CIS 600
Internet Of Things : Application Development
Spring 2025

Name : Venkata Sri Siva Ramakrishna Palaparthy
SUID : 433976193
NetID : vpalapar
Date : 04 - April - 2025

In Class Exercise - Week 10

Brief Explanation of the Steps Used in Developing the IoT System :

1. **Creation of an IoT Thing in AWS IoT Core :**
   ➔ Started with creating a new IoT thing in the AWS IoT Core console, which automatically generated a unique certificate and associated security credentials including a private key, public key, and root CA files.

2. **Downloading the Required Credentials :**
   ➔ Security credentials such as AmazonRootCA1, the certificate, and the private key were downloaded and stored for use in the simulation script.

3. **Python MQTT Script Configuration :**
   ➔ Wrote a python script to simulate environmental sensor data such as temperature, humidity, and CO2 levels. The script includes the endpoint taken from AWS and local paths to the downloaded certificates and key files.

4. **Testing MQTT Communication :**
   ➔ The AWS IoT MQTT Test Client was used to verify that the Python script was correctly publishing data to the topic update/environment/dht1. Real-time data flow was confirmed by observing the incoming messages in AWS IoT MQTT Test Client.

5. **Setting Up an S3 Bucket :**
   ➔ An Amazon S3 bucket (my-s3-bucket-iot) was created to serve as the final storage location for the incoming sensor data.

6. **Creating a Kinesis Firehose Stream :**
   ➔ A Firehose delivery stream was configured to accept data via the Direct PUT method and deliver it to the S3 bucket. Basic settings were selected, with no data transformation or compression enabled.

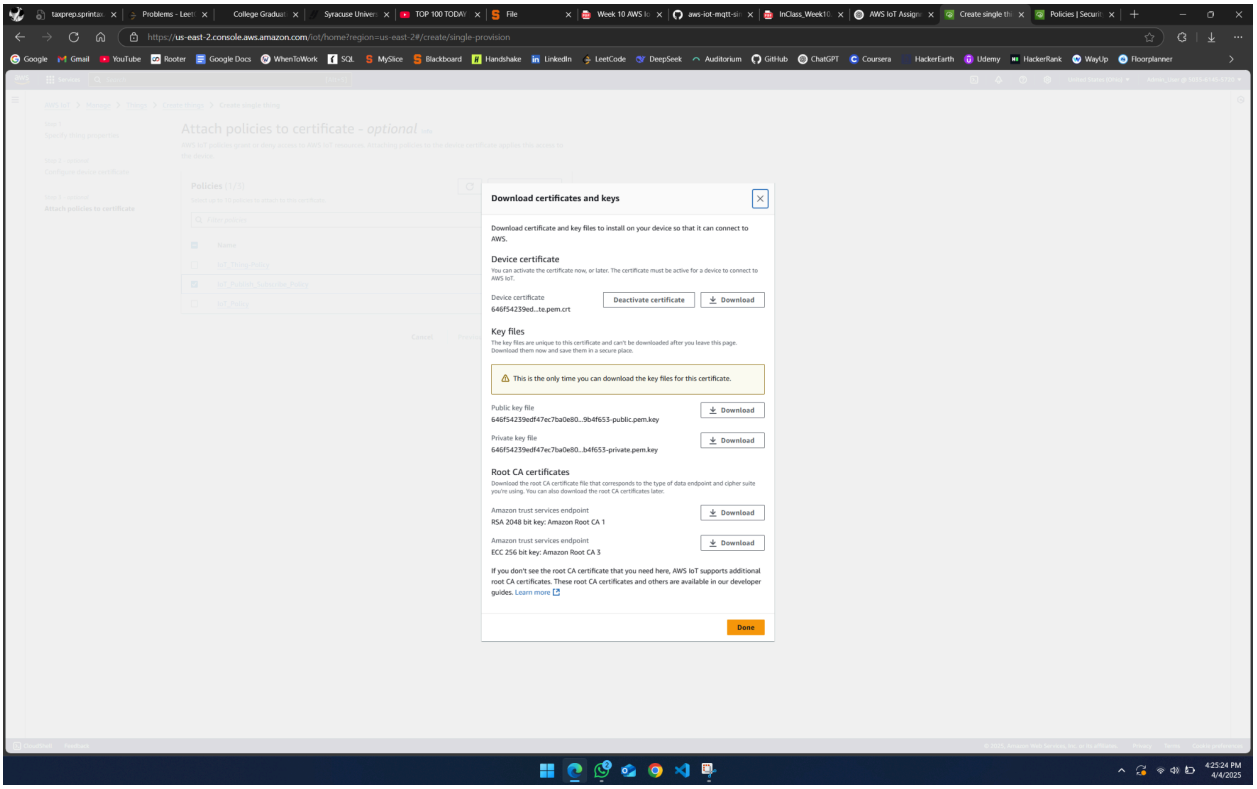
7. **Creating an IoT Rule :**
   ➔ An AWS IoT rule was configured to forward messages from the MQTT topic update/environment/dht1 to the Firehose delivery stream. An appropriate IAM role was created to allow this connection.

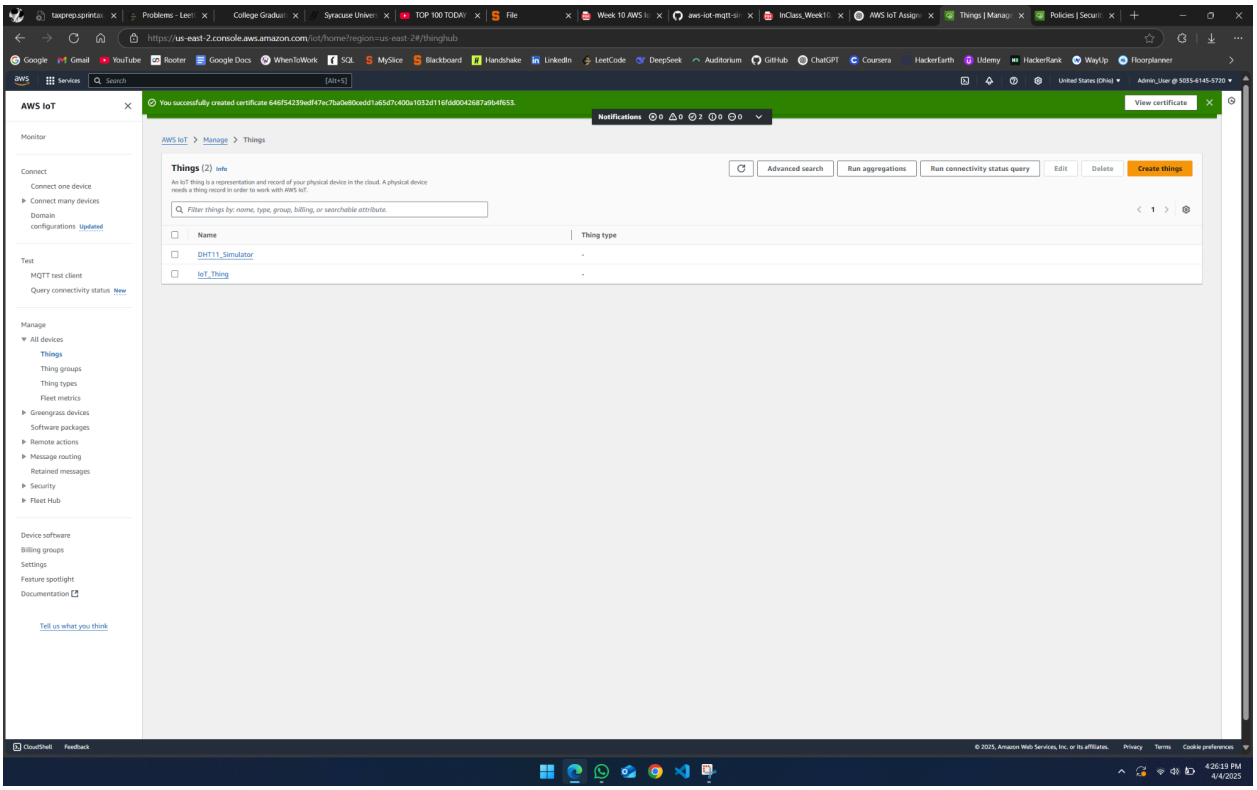
8. **Confirming Data Flow and Storage :**
   ➔ The Python script was run again to push data. Monitoring tools in AWS Firehose confirmed successful delivery, and the presence of JSON files in the S3 bucket validated that data storage was functioning as expected.

