<https://randomnerdtutorials.com/esp32-cam-troubleshooting-guide/>

**ESP32-CAM Video Streaming and Face Recognition with Arduino IDE**

We’ll show you how to setup a video streaming web server with face recognition and detection.



**Note:**in this tutorial we use the example from the arduino-esp32 library. This tutorial doesn’t cover how to modify the example.

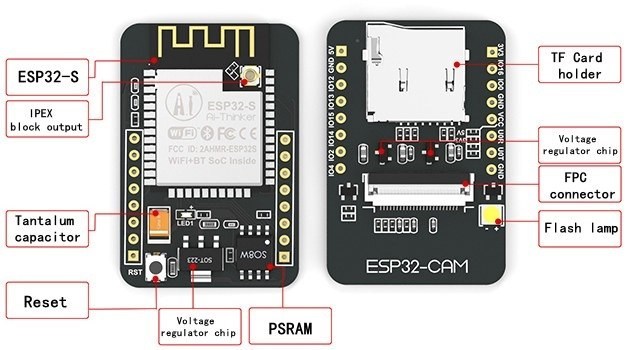
**Related project:** [ESP32-CAM Video Streaming Web Server](https://randomnerdtutorials.com/esp32-cam-video-streaming-web-server-camera-home-assistant/) (works with Home Assistant and Node-Red)

To follow this tutorial you need the following components:

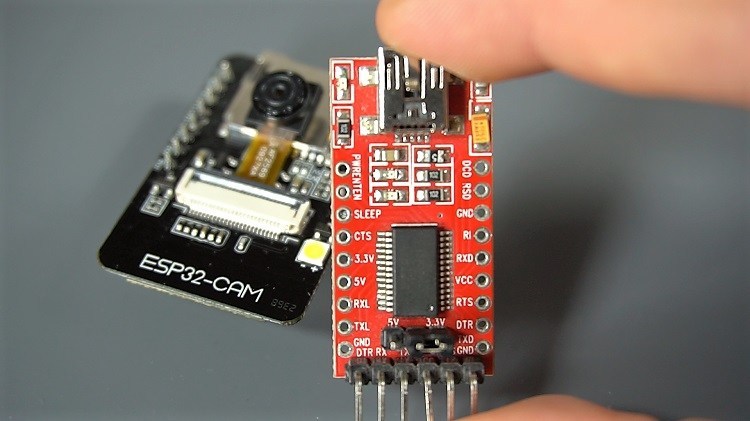
* [**ESP32-CAM with OV2640**](https://makeradvisor.com/tools/esp32-cam/)
* [FTDI programmer](https://makeradvisor.com/tools/ftdi-programmer-board/)
* [Female-to-female jumper wires](https://makeradvisor.com/tools/jumper-wires-kit-120-pieces/)

You can use the preceding links or go directly to [MakerAdvisor.com/tools](https://makeradvisor.com/tools/?utm_source=rnt&utm_medium=post&utm_campaign=post) to find all the parts for your projects at the best price!

The [ESP32-CAM](https://makeradvisor.com/tools/esp32-cam/) is a very small camera module with the ESP32-S chip that costs approximately $10. Besides the OV2640 camera, and several GPIOs to connect peripherals, it also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

[Image source – Seeed Studio](https://www.seeedstudio.com/media/catalog/product/cache/ef3164306500b1080e8560b2e8b5cc0f/b/a/bazaar1003542_esp32cam2.jpg)

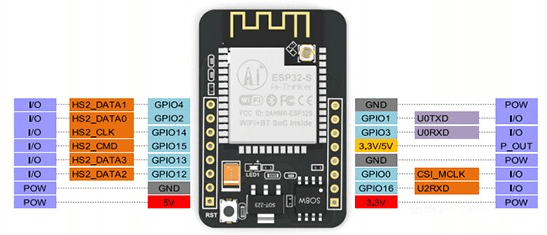
The [ESP32-CAM](https://makeradvisor.com/tools/esp32-cam/) doesn’t come with a USB connector, so you need an [FTDI programmer](https://makeradvisor.com/tools/ftdi-programmer-board/) to upload code through the U0R and U0T pins (serial pins).



