title: 'Lab 5: Connecting IoT to the Cloud with Microsoft Azure'

Lab 5: Connecting IoT to the Cloud with Microsoft Azure

Group Members:

- Member 1: Khalil Ahmad
- Member 2: Apala Pramanik
- Member 3: Audrey Vazzana
- Member 4: Raoul

Table of Contents

1. Introduction

In this lab, we utilized Azure IoT Central to simulate a cloud-based IoT application. Azure IoT Central is a FaaS that simplifies device management and telemetry data collection. Our goal was to create an IoT Central application, configure virtual devices to send telemetry data, and visualize this data on a custom dashboard. Each group member created and deployed their own virtual device to the cloud.

2. Development Process

In this section, we document the steps taken during the development of our IoT Central integration and highlight the challenges we faced along the way.

1. **Project Planning and Initial Setup**

The initial phase of development involved researching the Azure IoT Central platform and setting up the environment. The main goals were:

- Set up Azure IoT Central.
- Create a virtual device to simulate telemetry data transmission.
- Develop Python scripts for device-to-cloud communication.

Resources that helped during this phase:

- Azure IoT Central Documentation: Provided guidance on how to configure the IoT Central application and set up device credentials.

- Open-source projects and Python IoT libraries: These were referenced for implementing device command handling and telemetry data transmission.

2. **Coding and Device Implementation**

Each team member created a Python script to simulate a virtual device using the `iotc` Python library. Below were the key steps:

- **IoT Central connection**: We used the scope ID, device ID, and device key from our IoT Central application to connect each device.
- **Command Handling**: The devices were designed to receive a `SendData` command from IoT Central. When the command was received, telemetry data for temperature, wind speed, and humidity was generated and sent to the cloud.
- **Telemetry Data**: In addition to command-based data transmission, devices sent telemetry data at regular intervals (every 60 seconds).

3. **Challenges and Troubleshooting**

Several challenges arose during development, and the following troubleshooting techniques were used:

- **IoT Central Connection Issues**: At times, devices failed to connect due to incorrect credentials. The error was traced back to a misconfiguration in the device key or scope ID. Double-checking credentials resolved this issue.
- **Telemetry Data Format**: Initially, telemetry data was not displayed properly on the IoT Central dashboard. We discovered that the data format being sent was incorrect, and converting numerical data to strings resolved the problem.
- **Command Execution Failures**: While testing command execution, some devices failed to respond to the `SendData` command. This was resolved by ensuring that the `on_commands` function was properly registered and called upon receiving commands.

4. **Testing and Validation**

After resolving the major issues, the devices were tested thoroughly to ensure they could:

- Connect to IoT Central.
- Respond to commands.
- Send telemetry data at regular intervals.

We used the IoT Central dashboard to monitor the devices in real-time and validate the data being received.

5. **Lessons Learned**

Through the development process, we gained valuable insights into the complexities of IoT integration. Azure IoT Central simplifies many tasks, but ensuring accurate device communication and telemetry data requires careful attention to detail.

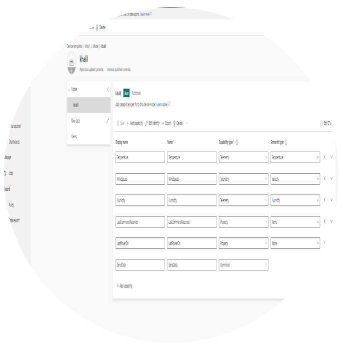
---

3. Device Configuration

While the rest of the code remains same, the IoT central credentials are different for each member. So, we only change the following 3 lines of code and replace the credentials with our own.

