HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 2

BÁO CÁO LAB 3

Xây dựng hệ thống Face Mask Detection

- **Môn học:** IoT và ứng dụng (INT14149)
- **Giảng viên hướng dẫn :** Đàm Minh Lịnh
- **Nhóm:** 05
- **Thành viên:**

N20DCCN034	-	Vũ Văn Lâm
N20DCCN035	-	Trần Gia Long
N20DCCN039	-	Quách Nam Lương
N20DCCN058	-	Lương Thanh Quý
N20DCCN062	-	Đặng Thành Tân
N20DCCN073	-	Phạm Đức Thắng
N20DCCN078	-	Nguyễn Thị Thùy Trang
N20DCCN085	-	Lương Thúy Vy
N19DCAT087	-	Phạm Văn Thủy

Thành phố Hồ Chí Minh, 10

Mục lục

Chương I. Tổng quan 3
Chương II. Mô hình 4
1. Đặc tả mô hình. 5
2. ESP32-CAM. 6
3. Teachable Machine. 7
4. Telegram. 8
Chương III. Xây dựng mô hình. 9
1. Cài đặt thư viện. 9
2. Cài đặt Telegram và tạo bot. 11
3. Code. 13
Chương IV. Kết quả. 19

Chương I. Tổng quan

Đại dịch COVID-19 là một đại dịch bệnh truyền nhiễm với tác nhân là virus corona và các biến thể của nó đang diễn ra trên phạm vi toàn cầu. Khởi nguồn vào cuối tháng 12 năm 2019 với tâm dịch đầu tiên tại thành phố Vũ Hán thuộc miền Trung Trung Quốc đại lục, bắt nguồn từ một nhóm người mắc viêm phổi không rõ nguyên nhân. Sau đó dịch bệnh đã lan rộng ra toàn cầu, gây ảnh hưởng nghiệm trọng đến tất cả lĩnh vực.

Virus corona chủ yếu lây lan qua các giọt bắn trong không khí khi một cá nhân bị nhiễm bệnh ho hoặc hắt hơi trong phạm vi khoảng 3 foot (0,91 m) đến 6 foot (1,8 m). Do vậy, WHO đã khuyến nghị nhiều biện pháp góp phần ngăn ngừa sự lây lan của dịch bệnh, trong đó có biện pháp đeo khẩu trang ở nơi công cộng.. Để kiểm soát việc đeo khẩu trang tại nơi công cộng (sân bay, bệnh viện,…) hoặc nơi làm việc, nhiều công ty, đơn vị quản lý đã sử dụng hệ thống phát hiện người đeo khẩu trang và người không đeo để có biện pháp cảnh báo, nhắc nhở kịp thời. Ngoài ra việc kiểm soát này có thể sử dụng để truy vết kịp thời nếu có ca bệnh liên quan. Hệ thống trên cần một công nghệ đáp ứng được hiệu suất tốt về mặt thời gian và độ chính xác..

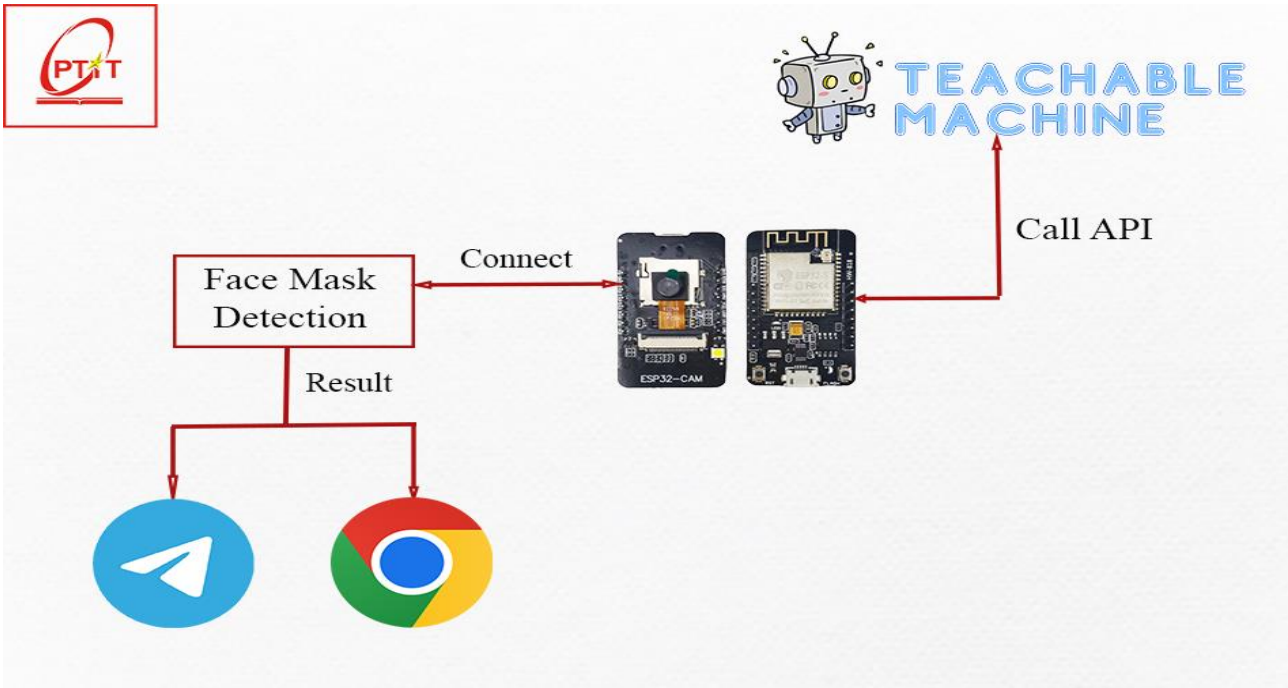
Đề tài "Face Mask Detection" xuất phát từ nhu cầu này và mục tiêu xây dựng một hệ thống thông minh có khả năng nhận diện việc đeo khẩu trang thông qua hình ảnh.

Chương II. Mô hình

Mô hình được đề xuất cho đề tài "Face Mask Detection" là một hệ thống thông minh tích hợp IoT (Internet of Things), trí tuệ nhân tạo (AI), và giao tiếp qua Telegram. Mô hình này nhằm mục tiêu nhận diện và cảnh báo việc đeo khẩu trang trên khuôn mặt con người, đặc biệt trong bối cảnh đại dịch COVID-19. Mô hình này bao gồm các thành phần chính:

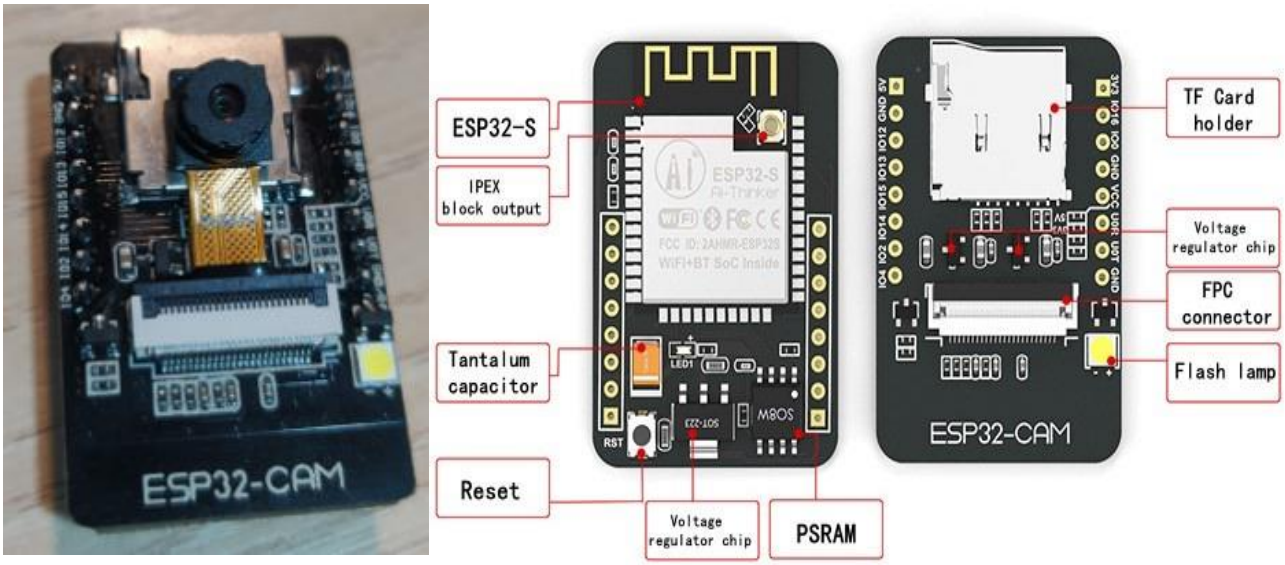
* ESP32-CAM: Thiết bị IoT này là trung tâm của hệ thống, chịu trách nhiệm thu thập hình ảnh từ camera và kết nối với Teachable Machine.
* Teachable Machine: Nền tảng trí tuệ nhân tạo này được sử dụng để phân loại hình ảnh khuôn mặt thành hai loại: có đeo khẩu trang và không đeo khẩu trang.
* Telegram: Ứng dụng giao tiếp này cho phép người dùng gửi yêu cầu và nhận thông báo về việc đeo khẩu trang thông qua bot Telegram.

1. Đặc tả mô hình.



- Các thành phần:
  + Dùng thiết bị ESP32-CAM.
  + Dùng Arduino nạp code cho ESP32-CAM.
  + ESP-CAM sử dụng API của Teachable Machine.
  + Thực hiện chức năng phát hiện mang khẩu trang và trả kết quả về Telegram.

2. ESP32-CAM.



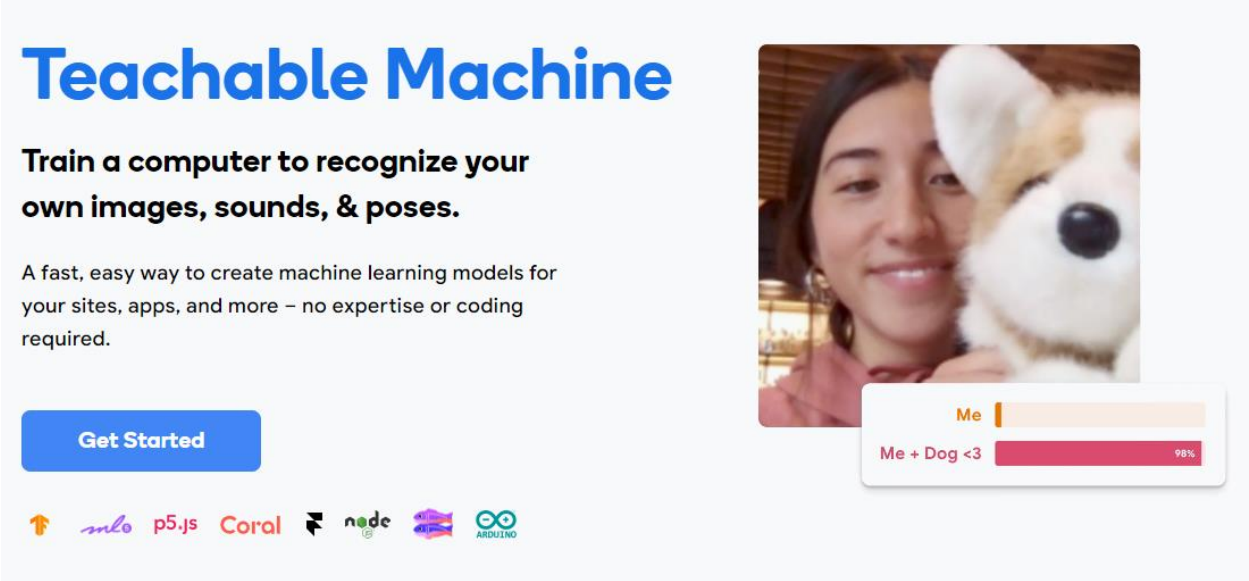
- ESP32-CAM là một mô-đun máy ảnh rất nhỏ có gắn chip ESP32-S kết hợp với máy ảnh OV2640 và một số GPIO(General Purpose Input Output) để kết nối các thiết bị ngoại vi, nó còn có khe cắm thẻ nhớ microSD có thể hữu ích để lưu trữ hình ảnh được chụp bằng máy ảnh hoặc lưu trữ các tập tin để phục vụ khách hàng.

- Đế nạp chương trình ESP32-CAM USB Programming Adapter giúp mạch ESP32-CAM nạp chương trình và giao tiếp truyền dữ liệu với máy tính qua giao IC chuyển USB-UART CH340 một cách dễ dàng, đế còn tích hợp hai nút nhấn cho chân RST và IO0 của ESP32-CAM để sử dụng trong quá trình nạp chương trình.

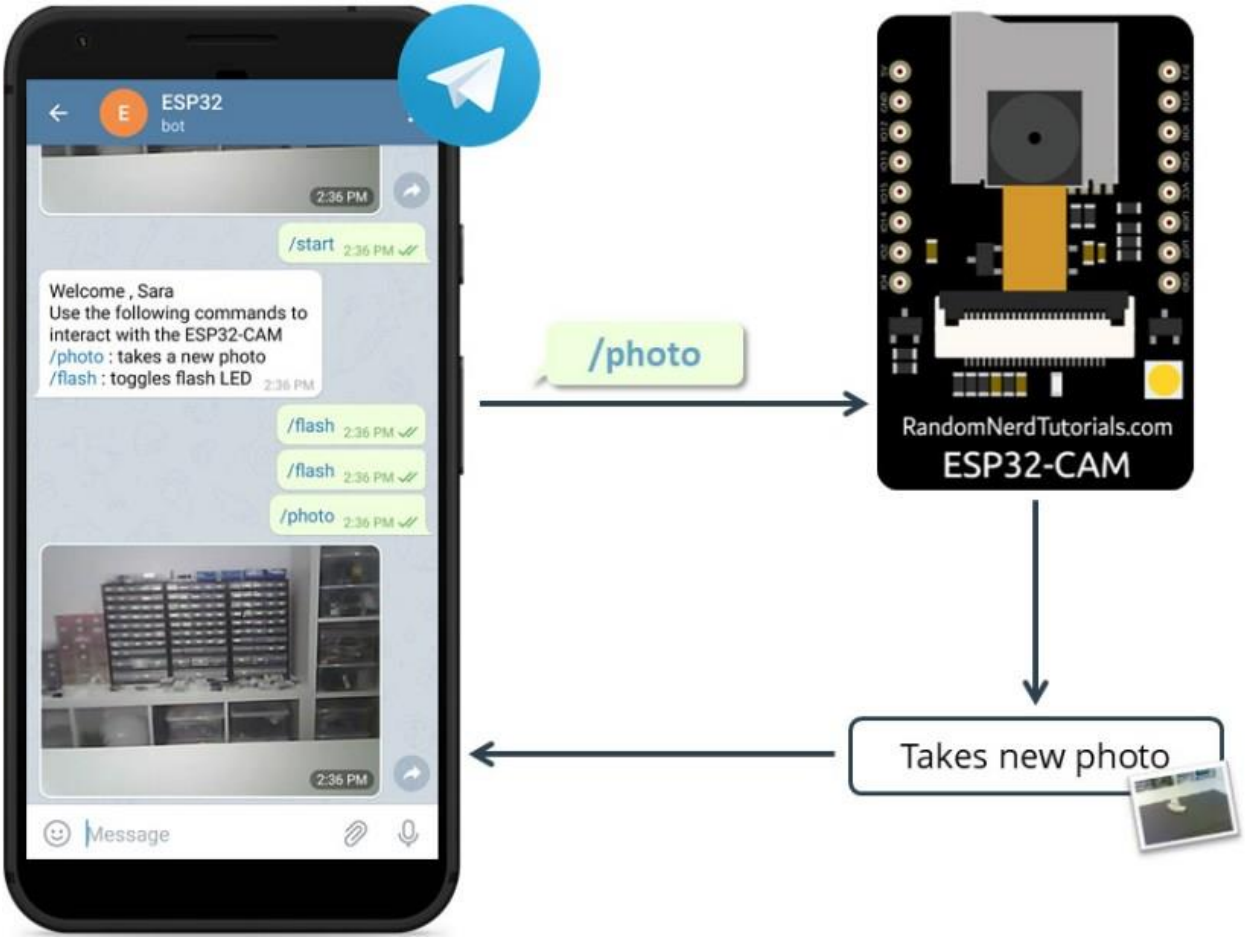
3. Teachable Machine.

