1. Połączenie z urządzeniem (serwerem OPC UA) oraz Konfiguracja agenta

- Włączamy program Visual Studio  
- otwieramy plik projektu  


- Włączamy program imitujący zachowanie urządzeń (IIoTSim)



- przechodzimy przez instalację

- dodajemy 5 urządzeń klikając w przycisk ,,New Device”

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

- uruchamiamy program w Visual Studio

Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie

- konfigurujemy połączenia z urządzeniem z Azure oraz serwerem OPC (zatwierdzamy Enterem)

Obraz zawierający tekst, Czcionka, oprogramowanie, zrzut ekranu

Opis wygenerowany automatycznie

(te dane są zapisywane w pliku INI – config.ini, dzięki czemu przy kolejnych odpaleniach programu nie trzeba ich konfigurować. Chcąc je zmienić należy znaleźć plik config.ini w plikach projektu oraz zmienić je ręcznie  
)

2. Rodzaj, format i częstotliwość wiadomości (D2C messages) wysyłanych przez agenta do IoT Hub

  
Powyżej znajduje się wywołanie funkcji wysyłającej wiadomości z parametrami (10 – ilość wiadomości dla jednego urządzenia, 1000- odstęp czasowy między wiadomościami)  
Można to skonfigurować w projekcie Visual Studio, w pliku Program.cs w linii 40.

Przykładowa wiadomość wysyłana do IoT Hub

Obraz zawierający zrzut ekranu, tekst, Czcionka, linia

Opis wygenerowany automatycznie

3. Rodzaj i format danych przechowywanych w Device Twin

Informacje w Device Twin przekazywane są przy pierwszym odpaleniu programu oraz przy zmianach wprowadzonych w ustawieniach urządzeń  
Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

4. Direct Methods

Tak jak było wymagane, możemy wywołać dwie Direct Metody - EmergencyStop oraz ResetErrorStatus  
Testowo można sprawdzić działanie na Device 1

- Wchodzimy w urządzenie w IoT Hub

- wybieramy Metoda bezpośrednia



- wpisujemy wybraną metodę

Obraz zawierający tekst, linia, Czcionka, numer

Opis wygenerowany automatycznie

(wystarczy nazwa metody – nie jest potrzebne nic więcej)

- klikamy ,,Wywołaj metodę”

Jeśli się powiodło dostajemy wynik:  
Obraz zawierający tekst, zrzut ekranu, linia, Czcionka

Opis wygenerowany automatycznie

Oraz efekt:

Obraz zawierający tekst, numer, Czcionka, linia

Opis wygenerowany automatycznie  
- tak samo zadziała z ResetErrorStatus  
efekt:

Obraz zawierający tekst, numer, oprogramowanie, linia

Opis wygenerowany automatycznie

5. Kalkulacje

Podstawowe wymagane:

1.Production KPIs

•% of good production(see good count) in total volume, grouped bydevice in5-minute windows

Obraz zawierający tekst, Czcionka, linia, zrzut ekranu

Opis wygenerowany automatycznie

SELECT

IoTHub.ConnectionDeviceId as deviceId,

MAX(goodCount) as goodCount,

MAX(badCount) as badCount,

CASE

WHEN MAX(goodCount) + MAX(badCount) = 0 THEN NULL

ELSE MAX(goodCount) \* 100.0 / (MAX(goodCount) + MAX(badCount))

END as goodProductionPercentage

INTO

[production-kpis]

FROM

[iot]

TIMESTAMP BY

IoTHub.EnqueuedTime

GROUP BY

IoTHub.ConnectionDeviceId,

TumblingWindow(minute, 5);

2.Temperature

•Every 1 minute give me the average, minimum and maximum temperature over the last 5 minutes (grouped by device)

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

SELECT

System.Timestamp() as windowEndTime,

IoTHub.ConnectionDeviceId as deviceId,

AVG(temperature) as avgTemperature,

MIN(temperature) as minTemperature,

MAX(temperature) as maxTemperature

INTO

[temperature-stats]

FROM

[iot]

TIMESTAMP BY

IoTHub.EnqueuedTime

GROUP BY

IoTHub.ConnectionDeviceId,

TumblingWindow(minute, 5, 1)

3.Device errors

Obraz zawierający tekst, zrzut ekranu, numer, Czcionka

Opis wygenerowany automatycznie

•Situations whenever a device experiences more than 3 errors in under 1minute

SELECT

System.Timestamp() as windowEndTime,

IoTHub.ConnectionDeviceId as deviceId,

COUNT(\*) as errorCount

INTO

[error-stats]

