



TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO

PBL5 - ĐỒ ÁN KỸ THUẬT MÁY TÍNH

HỆ THỐNG MỞ CỬA TỰ ĐỘNG SỬ DỤNG NHẬN DIỆN KHUÔN MẶT

Cán bộ doanh nghiệp hướng dẫn: Trần Phương Nam

Nguyễn Đức Huy

Giảng viên đồng hướng dẫn: TS. Ninh Khánh Duy

STT NHÓM: 7	LỚP HỌC PHẦN
HỌ VÀ TÊN SINH VIÊN	ĐỒ ÁN
Bùi An Nguyên	18N14B
Nguyễn Hồng Huy	18N14B
Trần Thế Dân	18N14B
Lương Thế Long	18N14B

ĐÀ NẴNG, 06/2021

TÓM TẮT ĐỒ ÁN

Chắc hẳn ai trong chúng ta cũng có ít nhất một lần làm mất hay để quên chìa khóa xe, chìa khóa nhà, ... lúc đó bạn đã làm gì? Đa số chúng ta đều lựa chọn phương án phá ổ khóa và sau đó thay lại ổ khóa mới. Nhưng làm như vậy vừa tốn một số tiền thay khóa mới, đối với những ổ khóa có thiết kế gắn liền với cửa hay đồ vật, nếu phá khóa có thể gây hư hại những đồ vật đó. Vậy nếu như chúng ta có một hệ thống mở khoá một cách tự động bằng cách sử dụng nhận diện khuôn mặt thì chúng ta có thể sử dụng khuôn mặt của chính mình thay cho chìa khoá đó.

Hệ thống sẽ gồm 5 phần là ổ khoá điện tử, camera để chụp ảnh mỗi khi phát hiện khuôn mặt, server xử lý việc nhận diện khuôn mặt và ứng dụng trên điện thoại để người dùng có thể dễ dàng điều khiển hệ thống. Ổ khoá sẽ mở khi server nhận diện được khuôn mặt của người trong database. Khi không nhận diện được khuôn mặt, hệ thống sẽ gửi thông báo đến điện thoại để người dùng quyết định việc mở cửa hay không.

Thông qua quá trình làm đồ án, nhóm đã xây dựng thành công hệ thống mở cửa tự động bằng nhận diện khuôn mặt.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tự đánh giá theo 3 mức (Đã hoàn thành/Chưa hoàn thành/Không triển khai)
Bùi An Nguyễn	<ul style="list-style-type: none"> – Phát hiện khuôn mặt – Gửi hình ảnh từ camera đến server 	<ul style="list-style-type: none"> – Đã hoàn thành – Đã hoàn thành
Nguyễn Hồng Huy	<ul style="list-style-type: none"> – Xây dựng API server – Xây dựng database – Xây dựng kết nối giữa khoá và server 	<ul style="list-style-type: none"> – Đã hoàn thành – Đã hoàn thành – Đã hoàn thành
Trần Thế Dâng	<p>Xây dựng app điện thoại có các chức năng:</p> <ul style="list-style-type: none"> – Đăng nhập – Xem thông tin người quen trong database – Thông báo có người trước cửa – Mở cửa cho người lạ – Lưu thông tin người lạ 	<ul style="list-style-type: none"> – Đã hoàn thành – Đã hoàn thành – Đã hoàn thành – Đã hoàn thành – Đã hoàn thành
Lương Thế Long	<ul style="list-style-type: none"> – Xây dựng module nhận diện khuôn mặt 	<ul style="list-style-type: none"> – Đã hoàn thành

MỤC LỤC

DANH MỤC HÌNH VẼ.....	4
DANH MỤC BẢNG BIỂU	5
1. Giới thiệu	6
2. Giải pháp	6
2.1. Giải pháp về phần cứng và truyền thông	7
2.1.1. Giải pháp về phần cứng.....	7
2.1.2. Giải pháp về truyền thông	8
2.2. Giải pháp về phần mềm.....	8
2.2.1. Phát hiện khuôn mặt	8
2.2.2. Nhận diện khuôn mặt	9
2.2.3. Server	11
2.2.4. Điều khiển khoá	12
2.2.5. Mobile app	12
3. Kết quả.....	17
3.1. Phát hiện khuôn mặt.....	17
3.2. Nhận diện khuôn mặt	19
3.3. Server	20
3.4. Điều khiển khoá.....	20
3.5. Mobile app	21
4. Kết luận	24
4.1. Đánh giá	24
4.2. Hướng phát triển.....	24
5. Danh mục tài liệu tham khảo	24

DANH MỤC HÌNH VẼ

Hình 1. Sơ đồ tổng quan hệ thống	6
Hình 2. Phát hiện khuôn mặt	9
Hình 3. Nhận diện khuôn mặt.....	10
Hình 4. Thành phần ứng dụng	13
Hình 5. Chức năng đăng nhập	14
Hình 6. Chức năng thông báo	15
Hình 7. Chức năng mở cửa.....	15
Hình 8. Chức năng thêm người lạ	16
Hình 9. Chức năng xem người quen.....	16
Hình 10. Chức năng xoá người quen.....	17
Hình 11. Hình ảnh từ PiCamera	18
Hình 12. Kết quả sau khi phát hiện khuôn mặt	19
Hình 13. Dữ liệu huấn luyện phát hiện khuôn mặt.....	19
Hình 14. Kết quả thử nghiệm phát hiện khuôn mặt	20
Hình 15. Module wifi gọi API open-door của server.....	20
Hình 16. Khoá được mở	20
Hình 17. Khoá tự động đóng	21
Hình 18. Giao diện khởi động	21
Hình 19. Giao diện nhập ip server.....	22
Hình 20. Giao diện đăng nhập	22
Hình 21. Giao diện khi xuất hiện người quen	23
Hình 22. Giao diện khi xuất hiện người lạ	23
Hình 23. Giao diện xem người quen	24

