# 屏東縣第 63 屆國中小學科學展覽會

# 作品說明書

科 別： 生活與應用科學科(一)

組 別： 國中組

作品名稱： 數位自動化空氣品質旗幟

關 鍵 詞： 空氣品質指標 、 AQI 、 物聯網

（最多三個）

編號：

# 摘要

本作品利用 D1 mini 控制板＋LED 模組，設計出一款「數位自動化空氣品質旗幟」，用以取代傳統布質空氣品質旗幟。本作品以 LED 模組呈現出各項 AQI 資訊，其優點在於能即時更新 AQI 資訊，免去人工查詢 AQI 及更換旗幟的不便。可輕易推廣至不同學校，畫面非常醒目，在遠處即可觀看到，改善布質旗幟根本就看不清楚內容的缺點。更重要的是，將「空氣品質指標 AQI」所代表的含意以及對人體健康的影響，直接呈現出來，讓敏感族群能夠依此訊息，提早應變，真正發揮「空氣品質旗幟」的作用。


# 壹、 前言

## 研究動機

時常會在學校 2 樓教務處走廊看見懸掛一面彩色的旗子，而且有時會有負責的同學去更換成不同的顏色，經過詢問師長之後，發現原來這是「空氣品質旗幟」。藉由此面旗幟，讓同學們了解目前的空氣狀態，好讓不同的敏感族群有相關的因應作為。

旗幟的更換，是由負責同學先到學務處請老師上網查詢空氣品質指標(AQI)，再根據數值更換不同顏色的旗幟，每天中午更換一次。但是這樣的資訊很明顯不夠即時，象徵意義大於實際作用。

有一次到了潮州高中，發現他們學校使用的應該是自製的資訊板，上面有不同顏色的燈號，每顆燈號下方各有一顆按鈕，只要按下按鈕，該燈就會發亮。看見這樣的資訊板不禁想到，校園中是否能夠有一種更自動化的方式能夠呈現空氣品質指標，不需要有人先去上網查詢再去換旗幟或去按按鈕。因為這樣的念頭，所以開始本次的研究。

(教材相關性：國中 8 年級自然課本跨科主題-空氣汙染與自我防護)

| 空氣品質指標 AQI | 0-50 | 51-100 | 101-150 | 151-200 | 201-300 | 301-500 |
|---|---|---|---|---|---|---|
| 對健康影響與活動建議 | 良好 | 普通 | 對敏感族群不健康 | 對所有族群不健康 | 非常不健康 | 危害 |
| 狀態色塊 | 綠 | 黃 | 橘 | 紅 | 紫 | 褐紅 |

# 目的

希望能夠設計出一款作品，能夠即時顯示空氣品質指標，並且達到以下幾個效果：

一、即時性：資訊能夠即時更新，而不是每天中午才換一次旗幟。

二、 明顯性：「空氣品質旗幟」可懸掛的地方非常有限，而且布質旗子本身垂下來後，根本看不清楚它的內容，所以希望設計出的作品可以非常的醒目，即使在遠處也能一目了然

三、推廣性：設計出的作品，具有高度可複製性，製作簡單，可以推廣到各國中小

四、教育性：「空氣品質旗幟」是以顏色來提供訊息，但多數人根本不會記得顏色所代表的

含意。因此希望設計出的作品，可以將「空氣品質指標 AQI」所代表的含意以及對人體健康的影響，直接呈現出來。


## 貳、 研究設備及器材

一、 硬體：筆記型電腦、雷射切割機、3D 列印機、焊接工具

二、 軟體：Arduino IDE、PCtoLCD2002(中文建模軟體)、fritzing (電路繪製軟體)、

三、 材料：Esp8266 控制板、壓克力板、3D 列印線材、銅柱、螺絲、杜邦接線、max7219 LED 模組

## 參、 研究過程或方法

一、 研究流程：

1.資料蒐集與討論 → 2.確定製作方向 → 3.軟硬體設計 → 4.進行程式測試及修正 → 5.實機組裝與測試

二、 研究過程

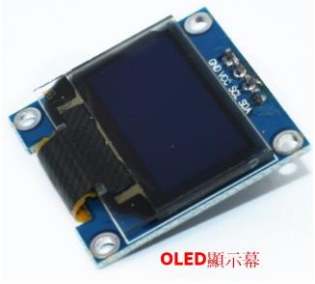
(一)、 資料蒐集討論並確定製作方向：

1. 經過一段時間資料的查詢與收集，發現使用物聯網的方式，可以達到**即時性**的目的，而且只要設定及硬體設備沒問題，根本就不需要浪費人力去上網查詢及更換旗幟或按按鈕。本次設計採用的是可連接無線網路的 Esp8266 控制板 (D1 mini)。


2. 「空氣品質旗幟」的體積較大，使用控制板及馬達去做更換並不容易，而且製作出的成品過大也不利於推廣，所以決定捨棄使用旗幟，改以數位顯示的方式進行。

3. 為達到**推廣性**的目的，需要設計出一款可因應不同學校所在地，查詢所需 AQI 資訊的成品，因此需提供一個控制介面，可以調整部分查詢參數。所以決定讓 Esp8266 控制板模擬成 web 網頁伺服器，只要連接到此網頁，即可透過手機、平板、電腦等各種設備，進行各項參數的調整，非常方便。

4. 為達到**教育性**的目的，設計出的成品需能夠顯示 AQI 的燈號顏色、AQI 的數值、以及此燈號所代表的含意(對人體健康如何影響)。所以最終決定採用「MAX7219 LED 點矩陣顯示模組」進行成品設計，而且 MAX7219 LED 的體積比 OLED、TFT Display 等數位顯示素材大得多，又可以兼顧**明顯性**的目的。

