

Họ tên: Vũ Ngọc Trang

Lớp K56KMT.01

Mssv: K205480106035

### **Báo cáo bài tập**

Bài tập python ~ điểm thi học kỳ.

1. Dùng FastApi của python, xây dựng API (tự đưa vào logic xử lý input => output)
2. Cài đặt Node-Red trên windows (ko cần máy ảo), tạo chu trình tự động hoá gửi dữ liệu tới api, nhận về kết quả, lưu trữ vào database Sql server.
3. Tạo web đơn giản (html+js+css) với backend có thể là c# asp dot net, php, node-red, hoặc chính là python FastApi để lấy dữ liệu từ database Sql server, vẽ biểu đồ dữ liệu đã lưu. (Chart có thể dùng tùy ý thư viện thích hợp)

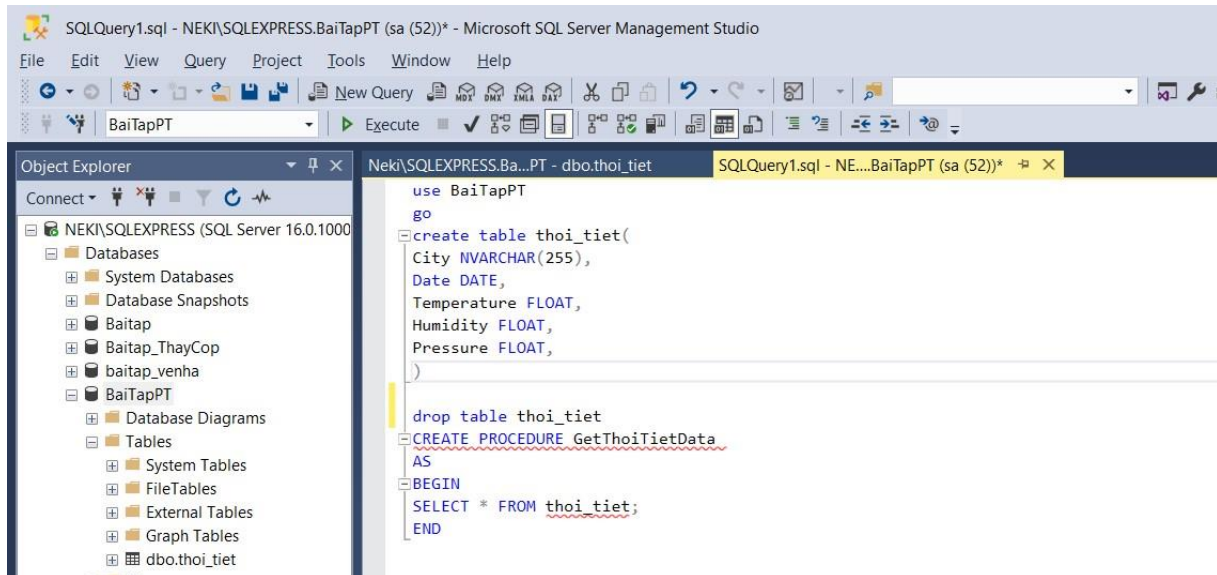
Yêu cầu:

1. Trình bày thuật toán xử lý của api, ý nghĩa của nó
2. Mô tả các bước cài đặt+ snap màn hình minh hoạ.
3. Mô tả quá trình chạy demo, hiểu được luồng xử lý dữ liệu. Hình ảnh minh hoạ
4. Kết luận: đã tìm hiểu được những kỹ thuật gì? Đã cài đặt và cấu hình thành công phần mềm nào? Đã tạo dc api gì? Đã phối hợp các kỹ thuật lập trình gì để đạt được điều gì? Kết quả cuối cùng xấu đẹp ra sao?....

Bước đầu tiên: Tải các môi trường để lập trình

- Cài sql (để lưu dữ liệu database)

Ở đây thì em tạo 2 bảng



1 bảng và lưu các trường thông tin và 1 bảng sp\_db

- Tải visual studio code (để viết code)

ở đây em không dùng visual code, em dùng **Pycharm**(thao tác dễ dàng)

link tải: [Download PyCharm: The Python IDE for data science and web development by JetBrains](#)

cần tải các thư viện như:

+ FastAPI và 1 số thư viện liên quan mà mình code sẽ có phần thông tin gợi ý tải...

