16 OCTOBER 2023

CLDV6212

POE PART 1 & 2

ST10075585

NSIKELELO TSOELOPELE THAMSANQA KUMALO

CLDV6212 POE PART 1 & 2 ST10075585

Nsikelelo Kumalo

Contents

SECTION A.

Traditional On-Premises		Modern Cloud	
On-premises definition	On-premises example	Cloud definition	Cloud example
Monolithic: a self-contained software model that operates with a single code base. It is commonly linked with large and inflexible environments, which can lead to time-consuming and restrictive updates. However, monoliths provide advantages in terms of code management, reduced cognitive complexity, and streamlined deployment, enabling the quick release of all components at once. (Harris, 2023)	An online marketplace operates using a monolithic architecture within an on-premises environment. All the functions of the platform, such as product search, user authentication, and payment processing, are tightly integrated into a single software application hosted on the company's own servers. This setup requires manual scaling and maintenance, making it more challenging to adapt to changing demands compared to a cloud-based decomposed architecture.	Decomposed: A decomposed architecture, commonly known as microservices architecture, is a design approach that involves dividing a complex application into smaller, self-contained units called microservices. Each microservice is responsible for a specific business function and can be developed, deployed, and scaled independently without reliance on other services. By breaking down major business concerns into separate, autonomous code bases, microservices become more visible and easier to handle. These smaller processes operate independently, working together to create the complete solution (Harris, 2023)	An online marketplace runs on a cloud-based decomposed architecture. Various microservices handle functions like product search, user authentication, and payment processing. These microservices communicate over the internet and can be independently scaled to meet demand.
Designed for predictable scalability: the ability of a system to handle increasing demands without affecting its performance. In the context of On-premises, predictable scalability means that the system can be easily expanded to meet the needs of a growing business.	DStv, a popular African satellite television service, is designed for predictable scalability due to its fixed number of satellite transponders. This scalability is more predictable than streaming platforms. During major sporting events like the FIFA World Cup or Olympics, DStv can anticipate a significant increase in viewership. To handle this, they	Designed for elastic scale refers to the ability of an on-premises IT infrastructure to be scaled up or down as needed. This can be done by adding or removing physical servers, virtual machines, or containers. (madhav_mohan, 2023)	Netflix is a prime example of a service that is designed for elastic scale. They operate in the cloud and have built their architecture to automatically scale up or down based on real-time demand. During peak times, like when a popular show releases new episodes, or during weekends and

This can be done by adding more servers, storage, or network bandwidth. (Glossary, n.d.)	plan and allocate additional transponders. By leveraging historical data and understanding viewing patterns, they can estimate the required scalability to maintain high-quality broadcasting without disruptions. This scalability is foreseeable due to the nature of events and historical viewership data, allowing DStv to plan ahead and allocate resources for uninterrupted broadcasting during high-demand periods.		evenings when more people are streaming, Netflix can dynamically allocate additional resources to ensure smooth playback and user experience. Conversely, during off-peak times, resources are scaled down to save costs. This elastic scaling allows Netflix to maintain a consistent user experience regardless of varying levels of demand.
Relational database: allows administrators to update physical storage without changing the logical data structure by organising data points with established relationships for simple access. They function effectively with organised data and assist organisations in making efficient business decisions and minimising expenditures. (Lutkevich, 2021)	In the context of a banking system like ABSA, a relational database could be used to manage core customer data, account information, transaction history, and user authentication. For instance, ABSA might use a relational database to store structured information about customers, their personal details, account balances, and transaction records. This data would be stored in tables with predefined schemas using SQL queries for retrieval and management. This approach ensures data integrity, ACID (Atomicity, Consistency, Isolation, Durability)	Polyglot persistence (mix of storage technologies): is a concept involving multiple data storage approaches and technologies for enterprise applications. It suggests that database engineers should prioritize determining the best database technology for application data manipulation to address storage performance issues, simplify operations, and prevent fragmentation. (Brunskill, 2023)	In the case of Polyglot Persistence, ABSA might use different storage technologies to address diverse data needs. For instance, ABSA could utilize a NoSQL database for handling unstructured or semi-structured data, such as customer feedback, social media interactions, or logs. This allows ABSA to store and process large volumes of unstructured data more efficiently than a traditional relational database. The bank might also use a specialized time-series database to store and analyse real-time market data for investment purposes. By adopting a polyglot

	compliance, and robust data relationships, all crucial aspects of a banking system.		persistence approach, ABSA can optimize data storage and retrieval for different types of information, achieving better performance and scalability.
Synchronized processing is a way of ensuring that multiple processes access shared resources in a consistent manner. This is done by using locks or semaphores to prevent one process from accessing a resource that is being used by another process. (GeeksforGeeks, 2023)	In Google Docs, when multiple users collaborate on a document in real time, their changes and edits are synchronized in a synchronized processing manner. When one user types or makes changes, others can immediately see the modifications happening in real time. Each change is synchronized and displayed to all collaborators as soon as it's made, ensuring that everyone is working on the most up-to-date version of the document.	Asynchronous processing: provides a means of distributing the processing that is required by an application between systems in an intercommunication environment. distributes the processing required by an application between intercommunicating systems. The processing is independent of the sessions in which requests are sent and replies are received. Remains the same as the general concept, involving tasks that are decoupled and can run independently. (IBM, 2023)	Google Photos Backup When you upload photos to Google Photos from your mobile device, the backup process uses asynchronous processing. Your device starts the upload process, and Google Photos continues the backup in the background, even if you exit the app or turn off your device. The photos are uploaded asynchronously, allowing you to continue using your device without waiting for the entire backup to complete. This asynchronous approach ensures that your photos are securely backed up to the cloud without interrupting your device's performance or your usage.
Design to avoid failures (MTBF): is a metric used to measure the reliability and availability of a technology product.	SHEIN implements a Design to Avoid Failures (MTBF) strategy for Black Friday by carefully planning and optimizing its infrastructure to	Design for failure (MTTR): the average time it takes to fully resolve a failure, including detecting, diagnosing, and repairing the issue,	SHEIN also employs a Design for Failure (MTTR) approach during Black Friday to ensure quick recovery in case of any unexpected system failures. They

Companies aim to maintain high MTBF to minimize failure frequency. On-premises design aims to increase system reliability by minimizing failure frequency through redundancy and careful system design. (ATLASSIAN, 2023)	handle the anticipated surge in traffic. The company invests in high-quality servers, redundant hardware components, and load balancers to ensure stable performance throughout the Black Friday sale. They analyse historical data to predict traffic patterns and allocate resources based on expected demand. By doing so, SHEIN aims to provide a seamless shopping experience for its customers, minimizing the chances of server crashes or slowdowns during peak hours.	and ensuring it doesn't happen again. It extends the team's responsibility to improve long-term performance, similar to fireproofing a house. It's strongly correlated with customer satisfaction, and in the cloud, design for failure emphasizes rapid recovery using cloud-specific services and tools. (ATLASSIAN, 2023)	set up automated monitoring tools that constantly track server health and response times. If any server or component experiences issues, the system triggers an automatic alert. In response, the platform immediately redirects traffic away from the affected server and provisions a new instance to replace the problematic one. This approach minimizes downtime and ensures that customers can continue shopping without disruption. SHEIN's focus is on rapid recovery and maintaining availability rather than trying to prevent all failures.
Occasional large updates: software updates that are released less frequently than small, incremental updates. They typically contain more significant changes, such as new features, major bug fixes, or performance improvements (reddit, 2011)	Microsoft 365 (formerly Office 365) often receives occasional large updates. These updates are significant and may introduce major new features, user interface changes, and improvements to the suite of applications. For instance, every few months, Microsoft might release a substantial update that includes new tools, enhanced collaboration features, or a redesigned user	Frequent small updates are software updates that are released on a regular basis, typically weekly or monthly. They typically include bug fixes, security patches, and minor new features. (Eckert, 2012)	On the other hand, Microsoft 365 also experiences frequent small updates. These updates are more incremental and focus on bug fixes, performance enhancements, security patches, and minor feature improvements. These updates are rolled out more regularly, sometimes even on a weekly or bi-weekly basis, to ensure that the software remains stable, secure, and

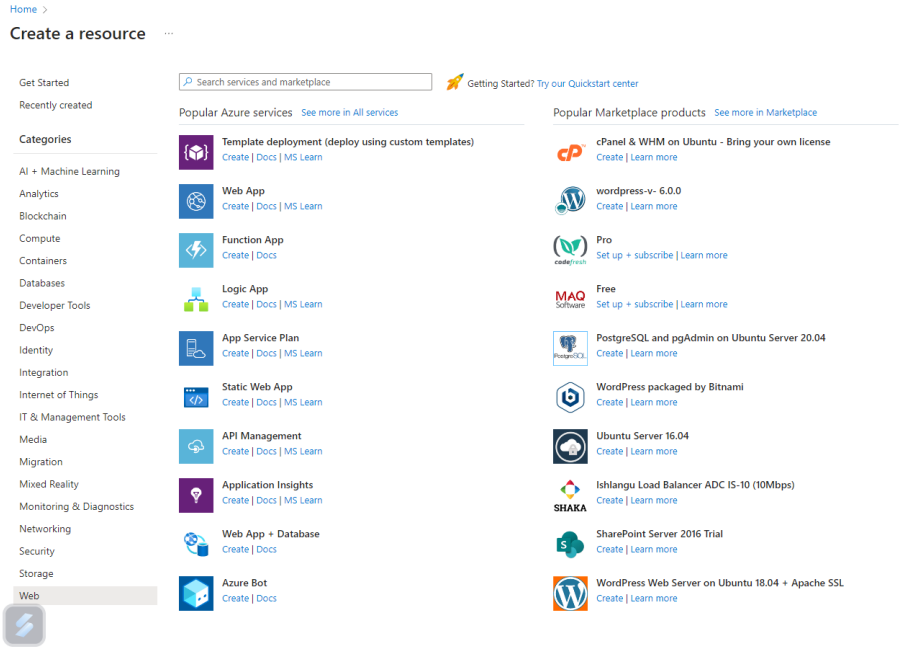
	interface. These updates require users to adjust to the changes and sometimes come with accompanying learning resources to help users adapt to the new features.		up to date. These small updates might not introduce major changes but collectively contribute to the overall user experience and reliability of the software suite.
Manual management: is the manual management of a system or process using pen and paper, spreadsheets, or other non-automated technologies. It can be applied in a variety of scenarios, although it is most typically applied to minor systems or processes that do not require a high level of complexity or sophistication. It can also be used as a stopgap measure until a more automated system can be put in place. (LinkedIn, 2023)	Imagine a company that operates its network infrastructure on-premises. The IT team is responsible for configuring routers, switches, firewalls, and load balancers manually. If a new application needs to be deployed, the team must physically set up the required networking components, configure routing rules, and ensure proper security settings. This process is time-consuming and prone to human errors, and any changes require manual intervention. In case of a sudden traffic surge, the IT team needs to manually adjust the network resources to handle the increased load, which might lead to delays in responding to the changing demands.	Automated self-management: Involves the utilization of automation and orchestration tools to manage activities such as deployment, scaling, and recovery autonomously, minimizing the need for constant human involvement. (Lynn, et al., 2016) (Puviani & Frei, 2013)	Contrast this with a cloud-based network environment. An organization utilizes a cloud service provider for its network infrastructure. The cloud platform offers automated networking services like load balancing, virtual networking, and firewall configuration. When a new application is deployed, the networking components are provisioned automatically according to predefined templates. As traffic patterns change, the cloud platform can automatically scale the network resources up or down based on demand, ensuring optimal performance and minimal downtime. Security updates and patches for network components are also applied automatically by the cloud provider, reducing the burden on the IT team. This

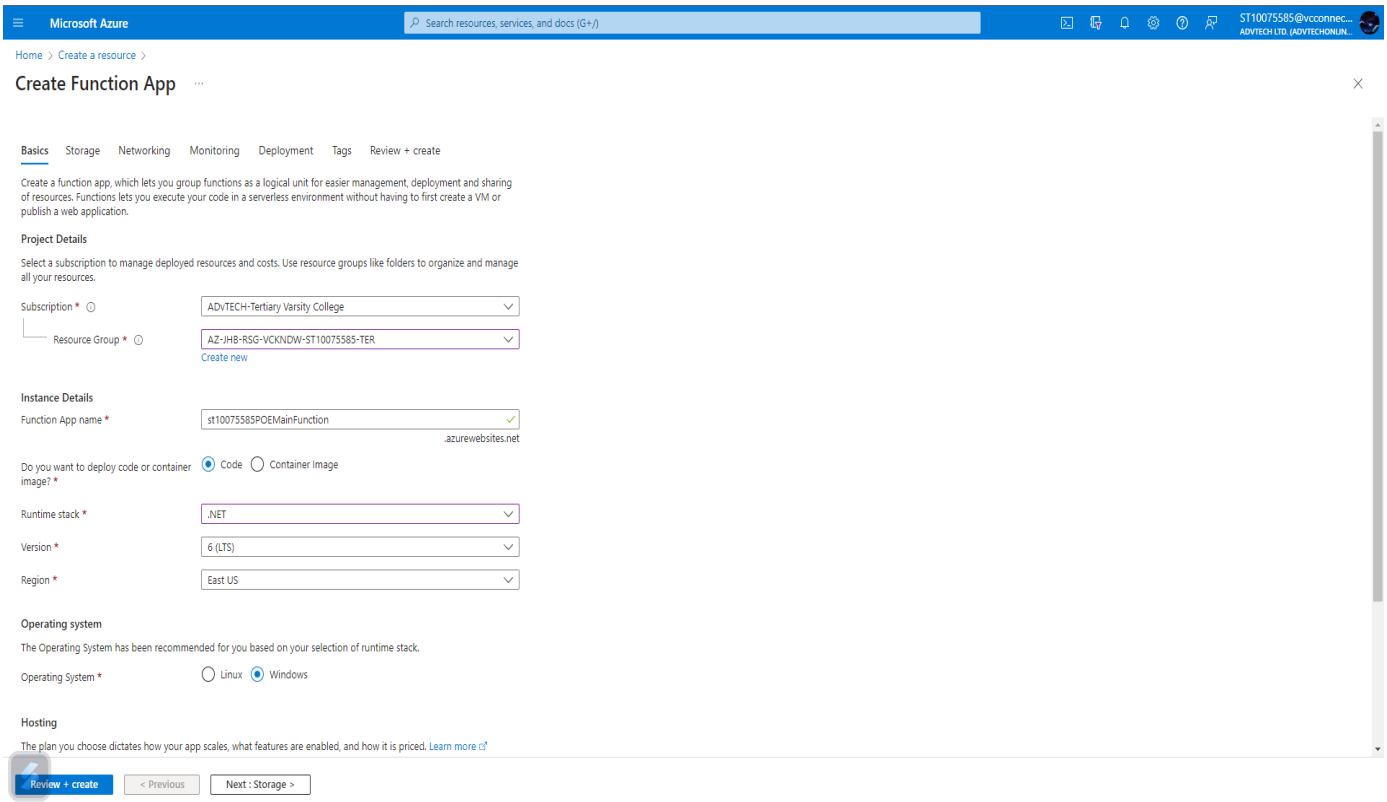
			automated approach allows the organization to focus more on application development and business needs rather than routine network management tasks.
Snowflake servers: launched mission-critical software that can only be launched on a specified configuration of operating system and application server. Because these servers cannot be upgraded, they drift further and further away from normal setups, requiring more and more IT resources to monitor. Snowflake servers make automation techniques like Infrastructure as Code harder to implement. Having many Snowflake Servers results in what is known as a fragile infrastructure. (Kemp, 2023)	Imagine an on-premises version of Instagram where the platform's servers are manually configured with unique setups for handling different features. For instance, there might be specific servers optimized for photo uploads, others tailored for video processing, and more dedicated to user authentication. Over time, as new features are added and server configurations become more intricate, managing these servers becomes complex. If one server experiences a performance issue, replicating its unique configuration to troubleshoot the issue across other servers is challenging. This diversity of server setups can lead to inconsistencies and difficulties in maintaining a	Immutable infrastructure: refers to computer infrastructure (virtual machines, containers, and network appliances) that once deployed cannot be modified. This can be enforced by an automated procedure that overwrites unauthorised changes or by a system that does not allow changes to begin with. (Cloud Native Glossary Authors, 2022)	Consider the cloud-based version of Instagram, where the platform follows immutable infrastructure principles. Whenever updates or changes are required, the platform creates new instances from standardized images or templates. For instance, a new version of the image might include the latest codebase, software libraries, and configurations. In this scenario, if an issue arises after a deployment, the platform can quickly revert to the previous version by launching instances from the earlier image. This ensures that all instances are consistent and eliminates the risk of configuration drift. Updates become streamlined, and the platform benefits from reproducible environments, making it easier to maintain a reliable and scalable social media platform like

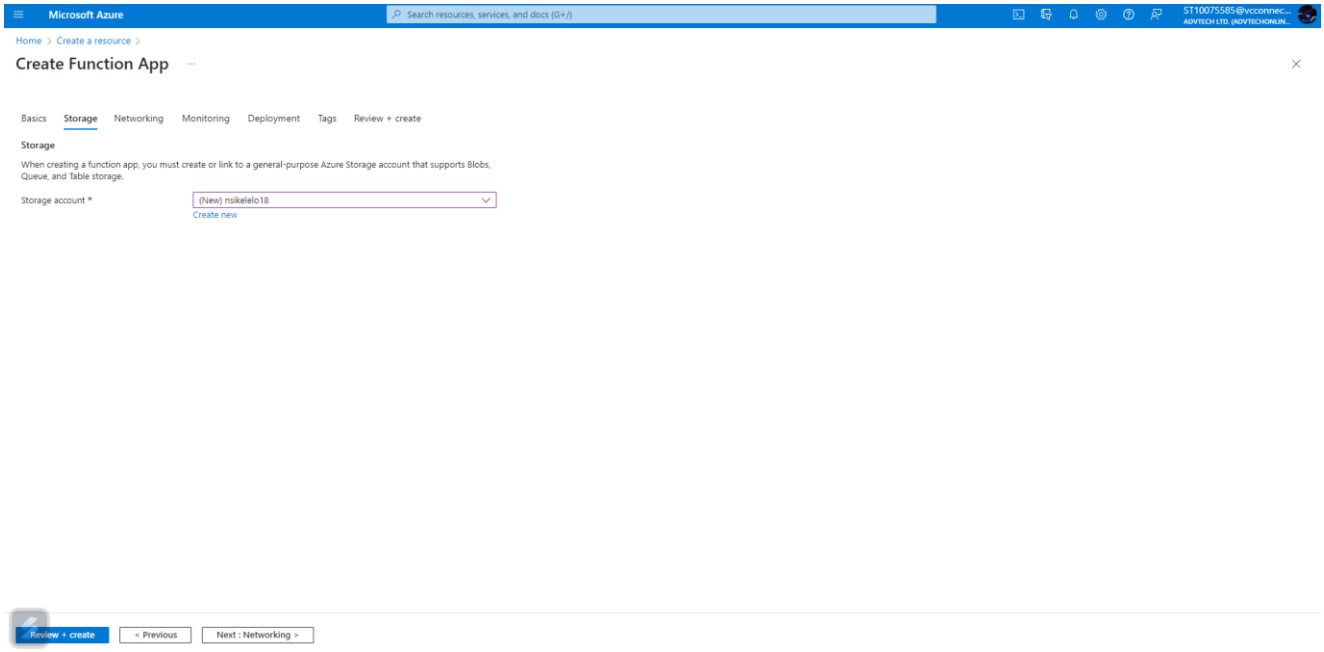
	cohesive and reliable user experience.		Instagram in the cloud.

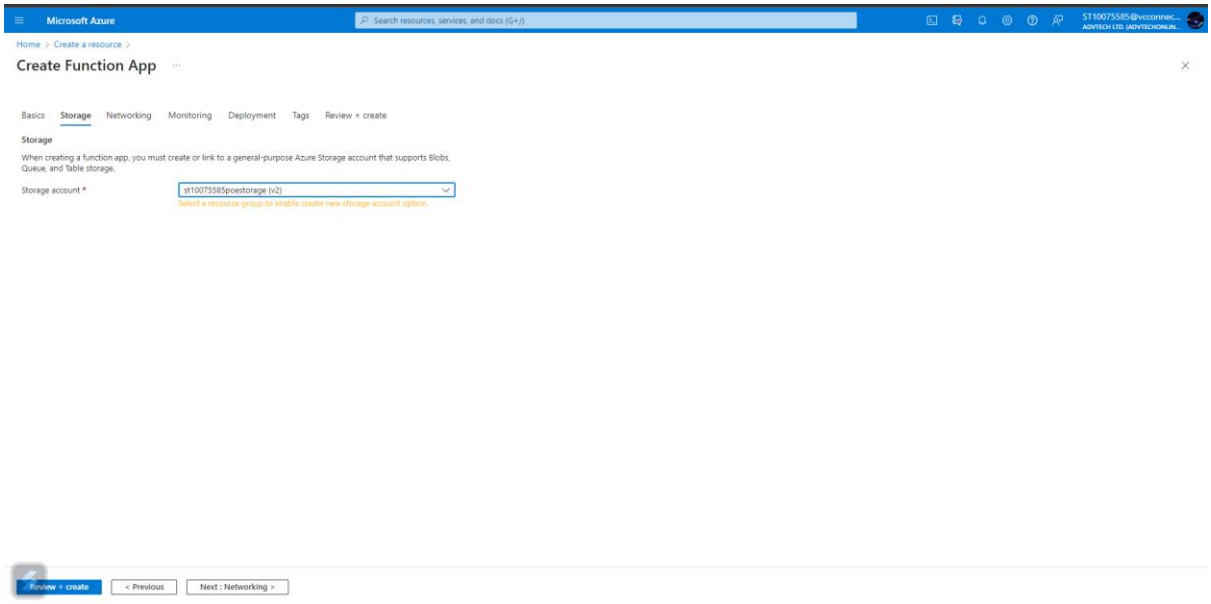
SECTION B.

