# Indian Institute of Information Technology, Bhopal

IoT Practical

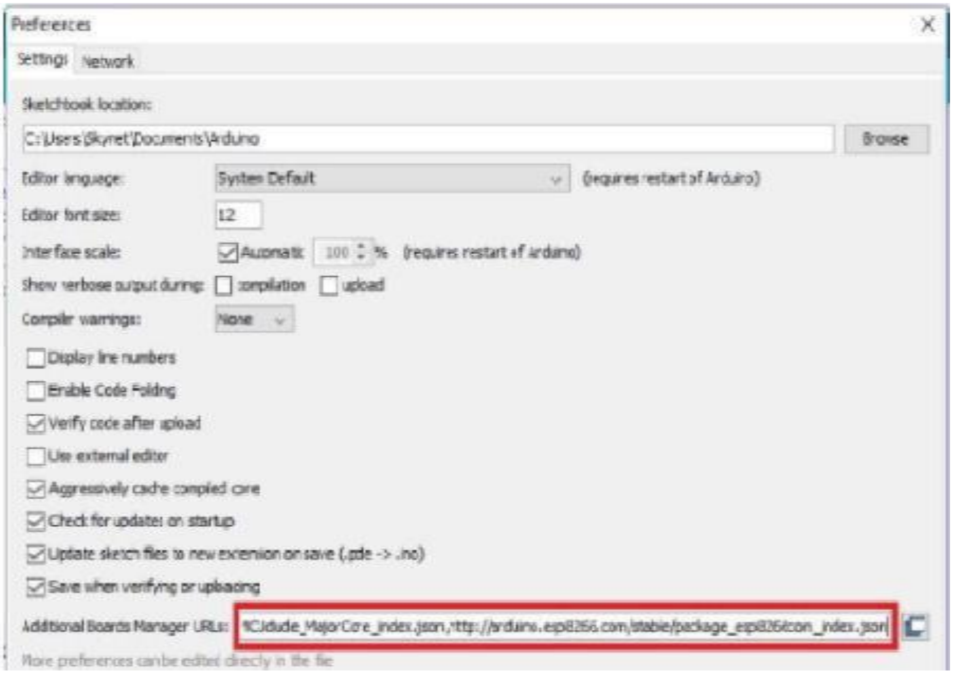
Submitted To: Dr. Nitesh Bharadwaj Exp:09

Submitted By: Anurag Soni(18U03037)
Rahul Bharadwaaj(18U03038)
Abhishek Maurya(18U03044)

AIM: To demonstrate, how to send data to things speak using ESP8266 and Arduino in Node-Red enviorment.

PROCEDURE:

1. Install ESP8266 board package to Arduino IDE
2. Copy this link htpp://arduino.esp8266.cpm/stable/package_esp68266com_index.json
3. Now open Arduino IDE and click on File>Preferences
4. A window will open like this:



Preferences

Settings Network

Sketchbook location: C:\Users\Skynet\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: □ compilation □ upload

Compiler warnings: None

□ Display line numbers

□ Enable Code Folding

☑ Verify code after upload

□ Use external editor

☑ Aggressively cache compiled core

☑ Check for updates on startup

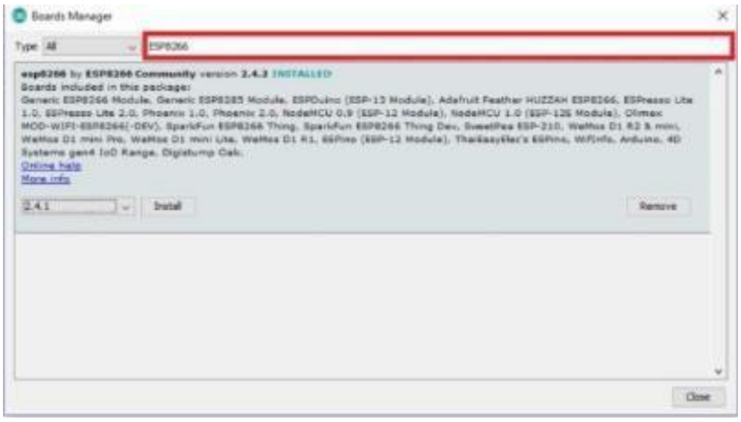
☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: %Cldude_MajorCore_index.json,http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

