物聯網作業5

HW5-1

question

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".

解決辦法

使用它（氣象局網站）給的api解決
like below
url = 'https://opendata.cwb.gov.tw/fileapi/v1/opendataapi/F-A0010-001?Authorization=here is where to place&downloadType=WEB&format=JSON'
注意 因為氣象局升格為氣象屬所以網域不一樣了！！
cwb -> cwa

```
import requests
def dowload(url, save_path):
        headers = {
        'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36'
        }
        response = requests.get(url)
        if response.status_code == 200:
                with open(save_path, 'wb') as f:
                        f.write(response.content)
                        print("Image downloaded successfully!")
        else:
                print("Failed to download image.")
        return response

url = 'https://opendata.cwa.gov.tw/fileapi/v1/opendataapi/F-A0010-001?
Authorization=???????&downloadType=WEB&format=JSON'

r = dowload(url,'datapage.html')
r.encoding = 'utf-8'
print(r.text)
```

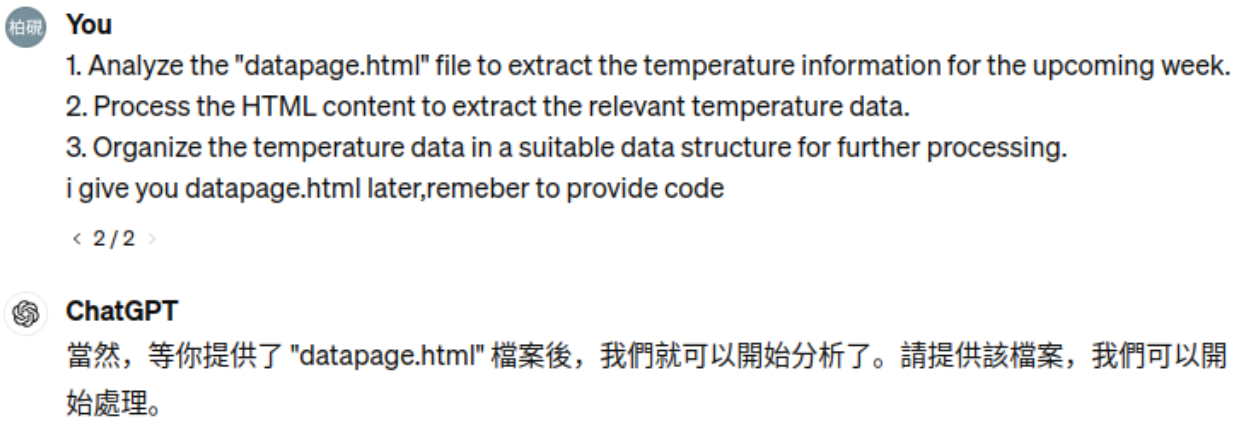
```
Image downloaded successfully!
{
 "cwaopendata": {
  "@xmlns": "urn:cwa:gov:tw:cwacommon:0.1",
  "identifier": "472c7047-eb48-4aa1-8fc0-9be46a234889",
  "datasetName": "一週農業氣象預報",
  "dataid": "F-A0010-001",
  "sender": "shchiang@cwa.gov.tw",
  "sent": "2024-04-26T18:15:02+08:00",
  "status": "Actual",
  "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"
       },
...
    }
   }
  }
 }
}
```


