

**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: Hoàng Mạnh Tiến

LỚP: K56KMT.01

GIÁO VIÊN HƯỚNG DẪN: ĐỖ DUY CỐP

Thái Nguyên 2024

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

I. Thông tin sinh viên

Sinh viên thực hiện: Hoàng Mạnh Tiến

Mã SV: K205480106025

II. Tên đề tài

XÂY DỰNG WEBSITE THEO DÕI DÂN SỐ CỦA THẾ GIỚI

III. Mục tiêu

- Lấy dữ liệu dân số thế giới
- 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 dân số
2. Tạo một cơ sở dữ liệu trong SQL Server để lưu trữ dữ liệu dân số
3. Sử dụng python để lấy dữ liệu và xử lý dữ liệu sau đó dùng FastAPI gửi dữ liệu đi .
4. Sử dụng Node-red lấy dữ liệu từ FastAPI trước đó thêm vào database
5. Sử dụng các công nghệ front-end (HTML, CSS, JavaScript, React.js) để xây dựng giao diện hiển thị biểu đồ dân số thế giới.

V. Ngày giao nhiệm vụ: 15/5/2024

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

MỤC LỤC	4
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	7
II. CƠ SỞ LÝ THUYẾT.....	8
2.1. SQL Server Management Studio	8
2.2. Ngôn ngữ lập trình Python	9
2.3. Visual Studio 2022.....	9
2.4. Node_red	11
III. NỘI DUNG THỰC HIỆN	12
IV. KẾT LUẬN.....	24

I. TỔNG QUAN CHUNG

1.1. Đặt vấn đề

Dữ liệu dân số thế giới là một yếu tố quan trọng trong việc phát triển các chính sách kinh tế, xã hội và y tế trên toàn cầu. Số liệu dân số không chỉ phản ánh quy mô và mật độ dân số của các quốc gia, mà còn giúp chúng ta hiểu rõ hơn về xu hướng tăng trưởng dân số, cơ cấu dân số theo độ tuổi, giới tính và các yếu tố khác.

Tuy nhiên, việc theo dõi và quản lý dữ liệu dân số thế giới đang gặp nhiều thách thức lớn. Dữ liệu từ các nguồn khác nhau thường không đồng nhất và phân tán, gây khó khăn trong việc thu thập, xử lý và phân tích thông tin. Điều này cản trở quá trình đưa ra các quyết định chính xác và kịp thời cho các nhà hoạch định chính sách, các tổ chức quốc tế và cả các nhà nghiên cứu.

Để giải quyết vấn đề này, cần xây dựng một hệ thống toàn diện để thu thập, xử lý và hiển thị dữ liệu dân số thế giới. Hệ thống này sẽ giúp cải thiện khả năng quản lý dữ liệu dân số, hỗ trợ việc phân tích và dự báo các xu hướng dân số, từ đó đưa ra các quyết định chính sách hiệu quả và phù hợp hơn. Mục tiêu của đề tài là tạo ra một công cụ trực quan và tiện dụng, giúp cung cấp thông tin chính xác và cập nhật về dân số thế giới cho tất cả các bên liên quan.

1.2. Mục tiêu

Để giải quyết vấn đề quản lý và theo dõi dữ liệu dân số thế giới một cách hiệu quả và chính xác, cần thiết phải xây dựng một hệ thống toàn diện. Hệ thống này không chỉ giúp cung cấp thông tin kịp thời cho người dùng mà còn hỗ trợ các nhà nghiên cứu và cơ quan chính phủ trong việc phân tích và dự báo các xu hướng dân số.

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

Thu thập dữ liệu dân số: 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. Ví dụ, sử dụng API của World Bank hoặc các nguồn dữ liệu dân số đáng tin cậy khác.

-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 và dễ dàng sử dụng.

- 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 liệu dân số. Trang web sẽ bao gồm các biểu đồ trực quan giúp người dùng dễ dàng theo dõi và hiểu rõ các xu hướng dân số của các quốc gia khác nhau.

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 dân số, 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 dân số.

