# -\*- coding: utf-8 -\*-

# Script để nhập lịch âm từ file ICS cục bộ vào Google Calendar

# Có kiểm tra trùng lặp (dựa trên ngày), thêm tên ngày lễ vào tiêu đề và sử dụng batching.

from datetime import date, datetime, timedelta

import os

import time

import re

import json

import sys

from google.oauth2.credentials import Credentials

from google\_auth\_oauthlib.flow import InstalledAppFlow

from googleapiclient.discovery import build

from google.auth.transport.requests import Request

from googleapiclient.http import BatchHttpRequest

from googleapiclient.errors import HttpError

from icalendar import Calendar

from pytz import timezone

import pytz

# --- CẤU HÌNH ---

SCOPES = ['https://www.googleapis.com/auth/calendar'] # Scope để truy cập Google Calendar

LOCAL\_ICS\_INPUT\_FILE = 'amlichvn.ics' # <-- ĐẶT TÊN FILE ICS CỦA FEN Ở ĐÂY

ICS\_FILE = 'amlichvn\_local.ics' # Tên file tạm sau khi sửa định dạng

CALENDAR\_NAME = 'Lịch ÂM VN'

VIETNAM\_TZ = timezone('Asia/Ho\_Chi\_Minh')

# Cấu hình Batching và Rate Limit

BATCH\_SIZE = 100 # Số sự kiện trong mỗi gói batch

MIN\_DELAY\_BETWEEN\_BATCHES = BATCH\_SIZE / 9.0

# Phạm vi năm để kiểm tra sự kiện đã tồn tại trên Google Calendar

START\_YEAR\_ICS = 2000

END\_YEAR\_ICS = 2050

# Danh sách các ngày lễ đặc biệt (Âm lịch) và tên hiển thị

special\_days = {

"1/1": "Tết Nguyên đán",

"15/1": "Tết Nguyên Tiêu",

"3/3": "Tết Hàn thực",

"10/3": "Giỗ tổ Hùng Vương",

"15/4": "Lễ Phật Đản",

"5/5": "Tết Đoan ngọ",

"7/7": "Lễ Thất tịch",

"15/7": "Lễ Vu Lan",

"15/8": "Tết Trung thu",

"9/9": "Tết Trùng cửu",

"10/10": "Tết Trùng thập",

"15/10": "Tết Hạ Nguyên",

"23/12": "Ông Táo về trời"

}

# --- BIẾN ĐẾM (GLOBAL) ---

successful\_imports\_count = 0

conflict\_409\_count = 0

forbidden\_403\_count = 0

other\_api\_errors\_count = 0

batch\_execution\_error\_count = 0

# --- HÀM XÁC THỰC VÀ LẤY GOOGLE SERVICE ---

def get\_service():

"""Xác thực với Google API và trả về đối tượng service cho Google Calendar."""

creds = None

if os.path.exists('token.json'):

try:

creds = Credentials.from\_authorized\_user\_file('token.json', SCOPES)

except Exception as e:

print(f"Cảnh báo: Lỗi đọc token.json - {e}. Sẽ tiến hành xác thực mới.", file=sys.stderr)

creds = None

if creds and not creds.valid:

if creds.expired and creds.refresh\_token:

print("Token hết hạn, đang làm mới token xác thực...")

try:

creds.refresh(Request())

print("Đã làm mới token thành công.")

except Exception as e:

print(f"Lỗi làm mới token: {e}. Cần xác thực lại từ đầu.", file=sys.stderr)

if "invalid\_scope" in str(e):

print("Lỗi: Phạm vi xác thực (scope) không hợp lệ. Vui lòng kiểm tra lại SCOPES.", file=sys.stderr)

creds = None

else:

print("Token không hợp lệ hoặc không thể làm mới, cần xác thực lại từ đầu.", file=sys.stderr)

creds = None

if not creds or not creds.valid:

if os.path.exists('credentials.json'):

print("Đang tiến hành xác thực mới từ credentials.json...")

try:

flow = InstalledAppFlow.from\_client\_secrets\_file('credentials.json', SCOPES)

creds = flow.run\_local\_server(port=0)

print("Xác thực thành công.")

