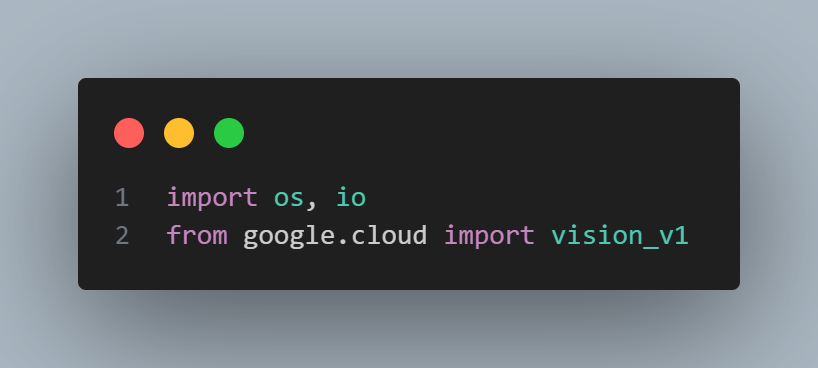
**NHẬN DẠNG BIỂU CẢM KHUÔN MẶT VỚI RASPBERRY PI THÔNG QUA GOOGLE VISION API**

1. **Cài đặt thư viện cần thiết**

* Cài thư viện: **pip install google-cloud-vision**

1. **Import các thư viện**



- **os**: Cung cấp các chức năng cho việc tương tác với hệ điều hành, trong trường hợp này, thiết lập biến môi trường.

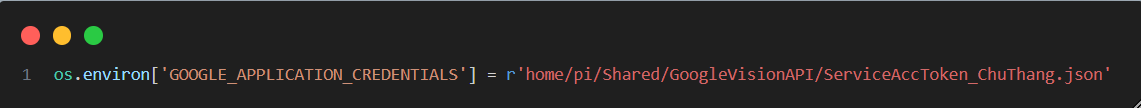
- **io**: Cung cấp các hoạt động đầu vào và đầu ra, đặc biệt là cho việc xử lý tệp tin.

- **google.cloud.vision\_v1**: Thư viện Google Cloud Vision API cho việc xử lý hình ảnh và nhận diện văn bản.

Chi tiết code tại đây

|  |
| --- |
| import os, io  from google.cloud import vision\_v1 |

1. **Thiết lập thông tin xác thực của google API**

****

Câu lệnh trên được sử dụng để thiết lập biến môi trường **GOOGLE\_APPLICATION\_CREDENTIALS** trong môi trường thực thi của chương trình.

Giải thích chi tiết hơn:

- **os.environ** là một từ điển chứa các biến của môi trường hệ thống

- **['GOOGLE\_APPLICATION\_CREDENTIALS']**: Đây là cách truy cập hoặc thiết lập giá trị của biến môi trường cụ thể trong từ điển os.environ

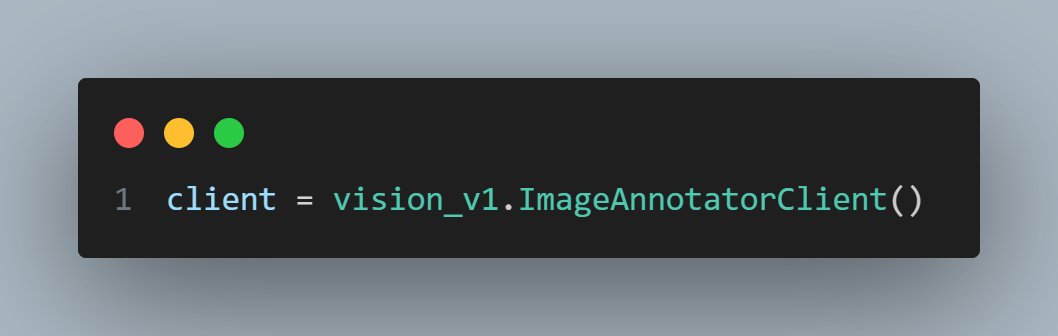
- **ServiceAccToken\_ChuThang.json** Đây là giá trị mà biến môi trường ['GOOGLE\_APPLICATION\_CREDENTIALS'] sẽ được thết lập thành. Trong trường hợp này, giá trị là một đường dẫn đến tệp JSON (tệp này được tải về tự động sau khi hoàn thành bước 4 của cách hướng dẫn dùng GGAPI ở phía trên)

Khi biến môi trường **['GOOGLE\_APPLICATION\_CREDENTIALS']** được thiết lập với giá trị của đường dẫn đến tệp JSON chứa thông tin xác thực, các thư viện và API của google cloud sẽ tự động sử dụng thông tin trong tệp JSON này để xác thực và cho phép ứng dụng truy cập vào các dịch vụ của google Cloud.