5. Paste the url on the box and click ok.
6. Now go to Tools>Board>Board Manager.
7. A new window will pop up.

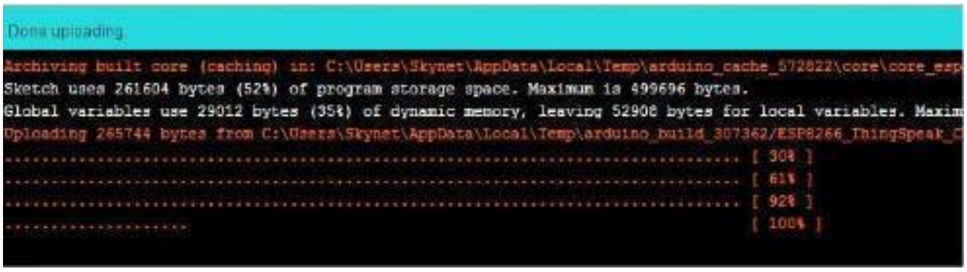


Boards Manager

Type All ESP8266

**esp8266** by **ESP8266 Community** version **2.4.2** INSTALLED
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SweetPea ESP-210, WeMos D1 R2 & mini, WeMos D1 mini Pro, WeMos D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak.
Online help
More info

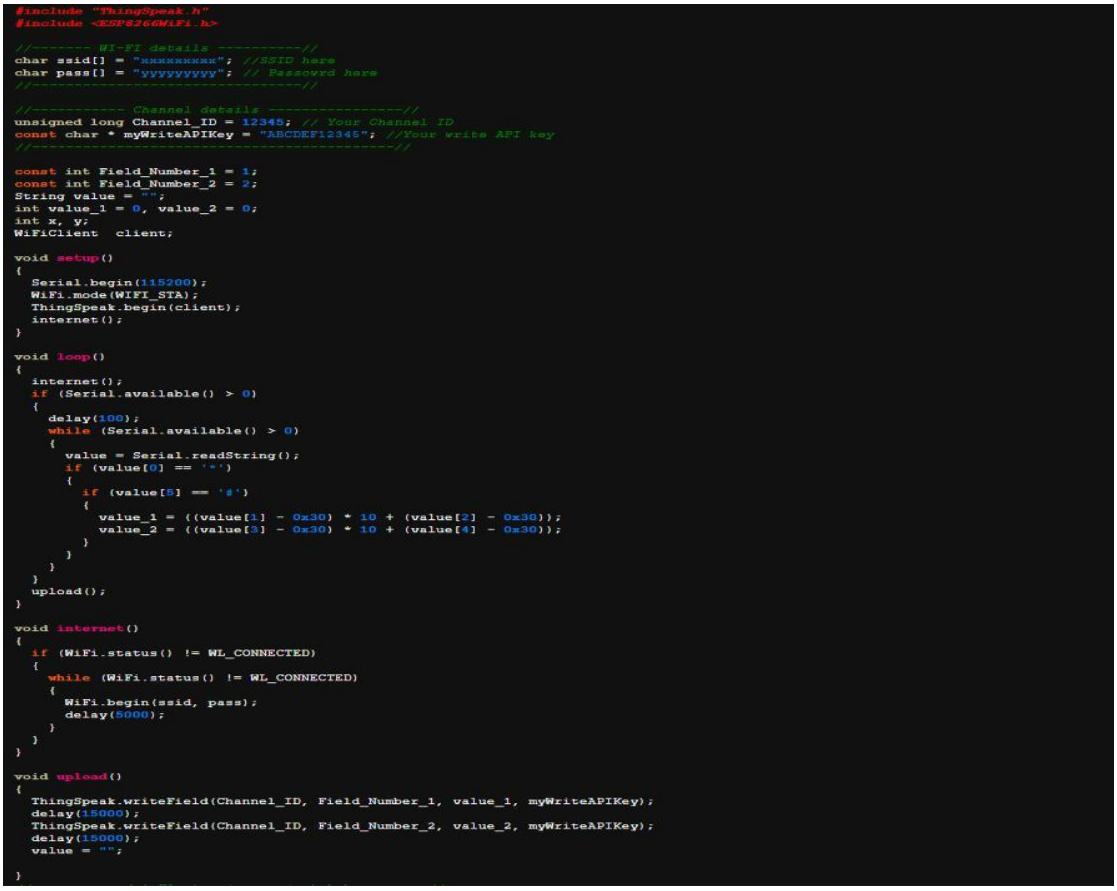
8. Type ESP8266" on the box as shown and you will get installation option select the latest version and click install.
9. Now the IDE will download the necessary packages and this could take more than 5 minutes to complete
10. Now go to Tools > Board > select "Generic ESP8266 Module".
11. Now copy the given below ESP8266 program code and paste it on to Arduino IDE software and insert your SSID and Password of your Wi-Fi hotspot and Copy the write API key" and "channelID" from your thingspeak account and insert it to the code.
12. **• Now press compile button (Green tick button. The compilation of code may take more** than couple of minutes so be patient. If the compilation failed, please check whether have you selected the "Generic ESP8266 Module' in the board option or not
13. After successful compilation of the code connect the Arduino to your PC and go to Tools > Port and make sure that correct COM port is selected
14. Now Press and hold the FLASH button and press RESET button momentarily once and leave the FLASH button. By doing this ESP8266 will get ready for a new program to be uploaded.
15. Now click upload the code will compile again and begin to upload and uploading could take 1-2 minutes.
16. Once the code is successfully uploaded, it looks something like this:



