CIS600 Internet of Things: Application Development

## Step-by-Step Implementation:

* Python MQTT Simulator
  + - Developed a Python script (mqtt\_aws\_simulator.py) to simulate environmental sensor data (temperature, humidity, and CO₂).
    - Set up secure MQTT communication using paho-mqtt and AWS IoT certificates.
    - Generated random sensor data and published to the topic: update/environment/dht1.
* AWS IoT Core Configuration
  + - Created a Thing and downloaded certificates (certificate, private key, public key, root CA).
    - Used endpoint from AWS IoT Core (\*.amazonaws.com) and configured it in the simulator.
* Test Connection
  + - Used AWS IoT Core's MQTT test client to verify data publishing on the topic.
    - Successfully saw live messages appearing in the test client window.
* Create AWS IoT Rule
  + - Created a rule named RouteSensorData.

SQL Query:

SELECT \* FROM 'update/environment/dht1'

* + - The rule was configured to route messages to the AWS IoT Analytics pipeline.
* IoT Analytics Setup
  + - Channel: Created to collect raw MQTT data.
    - Pipeline: Connected the channel to a data store.
    - Data Store: Configured to store incoming JSON data.
    - Dataset: Created a SQL query to retrieve and preview sensor data.
* Testing and Monitoring
  + - Monitored message ingestion in AWS IoT Analytics → Channels.
    - Verified data pipeline and successful dataset generation.

## Screenshots of Output:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

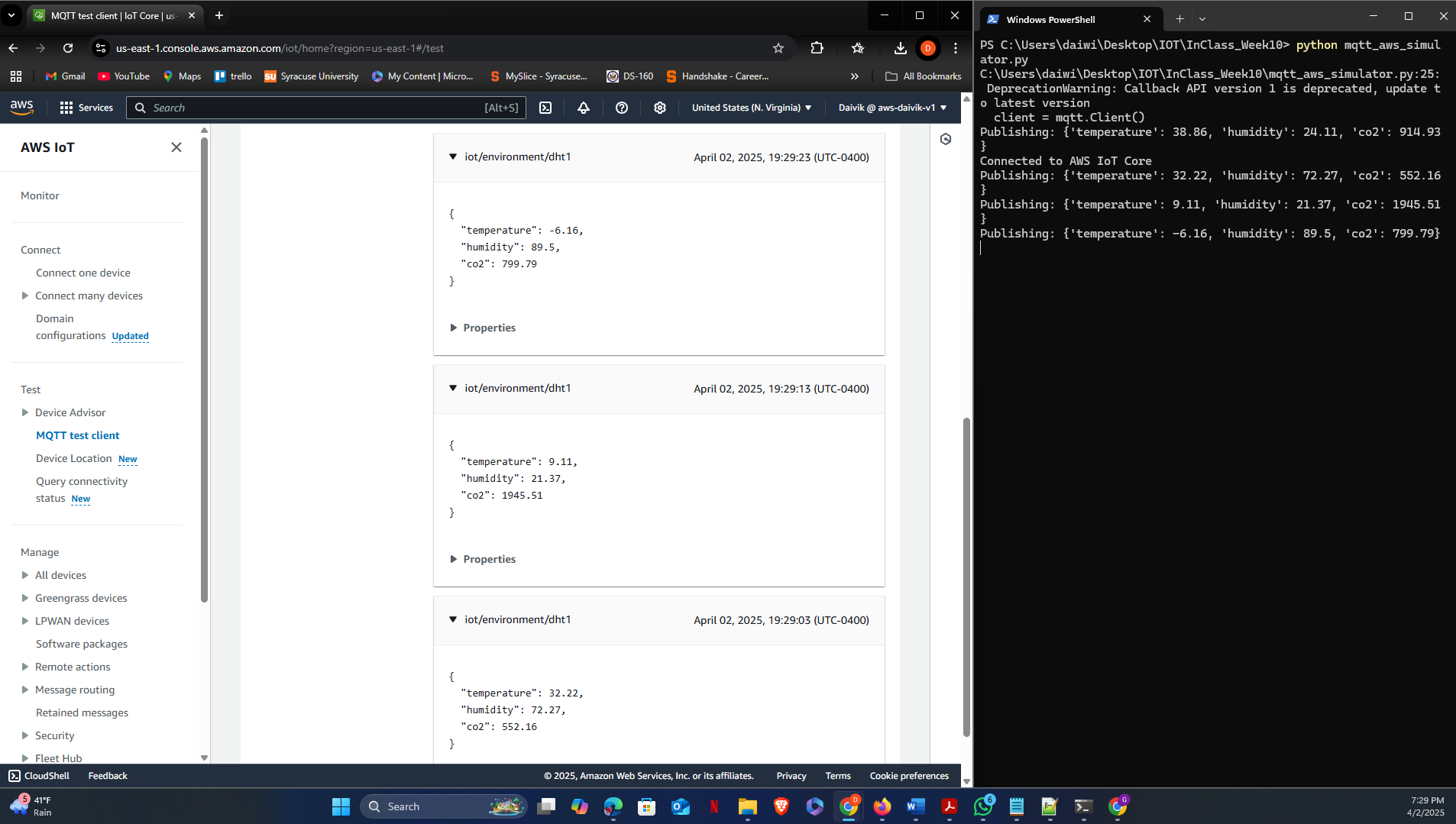
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



## Reflection:

I obtained practical experience simulating real-time data from virtual sensors using AWS IoT services during this assignment. Setting up a secure connection with MQTT using AWS IoT Core was one of the first difficulties I encountered. I had some issues with wrong locations and mismatched certificate files, which at first resulted in connection failures. It seemed like a huge victory when I was able to connect and publish sensor data after troubleshooting and fixing those problems.

I gained a thorough grasp of how AWS manages streaming IoT data by developing the IoT Analytics pipeline, which includes the channel, pipeline, data store, and dataset. It was fascinating to observe how each part contributed to the processing and storing of data as well as how insights could be extracted from the dataset using SQL queries. I felt more assured that the system was functioning flawlessly after testing the data flow with AWS's MQTT client and viewing real-time updates.

Overall, this assignment improved my understanding of cloud-based IoT systems and how various AWS services work together. It was a terrific learning experience that increased both my technical abilities and confidence in creating scalable IoT solutions.

## GitHub Repository:

https://github.com/Daivik-Gangappa/aws-iot-mqtt-simulator-2