except FileNotFoundError:

print("Lỗi: Không tìm thấy tệp credentials.json!", file=sys.stderr)

return None

except Exception as e:

print(f"Lỗi trong quá trình xác thực mới: {e}", file=sys.stderr)

if "could not locate runnable browser" in str(e):

print("Script không tìm thấy trình duyệt để hoàn tất xác thực OAuth.", file=sys.stderr)

return None

else:

print("Lỗi: Không tìm thấy tệp credentials.json!", file=sys.stderr)

return None

try:

with open('token.json', 'w') as token:

token.write(creds.to\_json())

except Exception as e:

print(f"Cảnh báo: Không thể lưu token.json - {e}.", file=sys.stderr)

try:

service = build('calendar', 'v3', credentials=creds)

return service

except Exception as e:

print(f"Lỗi khi xây dựng service hoặc kết nối API: {e}", file=sys.stderr)

return None

# --- HÀM TÌM HOẶC TẠO LỊCH GOOGLE ---

def get\_or\_create\_calendar(service):

"""Tìm lịch Google Calendar theo tên, nếu không tìm thấy thì tạo mới."""

if not service:

return None

try:

print(f"Đang tìm lịch Google Calendar có tên '{CALENDAR\_NAME}'...")

page\_token = None

while True:

calendar\_list = service.calendarList().list(pageToken=page\_token).execute()

calendars = calendar\_list.get('items', [])

for cal in calendars:

if cal.get('summary') == CALENDAR\_NAME:

print(f"Đã tìm thấy lịch: ID = {cal['id']}")

return cal['id']

page\_token = calendar\_list.get('nextPageToken')

if not page\_token:

break

print(f"Không tìm thấy lịch '{CALENDAR\_NAME}'. Đang tạo một lịch mới...")

calendar = {

'summary': CALENDAR\_NAME,

'timeZone': 'Asia/Ho\_Chi\_Minh'

}

created\_calendar = service.calendars().insert(body=calendar).execute()

print(f"Đã tạo lịch mới thành công: Tên = '{created\_calendar['summary']}', ID = {created\_calendar['id']}")

return created\_calendar['id']

except Exception as e:

print(f"Lỗi khi tìm hoặc tạo lịch Google Calendar: {e}", file=sys.stderr)

return None

# --- HÀM SỬA LỖI ĐỊNH DẠNG FILE ICS ---

def fix\_ics\_format(raw\_text):

"""Cố gắng sửa các lỗi định dạng phổ biến trong nội dung ICS."""

lines = raw\_text.strip().splitlines()

final\_fixed\_lines = []

prop\_line\_pattern = re.compile(r'^([A-Z-]+)(;[^:]\*)?:(.\*)$')

dt\_malformed\_val\_pattern = re.compile(r'^(\d{8})(\d{7})(Z?)$')

print("Đang tiến hành sửa lỗi định dạng cơ bản trong nội dung ICS...")

space\_fix\_count = 0

dt\_fix\_count = 0

temp\_lines\_pass1 = []

for line in lines:

stripped\_line = line.strip()

if not stripped\_line:

continue

match = prop\_line\_pattern.match(stripped\_line)

if match:

prop\_name = match.group(1)

prop\_params = match.group(2) or ''

colon\_index\_after\_params = stripped\_line.find(':', len(prop\_name) + len(prop\_params))

if colon\_index\_after\_params != -1 and colon\_index\_after\_params + 1 < len(stripped\_line):

if stripped\_line[colon\_index\_after\_params + 1] == ' ':

line\_after\_space\_fix = stripped\_line[:colon\_index\_after\_params + 1] + stripped\_line[colon\_index\_after\_params + 2:]

temp\_lines\_pass1.append(line\_after\_space\_fix)

space\_fix\_count += 1

else:

temp\_lines\_pass1.append(stripped\_line)

else:

temp\_lines\_pass1.append(stripped\_line)

else:

temp\_lines\_pass1.append(stripped\_line)

for line in temp\_lines\_pass1:

if line.startswith('DTSTART:') or line.startswith('DTEND:'):

parts = line.split(':', 1)

if len(parts) == 2:

prop\_part = parts[0]

value\_part = parts[1]

if 'T' not in value\_part and 'VALUE=DATE' not in prop\_part:

match\_val = dt\_malformed\_val\_pattern.match(value\_part)

if match\_val:

date\_part = match\_val.group(1)

time\_part\_malformed = match\_val.group(2)

z\_part = match\_val.group(3)

if len(time\_part\_malformed) >= 6:

fixed\_value = date\_part + 'T' + time\_part\_malformed[:6] + z\_part

final\_fixed\_lines.append(f"{prop\_part}:{fixed\_value}")

dt\_fix\_count += 1

else:

print(f"Cảnh báo: Không sửa được định dạng DTSTART/DTEND (phần giờ quá ngắn): '{line}'", file=sys.stderr)

final\_fixed\_lines.append(line)

else:

final\_fixed\_lines.append(line)

else:

final\_fixed\_lines.append(line)

else:

print(f"Cảnh báo: Dòng bắt đầu bằng DTSTART/DTEND nhưng lỗi định dạng: '{line}'", file=sys.stderr)

final\_fixed\_lines.append(line)

else:

final\_fixed\_lines.append(line)

if space\_fix\_count > 0 or dt\_fix\_count > 0:

print(f"Hoàn tất sửa định dạng. Đã sửa {space\_fix\_count} lỗi khoảng trắng và {dt\_fix\_count} lỗi định dạng DTSTART/DTEND.")

else:

print("Không tìm thấy lỗi định dạng phổ biến nào cần sửa.")

return '\n'.join(final\_fixed\_lines) + '\n'

# --- HÀM LẤY SỰ KIỆN ĐÃ TỒN TẠI TỪ GOOGLE CALENDAR ---

def fetch\_google\_events\_in\_range(service, calendar\_id, start\_year, end\_year):

"""Lấy ngày dương của tất cả sự kiện cả ngày trong phạm vi năm từ Google Calendar để kiểm tra trùng lặp."""

existing\_dates\_set = set()

print(f"\nĐang lấy ngày sự kiện đã tồn tại từ lịch Google '{calendar\_id}' từ năm {start\_year} đến {end\_year} để kiểm tra trùng lặp...")

try:

start\_date\_range = date(start\_year, 1, 1)

end\_date\_range = date(end\_year, 12, 31)

time\_min\_iso = datetime.combine(start\_date\_range, datetime.min.time()).astimezone(pytz.utc).isoformat()

time\_max\_iso = datetime.combine(end\_date\_range + timedelta(days=1), datetime.min.time()).astimezone(pytz.utc).isoformat()

except ValueError as e:

print(f"Lỗi cấu hình năm không hợp lệ: {e}. Không thể lấy sự kiện.", file=sys.stderr)

return existing\_dates\_set

page\_token = None

try:

while True:

events\_result = service.events().list(

calendarId=calendar\_id,

timeMin=time\_min\_iso,

timeMax=time\_max\_iso,

singleEvents=True, # Mở rộng sự kiện định kỳ

orderBy='startTime',

pageToken=page\_token,

fields='nextPageToken,items(start)' # Chỉ cần trường 'start'

).execute()

events = events\_result.get('items', [])

if not events:

break

for event in events:

if 'date' in event.get('start', {}): # Chỉ lấy sự kiện cả ngày

event\_date\_str = event['start']['date'] # Định dạngYYYY-MM-DD

existing\_dates\_set.add(event\_date\_str)

page\_token = events\_result.get('nextPageToken')

if not page\_token:

break

except Exception as e:

print(f"\nLỗi khi lấy ngày sự kiện đã tồn tại từ Google Calendar: {e}", file=sys.stderr)

print("Việc kiểm tra trùng lặp có thể không chính xác.", file=sys.stderr)

return set()

print(f"Hoàn tất lấy dữ liệu. Tổng số ngày dương duy nhất đã lấy sự kiện từ lịch Google: {len(existing\_dates\_set)}")

return existing\_dates\_set

# --- HÀM NHẬP SỰ KIỆN VÀO GOOGLE CALENDAR (SỬ DỤNG BATCHING VÀ KIỂM TRA TRÙNG LẶP) ---

def import\_ics\_batched(service, calendar\_id):

"""

Đọc sự kiện từ file ICS đã sửa (ICS\_FILE), biến đổi tiêu đề, tô màu đỏ các ngày lễ đặc biệt,

và nhập các sự kiện vào Google Calendar đích (calendar\_id) sử dụng batching.

Trước khi nhập, sẽ kiểm tra các ngày sự kiện đã tồn tại trong lịch Google

trong phạm vi năm cấu hình (START\_YEAR\_ICS, END\_YEAR\_ICS) để tránh tạo trùng lặp

dựa trên ngày dương.

