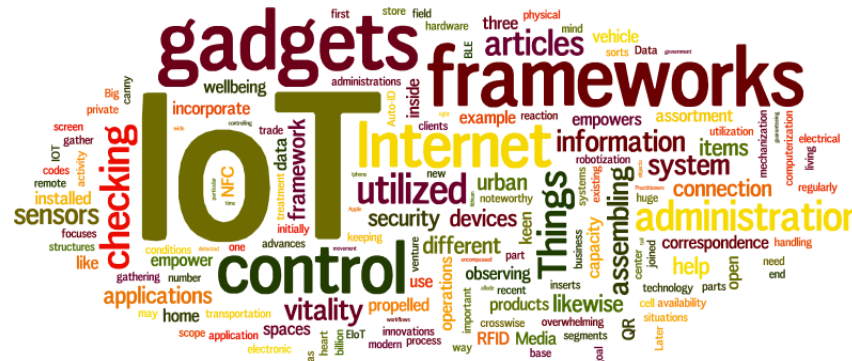


CS578: Internet of Things

Smart Home Monitoring Using ESP8266 and Webserver



Dr. Manas Khatua

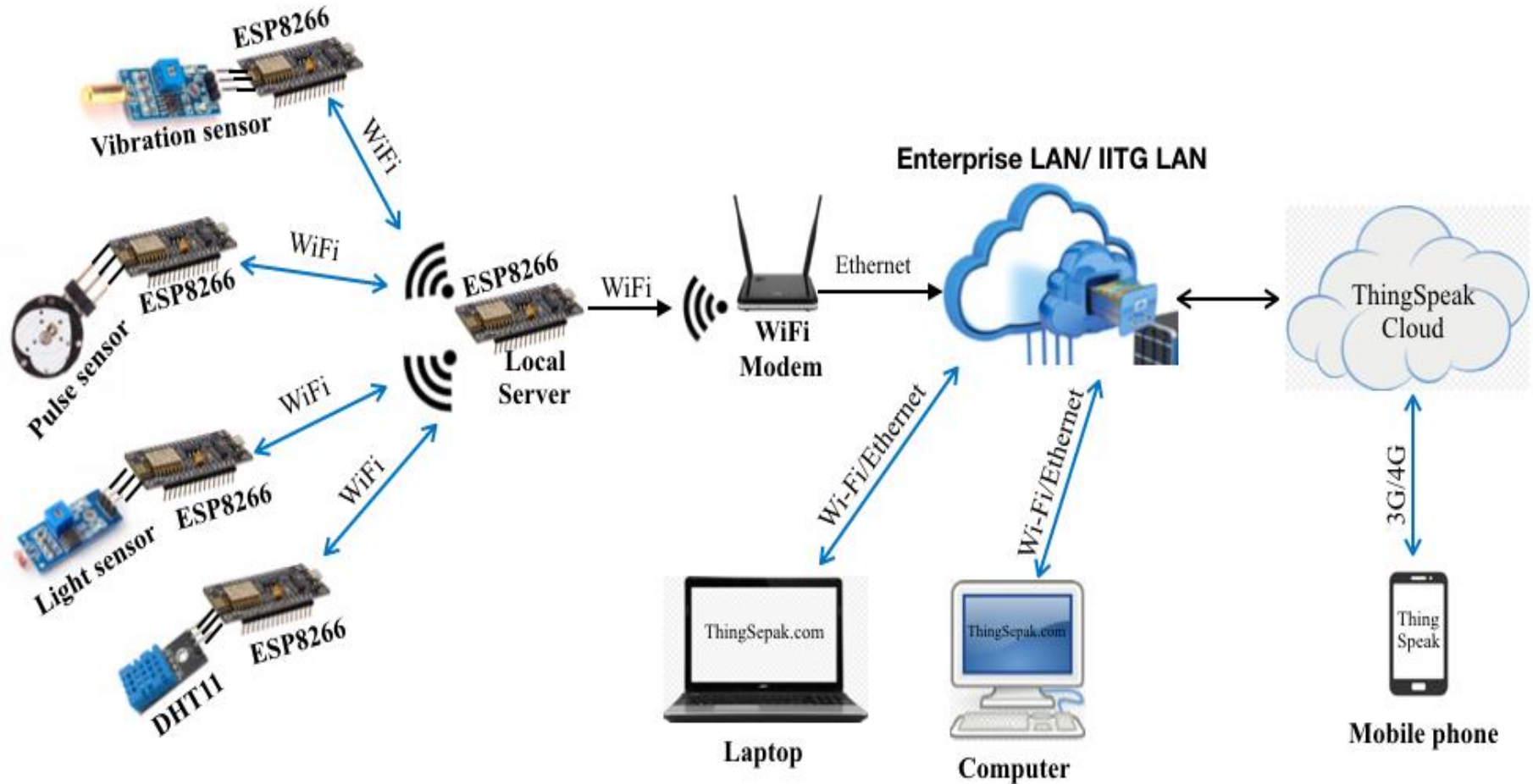
Assistant Professor

Dept. of CSE, IIT Guwahati

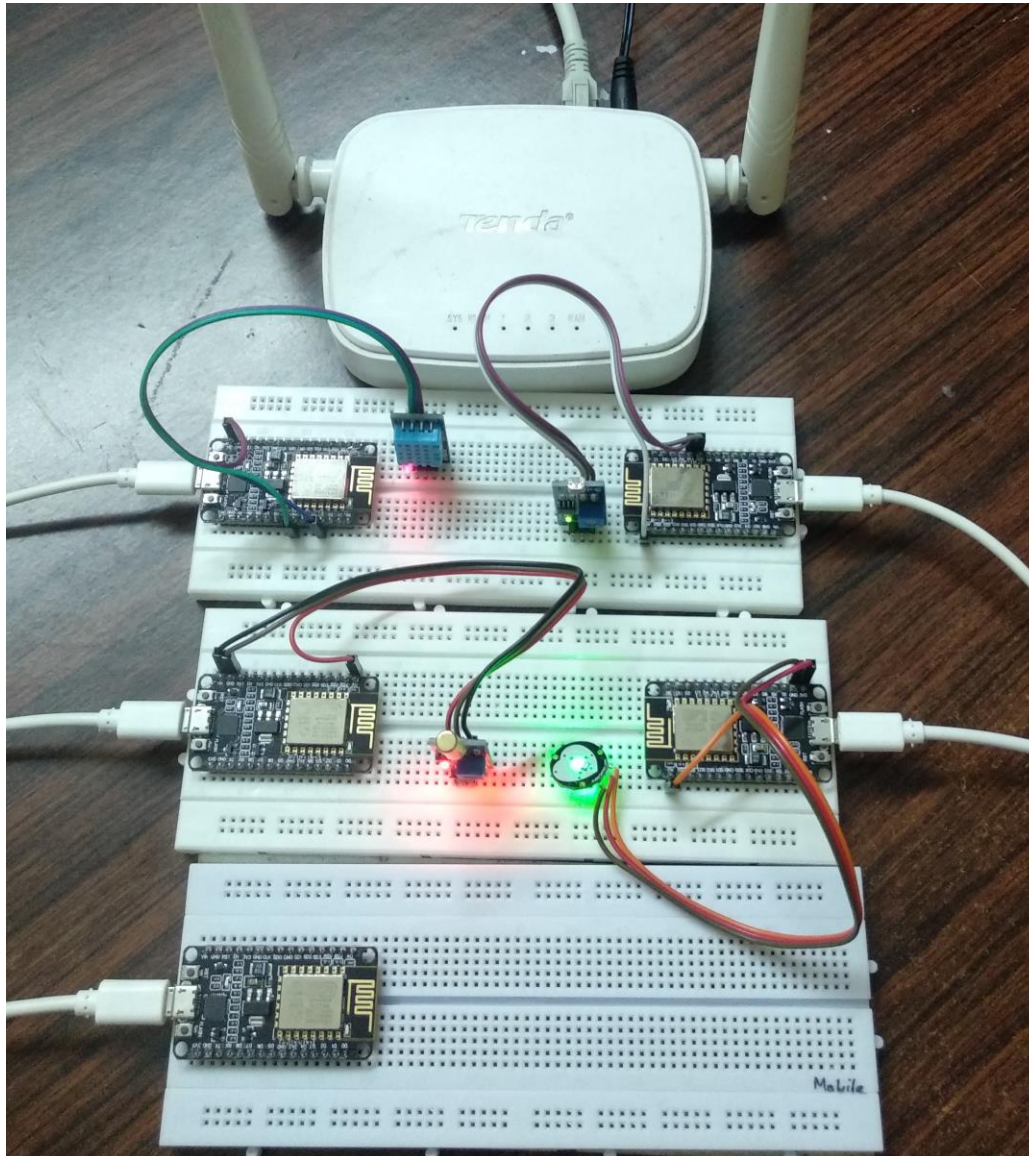
E-mail: manaskhatua@iitg.ac.in

“Try not to become a man of success. Rather become a man of value.” – Albert Einstein

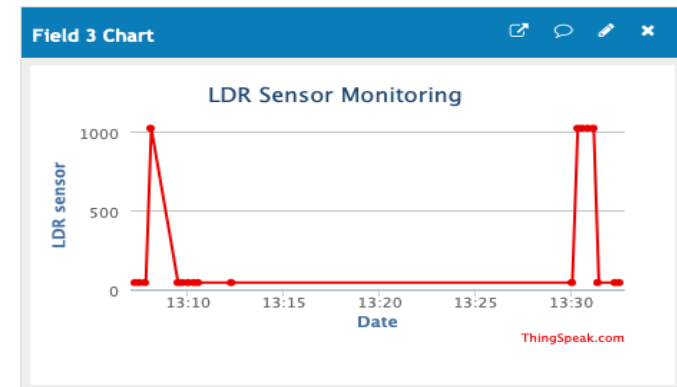
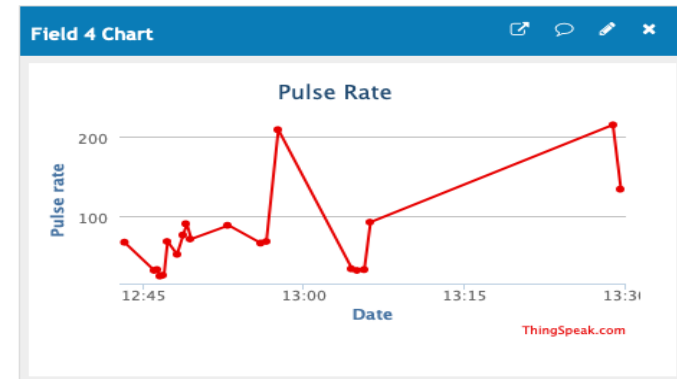
System Diagram



Physical Setup



ThingSpeak cloud server
accessing from a
Laptop/PC/Smartphone




Router Configuration To Connect with IITG Internet

Router Configuration



- This is Tenda WiFi Router
 - ESP8266 (local server) will connect to this WiFi AP
 - Sensor data will be uploaded to ThingSpeak server through this WiFi AP.
-
- Login Tenda WiFi using given IP (**192.168.0.1**) and user ID (**admin**) and password (**admin**)
 - Do the following:
 - Tenda WiFi SSID and Password under “Wireless” tab
 - **SSID**: Tenda_8060A0; **Password**: 12345678
 - Time and Date settings under “Tools” tab
 - You can change admin password under “Tools” tab
 - **Setup Internet Connection** by Advanced → Internet Connection Setup
 - Set the **Static IP, Subnet Mask, Default Gateway, DNS Server, Alternate DNS Server**
 - Reboot the router from “Tools” tab



[Home](#)[Advanced](#)[Wireless](#)[QoS](#)[Applications](#)[Security](#)[Tools](#)

[Wireless Basic Settings](#)[Wireless Security](#)[Access Control](#)[Wireless Connection Status](#)[Wireless Extender](#)

Wireless Basic Settings

Enable Wireless ☒

Primary SSID

Secondary SSID

Network Mode

SSID Broadcast ☒ Enable ☐ Disable

AP Isolation ☐ Enable ☒ Disable

Channel

Channel Bandwidth ☐ 20 ☒ 20/40

Extension Channel

WMM Capable ☒ Enable ☐ Disable

APSD Capable ☐ Enable ☒ Disable

Help

In this section you can configure the wireless settings of the router such as the SSID (name of the network) and Broadcast Channel.

SSID: This is the public name of your wireless network. It is preset to "Tenda_XXXXXX" (where "XXXXXX" represents the last six characters in device MAC address.) by default. Please change it for better security. Note that this field should not be left blank.

SSID Broadcast: This option allows you to have your network names (SSIDs) publicly broadcast or if you choose to disable it, the SSID will be hidden.

Tenda®

HomeAdvancedWirelessQoSApplicationsSecurityTools

Wireless Basic SettingsWireless SecurityAccess ControlWireless Connection StatusWireless Extender

Wireless Security Setup

Select SSIDTenda_8060A0

Security ModeWPA - PSK(Recommended)

WPA Algorithms☒ AES(Recommended) ☐ TKIP ☐ TKIP&AES

Security Key.....
Default: 12345678

To configure a wireless security key, disable the WPS below!

