TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ
BỘ MÔN: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN
LẬP TRÌNH PYTHON

Giáo viên hướng dẫn: **Đỗ Duy Cốp**
Sinh viên thực hiện: **Phương Minh Duy**
Ngành học: **Kỹ thuật máy tính**
Lớp: **K56KMT.01**

Thái Nguyên 2024

TRƯỜNG ĐHKTCN
KHOA ĐIỆN TỬ

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ BÀI TẬP LỚN MÔN

Sinh viên: Phương Minh Duy MSSV: K205480106011

Lớp: K56KMT Khóa: 56

Giáo viên hướng dẫn: Đỗ Duy Cốp

1. Tên đề tài:

WEBSITE THEO DÕI GIÁ ĐỒNG GBP (BẢNG ANH)

2. Nội dung thực hiện

- Cơ sở dữ liệu: Lưu thông tin gồm thời gian và số liệu
- Sử dụng FastApi và python để tạo ra 1 API từ các trang web về giá đồng GBP
- Xây dựng một chu trình trong Node-RED để tự động gọi API Python để lấy dữ liệu. Sau đó, xử lý dữ liệu và ghi dữ liệu vào database
- Xây dựng trang Web để hiện thị thông tin : Xây dựng một ứng dụng web để hiển thị dữ liệu từ cơ sở dữ liệu. Hiển thị giá đồng yên theo ngày hoặc giờ

3. Các sản phẩm, kết quả :
    - Sản phẩm phần mềm theo yêu cầu của bài tập lớn.
    - Thuyết minh đồ án theo mẫu chung của khoa Điện tử.
4. Ngày giao nhiệm vụ: 15/05/2024
5. Ngày hoàn thành nhiệm vụ: 26/05/2024

GIÁO VIÊN HƯỚNG DẪN

Đỗ Duy Cốp

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

……
……
……
……
……
……
……
……
……
……
……

Thái Nguyên , ngày…tháng…năm…

Giáo viên hướng dẫn

(Ký và ghi rõ họ tên)

MỤC LỤC

LỜI NÓI ĐẦU

Dưới bối cảnh toàn cầu hóa và sự phát triển nhanh chóng của thị trường tài chính, tỷ giá hối đoái đóng một vai trò quan trọng trong các hoạt động kinh tế và thương mại quốc tế. Hiểu và theo dõi sự biến động của tỷ giá hối đoái là cần thiết không chỉ đối với các tổ chức tài chính mà còn đối với các doanh nghiệp và cá nhân tham gia vào thị trường quốc tế.

Bài toán này tập trung vào việc xây dựng một hệ thống tự động cập nhật và hiển thị tỷ giá hối đoái giữa đồng GBP (bảng Anh) và VND theo thời gian thực. Hệ thống này không chỉ cung cấp thông tin kịp thời và chính xác về tỷ giá hối đoái mà còn giúp người dùng dễ dàng theo dõi sự biến động của tỷ giá qua các biểu đồ trực quan.

Em xin cảm ơn Thầy Đỗ Duy Cốp đã hướng dẫn em hoàn thành đề tài này.

CHƯƠNG I. CƠ SỞ LÝ THUYẾT

1.1. Giới thiệu về Python

1.1.1. Python là gì?

Python là một ngôn ngữ lập trình được tạo ra bởi Guido van Rossum và phát hành lần đầu vào năm 1991. Ban đầu, Python được phát triển như một dự án sở thích nhằm tạo ra một ngôn ngữ lập trình dễ đọc và dễ viết hơn so với các ngôn ngữ hiện có. Tên "Python" được lấy cảm hứng từ nhóm hài kịch Monty Python, thể hiện sự hài hước và dễ tiếp cận của ngôn ngữ này.

Trong suốt thập kỷ đầu tiên, Python đã thu hút được sự quan tâm từ cộng đồng lập trình viên nhờ vào triết lý thiết kế nhấn mạnh vào sự rõ ràng và đơn giản. Phiên bản 2.0 ra mắt năm 2000, mang đến nhiều cải tiến quan trọng như khả năng xử lý Unicode và bộ sưu tập các công cụ chuẩn (standard library) phong phú hơn. Phiên bản 3.0, ra mắt vào năm 2008, là một bước tiến lớn với nhiều thay đổi không tương thích ngược, nhằm loại bỏ các khuyết điểm của ngôn ngữ và cải thiện hiệu năng.

1.1.2. Đặc Điểm Chính của Python

Dễ Học và Sử Dụng: Python được thiết kế với cú pháp rõ ràng và dễ đọc, làm cho việc học lập trình trở nên dễ dàng hơn cho người mới bắt đầu. Các khái niệm cơ bản như biến, hàm, và vòng lặp được thể hiện một cách trực quan.

Đa Nền Tảng: Python có thể chạy trên nhiều hệ điều hành khác nhau như Windows, macOS, Linux, và các hệ điều hành khác, giúp lập trình viên dễ dàng phát triển và triển khai ứng dụng trên nhiều môi trường.

Thư Viện Phong Phú: Python có một hệ thống thư viện phong phú và đa dạng, hỗ trợ nhiều lĩnh vực như web development (Django, Flask), data analysis (Pandas, NumPy), machine learning (Scikit-learn, TensorFlow), và nhiều lĩnh vực khác.