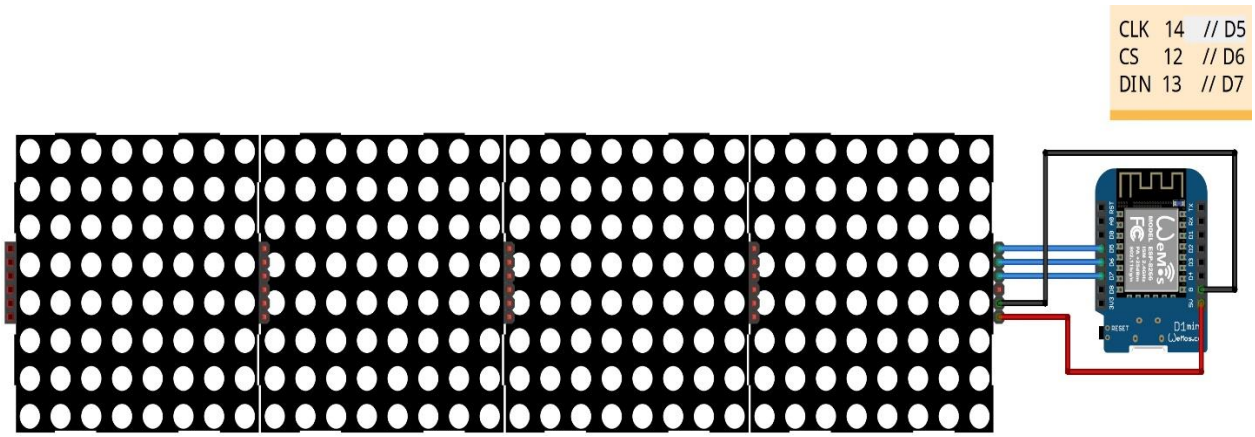


(二)、 軟硬體及外殼設計

1. 硬體焊接：

(1) 將 MAX7219 LED 與 D1 mini 控制板進行焊接

(2) 系統線路連接圖如下




2. 程式設計：

(1) 學習使用 Arduino IDE 進行程式開發設計。

> Arduino 基本語法筆記
>
> Arduino 的程式語法基於 C/C++, 其實就是客製化的 C/C++ 語言, 其程式架構仿自廣為藝術與設計界人士熟悉的 Processing 語言, 而其開發工具 Arduino IDE 則是衍生自以 Processing 為基礎的電子開發設計平台 Wiring. 由於 Processing IDE 使用 Java 撰寫, 因此 Arduino IDE 有自帶一個 JRE. Processing 語言撰寫的程式稱為 Sketch (草稿碼), 乃是簡化後之 Java 語法, 經 IDE 編譯變成可執行的 Java 類別. 而 Arduino IDE 則是以 Processing IDE 為架構, 但是採用了 C/C++ 語法.

(2) D1 mini 控制板晶片為 Esp8266，需先新增開發版驅動程式。


(3) MAX7219 LED 的驅動與畫面呈現，使用的是 MD_Parola、MD_MAX72xx 等函式庫，來簡化並縮短開發的時間。

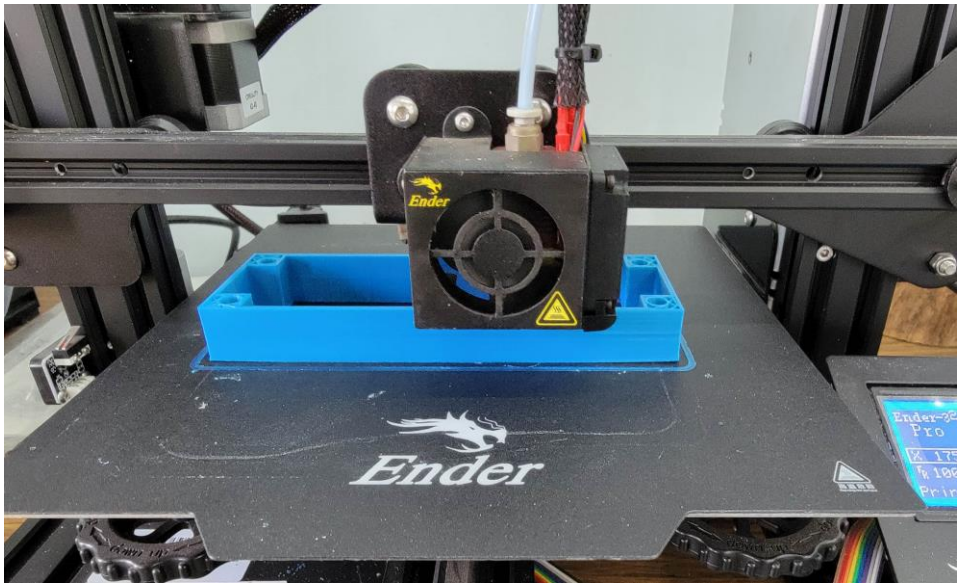
(4) 需要定時上網抓取資料，所以使用 TimeLib、NtpClientLib、SimpleTimer 等函式庫，連接到時間伺服器(NTP 伺服器)，藉由網路來校正時間。

