Cloud-based artificial intelligence had to be limited in various solution developments due to the limitations of constant network connectivity. LGE has developed the On-Device artificial intelligence chip, the LG8111, that can provide AI functions even when the network is disconnected.

The LG8111 supports hardware processing of artificial intelligence functions categorized as video, voice, and control intelligence. The LG8111 utilizes LG's unique AI processor to efficiently process the deep learning algorithm (low power, low latency) to achieve "On-Device" artificial intelligence, which escapes the various limitations of always-on network connectivity, which has been pointed out as difficult for artificial intelligence.

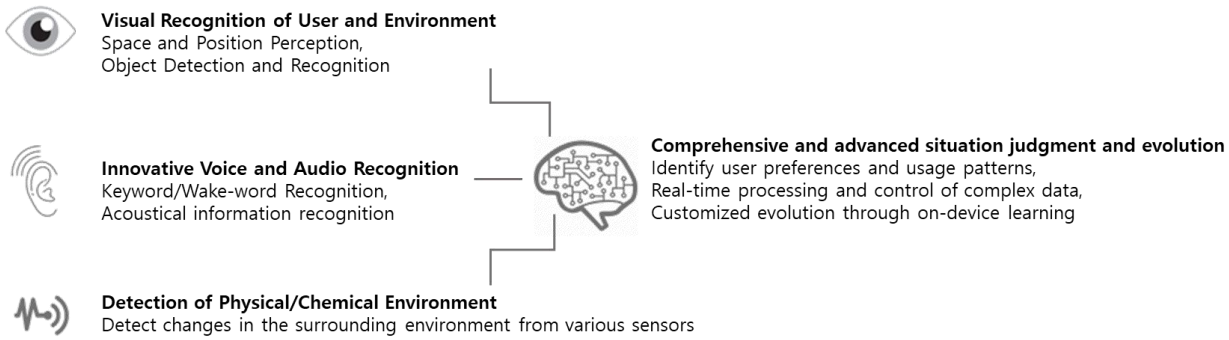


Figure 1. LG8111 Solution Highlights

The AWS IoT Greengrass extends AWS IoT to the device, taking advantage of the cloud, and providing an environment in which data generated through the device can be processed locally. With the On-Device Artificial Intelligence feature mentioned above, the LG8111 can provide improved Sensor Data collection/analysis and ML Inference performance. At the same time, the Eris board, the development board of the LG8111, allows you to integrate the AWS IoT Greengrass more easily for a variety of solutions.

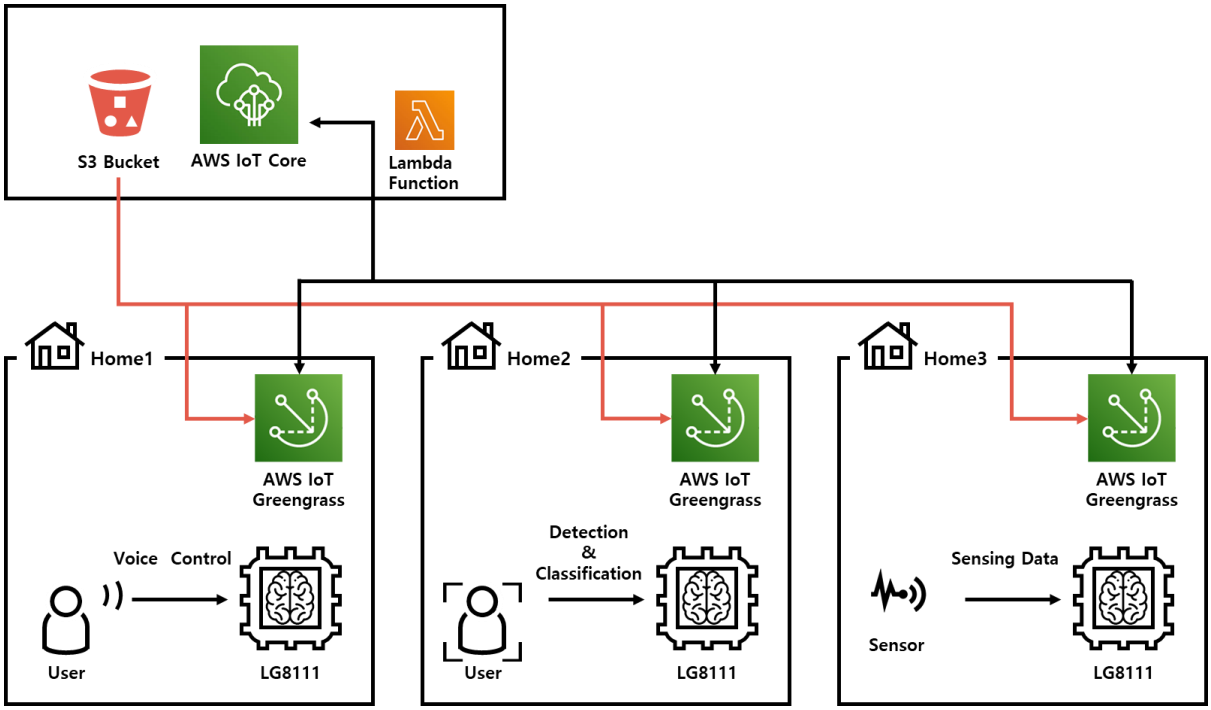


Figure 2. LG8111 and AWS IoT Greengrass

LG8111 Development Board

[LG8111 Development Board Configuration]

The LG8111 development board supports Ubuntu 18.04 environment and consists of Reset, USB Host, UART Debug Port, and Power.

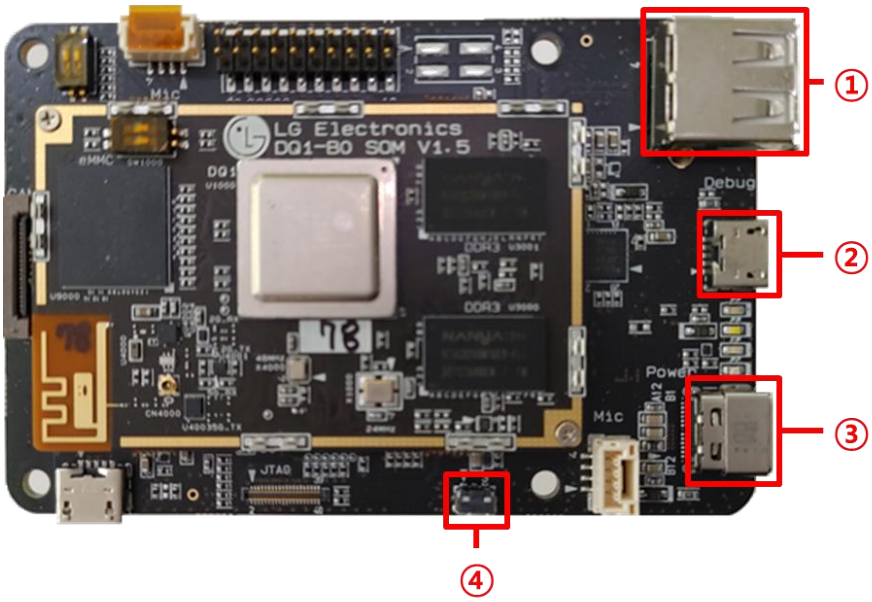


Figure 3. Reference Board(Eris) Embedding LG8111 AI Chip

NUM	REGION NAME	DESCRIPTION
1	USB Host	This is the port used to connect the USB Device.
2	UART Debug Port	Serial Port.
3	Power	Connect the USB Type-C to the Power and turn it on.
4	Reset	Reset the board.

[Board Connect]

Describes how to connect to the LG8111 development board.

1. Connect the Host (PC) and USB wire to the UART Debug Port on the LG8111 Development Board.

2. Grant power to the LG8111 Development Board Power.

The settings required to connect UART to the LG8111 development board are as follows:

RULES	VALUE
BAUD RATE	115200 bps
DATE BITS	8 bit
PARITY SET	N
STOP BITS	1

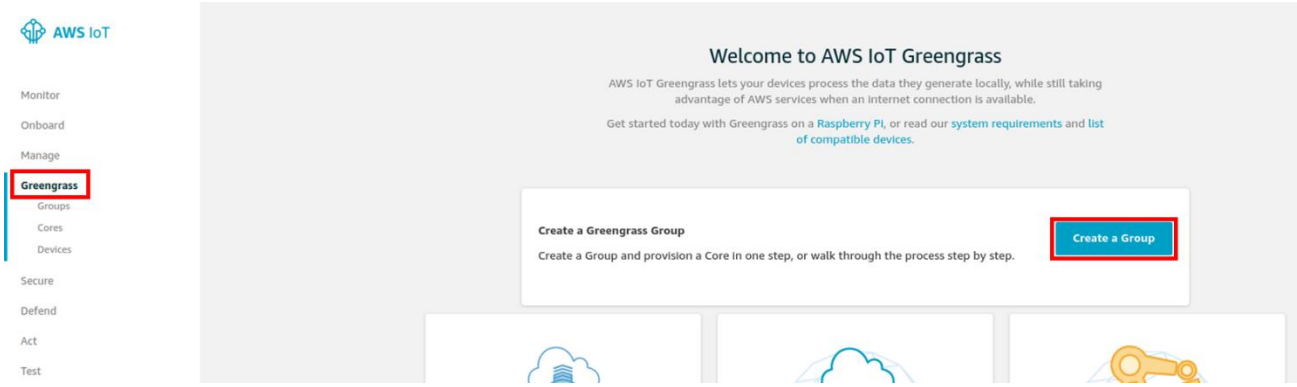
LG8111 uses AWS Greengrass on its development board

