Final Project - Report

0713221 黃家綺 / 109550172 何彥寬

1. Objective:

- 改良智慧植物系統，增加更多功能以及優化程式。
- 利用Siri 實現語音控制，提供使用者更便利的操作。
- 利用樹莓派與MCS溝通，透過繼電器去提供自動與手動控制燈與澆花裝置去實現智慧植物系統。
- 製作一個易於擴充的系統，像是增加每日氣溫以及空氣品質報告等等，讓它除了照顧植物外還能提供使用者其它方便性。
- 製作一個紀錄溫溼度的Database，以便觀察改良。

2. Sensors and Actuators Used:

- Soil Moisture Sensor
- Light Sensor
- I2C LCD
- DHT22
- Buzzer
- RGB LED
- Raindrop Sensor
- 4 Channel Capacitive Touch Module
- Dust Sensor

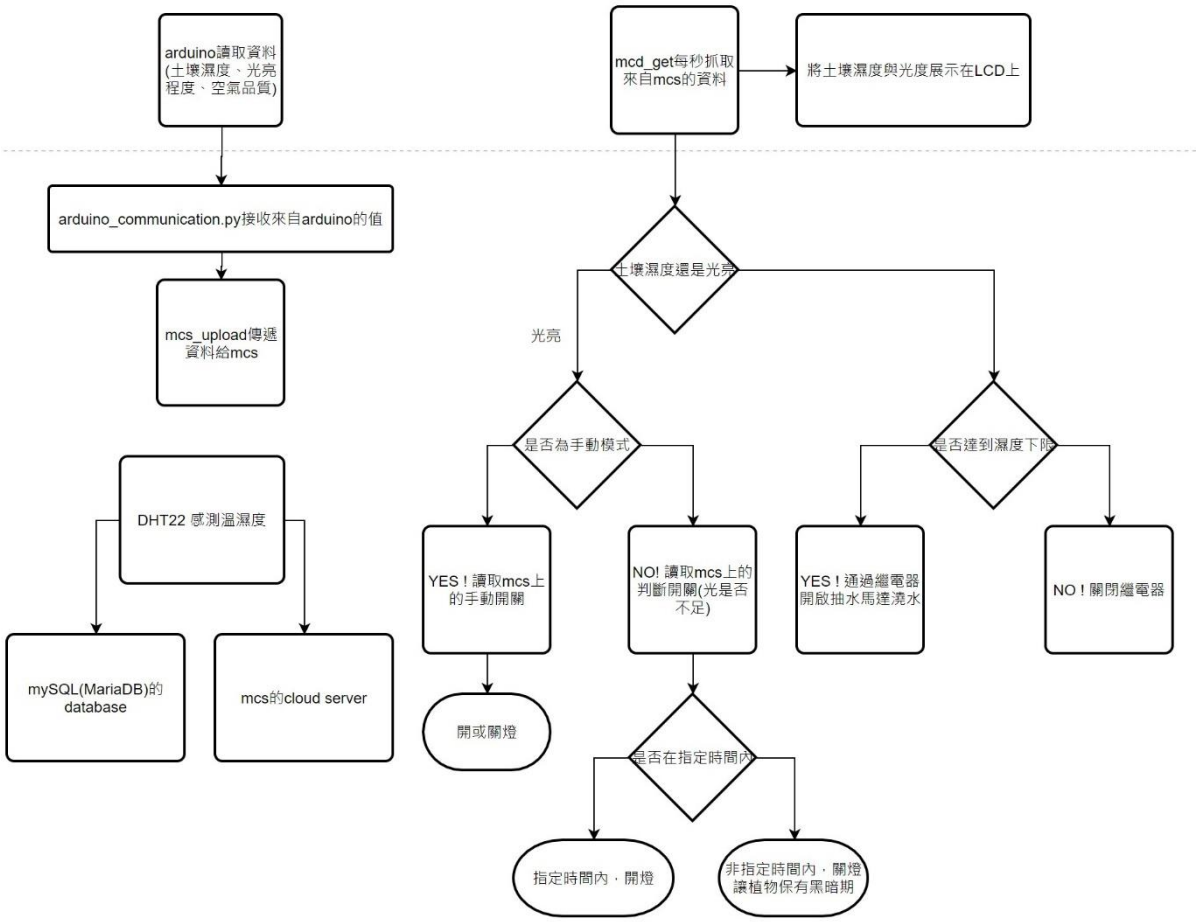
Extra:

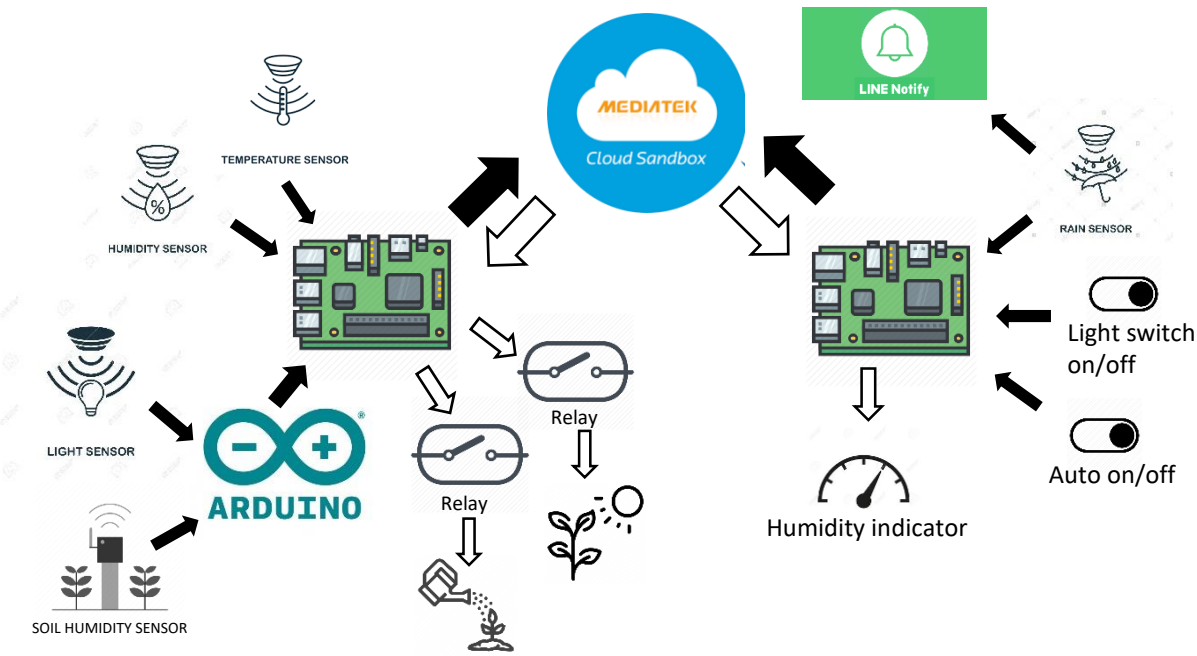
- Arduino
- Siri Control
- Line Notify
- Light system
  - 12V 5050 12SMD LED White Light
  - Relay Module
  - Transformer
  - Plug
- Watering system
  - Mini Submersible Water Pump
  - Water Pipe
  - Relay Module
  - Bottle with Water