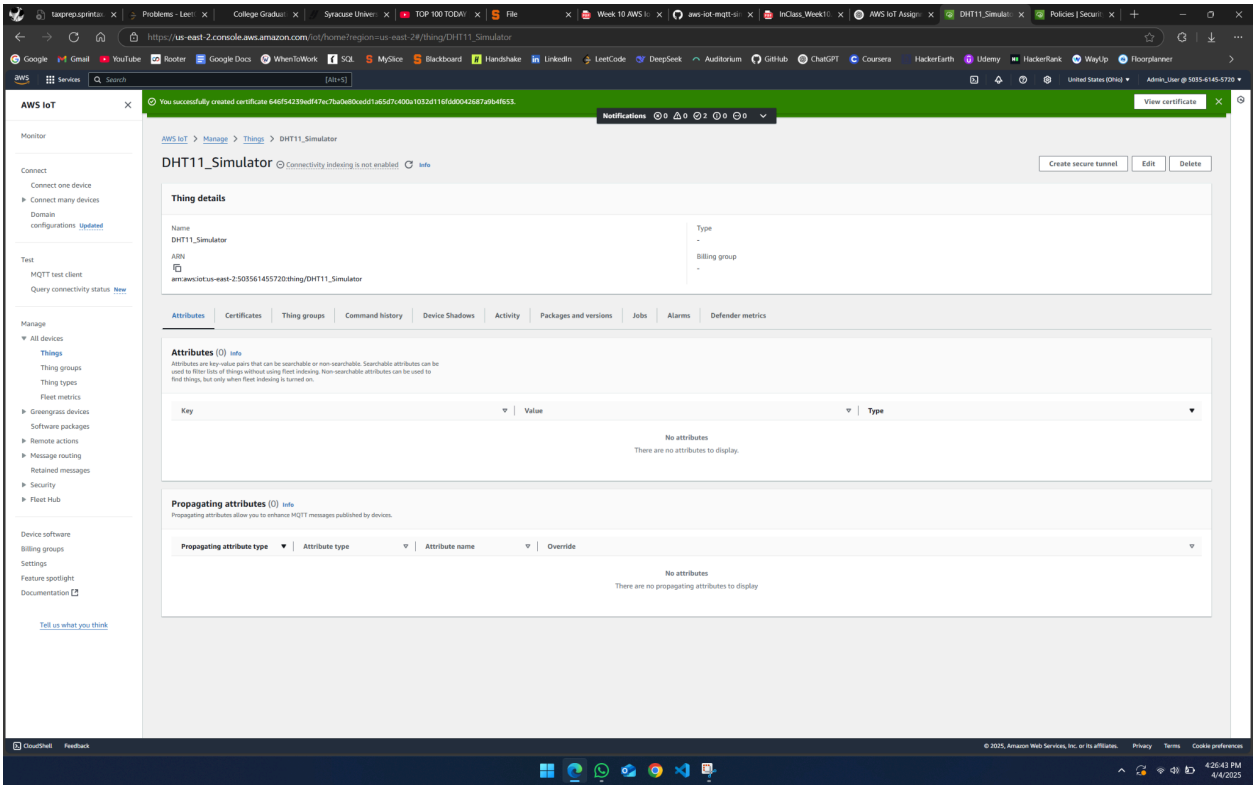
9. **Verifying Output :**
   ➔ Files downloaded from the S3 bucket contained the expected JSON-formatted sensor data, confirming that the entire IoT data pipeline was functioning correctly.

Screenshots :

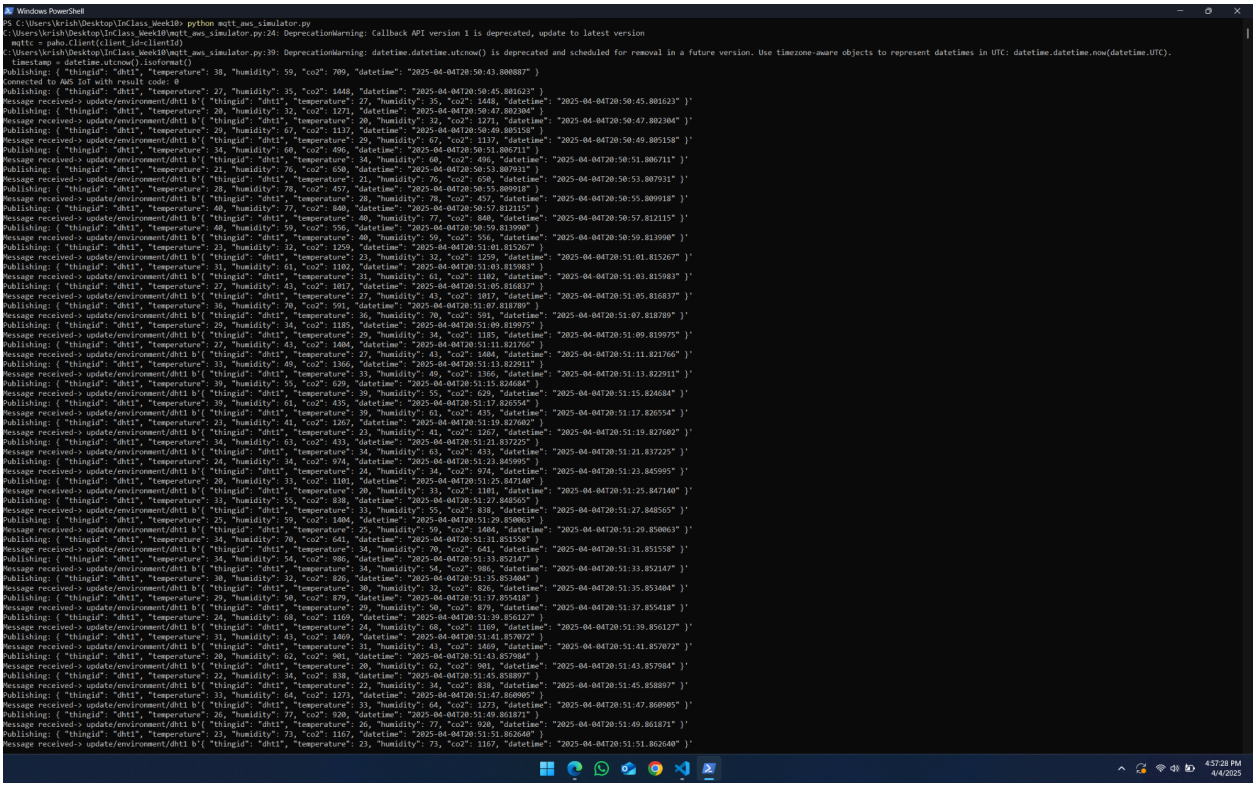
Creating a Thing in AWS IoT Core and Setting Up AWS IoT Certificates and Policies :



