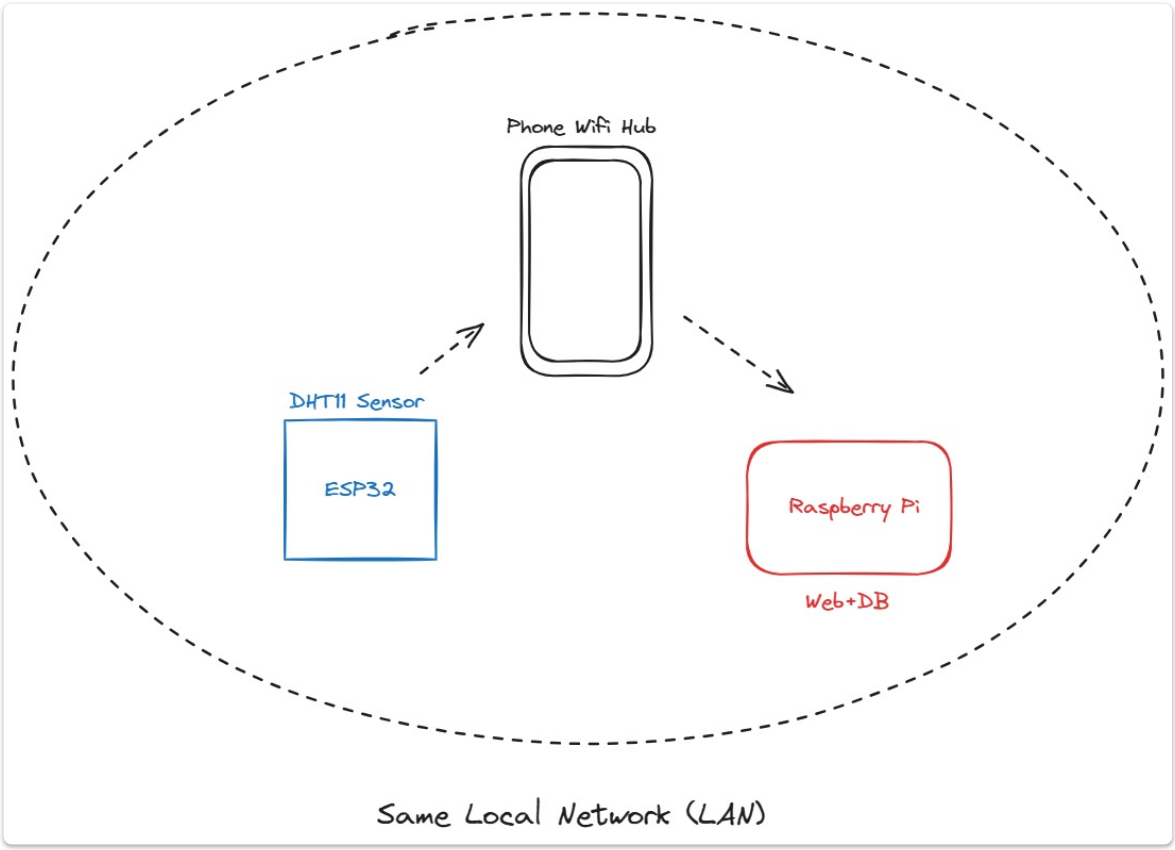
Aiot_HW6_4108042003

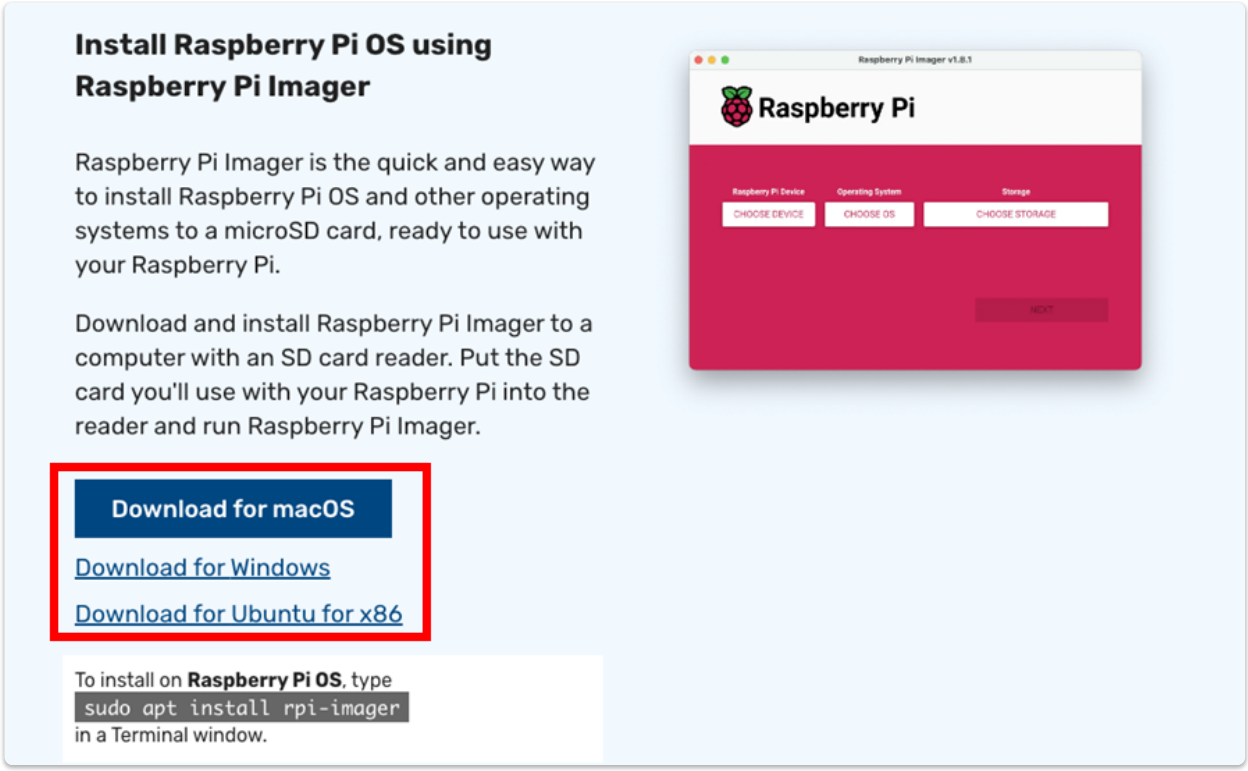
#物聯網 #作業

本次重點

