Hope Foundation's
Finolex Academy of Management & Technology, Ratnagiri
Department of MCA
MCALE232 Internet of Things Lab

Practical 12

Aim: - To interface Temperature sensor and DHT11 sensor, with Arduino using NodeMCU ESP8266 Module and write a program to send sensor data to the cloud using ThingSpeak.

Components Required:

NodeMCU, Bread Board, LM35 Temperature Sensor, DHT11 Temperature and Humidity sensor, Connecting wires.

Software Components

Arduino IDE

Theory:

NodeMCU ESP8266 Wifi Module is an open-source Lua based firmware and development board specially targeted for **IoT based Applications**. It includes firmware that runs on the **ESP8266 Wi-Fi SoC** from **Espressif Systems**, and hardware which is based on the **ESP-12 module**.

Nodemcu ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB

- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play

Small Sized module to fit smartly inside your IoT projects.

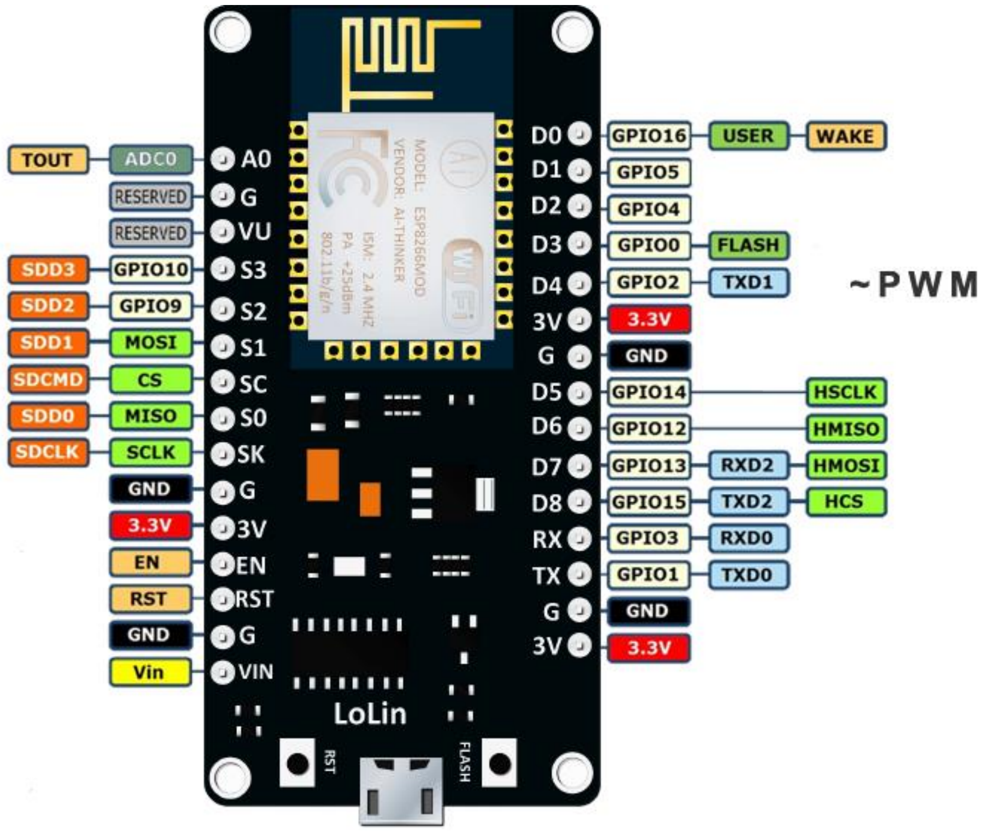
Nodemcu ESP8266 Pinout:

For practical purposes **ESP8266 NodeMCU** V2 and V3 boards present identical pinouts.

While working on the NodeMCU based projects we are interested in the following pins.

- Power pins (3.3 V).
- Ground pins (GND).
- Analog pins (A0).
- Digital pins (D0 – D8, SD2, SD3, RX and TX – GPIO XX)

Most ESP8266 NodeMCU boards have one input voltage pin (Vin), three power pins (3.3v), four ground pins (GND), one analog pin (A0) and several digital pins (GPIO XX).



