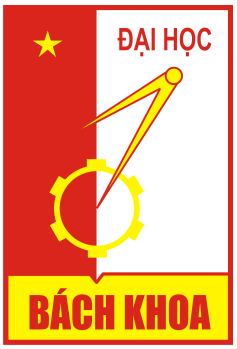
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ Thông tin và Truyền thông

Báo cáo IOT và Ứng dụng



Hệ thống điểm danh bằng công nghệ nhận diện khuôn mặt

Sinh viên:
Đào Hoàng Long - 20183579
Phạm Quốc Việt - 20183668
Nguyễn Tiến Mạnh - 20180132

Giảng viên hướng dẫn:
TS. Nguyễn Đình Thuận

Mục lục

Chương 1

Giới thiệu

1.1 Lí do chọn đề tài

Kỷ nguyên công nghệ 4.0, Smart IoT và AI IoT đang định hình lại và ảnh hưởng lên tất cả lĩnh vực đời sống, sản xuất, kinh doanh, quản trị và điều hành xã hội. Đặc biệt, các ứng dụng nhận dạng khuôn mặt kết hợp trí tuệ nhân tạo AI đang trở thành trào lưu và được quan tâm rất sâu và rộng trong nhiều tổ chức, doanh nghiệp, cá nhân nhằm giải quết nhiều bài toán quản lý khác nhau tạo nên những thay đổi to lớn và mang lại hiệu quả rất tích cực.

Có thể dễ dàng nhận thấy công nghệ nhận dạng khuôn mặt kết hợp với các thiết bị phần cứng chuyên dụng được dùng khá phổ biến trong các mục đích như: Chấm công điểm danh, kiểm soát ra vào được dùng nhiều tại các văn phòng, nhà máy từ nhiều năm trở lại đây. Công nghệ nhận dạng khuôn mặt cũng đang được triển khai cho mục đích kiểm soát an ninh như: Kiểm soát thang máy, kiểm soát ra vào sảnh tòa nhà, kiểm soát vào ra bãi đỗ xe, kiểm soát khách ra vào...

Vì vậy, nhóm chúng em sẽ xây dựng ứng dụng nhận diện khuôn mặt để điểm danh, trong đó có sử dụng ESP32-CAM.

1.2 Nhận diện khuôn mặt là gì?

Nhận diện khuôn mặt thông minh là các giải pháp sử dụng trí tuệ nhân tạo để phát hiện khuôn mặt từ hình ảnh hoặc video được ghi nhận bởi camera, so khớp với các hỉnh ảnh khuôn mặt sẵn có trong cơ sở dữ liệu khuôn mặt đã được lưu trước đó.

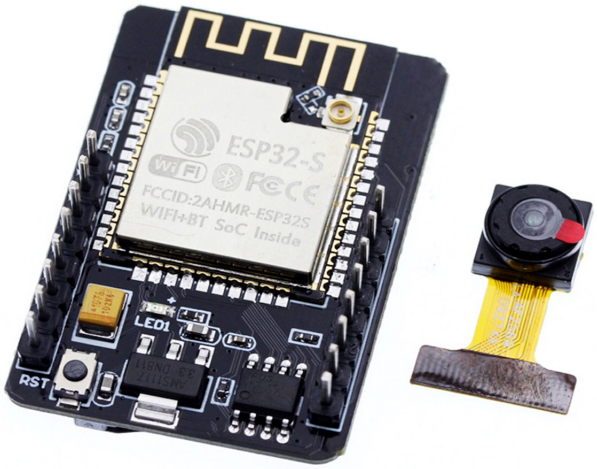
Chương 2

Phần cứng và phần mềm

2.1 Phần cứng

2.1.1 ESP32-CAM

ESP32-CAM là một mô-đun hoàn chỉnh với một bộ vi điều khiển tích hợp, có thể hoạt động độc lập. Ngoài kết nối WiFi + Bluetooth, mô-đun này còn có một máy quay video tích hợp và một khe cắm thẻ nhớ microSD để lưu trữ.



