Building an IoT Data-to-Dashboard Solution with AWS IoT Core, Timestream, and Grafana

Table of Contents

---

1. Introduction

This document outlines a step-by-step guide on creating an IoT system using AWS services to collect, store, and visualize real-time data. By leveraging AWS IoT Core, Amazon Timestream, and Amazon Managed Grafana, we will demonstrate how to create an end-to-end IoT data solution that is both scalable and secure.

High-Level Overview

1. Create an **AWS IoT Core Thing** (a virtual representation of your IoT device).
2. **Send telemetry data** from the IoT device (laptop) to AWS IoT Core using the AWS IoT Python SDK.
3. **Store telemetry data** in **Amazon Timestream**.
4. **Visualize the data** in **Amazon Managed Grafana** using time-series dashboards.

2. Prerequisites

To follow along with this guide, you will need:

- **Device with Python installed**: Your laptop will act as the IoT device.
- **AWS Account**: Make sure you have an active AWS account with permissions to use AWS IoT Core, Amazon Timestream, and Amazon Managed Grafana.
- **Basic understanding of AWS services and Python**.

5.

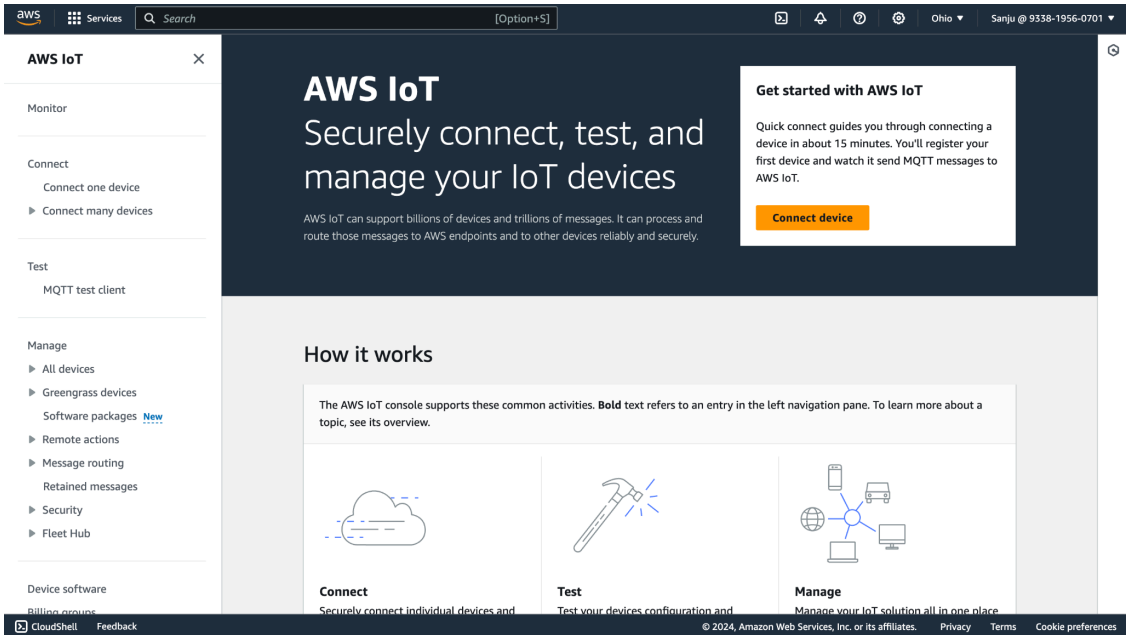
3. Step-by-Step Implementation

Open AWS Console:

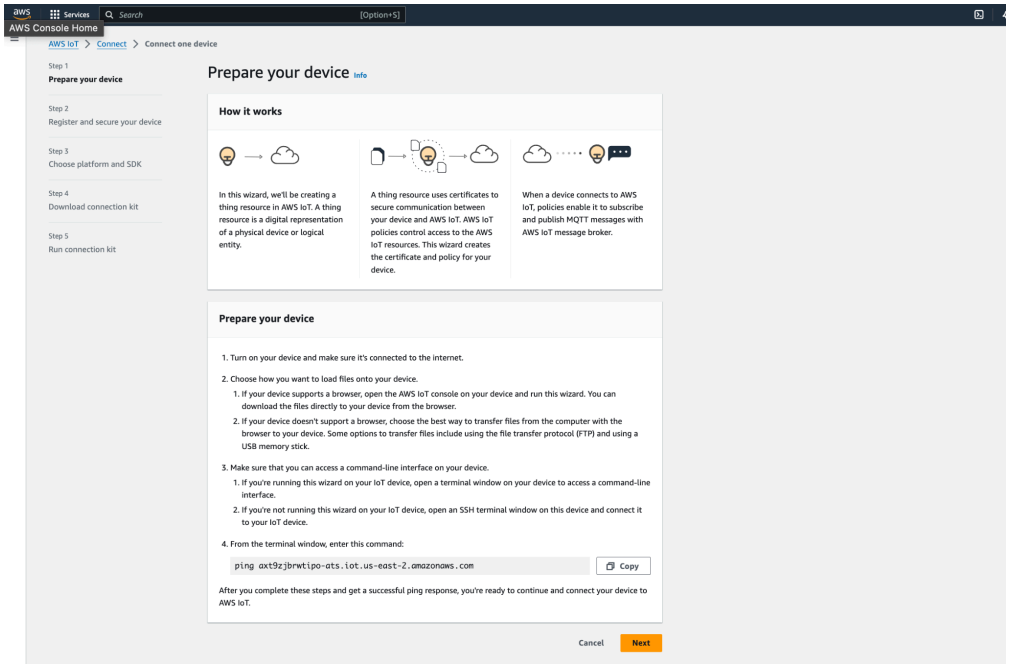
- Go to the AWS Console and sign in with your AWS credentials.
- Search for AWS IoT Core and open it.

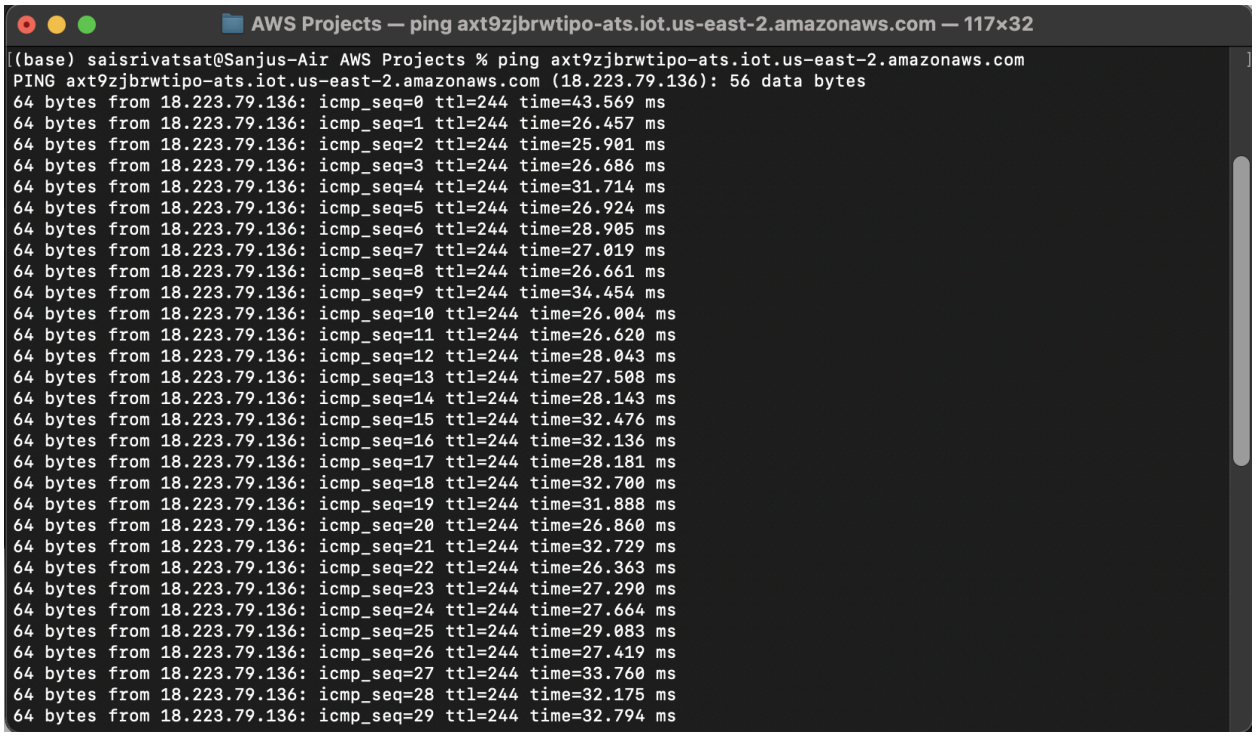
Creating an AWS IoT Thing

- Log in to the AWS and search for IoT Core console.

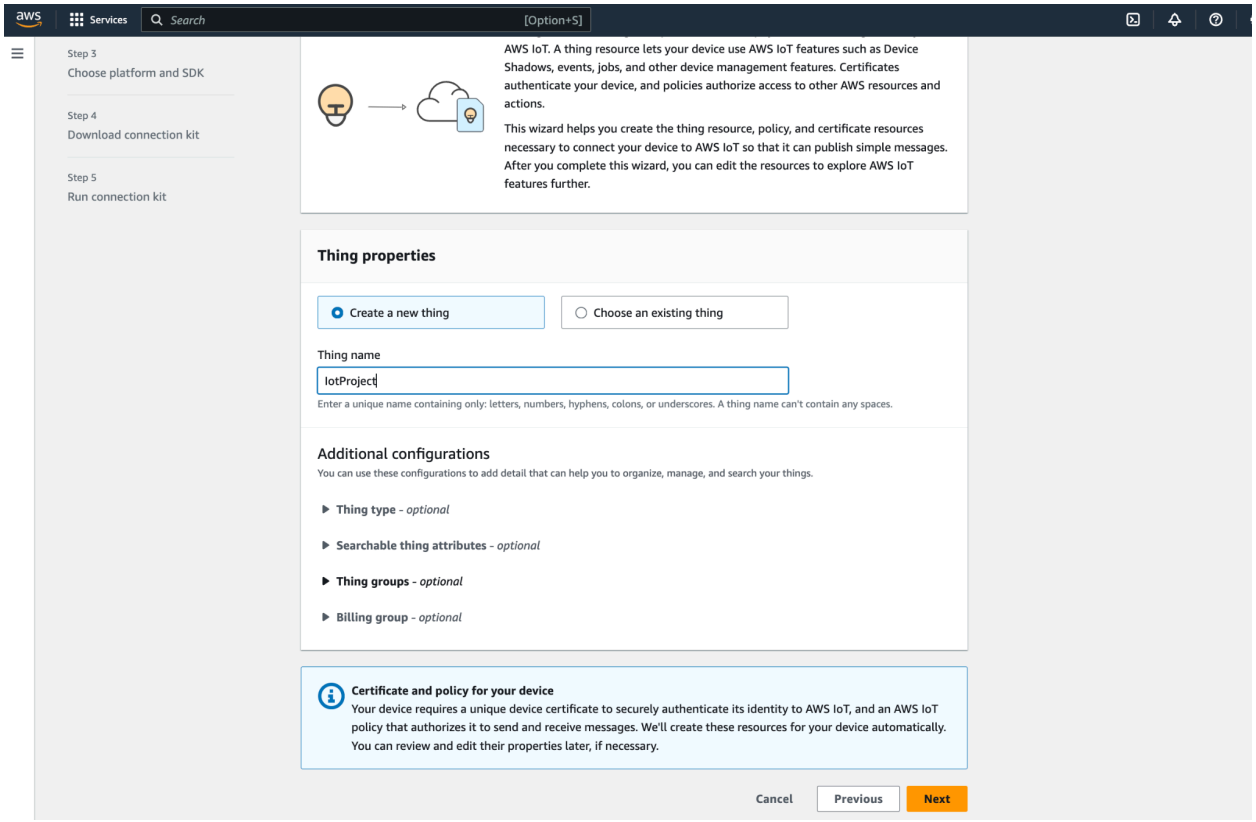


- Click on Connect Device and Please Copy the code and enter in the terminal on the laptop.

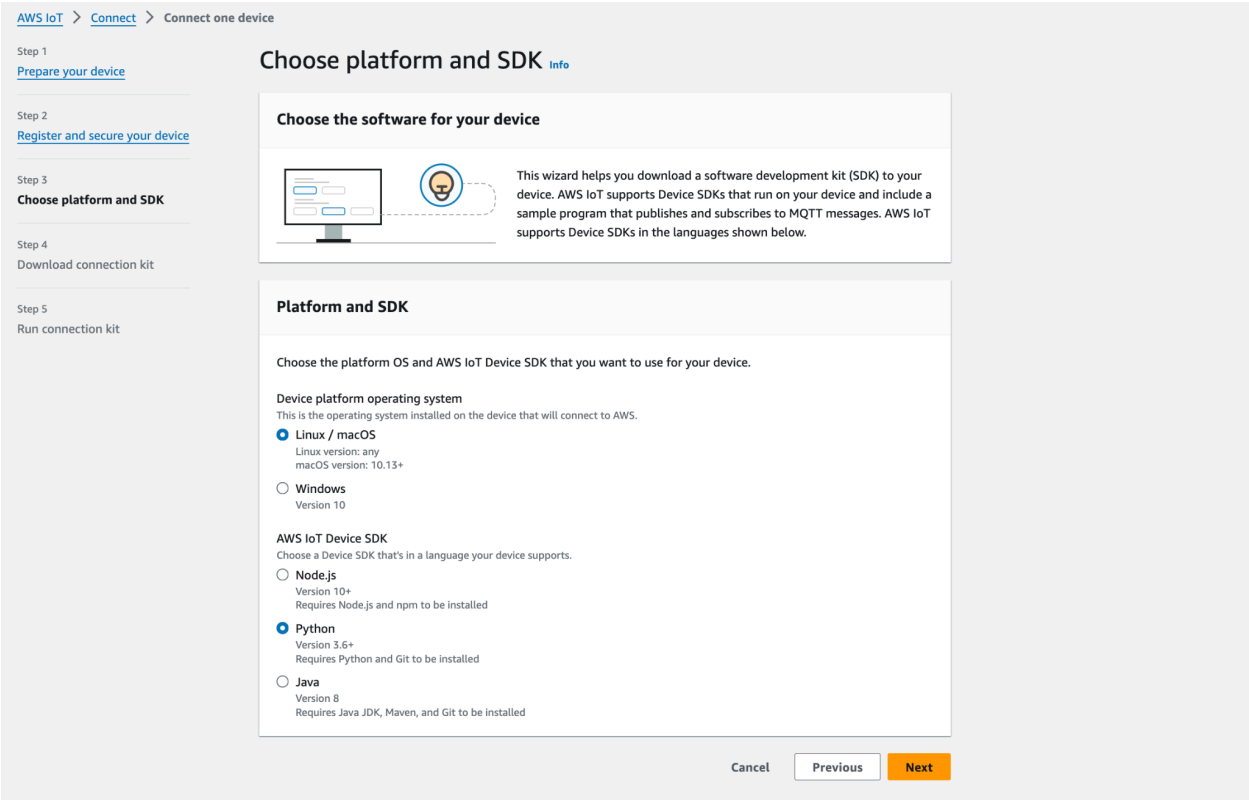




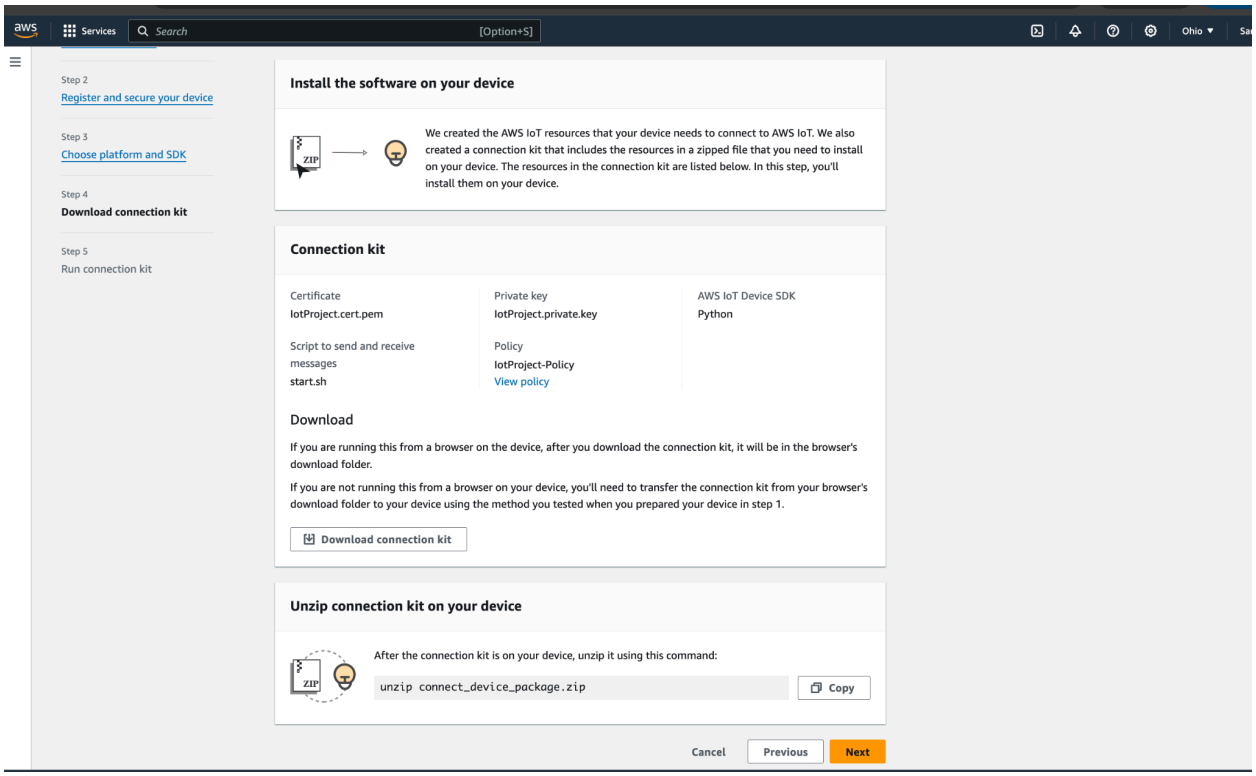
- Select Create a new thing naming as IotProject and click next



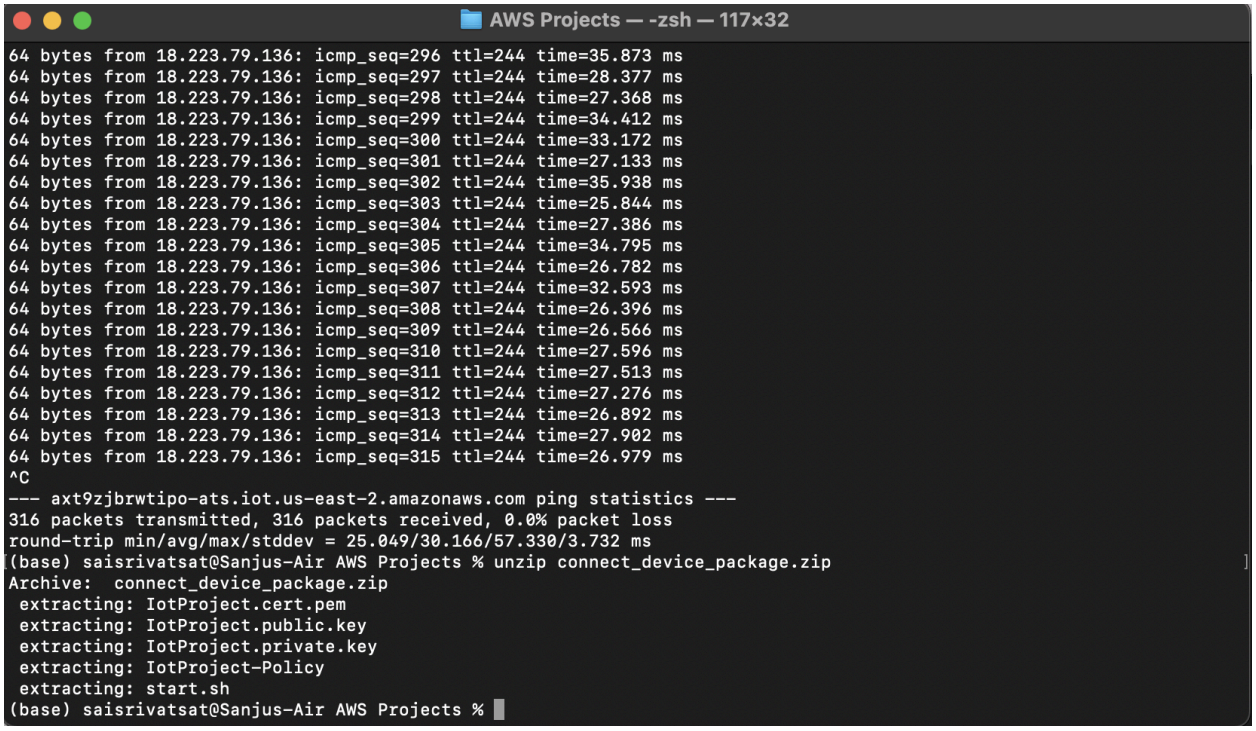
- Please select platform and SDK, AWS IoT Core will generate a connection kit that includes all necessary certificates, keys, and a sample Python SDK application.