WPS Settings☒ Disable ☐ Enable

Reset OOB

OKCancel

Help

Here you can set the wireless password for your wireless network. You are recommended to select WPA-PSK as Security Mode and AES as WPA Algorithms Type.

WEP Key: Must be either 5 or 13 ASCII characters or 10 or 26 Hex characters.

WPA/WPA2-Personal: You can enable personal (PSK) or mixed mode, but you must make sure that the wireless client also supports the selected Security mode.

Security Key: Must be between 8~63 case-sensitive ASCII characters.

Tenda

HomeAdvancedWirelessQoSApplicationsSecurityTools

Status

Internet Connection Setup

MAC Clone

WAN Speed

LAN Settings

DNS Settings

DHCP Server

DHCP Client List

Internet Connection Setup

Internet Connection TypeStatic IP

IP Address172.16.88.50

Subnet Mask255.255.255.0

Gateway172.16.88.254

DNS Server172.17.1.1

Alternate DNS Server172.17.1.2(Optional)

MTU1500


(The default value is 1500. Do not modify it unless required by your ISP.)

OKCancel

Help

Static IP: Static IP is a connection type that allows you to specify the Static IP information provided by your ISP or that corresponds with your existing networking equipment. If you have a fixed (or static IP) address, your ISP will have provided you with the required information. Select Static IP option and type the IP Address, Subnet Mask and Gateway IP Address into the correct boxes.

Contact your ISP for help if you are not sure about which Internet connection type to use.



Home

Advanced

Wireless

QoS

Applications

Security

Tools

Status

Internet Connection Setup

MAC Clone

WAN Speed

LAN Settings

DNS Settings

DHCP Server

DHCP Client List

WAN Status

Connection Status	Connected
Internet Connection Type	Static IP
WAN IP	10.11.10.34
Subnet Mask	255.255.192.0
Gateway	10.11.0.254
DNS Server	172.17.1.1
Alternate DNS Server	172.17.1.2

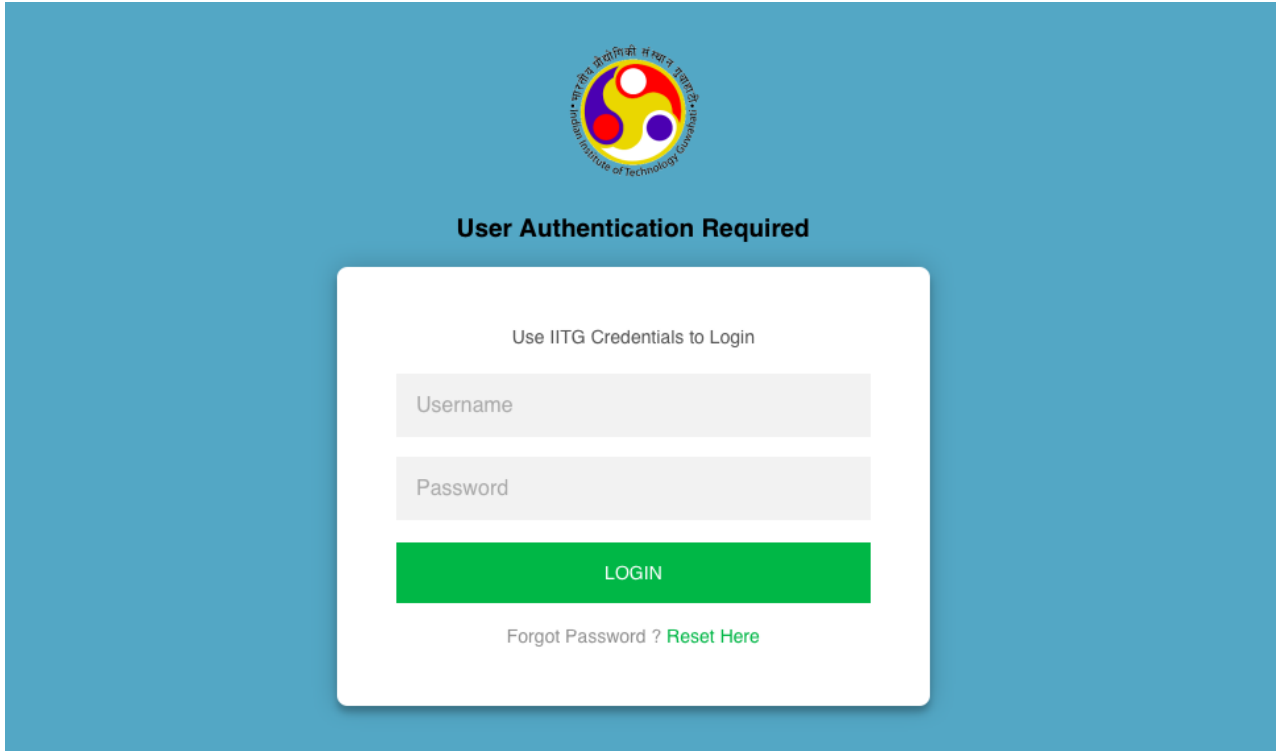
Help

Connection Status:Refers to the connection between the router and the device connected to the router's WAN.

Internet Connection Type:
This can be set in Advanced > Internet Connection Setup. DHCP and PPPoE are the most common.

Connection Time:Displays WAN connection duration

Connecting with Internet

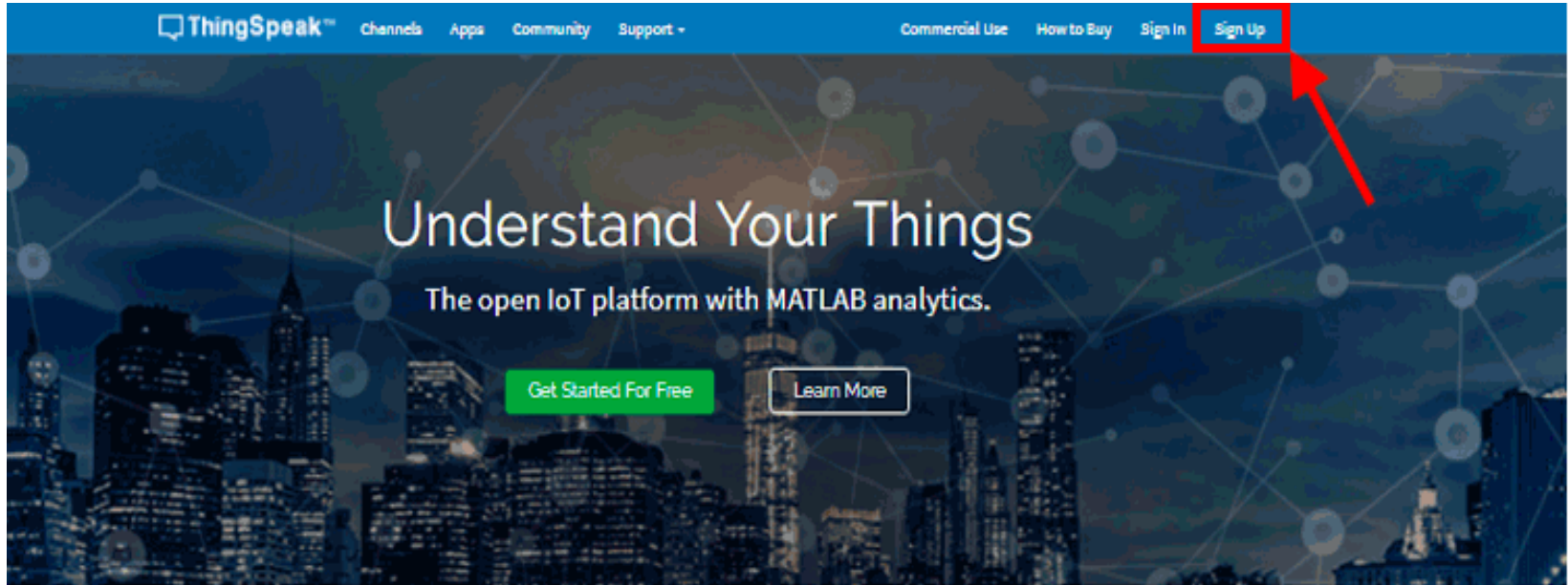


The image shows a user authentication interface on a blue background. At the top center is the IIT Guwahati logo. Below it, the text "User Authentication Required" is displayed. A white login box contains the text "Use IITG Credentials to Login". Inside the box are two input fields: "Username" and "Password". Below these fields is a green "LOGIN" button. At the bottom of the box, there is a link: "Forgot Password ? [Reset Here](#)".

- You should be able to access Internet in your Mobile/Laptop using Tenda WiFi AP

Cloud Server Configuration to Access Web Service

Configure to use Cloud Server



- We use ThingSpeak server <http://www.thingspeak.com>
- First create an user account
- Then [create a channel](#) on the ThingSpeak to upload the data

Cont...



ThingSpeak™

Channels ▾

Apps ▾

Community

Support ▾

Commercial Use

How to Buy

Account ▾

Sign Out

My Channels

New Channel

Search by tag



Name	Created	Updated
Temperature & Humidity Monitoring Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 06:44
Monitoring Four sensors in Star Topology Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 11:30
LED Control from Web Private Public Settings Sharing API Keys Data Import / Export	2019-07-12	2019-07-12 06:53

Help

Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

Click **New Channel** to create a new ThingSpeak channel.

Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.

Learn to [create channels](#), explore and transform data.

Learn more about [ThingSpeak Channels](#).

Examples

- [Arduino](#)
- [Arduino MKR1000](#)
- [ESP8266](#)
- [Raspberry Pi](#)
- [Netduino Plus](#)

Upgrade

Need to send more data faster?

Need to use ThingSpeak for a commercial project?

Channel ID 814887

Name

DEMO 2

Description

Getting different sensors data

Field 1

Temperature



Field 2

Humidity



Field 3

LDR sensor



Field 4

Pulse rate



Field 5

Vibration Sensor



Field 6



Field 7



Field 8



Metadata

Tags

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
 - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.
 - **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is -0.1275.
 - **Elevation:** Specify the elevation position meters. For example, the elevation of the city of London is 35.052.
- **Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the GitHub repository URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using [ThingSpeak Apps](#).

See [Tutorial: ThingSpeak and MATLAB](#) for an example of measuring dew point from a

Cont...



ThingSpeak™

Channels ▾

Apps ▾

Community

Support ▾

Commercial Use

How to Buy

Account ▾

Sign Out

My Channels

New Channel

Search by tag



Name	Created	Updated
Temperature & Humidity Monitoring Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 06:44
Monitoring Four sensors in Star Topology Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 11:30
LED Control from Web Private Public Settings Sharing API Keys Data Import / Export	2019-07-12	2019-07-12 06:53

Help

Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

Click **New Channel** to create a new ThingSpeak channel.

Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.

Learn to [create channels](#), explore and transform data.

Learn more about [ThingSpeak Channels](#).

Examples

- [Arduino](#)
- [Arduino MKR1000](#)
- [ESP8266](#)
- [Raspberry Pi](#)
- [Netduino Plus](#)

Upgrade

Need to send more data faster?

Need to use ThingSpeak for a commercial project?

Create Channel Display

Private View Public View Channel Settings Sharing API Keys Data Import / Export

+ Add Visualizations + Add Widgets Export recent data

Temperature Options ? x

Name Temperature

Field Field 1

Update Interval 15 second(s)

Units degree Celsius

Data Type ☐ Integer ☒ Decimal 2 (# of places)

Save Cancel

- Select **Private View** of the created channel.
- Click **Add Widgets**
- Select the Numeric Display widget, and then set the display options.

