

Software and Data Engineering Project

Live Streaming of Data Through Edge Gateways Using Azure.

Submitted By:-

ADITYA THITE - M22AI1001

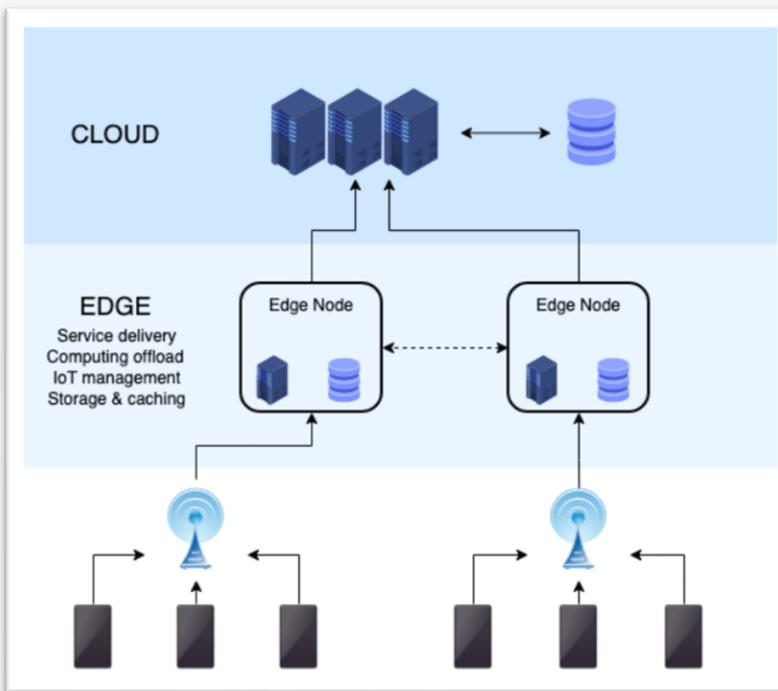
NIKHIL SETHI - M22AI1006



What is Edge Computing

What is edge computing?

Edge computing takes place at or near the physical location of either the user or the source of the data. By placing computing services closer to these locations, users benefit from faster, more reliable services with better user experiences, while companies benefit by being better able to support latency-sensitive applications, identify trends, and offer better products and services.



Example : When transporting petroleum in a large containers in ship, there is a possibility of leakage in the pipes which can cause loss of millions of dollars. This can be avoided prior by utilizing the power of edge computing and edge gateways.

Ref Link: <https://www.redhat.com/en/topics/edge-computing/iot-edge-computing-need-to-work-together#:~:text=An%20IoT%20device%20is%20a,data%20is%20collected%20and%20processed>.

What are Edge Gateways

An Edge Gateway is a device or software that acts as an intermediary between edge devices and the cloud. It collects data from the edge devices, performs some processing, and then forwards the data to the cloud or a centralized data center.

There are several companies that manufacture this edge gateways .Eg. Dell, Red lion , etc.

These devices earlier used to have Modbus TCP protocol .Since the trend in the latest MQTT or AMQP protocol , nowadays , these devices are also able to communicate with this protocol. Now, of these devices , some of these devices are able to directly throw the telemetry messages to other sources like cloud technologies like Azure, AWS, and others.

For the legacy devices which are not able to throw those messages, we have to install some modules to connect them to the respective sources.

Edge Gateway



An example of an edge gateway of 3000 series from Dell

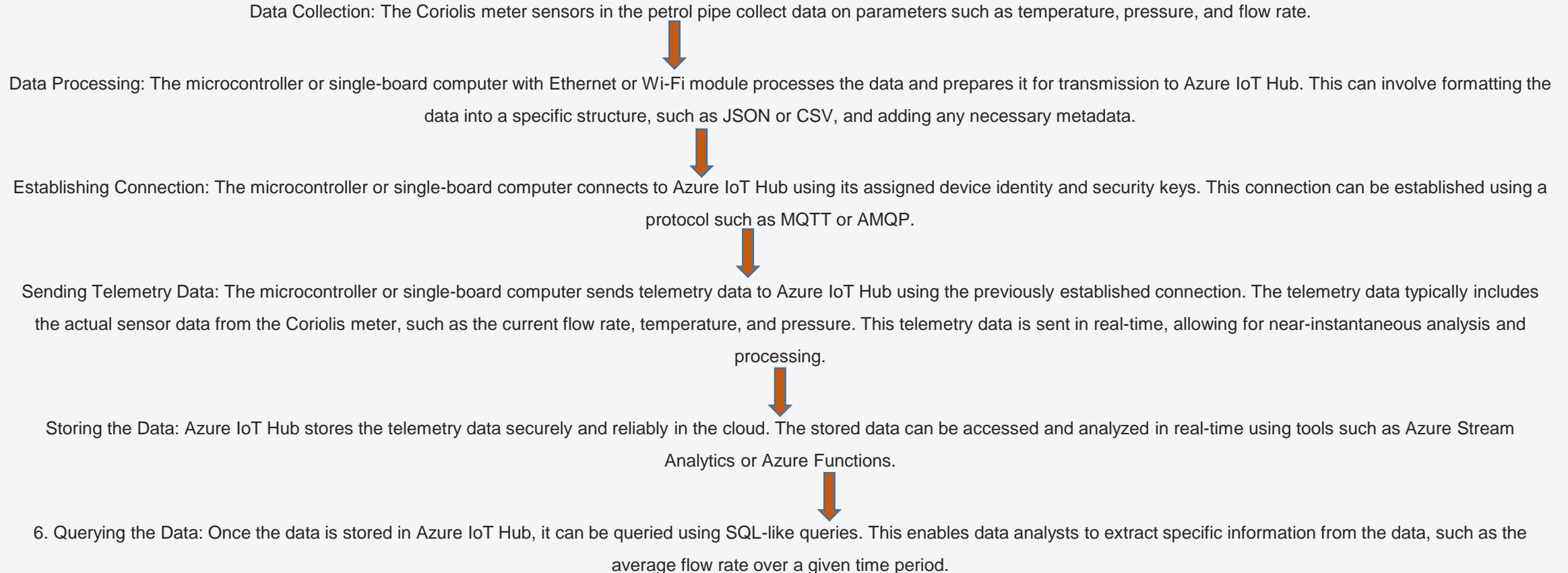
What is Azure IoT Hub?

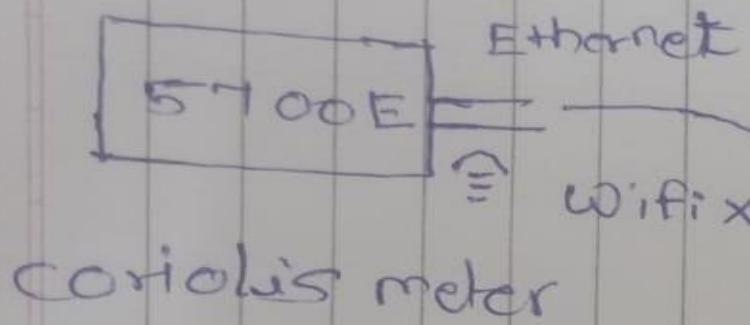
Azure IoT Hub is a cloud-based service provided by Microsoft Azure that enables secure and scalable communication between IoT devices and the cloud. It provides features like device-to-cloud and cloud-to-device messaging, device management, and security for IoT solutions.

Azure IoT Hub helps in implementing edge computing by providing a way to manage and communicate with IoT devices at scale. Azure IoT Hub can be integrated with Azure IoT Edge, which is a service that allows you to deploy and run cloud workloads on edge devices. Azure IoT Edge provides a way to run custom code and Azure services on edge devices, allowing you to perform analytics and processing at the edge.

