東海大學資訊工程學系

專題報告書

樹莓派空氣汙染監測系統之實作

Raspberry Pie Air Pollution Monitoring System For The Implementation

中華民國107年6月8號

致謝

首先要感謝我們指導教授的細心指導，教授在觀念上給予我們很多的建議，進行很多次的開會討論，我們才可以有今天的成果。

在平常的討論中，教授告訴我們很多需要改進的細節，這些都是我們沒有注意到的。

此外，當我們遇到問題與困難的時候，教授總是給了我們很多的建議與方法，讓我們在專題研究上面能夠如此順利。

教授也時常和我們分享許多以前帶專題生的經驗，讓我們受益良多。謝謝老師的教誨，在此致上我們的感謝。

在這段專題時間，要謝謝學長學姐在百忙之中抽空指導我們，在專題上給了我們很多的建議，讓我們的方向更明確，我們有很多的小問題無法解決時，學長學姐總是能夠給我們很好的建議。

除此之外，因為我們是第一次碰這種機器，所以有很多不了解的地方，學姐在機器的操作上也給予我們很多的指導，讓我們能夠很快的進入狀況，所以我們要特別感謝學姐，如果沒有學姐的幫忙，我們可能會遇到更多技術上的問題。謝謝學長學姐的幫忙，使我們能夠順利的完成專題。

中華民國107年4月26號

摘要

隨著科技的進步、生活型態的轉變，衛生環境的改善是現代人所注重的觀念。

希望藉由資訊科技掌握室外環境，對環境的監控提出預測與警告。

因此我們實作了空氣汙染監測系統，來收集有關室外環境的氣體資料。

結合現行運作中的MakerProject──「LASS開源公益環境感測網路」，開發更低成本的感測終端，並將該LASS的公開數據導入我們的資料庫，來進行整合呈現。

目前使用的感測器是屬於偵測空氣品質PM1、PM2.5、PM10、溫度和濕度，此次研究著重於室外環境上的空氣感測，但是此專案架構設計可應用於任何形式的感測資料。

除了過往的數據比較呈現，系統還整合了IFTTT來達到即時警示的功能，使人使用者免於受到室外汙染物的影響。

本系統包含伺服器、系統、感測終端三部分。在伺服器，我們部署了一個Arch Linux 加樹莓派作為中央伺服器，處理來自感測終端及LASS的參數資料。

系統的部分，則透過Python將LASS資料存入資料庫，並使用網頁平台呈現使用者介面，以達到跨平台的優勢。

至於感測終端，則是使用Arduino來部屬，透過WIFI將資料傳送至LASS伺服器和本專案的樹莓派伺服器。

經過修改，其成本雖比市面上的LASS感測器要來的低廉，但是卻可以達到LASS專案及本專案的雙重使用需求。

利用上述的方法，讓我們可以在任何時間、任何地點只要透過網際網路就可以監控環境，輕易取得環境相關資訊，藉此來做為環境改善的判斷依據，讓我們能夠生活在更舒適的環境。

關鍵詞：感測器、即時監控、環境監控、樹莓派

**目錄**

[第一章 緒論 1](#_Toc512521317)

[1.1 研究動機 1](#_Toc512521318)

[1.2 研究目的 2](#_Toc512521319)

[1.3 研究方法 2](#_Toc512521320)

[1.4 研究流程 3](#_Toc512521321)

[第二章 相關文獻回顧 4](#_Toc512521322)

[2.1 空氣污染 4](#_Toc512521323)

[2.1.1 污染物(當中顆粒物是我們專題主要目標)........4](#_Toc512521324)

[2.2 建構氣象資料監視系統 6](#_Toc512521325)

[2.3 LASS環境感測網路系統 7](#_Toc512521326)

[第三章 環境監控系統 8](#_Toc512521327)

[3.1 研究方法 8](#_Toc512521328)

[3.2 系統架構 9](#_Toc512521329)

[3.2.1系統架構....................................9](#_Toc512521330)

[3.2.2環境監控資料接收流程.......................10](#_Toc512521331)

[3.3實驗環境架設....................................10](#_Toc512521332)

[3.4相關開發技........................................14](#_Toc512521333)

[3.4.1 Angular (Angular 2) ..........................14](#_Toc512521334)

[3.4.2 MySQL.....................................14](#_Toc512521335)

[3.4.3 無線感測網路（Wireless Sensor Networks,WSN）…14](#_Toc512521336)

[第四章 實作分析............................................16](#_Toc512521337)

[4.1實驗設計..........................................16](#_Toc512521338)

[4.2系統功能說明......................................16](#_Toc512521339)

[4.3資料量問題........................................2](#_Toc512521340)1

[4.4 IFTTT串接設計....................................2](#_Toc512521340)2

[4.5 Libsodium加密庫..................................2](#_Toc512521340)4

[第五章 結論.............................. .................](#_Toc512521341)25

[5.1結論 ..............................................25](#_Toc512521342)

[5.2心得..............................................2](#_Toc512521343)5

[附錄 .......................................................2](#_Toc512521344)7

[參考資料 ...................................................56](#_Toc512521345)

**圖目錄**

圖 3-1系統架構圖...................................................9

圖 3-2 Bundle - Pi3 and Red/White Official Case ZigBee.........11

圖 3-3 樹莓派電路板實體圖 .........................................11

圖3-4 樹莓派示意圖............................................... . 11

圖3-5 攀藤pm2.5/温湿度二合一传感器 PMS5003T ......................12

圖 3-6 ZigBee瑞昱 Ameba RTL8710AF 開發板.......................12

圖 3-7 ArchLinux....................................................13

圖 3-8 感測網路系統基本架構........................................14

圖 4-1 Google地圖測站資訊 .......................................16

圖 4-2 手機版響應式設計............................................17

圖 4-3 總覽資訊頁面................................................18

圖 4-4 當下測站資訊................................................18

圖 4-5 歷史數據....................................................19

圖 4-6 設定頁面....................................................19

圖 4-7 平均數據....................................................20

圖 4-8 比較數據....................................................21

圖 4-9 IFTTT設定...................................................23

圖 4-10 IFTTT串接Pushbullet到手機..............................24圖 4-11 IFTTT資料庫密碼加密........................................24

**表目錄**

表 2-1空氣汙染程度表......................................................6

# 緒論

## 研究動機

東海擁有佔地廣闊的廣大校園，校園內除了精心設計的各式建築外，多的是樹木和綠地，宛如一座小森林，每逢假日都會吸引很多遊客來學校裡朝聖，樹木之眾，更讓東海有「台中之肺」的美稱。

但是在近年來的新聞報導裡，我們卻驚覺原來東海的空氣汙染是台灣最嚴重的地區，與印象中的綠地森林截然不同。

其實只要我們打開地圖，也就不難發現東海為何是台灣空污最嚴重的地方了。

東海北側除了有火葬場，也緊鄰交通繁忙的台灣大道。

南側更與台中工業區為鄰。

西側不但有國道三號，更有台中火力發電廠。

東側則有國道一號。在東海周遭，有各種大大小小的汙染源，也難怪東海的空氣品質會這麼糟糕。

但

究竟，問題出在哪？

究竟，我們的不滿，有誰聽見了嗎？

究竟，我們是不是只能自認倒楣呢？

這次就讓我們勇敢的向空污說不。

因為如此，我們設計了這個專題除了因為我們是東海大學的一員，更是為了學校將來的學弟妹們及教職員們的身體健康。

## 1.2研究目的

為此，我們希望能研發出一套環境監控系統，為生活環境出一份力。此專案應包含整套的感測器及監控網路，並接受異質平台感測器的數據加入。

除了可以單獨架設，應可兼容其他現有的感測網路，讓我們可以以最少的成本來達到專題目的。

## 1.3研究方法

因為專題的預想狀況下附載不高，於是我們選擇Raspberry Pi 3B作為中央伺服器，以節省裝置成本及用電成本。

作業系統選用Arch Linux，輕量化且持續更新；網頁伺服器採用LAMP (Linux + Apache + MySQL + PHP)，為主流架構之一，可靠穩定。

之所以不選用Raspbian，是考慮到了將來移植的方便性。若未來面臨樹莓派效能不敷使用的情況，規劃初始便選用常規Linux系統可大大的減少移植成本。

此套系統設計完全可以轉移到正規的PC或是伺服主機上運作。此專題接受LASS專案及ThingSpeak (一流行的IOT接口服務) 的數據輸入，未來希望能再加入Edimax airbox、Probecube、EPA、Airq等其他專案的支援。

除了外來數據源之外，我們也設計了自己的感測終端，可以在屏除其他專案支持的情況下運作。感測終端使用Arduino Ameba RTL8710 + 攀藤PMS5003T，是兼顧規格和成本的最佳選擇。

感測終端設計可以連接此專案伺服器、直接上傳ThingSpeak、符合LASS規格(MQTT連接LASS Server)、定時網路校時、WIFI斷線重連等功能。

系統採用網頁平台，為的是可以跨平台使用，並搭配響應式網頁設計，在電腦、平板、手機都能有良好使用體驗。

功能設計預計有「使用者帳號註冊\登入\各種設定」、「即時數據」、「歷史數據」、「地圖總覽」、「測站比較」、「平均數據」等。

開啟網頁首先會看到最靠近之測站即時數值，並且在地圖上可點擊查看其他測站數值。在「歷史數據」頁面可查詢歷史數據，以折線圖和表格來呈現。「測站比較」頁面使用者可將不同測站不同時間的折線圖繪製在一起，方便研究比較。

「平均數據」可顯示測站在一段時間之平均數值，讓使用者一目瞭然這段時間的空氣品質。

若進行帳號註冊並登入，將提供使用者設定欲觀測的測站。只要維持登入，將預設顯示這些測站資料。另外提供警示功能，在觀測之測站數值超標時透過email或IFTTT服務推送警告。

## 1.4研究流程

專題分為三部分同時進行：樹莓派部分、感測終端部分，及網頁部分。

1. 樹莓派：灌入Arch Linux並架設LAMP。

2. 感測終端：硬體外殼、PMS5003T感測器連接測試、WIFI連接、MQTT

server連接、http request推送測試、NTP校時等。

3. 網頁部分：著重前端，採用AngularJS 2框架開發，透過PHP連接MySQL

資料庫。包括後端資料庫建置、資料庫PHP連接程式設計，前端版型

調整、各頁面撰寫、折線圖表外掛測試、地圖嵌入撰寫測試等。

# 相關文獻回顧

# 2.1空氣污染

空氣污染（或大氣污染）指一些危害人體健康及周邊環境的物質對大氣層所造成的污染。

這些物質可能是氣體、固體或液體懸浮物等。

我們在日常生活中呼吸的空氣，由多種化學物質組成，最普遍的元素是氮，其次是氧，然後是其他氣體。

每種氣體的成份並不是固定的，會有輕微的轉變，當轉變出來的物質過多時，我們就會視它為空氣污染。

如果空氣中的污染物數量少的話，對人體和環境的影響會比較輕微，但當這些污染物增加至危險的水平，我們就要想辦法把他們從空氣裡消除。

空氣污染主要可以分為化學污染和生物污染兩部份。

也有人把噪音、熱量、輻射和光的污染歸入空氣污染的類別裡。

2008年，布萊克史密斯研究所在世界污染最嚴重地區報告中將室內空氣污染和城市空氣品質被列為世界最嚴重的有毒污染問題。

根據2014年世界衛生組織報告，2012年空氣污染導致全球700萬人死亡。

### 2.1.1污染物(當中顆粒物是我們專題主要目標)

顆粒物，或稱懸浮微粒(particulate matter,PM)、大氣微粒物質，是懸浮在空氣中微小的固態或液態粒子。

與之相對的是氣膠，常常指微粒和氣體的混合物。一些顆粒是自然生成的，如火山爆發、沙塵暴、森林或草地火災、植物或海上浪花。人類活動，如機動車燃燒化石燃料、電廠或其它工業過程都會產生大量氣膠。從世界平均水平來看，由人類活動而產生的氣膠只占大氣中的10%。

空氣中顆粒物增加與健康威脅如心臟病、肺病有關。懸浮物通常以顆粒的大小來區分，而當中的可吸入懸浮粒子則可影響人體健康。

其中，空氣動力學直徑（以下簡稱直徑）小於等於10微米的顆粒物稱為可吸入顆粒物（PM10）；直徑小於等於2.5微米的顆粒物稱為細顆粒物（PM2.5）。顆粒物能夠在大氣中停留很長時間，並可隨呼吸進入體內，積聚在氣管或肺中，影響身體健康。

