TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN MÔN HỌC

LẬP TRÌNH PYTHON

SINH VIÊN THỰC HIỆN: **NGUYỄN THỊ THẢO**

LỚP: **K56KMT.01**

GIÁO VIÊN HƯỚNG DẪN: **ĐỖ DUY CỐP**

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: Nguyễn Thị Thảo Mã SV: K205480106027

II. Tên đề tài

XÂY DỰNG WEBSITE THEO DÕI SỐ CA NHIỄM COVID CỦA THẾ GIỚI

III. Mục tiêu

- Lấy dữ liệu số ca nhiễm covid
- 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 Johns Hopkins University.Lấy dữ liệu số ca mắc covid từ : https://disease.sh/v3/covid-19/countries (có update realtime)
2. Tạo một cơ sở dữ liệu trong SQL Server để lưu trữ dữ liệu COVID-19
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 số ca nhiễm 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 số ca nhiễm của các quốc gia bao gồm: số ca nhiễm, số ca tử vong, số ca hồi phục, số ca đang điều trị.

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

1.1. Đặt vấn đề

Đại dịch COVID-19 đã ảnh hưởng sâu rộng đến mọi mặt của cuộc sống trên toàn thế giới, gây ra nhiều thách thức lớn cho hệ thống y tế, kinh tế và xã hội. Việc theo dõi và quản lý số ca nhiễm COVID-19 là một nhiệm vụ cấp bách và quan trọng để ngăn chặn sự lây lan của virus, đồng thời hỗ trợ các cơ quan chức năng trong việc ra quyết định.

Tuy nhiên, việc quan sát tổng thể các ca nhiễm, ca tử vong và ca hồi phục hiện đang gặp nhiều khó khăn do dữ liệu phân tán và không đồng nhất. Điều này gây cản trở trong việc phân tích và dự báo tình hình dịch bệnh, cũng như trong việc cung cấp thông tin kịp thời và chính xác cho người dân và các nhà hoạch định chính sách.

Do đó, cần phải giải quyết vấn đề này một cách nhanh chóng và hiệu quả bằng cách xây dựng một hệ thống thu thập, xử lý và hiển thị dữ liệu COVID-19 toàn diện, giúp cải thiện khả năng quản lý và ứng phó với đại dịch.

1.2. Mục tiêu

Để giải quyết vấn đề này, cần thiết phải xây dựng một hệ thống thu thập, xử lý và hiển thị dữ liệu số ca nhiễm COVID-19 một cách hiệu quả và chính xác. Hệ thống này không chỉ giúp cung cấp thông tin kịp thời cho người dân mà còn hỗ trợ các nhà nghiên cứu và cơ quan y tế trong việc phân tích và dự báo tình hình dịch bệnh.

Mục tiêu của đề tài này là phát triển một giải pháp toàn diện bao gồm các bước sau:

- Thu thập dữ liệu số ca nhiễm COVID-19: Dữ liệu sẽ được lấy từ các nguồn uy tín và được cập nhật liên tục để đảm bảo tính chính xác và kịp thời.
- Xử lý dữ liệu: Sử dụng FastAPI để xây dựng các API dịch vụ và Node-RED để tổ chức các luồng xử lý dữ liệu, giúp dữ liệu được chuẩn hóa.
- Lưu trữ dữ liệu: Dữ liệu sau khi được xử lý sẽ được lưu trữ vào cơ sở dữ liệu SQL, đảm bảo khả năng truy xuất và quản lý dữ liệu một cách hiệu quả.
- Xây dựng trang web: Một trang web thân thiện với người dùng sẽ được phát triển để hiển thị thông tin chi tiết về số ca nhiễm. Trang web sẽ bao gồm các biểu đồ trực quan giúp người dùng dễ dàng theo dõi tình hình dịch bệnh.

Để đảm bảo tính chính xác và hiệu quả trong việc lưu trữ và hiển thị dữ liệu, chúng ta sẽ sử dụng một cơ sở dữ liệu SQL. Cơ sở dữ liệu này sẽ lưu trữ các thông tin liên quan đến số ca nhiễm, giúp cho việc truy xuất và phân tích dữ liệu trở nên thuận tiện hơn. Ngoài ra, các biểu đồ trực quan sẽ được vẽ dựa trên dữ liệu từ cơ sở dữ liệu, giúp người dùng có cái nhìn tổng quan và dễ hiểu về diễn biến của dịch bệnh.

Với giải pháp này, chúng tôi hy vọng sẽ góp phần nâng cao hiệu quả trong công tác phòng chống dịch COVID-19, đồng thời cung cấp một công cụ hữu ích cho cộng đồng và các cơ quan chức năng.

1.3. Hướng giải quyết

Để giải quyết vấn đề quản lý dữ liệu số ca nhiễm COVID-19 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 số ca nhiễm COVID-19 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 lai 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. Visual Studio 2022

Microsoft Visual Studio là môi trường phát triển tích hợp (IDE) được thiết kế dành cho giới lập trình viên và các nhà phát triển ứng dụng. Đây là công cụ hỗ trợ phát triển phần mềm mạnh mẽ của Microsoft, cho phép người dùng viết, dịch mã và gỡ lỗi các ứng dụng dựa trên nhiều ngôn ngữ lập trình khác nhau như C++, C#, Visual Basic, Python, JavaScript... Visual Studio bao gồm một trình biên tập mã nguồn, các công cụ gỡ lỗi và xây dựng ứng dụng đa nền tảng. Nó giúp tăng năng suất và hiệu quả công việc cho các lập trình viên.

