

**Samsung Training Lab**

**ARTIK Modules**

TABLE OF CONTENTS

Lab 2: Create a Device Type 2

Lab 3: ARTIK 530s onboarding 5

Lab 4: OTA 6

4.1 Enable device management and OTA capabilities 6

4.2 Upload the OTA image to ARTIK cloud 8

4.3 Trigger the OTA Update 10

Lab 5: Running Google Assistant on ARTIK 530s (Python) 13

5.1 Configure and Test the Audio on ARTIK 530 13

5.2 Configure an Actions Console Project 13

Configure an Actions Console Project 13

Set activity controls for your account 14

5.3 Register the Device Model 15

Create Model 16

Download credential file and transfer it to your ARTIK530s 17

Specify traits 17

5.4 Install the SDK and Sample Code 18

Generate credentials 18

5.5 Run the sample app 18

Project ID 19

Device Model ID 19

Run the Sample App 19

5.6 Define your own actions 20

Bonus Question: 21

Lab 6: MQTT 22

6.1 Listen for MQTT message on ARTIK 530s 22

6.2 Connect to ARTIK 055s 23

6.3 Build an Application for ARTIK 055s 24

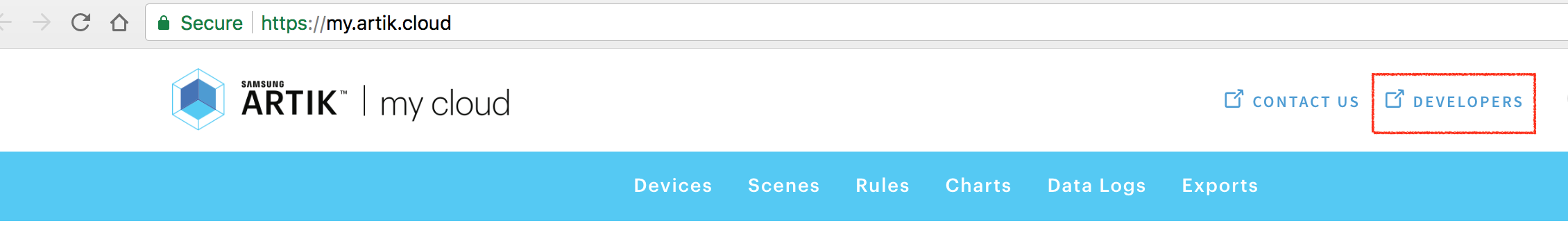
6.4 Publish MQTT message from ARTIK055s 27

6.5 Publish MQTT message from ARTIK055s 31

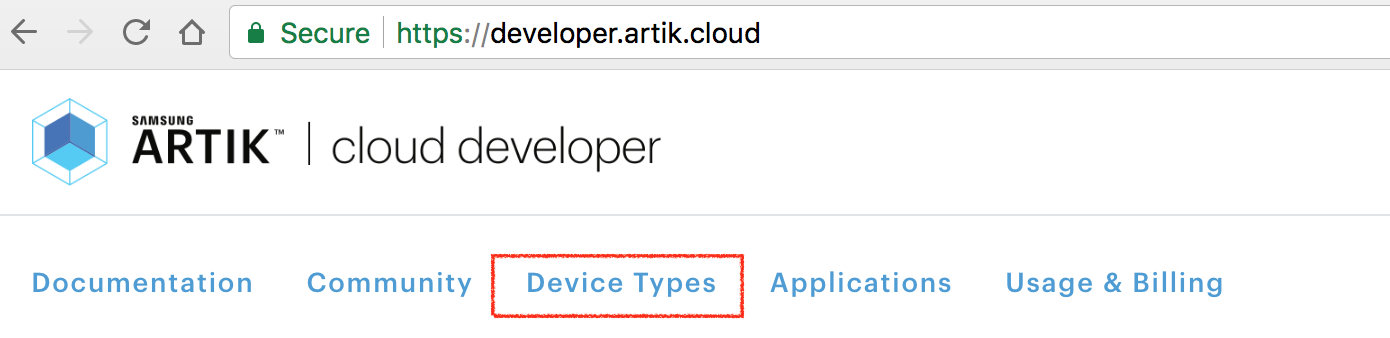
# Lab 2: Create a Device Type

In this lab, we are going to create a device type in ARTIK Cloud. Once you have the device type, we will show you how to connect to this device type from your ARTIK devices in the following labs.

1. Open the Developer portal of ARTIK Cloud by using this URL <https://developer.artik.cloud/> or by clicking on “ARTIK CLOUD DEVELOPERS” from ARTIK Cloud user portal.



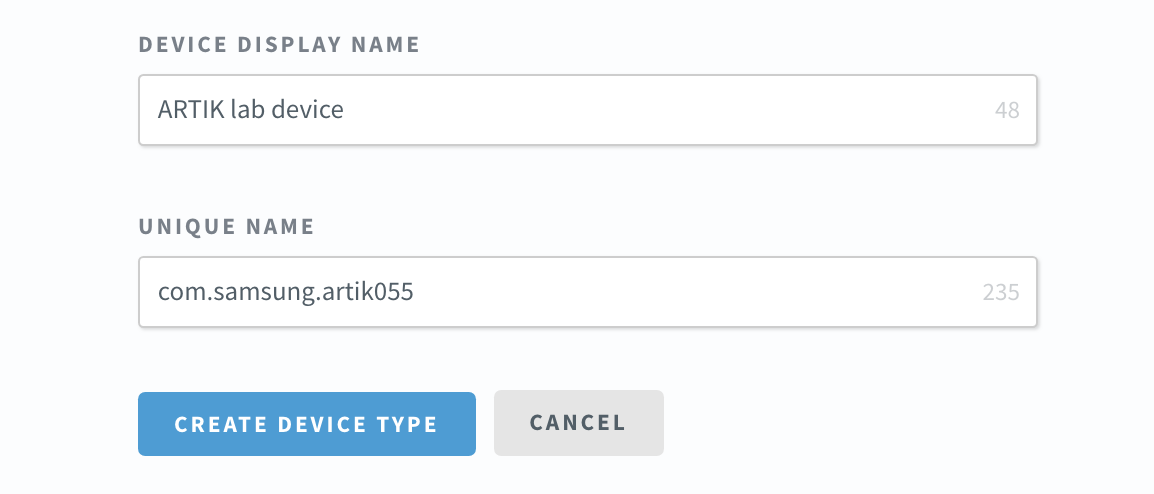
2. Click on Device Types



3. Click on “+NEW” button



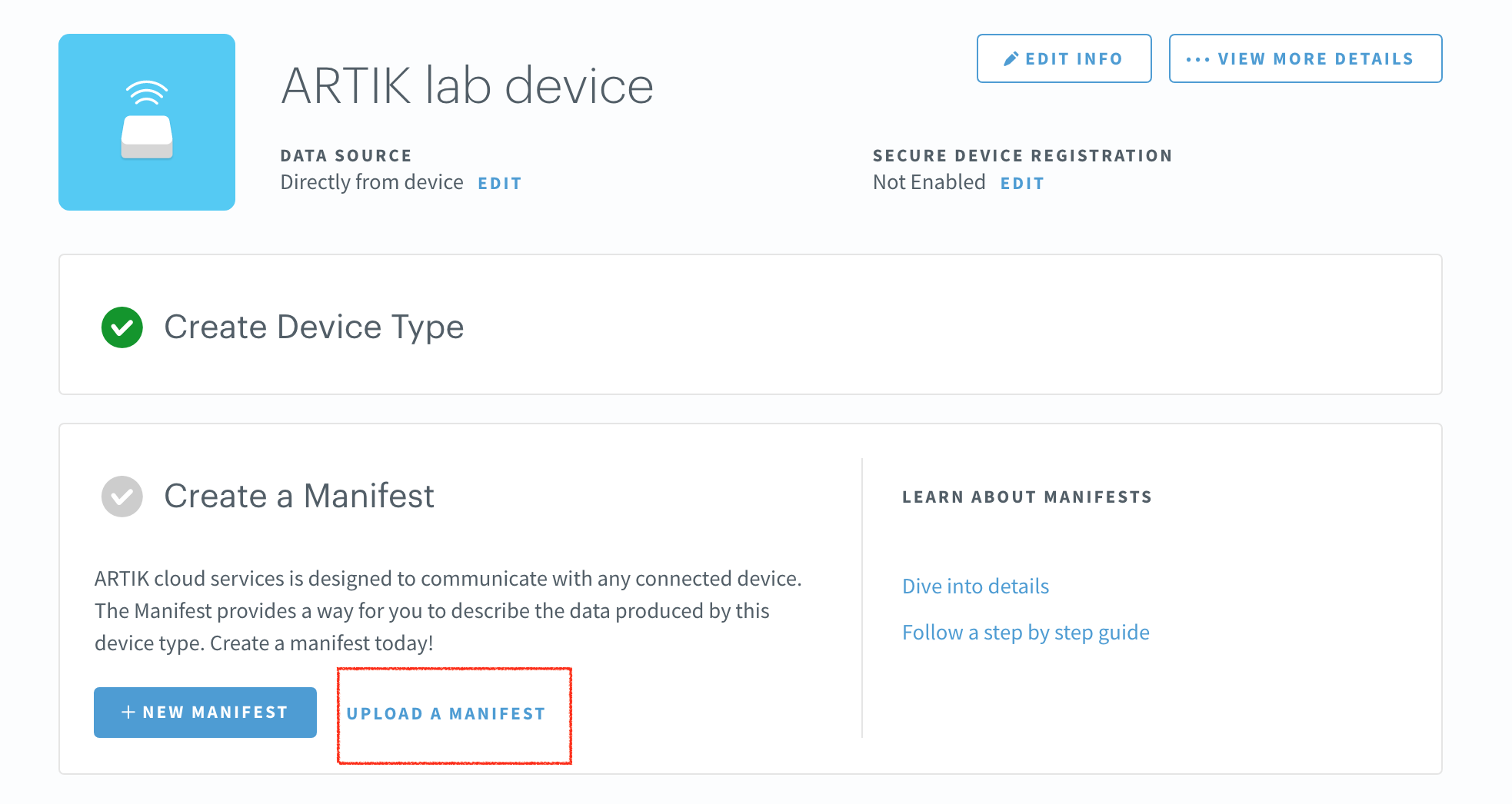
4. Give your device type a Device Display Name such as “your\_company lab device” (eg. ARTIK lab device ) and give it an unique name such as com.your\_company.your\_project\_device (eg. com.samsung.artik055) , then click on “CREATE DEVICE TYPE”. This will enable us to create a digital representation for our device type in ARTIK Cloud. (compare this to reserving a name space for a data type in programing languages)

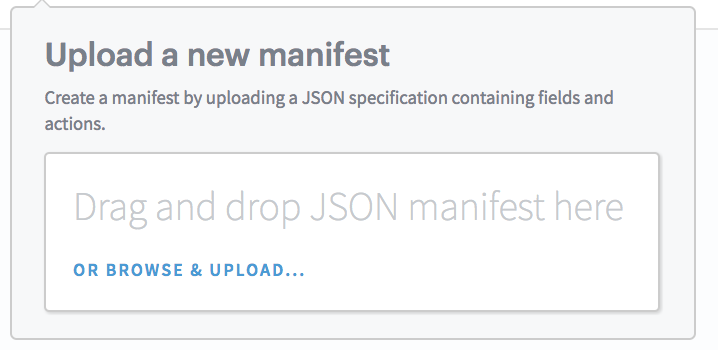


1. Now a name space is reserved for the device type. Actual definition of the device type needs to be done further. This is done using a **Manifest**. A Manifest is the representation of a device’s data properties in ARTIK cloud. Any data, a device produces and sends to ARTIK cloud are represented as “Device Fields”. Any data, the device can respond to is represented as “Actions”. An action can have parameters as well, like an argument.

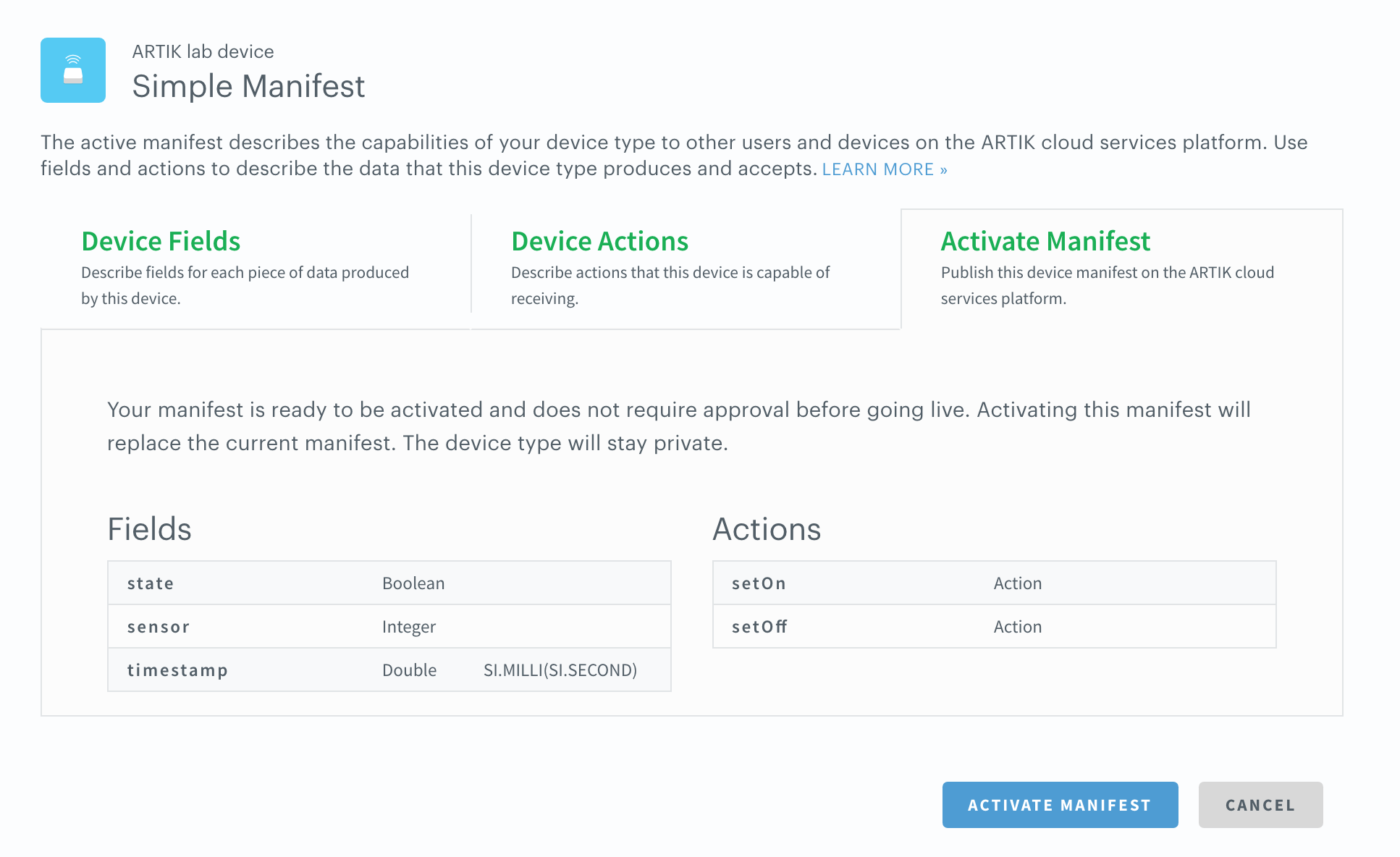
A Manifest can be created step-by-step using an intuitive user interface or by simply uploading from a JSON file. The steps below explain the creating a Manifest from a JSON file.

To create a manifest from a JSON file, select UPLOAD A MANIFEST.

