**TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM  
KHOA CÔNG NGHỆ THÔNG TIN**

**-----\*\*\*-----**

****

**ĐỒ ÁN THỰC TẬP**

***Đề tài:***

***NGHIÊN CỨU HỆ THỐNG IOT, TRÍ TUỆ NHÂN TẠO VÀ XÂY DỰNG HỆ THỐNG CỬA TỰ ĐỘNG THÔNG MINH***

***GVHD: TS.Phạm Trung Minh***

***Sinh viên thực hiện: Đinh Văn Khoa – Mã sv: 78479***

***Vũ Tuấn Hùng – Mã sv: 78238***

***Lê Gia An – Mã sv: 77004***

***Hải Phòng, tháng 03 năm 2022*TRƯỜNG ĐẠI HỌC HÀNG HẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN THỰC TẬP**

**-----\*\*\*-----**

1. **Tên đề tài**

**Nghiên cứu hệ thống iot, trí tuệ nhân tạo và xây dựng hệ thống cửa tự động thông minh.**

1. **Công việc cần thực hiện**

**Nhiệm vụ sinh viên 1:**

* Nghiên cứu, thiết kế, chế tạo mô hình cửa tự động sử dụng vi điều khiển và các cảm biến phù hợp với mục tiêu của đề tài.

**Nhiệm vụ sinh viên 2:**

* Nghiên cứu, cà đặt mô hình huấn luyện nhận dạng khuôn mặt con người.

**Nhiệm vụ sinh viên 3:**

* Nghiên cứu, xây dựng hệ thống server điều khiển, giám sát hoạt động của cửa tự động.

1. **Ngôn ngữ, thiết bị, công cụ, nền tảng lập trình**

* **Ngôn ngữ lập trình:** C++, Python, Nodejs, SQL,...
* **Công cụ lập trình:** Arduino IDE, Pycharm Community Edition,...
* **Thiết bị:** Mạch vi xử lý Arduino Uno R3, mạch NodeMCU ESP8266, mạch điều khiển động cơ ULN2003, Step motor 28BYJ-48, cảm biến sóng siêu âm HC-SR04, Adapter power 5-12V,...

|  |  |
| --- | --- |
|  | ***Hải phòng, tháng 03 năm 2022*** |

# MỤC LỤC

[MỤC LỤC i](#_Toc99394793)

[DANH MỤC CÁC HÌNH VẼ iii](#_Toc99394794)

[CHƯƠNG 1: TỔNG QUAN BÀI TOÁN 1](#_Toc99394795)

[1.1 Mô tả bài toán 1](#_Toc99394796)

[1.1.1 Mục đích, yêu cầu 1](#_Toc99394797)

[1.1.2 Mục tiêu cần đặt được 1](#_Toc99394798)

[1.1.3 Các phân hệ của hệ thống 1](#_Toc99394799)

[1.2 Chi tiết từng phân hệ 1](#_Toc99394800)

[1.2.1 Hệ thống điều khiển hoạt động đóng/mở cửa 1](#_Toc99394801)

[1.2.2 Hệ thống máy chủ Web 2](#_Toc99394802)

[1.2.3 Hệ thống bảo mật nhận dạng khuôn mặt 2](#_Toc99394803)

[CHƯƠNG 2: THIẾT KẾ CHẾ TẠO BỘ ĐIỀU KHIỂN CỬA 3](#_Toc99394804)

[2.1 Sơ đồ hệ thống 3](#_Toc99394806)

[2.2 Thuật toán 4](#_Toc99394807)

[2.3 Linh kiện, thiết bị 9](#_Toc99394808)

[CHƯƠNG 3: XÂY DỰNG SERVER WEB 10](#_Toc99394809)

[3.1 Sơ đồ hệ thống 10](#_Toc99394811)

[3.2 Nguyên lý hệ thống Server 10](#_Toc99394812)

[3.2.1 Chế độ thủ công 10](#_Toc99394813)

[3.2.2 Chế độ tự động 12](#_Toc99394814)

[3.2.3 Chế độ nhận diện khuôn mặt 13](#_Toc99394815)

[3.3 Giao diện chương trình 15](#_Toc99394816)

[CHƯƠNG 4: HỆ THỐNG NHẬN DẠNG KHUÔN MẶT 17](#_Toc99394817)

[4.1 Cơ sở lý thuyết 17](#_Toc99394818)

[4.1.1 MTCNN (Multi-task Cascaded Convolutional Networks) 17](#_Toc99394819)

[4.1.2 FaceNet 23](#_Toc99394820)

[4.1.3 SVM (Support vector machines) 26](#_Toc99394821)

[4.2 Dữ liệu thu thập 27](#_Toc99394822)

[4.3 Các mô hình được sử dụng trong hệ thống 27](#_Toc99394823)

[4.4 Tham chiếu 27](#_Toc99394824)

[KẾT LUẬN 28](#_Toc99394825)

[TÀI LIỆU THAM KHẢO 29](#_Toc99394826)

# DANH MỤC CÁC HÌNH VẼ

[Hình 3.1‑1 Sơ đồ hệ thống server web 10](#_Toc99394780)

[Hình 3.3‑1 Giao diện web 15](#_Toc99394781)

[Hình 3.3‑2 Giao diện điều khiển 16](#_Toc99394782)

[Hình 3.3‑1 Quy trình cơ bản của hệ thộng nhận dạng khuôn mặt 17](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394783)

[Hình 4.1‑1 Cấu trúc mạng MTCNN 18](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394784)

[Hình 4.1‑2 Kim tự tháp ảnh 18](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394785)

[Hình 4.1‑3 P-Net 19](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394786)

[Hình 4.1‑4 R-Net 20](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394787)

[Hình 4.1‑5 O-Net 21](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394788)

[Hình 4.1‑6 Cấu trúc mạng FaceNet 24](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394789)

[Hình 4.1‑7 Trích xuất Embedding khuôn mặt bằng FaceNet Model 25](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394790)

[Hình 4.1‑8 4 Tìm khuôn mặt gần nhất với khuôn mặt đã cho trong số 4 hình ảnh dựa trên Embedding của chúng 25](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394791)

[Hình 4.1‑9 Thêm khuôn mặt của nhân viên mới Embedding vào tập dữ liệu hiện có 26](file:///D:\OneDrive\Máy%20tính\Report.docx#_Toc99394792)

# CHƯƠNG 1: TỔNG QUAN BÀI TOÁN

## Mô tả bài toán

### Mục đích, yêu cầu

* Thiết kế, chế tạo mô hình cửa tự động có các chế độ đống mở khác nhau: đóng mở cửa tự động, đóng mở thủ công, đóng mở bảo mật nhận dạng khuôn mặt,
* Áp dụng kiến thức công nghệ IoT thiết kế hệ thống máy chủ, ứng dụng web điều khiển, giám sát tình trạng hoạt động của cửa tự động thông qua mạng internet toàn cầu.
* Áp dụng kiến thức trí tuệ nhân tạo, nghiên cứu công nghệ nhận dạng mặt người nhằm xác định, xác thực đối tượng được phép đi qua cửa tự động.
* Kết nối các phân hệ lại để tạo nên một hệ thống hoàn chỉnh.

### Mục tiêu cần đặt được

### Các phân hệ của hệ thống

*Hệ thống bao gồm 3 phân hệ chính:*

* Bộ công cụ điều khiển cửa.
* Hệ thống máy chủ Web.
* Hệ thống bảo mật nhận dạng khuôn mặt.

## Chi tiết từng phân hệ

### Hệ thống điều khiển hoạt động đóng/mở cửa

Bao gồm:

* Các mạch vi điều khiển: Arduino Uno R3, NodeMCU ESP8266, ULN2003,
* Động cơ DC: Step motor 28BYJ-48
* Cảm biến: Cảm Biến Siêu Âm HC-SR04
* Các linh kiện phụ trợ khác: Nút nhấn, điện trở, dây dẫn,...

Hệ thống có chức năng thu nhận tin hiệu điều khiển từ phía máy chủ Web yêu cầu được gửi thông qua mạng internet để điều khiển các hoạt động đóng/mở cửa. Đồng thời phản hồi các trạng thái đóng/đóng mở cửa tới máy chủ Web.

