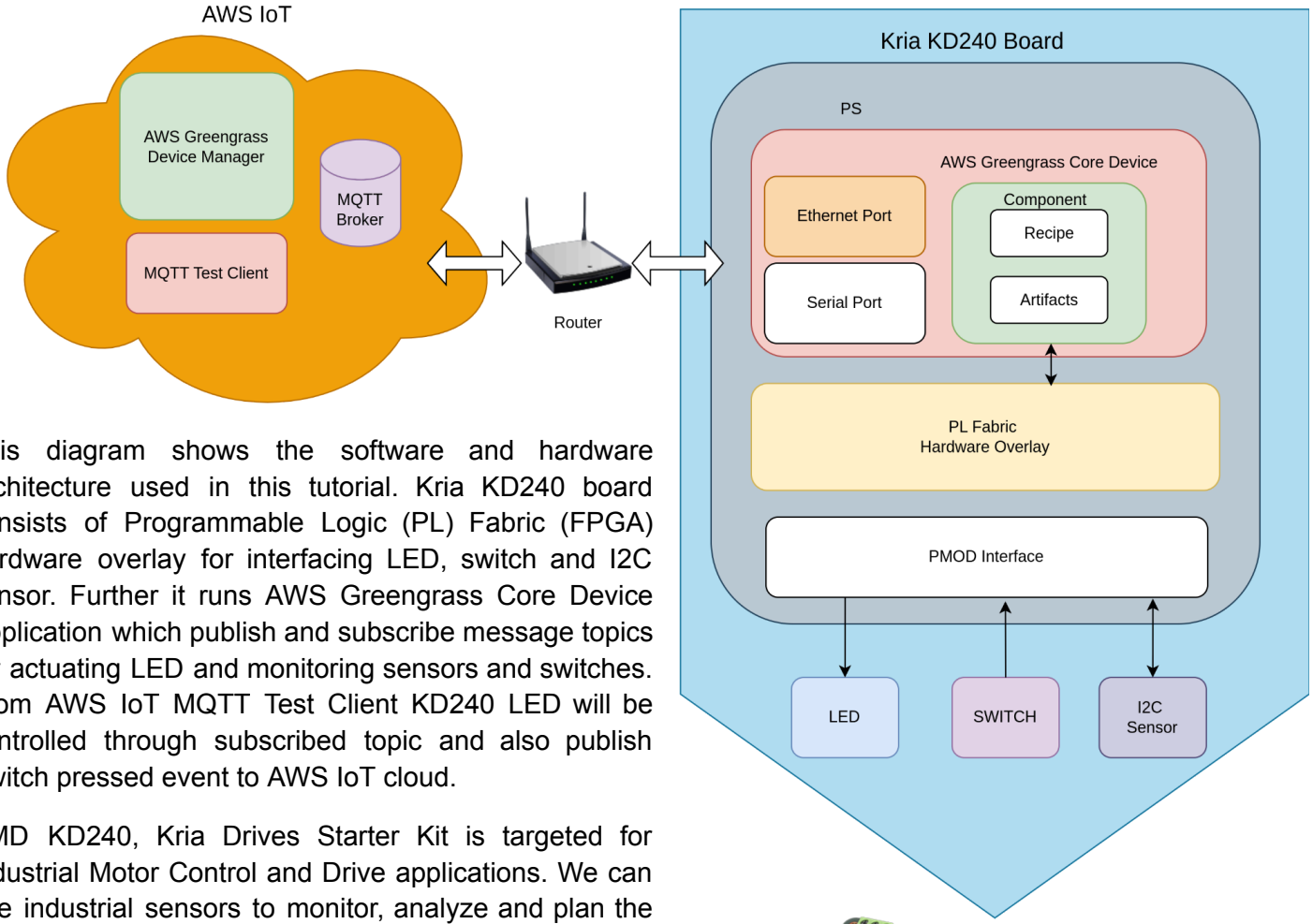
# KD240 to AWS IoT Greengrass Architecture

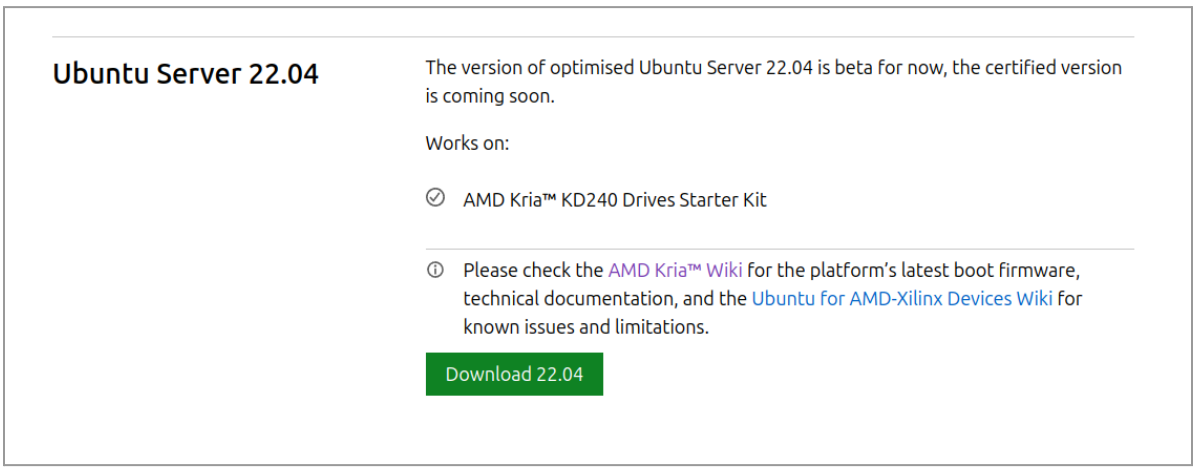



This diagram shows the software and hardware architecture used in this tutorial. Kria KD240 board consists of Programmable Logic (PL) Fabric (FPGA) hardware overlay for interfacing LED, switch and I2C sensor. Further it runs AWS Greengrass Core Device Application which publish and subscribe message topics for actuating LED and monitoring sensors and switches. From AWS IoT MQTT Test Client KD240 LED will be controlled through subscribed topic and also publish Switch pressed event to AWS IoT cloud.

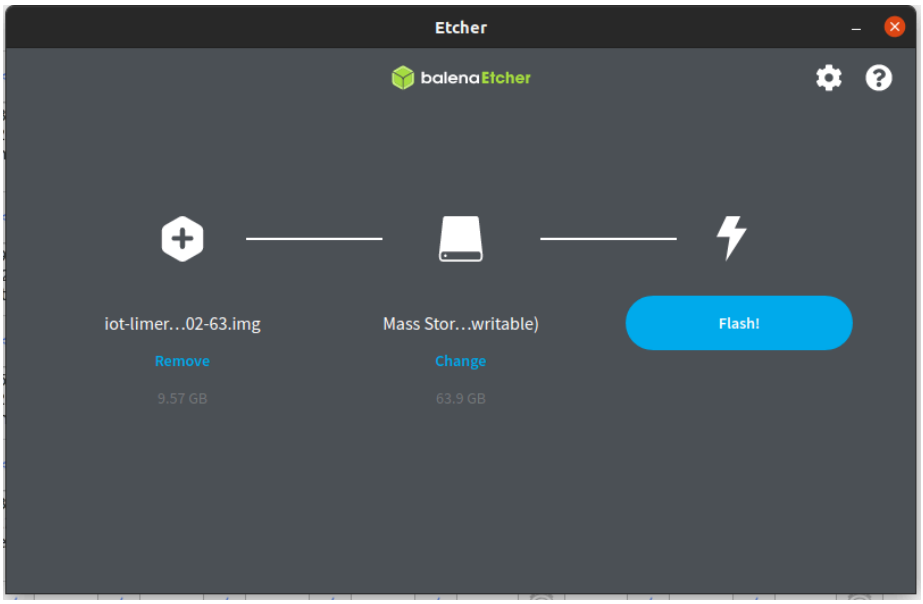
AMD KD240, Kria Drives Starter Kit is targeted for Industrial Motor Control and Drive applications. We can use industrial sensors to monitor, analyze and plan the industrial application in cloud platform. This board is also targeted for predictive maintenance type of applications.

# Preparing Ubuntu 22.04 OS for KRIA KD240 board

Download the Ubuntu 22.04 image from the download link




Next, prepare the SD card with the above downloaded Ubuntu image using burning tools like Balena Etcher.




Now boot the KD240 with the SD card with Ethernet and USB to Serial cable connected to board. We will be using Serial console for initial access and debugging and Ethernet network for accessing through SSH and KD240 connected to the internet.

For initial login here are the Login Details:
Username : ubuntu
Password: ubuntu
This will ask to change the password. So update the password and login the system.
After successful login, one can access the KD240 device console.

# Installing hardware overlay in KD240

Get the KD240 firmware folder. It contains:

- kd240-gpio-i2c.bit.bin
- kd240-gpio-i2c.dtbo
- shell.json

Copy these file to the KD240 board. For firmware to be loaded using xmutil (FPGA manager), one has to copy these file at “/lib/firmware/xilinx”.
For this create the folder at “kd240-gpio-i2c” at “/lib/firmware/xilinx” and copy the files in “kd240-gpio-i2c” folder.

```
cd /lib/firmware/xilinx
sudo mkdir kd240-gpio-i2c
Cd kd240-gpio-i2c
sudo cp <kd240-firmware directory>/kd240-gpio-i2c* ./
sudo cp <kd240-firmware directory>/shell.json ./
```

Next, check the available fpga firmware using `xmutil listapps` command. `kd240-gpio-i2c` will be available in the list.

```
ubuntu@kria:~$ sudo xmutil listapps
                  Accelerator         Accel_type                          Base          Base_type      #slots(PL+AIE)        Active_slot

              k24-starter-kits           XRT_FLAT              k24-starter-kits           XRT_FLAT             (0+0)                  0,
                kd240-gpio-i2c           XRT_FLAT                kd240-gpio-i2c           XRT_FLAT             (0+0)                  -1
ubuntu@kria:~$
```

Next load the `kd240-gpio-i2c` firmware, which contains necessary hardwares (gpio) and interfaces. In our Greengrass Demo we will be using these gpio to trigger the publishing data to AWS Greengrass IoT cloud server and also actuate GPIO on the message received from AWS cloud.

```
sudo xmutil unloadapp
sudo xmutil loadapp kd240-gpio-i2c
```

```
ubuntu@kria:~$ sudo xmutil loadapp kd240-gpio-i2c
[  827.076900] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/firmware-name
[  827.087054] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/resets
[  827.096939] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/afi0
[  827.106454] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/clocking0
[  827.116398] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_0
[  827.126422] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_1
[  827.136450] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_gpio_0
[  827.146477] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_iic_0
kd240-gpio-i2c: loaded to slot 0
ubuntu@kria:~$
```

Now to access GPIO in user application, we will be using `gpiod` library.

