HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

**⎯⎯⎯⎯⎯⎯⎯⎯⎯**



BÀI TẬP LỚN

**LẬP TRÌNH VỚI PYTHON**

**Đề tài : Nhận dạng khuôn mặt**

Nhóm 05 – Tổ 18 gồm 5 thành viên:

1. Lê Trung Hiếu - B21DCCN356

2. Nguyễn Tiến Hiệp - B21DCCN344

3. Nguyễn Anh Huân - B21DCCN402

4. Nguyễn Thế Linh - B21DCCN486

5. Trần Đức Lộc - B21DCCN492

**HÀ NỘI ,THÁNG 10/2023**

# Lời mở đầu

#### Trong thời đại số hóa ngày càng phát triển, ứng dụng công nghệ nhận dạng khuôn mặt đã trở thành một phần quan trọng trong cuộc sống hàng ngày của chúng ta. Các ứng dụng như mở khóa điện thoại di động, theo dõi thời gian làm việc của nhân viên, và kiểm soát an ninh trong các toà nhà và sân bay đều sử dụng công nghệ này. Python, một ngôn ngữ lập trình mạnh mẽ và phổ biến, đã trở thành một công cụ quan trọng để phát triển phần mềm nhận dạng khuôn mặt hiệu quả và linh hoạt.

#### Bài báo cáo này sẽ giới thiệu về việc sử dụng Python để phát triển một hệ thống nhận dạng khuôn mặt. Chúng ta sẽ tìm hiểu về các thư viện và công cụ có sẵn để xây dựng ứng dụng nhận dạng khuôn mặt, từ việc thu thập dữ liệu đào tạo đến việc triển khai hệ thống trong thực tế.

#### Nội dung của bài bài báo cáo này bao gồm 4 phần:

#### Phần I : Ứng dụng của nhận dạng khuân mặt trong đời sống

#### Phần II : Phương hướng triển khai của đề tài

#### Phần III : Cài đặt code

#### Phần IV : Kết quả

#### Nhóm chúng em hy vọng rằng thông qua bài báo cáo này, độc giả sẽ có cái nhìn tổng quan về cách Python có thể được áp dụng trong lĩnh vực nhận dạng khuôn mặt và sẽ cung cấp những kiến thức cơ bản để bạn có thể bắt đầu phát triển các ứng dụng nhận dạng khuôn mặt của riêng mình. Trong quá trình thực hiện đề tài không tránh khỏi những sai sót, nhóm em mong sẽ nhận được sự góp ý và đánh giá của thầy và các bạn.

**Xin chân thành cảm ơn!**

# CƠ HỘI VÀ NGUY CƠ CỦA CÔNG NGHẸ NHẬN DẠNG KHUÂN MẶT TRONG ĐỜI SỐNG

## Tiềm Năng Và Cơ Hội Của Công Nghệ Nhận Dạng Khuôn Mặt

### Công nghệ nhận dạng khuôn mặt đã thúc đẩy nhiều cơ hội và mang lại nhiều lợi ích trong nhiều lĩnh vực. Dưới đây là những khía cạnh tích cực của công nghệ này:

##### An ninh và bảo mật

#### Công nghệ nhận dạng khuôn mặt đóng vai trò quan trọng trong việc tăng cường an ninh và bảo mật. Nó giúp phát hiện và ngăn chặn tội phạm, đặc biệt là trong việc theo dõi người nghi ngờ và quản lý an toàn tại các cơ sở hạ tầng quan trọng.

##### Tiện Lợi và Hiệu Quả

#### Sử dụng khuôn mặt để đăng nhập vào các thiết bị và tài khoản trực tuyến giúp tiết kiệm thời gian và loại bỏ sự cần thiết của mật khẩu và mã PIN. Điều này không chỉ tạo ra trải nghiệm thuận tiện mà còn nâng cao bảo mật.

##### Ứng Dụng Y Tế

### Trong lĩnh vực y tế, công nghệ nhận dạng khuôn mặt có tiềm năng lớn. Nó có thể giúp xác định và theo dõi tình trạng sức khỏe của bệnh nhân, từ chẩn đoán bệnh đến quản lý dữ liệu y tế. Điều này giúp cải thiện chăm sóc sức khỏe và giúp các chuyên gia y tế làm việc hiệu quả hơn.

##### Giáo Dục

### Trong giáo dục, công nghệ nhận dạng khuôn mặt giúp kiểm soát học sinh, điểm danh tự động và quản lý dữ liệu giáo dục một cách hiệu quả. Nó có thể tạo ra môi trường học tập an toàn và hiệu quả hơn.

##### Cải Thiện Trải Nghiệm Khách Hàng

### Các doanh nghiệp có thể sử dụng công nghệ nhận dạng khuôn mặt để tạo ra trải nghiệm cá nhân hóa cho khách hàng. Điều này có thể bao gồm tư vấn sản phẩm hoặc dịch vụ dựa trên lịch sử mua sắm của khách hàng, tạo sự kết nối mạnh mẽ hơn.