Done uploading

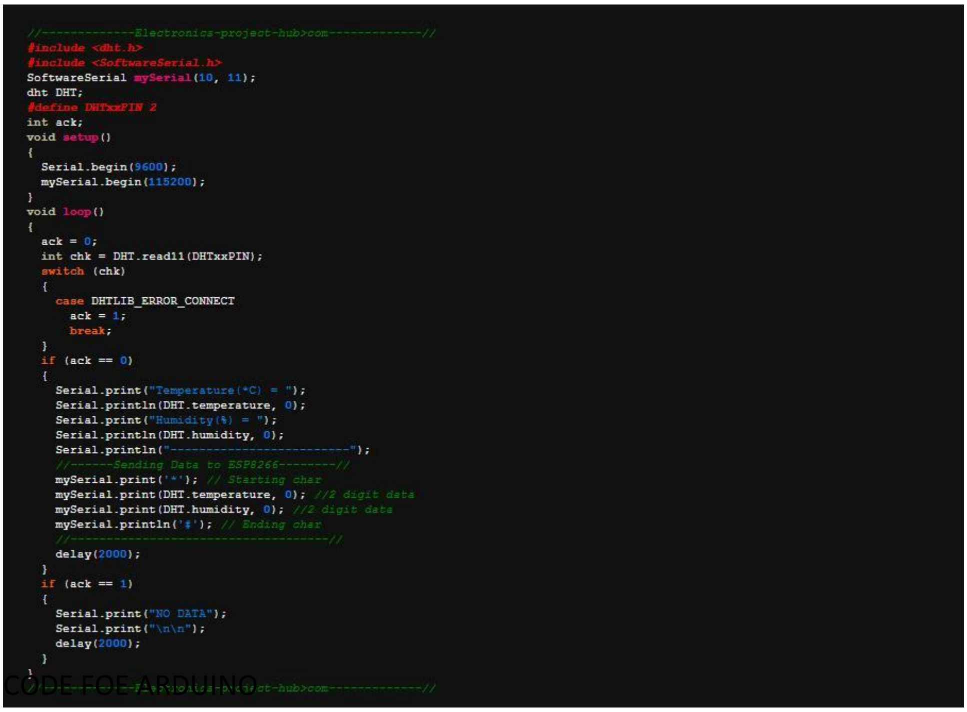
```
Archiving built core (caching) in: C:\Users\Skynet\AppData\Local\Temp\arduino_cache_572822\core\core_esp
Sketch uses 261604 bytes (52%) of program storage space. Maximum is 499696 bytes.
Global variables use 29012 bytes (35%) of dynamic memory, leaving 52908 bytes for local variables. Maxim
Uploading 265744 bytes from C:\Users\Skynet\AppData\Local\Temp\arduino_build_307362/ESP8266_ThingSpeak_C
.................................................................................. [ 30% ]
.................................................................................. [ 61% ]
.................................................................................. [ 92% ]
....................                                                               [ 100% ]
```

