CLDV6212

PoE Part 2

Nikita Gordhan

ST10108243

Contents

[Part A: Write a .NET core console application to put messages into a data storage queue. 2](#_Toc148188168)

[Screenshot of Code: 2](#_Toc148188169)

[At least 5 Messages in the portal: 3](#_Toc148188170)

[Part B: Change the Azure function in POE Part 1 from using an HTTP trigger to a Queue trigger. 3](#_Toc148188171)

[Screenshot of Code: 3](#_Toc148188172)

[Messages in the SQL Database: 5](#_Toc148188173)

# **Part A: Write a .NET core console application to put messages into a data storage queue.**

## **Screenshot of Code:**

using Azure.Messaging;

using Azure.Storage.Queues;

using System.Text;

//calls the libraries that are needed.

// CONNECTION STRING:

string sConnectionString = "DefaultEndpointsProtocol=https;AccountName=sast10108243;AccountKey=KROMgMKONLB2aMlnn9Ve4UZKN2lLvOUAziCpqY6BvxpIp3zVhPiDLmKYHmvP+motLk6AJwvodbBF+ASt640p/Q==;EndpointSuffix=core.windows.net";

// gets the connection string.

// CREATES THE QUEUE:

string sQueueName = "vaccination-queue";

//assings the queue name to a variable.

QueueClient queueClient = new QueueClient(sConnectionString, sQueueName);

// the line of code above was taken and adapted from Microsoft Documents.

// https://learn.microsoft.com/en-us/dotnet/api/azure.storage.queues.queueclient?view=azure-dotnet

Console.WriteLine($"Creating the queue {sQueueName}");

await queueClient.CreateIfNotExistsAsync();

//creates the queue if it does not exit.

//the line of code above was taken and adapted from Microsoft Documents.

//https://learn.microsoft.com/en-us/azure/storage/queues/storage-quickstart-queues-dotnet?tabs=passwordless%2Croles-azure-portal%2Cenvironment-variable-windows%2Csign-in-azure-cli#create-a-queue

// INSERTS MESSAGES:

Console.WriteLine($"\nAdding the messages to {sQueueName}.");

string m1 = "9405105021537:HealthClinic1:21-08-2023:VSNVAR2102023";

queueClient.SendMessage(Convert.ToBase64String(Encoding.UTF8.GetBytes(m1)));

string m2 = "A274914658:HealthClinic2:21-08-2023:VSNVAR21082023";

queueClient.SendMessage(Convert.ToBase64String(Encoding.UTF8.GetBytes(m2)));

string m3 = "VBNVAR21082023:21-08-2023:HealthClinic3:6285429502175";

queueClient.SendMessage(Convert.ToBase64String(Encoding.UTF8.GetBytes(m3)));

string m4 = "VBNVAR21082025:21-08-2023:HealthClinic4:A285936254";

queueClient.SendMessage(Convert.ToBase64String(Encoding.UTF8.GetBytes(m4)));

string m5 = "VBN2619476:21-08-2023:HealthClinic5:2741037572539";

queueClient.SendMessage(Convert.ToBase64String(Encoding.UTF8.GetBytes(m5)));

//messages to add to the queue.

// the code above(sending messages to the queue) was taken and adapted from StackOverflow.

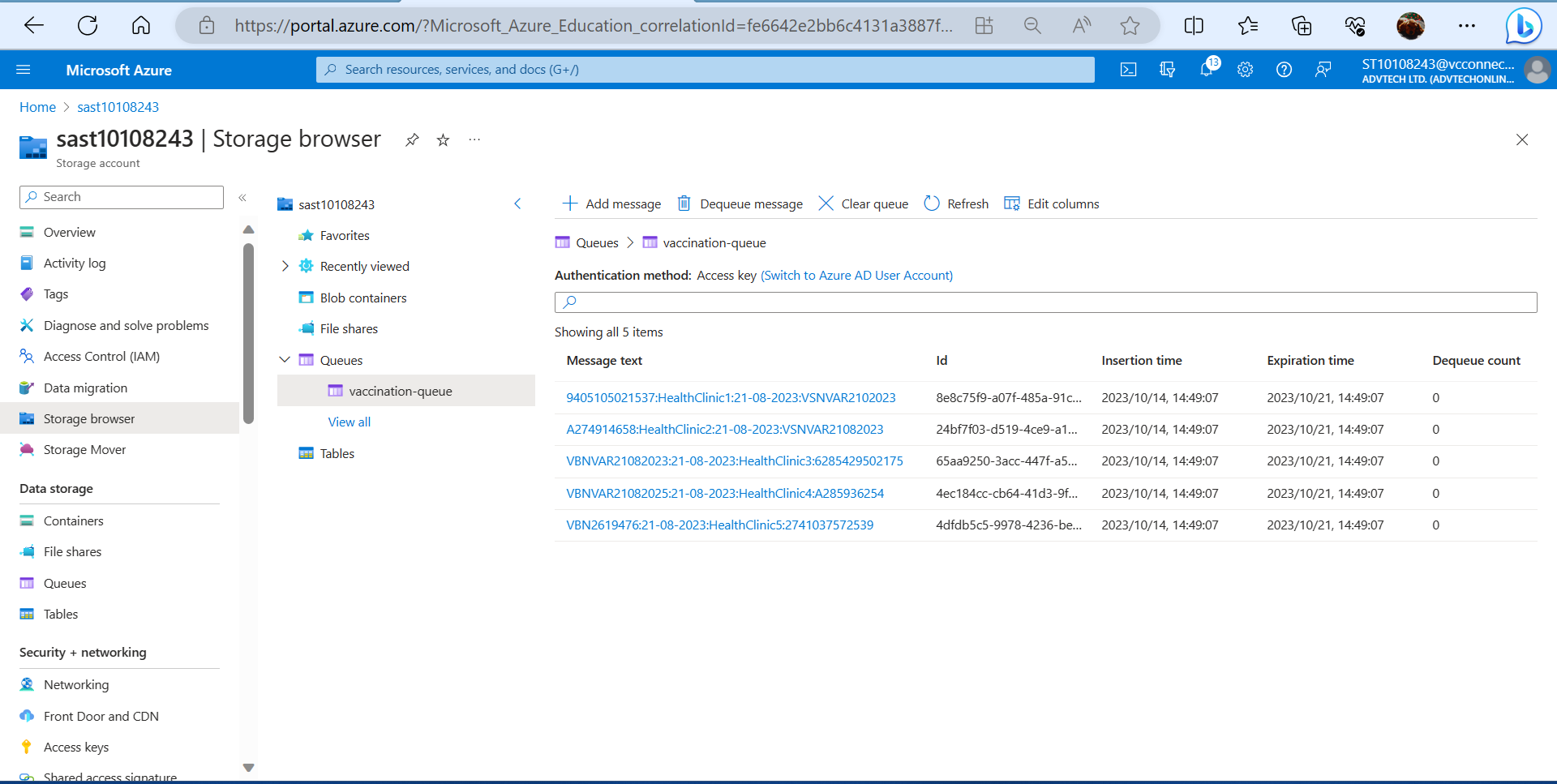
// https://stackoverflow.com/questions/63023481/azure-functions-queue-trigger-is-expecting-base-64-messages-and-doesnt-process

