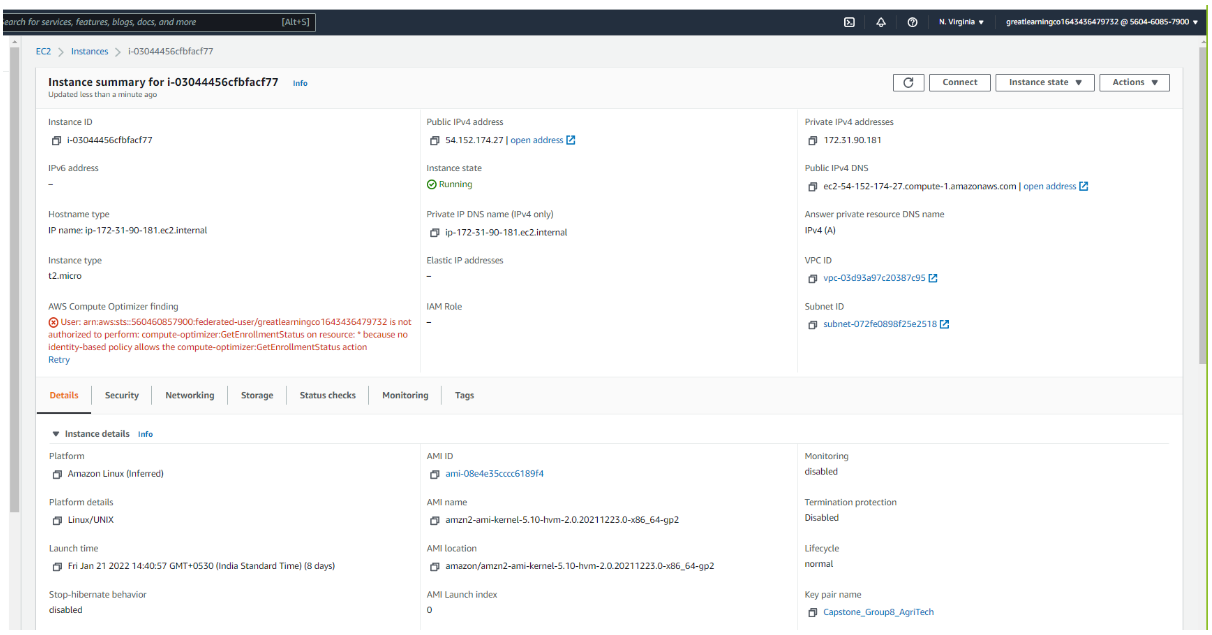
Screenshots of AWS Components and Configurations



earch for services, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
EC2 > Instances > i-03044456cfbfacf77
Connect
Instance state
Actions
Details
Security
Networking
Storage
Status checks
Monitoring
Tags

Instance summary for i-03044456cfbfacf77 Info
Updated less than a minute ago

Instance ID
i-03044456cfbfacf77

Public IPv4 address
54.152.174.27 | open address

Private IPv4 addresses
172.31.90.181

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-54-152-174-27.compute-1.amazonaws.com | open address

Hostname type
IP name: ip-172-31-90-181.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-90-181.ec2.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses
–

VPC ID
vpc-03d93a97c20387c95

AWS Compute Optimizer finding
User: arn:aws:sts::560460857900:federated-user/greatlearningco1643436479732 is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy allows the compute-optimizer:GetEnrollmentStatus action
Retry

IAM Role
–

Subnet ID
subnet-072fe0898f25e2518

Instance details Info

Platform
Amazon Linux (Inferred)

AMI ID
ami-08e4e35cccc6189f4

Monitoring
disabled

Platform details
Linux/UNIX

AMI name
amzn2-ami-kernel-5.10-hvm-2.0.20211223.0-x86_64-gp2

Termination protection
Disabled

Launch time
Fri Jan 21 2022 14:40:57 GMT+0530 (India Standard Time) (8 days)

AMI location
amazon/amzn2-ami-kernel-5.10-hvm-2.0.20211223.0-x86_64-gp2

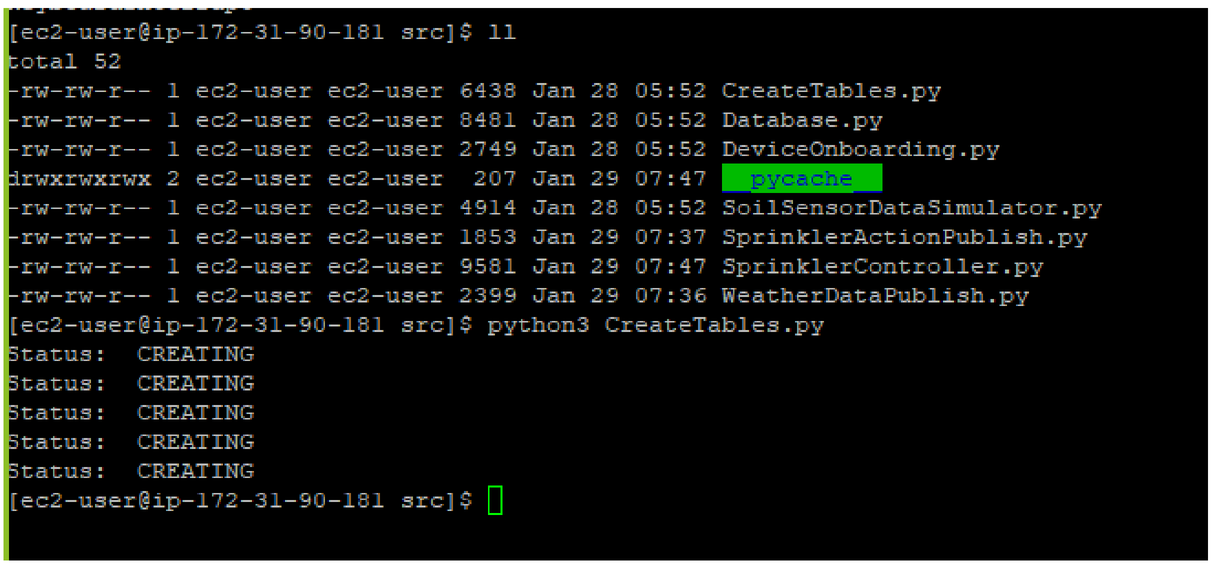
Lifecycle
normal

Stop-hibernate behavior
disabled

AMI Launch index
0

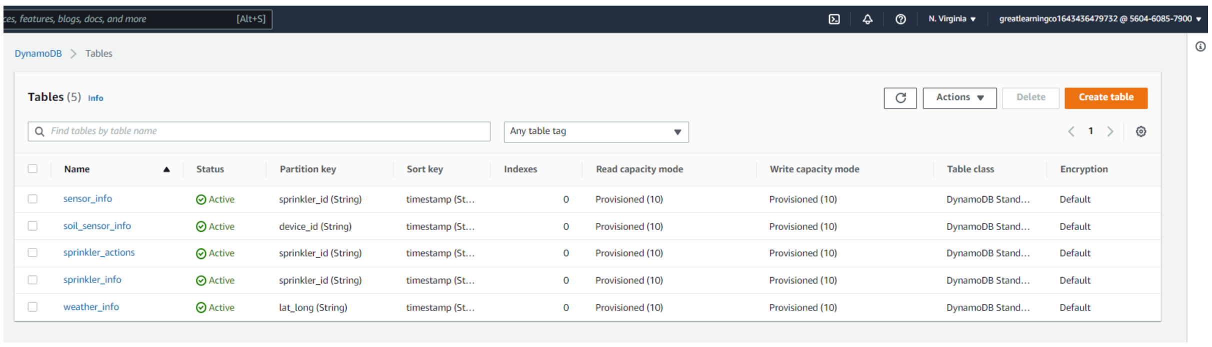
Key pair name
Capstone_Group8_AgriTech

1. EC2 instance where all the python modules are running



```
[ec2-user@ip-172-31-90-181 src]$ ll
total 52
-rw-rw-r-- 1 ec2-user ec2-user 6438 Jan 28 05:52 CreateTables.py
-rw-rw-r-- 1 ec2-user ec2-user 8481 Jan 28 05:52 Database.py
-rw-rw-r-- 1 ec2-user ec2-user 2749 Jan 28 05:52 DeviceOnboarding.py
drwxrwxrwx 2 ec2-user ec2-user  207 Jan 29 07:47  pycache
-rw-rw-r-- 1 ec2-user ec2-user 4914 Jan 28 05:52 SoilSensorDataSimulator.py
-rw-rw-r-- 1 ec2-user ec2-user 1853 Jan 29 07:37 SprinklerActionPublish.py
-rw-rw-r-- 1 ec2-user ec2-user 9581 Jan 29 07:47 SprinklerController.py
-rw-rw-r-- 1 ec2-user ec2-user 2399 Jan 29 07:36 WeatherDataPublish.py
[ec2-user@ip-172-31-90-181 src]$ python3 CreateTables.py
Status:  CREATING
Status:  CREATING
Status:  CREATING
Status:  CREATING
Status:  CREATING
[ec2-user@ip-172-31-90-181 src]$ 
```

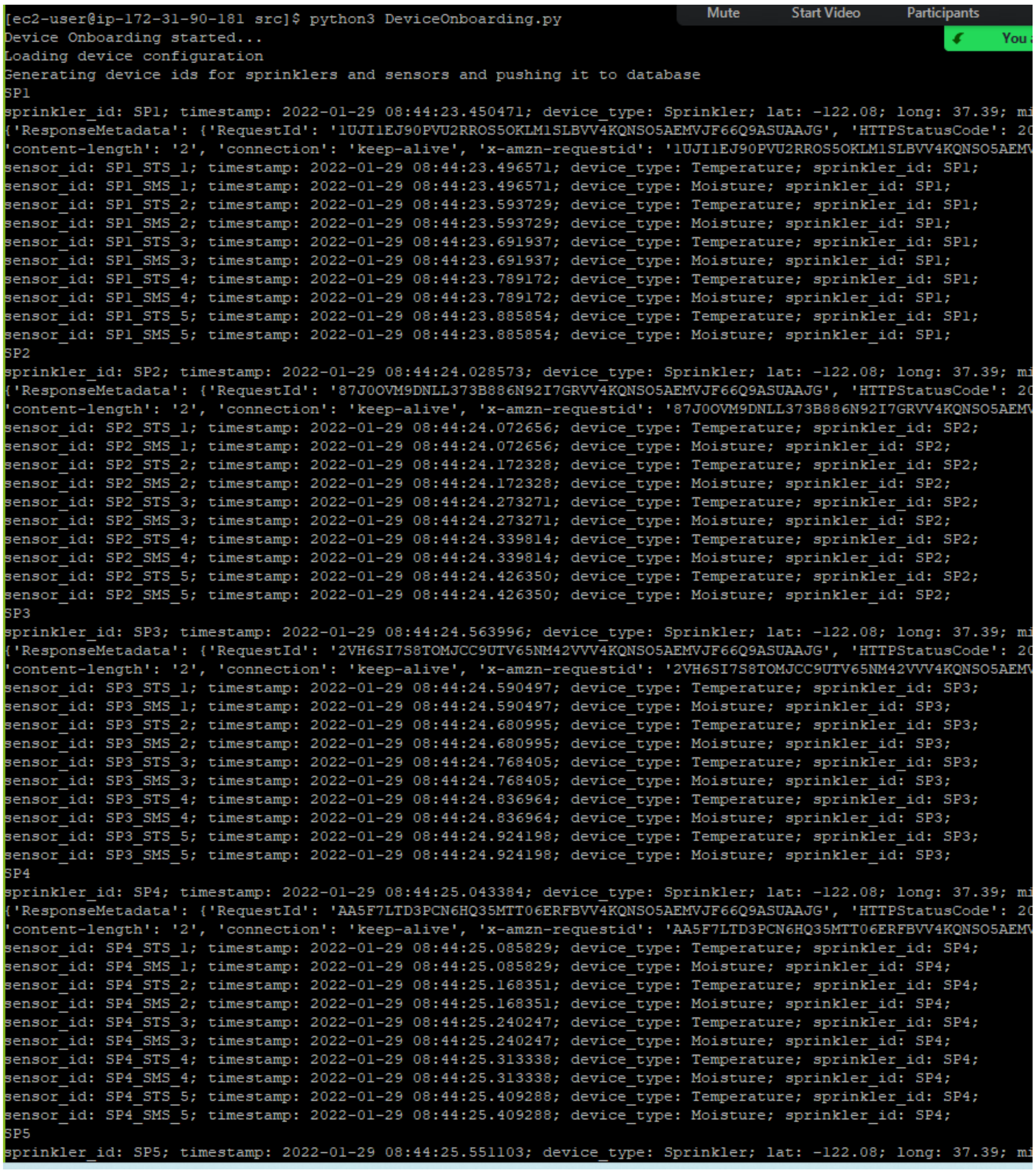
2. EC2 instance connected via Putty and ran modules to create tables used in project



es, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
DynamoDB > Tables
Tables (5) Info
Actions
Delete
Create table
Find tables by table name
Any table tag
1

