TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN MÔN HỌC
LẬP TRÌNH PYTHON

Giáo viên hướng dẫn: **Đỗ Duy Cốp**

Sinh viên thực hiện: **Vũ Công Anh**

MSSV: **K205480106003**

Lớp: **K56KMT.01**

Chuyên ngành: **Kỹ Thuật Máy Tính**

Thái Nguyên 2024

TRƯỜNG ĐẠI HỌC KTCN KHOA ĐIỆN TỬ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc
----------o0o----------

PHIẾU GIAO ĐỀ TÀI BÀI TẬP LỚN MÔN HỌC LẬP TRÌNH PYTHON

I. Thông tin sinh viên

Sinh viên thực hiện: Vũ Công Anh Mã SV: K205480106003

II. Tên đề tài

HỆ THÔNG GIÁM SÁT KHÔNG KHÍ

III. Mục tiêu

- Lấy dữ liệu không khí
- Xử lý dữ liệu :sử dụng FastAPI và Node-RED, sau đó lưu vào cơ sở dữ liệu.
- Xây dựng trang web

IV. Nội dung thực hiện

1. Sử dụng API của các nguồn dữ liệu như api weather. Lấy dữ liệu không khí từ : https://openweathermap.org/api/air-pollution (có update realtime)
2. Tạo một cơ sở dữ liệu trong SQL Server để lưu trữ dữ liệu không khí
3. Sử dụng Node-RED để xây dựng các luồng dữ liệu tự động, kết nối và xử lý dữ liệu từ API đến cơ sở dữ liệu.
4. Sử dụng FastAPI để tạo các endpoint API để truy xuất dữ liệu không khí từ cơ sở dữ liệu.
5. Sử dụng các công nghệ front-end (HTML, CSS, JavaScript, React.js) để xây dựng giao diện người dùng.
6. Tạo biểu đồ thể hiện không khí SO2, NO2, PM10, PM2.5, O3, CO.

V. Ngày giao nhiệm vụ: 15/5/20204

VI. Ngày hoàn thành: 25/5/2024

VII. Giáo viên hướng dẫn: Đỗ Duy Cốp

NHẬN XÉT CỦA GIÁO VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày.....tháng. ....năm 2024

CHỮ KÝ CỦA GIÁO VIÊN

(ký, ghi rõ họ tên)

MỤC LỤC

I. TỔNG QUAN CHUNG 5
1.1. Đặt vấn đề 5
1.2. Mục tiêu 5
1.3. Hướng giải quyết 6
1.4. Giới hạn đề tài 6
II. CƠ SỞ LÝ THUYẾT 7
2.1. SQL Server Management Studio 7
2.2. Ngôn ngữ lập trình Python 8
2.3. Visual Studio 2022 8
2.4. Node_red 10
III. NỘI DUNG THỰC HIỆN 11
IV. KẾT LUẬN 24

I. TỔNG QUAN CHUNG

1.1. Đặt vấn đề

Trong bối cảnh hiện nay, ô nhiễm không khí đang là một vấn đề nghiêm trọng ảnh hưởng đến sức khỏe con người và môi trường sống. Các chất gây ô nhiễm không khí như PM2.5, PM10, CO2, và các hợp chất hữu cơ dễ bay hơi (VOC) có thể gây ra các bệnh về hô hấp, tim mạch và thậm chí ung thư. Ngoài ra, ô nhiễm không khí còn góp phần làm trầm trọng thêm hiện tượng biến đổi khí hậu, gây thiệt hại về kinh tế và ảnh hưởng xấu đến chất lượng cuộc sống.

Việc giám sát chất lượng không khí giúp chúng ta có thể phát hiện kịp thời các vấn đề về ô nhiễm và đưa ra các biện pháp phòng ngừa hiệu quả. Thông qua việc thu thập và phân tích dữ liệu chất lượng không khí, chúng ta có thể:

- ❖ Xác định các nguồn gây ô nhiễm chính và mức độ ảnh hưởng của chúng.
- ❖ Cảnh báo sớm khi mức độ ô nhiễm vượt ngưỡng an toàn, từ đó giúp người dân và các cơ quan chức năng có thể đưa ra các biện pháp bảo vệ sức khỏe kịp thời.
- ❖ Đánh giá hiệu quả của các chính sách và biện pháp kiểm soát ô nhiễm đã triển khai, từ đó đưa ra các điều chỉnh cần thiết.
- ❖ Nâng cao nhận thức cộng đồng về tình trạng ô nhiễm không khí và tầm quan trọng của việc bảo vệ môi trường.

Tuy nhiên, việc giám sát chất lượng không khí hiện nay còn gặp nhiều khó khăn do dữ liệu phân tán và không đồng nhất. Để giải quyết vấn đề này, cần xây dựng một hệ thống giám sát chất lượng không khí toàn diện, cho phép thu thập, xử lý và hiển thị dữ liệu một cách hiệu quả. Hệ thống này không chỉ giúp cải thiện khả năng quản lý và ứng phó với ô nhiễm không khí mà còn góp phần bảo vệ sức khỏe cộng đồng và môi trường sống trong lành hơn

1.2. Mục tiêu

Trong bối cảnh hiện nay, ô nhiễm không khí đang là một vấn đề nghiêm trọng ảnh

hưởng đến sức khỏe con người và môi trường sống. Các chất gây ô nhiễm không khí như PM2.5, PM10, CO2, và các hợp chất hữu cơ dễ bay hơi (VOC) có thể gây ra các bệnh về hô hấp, tim mạch và thậm chí ung thư. Ngoài ra, ô nhiễm không khí còn góp phần làm trầm trọng thêm hiện tượng biến đổi khí hậu, gây thiệt hại về kinh tế và ảnh hưởng xấu đến chất lượng cuộc sống.

