**ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO ĐỒ ÁN**

**LẬP TRÌNH HỆ THỐNG VÀ VI ĐIỀU KHIỂN**

**ĐỀ TÀI:**

**HỆ THỐNG QUẢN LÝ NHÂN VIÊN BẰNG CÔNG NGHỆ NHẬN DẠNG KHUÔN MẶT**

GIẢNG VIÊN HƯỚNG DẪN: Ninh Khánh Duy

GIẢNG VIÊN CHẤM: Bùi Thị Thanh Thanh

|  |  |  |  |
| --- | --- | --- | --- |
| STT NHÓM: 44  HỌ VÀ TÊN SINH VIÊN | LỚP HỌC PHẦN ĐỒ ÁN | ĐIỂM QUÁ TRÌNH (do GVHD ghi) | ĐIỂM BẢO VỆ (do GV chấm ghi) |
| Nguyễn Vĩnh Đạt | **17N11A** |  |  |
| Hoàng Trọng Kiên | **17N11A** |  |  |
| Trần Minh Quang | **17N12B** |  |  |
| Phan Gia Sang | **17N11A** |  |  |

**ĐÀ NẴNG, 06/2020**

**TÓM TẮT ĐỒ ÁN**

“Hệ thống quản lý nhân viên bằng công nghệ nhận dạng khuôn mặt” nhằm giải quyết vấn đề là: Giúp người người quản lý của các tổ chức đơn vị sử dụng người lao động muốn quản lý thời gian đi làm của các nhân viên. Phương pháp giải quyết là sử dụng công nghệ nhận diện khuôn mặt với thư viện OpenCV để nhận dạng nhân viên sau đó được sử lý bằng Raspberry và gửi số liệu vào cơ sở dữ liệu. Qua đó, người quản lý sẽ nắm bắt được thông tin thời gian của từng nhân viên. Sau khi nghiên cứu và thực hiện, nhóm đã thực hiện được các chức năng cơ bản: Xem thông tin thời gian điểm danh của từng nhân viên, kết xuất thông tin tiền lương của mỗi nhân viên sau khi đã trừ tiền phạt khi nhân viên đi muộn hoặc nghỉ làm việc.

**LỜI NÓI ĐẦU**

Ngày nay, ngành khoa học kĩ thuật ngày càng phát triển, cùng với đó là nhu cầu đòi hỏi của con người ngày một nâng cao trong nhiều lĩnh vực của đời sống xã hội. Trong đó có nhu cầu về một cuộc sống tiện nghi, thông minh. Hiện tại, công nghệ thông tin được người ta quan tâm và nhắc đến nhiều hơn bao giờ hết vì nó là một phần không thể thiếu trong cuộc sống văn minh, tri thức. Máy vi tính, các thiết bị di động cùng với những phần mềm là công cụ đắc lực giúp ta quản lý, tổ chức, sắp xếp, điều khiển và xử lý công việc một cách nhanh chóng và chính xác. Là một sinh viên ngành Công nghệ thông tin, chúng em có cơ hội được nghiên cứu sâu hơn và đi vào các bước tạo ra sản phẩm thông qua **Đồ án Lập trình hệ thống và Vi điều khiển.**

Quản lý nhân viên bằng công nghệ nhận diện khuôn mặt là một đề tài còn khá mới lạ. Hầu hết các máy chấm công trên thị trường đều sử dụng công nghệ nhận diện vân tay. Vì vậy nhóm chúng em thực hiện đề tài này để nắm rõ được phương thức hoạt động cũng như cách vận hành của hệ thống điểm danh bằng công nghệ nhận dạng khuôn mặt.

Thông qua quá trình thực hiện đồ án, chúng em đã nắm bắt được những kỹ thuật quan trọng của việc xây dựng mã nguồn, lắp ráp và hoàn thiện sản phẩm. Chúng em xin chân thành cảm ơn thầy Ninh Khánh Duy đã bổ sung ý tưởng và tận tình giúp đỡ chúng em thực hiện đồ án này. Trong quá trình thực hiện, không tránh khỏi sai xót. Chúng em rất mong nhận được sự góp ý từ phía thầy cô để sản phẩm của chúng em được hoàn thiện hơn.

**MỤC LỤC**

[I. GIỚI THIỆU: 4](#_Toc42704237)

[II. GIẢI PHÁP : 6](#_Toc42704238)

[1. Mô tả hệ thống : 6](#_Toc42704239)

[2. Phát hiện khuôn mặt sử dụng đặc trưng Haar-like : 7](#_Toc42820226)

2.1. Các đặc trưng Haar-like : 7

2.2. Áp dụng đặc trưng Haar-like để phát hiện khuôn mặt trong ảnh : 8

2.3. Ảnh tích hợp (Integral image) : 9

3. Thuật toán Adaboost (Adaptive Boost): 10

[4. Mô hình phân lớp phân tầng (Cascade classifier) :](#_Toc42704241) 11

[5. Phát hiện khuôn mặt sử dụng thư viện OpenCV :](#_Toc42704241) 12

[6. Nhận dạng khuôn mặt :](#_Toc42704241) 13

6.1. Các tham số của thuật toán : 14

6.2. **Xây dựng mô hình khuôn mặt một người dùng vecto đặc trưng** : 15

6.3. **Trích xuất đặc trưng khuôn mặt áp dụng LBPH** : 15

6.4. Trích xuất biểu đồ Histogram : 16

6.5. Thực hiện nhận dạng khuôn mặt dùng công thức Euclid : 18

[7. Giới thiệu giao diện web :](#_Toc42704241) 20

7.1. Giao diện web : 20

7.2. Giao diện THÔNG TIN NHÂN VIÊN : 20

7.3. Giao diện ĐIỂM DANH : 21

7.4. Giao diện CHI TIẾT ĐIỂM DANH : 21

7.5. Chức năng tìm kiếm : 22

7.6. Chức năng thêm nhân viên : 23

7.7. Chức năng sửa thông tin nhân viên : 24

[8. Giới thiệu về Database :](#_Toc42704241) 24

8.1. Giao diện quản lý database : 24

8.2. Bảng nhanvien : 25

8.3. Bảng diemdanh : 25

8.4. Bảng datetime : 26

[9. Giới thiệu một số phần cứng, linh kiện sử dụng :](#_Toc42704241) 27

[III. KẾT QUẢ :](#_Toc42704250) 31

[1. Mô tả dữ liệu kiểm thử :](#_Toc42704241) 31

[2. Quy trình kiểm thử :](#_Toc42704241) 31

IV. KẾT LUẬN : 35

1. Ưu điểm : 35

2. Nhược điểm : 35

3. Kết luận : 35

4. Hướng phát triển : 35

[TÀI LIỆU THAM KHẢO 3](#_Toc42704251)6

1. GIỚI THIỆU:

Tên đề tài : Hệ thống quản lý nhân viên bằng công nghệ nhận dạng khuôn mặt.

* Đặt vấn đề :

Hiện nay trên thị trường đang sử dụng các loại máy chấm công ứng dụng công nghệ nhận diện bằng vân tay. Rất ít các loại máy chấm công bằng công nghệ nhận dạng khuôn mặt. Hệ thống điểm danh bằng công nghệ nhận dạng khuôn mặt được ứng dụng đáp ứng được nhu cầu sử dụng hằng ngày trong mỗi cơ quan, doanh nghiệp, giúp tiết kiệm công sức thay vì cần phải có nhân viên đi chấm công một cách thủ công.

* Các vấn đề cần giải quyết :

Hệ thống nhận dạng khuôn mặt hoạt động theo thời gian thực nên đòi hỏi tốc độ xử lý và độ chính xác cao. Thuật toán nhận dạng khuôn mặt cần được tối ưu nhất để giảm thời gian nhận dạng của mỗi nhân viên. Ngoài ra, hệ thống cần phải nhận dạng chính xác khuôn mặt cần nhận dạng, hạn chế tối đa sự nhầm lẫn.

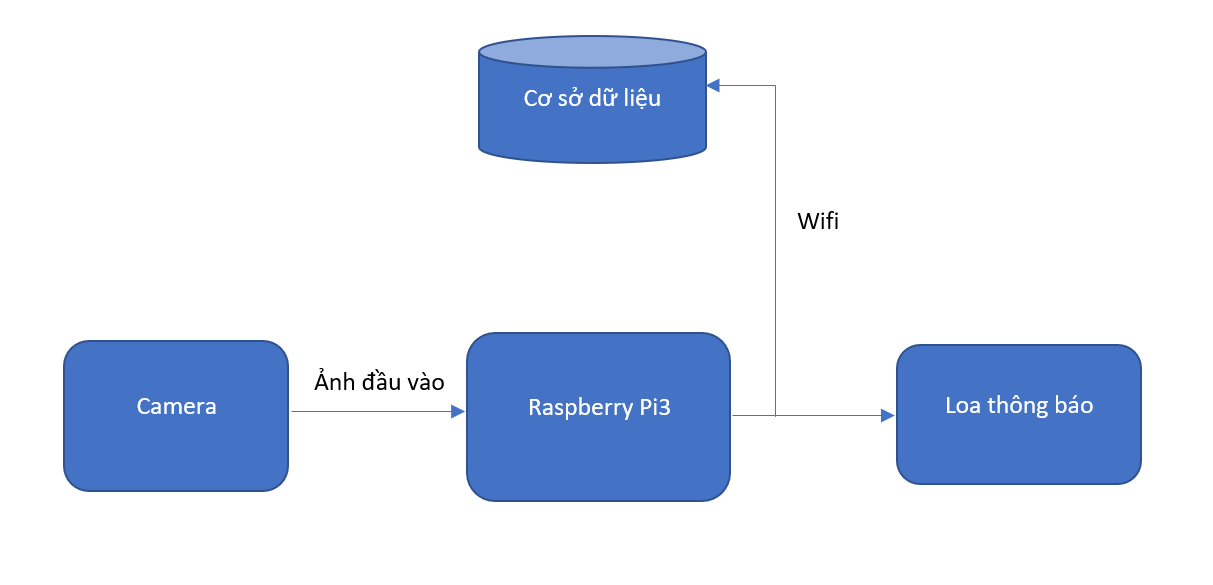
* Giải pháp thực hiện :

Trong phạm vi đồ án môn học này, chúng em thiết kế hệ thống chỉ có tính ổn định và chính xác tương đối. Tuy nhiên, vẫn đảm bảo được các chức năng cơ bản cho người dùng với cách sử dụng đơn giản. Giải pháp được đưa ra là: Nhân viên mới đăng ký khuôn mặt với bộ phận quản lý. Sau đó khi muốn điểm danh thì nhân viên chỉ cần đưa mặt tới camera của hệ thống sau đó bấm nút để bắt đầu quá trình nhận dạng, hệ thống sẽ tiến hành nhận dạng khuôn mặt và lưu trữ dữ liệu điểm danh.

