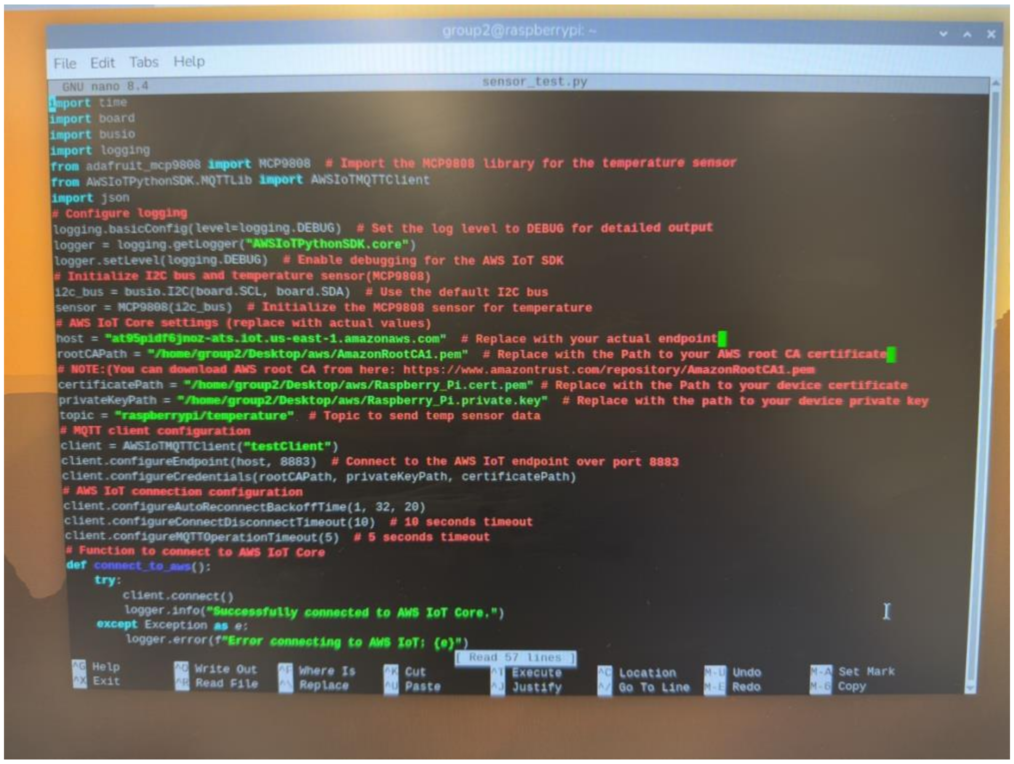
Final Project
Part 2 – Create the Python Script for Tempsensor

CYT160: Security for Cloud and Internet of Things
Professor Saeed Naghizadeh Qomi
November 14, 2025

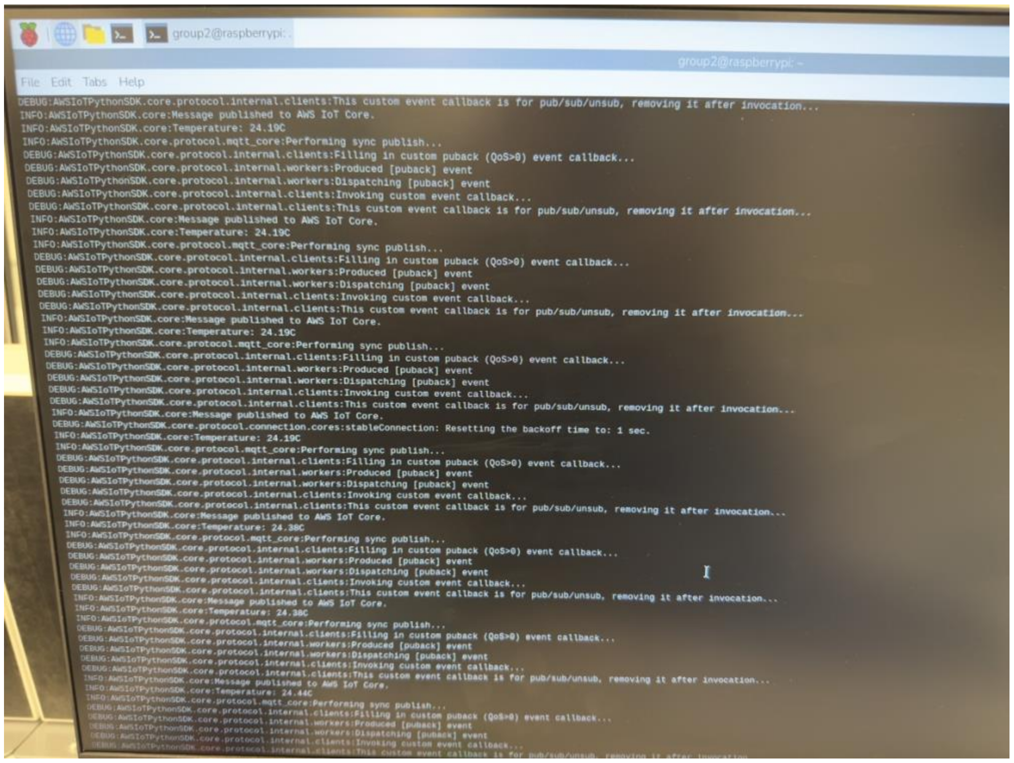
Group 2
Jyotpal Singh
Rickie Rihal
Yash Sanjaybhai Patel