這些可吸入懸浮顆粒還可以再細分為PM10和PM2.5兩類：

PM10（即直徑小於10微米）的顆粒，這種顆粒不能被身體的防禦機制阻擋，可以直達肺部，所以十分危險；

PM2.5的顆粒比PM10的更危險，因為它的顆粒小於2.5微米，可以穿透肺泡到達血液。

直徑約為0.1mm的粉塵顆粒有超過50%會沉積在肺部，含矽的粉塵更會對人體造成永久傷害，例如引致矽肺病。香菸塵是常見的懸浮顆粒，因粒徑小（在0.001至0.1mm之間）、擴散力強，在靜止空氣中幾乎可以不沉落，不僅即時可見污染惡果，長期吸入更可以導致肺癌。

其他常見的懸浮顆粒：

矽、鋁、鈣、錳、鐵（來源：泥土、地殼和建築活動）

釩、鎳（來源：以油作為燃料的燃燒工序）

鎘、碳氫化合物（來源：焚化）

碳、鉛、溴、碳氫化合物（來源：汽車廢氣）

鈉、氯、鎂、鉀（來源：海洋）

硝酸鹽、硫酸鹽、銨（來源：次生污染形成物）



表 2-1空氣汙染程度表

# 2.2建構氣象資料監視系統

在我讀完由淡江大學資訊工程學系碩士在職專班林倚萱研究生在107年出版的論文，她的論文主要是再說空氣污染的問題，因為和我們的專題有某些方面是類似的，所以我們有深入了解一下，林倚萱研究生的論文主要再說使用ESP8266來監視氣象資料的變化，利用此氣象監視系統持續監控氣象相關資訊並回報到使用者手機或提供給使用者由web查詢，給我們得啟發是我們不能是只有看大範圍的環境變化，也需要看每個小範圍的空氣汙染的精確變化，所以我們的專題有將這方面特別強調出來，並且架設屬於我們自己的監控網路平台，並且我們有將我們的網站寫成適合手機，平板及電腦螢幕不同會自動調整。

# 2.3 LASS環境感測網路系統

LASS（Location Aware Sensing System）是一個社群，以開源公益非營利的形式，設計與實現環境感測系統，藉由網路鄉民大家合力，以公民科學的型態，進行環境感測等相關的活動，希望能對周圍的環境，給出具體的幫助。

如前所述，LASS是一套具備環境感測器的網路系統架構，而且要讓Maker有能力自製所需要、且能與系統相容的感測器套件，再將感測到的資料提供給任何使用者。

為此，我們需要提供一套完成度夠高的系統雛型，以賦予所有使用者俱備監測全部感測器的能力。

# 第三章 環境監控系統

# 3.1研究方法

本實驗使用懸浮微粒感測器和溫溼度感測器，並整合現行其他專案的公開資料，讓使用者能透過網路監控各個場所的溫濕度、PM1、PM2.5與PM10濃度等，並透過網頁進行數據比較，或是以此做為依據改善生活品質。

因為專題的預想狀況下附載不高，於是我們選擇Raspberry Pi 3B作為中央伺服器，以節省裝置成本及用電成本。

作業系統選用Arch Linux，輕量化且持續更新；網頁伺服器採用LAMP (Linux + Apache + MySQL + PHP)，為主流架構之一，可靠穩定。

之所以不選用Raspbian，是考慮到了將來移植的方便性。

若未來面臨樹莓派效能不敷使用的情況，初始規劃便選用常規Linux系統可大大的降低移植成本。

此套系統設計完全可以轉移到正規的PC或是伺服主機上運作。

此專題接受LASS專案及ThingSpeak (一個流行的IOT接口服務) 的數據輸入，未來希望能再加入Edimax Airbox、Probecube、EPA、Airq等其他專案的支援。

除了外來數據源之外，我們也設計了自己的感測終端，可以在屏除其他專案支持的情況下運作。

感測終端使用Arduino瑞昱Ameba RTL8710 + 攀藤PMS5003T溫濕度二合一感測器，是兼顧規格和成本的最佳選擇。

感測終端設計可以連接此專案伺服器、直接上傳ThingSpeak、符合LASS規格(MQTT連接LASS Server)、定時網路校時、WIFI斷線重連等功能。

系統採用網頁平台，為的是可以跨平台使用，並搭配響應式網頁設計，在電腦、平板、手機都能有良好使用體驗。

功能設計預計有「地圖首頁」、「總覽」、「歷史數據」、「測站比較」、「平均數據」、「帳號註冊\登入\各種設定」等。

開啟網頁首先會看到最靠近之測站的即時數值，並且在地圖上可點擊查看其他測站數值。

在「歷史數據」頁面可查詢歷史數據，以折線圖和表格來呈現。

「測站比較」頁面使用者可將不同測站不同時間的折線圖繪製在一起，方便研究比較。

「平均數據」可顯示測站在一段時間之平均數值，讓使用者一目瞭然這段時間的空氣品質。

若進行帳號註冊並登入，將提供使用者設定欲觀測的測站。

只要維持登入，將預設顯示這些測站和資料。

另外提供警示功能，在設定測站數值超標時透過 IFTTT服務推送警告。

## 系統架構

### 3.2.1系統架構圖

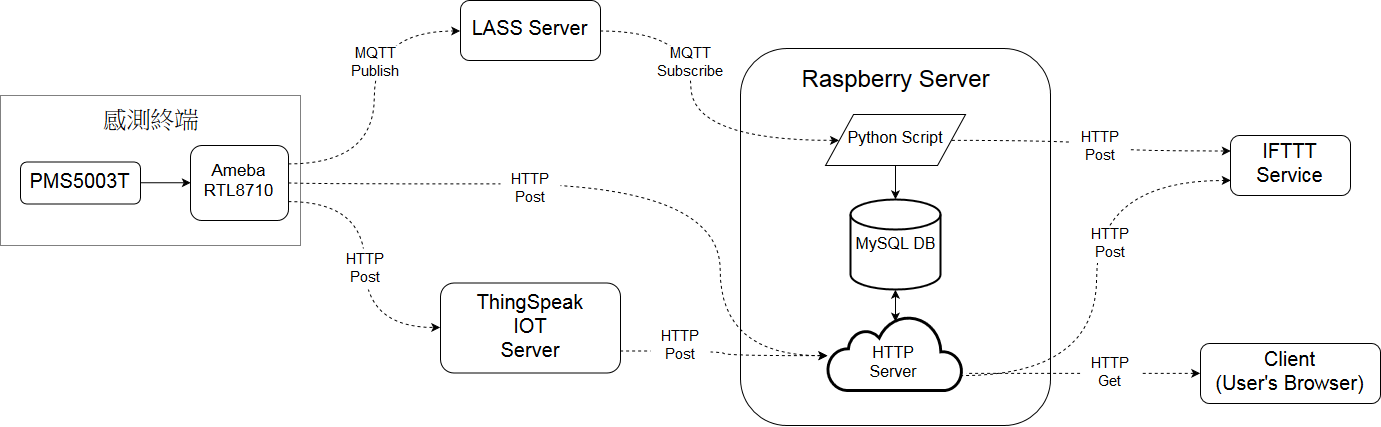


圖 3-1系統架構圖

### 3.2.2環境監控資料接收流程

瑞昱 Ameba RTL8710 本身是一整合了WIFI模組的Arduino開發版，在處理攀藤PMS5003T的感測資料後透過網路將資料送回樹莓派伺服器及其他網路主機。

伺服器端使用Python接收TCP和MQTT的各式資料，再寫進MySQL資料庫中，最後呈現到網頁上。

網頁採用響應式設計，方便跨平台使用需求。

當伺服器偵測到設定測站存在異常數值時，將會主動提出警告訊息通知。

我們著重在環境資料整合上。

除了自己的資料回傳外，還能整合進其他專案的感測資料，對廣範圍室外空氣進行有效的監測，並以此提出警告，使人免於受到室外環境汙染物的影響。

## 3.3實驗環境架設

**硬體需求**

* **Raspberry Pi：**樹莓派（英語：Raspberry Pi），是一款基於Linux的單晶片電腦。

它由英國的樹莓派基金會所開發，目的是以低價硬體及自由軟體促進學校的基本電腦科學教育。

樹莓派配備一枚博通（Broadcom）出產的ARM架構700MHz BCM2835處理器，256MB記憶體（B型已升級到512MB記憶體），使用SD卡當作儲存媒體，且擁有一個Ethernet、兩個USB介面、以及HDMI（支援聲音輸出）和RCA端子輸出支援。

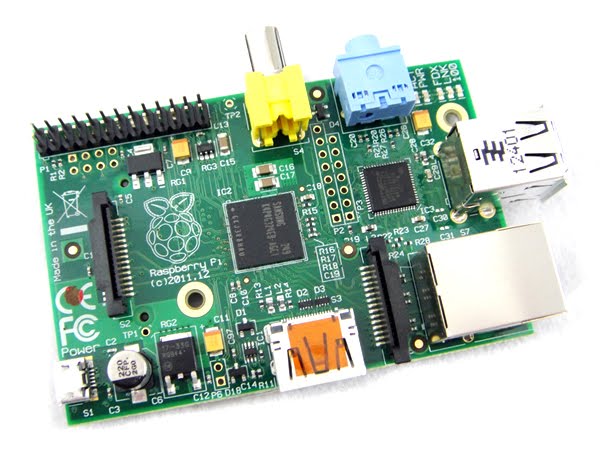
樹莓派面積只有一張信用卡大小，體積大概是一個火柴盒大小，可以執行像《雷神之鎚III競技場》的遊戲和進行1080p影片的播放。

操作系統採用開源的Linux系統：Debian、ArchLinux，自帶的Iceweasel、KOffice等軟體，能夠滿足基本的網路瀏覽、文字處理以及電腦學習的需要。

分A、B兩種型號，售價分別是A型25美元、B型35美元。樹莓派基金會從2012年2月29日開始接受B型的訂貨。

樹莓派基金會於2016年2月發布了樹莓派3，較前一代樹莓派2，樹莓派3的處理器升級為了64位元的博通BCM2837，新增了Wi-Fi無線網路及藍牙功能，而售價仍然是35美元。



**[](https://sites.google.com/site/rasberrypintust/home/shen-me-shi-shu-mei-pai/9(5)%20(1).jpg?attredirects=0)**圖 3-2Bundle - Pi3 and Red/White Official Case

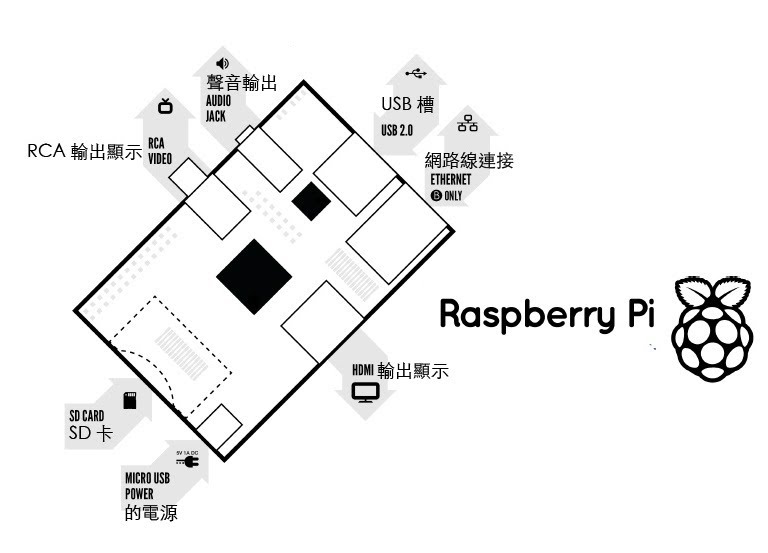
[](https://sites.google.com/site/rasberrypintust/home/shen-me-shi-shu-mei-pai/1-1-2%20(1).jpg?attredirects=0)

圖 3-3樹莓派示意圖 圖 3-4­­­樹莓派電路板實體圖

* **攀藤pm2.5/溫濕度二合一感測器 PMS5003T：**PMS5003T是一款可以同時監測空氣中顆粒物濃度及溫濕度的二合一感測器。其中顆粒物濃度的監測基於鐳射散射原理，可連續採集並計算單位體積內空氣中不同粒徑的懸浮顆粒物個數，即顆粒物濃度分佈，進而換算成為質量濃度。



圖 3-5攀藤pm2.5/溫濕度二合一感測器 PMS5003T

* **瑞昱 Ameba RTL8710 開發板：**Ameba RTL8710與Ameba RTL8195類似，都適合開發物聯網應用。它上面的介面有WiFi, GPIO, I2C, UART, SPI等等……，這些介面可以接Arduino常見的傳感器與模組，與Arduino不同的是這些資料可以直接經由內建的WiFi上傳至雲端，實現物聯網的應用。 RTL8710擁有與Arduino UNO近似相容的開發板，這個開發板與RTL8195開發板的大小一樣，它一樣由左邊的micro usb供電，而右邊是可拆卸的模塊，讓使用者在開發完成後，可以使用更小的空間來佈線與組裝。

