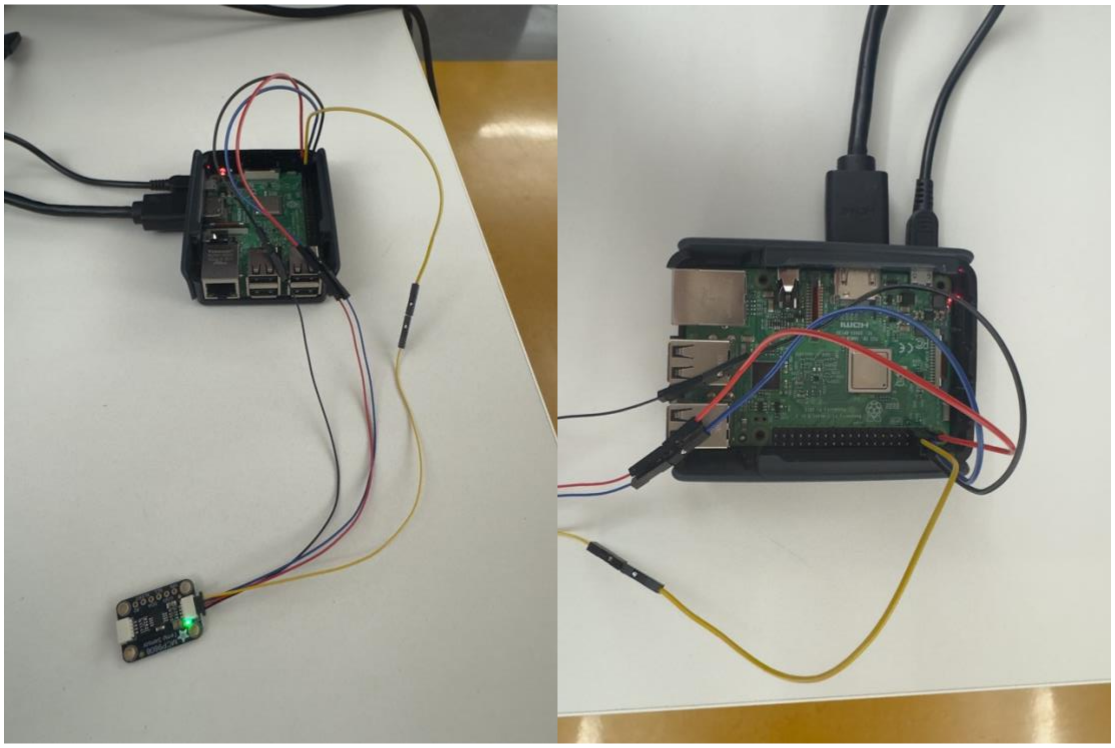
Final Project
Part 1 – Setting Up the IoT Device with AWS IoT Core

CYT160: Security for Cloud and Internet of Things
Professor Saeed Naghizadeh Qomi
November 7, 2025