(5) D1 mini 控制板網路連線至環保署及中央氣象局，是以 API 的方式

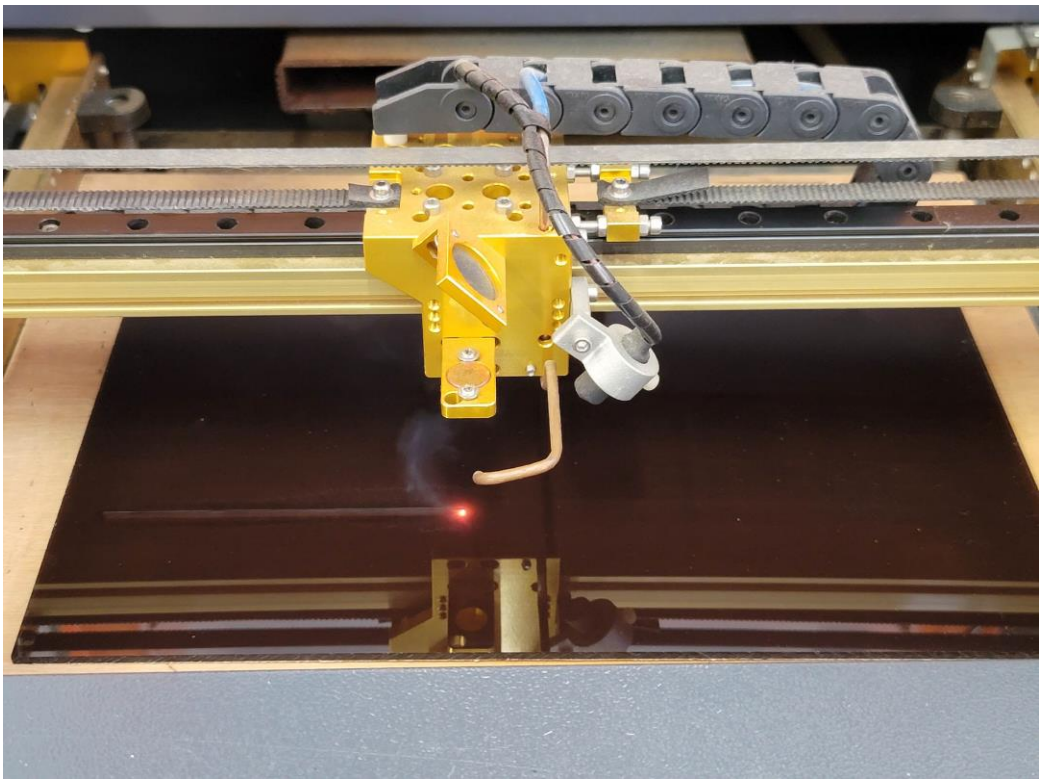
獲取所需的資料，故使用 ArduinoJson 函式庫，來解析下載到的訊息。

3. 外殼設計：

(1) 使用 3D 列印機製作外殼




(2) 使用雷射切割機進行壓克力切割，製作外殼及顯示幕面板



(三)、 進行程式測試及修正：測試中遇到幾個問題

1. HTTPS 協定：一開始使用 Esp8266 函式庫提供的範例程式碼，透過 API

的方式連線取得環保署及中央氣象局的資料時，會發生錯誤。經過驗證後發現，以往是以 HTTP(超文本傳輸協定) 作為傳輸協定，但現在的網站轉為透過 HTTPS(超文本傳輸安全協定) 傳輸資料，HTTPS 在資料通訊過程使用 SSL/TLS 進行加密，所以 D1 mini 控制板無法順利獲取資料。後來經過修正改用 ESP8266HTTPClient 配合 WiFiClientSecure 函式庫，終於順利解決此問題。

修正前部分程式碼

```
if (client.connect(weatherHost, 80))   //連接至網路主機獲取資訊
{
   client.println("GET /data/2.5/weather?id=" + String(CityIDs[0]) + "&units=metric&APPID=" +
weatherAPIKEY);          //告知主機想下載那些資料
   client.println("Host: api.openweathermap.org");   //送出連線所需資料
   client.println("User-Agent: ArduinoWiFi/1.1");   //送出連線所需資料
   client.println("Connection: close");             //送出連線所需資料
   client.println();                                //送出連線所需資料
}
while (client.connected() || client.available())     //如連接主機成功
{
   char c = client.read();          //開始讀取資料
   result = result + c;             //將資料組合在一起
}
```