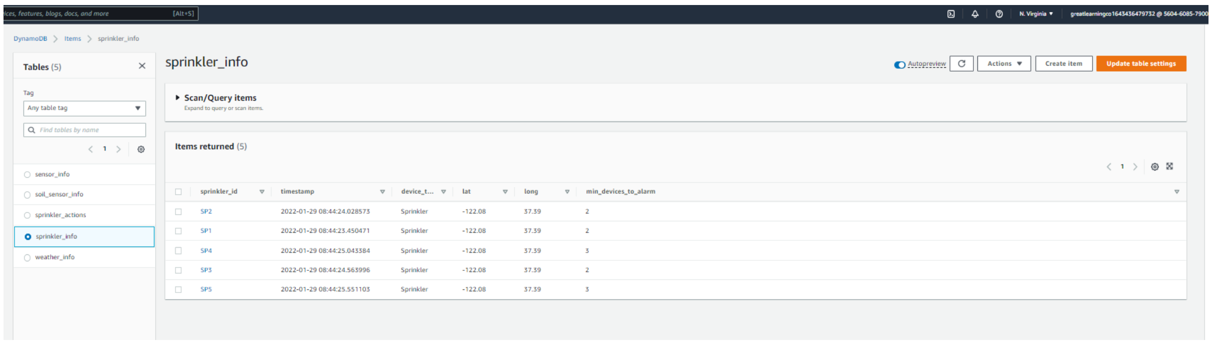
	Name	Status	Partition key	Sort key	Indexes	Read capacity mode	Write capacity mode	Table class	Encryption
☐	sensor_info	✓ Active	sprinkler_id (String)	timestamp (St…	0	Provisioned (10)	Provisioned (10)	DynamoDB Stand…	Default
☐	soil_sensor_info	✓ Active	device_id (String)	timestamp (St…	0	Provisioned (10)	Provisioned (10)	DynamoDB Stand…	Default
☐	sprinkler_actions	✓ Active	sprinkler_id (String)	timestamp (St…	0	Provisioned (10)	Provisioned (10)	DynamoDB Stand…	Default
☐	sprinkler_info	✓ Active	sprinkler_id (String)	timestamp (St…	0	Provisioned (10)	Provisioned (10)	DynamoDB Stand…	Default
☐	weather_info	✓ Active	lat_long (String)	timestamp (St…	0	Provisioned (10)	Provisioned (10)	DynamoDB Stand…	Default

3. DynamoDB where all tables are created



```
[ec2-user@ip-172-31-90-181 src]$ python3 DeviceOnboarding.py
Device Onboarding started...
Loading device configuration
Generating device ids for sprinklers and sensors and pushing it to database
SP1
sprinkler_id: SP1; timestamp: 2022-01-29 08:44:23.450471; device_type: Sprinkler; lat: -122.08; long: 37.39; mi
{'ResponseMetadata': {'RequestId': '1UJI1EJ90PVU2RROS5OKLM1SLBVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 20
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '1UJI1EJ90PVU2RROS5OKLM1SLBVV4KQNSO5AEMV
sensor_id: SP1_STS_1; timestamp: 2022-01-29 08:44:23.496571; device_type: Temperature; sprinkler_id: SP1;
sensor_id: SP1_SMS_1; timestamp: 2022-01-29 08:44:23.496571; device_type: Moisture; sprinkler_id: SP1;
sensor_id: SP1_STS_2; timestamp: 2022-01-29 08:44:23.593729; device_type: Temperature; sprinkler_id: SP1;
sensor_id: SP1_SMS_2; timestamp: 2022-01-29 08:44:23.593729; device_type: Moisture; sprinkler_id: SP1;
sensor_id: SP1_STS_3; timestamp: 2022-01-29 08:44:23.691937; device_type: Temperature; sprinkler_id: SP1;
sensor_id: SP1_SMS_3; timestamp: 2022-01-29 08:44:23.691937; device_type: Moisture; sprinkler_id: SP1;
sensor_id: SP1_STS_4; timestamp: 2022-01-29 08:44:23.789172; device_type: Temperature; sprinkler_id: SP1;
sensor_id: SP1_SMS_4; timestamp: 2022-01-29 08:44:23.789172; device_type: Moisture; sprinkler_id: SP1;
sensor_id: SP1_STS_5; timestamp: 2022-01-29 08:44:23.885854; device_type: Temperature; sprinkler_id: SP1;
sensor_id: SP1_SMS_5; timestamp: 2022-01-29 08:44:23.885854; device_type: Moisture; sprinkler_id: SP1;
SP2
sprinkler_id: SP2; timestamp: 2022-01-29 08:44:24.028573; device_type: Sprinkler; lat: -122.08; long: 37.39; mi
{'ResponseMetadata': {'RequestId': '87J0OVM9DNLL373B886N92I7GRVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 20
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '87J0OVM9DNLL373B886N92I7GRVV4KQNSO5AEMV
sensor_id: SP2_STS_1; timestamp: 2022-01-29 08:44:24.072656; device_type: Temperature; sprinkler_id: SP2;
sensor_id: SP2_SMS_1; timestamp: 2022-01-29 08:44:24.072656; device_type: Moisture; sprinkler_id: SP2;
sensor_id: SP2_STS_2; timestamp: 2022-01-29 08:44:24.172328; device_type: Temperature; sprinkler_id: SP2;
sensor_id: SP2_SMS_2; timestamp: 2022-01-29 08:44:24.172328; device_type: Moisture; sprinkler_id: SP2;
sensor_id: SP2_STS_3; timestamp: 2022-01-29 08:44:24.273271; device_type: Temperature; sprinkler_id: SP2;
sensor_id: SP2_SMS_3; timestamp: 2022-01-29 08:44:24.273271; device_type: Moisture; sprinkler_id: SP2;
sensor_id: SP2_STS_4; timestamp: 2022-01-29 08:44:24.339814; device_type: Temperature; sprinkler_id: SP2;
sensor_id: SP2_SMS_4; timestamp: 2022-01-29 08:44:24.339814; device_type: Moisture; sprinkler_id: SP2;
sensor_id: SP2_STS_5; timestamp: 2022-01-29 08:44:24.426350; device_type: Temperature; sprinkler_id: SP2;
sensor_id: SP2_SMS_5; timestamp: 2022-01-29 08:44:24.426350; device_type: Moisture; sprinkler_id: SP2;
SP3
sprinkler_id: SP3; timestamp: 2022-01-29 08:44:24.563996; device_type: Sprinkler; lat: -122.08; long: 37.39; mi
{'ResponseMetadata': {'RequestId': '2VH6SI7S8TOMJCC9UTV65NM42VVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 20
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '2VH6SI7S8TOMJCC9UTV65NM42VVV4KQNSO5AEMV
sensor_id: SP3_STS_1; timestamp: 2022-01-29 08:44:24.590497; device_type: Temperature; sprinkler_id: SP3;
sensor_id: SP3_SMS_1; timestamp: 2022-01-29 08:44:24.590497; device_type: Moisture; sprinkler_id: SP3;
sensor_id: SP3_STS_2; timestamp: 2022-01-29 08:44:24.680995; device_type: Temperature; sprinkler_id: SP3;
sensor_id: SP3_SMS_2; timestamp: 2022-01-29 08:44:24.680995; device_type: Moisture; sprinkler_id: SP3;
sensor_id: SP3_STS_3; timestamp: 2022-01-29 08:44:24.768405; device_type: Temperature; sprinkler_id: SP3;
sensor_id: SP3_SMS_3; timestamp: 2022-01-29 08:44:24.768405; device_type: Moisture; sprinkler_id: SP3;
sensor_id: SP3_STS_4; timestamp: 2022-01-29 08:44:24.836964; device_type: Temperature; sprinkler_id: SP3;
sensor_id: SP3_SMS_4; timestamp: 2022-01-29 08:44:24.836964; device_type: Moisture; sprinkler_id: SP3;
sensor_id: SP3_STS_5; timestamp: 2022-01-29 08:44:24.924198; device_type: Temperature; sprinkler_id: SP3;
sensor_id: SP3_SMS_5; timestamp: 2022-01-29 08:44:24.924198; device_type: Moisture; sprinkler_id: SP3;
SP4
sprinkler_id: SP4; timestamp: 2022-01-29 08:44:25.043384; device_type: Sprinkler; lat: -122.08; long: 37.39; mi
{'ResponseMetadata': {'RequestId': 'AA5F7LTD3PCN6HQ35MTT06ERFBVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 20
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': 'AA5F7LTD3PCN6HQ35MTT06ERFBVV4KQNSO5AEMV
sensor_id: SP4_STS_1; timestamp: 2022-01-29 08:44:25.085829; device_type: Temperature; sprinkler_id: SP4;
sensor_id: SP4_SMS_1; timestamp: 2022-01-29 08:44:25.085829; device_type: Moisture; sprinkler_id: SP4;
sensor_id: SP4_STS_2; timestamp: 2022-01-29 08:44:25.168351; device_type: Temperature; sprinkler_id: SP4;
sensor_id: SP4_SMS_2; timestamp: 2022-01-29 08:44:25.168351; device_type: Moisture; sprinkler_id: SP4;
sensor_id: SP4_STS_3; timestamp: 2022-01-29 08:44:25.240247; device_type: Temperature; sprinkler_id: SP4;
sensor_id: SP4_SMS_3; timestamp: 2022-01-29 08:44:25.240247; device_type: Moisture; sprinkler_id: SP4;
sensor_id: SP4_STS_4; timestamp: 2022-01-29 08:44:25.313338; device_type: Temperature; sprinkler_id: SP4;
sensor_id: SP4_SMS_4; timestamp: 2022-01-29 08:44:25.313338; device_type: Moisture; sprinkler_id: SP4;
sensor_id: SP4_STS_5; timestamp: 2022-01-29 08:44:25.409288; device_type: Temperature; sprinkler_id: SP4;
sensor_id: SP4_SMS_5; timestamp: 2022-01-29 08:44:25.409288; device_type: Moisture; sprinkler_id: SP4;
SP5
sprinkler_id: SP5; timestamp: 2022-01-29 08:44:25.551103; device_type: Sprinkler; lat: -122.08; long: 37.39; mi
```