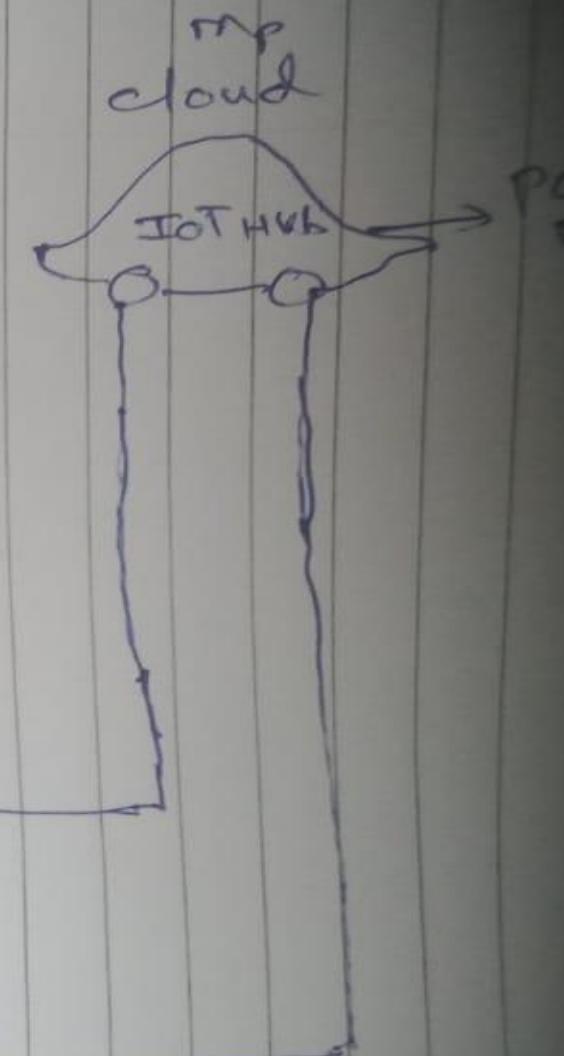
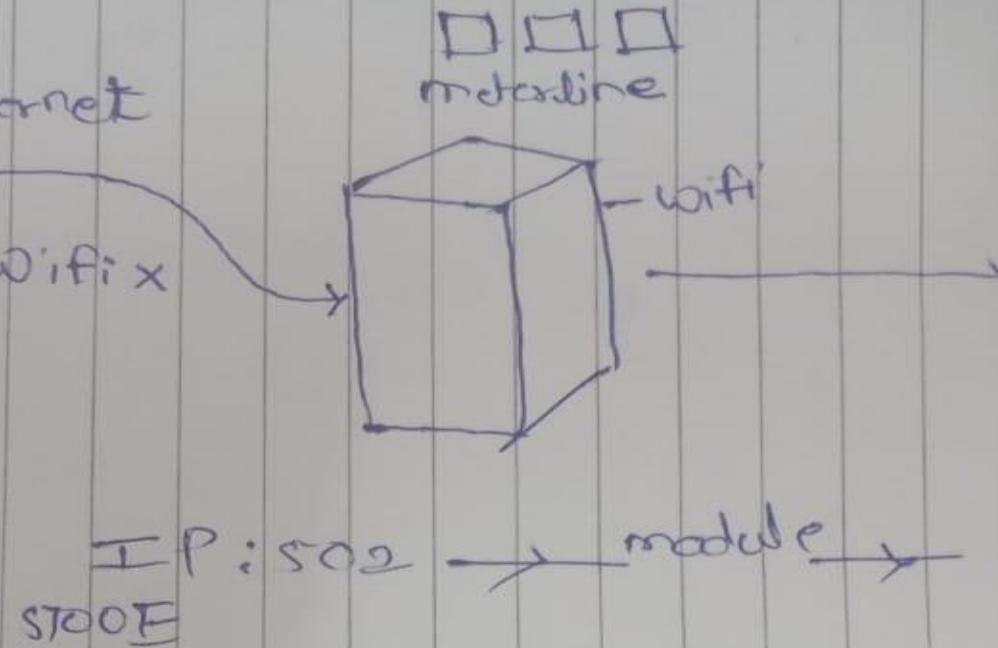
By using Azure IoT Hub with Azure IoT Edge, you can process and analyze data at the edge, reducing the amount of data sent to the cloud, improving response times, and enhancing security by keeping sensitive data on the edge. Edge gateways can be used to connect the edge devices to Azure IoT Hub and Azure IoT Edge, allowing you to manage and communicate with them securely and reliably.

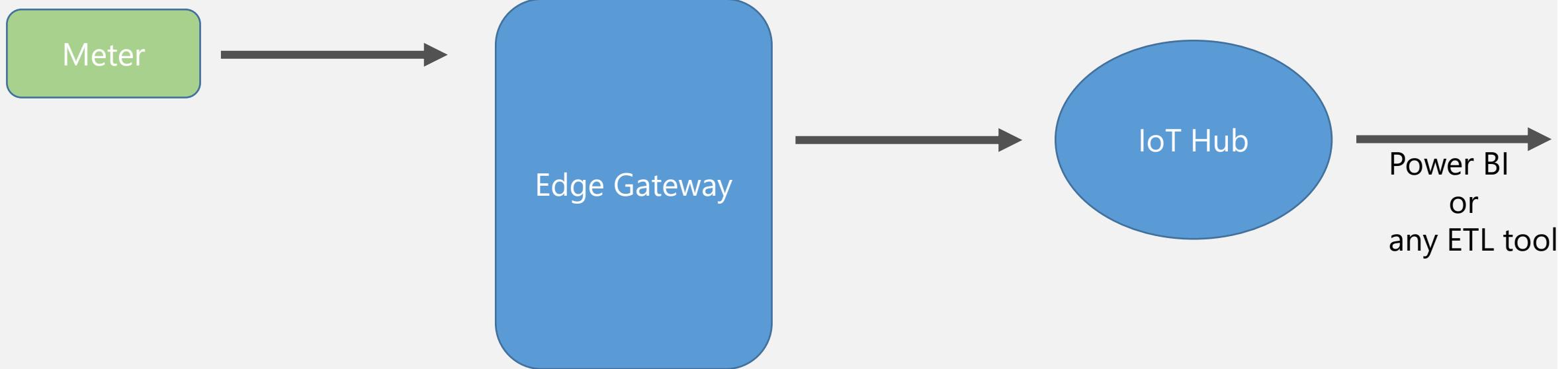
Ref Link: <https://azure.microsoft.com/en-us/products/iot-hub>





Temperature
Pressure
Dif.
MF





Creating the Azure IoT Hub

- Here, we will be showing streaming of data from those edge devices using Azure technologies like Azure IoT Hub which is specifically design for IoT devices, Azure Function app which will be useful for the transformation of the semi structured data that we get from those devices and others like Azure SQL, and others.
- Due to limitations in the subscription for our Azure account(students version), we were not able to fully utilize the services and thus the performance issues were quite significant and we were not able to utilize the full capabilities of the Azure as it is available in paid version .
- However , we can fully understand the capabilities that one can have in Azure and understand the end to end data pipeline that we can built for live streaming of data using the students subscription.

The UI of the Azure looks similar to this

Here, we have created the students free trial account in the Azure

The screenshot shows the Azure portal interface. At the top, there's a navigation bar with the Microsoft logo and a search bar. Below the search bar is a "Azure services" section with various icons and links: "Create a resource" (plus sign), "Function App" (lightning bolt icon), "IoT Hub" (blue square icon), "Azure Databricks" (red cube icon), "Storage accounts" (green bars icon), "Stream Analytics..." (blue gear icon), "Quickstart Center" (orange rocket icon), "Virtual machines" (monitor icon), "App Services" (blue globe icon), and "More services" (right-pointing arrow). Below this is a "Resources" section with tabs for "Recent" (underlined) and "Favorite". It lists recent resources with columns for Name, Type, and Last Viewed.