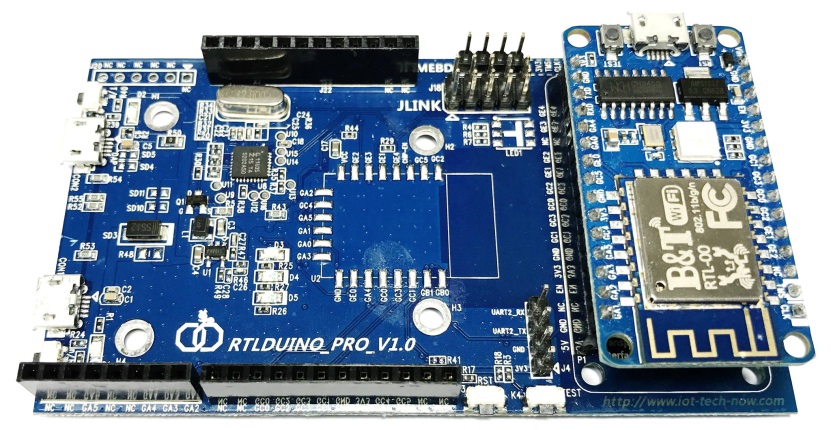


圖 3-6瑞昱 Ameba RTL8710 開發板

**軟體需求**

**Arch Linux：**Arch Linux是一款基於x86-64架構的Linux發行版。系統主要由自由和開源軟體組成，支援社群參與。系統設計以KISS原則（保持簡單和愚蠢）為總體指導原則，注重代碼正確、優雅和極簡主義，期待用戶能夠願意去理解系統的操作。Arch Linux採用滾動發行模式來取得系統更新和軟體的最新版本。系統安裝映像只簡單地包含系統主要元件。Arch Linux以社群Wiki的形式提供文件，稱為ArchWiki。該Wiki經常編有特定主題的最新資訊，受到了Linux社群的廣泛認可，內容也應用在Arch Linux以外的領域。



圖 3-7 ArchLinux

* **Apache：**是一種開放原始碼的網頁伺服器，可以在大多數電腦作業系統中運行，由於其跨平台與安全性，被廣泛使用，它快速、可靠且可通過簡單的API擴充。
* **MariaDB：**MariaDB資料庫管理系統是MySQL的一個分支，主要由開源社群在維護，採用GPL授權許可。開發這個分支的原因之一是：甲骨文公司收購了MySQL後，有將MySQL閉源的潛在風險，因此社群採用分支的方式來避開這個風險。MariaDB的目的是完全相容MySQL，包括API和命令列，使之能輕鬆成為MySQL的代替品。在儲存引擎方面，10.0.9版起使用XtraDB（名稱代號為Aria）來代替MySQL的InnoDB。
* **PHP：**是一種開放原始碼的通用電腦指令碼語言，尤其適用於網路開發並可嵌入HTML中使用。PHP的語法吸收了C語言、JAVA和Perl等電腦語言的特點，主要標的是允許網路開發人員快速編寫動態頁面。
* **Python：**Python，是一種廣泛使用的高階程式語言，屬於通用型程式語言，由吉多·范羅蘇姆 創造，第一版釋出於 1991 年。可以視之為一種改良（加入一些其他程式語言的優點，如物件導向）的 LISP。作為一種直譯語言，Python 的設計哲學強調代碼的可讀性和簡潔的語法（尤其是使用空格縮排劃分代碼塊，而非使用大括號或者關鍵詞）。相比於 C++ 或 Java，Python 讓開發者能夠用更少的代碼表達想法。不管是小型還是大型程式，該語言都試圖讓程式的結構清晰明了。

## 3.4相關開發技術

### 3.4.1 Angular (Angular 2)

Angular是一個前端應用框架，由Google維護。它的目標是透過MVC模式增強基於瀏覽器的應用，使跨平台開發和測試變得更加容易。Angular.js在版本2大改（最突出的改變的全面使用TypeScript）之後改名Angular，第N版以Angular N的形式命名。目前最新版本為Angular 6，不過習慣上仍就常被稱為Angular 2，以和Angular.js區別。

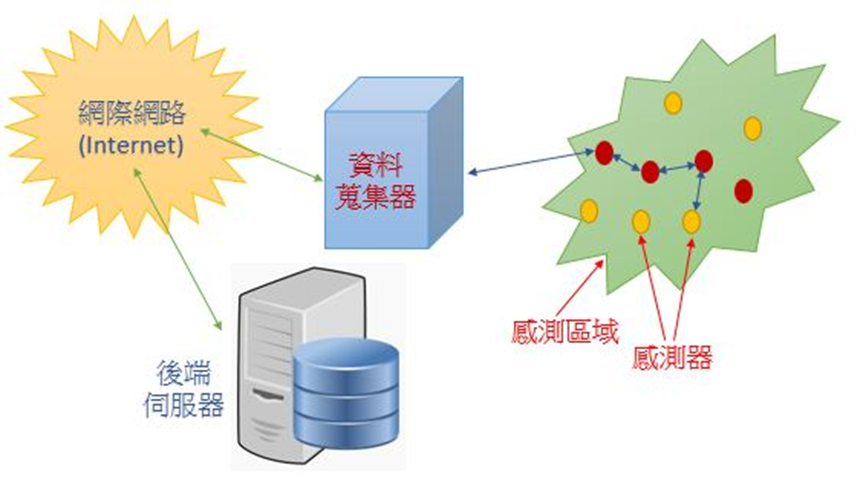
### 3.4.2 MySQL

MySQL 是一個快速、多執行緒（multithread）、多使用者且功能強大的關聯式資料庫管理系統（relational database management system, RDBMS），可以與 C、C++、Java、Perl、PHP 等語言輕易的連結使用。可以集中管理資料，並可以做出新增、刪除或查詢資料等等動作。而MySQL採用主從式架構，一個伺服器可以同時服務多個Client。

### 3.4.3無線感測網路（Wireless Sensor Networks, WSN）

無線感測網路（WSN）是由一到數個無線資料收集器（Wireless Data Collector）以及為數眾多的感測器所構成的網路系統，而元件之間的溝通則是採用無線通訊（Wireless Communicate）的方式。

在無線感測網路的架構下，感測器的設計，以省電、價格低廉、體積小、且具有感應環境裝置為目標，感測器本身就像是一台小型電腦，並配備了簡單的感測裝置、運算裝置、無線傳輸裝置，感測裝置可以針對環境中我們所感興趣的事物（如溫度、光源、氣體等）做偵測行為，並將所收集的資料先做簡單運算處理後，再透過無線傳輸裝置，將資料回傳給資料收集器，最後，我們就可以根據資料收集器所收集的資料，了解環境的狀態，並開發新穎有趣的應用了。

圖 3-8感測網路系統基本架構

感測網路系統的基本架構如Figure3-8所示，首先將大量的感測器（Sensor Nodes）任意地散佈於待感測的感測區域（Sensor Field） 來蒐集各種環境資料，偵測資料可為溫度、溼度、光度、壓力、二氧化碳濃度等，再藉由無線網路（紅外線、無線電波、光纖介質等），將蒐集的資訊透過無線資料蒐集器（Wireless Data Collector）傳回給管理者或使用者的手中。由於感測器在任意散佈的情形下，每個感測器並不知道自己與其他感測器的相對位置，因此感測網路必須使用自我組態（Self-Organization）的協定，將感測器之間自動組織出一個通訊網路，使得所有感測區域中的感測資料，皆能透過自我組態所建立的網路，將資訊送到無線資料蒐集器。無線資料蒐集器的功用則類似閘道器（Gateway），將感測網路的資料透過網際網路（Internet）、衛星，或是其他傳輸媒介，傳送到後端的伺服器（Server）：應用程式或管理者手中。

# 第四章 實作分析

## 4.1實驗設計

主要目的是希望能夠透過接收各種感測器的數值，透過網頁監控環境，並在數值超標時主動警示。

## 4.2系統功能說明

我們將環境監控系統網頁分成數個頁面，分別是「地圖首頁」、「總覽」、「歷史數據」、「測站比較」、「平均數據」、「帳號註冊\登入\各種設定」。

線上網站為: https://rpiarch.ddns.net/ (<https://140.128.102.200>)。

Github程式碼: https://github.com/jim60105/PMS5003TAirQuality

「首頁」主要分成兩個部分，「Google地圖」和「測站資訊窗」。在「Google地圖」中主要呈現當下所有測站的位置。若未設定或未登入，則會顯示最靠近使用者的測站；若使用者不同意分享位置資訊，則會顯示東海大學附近的測站。「測站資訊窗」呈現選定到的觀測點之即時資訊。

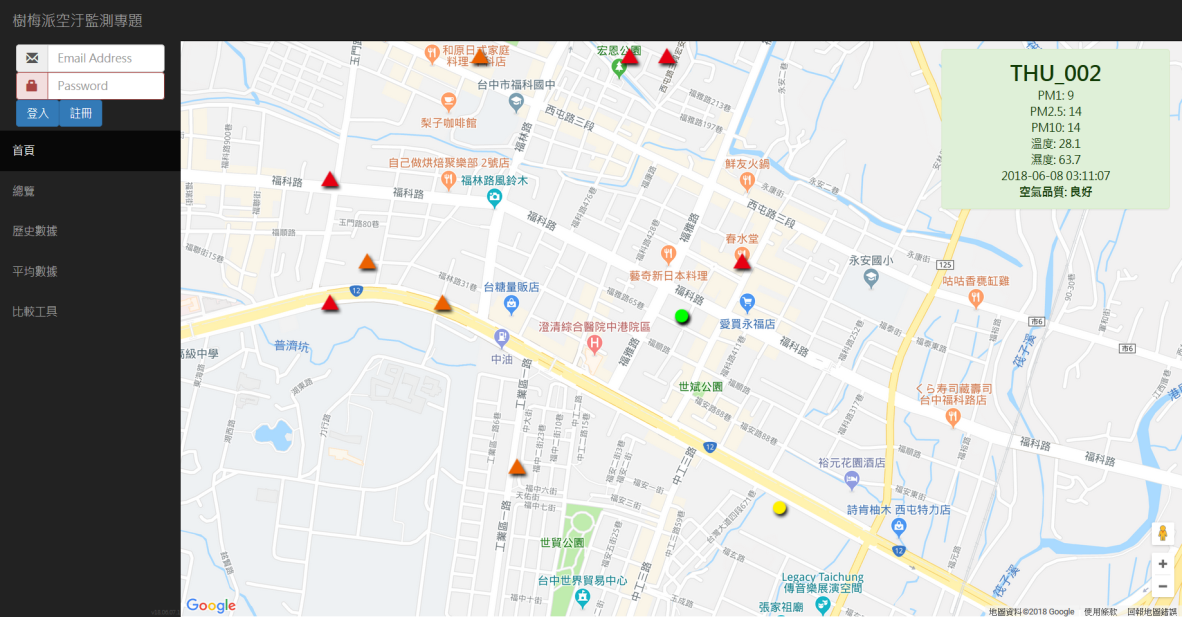


圖 4-1 Google地圖測站資訊



圖 4-2手機版響應式設計

「總覽」主要分成上下兩個部分，「即時環境資訊」和「詳細數據」。在「即時環境資訊」中主要呈現當下測站環境的各種資訊。



圖 4-3總覽資訊頁面

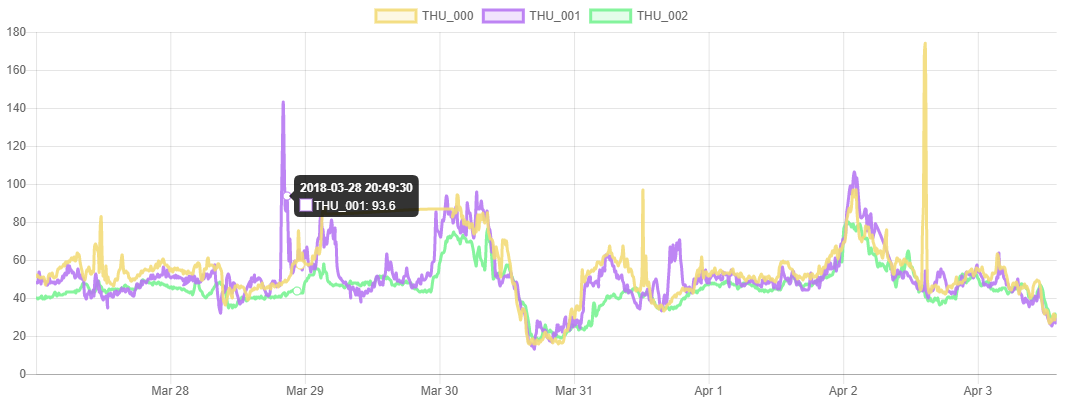


圖 4-4當下測站資訊

「歷史數據」則主要分成三個部分，「測站折線圖表」、「危害程度數據」和「數據總覽」；「測站折線圖表」包括項目有五種，「PM1」、「PM2.5」、「PM10」、「溫度」和「濕度」。

