



Hw5

4110056030 資工三 鄭詠謙



hw5流程: 爬蟲(各地氣象) → 資料型態轉換(分析) → 資料儲存(DB) → 資料視覺化(highchart)

▼ 詢問chatgpt大致上的總步驟

SI You

- 1: 爬取中央氣象局網站 <https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001> save as "datapage.html"
- 2: 分析"datapage.html" 取得一周溫度資訊
- 3: Save as data.db in SQLite3 format HW
- 4: use highcharts to display

ChatGPT

你正在請求幫助實現一個專案，這個專案包括從中央氣象局網站獲取天氣資訊，分析並保存至SQLite3 數據庫中，然後使用Highcharts來展示這些資訊。這是一個涉及網路爬蟲、資料分析、數據庫操作和資料視覺化的完整流程。我們來一步步實現它。

首先，我們需要爬取中央氣象局網站的資料並保存為 HTML 文件。你可以使用 Python 的 requests 和 BeautifulSoup 庫來完成這一步。以下是一個範例：

```
python
import requests

# 下載網頁資料
url = "https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001"
```

```
python
import requests

# 下載網頁資料
url = "https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001"
response = requests.get(url)

# 將網頁資料保存為 HTML 文件
with open("datapage.html", "wb") as f:
    f.write(response.content)
```

接下來，我們需要解析 HTML 文件，提取我們需要的資訊。由於我們需要的是一周的溫度資訊，因此我們需要找到相關的 HTML 元素並提取其中的資訊。這可以使用 BeautifulSoup 來完成。

```
python
from bs4 import BeautifulSoup

# 讀取 HTML 文件
with open("datapage.html", "r", encoding="utf-8") as f:
    html_content = f.read()
```

```
python Copy code

from bs4 import BeautifulSoup

# 讀取 HTML 文件
with open("datapage.html", "r", encoding="utf-8") as f:
    html_content = f.read()

# 使用 BeautifulSoup 解析 HTML
soup = BeautifulSoup(html_content, "html.parser")

# 找到包含溫度資訊的元素，這裡可能需要查看網頁源碼來確定相關元素
# 提取溫度資訊並保存為一個資料結構，比如一個列表
temperatures = []

# 將溫度資訊保存至 temperatures 列表中
# ...

# 假設我們已經從網頁中成功提取到了一周的溫度資訊
# 接下來，我們將這些資訊保存至 SQLite3 數據庫中
import sqlite3

# 創建數據庫連接
conn = sqlite3.connect('data.db') ↓
```

```
# 創建數據庫連接
conn = sqlite3.connect('data.db')
c = conn.cursor()

# 創建一個表來存放溫度資訊
c.execute('''CREATE TABLE IF NOT EXISTS temperatures
            (day TEXT, temperature INTEGER)''')

# 將溫度資訊插入到數據庫中
for day, temperature in temperatures:
    c.execute("INSERT INTO temperatures (day, temperature) VALUES (?, ?)", (day, temp))

# 提交更改並關閉連接
conn.commit()
conn.close()
```

最後，我們需要使用 **Highcharts** 來展示這些資訊。你可以使用 **Highcharts** 的 JavaScript 函式庫來實現這一點。具體來說，你需要在 HTML 文件中引入 **Highcharts** 的 JavaScript 文件，並通過 JavaScript 從 **SQLite3** 數據庫中獲取資訊並顯示成圖表。這一步需要一些前端開發的知識。

以上是實現你提出的專案的大致步驟。如果你需要更多詳細的幫助，請隨時告訴我！

▼ HW 5-1: Web Scraping



爬中央氣象局的資料需要帳號才有API授權

▼ 與chatgpt的問答

You
Use Python and the requests library to send a GET request to the Central Weather Bureau (CWB) website: <https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001>.
Retrieve the HTML content of the webpage.
Save the HTML content as a file named "datapage.html".

∅

ChatGPT
Sure, here's how you can use Python and the requests library to accomplish that:

```
python
import requests

# Send a GET request to the Central Weather Bureau (CWB) website
url = "https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001"
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Retrieve the HTML content of the webpage
    html_content = response.text
```

Copy code

```

# Send a GET request to the Central Weather Bureau (CWB) website
url = "https://opendata.cwa.gov.tw/dataset/forecast/F-A0010-001"
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Retrieve the HTML content of the webpage
    html_content = response.text

    # Save the HTML content as a file named "datapage.html"
    with open("datapage.html", "w", encoding="utf-8") as f:
        f.write(html_content)
    print("HTML content saved as datapage.html")
else:
    print("Failed to retrieve HTML content. Status code:", response.status_code)

```

This code sends a GET request to the provided URL, retrieves the HTML content of the webpage, and saves it as a file named "datapage.html". If the request is successful, it prints a message indicating that the HTML content has been saved. If the request fails, it prints the status code of the response.