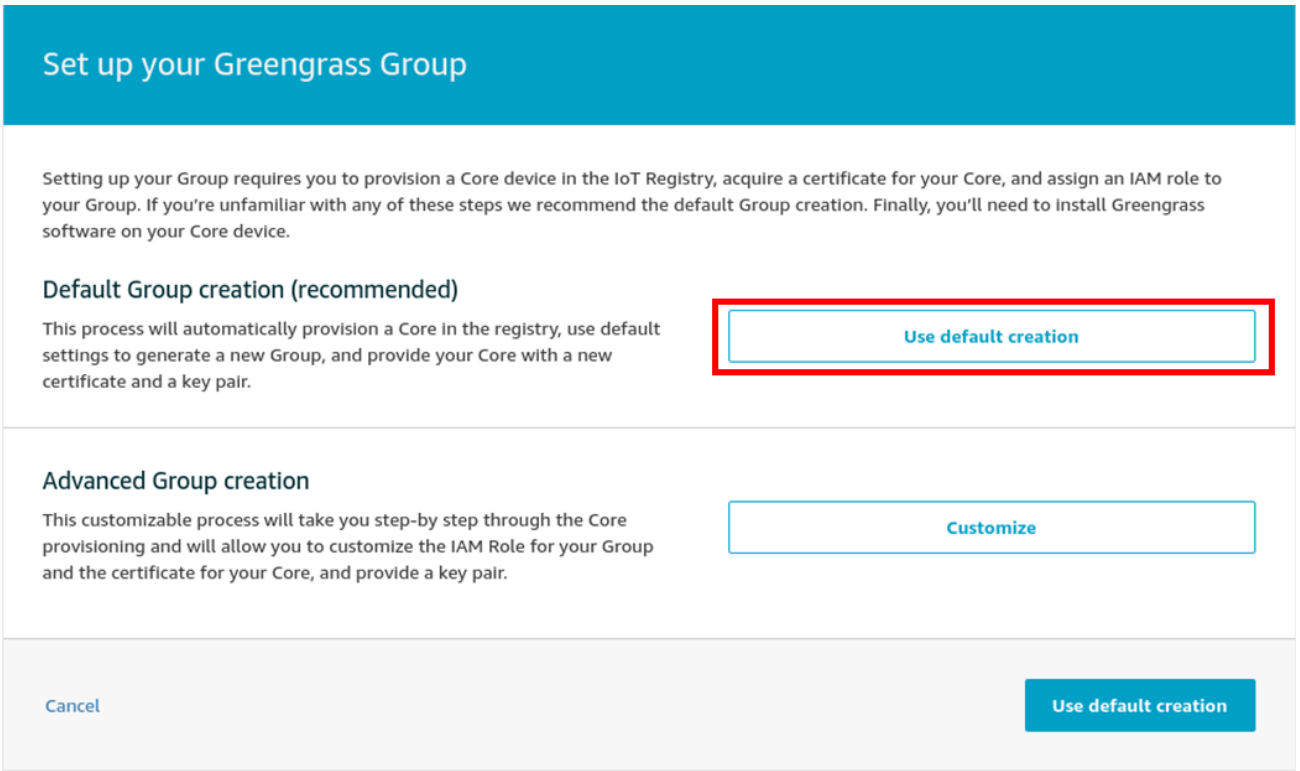
Follow the AWS IoT Greengrass Developer's Guide (https://docs.aws.amazon.com/ko_kr/greengrass/latest/developerguide/setup-filter.rpi.html) for configuring the AWS IoT Greengrass environment on the LG8111 before installing the Greengrass environment settings and the AWS IoT Greengrass core software.

[Obtaining AWS IoT Greengrass Certification]

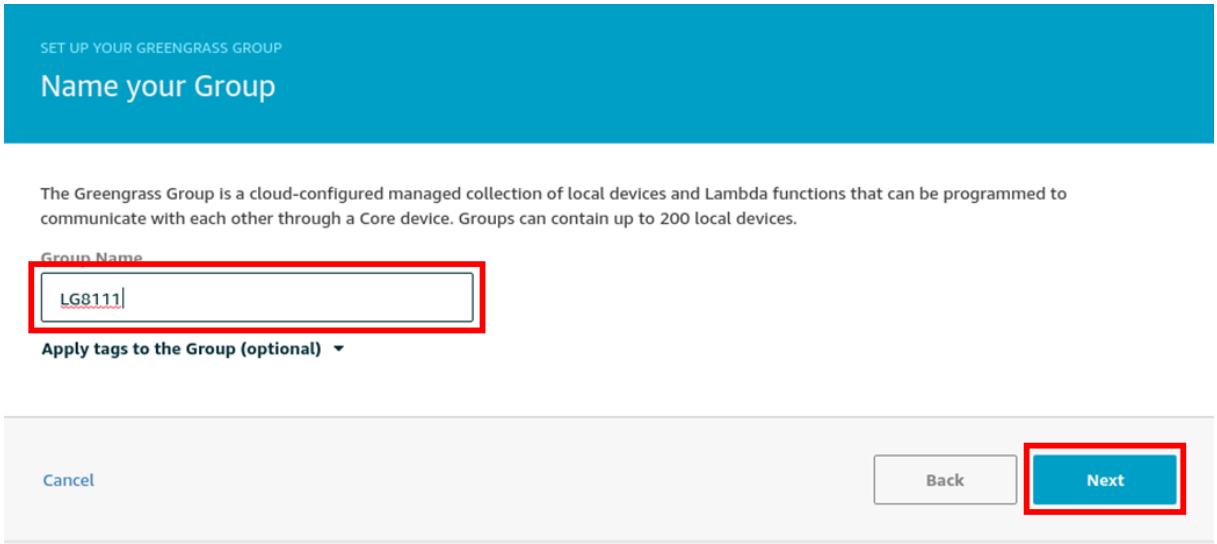
To run the AWS IoT Greengrass, you must obtain a unique certificate for each device. These unique certificates are separated by group in the AWS IoT Greengrass.

Create a group from the AWS IoT Greengrass start page to obtain a certificate.

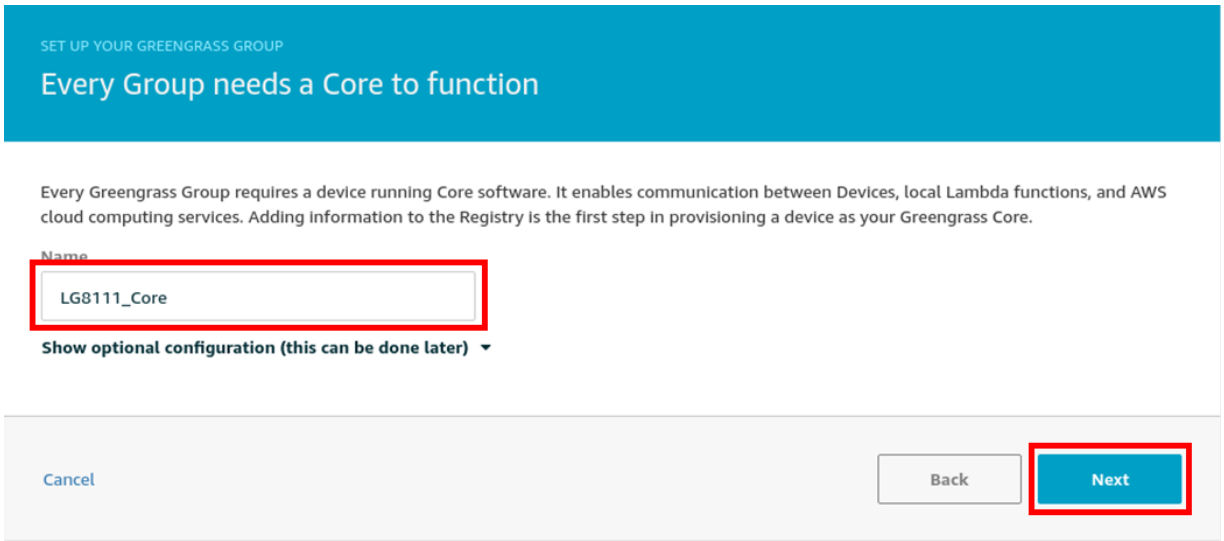




Enter the name of the group that you want to generate, then select:



Select the following using the basic configuration settings for the AWS IoT Greengrass group core name:



On the [Run a scripted easy group creation] page, select [Create Group and Core] to verify that the configuration is progressing without problems. If the settings are satisfactory, all settings are green and automatically proceed to the next page.

SET UP YOUR GREENGRASS GROUP

Review Group creation

In order to speed up and simplify Group creation AWS IoT Greengrass will handle the following processes and use default settings. By proceeding to the next step, you are giving permission for us to complete the following steps.

AWS IoT Greengrass will take these actions on your behalf using default settings:

Create a new Greengrass Group in the cloud	Learn more
Provision a new Core in the IoT Registry and add to the Group	Learn more
Generate public and private key set for your Core	Learn more
Generate a new security certificate for the Core using the keys	Learn more
Attach a default security policy to the certificate	Learn more
Enable stream manager on the Core device	Learn more

Cancel

Create Group and Core

SET UP YOUR GREENGRASS GROUP

Review Group creation

In order to speed up and simplify Group creation AWS IoT Greengrass will handle the following processes and use default settings. By proceeding to the next step, you are giving permission for us to complete the following steps.

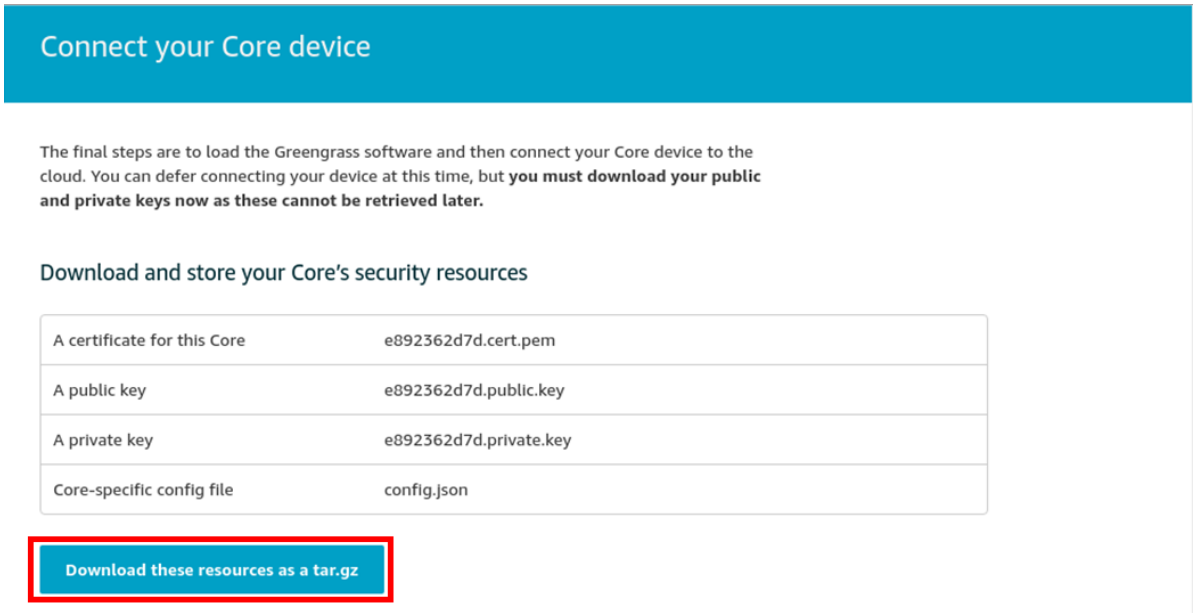
AWS IoT Greengrass will take these actions on your behalf using default settings:

✓ Create a new Greengrass Group in the cloud	Learn more
✓ Provision a new Core in the IoT Registry and add to the Group	Learn more
✓ Generate public and private key set for your Core	Learn more
✓ Generate a new security certificate for the Core using the keys	Learn more
✓ Attach a default security policy to the certificate	Learn more
✓ Enable stream manager on the Core device	Learn more

