instructables Super Cheap Security Camera with ESP32-cam Instruction Manual

instructables Super Cheap Security Camera with ESP32-cam Instruction Manual

Contents

Super Cheap Security Camera With ESP32-cam

by Giovanni Aggiustatutto

Today we are going to build this video surveillance camera that costs only 5€, like a pizza or an hamburger. This camera is connected to WiFi, so we will be able to control our home or what the camera sees from the phone anywhere, either on the local network or from outside. We will also add a motor that makes the camera move, so we can increase the angle that the camera can look. In addition to being used as a security camera, a camera like this can be used for many other purposes, such as checking to see if a 3D printer is working properly to stop it incase of problems. But now, let's get started

To see more details about this project, watch the video on my YouTube channel (it is in Italian but it has **English**

subtitles).
Supplies:

To build this camera we will need the ESP32 cam board, the tiny camera that is given with it, and a usb-to-serial adapter. The ESP32 cam board is a regular ESP32 with this little camera on it, all in one pcb. For those who don't know, the ESP32 is a programmable board similar to an Arduino, but with a much more powerful chip and the ability to connect to WiFi. This is why I have used the ESP32 for various smart home projects in the past. As I told you before the ESP32 cam board costs about €5 on Aliexpress.

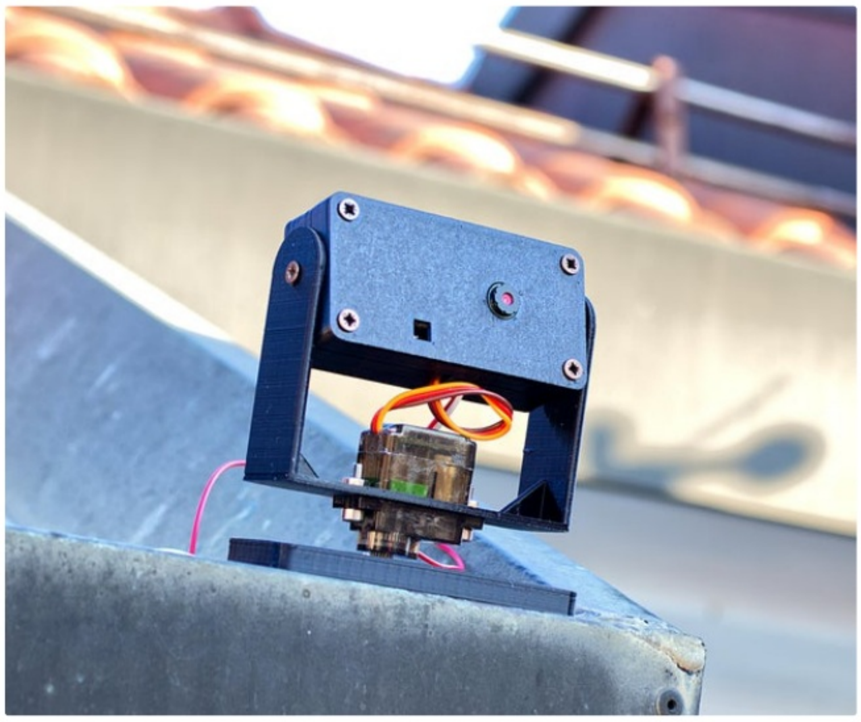
In addidtion to this, we will need:

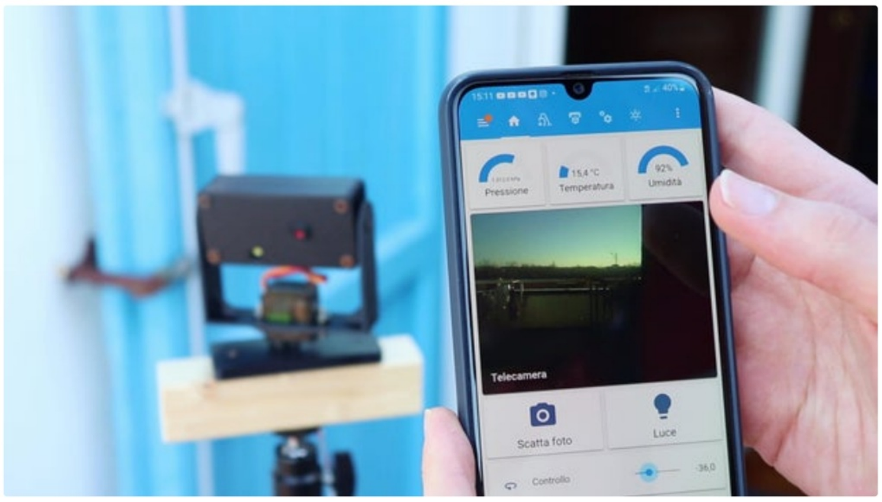
- a servo motor, which a motor that is able to reach a speci2c angle that is communicated to it by the microcontroller
- some wires