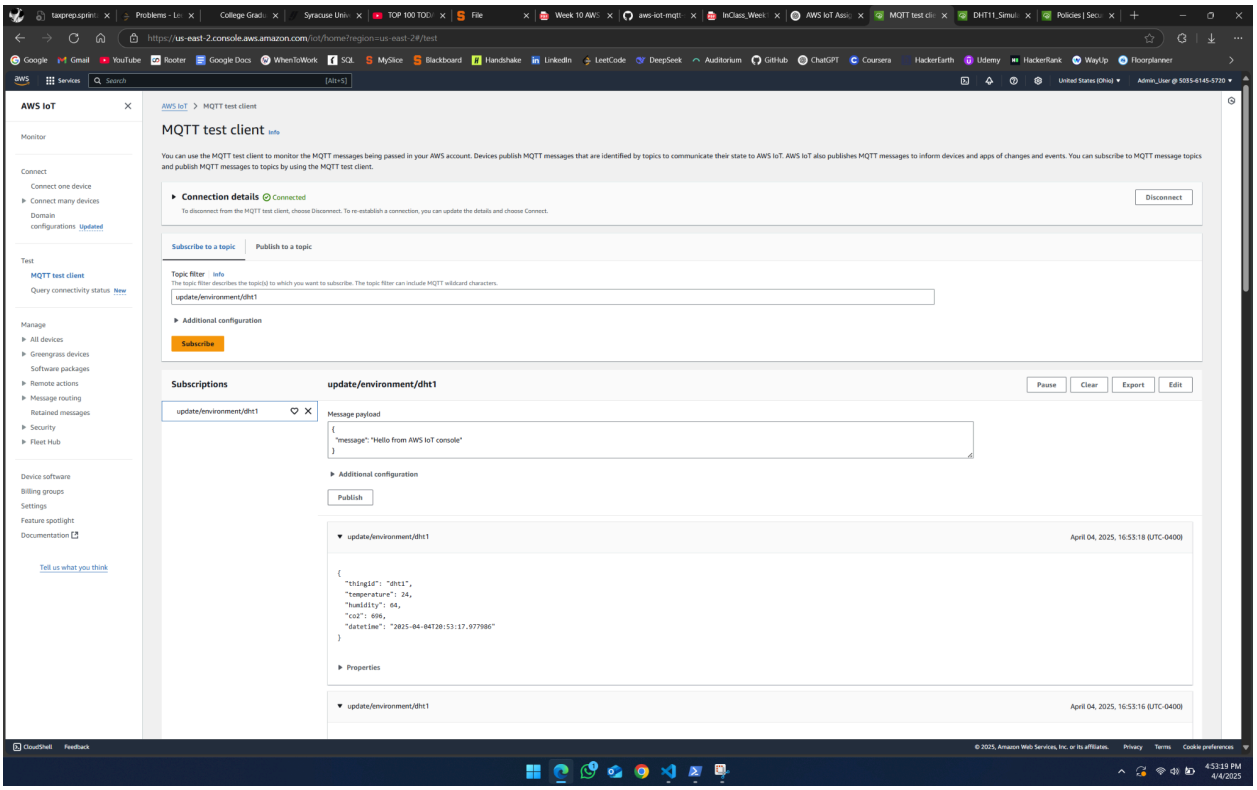

AWS IoT
You successfully created certificate 646f54239edf47ec7ba0e80ced1a65d7c400a1032d116fd80042687a9b4f653.
View certificate
Notifications
AWS IoT > Manage > Things
Things (2) Info
An IoT thing is a representation and record of your physical device in the cloud. A physical device needs a thing record in order to work with AWS IoT.
Advanced search
Run aggregations
Run connectivity status query
Edit
Delete
Create things
Filter things by: name, type, group, billing, or searchable attribute.
Name
Thing type
DHT11_Simulator
IoT_Thing
Monitor
Connect
Connect one device
Connect many devices
Domain configurations Updated
Test
MQTT test client
Query connectivity status New
Manage
All devices
Things
Thing groups
Thing types
Fleet metrics
Greengrass devices
Software packages
Remote actions
Message routing
Retained messages
Security
Fleet Hub
Device software
Billing groups
Settings
Feature spotlight
Documentation
Tell us what you think
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
4:26:19 PM
4/4/2025