**Bảng 1. BẢNG PHÂN CÔNG NHIỆM VỤ**

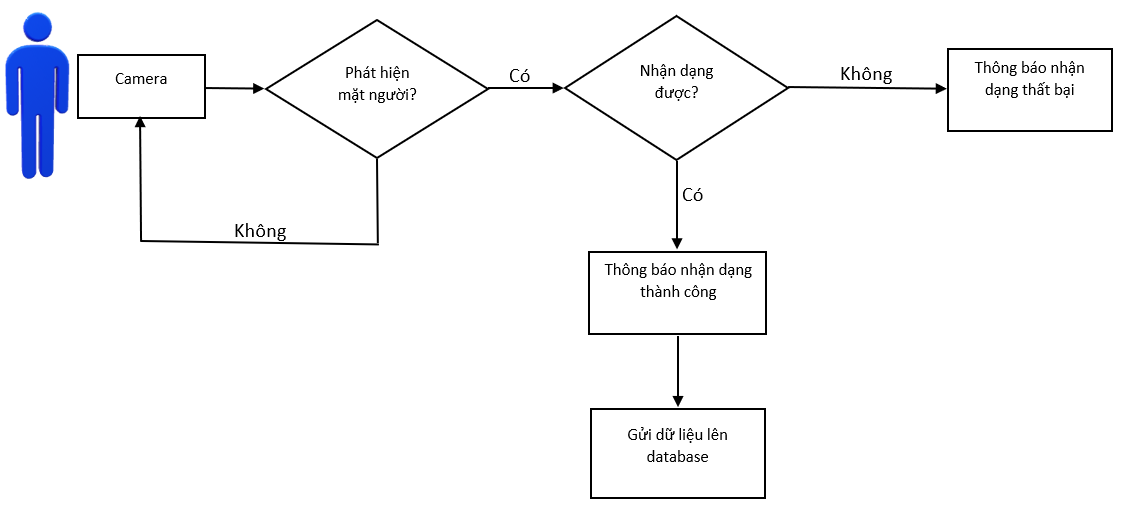
|  |  |  |
| --- | --- | --- |
| STT | Nội dung công việc | SV thực hiện |
| 1 | Thảo luận ý tưởng, lên kế hoạch chi tiết | Nguyễn Vĩnh Đạt  Hoàng Trọng Kiên  Trần Minh Quang  Phan Gia Sang |
| 2 | Tìm hiểu lý thuyết về Raspberry Pi 3, thư viện OpenCV, phần cứng | Nguyễn Vĩnh Đạt  Hoàng Trọng Kiên  Trần Minh Quang  Phan Gia Sang |
| 3 | Tìm hiểu về thuật toán phát hiện khuôn mặt sử dụng thư viện OpenCV | Nguyễn Vĩnh Đạt |
| 4 | Tìm hiểu về thuật toán nhận dạng khuôn mặt sử dụng thư viện OpenCV | Trần Minh Quang |
| 5 | Lập trình hàm tính khoảng cách Euclid để xác định khuôn mặt nhận dạng | Trần Minh Quang  Nguyễn Vĩnh Đạt |
| 6 | Lập trình server và thiết kế cơ sở dữ liệu | Hoàng Trọng Kiên |
| 7 | Thiết kế và lập trình giao diện web | Phan Gia Sang |
| 8 | Thiết kế, lập trình nút bấm, loa và hoàn thiện mô hình | Nguyễn Vĩnh Đạt  Trần Minh Quang |
| 9 | Viết báo cáo, slide thuyết trình. | Phan Gia Sang  Hoàng Trọng Kiên |

1. GIẢI PHÁP :
2. Mô tả hệ thống :



Hình 1. Sơ đồ tổng quát hệ thống

Hệ thống gồm một camera thu hình ảnh đầu vào của người cần nhận dạng, Raspberry Pi3 để xử lý nhận dạng và gửi dữ liệu đến cơ sở dữ liệu, sau khi tiến hành nhận dạng thì Raspberry sẽ gửi tín hiệu đến loa để thông báo hệ thống có nhận dạng ra được người cần nhận dạng hay không.



Hình 2. Quy trình nhận dạng

1. **Phát hiện khuôn mặt sử dụng đặc trưng Haar-like :**

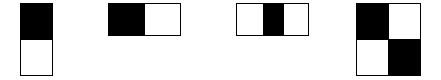
Vùng khuôn mặt trên ảnh là tập hợp các điểm ảnh có khác biệt so với các vùng ảnh khác, tạo nên các đặc trưng riêng của khuôn mặt. Tất cả khuôn mặt người đều có chung những đặc điểm sau khi đã chuyển qua ảnh xám, ví dụ:

* Vùng hai mắt sẽ tối hơn vùng má và vùng trán
* Vùng giữa sống mũi cũng tối hơn vùng hai bên mũi.

Và còn rất nhiều những đặc điểm khác của khuôn mặt và các đặc trưng Haar like dựa vào các đặc điểm này để nhận dạng.

* 1. **Các đặc trưng Haar-like:**

Mỗi đặc trưng Haar-like là một miền hình chữ nhật được chia thành 2,3 hoặc 4 hình chữ nhật nhỏ phân biệt quy ước bằng màu trắng và đen.



Hình 3: Các đặc trưng Haar-like [8]

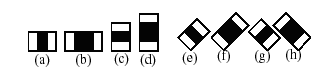
Như hình vẽ trên [8] là 3 loại đặc trưng Haar-like cơ bản gồm 2,3 và 4 hình chữ nhật hợp thành. Từ những đặc trưng cơ bản mở rộng ra thành tập các đặc trưng:

* + Đặc trưng cạnh [8]:



Hình 4: Đặc trưng cạnh [8]

* + Đặc trưng đường [8]:



Hình 5: Đặc trưng đường [8]

* + Đặc trưng tâm – xung quanh [8]:



Hình 6: Đặc trưng tâm - xung quanh [8]

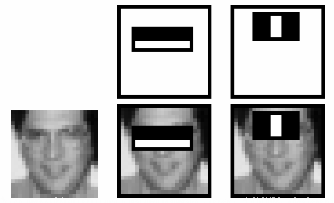
Giá trị của một đặc trưng Haar-like là sự khác biệt giữa tổng các giá trị xám của các pixel trong vùng “đen” với tổng các giá trị xám của các pixel trong vùng “trắng”:

*f(x) = Tổngvùng đen(pixel) - Tổngvùng trắng(pixel)*

Vậy khi được đặt lên một vùng ảnh, đặc trưng Haar-like sẽ tính toán và đưa ra giá trị đặc trưng *h(x)* của vùng ảnh đó.

* 1. **Áp dụng đặc trưng Haar-like để phát hiện khuôn mặt trong ảnh :**

Để phát hiện khuôn mặt, hệ thống sẽ cho một cửa sổ con quét lên toàn bộ ảnh đầu vào. Như vậy sẽ có rất nhiều ảnh con ứng với từng cửa sổ con, các đặc trưng Haar-like sẽ được đặc lên các cửa sổ con này để từ đó tính ra giá trị của đặc trưng. Sau đó các giá trị này được bộ phân loại xác nhận xem khung hình đó có phải khuôn mặt hay không.



Hình 7. Áp dụng đặc trưng Haar-like lên từng khung của ảnh [9]

Ứng với mỗi một đặc trưng như trên, một bộ phân lớp yếu xác định khi giá trị của đặc trưng Haar-like tại cửa sổ con vượt qua một ngưỡng đã xác định thì bộ phân lớp sẽ kết luận cửa sổ con là khuôn mặt, cònkhông vượt qua ngưỡng đó thì không là khuôn mặt.

* 1. **Ảnh tích hợp (Integral Image) :**

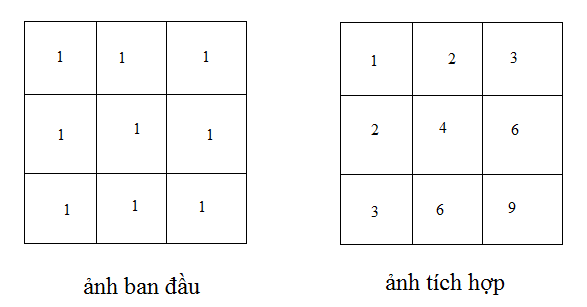
Giá trị của mỗi một đặc trưng Haar-like được tính bằng tổng giá trị các điểm ảnh vùng đen trừ đi tổng các điểm ảnh vùng trắng, như vậy là với số lượng lớn các đặc trưng sẽ tạo ra một khối lượng tính toán khổng lồ. Điều này là không phù hợp để đáp ứng thời gian thực vì thời gian xử lý rất lâu. Vì vậy ảnh tích hợp được đưa ra nhằm tính toán nhanh chóng các đặc trưng, giảm thời gian xử lý.

Ảnh tích hợp được định nghĩa theo công thức:



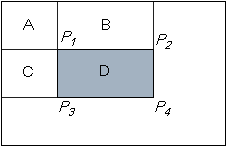
Giá trị của ảnh tích hợp tại vị trí (x,y) là tổng các điểm ảnh thuộc hình chữ nhật xác định bởi góc trái trên là (0,0) và góc phải dưới (x,y). Sau khi chuyển ảnh cần nhận dạng thành ảnh tích hợp, viêc tính toán giá trị các đặc trưng Haar-like sẽ rất đơn giản.

Ví dụ chuyển một ảnh 33 có giá trị xám như bên dưới thành ảnh tích hợp:



Hình 8. Chuyển ảnh ban đầu về ảnh tích hợp [2]

Sau khi chuyển về ảnh tích hợp, ta tiếp tục tính giá trị đặc trưng Haar-like, ta phải tính được tổng giá trị điểm ảnh trong một vùng hình chữ nhật trên ảnh. Ví dụ như vùng D trong hình vẽ:



Hình 9. Tính giá trị các điểm ảnh tại vùng D [10]

Với : A,B,C,D là tổng giá trị các điểm ảnh trong từng vùng.

P1,P2,P3,P4 là giá trị của ảnh tích hợp tại 4 đỉnh của D.

Nếu như là ảnh xám bình thường thì để tính D ta phải tính tổng tất cả các giá trị điểm ảnh trong D, miền D càng lớn thì số phép cộng càng nhiều. Nhưng với ảnh tích hợp dù miền D có kích thước như thế nào thì D cũng chỉ cần tính thông qua 4 giá trị tại 4 đỉnh.