### Hệ thống máy chủ Web

Bao gồm: Máy chủ, brower, database.

Máy chủ đảm nhiệm chức năng điều hướng, kết nối giữa các phân hệ, truy vấn database và hoạt động điều khiển đóng mở, giám sát của người dùng đối với hệ thống thông qua giao diện phía brower.

### Hệ thống bảo mật nhận dạng khuôn mặt

Hệ thống sử dụng các mô hình thuật toán AI – Trí tuệ nhân tạo:

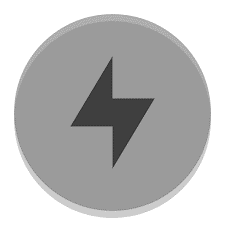
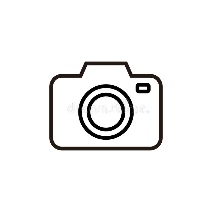
* **Model MTCNN** (Multi-task Cascaded Convolutional Networks): Phát hiện một hoặc nhiều khuôn mặt trong hình ảnh, căn chỉnh khuôn mặt đã được phát hiện.
* **Model FaceNet**: Trích xuất các tính năng quan trọng nhất từ hình ảnh của khuôn mặt.
* **Model SVM** (Support Vector Machine): Phân loại khuôn mặt dựa trên các đặc điểm đã trích xuất.

Hệ thống nhận dạng có chức năng nhận dạng đối tượng đã tồn tại trong tập huấn luyện sau đó phản hồi tín hiệu xác minh tới máy chủ để điều khiển hoạt động đóng/mở cửa.

# CHƯƠNG 2: THIẾT KẾ CHẾ TẠO BỘ ĐIỀU KHIỂN CỬA



## Sơ đồ hệ thống



Sensor

Motor

Node MCU ESP8266

ARDUINO UNO R3

## Thuật toán

**Lập trình Arduino:**

***Mục tiêu****: lập trình Arduino để có thể điều khiển được motor và nhận tín hiệu từ cảm biến.*

**\_ Điều khiển motor:**

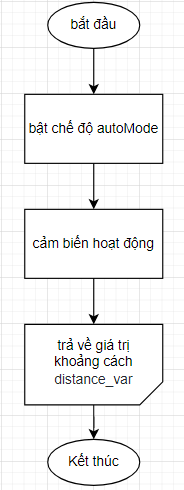
Arduino IDE có một thư viện hỗ trợ điều khiển động cơ bước tên là Stepper Motor, nó cực kì dễ sử dụng, chỉ việc kết nối như trên rồi nạp code.

Arduino có hỗ trợ sẵn bộ thư viện stepper trong file example. Nhưng thử thì thư viện đó bị delay. Nghĩa là trong quá trình động cơ bước hoạt động thì cả chương trình sẽ phải dừng đợi nó hoạt động theo (cơ chế này gọi là cơ chế đồng bộ - sync).

Để điều khiển được nhiều động cơ bước thì chúng ta phải biết ngõ xử lý bất đồng bộ, nghĩa là sao? Nghĩa là phải biết thay hàm delay bằng cách khác, và đó chính là dùng timer, chính là dùng hàm millis(). Và ý tưởng đó đã được thể hiện trong thư viện AccelStepper, thư viện này đã ứng dụng nguyên lý xử lý bất đồng bộ (async) để làm nên nó - một thư viện cực kì hữu ích!

**\_ Hàm nhận tín hiệu từ cảm biến:**

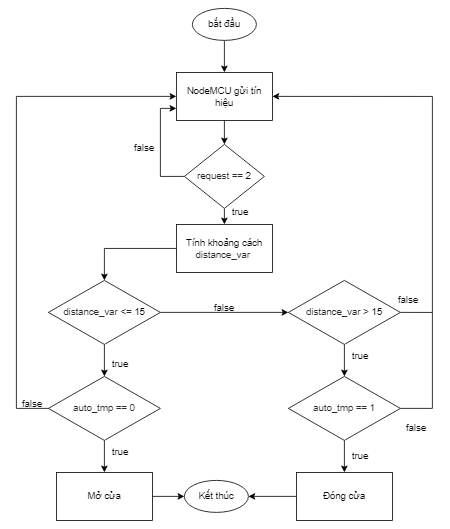
Cảm biến khoảng cách siêu âm HC-SR04 được sử dụng rất phổ biến để xác định khoảng cách vì RẺ và CHÍNH XÁC. Cảm biến sử dụng sóng siêu âm và có thể đo khoảng cách trong khoảng từ 2 -> 300 cm, với độ chính xác gần như chỉ phụ thuộc vào cách lập trình.



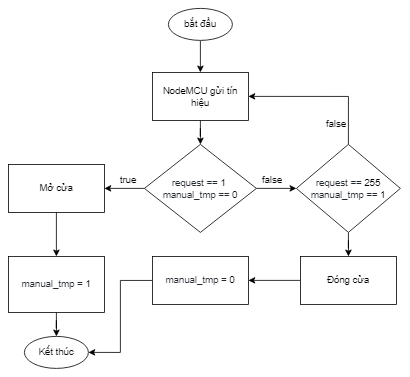
**Nguyên lý hoạt động:** Để đo khoảng cách, ta sẽ phát 1 xung rất ngắn (5 microSeconds - ú) từ chân **Trig.**Sau đó, cảm biến sẽ tạo ra 1 xung HIGH ở chân **Echo** cho đến khi nhận lại được sóng phản xạ ở pin này. Chiều rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biển và quay trở lại.

Tốc độ của âm thanh trong không khí là 340 m/s (hằng số vật lý), tương đương với 29,412 microSeconds/cm (106 / (340\*100)). Khi đã tính được thời gian, ta sẽ chia cho 29,412 để nhận được khoảng cách.

\_ **Hàm autoMode**:

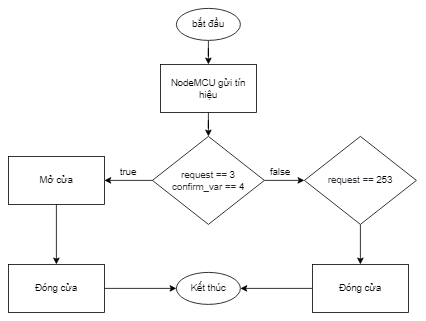


**Nguyên lý hoạt động:** Người dùng sẽ kích hoạt chế độ autoMode bằng nút bấm trên Control Interface. Khi nhấn nút On/Off autoMode, Control Interface sẽ truyền tín hiệu điều khiển tới WebServer, WebServer gửi tín hiệu điều khiển tới NodeMCU qua Internet và NodeMCU thông qua giao thức I2C gửi cho Arduino. Khi Arduino nhận được tín hiệu từ NodeMCU, Arduino bắt đầu so sánh giá trị mà NodeMCU gửi về. Nếu giá trị gửi về là request == 2 (2 là giá trị để kích hoạt chế độ làm việc auto), Arduino sẽ gửi tín hiệu đến cảm biến để thực hiện đo khoảng cách với vật cản. Nếu khoảng cách nhỏ hơn 15 thì tiếp tục so sánh giá trị auto\_tmp có bằng 0 hay không (giá trị 0 là cửa đang trạng thái đóng còn 1 là cửa đang trạng thái mở), nếu bằng 0 thì thực hiện hành động mở cửa.

**\_ Hàm manual:**

**Nguyên lý hoạt động:** Người dùng sẽ điều khiển mở và đóng cửa bằng nút bấm trên Control Interface.Khi nhấn nút On/Off Manual Mode, Control Interface sẽ truyền tín hiệu điều khiển tới WebServer, WebServer gửi tín hiệu điều khiển tới NodeMCU qua Internet và NodeMCU thông qua giao thức I2C gửi cho Arduino. Arduino bắt đầu phân tích tín hiệu từ NodeMCU gửi về, so sánh giá trị (manual\_tmp == 0 là trạng thái cửa đóng và manual\_tmp ==1 là trạng thái cửa mở). Từ đó điều khiển Step Motor khiến cửa đóng hay mở.