1.3. Hướng giải quyết

Để giải quyết vấn đề quản lý dữ không khi và hiển thị thông tin một cách hiệu quả, chúng ta sẽ thực hiện các bước sau:

a. Dùng request để lấy dữ liệu thô từ API và gửi nó lên endpoint của FastAPI

- Sử dụng Python để gửi các yêu cầu POST đến endpoint của FastAPI nhằm thu thập dữ liệu thô. Thư viện `requests` trong Python sẽ hỗ trợ chúng ta trong việc này.

b. Ở Node-RED dùng node https request lấy địa chỉ có chứa endpoint (thường là cổng 127.0.0.1:8000/xxx):

- Thiết lập một flow trong Node-RED để gửi HTTP request tới endpoint địa phương (localhost). Node-RED sẽ đóng vai trò trung gian, giúp ta gửi yêu cầu HTTP và nhận phản hồi từ endpoint.

c. Lưu vào SQL:

- Khi nhận được dữ liệu từ endpoint, chúng ta sẽ lưu trữ dữ liệu này vào cơ sở dữ liệu SQL. Sử dụng ORM (Object Relational Mapping) như SQLAlchemy trong Python để tương tác với cơ sở dữ liệu một cách dễ dàng và hiệu quả.

d. Xây dựng giao diện người dùng lấy dữ liệu từ SQL để vẽ biểu đồ:

- Sử dụng html,css,js để lấy dữ liệu từ sqlserver thông qua asp.net (api.aspx)

Với hướng giải quyết này, chúng ta sẽ có một hệ thống toàn diện cho phép thu thập, xử lý và hiển thị dữ liệu không khí một cách hiệu quả, đồng thời cung cấp thông tin đáng tin cậy và dễ hiểu cho người dùng.

1.4. Giới hạn đề tài

Bài làm của em cơ bản đã hoàn thành được những yêu cầu đặt ra ở đầu. Nhưng do kiến thực còn hạn hẹp nên bài làm của em còn nhiều thiếu sót. Trong tương lại em sẽ cố gắng khắc phục những hạn chế để giúp hệ thống trở nên hoàn thiện hơn, đáp ứng tốt hơn nhu cầu của người dùng.

II. CƠ SỞ LÝ THUYẾT

2.1. SQL Server Management Studio

SQL Server Management Studio (SSMS) là một ứng dụng phần mềm của Microsoft, được thiết kế để quản lý và tương tác với các cơ sở dữ liệu SQL Server. Được phát triển từ năm 2005, SSMS là một công cụ quản lý cơ bản và quan trọng cho các quản trị viên cơ sở dữ liệu, nhà phát triển và các chuyên gia dữ liệu.

SQL Server được phát triển lần đầu tiên vào năm 1989 bởi Microsoft, hợp tác với Sybase và Ashton-Tate. Từ đó, nó đã trải qua nhiều phiên bản cải tiến với những tính năng và khả năng mới, trở thành một trong những hệ quản trị cơ sở dữ liệu phổ biến nhất trên thế giới.

SSMS cung cấp một giao diện người dùng đồ họa (GUI) thân thiện và dễ sử dụng cho việc quản lý cơ sở dữ liệu SQL Server. Giao diện này cho phép người dùng thực hiện các tác vụ quản lý dữ liệu một cách dễ dàng và hiệu quả. SSMS cung cấp một loạt các công cụ quản lý tích hợp, cho phép người dùng thực hiện các tác vụ như tạo, sửa đổi và xóa cơ sở dữ liệu, bảng, chỉ mục và thủ tục lưu trữ. Nó cũng cho phép quản trị viên sao lưu và phục hồi dữ liệu, kiểm tra và theo dõi hiệu suất, và quản lý bảo mật. SSMS cho phép người dùng thực hiện các truy vấn SQL và xem dữ liệu từ các bảng trong cơ sở dữ liệu. Nó cung cấp một trình soạn thảo truy vấn mạnh mẽ với tính năng gợi ý cú pháp và điều hướng thông minh giúp tăng hiệu suất lập trình.

SQL Server cung cấp cho người dùng các công cụ và tính năng để quản lý, lưu trữ, xử lý các truy vấn dữ liệu, kiểm soát truy cập, xử lý giao dịch và hỗ trợ tích hợp dữ liệu từ nhiều nguồn khác nhau.

Ngoài ra, SQL Server cũng cung cấp các công cụ để tạo báo cáo, phân tích và quản lý cơ sở dữ liệu trực quan thông qua giao diện người dùng hoặc các script lệnh SQL.SQL Server được xây dựng dựa trên SQL, một ngôn ngữ lập trình tiêu chuẩn để tương tác với cơ sở dữ liệu quan hệ. SQL Server được liên kết với Transact-SQL hoặc T-SQL, triển khai SQL của Microsoft có bổ sung một tập hợp các cấu trúc lập trình độc quyền.

2.2. Ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình đa mục đích, dễ học và mạnh mẽ, được phát triển bởi Guido van Rossum và ra mắt lần đầu vào năm 1991. Với cú pháp đơn giản và gần gũi với ngôn ngữ tự nhiên, Python là một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới, được sử dụng rộng rãi trong các lĩnh vực khác nhau từ phát triển web, khoa học dữ liệu đến trí tuệ nhân tạo.

Python không chỉ dễ học mà còn linh hoạt và mở rộng, hỗ trợ nhiều phong cách lập trình và tích hợp tốt với các ngôn ngữ khác như C/C++, Java và .NET. Hệ sinh thái phong phú của Python cung cấp các thư viện và framework đa dạng, giúp lập trình viên dễ dàng phát triển các ứng dụng và dự án.

Điểm nổi bật của Python là cộng đồng lớn mạnh, với hàng triệu lập trình viên trên khắp thế giới, sẵn sàng chia sẻ kiến thức và kinh nghiệm. Nhờ vào điều này, Python không chỉ là một ngôn ngữ lập trình mà còn là một cộng đồng và một triển khai tri thức phong phú, đóng vai trò quan trọng trong việc giải quyết các thách thức hiện đại trong ngành công nghiệp và khoa học.

2.3. Pycharm

PyCharm là một phần mềm được phát triển bởi JetBrains, cung cấp các công cụ cần thiết giúp các lập trình viên Python tăng năng suất làm việc. Ngoài ra, PyCharm còn được tích hợp nhiều yếu tố mở rộng khác như: biên dịch mã, tô sáng cú pháp, điều hướng project nhanh chóng, công cụ cơ sở dữ liệu và trình soạn thảo văn bản có tích hợp lập trình nhằm mục đích thúc đẩy quá

trình phát triển Website.