Mã Nguồn Mở: Python là ngôn ngữ mã nguồn mở, cho phép người dùng tự do sử dụng, thay đổi và phân phối. Điều này đã thúc đẩy một cộng đồng lớn mạnh và sự phát triển liên tục của ngôn ngữ này.

1.1.3. Ứng dụng của Python

Web Development: Python được sử dụng rộng rãi trong phát triển web với các framework nổi tiếng như Django và Flask. Django cung cấp một bộ công cụ mạnh mẽ và hoàn chỉnh để xây dựng các ứng dụng web phức tạp, trong khi Flask là một framework nhẹ, linh hoạt cho các dự án nhỏ hơn.

Data Science và Machine Learning: Python là ngôn ngữ chính trong lĩnh vực khoa học dữ liệu và học máy. Các thư viện như Pandas, NumPy, và Matplotlib cung

cấp các công cụ mạnh mẽ để phân tích và trực quan hóa dữ liệu, trong khi Scikit- learn và TensorFlow hỗ trợ các thuật toán học máy phức tạp.

Tự Động Hóa: Python được sử dụng để viết các script tự động hóa các tác vụ lặp đi lặp lại, từ việc quản lý hệ thống đến xử lý dữ liệu.

Phát Triển Ứng Dụng: Python cũng được sử dụng trong phát triển ứng dụng desktop (với PyQt, Tkinter) và trò chơi (với Pygame).

1.1.4. Các công cụ và môi trường phát triển Python

IDLE: Một môi trường phát triển tích hợp đơn giản đi kèm với Python, phù hợp cho người mới bắt đầu.

PyCharm: Một IDE mạnh mẽ dành cho Python, hỗ trợ nhiều tính năng như gỡ lỗi, quản lý phiên bản, và tự động hoàn thành mã.

Jupyter Notebook: Một công cụ mạnh mẽ cho phép viết và chạy mã Python trong một môi trường tương tác, rất phổ biến trong lĩnh vực khoa học dữ liệu. Nó cho phép người dùng tạo và chia sẻ tài liệu chứa mã nguồn, hình ảnh, và các chú thích.

1.2. Giới thiệu về SQL Server

SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển. Nó là một trong những hệ quản trị cơ sở dữ liệu phổ biến nhất trên thế giới và được sử dụng rộng rãi trong các doanh nghiệp và tổ chức với các ứng dụng dữ liệu quy mô lớn.

Dưới đây là một số đặc điểm chính của SQL Server:

- Quản lý dữ liệu quan hệ: SQL Server sử dụng ngôn ngữ truy vấn cấu trúc (SQL) để lưu trữ và truy xuất dữ liệu theo mô hình quan hệ. Nó hỗ trợ nhiều loại dữ liệu như số, văn bản, hình ảnh, và ngày tháng.
- Hỗ trợ cho các ứng dụng doanh nghiệp: SQL Server cung cấp các tính năng mạnh mẽ như giao dịch, bảo mật dữ liệu, sao lưu và phục hồi dữ liệu, và quản lý hiệu suất.
- Phân tích dữ liệu: SQL Server cung cấp các tính năng phân tích dữ liệu như dịch vụ tích hợp (Integration Services), bộ phân tích (Analysis Services), và báo cáo (Reporting Services) để hỗ trợ việc phân tích và báo cáo dữ liệu.

1.3. Giới thiệu về Node-red

Node-RED là một công cụ mã nguồn mở được phát triển bởi IBM để dễ dàng tạo và quản lý các luồng dữ liệu và quy trình làm việc trực quan. Nó cho phép bạn kết nối và tự động hóa các thiết bị, dịch vụ và ứng dụng từ các nguồn khác nhau một cách linh hoạt và dễ dàng.

Dưới đây là một số tính năng chính của Node-RED:

- Các nút đa dạng: Node-RED cung cấp một bộ sưu tập đa dạng các nút (nodes) có

thể kéo và thả để tạo các luồng làm việc, từ gửi và nhận dữ liệu đến xử lý và điều khiển.

- Giao diện trực quan và dễ sử dụng: Node-RED cung cấp một giao diện trực quan dễ sử dụng cho việc kéo và thả các nút, kết nối chúng và cấu hình chúng để tạo ra các luồng làm việc.
- Tích hợp dễ dàng: Node-RED có thể tích hợp với nhiều thiết bị, dịch vụ và ứng dụng thông qua các nút tích hợp có sẵn hoặc thông qua việc viết các nút tùy chỉnh.
- Mở rộng và tuỳ biến: Node-RED cho phép bạn mở rộng và tuỳ biến các chức năng bằng cách viết mã JavaScript hoặc sử dụng các thư viện bên ngoài.
- Tích hợp dễ dàng với SQL Server: Bằng cách sử dụng các nút tích hợp có sẵn hoặc viết mã tùy chỉnh, bạn có thể kết nối và tương tác với cơ sở dữ liệu SQL Server từ Node-RED một cách dễ dàng.

CHƯƠNG II. PHÂN TÍCH VÀ THIẾT KẾ

2.1. Mục tiêu của dự án

Mục tiêu

1. **Thu thập dữ liệu tỷ giá**: Sử dụng API từ dịch vụ FastForex.
2. **Lưu trữ dữ liệu**: Lưu trữ thông tin vào cơ sở dữ liệu SQL Server.
3. **Hiển thị dữ liệu**: Phát triển giao diện hiển thị dữ liệu tỷ giá bằng biểu đồ.