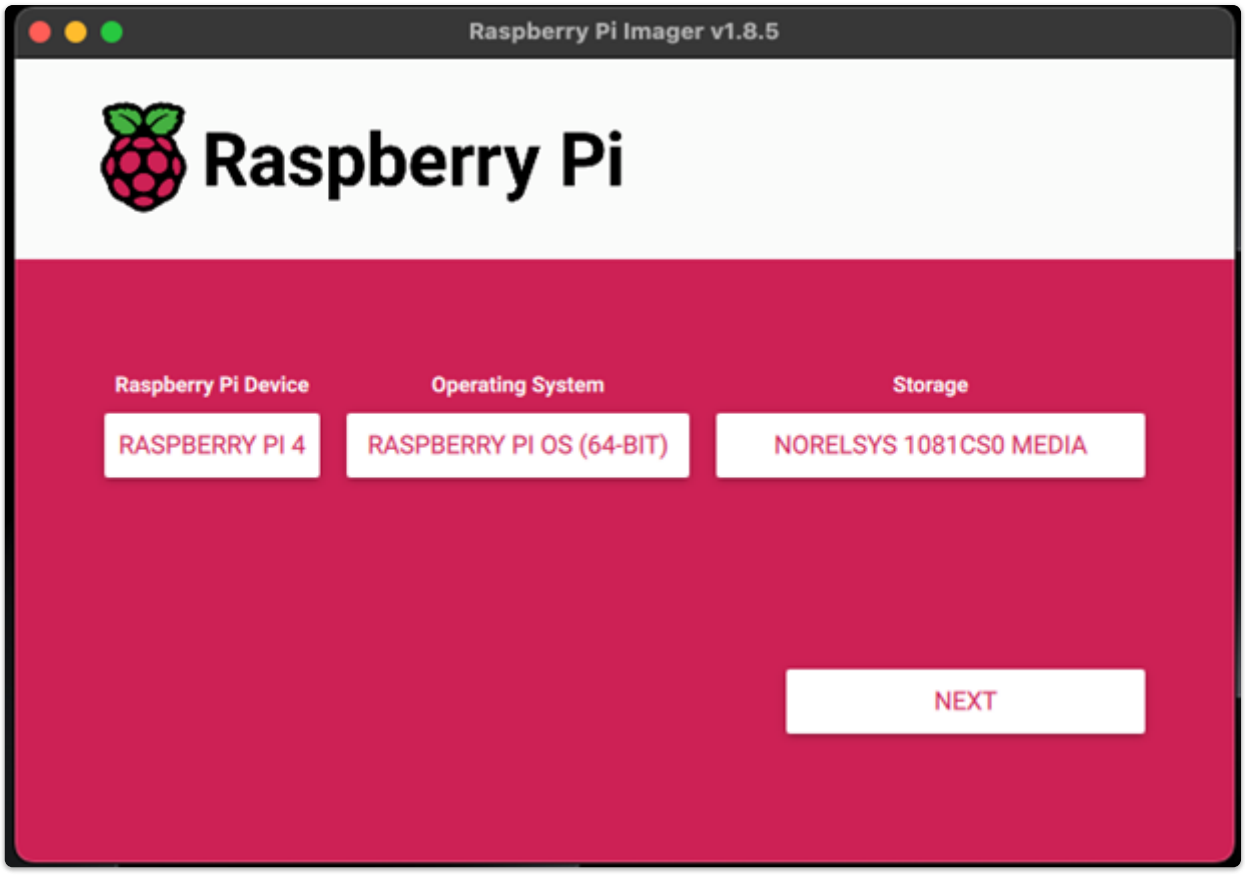
- 延續Lab2,3課程內容，於Raspberry Pi 4上進行實作
- 使用ESP32連接DHT11，取得溫溼度資料
- 在樹莓派上用flask架設伺服器，儲存資料到資料庫
- 在網頁上渲染資料並動態更新
- 整體架構
  - 

