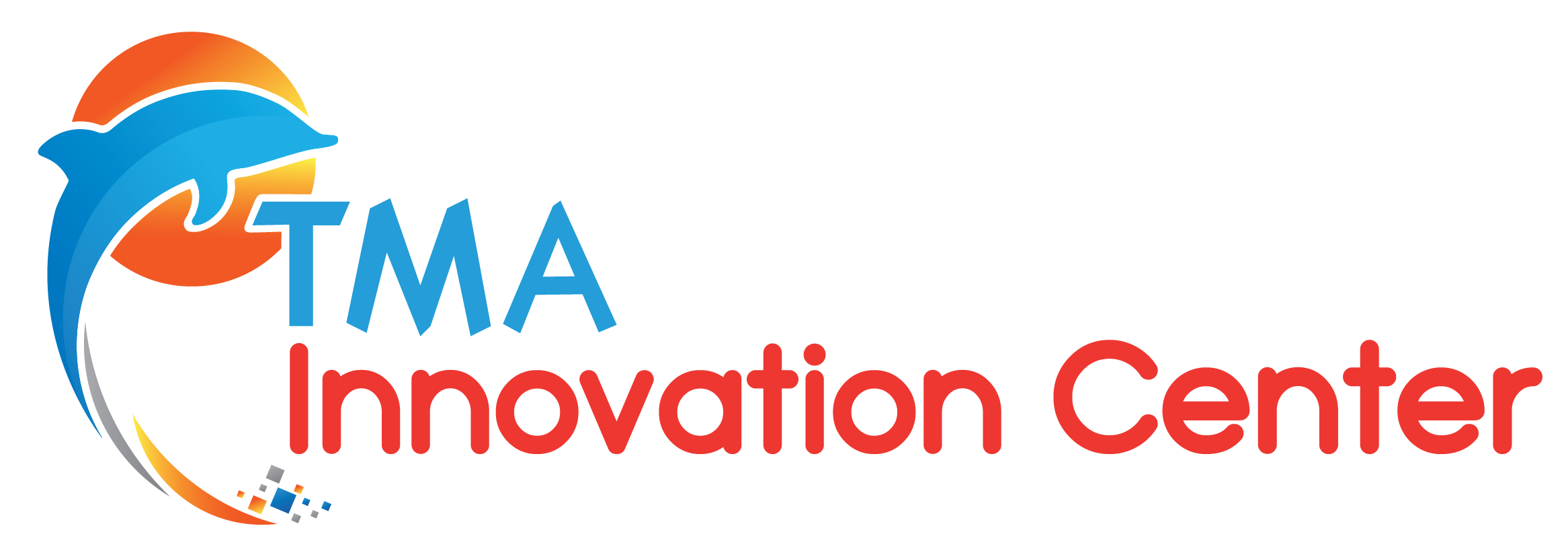
****

AWS IOT QUICK START

Contents

[**1.** **SETUP** 3](#_Toc504121562)

[**1.** **Software setup** 3](#_Toc504121563)

[**2.** **Hardware setup** 4](#_Toc504121564)

[**2.** **AWS with ESP32** 4](#_Toc504121565)

[**1.** **Register a device** 4](#_Toc504121566)

[**2.** **Run the device** 9](#_Toc504121567)

[**3.** **AWS with Rasberry Pi** 12](#_Toc504121568)

[**1.** **NodeJS** 12](#_Toc504121569)

[**2.** **Embedded C** 14](#_Toc504121570)

[**4.** **REFERENCE** 16](#_Toc504121571)

# **SETUP**

## **Software setup**

**ACCOUNT SETUP**

* Ask your manager for an account because it is not free
* In case you already have an account, login
* Login AWS IoT with your account