Với giải pháp này, chúng tôi hy vọng sẽ góp phần nâng cao hiệu quả trong việc quản lý và phân tích dữ liệu dân số, đồng thời cung cấp một công cụ hữu ích cho cộng đồng, các nhà nghiên cứu và các cơ quan chức năng.

1.3. Hướng giải quyết

Để giải quyết vấn đề quản lý dữ liệu dân số thế giới 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 đi bằng FastAPI

- Sử dụng Python để gửi các yêu cầu đến API dữ liệu dân số, xử lý dữ liệu sử dụng FastAPI để gửi dữ liệu đi.

b. Node-red ta sử dụng http request để lấy dữ liệu từ FastAPI

- Sử dụng http request lấy dữ liệu từ FastAPI
- Dùng 1 hàm function để xử lý sau mỗi khoảng 5s thì trả về dữ liệu là json 1 lần
- ở MSSQL căn cứ vào dữ liệu hàm function trả về lưu dữ liệu vào database với procedure tương ứng.

c. SQL

- Xây dựng Procedure để lưu dữ liệu từ MSSQL của Node-red yêu cầu.

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

- Sử dụng Frontend bao gồm html, css, js để tạo ra 1 trang web hiển thị biểu đồ dân số
- Sử dụng backend bao gồm ASP.NET và SQL SERVER để lấy dữ liệu từ database đưa lên trang web

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 thức 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.

Các hạn chế cụ thể bao gồm:

- Khả năng mở rộng hệ thống để tích hợp thêm nhiều nguồn dữ liệu khác nhau.
- Tối ưu hóa hiệu năng của hệ thống để xử lý lượng dữ liệu lớn.
- Cải thiện giao diện người dùng để thân thiện và trực quan hơn.
- Tích hợp thêm các tính năng phân tích và dự báo nâng cao.

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:

Cài đặt các thư viện cần sử dụng để request dữ liệu và tạo FastAPI

```
import requests
import json
import uvicorn
from datetime import datetime
from fastapi import FastAPI
```

- Lấy thời gian hiện tại truy vấn vào API dữ liệu dân số lấy về dân của thế giới và các quốc gia theo từng năm và trả về dữ liệu là một chuỗi json

```
8   app = FastAPI()
9   countries = {
10      "World": "1W",
11      "Vietnam": "VN",
12      "China": "CN",
13      "USA": "US",
14      "Japan": "JP",
15      "UK": "GB"
16   }
17   url_template = 'http://api.worldbank.org/v2/country/{}/indicator/SP.POP.TOTL?format=json&date={}'
18   @app.get("/")
19   def get_population_data():
20       # Kết quả lưu trữ dữ liệu dân số
21       population_data = {}
22       current_year = datetime.now().year
23       for country_name, country_code in countries.items():
24           population_data[country_name] = {}
25           for year in range(current_year - 10, current_year):
26               url = url_template.format(*args: country_code, year)
27               response = requests.get(url)
28               data = response.json()
29
30               if len(data) > 1 and 'indicator' in data[1][0]:
31                   population_value = data[1][0]['value']
32                   population_data[country_name][year] = population_value
33               else:
34                   population_data[country_name][year] = None
35       return population_data
```

Sau khi khởi chạy thì FastAPI sẽ trả về chuỗi json .

```
1 {
2   "World": {
3     "2014": 7317040295,
4     "2015": 7403850164,
5     "2016": 7490415449,
6     "2017": 7576441961,
7     "2018": 7660371127,
8     "2019": 7741774583,
9     "2020": 7820205606,
10    "2021": 7888305693,
11    "2022": 7950946801,
12    "2023": null
13  },
14  "Vietnam": {
15    "2014": 91235504,
16    "2015": 92191398,
17    "2016": 93126529,
18    "2017": 94033048,
19    "2018": 94914330,
20    "2019": 95776716,
21    "2020": 96648685,
22    "2021": 97468029,
23    "2022": 98186856,
24    "2023": null
25  },
26  "China": {
27    "2014": 1371860000,
28    "2015": 1379860000,
29    "2016": 1387790000,
30    "2017": 1396215000,
31    "2018": 1402760000,
32    "2019": 1407745000,
33    "2020": 1411100000,
34    "2021": 1412360000,
35    "2022": 1412175000,
36    "2023": null
37  },
38  "USA": {
39    "2014": 318386329,
40    "2015": 320738994,
41    "2016": 323071755,
42    "2017": 325122128,
43    "2018": 326838199,
44    "2019": 328329953,
45    "2020": 331511512,
46    "2021": 332031554,
47    "2022": 333287557,
48    "2023": null
49  },
```