"Teachable Machine" là một ứng dụng mạnh mẽ và dễ sử dụng phát triển bởi Google, được thiết kế để đơn giản hóa quá trình học máy. Ứng dụng này cho phép người dùng tạo mô hình học máy chỉ bằng cách sử dụng hình ảnh hoặc âm thanh, mà không yêu cầu kiến thức lập trình sâu. Bạn có thể nạp dữ liệu, định nghĩa các lớp, và sau đó, Teachable Machine sẽ tự động huấn luyện mô hình cho bạn.

Với Teachable Machine, bạn có khả năng tạo ra các ứng dụng thú vị như phân loại các đối tượng, nhận diện biểu hiện trong hình ảnh hoặc âm thanh, và thậm chí xây dựng các ứng dụng liên quan đến nhận diện khuôn mặt và đeo khẩu trang. Ứng dụng này là một công cụ tuyệt vời để giúp bạn hiểu và áp dụng lý thuyết học máy vào các dự án thực tế một cách dễ dàng và thú vị.



4. Telegram.



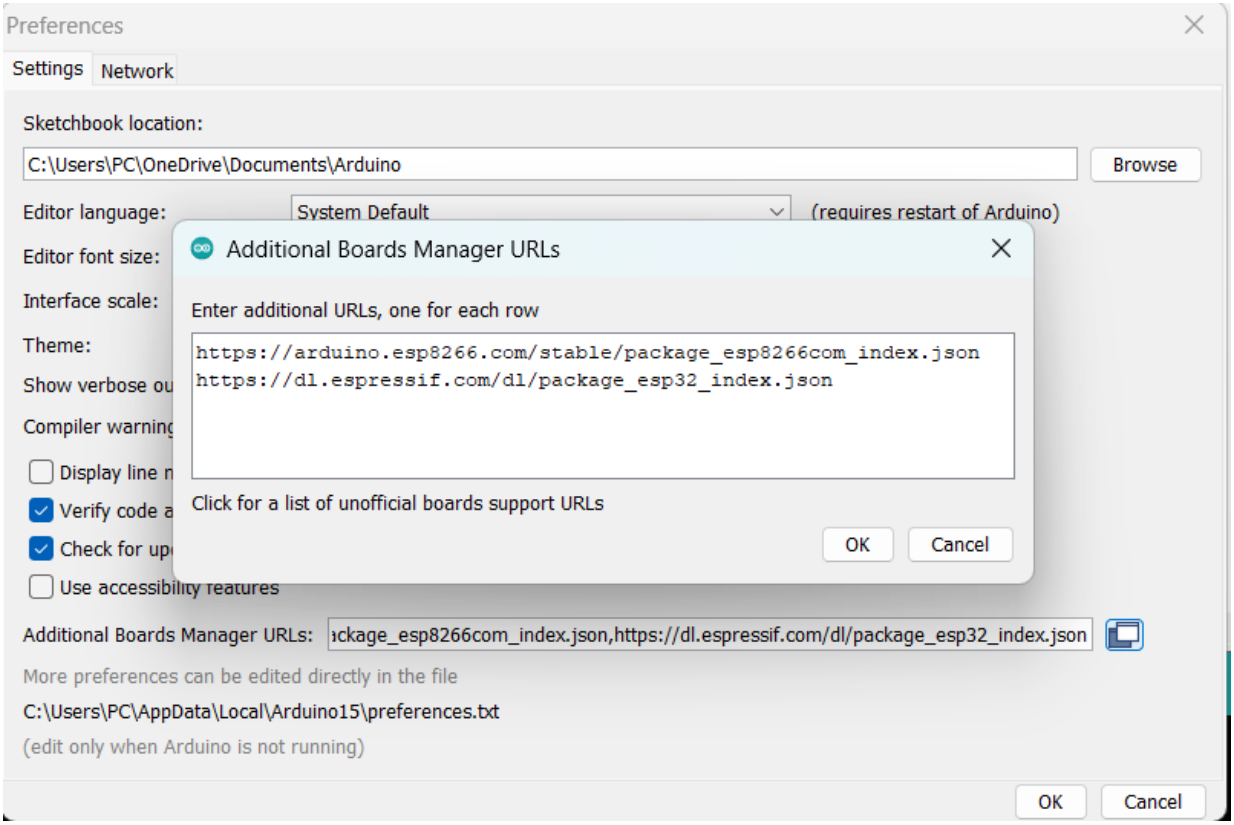
Telegram Messenger là dịch vụ cloud cho phép gửi tin nhắn và giọng nói qua IP. Telegram cho phép người dùng tạo các bot mà người dùng có thể tương tác.“Bot là ứng dụng của bên thứ ba chạy bên trong Telegram. Người dùng có thể tương tác với bot bằng cách gửi cho chúng tin nhắn, lệnh và yêu cầu nội tuyến. Người dùng kiểm soát các bot của mình bằng cách sử dụng các yêu cầu HTTPS tới API Telegram Bot”. ESP32-CAM sẽ tương tác với bot Telegram để nhận và xử lý tin nhắn cũng như gửi phản hồi.

Chương III. Xây dựng mô hình.