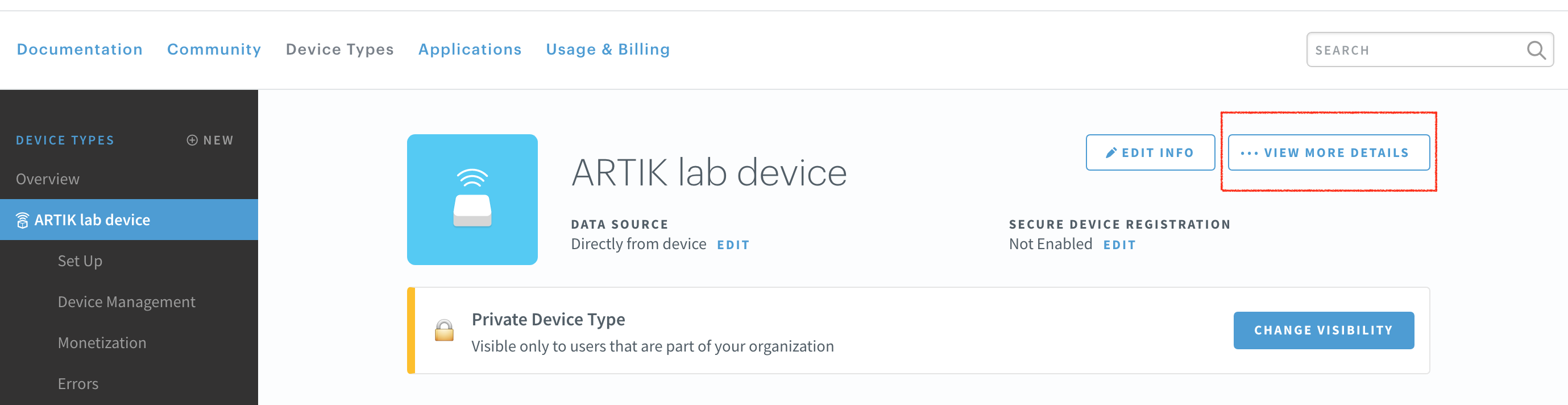


6. Upload the provided manifest file (com.samsung.artik-manifest.json)

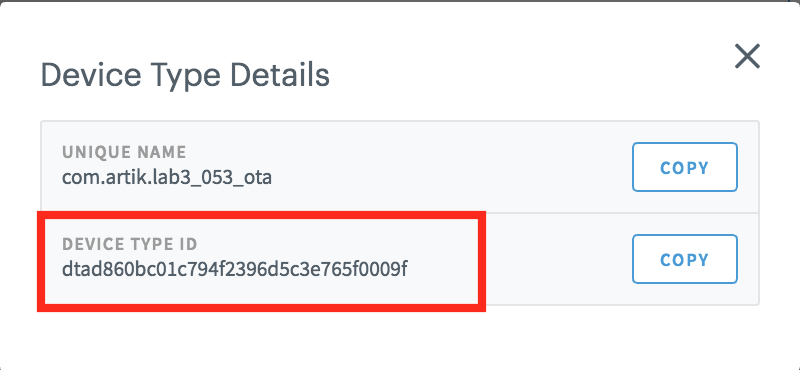
7. Upon uploading the JSON file, make sure the Manifest is as shown below. This Manifest represents a device that can send data to ARTIK Cloud for LED state, distance sensor reading and a timestamp. And it can also receive “setOn” and “setOff” actions from ARTIK Cloud in order to toggle the LED on the board.



1. Then activate the manifest, by clicking on the “ACTIVATE MANIFEST” button.
2. Make sure a device type is created. Access “VIEW MORE DETAILS” and copy the device type ID created



1. Make a note of the device type ID created. The Device type ID is dt­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ <please fill in>



1. Congratulations you have successful represented your device in ARTIK Cloud.

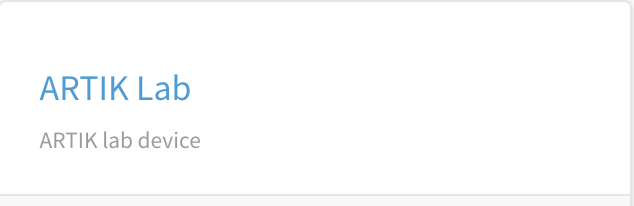
# Lab 3: ARTIK 530s onboarding

Please refer to “ARTIK\_BLE\_Onboarding\_V2.0.pdf”

**High-End Modules - ARTIK 530s, 710s**:

Press the button 403 on the board and use the ARTIK app to attach the high-end module (A530, A710) to the new device type being created.

Upon onboarding your ARTIK530s, a device should have been created in your ARTIK Cloud portal.

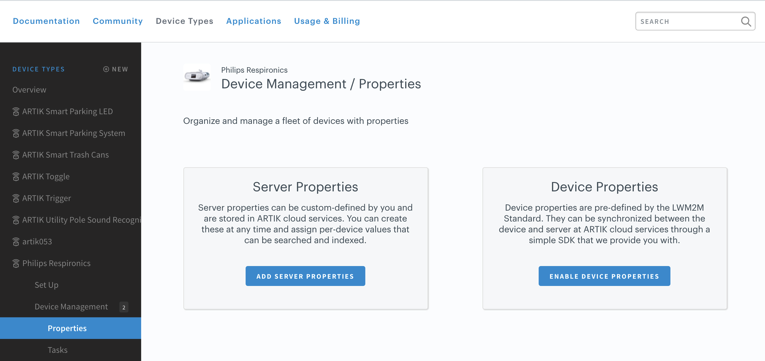


# Lab 4: OTA

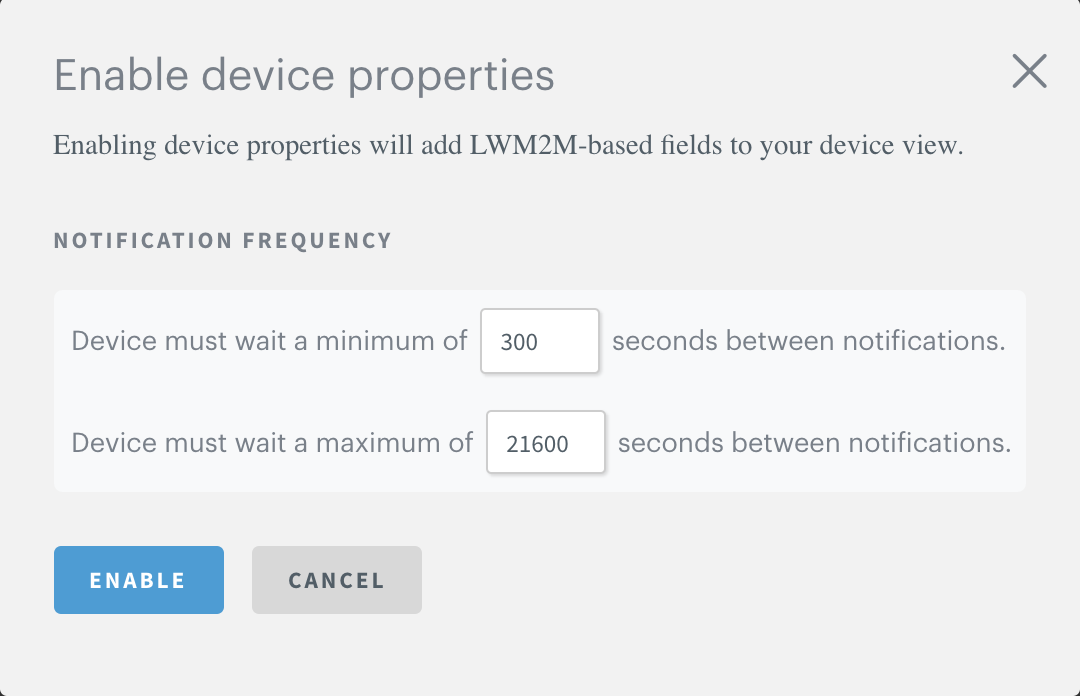
## 4.1 Enable device management and OTA capabilities

In this section, we will enable device management and Over-The-Air update capabilities for this device

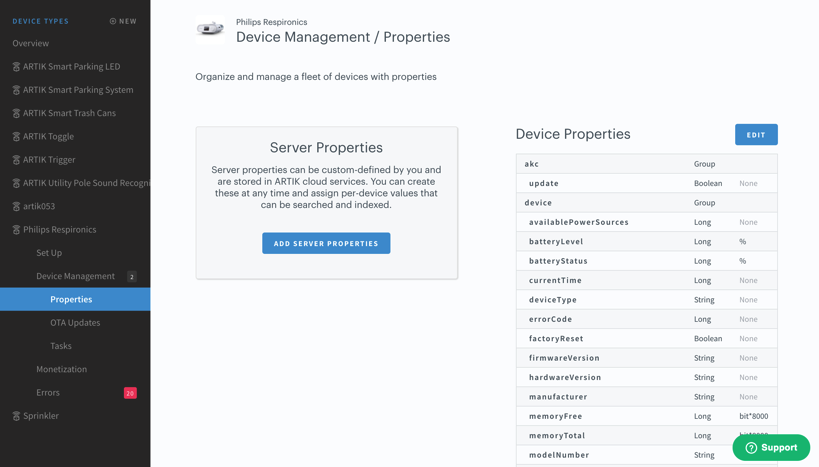
1. Open the Developer portal of ARTIK cloud by using this URL <https://developer.artik.cloud/> or by clicking on “ARTIK CLOUD DEVELOPERS” and open the list of device types as done previously in steps 1-3 of “CREATE DEVICE TYPE” section
2. Under the new device type created, go to “Device Management -> Properties ” and click on “enable device properties” to enable Light Weight Machine 2 Machine (LWM2M) based device management



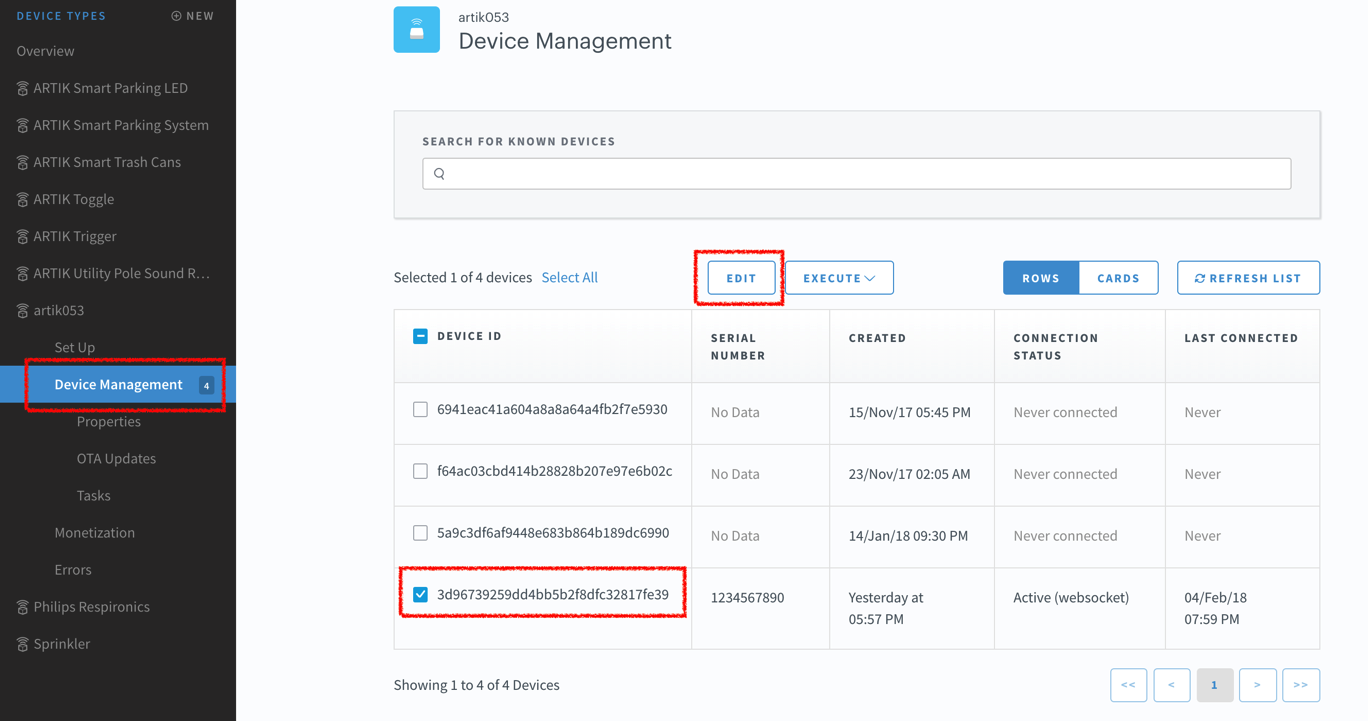
1. This provides an option for the device to connect with ARTIK Cloud at a determined frequency,