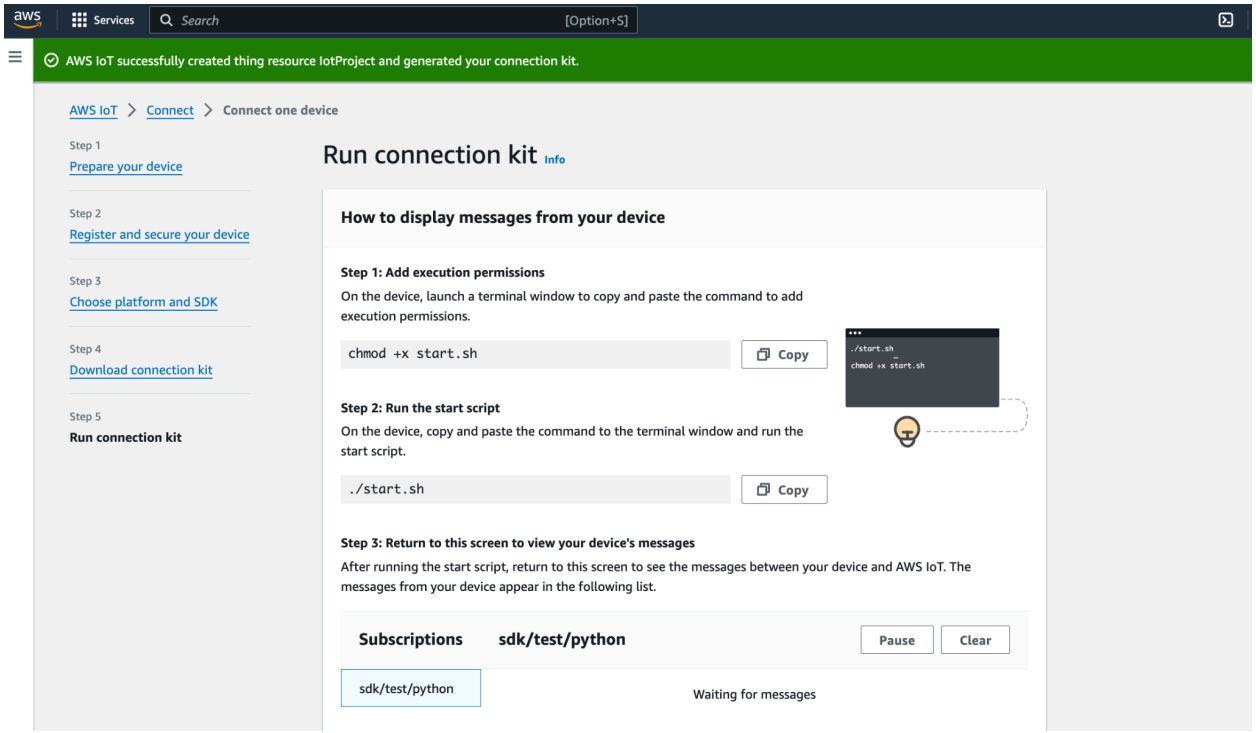
- Download the connection kit, including the **x509 certificate**, **public/private keys**, and **policy**.

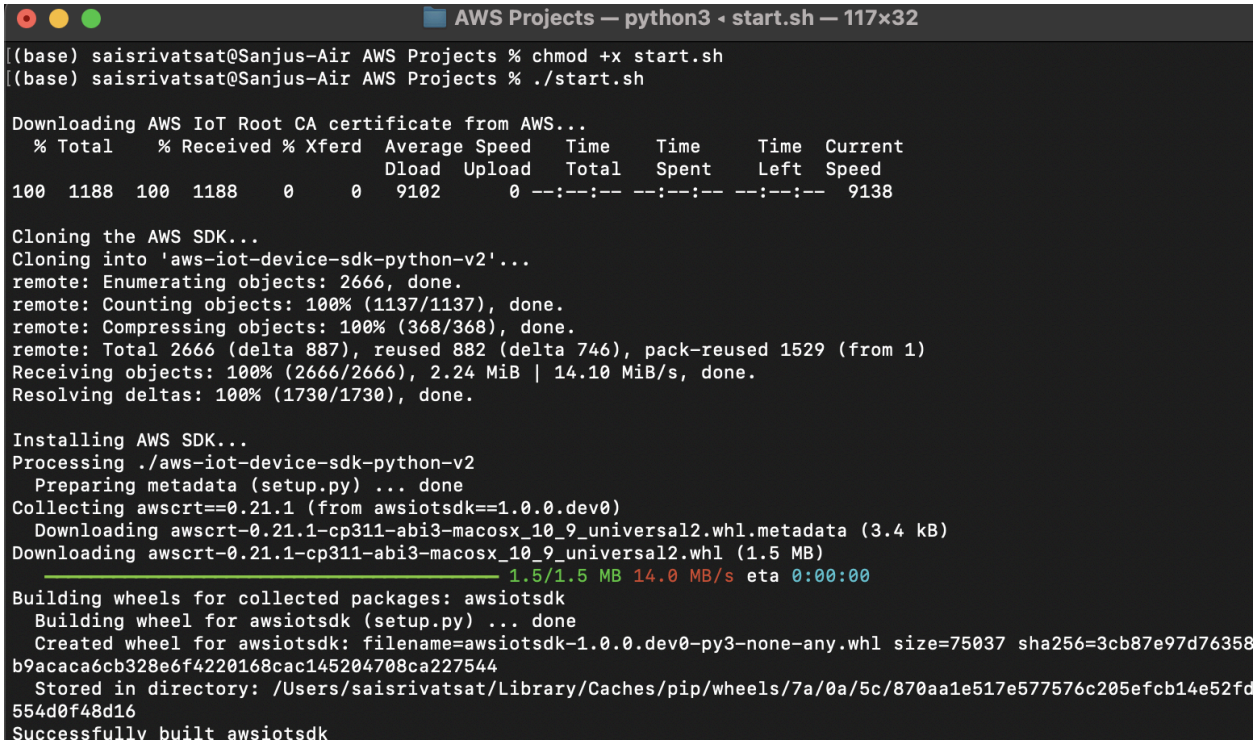


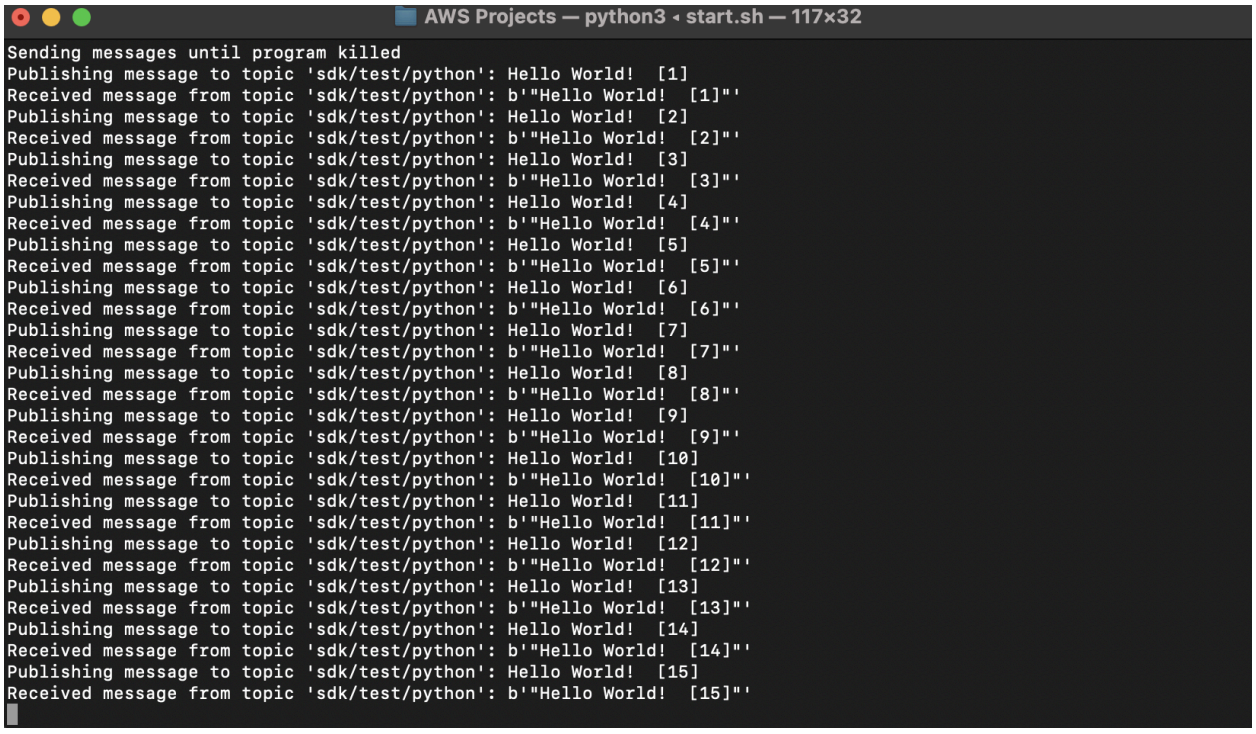
- Skip optional configurations and download the connection kit generated by AWS IoT Core.
- Unzip the connection kit on your laptop with the code given.



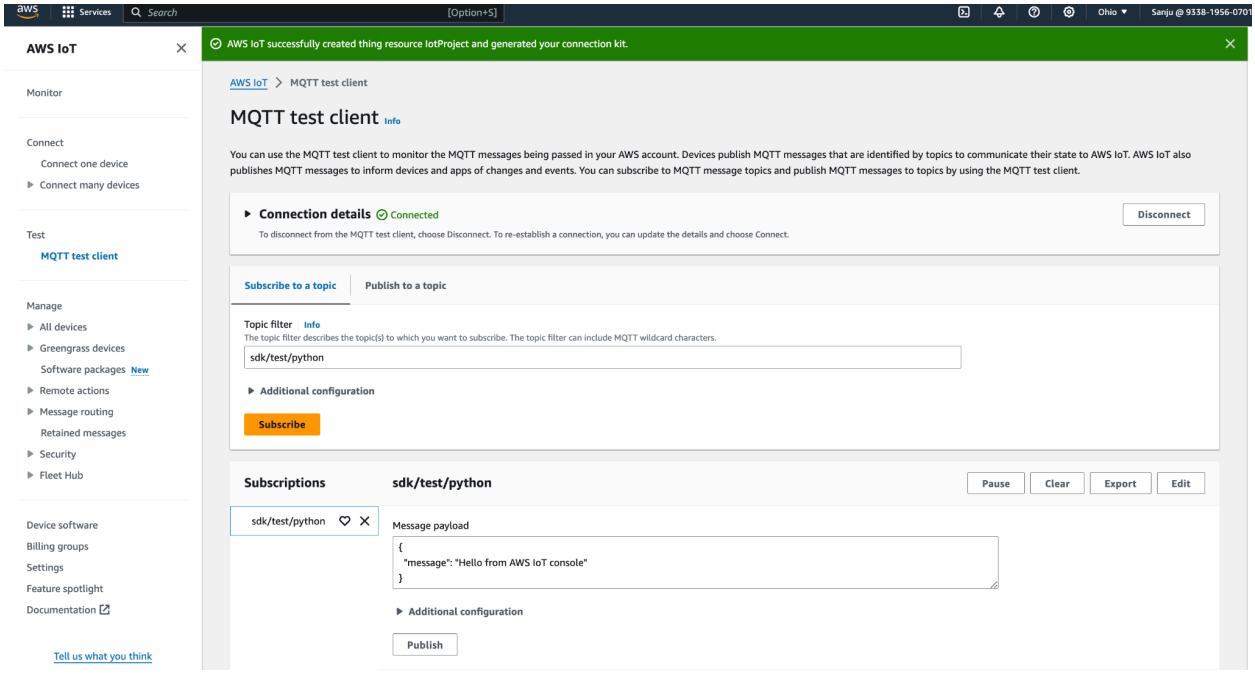
- Run the provided script in your terminal that will allow us to display messages from device.



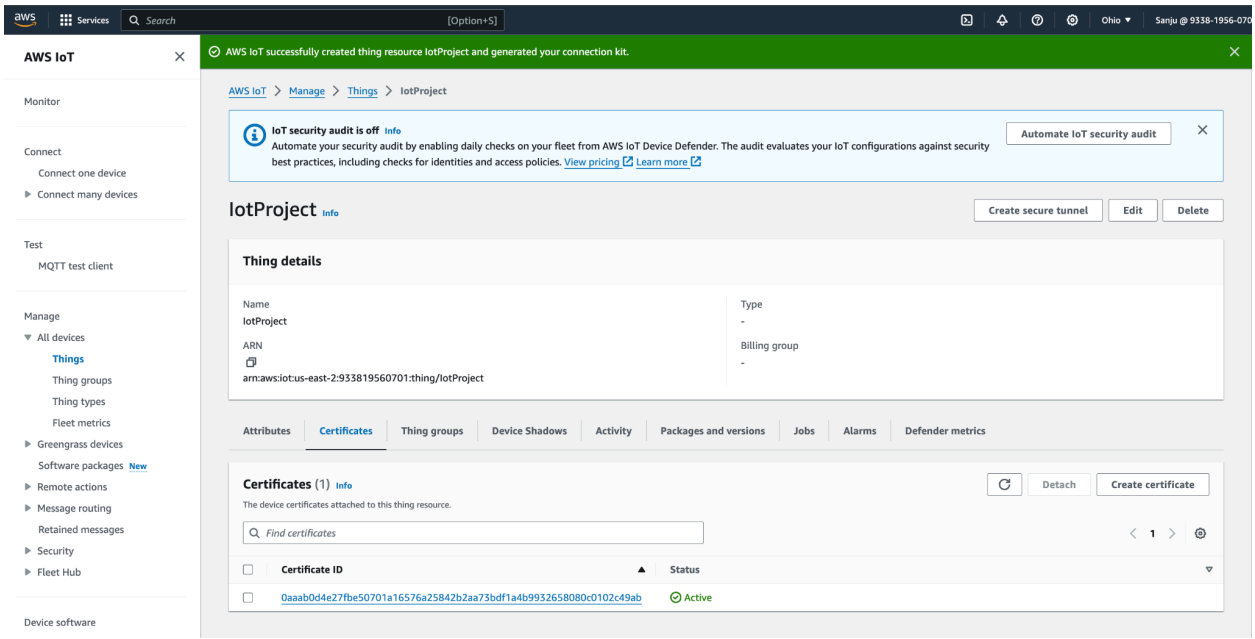


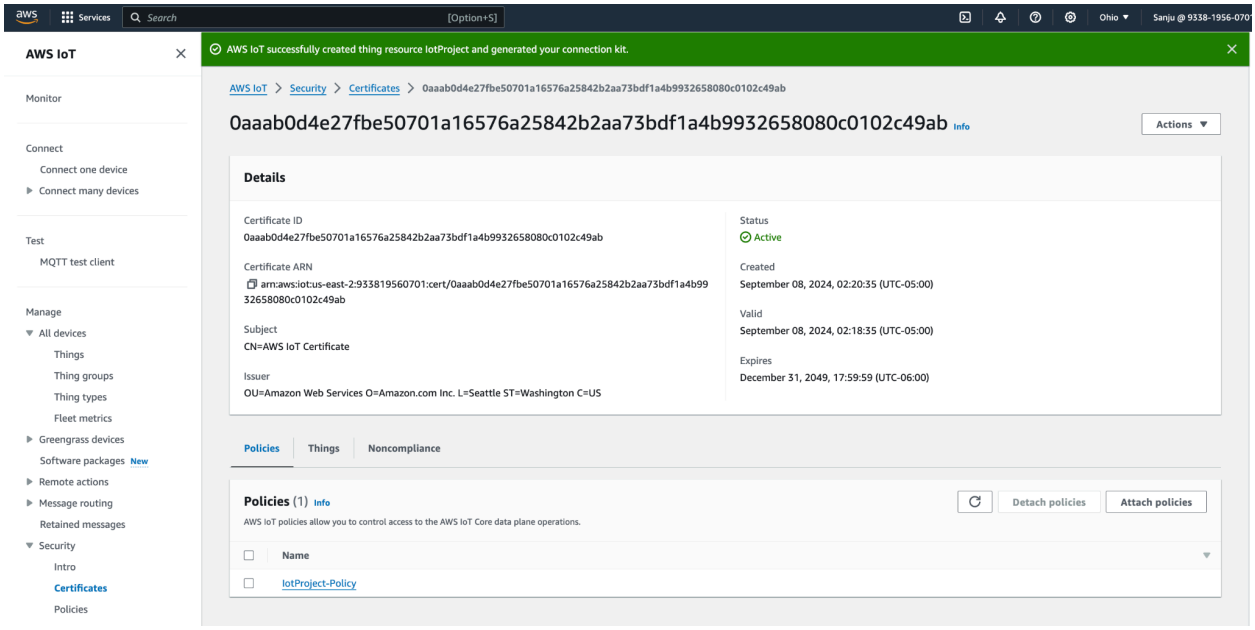


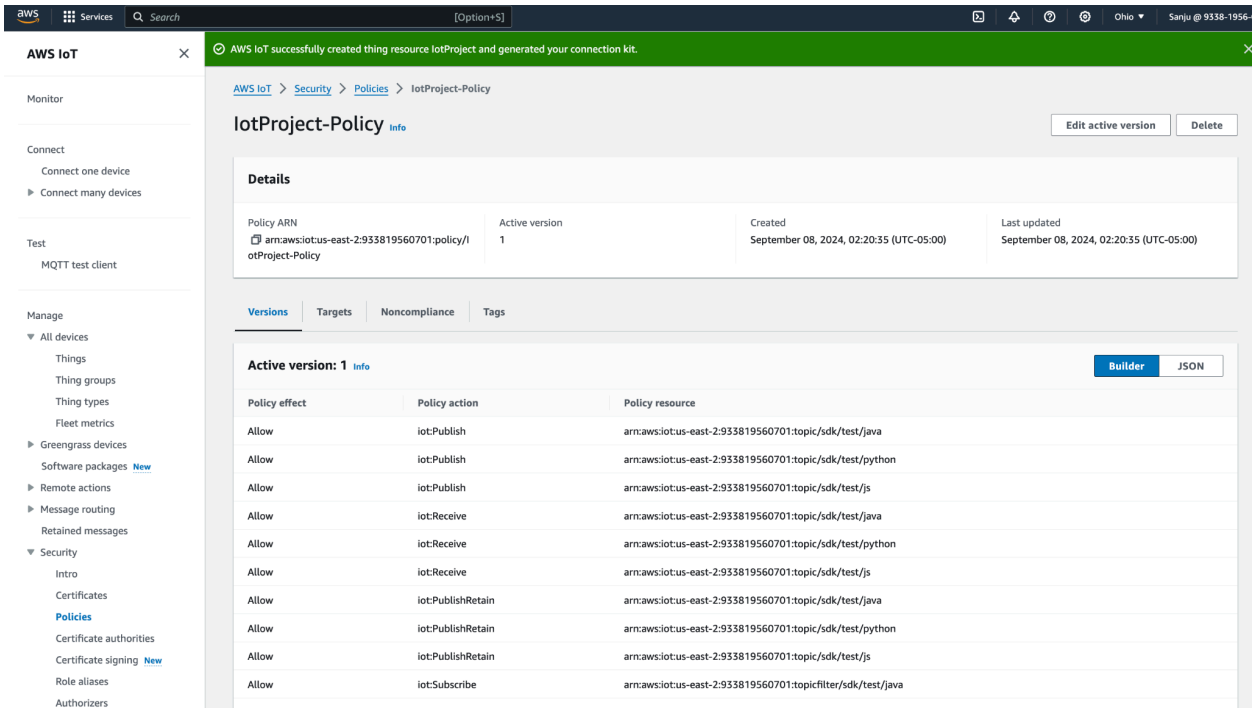
- Click on MQTT test client on AWS IoT console and put sdk/test/python in the topic filter then click subscribe.

