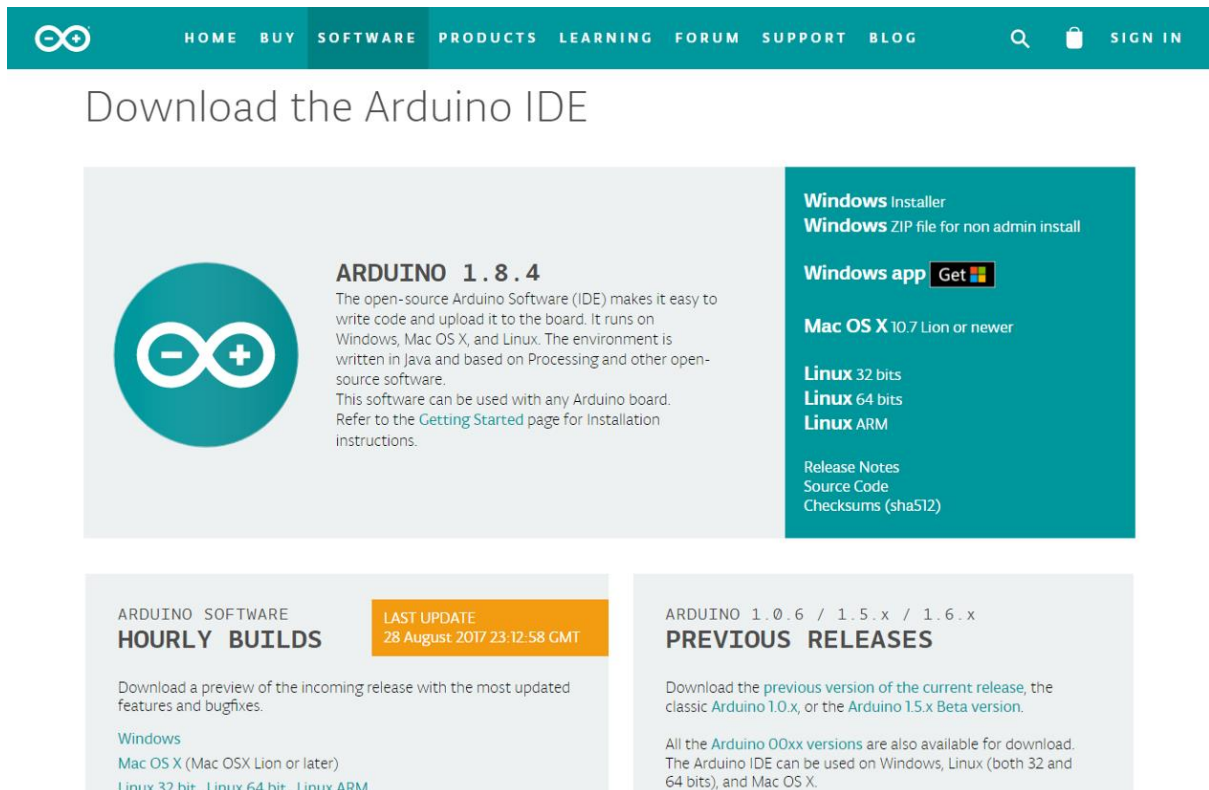


## คู่มือการใช้งาน

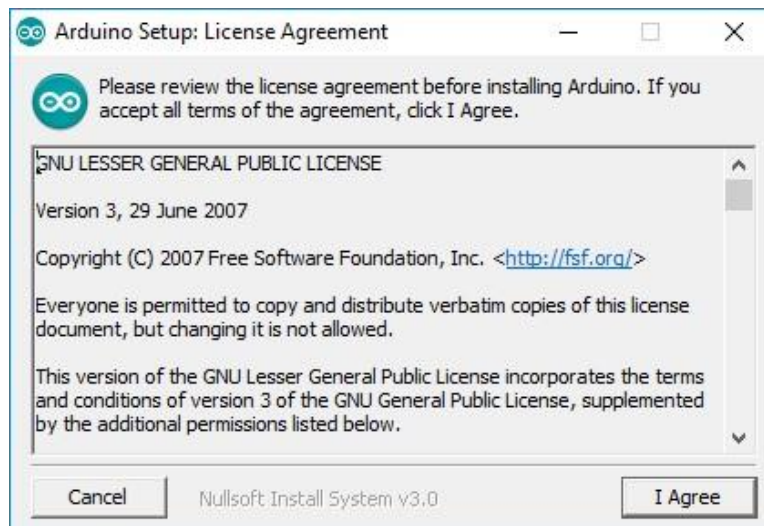
### 1. การติดตั้ง Arduino IDE

- สามารถดาวน์โหลดโปรแกรมได้ที่ <https://www.arduino.cc/en/Main/Software> โดยรองรับระบบปฏิบัติการ Windows, Mac OS, Linux ( ในการทำอุปกรณ์นี้ได้เลือกใช้เป็นระบบปฏิบัติการ Window 10 64 bit )

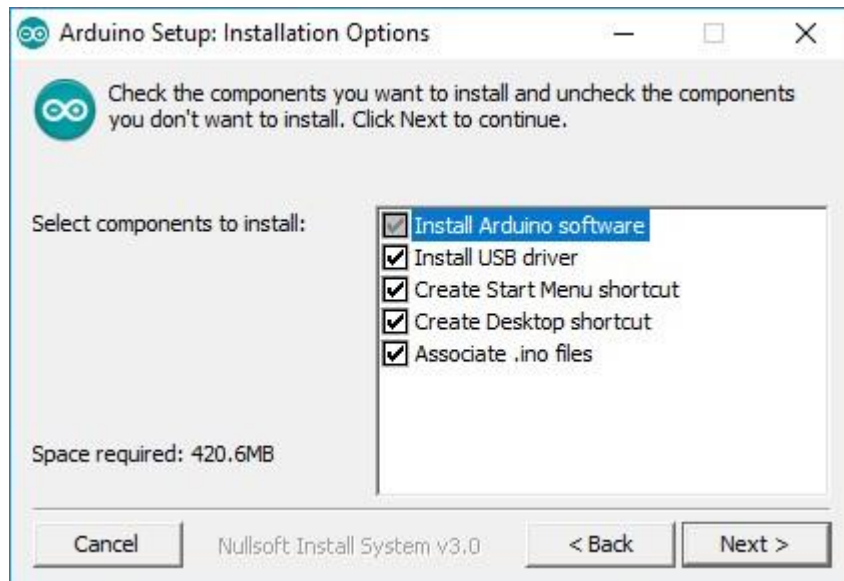


รูปที่ 1 หน้าเว็บไซต์สำหรับการดาวน์โหลด Arduino IDE

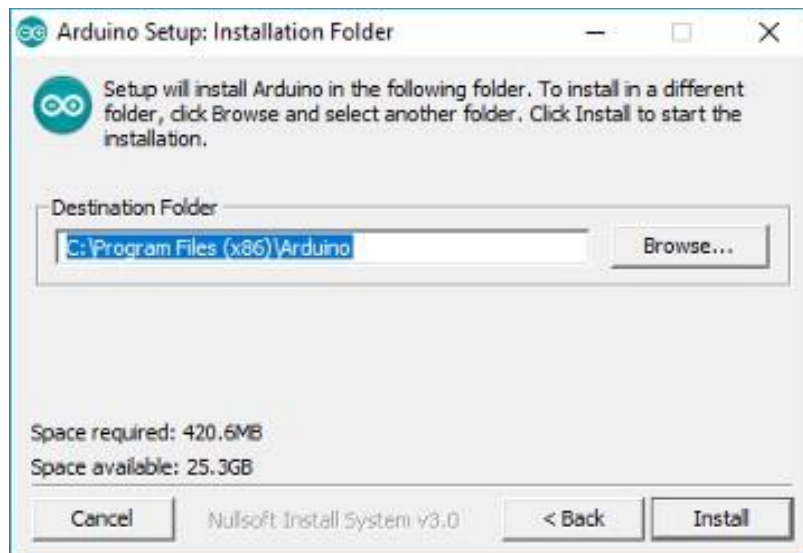
- ทำการติดตั้งโปรแกรมตามรูปที่ 2 – 5



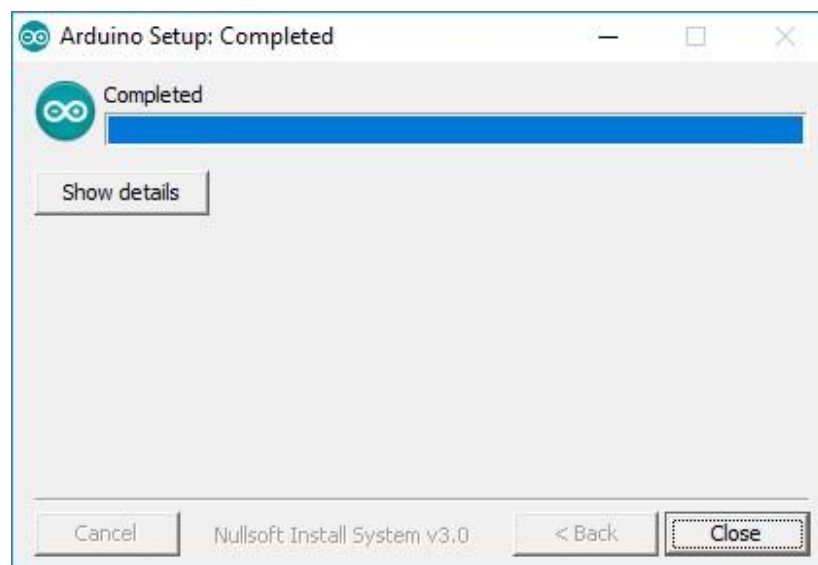
รูปที่ 2 กด I Agree เพื่อไปยังหน้าถัดไป



รูปที่ 3 เตรียมพื้นที่ว่างภายในหน่วยความจำให้เพียงพอแล้วกด Next



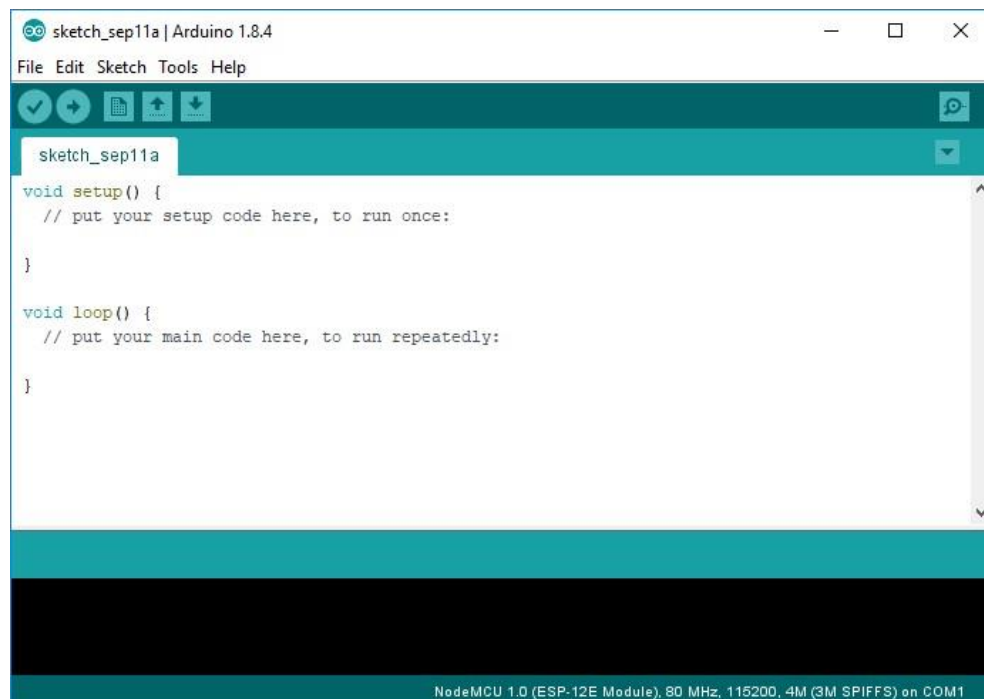
รูปที่ 4 เลือกโฟลเดอร์ที่ต้องการจะบันทึกไว้แล้วกด Install