Images of device configurations here

Khalil device configuration on Azure IOT Central

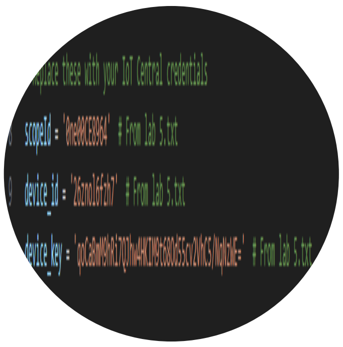


Device templates > khalil > Model > khalil

khalil

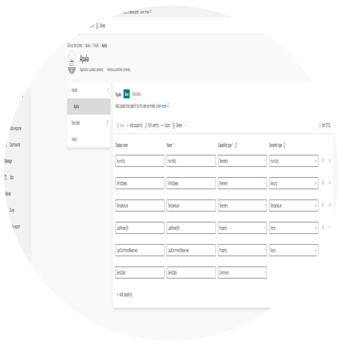
Display name	Name*	Capability type*	Semantic type
Temperature	Temperature	Telemetry	Temperature
WindSpeed	WindSpeed	Telemetry	Velocity
Humidity	Humidity	Telemetry	Humidity
LastCommandReceived	LastCommandReceived	Property	None
LastPowerOn	LastPowerOn	Property	None
SendData	SendData	Command	

Khalil IOT Central credentials



```
Replace these with your IoT Central credentials
scopeId = '0ne00CE8964' # From lab 5.txt
device_id = '26zno16fzh7' # From lab 5.txt
device_key = 'qoCa8mM9hRi7QJhw4HKIM9t680d55cv2VhC5/NqNzWE=' # From lab 5.txt
```

Apala device configuration on Azure IOT Central

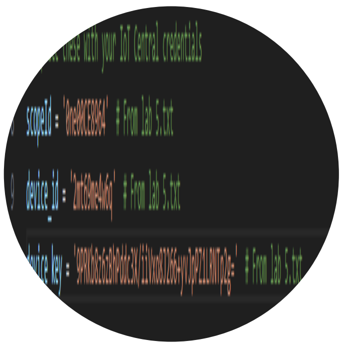


Device templates > Apala > Model > Apala

Apala

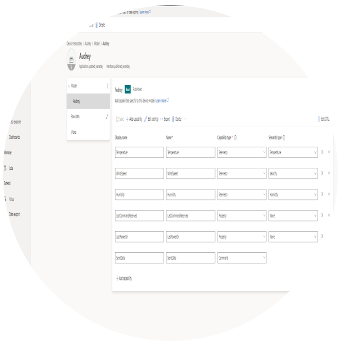
Display name	Name*	Capability type*	Semantic type
Humidity	Humidity	Telemetry	Humidity
WindSpeed	WindSpeed	Telemetry	Velocity
Temperature	Temperature	Telemetry	Temperature
LastPowerOn	LastPowerOn	Property	None
LastCommandReceived	LastCommandReceived	Property	None
SendData	SendData	Command	

Apala IOT Central credentials

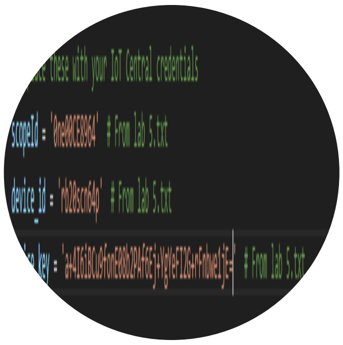


```
... these with your IoT Central credentials
scopeId = '0ne00CE8964' # From lab 5.txt
device_id = '2mt69me4w6q' # From lab 5.txt
device_key = '9PRKb8z6zBhPddc3X/iiVxo8J266+yvJpPZ1LRNTp2g=' # From lab 5.txt
```