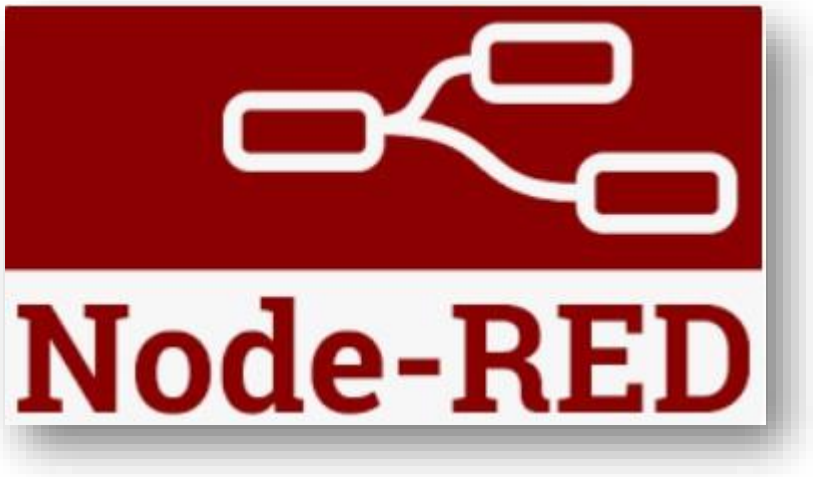
PyCharm cũng tương tự như Python, là một trong những IDE được nhiều người dùng ưa chuộng và sử dụng rộng rãi trong giới lập trình. Nhờ sự tiện lợi và hiệu quả cao, PyCharm đã được nhiều doanh nghiệp có tầm ảnh hưởng lớn sử dụng làm IDE Python như: Symantec, Twitter, Pinterest,...

Một trong những điểm mạnh của việc sử dụng PyCharm là khả năng cung cấp API cho các nhà phát triển, đồng thời cho phép họ viết các plugin riêng để mở rộng tính năng. Hơn nữa, phần mềm này còn tương thích với đa dạng các hệ điều hành như Linux, Windows và macOS. Điều này, giúp các lập trình viên Python tiết kiệm thời gian đáng kể trong quá trình phát triển một ứng dụng hay website nào đó.

2.4. Node_red

Node-RED là một công cụ mã nguồn mở được phát triển bởi IBM và cung cấp một giao diện trực quan để kết nối các thiết bị, dịch vụ và ứng dụng một cách linh hoạt và dễ dàng. Nó được xây dựng dựa trên Node.js và sử dụng một giao diện trực quan dựa trên trình duyệt để tạo, quản lý và triển khai các luồng làm việc (flow) dựa trên sự kết hợp của các "nút" và "luồng".

Các nút trong Node-RED đại diện cho các chức năng hoặc dịch vụ cụ thể, và chúng có thể được kéo và thả vào khung làm việc để tạo ra các luồng làm việc. Mỗi nút thường thực hiện một chức năng nhất định, từ xử lý dữ liệu đến gửi và nhận thông điệp qua các giao thức mạng khác nhau.



Node-RED được sử dụng rộng rãi trong Internet of Things (IoT) và trong các ứng dụng tự động hóa, nơi nó có thể giúp kết nối và tự động hóa các thiết bị và dịch vụ từ nhiều nhà sản xuất khác nhau. Nó cũng thích hợp cho việc xử lý dữ liệu thời gian thực và tích hợp các dịch vụ web khác nhau.

Node-RED cung cấp một cộng đồng lớn và sôi động, với nhiều nút và gói mở rộng được phát triển và chia sẻ miễn phí. Điều này giúp người dùng mở rộng và tùy chỉnh Node- RED theo nhu cầu và yêu cầu cụ thể của họ.

III. NỘI DUNG THỰC HIỆN

Sau đây là các bước thực hiện đề tài của em:

Dùng request để lấy dữ liệu thô từ API và gửi nó lên endpoint của FastAPI

Em tạo 1 file Python và sử dụng FastAPI để lấy dữ liệu về không khí từ một API công cộng và trả về dưới dạng JSON thông qua một API endpoint.

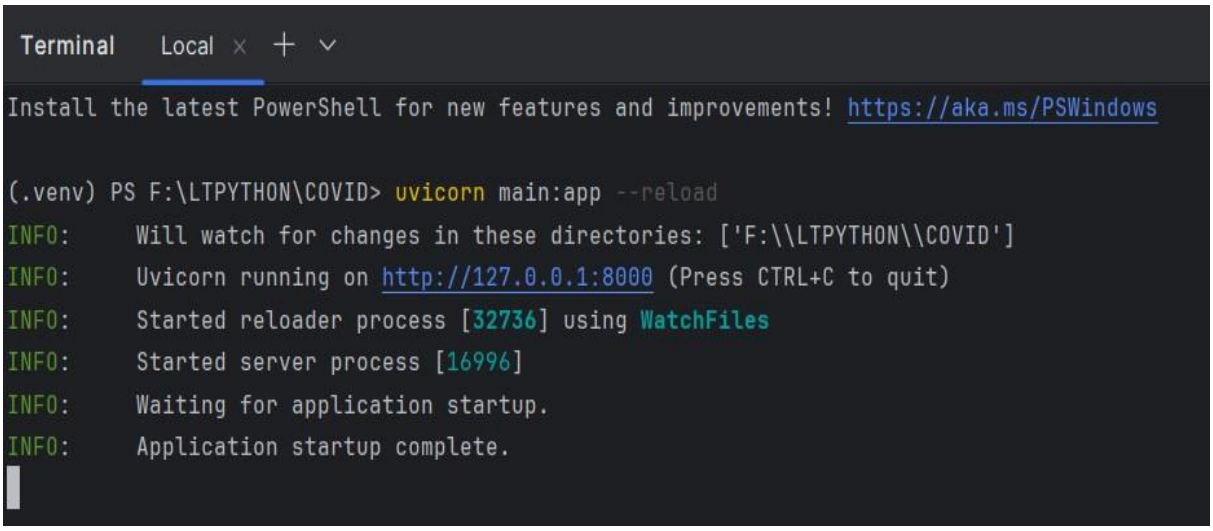
main.py

```python
from fastapi import FastAPI, HTTPException
import requests
import uvicorn

app = FastAPI()

# URL của API để lấy dữ liệu COVID-19 theo quốc gia
url = "https://disease.sh/v3/covid-19/countries"

THAO
@app.get("/covid-data")
def get_covid_data():
    # Gửi yêu cầu GET tới API
    response = requests.get(url)

    # Kiểm tra nếu yêu cầu thành công
    if response.status_code != 200:
        raise HTTPException(status_code=response.status_code, detail="Failed to retrieve data")

    # Lấy dữ liệu JSON từ phản hồi
    data = response.json()
```

