Functioneel Ontwerp

IoT Message Processing Azure Function

1. Inleiding

Dit document beschrijft het functioneel ontwerp van een Azure Function die berichten ontvangt van IoT-apparaten via een Event Hub. De berichten worden verwerkt op basis van hun type (data of vitals) en opgeslagen in InfluxDB voor verdere analyse. In geval van een fout wordt het bericht naar een Service Bus queue gestuurd voor latere herverwerking of diagnose.

---

2. Doelstelling

Het doel van dit systeem is het verwerken van IoT-berichten door deze te splitsen op basis van hun type en ze op te slaan in de juiste InfluxDB buckets. Het systeem moet tevens robuust zijn tegen fouten door foutieve berichten door te sturen naar een Service Bus queue voor later herstel.

---

3. Scope

In Scope

- Verwerken van IoT-berichten via Event Hub consumer group.
- Gebruik van InfluxDB voor opslag van data- en vitals-berichten.
- Foutafhandeling door foutieve berichten naar een Service Bus queue te sturen.
- Logging van fouten en successen voor monitoring.
- Infra Structure as Code voor Azure Function, Managed Identity & Servicebus Queues (ServicebusNamespace is al aanwezig)

Out of Scope

- Geavanceerde validaties van de berichtinhoud.
- Reprocessing en retentiestrategieën voor de Service Bus queue.

4. Systeemcomponenten

4.1 Azure Function

- **Beschrijving**: De Azure Function ontvangt berichten van een Event Hub. De berichten bevatten gegevens van IoT-apparaten en hebben twee typen: data en vitals.
- **Input**: Berichten in JSON-formaat van een Event Hub. In bijlage staan voorbeelden. Uiteraard zitten er nog Metadata elementen in het IoT hub. Deze moeten meegenomen worden. Belangrijk is dat de '"ts-e" & "ts-g" (optioneel)' geintepreteerd worden als Epoch datetime!

- **Output**: Verwerkte berichten worden opgeslagen in InfluxDB. In geval van fouten worden berichten naar de Service Bus queue gestuurd.

4.2 InfluxDB

- **Beschrijving**: Een time-series database die gebruikt wordt voor de opslag van de IoT-berichten. Er zijn twee buckets: één voor data-berichten en één voor vitals-berichten.
- **Functie**: Opslag van IoT-gegevens met timestamp (Epoch) en device ID als tags. Eventele Message Properties worden ook als tags opgeslagen. Reden is dat DeviceId's bijvoorbeeld opgenomen kunnen zijn als 'DeviceGroup' dit moet snel op te vragen zijn. Vandaar de Tagging

4.3 Azure Service Bus

- **Beschrijving**: Een messaging systeem dat wordt gebruikt om foutieve berichten op te vangen voor latere verwerking.
- **Functie**: Opslaan van berichten waarvoor fouten zijn opgetreden tijdens verwerking.

4.4. Azure App Confguration

- **Beschrijving**: De Azure App Configuration wordt gebruikt om configuratie van Azure Funciton, InfluxDb, Servicebus Queue , Topics etc op te slaan. Eventuele Feature Flags worden hier ook in opgeslagen

---

5. Functionele Specificaties

5.1 Ontvangst van berichten

Beschrijving

De Azure Function wordt getriggerd door nieuwe berichten in een Event Hub consumer group.

- **Trigger**: Event Hub Trigger
- **Input**:

    - Berichten in JSON-formaat.
    - Voorbeeld bericht (json) in bijlage
- **Validaties**:
    - Elk bericht moet een deviceId, epochTimestamp(s), value, en messageType bevatten.
    - Het messageType veld moet geldig zijn en de waarde “data” of “vitals” hebben.
    - En de waarden worden omgezet in voor InFluxdb juiste type.

5.2 Verwerking van berichten

Beschrijving

Na ontvangst wordt het bericht verwerkt door een C# ProcessMessageService(class), die het type (data of vitals) bepaalt en het naar de juiste InfluxDB bucket schrijft.