Audrey device configuration on Azure IOT Central

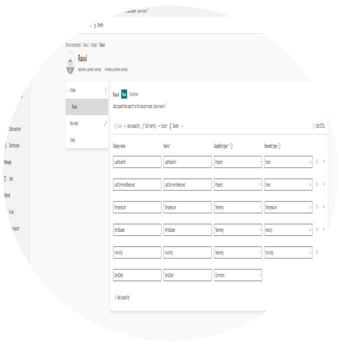


Audrey IOT Central credentials

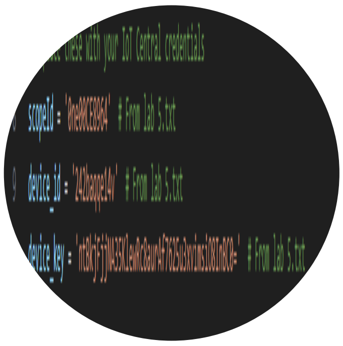


```
...ate these with your IoT Central credentials
scopeId = '0ne00CE8964' # From lab 5.txt
device_id = 'rb20scn64p' # From lab 5.txt
...ice_key = 'a+4I6iBCu9fonE08b2PAf6Ej+YgYeFI2G+rFnbwe1jE=' # From lab 5.txt
```

Raoul device configuration on Azure IOT Central



Raoul IOT Central credentials



```
...ate these with your IoT Central credentials
scopeId = '0ne00CE8964' # From lab 5.txt
device_id = '242baqqe14v' # From lab 5.txt
device_key = 'ntBkjFjjNA35K1ewRc8aurAf7625u3xvimsiO8InBC0=' # From lab 5.txt
```

---

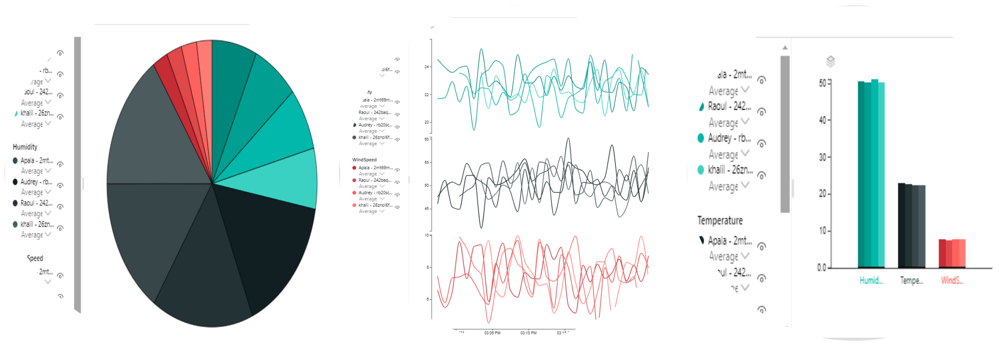
4. Dashboard Creation

We created a custom dashboard to visualize the telemetry data from all devices. The following charts were included:

1. **Pie Chart**: Visualizes temperature, humidity and windspeed readings from all 4 devices as a pie chart.

2. **Line Chart**: Visualizes temperature, humidity and windspeed readings from all 4 devices as a line chart.
3. **Bar Chart**: Visualizes temperature, humidity and windspeed readings from all 4 devices as a bar chart.

Dashboard screenshots here



---

5. Virtual Device Implementation

Each group member implemented their virtual device using the `iotc` Python library. Below is the code for one of the devices:

---

1. **IoT Central Credentials**

While the rest of the code remains same, the IoT central credentials are different for each member. So, we only change the following 3 lines of code and replace the credentials with our own. The scope ID, device ID, and device key are set with values obtained from a configuration file (`lab 5.txt`). These credentials allow the device to securely connect to IoT Central.

```
scopeId = '0ne00CE8964'  # From lab 5.txt
device_id = '26znol6fzh7'  # From lab 5.txt
device_key = 'qoCaBmM9hRi7QJhw4HKIM9t68Od55cv2VhC5/NqNzWE='  # From lab 5.txt
```

---

2. **Command Handling Function**

This function handles incoming commands sent from IoT Central. If the `SendData` command is received, the device generates and sends telemetry data (temperature, wind speed, and humidity) and updates a property (`LastCommandReceived`).

```
def on_commands(command: Command):
    print(f"Command received: {command.name}")

    if command.name == "SendData":
        print("Sending data as per the SendData command")
        iotc.send_telemetry({
            'Temperature': str(random.uniform(15.0, 30.0)),
            'WindSpeed': str(random.uniform(0.0, 15.0)),
            'Humidity': str(random.uniform(30.0, 70.0))
        })
        iotc.send_property({
            "LastCommandReceived": time.time()
        })
        print("Telemetry and LastCommandReceived property sent")

    command.reply()
```