Ta có: P1 = A, P2 = A + B, P3 = A + C, P4 = A + B + C + D

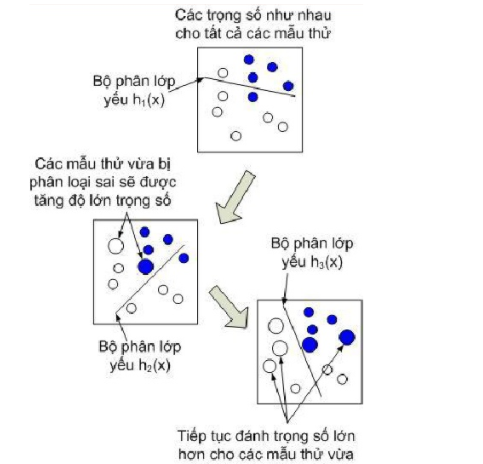
P1+ P4–P2–P3= A + (A + B + C + D) – (A + B) – (A + C) = D

Vậy ta có: **D = P1 + P4 – P2 – P3.**

1. **Thuật toán Adaboost (Adaptive Boost):**

Kỹ thuật Boosting: nguyên tắc cơ bản của Boosting là kết hợp các bộ phân lớp yếu (hay các bộ phân lớp cơ sở) để tạo nên một bộ phân lớp mạnh.

Adaboost là một thuật toán sử dụng kỹ thuật Boosting được đưa ra bởi Freund và Schapire vào 1996. Thuật toán này được sử dụng rất phổ biến cho bài toán phát hiện khuôn mặt. Điểm cải tiến của Adaboost là ta sẽ gán cho mỗi mẫu một trọng số. Ý nghĩa của việc gán trọng số như sau: Ở mỗi vòng lặp của quá trình huấn luyện, khi một bộ phân lớp yếu đã được xây dựng, ta sẽ tiến hành cập nhật trọng số cho các mẫu. Việc cập nhật này được tiến hành như sau: ta sẽ tăng trọng số của các mẫu bị phân lớp sai bởi bộ phân lớp yếu và giảm trọng số của các mẫu được phân lớp đúng. Bằng cách này, ở vòng lặp kế, ta sẽ xây dựng bộ phân lớp yếu tiếp theo theo hướng: tập trung vào các mẫu bị phân lớp sai bởi bộ phân lớp yếu trước đó. Cuối cùng, để có được bộ phân lớp mạnh, ta sẽ kết hợp tuyến tính các bộ phân lớp yếu đã tìm được lại với nhau.

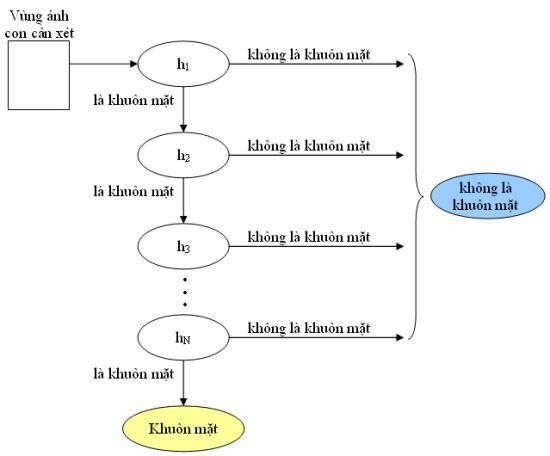


Hình 10: Áp dụng thuật toán Adaboost [8]

1. **Mô hình phân lớp phân tầng (Cascade classifier):**

Cascade classifier là một chuỗi phân lớp trong đó mỗi bộ phân lớp được xây dựng bằng thuật toán Adaboost.

Bộ phân lớp đầu tiên sẽ loại bỏ phần lớn các cửa sổ không phải khuôn mặt và cho đi qua các cửa sổ được cho là khuôn mặt. Ở đây, bộ phân lớp này rất đơn giản nên độ phức tạp tính toán cũng rất thấp. Bởi vì bộ phân lớp đầu tiên khá đơn giản nên trong số các cửa sổ được nhận dạng là khuôn mặt sẽ có một số khung lớn cửa sổ bị nhận dạng sai (không phải là khuôn mặt).



Hình 11: Thuật toán phát hiện khuôn mặt [10]

Những cửa chỗ được cho đi qua bởi bộ phân lớp đầu sẽ được xem xét bởi bộ phân lớp sau đó, nếu bộ phân lớp cho rằng đó không phải là khuôn mặt thì ta loại bỏ, nếu bộ phân lớp cho rằng đó là khuôn mặt thì ta lại cho đi qua và chuyển đến bộ phân lớp phía sau. Chỉ những cửa sổ đi qua được tất cả các bộ phân lớp thì ta mới quyết định đó là khuôn mặt. Sau khi đã xác định tất cả các khung có thể là khuôn mặt thì bước cuối cùng sẽ gộp các khung hình lân cận đó thành một khung mặt chứa khuôn mặt người duy nhất.

1. **Phát hiện khuôn mặt sử dụng thư viện OpenCV :**

Môhình phân tầng nhóm em sử dụng là một bộ phân lớp được xây dựng sẵn trên các bộ cơ sở dữ liệu chuẩn của thư viện OpenCV. Bộ phân lớp này được lưu ở file xml có tên haarcascade\_frontalface\_default.xml

Bên cạnh bộ phân lớp ở trên, một hàm quan trọng phục vụ công việc phát hiện khuôn mặt là hàm **detectMultiScale(image,scale\_factor,min\_neighbors)** thuộc lớp CascadeClassifier (lớp phục vụ tìm kiếm đối tượng của Opencv).

Các tham số đầu vào:

* **image** là ảnh cần phát hiện khuôn mặt.
* **scale\_factor** là tỉ lệ tăng kích thước của khung cửa sổ tìm kiếm. Ví dụ nếu scale\_factor=1.3 thì sau khi quét hết bức ảnh 1 lần, khung cửa sổ sẽ tăng kich thước lên 30% và thực hiện lần quét tiếp theo. Tham số này ảnh hưởng đến tốc độ xử lý và độ tin cậy của chương trình. Nếu để nó quá lớn thì tốc độ chương trình sẽ tăng lên do số lần quét giảm đi, tuy nhiên có thể chương trình có thể bỏ qua không phát hiện được một số khuôn mặt có kích thước nằm giữa 2 khung cửa sổ liên tiếp do độ tăng kích thước của khung là quá lớn. Nếu để nó quá thấp thì ta có thể không bỏ sót bất kì khuôn mặt nào nhưng chương trình sẽ tốn thời gian hơn vì tăng số lần quét lên.
* **min\_neighbors:** Trong quá trình tìm kiếm khuôn mặt chương trình sẽ tìm được nhiều những khung hình chữ nhật chứa khuôn mặt cho dù đó chỉ là một khuôn mặt và có nhưng trường hợp nhận diện nhầm. Do đó **min\_neighbors** giá trị tối thiểusố hình chữ nhật lân cận được gộp lại sau khi quá trình quét đã xong. Ví dụ min\_neighbors=5 thì sau khi quét xong bức ảnh, tối thiểu 5 hình chữ nhật lân cận sẽ được gộp lại thành một khung có chứa khuôn mặt.

1. Nhận dạng khuôn mặt :

Trong bài toán này, nhận diện khuôn mặt sử dụng thuật toán  **Biểu đồ mô hình nhị phân cục bộ (LBPH).** **Mô hình nhị phân cục bộ**(LBP) là một toán tử kết cấu đơn giản nhưng rất hiệu quả, gắn nhãn các pixel của hình ảnh bằng cách chia vùng lân cận của mỗi pixel và coi kết quả là số nhị phân. Sử dụng LBP kết hợp với biểu đồ, chúng ta có thể biểu diễn hình ảnh khuôn mặt bằng một vectơ dữ liệu đơn giản[6]

**Thuật toán bao gồm các bước :**

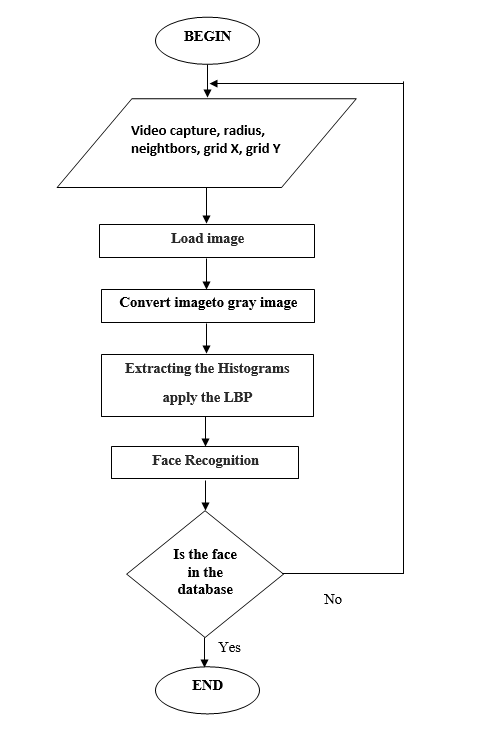
**-Parameters (Tham số)**

**-Training the Algorithm (Huấn luyện thuật toán)**

**-Applying the LBP operation (Áp dụng thao tác LBP)**

**-Extracting the Histograms (Trích xuất biểu đồ)**

**-Performing the face recognition (Thực hiện nhận dạng khuôn mặt)**



****Hình 12: Tổng quát hệ thống nhận diện khuôn mặt****

* 1. ****Các tham số của thuật toán:****

**Thuật toán sử dụng 4 tham số :**

****Radius**** : bán kính được sử dụng để xây dựng mô hình nhị phân cục bộ tròn và biểu thị bán kính xung quanh pixel trung tâm. Nó thường được đặt thành 1.

****Neighbors**** : số lượng điểm mẫu để xây dựng mô hình nhị phân cục bộ tròn. Hãy ghi nhớ: bạn bao gồm càng nhiều điểm mẫu, chi phí tính toán càng cao. Nó thường được đặt thành 8.

****Grid X**** : số lượng ô theo hướng ngang. Càng nhiều ô, lưới càng mịn, thì chiều của vectơ đặc trưng kết quả càng cao. Nó thường được đặt thành 8.

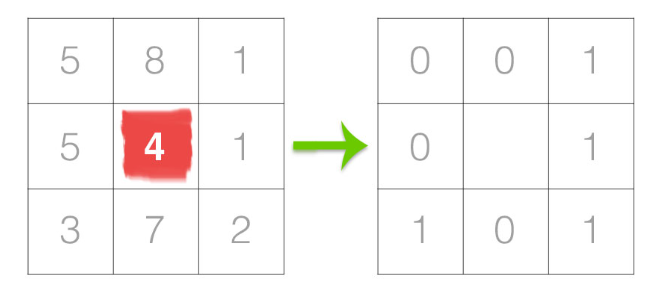
****Grid Y**** : số lượng ô theo hướng dọc. Càng nhiều ô, lưới càng mịn, thì chiều của vectơ đặc trưng kết quả càng cao. Nó thường được đặt thành 8.