Name	Type	Last Viewed
⚡ mayank22	Function App	2 days ago
⚡ temperaturesensor1	IoT Hub	2 days ago
⚡ fatest111as	Function App	2 days ago
⚡ sensordata1	Function App	4 days ago
📦 test1	Azure Databricks Service	4 days ago
🌐 iot_device_sensor_streaming	Resource group	5 days ago
💻 sadatabricksstorage	Storage account	2 weeks ago
🌐 DefaultResourceGroup-southindia	Resource group	2 weeks ago

[See all](#)

Setting up an IoT Hub

- Create an Azure account and log in to the Azure portal
- Create a new IoT Hub resource and select the desired settings, such as the pricing tier, resource group, and region
- Once the IoT Hub is created, select it and navigate to the "Shared access policies" section
- Create a new policy with "service connect" and "device connect" permissions

Microsoft Azure

Home > Create a resource

Get Started

Recently created

Categories

- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers
- Databases
- Developer Tools
- DevOps
- Identity
- Integration
- Internet of Things
- IT & Management Tools
- Media
- Migration

iot

All Services (11) Resource Groups (-1) Marketplace (20) Documentation (28) Resources (0)

Azure Active Directory (0)

Services

- IoT Hub
- IoT Central Applications
- Azure IoT Hub Device Provisioning Services
- Windows 10 IoT Core Services

See all

- Device Update for IoT Hubs
- Microsoft Defender for IoT
- Azure Percept Studio
- DICOM service

Marketplace

- IoTConnect Platform
- IoT Edge Metrics Collector
- DeviceTone Suite - Ready to Run IoT
- TCS Clever Energy™

See all

- Cynerio Platform
- IoT-TICKET® for Azure, Build IoT Applications Stunningly Fast
- Keyfactor Command for IoT
- Industrial IOT

Documentation

- Introduction to the Azure Internet of Things (IoT) | Microsoft Learn
- Overview of IoT workloads - Microsoft Azure Well-Architected Framework
- Tutorial - Create a hierarchy of Azure IoT Edge devices (nested edge devices)

See all

Continue searching in Azure Active Directory

Searching all subscriptions.

Create | Docs | MS Learn

Create | Learn more

Give feedback

- Create an IoT Hub - This hub acts as a central hub for all your IoT devices, and it will be the gateway for data exchange between devices and the cloud.

IoT hub

Microsoft

Create an IoT hub to help you connect, monitor, and manage billions of your IoT assets. [Learn more](#)

Project details

Choose the subscription you'll use to manage deployments and costs. Use resource groups like folders to help you organize and manage resources.

Subscription * ⓘ

Azure for Students

Resource group * ⓘ

(New) iot_device_sensor_streaming

[Create new](#)

Instance details

IoT hub name * ⓘ

temperaturesensor1

Region * ⓘ

South India

Tier *

Free

ⓘ Free trial explores the app with live data. Trials cannot scale or be upgraded later.

[Compare tiers](#)

Daily message limit * ⓘ

8,000 (₹0/month)

[Review + create](#)

< Previous

Next: Networking >

Create a new IoT Hub resource and select the desired settings, such as the pricing tier, resource group, and region

Creating the Azure IoT Hub ...

Home > IoT Hub >

IoT hub

Microsoft

Pricing

IoT hub ₹0 INR per month [Change basics](#)

Add-ons total [Change add-ons](#)

Basics

Subscription	Azure for Students
Resource group	iot_device_sensor_streaming
IoT hub name	temperaturesensor1
Region	South India
Disaster recovery enabled	Yes
Tier	Free
Daily message limit	8,000 (₹0/month)

Networking

Connectivity configuration	Public access
Private endpoint connections	None

[Create](#) | [< Previous: Tags](#) | [Next >](#) | Automation options

Deployment is progress

Home >

temperaturesensor1-217132942 | Overview

Deployment

Search

Delete Cancel Redeploy Download Refresh

Overview Inputs Outputs Template

... Deployment is in progress

Deployment name: temperaturesensor1-217132942
Subscription: Azure for Students
Resource group: iot_device_sensor_streaming

Start time: 2/17/2023, 1:29:45 PM
Correlation ID: ed2517b0-df0f-447e-8ac1-5cf7af2c1afe

Deployment details

Resource	Type	Status	Operation details
temperaturesensor1	Microsoft.Devices/IotHubs	Created	Operation details

Give feedback

Tell us about your experience with deployment

... Deployment in progress...

Deployment to resource group 'iot_device_sensor_streaming' is in progress.

 Microsoft Defender for Cloud
Secure your apps and infrastructure
[Go to Microsoft Defender for Cloud >](#)

 Free Microsoft tutorials
Use IoT Hub message routing to send device-to-cloud messages to different endpoints
Understanding the device identity registry
Understanding IoT Hub quotas and throttling

 Work with an expert
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.
[Find an Azure expert >](#)

IoT Hub is created

Now, our IoT Hub is ready here. We can now connect this created hub with the edge devices and can get the required data from there.

The screenshot shows the Azure IoT Hub Overview page for a deployment named "temperaturesensor1-217132942". The deployment status is marked as "complete" with a green checkmark. Deployment details include the name, subscription ("Azure for Students"), and resource group ("iot_device_sensor_streaming"). The start time was 2/17/2023, 1:29:45 PM, and the correlation ID is ed2517b0-df0f-447e-8ac1-5cf7af2c1afe. Below the deployment summary, there are sections for "Deployment details" and "Next steps", each with two recommended actions: "Add and configure IoT Devices" and "Configure routing rules for device messaging". A prominent blue "Go to resource" button is located at the bottom left. At the very bottom, there are links for "Give feedback" and "Tell us about your experience with deployment".

Now, once ready, we can register the device that we have to connect . For this task, it has 2 options as you can see in the image, viz Devices and IoT Edge .

- If some device has direct provision to throw messages to cloud , we can click on the device tab and then create it and then supply the required connection string in the device
- If some device has not the provision to throw messages to Azure, we can use the IoT Edge provided by the Azure and install the IoT Edge runtime on the device

The screenshot shows the 'Devices' blade in the Azure IoT Hub portal. The top navigation bar includes 'Home', 'temperaturesensor1-217132942 | Overview', and 'temperaturesensor1'. The main title is 'temperaturesensor1 | Devices'. On the left, a sidebar lists 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', and 'Events'. Under 'Device management', 'Devices' is selected, followed by 'IoT Edge', 'Configurations + Deployments', 'Updates', and 'Queries'. Under 'Hub settings', there are links for 'Built-in endpoints', 'Message routing', and 'File upload'. The main content area features a search bar, a toolbar with 'Add Device', 'Edit columns', 'Refresh', 'Assign tags', and 'Delete', and a 'Find devices using a query' button. A search bar for 'enter device ID' and a filter button are also present. A table header with columns 'Device ID', 'Type', 'Status', 'Last status update', 'Authentication type', 'C2D messages queued', and 'Tags' is shown, but the message 'There are no devices to display.' is centered below it.

Register a device: After creating the IoT Hub, you need to register a device to it. You can register a device using the Azure portal or using code. In this case, we will register a simulated device that sends temperature readings.

Creating the IoT Edge device

Here, we don't have these devices as these are costly and widely used in the industrial applications . So, thus , for testing we will be considering one Linux virtual machine which will be acting as the edge device for us .

The screenshot shows the Azure IoT Hub interface for managing IoT Edge devices. The top navigation bar includes 'Home', 'temperaturesensor1-217132942 | Overview', and 'temperaturesensor1'. The main title is 'temperaturesensor1 | IoT Edge' under 'IoT Hub'. A sidebar on the left lists 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', and 'Events'. Under 'Device management', 'Devices' is selected, while 'IoT Edge' is highlighted. Other options include 'Configurations + Deployments', 'Updates', 'Queries', 'Hub settings', 'Built-in endpoints', and 'Message routing'. The main content area has a search bar and a message: 'Now, view and manage all devices from the new [Devices](#) page.' Below this, tabs for 'IoT Edge Devices' and 'IoT Edge Deployments' are shown. A sub-section titled 'Deploy Azure services and solution-specific code to on-premises devices. Use IoT Edge devices to perform compute and analytics tasks on data before it's sent to the cloud.' includes a 'Learn more' link. A 'Device name' input field contains 'enter device ID' and a 'Find devices' button. Below this are buttons for '+ Add IoT Edge Device', 'Refresh', 'Assign tags', and 'Delete'. A table header with columns 'Device ID', 'Runtime Response', 'Module Count', 'Connected Client Count', and 'Deployment Count' is present, followed by the message 'There are no IoT Edge devices to display.'

Now, before jumping into device , first lets create the device in the Azure IoT Hub for connecting our device

Create a device

X

 Find Certified for Azure IoT devices in the Device Catalog

Device ID * ⓘ

sensor1



IoT Edge Device

Authentication type ⓘ

Symmetric key X.509 Self-Signed

Auto-generate keys ⓘ



Connect this device to an IoT hub ⓘ

Enable Disable

Parent device ⓘ

No parent device

[Set a parent device](#)

Child devices ⓘ

0

[Choose child devices](#)

The device is ready...