# Installing gpiod packages

GPIOD packages are required to access the GPIO channels. It also provides python binding for accessing GPIO in python programming. Install the package using apt-get:

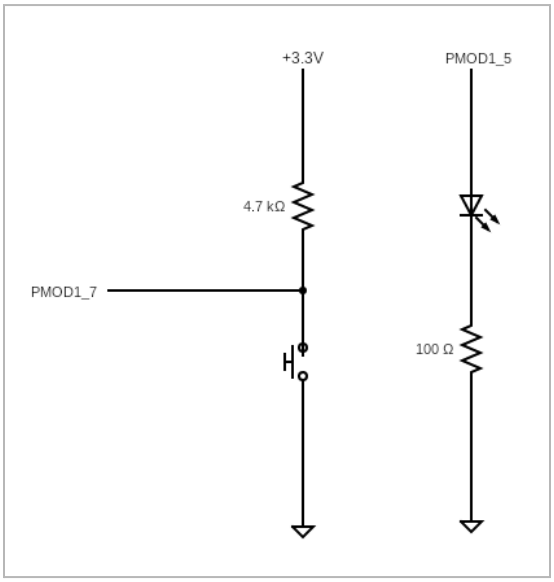
```
sudo apt update
sudo apt-get install gpiod python3-libgpiod
```

Now we can check the available gpio using gpiod  applications:

Using `gpiodetect` to get availabe gpio:

```
ubuntu@kria:~$ sudo gpiodetect
gpiochip0 [firmware:zynqmp-firmware:gpio] (4 lines)
gpiochip1 [zynqmp_gpio] (174 lines)
gpiochip2 [slg7xl45106] (8 lines)
gpiochip3 [80000000.gpio] (4 lines)
ubuntu@kria:~$
```

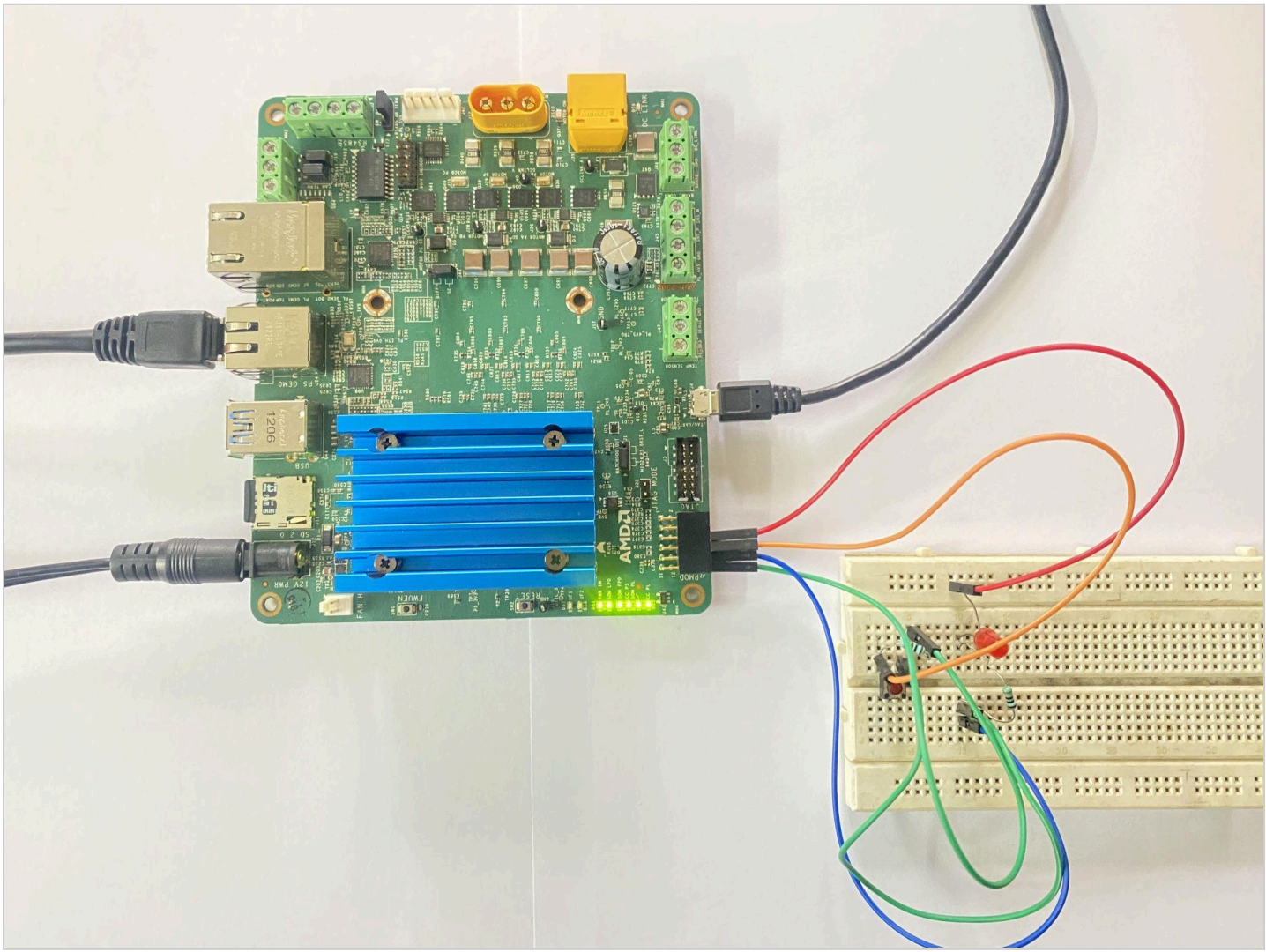
Here `gpiochip3` is the device corresponding to gpio in FPGA and it consists of 4 lines. Further these gpio lines are connected to PMOD 1 such that:
PMOD1-> 5 - gpiochip3 line 0
PMOD1-> 7 - gpiochip3 line 1




Schematic for LED and Switch Connection

| | | | | | | |
|---|---|---|---|---|---|---|
| 11 | 9 | 7 | 5 | 3 | 1 | PMOD UPPER |
| 12 | 10 | 8 | 6 | 4 | 2 | PMOD LOWER |
| Vcc | GND | I/O | I/O | I/O | I/O | |