Công nghệ sử dụng

- **FastAPI**: Xây dựng API.
- **SQL Server**: Lưu trữ dữ liệu.
- **Chart.js**: Tạo biểu đồ.
- **HTML/CSS và JavaScript**: Xây dựng giao diện người dùng.

Nội dung thực hiện

1. **Thiết lập cơ sở dữ liệu**: Tạo bảng JPY_VND.
2. **Phát triển API**: Sử dụng FastAPI để lấy và cập nhật dữ liệu tỷ giá.
3. **Tự động hóa cập nhật dữ liệu**: Sử dụng Node-RED để tự động cập nhật mỗi phút.
4. **Hiển thị dữ liệu**: Xây dựng giao diện người dùng với biểu đồ tỷ giá GBP /VND.

2.2. Các thành phần của dự án

Cơ Sở Dữ Liệu:

- Bảng: forex_rates : Lưu thông tin về giá GBP, các thông tin base (GBP), to_currency (VND), rate (25441.20195), và updated_at (thời gian cập nhật) vào bảng JPY_VND.

Module đọc dữ liệu: Sử dụng Python và FastAPI để tạo một API để lấy dữ liệu từ trang web chứa tỷ giá GBP như FastForex.....

Mô tả nguồn dữ liệu:

- Sử dụng Web Scraping hoặc lấy dữ liệu qua API của các trang web chuyên về Tỷ giá giá đồng GBP (bảng Anh).
- Dữ liệu bao gồm thông tin về Tý giá GBP và thời gian cập nhật. Node-RED:

- Xây dựng một chu trình trong Node-RED để tự động gọi Stored Procedure GBP_VND để lưu giá vàng vào cơ sở dữ liệu.
- Node-RED sẽ gọi API Python (hoặc một dịch vụ khác) để lấy dữ liệu và thời gian cập nhật. Sau đó, dữ liệu này sẽ được truyền sang vào bảng JPY_VND để lưu trữ. Web:
- Xây dựng một ứng dụng web để hiển thị dữ liệu từ cơ sở dữ liệu.
- Sử dụng các công nghệ như HTML, CSS, JavaScript để tạo giao diện web.

2.3. Các bước thực hiện.

Bước 1: Cài đặt môi trường

- Cài đặt Python và các thư viện cần thiết: FastAPI, requests, pyodbc, uvicorn.
- Cài đặt SQL Server và tạo cơ sở dữ liệu.
- Cài đặt Node-red

Bước 2: Tạo cơ sở dữ liệu và bảng

- Tạo bảng `JPY_VND` trong SQL Server để lưu trữ tỷ giá:

```
sql
Sao chép mã
SELECT TOP (1000) [ID]
      ,[base]
      ,[to_currency]
      ,[rate]
      ,[updated_at]
  FROM [Gia_JPY].[dbo].[JPY_VND]);
```

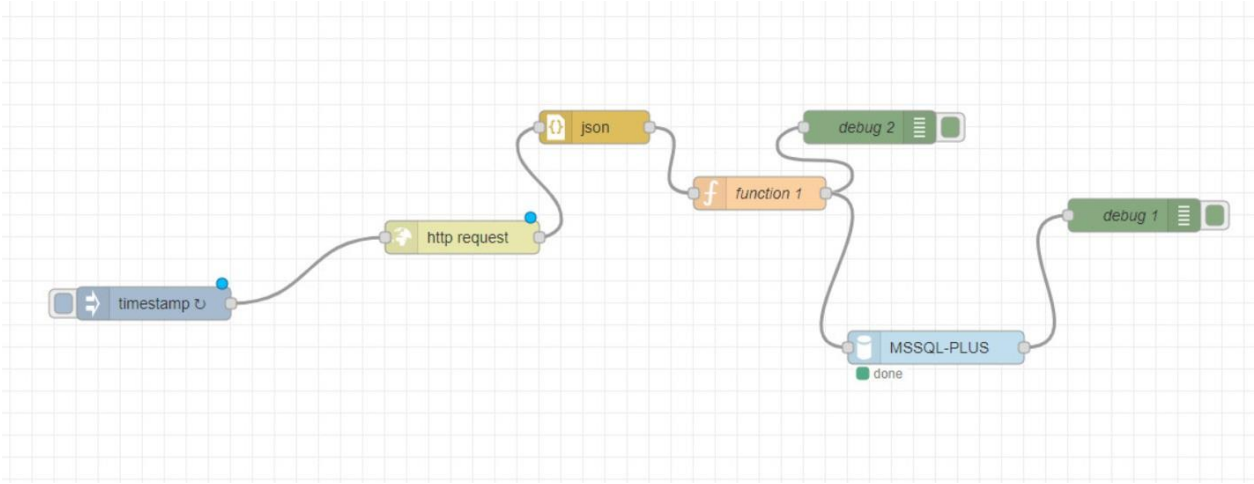
Bước 3: Xây dựng server FastAPI

```
import asyncio
from fastapi import FastAPI, HTTPException
 import requests
from datetime import datetime, timedelta, timezone

app = FastAPI()
```

