

# 下載資料顯示流程

## Part 1: 中央氣象局資料

Link used in class

1. 範例網頁 web

[https://www.cwa.gov.tw/V8/C/W/OBS\\_Temp.html](https://www.cwa.gov.tw/V8/C/W/OBS_Temp.html)

2. cwa login <https://opendata.cwa.gov.tw/userLogin>

3. API : CWA-1FFDDAEC-161F-46A3-BE71-93C32C52829F

4. 範例 data source in

JSON:<https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001>

link

<https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=CWA-1FFDDAEC-161F-46A3-BE71-93C32C52829F&downloadType=WEB&format=JSON>

5. requirement 將下載資料存成sqlite3 data.db and show on streamlit locally (截圖)

下面整理 完整可用的流程 ( 含 Python 程式碼 ) · 協助你做到 :

- ✓ 從中央氣象局 CWA 下載 JSON
- ✓ 存入 SQLite3 ( data.db )
- ✓ 用 Streamlit 顯示資料 ( 可截圖 )

你只需複製貼上即可跑在本機。

## ✓ Part 1 : 下載中央氣象局開放資料 → SQLite3 → Streamlit 顯示

### 1. 下載 JSON 資料 ( CWA )

你提供的 API URL :

```
pgsql
```

```
https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?
```

```
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
```

```
93C32C52829F&downloadType=WEB&format=JSON
```

## 2. Python：下載資料並寫入 SQLite data.db

請建立一支 `fetch_to_db.py`：

```
python

import requests
import json
import sqlite3

# CWA URL
URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
93C32C52829F&downloadType=WEB&format=JSON"

def download_data():
    r = requests.get(URL)
    data = r.json()
    return data

def save_to_db(data, db_name="data.db"):
    conn = sqlite3.connect(db_name)
    cursor = conn.cursor()

    # 建表
    cursor.execute("""
        CREATE TABLE IF NOT EXISTS forecast (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            location TEXT,
            wx TEXT,
            max_temp TEXT,
            min_temp TEXT
        )
    """)

    # 清空舊資料
    cursor.execute("DELETE FROM forecast")

    # 取得資料區段
    locations = data["cwaopendata"]["dataset"]["location"]

    for loc in locations:
        name = loc["locationName"]
```

```

        wx = loc["weatherElement"][0]["time"][0]["parameter"]
        ["parameterName"]
        max_temp = loc["weatherElement"][1]["time"][0]["parameter"]
        ["parameterName"]
        min_temp = loc["weatherElement"][2]["time"][0]["parameter"]
        ["parameterName"]

        cursor.execute("""
            INSERT INTO forecast (location, wx, max_temp, min_temp)
            VALUES (?, ?, ?, ?)
        """, (name, wx, max_temp, min_temp))

        conn.commit()
        conn.close()
        print("資料已成功寫入 data.db")

if __name__ == "__main__":
    data = download_data()
    save_to_db(data)

```

### 3. Streamlit 顯示 SQLite3 資料

建立 `app.py` :

```

python

import sqlite3
import pandas as pd
import streamlit as st

st.title("中央氣象局天氣預報資料顯示")

# 連線 SQLite
conn = sqlite3.connect("data.db")
df = pd.read_sql_query("SELECT * FROM forecast", conn)

st.subheader("資料表：forecast")
st.dataframe(df)

```

### 4. 執行流程

## (1) 下載並寫入 SQLite

```
bash

python fetch_to_db.py
```

成功後會出現：

```
kotlin

資料已成功寫入 data.db
```

並在資料夾看到：

```
kotlin

data.db
fetch_to_db.py
app.py
```

---

## (2) 啟動 Streamlit

```
bash

streamlit run app.py
```

會打開瀏覽器，顯示：

- ✓ 地點
- ✓ 天氣現象 ( wx )
- ✓ 最高溫
- ✓ 最低溫

然後你即可「截圖」提交。

---

若你想要我把這些程式整合成單一頁面、或加入圖表、cards 美化 Streamlit，也可以告訴我。

```
(base) PS C:\Users\raced\hw> python fetch_to_db.py
> >
Traceback (most recent call last):
  File "C:\Users\raced\hw\fetch_to_db.py", line 52, in
<module>
    save_to_db(data)
  File "C:\Users\raced\hw\fetch_to_db.py", line 32, in
save_to_db
    locations = data["cwaopendata"]["dataset"]
["location"]

~~~~~^
KeyError: 'dataset'
```

你遇到的錯誤：

```
vbnet
```

```
KeyError: 'dataset'
```

代表：

中央氣象局 API 回傳格式不是你預期的結構 → 很可能 API key 有誤 / 格式錯誤 / CORS / JSON 結構變動。

讓我們一步步修好。

## ✅ Step 1 — 先印出整份 JSON 看原始結構

請修改 `fetch_to_db.py` 前半段，加上：

```
python

print(json.dumps(data, indent=2, ensure_ascii=False))
```

📌 完整可貼上：

```
python

import requests
import json
import sqlite3

URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?"
```

```
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
93C32C52829F&downloadType=WEB&format=JSON"

def download_data():
    r = requests.get(URL)
    data = r.json()

    # ★ 印出完整 JSON 檢查結構
    print("\n=== API 回傳資料 ===")
    print(json.dumps(data, indent=2, ensure_ascii=False))

    return data

def save_to_db(data, db_name="data.db"):
    # 以下暫時註解掉避免錯誤
    # locations = data["cwaopendata"]["dataset"]["location"]
    pass

if __name__ == "__main__":
    data = download_data()
```

👉 請你執行：

```
nginx
```

```
python fetch_to_db.py
```

然後把執行結果 (JSON 前 20 行即可) 貼給我。  
我會依照你實際的 JSON 結構寫出正確的解析程式。

## 📌 常見原因 (給你先參考)

CWA JSON 結構通常如下：

### 1. 成功格式

```
json

{
  "cwaopendata": {
    "dataset": {
```

```
        "location": [...]  
    }  
}  
}
```

## 2. API 錯誤格式

```
json  
  
{  
    "success": "false",  
    "message": "Authorization failed"  
}
```

或：

```
json  
  
{  
    "cwaopendata": {  
        "language": "zh-TW",  
        "datasetDescription": "xxx",  
        "resource": []  
    }  
}
```

若你貼出完整 JSON，我會立即把完整可執行版本改好給你。



請貼上輸出的 JSON，我幫你修正解析程式。

```
(base) PS C:\Users\raced\hw> python fetch_to_db.py  
>>  
Traceback (most recent call last):  
  File "C:\Users\raced\hw\fetch_to_db.py", line 52, in  
<module>  
    save_to_db(data)  
  File "C:\Users\raced\hw\fetch_to_db.py", line 32, in  
    save_to_db  
    locations = data["cwaopendata"]["dataset"]  
                  ["location"]
```

```
~~~~~^ ^ ^ ^ ^ ^ ^ ^ ^ ^  
KeyError: 'dataset'
```

我需要看到 **API 回傳的 JSON 真正長什麼樣子** 才能修正解析，但你目前貼出的錯誤顯示你還沒有加入 **print JSON 的程式碼**，因此無法看到內容。

## ✅ 請你一定要先把這段程式碼加進去

在 `download_data()` 裡面加入：

```
python  
  
print(json.dumps(data, indent=2, ensure_ascii=False))
```

並且 **先不要呼叫** `save_to_db(data)`，避免 `KeyError`。

## 🔧 完整可直接貼上替換的版本（暫時專門用來 debug）

請把你的 `fetch_to_db.py` 替換成下面這份：

```
python  
  
import requests  
import json  
  
URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?  
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-  
93C32C52829F&downloadType=WEB&format=JSON"  
  
def download_data():  
    r = requests.get(URL)  
  
    try:  
        data = r.json()  
    except:  
        print("❌ 無法解析 JSON")  
        print(r.text)  
        return None  
  
    print("\n=== API 回傳資料 (前 200 行) ===\n")  
    print(json.dumps(data, indent=2, ensure_ascii=False)[:5000]) # 印前
```



5000 字

```
    return data

if __name__ == "__main__":
    download_data() # 只下載，不存 DB
```

## 請執行：

```
nginx

python fetch_to_db.py
```

然後 把輸出的 JSON 前 30 行貼給我。

( 不要擔心隱私，這是公開 API )

## 原因說明

你的錯誤：

```
vbnet

KeyError: 'dataset'
```

表示：

```
CSS

data["cwaopendata"]["dataset"]
```

裡沒有 `"dataset"` 這個欄位。

原因可能是：

- ✓ API Key 錯誤
- ✓ API 回傳錯誤訊息 ( 但格式不同 )
- ✓ dataset 名稱不一樣 ( 例如 `records` 、 `resources` )

