Alarm Project
Part II Camera Control

Project objective

To create a low-cost security camera that can be triggered wirelessly from an alarm system to capture either video or a series of stills, store them locally and then upload to cloud storage.

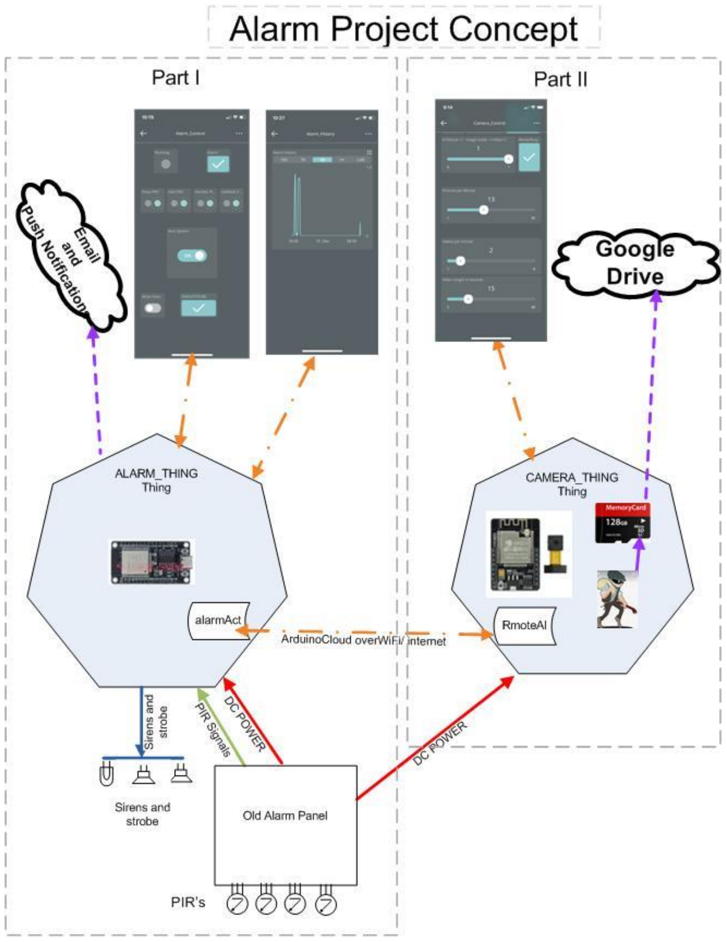
Background

To add a camera to Alarm Project Part 1.

System design requirements:

- Be powered from the exiting alarm system via PIR wiring.
- Use a mobile phone for all control and monitoring from anywhere.
- Connect to an internet based IoT service for monitoring, control, and communication with other IoT devices.
- To capture and store videos or still images when triggered by an alarm.
- To upload all images/videos to cloud .
- Be capable of receiving firmware updates via Wi-Fi, known as over the air (OTA).

Overall project concept including Part I



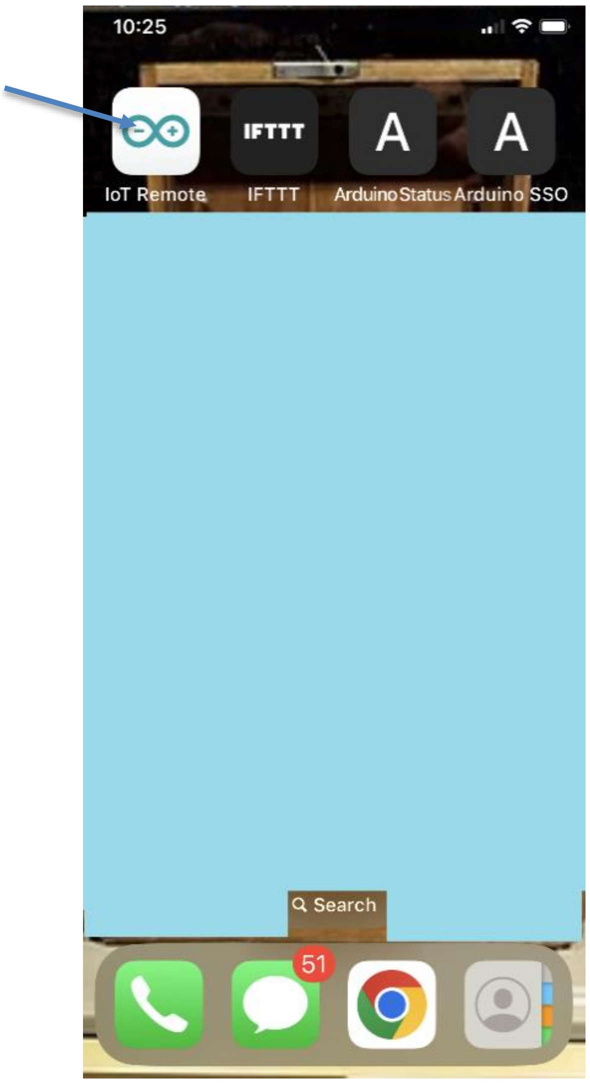
Finished project operation .

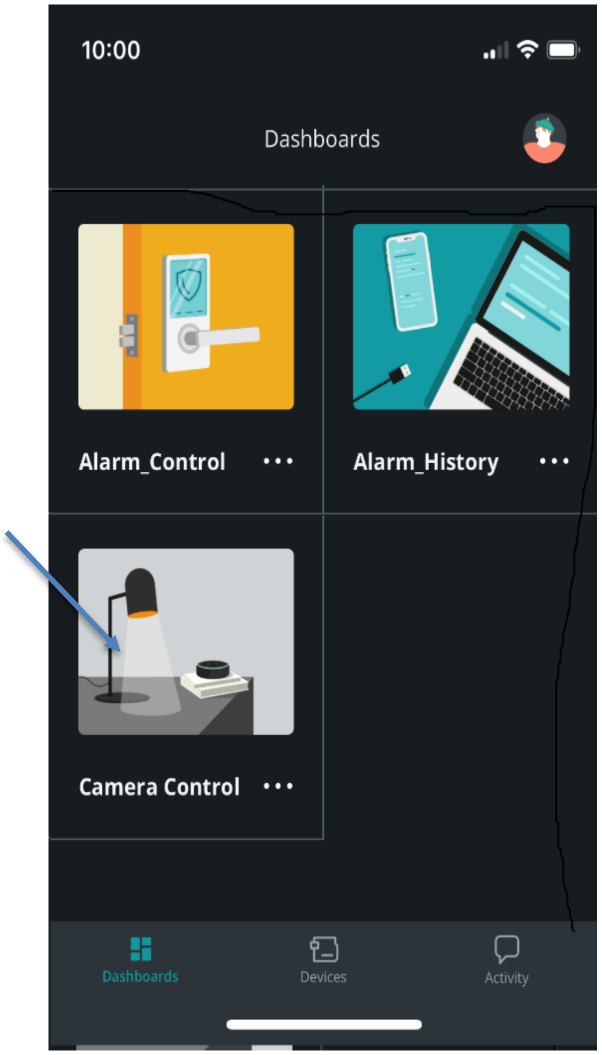
The system is very simple to operate, it can be controlled and monitored with a mobile phone from anywhere as long as there is coverage and internet..

Camera system operation

Start by opening the Camera control panel on your mobile phone.

Open the IoT Remote App , then select the Camera_Control from the dashboards.





Camera controls

Saved (indicator)
When showing as a green tick, the current are saved in the camera. A red means changes have been made that are not saved..

Save Changes button
Saves the current setting to be the new default, and sets saved indicator will change green tick.

Running
Blinks green once per second, if its not blinking there is a problem. The camera could be powered off or faulty, the local WiFi or internet or other critical service could be down where the camera is located.

Image Mode selector
Used to change the camera from image mode (jpg files) to video mode (avi files).*

Ready/Busy indicator
Will show a green tick when the system is ready. (not filming or sending images to google drive) Shows as red X when busy.
* note Changing the Image Mode is not possible if the system busy.

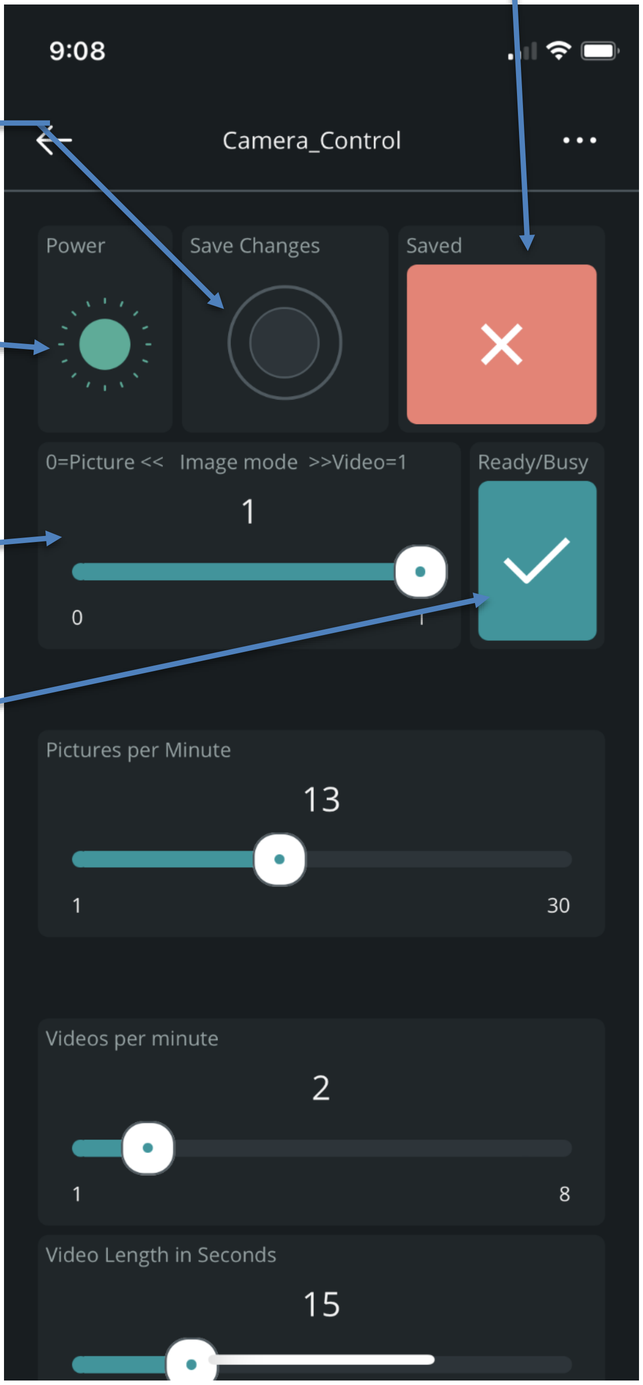
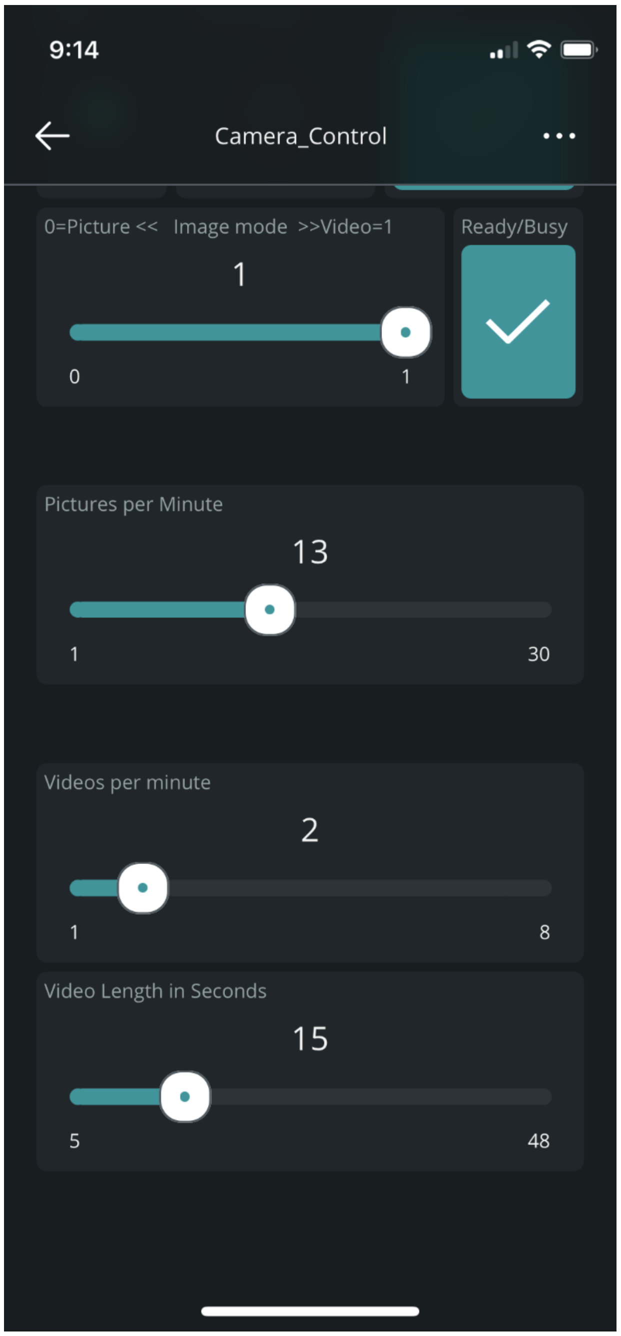