Group 2
Jyotpal Singh
Rickie Rihal
Yash Sanjaybhai Patel

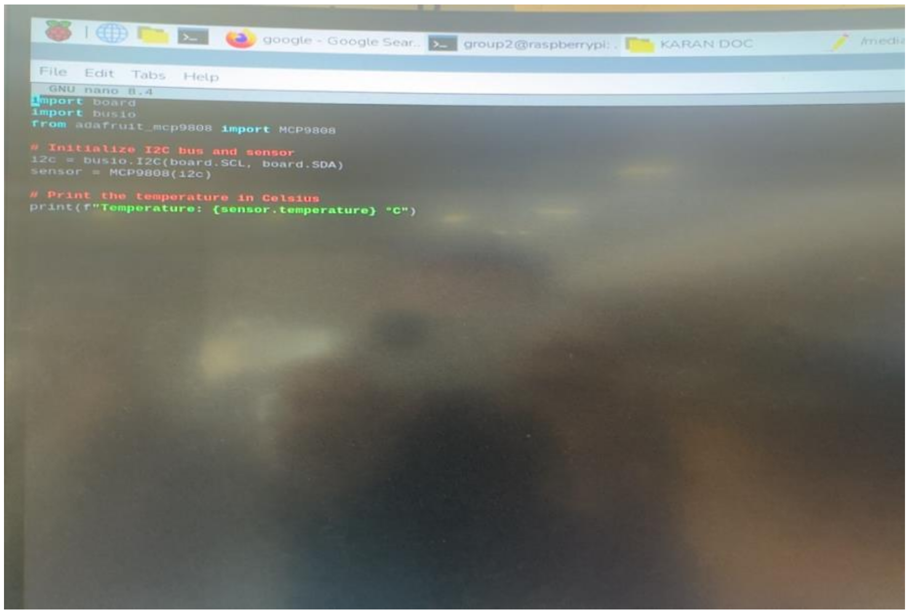
Hardware Setup

Below is an image of our hardware set up and assembly of all components. You can see that the Temperature sensor is correctly configured as we wired the red, black, blue, and yellow wires in the corresponding pinhole.



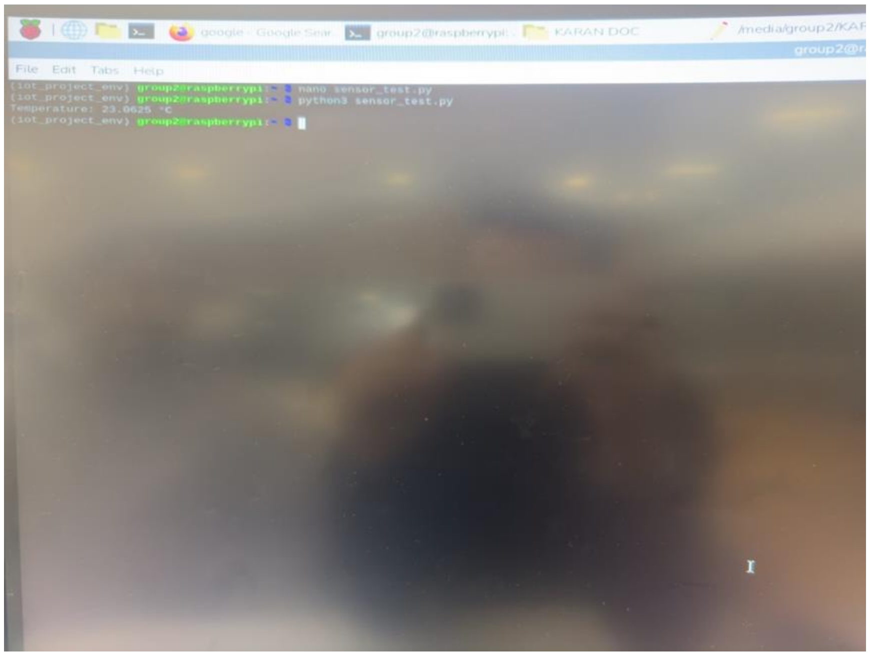
Testing the MCP9808 Temperature Sensor

We were able to test the temperature sensor by adding the following code in the test script.



```
import board
import busio
from adafruit_mcp9808 import MCP9808

# Initialize I2C bus and sensor
i2c = busio.I2C(board.SCL, board.SDA)
sensor = MCP9808(i2c)

# Print the temperature in Celsius
print(f"Temperature: {sensor.temperature} °C")
```

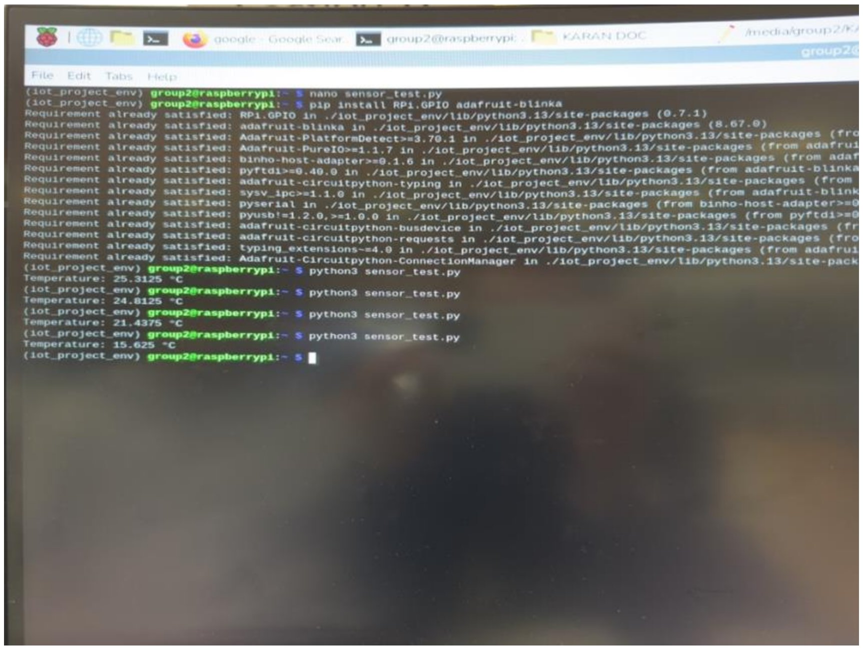
Below is a successful output of the temperature sensor indicating the temperature is 23°.



```
(iot_project_env) group2@raspberrypi:~ $ nano sensor_test.py
(iot_project_env) group2@raspberrypi:~ $ python3 sensor_test.py
Temperature: 23.0625 °C
(iot_project_env) group2@raspberrypi:~ $
```

