README.md

[CBIR] Dự án so sánh logo, phát hiện logo

Tài liệu hướng dẫn:

Mục đích chính của dự án:

- So sánh xem 2 ảnh đầu vào có phải cùng một logo hay không, nếu có sẽ in ra logo cần tìm
- Kiểm tra trong bức ảnh có phát hiện logo được yêu cầu

Các chức năng phụ:

- Thêm hình ảnh logo vào file dữ liệu
- Xóa toàn bộ ảnh của logo được chỉ định trong file dữ liệu

1. Cách cài đặt tại local:

1.1. Cài đặt python3, pip:

python:

- Đối với hệ điều hành window, xem hướng dẫn chi tiết tại: https://phoenixnap.com/kb/how-to-install-python-3-windows
- Đối với hệ điều hành ubuntu, xem hướng dẫn chi tiết tại: https://phoenixnap.com/kb/how-to-install-python-3-ubuntu Link tham khảo cài đặt python: https://www.python.org/downloads/release/python-3810/
  pip:
- Đối với hệ điều hành window, xem hướng dẫn chi tiết tại: https://phoenixnap.com/kb/install-pip-windows
- Đối với hệ điều hành ubuntu, xem hướng dẫn chi tiết tại: https://linuxize.com/post/how-to-install-pip-on-ubuntu-20.04/

1.2. Cài đặt môi trường:

Tại thư mục CBIR_logo, thực hiện lệnh trên command line để cài đặt toàn bộ thư viện sử dụng và các gói liên quan, cụ thể như sau:

```
pip install -r requirement.txt
```

Tại file **main.py**, địa chỉ IP mặc định là **host**: 0.0.0.0, cổng **port**: 5000, để thay đổi thì sửa lại **host** và **port** tương ứng, tại dòng 250 của file.
Cuối cùng, tại command line, thực hiện chạy lệnh sau để chạy chương trình:

```
python main.py
```

Thực hiện test API, được mô tả chi tiết ở file **README.md** (file cùng thư mục với file **main.py**):

- Check có logo trong ảnh
- Check 2 ảnh có cùng logo
- Thêm ảnh vào file dữ liệu logo đối sánh (**Yêu cầu**: cắt ảnh chỉ chứa logo cần check, không chứa các background không cần thiết, các yêu cầu liên quan đã nêu ở mục *2.1.1*)
- Xóa toàn bộ logo cần đối sánh

2. Hướng dẫn cách sử dụng test trên swagger:

2.1. Thêm hình ảnh logo vào file dữ liệu:

Truy cập vào trang http://46.137.245.145:5000 để bắt đầu test

2.1.1. Mô tả:

Đầu vào gồm tập hợp các ảnh chỉ chứa vùng logo, **YÊU CẦU**:

- **Số lượng ảnh tối thiểu** mỗi loại logo: 20 ảnh / loại logo
  - Ví dụ: Đối với logo của pepsi có nhiều loại, ví dụ 3 loại là logo trên nền đen, logo trên nền trắng, logo trên nền xanh. Đối với mỗi loại cần tối thiểu 20 ảnh thêm vào file dữ liệu, vậy tổng số lượng ảnh thêm vào của logo pepsi là 60 ảnh. **Chú ý:** Số lượng ảnh càng nhiều dẫn đến quá trình đọc càng chậm, do đó **số lượng thêm tối đa** của mỗi loại là 100 ảnh
  - Do sự đa dạng của mỗi logo khác nhau nên cần linh động về số lượng ảnh mỗi loại của logo
- Đối với mỗi bức ảnh, ảnh cắt chỉ nên chứa vùng **logo cần nhận diện** và thương hiệu đi kèm (nếu có)
- Ảnh thêm vào nên cắt ảnh có chất lượng không quá kém, kích thước không quá bé không quá lớn:
  - Dung lượng lý tưởng: 20KB - 200KB
  - Kích thước lý tưởng: 100px - 300px (tùy theo chiều ngang và chiều dọc của vùng chứa logo)
  - Ví dụ: Hình ảnh pepsi có dung lượng 22.4KB, kích thước chiều cao x chiều rộng tương ứng là: (113px, 90px)
- Hình ảnh logo nên đa dạng góc quay, kích thước.

2.1.2. Các bước thêm logo:

- Bước 1: Truy cập vào trang web đã được nêu ở trên
- Bước 2: Chọn mục add-logo
- Bước 3: Ấn vào button ***Try it out*** để thực hiện, gồm 2 bước:
  - ***Choose File*** để chọn file ảnh cần thêm
  - Điền vào ***tên logo*** cần thêm
- **Chú ý**: Hai thông tin này đều bắt buộc phải có

2.2. Xóa toàn bộ ảnh của logo được chỉ định trong file dữ liệu:

Đầu vào là tên của **logo cần xóa**. Sau khi gửi yêu cầu, toàn bộ thông tin (ảnh, thông tin nhận dạng) của logo yêu cầu đều bị xóa. Khi thực hiện xong thì không thể khôi phục lại thông tin bị xóa. Tuy nhiên, vẫn có thể thêm lại thông tin như mô tả ở mục trên, tiến hành thực hiện các bước thêm ảnh như trên.

