

Họ và tên : Vi Duy Quốc
Mssv K205480106034

Đề bài : Quản lí nhiệt độ thông minh

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 họa.
3. Mô tả quá trình chạy demo, hiểu được luồng xử lý dữ liệu. Hình ảnh minh họa
4. Kết luận: đã tìm hiểu được những kỹ thuật gì? Đã cài đặt và cấu hình thành công phần mềm nào? Đã tạo đc api gì? Đã phối hợp các kỹ thuật lập trình gì để đạt được điều gì? Kết quả cuối cùng xấu đẹp ra sao?....

Những mục đã làm :

- tạo file python sử dụng fashapi tạo dữ liệu random cho nhiệt độ
- sử dụng node-red lấy dữ liệu từ địa chỉ local của fas api
- sử dụng asp dot net để lấy dữ liệu và vẽ biểu đồ t

Quá trình làm bài :

Tạo file python sử dụng fashapi tạo dữ liệu random cho nhiệt độ
Install các thư viện cần thiết

```

from fastapi import FastAPI
import random
import time

app = FastAPI()

@app.get("/random-data")
async def get_random_data():
    data = {
        "room_id": random.randint(a: 1, b: 5),
        "temperature": round(random.uniform(a: 20.0, b: 30.0), 2),
        "humidity": round(random.uniform(a: 30.0, b: 60.0), 2),
        "timestamp": time.strftime('%Y-%m-%d %H:%M:%S')
    }
    return data

```

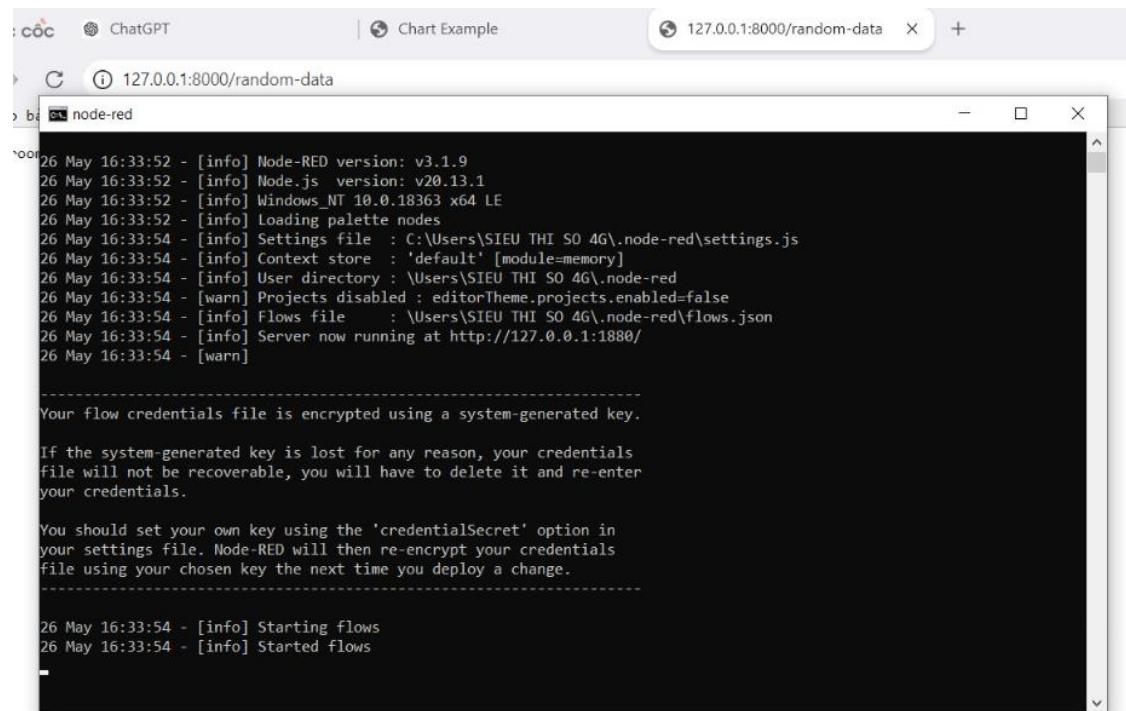
Chạy fast api

```

INFO: Started server process [2456]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://127.0.0.1:8001 (Press CTRL+C to quit)
INFO: 127.0.0.1:64916 - "GET / HTTP/1.1" 404 Not Found
INFO: 127.0.0.1:64916 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO: 127.0.0.1:64916 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:64966 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:64971 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:65006 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:65118 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:65269 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:49245 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:51463 - "GET /forex/vnd HTTP/1.1" 200 OK
INFO: 127.0.0.1:51636 - "GET /forex/vnd HTTP/1.1" 200 OK

