

How to create a C# console application, with .NET 8, to send/read messages to/from your Azure EventHub

1. Create an Azure Event Hub

We input the data for creating a new **Azure EventHub**:

Microsoft Azure Search resources, services, and docs (G+/I)

Home > Event Hubs >

Create Namespace

Event Hubs

Basics Advanced Networking Tags Review + create

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * Azure subscription 1

Resource group * (New) resourcegroup1
[Create new](#)

Instance Details

Enter required settings for this namespace, including a price tier and configuring the number of units (capacity).

Namespace name * resourcegroupnamespace1 ✓
servicebus.windows.net

Location * France Central
The region selected supports Availability zones. Your namespace will have Availability Zones enabled. [Learn more.](#)

Pricing tier * Basic (~\$11 USD per TU per Month)
[Browse the available plans and their features](#)

Throughput Units * 1

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












We select the **Basic** tier for optimizing the cost:

Choose your pricing tier



Browse the available plans and their features

★ Recommended | View all

Basic		Standard 	
1	Consumer group	20	Consumer groups
100	Brokered connections	1000	Brokered connections
	Ingress events \$0.028 per million		Ingress events \$0.028 per million
	Message retention 1 day		Message retention Up to 7 days
			Schema Registry
			Capture \$73/month
11.16 USD/MONTH/TU (ESTIMATED)		22.32 USD/MONTH/TU (ESTIMATED)	
Premium			
100	Consumer groups		
10 K	Brokered connections		
	Ingress events Included		
	Message retention Up to 90 days		
	Schema Registry		
	Capture Included		

Select

We leave the other options with the default values:

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the 'Microsoft Azure' logo and a search bar. The left sidebar contains navigation links: 'Create a resource', 'Home', 'Dashboard', 'All services', 'FAVORITES', 'All resources', 'Resource groups', 'Quickstart Center', 'App Services', and 'Function App'. The main content area displays the 'Create Namespace' page for Event Hubs. The 'Advanced' tab is selected, showing the 'Security' section. The 'Minimum TLS version' is set to 'Version 1.2', with a note indicating that increasing the version will prevent connections using a lower TLS version. The 'Local Authentication' is set to 'Enabled'.

The screenshot shows the Microsoft Azure portal interface, similar to the previous one, but with the 'Networking' tab selected. The 'Network connectivity' section is visible, stating that the namespace can be connected either publicly via public IP addresses or service endpoints, or privately using a private endpoint. A note indicates that private access is only available on Premium or Standard namespaces. The 'Connectivity method' is set to 'Public access'.

Then we press the "Create" button:

Microsoft Azure

Search resources, services, and docs (G+)

Create a resource

Home

Dashboard

All services

FAVORITES

All resources

Resource groups

Quickstart Center

App Services

Function App

SQL databases

Azure Cosmos DB

Virtual machines

Load balancers

Storage accounts

Virtual networks

Microsoft Entra ID

Monitor

Advisor

Microsoft Defender for Cloud

Cost Management + Billing

Home > Event Hubs >

Create Namespace

Event Hubs

Validation succeeded.

BasicsAdvancedNetworkingTagsReview + create

Event Hubs Namespace by Microsoft

Basics

Namespace nameresourcegroupnamespace1

SubscriptionAzure subscription 1

Resource groupresourcegroup1

LocationFrance Central

Pricing tierBasic

Throughput Units1

Availability Zones (Zone Redundancy)Enabled

Networking

Connectivity methodPublic access

Security

Minimum TLS version1.2

Local AuthenticationEnabled

Create< PreviousNext >

Microsoft Azure

Search resources, services, and docs (G+)

Home > resourcegroupnamespace1 | Overview

Deployment

Search

Delete Cancel Redeploy Download Refresh

Overview

Inputs

Outputs

Template

✓ Your deployment is complete

Deployment name : resourcegroupnamespace1

Subscription : Azure subscription 1

Resource group : resourcegroup1

Start time : 12/7/2023, 12:29:22 PM

Correlation ID : da436d84-44d4-420e-aa69-dbd857316c8d

> Deployment details

✓ Next steps

Go to resource

Microsoft Azure

Search resources, services, and docs (G+)

Home > resourcegroupnamespace1 | Overview

resourcegroupnamespace1

Event Hubs Namespace

Search

+ Event Hub Delete Refresh Give feedback

✓ You can start generating test data with the new Azure Event Hubs Data Generator. Click on this message to try the feature!

JSON View

Essentials

Resource group (move) : resourcegroup1

Status : Active

Location : France Central

Subscription (move) : Azure subscription 1

Subscription ID : 846901e6-da09-45c8-98ca-7cca2353ff0e

Host name : resourcegroupnamespace1.servicebus.windows.net

Created : Thursday, December 7, 2023 at 12:29:25 GMT+1

Updated : Thursday, December 7, 2023 at 12:29:47 GMT+1

Zone Redundancy : Enabled

Pricing tier : Basic

Throughput Units : 1 unit

Auto-inflate throughput ... : Not Supported

Local Authentication : Enabled

Tags (edit) : Add tags

NAMESPACE CONTENTS

0 EVENT HUBS

KAFKA SURFACE NOT SUPPORTED

ZONE REDUNDANCY ENABLED

Show data for the last: 1 hour 6 hours 12 hours 1 day 7 days 30 days

Requests	Messages	Throughput
100	100	100B
90	90	90B
80	80	80B
70	70	70B
60	60	60B
50	50	50B
40	40	40B