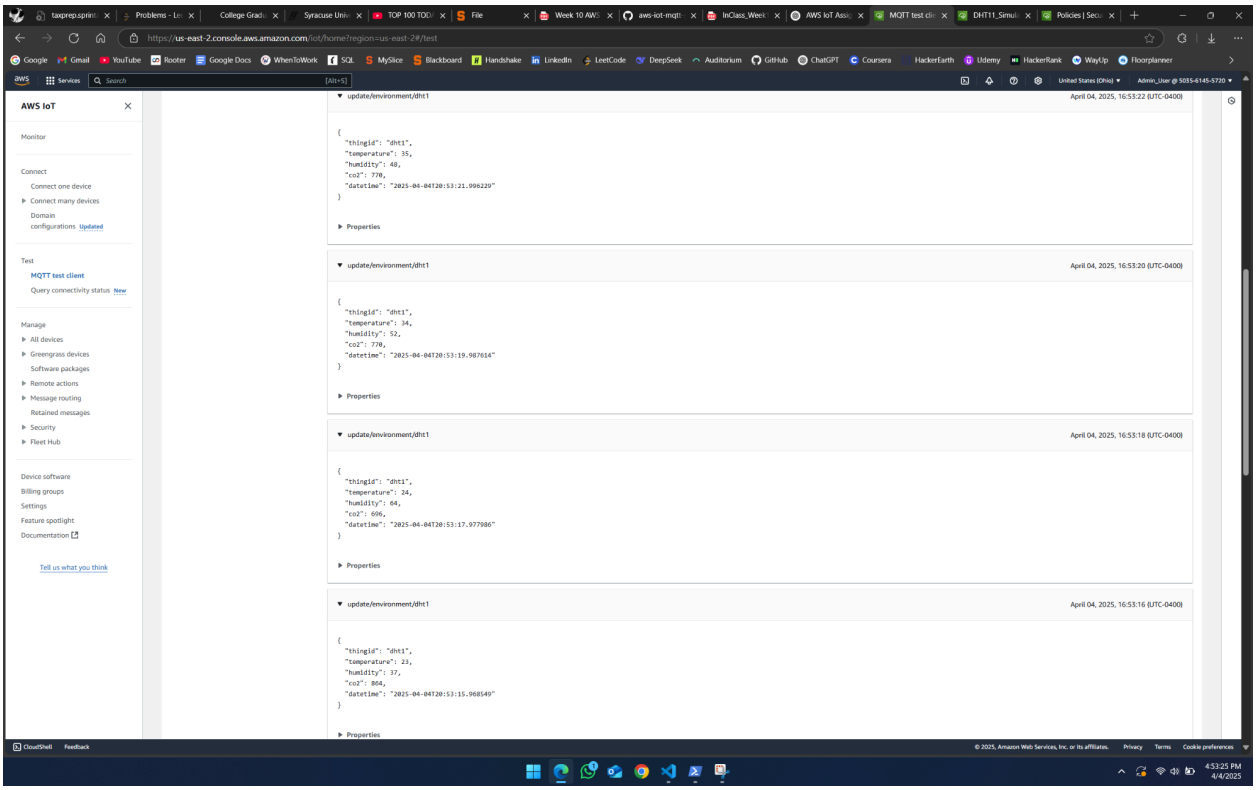

AWS IoT
You successfully created certificate 646f54239edf47ec7ba0e80ced1a65d7c400a1032d116fd80042687a9b4f653.
View certificate
Notifications
AWS IoT > Manage > Things > DHT11_Simulator
DHT11_Simulator
Connectivity indexing is not enabled
Info
Create secure tunnel
Edit
Delete
Thing details
Name
DHT11_Simulator
Type
-
ARN
arn:aws:iot:us-east-2:503561455720:thing/DHT11_Simulator
Billing group
-
Attributes
Certificates
Thing groups
Command history
Device Shadows
Activity
Packages and versions
Jobs
Alarms
Defender metrics
Attributes (0) Info
Attributes are key-value pairs that can be searchable or non-searchable. Searchable attributes can be used to filter lists of things without using fleet indexing. Non-searchable attributes can be used to find things, but only when fleet indexing is turned on.
Key
Value
Type
No attributes
There are no attributes to display.
Propagating attributes (0) Info
Propagating attributes allow you to enhance MQTT messages published by devices.
Propagating attribute type
Attribute type
Attribute name
Override
No attributes
There are no propagating attributes to display
Monitor
Connect
Connect one device
Connect many devices
Domain configurations Updated
Test
MQTT test client
Query connectivity status New
Manage
All devices
Things
Thing groups
Thing types
Fleet metrics
Greengrass devices
Software packages
Remote actions
Message routing
Retained messages
Security
Fleet Hub
Device software
Billing groups
Settings
Feature spotlight
Documentation
Tell us what you think
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
4:26:43 PM
4/4/2025

Running the Python Simulator Script :

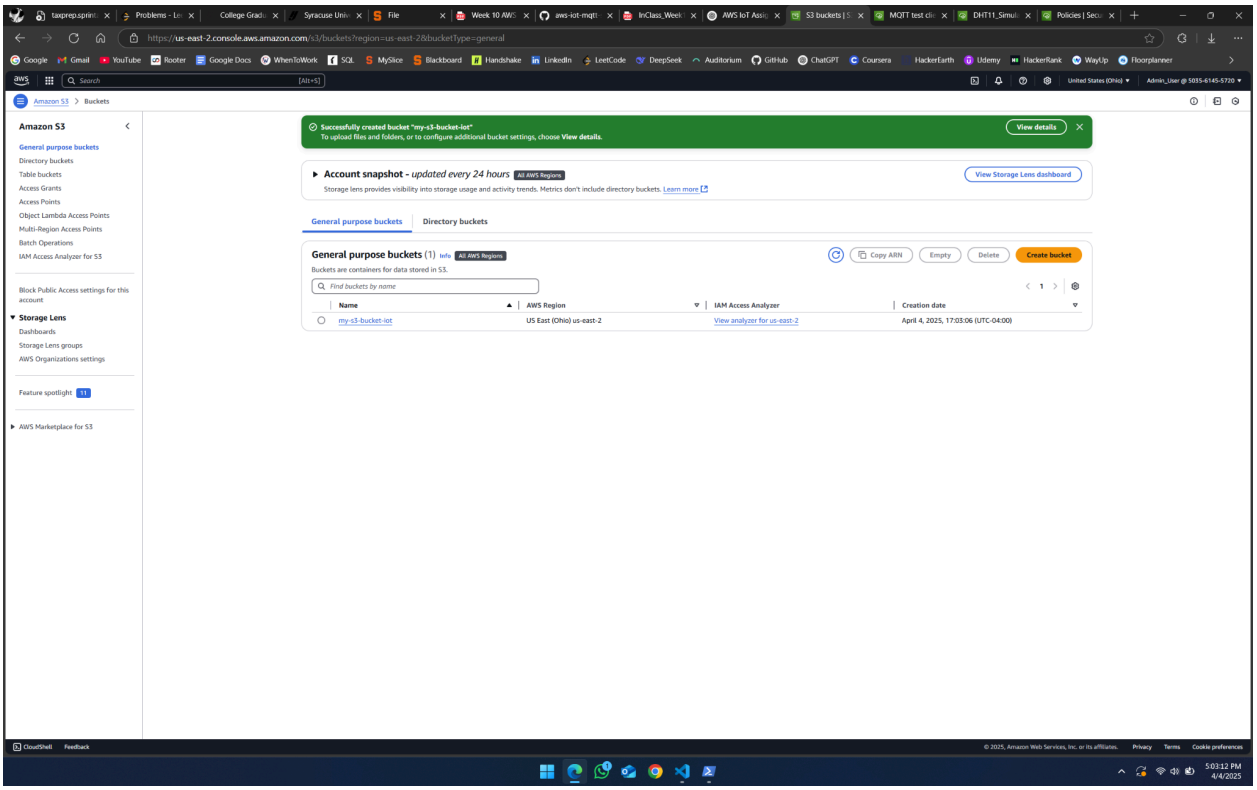


Viewing Live MQTT Messages in AWS IoT Core Test Client :