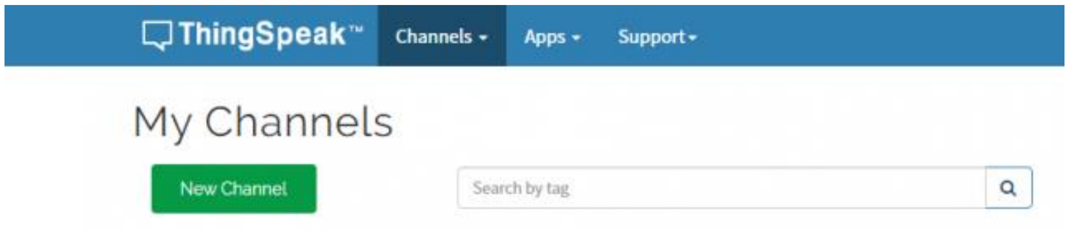
Applications of Nodemcu

- Prototyping of IoT devices.
- Low power battery operated applications.
- Network projects.
- Projects requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities.

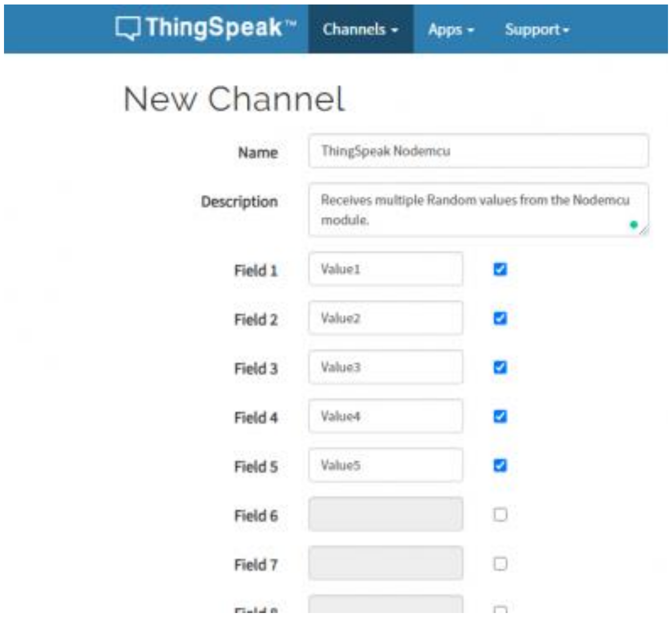
Implementation:

Setting ThingSpeak & Getting API Key:

- Go to https://thingspeak.com/ and create an account if you do not have one. The registration process on the ThingSpeak IoT platform is very simple. After your account is ready.
- Login to your account.



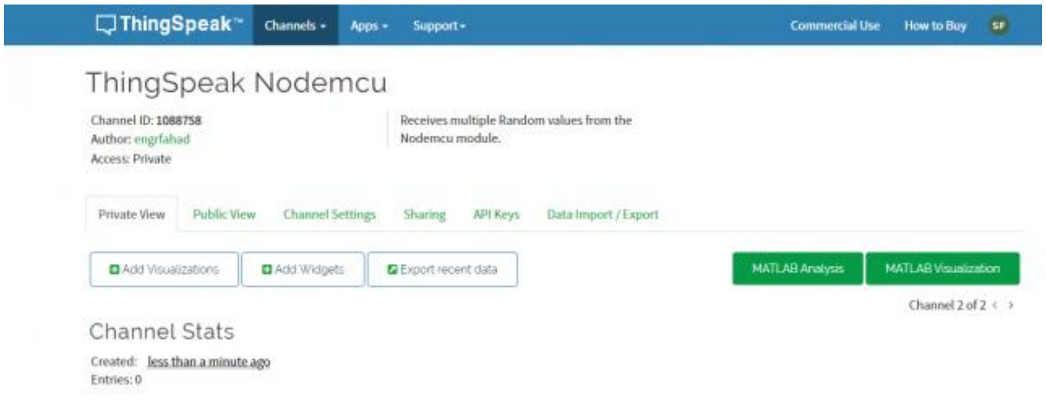
Click on the Channels menu and then click on the New Channel. This will open a new page as you can see in the picture below.



Name	ThingSpeak Nodemcu	
Description	Receives multiple Random values from the Nodemcu module.	
Field 1	Value1	☑
Field 2	Value2	☑
Field 3	Value3	☑
Field 4	Value4	☑
Field 5	Value5	☑
Field 6		☐
Field 7		☐

We selected the name as ThingSpeak Nodemcu and followed by the description. To activate a field, simply click on the check box, and you can assign different names to the fields, we selected Value1 to Value5.

Scroll down, check the show status check box, and click on the Save Channel Button. So, after we click on the Save button, a new page will open as we can see in the picture below.