  - 所有設備皆連結於相同網路

6-1 Raspberry Pi 4 刷機



- 下載raspberry pi imager，並安裝到一張64GB記憶卡(圖源:ilearning教材)



- 開啟並選擇正確的版本後刷機(圖源:ilearning教材)

- 刷機完成後，將MicroSD卡放置到Rpi4背面，並接上電源線、螢幕連接線、鍵盤、滑鼠後，開機
- 開機後，進行初始化設定（設定名稱、密碼、時區等資訊）
- 將Rpi4連接網路（本次使用手機熱點）

6-2 取得溫溼度資料

- 沿用個人HW3之程式碼(tempToDB.ino)進行實作
- 在自己的電腦上開啟Adruino執行
- ESP32連接DHT11來獲取溫濕度數值

程式碼

```
#include <WiFi.h>
#include <DHT11.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>

const char *ssid = "Anna333";
const char *password = "annaWiFi";

const char *serverAddress = "http://192.168.129.57:5000/post_data";
DHT11 dht11(15);

void setup() {
  Serial.begin(115200); //鮑率設定 115200
  WiFi.begin(ssid, password);//網路設定
  //等待網路連線
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.println("Connecting to WiFi...");
  }
  //連上網路並顯示目前 IP
  Serial.println("Connected to WiFi");
  Serial.println(WiFi.localIP());
}


void loop() {
  int temperature = 0;
  int humidity = 0;
  int result = dht11.readTemperatureHumidity(temperature, humidity);
  if (result == 0) {
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.print(" °C\tHumidity: ");
    Serial.print(humidity);
    Serial.println(" %");
  } else {
  // Print error message based on the error code.
    Serial.println(DHT11::getErrorString(result));
  }

 // Create an object JSON
  JsonDocument jsonDoc;
```

```
  jsonDoc["temperature"] = temperature;
  jsonDoc["humidity"] = humidity;
   // Serializa el JSON en una cadena
  String payload;
  serializeJson(jsonDoc, payload);

   // Crea una instancia de HTTPClient
  HTTPClient http;
  Serial.println("Server Address: " + String(serverAddress));

  // Envía la solicitud POST al servidor
  http.begin(serverAddress);

  http.addHeader("Content-Type", "application/json");
  int httpResponseCode = http.POST(payload);

  if (httpResponseCode > 0) {
    Serial.printf("HTTP Response code: %d\n", httpResponseCode);
    String response = http.getString();
    Serial.println(response);
  } else {
    Serial.printf("HTTP Request failed: %s\n",
    http.errorToString(httpResponseCode).c_str());
  }
  http.end();
  delay(5000);
  Serial.println();
}
```

執行結果

-
```
Temperature: 29 °C      Humidity: 67 %
Server Address: http://192.168.129.57:5000/post_data
HTTP Request failed: connection refused
```
  - 可看到溫溼度資料已取得
  - 但尚未開啟伺服器，因此還無法Post資料

6-3 架設伺服器與展現

- 在樹莓派上架設flask後端伺服器
- 前端網頁渲染資料並動態更新

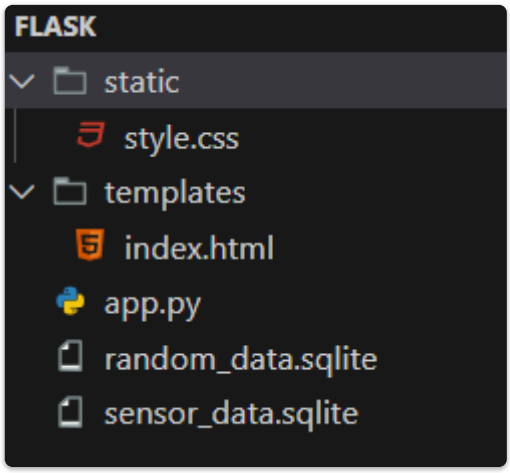
程式碼

- 開啟終端機執行以下指令

安裝python與套件

```
sudo apt update
sudo apt install python3
sudo apt install python3-pip
sudo pip3 install flask
```

編寫flask應用程式

- 沿用個人HW3之程式碼(flask folder內)進行實作
- 

  - /statics: 存放js,css等靜態資料
  - /templates: 存放html模板
  - app.py: 主程式進入點，管理所有前後端，分頁管理等資訊

後端(app.py)