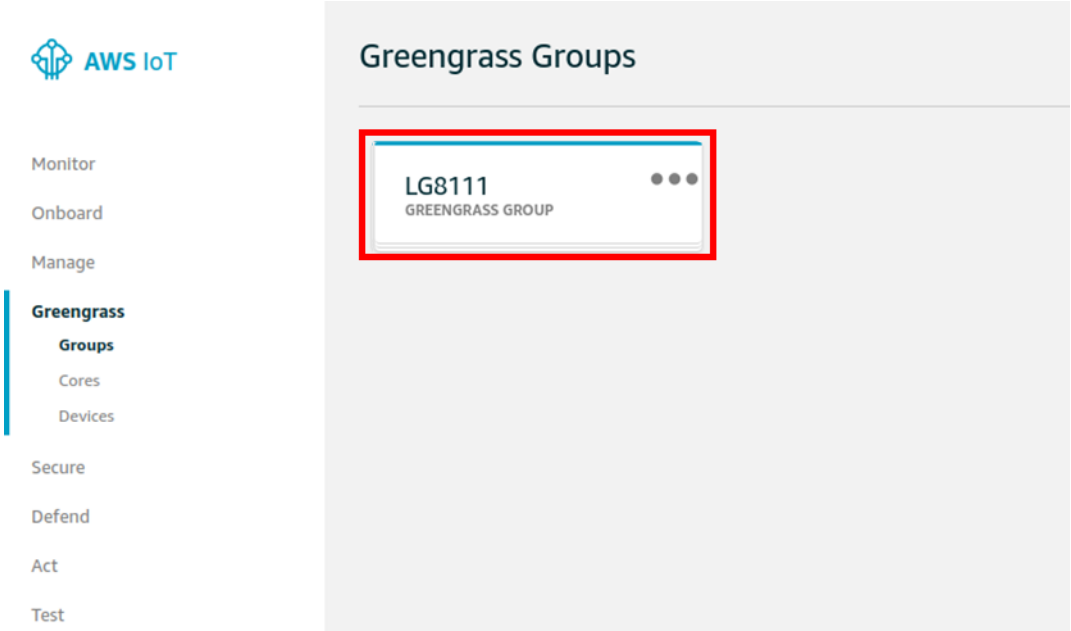
Cancel

Create Group and Core

Upon successful generation of the AWS IoT Greengrass group, you will receive core resources. You will never be able to download the downloaded core resource file again, so you must save and manage it. If you do, you will need to regenerate the core resources.



If the AWS IoT Greengrass group is successfully generated, the AWS IoT Greengrass start page displays the generated group in the group window.



Run AWS IoT Greengrass on the LG8111 Development Board

This is an example of inference of Camera Input data using local resources (Camera, LG AI processor) of the LG8111 development board, and sending the result of the inference to AWS Greengrass to check the result.

You will need to download the AWS IoT Greengrass SDK for python provided by the AWS IoT Greengrass from the Developer's Guide page (https://docs.aws.amazon.com/ko_kr/greengrass/latest/developerguide/what-is-gg.html#gg-core-sdk-download) for example execution.

This example was implemented in python2.7. Every 5 seconds, images captured by the camera are inferred using an artificial intelligence algorithm called MobileNet, and the inferred results are sent to AWS IoT Greengrass.

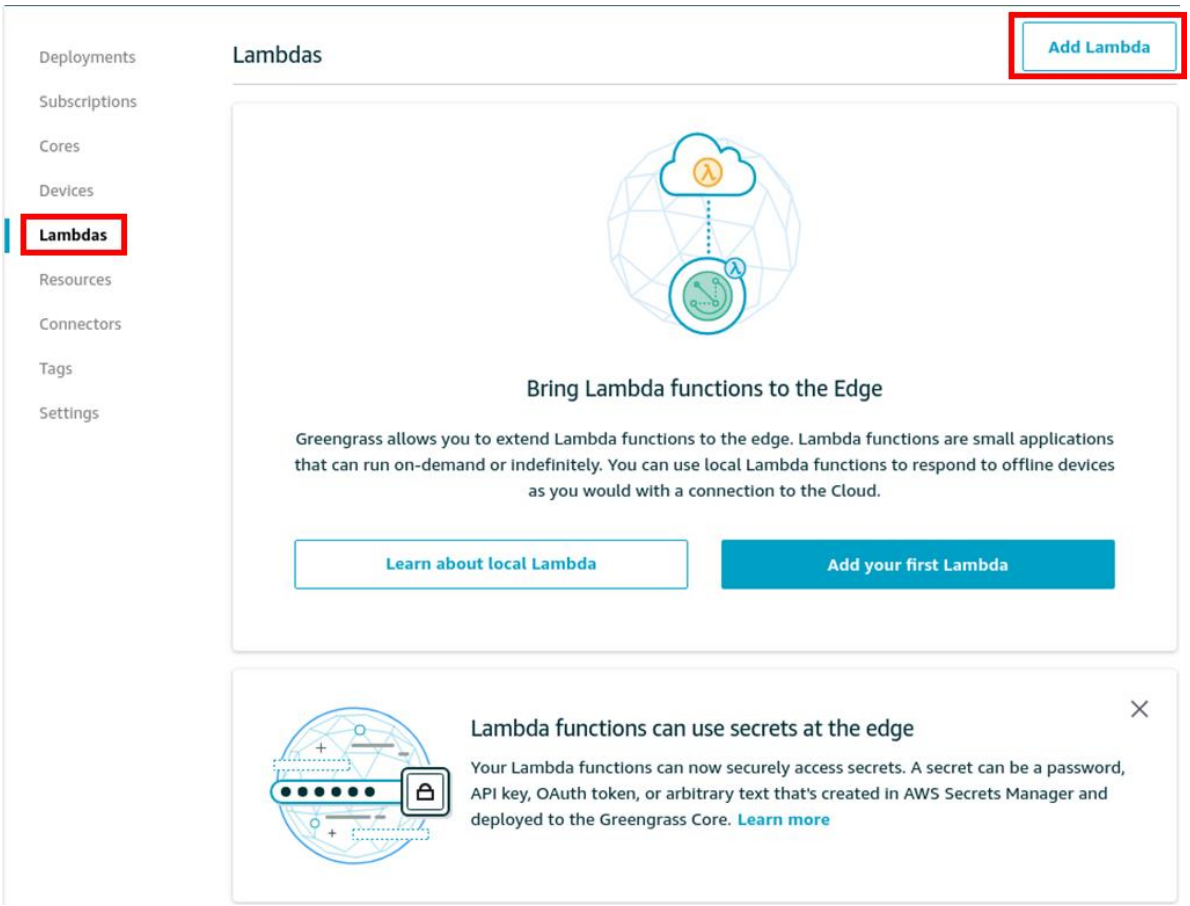
Simply route the contents in the device's "/home/ubuntu/sample/getting_started" folder and the AWS IoT Greengrass SDK for python and compress the same file.

```
$ zip -r greengrassMobilenet.zip labels.txt Mobilenet.lne greengrassMobilenet.py greengrasssdk
```

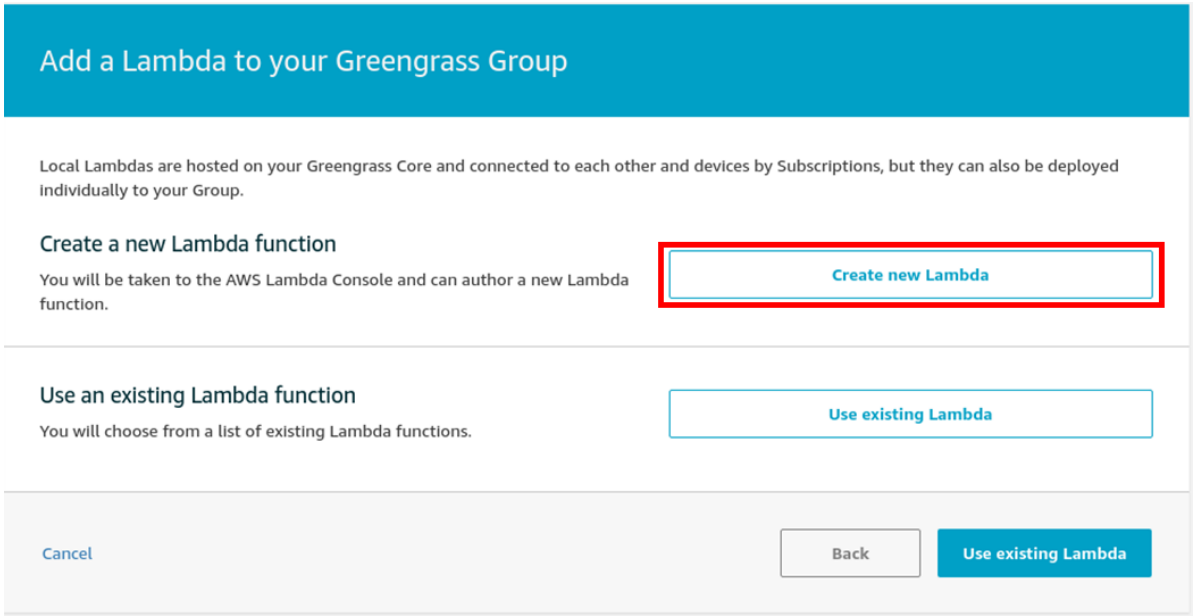
1. Lambda Registration

Connect with LG8111 development board group registered with AWS IoT Greengrass group.

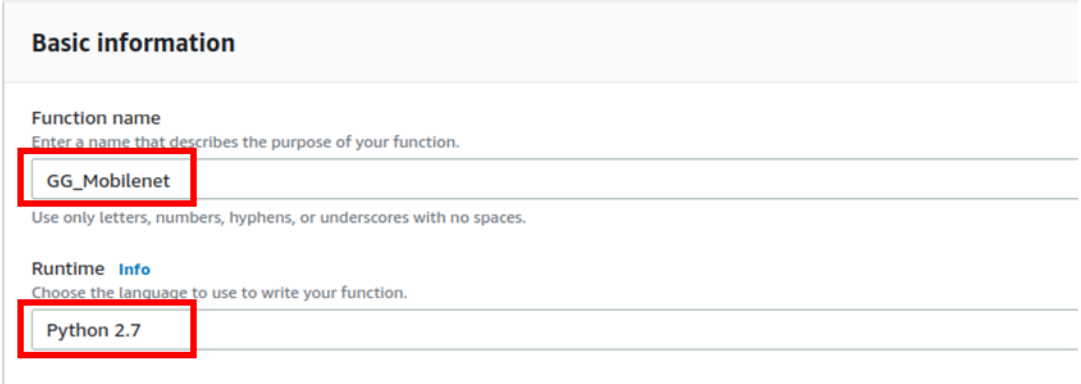
Register a new Lambda by selecting Lambda -> Add Lambda for the connected group.



Select [Create new Lambda] to add a Lambda function to test the AWS IoT Greengrass on the development board of LG8111.



The Function name is an example of a Python 2.7, registered with a Lambda-distinctive name and created based on python 2.7, so select the Runtime option to python 2.7, then move on to the next stage.