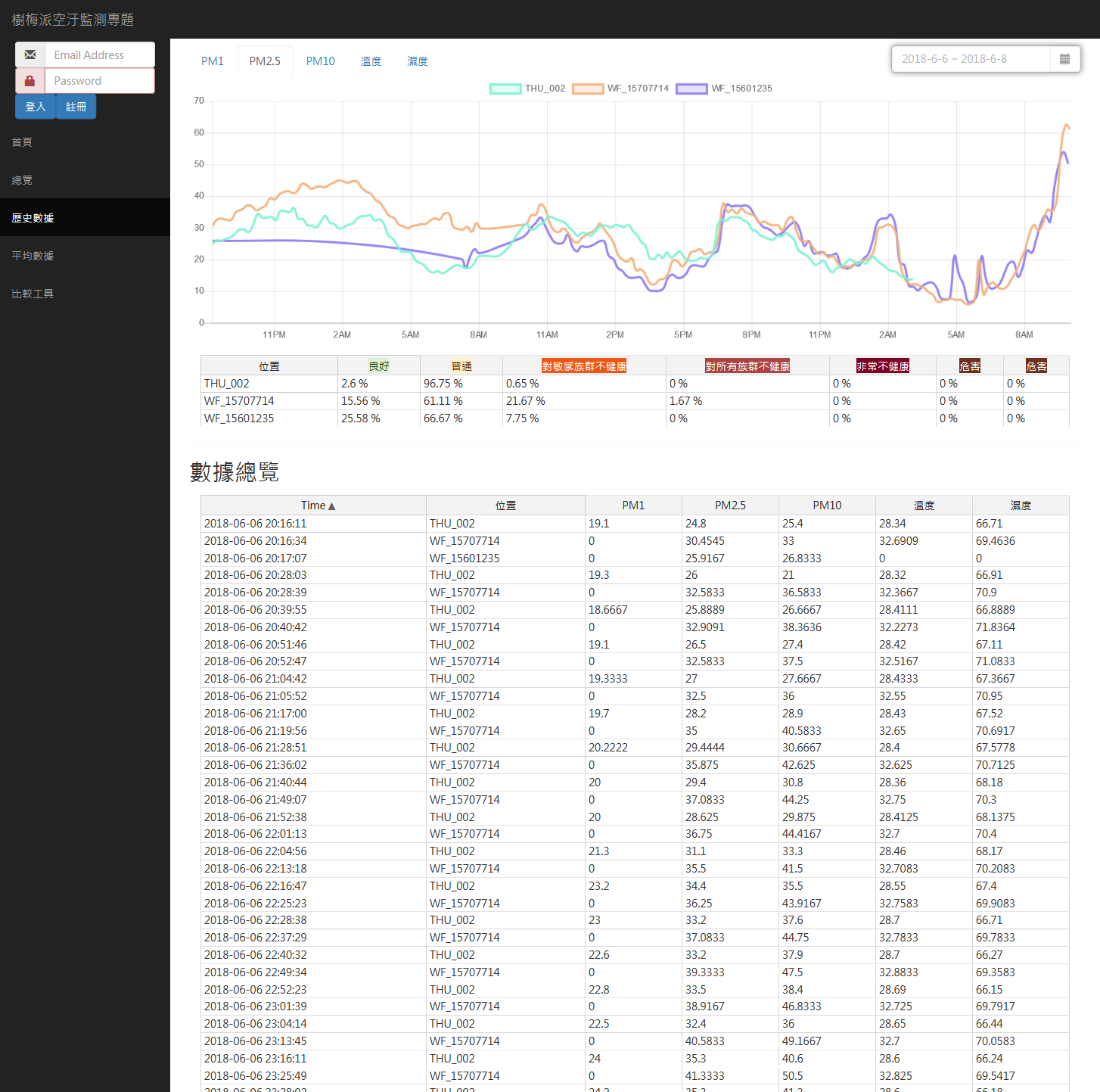


圖 4-5歷史數據

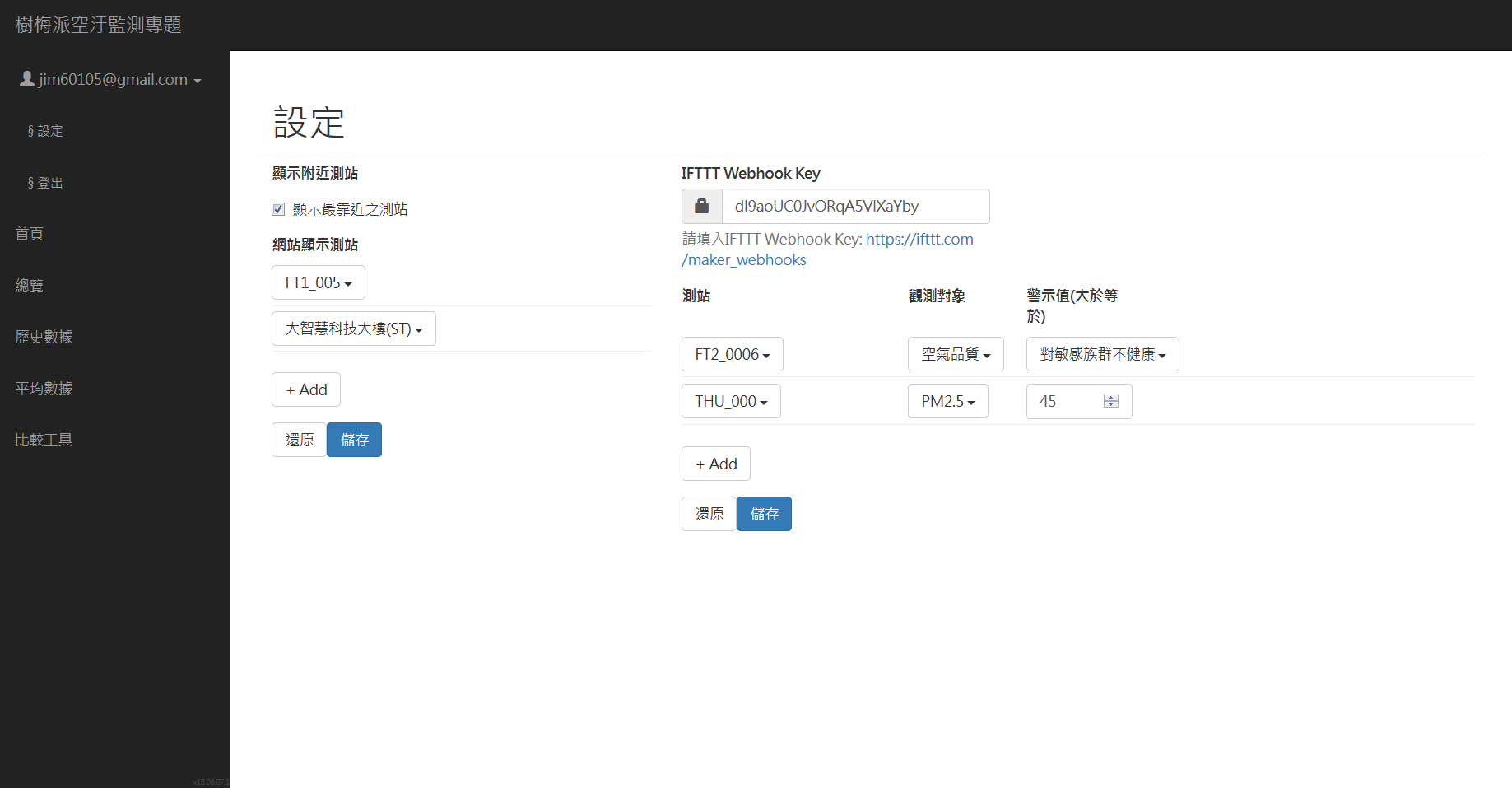


圖 4-6 設定頁面

在左上角登入後，則可以進到「設定」頁面，在這裡可以變更欲觀測的測站。經過設定後只要保持登入，則所有頁面顯示的都會是指定測站的資料。設定頁之右側即是IFTTT相關設定。將IFTTT Webhooks Key填入欄位保存，就可以設定超標通知。

「平均數據」頁面顯示所選測站時間範圍內之平均數據，可以一目瞭然得知測站狀況。

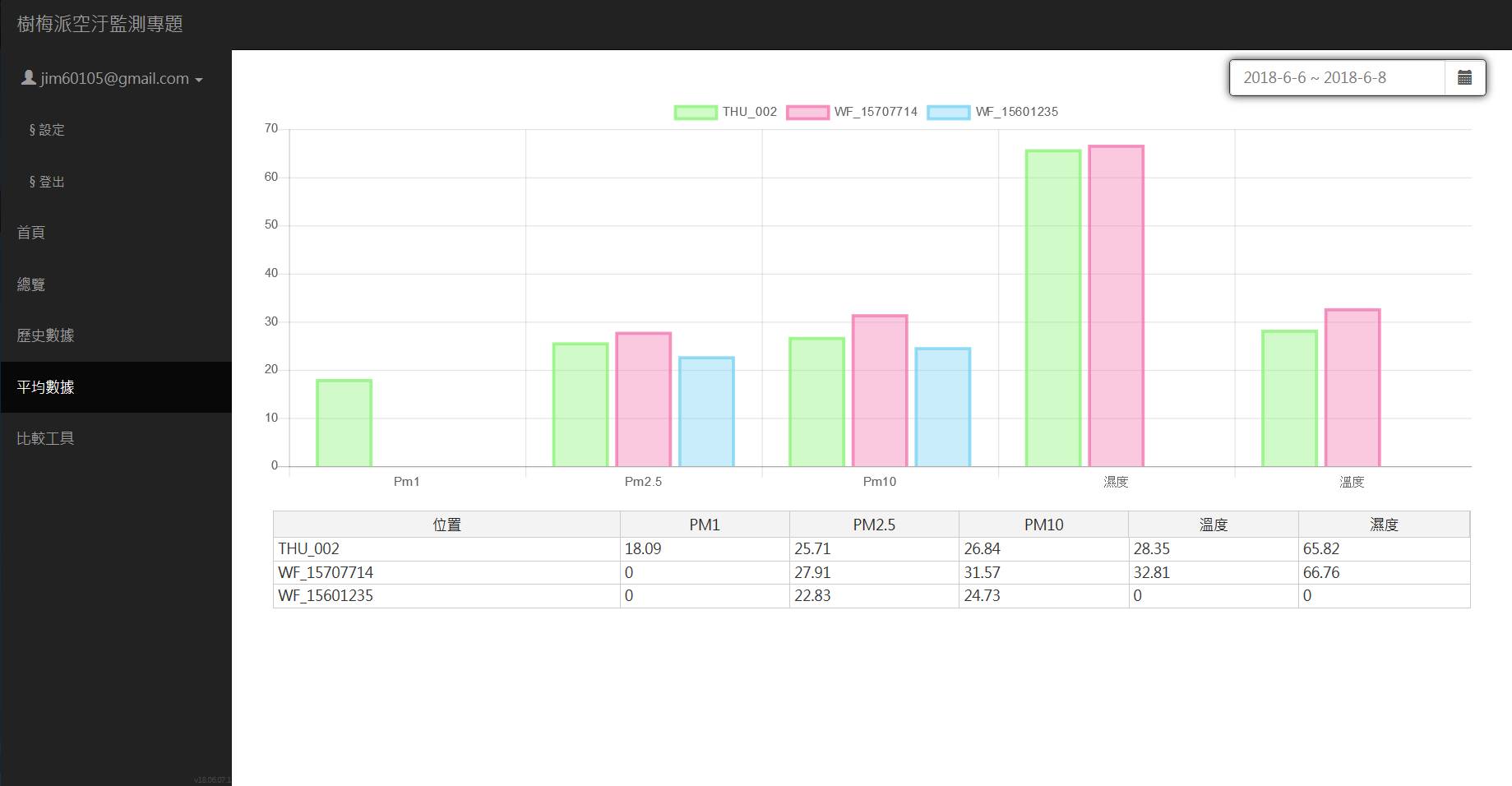


圖 4-7 平均數據

「比較工具」的部分，是以同樣時間長度為前提，進行數據繪圖比較。選擇測站的開始日期，並選擇時間長度，即可開始比較。也可以選擇同一測站的不同時間段作使用，觀察不同日期的差異情形。