Here is a list with the ESP32-CAM features:

* The smallest 802.11b/g/n Wi-Fi BT SoC module
* Low power 32-bit CPU,can also serve the application processor
* Up to 160MHz clock speed, summary computing power up to 600 DMIPS
* Built-in 520 KB SRAM, external 4MPSRAM
* Supports UART/SPI/I2C/PWM/ADC/DAC
* Support OV2640 and OV7670 cameras, built-in flash lamp
* Support image WiFI upload
* Support TF card
* Supports multiple sleep modes
* Embedded Lwip and FreeRTOS
* Supports STA/AP/STA+AP operation mode
* Support Smart Config/AirKiss technology
* Support for serial port local and remote firmware upgrades (FOTA)

The following figure shows the ESP32-CAM pinout (AI-Thinker module).

[](https://i1.wp.com/randomnerdtutorials.com/wp-content/uploads/2019/03/ESP32-CAM-pinout-1.png?ssl=1)

[Image source – Seeed Studio](https://www.seeedstudio.com/media/catalog/product/cache/ef3164306500b1080e8560b2e8b5cc0f/b/a/bazaar1003541_esp32cam3.jpg)

There are three GND pins and two pins for power: either 3.3V or 5V.

GPIO 1 and GPIO 3 are the serial pins. You need these pins to upload code to your board. Additionally, GPIO 0 also plays an important role, since it determines whether the ESP32 is in flashing mode or not. When GPIO 0 is connected to GND, the ESP32 is in flashing mode.

The following pins are internally connected to the microSD card reader:

* GPIO 14: CLK
* GPIO 15: CMD
* GPIO 2: Data 0
* GPIO 4: Data 1 (also connected to the on-board LED)
* GPIO 12: Data 2
* GPIO 13: Data 3

**Video Streaming Server**

Follow the next steps to build a video streaming web server with the ESP32-CAM that you can access on your local network.

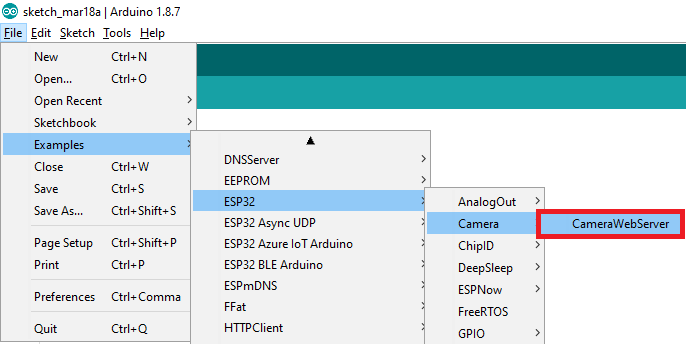
**1. Install the ESP32 add-on**

In this example, we use Arduino IDE to program the ESP32-CAM board. So, you need to have Arduino IDE installed as well as the ESP32 add-on. Follow one of the next tutorials to install the ESP32 add-on, if you haven’t already:

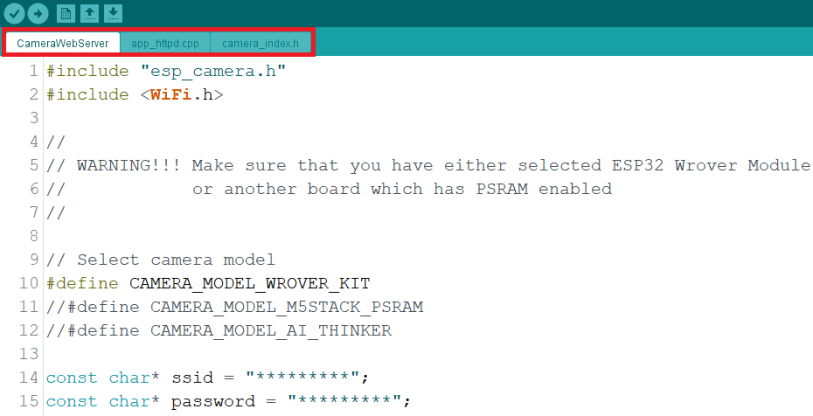
* [Installing the ESP32 Board in Arduino IDE (Windows instructions)](https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/)
* [Installing the ESP32 Board in Arduino IDE (Mac and Linux instructions)](https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-mac-and-linux-instructions/)

**2. CameraWebServer Example Code**

**File**> **Examples**> **ESP32**> **Camera**and open the **CameraWebServer** example.



The following code should load.