Here, as you can see , we have created the device name sensor1. The device is ready now.

The screenshot shows the Azure IoT Edge Devices page for the hub "temperaturesensor1". The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events, Device management (Devices selected), IoT Edge, Configurations + Deployments, Updates, Queries, Hub settings, Built-in endpoints, Message routing, and File upload. The main content area has a search bar and a message: "Now, view and manage all devices from the new [Devices](#) page." Below this, tabs for "IoT Edge Devices" (selected) and "IoT Edge Deployments" are shown. A sub-header says: "Deploy Azure services and solution-specific code to on-premises devices. Use IoT Edge devices to perform compute and analytics tasks on data before it's sent to the cloud. [Learn more](#)". A "Device name" input field contains "sensor1" and a "Find devices" button. Below are buttons for "+ Add IoT Edge Device", "Refresh", "Assign tags", and "Delete". A table lists the device "sensor1" with columns: Device ID (checkbox), Runtime Response (NA), Module Count (--), Connected Client Count (--), and Deployment Count (--).

<input type="checkbox"/> Device ID	Runtime Response	Module Count	Connected Client Count	Deployment Count
<input type="checkbox"/> sensor1	NA	--	--	--

When the device is created , we will get 2 modules created automatically with it , Edge Agent and Edge Hub . Each of them has its significance and is basically used to run the edge runtime on the device . This can be seen as below image. Also, we can note down the primary connection string from here , which , later we will be using it to connect the device to the Azure IoT Hub.

The screenshot shows the Azure IoT Edge Device Overview page for a device named 'sensor1'. The top navigation bar includes 'Home', 'temperaturesensor1-217132942 | Overview', 'temperaturesensor1 | IoT Edge', and a back arrow. Below the navigation is a toolbar with 'Save', 'Set modules', 'Manage child devices', 'Troubleshoot', 'Device twin', and 'Refresh' buttons. The main content area displays device identity information: 'Device ID: sensor1', 'Primary key: [REDACTED]', 'Secondary key: [REDACTED]', 'Primary connection string: [REDACTED]', 'Secondary connection string: [REDACTED]', 'IoT Edge runtime response: NA', and 'Tags: No tags'. There is also a 'Enable connection to IoT Hub' toggle switch set to 'Enable'. Under 'Parent device', it says 'No parent device'. At the bottom, there are three tabs: 'Modules' (selected), 'IoT Edge hub connections', and 'Deployments and Configurations'. The 'Modules' table lists two entries:

Name	Type	Specified in Deployment	Reported by Device	Runtime Status	Exit Code
\$edgeAgent	Module Identity	NA	NA	NA	NA
\$edgeHub	Module Identity	NA	NA	NA	NA

Reference link :

<https://learn.microsoft.com/en-us/azure/iot-edge/iot-edge-runtime?view=iotedge-1.4>

sensor1

temperaturesensor1

Save Set modules Manage child devices Troubleshoot Device twin Refresh

Device ID	sensor1	
Primary key	
Secondary key	
Primary connection string	
Secondary connection string	
IoT Edge runtime response	NA	
Tags (edit)	No tags	
Enable connection to IoT Hub	<input checked="" type="radio"/> Enable <input type="radio"/> Disable	
Parent device	No parent device	

Modules [IoT Edge hub connections](#) [Deployments and Configurations](#)

Name	Type	Specified in Deployment	Reported by Device	Runtime Status	Exit Code
------	------	-------------------------	--------------------	----------------	-----------

Setting up Temperature Sensor Module

Now, as we are not having any meter to connect to the edge gateway, we will be installing a dummy data module which is provided by the Azure which is called the Temperature Sensor Module. This module throws emits the dummy data as it is from real sensor. Azure has specifically created the module for testing purpose.

For this , we will have to click on the Set Modules tab which can be seen in the picture and then from the Market place section , we can directly instal it .

Note: Some legacy edge gateways doesn't have the capability to communicate in the MQTT/ AMQP protocol. For that, Azure has provided the readymade module name Modbus which after implementing on the device, readily converts Modbus to MQTT.

Going to Market place from Set Modules to search for Temperature Sensor

Home > temperaturesensor1-217132942 | Overview > temperaturesensor1 | IoT Edge > sensor1 >

Set modules on device: sensor1 ...

temperaturesensor1

Modules Routes Review + create

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
<input type="text" value="Name"/>	<input type="text" value="Address"/>	<input type="text" value="User name"/>	<input type="password" value="Password"/>

IoT Edge Modules

IoT Edge modules are Docker containers deployed to IoT Edge devices. They can communicate with other modules or send data to the IoT Edge runtime. Modules on devices count toward IoT Hub quota limits based on tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.

+ Add Runtime Settings

	DESIRED STATUS
+ IoT Edge Module	
+ Marketplace Module	
+ Azure Stream Analytics Module	

We use cookies to improve our products and services. Read our [privacy statement](#) to learn more. See [details](#) of what data is collected.

[Review + create](#) < Previous Next: Routes >

IoT Edge Module Marketplace

...

Tile view ▾

Marketplace

All

Categories

Internet of Things (73)

Analytics (25)

Compute (12)

Developer Tools (10)

IT & Management Tools (10)

Networking (10)

AI + Machine Learning (7)

Security (5)

Containers (4)

Databases (4)

Showing 1 to 20 of 73 results in Internet of Things with 1 selected filters. [Clear filters](#)



Azure Blob Storage on IoT Edge

Microsoft

IoT Edge Modules

Azure consistent block blob storage on IoT Edge



Simulated Temperature Sensor

Microsoft

IoT Edge Modules

IoT Edge module that simulates a temperature sensor.



Azure SQL Edge

Microsoft

IoT Edge Modules

Azure SQL Edge module for IoT Edge



OPC Publisher

Microsoft

IoT Edge Modules

Connects to OPC UA Servers and publishes OPC UA node data values in OPC UA Pub/Sub compatible format



IoT Edge Metrics Collector

Microsoft

IoT Edge Modules

Collect Prometheus metrics from IoT Edge modules and transport them to Azure Monitor or IoT Hub



YOLOv3 gRPC Extension for AVA Edge

motojin.com, Inc.



RedisEdge

Redis Labs



Modbus

Microsoft



Node-RED on IoT Edge

motojin.com, Inc.



Azure Security Center for IoT

Microsoft

Installed the temperature sensor Module for getting the dummy data for testing

Home > temperaturesensor1-217132942 | Overview > temperaturesensor1 | IoT Edge > sensor1 >

Set modules on device: sensor1 ...

temperaturesensor1

Modules Routes Review + create

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
<input type="text" value="Name"/>	<input type="text" value="Address"/>	<input type="text" value="User name"/>	<input type="password" value="Password"/>

IoT Edge Modules

IoT Edge modules are Docker containers deployed to IoT Edge devices. They can communicate with other modules or send data to the IoT Edge runtime. Modules on devices count toward IoT Hub quota limits based on tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.

[+ Add](#) [Runtime Settings](#)

NAME	DESIRED STATUS	
SimulatedTemperatureSensor	running	

Send usage data to Microsoft to help improve our products and services. Read our [privacy statement](#) to learn more. See [details](#) of what data is collected.

[Review + create](#)

< Previous

Next: Routes >

Set modules on device: sensor1

temperaturesensor1

Modules Routes Review + create

Validation passed.

Deployment

The text box below displays the deployment to be submitted.

```
1 {  
2     "modulesContent": {  
3         "$edgeAgent": {  
4             "properties.desired": {  
5                 "schemaVersion": "1.1",  
6                 "runtime": {  
7                     "type": "docker",  
8                     "settings": {}  
9                 },  
10                "systemModules": {  
11                    "edgeAgent": {  
12                        "settings": {  
13                            "image": "mcr.microsoft.com/azureiotedge-agent:1.4"  
14                        },  
15                        "type": "docker"  
16                    },  
17                    "edgeHub": {  
18                        "restartPolicy": "always",  
19                        "settings": {  
20                            "image": "mcr.microsoft.com/azureruniotedge-hub:1.4"  
21                        }  
22                    }  
23                }  
24            }  
25        }  
26    }  
27}
```



Create

< Previous

Next >

IoT Hub is Ready

Our IoT Hub is ready with the required modules in it that we will be needing to connect with our device. Now, let's connect these IoT hub with the Linux virtual machine that we will be using as a edge gateway.