---

3. **IoT Central Client Initialization**

The IoT Central client is initialized with the device credentials and the connection type is set to use a device key. This establishes the device's connection to IoT Central.

```
iotc = IoTCClient(
    device_id,
    scopeId,
    IOTCConnectType.IOTC_CONNECT_DEVICE_KEY,
    device_key
)

iotc.connect()
```

---

4. **Setting Up Command Listener**

Once the connection is established, a listener is set up to handle incoming commands. The `on_commands` function is passed to the command event handler.

```
iotc.on(IOTCEvents.IOTC_COMMAND, on_commands)
```

---

5. **Sending Device Properties**

Upon powering up the device, it sends an initial property (`LastPowerOn`) to IoT Central, which contains the timestamp of when the device was last powered on.

```
iotc.send_property({
    "LastPowerOn": time.time()
})

print("Device connected and LastPowerOn property sent")
```

6. **Telemetry Data Generation and Transmission**

In a continuous loop, telemetry data for temperature, wind speed, and humidity is randomly generated and sent to IoT Central every 60 seconds. The `is_connected()` method ensures that the loop runs only when the device is connected.

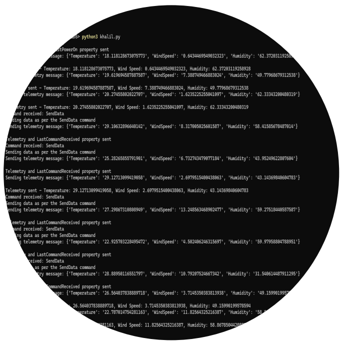
```
while iotc.is_connected():
    temperature = random.uniform(15.0, 30.0)
    wind_speed = random.uniform(0.0, 15.0)
    humidity = random.uniform(30.0, 70.0)

    iotc.send_telemetry({
        'Temperature': str(temperature),
        'WindSpeed': str(wind_speed),
        'Humidity': str(humidity)
    })

    print(f"Telemetry sent - Temperature: {temperature}, Wind Speed: {wind_speed}, Humidity: {humidity}")

    time.sleep(60)
```

6. Results

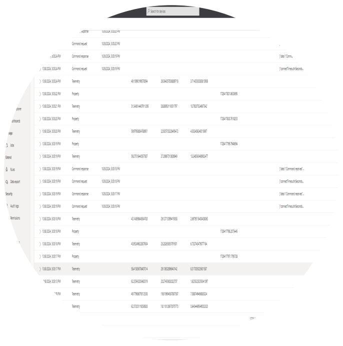
- All devices successfully sent telemetry data to IoT Central.
- The commands from the IoT Central were executed on the virtual devices.
- We monitored the data in real-time using the dashboard, observing telemetry values and device status.

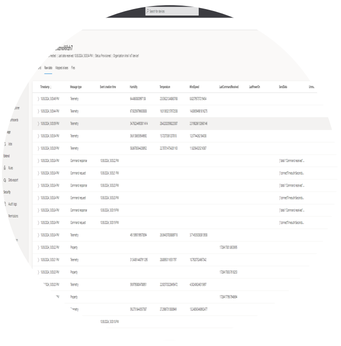
Images of data on Azure IoT Central here