4. Device Onboarding module which populates database with sprinkler, sensor and their mapping.

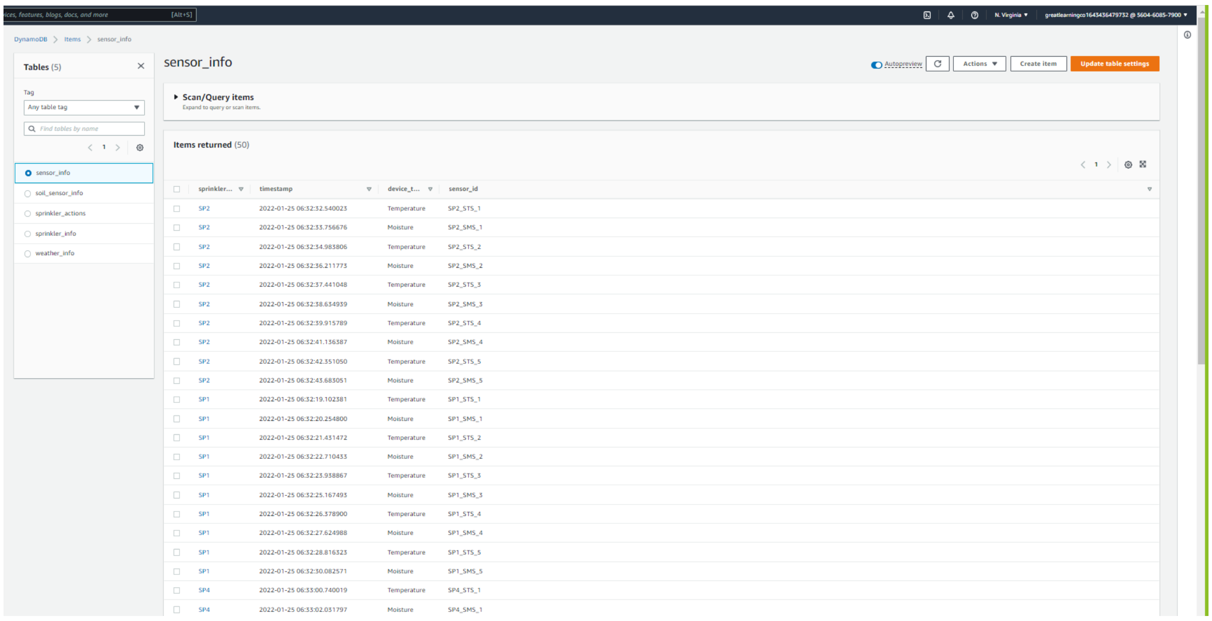


sprinkler_info

Items returned (5)

sprinkler_id	timestamp	device_t...	lat	long	min_devices_to_alarm
SP2	2022-01-29 08:44:24.028573	Sprinkler	-122.08	37.39	2
SP1	2022-01-29 08:44:23.450471	Sprinkler	-122.08	37.39	2
SP4	2022-01-29 08:44:25.043384	Sprinkler	-122.08	37.39	3
SP3	2022-01-29 08:44:24.563996	Sprinkler	-122.08	37.39	2
SP5	2022-01-29 08:44:25.551103	Sprinkler	-122.08	37.39	3

5. sprinkler_info table with data populated by DeviceOnboarding module

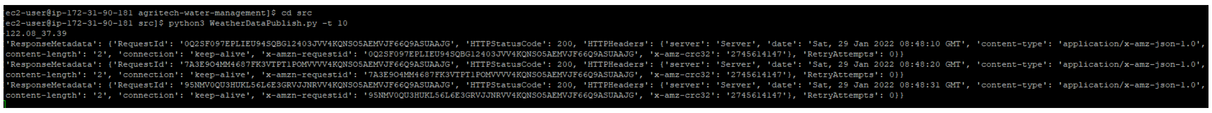


sensor_info

Items returned (50)

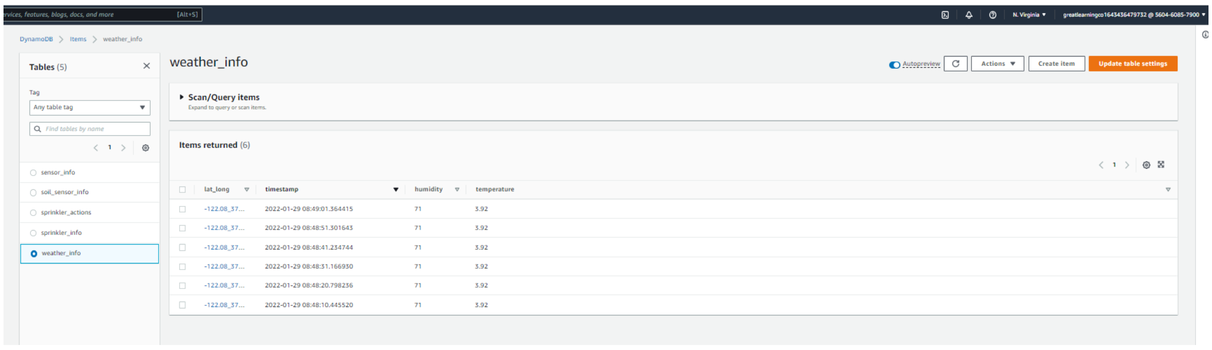
sprinkler...	timestamp	device_t...	sensor_id
SP2	2022-01-25 06:32:32.540023	Temperature	SP2_STS_1
SP2	2022-01-25 06:32:33.756676	Moisture	SP2_SMS_1
SP2	2022-01-25 06:32:34.983806	Temperature	SP2_STS_2
SP2	2022-01-25 06:32:36.211773	Moisture	SP2_SMS_2
SP2	2022-01-25 06:32:37.441048	Temperature	SP2_STS_3
SP2	2022-01-25 06:32:38.634939	Moisture	SP2_SMS_3
SP2	2022-01-25 06:32:39.915789	Temperature	SP2_STS_4
SP2	2022-01-25 06:32:41.136387	Moisture	SP2_SMS_4
SP2	2022-01-25 06:32:42.351050	Temperature	SP2_STS_5
SP2	2022-01-25 06:32:43.683051	Moisture	SP2_SMS_5
SP1	2022-01-25 06:32:19.102381	Temperature	SP1_STS_1
SP1	2022-01-25 06:32:20.254800	Moisture	SP1_SMS_1
SP1	2022-01-25 06:32:21.431472	Temperature	SP1_STS_2
SP1	2022-01-25 06:32:22.710433	Moisture	SP1_SMS_2
SP1	2022-01-25 06:32:23.938867	Temperature	SP1_STS_3
SP1	2022-01-25 06:32:25.167493	Moisture	SP1_SMS_3
SP1	2022-01-25 06:32:26.378900	Temperature	SP1_STS_4
SP1	2022-01-25 06:32:27.624988	Moisture	SP1_SMS_4
SP1	2022-01-25 06:32:28.816323	Temperature	SP1_STS_5
SP1	2022-01-25 06:32:30.082571	Moisture	SP1_SMS_5
SP4	2022-01-25 06:33:00.740019	Temperature	SP4_STS_1
SP4	2022-01-25 06:33:02.031797	Moisture	SP4_SMS_1