- 直接更新HW3內容，增加了取得資料的參數設定
- 增加能自由選擇獲取前幾筆資料(HW3固定為前30筆)
- 增加能選擇獲取哪天的所有資料(HW3無)

```
from flask import Flask, render_template, request, jsonify
import sqlite3
from datetime import datetime
import random
import threading
import time

app = Flask(__name__)

DB_SENSOR_NAME = "sensor_data"
DB_RANDOM_NAME = "random_data"

DB_SENSOR = DB_SENSOR_NAME + ".sqlite"
DB_RANDOM = DB_RANDOM_NAME + ".sqlite"

def create_table(conn, table_name):
    cursor = conn.cursor()
```

```
    cursor.execute(
        f"""CREATE TABLE IF NOT EXISTS {table_name} (
            id INTEGER PRIMARY KEY,
            temperature REAL NOT NULL,
            humidity REAL NOT NULL,
            timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
        )"""
    )
    conn.commit()

def init_databases():
    try:
        conn_sensor = sqlite3.connect(DB_SENSOR)
        create_table(conn_sensor, DB_SENSOR_NAME)
        conn_sensor.close()

        conn_random = sqlite3.connect(DB_RANDOM)
        create_table(conn_random, DB_RANDOM_NAME)
        conn_random.close()

        print("Databases initialized successfully.")
    except sqlite3.Error as e:
        print("Error occurred during database initialization:", e)

# Render the main page template
@app.route("/")
def index():
    return render_template("index.html")

# Insert data into the database
def insert_data(temperature, humidity, timestamp=None,
db_name=DB_SENSOR_NAME):
    if temperature == 0 and humidity == 0:
        print("Temperature and humidity are both zero, skipping insertion
into the database.")
        return

    db = DB_SENSOR if db_name == "sensor_data" else DB_RANDOM
    try:
        conn = sqlite3.connect(db)
        cursor = conn.cursor()

        if timestamp is None:
            timestamp = datetime.now().replace(microsecond=0)

        cursor.execute(
            f"""INSERT INTO {db_name} (temperature, humidity, timestamp)
VALUES (?, ?, ?)""",
            (temperature, humidity, timestamp),
```

```
        )

        conn.commit()
        conn.close()
        print(f"({db_name})Data inserted successfully.")
    except sqlite3.Error as e:
        print(f"{db_name})Error inserting data: {e}")

# Generate and insert random data every 2 seconds
def generate_random_data():
    while True:
        temp = random.randint(20, 40)
        humid = random.randint(20, 40)
        insert_data(temp, humid, db_name=DB_RANDOM_NAME)
        # Wait for 2 seconds
        time.sleep(2)


# Receive sensor data and insert it into the database
@app.post("/post_data")
def receive_data():
    try:
        content = request.get_json()
        temperature = content["temperature"]
        humidity = content["humidity"]

        insert_data(temperature, humidity)

        print(f"Received data: temperature={temperature}, humidity=
{humidity}")
        return jsonify({"success": True})
    except Exception as e:
        print(f"Error receiving data: {str(e)}")
        return jsonify({"success": False, "error": str(e)})

# Get recent sensor data
@app.route("/get_data/Mode=<db_name>/Count=<count>")
def get_data(db_name, count):
    db = DB_SENSOR if db_name == "sensor_data" else DB_RANDOM
    try:
        conn = sqlite3.connect(db)
        c = conn.cursor()
        c.execute(f'SELECT temperature, humidity, timestamp FROM {db_name}
ORDER BY id DESC LIMIT {count}')
        data = c.fetchall()
        conn.close()
        # Format the data into JSON format
        formatted_data = [{"temperature": row[0], "humidity": row[1],
"timestamp": row[2]} for row in data]
```

```
        return jsonify(formatted_data)
    except sqlite3.Error as e:
        print(f"Error retrieving data: {e}")
        return jsonify({"error": str(e)})

@app.route("/get_data_by_date/Mode=<db_name>/Date=<date>")
def get_data_by_date(db_name, date):
    db = DB_SENSOR if db_name == "sensor_data" else DB_RANDOM
    try:
        conn = sqlite3.connect(db)
        c = conn.cursor()
        c.execute(f'SELECT temperature, humidity, timestamp FROM {db_name}
WHERE DATE(timestamp) = ? ORDER BY timestamp', (date,))
        data = c.fetchall()
        conn.close()

        if not data:
            return jsonify({"error": "No data available for the selected
date."})

        # Calculate average temperature and humidity
        temperature_sum = sum(row[0] for row in data)
        humidity_sum = sum(row[1] for row in data)
        average_temperature = temperature_sum / len(data)
        average_humidity = humidity_sum / len(data)

        # Format the data into JSON format
        formatted_data = [{"temperature": row[0], "humidity": row[1],
"timestamp": row[2]} for row in data]
        return jsonify({"data": formatted_data, "average_temperature":
average_temperature, "average_humidity": average_humidity})
    except sqlite3.Error as e:
        print(f"Error retrieving data by date: {e}")
        return jsonify({"error": str(e)})

if __name__ == "__main__":
    # Initialize SQLite databases
    init_databases()
    # Start generating random sensor data
    random_data_thread = threading.Thread(target=generate_random_data)
    random_data_thread.daemon = True
    random_data_thread.start()
    # Run the application
    app.run(host="0.0.0.0", port=5000, debug=True)
```

前端(index.html、style.css)

- 直接更新HW3內容
- 增加能自由選擇獲取前幾筆資料的輸入框與js function

- 增加能選擇獲取哪天的所有資料的折線圖、日期選擇器與js function
- 增加了網頁說明

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sensor Data Visualization</title>
    <!-- 引入 Highcharts 库 -->
    <script src="https://code.highcharts.com/highcharts.js"></script>
    <script src="https://code.highcharts.com/modules/exporting.js">
</script>
    <script src="https://code.highcharts.com/modules/export-data.js">
</script>
    <script src="https://code.highcharts.com/modules/accessibility.js">
</script>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
    <link rel="stylesheet" href="../static/style.css">
</head>
<body>
    <div class="main">
        <!-- 折線圖容器 -->
        <div class="chart" id="sensor-chart"></div>

