Microsoft Azure
Lab Book

Capgemini Internal

Document Revision History

Date	Revision No.	Author	Summary of Changes
30-Jun-2018	1.0	Karthik Muthukrishnan	Initial Version

Table of Contents

Document Revision History 2
Table of Contents 3
Getting Started 4
Overview 4
Setup Checklist for Azure Services 4
Instructions 4
Lab 1. Creating a Linux virtual machine in the Azure portal 5
Lab 2. Creating Website using Azure portal 15
Lab 3. Creating a Function App in the Azure portal 19
Lab 4. Creating a Container and Upload a Blob 24
Lab 5. CRUD Operation in Azure Table Storage using C# 29
Lab 6. Creating a Logic App 35
Lab 7. Build a .NET web app with Azure Cosmos DB 44
Lab 8. Azure Management Client using C# Console App 50
Lab 9. Use RBAC to manage access with the REST API 59
Lab 10. Working with Key Vault 71

Getting Started

Overview

This lab book is a guided tour for learning Azure services and Developing Applications for Azure through .NET. It comprises scenario based applications and Stretched assignments. Flow diagrams and screen snap shots are provided where necessary.

Setup Checklist for Azure Services

Here is what is expected on your machine for the lab to work.

Minimum System Requirements

- Intel Core i3 and above
- Microsoft Windows 10 64 Bit OS
- Memory: 4GB or more recommended
- Google Chrome 65 or higher Internet Explorer 11.0 or higher
- High Speed Internet connection (Min 20 Mbps)

Please ensure that the following is done:

- A fast internet connection
- Visual Studio 2017, Visual Studio Code, Azure Storage Explorer, Azure Storage Emulator, Azure Cosmos DB Emulator, Puttygen.exe, putty.exe, postman chrome plugin or postman for windows should be installed in your System
- Create account in Azure, Github, and Docker

Instructions

- Before starting the lab exercises, Sign in to Azure Portal.

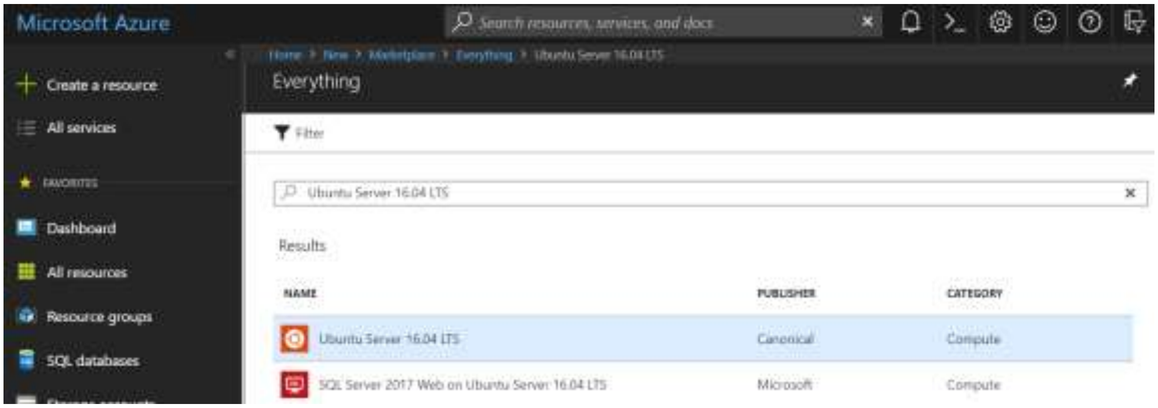
Lab 1. Creating a Linux virtual machine in the Azure portal

Goals	Deploy a Linux virtual machine (VM) in Azure that runs Ubuntu then SSH to the VM and install the NGINX web server.
Time	120 Mins

Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Choose Create a resource in the upper left-hand corner of the Azure portal.

Step 3: In the search box above the list of Azure Marketplace resources, search for and select Ubuntu Server 16.04 LTS by Canonical, then choose Create.