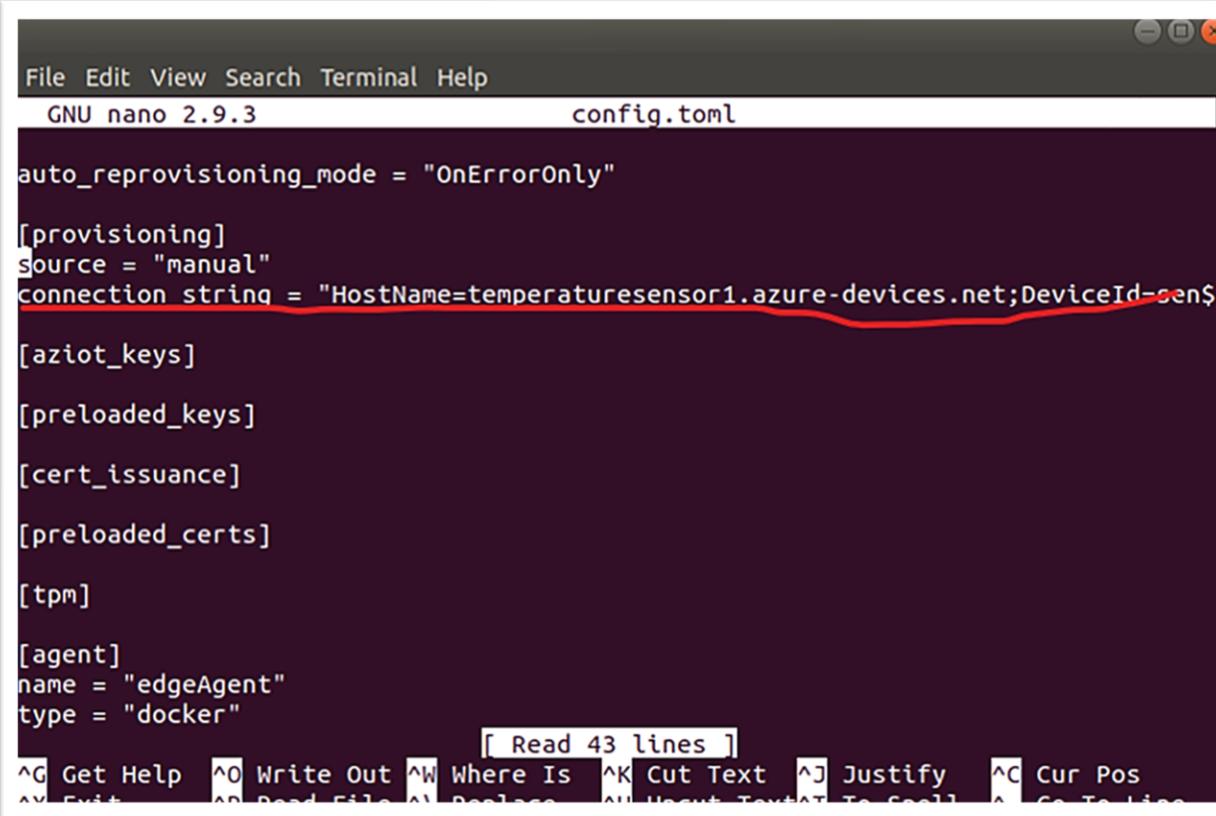
I have created the virtual machine with the OS as Ubuntu 18.04. As this machine is not capable of throwing messages directly, we will need to install the Azure IoT Edge runtime on the machine. The steps according to the OS is provided by Microsoft for the installation.

I have followed those steps and run commands accordingly on my Linux system for the installation of IoT Edge runtime.

The reference link : <https://learn.microsoft.com/en-us/azure/iot-edge/how-to-provision-single-device-linux-symmetric?view=iotedge-1.4&tabs=azure-portal%2Cubuntu>

The screenshot shows a Microsoft Learn article page. At the top, there is a navigation bar with links to 'Learn', 'Azure', 'Internet of Things', and 'IoT Edge'. Below the navigation bar, the title 'Create and provision an IoT Edge device on Linux using symmetric keys' is displayed. The article was published on 03/02/2023 and has a reading time of 19 minutes. There are 8 contributors and a 'Feedback' link. A sidebar on the left contains a 'Version' dropdown set to 'IoT Edge 1.4 (LTS)', a 'Filter by title' input field, and a list of IoT Edge topics. The main content area contains sections for 'In this article' (Prerequisites, Register your device, View registered devices and retrieve provisioning information, Install IoT Edge, Show 5 more) and 'Applies to' (IoT Edge 1.4). The text explains the end-to-end instructions for registering and provisioning a Linux IoT Edge device, mentioning the device ID used for tracking communications. It also lists the components included in the device ID: IoT hub hostname, Device ID, and Authentication details to connect to IoT Hub. A 'Download PDF' button is located at the bottom of the sidebar.

After installation of azure iotedge run time , the config file will be formed by the names config.toml in the installation path and thus, for connecting our IoT Hub with the device, we have to give our connection string that we have noted earlier in the configuration file as shown below



```
File Edit View Search Terminal Help
GNU nano 2.9.3           config.toml

auto_reprovisioning_mode = "OnErrorOnly"

[provisioning]
source = "manual"
connection string = "HostName=temperaturesensor1.azure-devices.net;DeviceId=sen$"

[aziot_keys]

[preloaded_keys]

[cert_issuance]

[preloaded_certs]

[tpm]

[agent]
name = "edgeAgent"
type = "docker"

[Read 43 lines]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit  ^D Read File  ^L Replace  ^U Undo  ^T To Spell  ^F Go To Line
```

We can save the config file now .

Now , lets check if the iot edge runtime is installed successfully on our machine.

We can check this by using the command - **sudo iotedge check**

This command will run and check if everything is in the right place as shown . Looks like the connectivity to the Azure iothub is successful.

```
nlck@ubuntu18:~$ sudo iotedge check
[sudo] password for rohan:

Configuration checks (aziot-identity-service)
-----
/ keyd configuration is well-formed - OK
/ certd configuration is well-formed - OK
/ tpmd configuration is well-formed - OK
/ identityd configuration is well-formed - OK
! daemon configurations up-to-date with config.toml - Warning
    /etc/aziot/config.toml was modified after keyd's config
        You must run 'aziotctl config apply' to update keyd's config with the latest config.toml
/ identityd config toml file specifies a valid hostname - OK
/ aziot-identity-service package is up-to-date - OK
/ host time is close to reference time - OK
/ preloaded certificates are valid - OK
/ keyd is running - OK
/ certd is running - OK
/ identityd is running - OK
/ read all preloaded certificates from the Certificates Service - OK
/ read all preloaded key pairs from the Keys Service - OK
/ check all EST server URLs utilize HTTPS - OK
/ ensure all preloaded certificates match preloaded private keys with the same ID - OK

Connectivity checks (aziot-identity-service)
-----
/ host can connect to and perform TLS handshake with iothub AMQP port - OK
/ host can connect to and perform TLS handshake with iothub HTTPS / WebSockets port - OK
/ host can connect to and perform TLS handshake with iothub MQTT port - OK

Configuration checks
-----
/ aziot-edged configuration is well-formed - OK
! configuration up-to-date with config.toml - Warning
    /etc/aziot/config.toml was modified after edged's config
        You must run 'iotedge config apply' to update edged's config with the latest config.toml
/ container engine is installed and functional - OK
/ configuration has correct URIs for daemon mgmt endpoint - OK
```

