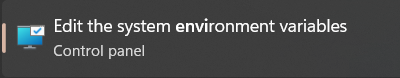
**Cài đặt các gói cần thiết cho dự án NHẬN DIỆN KHUÔN MẶT QUA ẢNH**

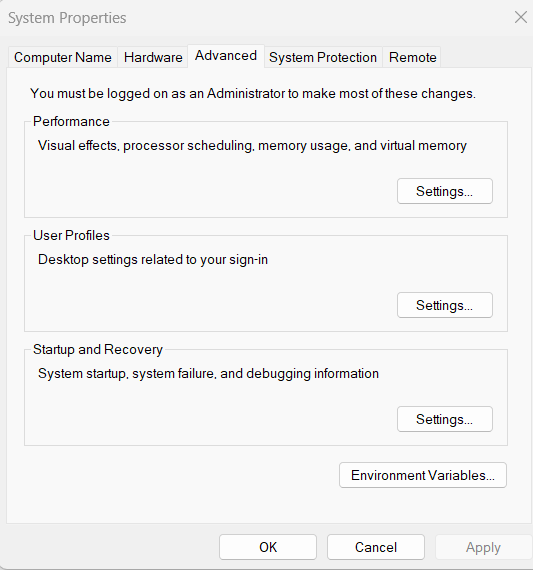
Trước hết cần có CMake, Visual C++, và Python.

Để cài đặt CMake, ta truy cập link sau: <https://cmake.org/download/> , sau khi đã cài bản phù hợp thì thêm Đường dẫn sau vào PATH của môi trường hệ thống như sau:

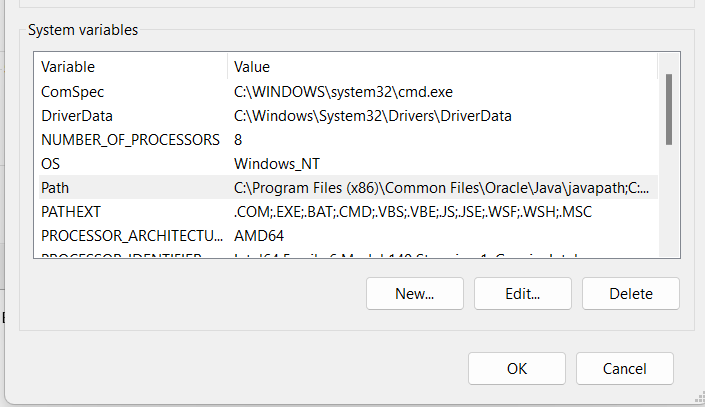
Nhập vào hộp thoại và mở ứng dụng: như bên dưới