API Key and Channel ID



https://thingspeak.com/channels/819306/api_keys

ThingSpeak™ Channels Apps Community Support Commercial Use How to Buy Account Sign Out

Monitoring Four sensors in Star Topology

Channel ID: **819306**
Author: mkhatuaitg
Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key: SW7ENB9IAXJT3STP

Generate New Write API Key

Read API Keys

Key: RY4XYJC0E1S542G2

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

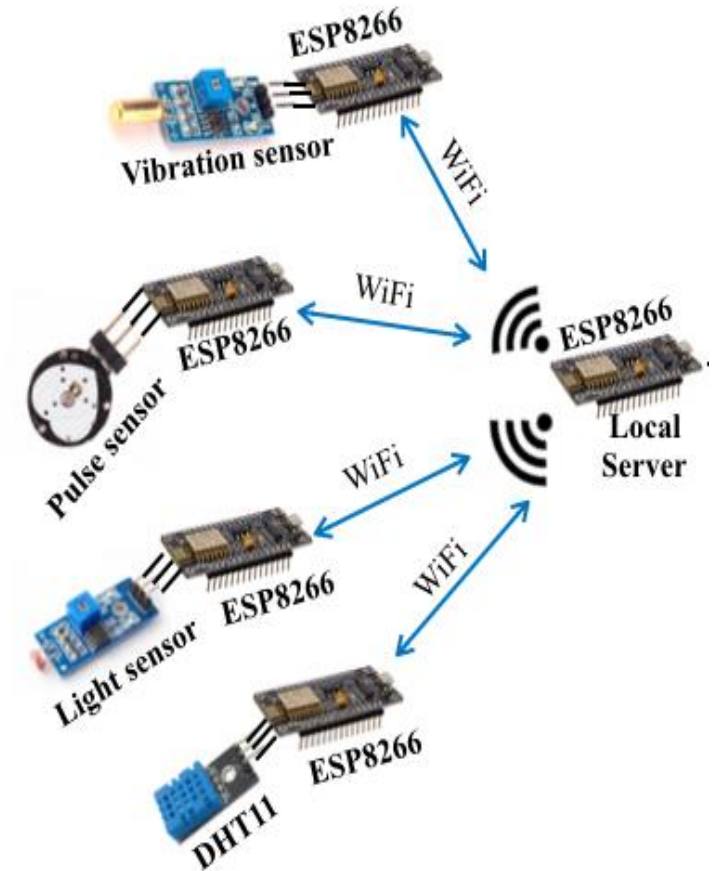
API Requests

- To send data to ThingSpeak, we need unique **API key** and **Channel ID**, which **will be used later in code** to upload the data to ThingSpeak website
- Click on “API Keys” button to get your unique “Write API Key”
- “Channel ID” is also given on the top

IoT Network Configuration

IoT Network Configuration

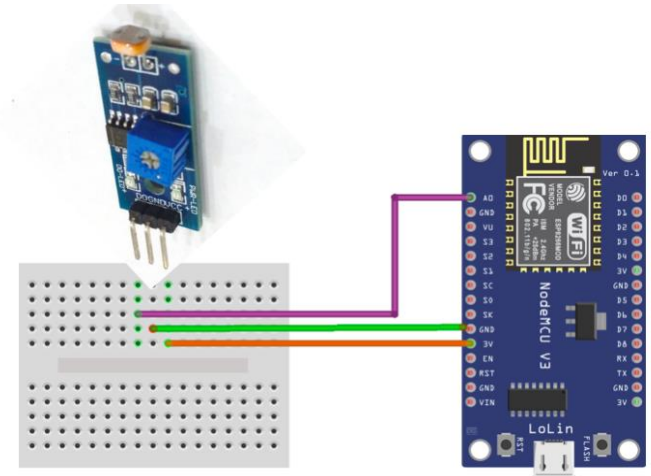
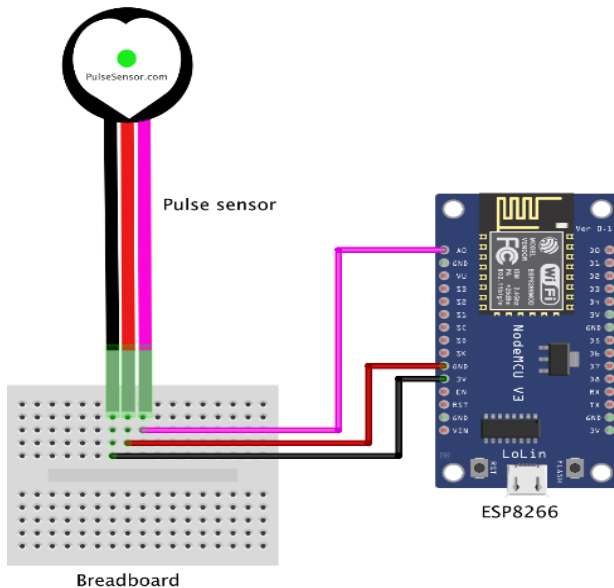
- There are total five ESP8266
 - one is acting as **server**,
 - other four as **clients** in local network.
- ESP1- ESP8266 acting as **local server**
- ESP2- ESP8266 with **Light** sensor
- ESP3- ESP8266 with **Pulse** sensor
- ESP4- ESP8266 with **vibration** sensor
- ESP5- ESP8266 with **temperature** & **humidity** sensor
- **Note:** Unique ID for each ESP will be needed in programming



Sensor Configuration

ESP8266 with LDR Sensor

- Connect VCC pin of LDR sensor with 3V3 pin of **ESP2**
- Connect GND pin of LDR sensor with GND of ESP2
- Connect DATA OUT pin of LDR sensor with A0 of ESP2.



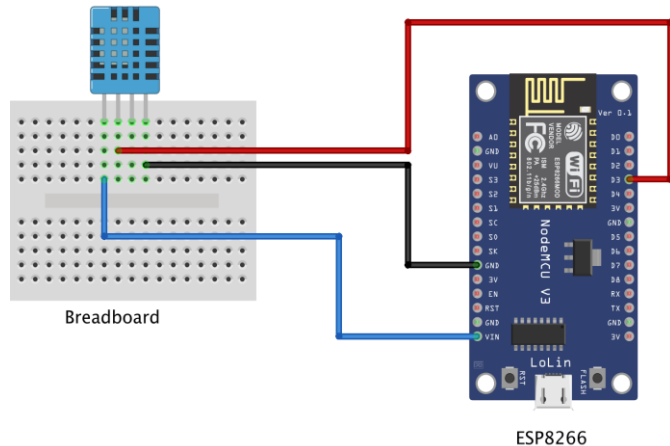
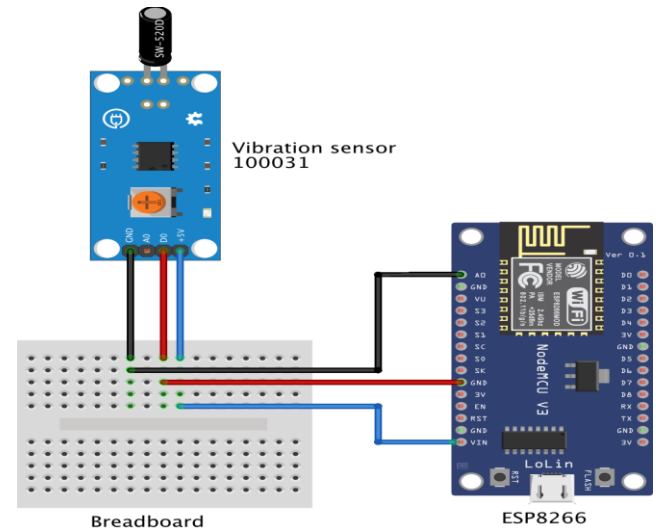
ESP8266 with **Pulse** Sensor

- Connect VCC pin of pulse sensor with 3V3 pin of **ESP3**
- Connect GND pin of sensor with GND pin of ESP3
- Connect SIGNAL pin of pulse sensor with A0 of ESP3

Cont...

ESP8266 with **Vibration** Sensor

- Connect VCC pin of vibration sensor with VIN pin of **ESP4**
- Connect GND pin of vibration sensor with ESP4 GND pin
- Connect DATA OUT pin of vibration sensor with A0 pin of ESP4



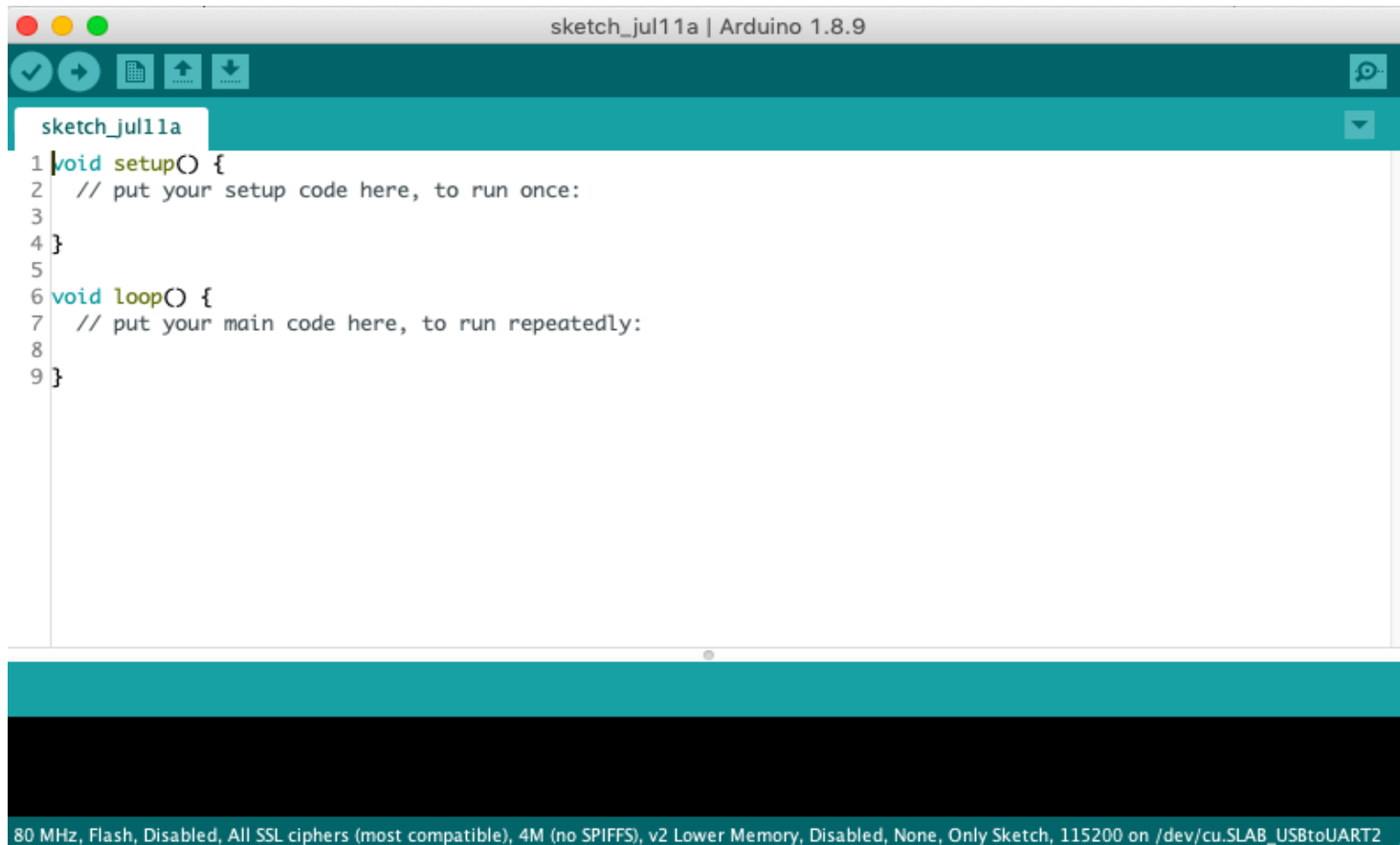
ESP8266 with **Temperature & Humidity** Sensor (DHT11)

- Connect VCC pin of DHT11 to VIN of **ESP5**
- Connect DATA OUT pin to D3 of ESP5
- Connect GND pin of DHT11 to GND of ESP5

Arduino Tool Configuration

Configure Arduino IDE

- Download and Install Arduino IDE <https://www.arduino.cc/en/Main/Software>
- When the Arduino IDE first opens, this is what you should see:

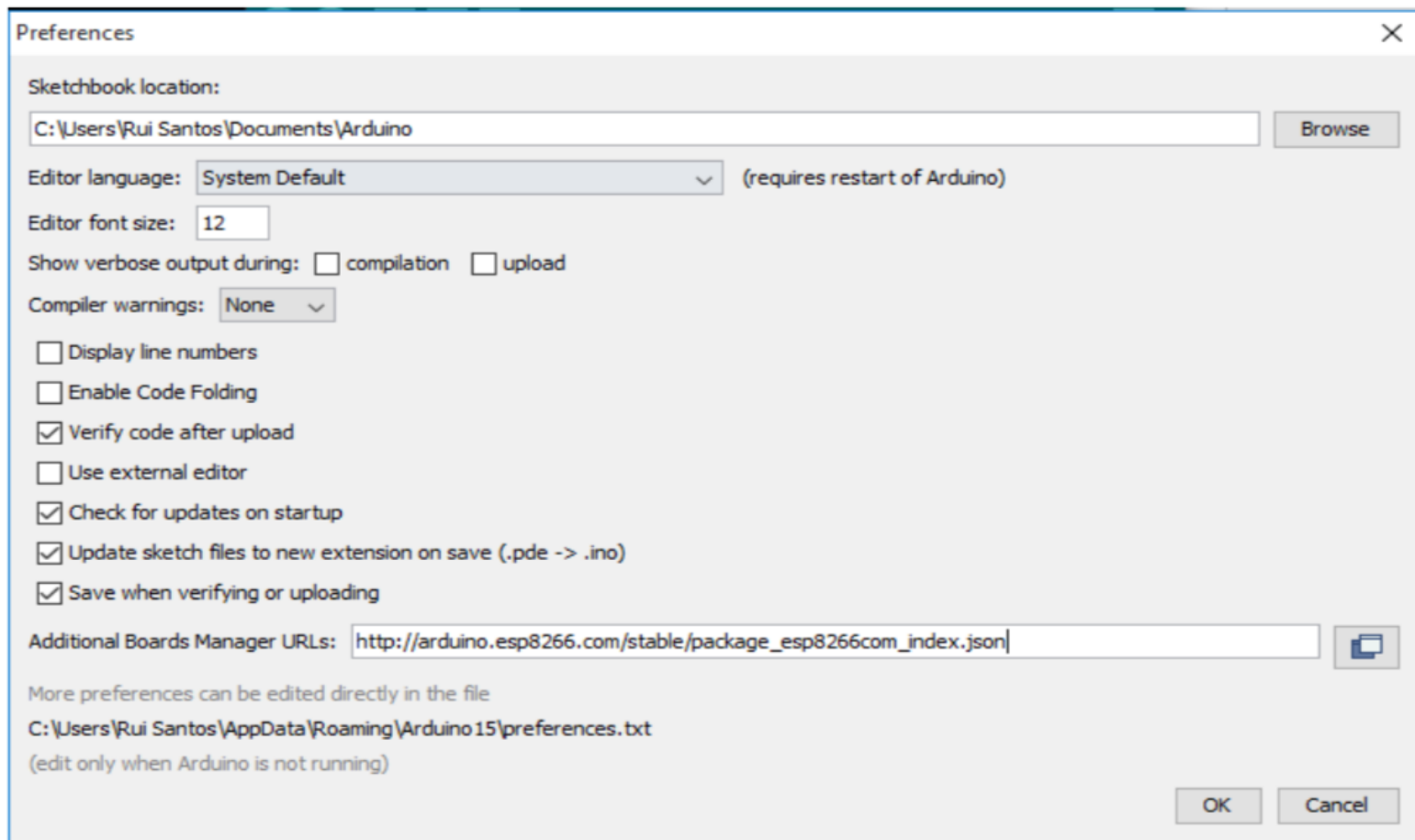


```
sketch_jul11a | Arduino 1.8.9  
✓ ↻ 📄 ⬆ ⬇  
sketch_jul11a  
1 void setup() {  
2   // put your setup code here, to run once:  
3  
4 }  
5  
6 void loop() {  
7   // put your main code here, to run repeatedly:  
8  
9 }  
  
80 MHz, Flash, Disabled, All SSL ciphers (most compatible), 4M (no SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB USBtoUART2
```

Install ESP8266 Board in IDE



- Go to **File --> Preferences**
- Enter the below URL into **Additional Board Manager URLs** field and press the “OK” button
http://arduino.esp8266.com/stable/package_esp8266com_index.json OR
https://github.com/esp8266/Arduino/releases/download/2.3.0/package_esp8266com_index.json

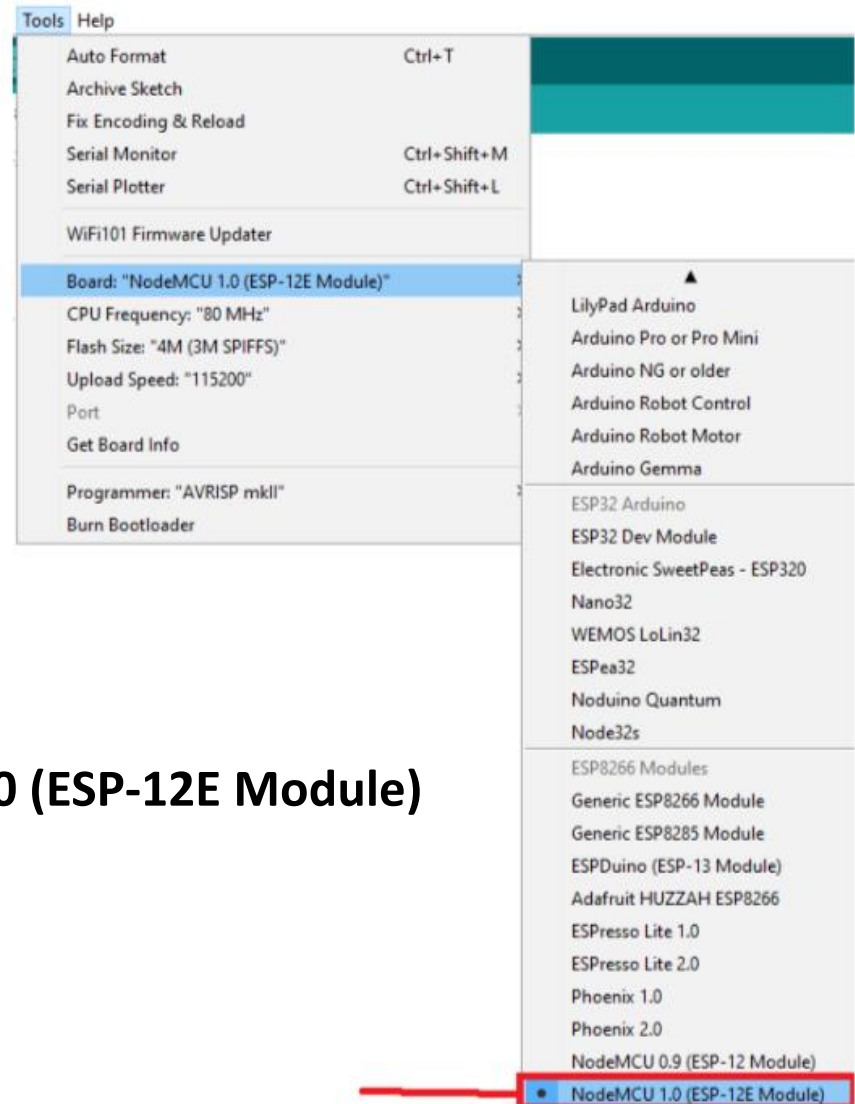


Cont...

- Go to **Tools > Board > Board Manager**
- Scroll down, select the ESP8266 board menu and **install** “**esp8266 by ESP8266 Community**”



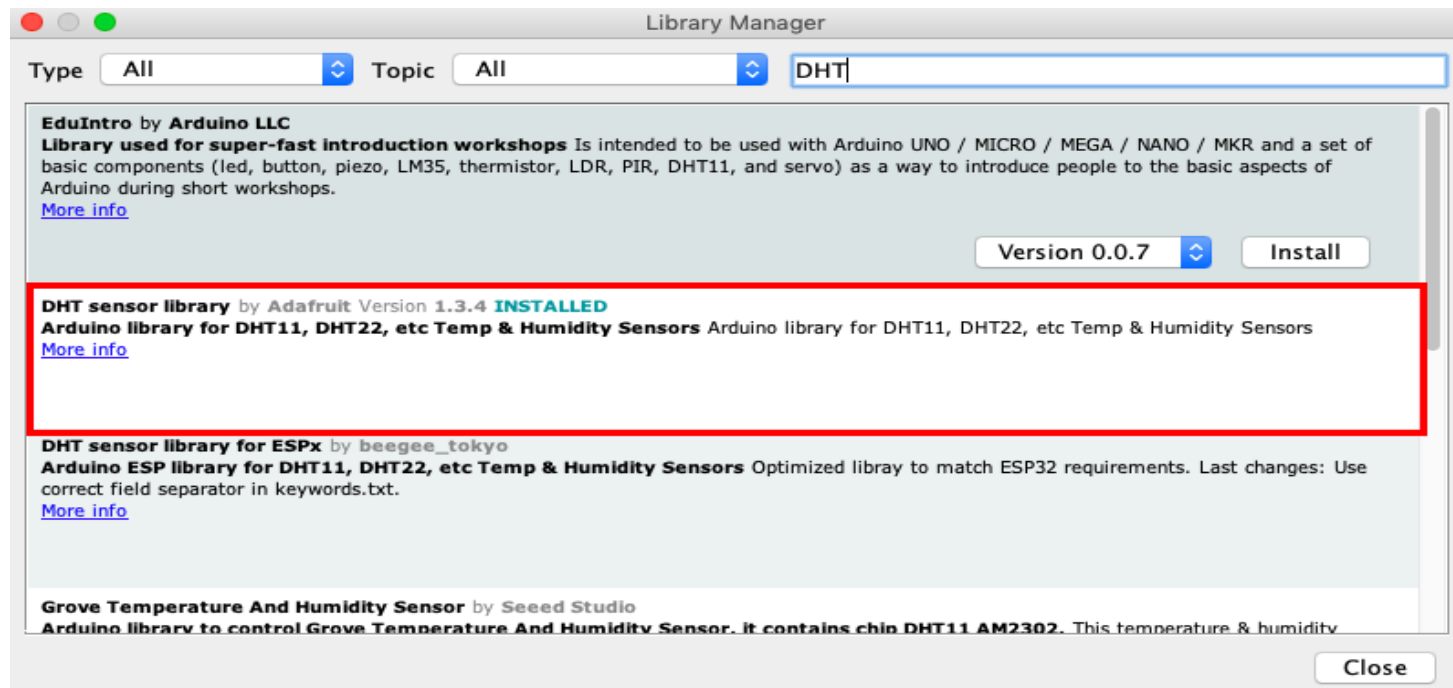
Cont...



- Select the appropriate board
 - Go to **Tools > Board > NodeMCU 1.0 (ESP-12E Module)**
- Finally, re-open the Arduino IDE

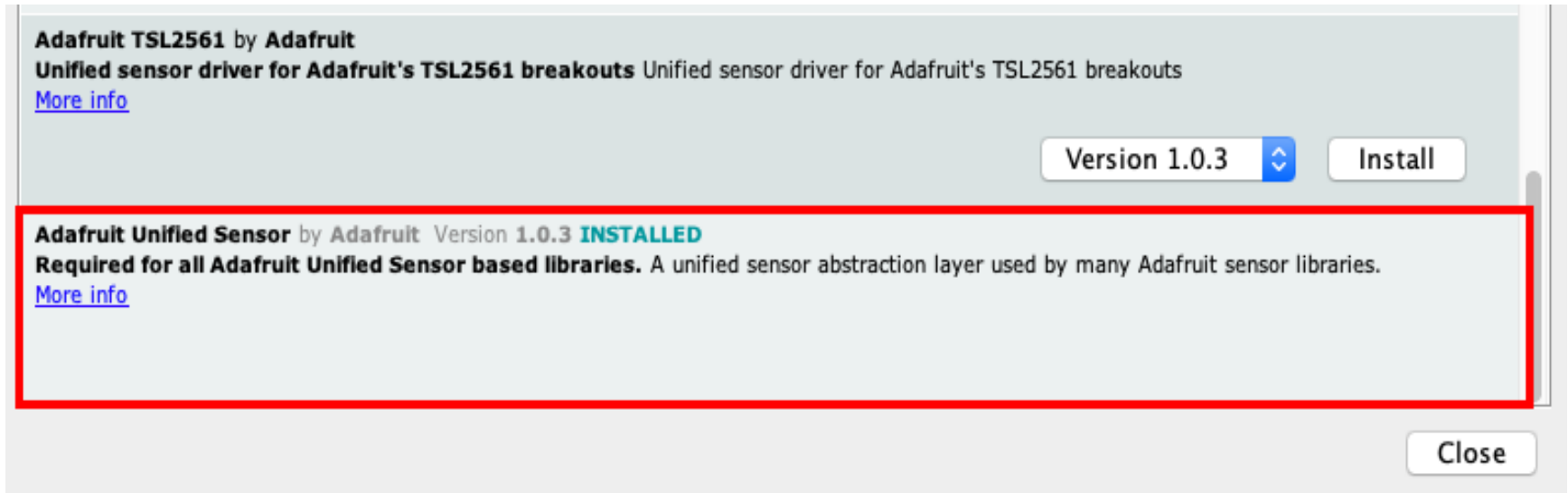
Install Sensor Libraries

- In this demo, we use **DHT11 sensor** for which we will be using **DHT.h** header file in the code. So, this header file should be **installed**.
- **Install Using the Library Manager**
 - click to **Sketch** menu then **Include Library > Manage Libraries**
 - Search for “**DHT**” on the Search box and **install** the DHT library from **Adafruit**.



Cont...

- After installing the DHT library from Adafruit, **install** “**Adafruit Unified Sensor**” libraries.