3. System Design:

- 在Lab2 、Lab3 中，我們已經介紹了有關植物的整個System Design 。現在我們除了想做好一個智慧植物系統之外，也希望帶給使用者更多的功能，讓它不單純只能照料植物。因此我們透過用來判斷植物生長環境的各種sensor 同時當作我們對於外面環境的檢測，利用Line Notify 以及在程式內判斷時間，在每天早上傳送一個訊息告知使用者現在外面的氣溫以及空氣品質如何，讓使用者在出門前可以有個參考依據。
- 為了增加使用者在操作上的便利性，我們選擇透過Siri 當作語音控制的部分，因為Siri 本身就已具備強大的語音判斷能力，而手機也會是我們平常就會用到的工具，讓使用者隨時隨地都能方便控制系統。而操作Siri 的部份我們選擇了較為輕鬆並結合課程的運用SSH，讓手機經此連線樹莓派運行指令。這種方式也能讓人很好擴充想要的功能與指令。
- 為了做出更全面的系統，我們也改良了原本的code，讓其更好去擴充系統。例如：我們可以將其放置於玄關或前面庭院，配合相機及紅外線偵測，以及現已做出的聲控開關燈的裝置，讓它可以身兼智慧家居的一部份系統，礙於成本與時間的考量，在final project 無法實作。

4. Flowchart:


arduino讀取資料
(土壤濕度、光亮程度、空氣品質)
arduino_communication.py接收來自arduino的值
mcs_upload傳遞資料給mcs
DHT22 感測溫濕度
mySQL(MariaDB)的database
mcs的cloud server
mcd_get每秒抓取來自mcs的資料
將土壤濕度與光度展示在LCD上
土壤濕度還是光亮
光亮
是否為手動模式
是否達到濕度下限
YES！讀取mcs上的手動開關
NO! 讀取mcs上的判斷開關(光是否不足)
YES！通過繼電器開啟抽水馬達澆水
NO！關閉繼電器
開或關燈
是否在指定時間內
指定時間內，開燈
非指定時間內，關燈讓植物保有黑暗期


MEDIATEK
Cloud Sandbox
LINE Notify
TEMPERATURE SENSOR
HUMIDITY SENSOR
LIGHT SENSOR
SOIL HUMIDITY SENSOR
ARDUINO
Relay
Relay
RAIN SENSOR
Light switch on/off
Auto on/off
Humidity indicator

5. Code Explanation and The Differences With Example Codes:

(You can find all the source code in the file named source code or on https://github.com/Kris57880/raspberry_git)

1) Source Code which Changed or Added:
   - Main Raspberry Pi
     - mcs_get
     - arduino_communication
     - dht22
   - Second Raspberry Pi
     - mcs_get_other
     - main
     - Siri_control_on
     - Siri_control_off
     - Siri_line
     - Siri_control_mode_off

2) Main Raspberry Pi:

   a) mcs_get:

mcs_get.py

```
80          GPIO.output(watering,GPIO.LOW)
81
82  def display(sensor):
83      endpoint = "/mcs/v2/devices/" + deviceId_host + "/datachannels/"+sensor+"/datapoints"
84      headers = {"Content-type": "application/json", "deviceKey": deviceKey_host}
85      url = host + endpoint
86      r = requests.get(url,headers=headers)
87      value = (r.json()["dataChannels"][0]["dataPoints"][0]["values"]["value"])
88      if sensor == 'soil_display':
89          print("soil(%) : "+str(value))
90          if value<=1:
91              i2c_lcd.message("                ",0,0,False)
92          i2c_lcd.message("soil(%) : "+str(value),  0, 0, False )
93      else :
94          i2c_lcd.message("ldr(%)  : "+str(value),  1, 0, False )
95          print("ldr(%)  : "+str(value))
96
97  while True:
98      ldr()
99      time.sleep(1)
100     soil()
101     time.sleep(1)
102
```

上次demo中我們加入手動開關的功能，但我們也發現那時候的程式有嚴重的延遲問題，做為改進方式，這次我們將get 函式獨立成一隻程式，讓他分別執行，實測後以Siri 手動控制時能達到極低的延遲。

b)　Arduino_communication :