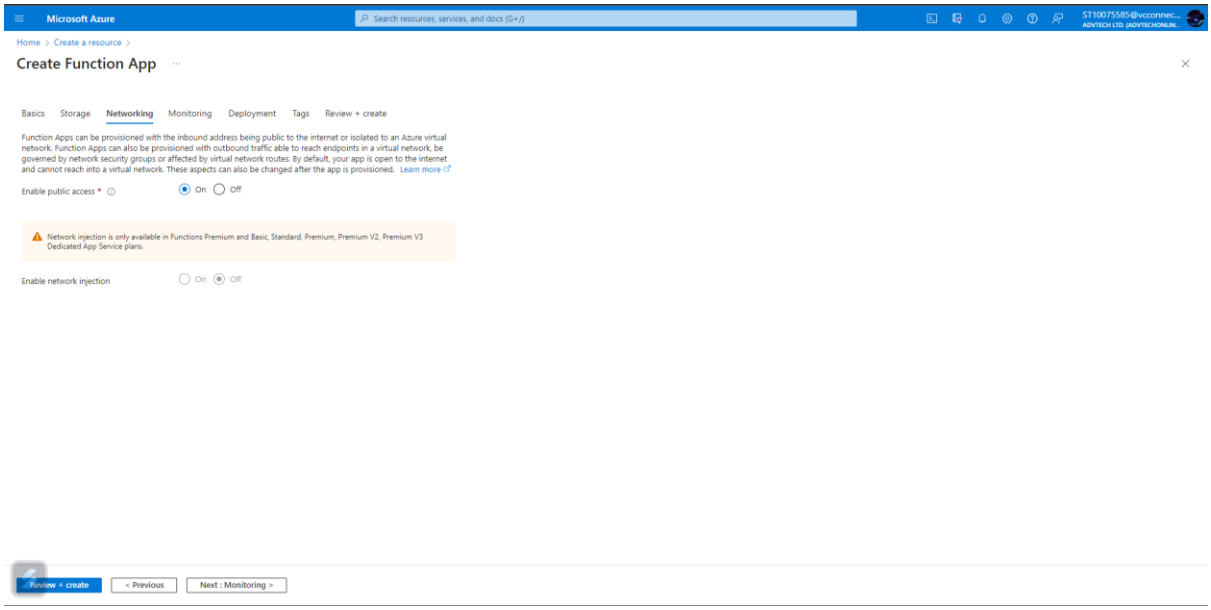
The steps you took to deploy the function (there must be several steps):

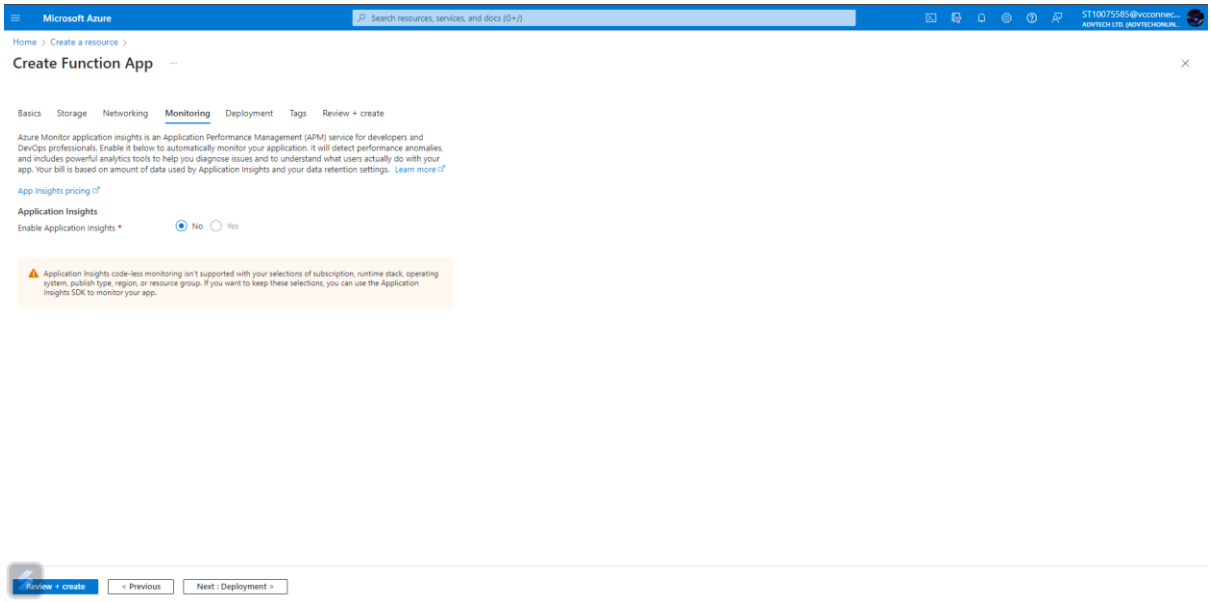


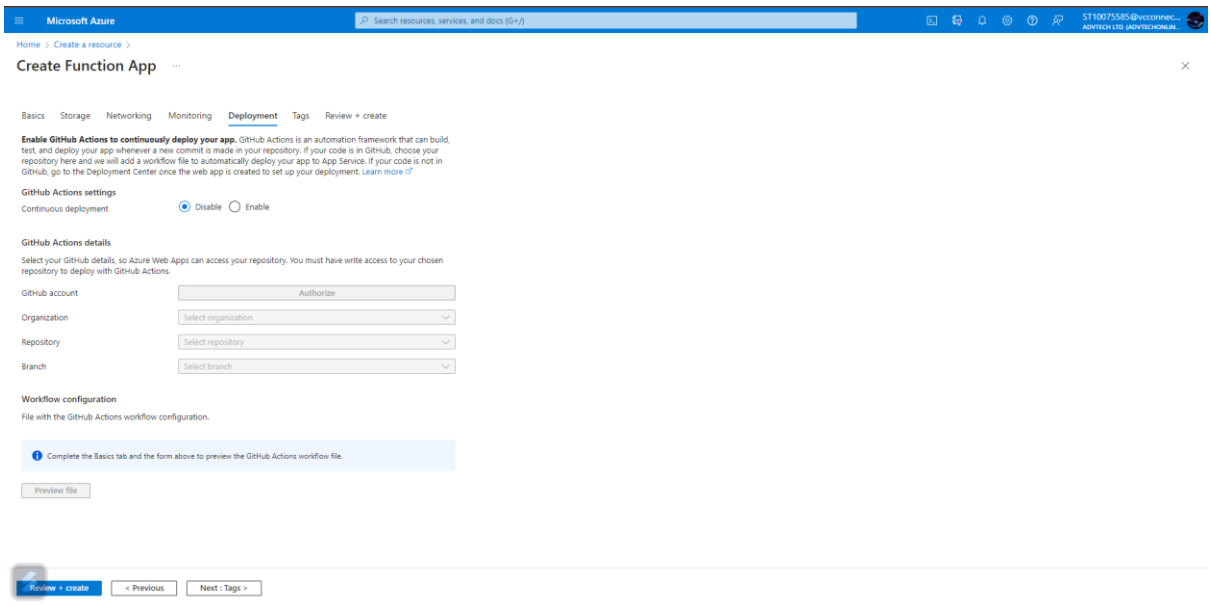


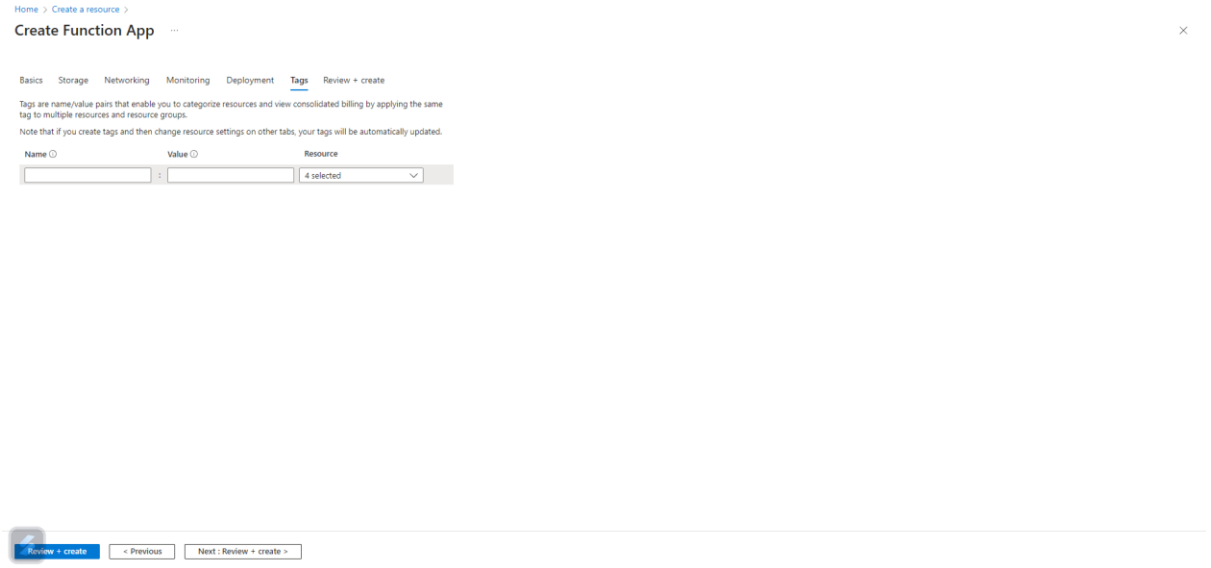

Microsoft Azure
Search resources, services, and docs (G+/)
ST10075585@vcconnec...
ADVTECH LTD. (ADVTECHONLIN...
Home > Create a resource >
Create Function App
Basics
Storage
Networking
Monitoring
Deployment
Tags
Review + create
Storage
When creating a function app, you must create or link to a general-purpose Azure Storage account that supports Blobs, Queue, and Table storage.
Storage account *
(New) nsikelelo18
Create new
Review + create
< Previous
Next : Networking >

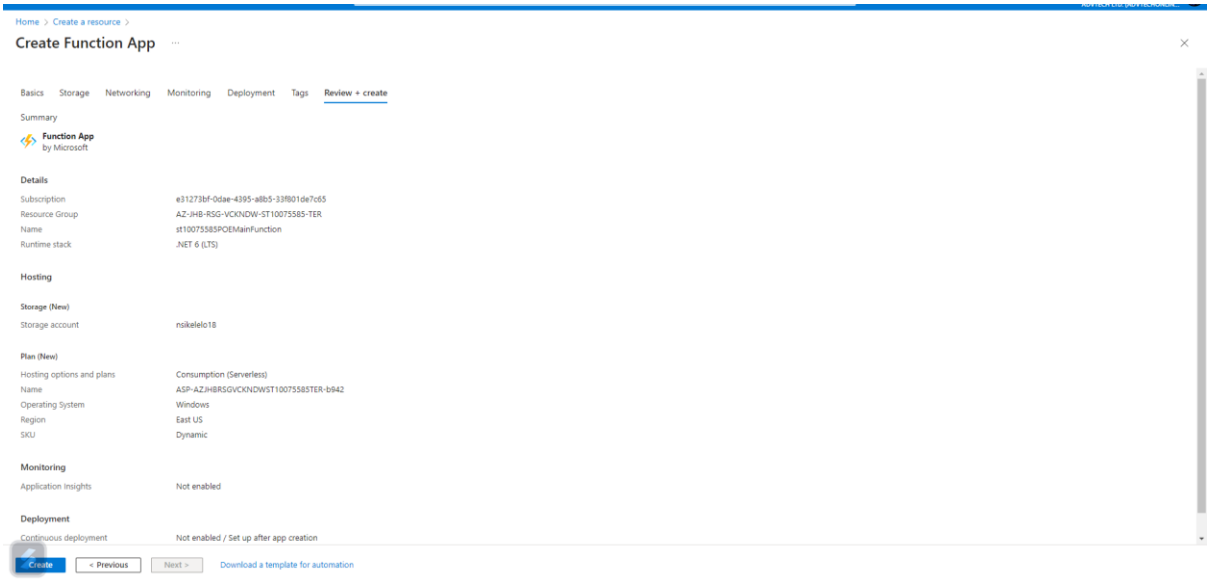

Microsoft Azure
Search resources, services, and docs (G+/)
ST10075585@vcconnec...
ADVTECH LTD. (ADVTECHONLIN...
Home > Create a resource >
Create Function App
Basics
Storage
Networking
Monitoring
Deployment
Tags
Review + create
Storage
When creating a function app, you must create or link to a general-purpose Azure Storage account that supports Blobs, Queue, and Table storage.
Storage account *
st10075585poestorage (v2)
Select a resource group to enable create new storage account option.
Review + create
< Previous
Next : Networking >

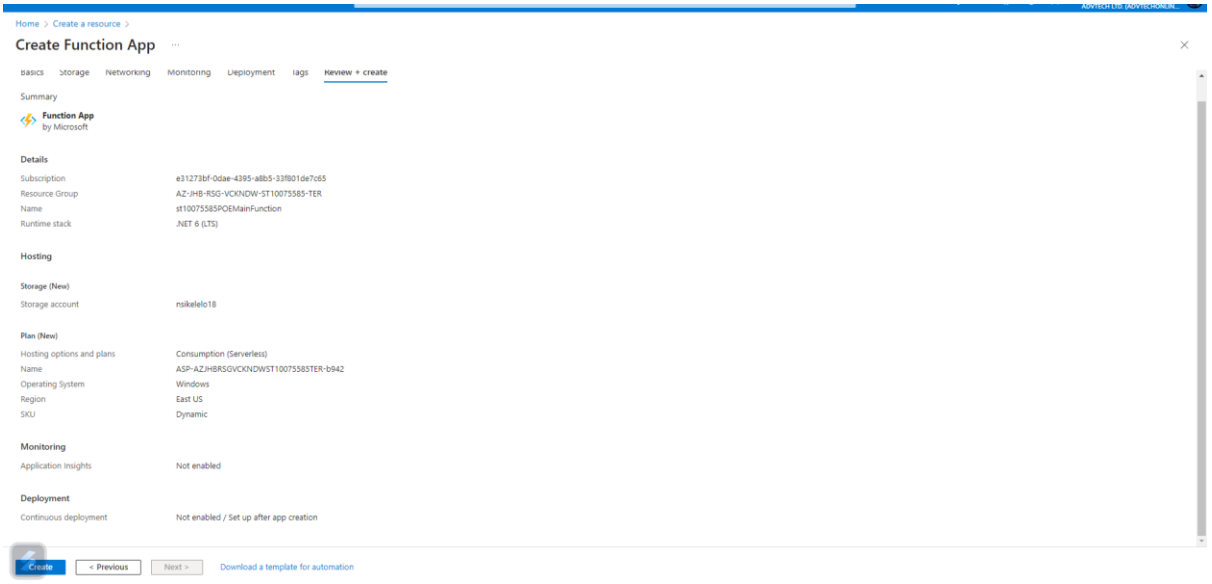

Microsoft Azure
Search resources, services, and docs (G+/)
ST10075585@vcconnec...
ADVTECH LTD. (ADVTECHONLIN...
Home > Create a resource >
Create Function App
Basics Storage Networking Monitoring Deployment Tags Review + create
Function Apps can be provisioned with the inbound address being public to the internet or isolated to an Azure virtual network. Function Apps can also be provisioned with outbound traffic able to reach endpoints in a virtual network, be governed by network security groups or affected by virtual network routes. By default, your app is open to the internet and cannot reach into a virtual network. These aspects can also be changed after the app is provisioned. Learn more
Enable public access *
On
Off
Network injection is only available in Functions Premium and Basic, Standard, Premium, Premium V2, Premium V3 Dedicated App Service plans.
Enable network injection
On
Off
Review + create
< Previous
Next : Monitoring >

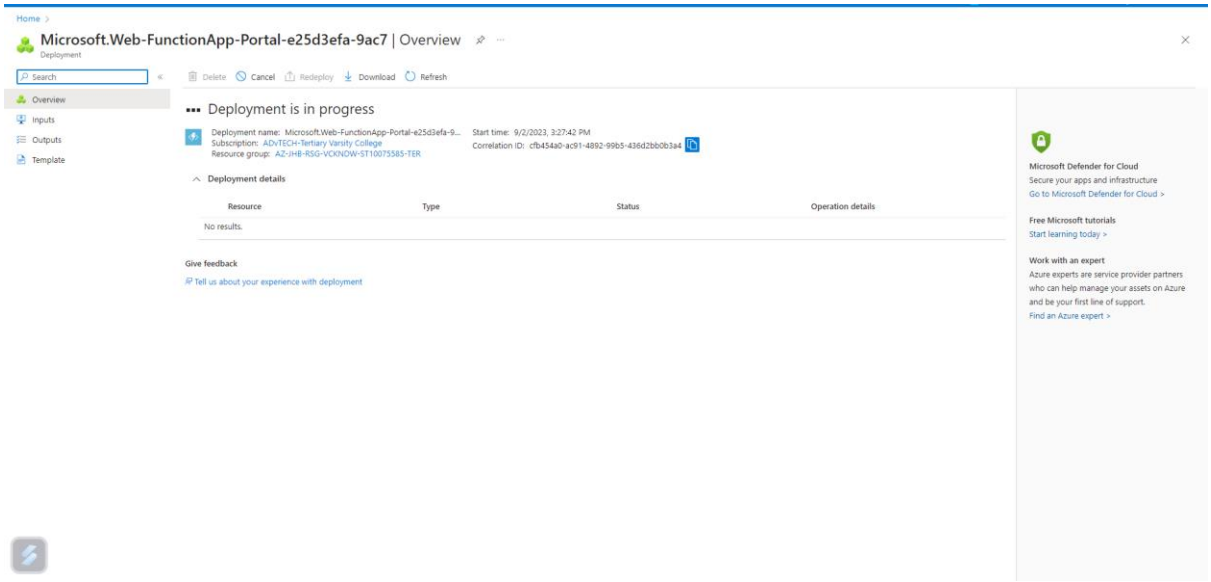

Microsoft Azure
Search resources, services, and docs (G+/)
ST10075585@vcconnec...
ADVTECH LTD. (ADVTECHONLIN...
Home > Create a resource >
Create Function App
Basics Storage Networking Monitoring Deployment Tags Review + create
Azure Monitor application insights is an Application Performance Management (APM) service for developers and DevOps professionals. Enable it below to automatically monitor your application. It will detect performance anomalies, and includes powerful analytics tools to help you diagnose issues and to understand what users actually do with your app. Your bill is based on amount of data used by Application Insights and your data retention settings. Learn more
App Insights pricing
Application Insights
Enable Application Insights *
No
Yes
Application Insights code-less monitoring isn't supported with your selections of subscription, runtime stack, operating system, publish type, region, or resource group. If you want to keep these selections, you can use the Application Insights SDK to monitor your app.
Review + create
< Previous
Next : Deployment >

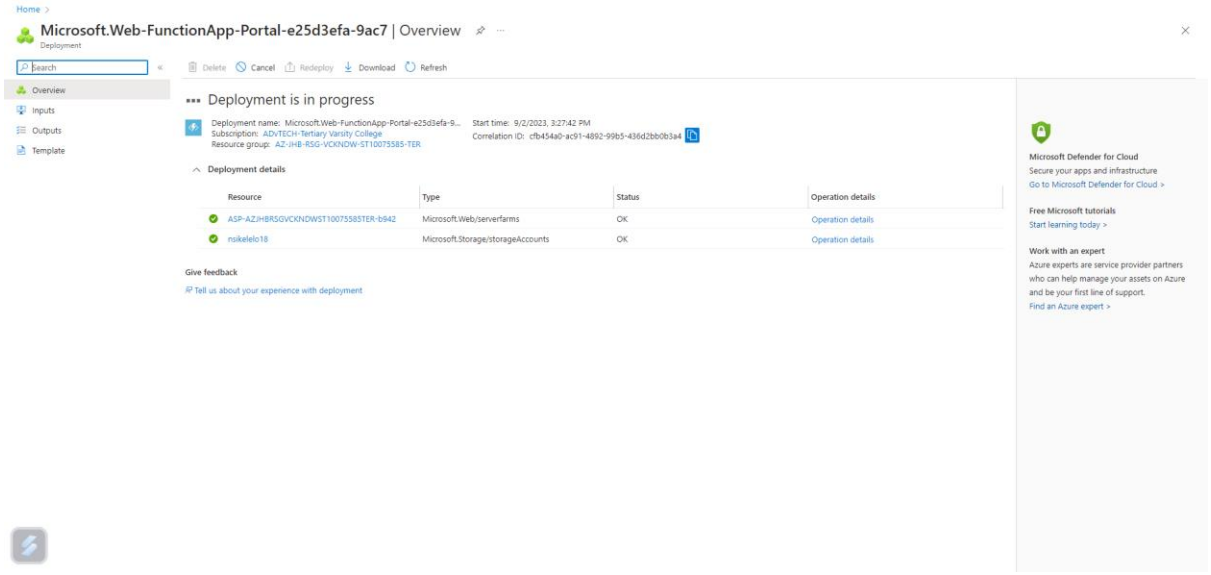

Microsoft Azure
Search resources, services, and docs (G+/)
ST10075585@vcconnec...
ADVTECH LTD. (ADVTECHONLIN...
Home > Create a resource >
Create Function App
Basics Storage Networking Monitoring Deployment Tags Review + create
Enable GitHub Actions to continuously deploy your app. GitHub Actions is an automation framework that can build, test, and deploy your app whenever a new commit is made in your repository. If your code is in GitHub, choose your repository here and we will add a workflow file to automatically deploy your app to App Service. If your code is not in GitHub, go to the Deployment Center once the web app is created to set up your deployment. Learn more
GitHub Actions settings
Continuous deployment
Disable
Enable
GitHub Actions details
Select your GitHub details, so Azure Web Apps can access your repository. You must have write access to your chosen repository to deploy with GitHub Actions.
GitHub account
Authorize
Organization
Select organization
Repository
Select repository
Branch
Select branch
Workflow configuration
File with the GitHub Actions workflow configuration.
Complete the Basics tab and the form above to preview the GitHub Actions workflow file.
Preview file
Review + create
< Previous
Next : Tags >

