**SMART ATTENDANCE USING FACE RECOGNITION**

INTRODUCTION: -

Nowadays corporate houses and offices are adopting biometric attendance systems where the attendance is recorded by putting your finger on finger print sensor. These records are saved on cloud server to be monitored from anywhere by the authorities.

Here we are building an **IoT based Biometric attendance system** which can store the attendance records **in Google sheet**. An [ESP32 NodeMCU module](https://iotdesignpro.com/tags/nodemcu),  ausing Arduino IDE and Google spreadsheet will be used to keep the attendance log for future references. This Project can be very helpful in various corporate sectors, educational institutions, hospitals etc for attendance purpose.

This **Biometric Attendance System** is more secure and easier to use.

OBJECTIVE: -

To maintain the attendance record with day to day activities is a challenging task. There is always a chance of proxy in case of manual attendance. the following system is based on face recognition to maintain the attendance record. The daily attendance of students is recorded and stored.

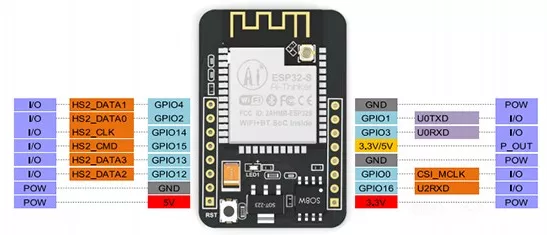
The system automatically starts taking snaps and then applies face detection and recognition technique to the given image and the recognized students are marked present and their attendance update with corresponding time is stored.

METHODOLOGY/PLANNING OF WORK:

1.Install the ESP32 and addon.

The [ESP32-CAM](https://makeradvisor.com/tools/esp32-cam/) is a very small camera module with the ESP32-S chip. 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.

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.



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

2.Camera web server.

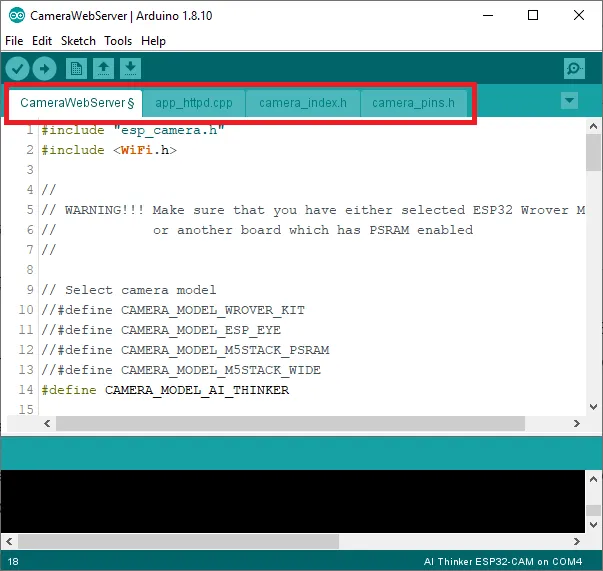
In your Arduino IDE, go to **File**> **Examples**> **ESP32**> **Camera**and open the **CameraWebServer** example.

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";

The following code should load.



3. A new partition scheme with persistent storage on the on-board flash is needed

Add this file to the directory containing the other partition schemes. This is found in one of two places, depending on how you installed the Arduino IDE.

Arduino IDE installed from the Windows Store:

C > Users > \*your-user-name\* > Documents > ArduinoData > packages > esp32 > hardware > esp32 > 1.0.4 > tools > partitions

Arduino IDE installed from the Arduino website:

C > Users > \*your-user-name\* > AppData > Local > Arduino15 > packages > esp32 > hardware > esp32 > 1.0.4 > tools > partitions

The new scheme has to be added to your ESP device in the board's manager configuration file – boards.txt. Again, this is found in one of two places.

Add this file to the directory containing the other partition schemes. This is found in one of two places, depending on how you installed the Arduino IDE.

Arduino IDE installed from the Windows Store:

C > Users > \*your-user-name\* > Documents > ArduinoData > packages > esp32 > hardware > esp32 > 1.0.4 > tools > partitions

Arduino IDE installed from the Arduino website:

C > Users > \*your-user-name\* > AppData > Local > Arduino15 > packages > esp32 > hardware > esp32 > 1.0.4 > tools > partitions

The new scheme has to be added to your ESP device in the board's manager configuration file – boards.txt. Again, this is found in one of two places.