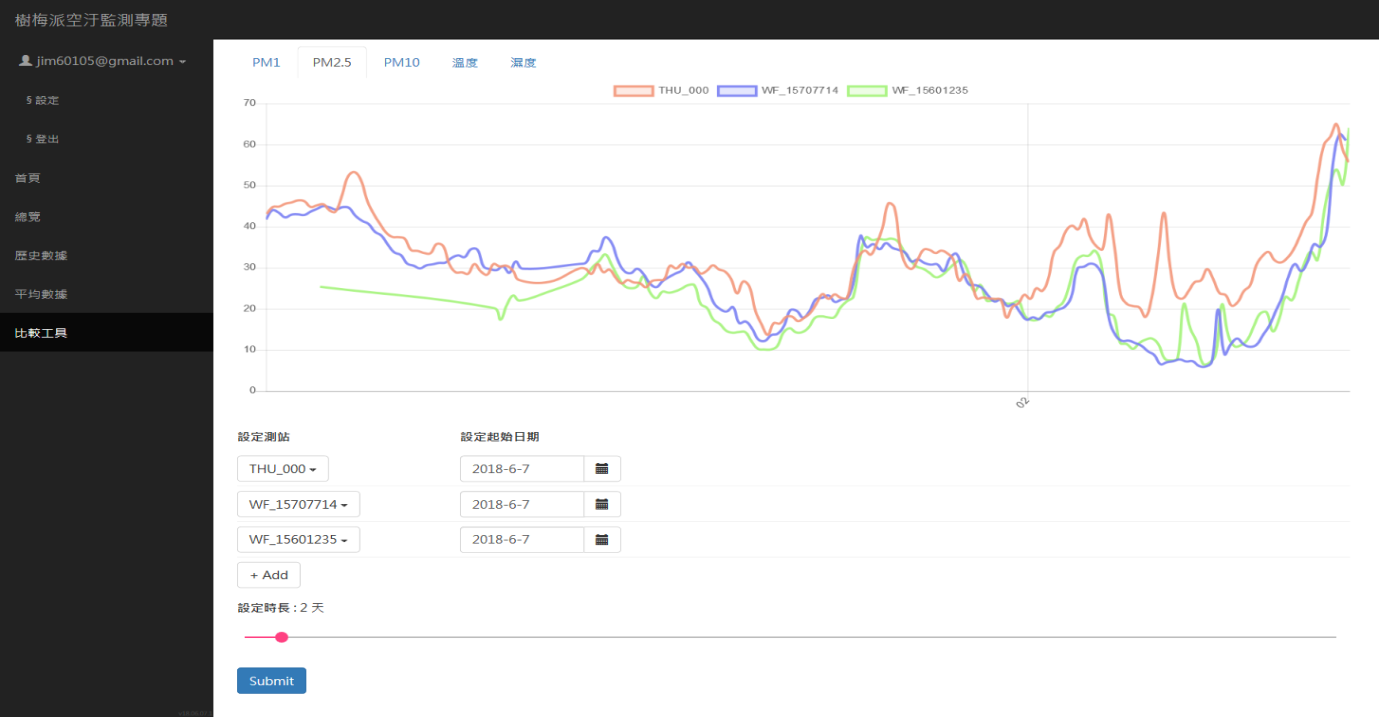


圖 4-8 比較工具

## 4.3資料量問題

實驗初期，我們將所有LASS Subscribe到的資料都儲存起來，每日約新增二十七萬筆資料。但是因LASS資料過於龐大，且資料上傳頻率過於密集，很快得便發現樹莓派出現效能瓶頸。經過商議，我們試著加入了一個佇列：「將各個感測器每十分鐘收到的數值做平均，再存入資料庫」。運行後目前每天約為三萬多筆資料，資料量大幅下降來到適當的規模，又不會過於降低資料價值。

而後，約於一個半月後遇到了SD卡儲存空間塞滿，此間約產生了27G的資料庫資料。原本欲實行之解決方案為增加儲存空間容量──樹莓派掛載外接式硬碟。與指導老師討論後，考量到專題需求時間不長，加上經費並不充足，沒有必要再花錢買硬碟，所以改行舊資料刪除的方案。此次評審只保存從2018/6/7開始的資料，更早的部分將移除。

## 4.4 IFTTT串接設計

一開始撰寫這部分時，是預計寫IFTTT Platform。考察後發現，Platform之免費方案只能讓包含開發者在內的三人使用程式，要開放多人使用則必須付費。所以改為採用免費的IFTTT Webhooks。由於IFTTT Webhooks是為Maker打造的功能，設定上較為繁瑣，使用體驗稍差，所以為次要方案。為彌補設定複雜的小缺陷，可撰寫說明文件以指示使用者。然而此次專案著重在技術上的實作，加上時間上不充裕，故最後並無撰寫此說明文件。

謹記以下幾點設定流程:

1. IFTTT設定，來源選擇Webhooks
2. Event Name必須設定為 “AirAlert”
3. 三個參數值內容:

Value1: 測站名稱

Value2: 觀測項目(PM1、PM2.5、PM10…等等)

Value3: 觀測數值

1. 設定完成後，將Webhooks Key填入敝站設定頁面

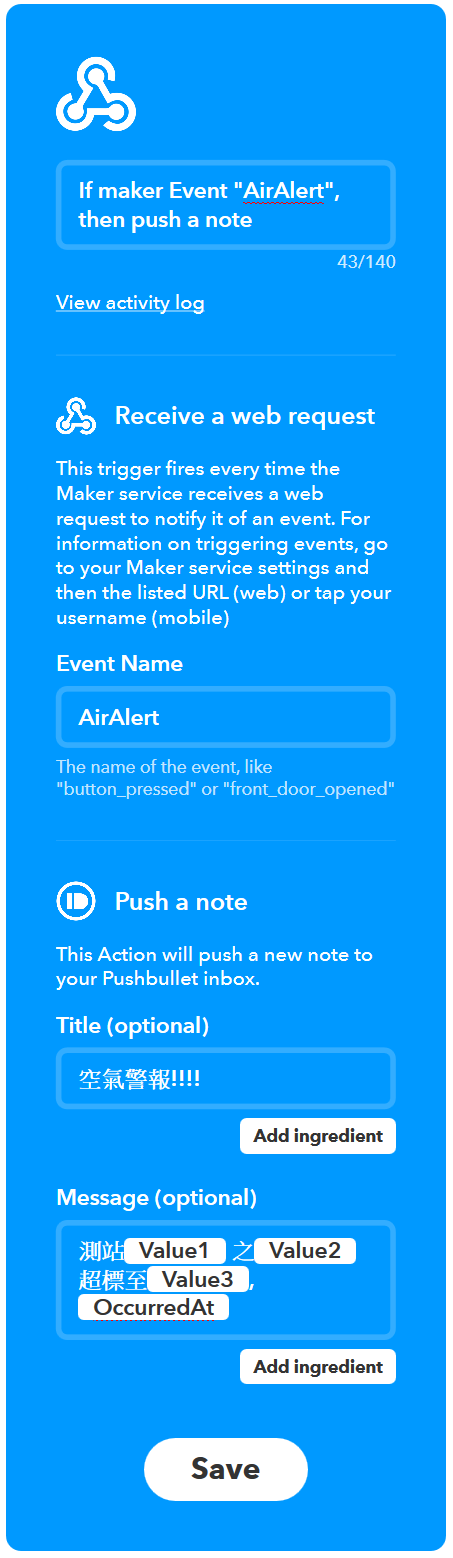
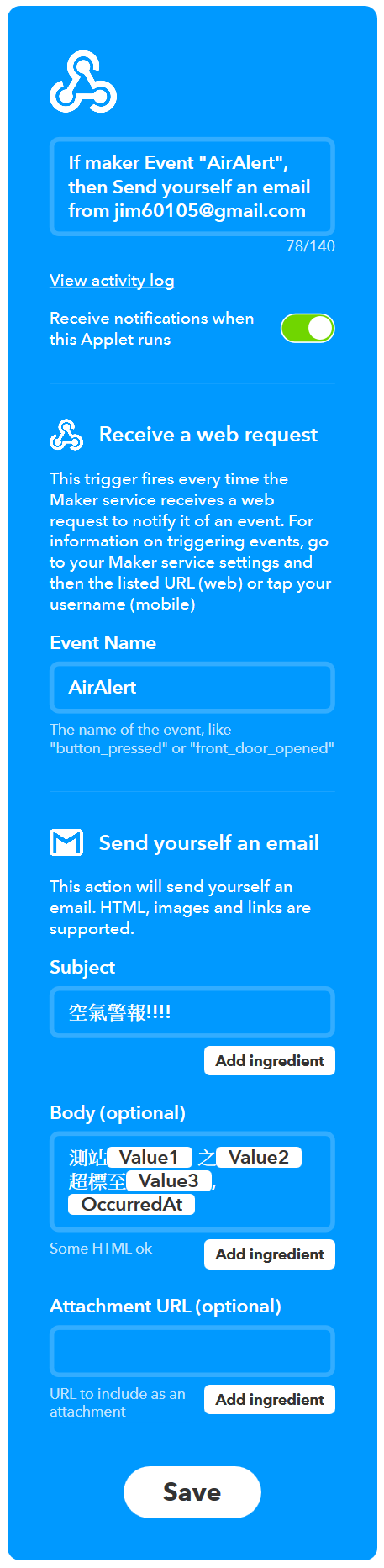
 

圖 4-9 IFTTT設定

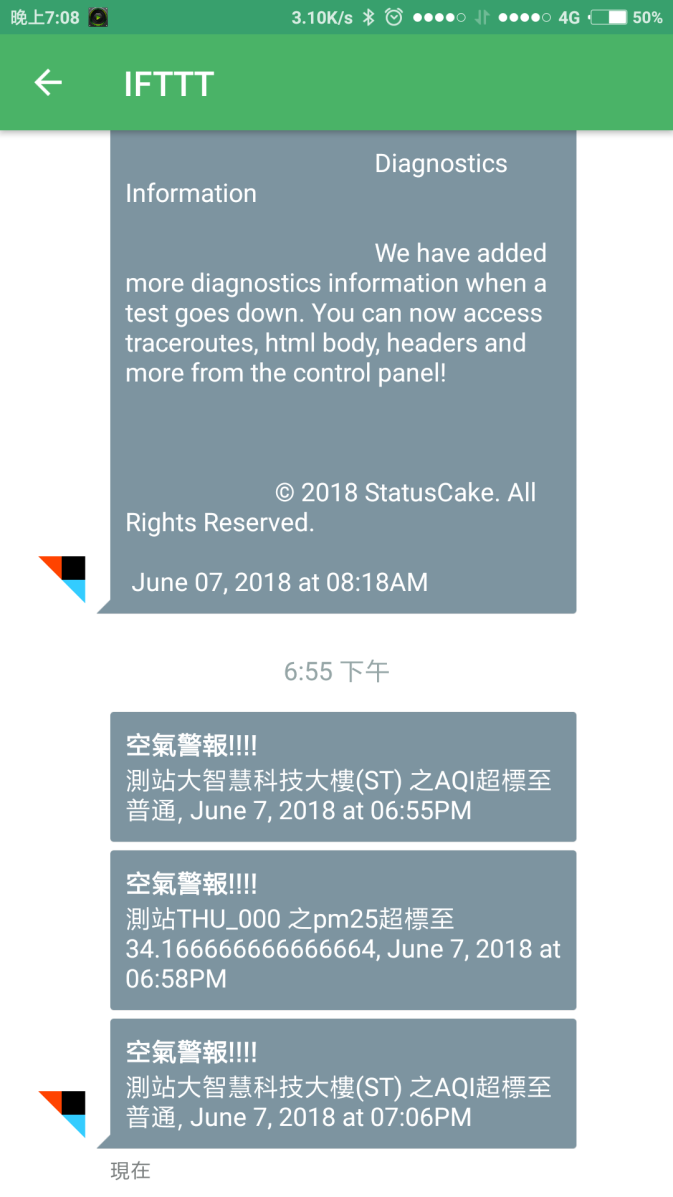


圖 4-10 IFTTT串接Pushbullet到手機

## 4.5 Libsodium加密庫

曾有人說過: 「除非你是密碼學專家，否則不要用自己寫的加密功能。」因此，此次專題在使用者密碼儲存和校驗特地用上了Libsodium加密庫。網站是透過PHP與資料庫溝通，而PHP更是第一個將Libsodium加密庫集成進內核的程式語言。