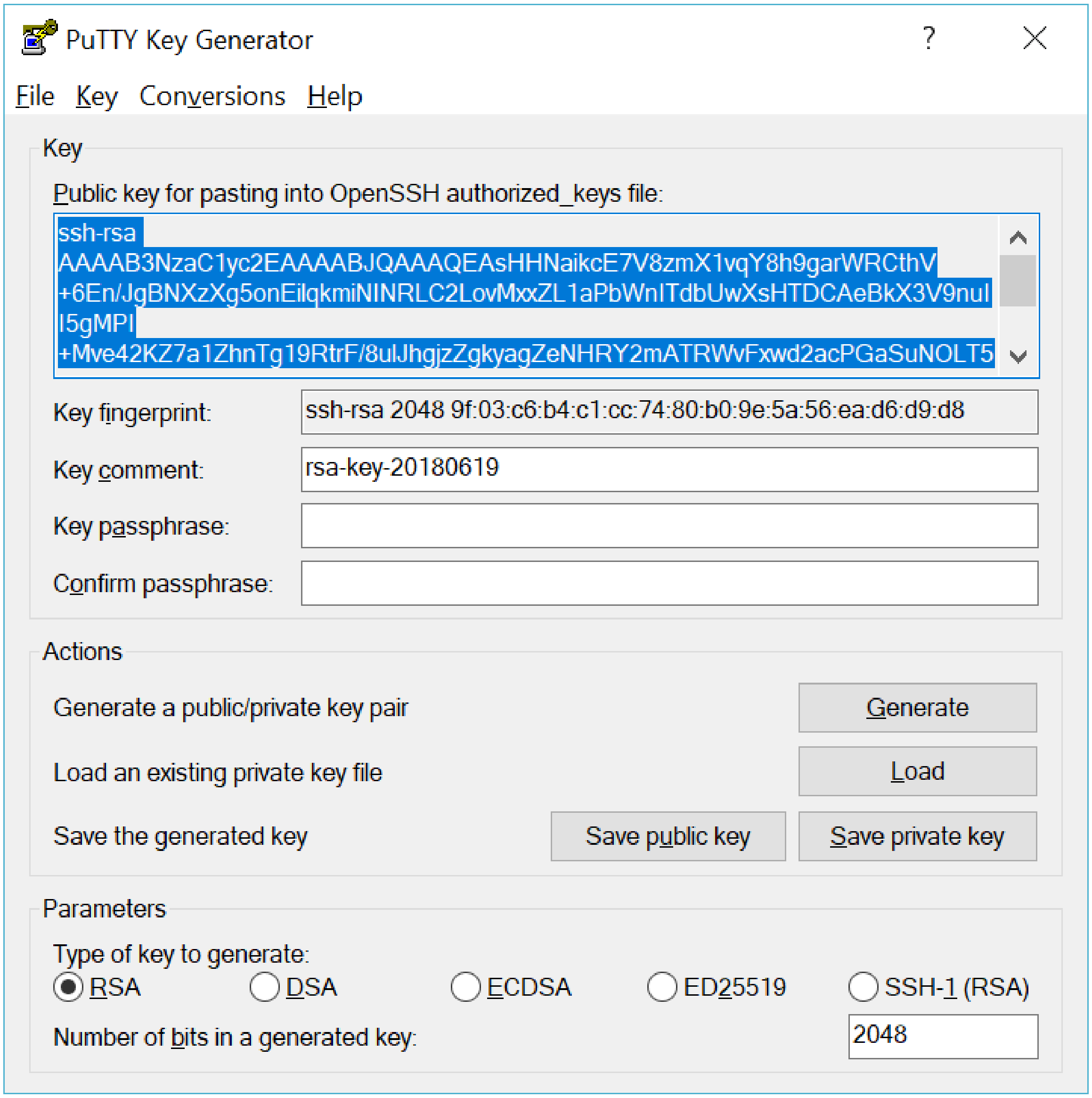
Ubuntu Server 16.04 LTS

Results

NAME	PUBLISHER	CATEGORY
Ubuntu Server 16.04 LTS	Canonical	Compute
SQL Server 2017 Web on Ubuntu Server 16.04 LTS	Microsoft	Compute

Step 4: Provide a VM name, such as karthikVM, leave the disk type as SSD, then provide a username, such as karthik.

Step 5: Create a SSH public key using Putty Key generator. Optionally provide Key passphrase and save the private key(linux-vm.ppk) in the system.