Now we create an Event Hub

Microsoft Azure

Search resources, services, and docs (G+/I)

Create a resource

Home

Dashboard

All services

FAVORITES

All resources

Resource groups

Quickstart Center

App Services

Function App

SQL databases

Azure Cosmos DB

Virtual machines

Load balancers

Storage accounts

Virtual networks

Microsoft Entra ID

Monitor

Advisor

Microsoft Defender for Cloud

Cost Management + Billing

Home > resourcegroupnamespace1 >

Create Event Hub

Event Hubs

Basics | Capture | Review + create

Event Hub Details

Enter required settings for this event hub, including partition count and message retention.

Name *

myEventHub

Partition count

1

Retention

Configure retention settings for this Event Hub. [Learn more](#)

Cleanup policy

Delete

Retention time (hrs) *

1

min. 1 hour, max. 24 hours (1day)

Review + create

< Previous

Next: Capture >

Microsoft Azure

Search resources, services, and docs (G+)

Create a resource

Home

Dashboard

All services

FAVORITES

All resources

Resource groups

Quickstart Center

App Services

Function App

SQL databases

Azure Cosmos DB

Virtual machines

Load balancers

Storage accounts

Virtual networks

Microsoft Entra ID

Monitor

Advisor

Microsoft Defender for Cloud

Cost Management + Billing

Home > resourcegroupnamespace1 >

Create Event Hub

Event Hubs

Validation succeeded.

BasicsCaptureReview + create

Event Hubs Instance by Microsoft

Basics

NamemyEventHub

Partition count1

Retention

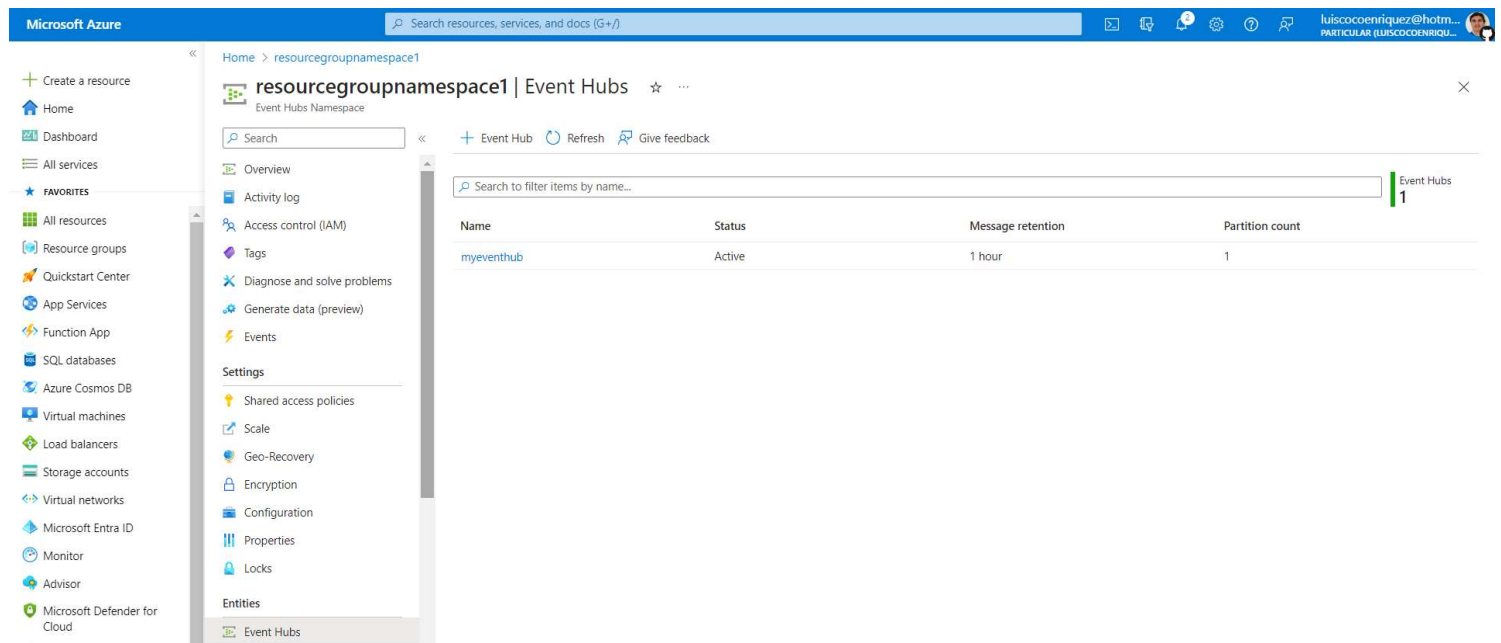
Cleanup policyDelete

Retention time (hrs)1

Capture

Capture StatusNot Supported

Create< PreviousNext >



2. Create a C# console application (with .NET 8) in Visual Studio 2022 Community Edition

Visual Studio 2022

Open recent

Search:

Today

- AzureEventHub_SendEvents.sln** 12/7/2023 12:46 PM
C:\Azure EventHub\AzureEventHub_SendEvents
- Extensions.sln** 12/7/2023 10:49 AM
C:\Repos\CCRef-Extensions\product\Extensions
- ChatGPTSolution.sln** 12/7/2023 10:39 AM
D:\ChatGPTSolutionWithImagesAndVoice\ChatGPTSolutionWithImagesA...
- ProtocolGateway.sln** 12/7/2023 9:27 AM
C:\Repos\ProtocolGateway

This week

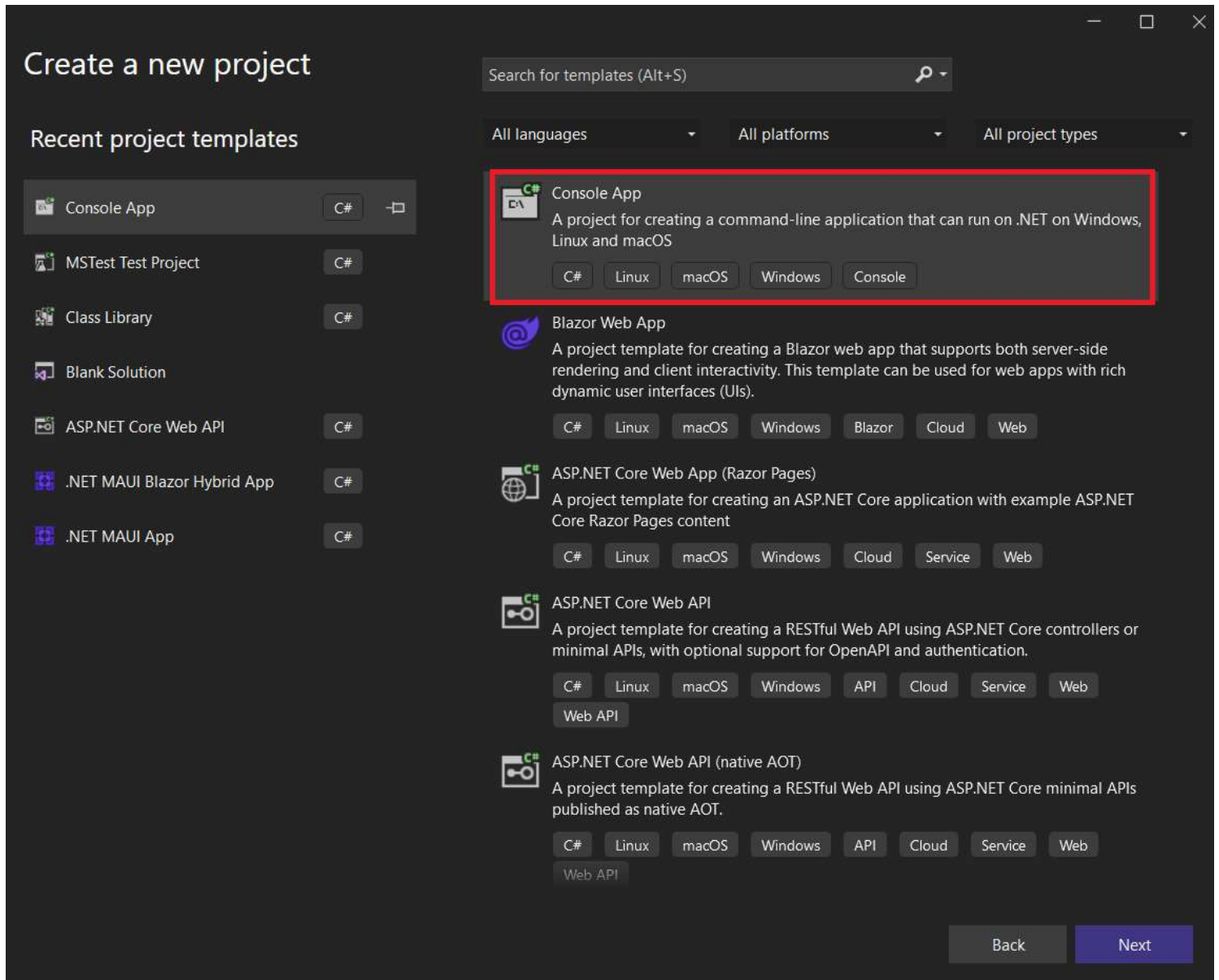
- CosmosGettingStarted.sln** 12/5/2023 6:54 PM
C:\Users\LEnriquez\3D Objects\Downloads\sql-dotnet
- AzureCosmosDBCreateDatabase.sln** 12/5/2023 6:09 PM
C:\Azure SDK for .NET CosmosDB\AzureCosmosDBCreateDatabase
- AzureSDKCosmosDBCreate.sln** 12/5/2023 3:52 PM
C:\Azure SDK for .NET CosmosDB\AzureSDKCosmosDBCreate

This month

- AvNext CompatibilityMatrix.sln** 11/29/2023 4:29 PM

Get started

- Clone a repository**
Get code from an online repository like GitHub or Azure DevOps
- Open a project or solution**
Open a local Visual Studio project or .sln file
- Open a local folder**
Navigate and edit code within any folder
- Create a new project**
Choose a project template with code scaffolding to get started
[Continue without code →](#)