Image Controls


9:14

Camera_Control

0=Picture << Image mode >>Video=1

1

0 1

Ready/Busy ✓

Pictures per Minute

13

1 30

Videos per minute

2

1 8

Video Length in Seconds

15

5 48

Pictures per Minute

When in picture mode set the number of images (jpg files) taken every minute while in an alarm condition. Range 1 to 30.

Videos per Minute

When in video mode sets the number of videos (avi files) taken every minute while in an alarm condition. Range 1 to 8.*

Videos Length in Seconds

When in video mode sets the length of each of video (avi file) taken. Range 5 to 48.*

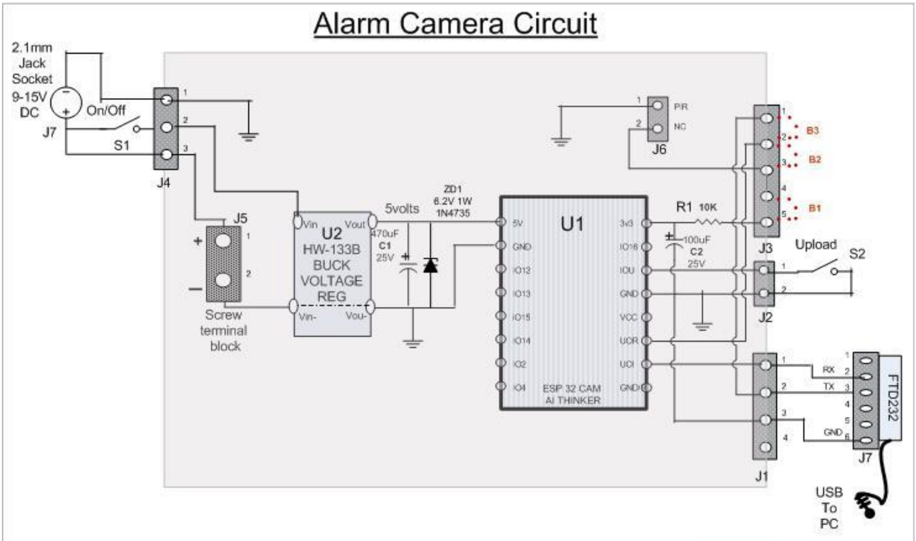
. * To ensure that .avi files do not become too large to deal with. The system limits the combined length of video per minute to 48 seconds. So you could have 2x24 = 48 seconds, but not 3x20 =60 seconds. In the later case the camera will automatically set the length to 16 seconds.

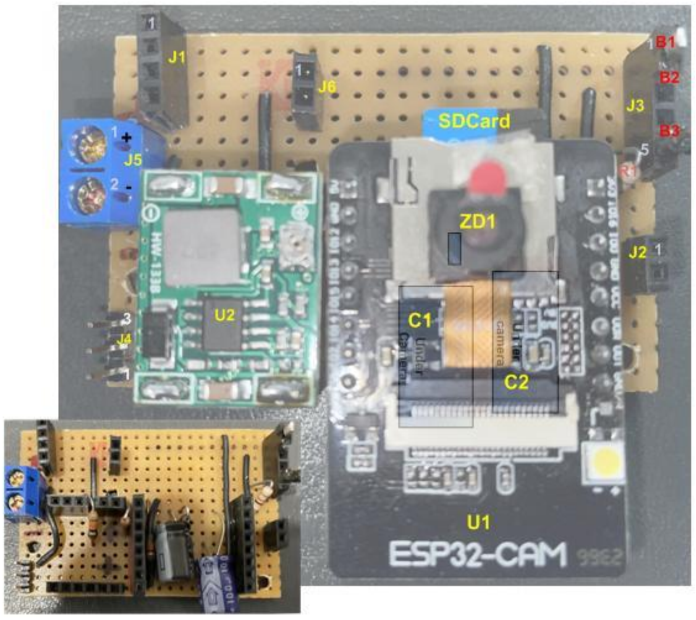
Circuit description

The design is based on an ESP32-CAM AI-Thinker type module. If you want to use the FLASH LED with the camera there is a small modification to use GPIO33 to drive the LED. The mod is detailed in C:\CameraProject\ESPCAM_MODS.jpg

A buck stepdown volage regulator is used to drop the alarm system DC power supply (nominally 13.6 volts) back to 5V for the ESP module. Power in is switched and can be fed via screw terminal block or 2.1mm power jack. IOU pin is switchable to ground to allow changing the ESP to upload mode. A socket is provided to allow connection of an FTD232 for programming. Provision is made for connection of a PIR instead of using an ArduinoCloud variable to signal an alarm, but there is currently no firmware support for that option.

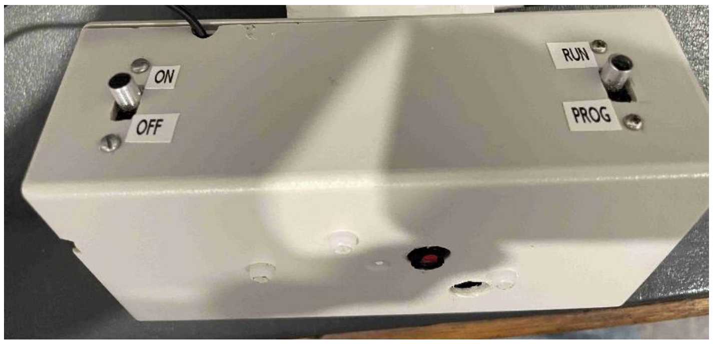
Hardware jumpers are provisioned to allow configuration of the potential PIR connection.
For normal operation just leave B1 closed, B2 and B3 open

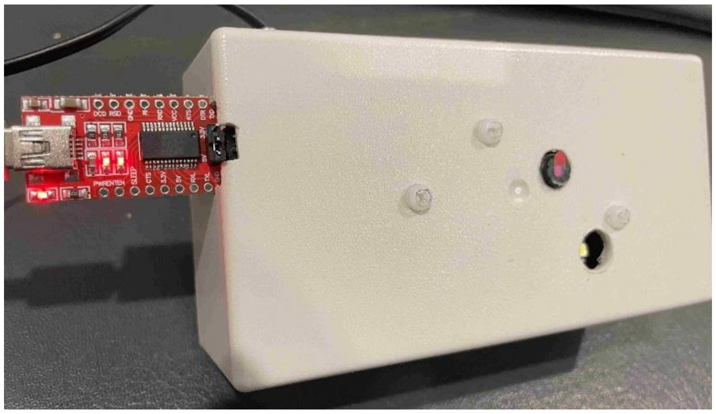


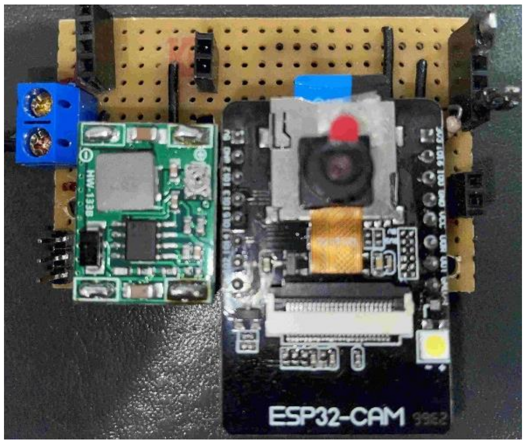


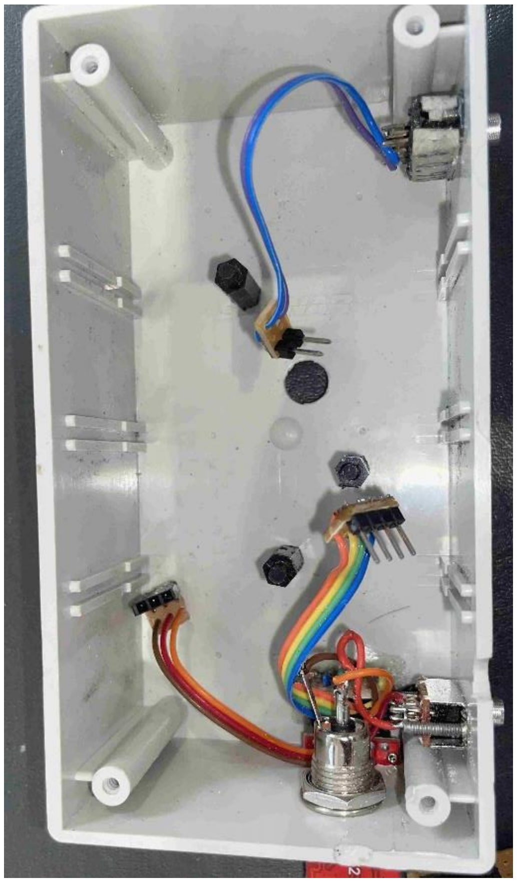
	Bridge Settings			
Triggering	Mode	B1	B2	B3
Soft camera trigger (IOT)	All modes	Closed	Open	Open
Hard camera trigger (PIR)	Operation mode no pullup on sensor IP	Open	Closed	Open
Hard camera trigger (PIR)	Operation mode with pullup on sensor IP	Open	Closed	Closed
Hard camera trigger (PIR)	Upload code pullup on sensor IP	Closed	Open	Don't Care