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)

Edit http request node

Delete

Cancel

Done

Properties

Method

GET

▼

URL

http://127.0.0.1:8000/

Payload

Ignore

▼

☐ Enable secure (SSL/TLS) connection

☐ Use authentication

☐ Enable connection keep-alive

☐ Use proxy

☐ Only send non-2xx responses to Catch node

☐ Disable strict HTTP parsing

Return


a parsed JSON object




▼


Tiếp đó em viết một function để xử lý dữ liệu và trả về từng đoạn dữ liệu phục vụ cho việc lưu trữ dữ liệu thuận tiện

```
1  var populationData = msg.payload;
2  var records = [];
3
4  // Lặp qua từng quốc gia trong dữ liệu
5  for (var country in populationData) {
6      if (populationData.hasOwnProperty(country)) {
7          // Lặp qua từng năm trong dữ liệu của mỗi quốc gia
8          for (var year in populationData[country]) {
9              if (populationData[country].hasOwnProperty(year)) {
10                 var population = populationData[country][year];
11                 // Kiểm tra dữ liệu dân số có null không
12                 if (population !== null) {
13                     // Chuẩn bị dữ liệu cho mỗi quốc gia và mỗi năm
14                     var record = {
15                         country: country,
16                         year: year,
17                         population: String(population) // Chuyển dân số thành
18                     };
19                     records.push(record);
20                 }
21             }
22         }
23     }
24 }
25
```


Thực thi procedure để thêm dữ liệu vào SQL


 **Properties**




 Connection

MyDatabase




 Name


Name

 Query mode

▼ Query


 Query

▼ Editor




1

SP_DANSO @action = 'THEM_DU_LIEU', @quocgia = @quocgia , @danso = @dar

 Parameters


▼ Editor


 ▼ Input

Name quocgia

Type NVarc

▼ msg. payload.country





 ▼ Input

Name danso

Type NVarc

▼ msg. payload.population





 ▼ Input


Name nam

Type NVarc

▼ msg. payload.year



 + add

 Parse Mustache


☒



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


Delete

Cancel


Update

 Properties




 Name


MyDatabase

 Server


127.0.0.1

 Port


1433


 Username

tiendzz


 Password


.....