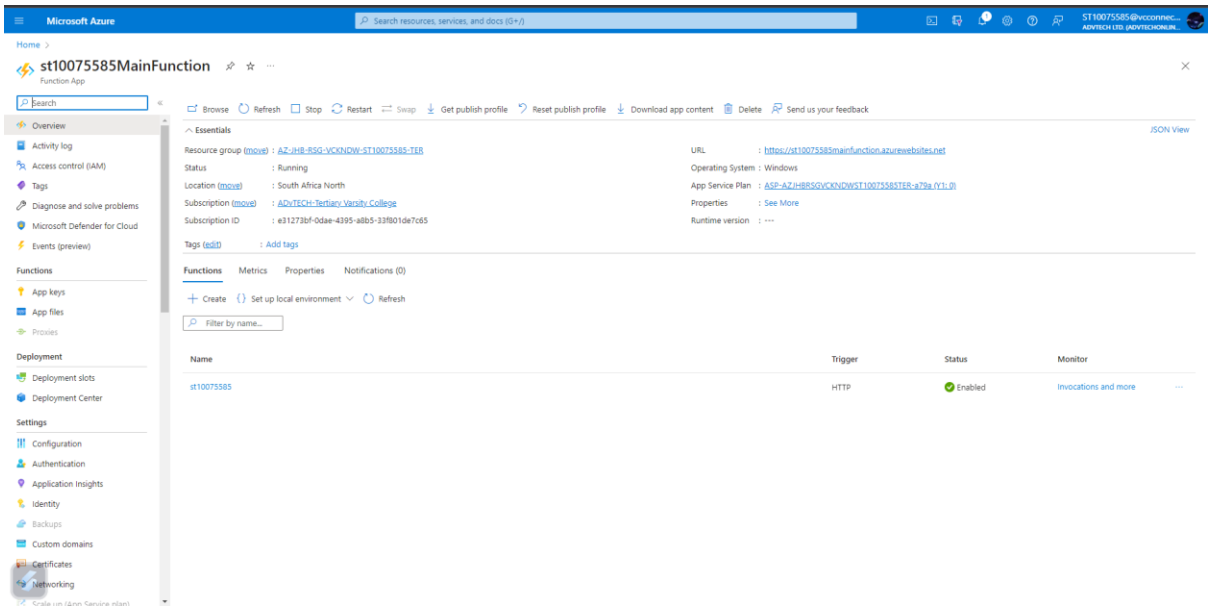

Home > Create a resource >
Create Function App
Basics Storage Networking Monitoring Deployment Tags Review + create
Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups.
Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.
Name
Value
Resource
4 selected
Review + create
< Previous
Next : Review + create >


Home > Create a resource >
Create Function App
Basics Storage Networking Monitoring Deployment Tags Review + create
Summary
Function App
by Microsoft
Details
Subscription e31273bf-0dae-4395-a8b5-33f801de7c65
Resource Group AZ-JHB-RSG-VCKNDW-ST10075585-TER
Name st10075585POEMainFunction
Runtime stack .NET 6 (LTS)
Hosting
Storage (New)
Storage account nsikelelo18
Plan (New)
Hosting options and plans Consumption (Serverless)
Name ASP-AZJHBRSGVCKNDWST10075585TER-b942
Operating System Windows
Region East US
SKU Dynamic
Monitoring
Application Insights Not enabled
Deployment
Continuous deployment Not enabled / Set up after app creation
Create
< Previous
Next >
Download a template for automation


Home > Create a resource >
Create Function App
Summary
Function App
by Microsoft
Details
Subscription e31273bf-0dae-4395-a8b5-33f801de7c65
Resource Group AZ-JHB-RSG-VCKNDW-ST10075585-TER
Name st10075585POEMainFunction
Runtime stack .NET 6 (LTS)
Hosting
Storage (New)
Storage account nsikelelo18
Plan (New)
Hosting options and plans Consumption (Serverless)
Name ASP-AZJHBRSGVCKNDWST10075585TER-b942
Operating System Windows
Region East US
SKU Dynamic
Monitoring
Application Insights Not enabled
Deployment
Continuous deployment Not enabled / Set up after app creation
Create
< Previous
Next >
Download a template for automation


Home
Microsoft.Web-FunctionApp-Portal-e25d3efa-9ac7 | Overview
Deployment
Search
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Download
Refresh
Deployment is in progress
Deployment name: Microsoft.Web-FunctionApp-Portal-e25d3efa-9...
Subscription: ADVTECH-Tertiary Varsity College
Resource group: AZ-JHB-RSG-VCKNDW-ST10075585-TER
Start time: 9/2/2023, 3:27:42 PM
Correlation ID: cfb454a0-ac91-4892-99b5-436d2bb0b3a4
Deployment details
Resource
Type
Status
Operation details
No results.
Give feedback
Tell us about your experience with deployment
Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >
Free Microsoft tutorials
Start learning today >
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 >


Home
Microsoft.Web-FunctionApp-Portal-e25d3efa-9ac7 | Overview
Deployment
Search
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Download
Refresh
Deployment is in progress
Deployment name: Microsoft.Web-FunctionApp-Portal-e25d3efa-9...
Subscription: ADVTECH-Tertiary Varsity College
Resource group: AZ-JHB-RSG-VCKNDW-ST10075585-TER
Start time: 9/2/2023, 3:27:42 PM
Correlation ID: cfb454a0-ac91-4892-99b5-436d2bb0b3a4
Deployment details
Resource
Type
Status
Operation details
ASP-AZJHBRSGVCKNDWST10075585TER-b942
Microsoft.Web/serverfarms
OK
Operation details
nsikelelo18
Microsoft.Storage/storageAccounts
OK
Operation details
Give feedback
Tell us about your experience with deployment
Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >
Free Microsoft tutorials
Start learning today >
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 >

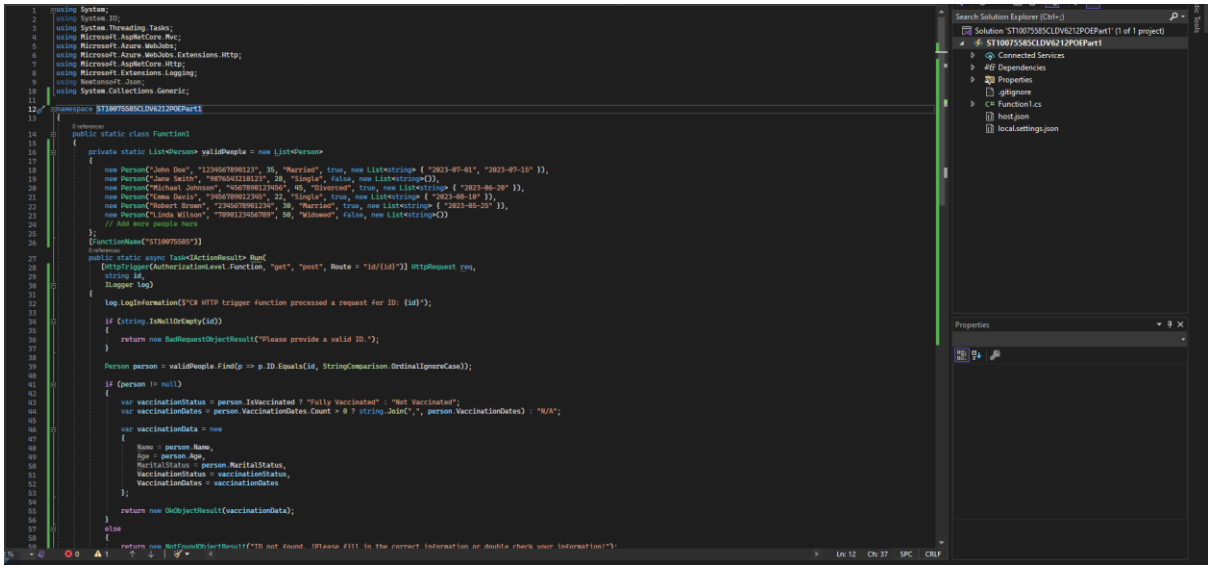

Microsoft Azure
Search resources, services, and docs (G+/)
Home
st10075585MainFunction
Function App
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Microsoft Defender for Cloud
Events (preview)
Functions
App keys
App files
Proxies
Deployment
Deployment slots
Deployment Center
Settings
Configuration
Authentication
Application Insights
Identity
Backups
Custom domains
Certificates
Networking
Browse
Refresh
Stop
Restart
Swap
Get publish profile
Reset publish profile
Download app content
Delete
Send us your feedback
Essentials
JSON View
Resource group (move) : AZ-JHB-RSG-VCKNDW-ST10075585-TER
Status : Running
Location (move) : South Africa North
Subscription (move) : ADVTECH-Tertiary Varsity College
Subscription ID : e31273bf-0dae-4395-a8b5-33f801de7c65
Tags (edit) : Add tags
URL : https://st10075585mainfunction.azurewebsites.net
Operating System : Windows
App Service Plan : ASP-AZJHBRSGVCKNDWST10075585TER-a79a (Y1: 0)
Properties : See More
Runtime version : ---
Functions
Metrics
Properties
Notifications (0)
Create
Set up local environment
Refresh
Filter by name...
Name
Trigger
Status
Monitor
st10075585
HTTP
Enabled
Invocations and more

(Leshaba, 2023)

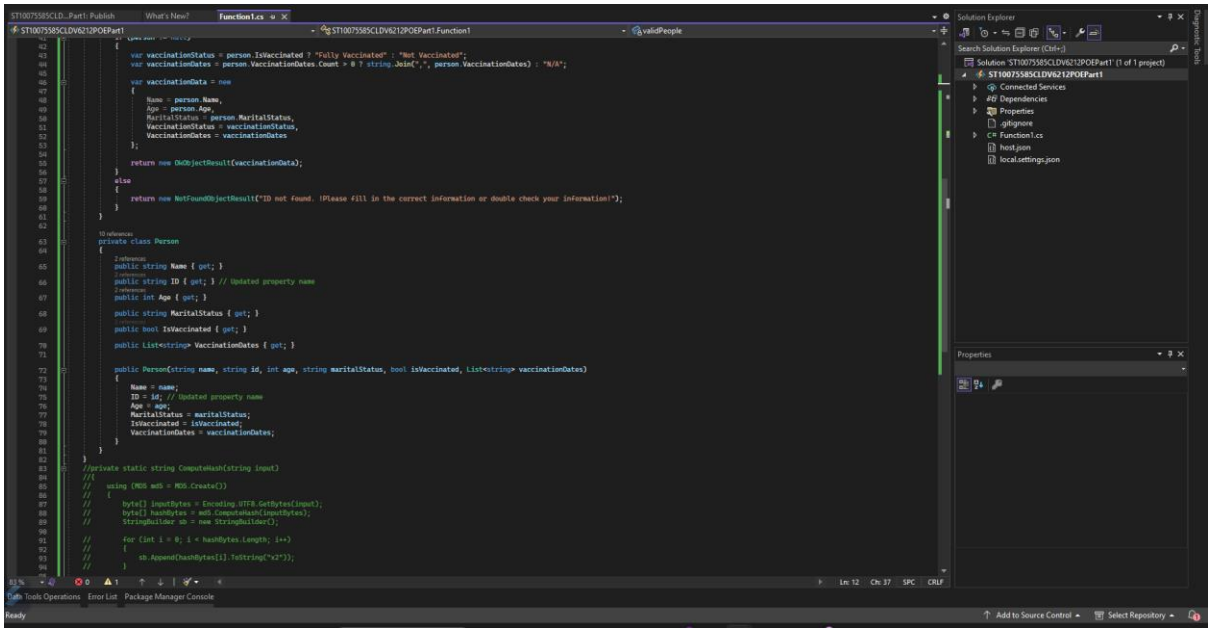
The function code:

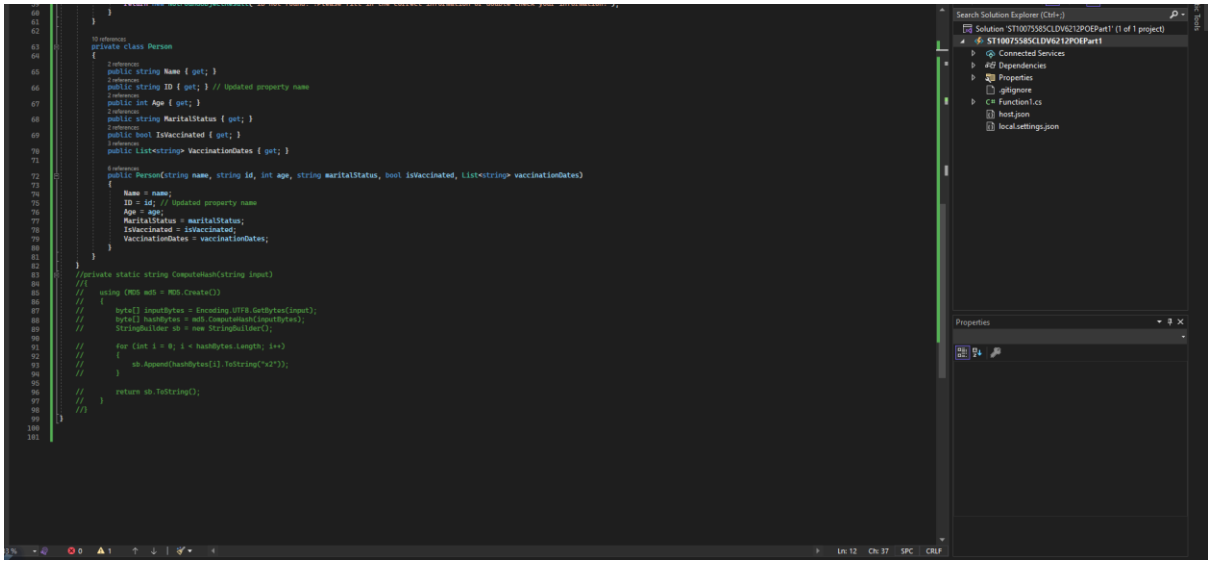
```
using System;
using System.IO;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.Logging;
using Newtonsoft.Json;
using System.Collections.Generic;

namespace ST10075585CLDV6212POEPart1
{
    public static class Function1
    {
        private static List<Person> validPeople = new List<Person>
        {
            new Person("John Doe", "1234567890123", 35, "Married", true, new
List<string> { "2023-07-01", "2023-07-15" }),
            new Person("Jane Smith", "9876543210123", 28, "Single", false, new
List<string>()),
            new Person("Michael Johnson", "4567890123456", 45, "Divorced", true,
new List<string> { "2023-06-20" }),
            new Person("Emma Davis", "3456789012345", 22, "Single", true, new
List<string> { "2023-08-10" }),
            new Person("Robert Brown", "2345678901234", 30, "Married", true, new
List<string> { "2023-05-25" }),
            new Person("Linda Wilson", "7890123456789", 50, "Widowed", false, new
List<string>())
            // Add more people here
        };
        [FunctionName("ST10075585")]
        public static async Task<IActionResult> Run(
           [HttpTrigger(AuthorizationLevel.Function, "get", "post", Route =
"id/{id}")] HttpRequest req,
            string id,
            ILogger log)
        {
            log.LogInformation($"C# HTTP trigger function processed a request for
ID: {id}");

            if (string.IsNullOrEmpty(id))
            {
                return new BadRequestObjectResult("Please provide a valid ID.");
            }

            Person person = validPeople.Find(p => p.ID.Equals(id,
StringComparison.OrdinalIgnoreCase));

            if (person != null)
            {
                var vaccinationStatus = person.IsVaccinated ? "Fully
Vaccinated" : "Not Vaccinated";
                var vaccinationDates = person.VaccinationDates.Count > 0 ?
string.Join(",", person.VaccinationDates) : "N/A";

                var vaccinationData = new
                {
                    Name = person.Name,
```

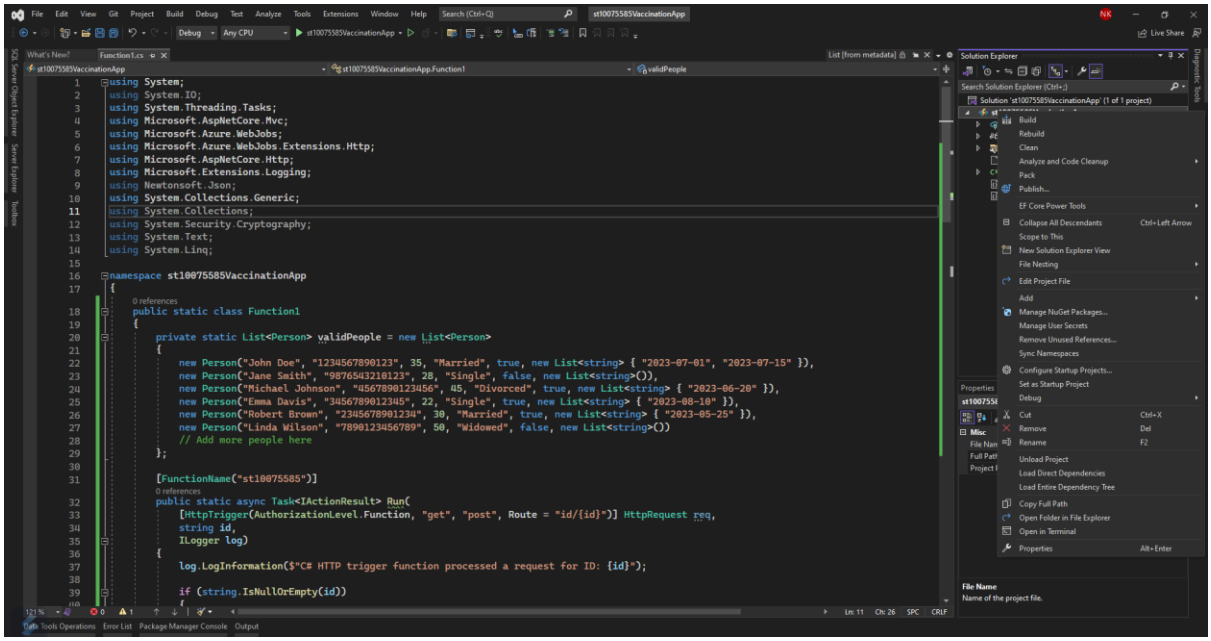
```
                    Age = person.Age,
                    MaritalStatus = person.MaritalStatus,
                    VaccinationStatus = vaccinationStatus,
                    VaccinationDates = vaccinationDates
                };

                return new OkObjectResult(vaccinationData);
            }
            else
            {
                return new NotFoundObjectResult("ID not found. !Please fill in
the correct information or double check your information!");
            }
        }

        private class Person
        {
            public string Name { get; }
            public string ID { get; } // Updated property name
            public int Age { get; }
            public string MaritalStatus { get; }
            public bool IsVaccinated { get; }
            public List<string> VaccinationDates { get; }

            public Person(string name, string id, int age, string maritalStatus,
bool isVaccinated, List<string> vaccinationDates)
            {
                Name = name;
                ID = id; // Updated property name
                Age = age;
                MaritalStatus = maritalStatus;
                IsVaccinated = isVaccinated;
                VaccinationDates = vaccinationDates;
            }
        }
    }
    //private static string ComputeHash(string input)
    //{
    //    using (MD5 md5 = MD5.Create())
    //    {
    //        byte[] inputBytes = Encoding.UTF8.GetBytes(input);
    //        byte[] hashBytes = md5.ComputeHash(inputBytes);
    //        StringBuilder sb = new StringBuilder();

    //        for (int i = 0; i < hashBytes.Length; i++)
    //        {
    //            sb.Append(hashBytes[i].ToString("x2"));
    //        }

    //        return sb.ToString();
    //    }
    //}
}
```



```
using System;
using System.IO;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.Logging;
using Newtonsoft.Json;
using System.Collections.Generic;

namespace ST10075585CLDV6212POEPart1
{
    0 references
    public static class Function1
    {
        private static List<Person> validPeople = new List<Person>
        {
            new Person("John Doe", "1234567890123", 35, "Married", true, new List<string> { "2023-07-01", "2023-07-15" }),
            new Person("Jane Smith", "9876543210123", 28, "Single", false, new List<string>()),
            new Person("Michael Johnson", "4567890123456", 45, "Divorced", true, new List<string> { "2023-06-20" }),
            new Person("Emma Davis", "3456789012345", 22, "Single", true, new List<string> { "2023-08-10" }),
            new Person("Robert Brown", "2345678901234", 38, "Married", true, new List<string> { "2023-05-25" }),
            new Person("Linda Wilson", "7890123456789", 50, "Widowed", false, new List<string>())
            // Add more people here
        };
        [FunctionName("ST10075585")]
        0 references
        public static async Task<IActionResult> Run(
            [HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = "id/{id}")] HttpRequest req,
            string id,
            ILogger log)
        {
            log.LogInformation($"C# HTTP trigger function processed a request for ID: {id}");

            if (string.IsNullOrEmpty(id))
            {
                return new BadRequestObjectResult("Please provide a valid ID.");
            }

            Person person = validPeople.Find(p => p.ID.Equals(id, StringComparison.OrdinalIgnoreCase));

            if (person != null)
            {
                var vaccinationStatus = person.IsVaccinated ? "Fully Vaccinated" : "Not Vaccinated";
                var vaccinationDates = person.VaccinationDates.Count > 0 ? string.Join(", ", person.VaccinationDates) : "N/A";

                var vaccinationData = new
                {
                    Name = person.Name,
                    Age = person.Age,
                    MaritalStatus = person.MaritalStatus,
                    VaccinationStatus = vaccinationStatus,
                    VaccinationDates = vaccinationDates
                };

                return new OkObjectResult(vaccinationData);
            }
            else
            {
                return new NotFoundObjectResult("ID not found. !Please fill in the correct information or double check your information!");
            }
        }

        10 references
        private class Person
        {
            2 references
            public string Name { get; }
            2 references
            public string ID { get; } // Updated property name
            2 references
            public int Age { get; }

            public string MaritalStatus { get; }

            public bool IsVaccinated { get; }

            public List<string> VaccinationDates { get; }


            public Person(string name, string id, int age, string maritalStatus, bool isVaccinated, List<string> vaccinationDates)
            {
                Name = name;
                ID = id; // Updated property name
                Age = age;
                MaritalStatus = maritalStatus;
                IsVaccinated = isVaccinated;
                VaccinationDates = vaccinationDates;
            }
        }
    }
    //private static string ComputeHash(string input)
    //{
    //    using (MD5 md5 = MD5.Create())
    //    {
    //        byte[] inputBytes = Encoding.UTF8.GetBytes(input);
    //        byte[] hashBytes = md5.ComputeHash(inputBytes);
    //        StringBuilder sb = new StringBuilder();

    //        for (int i = 0; i < hashBytes.Length; i++)
    //        {
    //            sb.Append(hashBytes[i].ToString("x2"));
    //        }
```