Microsoft Visual Studio nổi bật với khả năng hỗ trợ một loạt các ngôn ngữ lập trình, bao gồm JavaScript, TypeScript, Python, C#, Java, Go, Ruby... Điều này biến nó trở thành công cụ lý tưởng cho các nhà phát triển làm việc trên nhiều dự án với đa ngôn ngữ lập trình. Ngoài ra, khả năng hỗ trợ đa ngôn ngữ của Microsoft Visual Studio còn giúp giới lập trình viên dễ dàng chuyển đổi giữa các ngôn ngữ và dự án mà không cần phải thay đổi môi trường làm việc, từ đó tiết kiệm thời gian, đồng thời tối ưu hiệu quả công việc.

Visual Studio 2022 là một phiên bản mới nhất của môi trường phát triển tích hợp (IDE) Visual Studio, được phát triển bởi Microsoft. Được công bố vào tháng 11 năm 2021, Visual Studio 2022 mang đến nhiều cải tiến và tính năng mới so với các phiên bản trước đó.

Visual Studio 2022 đi kèm với hỗ trợ đầy đủ cho .NET 6, bao gồm C# 10 và F# 6. .NET 6 là một phiên bản mới của nền tảng phát triển phần mềm .NET, với nhiều cải tiến về hiệu suất, độ ổn định và tính năng mới. Visual Studio 2022 tích hợp với GitHub Codespaces, cho phép bạn phát triển ứng dụng trực tiếp từ trình duyệt web mà không cần cài đặt môi trường phát triển trên máy cục bộ.

Visual Studio 2022 được tối ưu hóa về hiệu suất, bao gồm tăng tốc khởi động và thời gian phản hồi của các tính năng và công cụ.

Visual Studio 2022 là một bước tiến quan trọng trong việc cung cấp một môi trường phát triển hiệu quả và mạnh mẽ cho các nhà phát triển phần mềm, với sự hỗ trợ đa nền tảng và tích hợp sâu sắc với các công nghệ và dịch vụ mới.

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 COVID-19 từ một API công cộng và trả về dưới dạng JSON thông qua một API endpoint.

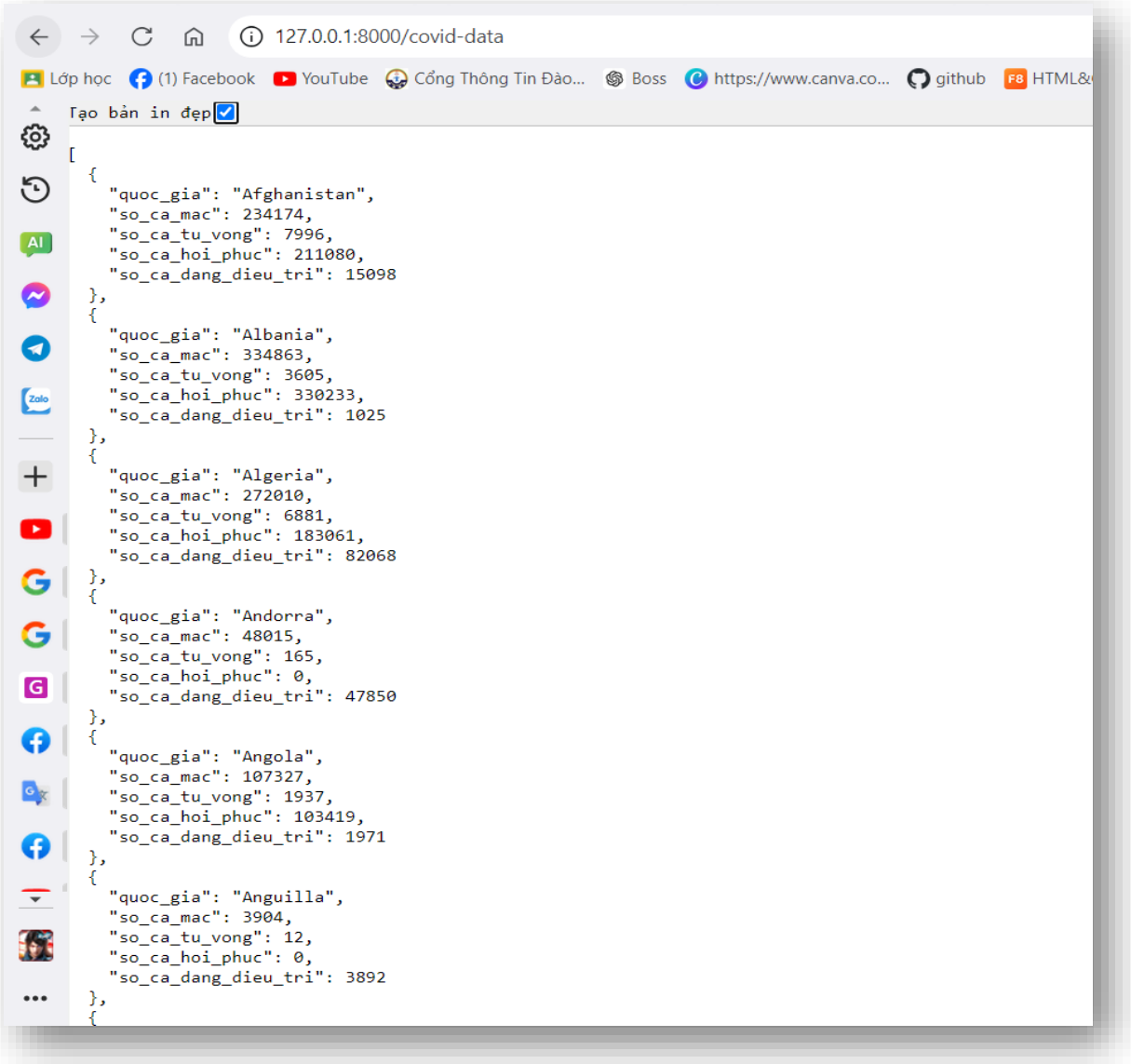
main.py

```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()
```