 Domain

 Database

DansoTG

 TDS Version

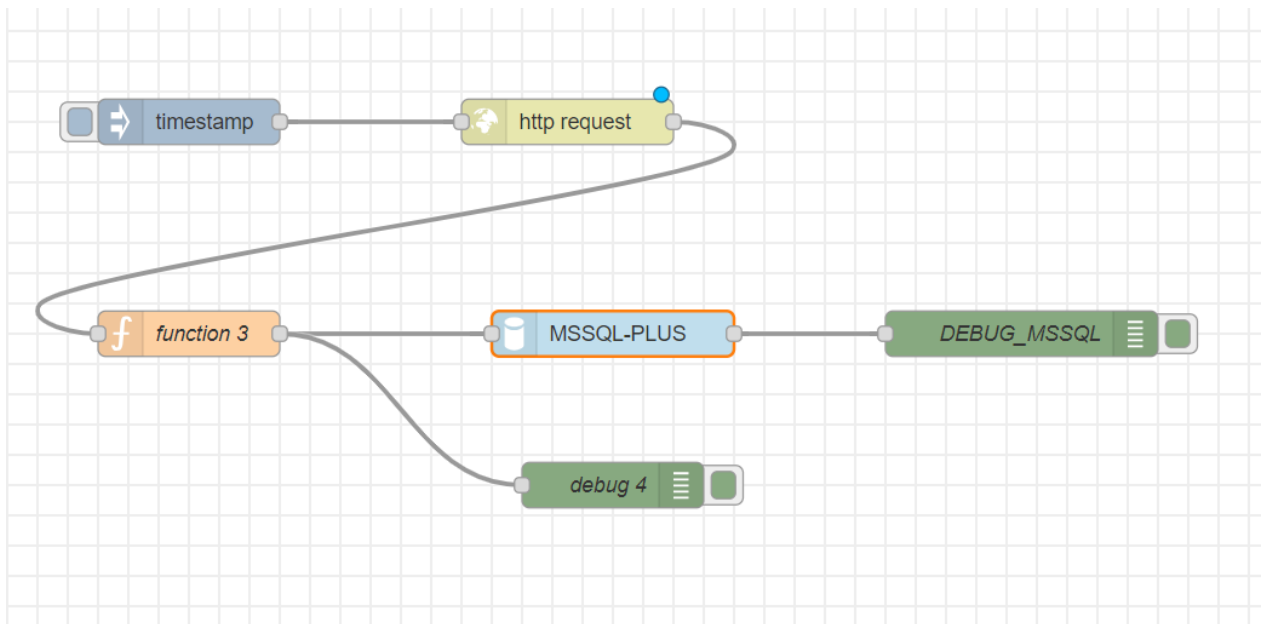
7_4 (SQL Server 2012 ~ 2022) 

 Use Encryption?

☒

SQL Databases hosted on Azure will need this checked.

Sơ đồ lấy dữ liệu và lưu dữ liệu vào database trên node-red



TRÊN SQL :

Tạo bảng lưu dữ liệu .

SQL Server Enterprise 16.0

Query Editor - SQL Server

Execute

DESKTOP-F4L41KI\...oTG - dbo.DAN_SO

Column Name	Data Type	Allow Nulls
id	int	<input type="checkbox"/>
Quoc_gia	nvarchar(50)	<input checked="" type="checkbox"/>
Dan_so	nvarchar(50)	<input checked="" type="checkbox"/>
Nam	nvarchar(50)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Dữ liệu được lưu vào bảng DAN_SO



	id	Quoc_gia	Dan_so	Nam
	28	China	1412175000	2022
	29	USA	318386329	2014
	30	USA	320738994	2015
	31	USA	323071755	2016
	32	USA	325122128	2017
	33	USA	326838199	2018
	34	USA	328329953	2019
	35	USA	331511512	2020
	36	USA	332031554	2021
	37	USA	333287557	2022
	38	Japan	127276000	2014
	39	Japan	127141000	2015
	40	Japan	127076000	2016
	41	Japan	126972000	2017
	42	Japan	126811000	2018
	43	Japan	126633000	2019
	44	Japan	126261000	2020
	45	Japan	125681593	2021
	46	Japan	125124989	2022
	47	UK	64602298	2014
	48	UK	65116219	2015
	49	UK	65611593	2016

Viết procedure để lưu dữ liệu từ node-red và trả về dữ liệu nếu có yêu cầu, sử dụng action để phân biệt đến chức năng mà hàm cần gọi.

```

AS
BEGIN
DECLARE @json nvarchar(max) = '';
IF(@action = 'LIST_DAN_SO')
BEGIN
BEGIN
SELECT @json+=FORMATMESSAGE(N'{"ID":%d,"QUOC_GIA": "%s", "DAN_SO": "%s", "NAM": "%s"},',
ID, Quoc_gia,Dan_so,Nam )
FROM DAN_SO

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
IF(@action = 'THEM_DU_LIEU')
BEGIN
BEGIN
INSERT INTO DAN_SO(Quoc_gia , Dan_so , Nam ) VALUES (@quocgia , @danso ,@nam)

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

```

Xây dựng giao diện website.