- Tải node-red (là công cụ kéo-thả để kết nối và tự động hóa IoT, API và dịch vụ web)

Cách tải node-red. Ta cần cài node.js

Link tải: [Node.js — Download Node.js® \(nodejs.org\)](#)

Sau khi tải xong ta copy ứng dụng vào note path của hệ thống

➔ Sau đó mở cmd or terminal cài node-red thì dùng lệnh sau:

```
install -g --unsafe-perm node-red
```

- cài xong thì ta dùng lệnh : **node-red** để tạo địa chỉ : <http://127.0.0.1:1880>

- copy đường dẫn đó vào duyệt trình web.

➔ Ta đã cài và vào được môi trường sử dụng node-red

## Bắt đầu vào làm việc của bài:

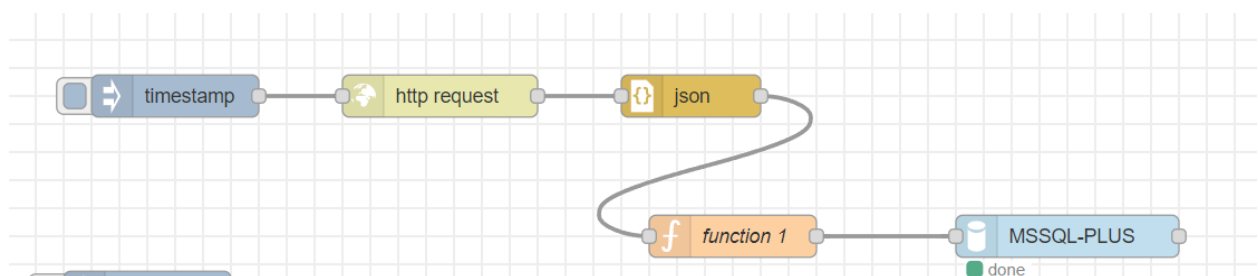
Ta lên **OpenWeather** (để tạo 1 tài khoản lấy key)

➔ File code py fastAPI để lấy dữ liệu.

```
1 from fastapi import FastAPI
2 from fastapi.responses import JSONResponse
3
4 app = FastAPI()
5
6 new *
7 @app.get("/weather") #endpoint
8 async def get_weather_data():
9     # đoạn mã lấy dữ liệu thời tiết
10    import requests
11    from datetime import datetime, timedelta
12
13    new *
14    def get_weather_data(city, api_key, date):
15        url = f"http://api.openweathermap.org/data/2.5/forecast?q={city}&appid={api_key}&units=metric"
16        response = requests.get(url)
17        data = response.json()
18        return data
19
20    new *
21    def extract_weather_info(data):
22        weather_info = {}
23        for item in data['list']:
```

Để lấy thông tin của openweather -> sử dụng fastAPI để gửi lên:

<http://127.0.0.1:8810/weather> -> sau đó ta sử dụng node-red để lấy dữ liệu này lưu về database.



- Chú ý: cần **install MSSQL-PLUS** ( để kết nối SQL và Node-red)

- Ở đây ta tạo gồm có: `inject`, `http request`, `json`, `function`, `MSSQL-Plus`
- Em hiểu và làm. Mình tạo 1 file py fastAPI có đường dẫn là <http://127.0.0.1:8000/weather> rồi lưu vào `http request` rồi gắn với json để chuyển sang thành file `json` từ json đến function để chuyển thành từ json sang thành mảng các đối tượng rồi lưu vào SQL.

```
CREATE PROCEDURE GetThoiTietData
AS
BEGIN
SELECT * FROM thoi_tiet;
END
```