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAQEAsHHNaikcE7V8zmX1vqY8h9garWRCthV
+6En/JgBNXzXg5onEilqkmiNINRLC2LovMxxZL1aPbWnITdbUwXsHTDCAeBkX3V9nul
I5gMPI
+Mve42KZ7a1ZhnTg19RtrF/8uIJhgjzZgkyagZeNHRY2mATRWvFxwd2acPGaSuNOLT5
```

Key fingerprint: ssh-rsa 2048 9f:03:c6:b4:c1:cc:74:80:b0:9e:5a:56:ea:d6:d9:d8

Key comment: rsa-key-20180619

Key passphrase:

Confirm passphrase:

Generate a public/private key pair

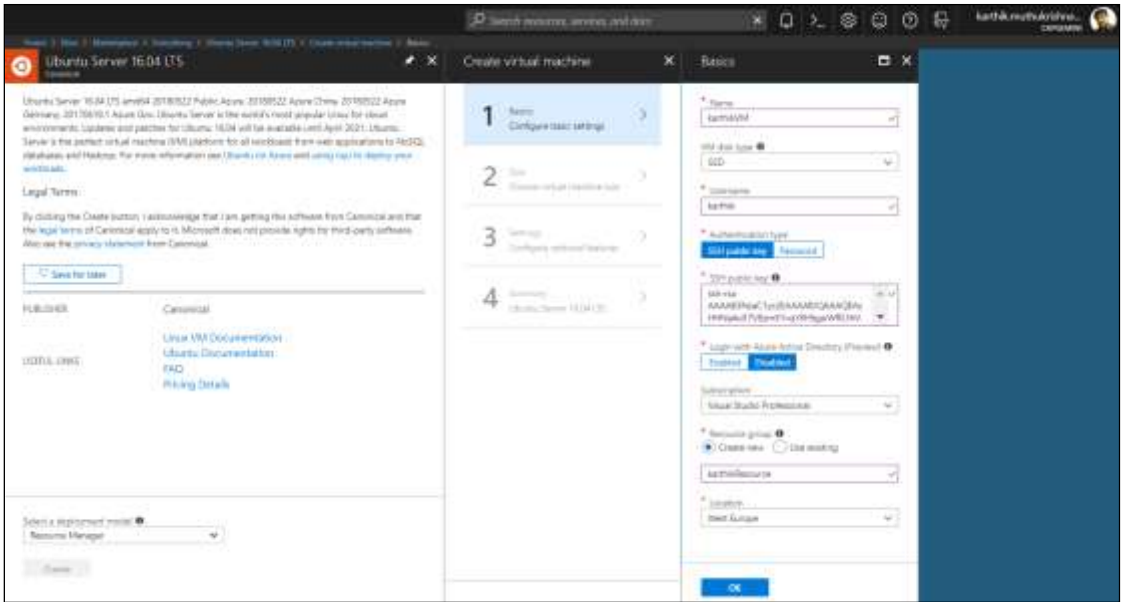
Load an existing private key file

Save the generated key

Type of key to generate:
● RSA ○ DSA ○ ECDSA ○ ED25519 ○ SSH-1 (RSA)

Number of bits in a generated key: 2048

Step 5: For Authentication type, select SSH public key, then paste your public key into the text box. Take care to remove any leading or trailing white space in your public key.



Ubuntu Server 16.04 LTS

Canonical

Ubuntu Server 16.04 LTS amd64 20180522 Public Azure, 20180522 Azure China, 20180522 Azure Germany, 20170619.1 Azure Gov. Ubuntu Server is the world's most popular Linux for cloud environments. Updates and patches for Ubuntu 16.04 will be available until April 2021. Ubuntu Server is the perfect virtual machine (VM) platform for all workloads from web applications to NoSQL databases and Hadoop. For more information see Ubuntu on Azure and using Juju to deploy your workloads.