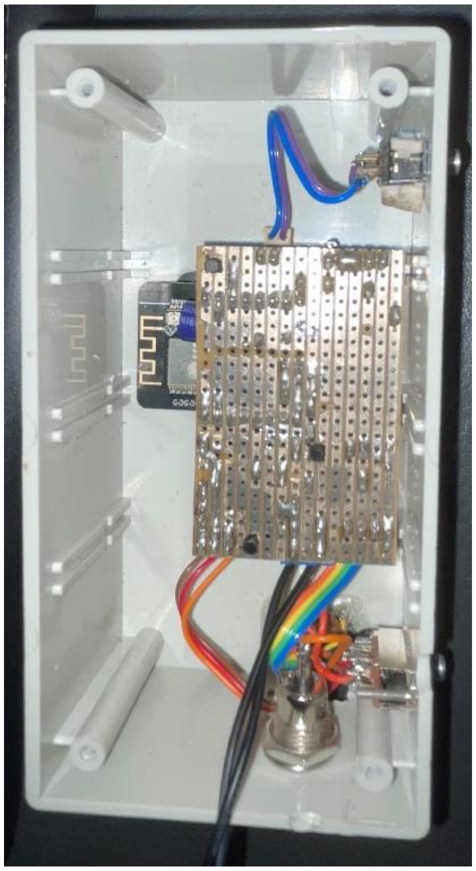
Camera in a box











Software description

<u>Overview:</u>
FreeRTOS is used on the ESP32-CAM AI Thinker, most of the functionality is split between the 2 tasks scheduled by RTOS one for each core.

Core 0
is used to: monitors the Alarm signal from the main unit and raises a flag to core 1 when in alarm, also handles capturing frames and storing them in memory as required.

Core 1
is used to: retrieve frames from memory and create the AVI or JPG file on the SD card.
Also uploads pre-captured images to google drive with the system is idle.

Loop
The main loop handle ArduinoCloud updates and blinking Dash board power indicator

Source code configuration options that set default behavior and timings in CameraProject.ini

Compile time options

```
// comment out the line below to turn off most serial printing
#define DEBUG
```

The project is dependent on two services

ArduinoIotCloud
You will need at least an Arduino base level subscription to implement this project, it currently costs about $20US per year. (that's the only ongoing cost for this project)

Google Drive and Google Scripting.

More detail is available in the Software flow diagram in the
C:\CameraProject\CameraProject_SoftwareFlow.pdf
or
https://github.com/lewlev/CameraProject/blob/main/CameraProject_SoftwareFlow.pdf

Construction and programming

A file Pack2.zip is included with this project includes all software source code files, templates for IOT Thing and Dashboards, CLI executable, higher res circuit and layouts. First thing to do if you are interested in building this is to create a folder in the root of the C drive C:\CameraProject. (you may need to be granted elevated privileges to do this) Extract Pack2.zip into that folder.

Prerequisites and hardware requirements

- An internet connected Wi-Fi service accessible from the location of the new
- system.(probably the location of the existing system)
- A Windows 10 computer (or later) with administrative user access.
- An Android or IOS mobile phone, with Arduino IOT remote app.

Parts List

Item	Description	QTY	Jaycar Stock#	@	Tot
1.	ESP32-CAM ESP32-CAM-MB MICRO USB ESP32 Serial to WiFi ESP32 CAM Development Board CH340 CH340G 5V Bluetooth+OV2640 Camera	1		$7.00	$7.00
2.	MP1584EN 3A Ultra-Small Size DC-DC Step Down Supply Module Adjustable power step-down descending output module	1		$2.00	$2.00
3.	FT232RL FTDI USB 3.3V 5.5V to TTL Serial Adapter Module for Arduino FT232 Pro Mini USB TO TTL 232	1		$2.00	$2.00
4.	Resistor 10K .5W	1		$0.20	$0.20
5.	Resistor 1K .5W	1		$0.20	$0.20
6.	Capacitor Electro 470uF 25v	1	RE6195	$0.75	$0.75
7.	Capacitor Electro 100uF 25v	1	?	$0.75	$0.75
8.	Zenner 6.2V 1W 1N4735	1	ZR1405	$0.95	$0.95
9.	ProtoType BOARD	1	HP9552	$8.95	$8.95
10.	Sub-miniature DPDT Panel Mount Switch	2	SS0812	$1.65	$3.30
11.	Jiffey Box https://www.jaycar.com.au/jiffy-box-grey-130-x-68-x-44mm/p/HB6023	1	HB6023	$3.95	$3.95
12.	Two position screw type terminal block https://www.jaycar.com.au/2-way-pcb-mount-screw-terminals-5mm-pitch/p/HM3172	1		$1.00	$1.00
13.	SD card 4Gig or more formatted to FAT32	1		$10.00	$10.00

Total ~$50.00

Other:
Sundry wire solder heat shrink etc
Tools: soldering iron, digital voltmeter, wire cutters, screw drivers etc

There is not any hardware build instructions with this project, as the finished hardware is just a single board wired point to point on matrix proto type board. I have included a circuit diagram and board layout as well some photos of the finished project.

Software setup

Summary steps

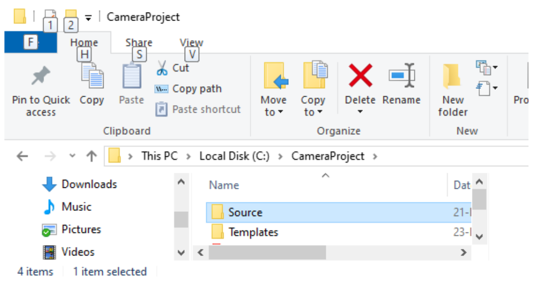
1. Copy the project package to your computer.
2. Setup Arduino Cloud account (if required)
3. Manually add the esp32cam hardware module to your cloud environment
4. Setup Arduino cloud CLI (if required)
5. Use CLI templates to create the CAMERA_THING and dashboard.
6. Associate device CAMERA_DEV to thing CAMERA_THING
7. Link the Camera_Thing to the Alarm_Thing (Part I project) through Arduino IoT
8. Load ESP32 code to Arduino Cloud Environment
9. Configure Mobile Phone apps
10. Configure Google App Script for google drive upload.
11. Configure software user specific items like Wi-Fi etc.
12. Set up SD Card.
13. Fix dashboard widget layout
14. Upload and test

Detail

#### Copy the project package to your computer. ####

1. Create a folder C:\CameraProject that you have full Read write change access to.
   (Elevated permission may be required)
   - Extract **CameraProject.zip** to **[C:\CameraProject](C:\CameraProject)**.
   - You should see Source Template and CLI folders Along with other files and folders

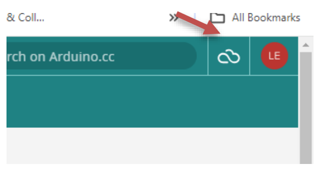
   (it is important to use the exact folder name and Location as the rest of this guide uses links to that Folder) You may want to add **C:\CameraProject** to Quick Access



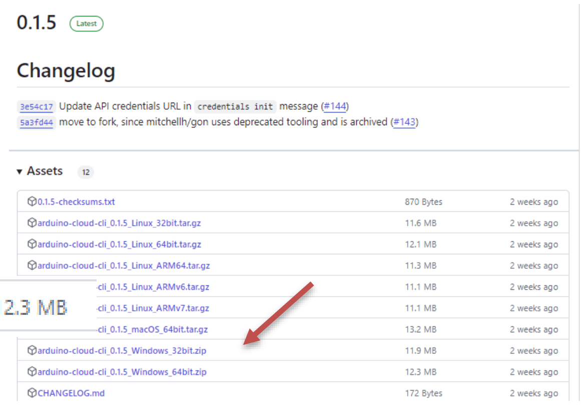
#### Setup Arduino Cloud account (if required) ####

2. Setup Arduino cloud account if you don't have one already.
   Ref https://support.arduino.cc/hc/en-us/articles/360016416280-Sign-up-for-an-Arduino-Cloud-plan
   - Choose at least 'Entry' plan currently $US1.99 per month.

   - Ctrl Click Arduino cloud and Login if you are not already. Click the cloud button (top right) to go to the home page.



   - Skip this if you extracted the Pack1.zip file the CLI is included. Download the ArduinoCloudCLI executable if required. Click on this link arduino-cloud-cli to download the zip file and extract to [C:\AlarmProject\templates](C:\AlarmProject\templates).



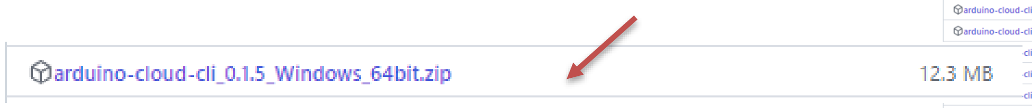
0.1.5 Latest

Changelog

`3e54c17` Update API credentials URL in `credentials init` message (#144)
`5a3fd44` move to fork, since mitchellh/gon uses deprecated tooling and is archived (#143)

▾ Assets 12

0.1.5-checksums.txt	870 Bytes	2 weeks ago
arduino-cloud-cli_0.1.5_Linux_32bit.tar.gz	11.6 MB	2 weeks ago
arduino-cloud-cli_0.1.5_Linux_64bit.tar.gz	12.1 MB	2 weeks ago
arduino-cloud-cli_0.1.5_Linux_ARM64.tar.gz	11.3 MB	2 weeks ago
cli_0.1.5_Linux_ARMv6.tar.gz	11.1 MB	2 weeks ago
cli_0.1.5_Linux_ARMv7.tar.gz	11.1 MB	2 weeks ago
cli_0.1.5_macOS_64bit.tar.gz	13.2 MB	2 weeks ago
arduino-cloud-cli_0.1.5_Windows_32bit.zip	11.9 MB	2 weeks ago
arduino-cloud-cli_0.1.5_Windows_64bit.zip	12.3 MB	2 weeks ago
CHANGELOG.md	172 Bytes	2 weeks ago