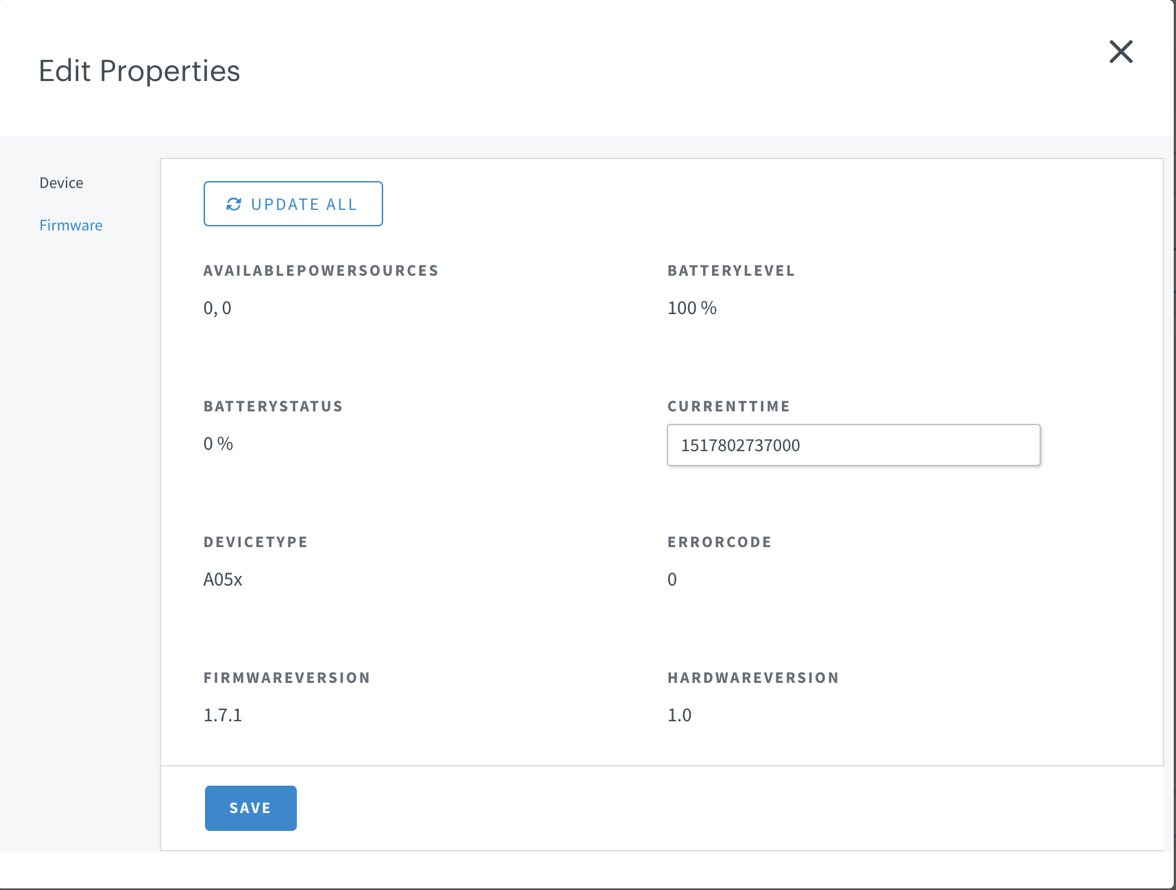


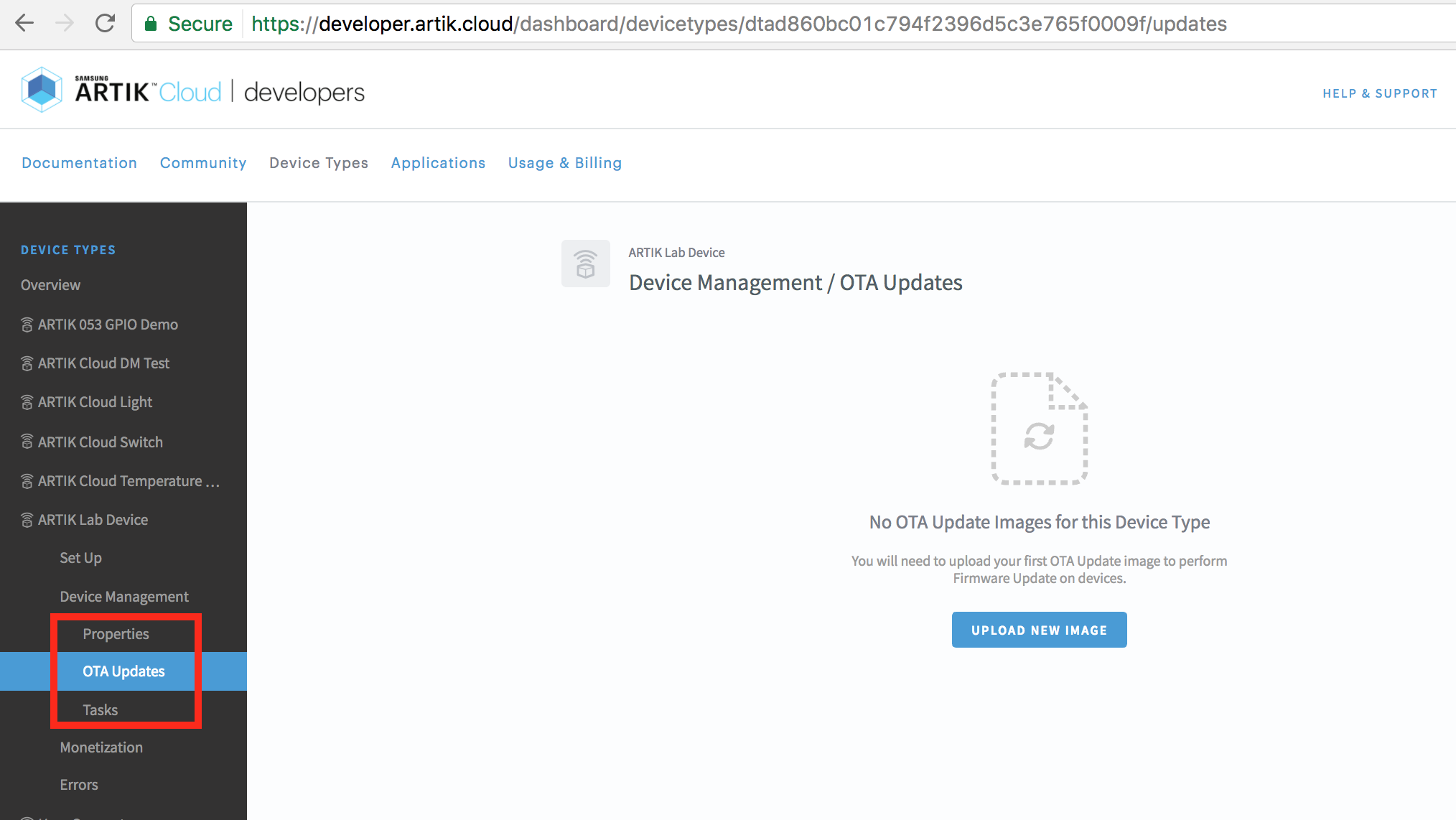
1. Upon successfully enabling the device properties, “OTA updates” will be available under device management for this device type and all the device properties will be available in ARTIK cloud. These properties are LWM2M device objects that are mirrored in ARTIK cloud from the device, based on the frequency defined earlier.



If you want to see an example of Device Properties, select Device Management of your device type on the left pane, click the Device ID of the connected device



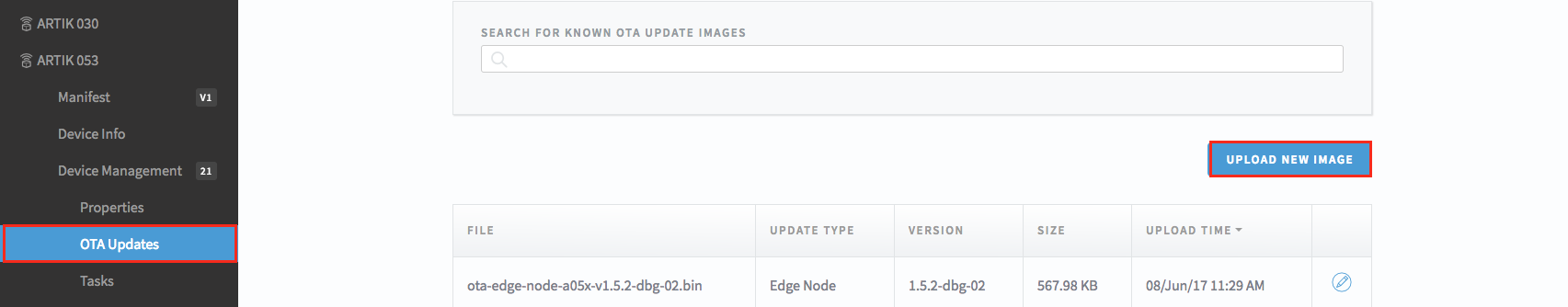




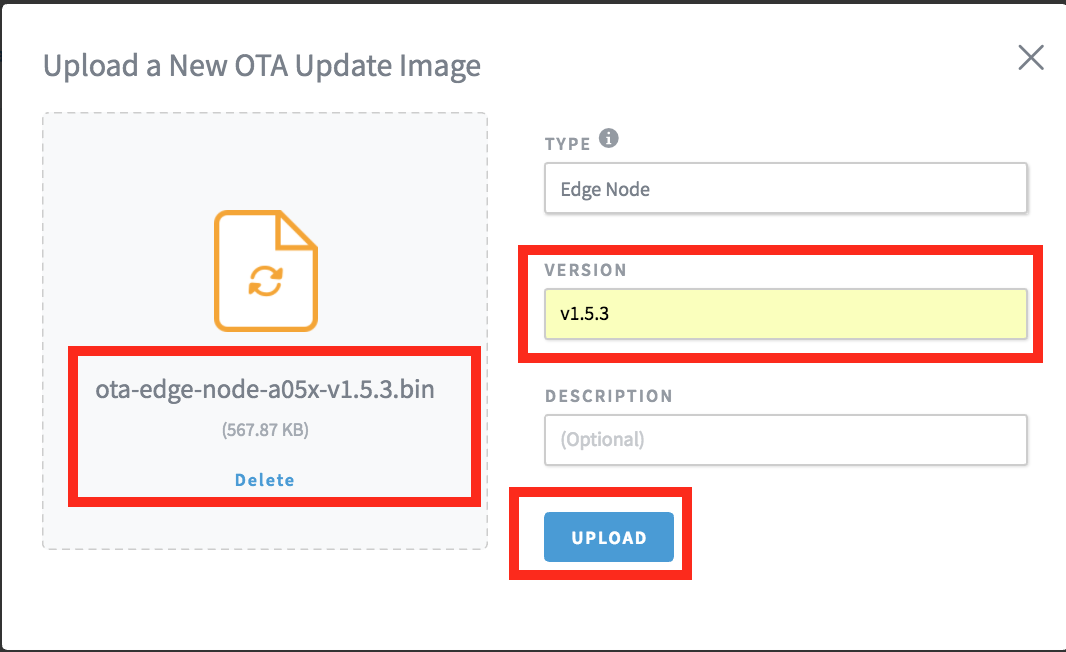
1. Congratulations this enables OTA images to be updated to the user account

## 4.2 Upload the OTA image to ARTIK cloud

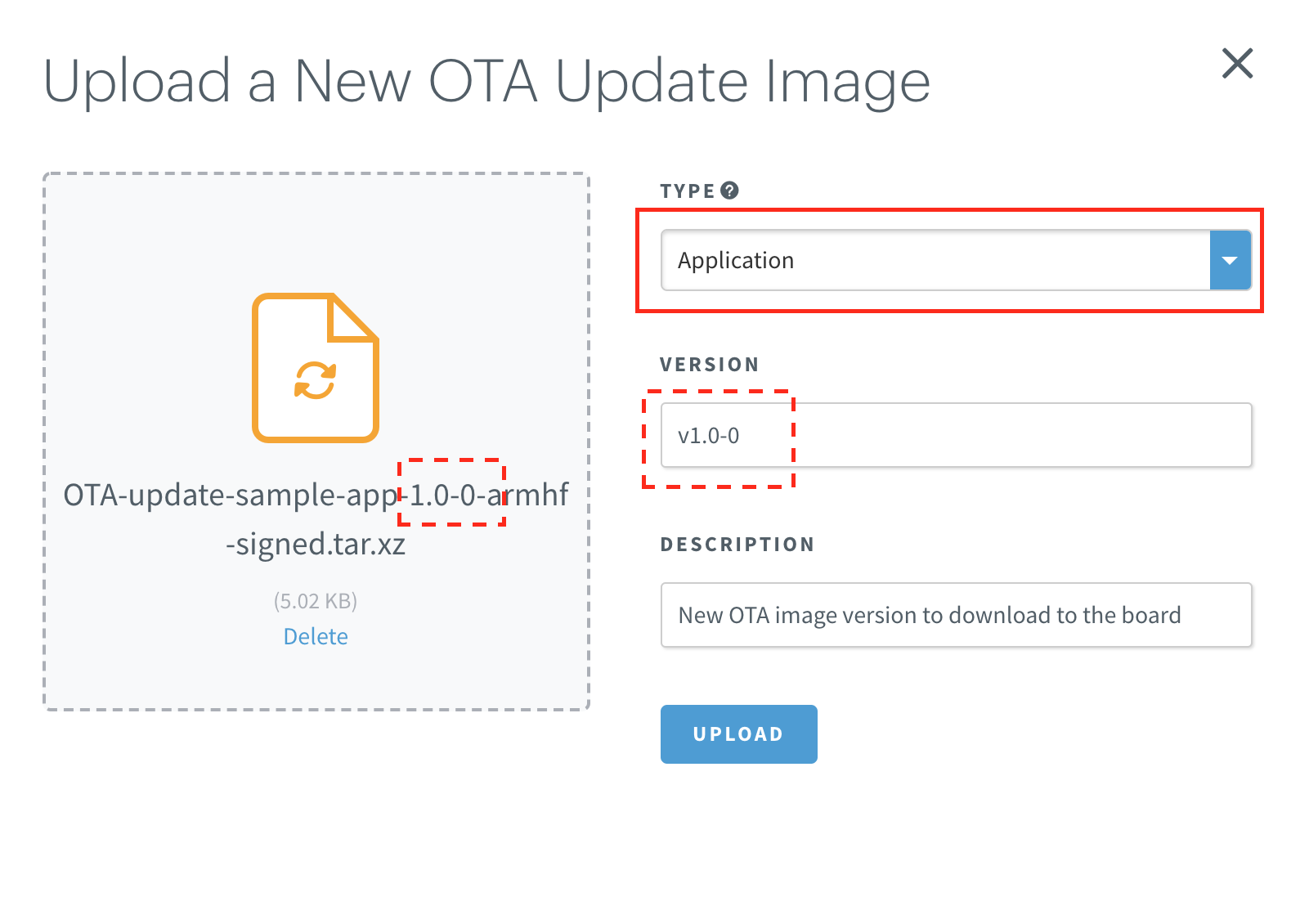
1. Upload the generated package to ARTIK cloud. In ARTIK Cloud's developer portal, go to the Device Management tab of the relevant device type, then click the OTA Updates sub-tab, then click the Upload new image button.



1. **ARTIK 05x**: In the upload window, **select Edge Node for ARTIK05x**as the image type then enter the version for the package, this string must exactly match the one you set earlier with ONBOARDING\_VERSION. Use the browse button to select the prepared package to upload



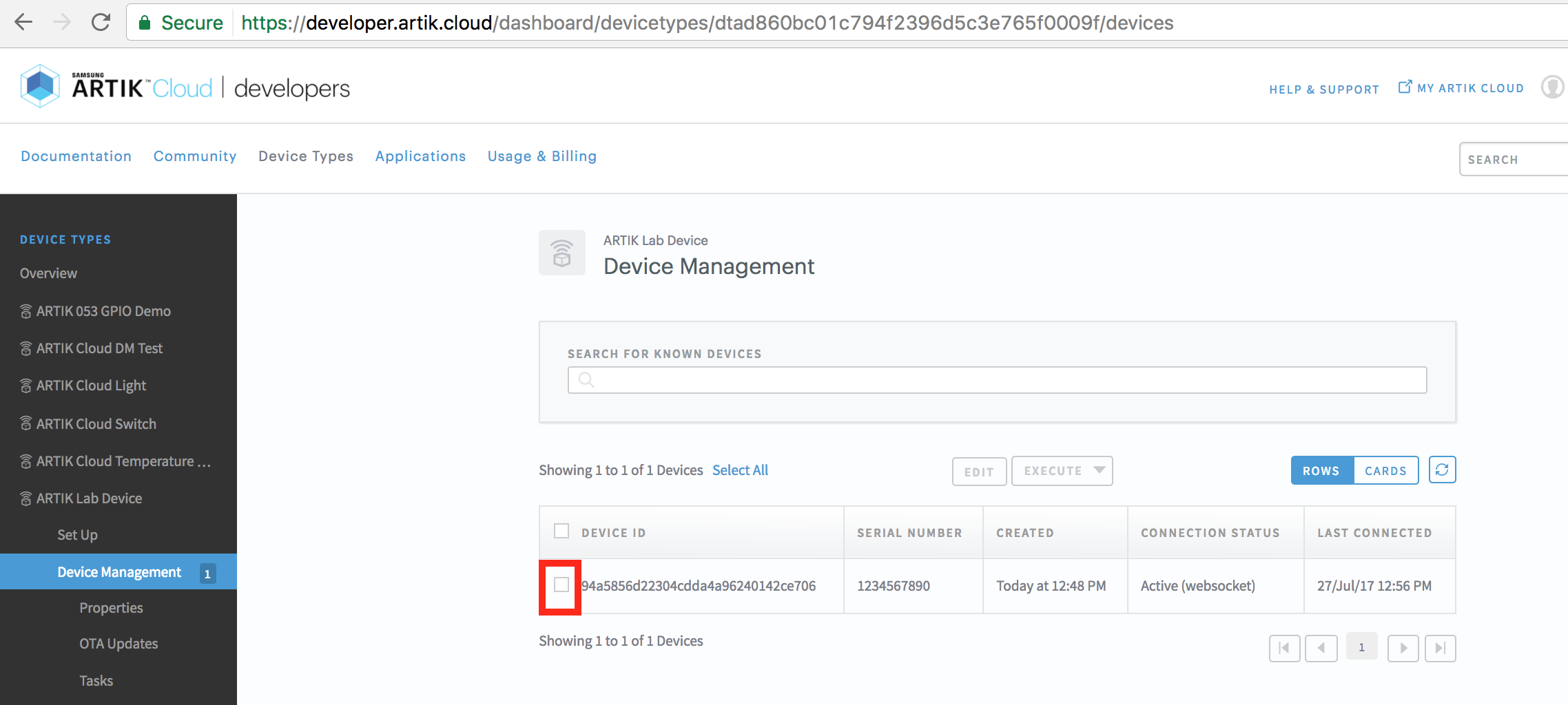
1. **ARTIK 530, 530s and 710s**: In the upload window, **select Application** as the image type then enter the version for the package, this string must exactly match the one you set earlier with ONBOARDING\_VERSION. Use the browse button to select the prepared package to upload



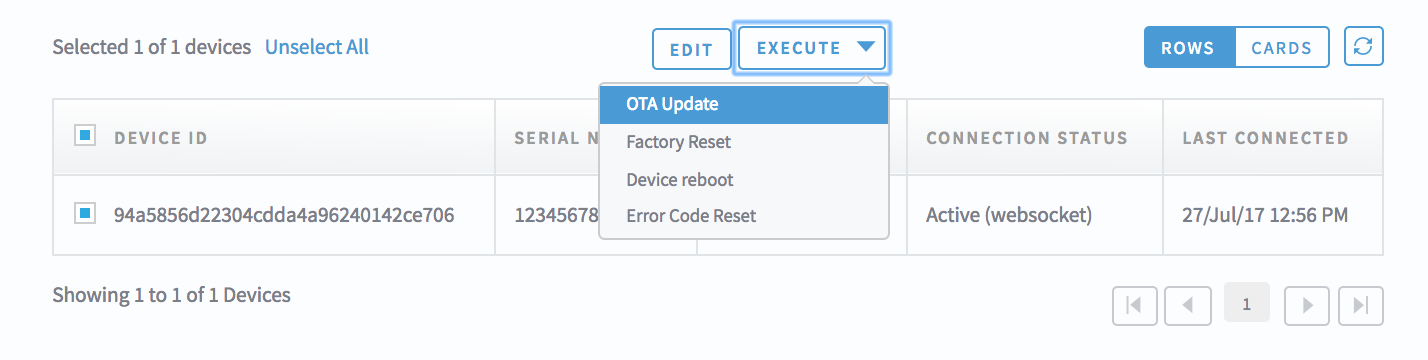
1. Click the Upload button to finalize the uploading. Note that after uploading a firmware, it is impossible to delete it from ARTIK Cloud portal.