รูปที่ 5 รอจนขึ้น Complete ตามรูปเป็นอันเสร็จสิ้น

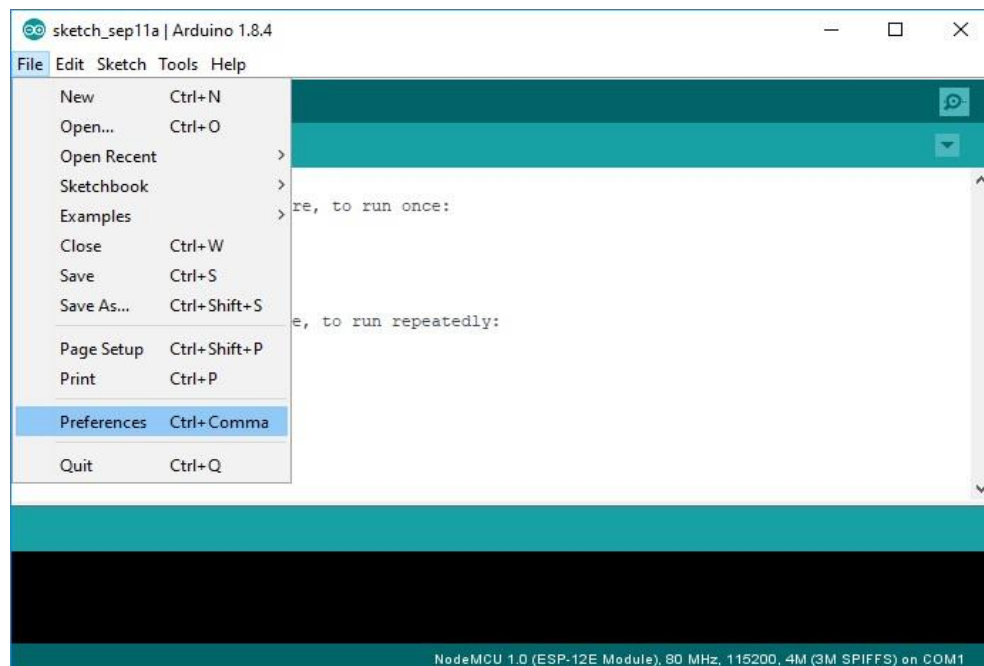
## 2. การติดตั้ง ESP8266 ลงใน Arduino IDE

- ทำการเปิดหน้าต่างโปรแกรม Arduino IDE ขึ้นมาตามรูปที่ 6

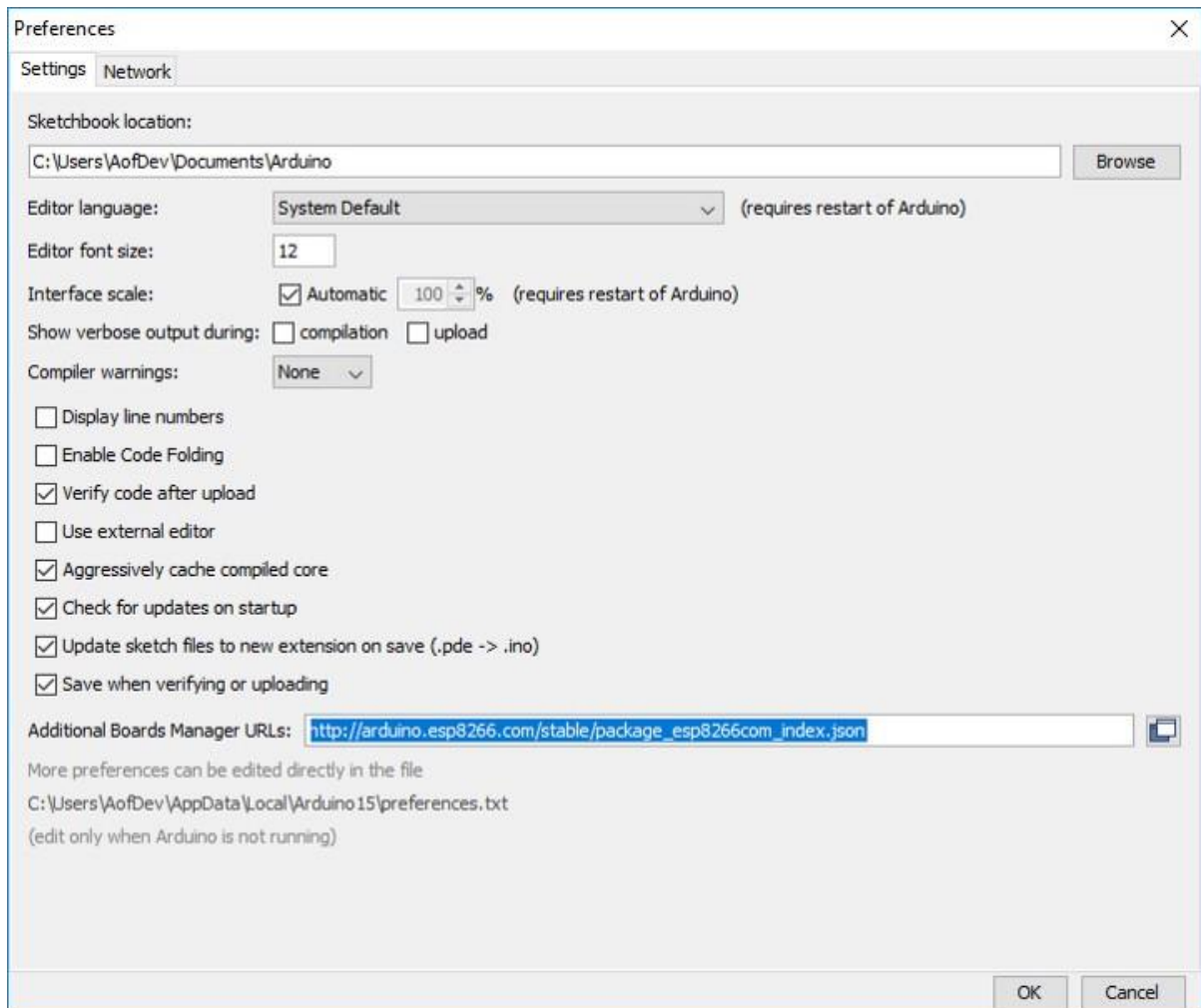


รูปที่ 6 หน้าต่างเริ่มต้นโปรแกรม Arduino IDE

- กดที่ File เลือก Performance ตามรูปที่ 7 และจะเจอหน้าต่างตามรูปที่ 8



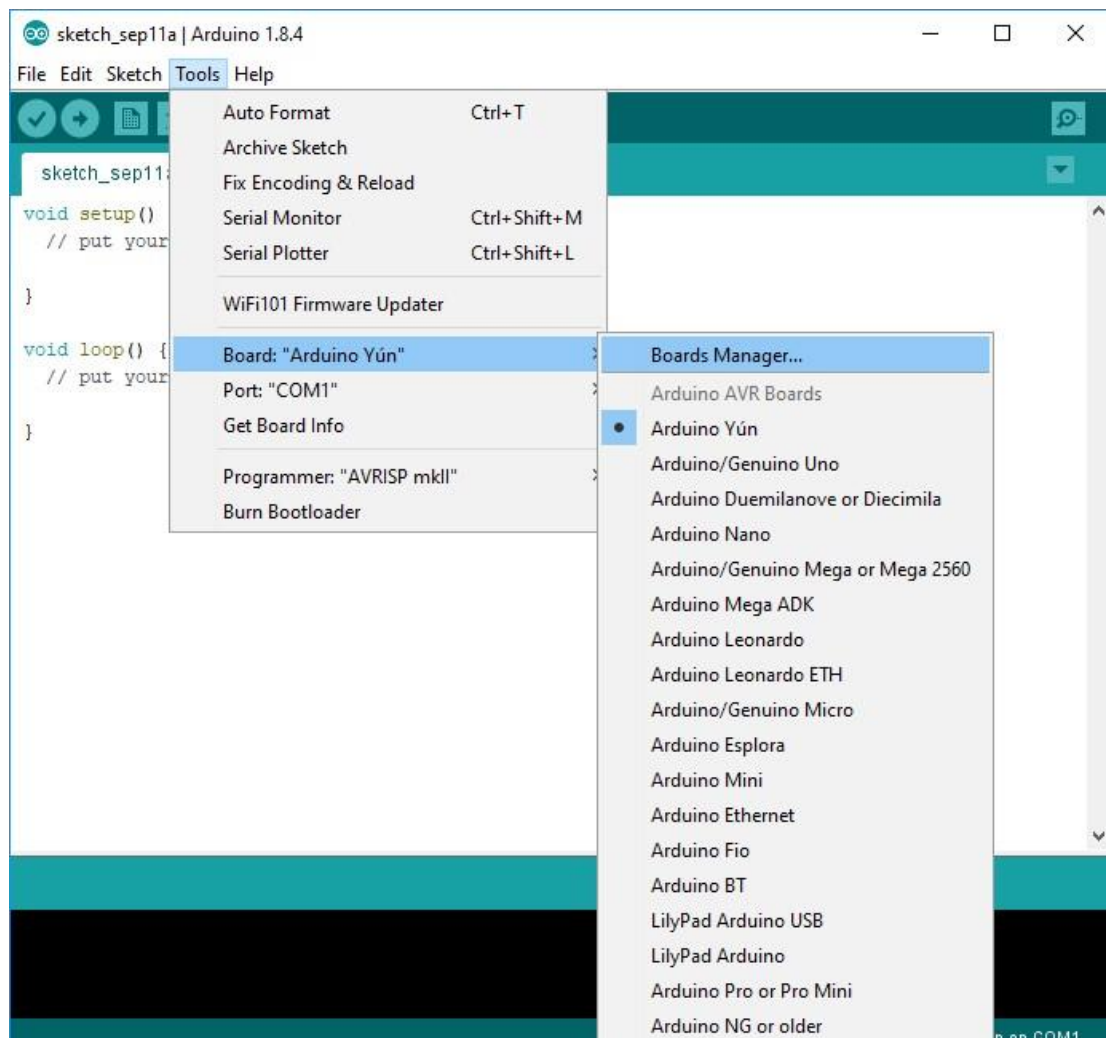
รูปที่ 7 เลือก File > Performance



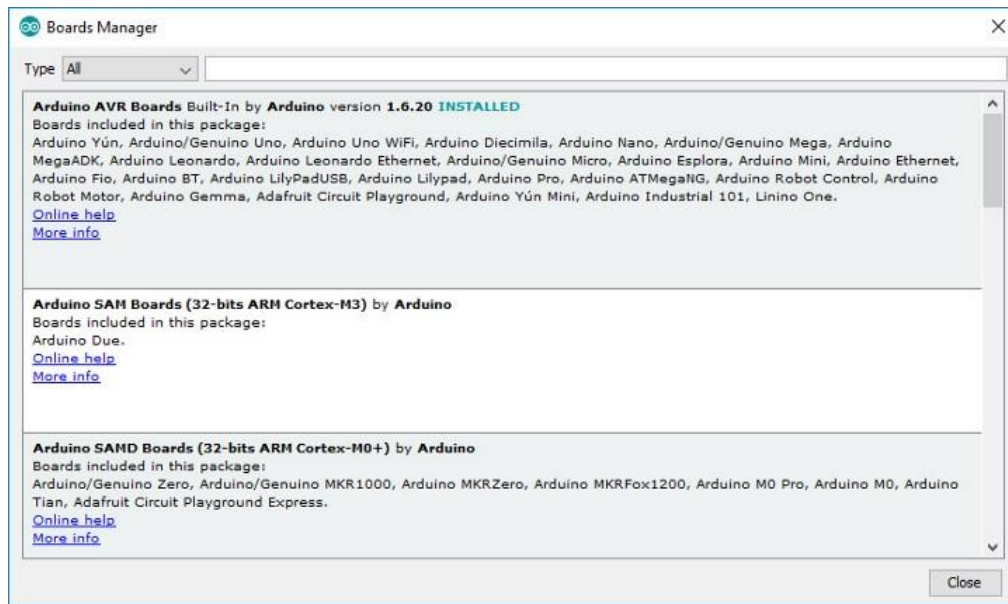
รูปที่ 8 หน้าต่าง Performance

- ให้นำ [http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json) ไปกรอกไว้ในช่อง Additional Boards Manager URLs: จากนั้น กด OK เพื่อบันทึกการตั้งค่าและปิดหน้าต่างนี้ไป