— □ ×

Configure your new project

Console App C# Linux macOS Windows Console

Project name

AzureEventHub_SendEvents

Location

C:\Azure EventHub

Solution name ⓘ

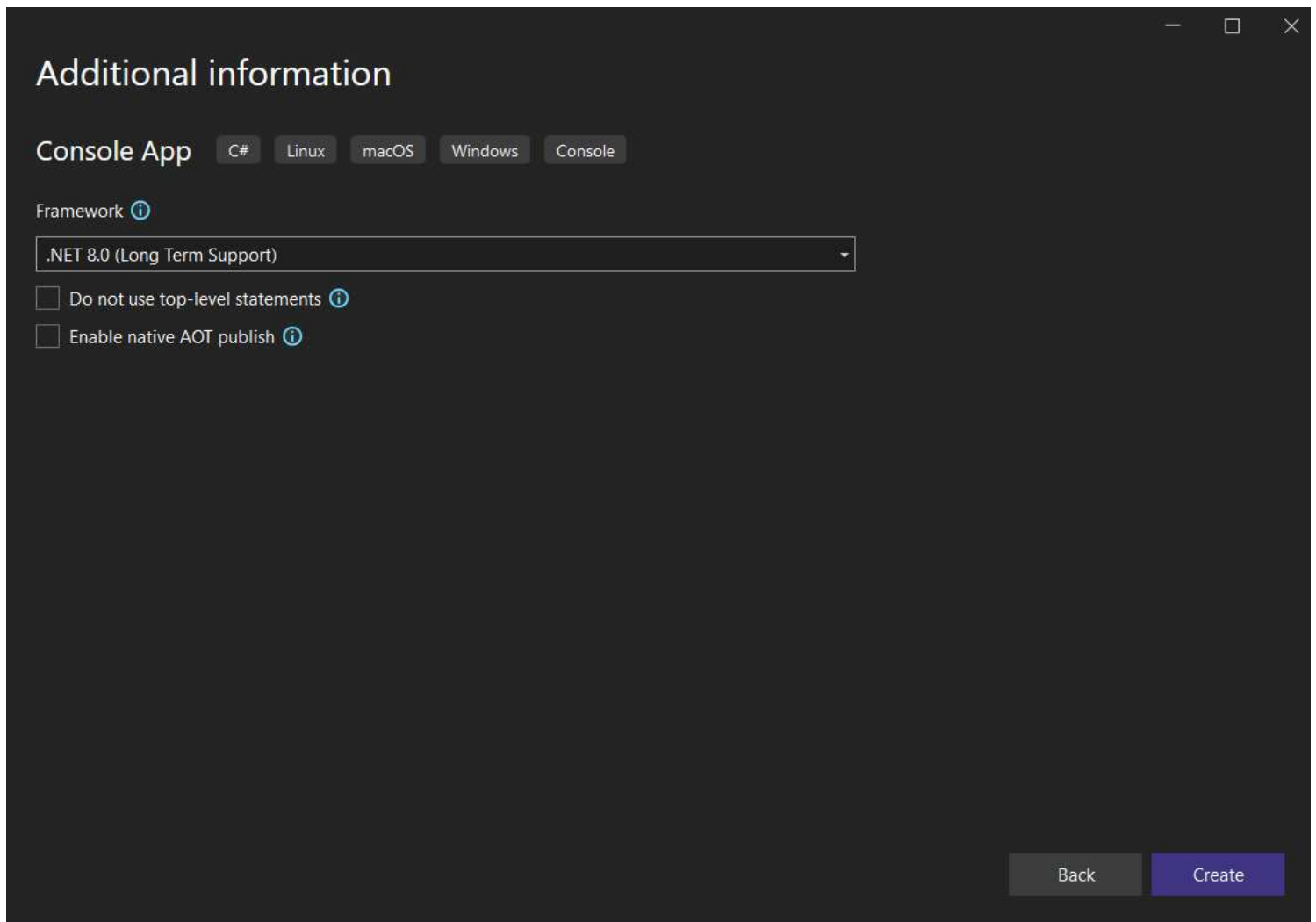
AzureEventHub_SendEvents

☒ Place solution and project in the same directory