**\_ Hàm faceRecognitionMode:**



**Nguyên lý hoạt động:** Người dùng sẽ kích hoạt chế độ faceRecognitionMode bằng nút bấm trên Control Interface.Khi nhấn nút On/Off faceRecognitionMode, Control Interface sẽ truyền tín hiệu điều khiển tới WebServer, WebServer gửi tín hiệu điều khiển tới NodeMCU qua Internet và NodeMCU thông qua giao thức I2C gửi cho Arduino. Khi Arduino nhận được tín hiệu từ NodeMCU, Arduino bắt đầu so sánh giá trị mà NodeMCU gửi về. Với giá trị request == 3 (3 là giá trị để kích hoạt chế độ faceRecognition Mode) và confirm\_var == 3, cửa sẽ tự động mở ra và đóng lại trong một khoảng thời gian nhất định.

## Linh kiện, thiết bị

* Cảm biến siêu âm HC-SR04[1]
* Động Cơ Bước 28BYJ-48[2]
* Mạch điều khiển động cơ ULN-2003[3]
* Arduino UNO R3[4]
* Node MCU ESP8266[5]
* Camera Hikvision DS-U02 full HD 1080P/mic[6]

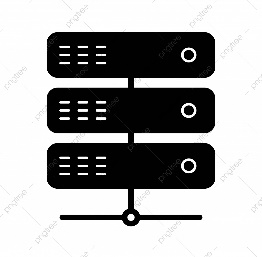
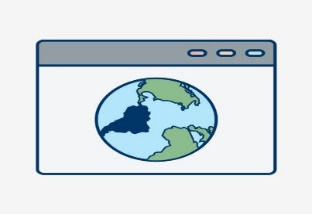
# CHƯƠNG 3: XÂY DỰNG SERVER WEB



## Sơ đồ hệ thống

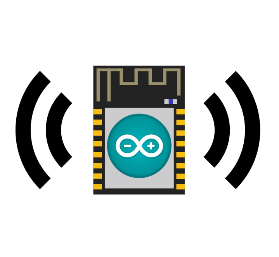
Hệ thống nhận diện khuôn mặt

Phản hồi từ NodeMCU



Server

Database



NodeMCU

Open

Close

Hình ‑ Sơ đồ hệ thống server web

## Nguyên lý hệ thống Server

Server được xây dựng bằng framework NodeJS[1] theo mô hình MVC[2] (Model – Views – Controller). Hệ thống điều khiển có 3 chế độ: Manual, Auto và Face Recognition. Server thực hiện trao đổi dữ liệu trực tiếp với NodeMCU qua giao thức WebSocket[3], ngoài ra có giao tiếp với hệ thống nhận dạng khuôn mặt để điều khiển hệ thống bằng chế độ Face Recognition. Với mỗi lần đóng mở cửa, server sẽ cập nhật cơ sở dữ liệu và hiển thị dữ liệu đó trên view Index (ID, Status, Time). Toàn bộ ứng dụng được triển khai trên Heroku[4] ở địa chỉ <http://autodoor.herokuapp.com/>.

### Chế độ thủ công

Browser

Server

Database

NodeMCU

Manual Mode On/Off

open/close successfully

open/close error

MANUAL\_ON/OFF

open/close successfully

open/close error

open/close/error

1. Truyền dữ liệu từ Server đến NodeMCU

Khi button switch MANUAL MODE thay đổi (do điều khiển của người dùng) thì sẽ xảy ra sự kiện:

* ON: Server sẽ gửi chuỗi “MANUAL\_ON” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại
* OFF: Server sẽ gửi chuỗi “MANUAL\_OFF” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại

1. Nhận dữ liệu từ NodeMCU đến Server

NodeMCU sẽ truyền 1 chuỗi thông báo tình trạng của cửa (open successfully/ close successfully/ open error/ close error) qua giao thức WebSocket đến Server và ứng với mỗi chuỗi, Server sẽ hiển thị trạng thái cửa trên giao diện View ControllPanel ( Open successfully/ Close successfully/ Open Error/ Close Error).

### Chế độ tự động

Browser

Server

Database

NodeMCU

Auto Mode On/Off

open/close successfully

open/close error

AUTO\_ON/OFF

open/close successfully

open/close error

open/close/error

1. Truyền dữ liệu từ Server đến NodeMCU

Khi button switch AUTO MODE thay đổi (do điều khiển của người dùng) thì sẽ xảy ra sự kiện:

* ON: Server sẽ gửi chuỗi “AUTO\_ON” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại
* OFF: Server sẽ gửi chuỗi “AUTO\_OFF” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại

1. Nhận dữ liệu từ NodeMCU đến Server

NodeMCU sẽ truyền 1 chuỗi thông báo tình trạng của cửa (open successfully/ close successfully/ open error/ close error) qua giao thức WebSocket đến Server và ứng với mỗi chuỗi, Server sẽ hiển thị trạng thái cửa trên giao diện View ControllPanel ( Open successfully/ Close successfully/ Open Error/ Close Error).

### Chế độ nhận diện khuôn mặt

Browser

Server

Database

NodeMCU

Face Recognition On/Off

open/close successfully

open/close error

FACE\_RECOGNITION \_ON/OFF

open/close successfully

open/close error

open/close/error

Face Recognition System

FACE\_RECOGNITION\_CONFIRM

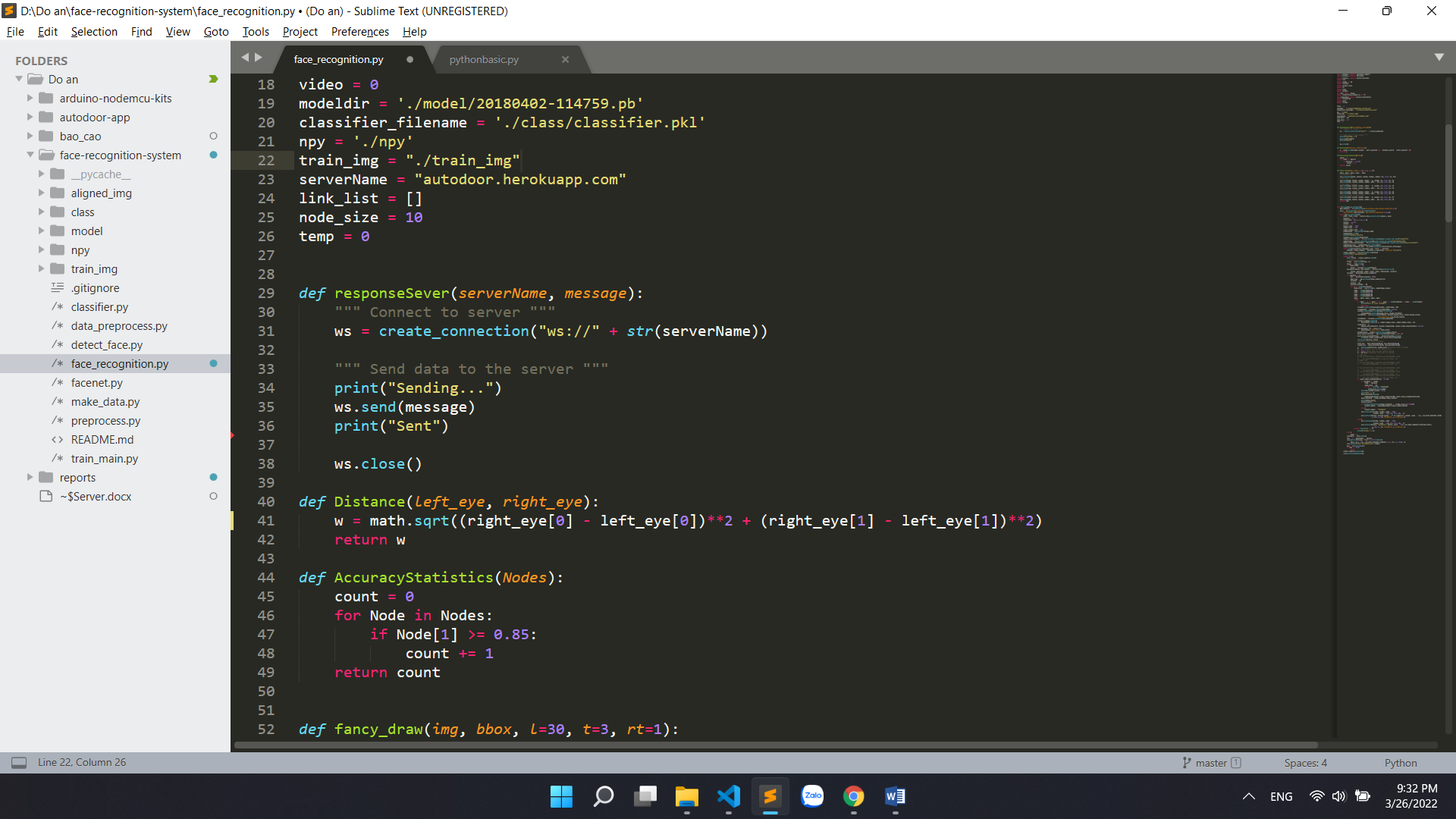
1. Truyền dữ liệu từ Server đến NodeMCU

Khi button switch FACE RECOGNITION MODE thay đổi (do điều khiển của người dùng) thì sẽ xảy ra sự kiện:

* ON: Server sẽ gửi chuỗi “FACE\_RECOGNITION\_ON” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại
* OFF: Server sẽ gửi chuỗi “FACE\_RECOGNITION\_OFF” cho NodeMCU đồng thời vô hiệu hóa 2 button switch của 2 chế độ còn lại

1. Truyền dữ liệu từ hệ thống nhận diện khuôn mặt đến Server và NodeMCU

Hệ thống nhận diện khuôn mặt sẽ khởi tạo 1 kết nối đến Server:



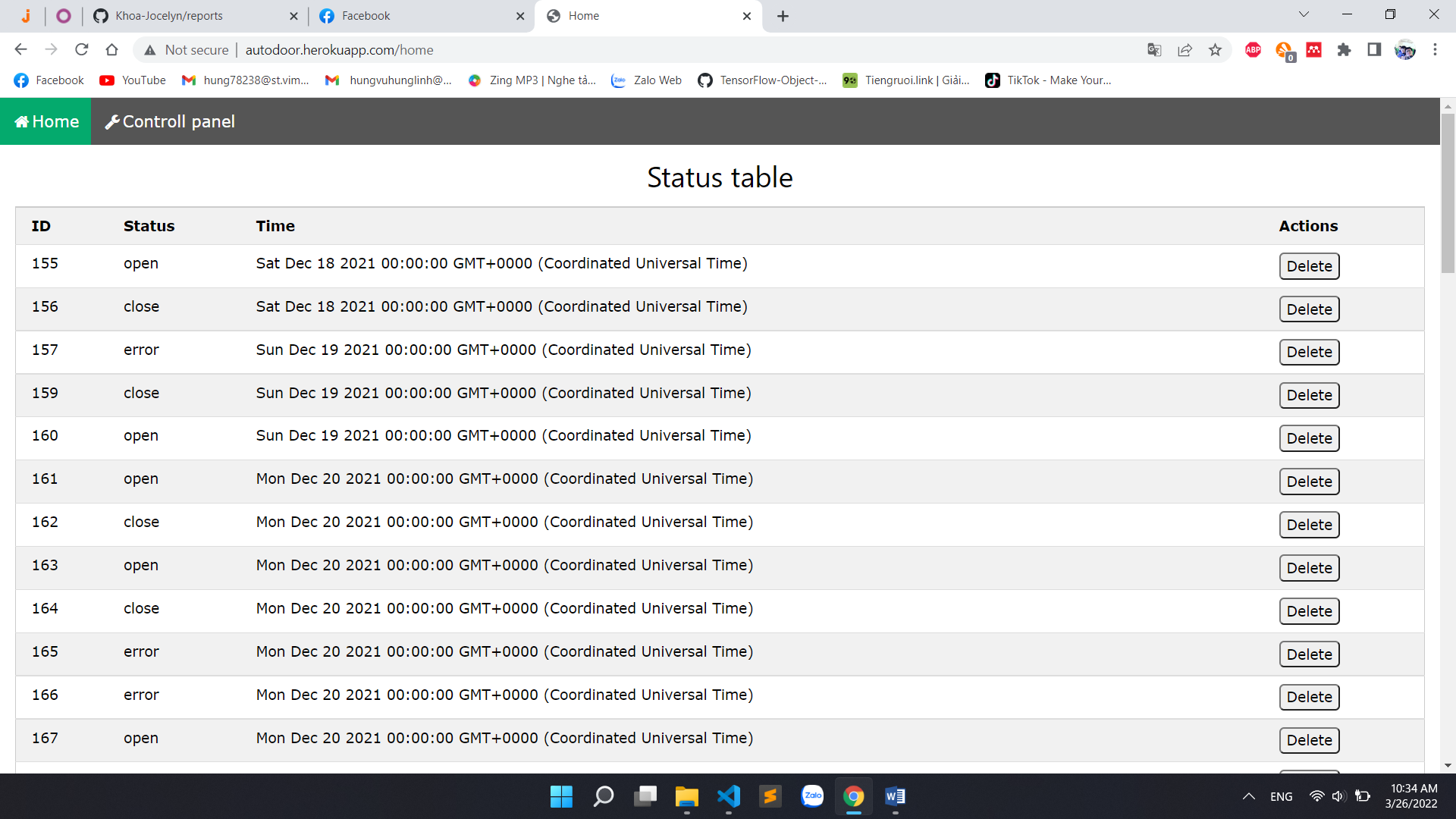
sau khi nhận hình ảnh từ Webcam gửi về sẽ đối chiếu với đối tượng có trong bộ dữ liệu đã train, nếu có tức nhận diện được đối tượng thì hệ thống nhận diện khuôn mặt (Face Recognition System) sẽ gửi message là một chuỗi “FACE\_RECOGNITION\_CONFIRM” đến Server và truyền thẳng đến NodeMCU để NodeMCU thực hiện điều khiển Arduino UNO đóng mở cửa.

1. Nhận dữ liệu từ NodeMCU đến Server

NodeMCU sẽ truyền 1 chuỗi thông báo tình trạng của cửa (open successfully/ close successfully/ open error/ close error) qua giao thức WebSocket đến Server và ứng với mỗi chuỗi, Server sẽ hiển thị trạng thái cửa trên giao diện View ControllPanel ( Open successfully/ Close successfully/ Open Error/ Close Error).