6. sensor_info table with data populated by DeviceOnboarding module



```
[ec2-user@ip-172-31-90-181 agritech-water-management]$ cd src
[ec2-user@ip-172-31-90-181 src]$ python3 WeatherDataPublish.py -t 10
-122.08_37.39
{'ResponseMetadata': {'RequestId': '0Q2SF097EPLIEU94SQBG12403JVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'server': 'Server', 'date': 'Sat, 29 Jan 2022 08:48:10 GMT', 'content-type': 'application/x-amz-json-1.0',
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '0Q2SF097EPLIEU94SQBG12403JVV4KQNSO5AEMVJF66Q9ASUAAJG', 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
{'ResponseMetadata': {'RequestId': '7A3E9O4MM4687FK3VTPT1POMVVVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'server': 'Server', 'date': 'Sat, 29 Jan 2022 08:48:20 GMT', 'content-type': 'application/x-amz-json-1.0',
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '7A3E9O4MM4687FK3VTPT1POMVVVV4KQNSO5AEMVJF66Q9ASUAAJG', 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
{'ResponseMetadata': {'RequestId': '95NMV0QU3HUKL56L6E3GRVJJNRVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'server': 'Server', 'date': 'Sat, 29 Jan 2022 08:48:31 GMT', 'content-type': 'application/x-amz-json-1.0',
'content-length': '2', 'connection': 'keep-alive', 'x-amzn-requestid': '95NMV0QU3HUKL56L6E3GRVJJNRVV4KQNSO5AEMVJF66Q9ASUAAJG', 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

7. WeatherDataPopulator module populates data to dynamodb

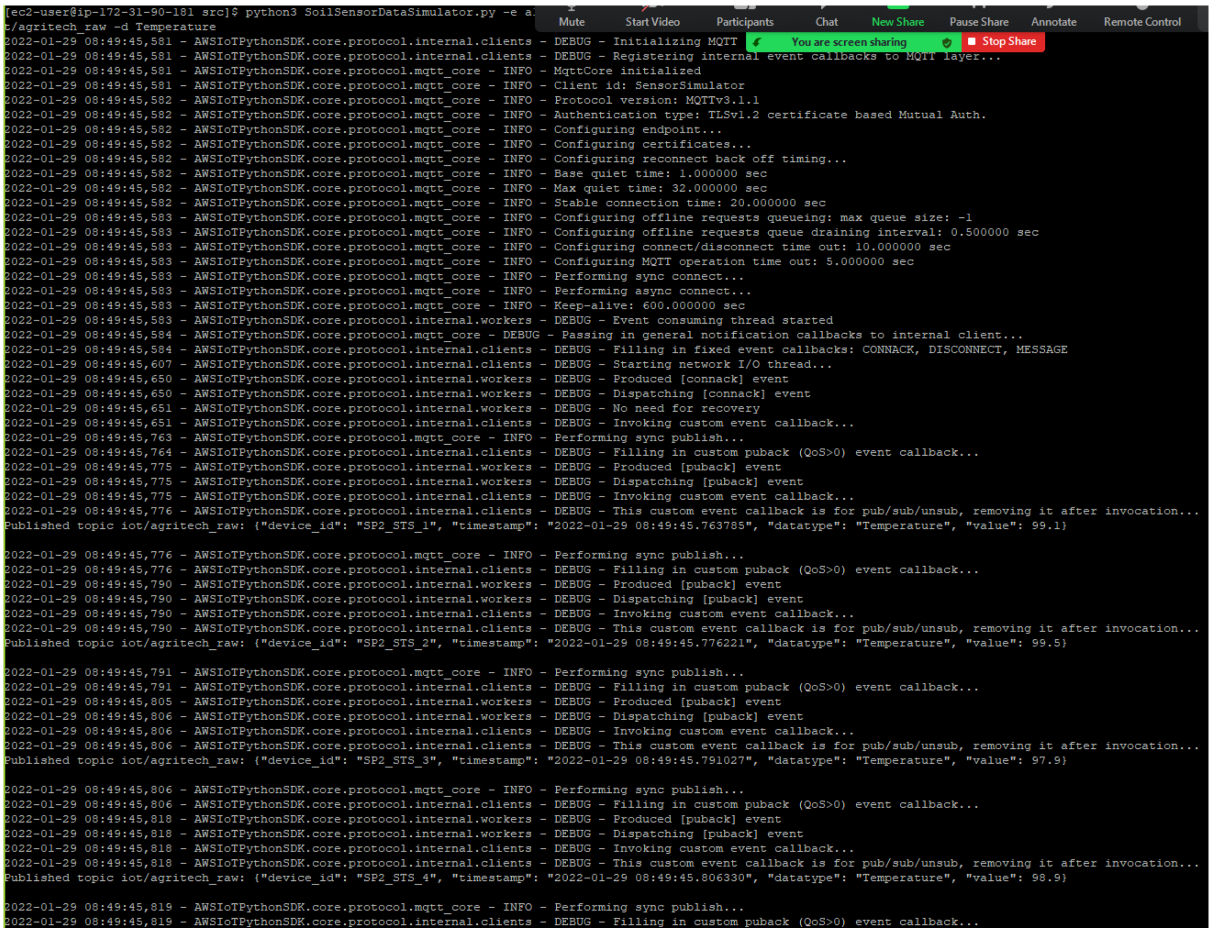


weather_info

Items returned (6)

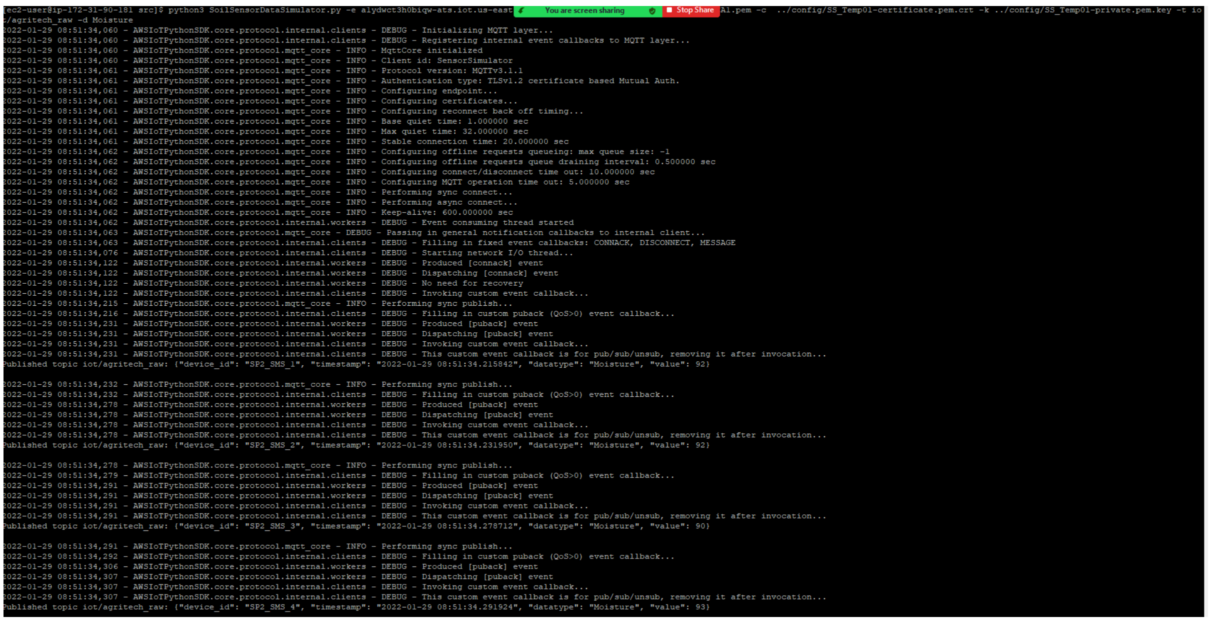
lat_long	timestamp	humidity	temperature
-122.08_37...	2022-01-29 08:49:01.364415	71	3.92
-122.08_37...	2022-01-29 08:48:51.301643	71	3.92
-122.08_37...	2022-01-29 08:48:41.234744	71	3.92
-122.08_37...	2022-01-29 08:48:31.166930	71	3.92
-122.08_37...	2022-01-29 08:48:20.798236	71	3.92
-122.08_37...	2022-01-29 08:48:10.445520	71	3.92

8. weather_info table with data populated by WeatherDataPopulator module



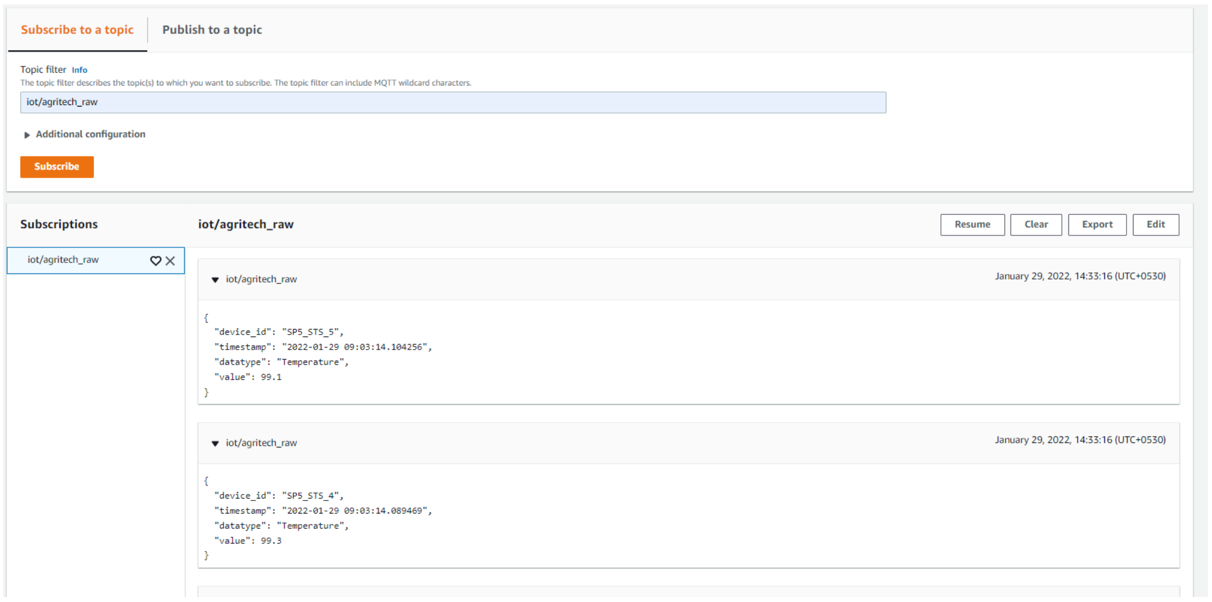
```
[ec2-user@ip-172-31-90-181 src]$ python3 SoilSensorDataSimulator.py -e a
t/agritech_raw -d Temperature
2022-01-29 08:49:45,581 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Initializing MQTT
2022-01-29 08:49:45,581 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Registering internal event callbacks to MQTT layer...
2022-01-29 08:49:45,581 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - MqttCore initialized
2022-01-29 08:49:45,581 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Client id: SensorSimulator
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Protocol version: MQTTv3.1.1
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Authentication type: TLSv1.2 certificate based Mutual Auth.
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring endpoint...
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring certificates...
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring reconnect back off timing...
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Base quiet time: 1.000000 sec
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Max quiet time: 32.000000 sec
2022-01-29 08:49:45,582 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Stable connection time: 20.000000 sec
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring offline requests queueing: max queue size: -1
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring offline requests queue draining interval: 0.500000 sec
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring connect/disconnect time out: 10.000000 sec
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring MQTT operation time out: 5.000000 sec
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync connect...
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing async connect...
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Keep-alive: 600.000000 sec
2022-01-29 08:49:45,583 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Event consuming thread started
2022-01-29 08:49:45,584 - AWSIoTPythonSDK.core.protocol.mqtt_core - DEBUG - Passing in general notification callbacks to internal client...
2022-01-29 08:49:45,584 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in fixed event callbacks: CONNACK, DISCONNECT, MESSAGE
2022-01-29 08:49:45,607 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Starting network I/O thread...
2022-01-29 08:49:45,650 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [connack] event
2022-01-29 08:49:45,650 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [connack] event
2022-01-29 08:49:45,651 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - No need for recovery
2022-01-29 08:49:45,651 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:49:45,763 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:49:45,764 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:49:45,775 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:49:45,775 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:49:45,775 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:49:45,776 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_STS_1", "timestamp": "2022-01-29 08:49:45.763785", "datatype": "Temperature", "value": 99.1}

2022-01-29 08:49:45,776 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:49:45,776 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:49:45,790 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:49:45,790 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:49:45,790 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:49:45,790 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_STS_2", "timestamp": "2022-01-29 08:49:45.776221", "datatype": "Temperature", "value": 99.5}

2022-01-29 08:49:45,791 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:49:45,791 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:49:45,805 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:49:45,806 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:49:45,806 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:49:45,806 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_STS_3", "timestamp": "2022-01-29 08:49:45.791027", "datatype": "Temperature", "value": 97.9}

2022-01-29 08:49:45,806 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:49:45,806 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:49:45,818 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:49:45,818 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:49:45,818 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:49:45,818 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_STS_4", "timestamp": "2022-01-29 08:49:45.806330", "datatype": "Temperature", "value": 98.9}

2022-01-29 08:49:45,819 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:49:45,819 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
```

9. SoilSensorDataSimulator module sends simulate Temperature data to IOT Core



```
[ec2-user@ip-172-31-90-181 src]$ python3 SoilSensorDataSimulator.py -e alydwct3h0biqw-ats.iot.us-east A1.pem -c  ../config/SS_Temp01-certificate.pem.crt -k ../config/SS_Temp01-private.pem.key -t io
t/agritech_raw -d Moisture
2022-01-29 08:51:34,060 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Initializing MQTT layer...
2022-01-29 08:51:34,060 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Registering internal event callbacks to MQTT layer...
2022-01-29 08:51:34,060 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - MqttCore initialized
2022-01-29 08:51:34,060 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Client id: SensorSimulator
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Protocol version: MQTTv3.1.1
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Authentication type: TLSv1.2 certificate based Mutual Auth.
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring endpoint...
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring certificates...
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring reconnect back off timing...
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Base quiet time: 1.000000 sec
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Max quiet time: 32.000000 sec
2022-01-29 08:51:34,061 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Stable connection time: 20.000000 sec
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring offline requests queueing: max queue size: -1
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring offline requests queue draining interval: 0.500000 sec
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring connect/disconnect time out: 10.000000 sec
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Configuring MQTT operation time out: 5.000000 sec
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync connect...
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing async connect...
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Keep-alive: 600.000000 sec
2022-01-29 08:51:34,062 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Event consuming thread started
2022-01-29 08:51:34,063 - AWSIoTPythonSDK.core.protocol.mqtt_core - DEBUG - Passing in general notification callbacks to internal client...
2022-01-29 08:51:34,063 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in fixed event callbacks: CONNACK, DISCONNECT, MESSAGE
2022-01-29 08:51:34,076 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Starting network I/O thread...
2022-01-29 08:51:34,122 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [connack] event
2022-01-29 08:51:34,122 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [connack] event
2022-01-29 08:51:34,122 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - No need for recovery
2022-01-29 08:51:34,122 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:51:34,215 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:51:34,216 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:51:34,231 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:51:34,231 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:51:34,231 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:51:34,231 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_SMS_1", "timestamp": "2022-01-29 08:51:34.215842", "datatype": "Moisture", "value": 92}

2022-01-29 08:51:34,232 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:51:34,232 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:51:34,278 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:51:34,278 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:51:34,278 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:51:34,278 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_SMS_2", "timestamp": "2022-01-29 08:51:34.231950", "datatype": "Moisture", "value": 92}

2022-01-29 08:51:34,278 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:51:34,279 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:51:34,291 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:51:34,291 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:51:34,291 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:51:34,291 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_SMS_3", "timestamp": "2022-01-29 08:51:34.278712", "datatype": "Moisture", "value": 90}

