Smart Cities Hackathon Tier 3 Challenge 12

Tier 3: Challenge 12

Tier 3 Challenge 12: Real-time IoT Control Challenge

This challenge will see you extend Tier 2 Challenge 6, if you haven't completed this challenge please complete this before starting this challenge. This challenge will develop a Python script to generate a series of random sensor values and upload these to core.aql.com using the https://api.core.aql.com/

Step 1

Install Docker or configure your local machine to run Python as per **Tier 3 Challenge 10,** or run the scripts directly from your local machine.

Step 2

Log into the aql Core IoT platform (https://core.aql.com/login) with your provided event credentials.

Step 3

Locate the Device ID for the virtual sensor created in Challenge 6

Step 4

Log into https://core.aql.com/login and create a bearer token if you haven't created one from an earlier challenge copy this as you will need this for a following step

Step 5

Create a new Python file in the hackathon folder called challenge12.py

Step 6

Setup the imports and constants, replace with your Bearer token and device ID

```python
import requests
import random
import time


# Constants
CORE_URL = 'https://api.core.aql.com/v1/'
TOKEN = 'REPLACE WITH YOUR TOKEN'
DEVICE_ID = 'REPLACE WITH YOUR BEARER TOKEN'
```

Tier 3: Challenge 12

Step 7

Add the definition to generate a random sensor reading, update this to match the device you created in Challenge 6

```python
def generate_reading():
    """
    Generate a random reading.
    """
    timestamp = int(time.time() * 1000)  # Current timestamp in milliseconds
    latitude = round(random.uniform(-90, 90), 4)  # Random latitude between -90 and 90
    longitude = round(random.uniform(-180, 180), 4)  # Random longitude between -180 and 180
    temperature = round(random.uniform(-20, 40), 1)  # Random temperature between -20 and 40
    humidity = random.randint(0, 100)  # Random humidity between 0 and 100
    co2 = random.randint(300, 1000)  # Random CO2 level between 300 and 1000 ppm
    pm25 = random.randint(0, 50)  # Random PM2.5 level between 0 and 50 µg/m³
    pm10 = random.randint(0, 100)  # Random PM10 level between 0 and 100 µg/m³

    return {"reading": {
        "timestamp": timestamp,
        "latitude": latitude,
        "longitude": longitude,
        "temperature": temperature,
        "humidity": humidity,
        "co2": co2,
        "pm25": pm25,
        "pm10": pm10
        }
    }
```

Step 8

Next add the definition to submit the random reading to the api.core.aql.com end point

```python
def submit_reading(payload):
    """
    Submit a reading to the API.
    """
    url = f"{CORE_URL}devices/{DEVICE_ID}/add-reading"
    try:
        headers = {'Authorization': f'Bearer {TOKEN}', 'Content-Type': 'application/json'}
        response = requests.post(url, json=payload, headers=headers)
        response.raise_for_status()
        print("Reading submitted successfully:", payload)
    except requests.exceptions.RequestException as e:
```

```
        print("Error submitting reading:", e)
```

Step 9

Finally add the generate and submission method

```python
def generate_and_submit_readings(num_readings):
    """
    Generate and submit a specified number of readings.
    """
    for _ in range(num_readings):
        reading = generate_reading()
        submit_reading(reading)
        time.sleep(1)  # Sleep for 1 second between readings

# Test the function
if __name__ == "__main__":
    num_readings = 10  # Number of readings to generate and submit
    generate_and_submit_readings(num_readings)
```

You can complete a final test of the code running

```
docker compose run python-app python challenge12.py
```