由於這次新增了Dust Sensor ，此傳感器透過加熱自動進氣裝置，使用計數法測試粉塵濃度，他可以通過在給定的單位時間內，計算羅脈衝佔用時間（LPO時間）來測量空氣中的顆粒物水平
先說明arduino 的程式碼：

```
int CDSPin = 0; // 光敏電阻接在A0接腳
int CDSVal = 0; // 設定初始光敏電阻值為0
int SoilPin = 1;
int SoilVal = 0;
void setup() {
  Serial.begin(9600);
  pinMode(DUST_PIN, INPUT);
  starttime = millis(); //get the current time;

}

void loop() {
  duration = pulseIn(DUST_PIN, LOW);
  lowpulseoccupancy = lowpulseoccupancy + duration;
  now = millis();
  if ((now - starttime) > DUST_SAMPLE_MS) {
        ratio = lowpulseoccupancy / (DUST_SAMPLE_MS * 10.0);  // Integer percentage 0=>100
        concentration = 1.1 * pow(ratio, 3) - 3.8 * pow(ratio, 2) + 520 * ratio + 0.62; // using spec sheet curve

        Console.print(ratio);
        Serial.print(ratio);
        delay(3000);
        CDSVal = analogRead(CDSPin);
        Serial.println(CDSVal);
        delay(3000)
        SoilVal = analogRead(SoilPin);
        Serial.println(SoilVal);
        lowpulseoccupancy = 0;
        starttime = millis();
   }

}
```

程式記錄了30秒內的低電平脈衝佔用時間，通過它計算環境粉塵濃度。 **“lo pulse occupancy”** 表示30秒內羅脈衝佔用時間，它的單位是微秒。 **“ratio”** 反映的是LPO 時間佔總時間的比例。 並在30秒後依序輸出灰塵濃度，光敏數值與土壤濕度，設定延遲避免同時輸出造成樹莓派端混亂。
同時我們也需要對樹莓派的python程式做修改：

```
while True:
    for i in range(3):
        read = ser.readline()
        if i== mode['AIR'] :
            read_serial =float(read[0:-2])
            mcs_upload.post_data(read_serial,"air")
            time.sleep(1)
        elif i == mode['SOIL'] :
            read_serial =int(read[0:-2])
            mcs_upload.post_data(read_serial,"soil_sensor")
            #print(read_serial)
        elif i == mode['LDR'] :
            read_serial =int(read[0:-2])
            mcs_upload.post_data(read_serial,"ldr_sensor1")
            #print(read_serial)