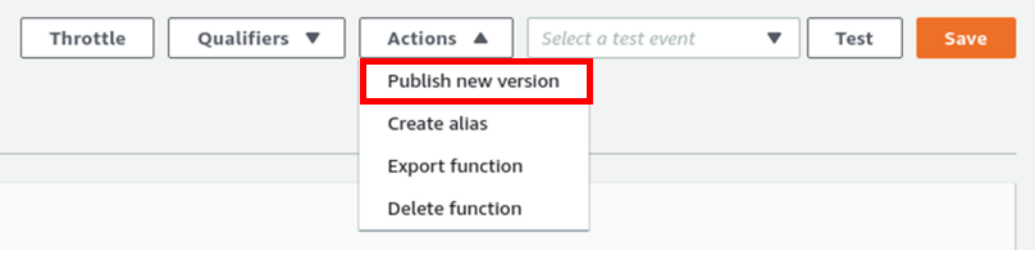


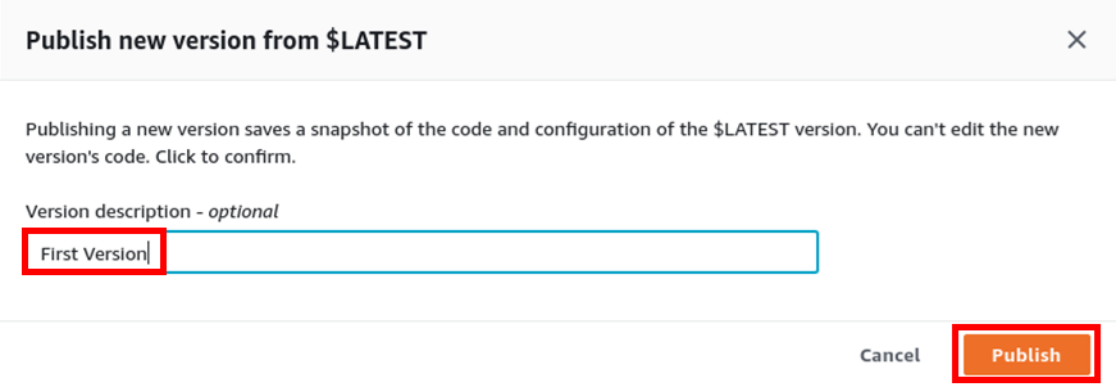
The Lambda function that runs AWS IoT Greengrass on the development board of LG8111 is in .zip format, so code entry type is zip and Runtime is python 2.7 to run Lambda on python 2.7.

Handler is "[Lambda file name] [Handler name in Lambda file]" is specified in. Register as "greengrassMobileet.function_handler" because the example code has the file name is "greengrassMobilnet" and the handler name in the example title lambda file is "function_handler.".

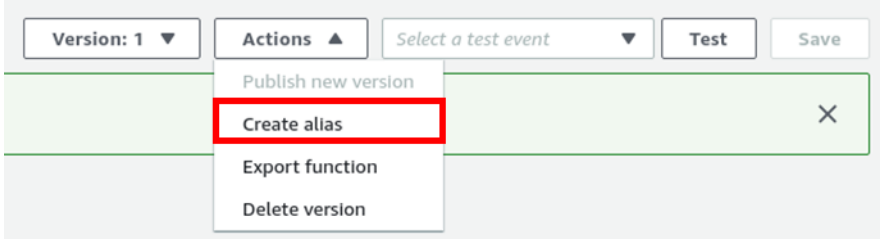
Finally, press the Upload button to select and register a zip file with the example code compressed.

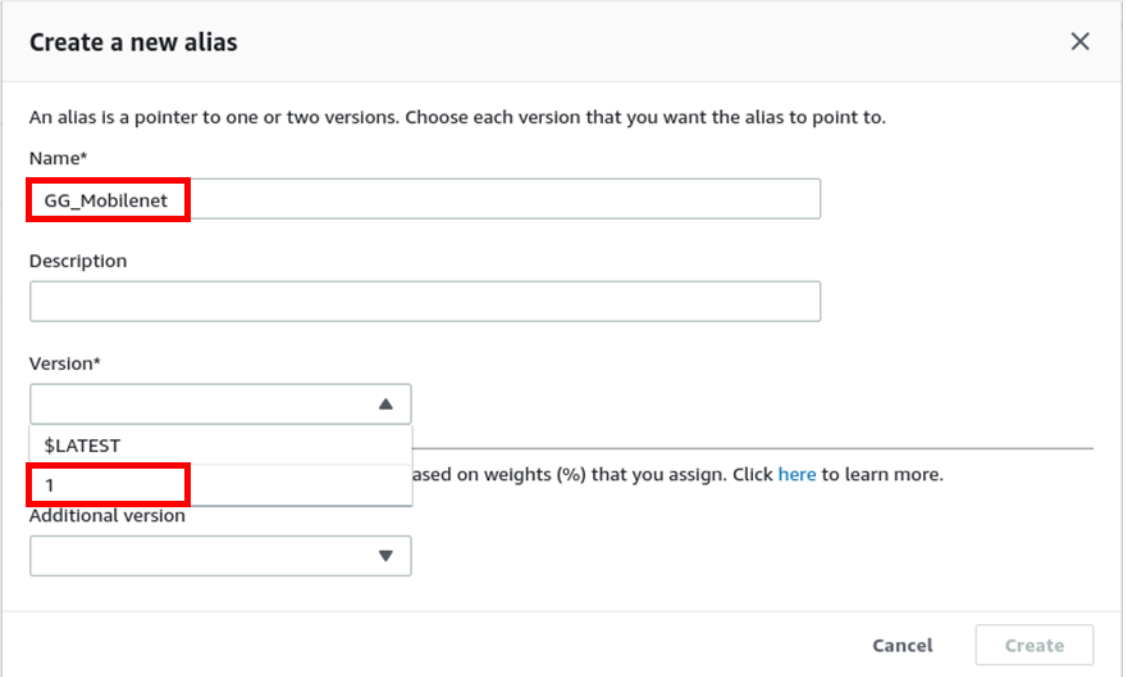
Press the save button in the upper right corner of the screen to save your registration, select Publish new version in Actions, and enter a description for the saved version.



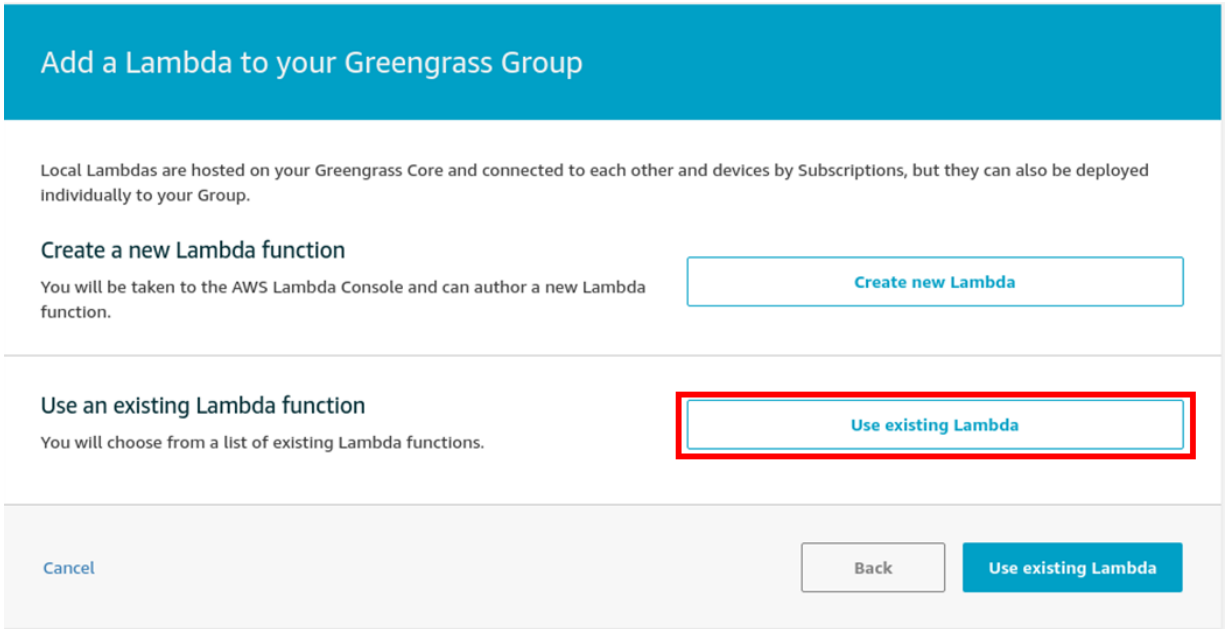


Saves the description for the version and saves it with alias and version for that version. (If you select Version with #LATEST, it may not work.)