Legal Terms

By clicking the Create button, I acknowledge that I am getting this software from Canonical and that the legal terms of Canonical apply to it. Microsoft does not provide rights for third-party software. Also see the privacy statement from Canonical.

PUBLISHER	Canonical
USEFUL LINKS	Linux VM Documentation Ubuntu Documentation FAQ Pricing Details

Select a deployment model: Resource Manager

Create virtual machine

1 Basics – Configure basic settings

2 Size – Choose virtual machine size

3 Settings – Configure optional features

4 Summary – Ubuntu Server 16.04 LTS

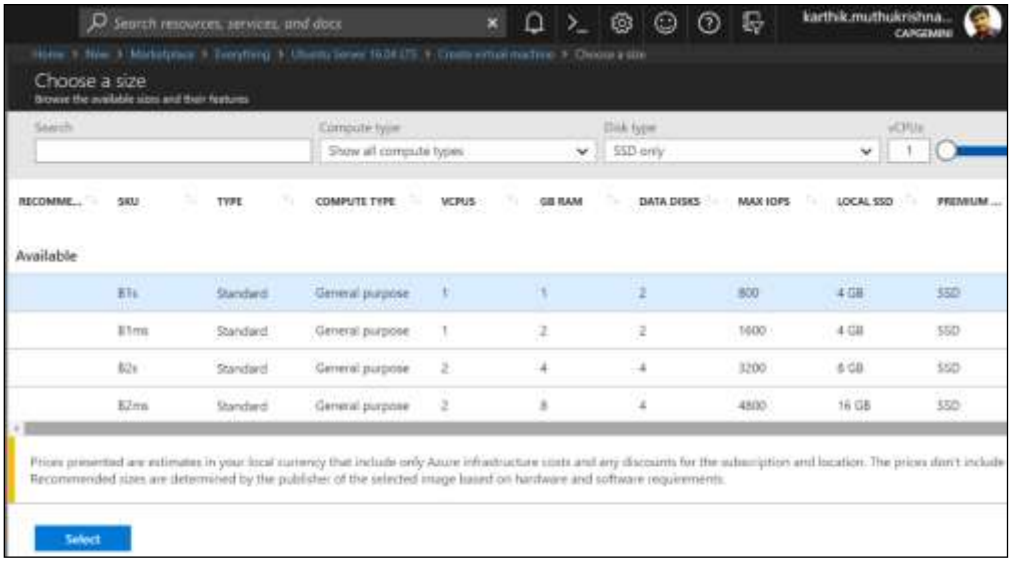
Basics

- * Name: karthikVM
- VM disk type: SSD
- * Username: karthik
- * Authentication type: [✓ SSH public key] Password
- * SSH public key: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAA...
- * Login with Azure Active Directory (Preview): Enabled [✓ Disabled]
- Subscription: Visual Studio Professional
- * Resource group: ● Create new ○ Use existing — karthikResource
- * Location: West Europe

Search resources, services, and docs