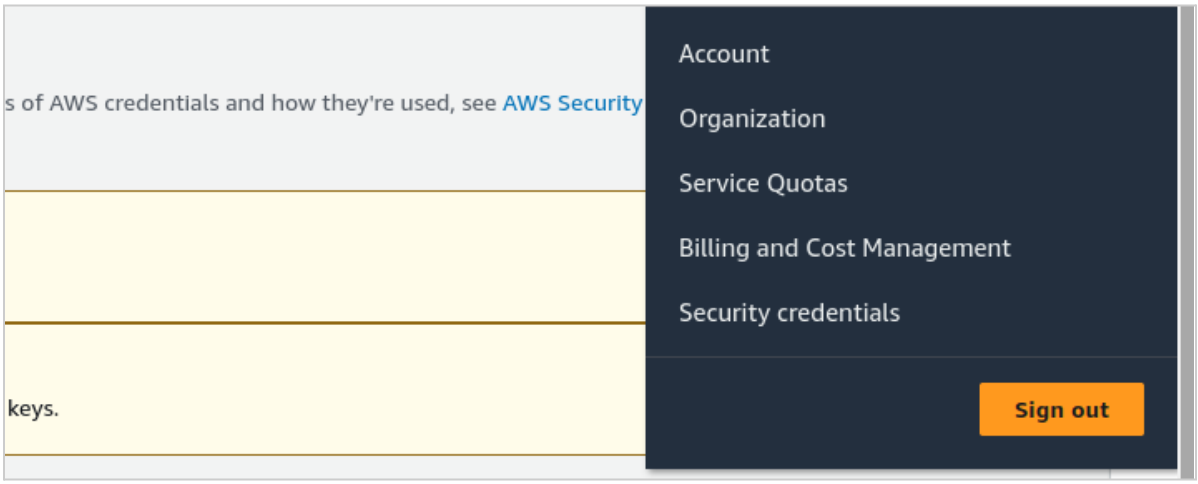
PMOD port numbering



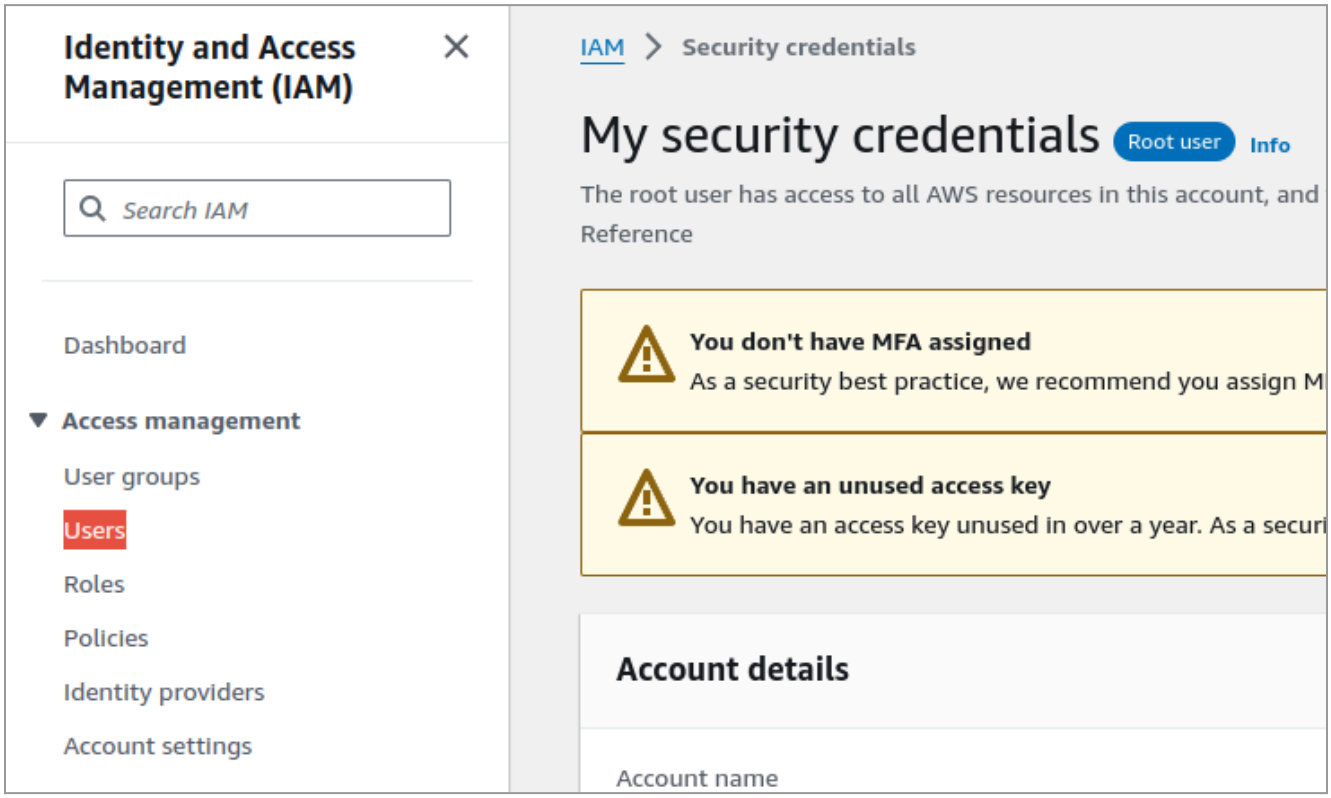
KD240 Board Connected to switch and LED through PMOD

# AWS IoT user creation

For and non human access to AWS services one has to create a user with required permissions.
- Login to AWS console
- Next go to `Security credentials` link available at root user drop down at top right corner of the AWS console

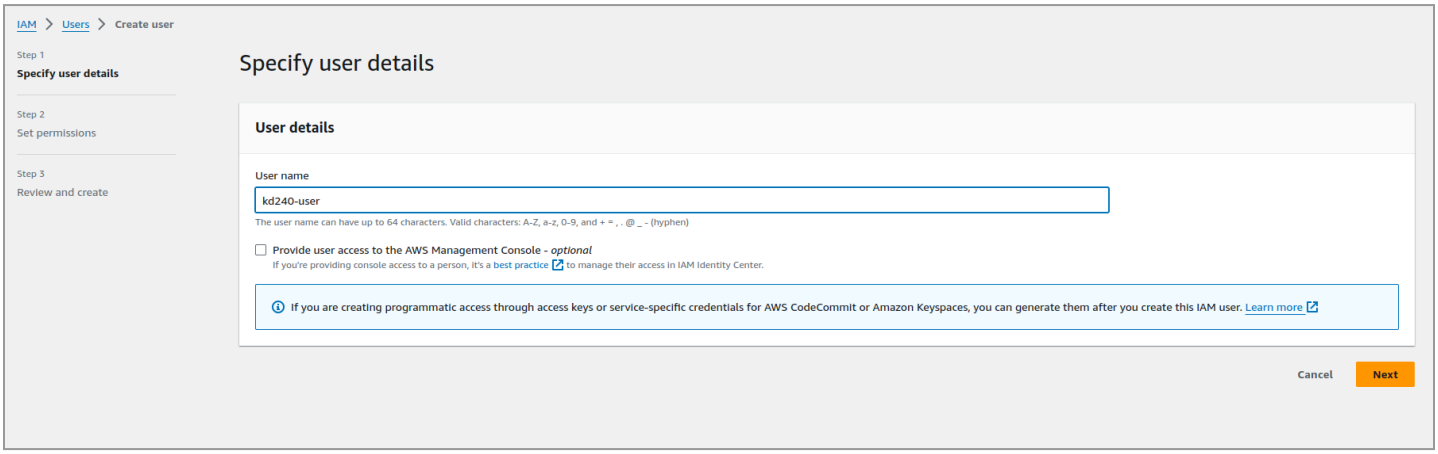



- Next Go to User management page by clicking at the User link at IAM sidebar.
This will list the available users.

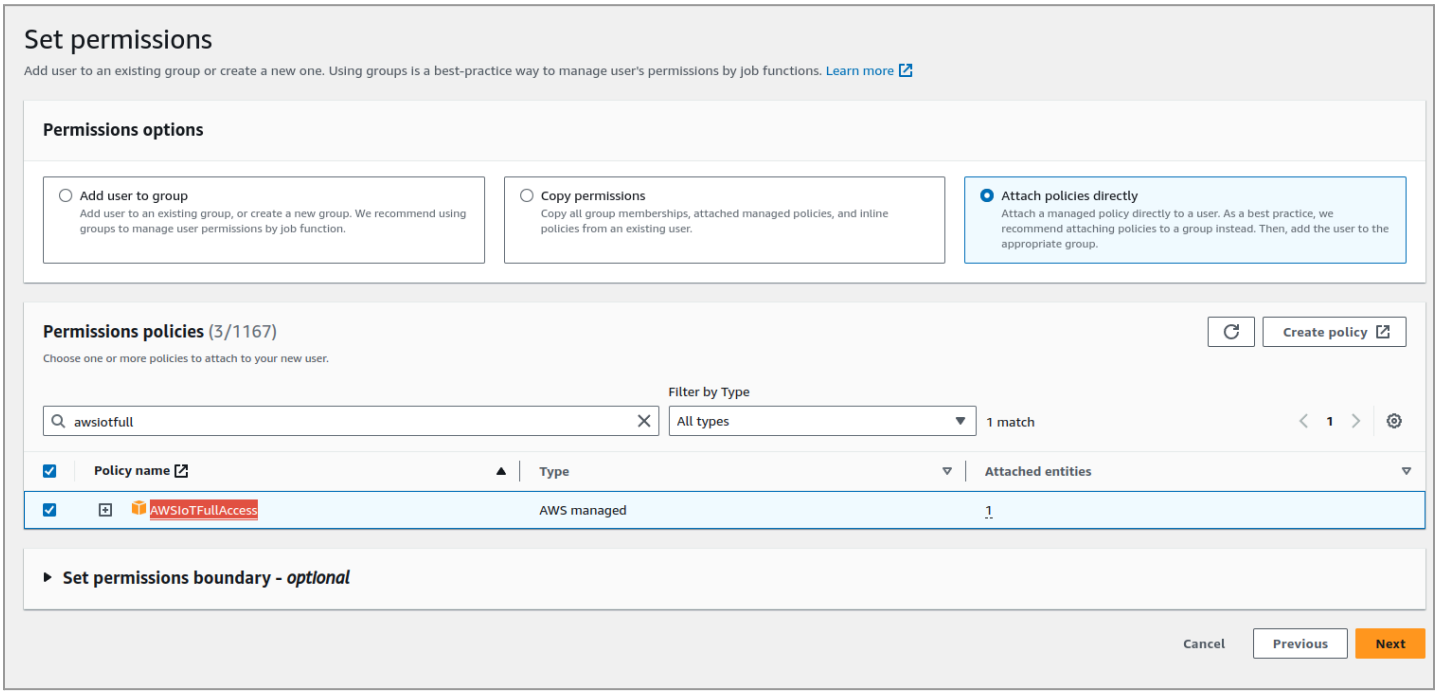


- Now create a new user for KD240 device by clicking the "Create User" button.

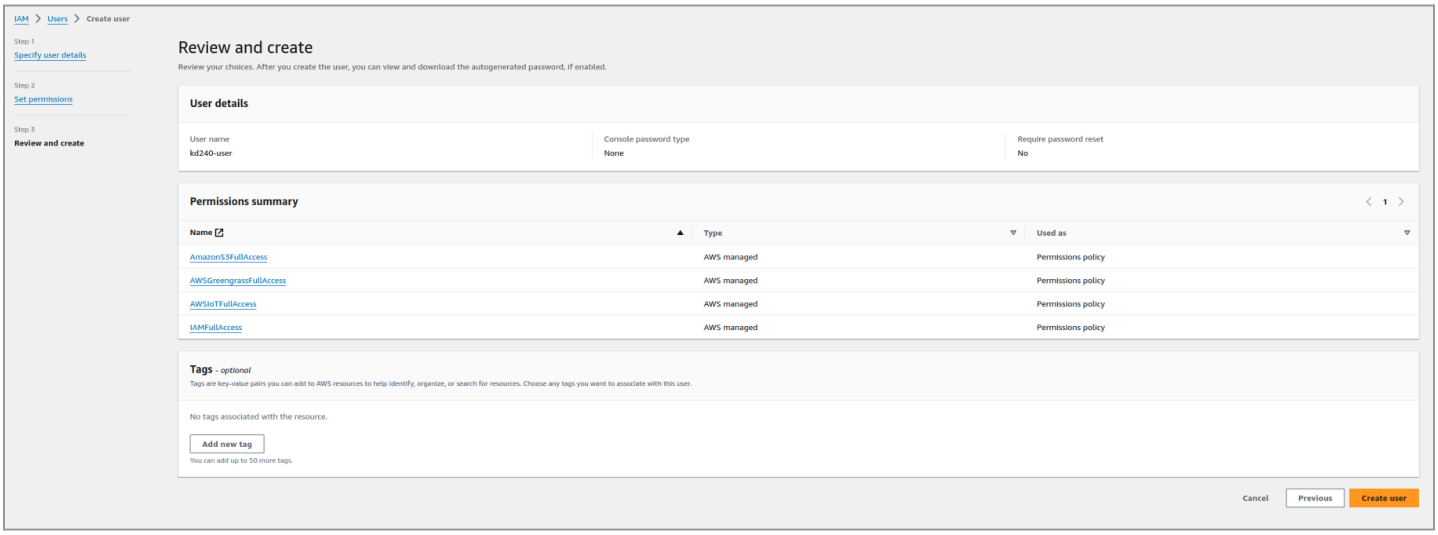

This will lead to step wise User creation forms. So fill the User details,
This will lead to step wise User creation forms.
So fill the User details, leave the console access unchecked as user does not have to access the AWS console through web.