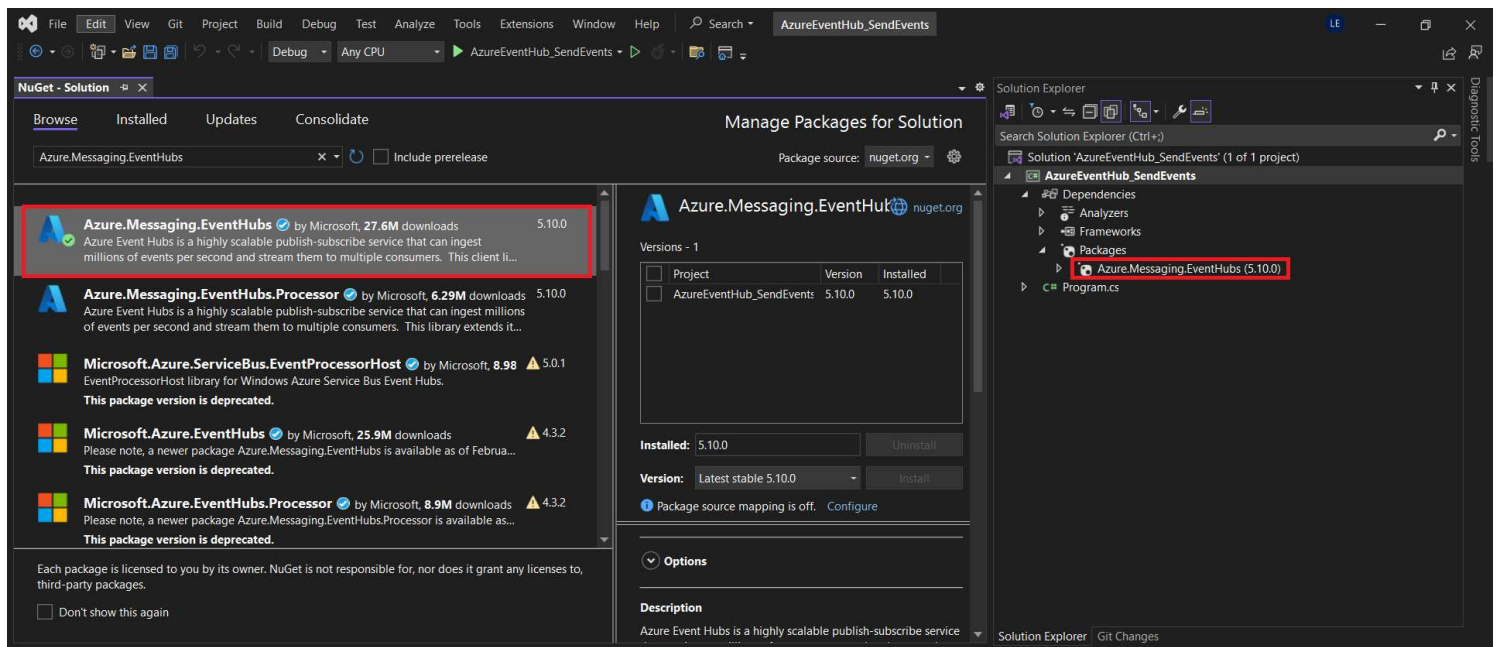
Project will be created in "C:\Azure EventHub\AzureEventHub_SendEvents\"

⚠ This directory is not empty.

Back Next

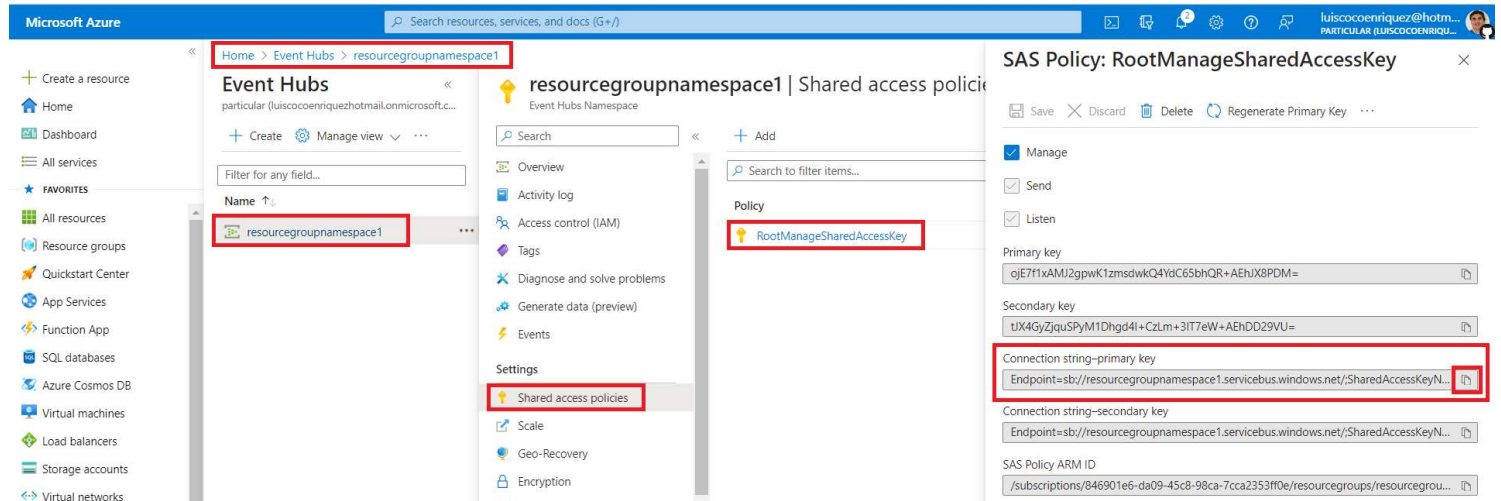


Load the Azure EventHub library with Nuget:

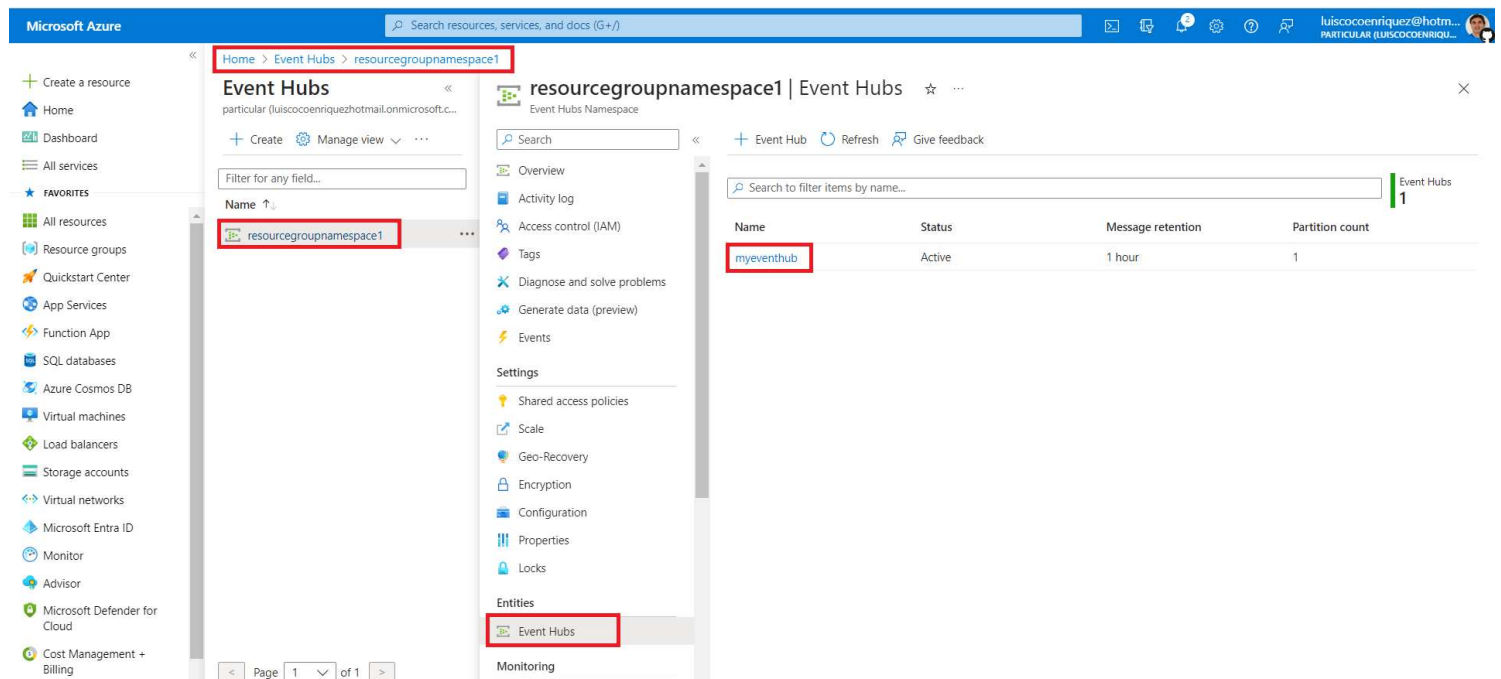


3. Copy the EventHub Connection String and EventHub Name from Azure Portal

We first copy the EventHub connection string:



Then we copy the EventHub name:



4. Input the application source code for "Sending" and "Receiving" messages to/from the Azure Event Hub

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading.Tasks;
using Azure.Messaging.EventHubs;
using Azure.Messaging.EventHubs.Consumer;
using Azure.Messaging.EventHubs.Producer;

string EventHubConnectionString = "Endpoint=sb://resourcegroupnamespace1.servicebus.windows.net/;
string EventHubName = "myeventhub";

string ConsumerGroup = "$Default";

await SendEventsAsync(5); // Specify the number of events you want to send
await ReceiveEventsAsync();

async Task SendEventsAsync(int numberOfEvents)
{
    await using (var producerClient = new EventHubProducerClient(EventHubConnectionString, EventH
    {
        List<EventData> eventBatch = new List<EventData>();

        for (int i = 0; i < numberOfEvents; i++)
        {
            try
            {
                string messageBody = $"Message {i}";
                var eventData = new EventData(Encoding.UTF8.GetBytes(messageBody));
                eventBatch.Add(eventData);
                Console.WriteLine($"Event added to batch: {messageBody}");
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error creating event: {ex.Message}");
            }
        }