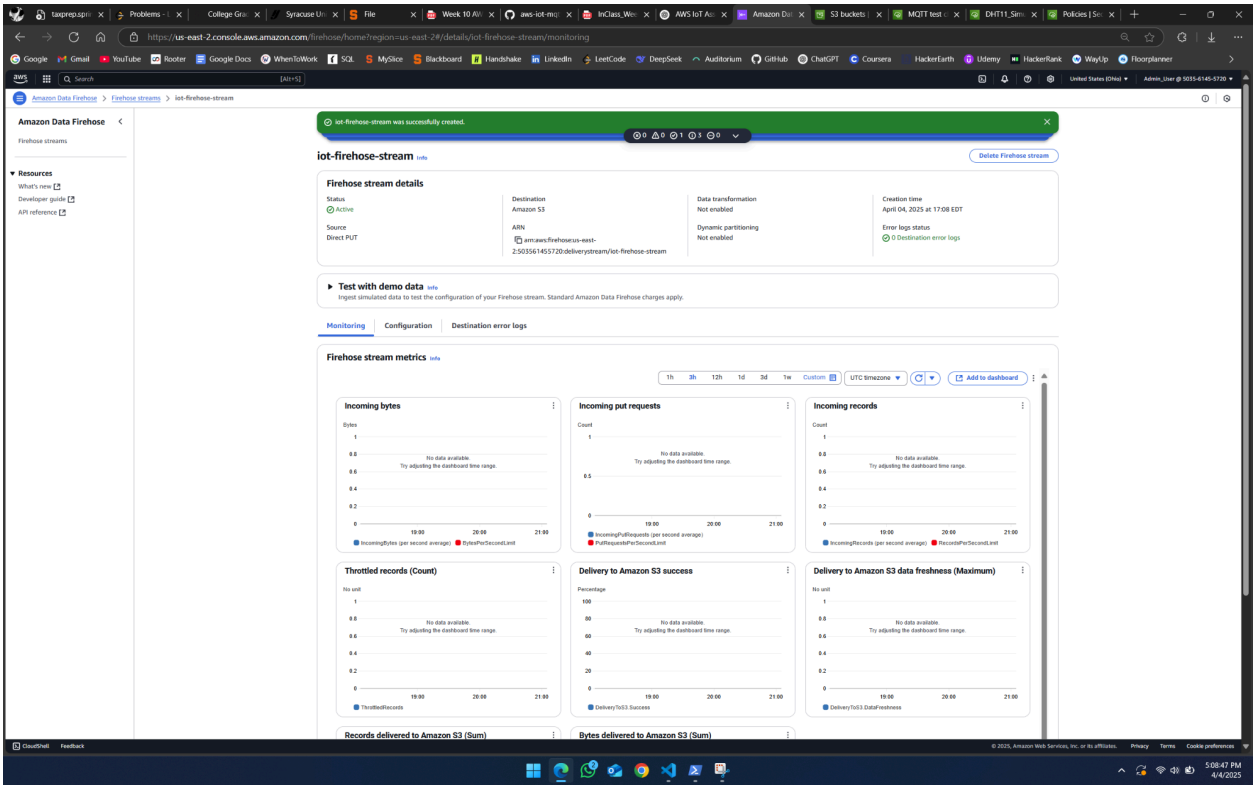


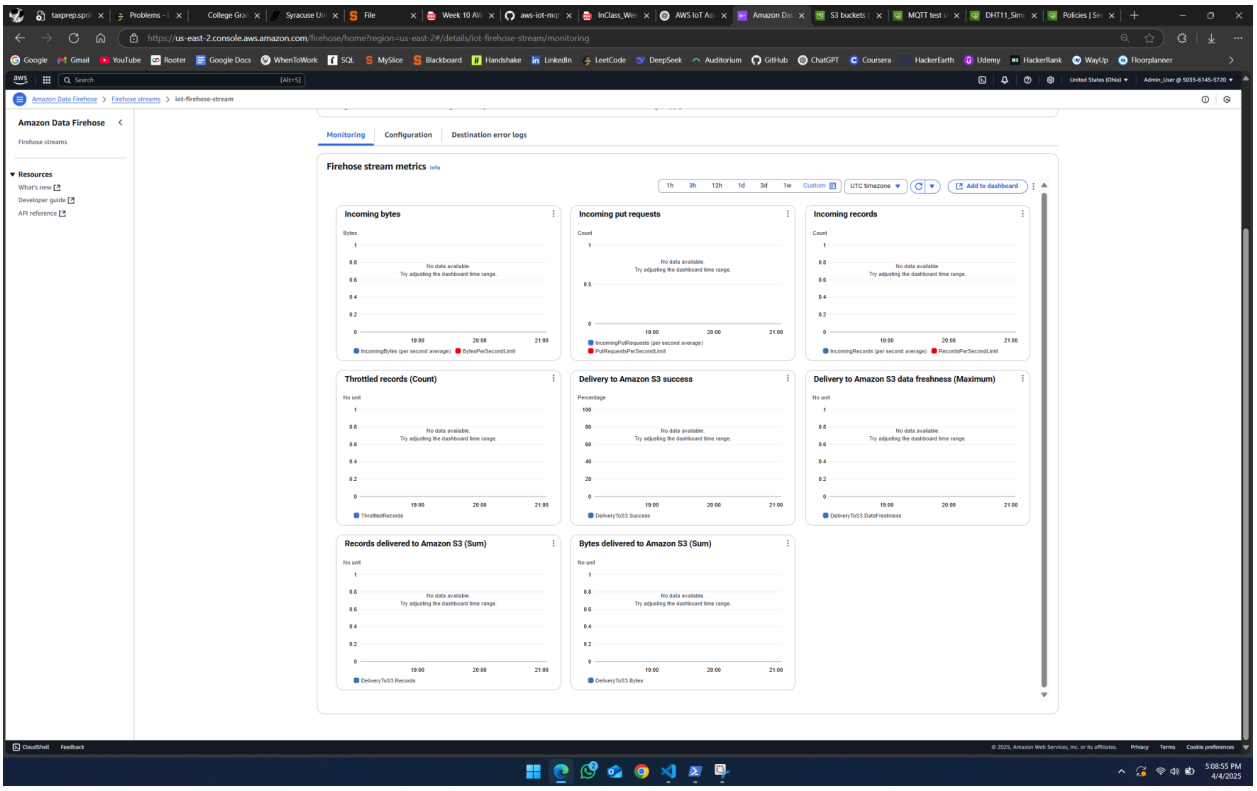


Creating and Configuring the Amazon S3 Bucket :