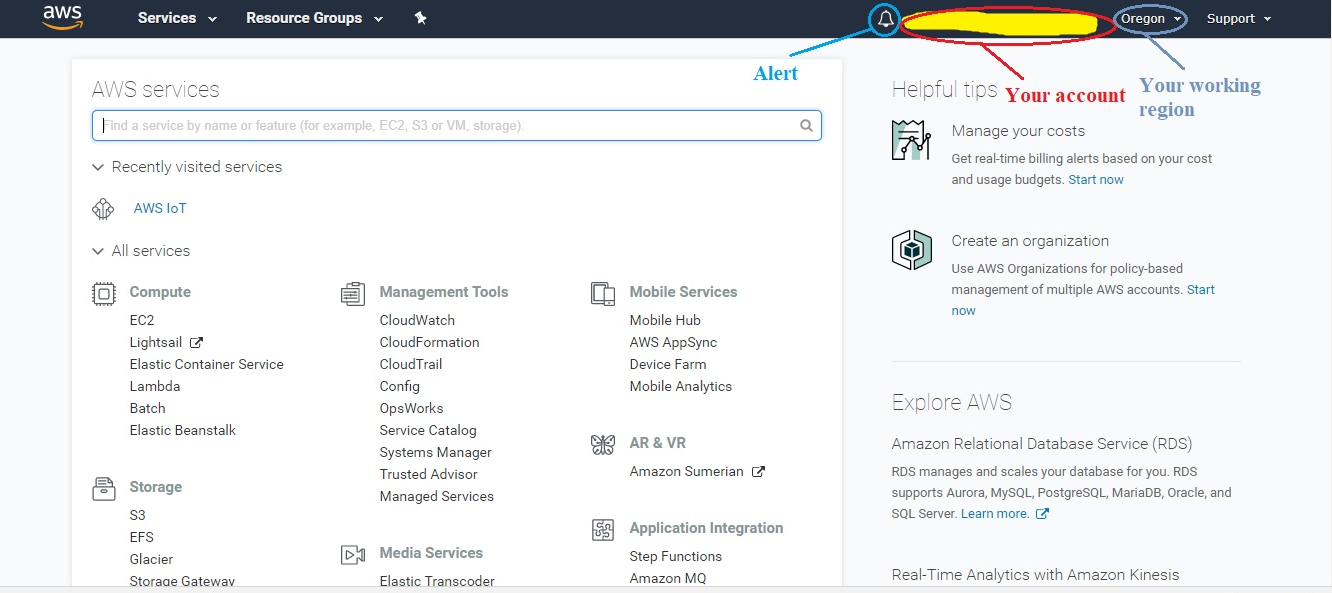


Figure 1 When you login successfully

**IDE & LIBRARY SETUP**

Here are some software and library setup before you can run the basic configuration and examples. It just only a few minutes to download all of them

**ESP32 software setup**

* Install Arduino core for ESP32 from <https://github.com/espressif/arduino-esp32>
* Download and install the AWS\_IOT library for ESP32 <https://github.com/ExploreEmbedded/Hornbill-Examples/tree/master/arduino-esp32/AWS_IOT>

**Raspberry Pi software setup**

* Clone the AWS IoT Embedded C SDK

$ git clone <https://github.com/aws/aws-iot-device-sdk-embedded-C>

* Clone some external libraries, then copy their source code inside the[*external\_libs*](https://github.com/aws/aws-iot-device-sdk-embedded-C/tree/master/external_libs)directory of the Embedded C SDK

$ git clone <https://github.com/cpputest/cpputest>

$ git clone <https://github.com/ARMmbed/mbedtls>

* Clone the prepared NodeJS demo (This one is like a separated SDK)

$ git clone <https://github.com/nbxtruong/AWS-IoT-Demo>

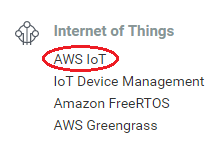
## **Hardware setup**

* An ESP32
* A Raspberry Pi 3 board
* DHT22 (optional – in case you want to understand deeper)
* Some LEDs (optional – in case you want to understand deeper)
* Jumper Wires (optional – in case you want to understand deeper)
* Some Buttons or Switches (optional – in case you want to understand deeper)
* Resistor 330 Ohm and 1k (optional – in case you want to understand deeper)
* A breadboard (optional – in case you want to understand deeper)

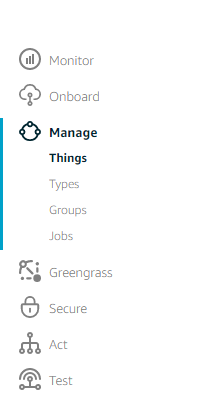
# **AWS with ESP32**

## **Register a device**

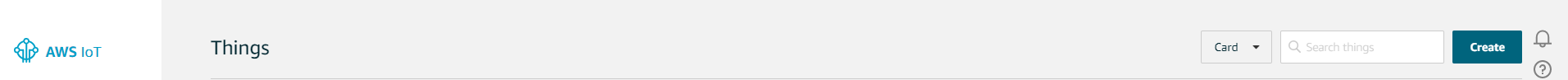
* From your console, find **Internet of Things** and choose **AWS IoT**