* 1. ****Xây dựng mô hình khuôn mặt một người dùng vecto đặc trưng :****

Để xây dựng mô hình, chúng ta cần sử dụng một bộ dữ liệu với hình ảnh khuôn mặt của những người mà chúng ta muốn nhận dạng. Chúng ta cũng cần đặt ID cho mỗi hình ảnh, thuật toán sẽ sử dụng thông tin này để nhận dạng hình ảnh đầu vào và cung cấp cho bạn đầu ra. Hình ảnh của cùng một người phải có cùng ID.[6]

* 1. ****Trích xuất đặc trưng khuôn mặt áp dụng LBPH :****

Bước tính toán đầu tiên của LBPH là tạo ra một hình ảnh trung gian mô tả hình ảnh gốc theo cách tốt hơn, bằng cách làm nổi bật các đặc điểm khuôn mặt.



Hình 13: ****Trích xuất đặc trưng khuôn mặt**** áp dụng LBPH [7]

Dựa trên hình ảnh trên, chia nó thành nhiều bước nhỏ để chúng ta có thể hiểu nó dễ dàng:

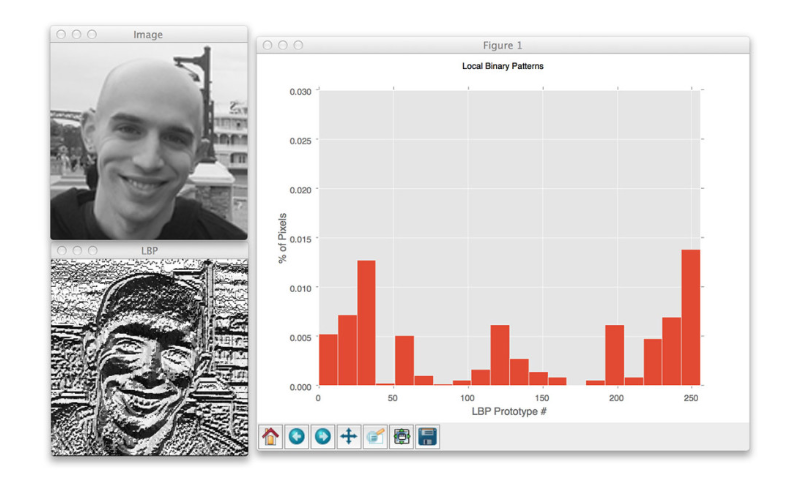
Giả sử chúng ta có một hình ảnh khuôn mặt trong thang độ xám. Chúng ta có thể lấy một phần của hình ảnh này dưới dạng cửa sổ 3x3 pixel (như hình 7). Với 8 pixel xung quanh, chúng ta có tổng cộng  2 ^ 8 = 256 tổ hợp mã LBP có thể.

Sau đó, chúng ta cần lấy giá trị trung tâm của ma trận được sử dụng làm ngưỡng. Giá trị này sẽ được sử dụng để xác định các giá trị mới lân cận.

Mỗi neighbor của các giá trị ngưỡng được đặt một giá trị nhị phân mới. Đặt 1 cho các giá trị bằng hoặc cao hơn ngưỡng và 0 cho các giá trị thấp hơn ngưỡng. Bây giờ, ma trận sẽ chỉ chứa các giá trị nhị phân. Chúng ta cần nối từng giá trị nhị phân từ mỗi vị trí từ dòng ma trận theo từng dòng thành một giá trị nhị phân mới (ví dụ ở hình trên : 00101101). Sau đó, chuyển đổi giá trị nhị phân này thành giá trị thập phân và đặt nó thành giá trị trung tâm của ma trận. Vào cuối quy trình này (thủ tục LBP), ta có một hình ảnh mới thể hiện rõ hơn các đặc điểm của hình ảnh gốc.

* 1. ****Trích xuất biểu đồ Histogram :****

Bây giờ, bằng cách sử dụng hình ảnh được tạo ở bước cuối cùng, chúng ta có thể sử dụng các tham số ****Grid X**** và ****Grid Y**** để chia hình ảnh thành nhiều lưới (ở đây sử dụng 1x1), như có thể thấy trong hình ảnh sau:



Hình 14: Trích xuất biểu đồ Histogram [7]

Dựa vào hình ảnh trên, có thể trích xuất biểu đồ của từng vùng như sau:

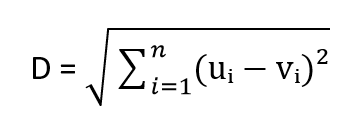
Khi có một hình ảnh ở thang độ xám, mỗi biểu đồ (từ mỗi lưới) sẽ chỉ chứa 256 vị trí (0 ~ 255) đại diện cho các lần xuất hiện của từng cường độ pixel. Sau đó nối từng biểu đồ để tạo ra biểu đồ mới và lớn hơn. Giả sử chúng ta có lưới 1x1, chúng ta sẽ có 256 đặc trưng LBP. Biểu đồ cuối cùng đại diện cho các đặc điểm của hình ảnh gốc hình ảnh.



****Hình 15. Sơ đồ khối trích xuất biểu đồ Histogram****

* 1. ****Thực hiện nhận dạng khuôn mặt dùng công thức Euclid :****

Trong bước này, thuật toán đã được đào tạo. Mỗi biểu đồ được tạo ra được sử dụng để thể hiện mỗi hình ảnh từ t ập dữ liệu đào tạo. Vì vậy, để tìm hình ảnh phù hợp với hình ảnh đầu vào, chúng ta chỉ cần so sánh hai biểu đồ và trả lại hình ảnh với biểu đồ gần nhất. Chúng ta có thể sử dụng các cách tiếp cận khác nhau để so sánh biểu đồ (tính khoảng cách giữa hai biểu đồ), ví dụ: khoảng cách euclide , chi bình phương , giá trị tuyệt đối,… Trong đồ án này, nhóm chúng em sử dụng khoảng cách Euclid theo công thức sau:



Hình 16: Công thức Euclid

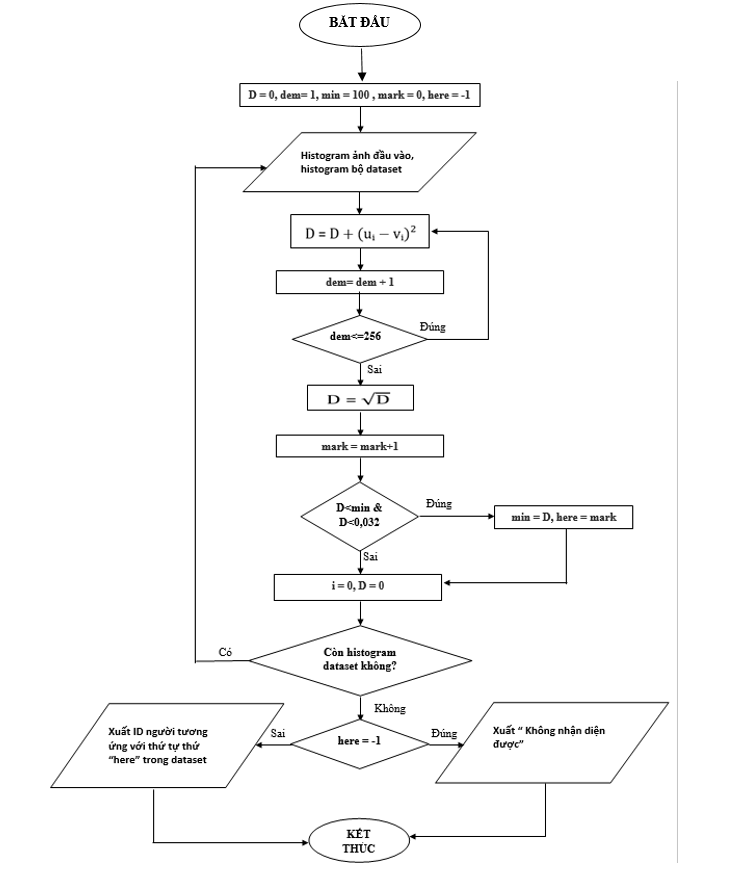
Trong đó:

D : khoảng cách euclid

uᵢ : histogram của dataset

vᵢ : histogram của ảnh đầu vào

n : tổ hợp mã LBP (256 đặc trưng)



**Hình 17. Sơ đồ khối áp dụng khoảng cách Euclid để nhận diện khuôn mặt**

Đầu ra thuật toán là ID từ hình ảnh với biểu đồ gần nhất. Thuật toán cũng sẽ trả về khoảng cách tính toán.

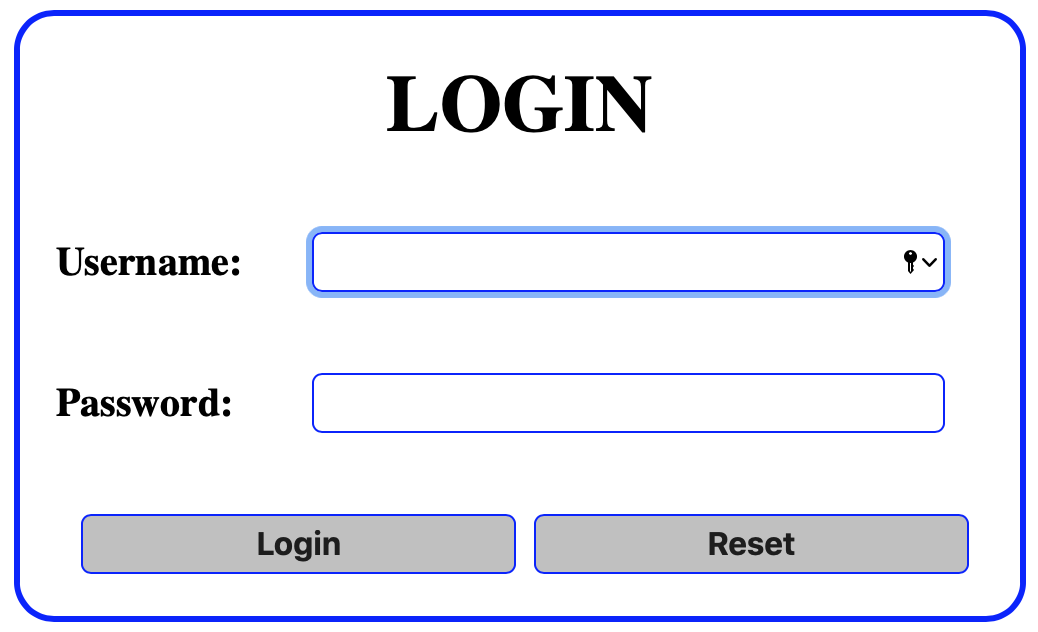
Sau đó, chúng ta có thể sử dụng ngưỡng để tự động ước tính nếu thuật toán đã nhận dạng chính xác hình ảnh. Chúng ta có thể giả định rằng thuật toán đã được công nhận thành công nếu khoảng cách tính toán thấp hơn ngưỡng được xác định.