        try
        {
            await producerClient.SendAsync(eventBatch);
            Console.WriteLine($"Batch of events sent successfully");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error sending batch of events: {ex.Message}");
        }
    }
}
```

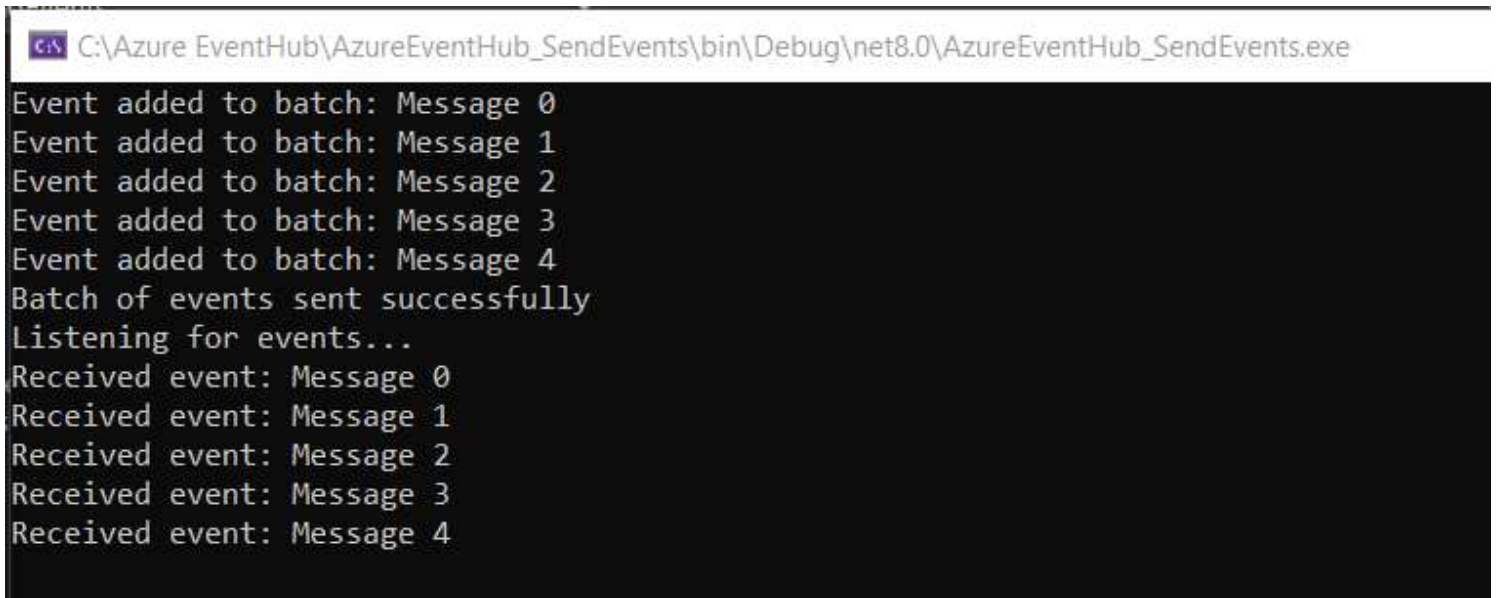


```
async Task ReceiveEventsAsync()
{
    await using (var consumerClient = new EventHubConsumerClient(ConsumerGroup, EventHubConnectio
    {
        Console.WriteLine("Listening for events...");