karthik.muthukrishna...
CAPGEMINI
Home > New > Marketplace > Everything > Ubuntu Server 16.04 LTS > Create virtual machine > Basics
Save for later
Create
OK

Step 6: Choose to Create new resource group, then provide a name, such as **karthikResource**. Choose the Location, then select OK.

Step 7: Select a size for the VM. You can filter by Compute type or Disk type, for example. A suggested VM size is B1s.



Choose a size

Browse the available sizes and their features

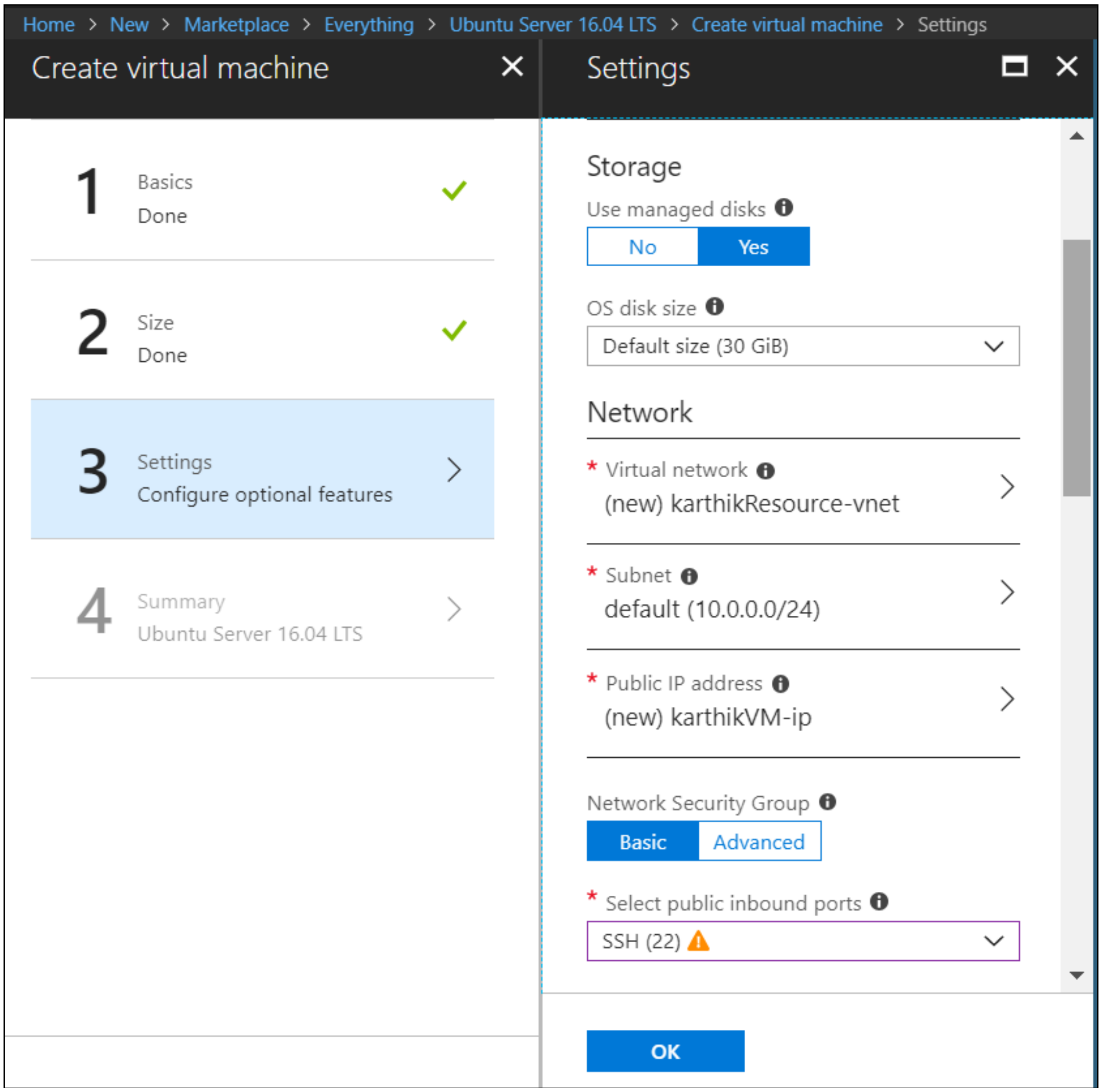
Search: ____ Compute type: Show all compute types Disk type: SSD only vCPUs: 1

RECOMME...	SKU	TYPE	COMPUTE TYPE	VCPUS	GB RAM	DATA DISKS	MAX IOPS	LOCAL SSD	PREMIUM ...
Available									
	B1s	Standard	General purpose	1	1	2	800	4 GB	SSD
	B1ms	Standard	General purpose	1	2	2	1600	4 GB	SSD
	B2s	Standard	General purpose	2	4	4	3200	8 GB	SSD
	B2ms	Standard	General purpose	2	8	4	4800	16 GB	SSD

Prices presented are estimates in your local currency that include only Azure infrastructure costs and any discounts for the subscription and location. The prices don't include
Recommended sizes are determined by the publisher of the selected image based on hardware and software requirements.