## Nguy Cơ và Thách Thức Của Công Nghệ Nhận Dạng Khuôn Mặt

#### Bên cạnh những lợi ích, công nghệ nhận dạng khuôn mặt cũng đặt ra nhiều nguy cơ và thách thức cần được quản lý cẩn thận:

##### 2.1. Quyền Riêng Tư và Bảo Mật

### Một trong những nguy cơ lớn nhất liên quan đến công nghệ nhận dạng khuôn mặt là việc vi phạm quyền riêng tư và bảo mật của cá nhân. Dữ liệu khuôn mặt có thể bị lộ ra và lạm dụng, đặc biệt khi nó được thu thập và lưu trữ một cách không an toàn.

##### 2.2. Nhận Diện Sai Lầm

### Công nghệ nhận dạng khuôn mặt không phải lúc nào cũng chính xác. Có nguy cơ xảy ra những trường hợp nhận dạng sai lầm, dẫn đến những hậu quả nghiêm trọng như việc giam giữ hoặc kiểm duyệt sai.

##### 2.3. Sử Dụng Sai Mục Đích

#### Công nghệ nhận dạng khuôn mặt có thể được sử dụng để theo dõi và kiểm soát người dân một cách không minh bạch và sai mục đích. Chính phủ có thể sử dụng nó để kiểm soát dân số hoặc làm gián đoạn cuộc biểu tình dân chủ.

##### 2.4. Lạm Dụng Thương Mại

### Các doanh nghiệp có thể sử dụng công nghệ nhận dạng khuôn mặt để theo dõi và thu thập thông tin về khách hàng mà họ không chấp thuận. Điều này có thể dẫn đến việc lạm dụng dữ liệu và vi phạm quyền riêng tư của người dùng.

##### 2.4. Tạo Ra Khoảng Cách Kỹ Thuật

### Không phải ai cũng có quyền truy cập vào công nghệ nhận dạng khuôn mặt. Điều này có thể tạo ra khoảng cách kỹ thuật giữa các công dân, dẫn đến việc bất bình đẳng và phân biệt xã hội.

#### Tóm lại, công nghệ nhận dạng khuôn mặt mang lại cả những cơ hội và nguy cơ. Quản lý cẩn thận và đảm bảo tính minh bạch, quyền riêng tư và bảo mật dữ liệu là quan trọng để tận dụng lợi

# PHƯƠNG HƯỚNG TRIỂN KHAI CỦA ĐỀ TÀI

## Giao diện

Để tạo một ứng dụng nhận diện khuôn mặt với giao diện người dùng bằng Python, ta có thể làm theo các bước sau:

- Khởi tạo cửa sổ GUI chính và đặt tiêu đề cho cửa sổ GUI.

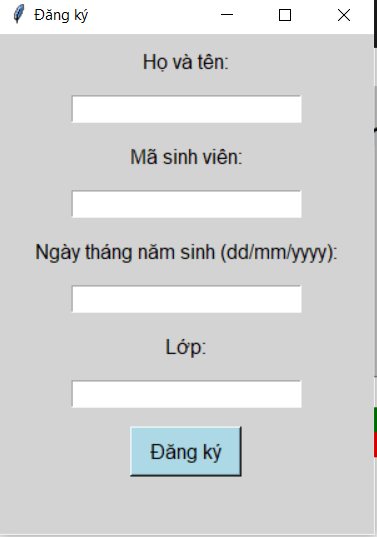
- Đặt kích thước, màu nền, kiểu chữ, màu chữ,… cho cửa sổ GUI chính.

- Tạo các nút để người dùng tương tác với hệ thống

- Tạo một biểu đồ cây (ttk.Treeview) với các cột dữ liệu và tiêu đề tương ứng, sẵn sàng để hiển thị dữ liệu trong ứng dụng, chẳng hạn như danh sách người dùng hoặc sinh viên cần điểm danh.



- Tạo một cửa sổ hộp thoại con (Toplevel) bên trong cửa sổ gốc (self.root) để hiển thị hộp thoại đăng ký. Hộp thoại con này là một cửa sổ phụ của cửa sổ gốc. Sau đó, tạo các trình nhập liệu để người dùng nhập thông tin tương ứng.



- Bắt đầu quá trình điểm danh.

- Hiển thị dữ liệu từ một tệp Excel trên cây (ttk.Treeview) trong giao diện người dùng.

Chi tiết triển khai sẽ được trình bày trong phần III.

## Hệ thống

1. Thuật toán LBPH (Local Binary Pattern Histogram)

### Sử dụng thư viện opencv với thuật toán LBPH thông qua “cv2.face\_LBPHFaceRecognizer.create()”

### Thuật toán Local Binary Pattern Histograms (LBPH) là một phương pháp trích xuất đặc trưng và nhận dạng khuôn mặt dựa trên ảnh. Nó không phụ thuộc vào mô hình toàn cục và được sử dụng rộng rãi trong các ứng dụng nhận dạng khuôn mặt bởi sự đơn giản và hiệu quả của nó. Dưới đây là cách thuật toán LBPH hoạt động để nhận dạng khuôn mặt bằng OpenCV:

### 1.1 Trích xuất đặc trưng Local Binary Pattern (LBP):

### - LBPH bắt đầu bằng việc chia ảnh thành các ô (cell) nhỏ hơn và đối tượng quan tâm (khuôn mặt) được chia thành các ô nhỏ hơn này.