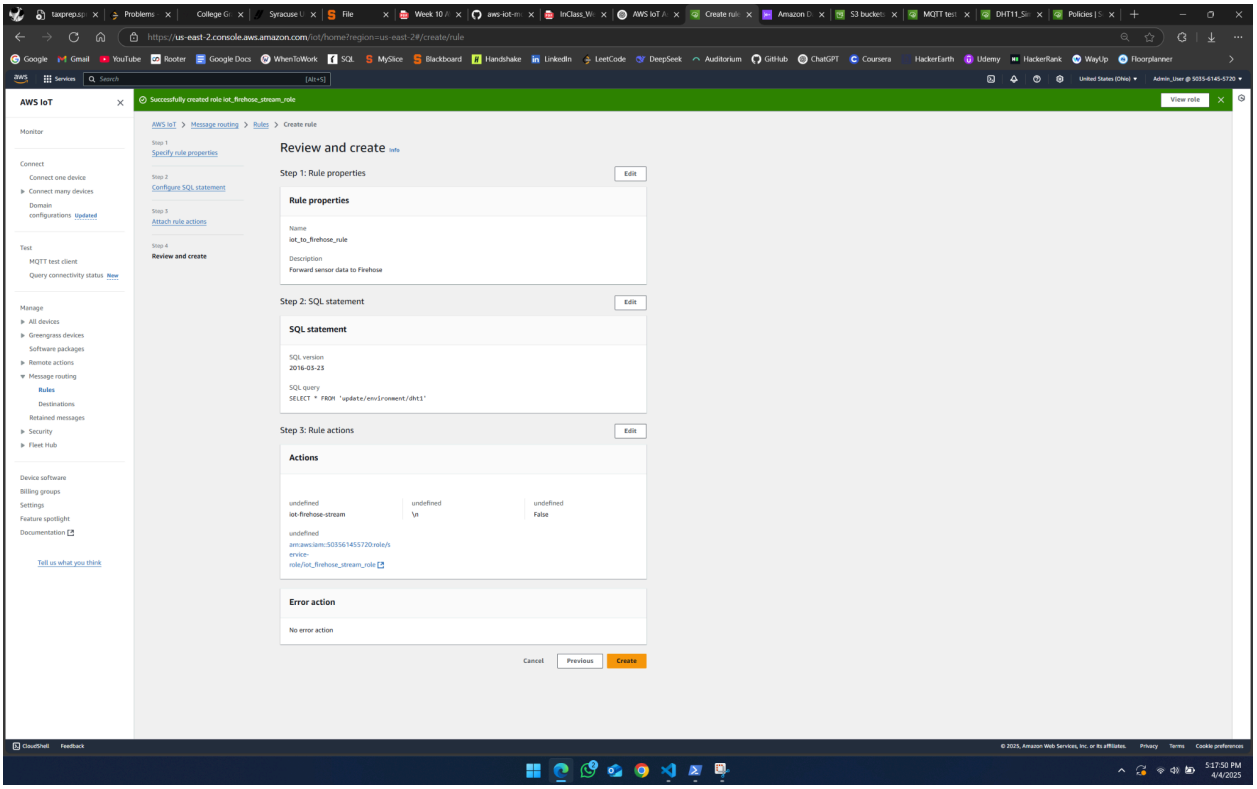


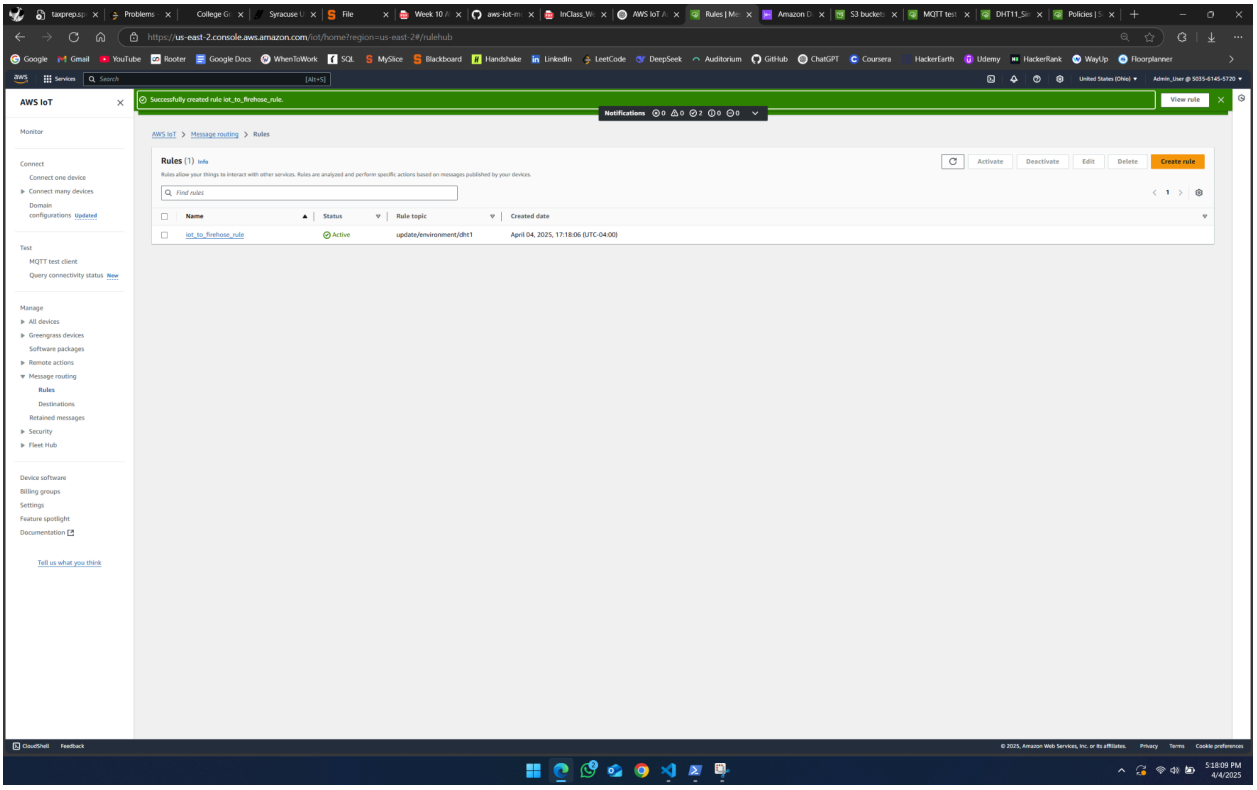
Creating Kinesis Data Firehose Stream to S3 :