If you can’t find the code in your Arduino IDE, you can [download it from our GitHub repository](https://github.com/RuiSantosdotme/arduino-esp32-CameraWebServer). After downloading, unzip the folder, and open the *CameraWebServer.ino*file.

Before uploading the code, you need to insert your network credentials in the following variables:

const char\* ssid = "REPLACE\_WITH\_YOUR\_SSID";

const char\* password = "REPLACE\_WITH\_YOUR\_PASSWORD";

Then, make sure you select the right camera module. In this case, we’re using the AI-THINKER Model.

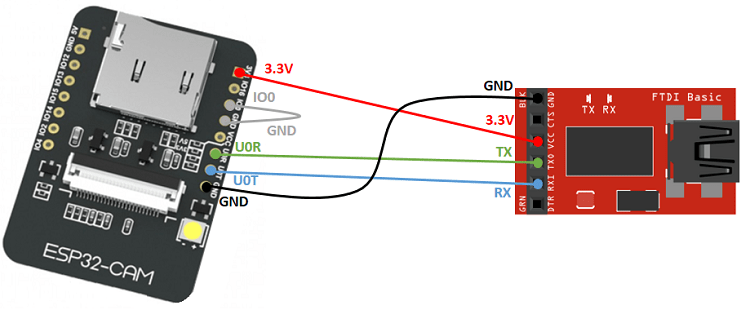
So, comment all the other models and uncomment this one:

#define CAMERA\_MODEL\_AI\_THINKER

Now, the code is ready to be uploaded to your ESP32

**ESP32-CAM Upload Code**

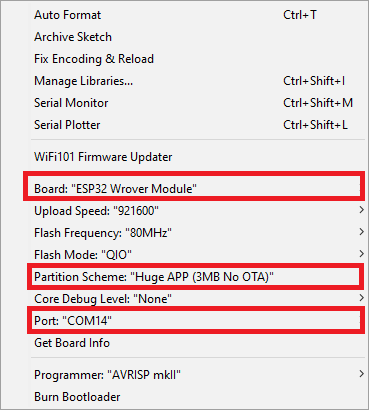
Connect the ESP32-CAM board to your computer using an FTDI programmer.



**Important:**GPIO 0 needs to be connected to GND so that you’re able to upload code.

To upload the code, follow the next steps:

1. Go to **Tools**> **Board**and select **ESP32 Wrover Module**
2. Go to **Tools**> **Port**and select the COM port the ESP32 is connected to
3. In **Tools**> **Partition Scheme**, select “**Huge APP (3MB No OTA)**“
4. Press the ESP32-CAM on-board RESET button
5. Then, click the upload button to upload the code



**Important:**if you can’t upload the code, double-check that GPIO 0 is connected to GND and that you selected the right settings in the **Tools**menu. You should also press the on-board Reset button to restart your ESP32 in flashing mode.

**Getting the IP address**

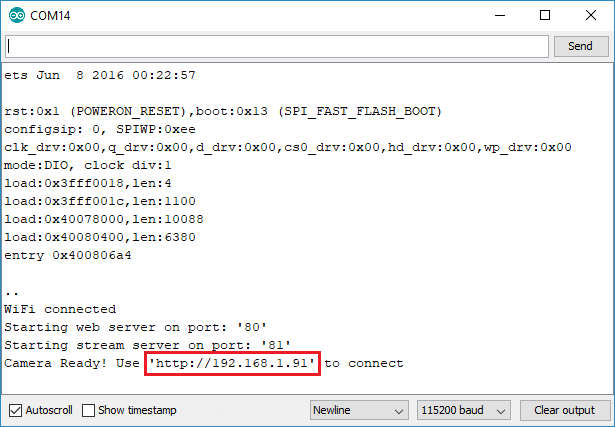
Después de cargar el código, desconecte GPIO 0 de GND

Abra el monitor serial a ratio de baudios 115,200

Presione en ESP32-CAM el botón de Reset.

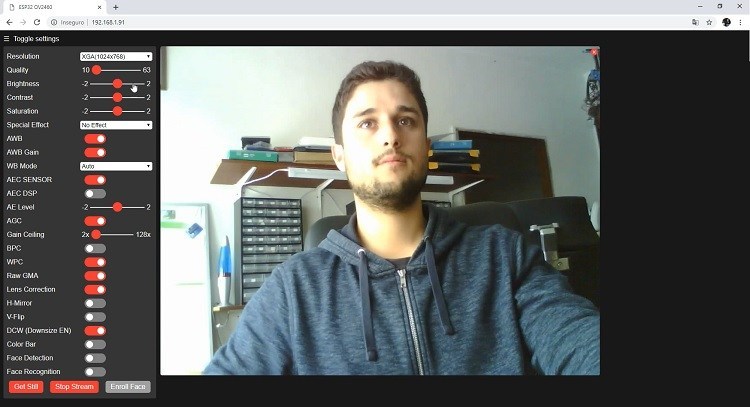
After uploading the code, disconnect GPIO 0 from GND.