DANH MỤC BẢNG BIỂU

Bảng 1. Bảng thống kê linh kiện	7
Bảng 2. Mô hình phát hiện khuôn mặt	17

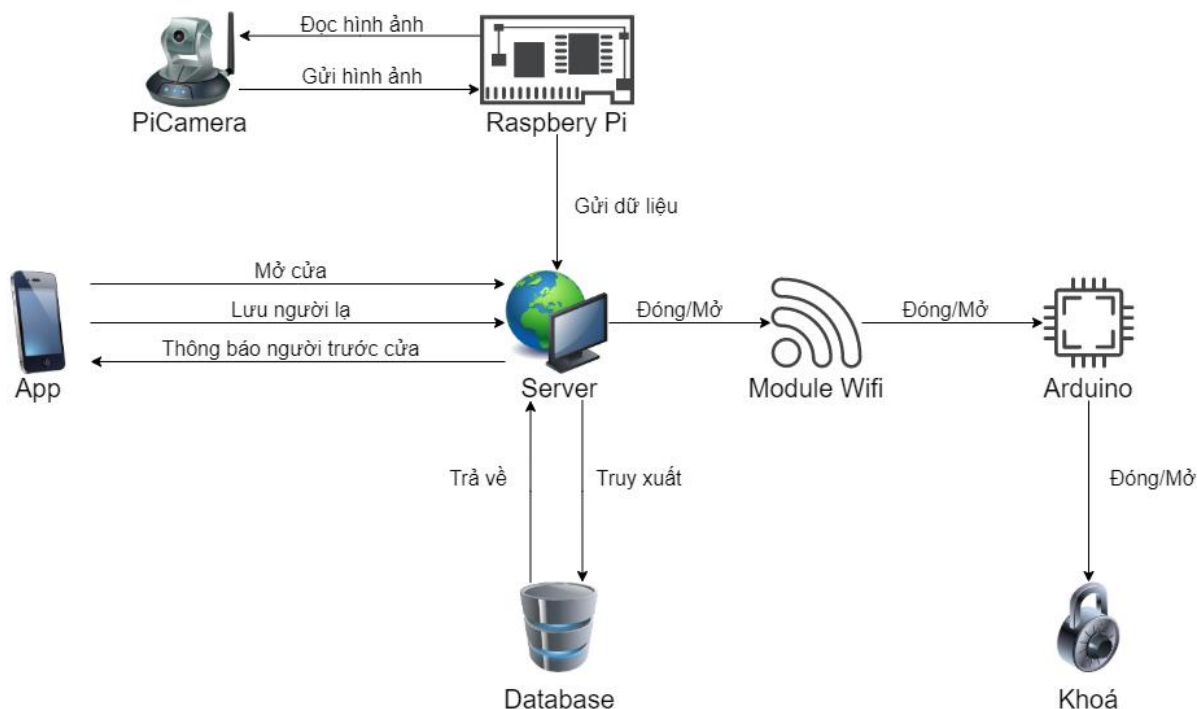
1. Giới thiệu

Nhận diện khuôn mặt là đề tài nhận được sự quan tâm, nghiên cứu của khá nhiều nhà khoa học trong suốt những thập niên vừa qua. Kết quả của những sự nghiên cứu này đã được ứng dụng vào bảo mật trên thiết bị di động như iPhone hay hệ thống điểm danh trong trường học hay doanh nghiệp.

Chắc hẳn ai trong chúng ta cũng có ít nhất một lần làm mất hay để quên chìa khóa xe, chìa khóa nhà,... lúc đó bạn đã làm gì? Đa số chúng ta đều lựa chọn phương án phá ổ khóa và sau đó thay lại ổ khóa mới. Nhưng làm như vậy vừa tốn một số tiền thay khóa mới, đối với những ổ khóa có thiết kế gắn liền với cửa hay đồ vật, nếu phá khóa có thể gây hư hại những đồ vật đó. Thay vào đó, với sự phát triển của công nghệ nhận diện khuôn mặt, chúng ta có thể thay khuôn mặt chúng ta cho những chiếc chìa khoá đó. Vì vậy mà nhóm em đã chọn đề tài “Hệ thống mở cửa tự động sử dụng nhận diện khuôn mặt” là đề tài cho đồ án nhóm em. Đây là một đề tài có ý nghĩa nhân văn sâu sắc vì thông qua đề tài nhóm có thể phát triển ra các hệ thống hỗ trợ con người tốt hơn.

2. Giải pháp

Sơ đồ tổng quan hệ thống:



Hình 1. Sơ đồ tổng quan hệ thống



Mô tả:



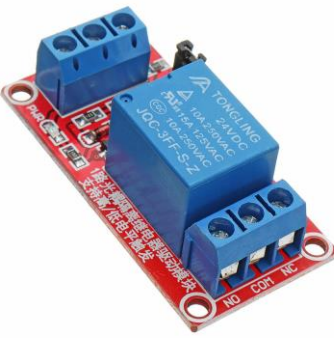
- Camera ghi lại hình ảnh và gửi về cho raspberry xử lý.
- Raspberry đọc hình ảnh từ camera, xử lý phát hiện khuôn mặt. Khi phát hiện khuôn mặt thì gửi về server xử lý.
- Server nhận hình ảnh đã phát hiện khuôn mặt từ raspberry, sử dụng module nhận diện khuôn mặt và so khớp với các khuôn mặt có trong database. Khi nhận diện được thì gửi thông báo đến app và cấp quyền mở khoá cho ổ khoá. Khi không nhận diện được thì gửi thông báo đến app yêu cầu quyền mở khoá.
- App đảm nhận việc xử lý yêu cầu mở khoá khi có người lạ cũng như gửi yêu cầu lưu thông tin đến server để lưu người lạ vào hệ thống.
- Ổ khoá được điều khiển qua module role kết nối với Arduino. Arduino nhận lệnh mở cửa thông qua module wifi ESP8266 kết nối với server.

2.1. Giải pháp về phần cứng và truyền thông

2.1.1. Giải pháp về phần cứng

Bảng 1. Bảng thống kê linh kiện

Tên linh kiện	Hình ảnh	Mô tả
Raspberry Pi 3 Model B		Nguồn 5V, giá: 1090000đ
Picamera		Giá: 90000đ