- กดเลือกไปที่ Tool เลือกไปที่ Board และเลือก Board Manager... ตามรูปที่ 9 และจะเห็นหน้าต่างการดาวน์โหลด Library ต่างๆ ตามรูปที่ 10

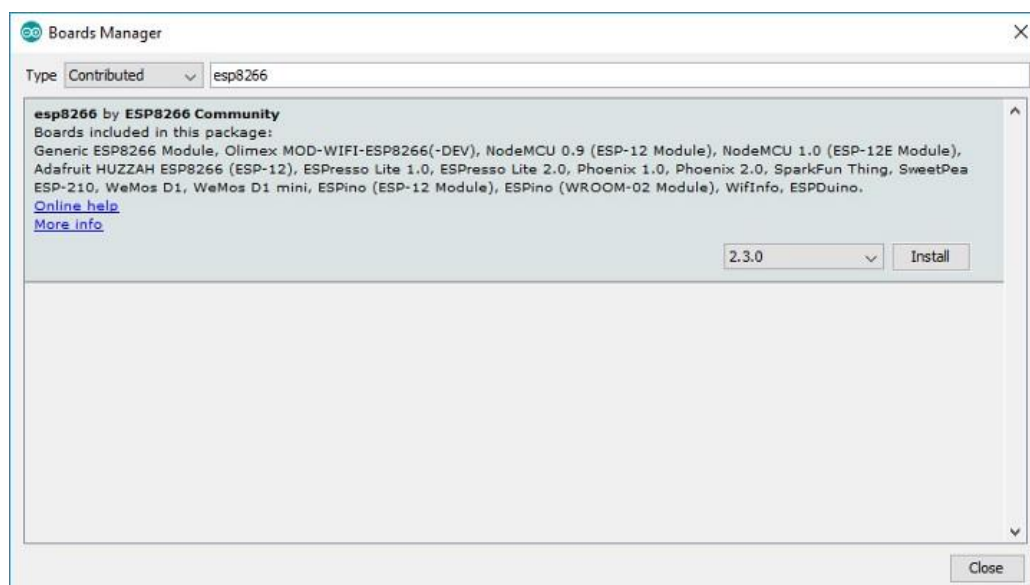


รูปที่ 9 เลือก Tool > Board > Boards Manager...



รูปที่ 10 หน้าต่าง Board Manager

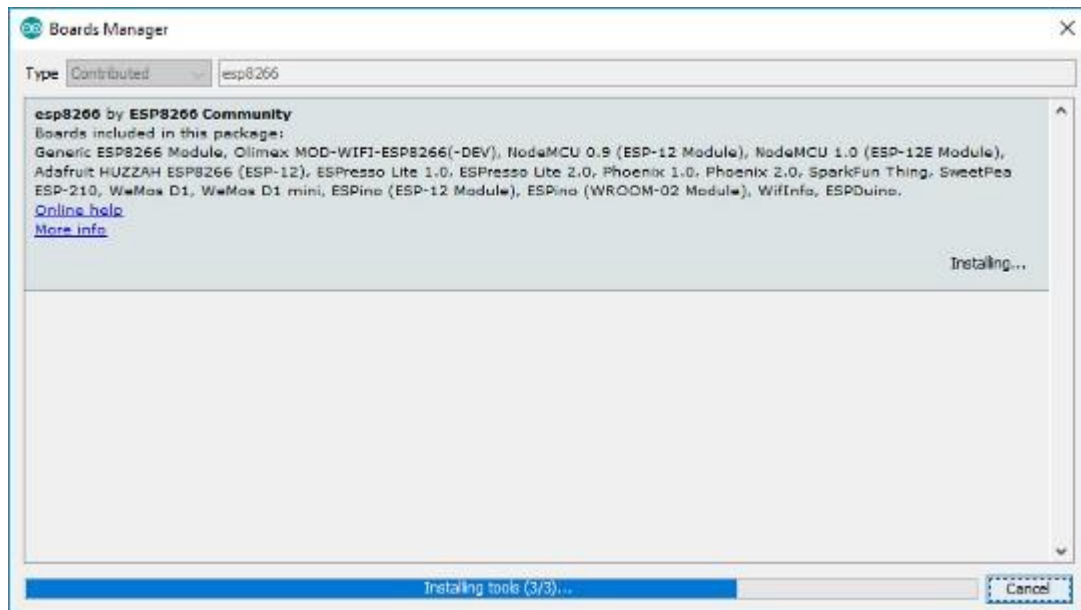
- ให้เลือก Type เป็น “Contributed” หรือจะค้นหาคำว่า “esp8266” ก็จะเจอตามรูปที่ 11



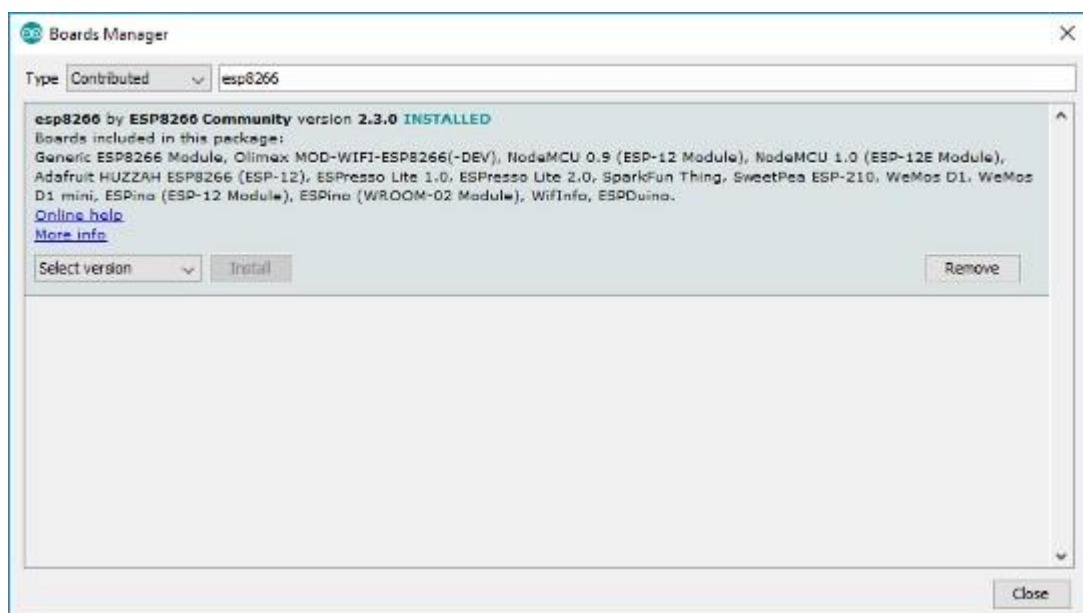
รูปที่ 11 เมื่อเลือก Type เป็น Contributed และค้นหาคำว่า esp8266



- ทำการกด Install ที่หัวข้อ esp8266 เพื่อทำการติดตั้ง Library ESP8266 ตามรูปที่ 12 และรูปที่ 13



รูปที่ 12 หลังจากกด Install จะขึ้นแถบดาวน์โหลดแสดงให้เห็น

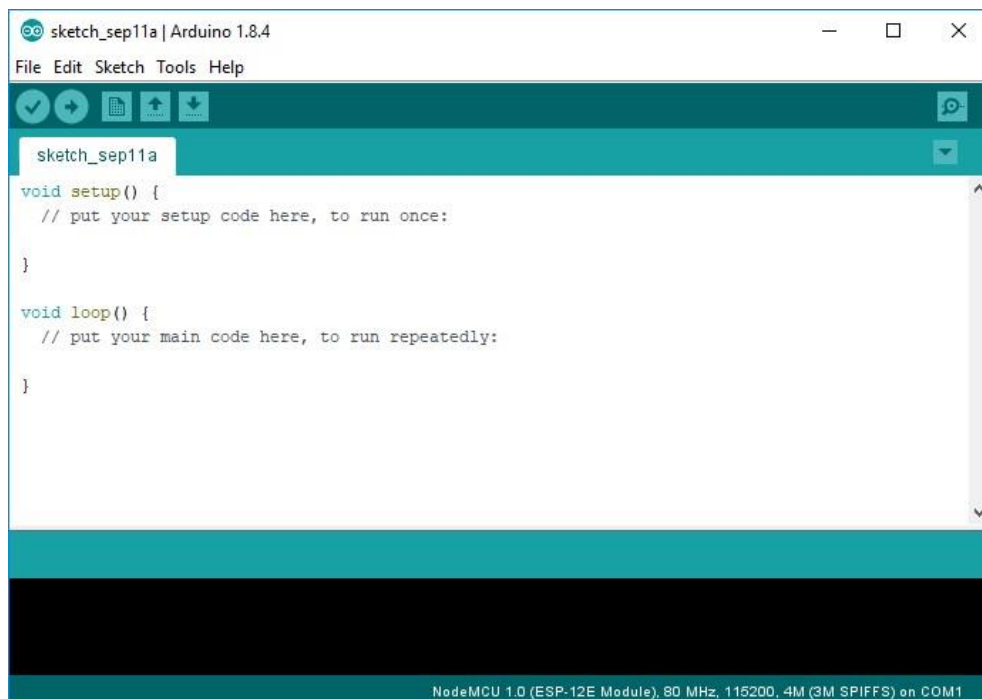


รูปที่ 13 เมื่อ Install เสร็จสิ้นจะมีคำว่า Installed หลัง Library ที่เราเลือก

- เเท่นีก็เป็นอันเสร้จสิ้นการติดตั้ง ESP8266 ลงใน Arduino IDE

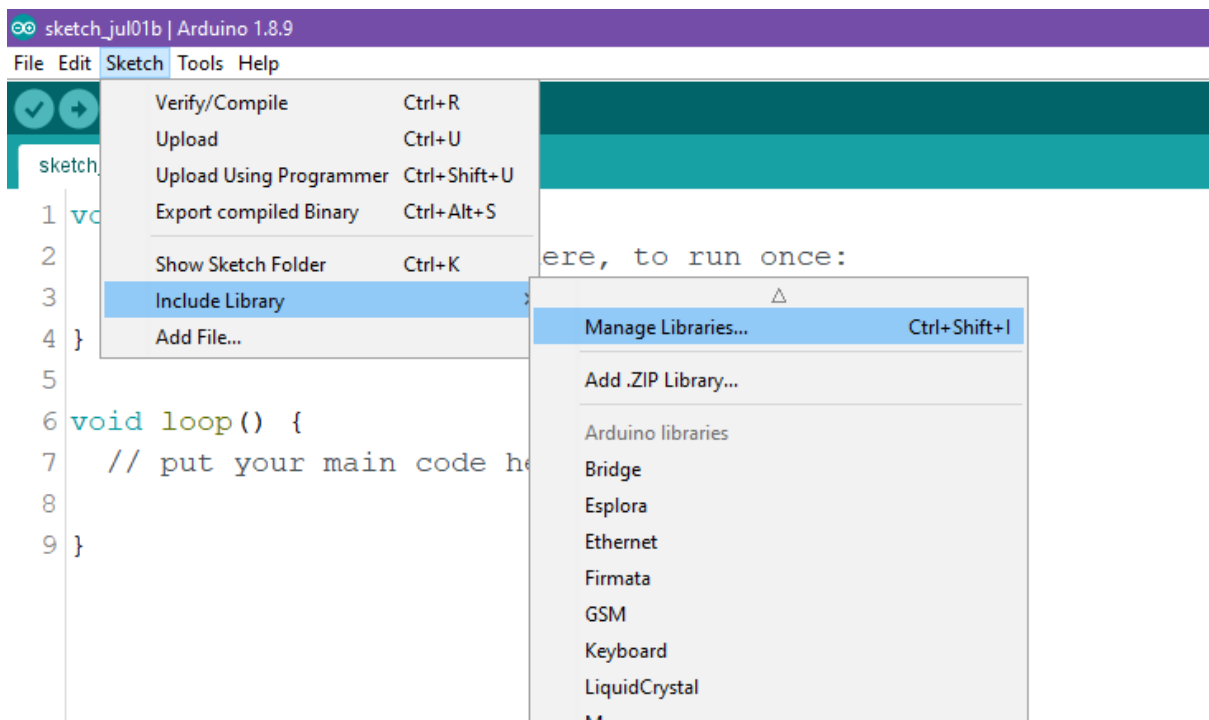
### 3. การติดตั้ง Library สำหรับใช้งานโปรแกรม