Interpreting the Temperature Sensor Reading

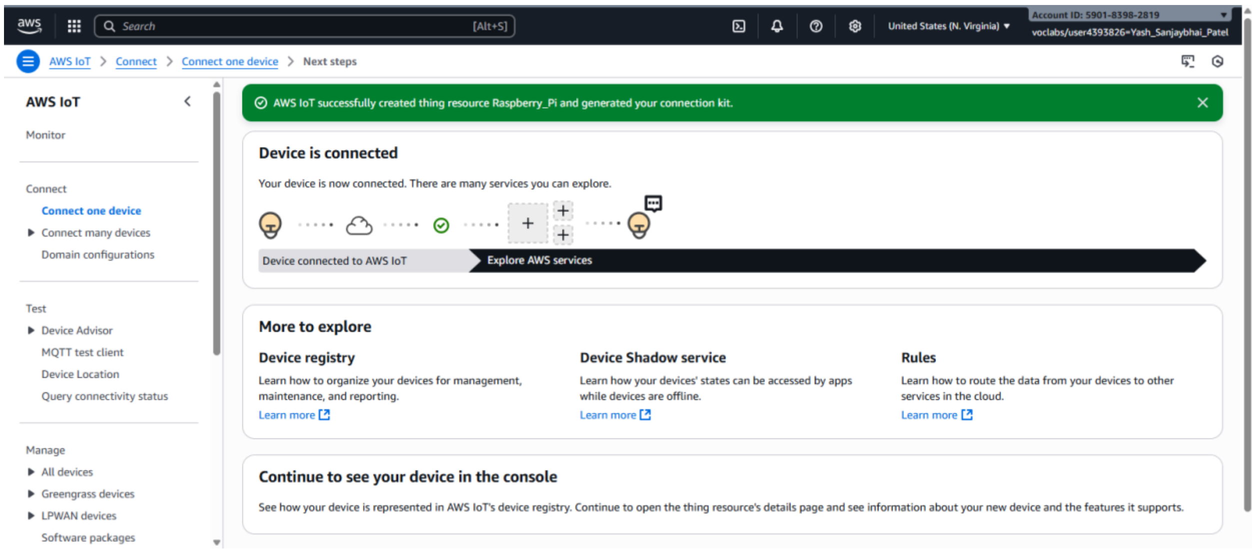
We were able to change the temperature readings by introducing the sensor to a cool environment. Below you can see the gradual change in temperature dropping from 25° to 15°.



```
(iot_project_env) group2@raspberrypi:~ $ nano sensor_test.py
(iot_project_env) group2@raspberrypi:~ $ pip install RPi.GPIO adafruit-blinka
Requirement already satisfied: RPi.GPIO in ./iot_project_env/lib/python3.13/site-packages (0.7.1)
Requirement already satisfied: adafruit-blinka in ./iot_project_env/lib/python3.13/site-packages (8.67.0)
Requirement already satisfied: Adafruit-PlatformDetect>=3.70.1 in ./iot_project_env/lib/python3.13/site-packages (fro
Requirement already satisfied: Adafruit-PureIO>=1.1.7 in ./iot_project_env/lib/python3.13/site-packages (from adafrui
Requirement already satisfied: binho-host-adapter>=0.1.6 in ./iot_project_env/lib/python3.13/site-packages (from adaf
Requirement already satisfied: pyftdi>=0.40.0 in ./iot_project_env/lib/python3.13/site-packages (from adafruit-blinka
Requirement already satisfied: adafruit-circuitpython-typing in ./iot_project_env/lib/python3.13/site-packages (from
Requirement already satisfied: sysv_ipc>=1.1.0 in ./iot_project_env/lib/python3.13/site-packages (from adafruit-blink
Requirement already satisfied: pyserial in ./iot_project_env/lib/python3.13/site-packages (from binho-host-adapter>=0
Requirement already satisfied: pyusb!=1.2.0,>=1.0.0 in ./iot_project_env/lib/python3.13/site-packages (from pyftdi>=0
Requirement already satisfied: adafruit-circuitpython-busdevice in ./iot_project_env/lib/python3.13/site-packages (fr
Requirement already satisfied: adafruit-circuitpython-requests in ./iot_project_env/lib/python3.13/site-packages (fro
Requirement already satisfied: typing_extensions~=4.0 in ./iot_project_env/lib/python3.13/site-packages (from adafrui
Requirement already satisfied: Adafruit-Circuitpython-ConnectionManager in ./iot_project_env/lib/python3.13/site-pack
(iot_project_env) group2@raspberrypi:~ $ python3 sensor_test.py
Temperature: 25.3125 °C
(iot_project_env) group2@raspberrypi:~ $ python3 sensor_test.py
Temperature: 24.8125 °C
(iot_project_env) group2@raspberrypi:~ $ python3 sensor_test.py
Temperature: 21.4375 °C
(iot_project_env) group2@raspberrypi:~ $ python3 sensor_test.py
Temperature: 15.625 °C
(iot_project_env) group2@raspberrypi:~ $
```