Khalil's Data

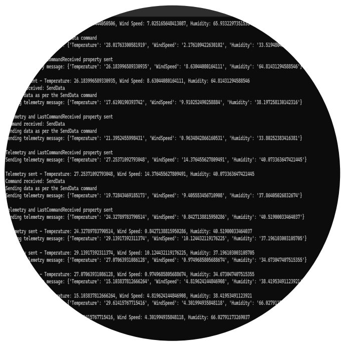


```
5> python3 khalil.py

astPowerOn property sent
ssage: {'Temperature': '18.118128673075773', 'WindSpeed': '0.6434469549032323', 'Humidity': '62.3720311925

- Temperature: 18.118128673075773, Wind Speed: 0.6434469549032323, Humidity: 62.37203119258928
etry message: {'Temperature': '19.619694587887587', 'WindSpeed': '7.388749466883024', 'Humidity': '49.77968679312538'}

 sent - Temperature: 19.619694587887587, Wind Speed: 7.388749466883024, Humidity: 49.77968679312538
telemetry message: {'Temperature': '20.27455802022707', 'WindSpeed': '1.6235225255041097', 'Humidity': '62.33343200480319'}

etry sent - Temperature: 20.27455802022707, Wind Speed: 1.6235225255041097, Humidity: 62.33343200480319
and received: SendData
ding data as per the SendData command
ending telemetry message: {'Temperature': '29.10632896640142', 'WindSpeed': '8.317005025601587', 'Humidity': '58.41585678487014'}

elemetry and LastCommandReceived property sent
Command received: SendData
Sending data as per the SendData command
Sending telemetry message: {'Temperature': '25.28265855791901', 'WindSpeed': '6.732743479077184', 'Humidity': '43.95249622807604'}

Telemetry and LastCommandReceived property sent
Sending telemetry message: {'Temperature': '29.12713099419058', 'WindSpeed': '2.6979515400438063', 'Humidity': '43.14369840604783'}

Telemetry sent - Temperature: 29.12713099419058, Wind Speed: 2.6979515400438063, Humidity: 43.14369840604783
Command received: SendData
Sending data as per the SendData command
ending telemetry message: {'Temperature': '27.29867310808949', 'WindSpeed': '13.248563468902477', 'Humidity': '59.27518440557587'}

lemetry and LastCommandReceived property sent
mand received: SendData
ing data as per the SendData command
g telemetry message: {'Temperature': '22.925703228495472', 'WindSpeed': '4.502406246315697', 'Humidity': '59.97958804788951'}

 and LastCommandReceived property sent
ceived: SendData
 as per the SendData command
try message: {'Temperature': '28.88950116551797', 'WindSpeed': '10.79207524667342', 'Humidity': '31.540614487911295'}

CommandReceived property sent
sage: {'Temperature': '26.564037838889718', 'WindSpeed': '3.7145350383813938', 'Humidity': '49.1599019957

 26.564037838889718, Wind Speed: 3.7145350383813938, Humidity: 49.15990199578594
perature': '22.787014754281163', 'WindSpeed': '11.82564325216387', 'Humidity': '58.

281163, Wind Speed: 11.82564325216387, Humidity: 58.867850442080
```