但如果看不到回傳內容，我無法寫出正確解析方式。



請貼上印出的 JSON，我會立刻修正你的程式碼。

```
{
  "cwaopendata": {
    "@xmlns": "urn:cwa:gov:tw:cwacommon:0.1",
    "identifier": "472c7047-eb48-4aa1-8fc0-9be46a234889",
    "datasetName": "一週農業氣象預報",
    "dataid": "F-A0010-001",
    "sender": "shchiang@cwa.gov.tw",
    "sent": "2025-12-03T18:00:06+08:00",
    "status": "Actual",
    "scope": "Public",
    "msgType": "Issue",
    "resources": {
      "resource": {
        "metadata": {
          "resourceName": "中央氣象署氣候服務_農業氣象一週預報",
          "resourceDescription": "提供氣象署預報資料和農事建議，供農業相關單位及農民參考",
          "language": "zh",
          "homepage": "https://www.cwa.gov.tw/V8/C/L/agriculture.html",
          "sources": {
            "source": [
              {
                "title": "1週預報",
                "path": "https://www.cwa.gov.tw/V8/C/W/week.html"
              },
              {
                "title": "長期天氣預報-季長期天氣展望",
                "path": "https://www.cwa.gov.tw/Data/fcst_pdf/FW15.pdf"
              }
            ]
          }
        }
      }
    },
    "temporal": {
      "issueTime": "2025-12-03T17:00:00+08:00",
      "validTime": {
```

```

"startTime": "2025-12-04T00:00:00+08:00",
"endTime": "2025-12-11T00:00:00+08:00"
}
},
"weatherElements": {
"weatherElement": [
{
"tagName": "weather",
"description": "天氣描述"
},
{
"tagName": "temperature",
"description": "氣溫",
"units": "°C"
},
{
"tagName": "agrWeatherForecasts",
"description": "一週農業天氣預報"
},
{
"tagName": "weatherProfile",
"description": "天氣概況"
},
{
"tagName": "weatherForecasts",
"description": "天氣預報"
},
{
"tagName": "agrAdvices",
"description": "農事建議"
},
{
"tagName": "degreeDay",
"description": "度日(植物發育時所需之有效溫度單位)",
"units": "GDD"
},
{
"tagName": "accumulatedTemperature",
"description": "積溫(植物發育過程中，當各生育階段有效溫度累積至一定數值時，即可完成發育。該數值即稱為此一發育階段所需的積溫。)"
},
{

```

```

"tagName": "cardinalTemperatures",
"description": "作物生長溫度區間",
"units": "°C",
"statisticalMethods": {
  "statisticalMethod": [
    {
      "methodTagName": "minimum",
      "description": "生長最低溫"
    },
    {
      "methodTagName": "maximum",
      "description": "生長最高溫"
    },
    {
      "methodTagName": "optimum",
      "description": "生長最適溫"
    }
  ]
},
{
  "tagName": "growingDays",
  "description": "生育日數"
},
{
  "tagName": "cropStatistics",
  "description": "農業積溫統計"
}
]
}
},
"data": {
  "agrWeatherForecasts": {
    "weatherProfile": "4日、5日東北季風影響,北部、東北部及東部天氣較涼,其他地區早晚亦涼;4日基隆北海岸及東北部地區有局部短暫雨,東部、東南部地區、恆春半島及大臺北、中南部山區有零星短暫雨,其他地區為多雲到晴;5日基隆北海岸、東北部地區及大臺北山區有局部短暫雨,桃園以北、東部、東南部地區及恆春半島有零星短暫雨,其他地區為多雲到晴。6日、7日東北季風稍減弱,各地氣溫逐漸回升,早晚仍較涼,中南部日夜溫差大;各地大多為晴到多雲,僅基隆北海岸、大臺北山區、東半部地區及恆春半島有零星短暫雨。8日東北季風增強,9日東北季風影響,北部及東北部天氣稍轉涼,其他地區早晚亦涼;桃園以北、東半部地區及恆春半