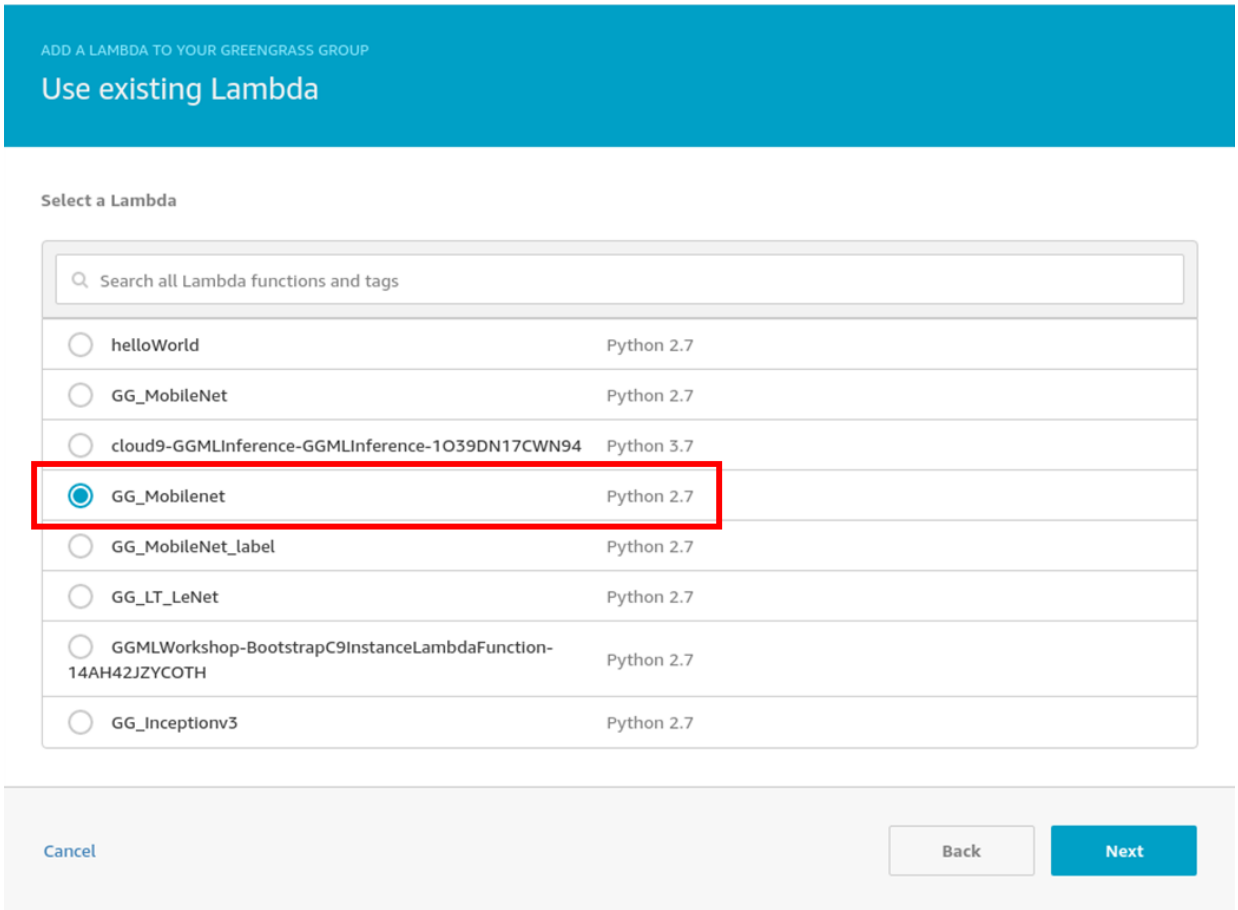


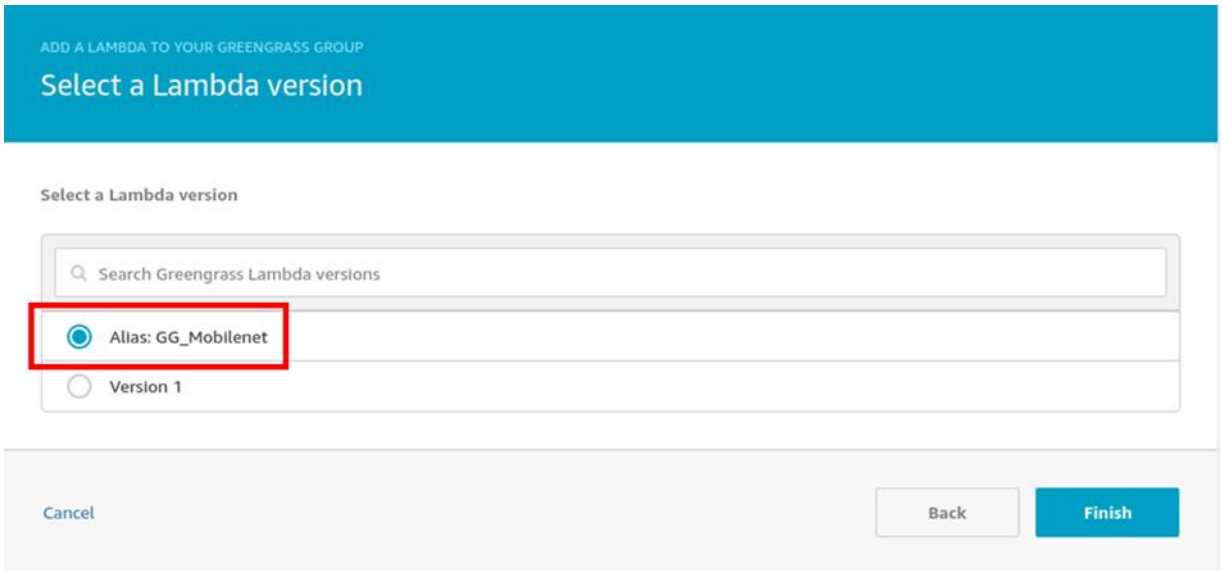


Select Lambda->Add Lambda and select "Use existing Lambda" to use the saved lambda.

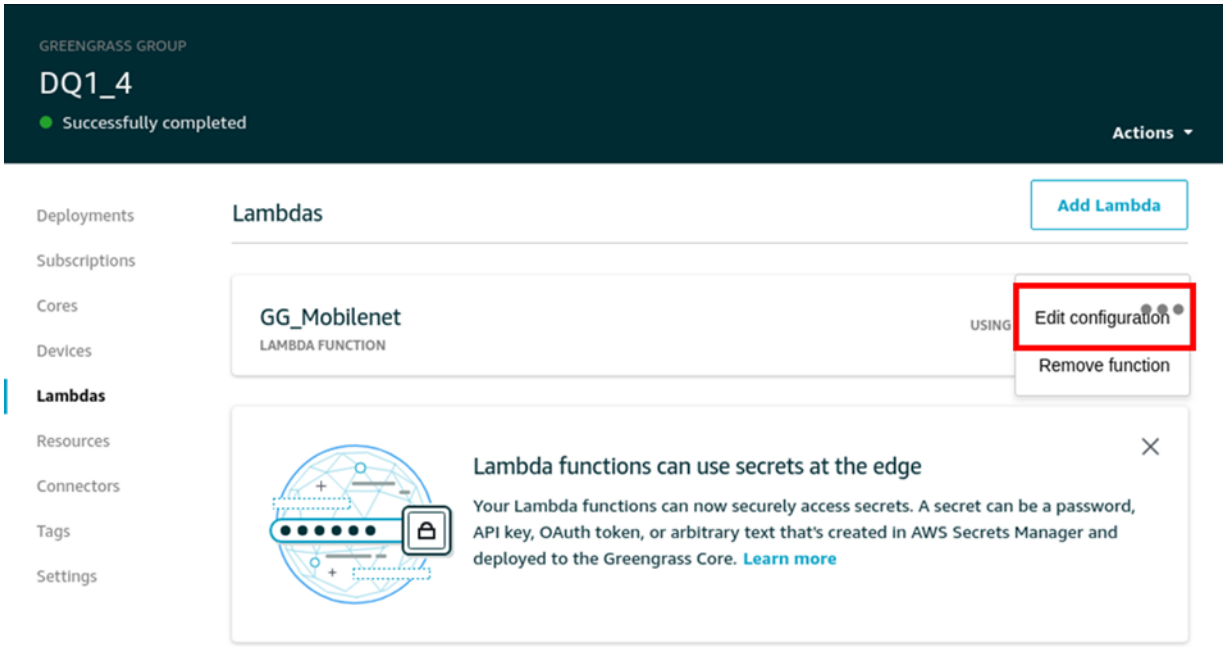


Select Lambda name, Lambda version of Lambda that you registered for testing.





For the test, you must modify the configuration of the Lambda function that you registered. Select Edit configuration by pressing the three circles to the right of the Lambda function added for this purpose.

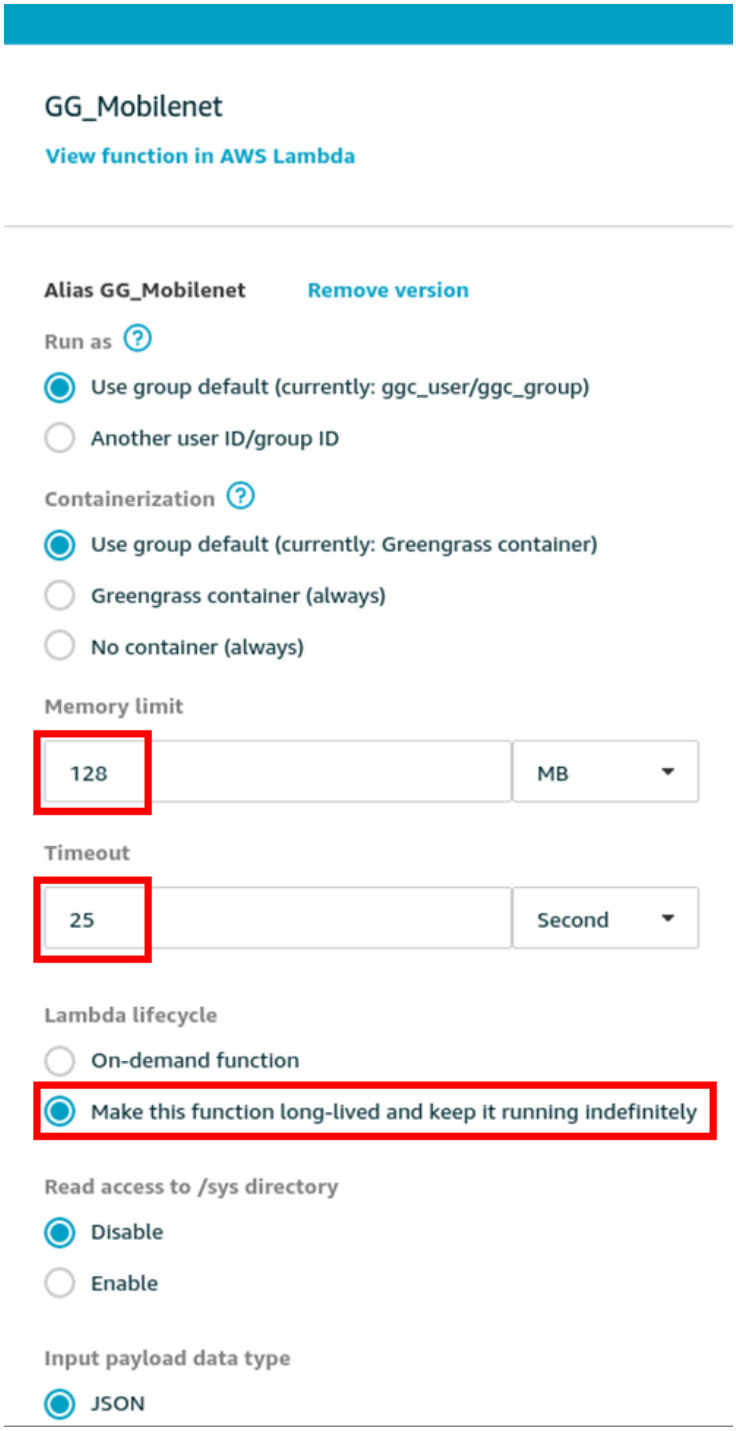


Correct the Memory limit, Timeout, and Lambda lifetime during the Lambda configuration.

Memory limit sets the memory limit to 128 MB when running the Lambda function.

Timeout is the Lambda function operation wait time, set to 25 seconds because Test Lambda transfers results every 5 seconds.

Lambda lifecycle saves the changes made by setting it to "Make this function log-live and keep it running indefinitely" running in the lifecycle setting of the Lambda function so that it can always operate for Test.