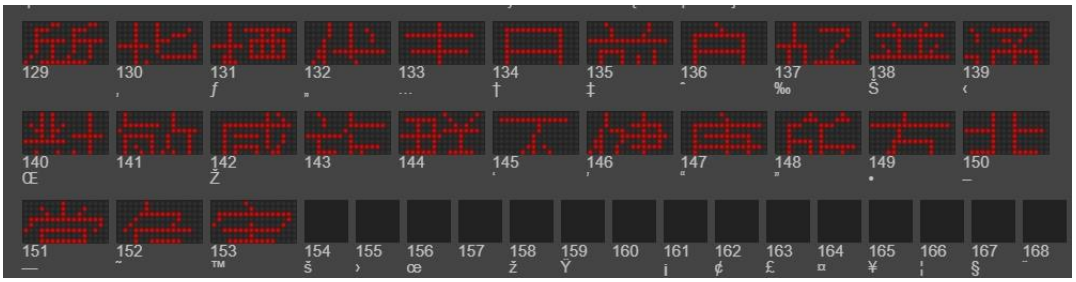
修正後部分程式碼

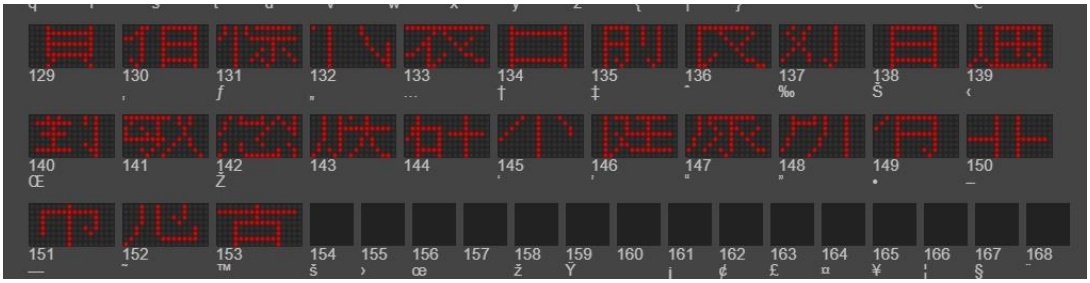
```
   WiFiClientSecure client;                  //宣告使用 client 物件
   client.setInsecure();                     //設定 https 協定安全性，不用經過身分認證即可通訊
   HTTPClient https;                         //宣告使用 https 物件
   if (https.begin(client, fullUrl)) {       //與主機連線
      int httpCode = https.GET();            //從主機端傳回連接訊息
      Serial.println("======== Response code: " + String(httpCode));   //在監控視窗中列印訊息
      if (httpCode > 0) {       //如果連接成功
         Serial.println(https.getString());          //在監控視窗中列印訊息
         String line2 = https.getString();           //將下載的資料存在 line2 變數中
```

2. Json 格式問題：透過 API 的方式連線取得環保署資料時，會獲取大量的資料，這些資料因佔據太多 D1 mini 控制板的記憶空間，導致 D1 mini

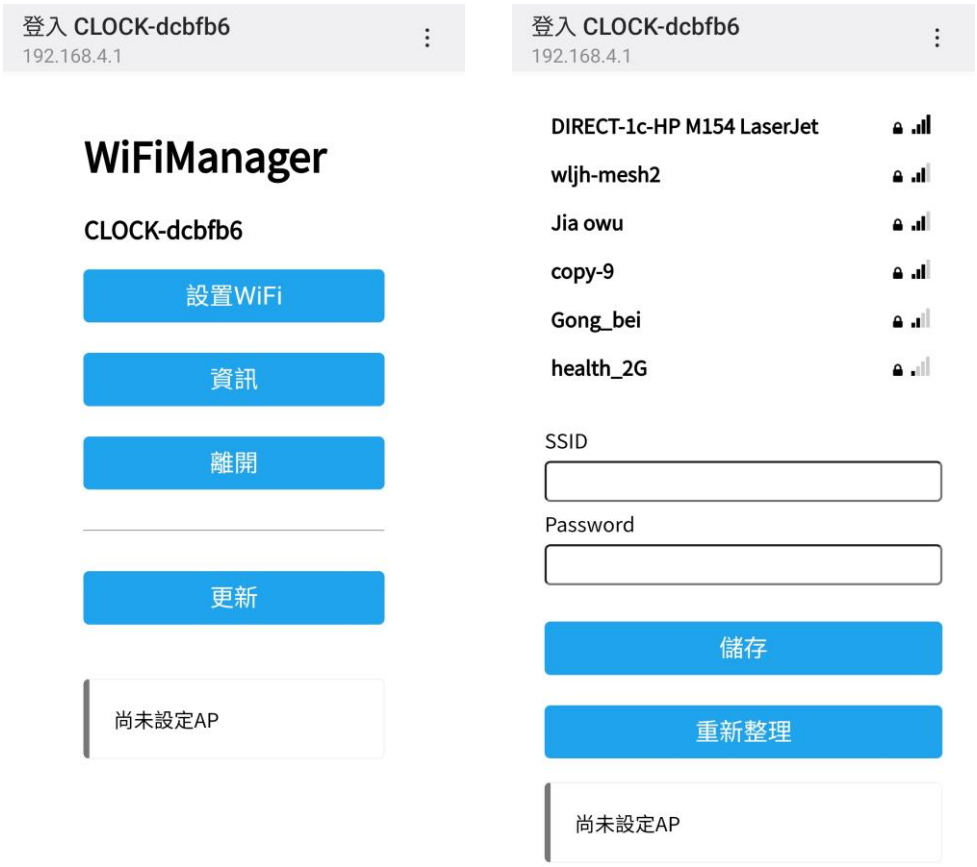
控制板非常容易當機，極度不穩定且時常會重開機。後來經過分析後，發現我們因為太多的 API 範例都是使用 Json 格式，導致思維陷入盲點，其實 API 取得的資料不一定只限定為 Json 格式，只要將資料下載的格式改成 xml 格式，即可減少大量的資料傳輸，進而解決此問題。但是同樣使用 Json 格式下載中央氣象局的資料，卻不會有此資料量過多的問題，可見兩個政府機構的 API 系統是有所差異的。


3. 中文顯示問題：最初的作品，因為只使用一組 32x8 的「MAX7219 LED 點矩陣顯示模組」，所以呈現方式只限英文與數字，但我們更希望能以中文的方式呈現，讓同學們一目了然。但因中文字型較複雜，8 個點的高度是無法完美呈現的，至少要 16 點的高度。而市面上買不到 32x16 的 MAX7219 LED，所以改採用 2 組 MAX7219 LED 串接的方式，形成 32x16 的點陣面，使中文顯示沒有問題。硬體方面解決後，接著就是程式碼的問題了。MD_Parola、MD_MAX72xx 等函式庫都是由外國人開發，所以其字型並不含中文字體，所幸透過網路上的資源，利用取模軟體及 MD_Parola 字型工具，我們成功的將所需的中文字，一個字一個字的拆分為上下兩部分，加入原本的 2 個字型中，上面那塊 32x8 的 MAX7219 LED 讀取上半部的中文字，下面那塊 32x8 的 MAX7219 LED 讀取下半部的中文字，上下組合起來，就是一個完整的中文字了。