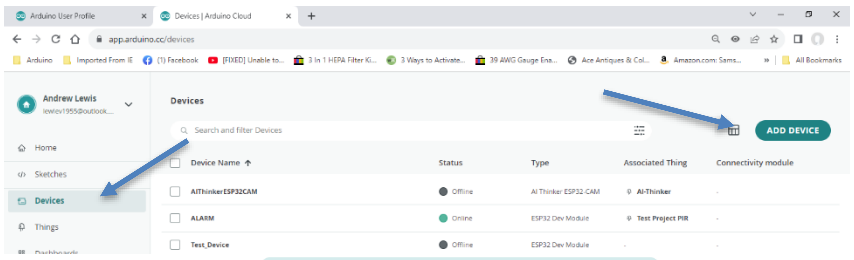


arduino-cloud-cli_0.1.5_Windows_64bit.zip 12.3 MB

####   Manually add the AI Thinker ESP32-Cam to your cloud environment ####

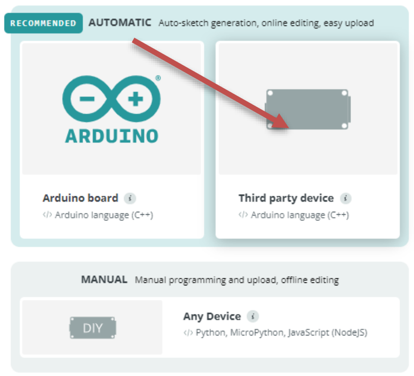
3.   Add the ESP32 as a device into Arduino CLoud

- Open **Arduino Cloud.**
- Select **Devices**
- Select **ADD DEVICE**

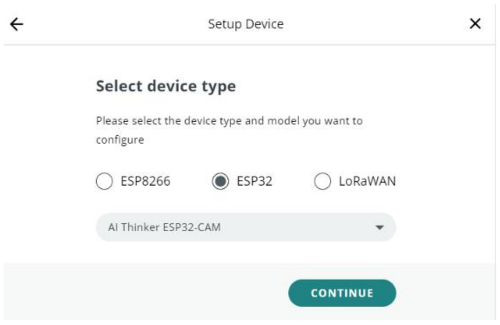


Device Name	Status	Type	Associated Thing	Connectivity module
AIThinkerESP32CAM	Offline	AI Thinker ESP32-CAM	AI-Thinker	-
ALARM	Online	ESP32 Dev Module	Test Project PIR	-
Test_Device	Offline	ESP32 Dev Module	-	-

- Choose **third party device**.


RECOMMENDED
AUTOMATIC Auto-sketch generation, online editing, easy upload
ARDUINO
Arduino board
Arduino language (C++)
Third party device
Arduino language (C++)
MANUAL Manual programming and upload, offline editing
DIY
Any Device
Python, MicroPython, JavaScript (NodeJS)

- Select device type and model
- **ESP32   AI Thinker ESP32-Cam**
- Click on **Continue**.



Select device type

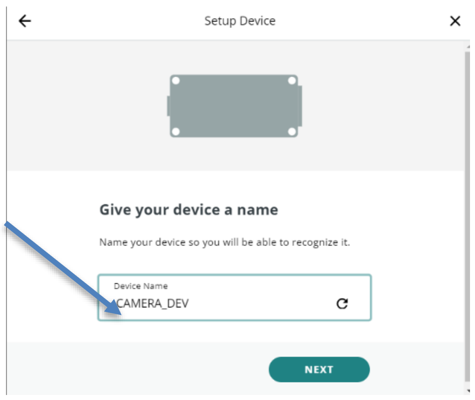
Please select the device type and model you want to configure

ESP8266 [✓ ESP32] LoRaWAN

AI Thinker ESP32-CAM

- Type in **CAMERA_DEV** as the Device Name
- Click Next

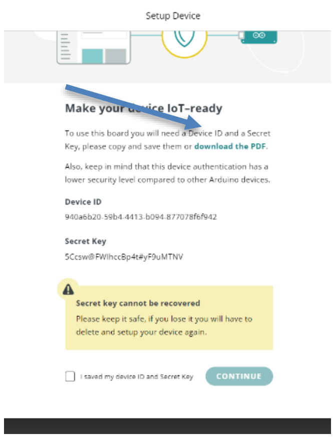
- A dialogue will come up with the 'Device ID' and 'Secret Key'
- Click '**Download the PDF'**
- Click the downloads button in the top right of the browser
- Then hover over the file and click on the folder button



Give your device a name

Name your device so you will be able to recognize it.

Device Name
CAMERA_DEV



Make your device IoT–ready

To use this board you will need a Device ID and a Secret Key, please copy and save them or download the PDF.

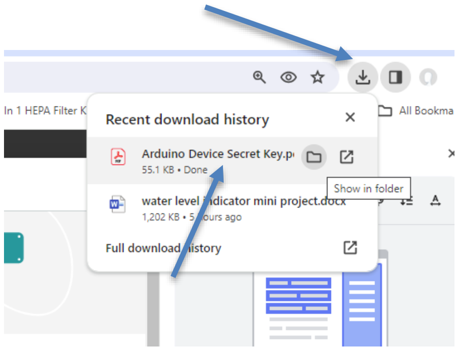
Also, keep in mind that this device authentication has a lower security level compared to other Arduino devices.

Device ID
940a6b20-59b4-4413-b094-877078f6f942

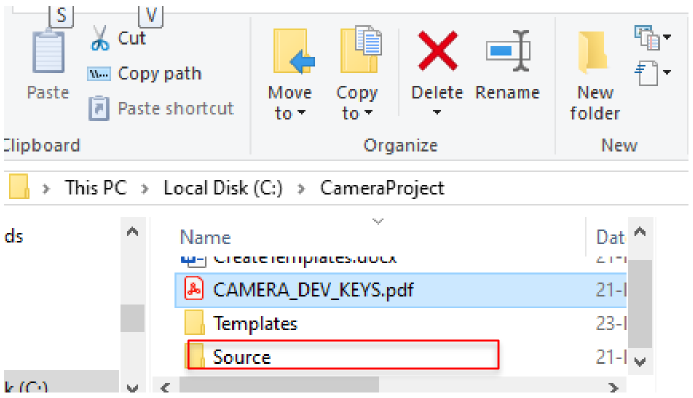
Secret Key
5Ccsw@FWlhccBp4t#yF9uMTNV

Secret key cannot be recovered
Please keep it safe, if you lose it you will have to delete and setup your device again.

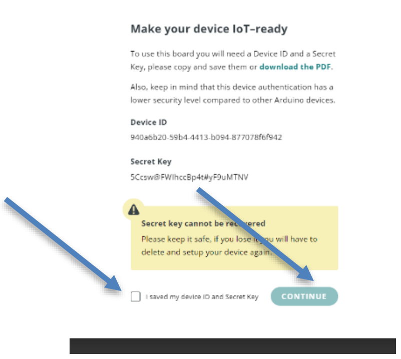
I saved my device ID and Secret Key



- Right click on the pdf file and rename it
- **CAMERA_DEV_KEYS.pdf**
- Copy and paste to the project folder
- Save it in the project folder C:\CameraProject



- Back in the browser
- Click on the '**I saved my Device ID and Secret'** check box then **Continue** and **Done**..



Make your device IoT–ready

To use this board you will need a Device ID and a Secret Key, please copy and save them or download the PDF.

Also, keep in mind that this device authentication has a lower security level compared to other Arduino devices.

Device ID

940a6b20-59b4-4413-b094-877078f6f942

Secret Key

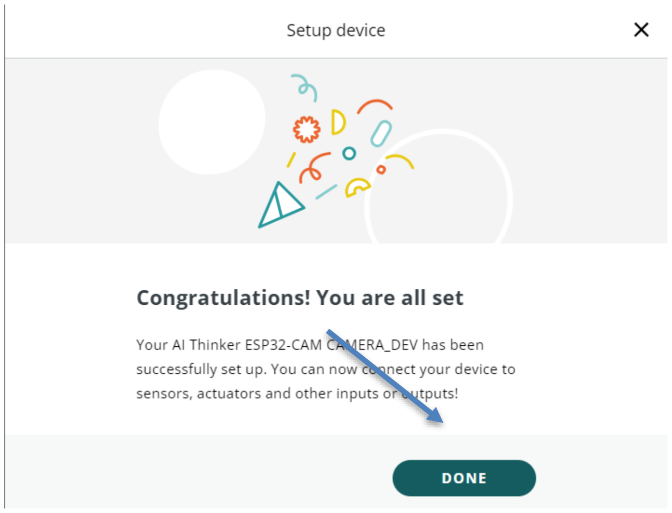
5Ccsw@FWlhccBp4t#yF9uMTNV

Secret key cannot be recovered

Please keep it safe, if you lose it you will have to delete and setup your device again.

I saved my device ID and Secret Key

CONTINUE



Setup device

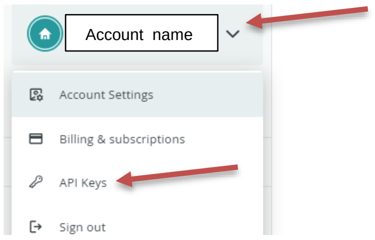
Congratulations! You are all set

Your AI Thinker ESP32-CAM CAMERA_DEV has been successfully set up. You can now connect your device to sensors, actuators and other inputs or outputs!

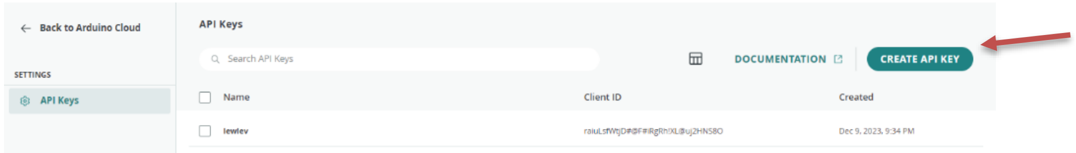
DONE

#### Setup Arduino cloud CLI (if required) ####

- The task involves using the ArduinoCloud CLI if you have not used this before it needs to be authenticated with your Arduino Cloud Account if you have already done that, skip to **Already have an account**.
- Go to your Adruino Cloud home page
- In the top left corner, you will find your account name



- Click on the down arrow beside your name select API Keys



Back to Arduino Cloud

SETTINGS

API Keys

API Keys

Search API Keys

DOCUMENTATION

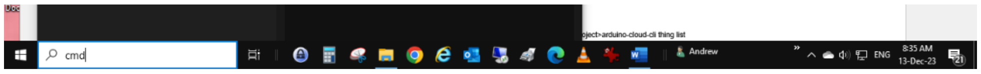
CREATE API KEY

	Name	Client ID	Created
☐	lewlev	raiuLsfWtjD#@F#iRgRh!XL@uj2HNS8O	Dec 9, 2023, 9:34 PM

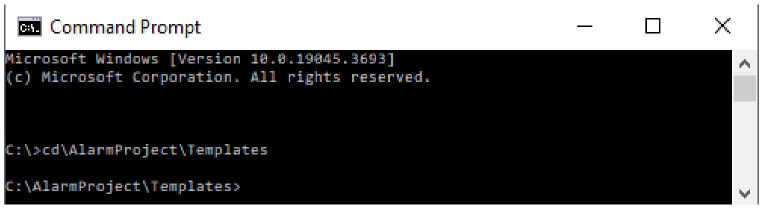
- Click on the Create API Key button
- A new key will be created.
- You will be prompted to save a pdf file with the,new keys details.
- Save it to *C:\CameraProject\Templates say call it CloudCLI_API_Keys.pdf*

4. Already have an account

- Open a command window by typing ‘**cmd**’ into the windows search bar.