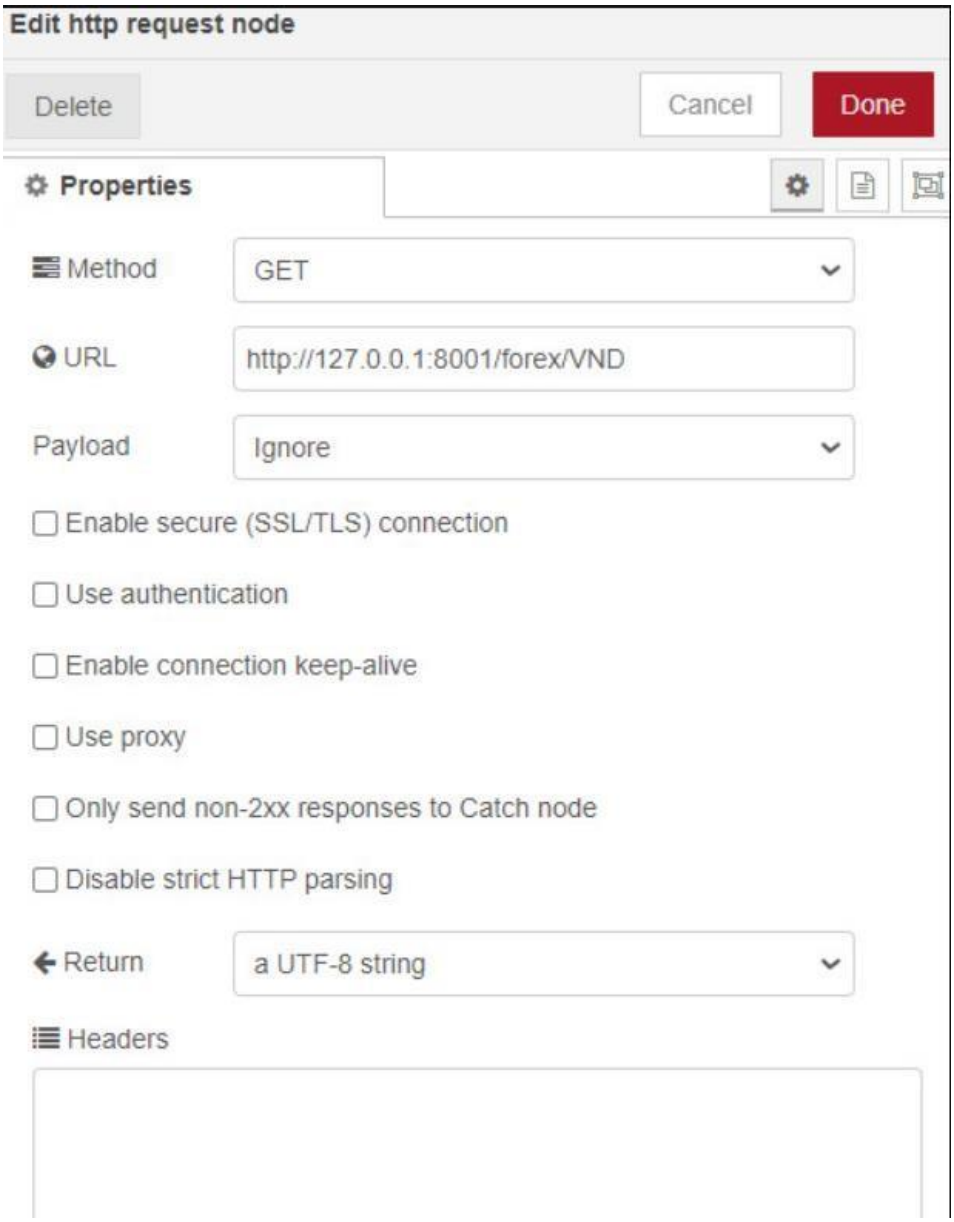
```
API_KEY = '1af198eb76-8362cbab0b-sdx021'  # Thay bằng key API của bạn
BASE_URL = 'https://api.fastforex.io/'  # URL cơ bản của API fastFOREX

# Định nghĩa múi giờ UTC và múi giờ Việt Nam
UTC_timezone = timezone.utc
VN_timezone = timezone(timedelta(hours=7))  # UTC+7

async def fetch_forex_data():
    while True:
        from_currency = 'GBP'  # Tiền tệ mặc định là JPY

        url = f"{BASE_URL}fetch-one?from={from_currency}&to=GBP&api_key={API_KEY}"  # Chỉnh sửa đây
        response = requests.get(url)

        if response.status_code == 200:
            forex_data = response.json()
            # Chuyển đổi thời gian từ múi giờ UTC sang múi giờ Việt Nam
            updated_time_utc = datetime.strptime(forex_data['updated'], "%Y-%m-%d %H:%M:%S")
            updated_time_vn = updated_time_utc.replace(tzinfo=UTC_timezone).astimezone(VN_timezone)
            forex_data['updated'] = updated_time_vn.strftime("%Y-%m-%d %H:%M:%S %Z%z")
            print(forex_data)  # In ra dữ liệu JSON cập nhật
        elif response.status_code == 401:
            raise HTTPException(status_code=401, detail="Unauthorized. Check your API key.")
        elif response.status_code == 404:
            raise HTTPException(status_code=404, detail="Currency pair not found.")
        else:
            raise HTTPException(status_code=response.status_code, detail=response.text)

        await asyncio.sleep(30)  # Chờ 30 giây trước khi gửi yêu cầu kế tiếp

@app.on_event("startup")
async def startup_event():
    asyncio.create_task(fetch_forex_data())  # Bắt đầu lặp vô hạn để cập nhật dữ liệu

@app.get("/forex/{to_currency}")
def get_forex_rate(to_currency: str):
    from_currency = 'GBP'  # Tiền tệ mặc định là JPY

    # Kiểm tra định dạng mã tiền tệ
    if len(to_currency) != 3 or not to_currency.isalpha():
        raise HTTPException(status_code=400, detail="Invalid currency code format. Use a 3-letter currency code.")

    url = f"{BASE_URL}fetch-one?from={from_currency}&to={to_currency}&api_key={API_KEY}"
    response = requests.get(url)

    if response.status_code == 200:
        forex_data = response.json()
        # Chuyển đổi thời gian từ múi giờ UTC sang múi giờ Việt Nam
        updated_time_utc = datetime.strptime(forex_data['updated'], "%Y-%m-%d %H:%M:%S")
        updated_time_vn = updated_time_utc.replace(tzinfo=UTC_timezone).astimezone(VN_timezone)
        forex_data['updated'] = updated_time_vn.strftime("%Y-%m-%d %H:%M:%S %Z%z")
        return forex_data # Trả về dữ liệu JSON trực tiếp elif
    response.status_code == 401:
        raise HTTPException(status_code=401, detail="Unauthorized. Check your API key.")
```

```
    elif response.status_code == 404:
        raise HTTPException(status_code=404, detail="Currency pair not found.")
    else:
        raise HTTPException(status_code=response.status_code, detail=response.text)

if _name_== "_main_": import
    uvicorn

    uvicorn.run(app, host="127.0.0.1", port=8001)
```