FROM

[iot]

TIMESTAMP BY

IoTHub.EnqueuedTime

WHERE

TRY\_CAST(deviceError AS bigint) > 0

GROUP BY

IoTHub.ConnectionDeviceId,

TumblingWindow(minute, 1)

HAVING

COUNT(\*) > 3;

Dodatkowe:   
Srednia temperatura

SELECT System.Timestamp() as windowEndTime,

IoTHub.ConnectionDeviceId as deviceId,

COUNT(\*) as nrOfMeasurements,

AVG(temperature) as avgTemp

INTO

[avg-temperature]

FROM

iot

TIMESTAMP BY

IoTHub.EnqueuedTime

GROUP BY

IoTHub.ConnectionDeviceId, TumblingWindow(second,10);

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Różnica w temperaturze  
  
SELECT

IoTHub.EnqueuedTime, IoTHub.ConnectionDeviceId as deviceId,

temperature, tempGrowth = temperature - LAG(temperature)

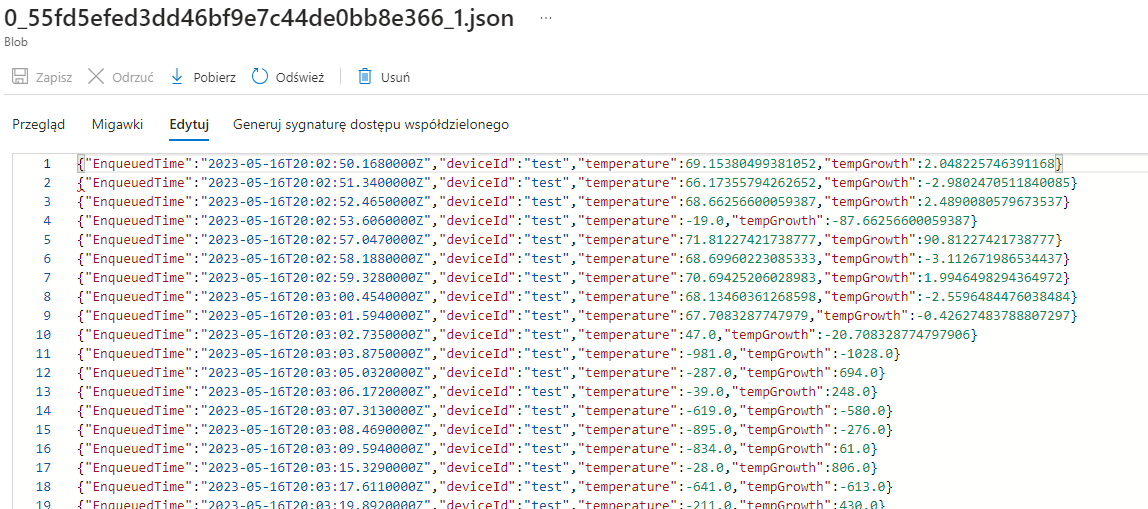
OVER (PARTITION BY deviceId LIMIT DURATION(hour, 1))

INTO [temperature-growth]

FROM iot

TIMESTAMP BY

IoTHub.EnqueuedTime;



Sprawdzanie czasu bez alertów

WITH SelectPreviousEvent AS

(

SELECT temperature,

IoTHub.EnqueuedTime as enqueuedTime,

IoTHub.ConnectionDeviceId as deviceId,

LAG(IoTHub.EnqueuedTime) OVER (LIMIT DURATION(hour,24)) as previousTime,

LAG(temperature) OVER (LIMIT DURATION(hour, 24)) as previousTemperature

FROM iot

TIMESTAMP BY

IoTHub.EnqueuedTime

)

SELECT

EnqueuedTime,

temperature,

previousTemperature,

LAG(enqueuedTime) OVER (LIMIT DURATION(hour,24)

WHEN previousTemperature > 70) as lastAlertOFFtime,

DATEDIFF(second, LAG(enqueuedTime) OVER (LIMIT DURATION(hour,24) WHEN previousTemperature > 70),

enqueuedTime) as timeWithoutAlert

INTO [temperature-noalert-duration]

FROM SelectPreviousEvent

WHERE temperature > 70 AND previousTemperature < 70;

Obraz zawierający tekst, Czcionka, numer, linia

Opis wygenerowany automatycznie