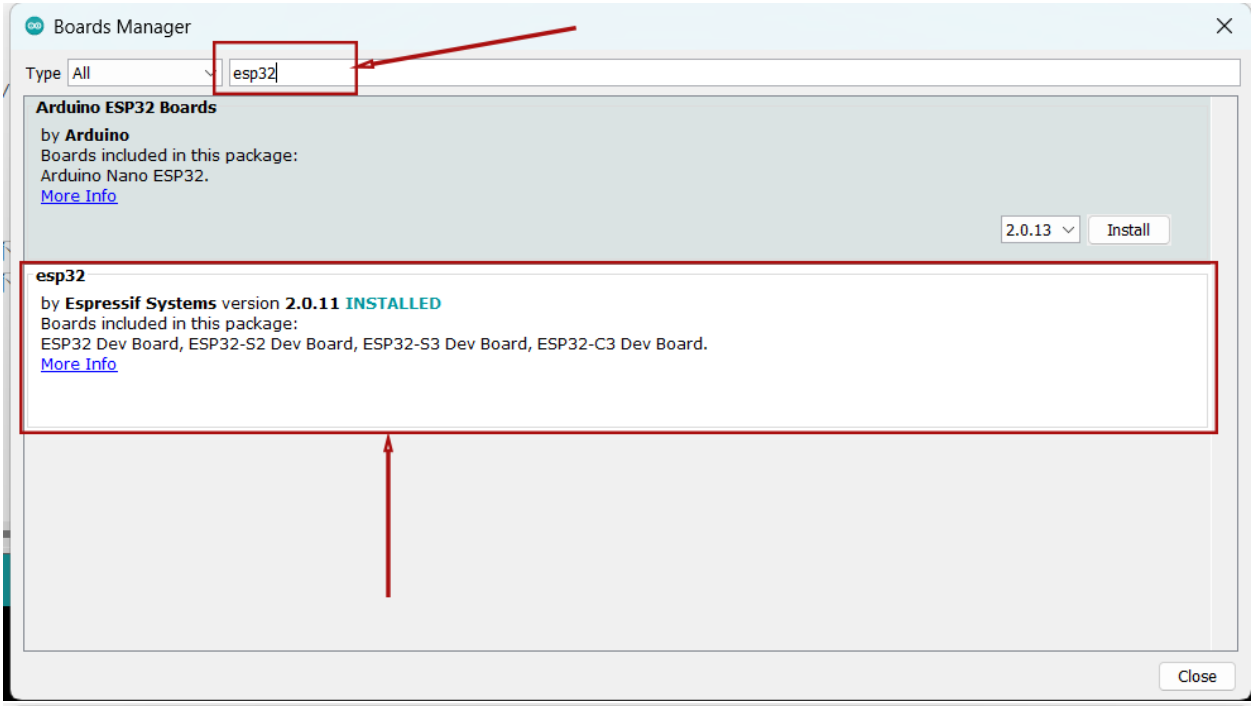
1. Cài đặt thư viện.

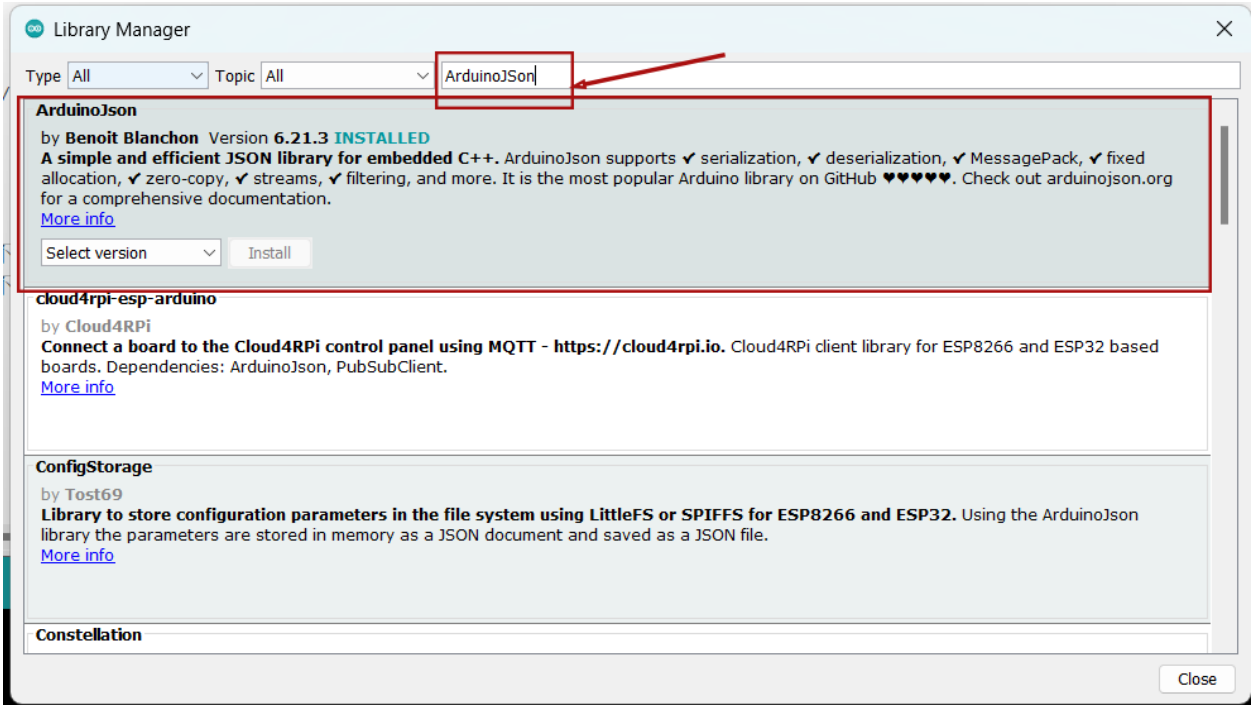
- Cài đặt các thư viện cần thiết để xây dựng mô hình. Thêm Boards Manager URLs:

https://dl.espressif.com/dl/package_esp32_index.json,http://arduino.esp8266.com/stable/package_esp8266com_index.json

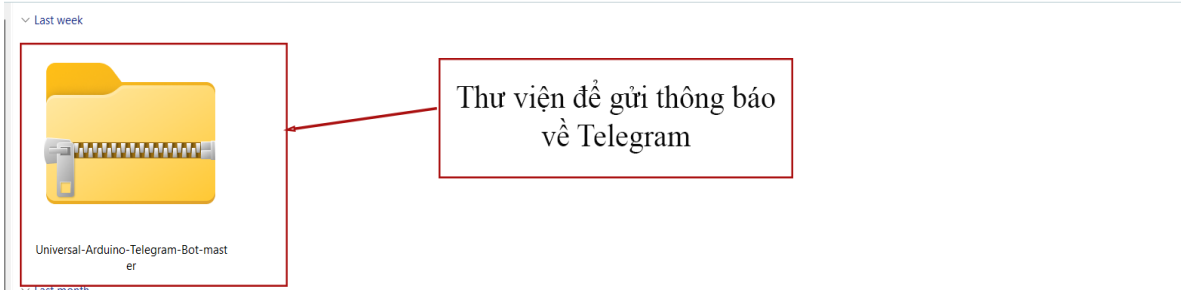


- Sau đó thực hiện cài đặt các thư viện:





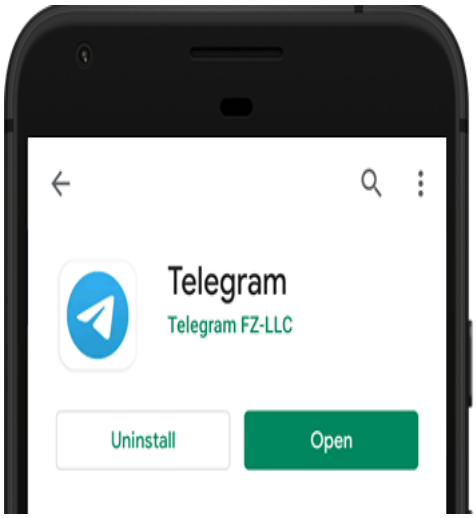
- Thêm thư viện gửi thông báo về Telegram:



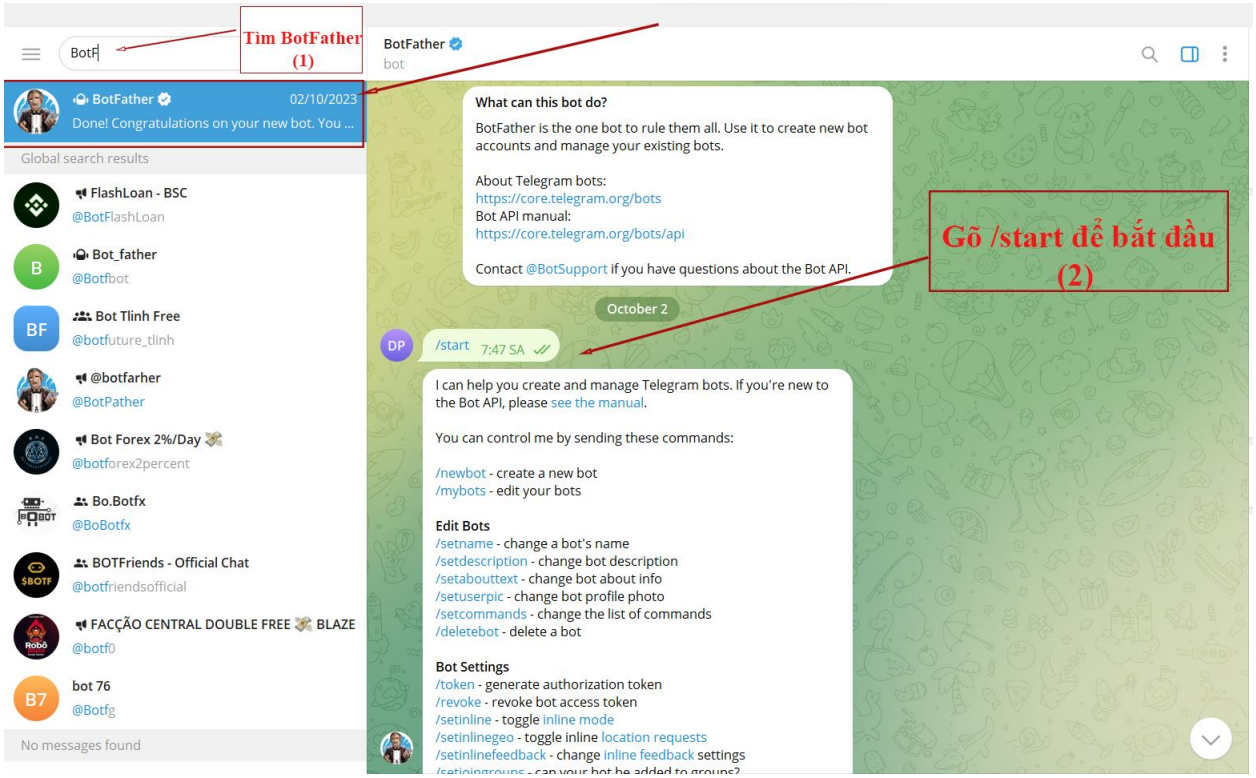
Link dowload: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot.git .

2. Cài đặt Telegram và tạo bot.

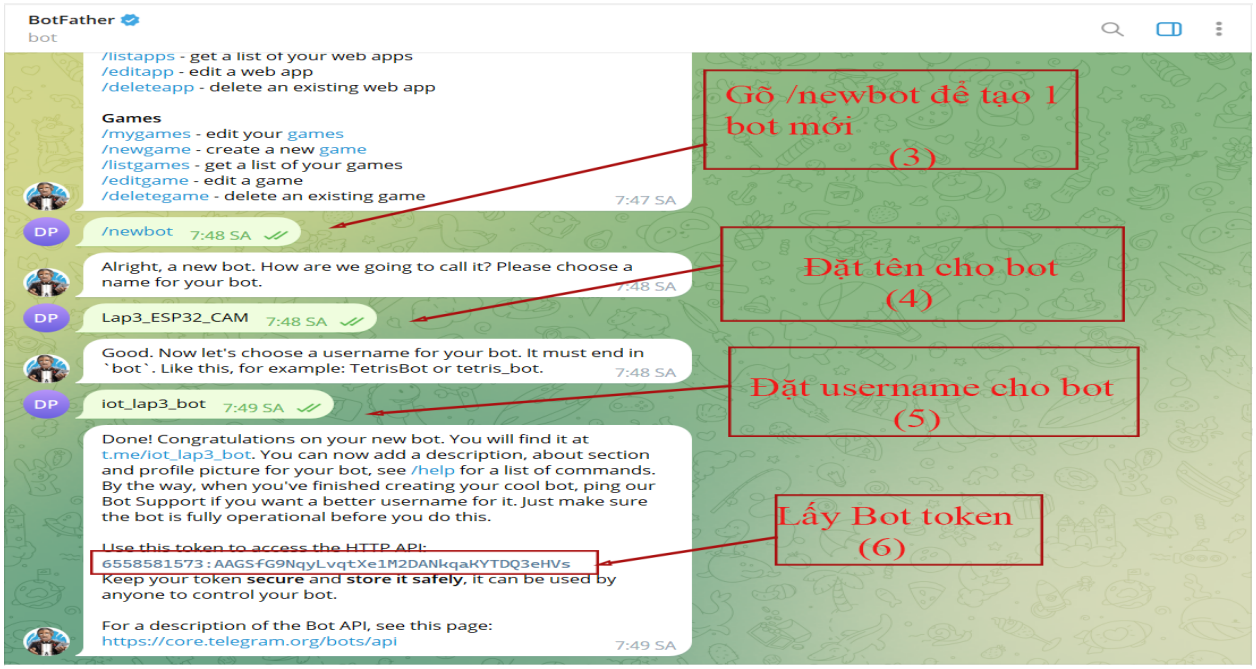
- Truy cập Google Play hoặc App Store, tải xuống và cài đặt Telegram. Hoặc tải xuống và cài đặt trên desktop thông qua link dowload: https://desktop.telegram.org/ .