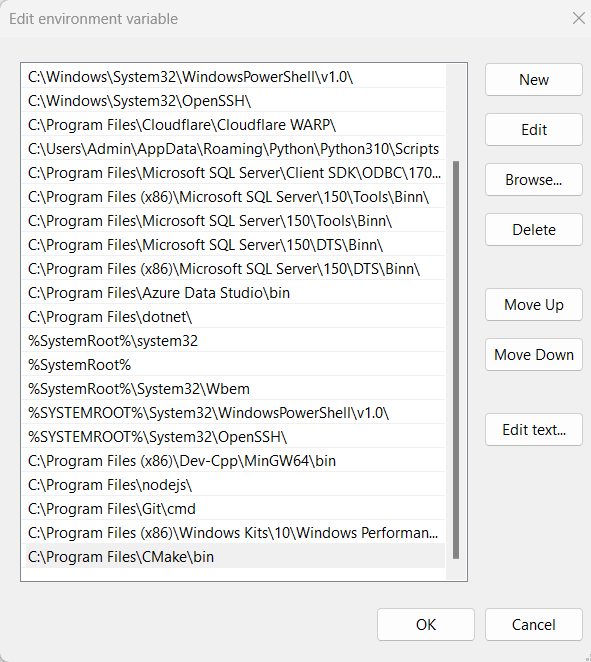
Mở Environment Variables…



Tiếp tới chọn Path và nhấn vào Edit…



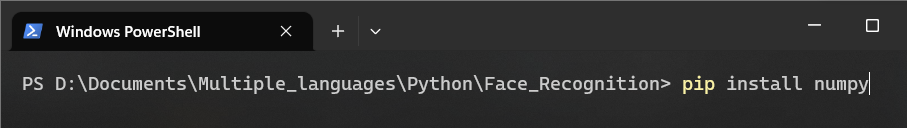
Chọn New và thêm đường dẫn C:\Program Files\CMake\bin