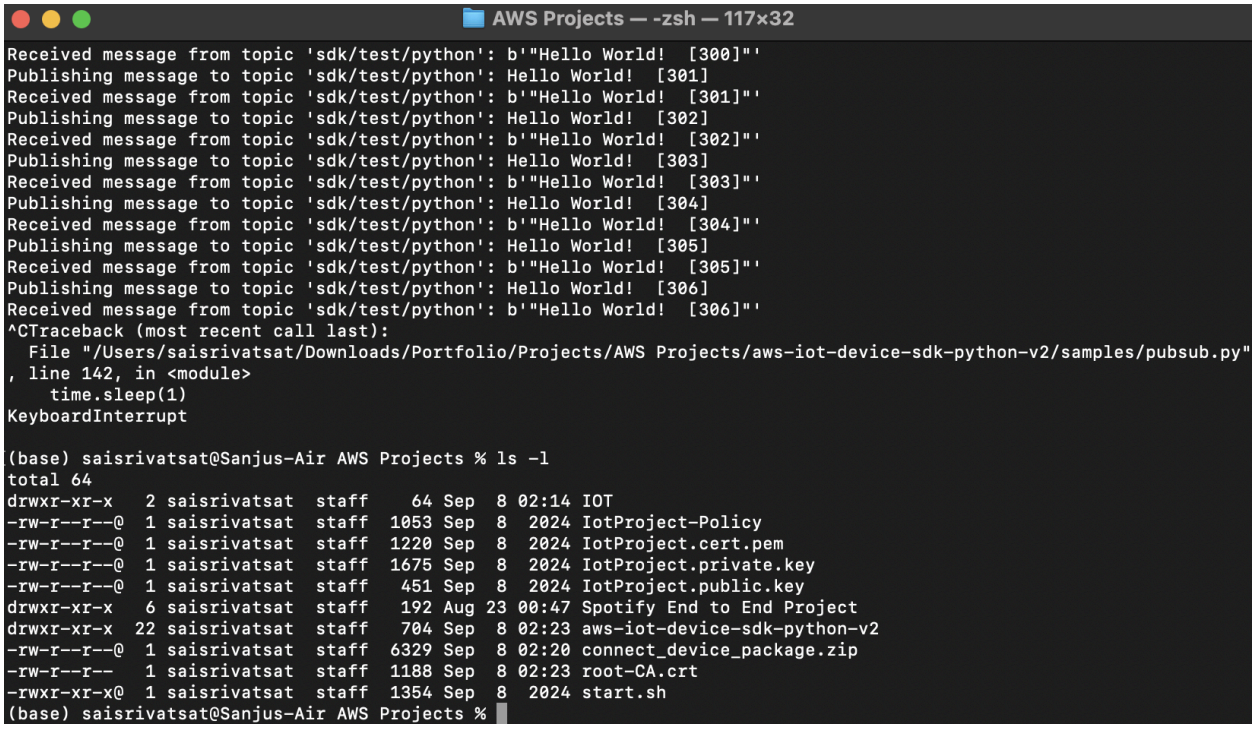


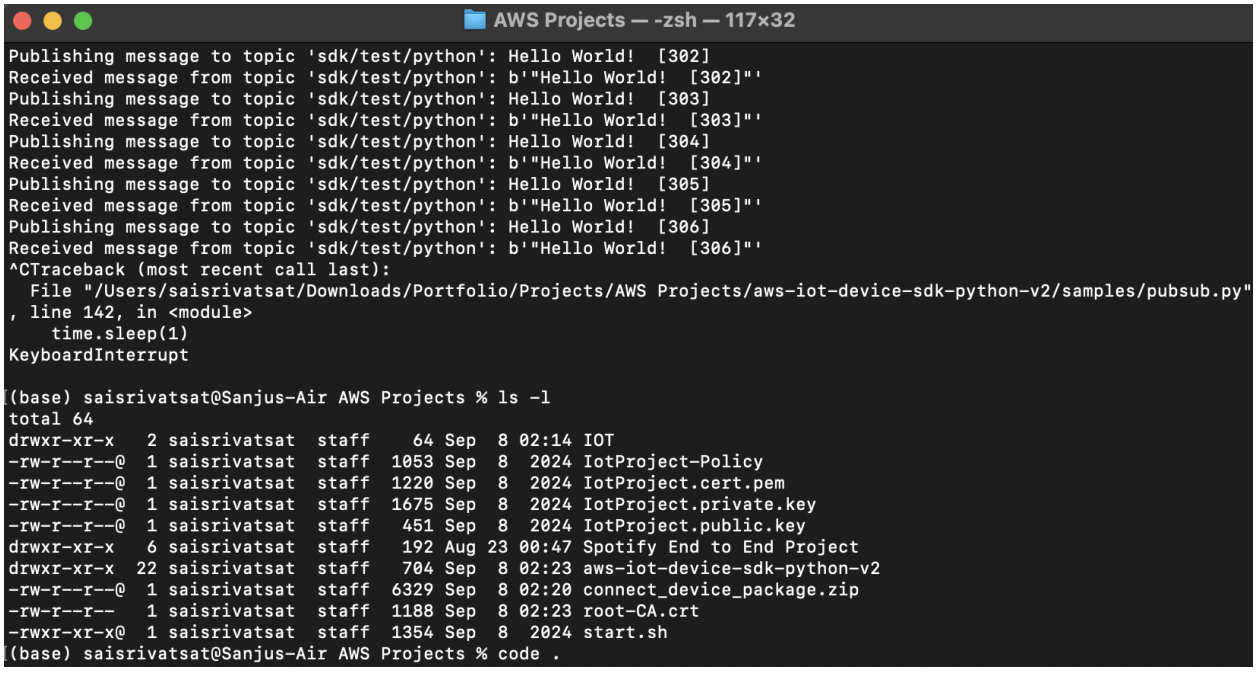
- Next In the IoT Core dashboard, go to **Manage** > **Things > IoTProject > Certificates**




AWS IoT
Monitor
Connect
Connect one device
Connect many devices
Test
MQTT test client
Manage
All devices
Things
Thing groups
Thing types
Fleet metrics
Greengrass devices
Software packages New
Remote actions
Message routing
Retained messages
Security
Intro
Certificates
Policies
AWS IoT successfully created thing resource IotProject and generated your connection kit.
AWS IoT > Security > Certificates > 0aaab0d4e27fbe50701a16576a25842b2aa73bdf1a4b9932658080c0102c49ab
0aaab0d4e27fbe50701a16576a25842b2aa73bdf1a4b9932658080c0102c49ab Info
Actions
Details
Certificate ID
0aaab0d4e27fbe50701a16576a25842b2aa73bdf1a4b9932658080c0102c49ab
Certificate ARN
arn:aws:iot:us-east-2:933819560701:cert/0aaab0d4e27fbe50701a16576a25842b2aa73bdf1a4b9932658080c0102c49ab
Subject
CN=AWS IoT Certificate
Issuer
OU=Amazon Web Services O=Amazon.com Inc. L=Seattle ST=Washington C=US
Status
Active
Created
September 08, 2024, 02:20:35 (UTC-05:00)
Valid
September 08, 2024, 02:18:35 (UTC-05:00)
Expires
December 31, 2049, 17:59:59 (UTC-06:00)
Policies
Things
Noncompliance
Policies (1) Info
Detach policies
Attach policies
AWS IoT policies allow you to control access to the AWS IoT Core data plane operations.
Name
IotProject-Policy

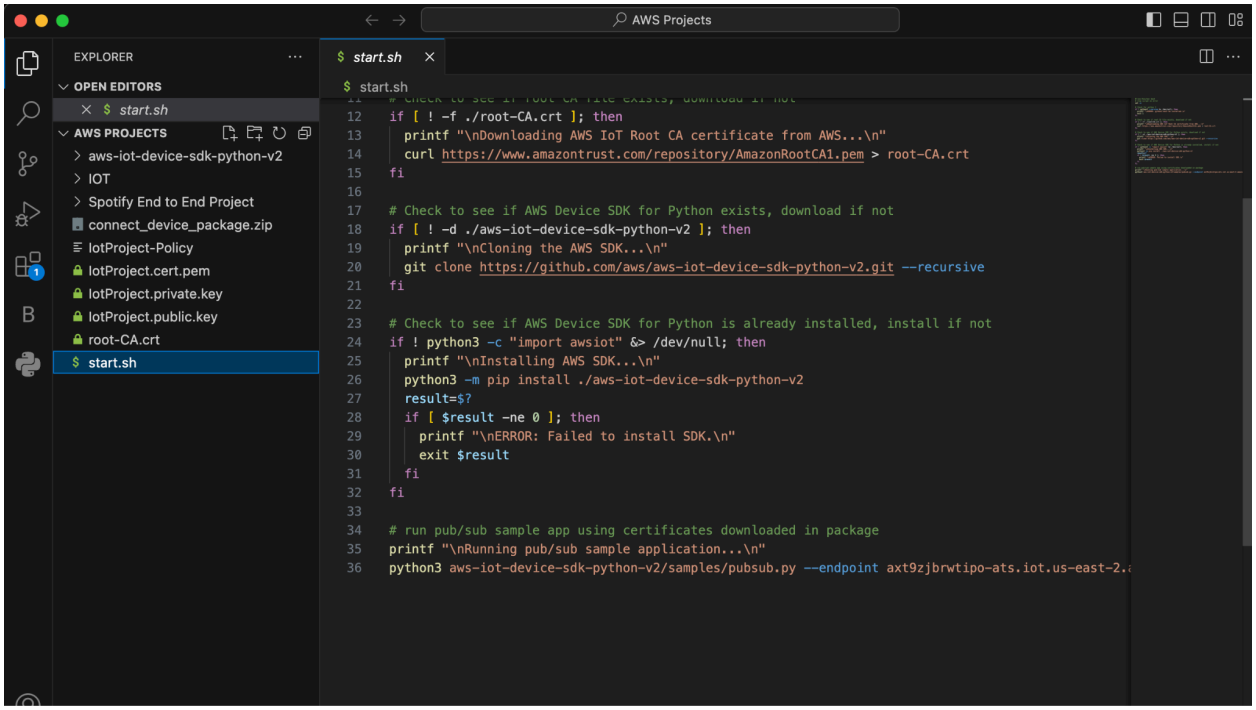

AWS IoT
Monitor
Connect
Connect one device
Connect many devices
Test
MQTT test client
Manage
All devices
Things
Thing groups
Thing types
Fleet metrics
Greengrass devices
Software packages New
Remote actions
Message routing
Retained messages
Security
Intro
Certificates
Policies
Certificate authorities
Certificate signing New
Role aliases
Authorizers
AWS IoT successfully created thing resource IotProject and generated your connection kit.
AWS IoT > Security > Policies > IotProject-Policy
IotProject-Policy Info
Edit active version
Delete
Details
Policy ARN
arn:aws:iot:us-east-2:933819560701:policy/IotProject-Policy
Active version
1
Created
September 08, 2024, 02:20:35 (UTC-05:00)
Last updated
September 08, 2024, 02:20:35 (UTC-05:00)
Versions
Targets
Noncompliance
Tags
Active version: 1 Info
Builder
JSON
Policy effect
Policy action
Policy resource
Allow iot:Publish arn:aws:iot:us-east-2:933819560701:topic/sdk/test/java
Allow iot:Publish arn:aws:iot:us-east-2:933819560701:topic/sdk/test/python
Allow iot:Publish arn:aws:iot:us-east-2:933819560701:topic/sdk/test/js
Allow iot:Receive arn:aws:iot:us-east-2:933819560701:topic/sdk/test/java
Allow iot:Receive arn:aws:iot:us-east-2:933819560701:topic/sdk/test/python
Allow iot:Receive arn:aws:iot:us-east-2:933819560701:topic/sdk/test/js
Allow iot:PublishRetain arn:aws:iot:us-east-2:933819560701:topic/sdk/test/java
Allow iot:PublishRetain arn:aws:iot:us-east-2:933819560701:topic/sdk/test/python
Allow iot:PublishRetain arn:aws:iot:us-east-2:933819560701:topic/sdk/test/js
Allow iot:Subscribe arn:aws:iot:us-east-2:933819560701:topicfilter/sdk/test/java





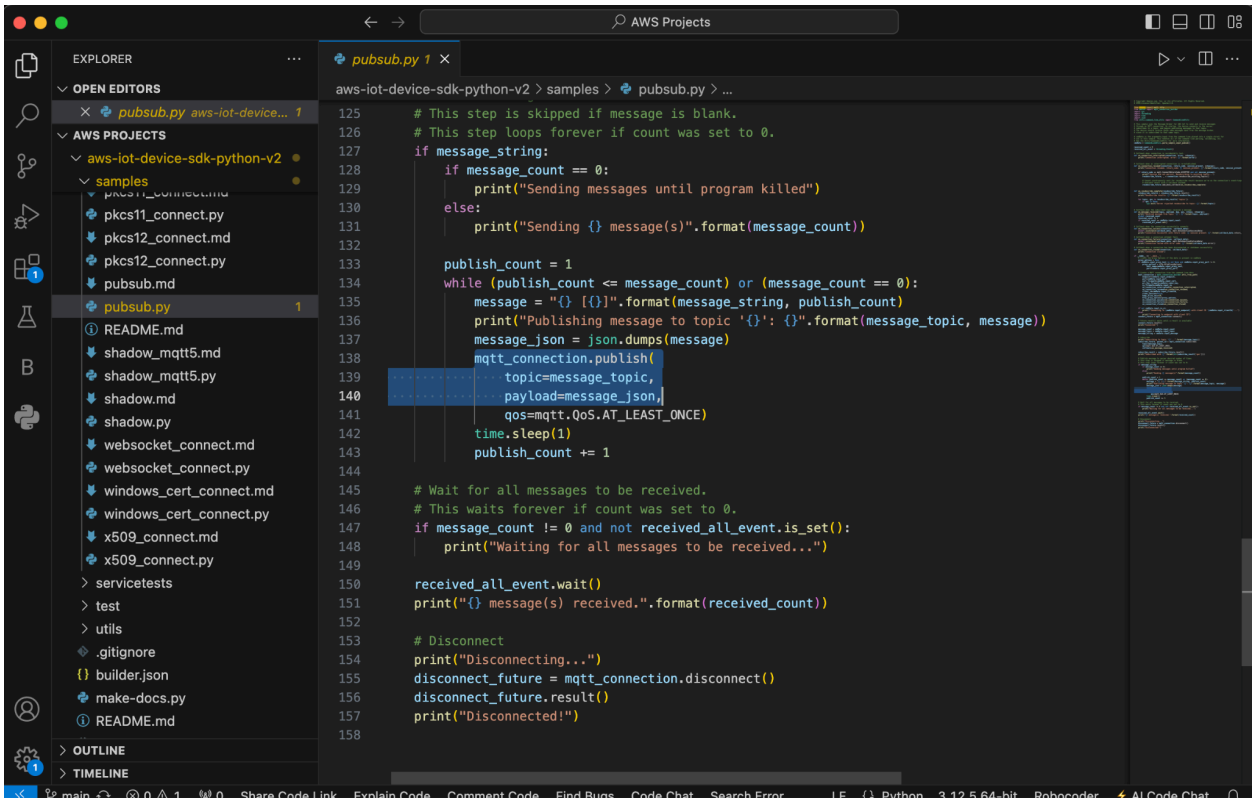
Modify the Sample Client

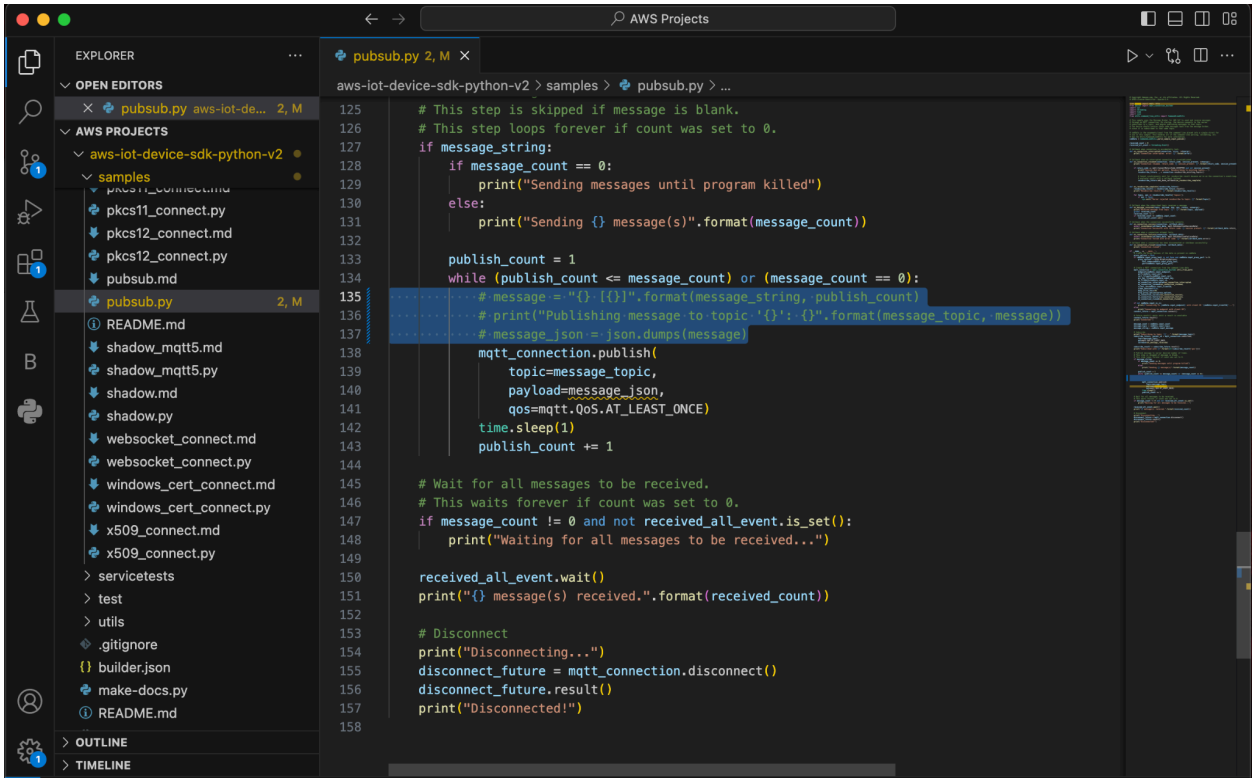
1. **Open the Python Script**:
    - Open the `pubsub.py` file from the connection kit.