- There exist **other methods** for installing libraries
 - **Importing a .zip Library**
 - Sketch --> Include Library --> Add .Zip Library
 - **Manual Installation of Library**

MCU Programming

ESP8266 with Local Server



For ESP5, write the following code in the Arduino IDE and save as **Local_Server_ESP1.ino**
Install **ThingSpeak.h** library. Change the **red colored text** in code according to your setup.

```
#include <ESP8266WiFi.h>           //Including ESP8266 library
#include<ESP8266WebServer.h>       //Including ESP8266WebServer library for web server
#include<ThingSpeak.h>             //Including ThingSpeak library

IPAddress IP(192,168,4,15);        //Static IP address of local server
IPAddress gateway(192,168,4,1);    //Gateway of the network
IPAddress mask(255, 255, 255, 0);  //Subnet mask of the network
WiFiClient client;
WiFiServer server(80);

unsigned long myChannelNumber = 819306; //Replace with channelID of ThingSpeak channel ID
const char * myWriteAPIKey = "SW7ENB9IAXJT3STP"; //Replace WriteAPIKey of channel

const char* softAPssid = "ESP1_Server"; //SSID of the hotspot of ESP8266 acting as local server
const char* password = "12345678";      //Password of the hotspot of ESP8266 acting as local server

const char* wifissid = "Tenda_8060A0";  //Replace with SSID of WIFI router providing internet access
const char* pass = "12345678";          //Password of WIFI router providing internet access
```

Cont...

```
void setup() {  
  WiFi.mode(WIFI_AP_STA);           //station mode and access point mode both at the same time  
  Serial.begin(9600);                //Serial communication at baud rate of 9600 for debugging purpose  
  delay(100);  
  Serial.println(WiFi.getMode());  
  Serial.print("Configuring SoftAP....");  
  Serial.println(WiFi.softAPConfig(IP, gateway, mask)? "Ready" : "Failed");  
  delay(10);  
  Serial.println("Setting SoftAP...");  
  Serial.println(WiFi.softAP(softAPssid, password));  
  delay(10);  
  Serial.println(WiFi.softAPIP());  
  delay(500);  
  WiFi.begin(wifissid, pass);  
  while(WiFi.status() != WL_CONNECTED) {  
    Serial.print(".");  
    delay(500);  
  }  
  Serial.print("Connected to Wifi with ssid ");  
  Serial.println(wifissid);  
  Serial.print("WiFi IP address: ");  
  Serial.println(WiFi.localIP());    // WIFI router IP address  
  ThingSpeak.begin(client);  
  server.begin();                    //Start local server  
}
```

- Two functions exist in the programme: `setup ()` and `loop ()`
 - **setup():** This function runs once when ESP first boots
 - **loop():** This function reads the LDR sensor value and connects to local server then send sends data to local server

Cont...



```
void loop() {
  Serial.printf("Stations connected = %d\n", WiFi.softAPgetStationNum());
  WiFiClient client = server.available();           //Waiting for the incoming data if client is ready to send
  if (!client) {return;}
  String select_fun = client.readStringUntil('\r');           //Reads the ESP8266 ID (of clients)

  if(select_fun=="5") {           //If ESP5 sends the data
    String temp = client.readStringUntil('\r');           //Reads the temperature value
    String Humidity = client.readStringUntil('\r');           //Reads the humidity value
    //Upload the temp value to ThingSpeak server as first field of channel

    ThingSpeak.writeField(myChannelNumber, 1, temp, myWriteAPIKey);
    delay(15000);           //Wait for 15 sec after one entry
    //Upload the humidity value to ThingSpeak server as second field of channel

    ThingSpeak.writeField(myChannelNumber, 2, Humidity, myWriteAPIKey);
    Serial.print("Temperature: ");
    Serial.print(temp);
    Serial.print(" degree celsius, Humidity: ");
    Serial.print(Humidity);
    Serial.print("%. ");
    Serial.println("Sent to ThingSpeak Server...");
  }
}
```


Cont...



```
if(select_fun=="2") {                                //If ESP2 sends the data
    String LDRval = client.readStringUntil('\r');      //Reads light sensor value
                //Upload the light sensor value to ThingSpeak server as third field of channel
    ThingSpeak.writeField(myChannelNumber, 3, LDRval, myWriteAPIKey);
    Serial.print("LDR sensor data value: ");
    Serial.println(LDRval);
    Serial.println("Sent to ThingSpeak Server...");
}
if(select_fun=="3") {                                //If ESP3 sends the data
    String pulseRate = client.readStringUntil('\r');  //Reads pulse rate
                //Upload the pulse rate to ThingSpeak server as fourth field of channel
    ThingSpeak.writeField(myChannelNumber, 4, pulseRate, myWriteAPIKey);
    Serial.print("Pulse rate: ");
    Serial.print(pulseRate);
    Serial.println(" BPM. Sent to ThingSpeak Server..");
}
if(select_fun=="4"){                                //If ESP4 sends the data
    String Vibval = client.readStringUntil('\r');      //Reads vibration sensor data
                //Upload the vibration sensor data value to ThingSpeak server as fifth field of channel
    ThingSpeak.writeField(myChannelNumber, 5, Vibval, myWriteAPIKey);
    Serial.print("Vibration Sensor data: ");
    Serial.print(Vibval);
    Serial.println(" Sent to ThingSpeak server..");
}
delay(15000);    //waits for 15 secs after each transmission
}
```

ESP8266 with LDR Sensor



For **ESP2**, write the following code in the Arduino IDE and save as **LDR_client.ino**

```
#include<ESP8266WiFi.h>           // Including ESP8266 library

char ssid[]="ESP1_Server";        //Network ssid of hotspot of local server
char pass[]="12345678";           // Password of hotspot of local server
int val;
int LDRpin = A0;                  //LDR Pin Connected to A0 pin
IPAddress server(192,168,4,15);    // IP address of local server
WiFiClient client;
```

- Change the IP address of Local Server (i.e. ESP1)
- Change the SSID and Password of WiFi AP hosted in Local Server
- Two functions exist in the programme: setup () and loop ()
 - **setup():** This function runs once when ESP first boots
 - **loop():** This function reads the LDR sensor value and connects to local server then send sends data to local server

Cont...



void setup()

```
{
  Serial.begin(9600);           // Serial communication at baud rate of 9600 for debugging purpose
  delay(10);
  WiFi.mode(WIFI_STA);        // ESP8266 in station mode
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);
  Serial.println();
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.println("WiFi connected");
  Serial.print("LocalIP:"); Serial.println(WiFi.localIP());
  Serial.println("MAC:" + WiFi.macAddress());
  Serial.print("Gateway:"); Serial.println(WiFi.gatewayIP());
  Serial.print("AP MAC:"); Serial.println(WiFi.BSSIDstr());           // MAC address of access point
}
```

Cont...



```
void loop()
```

```
{
```

```
  val = analogRead(LDRpin);
```

```
// Reads the light sensor value
```

```
  if(client.connect(server,80))
```

```
// Connect to local server
```

```
  {
```

```
    client.print("2\r");
```

```
// before sending data first send ESP8266 ID as 2
```

```
    Serial.print("LDR sensor value: ");
```

```
    Serial.println(val);
```

```
    String LDRval = String(val);
```

```
    LDRval += "\r";
```

```
// Add end delimiter
```

```
    client.print(LDRval);
```

```
// Send to local server
```

```
    Serial.println("Sent to Local Server..");
```

```
    delay(15000);
```

```
  }
```

```
  client.stop();
```

```
}
```

ESP8266 with Pulse Sensor



```
#define pulsePin A0          // Pulse sensor input pin A0
#include<ESP8266WiFi.h>      // Including ESP8266 library

char ssid[] = "ESP1_Server"; // Replace with SSID of hotspot of local server
char pass[] = "12345678";    // Replace with password of hotspot of local server
IPAddress server(192,168,4,15); // IP address of local server
WiFiClient client;

int rate[10];                // array to hold last ten IBI value
unsigned long sampleCounter = 0; // used to determine pulse timing
unsigned long lastBeatTime = 0; // used to find IBI
unsigned long lastTime = 0, N;

int BPM = 0;                 // int that holds raw analog in 0. updated every 2mS
int IBI = 0;                 // int that holds time interval between beats! Must be seeded!
int P = 512;                 // used to find peak in pulse wave, seeded
int T = 512;                 // used to find trough in pulse wave, seeded
int thresh = 512;            // used to find instant moment of heart beat, seeded
int amp = 100;               // used to hold amplitude of pulse waveform, seeded
int Signal;                  // holds incoming raw data
boolean Pulse = false;       // "True" when heartbeat is detected. "False" when not a "live beat".
boolean firstBeat = true;    // used to seed rate array so we startup with reasonable BPM
boolean secondBeat = true;   // used to seed rate array so we startup with reasonable BPM
boolean QS = false;          // Becomes true when ESP8266 finds a beat
```

For **ESP3**, write the following code in the Arduino IDE and save as **Pulse_client.ino**

Cont...

void setup()

```
{  
  Serial.begin(9600);           // Serial communication at baud rate of 9600 for debugging purpose  
  delay(10);  
  WiFi.mode(WIFI_STA);         // ESP8266 in station mode  
  Serial.print("Connecting to ");  
  Serial.println(ssid);  
  WiFi.begin(ssid, pass);  
  Serial.println();  
  while (WiFi.status() != WL_CONNECTED)  
  {  
    Serial.print(".");  
    delay(500);  
  }  
  Serial.println();  
  Serial.println("WiFi connected");  
  Serial.print("LocalIP:"); Serial.println(WiFi.localIP());  
  Serial.println("MAC:" + WiFi.macAddress());  
  Serial.print("Gateway:"); Serial.println(WiFi.gatewayIP());  
  Serial.print("AP MAC:"); Serial.println(WiFi.BSSIDstr()); // MAC address of access point  
}
```

Cont...

```
void loop(){
  if (QS == true){
    if (client.connect(server, 80)){
      client.print("3\r");
      String pulseRate = String(BPM);
      pulseRate += "\r";
      Serial.print("Pulse rate: ");
      Serial.print(BPM);
      Serial.println(" BPM.");
      client.print(pulseRate);
      Serial.println("Sent to local server..");
    }
    QS = false;
    client.stop();
    delay(15000);
  }
  else if(millis() >= (lastTime + 2)) {
    readPulse();
    lastTime = millis();
  }
}
```

//if ESP8266 finds a beat
// Connect to local server
// before sending data first send ESP8266 ID as 3
// Convert into string
// Add "r" as end delimiter

// send data to local server

Cont...

```
void readPulse() {
    Signal = analogRead(pulsePin);           //Read pulse sensor value
    sampleCounter += 2;                       // Keeps track of the time in mS
    int N = sampleCounter - lastBeatTime;    // Monitor the time since the last beat to avoid noise
    detectSetHighLow();                      // find the peak and trough of the pulse wave
                                           // Now it's time to look for the heart beat
                                           // signal surges up in value every time there is a pulse
    if(N > 250){                             // avoid high frequency noise
        if((Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3))
            pulseDetected();
    }
    if (Signal < thresh && Pulse == true) {
        Pulse = false;
        amp = P - T;
        thresh = amp / 2 + T;
        P = thresh;
        T = thresh;
    }
    if (N > 2500) {
        thresh = 512;
        P = 512;
        T = 512;
        lastBeatTime = sampleCounter;
        firstBeat = true;
        secondBeat = true;
    }
}

void detectSetHighLow() {
    if(Signal < thresh && N > (IBI/5)* 3)
        // avoid dichrotic noise by waiting 3/5 of last IBI
    {
        if (Signal < T) {                   // T is the trough
            T = Signal;                     // Keep track of lowest point in pulse wave
        }
    }
    if (Signal > thresh && Signal > P) // thresh condition helps avoid noise
    {
        P = Signal;                        // P is the peak
    }                                     // Keep track of highest point in pulse wave
}
```


Cont...

void pulseDetected()

```
{
    Pulse = true;          // set the pulse flag when there is a pulse
    IBI = sampleCounter - lastBeatTime; // time between beats in mS
    lastBeatTime = sampleCounter; //keep track of time for next pulse
    if (firstBeat)          // if it's the first time beat is found
    {
        firstBeat = false;    //clear firstBeat flag
        return;
    }
    if (secondBeat)          // if this is second beat
    {
        secondBeat = false; // clear secondBeat flag
        for (int i = 0; i <= 9; i++)
        {
            rate[i] = IBI;
        }
    }
    word runningTotal = 0; // clear the runningTotal variable
    for (int i = 0; i <= 8; i++) //Shift data in the rate array
    {
        rate[i] = rate[i + 1]; // and drop the oldest IBI value
        runningTotal += rate[i]; // add up the 9 oldest IBI value
    }
    rate[9] = IBI;          // add the latest IBI to the rate array
    runningTotal += rate[9]; //add the latest IBI to runningTotal
    runningTotal /= 10;     // average the last 10 IBI values
```

```
BPM = 60000 / runningTotal;
// how many beats can fit into a minute? that's BPM!
QS = true;
if (client.connect(server, 80)) //Connects to local server
{
    client.print("3\r");
    //before sending the data sends ESP8266 ID as 3
    String pulseRate = String(BPM);
    // Converting integer data into string
    pulseRate += "\r";
    // Add end Delimiter "r" in the data
    Serial.print("Pulse rate: ");
    Serial.print(BPM);
    Serial.println(" BPM.");
    client.print(pulseRate); //sends data to locals server
    Serial.println("Sent to local server..");
}
client.stop();
delay(15000);
// Wait for 15 seconds after each transmission
}
```