⬇️

▼ 開始實作

- 爬取<https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001>的氣象資料
 - 先存成datapage.html再來分析

```

import requests
import json
import sqlite3
# 存成html
# 發送 GET 請求
url = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi
response = requests.get(url)

with open("datapage.html", "w", encoding="utf-8") as file
    file.write(response.text)

# 爬蟲
def get_data():

    url = "https://opendata.cwa.gov.tw/fileapi/v1/opendat

```

```

params = {
    "Authorization": "CWA-EC120CE1-9656-4492-AA98-7AA
    "locationName": "臺中市",
}

response = requests.get(url, params=params)
if response.status_code == 200:
    data = json.loads(response.text)
    for area in range(6):
        location = data["cwaopendata"]["resources"]['
            print(data["cwaopendata"]["resources"]['resou

        Max = data["cwaopendata"]["resources"]['resou
        min = data["cwaopendata"]["resources"]['resou

        for i in range(len(Max)):

            print(f'Day {i+1}: {Max[i]["dataDate"]} M
            entry = {
                "date": Max[i]["dataDate"],
                "maxTemp": Max[i]["temperature"],
                "minTemp": min[i]["temperature"]
            }
get_data()

```

- 印出整理好的資料

北部地區

Day 1: 2024-04-23 Max Temp: 29 min Temp: 23
 Day 2: 2024-04-24 Max Temp: 26 min Temp: 22
 Day 3: 2024-04-25 Max Temp: 27 min Temp: 21
 Day 4: 2024-04-26 Max Temp: 28 min Temp: 22
 Day 5: 2024-04-27 Max Temp: 29 min Temp: 23
 Day 6: 2024-04-28 Max Temp: 29 min Temp: 22

Day 7: 2024-04-29 Max Temp: 29 min Temp: 23
中部地區

Day 1: 2024-04-23 Max Temp: 31 min Temp: 23

Day 2: 2024-04-24 Max Temp: 29 min Temp: 23

Day 3: 2024-04-25 Max Temp: 29 min Temp: 23

Day 4: 2024-04-26 Max Temp: 29 min Temp: 23

Day 5: 2024-04-27 Max Temp: 30 min Temp: 23

Day 6: 2024-04-28 Max Temp: 31 min Temp: 23

Day 7: 2024-04-29 Max Temp: 31 min Temp: 23

南部地區

Day 1: 2024-04-23 Max Temp: 34 min Temp: 25

Day 2: 2024-04-24 Max Temp: 32 min Temp: 25

Day 3: 2024-04-25 Max Temp: 31 min Temp: 24

Day 4: 2024-04-26 Max Temp: 31 min Temp: 24

Day 5: 2024-04-27 Max Temp: 29 min Temp: 24

Day 6: 2024-04-28 Max Temp: 32 min Temp: 24

Day 7: 2024-04-29 Max Temp: 33 min Temp: 24

東北部地區

Day 1: 2024-04-23 Max Temp: 28 min Temp: 23

Day 2: 2024-04-24 Max Temp: 26 min Temp: 22

Day 3: 2024-04-25 Max Temp: 26 min Temp: 21

Day 4: 2024-04-26 Max Temp: 28 min Temp: 23

Day 5: 2024-04-27 Max Temp: 28 min Temp: 23

Day 6: 2024-04-28 Max Temp: 28 min Temp: 23

Day 7: 2024-04-29 Max Temp: 28 min Temp: 24

東部地區

Day 1: 2024-04-23 Max Temp: 29 min Temp: 24

Day 2: 2024-04-24 Max Temp: 28 min Temp: 24

Day 3: 2024-04-25 Max Temp: 27 min Temp: 23

Day 4: 2024-04-26 Max Temp: 29 min Temp: 23

Day 5: 2024-04-27 Max Temp: 29 min Temp: 23

Day 6: 2024-04-28 Max Temp: 29 min Temp: 24
Day 7: 2024-04-29 Max Temp: 30 min Temp: 25
東南部地區