17. The orange-coloured text is not error or warning, but the uploading progress bar
18. Now the code is successfully uploaded to ESP8266 Module and now it can accept the data serially via its UART pins.



```
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>

//------- WI-FI details ----------//
char ssid[] = "xxxxxxxxxx"; //SSID here
char pass[] = "yyyyyyyyyy"; // Passowrd here
//-----------------------------------//

//----------- Channel details ----------------//
unsigned long Channel_ID = 12345; // Your Channel ID
const char * myWriteAPIKey = "ABCDEF12345"; //Your write API key
//-------------------------------------------//

const int Field_Number_1 = 1;
const int Field_Number_2 = 2;
String value = "";
int value_1 = 0, value_2 = 0;
int x, y;
WiFiClient  client;

void setup()
{
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);
  internet();
}

void loop()
{
  internet();
  if (Serial.available() > 0)
  {
    delay(100);
    while (Serial.available() > 0)
    {
      value = Serial.readString();
      if (value[0] == '*')
      {
        if (value[5] == '#')
        {
          value_1 = ((value[1] - 0x30) * 10 + (value[2] - 0x30));
          value_2 = ((value[3] - 0x30) * 10 + (value[4] - 0x30));
        }
      }
    }
  }
  upload();
}

void internet()
{
  if (WiFi.status() != WL_CONNECTED)
  {
    while (WiFi.status() != WL_CONNECTED)
    {
      WiFi.begin(ssid, pass);
      delay(5000);
    }
  }
}

void upload()
{
  ThingSpeak.writeField(Channel_ID, Field_Number_1, value_1, myWriteAPIKey);
  delay(15000);
  ThingSpeak.writeField(Channel_ID, Field_Number_2, value_2, myWriteAPIKey);
  delay(15000);
  value = "";

}
```

MAIN CODE FORESP8266



```
//------------Electronics-project-hub>com------------//
#include <dht.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11);
dht DHT;
#define DHTxxPIN 2
int ack;
void setup()
{
  Serial.begin(9600);
  mySerial.begin(115200);
}
void loop()
{
  ack = 0;
  int chk = DHT.read11(DHTxxPIN);
  switch (chk)
  {
    case DHTLIB_ERROR_CONNECT
      ack = 1;
      break;
  }
  if (ack == 0)
  {
    Serial.print("Temperature(*C) = ");
    Serial.println(DHT.temperature, 0);
    Serial.print("Humidity(%) = ");
    Serial.println(DHT.humidity, 0);
    Serial.println("-------------------------");
    //------Sending Data to ESP8266--------//
    mySerial.print('*'); // Starting char
    mySerial.print(DHT.temperature, 0); //2 digit data
    mySerial.print(DHT.humidity, 0); //2 digit data
    mySerial.println('#'); // Ending char
    //------------------------------------//
    delay(2000);
  }
  if (ack == 1)
  {
    Serial.print("NO DATA");
    Serial.print("\n\n");
    delay(2000);
  }
}
//------------Electronics-project-hub>com------------//
```

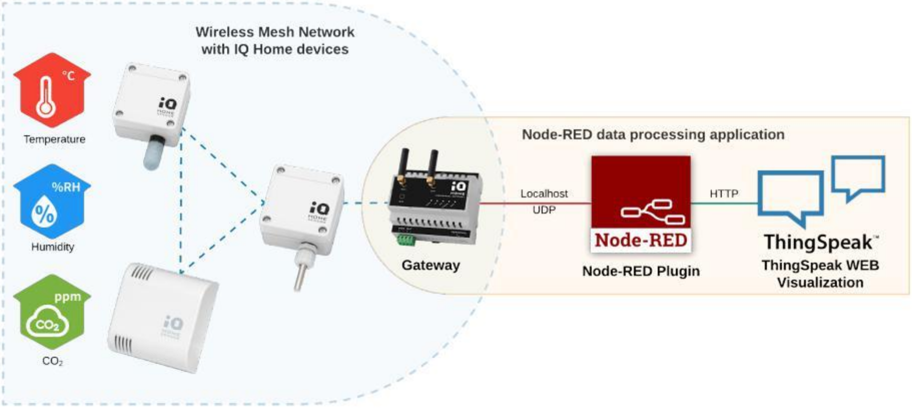
CODE FOE ARDUINO

Upload the above code to Arduino Uno board after inserting ATmega328P to the board. Don ’t forget to select the board back to the Arduino Uno, you can do this by going to Tools>Board>Arduino Uno, otherwise you will get compilation error.

# Uploading sensor data to ThingSpeak with Node-RED

The pre-installed Node-RED service and IQHome package on your Gateway will be used to periodically collect and upload your sensor data to ThingSpeak.