```

Bước tiếp theo chúng ta mở CMD chạy node-red



The image shows a web browser window at the top with tabs for 'ChatGPT', 'Chart Example', and '127.0.0.1:8000/random-data'. Below the browser is a terminal window titled 'node-red'. The terminal displays the following logs:

```
26 May 16:33:52 - [info] Node-RED version: v3.1.9
26 May 16:33:52 - [info] Node.js version: v20.13.1
26 May 16:33:52 - [info] Windows_NT 10.0.18363 x64 LE
26 May 16:33:52 - [info] Loading palette nodes
26 May 16:33:54 - [info] Settings file : C:\Users\SIEU THI SO 4G\.node-red\settings.js
26 May 16:33:54 - [info] Context store : 'default' [module=memory]
26 May 16:33:54 - [info] User directory : \Users\SIEU THI SO 4G\.node-red
26 May 16:33:54 - [warn] Projects disabled : editorTheme.projects.enabled=false
26 May 16:33:54 - [info] Flows file : \Users\SIEU THI SO 4G\.node-red\flows.json
26 May 16:33:54 - [info] Server now running at http://127.0.0.1:1880/
26 May 16:33:54 - [warn]

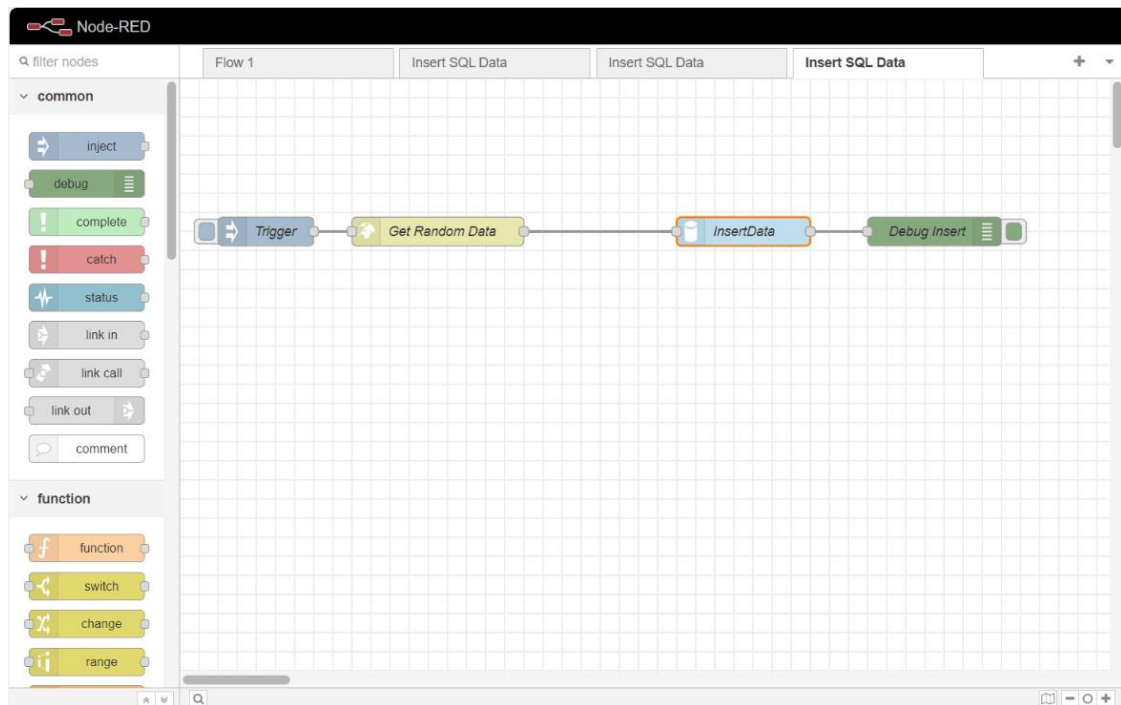
-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----

26 May 16:33:54 - [info] Starting flows
26 May 16:33:54 - [info] Started flows
```

