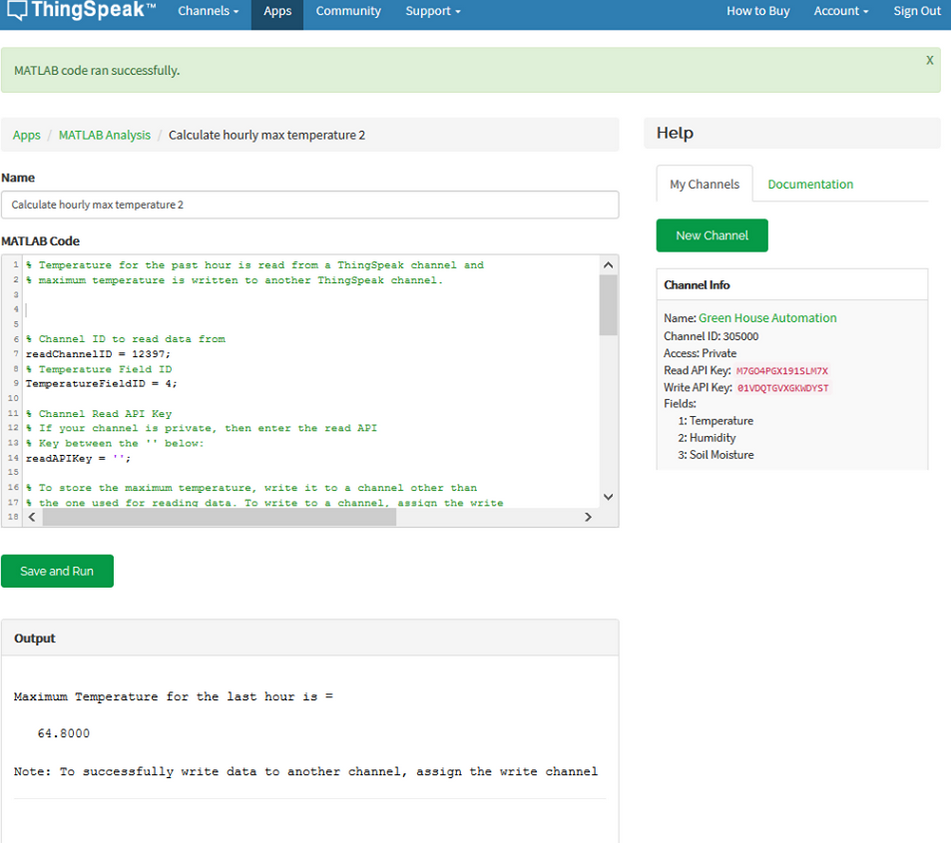
**Updates Values in the server ThingSpeak.com**

****

****

**Using Matlab to generate different types of values and Visualization in the server:**

****

****

**Algorithm:**

**/\* YourDuino Example: Relay Control 1.10**

**Handles "Relay is active-low" to assure**

**no relay activation from reset until**

**application is ready.**

**terry@yourduino.com \*/**

**// Code to use SoftwareSerial**

**#include <SoftwareSerial.h>**

**SoftwareSerial espSerial = SoftwareSerial(15,14); // arduino RX pin=15 arduino TX pin=3 connect the arduino RX pin to esp8266 module TX pin - connect the arduino TX pin to esp8266 module RX pin**