Chỉ thêm các sự kiện từ ICS NẾU chúng chưa tồn tại trong lịch Google vào cùng ngày đó.

Báo cáo số lượng sự kiện đã import thành công và thống kê lỗi API phổ biến (409, 403).

"""

if not service:

print("Không có đối tượng service API, không thể nhập sự kiện.", file=sys.stderr)

return

if not calendar\_id:

print("Không có Calendar ID đích, không thể nhập sự kiện.", file=sys.stderr)

return

# --- BƯỚC 0: LẤY CÁC NGÀY CỦA SỰ KIỆN ĐÃ TỒN TẠI TRÊN GOOGLE CALENDAR ---

existing\_google\_event\_dates\_set = fetch\_google\_events\_in\_range(service, calendar\_id, START\_YEAR\_ICS, END\_YEAR\_ICS)

try:

with open(ICS\_FILE, 'rb') as f:

gcal = Calendar.from\_ical(f.read())

except FileNotFoundError:

print(f"Lỗi đọc file ICS: Không tìm thấy tệp ICS đã sửa '{ICS\_FILE}'.", file=sys.stderr)

print("Hãy chắc chắn rằng bước xử lý file đã thành công trước khi chạy nhập.", file=sys.stderr)

return

except Exception as e:

print(f"Lỗi khi đọc và phân tích tệp ICS '{ICS\_FILE}': {e}", file=sys.stderr)

if hasattr(e, '\_\_cause\_\_') and e.\_\_cause\_\_:

print(f"Chi tiết lỗi từ icalendar: {e.\_\_cause\_\_}", file=sys.stderr)

print("Có thể file ICS vẫn còn lỗi định dạng hoặc thư viện icalendar gặp vấn đề.", file=sys.stderr)

return

# Lọc ra chỉ các VEVENT

events\_to\_import = [component for component in gcal.walk() if component.name == "VEVENT"]

total\_events\_ics = len(events\_to\_import)

if total\_events\_ics == 0:

print(f"Không tìm thấy sự kiện nào (VEVENT) trong tệp ICS '{ICS\_FILE}' để nhập.")

return

# --- KHỞI TẠI Batching ---

batch = service.new\_batch\_http\_request()

batch\_requests\_count = 0

total\_skipped\_duplicate\_count = 0

total\_error\_processing\_count = 0

events\_chosen\_for\_insert = 0

# --- HÀM CALLBACK XỬ LÝ KẾT QUẢ CỦA TỪNG YÊU CẦU TRONG BATCH ---

def batch\_callback(request\_id, response, exception):

global successful\_imports\_count

global conflict\_409\_count

global forbidden\_403\_count

global other\_api\_errors\_count

if exception:

if isinstance(exception, HttpError):

status\_code = exception.resp.status

if status\_code == 409:

conflict\_409\_count += 1

elif status\_code == 403:

forbidden\_403\_count += 1

print(f"--- Lỗi CẤM (403 Forbidden) cho sự kiện gốc #{request\_id} ---", file=sys.stderr)

else:

other\_api\_errors\_count += 1

print(f"--- Lỗi HTTP API khác (Status: {status\_code}) cho sự kiện gốc #{request\_id} ---", file=sys.stderr)

else:

other\_api\_errors\_count += 1

print(f"--- Lỗi API không xác định cho sự kiện gốc #{request\_id} ---", file=sys.stderr)

else:

successful\_imports\_count += 1

print(f"\nĐang chuẩn bị nhập {total\_events\_ics} sự kiện từ file ICS vào lịch có ID '{calendar\_id}' (sử dụng batching và kiểm tra trùng lặp chỉ dựa trên ngày)...")

last\_batch\_execute\_time = time.time()

# --- BẮT ĐẦU VÒNG LẶP DUYỆT CÁC SỰ KIỆN TỪ FILE ICS ---

for idx, component in enumerate(events\_to\_import, start=1):

raw\_summary = str(component.get('summary', ''))

summary = raw\_summary

event\_color\_id = None

start\_date\_only = None

try:

# --- BƯỚC Xử LÝ 1: Xử lý dữ liệu thời gian ---

dtstart\_ical = component.get('dtstart')

if not dtstart\_ical:

print(f"[{idx}/{total\_events\_ics}] Bỏ qua sự kiện (tiêu đề gốc: '{raw\_summary}') do thiếu DTSTART.", file=sys.stderr)

total\_error\_processing\_count += 1

continue

dtstart\_obj = dtstart\_ical.dt

start\_date\_only = dtstart\_obj.date() if isinstance(dtstart\_obj, datetime) else dtstart\_obj