ThingSpeak Nodemcu

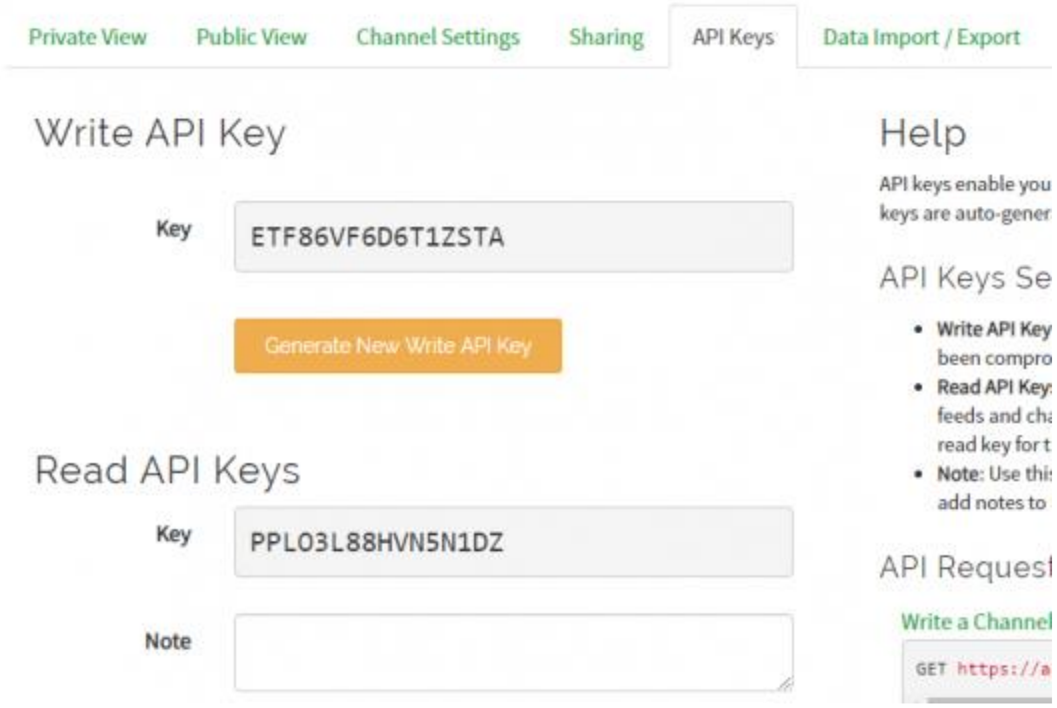
Channel ID: 1088758
Author: engrfahad
Access: Private

Receives multiple Random values from the Nodemcu module.

Channel Stats

Created: less than a minute ago
Entries: 0

As you can see the channel name at the top "ThingSpeak Nodemcu". The Channel ID:1088758, and other things. All the selected fields are automatically added, if we scroll down the page. To Copy the API Keys, click on the API Keys,



Write API Key

Key ETF86VF6D6T1ZSTA

Read API Keys

Key PPLO3L88HVN5N1DZ

Help

API keys enable you
keys are auto-gener

API Keys Se

- Write API Key been compro
- Read API Key: feeds and ch read key for t
- Note: Use thi add notes to

API Request

Write a Channel

GET https://a

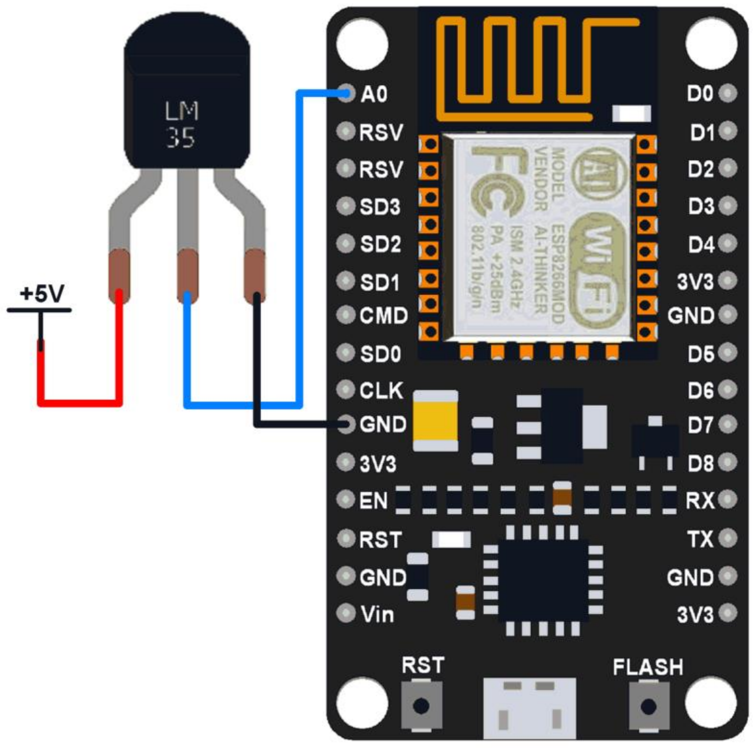
You can always generate new keys. For now, we will only be using the Channel ID and the Write API key, as we will only be sending data from the Nodemcu ESP8266 to the ThingSpeak IoT Platform.