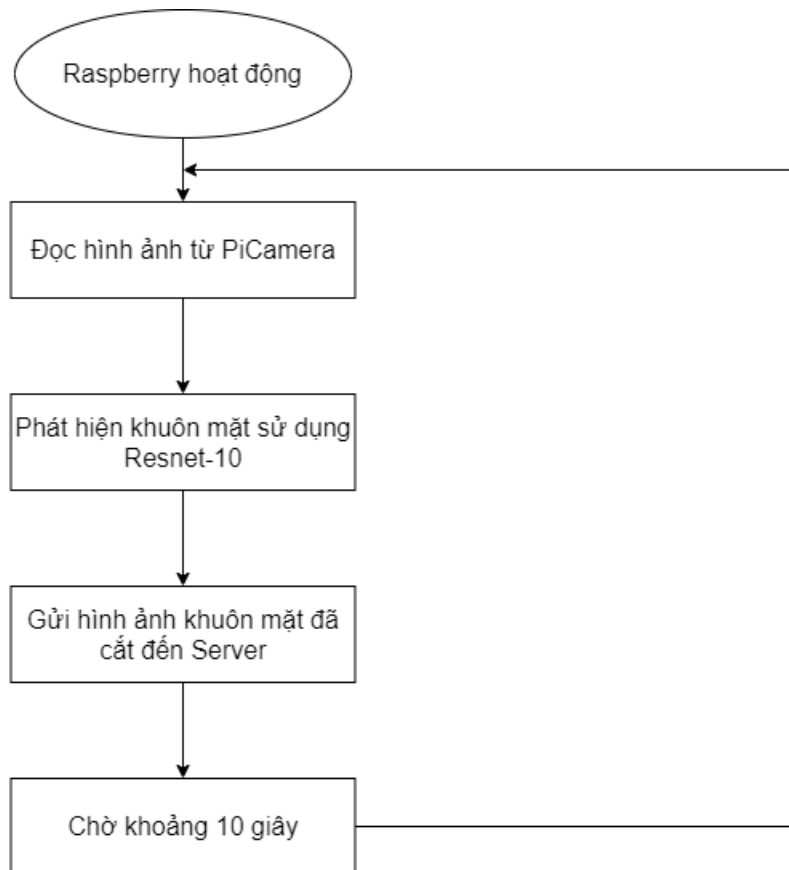
Arduino uno		Nguồn 5V, giá: 100000đ
ModuleWifi ESP8266		Nguồn 5V, giá: 60000đ
Module Rơ le		Nguồn 5V, giá: 25000đ
Tổng cộng		1365000đ

2.1.2. Giải pháp về truyền thông

- Picamera kết nối với raspberry thông qua MIPI CSI camera port trên raspberry.
- Raspberry giao tiếp với server thông qua Wifi.
- Server và mobile app giao tiếp với nhau thông qua các API được cung cấp bởi server.

2.2. Giải pháp về phần mềm

2.2.1. Phát hiện khuôn mặt

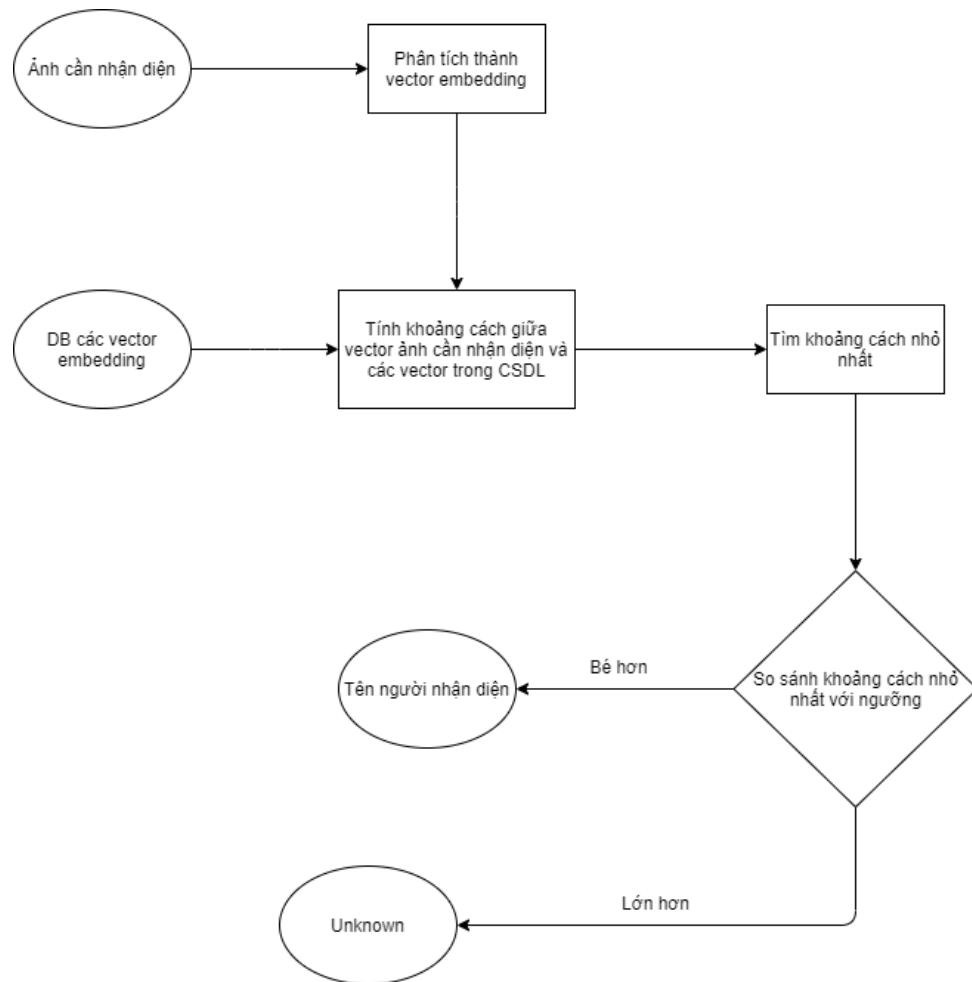


Hình 2. Phát hiện khuôn mặt

Về mô hình phát hiện khuôn mặt, nhóm được đề xuất thử và chọn một trong các mô hình sau:

- Haar Cascade Classifier via OpenCV.
- Histogram of Oriented Gradients (HOG) via DLIB.
- Deep Neural Network via DLIB.
- Single Shot Detector with ResNet-10 via OpenCV.
- Multi-task Cascaded CNN (MTCNN) via Tensorflow.
- FaceNet MTCNN via Tensorflow.

2.2.2. Nhận diện khuôn mặt



Hình 3. Nhận diện khuôn mặt

Nhận diện khuôn mặt:

- Thư viện dùng để phân tích hình ảnh thành các Vector Embedding là Dlib.
- Dlib: Dlib là một thư viện mạnh mẽ được chấp nhận rộng rãi trong cộng đồng xử lý ảnh tương tự như OpenCV. Các nhà nghiên cứu chủ yếu sử dụng mô-đun phát hiện và căn chỉnh khuôn mặt của nó. Ngoài ra, dlib cũng cung cấp một mô-đun nhận dạng khuôn mặt mạnh mẽ. Mặc dù nó được viết bằng c++, nó cũng có giao diện python.
- Để phân tích vector Embedding bằng Dlib thì cần phải có 2 file cần thiết: “facial landmark detector” và “resnet model”.
- Facial landmark detector: là một file bổ sung của dlib, giống như cái tên thì đây giúp cho máy tính có thể phân biệt được các đặc trưng của khuôn mặt như mắt, mũi, miệng. Từ đó có thể tăng độ chính xác của việc nhận diện khuôn mặt.
- Sau khi đưa các đặc trưng của bức ảnh về các vector embedding thì chúng ta sẽ dùng tới file Resnet Model.
- Khi đưa ra được một mảng bao gồm các vector của các ảnh thì đem so sánh ảnh cần nhận diện với mảng đó.
- Dùng khoảng cách euclid để đo, lấy giá trị nhỏ nhất và xét với ngưỡng xem là giá trị nhỏ nhất đó có dưới ngưỡng không

2.2.3. Server

Server là một trong những thành phần quan trọng của hệ thống. Nhiệm vụ của nó là cung cấp các API cho mobile, ESP8266 cũng như raspberry để thực hiện những chức năng.