# --- BƯỚC Xử LÝ 2: Xử lý tiêu đề và màu sắc (Bao gồm thêm tên lễ) ---

match = re.match(r'^(\d{1,2})\/(\d{1,2})N?$', raw\_summary)

if match:

day\_str = match.group(1)

month\_str = match.group(2)

try:

day = int(day\_str)

month = int(month\_str)

key\_day\_month = f"{day}/{month}"

summary = f"{day}/{month} AL"

if key\_day\_month in special\_days:

special\_day\_name = special\_days[key\_day\_month]

event\_color\_id = '11' # ID màu '11' thường là màu đỏ

summary = f"{day}/{month} AL - {special\_day\_name}"

except ValueError as e:

print(f"[{idx}/{total\_events\_ics}] Cảnh báo: Lỗi chuyển đổi ngày/tháng từ tiêu đề gốc '{raw\_summary}': {e}. Giữ nguyên tiêu đề gốc.", file=sys.stderr)

summary = raw\_summary

# --- BƯỚC Xử LÝ 3: Tạo khóa ngày dương để kiểm tra trùng lặp ---

event\_date\_str = start\_date\_only.strftime('%Y-%m-%d') if start\_date\_only else None

except Exception as e:

print(f"\n--- Lỗi xử lý dữ liệu cho sự kiện [{idx}/{total\_events\_ics}] (tiêu đề gốc: '{raw\_summary}') ---", file=sys.stderr)

print(f"Exception: {e}", file=sys.stderr)

print("--------------------------", file=sys.stderr)

total\_error\_processing\_count += 1

continue

# --- BƯỚC 4: KIỂM TRA TRÙNG LẶP TRƯỚC KHI THÊM VÀO BATCH (CHỈ DỰA TRÊN NGÀY) ---

if event\_date\_str and event\_date\_str in existing\_google\_event\_dates\_set:

print(f"[{idx}/{total\_events\_ics}] Bỏ qua: Sự kiện vào ngày {event\_date\_str} có thể đã tồn tại trên lịch Google (dựa trên ngày). Tiêu đề ICS: '{summary}'", file=sys.stderr)

total\_skipped\_duplicate\_count += 1

continue

# --- BƯỚC 5: CHUẨN BỊ BODY VÀ THÊM VÀO BATCH ---

try:

dtend\_ical = component.get('dtend')

dtend\_obj = dtend\_ical.dt if dtend\_ical else None

end\_date\_only = None

if dtend\_obj:

end\_date\_only = dtend\_obj.date() if isinstance(dtend\_obj, datetime) else dtend\_obj

event\_body = {

'summary': summary,

'description': str(component.get('description', '')),

'reminders': {'useDefault': False, 'overrides': [{'method': 'popup', 'minutes': 10}]},

'start': {

'date': event\_date\_str,

'timeZone': 'Asia/Ho\_Chi\_Minh'

},

}

# Xử lý DTEND cho sự kiện cả ngày

if end\_date\_only:

end\_date\_exclusive = end\_date\_only

if end\_date\_exclusive <= start\_date\_only:

end\_date\_exclusive = start\_date\_only + timedelta(days=1)

event\_body['end'] = {

'date': end\_date\_exclusive.strftime('%Y-%m-%d'),

'timeZone': 'Asia/Ho\_Chi\_Minh'

}

else:

event\_body['end'] = {

'date': (start\_date\_only + timedelta(days=1)).strftime('%Y-%m-%d'),

'timeZone': 'Asia/Ho\_Chi\_Minh'

}

if event\_color\_id:

event\_body['colorId'] = event\_color\_id

except Exception as e:

print(f"\n--- Lỗi chuẩn bị event body cho sự kiện [{idx}/{total\_events\_ics}] (tiêu đề cuối: '{summary}') ---", file=sys.stderr)

print(f"Exception: {e}", file=sys.stderr)

total\_error\_processing\_count += 1

continue

# --- BƯỚC 6: THÊM YÊU CẦU THÊM SỰ KIỆN NÀY VÀO BATCH ---

batch.add(

service.events().insert(calendarId=calendar\_id, body=event\_body),

callback=batch\_callback,

request\_id=str(idx)