Day 1: 2024-04-23 Max Temp: 33 min Temp: 24
Day 2: 2024-04-24 Max Temp: 29 min Temp: 24
Day 3: 2024-04-25 Max Temp: 28 min Temp: 23
Day 4: 2024-04-26 Max Temp: 31 min Temp: 24
Day 5: 2024-04-27 Max Temp: 30 min Temp: 24
Day 6: 2024-04-28 Max Temp: 30 min Temp: 24
Day 7: 2024-04-29 Max Temp: 31 min Temp: 24

▼ HW 5-2: Data Analysis

▼ 與chatgpt的問答

You
爬取<https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=CWA-EC120CE1-9656-4492-AA98-7AAFB831FC1F&downloadType=WEB&format=JSON>並且存成“**datapage.html**”文件,然後取出未來一周的溫度資訊
< 2 / 2 > ⌂

ChatGPT
Sure, to accomplish this task, we'll first need to make a request to the provided URL to fetch the JSON data. Then we'll extract the temperature information for the upcoming week from the JSON response. Here's how you can do it using Python and the requests library:

```
python
import requests
import json

# Define the URL to fetch the JSON data
url = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=CWA-EC120CE1-9656-4492-AA98-7AAFB831FC1F&downloadType=WEB&format=JSON"

# Send a GET request to fetch the data
response = requests.get(url)
```

```

# Send a GET request to fetch the JSON data
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the JSON response
    data = response.json()

    # Save the JSON response as a file named "datapage.html"
    with open("datapage.html", "w", encoding="utf-8") as f:
        f.write(json.dumps(data, ensure_ascii=False, indent=4))

    # Extract the temperature information for the upcoming week
    # The structure of the JSON data may vary, so you'll need to inspect it to find the
    temperature_data = []

    # Assuming the temperature information is under a key named "temperature" in the
    if "temperature" in data:
        # Iterate over the temperature data for the upcoming week
        for temperature_info in data["temperature"]:
            # Extract the temperature value and add it to the temperature_data list
            temperature = temperature_info["value"]
            temperature_data.append(temperature)

```

```

# Extract the temperature value and add it to the temperature_data list
temperature = temperature_info["value"]
temperature_data.append(temperature)

# Now, temperature_data contains the temperature information for the upcoming week
print("Temperature information for the upcoming week:", temperature_data)

else:
    print("Failed to retrieve JSON data. Status code:", response.status_code)

```

In this code:

1. We define the URL to fetch the JSON data.
2. We send a GET request to fetch the JSON data from the URL.
3. If the request is successful, we parse the JSON response and save it as a file named "datapage.html".
4. We extract the temperature information for the upcoming week from the JSON response. You may need to adjust the code based on the structure of the JSON data.
5. Finally, we print the extracted temperature information.



▼ 實作資料分析&整理

- 進行分析並整理所需要寫入DB和繪製Highchart的資料欄位

```
import requests
import json
import sqlite3
# 存成html
# 發送 GET 請求
url = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi"
response = requests.get(url)

with open("datapage.html", "w", encoding="utf-8") as file
    file.write(response.text)

save = {
    "北部地區": [],
    "中部地區": [],
    "南部地區": [],
    "東北部地區": [],
    "東部地區": [],
    "東南部地區": [],
}

# 爬蟲
def get_data():

    url = "https://opendata.cwa.gov.tw/fileapi/v1/opendat
    params = {
        "Authorization": "CWA-EC120CE1-9656-4492-AA98-7AA
        "locationName": "臺中市",
    }

    response = requests.get(url, params=params)
    if response.status_code == 200:
        data = json.loads(response.text)
        for area in range(6):
            location = data["cwaopendata"]["resources"]['
                # print(data["cwaopendata"]["resources"]['res

            Max = data["cwaopendata"]["resources"]['resou
            min = data["cwaopendata"]["resources"]['resou
```

```

        for i in range(len(Max)):

            # print(f'Day {i+1}: {Max[i]["dataDate"]}')
            entry = {
                "date": Max[i]["dataDate"],
                "maxTemp": Max[i]["temperature"],
                "minTemp": min[i]["temperature"]
            }

            save[location].append(entry)
        print(save)

get_data()