1. Giới thiệu giao diện Web:

*Địa chỉ Web:* *http://doanvdk44.000webhostapp.com*

* 1. Giao diện Web:

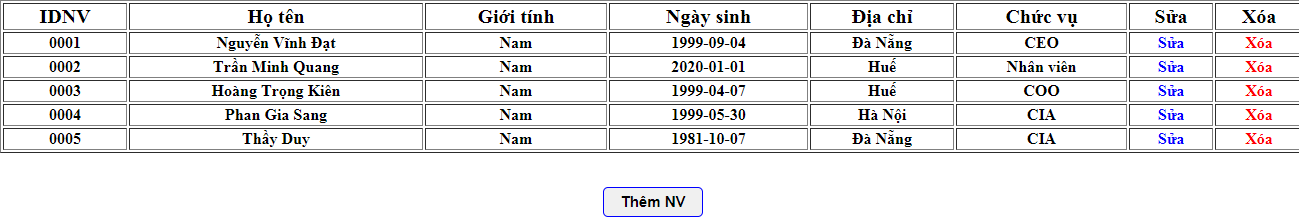
*Form đăng nhập:* Tại đây người quản trị hệ thống có thể đăng nhập vào và kiểm tra các lượt điểm danh của nhân viên. Từ đây, có thể cải thiện đồ án bằng cách phân quyền cho web nhằm cung cấp khả năng xem lượt điểm danh của chính bản thân nhân viên. ( Chức năng này nhóm em chưa phát triển. )



Hình 18: Form đăng nhập

* 1. Giao diện THÔNG TIN NHÂN VIÊN:

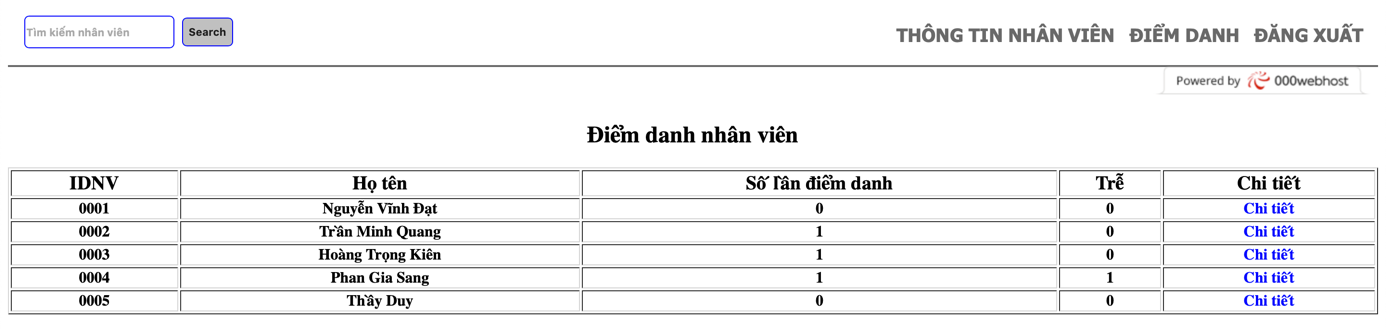
Là nơi hiển thị toàn bộ thông tin nhân viên có trong công ty.



Hình 19: Giao diện thông tin nhân viên

* 1. Giao diện ĐIỂM DANH:

Là nơi hiển thị toàn bộ số lần điểm danh của các nhân viên có trong công ty.



Hình 20: Giao diện điểm danh nhân viên

* 1. Giao diện CHI TIẾT ĐIỂM DANH:

Là nơi hiển thị chi tiết thời điểm mà nhân viên đó điểm danh. Nếu nhân viên điểm danh trước 8:00 AM thì thời gian có màu đen. Ngược lại, nếu nhân viên điểm danh sau 8:00 AM thì thời gian có màu đỏ. Có nghĩa là nhân viên đã đi trễ trong lần điểm danh đó.



Hình 21: Giao diện điểm danh đúng giờ



Hình 22: Giao diện điểm danh trễ giờ

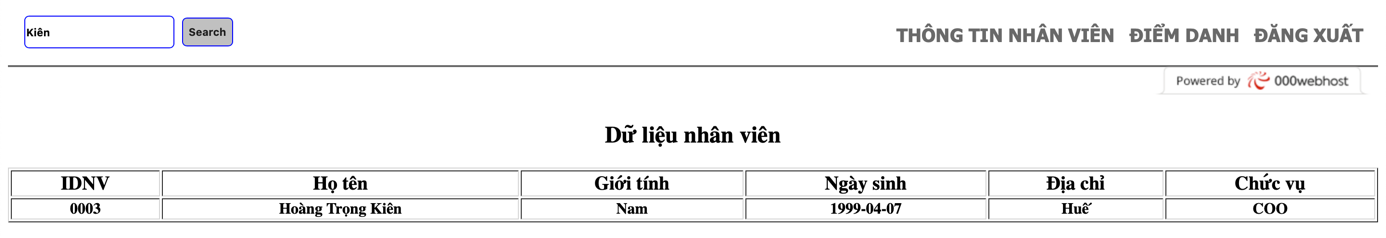
* 1. Chức năng tìm kiếm:



Hình 23: Form tìm kiếm

Khi hệ thống có quá nhiều nhân viên. Việc tìm kiếm nhân viên cho 1 mục đích nào đó trở nên khó khăn. Vì vậy chức năng tìm kiếm ra đời nhằm giúp người quản trị tìm kiếm 1 cách nhanh gọn.

Kết quả:



Hình 24: Giao diện tìm kiếm

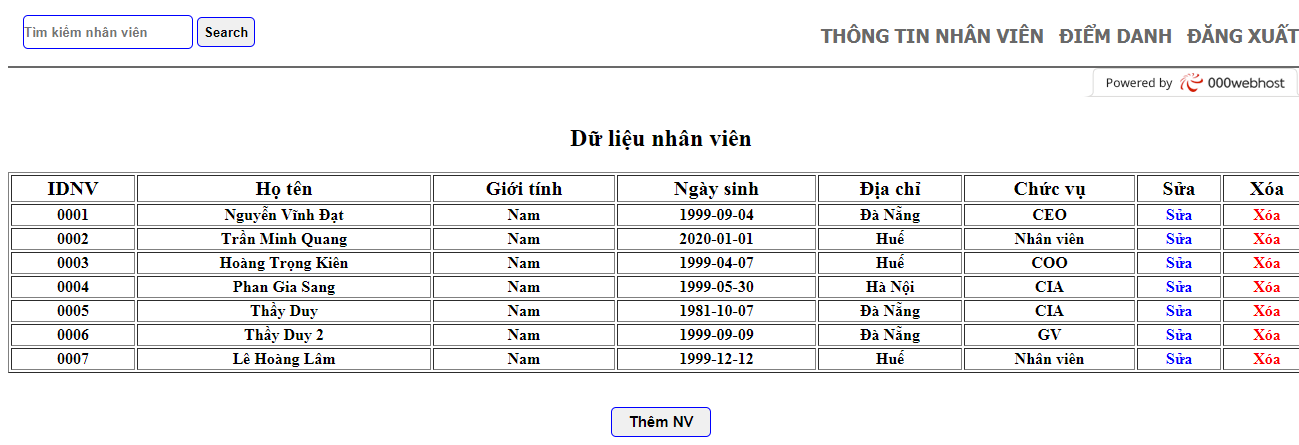
7.6. Chức năng thêm nhân viên :



Hình 25: Form thêm nhân viên

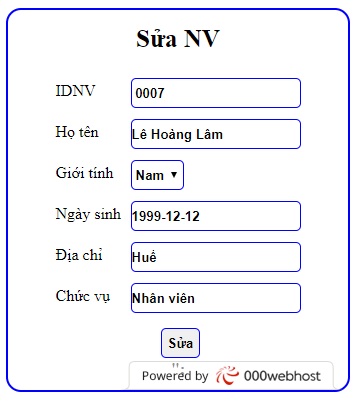
Khi muốn có một nhân viên mới vào làm việc thì để giúp người quản trị có thêm thông tin nhân viên mới vào để quản lí nên chức năng thêm nhân viên được cài đặt vào hệ thống.

Kết quả :



Hình 26: Kết quả sau khi thêm nhân viên

**7.7 Chức năng sửa thông tin nhân viên :**



**Hình 27: Form sửa thông tin nhân viên**

Chức năng sửa thông tin nhân viên ra đời để giúp người quản trị dễ dàng muốn thay đổi thông tin của nhân viên nào đó mà không phải vào CSDL để sửa.