**/\*-----( Import needed libraries )-----\*/**

**#include "DHT.h"**

**/\*-----( Declare Constants )-----\*/**

**#define RELAY\_ON 0**

**#define RELAY\_OFF 1**

**/\*-----( Declare objects )-----\*/**

**/\*-----( Declare Variables )-----\*/**

**#define Relay\_1 2 // Arduino Digital I/O pin number**

**#define Relay\_2 3**

**#define Relay\_3 4**

**#define Relay\_4 5**

**#define Relay\_5 7**

**#define Relay\_6 8**

**#define Relay\_7 9**

**#define Relay\_8 10**

**#define DHTPIN 6 // what digital pin we're connected to**

**#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321**

**DHT dht(DHTPIN, DHTTYPE);**

**String apiKey = "01VDQTGVXGKWDYST"; // replace with your channel's thingspeak WRITE API key**

**String ssid="No\_Internet"; // Wifi network SSID**

**String password ="angel3070"; // Wifi network password**

**int sensorPin = A0;**

**int sensorValue = 0;**

**int percent = 0;**

**boolean DEBUG=true;**

**//======================================================================== showResponce**

**void showResponse(int waitTime){**

**long t=millis();**

**char c;**

**while (t+waitTime>millis()){**

**if (espSerial.available()){**

**c=espSerial.read();**

**if (DEBUG) Serial.print(c);**

**}**

**}**

**}**

**//========================================================================**

**boolean thingSpeakWrite(float value1, float value2){**

**String cmd = "AT+CIPSTART=\"TCP\",\""; // TCP connection**

**cmd += "184.106.153.149"; // api.thingspeak.com**

**cmd += "\",80";**

**espSerial.println(cmd);**

**if (DEBUG) Serial.println(cmd);**

**if(espSerial.find("Error")){**

**if (DEBUG) Serial.println("AT+CIPSTART error");**

**return false;**

**}**

**String getStr = "GET /update?api\_key="; // prepare GET string**

**getStr += apiKey;**

**getStr +="&field1=";**

**getStr += String(value1);**

**getStr +="&field2=";**

**getStr += String(value2);**

**// getStr +="&field3=";**

**// getStr += String(value3);**

**// ...**

**getStr += "\r\n\r\n";**

**// send data length**

**cmd = "AT+CIPSEND=";**

**cmd += String(getStr.length());**

**espSerial.println(cmd);**

**if (DEBUG) Serial.println(cmd);**

**delay(100);**

**if(espSerial.find(">")){**

**espSerial.print(getStr);**

**if (DEBUG) Serial.print(getStr);**

**}**

**else{**

**espSerial.println("AT+CIPCLOSE");**

**// alert user**

**if (DEBUG) Serial.println("AT+CIPCLOSE");**

**return false;**

**}**

**return true;**

**}**

**void setup() /\*\*\*\*\*\* SETUP: RUNS ONCE \*\*\*\*\*\*/**

**{**

**DEBUG=true; // enable debug serial**

**//-------( Initialize Pins so relays are inactive at reset)----**

**digitalWrite(Relay\_1, RELAY\_OFF);**

**digitalWrite(Relay\_2, RELAY\_OFF);**

**digitalWrite(Relay\_3, RELAY\_OFF);**

**digitalWrite(Relay\_4, RELAY\_OFF);**

**digitalWrite(Relay\_5, RELAY\_OFF);**

**digitalWrite(Relay\_6, RELAY\_OFF);**

**digitalWrite(Relay\_7, RELAY\_OFF);**

**digitalWrite(Relay\_8, RELAY\_OFF);**

**//---( THEN set pins as outputs )----**

**pinMode(Relay\_1, OUTPUT);**

**pinMode(Relay\_2, OUTPUT);**

**pinMode(Relay\_3, OUTPUT);**

**pinMode(Relay\_4, OUTPUT);**

**pinMode(Relay\_5, OUTPUT);**

**pinMode(Relay\_6, OUTPUT);**

**pinMode(Relay\_7, OUTPUT);**

**pinMode(Relay\_8, OUTPUT);**

**delay(4000); //Check that all relays are inactive at Reset**

**dht.begin();**

**espSerial.begin(115200); // enable software serial**

**espSerial.println("AT+CWMODE=1"); // set esp8266 as client**

**showResponse(1000);**

**espSerial.println("AT+CWJAP=\""+ssid+"\",\""+password+"\""); // set your home router SSID and password**

**showResponse(5000);**

**if (DEBUG) Serial.println("Setup completed");**

**}//--(end setup )---**

**int convertToPercent(int value)**

**{**

**int percentValue = 0;**

**percentValue = map(value, 1023, 465, 0, 100);**

**return percentValue;**

**}**

**void printValuesToSerial()**

**{**

**Serial.print("\n\nAnalog Value: ");**

**Serial.print(sensorValue);**

**Serial.print("\nPercent: ");**

**Serial.print(percent);**

**Serial.print("%");**

**Serial.print("\n");**

**}**

**void loop() /\*\*\*\*\*\* LOOP: RUNS CONSTANTLY \*\*\*\*\*\*/**

**{**

**// Wait a few seconds between measurements.**

**delay(2000);**

**// Reading temperature or humidity takes about 250 milliseconds!**

**// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)**

**float h = dht.readHumidity();**

**// Read temperature as Celsius (the default)**

**float t = dht.readTemperature();**

**// Read temperature as Fahrenheit (isFahrenheit = true)**

**float f = dht.readTemperature(true);**

**// Check if any reads failed and exit early (to try again).**

**if (isnan(h) || isnan(t) || isnan(f)) {**

**Serial.println("Failed to read from DHT sensor!");**

**return;**

**}**

**// Compute heat index in Fahrenheit (the default)**

**float hif = dht.computeHeatIndex(f, h);**

**// Compute heat index in Celsius (isFahreheit = false)**

**float hic = dht.computeHeatIndex(t, h, false);**

**sensorValue = analogRead(sensorPin);**

**percent = convertToPercent(sensorValue);**

**printValuesToSerial();**

**delay(1000);**

**Serial.print("Humidity: ");**

**Serial.print(h);**

**Serial.print(" %\t");**

**Serial.print("Temperature: ");**

**Serial.print(t);**

**Serial.print(" \*C ");**

**Serial.print(f);**

**Serial.print(" \*F\t");**

**Serial.print("Heat index: ");**

**Serial.print(hic);**

**Serial.print(" \*C ");**

**Serial.print(hif);**

**Serial.println(" \*F");**

**if(f>80) // Temp High**

**{**

**digitalWrite(Relay\_3, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_4, RELAY\_ON);// set the Relay ON**

**delay(1000); // wait for a second**

**Serial.println("Fan On");**

**}**

**else {**

**digitalWrite(Relay\_3, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_4, RELAY\_OFF);// set the Relay ON**

**delay(1000); // wait for a second**

**Serial.println("Fan OFF");**

**}**

**if(f<60) // Temp Low**

**{**

**digitalWrite(Relay\_7, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_8, RELAY\_ON);// set the Relay ON**

**delay(1000); // wait for a second**

**}**

**else{**

**digitalWrite(Relay\_7, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_8, RELAY\_OFF);// set the Relay ON**

**delay(1000); // wait for a second**

**}**

**if(h>70) // Humidity High**

**{**

**digitalWrite(Relay\_1, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_2, RELAY\_ON);// set the Relay ON**

**delay(1000); // wait for a second**

**Serial.println("Exhaust Fan On");**

**}**

**else {**

**digitalWrite(Relay\_1, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_2, RELAY\_OFF);// set the Relay ON**

**delay(1000); // wait for a second**

**Serial.println("Exhaust Fan OFF");**

**}**

**if(percent <30)**

**{**

**digitalWrite(Relay\_5, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_6, RELAY\_ON);// set the Relay ON**

**delay(1000); // wait for a second**

**}**

**else {**

**digitalWrite(Relay\_5, RELAY\_OFF);// set the Relay OFF**

**delay(1000); // wait for a second**

**digitalWrite(Relay\_6, RELAY\_OFF);// set the Relay ON**

**delay(1000); // wait for a second**

**}**

**thingSpeakWrite(t,h);**

**delay(20000);**

**}//--(end main loop )---**

**//\*\*\*\*\*\*\*\*\*( THE END )\*\*\*\*\*\*\*\*\*\*\***