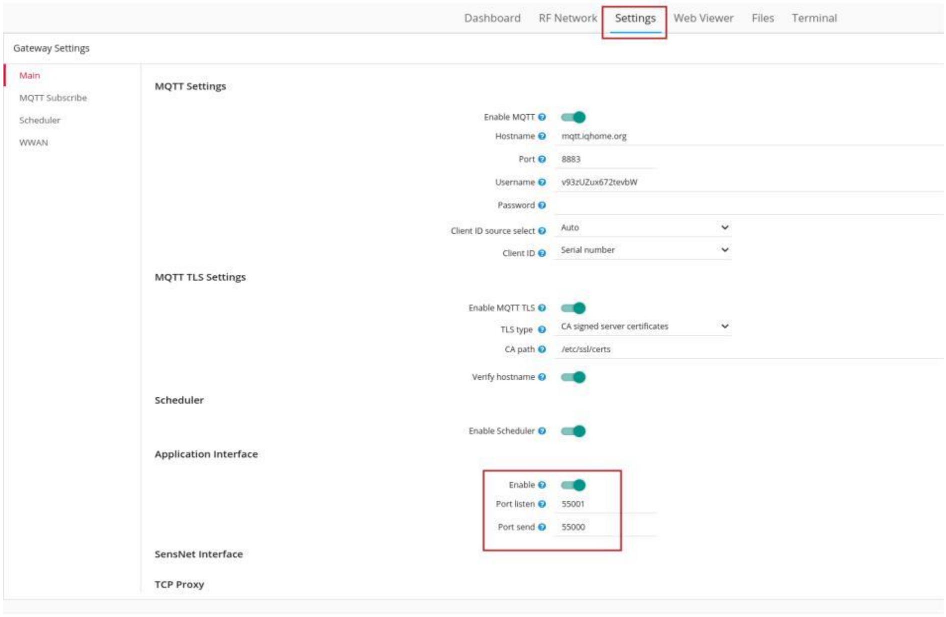
What you will need:



2 Enable the Application Interface

To connect to the Thingspeak service, first, we need to enable the Application Interface feature on the gateway.

Connect to the Gateway using the “Link It!” Software
Go to the “Settings” tab
Enable “Application Interface” and set a “Port Send” value (e.g. 55000)



Gateway Settings

MQTT Settings

Enable MQTT ●
Hostname mqtt.iqhome.org
Port 8883
Username v93zUZux672tevbW
Password
Client ID source select Auto
Client ID Serial number

MQTT TLS Settings

Enable MQTT TLS ●
TLS type CA signed server certificates
CA path /etc/ssl/certs
Verify hostname ●

Scheduler

Enable Scheduler ●

Application Interface

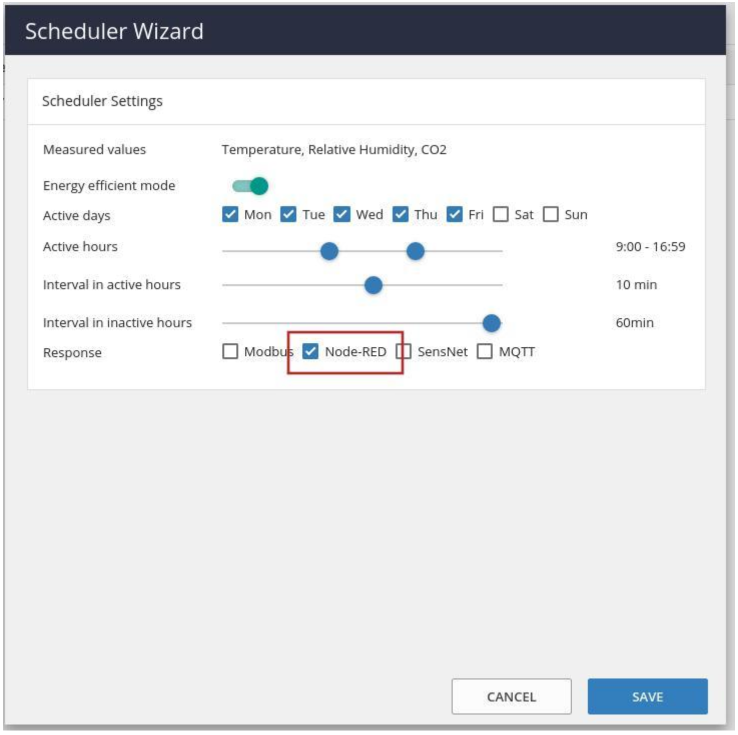
Enable ●
Port listen 55001
Port send 55000

SensNet Interface

TCP Proxy

Then enable the Node-RED scheduler for the sensors.