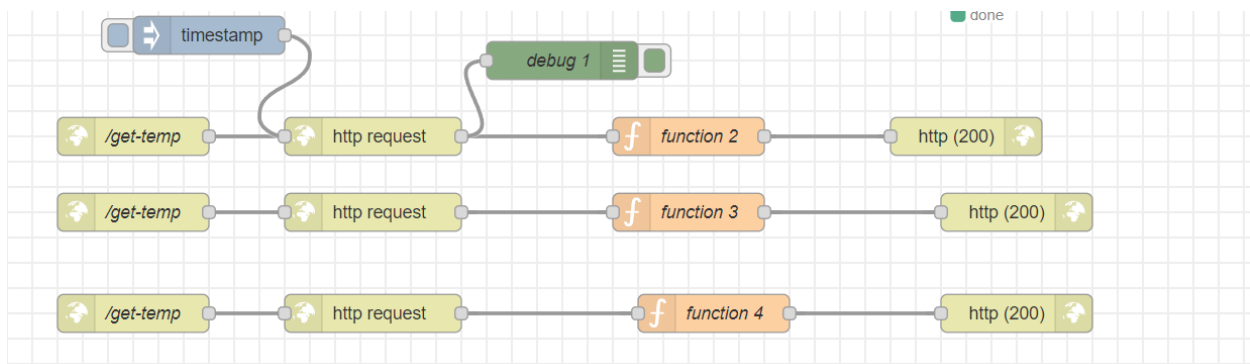
- ➔ Sau đó từ db lấy dữ liệu ra endpoint bằng py api bằng cách gọi `sp_getthoietdata`.
- Ta cần phải tạo thêm 1 file.py(trong file này gồm có fastAPI và pyodbc)

```
main.py fastAPL_SP.py x script.js <> index.html
22
23 # Định nghĩa route để lấy dữ liệu từ stored procedure
new *
24 @app.get(path="/thoietdata/", response_model=List[ThoiTietData])
25 async def get_thoiet_data():
26     try:
27         cursor = cnxn.cursor()
28         # Gọi stored procedure GetThoiTietData
29         cursor.execute("{CALL GetThoiTietData}")
30         rows = cursor.fetchall()
31
32         # Chuyển dữ liệu thành danh sách các từ điển
33         result = []
34         for row in rows:
35             result.append({
36                 'City': row.City,
37                 'Temperature': row.Temperature,
38                 'Humidity': row.Humidity,
39                 'Pressure': row.Pressure,
40                 'Date': row.Date
41             })
42
43     return result
```

Tạo xong ta có đường dẫn: [127.0.0.1:8810/thoietdata/](http://127.0.0.1:8810/thoietdata/) nó sẽ đẩy hết thông tin của bảng đó ra.

```
1 [
2   {
3     "City": "Thai Nguyen",
4     "Temperature": 25.88,
5     "Humidity": 92,
6     "Pressure": 1011,
7     "Date": "2024-05-14"
8   },
9   {
10    "City": "Thai Nguyen",
11    "Temperature": 25.97,
12    "Humidity": 90,
13    "Pressure": 1012,
14    "Date": "2024-05-15"
15  },
16  {
17    "City": "Thai Nguyen",
18    "Temperature": 23.32,
19    "Humidity": 95,
20    "Pressure": 1015,
21    "Date": "2024-05-16"
22  },
23  {
24    "City": "Thai Nguyen",
25    "Temperature": 23.76,
26    "Humidity": 91,
27    "Pressure": 1012,
28    "Date": "2024-05-17"
29  },
30  {
31    "City": "Thai Nguyen",
32    "Temperature": 24.68,
33    "Humidity": 96,
34    "Pressure": 1007,
35    "Date": "2024-05-18"
36  },
37  {
38    "City": "Thai Nguyen",
39    "Temperature": 23.55,
40    "Humidity": 96,
41    "Pressure": 1006,
```

➔ Sau khi gọi api thì sử dụng http request lấy thông tin và đưa vào endpoint.



ở đây có 3 hàng tượng trưng cho 3 trường. “Temperature, Humidity, Pressure”

đầu tiên ta gọi và đặt endpoint get cho từng trường sau chuyển file json sang hướng đối tượng đó sử dụng địa chỉ: [127.0.0.1:8810/thoitietdata/](http://127.0.0.1:8810/thoitietdata/) để hiển thị cho thông tin của mỗi trường đã được lọc.