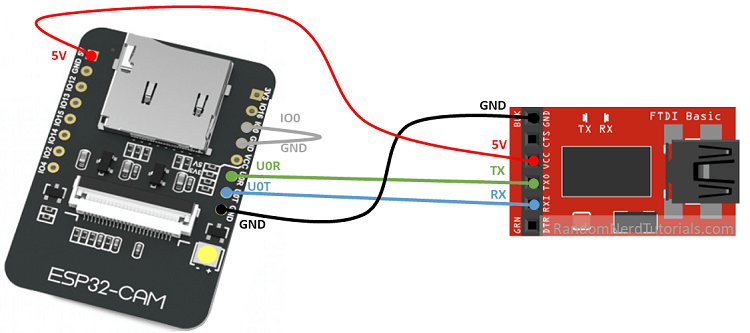
Add the following three lines below the existing partitionScheme options for the esp32wrover board in this boards.txt file.

esp32wrover.menu.PartitionScheme.rzo\_partition=Face Recognition (2621440 bytes with OTA)esp32wrover.menu.PartitionScheme.rzo\_partition. build.partitions=rzo\_partitionsesp32wrover.menu.PartitionScheme.rzo\_partition. upload.maximum\_size=2621440

Close and reopen the IDE to confirm the new ‘Face Recognition’ partition scheme is available in the Tools menu.

4.ESP32 cam upload code using FTDI program.

Connect the ESP32-CAM board to your computer using an FTDI programmer. Follow the next schematic diagram:



Many FTDI programmers have a jumper that allows you to select 3.3V or 5V. Make sure the jumper is in the right place to select 5V.

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

|  |  |
| --- | --- |
| **ESP32-CAM** | **FTDI Programmer** |
| GND | GND |
| 5V | VCC (5V) |
| U0R | TX |
| U0T | RX |
| GPIO 0 | GND |

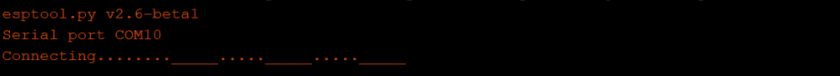
To upload the code, follow the next steps:

1) Go to **Tools**> **Board**and select **AI-Thinker ESP32-CAM**.

2) Go to **Tools**> **Port**and select the COM port the ESP32 is connected to.

3) Then, click the upload button to upload the code.

4) When you start to see these dots on the debugging window as shown below, press the ESP32-CAM on-board RST button.

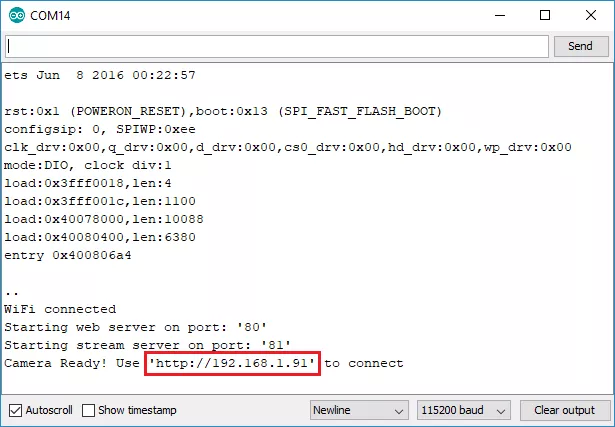


After a few seconds, the code should be successfully uploaded to your board.

5. Getting the Ip address.

After uploading the code, disconnect GPIO 0 from GND.

Open the Serial Monitor at a baud rate of 115200. Press the ESP32-CAM on-board Reset button.The ESP32 IP address should be printed in the serial monitor



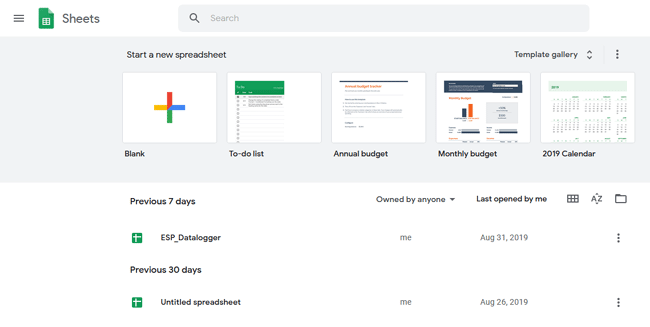
Now, you can access your camera streaming server on your local network. Open a browser and type the ESP32-CAM IP address. Press the **Start Streaming** button to start video streaming

Finally, you can do face recognition and detection

5. Working with google spread sheet.

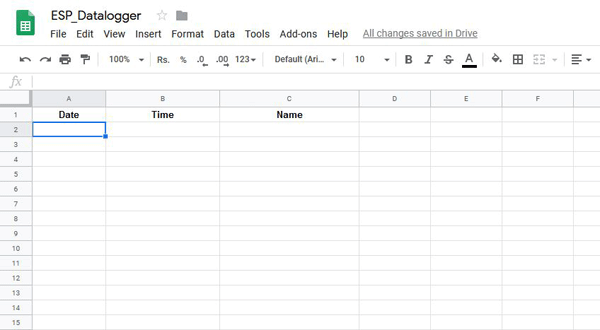
* Creating a new sheet

First login to Google docs with your Google account credentials and then select for Google sheet there and opt to “**Start a new spreadsheet**”.