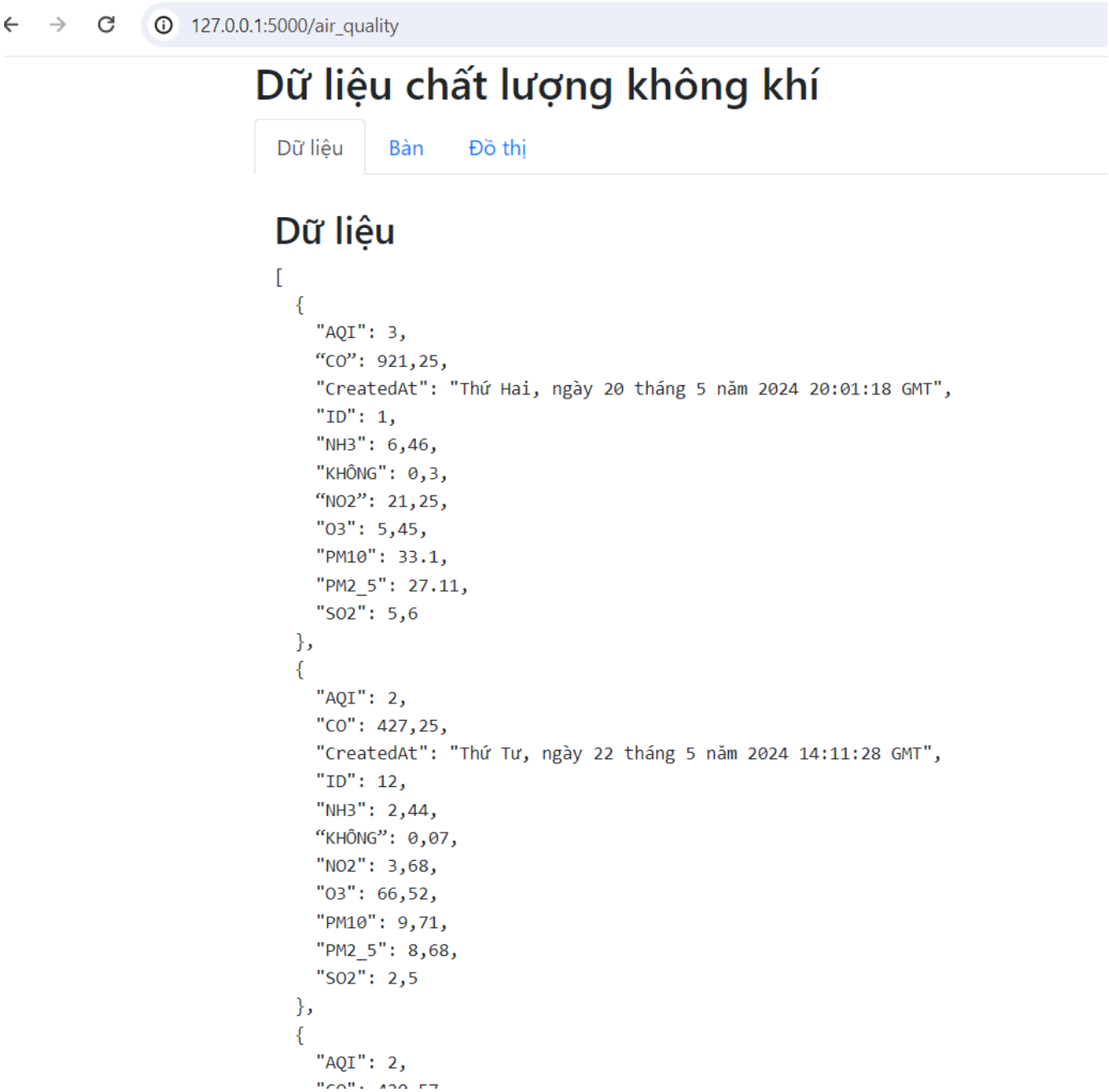
index.html | inde.html | chart.html | code1.py | code2.py

```python
from fastapi import FastAPI, HTTPException
import httpx
import asyncio
import logging

app = FastAPI()

AIR_QUALITY_API_URL = "https://api.openweathermap.org/data/2.5/air_pollution"
API_KEY = "1df7d279e27869dccb309c39f5973269"

air_quality_data = None
UPDATE_INTERVAL = 3600  # Ví dụ: cập nhật mỗi giờ

# Thiết lập logging
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

2 usages  conganh
async def fetch_air_quality_data():
    global air_quality_data
    try:
        # Thay thế các giá trị vĩ độ và kinh độ dưới đây bằng các giá trị tương ứng cu
        params = {"lat": "21.0285", "lon": "105.8542", "appid": API_KEY}
        async with httpx.AsyncClient() as client:
            response = await client.get(AIR_QUALITY_API_URL, params=params)
            if response.status_code == 200:
                data = response.json()
                air_quality_data = {
                    "AQI": data["list"][0]["main"]["aqi"],
```