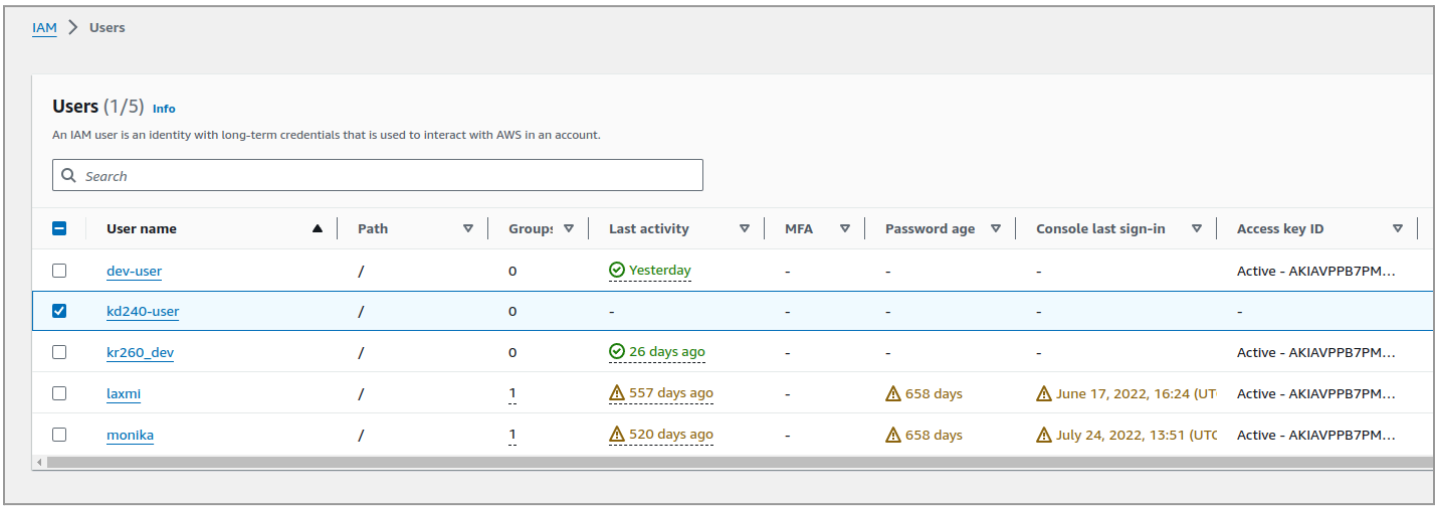

Next, update the Permissions options by attaching following  policies by selecting “Attach policies directly” in Permission options:
- AWSGreengrassFullAccess
- IAMFullAccess
- AWSIoTFullAccess
- AmazonS3FullAccess

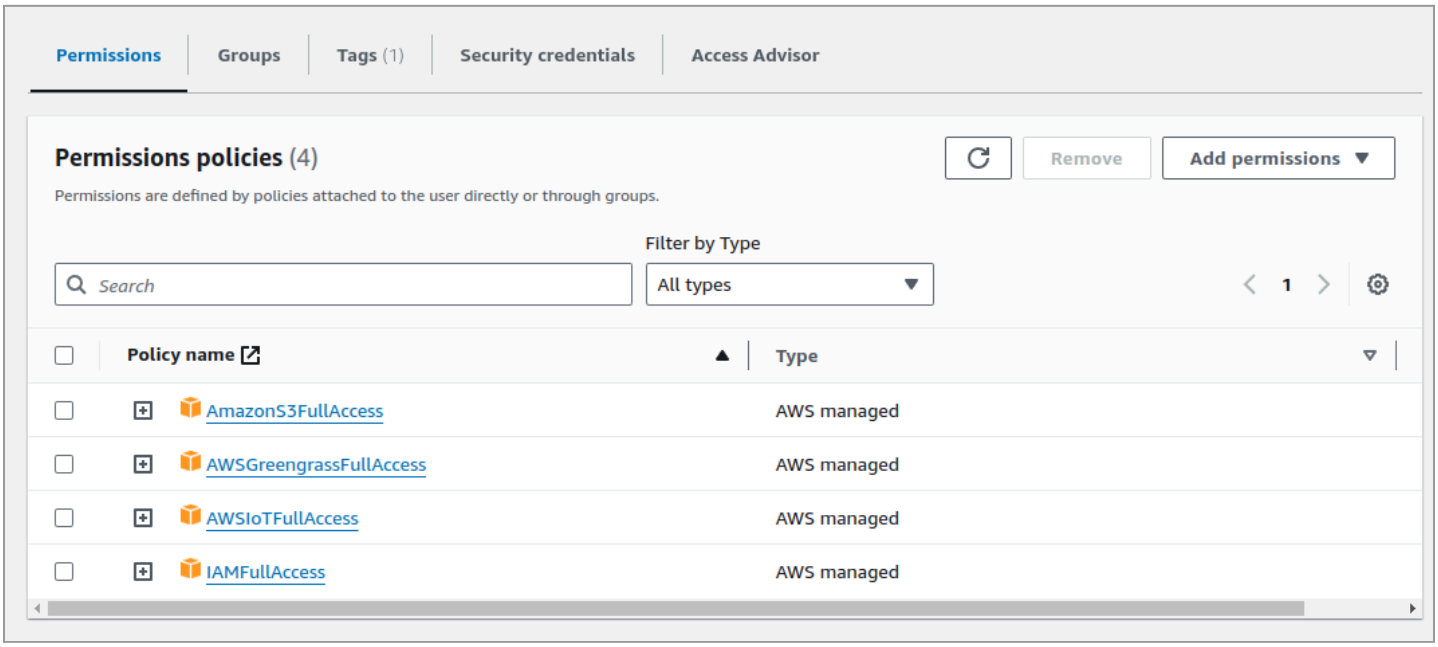



After finishing the above steps click "Create User" to finish the user creation.

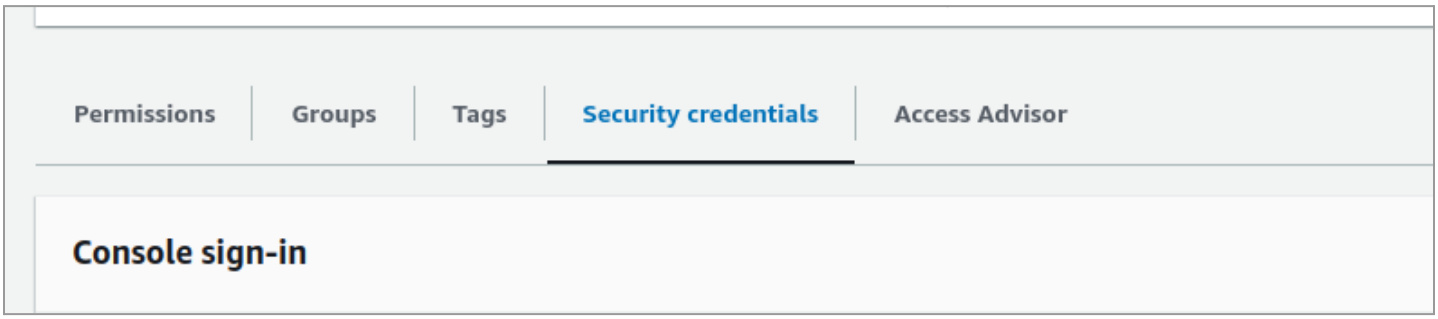



Next get the access token and access key for the user. For this open the user details by clicking on the user link in the Users table.