Set up AWS IoT Core and Acquire Required Keys, ID, and Other Information

We signed into AWS IoT Core to establish a connection and communication between the Raspberry PI and AWS. We successfully connected the device and were able to download the connection kit.



AWS IoT successfully created thing resource Raspberry_Pi and generated your connection kit.

Device is connected

Your device is now connected. There are many services you can explore.

More to explore

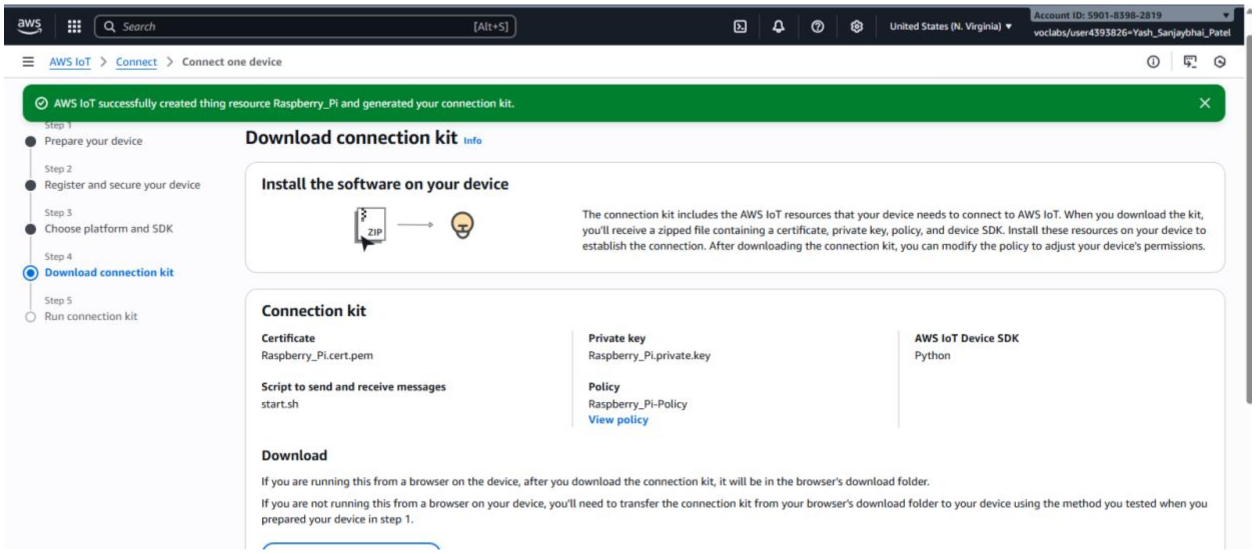
Device registry
Learn how to organize your devices for management, maintenance, and reporting.
Learn more

Device Shadow service
Learn how your devices' states can be accessed by apps while devices are offline.
Learn more

Rules
Learn how to route the data from your devices to other services in the cloud.
Learn more

Continue to see your device in the console

See how your device is represented in AWS IoT's device registry. Continue to open the thing resource's details page and see information about your new device and the features it supports.