Tạo 1 file index.html

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>DanSoTG </title>

<script src="Asset/jquery.min.js"></script>
<script src="Asset/jquery-confirm.min.js"></script>
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<link href="Asset/bootstrap.min.css" rel="stylesheet" />
<link href="Asset/jquery-confirm.min.css" rel="stylesheet" />

<script type="module" src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.esm.js"></script>
<script nomodule src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.js"></script>
<!--<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>-->
<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/1.3.5/jspdf.min.js"></script>

<script src="Asset/MyCode.js"></script>
</head>
<body>
<div class="container">
<h1>Population Data</h1>
<div id="max_population_table"></div>
<div id="chart_div" style="width: 100%; height: 500px;"></div>
</div>
</body>
</html>

```

file js lấy dữ liệu và xử lý dữ liệu

Vị trí hiện thị bảng

vị trí trả về biểu đồ

Kết nối SQL với ASP.NET

```
<appSettings />
<connectionStrings>
  <add name="ConnectionString" connectionString="Data Source=DESKTOP-F4L41KI\SQLEXPRESS;Initial Ca
</connectionStrings>
```

Tạo file API.ASPX để gọi đến SQL SEVER

```
0 references
protected void Page_Load(object sender, EventArgs e)
{
    string action = Request["action"];
    switch (action)
    {
        case "LIST_DAN_SO":
            xu_ly(action);
            break;
    }
}

1 reference
void xu_ly(string action)
{
    SqlServer db = new SqlServer();
    SqlCommand cm = db.GetCmd("SP_DANSO", action); //tạo cm với "SP_Company" và @action từ method GetCmd của db
    switch (action)
    {
        case "LIST_DAN_SO":
            break;
    }

    string json = (string)db.Scalar(cm); //thuc thi SqlCommand cm này để thu về json
    Response.Write(json); //trả json về trình duyệt
}
}
```

Gửi yêu cầu đến API.ASPX để lấy dữ liệu từ SQL SEVER và hiển thị lên web

```
4 references
$(document).ready(function () {
    const api = '/api.aspx';
    4 references
    $.post(api, { action: 'LIST_DAN_SO' }, function (data) {

        var json = JSON.parse(data);
        var rawData = json.data;

        // Tìm dân số lớn nhất của mỗi quốc gia
        var maxPopulationByCountry = {};
        var populationByCountryAndYear = {};

        2 references
        rawData.forEach(function (item) {
            var country = item.QUOC_GIA;
            var year = parseInt(item.NAM, 10) + 2;
            var population = parseInt(item.DAN_SO);

            if (!maxPopulationByCountry[country] || maxPopulationByCountry[country] < population) {
                maxPopulationByCountry[country] = population;
            }

            if (!populationByCountryAndYear[country]) {
                populationByCountryAndYear[country] = {};
            }

            populationByCountryAndYear[country][year] = population;
        });
    });
});
```

```

// Hiển thị bảng dân số lớn nhất của mỗi quốc gia
var tableHtml = '<table class="table table-bordered"><thead><tr><th>Country</th><th>Max Population</th></tr></thead>';
for (var country in maxPopulationByCountry) {
    tableHtml += '<tr><td>' + country + '</td><td>' + maxPopulationByCountry[country] + '</td></tr>';
}
tableHtml += '</tbody></table>';
$('#max_population_table').html(tableHtml);