➔ Ta tạo được địa chỉ của 3 trường này:

<http://127.0.0.1:1880/get-temp>

<http://127.0.0.1:1880/get-humi>

<http://127.0.0.1:1880/get-press>

- Từ đó ta có 3 file js để vẽ biểu đồ mà ta muốn.

```
127.0.0.1:1880/get-press
1 [
2 {
3   "Pressure": 1011,
4   "Date": "2024-05-14"
5 },
6 {
7   "Pressure": 1012,
8   "Date": "2024-05-15"
9 },
10 {
11   "Pressure": 1015,
12   "Date": "2024-05-16"
13 },
14 {
15   "Pressure": 1012,
16   "Date": "2024-05-17"
17 },
18 {
19   "Pressure": 1007,
20   "Date": "2024-05-18"
21 },
22 {
23   "Pressure": 1006,
24   "Date": "2024-05-19"
25 },
26 ]

127.0.0.1:1880/get-humi
1 [
2 {
3   "Humidity": 92,
4   "Date": "2024-05-14"
5 },
6 {
7   "Humidity": 90,
8   "Date": "2024-05-15"
9 },
10 {
11   "Humidity": 95,
12   "Date": "2024-05-16"
13 },
14 {
15   "Humidity": 91,
16   "Date": "2024-05-17"
17 },
18 {
19   "Humidity": 96,
20   "Date": "2024-05-18"
21 },
22 {
23   "Humidity": 96,
24   "Date": "2024-05-19"
25 },
26 ]

127.0.0.1:1880/get-temp
1 [
2 {
3   "Temperature": 25.88,
4   "Date": "2024-05-14"
5 },
6 {
7   "Temperature": 25.97,
8   "Date": "2024-05-15"
9 },
10 {
11   "Temperature": 23.32,
12   "Date": "2024-05-16"
13 },
14 {
15   "Temperature": 23.76,
16   "Date": "2024-05-17"
17 },
18 {
19   "Temperature": 24.68,
20   "Date": "2024-05-18"
21 },
22 {
23   "Temperature": 23.55,
24   "Date": "2024-05-19"
25 },
26 ]
```

➔ Ta đã được 3 file js và cuối cùng thì vẽ biểu đồ.

Sau khi muốn tại file.js thì chạy node-red và fastapi để lấy dữ liệu vẽ biểu đồ.

```
Project: BaiTapTHAYCOP
main.py fastAPISLP.py scriptjs index.html

// Hàm để vẽ biểu đồ áp suất
// usages: new *
function drawPressureChart() :void {
  loadData( url: 'http://127.0.0.1:1880/get-press', callback: function (data) :void {
    // Tạo mảng dữ liệu cho biểu đồ
    var chartData :any[] = [];
    chartData.push(['Time', 'Pressure']); // Thêm tiêu đề cột

    // Lặp qua dữ liệu JSON và thêm vào mảng dữ liệu
    data.forEach(function (entry) :void {
      chartData.push([new Date(entry.Date), entry.Pressure]);
    });

    // Tạo DataTable từ mảng dữ liệu
    var dataTable = google.visualization.arrayToDataTable(chartData);

    // Cấu hình tùy chọn cho biểu đồ
    var options :{...} = {
      title: 'Pressure Chart',
      curveType: 'function',
      legend: {position: 'bottom'}
    };

    // Vẽ biểu đồ
    var chart : google.visualization.LineChart = new google.visualization.LineChart(document.getElementById('pressure_chart'));
    chart.draw(dataTable, options);
  });
}
```

➔ Ta có kết quả:

## 1. SQL:

Neki\SQLEXPRESS.BaiTapPT - dbo.thoi\_tiet - Microsoft SQL Server Management Studio