        <!-- 控制面板 -->
        <div class="container">
            <!-- 切換顯示模式的下拉選單 -->
            <div class="control-container">
                <label for="mode-select">顯示模式：</label>
                <select id="mode-select" onchange="toggleMode()">
                    <option value="sensor_data" selected>Sensor
Data</option>
                    <option value="random_data">Random Data</option>
                </select>
            </div>

            <!-- 輸入框，用於設置更新頻率 -->
            <div class="control-container">
                <label for="update-frequency">更新頻率(秒)：</label>
                <input type="number" id="update-frequency" min="1"
value="2" onchange="setUpdateFrequency()">
            </div>

            <div class="control-container">
```

```
                <label for="display-count">顯示筆數(5-30)：</label>
                <input type="number" id="display-count" min="5" max="30"
value="30" onchange="setDisplayCount()">
            </div>
        </div>
    </div>

    <div class="main">
        <div class="chart" id="data-by-date" ></div>
        <div class="container">
            <div class="control-container">
                <label for="date-input">查詢日期：</label>
                <input type="date" id="date-input"
onchange="getDataByDate()">
            </div>
        </div>
    </div>

    <div class="tooltip" id="infoButton">?
        <span class="tooltiptext">Click to view webpage
instructions</span>
    </div>

    <script>
        var currentMode = "sensor_data"; // 初始顯示模式
        var updateInterval = 2000; // 初始更新頻率（毫秒）
        var currentDisplayCount = 30;

        // 切換顯示模式
        function toggleMode() {
            currentMode = document.getElementById("mode-select").value;
        }

        // 切換顯示筆數
        function setDisplayCount() {
            currentDisplayCount = document.getElementById("display-
count").value;
        }

        // 設置更新頻率
        function setUpdateFrequency() {
            updateInterval = parseInt(document.getElementById("update-
frequency").value) * 1000; // 轉換為毫秒
        }

        // 函數來更新圖表數據
        function updateChartData() {
            // 使用 jQuery 發送 GET 請求獲取數據
```

```
$.get(`/get_data/Mode=${currentMode}/Count=${currentDisplayCount}`,
function(data) {

                // 解析 JSON 數據
                var temperatureData = [];
                var humidityData = [];
                for (var i = 0; i < data.length; i++) {
                    // x 座標為 1 到 30
                    temperatureData.push({
                        x: i + 1,
                        y: data[i].temperature
                    });
                    humidityData.push({
                        x: i + 1,
                        y: data[i].humidity
                    });
                }

                // 更新圖表數據
                chart.series[0].setData(temperatureData);
                chart.series[1].setData(humidityData);
            });
        }

        function getToday() {
            const today = new Date();
            const year = today.getFullYear();
            const month = String(today.getMonth() + 1).padStart(2, '0');
            const day = String(today.getDate()).padStart(2, '0');
            return `${year}-${month}-${day}`;
        }

        function getDataByDate() {
            var selectedDate = document.getElementById("date-
input").value;
            var mode = document.getElementById("mode-select").value; //
Get current mode
            var url =
`/get_data_by_date/Mode=sensor_data/Date=${selectedDate}`;

            fetch(url)
                .then(response => response.json())
                .then(data => {
                    var resultDiv = document.getElementById("data-by-
date");
                    resultDiv.innerHTML = ""; // Clear previous results
                    if (data.error) {
                        resultDiv.innerText = data.error;
                    } else {
```

```
                        var dataList = data.data;
                        var temperatureData = [];
                        var humidityData = [];
                        console.log(dataList);
                        dataList.forEach(item => {
                            temperatureData.push([new
Date(item.timestamp).getTime(), item.temperature]);
                            humidityData.push([new
Date(item.timestamp).getTime(), item.humidity]);
                        });

                        Highcharts.chart('data-by-date', {
                            title: {
                                text: 'Temperature and Humidity Data for a
Specific Day(Only sensor)'
                            },
                            xAxis: {
                                type: 'datetime',
                                title: {
                                    text: 'Time'
                                }
                            },
                            yAxis: [{
                                title: {
                                    text: 'Temperature (°C)'
                                }
                            }, {
                                title: {
                                    text: 'Humidity (%)'
                                },
                                opposite: true
                            }],
                            series: [{
                                name: 'Temperature',
                                data: temperatureData
                            }, {
                                name: 'Humidity',
                                data: humidityData,
                                yAxis: 1
                            }]
                        });
                    }
                })
                .catch(error => console.error("Error fetching data:",
error));
        }
```