We can see our respective module here as shown through the commands as shown in the figure and also can see the logs for the temperature sensor that we are getting.

```
nick@Ubuntu18:~$ sudo iotedge list
NAME          STATUS      DESCRIPTION      Config
edgeAgent     running     Up 37 minutes   mcr.microsoft.com/azureiotedge-agent:1.4
edgeHub       running     Up 36 minutes   mcr.microsoft.com/azureiotedge-hub:1.4
nick@Ubuntu18:~$ sudo iotedge list
NAME          STATUS      DESCRIPTION      Config
SimulatedTemperatureSensor  running     Up 16 seconds, 180 ms, 24 µs and 35 ns  mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:latest
edgeAgent     running     Up 39 minutes   mcr.microsoft.com/azureiotedge-agent:1.4
edgeHub       running     Up 39 minutes   mcr.microsoft.com/azureiotedge-hub:1.4
nick@Ubuntu18:~$ sudo iotedge logs SimulatedTemperatureSensor
[2023-03-08 14:09:38 : Starting Module
SimulatedTemperatureSensor Main() started.
Initializing simulated temperature sensor to send 500 messages, at an interval of 5 seconds.
To change this, set the environment variable MessageCount to the number of messages that should be sent (set it to -1 to send unlimited messages).
[Information]: Trying to initialize module client using transport type [Amqp_Tcp_Only].
[Information]: Successfully initialized module client of transport type [Amqp_Tcp_Only].
    03/08/2023 14:09:43> Sending message: 1, Body: [{"machine": {"temperature": 21.312498598349954, "pressure": 1.035601106141134}, "ambient": {"temperature": 20.8050586643632, "humidity": 62.3}, "time": "2023-03-08T14:09:43.8293016Z"}]
    03/08/2023 14:09:48> Sending message: 2, Body: [{"machine": {"temperature": 22.1834917629455, "pressure": 1.1348281755254368}, "ambient": {"temperature": 21.30293656967531, "humidity": 63.1}, "time": "2023-03-08T14:09:48.8888367Z"}]
    03/08/2023 14:09:53> Sending message: 3, Body: [{"machine": {"temperature": 22.398419625944733, "pressure": 1.1593136282721848}, "ambient": {"temperature": 20.81420284999966, "humidity": 63.9}, "time": "2023-03-08T14:09:53.8936249Z"}]
    03/08/2023 14:09:58> Sending message: 4, Body: [{"machine": {"temperature": 22.91277812667391, "pressure": 1.2179114321527238}, "ambient": {"temperature": 21.47329322636711, "humidity": 64.7}, "time": "2023-03-08T14:09:58.896127Z"}]
    03/08/2023 14:10:03> Sending message: 5, Body: [{"machine": {"temperature": 22.813684218442653, "pressure": 1.2066222527339732}, "ambient": {"temperature": 21.47998159623865, "humidity": 65.5}, "time": "2023-03-08T14:10:03.89901Z"}]
    03/08/2023 14:10:08> Sending message: 6, Body: [{"machine": {"temperature": 24.039492751684733, "pressure": 1.3462713261412986}, "ambient": {"temperature": 20.996193947356442, "humidity": 66.3}, "time": "2023-03-08T14:10:08.9050542Z"}]
    03/08/2023 14:10:13> Sending message: 7, Body: [{"machine": {"temperature": 24.9902028641099, "pressure": 1.4545800731264444}, "ambient": {"temperature": 20.549690215617687, "humidity": 67.1}, "time": "2023-03-08T14:10:13.9090069Z"}]
    03/08/2023 14:10:18> Sending message: 8, Body: [{"machine": {"temperature": 24.740913615378158, "pressure": 1.4261800321316889}, "ambient": {"temperature": 20.585224433304944, "humidity": 67.9}, "time": "2023-03-08T14:10:18.9118905Z"}]
```

Creating a function to the IoT Hub

As we can see here , the data is in semi structured format and thus we will have to do some processing over the data. For Serverless computation , Azure provides the service know as Function App where we can write the code in Python, Java, JavaScript (NodeJS). Here, there are different triggers available for the function meaning when we have to run that function . There is specific trigger also available for IoTHub on which the function will run whenever the data will come to the Hub.

Create function

Select development environment
Instructions will vary based on your development environment. [Learn more](#)

Development environ... [Develop in portal](#)

Select a template
Use a template to create a function. Triggers describe the type of events that invoke your functions. [Learn more](#)

Filter

Azure Service Bus Queue trigger	A function that will be run whenever a message is enqueued in a specified Service Bus queue
Azure Service Bus Topic trigger	A function that will be run whenever a message is added to the specified Service Bus topic
Azure Blob Storage trigger	A function that will be run whenever a blob is added to a specified container
Azure Event Hub trigger	A function that will be run whenever an event hub receives a new event
Azure Cosmos DB trigger	A function that will be run whenever documents change in a document collection
Azure Event Grid trigger	A function that will be run whenever an event grid receives a new event
Durable Functions HTTP starter	A function that will trigger whenever it receives an HTTP request to execute an orchestrator function.
Durable Functions activity	A function that will be run whenever an Activity is called by an orchestrator function.
Durable Functions entity	Durable Functions entity that stores state.

{fx} m22ai1001 | Functions

Function App

 Search[Create](#)[Delete](#) Filter by name... Name ↑↓

Trigger ↑↓

 EventHubTrigger1

EventHub

[Overview](#)[Activity log](#)[Access control \(IAM\)](#)[Tags](#)[Diagnose and solve problems](#)[Microsoft Defender for Cloud](#)[Events \(preview\)](#)

Functions

[Functions](#)[App keys](#)[App files](#)[Proxies](#)

Deployment

[Deployment slots](#)[Deployment Center](#)

Create function

[Durable Functions HTTP starter](#)

A function that will trigger whenever it receives an HTTP request to execute an orchestrator function

[Durable Functions activity](#)

A function that will be run whenever an Activity is called by an orchestrator function.

[Durable Functions entity](#)

Durable Functions entity that stores state.

[Durable Functions orchestrator](#)

An orchestrator function that invokes activity functions in a sequence.

[Kafka output](#)

A function that will send messages to a specified Kafka topic

[Kafka trigger](#)

A function that will be run whenever a message is added to a specified Kafka topic

[RabbitMQ trigger](#)

A function that will be run whenever a message is added to a specified RabbitMQ queue

Template details

We need more information to create the Azure Event Hub trigger function. [Learn more](#)

New Function* EventHubTrigger2**Event Hub connection*** ⓘ temperaturesensor1_events_IOTHUB[New](#)**Event Hub name*** ⓘ samples-workitems**Event Hub consumer group** ⓘ \$Default

Code

Here , inside the function , we can code as we want and fetch the results from any kind of unstructured data.

Home > EventHubTrigger1

EventHubTrigger1 | Code + Test

Function

Search Save Discard Refresh Test/Run Test integration Upload

Overview Adding third party dependencies in the Azure portal is currently not supported for Linux Consumption Function Apps. Click here to setup local environment. [Learn more](#)