### - Trong mỗi ô, LBPH sẽ xác định một điểm trung tâm và so sánh giá trị của các điểm lân cận với điểm trung tâm. Nếu giá trị của điểm lân cận lớn hơn hoặc bằng giá trị điểm trung tâm, nó sẽ được biểu diễn bằng "1," ngược lại, nó sẽ được biểu diễn bằng "0".

### - Chuỗi nhị phân được tạo ra từ việc so sánh này gọi là Local Binary Pattern (LBP) cho mỗi ô. LBP mô tả các biểu đồ cục bộ của khu vực quan tâm.

### 1.2. Xây dựng Histograms của LBP:

### - Sau khi đã rút trích các giá trị LBP cho từng ô, chúng ta sẽ xây dựng một histogram cho toàn bức ảnh bằng cách tổng hợp các giá trị LBP từ các ô.

### - Mỗi giá trị LBP sẽ là một bin trong histogram.

### 1.3. Đào tạo mô hình (tuỳ chọn):

### - LBPH có thể được đào tạo bằng cách cung cấp các ảnh khuôn mặt cùng với nhãn tương ứng của từng khuôn mặt để xây dựng một mô hình.

### - Mô hình này sẽ sử dụng các histogram LBP của các ảnh đào tạo để học cách phân biệt khuôn mặt từ các khu vực khác trong ảnh.

### 1.4. Nhận dạng:

### - Khi bạn muốn nhận dạng khuôn mặt trong một ảnh mới, bạn sẽ trích xuất histogram LBP từ ảnh đó.

### - Sau đó, bạn so sánh histogram LBP của ảnh này với các histogram đã học từ các ảnh đào tạo.

### - Kết quả là một điểm số (confidence score) thể hiện độ tương tự giữa ảnh mới và các khuôn mặt đã được đào tạo. Nếu điểm số này vượt qua một ngưỡng nhất định, thì khuôn mặt được nhận dạng.

1. Triển khai thuật toán

Để triển khai thuật toán LBPH (Local Binary Pattern Histogram) cho chương trình nhận dạng khuôn mặt bằng thư viện OpenCV, chúng ta cần thực hiện các bước sau:

Bước 1: Tạo Mô hình Nhận dạng Khuôn mặt LBPH:

Bước đầu tiên là tạo một mô hình nhận dạng khuôn mặt sử dụng thuật toán LBPH bằng cách sử dụng “cv2.face\_LBPHFaceRecognizer.create()”. Sau khi tạo mô hình, ta có thể tiến hành huấn luyện nó và sau đó sử dụng nó để nhận dạng khuôn mặt.



Bước 2: Huấn Luyện Mô hình Nhận dạng:

Để huấn luyện mô hình nhận dạng khuôn mặt, ta cần cung cấp cho nó dữ liệu huấn luyện, bao gồm ảnh khuôn mặt và nhãn tương ứng của người trong ảnh. Chúng ta cần duyệt qua từng hình ảnh và nhãn và gọi phương thức “train” của mô hình.

# Đọc dữ liệu huấn luyện và thêm ảnh và nhãn tương ứng vào danh sách



Bước 3: Nhận Dạng Khuôn mặt:

Sau khi mô hình đã được huấn luyện, ta có thể sử dụng nó để nhận dạng khuôn mặt trong các hình ảnh mới. Gọi phương thức `predict` của mô hình để nhận dạng khuôn mặt.



“serial” là nhãn của người được dự đoán, và `confidence` là độ tự tin của dự đoán.

Bước 4: Lưu Mô hình (Tuỳ chọn):

Sau khi mô hình đã được huấn luyện, ta có thể lưu nó vào một tệp để sử dụng lại sau này mà không cần huấn luyện lại. Điều này giúp tiết kiệm thời gian và công sức.



Bước 5: Sử Dụng Mô hình Đã Huấn Luyện:

Khi ta muốn sử dụng mô hình đã huấn luyện, ta có thể tải nó từ tệp và sử dụng nó để nhận dạng khuôn mặt trên hình ảnh mới.



Chi tiết triển khai thuật toán sẽ được trình bày trong phần III.

# CÀI ĐẶT CODE

## Cài đặt python

- Cài đặt python 3.12

## Cài đặt thư viện

- Cài đặt thư viện opencv

- Cài đặt thư viện openpyxl

## Cài đặt code

1. app.py

import tkinter as tk  
import openpyxl  
from tkinter import Toplevel, ttk  
from datetime import datetime  
from register import FaceRegister  
from attendance import FaceRecognition  
import threading  
import os  
import shutil  
  
class FaceRecognitionApp:  
 def \_\_init\_\_(self, root):  
 self.root = root  
  
 """  
 Tạo đường dẫn đến tệp Excel dữ liệu và lưu nó trong biến self.excel\_path,  
 os.path.join(os.path.dirname(\_\_file\_\_), "data1.xlsx") kết hợp đường dẫn của thư   
 mục hiện tại với tên tệp "data1.xlsx" để tạo đường dẫn đầy đủ đến tệp dữ liệu . Excel.  
 Điều này sẽ được sử dụng sau này để lưu và truy cập dữ liệu học sinh.  
 """  
 self.excel\_path = os.path.join(os.path.dirname(\_\_file\_\_), "data1.xlsx")  
  