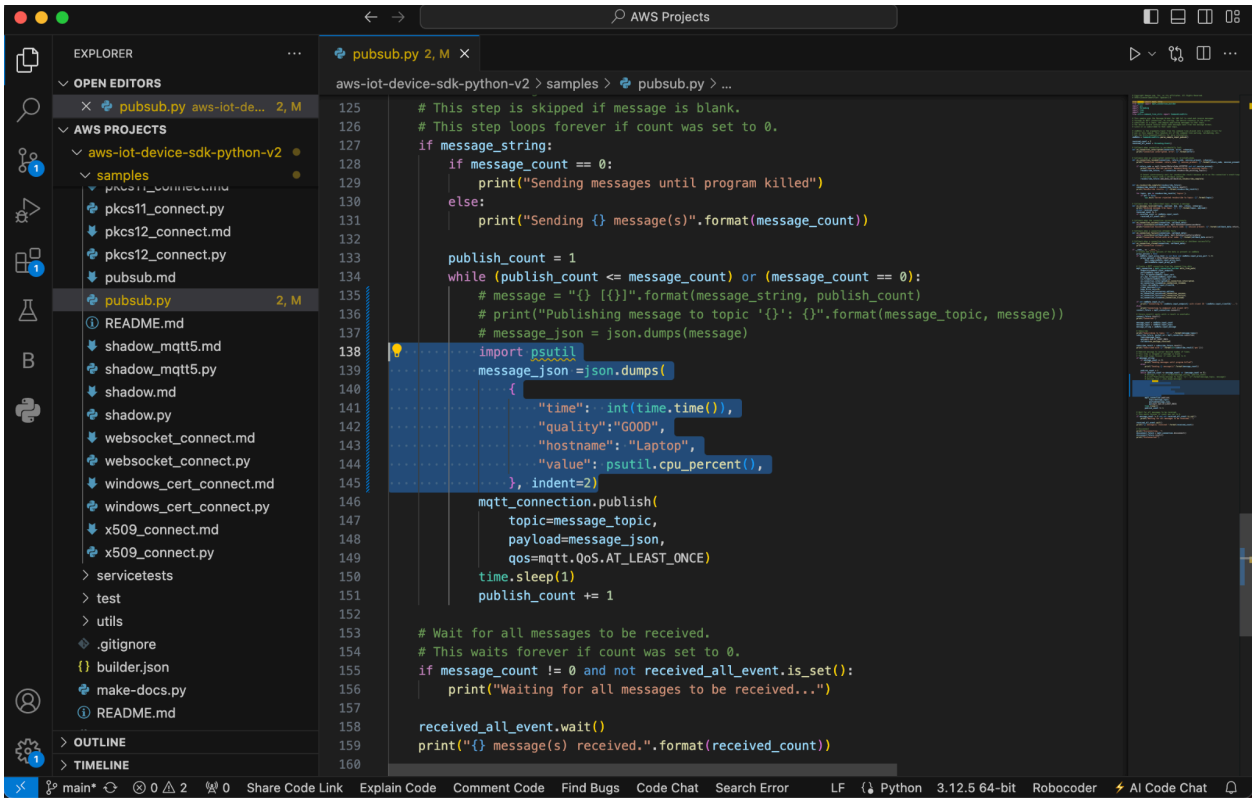


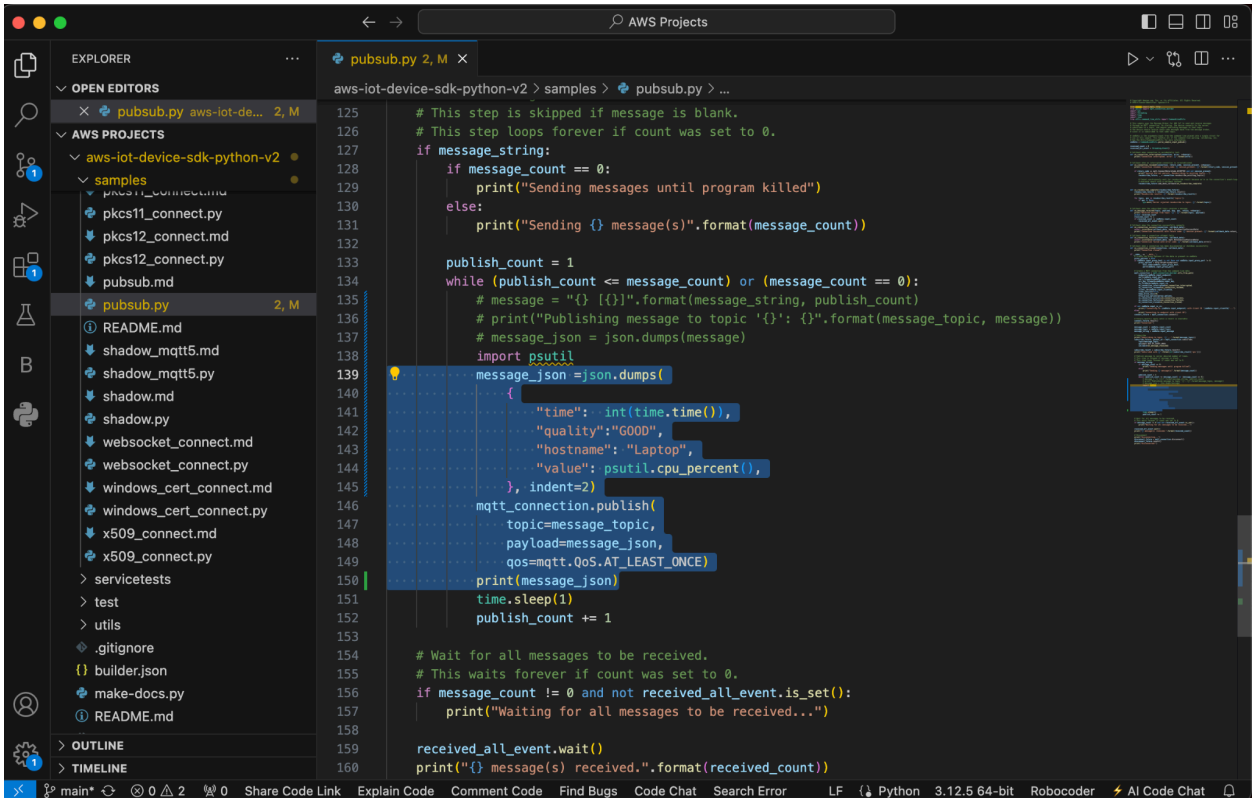
2. **Add JSON-formatted Payload**:

Modify the script to send CPU utilization data in a JSON format:

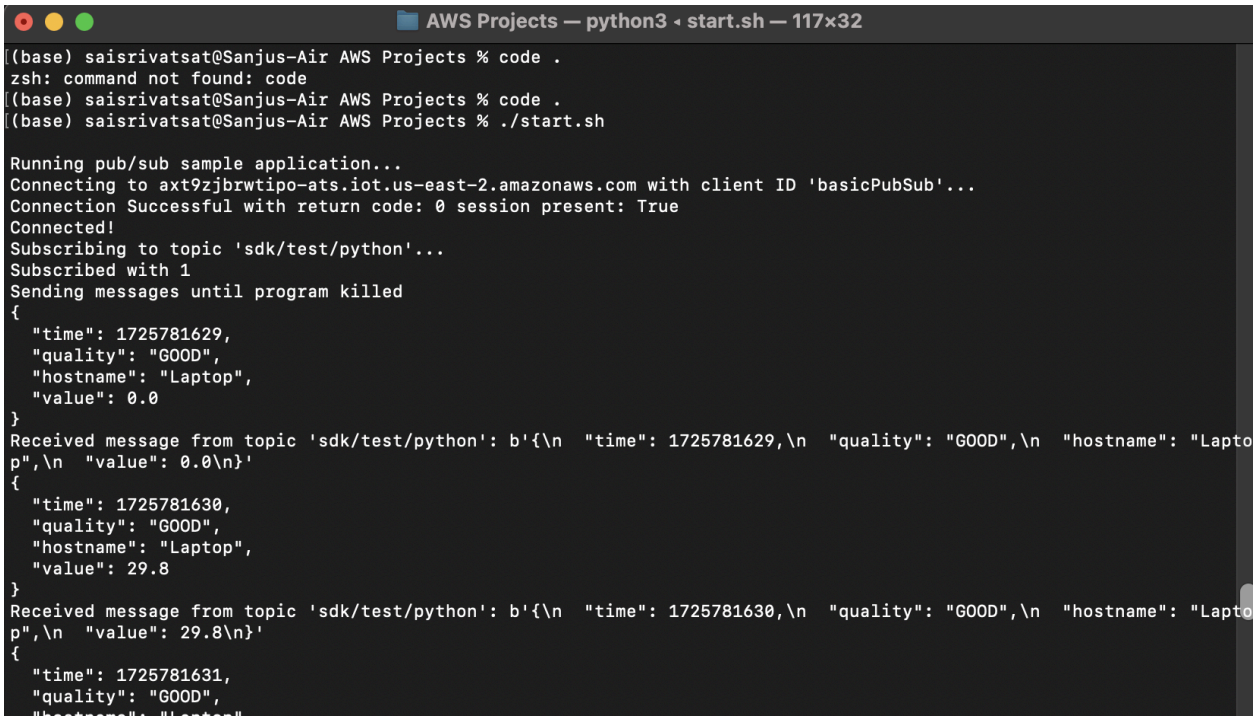


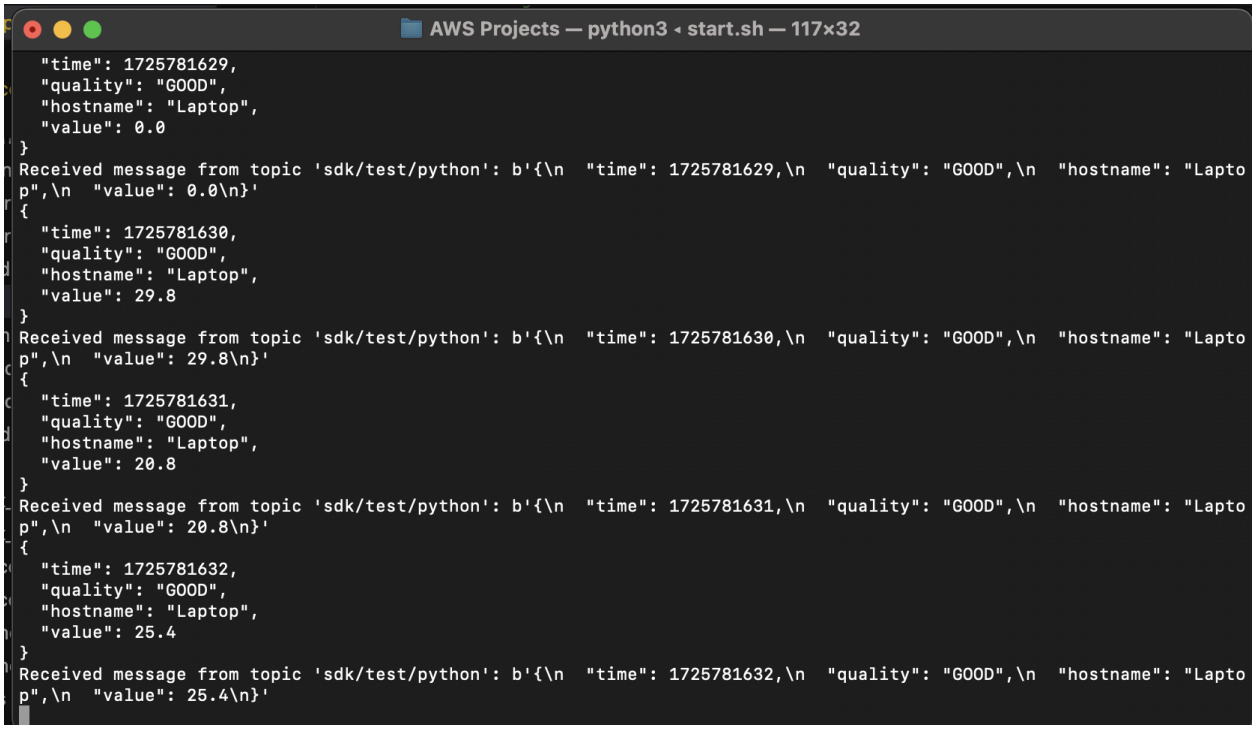

AWS Projects
EXPLORER
OPEN EDITORS
pubsub.py aws-iot-de... 2, M
AWS PROJECTS
aws-iot-device-sdk-python-v2
samples
pkcs11_connect.py
pkcs12_connect.md
pkcs12_connect.py
pubsub.md
pubsub.py 2, M
README.md
shadow_mqtt5.md
shadow_mqtt5.py
shadow.md
shadow.py
websocket_connect.md
websocket_connect.py
windows_cert_connect.md
windows_cert_connect.py
x509_connect.md
x509_connect.py
servicetests
test
utils
.gitignore
builder.json
make-docs.py
README.md
OUTLINE
TIMELINE
pubsub.py 2, M
aws-iot-device-sdk-python-v2 > samples > pubsub.py > ...
125 # This step is skipped if message is blank.
126 # This step loops forever if count was set to 0.
127 if message_string:
128 if message_count == 0:
129 print("Sending messages until program killed")
130 else:
131 print("Sending {} message(s)".format(message_count))
132
133 publish_count = 1
134 while (publish_count <= message_count) or (message_count == 0):
135 # message = "{} [{}]".format(message_string, publish_count)
136 # print("Publishing message to topic '{}': {}".format(message_topic, message))
137 # message_json = json.dumps(message)
138 mqtt_connection.publish(
139 topic=message_topic,
140 payload=message_json,
141 qos=mqtt.QoS.AT_LEAST_ONCE)
142 time.sleep(1)
143 publish_count += 1
144
145 # Wait for all messages to be received.
146 # This waits forever if count was set to 0.
147 if message_count != 0 and not received_all_event.is_set():
148 print("Waiting for all messages to be received...")
149
150 received_all_event.wait()
151 print("{} message(s) received.".format(received_count))
152
153 # Disconnect
154 print("Disconnecting...")
155 disconnect_future = mqtt_connection.disconnect()
156 disconnect_future.result()
157 print("Disconnected!")
158

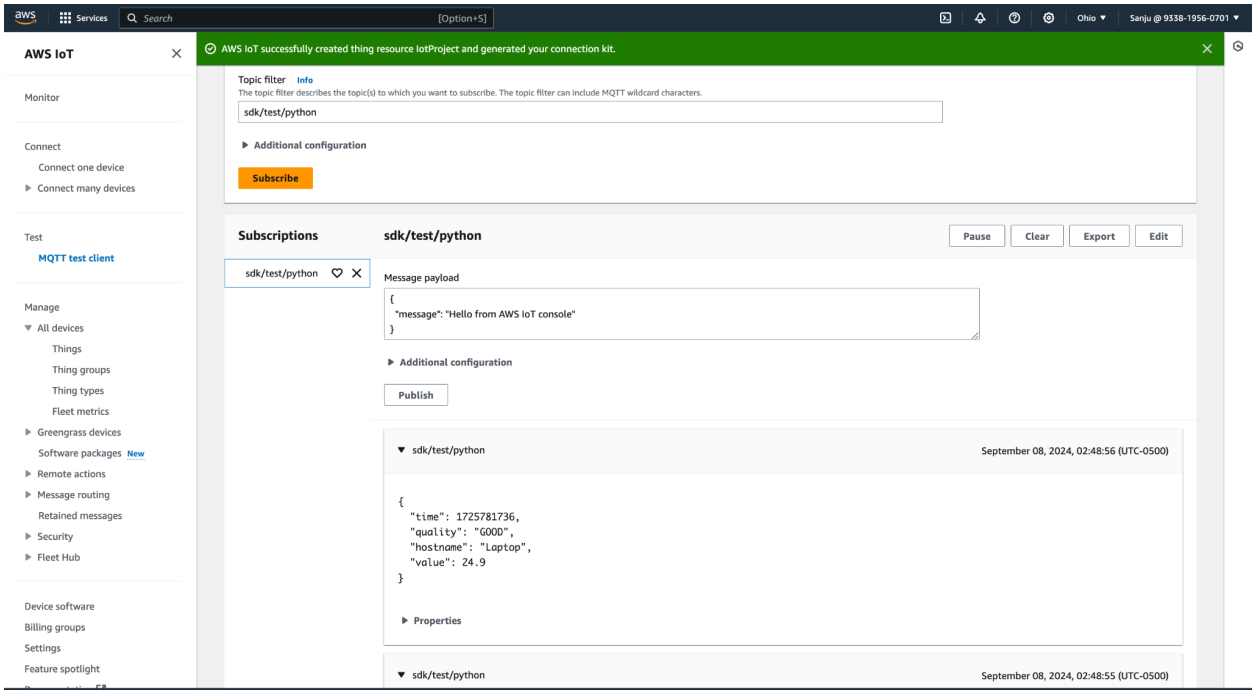

AWS Projects
EXPLORER
OPEN EDITORS
pubsub.py aws-iot-de... 2, M
AWS PROJECTS
aws-iot-device-sdk-python-v2
samples
pkcs11_connect.py
pkcs12_connect.md
pkcs12_connect.py
pubsub.md
pubsub.py 2, M
README.md
shadow_mqtt5.md
shadow_mqtt5.py
shadow.md
shadow.py
websocket_connect.md
websocket_connect.py
windows_cert_connect.md
windows_cert_connect.py
x509_connect.md
x509_connect.py
servicetests
test
utils
.gitignore
builder.json
make-docs.py
README.md
OUTLINE
TIMELINE
pubsub.py 2, M
aws-iot-device-sdk-python-v2 > samples > pubsub.py > ...
125 # This step is skipped if message is blank.
126 # This step loops forever if count was set to 0.
127 if message_string:
128 if message_count == 0:
129 print("Sending messages until program killed")
130 else:
131 print("Sending {} message(s)".format(message_count))
132
133 publish_count = 1
134 while (publish_count <= message_count) or (message_count == 0):
135 # message = "{} [{}]".format(message_string, publish_count)
136 # print("Publishing message to topic '{}': {}".format(message_topic, message))
137 # message_json = json.dumps(message)
138 import psutil
139 message_json =json.dumps(
140 {
141 "time": int(time.time()),
142 "quality":"GOOD",
143 "hostname": "Laptop",
144 "value": psutil.cpu_percent(),
145 }, indent=2)
146 mqtt_connection.publish(
147 topic=message_topic,
148 payload=message_json,
149 qos=mqtt.QoS.AT_LEAST_ONCE)
150 time.sleep(1)
151 publish_count += 1
152
153 # Wait for all messages to be received.
154 # This waits forever if count was set to 0.
155 if message_count != 0 and not received_all_event.is_set():
156 print("Waiting for all messages to be received...")
157
158 received_all_event.wait()
159 print("{} message(s) received.".format(received_count))
160
main* 0 2 0 Share Code Link Explain Code Comment Code Find Bugs Code Chat Search Error LF Python 3.12.5 64-bit Robocoder AI Code Chat