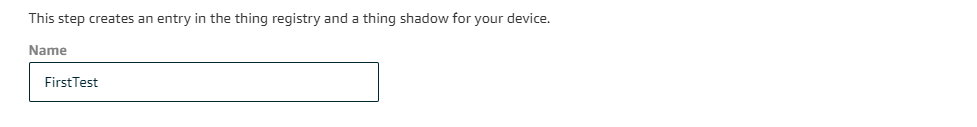
* Choose **Manage -> Things**



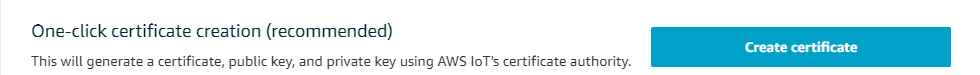
* Click on **Create** to create a new **Thing**



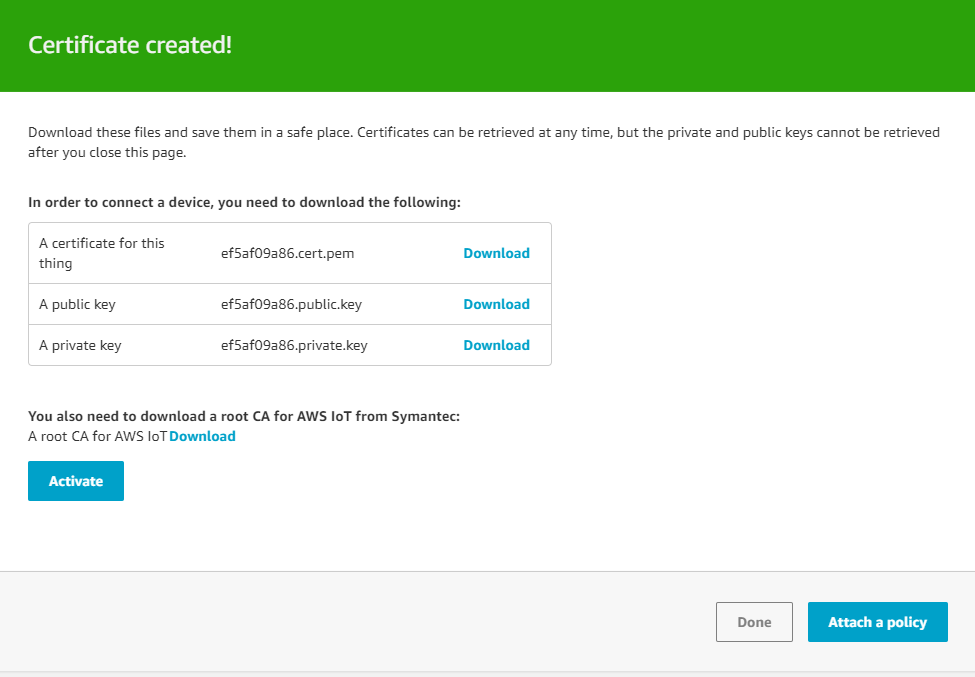
* Select **Create a single thing**
* Name your **Thing** then select **Next** (I named my **Thing** *FirstTest*)



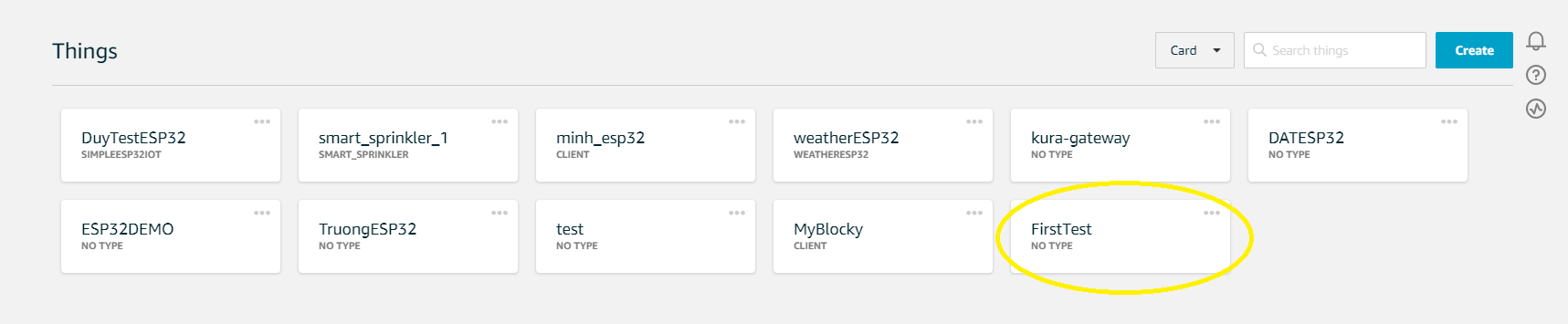
* Create a certificate by clicking **Create certificate**