Tools:

- soldering iron (optional)
- 3D printer (optional)

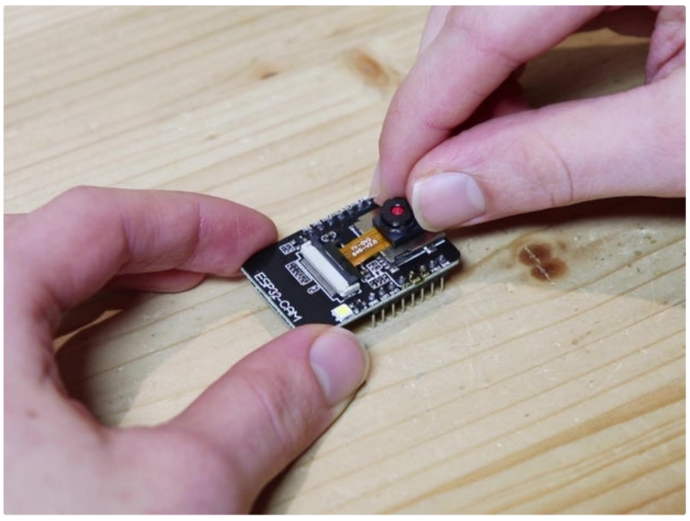
To see what the camera sees from the phone or computer and to take pictures we will use **Home Assistant** and ESPhome, but we will talk about that later.

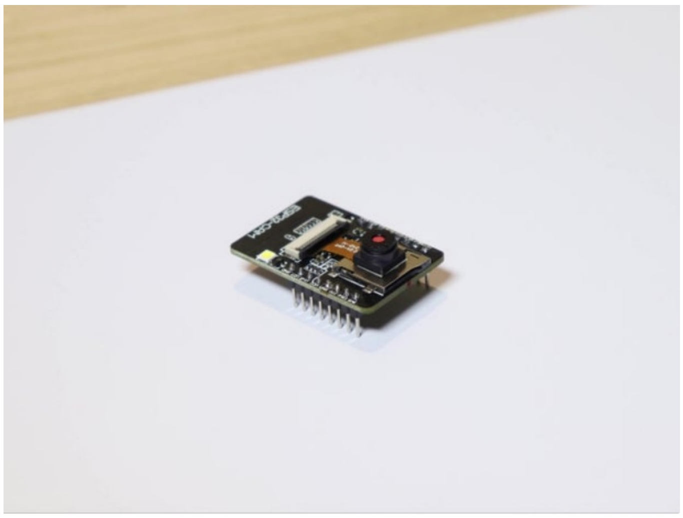




Step 1: Preparing ESP32-cam

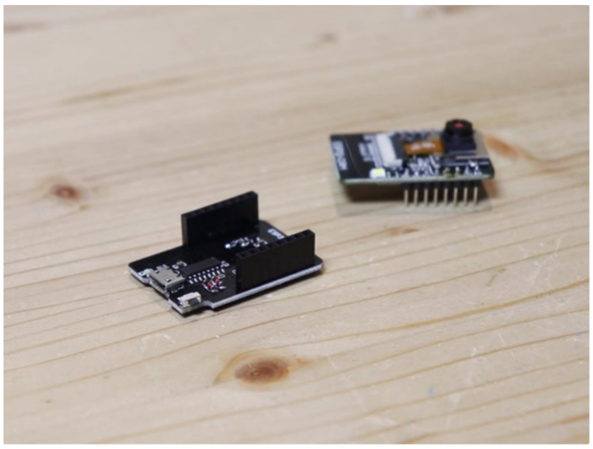
First you have to connect the camera to the board with the small connector, which is very fragile. Once you put the connector in you can lower the lever. Then I attached the camera on top of the board with a piece of double-sided tape. The ESP32 cam also has the ability to insert a micro SD, and although we will not use it today it allows us to take pictures and save them directly there.

