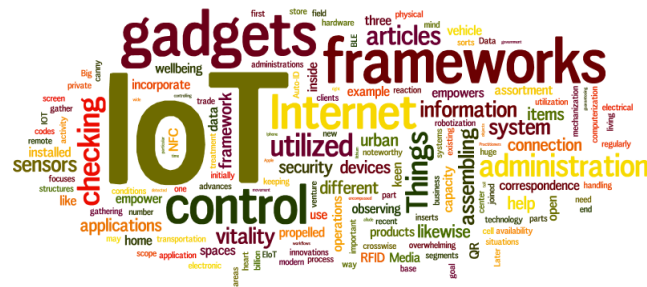


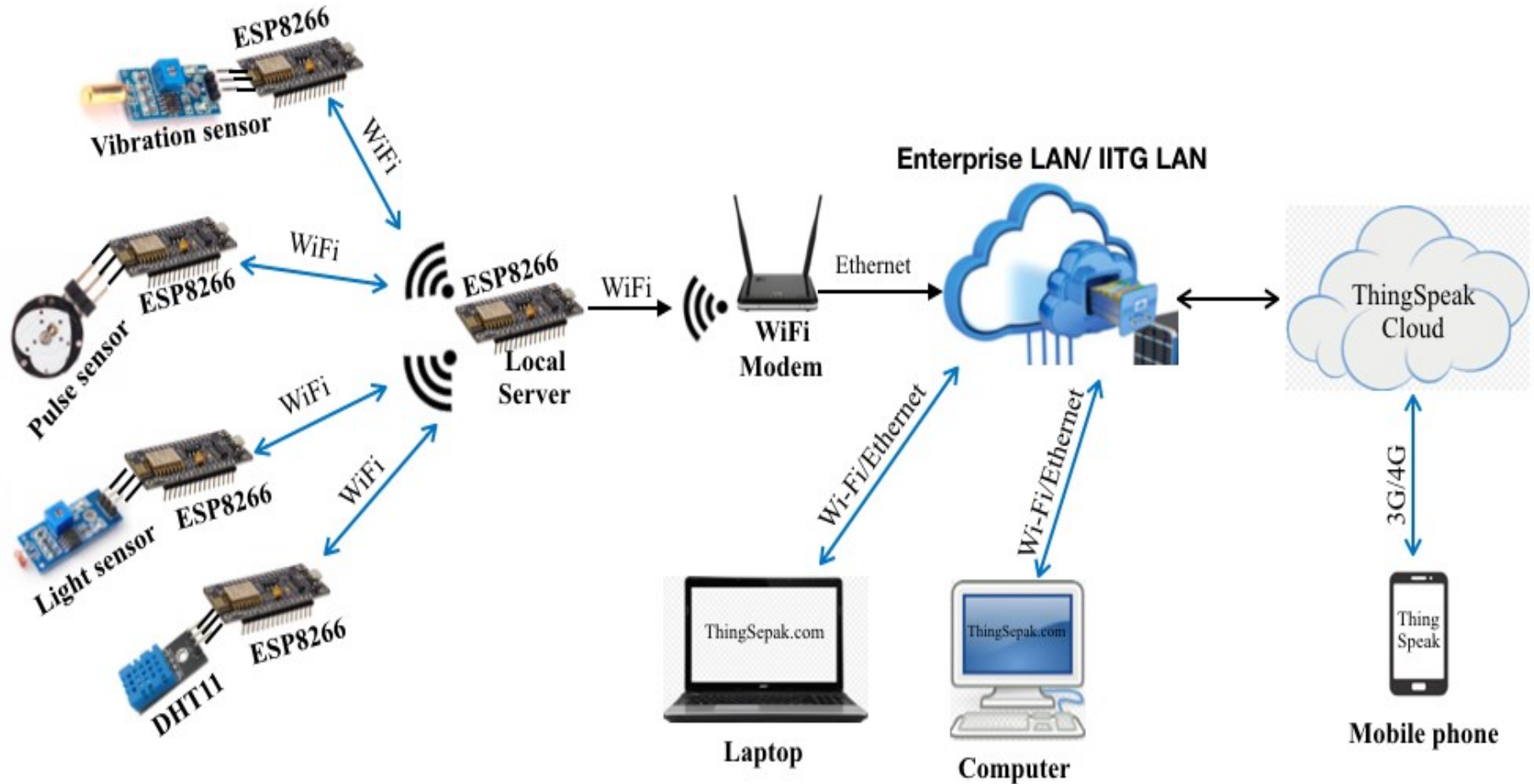
# Smart Home Monitoring Using ESP8266 and Webserver



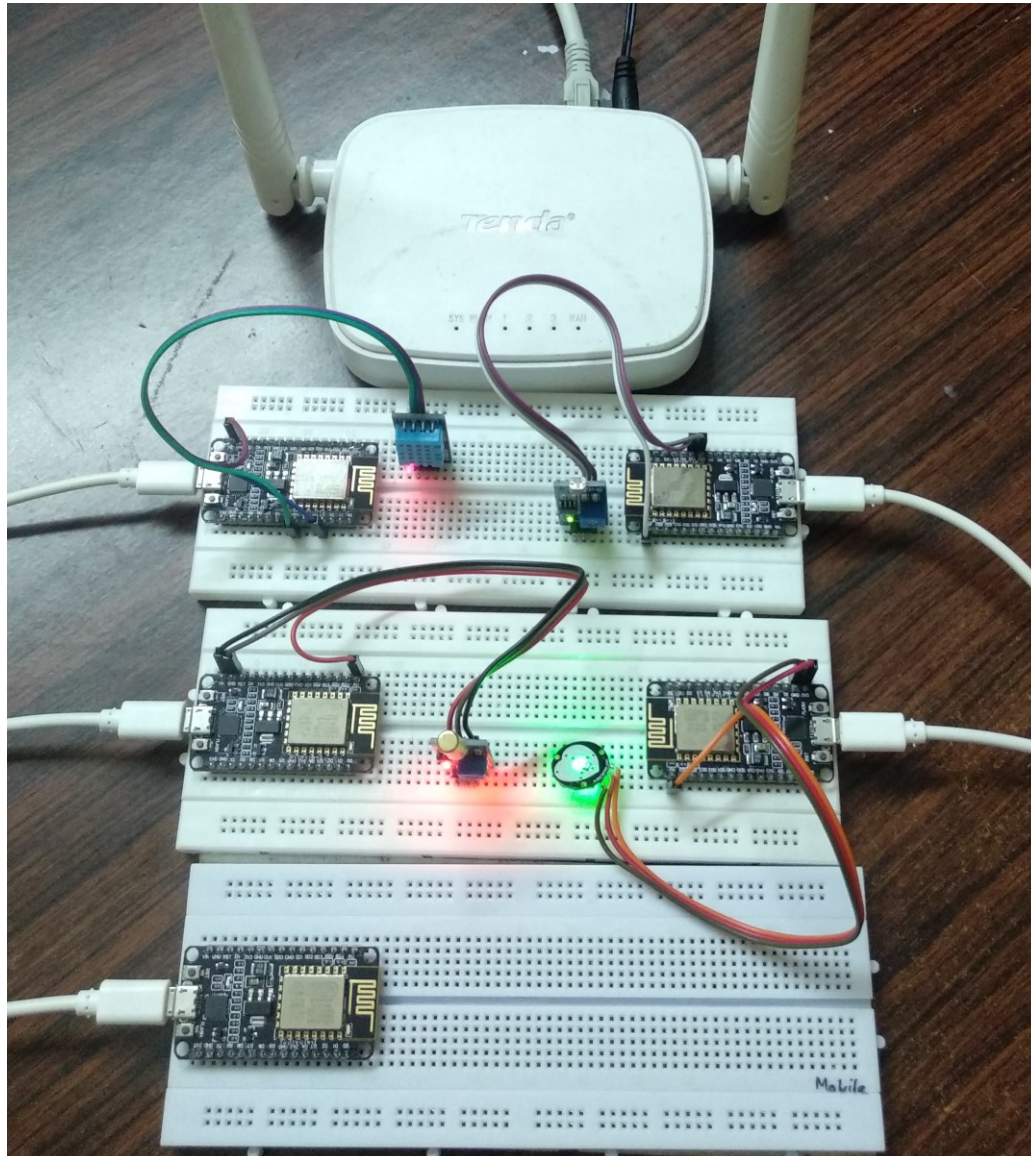
## Thanks to Dr. Manas Khatua

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

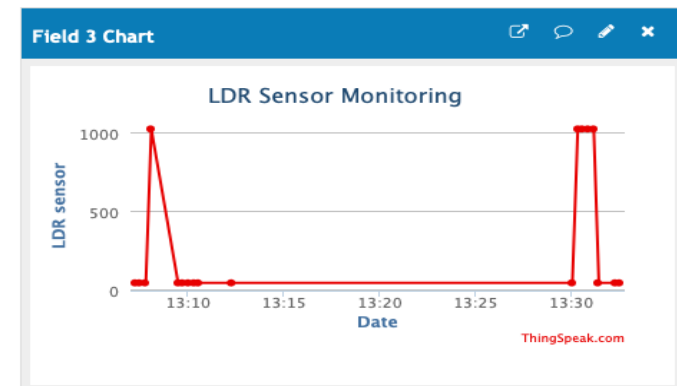
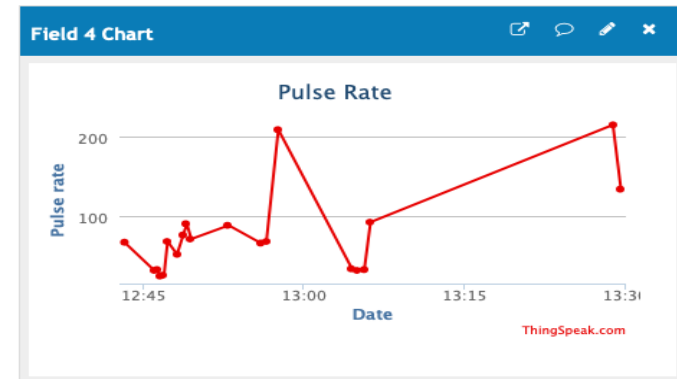
# System Diagram



# Physical Setup



ThingSpeak cloud server  
accessing from a  
Laptop/PC/Smartphone



---

# **Router Configuration To Connect with IITG Internet**

# Router Configuration



- This is TP-Link WiFi Router
  - ESP8266 (local server) will connect to this WiFi AP
  - Sensor data will be uploaded to ThingSpeak server through this WiFi AP.
- 
- Login TP-Link WiFi using given IP (**192.168.0.1**) and password written on its label.
  - Do the following:
    - Go to Quick Setup and click on Next.
    - Choose Operation Mode as Wireless Router and click on Next.
    - Select WAN Connection Type as Static IP and click on Next.
    - Set the **Static IP, Subnet Mask, Default Gateway, Primary DNS Server, Secondary DNS Server** and click on Next.
    - Select the radio bands (2.4 GHz and/or 5 GHz) and click on Next.
    - Setup the Wireless radio bands selected above and click on Next.
    - Confirm the setup by clicking on Save. The router reboots and reconnects.

# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

### Quick Setup - Start

Run the Quick Setup to manually configure your internet connection and wireless settings.

To continue, please click the **Next** button.

To exit, please click the **Exit** button.

Exit

Next



# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

### Quick Setup - Operation Mode

Choose Operation Mode:

☒ **Wireless Router**

Share Internet connection from an Ethernet cable. For example, hotel room, small office...

☐ **Access Point**

☐ **Range Extender**

Back

Next

# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

Access Control

Advanced Routing

Bandwidth Control

IP & MAC Binding

### Quick Setup - WAN Connection Type

The Quick Setup is preparing to set up your internet connection, please choose one type below according to your ISP. The detailed description will be displayed after you choose the corresponding type.

☐ Auto-Detect

☐ Dynamic IP (Most common option)

☒ Static IP

Your ISP provides you specified IP parameters.

☐ PPPoE/Russian PPPoE

☐ L2TP/Russian L2TP

☐ PPTP/Russian PPTP

Note: For users in some areas(such as Russia, Ukraine etc.), please contact your ISP to choose connection type manually.

☐ More Advanced Settings

Back

Next



# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

Access Control

Advanced Routing

### Quick Setup - Static IP

Please enter the basic parameter settings provided by your ISP. If basic parameters are unknown, please contact ISP.

IP Address:

Subnet Mask:

Default Gateway:

Primary DNS Server:

Secondary DNS Server:  (optional)

Back

Next

# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

### Quick Setup - Wireless Dual Band Selection

Please select or clear the check box to enable or disable a given radio band.

☒ 2.4GHz

☒ 5GHz

Back

Next

# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

Access Control

Advanced Routing

Bandwidth Control

IP & MAC Binding

Dynamic DNS

IPv6

System Tools

### Quick Setup - Wireless 2.4GHz

Wireless Network Name:  (Also called SSID)

Security:



**WPA2-PSK (Recommended)**

Wireless Password

(Enter ASCII characters between 8 and 63 or Hexadecimal characters between 8 and 64.)



**Disable Wireless Security**

☒ **More Advanced Wireless Settings**

Band: 2.4GHz

Mode:

Channel Width:

Channel:

Back

Next

# Cont...



## AC750 Wireless Dual Band Router Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

Access Control

Advanced Routing

Bandwidth Control

IP & MAC Binding

Dynamic DNS

IPv6

System Tools

### Quick Setup - Wireless 5GHz

Wireless Network Name:  (Also called SSID)

Security:



**WPA2-PSK (Recommended)**

Wireless Password

\*\*\*\*\*

(Enter ASCII characters between 8 and 63 or Hexadecimal characters between 8 and 64.)



**Disable Wireless Security**

☒ **More Advanced Wireless Settings**

Band: 5GHz

Mode:

Channel Width:

Channel:

Back

Next

# Cont...



## AC750 Wireless Dual Band Router

Model No. Archer C20

Status

Quick Setup

Operation Mode

Network

Dual Band Selection

Wireless 2.4GHz

Wireless 5GHz

Guest Network

DHCP

Forwarding

Security

Parental Controls

Access Control

Advanced Routing

Bandwidth Control

IP & MAC Binding

Dynamic DNS

IPv6

System Tools

Logout

### Quick Setup - Confirm

The Quick Setup is complete. Please confirm all parameters below. Click BACK to modify any settings or click SAVE to save and apply your configurations.

Parameters Summary:

Connection Type: Static IP  
IP Address: 172.16.117.192  
Subnet Mask: 255.255.248.0  
Gateway: 172.16.112.1  
DNS Server: 172.17.1.1, 172.17.1.2

Wireless 2.4GHz: Enabled  
Wireless Network Name(SSID): TP-Link\_A522  
Channel: Auto  
Mode: 11bgn mixed  
Channel Width: Auto  
Security: WPA2-Personal  
Wireless Password: \*\*\*\*\*

Wireless 5GHz: Enabled  
Wireless Network Name(SSID): TP-Link\_A522\_5G  
Channel: Auto  
Mode: 11a/n/ac mixed  
Channel Width: Auto  
Security: WPA2-Personal  
Wireless Password: 39508324

Back

Save

# Cont...



## AC750 Wireless Dual Band Router Model No. Archer C20

### Status

- Quick Setup
- Operation Mode
- Network
- Dual Band Selection
- Wireless 2.4GHz
- Wireless 5GHz
- Guest Network
- DHCP
- Forwarding
- Security
- Parental Controls
- Access Control
- Advanced Routing
- Bandwidth Control
- IP & MAC Binding
- Dynamic DNS
- IPv6
- System Tools
- Logout

Subnet Mask: 255.255.255.0

### Wireless 2.4GHz

Operation Mode: **Router**  
Wireless Radio: Enabled  
Name(SSID): TP-Link\_A522  
Mode: 11bgn mixed  
Channel: Auto(Channel 4)  
Channel Width: Auto  
MAC Address: E8:48:B8:61:A5:22

### Wireless 5GHz

Operation Mode: **Router**  
Wireless Radio: Enabled  
Name(SSID): TP-Link\_A522\_5G  
Mode: 11a/n/ac mixed  
Channel: Auto(Channel 149)  
Channel Width: Auto  
MAC Address: E8:48:B8:61:A5:21

### WAN

MAC Address: E8:48:B8:61:A5:23  
IP Address: 172.16.117.192(Static IP)  
Subnet Mask: 255.255.248.0  
Default Gateway: 172.16.112.1  
DNS Server: 172.17.1.1 172.17.1.2


### Ethernet

Internet: 100Mbps full duplex  
LAN1: Unplugged  
LAN2: Unplugged  
LAN3: Unplugged  
LAN4: Unplugged

System Up Time: 0 day(s) 00:46:11

[Refresh](#)

# Connecting with Internet



**User Authentication Required**

Use IITG Credentials to Login

Username

Password

**LOGIN**

Forgot Password ? [Reset Here](#)

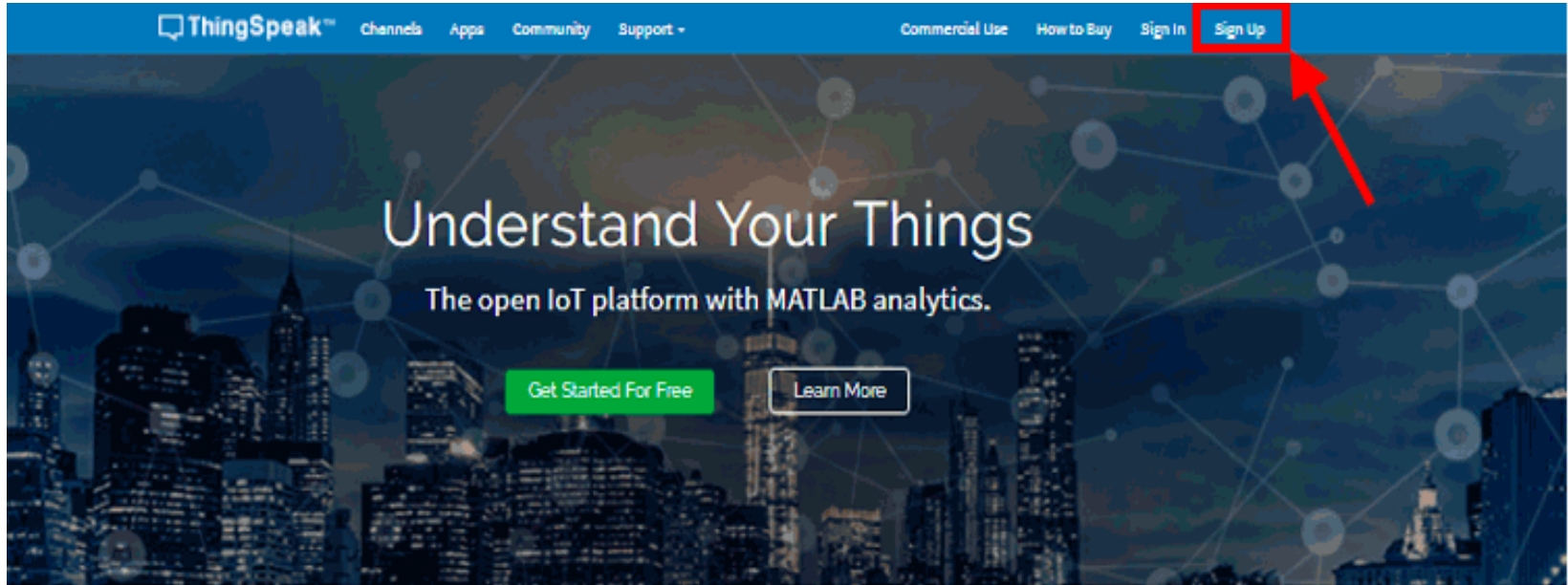
- You should be able to access Internet in your Mobile/Laptop using TP-Link WiFi AP



---

# **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...

## My Channels

New Channel

Search by tag



Name	Created	Updated
Temperature & Humidity Monitoring <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 06:44
Monitoring Four sensors in Star Topology <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 11:30
LED Control from Web <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	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...