- Run ./start.sh on your terminal this will start sending data


AWS Projects — python3 ◂ start.sh — 117×32
(base) saisrivatsat@Sanjus-Air AWS Projects % code .
zsh: command not found: code
(base) saisrivatsat@Sanjus-Air AWS Projects % code .
(base) saisrivatsat@Sanjus-Air AWS Projects % ./start.sh
Running pub/sub sample application...
Connecting to axt9zjbrwtipo-ats.iot.us-east-2.amazonaws.com with client ID 'basicPubSub'...
Connection Successful with return code: 0 session present: True
Connected!
Subscribing to topic 'sdk/test/python'...
Subscribed with 1
Sending messages until program killed
{
"time": 1725781629,
"quality": "GOOD",
"hostname": "Laptop",
"value": 0.0
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781629,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 0.0\n}'
{
"time": 1725781630,
"quality": "GOOD",
"hostname": "Laptop",
"value": 29.8
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781630,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 29.8\n}'
{
"time": 1725781631,
"quality": "GOOD",

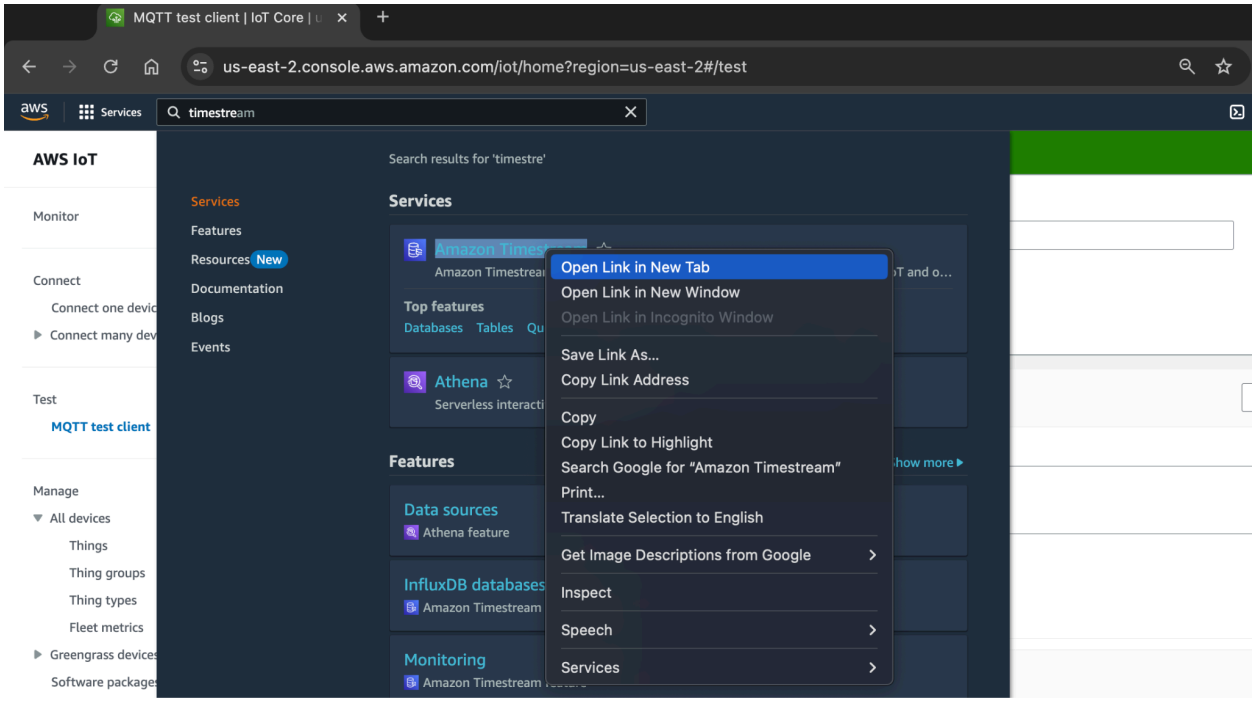

AWS Projects — python3 ◂ start.sh — 117×32
"time": 1725781629,
"quality": "GOOD",
"hostname": "Laptop",
"value": 0.0
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781629,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 0.0\n}'
{
"time": 1725781630,
"quality": "GOOD",
"hostname": "Laptop",
"value": 29.8
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781630,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 29.8\n}'
{
"time": 1725781631,
"quality": "GOOD",
"hostname": "Laptop",
"value": 20.8
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781631,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 20.8\n}'
{
"time": 1725781632,
"quality": "GOOD",
"hostname": "Laptop",
"value": 25.4
}
Received message from topic 'sdk/test/python': b'{\n "time": 1725781632,\n "quality": "GOOD",\n "hostname": "Lapto
p",\n "value": 25.4\n}'



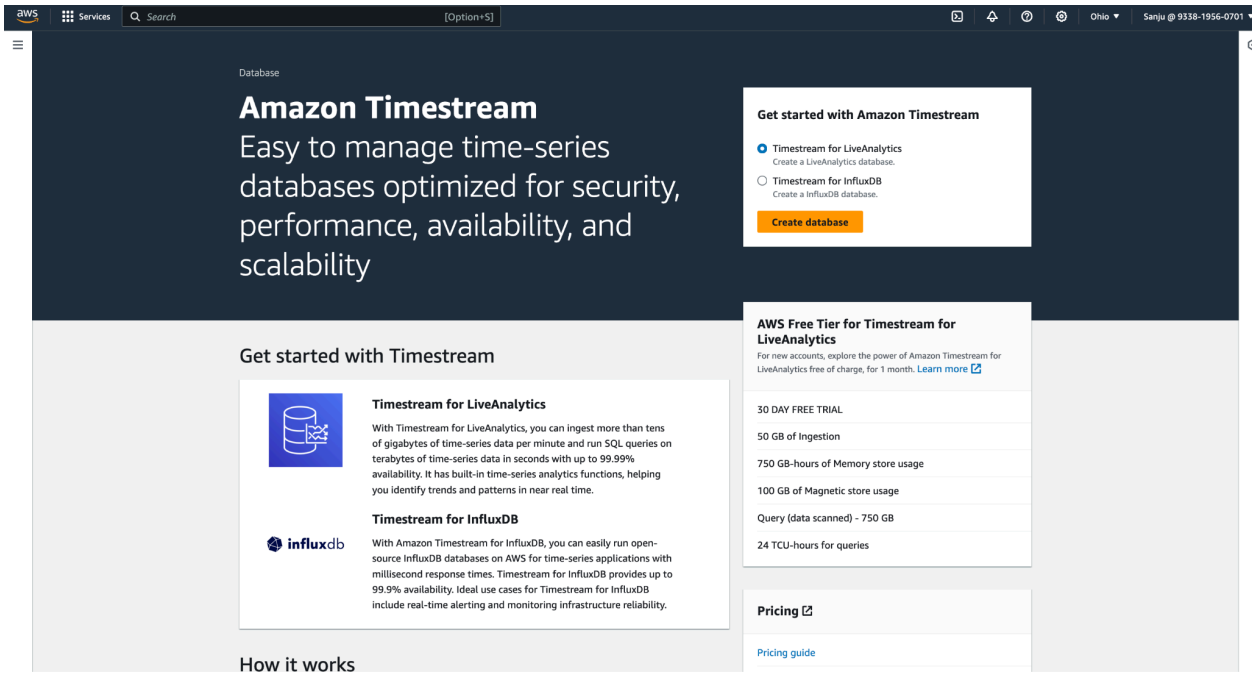
Storing Data in Amazon Timestream

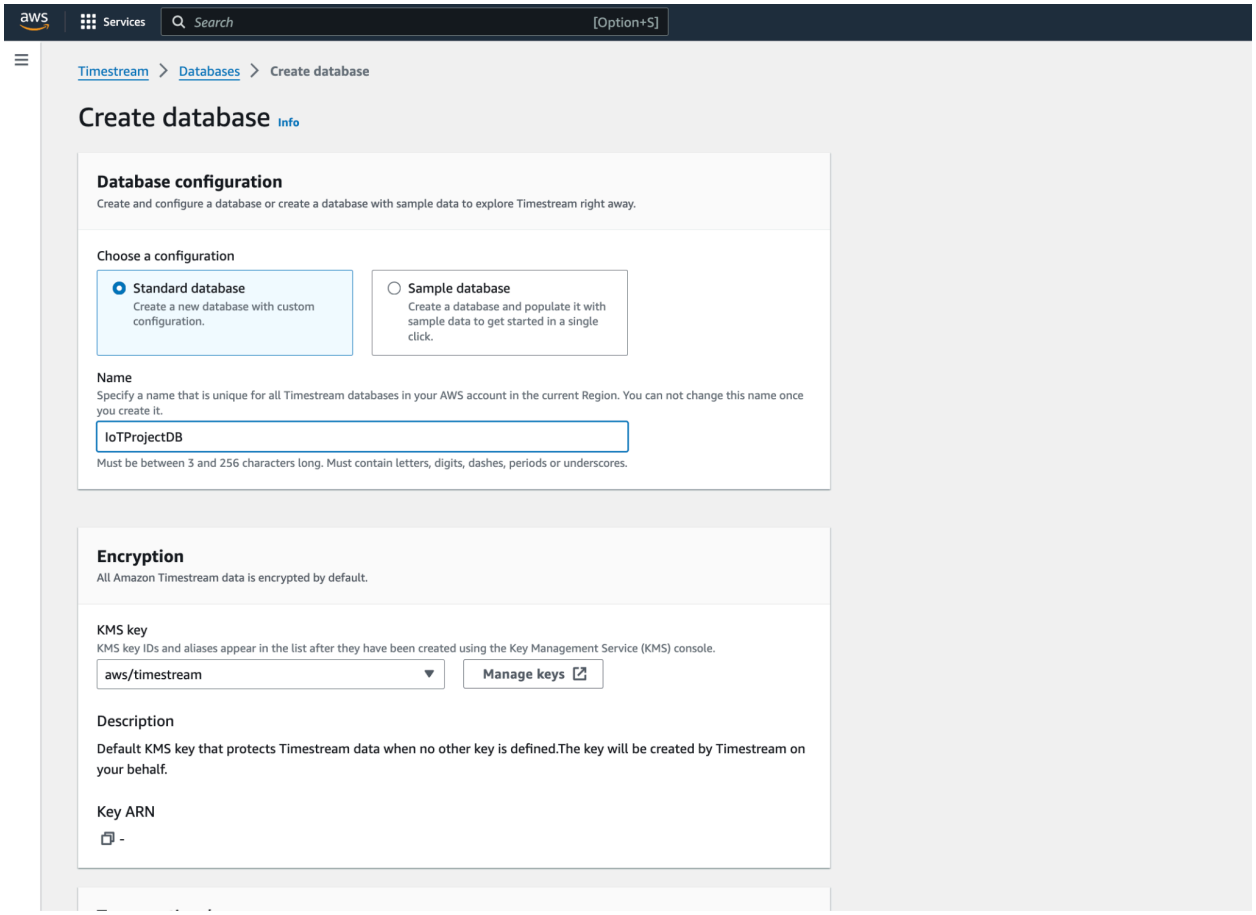
Create a Timestream Database

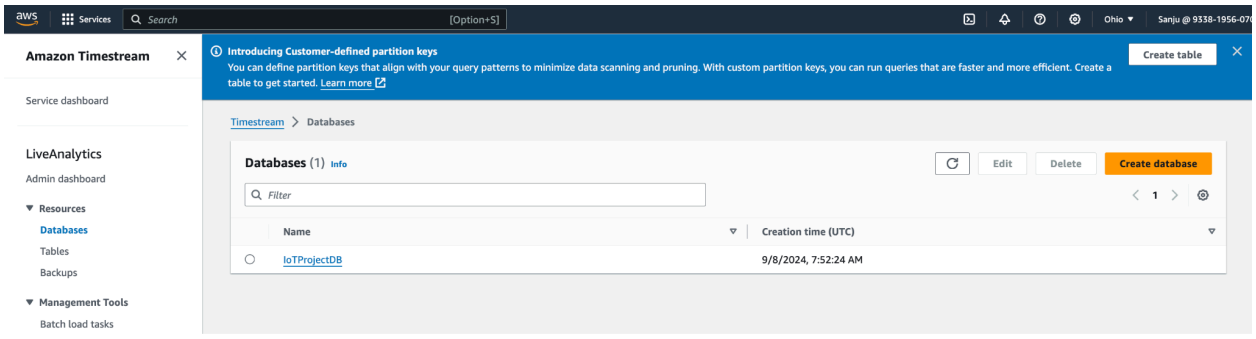
1. **Open Amazon Timestream**:
    - In the AWS Console, search for **Amazon Timestream** and open it.



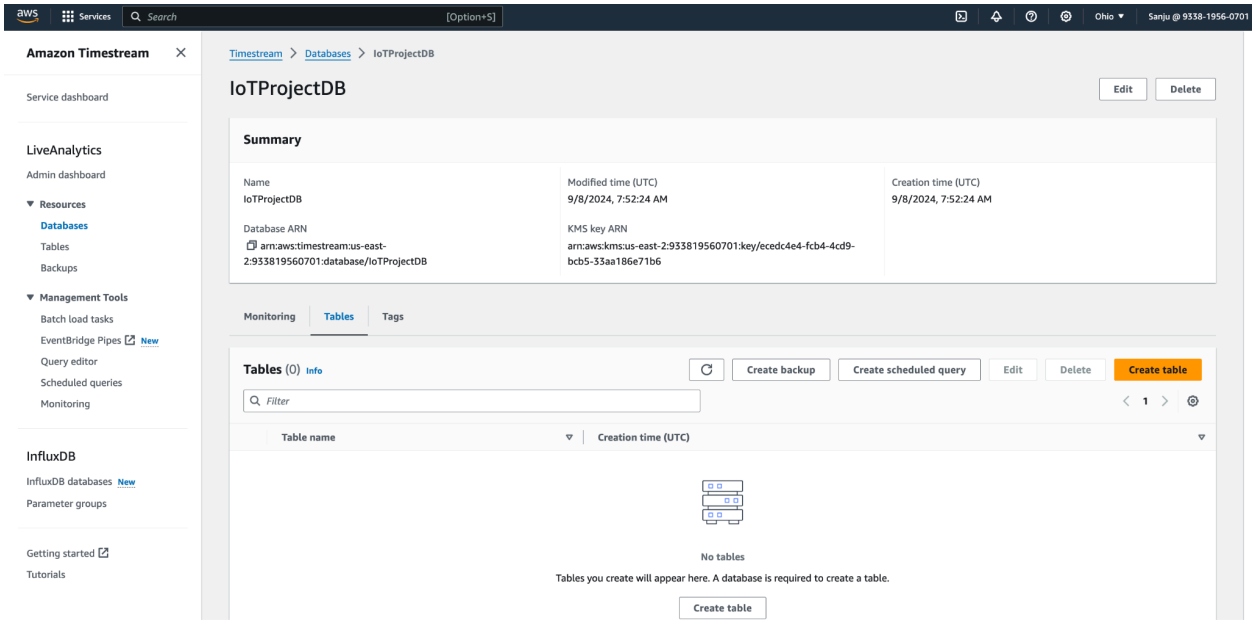
2. **Create a New Database**:
    - Click **Create Database**, provide a name (e.g., `IoTDataDB`), and accept the default encryption settings.