- ทำการเปิดหน้าต่าง Arduino IDE ให้ขึ้นมาตามรูปที่ 14



รูปที่ 14 หน้าต่าง Arduino IDE

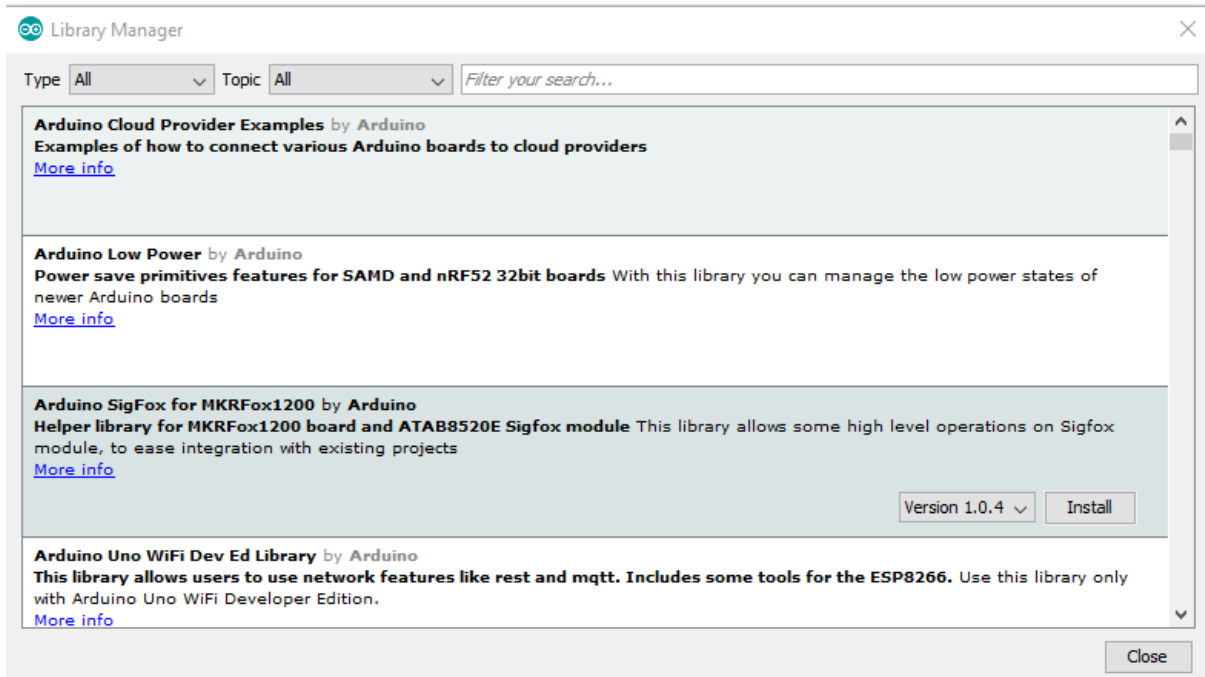
- ทำการกดไปที่เมนู Sketch แล้วเลือก Include Library และทำการเลือก Manage Libraries... เพื่อทำการดาวน์โหลด Library ที่ต้องการมาใช้งาน ตามรูปที่ 15



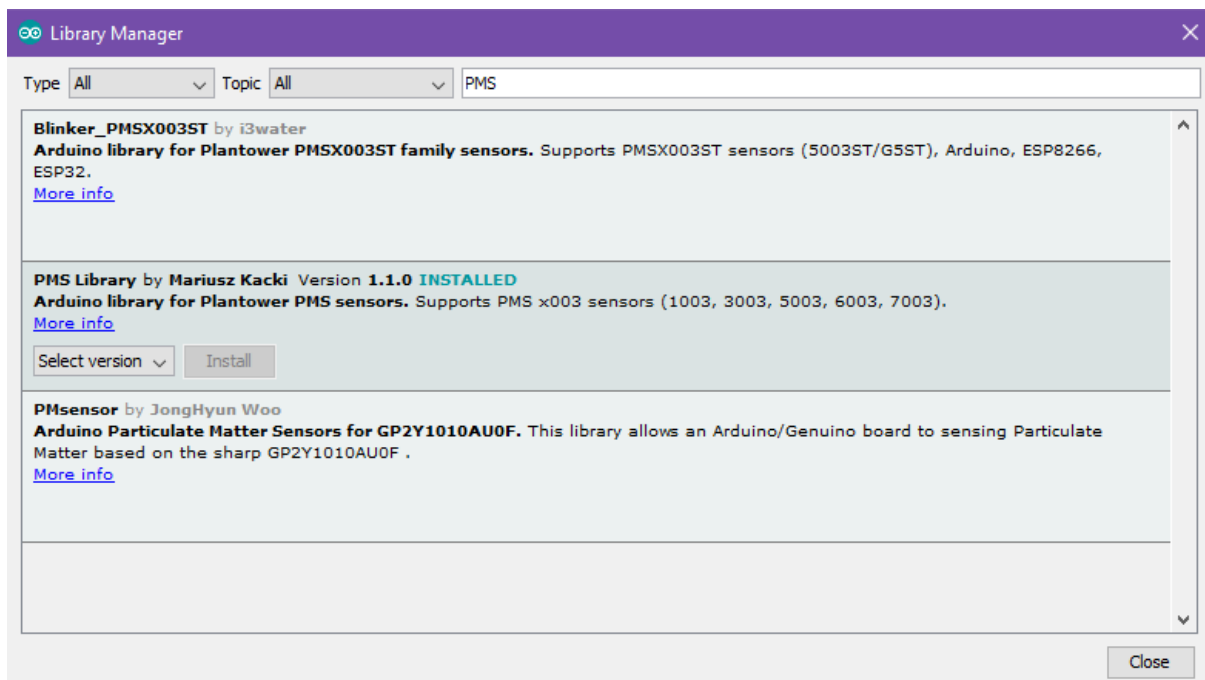
รูปที่ 15 เลือก Sketch > Include Library > Manage Libraries...



- จะทำการแสดงหน้าต่าง Library Manager ขึ้นมา ตามรูปที่ 16 ให้ทำการค้นหาคำว่า “PMS” ลงไปในช่องค้นหาจะแสดง Library ที่เราค้นหาขึ้นมาและ ให้ทำการติดตั้งให้เหมือนภายในรูปที่ 17 ให้เรียบร้อย โดย Library ตัวนี้จะเป็น Library สำหรับใช้งานกับ Sensor วัดฝุ่น PM2.5

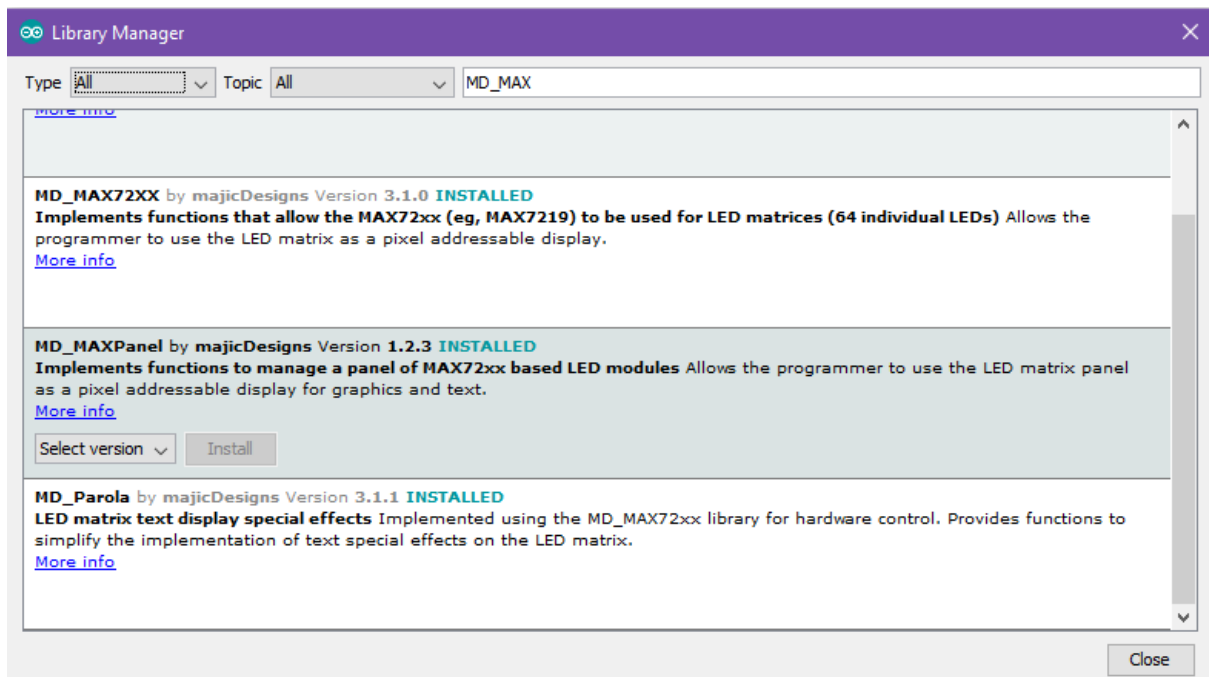


รูปที่ 16 หน้าต่าง Library Manager



รูปที่ 17 ค้นหาคำว่า PMS และทำการติดตั้งให้เรียบร้อย

- ทำการค้นหาคำว่า “MD\_MAX” ต่อเพื่อทำการโหลด Library ตัวถัดไป ตามรูปที่ 18 โดยตัวนี้จะเป็น Library ของตัวแสดงผล LED Dot Matrix 8\*32 ที่ใช้สำหรับแสดงค่า PM2.5