```

島有局部短暫雨,其他地區為多雲到晴。10日東北季風減弱,各地氣溫回升;各地大多為多雲到晴,僅基隆北海岸、大臺北山區、東半部地區及恆春半島有零星短暫雨。",

```
"weatherForecasts": {
  "location": [
    {
      "locationName": "北部地區",
      "weatherElements": {
        "Wx": {
          "daily": [
            {
              "dataDate": "2025-12-04",
              "weather": "多雲時晴",
              "weatherid": "3"
            },
            {
              "dataDate": "2025-12-05",
              "weather": "多雲短暫雨",
              "weatherid": "8"
            },
            {
              "dataDate": "2025-12-06",
              "weather": "多雲時晴",
              "weatherid": "3"
            },
            {
              "dataDate": "2025-12-07",
              "weather": "晴時多雲",
              "weatherid": "2"
            },
            {
              "dataDate": "2025-12-08",
              "weather": "多雲短暫雨",
              "weatherid": "8"
            },
            {
              "dataDate": "2025-12-09",
              "weather": "多雲短暫雨",
              "weatherid": "8"
            },
            {
              "dataDate": "2025-12-10",
              "weather": "晴時多雲",
              "weatherid": "2"
            }
          ]
        }
      }
    }
  ]
}
```

```
}  
]  
,  
"MaxT": {  
  "daily": [  
    {  
      "dataDate": "2025-12-04",  
      "temperature": "21"  
    },  
    {  
      "dataDate": "2025-12-05",  
      "temperature": "23"  
    },  
    {  
      "dataDate": "2025-12-06",  
      "temperature": "24"  
    },  
    {  
      "dataDate": "2025-12-07",  
      "temperature": "26"  
    },  
    {  
      "dataDate": "2025-12-08",  
      "temperature": "23"  
    },  
    {  
      "dataDate": "2025-12-09",  
      "temperature": "23"  
    },  
    {  
      "dataDate": "2025-12-10",  
      "temperature": "26"  
    }  
  ]  
,  
  "MinT": {  
    "daily": [  
      {  
        "dataDate": "2025-12-04",  
        "temperature": "15"  
      },  
      {  
        "dataDate": "2025-12-05",  
        "temperature": "15"
```

```
    },  
    {  
      "dataDate": "2025-12-06",  
      "temperature": "15"  
    },  
    {  
      "dataDate": "2025-12-07",  
      "temperature": "15"  
    },  
    {  
      "dataDate": "2025-12-08",  
      "temperature": "16"  
    },  
    {  
      "dataDate": "2025-12-09",  
      "temperature": "17"  
    },  
    {  
      "dataDate": "2025-12-10",  
      "temperature": "16"  
    }  
  ]  
}  
},  
{  
  "locationName": "中部地區",  
  "weatherElements": {  
    "Wx": {  
      "daily": [  
        {  
          "dataDate": "2025-12-04",  
          "weather": "晴時多雲",  
          "weatherid": "2"  
        },  
        {  
          "dataDate": "2025-12-05",  
          "weather": "晴時多雲",  
          "weatherid": "2"  
        },  
        {  
          "dataDate": "2025-12-06",  
          "weather": "晴時多雲",  
          "weatherid": "2"
```

```
},
{
  "dataDate": "2025-12-07",
  "weather": "晴時多雲",
  "weatherid": "2"
},
{
  "dataDate": "2025-12-08",
  "weather": "晴時多雲",
  "weatherid": "2"
},
{
  "dataDate": "2025-12-09",
  "weather": "晴時多雲",
  "weatherid": "2"
},
{
  "dataDate": "2025-12-10",
  "weather": "晴時多雲",
  "weatherid": "2"
}
]
},
"MaxT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "25"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "26"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "26"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "27"
    },
    {
      "dataDate": "2025-12-08",
      "temperature": "26"
    }
  ]
}
```



```
},
{
  "dataDate": "2025-12-09",
  "temperature": "26"
},
{
  "dataDate": "2025-12-10",
  "temperature": "28"
}
]
},
"MinT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "15"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "16"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "16"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "16"
    },
    {
      "dataDate": "2025-12-08",
      "temperature": "16"
    },
    {
      "dataDate": "2025-12-09",
      "temperature": "17"
    },
    {
      "dataDate": "2025-12-10",
      "temperature": "17"
    }
  ]
}
}
```

```
},
{
  "locationName": "南部地區",
  "weatherElements": {
    "Wx": {
      "daily": [
        {
          "dataDate": "2025-12-04",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-05",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-06",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-07",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-08",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-09",
          "weather": "晴時多雲",
          "weatherid": "2"
        },
        {
          "dataDate": "2025-12-10",
          "weather": "晴時多雲",
          "weatherid": "2"
        }
      ]
    },
    "MaxT": {
```

```
"daily": [  
  {  
    "dataDate": "2025-12-04",  
    "temperature": "25"  
  },  
  {  
    "dataDate": "2025-12-05",  
    "temperature": "26"  
  },  
  {  
    "dataDate": "2025-12-06",  
    "temperature": "27"  
  },  
  {  
    "dataDate": "2025-12-07",  
    "temperature": "28"  
  },  
  {  
    "dataDate": "2025-12-08",  
    "temperature": "27"  
  },  
  {  
    "dataDate": "2025-12-09",  
    "temperature": "27"  
  },  
  {  
    "dataDate": "2025-12-10",  
    "temperature": "29"  
  }  
],  
"MinT": {  
  "daily": [  
    {  
      "dataDate": "2025-12-04",  
      "temperature": "16"  
    },  
    {  
      "dataDate": "2025-12-05",  
      "temperature": "18"  
    },  
    {  
      "dataDate": "2025-12-06",  
      "temperature": "18"
```

```
},
{
  "dataDate": "2025-12-07",
  "temperature": "18"
},
{
  "dataDate": "2025-12-08",
  "temperature": "19"
},
{
  "dataDate": "2025-12-09",
  "temperature": "19"
},
{
  "dataDate": "2025-12-10",
  "temperature": "19"
}
]
}
}
},
{
  "locationName": "東北部地區",
  "weatherElements": {
    "Wx": {
      "daily": [
        {
          "dataDate": "2025-12-04",
          "weather": "陰短暫雨",
          "weatherid": "11"
        },
        {
          "dataDate": "2025-12-05",
          "weather": "陰短暫雨",
          "weatherid": "11"
        },
        {
          "dataDate": "2025-12-06",
          "weather": "陰時多雲短暫雨",
          "weatherid": "10"
        },
        {
          "dataDate": "2025-12-07",
          "weather": "多雲短暫雨",
```

```
"weatherid": "8"
},
{
  "dataDate": "2025-12-08",
  "weather": "陰短暫雨",
  "weatherid": "11"
},
{
  "dataDate": "2025-12-09",
  "weather": "多雲時陰短暫雨",
  "weatherid": "9"
},
{
  "dataDate": "2025-12-10",
  "weather": "多雲短暫雨",
  "weatherid": "8"
}
]
},
"MaxT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "19"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "20"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "22"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "25"
    },
    {
      "dataDate": "2025-12-08",
      "temperature": "21"
    },
    {
      "dataDate": "2025-12-09",
      "temperature": "22"
```

```
    },  
    {  
      "dataDate": "2025-12-10",  
      "temperature": "24"  
    }  
  ]  
},  
"MinT": {  
  "daily": [  
    {  
      "dataDate": "2025-12-04",  
      "temperature": "16"  
    },  
    {  
      "dataDate": "2025-12-05",  
      "temperature": "17"  
    },  
    {  
      "dataDate": "2025-12-06",  
      "temperature": "17"  
    },  
    {  
      "dataDate": "2025-12-07",  
      "temperature": "18"  
    },  
    {  
      "dataDate": "2025-12-08",  
      "temperature": "18"  
    },  
    {  
      "dataDate": "2025-12-09",  
      "temperature": "18"  
    },  
    {  
      "dataDate": "2025-12-10",  
      "temperature": "18"  
    }  
  ]  
}  
}  
},  
{  
  "locationName": "東部地區",  
  "weatherElements": {
```

```
"Wx": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "weather": "陰時多雲",
      "weatherid": "6"
    },
    {
      "dataDate": "2025-12-05",
      "weather": "陰時多雲短暫雨",
      "weatherid": "10"
    },
    {
      "dataDate": "2025-12-06",
      "weather": "多雲時陰",
      "weatherid": "5"
    },
    {
      "dataDate": "2025-12-07",
      "weather": "多雲時晴",
      "weatherid": "3"
    },
    {
      "dataDate": "2025-12-08",
      "weather": "多雲時陰短暫雨",
      "weatherid": "9"
    },
    {
      "dataDate": "2025-12-09",
      "weather": "多雲時陰短暫雨",
      "weatherid": "9"
    },
    {
      "dataDate": "2025-12-10",
      "weather": "多雲時晴",
      "weatherid": "3"
    }
  ]
},
"MaxT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "20"
```

```
},
{
  "dataDate": "2025-12-05",
  "temperature": "21"
},
{
  "dataDate": "2025-12-06",
  "temperature": "23"
},
{
  "dataDate": "2025-12-07",
  "temperature": "25"
},
{
  "dataDate": "2025-12-08",
  "temperature": "24"
},
{
  "dataDate": "2025-12-09",
  "temperature": "23"
},
{
  "dataDate": "2025-12-10",
  "temperature": "24"
}
]
},
"MinT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "17"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "19"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "19"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "19"
    }
  ]
}
```



```
    },  
    {  
      "dataDate": "2025-12-08",  
      "temperature": "20"  
    },  
    {  
      "dataDate": "2025-12-09",  
      "temperature": "20"  
    },  
    {  
      "dataDate": "2025-12-10",  
      "temperature": "20"  
    }  
  ]  
}  
},  
{  
  "locationName": "東南部地區",  
  "weatherElements": {  
    "Wx": {  
      "daily": [  
        {  
          "dataDate": "2025-12-04",  
          "weather": "多雲時陰",  
          "weatherid": "5"  
        },  
        {  
          "dataDate": "2025-12-05",  
          "weather": "多雲時晴",  
          "weatherid": "3"  
        },  
        {  
          "dataDate": "2025-12-06",  
          "weather": "多雲",  
          "weatherid": "4"  
        },  
        {  
          "dataDate": "2025-12-07",  
          "weather": "多雲時晴",  
          "weatherid": "3"  
        },  
        {  
          "dataDate": "2025-12-08",
```

```
"weather": "多雲",
"weatherid": "4"
},
{
  "dataDate": "2025-12-09",
  "weather": "多雲",
  "weatherid": "4"
},
{
  "dataDate": "2025-12-10",
  "weather": "晴時多雲",
  "weatherid": "2"
}
]
},
"MaxT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "22"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "23"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "24"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "26"
    },
    {
      "dataDate": "2025-12-08",
      "temperature": "26"
    },
    {
      "dataDate": "2025-12-09",
      "temperature": "24"
    },
    {
      "dataDate": "2025-12-10",
      "temperature": "26"
    }
  ]
}
```

```
}
]
},
"MinT": {
  "daily": [
    {
      "dataDate": "2025-12-04",
      "temperature": "18"
    },
    {
      "dataDate": "2025-12-05",
      "temperature": "19"
    },
    {
      "dataDate": "2025-12-06",
      "temperature": "19"
    },
    {
      "dataDate": "2025-12-07",
      "temperature": "20"
    },
    {
      "dataDate": "2025-12-08",
      "temperature": "20"
    },
    {
      "dataDate": "2025-12-09",
      "temperature": "20"
    },
    {
      "dataDate": "2025-12-10",
      "temperature": "20"
    }
  ]
}
}
}
]
},
"agrAdvices": {
  "agrForecasts": {
    "location": [
      {
        "locationName": "北部地區",
```

```
"weatherElements": {  
  "daily": [  
    {  
      "dataDate": "2025-12-04",  
      "degreeDay": "8.0",  
      "accumulatedTemperature": "8.0"  
    },  
    {  
      "dataDate": "2025-12-05",  
      "degreeDay": "9.0",  
      "accumulatedTemperature": "17.0"  
    },  
    {  
      "dataDate": "2025-12-06",  
      "degreeDay": "9.5",  
      "accumulatedTemperature": "26.5"  
    },  
    {  
      "dataDate": "2025-12-07",  
      "degreeDay": "10.5",  
      "accumulatedTemperature": "37.0"  
    },  
    {  
      "dataDate": "2025-12-08",  
      "degreeDay": "9.5",  
      "accumulatedTemperature": "46.5"  
    },  
    {  
      "dataDate": "2025-12-09",  
      "degreeDay": "10.0",  
      "accumulatedTemperature": "56.5"  
    },  
    {  
      "dataDate": "2025-12-10",  
      "degreeDay": "11.0",  
      "accumulatedTemperature": "67.5"  
    }  
  ]  
}  
,  
{  
  "locationName": "中部地區",  
  "weatherElements": {  
    "daily": [  