HW5-2

question

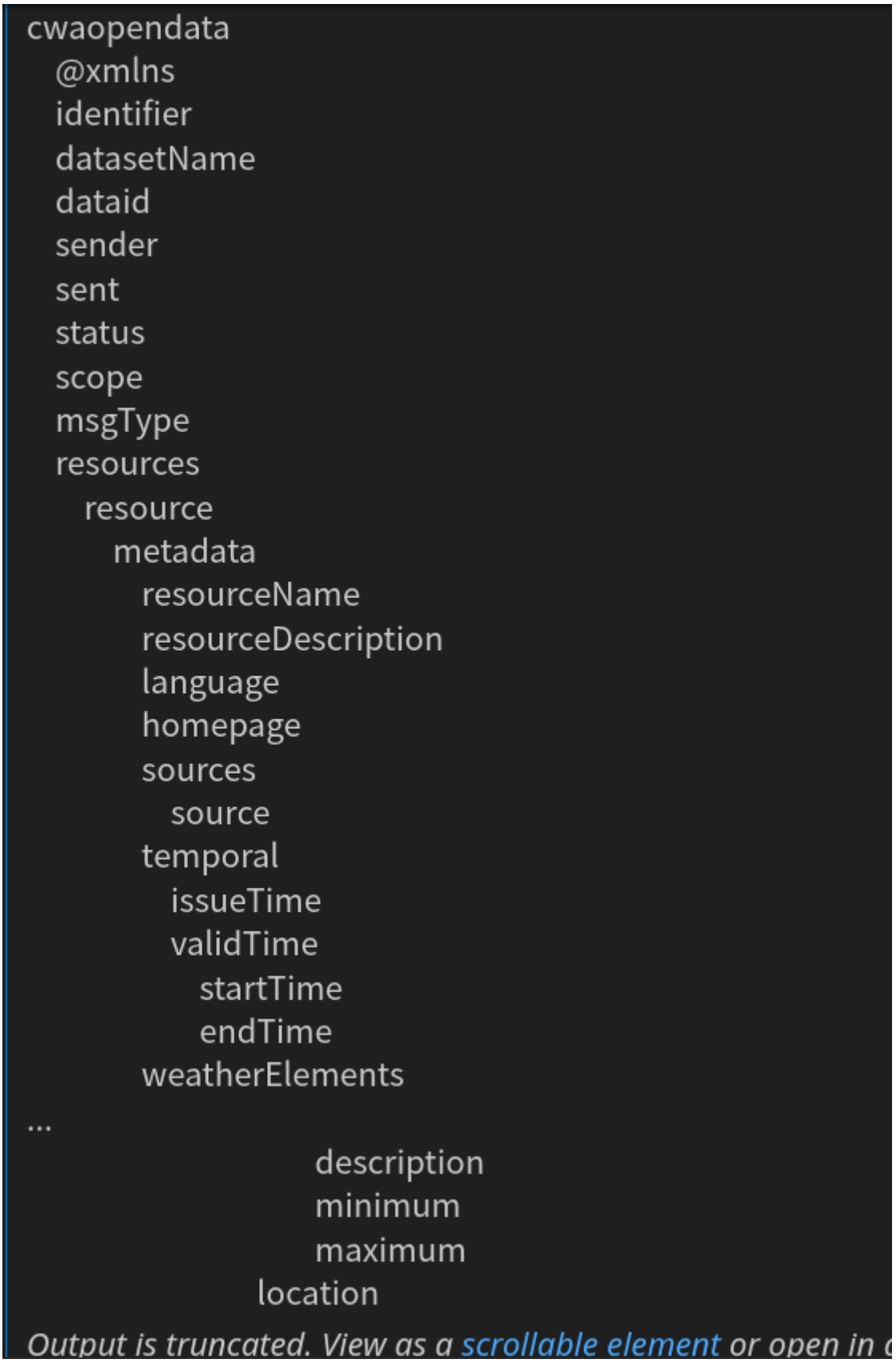
1. Analyze the "datapage.html" file to extract the temperature information for the upcoming week.
2. Process the HTML content to extract the relevant temperature data.
3. Organize the temperature data in a suitable data structure for further processing.

咒言



分析結構

```
import json
# 讀取 JSON 檔案
with open("datapage.html", "r", encoding="utf-8") as f:
        json_data = json.load(f)

# 定義一個遞迴函式，用於印出 JSON 結構
def print_json_structure(data, indent=0):
        for key, value in data.items():
                print(" " * indent + key)
                if isinstance(value, dict):
                        print_json_structure(value, indent + 4)
# 列印 JSON 結構
print_json_structure(json_data)
```