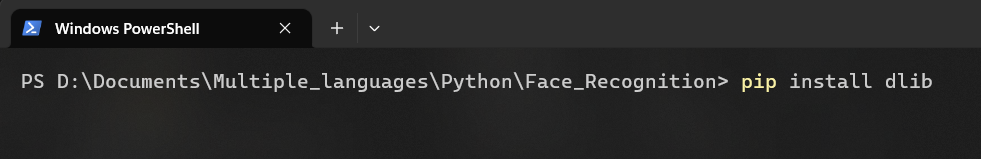
Nhấn OK đến khi kết thúc

Sau khi đã cài đầy đủ như các phần trên, ta thực hiện cài các gói sau

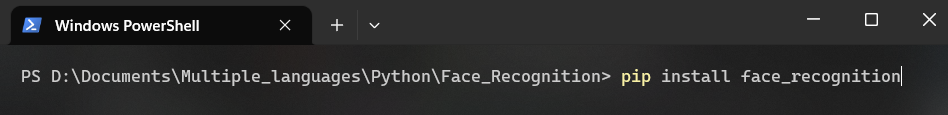
* Cài gói **Numpy**: pip install numpy



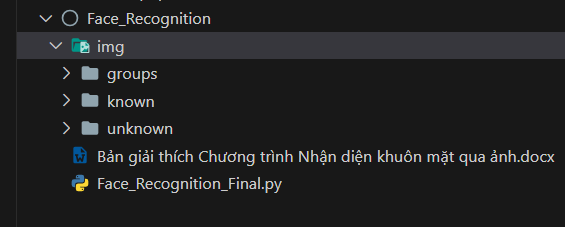
* Cài gói **dlib**: pip install dlib



* Cài gói **face\_recognition**: pip install face\_recognition



**Cấu trúc thư mục dự án**

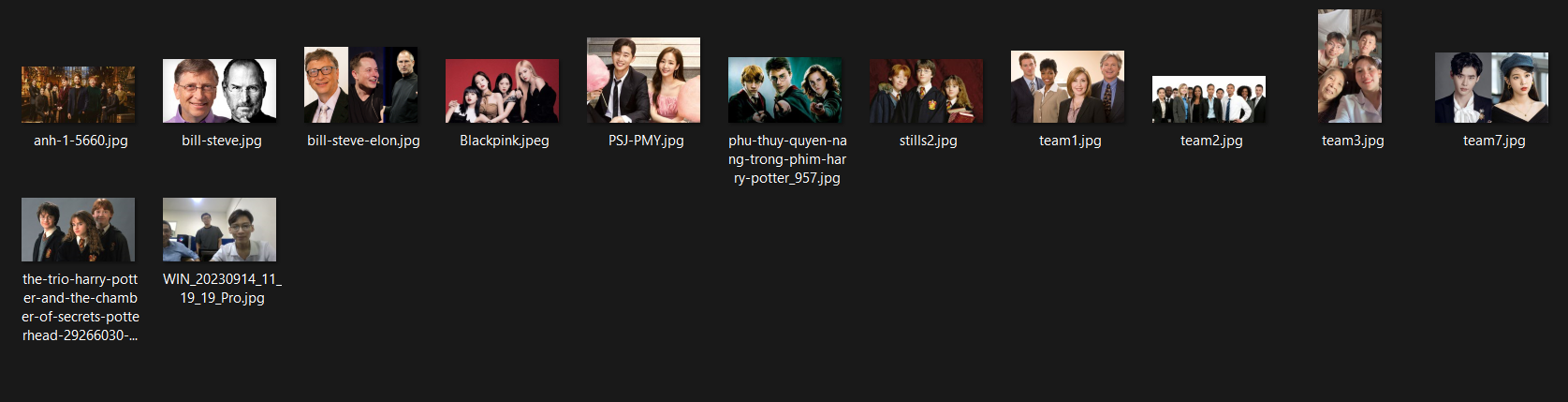
****

Tạo các thư mục dự án theo hình ảnh trên

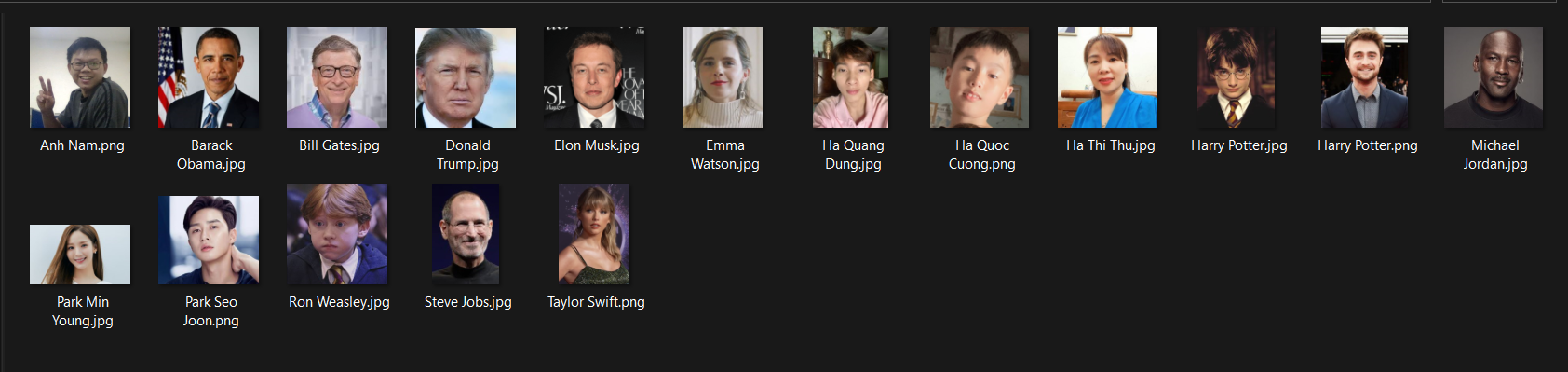
* Trước tiên là Folder dự án: Face\_Recognition
* Trong đó sẽ tạo Thư mục img dùng để chứa ảnh
* Tạo file Dự án Face\_Recognition.py ngang cấp với thư mục img

Tại thư mục img:

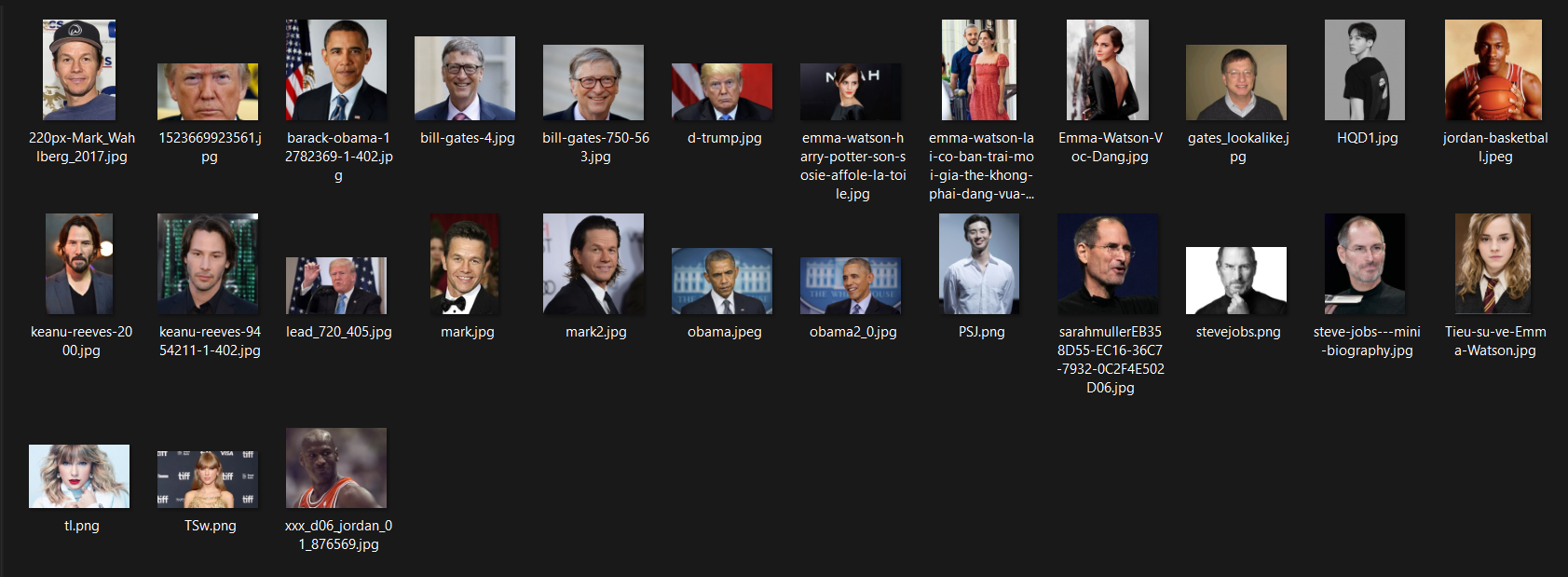
* Tạo 3 thư mục con chứa ảnh theo nhóm
* Groups: dùng để chứa ảnh một nhóm người



* known: dùng để chứa khuôn mặt của những người đã biết, Cần đặt tên file theo đúng tên của người đó khi nhận diện có thể hiện tên đúng



* unknown: dùng để chứa khuôn mặt những người chưa biết hoặc cần nhận diện



**Các bước thực hiện dự án**

1. **Trước tiên ta import các thư viện sau**

|  |  |  |
| --- | --- | --- |
| **os** | Dùng để tương tác với hệ điều hành. Cụ thể như làm việc với thư mục, với tệp, Thực thi lệnh hệ thống, làm việc với môi trường   |  | | --- | | **import os** | |
| **face\_recognition** | Dùng để nhận diện khuôn mặt, nhận diện khuôn mặt đã biết, xác định các điểm trên khuôn mặt (mắt, mũi, miệng,…), thao tác trên khuôn mặt (cắt, xử lý, thay đổi kích thước khuôn mặt); Làm việc với video (Nhận diện khuôn mặt trong video)   |  | | --- | | **import face\_recognition** | |
| **numpy** | Dùng để thao tác Mảng, Phép toán số học, toán học và thống kê, thao tác trên mảng, lập trình vector hóa   |  | | --- | | **import numpy as np** | |
| **PIL** | Từ thư viện **PIL (Python Imaging Library)** import lớp:   * **Image**: Dùng để tạo đối tượng hình ảnh, thực hiện các thao tác như: mở hình ảnh, thay đổi định dạng, thay đổi kích thước * **ImageDraw**: vẽ các hình vẽ lên ảnh  |  | | --- | | **from PIL import Image, ImageDraw** | |
| **tkinter** | GUI - Thư viện giao diện người dùng, tạo các ứng dụng GUI đơn giản. as tk dùng để đặt tên ngắn gọn lại thay cho tkinter  Từ thư viện tkinter import module filedialog: Cho phép mở hộp thoại chọn và lưu tệp từ hệ thống tệp của máy tính   |  | | --- | | **import tkinter as tk**  **from tkinter import filedialog** | |