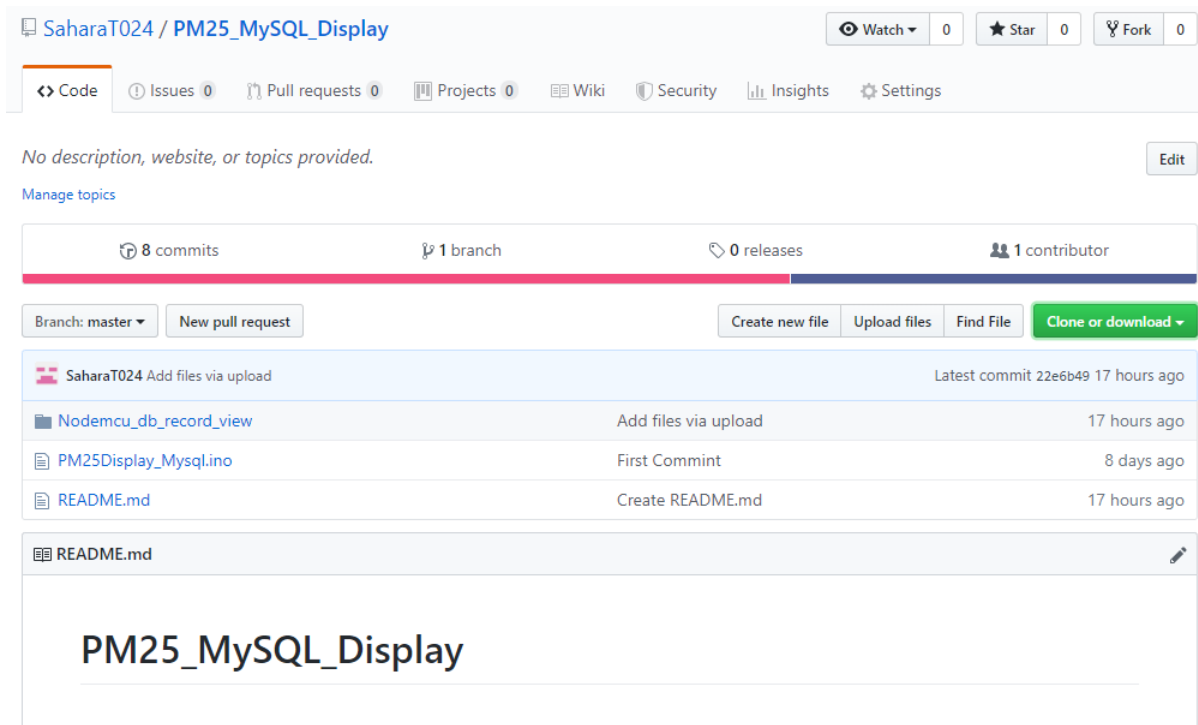


รูปที่ 18 ทำการโหลดให้ครบทั้ง 3 ตัวเพื่อสามารถปรับเปลี่ยนตัว Display ได้

- เหน้ Library ที่ใช้ในการทำงานของโปรแกรมนี้ก็ครบถ้วนเรียบร้อยแล้วสำหรับการเตรียมตัวจัดทำโปรแกรมนี้

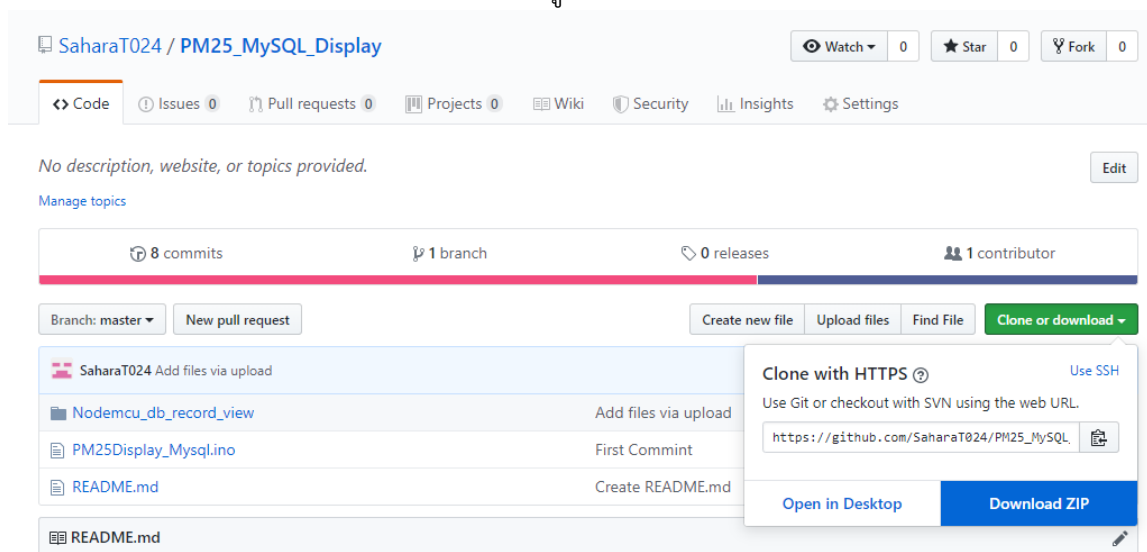
#### 4. Code สำหรับนำมาใช้งานกับตัวอุปกรณ์

- กดเข้าไปที่ลิงก์ตามนี้ [https://github.com/SaharaT024/PM25\\_MySQL\\_Display.git](https://github.com/SaharaT024/PM25_MySQL_Display.git) จะทำการดึงมาหน้า github ตามรูปที่ 19 ไว้สำหรับดาวน์โหลดไฟล์ Code ของตัวโปรแกรมนี



รูปที่ 19 หน้าต่างเมื่อกดลิงค์เข้ามาจะเป็นไฟล์สำหรับใช้งาน

- กดที่ปุ่ม Clone or Download แล้วเลือกเป็น Download ZIP ตามรูปที่ 20 เพื่อให้สามารถได้ไฟล์ Code มาใส่ไว้ในตัวโปรแกรม Arduino IDE หรือสามารถกดเข้าไปที่ไฟล์ PM25Display\_Mysql.ino แล้วกดที่ปุ่ม Raw ก็สามารถ Copy แล้ววางในตัวโปรแกรม Arduino IDE ก็ได้เหมือนกัน ตามรูปที่ 21 และ 22



รูปที่ 20 กดที่ปุ่มดาวน์โหลดเพื่อทำการโหลดตัวไฟล์มาใช้งาน

SaharaT024 / PM25\_MySQL\_Display

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Branch: master PM25\_MySQL\_Display / PM25Display\_Mysql.ino Find file Copy path

SaharaT024 First Commit 4da60f2 9 minutes ago

1 contributor

232 lines (195 sloc) 6.47 KB Raw Blame History

```

1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <ESP8266WebServer.h>
4 #include <ESP8266HTTPClient.h>
5 #include <PMS.h>
6 #include <MD_MAX72xx.h>

```

รูปที่ 21 หลังจากกดเข้าไฟล์มาแล้วจะได้หน้าต่างที่มีรายละเอียดของ Code

```

#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266HTTPClient.h>
#include <PMS.h>
#include <MD_MAX72xx.h>

// Parameter LED Dot Matrix
#define PRINT(s, v) { Serial.print(F(s)); Serial.print(v); }
#define MAX_DEVICES 4
#define CLK_PIN   D5 // or SCK
#define DATA_PIN  D7 // or MOSI
#define CS_PIN     D4 // or SS

// Text parameters
#define CHAR_SPACING 1 // pixels between characters

// Config Sensor PM2.5
PMS pms(Serial);
PMS::DATA data;

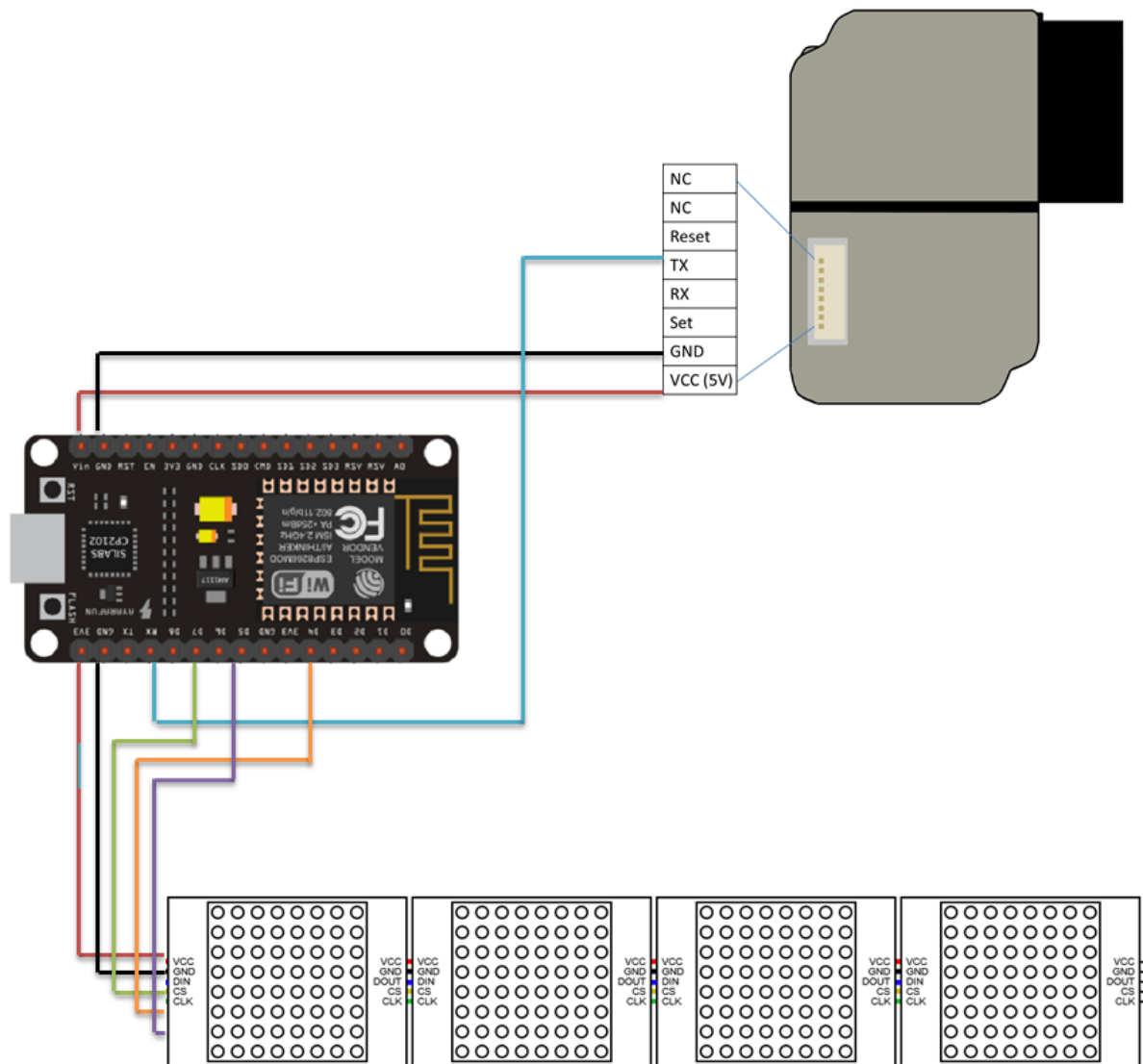
MD_MAX72XX mx = MD_MAX72XX(CS_PIN, MAX_DEVICES);

```

รูปที่ 22 เมื่อกดที่ปุ่ม Raw จะได้เป็นแบบ Code ที่ง่ายต่อการใช้งาน

- นำ Code ไปวางในโปรแกรม Arduino IDE ก็จะสามารถใช้งานและปรับแต่ง Code ได้ตามที่ต้องการ

## 5. การต่ออุปกรณ์สำหรับเตรียมใช้งาน



รูปที่ 23 แผนภาพการต่ออุปกรณ์

<b>PIN1</b>	VCC	Positive power 5V	1
<b>PIN2</b>	GND	Negative power	2
<b>PIN3</b>	SET	Set pin /TTL level@3.3V, high level or suspending is normal working status, while low level is sleeping mode.	
<b>PIN4</b>	RX	Serial port receiving pin/TTL level@3.3V	
<b>PIN5</b>	TX	Serial port sending pin/TTL level@3.3V	3
<b>PIN6</b>	RESET	Module reset signal /TTL level@3.3V, low reset.	
<b>PIN7/8</b>	NC		