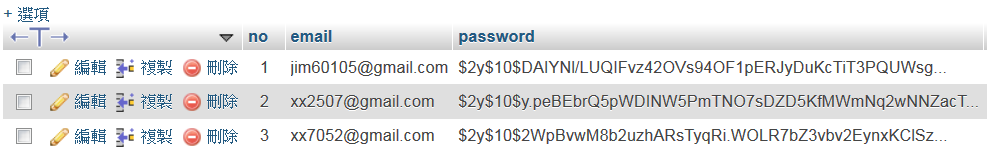


圖 4-11 資料庫密碼加密

# 第五章 結論

## 5.1結論

1.研究限制:我們使用的空氣感測器為小型零件，所以無法像氣象局的測站測得數據那麼準確，但是我們卻可以透過價格不是很昂貴的自制監控儀器來即時看到我們生活周遭的空氣變化。

2.研究成果:透過我們架設測站點可以了解我們生活的地方當下的空氣變化。

3.研究貢獻:將我們架設測站的資訊上傳到LASS環境感測網路系統，可以使對這個議題有興趣的每位自造者可以有更準確的空氣變化。

4.後續研究方向:希望可以找到更好的感測器來更換目前的感測器，也希望可以透過我們的網路平台可以讓政府及民眾來更多關注這個議題，使我們生活環境可以透過更多人關注而來改善。

## 5.2心得

一開始我們以為這份專題很容易，只要寫好網站，架好伺服器，擺好感測器終端，就大功告成。但是後來經過檢討，決定在中途大幅修改專題內容，引入LASS等其他數據源，並調整專題核心使之更加完整！

《小王子》這本書中提到：「所有的價值形成都來自於你的愛與你的付出，如果你真的擁有一個屬於你自己的夢想，你就要努力付出心力與時間，並堅持且全力以赴去完成。」於是燃起了我的東海魂，重新檢討了內容。而後加入的LASS環境感測網路系統，我們並不只是將別人的網站連結過來，而是為了取得更多測站點的資訊，並且將公開資料加入我們的數據源，透過我們的網路平台將所有的站點的資訊整合呈現，以滿足更多的使用需求。

在這次的專題過程中，我們學習到了如何包裝整合組員的想法，如何擬定計畫逐步完成，也發現了工作分配以及時程安排的重要性。相信未來我們一定能夠在職場上發揮這次的寶貴經驗。

# 附錄

網頁部分採用AngularJS 2撰寫，其原始碼結構型態與傳統網頁結構相異。

以下列出網頁部分目錄架構。

..\PMS5003TAIRQUALITY\PMS5003TAIRQUALITY

| .editorconfig

| angular.json

| karma.conf.js

| package-lock.json

| package.json

| protractor.conf.js

| README.md

| systemjs.config.js

| tree.txt

| tsconfig.json

| tslint.json

|

+---dist

| | .htaccess

| | 3rdpartylicenses.txt

| | favicon.ico

| | glyphicons-halflings-regular.448c34a56d699c29117a.woff2

| | glyphicons-halflings-regular.89889688147bd7575d63.svg

| | glyphicons-halflings-regular.e18bbf611f2a2e43afc0.ttf

| | glyphicons-halflings-regular.f4769f9bdb7466be6508.eot

| | glyphicons-halflings-regular.fa2772327f55d8198301.woff

| | index.html

| | main.a325e5f021bb81a9135f.js

| | polyfills.23d02fdfcdd4411c0529.js

| | runtime.a66f828dca56eeb90e02.js

| | scripts.e0e1e3e72d8673ec04f4.js

| | styles.6669f03dfcf0c99a089e.css

| |

| \---assets

| | animation.css

| | generateNewKey.php

| | info.php

| | mock-data.ts

| | mock-device.ts

| | mock-singleData.ts

| | rcolor.js

| |

| +---php

| | .htaccess

| | addData.php

| | encrypt.php

| | getDBByTimeById.php

| | getDBByTimeSingleData.php

| | getDBDevice.php

| | getLASSByTimeById.php

| | getLASSByTimeByIdSingleData.php

| | getLASSDevices.php

| | getUserDevices.php

| | login.php

| | proxy.php

| | registration.php

| | setMySQL.php

| | setUserDevices.php

| | setUserSettings.php

| |

| +---pic

| | AQI0.png

| | AQI1.png

| | AQI2.png

| | AQI3.png

| | AQI4.png

| | AQI5.png

| | AQI6.png

| | loading.png

| | lungs.png

| | shape.psd

| |

| \---template

| | sb-admin.css

| | sb-admin.js

| |

| \---startbootstrap-sb-admin-3.3.7

| | blank-page.html

| | bootstrap-elements.html

| | bootstrap-grid.html

| | charts.html

| | forms.html

| | index-rtl.html

| | index.html

| | LICENSE

| | README.md

| | tables.html

| |

| +---css

| | | bootstrap-rtl.css

| | | bootstrap-rtl.min.css

| | | bootstrap.css

| | | bootstrap.min.css

| | | sb-admin-rtl.css

| | | sb-admin.css

| | |

| | \---plugins

| | morris.css

| |

| +---font-awesome

| | +---css

| | | font-awesome.css

| | | font-awesome.min.css

| | |

| | +---fonts

| | | fontawesome-webfont.eot

| | | fontawesome-webfont.svg

| | | fontawesome-webfont.ttf

| | | fontawesome-webfont.woff

| | | FontAwesome.otf

| | |

| | +---less

| | | bordered-pulled.less

| | | core.less

| | | fixed-width.less

| | | font-awesome.less

| | | icons.less

| | | larger.less

| | | list.less

| | | mixins.less

| | | path.less

| | | rotated-flipped.less

| | | spinning.less

| | | stacked.less

| | | variables.less

| | |

| | \---scss

| | font-awesome.scss

| | \_bordered-pulled.scss

| | \_core.scss

| | \_fixed-width.scss

| | \_icons.scss

| | \_larger.scss

| | \_list.scss

| | \_mixins.scss

| | \_path.scss

| | \_rotated-flipped.scss

| | \_spinning.scss

| | \_stacked.scss

| | \_variables.scss

| |

| +---fonts

| | glyphicons-halflings-regular.eot

| | glyphicons-halflings-regular.svg

| | glyphicons-halflings-regular.ttf

| | glyphicons-halflings-regular.woff

| | glyphicons-halflings-regular.woff2

| |

| \---js

| | bootstrap.js

| | bootstrap.min.js

| | jquery.js

| |

| \---plugins

| +---flot

| | excanvas.min.js

| | flot-data.js

| | jquery.flot.js

| | jquery.flot.pie.js

| | jquery.flot.resize.js

| | jquery.flot.tooltip.min.js

| |

| \---morris

| morris-data.js

| morris.js

| morris.min.js

| raphael.min.js

|

+---e2e

| app.e2e-spec.ts

| app.po.ts

| tsconfig.e2e.json

|

\---src

| .htaccess

| favicon.ico

| index.html

| main.ts

| polyfills.ts

| styles.css

| test.ts

| tsconfig.app.json

| tsconfig.spec.json

| typings.d.ts

|

+---app

| | app-routing.module.ts

| | app.component.html

| | app.component.ts

| | app.module.ts

| | calc-AQI.component.ts

| | daterangepicker.component.ts

| | loading.component.ts

| | portraitWarning.component.ts

| | safe-html.component.ts

| |

| +---data-table

| | data-table.component.css

| | data-table.component.html

| | data-table.component.ts

| |

| +---login

| | login.component.css

| | login.component.html

| | login.component.ts

| |

| +---pages

| | +---average-page

| | | average-page.component.css

| | | average-page.component.html

| | | average-page.component.ts

| | |

| | +---compare-page

| | | compare-page.component.css

| | | compare-page.component.html

| | | compare-page.component.ts

| | |

| | +---history-page

| | | history-page.component.css

| | | history-page.component.html

| | | history-page.component.ts

| | |

| | +---main-page

| | | main-page.component.css

| | | main-page.component.html

| | | main-page.component.ts

| | |

| | +---map-page

| | | map-page.component.css

| | | map-page.component.html

| | | map-page.component.ts

| | |

| | +---setting-page

| | | setting-page.component.css

| | | setting-page.component.html

| | | setting-page.component.ts

| | |

| | \---signup-page

| | signup-page.component.css

| | signup-page.component.html

| | signup-page.component.ts

| |

| \---services

| get-data.service.ts

| get-device.service.ts

| get-http-edimax.service.ts

| get-http-epa.service.ts

| get-http-template.service.ts

| get-lassdata.service.ts

| get-lassdevice.service.ts

| get-single-data.service.ts

| get-single-lassdata.service.ts

| get-thingspeak-data.service.ts

| get-user-device.service.ts

| login.service.ts

| set-user-device.service.ts

| set-user-settings.service.ts

|

+---assets

| | .gitkeep

| | animation.css

| | generateNewKey.php

| | info.php

| | mock-data.ts

| | mock-device.ts

| | mock-singleData.ts

| | rcolor.js

| |

| +---php

| | .htaccess

| | addData.php

| | encrypt.php

| | getDBByTimeById.php

| | getDBByTimeSingleData.php

| | getDBDevice.php

| | getLASSByTimeById.php

| | getLASSByTimeByIdSingleData.php

| | getLASSDevices.php

| | getUserDevices.php

| | login.php

| | proxy.php

| | registration.php

| | setMySQL.php

| | setUserDevices.php

| | setUserSettings.php

| |

| +---pic

| | AQI0.png

| | AQI1.png

| | AQI2.png

| | AQI3.png

| | AQI4.png

| | AQI5.png

| | AQI6.png

| | loading.png

| | lungs.png

| | shape.psd

| |

| \---template

| | sb-admin.css

| | sb-admin.js

| |

| \---startbootstrap-sb-admin-3.3.7

| | blank-page.html

| | bootstrap-elements.html

| | bootstrap-grid.html

| | charts.html

| | forms.html

| | index-rtl.html

| | index.html

| | LICENSE

| | README.md

| | tables.html

| |

| +---css

| | | bootstrap-rtl.css

| | | bootstrap-rtl.min.css

| | | bootstrap.css

| | | bootstrap.min.css

| | | sb-admin-rtl.css

| | | sb-admin.css

| | |

| | \---plugins

| | morris.css

| |

| +---font-awesome

| | +---css

| | | font-awesome.css

| | | font-awesome.min.css

| | |

| | +---fonts

| | | fontawesome-webfont.eot

| | | fontawesome-webfont.svg

| | | fontawesome-webfont.ttf

| | | fontawesome-webfont.woff

| | | FontAwesome.otf

| | |

| | +---less

| | | bordered-pulled.less

| | | core.less

| | | fixed-width.less

| | | font-awesome.less

| | | icons.less

| | | larger.less

| | | list.less

| | | mixins.less

| | | path.less

| | | rotated-flipped.less

| | | spinning.less

| | | stacked.less

| | | variables.less

| | |

| | \---scss

| | font-awesome.scss

| | \_bordered-pulled.scss

| | \_core.scss

| | \_fixed-width.scss

| | \_icons.scss

| | \_larger.scss

| | \_list.scss

| | \_mixins.scss

| | \_path.scss

| | \_rotated-flipped.scss

| | \_spinning.scss

| | \_stacked.scss

| | \_variables.scss

| |

| +---fonts

| | glyphicons-halflings-regular.eot

| | glyphicons-halflings-regular.svg

| | glyphicons-halflings-regular.ttf

| | glyphicons-halflings-regular.woff

| | glyphicons-halflings-regular.woff2

| |

| \---js

| | bootstrap.js

| | bootstrap.min.js

| | jquery.js

| |

| \---plugins

| +---flot

| | excanvas.min.js

| | flot-data.js

| | jquery.flot.js

| | jquery.flot.pie.js

| | jquery.flot.resize.js

| | jquery.flot.tooltip.min.js

| |

| \---morris

| morris-data.js

| morris.js

| morris.min.js

| raphael.min.js

|

\---environments

environment.prod.ts

environment.ts

部分程式碼：lass2mysql.py (處理將LASS公開數據倒入mysql資料庫)

#!/usr/bin/env python

# Author: jim60105@gmail.com

# Version: v18.05.31.0

import paho.mqtt.client as mqtt

import re

import mysql.connector

import ast

import traceback

import datetime

import requests

import json as json

MQTT\_SERVER = "gpssensor.ddns.net"

MQTT\_PORT = 1883

MQTT\_ALIVE = 60

MQTT\_TOPIC = "LASS/Test/#"

f = open("dbpassword.pas","r")

dbPas = str(f.readline())[0:-1]

f.close()

conn = mysql.connector.connect(

user='air',

password=dbPas,

host='127.0.0.1',

database='air')

cur = conn.cursor(dictionary=True)

# The callback for when the client receives a CONNACK response from the server.

def on\_connect(client, userdata, flags, rc):

print("MQTT Connected with result code "+str(rc) , flush=True)

# Subscribing in on\_connect() means that if we lose the connection and

# reconnect then subscriptions will be renewed.

client.subscribe(MQTT\_TOPIC, 0)