Search
[Alt+S]
United States (N. Virginia)
Account ID: 5901-8398-2819
voclabs/user4393826=Yash_Sanjaybhai_Patel
AWS IoT > Connect > Connect one device
Step 1
Prepare your device
Step 2
Register and secure your device
Step 3
Choose platform and SDK
Step 4
Download connection kit
Step 5
Run connection kit
ZIP

AWS IoT successfully created thing resource Raspberry_Pi and generated your connection kit.

Download connection kit Info

Install the software on your device

The connection kit includes the AWS IoT resources that your device needs to connect to AWS IoT. When you download the kit, you'll receive a zipped file containing a certificate, private key, policy, and device SDK. Install these resources on your device to establish the connection. After downloading the connection kit, you can modify the policy to adjust your device's permissions.

Connection kit

Certificate
Raspberry_Pi.cert.pem

Script to send and receive messages
start.sh

Private key
Raspberry_Pi.private.key

Policy
Raspberry_Pi-Policy
View policy

AWS IoT Device SDK
Python

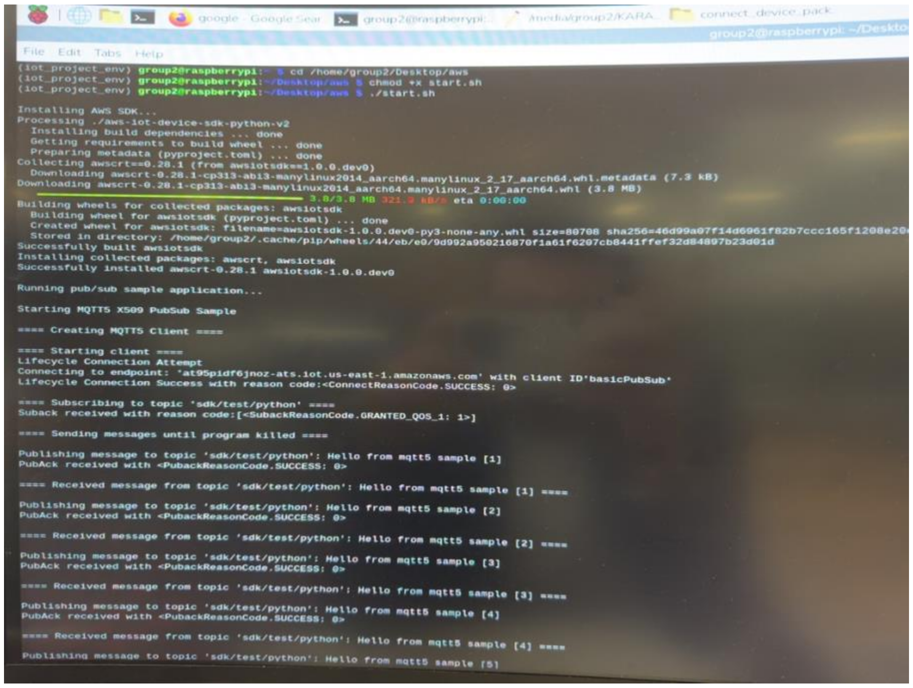
Download

If you are running this from a browser on the device, after you download the connection kit, it will be in the browser's download folder.