 # tạo một đường dẫn đầy đủ đến tệp sao lưu dữ liệu Excel   
 self.backup\_excel\_path = os.path.join(os.path.dirname(\_\_file\_\_), “data1\_backup.xlsx")

"""  
 Tạo một đối tượng self.face\_attendance từ lớp FaceRecognition. Lớp   
 FaceRecognition được khởi tạo với hai đối số: đường dẫn đến tệp XML chứa mô   
 hình ("haarcascade\_frontalface\_default.xml") và đường dẫn đến tệp   
 nhận diện khuôn mặt Excel chứa dữ liệu học sinh ("data1.xlsx").   
 Đối tượng này được sử dụng để quản lý quá trình nhận diện khuôn mặt, điểm danh.  
 """  
 self.face\_attendance = FaceRecognition(  
 os.path.join(os.path.dirname(\_\_file\_\_), “haarcascade\_frontalface\_default.xml"),  
 os.path.join(os.path.dirname(\_\_file\_\_), "data1.xlsx")  
 )

"""  
 Tạo một đối tượng self.face\_register từ lớp FaceRegister.   
 Lớp FaceRegister được khởi tạo với hai đối số: đường dẫn đến thư mục   
 chứa hình ảnh và dữ liệu đăng ký khuôn mặt ("data") và đường dẫn đến   
 tệp YAML chứa mô hình nhận diện khuôn mặt đã đào tạo ("trainer.yml").   
 Đối tượng này được sử dụng để quản lý quá trình đăng ký khuôn mặt.  
 """  
 self.face\_register = FaceRegister(  
 data\_path=os.path.join(os.path.dirname(\_\_file\_\_), "data"),  
 recognizer\_path=os.path.join(os.path.dirname(\_\_file\_\_), "trainer.yml"),  
 )  
 self.root.title("Face Recognition")  
 self.root.geometry("600x600")  
 self.root.configure(bg="lightgrey")  
  
 """  
 Sử dụng phương thức after của cửa sổ để thực hiện phương thức   
 update\_table\_periodically sau 3000 miligiây (tức là 3 giây).   
 Điều này cho phép cập nhật dữ liệu hiển thị trong cây Treeview của ứng dụng một cách định kỳ.  
 """  
 self.root.after(3000, self.update\_table\_periodically)  
  
 #Tạo một đối tượng Label có tên self.frame\_label trong cửa sổ gốc self.root.  
 self.frame\_label = tk.Label(  
 self.root,  
 text="Hệ thống Nhận diện Khuôn mặt",  
 font=("Arial", 20, "bold"),  
 bg="lightgrey",  
 fg="black",  
 )  
 # Đặt nhãn self.frame\_label vào cửa sổ gốc bằng phương thức .pack()  
 self.frame\_label.pack(pady=20)  
  
 """  
 Tạo một đối tượng Button có tên self.register\_button   
 trong cửa sổ gốc self.root. Button này là một nút đăng ký  
 """  
 self.register\_button = tk.Button(  
 self.root,  
 text="Đăng ký",  
 command=self.show\_registration\_dialog,  
 font=("Arial", 14),  
 bg="lightblue",  
 fg="black",  
 padx=20,  
 pady=10,  
 )  
 self.register\_button.pack(pady=10)  
  
 self.attendance\_button = tk.Button(  
 self.root,  
 text="Điểm danh",  
 command=self.attendance,  
 font=("Arial", 14),  
 bg="lightgreen",  
 fg="black",  
 padx=20,  
 pady=10,  
 )  
 self.attendance\_button.pack(pady=10)  
  
 self.attendance\_process = None  
  
 #Cây này được sử dụng để hiển thị danh sách thông tin học sinh  
 self.tree = ttk.Treeview(self.root)  
 #Đặt các cột cho cây Treeview với các tên  
 self.tree["columns"] = ("Tên", "Mã SV", "Ngày Sinh", "Status")  
 self.tree.heading("#0", text="STT")  
 self.tree.column("#0", anchor="center", width=50)  
 self.tree.heading("Tên", text="Tên")  
 self.tree.heading("Mã SV", text="Mã SV")  
 self.tree.heading("Ngày Sinh", text="Ngày Sinh")  
 self.tree.heading("Status", text="Status")  
  
 # Điều chỉnh độ rộng của từng cột  
 self.tree.column(  
 "Tên", width=150  
 ) # Điều chỉnh kích thước cột "Tên" cho phù hợp  
 self.tree.column(  
 "Mã SV", width=100  
 ) # Điều chỉnh kích thước cột "Mã SV" cho phù hợp  
 self.tree.column(  
 "Ngày Sinh", width=150  
 ) # Điều chỉnh kích thước cột "Ngày Sinh" cho phù hợp  
 self.tree.column(  
 "Status", width=100  
 ) # Điều chỉnh kích thước cột "Status" cho phù hợp  
  
 self.tree.pack(expand=True, fill="both")  
 # Thiết lập tags cho các giá trị "X" và "V"  
 self.tree.tag\_configure("X", background="green")  
 self.tree.tag\_configure("V", background="red")  
  