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...

## My Channels

New Channel

Search by tag



Name	Created	Updated
Temperature & Humidity Monitoring <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 06:44
Monitoring Four sensors in Star Topology <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 11:30
LED Control from Web <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	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

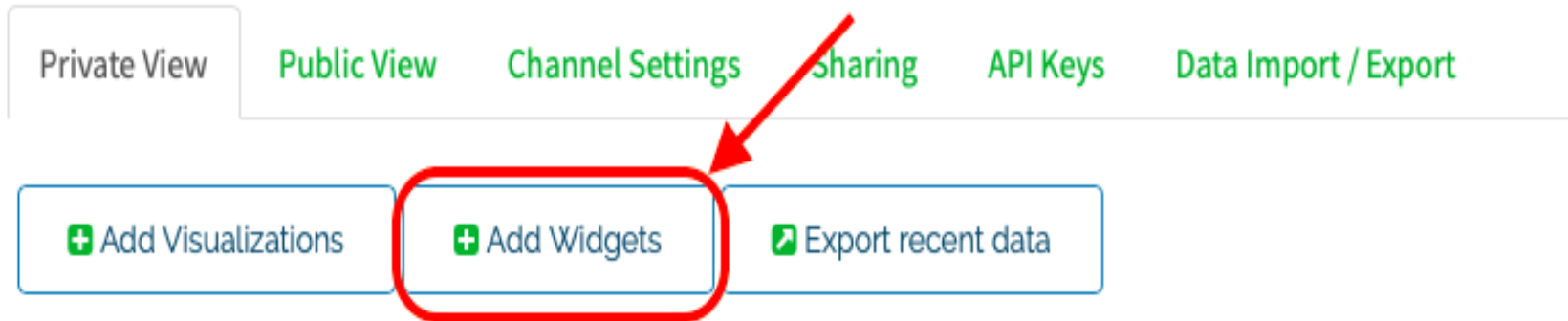
- [Arduino](#)
- [Arduino MKR1000](#)
- [ESP8266](#)
- [Raspberry Pi](#)
- [Netduino Plus](#)

## Upgrade

Need to send more data faster?

Need to use ThingSpeak for a commercial project?

# Create Channel Display



**Temperature Options** ? x

**Name**

**Field**

**Update Interval**

**Units**

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

- 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

Monitoring Four sensors in Star Topology

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 **write 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

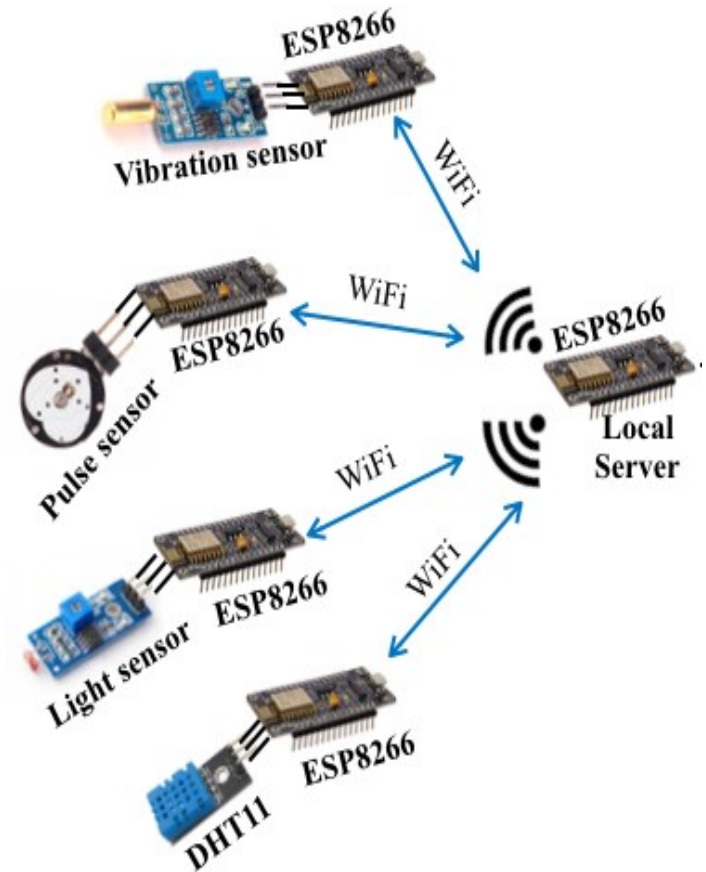


---

# IoT Network Configuration

# IoT Network Configuration

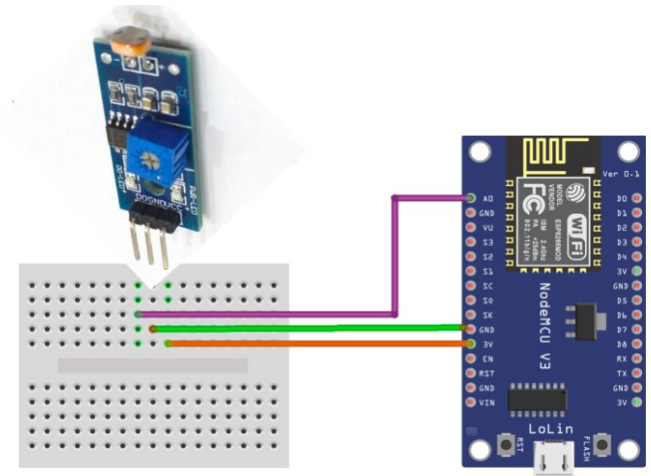
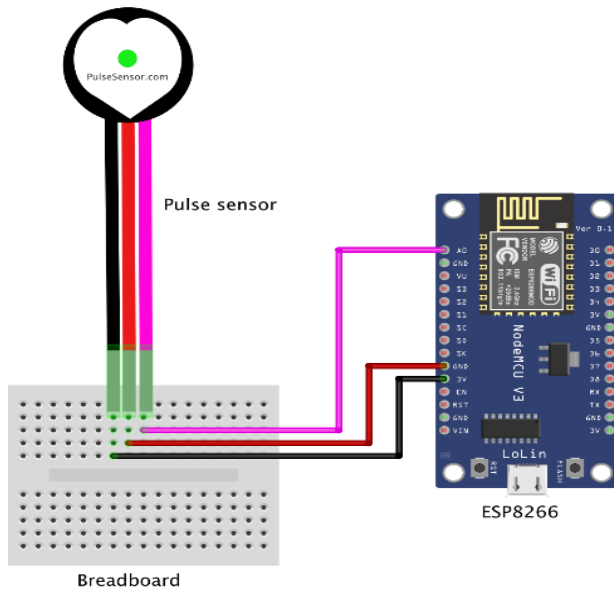
- There are total five ESP8266
  - one is acting as **server**,
  - other four as **clients** in local network.
- ESP1- ESP8266 acting as **local server**
- ESP2- ESP8266 with **Light** sensor
- ESP3- ESP8266 with **Pulse** sensor
- ESP4- ESP8266 with **vibration** sensor
- ESP5- ESP8266 with **temperature** & **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 pin of ESP2
- Connect DATA OUT pin of LDR sensor with A0 pin of ESP2.



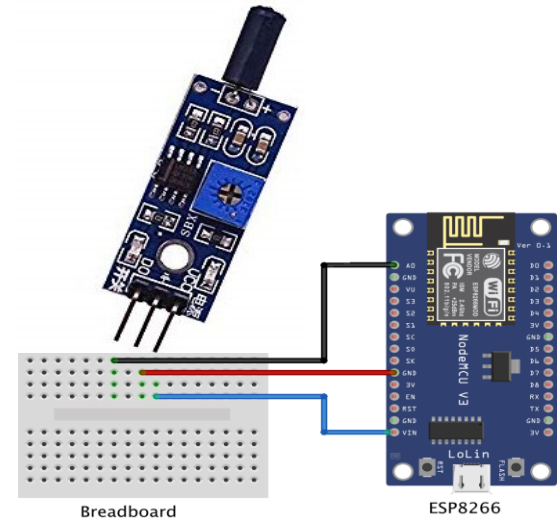
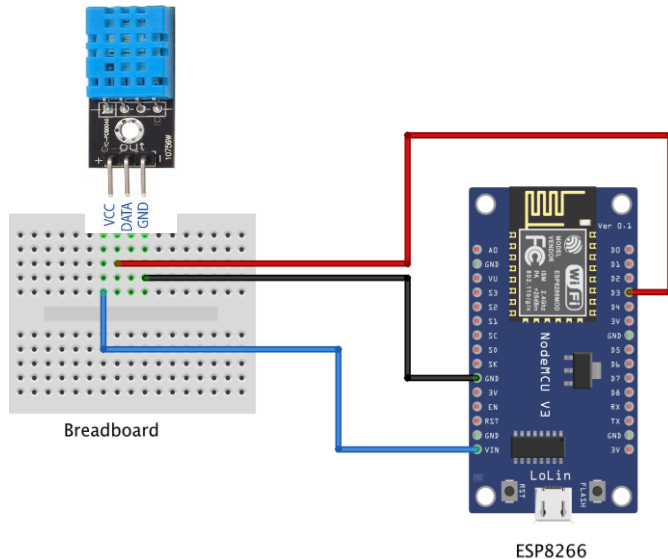
## ESP8266 with Pulse Sensor

- Connect VCC pin of pulse sensor with 3V3 pin of **ESP3**
- Connect GND pin of pulse sensor with GND pin of ESP3
- Connect SIGNAL pin of pulse sensor with A0 pin 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 GND pin of ESP4
- Connect DATA OUT pin of vibration sensor with A0 pin of ESP4