```
        // 初始化折線圖
        var chart = Highcharts.chart('sensor-chart', {
            title: {
                text: 'Latest Temperature and Humidity Data'
            },
            xAxis: {
                categories: Array.from({length: 30}, (_, i) =>
(i).toString())
            },
            yAxis: [{
                title: {
                    text: 'Temperature (°C)'
                },
                opposite: false,
                min: 0,
                max: 40
            }, {
                title: {
                    text: 'Humidity (%)'
                },
                opposite: true,
                min: 10,
                max: 80
            }],
            series: [{
                name: 'Temperature',
                data: []
            }, {
                name: 'Humidity',
                data: [],
                yAxis: 1
            }]
        });

        function showInfo() {
            var message = "This webpage visualizes sensor data including
temperature and humidity. You can switch between sensor data and random
data using the dropdown menu. You can also customize the update frequency
and the number of data points to display. Use the date input to query data
for a specific date.";
            alert(message);
        }

        // 每隔一段時間更新一次數據
        setInterval(updateChartData, updateInterval);
        document.getElementById("date-input").value = getToday();
        getDataByDate();
        document.getElementById("infoButton").addEventListener("click",
showInfo);
```

```
    </script>
</body>
</html>
```

CSS

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
    height: 100vh;
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 10px;
}

.main {
    font-family: Arial, sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 20px;
}

.chart {
    width: 800px;
    height: 400px;
    display: flex;
    justify-content: center;
    align-items: center;
    box-shadow: rgba(0, 0, 0, 0.24) 0px 3px 8px;
    background-color: #fff;
    border-radius: 5px;
}

.tooltip {
    position: fixed;
    bottom: 20px;
    right: 20px;
    width: 40px;
    height: 40px;
    background-color: #fff;
    box-shadow: rgba(0, 0, 0, 0.24) 0px 3px 8px;
    border-radius: 100%;
    display: flex;
    justify-content: center;
    align-items: center;
```

```
  }

  .tooltip .tooltiptext {
    visibility: hidden;
    width: 270px;
    background-color: black;
    color: #fff;
    text-align: center;
    border-radius: 6px;
    padding: 5px 0;
    font-family: Arial, sans-serif;
    /* Position the tooltip */
    position: absolute;
    z-index: 1;
    top: 5px;
    right: 120%;
  }

  .tooltip:hover .tooltiptext {
    visibility: visible;
  }

.container {
    margin: 20px;
}

.control-container {
    margin-bottom: 10px;
    display: flex;
    justify-content: space-between;
    width: 320px;
}

label {
    font-weight: bold;
}

input[type="number"],
input[type="date"],
select {
    padding: 8px;
    border-radius: 5px;
    border: 1px solid #ccc;
    font-size: 16px;
    width: 150px;
}

input[type="number"]:focus,
input[type="date"]:focus,
```

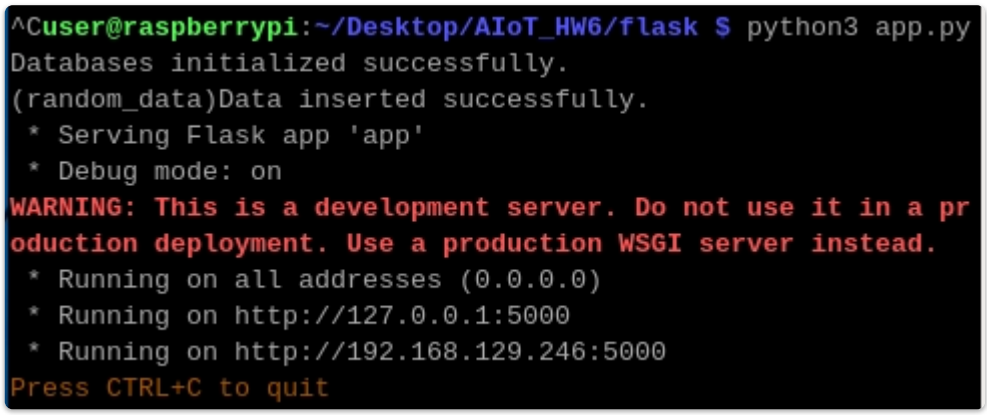
```
select:focus {
    outline: none;
    border-color: #007bff;
    box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);
}

#mode-select {
    width: 167px;
}

#sensor-chart {
    background-color: #fff;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

#mode-text {
    font-weight: bold;
    margin-left: 5px;
}
```