```

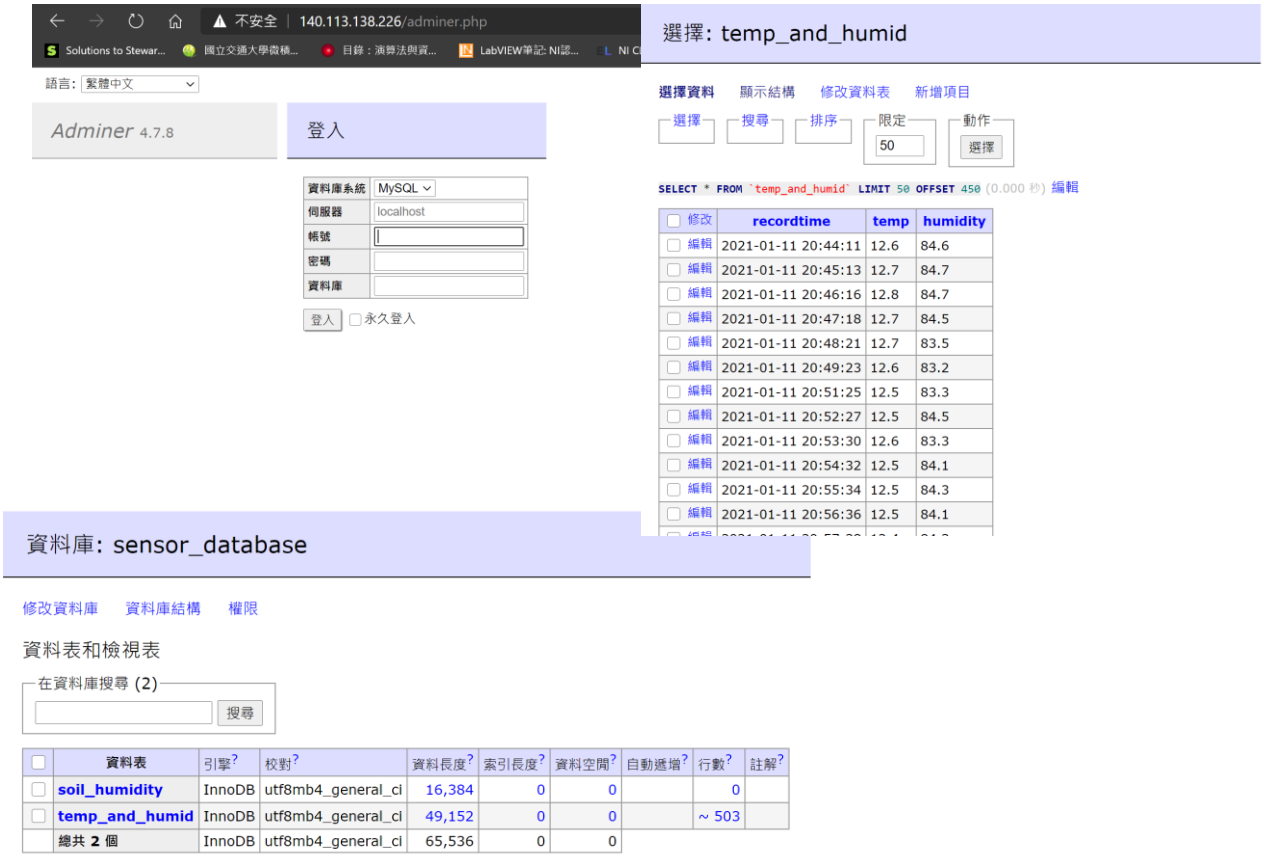
我們在MCS 上加入新的Device ID 並在程式中設定新的順序、list與迴圈，由於之前mcs_upload 設計成通用的格式，只要輸入ID 跟值即可以使用，code不必做大幅更動，顯示了這個系統的高度擴充性。

c)　dht22:

我們使用mysql 的database 創建一個溫溼度的資料庫，並將值上傳至該資料庫，同時我們也使用php 創建一個網站供使用者瀏覽上傳的紀錄。

```
aa=datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
try:
    t = dht_device.temperature
    h = dht_device.humidity
    if h is not None and t is not None:
        print('temperature={0:0.1f} C humidity={1:0.1f}%'.format(t, h))
        #i2c_lcd.message("tempture "+str(t)+"c",0,0)
        #i2c_lcd.message("humidity "+str(h)+"%",1,0,False)
        mcs_upload.post_data(t,"celsius")
        conn= pymysql.connect(
        host='localhost',
        port = 3306,
        user='root',
        passwd='12345',
        db ='sensor_database',
        )
        cur = conn.cursor()
        # cur.execute("creat table if not exists temp(time int,temperature int,humidity int)")
        cur.execute("insert into temp_and_humid values('%s','%f','%f')"%(aa,t,h))
        conn.commit()
```

我們先在shell中創建資料庫與資料表格，透過python 插入資料，其中我們必須import pymysql 函式庫，表格中包含時間戳與溫溼度資料，一開始我們使用time.localtime 函式獲得時間，但發現時間有誤後改成datetime 函式庫的now()便恢復正常，同時我們也使用adminer 建立一個網站供線上瀏覽資料庫(可使用外網瀏覽)。



3) Second Raspberry Pi

a) Codes about siri:

- Siri_control_mode_off

```python
import RPi.GPIO as GPIO
import time
import mcs_upload_other

mcs_upload_other.post_to_mcs('mode', 0)
```

- Siri_control_on

```python
import RPi.GPIO as GPIO
import time
import mcs_upload_other

mcs_upload_other.post_to_mcs('mode', 1)
time.sleep(1)
mcs_upload_other.post_to_mcs('man_ctl', 1)
```