ESP8266 with Vibration Sensor



For **ESP4**, write the following code in the Arduino IDE and save as **Vibration_client.ino**

```
#include <ESP8266WiFi.h>           // Including ESP8266 library
#define vib A0                     // sensor input from A0 pin of ESP8266

char ssid[] = "ESP1_Server";       //Replace with SSID of hotspot of local server
char pass[] = "12345678";         // Replace with password of hotspot of local server

IPAddress server(192,168,4,15);    // IP address of local server
WiFiClient client;
```

- Change the **IP address** of Local Server (i.e. ESP1)
- Change the **SSID** and **Password** of WiFi AP hosted in Local Server

Cont...



```
void setup(){
  Serial.begin(9600);           // Serial communication at baud rate of 9600 for debugging purpose
  delay(10);
  pinMode(vib, INPUT);         // Input of vibration sensor
  WiFi.mode(WIFI_STA);         // ESP8266 as station mode
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);
  Serial.println();
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.println("WiFi connected");
  Serial.print("LocalIP:"); Serial.println(WiFi.localIP()); // IP address of local server
  Serial.println("MAC:" + WiFi.macAddress());
  Serial.print("Gateway:"); Serial.println(WiFi.gatewayIP());
  Serial.print("AP MAC:"); Serial.println(WiFi.BSSIDstr()); // MAC address of access point
}
```

Cont...



```
void loop(){
  int val = analogRead(vib);           // Reads the sensor value
  if(client.connect(server,80))        //connects to local server
  {
    client.print("4\r");                // Before sending the data sends ESP8266 ID as 4
    Serial.print("Vibration sensor value: ");
    Serial.println(val);
    String data = String(val);          // Converting integer data into string type
    data += "\r";                       // Add end delimiter "r" in the data
    client.print(data);                 // sends sensor data to local server
    Serial.println("Sent to Local server..!! ");
    delay(15000);                       // After each transmission wait for 15 seconds
    client.stop();
  }
}
```

ESP8266 with DHT11 Sensor



For **ESP5**, write the following code in the Arduino IDE and save as **Temp_Humidity_Client.ino**

```
#include<DHT.h>                //Including temperature and Humidity sensor library
#include<ESP8266WiFi.h>          //Including ESP8266 library
#define DHTPIN 0                // D3 pin of ESP8266

char ssid[] = "ESP1_Server";    //Replace with ssid of hotspot of local server
char pass[] = "12345678";       // Replace with password of hotspot of local server

IPAddress server(192,168,4,15); // Static IP address of local server. Replace whatever you want.
WiFiClient client;

DHT dht(DHTPIN, DHT11);        // Data of DHT11 sensor in D3 pin of ESP8266
```

- Change the IP address of Local Server (i.e. ESP1)
- Change the SSID and Password of WiFi AP hosted in Local Server
- Install the **DHT11** library and **Adafruit Unified Sensor** library for DHT11 sensor

Cont...



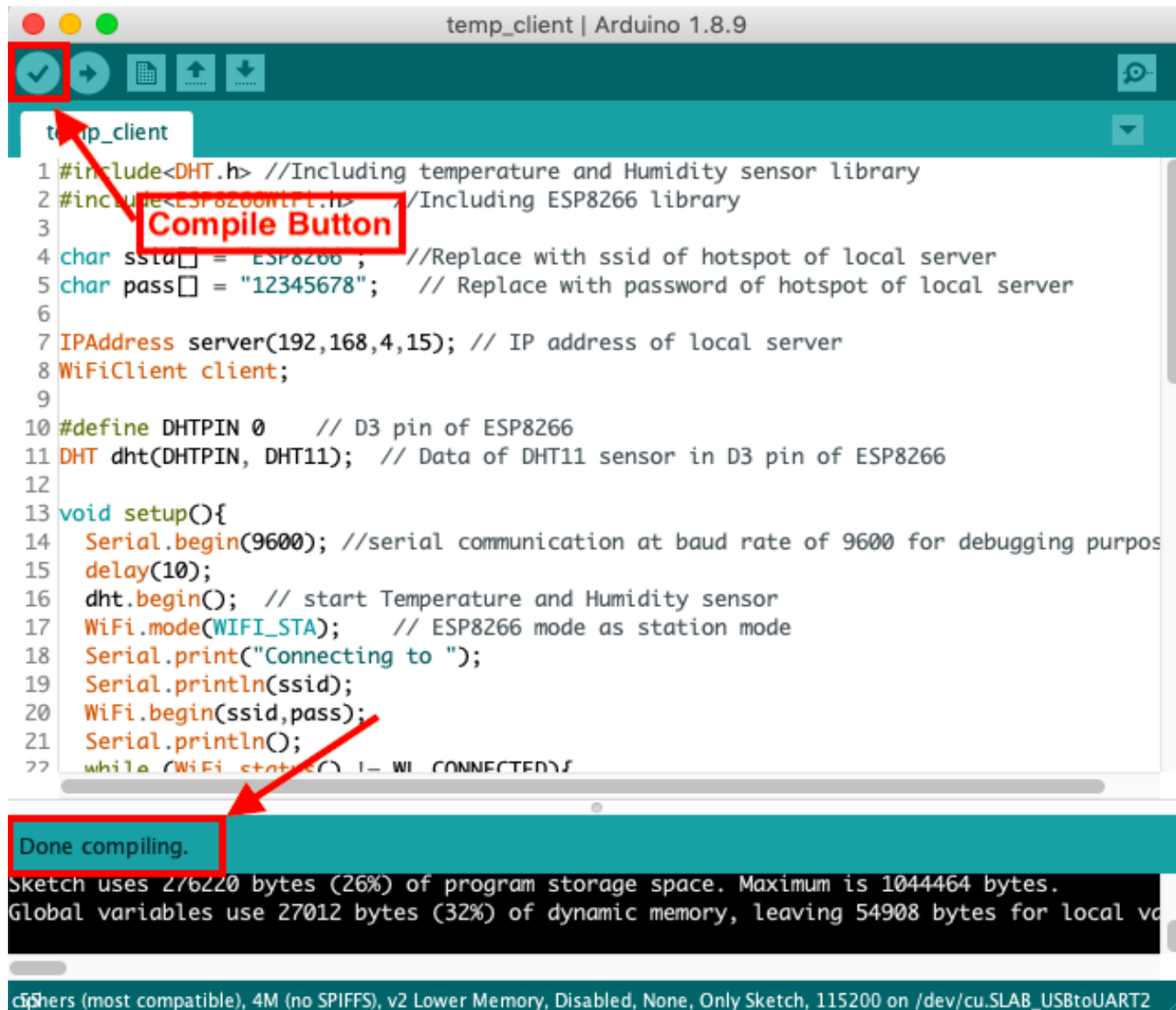
```
void setup() {  
    Serial.begin(9600);           //serial communication at baud rate of 9600 for debugging purpose  
    delay(10);  
    dht.begin();                 // start Temperature and Humidity sensor  
    WiFi.mode(WIFI_STA);        // ESP8266 mode as station mode  
    Serial.print("Connecting to ");  
    Serial.println(ssid);  
    WiFi.begin(ssid, pass);  
    Serial.println();  
    while (WiFi.status() != WL_CONNECTED) {  
        Serial.print(".");  
        delay(500);  
    }  
    Serial.println();  
    Serial.println("WiFi connected");  
    Serial.print("LocalIP:"); Serial.println(WiFi.localIP());  
    Serial.println("MAC:" + WiFi.macAddress());  
    Serial.print("Gateway:"); Serial.println(WiFi.gatewayIP());  
    Serial.print("AP MAC:"); Serial.println(WiFi.BSSIDstr()); // MAC address of access point  
}
```

Cont...

```
void loop() {  
    float h = dht.readHumidity();           // Read Humidity value from sensor  
    float t = dht.readTemperature();        // Read temp value from sensor  
    if(isnan(h) || isnan(t)) {  
        Serial.println("Failed to read from DHT sensor");           // Error message  
        return;  
    }  
    if(client.connect(server,80))           // Connect to local server  
    {  
        client.print("5\r");                // before sending the data first send ESP8266 ID as 5  
        String temp = String(t);  
        temp += "\r";                       // Add "r" as end delimiter  
        client.print(temp);                 // send temperature to local server  
        Serial.print("Temperature: ");  
        Serial.print(t);  
        Serial.print(" degree celsius, Humidity: ");  
        Serial.print(h);  
        Serial.print("%. ");  
        String humidity = String(h);  
        humidity += "\r";                   // Add "r" in data as end delimiter  
        client.print(humidity);             // send to Local server  
        Serial.println("Sent to local server ");  
        delay(15000);                       // delay of 15sec after each transmission  
    }  
    client.stop();  
}
```

Code Compilation and Upload

Code Compilation



```
temp_client | Arduino 1.8.9

temp_client

1 #include<DHT.h> //Including temperature and Humidity sensor library
2 #include<ESP8266WiFi.h> //Including ESP8266 library
3
4 char ssid[] = "ESP8266"; //Replace with ssid of hotspot of local server
5 char pass[] = "12345678"; // Replace with password of hotspot of local server
6
7 IPAddress server(192,168,4,15); // IP address of local server
8 WiFiClient client;
9
10 #define DHTPIN 0 // D3 pin of ESP8266
11 DHT dht(DHTPIN, DHT11); // Data of DHT11 sensor in D3 pin of ESP8266
12
13 void setup(){
14   Serial.begin(9600); //serial communication at baud rate of 9600 for debugging purpos
15   delay(10);
16   dht.begin(); // start Temperature and Humidity sensor
17   WiFi.mode(WIFI_STA); // ESP8266 mode as station mode
18   Serial.print("Connecting to ");
19   Serial.println(ssid);
20   WiFi.begin(ssid,pass);
21   Serial.println();
22   while (WiFi.status() != WL_CONNECTED){
23
24   }
25 }
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

Done compiling.

Sketch uses 276220 bytes (26%) of program storage space. Maximum is 1044464 bytes.
Global variables use 27012 bytes (32%) of dynamic memory, leaving 54908 bytes for local va

Arduino Uno (most compatible), 4M (no SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB_USBtoUART2
```

Compilation
successful
message in
bottom left
corner.

Code Uploading

- Plug in the ESP8266 boards one by one to PC/Laptop via USB cable
- Go to **Tool** menu, select Board “**NodeMCU 1.0 (ESP-12E Module)**” and Port “**COM3**”.
- Open the corresponding code and do uploading code in Node MCU.

Note: If COM port is not detected automatically then it is needed to install.

Download port drivers from the given link and then install and then restart the system:

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>



```
local_server | Arduino 1.8.9

1 #include <ESP8266WiFi.h> //Including ESP8266 library
2 #include<ESP8266WebServer.h> //Including ESP8266WebServer library for web serv
3 #include<ThingSpeak.h> //Including ThingSpeak library
4
5 IPAddress IP(192,168,4,15); //Static IP address of local server
6 IPAddress gateway(192,168,4,1); //Gateway of the network
7 IPAddress mask(255, 255, 255, 0); //Subnet mask of the network
8
9 WiFiClient client;
10 WiFiServer server(80);
11
12 unsigned long myChannelNumber = 814887; //Replace with channelID of ThingSpeak
13 const char * myWriteAPIKey = "EK4LTPHWU4GGEOVP"; //Replace with WriteAPIKey of
14
15 const char* softAPssid = "ESP8266"; //SSID of the hotspot of ESP8266 acting
16 const char* password = "12345678"; //Password of the hotspot of ESP8266 act
17
18 const char* wifissid = "Tenda_8060A0"; //Replace with SSID of WIF router provi
19 const char* pass = "12345678"; //Password of WIFI router providing inte

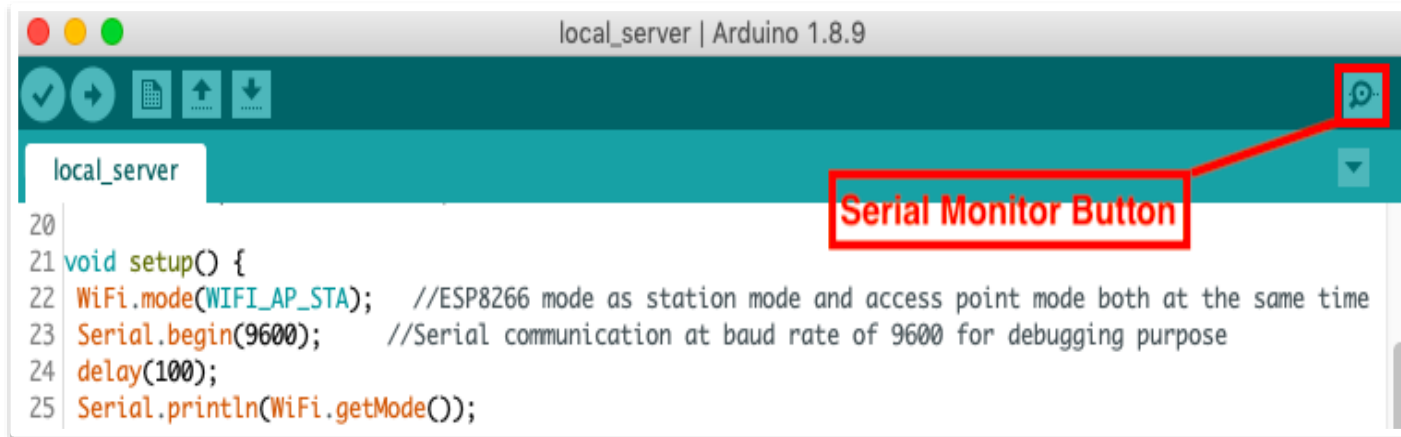
Done uploading.
TRACE +0.000 Received full packet: 01120200000000000000
Hard resetting via RTS pin...

NodeMCU 1.0 (ESP-12E Module) on /dev/cu.SLAB_USBtoUART
```