執行

```
python3 app.py
```

執行結果

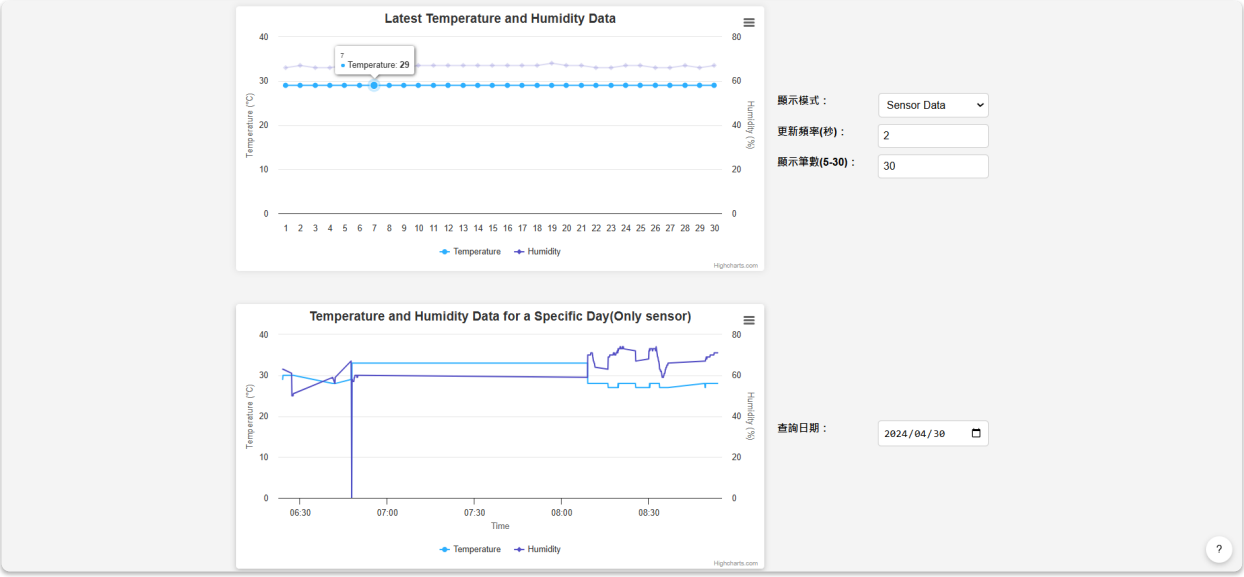
連接伺服器

- 
```
^Cuser@raspberrypi:~/Desktop/AIoT_HW6/flask $ python3 app.py
Databases initialized successfully.
(random_data)Data inserted successfully.
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a pr
oduction deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.129.246:5000
Press CTRL+C to quit
```
    - 開啟伺服器

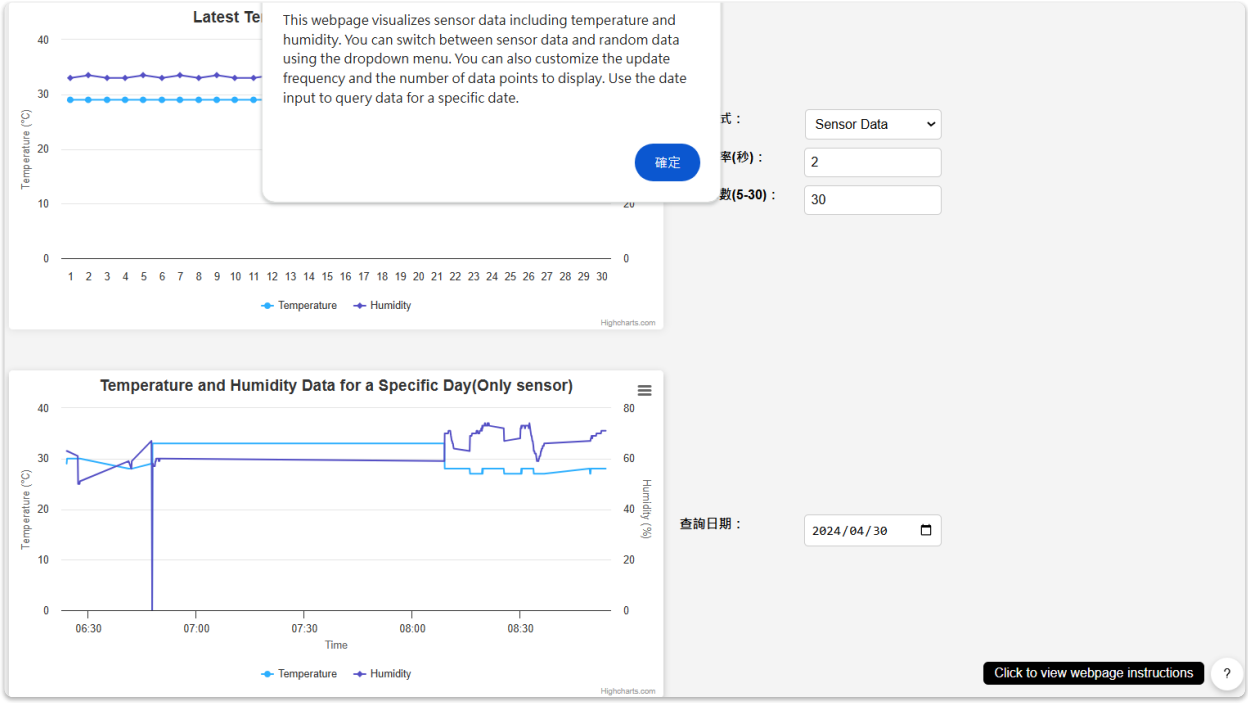
```
ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1184
load:0x40078000,len:13260
load:0x40080400,len:3028
entry 0x400805e4
Connecting to WiFi...
Connecting to WiFi...
Connecting to WiFi...
Connected to WiFi
192.168.129.115
Temperature: 29 °C      Humidity: 67 %
Server Address: http://192.168.129.57:5000/post_data
HTTP Response code: 200
{
  "success": true
}
```