Hình 2.1: ESP32-CAM

Đặc tính kỹ thuật ESP32-CAM:

- Conectividad: WiFi 802.11b/g/n + Bluetooth 4.2 với BLE. Hỗ trợ tải lên hình ảnh qua WiFi.
- Kết nối: UART, SPI,I2C, và PWM. Nó có 9 chân GPIO.
- Tần số đồng hồ: lên đến 160Mhz.

- Sức mạnh tính toán của vi điều khiển: lên đến 600 DMIPS.
- Bộ nhớ: 520KB SRAM + 4MB PSRAM + khe cắm thẻ SD
- Extras: có nhiều chế độ nghỉ, có thể nâng cấp phần sụn bằng OTA và đèn LED để sử dụng bộ nhớ flash tích hợp.
- Máy ảnh: Hỗ trợ camera OV2640 có thể đi kèm hoặc mua độc lập. Các loại máy ảnh này có:
  - 2 MP trên cảm biến của bạn.
  - Kích thước mảng UXGA 1622 × 1200 px.
  - Định dạng đầu ra YUV422, YUV420, RGB565, RGB555 và nén dữ liệu 8-bit.
  - Có thể truyền hình ảnh từ 15 đến 60 FPS.

2.1.2 Mạch nạp Arduino Pro Mini CP2102

Hình 2.2: Cp2102

Tính năng: Mạch nạp tương thích cho arduino Pro mini hoặc cho các mạch arduino tự làm. Chỉ cần nối thẳng chân tương ứng rồi dùng ứng dụng arduino IDE trên PC nạp code.

Thông số kỹ thuật:

- Mạch sử dụng chip CP2102 của hãng SILICON LABS.
- Hỗ trợ nhiều Hệ điều hành WinXP, Vista, Win7, Win8, Win10, Server 2003, Mac OS - X, Linux...

- Mạch không dùng thạch anh ngoài.
- Có 2 đường nguồn 5V và 3V3.
- Trên mạch có 6 cổng đầu ra: 3.3V DTR 5V Tx Rx Gnd. Trong đó chân DTR được sử dụng để reset vi điều khiển trong quá trình nạp.
- Tốc độ baurate tối đa 115200 bps.
- Có LED báo hiệu Tx/Rx khi nhận/gửi dữ liệu.

Mô tả chân:

- 5V: Áp ra 5V (tối đa 500mA).
- 3.3V: Áp ra 3.3V.
- GND: chân mass hoặc nối đất.
- DTR: Chân reset để nạp cho vi điều khiển.
- TXD: chân truyền dữ liệu UART.
- RXD: chân nhận dữ liệu UART.

2.2 Phần mềm

2.2.1 Arduino IDE

Arduino IDE sử dụng ngôn ngữ lập trình C/C++ rất phổ biến trong giới lập trình. Bất kỳ đoạn code nào của C/C++ thì Arduino IDE đều có thể nhận dạng, giúp các lập trình viên thuận tiện trong việc thiết kế chương trình lập cho các bo mạch Arduino.

Arduino có một module quản lý bo mạch, nơi người dùng có thể chọn bo mạch mà họ muốn làm việc cùng và có thể thay đổi bo mạch thông qua Menu. Quá trình sửa đổi lựa chọn cũng liên tục tự động cập nhật để các dữ liệu có sẵn trong bo mạch và dữ liệu sửa đổi đồng nhất với nhau. Bên cạnh đó, Arduino IDE cũng giúp tìm ra lỗi từ code mà bạn biết giúp bạn sửa lỗi kịp thời tránh tình trạng bo mạch Arduino làm việc với code lỗi quá lâu dẫn đến hư hỏng hoặc tốc độ xử lý bị giảm sút.

2.2.2 Django

Django là một framework lập trình web bậc cao, mã nguồn mở được viết bằng python, hỗ trợ lập trình web trong thời gian ngắn.