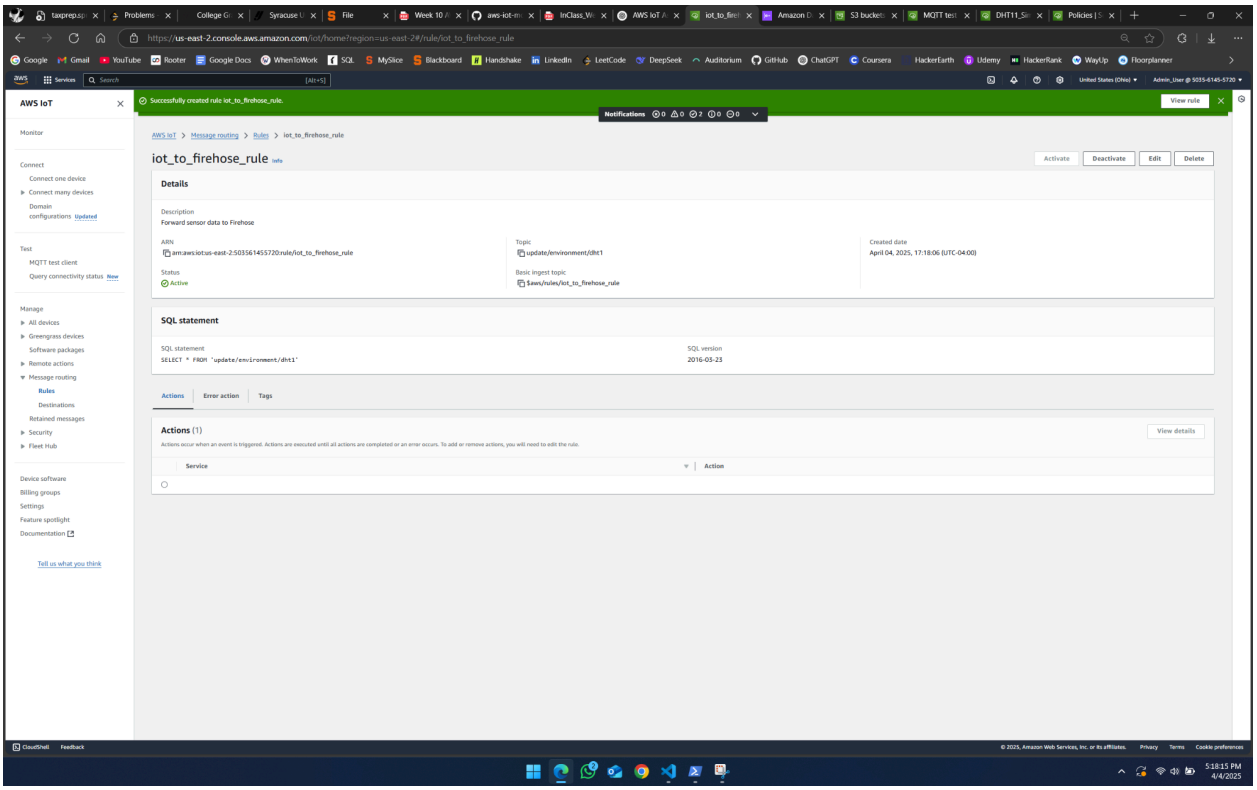




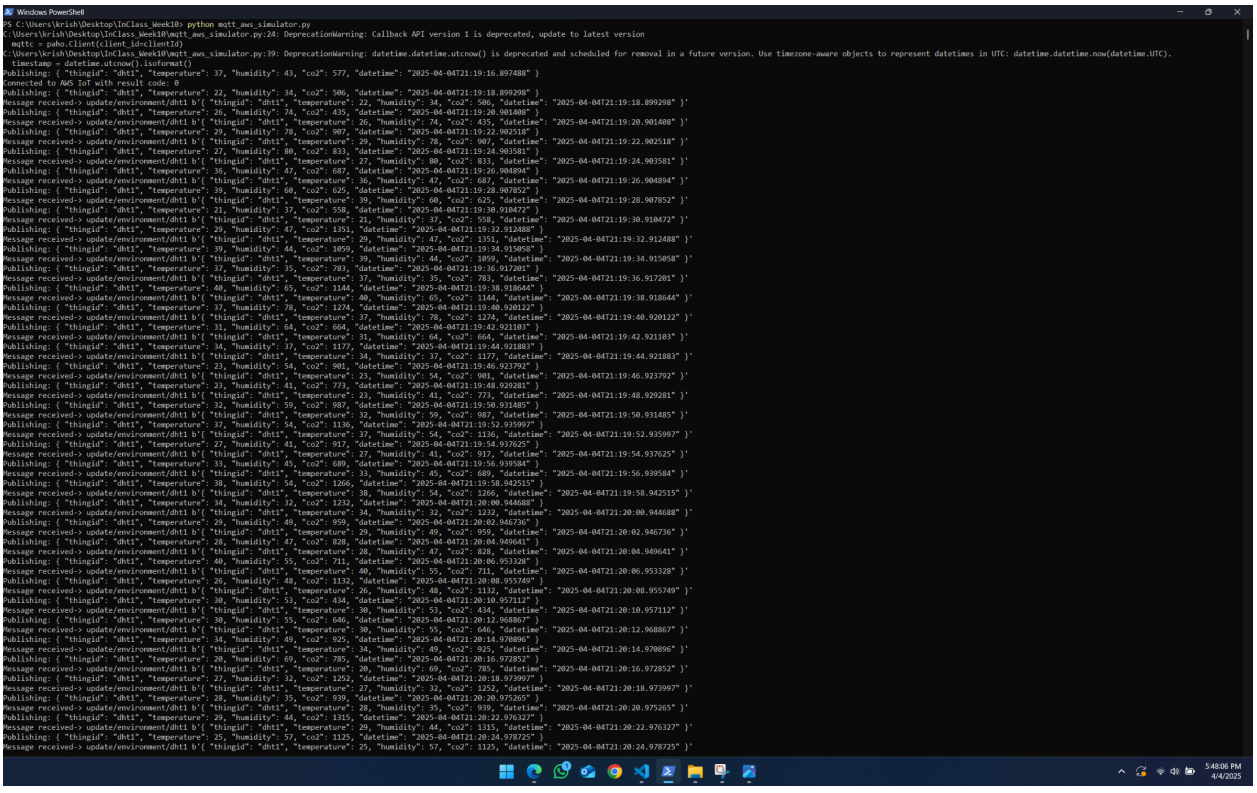
Creating IoT Rule to Forward MQTT Data to Firehose - Completing IoT Rule Setup and Activating Data Pipeline :



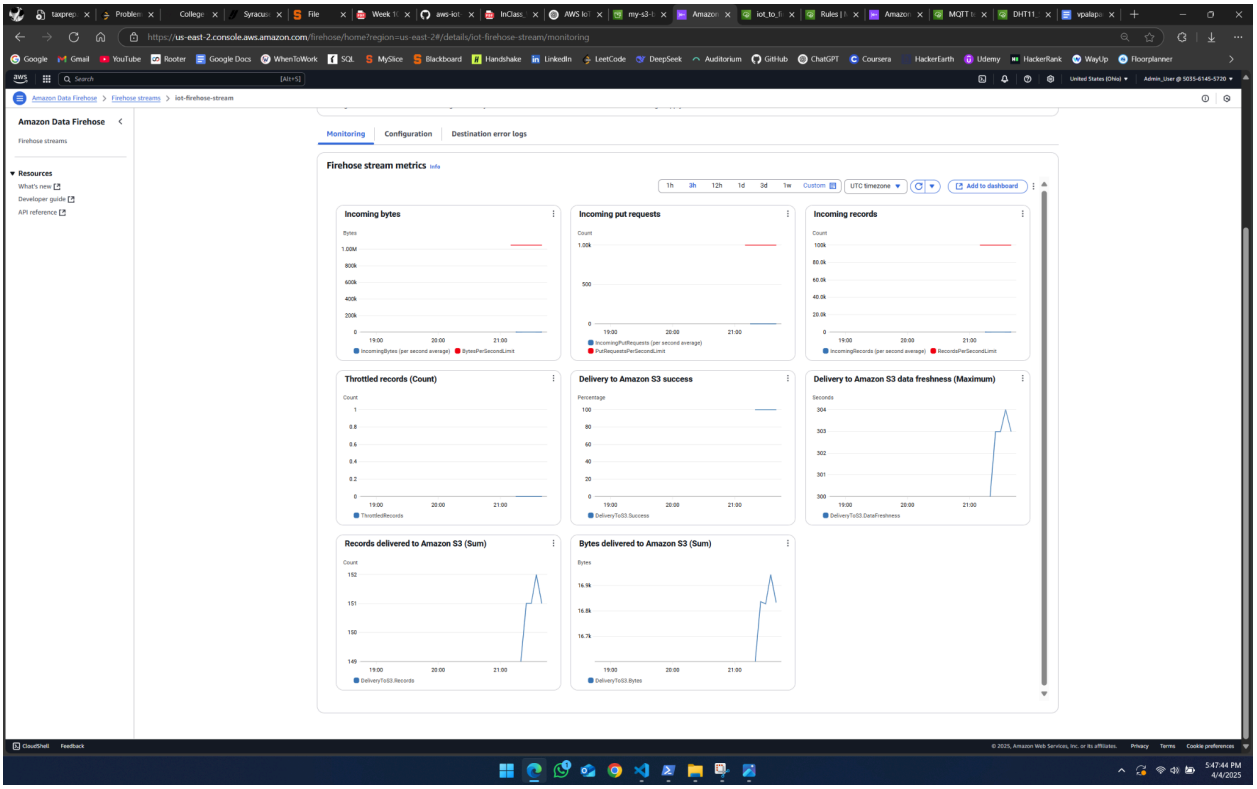




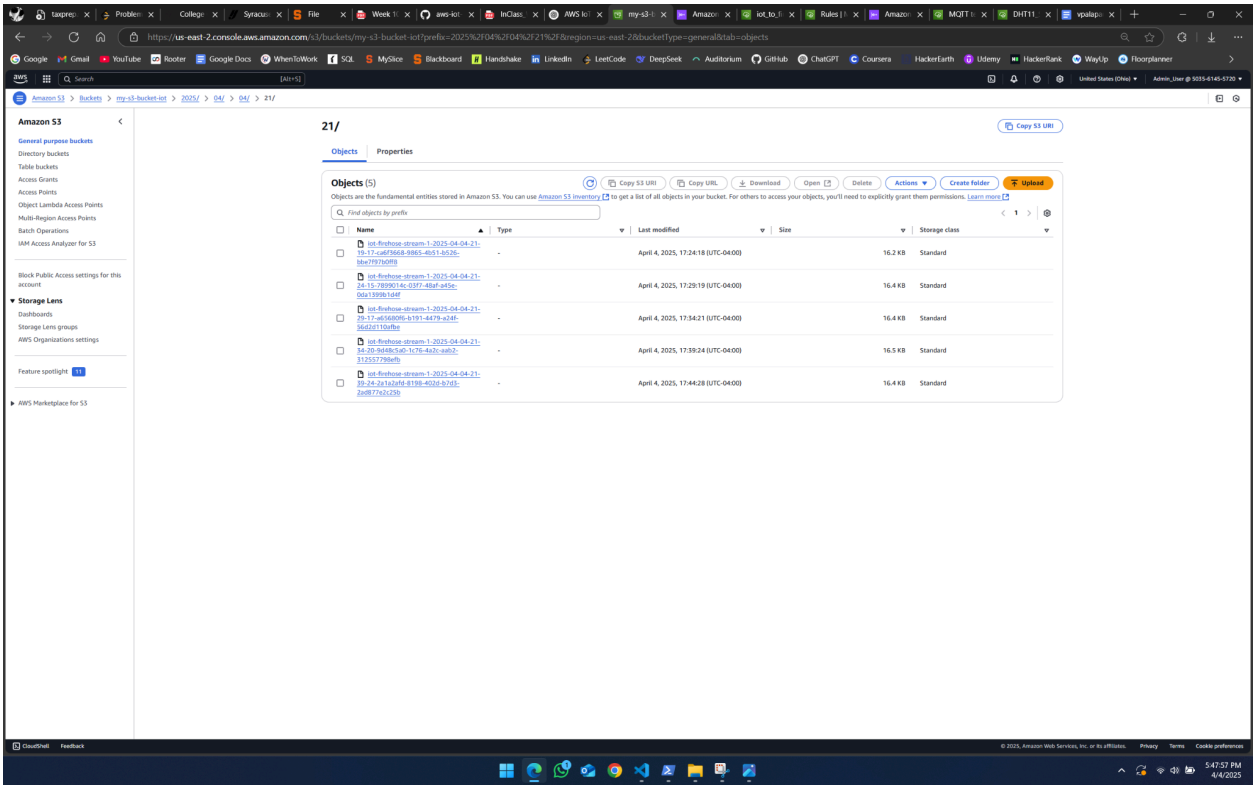
Re-running Python Script to Send Data :