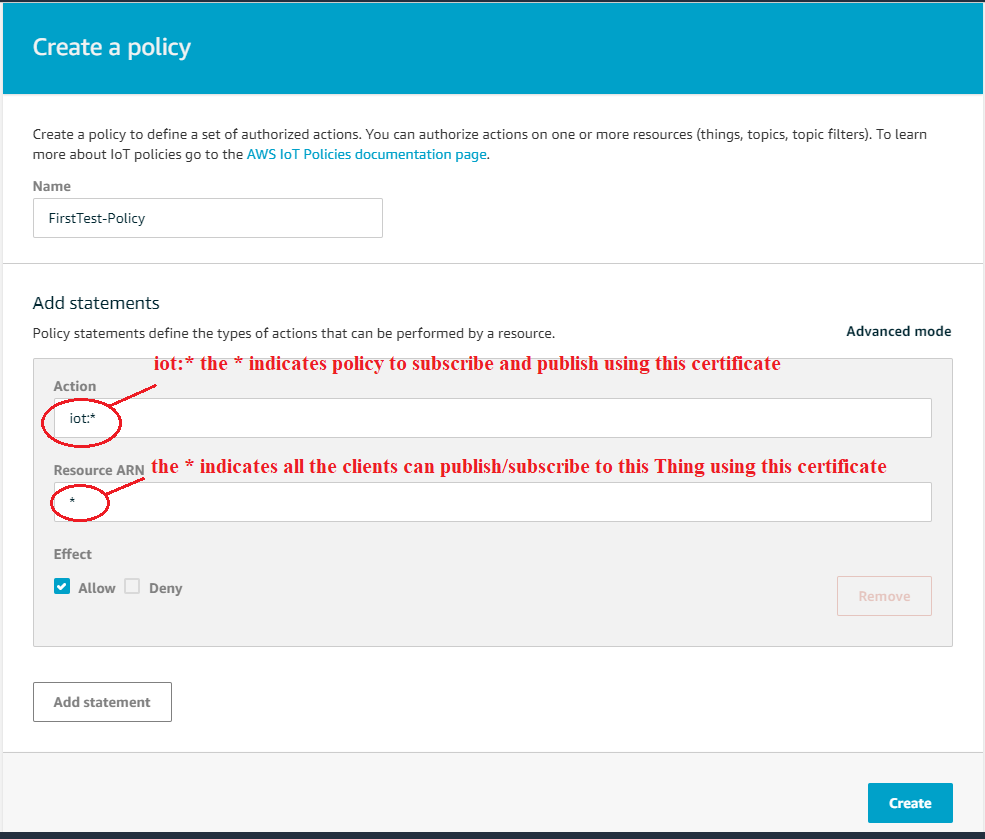
* Download all the certificates (the root CA should be saved in Notepad under the name **aws-root-ca.pem**) then activate all. Remember not to give these keys to anyone.