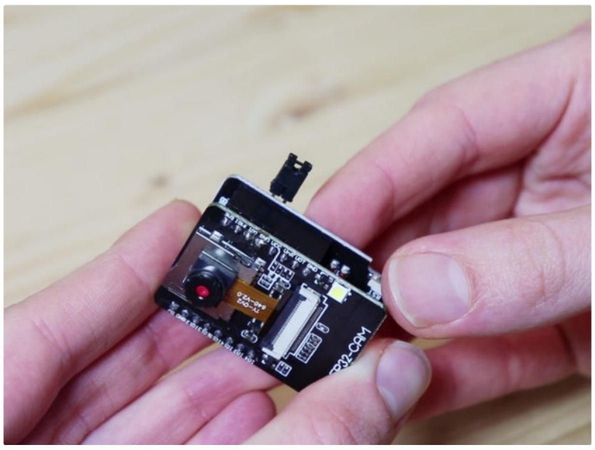




Step 2: Uploading Code

Usually Arduino and ESP boards also have a usb socket to load the program from the computer. However, this one does not have a usb socket, so to connect it to the computer to load the program you need a usb-to-serial adapter, which communicates with the chip directly through the pins. The one I found is made speci2cally for this type of board, so it simply connects to the pins without having to make any other connections. However, universal usb-to-serial adapters should also be 2ne. To load the program you also have to connect pin 0 to ground. To do this I soldered a jumper connector to these two pins. So when I need to program the board I just put a jumper between the two pins.





Step 3: Connecting the Camera to Home Assistant

But now let’s take a look at the software that will operate the camera. As I told you before, the camera will be connected to Home Assistant. Home Assistant is a home automation system that works locally which allows us to control all our home automation devices like smart bulbs and sockets from one interface.

To run Home Assistant I use and old Windows PC running a virtual machine, but if you have it you can use a Raspberry pi, which consumes less power. To see the data from your smartphone you can download the Home Assistant app. To connect from outside the local network I’m using the Nabu Casa Cloud, which is the simplest solution but it’s not free. There are other solutions but they are not totally safe.

So from the Home Assistant app we will be able to see the camera live video. To connect the camera to Home Assistant we will use ESPhome. ESPhome is an add-on that allows us to connect ESP boards to Home Assistant via WiFi. To connect the ESP32-cam to ESPhome you can follow these steps:

- Install the ESPhome plugin in Home Assistant
- On ESPhome’s dashboard, click on New device and on Continue
- Give your device a name
- Select ESP8266 or the board you used
- Copy the encryption key that is given, we will need it later
- Click on EDIT to see the device’s code
- Under esp32: paste this code (with framework: and type: commented)

esp32

board: esp32cam
#framework:
# **type:** arduino

- Under with, insert your wi2 ssid and password
- To make the connection more stable, you can give the board a static IP address, with this code:

wifi:

ssid: yourssid
password: yourwifipassword

```
manual_ip

# Set this to the IP of the ESP
static_ip: 192.168.1.61
# Set this to the IP address of the router. Often ends with .1
gateway: 192.168.1.1
# The subnet of the network. 255.255.255.0 works for most home networks.
subnet: 255.255.255.0
```