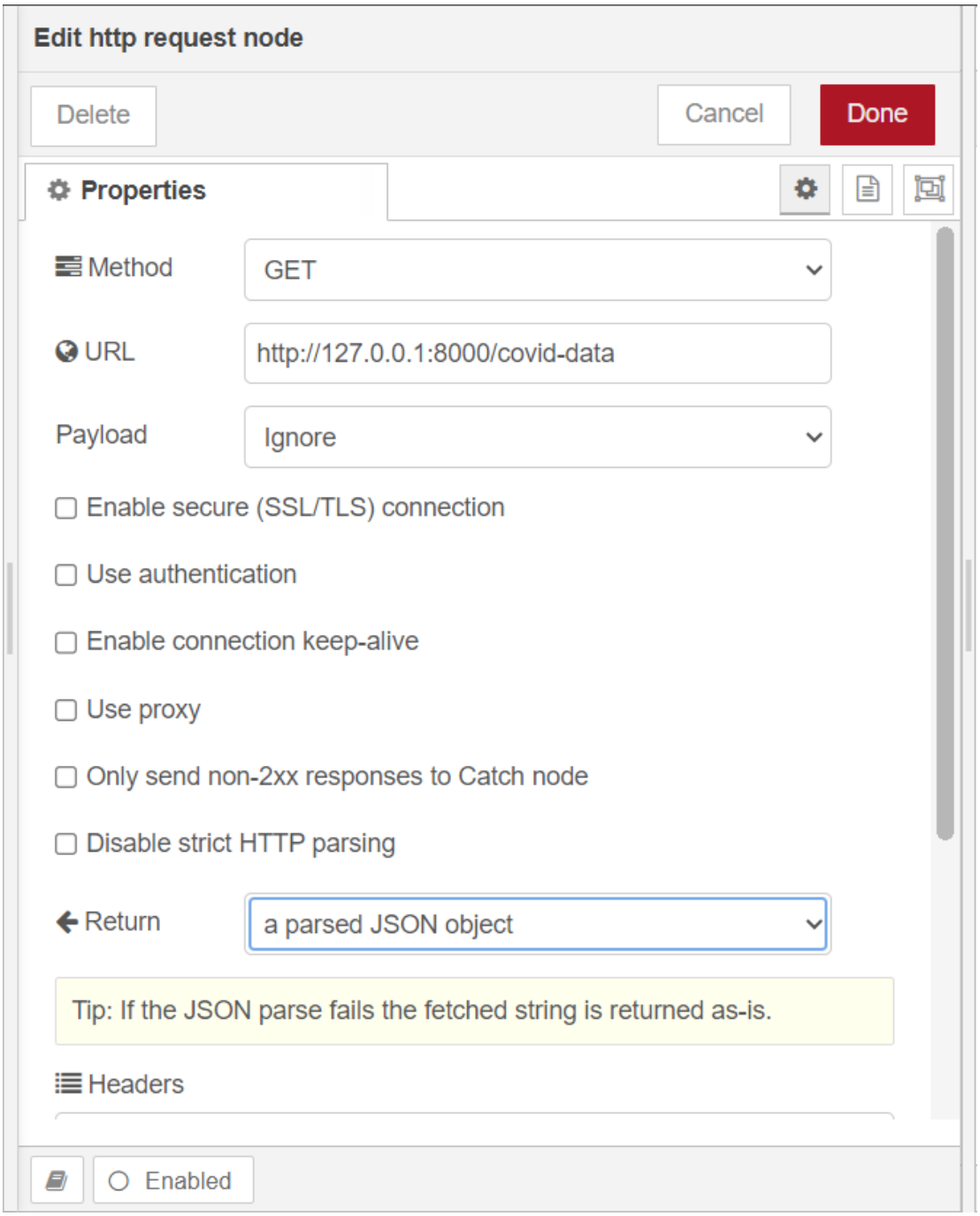
Khởi chạy FastAPI

```
Terminal    Local

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

(.venv) PS F:\LTPYTHON\COVID> uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['F:\\LTPYTHON\\COVID']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [32736] using WatchFiles
INFO:     Started server process [16996]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

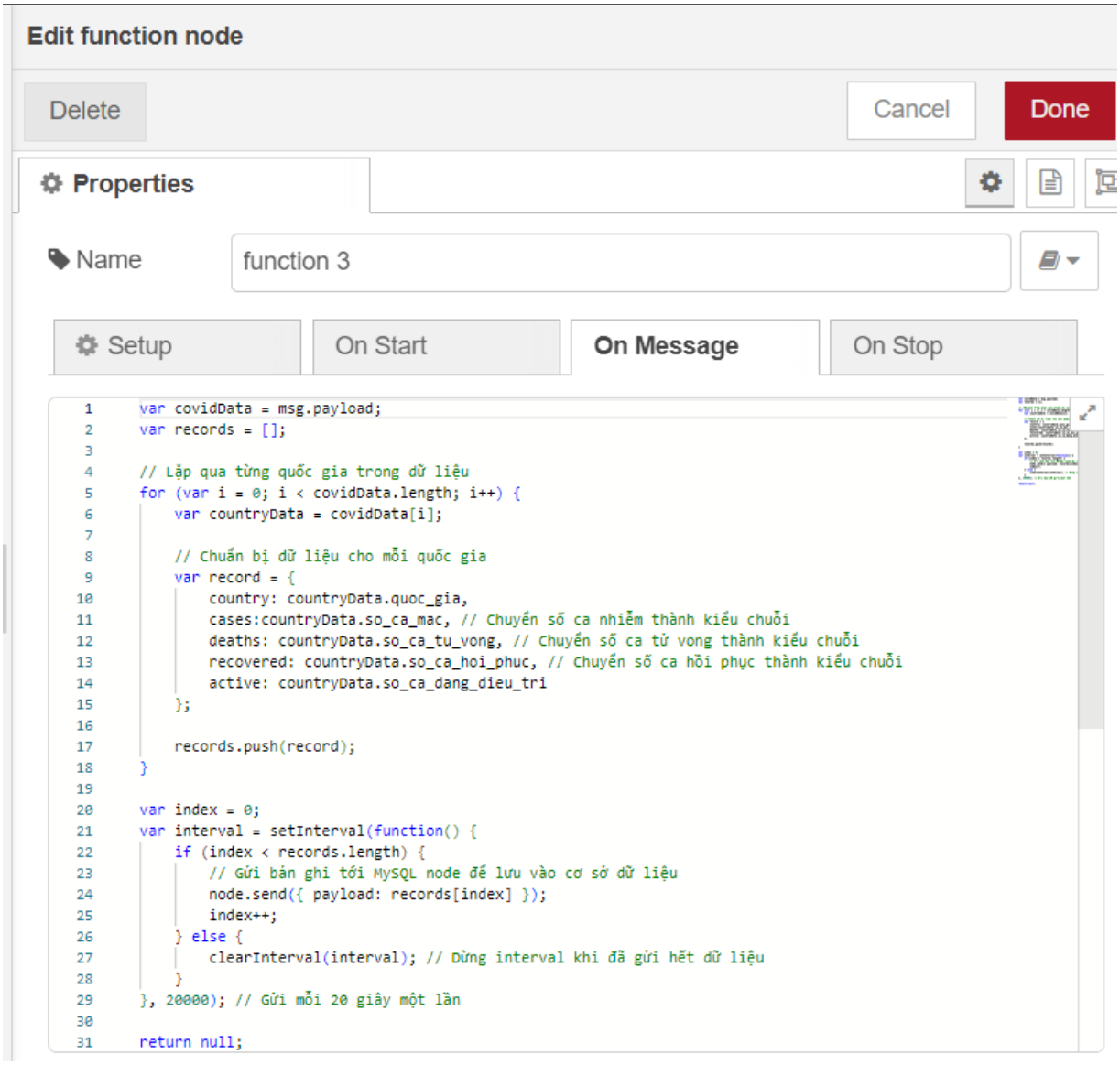
Sau khi khởi chạy nó sẽ trả về một chuỗi dạng json trên local của mình



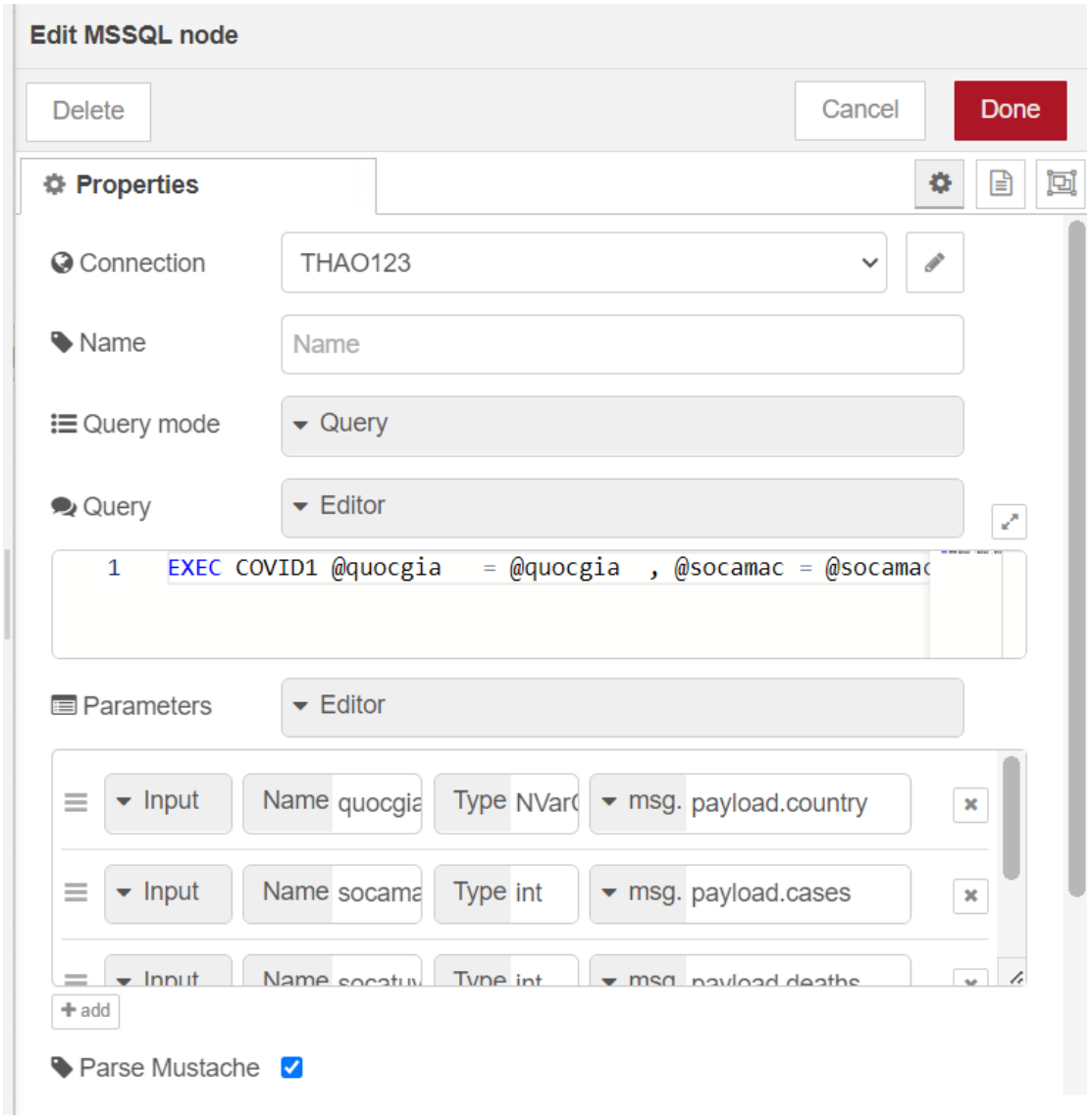