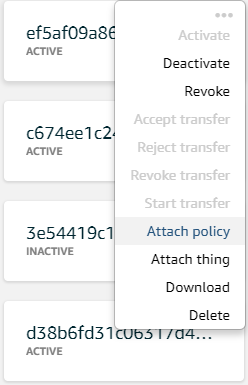
* Your **Thing** is created

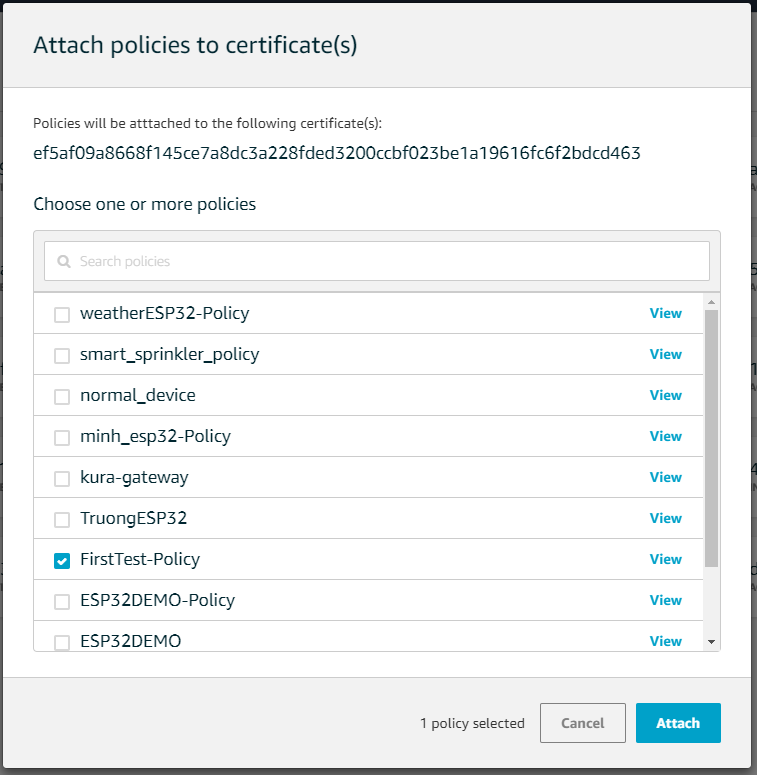


* Go to **Security->Policies->Create** to create a policy for it
* Name your policy and add statements for it (I named mine *FirstTest-Policy*)

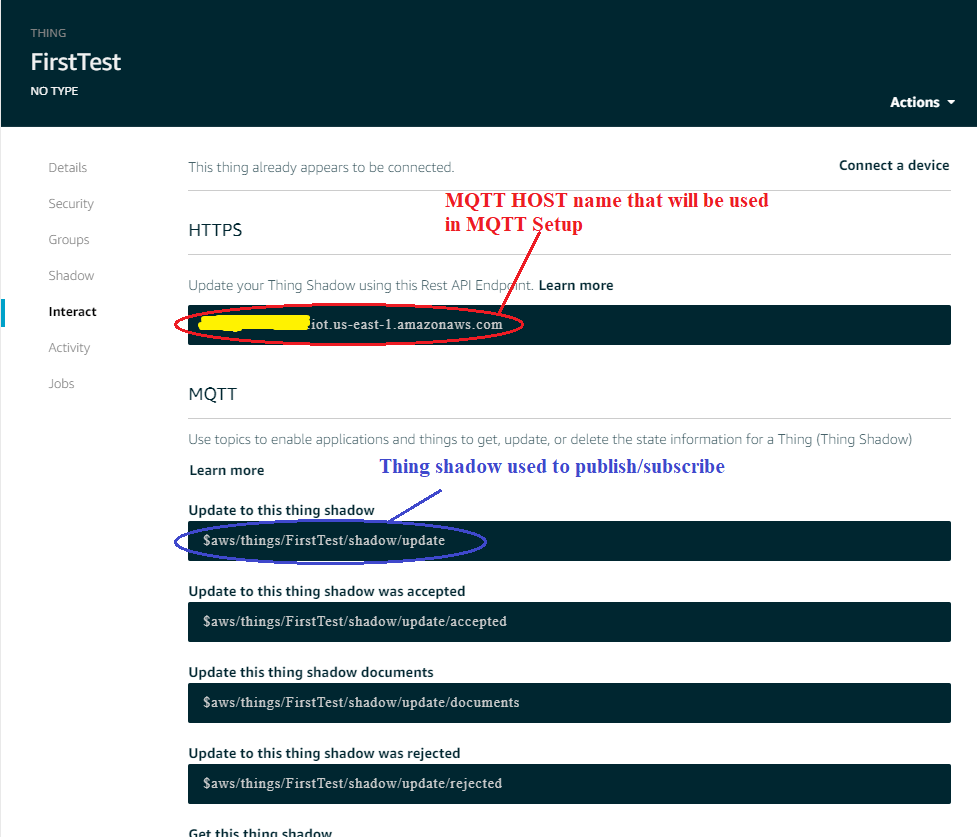


* Now go back to **Certificates** and attach the **Policy** that was defined above, you can view your **Things** **Certificates** in the **Security** section if there are many certificates