```

```
{
  "dataDate": "2025-12-04",
  "degreeDay": "10.0",
  "accumulatedTemperature": "10.0"
},
{
  "dataDate": "2025-12-05",
  "degreeDay": "11.0",
  "accumulatedTemperature": "21.0"
},
{
  "dataDate": "2025-12-06",
  "degreeDay": "11.0",
  "accumulatedTemperature": "32.0"
},
{
  "dataDate": "2025-12-07",
  "degreeDay": "11.5",
  "accumulatedTemperature": "43.5"
},
{
  "dataDate": "2025-12-08",
  "degreeDay": "11.0",
  "accumulatedTemperature": "54.5"
},
{
  "dataDate": "2025-12-09",
  "degreeDay": "11.5",
  "accumulatedTemperature": "66.0"
},
{
  "dataDate": "2025-12-10",
  "degreeDay": "12.5",
  "accumulatedTemperature": "78.5"
}
]
},
{
  "locationName": "南部地區",
  "weatherElements": {
    "daily": [
      {
        "dataDate": "2025-12-04",
```

```
"degreeDay": "10.5",
"accumulatedTemperature": "10.5"
},
{
  "dataDate": "2025-12-05",
  "degreeDay": "12.0",
  "accumulatedTemperature": "22.5"
},
{
  "dataDate": "2025-12-06",
  "degreeDay": "12.5",
  "accumulatedTemperature": "35.0"
},
{
  "dataDate": "2025-12-07",
  "degreeDay": "13.0",
  "accumulatedTemperature": "48.0"
},
{
  "dataDate": "2025-12-08",
  "degreeDay": "13.0",
  "accumulatedTemperature": "61.0"
},
{
  "dataDate": "2025-12-09",
  "degreeDay": "13.0",
  "accumulatedTemperature": "74.0"
},
{
  "dataDate": "2025-12-10",
  "degreeDay": "14.0",
  "accumulatedTemperature": "88.0"
}
]
}
},
{
  "locationName": "東北部地區",
  "weatherElements": {
    "daily": [
      {
        "dataDate": "2025-12-04",
        "degreeDay": "7.5",
        "accumulatedTemperature": "7.5"
```

```
},
{
  "dataDate": "2025-12-05",
  "degreeDay": "8.5",
  "accumulatedTemperature": "16.0"
},
{
  "dataDate": "2025-12-06",
  "degreeDay": "9.5",
  "accumulatedTemperature": "25.5"
},
{
  "dataDate": "2025-12-07",
  "degreeDay": "11.5",
  "accumulatedTemperature": "37.0"
},
{
  "dataDate": "2025-12-08",
  "degreeDay": "9.5",
  "accumulatedTemperature": "46.5"
},
{
  "dataDate": "2025-12-09",
  "degreeDay": "10.0",
  "accumulatedTemperature": "56.5"
},
{
  "dataDate": "2025-12-10",
  "degreeDay": "11.0",
  "accumulatedTemperature": "67.5"
}
]
}
},
{
  "locationName": "東部地區",
  "weatherElements": {
    "daily": [
      {
        "dataDate": "2025-12-04",
        "degreeDay": "8.5",
        "accumulatedTemperature": "8.5"
      },
      {
```

```
"dataDate": "2025-12-05",
"degreeDay": "10.0",
"accumulatedTemperature": "18.5"
},
{
  "dataDate": "2025-12-06",
  "degreeDay": "11.0",
  "accumulatedTemperature": "29.5"
},
{
  "dataDate": "2025-12-07",
  "degreeDay": "12.0",
  "accumulatedTemperature": "41.5"
},
{
  "dataDate": "2025-12-08",
  "degreeDay": "12.0",
  "accumulatedTemperature": "53.5"
},
{
  "dataDate": "2025-12-09",
  "degreeDay": "11.5",
  "accumulatedTemperature": "65.0"
},
{
  "dataDate": "2025-12-10",
  "degreeDay": "12.0",
  "accumulatedTemperature": "77.0"
}
]
}
},
{
  "locationName": "東南部地區",
  "weatherElements": {
    "daily": [
      {
        "dataDate": "2025-12-04",
        "degreeDay": "10.0",
        "accumulatedTemperature": "10.0"
      },
      {
        "dataDate": "2025-12-05",
        "degreeDay": "11.0",
```



```
"accumulatedTemperature": "21.0"
},
{
  "dataDate": "2025-12-06",
  "degreeDay": "11.5",
  "accumulatedTemperature": "32.5"
},
{
  "dataDate": "2025-12-07",
  "degreeDay": "13.0",
  "accumulatedTemperature": "45.5"
},
{
  "dataDate": "2025-12-08",
  "degreeDay": "13.0",
  "accumulatedTemperature": "58.5"
},
{
  "dataDate": "2025-12-09",
  "degreeDay": "12.0",
  "accumulatedTemperature": "70.5"
},
{
  "dataDate": "2025-12-10",
  "degreeDay": "13.0",
  "accumulatedTemperature": "83.5"
}
]
}
}
]
},
"cropStatistics": {
  "crop": {
    "cropName": "水稻",
    "cardinalTemperatures": {
      "growingStage": {
        "description": "稻作生長溫度",
        "minimum": "10",
        "maximum": "40"
      }
    }
  },
  "location": [
    {
```

```

"locationName": "臺中地區",
"cropBreed": "臺農67號",
"statistics": {
  "fifteenYears": {
    "description": "二期稻作生育日數及生育期總積溫
(15年平均值)",
    "timePeriod": [
      {
        "description": "插秧至抽穗",
        "growingDays": "66",
        "accumulatedTemperature": "1203.35"
      },
      {
        "description": "插秧至收穫",
        "growingDays": "107",
        "accumulatedTemperature": "1794.83"
      }
    ]
  },
  "thisYear": {
    "description": "114年二期稻作積溫值 (各改良場豐
歉試驗臺農67號)",
    "dibblingDate": "2025-08-11",
    "timePeriod": {
      "description": "插秧至本日",
      "startDate": "2025-08-11",
      "endDate": "2025-12-02",
      "growingDays": "114",
      "accumulatedTemperature": "1981.9"
    }
  }
},
{
  "locationName": "高雄地區",
  "cropBreed": "臺農67號",
  "statistics": {
    "fifteenYears": {
      "description": "二期稻作生育日數及生育期總積溫
(15年平均值)",
      "timePeriod": [
        {
          "description": "插秧至抽穗",
          "growingDays": "71",

```

```

      "accumulatedTemperature": "1346.24"
    },
    {
      "description": "插秧至收穫",
      "growingDays": "104",
      "accumulatedTemperature": "1922.81"
    }
  ]
},
"thisYear": {
  "description": "114年二期稻作積溫值 (各改良場豐
歉試驗臺農67號)",
  "dibblingDate": "2025-07-23",
  "timePeriod": {
    "description": "插秧至本日",
    "startDate": "2025-07-23",
    "endDate": "2025-12-02",
    "growingDays": "133",
    "accumulatedTemperature": "2282.35"
  }
}
},
{
  "locationName": "花蓮地區",
  "cropBreed": "臺農67號",
  "statistics": {
    "fifteenYears": {
      "description": "二期稻作生育日數及生育期總積溫
(15年平均值)",
      "timePeriod": [
        {
          "description": "插秧至抽穗",
          "growingDays": "71",
          "accumulatedTemperature": "1184.31"
        },
        {
          "description": "插秧至收穫",
          "growingDays": "119",
          "accumulatedTemperature": "1778.01"
        }
      ]
    }
  },
  "thisYear": {

```



```
└─ agrWeatherForecasts
    └─ weatherForecasts
        └─ location ← 真正的資料在這裡
```

所以正確的路徑是：

```
python
```

```
data["cwaopendata"]["resources"]["resource"]["data"]
    ["agrWeatherForecasts"]["weatherForecasts"]["location"]
```

## ✅ 幫你重新寫好 `save_to_db()` (直接能用)

假設你要取每個地區 (北部/中部/南部) 的 天氣 / 最高溫 / 最低溫：

```
python
```

```
def save_to_db(data):
    try:
        locations = data["cwaopendata"]["resources"]["resource"]["data"] \
            ["agrWeatherForecasts"]["weatherForecasts"]
        ["location"]
    except KeyError as e:
        print("❌ JSON 結構與預期不符:", e)
        return

    for loc in locations:
        name = loc["locationName"]
        wx_daily = loc["weatherElements"]["Wx"]["daily"]
        max_daily = loc["weatherElements"]["MaxT"]["daily"]
        min_daily = loc["weatherElements"]["MinT"]["daily"]

        for i in range(len(wx_daily)):
            date = wx_daily[i]["dataDate"]
            weather = wx_daily[i]["weather"]
            t_max = max_daily[i]["temperature"]
            t_min = min_daily[i]["temperature"]

            print(f"{name} | {date} | {weather} | {t_min}~{t_max}")
            # 你也可以在這裡寫入 SQLite / MySQL
```