)

batch\_requests\_count += 1

events\_chosen\_for\_insert += 1

# --- BƯỚC 7: KIỂM TRA VÀ THỰC THI BATCH NẾU ĐẦY ---

if batch\_requests\_count >= BATCH\_SIZE:

print(f"Đang thực thi batch (gói {batch\_requests\_count} sự kiện). Đã xử lý {idx}/{total\_events\_ics} sự kiện ICS...", file=sys.stdout)

batch\_start\_time = time.time()

required\_batch\_duration = MIN\_DELAY\_BETWEEN\_BATCHES

elapsed\_since\_last\_batch\_start = batch\_start\_time - last\_batch\_execute\_time

if elapsed\_since\_last\_batch\_start < required\_batch\_duration:

sleep\_duration = required\_batch\_duration - elapsed\_since\_last\_batch\_start

print(f"Ngủ {sleep\_duration:.4f}s để tuân thủ rate limit giữa các batch.", file=sys.stdout)

time.sleep(sleep\_duration)

last\_batch\_execute\_time = time.time()

try:

batch.execute()

except Exception as e:

print(f"\nLỗi ở cấp độ Batch khi thực thi gói {idx}/{total\_events\_ics}: {e}.", file=sys.stderr)

batch = service.new\_batch\_http\_request()

batch\_requests\_count = 0

# --- SAU VÒNG LẶP DUYỆT ICS: THỰC THI CÁC YÊU CẦU CÒN LẠI TRONG BATCH CUỐI CÙNG ---

if batch\_requests\_count > 0:

print(f"Đang thực thi batch cuối cùng ({batch\_requests\_count} sự kiện). Đã xử lý hết {total\_events\_ics}/{total\_events\_ics} sự kiện ICS...", file=sys.stdout)

batch\_start\_time = time.time()

required\_batch\_duration = MIN\_DELAY\_BETWEEN\_BATCHES

elapsed\_since\_last\_batch\_start = batch\_start\_time - last\_batch\_execute\_time

if elapsed\_since\_last\_batch\_start < required\_batch\_duration:

sleep\_duration = required\_batch\_duration - elapsed\_since\_last\_batch\_start

print(f"Ngủ {sleep\_duration:.4f}s cho batch cuối cùng.", file=sys.stdout)

time.sleep(sleep\_duration)

last\_batch\_execute\_time = time.time()

try:

batch.execute()

except Exception as e:

print(f"\nLỗi ở cấp độ Batch khi thực thi gói cuối cùng: {e}.", file=sys.stderr)

# --- BƯỚC CUỐI CÙNG: TỔNG KẾT QUÁ TRÌNH NHẬP ---

print(f"\n--- Tổng kết quá trình nhập lịch (Kiểm tra trùng lặp chỉ dựa trên ngày + Batching) ---", file=sys.stdout)

print(f"Tổng số sự kiện tìm thấy trong tệp ICS: {total\_events\_ics}", file=sys.stdout)

print(f"Tổng số ngày dương duy nhất đã lấy sự kiện từ lịch Google để kiểm tra: {len(existing\_google\_event\_dates\_set)}", file=sys.stdout)

print(f"Số sự kiện từ ICS bị bỏ qua do đã tồn tại sự kiện trên cùng ngày trong lịch Google: {total\_skipped\_duplicate\_count}", file=sys.stdout)

print(f"Số sự kiện từ ICS gặp lỗi xử lý dữ liệu ban đầu (không thêm vào batch): {total\_error\_processing\_count}", file=sys.stdout)

print(f"Số sự kiện từ ICS được chọn để cố gắng nhập vào Google Calendar: {events\_chosen\_for\_insert}", file=sys.stdout)

print(f"Đã import thành công {successful\_imports\_count}/{events\_chosen\_for\_insert} sự kiện vào Google Calendar.", file=sys.stdout)

if conflict\_409\_count > 0:

print(f"Số lỗi 409 (Conflict - có thể do trùng lặp) gặp phải khi gọi API: {conflict\_409\_count}", file=sys.stdout)

if forbidden\_403\_count > 0:

print(f"Số lỗi 403 (Forbidden - không có quyền) gặp phải khi gọi API: {forbidden\_403\_count}", file=sys.stdout)

if other\_api\_errors\_count > 0:

print(f"Số lỗi API khác (không phải 409, 403) gặp phải khi gọi API: {other\_api\_errors\_count}", file=sys.stdout)

print("\nQuy trình kiểm tra và nhập sự kiện hoàn thành.", file=sys.stdout)