File Edit View Project Query Designer Tools Window Help

Connect - BaiTapPT

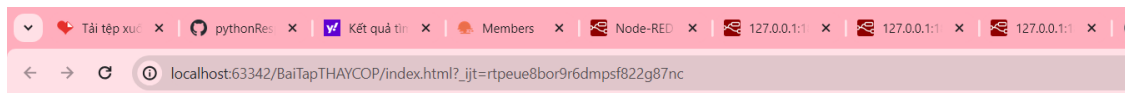
Object Explorer

- NEKI\SQLEXPRESS (SQL Server 16.0.1000)
- Databases
  - System Databases
  - Database Snapshots
  - Baitap
  - Baitap\_ThayCop
  - baitap\_venha
  - BaiTapPT
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.thoi\_tiet

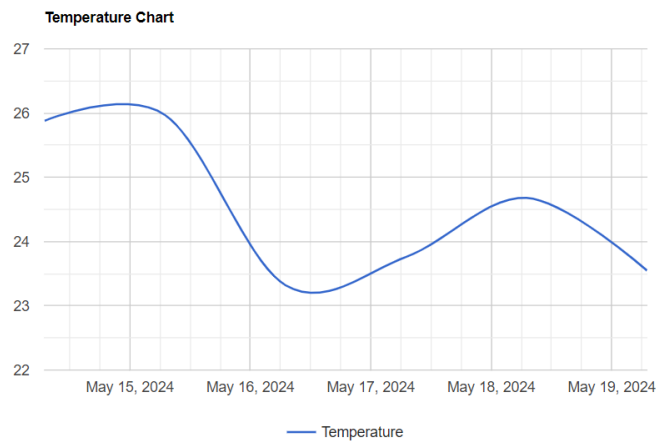
SQLQuery1.sql - NE...BaiTapPT (sa (52))\*

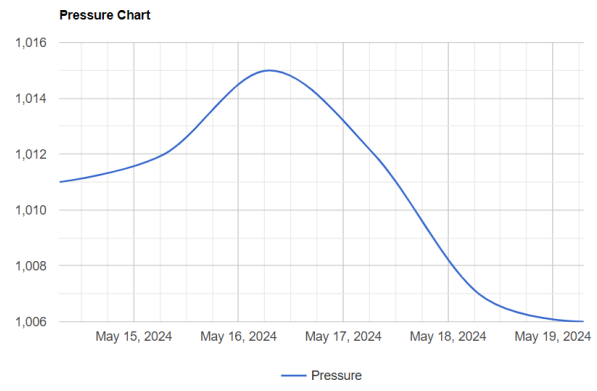
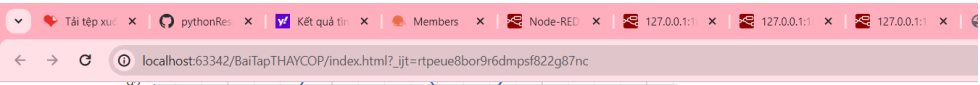
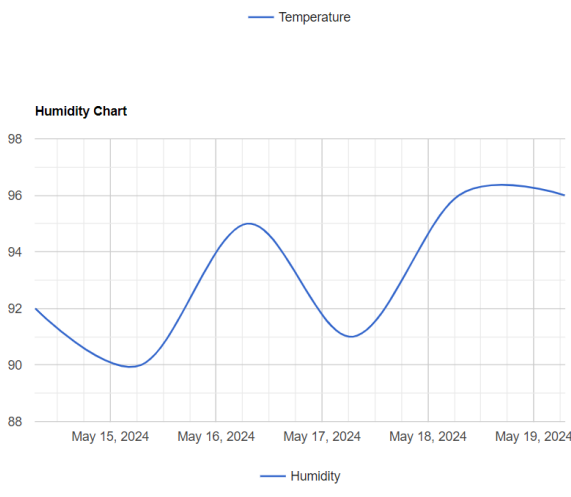
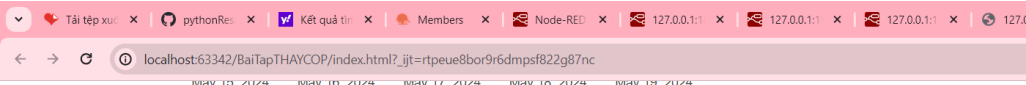
	City	Date	Temperature	Humidity	Pressure
▶	Thai Nguyen	2024-05-14	25.88	92	1011
	Thai Nguyen	2024-05-15	25.97	90	1012
	Thai Nguyen	2024-05-16	23.32	95	1015
	Thai Nguyen	2024-05-17	23.76	91	1012
	Thai Nguyen	2024-05-18	24.68	96	1007
	Thai Nguyen	2024-05-19	23.55	96	1006
*	NULL	NULL	NULL	NULL	NULL