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

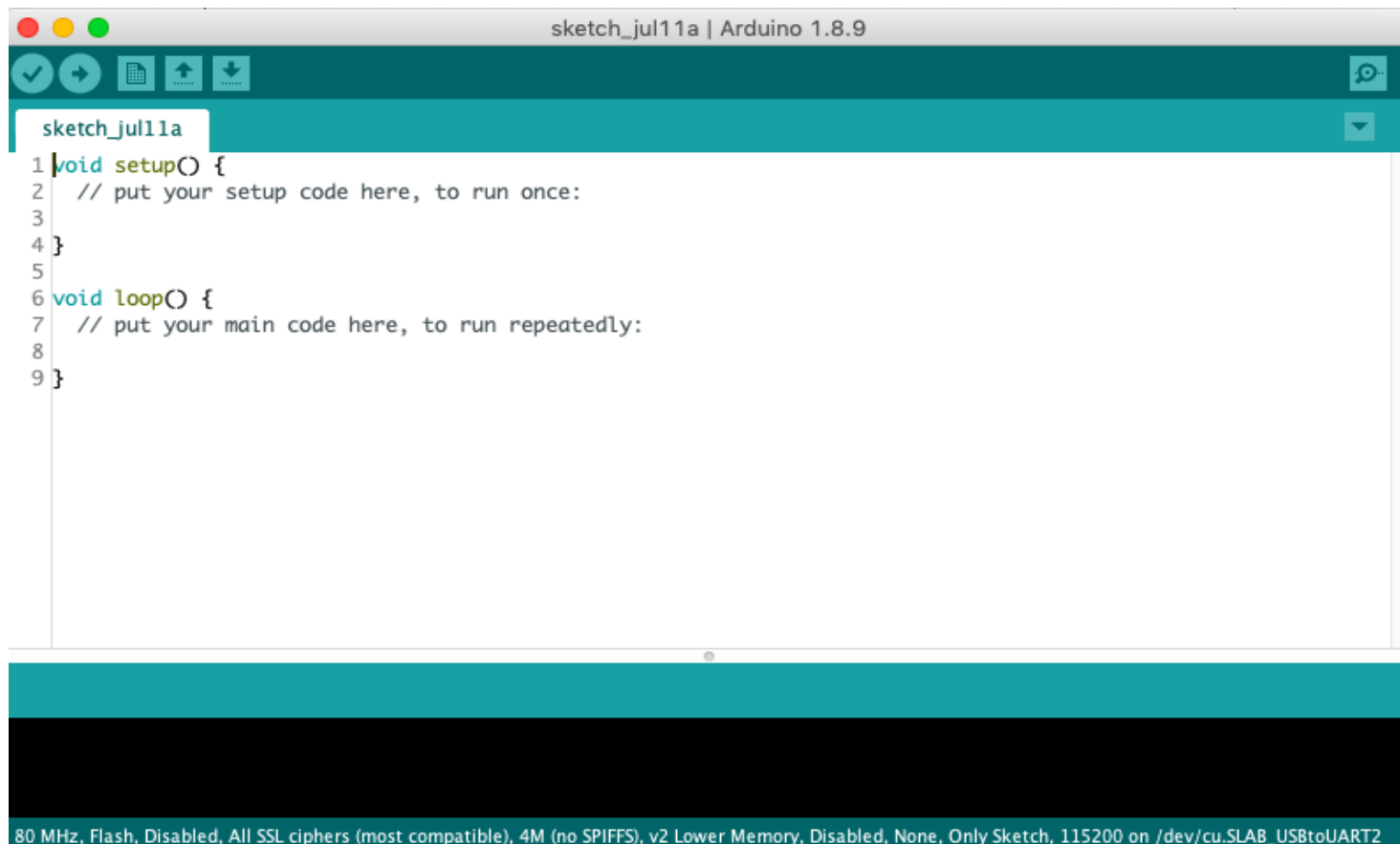
- Connect VCC pin of DHT11 with VIN pin of **ESP5**
- Connect DATA OUT pin of DHT11 with D3 pin of ESP5
- Connect GND pin of DHT11 with GND pin of ESP5

---

# **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:



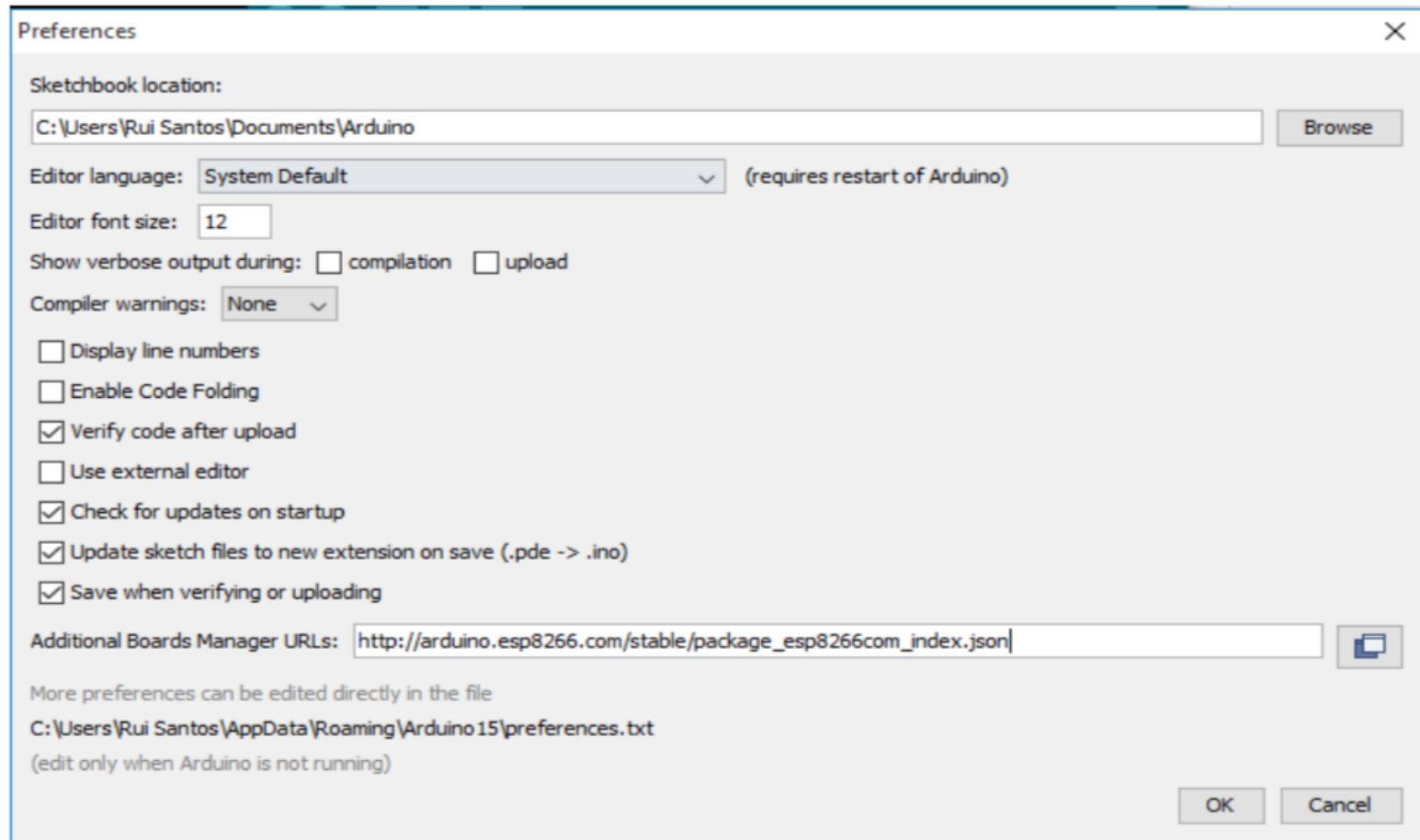
The screenshot shows the Arduino IDE window titled "sketch\_jul11a | Arduino 1.8.9". The interface includes a toolbar with icons for checking, running, serial monitor, upload, and download. The main text area contains the following code:

```
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
```

At the bottom of the window, a status bar displays hardware and software details: "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](http://arduino.esp8266.com/stable/package_esp8266com_index.json) OR  
[https://github.com/esp8266/Arduino/releases/download/2.3.0/package\\_esp8266com\\_index.json](https://github.com/esp8266/Arduino/releases/download/2.3.0/package_esp8266com_index.json)