- At the end of the code, paste this one:

```
2_camera:
name: Telecamera 1
external_clock:
pin: GPIO0
frequency: 20MHz
i2c_pins:
sda: GPIO26
scl: GPIO27
data_pins: [GPIO5, GPIO18, GPIO19, GPIO21, GPIO36, GPIO39, GPIO34, GPIO35]
vsync_pin: GPIO25
href_pin: GPIO23
pixel_clock_pin: GPIO22
power_down_pin: GPIO32
resolution: 800×600
jpeg_quality: 10
vertical_flip: False
output:
– platform: gpio
pin: GPIO4
id: gpio_4
– platform: ledc
id: pwm_output
pin: GPIO2
frequency: 50 Hz
light:
– platform: binary
output: gpio_4
name: Luce telecamera 1
number:
– platform: template
name: Servo Control
min_value: -100
max_value: 100
step: 1
optimistic: true
set_action:
then:
– servo.write:
id: my_servo
level: !lambda ‘return x / 100.0;’
servo:
– id: my_servo
output: pwm_output
transition_length: 5s
```

The 2rst part of the code, under esp32_camera:, de2nes all the pins for the actual camera. Then with light: is de2ned the camera's led. At the end of the code is de2ned the servo motor, and the value used by the servo to set the rotation angle is read from Home Assistant with number:.

In the end the code should look like this, but **not paste directly the code below,** to every device is given a different encryption key.

```
phome:
name: camera-1
esp32:
board: esp32cam
#framework:
# type: arduino
# Enable logging

ger:
# Enable Home Assistant API
api:
encryption:
key: “encryptionkey”
ota:
password: “password”
wifi:
ssid: “yourssid”
password: “yourpassword”
# Enable fallback hotspot (captive portal) in case wifi connection fails
ap:
ssid: “Camera-1 Fallback Hotspot”
password: “password”
captive_portal:
esp32_camera:
name: Telecamera 1
external_clock:
pin: GPIO0
frequency: 20MHz
i2c_pins:
sda: GPIO26
scl: GPIO27
data_pins: [GPIO5, GPIO18, GPIO19, GPIO21, GPIO36, GPIO39, GPIO34, GPIO35]
vsync_pin: GPIO25
href_pin: GPIO23
pixel_clock_pin: GPIO22
power_down_pin: GPIO32
resolution: 800×600
jpeg_quality: 10
vertical_flip: False
output:
– platform: gpio
pin: GPIO4
id: gpio_4
– platform: ledc
id: pwm_output
pin: GPIO2
frequency: 50 Hz
light:
– platform: binary
output: gpio_4
name: Luce telecamera 1
number:
```

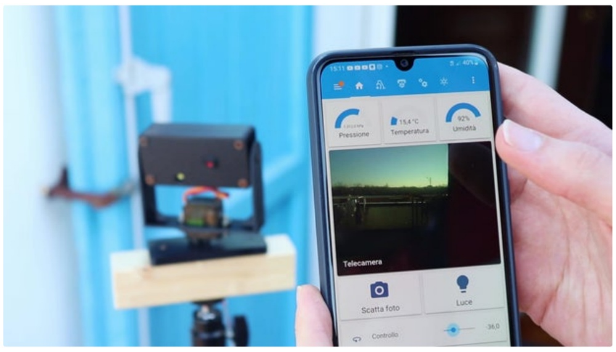
```
– platform: template
name: Servo Control
min_value: -100
max_value: 100
step: 1
optimistic: true
set_action:
then:
– servo.write:
id: my_servo
level: !lambda ‘return x / 100.0;’
```

Super Cheap Security Camera With ESP32-cam: Page 12
Step 4: Connections

```
servo:
– id: my_servo
output: pwm_output
transition_length: 5s
```