 # tạo một hộp thoại đăng ký (registration dialog) để cho phép  
 # người dùng nhập thông tin học sinh và thực hiện đăng ký khuôn mặt.  
 def show\_registration\_dialog(self):  
 *"""  
 Tạo 1 hộp thoại con (Toplevel) registration\_dialog trên cửa sổ gốc (self.root).  
 Hộp thoại con này sẽ hiển thị trong cửa sổ chính  
 và là nơi để người dùng nhập thông tin đăng ký.  
 """* registration\_dialog = Toplevel(self.root)  
 registration\_dialog.title("Đăng ký")  
 registration\_dialog.geometry("300x400")  
 registration\_dialog.configure(bg="lightgrey")  
  
 name\_label = tk.Label(  
 registration\_dialog, text="Họ và tên:", font=("Arial", 12), bg="lightgrey"  
 )  
 name\_label.pack(pady=10)  
 name\_entry = tk.Entry(registration\_dialog, font=("Arial", 12))  
 name\_entry.pack(pady=5)  
  
 student\_id\_label = tk.Label(  
 registration\_dialog,  
 text="Mã sinh viên:",  
 font=("Arial", 12),  
 bg="lightgrey",  
 )  
 student\_id\_label.pack(pady=10)  
 student\_id\_entry = tk.Entry(registration\_dialog, font=("Arial", 12))  
 student\_id\_entry.pack(pady=5)  
  
 dob\_label = tk.Label(  
 registration\_dialog,  
 text="Ngày tháng năm sinh (dd/mm/yyyy):",  
 font=("Arial", 12),  
 bg="lightgrey",  
 )  
 dob\_label.pack(pady=10)  
  
 #Tạo một ô nhập liệu (Entry)  
 dob\_entry = tk.Entry(registration\_dialog, font=("Arial", 12))  
 dob\_entry.pack(pady=5)  
  
 class\_label = tk.Label(  
 registration\_dialog, text="Lớp:", font=("Arial", 12), bg="lightgrey"  
 )  
 class\_label.pack(pady=10)  
 class\_entry = tk.Entry(registration\_dialog, font=("Arial", 12))  
 class\_entry.pack(pady=5)  
  
 submit\_button = tk.Button(  
 registration\_dialog,  
 text="Đăng ký",  
 command=lambda: self.register\_student(  
 name\_entry.get(),  
 student\_id\_entry.get(),  
 dob\_entry.get(),  
 class\_entry.get(),  
 registration\_dialog,  
 ),  
 font=("Arial", 12),  
 bg="lightblue",  
 fg="black",  
 padx=10,  
 pady=5,  
 )  
 submit\_button.pack(pady=10)  
  
  
 """  
 được gọi khi người dùng nhấn nút "Đăng ký" trong hộp thoại đăng ký.  
 Phương thức này thực hiện các thao tác để đăng ký một học sinh và   
 thu thập dữ liệu khuôn mặt của họ  
 """  
 def register\_student(self, name, student\_id, dob, class\_name, registration\_dialog):  
 if name and student\_id and dob and class\_name:  
 self.face\_register.collect\_data()  
 self.face\_register.train\_model()  
 self.face\_register.save\_data(name, student\_id, dob, class\_name)  
 registration\_dialog.destroy() # đóng đối tượng  
  
  
 #tạo cột "Status" trong tệp Excel và điền "V" vào  
 # tất cả các ô trong cột "Status" cho danh sách học sinh  
 def create\_status\_column(self, column\_name):  
 wb = openpyxl.load\_workbook(self.excel\_path) # mở và tải một tệp Excel  
  
 #Đối tượng này là trang tính (sheet) hoạt động (active) trong tệp Excel  
 ws = wb.active  
 column\_exists = False  
   
 # kiểm tra xem cột "Status" đã tồn tại trong danh sách cột của trang tính (ws)  
 for cell in ws[1]:  
 if cell.value == column\_name:  
 column\_exists = True  
 break  
 if not column\_exists:  
 ws.cell(row=1, column=ws.max\_column + 1, value=column\_name)  
 # Ghi "V" vào tất cả các ô trong cột "Status"  
 for row in ws.iter\_rows(  
 min\_row=2,  
 max\_row=ws.max\_row,  
 min\_col=ws.max\_column,  
 max\_col=ws.max\_column,  
 ):  
 for cell in row:  
 cell.value = "V"  
 wb.save(self.excel\_path)  
  
 def attendance(self):  
 if (  
 self.attendance\_process is None  
 or not self.attendance\_process.poll() is None  
 ):  
 now = datetime.now()  
 column\_name = now.strftime("%Y-%m-d %H")  
 self.create\_status\_column(column\_name)  
 self.update\_table\_periodically()  
 if self.face\_attendance:  
 self.face\_attendance.video.release() # Giải phóng camera  
 self.face\_attendance = FaceRecognition() # Khởi tạo lại FaceRecognition  
 attendance\_thread = threading.Thread(target=self.face\_attendance.run)  
 attendance\_thread.start()  
  
 def update\_table\_periodically(self):  
 self.show\_table()  
 self.root.after(1000, self.update\_table\_periodically) # Lặp lại sau 3 giây  
  
 def show\_table(self):  
 self.tree.delete(\*self.tree.get\_children())  
   