- **Verwerkingsstappen**:
    1. Valideer de berichtinhoud (structuur en velden).
    2. Gebruik de messageType om te bepalen welk bucket in InfluxDB wordt gebruikt.
    3. Sla het bericht op in het juiste bucket met de epochTimestamp als timestamp en de deviceId als tag.
    4. Log het resultaat van de verwerking.
- **Data Mapping**:
    - epochTimestamp → **Timestamp** in InfluxDB.
    - deviceId → **Tag** in InfluxDB.
    - value → **Field** in InfluxDB.
    - Properties => loop and create Tags
- **Opslaglocaties**:
    - **Data bucket**: Voor berichten met messageType gelijk aan “data”.
    - **Vitals bucket**: Voor berichten met messageType gelijk aan “vitals”.

5.3 Foutafhandeling

Beschrijving

Als er tijdens de verwerking een fout optreedt, moet het bericht naar een Azure Service Bus queue worden gestuurd.

- **Trigger**: Elke fout, zoals een netwerkfout of een InfluxDB schrijffout.
- **Service Bus Queue**: sbq-failed-<messagetype>-messages-queue.
- **Berichtinhoud**: Het originele bericht, samen met een foutmelding.
- **Bericht Properties**
    - **IOT Properties (enqueue time, messageid, deviceId, optionele message properties zoals group)**

5.4 Logging

Beschrijving

Het systeem logt zowel succesvolle als mislukte verwerkingen.

- **Logs**:
    - Succesvolle berichten: Bevatten device ID, berichttype, en een bevestiging van opslag.
    - Foutieve berichten: Bevatten device ID, berichttype, en de foutmelding. Uiteraard payload

5.5 Performance Requirements

- De functie moet ten minste 100 berichten per seconde kunnen verwerken.
- De berichtenverwerking moet binnen 2 seconden per bericht plaatsvinden.

5.6 Veiligheid en Authenticatie

- De toegang tot de Azure Service Bus moet beveiligd zijn met **Managed Identity** of andere veilige authenticatiemethodes.
- Toegang tot InfluxDb zal via secure Client zijn. Conform the specs van InfluxData
- Alle berichten moeten gecontroleerd worden op formaat en inhoud om te voorkomen dat ongeldige gegevens verwerkt worden.

6. Niet-functionele specificaties

6.1 Schaalbaarheid

- De Azure Function moet automatisch kunnen opschalen op basis van het aantal binnenkomende berichten.
- In eerste opzet wordt gebruik gemaakt van Azure Function Flex Serviceplan. Dit is feitelijk een Consumptionplan, waarbij schaalbaarheid niet echt inzetbaar is.

6.2 Robuustheid

- Bij netwerkfouten moet de foutafhandelingsprocedure berichten correct naar de Service Bus queue sturen zonder dat gegevens verloren gaan.