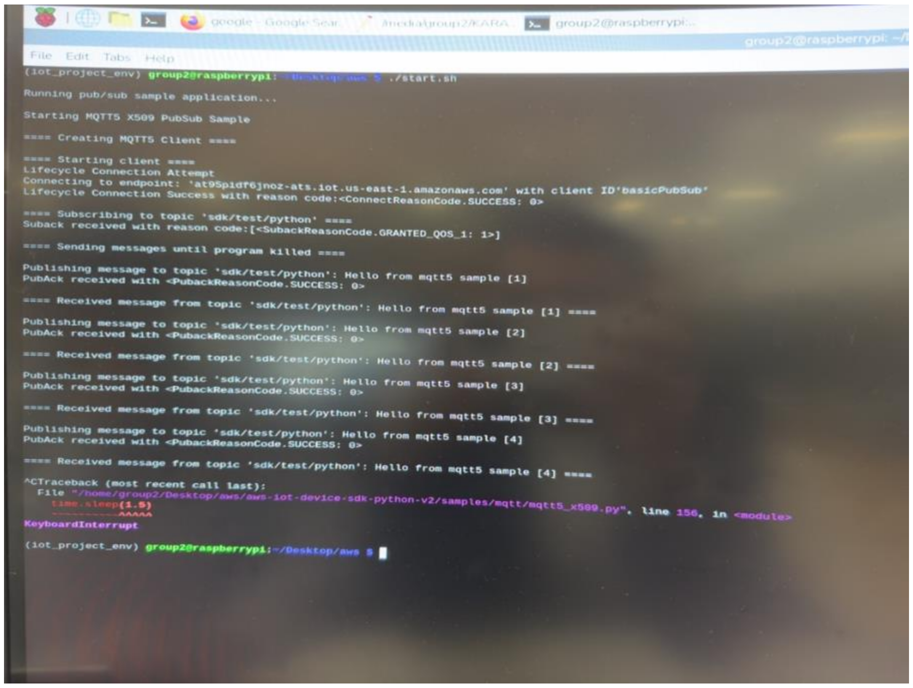
If you are not running this from a browser on your device, you'll need to transfer the connection kit from your browser's download folder to your device using the method you tested when you prepared your device in step 1.

Run the start.sh Script and View Messages from your Device

To ensure that communication was occurring between AWS and our Raspberry PI we executed ***chmod +x start.sh*** and then ran ***./start.sh***.