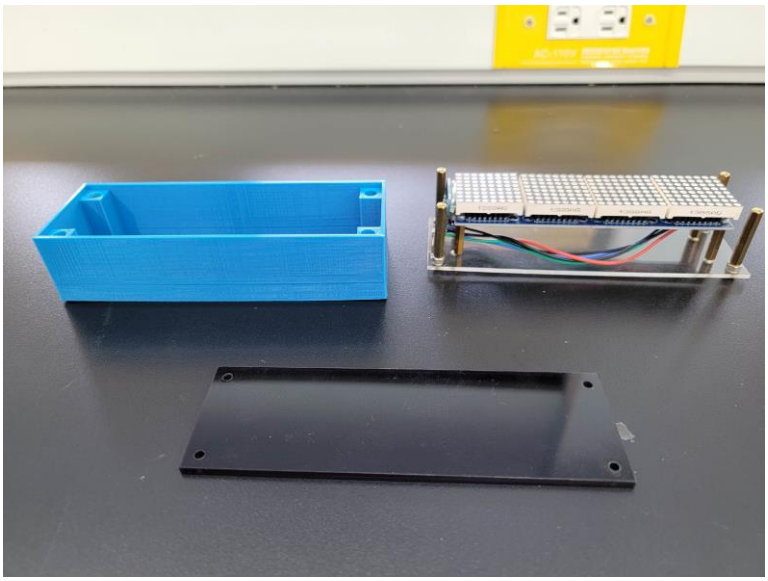


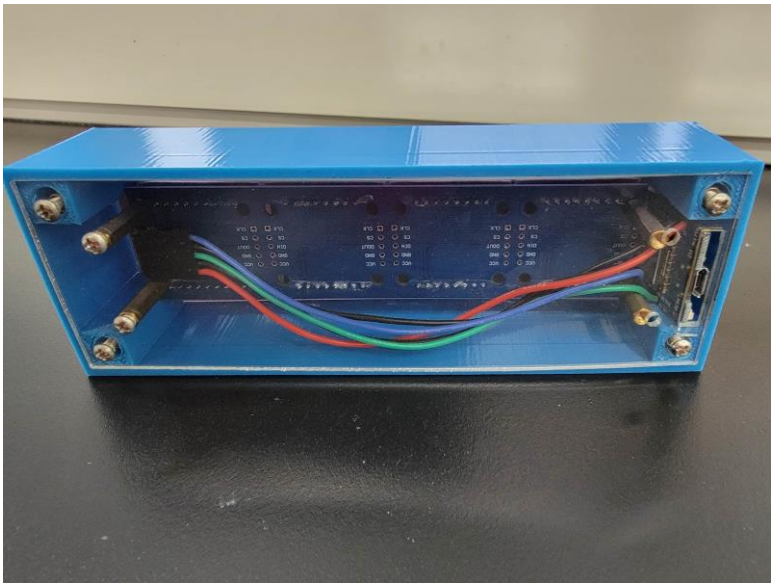
4. 連接 wifi 問題：D1 mini 控制板採用 WIFI 來連接網路，設計程式時須將連接的 ssid 及密碼，一同寫入程式碼並燒錄進 D1 mini 控制板，才能順利連接網路。但我們的設計目的是希望能推廣至各校，而各校的 WIFI 環境是不同的，如果每次換個環境就要進程式碼裡修改 ssid 及密碼，然後再燒錄進 D1 mini，這樣的方式是非常不方便的，且操作者也須有 arduino 的基礎才行。所以我們找到一個 WiFiManager 函式庫，該函式庫提供一個網頁介面，讓使用者可以透過各種連網裝置，隨時修改 ssid 及密碼，提升方便及易用性。