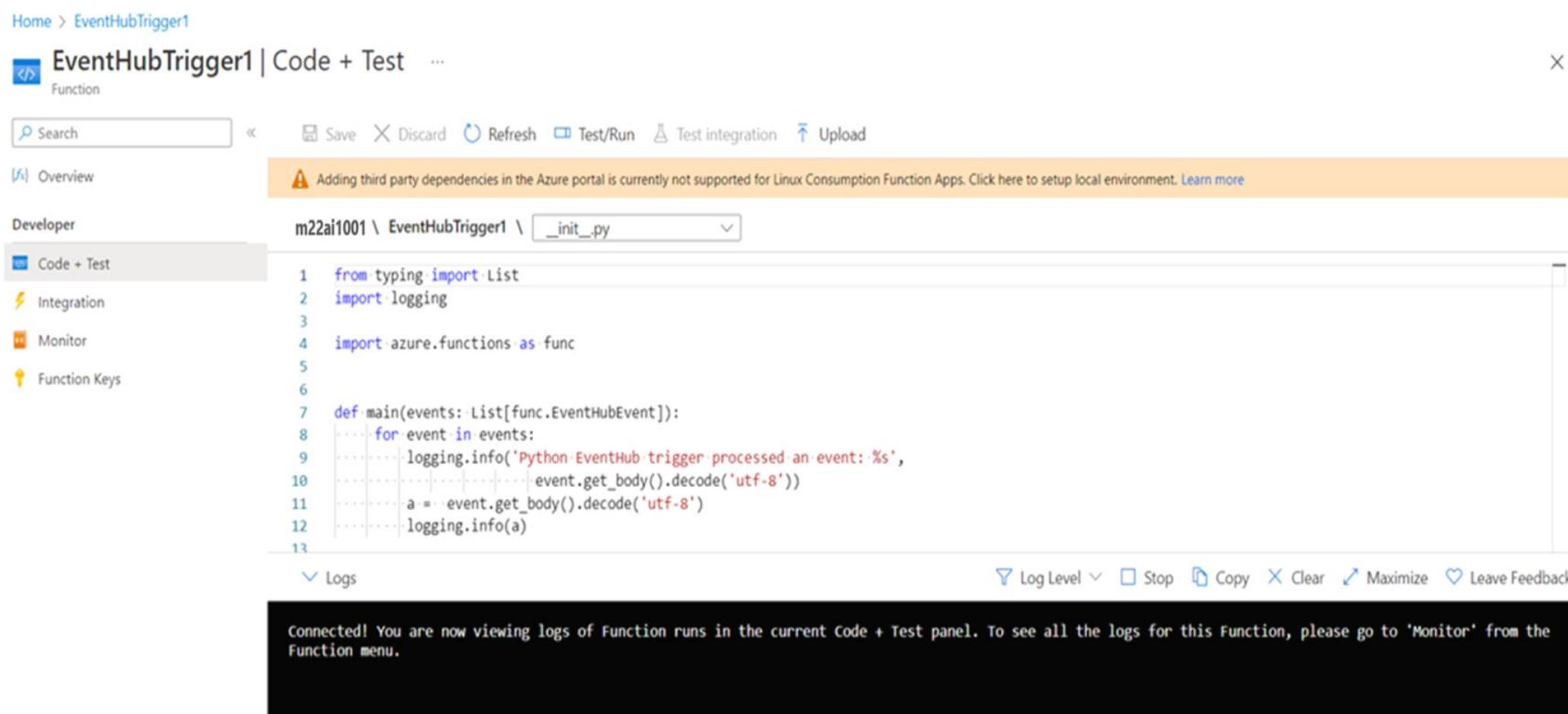
Developer m22ai1001 \ EventHubTrigger1 \ _init_.py

Code + Test Integration Monitor Function Keys

```
1  from typing import List
2  import logging
3
4  import azure.functions as func
5
6
7  def main(events: List[func.EventHubEvent]):
8      for event in events:
9          logging.info('Python EventHub trigger processed an event: %s',
10                      event.get_body().decode('utf-8'))
11          a = event.get_body().decode('utf-8')
12          logging.info(a)
```

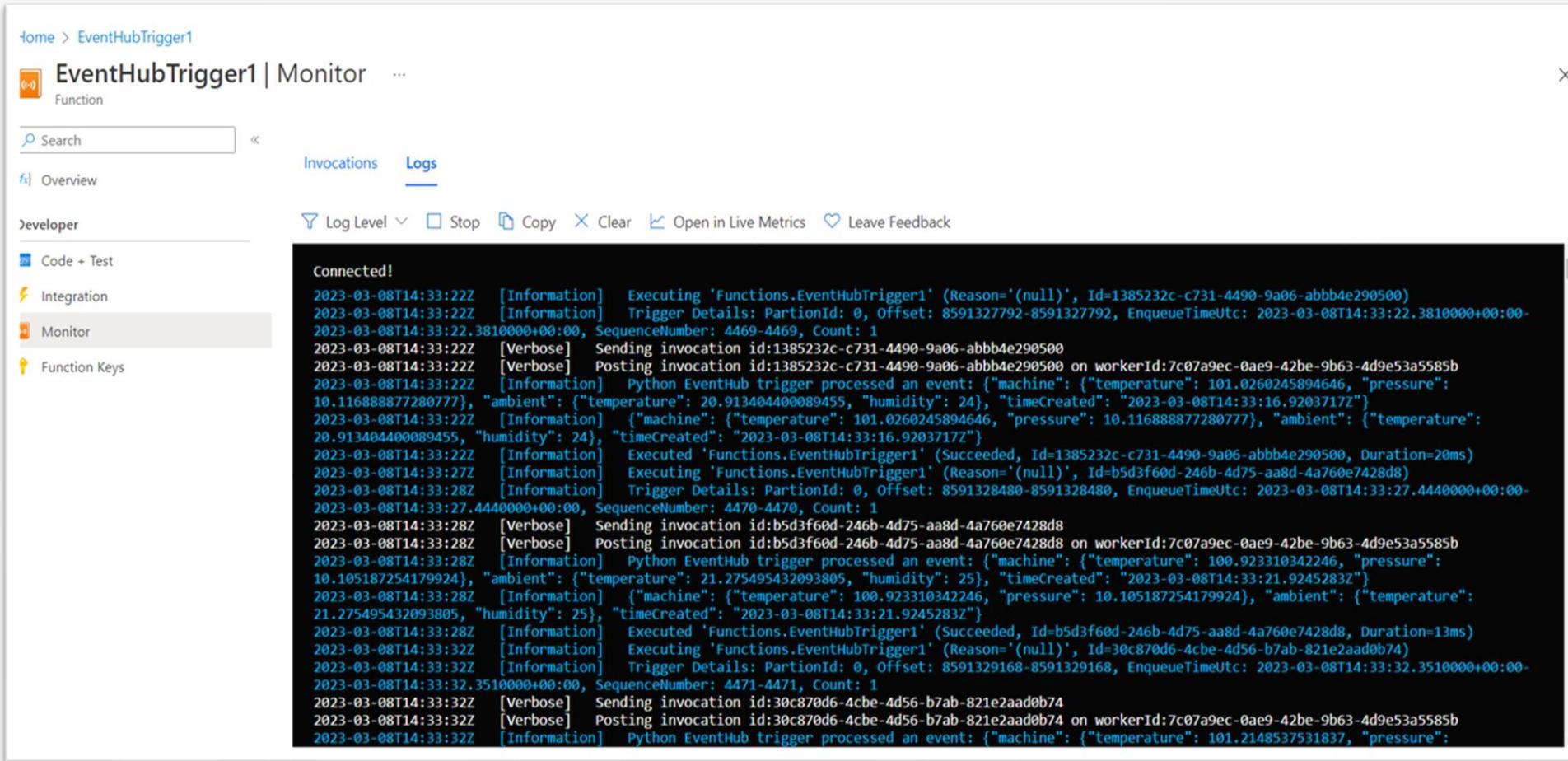
Logs Log Level Stop Copy Clear Maximize Leave Feedback

Connected! You are now viewing logs of Function runs in the current Code + Test panel. To see all the logs for this Function, please go to 'Monitor' from the Function menu.



Data coming ...

Here, as you can see, we are getting the data from the temperature sensor per 5 seconds ,which is a live stream. We can write code and fetch the results as needed per message coming and thus can append that results to various targets like dbms , powerbi , and others.