Đây là cấu trúc node-red của bài



Tiếp theo cài 1 phút lấy dữ liệu 1 lần

Edit inject node

Delete Cancel Done

Properties

Name: Name

msg.payload = timestamp

msg.topic = a_z

+ add inject now

☐ Inject once after 0.1 seconds, then

Repeat: interval

every 1 minutes

Enabled

Điền địa chỉ api mình chạy fast Api

Edit http request node

Delete Cancel Done

Properties

Method: GET

URL: http://127.0.0.1:8000/random-data

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

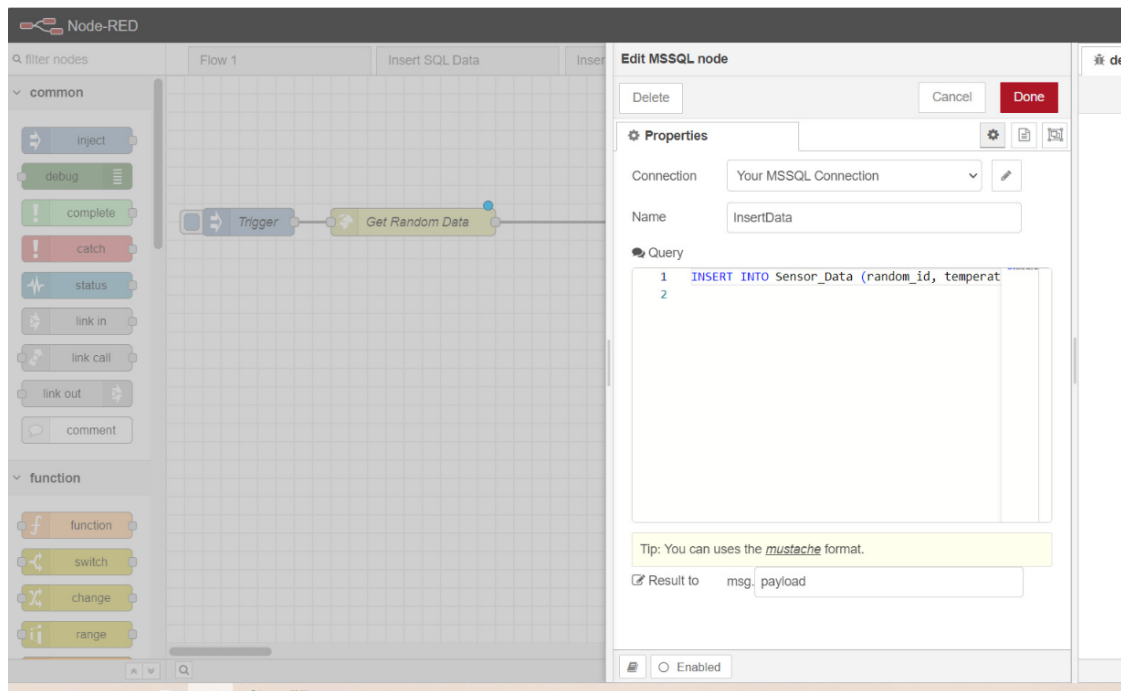
Return: a parsed JSON object

Tip: If the JSON parse fails the fetched string is returned as-is.