Kết quả:

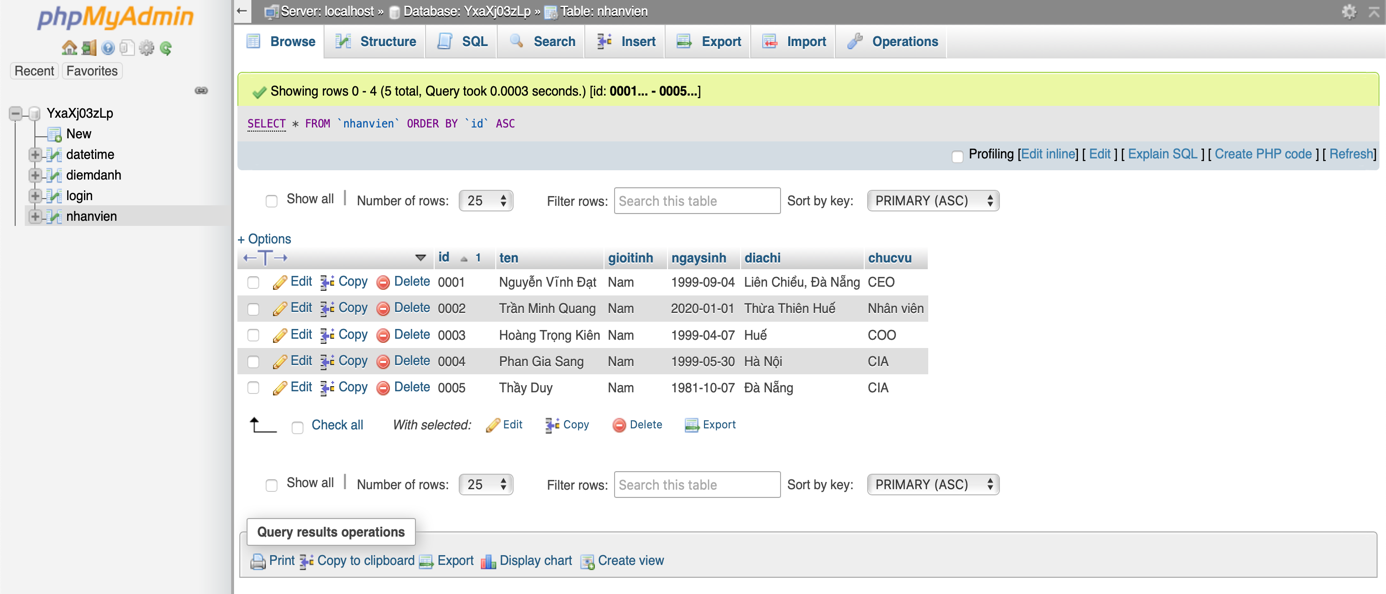


**Hình 28: Kết quả sau khi sửa thông tin nhân viên**

1. Giới thiệu về Database:
   1. Giao diện quản lý database:

Ở đồ án này, chúng em sử dụng MySQL. Bằng phpMyAdmin để xử lý & quản trị MySQL thông qua một trình duyệt web.

Database gồm 3 bảng chính: nhanvien, diemdanh, datetime.

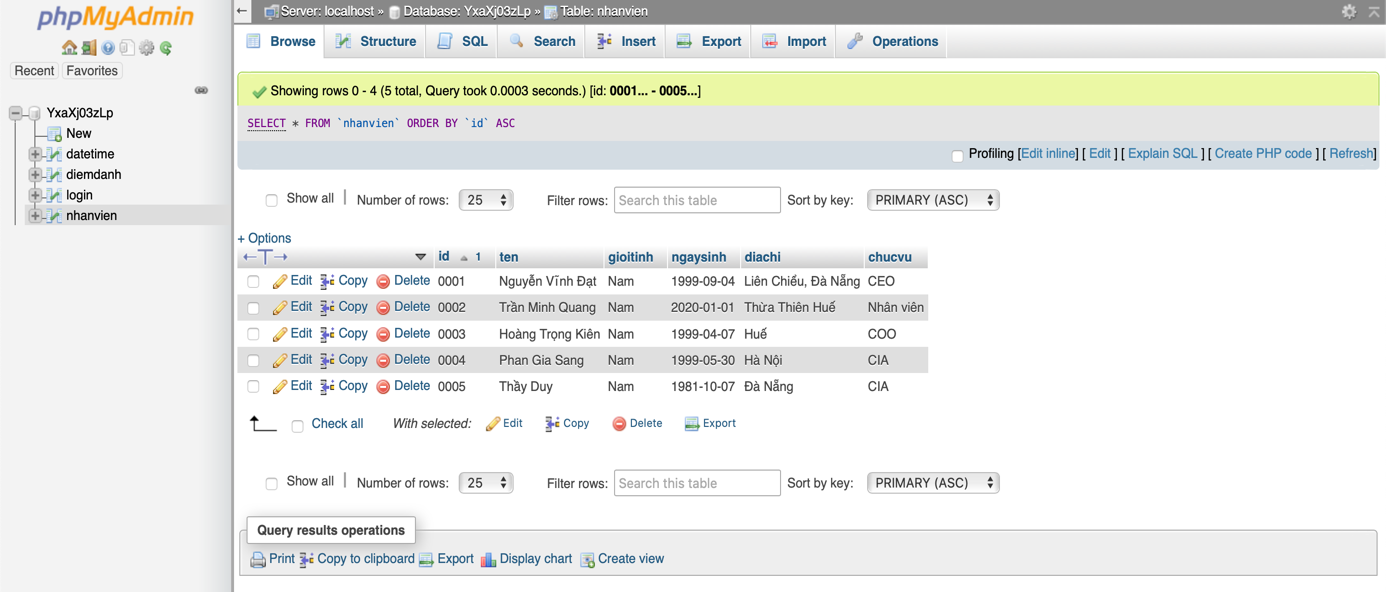


Hình 29: Giao diện quản lý database thông qua trình duyệt web

* 1. Bảng nhanvien:

Chứa toàn bộ thông tin nhân viên có trong công ty:

* id: Là mã số của mỗi nhân viên. Mỗi nhân viên chỉ có 1 mã id mà mã này là duy nhất (primary key).
* ten: Là họ và tên của nhân viên.
* gioitinh: Là giới tính của nhân viên.
* ngaysinh: Là ngày tháng năm sinh của nhân viên.
* diachi: Là địa chỉ hiện tại của nhân viên.
* chucvu: Là chức vụ hiện tại của nhân viên trong công ty.

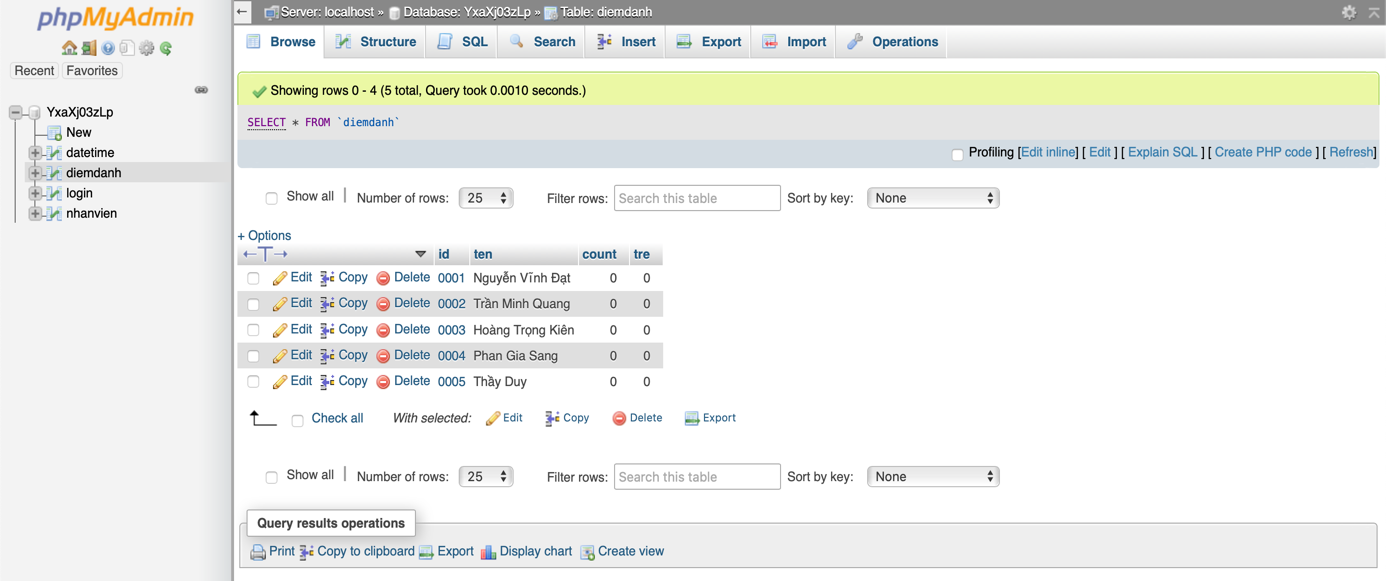


Hình 30: Bảng NHANVIEN

* 1. Bảng diemdanh:

Chứa toàn bộ thông tin số lần điểm danh của nhân viên có trong công ty.

* id: Là khoá ngoại của bảng này. Nó trỏ đến trường id của bảng nhanvien.
* ten: Là họ và tên của nhân viên.
* count: Là tổng số lần điểm danh của nhân viên. Số lần điểm danh này có thể lấy 1 tháng để làm đơn vị cơ sở. Sau 1 tháng thì bộ phận nhân sự của công ty có thể lấy toàn bộ thông tin điểm danh trong tháng về và reset database ( nghĩa là cột count này sẽ bằng 0 ).
* tre: Là tổng số lần trễ của nhân viên trong 1 đơn vị thời gian ( ở đây chúng em giả sử là 1 tháng ).

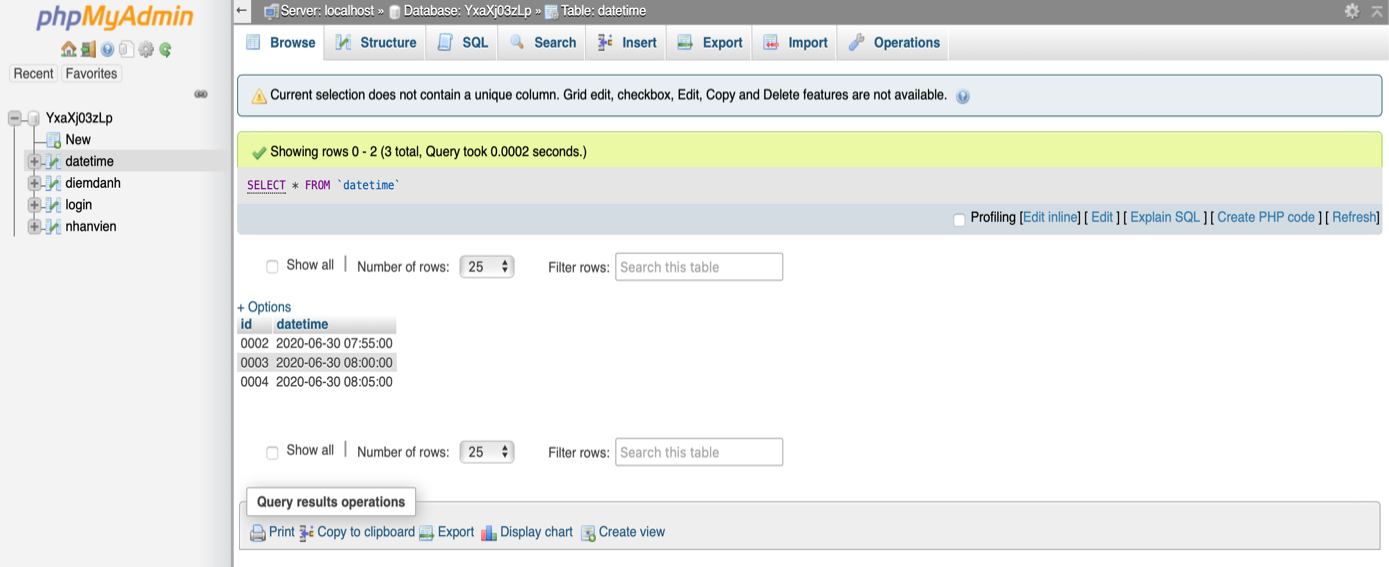


Hình 31: Bảng DIEMDANH

* 1. Bảng datetime:

Chứa toàn bộ thông tin số lần điểm danh của nhân viên có trong công ty.