Open the “RF Network” tab
Switch to “Sensor Data”
Click on the clock icon in the top right corner labeled “Create Scheduler”
Enable the “Node-RED” Response option



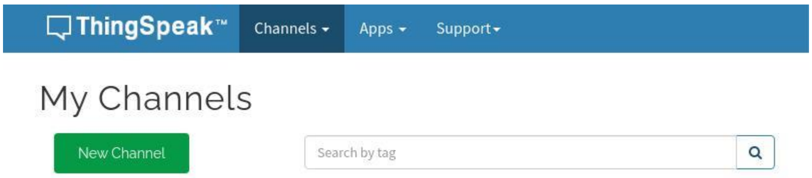
Scheduler Wizard

Scheduler Settings

| | | |
|---|---|---|
| Measured values | Temperature, Relative Humidity, CO2 | |
| Energy efficient mode | ● | |
| Active days | ☑ Mon ☑ Tue ☑ Wed ☑ Thu ☑ Fri □ Sat □ Sun | |
| Active hours | | 9:00 - 16:59 |
| Interval in active hours | | 10 min |
| Interval in inactive hours | | 60min |
| Response | □ Modbus ☑ Node-RED □ SensNet □ MQTT | |

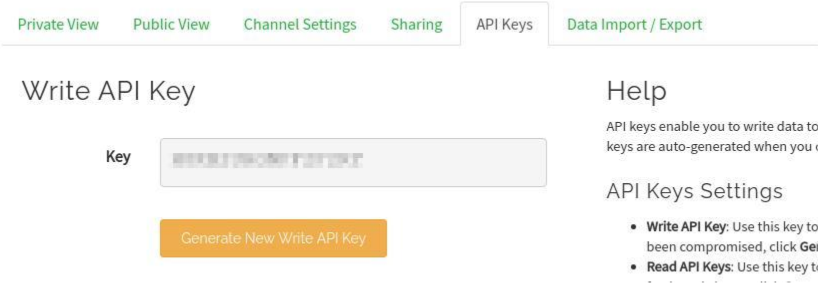
3 Configure your Thingspeak account

Open Thingspeak and log in with your existing Matlab account, or create a new account here After logging in, click on New Channel



Here we made two channels temperature and humidity .Click on Save Channel

This will create a default line graph for each of your fields.
Open the API Keys tab, and note down your Write API Key. It will be needed later in Node-RED.



Write API Key

Key [redacted]

Help

API keys enable you to write data to
keys are auto-generated when you c

API Keys Settings

- **Write API Key**: Use this key to
been compromised, click **Ge**
- **Read API Keys**: Use this key t

4 Set up a Node-RED network to forward the sensor data

Switch to the Node-RED tab.

The IQHome nodes can be found in the bottom of the panel on the left side of your screen.

Add an iqhome gateway node. If you changed the used port in the first step, you can set it here by double-clicking on the node

Add the sensor nodes corresponding to the sensors you are using in your IQHome network.

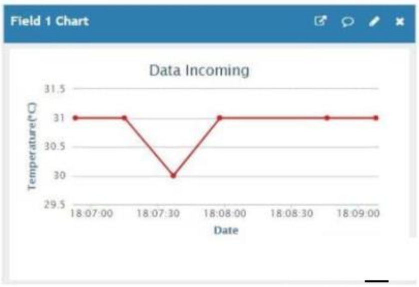
Set the device addresses corresponding to your sensor's addresses as seen , RF Network tab by double-clicking the sensor nodes.
We are using the default topic names generated by the sensor nodes, so you can leave the boxes under "Topics" empty.

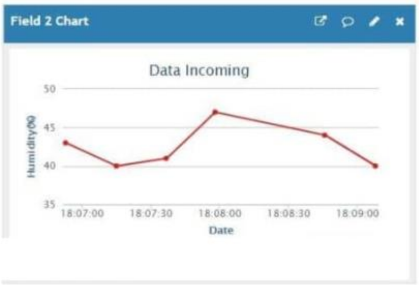
Add a Thingspeak node and connect it to the outputs of the sensor nodes.

Double click the Thingspeak node and enter your Thingspeak API Key. Then enter the same topic names as used in the sensor nodes (default {sensor type}-{device address} to the topic numbers used on the Thingspeak website.

On the Thingspeak website, we previously set the label tempature for Field 1. Thus we have to set the Topic 1 in the Thingspeak node to the topic of the sensor node: temperature-1.

Deploy your Node-RED network by clicking the Deploy button in the top right corner of your window





We will see the output on thingspeak website as above.