 # Xóa tệp sao lưu hiện tại (nếu có)  
 if os.path.exists(self.backup\_excel\_path):  
 os.remove(self.backup\_excel\_path)  
  
 # Sao lưu tệp Excel gốc  
 shutil.copyfile(self.excel\_path, self.backup\_excel\_path)  
  
 # Mở tệp sao lưu để đọc dữ liệu  
 wb = openpyxl.load\_workbook(self.backup\_excel\_path)  
 ws = wb.active  
  
 row\_number = 1 # Bắt đầu từ số thứ tự 1  
  
 for row in ws.iter\_rows(min\_row=2, values\_only=True):  
 if len(row) >= 5:  
 name, maSV, date, status = row[1], row[2], row[3], row[-1]  
  
 # Thiết lập tag cho hàng dựa trên giá trị "Status"  
 if status == "X":  
 tags = ("X",)  
 else:  
 tags = ("V",)  
  
 self.tree.insert(  
 "",  
 "end",  
 text=row\_number,  
 values=(name, maSV, date, status),  
 tags=tags,  
 )  
 row\_number += 1  
  
 # Đảm bảo rằng bạn đã đóng tệp sao lưu  
 wb.close()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 app = FaceRecognitionApp(root)  
 root.mainloop()

1. register.py

from tkinter import messagebox  
import cv2  
import os  
import numpy as np  
from PIL import Image  
import openpyxl  
  
# Lớp chức năng liên quan đến việc đăng ký khuôn mặt  
# và quản lý dữ liệu của học sinh  
class FaceRegister:  
 def \_\_init\_\_(self, data\_path, recognizer\_path, excel\_path):  
 #tạo một đối tượng mô hình nhận diện khuôn mặt sử dụng thuật toán LBPH  
 self.recognizer = cv2.face\_LBPHFaceRecognizer.create()  
  
 #Đường dẫn tới thư mục lưu trữ dữ liệu khuôn mặt của học sinh.  
 self.data\_path = data\_path  
  
 #Đường dẫn tới tệp lưu trữ mô hình nhận diện khuôn mặt.  
 self.recognizer\_path = recognizer\_path  
  
 #Đường dẫn tới tệp Excel lưu trữ thông tin về học sinh.  
 self.excel\_path = excel\_path  
  
 """  
 Phương thức này được sử dụng để thu thập dữ liệu khuôn mặt của học   
 sinh thông qua camera bằng cách chụp ảnh khuôn mặt và lưu vào thư mục   
 """  
 def collect\_data(self):  
 *"""  
 Trước khi thu thập dữ liệu, phương thức gọi phương thức  
 get\_next\_id để xác định ID của học sinh tiếp theo  
 """* user\_id = self.get\_next\_id()  
 video = cv2.VideoCapture(0)  
  
 """  
 Sử dụng CascadeClassifier của OpenCV để tạo một bộ phát hiện khuôn mặt  
 dựa trên tệp mẫu được chỉ định ("haarcascade\_frontalface\_default.xml").  
 """  
 facedetect = cv2.CascadeClassifier\  
 ("haarcascade\_frontalface\_default.xml")  
 count = 0  
  
 while True:  
 """  
 đọc một khung (frame) từ camera và lưu nó vào biến frame.   
 Biến ret chứa giá trị True nếu việc đọc frame thành công.  
 """  
 ret, frame = video.read()  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 faces = facedetect.detectMultiScale(gray, 1.3, 5)  
  
 for x, y, w, h in faces:  
 count += 1  
 cv2.imwrite(  
 os.path.join(self.data\_path,  
 f"User.{user\_id}.{count}.jpg"),  
 gray[y : y + h, x : x + w],  
 )  
 cv2.rectangle(frame, (x, y), (x + w, y + h),  
 (50, 50, 255), 1)  
  
 """  
 Hiển thị frame đã được ghi lại với các hình chữ nhật xung   
 quanh khuôn mặt và kết quả thu thập trực tiếp trên màn hình.  
 """  
 cv2.imshow("Frame", frame)  
  
 k = cv2.waitKey(1)  
 if k == ord("q"):  
 break  
 if count > 200:  
 break  
  
 video.release()  
 cv2.destroyAllWindows()  
 messagebox.showinfo("Thông báo", "Thu thập dữ liệu thành công.")  
  
 """  
 Phương thức này được sử dụng để đào tạo mô hình nhận  
 diện khuônmặt từ các dữ liệu đã thu thập.   
 """  
 def train\_model(self):  
 *"""  
 Tạo một danh sách imagePaths chứa đường  
 dẫn đến các tệp ảnh khuôn mặt trong thư mục data\_path.* ***:return****:  
 """* imagePaths = [  
 os.path.join(self.data\_path, f)  
 for f in os.listdir(self.data\_path)  
 ]  
  
 # lưu trữ các ảnh khuôn mặt và ID tương ứng.  
 faces = []  
 IDs = []  
  
 for imagePath in imagePaths:  
 #Mở tệp ảnh, chuyển đổi nó thành ảnh xám (grayscale)  
 faceImg = Image.open(imagePath).convert("L")  
  
 #chuyển đổi thành mảng numpy (faceImg, faceNp).  
 faceNp = np.array(faceImg, "uint8")  
  