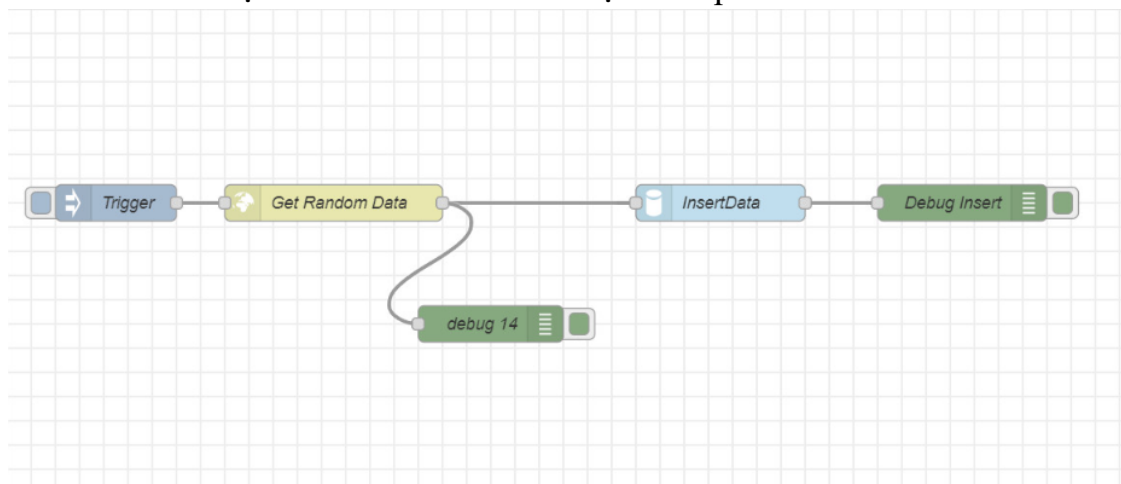
Headers

Enabled

Cài đặt node-red-contrib-mssql-plus ---> cài đặt cấu hình cho node

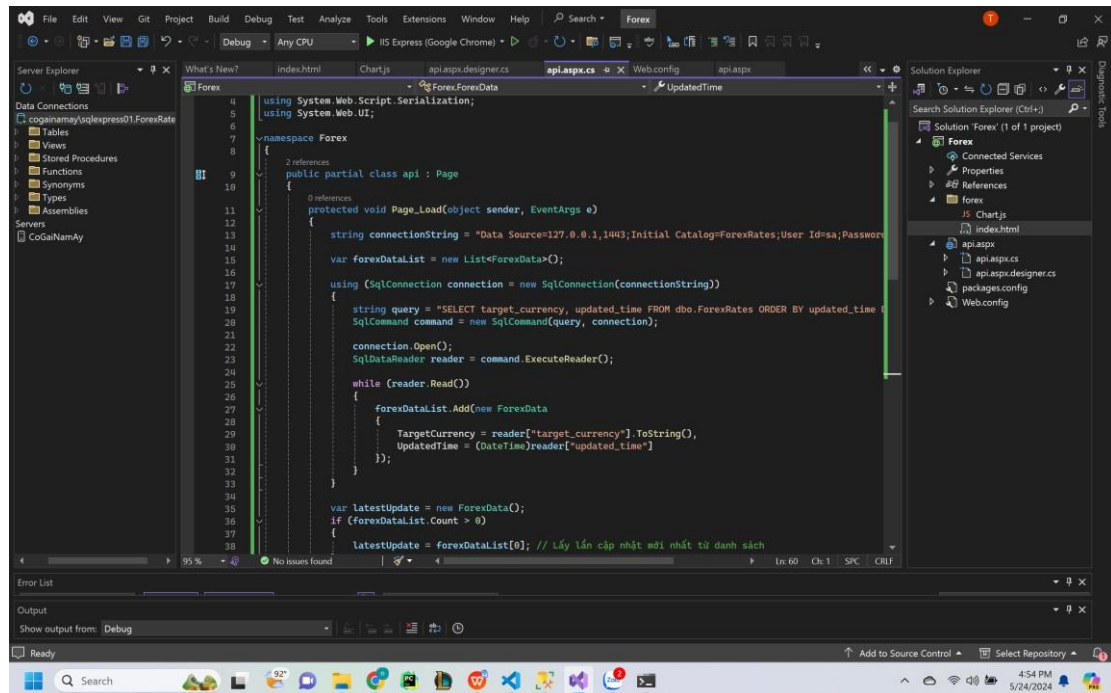


Cấu trúc của đoạn node-red để lưu dữ liệu về sql



Sau đó em viết SP trả dữ liệu dạng json rồi đẩy lên web

Tiếp theo em kết nối SQL với VS studio lấy dữ liệu lần cập nhật cuối cùng và dữ liệu tiền và datetime đưa lên web bằng APS.NET