- Type or copy and past the text below shown in **”bold text“** at the Command prompt. (don’t include the “”)



```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.


C:\>cd\AlarmProject\Templates

C:\AlarmProject\Templates>
```

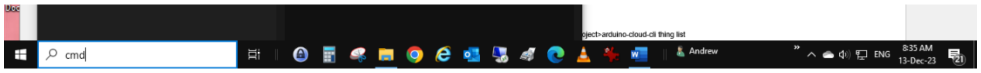
- **“cd\CameraProject\Templates”**

Enter the command below (in bold type) , when prompted enter the ‘ClientID’ and ‘Client Secret’ Obtained earlier.

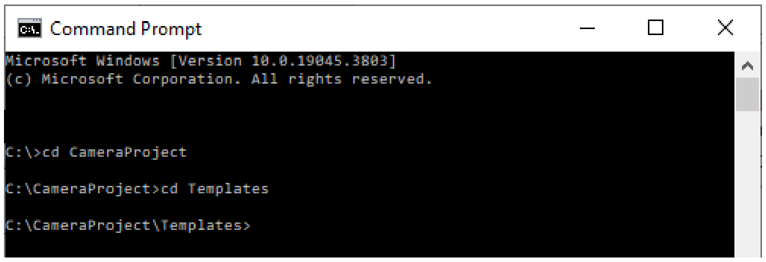
- **“arduino-cloud-cli credentials init”**
  To obtain your API credentials visit https://create.arduino.cc/iot/integrations
  Please enter the Client ID^^&%YUYUIYT*&*&^*&*&^*(^*(&^
  Please enter the Client Secret: **
  v Please enter the Organization ID - if any - Leave empty otherwise:
  Credentials file successfully initialized at:
  D:\Users\User\AppData\Local\Arduino15\arduino-cloud-credentials.yaml
- If successful you should see a response like the one above.

- Open ArduinoCloud CLI on your computer
  (If you just added a new set of keys skip to step 10)
  (computer must be running Windows 10 at a minimum)
  the CLI executable will be in the C:\CameraProject \Templates folder.
  Ref Cli guide docs https://docs.arduino.cc/arduino-cloud/arduino-cloud-cli/getting-started

- If you don’t have one open, open a command window by typing ‘cmd’ into the windows search bar



- Type or copy and paste the text below shown in **”bold text“** to the Command prompt. (don’t include the “”) followed by the Enter key



```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.


C:\>cd CameraProject

C:\CameraProject>cd Templates

C:\CameraProject\Templates>
```

“cd\CameraProject\Templates”

C:\CameraProject\Templates>

#### Use CLI templates to create the CAMERA_THING and the dasboard ####

5. Create the **CAMERA_THING** thing from template.

- Type or copy and past the text below shown in **"bold text"** to the Command prompt. (don't include the "") followed by the Enter key

- **"arduino-cloud-cli thing create --name CAMERA_THING --template CAMERA_THING_Temp.yaml name: CAMERA_THING"**

  name: CAMERA_THING
  id: c5713bfc-00ad-4a9b-b75a-6564d207606b
  device_id:
  variables: DBImageMode, DBPicturesPerMinute, DBVideoLength, DBVideosPerMinute, DBBlinkPower, DBSaveChanges, DBSaved, DBSysidle, RemoteAI
- If you get a response like the one above the new CAMERA_THING was created OK. The ID (Yellow highlight) will of course be different and unique to the new thing.

  Take note of the thing id above it is needed to create the dashboards via the CLI in next steps. (tip, to copy txt from the command prompt, right click the top white command prompt bar, select edit select mark, use the mouse to select txt, it will highlight in white, hit the return key. The text can pasted anywhere with Ctrl and P keys)

- Create the **Camera_Control** dashboard from template.

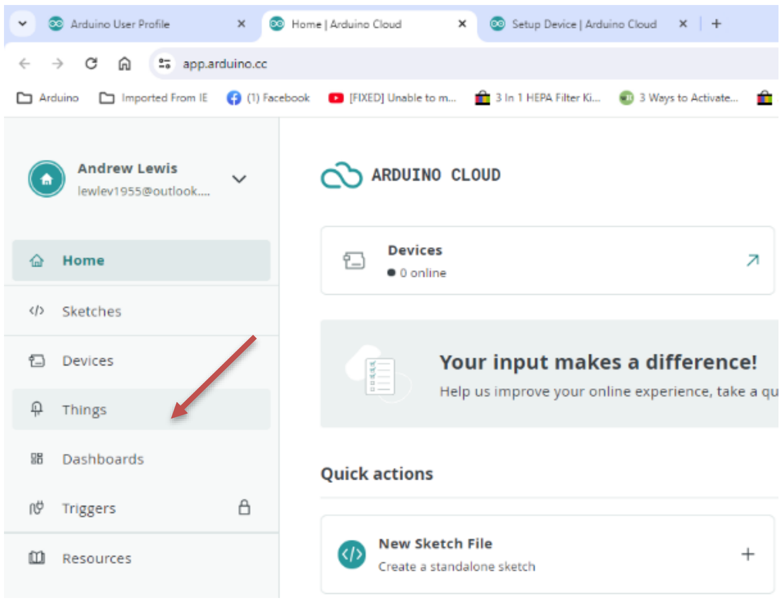
- Please copy just the ID in yellow and past it into the command window. Don't hit return or any other key except the left arrow to move the cursor back to the beginning of command prompt, like this
  C:\CameraProject\Templates >| *c5713bfc-00ad-4a9b-b75a-6564d207606b*

- Now type or copy and paste the section of the command below in bold (don't copy the quotes) followed by the Enter key

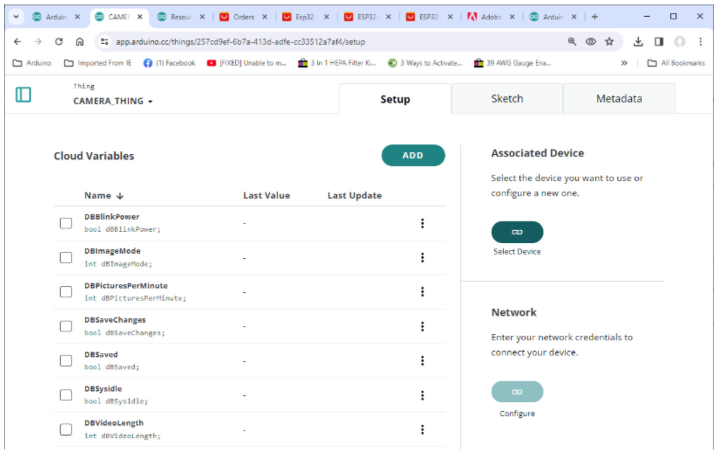
- **"arduino-cloud-cli dashboard create --name Camera_Control --template Camera_Control_Temp.yaml --override CAMERA_THING="** *c5713bfc-00ad-4a9b-b75a-6564d207606b*

  name: Camera_Control
  id: d5355171-c4fe-4818-841f-42df411738cc
  updated_at: 2023-12-13 00:25:20.7042 +0000 UTC
  widgets: Saved, Power, Video Length in Seconds, Videos per minute, Save Changes, 0=Picture << Image mode >>Video=1, Ready/Busy, Pictures per Minute

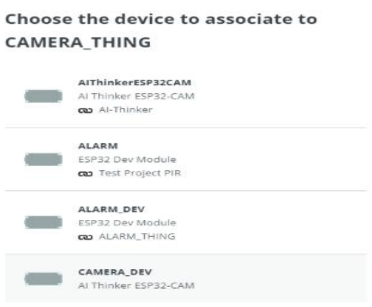
#### Associate CAMERA_DEV and CAMERA_THING. ####



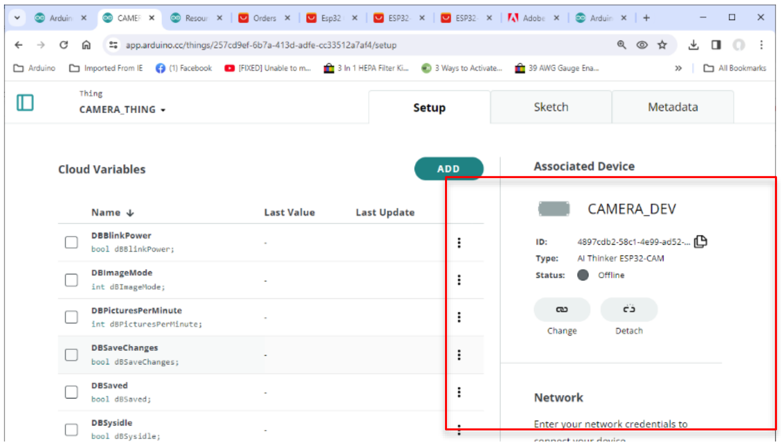
6. From the Arduino Cloud home page choose **Things**



- Then choose **CAMERA_THING** Then click on '**Select Device'**



- Then choose **CAMERA_DEVICE**

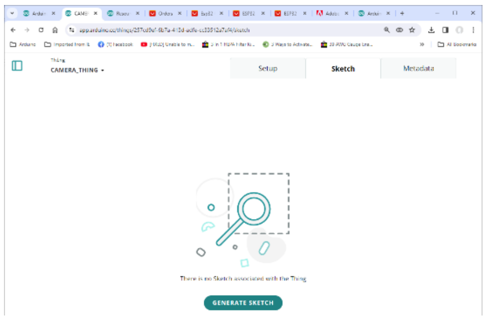


- Done

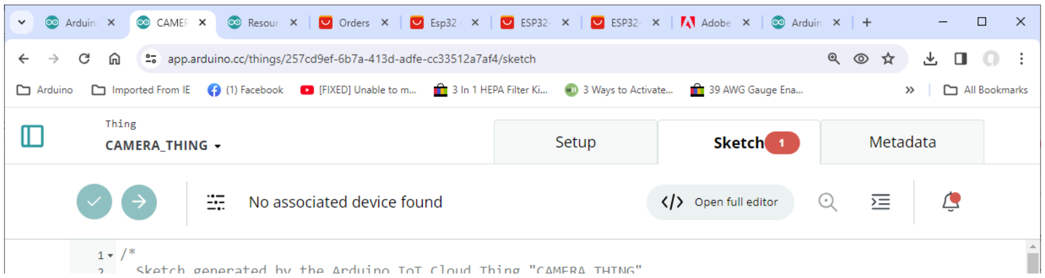
#### Copy ESP32 code to Arduino cloud environment ####

7. Load the source files into your cloud environment, there is a little bit of file manipulation to do, because of the project and file name conventions used.