Modified Existing Python Code Policy Permissions

Below is a screenshot of our modified policy permissions. This modification allows us to send continuous temperature data to the AWS IoT core.



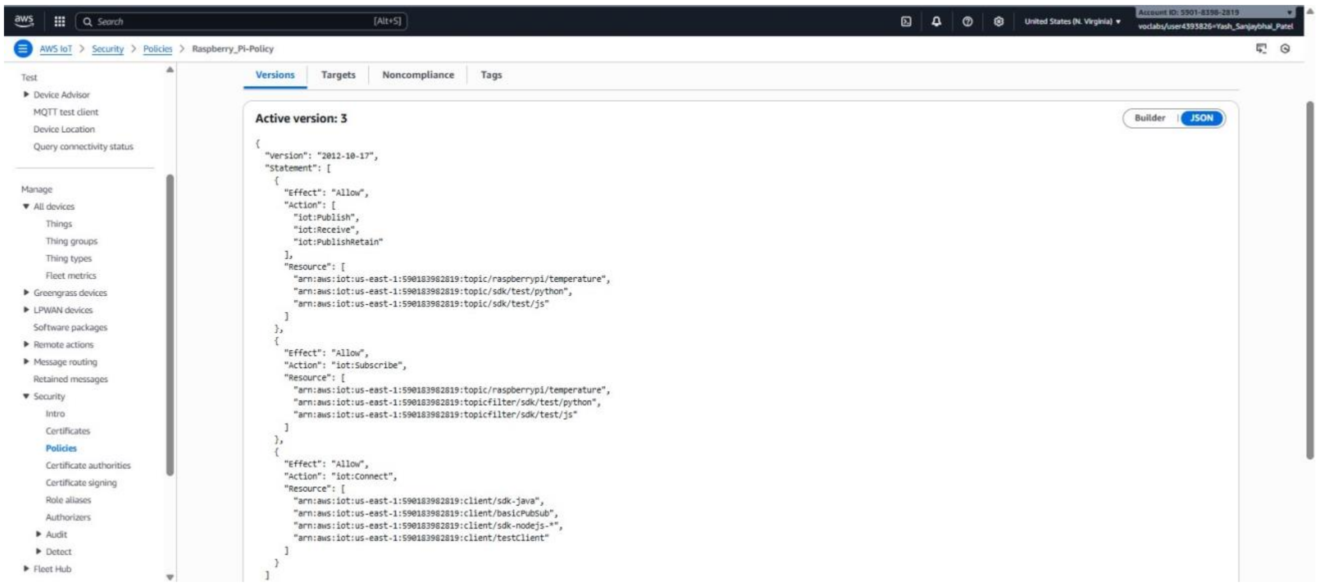
```
GNU nano 8.4                              sensor_test.py
import time
import board
import busio
import logging
from adafruit_mcp9808 import MCP9808  # Import the MCP9808 library for the temperature sensor
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
import json
# Configure logging
logging.basicConfig(level=logging.DEBUG)  # Set the log level to DEBUG for detailed output
logger = logging.getLogger("AWSIoTPythonSDK.core")
logger.setLevel(logging.DEBUG)  # Enable debugging for the AWS IoT SDK
# Initialize I2C bus and temperature sensor(MCP9808)
i2c_bus = busio.I2C(board.SCL, board.SDA)  # Use the default I2C bus
sensor = MCP9808(i2c_bus)  # Initialize the MCP9808 sensor for temperature
# AWS IoT Core settings (replace with actual values)
host = "at95pidf6jnoz-ats.iot.us-east-1.amazonaws.com"  # Replace with your actual endpoint
rootCAPath = "/home/group2/Desktop/aws/AmazonRootCA1.pem"  # Replace with the Path to your AWS root CA certificate
# NOTE:(You can download AWS root CA from here: https://www.amazontrust.com/repository/AmazonRootCA1.pem
certificatePath = "/home/group2/Desktop/aws/Raspberry_Pi.cert.pem" # Replace with the Path to your device certificate
privateKeyPath = "/home/group2/Desktop/aws/Raspberry_Pi.private.key"  # Replace with the path to your device private key
topic = "raspberrypi/temperature"  # Topic to send temp sensor data
# MQTT client configuration
client = AWSIoTMQTTClient("testClient")
client.configureEndpoint(host, 8883)  # Connect to the AWS IoT endpoint over port 8883
client.configureCredentials(rootCAPath, privateKeyPath, certificatePath)
# AWS IoT connection configuration
client.configureAutoReconnectBackoffTime(1, 32, 20)
client.configureConnectDisconnectTimeout(10)  # 10 seconds timeout
client.configureMQTTOperationTimeout(5)  # 5 seconds timeout
# Function to connect to AWS IoT Core
def connect_to_aws():
    try:
        client.connect()
        logger.info("Successfully connected to AWS IoT Core.")
    except Exception as e:
        logger.error(f"Error connecting to AWS IoT: {e}")
                                   [ Read 57 lines ]
^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy
```



```
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.19C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.19C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.19C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
DEBUG:AWSIoTPythonSDK.core.protocol.connection.cores:stableConnection: Resetting the backoff time to: 1 sec.
INFO:AWSIoTPythonSDK.core:Temperature: 24.19C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.38C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.38C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation...
INFO:AWSIoTPythonSDK.core:Message published to AWS IoT Core.
INFO:AWSIoTPythonSDK.core:Temperature: 24.44C
INFO:AWSIoTPythonSDK.core.protocol.mqtt_core:Performing sync publish...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Filling in custom puback (QoS>0) event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Produced [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.workers:Dispatching [puback] event
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:Invoking custom event callback...
DEBUG:AWSIoTPythonSDK.core.protocol.internal.clients:This custom event callback is for pub/sub/unsub, removing it after invocation
```