Ở Node-RED dùng node https request lấy địa chỉ có chứa endpoint. Thiết lập một flow trong Node-RED để gửi HTTP request tới endpoint địa phương (localhost)



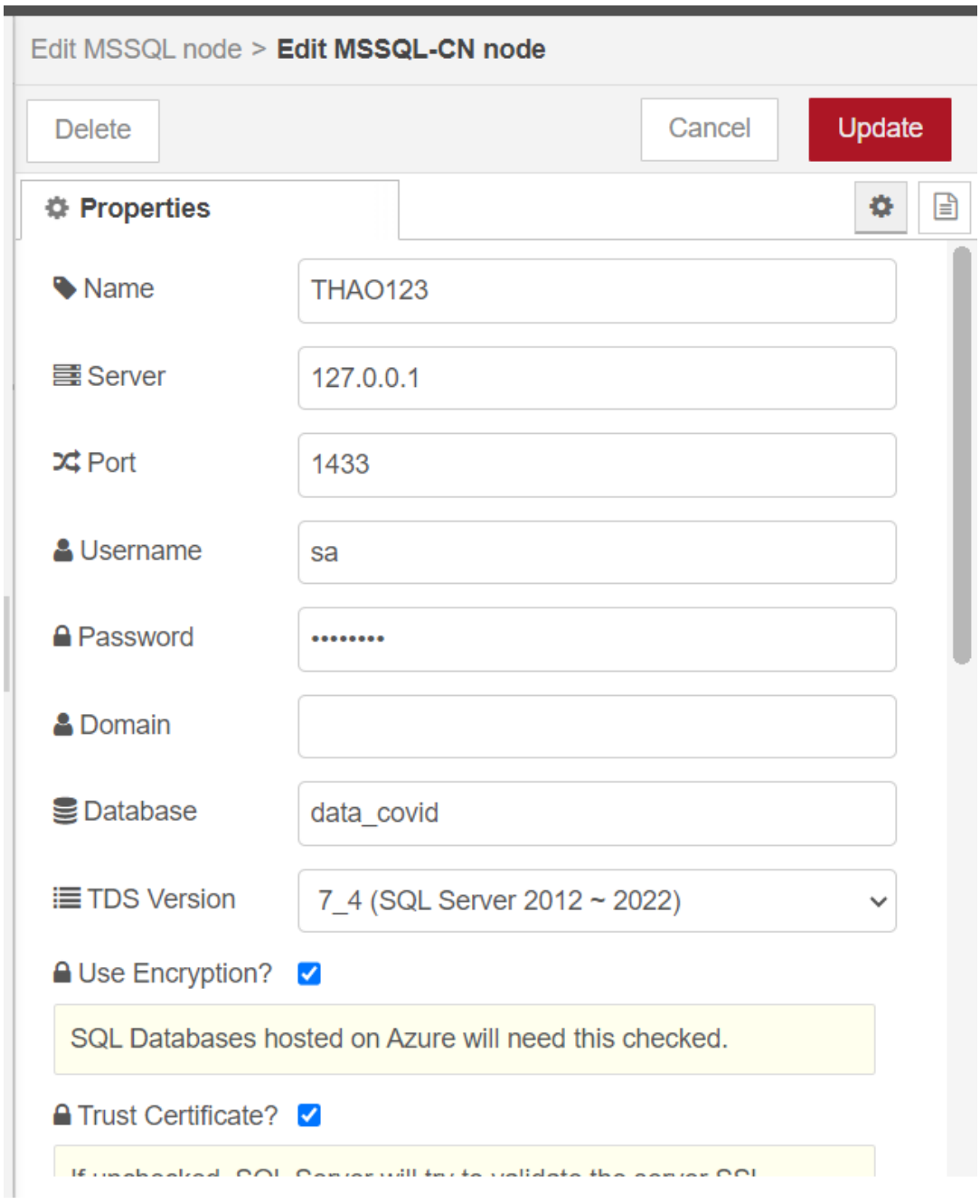
Tiếp đó em viết một function để xử lí dữ liệu và trả về json



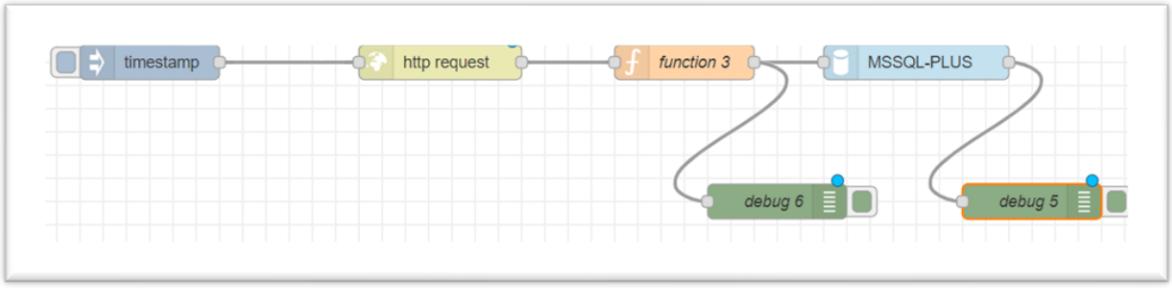
Thực thi proceduce để thêm dữ liệu vào SQL



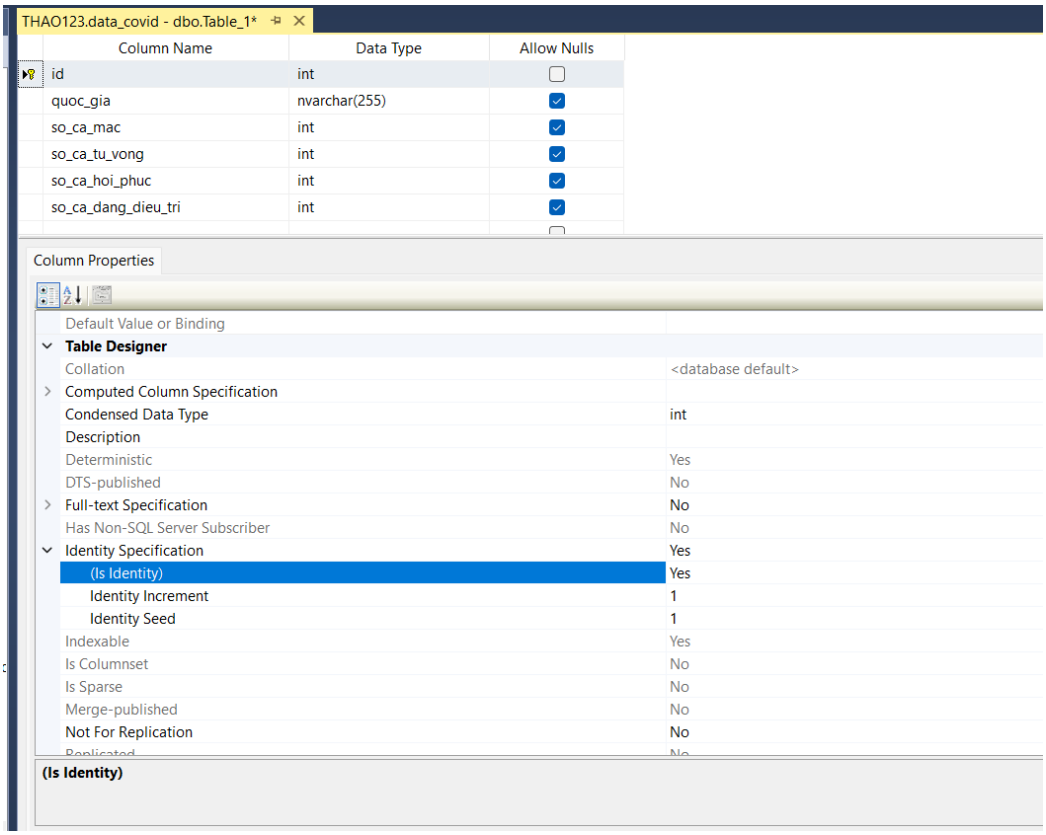
Cài đặt NODE-RED- MSSQL-PLUS: sau khi cài đặt cấu hình các thông tin cho node



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:



Dữ liệu được lưu vào dbo.dulieu