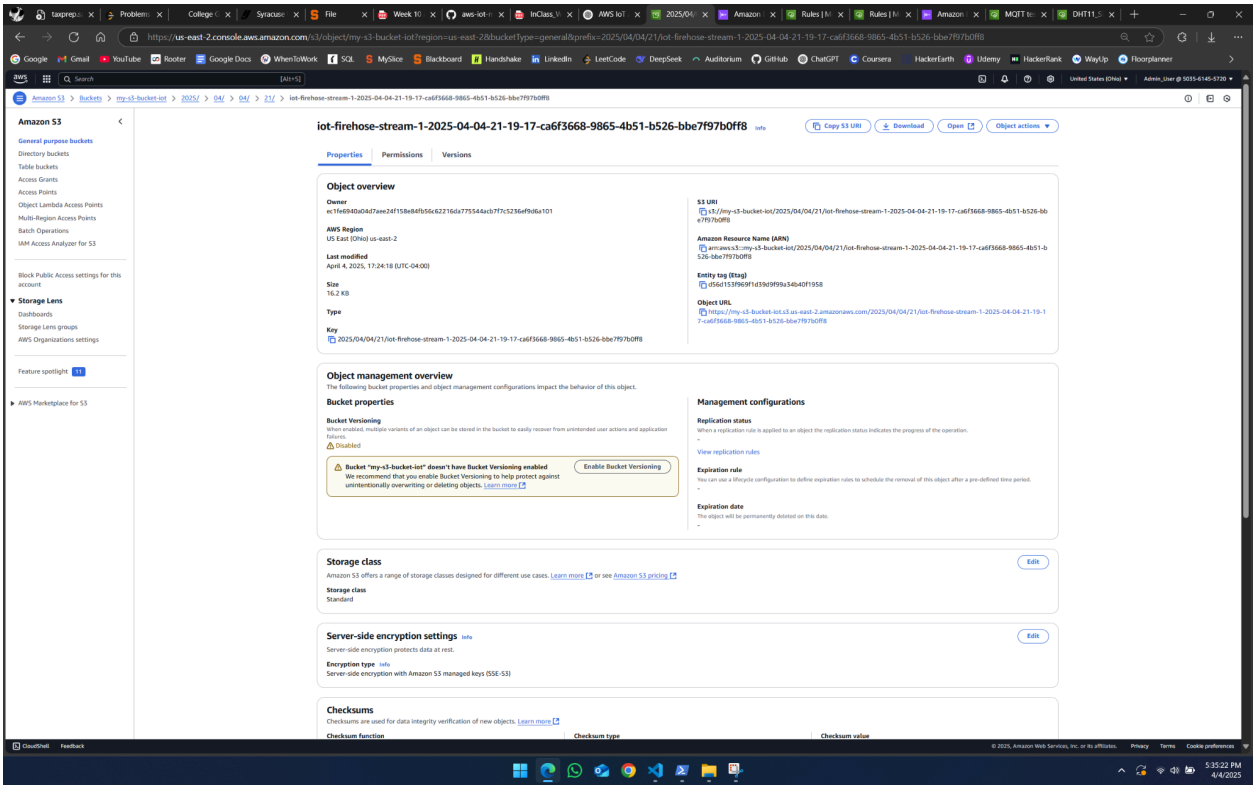
Confirming Firehose Delivery Metrics and Success :



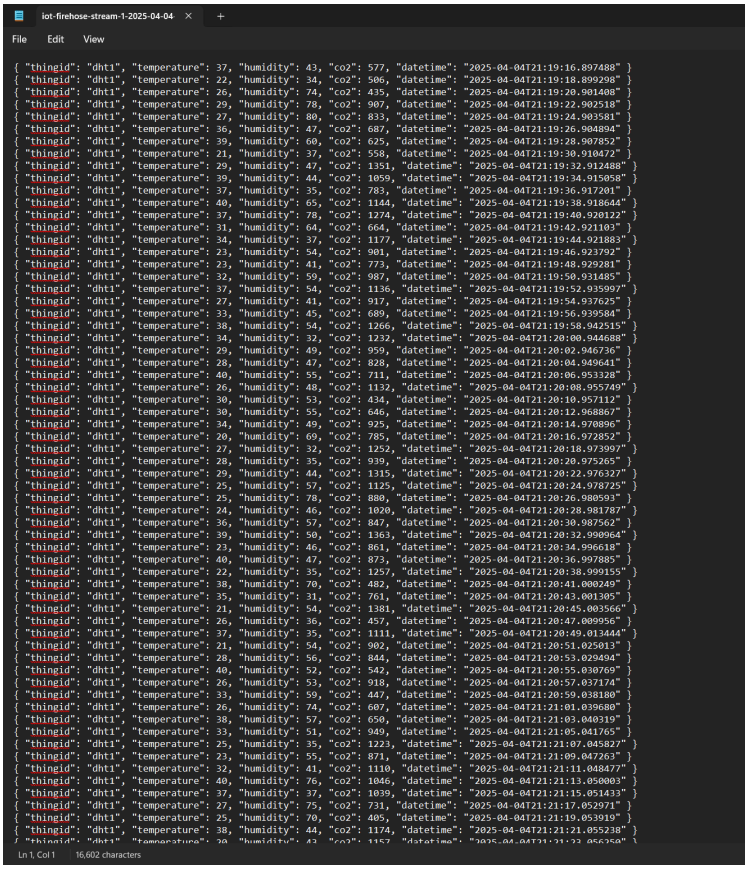
S3 Bucket Showing Delivered IoT Data Files :



Inspecting Metadata of an S3 Data Object :



Viewing JSON Data from Downloaded S3 Object :



Reflection :

This exercise gave me hands-on experience in developing an actual IoT data pipeline using AWS services. It made me understand how various cloud components - AWS IoT Core, Kinesis Firehose, and S3 - can be integrated together to mimic and store sensor data effectively. Key management and security certificate implementation made sure there is firm compliance with the function of secure communication in IoT usage. On top of that, Firehose setup and IAM role creation gave practical insight into data movement and access control management with AWS tools. There were a few technical problems that I faced during this process, especially when I tried to integrate the MQTT simulator with the AWS IoT Core endpoint and configure the Firehose destination. These were dealt with by proper troubleshooting and referring to AWS documentation and examples. Overall, this exercise taught key concepts in cloud-based IoT development and how data can flow seamlessly from mock devices to the cloud storage. Effective end-to-end implementation gave a sense of achievement and a solid grounding for more involved IoT and analytics projects.

GitHub URL : https://github.com/sivaramakrishna6768/aws-iot-mqtt-simulator-2