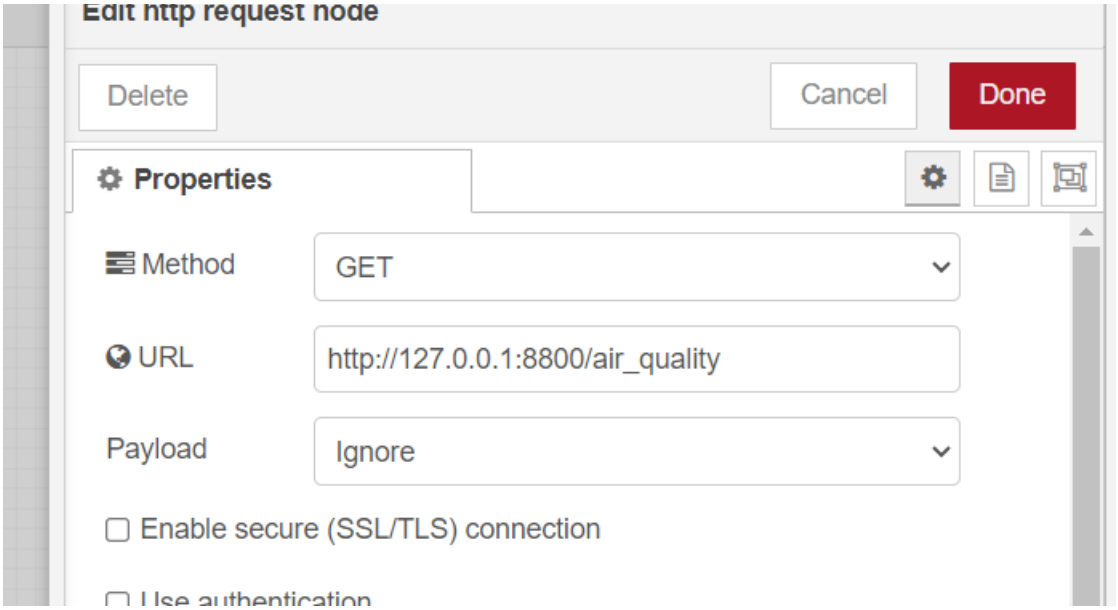
Khởi chạy FastAPI



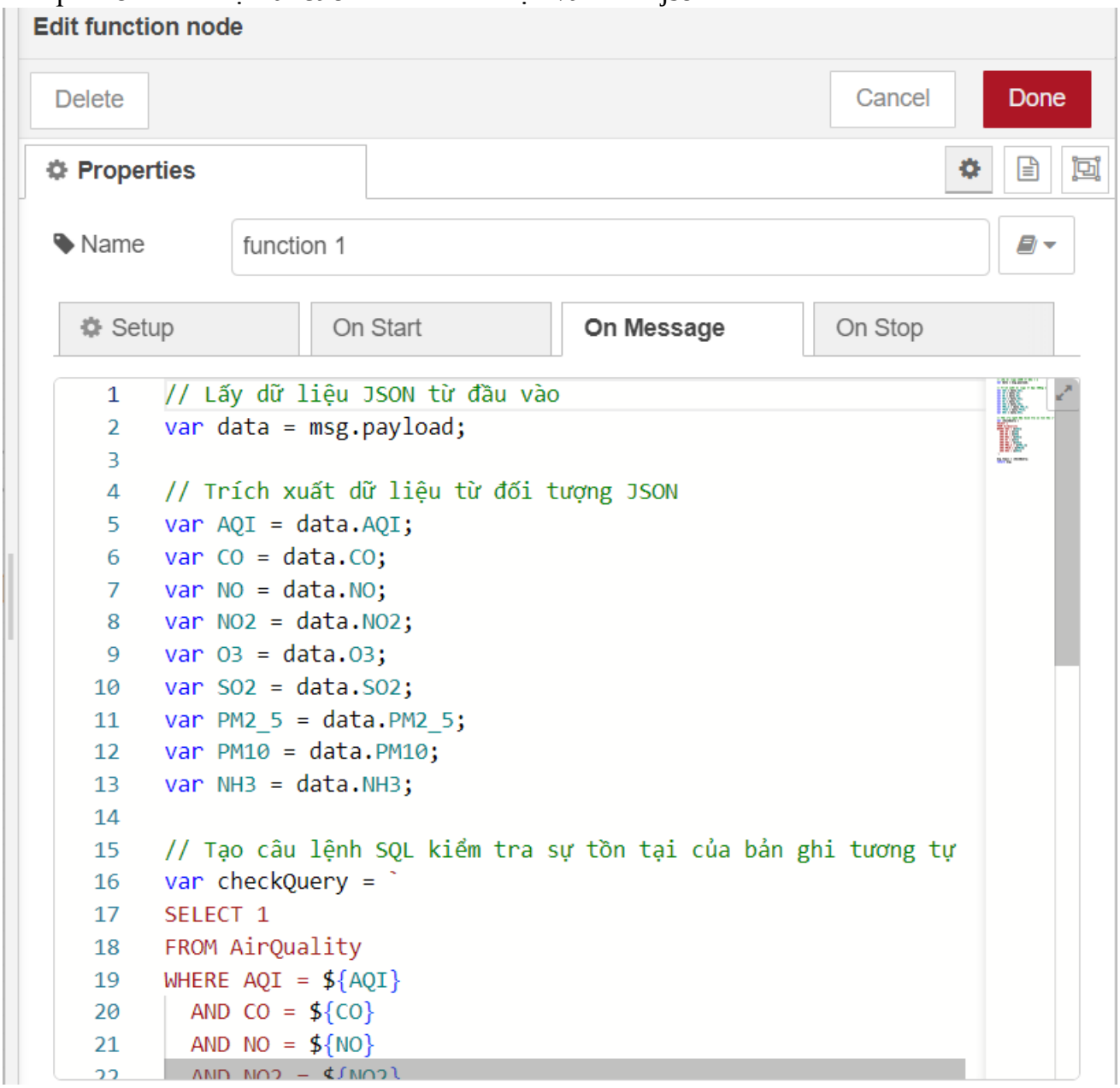
Sau khi khởi chạy nó sẽ trả về một chuỗi dạng json trên local của mình



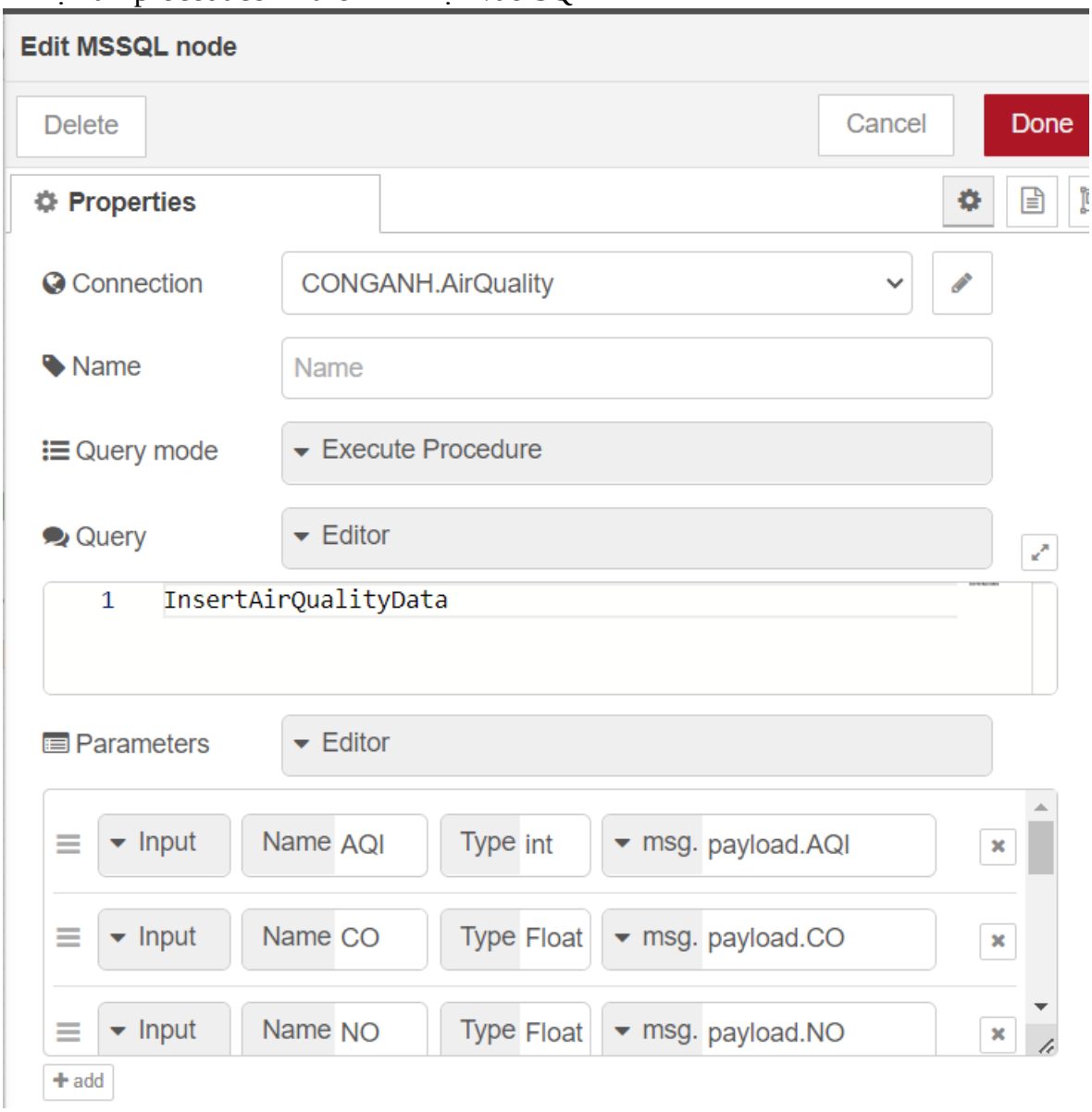
Ở Node-RED dùng node https request lấy địa chỉ có chứa endpoint. Thiết lập một flow trong Node-RED để gửi HTTP request tới endpoint địa phương (localhost)