Chi tiết import các thư viện tại đây:

|  |
| --- |
| import os  import face\_recognition  import numpy as np  from PIL import Image, ImageDraw  import tkinter as tk  from tkinter import filedialog |

1. **Các hàm cần thiết cho việc nhận diện:**
2. **Hàm xử lý nhận diện khuôn mặt (FaceRecognition)**

|  |  |  |
| --- | --- | --- |
| Bước 0 | Đặt tên hàm và tham số được truyền vào là đường dẫn của file   |  | | --- | | **def FaceRecognition(file\_path):** | |
| Bước 1 | Khởi tạo 2 danh sách rỗng   * **known\_face\_ecodings**: Lưu trữ các khuôn mặt đã biết * **known\_face\_names**: Lưu trữ tên các khuôn mặt đã biết   Chi tiết code tại đây:   |  | | --- | | **known\_face\_encodings = []**  **known\_face\_names = []** | |
| Bước 2 | Vòng lặp **for img in os.listdir(f'./img/known'):** xử lý các khuôn mặt đã biết. Chương trình sẽ lặp toàn bộ những gì có trong thư mục known của hệ thống.   * Chúng ta cần tải khuôn mặt đã biết từ đường dẫn hình ảnh dựa vào câu lệnh   **face\_image = face\_recognition.load\_image\_file(f'./img/known/{img}')**   * Tiếp theo, chương trình sử dụng **face\_recognition.face\_encodings** để tính toán mã khuôn mặt của những khuôn mặt đã biết:   **face\_encoding = face\_recognition.face\_encodings(face\_image)[0]**   * Thực hiện lưu lại các mã khuôn mặt, tên khuôn mặt đã biết vào 2 mảng đã khai báo ở bước trên:   **known\_face\_encodings.append(face\_encoding)**  **known\_face\_names.append(img)**  Chi tiết code tại đây:   |  | | --- | | **for img in os.listdir(f'./img/known'):**  **face\_image = face\_recognition.load\_image\_file(f'./img/known/{img}')**  **face\_encoding = face\_recognition.face\_encodings(face\_image)[0]**  **known\_face\_encodings.append(face\_encoding)**  **known\_face\_names.append(img)** | |
| Bước 3 | Sau khi thoát khỏi vòng lặp, chúng ta sẽ tải ảnh lên bức ảnh cần test dựa vào câu lệnh  test\_image = face\_recognition.load\_image\_file(file\_path) |
| Bước 4 | Tiếp theo, chương trình sẽ thực hiện phân tích vị trí các khuôn mặt có trong ảnh thông qua câu lệnh **face\_locations = face\_recognition.face\_locations(test\_image)**, face\_recognition.locations sẽ quét test\_image.   * Kết quả trả về sẽ là 1 danh sách, mỗi phần tử trong đó đại diện cho 1 vị trí khuôn mặt. * Mỗi phần tử là 1 touple chứa 4 giá trị (top, right, bottom, left)   + top: tọa độ y góc trên cùng   + right: tọa độ x góc trên bên phải   + bottom: tọa độ y góc dưới   + left: tọa độ x góc trái bên dưới * Chương trình sẽ chạy **face\_encodings = face\_recognition.face\_encodings(test\_image, face\_locations)** để tính toán mã khuôn mặt và lưu vào biến face\_ecodings   Chi tiết code tại đây:   |  | | --- | | **face\_locations = face\_recognition.face\_locations(test\_image)**  **face\_encodings = face\_recognition.face\_encodings(test\_image, face\_locations)** | |
| Bước 5 | Chuyển bức ảnh cần test (test\_image) sang định dạng pil để có thể vẽ lên ảnh thông qua câu lệnh:  pil\_image = Image.fromarray(test\_image) |
| Bước 6 | Tạo mới draw để có thể vẽ hình vẽ và dán nhãn lên ảnh  draw = ImageDraw.Draw(pil\_image) |