- Open [Arduino Cloud]() home page login if required,
- Open **Things**
- Open thing **CAMERA_THING** then click on the **sketch** tab
- You should see the dialogue below click on "**Generate Sketch**"

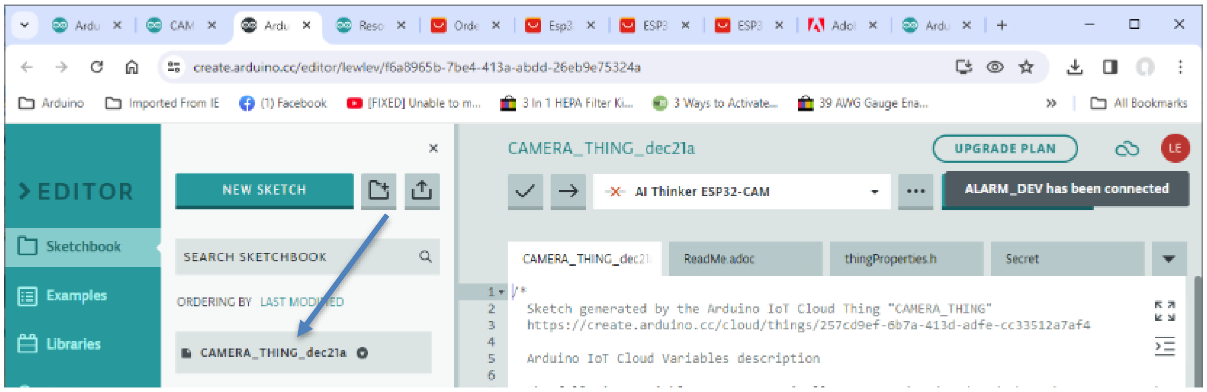


- The sketch will open in the basic editor


app.arduino.cc/things/257cd9ef-6b7a-413d-adfe-cc33512a7af4/sketch
Thing
CAMERA_THING
Setup
Sketch 1
Metadata
No associated device found
Open full editor

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "CAMERA_THING"
```

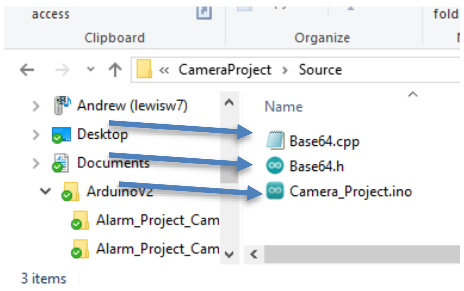
- Click on Open full editor.
  The web editor will open with 4 tabs
  CAMERA_THINGxxxx.ino, ReadMe,thingProperties.h and Secret
  We will effectively replace CAMERA_THINGxxxx.ino code , with the CameraProject code.



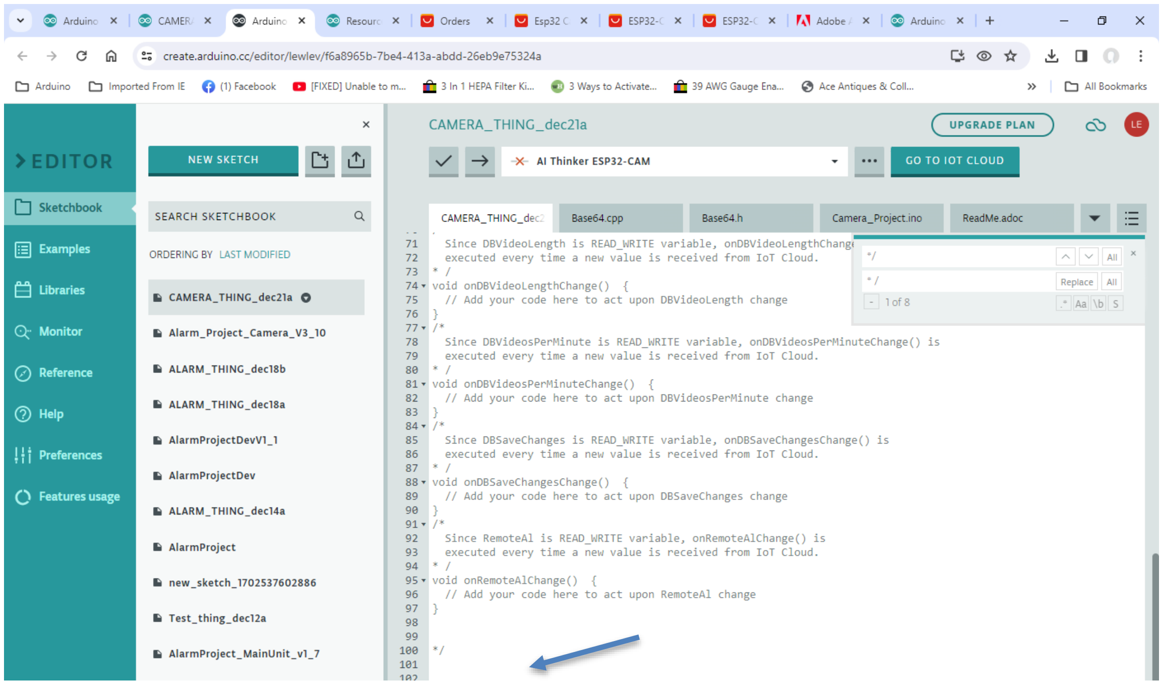
```
/*
  Sketch generated by the Arduino IoT Cloud Thing "CAMERA_THING"
  https://create.arduino.cc/cloud/things/257cd9ef-6b7a-413d-adfe-cc33512a7af4

  Arduino IoT Cloud Variables description
```

- The Web editor will have **CAMERA_THINGxxxx.ino** at the top of the sketch list in the left pane.
- Click on the down arrow and choose **'Import File into Sketch'**
- Browse to **C:\CameraProject\Source** and **Open**
- Select **CameraProject.ino,Base64.cpp and Baes64.h** then **Open**



- Open **CAMERA_THINGxxxx.ino** in the web editor
- If you want to keep auto generated comments etc follow the next steps. If not just **Select All** and **delete.**
    - right click andchoose '**Find and Replace'**
    - Type ***/** into the find box and *** /** into the replace box click on the **ALL** button
    - Move the cursor past the last of the text in the file, then type in ***/** to close the last comment. This will effectively comment out the auto generated section of code that will be replaced by the incoming .ino file



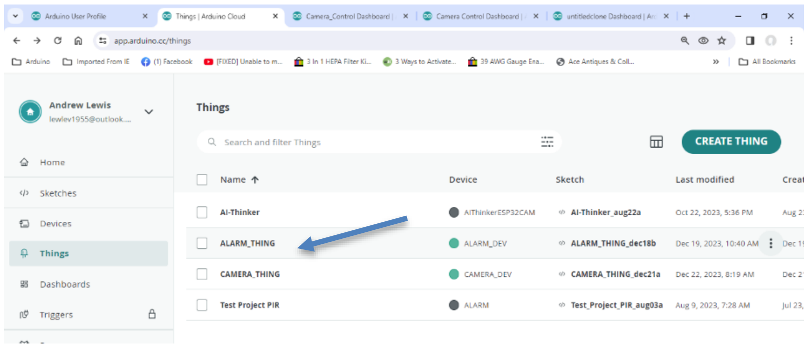
```
  Since DBVideoLength is READ_WRITE variable, onDBVideoLengthChange
  executed every time a new value is received from IoT Cloud.
* /
void onDBVideoLengthChange()  {
  // Add your code here to act upon DBVideoLength change
}
/*
  Since DBVideosPerMinute is READ_WRITE variable, onDBVideosPerMinuteChange() is
  executed every time a new value is received from IoT Cloud.
* /
void onDBVideosPerMinuteChange()  {
  // Add your code here to act upon DBVideosPerMinute change
}
/*
  Since DBSaveChanges is READ_WRITE variable, onDBSaveChangesChange() is
  executed every time a new value is received from IoT Cloud.
* /
void onDBSaveChangesChange()  {
  // Add your code here to act upon DBSaveChanges change
}
/*
  Since RemoteAl is READ_WRITE variable, onRemoteAlChange() is
  executed every time a new value is received from IoT Cloud.
* /
void onRemoteAlChange()  {
  // Add your code here to act upon RemoteAl change
}