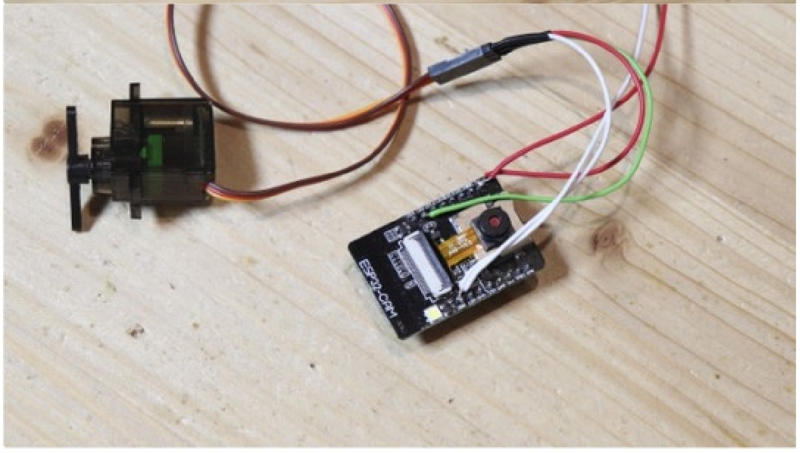
- After the code is complete, we can click on Install, connect the serial adapter of the ESP32 to our computer with an USB cable and follow the instructions on screen to upload the code like you have seen in the last step (it’s pretty easy!)
- When the ESP32-cam is connected to the WiFi, we can go to the Home Assistant settings, where we will probably see that Home Assistant has discovered the new device
- Click on configure and paste there the encryption key you have copied before.

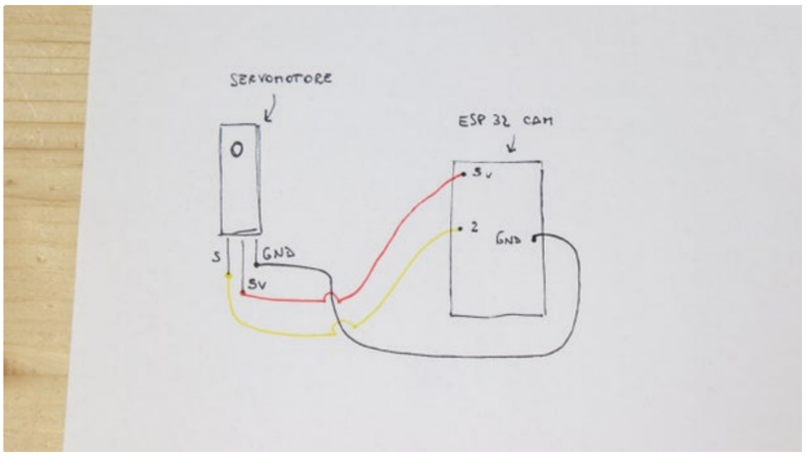
Once the program is loaded you can **remove the jumper between ground and pin 0,** and power up the board (if the jumper is not removed the board won’t work). If you look at the device’s logs, you should see that the ESP32-cam connects to the WiFi. In the following steps we will see how to con2gure the Home Assistant dashboard to see the live video from the camera, to move the motor and to take photos from the camera



Step 4: Connections

Once we have programmed the ESP32 we can remove the usb to serial adapter and power the board directly from the 5v pin. And at this point the camera only lacks an enclosure in which to mount it. However, leaving the camera standing still is boring, so I decided to add a motor to make it move. Speci2cally, I will use a servo motor, which is able to reach a speci2c angle that is communicated to it by the ESP32. I connected the brown and red wires of the servomotor to the power supply, and the yellow wire which is the signal to pin 2 of ESP32. In the picture above you can 2nd the schematics.