2022-01-29 08:51:34,291 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2022-01-29 08:51:34,292 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Filling in custom puback (QoS>0) event callback...
2022-01-29 08:51:34,306 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
2022-01-29 08:51:34,307 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2022-01-29 08:51:34,307 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
2022-01-29 08:51:34,307 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - This custom event callback is for pub/sub/unsub, removing it after invocation...
Published topic iot/agritech_raw: {"device_id": "SP2_SMS_4", "timestamp": "2022-01-29 08:51:34.291924", "datatype": "Moisture", "value": 93}
```

10. SoilSensorDataSimulator module sends simulate Moisture data to AWS IoT Core



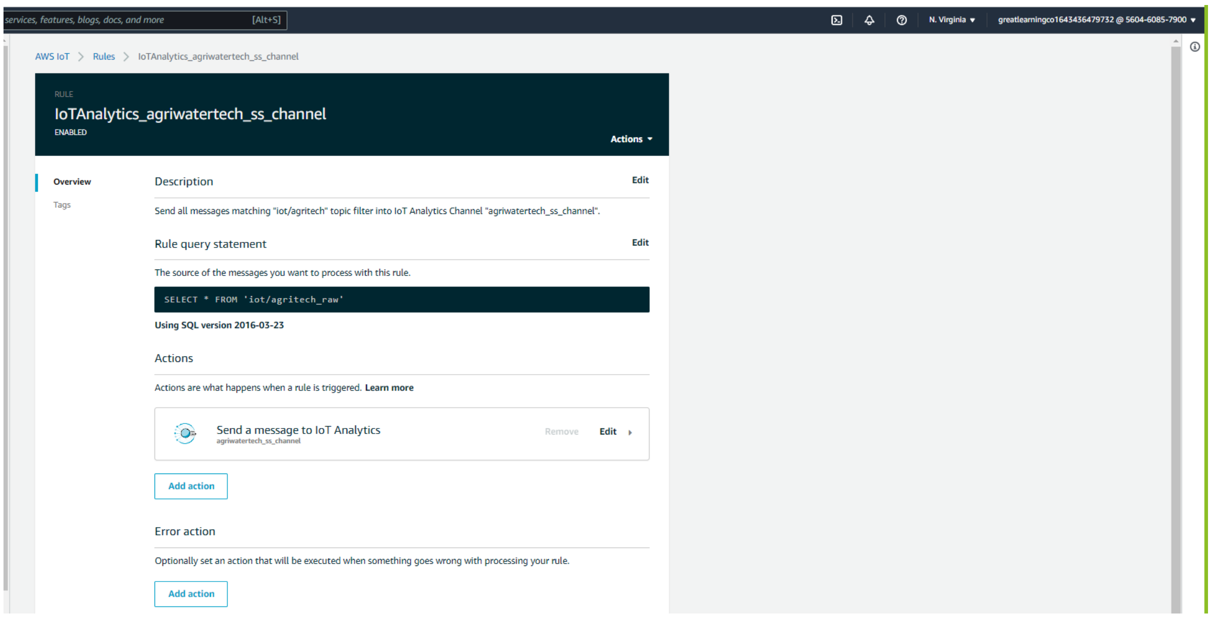
iot/agritech_raw January 29, 2022, 14:33:16 (UTC+0530)

```
{
  "device_id": "SPS_STS_5",
  "timestamp": "2022-01-29 09:03:14.104256",
  "datatype": "Temperature",
  "value": 99.1
}
```

iot/agritech_raw January 29, 2022, 14:33:16 (UTC+0530)

```
{
  "device_id": "SPS_STS_4",
  "timestamp": "2022-01-29 09:03:14.089469",
  "datatype": "Temperature",
  "value": 99.3
}
```

11. Data published are received in MQTT in AWS IoT Core



services, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
AWS IoT > Rules > IoTAnalytics_agriwatertech_ss_channel
RULE
Actions
Overview
Tags
Edit
Remove
Edit
Add action
Add action

IoTAnalytics_agriwatertech_ss_channel

ENABLED

Description

Send all messages matching "iot/agritech" topic filter into IoT Analytics Channel "agriwatertech_ss_channel".

Rule query statement

The source of the messages you want to process with this rule.

```
SELECT * FROM 'iot/agritech_raw'
```

Using SQL version 2016-03-23

Actions

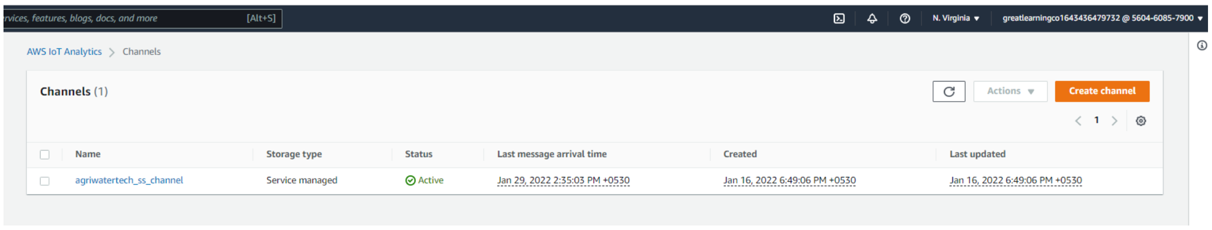
Actions are what happens when a rule is triggered. Learn more

Send a message to IoT Analytics
agriwatertech_ss_channel

Error action

Optionally set an action that will be executed when something goes wrong with processing your rule.

12. Rule to send data to IoT Analytics channel

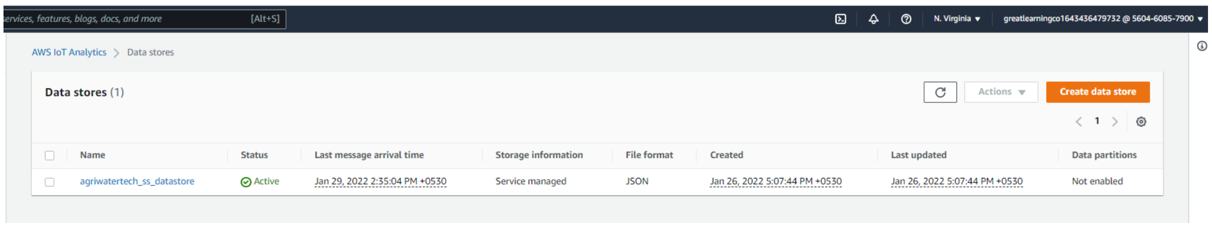


rvices, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
AWS IoT Analytics > Channels
Actions
Create channel
1

Channels (1)

Name	Storage type	Status	Last message arrival time	Created	Last updated
agriwatertech_ss_channel	Service managed	✓ Active	Jan 29, 2022 2:35:03 PM +0530	Jan 16, 2022 6:49:06 PM +0530	Jan 16, 2022 6:49:06 PM +0530

13. IoT Analytics channel which received data

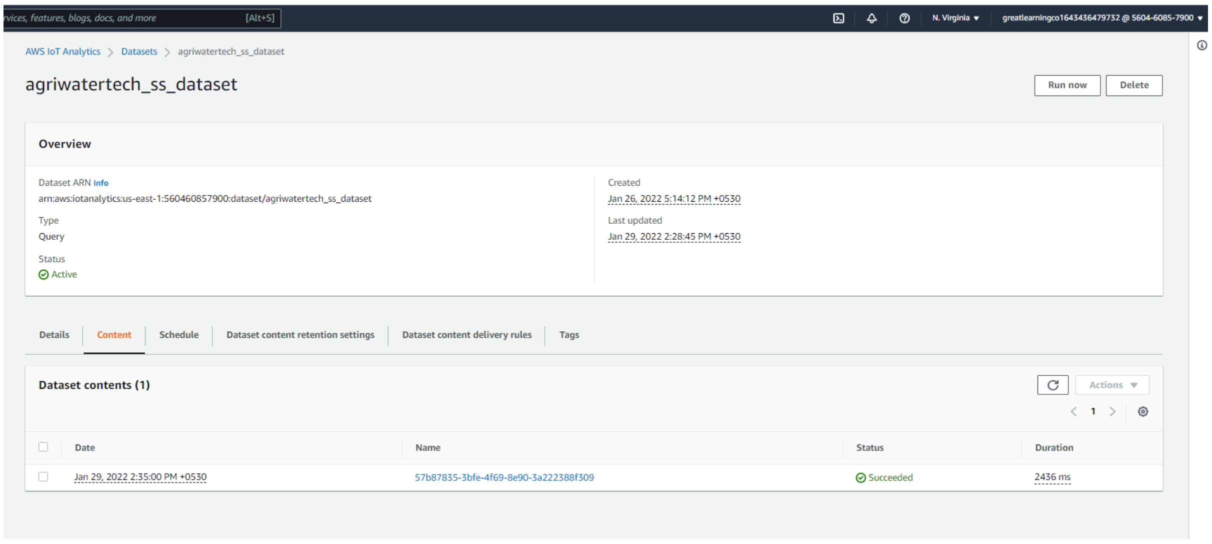

services, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
Actions
Create data store

AWS IoT Analytics > Data stores

Data stores (1)

Name	Status	Last message arrival time	Storage information	File format	Created	Last updated	Data partitions
agriwatertech_ss_datastore	Active	Jan 29, 2022 2:35:04 PM +0530	Service managed	JSON	Jan 26, 2022 5:07:44 PM +0530	Jan 26, 2022 5:07:44 PM +0530	Not enabled

14. IoT Analytics data store which received data


vices, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
Run now
Delete
Details
Content
Schedule
Dataset content retention settings
Dataset content delivery rules
Tags
Actions

AWS IoT Analytics > Datasets > agriwatertech_ss_dataset

agriwatertech_ss_dataset

Overview

Dataset ARN Info
arn:aws:iotanalytics:us-east-1:560460857900:dataset/agriwatertech_ss_dataset

Type
Query

Status
Active

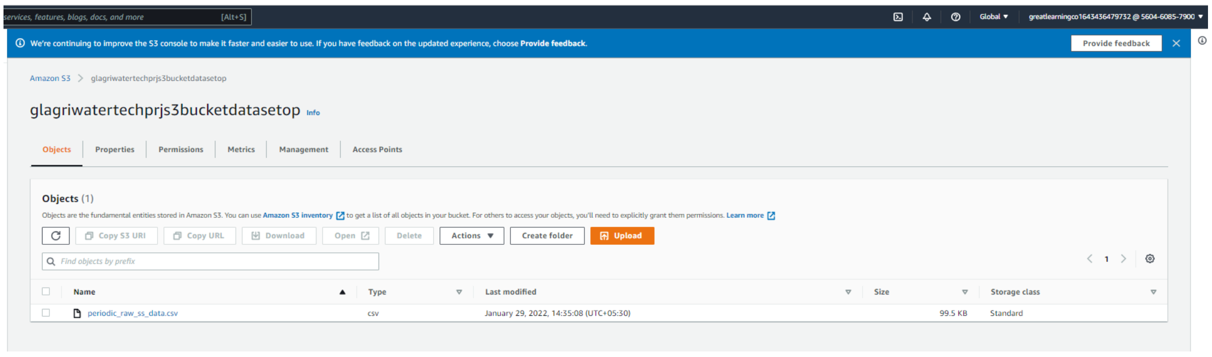
Created
Jan 26, 2022 5:14:12 PM +0530

Last updated
Jan 29, 2022 2:28:45 PM +0530

Dataset contents (1)

Date	Name	Status	Duration
Jan 29, 2022 2:35:00 PM +0530	57b87835-3bfe-4f69-8e90-3a222388f309	Succeeded	2436 ms

15. IoT Analytics dataset with 5 mins schedule



We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose **Provide feedback**.

Amazon S3 > glagriwatertechprjs3bucketdatasetop

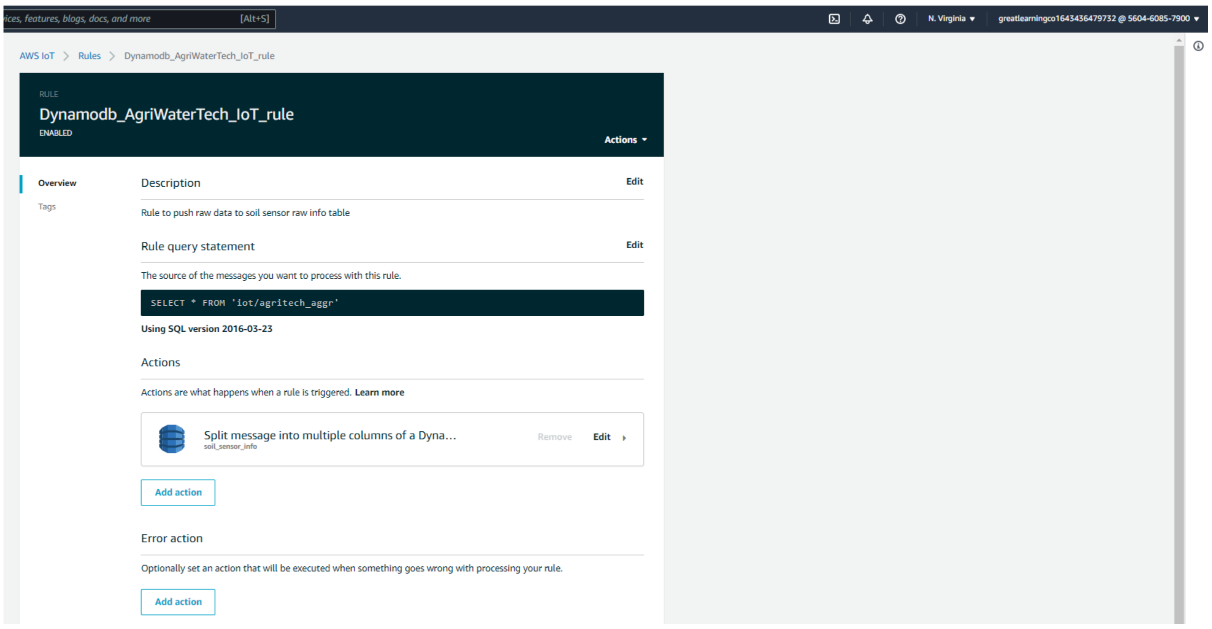
glagriwatertechprjs3bucketdatasetop Info

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
periodic_raw_ss_data.csv	csv	January 29, 2022, 14:35:08 (UTC+05:30)	99.5 KB	Standard

16. S3 bucket which holds the raw data in csv format



AWS IoT > Rules > Dynamodb_AgriWaterTech_IoT_rule

RULE

Dynamodb_AgriWaterTech_IoT_rule

ENABLED

Description

Rule to push raw data to soil sensor raw info table

Rule query statement

The source of the messages you want to process with this rule.