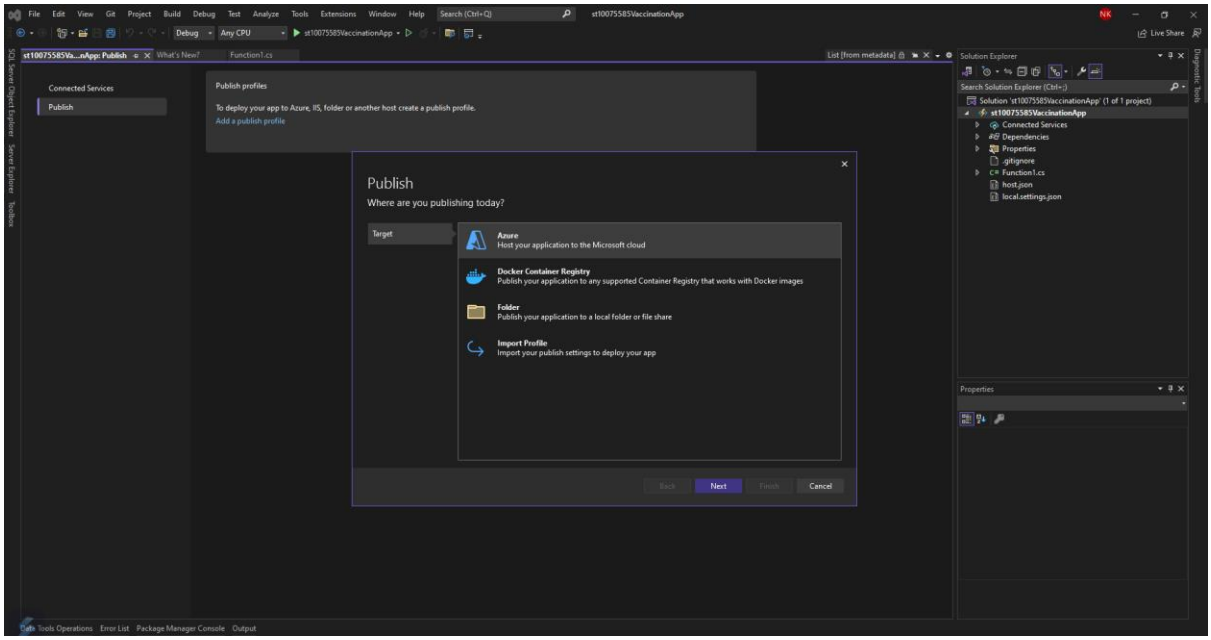


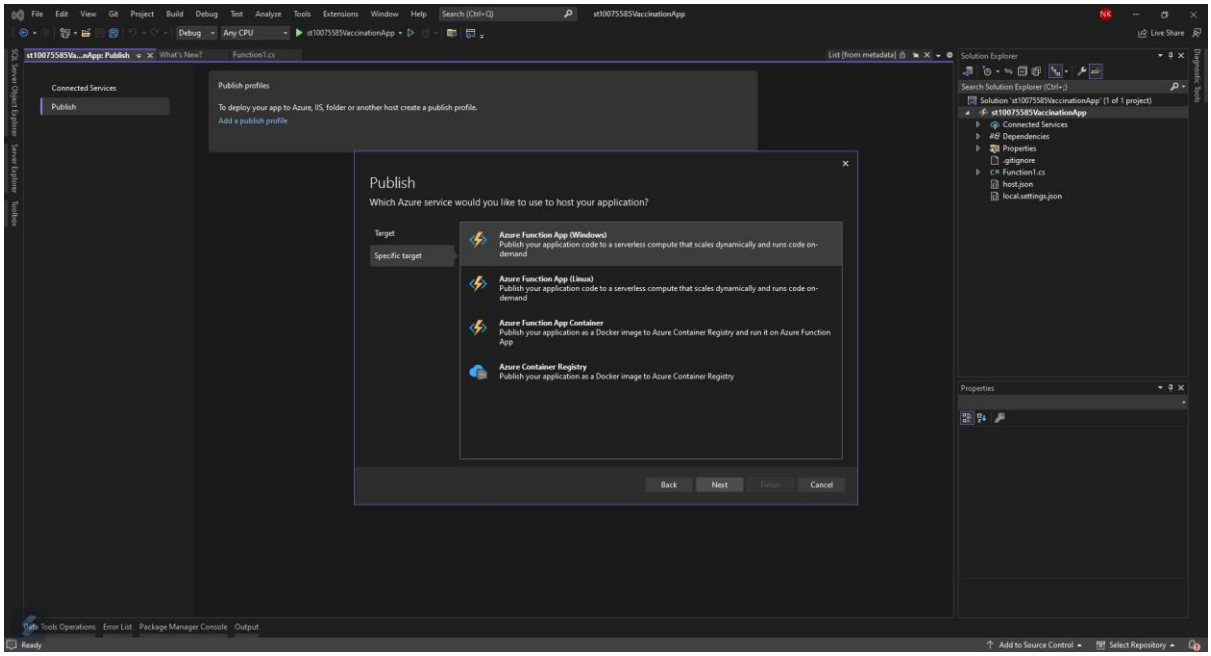


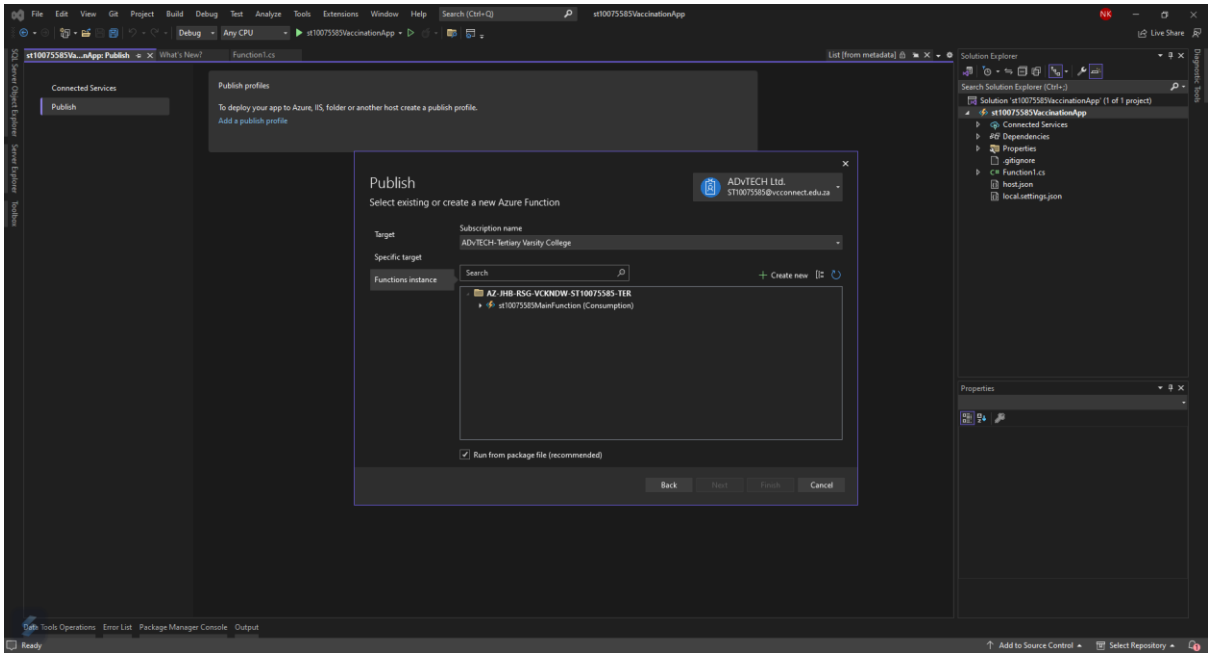


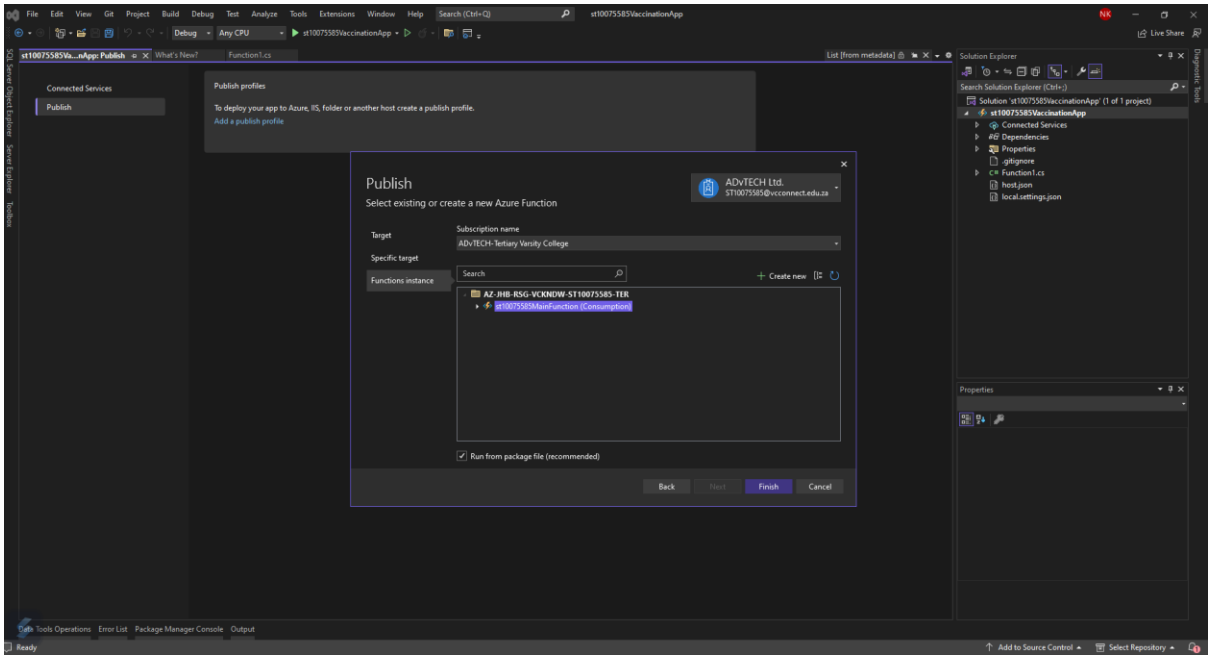
```
using System;
using System.IO;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.Logging;
using Newtonsoft.Json;
using System.Collections.Generic;
using System.Collections;
using System.Security.Cryptography;
using System.Text;
using System.Linq;

namespace st10075585VaccinationApp
{
    0 references
    public static class Function1
    {
        private static List<Person> validPeople = new List<Person>
        {
            new Person("John Doe", "1234567890123", 35, "Married", true, new List<string> { "2023-07-01", "2023-07-15" }),
            new Person("Jane Smith", "9876543210123", 28, "Single", false, new List<string>()),
            new Person("Michael Johnson", "4567890123456", 45, "Divorced", true, new List<string> { "2023-06-20" }),
            new Person("Emma Davis", "3456789012345", 22, "Single", true, new List<string> { "2023-08-10" }),
            new Person("Robert Brown", "2345678901234", 30, "Married", true, new List<string> { "2023-05-25" }),
            new Person("Linda Wilson", "7890123456789", 50, "Widowed", false, new List<string>())
            // Add more people here
        };

        [FunctionName("st10075585")]
        0 references
        public static async Task<IActionResult> Run(
            [HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = "id/{id}")] HttpRequest req,
            string id,
            ILogger log)
        {
            log.LogInformation($"C# HTTP trigger function processed a request for ID: {id}");

            if (string.IsNullOrEmpty(id))
```

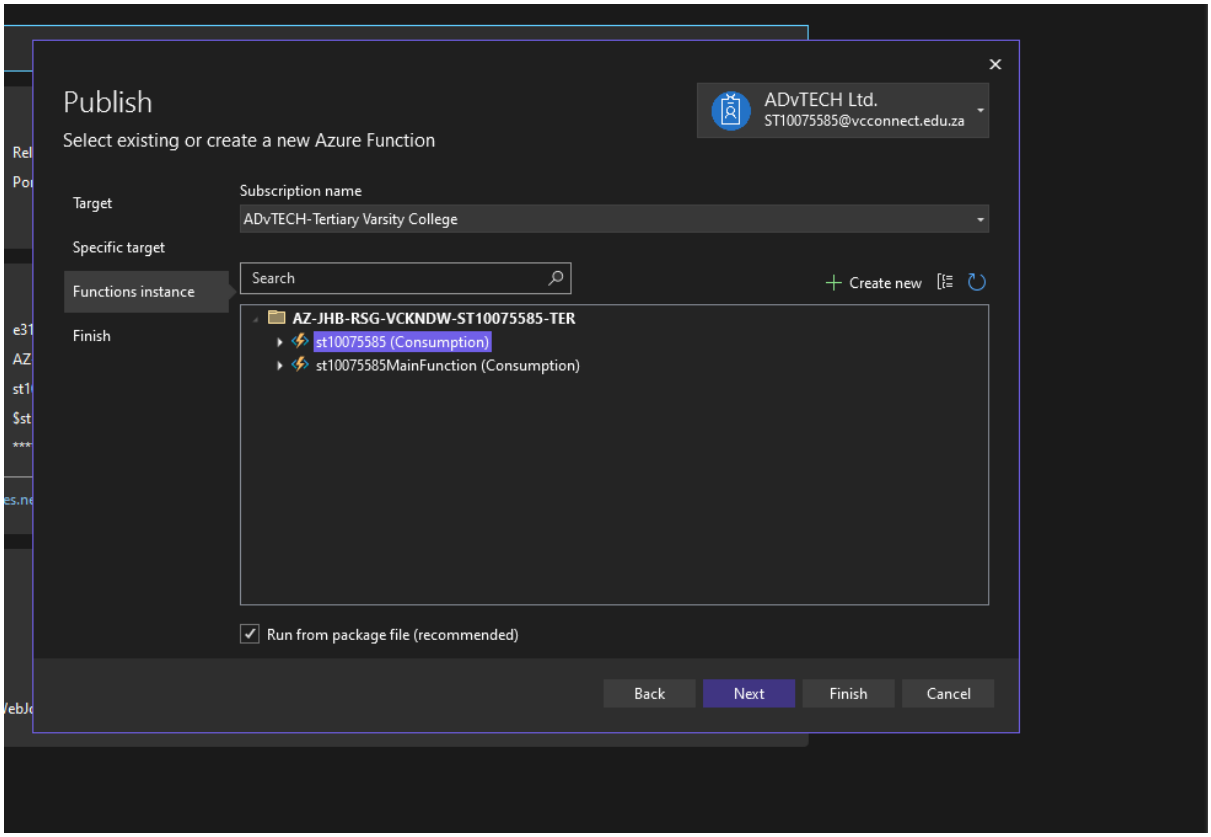
The steps you took to deploy/ publish the function (there must be several steps):

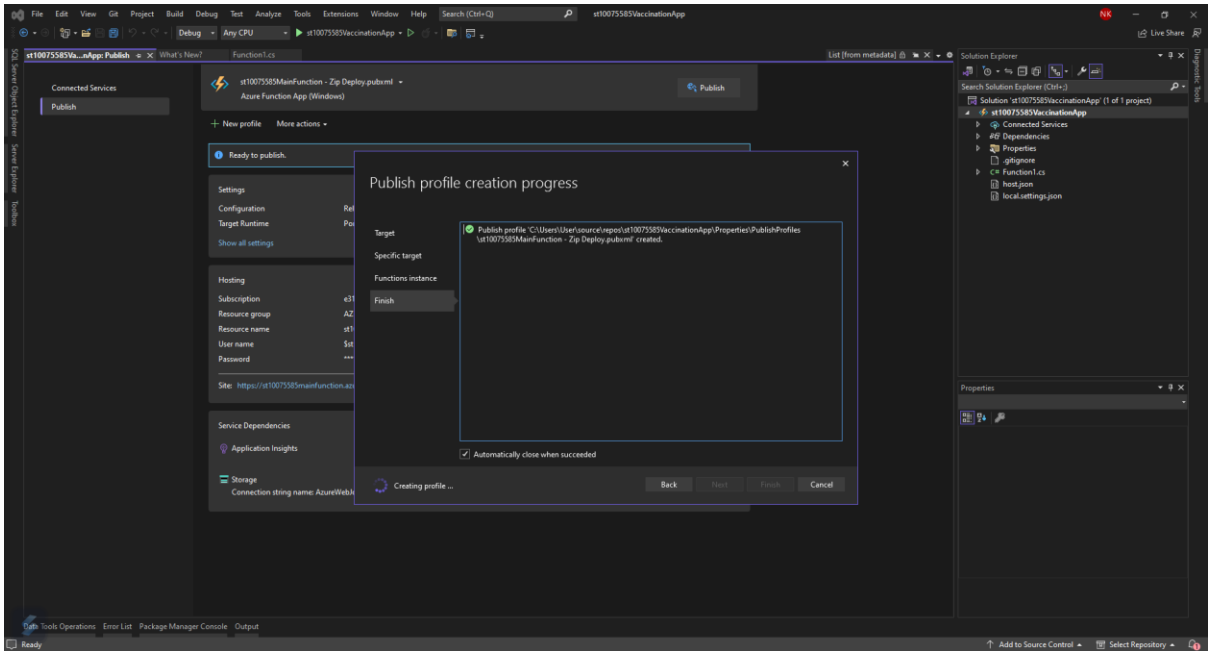


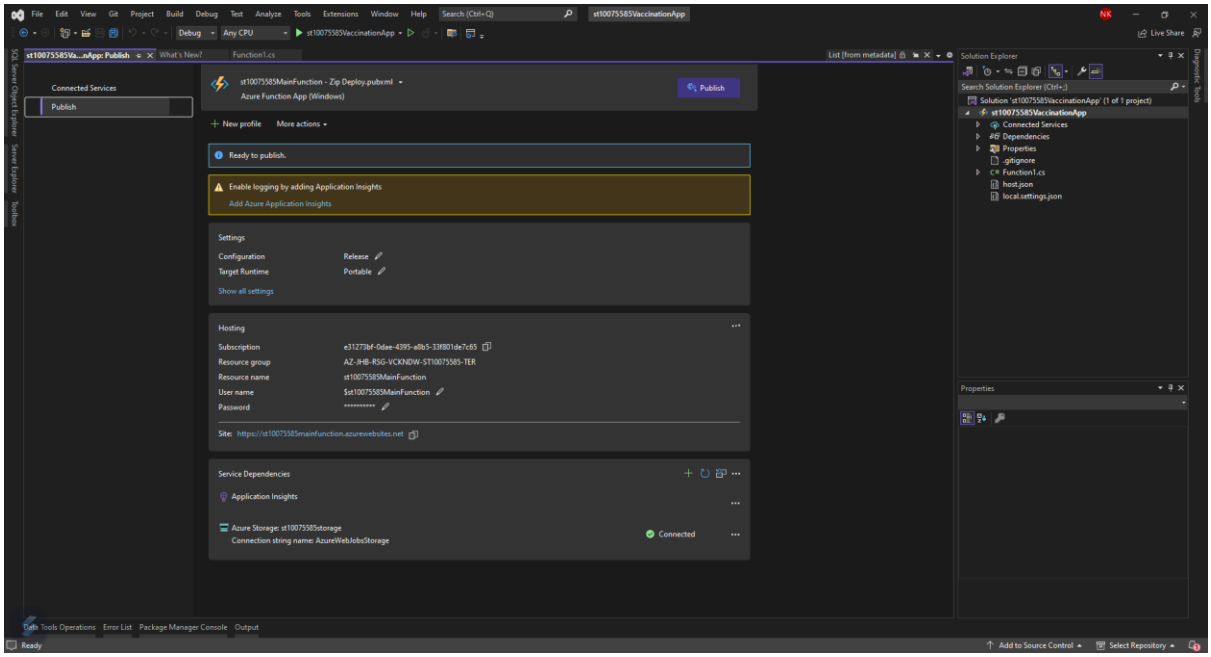

Publish
Which Azure service would you like to use to host your application?
Target
Specific target
Azure Function App (Windows)
Publish your application code to a serverless compute that scales dynamically and runs code on-demand
Azure Function App (Linux)
Publish your application code to a serverless compute that scales dynamically and runs code on-demand
Azure Function App Container
Publish your application as a Docker image to Azure Container Registry and run it on Azure Function App
Azure Container Registry
Publish your application as a Docker image to Azure Container Registry
Back
Next
Finish
Cancel

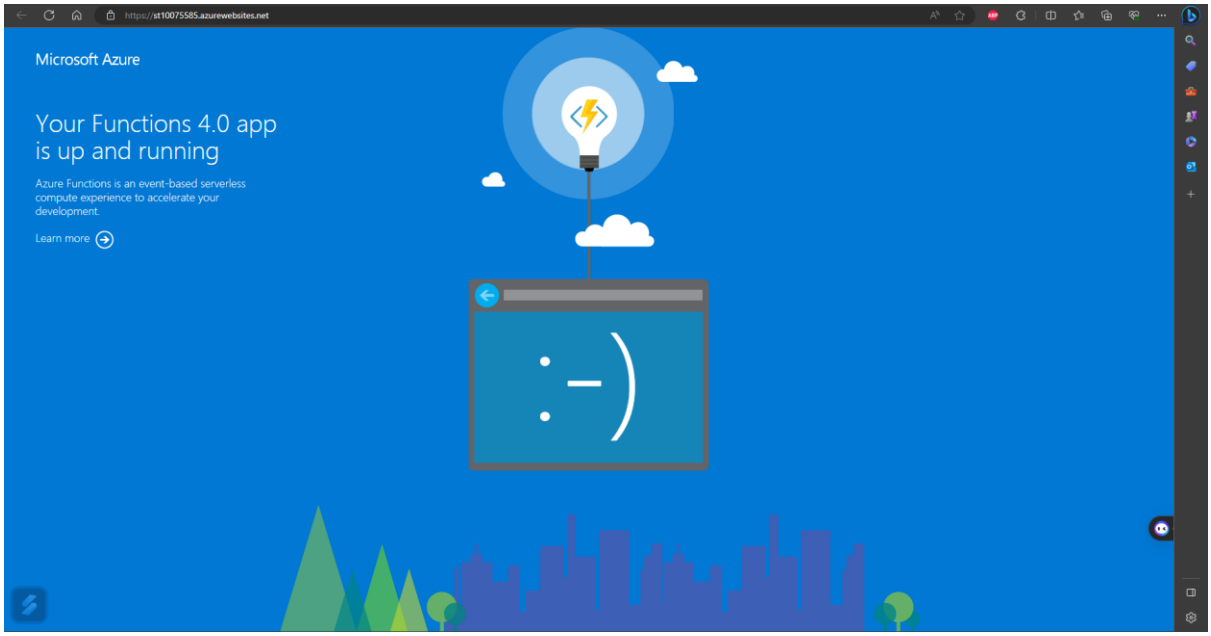

Publish
Select existing or create a new Azure Function
ADvTECH Ltd.
Target
Subscription name
ADvTECH-Tertiary Varsity College
Specific target
Functions instance
Search
Create new
AZ-JHB-RSG-VCKNDW-ST10075585-TER
st10075585MainFunction (Consumption)
Run from package file (recommended)
Back
Next
Finish
Cancel