# The callback for when a PUBLISH message is received from the server.

def on\_message(client, userdata, msg):

tempDict = {}

# print(str(msg.payload))

input = re.split('\'',str(msg.payload))

items = re.split('\|',input[1])

for item in items:

if item == '':

continue

pairs = re.split('=',item)

if (len(pairs)==1):

continue

tempDict[pairs[0]] = pairs[1]

# if (pairs[0] == "device\_id"):

# value\_device = pairs[1]

#if (value\_device == "THU\_000" or value\_device == "THU\_001" or value\_device == "THU\_002"):

#print ("Got the data from %s: %s" % (value\_device, msg.payload) , flush=True)

if 'device\_id' not in tempDict:

return

else:

tempDict['device\_id'] = tempDict['device\_id']

if 'app' not in tempDict or tempDict['app']=='':

tempDict['app'] = 'NULL'

else:

tempDict['app'] = tempDict['app']

if 'device' not in tempDict or tempDict['device']=='':

tempDict['device'] = 'NULL'

else:

tempDict['device'] = tempDict['device']

try:

if 'date' not in tempDict or 'time' not in tempDict:

tempDict['time'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

elif datetime.datetime.strptime(tempDict['date'],'%Y-%m-%d')<datetime.datetime(1970, 1, 1):

tempDict['time'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

else:

tempDict['time'] = datetime.datetime.strptime(tempDict['date'] + ' ' +tempDict['time'],'%Y-%m-%d %H:%M:%S')

except:

tempDict['time'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

if 's\_t0' in tempDict:

tempDict['temp'] = tempDict['s\_t0']

elif 's\_t1' in tempDict:

tempDict['temp'] = tempDict['s\_t1']

elif 's\_t2' in tempDict:

tempDict['temp'] = tempDict['s\_t2']

elif 's\_t3' in tempDict:

tempDict['temp'] = tempDict['s\_t3']

elif 's\_t4' in tempDict:

tempDict['temp'] = tempDict['s\_t4']

elif 's\_t5' in tempDict:

tempDict['temp'] = tempDict['s\_t5']

else:

tempDict['temp'] = 'NULL'

if 's\_h0' in tempDict:

tempDict['humid'] = tempDict['s\_h0']

elif 's\_h1' in tempDict:

tempDict['humid'] = tempDict['s\_h1']

elif 's\_h2' in tempDict:

tempDict['humid'] = tempDict['s\_h2']

elif 's\_h3' in tempDict:

tempDict['humid'] = tempDict['s\_h3']

elif 's\_h4' in tempDict:

tempDict['humid'] = tempDict['s\_h4']

elif 's\_h5' in tempDict:

tempDict['humid'] = tempDict['s\_h5']

else:

tempDict['humid'] = 'NULL'

if 'tick' not in tempDict:

tempDict['tick'] = 'NULL'

if 's\_d2' not in tempDict:

tempDict['s\_d2'] = 'NULL'

if 's\_d1' not in tempDict:

tempDict['s\_d1'] = 'NULL'

if 's\_d0' not in tempDict:

tempDict['s\_d0'] = 'NULL'

if 's\_g8' not in tempDict:

tempDict['s\_g8'] = 'NULL'

if 'gps\_lat' not in tempDict or tempDict['gps\_lat']=='':

tempDict['gps\_lat'] = 'NULL'

if 'gps\_lon' not in tempDict or tempDict['gps\_lon']=='':

tempDict['gps\_lon'] = 'NULL'

if 'gps\_alt' not in tempDict or tempDict['gps\_alt']=='':

tempDict['gps\_alt'] = 'NULL'

try:

query = ('SELECT \* FROM `lassdataqueue` WHERE `device\_id` = %(device\_id)s AND `time` = %(time)s')

cur.execute(query,tempDict)

rows = cur.fetchall()

conn.commit()

if len(rows) == 0:

# 插入佇列

query = ('INSERT INTO `lassdataqueue` '

'(`no`, `device\_id`, `time`, `pm1`, `pm10`, `pm25`, `temp`, `humid`, `co2`) '

'VALUES (NULL, %(device\_id)s, %(time)s, %(s\_d2)s, %(s\_d1)s, %(s\_d0)s, %(temp)s, %(humid)s, %(s\_g8)s)')

cur.execute(query,tempDict)

conn.commit()

def calcAQI(pm25,pm10):

temp25 = 0;

temp10 = 0;

# 顏色及數值依照政府標準

# https://taqm.epa.gov.tw/taqm/tw/b0201.aspx

if pm25<=15.4: # green

temp25 = 1

elif pm25<=35.4: # yellow

temp25 = 2

elif pm25<=54.4: # orange

temp25 = 3

elif pm25<=150.4: # red

temp25 = 4

elif pm25<=250.4: # purple

temp25 = 5

else: # maroon

temp25 = 6

if pm10<=54:

temp10 = 1

elif pm10<=125:

temp10 = 2

elif pm10<=254:

temp10 = 3

elif pm10<=354:

temp10 = 4

elif pm10<=424:

temp10 = 5

else:

temp10 = 6

return max(temp10, temp25)

# Flush以後檢查iftttDeviceList，並做通知

def sendIFTTT(iftttDevice,avgData):

if iftttDevice['air\_type'] == 'AQI':

select = {

0:'(離線)',

1:'良好',

2:'普通',

3:'對敏感族群不健康',

4:'對所有族群不健康',

5:'非常不健康',

6:'危害'

}

avgData['AQI'] = select[avgData['AQI']]

# print (avgData['AQI'], sep='\n', flush=True)

try:

# print ('URL: ' + 'https://maker.ifttt.com/trigger/AirAlert/with/key/' + iftttDevice['iftttKey'], sep='\n', flush=True)

r = requests.post('https://maker.ifttt.com/trigger/AirAlert/with/key/' + iftttDevice['iftttKey'], json = {'value1':avgData['device\_id'],'value2':iftttDevice['air\_type'],'value3':avgData[iftttDevice['air\_type']]})

# print ('IFTTT sended.', sep='\n', flush=True)

except requests.exceptions.RequestException as e: # This is the correct syntax

print ('Post IFTTT error: '+e, sep='\n', flush=True)

def checkIFTTTDevice(avgData):

query = ('SELECT `a`.\*,`b`.`iftttKey` FROM `useriftttdevice` as `a`,`user` as `b` '

'WHERE `type` = "LASS" '

'AND `device\_id` = %(device\_id)s '

'AND `a`.`user\_no` = `b`.`no`')

cur.execute(query,avgData)

rows = cur.fetchall()

conn.commit()

for row in rows:

if row['air\_type'] == 'AQI':

# print ('calcAQI'+ str(calcAQI(avgData['pm25'],avgData['pm10'])), sep='\n', flush=True)

avgData['AQI'] = calcAQI(avgData['pm25'],avgData['pm10'])

if row['monitor\_value'] <= avgData[row['air\_type']]:

sendIFTTT(row,avgData)

# Flush: 每次檢查時間排序頭尾的差距，如果大於10分鐘，則丟一筆平均進資料庫，然後把queue裡的砍掉

def checkFlush():

query = ('SELECT TIMESTAMPDIFF(MINUTE,'

'(SELECT time FROM lassdataqueue WHERE device\_id = %(device\_id)s ORDER BY time ASC LIMIT 1),'

'(SELECT time FROM lassdataqueue WHERE device\_id = %(device\_id)s ORDER BY time DESC LIMIT 1)'

') AS timediff;')

cur.execute(query,tempDict)

row = cur.fetchone()

conn.commit()

if row['timediff'] is not None:

tempDict['timediff'] = row['timediff'] / 2

return row['timediff']

else:

return 0

def doFlush():

query = ('SELECT time FROM lassdataqueue WHERE device\_id = %(device\_id)s ORDER BY time ASC LIMIT 1;')

cur.execute(query,tempDict)

row = cur.fetchone()

conn.commit()

tempDict['startTime'] = row['time']

query = ('SELECT AVG(pm1) AS pm1, AVG(pm10) AS pm10, AVG(pm25) AS pm25, AVG(temp) AS temp, AVG(humid) AS humid, AVG(co2) AS co2 FROM lassdataqueue WHERE device\_id = %(device\_id)s;')

cur.execute(query,tempDict)

rows = cur.fetchall()

conn.commit()

row = rows[0]

for k,v in row.items():

if v == None:

row[k] = 'Null'

row['device\_id'] = tempDict['device\_id']

row['timediff'] = tempDict['timediff']

row['startTime'] = tempDict['startTime']

query = ('INSERT INTO lassdata (`no`, `device\_id`, `time`, `pm1`, `pm10`, `pm25`, `temp`, `humid`, `co2`) '

'VALUES (NULL, %(device\_id)s, (SELECT TIMESTAMPADD(MINUTE, %(timediff)s, %(startTime)s)), %(pm1)s, %(pm10)s, %(pm25)s, %(temp)s, %(humid)s, %(co2)s)')

cur.execute(query,row)

conn.commit()

query = ('DELETE FROM lassdataqueue WHERE device\_id = %(device\_id)s;')

cur.execute(query,tempDict)

conn.commit()

checkIFTTTDevice(row)

while checkFlush()>10:

doFlush()

# 寫入測站資料

query = ('SELECT `device\_id` FROM `lassdevice` WHERE `device\_id` = %(device\_id)s')

cur.execute(query,tempDict)

rows = cur.fetchall()

conn.commit()

if len(rows) == 0:

query = ('INSERT INTO `lassdevice` '

'(`device\_id`, `app`, `device`, `gps\_lat`, `gps\_lon`, `gps\_alt`, `time`, `pm1`, `pm10`, `pm25`, `temp`, `humid`, `co2`) '

'VALUES (%(device\_id)s, %(app)s, %(device)s, %(gps\_lat)s, %(gps\_lon)s, %(gps\_alt)s, %(time)s, %(s\_d2)s, %(s\_d1)s, %(s\_d0)s, %(temp)s, %(humid)s, %(s\_g8)s)')

# print(query, sep='\n', flush=True)

# print(query, sep='\n', flush=True)

else:

# tempDict['no'] = rows[0]['no']

query = ('UPDATE `lassdevice` '

'SET `app` = %(app)s, `device` = %(device)s, `gps\_lat` = %(gps\_lat)s, `gps\_lon` = %(gps\_lon)s, `gps\_alt` = %(gps\_alt)s, `time` = %(time)s, `pm1` = %(s\_d2)s, `pm10` = %(s\_d1)s, `pm25` = %(s\_d0)s, `temp` = %(temp)s, `humid` = %(humid)s, `co2` = %(s\_g8)s '

'WHERE `lassdevice`.`device\_id` = %(device\_id)s')

# print(query, sep='\n', flush=True)

# print(query, sep='\n', flush=True)

cur.execute(query,tempDict)

conn.commit()

tempDict = {}

except:

traceback.print\_exc()

# mqtt\_client.disconnect()

# return

def on\_disconnect(client, userdata,rc=0):

print("DisConnected result code "+str(rc))

client.loop\_stop()

cur.close()

conn.close()

mqtt\_client.connect(MQTT\_SERVER, MQTT\_PORT, MQTT\_ALIVE)

mqtt\_client = mqtt.Client()

mqtt\_client.on\_connect = on\_connect

mqtt\_client.on\_message = on\_message

mqtt\_client.connect(MQTT\_SERVER, MQTT\_PORT, MQTT\_ALIVE)

mqtt\_client.loop\_forever()

終端程式碼

/\*

This example demonstrate how to read pm2.5 value on PMS 3003 air condition sensor

PMS 3003 pin map is as follow:

PIN1 :VCC, connect to 5V

PIN2 :GND

PIN3 :SET, 0:Standby mode, 1:operating mode

PIN4 :RXD :Serial RX

PIN5 :TXD :Serial TX

PIN6 :RESET

PIN7 :NC

PIN8 :NC

In this example, we only use Serial to get PM 2.5 value.