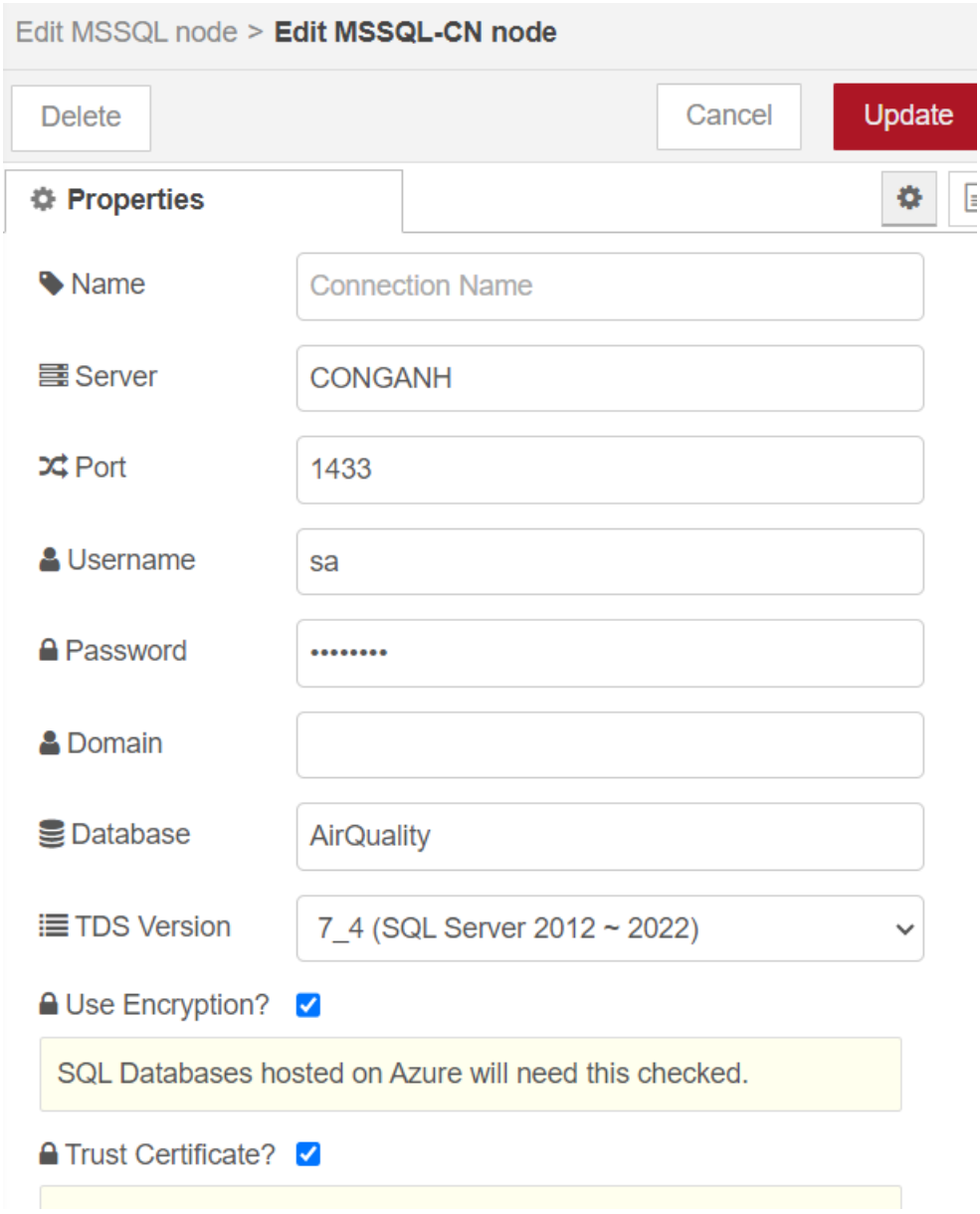
Tiếp đó em viết một function để xử lí dữ liệu và trả về json