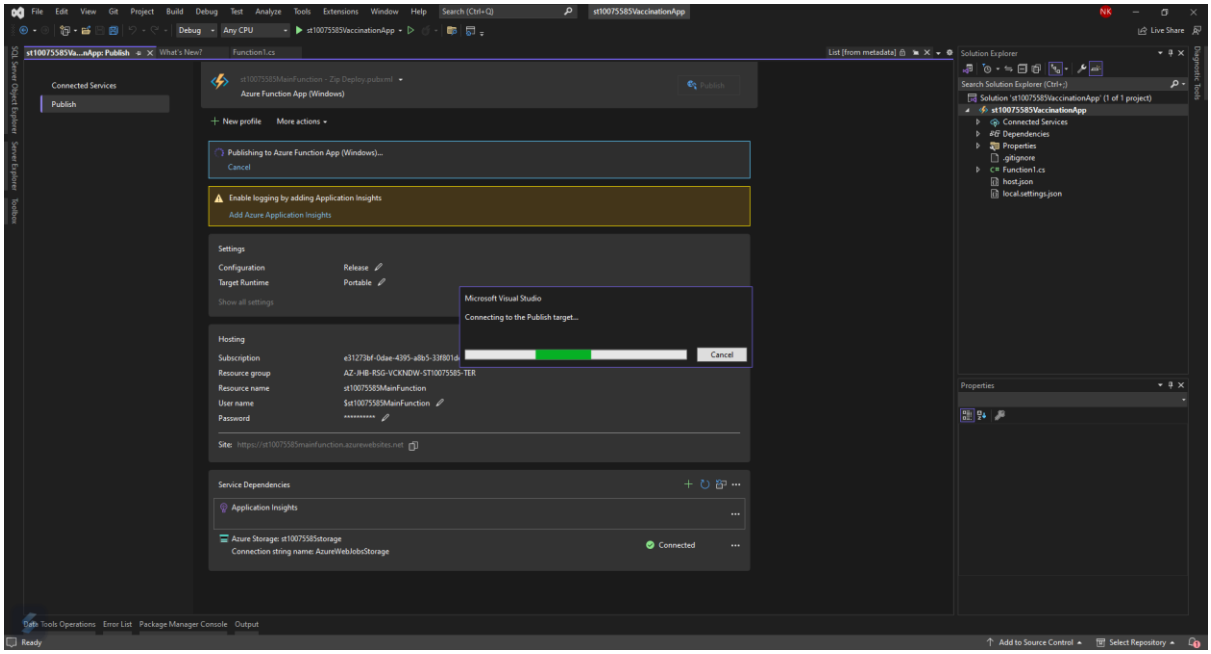

Publish
Select existing or create a new Azure Function
ADvTECH Ltd.
ST10075585@vcconnect.edu.za
Target
Subscription name
ADvTECH-Tertiary Varsity College
Specific target
Functions instance
Search
Create new
AZ-JHB-RSG-VCKNDW-ST10075585-TER
st10075585MainFunction (Consumption)
Run from package file (recommended)
Back
Next
Finish
Cancel

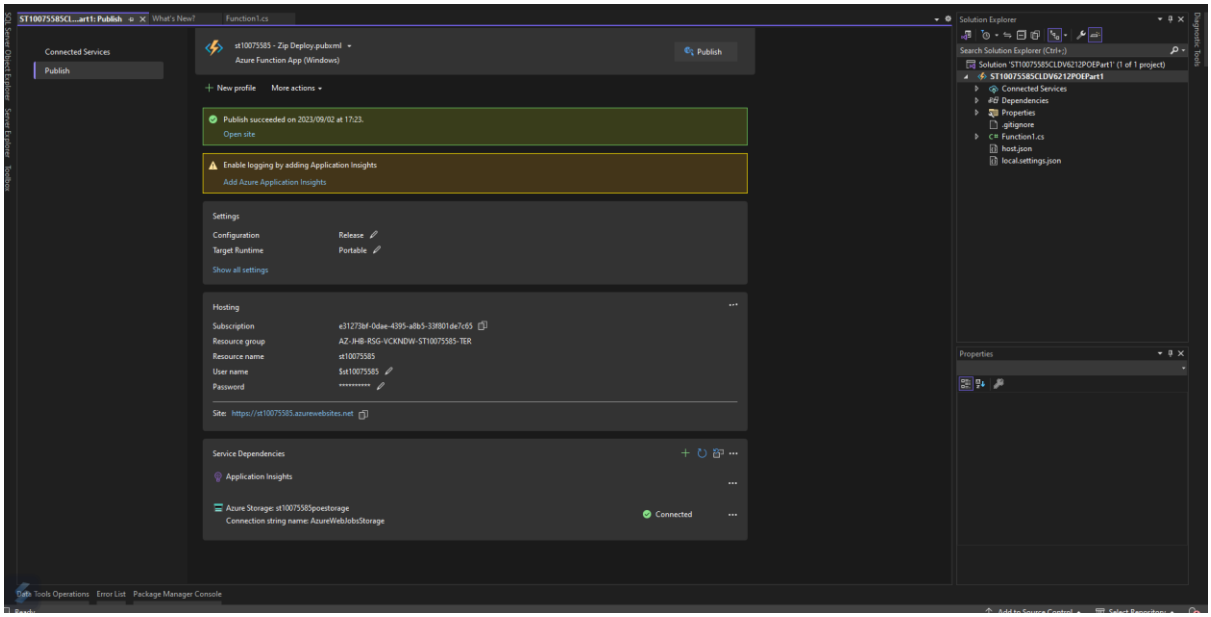

Publish
Select existing or create a new Azure Function
ADvTECH Ltd.
ST10075585@vcconnect.edu.za
Target
Subscription name
ADvTECH-Tertiary Varsity College
Specific target
Functions instance
Search
Create new
Finish
AZ-JHB-RSG-VCKNDW-ST10075585-TER
st10075585 (Consumption)
st10075585MainFunction (Consumption)
Run from package file (recommended)
Back
Next
Finish
Cancel


Publish profile creation progress
Target
Specific target
Functions instance
Finish
Publish profile 'C:\Users\User\source\repos\st10075585VaccinationApp\Properties\PublishProfiles\st10075585MainFunction - Zip Deploy.pubxml' created.
Automatically close when succeeded
Creating profile ...
Back
Next
Finish
Cancel
Publish
Ready to publish.
Connected Services
Publish
New profile
More actions
Settings
Configuration
Target Runtime
Show all settings
Hosting
Subscription
Resource group
Resource name
User name
Password
Service Dependencies
Application Insights
Storage
Solution Explorer
Properties


Publish
Ready to publish.
Enable logging by adding Application Insights
Add Azure Application Insights
Settings
Configuration
Release
Target Runtime
Portable
Show all settings
Hosting
Subscription
e31273bf-0dae-4395-a8b5-33f801de7c65
Resource group
AZ-JHB-RSG-VCKNDW-ST10075585-TER
Resource name
st10075585MainFunction
User name
$st10075585MainFunction
Password
Site
https://st10075585mainfunction.azurewebsites.net
Service Dependencies
Application Insights
Azure Storage: st10075585storage
Connection string name: AzureWebJobsStorage
Connected
Solution Explorer
Properties


https://st10075585.azurewebsites.net
Microsoft Azure
Your Functions 4.0 app is up and running
Azure Functions is an event-based serverless compute experience to accelerate your development.
Learn more


Connected Services
Publish
Azure Function App (Windows)
New profile
More actions
Publishing to Azure Function App (Windows)...
Cancel
Enable logging by adding Application Insights
Add Azure Application Insights
Settings
Configuration
Release
Target Runtime
Portable
Show all settings
Hosting
Subscription
Resource group
AZ-IHB-RSG-VCKNDW-ST10075585-TER
Resource name
st10075585MainFunction
User name
$st10075585MainFunction
Password
Site:
Service Dependencies
Application Insights
Azure Storage: st10075585storage
Connection string name: AzureWebJobsStorage
Connected
Microsoft Visual Studio
Connecting to the Publish target...
Cancel
Solution Explorer
Solution 'st10075585VaccinationApp' (1 of 1 project)
st10075585VaccinationApp
Connected Services
Dependencies
Properties
.gitignore
Function1.cs
host.json
local.settings.json
Properties
Ready
Add to Source Control
Select Repository

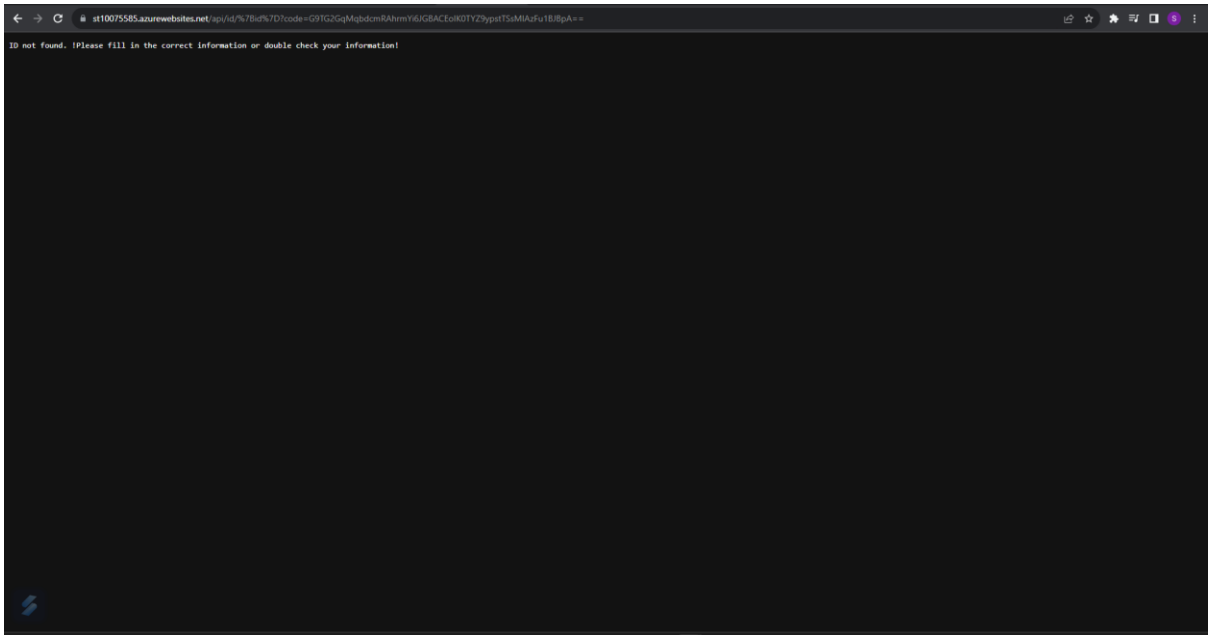


(Leshaba, 2023)

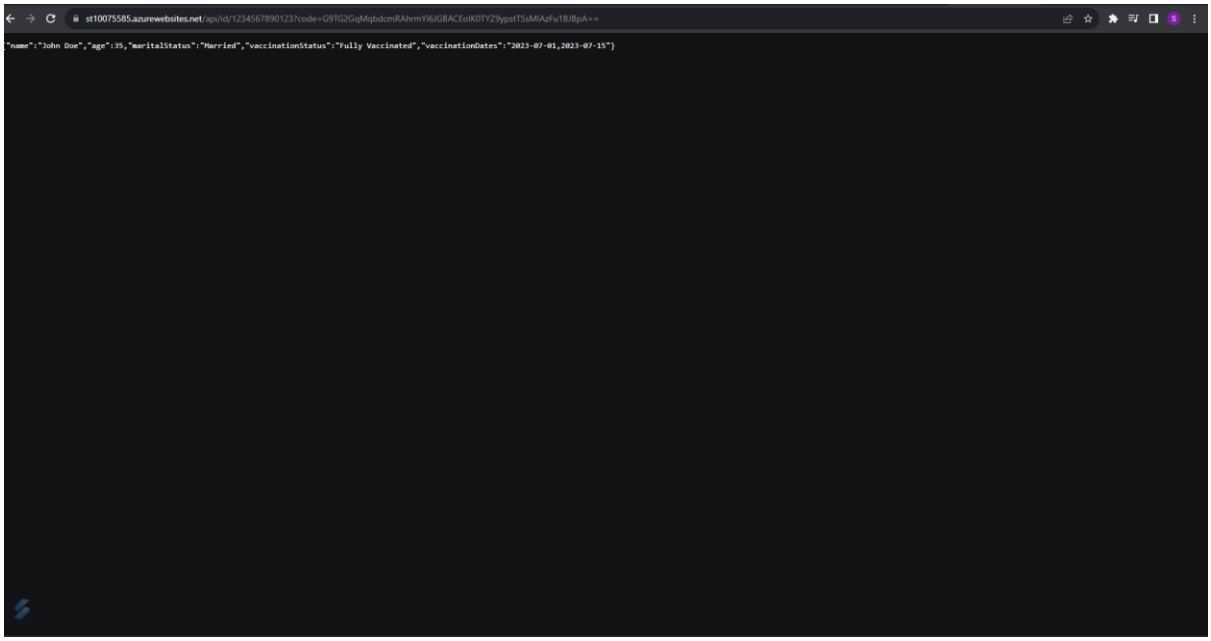
The function working in a web browser with the URL clearly visible (which must start with your student number):

https://st10075585.azurewebsites.net/api/id/{id}?code=G9TG2GqMqbdcmRAhrmYi6JGBACEoIK0TYZ9ypstTSsMIAzFu1BJBpA==

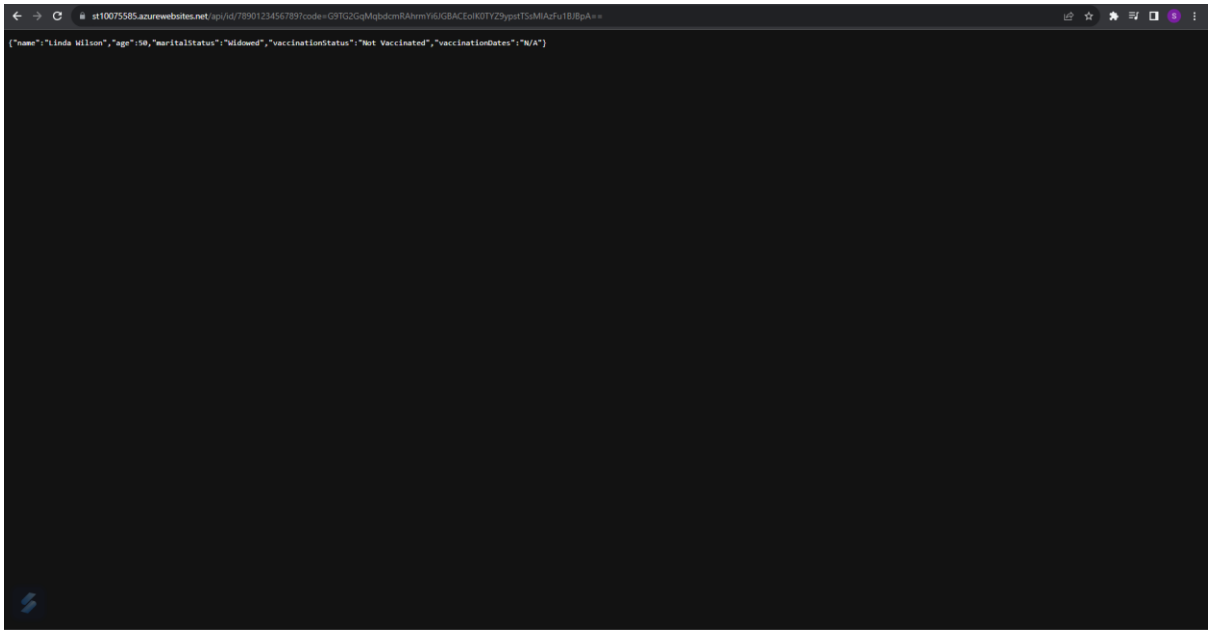
(ST10075585, 2023)



VACCINATED



NOT VACCINATED:



Bibliography

ATLASSIAN, 2023. *MTBF, MTTR, MTTA, and MTTF.* [Online]
Available at: https://www.atlassian.com/incident-management/kpis/common-metrics#:~:text=MTBF%20(mean%20time%20between%20failures,the%20more%20reliable%20the%20system.
[Accessed 20 August 2023].

Brunskill, V.-L., 2023. *polyglot persistence.* [Online]
Available at: https://www.techtarget.com/searchapparchitecture/definition/polyglot-persistence
[Accessed 20 August 2023].

Cloud Native Glossary Authors, 2022. *Immutable Infrastructure.* [Online]
Available at: https://glossary.cncf.io/immutable-infrastructure/#:~:text=Immutable%20Infrastructure%20refers%20to%20computer,changes%20in%20the%20first%20place.
[Accessed 20 August 2023].

Eckert, M., 2012. *Which approach is better: Frequent small updates, or occasional large ones?.* [Online]
Available at: https://gamedev.stackexchange.com/questions/25167/which-approach-is-better-frequent-small-updates-or-occasional-large-ones
[Accessed 28 August 2023].

GeeksforGeeks, 2023. *Introduction of Process Synchronization.* [Online]
Available at: https://www.geeksforgeeks.org/introduction-of-process-synchronization/
[Accessed 28 August 2023].

Glossary, G., n.d. *Scalability.* [Online]
Available at: https://www.gartner.com/en/information-technology/glossary/scalability
[Accessed 20 August 2023].

Harris, C., 2023 . *Microservices vs. monolithic architecture.* [Online]
Available at: https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith
[Accessed 3 August 2023].

IBM, 2023. *Asynchronous processing.* [Online]
Available at: https://www.ibm.com/docs/en/cics-ts/5.4?topic=intercommunication-asynchronous-processing
[Accessed 28 August 2023].

Kemp, P., 2023. *Snowflake Servers.* [Online]
Available at: https://kemptechnologies.com/resources/glossary/snowflake-servers#:~:text=Snowflake%20servers%20run%20a%20mission,operating%20system%20and%20application%20server.
[Accessed 20 August 2023].

Leshaba, I., 2023. *Function App set up.mp4.* [Online]
Available at: https://drive.google.com/file/d/1nDWcNudJoI2aoD7PRC71ZAk__Jh_oW3Y/view
[Accessed 20 August 2023].

Leshaba, I., 2023. *HTTP Trigger Tutorial.mp4.* [Online]
Available at: https://drive.google.com/file/d/1Djnw-1KBmRToPnJG_JvyXpUlyqJoEEL7/view
[Accessed 20 August 2023].

Leshaba, I., 2023. *Queue Storage Console App Basics.mp4,* s.l.: IIE Varsity College.

Leshaba, I., 2023. *Queue Storage With Azure SQL Database'.mp4,* s.l.: IIE varsity College.

LinkedIn, 2023. *What are some common challenges or pitfalls of using a paper-based or manual maintenance work order system?.* [Online]
Available at: https://www.linkedin.com/advice/0/what-some-common-challenges-pitfalls-using#:~:text=A%20manual%20system%20is%20an,paper%20waste%20and%20storage%20space.
[Accessed 20 August 2023].

Lutkevich, B., 2021. *relational database.* [Online]
Available at: https://www.techtarget.com/searchdatamanagement/definition/relational-database
[Accessed 20 August 2023].

Lynn, T. et al., 2016. *CLOUDLIGHTNING: A Framework for a Self-organising and Self-managing Heterogeneous Cloud.* [Online]
Available at: https://www.scitepress.org/Link.aspx?doi=10.5220/0005921503330338
[Accessed 20 August 2023].

madhav_mohan, 2023. *Scalability and Elasticity in Cloud Computing.* [Online]
Available at: https://www.geeksforgeeks.org/scalability-and-elasticity-in-cloud-computing/
[Accessed 20 August 2023].

Microsoft, 2023. *Getting started with Azure Functions.* [Online]
Available at: https://learn.microsoft.com/en-us/azure/azure-functions/functions-get-started?pivots=programming-language-csharp
[Accessed 1 October 2023].

Microsoft, 2023. *Quickstart: Azure Queue Storage client library for .NET.* [Online]
Available at: https://learn.microsoft.com/en-us/azure/storage/queues/storage-quickstart-queues-dotnet?tabs=connection-string%2Croles-azure-portal%2Cenvironment-variable-windows%2Csign-in-azure-cli
[Accessed 1 October 2023].