Mô hình MVT được sử dụng trong khi tạo một ứng dụng với Tương tác người dùng. Mô hình này thì bao gồm code HTML với Django Templage Language (DTL). Controller là mã được viết để kiểm soát sự tương tác giữa Model và View và Django dễ dàng chăm sóc nó. Bất cứ khi nào người dùng người request, nó xử lý request của người dùng đó bằng Model, View và Template. Nó hoạt động như một Controller để kiểm tra xem nó có khả dụng hay không bằng cách ánh xạ URL và nếu URL ánh xạ thành công thì View sẽ bắt đầu tương tác với Model và gửi lại Template cho người dùng dưới dạng respone.

Dijango hỗ trợ kĩ thuật ORM(Object-Relational Mapping) cho phép truy vấn và quản lý dữ liệu từ database bằng cách sử dụng mô hình hướng đối tượng. Qua đó tương tác với database một cách dễ dàng, hiệu quả. Django cũng hỗ trợ việc hiển thị database trên giao diện Admin.

2.2.3 Thư viện Face Recognition

Đây là một thư viện nhạn dạng khuôn mặt rất nổi tiếng và độ chính xác khá tốt xây dựng dựa trên dlib của python và được viết bằng C++.

Độ chính xác của mô hình trên tập Labeled Faces in the Wild là $99,38\%$.

2.2.4 MQTT broker

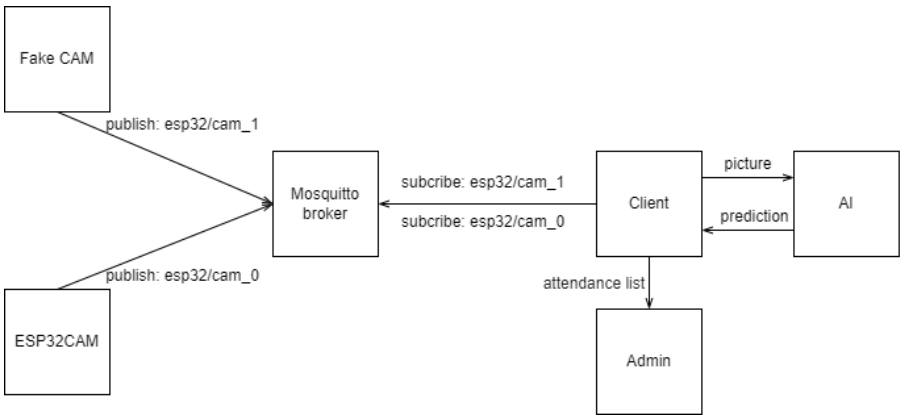
Trong project này, để tiết kiệm chi phí, bọn em sử dụng phần mềm Eclipse Mosquitto dành cho Window. (Link dowload `https://mosquitto.org/download/`)

Sau khi cài đặt, bọn em sẽ có 1 MQTT broker trên máy local và thể kết nối đến nó thông qua 1 địa chỉ IP cố định, tuy nhiên chỉ các máy trong cùng 1 host mới có thể kết nối được. Ngoài ra, bọn em cũng đã thử dùng dịch vụ AWS IoT, tuy nhiên để có được tốc độ tốt thì phải bỏ ra chi phí khá lớn. Do đó, bọn em quyết định sử dụng bản MQTT này để tiết kiệm chi phí và đảm bảo tốc độ kết nối.

Chương 3

Xây dựng ứng dụng

3.1 Tổng quan hệ thống



Hình 3.1: Tổng quan hệ thống

Hệ thống có lắp đặt ESP32CAM, nạp code và publish ảnh lên Mosquitto broker. Tuy nhiên vì sử dụng bản miễn phí của MQTT nên chất lượng chưa đủ tốt vì vậy có thêm một camera giả lập để phục vụ demo chương trình.

Sau khi publish lên MQTT thì sẽ xây dựng Client để subcribe ảnh xuống, gửi đến model AI để dự đoán khuôn mặt từ đó trích xuất ra danh sách những người có mặt để gửi về admin, phục vụ công tác điểm danh.