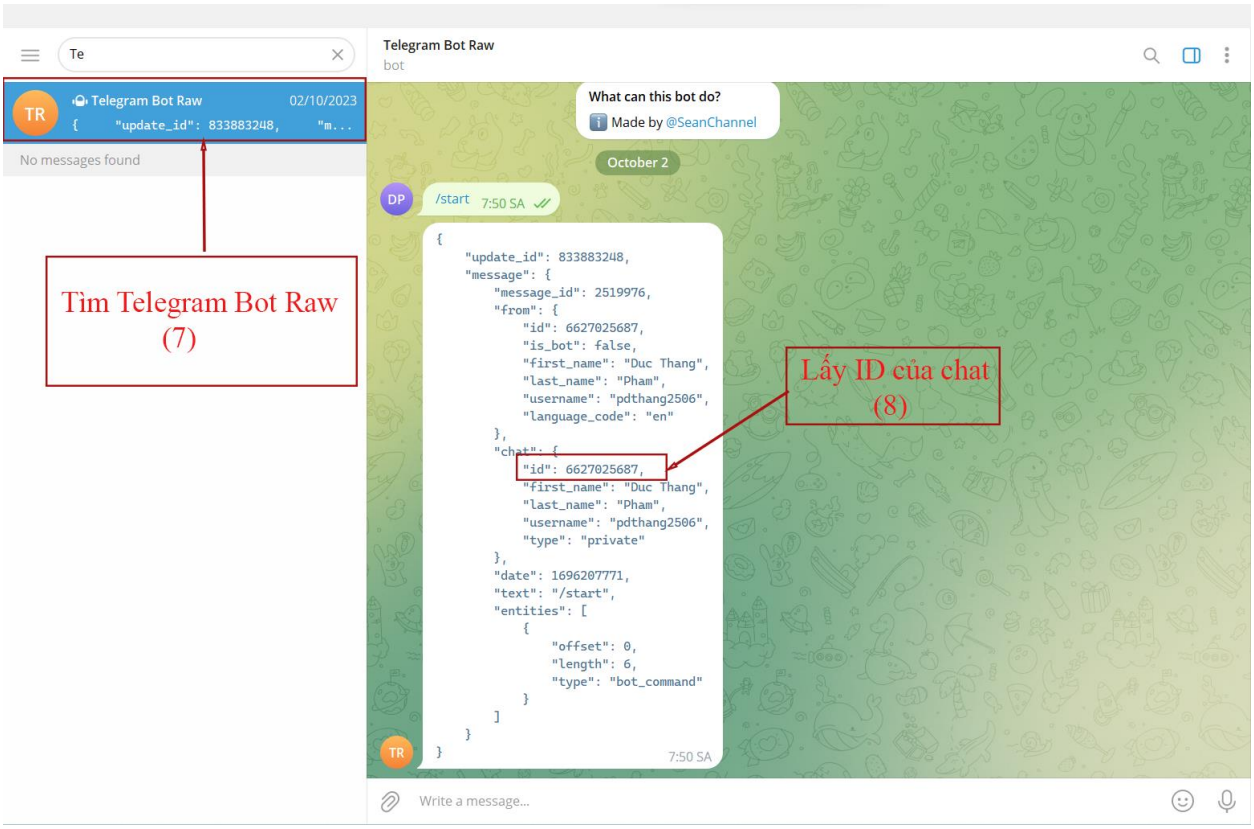
- Tìm kiếm “botfather” và nhấp vào BotFather.



- Nhập /newbot và làm theo hướng dẫn để tạo bot của bạn. Đặt cho bot một cái tên và tên người dùng.



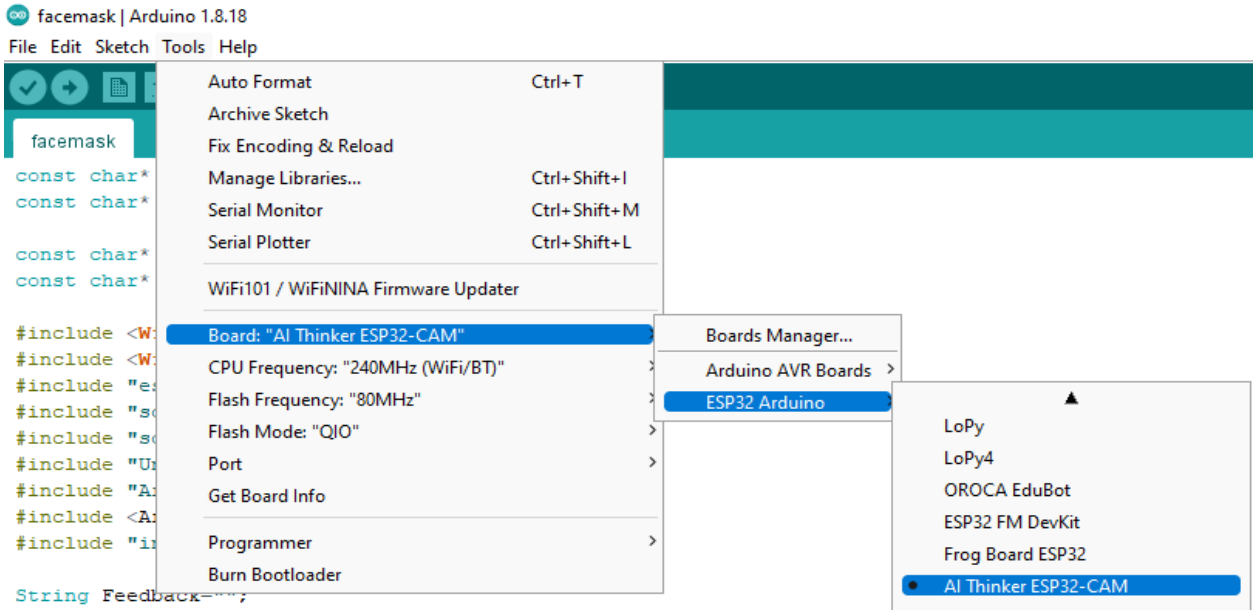
* Sau đó bạn có thể lấy ID của chat bằng cách tìm kiếm “Telegram Bot Raw”.



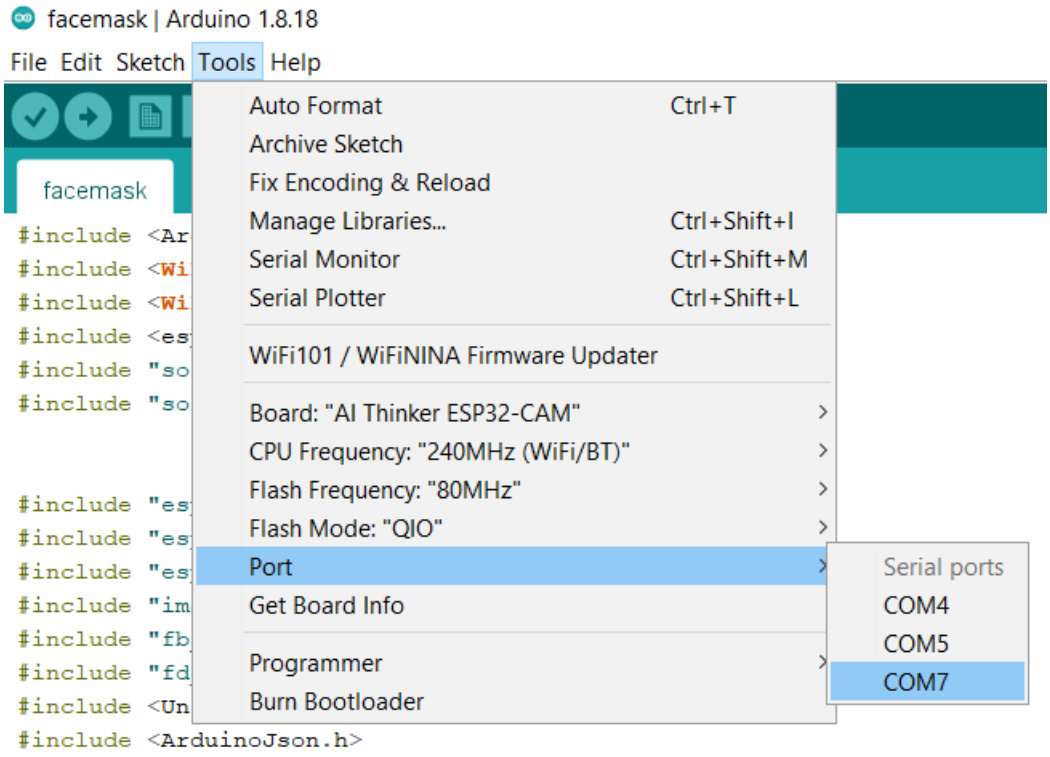
Hoặc "IdBot" và nhập vào "/getid".