Server sử dụng django framework (được xây dựng trên nền python), resful API.

Mô tả chức năng của từng API:

<http://host:port/users/get-user>

- Method: GET
- Body: None
- Chức năng: Xem tất cả những gương mặt hiện có trong database.

<http://host:port/users/get-result>

- Method: GET
- Body: None
- Chức năng: Lấy kết quả từ việc nhận diện gương mặt.

<http://host:port/users/recognize>

- Method: POST
- Body: file
- Chức năng: Nhận diện ảnh được gửi đến từ raspberry.

<http://host:port/users/create>

- Method: POST
- Body: name, imageUrl
- Chức năng: Thêm một người mới vào database.

<http://host:port/users/delete>

- Method: POST
- Body: id
- Chức năng: Xóa một người trong database.

<http://host:port/users/sign-in>

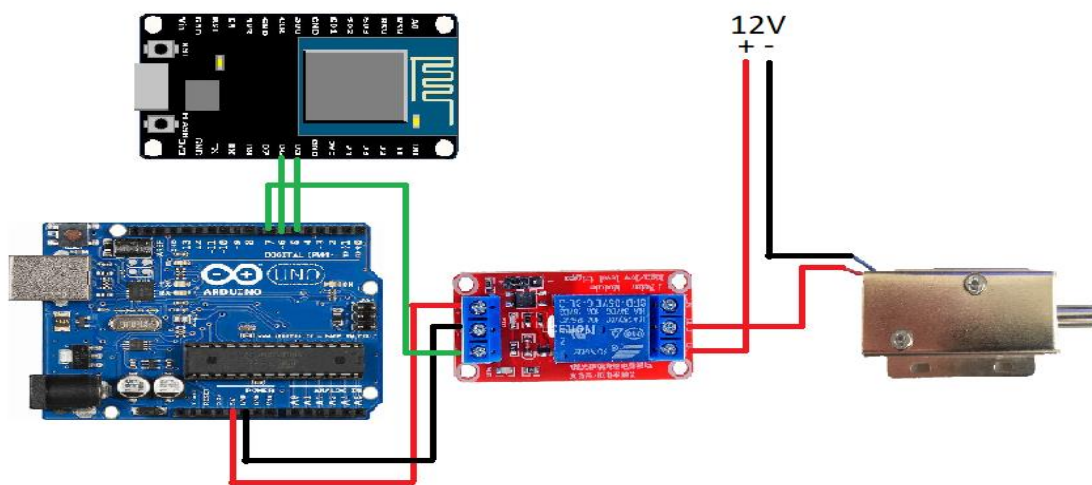
- Method: POST

- Body: email, password
- Chức năng: Đăng nhập vào app mobile.

<http://host:port/users/open-door>

- Method: GET
- Body: None
- Chức năng: Mở cửa.

2.2.4. Điều khiển khoá

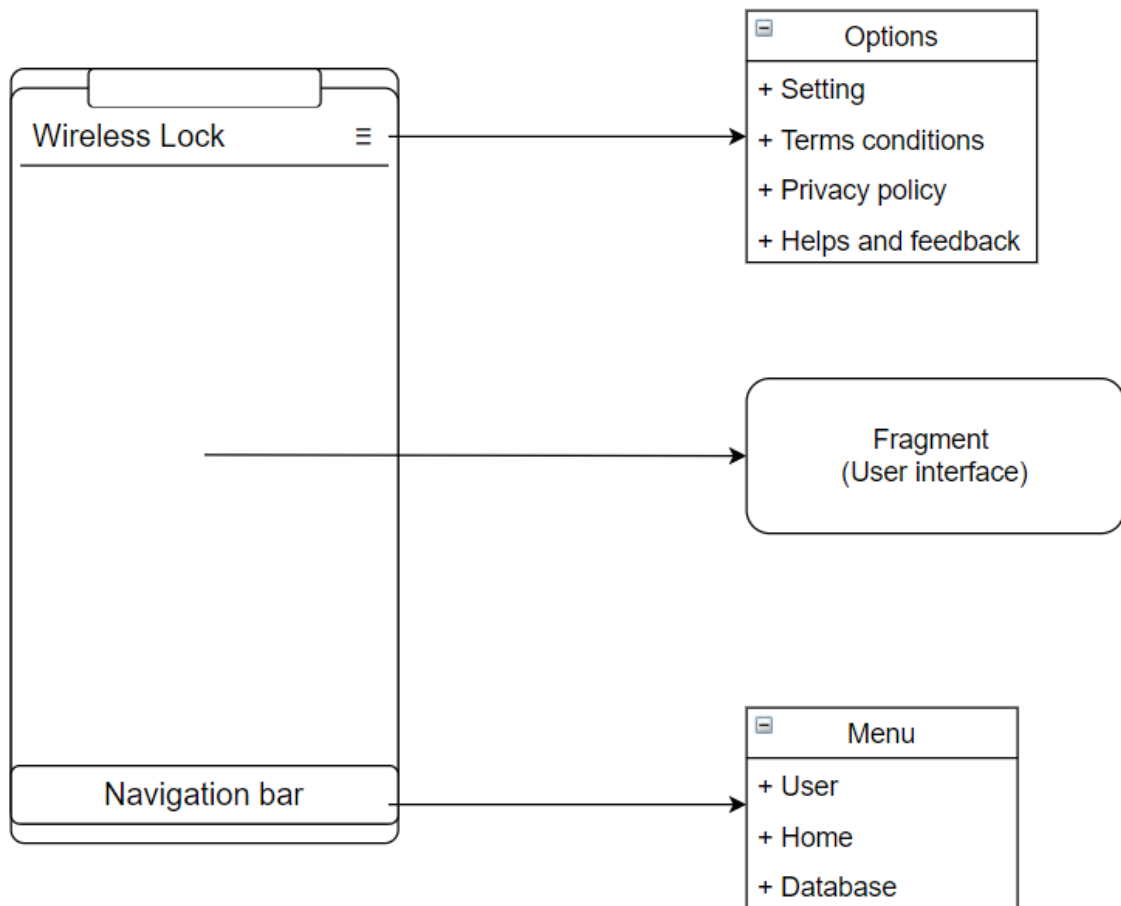


Module wifi sẽ liên lạc với server để nhận lệnh, khi có lệnh mở cửa thì nó sẽ chuyển dữ liệu qua cho arduino, arduino nhận được lệnh đó sẽ điều khiển module rơ le mở cửa. Sau một khoảng thời gian, khóa được điều khiển từ module rơ le sẽ đóng lại.