3.2 Các thành phần chi tiết

3.2.1 ESP32CAM

Các thư viện được sử dụng trong Arduino IDE: "Arduino.h", "WiFi.h", "esp_camera.h", "base64.h" và "PubSubClient.h". Trong đó, "esp_camera.h" dùng để thiết lập cho ESP32CAM, "base64.h" dùng để mã hóa dữ liệu sang

định dạng ASCII và "PubSubClient.h" dùng để kết nối với MQTT broker.

Phần lập trình cho ESP32CAM cao gồm các thành phần chính như sau:

- Hàm set up bao gồm các nhiệm vụ: cài đặt các chân của ESP32CAM, khởi tạo camera, kết nối wifi và kết nối với MQTT broker.

```
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000; // was at 20
config.pixel_format = PIXFORMAT_JPEG;
config.frame_size = FRAMESIZE_SVGA; //800 x 600 necessary for Adafruit IO
config.jpeg_quality = 30;
config.fb_count = 1;

// camera init
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK)
{
  Serial.printf("Camera init failed with error 0x%x", err);
  return;
}

setup_wifi();
client.setServer(mqtt_server, 1883);
client.setCallback(callback);
```

Hình 3.2: Phần lập trình cho hàm set up.

- Hàm Loop thực hiện nhiệm vụ capture ảnh và publish lên topic cho trước.

```
// Capture picture
//digitalWrite(ESP32CAM_LED_FLASH, HIGH);
camera_fb_t *fb = NULL;
fb = esp_camera_fb_get();

if (!fb)
{
  Serial.println("Camera capture failed");
  return;
}
else
{
  Serial.println("Camera Captured");
}
//digitalWrite(ESP32CAM_LED_FLASH, LOW);
// delay(1000);

// size_t size = fb->len;
String buffer = base64::encode((uint8_t *)fb->buf, fb->len);
```

Hình 3.3: Phần lập trình cho hàm loop.

- Sau khi capture, ảnh sẽ được chuyển sang dạng mã ASCII để publish lên MQTT broker.

```
if (client.publish(mqtt_publish_topic, buffer.c_str()))
{
  count++;
  long int ctime = millis();
  float frps = float( (float) count/( (ctime-stime)/1000 ) );
  Serial.print("Published at ");
  Serial.print(frps);
  Serial.print(" frames/s");
}
```

Hình 3.4: Phần lập trình publish ảnh lên MQTT broker.

Chương trình sẽ hiển thị tốc độ pushlish ra Serial. Tốc độ publish trung bình của ESP32CAM là 8.6 frames/s.

3.2.2 Fake Cam

Ở dưới là phần lập trình của Fake Cam, trong đó chương trình sẽ đọc các hình ảnh được lưu ở folder "demo_image/data" (các ảnh trong folder này

được trích xuất ra từ 1 video ngắn). Sau đó, chương trình sẽ publish các ảnh vào topic "esp32/cam_1". Tốc độ publish trung bình của Fake Cam là 16.2 frames/s.

```python
import paho.mqtt.client as mqtt
import base64
import time

if __name__=="__main__":
    broker_address="192.168.1.68"
    client = mqtt.Client()
    client.connect(broker_address)
    client.loop_start() #start the loop
    root = "demo_image/data"
    count = 0
    ctime = time.time()
    while True:
        for i in range(229):
            path = "demo_image/data/frame{}.jpg".format(i)
            with open(path, "rb") as image:
                message = base64.b64encode(image.read())
            client.publish("esp32/cam_1", message, qos=2)
            time.sleep(0.05)
            count += 1
            print("Image pushed at {} frames/s".format(count/(time.time()-ctime)))
    time.sleep(4) # wait
    client.loop_stop() #stop the loop
```

Hình 3.5: Phần lập trình cho Fake Cam.

3.2.3 AI

AI dùng thư viện dlib và face_recognition (phát hiện khuôn mặt, so sánh với encodings của các khuôn mặt được lưu trong folder "dataset"). Mỗi khuôn mặt được cho qua hàm encoding để trích xuất encoding của nó.