 #Trích xuất ID của người dùng từ tên tệp ảnh  
 # và thêm nó vào danh sách IDs.  
 #Thêm mảng numpy của ảnh khuôn mặt vào danh sách faces.  
 user\_id = int(os.path.split(imagePath)[-1].split(".")[1])  
 faces.append(faceNp)  
 IDs.append(user\_id)  
  
 if len(IDs) > 0:  
 messagebox.showinfo("Thông báo", "Processing...")  
 self.recognizer.train(faces, np.array(IDs))  
 self.recognizer.save(self.recognizer\_path)

messagebox.showinfo("Thông báo", "Huấn luyện hoàn tất.")  
 else:  
 messagebox.showerror("Lỗi", "Không tìm thấy dữ liệu huấn luyện.")  
  
 """  
 Phương thức này được sử dụng để lưu thông tin của học sinh   
 (tên, mã số sinh viên, ngày tháng năm sinh, lớp) vào tệp Excel.   
 """  
 def save\_data(self, name, student\_id, dob, class\_name):  
 next\_id = self.get\_next\_id()  
  
 #Mở tệp Excel (self.excel\_path)  
 # và lấy trang tính (worksheet) hoạt động (active).  
 wb = openpyxl.load\_workbook(self.excel\_path)  
 ws = wb.active  
   
 ws.cell(row=next\_id, column=1, value=next\_id)   
 ws.cell(row=next\_id, column=2, value=name)   
 ws.cell(row=next\_id, column=3, value=student\_id)   
 ws.cell(row=next\_id, column=4, value=dob)   
 ws.cell(row=next\_id, column=5, value=class\_name)   
 wb.save(self.excel\_path)  
  
 def get\_next\_id(self):  
 wb = openpyxl.load\_workbook(self.excel\_path)  
 ws = wb.active  
   
 """  
 Trả về số hàng (rows) hiện có trên trang tính,   
 tương đương với ID tiếp theo. ID này sẽ được   
 sử dụng để đặt tên cho tệp ảnh khuôn mặt của học sinh.  
 """  
 return ws.max\_row + 1

1. attendence.py

import time  
  
import cv2  
import openpyxl  
import os  
import shutil

"""  
Lớp FaceRecognition chịu trách nhiệm cho việc nhận diện khuôn   
mặt từ camera và cập nhật trạng thái điểm danh trong tệp Excel.  
"""  
class FaceRecognition:  
 def \_\_init\_\_(self, face\_detect\_path, excel\_path):  
 # sử dụng để truy cập video từ máy ảnh.  
 # Giá trị 0 truyền vào đây đại diện cho máy ảnh mặc định của máy tính.  
 self.video = cv2.VideoCapture(0)  
  
 #Đường dẫn đến thư mục hiện tại của tệp mã nguồn Python đang chạy  
 self.path = os.path.dirname(\_\_file\_\_)  
  
 # tải tệp XML chứa thông tin về việc nhận diện khuôn mặt  
 self.facedetect = cv2.CascadeClassifier(face\_detect\_path)  
 self.excel\_path = excel\_path  
  
 #Local Binary Pattern Histograms  
 self.recognizer = cv2.face\_LBPHFaceRecognizer.create()  
  
 """  
 sử dụng để theo dõi trạng thái của việc tải mô hình nhận diện khuôn   
 mặt từ tệp "trainer.yml". Ban đầu, nó được đặt là True.  
 """  
 self.trainer\_status = True  
  
 #Đường dẫn tới tệp "trainer.yml" trong thư mục hiện tại.  
 trainer\_file = os.path.join(self.path, "trainer.yml")  
 try:  
 # nạp vào mô hình nhận diện khuôn mặt  
 self.recognizer.read(trainer\_file)  
 except:  
 print(  
 "Cảnh báo: Tệp 'trainer.yml' không tồn tại, trống hoặc không hợp lệ. Hãy đảm bảo bạn đã đào tạo mô hình trước."  
 )  
 self.trainer\_status = False  
 self.student\_ids = self.read\_ids\_from\_excel() # đọc thông tin từ tệp excel  
  
 def read\_ids\_from\_excel(self):  
 student\_ids = {}  
 try:  
 """  
 Dùng thư viện OpenPyXL để nạp tệp Excel "data1.xlsx" trong thư mục hiện   
 tại của ứng dụng. Đường dẫn tới tệp này được xây dựng bằng cách kết hợp   
 self.path (đường dẫn đến thư mục hiện tại của ứng dụng) và tên tệp " data1.xlsx".  
 """  
 workbook = openpyxl.load\_workbook(os.path.join(self.path, "data1.xlsx"))  
  
 #Lấy sheet hoạt động (active sheet) từ tệp Excel.  
 # Đây là nơi chứa dữ liệu sinh viên.  
 sheet = workbook.active  
  
 # Sử dụng vòng lặp để duyệt qua từng dòng (sinh viên) trong tệp Excel.  
 for row in sheet.iter\_rows(min\_row=2, values\_only=True):  
 if len(row) >= 3:  
 id, name, student\_id = int(row[0]), row[1], row[2]  
 student\_ids[id] = student\_id  
 except Exception as e:  
 print(f"An error occurred while reading the Excel file: {e}")  
  
 return student\_ids  
  
 def run(self):  
  