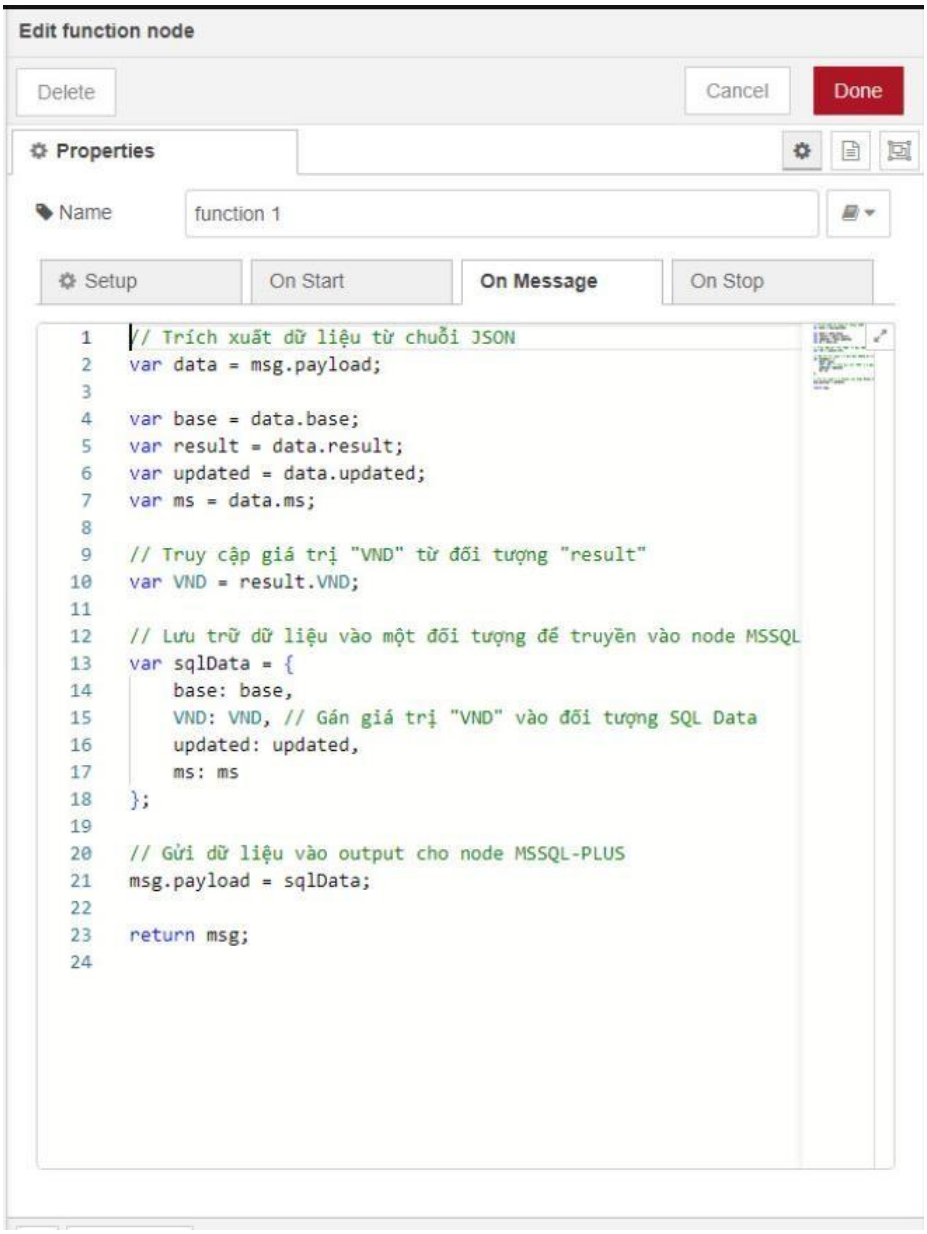
Bước 4: Cấu hình node-red để đọc dữ liệu từ request và gửi vào database