Search resources, services, and docs
karthik.muthukrishna...
CAPGEMINI
Home > New > Marketplace > Everything > Ubuntu Server 16.04 LTS > Create virtual machine > Choose a size
Select

Step 8: Under Settings, select public inbound ports SSH (22) and leave the others with defaults and select OK.



Home > New > Marketplace > Everything > Ubuntu Server 16.04 LTS > Create virtual machine > Settings
Create virtual machine
1 Basics Done
2 Size Done
3 Settings Configure optional features
4 Summary Ubuntu Server 16.04 LTS
Settings

Storage

Use managed disks: No [✓ Yes]

OS disk size: Default size (30 GiB)

Network

* Virtual network: (new) karthikResource-vnet

* Subnet: default (10.0.0.0/24)

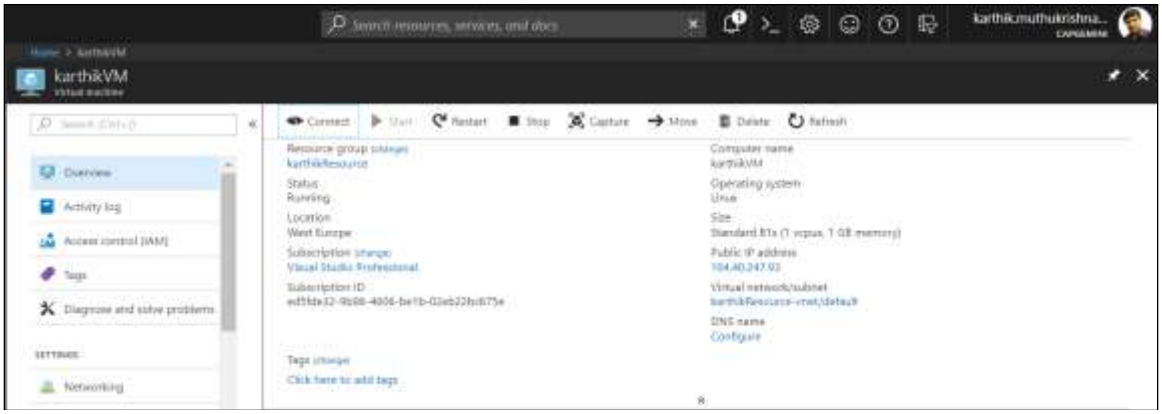
* Public IP address: (new) karthikVM-ip

Network Security Group: [✓ Basic] Advanced

* Select public inbound ports: SSH (22) ⚠

Step 9: On the summary page, select Create to start the VM deployment.

Step 10: The VM is pinned to the Azure portal dashboard. Once the deployment has completed, the VM summary automatically opens.



karthikVM
Virtual machine

Resource group (change)
karthikResource

Status
Running

Location