| Bước 7 | Chương trình sẽ lặp qua các khuôn mặt thông qua vòng lặp **for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):**   * Vòng lặp này sẽ lặp đồng thời thông qua hai danh sách face\_location và face\_ecoding   + face\_locations: chứa các touple vị trí của khuôn mặt   + face\_ecodings: chứa mã khuôn mặt của các khuôn mặt tương ứng với face\_location  |  | | --- | | **matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding, tolerance=0.6)** |  * Đoạn mã này sẽ so sánh mã khuôn mặt hiện tại face\_ecoding với tất cả các khuôn mặt đã biết known\_face\_ecodings để xác định xem có trùng khớp không. Kết quả sẽ được lưu trong danh sách matches theo giá trị Boolean. Tham số tolerance xác định mức độ chấp nhận so sánh, nếu độ càng lớn thì việc chấp nhận càng cao, khả năng 2 khuôn mặt sẽ bị nhận dạng nhầm  |  | | --- | | **match\_scores = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)** |  * tính toán khoảng cách giữa khuôn mặt hiện tại và các mã khuôn mặt đã biết, kết quả tính toán được lưu trong **match\_scores** dưới dạng mảng * Gán giá trị mặc định cho **name = “Unknown”** * Gán **best\_match\_index = -1**, mặc định chỉ số này biểu trưng cho khuôn mặt không giống nhau. * Tiếp theo cần kiểm tra toàn bộ các giá trị đúng trong mảng matches bằng câu lệnh if, nếu đúng thì gán **best\_match\_index = np.argmin(match\_scores)** tức là lấy chỉ số index của giá trị min trong mảng match\_scores. Bởi giá trị match\_score càng thấp thì độ sai lệch khuôn mặt càng nhỏ. index đó sẽ được lưu vào biến **best\_match\_index** * Tiếp theo, Thực hiện câu lệnh điều kiện i**f best\_match\_index != -1:**   + Nếu đúng thì gán **name = known\_face\_names[best\_match\_index]** tại vị trí best\_match\_index vừa tìm được. Thực hiện vẽ hình vẽ lên khuôn mặt dựa theo chỉ số (top, right, bottom, left) thông qua câu lệnh   **draw.rectangle(((left, top), (right, bottom)), outline=(0, 255, 0)).** Tiếp đó sẽ vẽ nhãn tên dựa vào câu lệnh   |  | | --- | | **draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(0, 255, 0), outline=(0, 255, 0))**  **draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))** |   và hình vẽ nhận được sẽ có màu xanh lá theo mã màu fill=(0,255,0) theo mã RGB(đỏ, xanh lục, xanh lam)   * + Nếu sai thì biến name không được gán lại, nghĩa là “Unknown”. Và cũng sẽ vẽ hình vẽ và nhãn tên lên khuôn mặt dựa theo các chỉ số của location nhưng là màu đỏ fill(255,0,0)   Chi tiết code tại đây:   |  | | --- | | for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):      matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding, tolerance=0.6)      match\_scores  = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)      name = "Unknown"      best\_match\_index = -1      if True in matches:          best\_match\_index = np.argmin(match\_scores)      if best\_match\_index != -1:          name = known\_face\_names[best\_match\_index]          draw.rectangle(((left, top), (right, bottom)), outline=(0, 255, 0)) # Draw Box          draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(0, 255, 0), outline=(0, 255, 0)) # Draw label          draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))      else:          draw.rectangle(((left, top), (right, bottom)), outline=(255, 0, 0))          draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(255, 0, 0), outline=(255, 0, 0))          draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255)) | |
| Bước 8 | Khi thoát khỏi vòng lặp, ta sẽ giải phóng tài nguyên được sử dụng bởi đối tượng ImageDraw qua câu lệnh **del draw** tránh rò rỉ tài nguyên  **del draw** |
| Bước 9 | Cuối cùng ta hiển thị ảnh với đầy đủ ảnh, hình vẽ, nhãn tên sau khi nhận diện.  **pil\_image.show()** |