รูปที่ 24 ขาต่อที่ต้องใช้ของ PMS3003 Sensor

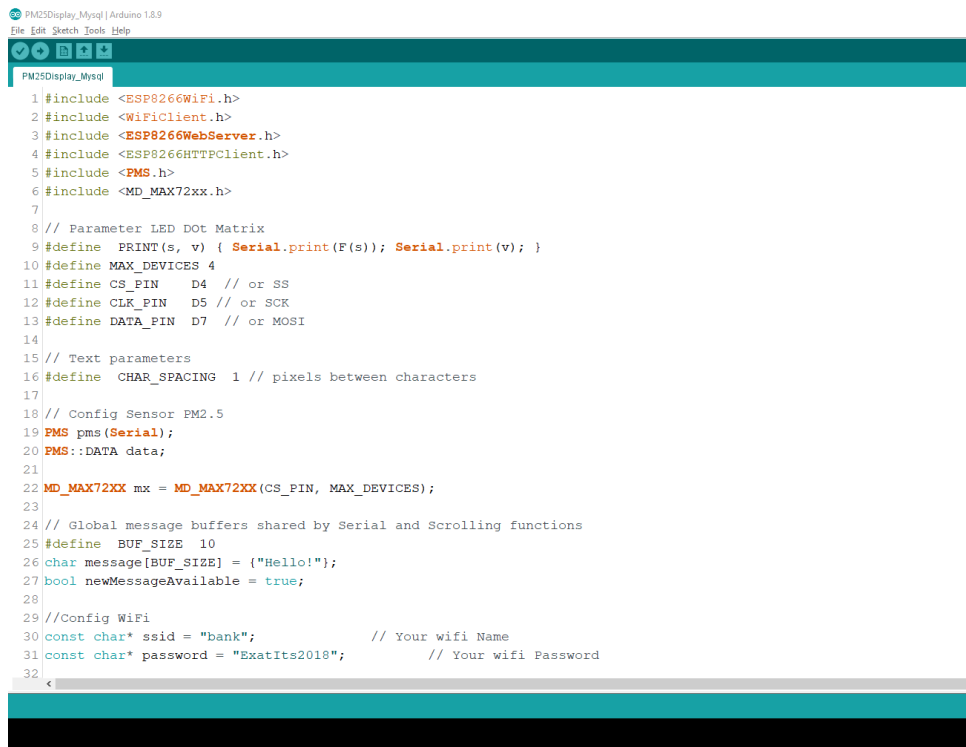
PIN1	VCC	Positive Power 3.3 – 5 V	ต่อที่ขาไฟ 3.3 V
PIN2	GND	Negative Power	ต่อที่ขา Ground
PIN3	DIN	ขาข้อมูลเข้า	ต่อที่ขา D7
PIN4	CS	ขาสัญญาณ	ต่อที่ขา D4
PIN5	CLK	ขา Clock	ต่อที่ขา D5

รูปที่ 25 ขาต่อที่ต้องใช้ของ LED Dot Matrix 8\*32

- ทำการต่ออุปกรณ์ตามรูปที่ 23 และดูข้อมูลการต่อขาตามรูปที่ 24 และ รูปที่ 25

## 6. Code ที่ใช้ภายในตัวโปรแกรมนี

- ให้ทำการเปิด Code ที่ได้โหลดมาในขั้นตอนที่ 4 จะได้ตามรูปที่ 26



```
1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <ESP8266WebServer.h>
4 #include <ESP8266HTTPClient.h>
5 #include <PMS.h>
6 #include <MD_MAX72xx.h>
7
8 // Parameter LED DOT Matrix
9 #define PRINT(s, v) { Serial.print(F(s)); Serial.print(v); }
10 #define MAX_DEVICES 4
11 #define CS_PIN D4 // or SS
12 #define CLK_PIN D5 // or SCK
13 #define DATA_PIN D7 // or MOSI
14
15 // Text parameters
16 #define CHAR_SPACING 1 // pixels between characters
17
18 // Config Sensor PM2.5
19 PMS pms(Serial);
20 PMS::DATA data;
21
22 MD_MAX72XX mx = MD_MAX72XX(CS_PIN, MAX_DEVICES);
23
24 // Global message buffers shared by Serial and Scrolling functions
25 #define BUF_SIZE 10
26 char message[BUF_SIZE] = {"Hello!"};
27 bool newMessageAvailable = true;
28
29 //Config WiFi
30 const char* ssid = "bank"; // Your wifi Name
31 const char* password = "ExatIts2018"; // Your wifi Password
32
```

รูปที่ 26 หน้าตา Code เมื่อทำการเปิดขึ้นมา

- โดยในส่วนแรกตามรูปที่ 27 จะเป็นส่วนในการ Import Library ที่จำเป็นต้องใช้มาไว้ในตัวโปรแกรมนีโดยในบรรทัด 1 – 4 จะเป็นการ Import Library เกี่ยวกับการเชื่อมต่อ WiFi ต่างๆและบรรทัดที่ 5 จะเป็นการ Import Library เกี่ยวกับ PMS Sensor และในบรรทัดที่ 7 จะเป็นของ LED Dot Matrix

```
1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <ESP8266WebServer.h>
4 #include <ESP8266HTTPClient.h>
5 #include <PMS.h>
6 #include <MD_MAX72xx.h>
```

รูปที่ 27 ส่วนการ Import Library มาใช้งาน



- ในส่วนของบรรทัดที่ 9 – 13 จะเป็นการกำหนดตัวแปรที่จะนำไปใช้และเป็นตัวแปรของ LED Dot Matrix เพื่อให้สามารถเชื่อมต่อกับตัวโปรแกรมได้ตามขาที่เราได้ทำการต่อเข้ากับ ตัว NodeMCU ตามรูปที่ 28

```

8 // Parameter LED Dot Matrix
9 #define PRINT(s, v) { Serial.print(F(s)); Serial.print(v); }
10 #define MAX_DEVICES 4
11 #define CS_PIN D4 // or SS
12 #define CLK_PIN D5 // or SCK
13 #define DATA_PIN D7 // or MOSI

```

รูปที่ 28 ส่วนของการกำหนดตัวแปรของ LED Dot Matrix

- ในส่วนของบรรทัดที่ 16 จะเป็นการกำหนดตัวแปรเพื่อใช้ในการแสดงผลของหน้าจอ LED Dot Matrix ตามรูปที่ 29

```

15 // Text parameters
16 #define CHAR_SPACING 1 // pixels between characters

```

รูปที่ 29 ส่วนของการกำหนดตัวแปรใช้ในการแสดงของ LED Dot Matrix

- ในส่วนของบรรทัดที่ 19 – 20 จะเป็นการกำหนดตัวแปรของ PMS3003 เพื่อให้สามารถเรียกใช้ฟังก์ชันใน Library ของ PMS ที่ Import มาได้ เช่น Library อ่านค่า PM2.5 ตาม

รูปที่ 30

```

18 // Config Sensor PM2.5
19 PMS pms(Serial);
20 PMS::DATA data;

```

รูปที่ 30 ส่วนของการกำหนดตัวแปรเพื่อเรียกใช้งานฟังก์ชันภายใน Library

- ในส่วนของบรรทัดที่ 22 จะเป็นการกำหนดตัวแปรของ จอแสดงผล LED Dot Matrix เพื่อให้สามารถใช้งานฟังก์ชันใน Library ได้ ตามรูปที่ 31

```

22 MD_MAX72XX mx = MD_MAX72XX(CS_PIN, MAX_DEVICES);

```

รูปที่ 31 ส่วนของการกำหนดตัวแปรเพื่อเรียกใช้งานฟังก์ชันใน Library

- ในส่วนของบรรทัดที่ 26 – 27 จะเป็นการกำหนดตัวแปรเพื่อรับค่าตัวอักษรจากโปรแกรมไปแสดงบนหน้าจอ LED Dot Matrix ตามรูปที่ 32

```
24 // Global message buffers shared by Serial and Scrolling functions
25 #define BUF_SIZE 10
26 char message[BUF_SIZE] = {"Hello!"};
27 bool newMessageAvailable = true;
```

รูปที่ 32 ส่วนของการกำหนดตัวแปรเพื่อใช้งานการรับค่ามาแสดงผลบนจอ LED Dot Matrix

- ในส่วนของบรรทัดที่ 30 – 31 จะเป็นการกำหนดตัวแปรเพื่อใช้สำหรับการเข้าใช้รหัส WIFI โดยจะเป็นการใส่ SSID หรือชื่อ WIFI และ Password ของ WIFI ที่จะใช้ลงไป ตามรูปที่ 33

```
29 //Config WiFi
30 const char* ssid = "SSID WiFi";          // Your wifi Name
31 const char* password = "Password WiFi"; // Your wifi Password
```

รูปที่ 33 ในส่วนของการกำหนดการเข้าใช้งาน WIFI

- ในส่วนของบรรทัดที่ 36 – 38 จะเป็นการกำหนด IP ให้ตัว NodeMCU เพื่อให้สามารถเข้าใช้งานในเครือข่ายส่วนตัวที่เราต้องการได้เพื่อให้ไม่เกิดการคีย์ IP ไม่ตรงกับที่กำหนดไว้แล้ว จะเกิดการเข้าเครือข่ายมาได้ตามรูปที่ 34

```
35 // Config IP Address
36 IPAddress ip(172, 20, 3, 181);
37 IPAddress gateway(172, 20, 3, 254);
38 IPAddress subnet(255, 255, 255, 0);
```

รูปที่ 34 ในส่วนของการกำหนด IP ต่างๆให้ NodeMCU

- ในส่วนของบรรทัดที่ 40 – 66 จะเป็นการตั้งค่าให้กับตัว NodeMCU ว่าให้ส่งอุปกรณ์ไหนทำงานบ้าง ให้เริ่มทำการต่อ WIFI เลยหรือไม่ ตามรูปที่ 35

```

40 void setup() {
41   // put your setup code here, to run once:
42   delay(1000);
43   pms.passiveMode();
44   mx.begin();
45   Serial.begin(9600);
46   WiFi.mode(WIFI_OFF);           //Prevents reconnection issue (taking too long to connect)
47   delay(1000);
48   WiFi.mode(WIFI_STA);           //This line hides the viewing of ESP as wifi hotspot
49
50   WiFi.config(ip, gateway, subnet);
51   WiFi.begin(ssid, password);    //Connect to your WiFi router
52   Serial.println("");
53
54   Serial.print("Connecting");
55   // Wait for connection
56   while (WiFi.status() != WL_CONNECTED) {
57     delay(250);
58     Serial.print(".");
59     delay(250);
60   }
61
62   //If connection successful show IP address in serial monitor
63   Serial.println("");
64   Serial.println("Connected to Network/SSID");
65   Serial.print("IP address: ");
66   Serial.println(WiFi.localIP()); //IP address assigned to your ESP
67 }

```

รูปที่ 35 ในส่วนของการทำงานตั้งค่าเพื่อให้ NodeMCU เริ่มส่งส่วนต่างๆทำงาน

- ในบรรทัดที่ 43 จะเป็นการสั่งให้ PMS3003 Sensor ทำงานในรูปแบบ Passive Mode
- ในบรรทัดที่ 44 จะเป็นการสั่งให้ LED Dot Matrix เริ่มทำงาน
- ในบรรทัดที่ 45 จะเป็นการสั่งให้แสดงข้อมูลต่างๆที่เกิดขึ้นในโปรแกรมผ่านทาง Serial Port ช่องทางที่ 9600
- ในบรรทัดที่ 46 – 59 จะเป็นการสั่งให้ NodeMCU เริ่มทำการเชื่อมต่อกับ WIFI ที่เราได้กำหนดไว้ตามตัวแปรข้างต้นให้ NodeMCU สามารถเข้าถึงการใช้งาน Internet ผ่าน WIFI
- ในบรรทัดที่ 63 – 66 จะเป็นการแสดงการทำงานต่างๆว่าสามารถเชื่อมต่อกับ WIFI ได้หรือไม่มีการมี IP ถูกต้องตามที่เรากำหนดไว้หรือไม่ผ่าน Serial Port ตามที่เรากำหนด

- ในช่วงบรรทัดที่ 69 – 169 จะเป็นการทำงานในฟังก์ชันของ Loop ที่จะสั่งให้ตัว NodeMCU ทำงานวนไปเรื่อยๆตามคำสั่งที่กำหนดไว้
- ในบรรทัดที่ 71 จะเป็นการกำหนดตัวแปรเพื่อให้สามารถส่งค่าไปยังหน้า Client ที่เราเตรียมไว้เพื่อให้สามารถเก็บค่าข้อมูลที่อ่านได้ไว้ในฐานข้อมูลตามรูปที่ 36

```
70 | // put your main code here, to run repeatedly:
71 | HTTPClient http; //Declare object of class HTTPClient
```