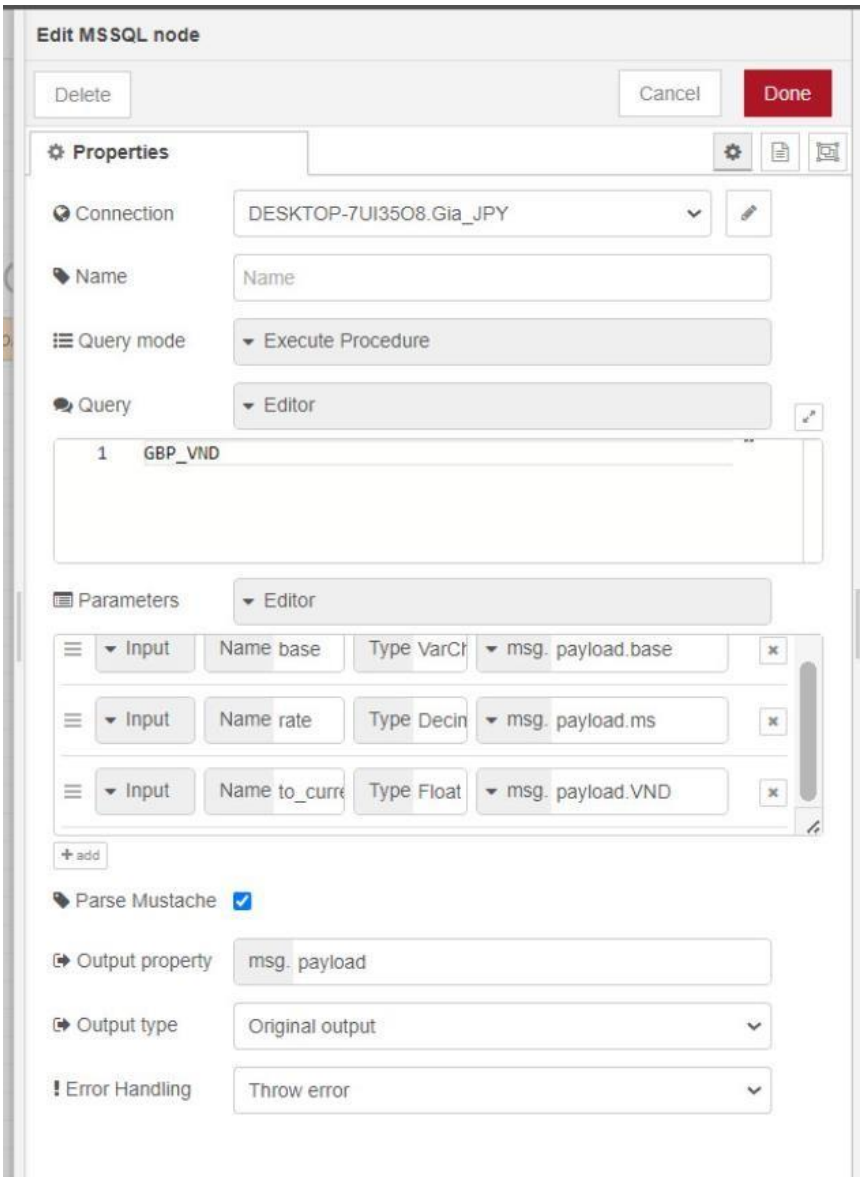
- Thực hiện cấu hình nốt request để lấy dữ liệu:



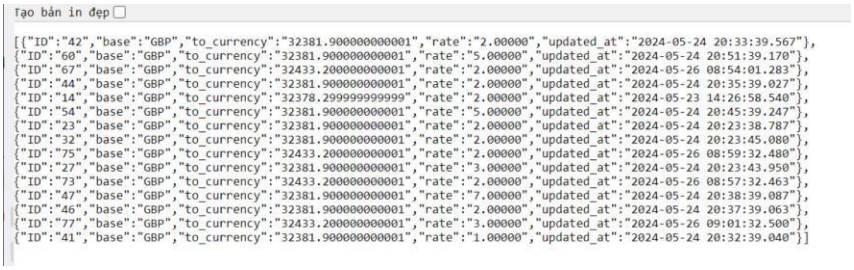
- Viết hàm sử lý dữ liệu ở nốt funtion để gửi vào SQL:



- Thực hiện các cấu hình ở nốt SQL để nhận dữ liệu từ request:



- Dữ liệu được gọi thành công:



Bước 5: Xây dựng giao diện bằng php và gọi dữ liệu từ Database để hiển thị ra giao diện.