THAO123.data_covid - dbo.dulieu | SQLQuery2.sql - TH...ata_covid (sa (76)) | SQL

quoc_gia	so_ca_mac	so_ca_tu_vong	so_ca_hoi_phuc	so_ca_dang_dieu_...
Azerbaijan	835234	10400	824089	745
Bahamas	38084	844	36366	874
Bahrain	729549	1574	727915	60
Afghanistan	234174	7996	211080	15098
Albania	334863	3605	330233	1025
Algeria	272010	6881	183061	82068
Andorra	48015	165	0	47850
Angola	107327	1937	103419	1971
Anguilla	3904	12	0	3892
Antigua and...	9106	146	8954	6
Argentina	10128845	130841	9997258	746
Armenia	451831	8777	435162	7892
Aruba	44224	292	42438	1494
Australia	11853144	24414	11820014	8716
Austria	6081287	22542	6054934	3811
Azerbaijan	835234	10400	824089	745
Bahamas	38084	844	36366	874
Bahrain	729549	1574	727915	60
Bangladesh	2049377	29493	0	2019884
Barbados	110578	648	108647	1283
Belarus	994037	7118	985592	1327
Belgium	4861695	34376	4826798	521
Belize	71409	688	0	70721
Benin	28036	163	27847	26
Bermuda	18860	165	18685	10
Bhutan	62697	21	61564	1112
Bolivia	1212131	22407	1177145	12579
Bosnia	403615	16388	379084	8143
Botswana	330638	2801	327049	788

Store procedure COVID1: Thêm dữ liệu vào database