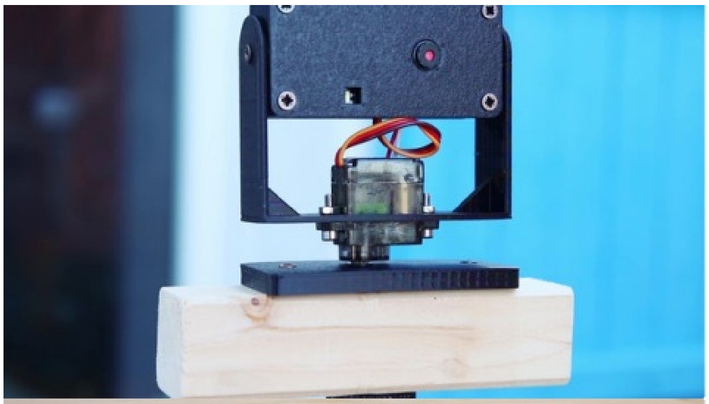
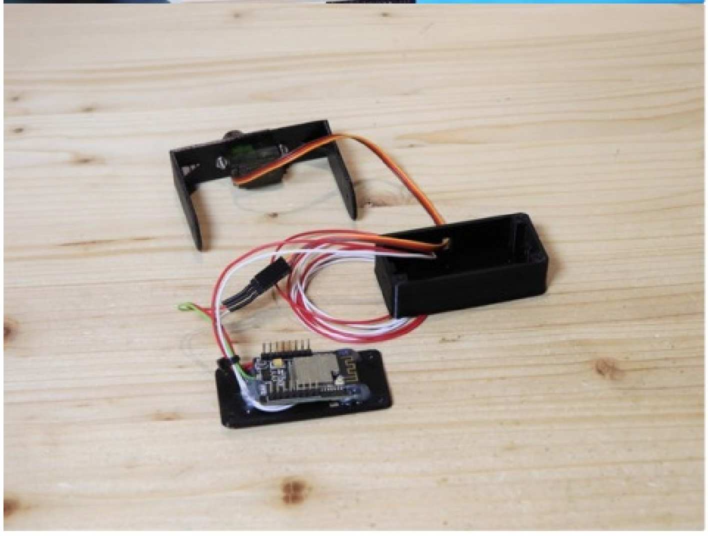
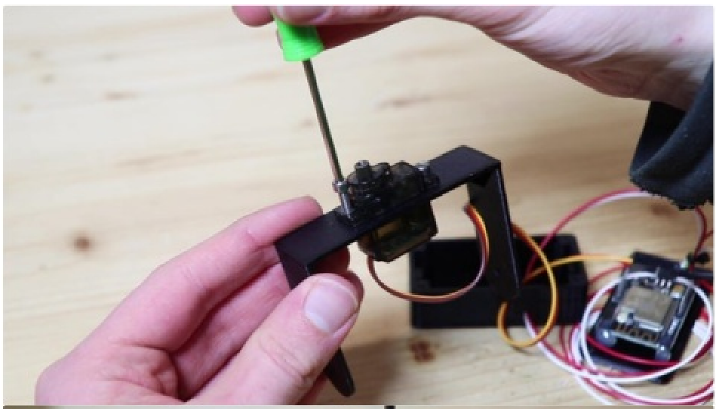
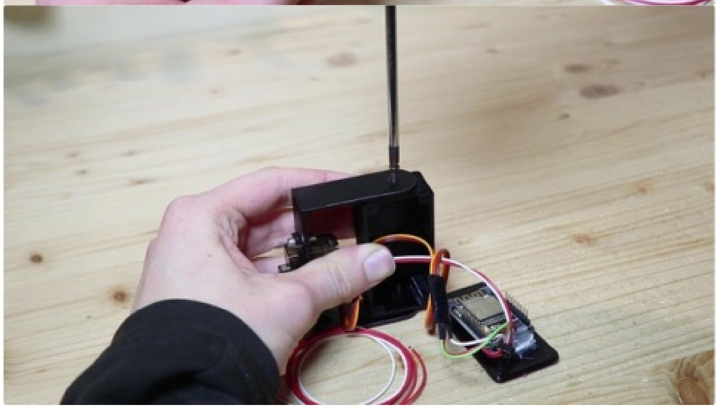


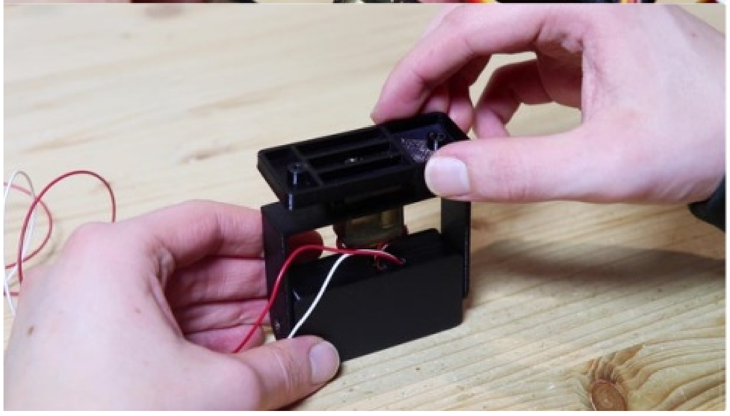
Step 5: Building the Enclosure

Now I need to turn the test circuit into something that looks more like a 2nished product. So I designed and 3D printed all the parts to make the little box in which to mount the camera. Below you can 2nd the .stl 2les for 3D printing. Then soldered the wires for the power supply and servo motor signal to the pins on the ESP32. To connect the servomotor connector, I soldered a jumper connector to the wires. So the circuit is 2nished, and as you can see it is quite simple.