## 4.3 Trigger the OTA Update

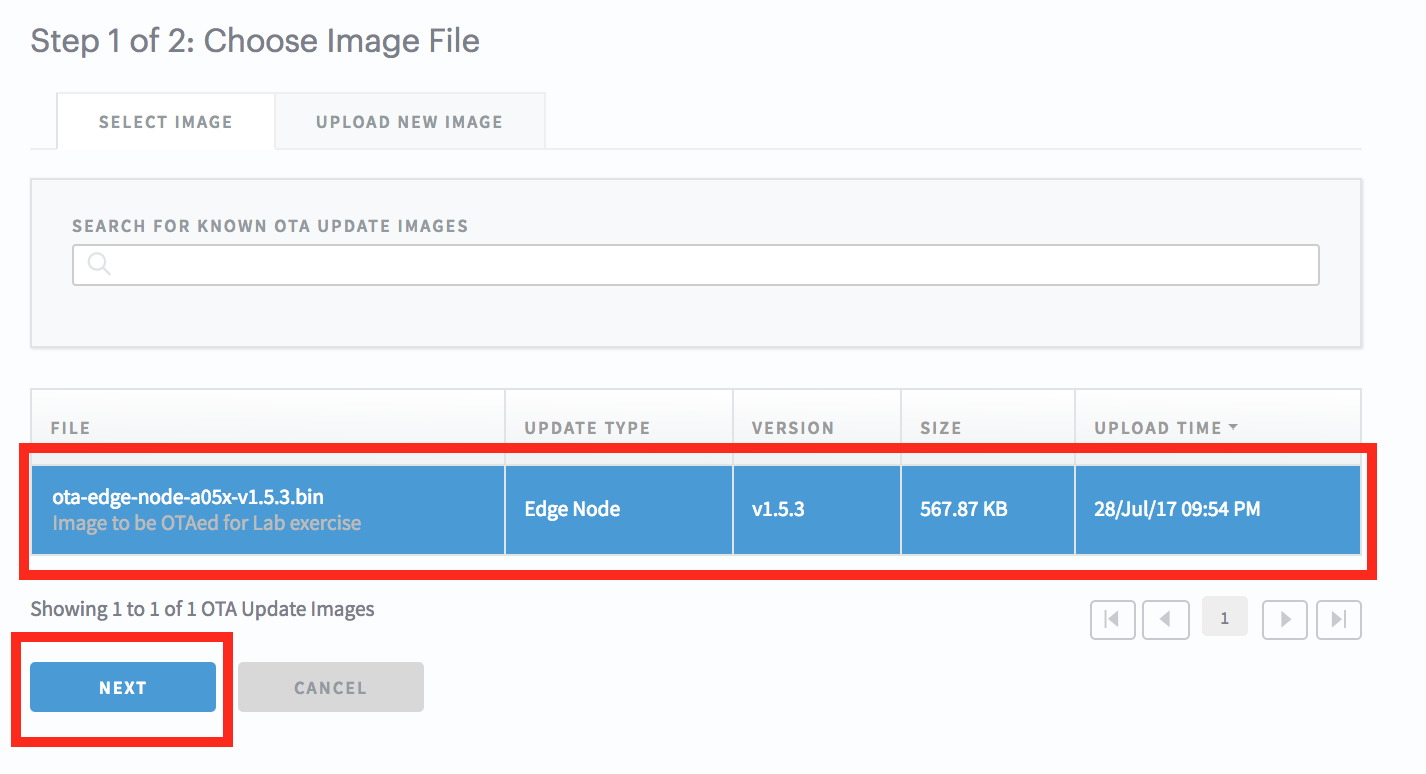
1. Open ARTIK Cloud's developer portal, and browse to the device type's Device Management pane. In the devices list, identify the currently connected board by its device ID



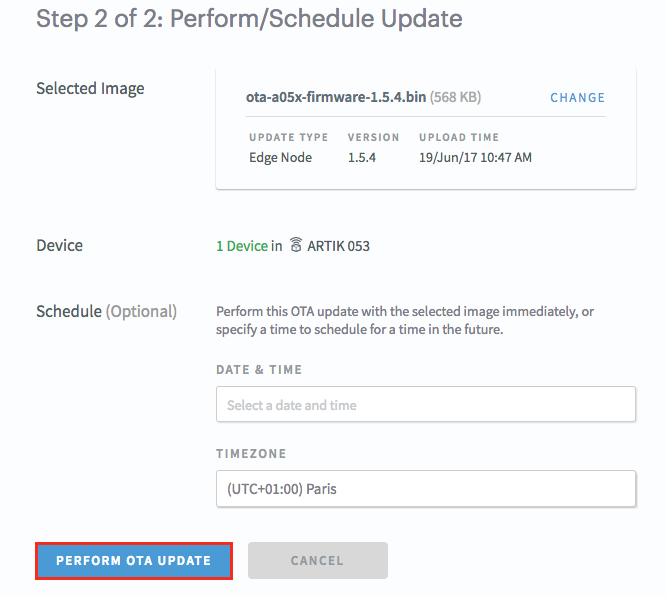
1. Select the device (or devices) you want to update, click Execute then OTA update



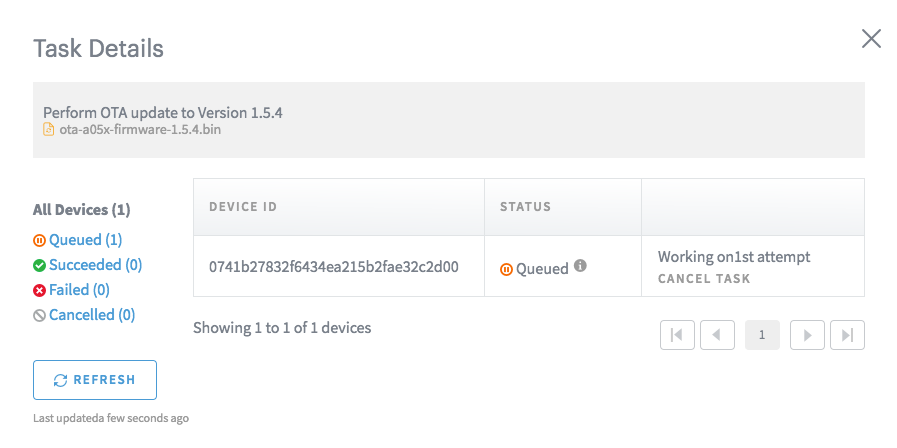
1. Select the package you want to update, then click Next.



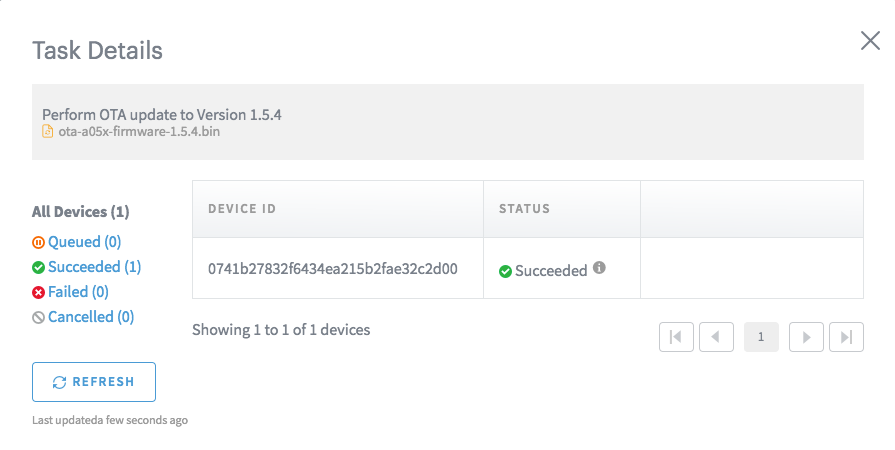
1. In the next window, you can select a date to schedule the update, or leave empty to launch the update right now. Click Perform OTA Update when ready.



1. To follow the update process from ARTIK Cloud's portal, go to the Tasks subtab under your device type's Device Management pane. Identify the OTA update task from the list then click it to get the details.



1. On the ARTIK Cloud portal side, the task status will be updated after a bit (usually a couple minutes) to reflect the success of the OTA update



# Lab 5: Running Google Assistant on ARTIK 530s (Python)

In this lab, we will install Google Assistant on ARTIK 530s

## 5.1 Configure and Test the Audio on ARTIK 530

As the 1st step, let’s verify that recording and playback work on ARTIK 530:

1. Plug in your earplugs into ARTIK530 and play a test sound. From your terminal window, run

[root@artik ~]# speaker-test –t wav

Press the Ctrl + C when done.

If the volume is too low, you can adjust it by using alsamixer.Press the up arrow key to set the volume. Press ESC when you are done.

[root@artik ~]# alsamixer

[root@artik~]#arecord --format=S16\_LE --duration=5 --rate=16000 --file-type=raw out.raw

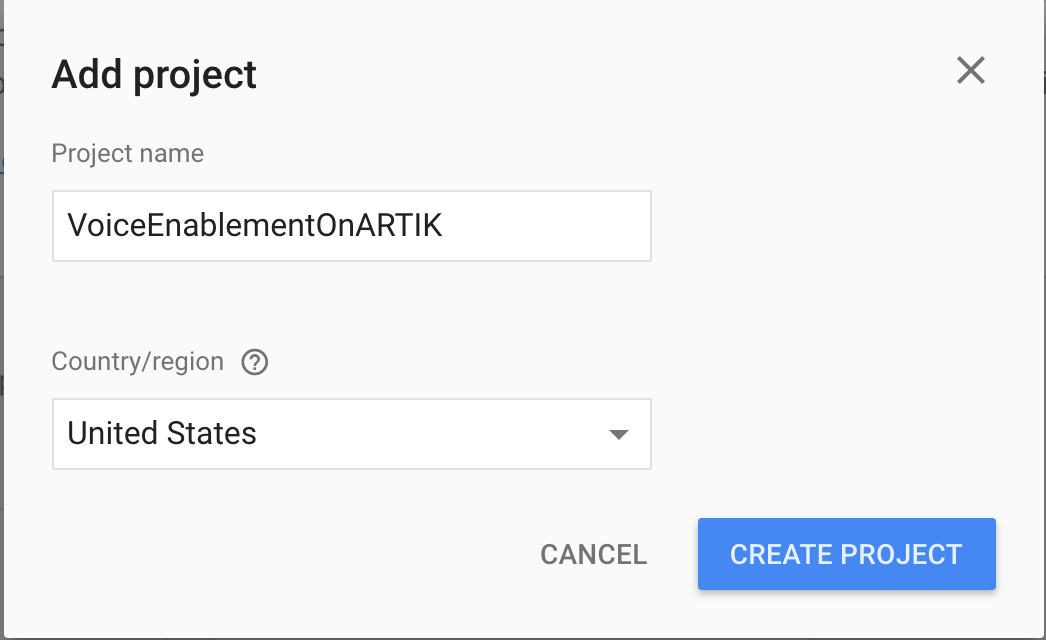
[root@artik~]#aplay --format=S16\_LE --rate=16000 out.raw

1. Record a short audio clip and play it back.

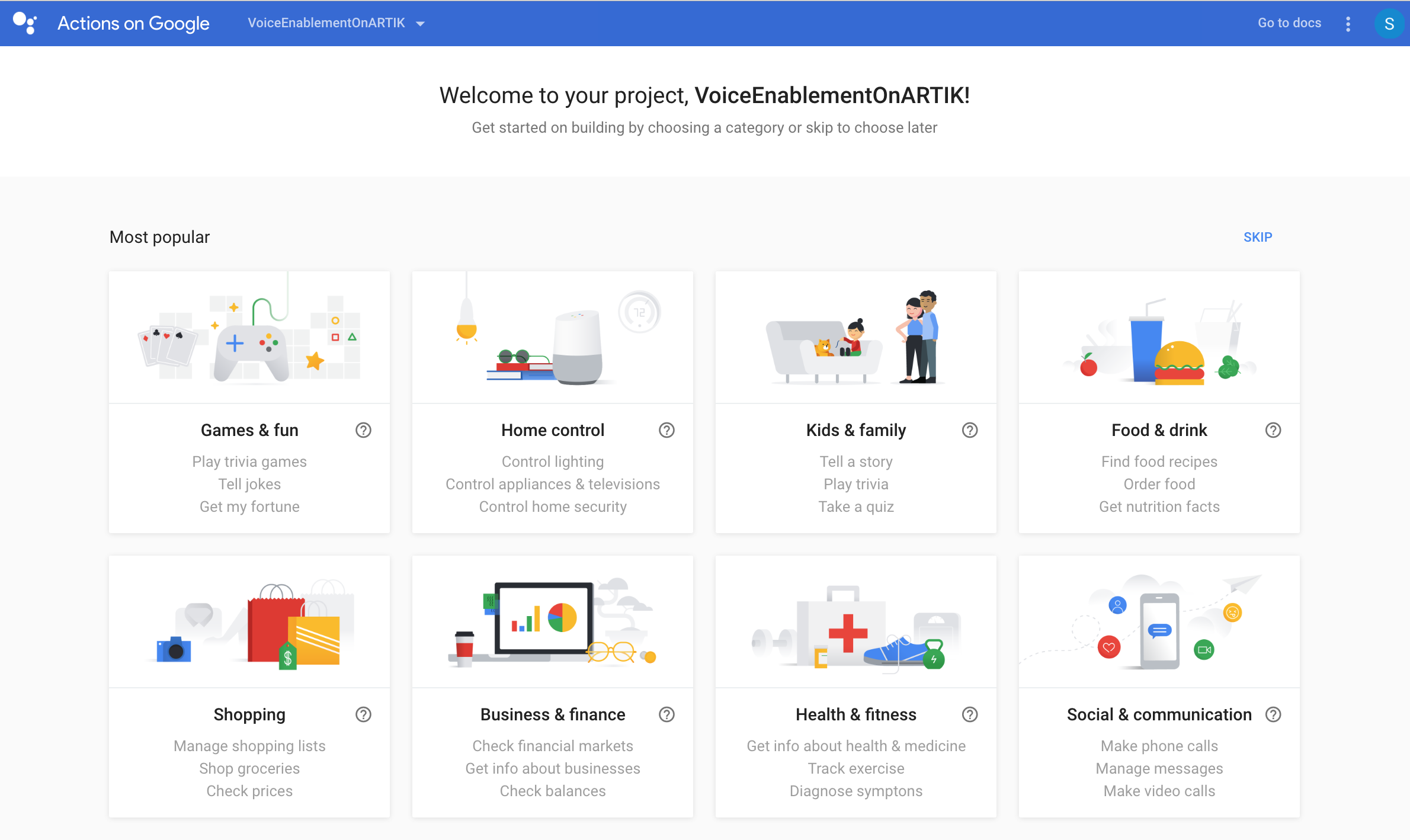
## 5.2 Configure an Actions Console Project

### Configure an Actions Console Project

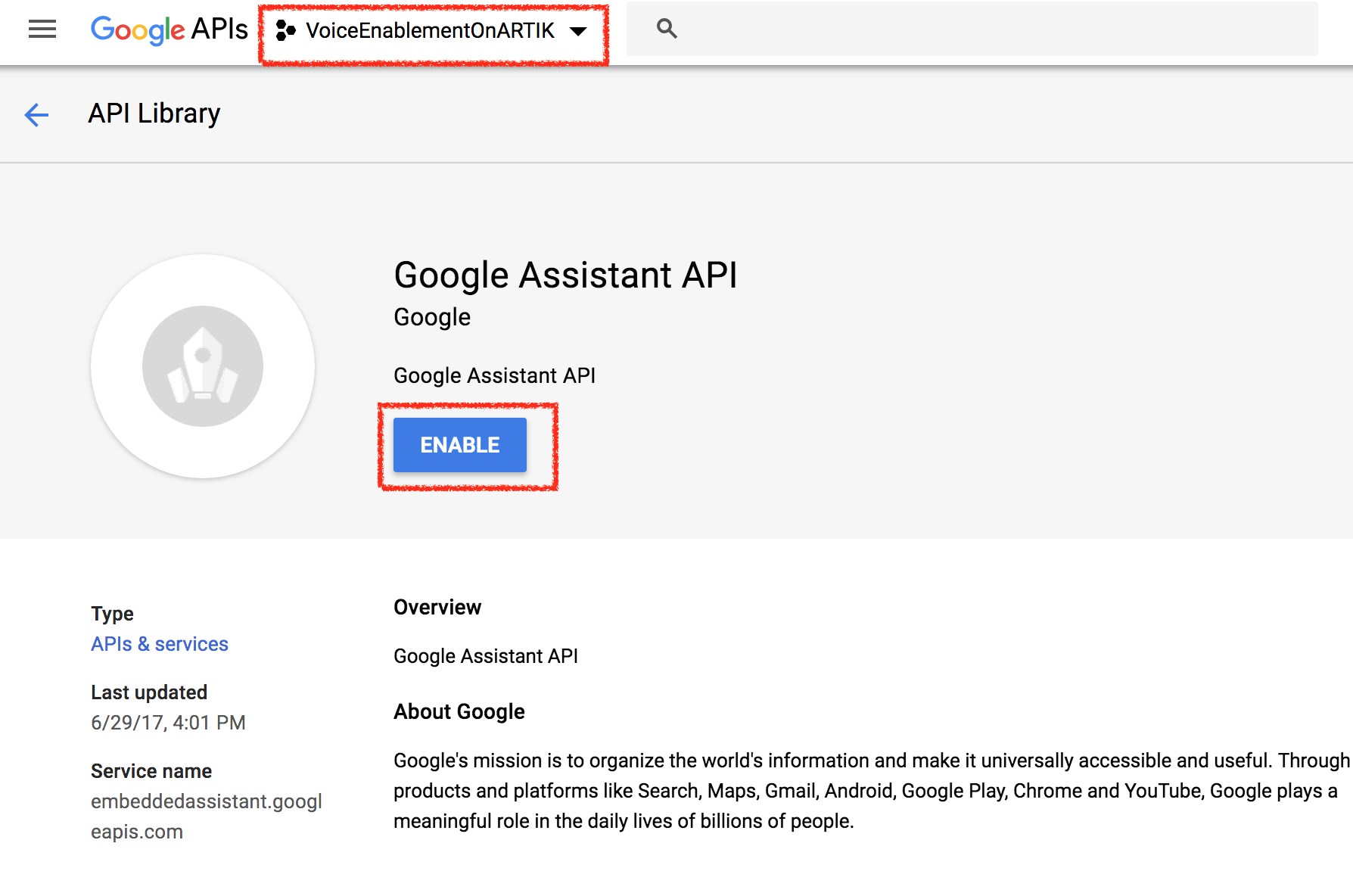
1. Open the Actions Console: <https://console.actions.google.com/u/1/>.
2. Click on Add/import project.
3. To create a new project, enter your Project name and click CREATE PROJECT



You will see the page below, keep your browser open.