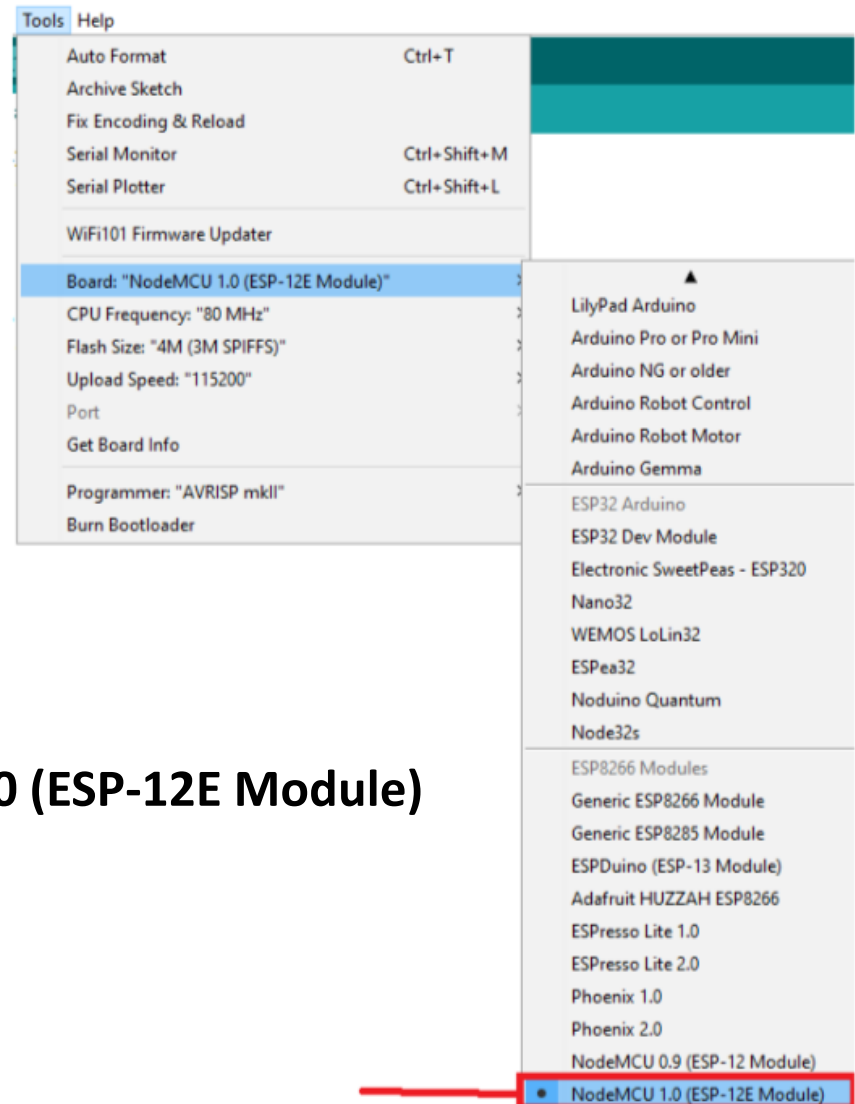


# Cont...

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



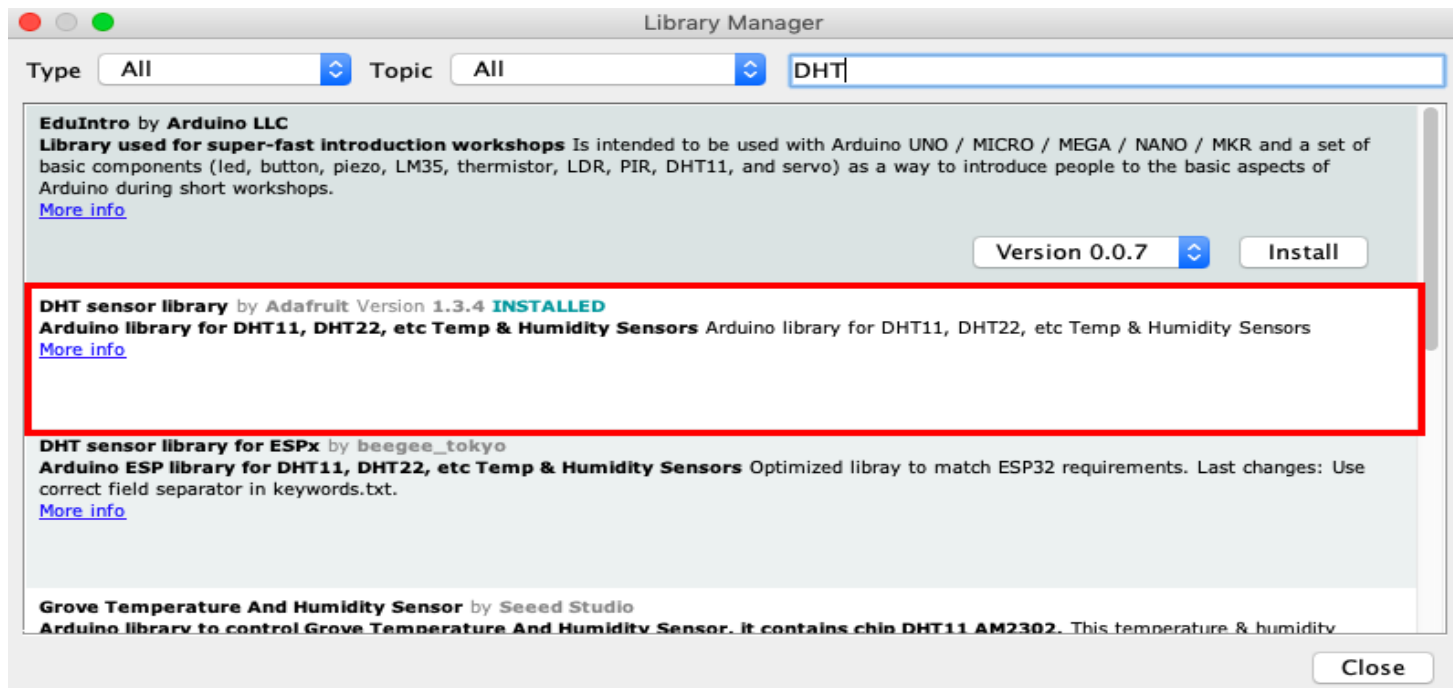
# Cont...



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

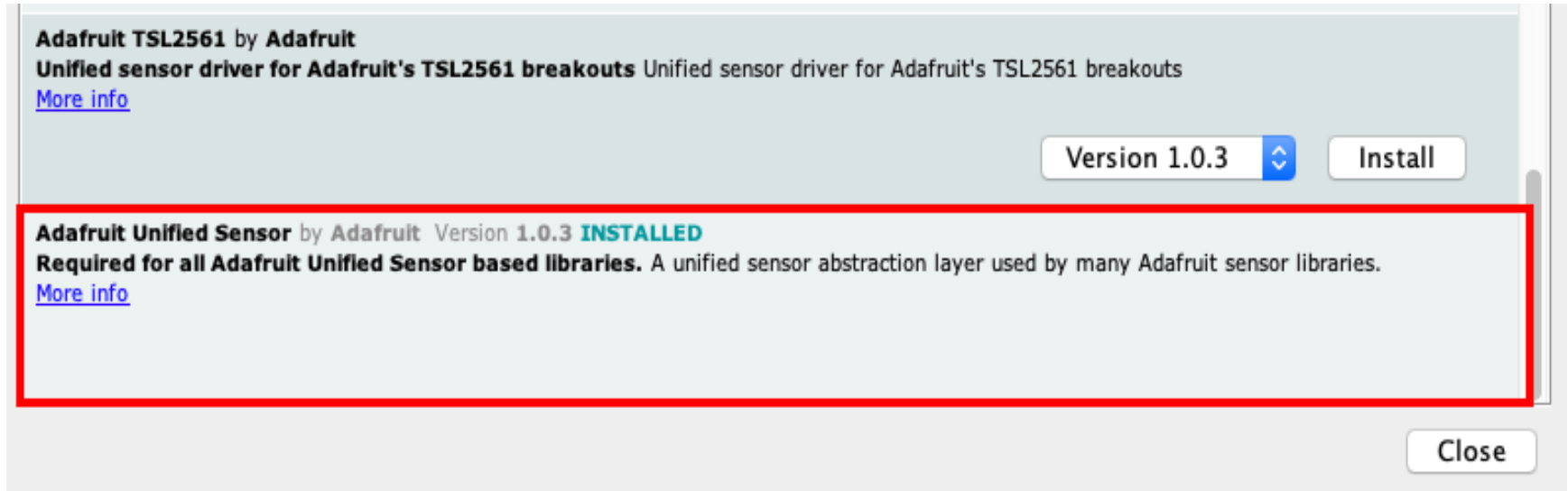
# Install Sensor Libraries

- 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**
    - Download the library as .Zip --> extract it
    - Place the files in File --> Preferences --> Sketchbook location
    - Restart Arduino IDE

---

# 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(172,16,117,192);      //Static IP address of local server
IPAddress gateway(172,16,112,1);   //Gateway of the network
IPAddress mask(255, 255, 248, 0); //Subnet mask of the network
WiFiClient client;
WiFiServer server(80);

unsigned long myChannelNumber = 2244718; //Replace with channelID of ThingSpeak channel ID
const char * myWriteAPIKey = "T4N14GFNKOPDWIWL"; //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 = "TP-Link_A522";   //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(ssid, password));  
  delay(10);  
  Serial.println(WiFi.softAPIP());  
  delay(500);  
  WiFi.begin(ssid, pass);  
  while(WiFi.status() != WL_CONNECTED) {  
    Serial.print(".");  
    delay(500);  
  }  
  Serial.print("Connected to Wifi with ssid ");  
  Serial.println(ssid);  
  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(172,16,117,192);  // 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(172,16,117,192); // 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(172,16,117,192);   // 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(172,16,117,192); // 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...

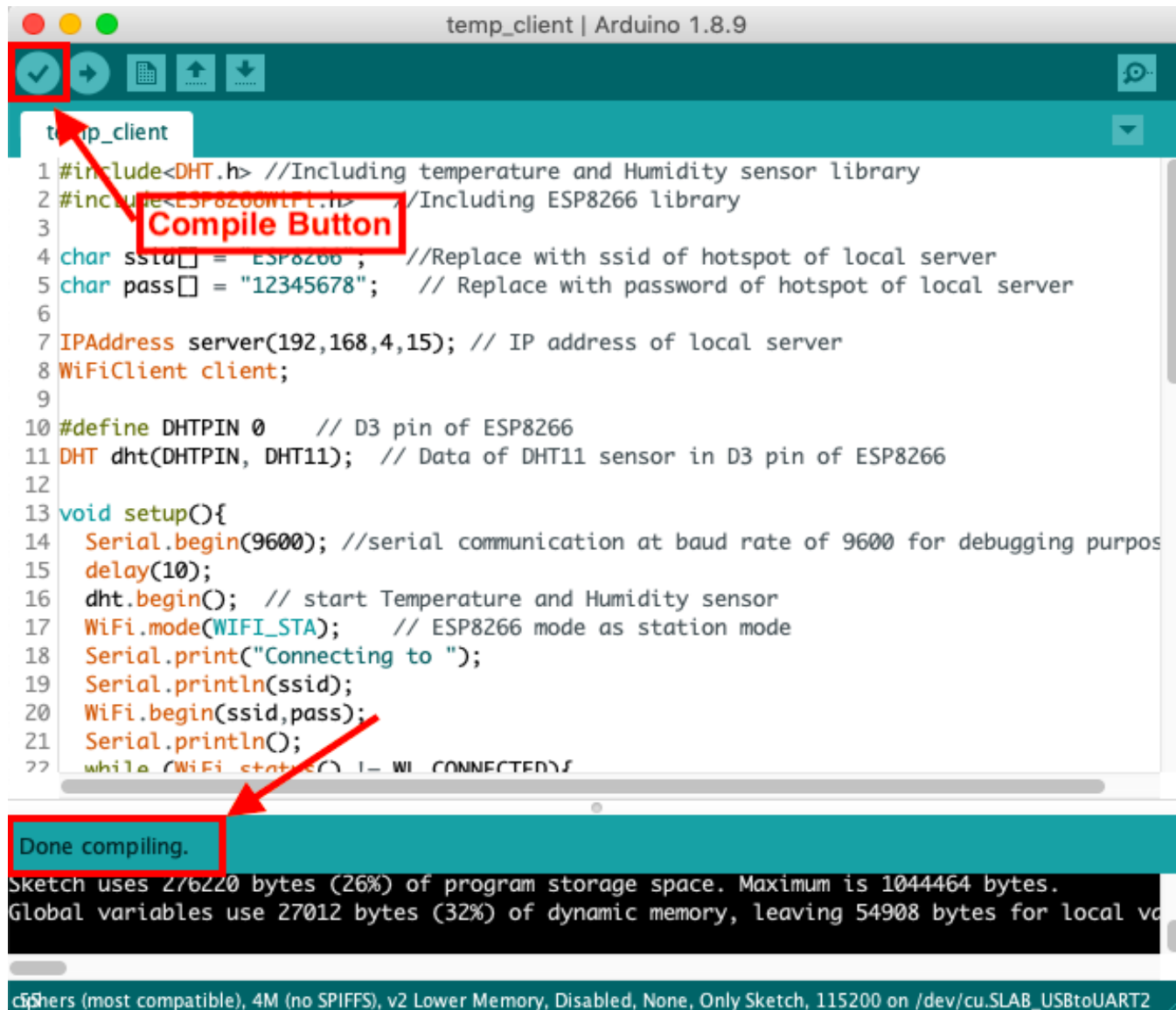
```
void loop() {
    float h = dht.readHumidity();    // Read Humidity value from sensor
    float t = dht.readTemperature(); // Read temp value from sensor
    if(isnan(h) || isnan(t)) {
        Serial.println("Failed to read from DHT sensor"); // Error message
        return;
    }
    if(client.connect(server,80))    // 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

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
```