在擷取資料

```
import json

# 讀取 JSON 檔案
with open("datapage.json", "r", encoding="utf-8") as f:
        json_data = f.read()
```

```
# 解析 JSON 資料
parsed_data = json.loads(json_data)
json_data1 = parsed_data["cwaopendata"]["resources"]["resource"]["data"]

# 提取氣溫資料
for location in json_data1["agrWeatherForecasts"]["weatherForecasts"]
["location"]:
        location_name = location["locationName"]
        wx_daily = location["weatherElements"]["Wx"]["daily"]
        MaxT_daily = location["weatherElements"]["MaxT"]["daily"] # 提取最
高氣溫資料
        MinT_daily = location["weatherElements"]["MinT"]["daily"] # 提取最
低氣溫資料
        print(f"地區: {location_name}")
        for wx_data, max_data, min_data in zip(wx_daily, MaxT_daily,
MinT_daily):
                date = wx_data['dataDate']
                weather = wx_data['weather']
                weather_id = wx_data['weatherid']
                max_temp = max_data['temperature'] # 最高氣溫
                min_temp = min_data['temperature'] # 最低氣溫
                print(f"日期: {date}, 天氣: {weather} (天氣編號:
{weather_id}), 最高溫: {max_temp}°C, 最低溫: {min_temp}°C")
        print()
```

```
地區：北部地區
日期：2024-04-27, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：29°C, 最低溫：22°C
日期：2024-04-28, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：30°C, 最低溫：23°C
日期：2024-04-29, 天氣：多雲時晴 (天氣編號：3), 最高溫：31°C, 最低溫：23°C
日期：2024-04-30, 天氣：多雲短暫陣雨 (天氣編號：8), 最高溫：31°C, 最低溫：24°C
日期：2024-05-01, 天氣：陰短暫陣雨或雷雨 (天氣編號：18), 最高溫：28°C, 最低溫：22°C
日期：2024-05-02, 天氣：多雲時陰短暫陣雨或雷雨 (天氣編號：16), 最高溫：25°C, 最低溫：21°C
日期：2024-05-03, 天氣：晴時多雲 (天氣編號：2), 最高溫：27°C, 最低溫：21°C

地區：中部地區
日期：2024-04-27, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：30°C, 最低溫：21°C
日期：2024-04-28, 天氣：多雲短暫陣雨或雷雨 (天氣編號：15), 最高溫：32°C, 最低溫：22°C
日期：2024-04-29, 天氣：多雲短暫陣雨 (天氣編號：8), 最高溫：33°C, 最低溫：23°C
日期：2024-04-30, 天氣：多雲短暫陣雨 (天氣編號：8), 最高溫：33°C, 最低溫：24°C
日期：2024-05-01, 天氣：多雲時陰短暫陣雨或雷雨 (天氣編號：16), 最高溫：30°C, 最低溫：24°C
日期：2024-05-02, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：28°C, 最低溫：23°C
日期：2024-05-03, 天氣：多雲時晴 (天氣編號：3), 最高溫：31°C, 最低溫：22°C

地區：南部地區
日期：2024-04-27, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：31°C, 最低溫：22°C
日期：2024-04-28, 天氣：多雲短暫陣雨或雷雨 (天氣編號：15), 最高溫：33°C, 最低溫：24°C
日期：2024-04-29, 天氣：晴時多雲 (天氣編號：2), 最高溫：33°C, 最低溫：25°C
日期：2024-04-30, 天氣：多雲短暫陣雨 (天氣編號：8), 最高溫：33°C, 最低溫：25°C
日期：2024-05-01, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：31°C, 最低溫：25°C
日期：2024-05-02, 天氣：陰時多雲短暫陣雨或雷雨 (天氣編號：17), 最高溫：29°C, 最低溫：24°C
```

HW 5-3

question

HW 5-3: Data Storage

1. Create a SQLite database named "data.db" using Python and the sqlite3 library.
2. Define an appropriate table structure to store the temperature data.
3. Insert the temperature data into the SQLite database.