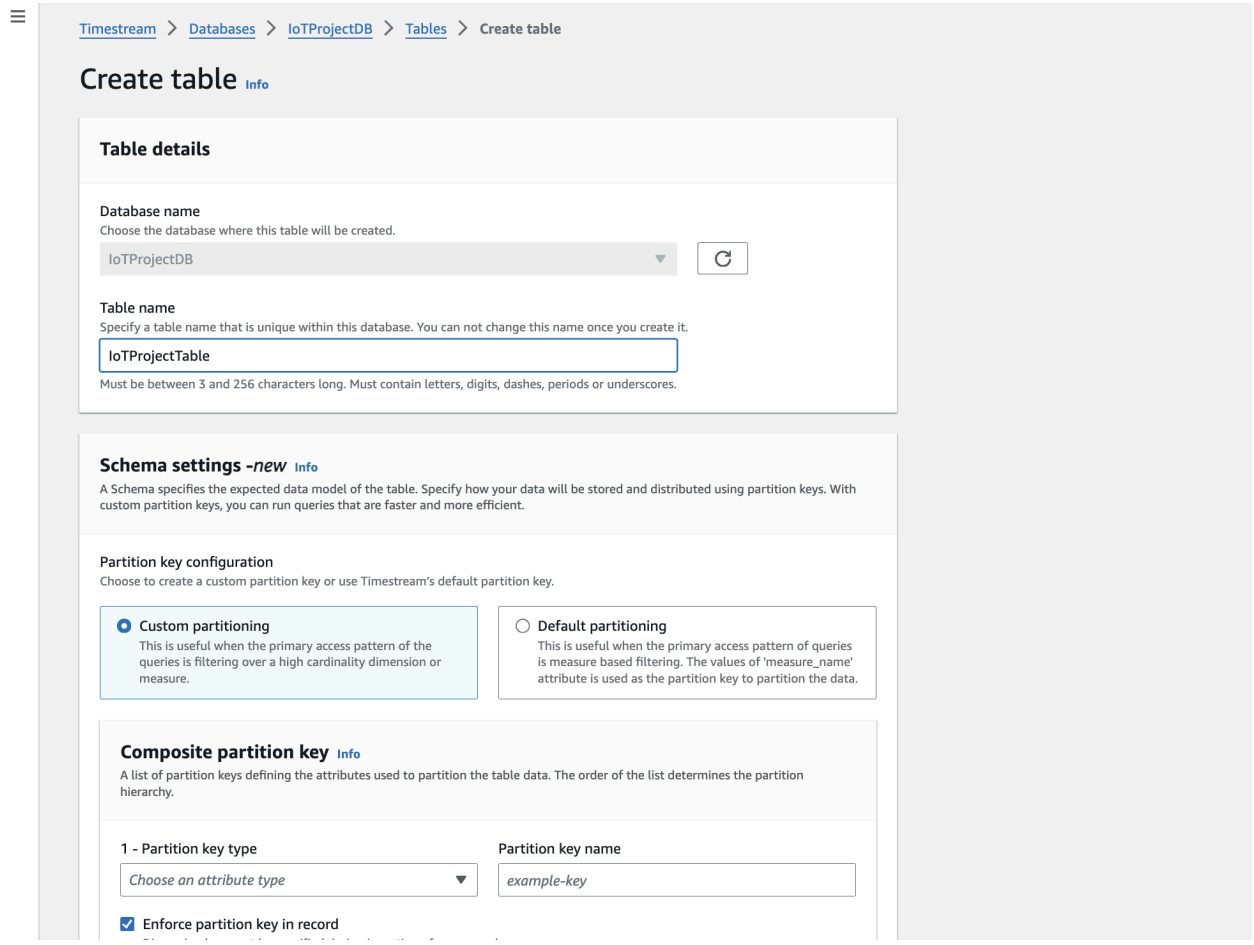


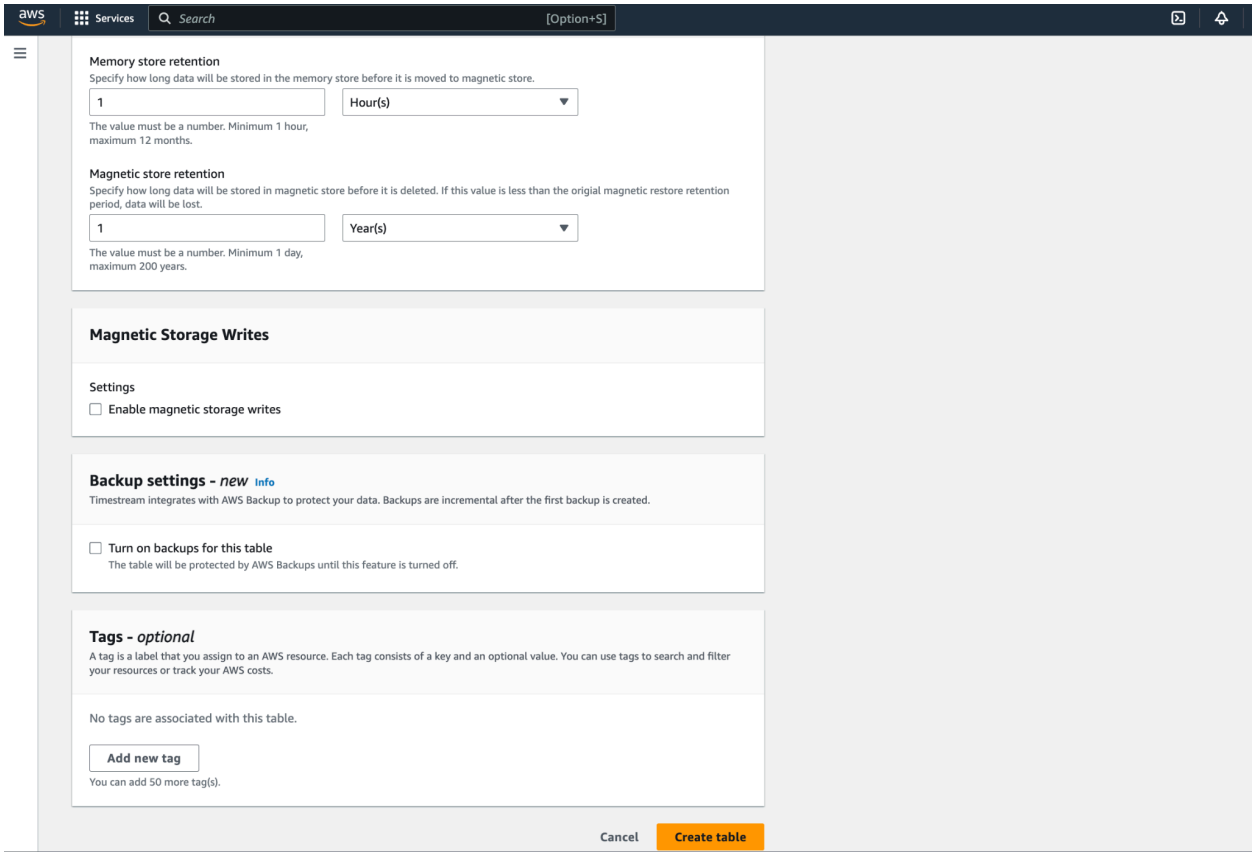




3. **Create a Table**:
    - After the database is created, click **Create Table**.
    - Name the table (e.g., `IoTProjectTable`), and configure the memory/magnetic storage options for storing time-series data.

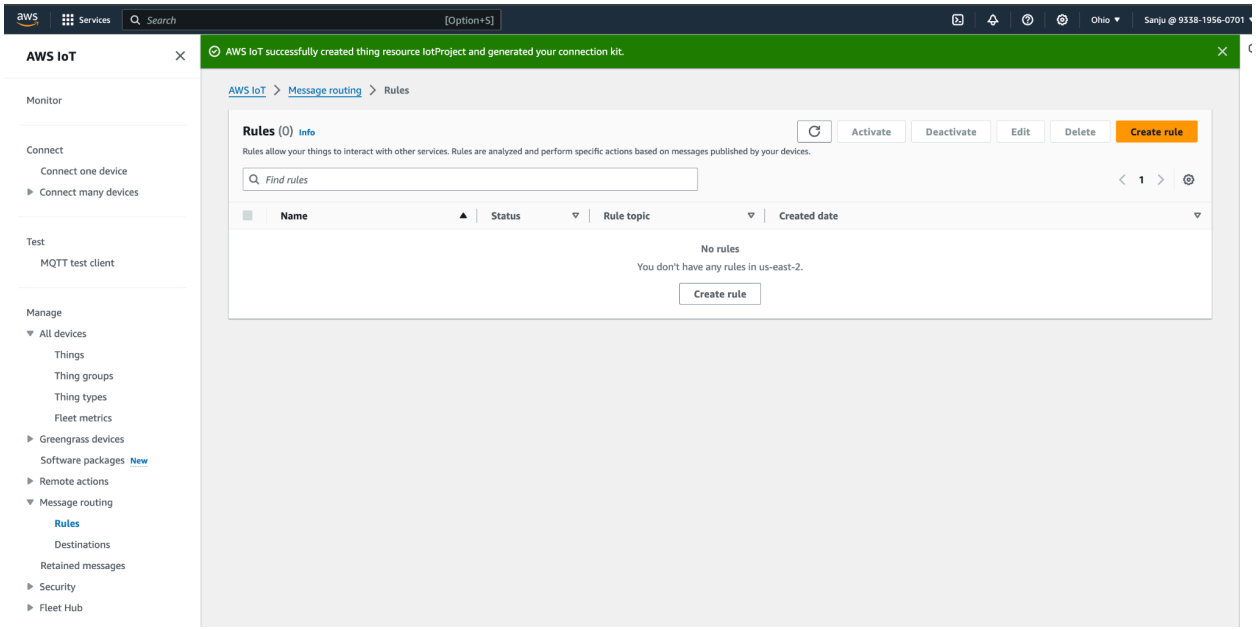

Amazon Timestream
Service dashboard
LiveAnalytics
Admin dashboard
Resources
Databases
Tables
Backups
Management Tools
Batch load tasks
EventBridge Pipes New
Query editor
Scheduled queries
Monitoring
InfluxDB
InfluxDB databases New
Parameter groups
Getting started
Tutorials
Timestream > Databases > IoTProjectDB
IoTProjectDB
Edit
Delete
Summary
Name
IoTProjectDB
Modified time (UTC)
9/8/2024, 7:52:24 AM
Creation time (UTC)
9/8/2024, 7:52:24 AM
Database ARN
arn:aws:timestream:us-east-2:933819560701:database/IoTProjectDB
KMS key ARN
arn:aws:kms:us-east-2:933819560701:key/ecedc4e4-fcb4-4cd9-bcb5-33aa186e71b6
Monitoring
Tables
Tags
Tables (0) Info
Create backup
Create scheduled query
Edit
Delete
Create table
Filter
Table name
Creation time (UTC)
No tables
Tables you create will appear here. A database is required to create a table.
Create table


Timestream > Databases > IoTProjectDB > Tables > Create table
Create table Info
Table details
Database name
Choose the database where this table will be created.
IoTProjectDB
Table name
Specify a table name that is unique within this database. You can not change this name once you create it.
IoTProjectTable
Must be between 3 and 256 characters long. Must contain letters, digits, dashes, periods or underscores.
Schema settings -new Info
A Schema specifies the expected data model of the table. Specify how your data will be stored and distributed using partition keys. With custom partition keys, you can run queries that are faster and more efficient.
Partition key configuration
Choose to create a custom partition key or use Timestream's default partition key.
Custom partitioning
This is useful when the primary access pattern of the queries is filtering over a high cardinality dimension or measure.
Default partitioning
This is useful when the primary access pattern of queries is measure based filtering. The values of 'measure_name' attribute is used as the partition key to partition the data.
Composite partition key Info
A list of partition keys defining the attributes used to partition the table data. The order of the list determines the partition hierarchy.
1 - Partition key type
Choose an attribute type
Partition key name
example-key
Enforce partition key in record

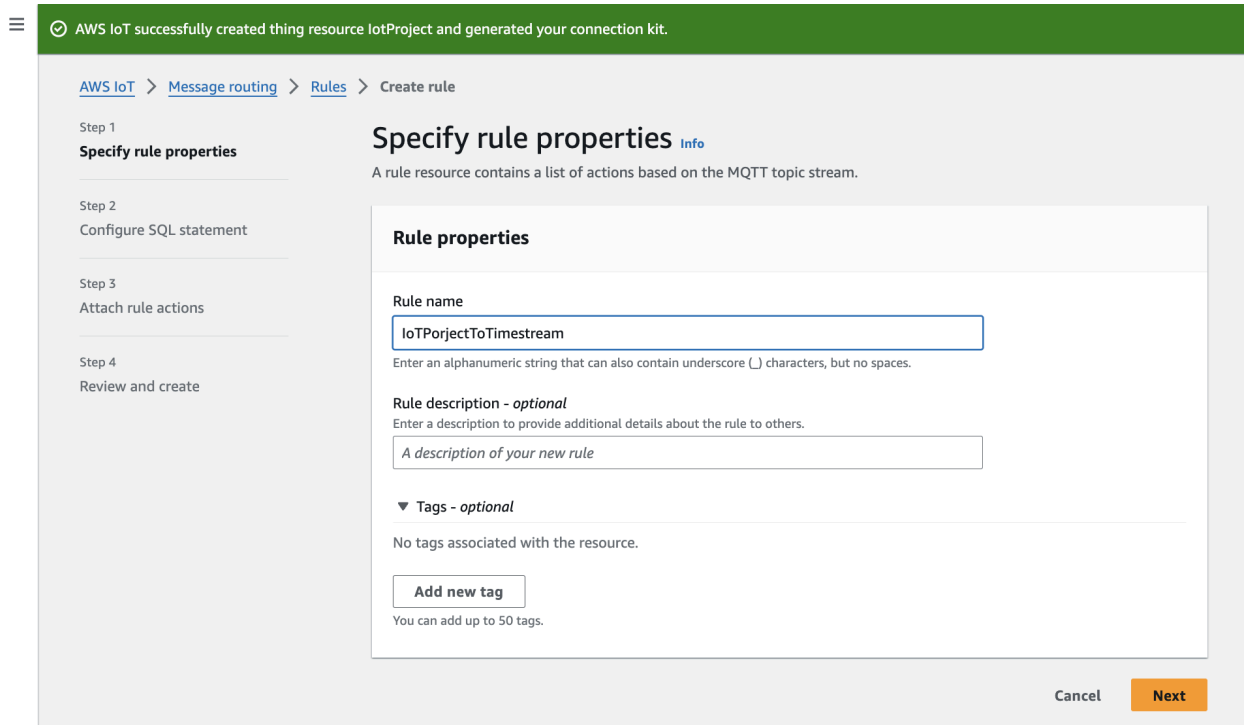


Create an IoT Core Rule:

- In the **AWS IoT Core Console**, go to **Message routing > Rules > Create a Rule**.

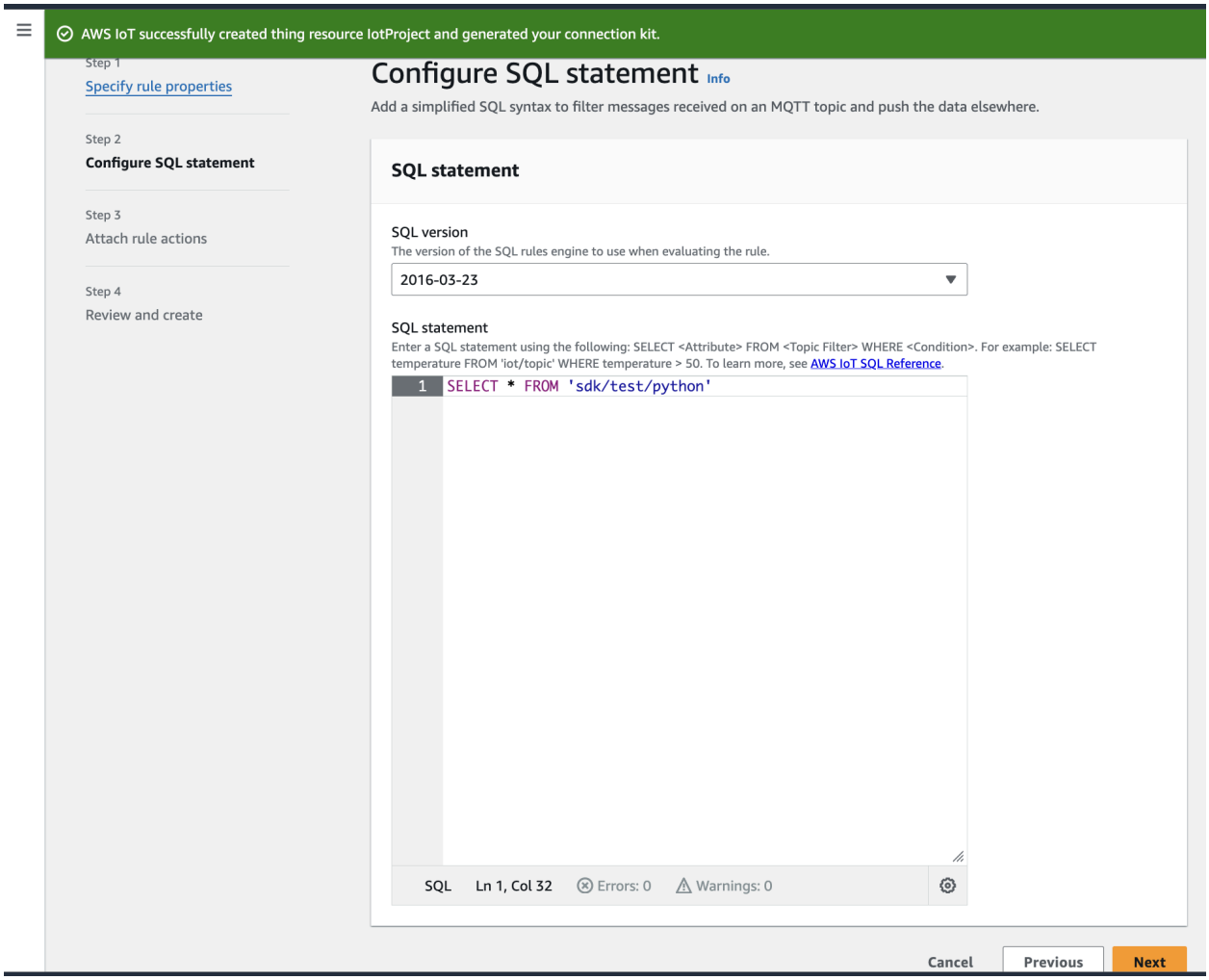


- Name your rule (e.g., `IoTProjectToTimestream`).



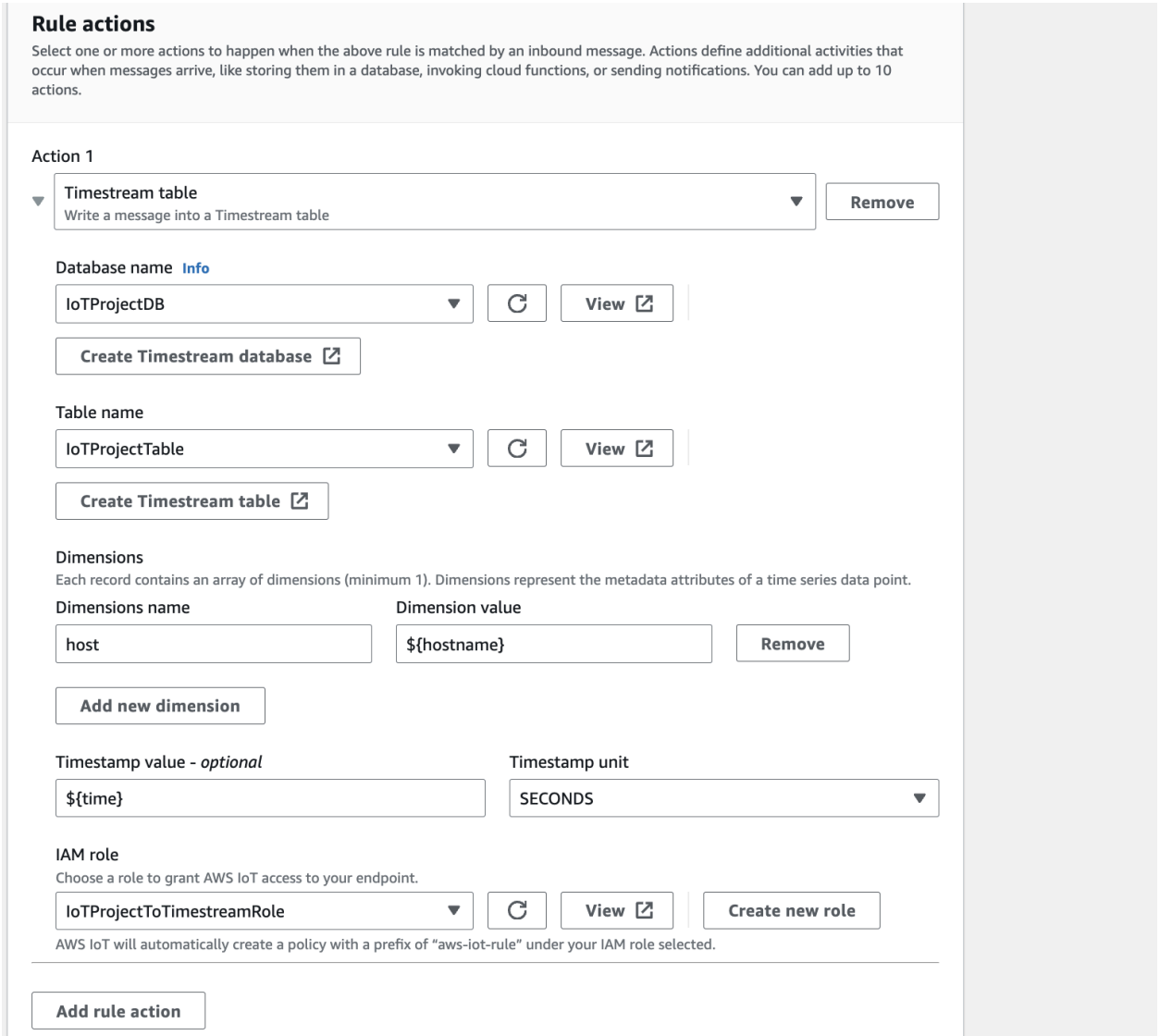
Define SQL Statement:

- In the SQL editor, select the incoming MQTT messages from your topic (e.g., `SDK/test/python`):

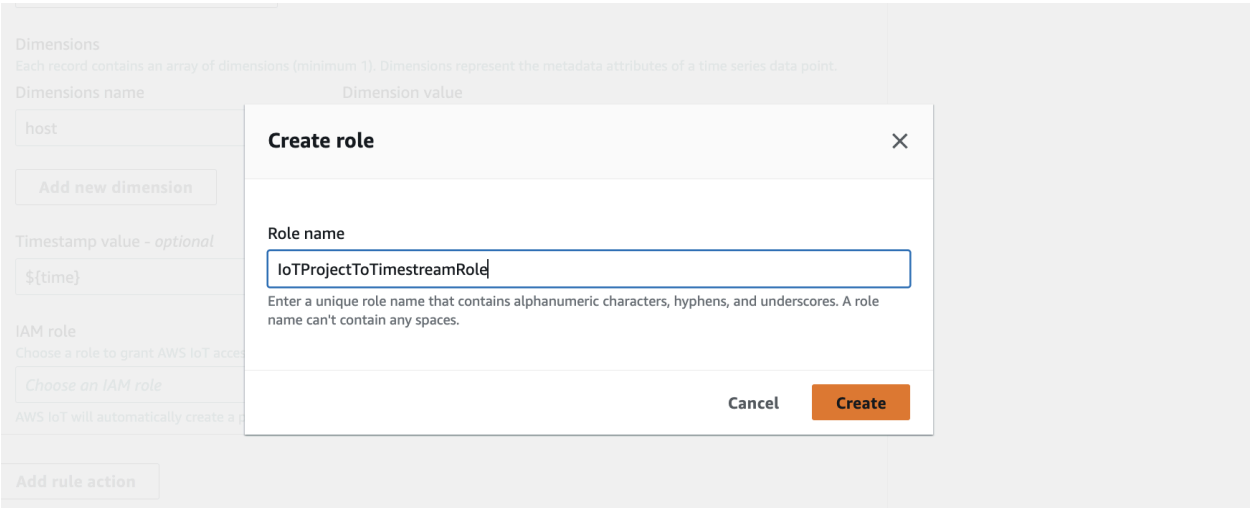


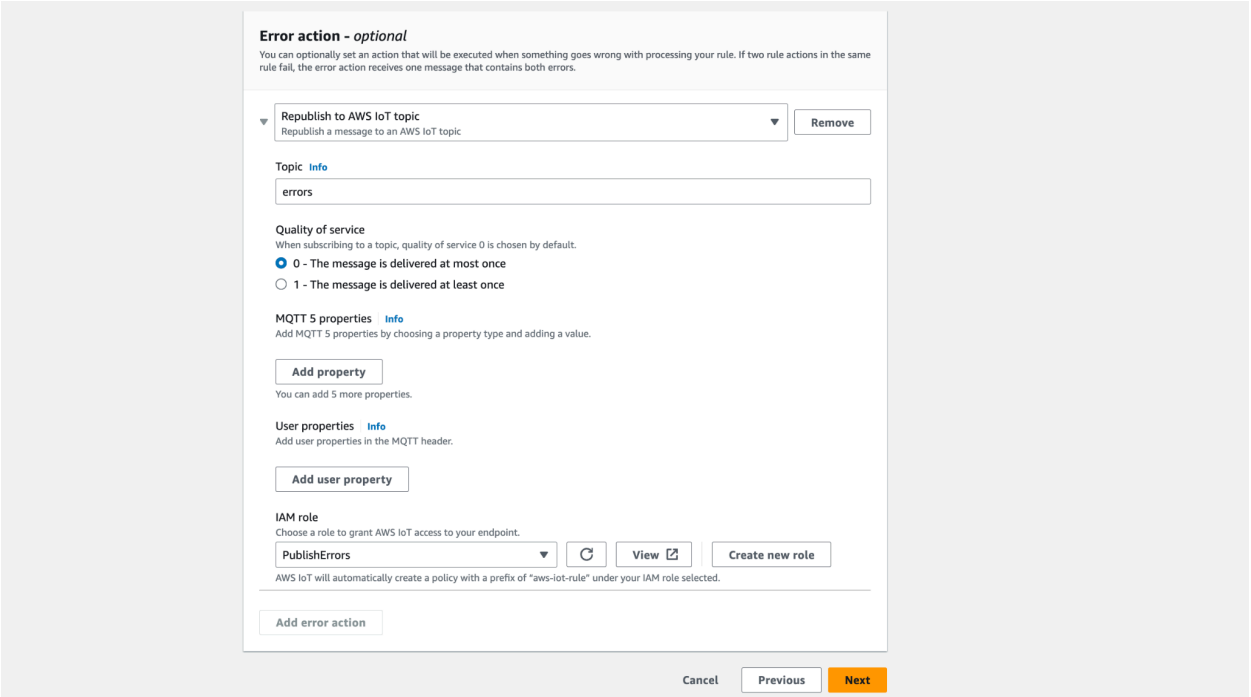
Configure Rule Action:

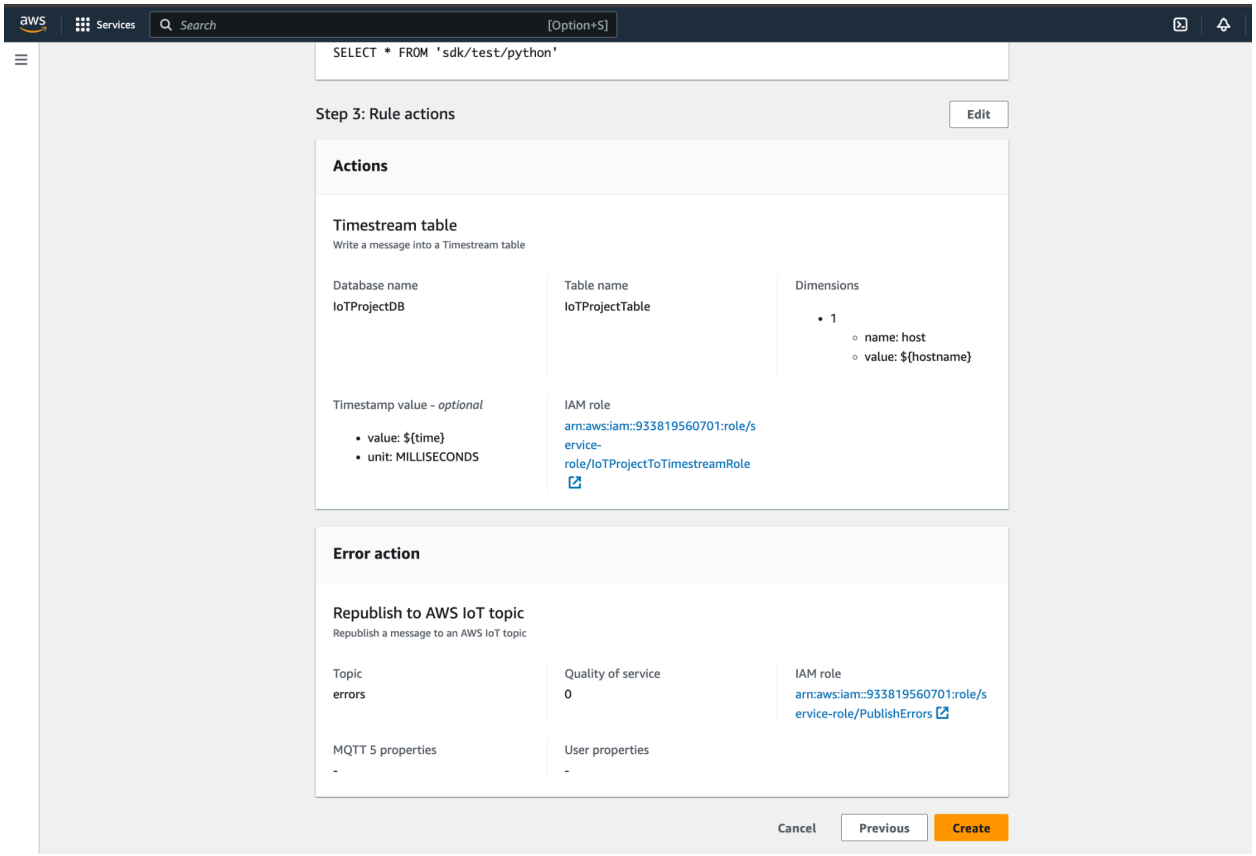
- Set the rule action to route the data to **Amazon Timestream**.

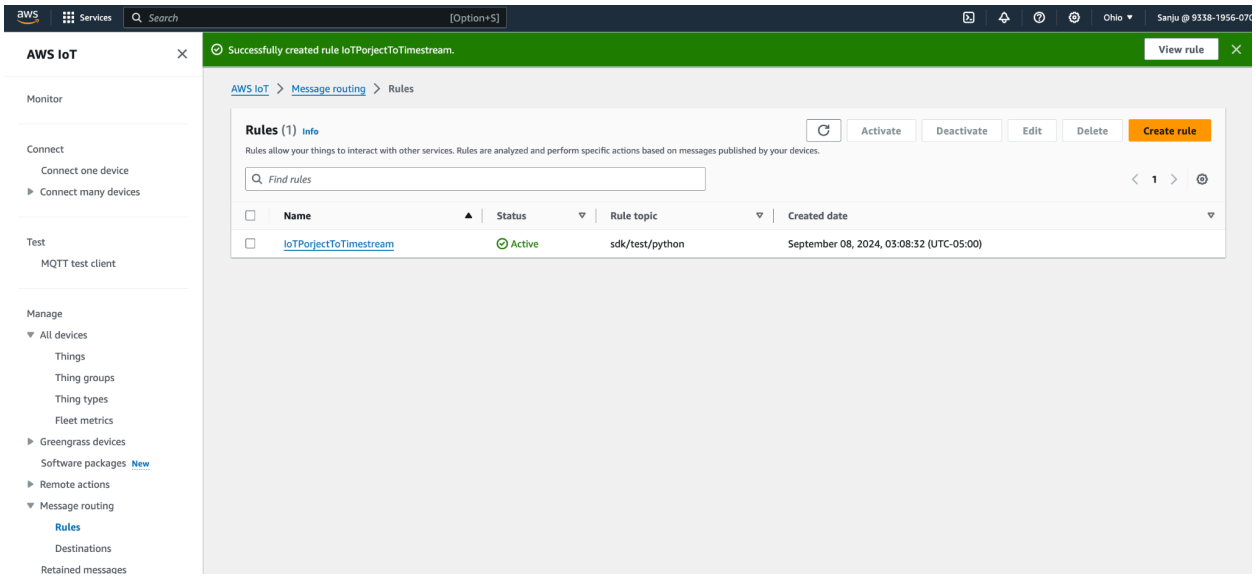


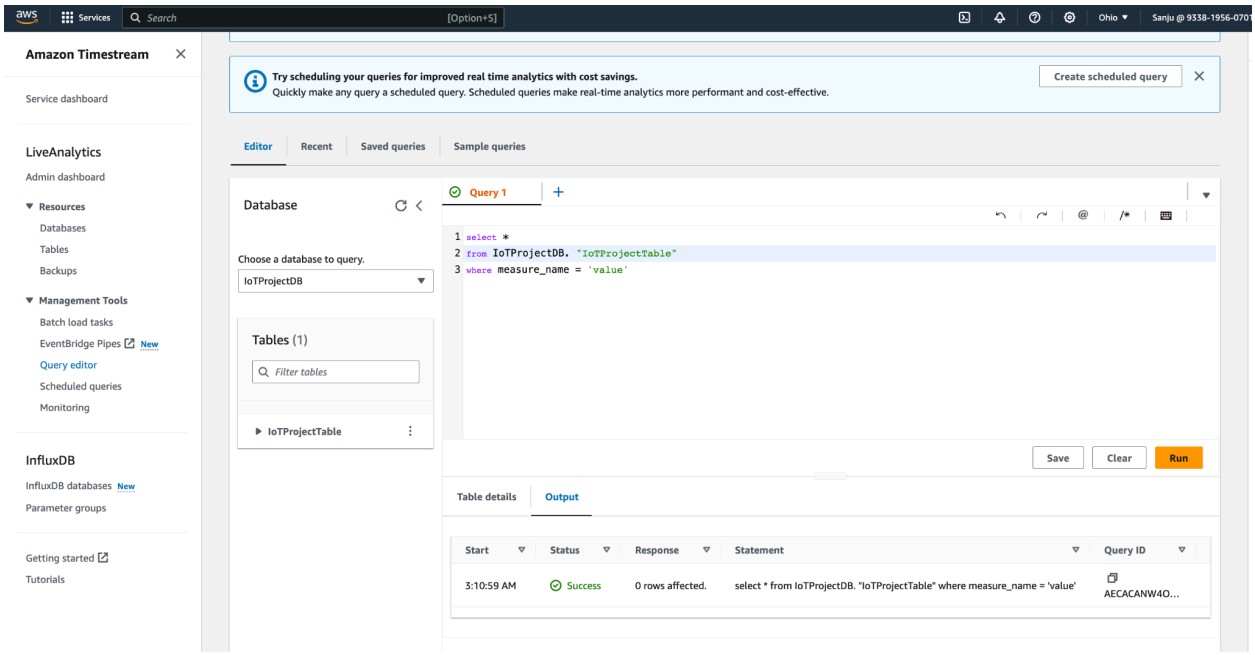
- Assign an **IAM role** that allows the rule to write data to the Timestream database.

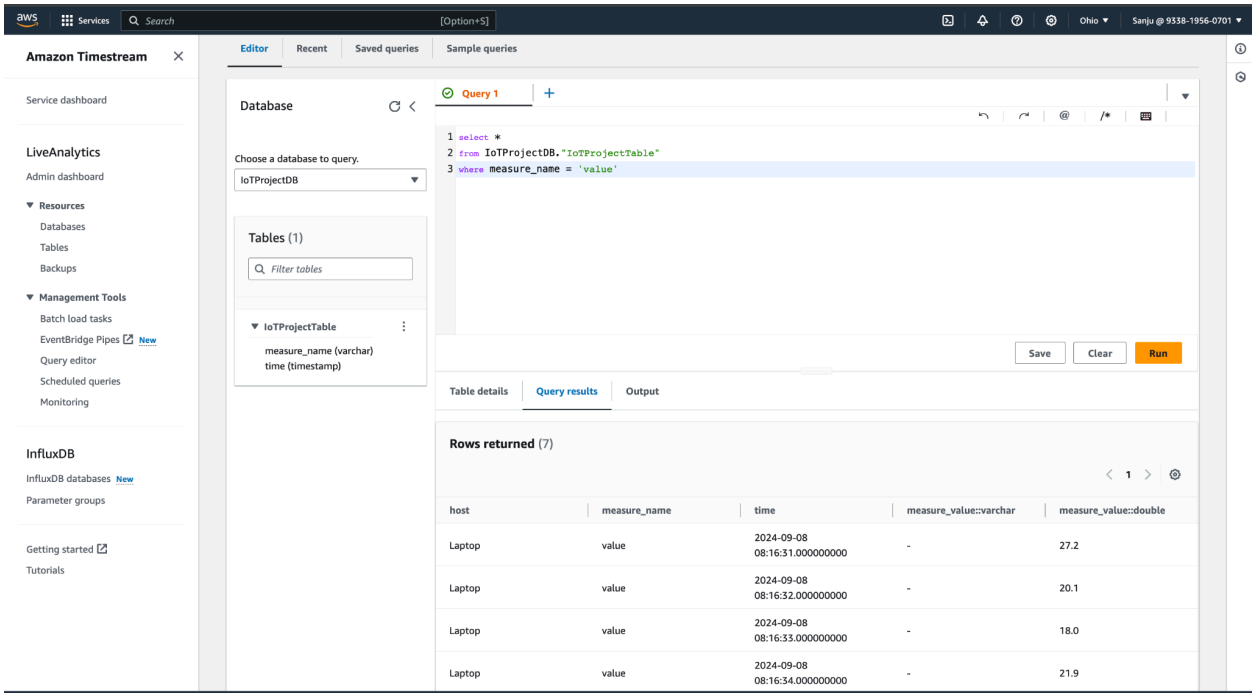

Dimensions
Each record contains an array of dimensions (minimum 1). Dimensions represent the metadata attributes of a time series data point.
Dimensions name
Dimension value
host
Add new dimension
Timestamp value - optional
${time}
IAM role
Choose a role to grant AWS IoT acces
Choose an IAM role
AWS IoT will automatically create a p
Add rule action
Create role
Role name
IoTProjectToTimestreamRole
Enter a unique role name that contains alphanumeric characters, hyphens, and underscores. A role name can't contain any spaces.
Cancel
Create


Error action - optional
You can optionally set an action that will be executed when something goes wrong with processing your rule. If two rule actions in the same rule fail, the error action receives one message that contains both errors.
Republish to AWS IoT topic
Republish a message to an AWS IoT topic
Remove
Topic Info
errors
Quality of service
When subscribing to a topic, quality of service 0 is chosen by default.
0 - The message is delivered at most once
1 - The message is delivered at least once
MQTT 5 properties Info
Add MQTT 5 properties by choosing a property type and adding a value.
Add property
You can add 5 more properties.
User properties Info
Add user properties in the MQTT header.
Add user property
IAM role
Choose a role to grant AWS IoT access to your endpoint.
PublishErrors
View
Create new role
AWS IoT will automatically create a policy with a prefix of "aws-iot-rule" under your IAM role selected.
Add error action
Cancel
Previous
Next


aws
Services
Search
[Option+S]
SELECT * FROM 'sdk/test/python'
Step 3: Rule actions
Edit
Actions
Timestream table
Write a message into a Timestream table
Database name
IoTProjectDB
Table name
IoTProjectTable
Dimensions
1
name: host
value: ${hostname}
Timestamp value - optional
value: ${time}
unit: MILLISECONDS
IAM role
arn:aws:iam::933819560701:role/service-role/IoTProjectToTimestreamRole
Error action
Republish to AWS IoT topic
Republish a message to an AWS IoT topic
Topic
errors
Quality of service
0
IAM role
arn:aws:iam::933819560701:role/service-role/PublishErrors
MQTT 5 properties
-
User properties
-
Cancel
Previous
Create