```

- 印出的data格式(準備繪圖 or insertDB用)

```

{'北部地區': [{"date": '2024-04-23', 'maxTemp': '29', 'minTemp': '23'}, {"date": '2024-04-24', 'maxTemp': '26', 'minTemp': '22'}, {"date": '2024-04-25', 'maxTemp': '27', 'minTemp': '21'}, {"date": '2024-04-26', 'maxTemp': '28', 'minTemp': '22'}, {"date": '2024-04-27', 'maxTemp': '29', 'minTemp': '23'}, {"date": '2024-04-28', 'maxTemp': '29', 'minTemp': '22'}, {"date": '2024-04-29', 'maxTemp': '29', 'minTemp': '23'}], '中部地區': [{"date": '2024-04-23', 'maxTemp': '31', 'minTemp': '23'}, {"date": '2024-04-24', 'maxTemp': '29', 'minTemp': '23'}, {"date": '2024-04-25', 'maxTemp': '29', 'minTemp': '23'}, {"date": '2024-04-26', 'maxTemp': '29', 'minTemp': '23'}, {"date": '2024-04-27', 'maxTemp': '30', 'minTemp': '23'}, {"date": '2024-04-28', 'maxTemp': '31', 'minTemp': '23'}, {"date": '2024-04-29', 'maxTemp': '31', 'minTemp': '23'}], '南部地區': [{"date": '2024-04-23', 'maxTemp': '34', 'minTemp': '25'}]}

```

```
'2024-04-24', 'maxTemp': '32', 'minTemp': '25'}, {'date': '2024-04-25', 'maxTemp': '31', 'minTemp': '24'}, {'date': '2024-04-26', 'maxTemp': '31', 'minTemp': '24'}, {'date': '2024-04-27', 'maxTemp': '29', 'minTemp': '24'}, {'date': '2024-04-28', 'maxTemp': '32', 'minTemp': '24'}, {"date": "2024-04-29", "maxTemp": "33", "minTemp": "24"}], '東北部地區': [{"date": "2024-04-23", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "26", "minTemp": "22"}, {"date": "2024-04-25", "maxTemp": "26", "minTemp": "21"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "24"}], '東部地區': [{"date": "2024-04-23", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-24", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-25", "maxTemp": "27", "minTemp": "23"}, {"date": "2024-04-26", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-27", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "30", "minTemp": "25"}], '東南部地區': [{"date": "2024-04-23", "maxTemp": "33", "minTemp": "24"}, {"date": "2024-04-24", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-25", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-26", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "30", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "30", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "31", "minTemp": "24"}]}
```

▼ HW 5-3: Data Storage

▼ 與chatgpt的問答

You
幫我把這幾個變數資料寫進sqlite:Max[i]["dataDate"] ,Max[i]["temperature"] ,min[i]["temperature"]
,data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]["weatherForecasts"]
["location"][area]["locationName"]

程式:

```
data = json.loads(response.text)
for area in range(6):
    print(data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]
          ["weatherForecasts"]["location"][area]["locationName"])

    Max = data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]
          ["weatherForecasts"]["location"][area]["weatherElements"]["MaxT"]["daily"]
    min = data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]
          ["weatherForecasts"]["location"][area]["weatherElements"]["MinT"]["daily"]

for i in range(len(Max)):

    print(f'Day {i+1}: {Max[i]["dataDate"]} Max Temp: {Max[i]["temperature"]} min Temp: {min[i]
        ["temperature"]}')
```

ChatGPT
你需要使用 Python 中的 SQLite 模块来创建数据库并将数据插入其中。下面是你的代码的扩展版本，用于将数据写入 SQLite 数据库：

```
python
import sqlite3
import json

# 解析 JSON 数据
data = json.loads(response.text)

# 连接到 SQLite 数据库
conn = sqlite3.connect('weather_data.db')
c = conn.cursor()

# 创建表格
c.execute('''CREATE TABLE IF NOT EXISTS weather_forecast
            (id INTEGER PRIMARY KEY,
             locationName TEXT,
             ↓
             dataDate TEXT,
```

```
# 循环处理每个地区的天气数据
for area in range(6):
    location_name = data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]
    Max = data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]
    min = data["cwaopendata"]["resources"]["resource"]["data"]["agrWeatherForecasts"]

# 循环处理每个日期的最高和最低温度数据
for i in range(len(Max)):
    data_date = Max[i]["dataDate"]
    max_temp = Max[i]["temperature"]
    min_temp = min[i]["temperature"]

# 插入数据到表格
c.execute('''INSERT INTO weather_forecast (locationName, dataDate, maxTemperature, minTemperature)
VALUES (?, ?, ?, ?)''', (location_name, data_date, max_temp, min_temp))

# 提交更改并关闭连接
conn.commit()
conn.close()
```