## 2. Biểu đồ:



### Vẽ biểu đồ thời tiết







### 3.Code

- code gọi dữ liệu openweather về để lưu vào sql

```
from fastapi import FastAPI
from fastapi.responses import JSONResponse

app = FastAPI()

@app.get("/weather") #endpoint
async def get_weather_data():
    # Đoạn mã lấy dữ liệu thời tiết
    import requests
    from datetime import datetime, timedelta

    def get_weather_data(city, api_key, date):
        url =
f"http://api.openweathermap.org/data/2.5/forecast?q={city}&appid={api_key}&un
its=metric"
        response = requests.get(url)
        data = response.json()
        return data

    def extract_weather_info(data):
        weather_info = {}
        for item in data['list']:
            # Lấy chỉ ngày từ chuỗi datetime
            date = item['dt_txt'].split()[0]
            if date not in weather_info:
                temperature = item['main']['temp']
                humidity = item['main']['humidity']
                pressure = item['main']['pressure']
                rain = item.get('rain', {}).get('3h', 0) # Lượng mưa trong 3
giờ (nếu có)
                wind_speed = item['wind']['speed']
                weather_info[date] = {'Date': date, 'Temperature':
temperature, 'Humidity': humidity,
                                     'Pressure': pressure, 'City': city}
            return list(weather_info.values())

    def get_past_date(days):
        today = datetime.now()
        past_date = today - timedelta(days=days)
        return past_date.strftime("%Y-%m-%d")

    city = "Thai Nguyen"
    api_key = "093bc10f5336f52a5c3c761e7b731280"
    past_date = get_past_date(7) # Lấy dữ liệu 1 tuần trước

    weather_data = get_weather_data(city, api_key, past_date)
    weather_info = extract_weather_info(weather_data)

    return JSONResponse(content=weather_info)
```

- Code fastAPI.py

```
- from fastapi import FastAPI, HTTPException
from typing import List
from pydantic import BaseModel
import pyodbc

app = FastAPI()

# Thiết lập thông tin kết nối tới cơ sở dữ liệu SQL Server
server = 'Neki'
database = 'BaiTapPT'
username = 'sa'
password = '2402'
cnxn = pyodbc.connect('DRIVER={SQL
Server};SERVER='+server+';DATABASE='+database+';UID='+username+';PWD='+
password)

# Định nghĩa model cho dữ liệu thời tiết
class ThoiTietData(BaseModel):
    City: str
    Temperature: float
    Humidity: int
    Pressure: int
    Date: str

# Định nghĩa route để lấy dữ liệu từ stored procedure
@app.get("/thoietietdata/", response_model=List[ThoiTietData])
async def get_thoietiet_data():
    try:
        cursor = cnxn.cursor()
        # Gọi stored procedure GetThoiTietData
        cursor.execute("{CALL GetThoiTietData}")
        rows = cursor.fetchall()

        # Chuyển dữ liệu thành danh sách các từ điển
        result = []
        for row in rows:
            result.append({
                'City': row.City,
                'Temperature': row.Temperature,
                'Humidity': row.Humidity,
                'Pressure': row.Pressure,
                'Date': row.Date
            })

        return result
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

if __name__ == "__main__":
    import uvicorn
    uvicorn.run(app, host="127.0.0.1", port=8810)
```