==========

[CBIR] Project compare logo

API request tutorial document

Main objective This project:

- Compares images, are they similar?
- Images have logo predefined

Component function:

- Add logo to do compare data
- Delete logo in data

I. Main API:

I.1. Compares images, are they similar?

I.1.1. Request:

- URL: http://127.0.0.1:5000/compare

- Method: `POST`
- Format: `JSON`
- Input:

```
{
  "image": "link image or image 1",
  "image": "link image or image 2"
}
```

or if compare with label

```
{
  "image": [
    "link image or image 1",
    "link image or image 2"
  ],
  "label": "coca"
}
```

- Description:

```
|-------------------------------------------------------------|
| Name |  Type  |               Description                   |
|------|------- |---------------------------------------------|
|image | string |link image or image be encoded format base64|
|label | string |name label image to compare                  |
|-------------------------------------------------------------|
```

I.1.2. Response:

- Result:

```
{
    "label": "coca",
    "message": "success",
    "same": true,
    "status_code": 200
}
```

- Description:

```
|-------------------------------------------------------------------------|
|    Name   |  Type  |               Description                          |
|-----------|--------|----------------------------------------------------|
|label      | string |name logo or null                                   |
|message    | string |success or unsuccess                                |
|same       | boolean|value compare same or difference logo               |
|status_code| int    | status code: 200/400/500 corresponding to message|
|-------------------------------------------------------------------------|
```

I.2. Images have logo predefined

I.2.1. Request:

- URL: http://127.0.0.1:5000/check-logo
- Method: `POST`
- Format: `JSON`
- Input:

```
{
  "image": "link image",
```

```
  "label": "name label"
}
```

- Description:

```
|-------------------------------------------------------------|
| Name |  Type  |               Description                   |
|------|------- |---------------------------------------------|
|image | string |link image or image be encoded format base64|
|label | string |name logo necessary checked                  |
|-------------------------------------------------------------|
```

I.2.2. Response:

- Result:

```
{
    "has_logo": true,
    "message": "success",
    "status_code": 200
}
```

- Description:

```
|-----------------------------------------------------------------------------------------|
|    Name   |  Type   |               Description                                         |
|-----------|---------|---------------------------------------------------------------|
|has_logo   | boolean |if image have logo is true and false if not have logo, orther null|
|message    | string  |success or unsuccess                                           |
|status_code| int     | status code: 200/400/500 corresponding to message             |
|-----------------------------------------------------------------------------------------|
```

II. Other API:

II.1 Add logo to file json

II.1.1. Request:

- URL: http://127.0.0.1:5000/add-logo
- Method: `POST`
- Format: `JSON`
- Input:

```
{
  "image": "link image or image or list image",
  "label": "name label"
}
```

- Description:

```
|-------------------------------------------------------------|
| Name |  Type  |               Description                   |
|------|------- |---------------------------------------------|
|image | string |link image or image be encoded format base64|
|label | string |name label necessary add file json           |
|-------------------------------------------------------------|
```

II.1.2. Response:

- Result:

```
{
    "add_logo": true,
    "message": "success",
    "label": "coca",
    "status_code": 200
}
```

- Description:

```
|-------------------------------------------------------------------------|
|    Name   |  Type   |               Description                         |
|-----------|---------|---------------------------------------------------|
|add_logo   | boolean |value True if add success                          |
|message    | string  |success or unsuccess                               |
|label      | string  |label of image added file json                     |
|status_code| int     |status code: 200/400/500 corresponding to message  |
|-------------------------------------------------------------------------|
```

II.2 Delete logo in file json

II.1.1. Request:

- URL: http://127.0.0.1:5000/delete_logo
- Method: `POST`
- Format: `JSON`
- Input:

```
{
  "label": "name label necessary delete"
}
```

- Description:

```
|-------------------------------------------------------------|
| Name |  Type  |               Description                   |
|------|------- |---------------------------------------------|
|label | string |name label necessary delete                  |
|-------------------------------------------------------------|
```

II.1.2. Response:

- Result:

```
{
    "deleted": true,
    "message": "success",
    "label": "coca",
    "status_code": 200
}
```

- Description:

```
|-------------------------------------------------------------------------|
|    Name   |  Type   |               Description                         |
|-----------|---------|---------------------------------------------------|
|deleted    | boolean |value True if label deleted                        |
|message    | string  |success or unsuccess                               |
|label      | string  |name label is deleted                              |
|status_code| int     |status code: 200/400/500 corresponding to message  |
|-------------------------------------------------------------------------|
```

==========