此代码将数据写入名为 `weather_data.db` 的 SQLite 数据库中，其中包含一个名为

```
# 提交更改并关闭连接
conn.commit()
conn.close()
```

此代码将数据写入名为 `weather_data.db` 的 SQLite 数据库中，其中包含一个名为 `weather_forecast` 的表格，用于存储天气预报数据。

◀ ▶ ⌂

▼ 實作insert data to DB

- 將整理好的資料存進DB

```
import requests
import json
import sqlite3
# 存成html
# 發送 GET 請求
url = "https://opendata.cwa.gov.tw/fileapi/v1/opendataapi"
response = requests.get(url)

with open("datapage.html", "w", encoding="utf-8") as file
    file.write(response.text)
```

```
save = {
    "北部地區": [],
    "中部地區": [],
    "南部地區": [],
    "東北部地區": [],
    "東部地區": [],
    "東南部地區": []
}

# 爬蟲
def get_data():

    url = "https://opendata.cwa.gov.tw/fileapi/v1/opendat
    params = {
        "Authorization": "CWA-EC120CE1-9656-4492-AA98-7AA
        "locationName": "臺中市",
    }

    response = requests.get(url, params=params)
    if response.status_code == 200:
        data = json.loads(response.text)
        for area in range(6):
            location = data["cwaopendata"]["resources"]['
                # print(data["cwaopendata"]["resources"]['resou

            Max = data["cwaopendata"]["resources"]['resou
            min = data["cwaopendata"]["resources"]['resou

            for i in range(len(Max)):

                # print(f'Day {i+1}: {Max[i]["dataDate"]}')
                entry = {
                    "date": Max[i]["dataDate"],
                    "maxTemp": Max[i]["temperature"],
                    "minTemp": min[i]["temperature"]
                }

```

```
        save[location].append(entry)
print(save)

# 存到DB
# 連接SQLite
conn = sqlite3.connect('weather_data.db')
c = conn.cursor()

# 建立table
c.execute('''CREATE TABLE IF NOT EXISTS weather_forec
            (id INTEGER PRIMARY KEY,
             locationName TEXT,
             dataDate TEXT,
             maxTemperature REAL,
             minTemperature REAL)'''')

# loop處理
for area in range(6):
    location_name = data["cwaopendata"]["resources"][area]
    Max = data["cwaopendata"]["resources"]['resource'][0]
    min = data["cwaopendata"]["resources"]['resource'][1]

    # 每個地區各7天(未來一周)
    for i in range(len(Max)):
        data_date = Max[i]["dataDate"]
        max_temp = Max[i]["temperature"]
        min_temp = min[i]["temperature"]

        # insert DB
        c.execute('''INSERT INTO weather_forecast (locationName, dataDate, maxTemperature, minTemperature)
VALUES (?, ?, ?, ?)''', (location_name, data_date, max_temp, min_temp))

    # 提交並關閉
    conn.commit()
    conn.close()
```

get_data()

- 用sqlite viewer檢查DB是否有正確insert氣象資料

weather_data.db						
		Reset Filters Records: 42				
Tables (1)		weather_forecast	id	locationName	dataDate	maxTemperature
Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	北部地區	2024-04-23	29	23	
2	2	北部地區	2024-04-24	26	22	
3	3	北部地區	2024-04-25	27	21	
4	4	北部地區	2024-04-26	28	22	
5	5	北部地區	2024-04-27	29	23	
6	6	北部地區	2024-04-28	29	22	
7	7	北部地區	2024-04-29	29	23	
8	8	中部地區	2024-04-23	31	23	
9	9	中部地區	2024-04-24	29	23	
10	10	中部地區	2024-04-25	29	23	
11	11	中部地區	2024-04-26	29	23	
12	12	中部地區	2024-04-27	30	23	
13	13	中部地區	2024-04-28	31	23	
14	14	中部地區	2024-04-29	31	23	
15	15	南部地區	2024-04-23	34	25	
16	16	南部地區	2024-04-24	32	25	
17	17	南部地區	2024-04-25	31	24	
18	18	南部地區	2024-04-26	31	24	
19	19	南部地區	2024-04-27	29	24	
20	20	南部地區	2024-04-28	32	24	
21	21	南部地區	2024-04-29	33	24	