3. Code.

- Thiết lập board: Chọn Tools > Board > ESP32 Arduino > AI-Thinker ESP32-CAM.



- Thiết lập PORT: Tools > Port và chọn COM port mà ESP32-CAM được kết nối.



- Khai báo những thư viện cần thiết.

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "esp_camera.h"
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "UniversalTelegramBot.h"
#include "ArduinoJson.h"
#include <Arduino.h>
#include "index.h"
```

- Cấu hình kết nối telegram và wifi.

```
String BOTtoken = "6558581573:AAGSfG9NqyLvqtXe1M2DANk[redacted]HVs";
String CHAT_ID = "6[redacted]5687";
bool flashState = LOW;
int botRequestDelay = 1000;
unsigned long lastTimeBotRan;

bool sendPhoto = false;
WiFiClientSecure clientTCP;
UniversalTelegramBot bot(BOTtoken, clientTCP);
```

* BOTtoken : giá trị Telegram Bot token lấy được từ BotFather.
* CHAT_ID: mã ID cuộc trò chuyện.
* botRequestDelay, lastTimeBotRan: là các biến sử dụng để kiểm tra tin nhắn Telegram mới mỗi giây (1000 mili giây) – có thể thay đổi giá trị trễ này trong biến botRequestDelay.
* sendPhoto: biến kiểm tra đã đến lúc gửi ảnh mới tới tài khoản Telegram hay chưa. Mặc định biến được đặt bằng false.
* clientTCP: tạo mới Wifi client với WiFiClientSecure.
* bot(BOTtoken, clientTCP) : tạo bot với token và client.

- Định nghĩa các chân GPIO trên thiết bị ESP32-CAM.

```
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
#define FLASH_LED_PIN 4
```

- Hàm khởi tạo camera ESP32.

```
void configInitCamera(){
 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;
  config.pixel_format = PIXFORMAT_JPEG;
  //init with high specs to pre-allocate larger buffers

  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;  //0-63 lower number means higher quality
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;  //0-63 lower number means higher quality
    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);
    delay(1000);
    ESP.restart();
  }
}
```

- Hàm handNewMessages() xử lý khi có tin nhắn mới đến.

```
void handleNewMessages(int numNewMessages) {
  Serial.print("Handle New Messages: ");
  Serial.println(numNewMessages);
  for (int i = 0; i < numNewMessages; i++) {
    String chat_id = String(bot.messages[i].chat_id);
    if (chat_id != CHAT_ID){
      bot.sendMessage(chat_id, "Unauthorized user", "");
      continue;
    }
    // Print the received message
    String text = bot.messages[i].text;
    Serial.println(text);
    String from_name = bot.messages[i].from_name;
    if (text == "/start") {
      String welcome = "Welcome , " + from_name + "\n";
      welcome += "Use the following commands to interact with the ESP32-CAM \n";
      welcome += "/photo : takes a new photo\n";
      welcome += "/flash : toggles flash LED \n";
      bot.sendMessage(CHAT_ID, welcome, "");
    }
    if (text == "/flash") {
      flashState = !flashState;
      digitalWrite(FLASH_LED_PIN, flashState);
      Serial.println("Change flash LED state");
    }
    if (text == "/photo") {
      sendPhoto = true;
      Serial.println("New photo request");
    }
  }
}
```

Hàm kiểm tra tin nhắn gửi đến có phải từ CHAT_ID của bạn hay không? Nếu không phải thì bỏ qua, ngược lại thì sẽ thực hiện kiểm tra tin nhắn gửi đến bot.

* "/start": trả về các lệnh để điều khiển ESP thông qua phương thức sendMessage() với các tham số chat ID, tin nhắn vàchế độ phân tích văn bản (parse_mode – mặc định chuỗi rỗng là không sử dụng, Tin nhắn sẽ hiển thị văn bản theo cách mà nó được gửi mà không định dạng đặc biệt).

```
bool sendMessage(String chat_id, String text, String parse_mode = "");
```

* "/flash": đặt lại biến flashState, trước đó là LOW thì đặt lại thành HIGH, trước đó là HIGH thì đặt lại thành LOW.
* "/photo": đặt biến sendPhoto thành true.

- Hàm sendPhotoTelegram() chụp ảnh bằng ESP32-CAM.

```
camera_fb_t * fb = NULL;
fb = esp_camera_fb_get();
esp_camera_fb_return(fb); // dispose the buffered image

//Take a new photo
fb = NULL;
fb = esp_camera_fb_get();
if(!fb) {
  Serial.println("Camera capture failed");
  delay(1000);
  ESP.restart();
  return "Camera capture failed";
}
```

Đôi khi bức ảnh đầu tiên đẹp do cảm biến chưa điều chỉnh cân bằng màu trắng. Do đó loại bỏ bức ảnh đầu tiên.

```
    clientTCP.println("POST /bot"+BOTtoken+"/sendPhoto HTTP/1.1");
    clientTCP.println("Host: " + String(myDomain));
    clientTCP.println("Content-Length: " + String(totalLen));
    clientTCP.println("Content-Type: multipart/form-data; boundary=RandomNerdTutorials");
    clientTCP.println();
    clientTCP.print(head);
```

Thực hiện tạo yêu cầu HTTP POST để gửi ảnh đến telegram bot.

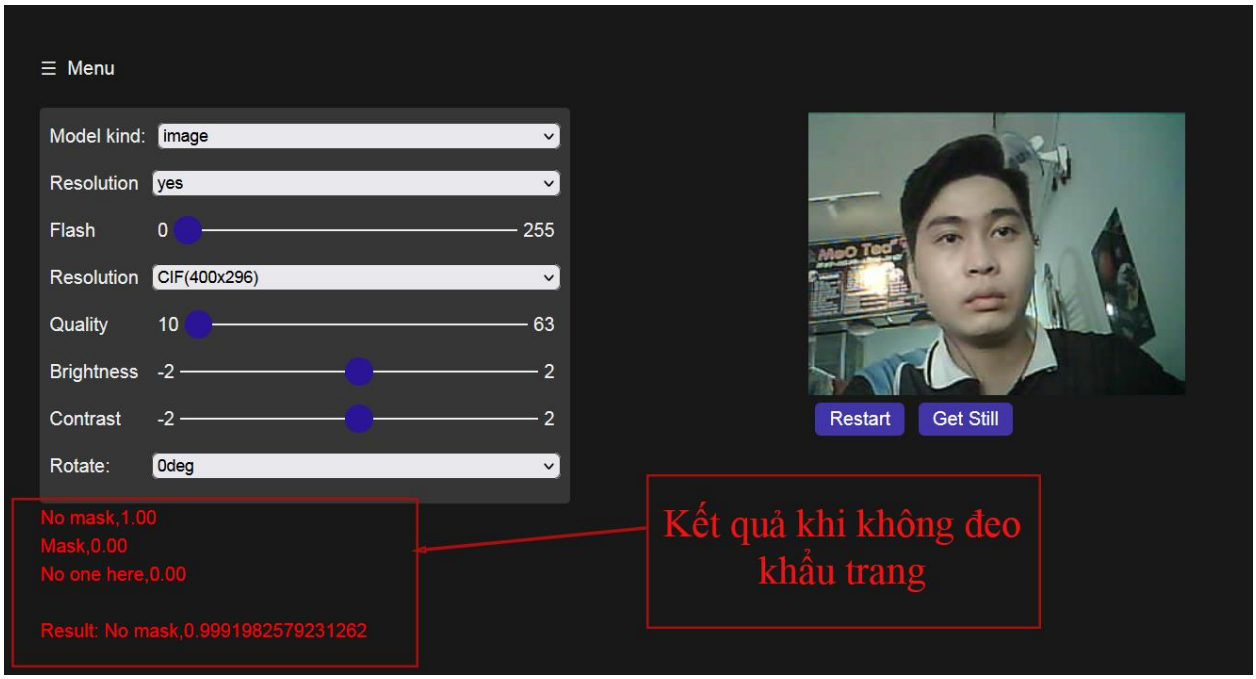
- Hướng dẫn trình duyệt cho phép hình ảnh Camera và Teachable Machine, có nguồn gốc khác nhau hoạt động cùng nhau trong chương trình.