Open the Serial Monitor at a baud rate of 115200.



**Accessing the Video Streaming Server**

Now, you can access your camera streaming server on your local network. Open a browser and type the ESP32-CAM IP address. Press the **Start Streaming** button to start video streaming.



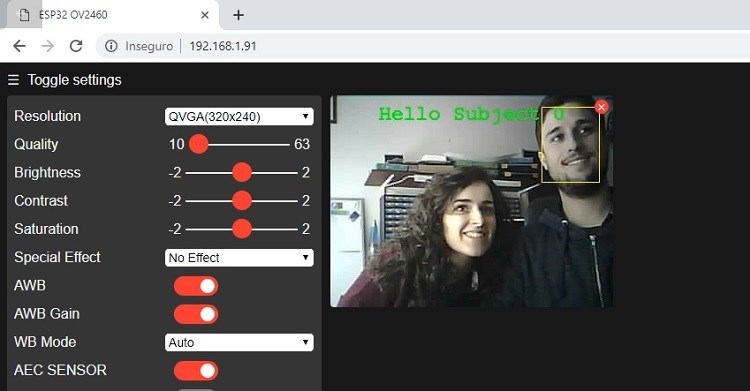
You also have the option to take photos by clicking the **Get Still** button. Unfortunately, this example doesn’t save the photos, but you can modify it to use the on board microSD Card to store the captured photos.

There are also several camera settings that you can play with to adjust the image settings.

Finally, you can do face recognition and detection.



First, you need to enroll a new face. It will make several attempts to save the face. After enrolling a new user, it should detect the face later on (subject 0).



And that’s it. Now you have your video streaming web server up and running with face detection and recognition with the example from the library.

**Troubleshooting**

If you’re getting any of the following errors, read our [**ESP32-CAM Troubleshooting Guide: Most Common Problems Fixed**](https://randomnerdtutorials.com/esp32-cam-troubleshooting-guide/)

* Failed to connect to ESP32: Timed out waiting for packet header
* Camera init failed with error 0x20001 or similar
* Brownout detector or Guru meditation error
* Sketch too big error – Wrong partition scheme selected
* Board at COMX is not available – COM Port Not Selected
* Psram error: GPIO isr service is not installed
* Weak Wi-Fi Signal
* No IP Address in Arduino IDE Serial Monitor
* Can’t open web server
* The image lags/shows lots of latency

**Wrapping Up**

In this tutorial we’ve tested the CameraWebServer example to test the camera functionalities. Now, the idea is to modify the example or write a completely new code to build other projects. For example, take photos and save them to the microSD card when motion is detected, [integrate video streaming in your home automation platform (like Node-RED or Home Assistant)](https://randomnerdtutorials.com/esp32-cam-video-streaming-web-server-camera-home-assistant/), and much more.

We hope you’ve find this tutorial useful. If you don’t have an ESP32-CAM yet, you can [grab it here](https://makeradvisor.com/tools/esp32-cam/).

If you like this project, you may also like other projects about cameras:

* [ESP32-CAM Video Streaming Web Server](https://randomnerdtutorials.com/esp32-cam-video-streaming-web-server-camera-home-assistant/) (works with Home Assistant and Node-RED)
* [ESP32-CAM Take Photo and Save to MicroSD Card](https://randomnerdtutorials.com/esp32-cam-take-photo-save-microsd-card/)
* [CCTV Raspberry Pi Based System with Storage using MotionEyeOS](https://randomnerdtutorials.com/cctv-raspberry-pi-based-system-storage-motioneyeos/)
* [Video Streaming with Raspberry Pi Camera](https://randomnerdtutorials.com/video-streaming-with-raspberry-pi-camera/)
* [Node-RED with Raspberry Pi Camera (Take Photos)](https://randomnerdtutorials.com/node-red-with-raspberry-pi-camera-take-photos/)