weather_data.db						
		Reset Filters Records: 42				
Tables (1)		weather_forecast	id	locationName	dataDate	maxTemperature
Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
22	22	東北部地區	2024-04-23	28	23	
23	23	東北部地區	2024-04-24	26	22	
24	24	東北部地區	2024-04-25	26	21	
25	25	東北部地區	2024-04-26	28	23	
26	26	東北部地區	2024-04-27	28	23	
27	27	東北部地區	2024-04-28	28	23	
28	28	東北部地區	2024-04-29	28	24	
29	29	東部地區	2024-04-23	29	24	
30	30	東部地區	2024-04-24	28	24	
31	31	東部地區	2024-04-25	27	23	
32	32	東部地區	2024-04-26	29	23	
33	33	東部地區	2024-04-27	29	23	
34	34	東部地區	2024-04-28	29	24	
35	35	東部地區	2024-04-29	30	25	
36	36	東南部地區	2024-04-23	33	24	
37	37	東南部地區	2024-04-24	29	24	
38	38	東南部地區	2024-04-25	28	23	
39	39	東南部地區	2024-04-26	31	24	
40	40	東南部地區	2024-04-27	30	24	
41	41	東南部地區	2024-04-28	30	24	
42	42	東南部地區	2024-04-29	31	24	

▼ HW 5-4: Data Visualization

▼ 與chatgpt的問答

SI You

這個格式的資料我要用highchart畫出maxTemp和minTemp,要有下拉式選單可以選擇要顯示哪一個地區的資料chart:

```
data = {'北部地區': [{"date": "2024-04-23", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "26", "minTemp": "22"}, {"date": "2024-04-25", "maxTemp": "27", "minTemp": "21"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "22"}, {"date": "2024-04-27", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "29", "minTemp": "22"}, {"date": "2024-04-29", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-25", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-26", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-27", "maxTemp": "30", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-29", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "34", "minTemp": "25"}, {"date": "2024-04-24", "maxTemp": "32", "minTemp": "25"}, {"date": "2024-04-25", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-26", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "32", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "26", "minTemp": "22"}, {"date": "2024-04-25", "maxTemp": "26", "minTemp": "21"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-24", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-25", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "24"}]
```

```
'minTemp': '24}], [{"date": "2024-04-29", "maxTemp": "30", "minTemp": "25"}], '東南部地區': [{"date": "2024-04-23", "maxTemp": "33", "minTemp": "24"}, {"date": "2024-04-24", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-25", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-26", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "30", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "31", "minTemp": "24"}]}
```

ChatGPT

你可以使用以下的 Python 代碼來生成 Highcharts 圖表，包括下拉式選單以選擇不同地區的資料。這個例子使用了 Python 的 Flask 框架作為後端服務，並使用 Jinja2 模板引擎來動態生成 HTML。

```
python
from flask import Flask, render_template, request, jsonify

app = Flask(__name__)

data = {
    '北部地區': [
        {"date": "2024-04-23", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "26", "minTemp": "22"}, {"date": "2024-04-25", "maxTemp": "27", "minTemp": "21"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "22"}, {"date": "2024-04-27", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "29", "minTemp": "22"}, {"date": "2024-04-29", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-25", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-26", "maxTemp": "29", "minTemp": "23"}, {"date": "2024-04-27", "maxTemp": "30", "minTemp": "23"}, {"date": "2024-04-28", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-29", "maxTemp": "31", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "34", "minTemp": "25"}, {"date": "2024-04-24", "maxTemp": "32", "minTemp": "25"}, {"date": "2024-04-25", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-26", "maxTemp": "31", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "32", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-24", "maxTemp": "26", "minTemp": "22"}, {"date": "2024-04-25", "maxTemp": "26", "minTemp": "21"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "22"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "22"}, {"date": "2024-04-28", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "23"}, {"date": "2024-04-23", "maxTemp": "29", "minTemp": "24"}, {"date": "2024-04-24", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-25", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-26", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-27", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-28", "maxTemp": "28", "minTemp": "24"}, {"date": "2024-04-29", "maxTemp": "28", "minTemp": "24"}]
```

```

        {'date': '2024-04-25', 'maxTemp': '27', 'minTemp': '21'},
        {'date': '2024-04-26', 'maxTemp': '28', 'minTemp': '22'},
        {'date': '2024-04-27', 'maxTemp': '29', 'minTemp': '23'},
        {'date': '2024-04-28', 'maxTemp': '29', 'minTemp': '22'},
        {'date': '2024-04-29', 'maxTemp': '29', 'minTemp': '23'}
    ],
    '中部地區': [
        {'date': '2024-04-23', 'maxTemp': '31', 'minTemp': '23'},
        {'date': '2024-04-24', 'maxTemp': '29', 'minTemp': '23'},
        {'date': '2024-04-25', 'maxTemp': '29', 'minTemp': '23'},
        {'date': '2024-04-26', 'maxTemp': '29', 'minTemp': '23'},
        {'date': '2024-04-27', 'maxTemp': '30', 'minTemp': '23'},
        {'date': '2024-04-28', 'maxTemp': '31', 'minTemp': '23'},
        {'date': '2024-04-29', 'maxTemp': '31', 'minTemp': '23'}
    ],
    # 省略其餘地區資料...
}

@app.route('/')
def index():
    regions = list(data.keys())
    return render_template('index.html', regions=regions)