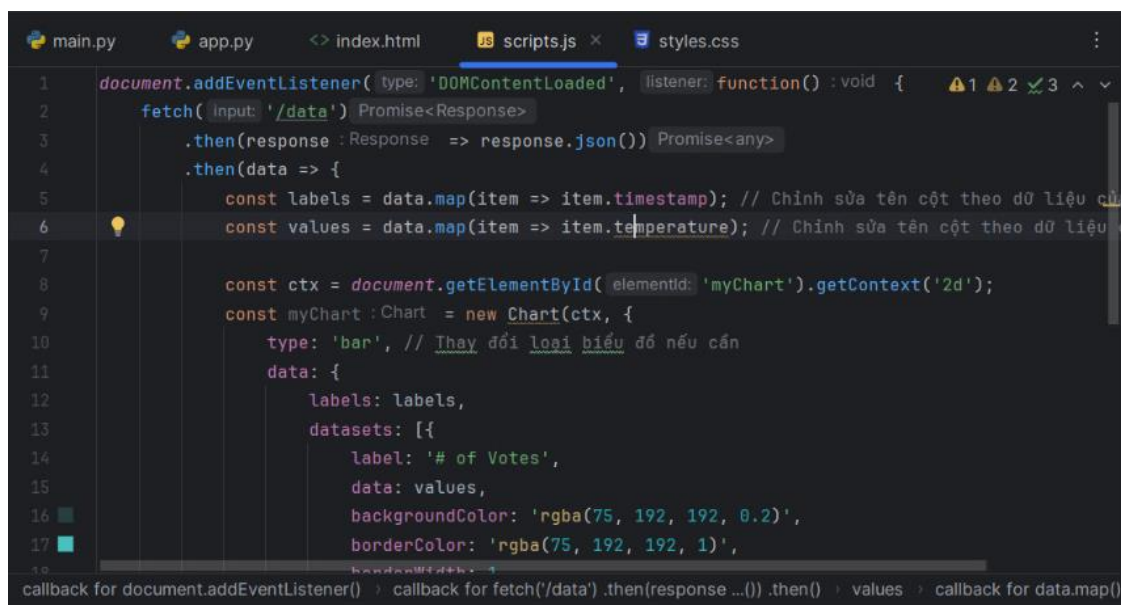


The screenshot shows the Visual Studio IDE with a C# file named `api.aspx.cs` open. The code is for an ASP.NET page that connects to a SQL database and retrieves data. The code is as follows:

```
using System.Web.Script.Serialization;
using System.Web.UI;

namespace Forex
{
    2 references
    public partial class api : Page
    {
        0 references
        protected void Page_Load(object sender, EventArgs e)
        {
            11 string connectionString = "Data Source=127.0.0.1,1443;Initial Catalog=ForexRates;User Id=sa;Password=
            12
            13 var forexDataList = new List<ForexData>();
            14
            15 using (SqlConnection connection = new SqlConnection(connectionString))
            16 {
            17     string query = "SELECT target_currency, updated_time FROM dbo.ForexRates ORDER BY updated_time
            18     SqlCommand command = new SqlCommand(query, connection);
            19
            20     connection.Open();
            21     SqlDataReader reader = command.ExecuteReader();
            22
            23     while (reader.Read())
            24     {
            25         forexDataList.Add(new ForexData
            26         {
            27             TargetCurrency = reader["target_currency"].ToString(),
            28             UpdatedTime = (DateTime)reader["updated_time"]
            29         });
            30     }
            31
            32 }
            33
            34 var latestUpdate = new ForexData();
            35 if (forexDataList.Count > 0)
            36 {
            37     latestUpdate = forexDataList[0]; // Lấy lần cập nhật mới nhất từ danh sách
            38 }
        }
    }
}
```

Tiếp theo viết mã code js và html



The screenshot shows a code editor with a JavaScript file named `scripts.js` open. The code is for a web page that fetches data from a server and updates a chart. The code is as follows:

```
1 document.addEventListener( type: 'DOMContentLoaded', listener: function() :void {
2     fetch( input: '/data' ) Promise<Response>
3     .then( response : Response => response.json() ) Promise<any>
4     .then( data => {
5         const labels = data.map( item => item.timestamp ); // Chính sửa tên cột theo dữ liệu cũ
6         const values = data.map( item => item.temperature ); // Chính sửa tên cột theo dữ liệu cũ
7
8         const ctx = document.getElementById( elementId: 'myChart' ).getContext( '2d' );
9         const myChart : Chart = new Chart( ctx, {
10             type: 'bar', // Thay đổi loại biểu đồ nếu cần
11             data: {
12                 labels: labels,
13                 datasets: [{
14                     label: '# of Votes',
15                     data: values,
16                     backgroundColor: 'rgba(75, 192, 192, 0.2)',
17                     borderColor: 'rgba(75, 192, 192, 1)',
18                 }],
19             },
20         } );
21     } );
22 }
```

```
main.py app.py json_node-red.json index.html x scripts.js style
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Chart Example</title>
7   <link rel="stylesheet" href="styles.css">
8   <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
9 </head>
10 <body>
11   <h1>Biểu đồ từ dữ liệu SQL Server</h1>
12   <canvas id="myChart" width="400" height="200"></canvas>
13   <script src="scripts.js"></script>
14 </body>
15 </html>
16
```

Kết quả đạt được