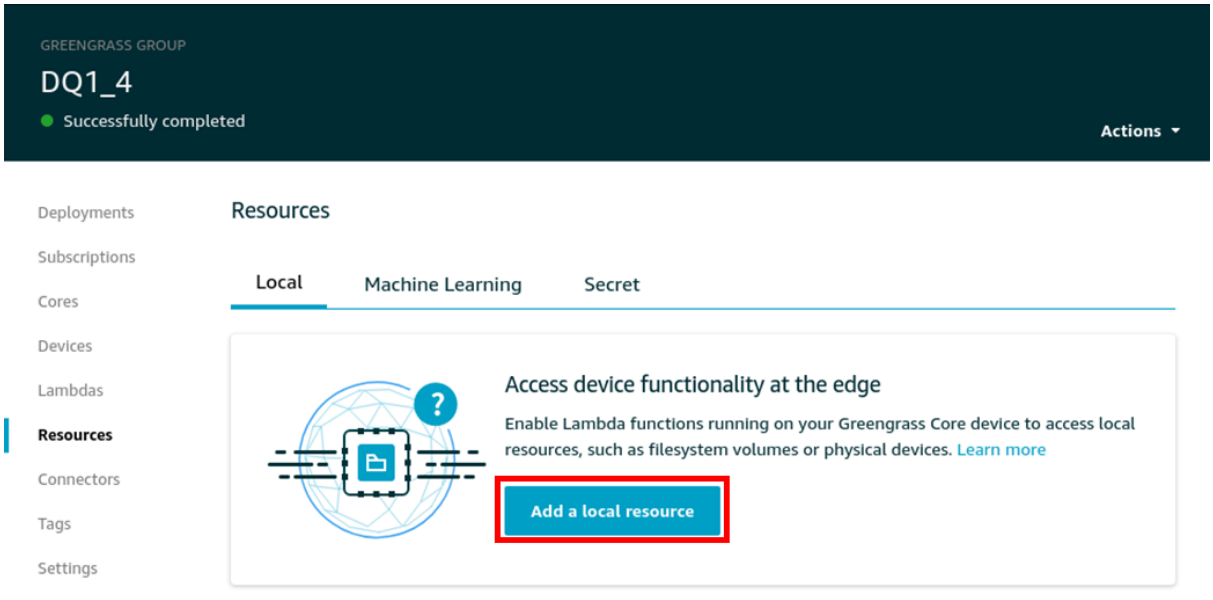
2. Local Resource Registration

Register the local resources (ex. camera, sensor, LG AI processors, etc.) of the Device in the development board of LG8111, which is used to run the AWS IoT Greengrass. The registered local resource is the local resource used by the Lambda function to operate.

NAME	DEVICE PATH	DESCRIPTION

LG AI PROCESSOR	/dev/dq1_lne	LG-specific AI processor that efficiently processes deep learning algorithms (power saving, low latency)
CAMERA	/dev/video0	LG8111 Development Device의 Camera

Select the Local tab for the Resource to add local resource.



2-1. Added LG AI Processor Local Resource

Configure the device path and lambda function to use the local resource in order to add local resource for the LG AI processor. At this time, set the setting for Lambda function to read and write access.

Add a new local resource

Resource name

LNE_Driver

Resource type

- (●) Device
- ( ) Volume

Device path

/dev/dq1_lne

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

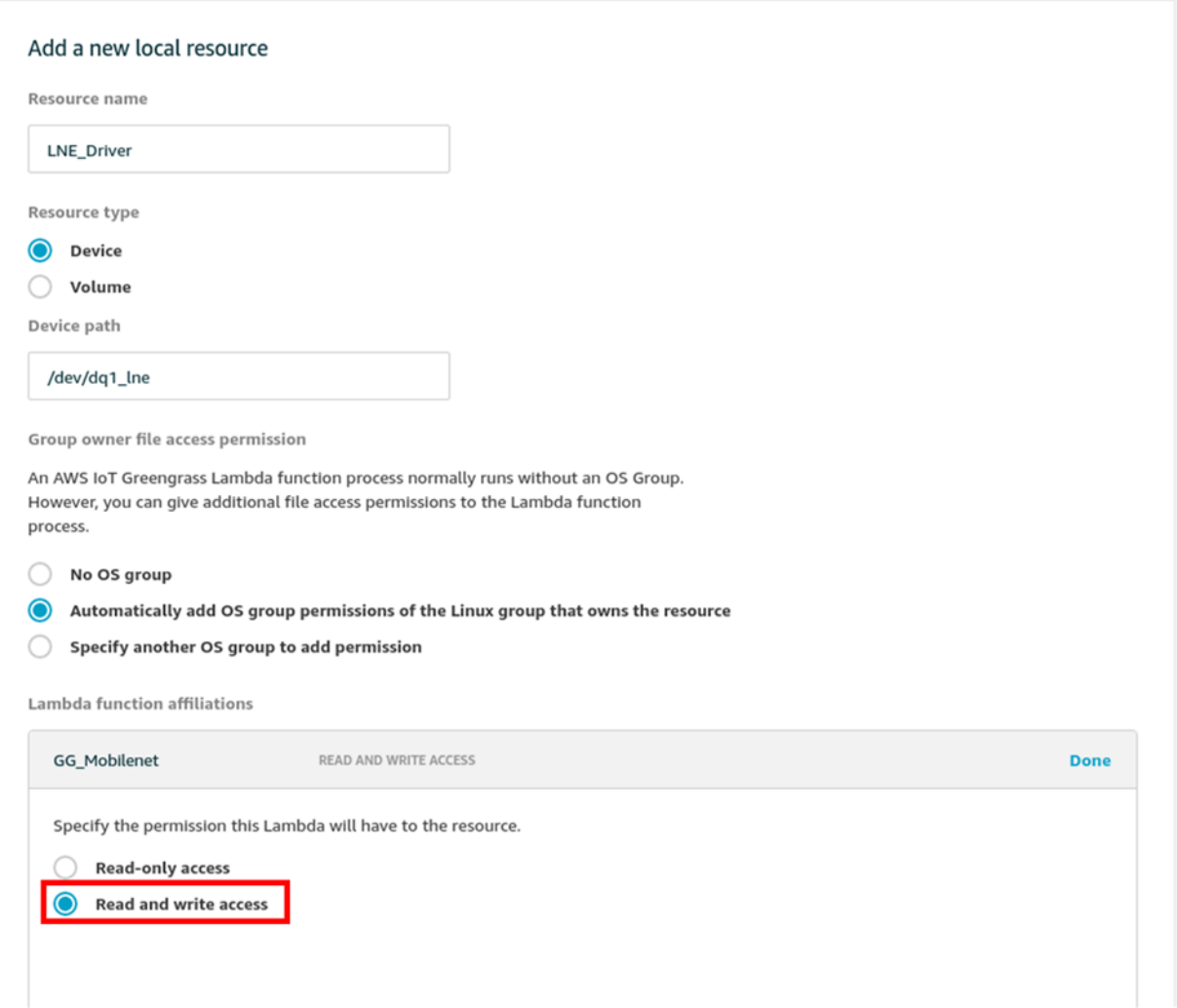
- ( ) No OS group
- (●) Automatically add OS group permissions of the Linux group that owns the resource
- ( ) Specify another OS group to add permission

Lambda function affiliations

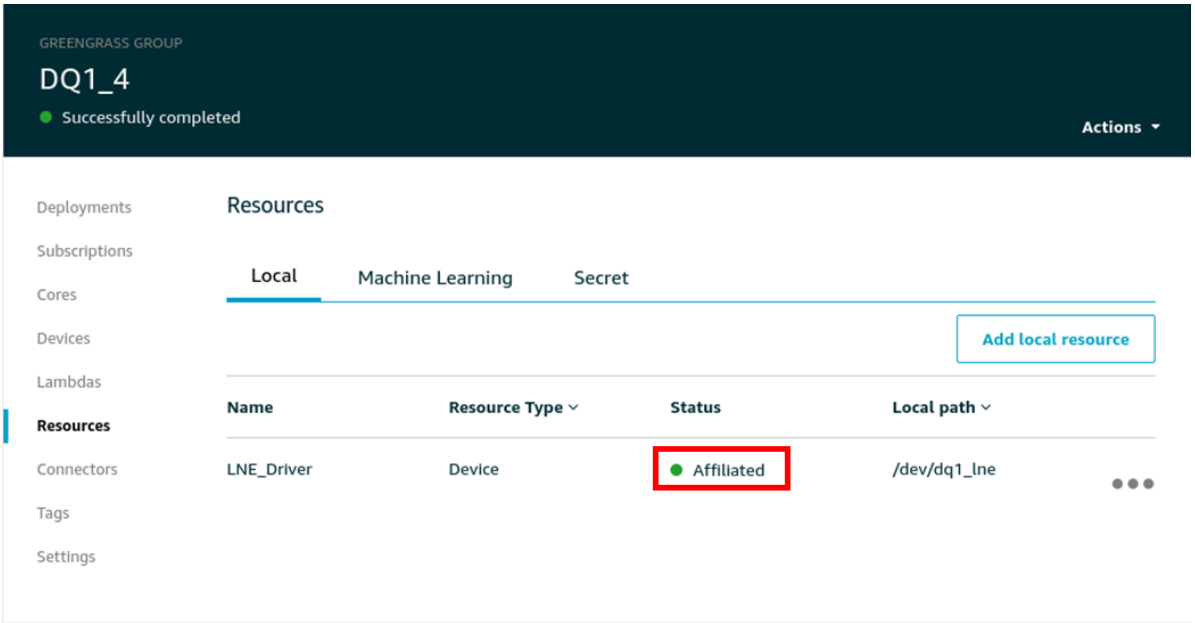
Resources must be affiliated with a Lambda function before deployment Done