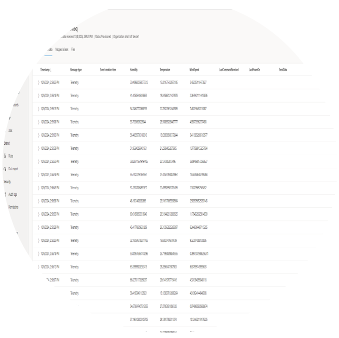




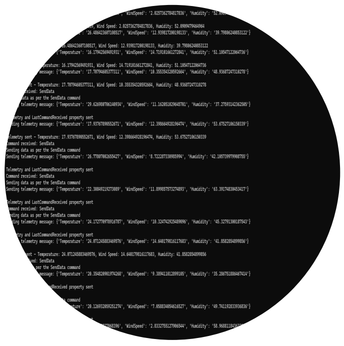
Apala's Data



```
4050506, Wind Speed: 7.025165648413087, Humidity: 65.93322973515
ta command
: {'Temperature': '28.01763300581919', 'WindSpeed': '2.176109422630102', 'Humidity': '33.5194

CommandReceived property sent
y message: {'Temperature': '26.183996589330935', 'WindSpeed': '8.63044080164111', 'Humidity': '64.81431294588546'}

ent - Temperature: 26.183996589330935, Wind Speed: 8.63044080164111, Humidity: 64.81431294588546
ceived: SendData
data as per the SendData command
g telemetry message: {'Temperature': '17.6190190393742', 'WindSpeed': '9.910252490258884', 'Humidity': '38.197258130142316'}

emetry and LastCommandReceived property sent
mand received: SendData
ending data as per the SendData command
ending telemetry message: {'Temperature': '21.3952455998431', 'WindSpeed': '0.9634842866160531', 'Humidity': '33.80252383416381'}

Telemetry and LastCommandReceived property sent
Sending telemetry message: {'Temperature': '27.25371092793048', 'WindSpeed': '14.376455627809491', 'Humidity': '40.0733636474214445'}

Telemetry sent - Temperature: 27.25371092793048, Wind Speed: 14.376455627809491, Humidity: 40.073363647421445
Command received: SendData
Sending data as per the SendData command
Sending telemetry message: {'Temperature': '19.72843469185173', 'WindSpeed': '9.4055534567109008', 'Humidity': '37.86405026832674'}

elemetry and LastCommandReceived property sent
nding telemetry message: {'Temperature': '24.32789783790514', 'WindSpeed': '0.8427138815950286', 'Humidity': '40.51900033464037'}

metry sent - Temperature: 24.32789783790514, Wind Speed: 0.8427138815950286, Humidity: 40.51900033464037
g telemetry message: {'Temperature': '29.13917392311374', 'WindSpeed': '10.124432119176225', 'Humidity': '37.196103003105705'}

 sent - Temperature: 29.13917392311374, Wind Speed: 10.124432119176225, Humidity: 37.196103003105705
lemetry message: {'Temperature': '27.07063931086128', 'WindSpeed': '0.9749685805688674', 'Humidity': '34.673047407515355'}

 - Temperature: 27.07063931086128, Wind Speed: 0.9749685805688674, Humidity: 34.673047407515355
 message: {'Temperature': '15.103837812666264', 'WindSpeed': '4.8196241448846908', 'Humidity': '38.41953491123921

ature: 15.103837812666264, Wind Speed: 4.819624144846908, Humidity: 38.41953491123921
 {'Temperature': '29.61415767715416', 'WindSpeed': '4.301994935848118', 'Humidity': '66.02791

415767715416, Wind Speed: 4.301994935848118, Humidity: 66.02791173269037
```