* id: Là id của nhân viên trong lần điểm danh đó.
* datetime: Là thời điểm mà nhân viên điểm danh.



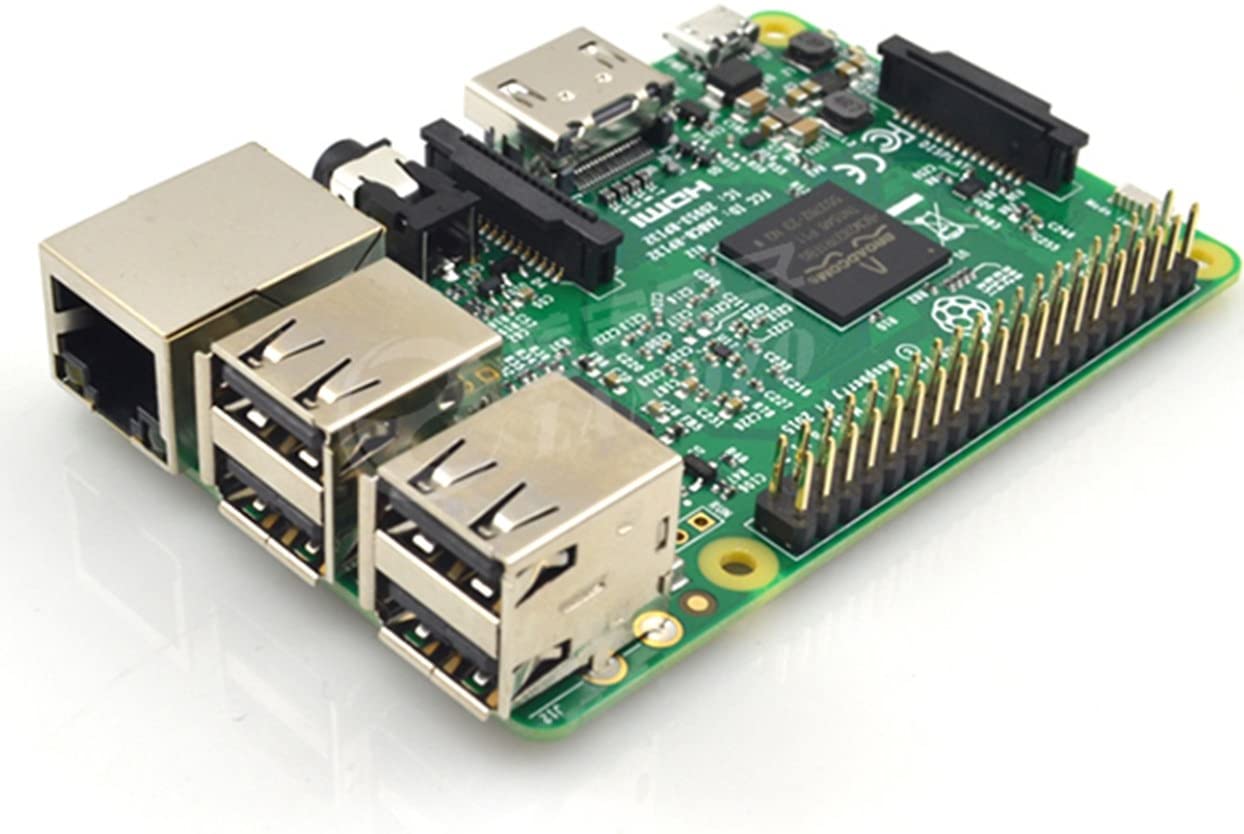
Hình 32: Bảng DATETIME

1. Giới thiệu một số phần cứng, linh kiện sử dụng:

Hệ thống bao gồm : 1 raspberry Pi 3, 1 camera, 1 loa, nút bấm, dây dẫn, nguồn, breadboard

* Raspberry Pi 3 :

Thông số kĩ thuật :



* 64-bit quad-core ARM Cortex-A53
* 802.11 b/g/n Wireless LAN
* Bluetooth 4.1 (Classic & Low Energy)
* 1 GB LPDDR2 memory
* MicroUSB connector for 2.5 A power supply

Hình 33: Raspberry Pi 3

* 1 cổng 10/100 Ethernet
* 1 cổng HDMI
* 4 cổng USB 2.0
* 40 chân GPIO
* Kích thước: 85 x 56 x 17 mm

# Camera Genius Facecam 2020 :

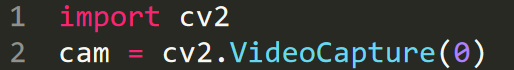


Thông số kĩ thuật :

* Cổng kết nối: USB
* Độ phân giải: HD720P/ 2 megapixel 30fps.

Đoạn code kết nối với Raspberry :

Hình 34: Camera Genius



Câu lệnh thứ 2 dùng để mở camera và nhận hình ảnh đầu vào từ camera. Phương thức VideoCapture(int index) có 1 tham số đầu vào đó là id của camera cần dùng.

* Loa:

Dùng để thông báo đến người dùng kết quả nhận dạng thành công hay không thể nhận dạng được.



Thông số kĩ thuật:

* Công suất: 5W
* Điện áp hoạt động: 5V/1A
* Kích thước: 74 x 34 mm

Hình 35: Loa

Đoạn code kết nối với Raspberry:



Câu lệnh đầu tiên khai báo một file âm thanh, trong ảnh đang khai báo 1 file âm thanh có tên point.wav. Câu lệnh thứ hai phát file âm thanh đó.



* Nguồn :

Trong đồ án này nhóm em sử dụng nguồn là pin sạc dự phòng Xiaomi Mi Gen 2 có khả năng điều chỉnh cường độ dòng điện phù hợp với Raspberry Pi 3.

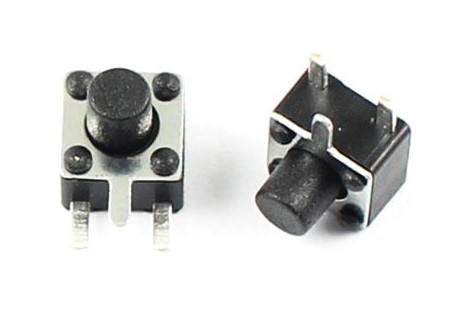
Thông số kĩ thuật:

* Dòng điện ra: 5.1V-2.4A / 9V-1.6A (MAX) và 12V-1.2A (MAX) - MAX 15W

Hình 36: Pin sạc

* Nút nhấn :

Hệ thống này được trang bị 3 nút nhấn có các chức năng: bắt đầu nhận dạng để điểm danh, đăng ký mới khuôn mặt và hủy bỏ các quá trình.



Thông số kĩ thuật:

Hình 37: Nút nhấn

* Kích thước: 4.5 x 4.5 x 8 mm
* Số chân: 4

Đoạn code kết nối với Raspberry :



Câu lệnh trên khai báo một nút nhấn được nối với chân GPIO trên raspberry. Trong đồ án này, nhóm chúng em sử dụng 3 chân GPIO2, GPIO3, GPIO4 trên raspberry.

Bảng 2. Bảng kê chi phí các linh kiện

|  |  |  |
| --- | --- | --- |
| STT | Tên linh kiện | Giá |
| 1 | Raspberry Pi 3 | 990000 VNĐ |
| 2 | Camera Genius Facecam 2020 | 490000 VNĐ |
| 3 | Pin sạc dự phòng Xiaomi | 300000 VNĐ |
| 4 | Loa | 100000 VNĐ |
| 5 | Breadboard socket | 32000 VNĐ |
| 6 | Nút nhấn | 10000 VNĐ |
| 7 | Dây dẫn | 10000 VNĐ |
|  | Tổng cộng: | 1932000 VNĐ |

# KẾT QUẢ :

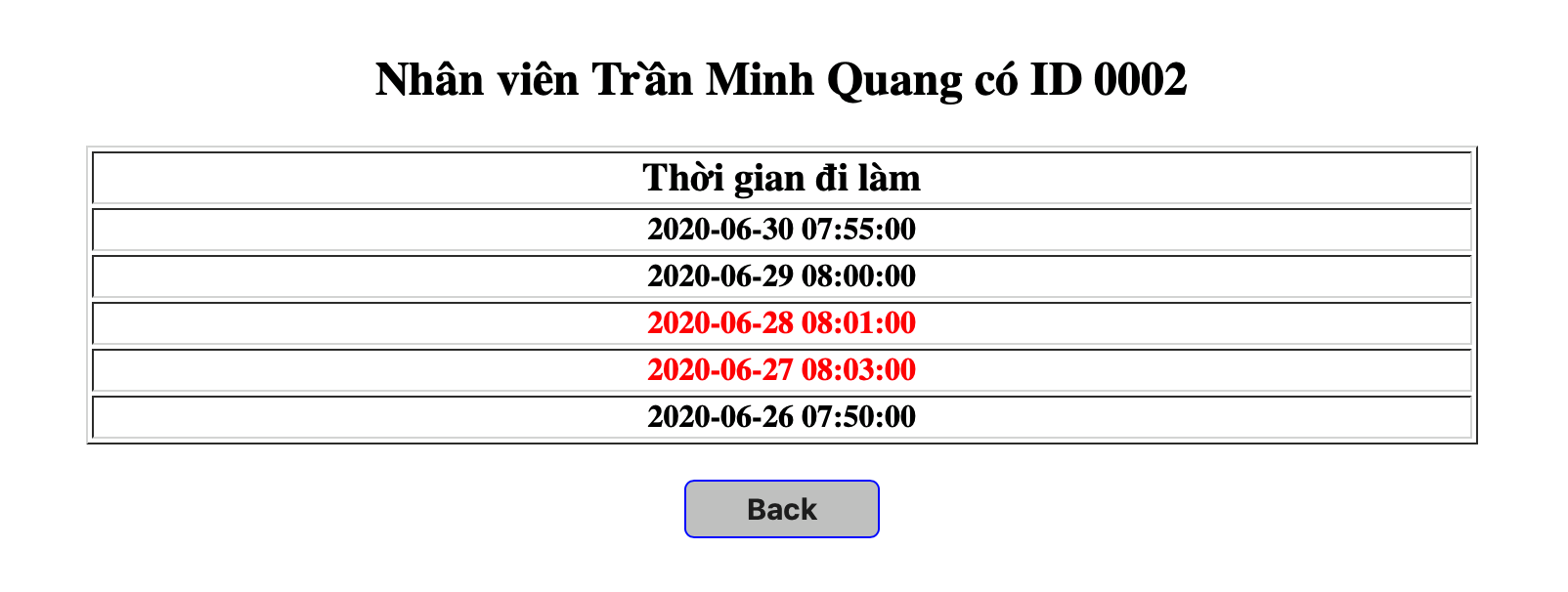
1. **Mô tả dữ liệu kiểm thử :**

# Nhóm chúng em sử dụng tập dữ liệu từ 4 người gồm các thành viên trong nhóm. Mỗi người được thu thập 10 ảnh chụp khuôn mặt bằng một chương trình được viết bằng ngôn ngữ Python. Mỗi ảnh trắng đen có độ phân giải từ 180x180 pixel đến 210x210 pixel tùy thuộc vào cấu trúc và kích thước khuôn mặt của mỗi người. Mười ảnh của cùng một người được lưu trữ ở 1 thư mục có tên là ID của nhân viên đó. Các ảnh đăng ký này được chụp lại trong điều kiện ánh sáng đầy đủ, không có tia sáng chiếu thẳng vào khuôn mặt và camera.