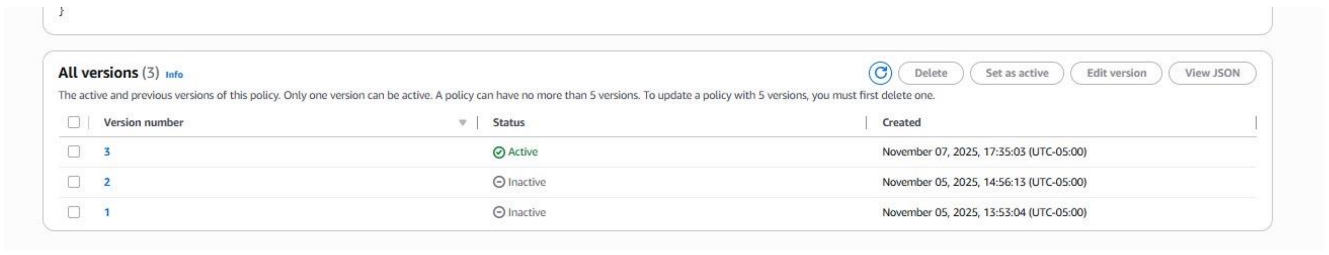
Modified AWS IoT Policy Permissions

The screenshots below display the changes we made to the AWS IoT policy. We were able to configure the JSON file and establish a “connected” status allowing us to see the temperature message in the console.