// Johan Classon

// https://stackoverflow.com/users/1482279/johan-classon

Console.WriteLine("Execution Completed");

## **At least 5 Messages in the portal:**



# **Part B: Change the Azure function in POE Part 1 from using an HTTP trigger to a Queue trigger.**

## **Screenshot of Code:**

using System;

using Microsoft.Azure.WebJobs;

using Microsoft.Data.SqlClient;

using Microsoft.Extensions.Logging;

using System.Text;

using Microsoft.AspNetCore.Hosting.Server;

using Microsoft.Azure.Documents;

//calls the libraries that are needed.

namespace ST10108243\_AwehProd\_QueueTrigger

{

public class QueueTrigger

{

[FunctionName("QueueTrigger")]

public async void Run([QueueTrigger("vaccination-queue", Connection = "CONNECTION\_STRING")]string myQueueItem, ILogger log)

{

try

{

string sEncodeMessage = Convert.ToBase64String(Encoding.UTF8.GetBytes(myQueueItem));

// encodes the message to a 64 byte.

// the line of code was taken and adapted from StackOverflow.

// https://stackoverflow.com/questions/11743160/how-do-i-encode-and-decode-a-base64-string

// Zeigeist

// https://stackoverflow.com/users/2649698/zeigeist

string[] messageParts = Encoding.UTF8.GetString(Convert.FromBase64String(sEncodeMessage)).Split(':');

// seperates the string when ever there is a ':'.

// the line of code above was taken and adapted from Microsoft Documenets.

// https://learn.microsoft.com/en-us/dotnet/api/system.convert.frombase64string?view=net-7.0

//assumptions

//if else firt element is 13 or 10 then it first format else its the second format

string sID = "";

string sCenter = "";

string sVaccination\_Date = "";

string sSerial\_Number = "";

if (messageParts[0].Length == 13 || messageParts[0].Length == 10)

{

// format 1- Id:VaccinationCenter:VaccinationDate:VaccineSerialNumber

sID = messageParts[0];

sCenter = messageParts[1];

sVaccination\_Date = messageParts[2];

sSerial\_Number = messageParts[3];

}

else

{

// format 2- VaccineBarcode:VaccinationDate:VaccinationCenter:Id

sID = messageParts[3];

sCenter = messageParts[2];

sVaccination\_Date = messageParts[1];

sSerial\_Number = messageParts[0];

}

// assigns the columns into the array elements.

using (SqlConnection connection = new SqlConnection("Server=tcp:st10108243queuestorageserver.database.windows.net,1433;Initial Catalog=ST10108243QueueStorageDB;Persist Security Info=False;User ID=nikki;Password=CLDV6212PoEPart2;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;"))

{

connection.Open();

using (SqlCommand command = connection.CreateCommand())

{

command.CommandText = "INSERT INTO Messages (ID,Center,Vaccination\_Date,Serial\_Number) VALUES (@ID,@Center,@Vaccination\_Date,@Serial\_Number)";

command.Parameters.AddWithValue("@ID", sID);

command.Parameters.AddWithValue("@Center", sCenter);

command.Parameters.AddWithValue("@Vaccination\_Date", sVaccination\_Date);

command.Parameters.AddWithValue("@Serial\_Number", sSerial\_Number);

command.ExecuteNonQuery();

//adds the message to the SQL Database with the values corresponding to the table colunms.

// the code above was taken and adapted from StackOverflow.

// https://stackoverflow.com/questions/9155004/difference-with-parameters-add-and-parameters-addwithvalue

// John Woo

// https://stackoverflow.com/users/491243/john-woo

}

}

//PROCESS THE MESSAGE:

// log.LogInformation($"Processing the message: {myQueueItem}");

log.LogInformation($"Queue message has been stored successfully in the database ID = {sID}, Center = {sCenter}, Vaccination\_Date = {sVaccination\_Date}, Serial\_Number = {sSerial\_Number} ");

}

catch (Exception ex)

{

log.LogError($"Error in processing the queue message: {ex.Message}");

}

}

}

}

## **Messages in the SQL Database:**