รูปที่ 36 ในส่วนของการกำหนดตัวแปรเพื่อเข้าใช้งานฟังก์ชันต่างของ HTTPClient

- ในบรรทัดที่ 73 จะเป็นการกำหนดให้ตัวเครื่อง PMS3003 Sensor เริ่มทำงานอ่านค่า PM ต่างๆตามรูปที่ 37

```
73 | pms.wakeUp() ;
```

รูปที่ 37 ในส่วนของการกำหนดให้ PMS Sensor ทำงาน

- ในบรรทัดที่ 76 – 88 จะเป็นการกำหนดตัวแปรเพื่อรองรับค่าที่อ่านได้จาก PMS Sensor เพื่อนำมาแสดงผลผ่านจอ LED Dot Matrix โดยการแปลงค่าจากตัวเลขให้กลายเป็นตัวอักษรเพื่อให้สามารถแสดงผลจอสวยงามได้อย่างถูกต้อง ตามรูปที่ 38

```
76 | String PM25ValueSend, PM10ValueSend, PM100ValueSend;
77 |
78 | // Reading PM2.5
79 | int pm25value = data.PM_AE_UG_2_5; // Get Value PM2.5
80 | PM25ValueSend = String(pm25value); //String to interger conversion
81 |
82 | // Reading PM1.0
83 | int pm10value = data.PM_AE_UG_1_0; // Get Value PM1.0
84 | PM10ValueSend = String(pm10value); // String to interger conversion
85 |
86 | // Reading PM10.0
87 | int pm100value = data.PM_AE_UG_10_0; // Get value PM10.0
88 | PM100ValueSend = String(pm100value); // String to interger conversion
```

รูปที่ 38 ในส่วนของการกำหนดตัวแปรเพื่อมารับค่าและแปลงค่าที่รับมา

- ในบรรทัดที่ 90 – 100 จะเป็นส่วนในการนำค่า PM2.5 ที่ได้จาก Sensor มาแยกเป็นหลักหน่วย สิบ และ ร้อย เพื่อให้สามารถแสดงค่าออกผ่านหน้าจอแสดงผล LED Dot Matrix ได้อย่างถูกต้อง ตามรูปที่ 39

```

90 // Value Show on Display LED dot Matrix
91 int a=0;
92 int b=0;
93 int c=0;
94
95 if(pm25value%10 != 0)
96     a = pm25value%10;
97 if(pm25value/10 != 0)
98     b = (pm25value/10)%10;
99 if(pm25value/100 != 0)
100    c = pm25value/100;

```

รูปที่ 39 ทำการแยกค่าต่างๆไปตามหลักเพื่อแสดงผลได้อย่างถูกต้อง

- ในบรรทัดที่ 103 จะเป็นการสร้างตัวแปรเพื่อเก็บข้อความ http ที่ไว้สำหรับเรียกใช้งานหน้าเว็บ InsertDB เพื่อให้สามารถนำค่าข้อมูลที่อ่านได้ไปใส่ฐานข้อมูลได้

```

103 String url = "http://172.20.3.3/dust/InsertDB.php?&n=RAE01&r="+String(pm25value)+"&s="+String(pm10value)+"&t="+String(pm100value);
104 Serial.println(url);

```

รูปที่ 40 ทำการเก็บตัวแปร http เพื่อให้สะดวกในการเรียกใช้งาน

- ในบรรทัดที่ 106 – 124 จะมีการเรียกใช้งานฟังก์ชันของ http เพื่อให้สามารถเรียกใช้งานไฟล์ php ที่เราเตรียมไว้เพื่อให้สามารถนำข้อมูลต่างๆเข้าไปยังฐานข้อมูลได้อย่างถูกต้องและครบถ้วน ตามรูปที่ 41 โดยจะมีการเช็คเงื่อนไขว่าสามารถส่งไปยังฐานข้อมูลได้หรือไม่ได้ และให้แสดงผลผ่านหน้าจอ Serial Port ที่กำหนดไว้

```

106 if(pms.readUntil(data)){
107     http.begin(url); //Specify request destination
108     http.addHeader("Content-Type", "application/x-www-form-urlencoded"); //Specify content-type header
109
110     int httpCode = http.GET(); //Send the request
111     String payload = http.getString(); //Get the response payload
112
113     if( httpCode > 0){
114         Serial.println(httpCode); //Print HTTP return code
115         Serial.println(payload); //Print request response payload
116         Serial.println("PM2.5=" + PM25ValueSend);
117         Serial.println("PM1.0=" + PM10ValueSend);
118         Serial.println("PM10.0=" + PM100ValueSend);
119     }
120     else{
121         Serial.printf("[HTTP] GET ... Failed, error: %s\n", http.errorToString(httpCode).c_str());
122         Serial.print("HttpCode = ");
123         Serial.println(httpCode);
124     }

```

รูปที่ 41 ทำการเรียกใช้ไฟล์ที่เตรียมไว้และเช็คข้อมูลถูกส่งไปถูกต้องหรือไม่

- ในบรรทัดที่ 125 – 154 จะเป็นส่วนในการกำหนดเงื่อนไขเพื่อแสดงผลข้อมูลผ่านหน้าจอ

LED Dot Matrix ให้ถูกต้องตามฟังก์ชันที่กำหนดไว้ ตามรูปที่ 42

```

125     if(b == 0 && c == 0){
126         message[0]=' ';
127         message[1]='-';
128         message[2]='0';
129         message[3]='0';
130         message[4]=char(a)+48;
131         message[5]='-';
132         message[6]='\0';
133         printText(0, MAX_DEVICES-1, message);
134     }
135     else if(b == 0 & c != 0){
136         message[0]=' ';
137         message[1]='-';
138         message[2]='0';
139         message[3]=char(b)+48;
140         message[4]=char(a)+48;
141         message[5]='-';
142         message[6]='\0';
143         printText(0, MAX_DEVICES-1, message);
144     }
145     else{
146         message[0]=' ';
147         message[1]='-';
148         message[2]=char(c)+48;
149         message[3]=char(b)+48;
150         message[4]=char(a)+48;
151         message[5]='-';
152         message[6]='\0';
153         printText(0, MAX_DEVICES-1, message);
154     }

```

รูปที่ 42 ส่วนการแสดงผลผ่าน LED Dot Matrix

- ในบรรทัดที่ 156 – 165 จะเป็นส่วนในการแสดงผลผ่าน LED Dot Matrix โดยจะเป็นการแจ้งว่าไม่มีการได้รับข้อมูลจากตัว Sensor ให้แสดงข้อความออกมาเพื่อให้ผู้ใช้งานทราบถึงปัญหา ตามรูปที่ 43

```

156     else{
157         Serial.println("No Data");
158         message[0]='N';
159         message[1]='O';
160         message[2]='D';
161         message[3]='A';
162         message[4]='T';
163         message[5]='A';
164         message[6]='\0';
165         printText(0, MAX_DEVICES-1, message);
166     }

```

รูปที่ 43 ส่วนการแสดงผลข้อมูลเมื่อได้รับข้อมูลจาก Sensor

- ในบรรทัดที่ 167 จะเป็นคำสั่งในการตัดการเชื่อมต่อกับตัวไฟล์ php ตามรูปที่ 44 ที่เรียกใช้การหน้านี้เพื่อเป็นการจบโปรแกรม

```
167 | http.end(); //Close connection
```

รูปที่ 44 ส่วนในการตัดการเชื่อมต่อกับไฟล์ php

- ในบรรทัดที่ 172 – 230 จะเป็นคำสั่งในการทำงานของตัวแสดงผลซึ่งเราสามารถนำมาใช้ได้เลยโดยไม่ต้องไปดัดแปลงใดๆโดยตัว Code นี้จะเป็นการแสดงผลแบบไจว์ตลอดเวลาไม่ได้มีการเลื่อนหรือกระพริบแต่อย่างใด ดู Code ได้ตามรูปที่ 45

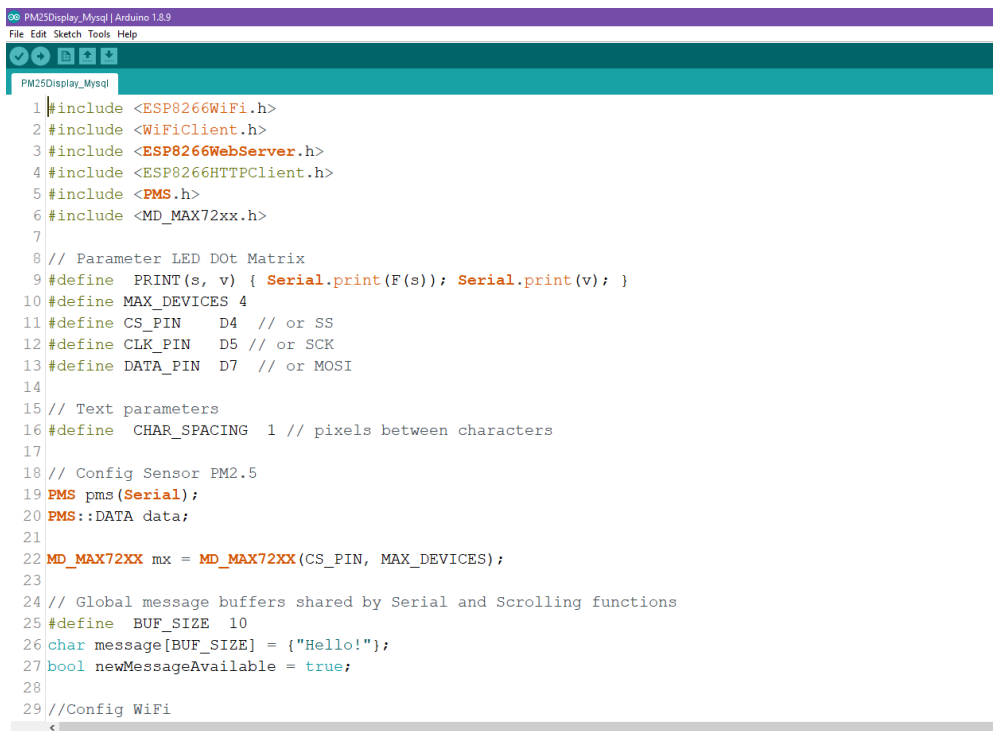
```
172 void printText(uint8_t modStart, uint8_t modEnd, char *pMsg)
173
174 {
175     uint8_t state = 0;
176     uint8_t curLen;
177     uint16_t showLen;
178     uint8_t cBuf[8];
179     int16_t col = ((modEnd + 1) * COL_SIZE) - 1;
180
181     mx.control(modStart, modEnd, MD_MAX72XX::UPDATE, MD_MAX72XX::OFF);
182
183     do // finite state machine to print the characters in the space available
184     {
185         switch(state)
186         {
187             case 0: // Load the next character from the font table
188                 // if we reached end of message, reset the message pointer
189                 if (*pMsg == '\0')
190                 {
191                     showLen = col - (modEnd * COL_SIZE); // padding characters
192                     state = 2;
193                     break;
194                 }
195
196                 // retrieve the next character form the font file
197                 showLen = mx.getChar(*pMsg++, sizeof(cBuf)/sizeof(cBuf[0]), cBuf);
198                 curLen = 0;
199                 state++;
200                 // !! deliberately fall through to next state to start displaying
201
202             case 1: // display the next part of the character
203                 mx.setColumn(col--, cBuf[curLen++]);
204
205                 // done with font character, now display the space between chars
206                 if (curLen == showLen)
207                 {
208                     showLen = CHAR_SPACING;
209                     state = 2;
210                 }
211                 break;
212
213             case 2: // initialize state for displaying empty columns
214                 curLen = 0;
215                 state++;
216                 // fall through
217
218             case 3: // display inter-character spacing or end of message padding (blank columns)
219                 mx.setColumn(col--, 0);
220                 curLen++;
221                 if (curLen == showLen)
222                     state = 0;
223                 break;
224
225             default:
226                 col = -1; // this definitely ends the do loop
227         }
228     } while (col >= (modStart * COL_SIZE));
229
230     mx.control(modStart, modEnd, MD_MAX72XX::UPDATE, MD_MAX72XX::ON);
231 }
```