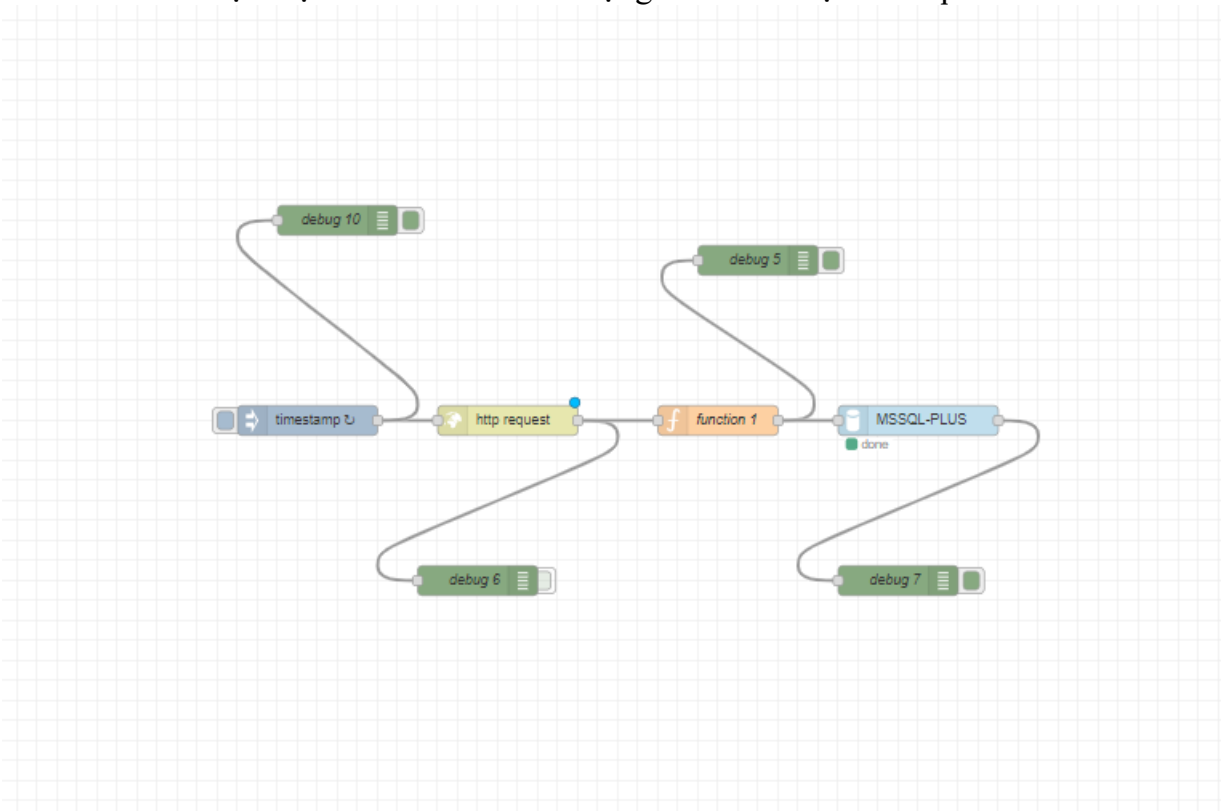
Thực thi proceduce để thêm dữ liệu vào SQL



Cài đặt NODE-RED- MSSQL-PLUS: sau khi cài đặt cấu hình các thông tin cho node



Cấu trúc của một đoạn code node-red sử dụng để lưu dữ liệu vào sql



Tạo bảng để lưu dữ trữ dữ liệu ở SQL:

CONGANH\SQLEXPRE...- dbo.AirQuality | SQLQuery3.sql - CON...irQuality (sa (52))

Column Name	Data Type	Allow Nulls
ID	int	☐
AQI	int	☑
CO	float	☑
NO	float	☑
NO2	float	☑
O3	float	☑
SO2	float	☑
PM2_5	float	☑
PM10	float	☑
NH3	float	☑
CreatedAt	datetime	☑
		☐

Dữ liệu được lưu vào dbo. AirQuality

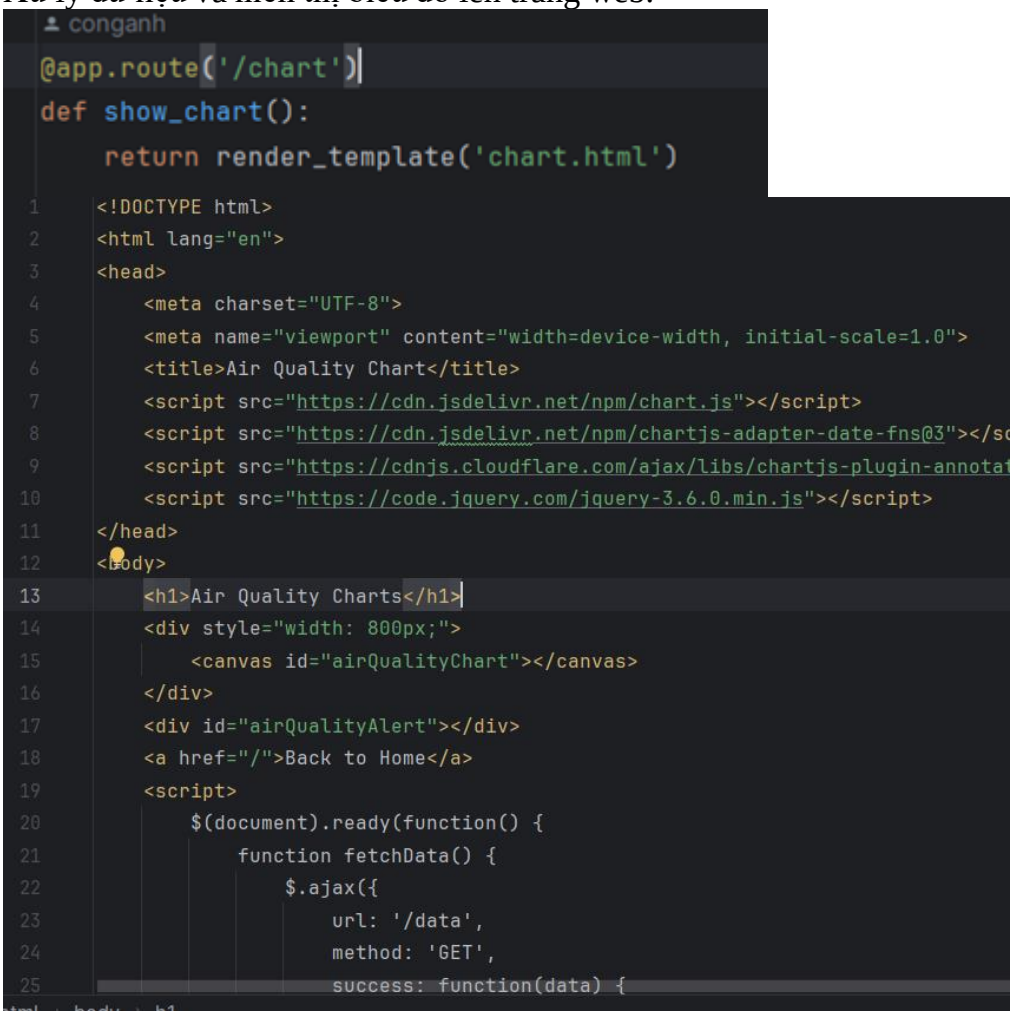
ID	AQI	CO	NO	NO2	O3	SO2	PM2_5	PM10	NH3	CreatedAt
1	3	921.25	0.3	21.25	5.45	5.6	27.11	33.1	6.46	2024-05-20 ...
12	2	427.25	0.07	3.68	66.52	2.5	8.68	9.71	2.44	2024-05-22 ...
15	2	420.57	0.07	3.81	64.37	2.65	7.76	8.61	2.25	2024-05-22 ...
24	2	423.91	0.04	3.98	63.66	2.89	7.53	8.56	2.25	2024-05-22 ...
1034	5	1214.98	1.24	19.71	158.79	12.04	88.63	100.39	5.89	2024-05-23 ...
1048	4	921.25	0.52	11.14	158.79	10.13	71.68	79.36	4.94	2024-05-23 ...
1056	4	854.49	0.35	10.37	154.5	10.49	68.22	74.1	4.88	2024-05-23 ...
1072	4	961.3	0.43	17.99	138.76	15.97	71.16	77.74	7.41	2024-05-23 ...
1103	5	1335.14	0.45	33.59	123.02	29.8	93.76	103.08	10.77	2024-05-23 ...
1126	5	2590.18	0.08	74.03	72.24	53.88	179.53	194.2	14.19	2024-05-23 ...