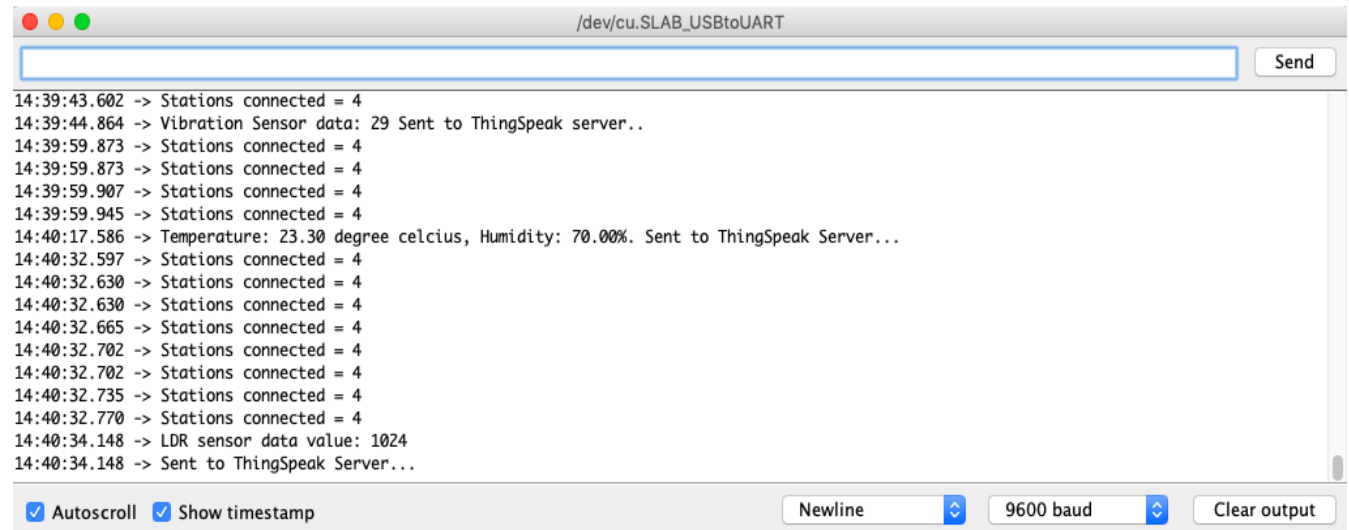
Observe Outputs

Open Serial Monitor

- First **select the port** (go to **Tools > Port:**) to which the board is connected then click the icon of **Serial Monitor** on the top right side of the Arduino IDE



Serial Monitor of Local Server



Cont...

```
13:32:04.313 -> I Kf8sConnecting to ESP8266
13:32:04.413 ->
13:32:04.413 -> .
13:32:04.912 -> WiFi connected
13:32:04.912 -> LocalIP:192.168.4.114
13:32:04.945 -> MAC:3C:71:BF:32:73:3C
13:32:04.982 -> Gateway:192.168.4.1
13:32:04.982 -> AP MAC:3E:71:BF:32:6E:AD
13:32:05.015 -> LDR sensor value: 1024
13:32:05.049 -> Sent to Local Server..
13:32:19.956 -> LDR sensor value: 36
13:32:19.956 -> Sent to Local Server..
13:32:34.940 -> LDR sensor value: 36
13:32:34.974 -> Sent to Local Server..
```

☒ Autoscroll ☒ Show timestamp

Newline 9600 baud Clear output

Serial Monitor
of ESP2

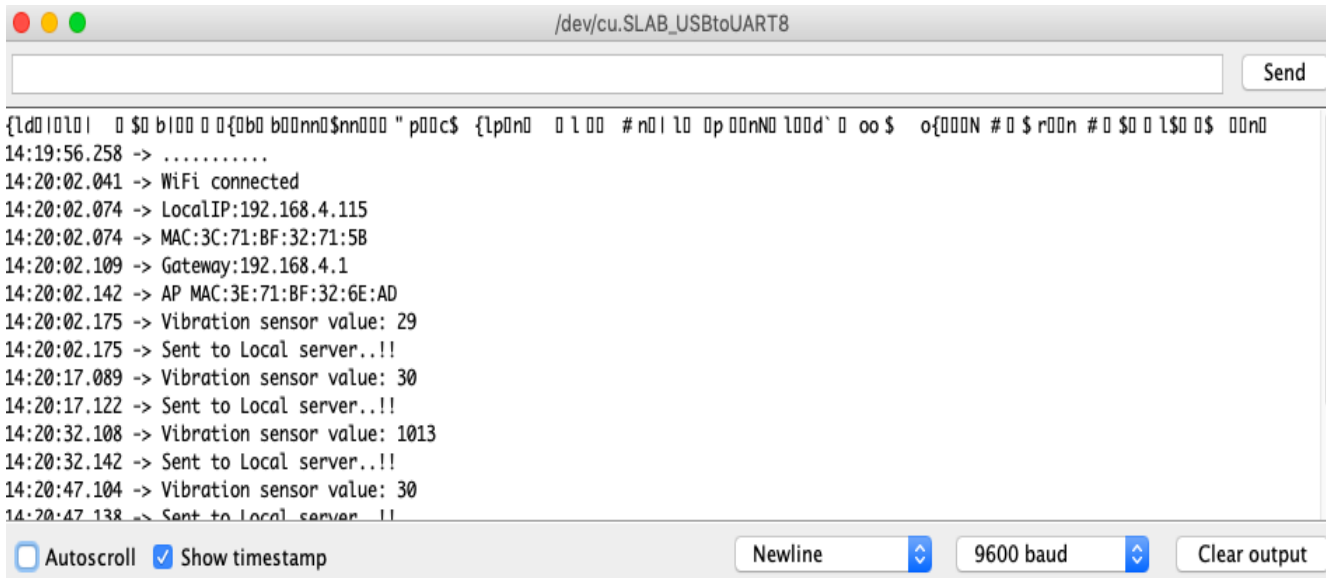
Serial Monitor
of ESP3

```
..`Hf8sConnecting to ESP8266
13:55:10.018 ->
13:55:10.018 -> .....
13:55:16.314 -> WiFi connected
13:55:16.347 -> LocalIP:192.168.4.118
13:55:16.347 -> MAC:3C:71:BF:32:44:4E
13:55:16.380 -> Gateway:192.168.4.1
13:55:16.418 -> AP MAC:3E:71:BF:32:6E:AD
13:55:47.738 -> Pulse rate: 71 BPM.
13:55:47.738 -> Sent to local server..
13:56:03.260 -> Pulse rate: 71 BPM.
13:56:03.260 -> Sent to local server..
13:56:24.758 -> Pulse rate: 236 BPM.
13:56:24.758 -> Sent to local server..
```

☒ Autoscroll ☒ Show timestamp

Newline 9600 baud Clear output

Cont...



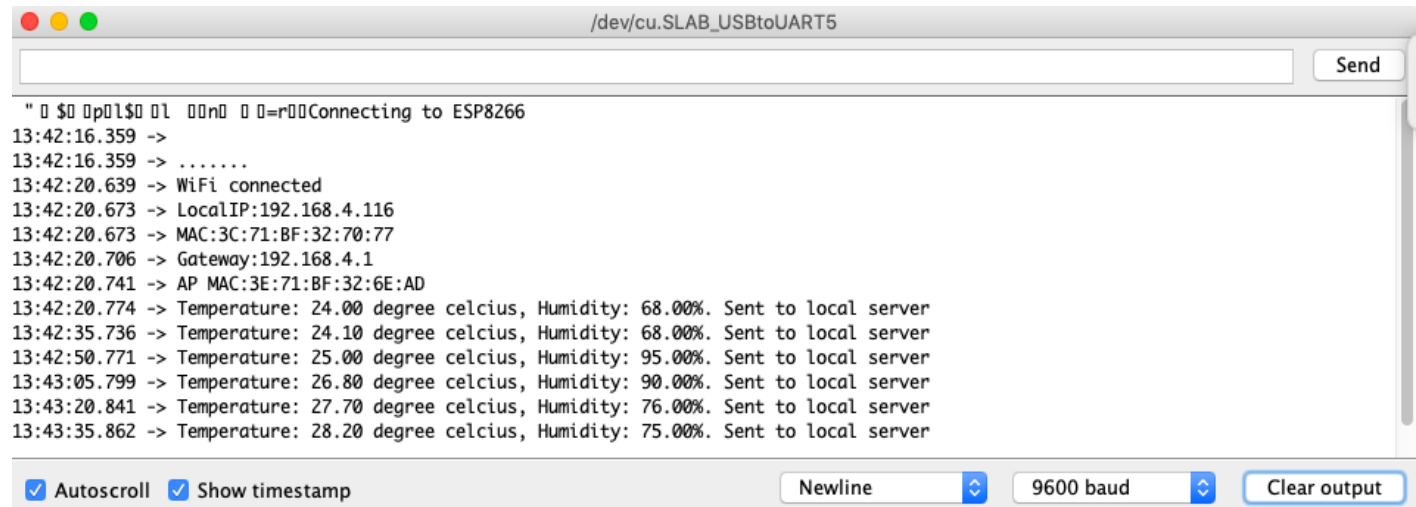
Serial Monitor window for ESP4. The window title is "/dev/cu.SLAB_USBtoUART8". The output shows the following log entries:

```
{ld0|010| 0 $0 b|00 0 0{0b0 b00nn0$nn000 " p00c$ {lp0n0 0 l 00 # n0 l0 0p 00nND 00d' 0 oo $ o{000N #0 $ r00n #0 $0 0 l$0 0$ 00n0
14:19:56.258 -> .....
14:20:02.041 -> WiFi connected
14:20:02.074 -> LocalIP:192.168.4.115
14:20:02.074 -> MAC:3C:71:BF:32:71:5B
14:20:02.109 -> Gateway:192.168.4.1
14:20:02.142 -> AP MAC:3E:71:BF:32:6E:AD
14:20:02.175 -> Vibration sensor value: 29
14:20:02.175 -> Sent to Local server..!!
14:20:17.089 -> Vibration sensor value: 30
14:20:17.122 -> Sent to Local server..!!
14:20:32.108 -> Vibration sensor value: 1013
14:20:32.142 -> Sent to Local server..!!
14:20:47.104 -> Vibration sensor value: 30
14:20:47.138 -> Sent to Local server..!!
```

Controls: ☐ Autoscroll ☒ Show timestamp Newline 9600 baud Clear output

Serial Monitor of ESP4

Serial Monitor of ESP5



Serial Monitor window for ESP5. The window title is "/dev/cu.SLAB_USBtoUART5". The output shows the following log entries:

```
" 0 $0 0p0l$0 0l 00n0 0 0=r00Connecting to ESP8266
13:42:16.359 ->
13:42:16.359 -> .....
13:42:20.639 -> WiFi connected
13:42:20.673 -> LocalIP:192.168.4.116
13:42:20.673 -> MAC:3C:71:BF:32:70:77
13:42:20.706 -> Gateway:192.168.4.1
13:42:20.741 -> AP MAC:3E:71:BF:32:6E:AD
13:42:20.774 -> Temperature: 24.00 degree celcius, Humidity: 68.00%. Sent to local server
13:42:35.736 -> Temperature: 24.10 degree celcius, Humidity: 68.00%. Sent to local server
13:42:50.771 -> Temperature: 25.00 degree celcius, Humidity: 95.00%. Sent to local server
13:43:05.799 -> Temperature: 26.80 degree celcius, Humidity: 90.00%. Sent to local server
13:43:20.841 -> Temperature: 27.70 degree celcius, Humidity: 76.00%. Sent to local server
13:43:35.862 -> Temperature: 28.20 degree celcius, Humidity: 75.00%. Sent to local server
```

Controls: ☒ Autoscroll ☒ Show timestamp Newline 9600 baud Clear output

Results & Graphs in Web



- Open the ThingSpeak page and click on **Channels > My channels**
- Now select the channel that is created for this experiment (In this case '**Monitoring Four Sensors in Star Topology**').

https://thingspeak.com/channels

ThingSpeak™ Channels Apps Community Support Commercial Use How to Buy Account Sign Out

My Channels

New Channel Search by tag

Name	Created	Updated
Temperature & Humidity Monitoring Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 06:44
Monitoring Four sensors in Star Topology Private Public Settings Sharing API Keys Data Import / Export	2019-07-09	2019-07-09 11:30
LED Control from Web Private Public Settings Sharing API Keys Data Import / Export	2019-07-12	2019-07-12 06:53

Help

Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

Click **New Channel** to create a new ThingSpeak channel.

Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.

Learn to [create channels](#), explore and transform data.

Learn more about [ThingSpeak Channels](#).

Examples

- [Arduino](#)
- [Arduino MKR1000](#)
- [ESP8266](#)
- [Raspberry Pi](#)
- [Netduino Plus](#)

Upgrade

Need to send more data faster?

Need to use ThingSpeak for a commercial project?

Cont...

- click on **'Private View'** to see the uploaded data

https://thingspeak.com/channels/819306/private_show

ThingSpeak™ Channels Apps Community Support Commercial Use How to Buy Account Sign Out

Private View Public View Channel Settings Sharing API Keys Data Import / Export

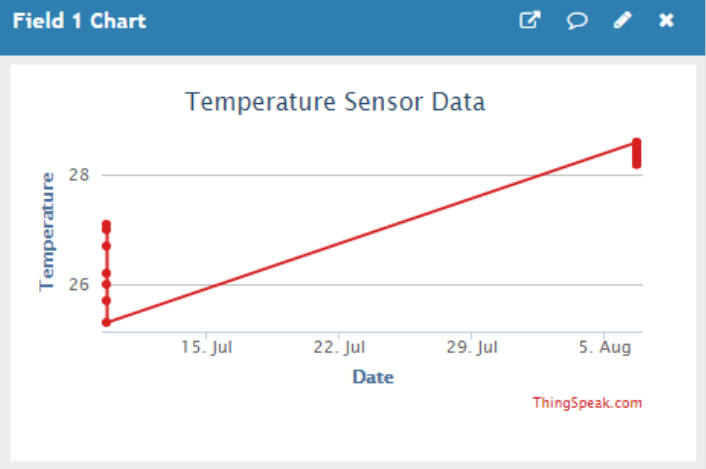
+ Add Visualizations + Add Widgets Export recent data

MATLAB Analysis MATLAB Visualization

Channel Stats
Created: 28 days ago
Last entry: less than a minute ago
Entries: 77

Field 1 Chart

Temperature Sensor Data



Temperature

28

26

15 Jul 22 Jul 29 Jul 5 Aug

Date

ThingSpeak.com

Temperature

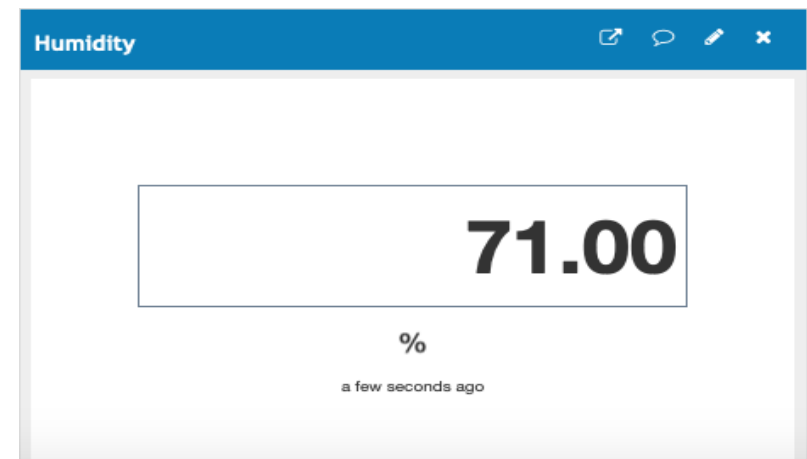
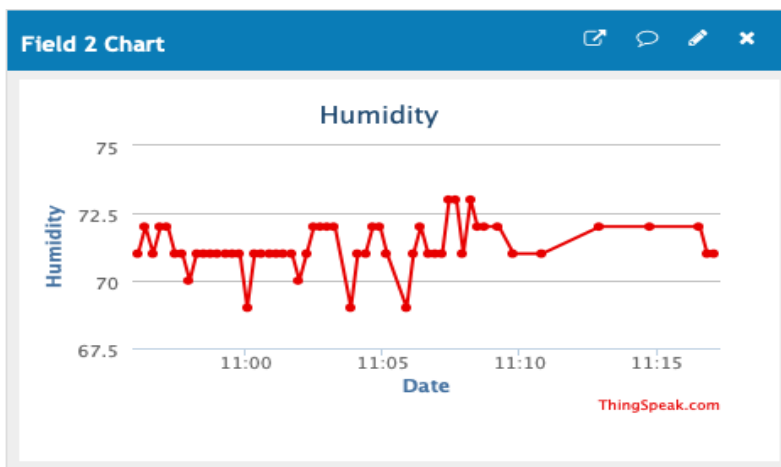
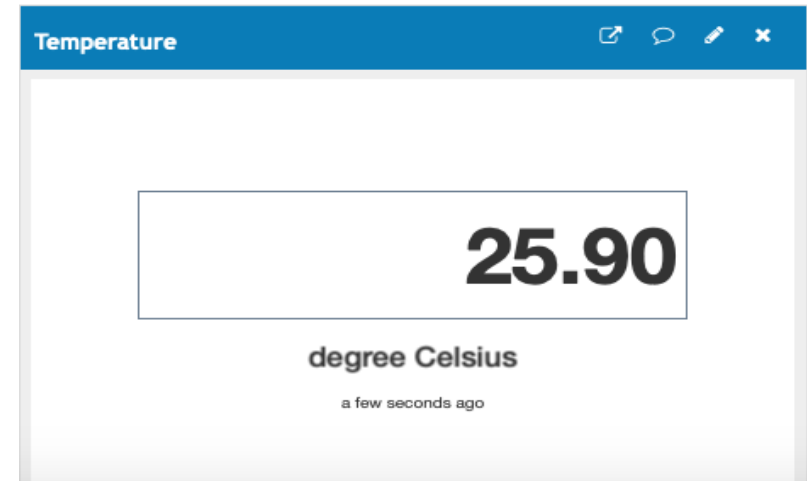
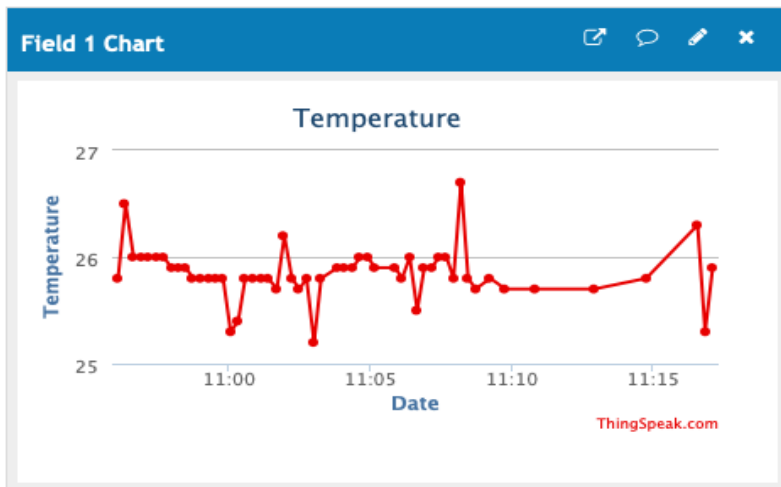
28.20

Degree

a few seconds ago

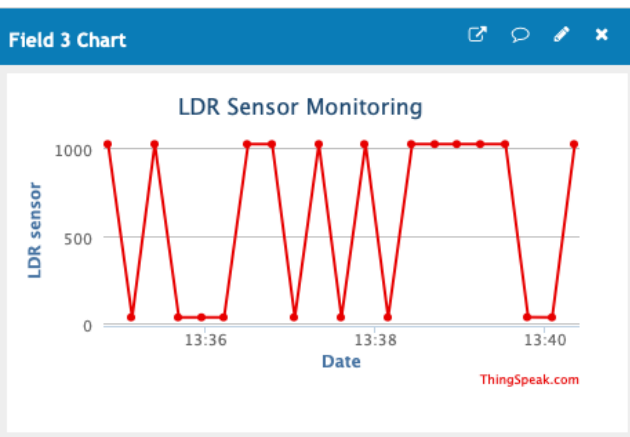
Cont...

- Temperature and Humidity

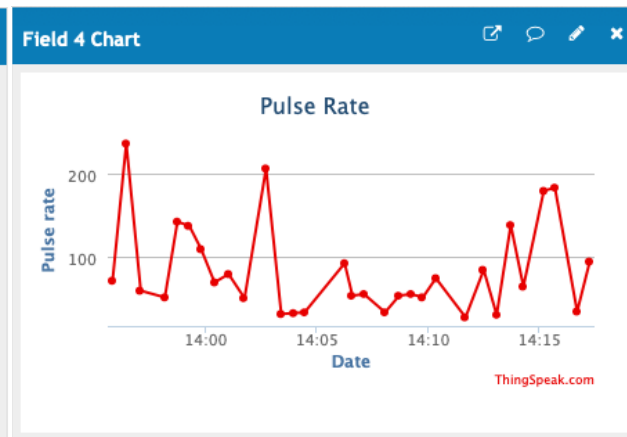


Cont...

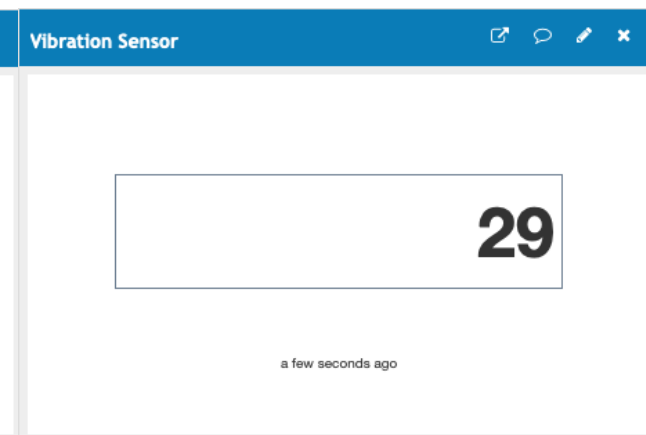
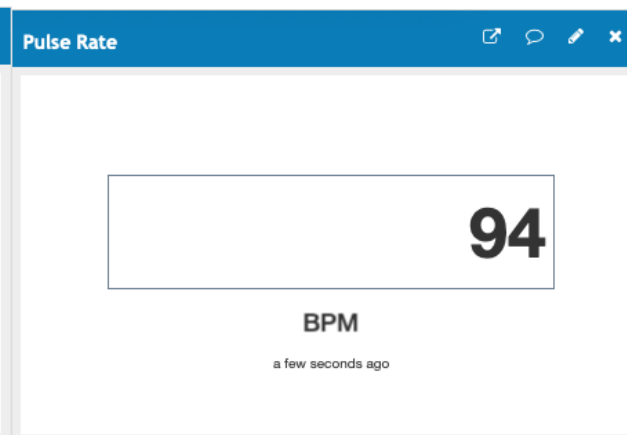
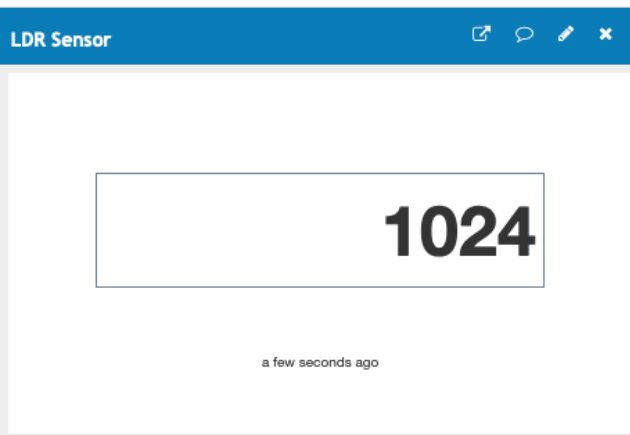
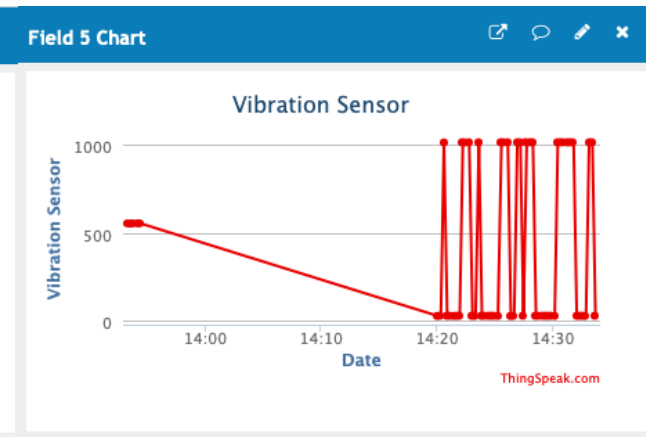
Light Sensor



Pulse Sensor



Vibration Sensor



Thanks!