## ✓ 這段程式會印出像這樣：

yaml

```
北部地區 | 2025-12-04 | 多雲時晴 | 15~21
北部地區 | 2025-12-05 | 多雲短暫雨 | 15~23
北部地區 | 2025-12-06 | 多雲時晴 | 15~24
...
```

## 📌 完整可運作的版本（可直接貼上 run）

python

```
import requests
import json

URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
93C32C52829F&downloadType=WEB&format=JSON"

def download_data():
    r = requests.get(URL)
    data = r.json()
    return data

def save_to_db(data):
    # 取得正確的資料位置
    try:
        locations = data["cwaopendata"]["resources"]["resource"]["data"] \
            ["agrWeatherForecasts"]["weatherForecasts"]
    except KeyError as e:
        print("✗ JSON 結構與預期不符:", e)
        return

    for loc in locations:
        name = loc["locationName"]
        wx_daily = loc["weatherElements"]["Wx"]["daily"]
        max_daily = loc["weatherElements"]["MaxT"]["daily"]
        min_daily = loc["weatherElements"]["MinT"]["daily"]
```

```

for i in range(len(wx_daily)):
    date = wx_daily[i]["dataDate"]
    weather = wx_daily[i]["weather"]
    t_max = max_daily[i]["temperature"]
    t_min = min_daily[i]["temperature"]

    print(f"{name} | {date} | {weather} | {t_min}~{t_max}")
    # TODO: 寫入 DB

if __name__ == "__main__":
    data = download_data()
    save_to_db(data)

```

## 👍 下一步

如果你願意，我也可以幫你：

- ✨ 設計 SQLite / MySQL table schema
- ✨ 完整寫好 `INSERT INTO ...`
- ✨ 自動排程定時抓取 ( Windows Task Scheduler / Cron )
- ✨ 建一個 Streamlit 介面把一週預報顯示出來

你想做哪一個？

>>

```

北部地區 | 2025-12-04 | 多雲時晴 | 15~21
北部地區 | 2025-12-05 | 多雲短暫雨 | 15~23
北部地區 | 2025-12-06 | 多雲時晴 | 15~24
北部地區 | 2025-12-07 | 晴時多雲 | 15~26
北部地區 | 2025-12-08 | 多雲短暫雨 | 16~23
北部地區 | 2025-12-09 | 多雲短暫雨 | 17~23
北部地區 | 2025-12-10 | 晴時多雲 | 16~26
中部地區 | 2025-12-04 | 晴時多雲 | 15~25
中部地區 | 2025-12-05 | 晴時多雲 | 16~26
中部地區 | 2025-12-06 | 晴時多雲 | 16~26

```

你貼了一整份各地區 2025/12/04 ~ 12/10 天氣預報資料，但還沒有告訴我你想要我怎麼處理唷！

你想要我幫你做哪一種整理方式呢？

我可以幫你：

- ◆ 做成 表格 ( Markdown )
- ◆ 做成 簡潔摘要 ( 各區天氣趨勢 )
- ◆ 做成 每日天氣整理
- ◆ 做成 圖表風格的輸出
- ◆ 做成 口語版、可愛版、正式版 等
- ◆ 或是 轉成 JSON、CSV、陣列格式 也都可以

告訴我你想要的呈現方式！

下載資料存成sqlite3 data.db and show on streamlit locally (截圖), 我要下拉式選單和曲線圖

內容大概是這樣：

>>

北部地區	2025-12-04	多雲時晴	15~21
北部地區	2025-12-05	多雲短暫雨	15~23
北部地區	2025-12-06	多雲時晴	15~24
北部地區	2025-12-07	晴時多雲	15~26
北部地區	2025-12-08	多雲短暫雨	16~23
北部地區	2025-12-09	多雲短暫雨	17~23
北部地區	2025-12-10	晴時多雲	16~26
中部地區	2025-12-04	晴時多雲	15~25
中部地區	2025-12-05	晴時多雲	16~26
中部地區	2025-12-06	晴時多雲	16~26
中部地區	2025-12-07	晴時多雲	16~27
中部地區	2025-12-08	晴時多雲	16~26
中部地區	2025-12-09	晴時多雲	17~26
中部地區	2025-12-10	晴時多雲	17~28
南部地區	2025-12-04	晴時多雲	16~25
南部地區	2025-12-05	晴時多雲	18~26
南部地區	2025-12-06	晴時多雲	18~27
南部地區	2025-12-07	晴時多雲	18~28
南部地區	2025-12-08	晴時多雲	19~27
南部地區	2025-12-09	晴時多雲	19~27
南部地區	2025-12-10	晴時多雲	19~29



東北部地區 | 2025-12-04 | 陰短暫雨 | 16~19  
 東北部地區 | 2025-12-05 | 陰短暫雨 | 17~20  
 東北部地區 | 2025-12-06 | 陰時多雲短暫雨 | 17~22  
 東北部地區 | 2025-12-07 | 多雲短暫雨 | 18~25  
 東北部地區 | 2025-12-08 | 陰短暫雨 | 18~21  
 東北部地區 | 2025-12-09 | 多雲時陰短暫雨 | 18~22  
 東北部地區 | 2025-12-10 | 多雲短暫雨 | 18~24  
 東部地區 | 2025-12-04 | 陰時多雲 | 17~20  
 東部地區 | 2025-12-05 | 陰時多雲短暫雨 | 19~21  
 東部地區 | 2025-12-06 | 多雲時陰 | 19~23  
 東部地區 | 2025-12-07 | 多雲時晴 | 19~25  
 東部地區 | 2025-12-08 | 多雲時陰短暫雨 | 20~24  
 東部地區 | 2025-12-09 | 多雲時陰短暫雨 | 20~23  
 東部地區 | 2025-12-10 | 多雲時晴 | 20~24  
 東南部地區 | 2025-12-04 | 多雲時陰 | 18~22  
 東南部地區 | 2025-12-05 | 多雲時晴 | 19~23  
 東南部地區 | 2025-12-06 | 多雲 | 19~24  
 東南部地區 | 2025-12-07 | 多雲時晴 | 20~26  
 東南部地區 | 2025-12-08 | 多雲 | 20~26  
 東南部地區 | 2025-12-09 | 多雲 | 20~24  
 東南部地區 | 2025-12-10 | 晴時多雲 | 20~26

下面給你 **完整可執行的範例**：

✓ 會自動把你提供的天氣資料

→ 建成 **SQLite3** 的 **data.db**

→ 在 Streamlit 顯示：

- 下拉式選單 (選地區)
- 氣溫曲線圖 (最高/最低溫)