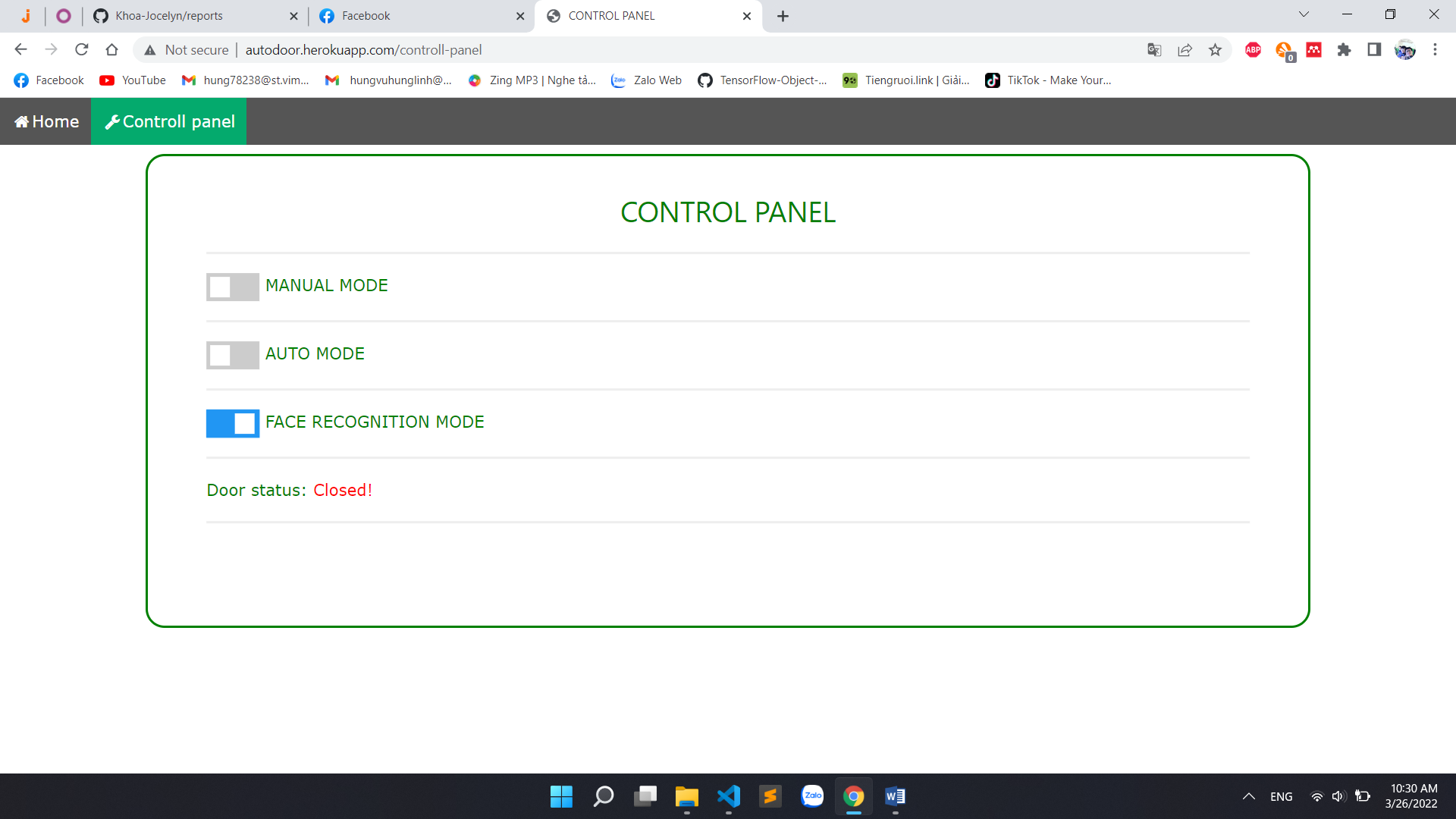
## Giao diện chương trình

Giao diện trang Home (Dữ liệu trạng thái cửa sẽ được hiển thị ở trang này)



Hình ‑ Giao diện web

Giao diện trang Controll panel (Trung tâm điều khiển hệ thống cửa)



Hình ‑ Giao diện điều khiển

# CHƯƠNG 4: HỆ THỐNG NHẬN DẠNG KHUÔN MẶT

Hình ‑ Quy trình cơ bản của hệ thộng nhận dạng khuôn mặt

* **Face detection**: Phát hiện một hoặc nhiều khuôn mặt trong hình ảnh.
* **Face Alignment**: Căn chỉnh khuôn mặt đã được phát hiện.
* **Feature extraction**: Trích xuất các tính năng quan trọng nhất từ hình ảnh của khuôn mặt.
* **Face classification**: Phân loại khuôn mặt dựa trên các đặc điểm đã trích xuất.

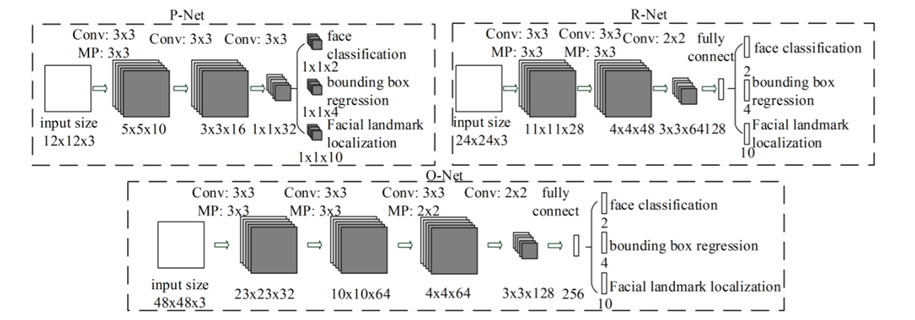
Có nhiều cách khác nhau để thực hiện từng bước trong quy trình nhận dạng khuôn mặt. Tuy nhiên, chúng ta sẽ tập trung vào các phương pháp học sâu phổ biến trong đó chúng ta thực hiện phát hiện khuôn mặt bằng **MTCNN**, trích xuất tính năng bằng **FaceNet** và phân loại Softmax bằng bộ phân loại **SVM**.

## Cơ sở lý thuyết

### MTCNN (Multi-task Cascaded Convolutional Networks)

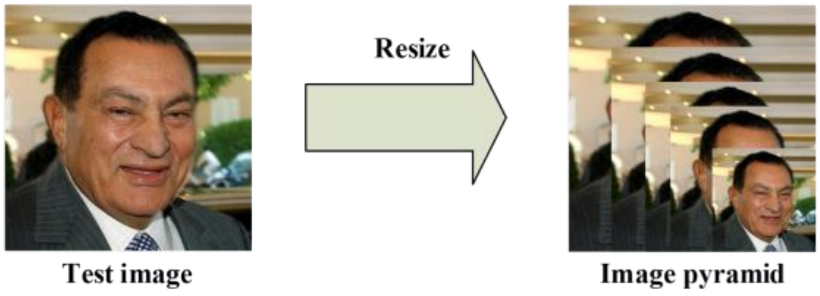
MTCNN hay mạng thần kinh kết nối đa nhiệm theo tầng là một mạng nơ-ron phát hiện khuôn mặt và các điểm mốc trên khuôn mặt trên hình ảnh. Nó được xuất bản vào năm 2016 bởi Zhang et al.[5]

#### Cấu trúc mạng MTCNN

**MTCNN** là một trong những công cụ nhận diện khuôn mặt phổ biến và chính xác nhất hiện nay. Mạng MTCNN bao gồm 3 mạng nơ-ron được kết nối trong một tầng (P-Net, R-Net và O-Net).

Hình ‑ Cấu trúc mạng MTCNN

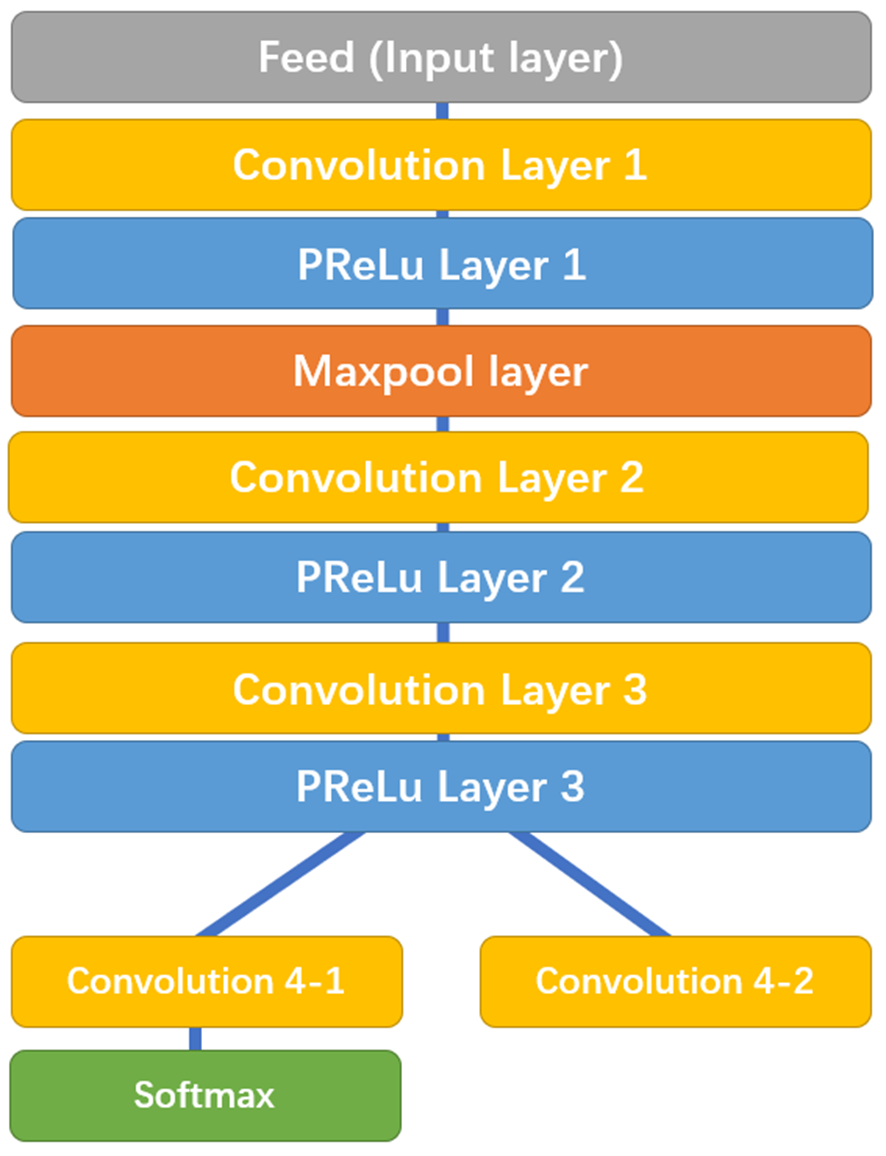
Với mỗi hình ảnh truyền vào, mạng sẽ tạo ra một kim tự tháp hình ảnh, nó tạo ra nhiều bản sao của hình ảnh đó với các kích thước khác nhau.