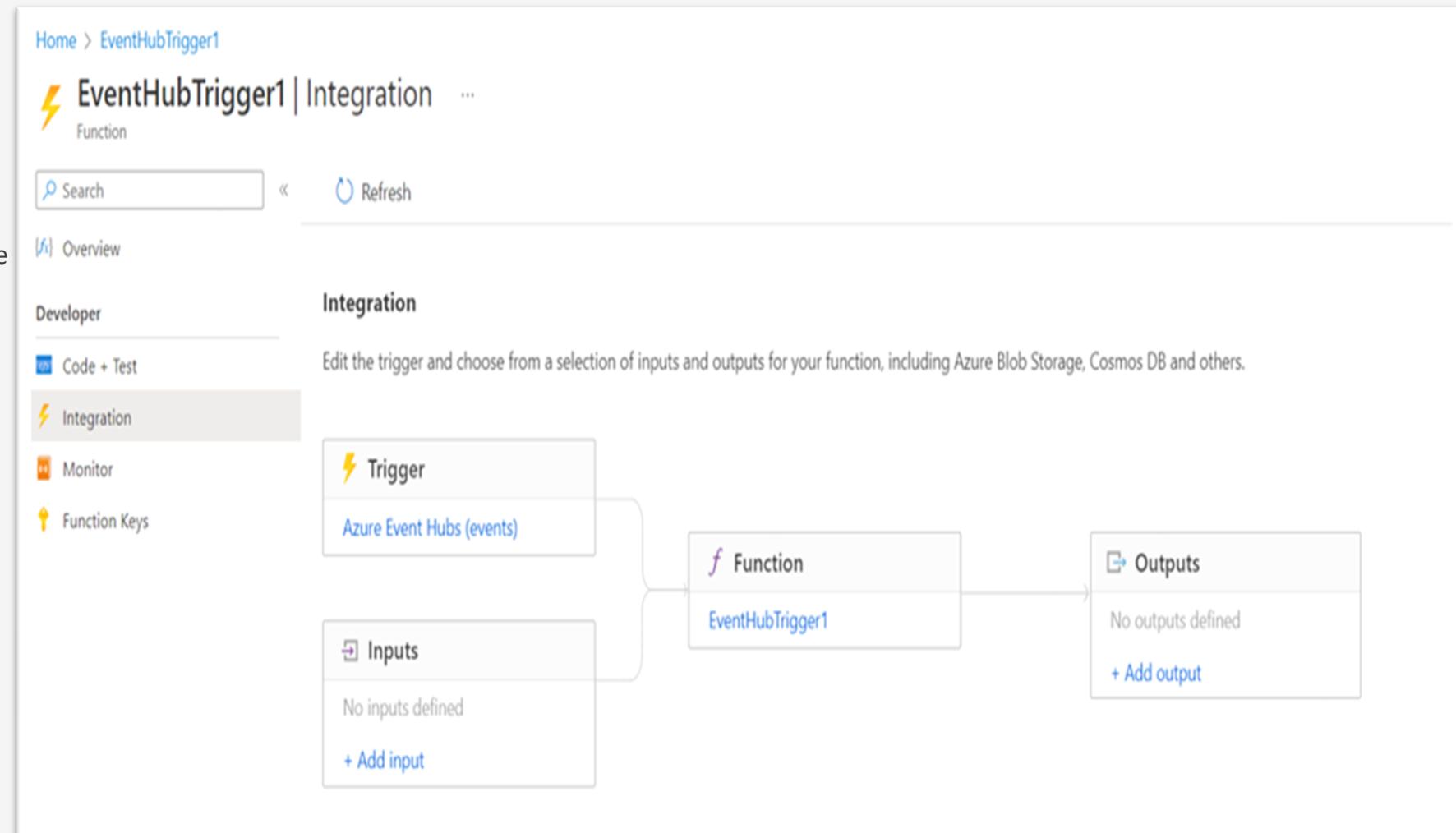


The screenshot shows the Azure Functions monitor interface for the 'EventHubTrigger1' function. The left sidebar has tabs for 'Code + Test', 'Integration', 'Monitor' (which is selected), and 'Function Keys'. The main area has tabs for 'Invocations' (selected) and 'Logs'. Below these are buttons for 'Log Level', 'Stop', 'Copy', 'Clear', 'Open in Live Metrics', and 'Leave Feedback'. The log output window displays the following text:

```
Connected!
2023-03-08T14:33:22Z [Information] Executing 'Functions.EventHubTrigger1' (Reason='(null)', Id=1385232c-c731-4490-9a06-abbb4e290500)
2023-03-08T14:33:22Z [Information] Trigger Details: PartitionId: 0, Offset: 8591327792-8591327792, EnqueueTimeUtc: 2023-03-08T14:33:22.3810000+00:00-
2023-03-08T14:33:22.3810000+00:00, SequenceNumber: 4469-4469, Count: 1
2023-03-08T14:33:22Z [Verbose] Sending invocation id:1385232c-c731-4490-9a06-abbb4e290500 on workerId:7c07a9ec-0ae9-42be-9b63-4d9e53a5585b
2023-03-08T14:33:22Z [Verbose] Posting invocation id:1385232c-c731-4490-9a06-abbb4e290500 on workerId:7c07a9ec-0ae9-42be-9b63-4d9e53a5585b
2023-03-08T14:33:22Z [Information] Python EventHub trigger processed an event: {"machine": {"temperature": 101.0260245894646, "pressure": 10.11688877280777}, "ambient": {"temperature": 20.913404400089455, "humidity": 24}, "timeCreated": "2023-03-08T14:33:16.9203717Z"}
2023-03-08T14:33:22Z [Information] {"machine": {"temperature": 101.0260245894646, "pressure": 10.11688877280777}, "ambient": {"temperature": 20.913404400089455, "humidity": 24}, "timeCreated": "2023-03-08T14:33:16.9203717Z"}
2023-03-08T14:33:22Z [Information] Executed 'Functions.EventHubTrigger1' (Succeeded, Id=1385232c-c731-4490-9a06-abbb4e290500, Duration=20ms)
2023-03-08T14:33:22Z [Information] Executing 'Functions.EventHubTrigger1' (Reason='(null)', Id=b5d3f60d-246b-4d75-aa8d-4a760e7428d8)
2023-03-08T14:33:22Z [Information] Trigger Details: PartitionId: 0, Offset: 8591328480-8591328480, EnqueueTimeUtc: 2023-03-08T14:33:27.4440000+00:00-
2023-03-08T14:33:27.4440000+00:00, SequenceNumber: 4470-4470, Count: 1
2023-03-08T14:33:28Z [Verbose] Sending invocation id:b5d3f60d-246b-4d75-aa8d-4a760e7428d8
2023-03-08T14:33:28Z [Verbose] Posting invocation id:b5d3f60d-246b-4d75-aa8d-4a760e7428d8 on workerId:7c07a9ec-0ae9-42be-9b63-4d9e53a5585b
2023-03-08T14:33:28Z [Information] Python EventHub trigger processed an event: {"machine": {"temperature": 100.923310342246, "pressure": 10.105187254179924}, "ambient": {"temperature": 21.275495432093805, "humidity": 25}, "timeCreated": "2023-03-08T14:33:21.9245283Z"}
2023-03-08T14:33:28Z [Information] {"machine": {"temperature": 100.923310342246, "pressure": 10.105187254179924}, "ambient": {"temperature": 21.275495432093805, "humidity": 25}, "timeCreated": "2023-03-08T14:33:21.9245283Z"}
2023-03-08T14:33:28Z [Information] Executed 'Functions.EventHubTrigger1' (Succeeded, Id=b5d3f60d-246b-4d75-aa8d-4a760e7428d8, Duration=13ms)
2023-03-08T14:33:32Z [Information] Executing 'Functions.EventHubTrigger1' (Reason='(null)', Id=30c870d6-4cbe-4d56-b7ab-821e2aad0b74)
2023-03-08T14:33:32Z [Information] Trigger Details: PartitionId: 0, Offset: 8591329168-8591329168, EnqueueTimeUtc: 2023-03-08T14:33:32.3510000+00:00-
2023-03-08T14:33:32.3510000+00:00, SequenceNumber: 4471-4471, Count: 1
2023-03-08T14:33:32Z [Verbose] Sending invocation id:30c870d6-4cbe-4d56-b7ab-821e2aad0b74
2023-03-08T14:33:32Z [Verbose] Posting invocation id:30c870d6-4cbe-4d56-b7ab-821e2aad0b74 on workerId:7c07a9ec-0ae9-42be-9b63-4d9e53a5585b
2023-03-08T14:33:32Z [Information] Python EventHub trigger processed an event: {"machine": {"temperature": 101.2148537531837, "pressure": 10.105187254179924}, "ambient": {"temperature": 21.275495432093805, "humidity": 25}, "timeCreated": "2023-03-08T14:33:32.3510000+00:00"}
```

Integrating with Other Sources

We can integrate this transformed results to various targets such as Azure SQL , PowerBI, data bricks where we can write the machine learning code, analyse or visualise the data from the Integration tab in the Function as shown.



EventHubTrigger1 | Integration

Function



<

[Overview](#)

Developer

[Code + Test](#)[Integration](#)[Monitor](#)[Function Keys](#)

Integration

Edit the trigger and choose from a selection of inputs and outputs for your function, including Azure Blob Storage, Cosmos DB and others.



THANK
YOU!