The **circuit connections** are made as follows:

Pin 1 of the LM35 goes into **+3v** of the NodeMCU.

Pin 2 of the LM35 goes into Analog Pin **A0** of the NodeMCU.

Pin 3 of the LM35 goes into Ground Pin (**GND**) of the NodeMCU.

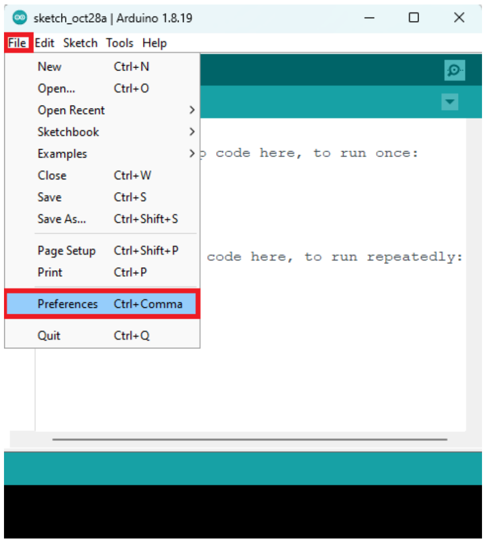


There are many different ways to program the ESP8266 using different programming languages: Arduino C/C++ using the Arduino core for the ESP32, Micropython, LUA, and others.

Our preferred method is programming the ESP8266 using the “Arduino programming language” with Arduino IDE. To be able to program the ESP8266 NodeMCU using Arduino IDE, you need to add support for the ESP8266 boards.

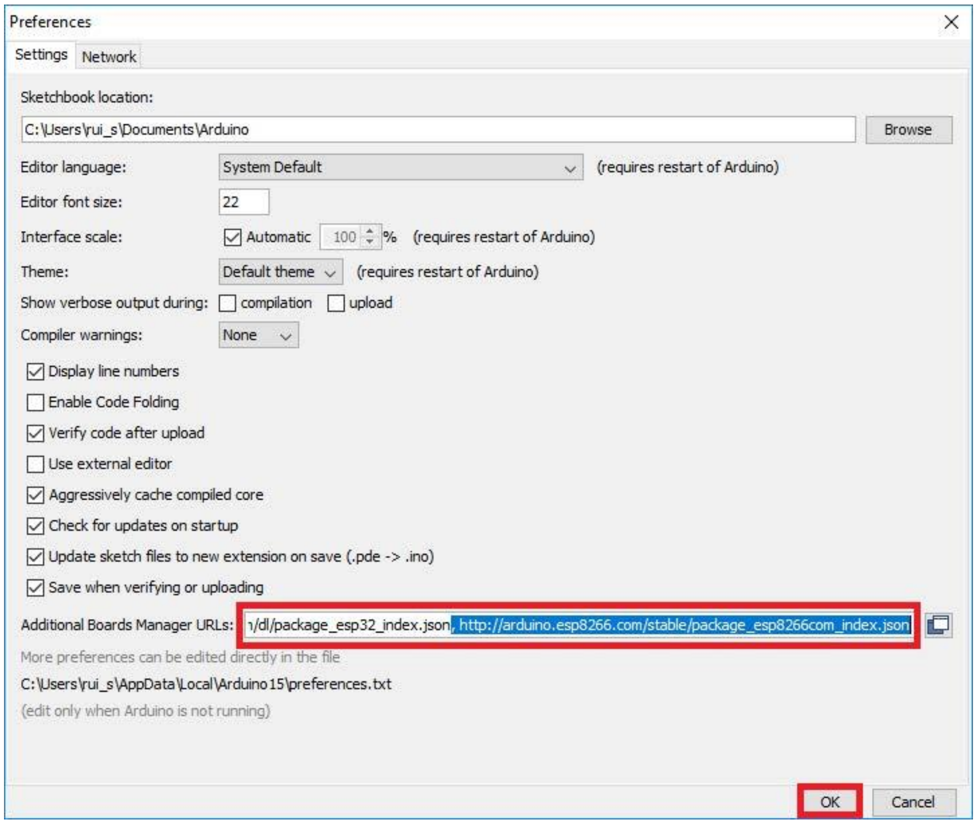
Follow the next steps:

1. Go to **File** > **Preferences**.



```
p code here, to run once:

code here, to run repeatedly:
```

2. Enter the following into the "*Additional Board Manager URLs*" field. https://arduino.esp8266.com/stable/package_esp8266com_index.jsonThen, click the "**OK**" button.



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

Editor language: System Default (requires restart of Arduino)

Editor font size: 22

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

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

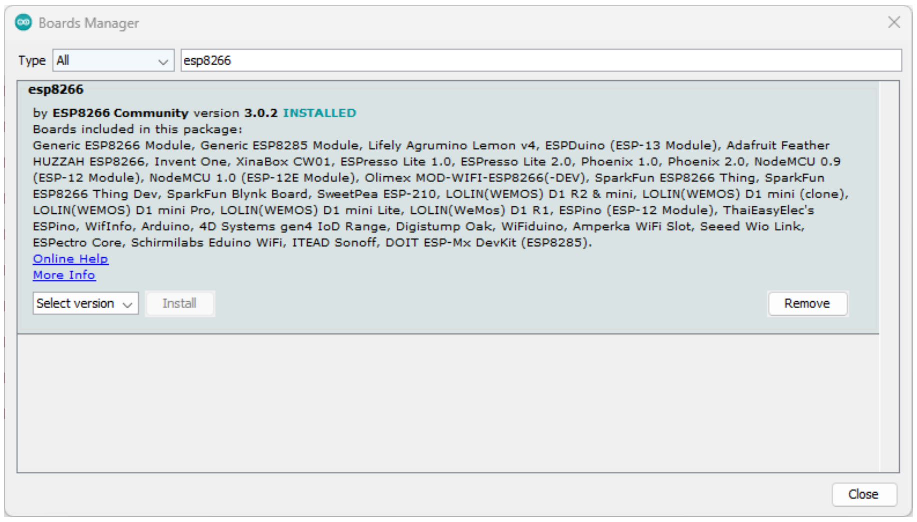
☑ Display line numbers
☐ Enable Code Folding
☑ Verify code after upload
☐ Use external editor
☑ Aggressively cache compiled core
☑ Check for updates on startup
☑ Update sketch files to new extension on save (.pde -> .ino)
☑ Save when verifying or uploading

Additional Boards Manager URLs: ı/dl/package_esp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file
C:\Users\rui_s\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

3. Open the **Boards Manager**. Go to **Tools** > **Board** >**Boards Manager…**

4. Search for **ESP8266** and install the “**ESP8266 by ESP8266 Community** “. That’s it. It will be installed after a few seconds.