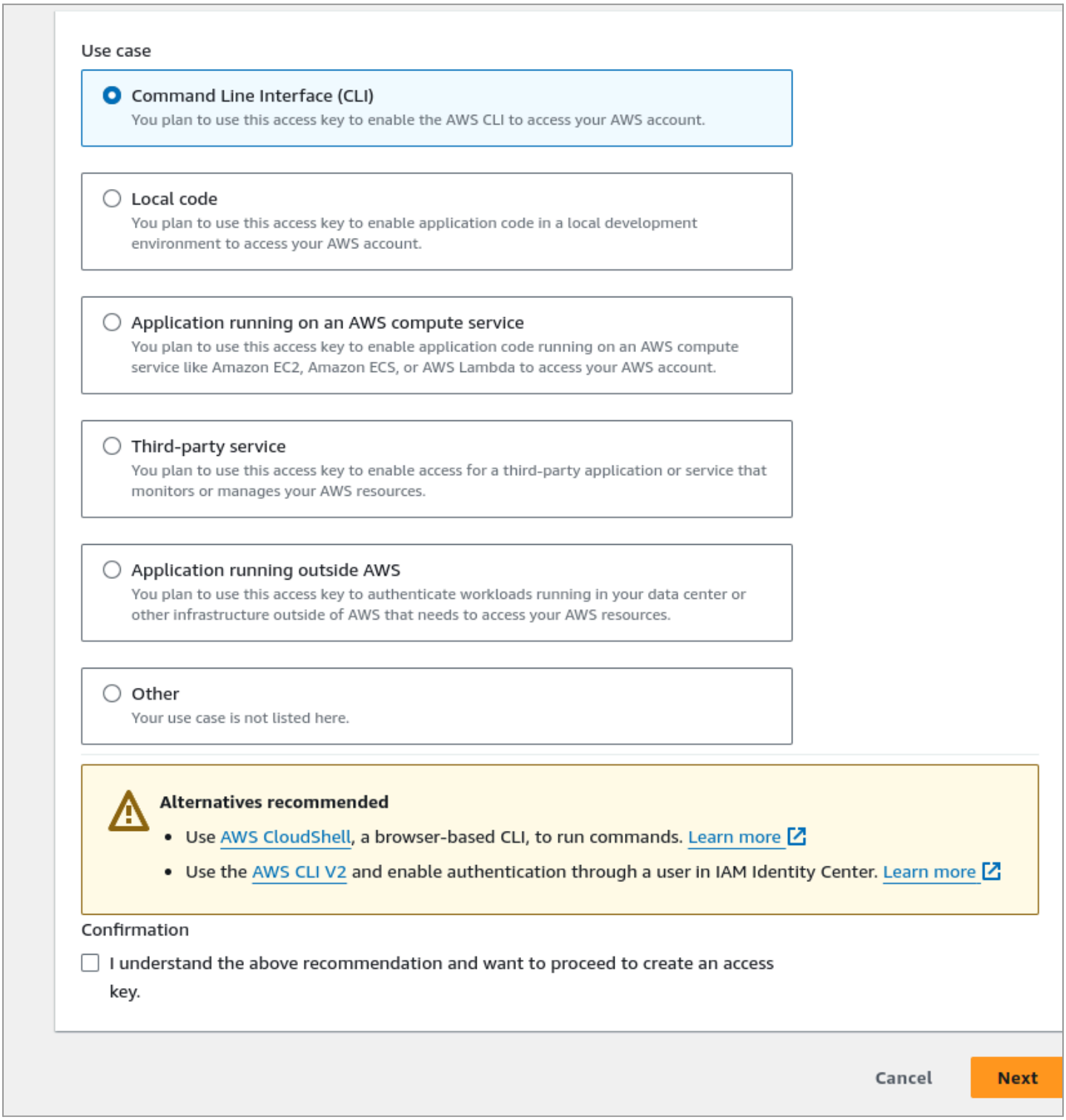



And go to "Security credentials" for creating the Access Key for the user.



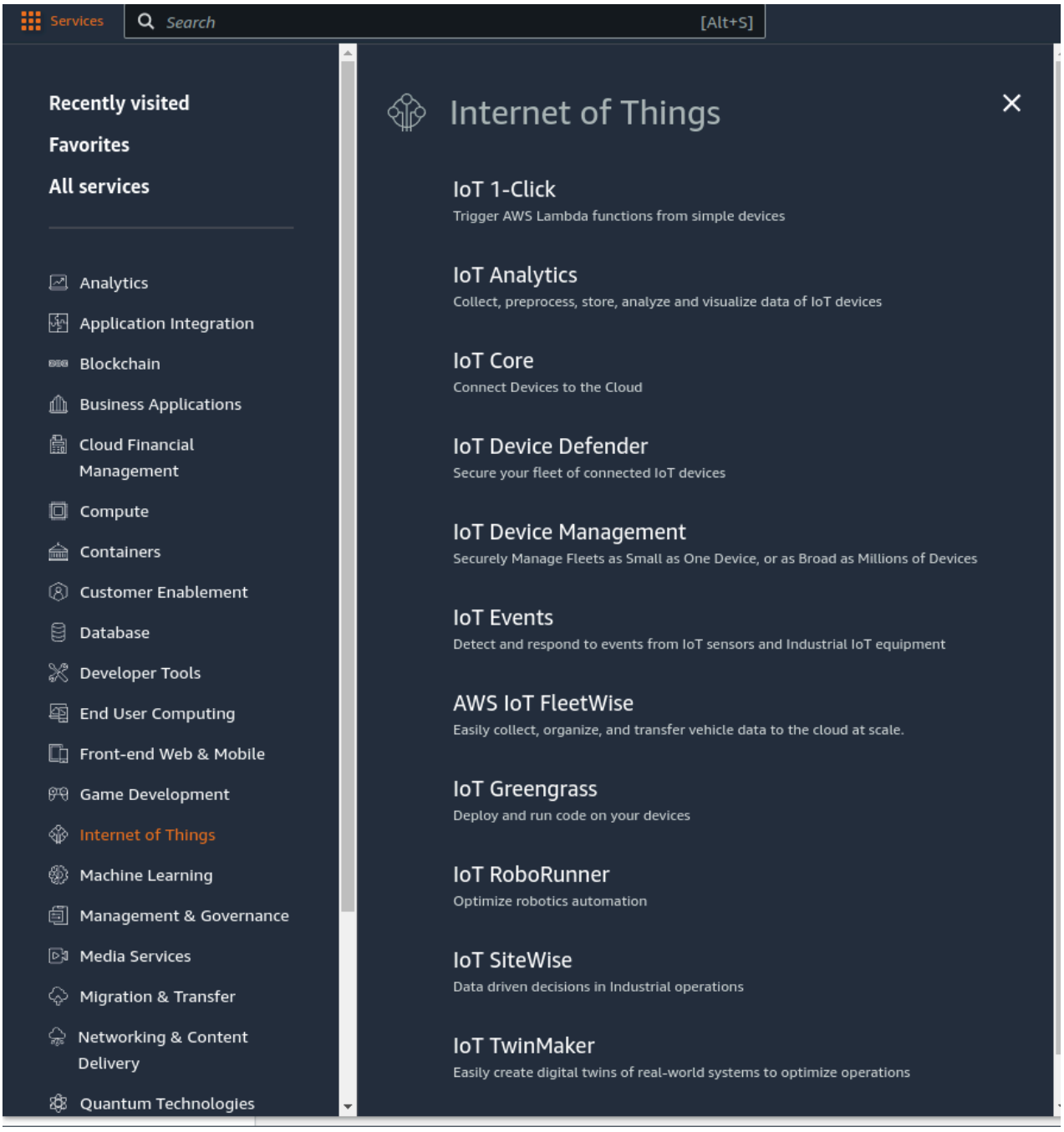
Select access key for command line based access control for user.

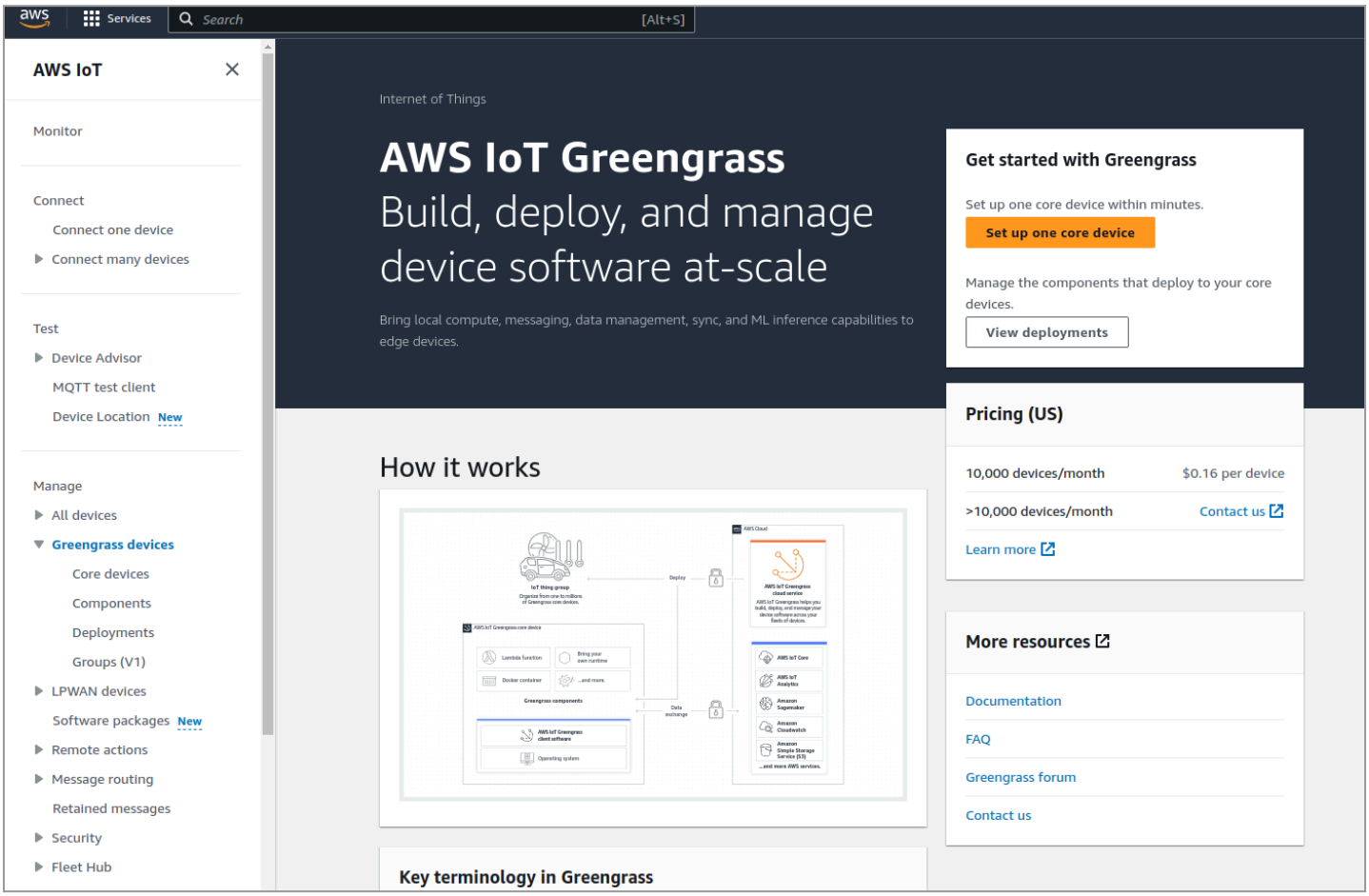



Next save the "Access Key" and "Secreat Access Key" . We will need this later while using greengrass CLI in KD240 console or downloading the csv file.