Chi tiết code tại đây:

|  |
| --- |
| os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = r'home/pi/Shared/GoogleVisionAPI/ServiceAccToken\_ChuThang.json' |

1. **Tạo một Client cho Google Cloud Vision API**



Dòng mã này để tạo một đối tượng client, tương tác với dịch vụ Google Cloud Vision API

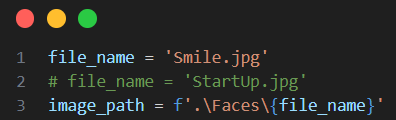
Giải thích chi tiết hơn:

* **client**: là tiên biến sử dụng để lưu trữ một đối tượng của lớp **ImageAnotatoClient** đã được import từ thư viện **google.cloud.vision\_v1**. Đối tượng nay sẽ là cầu nối giữa ứng dụng của mình và dịch vụ bên Google Vision API. Nó cung cấp các phương thức để gửi yêu cầu đến API và xử lý phản hồi từ API.

Chi tiết code tại đây:

|  |
| --- |
| client = vision\_v1.ImageAnnotatorClient() |

1. **Xác định tệp hình ảnh cần xử lý:**



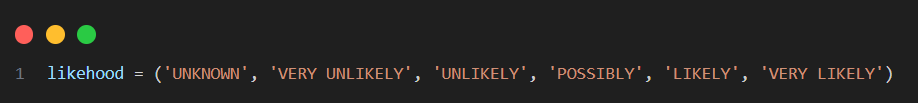
**file\_name**: Tên tệp hình ảnh (trong trường hợp này là "Smile.jpg"). Bạn có thể thay đổi tên tệp nếu cần.

**image\_path**: Đường dẫn đến tệp hình ảnh, được tạo ra bằng cách sử dụng file\_name. Lưu ý rằng tệp hình ảnh được mong đợi nằm trong một thư mục con có tên "Faces."

Chi tiết code tại đây:

|  |
| --- |
| file\_name = 'Smile.jpg'  # file\_name = 'StartUp.jpg'  image\_path = f'.\Faces\{file\_name}' |

1. Khai báo tuple chỉ mức độ



* Khai báo một tuple có tên là likehood. Tuple này chứa các chuỗi mô tả xác suất (likelihood) tương ứng với các cảm xúc của khuôn mặt được phát hiện. Cụ thể, nó đang định nghĩa các mức xác suất sau:

|  |  |
| --- | --- |
| 'UNKNOWN' | Không xác định |
| 'VERY UNLIKELY' | Rất không có khả năng |
| 'UNLIKELY' | Không có khả năng |
| 'POSSIBLY' | Có khả năng |
| 'LIKELY' | Có thể |
| 'VERY LIKELY' | Rất có khả năng |

* Các mức xác suất này liên quan đến cách Google Cloud Vision API đánh giá cảm xúc của khuôn mặt sau khi phân tích hình ảnh. Khi API phát hiện một khuôn mặt và ước tính xác suất cho các cảm xúc như tức giận, vui mừng, buồn, ngạc nhiên, nó sẽ gán một giá trị xác suất vào các thuộc tính tương ứng của khuôn mặt. Sau đó, trong mã, các giá trị này được sử dụng để tìm mô tả tương ứng trong tuple likehood.

Ví dụ, nếu API ước tính rằng một khuôn mặt có xác suất "POSSIBLY" (Có khả năng) cho cảm xúc vui mừng, thì likehood[face.joy\_likelihood] sẽ trả về chuỗi "POSSIBLY," và thông tin này sẽ được in ra màn hình trong dòng print(' Joy likelyhood : {0}'.format(likehood[face.joy\_likelihood])).