- 伺服器已開啟，可以看到資料傳送成功

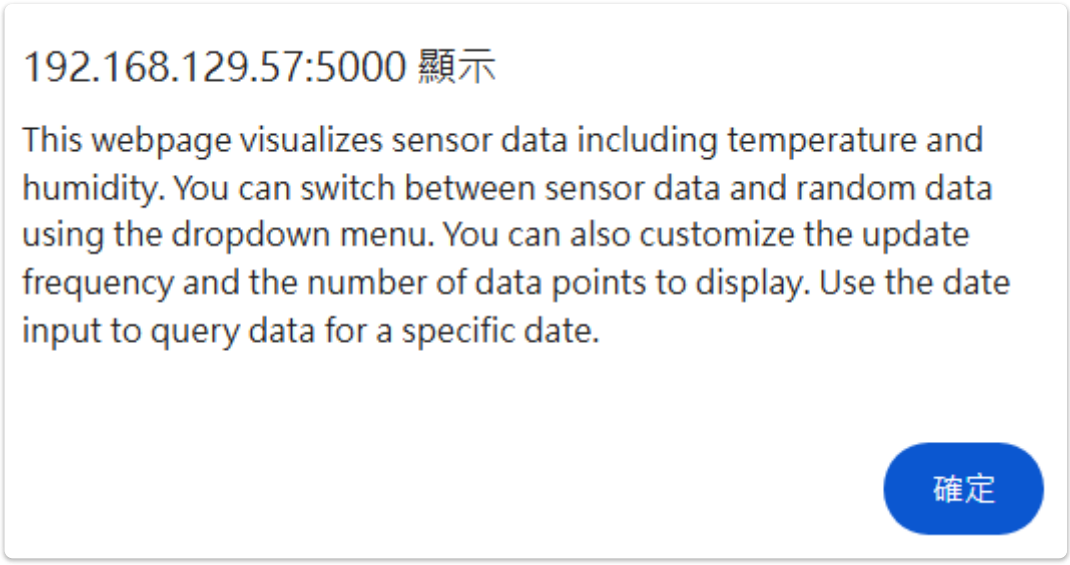
網頁渲染



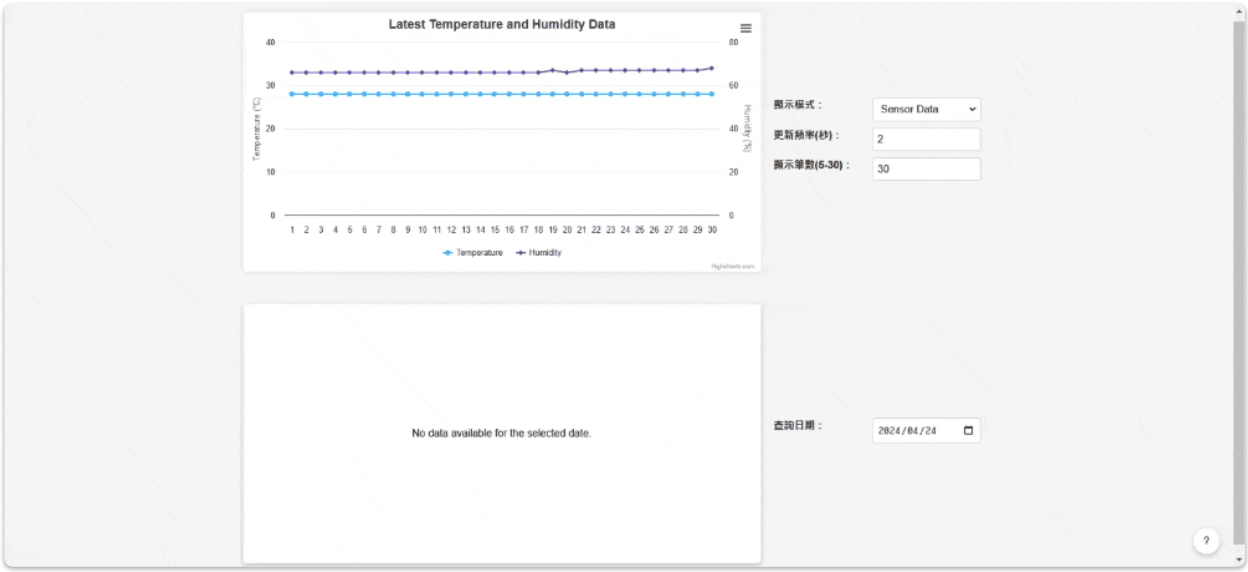
- 上方折線圖為資料庫內最新幾筆的數據
  - 可設定之選項
    - 模式: 隨機生成之資料、溫溼度感測器取得之資料
    - 更新秒數
    - 顯示筆數
- 下方折線圖可選擇某天之所有資料，此處只顯示感測器取得之資料
  - 若該天無資料，則顯示No data available for the selected date
  - 顯示時發現傳遞錯誤時會有溫溼度為0的情況，已修改app.py，使資料存於資料庫之前先檢查值是否合理。



- 增加網頁說明，讓使用者可了解此網頁用途



- 網頁說明示意圖



- 網頁操作GIF圖

6-4 附件

- GitHub: AIoT_HW6
- 參考資料: 3311_Lab6_student