Find

GG_Mobilenet

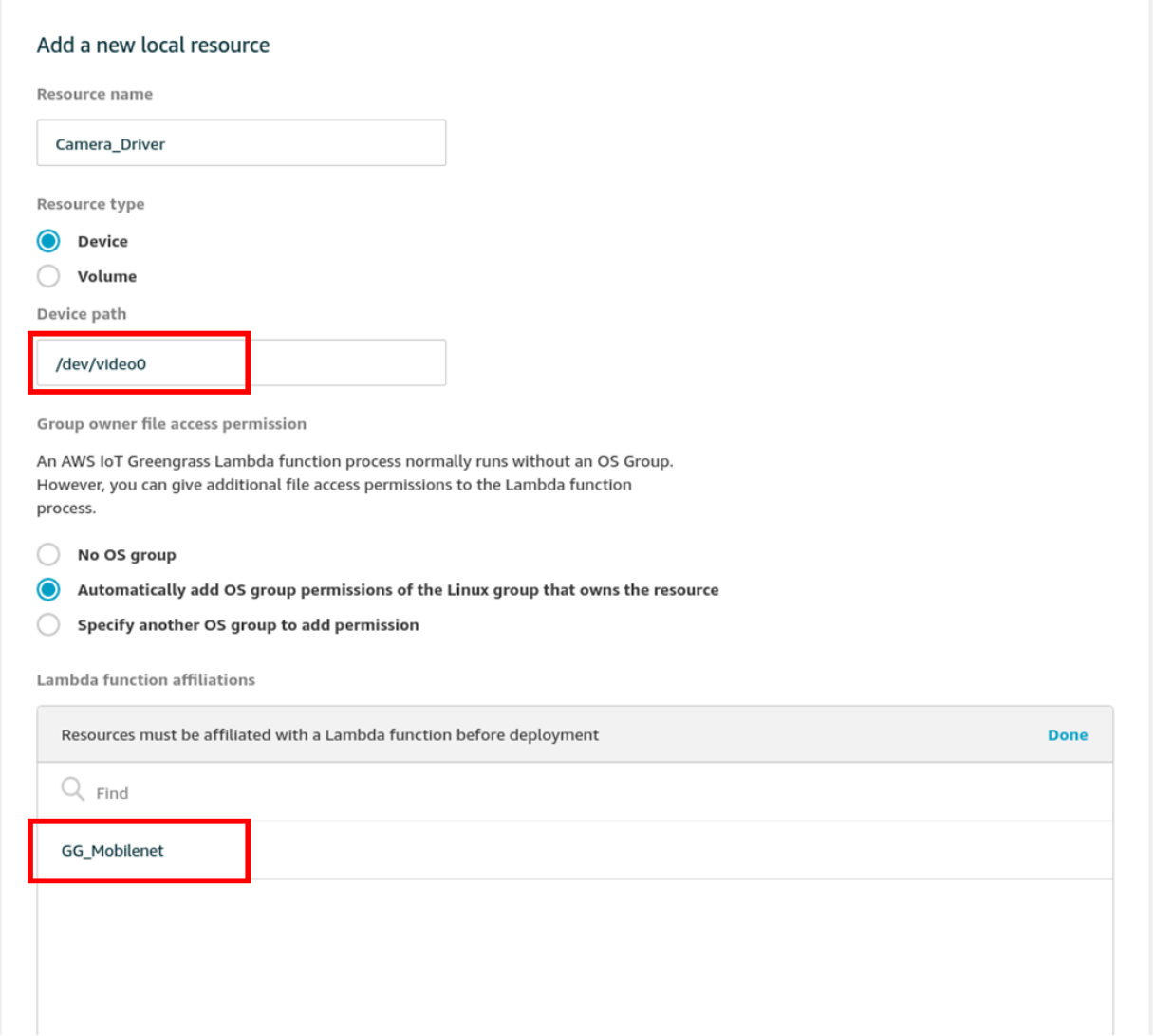


After the local resource registration of the LG AI processor is completed, the status changes to green when it is successfully connected to Lambda function.



2-2. Added Camera Local Resource

In order to use the LG8111 development board's Camera, the same process will be used to change only the Device path and register the LG AI processor as local resource.



Add a new local resource

Resource name

Camera_Driver

Resource type

- (●) Device
- ( ) Volume

Device path

/dev/video0

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

- ( ) No OS group
- (●) Automatically add OS group permissions of the Linux group that owns the resource
- ( ) Specify another OS group to add permission

Lambda function affiliations

GG_Mobilenet READ AND WRITE ACCESS Done

Specify the permission this Lambda will have to the resource.

- ( ) Read-only access
- (●) Read and write access

GREENGRASS GROUP

DQ1_4

● Successfully completed

Actions

- Deployments
- Subscriptions
- Cores
- Devices
- Lambdas
- **Resources**
- Connectors
- Tags
- Settings

Resources

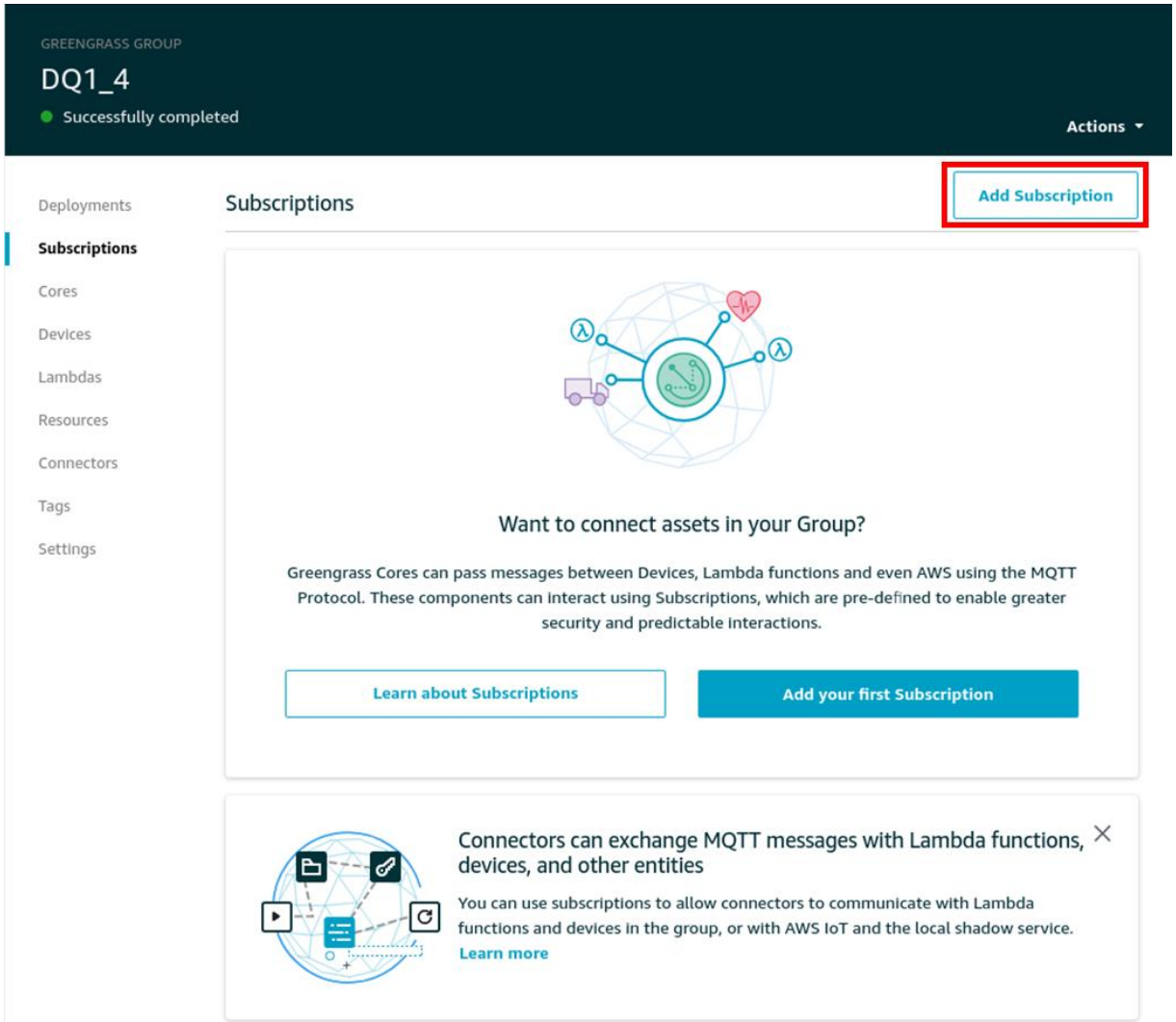
Local | Machine Learning | Secret

Add local resource

Name	Resource Type	Status	Local path
Camera_Driver	Device	● Affiliated	/dev/video0
LNE_Driver	Device	● Affiliated	/dev/dq1_lne

2-3. Subscription & Lambda Deploy

Before deploying Lambda, you must configure which Topic to subscribe to. Select and configure Add Subscriptions in Subscriptions.



In the Subscription, source is the name of the Lambda that you registered, and target is the AWS IoT Greengrass, so select IoT Cloud and enter the Topic name. In this test, we set the topic name to "tflite/lne", so we set the topic to "tflite/lne".

greengrassMobilenet.py

```
..
85          client.publish(topic="tflite/lne", queueFullPolicy="AllOrException", ₩
86                  payload="Result: {}".format(labels[lne_answer]))
...
```

CREATE A SUBSCRIPTION

Select your source and target

A Subscription consists of a source, target, and topic. The source is the originator of the message. The target is the destination of the message. The first step is selecting your source and target.