```
SELECT * FROM 'iot/agritech_aggr'
```

Using SQL version 2016-03-23

Actions

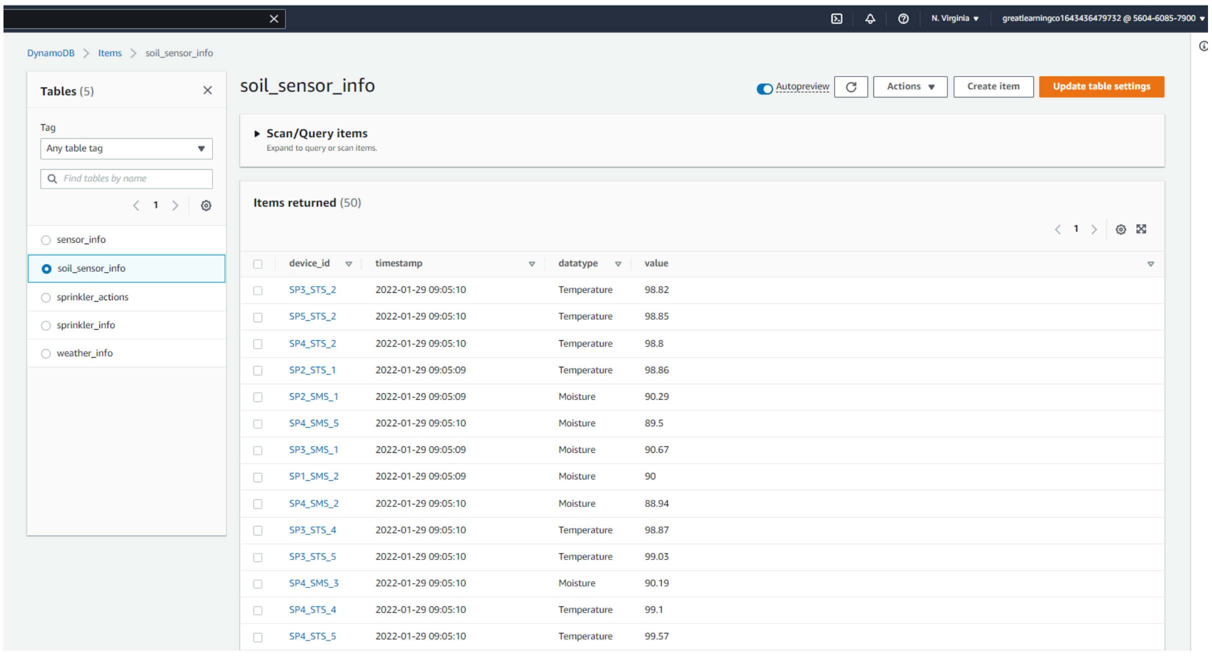
Actions are what happens when a rule is triggered. Learn more

Split message into multiple columns of a Dyna...
soil_sensor_info

Error action

Optionally set an action that will be executed when something goes wrong with processing your rule.

17. IoT Rule to aggregate data and send it to dynamoDB table soil_sensor_info



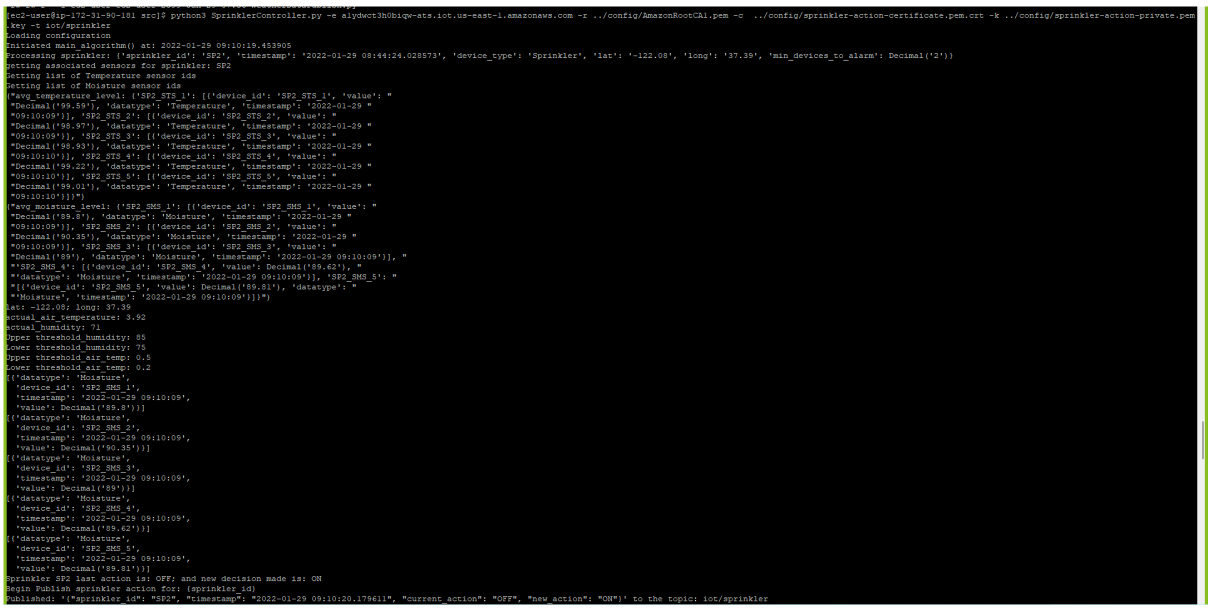
DynamoDB > Items > soil_sensor_info

soil_sensor_info

Items returned (50)

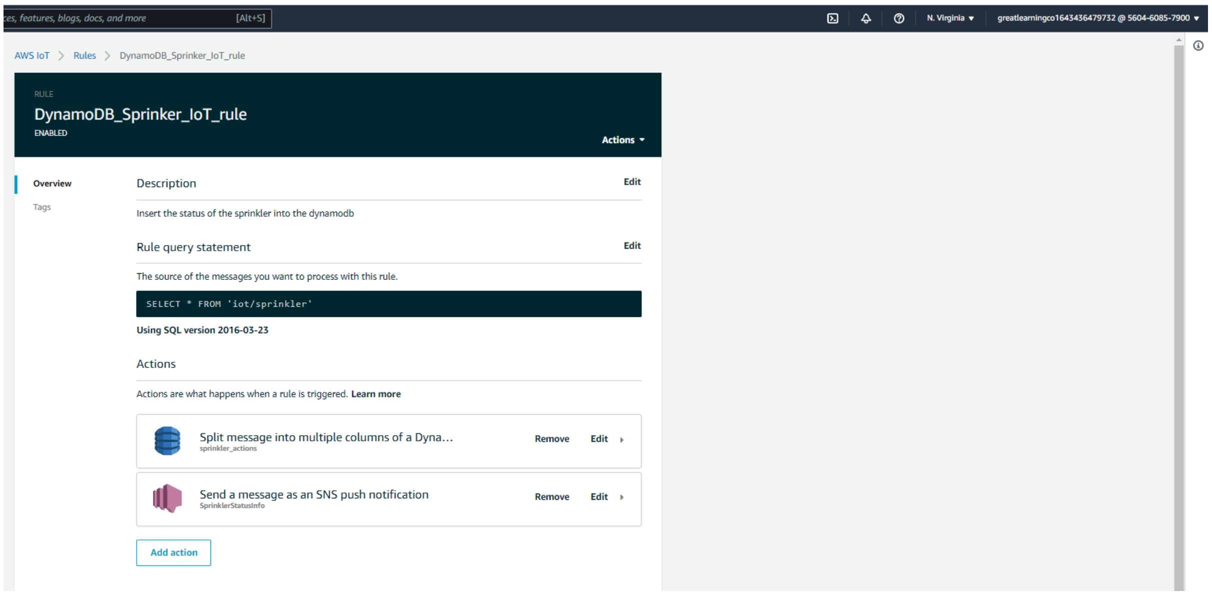
device_id	timestamp	datatype	value
SP3_STS_2	2022-01-29 09:05:10	Temperature	98.82
SP5_STS_2	2022-01-29 09:05:10	Temperature	98.85
SP4_STS_2	2022-01-29 09:05:10	Temperature	98.8
SP2_STS_1	2022-01-29 09:05:09	Temperature	98.86
SP2_SMS_1	2022-01-29 09:05:09	Moisture	90.29
SP4_SMS_5	2022-01-29 09:05:10	Moisture	89.5
SP3_SMS_1	2022-01-29 09:05:09	Moisture	90.67
SP1_SMS_2	2022-01-29 09:05:09	Moisture	90
SP4_SMS_2	2022-01-29 09:05:10	Moisture	88.94
SP3_STS_4	2022-01-29 09:05:10	Temperature	98.87
SP3_STS_5	2022-01-29 09:05:10	Temperature	99.03
SP4_SMS_3	2022-01-29 09:05:10	Moisture	90.19
SP4_STS_4	2022-01-29 09:05:10	Temperature	99.1
SP4_STS_5	2022-01-29 09:05:10	Temperature	99.57

18. soil_sensor_info table populated with every 5 mins aggregated data