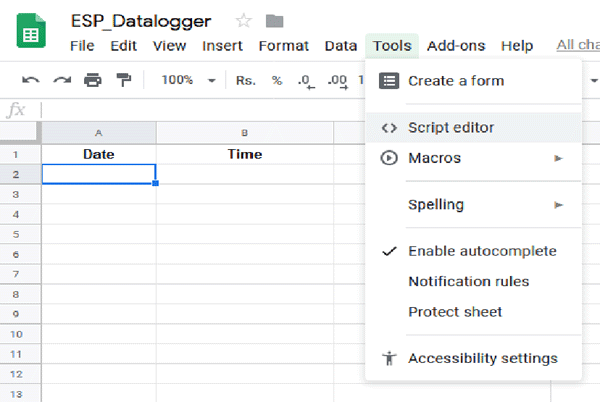
* Rename the sheet

Rename the blank sheet to any name of your choice. In my case, it is **ESP\_Datalogger**. Then create columns in the sheet for **Date**, **Time** and **Name**



* write functions to insert data into the sheet.

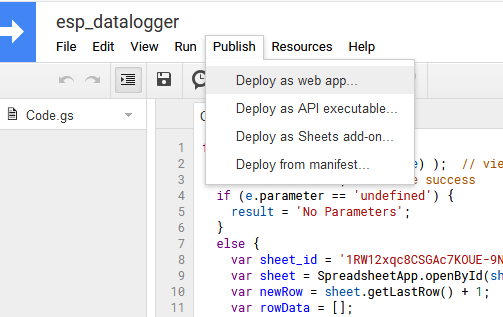
Now go to Tools and click on the option “**Script editor**” where we will write functions to insert data into the sheet.



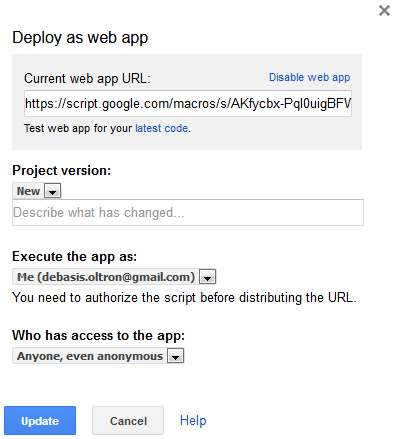
* paste the Google script.

The new Google Script is created with default name “Untitled project”. You can rename this Google Script File to any name of your choice. In my Case, I have renamed it to “esp\_datalogger”.

1. Go to **Publish**and select**“Deploy as web app”.**



**2. Select the “Project version” as “New”. Select “email id” in the “Execute the app as” field. Choose “Anyone, even anonymous” in the “Who has access to the app” field. And then Click on “Deploy”.**



3. In the next step, provide all the required permissions. Now you can see a new screen with a given link and named as “Current web app URL”. This URL contains Google Script ID. Just copy the URL and save it in notepad for future use.

EXPLANATION:

Initially, we have to install the library of ESP 32 on ARDUINO IDE. Our project FACE RECOGNITION ATTENDENCE is having 4 files: -

1. face attendance recognition – which is having the main code.

2. app\_http.cpp - which is used to recognise and enrol it.

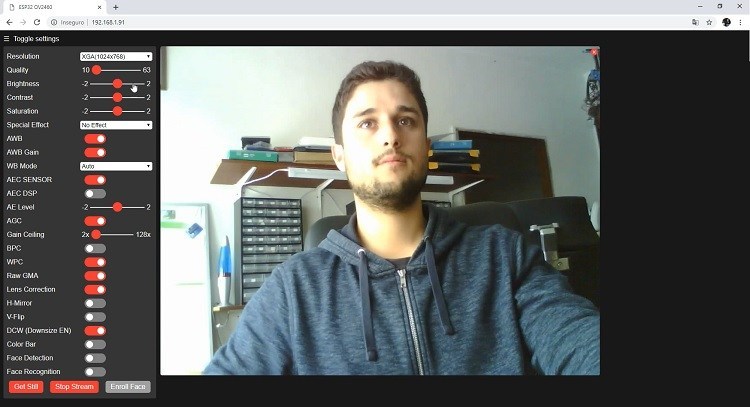
3.camera \_index.h - which is used to configure the camera.

4. camera\_pins.h - which is used to configure the camera.

Using GOOGLE SHEETS, a data base is created which will enrol the face and shows the face ID’s and later when it detects the face in the log sheet, it will show the date and name of the person.

When the web page is deployed, we get a link of our script link which is added in our code in ARDUINO.

When the code is uploaded to ESP32 cam module, it will print the local IP to use the camera, which allows us to enrol and detect faces



SUMMARY: -

In this project, we have built FACE RECOGNITION SYSTEM using ESP32 CAM which will also work as a security system by recognising the face of unauthorised persons.

So, by following the above steps an ESP32 NODEMCU module using Arduino IDE and successfully google spread sheet is generated. Therefore, ESP32cam is detecting and recognising the face and the corresponding details are stored in the google sheet and respective person name is printed. This is how a smart attendance using face recognition is executed.