1. Go to Google API Console, and enable Google Assistant API for the project you just created.



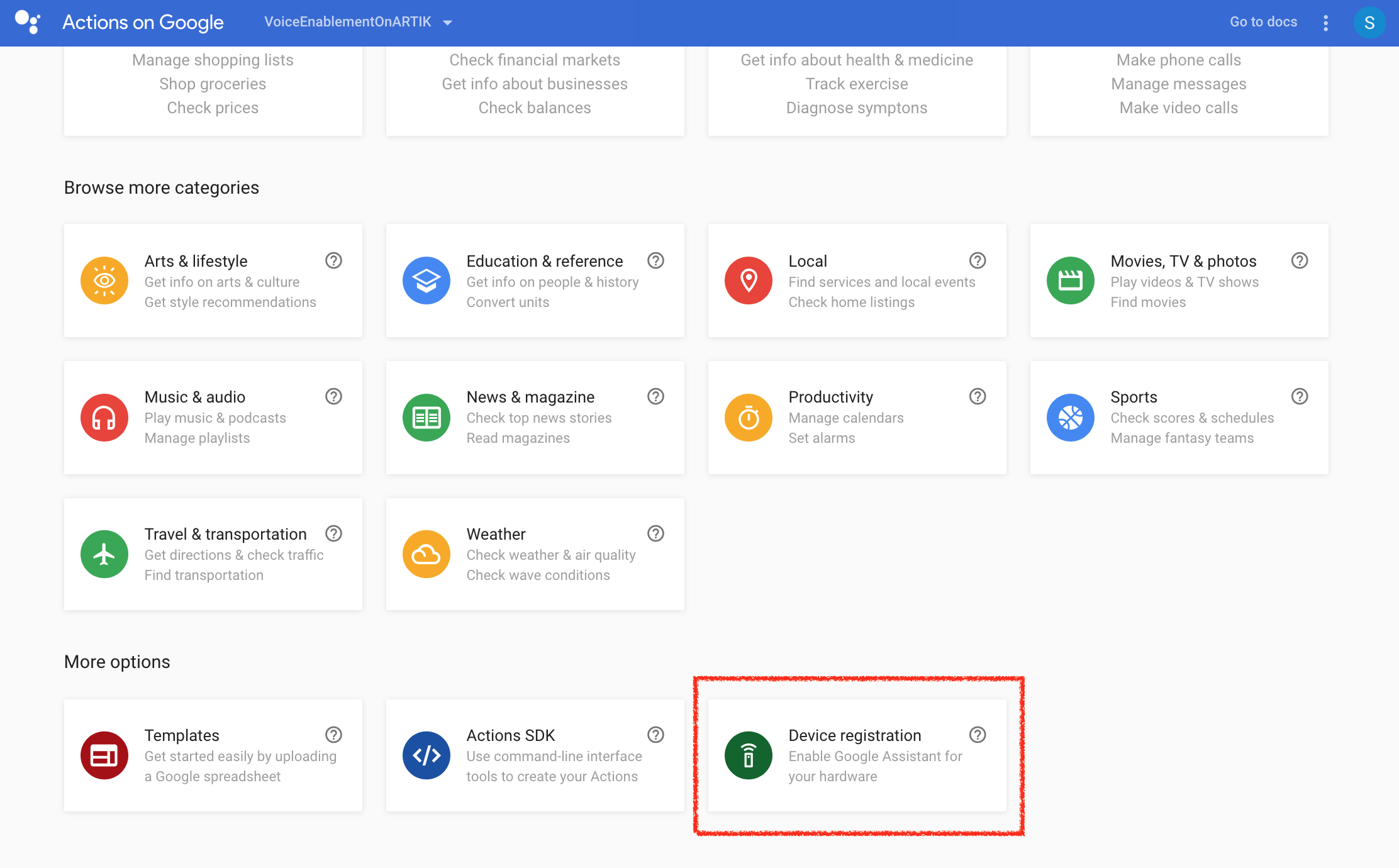
### Set activity controls for your account

Open the Activity Controls page of your Google Account https://myaccount.google.com/u/1/activitycontrols, and enable the options below.

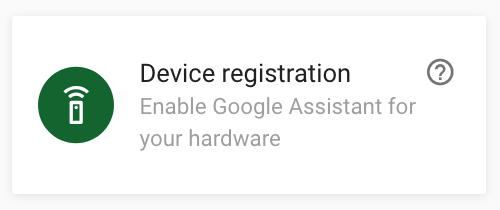
* Web & App Activity: In addition, be sure to select the **Include Chrome browsing history and activity from websites and apps that use Google services** checkbox.
* Device Information
* Voice & Audio Activity

## 5.3 Register the Device Model

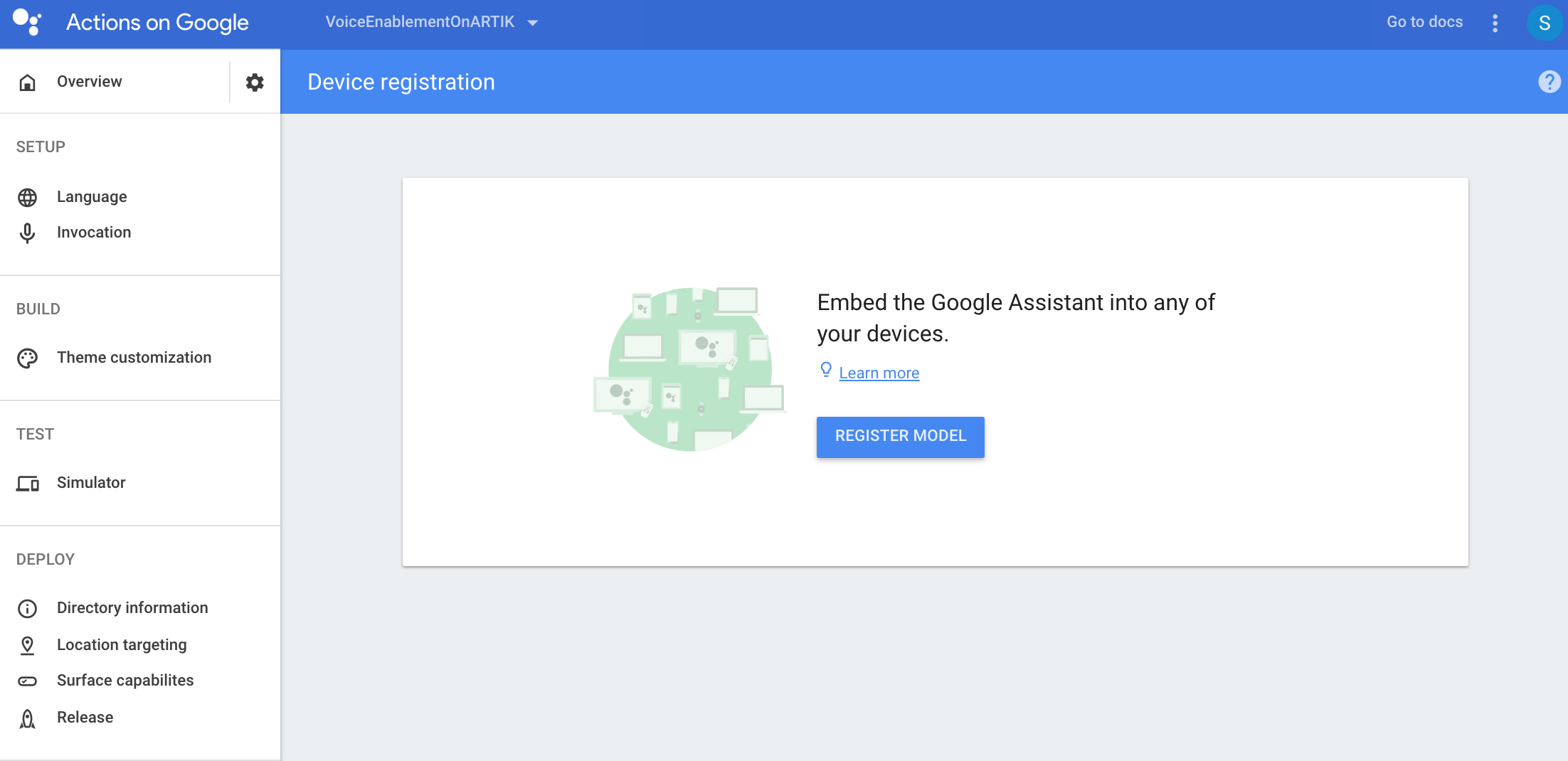
1. Go back to the browser page where we created our Actions project, scroll to the bottom, and you should see a “Device registration” button under “More options”.



Click the button

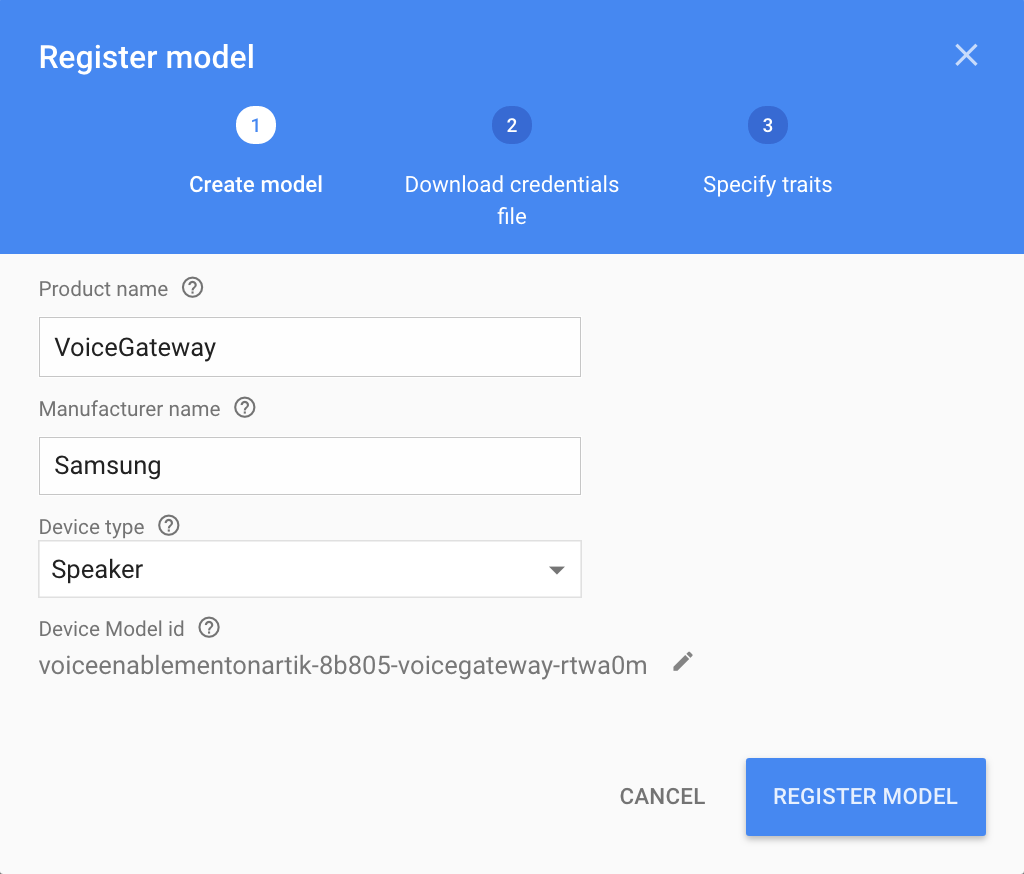


2. click the REGISTER MODEL button.

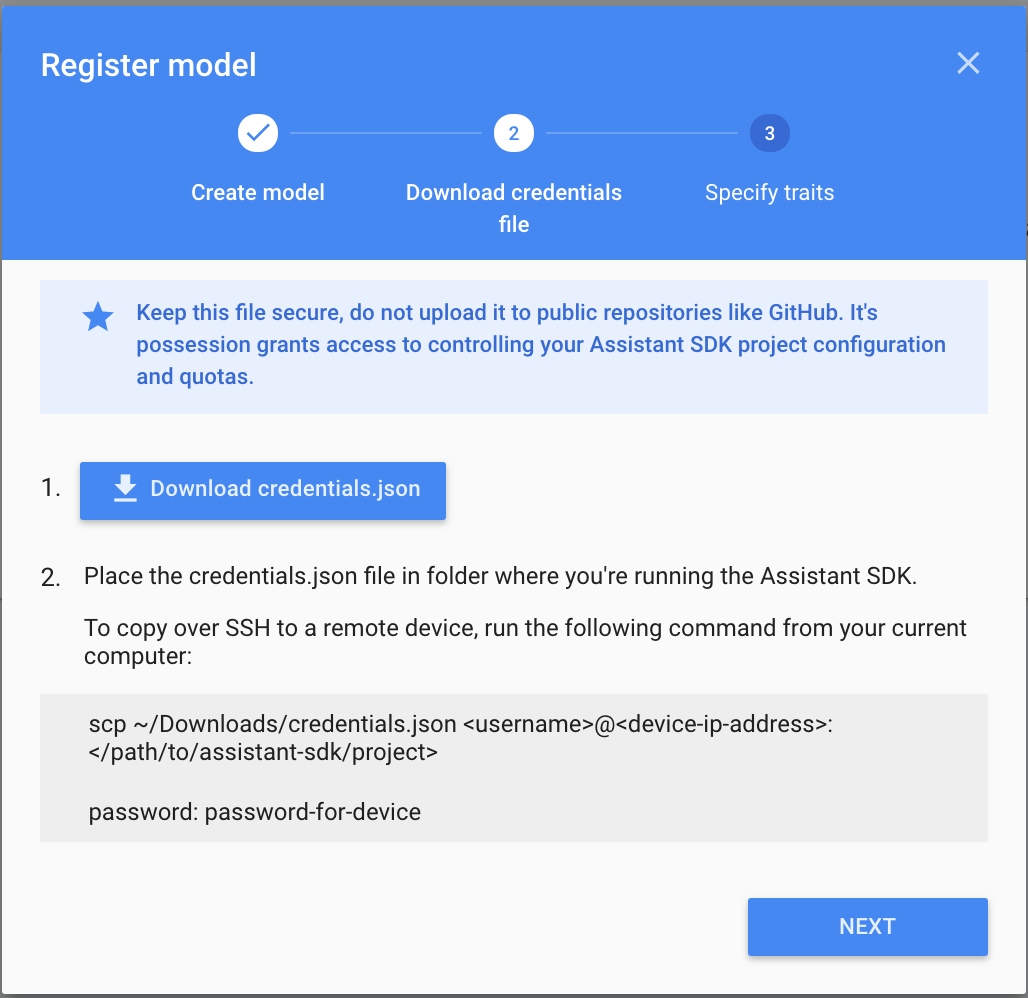


### Create Model

Fill out Product Name, Manufacturer name and select Device type. Click REGISTER MODEL.