Boards Manager

Type All esp8266

esp8266

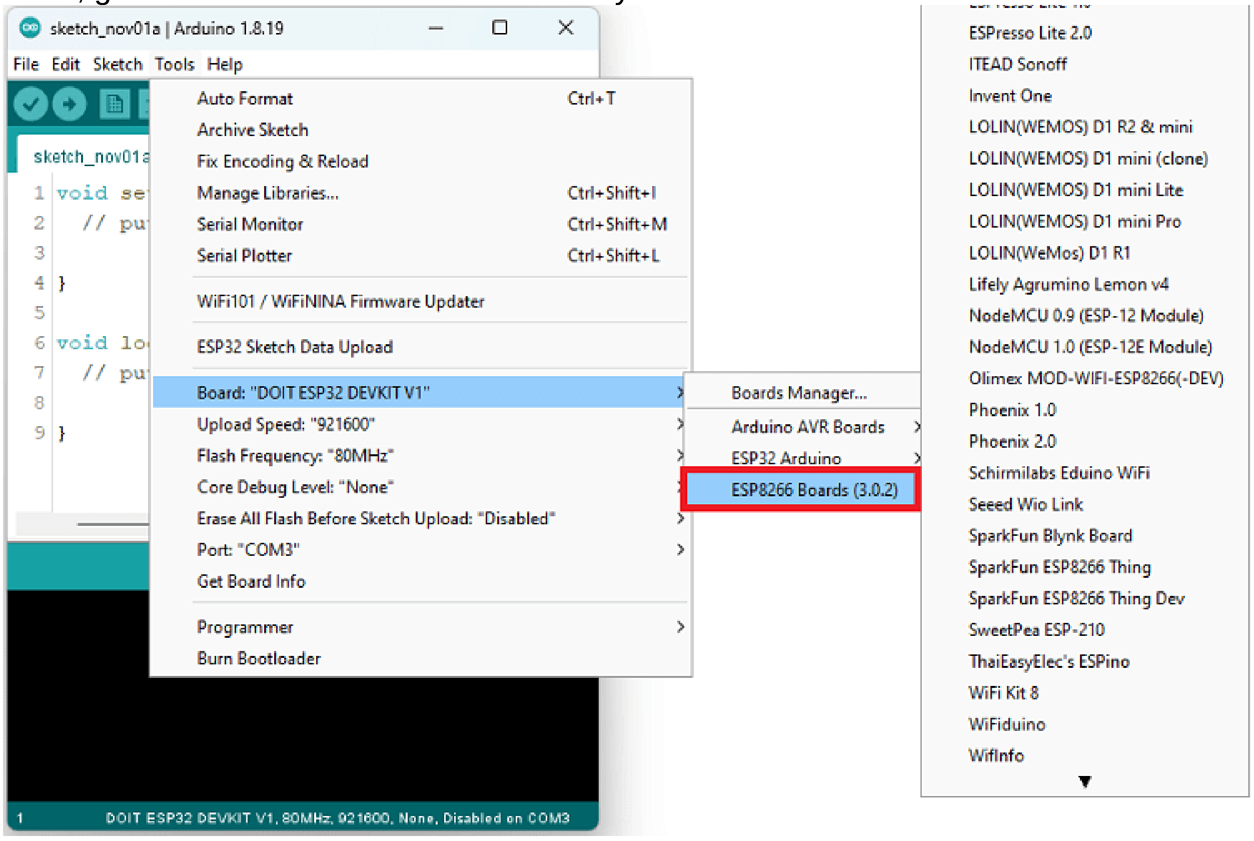
by **ESP8266 Community** version **3.0.2** INSTALLED
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 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, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).
Online Help
More Info

Select version Install Remove

Close

After this, restart your Arduino IDE.

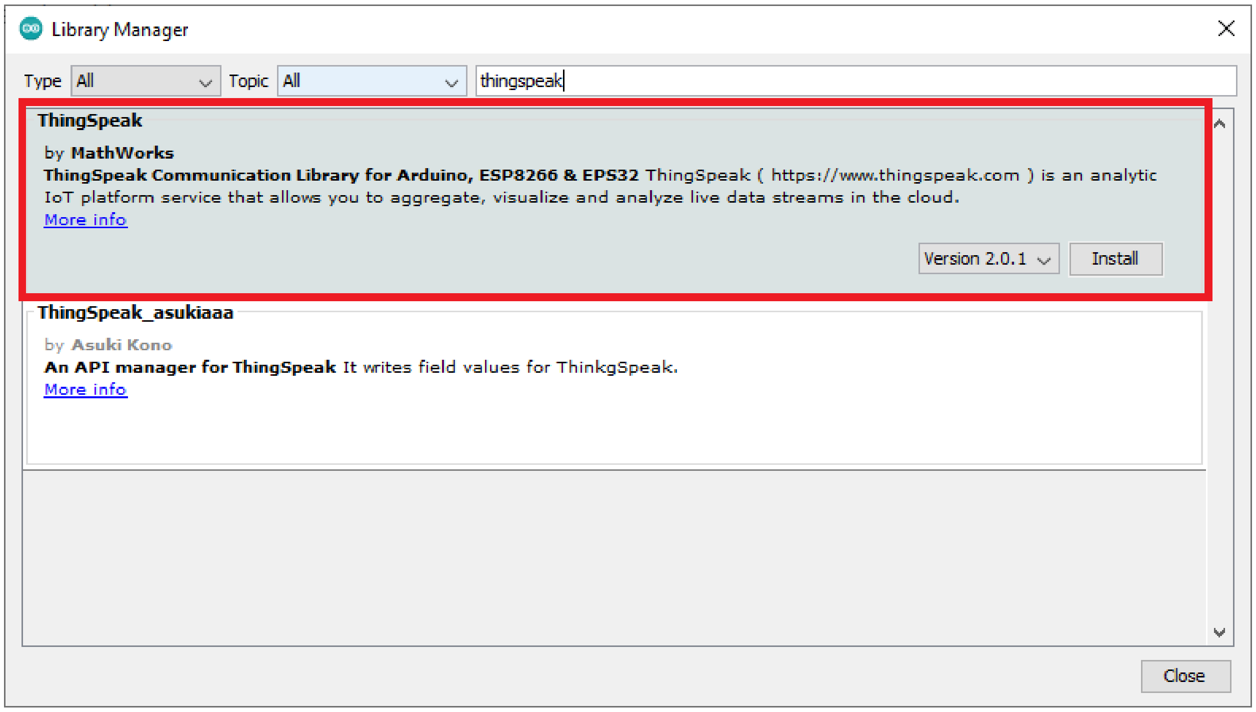
Then, go to **Tools** > **Board** and check that you have ESP8266 boards available.