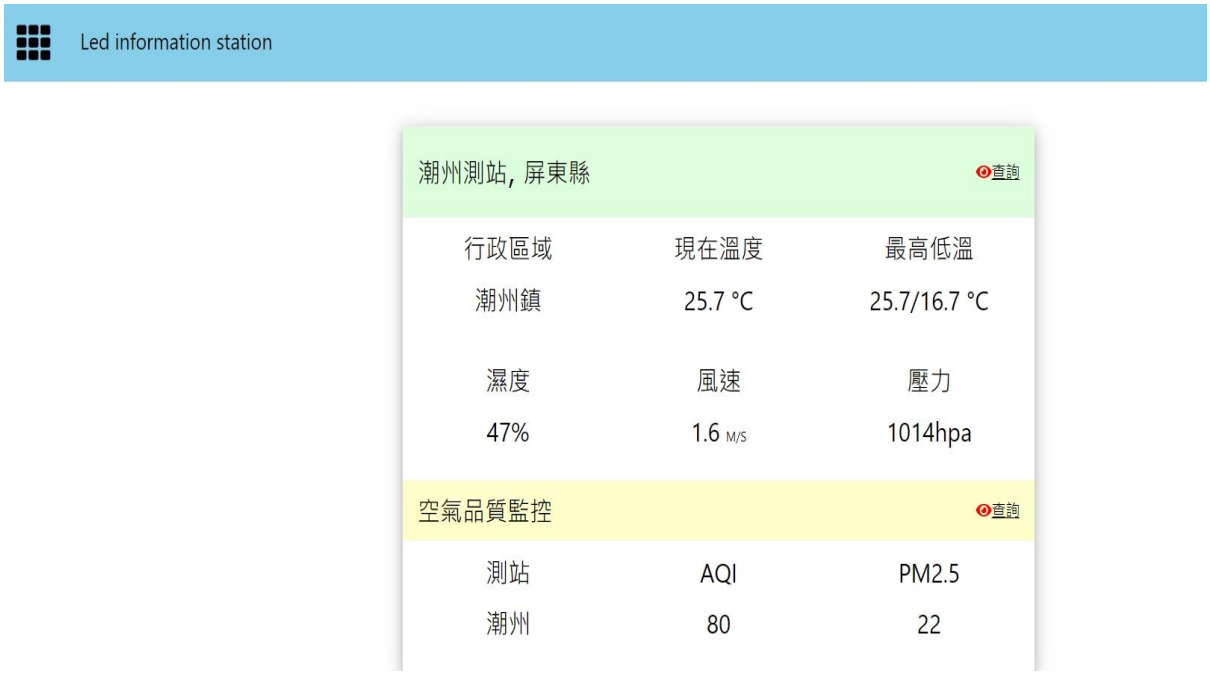

(四)、 實機組裝及測試

1. 組裝測試：





2. 網頁控制介面：


Led information station
潮州測站, 屏東縣
查詢
行政區域
現在溫度
最高低溫
潮州鎮
25.7 °C
25.7/16.7 °C
濕度
風速
壓力
47%
1.6 M/S
1014hpa
空氣品質監控
查詢
測站
AQI
PM2.5
潮州
80
22


# 肆、 研究結果

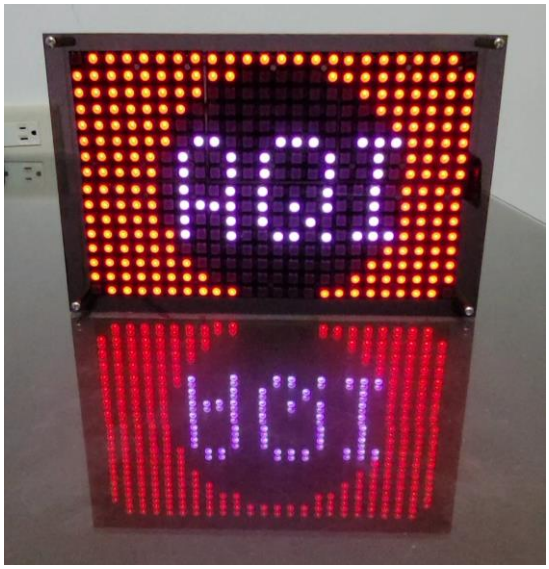
一、 本次共製作出 3 種作品，作品一使用 32x8 的 MAX7219 LED，所以僅可以呈現出英文及數字，但是藉由 MD_Parola 函式庫的支援，使作品一的畫面呈現變化多端的風格，各種有趣的圖案穿插出現，能夠吸引同學們的目光，讓同學們駐足觀看有關 AQI 的資訊，且此作品小巧可愛，攜帶方便，適合帶至各種教學現場展示。


二、 作品二使用 2 組 32x8 的 MAX7219 LED 串接成 32x16 的 MAX7219 LED 面板，所以可以顯示中文訊息，因此我們將 AQI 數值所代表的含意以跑馬燈的方式呈現，使同學們知道目前的空氣品質對人體的影響為何，進而針對自身身體狀況，採取應變措施。但可惜的是 MAX7219 LED 的色彩是單色的，無法配合展示 AQI 燈號的顏色變化，所以我們又加裝一個 LED 燈環，配合不同的 AQI 數值，呈現不

同的燈號顏色。



三、 作品三的設計，是為了改善 MAX7219 LED 的單色限制，所以 LED 方面全部改採 WS2812 LED 面板，此種面板經由程式控制，可呈現五顏六色的色彩，故可配合 AQI 燈號的顏色變化。作品三是由 6 塊 8x8 WS2812 LED 串接組成 24x16 的面板，體積較大，但同樣可顯示中文訊息，再搭配顏色的變化，可讓觀看者完整了解 AQI 相關訊息。



四、 部分程式碼

```
// ========================================================================
// 獲取 AQI
// ========================================================================
void getAQI() {
  WiFiClientSecure client;        //宣告使用 client 物件
  client.setInsecure();           //設定 https 協定安全性，不用經過身分認證即可通訊
  HTTPClient https;               //宣告使用 https 物件
  String url = "https://data.epa.gov.tw/api/v2/aqx_p_432?filters=SiteName,EQ," + String(AIRZONE) + "&api_key=";
  String data = EPAAPIKEY;
  String url2 = "&fields=sitename,aqi,status,pm2.5,publishtime&format=xml";      //下載格式為 xml，因為如使用
json 格式，會下載許多多餘的資料，占太多記憶體，導致 D1 mini 接收資料發生錯誤
  String fullUrl = url + data + url2;     //連線主機網址
  Serial.println("Requesting " + fullUrl);              //在監控視窗中列印訊息
  if (https.begin(client, fullUrl)) {             //與主機連線
    int httpCode = https.GET();               //從主機端傳回連接訊息
    Serial.println("============== Response code: " + String(httpCode));              //在監控視窗中列印訊息
    if (httpCode > 0) {                 //如果連接成功
      Serial.println(https.getString());                  //在監控視窗中列印訊息
      String line2 = https.getString();                   //將下載的資料存在 line2 變數中

      //解析下載到的 xml
      if (line2.indexOf("SITENAME") >= 0) {       //取得測站名稱
        SiteName = line2.substring(line2.indexOf("<SITENAME>") + 10, line2.indexOf("</SITENAME>") );
        SiteName.trim();
        Serial.println("SiteName2: " + SiteName);
      }
      if (line2.indexOf("AQI") >= 0) {                  //取得 AQI
        AQI = line2.substring(line2.indexOf("<AQI>") + 5, line2.indexOf("</AQI>"));
        //取得<AQI>與</AQI>之間的資料
        AQI.trim();           //清除空格
        Serial.println("AQI2: " + AQI);        //在監控視窗中列印訊息
      }
      if (line2.indexOf("PM2.5") >= 0) {                //取得 PM2.5
        PM25 = line2.substring(line2.indexOf("<PM2.5>") + 7, line2.indexOf("</PM2.5>"));
        PM25.trim();
        Serial.println("PM2.5: " + PM25);
      }
      if (line2.indexOf("PUBLISHTIME") >= 0) {          //取得 AQI 更新時間
```

```
            PublishTime = line2.substring(line2.indexOf("<PUBLISHTIME>") + 13, line2.indexOf("</PUBLISHTIME>"));
            PublishTime.trim();
            Serial.println("PublishTime: " + PublishTime);
          }
        }
        https.end();                          //關閉 https
    } else {
        Serial.printf("[HTTPS] Unable to connect\n");                    //在監控視窗中列印訊息
    }
  }
```