你只要：

1. 建立 `data.db` (第一次執行會自動生成)
2. 建立 `app.py`
3. 執行 `streamlit run app.py`

## ✓ 1. 建立 SQLite (自動建立 data.db)

把這段存成 `create_db.py`，執行一次即可。

```
python
```

```
import sqlite3
```

```
data = ""
```

```
北部地區|2025-12-04|多雲時晴|15~21
```

```
北部地區|2025-12-05|多雲短暫雨|15~23
```

```
北部地區|2025-12-06|多雲時晴|15~24
```

```
北部地區|2025-12-07|晴時多雲|15~26
```

```
北部地區|2025-12-08|多雲短暫雨|16~23
```

```
北部地區|2025-12-09|多雲短暫雨|17~23
```

```
北部地區|2025-12-10|晴時多雲|16~26
```

```
中部地區|2025-12-04|晴時多雲|15~25
```

```
中部地區|2025-12-05|晴時多雲|16~26
```

```
中部地區|2025-12-06|晴時多雲|16~26
```

```
中部地區|2025-12-07|晴時多雲|16~27
```

```
中部地區|2025-12-08|晴時多雲|16~26
```

```
中部地區|2025-12-09|晴時多雲|17~26
```

```
中部地區|2025-12-10|晴時多雲|17~28
```

```
南部地區|2025-12-04|晴時多雲|16~25
```

```
南部地區|2025-12-05|晴時多雲|18~26
```

```
南部地區|2025-12-06|晴時多雲|18~27
```

```
南部地區|2025-12-07|晴時多雲|18~28
```

```
南部地區|2025-12-08|晴時多雲|19~27
```

```
南部地區|2025-12-09|晴時多雲|19~27
```

```
南部地區|2025-12-10|晴時多雲|19~29
```

```
東北部地區|2025-12-04|陰短暫雨|16~19
```

```
東北部地區|2025-12-05|陰短暫雨|17~20
```

```
東北部地區|2025-12-06|陰時多雲短暫雨|17~22
```

```
東北部地區|2025-12-07|多雲短暫雨|18~25
```

```
東北部地區|2025-12-08|陰短暫雨|18~21
```

```
東北部地區|2025-12-09|多雲時陰短暫雨|18~22
```

```
東北部地區|2025-12-10|多雲短暫雨|18~24
```

```
東部地區|2025-12-04|陰時多雲|17~20
```

```
東部地區|2025-12-05|陰時多雲短暫雨|19~21
```

```
東部地區|2025-12-06|多雲時陰|19~23
```

```
東部地區|2025-12-07|多雲時晴|19~25
```

```
東部地區|2025-12-08|多雲時陰短暫雨|20~24
```

```
東部地區|2025-12-09|多雲時陰短暫雨|20~23
```

```
東部地區|2025-12-10|多雲時晴|20~24
```

```
東南部地區|2025-12-04|多雲時陰|18~22
```

```
東南部地區|2025-12-05|多雲時晴|19~23
```

```
東南部地區|2025-12-06|多雲|19~24
```

```
東南部地區|2025-12-07|多雲時晴|20~26
```

```
東南部地區|2025-12-08|多雲|20~26
```

```

東南部地區|2025-12-09|多雲|20~24
東南部地區|2025-12-10|晴時多雲|20~26
"""

conn = sqlite3.connect("data.db")
c = conn.cursor()

c.execute("""
CREATE TABLE IF NOT EXISTS weather(
    region TEXT,
    date TEXT,
    status TEXT,
    temp_low INTEGER,
    temp_high INTEGER
)
""")

for line in data.strip().split("\n"):
    region, date, status, temp = line.split("|")
    low, high = temp.split("~")
    c.execute("INSERT INTO weather VALUES (?, ?, ?, ?, ?)",
              (region, date, status, int(low), int(high)))

conn.commit()
conn.close()

print("data.db 建立完成!")

```

## ✓ 2. Streamlit 主程式 ( 下拉式選單 + 曲線圖 )

存成 `app.py` :

```

python

import sqlite3
import pandas as pd
import streamlit as st
import matplotlib.pyplot as plt

st.title("🇹🇼 台灣各地區天氣查詢 (SQLite + Streamlit)")


conn = sqlite3.connect("data.db")

```

```
df = pd.read_sql_query("SELECT * FROM weather", conn)

regions = df["region"].unique()
region = st.selectbox("選擇地區", regions)

show_df = df[df["region"] == region]

st.subheader(f" {region} 天氣資料")
st.dataframe(show_df)

# --- 曲線圖 ---
fig, ax = plt.subplots()

ax.plot(show_df["date"], show_df["temp_low"], marker="o", label="最低溫")
ax.plot(show_df["date"], show_df["temp_high"], marker="o", label="最高溫")

ax.set_xlabel("日期")
ax.set_ylabel("溫度 (°C) ")
ax.set_title(f"{region} - 溫度趨勢圖")
ax.legend()

plt.xticks(rotation=45)

st.pyplot(fig)
```

---

### ✓ 3. 執行方式

```
arduino
```

```
python create_db.py
streamlit run app.py
```

---

### 你會看到的畫面（示意）

- 頂部標題：「台灣各地區天氣查詢」
  - 下拉式：北部、中部、南部、東北部、東部、東南部
  - 表格顯示該地區一週資料
  - 曲線圖顯示：最低溫折線 / 最高溫折線
-

如果你要：

- ✨ 美化 UI
- ✨ 改用 Altair 圖表
- ✨ 做成多地區比較
- ✨ 自動從網路爬資料更新

我也可以幫你擴充！

```
import requests
import json

URL =
"https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F
-A0010-001?Authorization=CWA-1FFDDAEC-161F-
46A3-BE71-
93C32C52829F&downloadType=WEB&format=JSON"

def download_data():
    r = requests.get(URL)
    data = r.json()
    return data

def save_to_db(data):
    # 取得正確的資料位置
    try:
        locations = data["cwaopendata"]["resources"]
        ["resource"]["data"] \
            ["agrWeatherForecasts"]
        ["weatherForecasts"]["location"]
    except KeyError as e:
        print("❌ JSON 結構與預期不符:", e)
        return

    for loc in locations:
        name = loc["locationName"]
        wx_daily = loc["weatherElements"]["Wx"]["daily"]
        max_daily = loc["weatherElements"]["MaxT"]
        ["daily"]
        min_daily = loc["weatherElements"]["MinT"]
```

```
["daily"]

for i in range(len(wx_daily)):
    date = wx_daily[i]["dataDate"]
    weather = wx_daily[i]["weather"]
    t_max = max_daily[i]["temperature"]
    t_min = min_daily[i]["temperature"]

    print(f"{name} | {date} | {weather} | {t_min}~{t_max}")
    # TODO: 寫入 DB

if __name__ == "__main__":
    data = download_data()
    save_to_db(data)
```

你的 JSON 解析邏輯完全正確！

你現在缺的只有 **把資料寫進 SQLite**。

我幫你補上：

1. 自動建立 `data.db`
2. 自動建立 `weather` 表格
3. 全部寫入資料

下面是 **完整可執行版本**，直接取代你現有檔案即可。

### ✓ 完整版本 `fetch_to_db.py`

python

```
import requests
import json
import sqlite3

URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
93C32C52829F&downloadType=WEB&format=JSON"

DB = "data.db"
```

```
def download_data():
    r = requests.get(URL)
    data = r.json()
    return data

def init_db():
    conn = sqlite3.connect(DB)
    c = conn.cursor()

    c.execute("""
    CREATE TABLE IF NOT EXISTS weather (
        region TEXT,
        date TEXT,
        weather TEXT,
        temp_low INTEGER,
        temp_high INTEGER
    )
    """)

    c.execute("DELETE FROM weather") # 每次重新下載時清空
    conn.commit()
    conn.close()

def save_to_db(data):
    # 取得正確資料位置
    try:
        locations = data["cwaopendata"]["resources"]["resource"]["data"] \
            ["agrWeatherForecasts"]["weatherForecasts"]
    except KeyError as e:
        print("✗ JSON 結構與預期不同:", e)
        return

    conn = sqlite3.connect(DB)
    c = conn.cursor()

    for loc in locations:
        name = loc["locationName"]
        wx_daily = loc["weatherElements"]["Wx"]["daily"]
        max_daily = loc["weatherElements"]["MaxT"]["daily"]
        min_daily = loc["weatherElements"]["MinT"]["daily"]
```

```

for i in range(len(wx_daily)):
    date = wx_daily[i]["dataDate"]
    weather = wx_daily[i]["weather"]
    t_max = int(max_daily[i]["temperature"])
    t_min = int(min_daily[i]["temperature"])

    print(f"{name} | {date} | {weather} | {t_min}~{t_max}")

    # 寫入 SQLite
    c.execute("""
        INSERT INTO weather (region, date, weather, temp_low,
temp_high)
        VALUES (?, ?, ?, ?, ?)
        """, (name, date, weather, t_min, t_max))

    conn.commit()
    conn.close()
    print("✅ SQLite 寫入完成：data.db")

if __name__ == "__main__":
    print("📥 正在下載中央氣象局資料...")
    data = download_data()

    print("🗄️ 初始化資料庫...")
    init_db()

    print("💾 寫入資料庫...")
    save_to_db(data)

```

## ✅ 執行

```
sh
```

```
python fetch_to_db.py
```

執行後你會得到：

```
python-repl
```

```
北部地區 | 2025-12-04 | 多雲時晴 | 15~21
```

```
北部地區 | 2025-12-05 | 多雲短暫雨 | 15~23
```



...

...