        try
        {
            await foreach (PartitionEvent partitionEvent in consumerClient.ReadEventsAsync())
            {
                string messageBody = Encoding.UTF8.GetString(partitionEvent.Data.Body.ToArray());
                Console.WriteLine($"Received event: {messageBody}");
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error reading events: {ex.Message}");
        }
    }
}
```

5. Build and run the application in Visual Studio 2022 Community Edition

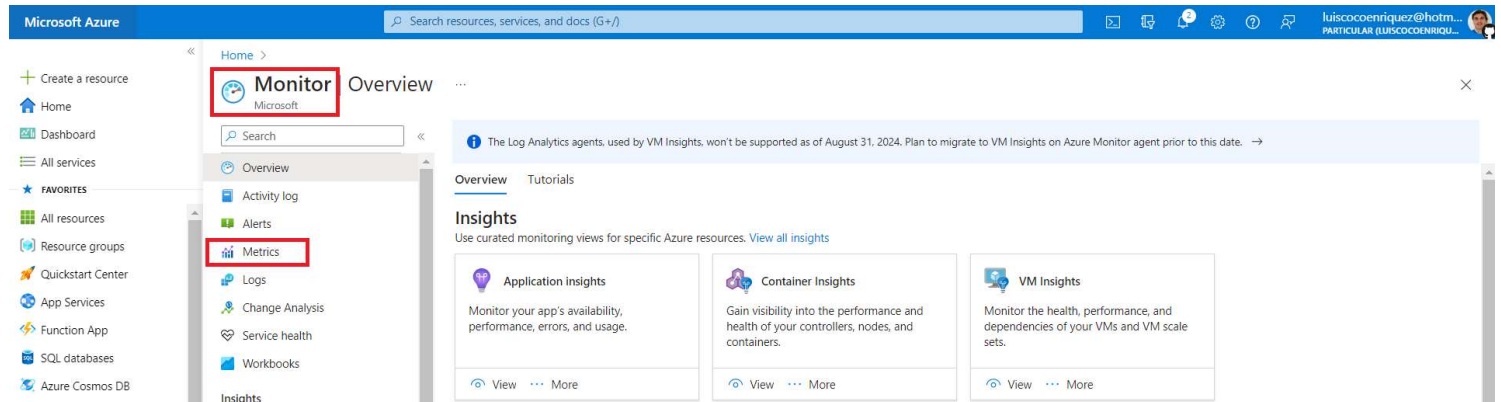
This is the output we get after running the application:



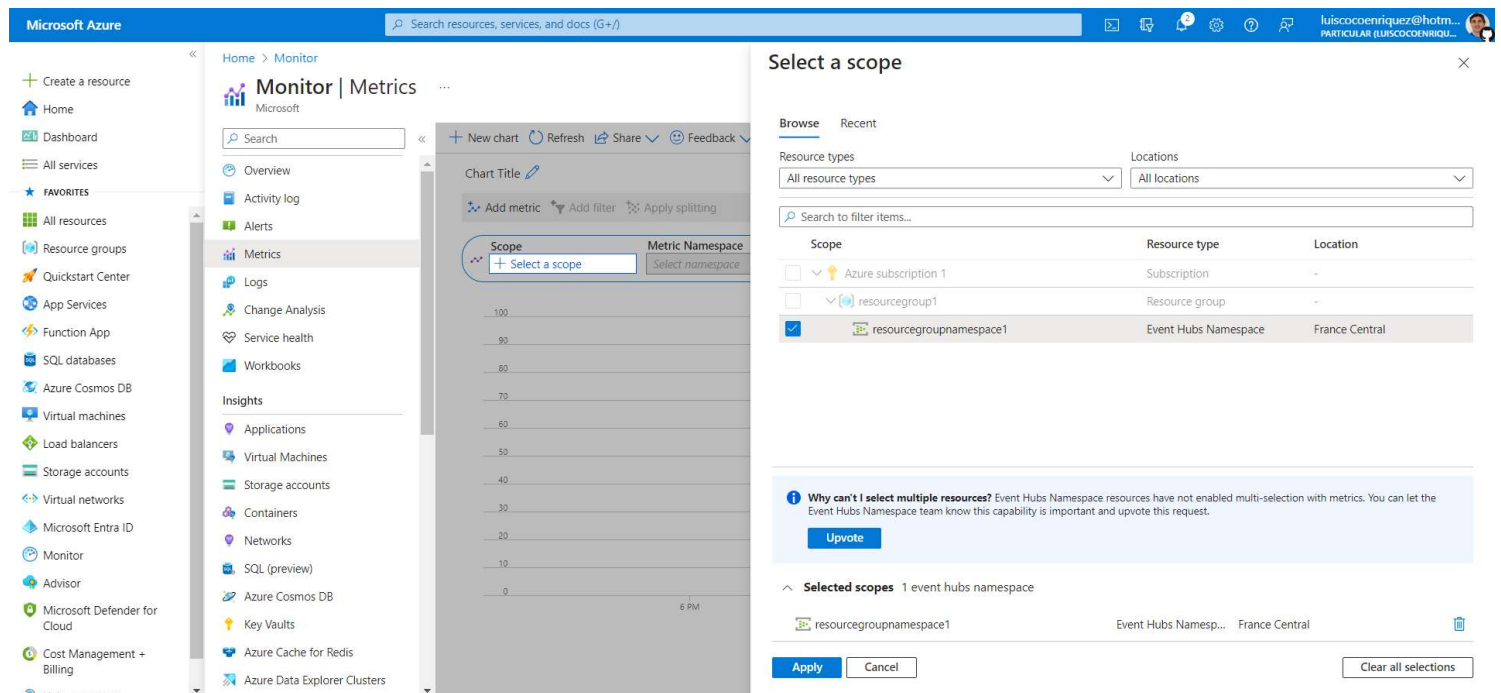
```
C:\Azure EventHub\AzureEventHub_SendEvents\bin\Debug\net8.0\AzureEventHub_SendEvents.exe
Event added to batch: Message 0
Event added to batch: Message 1
Event added to batch: Message 2
Event added to batch: Message 3
Event added to batch: Message 4
Batch of events sent successfully
Listening for events...
Received event: Message 0
Received event: Message 1
Received event: Message 2
Received event: Message 3
Received event: Message 4
```

6. Check in Azure portal we received the messages in the EventHub

Open Azure portal and navigate to the **Monitor** service.



Then select **Metrics** and the EventHub we would like to monitor



Microsoft Azure Monitor | Metrics

Search resources, services, and docs (G+)

Home > Monitor

Overview, Activity log, Alerts, Metrics, Logs, Change Analysis, Service health, Workbooks, Insights, Applications, Virtual Machines, Storage accounts, Containers, Networks, SQL (preview), Azure Cosmos DB, Key Vaults, Azure Data Explorer Clusters

Chart Title

Add metric, Add filter, Apply splitting, Line chart, Drill into Logs, New alert rule, Save to dashboard

Scope: resourcegroupnamespace1, Metric Namespace: Event Hub standard, Metric: Incoming Messages, Aggregation: Sum

Incoming Messages

Incoming Messages for Microsoft.EventHub.

Metric ID: IncomingMessages, Namespace: microsoft.eventhub/namespaces

Supports filtering and grouping

Pin charts to your dashboards

Filter + Split, Plot multiple metrics

Apply filters and splits to identify outlying segments, Create charts with multiple metrics and resources

Local Time: Last 24 hours (Automatic)

Microsoft Azure Monitor | Metrics

Search resources, services, and docs (G+)

Home > Monitor

Overview, Activity log, Alerts, Metrics, Logs, Change Analysis, Service health, Workbooks, Insights, Applications, Virtual Machines, Storage accounts, Containers, Networks, SQL (preview), Azure Cosmos DB, Key Vaults

Sum Incoming Messages for resourcegroupnamespace1

Add metric, Add filter, Apply splitting, Line chart, Drill into Logs, New alert rule, Save to dashboard

Scope: resourcegroupnamespace1, Metric Namespace: Event Hub standard, Metric: Incoming Messages, Aggregation: Sum

Incoming Messages (Sum) resourcegroupnamespace1

5

Local Time: Last 24 hours (Automatic - 5 minutes)