### Download credential file and transfer it to your ARTIK530s



Copy credentials.json over to your ARTIK /home directory. For Windows users, you can use a SCP client like WinSCP or FileZilla. For Mac users, you can simply run scp command from your host machine. Replace ARTIK-IP-ADDRESS by using the IP address of your ARTIK board.

#scp credentials.json root@ARTIK-IP-ADDRESS:/home

### Specify traits

Enable all 7 traits and SAVE TRAITS.



## 5.4 Install the SDK and Sample Code

Install Google Assistant SDK

Install a Python virtual environment to isolate the SDK and its dependencies from the system Python.

[root@artik home]#apt-get install python-dev python-virtualenv virtualenv build-essential

[root@artik home]#virtualenv env --no-site-packages

[root@artik home]#env/bin/python -m pip install --upgrade pip setuptools wheel

[root@artik home]#source env/bin/activate

(env) [root@artik home]#

Install package dependencies and the latest version of Python package in the virtual environment.

(env) [root@artik home]# sudo apt-get install portaudio19-dev libffi-dev libssl-dev

(env) [root@artik home]# python -m pip install --upgrade google-assistant-library

(env)[root@artik home]# python -m pip install --upgrade google-assistant-sdk[samples]

### Generate credentials

(env) [root@artik home]# python -m pip install --upgrade google-auth-oauthlib[tool]

(env) [root@artik home]# google-oauthlib-tool --scope https://www.googleapis.com/auth/assistant-sdk-prototype --save --headless --client-secrets /home/client\_secret\_*client-id*.json

Please visit this URL to authorize this application: https://accounts.google.com/o/oauth2/auth?response\_type=code&client\_id=..

Enter the authorization code: 4/Md1uiKkfkqTKmb0xRY-O\_3Ade8X6S5pIAdjMTk8ZHyY

credentials saved: /root/.config/google-oauthlib-tool/credentials.json

You should see a URL displayed. Open the URL in a browser, then copy and past the authorization code to your terminal. If authorization was successful, you will see a response saying credentials saved.

Enter the authorization code: 4/Md1uiKkfkqTKmb0xRY-O\_3Ade8X6S5pIAdjMTk8ZHyY

credentials saved: /root/.config/google-oauthlib-tool/credentials.json

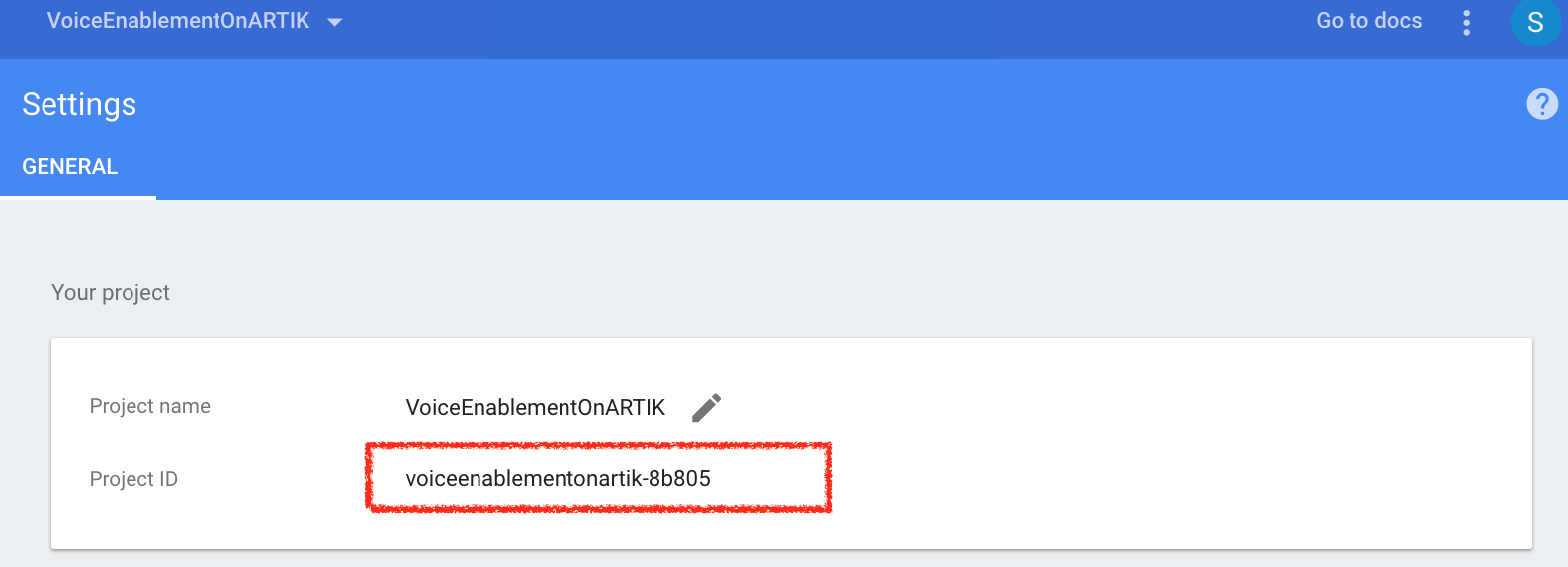
## 5.5 Run the sample app

### Project ID

From Google Actions Console, Overview -> Project settings

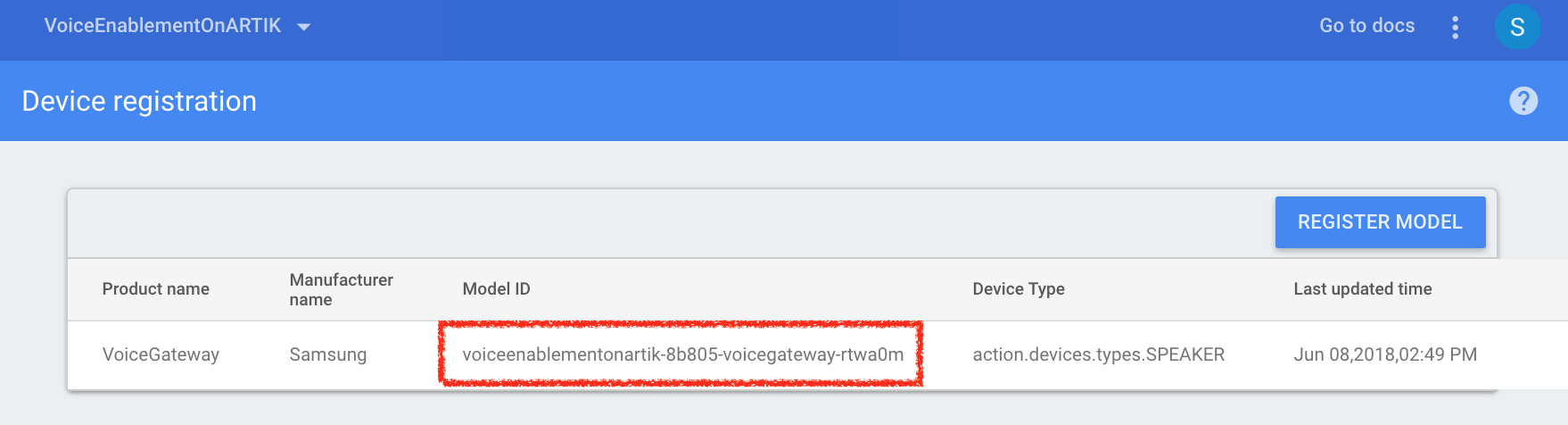


find your Project ID and take a note it.



### Device Model ID

From Actions Console, Device registration(Under “ADVANCED OPTIONS”), find out your model-id



### Run the Sample App

(env) [root@artik home]# googlesamples-assistant-pushtotalk --project-id YOUR\_PROJECT\_ID --device-model-id YOUR\_DEVICE\_MODEL\_ID

INFO:root:Connecting to embeddedassistant.googleapis.com

Press Enter to send a new request...

INFO:root:Recording audio request.

INFO:root:Transcript of user request: "what".

INFO:root:Transcript of user request: "what's".

INFO:root:Transcript of user request: "what's the".

INFO:root:Transcript of user request: "what's the weather".

INFO:root:Transcript of user request: "what's the weather".

INFO:root:Transcript of user request: "what's the weather".

INFO:root:Transcript of user request: "what's the weather in".

INFO:root:Transcript of user request: "what's the weather in".

INFO:root:Transcript of user request: "what's the weather in Key".

INFO:root:Transcript of user request: "what's the weather in Chico".

INFO:root:Transcript of user request: "what's the weather in Chicago".

INFO:root:Transcript of user request: "what's the weather in Chicago".

INFO:root:Transcript of user request: "what's the weather in Chicago".

INFO:root:End of audio request detected.

INFO:root:Stopping recording.

INFO:root:Transcript of user request: "what's the weather in Chicago".

INFO:root:Playing assistant response.

INFO:root:Finished playing assistant response.

## 5.6 Define your own actions

1. Clone Google Assistantt SDK

(env)$ git clone https://github.com/googlesamples/assistant-sdk-python

2. From your host machine, scp the downloaded actions.js and gactions command line tool to ARTIK530s, /home directory

For Windows users, you can use a SCP client like WinSCP or FileZilla. For Mac users, you can simply run scp command from your host machine. Replace ARTIK-IP-ADDRESS by using the IP address of your ARTIK board.

#scp actions.js gactions [root@ARTIK-IP-ADDRESS:~/assistant-sdk-python/google-assistant-sdk/actions.json](mailto:root@ARTIK-IP-ADDRESS:~/assistant-sdk-python/google-assistant-sdk/actions.json) .

3. Remove any existing credentials from the same directory as the gactions tool.

(env)$rm creds.data

Save your Action Package to Google by using the gactions CLI. Replace ***project\_id*** with your Actions Console project [ID](https://support.google.com/cloud/answer/6158840).

./gactions update --action\_package actions.json --project ***project\_id***

4. Bonus QueThe first time you run this command you will be given a URL and be asked to sign in. Copy the URL and paste it into a browser (this can be done on any machine). The page will ask you to sign in to your Google account. Sign into the Google account that created the project in a previous [step](https://developers.google.com/assistant/sdk/guides/library/python/embed/config-dev-project-and-account).

5. After you approve the permission request from the API, a code will appear in your browser, such as "4/XXXX". Copy and paste this code into the terminal:

Enter the authorization code:

If authorization was successful, you will see a response similar to the following:

Your app for the Assistant for project my-devices-project was successfully updated with your actions.

6. Deploy your action package into test mode by using the gactions CLI. You must have saved your Action Package to Google at least once before running this command. Test mode enables the action package on your user account only.

./gactions test --action\_package actions.json --project ***project\_id***

Now, run sample app again with questions like “What is Samsung ARTIK?”

## Bonus Question:

We enabled all 7 traits on the Google Assistant device model created, as the next step, please use these traits to control peripherals.

# Lab 6: MQTT

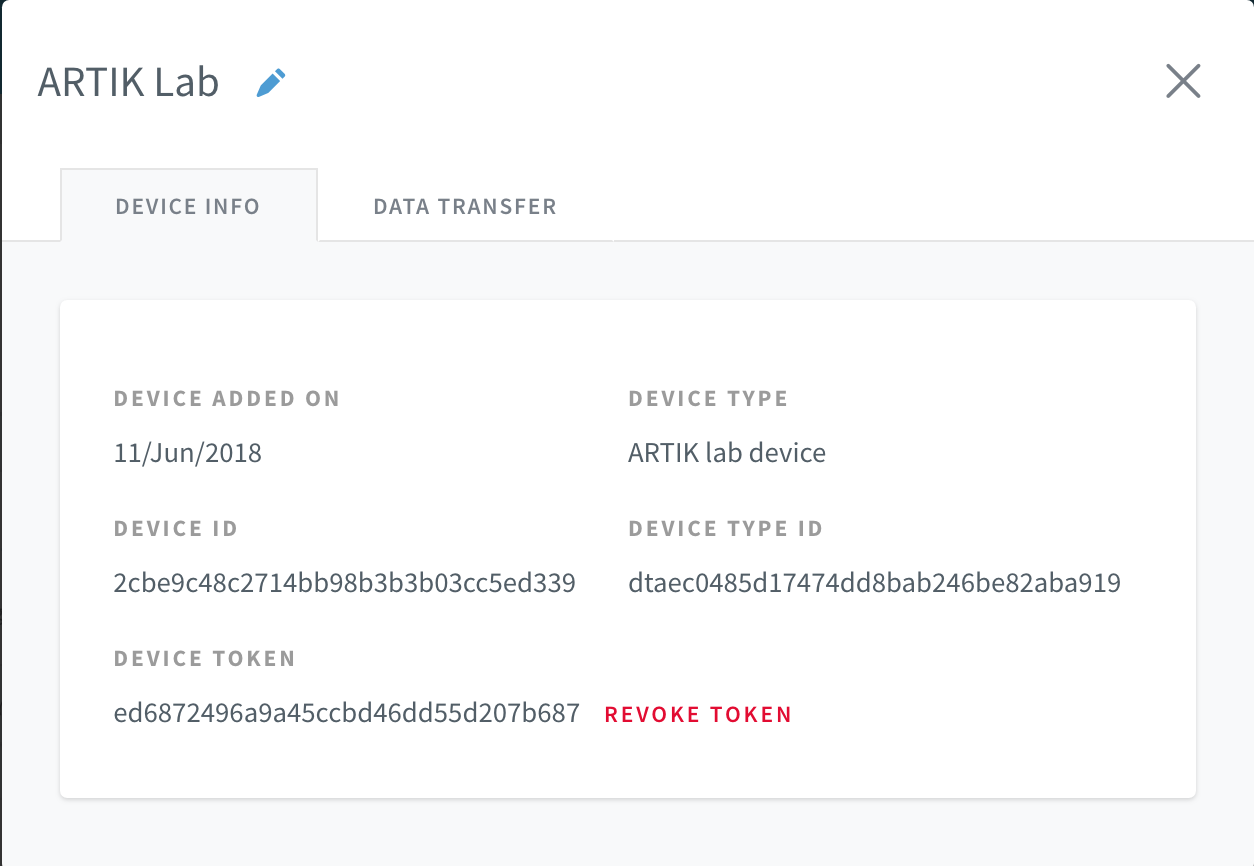
## Listen for MQTT message on ARTIK 530s

From your host machine, scp the downloaded mqtt.js to ARTIK530s, /home directory

For Windows users, you can use a SCP client like WinSCP or FileZilla. For Mac users, you can simply run scp command from your host machine. Replace ARTIK-IP-ADDRESS by using the IP address of your ARTIK board.

#scp mqtt.js root@ARTIK-IP-ADDRESS:/home

In ARTIK Cloud user portal, click on the device created during onboarding, and retrieve device id and device token.



On ARTIK530s, MQTT broker mosquito should have been auto-launched.

Run mqtt.js with your own DEVICE\_ID and DEVICE\_TOKEN.

[root@artik ~]# cd /home/

[root@artik ~]# nodejs mqtt.js YOUR\_DEVICE\_INSTANCE\_DEVICE\_ID

YOUR\_DEVICE\_INSTANCE\_DEVICE\_TOKEN

Device ID:7a1fb71xxxxxxxxxxxxxxxxx

Device Token: 72c660c6363xxxxxxxxxxx34908

Registered

Subscribed to MQTT broker

Now, Let’s move to ARTIK055s.