```
USE [data_covid]
GO
/****** Object:  StoredProcedure [dbo].[COVID1]    Script Date: 5/25/2024 12:25:32 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[COVID1]
    -- Add the parameters for the stored procedure here

    @quocgia nvarchar(50) = null,
    @socamac int = null,
    @socatuvong int = null ,
    @socahoiphuc int null,
    @socadangdieutri int = null

AS
BEGIN
    DECLARE @json nvarchar(max) ='';

        BEGIN
            BEGIN
                INSERT INTO dulieu(quoc_gia , so_ca_mac, so_ca_tu_vong , so_ca_hoi_phuc , so_ca_dang_dieu_tri ) VALUES (@quocgia  , @socamac ,
                @socatuvong  , @socahoiphuc ,@socadangdieutri  )

                SELECT @json+=FORMATMESSAGE(N'{Thêm dữ liệu thành công}')
                IF((@json is null)or(@json=''))
                    SELECT N'{"ok":0,"msg":"không có dữ liệu","data":[]}' as json;
                ELSE
                  BEGIN
                    SELECT @json=REPLACE(@json,'(null)','null')
                    SELECT N'{"ok":1,"msg":"ok","data":['+left(@json,len(@json)-1)+']}' as json;
                  END
            END
        END
END
```

Store procedure COVID2: Lấy dữ liệu ra

```
USE [data_covid]
GO
/****** Object:  StoredProcedure [dbo].[COVID2]    Script Date: 5/25/2024 12:25:53 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER  PROCEDURE [dbo].[COVID2]
    @action nvarchar(50)
AS
BEGIN
    DECLARE @json nvarchar(max) ='';
        BEGIN
        if(@action ='LIST')
            BEGIN
                SELECT @json+=FORMATMESSAGE(N'{"ID":%d,"QUOCGIA":"%s","SOCAMAC":%d,"SOCATUVONG":%d,"SOCAHOIPHUC":%d,"SOCADANGDIEUTRI":%d},',
                                id,quoc_gia, so_ca_mac, so_ca_tu_vong,so_ca_hoi_phuc, so_ca_dang_dieu_tri )
                FROM dulieu

                IF((@json is null)or(@json=''))
                    SELECT N'{"ok":0,"msg":"không có dữ liệu","data":[]}' as json;
                ELSE
                  BEGIN
                    SELECT @json=REPLACE(@json,'(null)','null')
                    SELECT N'{"ok":1,"msg":"ok","data":['+left(@json,len(@json)-1)+']}' as json;
                  END
            END
        END
    END
```

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

- Em sử dụng html,css,js để lấy dữ liệu từ sqlserver thông qua asp.net (api.aspx)
  Kết nối với SQL