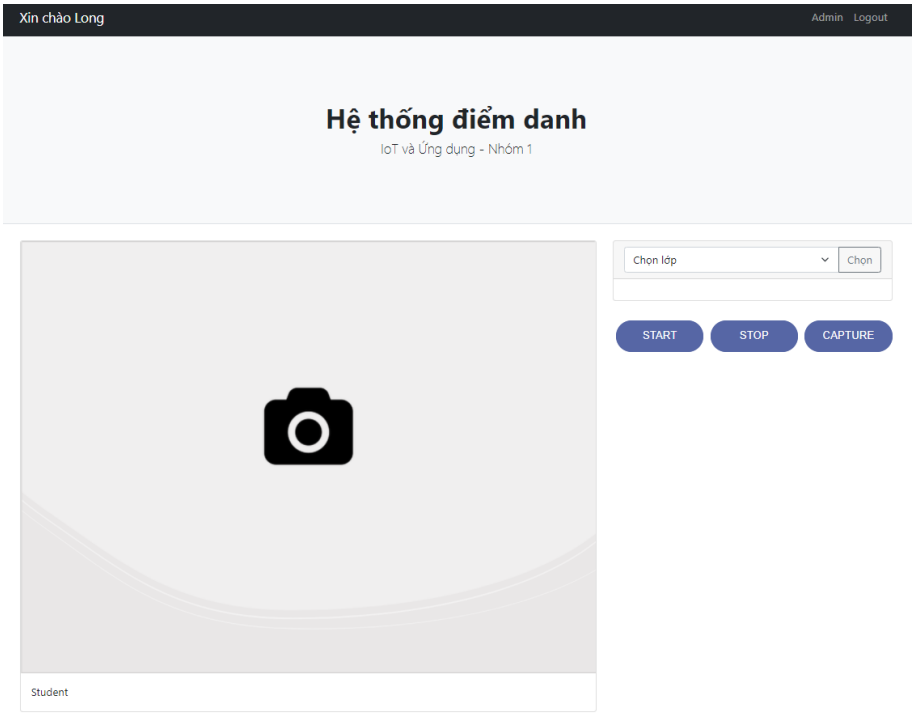
Thư viện face_recognition cung cấp các hàm để phát hiện khuôn mặt ở trên bức ảnh xong lấy encoding khuôn mặt được phát hiện và so sánh với các encoding đã được lưu sẵn để cho ra dự đoán. Sau khi dự đoán sẽ hiển thị mã số sinh viên tương ứng dưới hình ảnh trên web và cập nhật danh sách điểm danh để gửi về admin.

Trong thư viện face_recognition có 2 mô hình có thể sử dụng là "hog" và "cnn". Trong đó, mô hình "hog" tốn ít tài nguyên tính toán hơn nhưng đòi hỏi ảnh có độ sáng tốt, góc quay chính diện và độ chính xác cũng kém hơn, còn "cnn" có thể nhận diện với bất cứ góc mặt nào với độ chính xác cao hơn.

Ở trong project này, do hạn chế về tài nguyên nên bọn em sẽ sử dụng mô hình "hog" để thực nghiệm.

3.2.4 Client

Client được thiết kế theo chuẩn template của Django hỗ trợ kết hợp sử dụng html, css, js và có gọi thêm 1 số thư viện của bootstrap. Qua đó xây dựng một website nhanh với front-end tinh gọn và tương tác tốt với back-end.



Hình 3.6: Giao diện chính của Client.