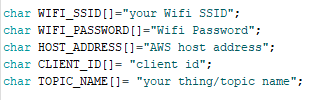


* In the **Interact** section of your **Thing**, please mind that



## **Run the device**

* Open the pubSubTest example of AWS\_IOT library
* Change these with yours, the **CLIENT\_ID** can be named what ever you want, for the **TOPIC\_NAME**, I chose the Thing Shadow update: **$aws/things/<your-thing-name>/shadow/update**

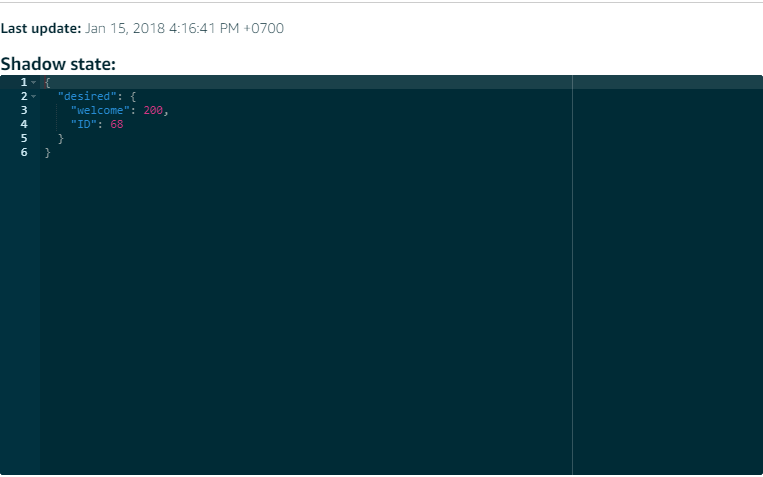


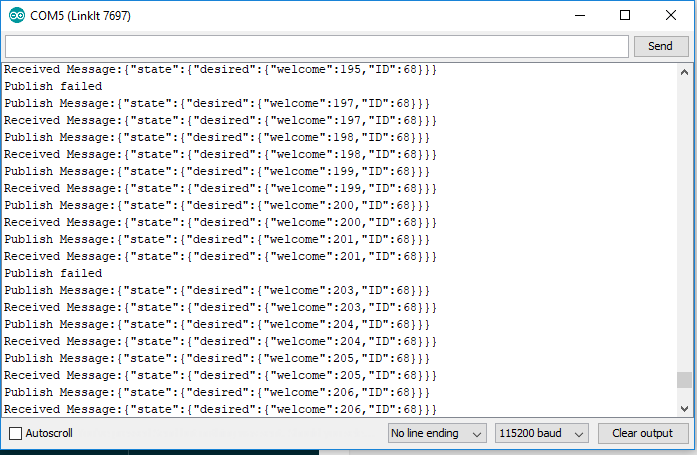
* Copy the containing of certificates **(root-CA, certificate, private-key)** into **ws\_iot\_certificate.c** file (**don’t delete the \n\**).
* Add some lines of code like the image below



* Upload the code

*Result*: **Thing** **Shadow** updates the counter value continuously and the device also receives the value in the subscribe message.





# **AWS with Rasberry Pi**

## **NodeJS**

This section will show you how to update data to AWS Thing Shadow using Node JS. A repository has been created, on your Raspberry Pi terminal

$ git clone [**https://github.com/nbxtruong/AWS-IoT-Demo**](https://github.com/nbxtruong/AWS-IoT-Demo)

$ cd AWS-IoT-Demo

$ npm install

Then create a folder call “**cert**” and copy all your certificates in it **(root-CA, certificate, private-key)**

Open ThingShadow.js file and do the following steps

* edit the all certificate names in these lines

var thingShadows = awsIot.thingShadow({

keyPath: './cert/<your-private-key>.<key-format>',

caPath: './cert/<your-ca>.<ca-key-format>',

clientId: ‘<your-client-ID>',

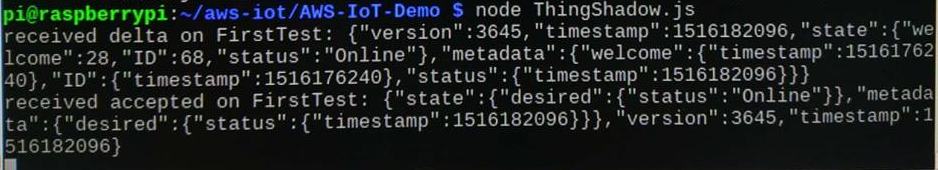
host: ‘<your-host-URL>'