```
<appSettings />
<connectionStrings>
    <add name="ConnectionString" connectionString="Data Source=THAO123;Initial Catalog=data_c
</connectionStrings>
```

Gửi yêu cầu đến máy chủ để thực thi và lấy dữ liệu từ database thông qua api.aspx

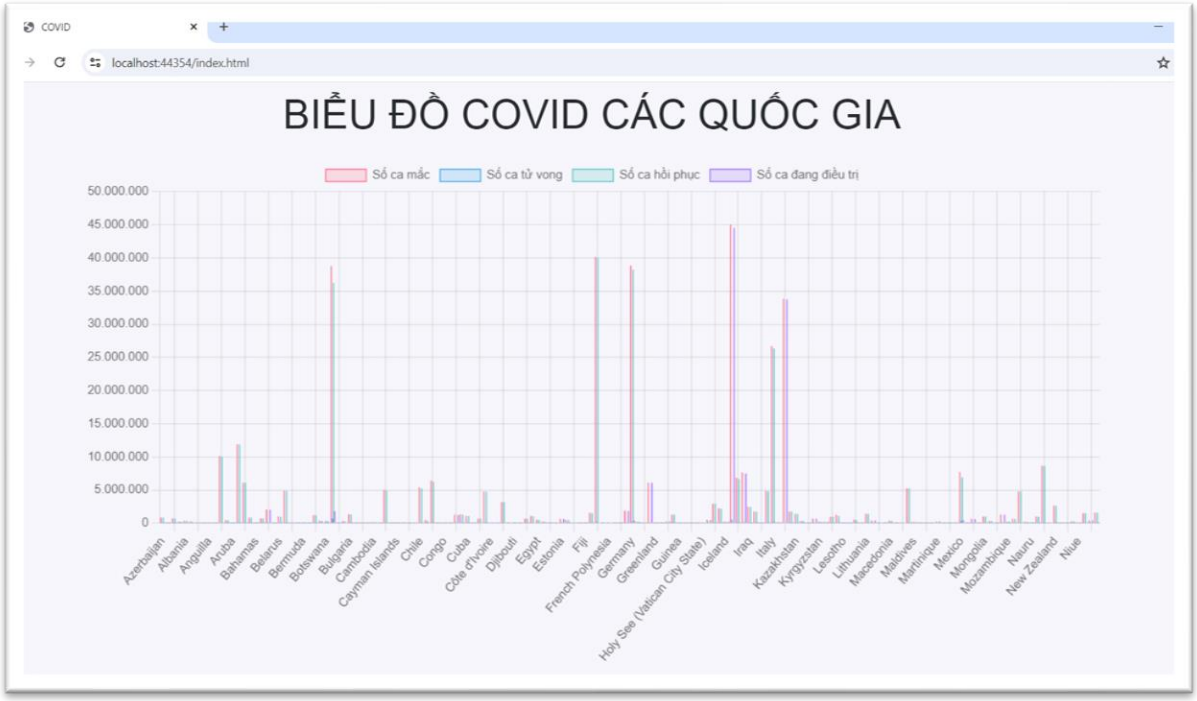
```
$(document).ready(function () {
    const api = '/api.aspx';
    var data_dui_di = {
        action: 'LIST'
    };

    3 references
    $.post(api, data_dui_di, function (data) {
        var json = JSON.parse(data);
        console.log(data)
        if (json.ok === 1) {
            var labels = [];
            var soCaMac = [];
            var soCaTuVong = [];
            var soCaHoiPhuc = [];
            var soCaDangDieuTri = [];
```

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

```
<body>
    <h1>BIỂU ĐỒ COVID CÁC QUỐC GIA</h1>
    <div class="chart-container">
        <canvas id="covidChart"></canvas>
    </div>
    <script>
        $(document).ready(function () {
            const api = '/api.aspx';
            var data_dui_di = {
                action: 'LIST'
            };

            $.post(api, data_dui_di, function (data) {
                var json = JSON.parse(data);
                console.log(data)
                if (json.ok === 1) {
                    var labels = [];
                    var soCaMac = [];
                    var soCaTuVong = [];
                    var soCaHoiPhuc = [];
                    var soCaDangDieuTri = [];

                    for (var cov of json.data) {
                        labels.push(cov.QUOCGIA);
                        soCaMac.push(cov.SOCAMAC);
                        soCaTuVong.push(cov.SOCATUVONG);
                        soCaHoiPhuc.push(cov.SOCAHOIPHUC);
                        soCaDangDieuTri.push(cov.SOCADANGDIEUTRI);
                    }
```

```
const ctx = document.getElementById('covidChart').getContext('2d');
const covidChart = new Chart(ctx, {
    type: 'bar',
    data: {
        labels: labels,
        datasets: [
            {
                label: 'Số ca mắc',
                data: soCaMac,
                backgroundColor: 'rgba(255, 99, 132, 0.2)',
                borderColor: 'rgba(255, 99, 132, 1)',
                borderWidth: 1
            },
            {
                label: 'Số ca tử vong',
                data: soCaTuVong,
                backgroundColor: 'rgba(54, 162, 235, 0.2)',
                borderColor: 'rgba(54, 162, 235, 1)',
                borderWidth: 1
            },
            {
                label: 'Số ca hồi phục',
                data: soCaHoiPhuc,
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                borderColor: 'rgba(75, 192, 192, 1)',
                borderWidth: 1
            },
            {
                label: 'Số ca đang điều trị',
                data: soCaDangDieuTri,
                backgroundColor: 'rgba(153, 102, 255, 0.2)',
                borderColor: 'rgba(153, 102, 255, 1)',
                borderWidth: 1
            }
        ]
    },
    options: {
        scales: {
```

Kết quả trên web



IV. 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 COVID-19 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 dịch bệnh. 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.