- Siri_control_off

```python
import RPi.GPIO as GPIO
import time
import mcs_upload_other

mcs_upload_other.post_to_mcs('mode', 1)
time.sleep(1)
mcs_upload_other.post_to_mcs('man_ctl', 0)
```

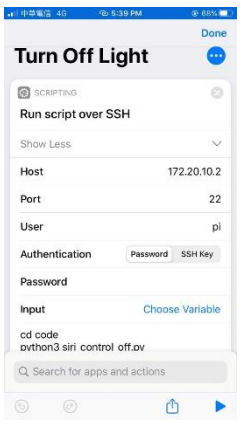
- Siri_line

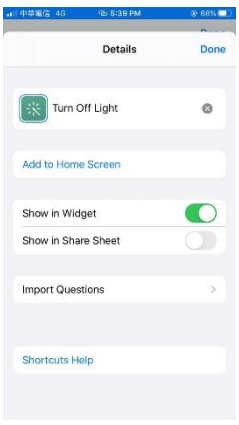
```python
import time
import RPi.GPIO as GPIO
import mcs_upload_other
import mcs_get_other
import Line_notifier as notifier

value1 = mcs_get_other.status('ldr_display')
value2 = mcs_get_other.status('soil_display')
value3 = mcs_get_other.status('celsius')
value4 = mcs_get_other.status('air_dis')
notifier.line_message("\nLdr(%)  :" + str(value1) + "\nSoil(%) :" + str(value2) + "\nTemp :" + str(value3) + "\nAir Quality(%) :" + str(value4))
```

透過原本就已經寫好的manual control 的function ，利用與touch_switch 相似的方式去實行控制手動與自動控制的切換，而Line的部分也是透過原本就寫好的line notifier 的function 與mcs_get_other 實作。

b) Siri 實行

- 考量系統設計，我們決定不以Node.js 去實作將樹莓派以HomePad裝置顯示在iphone 去操作，而是用更簡單的SSH去運行，利用Shortcuts 運行腳本以連接樹莓派完成想要的操作。
- Shortcut 名稱可以以Siri 呼叫，也可建立Home screen 捷徑。
- 第二張圖下方為想運行的腳本，利用它執行上方已經寫好的code。
- (實際操作講解在影片中)





c) mcs_get_other:

```python
import time
import http.client, urllib
import json
import RPi.GPIO as GPIO
import requests
import socket
import serial

deviceId = "DfgskicX" #109550172
deviceKey = "0WXK7SwuDx2Vp9Qc"
#deviceId = "D5Wu3e3g"
#deviceKey = "DQ4P5W3mkcHIepq8"

host = "http://api.mediatek.com"
headers = {"Content-type": "application/json", "deviceKey": deviceKey}

def status(sensor):
    endpoint = "/mcs/v2/devices/" + deviceId + "/datachannels/" + sensor + "/datapoints"
    url = host + endpoint
    r = requests.get(url,headers=headers)
    value = (r.json()["dataChannels"][0]["dataPoints"][0]["values"]["value"])
    return value
```