```
[ec2-user@ip-172-31-90-181 src]$ python3 SprinklerController.py -e alydwct3h0biqw-ats.iot.us-east-1.amazonaws.com -r ../config/AmazonRootCA1.pem -c  ../config/sprinkler-action-certificate.pem.crt -k ../config/sprinkler-action-private.pem
.key -t iot/sprinkler
Loading configuration
Initiated main_algorithm() at: 2022-01-29 09:10:19.453905
Processing sprinkler: {'sprinkler_id': 'SP2', 'timestamp': '2022-01-29 08:44:24.028573', 'device_type': 'Sprinkler', 'lat': '-122.08', 'long': '37.39', 'min_devices_to_alarm': Decimal('2')}
getting associated sensors for sprinkler: SP2
Getting list of Temperature sensor ids
Getting list of Moisture sensor ids
("avg_temperature_level: {'SP2_STS_1': [{'device_id': 'SP2_STS_1', 'value': "
 "Decimal('99.59'), 'datatype': 'Temperature', 'timestamp': '2022-01-29 "
 "09:10:09'}], 'SP2_STS_2': [{'device_id': 'SP2_STS_2', 'value': "
 "Decimal('98.97'), 'datatype': 'Temperature', 'timestamp': '2022-01-29 "
 "09:10:09'}], 'SP2_STS_3': [{'device_id': 'SP2_STS_3', 'value': "
 "Decimal('98.93'), 'datatype': 'Temperature', 'timestamp': '2022-01-29 "
 "09:10:10'}], 'SP2_STS_4': [{'device_id': 'SP2_STS_4', 'value': "
 "Decimal('99.22'), 'datatype': 'Temperature', 'timestamp': '2022-01-29 "
 "09:10:10'}], 'SP2_STS_5': [{'device_id': 'SP2_STS_5', 'value': "
 "Decimal('99.01'), 'datatype': 'Temperature', 'timestamp': '2022-01-29 "
 "09:10:10'}]}")
("avg_moisture_level: {'SP2_SMS_1': [{'device_id': 'SP2_SMS_1', 'value': "
 "Decimal('89.8'), 'datatype': 'Moisture', 'timestamp': '2022-01-29 "
 "09:10:09'}], 'SP2_SMS_2': [{'device_id': 'SP2_SMS_2', 'value': "
 "Decimal('90.35'), 'datatype': 'Moisture', 'timestamp': '2022-01-29 "
 "09:10:09'}], 'SP2_SMS_3': [{'device_id': 'SP2_SMS_3', 'value': "
 "Decimal('89'), 'datatype': 'Moisture', 'timestamp': '2022-01-29 09:10:09'}], "
 "'SP2_SMS_4': [{'device_id': 'SP2_SMS_4', 'value': Decimal('89.62'), "
 "'datatype': 'Moisture', 'timestamp': '2022-01-29 09:10:09'}], 'SP2_SMS_5': "
 "[{'device_id': 'SP2_SMS_5', 'value': Decimal('89.81'), 'datatype': "
 "'Moisture', 'timestamp': '2022-01-29 09:10:09'}]}")
lat: -122.08; long: 37.39
actual_air_temperature: 3.92
actual_humidity: 71
Upper threshold_humidity: 85
Lower threshold_humidity: 75
Upper threshold_air_temp: 0.5
Lower threshold_air_temp: 0.2
[{'datatype': 'Moisture',
  'device_id': 'SP2_SMS_1',
  'timestamp': '2022-01-29 09:10:09',
  'value': Decimal('89.8')}]
[{'datatype': 'Moisture',
  'device_id': 'SP2_SMS_2',
  'timestamp': '2022-01-29 09:10:09',
  'value': Decimal('90.35')}]
[{'datatype': 'Moisture',
  'device_id': 'SP2_SMS_3',
  'timestamp': '2022-01-29 09:10:09',
  'value': Decimal('89')}]
[{'datatype': 'Moisture',
  'device_id': 'SP2_SMS_4',
  'timestamp': '2022-01-29 09:10:09',
  'value': Decimal('89.62')}]
[{'datatype': 'Moisture',
  'device_id': 'SP2_SMS_5',
  'timestamp': '2022-01-29 09:10:09',
  'value': Decimal('89.81')}]
Sprinkler SP2 last action is: OFF; and new decision made is: ON
Begin Publish sprinkler action for: {sprinkler_id}
Published: '{"sprinkler_id": "SP2", "timestamp": "2022-01-29 09:10:20.179611", "current_action": "OFF", "new_action": "ON"}' to the topic: iot/sprinkler
```

19. SprinklerController modules which runs on 5 mins schedule and makes decision from 5 mins aggregate data from soil_sensor_info and latest weather data from weather_info table and sends the decision to MQTT on IoT Core


es, features, blogs, docs, and more [Alt+S]
N. Virginia
greatlearningco1643436479732 @ 5604-6085-7900
AWS IoT > Rules > DynamoDB_Sprinker_IoT_rule
Actions
Overview
Tags
Edit
Remove
Edit
Add action

RULE
DynamoDB_Sprinker_IoT_rule
ENABLED

Description

Insert the status of the sprinkler into the dynamodb

Rule query statement

The source of the messages you want to process with this rule.

```
SELECT * FROM 'iot/sprinkler'
```

Using SQL version 2016-03-23

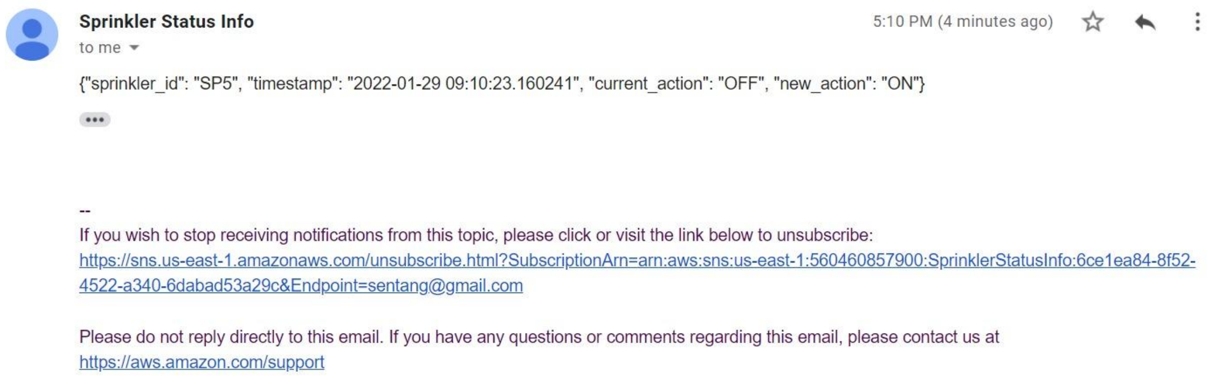
Actions

Actions are what happens when a rule is triggered. Learn more

Split message into multiple columns of a Dyna…
sprinkler_actions

Send a message as an SNS push notification
SprinklerStatusInfo

20. Rule which sends the decision to dynamoDB table and also SNS notification


Sprinkler Status Info
to me

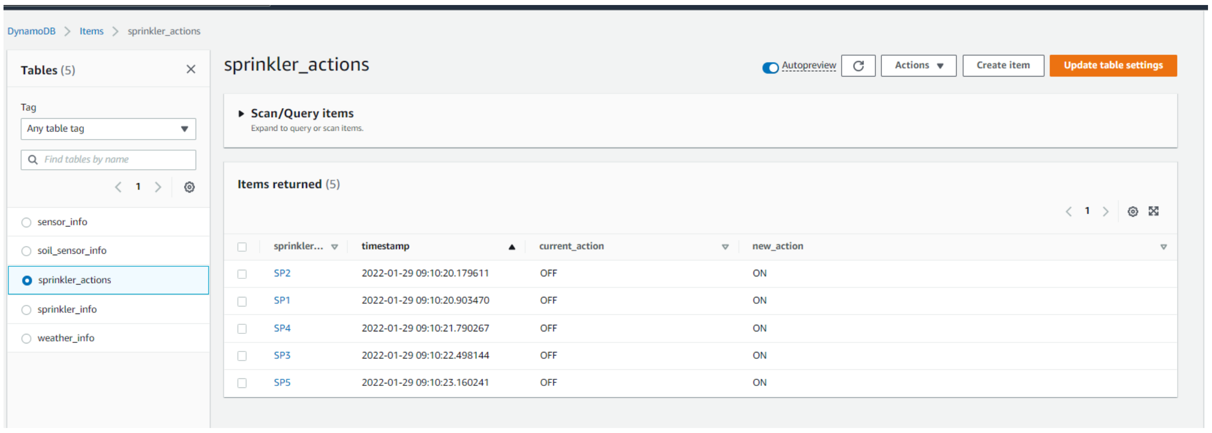
5:10 PM (4 minutes ago)

{"sprinkler_id": "SP5", "timestamp": "2022-01-29 09:10:23.160241", "current_action": "OFF", "new_action": "ON"}

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:560460857900:SprinklerStatusInfo:6ce1ea84-8f52-4522-a340-6dabad53a29c&Endpoint=sentang@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at
https://aws.amazon.com/support

21. The email received about the decision made for a particular sprinkler

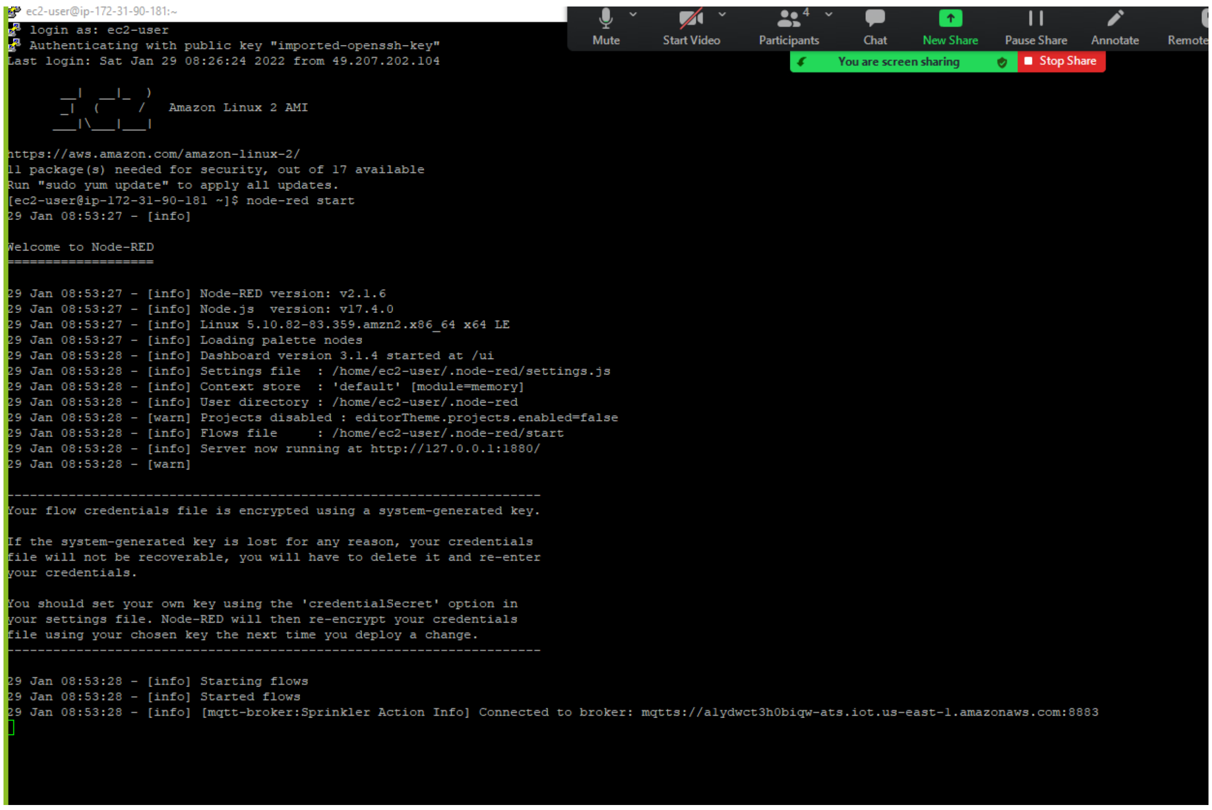


sprinkler_actions

Items returned (5)

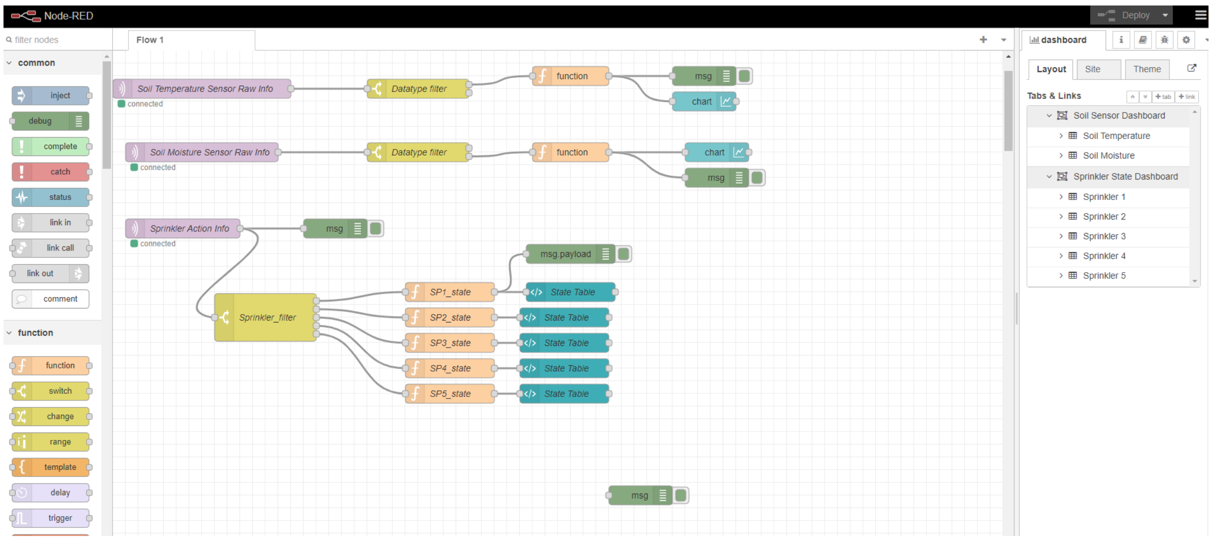
sprinkler...	timestamp	current_action	new_action
SP2	2022-01-29 09:10:20.179611	OFF	ON
SP1	2022-01-29 09:10:20.903470	OFF	ON
SP4	2022-01-29 09:10:21.790267	OFF	ON
SP3	2022-01-29 09:10:22.498144	OFF	ON
SP5	2022-01-29 09:10:23.160241	OFF	ON

22. The dynamoDB table which is populated with the decision made for each sprinkler on each iteration

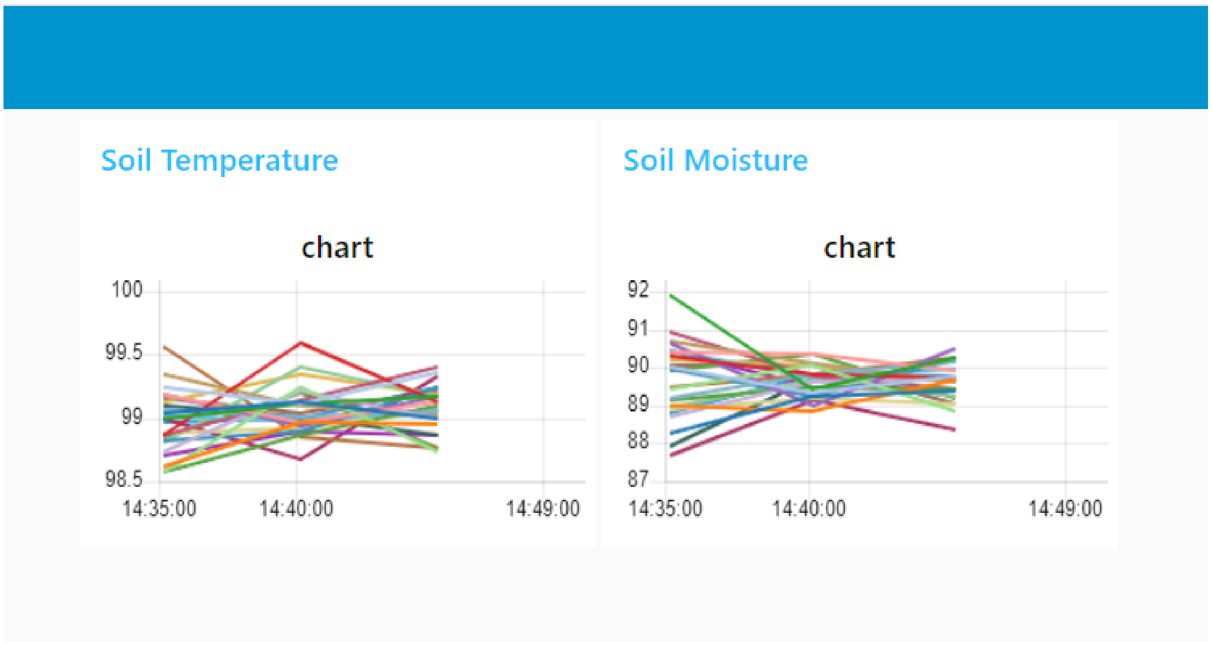