Now, you’re ready to start programming your ESP8266 using Arduino IDE.

Installing the ThingSpeak Library

To send sensor readings to ThingSpeak, we'll use the thingspeak-arduino library. You can install this library through the Arduino Library Manager. Go to Sketch **>** Include Library **>** Manage Libraries… **and search for "**ThingSpeak**" in the Library Manager. Install the ThingSpeak library by MathWorks.**



ThingSpeak
by **MathWorks**
ThingSpeak Communication Library for Arduino, ESP8266 & EPS32 ThingSpeak (https://www.thingspeak.com) is an analytic IoT platform service that allows you to aggregate, visualize and analyze live data streams in the cloud.
More info

ThingSpeak_asukiaaa
by **Asuki Kono**
An API manager for ThingSpeak It writes field values for ThinkgSpeak.
More info

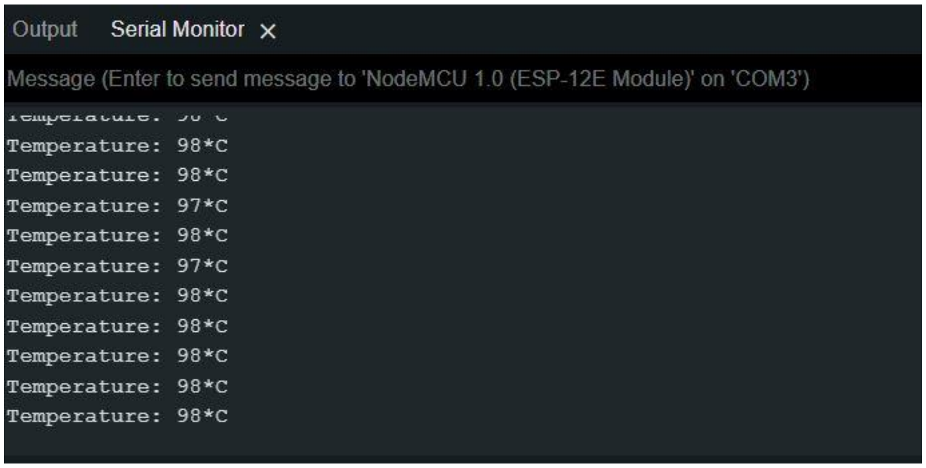
Now write the sketch for sending the data from the temperature sensor to the Things Speak Cloud and upload it on the NodeMCU using the Arduino IDE.

Code :