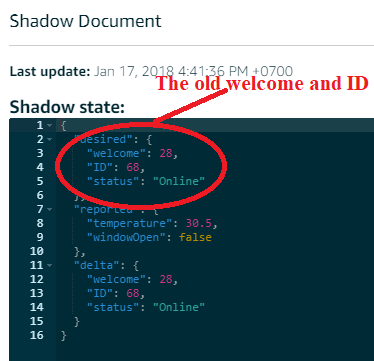
});

* edit your device name in **thingShadow.register** and **thingShadow.update** function

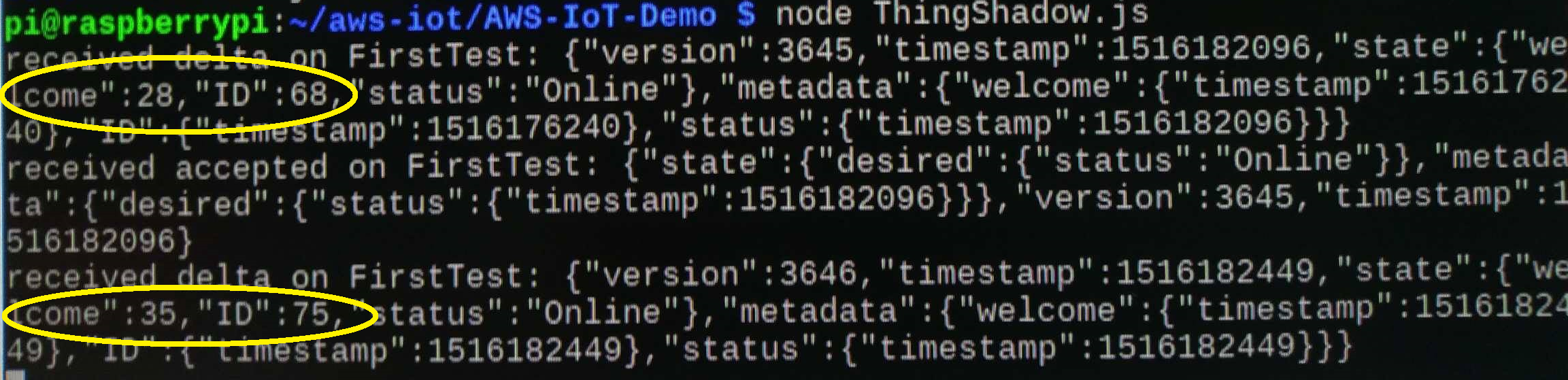
$ node ThingShadow.js

*Result*: **Thing** **Shadow** updates the counter value continuously…now get to the Shadow section on AWS and enter the new value for **welcome** field and **ID** field



* check your result on your Raspberry terminal, the 2 yellow circles prove that the values of **welcome** and **ID** field are also updated by the device



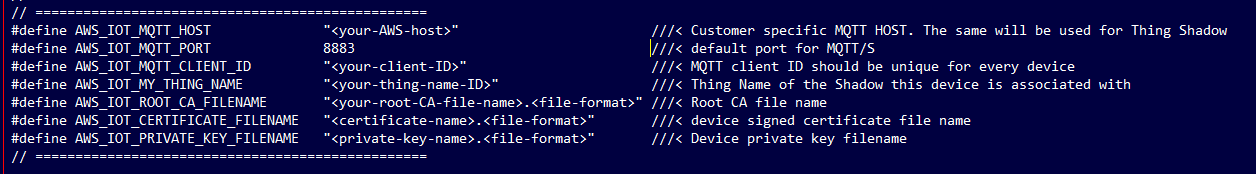
## **Embedded C**

## This section will show you how to publish/subsribe and update data to Thing Shadow using the Embedded C SDK

$ cd aws-iot-device-sdk-embedded-C

## Copy your certificate, private key, and root CA certificate into the certs directory

* Go to **sample\_apps/subscribe\_publish\_sample** directory and config the **aws\_iot\_config.h** as follow