```
    client.println("HTTP/1.1 200 OK");
    client.println("Access-Control-Allow-Origin: *");
    client.println("Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept");
    client.println("Access-Control-Allow-Methods: GET,POST,PUT,DELETE,OPTIONS");
    client.println("Content-Type: image/jpeg");
    client.println("Content-Disposition: form-data; name=\"imageFile\"; filename=\"picture.jpg\"");
    client.println("Content-Length: " + String(fb->len));
    client.println("Connection: close");
    client.println();
```

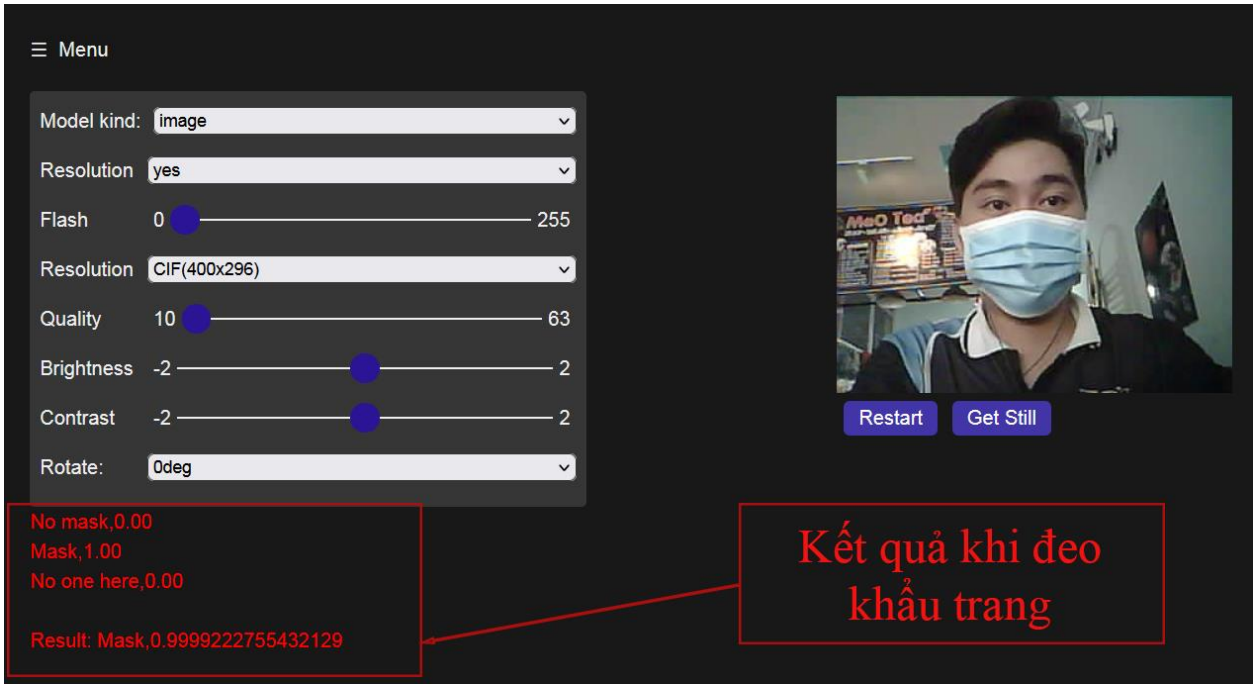
Chương IV. Kết quả.

- Mô hình nhận diện được khuôn mặt người không đeo khẩu trang, có đeo khẩu trang và các trường hợp còn lại.

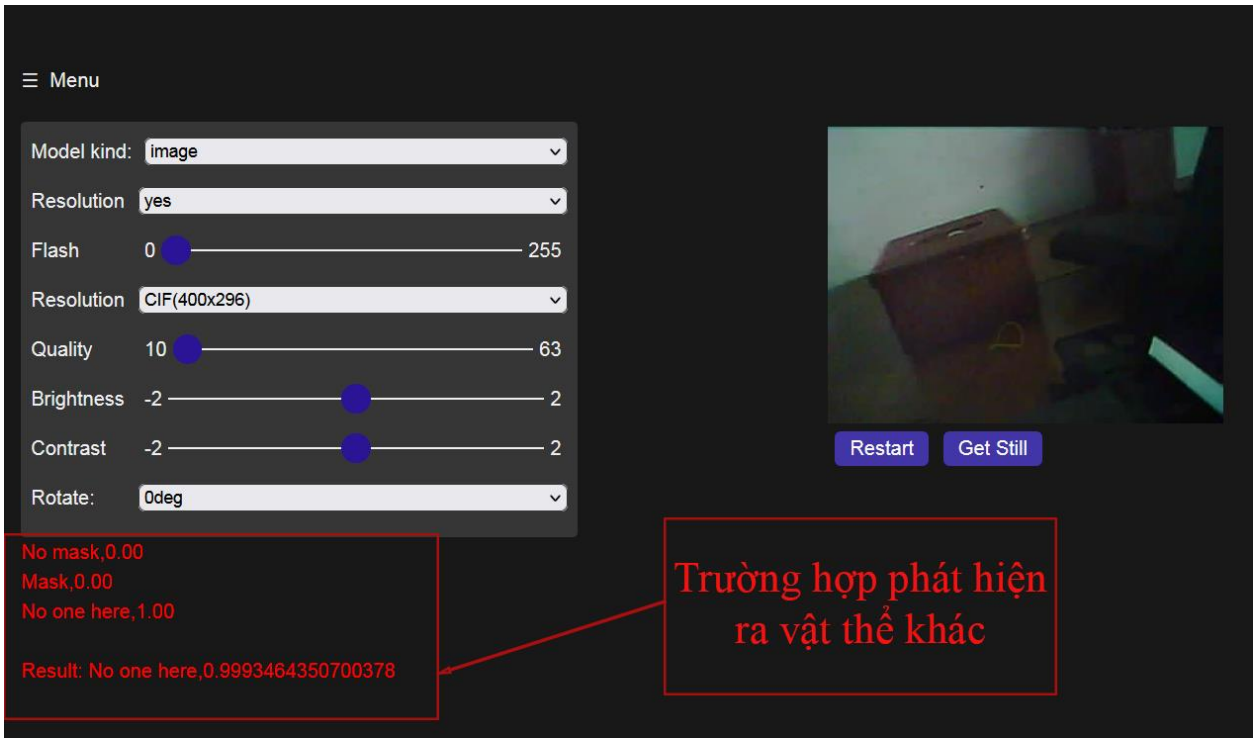
- Kết quả từ mô hình nhận diện không đeo khẩu trang là ACC = 100%.



- Kết quả từ mô hình nhận diện đeo khẩu trang là ACC = 100%.



- Kết quả từ mô hình nhận diện Không phải khuôn mặt người là ACC = 100%



- Kết quả gửi lên Telegram: Đeo khẩu trang an toàn.