咒語(就是上次的code加上這次的question)

```python
import json

# 讀取 JSON 檔案
with open("datapage.json", "r", encoding="utf-8") as f:
        json_data = f.read()

# 解析 JSON 資料
parsed_data = json.loads(json_data)
json_data1 = parsed_data["cwaopendata"]["resources"]["resource"]["data"]

# 提取氣溫資料
```

```
for location in json_data1["agrWeatherForecasts"]["weatherForecasts"]
["location"]:
        location_name = location["locationName"]
        wx_daily = location["weatherElements"]["Wx"]["daily"]
        MaxT_daily = location["weatherElements"]["MaxT"]["daily"] # 提取最
高氣溫資料
        MinT_daily = location["weatherElements"]["MinT"]["daily"] # 提取最
低氣溫資料
        print(f"地區: {location_name}")
        for wx_data, max_data, min_data in zip(wx_daily, MaxT_daily,
MinT_daily):
                date = wx_data['dataDate']
                weather = wx_data['weather']
                weather_id = wx_data['weatherid']
                max_temp = max_data['temperature'] # 最高氣溫
                min_temp = min_data['temperature'] # 最低氣溫
                print(f"日期: {date}, 天氣: {weather} (天氣編號:
{weather_id}), 最高溫: {max_temp}°C, 最低溫: {min_temp}°C")
        print()
```

1. Create a SQLite database named "data.db" using Python and the sqlite3 library.
2. Define an appropriate table structure to store the temperature data.
3. Insert the temperature data into the SQLite database.

Result

```
import json
import sqlite3

# 讀取 JSON 檔案
with open("datapage.html", "r", encoding="utf-8") as f:
        json_data = f.read()
# 解析 JSON 資料
parsed_data = json.loads(json_data)
json_data1 = parsed_data["cwaopendata"]["resources"]["resource"]["data"]

# 連接到 SQLite 資料庫，如果不存在將會自動建立一個新的
conn = sqlite3.connect('data.db')

# 創建一個游標
cursor = conn.cursor()
# 創建氣溫資料表
cursor.execute('''CREATE TABLE IF NOT EXISTS temperature (
id INTEGER PRIMARY KEY,
location TEXT,
date TEXT,
weather TEXT,
weather_id TEXT,
```

```
max_temp INTEGER,
min_temp INTEGER
)''')

# 提取氣溫資料並插入資料庫
for location in json_data1["agrWeatherForecasts"]["weatherForecasts"]
["location"]:
        location_name = location["locationName"]
        wx_daily = location["weatherElements"]["Wx"]["daily"]
        MaxT_daily = location["weatherElements"]["MaxT"]["daily"] # 提取最
高氣溫資料
        MinT_daily = location["weatherElements"]["MinT"]["daily"] # 提取最
低氣溫資料
                for wx_data, max_data, min_data in zip(wx_daily,
MaxT_daily, MinT_daily):
                        date = wx_data['dataDate']
                        weather = wx_data['weather']
                        weather_id = wx_data['weatherid']
                        max_temp = max_data['temperature'] # 最高氣溫
                        min_temp = min_data['temperature'] # 最低氣溫

# 插入資料到 temperature 資料表
cursor.execute("INSERT INTO temperature (location, date, weather,
weather_id, max_temp, min_temp) VALUES (?, ?, ?, ?, ?, ?)",
(location_name, date, weather, weather_id, max_temp, min_temp))

# 確認並儲存變更
conn.commit()
# 關閉連接
conn.close()

print("氣溫資料已成功插入到 SQLite 資料庫中。")
```

```
氣溫資料已成功插入到 SQLite 資料庫中。
```