1. **Quy trình kiểm thử :**

Sau khi hoàn thiện sản phẩm, nhóm đã tiến hành triển khai thử nghiệm đăng ký khuôn mặt và nhận dạng khuôn mặt cho thấy kết quả khá chính xác.

Hình 38. Dữ liệu điểm danh của một nhân viên



Với giờ làm việc đã quy định trước, ví dụ quy định giờ vào làm việc là 8 giờ thì ở hình trên, những dòng màu đỏ biểu thị những thời gian đó nhân viên đã đi làm muộn (sau 8 giờ) và những dòng màu đen cho thấy thời gian đó nhân viên đi làm đúng giờ (trước hoặc đúng 8 giờ).

Với hàm detectMultiScale, giá trị scale\_factor = 1.3 và min\_neighbor = 5, thu được kết quả như sau:

**Bảng 3. Giá trị 4 lần đo thời gian đăng ký mới khuôn mặt (10 ảnh)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lần thứ** | **Nhân viên** | **Khoảng cách từ mặt đến camera (cm)** | **Ánh sáng** | **Thời gian hoàn thành (s)** |
| 1 | Đạt | 30 | Đủ sáng | 20.21 |
| 2 | Kiên | 30 | Đủ sáng | 20.96 |
| 3 | Quang | 30 | Đủ sáng | 21.24 |
| 4 | Sang | 30 | Đủ sáng | 22.87 |

Thời gian hoàn thành đăng ký trung bình 4 lần đo: **21.32s**

Thời gian chênh lệch trung bình 4 lần đo:  **0.77s**

**Bảng 4. Giá trị 24 lần đo thời gian nhận dạng khuôn mặt**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lần thứ** | **Nhân viên** | **Khoảng cách từ mặt đến camera (cm)** | **Ánh sáng** | **Kết quả nhận dạng** |
| 1 | Đạt | 30 | Mạnh, mặt ít vùng tối | Sai |
| 2 | Đạt | 60 | Mạnh, mặt ít vùng tối | Sai |
| 3 | Đạt | 30 | Đủ sáng | Đúng |
| 4 | Đạt | 60 | Đủ sáng | Đúng |
| 5 | Đạt | 30 | Yếu, mặt nhiều vùng tối | Đúng |
| 6 | Đạt | 60 | Yếu, mặt nhiều vùng tối | Đúng |
| 7 | Kiên | 30 | Mạnh, mặt ít vùng tối | Sai |
| 8 | Kiên | 60 | Mạnh, mặt ít vùng tối | Sai |
| 9 | Kiên | 30 | Đủ sáng | Đúng |
| 10 | Kiên | 60 | Đủ sáng | Đúng |
| 11 | Kiên | 30 | Yếu, mặt nhiều vùng tối | Đúng |
| 12 | Kiên | 60 | Yếu, mặt nhiều vùng tối | Đúng |
| 13 | Quang | 30 | Mạnh, mặt ít vùng tối | Sai |
| 14 | Quang | 60 | Mạnh, mặt ít vùng tối | Sai |
| 15 | Quang | 30 | Đủ sáng | Đúng |
| 16 | Quang | 60 | Đủ sáng | Sai |
| 17 | Quang | 30 | Yếu, mặt nhiều vùng tối | Đúng |
| 18 | Quang | 60 | Yếu, mặt nhiều vùng tối | Sai |
| 19 | Sang | 30 | Mạnh, mặt ít vùng tối | Đúng |
| 20 | Sang | 60 | Mạnh, mặt ít vùng tối | Sai |
| 21 | Sang | 30 | Đủ sáng | Đúng |
| 22 | Sang | 60 | Đủ sáng | Sai |
| 23 | Sang | 30 | Yếu, mặt nhiều vùng tối | Đúng |
| 24 | Sang | 60 | Yếu, mặt nhiều vùng tối | Đúng |

Với ảnh đầu vào và ảnh trong dataset được trích xuất 256 tổ hợp mã LBP làm vectơ đặc trưng Histogram. Sau khi chạy thử trên 4 người với các điều kiện ánh sáng, khoảng cách từ mặt người đến camera khác nhau, ra được kết quả:

Thời gian hoàn thành nhận dạng trung bình 24 lần đo: **3.07s**

Thời gian chênh lệch trung bình 24 lần đo:  **0.17s.**

# Tỉ lệ nhận dạng thành công trong 24 lần thử: 58.33%

Với số liệu nhận dạng ở bảng trên, ta thấy trong điều kiện ánh sáng mạnh hoặc yếu hệ thống hoạt động thiếu chính xác. Về yếu tố khoảng cách giữa khuôn mặt và camera thu nhận ảnh, khi đưa mặt cách khá xa (60cm) thì hệ thống hoạt động thiếu chính xác hơn so với để khoảng cách gần (30cm). Từ đó, ra rút ra được kết luận hệ thống nhận dạng này hoạt động thiếu chính xác trong điều kiện ánh sáng mạnh hoặc yếu và khoảng cách từ mặt đến camera quá xa so với giai đoạn đăng ký khuôn mặt.

# 

1. **KẾT LUẬN :**
2. **Ưu điểm:**

* Sản phẩm đáp ứng các yêu cầu đề ra, triển khai và vận hành tương đối tốt
* Có thông báo cho người sử dụng bằng loa
* Hệ thống dễ sử dụng
* Theo dõi thông tin thời gian điểm danh dễ dàng

1. **Nhược điểm:**

* Linh kiện còn rườm rà, hình thức bên ngoài chưa gọn nhẹ
* Quá trình nhận dạng chưa thực sự nhanh và chính xác
* Chỉ hoạt động được khi có kết nối Internet

1. **Kết luận:**

* Hệ thống đã đáp ứng được các yêu cầu đề ra ban đầu.
* Quá trình nhận dạng tương đối chính xác.
* Có thể đáp ứng được nhu cầu của các doanh nghiệp nhỏ.

1. **Hướng phát triển:**

* Cải thiện quá trình nhận dạng nhanh và chính xác hơn.
* Có thể hoạt động ngay cả khi không có kết nối Internet.
* Thiết kế giao diện web trực quan và đẹp mắt hơn.
* Phân quyền cho ứng dụng.

# TÀI LIỆU THAM KHẢO

[1] OpenCV Dev team, “Face Recognition with OpenCV”, OpenCV, Dec 31, 2019. [Online].Available:https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec\_tutorial.html. [Accessed: Feb 12, 2020].

[2] [Zuzana Képešiová](https://www.researchgate.net/profile/Zuzana_Kepesiova), “Integral Image”, [Online]. Availble: <https://www.researchgate.net/figure/mage-conversion-to-summed-area-table-also-known-as-Integral-image_fig1_324554906>. [Accessed: Feb 13, 2020].

[3] M. Grgic, and K. Delac, “Face recognition homepage,” FACE RECOGNITION HOMEPAGE, 28 August 2019. [Online]. Available: http://www.face-rec.org.[Accessed: Feb 12, 2020].

[4] Le Van Lap, “Nhận diện khuôn mặt OpenCV Python,”, Apr 24, 2019. [Online]. Availble: <https://github.com/leevawns/opencv-haar-gui>. [Accessed: Feb 12, 2020].

[5] Gpiozero, “Basic Recipes,”. [Online]. Available: <https://gpiozero.readthedocs.io/en/stable/recipes.html>. [Accessed: March 28, 2020].

[6] [Kelvin Salton do Prado](https://towardsdatascience.com/@kelvin_sp?source=post_page-----90ec258c3d6b----------------------), “Face Recognition: Understanding LBPH Algorithm,”, 11 November 2017. [Online]. Available:  
<https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b>. [Accessed: Feb 08, 2020].

[7] [Adrian Rosebrock](https://www.pyimagesearch.com/author/adrian/), “Local Binary Patterns with Python & OpenCV,”, 7 December 2015. [Online]. Available:  
https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/ [Accessed: June 30, 2020].

[8] Pham Minh Phuong, “Giải thuật Adaboost ứng dụng trong nhận dạng biển số xe(Data Mining)”, 20 May 2019. [Online]. Available:  
https://viblo.asia/p/giai-thuat-adaboost-ung-dung-trong-nhan-dang-bien-so-xedata-mining-07LKXxbPKV4?fbclid=IwAR3pFbcOTA6Q1pLBaBmZSZMK6HJSd4sB9Oc3\_xk95EbbWnWCCYcBLpngxsA [Accessed: June 30, 2020].

[9] Jehoshaphat Abu, “100 Days Of ML Code — Day 040”, 18 August 2018. [Online]. Available:  
https://medium.com/@jehoshaphatia/100-days-of-ml-code-day-040-c2d5156dde93/ [Accessed: June 30, 2020].

[10] Hải Hà, “Tìm hiểu về phương pháp nhận diện khuôn mặt của Violas & John”, 16 March 2019. [Online]. Available:  
https://viblo.asia/p/tim-hieu-ve-phuong-phap-nhan-dien-khuon-mat-cua-violas-john-ByEZkNVyKQ0?fbclid=IwAR1KA-B7JLDjDpG9P8ymTVu09DQetpsWG9Pma7H3TP--QtuKW2M3ysrBjVo/ [Accessed: June 30, 2020].