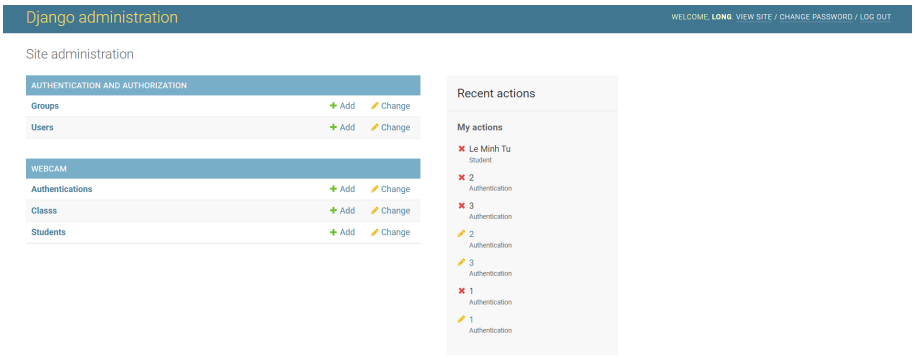
Chức năng đầu tiên của Client là chọn lớp, người dùng sẽ chọn 1 lớp từ danh sách các lớp cho trước ở bên tay phải và click "chọn". Sau đó, người dùng click "START" để lấy hình ảnh từ camera tương ứng với lớp đó, người dùng có thể dừng nhận hình ảnh bằng việc click "STOP". Ngoài ra, client có thể chọn "CAPTURE" để nhận diện khuôn mặt đang xuất hiện trên khung hình. Một chức năng khác là client có thể truy cập vào trang admin bằng cách click vào "admin" ở góc trên bên phải màn hình (chỉ áp dụng với user có quyền admin).

3.2.5 Admin

Django framework hỗ trợ xây dựng giao diện Admin bằng cách sử dung Object-Relational Mapping (ORM) là một kĩ thuật cho phép truy vấn và quản lý dữ liệu từ database bằng cách sử dụng mô hình hướng đối tượng.

ORM là những thư viện đã hoàn chỉnh được viết bằng ngôn ngữ lập trình đang dùng và nó đã gói nhóm code cần thiết để quản lý data, như vậy không cần sử dụng SQL nữa, sẽ tương tác trực tiếp qua cách viết hướng đối tượng của ngôn ngữ bạn đang sử dụng.

Thông qua tạo các user để đăng nhập được vào hệ thống, phía back-end cũng định nghĩa những model tương tác với database và cách hiển thị của chúng.

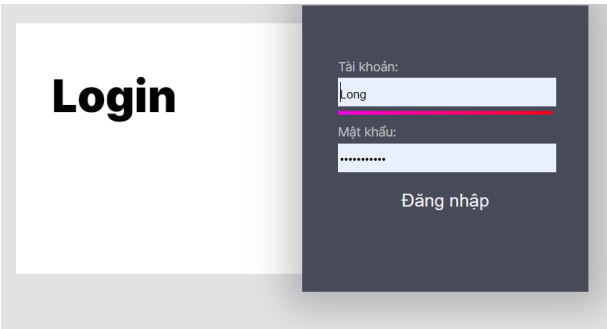


Hình 3.7: Giao diện chính của Admin

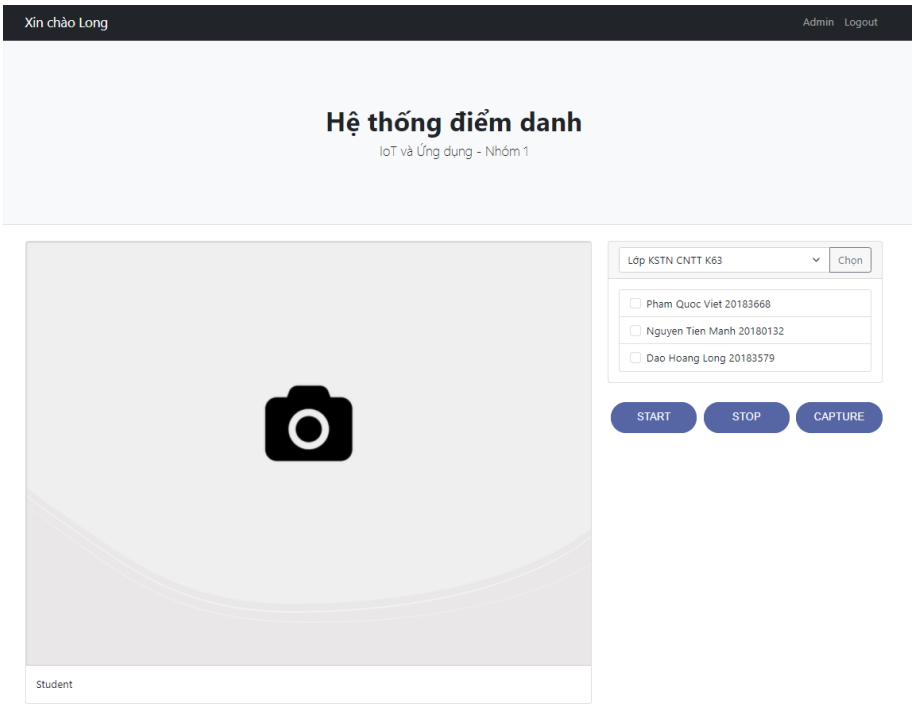
Admin sẽ có nhiệm vụ quản lí (thêm, sửa, xóa) danh sách các người dùng (Users), danh sách lớp (Class), danh sách học sinh (Students) và danh sách điểm danh (Authenications).