2.2.5. Mobile app

Được viết trên ngôn ngữ lập trình Java và ngôn ngữ đánh dấu XML .

Các thành phần ứng dụng:



Hình 4. Thành phần ứng dụng

Sử dụng các thư viện chính sau:

- Retrofit (Giao tiếp với Server qua API).
- Gson và RetrofitConverter (Chuyển đổi chuỗi JSON thành GSON để Android xử lý).
- Picasso, Glide (Hiển thị ảnh bằng Url).

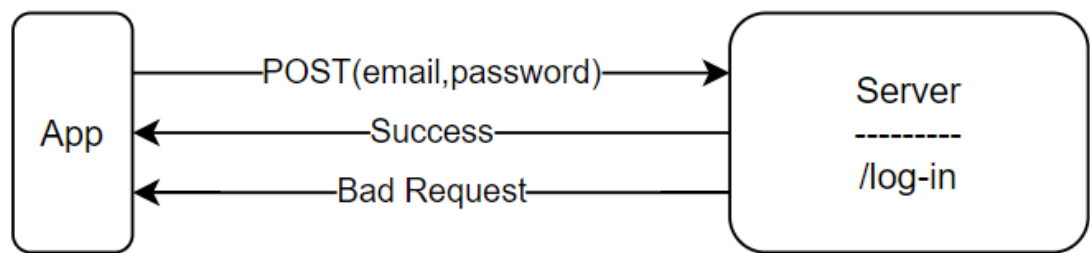
Chức năng:

- Đăng nhập.
- Nhận thông báo khi có người xuất hiện.
- Mở cửa cho người lạ.
- Thêm người lạ vào database.
- Xem người quen trên database.
- Xóa người quen khỏi database.

Cách triển khai chức năng:

- Đăng nhập: Sử dụng method POST để gửi 2 String là email và password cho Server. Nếu Server trả về Success thì tắt activity đăng nhập và thông báo Toast đăng nhập thành công. Nếu Server trả về Bad request thì thông báo Toast đăng nhập thất bại.

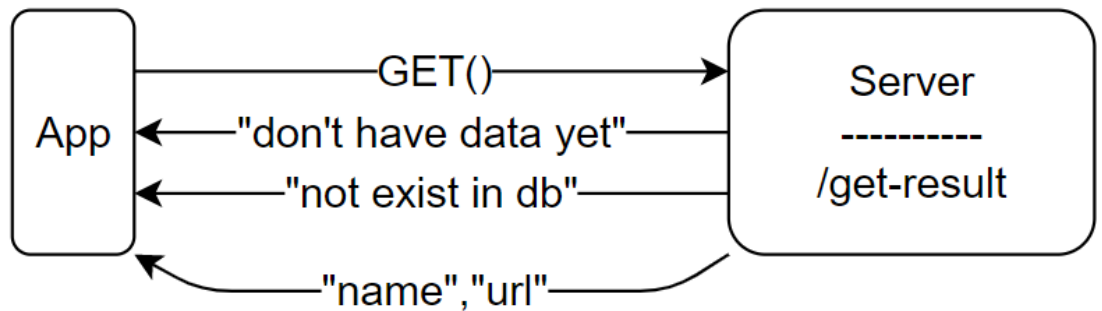
Login



Hình 5. Chức năng đăng nhập

- Thông báo: Tạo một Handler quản lí 1 thread chạy ngầm liên tục gửi yêu cầu GET đến Server. Nhận được phản hồi thì xét xem đó là không có người, người lạ hoặc người quen. Nếu không có người thì không làm gì cả, nếu là người lạ thì thông báo “Có người lạ xuất hiện, vui lòng chọn thao tác của bạn” và hiển thị ảnh của người lạ, nếu là người quen thì thông báo “Có người” + tên người quen “ đến thăm nhà bạn, cửa đang tự động mở” và hiển thị ảnh người quen.

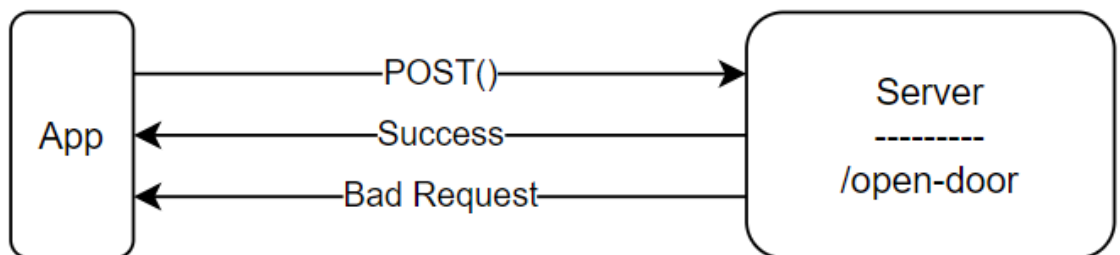
Get Result



Hình 6. Chức năng thông báo

- Mở cửa: Sử dụng method POST gửi đến api “open-door”, nếu nhận được response thì thông báo Toast “Đã mở cửa”.

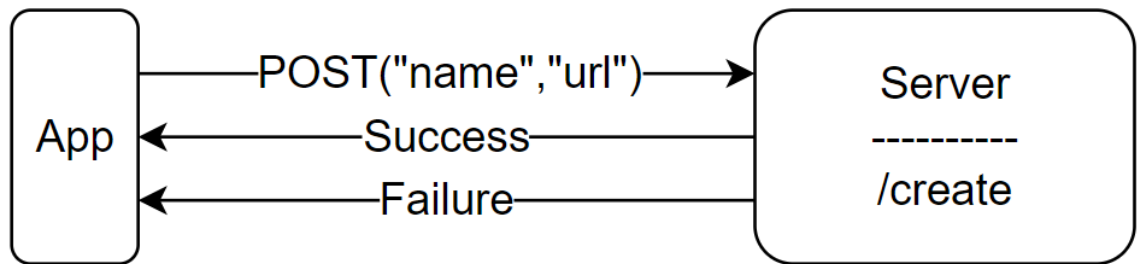
OpenDoor



Hình 7. Chức năng mở cửa

- Thêm người lạ vào database: Sau khi nhận được thông báo có người lạ xuất hiện, click vào Button để gửi POST request chứa tên đến Server.