## Connect to ARTIK 055s

1. Connect the IR distance sensor to your ARTIK 055s as shown below.

Yellow wire: A0

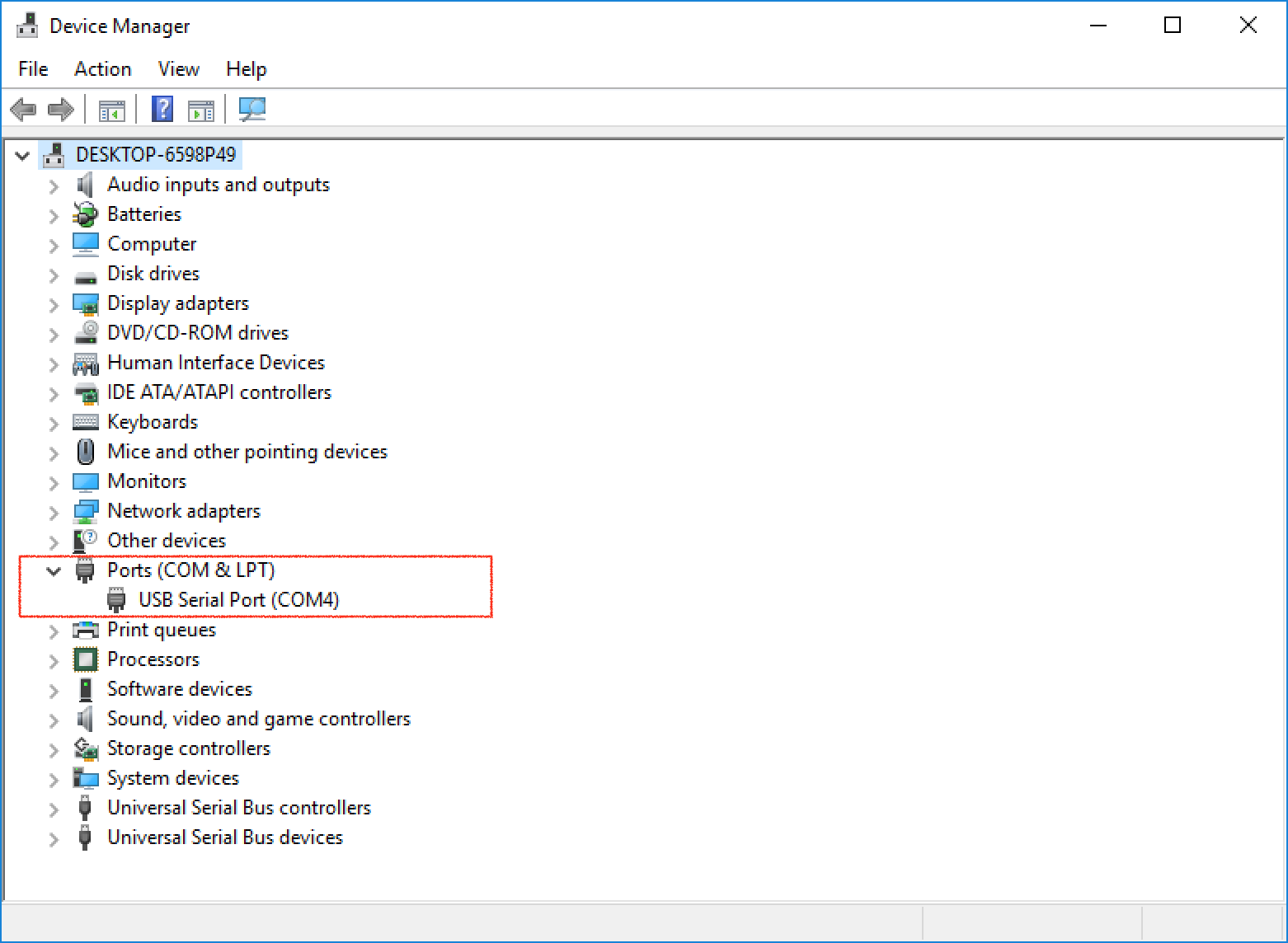
Black wire: GND

Red wire: 3.3V

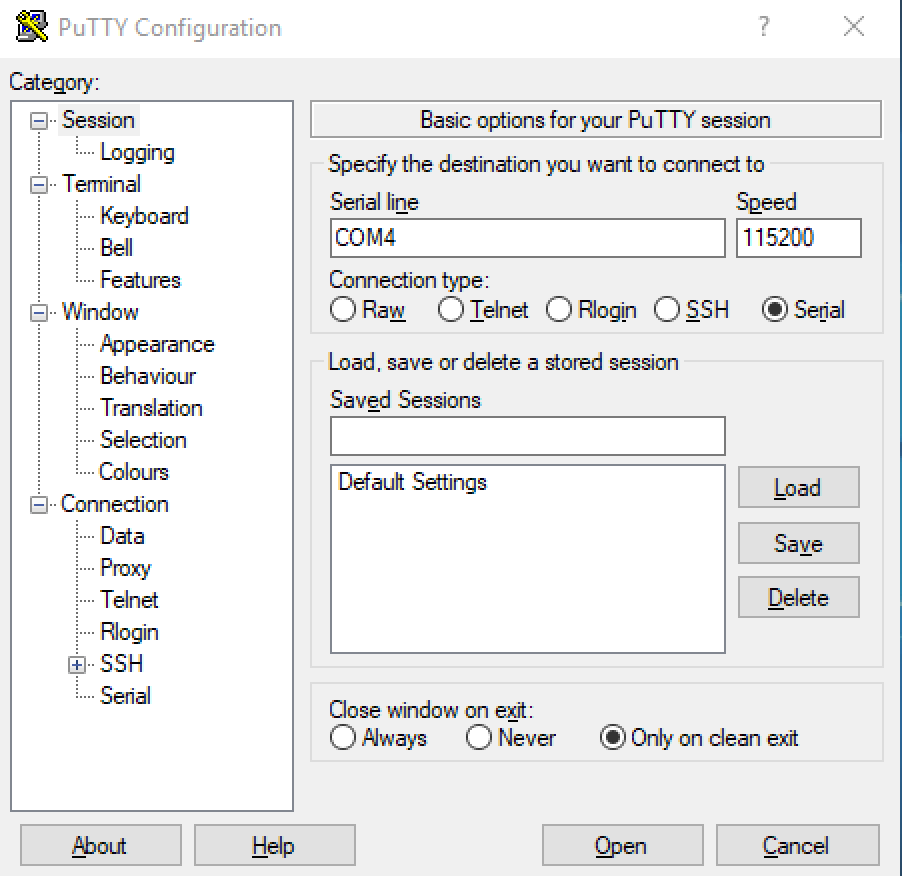


2. Connect ARTIK 055s development board to your host machine using an USB cable.

1. Find your serial device in your Device Manager / Ports and remember the COM port your ARTIK board is using. In our example, the board uses COM4 port.



1. On Windows host machine, please install and launch a serial console application PuTTY.. Select “Serial” as the Connection type, use the COM port number you find from Device Manager and Speed 115200 for serial access.



5. For Mac users, you can use the built-in *screen* utility to access the serial console. Normally the device name ends with 141B or 142B.

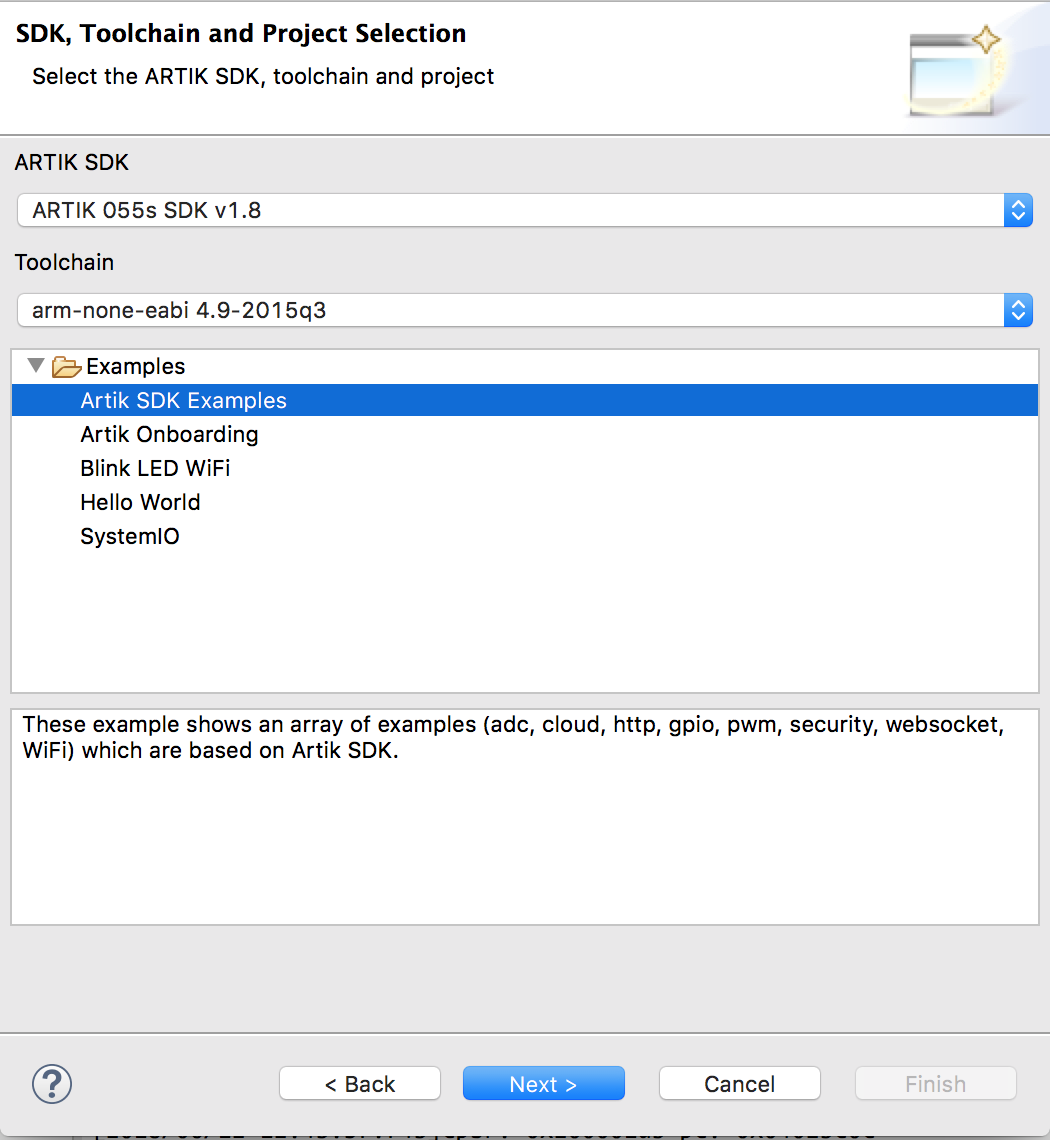
screen /dev/cu.usbserial-XXXXXXX 115200

## Build an Application for ARTIK 055s

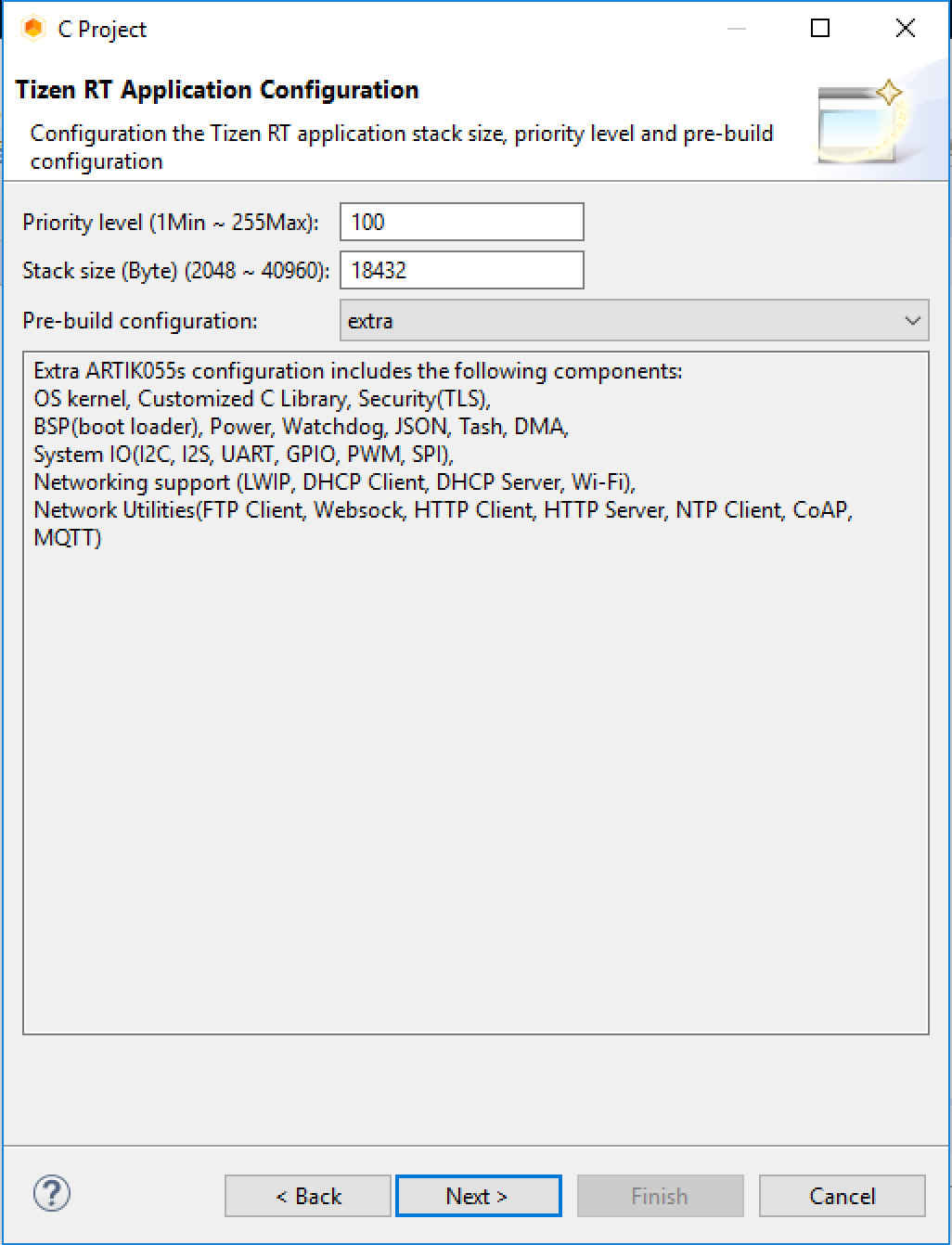
1. Launch your ARTIK IDE, go to File->New->C Project.
2. In the “C Project” dialog, select “ARTIK 055s C project”, and choose “gcc-arm-none-eabi” as the default toolchain. Name the Project name as “artik055s”. Then, click Next.



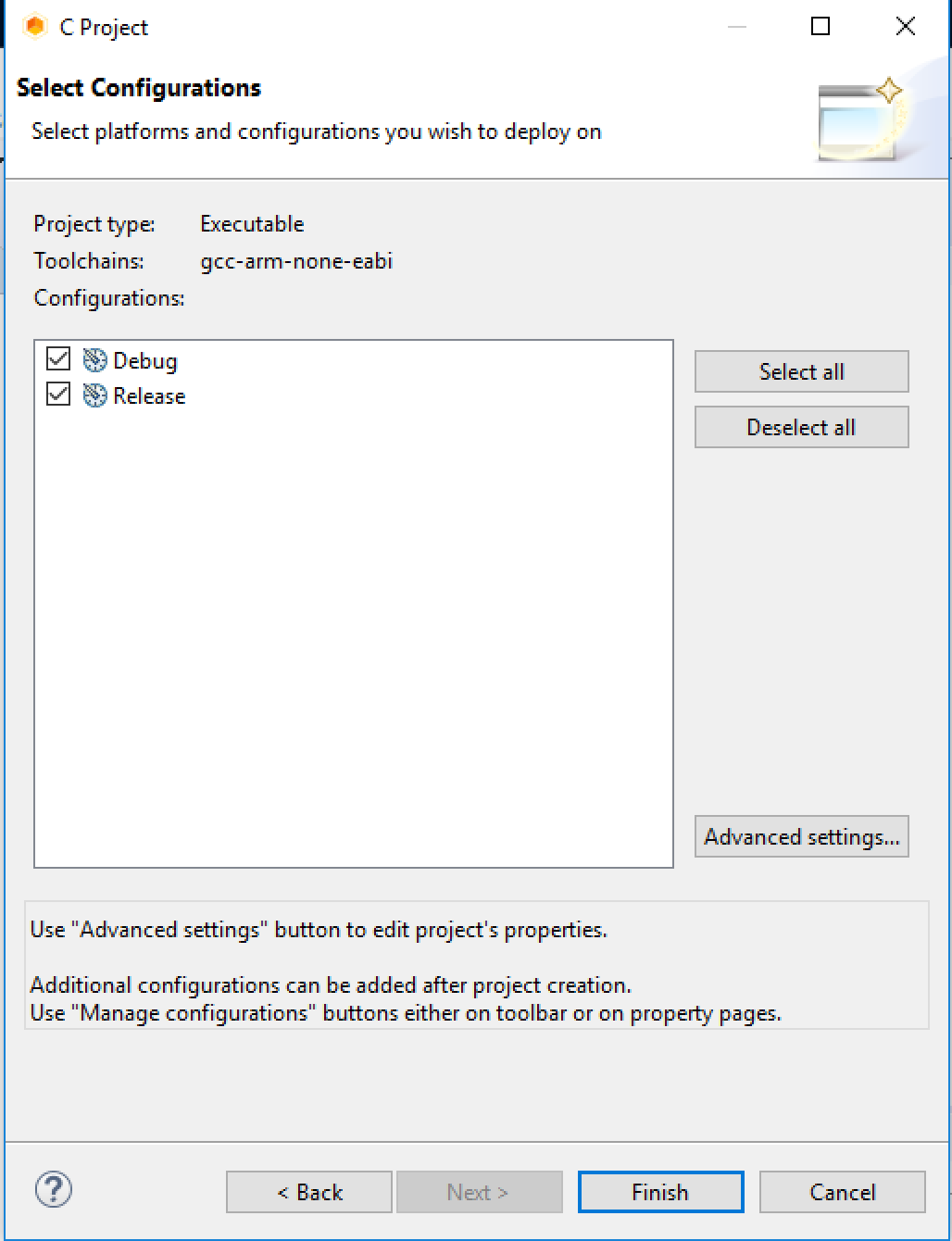
1. In “SDK, Toolchain and Project Selection” dialog, select “ARTIK 055s SDK v1.8(or 1.7.1)” as the target SDK version. Click on “Artik SDK Example” example, we will use this example as our template to generate an ARTIK055s application. Then, click Next.