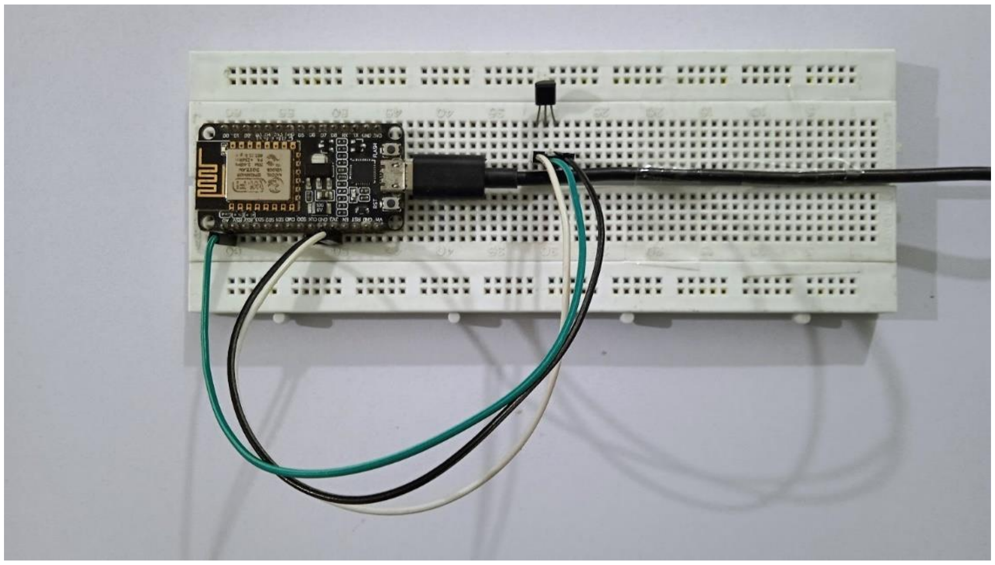
```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ThingSpeak.h>

const char* ssid = "MCAwifi";   // Your Network SSID
const char* password = "MCA12345";       // Your Network Password

int val;
int pin = A0; // LM35 Pin Connected at A0 Pin

WiFiClient client;

unsigned long myChannelNumber = 2157399; //Your Channel Number (Without Brackets)
const char * myWriteAPIKey = "Q2N0JZCHTPE64P90"; //Your Write API Key
```

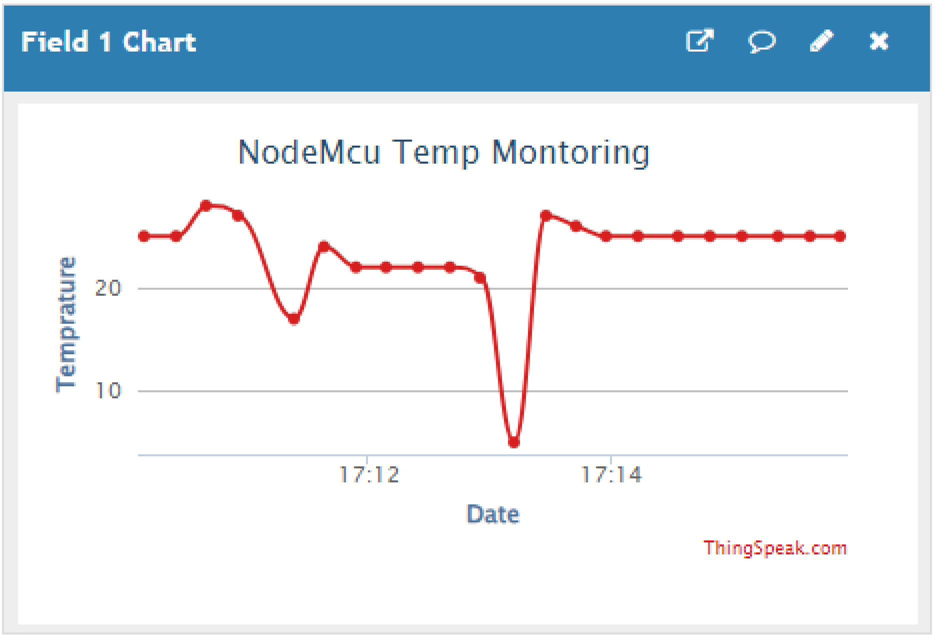
```
void setup()
{
  Serial.begin(9600);
  delay(10);
  // Connect to WiFi network
  WiFi.begin(ssid, password);
  ThingSpeak.begin(client);
}

void loop()
{
  val = analogRead(pin)*0.322265; // Read Analog values and Store in val variable
  Serial.print("Temperature: ");
  Serial.print(val);                 // Print on Serial Monitor
  Serial.println("*C");
  delay(1000);
  ThingSpeak.writeField(myChannelNumber, 1,val, myWriteAPIKey); //Update in ThingSpeak
  delay(100);
}
```

Output :


Output
Serial Monitor
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM3')

```
Temperature: 98*C
Temperature: 98*C
Temperature: 97*C
Temperature: 98*C
Temperature: 97*C
Temperature: 98*C
Temperature: 98*C
Temperature: 98*C
Temperature: 98*C
Temperature: 98*C
```





Conclusion: Thus we studied interfacing Temperature sensor and DHT11 sensor, with Arduino using NodeMCU ESP8266 Module and sending the sensor data to the cloud using ThingSpeak.