```

```

@app.route('/')
def index():
    regions = list(data.keys())
    return render_template('index.html', regions=regions)

@app.route('/get_chart_data', methods=['POST'])
def get_chart_data():
    region = request.json['region']
    region_data = data.get(region, [])

    dates = [item['date'] for item in region_data]
    max_temps = [int(item['maxTemp']) for item in region_data]
    min_temps = [int(item['minTemp']) for item in region_data]

    return jsonify({'dates': dates, 'maxTemps': max_temps, 'minTemps': min_temps})

if __name__ == '__main__':
    app.run(debug=True)

```

然後，在同一個目錄下創建一個名為 `temp` 的文件夾，在這個文件夾中創建一個名為 `index.html` 的 HTML 文件，並使用以下代碼填充：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Weather Chart</title>
    <script src="https://code.highcharts.com/highcharts.js"></script>
</head>
<body>
    <h1>Weather Chart</h1>
    <label for="region">Select Region:</label>
    <select id="region">
        {% for region in regions %}
            <option value="{{ region }}>{{ region }}</option>
        {% endfor %}
    </select>

    <div id="chartContainer" style="width: 600px; height: 400px;"></div>

    <script>
        document.addEventListener('DOMContentLoaded', function () {
            var regionSelect = document.getElementById('region');


```

```
        document.addEventListener('DOMContentLoaded', function () {
            var regionSelect = document.getElementById('region');
            var chartContainer = document.getElementById('chartContainer');

            regionSelect.addEventListener('change', function () {
                var selectedRegion = regionSelect.value;
                fetch('/get_chart_data', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json',
                        'Accept': 'application/json'
                    },
                    body: JSON.stringify({ region: selectedRegion })
                })
                .then(response => response.json())
                .then(data => {
                    var chartData = {
                        chart: {
                            type: 'line'
                        },
                        title: {
                            text: 'Weather Chart'
                        },

```

```
    text: 'Weather Chart'
  },
  xAxis: {
    categories: data.dates
  },
  yAxis: {
    title: {
      text: 'Temperature (°C)'
    }
  },
  series: [{
    name: 'Max Temperature',
    data: data.maxTemps
  }, {
    name: 'Min Temperature',
    data: data.minTemps
  }]
};

Highcharts.chart(chartContainer, chartData);
})
.catch(error => console.error('Error:', error));
});
```

```
}, {
  name: 'Min Temperature',
  data: data.minTemps
}]
};

Highcharts.chart(chartContainer, chartData);
})
.catch(error => console.error('Error:', error));
});

// Trigger initial chart rendering
regionSelect.dispatchEvent(new Event('change'));
});
</script>
</body>
</html>
```

這個代碼將允許你在瀏覽器中選擇不同的地區，並顯示該地區的最高和最低溫度數據。

▼ 實作繪製highchart並顯示在網頁上以實現視覺化

- 用Flask建立前端網頁

data用爬蟲取得的資料

```
# <app.py>
from flask import Flask, render_template, request, jsonify
from weather import save

app = Flask(__name__)

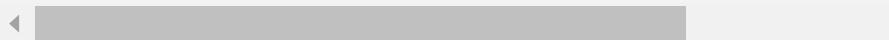
data = save
@app.route('/')
def index():
    regions = list(data.keys())
    return render_template('index.html', regions=regions)