Hình ‑ Kim tự tháp ảnh

#### Neural Networks P-Net

Trong P-Net, đối với mỗi hình ảnh được chia tỷ lệ, một kernel 12x12 chạy qua hình ảnh, tìm kiếm một khuôn mặt. Kernel 12x12 ở góc trên cùng bên phải. Sau khi quét góc này, nó sẽ dịch chuyển sang một bên (hoặc xuống dưới) 1 pixel và tiếp tục làm điều đó cho đến khi nó đi qua toàn bộ hình ảnh. Ngoài ra, một lớp maxpool được đưa vào sau lớp prelu đầu tiên (maxpool loại bỏ mọi pixel khác, chỉ để lại pixel lớn nhất trong vùng lân cận).

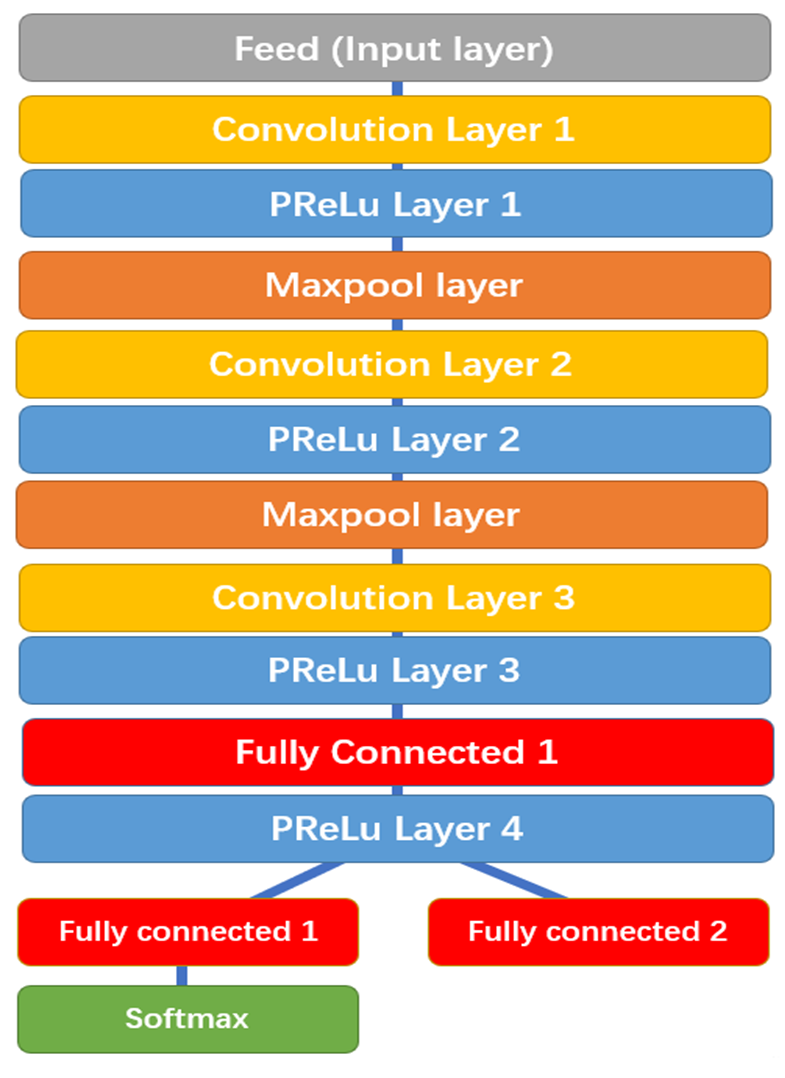
Sau lớp chập thứ ba, mạng chia thành hai lớp. Các kích hoạt từ lớp thứ ba được chuyển đến hai lớp tích chập riêng biệt và một lớp softmax sau một trong các lớp tích chập đó (softmax gán xác suất thập phân cho mọi kết quả và xác suất cộng lại là 1. Trong trường hợp này, nó xuất ra 2 xác suất: xác suất có mặt trong khu vực và xác suất không có mặt ).

Hình ‑ P-Net

Phép chập 4–1 xuất ra xác suất của một mặt trong mỗi hộp giới hạn và phép chập 4–2 xuất ra tọa độ của các hộp giới hạn.

#### Neural Networks R-Net

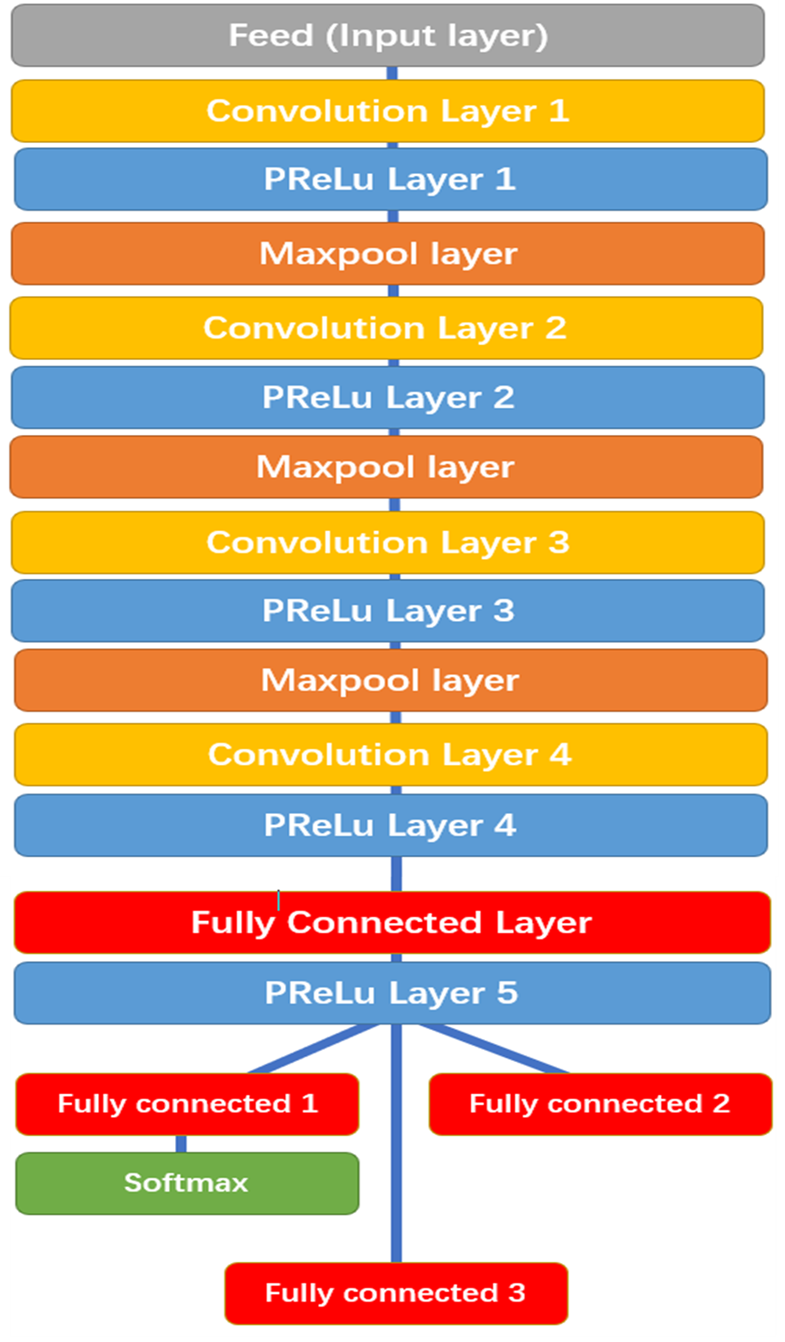
R-Net có cấu trúc tương tự, nhưng có nhiều lớp hơn. Nó lấy các hộp giới hạn P-Net làm đầu vào và tinh chỉnh tọa độ của nó. Tương tự, R-Net cuối cùng chia thành hai lớp, đưa ra hai kết quả: tọa độ của các hộp giới hạn mới và độ tin cậy của máy đối với mỗi hộp giới hạn.