*/
```

- Open the CameraProject.ino tab in the web editor, right click and choose '**Select All**' right click again and choose **'Copy'**
- Open the **CAMERA_THINGxxxx.ino** tab again and place the cursor below any existing text. Right click and choose **'Paste'**
- Now you should have the code from CameraProject.ino in CAMERA_THINGxxxx.ino.
- Select the **CameraProject.ino** tab then click on the little down arrow tab and select **delete** CameraProject.ino.

#### Link the Camera_Thing to Alarm_Thing via an IoT shared variable. ####

8. Open **Alarm_thing**



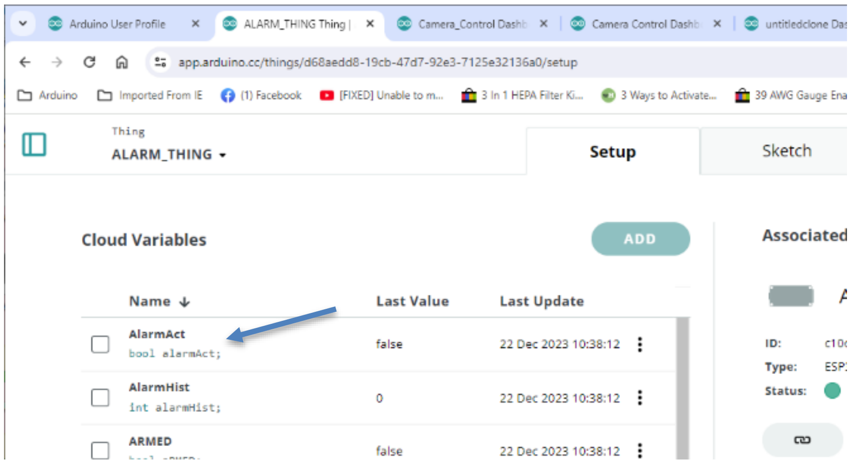
app.arduino.cc/things

Andrew Lewis
lewlev1955@outlook....

Things

	Name	Device	Sketch	Last modified	Creat
	AI-Thinker	AIThinkerESP32CAM	AI-Thinker_aug22a	Oct 22, 2023, 5:36 PM	Aug 2
	ALARM_THING	ALARM_DEV	ALARM_THING_dec18b	Dec 19, 2023, 10:40 AM	Dec 1
	CAMERA_THING	CAMERA_DEV	CAMERA_THING_dec21a	Dec 22, 2023, 8:19 AM	Dec 2
	Test Project PIR	ALARM	Test_Project_PIR_aug03a	Aug 9, 2023, 7:28 AM	Jul 23,

- Select variable **AlarmAct**



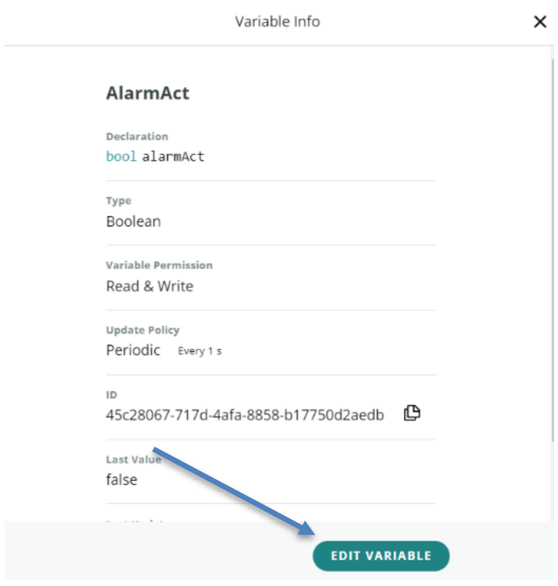
app.arduino.cc/things/d68aedd8-19cb-47d7-92e3-7125e32136a0/setup

Thing
ALARM_THING

Cloud Variables

Name	Last Value	Last Update
AlarmAct `bool alarmAct;`	false	22 Dec 2023 10:38:12
AlarmHist `int alarmHist;`	0	22 Dec 2023 10:38:12
ARMED	false	22 Dec 2023 10:38:12

- Select **Edit variable** then '**Sync with Other Thing'**



Variable Info

AlarmAct

Declaration
`bool alarmAct`

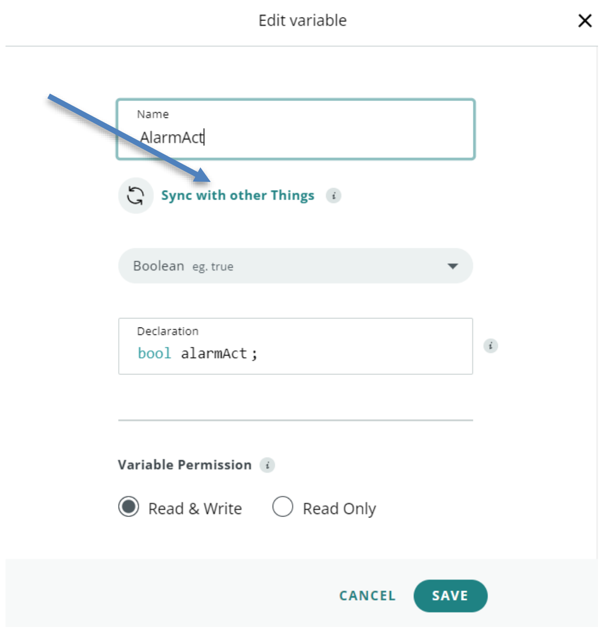
Type
Boolean

Variable Permission
Read & Write

Update Policy
Periodic Every 1 s

ID
45c28067-717d-4afa-8858-b17750d2aedb

Last Value
false



Edit variable

Name
AlarmAct

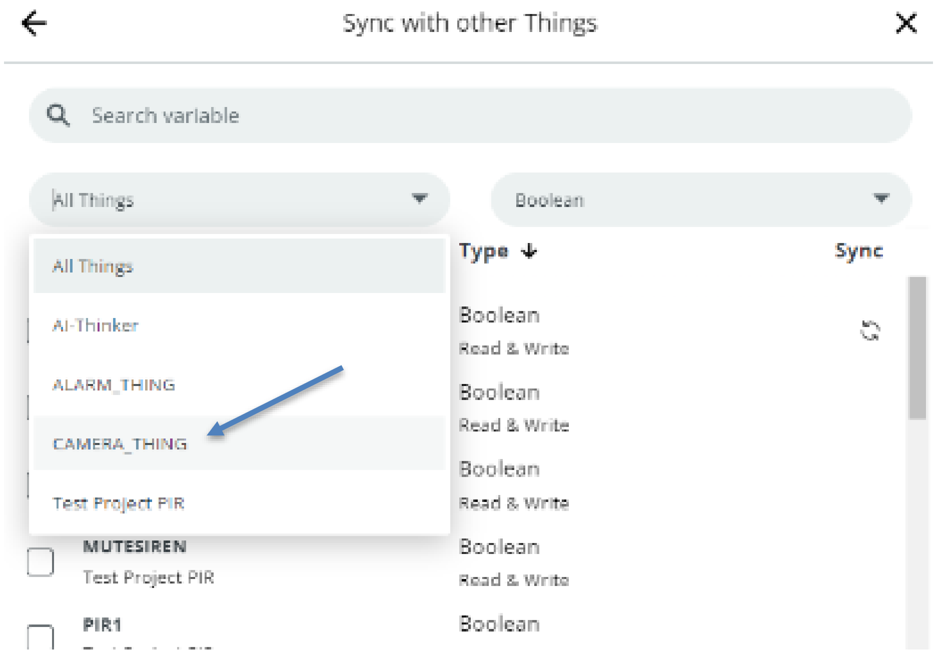
Sync with other Things

Boolean eg. true

Declaration
`bool alarmAct ;`

Variable Permission
[✓ Read & Write] Read Only

- Select **Camera_Thing.**



Sync with other Things

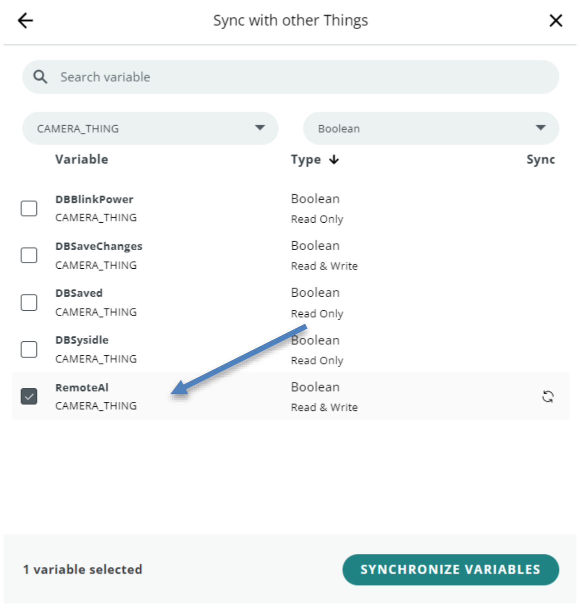
Search variable

All Things | Boolean

All Things
AI-Thinker
ALARM_THING
CAMERA_THING
Test Project PIR

Variable	Type	Sync
	Boolean Read & Write	
	Boolean Read & Write	
	Boolean Read & Write	
MUTESIREN Test Project PIR	Boolean Read & Write	
PIR1	Boolean	

- Select **RemoteAI** then **Save.**



Sync with other Things

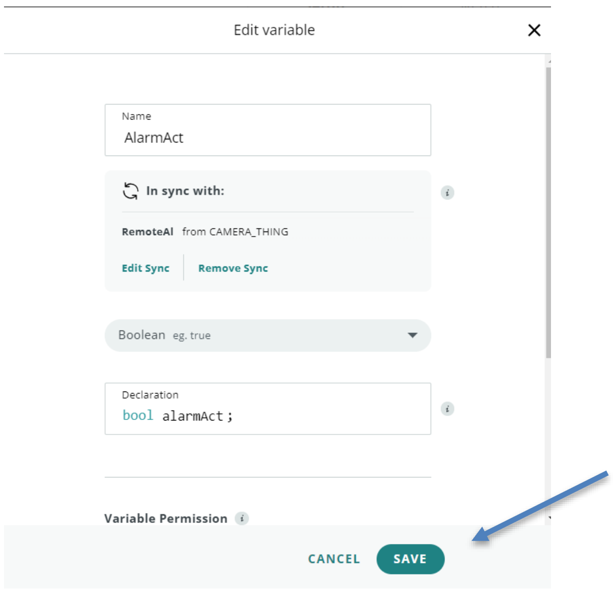
Search variable

CAMERA_THING | Boolean

Variable	Type	Sync
DBBlinkPower CAMERA_THING	Boolean Read Only	
DBSaveChanges CAMERA_THING	Boolean Read & Write	
DBSaved CAMERA_THING	Boolean Read Only	
DBSysidle CAMERA_THING	Boolean Read Only	
✓ RemoteAI CAMERA_THING	Boolean Read & Write	

1 variable selected

SYNCHRONIZE VARIABLES



Edit variable

Name: AlarmAct

In sync with:
RemoteAI from CAMERA_THING

Edit Sync | Remove Sync

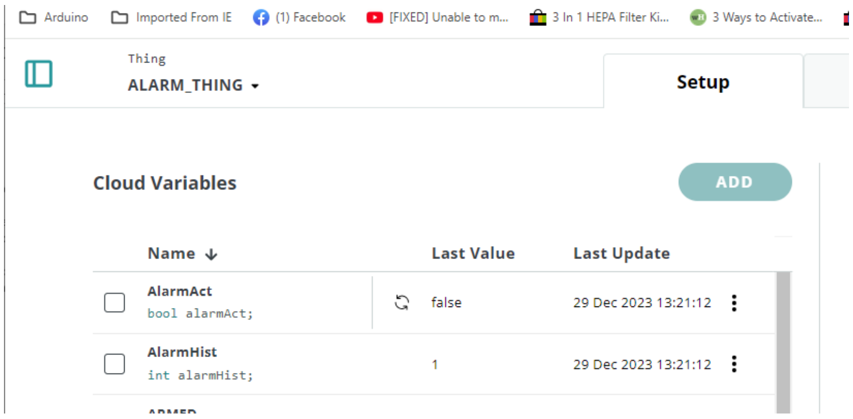
Boolean eg. true

Declaration: `bool alarmAct;`

Variable Permission

CANCEL SAVE

- Done



Arduino | Imported From IE | (1) Facebook | [FIXED] Unable to m... | 3 In 1 HEPA Filter Ki... | 3 Ways to Activate...

Thing
ALARM_THING

Setup

Cloud Variables

ADD

Name	Last Value	Last Update
AlarmAct `bool alarmAct;`	false	29 Dec 2023 13:21:12
AlarmHist `int alarmHist;`	1	29 Dec 2023 13:21:12
ARMED		

#### Configure Mobile Phone apps. ####

9. Goto the app store

- Install the Arduino IOT remote app use the Arduino Cloud Account credentials. A brief guide is included in the pack C:\CameraProject\Alarm_Project_Mobile App installs.pdf "Mobile App Installs.pdf" if required.

#### Configure Google App Script for google drive upload. ####

10. If you don't already have a Google account goto https://www.google.com/script/start/ set one up. A guide can be found at: C:\CameraProject\Camera _project _Google_Account.pdf
    - Create the Google Apps Script, the required script and a guide can be found at: C:\CameraProject\Camera _project _Google_Script.pdf.

#### Configure software user specific items Wi-Fi and Google Script key. ####

11. Click on the cloud button (top right) click on Sketches.

Secrets.h

- Open the Secrets tab and enter your Wi-Fi SSID and password and the **Secret Key** from the **CAMERA_DEV**, it will be in the pdf file **CAMERA_DEV_KEYS.pdf** file saved earlier to the project folder.
- Note: this will automatically populate WiFi credentials and Secret ID in CAMERA_THING as well.

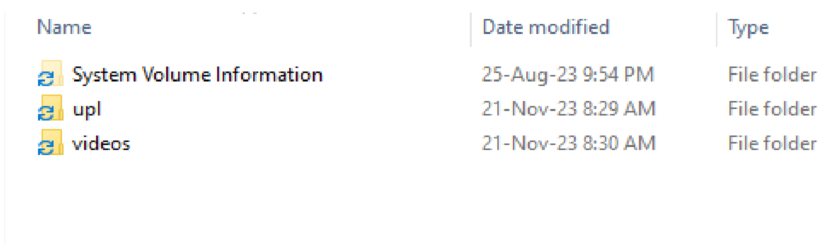
CameraProject.ino

Find the comment