	id	location	date	weather	weather_id	max_temp	min_temp
	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	北部地區	2024-04-27	陰時多雲短暫陣雨或...	17	29	22
2	2	北部地區	2024-04-28	陰時多雲短暫陣雨或...	17	30	23
3	3	北部地區	2024-04-29	多雲時晴	3	31	23
4	4	北部地區	2024-04-30	多雲短暫陣雨	8	31	24
5	5	北部地區	2024-05-01	陰短暫陣雨或雷雨	18	28	22
6	6	北部地區	2024-05-02	多雲時陰短暫陣雨或...	16	25	21
7	7	北部地區	2024-05-03	晴時多雲	2	27	21
8	8	中部地區	2024-04-27	陰時多雲短暫陣雨或...	17	30	21
9	9	中部地區	2024-04-28	多雲短暫陣雨或雷雨	15	32	22
10	10	中部地區	2024-04-29	多雲短暫陣雨	8	33	23
11	11	中部地區	2024-04-30	多雲短暫陣雨	8	33	24
12	12	中部地區	2024-05-01	多雲時陰短暫陣雨或...	16	30	24
13	13	中部地區	2024-05-02	陰時多雲短暫陣雨或...	17	28	23
14	14	中部地區	2024-05-03	多雲時晴	3	31	22
15	15	南部地區	2024-04-27	陰時多雲短暫陣雨或...	17	31	22
16	16	南部地區	2024-04-28	多雲短暫陣雨或雷雨	15	33	24
17	17	南部地區	2024-04-29	晴時多雲	2	33	25
18	18	南部地區	2024-04-30	多雲短暫陣雨	8	33	25
19	19	南部地區	2024-05-01	陰時多雲短暫陣雨或...	17	31	25
20	20	南部地區	2024-05-02	陰時多雲短暫陣雨或...	17	29	24
21	21	南部地區	2024-05-03	多雲時晴	3	31	23
22	22	東北部地區	2024-04-27	陰時多雲短暫陣雨或...	17	28	22
23	23	東北部地區	2024-04-28	陰時多雲短暫陣雨或...	17	29	23

HW5-4

question

HW 5-4: Data Visualization

1. Use Highcharts, a JavaScript charting library, to create interactive charts for displaying the temperature data.
2. integrate the Highcharts library into a HTML file.
3. Fetch the temperature data from the SQLite database using JavaScript.
4. Use the fetched data to populate the Highcharts chart with the temperature information.
5. Display the chart on a webpage for visualization.

咒語

HW 5-4: Data Visualization

Use Highcharts, a JavaScript charting library, to create interactive charts for displaying the temperature data.
Integrate the Highcharts library into a HTML file.
Fetch the temperature data from the SQLite database using JavaScript.
Use the fetched data to populate the Highcharts chart with the temperature information.
Display the chart on a webpage for visualization.