# --- KHỐI THỰC THI CHÍNH CỦA SCRIPT ---

if \_\_name\_\_ == '\_\_main\_\_':

print("--- BẮT ĐẦU QUY TRÌNH NHẬP LỊCH ÂM TỪ FILE CỤC BỘ (BATCHING + KIỂM TRA TRÙNG LẶP) ---", file=sys.stdout)

# Bước 1: Đọc và sửa lỗi định dạng file ICS TỪ FILE CỤC BỘ

print(f"\n--- BƯỚC 1: ĐỌC VÀ SỬA FILE ICS CỤC BỘ '{LOCAL\_ICS\_INPUT\_FILE}' ---", file=sys.stdout)

local\_file\_exists = os.path.exists(LOCAL\_ICS\_INPUT\_FILE)

file\_processed\_successfully = False

if local\_file\_exists:

try:

with open(LOCAL\_ICS\_INPUT\_FILE, 'r', encoding='utf-8') as f:

raw\_text = f.read()

fixed\_text = fix\_ics\_format(raw\_text)

with open(ICS\_FILE, 'w', encoding='utf-8') as f:

f.write(fixed\_text)

print(f"Đã đọc file '{LOCAL\_ICS\_INPUT\_FILE}' và lưu nội dung đã sửa vào '{ICS\_FILE}'.", file=sys.stdout)

file\_processed\_successfully = True

except FileNotFoundError:

print(f"Lỗi: Không tìm thấy tệp ICS cục bộ '{LOCAL\_ICS\_INPUT\_FILE}'. Vui lòng kiểm tra tên file và đường dẫn.", file=sys.stderr)

file\_processed\_successfully = False

except Exception as e:

print(f"Lỗi khi đọc hoặc sửa file ICS cục bộ '{LOCAL\_ICS\_INPUT\_FILE}': {e}", file=sys.stderr)

print("Đảm bảo file không bị hỏng và có quyền đọc.", file=sys.stderr)

file\_processed\_successfully = False

else:

print(f"Lỗi: Không tìm thấy tệp ICS cục bộ '{LOCAL\_ICS\_INPUT\_FILE}' trong thư mục hiện tại.", file=sys.stderr)

print("Hãy chắc chắn file ICS đầu vào nằm cùng thư mục với script hoặc cung cấp đường dẫn đầy đủ.", file=sys.stderr)

file\_processed\_successfully = False

# Chỉ tiếp tục nếu bước 1 thành công

if file\_processed\_successfully:

# Bước 2: Xác thực với Google API

print("\n--- BƯỚC 2: XÁC THỰC GOOGLE API ---", file=sys.stdout)

google\_service = get\_service()

if google\_service:

# Bước 3: Tìm hoặc tạo lịch Google Calendar đích

print("\n--- BƯỚC 3: TÌM HOẶC TẠO LỊCH ĐÍCH ---", file=sys.stdout)

target\_calendar\_id = get\_or\_create\_calendar(google\_service)

if target\_calendar\_id:

# Bước 4: Nhập các sự kiện từ file ICS đã sửa vào lịch đích

print(f"\n--- BƯỚC 4: NHẬP LỊCH VÀO GOOGLE CALENDAR (BATCHING + KIỂM TRA TRÙNG LẶP) TỪ FILE '{ICS\_FILE}' ---", file=sys.stdout)

import\_ics\_batched(google\_service, target\_calendar\_id)

print("\nQuy trình nhập sự kiện hoàn thành.", file=sys.stdout)

else:

print("\nKết thúc quy trình: Không thể tạo hoặc lấy lịch Google Calendar đích. Vui lòng kiểm tra lại quyền truy cập và cấu hình.", file=sys.stderr)

else:

print("\nKết thúc quy trình: Lỗi xác thực Google API. Không thể kết nối tới dịch vụ. Vui lòng kiểm tra lại credentials.json và kết nối mạng.", file=sys.stderr)

else:

print("\nKết thúc quy trình: Xử lý file ICS cục bộ không thành công. Không có dữ liệu nguồn để nhập.", file=sys.stderr)

print("\n--- KẾT THÚC TOÀN BỘ QUY TRÌNH SCRIPT ---", file=sys.stdout)