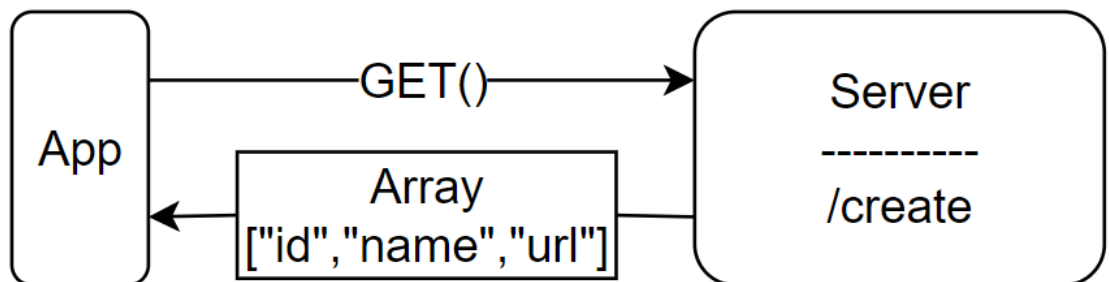
Add user



Hình 8. Chức năng thêm người lạ

- Xem người quen trên database: Chạy method GET khi mở Fragment Database để lấy dữ liệu tất cả người quen từ Server.

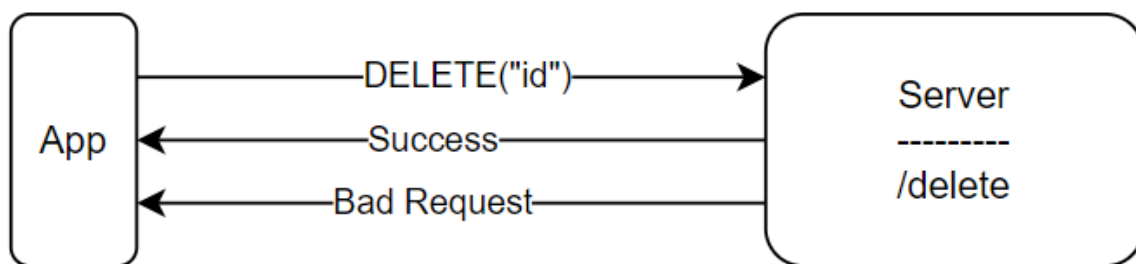
Get User



Hình 9. Chức năng xem người quen

- Xóa người quen khỏi database: Trong danh sách người quen hiển thị trong Fragment Database, sử dụng onItemClickLongClickListener để khi nhấn giữ vào User nào thì sẽ gửi request bằng method DELETE chứa ID của Item đó đến Server để xóa người đó khỏi Database.

DeleteUser



Hình 10. Chức năng xóa người quen

3. Kết quả

3.1. Phát hiện khuôn mặt

Dữ liệu sử dụng: từ trang [Kaggle](#) bao gồm 8196 ảnh và 2 file csv chứa số lượng khuôn mặt trong mỗi bức ảnh.

Kết quả thu được:

Bảng 2. Mô hình phát hiện khuôn mặt

Mô hình	Thời gian xử lý (s)	Độ chính xác (%)
Haar Cascade Classifier via OpenCV	260	10.371
Histogram of Oriented Gradients (HOG) via DLIB	1090	30.1
Deep Neural Network via DLIB	Chưa rõ (khoảng 20s/ảnh => dự kiến khoảng 160000 s)	Chưa rõ
Single Shot Detector with ResNet-10 via OpenCV	226	34.846
Multi-task Cascaded CNN via Tensorflow	3892	50.366

Sau khi thử các mô hình trên, nhóm quyết định sử dụng mô hình “Single Shot Detector with ResNet-10 via OpenCV” để phát hiện khuôn mặt.

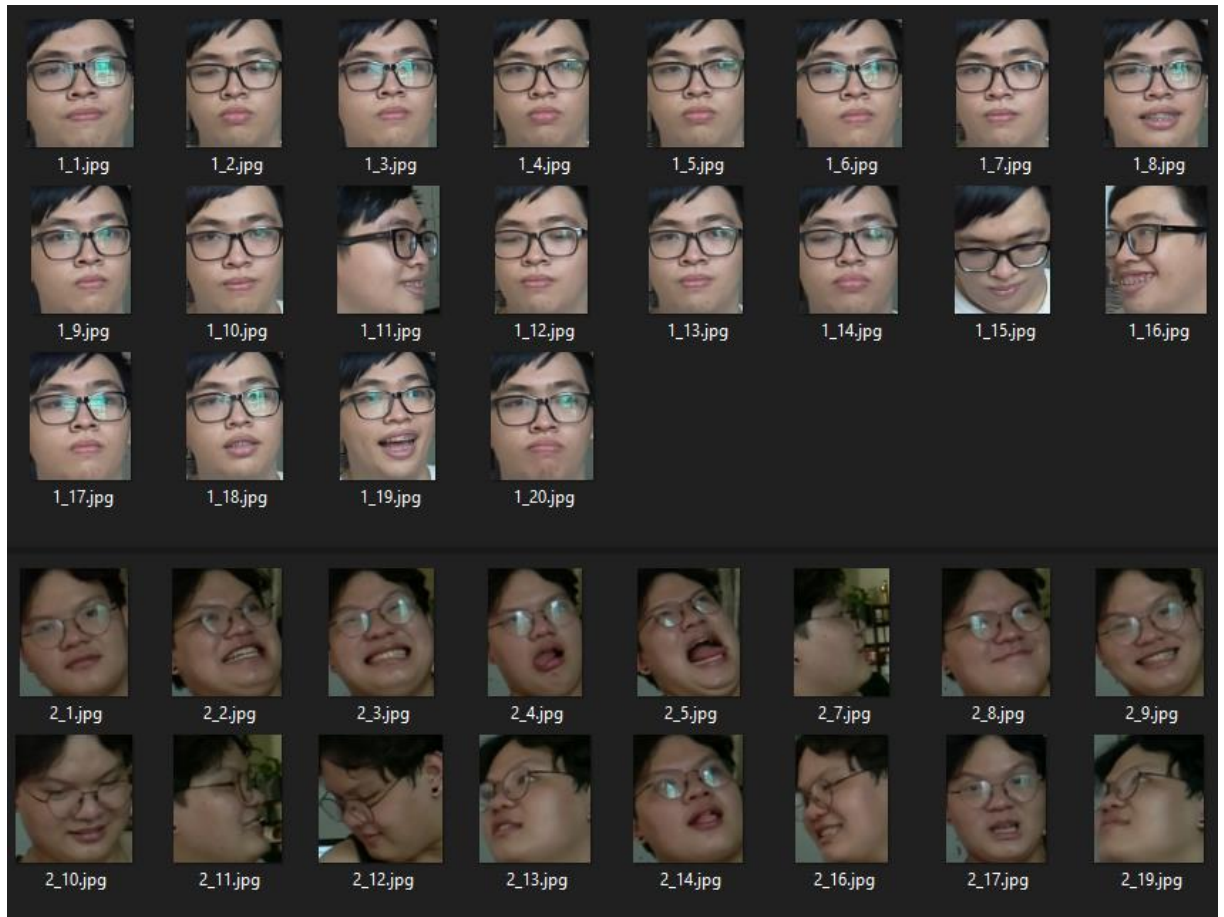
Áp dụng trên Raspberry Pi:

Dữ liệu sử dụng: 20 ảnh được chụp từ PiCamera. Mỗi bức ảnh chứa chính xác 2 khuôn mặt.



Hình 11. Hình ảnh từ PiCamera

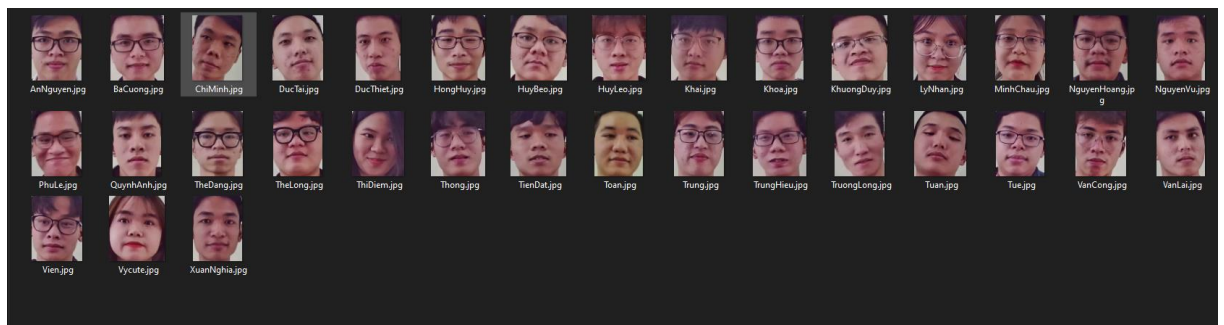
Kết quả thu được: Trên 20 ảnh bức ảnh, có 16 bức ảnh phát hiện đúng số lượng khuôn mặt trong ảnh => Độ chính xác rơi vào khoảng 80% (vì tình hình dịch bệnh phức tạp nên kích thước dữ liệu khá nhỏ nên độ chính xác thực tế có thể khác).



Hình 12. Kết quả sau khi phát hiện khuôn mặt

3.2. Nhận diện khuôn mặt

Dữ liệu tập huấn luyện: 33 ảnh mang đầy đủ đặc trưng của từng người.



Hình 13. Dữ liệu huấn luyện phát hiện khuôn mặt

Kết quả trích xuất đặc trưng: cho ra 33 vector embedding ứng với 33 khuôn mặt.

Dữ liệu thử nghiệm: 36 ảnh (được chụp từ PiCamera và đã xử lý phát hiện khuôn mặt) của 2 người nằm trong tập huấn luyện.

Kết quả thử nghiệm:

0.680638	0.679627
0.619355	0.595392
0.481983	0.524176
0.636866	0.374014
0.693541	0.553874
0.654418	0.654779
0.304146	0.307677
0.60043	0.60284
0.674446	0.321336
0.658323	0.608799
0.676352	0.574668
0.659388	0.611975
0.712037	0.606536
0.694018	0.343689
0.687948	0.593247
0.683879	0.653961
0.659361	0.537912
0.623733	
0.681751	
0.687028	
0.638482	

Hình 14. Kết quả thử nghiệm phát hiện khuôn mặt

Độ chính xác của việc nhận diện khuôn mặt rơi vào khoảng 50-70% (vì tình hình dịch bệnh phức tạp nên chỉ có thể thử ở 2 người trong nhóm nên độ chính xác thực tế có thể khác).

3.3. Server

Kết quả cho thấy các API của server chạy đúng như mô tả.

3.4. Điều khiển khoá

Chạy thử nghiệm cho thấy khóa hoạt động đúng theo lý thuyết đã trình bày.

```
ESP8266HTTPClient
[08/Jun/2021 12:33:38] "GET /users/open-door HTTP/1.1" 200 34
```

Hình 15. Module wifi gọi API open-door của server



Hình 16. Khóa được mở

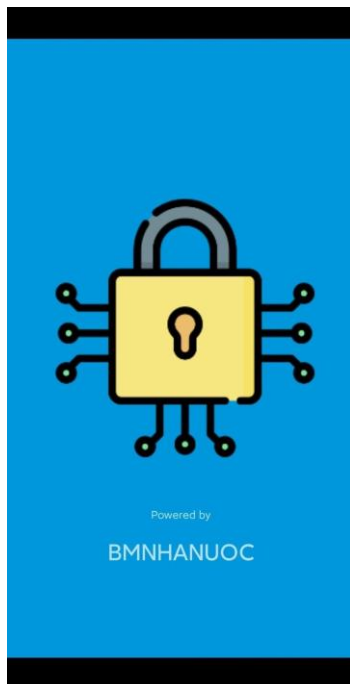
Sau một khoảng thời gian, khóa sẽ tự động đóng lại:



Hình 17. Khóa tự động đóng

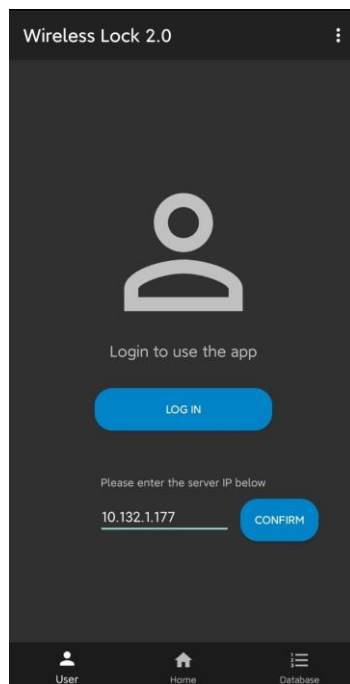
3.5. Mobile app

Khi mở app, sẽ hiện ra màn hình khởi động.