@app.route('/get_chart_data', methods=['POST'])
def get_chart_data():
    region = request.json['region']
    region_data = data.get(region, [])

    dates = [item['date'] for item in region_data]
    max_temps = [int(item['maxTemp']) for item in region_data]
    min_temps = [int(item['minTemp']) for item in region_data]

    return jsonify({'dates': dates, 'maxTemps': max_temps, 'minTemps': min_temps})

if __name__ == '__main__':
    app.run(debug=True)
```



- 網頁模板html(置於templates file中)

```
<!-- <index.html> -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Weather Chart</title>
```

```
<script src="https://code.highcharts.com/highcharts.js">
</head>
<body style="text-align: center;">
    <h1>Weather Chart</h1>
    <label for="region">Select Region:</label>
    <select id="region">
        {% for region in regions %}
            <option value="{{ region }}>{{ region }}</option>
        {% endfor %}
    </select>

    <div id="chartContainer" style="width: 80%; margin: 0 auto;">
        <script>
            document.addEventListener('DOMContentLoaded', function() {
                var regionSelect = document.getElementById('region');
                var chartContainer = document.getElementById('chartContainer');

                regionSelect.addEventListener('change', function() {
                    var selectedRegion = regionSelect.value;
                    fetch('/get_chart_data', {
                        method: 'POST',
                        headers: {
                            'Content-Type': 'application/json',
                            'Accept': 'application/json'
                        },
                        body: JSON.stringify({ region: selectedRegion })
                    })
                    .then(response => response.json())
                    .then(data => {
                        var chartData = {
                            chart: {
                                type: 'line'
                            },
                            title: {
                                text: 'Weather Chart'
                            },
                            xAxis: {

```

```

        categories: data.dates
    },
    yAxis: {
        title: {
            text: 'Temperature (°C)'
        }
    },
    series: [{
        name: 'Max Temperature',
        data: data.maxTemps
    }, {
        name: 'Min Temperature',
        data: data.minTemps
    }]
};

Highcharts.chart(chartContainer, chartOptions)
    .catch(error => console.error('Error:', error));
}

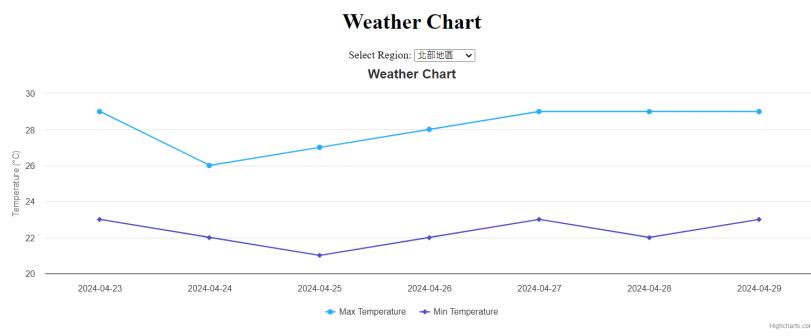
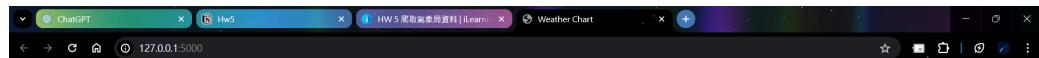
// Trigger initial chart rendering
regionSelect.dispatchEvent(new Event('change'));
</script>
</body>
</html>

```

▼ show出圖形

| 總共有6個區域

- 北部

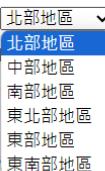


- 下拉式選單

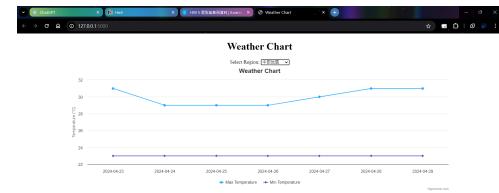
Weather Chart

Select Region: 北部地區

Weather

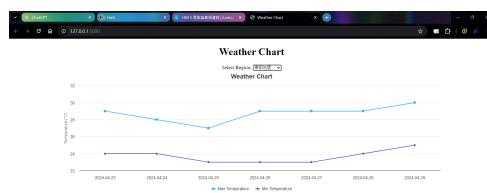


- 中部

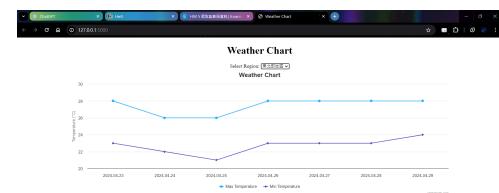


- 東北部

- 東部



- 南部



- 東南部