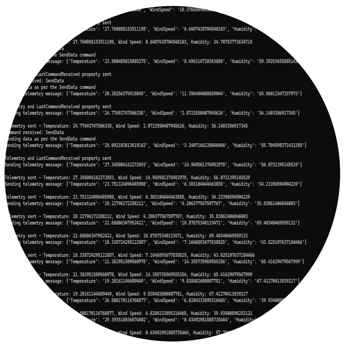


Audrey's Data

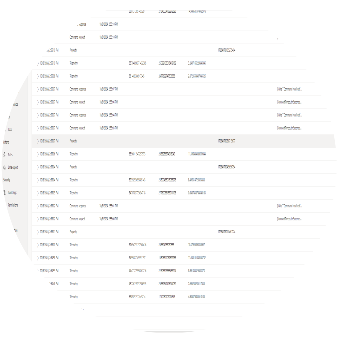


```
 'WindSpeed': '2.0257362784817836', 'Humidity': '52.090

9, Wind Speed: 2.0257362784817836, Humidity: 52.0909479464964
: '26.486423607108517', 'WindSpeed': '12.939817200198133', 'Humidity': '39.79806240853122'}

26.486423607108517, Wind Speed: 12.939817200198133, Humidity: 39.79806240853122
{'Temperature': '16.17942569491931', 'WindSpeed': '14.719101661272841', 'Humidity': '51.10547122064736'}

perature: 16.17942569491931, Wind Speed: 14.719101661272841, Humidity: 51.10547122064736
 message: {'Temperature': '17.70794605377311', 'WindSpeed': '10.355354320592664', 'Humidity': '48.93687247310278'}

t - Temperature: 17.70794605377311, Wind Speed: 10.355354320592664, Humidity: 48.93687247310278
ceived: SendData
ata as per the SendData command
 telemetry message: {'Temperature': '29.626988706140934', 'WindSpeed': '11.162851029645781', 'Humidity': '37.27593142362585'}

metry and LastCommandReceived property sent
ding telemetry message: {'Temperature': '17.93767890552671', 'WindSpeed': '12.398664928196474', 'Humidity': '53.67527106150339'}

elemetry sent - Temperature: 17.93767890552671, Wind Speed: 12.398664928196474, Humidity: 53.67527106150339
ommand received: SendData
Sending data as per the SendData command
Sending telemetry message: {'Temperature': '26.77807002655427', 'WindSpeed': '8.722287330985994', 'Humidity': '42.105739979908755'}

Telemetry and LastCommandReceived property sent
Command received: SendData
Sending data as per the SendData command
Sending telemetry message: {'Temperature': '22.38849219273089', 'WindSpeed': '11.899857573274893', 'Humidity': '63.39174830453417'}

elemetry and LastCommandReceived property sent
ommand received: SendData
nding data as per the SendData command
ding telemetry message: {'Temperature': '24.172770978916787', 'WindSpeed': '10.324742925489096', 'Humidity': '45.32791300187543'}

etry and LastCommandReceived property sent
 telemetry message: {'Temperature': '24.071245883469576', 'WindSpeed': '14.640179816117683', 'Humidity': '41.8582854899856'}

ent - Temperature: 24.071245883469576, Wind Speed: 14.640179816117683, Humidity: 41.8582854899856
ved: SendData
 per the SendData command
 message: {'Temperature': '20.354820901974268', 'WindSpeed': '9.389411012899105', 'Humidity': '35.286751886467414'}

ndReceived property sent

a command
ture': '20.126932059251274', 'WindSpeed': '7.058834054616527', 'Humidity': '49.741192833936836'}

7968396', 'WindSpeed': '2.8332755127906544', 'Humidity': '58.968811843637
```

Raouls's Data



```
 'WindSpeed': '10.37665090

y sent
ure': '27.760088153911198', 'WindSpeed': '0.6407438704540103', 'Humidity':

: 27.760088153911198, Wind Speed: 0.6407438704540103, Humidity: 34.70783773634718
ta
he SendData command
essage: {'Temperature': '23.500485015085275', 'WindSpeed': '8.496314720363888', 'Humidity': '59.3925365588514

LastCommandReceived property sent
ived: SendData
a as per the SendData command
elemetry message: {'Temperature': '20.38256374918049', 'WindSpeed': '11.396404080659044', 'Humidity': '65.96011547257973'}

try and LastCommandReceived property sent
ing telemetry message: {'Temperature': '24.77692747506338', 'WindSpeed': '2.8723550407945626', 'Humidity': '36.1403386917345'}

emetry sent - Temperature: 24.77692747506338, Wind Speed: 2.8723550407945626, Humidity: 36.1403386917345
mmand received: SendData
ending data as per the SendData command
ending telemetry message: {'Temperature': '25.092103013419162', 'WindSpeed': '3.2407166220846046', 'Humidity': '55.704898371432385'}

Telemetry and LastCommandReceived property sent
Sending telemetry message: {'Temperature': '27.345004162272893', 'WindSpeed': '14.949501374982978', 'Humidity': '56.0731395145529'}

Telemetry sent - Temperature: 27.345004162272893, Wind Speed: 14.949501374982978, Humidity: 56.0731395145529
Sending telemetry message: {'Temperature': '23.751324096485988', 'WindSpeed': '6.303104644463858', 'Humidity': '34.23396594904239'}

Telemetry sent - Temperature: 23.751324096485988, Wind Speed: 6.303104644463858, Humidity: 34.23396594904239
ending telemetry message: {'Temperature': '20.22796172288212', 'WindSpeed': '4.206377567597767', 'Humidity': '35.83861406846003'}

lemetry sent - Temperature: 20.22796172288212, Wind Speed: 4.206377567597767, Humidity: 35.83861406846003
ding telemetry message: {'Temperature': '22.686063479524222', 'WindSpeed': '10.87675340133471', 'Humidity': '69.40340469559132'}

etry sent - Temperature: 22.68606347952422, Wind Speed: 10.87675340133471, Humidity: 69.40340469559132
 telemetry message: {'Temperature': '18.338724295122887', 'WindSpeed': '7.1646893677838825', 'Humidity': '43.825107637184466'}

 sent - Temperature: 18.338724295122887, Wind Speed: 7.1646893677838825, Humidity: 43.825107637184466
emetry message: {'Temperature': '21.382951589560978', 'WindSpeed': '14.355735969555356', 'Humidity': '65.41629479567999'}

 - Temperature: 21.382951589560978, Wind Speed: 14.355735969555356, Humidity: 65.41629479567999
 message: {'Temperature': '19.20161144689449', 'WindSpeed': '9.838482600807781', 'Humidity': '67.41270413839217'}

rature: 19.20161144689449, Wind Speed: 9.838482600807781, Humidity: 67.41270413839217
 {'Temperature': '26.508170116768877', 'WindSpeed': '6.8284333895316465', 'Humidity': '39.93400

508170116768877, Wind Speed: 6.8284333895316465, Humidity: 39.93400896253121
ure': '29.393518036074802', 'WindSpeed': '0.43492981805735464', 'Humidity'

 Wind Speed: 0.43492981805735464, Humidity: 57.70
```