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...

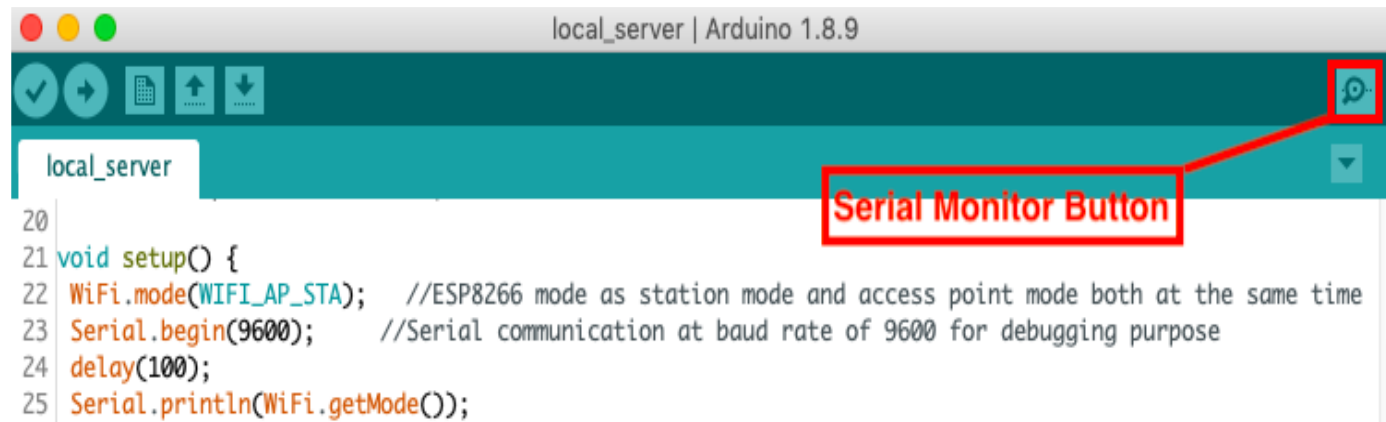
NodeMCU 1.0 (ESP-12E Module) on /dev/cu.SLAB\_USBtoUART

---

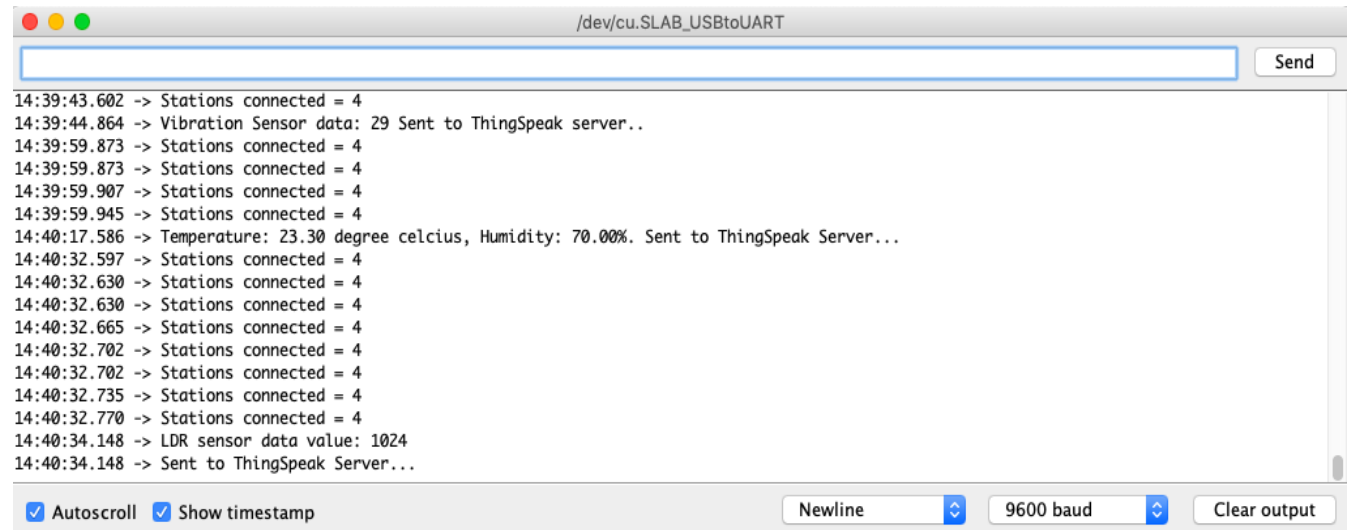
# Observe Outputs

# Open Serial Monitor

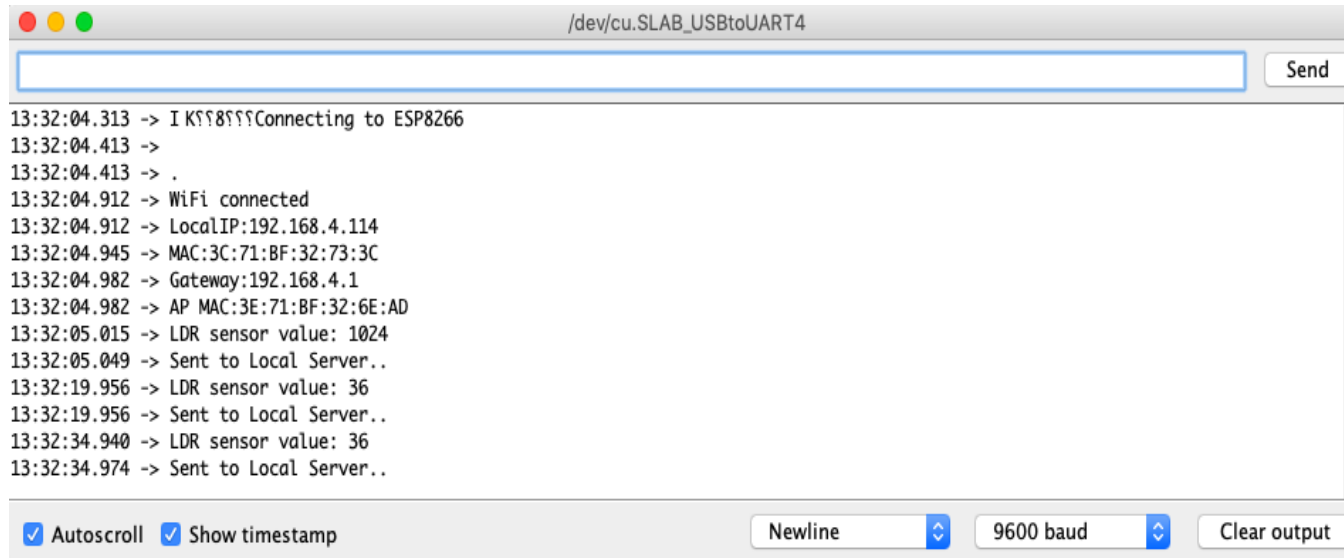
- First **select the port** (go to [Tools > Port:](#) ) to which the board is connected then click the icon of **Serial Monitor** on the top right side of the Arduino IDE



## Serial Monitor of Local Server



# Cont...



A screenshot of a serial monitor window titled "/dev/cu.SLAB\_USBtoUART4". The window has a text input field at the top and a "Send" button. The main area displays a log of messages from an ESP8266. The messages include connection status, IP address, MAC address, gateway, and sensor data. At the bottom, there are checkboxes for "Autoscroll" and "Show timestamp", and buttons for "Newline", "9600 baud", and "Clear output".