在 LED 上顯示 AQI 訊息

```
    if (SHOW_AQI ) {

        P.displayAnimate();                //偵測 LED 顯示狀態
        if (P.getZoneStatus(0))            //如果上一個 LED 訊息顯示完成
        {
            String t,t1,t2,t3 ;            //定義變數
            for (int i = 0; i < 6; i++) {              //使用迴圈
                t = t + String(char(i + 126));     //整合中文字串: t= "空氣品質指標"
            }
            //t2=代表目前空氣品質：
            t2 = String(char(132)) + String(char(133)) + String(char(134)) + String(char(135)) + String(char(126)) +
String(char(127)) + String(char(128)) + String(char(129)) + ": " ;
            t = t + " AQI: " + AQI + "," + t2;
            //整合中文字串：t= "空氣品質指標 AQI:OO 代表目前空氣品質："
            if (AQI.toInt() < 51)
            {
                t1 = "代表目前空氣品質：良好";
                t3 = String(char(136)) + String(char(137))    ; //t3=良好
                //點亮光帶所有燈珠-START
                fill_solid(leds, NUM_LEDS, CRGB::Green);
                FastLED.show();
                //點亮光帶所有燈珠-END
            }
            if (AQI.toInt() > 50 && AQI.toInt() < 101)
            {
                t1 = "代表目前空氣品質：普通";
                t3 = String(char(138)) + String(char(139))    ; //t3=普通
                //點亮光帶所有燈珠-START
```

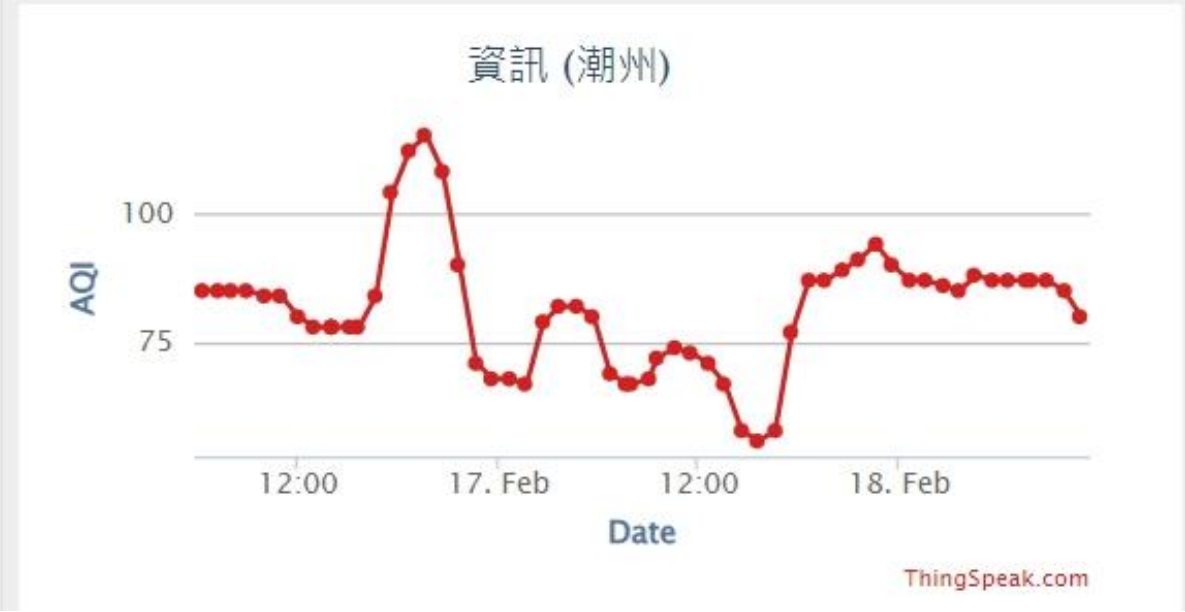
```
            fill_solid(leds, NUM_LEDS, CRGB::Yellow);   //點亮光帶所有燈珠
            FastLED.show();
            //點亮光帶所有燈珠-END
                }
        if (AQI.toInt() > 100 && AQI.toInt() < 151)
        {
            t1 = "代表目前空氣品質:對敏感族群不健康";
            t3 = String(char(140)) + String(char(141)) + String(char(142)) + String(char(143)) + String(char(144)) +
String(char(145)) + String(char(146)) + String(char(147))   ; //  對敏感族群不健康
            //點亮光帶所有燈珠-START
            fill_solid(leds, NUM_LEDS, CRGB::OrangeRed);   //點亮光帶所有燈珠
            FastLED.show();
            //點亮光帶所有燈珠-END
        }
        if (AQI.toInt() > 150 && AQI.toInt() < 201)
        {
            t1 = "代表目前空氣品質:對所有族群不健康";
            t3 = String(char(140)) + String(char(148)) + String(char(149)) + String(char(143)) + String(char(144)) +
String(char(145)) + String(char(146)) + String(char(147))   ; //  對所有族群不健康
            //點亮光帶所有燈珠-START
            fill_solid(leds, NUM_LEDS, CRGB::Red);   //點亮光帶所有燈珠
            FastLED.show();
            //點亮光帶所有燈珠-END
        }
        if (AQI.toInt() > 200 && AQI.toInt() < 301)
        {
            t1 = "代表目前空氣品質:非常不健康";
            t3 = String(char(150)) + String(char(151)) + String(char(145)) + String(char(146)) + String(char(147))   ; //
非常不健康
            //點亮光帶所有燈珠-START
            fill_solid(leds, NUM_LEDS, CRGB::Purple);   //點亮光帶所有燈珠
            FastLED.show();
            //點亮光帶所有燈珠-END
        }
        if (AQI.toInt() > 300 && AQI.toInt() < 501)
        {
            t1 = "代表目前空氣品質:危害";
            t3 = String(char(152)) + String(char(153))   ;//  危害
            //點亮光帶所有燈珠-START
            fill_solid(leds, NUM_LEDS, CRGB::Chocolate);   //點亮光帶所有燈珠
```

```
            FastLED.show();
            //點亮光帶所有燈珠-END
        }
        t = t +   t3;     //整合中文字串
        showText2(0, t, PA_CENTER, 50, 500, PA_SCROLL_LEFT, PA_SCROLL_LEFT);    //LED 顯示中文上半部
        showText2(1, t, PA_CENTER, 50, 500, PA_SCROLL_LEFT, PA_SCROLL_LEFT);    //LED 顯示中文下半部
        // synchronise the start
        P.displayClear();            //清除 LED 顯示內容
        P.synchZoneStart();        //同步上下半部的顯示


    }
}
```