Chi tiết code hàm FaceRecognition tại đây:

|  |
| --- |
| # Hàm nhận diện hình ảnh  def FaceRecognition(file\_path):      known\_face\_encodings = []      known\_face\_names = []      # Xử lý các khuôn mặt đã biết      for img in os.listdir(f'./img/known'):          face\_image = face\_recognition.load\_image\_file(f'./img/known/{img}')          face\_encoding = face\_recognition.face\_encodings(face\_image)[0]          known\_face\_encodings.append(face\_encoding)          known\_face\_names.append(img)      # Tải ảnh lên      test\_image = face\_recognition.load\_image\_file(file\_path)      # Tìm kiếm tất cả các khuôn mặt có trong bức ảnh      face\_locations = face\_recognition.face\_locations(test\_image)      face\_encodings = face\_recognition.face\_encodings(test\_image, face\_locations)      # Chuyển đổi sang định dạng pil      pil\_image = Image.fromarray(test\_image)      # Tạo ImageDraw      draw = ImageDraw.Draw(pil\_image)      # Lặp qua các khuôn mặt      for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):          matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding, tolerance=0.6)          match\_scores  = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)          name = "Unknown"          best\_match\_index = -1          if True in matches:              best\_match\_index = np.argmin(match\_scores)          if best\_match\_index != -1:              name = known\_face\_names[best\_match\_index]              draw.rectangle(((left, top), (right, bottom)), outline=(0, 255, 0)) # Draw Box              draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(0, 255, 0), outline=(0, 255, 0)) # Draw label              draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))          else:              draw.rectangle(((left, top), (right, bottom)), outline=(255, 0, 0))              draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(255, 0, 0), outline=(255, 0, 0))              draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))      del draw      # Hiển thị ảnh đã nhận diện      pil\_image.show() |

1. **Hàm Mở hộp thoại hệ thống file và lấy đường dẫn file -**

|  |  |
| --- | --- |
| Bước 0 | Đặt tên hàm  **def open\_and\_recognize():** |
| Bước 1 | |  | | --- | | **file\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.jpg \*.jpeg \*.png \*.gif \*.bmp")])** |   Câu lệnh này sẽ khiến hộp thoại file được bật lên và cho phép người dùng chọn file ảnh (Chỉ nhận các định dạng file .jpg .jpeg .png .gif .bmp) và ảnh được chọn sẽ có đường dẫn được lưu vào biến **file\_path** |
| Bước 2 | Nếu file ảnh được chọn đúng định dạng, đường dẫn sẽ được in ra trong terminal và sau đó chương trình sẽ gọi tới hàm **FaceRecognition(file\_path)**   |  | | --- | | **if file\_path:**  **print('Đường dẫn:', file\_path)**  **FaceRecognition(file\_path)** | |

Chi tiết code tại đây:

|  |
| --- |
| def open\_and\_recognize():      file\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.jpg \*.jpeg \*.png \*.gif \*.bmp")])      if file\_path:          print('Đường dẫn:', file\_path)          FaceRecognition(file\_path) |

1. **Cửa sổ giao diện đồ họa**

* Giao diện đồ họa người dùng sẽ được chạy đầu tiên

|  |  |
| --- | --- |
| Bước 1 | Tạo một cửa sổ giao diện đồ hoạ và lưu đối tượng vào biến root.  **root = tk.TK()** |
| Bước 2 | Dùng để đặt kích thước cho cửa sổ root với kích thước chiều ngang 400px và chiều cao là 200px  **root.geometry("400x200")** |
| Bước 3 | Tiêu đề hiển thị trên cùng của cửa sổ bằng câu lệnh:  **root.title("App nhận diện khuôn mặt qua ảnh")** |
| Bước 4 | Chương trình sẽ tạo nút “Chọn ảnh” bằng câu lệnh  **open\_button = tk.Button(root, text="Chọn ảnh", command=open\_and\_recognize)**  Nút này sẽ có liên kết tới hàm open\_and\_recognize(), đồng nghĩa với việc khi nhấn nút này thì hàm open\_and\_recognize sẽ được gọi |
| Bước 5 | Đặt nút “Chọn ảnh” vào cửa sổ giao diện người dùng với khoảng cách dưới nút là 50px  **open\_button.pack(pady=50)** |
| Bước 6 | Bắt đầu vòng lặp với giao diện đồ hoạ, người dùng sẽ thực hiện các thao tác trên cửa sổ  **root.mainloop()** |