Store procedure COVID1: Thêm dữ liệu vào database

```
BEGIN
    INSERT INTO AirQuality (
        AQI, CO, NO, NO2, O3, SO2, PM2_5, PM10, NH3
    ) VALUES (
        @AQI, @CO, @NO, @NO2, @O3, @SO2, @PM2_5, @PM10, @NH3
    );
END;
```

Cuối cùng em xây dựng giao diện người dùng lấy dữ liệu từ SQL để vẽ biểu đồ

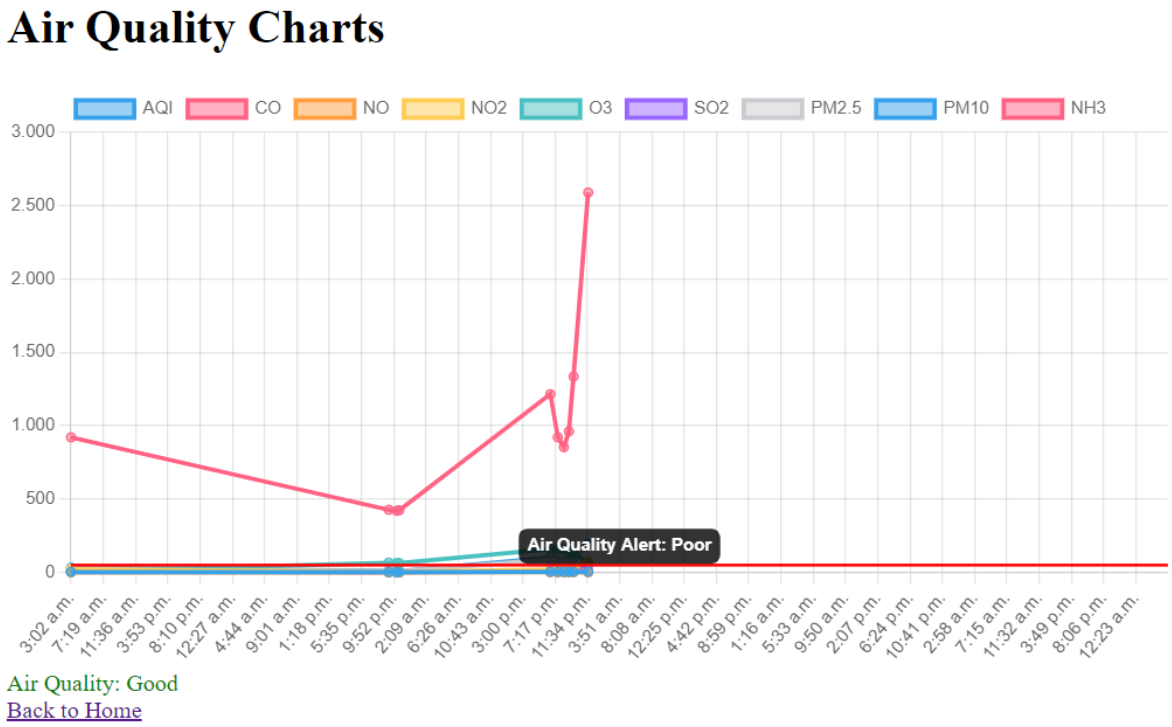
Xử lý dữ liệu và hiển thị biểu đồ lên trang web:



```
conganh
@app.route('/chart')
def show_chart():
    return render_template('chart.html')
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Air Quality Chart</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chartjs-adapter-date-fns@3"></sc
    <script src="https://cdnjs.cloudflare.com/ajax/libs/chartjs-plugin-annotat
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <h1>Air Quality Charts</h1>
    <div style="width: 800px;">
        <canvas id="airQualityChart"></canvas>
    </div>
    <div id="airQualityAlert"></div>
    <a href="/">Back to Home</a>
    <script>
        $(document).ready(function() {
            function fetchData() {
                $.ajax({
                    url: '/data',
                    method: 'GET',
                    success: function(data) {
```

Kết quả trên web



Link sản phẩm:

KẾT LUẬN

Trong bài tập lớn này, em đã xây dựng thành công một hệ thống tự động thu thập, xử lý và hiển thị dữ liệu về không khí từ một nguồn dữ liệu công cộng. Sự kết hợp giữa FastAPI, Node-RED và cơ sở dữ liệu SQL đã tạo ra một giải pháp mạnh mẽ và linh hoạt, giúp chúng ta hiểu rõ hơn về quá trình xây dựng và triển khai các ứng dụng phức tạp.

FastAPI đã cho phép chúng ta dễ dàng tạo ra các API RESTful, giúp giao tiếp giữa các phần của hệ thống trở nên đơn giản và hiệu quả. Node-RED đã đóng vai trò quan trọng trong việc thiết lập các luồng công việc và tích hợp các thành phần khác nhau của hệ thống, từ việc gửi yêu cầu HTTP đến xử lý dữ liệu. Sử dụng cơ sở dữ liệu SQL giúp chúng ta lưu trữ dữ liệu một cách có tổ chức và dễ dàng truy xuất, đảm bảo tính toàn vẹn và ổn định của hệ thống.

Với giao diện người dùng, chúng ta có thể hiển thị dữ liệu một cách trực quan và thân thiện, giúp người dùng dễ dàng nắm bắt thông tin quan trọng về tình hình chất lượng không khí. Hệ thống này cũng có tiềm năng để mở rộng và phát triển trong tương lai, với khả năng tích hợp thêm các tính năng mới và xử lý nhiều loại dữ liệu khác nhau.

Tóm lại, qua bài tập lớn lần em này giúp em hiểu hơn về ngôn ngữ lập trình Python cùng một số công cụ khác như node-red, sql,...

Cuối cùng em xin cảm ơn thầy Đỗ Duy Cốp đã nhiệt tình giúp đỡ để em hoàn thành bài tập lớn này.