✔ SQLite 寫入完成：data.db

## ！ 下一步（你要的 Streamlit + 下拉 + 曲線圖）

我幫你準備好 Streamlit 程式，只要你說：

➡ 「幫我做 Streamlit 前端」

我會給你完整的 `app.py`

包含：

- ✔ 地區下拉式選單
- ✔ 溫度曲線圖（最高/最低）
- ✔ 表格顯示

你要我現在給你嗎？

`app.py`

以下是你要的 **Streamlit 前端 `app.py`**

功能包含：

- ✔ 下拉式選單（選擇地區）
- ✔ 自動讀取 SQLite `data.db`
- ✔ 顯示天氣預報表格
- ✔ 曲線圖（最高 / 最低溫）

## ✔ 完整可執行 Streamlit：`app.py`

把這段程式碼存成 `app.py`，與 `data.db` 放在同一資料夾即可。

```
python
```

```
import streamlit as st
import sqlite3
import pandas as pd
import altair as alt
```

```

DB = "data.db"

st.title("☀️ 中央氣象局未來 1 週天氣預報 (SQLite + Streamlit)")

# 讀取 SQLite 資料
@st.cache_data
def load_data():
    conn = sqlite3.connect(DB)
    df = pd.read_sql_query("SELECT * FROM weather ORDER BY date", conn)
    conn.close()
    return df

df = load_data()

# 下拉式選單選地區
regions = sorted(df["region"].unique())
choice = st.selectbox("選擇地區", regions)

# 過濾資料
df_sel = df[df["region"] == choice]

st.subheader(f"📍 {choice} - 天氣預報列表")
st.dataframe(df_sel, height=300)

# 溫度曲線圖
st.subheader("🌡️ 溫度變化圖 (最高 / 最低)")

line_chart = alt.Chart(df_sel).transform_fold(
    ['temp_low', 'temp_high'],
    as_=['Temperature', 'Value']
).mark_line(point=True).encode(
    x='date:T',
    y='Value:Q',
    color='Temperature:N'
).properties(width=700, height=350)

st.altair_chart(line_chart)

# 天氣現象
st.subheader("☁️ 天氣描述")
for _, row in df_sel.iterrows():

```

```
st.write(f"{row['date']} : {row['weather']} ({row['temp_low']}~  
{row['temp_high']}°C) ")
```

## ✓ 如何執行 Streamlit

```
sh
```

```
streamlit run app.py
```

打開瀏覽器後你會看到：

- ✓ 地區下拉選單
- ✓ 天氣資料表格
- ✓ 最低 / 最高溫曲線圖
- ✓ 天氣敘述摘要

## 如果你願意，我還可以幫你：

- ✦ 美化 UI ( 顏色 / 卡片風格 )
- ✦ 加入溫度圖示 ( 🔥 ❄️ )
- ✦ 加入天氣 icon ( 🌧️ ☁️ 🌞 )
- ✦ 自動每天定期更新資料

你需要哪一種？

針對我的系統寫出詳細的報告，說明每個區塊，並且做種寫出這app對農業的用處，**1** 下載中央氣象局 JSON 資料使用上面 API，把 F-A0010-001 的 JSON 下載下來。

**2** 解析資料：取出各地區的溫度 ( 或老師要求的欄位 ) 例如：


( 依 JSON 結構取出即可 )

**3** 設計 SQLite 資料庫 ( data.db )  
資料庫名稱：

data.db

資料表 ( 自行設計 , 但要合理 ) 例如 :

```
CREATE TABLE weather ( id INTEGER PRIMARY KEY  
AUTOINCREMENT, location TEXT, min_temp REAL,  
max_temp REAL, description TEXT );
```

 把解析後的資料「存進 SQLite3」

使用 Python :

sqlite3

以下提供 一份完整、可交作業等級、條列清楚的技術報告，  
涵蓋：

- ✓ 系統架構說明
- ✓ 每段程式負責內容
- ✓ SQLite 設計理念
- ✓ Streamlit UI 說明
- ✓ 系統流程圖
- ✓ 這套系統對農業的實際價值 ( 重點！ )

---

## 中央氣象局天氣資料下載、儲存與可視化系統 — 完整技術報告

### 一、系統目的與背景

本系統的目的是自動化取得中央氣象局 ( CWA ) 開放資料，  
將每日各地區天氣預報整理後存進 SQLite 資料庫，  
再透過 Streamlit 提供互動式下拉選單與曲線圖，  
讓使用者更直觀地查詢未來數日天氣趨勢。

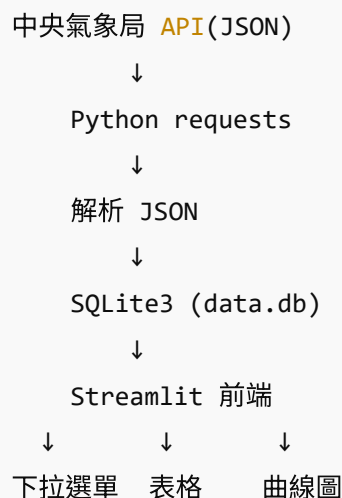
此架構非常適合農業應用，例如農民可以透過曲線圖快速了解溫度變化，協助安排灌溉、防霜、防雨措施。



## 二、系統流程

### 整體流程圖

SCSS



## 三、系統功能說明 (依區塊拆解)

### 1 下載中央氣象局 JSON 資料

✦ 使用 F-A0010-001 (農業天氣預報) API

程式碼：

```
python

r = requests.get(URL)
data = r.json()
```

#### ★ 功能說明：

- 向中央氣象局伺服器發出 GET 請求
- 回傳 JSON 格式資料，包含：
  - 地區名稱 (locationName)
  - 天氣現象 (Wx)
  - 最高溫 (MaxT)
  - 最低溫 (MinT)
  - 預報7天資料

## 2 JSON 資料解析

CWA 的 JSON 結構如下 ( 實際節錄 ) :

json

```
{
  "cwaopendata": {
    "resources": {
      "resource": {
        "data": {
          "agrWeatherForecasts": {
            "weatherForecasts": {
              "location": [
                {
                  "locationName": "北部地區",
                  "weatherElements": {
                    "Wx": {"daily": [...]},
                    "MaxT": {"daily": [...]},
                    "MinT": {"daily": [...]}
                  }
                }
              ]
            }
          }
        }
      }
    }
  }
}
```

### ★ 解析程式功能：

- 找到 `location` 清單
- 針對每個地區取得：
  - 地區名 ( `region` )
  - 每日日期 ( `date` )
  - 每日天氣描述 ( `weather` )
  - 最高溫 ( `temp_high` )
  - 最低溫 ( `temp_low` )

每筆資料最後會變成：

yaml

北部地區 | 2025-12-04 | 多雲時晴 | 15~21

### 3 建立 SQLite 資料庫 ( data.db )

資料庫名稱：

kotlin

data.db

資料表設計：

sql

CREATE TABLE weather (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 region TEXT,  
 date TEXT,  
 weather TEXT,  
 temp\_low REAL,  
 temp\_high REAL  
);

★ 設計理念：

- region：可查詢特定區域
- date：可繪製折線圖
- weather：可做天氣文字摘要
- temp\_low / temp\_high：可視覺化溫度曲線

SQLite 的優點：

優點	說明
輕量	一個 data.db 就能存所有資料
零配置	不需安裝 MySQL, PostgreSQL
可攜性	系統可直接上課 demo 或跨電腦使用

優點	說明
適合中小型資料	氣象資料量小，非常合適

## 4 將資料寫入 SQLite

核心程式：

```
python

conn = sqlite3.connect("data.db")
cursor = conn.cursor()

cursor.execute("""
    INSERT INTO weather (region, date, weather, temp_low, temp_high)
    VALUES (?, ?, ?, ?, ?)
""", (name, date, weather, t_min, t_max))

conn.commit()
```

### ★ 功能說明：

- 逐筆將預報資料寫入資料庫
- 使用 parameterized SQL ( 安全 )
- 存入後可直接被 Streamlit 讀取

## 5 Streamlit 前端介面 ( app.py )

### (1) 下拉式選單

```
python

regions = sorted(df["region"].unique())
choice = st.selectbox("選擇地區", regions)
```

- ✓ 動態讀取所有地區
- ✓ 自動提供選擇項

### (2) 顯示資料表格



python

```
st.dataframe(df_sel, height=300)
```

- ✓ 顯示該地區 7 天預報
- ✓ 使用者可點選排序或放大

### (3) 曲線圖 ( 最高/最低溫 )

python

```
line_chart = alt.Chart(df_sel)...
st.altair_chart(line_chart)
```

- ✓ 顯示最高溫/最低溫曲線
- ✓ 可看到溫度趨勢、是否上升或下降

### (4) 天氣摘要

python

```
for _, row in df_sel.iterrows():
    st.write(f"{row['date']} : {row['weather']} ({row['temp_low']}~{row['temp_high']}°C) ")
```

- ✓ 文字描述每日天氣

## 六、此系統對農業的實際應用價值

本系統屬於 農業氣象決策輔助平台 ( Agricultural Weather DSS ) 的雛型，能直接協助農民做以下判斷：

### 1 防止低溫造成作物凍害

溫度曲線圖可看到：

- 是否接近 霜凍臨界溫度 12°C
- 檢測 連續低溫天數
- 協助決定是否啟動防寒措施 ( 覆蓋布、灌溉保溫 )