รูปที่ 45 ในส่วนของการแสดงผลจอ LED Dot Matrix



## 7. การ Upload ข้อมูลลงในบอร์ด NodeMCU

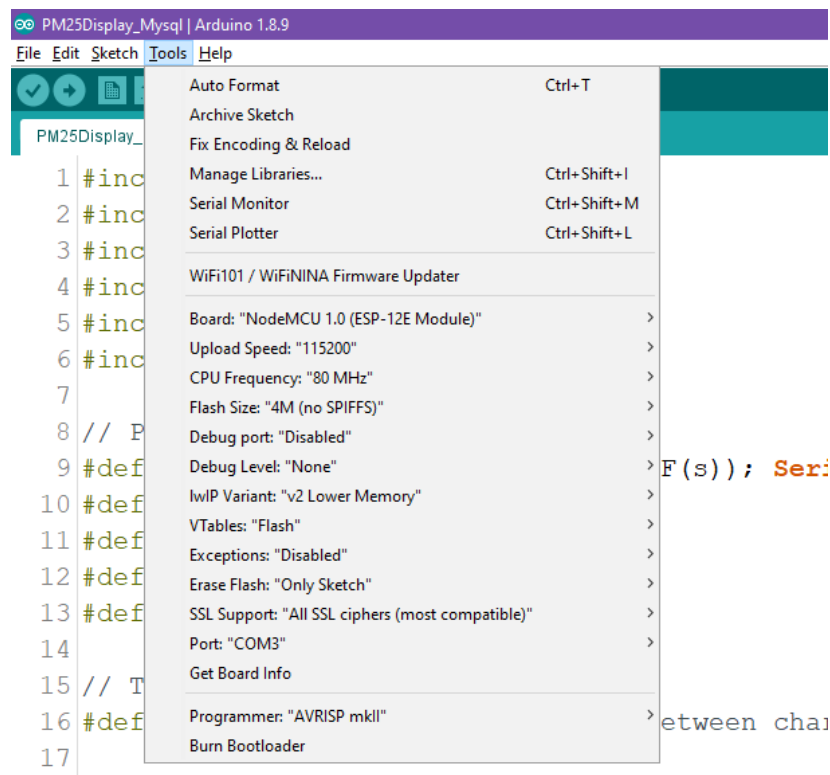
- เตรียม Code ที่โหลดมาตามรูปที่ 46



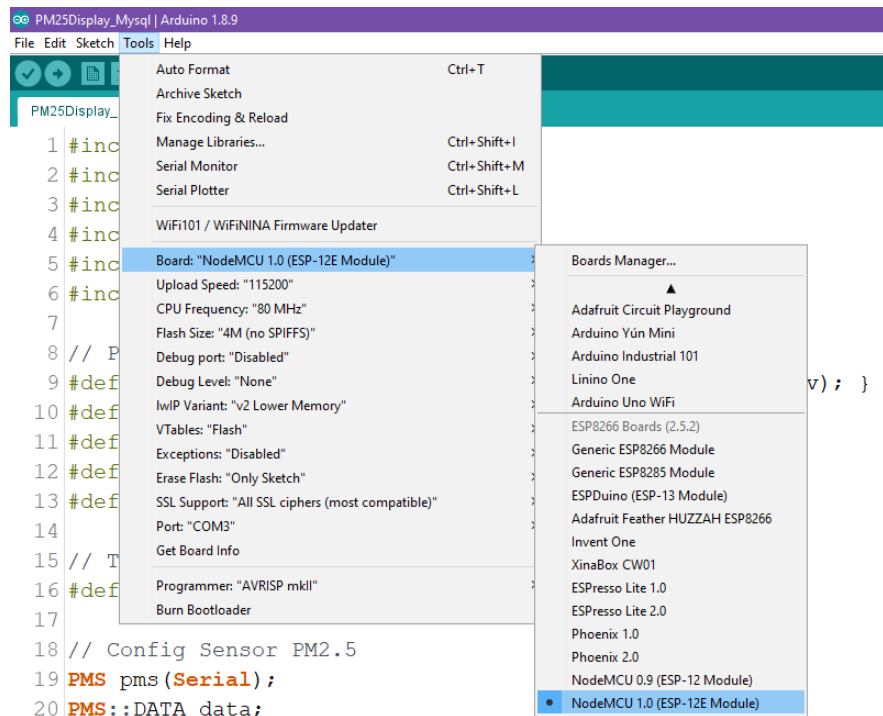
```
1#include <ESP8266WiFi.h>
2#include <WiFiClient.h>
3#include <ESP8266WebServer.h>
4#include <ESP8266HTTPClient.h>
5#include <PMS.h>
6#include <MD_MAX72xx.h>
7
8// Parameter LED DOT Matrix
9#define PRINT(s, v) { Serial.print(F(s)); Serial.print(v); }
10#define MAX_DEVICES 4
11#define CS_PIN D4 // or SS
12#define CLK_PIN D5 // or SCK
13#define DATA_PIN D7 // or MOSI
14
15// Text parameters
16#define CHAR_SPACING 1 // pixels between characters
17
18// Config Sensor PM2.5
19PMS pms(Serial);
20PMS::DATA data;
21
22MD_MAX72XX mx = MD_MAX72XX(CS_PIN, MAX_DEVICES);
23
24// Global message buffers shared by Serial and Scrolling functions
25#define BUF_SIZE 10
26char message[BUF_SIZE] = {"Hello!"};
27bool newMessageAvailable = true;
28
29//Config WiFi
```

รูปที่ 46 เตรียม Code สำหรับการ Upload ข้อมูล

- กดไปที่ปุ่ม Tools ตามรูปที่ 47 และเลือกไปที่เลือกประเภทของบอร์ดให้ตรงกับบอร์ด NodeMCU ที่เราเตรียมเอาไว้ ตามรูปที่ 48

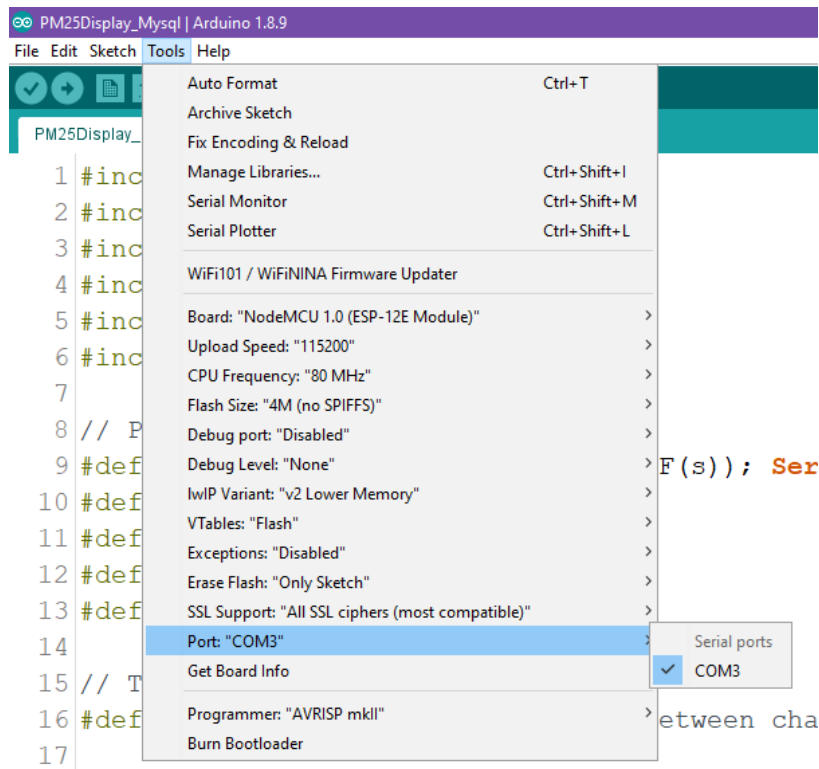


รูปที่ 47 หน้าต่าง Tool เมื่อเปิดขึ้นมา



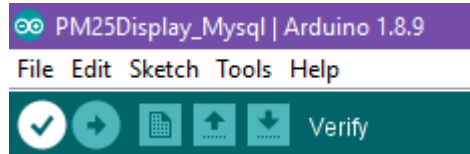
รูปที่ 48 เลือก Board ให้ตรงกับบอร์ดที่เราต้องการ

- ทำการเลือก Port ให้ตรงกับช่องที่เสียบสายตามรูปที่ 49 หากเลือกไม่ตรงจะไม่สามารถ Upload ข้อมูลลงไปได้

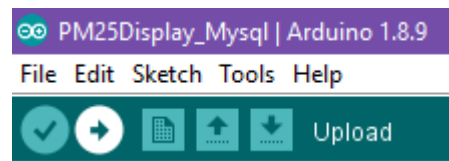


รูปที่ 49 เลือก Port ตามที่เสียบสายเข้ากับคอมพิวเตอร์

- เมื่อเตรียมพร้อมก่อน Upload เรียบร้อยแล้วให้ทำการกดที่ปุ่ม Verify ด้านซ้ายบนตามรูปที่ 50 และ กด Upload ปุ่มที่อยู่ติดๆกันตามรูปที่ 51 เพื่อทำการ Upload Code ลงไปที่บอร์ด NodeMCU

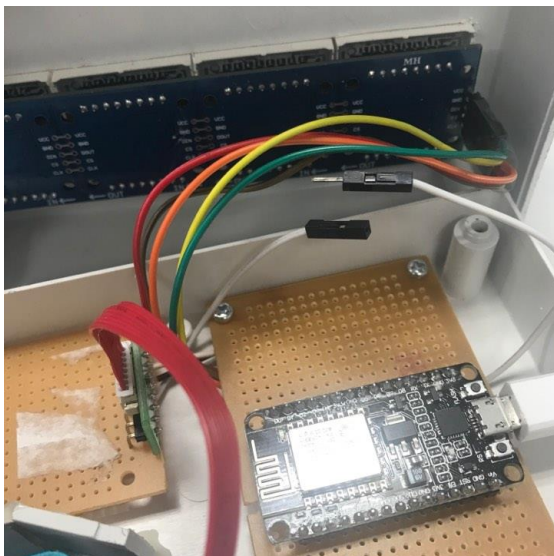


รูปที่ 50 ปุ่ม Verify เพื่อตรวจสอบความถูกต้องของ Code

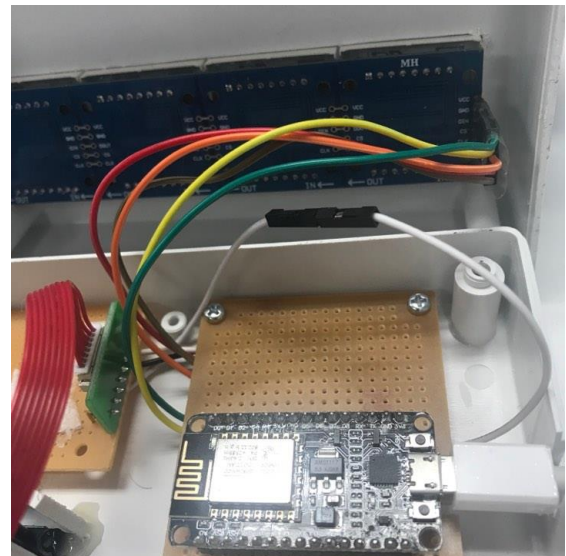


รูปที่ 51 ปุ่ม Upload เพื่อทำการ Upload ข้อมูลลงไปที่บอร์ด

- ก่อน Upload ทุกครั้งให้ทำการถอดสายเส้นสีขาวตามรูปที่ 52 ก่อนเพื่อให้สามารถ Upload Code ได้และค่อยต่อเข้าเหมือนเดิมตามรูปที่ 53 หลัง Upload เสร็จ



รูปที่ 52 ถอดสายสีขาวก่อน Upload



รูปที่ 53 หลัง Upload ให้ต่อสายดังเดิม