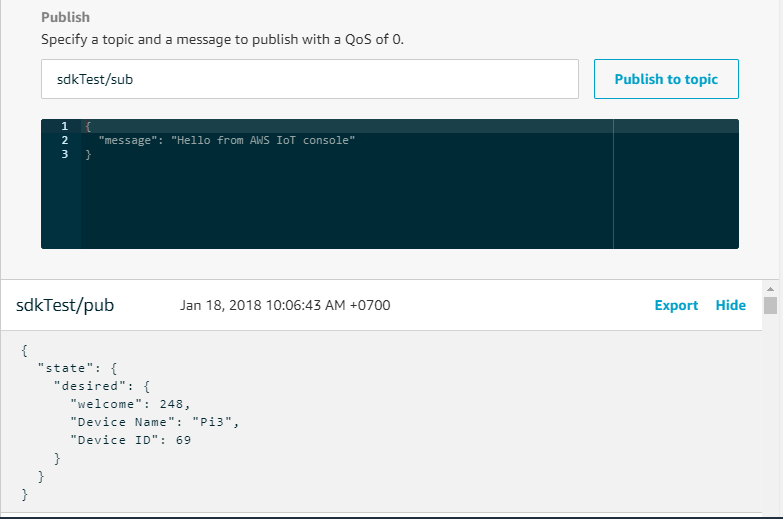
* Replace the subscribe\_publish\_sample.c file with the subscribe\_publish\_sample.c attached with this document

$ cd aws-iot-device-sdk-embedded-C

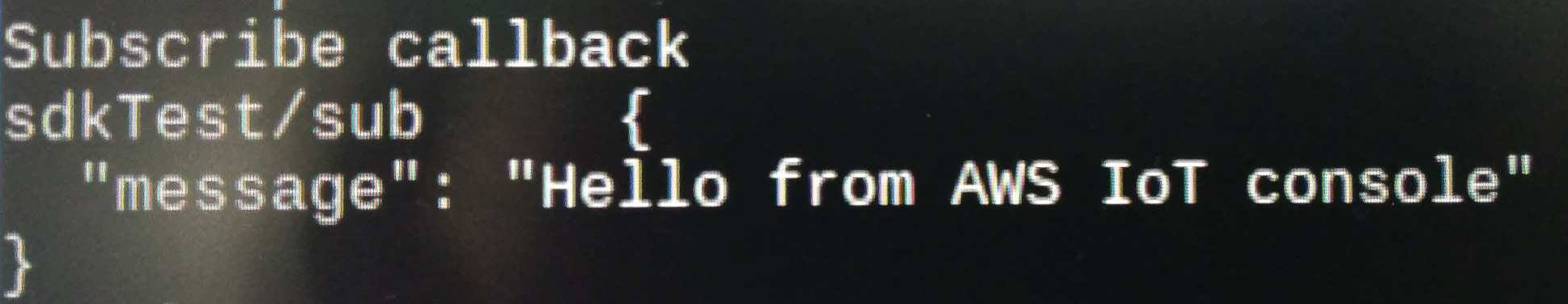
$ ./subscribe\_publish\_sample

* Go to the *Test* section of your AWS Thing online, subscribe to topic *sdkTest/pub* and…here is what you have

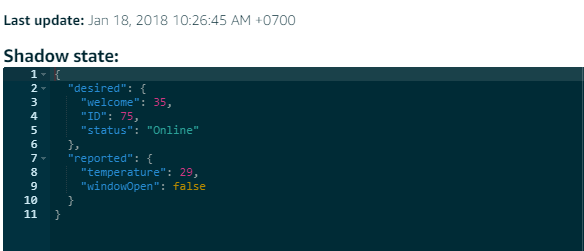
*Result*: your message has been uploaded successfully. If you press ***Publish to topic***, go to your Raspberry terminal

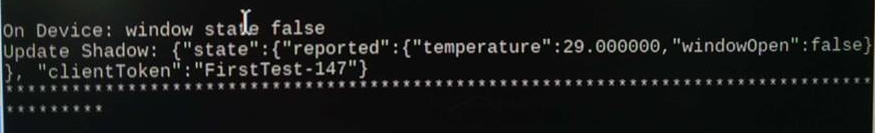


*Result*: your message is received by the Raspberry Pi



* In case you want to publish/subscribe to **Thing Shadow**, go to the ***shadow\_sample*** directory, do the same configuration and run the sample
* Check your **Thing Shadow** and Raspberry Terminal





# **REFERENCE**

For more details, please have a look at

* <https://docs.aws.amazon.com/iot/latest/developerguide/iot-embedded-c-sdk.html>
* <http://exploreembedded.com/wiki/AWS_IOT_with_Arduino_ESP32>
* <https://aws.amazon.com/documentation/iot/>

Last but not least: please read the official document about AWS IoT Developer Guide for an ***extremely detailed guidance***