使用 flask & flask.g 用於處理資料存取
並且使用下拉是選單選擇不同的地區

結果

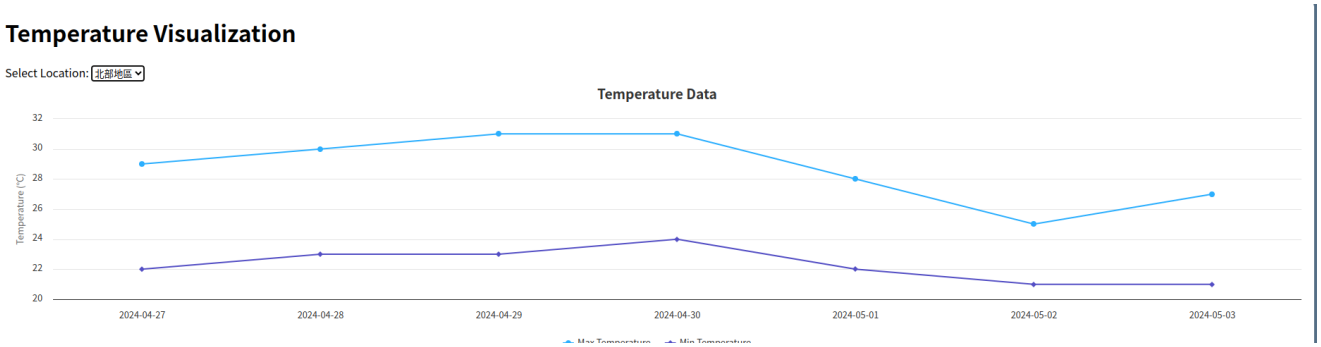
app.py

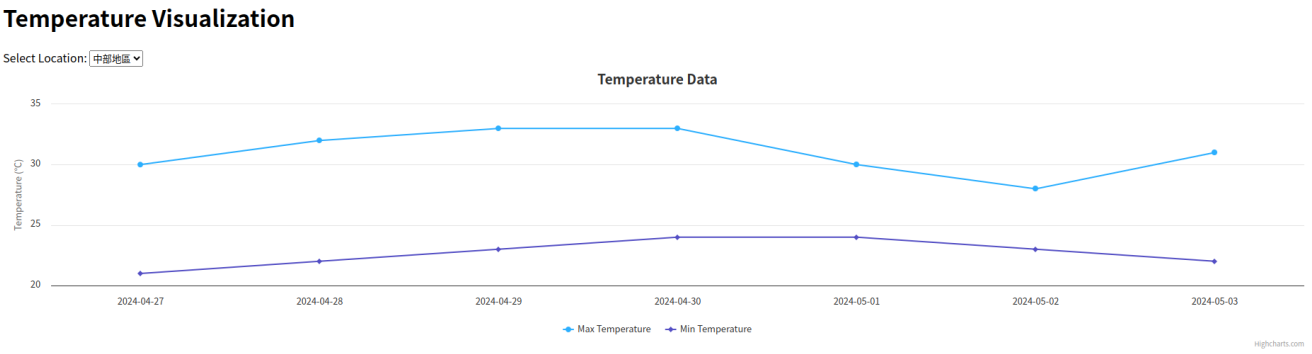
```
from flask import Flask, render_template, g, jsonify,send_file
import sqlite3
import os
app = Flask(__name__)

DATABASE = 'data.db'
# connect to "mydatabase.db"
def get_db():
        if 'db' not in g:
                g.db = sqlite3.connect('data.db')
                g.cursor = g.db.cursor()
        return g.db, g.cursor

# close_db when teardown Flask's application context
def close_db(db):
        if db in g:
                g.db.close()

@app.route('/<path:filename>')
def serve_static(filename):
        root_dir = os.path.abspath(os.path.dirname(__file__))
        static_dir = os.path.join(root_dir, 'template')
        return send_file(os.path.join(static_dir, filename))

# API 路由: 取得指定地區的溫度資料
@app.route('/temperature/<location>')
def temperature(location):
        db,cursor =get_db()
        cursor.execute("SELECT date, max_temp, min_temp FROM temperature
WHERE location=?", (location,))
        data = cursor.fetchall()
        result = [{'date': row[0], 'max_temp': row[1], 'min_temp': row[2]}
for row in data]
        print(result)
        close_db(db)
        return jsonify(result)

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

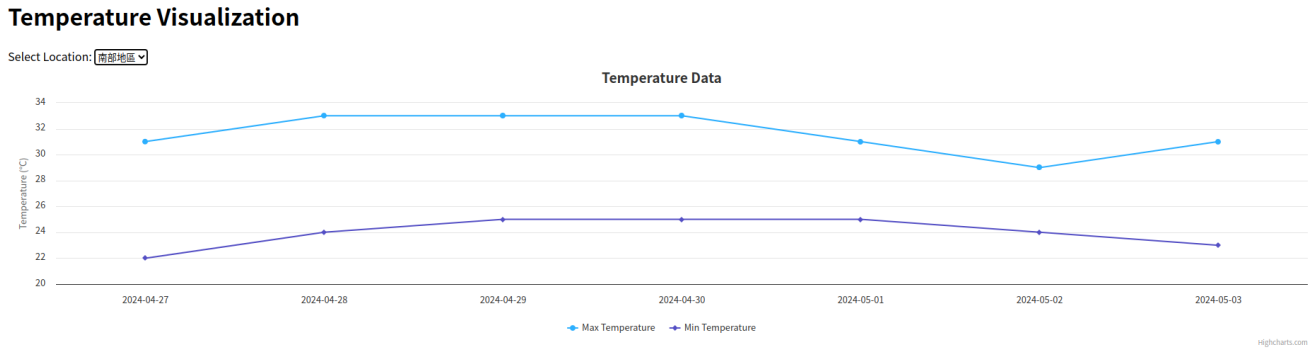
index.html

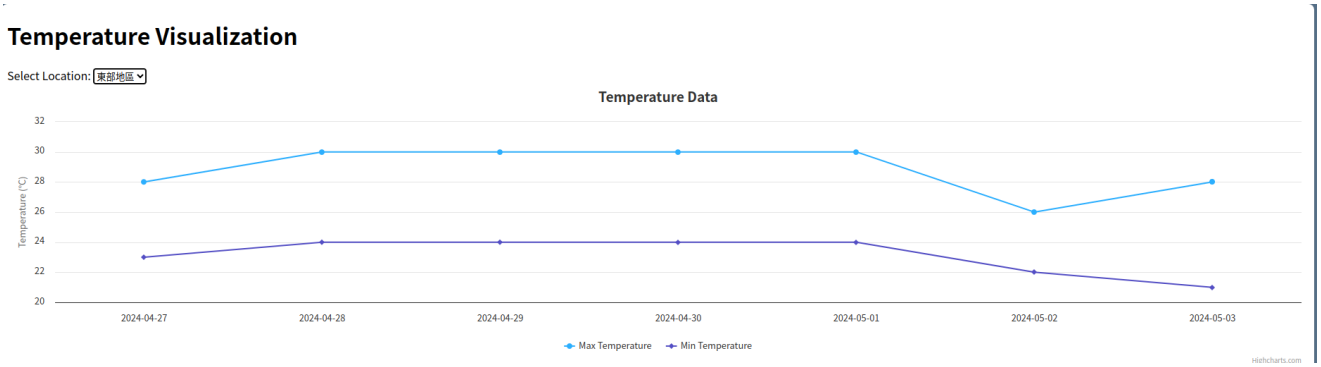
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Temperature Visualization</title>
<script src="https://code.highcharts.com/highcharts.js"></script>
</head>
<body>
<h1>Temperature Visualization</h1>
<label for="location-select">Select Location:</label>
<select id="location-select" onchange="fetchTemperatureData()">
<option value="北部地區">北部地區</option>
<option value="中部地區">中部地區</option>
<option value="南部地區">南部地區</option>
<option value="東部地區">東部地區</option>
<!-- Add more options for other regions -->
</select>
<div id="temperature-chart"></div>
<script>
function fetchTemperatureData() {
var location = document.getElementById('location-select').value;
fetch('/temperature/' + location)
.then(response => response.json())
.then(data => {
console.log(data);
renderTemperatureChart(data);
})
.catch(error => console.error('Error fetching temperature data:', error));
}

function renderTemperatureChart(data) {
Highcharts.chart('temperature-chart', {
title: {
text: 'Temperature Data'
},
xAxis: {
categories: data.map(item => item.date)
},
yAxis: {
title: {
text: 'Temperature (°C)'
}
},
series: [{
name: 'Max Temperature',
data: data.map(item => parseFloat(item.max_temp))
}, {
name: 'Min Temperature',
```

```
data: data.map(item => parseFloat(item.min_temp))
}]
});
}

// 初始載入時顯示北部地區的溫度資料
fetchTemperatureData();
</script>
</body>
</html>
```