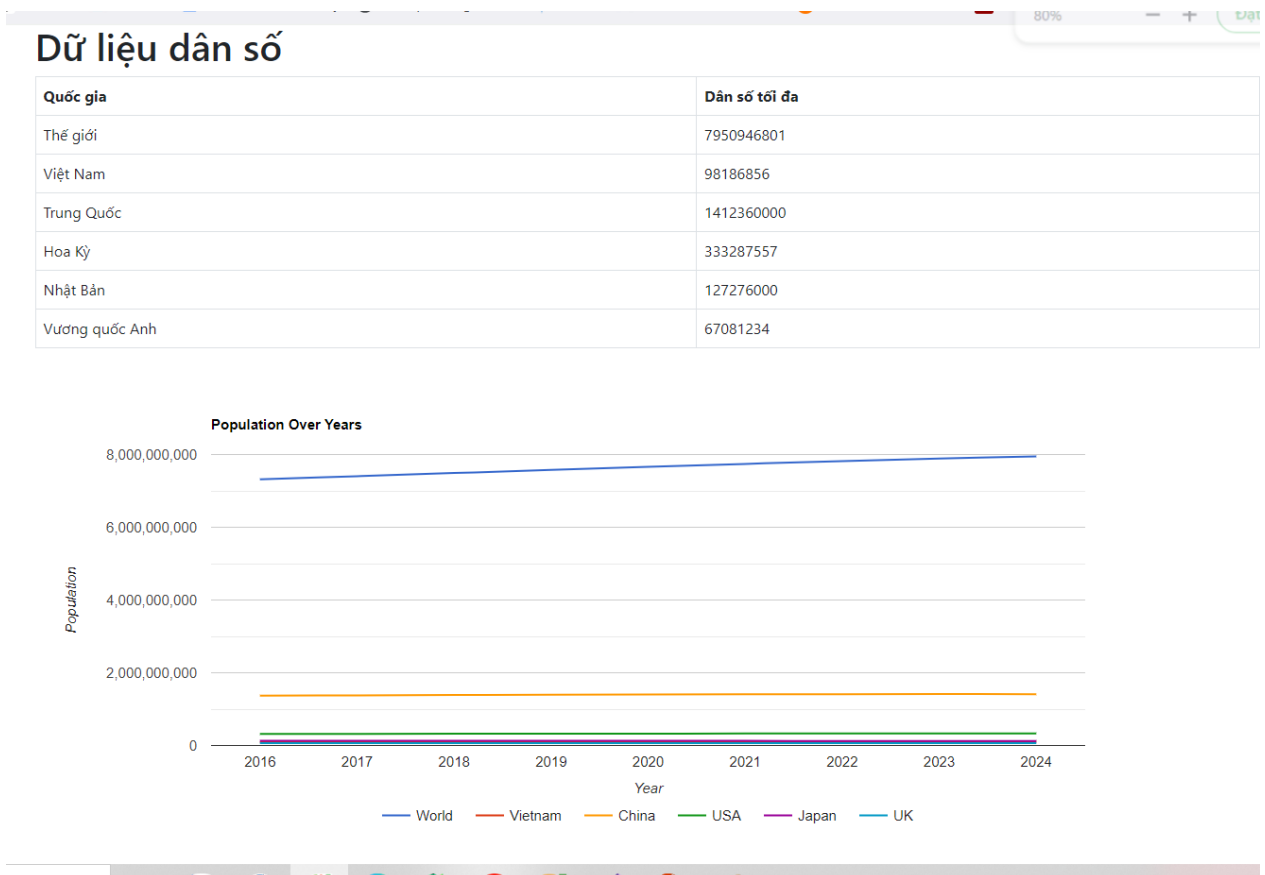
// Chuẩn bị dữ liệu cho biểu đồ
var years = [];
for (var year in populationByCountryAndYear["World"]) {
    years.push(year);
}
years.sort();

var chartData = [['Year']];
for (var country in populationByCountryAndYear) {
    chartData[0].push(country);
}

4 references
years.forEach(function (year) {
    var row = [year];
    for (var country in populationByCountryAndYear) {
        row.push(populationByCountryAndYear[country][year] || 0);
    }
    chartData.push(row);
});

```

Kết quả :



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 dân số 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 dân số. 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 này, em đã hiểu rõ hơn về ngôn ngữ lập trình Python cùng một số công cụ khác như Node-RED và SQL. Em đã học được cách xây dựng một hệ thống từ khâu thu thập dữ liệu đến xử lý và hiển thị dữ liệu một cách hiệu quả.

Cuối cùng, em xin cảm ơn thầy Đỗ Duy Cốp đã nhiệt tình giúp đỡ để em hoàn thành bài tập lớn này. Sự hướng dẫn và hỗ trợ của thầy đã giúp em vượt qua các khó khăn và hoàn thành tốt đề tài của mình. Em hy vọng trong tương lai sẽ có cơ hội phát triển và hoàn thiện hơn những kiến thức và kỹ năng đã học được từ bài tập này.