Trong thư mục php gồm có:

- File api.php để thực hiện kết nối với SQL

```
<?php
header('Content-Type: application/json');

// Kết nối server
$server = "127.0.0.1,1433";
$database = "Gia_JPY";
$username = "sa";
$password = "123";

try {
    $conn = new PDO("sqlsrv:Server=$server;Database=$database", $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE,  PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo json_encode(array("error" => "Connection DB failed: " . $e->getMessage())); exit();
}

// Method GET
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    // Kiểm tra tham số đầu vào
    if(isset($_GET["action"])) {
        // Lấy tham số action
        $action = $_GET["action"];
        if($action == "get_all") {
            // Truy vấn lấy 15 dòng ngẫu nhiên từ bảng JPY_VND
            $stmt = $conn->query("SELECT TOP 15 * FROM JPY_VND ORDER BY NEWID()");

            // Xử lý kết quả
            $array_kq = [];
            while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
                $array_kq[] = $row; // Thêm dòng hiện tại vào mảng
            }
            $json_result = json_encode($array_kq);
            echo $json_result;
        } elseif($action == "get_latest") {
            // Truy vấn lấy tỷ giá mới nhất từ bảng JPY_VND
            $stmt = $conn->query("SELECT TOP 1 * FROM JPY_VND ORDER BY rate DESC");
```

```
                // Xử lý kết quả
                $array_kq = [];
                while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
                    $array_kq[] = $row; // Thêm dòng hiện tại vào mảng
                }
                $json_result = json_encode($array_kq);
                echo $json_result;
            } else {
                echo json_encode(array("error" => "Invalid action parameter"));
            }
        } else {
            echo json_encode(array("error" => "Missing action parameter"));
        }
    } else {
        echo json_encode(array("error" => "Invalid request method"));
    }
?>
```

- File index.js, index.html và index.css để thực hiện tạo giao diện hiển thị index.js

```
document.addEventListener("DOMContentLoaded", function () {
  console.log("Đã tải xong index.js");
});

function fetchForexRates(action, callback) {
  // dòng này gọi api để lấy dữ liệu dữ liệu trả về = {"error":"Connection DB failed: could not find
driver"} thì lỗi còn j
  //hướng giải quyết??
  fetch(`http://localhost/bt_python/api.php?action=${action}`)
    .then((response) => {
      if (!response.ok) {
        throw new Error("Network response was not ok");
      }
      return response.json();
    })
    .then((json_data) => {
      callback(json_data);
    })
    .catch((error) => {
      console.error("There was a problem with the fetch operation:", error);
    });
}

function updateTable(data) {
  let tableHTML = `
    <table class="table tablSe-bordered table-success">
      <thead>
        <tr>
          <th>STT</th>
          <th>Base</th>
          <th>To Currency</th>
          <th>Rate</th>
          <th>Updated At</th>
```

```
        </tr>
      </thead>
      <tbody>`;

  data.forEach((item, index) => {
   tableHTML += `
     <tr>
      <td>${index + 1}</td>
      <td>${item.base}</td>
      <td>${item.to_currency}</td>
      <td>${item.rate}</td>
      <td>${item.updated_at}</td>
     </tr>`;
  });

  tableHTML += `</tbody></table>`;
  document.getElementById("table_now").innerHTML = tableHTML;
 }

 function updateChart(data) {
  const rates = data.map(d => ({ x: new Date(d.updated_at), y: d.rate }));

  const ctx = document.getElementById('myChart').getContext('2d');
  if (window.myChart) {
   window.myChart.data.datasets[0].data = rates;
   window.myChart.update();
  } else {
   window.myChart = new Chart(ctx, {
    type: 'line',
    data: {
     datasets: [{
     data: rates,
      borderColor: 'rgba(75, 192, 192, 1)',
      backgroundColor: 'rgba(75, 192, 192, 0.2)',
      borderWidth: 1,
      fill: false,
     }]
    },
    options: {
     scales: {
     x: {
        type: 'time',
        time: {
         unit: 'minute'
        },
        title: {
         display: true,
         text: 'Thời gian'
        }
       },
       y: {
        title: {
         display: true,
         text: 'Tỷ giá'
        }
       }
      },
```

```
                plugins:          {
                  tooltip:        {
                  mode: 'index',
                    intersect: false,
                  },
                  hover: {
                    mode:    'nearest',
                    intersect: true
                  }
                }
              }
            });
          }
        }

        function              refreshData()             {
          fetchForexRates("get_all",   (data)   =>   {
          updateTable(data);
            updateChart(data);
          });
        }

        refreshData();
etInterval(refreshData, 1000 * 30); // Cập nhật mỗi 1 phút
```

index.html

```
        <!DOCTYPE html>
        <html lang="en">

        <head>
            <meta charset="UTF-8">
            <meta name="viewport" content="width=device-width, initial-scale=1.0">
            <title>btl_py</title>
            <link rel="stylesheet" href="css/index.css">
            <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
            <link                                                    rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css">
            <script src="https://cdn.jsdelivr.net/npm/jquery@3.7.1/dist/jquery.slim.min.js"></script>
            <script
        src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
            <script
        src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.bundle.min.js"></script>
            <script
        src="https://cdnjs.cloudflare.com/ajax/libs/datepicker/1.0.10/datepicker.min.js"></script>
            <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
        </head>

        <body>
            <header>
                <div class="container">
                    <h1>Bài tập</h1>
                </div>
            </header>
            <main class="main_container">
                <div id="table_now">
                    <!-- Bảng sẽ được tạo động tại đây -->
```

```
            </div>
            <canvas id="myChart" width="800" height="400"></canvas>
        </main>

        <script src="js/index.js"></script>
    </body>

</html>
```