 *"""  
 Bắt đầu vòng lặp vô hạn, cho phép chương trình chạy  
 liên tục để theo dõi và nhận diện khuôn mặt từ máy ảnh.  
 """* while True:  
 self.create\_backup() #tạo một bản sao lưu của tệp Excel  
  
 #Đọc khung hình mới từ máy ảnh và lưu nó trong biến frame.  
 # Giá trị ret cho biết liệu việc đọc khung hình đã thành công hay không.  
 ret, frame = self.video.read()  
 if not ret:  
 print("Không thể đọc khung hình từ camera.")  
 break  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
  
 #Sử dụng phân loại viên cascade đã được nạp  
 # (self.facedetect) để nhận diện khuôn mặt trong khung hình  
 faces = self.facedetect.detectMultiScale(gray, 1.3, 5)  
 for x, y, w, h in faces:  
 # serial là id dự đoán  
 # confident là độ tự tin của đự đoán  
 serial, confident = self.recognizer.predict(gray[y : y + h, x : x + w])  
  
 if confident < 50:  
 id = self.student\_ids.get(serial)  
  
 # hiển thị tên sinh viên trên khung hình.  
 cv2.putText(  
 frame,  
 f"{str(id).upper()}",  
 (x - 100, y + h - 300),  
 cv2.FONT\_HERSHEY\_COMPLEX,  
 1,  
 (0, 255, 0),  
 2,  
 )  
 #vẽ khung xung quanh khuôn mặt  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
  
 #cập nhật trạng thái điểm danh của sinh viên.  
 self.update\_status\_in\_excel(serial)  
  
 else:  
 cv2.putText(  
 frame,  
 "Unknown",  
 (x, y + h - 300),  
 cv2.FONT\_HERSHEY\_COMPLEX,  
 1,  
 (0, 0, 255),  
 2,  
 )  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)  
  
 # Hiển thị khung hình với kết quả nhận diện khuôn mặt lên màn hình.  
 cv2.imshow("Face Recognition", frame)  
 if cv2.waitKey(1) & 0xFF == ord("q"):  
 break

self.video.release()  
 cv2.destroyAllWindows()

def update\_status\_in\_excel(self, student\_id):  
 try:  
 #Sử dụng thư viện OpenPyXL để nạp tệp Excel  
 # (self.excel\_path) vào biến wb.  
 wb = openpyxl.load\_workbook(self.excel\_path)  
 except:  
 wb = openpyxl.load\_workbook(self.excel\_path)  
  
 # Lấy sheet hoạt động (active sheet) từ tệp Excel đã được nạp.  
 ws = wb.active  
 found = False  
  
 # duyệt qua từ dòng thứ 2 trong sheet , chỉ đọc giá trị (cột) của các ô  
 for row in ws.iter\_rows(min\_row=2, values\_only=True):  
 """  
 Kiểm tra xem giá trị trong cột đầu tiên của dòng hiện tại có bằng   
 student\_id hay không. Nếu bằng, đó là sinh viên cần cập nhật.  
 """  
 if row[0] == student\_id:  
 found = True  
 #Lấy chỉ mục (số dòng) của dòng hiện tại trong tệp Excel.  
 row\_index = ws.\_current\_row  
 if row[-1] == "V":  
 # cập nhật trạng thái điểm danh bằng "X" trong cột cuối cùng.  
 ws.cell(row=row\_index, column=ws.max\_column, value="X")  
 if found:  
 wb.save(self.excel\_path)  
 else:  
 print(f"Không tìm thấy học sinh với student\_id: {student\_id}")  
  
 # Phương thức này được sử dụng để tạo bản sao lưu của tệp Excel.  
 def create\_backup(self):  
 if os.path.exists(self.excel\_path):  
 backup\_file = os.path.join(self.path, "backup\_data1.xlsx")  
   
 #Sử dụng thư viện shutil để sao lưu tệp Excel.   
 shutil.copyfile(self.excel\_path, backup\_file)

# IV.TỔNG KẾT

# Kết Quả

Dự án đã hoàn thành và đạt được các mục tiêu đề ra. Hệ thống nhận diện khuôn mặt có khả năng đăng ký và điểm danh sinh viên. Dữ liệu cá nhân và kết quả điểm danh được lưu trữ một cách hiệu quả trong tệp Excel.

## Học Hỏi và Hướng Phát Triển

Trong quá trình thực hiện dự án, chúng tôi đã học được cách sử dụng các thư viện Python để xây dựng hệ thống nhận diện khuôn mặt. Tuy nhiên, dự án còn có thể được phát triển bằng cách:

+ Cải thiện khả năng nhận diện và độ chính xác.

+ Tích hợp với các thiết bị thực tế như máy chấm công.

+ Tích hợp tính năng bảo mật và xác thực cao cấp.

1. Tài liệu tham khảo để hoàn thành dự án
2. Kết Luận

Dự án đã đạt được các mục tiêu và mang lại kiến thức thực tế về nhận diện khuôn mặt sử dụng Python. Đây là một cơ sở tốt để phát triển các ứng dụng liên quan đến quản lý điểm danh và an ninh.