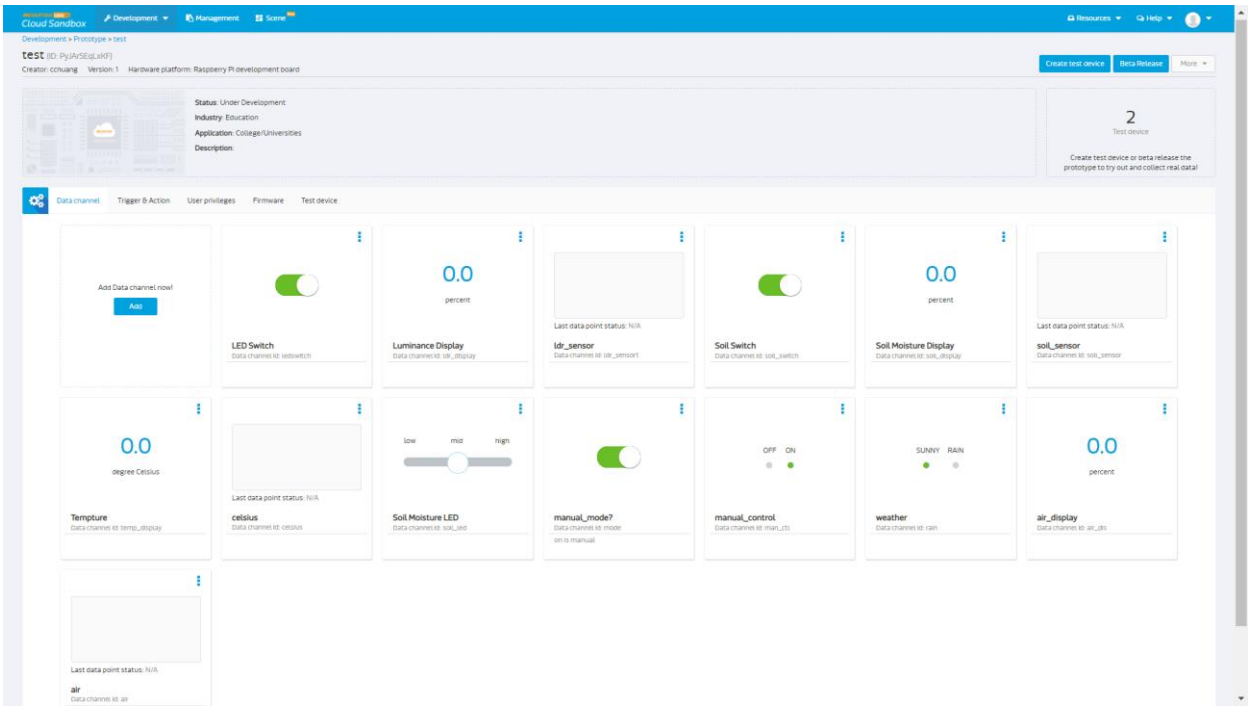
原本的Second Raspberry Pi 只有需要get Soil 的Data Channel 的值，因此專門寫了一個function 給它。但這次我們讓它從更多的Data Channel 中去取得數據以通知使用者，所以將程式改寫得更全面更通用，讓其它程式呼叫時可以以同一個function 配上不同的Channel ID 即可。

d) main:

```python
import RPi.GPIO as GPIO
import time
import Line_notifier as notifier
import mcs_get_other
import RGB_LED

while True:
    now_hour=int(time.strftime("%H"))
    now_min=int(time.strftime("%M"))
    now_sec=int(time.strftime("%S"))

    if(now_hour == 8 and now_min== 0 and now_sec < 5):
        value1 = mcs_get_other.status('celsius')
        value2 = mcs_get_other.status('air_dis')
        notifier.line_message("\nTemp  :" + str(value1) + "\nAir Quality(%)  :" + str(value2))

    value = mcs_get_other.status('soil_led')
    RGB_LED.display(int(value))
    time.sleep(1)
```

此為這次新增的程式，因為也會有其它程式需要呼叫get 的function，因此我把原本在get 所執行的部分移到了一個新的程式，讓get 能專司其職不被打擾。而在這也新增了一個判斷是否在每天早上八點的條件句，在所設的時間點之中，讓它傳送Line 訊息給使用者告知今日的溫度與空汙百分比，讓使用者每天早上出門前有資訊可以參考。而upload的部分則是由原本的程式呼叫，因先前報告提過不再贅述。

4) JavaScript on MCS



a) New Data Channel:

- Air (讓樹莓派回傳Dust Sensor的值）
- Air_dis （顯示空氣品質趴數）

b) Source Code which Changed or Added:

a. Air

因為新增了Dust Sensor ，所以新增了這個Data Channel 讓MCS 得以接收，並把值傳給air_dis 讓其顯示出給使用者觀看。