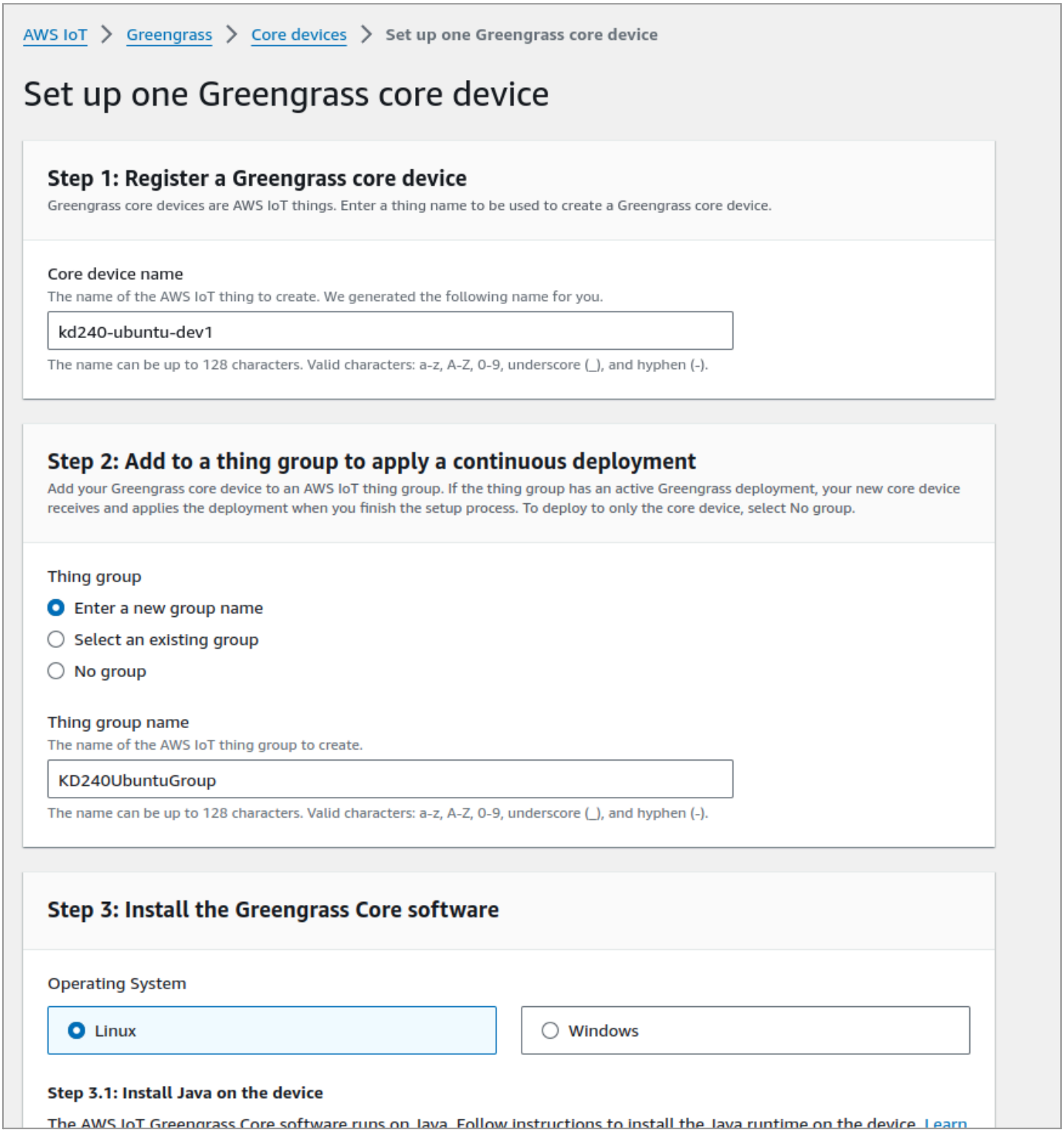
# Installing Greengrass CLI

Steps and scripts for installing greengrass device is provided by AWS Greengrass dashboard in AWS web console. So first access the AWS Greengrass IoT page, go to AWS Services -> Internet of Things -> IoT Greengrass link






Now click on “Set up one core device” button
This will open the Greengrass core device setup page:
Here you change the Core device name like `kd240-ubuntu-dev1”
And set Group for KD240 group device.




Now in KD240 terminal console run following commands and scripts:

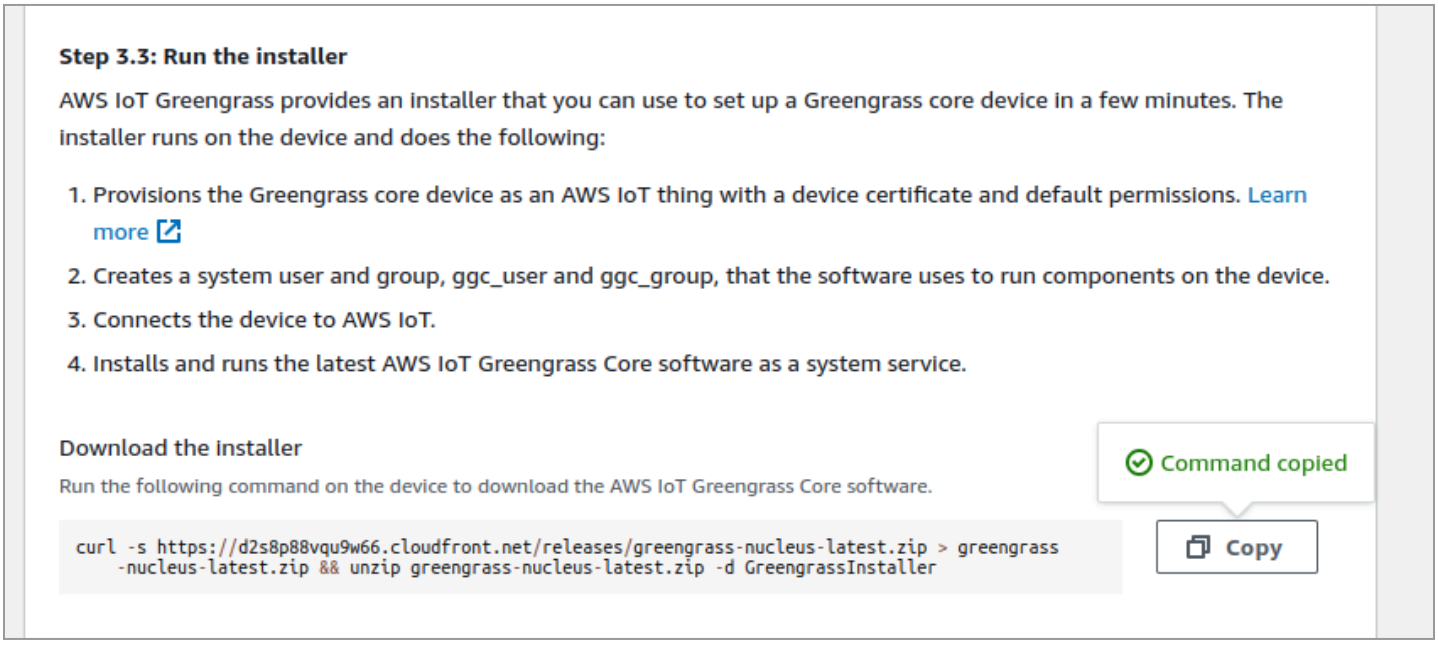
```
export AWS_ACCESS_KEY_ID=<AWS_ACCESS_KEY_ID>
export AWS_SECRET_ACCESS_KEY=<AWS_SECRET_ACCESS_KEY>
```

Greengrass CLI depends on Java. So to install the dependency run the following:

```
sudo apt install default-jre
sudo apt install default-jdk
sudo apt install unzip
```

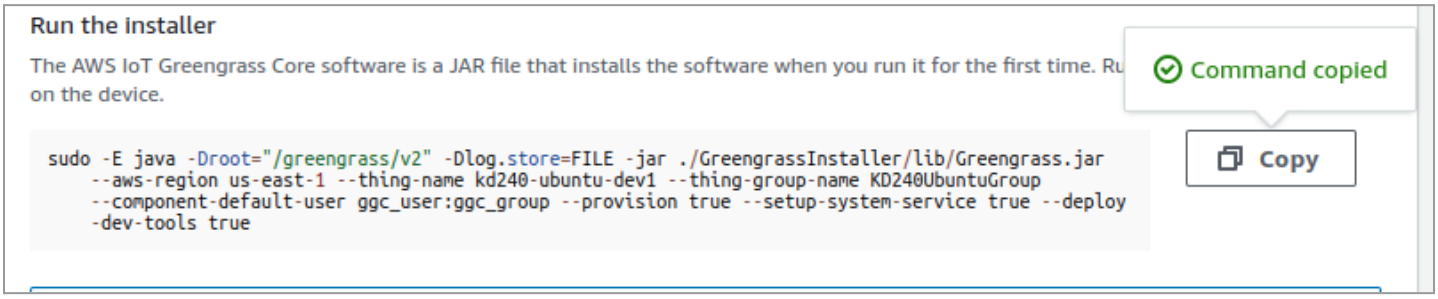
Download and install Greengrass core software as instructed in AWS setup step 3.

```
curl -s https://d2s8p88vqu9w66.cloudfront.net/releases/greengrass-nucleus-latest.zip > greengrass-nucleus-latest.zip && unzip greengrass-nucleus-latest.zip -d GreengrassInstaller
```




Next install the Greengrass core device as instructed in AWS setup step 3.:

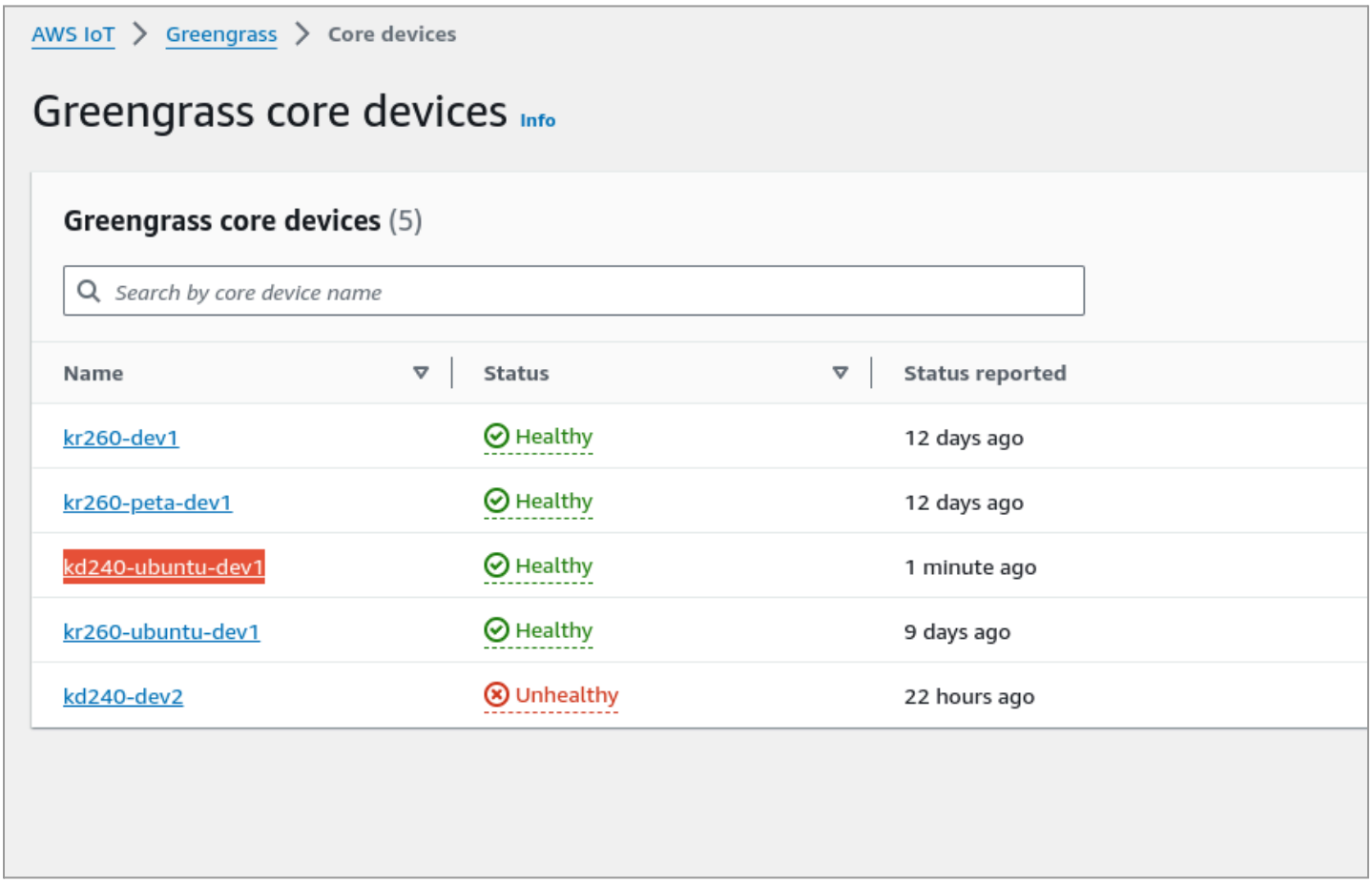
```
sudo -E java -Droot="/greengrass/v2" -Dlog.store=FILE -jar ./GreengrassInstaller/lib/Greengrass.jar --aws-region us-east-1 --thing-name kd240-ubuntu-dev1 --thing-group-name KD240UbuntuGroup --component-default-user ggc_user:ggc_group --provision true --setup-system-service true --deploy-dev-tools true
```



Here is the console log after running above command:

```
Creating group ggc_group
ggc_group created
Added ggc_user to ggc_group
Provisioning AWS IoT resources for the device with IoT Thing Name: [kd240-ubuntu-dev1]...
Found IoT policy "GreengrassV2IoTThingPolicy", reusing it
Creating keys and certificate...
Attaching policy to certificate...
Creating IoT Thing "kd240-ubuntu-dev1"...
Attaching certificate to IoT thing...
Successfully provisioned AWS IoT resources for the device with IoT Thing Name: [kd240-ubuntu-dev1]!
Adding IoT Thing [kd240-ubuntu-dev1] into Thing Group: [KD240UbuntuGroup]...
Successfully added Thing into Thing Group: [KD240UbuntuGroup]
Setting up resources for aws.greengrass.TokenExchangeService ...
Attaching TES role policy to IoT thing...
No managed IAM policy found, looking for user defined policy...
IAM policy named "GreengrassV2TokenExchangeRoleAccess" already exists. Please attach it to the IAM role if not already
Configuring Nucleus with provisioned resource details...
Downloading Root CA from "https://www.amazontrust.com/repository/AmazonRootCA1.pem"
Created device configuration
Successfully configured Nucleus with provisioned resource details!
Creating a deployment for Greengrass first party components to the thing group
Configured Nucleus to deploy aws.greengrass.Cli component
Successfully set up Nucleus as a system service
ubuntu@kria:~$
```

Now in Greengrass set up page, one can view the Greengrass core devices and find above `kd240-ubuntu-dev` in the list.



In KD240 terminal one can get the device components by using `greengrass-cli`:

```
sudo /greengrass/v2/bin/greengrass-cli component list
```

```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli component list
Components currently running in Greengrass:
Component Name: aws.greengrass.Nucleus
    Version: 2.12.1
    State: FINISHED
    Configuration: {"awsRegion":"us-east-1","componentStoreMaxSizeBytes":"10000000000","deploymentP
t":"","greengrassDataPlanePort":"8443","httpClient":{},"iotCredEndpoint":"c1uwyavs4wpvxg.credential
ions":"-Dlog.store=FILE","logging":{},"mqtt":{"spooler":{}},"networkProxy":{"proxy":{}},"platformOv
Component Name: DeploymentService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: UpdateSystemPolicyService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: FleetStatusService
    Version: null
    State: RUNNING
    Configuration: null
Component Name: TelemetryAgent
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: aws.greengrass.Cli
    Version: 2.12.1
    State: RUNNING
    Configuration: {"AuthorizedPosixGroups":null,"AuthorizedWindowsGroups":null}
ubuntu@kria:~$
```

Next we will be adding component to publish and subscribe the topic to the AWS cloud Broker.

# Installing the component

Get the `components` folder and copy in the KD240 home directory.
It contains:

**Artifacts**

- com.example.mqtt
  - 1.0.0
    - mqtt.py (This python code published the data on button press and actuates gpio on receiving the data in subscribed topic)

**Recipe**

- com.example.mqtt-1.0.0.json

Before installing the component need install 'python3-pip`

```
sudo apt install python3-pip
```

To install the above component run the following in the KD240 terminal:

```
sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.mqtt=1.0.0"
```

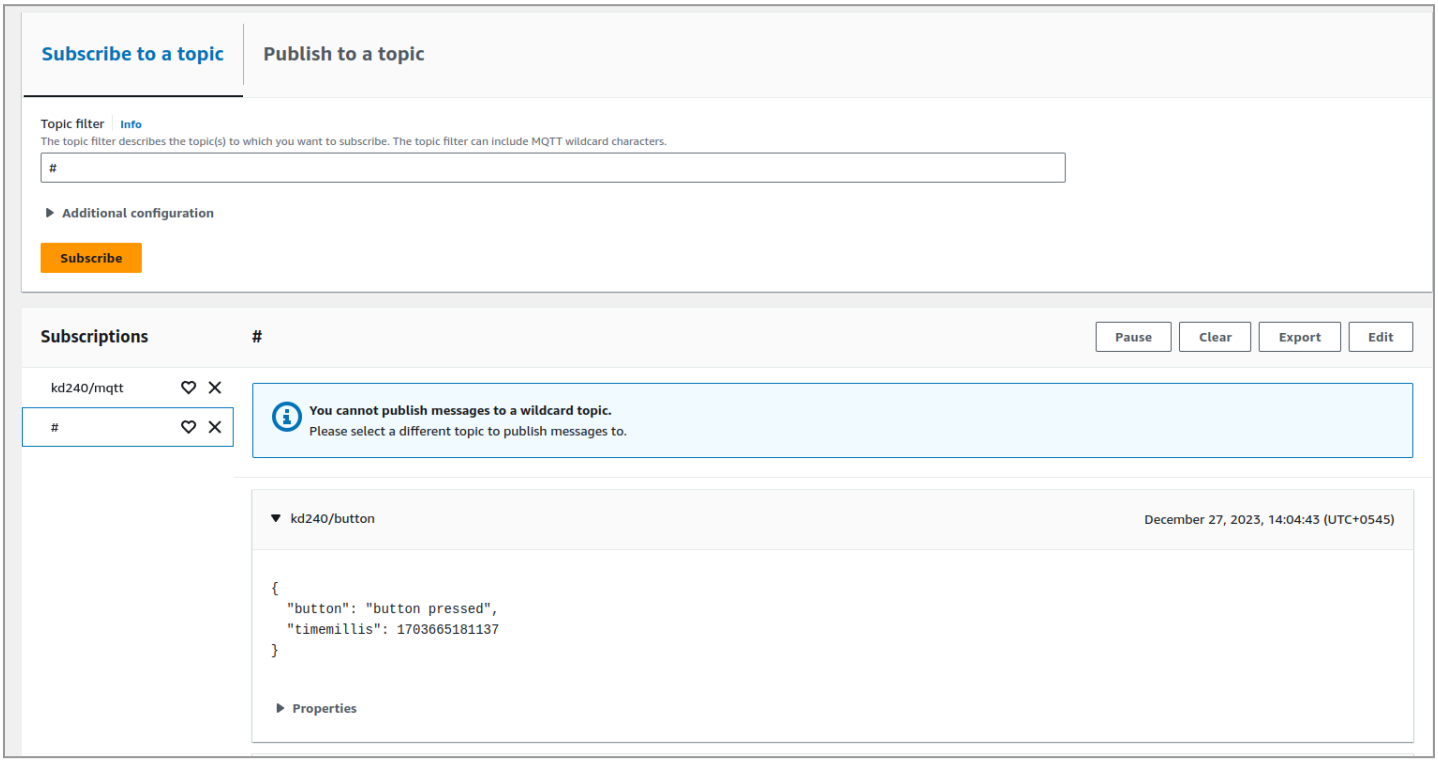
```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.mqtt=1.0.0"


Local deployment submitted! Deployment Id: 3e4cad17-9a79-4e76-bc20-58a3b5b0093f
ubuntu@kria:~$
```