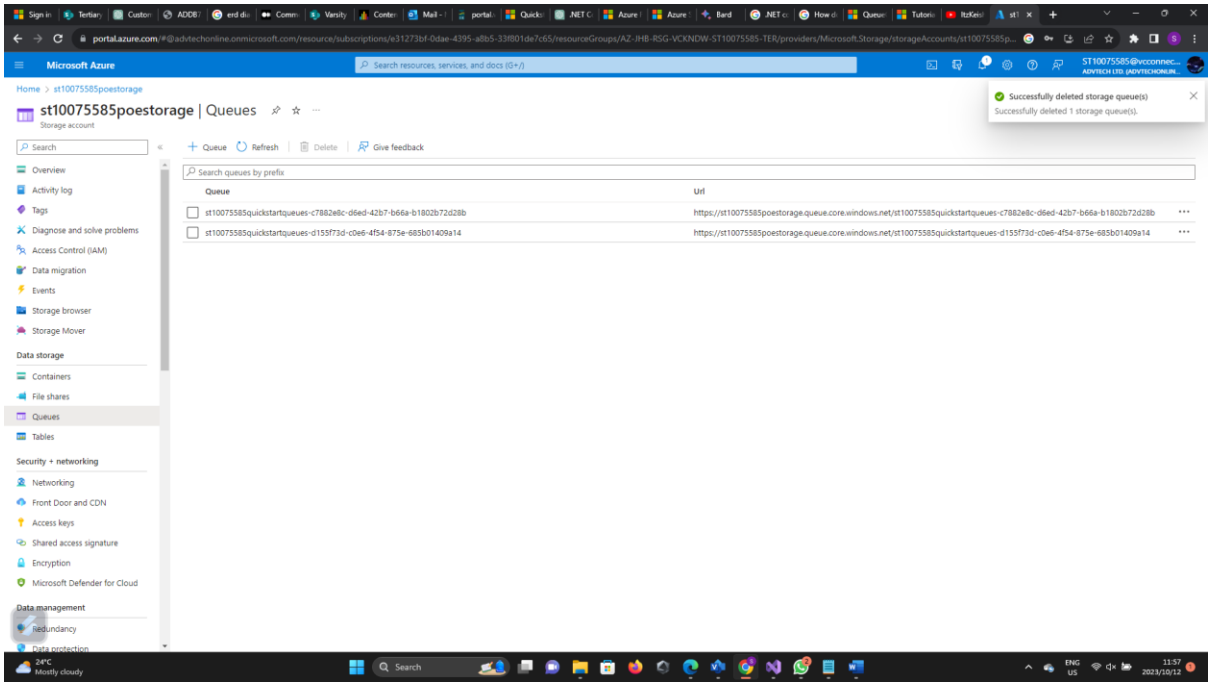
Puviani, M. & Frei, R., 2013. *Self-management for cloud computing.* [Online]
Available at: https://ieeexplore.ieee.org/document/6661855
[Accessed 20 August 2023].

reddit, 2011. *Which approach is better: Frequent small updates, or occasional large ones?.* [Online]
Available at:
https://www.reddit.com/r/gamedev/comments/qmew0/which_approach_is_better_frequent_small_updates/
[Accessed 29 August 2023].

ST10075585, N. K., 2023. *Nsikelelo Kumalo's Function.* [Online]
Available at:
https://st10075585.azurewebsites.net/api/id/{id}?code=G9TG2GqMqbdcmRAhrmYi6JGBACEoIK0TYZ9ypstTSsMIAzFu1BJBpA==
[Accessed 2 September 2023].

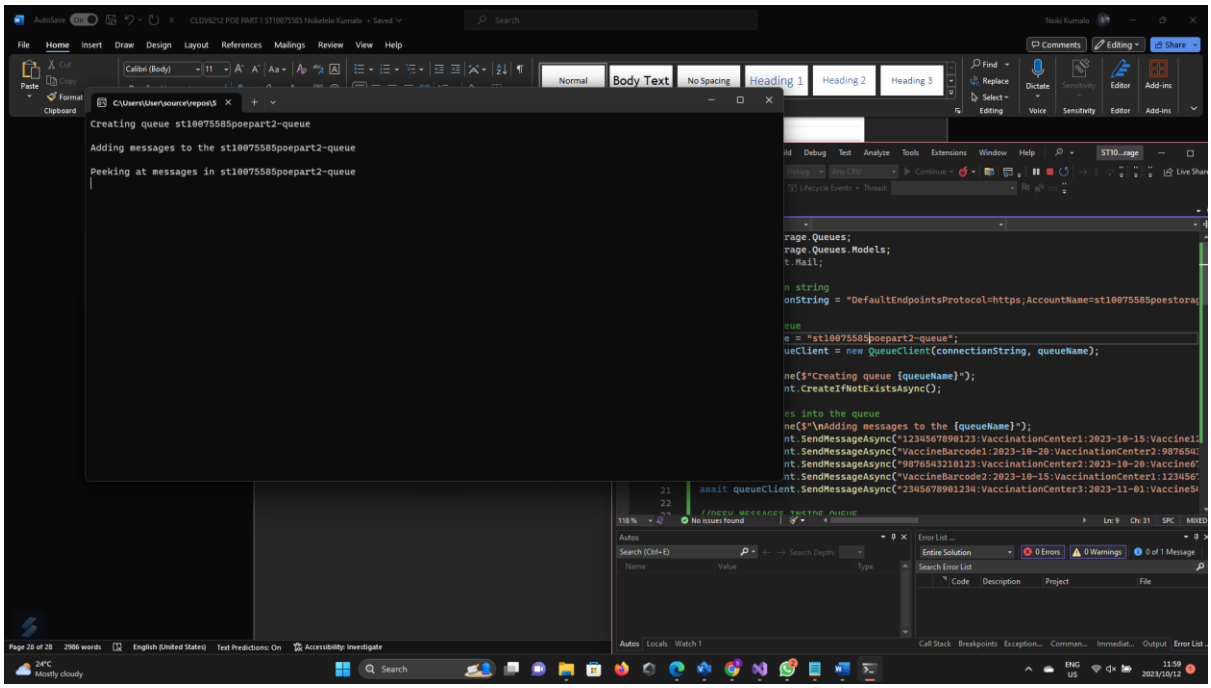
PART 2

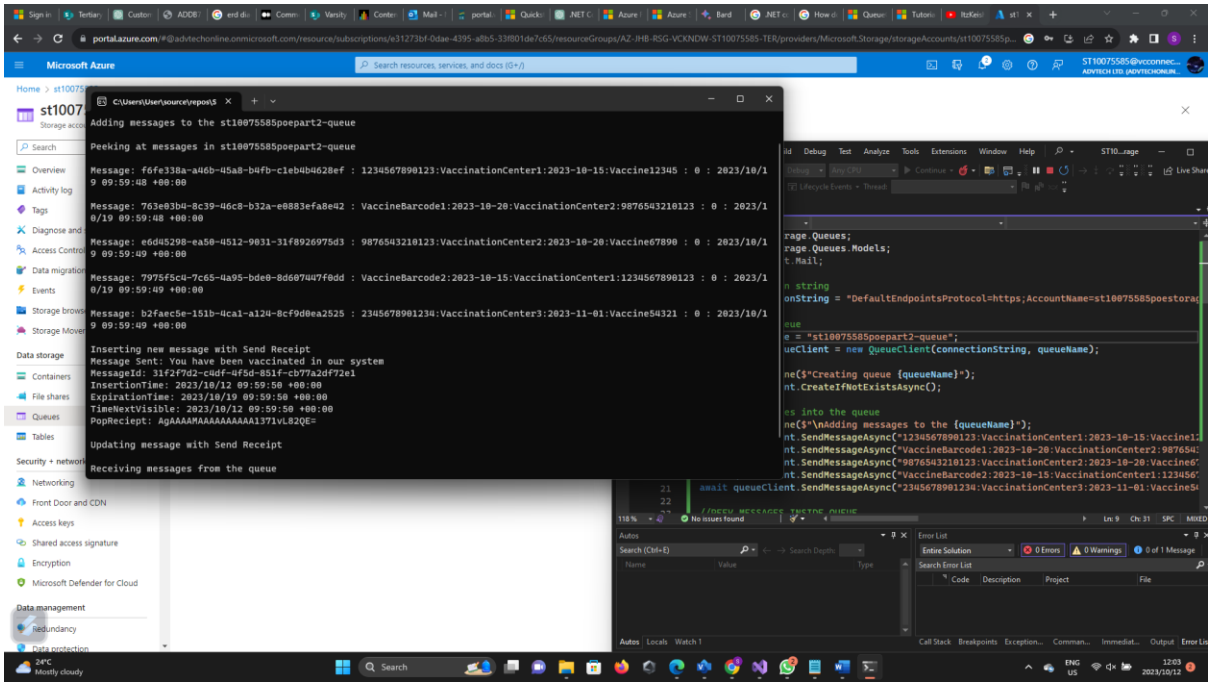
A. Write a .NET core console application to put messages into a data storage queue.

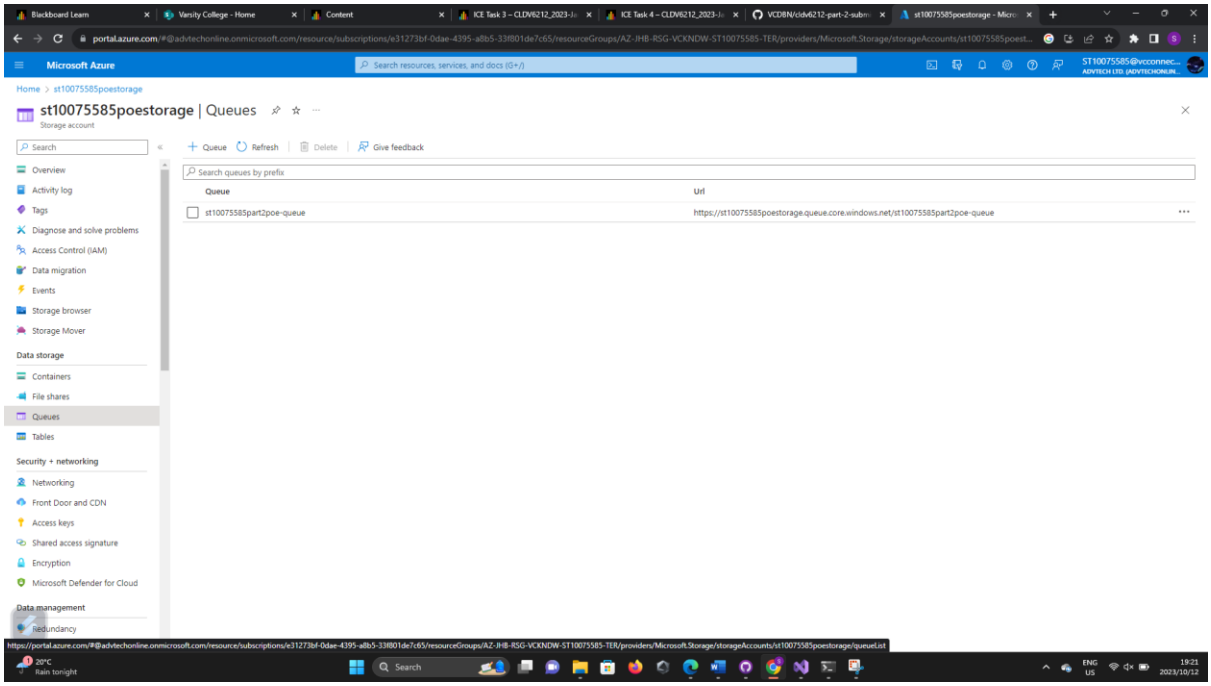


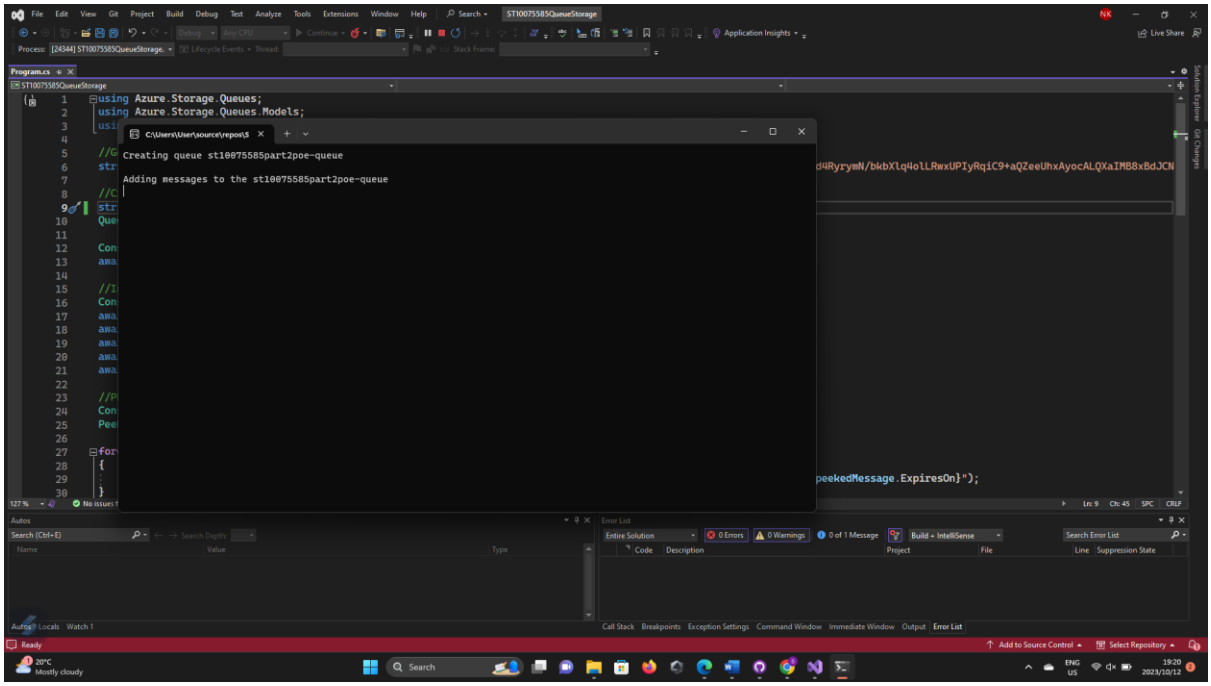
(Microsoft, 2023)

Storage queue is created in Azure:

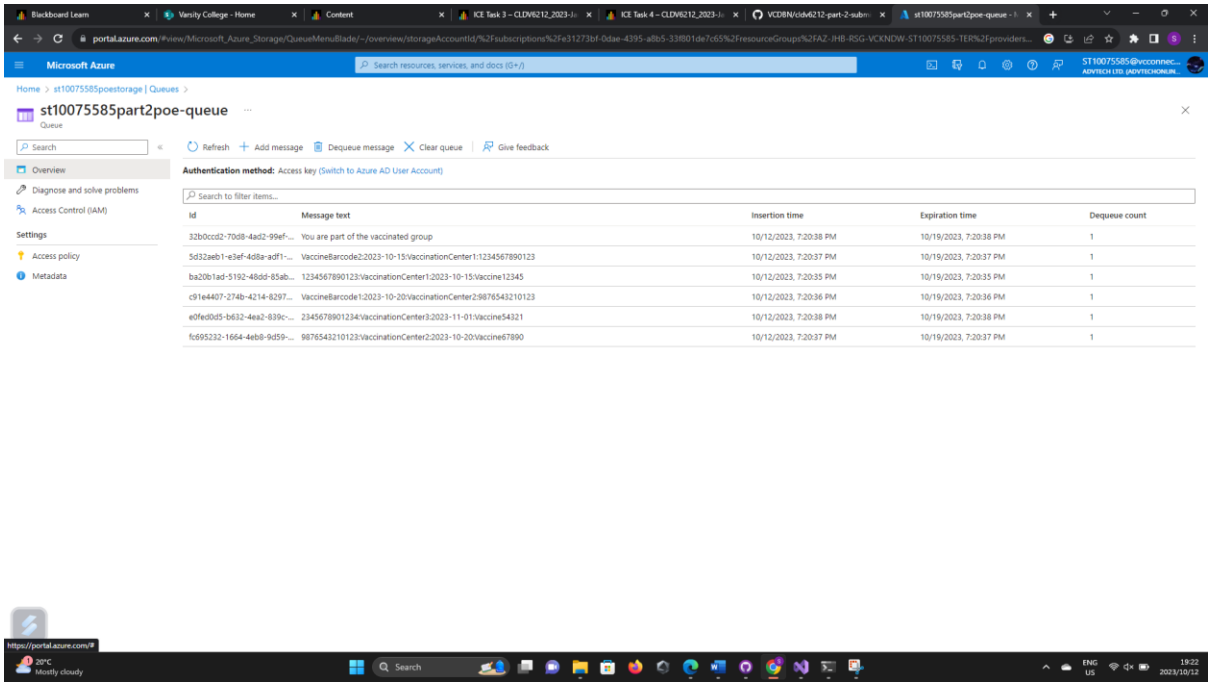



Adding messages to the st10075585poepart2-queue
Peeking at messages in st10075585poepart2-queue
Message: f6fe338a-a46b-45a8-b4fb-c1eb4b4628ef : 1234567890123:VaccinationCenter1:2023-10-15:Vaccine12345 : 0 : 2023/10/19 09:59:48 +00:00
Message: 763e03b4-8c39-46c8-b32a-e0883efa8e42 : VaccineBarcode1:2023-10-20:VaccinationCenter2:9876543210123 : 0 : 2023/10/19 09:59:48 +00:00
Message: e6d45298-ea58-4512-9031-31f8926975d3 : 9876543210123:VaccinationCenter2:2023-10-20:Vaccine67890 : 0 : 2023/10/19 09:59:49 +00:00
Message: 7975f5c4-7c65-4a95-bde0-8d607447f0dd : VaccineBarcode2:2023-10-15:VaccinationCenter1:1234567890123 : 0 : 2023/10/19 09:59:49 +00:00
Message: b2faec5e-151b-4ca1-a124-8cf9d0ea2525 : 2345678901234:VaccinationCenter3:2023-11-01:Vaccine54321 : 0 : 2023/10/19 09:59:49 +00:00
Inserting new message with Send Receipt
Message Sent: You have been vaccinated in our system
MessageId: 31f2f7d2-c4df-4f5d-851f-cb77a2df72e1
InsertionTime: 2023/10/12 09:59:50 +00:00
ExpirationTime: 2023/10/19 09:59:50 +00:00
TimeNextVisible: 2023/10/12 09:59:50 +00:00
PopReciept: AgAAAAMAAAAAAAAAA1371vL82QE=
Updating message with Send Receipt
Receiving messages from the queue

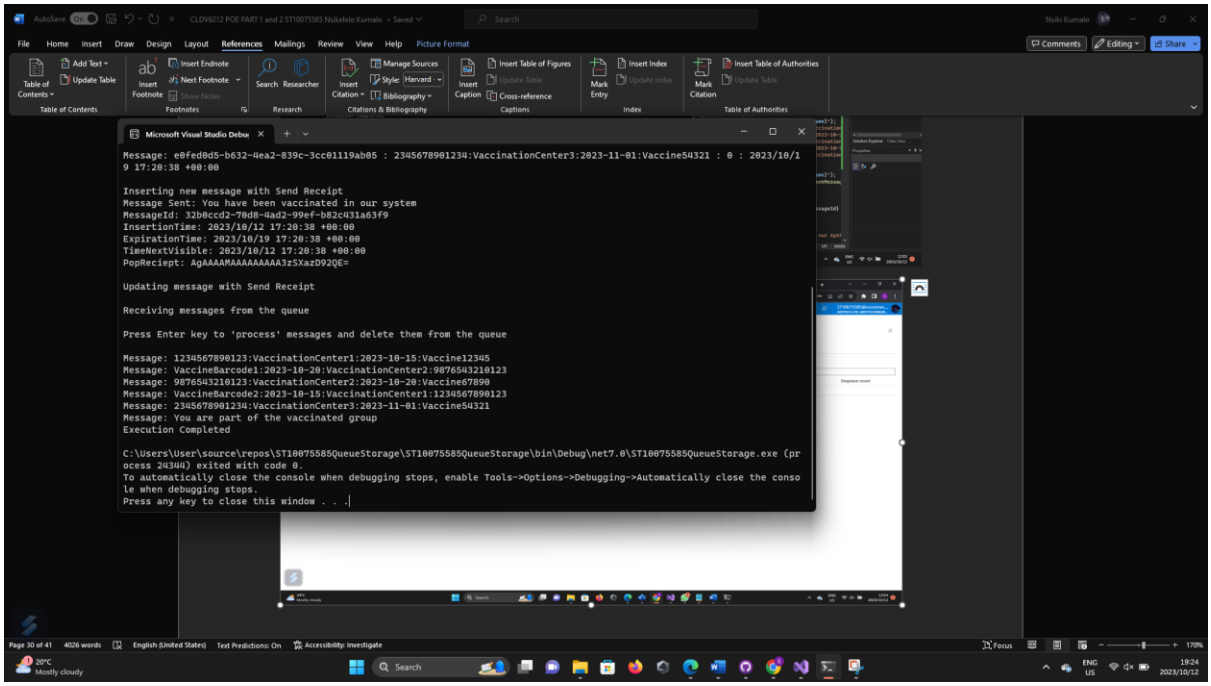

st10075585poestorage | Queues
Storage account
Queue
Refresh
Delete
Give feedback
Search queues by prefix
Queue
Url
st10075585part2poe-queue
https://st10075585poestorage.queue.core.windows.net/st10075585part2poe-queue

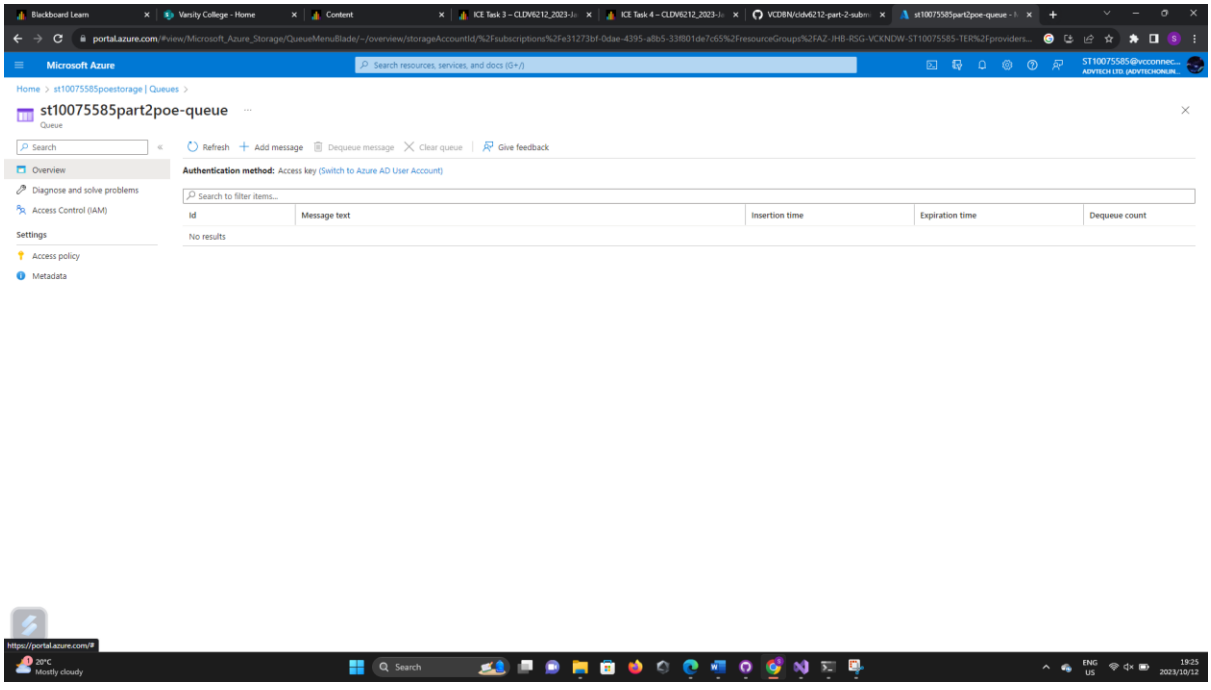


5 messages in Azure storage queue from Azure portal:

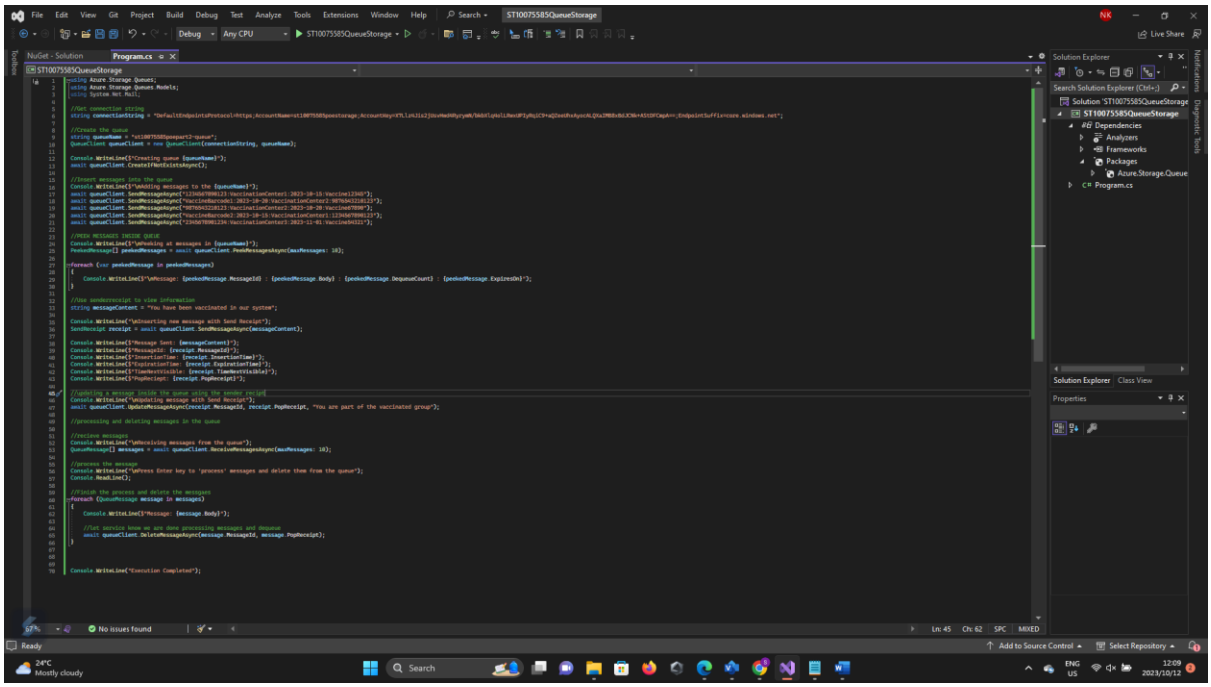


(Microsoft, 2023)


Microsoft Visual Studio Debug
Message: e0fed0d5-b632-4ea2-839c-3cc01119ab05 : 2345678901234:VaccinationCenter3:2023-11-01:Vaccine54321 : 0 : 2023/10/19 17:20:38 +00:00
Inserting new message with Send Receipt
Message Sent: You have been vaccinated in our system
MessageId: 32b0ccd2-70d8-4ad2-99ef-b82c431a63f9
InsertionTime: 2023/10/12 17:20:38 +00:00
ExpirationTime: 2023/10/19 17:20:38 +00:00
TimeNextVisible: 2023/10/12 17:20:38 +00:00
PopReciept: AgAAAAMAAAAAAAAA3zSXazD92QE=
Updating message with Send Receipt
Receiving messages from the queue
Press Enter key to 'process' messages and delete them from the queue
Message: 1234567890123:VaccinationCenter1:2023-10-15:Vaccine12345
Message: VaccineBarcode1:2023-10-20:VaccinationCenter2:9876543210123
Message: 9876543210123:VaccinationCenter2:2023-10-20:Vaccine67890
Message: VaccineBarcode2:2023-10-15:VaccinationCenter1:1234567890123
Message: 2345678901234:VaccinationCenter3:2023-11-01:Vaccine54321
Message: You are part of the vaccinated group
Execution Completed
C:\Users\User\source\repos\ST10075585QueueStorage\ST10075585QueueStorage\bin\Debug\net7.0\ST10075585QueueStorage.exe (process 24344) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .


Microsoft Azure
st10075585part2poe-queue
Queue
Refresh
Add message
Dequeue message
Clear queue
Give feedback
Authentication method: Access key (Switch to Azure AD User Account)
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Access policy
Metadata
Id
Message text
Insertion time
Expiration time
Dequeue count
No results

Code listing of the console application:

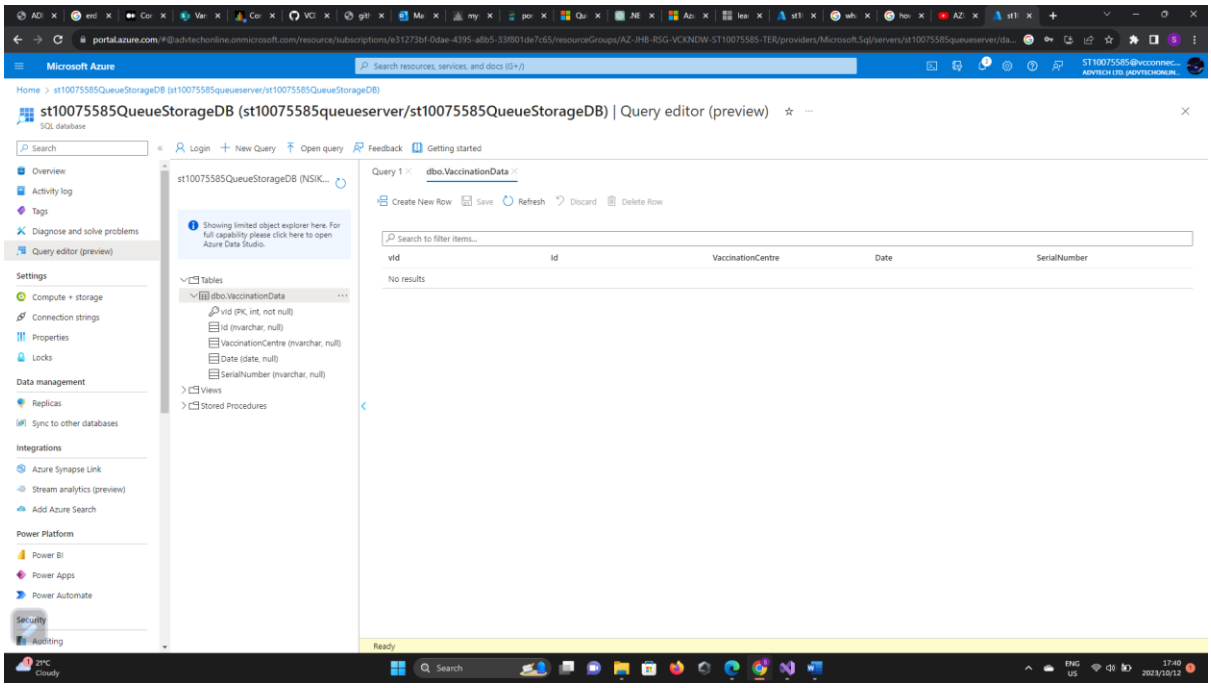


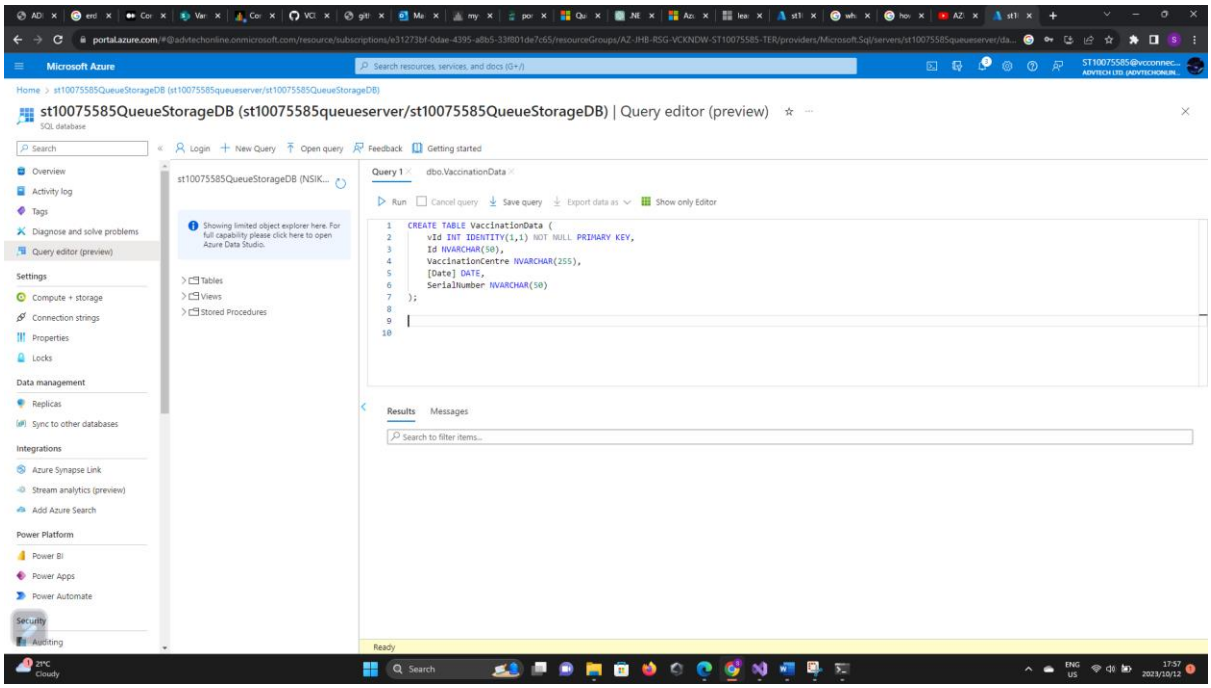
```
using Azure.Storage.Queues;
using Azure.Storage.Queues.Models;
using System.Net.Mail;

//Get connection string
string connectionString =
"DefaultEndpointsProtocol=https;AccountName=st10075585poestorage;AccountKey=X7Llz
4Jis2jUsvHwd4RyrymN/bkbXlq4olLRwxUPIyRqiC9+aQZeeUhxAyocALQXaIMB8xBdJCNk+AStDFCmpA
==;EndpointSuffix=core.windows.net";

//Create the queue
string queueName = "st10075585part2poe-queue";
QueueClient queueClient = new QueueClient(connectionString, queueName);

Console.WriteLine($"Creating queue {queueName}");
await queueClient.CreateIfNotExistsAsync();

//Insert messages into the queue
Console.WriteLine($"\nAdding messages to the {queueName}");
await queueClient.SendMessageAsync("1234567890123:VaccinationCenter1:2023-10-
15:Vaccine12345");
await queueClient.SendMessageAsync("VaccineBarcode1:2023-10-
20:VaccinationCenter2:98765432101123");
await queueClient.SendMessageAsync("98765432101123:VaccinationCenter2:2023-10-
20:Vaccine67890");
await queueClient.SendMessageAsync("VaccineBarcode2:2023-10-
15:VaccinationCenter1:1234567890123");
await queueClient.SendMessageAsync("2345678901234:VaccinationCenter3:2023-11-
01:Vaccine54321");

//PEEK MESSAGES INSIDE QUEUE
Console.WriteLine($"\nPeeking at messages in {queueName}");
PeekedMessage[] peekedMessages = await queueClient.PeekMessagesAsync(maxMessages:
10);

foreach (var peekedMessage in peekedMessages)
{
```

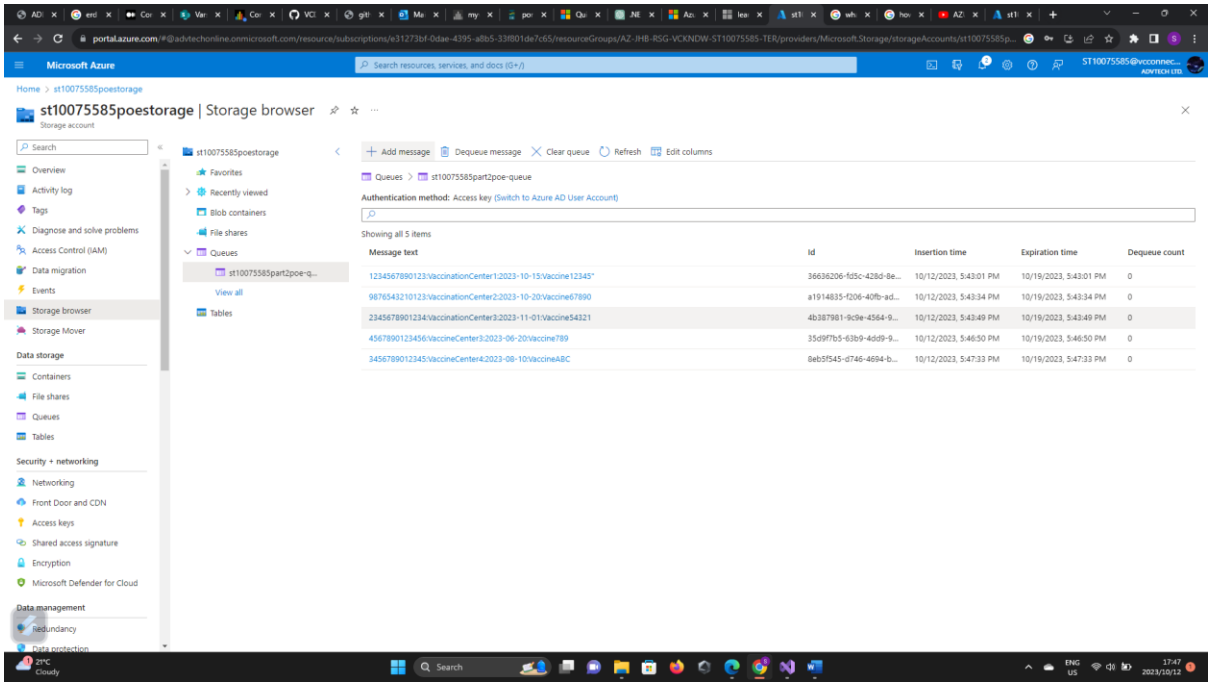
```
    Console.WriteLine($"\nMessage: {peekedMessage.MessageId} :
{peekedMessage.Body} : {peekedMessage.DequeueCount} :
{peekedMessage.ExpiresOn}");
}

//Use senderreceipt to view information
string messageContent = "You have been vaccinated in our system";

Console.WriteLine("\nInserting new message with Send Receipt");
SendReceipt receipt = await queueClient.SendMessageAsync(messageContent);

Console.WriteLine($"Message Sent: {messageContent}");
Console.WriteLine($"MessageId: {receipt.MessageId}");
Console.WriteLine($"InsertionTime: {receipt.InsertionTime}");
Console.WriteLine($"ExpirationTime: {receipt.ExpirationTime}");
Console.WriteLine($"TimeNextVisible: {receipt.TimeNextVisible}");
Console.WriteLine($"PopReciept: {receipt.PopReceipt}");

//updating a message inside the queue using the sender recipt
Console.WriteLine("\nUpdating message with Send Receipt");
await queueClient.UpdateMessageAsync(receipt.MessageId, receipt.PopReceipt, "You
are part of the vaccinated group");

//processing and deleting messages in the queue

//recieve messages
Console.WriteLine("\nReceiving messages from the queue");
QueueMessage[] messages = await queueClient.ReceiveMessagesAsync(maxMessages:
10);

//process the message
Console.WriteLine("\nPress Enter key to 'process' messages and delete them from
the queue");
Console.ReadLine();

//Finish the process and delete the messgaes
foreach (QueueMessage message in messages)
{
    Console.WriteLine($"Message: {message.Body}");

    //let service know we are done processing messages and dequeue
    await queueClient.DeleteMessageAsync(message.MessageId, message.PopReceipt);
}

Console.WriteLine("Execution Completed");
```

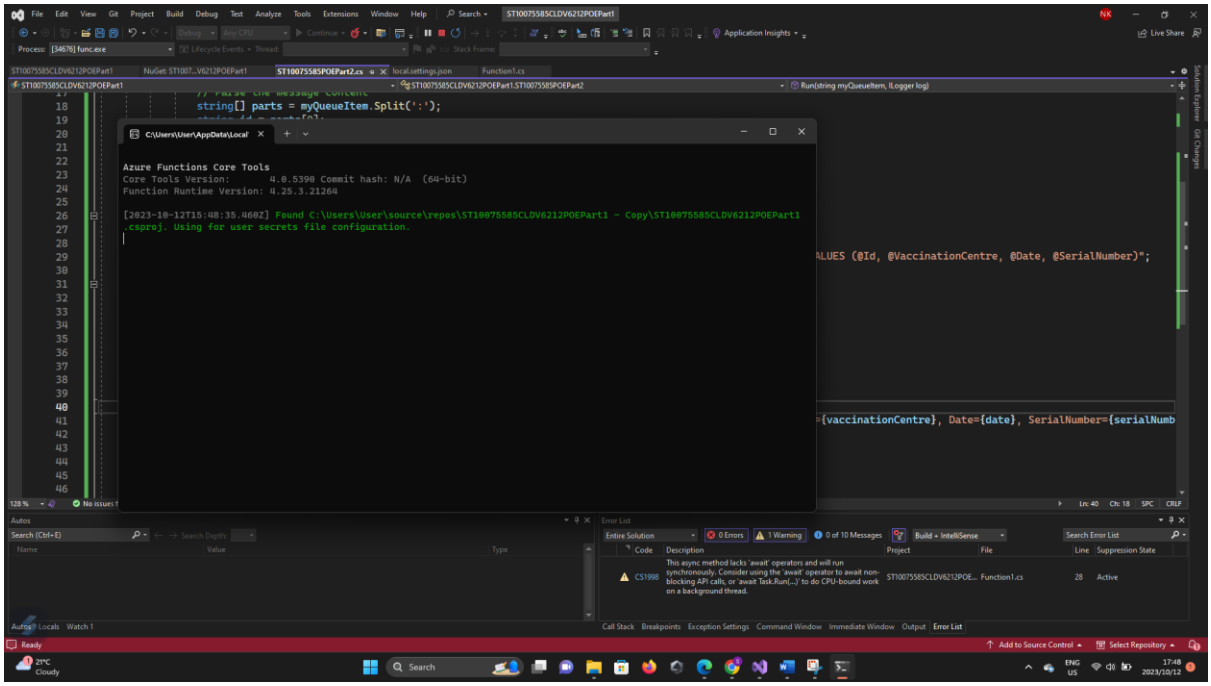
(Leshaba, 2023)