- Code script.js

```
- // Hàm để tải dữ liệu từ URL JSON
google.charts.load('current', {'packages':['corechart']});
google.charts.setOnLoadCallback(drawCharts);
function drawCharts() {
    drawTemperatureChart();
    drawHumidityChart();
    drawPressureChart();
}
function loadData(url, callback) {
    fetch(url)
        .then(response => response.json())
        .then(data => callback(data))
        .catch(error => console.error('Error loading data:', error));
}

// Hàm để vẽ biểu đồ nhiệt độ
function drawTemperatureChart() {
    loadData('http://127.0.0.1:1880/get-temp', function(data) {
        // Tạo mảng dữ liệu cho biểu đồ
        var chartData = [];
        chartData.push(['Date', 'Temperature']); // Thêm tiêu đề cột

        // Lặp qua dữ liệu JSON và thêm vào mảng dữ liệu
        data.forEach(function(entry) {
            chartData.push([new Date(entry.Date), entry.Temperature]);
        });

        // Tạo DataTable từ mảng dữ liệu
        var dataTable =
            google.visualization.arrayToDataTable(chartData);

        // Cấu hình tùy chọn cho biểu đồ
        var options = {
            title: 'Temperature Chart',
            curveType: 'function',
            legend: { position: 'bottom' }
        };

        // Vẽ biểu đồ
        var chart = new
            google.visualization.LineChart(document.getElementById('temperature_chart'));
        chart.draw(dataTable, options);
    });
}

// Hàm để vẽ biểu đồ độ ẩm
// Hàm để vẽ biểu đồ độ ẩm
function drawHumidityChart() {
    loadData('http://127.0.0.1:1880/get-humi', function(data) {
        // Tạo mảng dữ liệu cho biểu đồ
        var chartData = [];
        chartData.push(['Time', 'Humidity']); // Thêm tiêu đề cột
```

```

        // Lặp qua dữ liệu JSON và thêm vào mảng dữ liệu
        data.forEach(function(entry) {
            chartData.push([new Date(entry.Date), entry.Humidity]);
        });

        // Tạo DataTable từ mảng dữ liệu
        var dataTable =
google.visualization.arrayToDataTable(chartData);

        // Cấu hình tùy chọn cho biểu đồ
        var options = {
            title: 'Humidity Chart',
            curveType: 'function',
            legend: { position: 'bottom' }
        };

        // Vẽ biểu đồ
        var chart = new
google.visualization.LineChart(document.getElementById('humidity_chart'
));
        chart.draw(dataTable, options);
    });
}

// Hàm để vẽ biểu đồ áp suất
function drawPressureChart() {
    loadData('http://127.0.0.1:1880/get-press', function (data) {
        // Tạo mảng dữ liệu cho biểu đồ
        var chartData = [];
        chartData.push(['Time', 'Pressure']); // Thêm tiêu đề cột

        // Lặp qua dữ liệu JSON và thêm vào mảng dữ liệu
        data.forEach(function (entry) {
            chartData.push([new Date(entry.Date), entry.Pressure]);
        });

        // Tạo DataTable từ mảng dữ liệu
        var dataTable =
google.visualization.arrayToDataTable(chartData);

        // Cấu hình tùy chọn cho biểu đồ
        var options = {
            title: 'Pressure Chart',
            curveType: 'function',
            legend: {position: 'bottom'}
        };

        // Vẽ biểu đồ
        var chart = new
google.visualization.LineChart(document.getElementById('pressure_chart'
));
        chart.draw(dataTable, options);
    });
}

```

## - Code html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Charts</title>
  <!-- Bao gồm thư viện Google Charts -->
  <script type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>
  <script src="script.js"></script>
</head>
<body>
  <h1>Vẽ biểu đồ thời tiết</h1>
  <!-- Đây là nơi để hiển thị các biểu đồ -->
  <div id="temperature_chart" style="width: 900px; height: 500px;"></div>
  <div id="humidity_chart" style="width: 900px; height: 500px;"></div>
  <div id="pressure_chart" style="width: 900px; height: 500px;"></div>
</body>
</html>
```

## Kết luận

Sau bài này em cảm nhận được từ lập trình trên nhiều nền tảng rất là thú vị không thể thiếu đó là chatgpt cho ta am hiểu về code hơn. Phần mảng này và mảng nọ công dụng rất có ích cho tương lai sau này em sẽ cố gắng phát triển thêm...