Now check the installed component is in "running state":

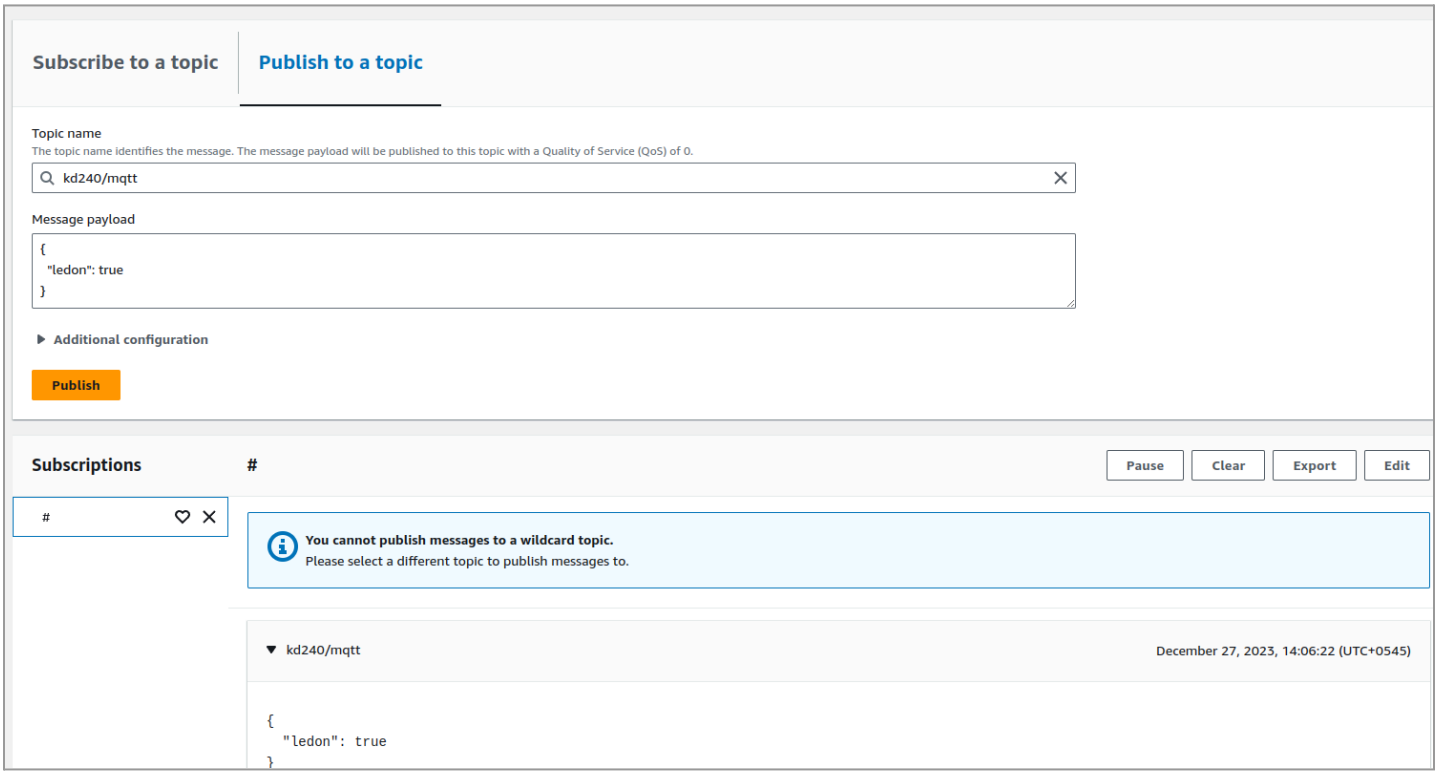
```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.mqtt=1.0.0"
Local deployment submitted! Deployment Id: 4835f786-9e25-4250-8b1c-c3acd3bc3de9
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli component list
Components currently running in Greengrass:
Component Name: aws.greengrass.Nucleus
    Version: 2.12.1
    State: FINISHED
    Configuration: {"awsRegion":"us-east-1","componentStoreMaxSizeBytes":"10000000000","deploymentPolli
t":"","greengrassDataPlanePort":"8443","httpClient":{},"iotCredEndpoint":"c1uwyavs4wpvxg.credentials.io
ions":"-Dlog.store=FILE","logging":{},"mqtt":{"spooler":{}},"networkProxy":{"proxy":{}},"platformOverri
Component Name: UpdateSystemPolicyService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: DeploymentService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: TelemetryAgent
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: FleetStatusService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: com.example.mqtt
    Version: 1.0.0
    State: BROKEN
    Configuration: {"accessControl":{"aws.greengrass.ipc.mqttproxy":{"com.example.mqtt:mqttproxy:1":{"o
ces":["kd240/mqtt","kd240/button"]}}},"message":"hello"}
Component Name: aws.greengrass.Cli
    Version: 2.12.1
    State: RUNNING
    Configuration: {"AuthorizedPosixGroups":null,"AuthorizedWindowsGroups":null}
ubuntu@kria:~$
```

Now in aws IoT console, open “MQTT test client” and subscribe to “#”

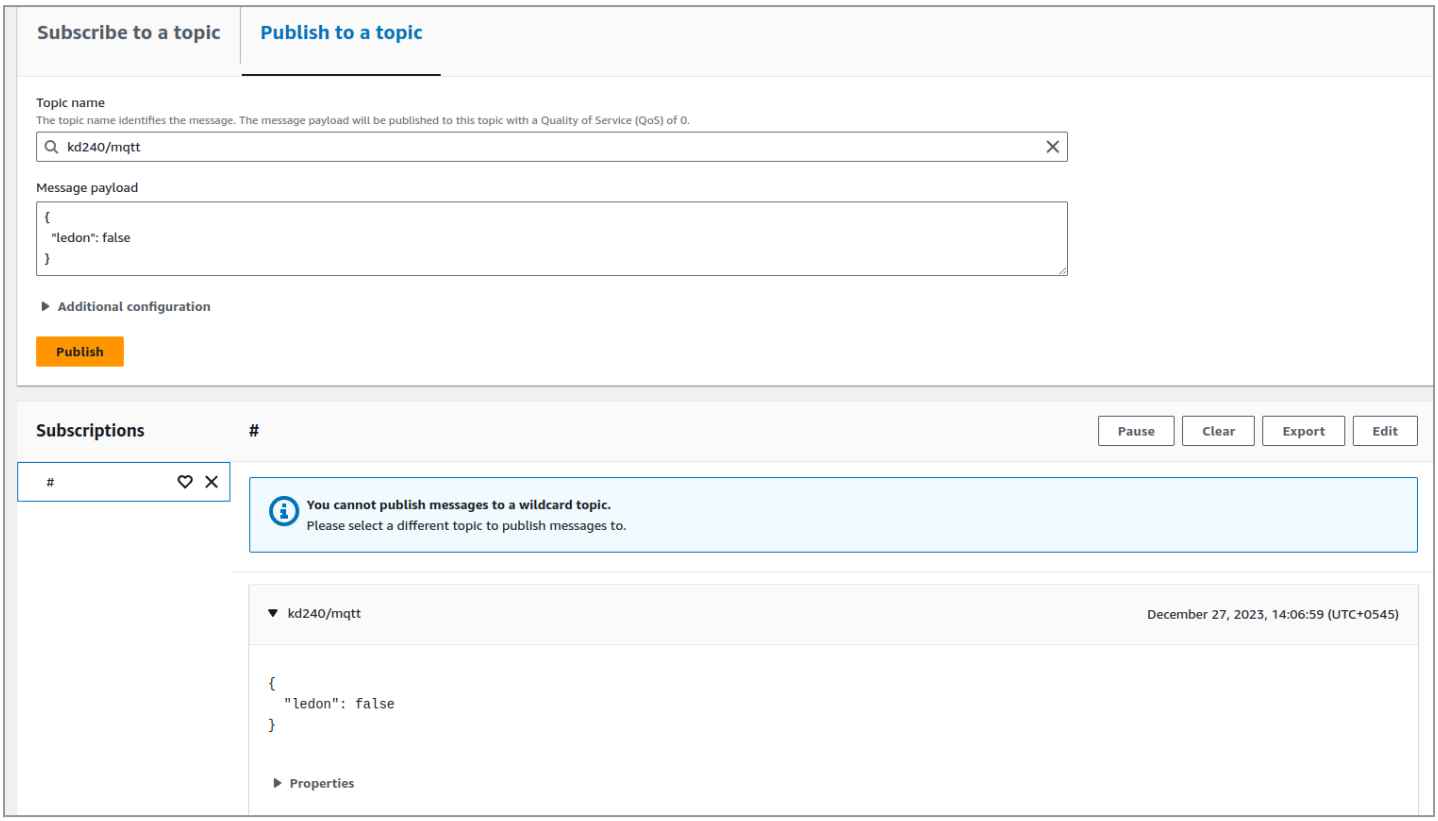



You can see the “button pressed” message once the button is pressed.

Now to control the LED, publish the message to “kd240/mqtt” topic. Here is the screenshot of the message which switch on the LED.



Now to switch off the LED send “false” message in the “kd240/mqtt” topic.




***