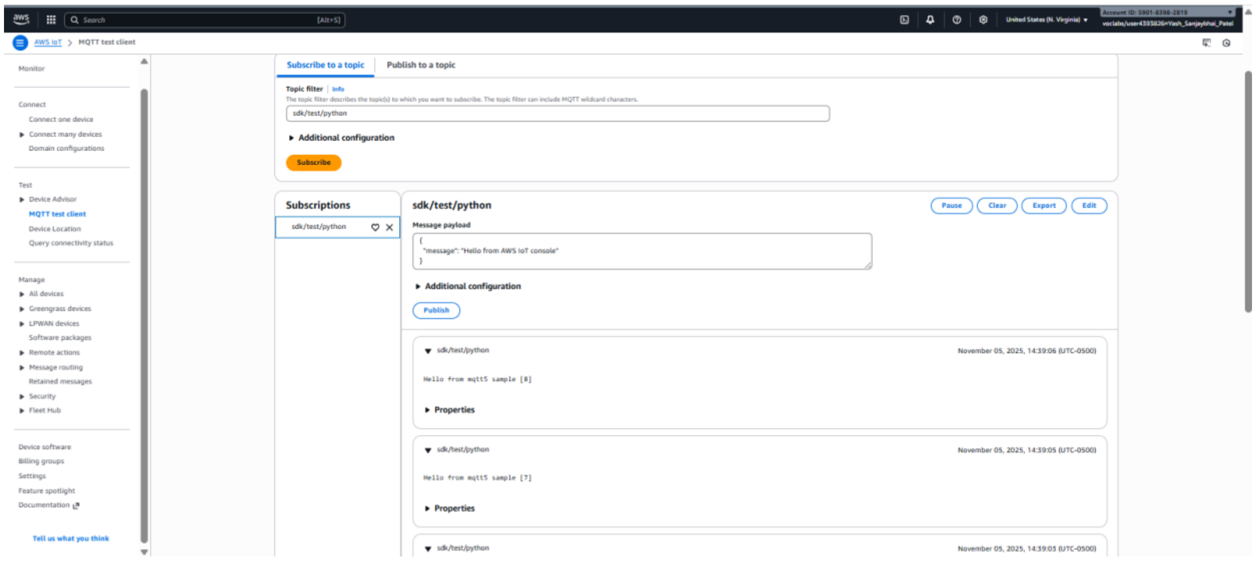
```
(iot_project_env) group2@raspberrypi:~ $ cd /home/group2/Desktop/aws
(iot_project_env) group2@raspberrypi:~/Desktop/aws $ chmod +x start.sh
(iot_project_env) group2@raspberrypi:~/Desktop/aws $ ./start.sh

Installing AWS SDK...
Processing ./aws-iot-device-sdk-python-v2
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting awscrt==0.28.1 (from awsiotsdk==1.0.0.dev0)
  Downloading awscrt-0.28.1-cp313-abi3-manylinux2014_aarch64.manylinux_2_17_aarch64.whl.metadata (7.3 kB)
Downloading awscrt-0.28.1-cp313-abi3-manylinux2014_aarch64.manylinux_2_17_aarch64.whl (3.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.8/3.8 MB 321.9 kB/s eta 0:00:00
Building wheels for collected packages: awsiotsdk
  Building wheel for awsiotsdk (pyproject.toml) ... done
  Created wheel for awsiotsdk: filename=awsiotsdk-1.0.0.dev0-py3-none-any.whl size=80708 sha256=46d99a07f14d6961f82b7ccc165f1208e20
  Stored in directory: /home/group2/.cache/pip/wheels/44/eb/e0/9d992a950216870f1a61f6207cb8441ffef32d84897b23d01d
Successfully built awsiotsdk
Installing collected packages: awscrt, awsiotsdk
Successfully installed awscrt-0.28.1 awsiotsdk-1.0.0.dev0

Running pub/sub sample application...

Starting MQTT5 X509 PubSub Sample

==== Creating MQTT5 Client ====

==== Starting client ====
Lifecycle Connection Attempt
Connecting to endpoint: 'at95pidf6jnoz-ats.iot.us-east-1.amazonaws.com' with client ID'basicPubSub'
Lifecycle Connection Success with reason code:<ConnectReasonCode.SUCCESS: 0>

==== Subscribing to topic 'sdk/test/python' ====
Suback received with reason code:[<SubackReasonCode.GRANTED_QOS_1: 1>]

==== Sending messages until program killed ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [1]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [1] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [2]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [2] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [3]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [3] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [4]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [4] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [5]
```



```
(iot_project_env) group2@raspberrypi:~/Desktop/aws $ ./start.sh

Running pub/sub sample application...

Starting MQTT5 X509 PubSub Sample

==== Creating MQTT5 Client ====

==== Starting client ====
Lifecycle Connection Attempt
Connecting to endpoint: 'at95pidf6jnoz-ats.iot.us-east-1.amazonaws.com' with client ID'basicPubSub'
Lifecycle Connection Success with reason code:<ConnectReasonCode.SUCCESS: 0>

==== Subscribing to topic 'sdk/test/python' ====
Suback received with reason code:[<SubackReasonCode.GRANTED_QOS_1: 1>]

==== Sending messages until program killed ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [1]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [1] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [2]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [2] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [3]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [3] ====

Publishing message to topic 'sdk/test/python': Hello from mqtt5 sample [4]
PubAck received with <PubackReasonCode.SUCCESS: 0>

==== Received message from topic 'sdk/test/python': Hello from mqtt5 sample [4] ====
^CTraceback (most recent call last):
  File "/home/group2/Desktop/aws/aws-iot-device-sdk-python-v2/samples/mqtt/mqtt5_x509.py", line 156, in <module>
    time.sleep(1.5)
    ~~~~~~~~~~^^^^^
KeyboardInterrupt

(iot_project_env) group2@raspberrypi:~/Desktop/aws $ 
```

Received messages in the AWS IoT console

We successfully received communication from the Raspberry Pi onto AWS confirming that the configuration worked.



Account ID: 5901-8398-2819
voclabs/user4393826=Yash_Sanjaybhai_Patel

Topic filter
The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.
sdk/test/python

Subscriptions: sdk/test/python

sdk/test/python

Message payload

```
{
  "message": "Hello from AWS IoT console"
}
```

sdk/test/python — November 05, 2025, 14:39:06 (UTC-0500)

```
Hello from mqtt5 sample [8]
```

sdk/test/python — November 05, 2025, 14:39:05 (UTC-0500)

```
Hello from mqtt5 sample [7]
```

sdk/test/python — November 05, 2025, 14:39:03 (UTC-0500)