7. Conclusion

This lab allowed us to understand how IoT devices connect to the cloud using Azure IoT Central. We were able to create virtual devices, send telemetry data, and respond to cloud commands. The simplicity of the IoT Central platform made it easier to manage multiple devices and analyze the data.

8. Appendix: Code

Python script with proper documentation.

```
import random
import time
from datetime import datetime
from iotc.models import Command
from iotc import IoTCClient, IOTCConnectType, IOTCEvents

# Replace these with your IoT Central credentials
scopeId = '0ne00CE8964'  # From lab 5.txt
device_id = '26znol6fzh7'  # From lab 5.txt
device_key = 'qoCaBmM9hRi7QJhw4HKIM9t68Od55cv2VhC5/NqNzWE='  # From lab 5.txt

# Function to handle incoming commands
def on_commands(command: Command):
    print(f"Command received: {command.name}")

    if command.name == "SendData":
        # Send data when the "SendData" command is received
        print("Sending data as per the SendData command")
        iotc.send_telemetry({
            'Temperature': str(random.uniform(15.0, 30.0)),
            'WindSpeed': str(random.uniform(0.0, 15.0)),
            'Humidity': str(random.uniform(30.0, 70.0))
        })
        # Update the LastCommandReceived property
        iotc.send_property({
            "LastCommandReceived": time.time()
        })
        print("Telemetry and LastCommandReceived property sent")

    # Acknowledge the command
    command.reply()

# Initialize IoTC client
iotc = IoTCClient(
    device_id,
    scopeId,
    IOTCConnectType.IOTC_CONNECT_DEVICE_KEY,
    device_key
)

iotc.connect()

# Set up the command listener
iotc.on(IOTCEvents.IOTC_COMMAND, on_commands)

# Send the LastPowerOn property once the device is powered on
iotc.send_property({
    "LastPowerOn": time.time()
})

print("Device connected and LastPowerOn property sent")
```

```
# Start sending telemetry data every 60 seconds
while iotc.is_connected():
    # Generate and send telemetry data for temperature, wind speed, and humidity
    temperature = random.uniform(15.0, 30.0)  # Simulated temperature in Celsius
    wind_speed = random.uniform(0.0, 15.0)    # Simulated wind speed in km/h
    humidity = random.uniform(30.0, 70.0)     # Simulated humidity in percentage

    # Send telemetry data
    iotc.send_telemetry({
        'Temperature': str(temperature),
        'WindSpeed': str(wind_speed),
        'Humidity': str(humidity)
    })

    print(f"Telemetry sent - Temperature: {temperature}, Wind Speed: {wind_speed}, Humidity: {humidity}")

    # Wait for 60 seconds before sending the next telemetry data
    time.sleep(5)
```