Select a source

No objects selected Close

Services Devices **Lambdas** Connectors

Search

GG_Mobilenet

Select a target

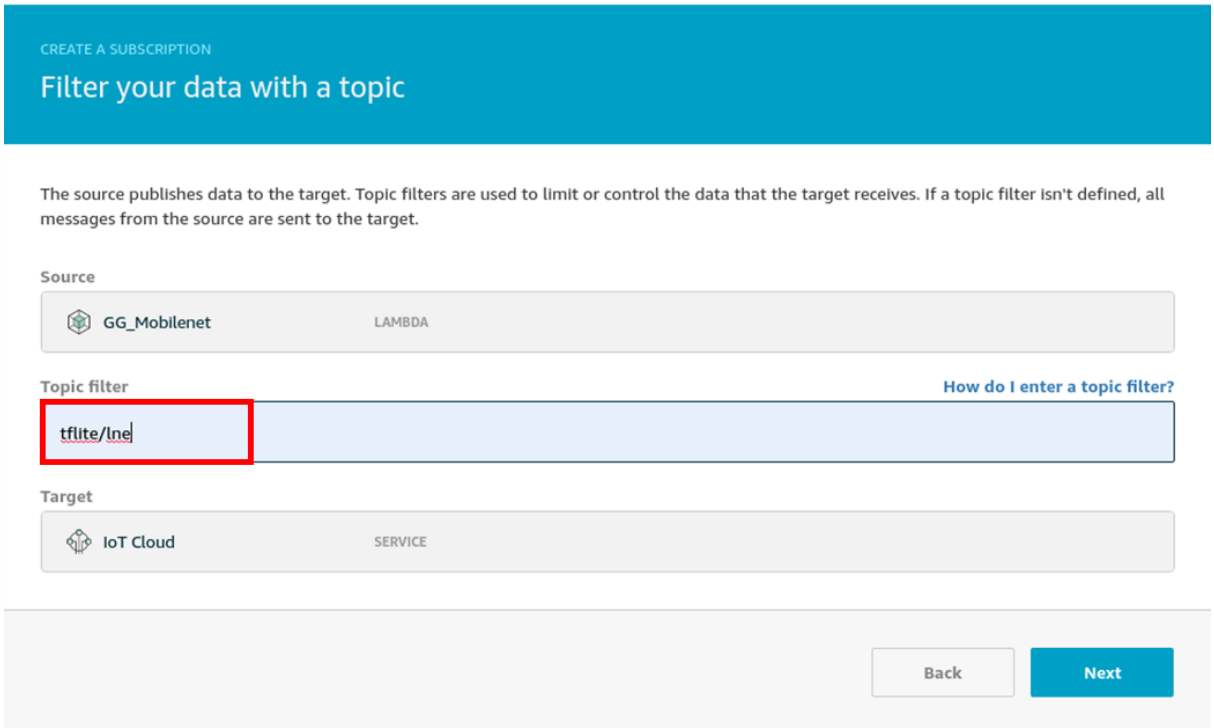
No objects selected Close

Services Devices Lambdas Connectors

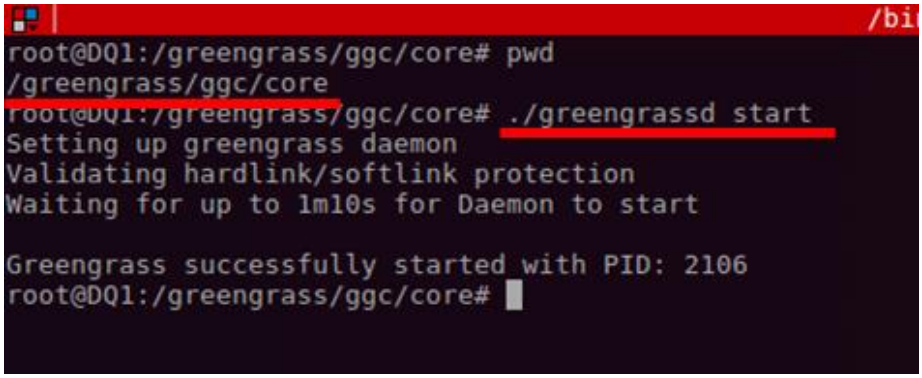
Search

IoT Cloud

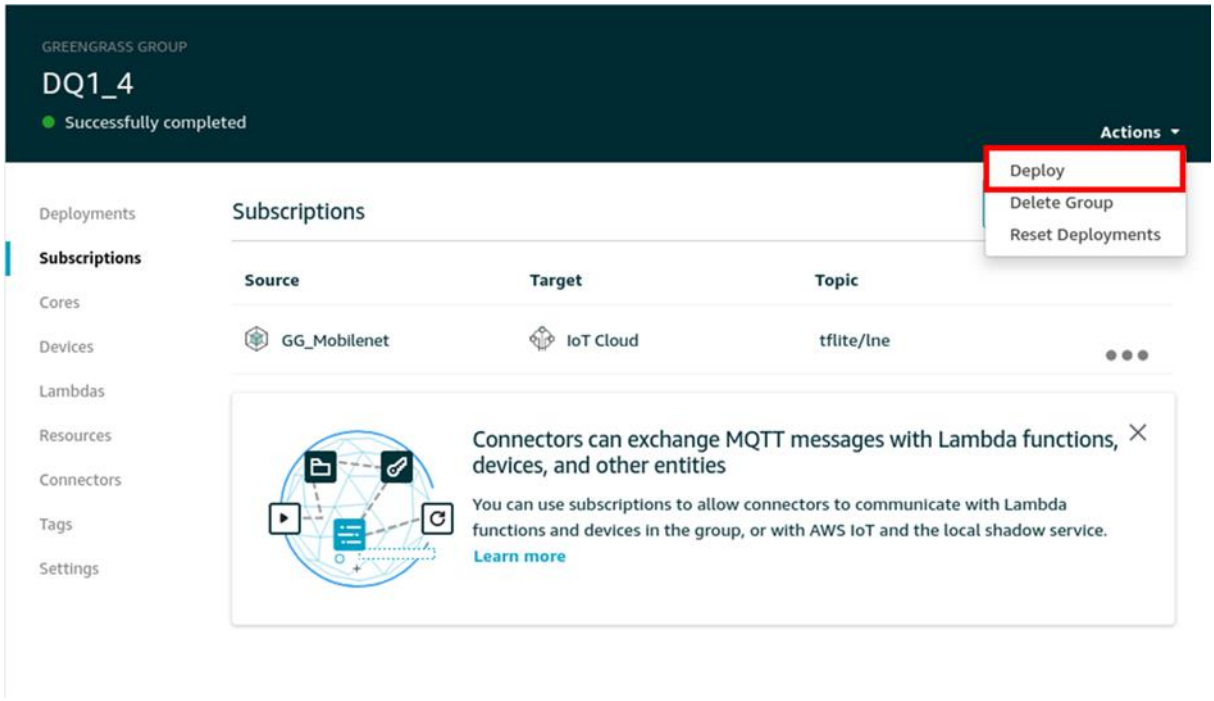
Local Shadow Service

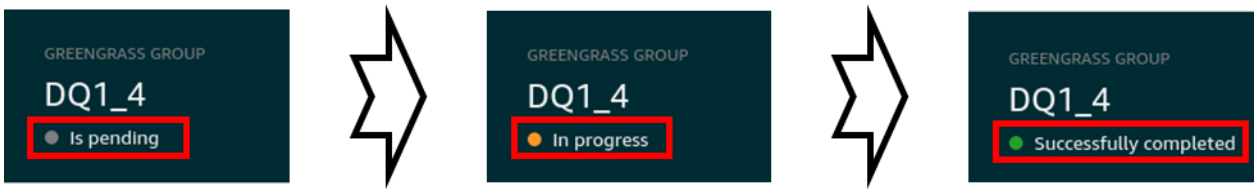


AWS IoT Greengrass Daemon must be running on the development board of the LG8111 before Deploying the Lambda function. Run the AWS IoT Greengrass Daemon with the "greengrass start" command at the "/greengrass/gcc/core" location during installation, and the AWS IoT Greengrass Daemon with root privileges is located at the "/greengrass/gcc/core" location.



Run AWS IoT Greengrass Daemon on the development board of the LG8111, press the Actions button in the upper right corner of the screen in the AWS IoT Greengrass console, select Deploy, and send the saved Lambda to the development board of the LG8111. When Deploy is in progress, the circle on the left side of the screen changes from gray to yellow to green, and each signifies preparation, transfer in progress, and deployment is complete.





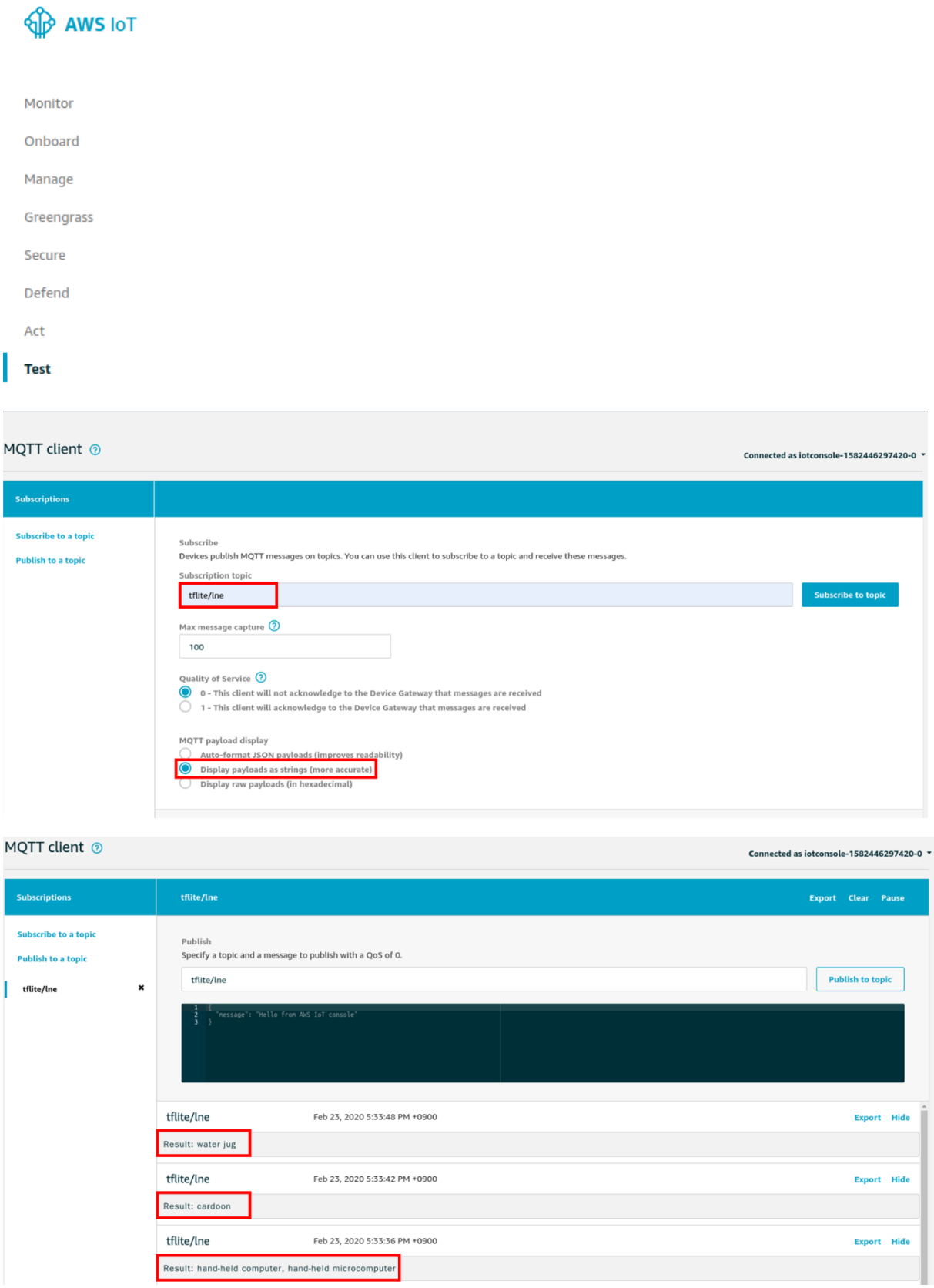
2-4. AWS IoT Greengrass Test

Use the test provided by the AWS IoT Greengrass to verify that the Lambda function deployed from the AWS IoT Greengrass works properly on the LG8111 development board.

What you set up in Test is the part that sets up the format in which MYTT, the method of exchanging messages with the topic name that you are subscribing to, and the AWS IoT Greengrass.

The topic used in the example is “tflite/lne”. Since the LG8111 Development Board will forward the result value as a string, set it as a string.

Once every five seconds when the configuration is complete, you can perform an inference through Mobilenet on the images received by the camera and see the result value sent to the AWS IoT Greengrass.


AWS IoT
Monitor
Onboard
Manage
Greengrass
Secure
Defend
Act
Test
MQTT client
Connected as iotconsole-1582446297420-0
Subscriptions
Subscribe to a topic
Publish to a topic
Subscribe
Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.
Subscription topic
tflite/lne
Subscribe to topic
Max message capture
100
Quality of Service
0 - This client will not acknowledge to the Device Gateway that messages are received
1 - This client will acknowledge to the Device Gateway that messages are received
MQTT payload display
Auto-format JSON payloads (improves readability)
Display payloads as strings (more accurate)
Display raw payloads (in hexadecimal)
MQTT client
Connected as iotconsole-1582446297420-0
Subscriptions
tflite/lne
Export
Clear
Pause
Subscribe to a topic
Publish to a topic
tflite/lne
Publish
Specify a topic and a message to publish with a QoS of 0.
tflite/lne
Publish to topic
{
"message": "Hello from AWS IoT console"
}
tflite/lne
Feb 23, 2020 5:33:48 PM +0900
Export
Hide
Result: water jug
tflite/lne
Feb 23, 2020 5:33:42 PM +0900
Export
Hide
Result: cardoon
tflite/lne
Feb 23, 2020 5:33:36 PM +0900
Export
Hide
Result: hand-held computer, hand-held microcomputer

AWS IoT Greengrass Operational References

Deploy completed while AWS IoT Greengrass is running, but you may need to verify that it works for debugging purposes.

At this time, the method used for debugging is as follows.

1. Error occurred during distribution

 - /greengrass/ggc/var/log/system/runtime.log

 - Log with errors seen during greengrass operation

2. Distributed but AWS IoT Greengrass Does Not Transfer Result Values

 - /greengrass/ggc/var/log/user/[Region]/[public key]/[lambda function].log

 - You can see the area by pressing the area displayed at the top of the AWS IoT Greengrass screen.