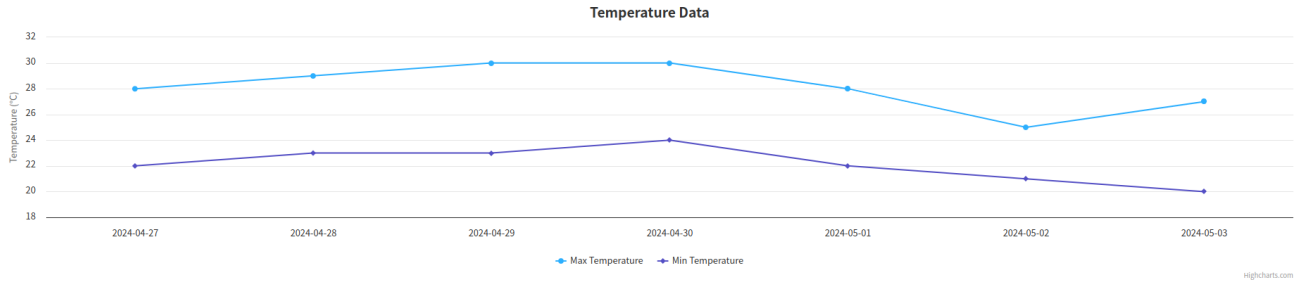






Temperature Visualization

Select Location: 東北部地區



Temperature Visualization

Select Location: 東南部地區