# 伍、 討論

一、 穩定性驗證：本次設計的作品，是為了可以代替布製的「空氣品質旗幟」，需長時間進行展示，其穩定性非常的重要。所以是否可以每天 24 小時不間斷地通電使用，是我們迫切想了解驗證的。但我們不可能半夜不休息盯著作品看，在經過構思之後，我們添加一段程式碼，將得到的 AQI 數據上傳至 ThingSpeak 雲端服務器進行記錄，透過數據圖形化，我們可觀察到作品的運作狀態，並從而驗證我們的作品，確實可以肩負週一至週五，連續五天 24 小時運作的期待(六日休息)。




二、耗電量與醒目性比較：作品三使用功率約為 1.7W，作品二使用功率約為 1W，與 32 吋液晶電視機約 50-100W 的功率相比，做為資訊呈現的用途，是非常省電的。最近

發現學校有一台液晶電視會顯示太陽能發電、天氣資訊及 AQI 資訊。我們的作品取之相比，除了非常省電，符合節能減碳的環保概念外，字體也非常大，站在遠處即可觀看了解 AQI 資訊，對於敏感族群的同學們來說，產生的實質作用更大。

三、 相關費用：三種作品的費用如下表

| 類型 | 使用元件 | 總金額 |
|---|---|---|
| 作品一 | D1 mini 控制板 (90)<br>MAX7219 LED 模組 (155)<br>M3 銅柱+螺絲 (40)<br>壓克力 (60) | 345 |
| 作品二 | D1 mini 控制板 (90)<br>MAX7219 LED 模組 (155x2=310)<br>M3 銅柱+螺絲 (40)<br>壓克力 (60) | 500 |
| 作品三 | WS2812B LED 模組 (113x6=678)<br>D1 mini 控制板 (90)<br>M2 銅柱+螺絲 (40)<br>壓克力 (60) | 868 |

四、 3D 列印機的局限性：作品三的尺寸較大，所以一般的 3D 列印機無法列印出這麼大的外殼，除非採用分割列印後，再組裝的方式，但如此一來就無法一體成形，完整性略差。當然 3D 列印成品的組裝也有更進階的方式，讓人看不出是組裝的，但這就不在我們的學習範圍內了。

五、 站在巨人的肩膀看世界：在作品設計過程中，使用了各種開源函式庫實現許多功能，使程式的學習與設計變的更容易，也節省許多開發的時間，這些都仰賴無數程式開發者的無私貢獻與付出，也因為踩著前人的腳步，我們後輩才有更多進步的空間。

# 陸、 結論

本組運用 D1 mini 控制板＋LED 模組，並配合 Arduino IDE 開發工具，成功設計出一款「數位自動化空氣品質旗幟」作品，此作品不僅能即時更新 AQI 資訊，免去人工更換旗幟的不便，並將「空氣品質指標 AQI」所代表的含意以及對人體健康的影響，直接呈現出來。而且非常的醒目，在遠處即可觀看到。本作品也可輕易推廣至學校或家庭，讓敏感族群能夠依此訊息，提早應變，真正發揮「空氣品質旗幟」的作用。

# 柒、 參考資料及其他


一、環保署環境資料開放平臺・取自

https://data.epa.gov.tw/

二、氣象資料開放平臺・取自

https://opendata.cwb.gov.tw/index

三、Arduino 基本語法筆記・取自

http://yhhuang1966.blogspot.com/2015/09/arduino_14.html

四、tzapu・WiFiManager・取自

https://github.com/tzapu/WiFiManager

五、MajicDesigns・Library for modular scrolling LED matrix text displays・取自

https://github.com/MajicDesigns/MD_Parola

六、Espressif Systems・WiFiClientSecure・取自

https://github.com/espressif/arduino-esp32/tree/master/libraries/WiFiClientSecure