Hình ‑ R-Net

#### Neural Networks P-Net

O-Net lấy các hộp giới hạn R-Net làm đầu vào và đánh dấu tọa độ của các điểm mốc trên khuôn mặt.



Hình ‑ O-Net

Cuối cùng, O-Net chia thành 3 lớp, đưa ra 3 kết quả khác nhau: xác suất của một khuôn mặt nằm trong hộp, tọa độ của hộp giới hạn và tọa độ của các điểm mốc trên khuôn mặt (vị trí của mắt, mũi và mồm).

#### Cách mô hình MTCNN hoạt động:

**Giai đoạn 1:**

1. Chuyển vào hình ảnh
2. Tạo nhiều bản sao được chia tỷ lệ của hình ảnh
3. Nạp hình ảnh được chia tỷ lệ vào P-Net
4. Thu thập đầu ra P-Net
5. Xóa các hộp giới hạn với độ tin cậy thấp
6. Chuyển đổi tọa độ kernel 12 x 12 thành tọa độ "hình ảnh chưa được chia tỷ lệ"
7. Loại bỏ không tối đa đối với kernel trong mỗi hình ảnh được chia tỷ lệ
8. Cấm không tối đa cho tất cả các kernel
9. Chuyển đổi tọa độ hộp giới hạn thành tọa độ "hình ảnh chưa được chia tỷ lệ"
10. Định hình lại các hộp giới hạn thành hình vuông

**Giai đoạn 2:**

1. Đệm hộp không bị ràng buộc
2. Nguồn cấp dữ liệu hình ảnh được chia tỷ lệ vào R-Net
3. Thu thập đầu ra R-Net
4. Xóa các hộp giới hạn với độ tin cậy thấp
5. Cấm tối đa cho tất cả các hộp
6. Chuyển đổi tọa độ hộp giới hạn thành tọa độ "hình ảnh chưa được chia tỷ lệ"
7. Định hình lại các hộp giới hạn thành hình vuông

**Giai đoạn 3:**

1. Đệm hộp không bị ràng buộc
2. Nguồn cấp dữ liệu hình ảnh được chia tỷ lệ vào O-Net
3. Thu thập đầu ra O-Net
4. Xóa các hộp giới hạn với độ tin cậy thấp
5. Chuyển đổi hộp giới hạn và tọa độ mốc trên khuôn mặt thành tọa độ "hình ảnh chưa được chia tỷ lệ"
6. Cấm tối đa cho tất cả các hộp

**Cung cấp kết quả:**

1. Đóng gói tất cả các tọa độ và mức độ tin cậy vào một từ điển
2. Trả lại từ điển

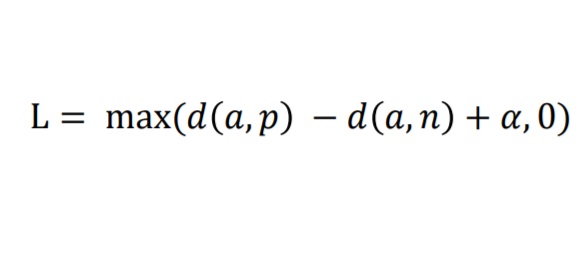
Để triển khai cũng như hiểu rõ hơn về MTCNN chúng ta có thể tìm hiểu thông qua bài báo của Zhang et al[6]

### FaceNet

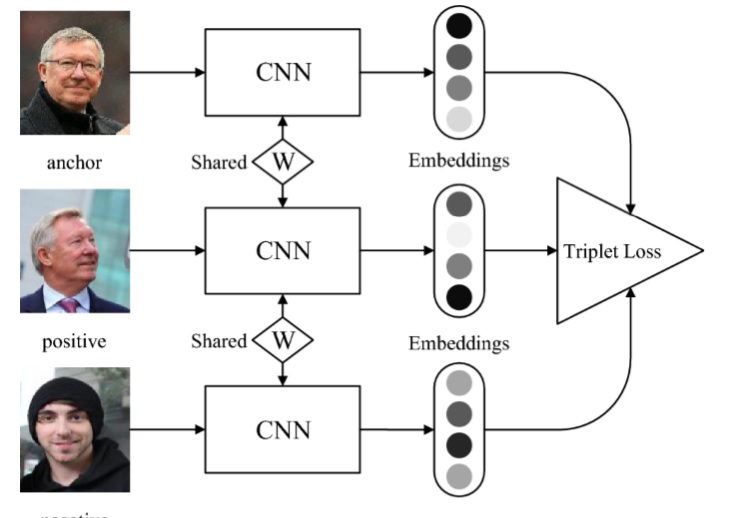
**FaceNet** là một hệ thống nhận dạng khuôn mặt sử dụng mạng thần kinh sâu được giới thiệu vào năm 2015 bởi các nhà nghiên cứu tại Google, **Florian Schroff**, **Dmitry Kalenichenko** và **James Philbin**[7]. Ý tưởng chính của hệ thống này là đào tạo một **CNN** trích xuất một **vector 128 gia tốc** từ một hình ảnh khuôn mặt, được gọi là **Embedding**. Các vectơ được trích xuất từ ​​hình ảnh của cùng một người rất gần nhau, trong khi khoảng cách giữa các vectơ được trích xuất từ ​​hình ảnh khuôn mặt của hai người khác nhau nên lớn hơn nhiều.

Để đạt được điều kiện này, CNN được đào tạo với ba hình ảnh ở mỗi bước, được đặt tên là **Anchor** , **Positive** và **Negative** . Hình ảnh Anchor và Positive thuộc về cùng một người và hình ảnh âm bản là của một cá nhân khác.

Chức năng **Triplet loss** nhằm mục đích làm cho CNN giảm khoảng cách giữa Anchor’s Embedding và Positive’s Embedding và tăng khoảng cách giữa Anchor’s Embedding và Negative’s Embedding trong khi đó:

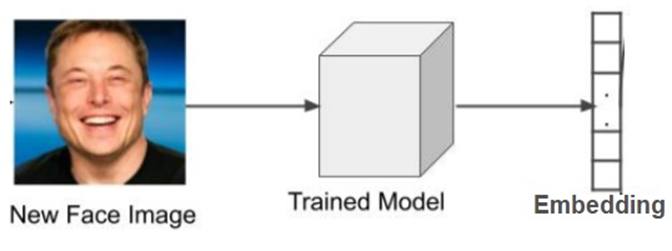


Trong đó a là **anchor**, p là **positive**, n là **negative** và **alpha** là **magin**, chúng ta chỉ định để đảm bảo biên (alpha +) giữa Embeddingkhuôn mặt của hai người khác nhau.

Đảm bảo rằng hình ảnh **A** (anchor) của một người cụ thể gần với tất cả các hình ảnh khác **P** (positive) của cùng một người hơn là với bất kỳ hình ảnh **N** (negative) nào của bất kỳ người nào khác.