Chương 4

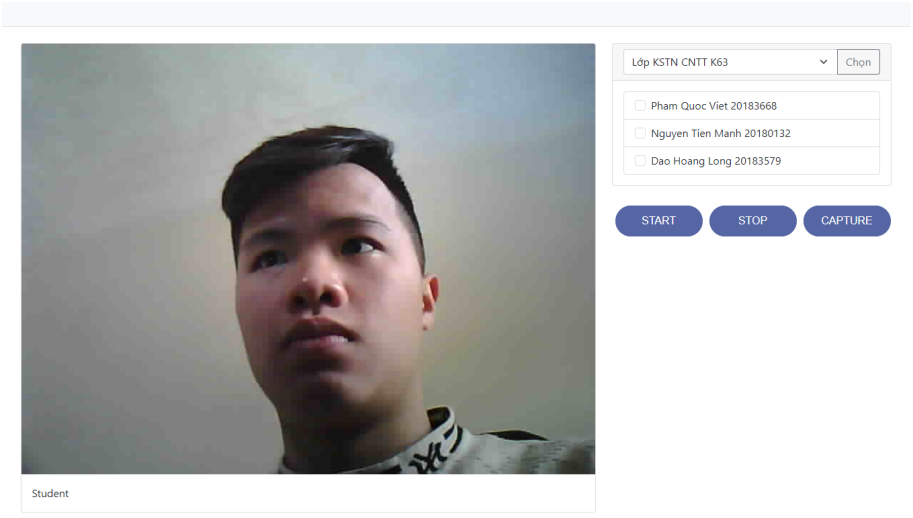
Demo



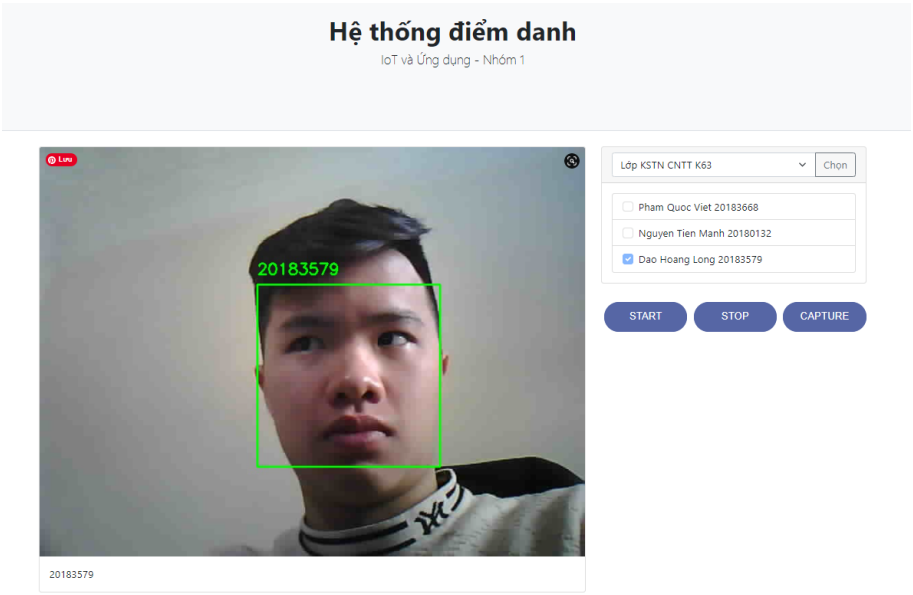
Hình 4.1: Người dùng đăng nhập vào hệ thống.