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

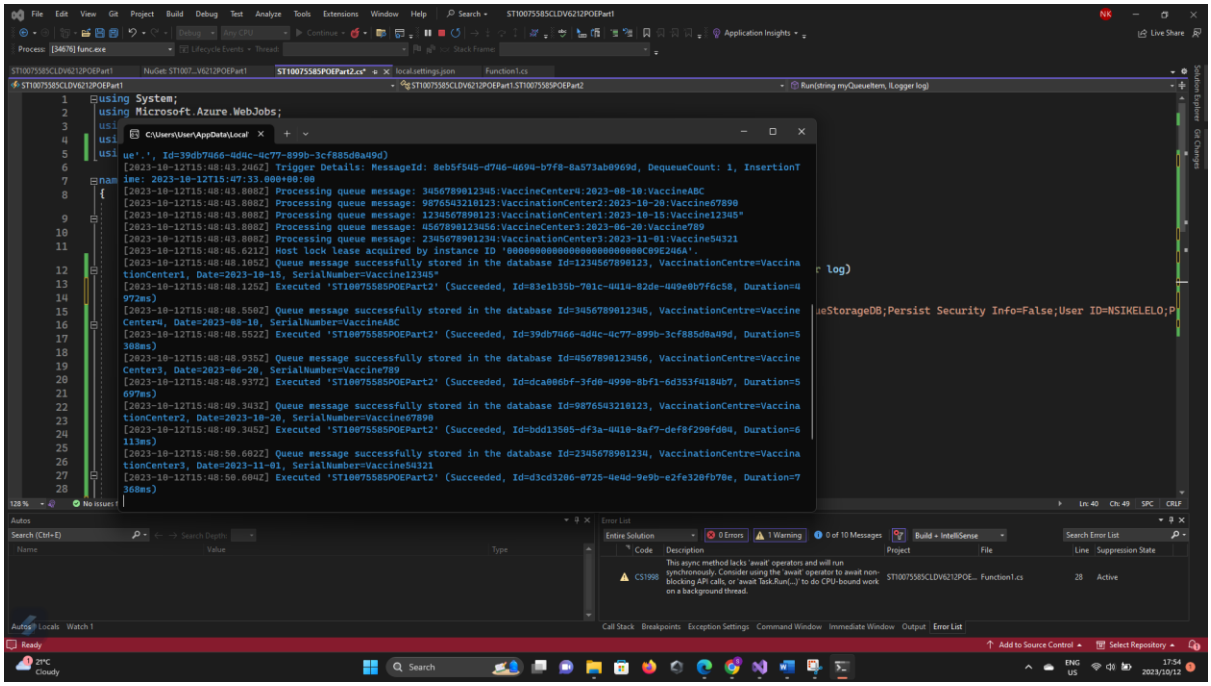


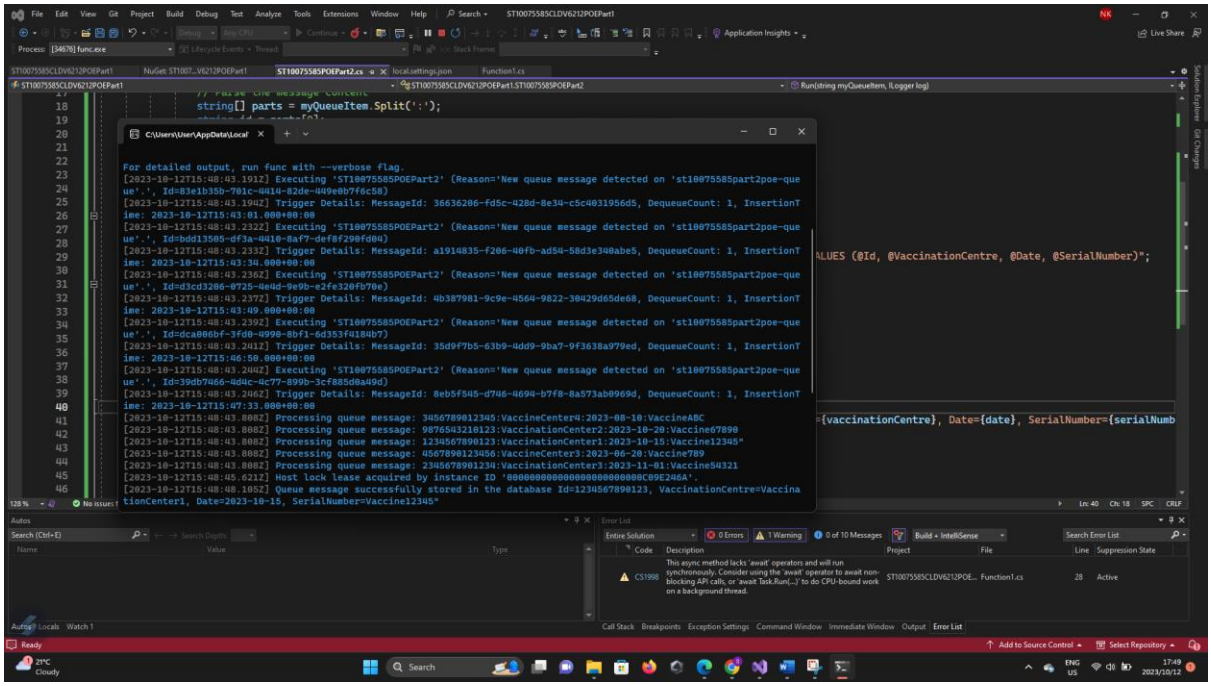


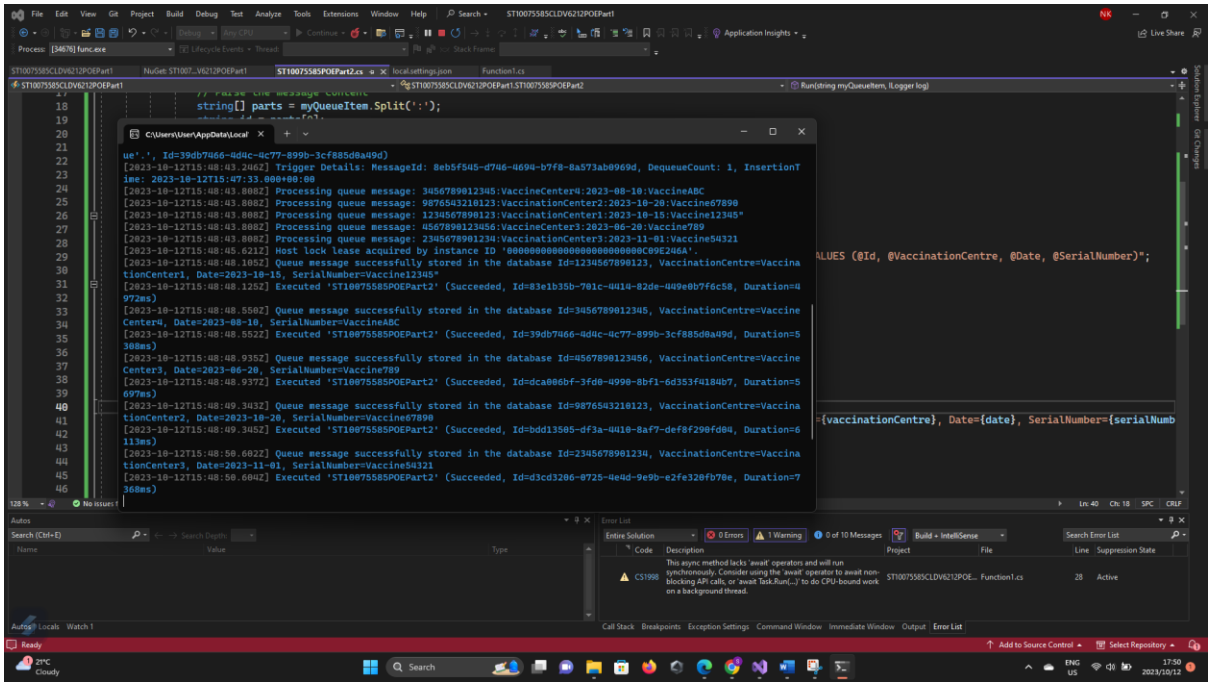

Microsoft Azure
st10075585poestorage | Storage browser
Storage account
Add message
Dequeue message
Clear queue
Refresh
Edit columns
Queues
st10075585part2poe-queue
Authentication method: Access key (Switch to Azure AD User Account)
Showing all 5 items
Message text
Id
Insertion time
Expiration time
Dequeue count
1234567890123:VaccinationCenter1:2023-10-15:Vaccine12345"
36636206-fd5c-428d-8e...
10/12/2023, 5:43:01 PM
10/19/2023, 5:43:01 PM
0
9876543210123:VaccinationCenter2:2023-10-20:Vaccine67890
a1914835-f206-40fb-ad...
10/12/2023, 5:43:34 PM
10/19/2023, 5:43:34 PM
0
2345678901234:VaccinationCenter3:2023-11-01:Vaccine54321
4b387981-9c9e-4564-9...
10/12/2023, 5:43:49 PM
10/19/2023, 5:43:49 PM
0
4567890123456:VaccineCenter3:2023-06-20:Vaccine789
35d9f7b5-63b9-4dd9-9...
10/12/2023, 5:46:50 PM
10/19/2023, 5:46:50 PM
0
3456789012345:VaccineCenter4:2023-08-10:VaccineABC
9eb5f545-d746-4694-b...
10/12/2023, 5:47:33 PM
10/19/2023, 5:47:33 PM
0


Azure Functions Core Tools
Core Tools Version: 4.0.5390 Commit hash: N/A (64-bit)
Function Runtime Version: 4.25.3.21264
[2023-10-12T15:48:35.460Z] Found C:\Users\User\source\repos\ST10075585CLDV6212POEPart1 - Copy\ST10075585CLDV6212POEPart1.csproj. Using for user secrets file configuration.
string[] parts = myQueueItem.Split(':');
ALUES (@Id, @VaccinationCentre, @Date, @SerialNumber)";
{vaccinationCentre}, Date={date}, SerialNumber={serialNumb
CS1998
This async method lacks 'await' operators and will run synchronously. Consider using the 'await' operator to await non-blocking API calls, or 'await Task.Run(...)' to do CPU-bound work on a background thread.
Function1.cs
28
Active

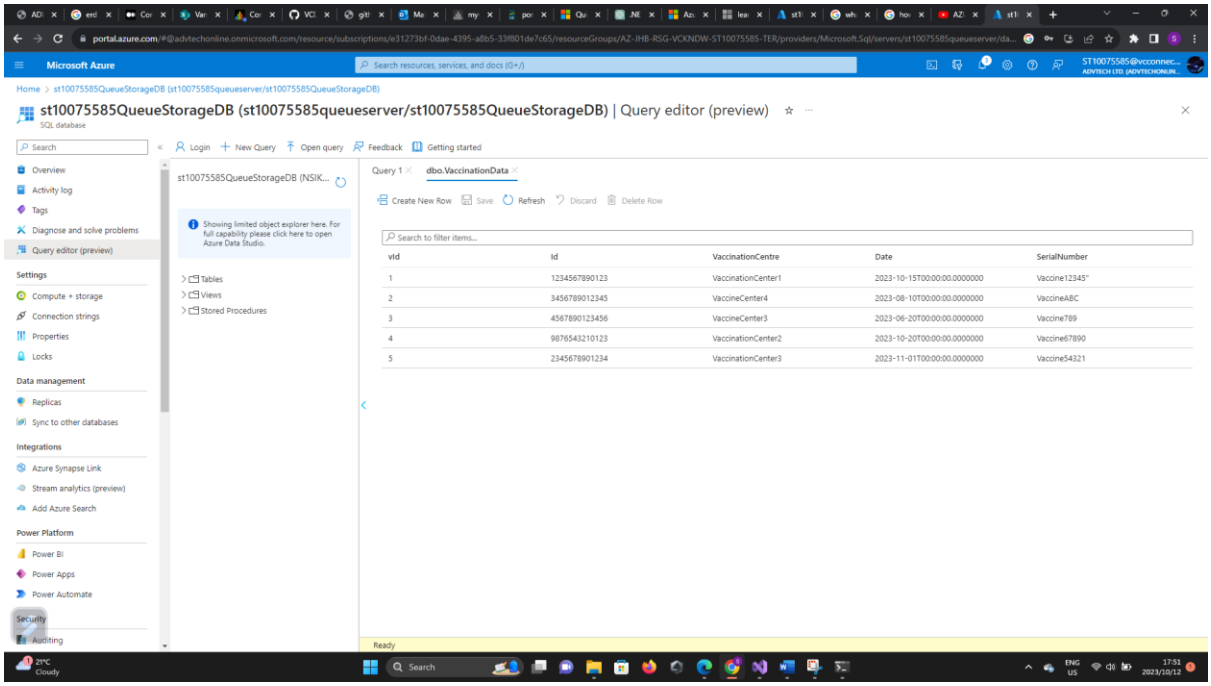
Azure function is triggered when message is placed in queue.



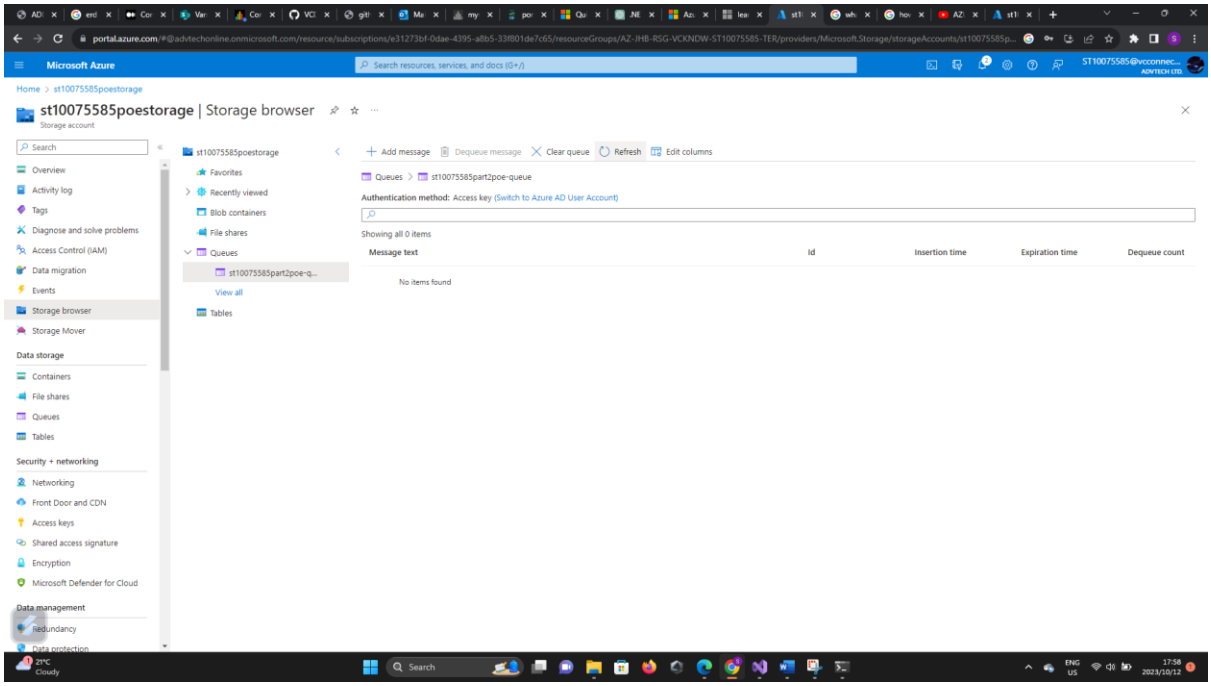




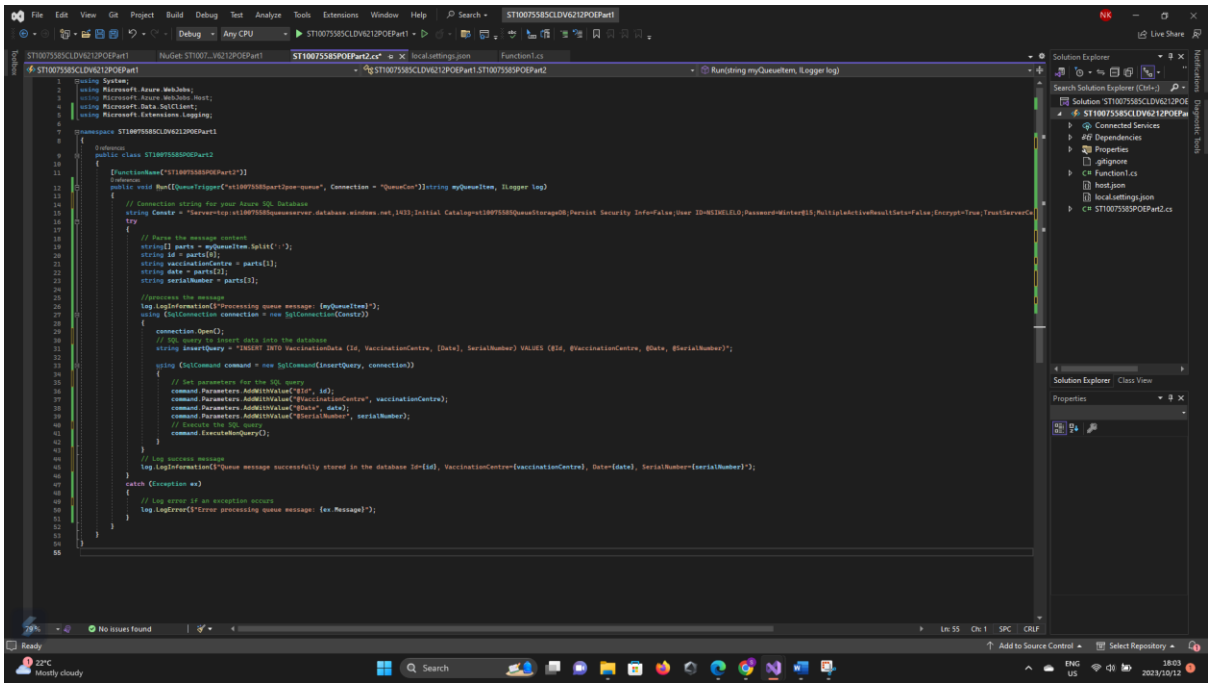
Database entries corresponding to storage queue:



Queue trigger after the code has called it and it has been entered in the database:



Queue trigger function code:



```
using System;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Host;
using Microsoft.Data.SqlClient;
using Microsoft.Extensions.Logging;

namespace ST10075585CLDV6212POEPart1
{
    public class ST10075585POEPart2
    {
        [FunctionName("ST10075585POEPart2")]
```

```
        public void Run([QueueTrigger("st10075585part2poe-queue", Connection =
"QueueCon")]string myQueueItem, ILogger log)
        {
            // Connection string for your Azure SQL Database
            string Constr =
"Server=tcp:st10075585queueserver.database.windows.net,1433;Initial
Catalog=st10075585QueueStorageDB;Persist Security Info=False;User
ID=NSIKELELO;Password=Winter@15;MultipleActiveResultSets=False;Encrypt=True;Trust
ServerCertificate=False;Connection Timeout=30;";
            try
            {
                // Parse the message content
                string[] parts = myQueueItem.Split(':');
                string id = parts[0];
                string vaccinationCentre = parts[1];
                string date = parts[2];
                string serialNumber = parts[3];

                //proccess the message
                log.LogInformation($"Processing queue message: {myQueueItem}");
                using (SqlConnection connection = new SqlConnection(Constr))
                {
                    connection.Open();
                    // SQL query to insert data into the database
                    string insertQuery = "INSERT INTO VaccinationData (Id,
VaccinationCentre, [Date], SerialNumber) VALUES (@Id, @VaccinationCentre, @Date,
@SerialNumber)";

                    using (SqlCommand command = new SqlCommand(insertQuery,
connection))
                    {
                        // Set parameters for the SQL query
                        command.Parameters.AddWithValue("@Id", id);
                        command.Parameters.AddWithValue("@VaccinationCentre",
vaccinationCentre);
                        command.Parameters.AddWithValue("@Date", date);
                        command.Parameters.AddWithValue("@SerialNumber",
serialNumber);
                        // Execute the SQL query
                        command.ExecuteNonQuery();
                    }
                }
                // Log success message
                log.LogInformation($"Queue message successfully stored in the
database Id={id}, VaccinationCentre={vaccinationCentre}, Date={date},
SerialNumber={serialNumber}");
            }
            catch (Exception ex)
            {
                // Log error if an exception occurs
                log.LogError($"Error processing queue message: {ex.Message}");
            }
        }
    }
}
```

(Leshaba, 2023)

Bibliography

ATLASSIAN, 2023. *MTBF, MTTR, MTTA, and MTTF.* [Online]
Available at: https://www.atlassian.com/incident-management/kpis/common-metrics#:~:text=MTBF%20(mean%20time%20between%20failures,the%20more%20reliable%20the%20system.
[Accessed 20 August 2023].

Brunskill, V.-L., 2023. *polyglot persistence.* [Online]
Available at: https://www.techtarget.com/searchapparchitecture/definition/polyglot-persistence
[Accessed 20 August 2023].

Cloud Native Glossary Authors, 2022. *Immutable Infrastructure.* [Online]
Available at: https://glossary.cncf.io/immutable-infrastructure/#:~:text=Immutable%20Infrastructure%20refers%20to%20computer,changes%20in%20the%20first%20place.
[Accessed 20 August 2023].

Eckert, M., 2012. *Which approach is better: Frequent small updates, or occasional large ones?.* [Online]
Available at: https://gamedev.stackexchange.com/questions/25167/which-approach-is-better-frequent-small-updates-or-occasional-large-ones
[Accessed 28 August 2023].

GeeksforGeeks, 2023. *Introduction of Process Synchronization.* [Online]
Available at: https://www.geeksforgeeks.org/introduction-of-process-synchronization/
[Accessed 28 August 2023].

Glossary, G., n.d. *Scalability.* [Online]
Available at: https://www.gartner.com/en/information-technology/glossary/scalability
[Accessed 20 August 2023].

Harris, C., 2023 . *Microservices vs. monolithic architecture.* [Online]
Available at: https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith
[Accessed 3 August 2023].

IBM, 2023. *Asynchronous processing.* [Online]
Available at: https://www.ibm.com/docs/en/cics-ts/5.4?topic=intercommunication-asynchronous-processing
[Accessed 28 August 2023].

Kemp, P., 2023. *Snowflake Servers.* [Online]
Available at: https://kemptechnologies.com/resources/glossary/snowflake-servers#:~:text=Snowflake%20servers%20run%20a%20mission,operating%20system%20and%20application%20server.
[Accessed 20 August 2023].

Leshaba, I., 2023. *Function App set up.mp4.* [Online]
Available at: https://drive.google.com/file/d/1nDWcNudJoI2aoD7PRC71ZAk__Jh_oW3Y/view
[Accessed 20 August 2023].

Leshaba, I., 2023. *HTTP Trigger Tutorial.mp4.* [Online]
Available at: https://drive.google.com/file/d/1Djnw-1KBmRToPnJG_JvyXpUlyqJoEEL7/view
[Accessed 20 August 2023].

Leshaba, I., 2023. *Queue Storage Console App Basics.mp4,* s.l.: IIE Varsity College.

Leshaba, I., 2023. *Queue Storage With Azure SQL Database'.mp4,* s.l.: IIE varsity College.

LinkedIn, 2023. *What are some common challenges or pitfalls of using a paper-based or manual maintenance work order system?.* [Online]
Available at: https://www.linkedin.com/advice/0/what-some-common-challenges-pitfalls-using#:~:text=A%20manual%20system%20is%20an,paper%20waste%20and%20storage%20space.
[Accessed 20 August 2023].

Lutkevich, B., 2021. *relational database.* [Online]
Available at: https://www.techtarget.com/searchdatamanagement/definition/relational-database
[Accessed 20 August 2023].

Lynn, T. et al., 2016. *CLOUDLIGHTNING: A Framework for a Self-organising and Self-managing Heterogeneous Cloud.* [Online]
Available at: https://www.scitepress.org/Link.aspx?doi=10.5220/0005921503330338
[Accessed 20 August 2023].

madhav_mohan, 2023. *Scalability and Elasticity in Cloud Computing.* [Online]
Available at: https://www.geeksforgeeks.org/scalability-and-elasticity-in-cloud-computing/
[Accessed 20 August 2023].

Microsoft, 2023. *Getting started with Azure Functions.* [Online]
Available at: https://learn.microsoft.com/en-us/azure/azure-functions/functions-get-started?pivots=programming-language-csharp
[Accessed 1 October 2023].

Microsoft, 2023. *Quickstart: Azure Queue Storage client library for .NET.* [Online]
Available at: https://learn.microsoft.com/en-us/azure/storage/queues/storage-quickstart-queues-dotnet?tabs=connection-string%2Croles-azure-portal%2Cenvironment-variable-windows%2Csign-in-azure-cli
[Accessed 1 October 2023].

Puviani, M. & Frei, R., 2013. *Self-management for cloud computing.* [Online]
Available at: https://ieeexplore.ieee.org/document/6661855
[Accessed 20 August 2023].

reddit, 2011. *Which approach is better: Frequent small updates, or occasional large ones?.* [Online]
Available at:
https://www.reddit.com/r/gamedev/comments/qmew0/which_approach_is_better_frequent_small_updates/
[Accessed 29 August 2023].

ST10075585, N. K., 2023. *Nsikelelo Kumalo's Function.* [Online]
Available at:
https://st10075585.azurewebsites.net/api/id/{id}?code=G9TG2GqMqbdcmRAhrmYi6JGBACEoIK0TYZ9ypstTSsMIAzFu1BJBpA==
[Accessed 2 September 2023].