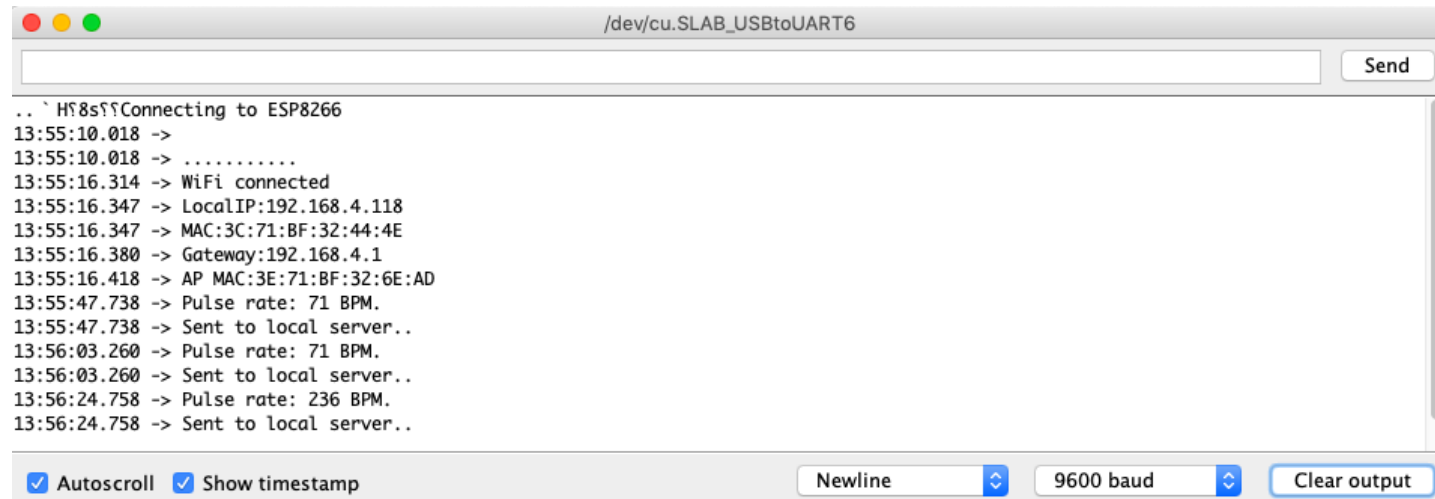
```
13:32:04.313 -> I Kf8sConnecting to ESP8266
13:32:04.413 ->
13:32:04.413 -> .
13:32:04.912 -> WiFi connected
13:32:04.912 -> LocalIP:192.168.4.114
13:32:04.945 -> MAC:3C:71:BF:32:73:3C
13:32:04.982 -> Gateway:192.168.4.1
13:32:04.982 -> AP MAC:3E:71:BF:32:6E:AD
13:32:05.015 -> LDR sensor value: 1024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



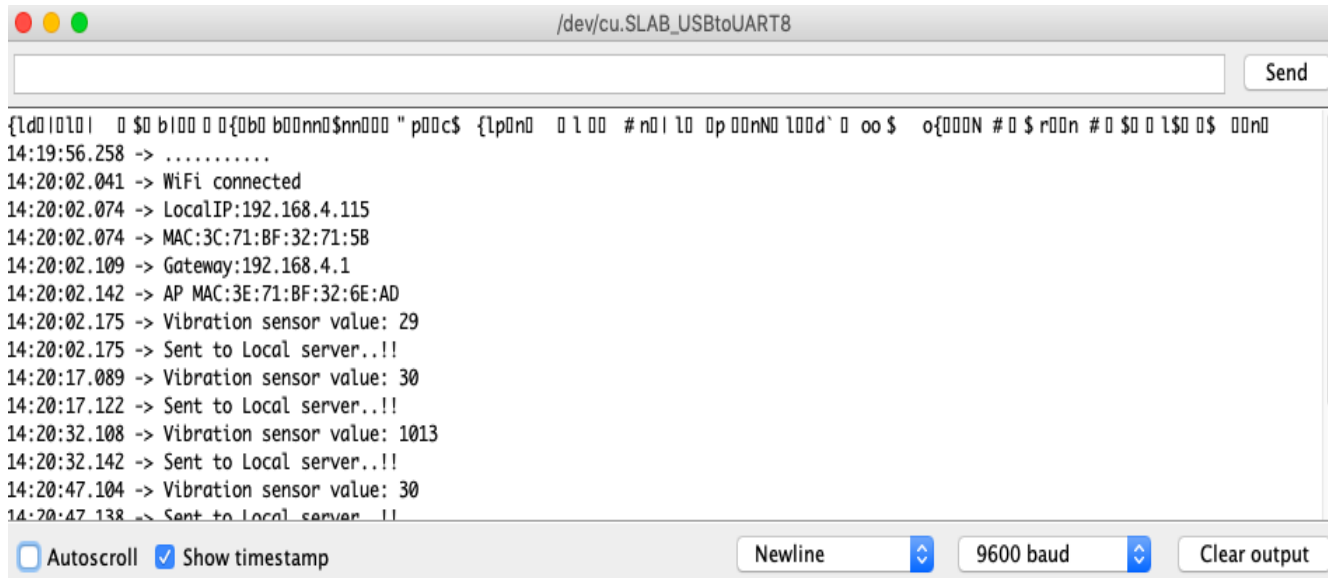
A screenshot of a serial monitor window titled "/dev/cu.SLAB\_USBtoUART6". The window has a text input field at the top and a "Send" button. The main area displays a log of messages from an ESP8266. The messages include connection status, IP address, MAC address, gateway, and pulse rate data. At the bottom, there are checkboxes for "Autoscroll" and "Show timestamp", and buttons for "Newline", "9600 baud", and "Clear output".

```
..`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...

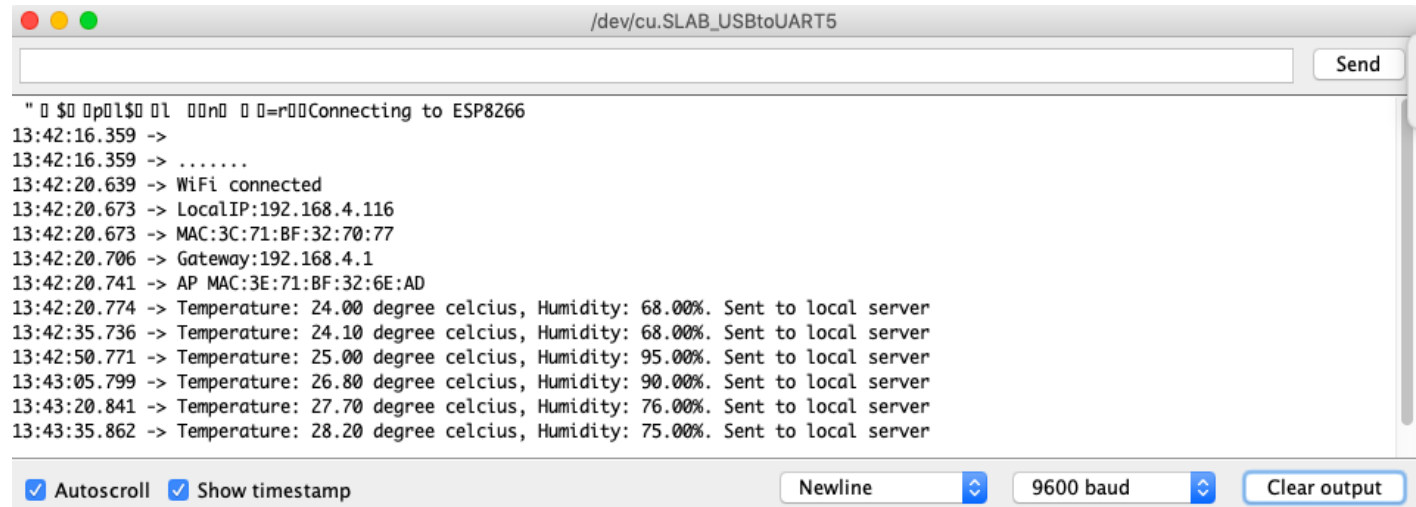


A screenshot of a macOS Serial Monitor window titled "/dev/cu.SLAB\_USBtoUART8". The window shows a log of system events. At the bottom, there are checkboxes for "Autoscroll" (unchecked) and "Show timestamp" (checked), along with dropdown menus for "Newline" and "9600 baud", and a "Clear output" button.

```
{ld0|0l0| 0 $0 b|00 0 0{0b0 b00n0$nn000 " p00c$ {lp0n0 0 l 00 # n0 l0 0p 00n00 l00d' 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..!!
```

## Serial Monitor of ESP4

## Serial Monitor of ESP5

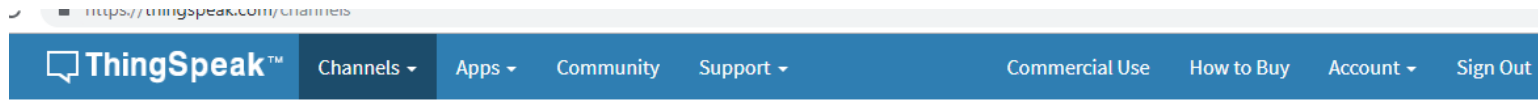


A screenshot of a macOS Serial Monitor window titled "/dev/cu.SLAB\_USBtoUART5". The window shows a log of system events, including temperature and humidity readings. At the bottom, there are checkboxes for "Autoscroll" (checked) and "Show timestamp" (checked), along with dropdown menus for "Newline" and "9600 baud", and a "Clear output" button.

```
" 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
```

# 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**').



## My Channels

New Channel

Search by tag

Name	Created	Updated
Temperature & Humidity Monitoring <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 06:44
<b>Monitoring Four sensors in Star Topology</b> <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	2019-07-09	2019-07-09 11:30
LED Control from Web <a href="#">Private</a> <a href="#">Public</a> <a href="#">Settings</a> <a href="#">Sharing</a> <a href="#">API Keys</a> <a href="#">Data Import / Export</a>	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