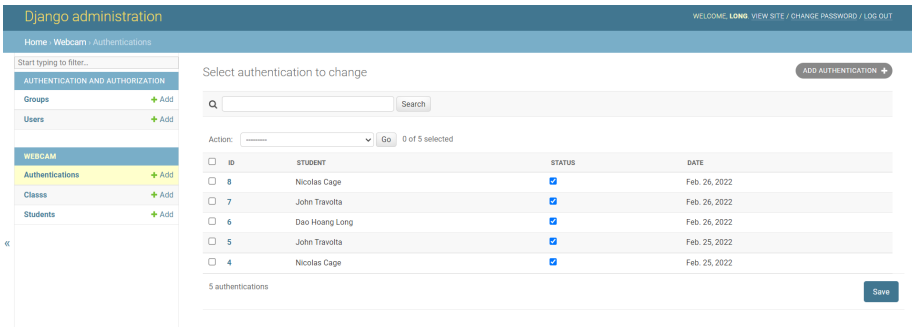
Hình 4.2: Người dùng chọn lớp và hệ thống hiện thị danh sách lớp.



Hình 4.3: Người dùng chọn "START" và hệ thống hiển thị hình ảnh.



Hình 4.4: Người dùng chọn "CAPTURE" và hệ thống thực hiện điểm danh. Hệ thống sẽ tích vào danh sách điểm danh ở bên tay phải và hiển thị mã số sinh viên ở dưới khung hình.



Hình 4.5: Người dùng vào trang Admin và chọn Authenications để xem danh sách điểm danh mới nhất.

Chương 5

Kết luận

Như vậy, thông qua project này nhóm đã hiểu về cách thức kết nối phần cứng, cụ thể là với ESP32 CAM, một chíp xử lý có tốc độ cao, khá phổ biến cho lập trình nhúng.Nhóm cũng đã lập trình ra một trang web đơn giản, áp dụng framework hỗ trợ lập trình nhanh để tạo ra app điểm danh bằng nhận dạng khuôn mặt. Từ đó hiểu được phần nào mức độ và quy mô của những ứng dụng sử dụng IoT và AI trên thực tế. Điều này giúp ích rất nhiều cho công việc sau này của các thành viên nhóm trong tương lai.

Nhận dạng khuôn mặt hiện nay là một lĩnh vực đã được nghiên cứu khá nhiều. Do đó nhóm cũng dễ dàng đề xuất các hướng phát triển như : Cải thiện tốc độ nhận dạng, nhận dạng bằng một góc bất kỳ thay vì phải quay nhiều góc như hiện tại hay nhận dạng được nhiều người một lúc. Về phần ESP32CAM thì có thể lắp thêm thẻ nhớ SD để tích hợp sẵn một số xử lý ngay trên chip.Ngoài ra nếu có điều kiện có thể thuê các dịch vụ của Amazon Webservice để chạy một server khỏe hơn thay vì chạy trên máy tính cá nhân như hiện tại.

Phân công công việc:

- Nguyễn Tiến Mạnh: front-end, kết nối các thành phần trong hệ thống để chạy.
- Phạm Quốc Việt: back-end, AI nhận diện khuôn mặt.
- Đào Hoàng Long: Lắp mạch phần cứng, lập trình Arduino, lập trình kết nối MQTT.