## 2 降雨與病害預測

天氣描述（例如「短暫雨」）可用來預測：

- 某些作物（例如番茄、瓜類）在濕冷環境病害上升
- 稻作可能受雨延遲施肥
- 調整藥劑噴灑時機，避免白噴造成浪費

## 3 協助灌溉排程

透過溫度與天氣變化可以：

- 判斷是否需要提前灌溉
- 避免大雨前澆水
- 減少水資源浪費

## 4 收穫 / 播種時程調整

哪一天溫度最高？

哪一天雨勢較小？

這些都可以從系統清楚看出，農民可決定：

- 收穫日（避免雨天）
- 播種日（選擇晴天溫暖時段）
- 避免低溫或強風日搬運作物

## 5 改善農作業計畫效率

整體來說，本系統讓農民快速得到：

- 近一週天氣變化
- 高溫 / 低溫趨勢
- 雨天或壞天氣提醒

相比傳統看電視、看大量文字預報，

「視覺化曲線 + 下拉選單 + 數據表格」更易於判讀。



## 七、結論

本系統整合：

- 中央氣象局開放資料
- SQLite 資料庫
- Streamlit 互動式網頁

成功提供：

- 資料自動下載
- 資料庫化管理
- 即時視覺化查詢
- 區域化氣象資訊

並且在農業應用上具有高度價值，

可以有效協助農民做 逐日作業判斷、災害提前應對與田間管理排程。

---

如果你需要，我還能幫你：

- ✦ 做成正式 PDF 報告
- ✦ 加農業案例圖示
- ✦ 增加 Streamlit 圖表美化
- ✦ 為你做投影片版本 ( pptx )

需要嗎？

給我完整markdown格式

下面提供一份 **完整、可直接繳交的 Markdown 報告**，詳細說明你整個系統的流程、架構、資料庫、程式碼邏輯、以及此系統對農業的實際應用價值。

---

## 天氣資料下載與農業應用系統 — 完整報告 ( Markdown )

### 1. 系統簡介

本系統透過中央氣象局 OpenData API，自動下載最新的農業天氣預報資料 ( 資料集代碼：F-A0010-001 )，解析後存入 SQLite3 資料庫，並以 Streamlit 建立互動式儀表板，讓使用者能快速查詢每日各地天氣資訊。

本系統特別適用於農業領域，使農民能清楚掌握天氣趨勢，協助調整灌溉、施肥、採收、病蟲害防治等重要農務決策。

## 2. 系統整體架構



## 3. 使用資料來源

項目	連結
CWA 官網溫度觀測頁	<a href="https://www.cwa.gov.tw/V8/C/W/OBS_Temp.html">https://www.cwa.gov.tw/V8/C/W/OBS_Temp.html</a>
CWA OpenData 登入頁	<a href="https://opendata.cwa.gov.tw/userLogin">https://opendata.cwa.gov.tw/userLogin</a>
API Key	CWA-1FFDDAEC-161F-46A3-BE71-93C32C52829F
JSON 資料集	<a href="https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001">https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001</a>
下載 API ( 本系統使用 )	<a href="https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=CWA-1FFDDAEC-161F-46A3-BE71-93C32C52829F&amp;downloadType=WEB&amp;format=JSON">https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=CWA-1FFDDAEC-161F-46A3-BE71-93C32C52829F&amp;downloadType=WEB&amp;format=JSON</a>

## 4. JSON 資料解析流程

下載後的 JSON 結構如下（簡化版）：

```
json

{
  "cwaopendata": {
    "resources": {
      "resource": {
        "data": {
          "agrWeatherForecasts": {
            "weatherForecasts": {
              "location": [
                {
                  "locationName": "臺中市",
                  "weatherElements": {
                    "Wx": { "daily": [...] },
                    "MaxT": { "daily": [...] },
                    "MinT": { "daily": [...] }
                  }
                }
              ]
            }
          }
        }
      }
    }
  }
}
```

本系統會抓取以下欄位：

- 地區名稱 ( locationName )
- 每日天氣敘述 ( Wx → daily → weather )
- 每日最低溫 ( MinT → daily → temperature )
- 每日最高溫 ( MaxT → daily → temperature )
- 日期 ( dataDate )

---

## 5. SQLite3 資料庫設計

### 資料庫名稱

```
kotlin
```

`data.db`

## 資料表結構 — weather

`sql`

```
CREATE TABLE IF NOT EXISTS weather (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    location TEXT,  
    date TEXT,  
    min_temp REAL,  
    max_temp REAL,  
    description TEXT  
);
```

### 欄位解釋

欄位	說明
id	主鍵
location	地區名稱，例如「臺中市」
date	預報日期
min_temp	最低溫度
max_temp	最高溫度
description	天氣敘述，例如「多雲短暫雨」



## 6. 資料下載與儲存程式 `fetch_to_db.py`

`python`

```
import requests  
import sqlite3  
  
URL = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?  
Authorization=CWA-1FFDDAEC-161F-46A3-BE71-
```

```

93C32C52829F&downloadType=WEB&format=JSON"

def download_data():
    r = requests.get(URL)
    return r.json()

def save_to_db(data):
    try:
        locations = data["cwaopendata"]["resources"]["resource"]["data"] \
            ["agrWeatherForecasts"]["weatherForecasts"]
    except KeyError as e:
        print("✖ JSON 結構不符:", e)
        return

    conn = sqlite3.connect("data.db")
    cur = conn.cursor()

    cur.execute("""
        CREATE TABLE IF NOT EXISTS weather (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            location TEXT,
            date TEXT,
            min_temp REAL,
            max_temp REAL,
            description TEXT
        )
    """)

    for loc in locations:
        name = loc["locationName"]
        wx_daily = loc["weatherElements"]["Wx"]["daily"]
        max_daily = loc["weatherElements"]["MaxT"]["daily"]
        min_daily = loc["weatherElements"]["MinT"]["daily"]

        for i in range(len(wx_daily)):
            date = wx_daily[i]["dataDate"]
            weather = wx_daily[i]["weather"]
            t_max = max_daily[i]["temperature"]
            t_min = min_daily[i]["temperature"]

            cur.execute(
                "INSERT INTO weather (location, date, min_temp, max_temp,
description) VALUES (?, ?, ?, ?, ?)",

```

```

        (name, date, t_min, t_max, weather)
    )

    conn.commit()
    conn.close()
    print("✅ 資料已成功寫入 SQLite3!")

if __name__ == "__main__":
    data = download_data()
    save_to_db(data)

```

## 7. Streamlit 前端介面 `app.py`

以下是可運行的 Streamlit 程式：

```

python

import sqlite3
import pandas as pd
import streamlit as st

st.title("🌾 農業氣象查詢系統")

conn = sqlite3.connect("data.db")
df = pd.read_sql_query("SELECT * FROM weather", conn)

locations = df["location"].unique()
selected_loc = st.selectbox("選擇地區", locations)

result = df[df["location"] == selected_loc]

st.subheader(f"📍 {selected_loc} 的天氣預報")
st.table(result)

```

## 8. 系統對農業的實際應用價值

### ★ 1. 協助灌溉排程

透過每日最高、最低溫及天氣狀態，農民可判斷是否需加強或減少灌溉，避免因降雨或高溫造成作物水分失衡。



## ★ 2. 病蟲害預警

潮濕、多雨天氣容易引起病害（如真菌感染），系統提供的每日天氣能協助農民提前施作防治措施。

## ★ 3. 採收與作業規劃

農民可依預測的晴雨狀況安排：

- 採收作業
- 除草
- 施肥
- 農藥噴灑

降低因天候不佳造成的損失。

## ★ 4. 作物生長模型與智慧農業整合

資料庫格式化後，可進一步與：

- 自動化溫室控制
- AI 生長模型
- 遙測資料

整合成智慧農業決策系統。

## ★ 5. 區域農業風險評估

提供不同縣市的溫度差異，協助農民選擇適作物品種、安排播種期。

---

## ■ 9. 結論

本系統成功整合中央氣象局資料 API，使用 SQLite3 儲存並可視化天氣資訊。透過 Streamlit，使用者可快速查詢各地天氣，對農業決策具有高度實用價值。

未來可加入：

- 氣象圖表（折線圖、熱力圖）
- 自動排程每天更新
- 與農機 IoT 整合
- 氣象災害告警功能

以強化完整的智慧農業生態系統。

---

如果你需要我 加插圖片 / 幫你做 PDF / 美化格式 / 加上目錄，我也可以幫你處理！