I ran the servomotor and power wires through the holes on the little box. Then I glued the ESP32 cam to the cover, aligning the camera with the hole. I mounted the servo motor on the bracket that will hold the camera up, and secured it with two bolts. I attached the bracket to the small box with two screws, so that the camera could be tilted. To prevent the screws inside from touching the cables, I protected them with heat shrink tubing. Then I closed the cover with the camera with four screws. At this point it only remains to assemble the base. I ran the servo motor shaft through the hole in the base, and screwed the small arm to the shaft. Then I glued the arm to the base. This way the servomotor is able to move the camera 180 degrees.

And so we 2nished building the camera. To power it we can use any 5v power supply. Using the holes in the base, we can screw the camera to a wall or wooden surface.



Step 6: Setting Up Home Assistant Dashboard

To see the live video from the camera, move the motor, turn the led on and move the motor from the Home Assistant interface we need four cards in the dashboard of Home Assistant.

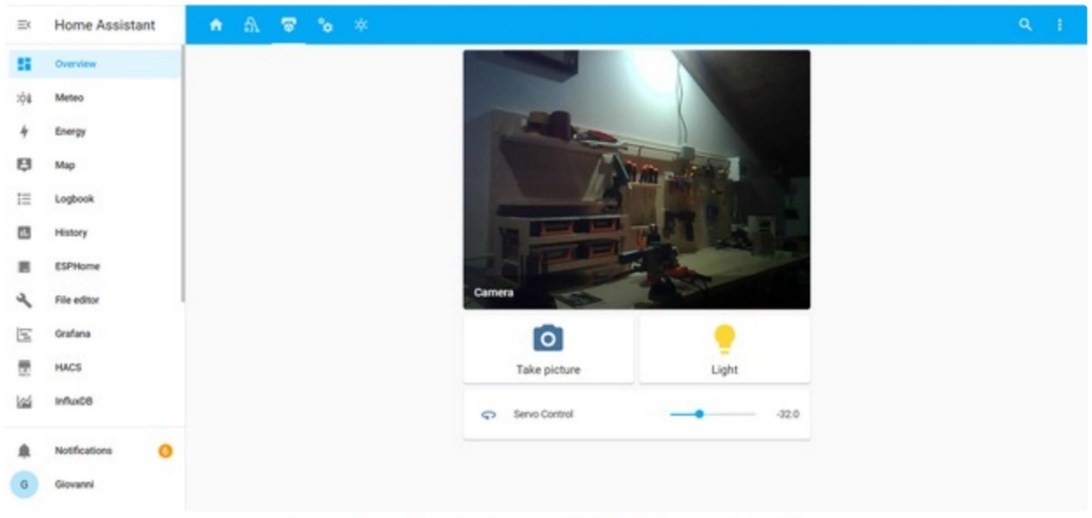
- The 2rst one is a picture glance card, that allows to see the live video from the camera. In the card's settings, just select the camera's entity and set Camera View to auto (this is important because if you set it to live the camera always sends the video and overheats).
- Then we need a button to take photos from the camera. This is a bit more di@cult. First we have to go in the

File Editor add-on (if you don't have it you can install it from the add-on store) in the con2g folder and create a new folder to save the photos, in this case called camera. The code for the text editor for the button is below.
ow_name: true

```
show_icon: true
type: button
tap_action:
action: call-service
service: camera.snapshot
data:
filename: /config/camera/telecamera_1_{{ now().strftime("%Y-%m-%d-%H:%M:%S") }}.jpg
#change the entity name above with the name of the entity of your camera
target:
entity_id:
– camera.telecamera_1 #change the entity name with the name of the entity of your camera
name: Take photo
icon_height: 50px
icon: mdi:camera
hold_action:
action: no
```

- The camera also has an led, even if it is not capable of lighting an entire room. For this I used an other button card, that toggles the led's entity when it is pressed.
- The last card is an entities card, that I set up with the servo motor entity. So with this card we have a very simple slider to control the angle of the motor and to move the camera.