6.3 Monitoring en Alerts

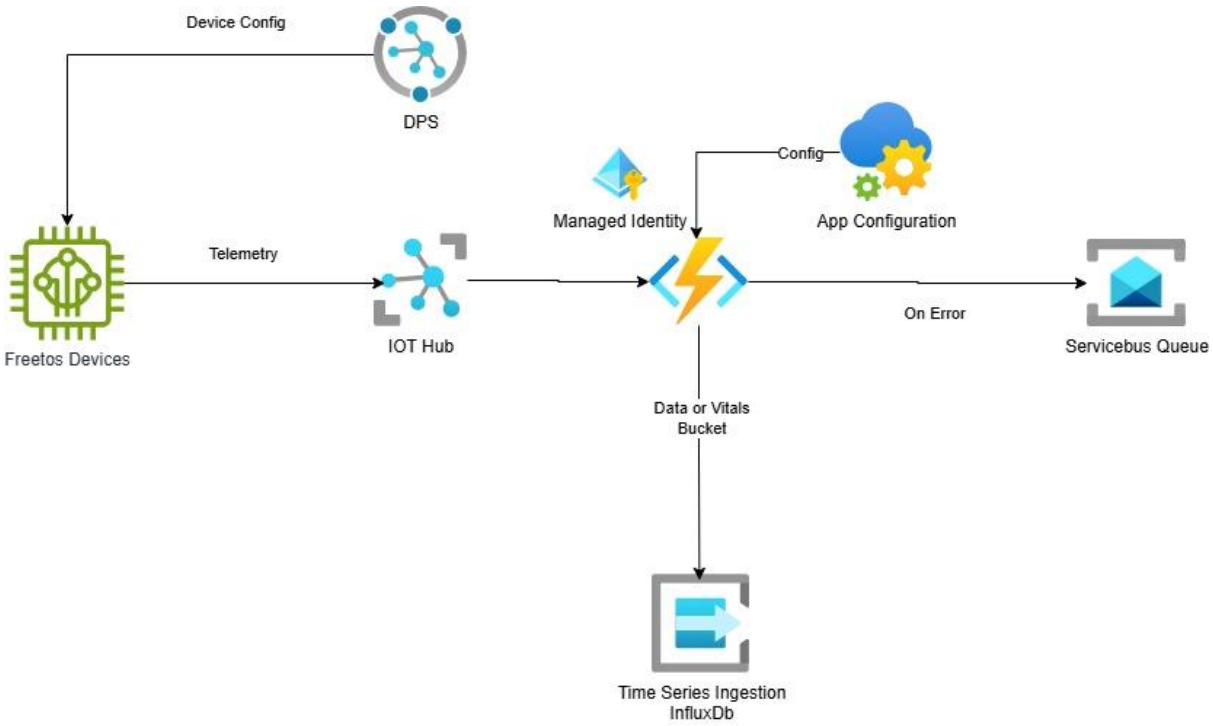
- Het systeem moet worden gemonitord via **Azure Application Insights** voor foutmeldingen en prestaties.
- Bij mislukte berichten moet een alert worden gegenereerd.

6.3 Architectuur

- Azure Function wordt opgebouwd met Dependency Injection voor de ProcessMessageService. InfluxDb client, .
- Alle Services worden met MOQ & XUnit framework voorzien van unitTests
- Unit Tests voor de Services en Dependency Injection (DI)
- Gebruik maken van (AI) Claude Sonnet 3.5 en Cline plugin wordt geadviseerd; Eventuele keys worden verstrekt.
- .NET Framework 8

---

7. Diagram



---

8. Conclusie

Dit ontwerp zou moeten leiden tot een schaalbare en robuuste oplossing voor het verwerken van de IoT-berichten in Azure. Door gebruik te maken van een scheiding van berichttypes, foutafhandeling via de Service Bus, en opslag in InfluxDB, wordt een flexibele architectuur gecreëerd die gemakkelijk kan worden uitgebreid en aangepast.

Bijlagen

Data Message voorbeeld


```
[
  "body": {
    "data": [
      {
        "ts-e": 1724652722,
        "e-id": "45303032363030303031303933373631 34",
        "t1": 28807267,
        "t2": 26194921,
        "t1-i": 2442843,
        "t2-i": 5816525,
        "p": 264,
        "p-i": 114,
        "p1": 152,
        "p2": 109,
        "p3": 0,
        "p1-i": 0,
        "p2-i": 0,
        "p3-i": 114,
        "i1": 1000,
        "i2": 0,
        "i3": 0,
        "ts-g": 1724652000,
        "g": 6406223
      }
    ]
  },
  "enqueuedTime": "Mon Aug 26 2024 08:12:04 GMT+0200 (Central European Summer Time)",
  "properties": {
    "env": "poc"
  }
}
```


Device Id is in Message Metadata

Vitals Message Voorbeeld


```
{
  "body": {
    "vitals": {
      "reboots": 2,
      "pre40": 0,
      "uptime": 244820,
      "ver": "1.4.0.1138",
      "ip": "192.168.178.83",
      "mac": "66:E8:33:60:77:50",
      "refresh": 60,
      "refresh-v": 600,
      "wifi": 0,
      "wifi-rssi": 0,
      "dtr-out": 0,
      "msg": "",
      "rebootreason": 1
    }
  },
  "enqueuedTime": "Mon Aug 26 2024 08:11:47 GMT+0200 (Central European Summer Time)",
  "properties": {
    "env": "poc"
  }
}
```