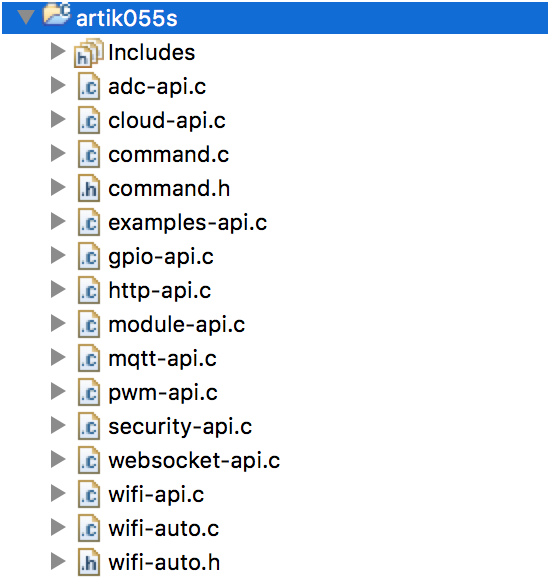
1. In “Tizen RT Application Configuration” dialog, use the default Priority level and Stack Size settings. Change the Pre-build configuration to “extra”, then click Next.



1. In “Select Configurations” dialog, keep the default Debug and Release builds settings, and click Finish.

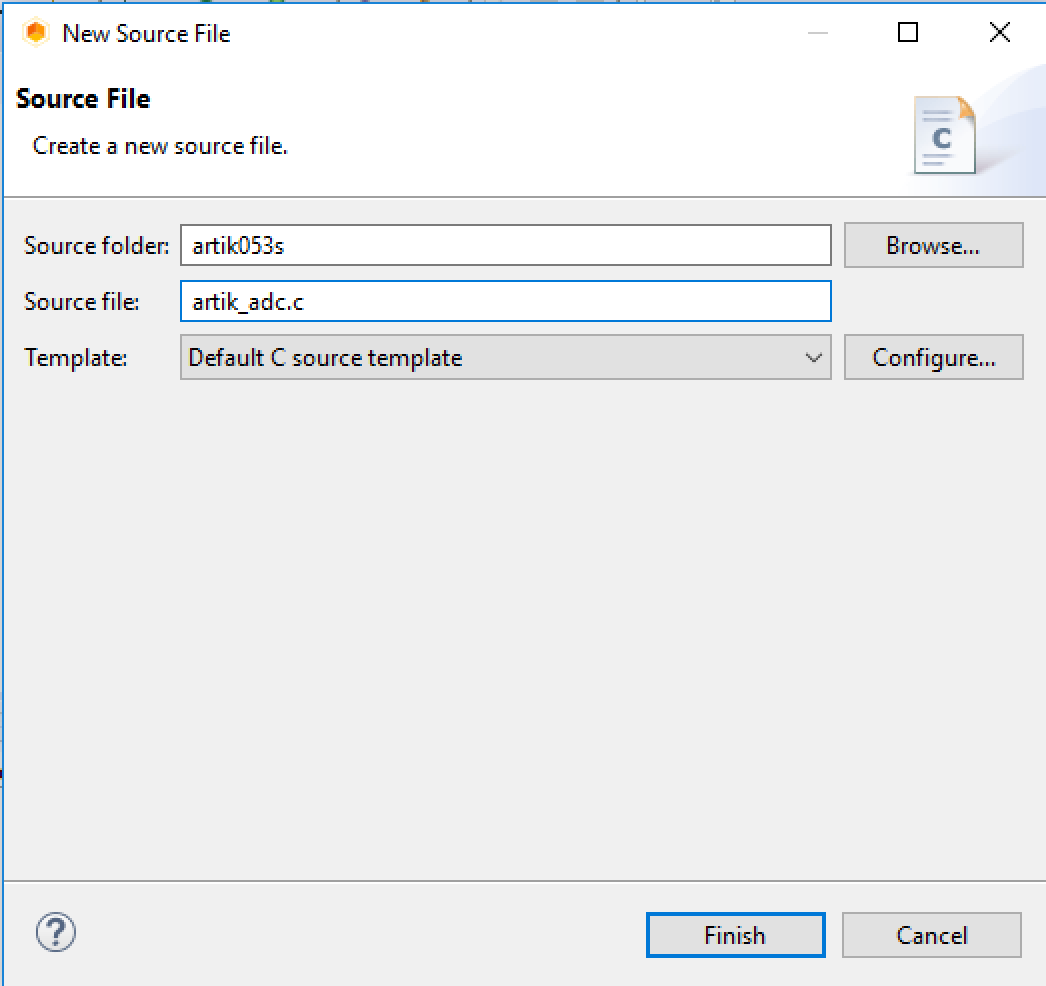


1. Now you have an application artik055s created in your IDE Project Explorer.



## Publish MQTT message from ARTIK055s

1. From your ARTIK IDE, go to File->New->Source File, enter artik\_adc.c as the source file name, and click “Finish”.



2. Here is what you need to include in your artik\_adc.c. In this file, we only have one function defined, which reads distance sensor data from the analog pin A0.

**#include** <stdio.h>

**#include** <string.h>

**#include** <artik\_module.h>

**#include** <artik\_error.h>

**#include** <artik\_adc.h>

**#include** "artik\_onboarding.h"

**int** ReadSensor(**int** pin)

{

artik\_adc\_module \*adc = NULL;

artik\_adc\_handle handle = NULL;

artik\_adc\_config config;

artik\_error err = S\_OK;

**int** val = 0;

**char** name[16] = "";

**int** ret = 0;

adc = (artik\_adc\_module \*)**artik\_request\_api\_module**("adc");

**if** (!adc) {

**fprintf**(stderr, "ADC module is not available\n");

**return** -1;

}

**memset**(&config, 0, **sizeof**(config));

config.pin\_num = pin;

**snprintf**(name, 16, "adc%d", config.pin\_num);

config.name = name;

err = adc->request(&handle, &config);

**if** (err != S\_OK) {

**fprintf**(stderr, "Failed to request ADC %d (%d)\n", config.pin\_num, err);

ret = -1;

**goto** exit;

}

err = adc->get\_value(handle, &val);

**if** (err != S\_OK) {

**fprintf**(stderr, "Failed to read ADC %d value (%d)\n", config.pin\_num, err);

adc->release(handle);

ret = -1;

**goto** exit;

}

fprintf(stdout, "ADC%d=%d\n", config.pin\_num, val);

adc->release(handle);

exit:

**artik\_release\_api\_module**(adc);

**return** val;

}

3. Open wifi-auto.c, in this file, we will define the SSID and PASSWORD of the WiFi access point we want to connect to. Of course, you can configure this from command line as well.

**#define** WIFI\_SSID "ARTIK2"

**#define** WIFI\_PASSPHRASE "artik123"

4. In mqtt-api.c, look for mqtt\_cmd\_connect, function. Add StartWifiConnection() to auto connect the WiFi we configured.

port = atoi(argv[4]);

StartWifiConnection();

ret = g\_mqtt->connect(g\_handle, host, port);

**if** (ret != S\_OK) {

…

Then replace mqtt\_publish function with the code below.

**int** **mqtt\_cmd\_publish**(**int** argc, **char** \*\*argv)

{

artik\_error ret = S\_OK;

**int** i=0;

**struct** timespec timestamp;

**if** (!g\_handle) {

**fprintf**(stderr, "MQTT connection is not active\n");

**return** -1;

}

**for** (i=0; i<10;i++) {

**int** val = ReadSensor(0);

**char** adc\_data[4];

**char** message[100] = "";

**char**\* topic = "ARTIKTraining";

**sprintf**(adc\_data,"%d",val);

**printf**("MQTT: publish %d on %s\n", val, topic);

ret = g\_mqtt->publish(g\_handle, 0, false, topic, **strlen**(adc\_data), adc\_data);

**if** (ret != S\_OK) {

**fprintf**(stderr, "Failed to publish message\n");

**return** -1;

}

**usleep**(10000 \* 1000);

}

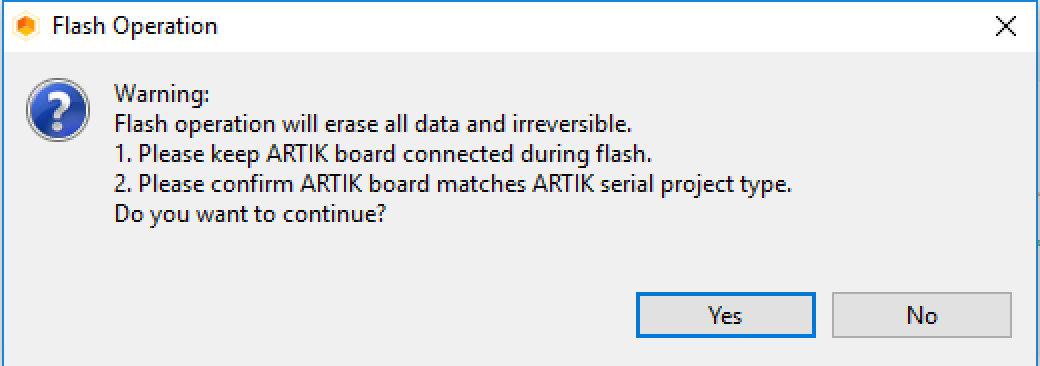
**return** 0;

}

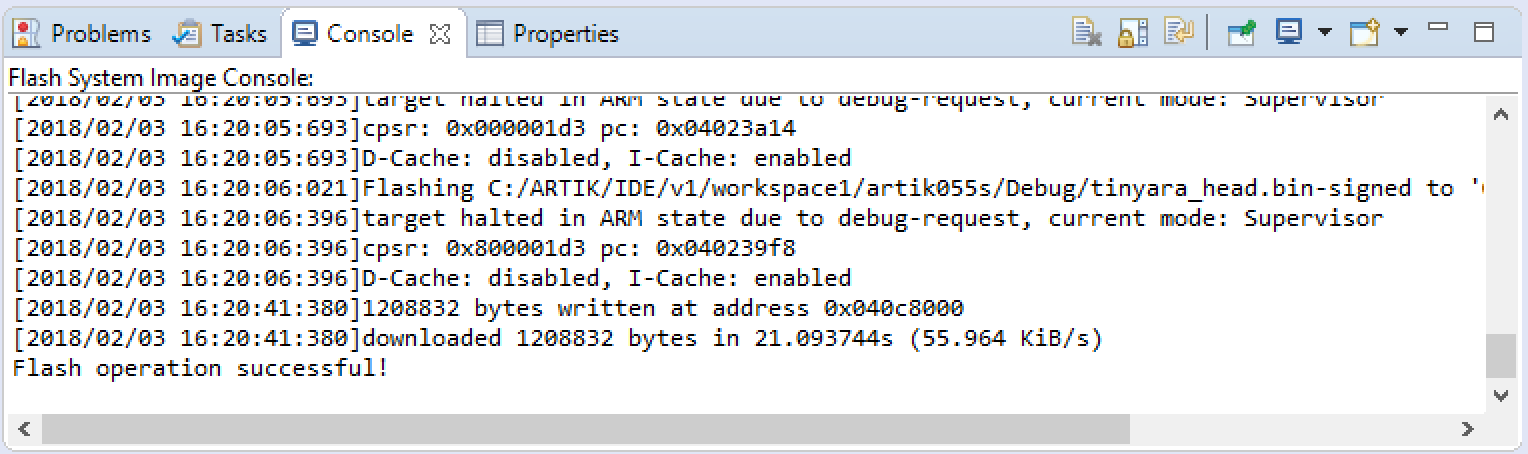
5. Build the application by selecting Project->Build Project. You can see the build log from the IDE Console tab.

6. Once you have a successful build, connect your ARTIK 055s board to your laptop by using an USB cable, click the “Flash System Image” button to flash the newly built image to the board.

As a precaution, a warning dialog will pop up to confirm you want to flash the device and make sure your ARTIK board matches ARTIK serial project type. Click “Yes” to proceed.



6. From your IDE console, you can view the flash progress.



## Publish MQTT message from ARTIK055s

1. Push the Reset button to reboot 055s.



2. From your serial console, you should be able to see the message below when the board boots up.

U-Boot 2017.01-00013-g814fa7d (Jan 08 2018 - 15:17:26 +0900)

CPU: Exynos200 @ 320 MHz

Model: ARTIK-05x based on Exynos T20

DRAM: 1.3 MiB

WARNING: Caches not enabled

BL1 released at 2017-3-13 15:00

SSS released at 2017-09-12

WLAN released at 2017-12-21

Flash: 8 MiB

\*\*\* Warning - bad CRC, using default environment

In: serial@80180000

Out: serial@80180000

Err: serial@80180000

Autoboot in 50 milliseconds

gpio: pin gpg16 (gpio 46) value is 1

## Starting application at 0x040C8020 ...

s5j\_sflash\_init: FLASH Quad Enabled

i2c\_uioregister: Registering /dev/i2c-0

i2c\_uioregister: Registering /dev/i2c-1

System Information:

Board: ARTIK055S

Version: 1.0.11

Commit Hash: 1872c22c0ec27fe39d8b00f1cba8f602ef143bbd

Build User: ARTIK@Samsung

Build Time: 2018-03-14 14:50:05

System Time: 01 Jan 2010, 00:00:00 [s] UTC Hardware RTC Support

TASH>>

3. Connect to MQTT broker

4. Use MQTT publish to publish data to the MQTT broker running on your ARTIK530s.

TASH>>mqtt connect mqtt://10.0.1.196 1883 NULL NULL

Starting supplicant in foreground...

Connected to ARTIK

Client mosq/T<OZR6Mm2;Q1SmdCUf sending CONNECT

TASH>>Client mosq/T<OZR6Mm2;Q1SmdCUf received CONNACK

MQTT connection result: OK

[root@artik ~]# cd /home/

[root@artik ~]# nodejs mqtt.js YOUR\_DEVICE\_INSTANCE\_DEVICE\_ID

YOUR\_DEVICE\_INSTANCE\_DEVICE\_TOKEN

Device ID:7a1fb71xxxxxxxxxxxxxxxxx

Device Token: 72c660c6363xxxxxxxxxxx34908

Registered

Subscribed to MQTT broker

publish sensor data 516 to ARTIK Cloud

publish sensor data 468 to ARTIK Cloud

publish sensor data 636 to ARTIK Cloud

publish sensor data 608 to ARTIK Cloud

publish sensor data 2543 to ARTIK Cloud

publish sensor data 777 to ARTIK Cloud

TASH>> mqtt publish

ADC0=1596

MQTT: publish 1596 on ARTIKTraining

Client mosq/\_bHDlMgXavuA@5\mN? sending PUBLISH (d0, q0, r0, m1, 'ARTIKTraining', ... (4 bytes))

MQTT message published: 1

ADC0=714

MQTT: publish 714 on ARTIKTraining

Client mosq/\_bHDlMgXavuA@5\mN? sending PUBLISH (d0, q0, r0, m2, 'ARTIKTraining', ... (3 bytes))

MQTT message published: 2

ADC0=1716

MQTT: publish 1716 on ARTIKTraining

Client mosq/\_bHDlMgXavuA@5\mN? sending PUBLISH (d0, q0, r0, m3, 'ARTIKTraining', ... (4 bytes))

MQTT message published: 3

ADC0=820

MQTT: publish 820 on ARTIKTraining

Client mosq/\_bHDlMgXavuA@5\mN? sending PUBLISH (d0, q0, r0, m4, 'ARTIKTraining', ... (3 bytes))

MQTT message published: 4

ADC0=1021

MQTT: publish 1021 on ARTIKTraining

Client mosq/\_bHDlMgXavuA@5\mN? sending PUBLISH (d0, q0, r0, m5, 'ARTIKTraining', ... (4 bytes))

MQTT message published: 5

On ARTIK530s console, you should be able to see the log as below:

From ARTIK Cloud portal, Select ‘sensor’ option as below to enable data visualization.