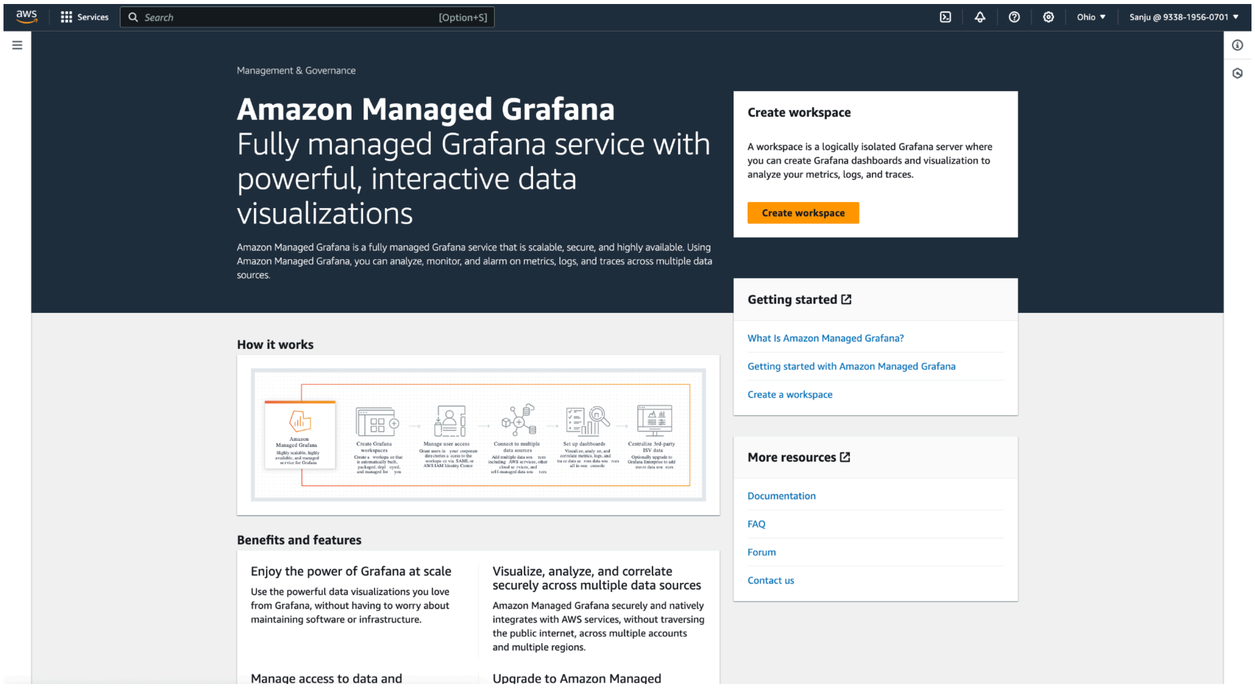

aws
Services
Search
[Option+S]
Ohio
AWS IoT
Monitor
Connect
Connect one device
Connect many devices
Test
MQTT test client
Manage
All devices
Things
Thing groups
Thing types
Fleet metrics
Greengrass devices
Software packages New
Remote actions
Message routing
Rules
Destinations
Retained messages
Successfully created rule IoTPorjectToTimestream.
View rule
AWS IoT > Message routing > Rules
Rules (1) Info
Rules allow your things to interact with other services. Rules are analyzed and perform specific actions based on messages published by your devices.
Activate
Deactivate
Edit
Delete
Create rule
Find rules
Name
Status
Rule topic
Created date
IoTPorjectToTimestream
Active
sdk/test/python
September 08, 2024, 03:08:32 (UTC-05:00)

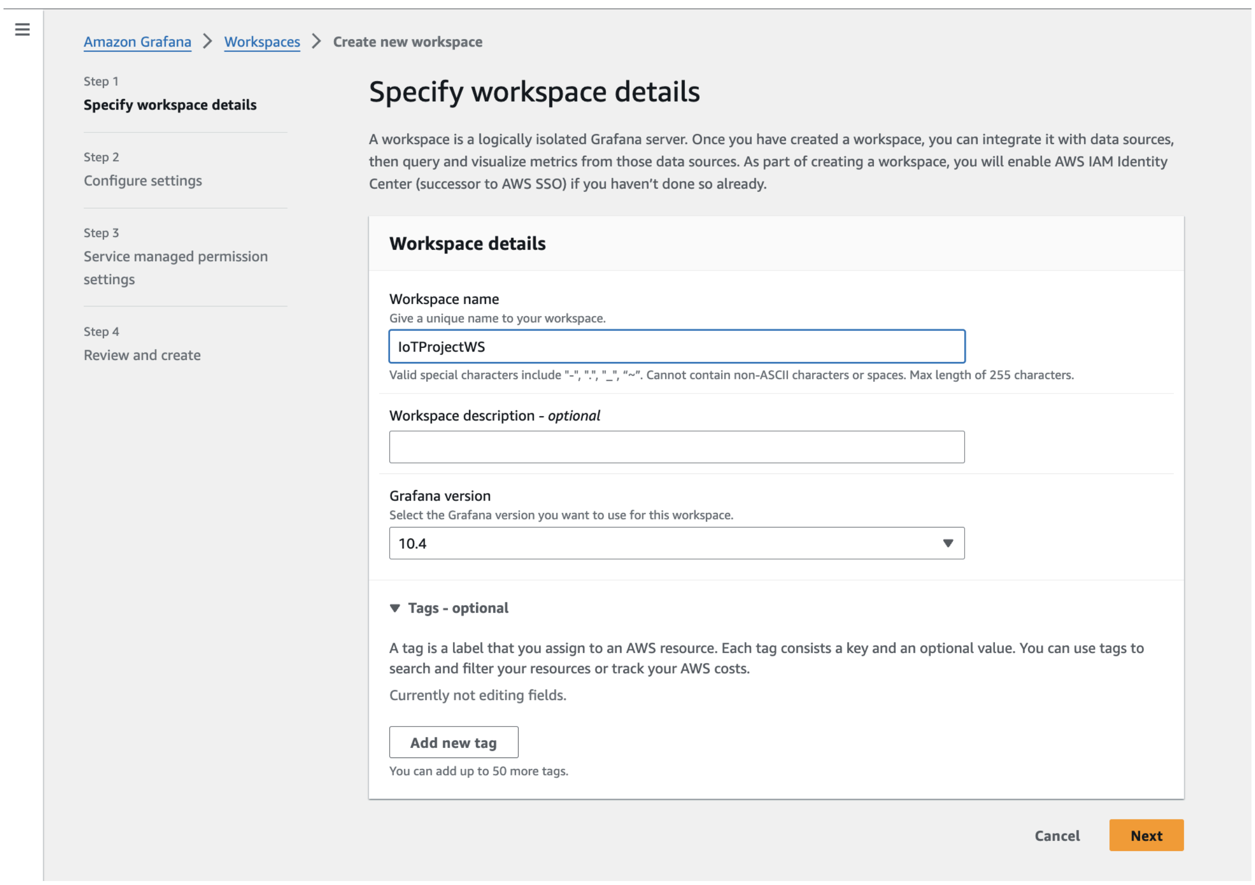


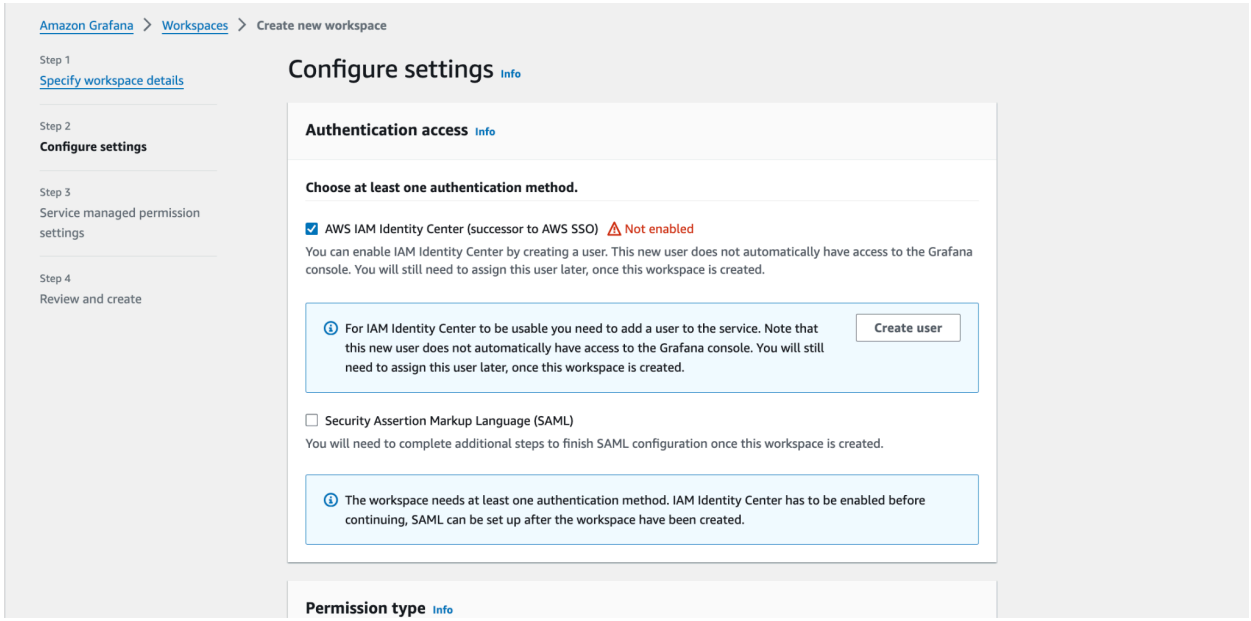


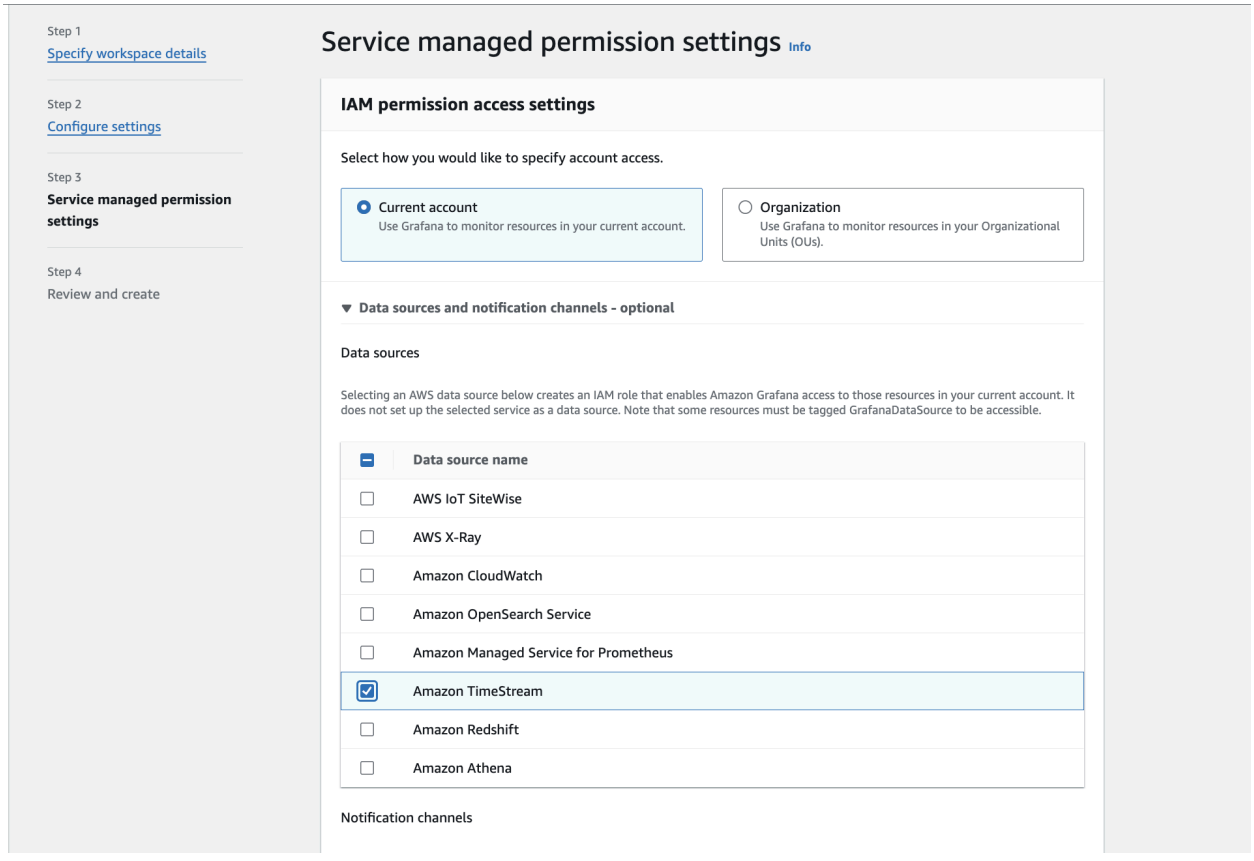
3.4 Visualizing Data with Amazon Managed Grafana

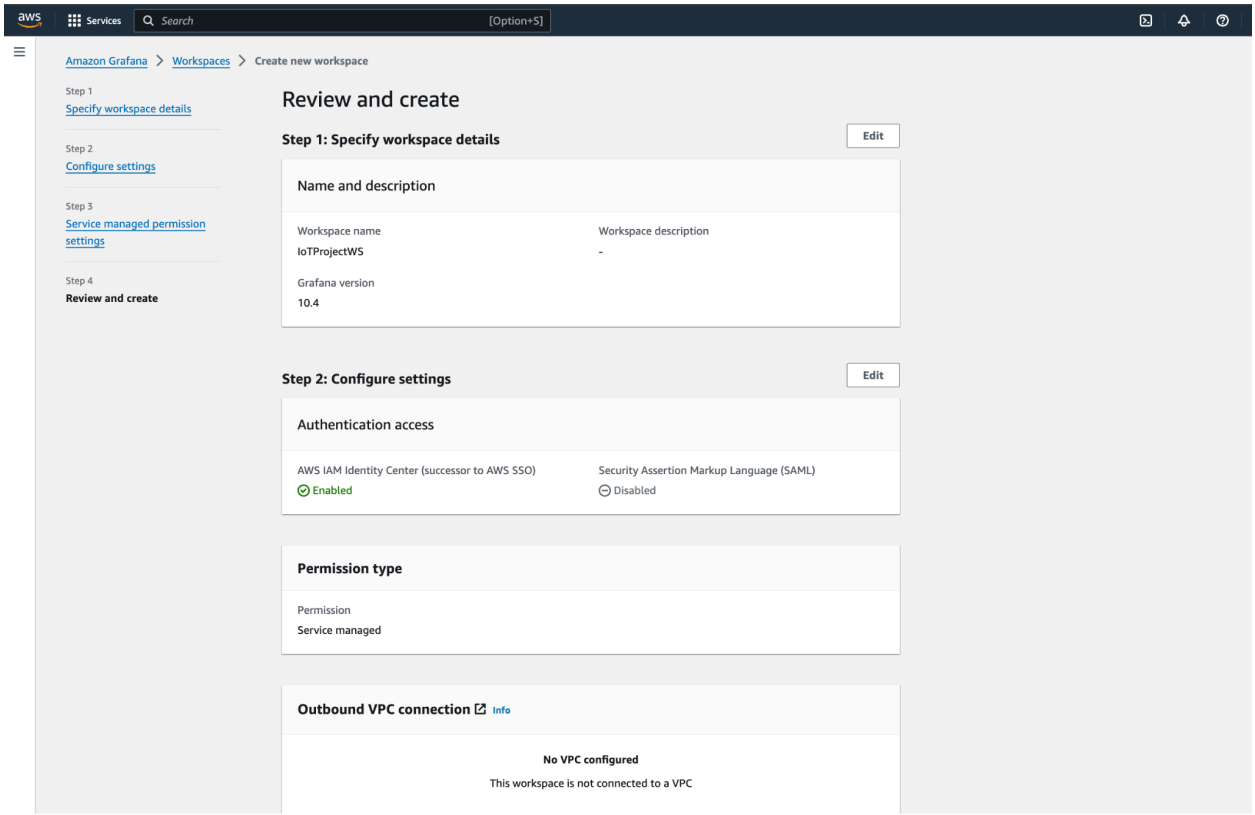
1. Set up a new Amazon Managed Grafana workspace.


Services
Search
[Option+S]
Ohio
Sanju @ 9338-1956-0701
Management & Governance
Amazon Managed Grafana
Fully managed Grafana service with powerful, interactive data visualizations
Amazon Managed Grafana is a fully managed Grafana service that is scalable, secure, and highly available. Using Amazon Managed Grafana, you can analyze, monitor, and alarm on metrics, logs, and traces across multiple data sources.
Create workspace
A workspace is a logically isolated Grafana server where you can create Grafana dashboards and visualization to analyze your metrics, logs, and traces.
Create workspace
Getting started
What Is Amazon Managed Grafana?
Getting started with Amazon Managed Grafana
Create a workspace
More resources
Documentation
FAQ
Forum
Contact us
How it works
Benefits and features
Enjoy the power of Grafana at scale
Use the powerful data visualizations you love from Grafana, without having to worry about maintaining software or infrastructure.
Visualize, analyze, and correlate securely across multiple data sources
Amazon Managed Grafana securely and natively integrates with AWS services, without traversing the public internet, across multiple accounts and multiple regions.
Manage access to data and
Upgrade to Amazon Managed


Amazon Grafana
Workspaces
Create new workspace
Step 1
Specify workspace details
Step 2
Configure settings
Step 3
Service managed permission settings
Step 4
Review and create
Specify workspace details
A workspace is a logically isolated Grafana server. Once you have created a workspace, you can integrate it with data sources, then query and visualize metrics from those data sources. As part of creating a workspace, you will enable AWS IAM Identity Center (successor to AWS SSO) if you haven't done so already.
Workspace details
Workspace name
Give a unique name to your workspace.
IoTProjectWS
Valid special characters include "-", ".", "_", "~". Cannot contain non-ASCII characters or spaces. Max length of 255 characters.
Workspace description - optional
Grafana version
Select the Grafana version you want to use for this workspace.
10.4
Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Currently not editing fields.
Add new tag
You can add up to 50 more tags.
Cancel
Next


Amazon Grafana > Workspaces > Create new workspace
Step 1
Specify workspace details
Step 2
Configure settings
Step 3
Service managed permission settings
Step 4
Review and create
Configure settings Info
Authentication access Info
Choose at least one authentication method.
AWS IAM Identity Center (successor to AWS SSO) Not enabled
You can enable IAM Identity Center by creating a user. This new user does not automatically have access to the Grafana console. You will still need to assign this user later, once this workspace is created.
For IAM Identity Center to be usable you need to add a user to the service. Note that this new user does not automatically have access to the Grafana console. You will still need to assign this user later, once this workspace is created.
Create user
Security Assertion Markup Language (SAML)
You will need to complete additional steps to finish SAML configuration once this workspace is created.
The workspace needs at least one authentication method. IAM Identity Center has to be enabled before continuing, SAML can be set up after the workspace have been created.
Permission type Info


Step 1
Specify workspace details
Step 2
Configure settings
Step 3
Service managed permission settings
Step 4
Review and create
Service managed permission settings Info
IAM permission access settings
Select how you would like to specify account access.
Current account
Use Grafana to monitor resources in your current account.
Organization
Use Grafana to monitor resources in your Organizational Units (OUs).
Data sources and notification channels - optional
Data sources
Selecting an AWS data source below creates an IAM role that enables Amazon Grafana access to those resources in your current account. It does not set up the selected service as a data source. Note that some resources must be tagged GrafanaDataSource to be accessible.
Data source name
AWS IoT SiteWise
AWS X-Ray
Amazon CloudWatch
Amazon OpenSearch Service
Amazon Managed Service for Prometheus
Amazon TimeStream
Amazon Redshift
Amazon Athena
Notification channels



2. Connect your Timestream database as a data source in Grafana.
3. Create a new dashboard and panel to visualize the incoming data in real-time.

---

4. Summary

In this demonstration, we created an AWS IoT Core device, routed its telemetry data to Amazon Timestream, and visualized the data using Amazon Managed Grafana. This end-to-end solution showcases how to set up a scalable, secure, and efficient IoT data pipeline using only three AWS services.

---

5. Additional Resources

- AWS IoT Core Documentation
- Amazon Timestream Documentation
- Amazon Managed Grafana Documentation