Chi tiết code tại đây:

|  |
| --- |
| # ======================= Giao diện đồ họa ====================  # Tạo cửa sổ GDDH  root = tk.Tk()  root.geometry("400x200")  root.title("App nhận diện khuôn mặt qua ảnh")  # Tạo nút chọn tệp  open\_button = tk.Button(root, text="Chọn ảnh", command=open\_and\_recognize)  open\_button.pack(pady=50)  # Khởi chạy giao diện đồ họa  root.mainloop() |

**CHI TIẾT CODE TOÀN DỰ ÁN**

|  |
| --- |
| import os  import face\_recognition  import numpy as np  from PIL import Image, ImageDraw  import tkinter as tk  from tkinter import filedialog  # ========================== Hàm phục vụ =====================  # Hàm nhận diện hình ảnh  def FaceRecognition(file\_path):      known\_face\_encodings = []      known\_face\_names = []      # Xử lý các khuôn mặt đã biết      for img in os.listdir(f'./img/known'):          face\_image = face\_recognition.load\_image\_file(f'./img/known/{img}')          face\_encoding = face\_recognition.face\_encodings(face\_image)[0]          known\_face\_encodings.append(face\_encoding)          known\_face\_names.append(img)      # Tải ảnh lên      test\_image = face\_recognition.load\_image\_file(file\_path)      # Tìm kiếm tất cả các khuôn mặt có trong bức ảnh      face\_locations = face\_recognition.face\_locations(test\_image)      face\_encodings = face\_recognition.face\_encodings(test\_image, face\_locations)      # Chuyển đổi sang định dạng pil      pil\_image = Image.fromarray(test\_image)      # Tạo ImageDraw      draw = ImageDraw.Draw(pil\_image)      # Lặp qua các khuôn mặt      for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):          matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding, tolerance=0.6)          match\_scores  = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)          name = "Unknown"          best\_match\_index = -1          if True in matches:              best\_match\_index = np.argmin(match\_scores)          if best\_match\_index != -1:              name = known\_face\_names[best\_match\_index]              draw.rectangle(((left, top), (right, bottom)), outline=(0, 255, 0)) # Draw Box              draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(0, 255, 0), outline=(0, 255, 0)) # Draw label              draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))          else:              draw.rectangle(((left, top), (right, bottom)), outline=(255, 0, 0))              draw.rectangle(((left, bottom - 20), (right, bottom)), fill=(255, 0, 0), outline=(255, 0, 0))              draw.text((left + 6, bottom - 20), name, fill=(255, 255, 255, 255))      del draw      # Hiển thị ảnh đã nhận diện      pil\_image.show()  # Hàm xử lý sự kiện người dùng nhấn nút "Chọn ảnh"  def open\_and\_recognize():      file\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.jpg \*.jpeg \*.png \*.gif \*.bmp")])      if file\_path:          print('Đường dẫn:', file\_path)          FaceRecognition(file\_path)  # ======================= Giao diện đồ họa ====================  # Tạo cửa sổ GDDH  root = tk.Tk()  root.geometry("400x200")  root.title("App nhận diện khuôn mặt qua ảnh")  # Tạo nút chọn tệp  open\_button = tk.Button(root, text="Chọn ảnh", command=open\_and\_recognize)  open\_button.pack(pady=50)  # Khởi chạy giao diện đồ họa  root.mainloop() |