The circuit: (BOARD RTL8195A)

\* RX is digital pin 0 (connect to TX of PMS 3003)

\* TX is digital pin 1 (connect to RX of PMS 3003)

pm25/pmtest.ino at master · x99wang/pm25 · GitHub

https://github.com/x99wang/pm25/blob/master/pmtest.ino

\*/

#include <WiFi.h>

#include <SoftwareSerial.h>

#include <PubSubClient.h>

#include <WiFiUdp.h>

//#if defined(BOARD\_RTL8195A)

//SoftwareSerial mySerial(0, 1); // RX, TX

//#elif defined(BOARD\_RTL8710)

SoftwareSerial PMS5003Serial(17, 5); // RX, TX

//#else

//SoftwareSerial mySerial(0, 1); // RX, TX

//#endif

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//WIFI設定

char ssid[] = "jim60105"; // your network SSID (name)

char pass[] = "000000000"; // your network password (use for WPA, or use as key for WEP)

int keyIndex = 0; // your network key Index number (needed only for WEP)

//終端編號

int clientNum = 0;

//server設定

int status = WL\_IDLE\_STATUS;

char server[] = "140.128.102.200"; // server address

int port = 80; // server port, use 80 for defult

//間隔時間

int delayTime = 60000; // interval for every http request (ms)

//送出計數

int count = 0;

//Fake GPS

char gps\_lat[] = "24.181598"; // device's gps latitude

char gps\_lon[] = "120.589623"; // device's gps longitude

char gps\_alt[] = "256"; // device's altitude above the sea level

//LASS MQTT

char mqttServer[] = "gpssensor.ddns.net"; // the MQTT server of LASS

char clientId[17] = "THU\_000"; // client id for MQTT

char outTopic[20] = "LASS/Test/PM25/live"; // MQTT publish topic

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WiFiClient client, client2;

PubSubClient mqttClient(client2);

WiFiUDP Udp;

const char ntpServer[] = "pool.ntp.org";

const int NTP\_PACKET\_SIZE = 48; // NTP time stamp is in the first 48 bytes of the message

const byte nptSendPacket[ NTP\_PACKET\_SIZE] = {

0xE3, 0x00, 0x06, 0xEC, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x31, 0x4E, 0x31, 0x34,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00

};

byte ntpRecvBuffer[ NTP\_PACKET\_SIZE ];

#define LEAP\_YEAR(Y) ( ((Y)>0) && !((Y)%4) && ( ((Y)%100) || !((Y)%400) ) )

static const uint8\_t monthDays[]={31,28,31,30,31,30,31,31,30,31,30,31}; // API starts months from 1, this array starts from 0

uint32\_t epochSystem = 0; // timestamp of system boot up

//Sensor資料格式宣告

struct SensorValuesBar {

float temp;

float humid;

float pm1;

float pm25;

float pm10;

};

struct SensorValuesBar PMS5003Value;

long pmcf10, pmcf25, pmcf100, pmat10, pmat25, pmat100;

long sch2o,stemp,shumid;

struct SensorValuesBar getPMS5003(){

struct SensorValuesBar result;

int count = 0;

unsigned char readBuff;

unsigned char highBits;

while (PMS5003Serial.available()) {

readBuff = PMS5003Serial.read();

if((count==0 && readBuff!=0x42) || (count==1 && readBuff!=0x4d)){

break;

}

if(count > 36){

break; // have read all data, read ends

}

else if(count == 4 || count == 6 || count == 8 || count == 10 || count == 12 || count == 14 || count == 24 || count == 26) highBits = readBuff;

else if(count == 5){

pmcf10 = 256\*highBits + readBuff;

}

else if(count == 7){

pmcf25 = 256\*highBits + readBuff;

//Serial.print("CF=1, PM2.5=");Serial.print(pmcf25);Serial.println(" ug/m3");

}

else if(count == 9){

pmcf100 = 256\*highBits + readBuff;

//Serial.print("CF=1, PM10=");Serial.print(pmcf100);Serial.println(" ug/m3");

}

else if(count == 11){

pmat10 = 256\*highBits + readBuff;

result.pm1 = pmat10; // pm1.0

//Serial.print("atmosphere, PM1.0=");Serial.print(pmat10);Serial.println(" ug/m3");

}

else if(count == 13){

pmat25 = 256\*highBits + readBuff;

result.pm25 = pmat25; // pm2.5

//Serial.print("atmosphere, PM2.5=");Serial.print(pmat25);Serial.println(" ug/m3");

}

else if(count == 15){

pmat100 = 256\*highBits + readBuff;

result.pm10 = pmat100; // pm10.0

//Serial.print("atmosphere, PM10=");Serial.print(pmat100);Serial.println(" ug/m3");

}

else if(count == 25){

stemp = 256\*highBits + readBuff;

result.temp = stemp/10.0;

//Serial.print("atmosphere, TEMP=");Serial.print(temp);Serial.println(" `C");

}

else if(count == 27){

shumid = 256\*highBits + readBuff;

result.humid = shumid/10.0;

//Serial.print("atmosphere, HUMID=");Serial.print(humid);Serial.println(" %");

}

count++;

}

Serial.println();

while(PMS5003Serial.available())

PMS5003Serial.read();

return result;

}

void connectToWifi() {

// attempt to connect to Wifi network:

while (status != WL\_CONNECTED) {

Serial.print("Attempting to connect to SSID: ");

Serial.println(ssid);

// Connect to WPA/WPA2 network. Change this line if using open or WEP network:

status = WiFi.begin(ssid, pass);

// wait 10 seconds for connection:

delay(10000);

}

Serial.println();

Serial.println("Connected to wifi.");

// print the SSID of the network you're attached to:

Serial.print("SSID: ");

Serial.println(WiFi.SSID());

// print your WiFi shield's IP address:

IPAddress ip = WiFi.localIP();

Serial.print("IP Address: ");

Serial.println(ip);

// print the received signal strength:

long rssi = WiFi.RSSI();

Serial.print("signal strength (RSSI):");

Serial.print(rssi);

Serial.println(" dBm");

// local port to listen for UDP packets

Udp.begin(2390);

}

// send an NTP request to the time server at the given address

void retrieveNtpTime() {

for (int retry = 0; retry < 5; retry ++) {

Serial.println("Send NTP packet");

Udp.beginPacket(ntpServer, 123); //NTP requests are to port 123

Udp.write(nptSendPacket, NTP\_PACKET\_SIZE);

Udp.endPacket();

delay(3000);

if(Udp.parsePacket()) {

Serial.println("NTP packet received");

Udp.read(ntpRecvBuffer, NTP\_PACKET\_SIZE); // read the packet into the buffer

unsigned long highWord = word(ntpRecvBuffer[40], ntpRecvBuffer[41]);

unsigned long lowWord = word(ntpRecvBuffer[42], ntpRecvBuffer[43]);

unsigned long secsSince1900 = highWord << 16 | lowWord;

const unsigned long seventyYears = 2208988800UL;

unsigned long epoch = secsSince1900 - seventyYears;

epochSystem = epoch - millis() / 1000;

break;

}

}

}

void getCurrentTime(unsigned long epoch, int \*year, int \*month, int \*day, int \*hour, int \*minute, int \*second) {

int tempDay = 0;

\*hour = (epoch % 86400L) / 3600;

\*minute = (epoch % 3600) / 60;

\*second = epoch % 60;

\*year = 1970; // epoch starts from 1970

\*month = 0;

\*day = epoch / 86400;

for (\*year = 1970; ; (\*year)++) {

tempDay += (LEAP\_YEAR(\*year) ? 366 : 365);

if (tempDay > \*day) {

tempDay -= (LEAP\_YEAR(\*year) ? 366 : 365);

break;

}

}

tempDay = \*day - tempDay; // the days left in a year

for ((\*month) = 0; (\*month) < 12; (\*month)++) {

if ((\*month) == 1) {

tempDay -= (LEAP\_YEAR(\*year) ? 29 : 28);

if (tempDay < 0) {

tempDay += (LEAP\_YEAR(\*year) ? 29 : 28);

break;

}

} else {

tempDay -= monthDays[(\*month)];

if (tempDay < 0) {

tempDay += monthDays[(\*month)];

break;

}

}

}

\*day = tempDay+1; // one for base 1, one for current day

(\*month)++;

}

void initializeMQTT() {

//byte mac[6];

//WiFi.macAddress(mac);

//RTL-00 MAC addresses are all the same.

//sprintf(clientId, "THU\_%02X%02X%02X%02X", mac[2], mac[3], mac[4], mac[5]);

Serial.print("MQTT client id:");

Serial.println(clientId);

Serial.print("MQTT topic:");

Serial.println(outTopic);

mqttClient.setServer(mqttServer, 1883);

}

void sendMQTT(SensorValuesBar PMS5003Value) {

String payload;

char timeStr[30];

unsigned long epoch = epochSystem + millis() / 1000;

int year, month, day, hour, minute, second;

getCurrentTime(epoch, &year, &month, &day, &hour, &minute, &second);

sprintf(timeStr,"Current time: date=%4d-%02d-%02d time=%02d:%02d:%02d", year, month, day, hour, minute, second);

Serial.println(timeStr);

if (!mqttClient.connected()) {

Serial.println("Attempting MQTT connection");

mqttClient.connect(clientId);

}

if (mqttClient.connected()) {

payload = "|ver\_format=3|Fake\_GPS=1|app=PM25|ver\_app=live|device\_id="+String(clientId)+"|tick="+millis()+"|date="+year+"-"+month+"-"+day+"|time="+hour+":"+minute+":"+second

+"|device=Ameba|s\_d0="+PMS5003Value.pm25

+"|s\_d1="+PMS5003Value.pm10

+"|s\_d2="+PMS5003Value.pm1

+"|s\_h0="+PMS5003Value.humid

+"|s\_t0="+PMS5003Value.temp

+"|gps\_lat="+gps\_lat

+"|gps\_lon="+gps\_lon

+"|gps\_fix=1|gps\_alt="+gps\_alt;

Serial.println(payload);

// Once connected, publish an announcement...

char companionchannel[50]="";

strcat(companionchannel,outTopic);

strcat(companionchannel,"/");

strcat(companionchannel,clientId);

mqttClient.publish((char\*)outTopic, payload.c\_str());

mqttClient.publish((char\*)companionchannel, payload.c\_str());

mqttClient.disconnect();

}

}

String serverStr = String(server);

void connect2server(String jsonStr) {

//送出資料到伺服器

Serial.println("\nStarting connection to server...");

// if you get a connection, report back via serial:

if (client.connect(server, port)) {

client.print("GET /assets/php/addData.php?");

client.print(jsonStr);

client.println(" HTTP/1.1");

client.println("Host: "+serverStr+":"+port);

client.println("User-Agent: Arduino/1.0");

client.println();

}else{

Serial.println("\nFailed to connect to the server.");

}

}

void setup() {

//Initialize serial and wait for port to open:

Serial.begin(38400);

while (!Serial) {

;

}

status = WiFi.status();

// check for the presence of the shield:

if (status == WL\_NO\_SHIELD) {

Serial.println("WiFi shield not present");

// don't continue:

while (true);

}

String fv = WiFi.firmwareVersion();

if (fv != "1.1.0") {

Serial.println("Please upgrade the firmware");

}

if(status != WL\_CONNECTED) {

connectToWifi();

}

PMS5003Serial.begin(9600); // PMS 3003 UART has baud rate 9600

retrieveNtpTime();

initializeMQTT();

}

void loop() { // run over and over

Serial.println("v18.03.21.0");

//斷線重連

if(WiFi.status()!= WL\_CONNECTED) {

connectToWifi();

retrieveNtpTime();

}

//每60次送出校時一次

if((count%60)==0){

retrieveNtpTime();

}

//硬件初始化

PMS5003Value = getPMS5003();

Serial.print("pm1:");

Serial.println(PMS5003Value.pm1);

Serial.print("pm10:");

Serial.println(PMS5003Value.pm10);

Serial.print("pm25:");

Serial.println(PMS5003Value.pm25);

Serial.print("temp:");

Serial.println(PMS5003Value.temp);

Serial.print("humid:");

Serial.println(PMS5003Value.humid);

//組合資料

String jsonStr = (String)"pm1="+PMS5003Value.pm1+"&pm10="+PMS5003Value.pm10+"&pm25="+PMS5003Value.pm25+"&temp="+PMS5003Value.temp+"&humid="+PMS5003Value.humid+"&clientNum="+clientNum;

connect2server(jsonStr);

sendMQTT(PMS5003Value);

mqttClient.loop();

Serial.println("Disconnecting from server...");

client.stop();

Serial.print("Interval ");

Serial.print(delayTime);

Serial.println(" ms...");

count++;

delay(delayTime);

}

# 參考資料

[1] Python

https://zh.wikipedia.org/wiki/Python

[2] MariaDB

https://zh.wikipedia.org/wiki/MariaDB

[3] Arch Linux

https://zh.wikipedia.org/wiki/Arch\_Linux

[4] 樹莓派

https://zh.wikipedia.org/wiki/%E6%A0%91%E8%8E%93%E6%B4%BE

[5] Realtek lot/Arduino Solution

https://www.amebaiot.com/ameba-arduino-getting-started-rtl8710/

[6] 淡江大學電子數位論文服務 http://etds.lib.tku.edu.tw/etdservice/view\_metadata?etdun=U0002-2001201815270500&from=DEPT&deptid=D0002003018

[7] 空氣污染 https://zh.wikipedia.org/wiki/%E7%A9%BA%E6%B0%A3%E6%B1%A1%E6%9F%93