Active version: 3

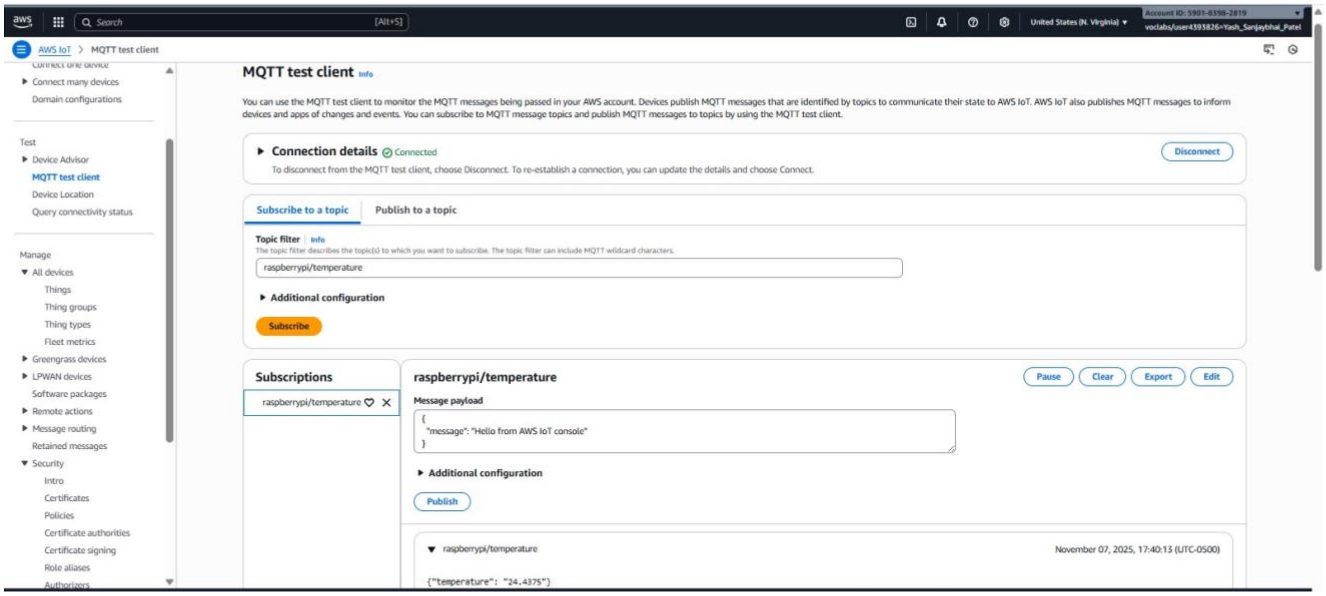
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive",
        "iot:PublishRetain"
      ],
      "Resource": [
        "arn:aws:iot:us-east-1:590183982819:topic/raspberrypi/temperature",
        "arn:aws:iot:us-east-1:590183982819:topic/sdk/test/python",
        "arn:aws:iot:us-east-1:590183982819:topic/sdk/test/js"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": [
        "arn:aws:iot:us-east-1:590183982819:topic/raspberrypi/temperature",
        "arn:aws:iot:us-east-1:590183982819:topicfilter/sdk/test/python",
        "arn:aws:iot:us-east-1:590183982819:topicfilter/sdk/test/js"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": [
        "arn:aws:iot:us-east-1:590183982819:client/sdk-java",
        "arn:aws:iot:us-east-1:590183982819:client/basicPubSub",
        "arn:aws:iot:us-east-1:590183982819:client/sdk-nodejs-*",
        "arn:aws:iot:us-east-1:590183982819:client/testClient"
      ]
    }
  ]
}
```



All versions (3) Info

The active and previous versions of this policy. Only one version can be active. A policy can have no more than 5 versions. To update a policy with 5 versions, you must first delete one.

Version number	Status	Created
3	Active	November 07, 2025, 17:35:03 (UTC-05:00)
2	Inactive	November 05, 2025, 14:56:13 (UTC-05:00)
1	Inactive	November 05, 2025, 13:53:04 (UTC-05:00)



MQTT test client Info

You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topics to communicate their state to AWS IoT. AWS IoT also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages to topics by using the MQTT test client.

Connection details Connected

To disconnect from the MQTT test client, choose Disconnect. To re-establish a connection, you can update the details and choose Connect.

Topic filter: raspberrypi/temperature

Subscriptions: raspberrypi/temperature

raspberrypi/temperature

Message payload

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

raspberrypi/temperature November 07, 2025, 17:40:13 (UTC-0500)

```
{"temperature": "24.4375"}
```