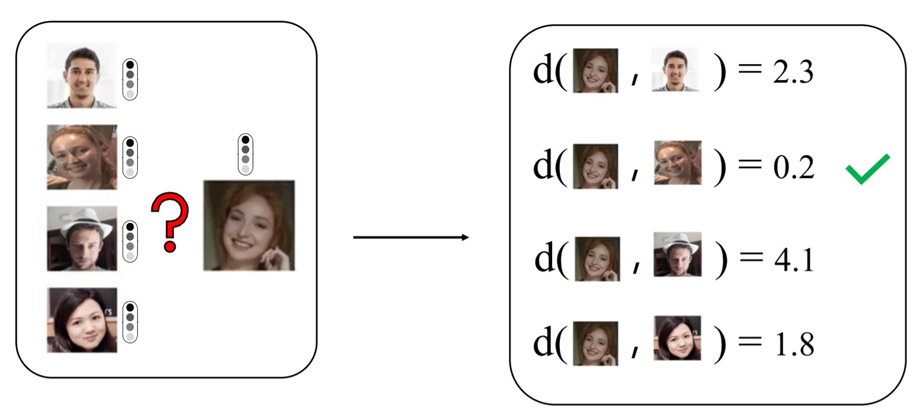
Hình ‑ Cấu trúc mạng FaceNet

Khi CNN được đào tạo tốt, nó có thể tạo ra Embeddingthống nhất cho từng khuôn mặt nhất định, thể hiện các tính năng của nó.



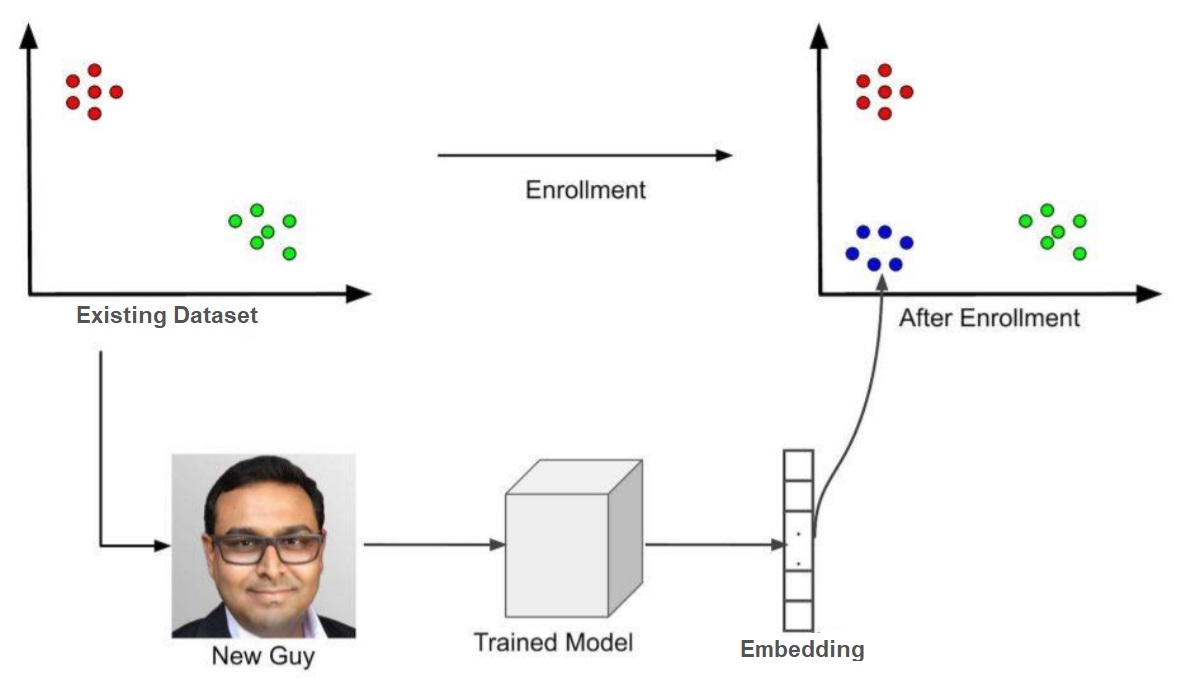
Hình ‑ Trích xuất Embedding khuôn mặt bằng FaceNet Model

Có thể áp dụng bất kỳ phương pháp so sánh nào để tìm khuôn mặt gần nhất trong tập dữ liệu cho một khuôn mặt nhất định dựa trên các Embedding của chúng (SVM, KNN,...). Quá trình tìm kiếm kết quả phù hợp nhất (mặt gần nhất) với một mặt đã cho được gọi là **Inference.**



Hình ‑ 4 Tìm khuôn mặt gần nhất với khuôn mặt đã cho trong số 4 hình ảnh dựa trên Embedding của chúng

Chúng ta có thể sử dụng các thước đo khoảng cách nổi tiếng để tính toán khoảng cách giữa hai Embedding chẳng hạn như các thước đo khoảng cách Minkowski hoặc Euclidean. Sau đó, sẽ dễ dàng tìm thấy khoảng cách tối thiểu trong số các khoảng cách được tính toán. Vì vậy, điều gì sẽ xảy ra nếu một công ty thuê một nhân viên mới và muốn nhận diện họ trên cổng ra vào? Họ chỉ cần chụp ảnh khuôn mặt của nhân viên, đưa nó vào cùng một mô hình FaceNet CNN để tạo ra Embedding và thêm nó vào tập dữ liệu hiện có.



Hình ‑ Thêm khuôn mặt của nhân viên mới Embedding vào tập dữ liệu hiện có

Để hiểu rõ hơn về mô hình FaceNet - hệ thống nhận dạng khuôn mặt sử dụng mạng thần kinh sâu cũng như cách triển khai hãy tìm hiểu bài báo của các nhà nghiên cứu tại Google[7].

### SVM (Support vector machines)

Máy vectơ hỗ trợ (SVM) là một họ các phương pháp học máy, ban đầu được giới thiệu cho vấn đề phân loại và sau đó được khái quát hóa cho nhiều tình huống khác. Chúng dựa trên các nguyên tắc của lý thuyết học thống kê và tối ưu hóa lồi, và hiện đang được sử dụng trong các lĩnh vực ứng dụng khác nhau, bao gồm tin sinh học, phân loại văn bản và thị giác máy tính. Copyright © 2009 John Wiley & Sons, Inc[8].

## Dữ liệu thu thập

## Các mô hình được sử dụng trong hệ thống

## Tham chiếu

# 

# **KẾT LUẬN**

# 

# TÀI LIỆU THAM KHẢO

[1] “Node.js”, Accessed: Mar. 14, 2022. [Online]. Available: https://www.w3schools.com/nodejs/default.asp

[2] J. Bucanek, “Model-View-Controller Pattern,” *Learn Object. Java Dev.*, pp. 353–402, 2009, doi: 10.1007/978-1-4302-2370-2\_20.

[3] V. Wang, F. Salim, and P. Moskovits, “The WebSocket Protocol,” *Defin. Guid. to HTML5 WebSocket*, pp. 33–60, 2013, doi: 10.1007/978-1-4302-4741-8\_3.

[4] “Heroku: Up and Running: Effortless Application Deployment and Scaling - Neil Middleton, Richard Schneeman - Google Sách.” https://books.google.com.vn/books?hl=vi&lr=&id=IS4KAgAAQBAJ&oi=fnd&pg=PR2&dq=heroku&ots=eFK3N5MhxB&sig=NBVusKWRkvC0UkLjLymGnY8BXhI&redir\_esc=y#v=onepage&q=heroku&f=false (accessed Mar. 14, 2022).

[5] R. Jin, H. Li, J. Pan, W. Ma, and J. Lin, “Face recognition based on MTCNN and Facenet,” 2021.

[6] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, 2016, doi: 10.1109/LSP.2016.2603342.

[7] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 815–823, 2015, doi: 10.1109/CVPR.2015.7298682.

[8] A. Mammone, M. Turchi, and N. Cristianini, “Support vector machines,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 1, no. 3, pp. 283–289, Nov. 2009, doi: 10.1002/wics.49.