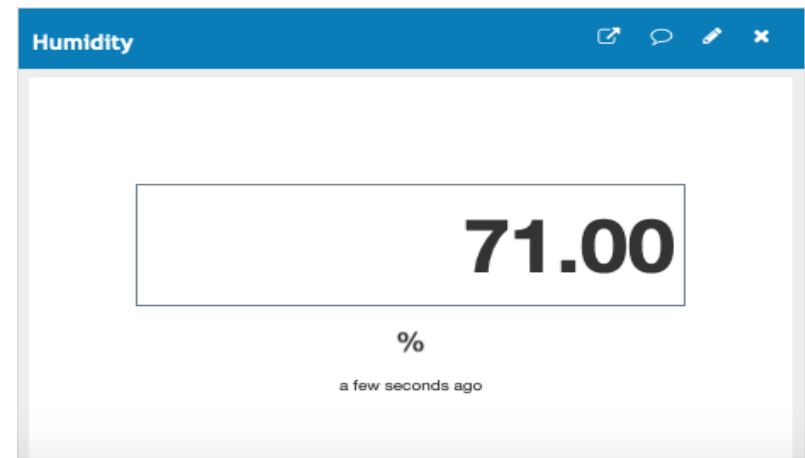
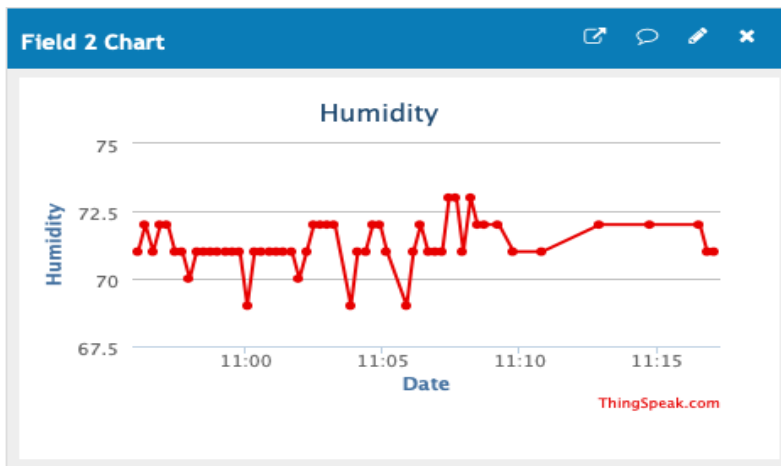
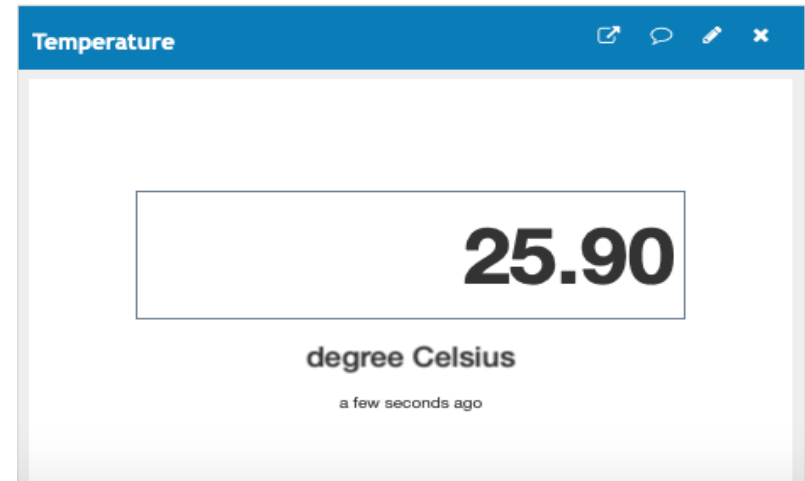
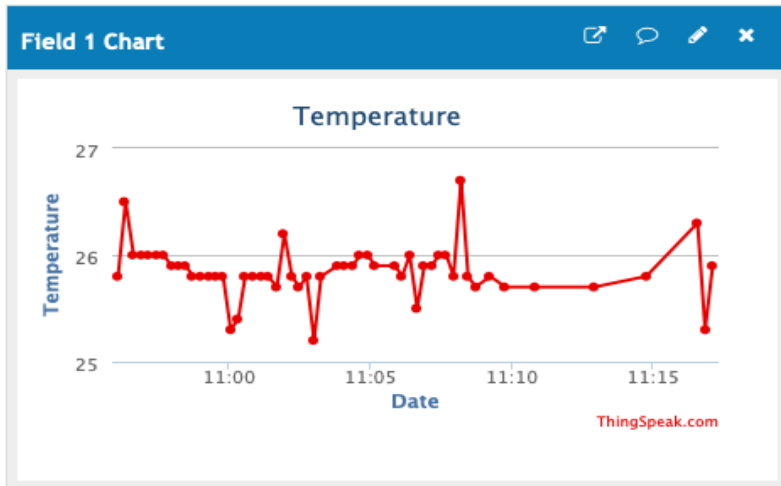
**Channel Stats**  
Created: 28 days ago  
Last entry: less than a minute ago  
Entries: 77

**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

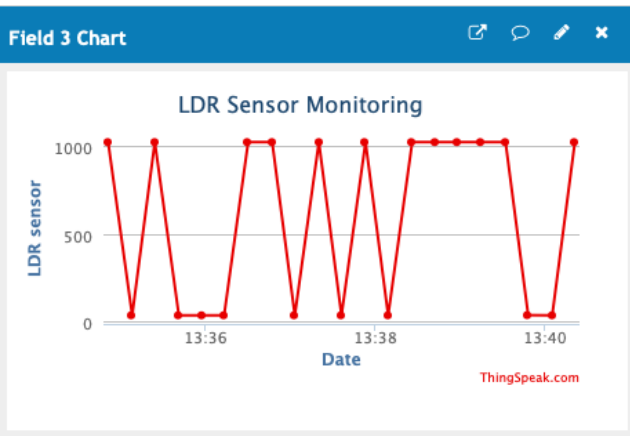
# Cont...

- Temperature and Humidity

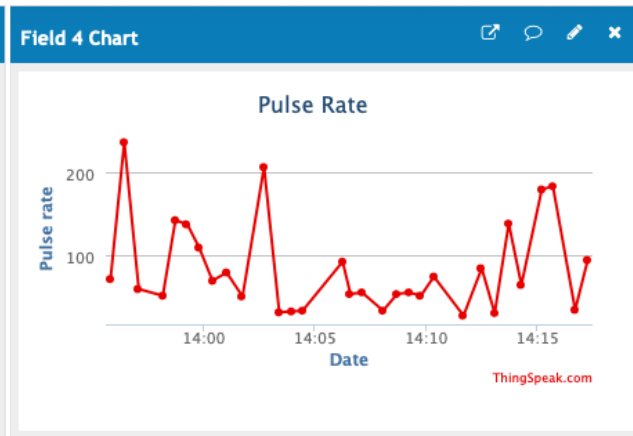


# Cont...

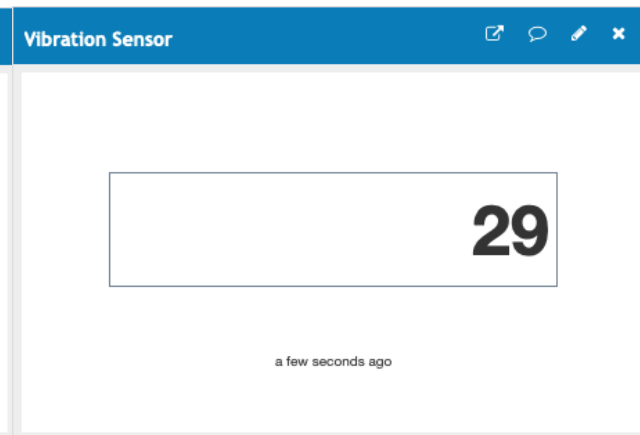
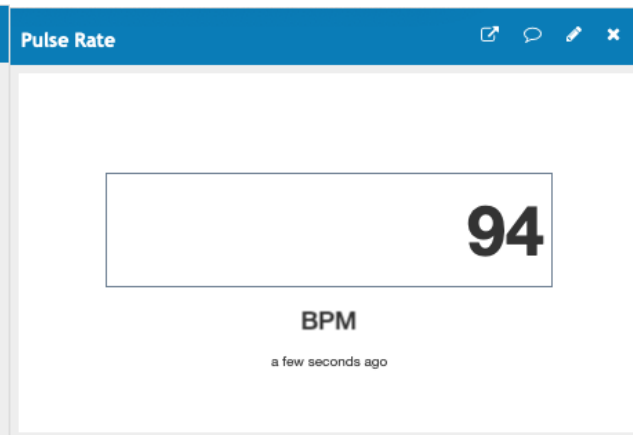
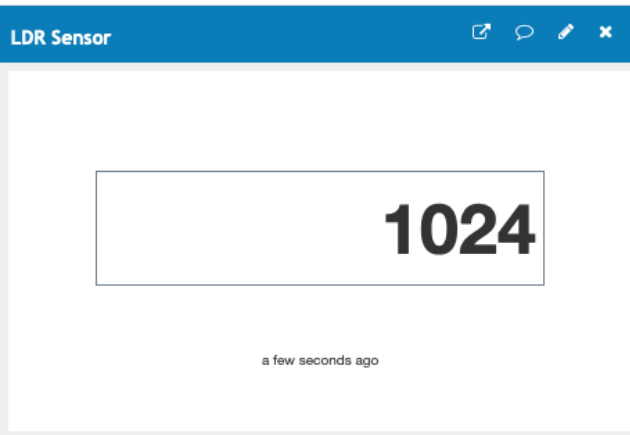
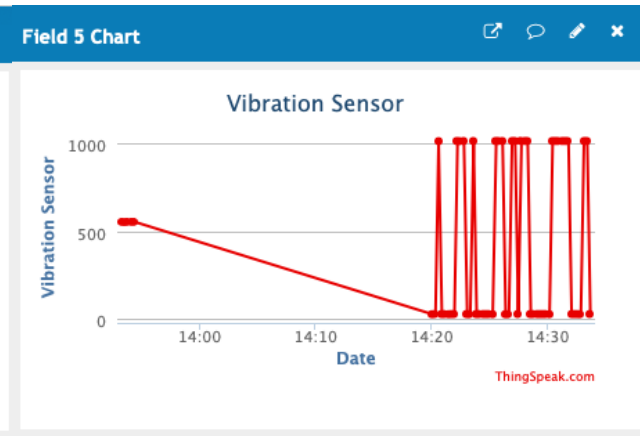
## Light Sensor



## Pulse Sensor



## Vibration Sensor



---

# Thanks!