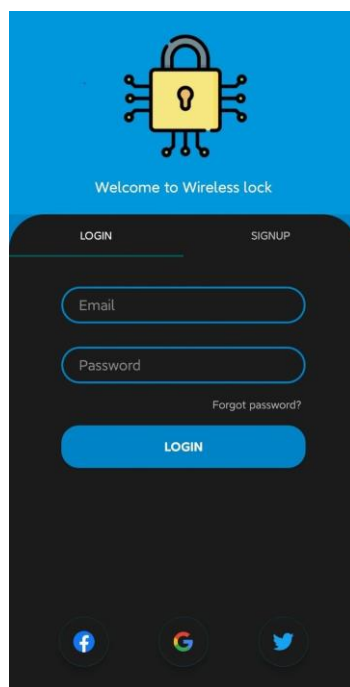
Hình 18. Giao diện khởi động

Sau khi khởi động, app sẽ chuyển đến màn hình nhập IP Server.



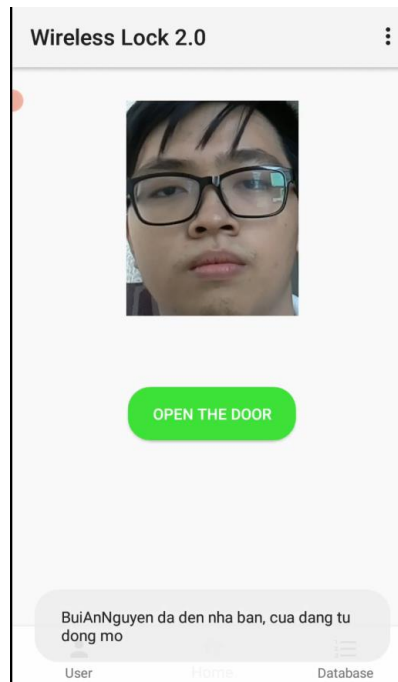
Hình 19. Giao diện nhập ip server

Sau khi nhập IP, app sẽ chuyển đến màn hình đăng nhập.



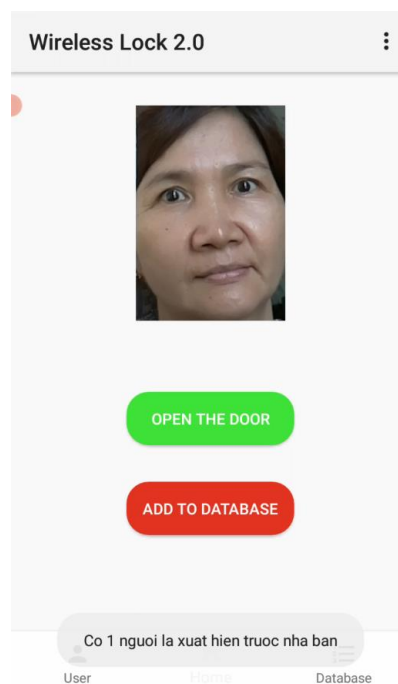
Hình 20. Giao diện đăng nhập

Sau khi đăng nhập thành công, app sẽ chuyển đến giao diện chính (nơi sẽ thông báo có người trước cửa)



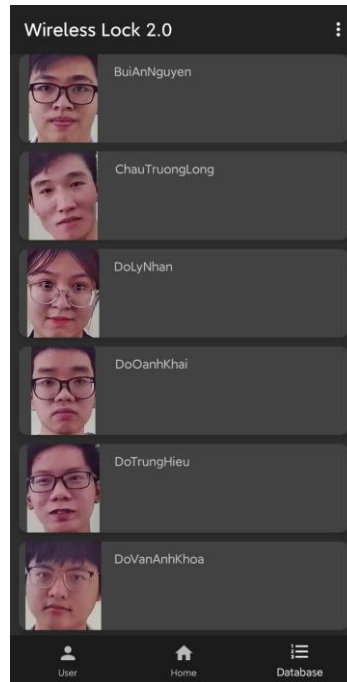
Hình 21. Giao diện khi xuất hiện người quen

Khi có người lạ, app sẽ hiện một EditText để nhập tên của người lạ và nút “ADD TO DATABASE” để thêm người đó vào database.



Hình 22. Giao diện khi xuất hiện người lạ

Ở giao diện chính, có thể chọn mục Database để xem danh sách người quen trong database.



Hình 23. Giao diện xem người quen

4. Kết luận

4.1. Đánh giá

Qua quá trình thực hiện đồ án, hệ thống đã đạt được một phần các yêu cầu đề ra:

- Về phát hiện và nhận diện khuôn mặt: Hệ thống đã nhận diện khá chính xác các khuôn mặt của người trong database cũng như các khuôn mặt của người lạ, tuy nhiên tùy thuộc vào điều kiện ánh sáng thì kết quả sẽ có một số sai sót.
- Về mobile app: App sở hữu một giao diện thân thiện và dễ sử dụng, đảm bảo các chức năng đặt ra.

4.2. Hướng phát triển

Để hệ thống hoạt động một cách chính xác hơn, ta có thể:

- Nâng cấp camera bằng cách thay một camera có độ phân giải cao hơn.
- Phát triển model nhận diện khuôn mặt để nâng cao độ chính xác.

5. Danh mục tài liệu tham khảo

Mô hình phát hiện và nhận diện khuôn mặt

<https://github.com/richmondu/libfaceid>

Mobile app

<https://github.com/bumptechnology/glide>

<https://square.github.io/picasso/>

<https://square.github.io/retrofit/>

<https://github.com/square/retrofit/tree/master/retrofit-converter/gson>

<https://github.com/google/gson>

Django Server

<https://docs.djangoproject.com/en/3.2/>

Hardware

[https://randomnerdtutorials.com/esp8266-nodemcu-http-get-post-](https://randomnerdtutorials.com/esp8266-nodemcu-http-get-post-arduino/?fbclid=IwAR15zujpzjqBI6ndNopUzbH6ri6u1kEj7efxLaOGNV7ZBEQtsbWDRADcEL0)

[arduino/?fbclid=IwAR15zujpzjqBI6ndNopUzbH6ri6u1kEj7efxLaOGNV7ZBEQts](https://randomnerdtutorials.com/esp8266-nodemcu-http-get-post-arduino/?fbclid=IwAR15zujpzjqBI6ndNopUzbH6ri6u1kEj7efxLaOGNV7ZBEQtsbWDRADcEL0)
[bWDRADcEL0](https://randomnerdtutorials.com/esp8266-nodemcu-http-get-post-arduino/?fbclid=IwAR15zujpzjqBI6ndNopUzbH6ri6u1kEj7efxLaOGNV7ZBEQtsbWDRADcEL0)

<https://randomnerdtutorials.com/guide-for-relay-module-with-arduino/>