```
// ##################  Enter Google Script Deployment ID here  ################
```

- Paste ID (saved earlier) in the next line after `"/macros/s/ and before /exec";`
  `String myScript = "/macros/s/your key~73characters/exec";`

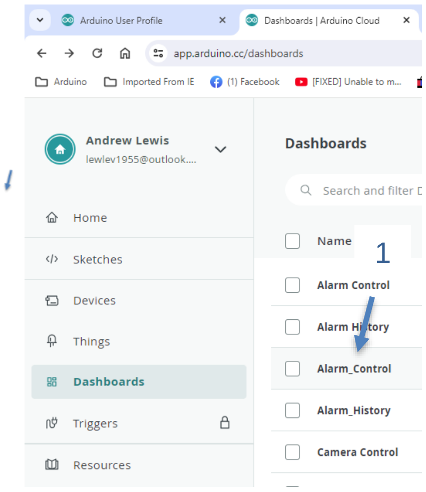
12. Set up SD card
    - Format a 4Gig or greater Sdcard to FAT32
    - Add two folders in the root of the SD drive videos and upl
      Videos folder holds captured images until they are uploaded to google drive after that they are moved to upl folder
    - Insert Sdcard into ESP32-CAM module

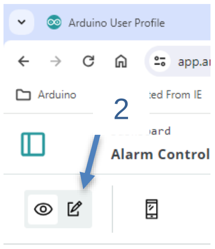


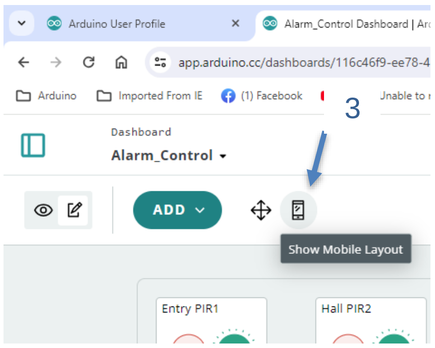
Name	Date modified	Type
System Volume Information	25-Aug-23 9:54 PM	File folder
upl	21-Nov-23 8:29 AM	File folder
videos	21-Nov-23 8:30 AM	File folder

13. Fix dashboard widget layout

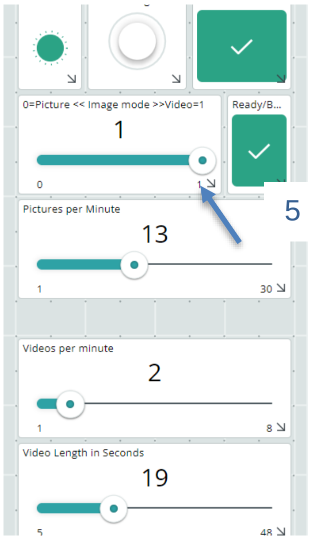
Note for reasons unknown to me, the Dashboards created through the CLI, don't automatically layout the widgets correctly in the Mobile view. So it is necessary to adjust Mobile layout. Open ArduinoCloud follow the steps below to edit the layout

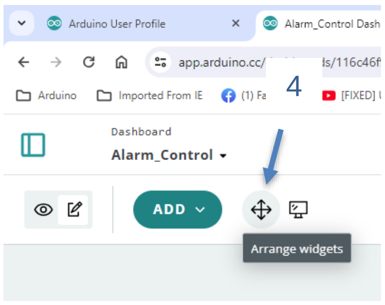


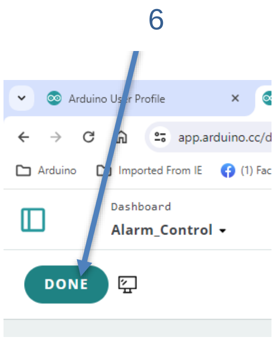




Move Widgets as desired

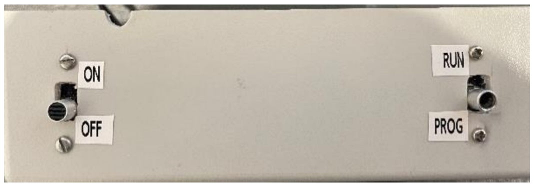


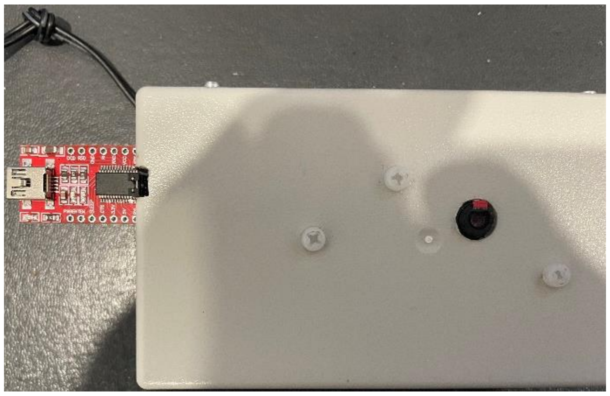




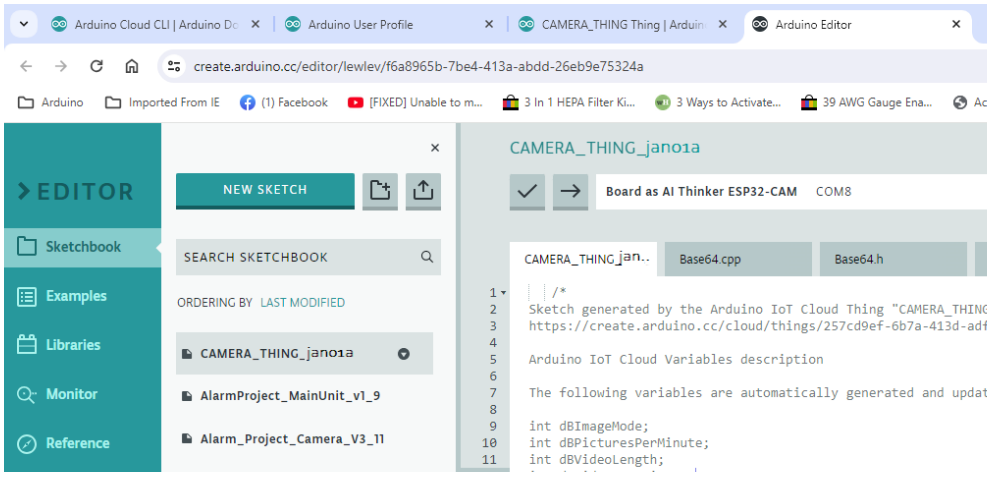
Upload code and Test

- Turn the camera **Off** and to PROG. Connect the FTDI232 module to the camera, and connect the DC power supply.





- Connect the FTDI232 to the PC via USB conditions.
- Open the Arduino Cloud Web Editor then the **CAMERA_THINGxxxx.ino** file
- Uncomment the **Debug** definition in the ino file so we get some definition serial monitoring
- Check /set the device connection to Board COMx **AI Thinker ESP32-Cam**



Board as AI Thinker ESP32-CAM COM8

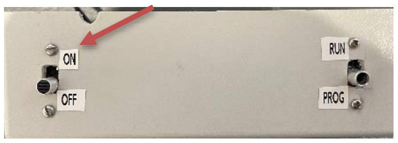
```
/*
Sketch generated by the Arduino IoT Cloud Thing "CAMERA_THING
https://create.arduino.cc/cloud/things/257cd9ef-6b7a-413d-adf

Arduino IoT Cloud Variables description

The following variables are automatically generated and updat

int dBImageMode;
int dBPicturesPerMinute;
int dBVideoLength;
```

- Open the serial monitor (@115200 baud.



- Put the Camera into upload mode by turning it on, you should see 'waiting for download' in the monitor.

- Compile and upload. If successful. Switch RUN/PROG switch to, then power to off and then back on.
- The ESP will go through setup and connect to Wi-Fi etc as shown below.
- If connected the FLASH LED will briefly flash.

```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1344
load:0x40078000,len:13964
load:0x40080400,len:3600
entry 0x400805f0

Connecting to your_ssid
initProperties();
Initialising camera
config.grab_mode: 1
Camera ready
Getting time.
Monday, January 01 2024 19:00:23
SD card ready
cam_hal: EV-EOF-OVF
  setDebugMessageLevel(4);
  ArduinoCloud.printDebugInfo();
***** Arduino IoT Cloud - configuration info *****
Device ID: Your deviceID
MQTT Broker: mqtts-up.iot.arduino.cc:8884
LOloop: 0
WiFi.status(): 3
++++++++++Connected to "lewlevguest"
```

```
+TimeServiceClass::sync  Drift: 0 RTC value: 1704099631
++++++++Connected to Arduino IoT Cloud
Thing ID: Your thing ID
+++TimeServiceClass::setTimeZoneData offset: 39600 dst_unitl 1712419200
Arduino synced to IoT Cloud
+DoYlast 364
dBImageMode: 0
ImageMode: 0
Alarm_DELAY Alarmdetected holdtime: 3000
Alarm_INTERVAL beween alarm checks: 500
GDLockoutInterval alarm to upload lockout: 300000
------avi-------------------
dBVideoLength dasboard: 5
MaxAviLen: running system5000
EPVideosPerMinute videos(avi's) per/min in eeprom:  2
dBVideosPerMinute: 2
AviInterval  lock out between writing avi files determind by dBVideosPerMinute: 30000
------jpg-------------------
dBPicturesPerMinute: 13
EPPicturesPerMinute: 13
JpgInterval      jpgtojpg: 4000
------end-------------------
Waiting for Arduino IOT sync
System ready!
```

- If not there is some trouble shooting to do
- If you are wanting to link the camera thing to the Alarm Controller (the Part I project)
  Once the Camera is fully up and running, you should be able to raise an alarm on the alarm controller and that in turn should trigger the camera to start capturing images or videos. If not go to things in Arduino cloud open ALARM_THING and check that variable **AlarmAct** is linked to CAMERA_THING variable **RemoteAl.**

14. Tested OTA not working at present may need firmware update on ESP32-Cam module.

15. Note if you want to do some coding on local Arduino IDE you will need to install eps-idf installed
    See https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/