Chi tiết code tại đây:

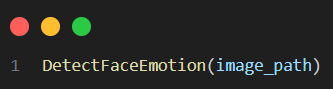
|  |
| --- |
| likehood = ('UNKNOWN', 'VERY UNLIKELY', 'UNLIKELY', 'POSSIBLY', 'LIKELY', 'VERY LIKELY') |

1. Hàm xử lý nhận diện biểu cảm khuôn mặt



|  |  |
| --- | --- |
| Bước 1 | **def DetectFaceEmotion(file\_path)**: Đây là khai báo hàm DetectFaceEmotion với một đối số file\_path, mục đích của hàm này là phân tích hình ảnh được chỉ định bởi file\_path để phát hiện khuôn mặt và đánh giá các cảm xúc của khuôn mặt trong hình. |
| Bước 2 | **with io.open(image\_path, 'rb') as image\_file**: Hàm bắt đầu bằng việc mở tệp hình ảnh được truyền vào (được chỉ định bởi file\_path) trong chế độ đọc nhị phân ('rb'). Câu lệnh này đảm bảo rằng tệp sẽ được đóng đúng cách sau khi kết thúc việc đọc.  **content = image\_file.read()**: Nội dung của tệp hình ảnh được đọc và lưu trữ trong biến content. Đây là dữ liệu hình ảnh mà hàm sau đó sẽ sử dụng để gửi đến Google Cloud Vision API để phân tích. |
| Bước 3 | **image = vision\_v1.Image(content=content)**: Đối tượng hình ảnh image được tạo ra từ dữ liệu hình ảnh content |
| Bước 4 | **response = client.face\_detection(image=image)**: Đoạn mã gửi image đến Google Cloud Vision API để phát hiện khuôn mặt trong hình. Kết quả từ API được lưu trữ trong biến response. |
| Bước 5 | **faceAnnotation = response.face\_annotations**: Thông tin về các khuôn mặt được phát hiện trong hình được trích xuất từ phản hồi của API và lưu trữ trong biến faceAnnotation |
| Bước 6 | In thông tin về các khuôn mặt đã phát hiện:  Dòng **print(f"Total number of people in the photo: {len(faceAnnotation)}")** in ra tổng số người trong hình dựa trên số khuôn mặt đã phát hiện.  Dòng **print(f"Faces:")** in ra dòng "Faces:" để bắt đầu in thông tin về các khuôn mặt. |
| Bước 7 | Trong từng vòng lặp, thông tin về từng khuôn mặt được in ra màn hình với các dòng sau:   * **print(f' Detection Confidence : {face.detection\_confidence}'):** In ra độ tin cậy của việc phát hiện khuôn mặt. * **print(f' Angry likelyhood : {likehood[face.anger\_likelihood]}')**: In ra xác suất tức giận của khuôn mặt dựa trên face.anger\_likelihood. * **print(f' Joy likelyhood : {likehood[face.joy\_likelihood]}')**: In ra xác suất vui mừng của khuôn mặt dựa trên face.joy\_likelihood. * **print(f' Sorrow likelyhood : {likehood[face.sorrow\_likelihood]}')**: In ra xác suất buồn của khuôn mặt dựa trên face.sorrow\_likelihood. * **print(f' Sup likelyhood : {likehood[face.surprise\_likelihood]}')**: In ra xác suất ngạc nhiên của khuôn mặt dựa trên face.surprise\_likelihood. * **print(f' Headwear likelyhood : {likehood[face.headwear\_likelihood]}')**: In ra xác suất đội đầu (headwear) của khuôn mặt dựa trên face.headwear\_likelihood. |

1. Gọi hàm thực hiện



* Gọi hàm và truyền vào đường dẫn ảnh

**CHI TIẾT TOÀN BỘ CODE**

|  |
| --- |
| import os, io  from google.cloud import vision\_v1  import pandas as pd  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = r'ServiceAccToken\_ChuThang.json'  client = vision\_v1.ImageAnnotatorClient()  file\_name = 'Smile.jpg'  # file\_name = 'StartUp.jpg'  image\_path = f'.\images\Faces\{file\_name}'  likehood = ('UNKNOWN', 'VERY UNLIKELY', 'UNLIKELY', 'POSSIBLY', 'LIKELY', 'VERY LIKELY')  def DetectFaceEmotion(file\_path):      with io.open(image\_path, 'rb') as image\_file:          content = image\_file.read()      image = vision\_v1.Image(content=content)      response = client.face\_detection(image=image)      faceAnnotation = response.face\_annotations      print(f"Total number of people in the photo: {len(faceAnnotation)}")      print(f"Faces:")      for face in faceAnnotation:          print(f'    Detection Confidence : {face.detection\_confidence}')          print(f'    Angry likelyhood     : {likehood[face.anger\_likelihood]}')          print(f'    Joy likelyhood       : {likehood[face.joy\_likelihood]}')          print(f'    Sorrow likelyhood    : {likehood[face.sorrow\_likelihood]}')          print(f'    Sup likelyhood       : {likehood[face.surprise\_likelihood]}')          print(f'    Headwear likelyhood  : {likehood[face.headwear\_likelihood]}')          print()    DetectFaceEmotion(image\_path) |