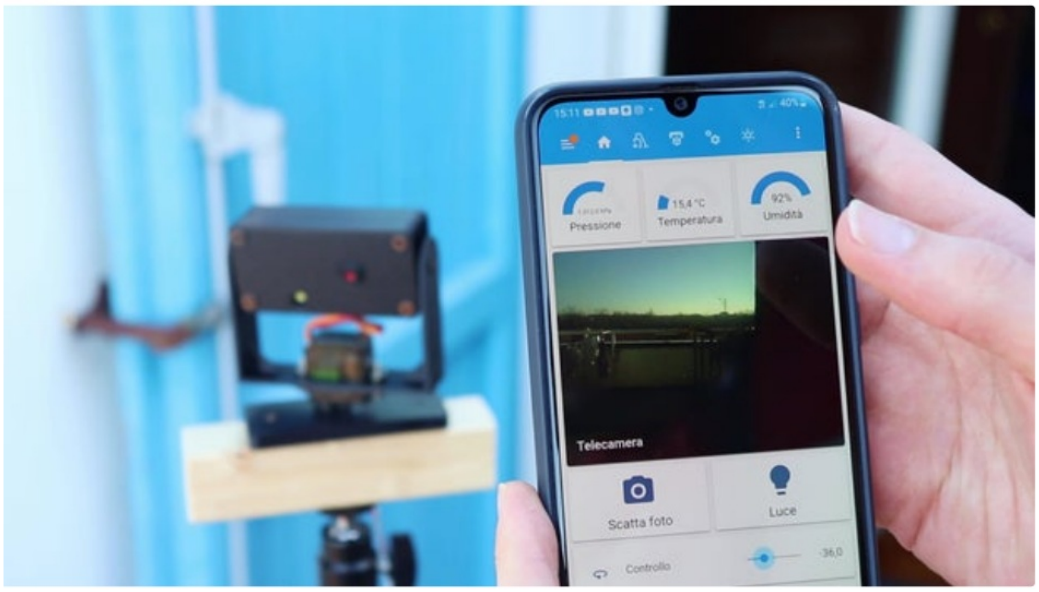
I organized my cards in a vertical stack and in a horizontal stack, but this is totally optional. However your dashboard should look similar to the one shown in the picture above. Of course you can customize the cards even more, to meet your needs.



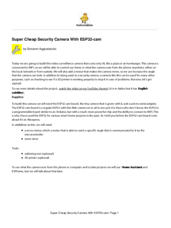
Step 7: It Works!

Finally, the camera works, and on the Home Assistant app I can see what the camera sees in real time. From the app I can also make the camera move by moving the slider, to look at a larger space. As I said before the camera also has an LED, although the light it makes does not allow you to see at night. From the app you can take pictures from the camera, but you cannot take videos. The pictures taken can be seen in the folder we have created before in Home Assistant. To take the camera to the next level, you can connect the camera to a motion sensor or a door opening sensor, which when it detects motion will take a picture with the camera.

So, this is the ESP32 cam security camera. It is not the most advanced camera, but for this price you can't 2nd anything better. I hope you enjoyed this guide, and maybe you found it useful. To see more details about this project, you can 2nd the video on my YouTube channel (it is in Italian but it has English subtitles).



Documents / Resources

	[instructables Super Cheap Security Camera with ESP32-cam](#) [pdf] Instruction Manual Super Cheap Security Camera with ESP32-cam, Super Cheap Security Camera, ESP32-cam, Cheap Security Camera, Security Camera, Camera

References

- [Yours for the making - Instructables](#)
- [Giovanni Aggiustatutto's Profile - Instructables](#)
- [Super Cheap Security Camera With ESP32-cam : 7 Steps (with Pictures) - Instructables](#)
- [content.instructables.com/F4Q/THF3/LE1ECREO/F4QTHF3LE1ECREO.stl](#)