index.css

```
/* Đặt lại mặc định của một số thẻ HTML */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: Arial, sans-serif;
}

/* Styling cho header */
header {
    background-color:  rgb(232,  132,  39);
    color: white;
    padding: 20px 0;
}

.container  {
    display: flex;
    justify-content: space-between;
    align-items: center;
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 20px;
}

.logo  img  {
    height: 50px;
}

nav ul {
    list-style: none;
    display: flex;
}

nav ul li {
    margin-left: 20px;
}

nav ul li a {
    text-decoration: none;
    color: white;
    font-weight:  bold;
    transition: color 0.3s;
}
```

```
        nav ul li a:hover {
            color: #f0f0f0;
        }

        /* MAIN */
        .main_container {

            margin: 10px;
        }

        th, td {
            padding: 10px;
            text-align: center; /* Căn giữa theo chiều ngang */
            vertical-align: middle; /* Căn giữa theo chiều dọc */
        }

        thead th {
            text-align: center; /* Căn giữa nội dung trong thead */
            vertical-align: middle; /* Căn giữa theo chiều dọc */
        }

        thead th:first-child, tbody td:first-child {
            max-width: 30px; /* Giới hạn độ rộng tối đa của cột đầu tiên */ overflow:
            hidden; /* Ẩn nội dung tràn */
            white-space: nowrap; /* Không xuống dòng */
            text-overflow: ellipsis; /* Thêm dấu ba chấm nếu nội dung quá dài */
        }

        thead th:nth-child(2), tbody td:nth-child(2) {
            max-width: 50px; /* Giới hạn độ rộng tối đa của cột thứ hai */ overflow:
            hidden; /* Ẩn nội dung tràn */
            white-space: nowrap; /* Không xuống dòng */
            text-overflow: ellipsis; /* Thêm dấu ba chấm nếu nội dung quá dài */
        }

        thead th:nth-child(4), tbody td:nth-child(4) {
            max-width: 50px; /* Giới hạn độ rộng tối đa của cột thứ hai */ overflow:
            hidden; /* Ẩn nội dung tràn */
            white-space: nowrap; /* Không xuống dòng */
            text-overflow: ellipsis; /* Thêm dấu ba chấm nếu nội dung quá dài */
}
```

Kết quả cuối cùng của bài toán:

Bài tập

STT	Base	To Currency	Rate	Updated At
1	GBP	32433.200000000001	5.00000	2024-05-26 11:27:06.677
2	GBP	32433.200000000001	5.00000	2024-05-26 12:20:38.027
3	GBP	32433.200000000001	4.00000	2024-05-26 15:47:10.890
4	GBP	32433.200000000001	5.00000	2024-05-26 15:25:10.683
5	GBP	32388.900000000001	5.00000	2024-05-24 20:54:39.313
6	GBP	32433.200000000001	5.00000	2024-05-26 09:29:14.510
7	GBP	32433.200000000001	4.00000	2024-05-26 15:59:10.980
8	GBP	32433.200000000001	5.00000	2024-05-26 11:36:06.990
9	GBP	32381.900000000001	1.00000	2024-05-24 20:26:38.970
10	GBP	32381.900000000001	2.00000	2024-05-24 20:23:38.970

CHƯƠNG III. KẾT LUẬN

Trong dự án này, em đã xây dựng một trang web theo dõi tỷ giá hối đoái giữa đồng GBP (bảng Anh) và VND bằng cách sử dụng FastAPI cho backend và Chart.js cho frontend. Em đã thành công trong việc lấy dữ liệu từ API của dịch vụ fastFOREX, xử lý và lưu trữ dữ liệu vào cơ sở dữ liệu SQL Server, và hiển thị dữ liệu trực quan trên giao diện người dùng.

Trang web không chỉ cung cấp khả năng theo dõi tỷ giá một cách liên tục và chính xác mà còn minh họa cách tích hợp và triển khai các công nghệ web hiện đại. Kết quả đạt được cho thấy tỷ giá hối đoái được cập nhật và hiển thị một cách hiệu quả, đáp ứng nhu cầu theo dõi và phân tích của người dùng.

Dự án này là một ví dụ điển hình cho việc sử dụng các công nghệ tiên tiến để giải quyết các vấn đề thực tế trong việc theo dõi và phân tích dữ liệu tài chính. Trong tương lai, chúng tôi có thể mở rộng thêm các tính năng như cảnh báo thay đổi tỷ giá, hỗ trợ nhiều cặp tiền tệ khác, và tối ưu hóa hiệu suất hệ thống để phục vụ người dùng tốt hơn.

Cuối cùng, em xin chân thành cảm ơn sự hỗ trợ và đồng hành của các đồng nghiệp, bạn bè, và những người đã giúp đỡ chúng tôi trong suốt quá trình thực hiện dự án. Em hy vọng rằng, với sự nỗ lực này, trang web theo dõi tỷ giá hối đoái sẽ trở thành một công cụ hữu ích, đóng góp vào sự phát triển chung của xã hội trong lĩnh vực giao tiếp đa ngôn ngữ.