```
 login as: ec2-user
 Authenticating with public key "imported-openssh-key"
Last login: Sat Jan 29 08:26:24 2022 from 49.207.202.104

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
11 package(s) needed for security, out of 17 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-90-181 ~]$ node-red start
29 Jan 08:53:27 - [info]

Welcome to Node-RED
===================

29 Jan 08:53:27 - [info] Node-RED version: v2.1.6
29 Jan 08:53:27 - [info] Node.js  version: v17.4.0
29 Jan 08:53:27 - [info] Linux 5.10.82-83.359.amzn2.x86_64 x64 LE
29 Jan 08:53:27 - [info] Loading palette nodes
29 Jan 08:53:28 - [info] Dashboard version 3.1.4 started at /ui
29 Jan 08:53:28 - [info] Settings file  : /home/ec2-user/.node-red/settings.js
29 Jan 08:53:28 - [info] Context store  : 'default' [module=memory]
29 Jan 08:53:28 - [info] User directory : /home/ec2-user/.node-red
29 Jan 08:53:28 - [warn] Projects disabled : editorTheme.projects.enabled=false
29 Jan 08:53:28 - [info] Flows file     : /home/ec2-user/.node-red/start
29 Jan 08:53:28 - [info] Server now running at http://127.0.0.1:1880/
29 Jan 08:53:28 - [warn]

---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

29 Jan 08:53:28 - [info] Starting flows
29 Jan 08:53:28 - [info] Started flows
29 Jan 08:53:28 - [info] [mqtt-broker:Sprinkler Action Info] Connected to broker: mqtts://alydwct3h0biqw-ats.iot.us-east-1.amazonaws.com:8883
```

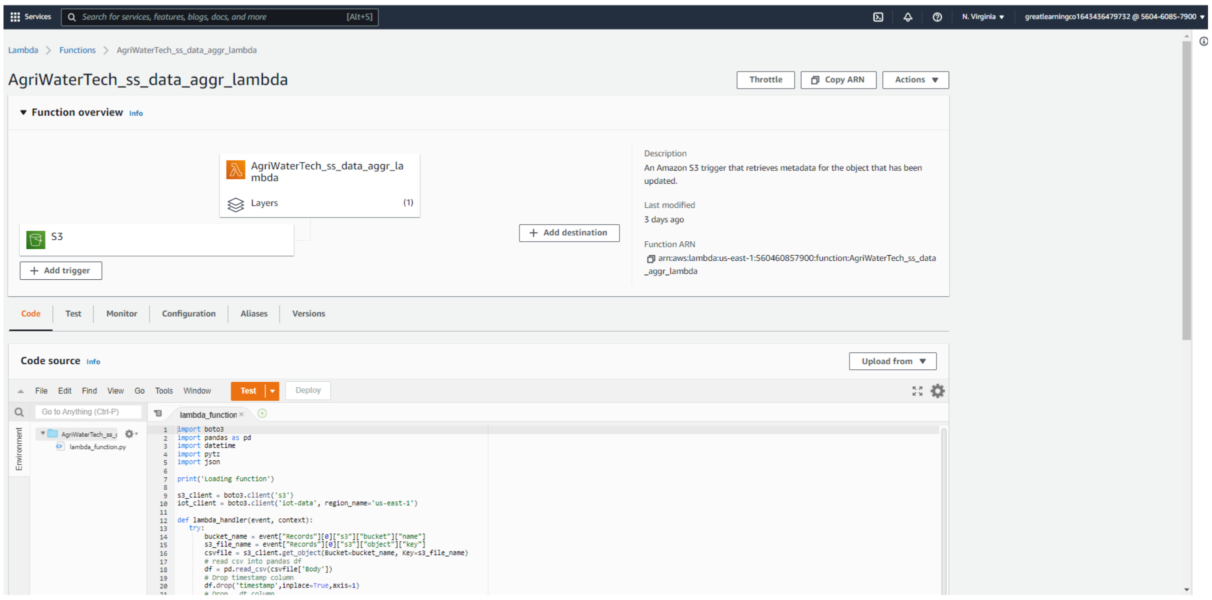
23. Starting NodeRed for analytical data



24. Node Red schema design



25. The graph generated out of the aggregated data (every 5 mins)



AgriWaterTech_ss_data_aggr_lambda

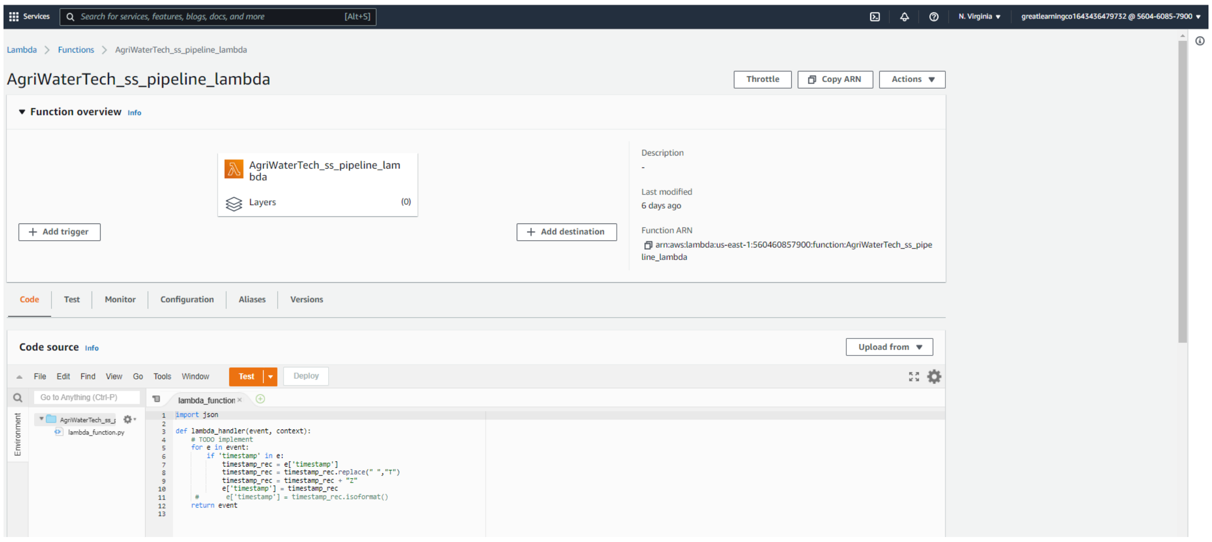
Description
An Amazon S3 trigger that retrieves metadata for the object that has been updated.

Last modified
3 days ago

Function ARN
arn:aws:lambda:us-east-1:560460857900:function:AgriWaterTech_ss_data_aggr_lambda

```
import boto3
import pandas as pd
import datetime
import pytz
import json

print('Loading function')

s3_client = boto3.client('s3')
iot_client = boto3.client('iot-data', region_name='us-east-1')

def lambda_handler(event, context):
    try:
        bucket_name = event["Records"][0]["s3"]["bucket"]["name"]
        s3_file_name = event["Records"][0]["s3"]["object"]["key"]
        csvfile = s3_client.get_object(Bucket=bucket_name, Key=s3_file_name)
        # read csv into pandas df
        df = pd.read_csv(csvfile['Body'])
        # Drop timestamp column
        df.drop('timestamp',inplace=True,axis=1)
```

26. The lambda which has the logic to aggregate data for every 5 mins and sends the average data for every sensor info



AgriWaterTech_ss_pipeline_lambda

Description
-

Last modified
6 days ago

Function ARN
arn:aws:lambda:us-east-1:560460857900:function:AgriWaterTech_ss_pipeline_lambda

```python
import json

def lambda_handler(event, context):
    # TODO implement
    for e in event:
        if 'timestamp' in e:
            timestamp_rec = e['timestamp']
            timestamp_rec = timestamp_rec.replace(" ","T")
            timestamp_rec = timestamp_rec + "Z"
            e['timestamp'] = timestamp_rec
     #       e['timestamp'] = timestamp_rec.isoformat()
    return event
```

27. Lambda which converts timestamp to UTC