West Europe

Subscription (change)
Visual Studio Professional

Subscription ID
ed5fde32-9b98-4006-be1b-02eb22bc675e

Tags (change)
Click here to add tags

Computer name
karthikVM

Operating system
Linux

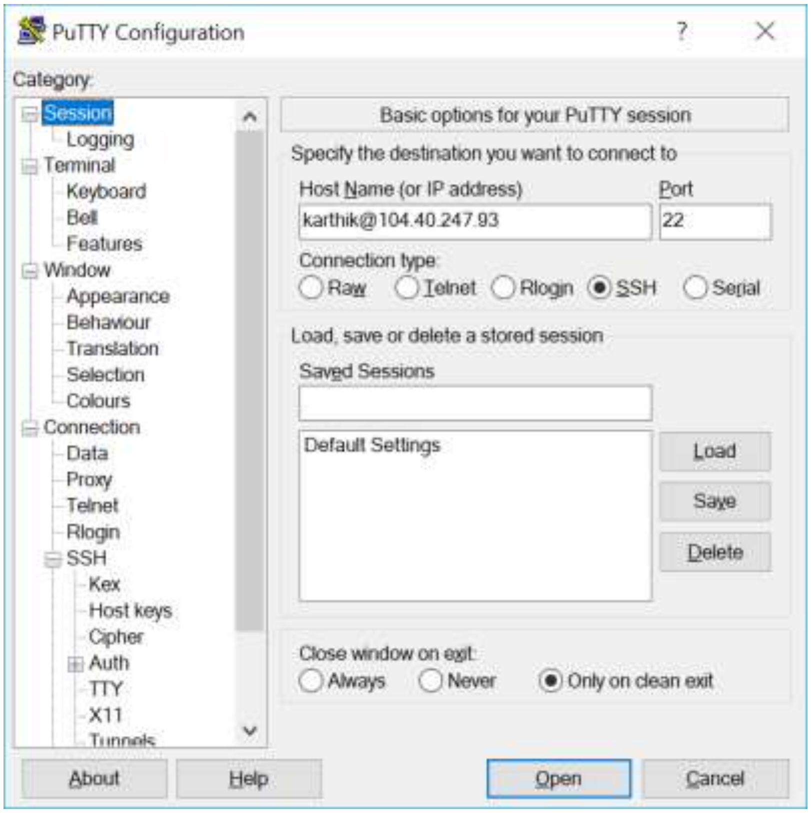
Size
Standard B1s (1 vcpus, 1 GB memory)

Public IP address
104.40.247.93

Virtual network/subnet
karthikResource-vnet/default

DNS name
Configure

Step 11: Connect the Linux VM using putty by fill in the host name or IP address of your VM from the Azure portal.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): karthik@104.40.247.93

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

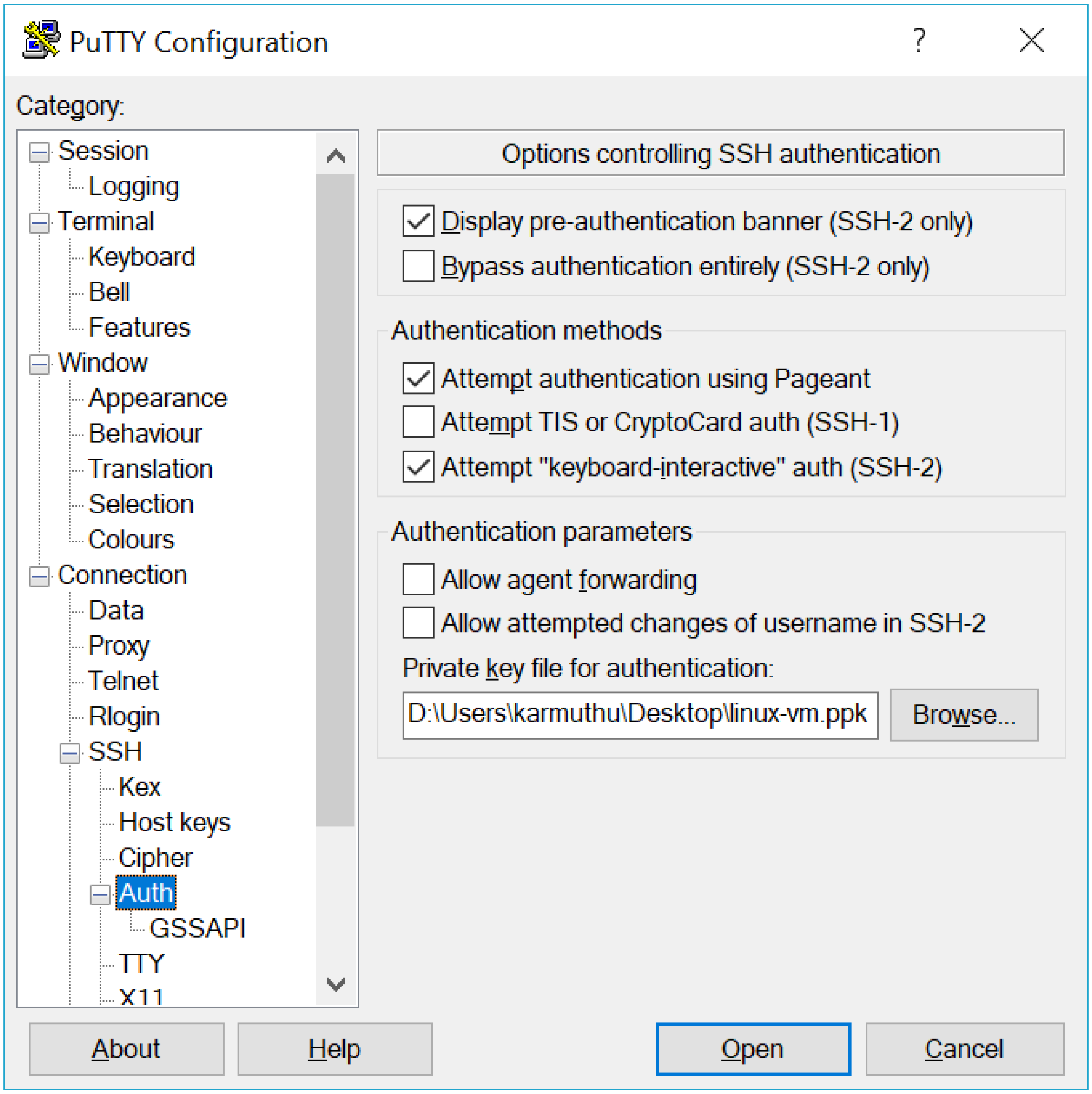
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Step 12: Before selecting Open, click Connection > SSH > Auth tab. Browse to and select your PuTTY private key (.ppk file) which we have save in the system.



Options controlling SSH authentication

☑ Display pre-authentication banner (SSH-2 only)

☐ Bypass authentication entirely (SSH-2 only)

Authentication methods

☑ Attempt authentication using Pageant

☐ Attempt TIS or CryptoCard auth (SSH-1)

☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

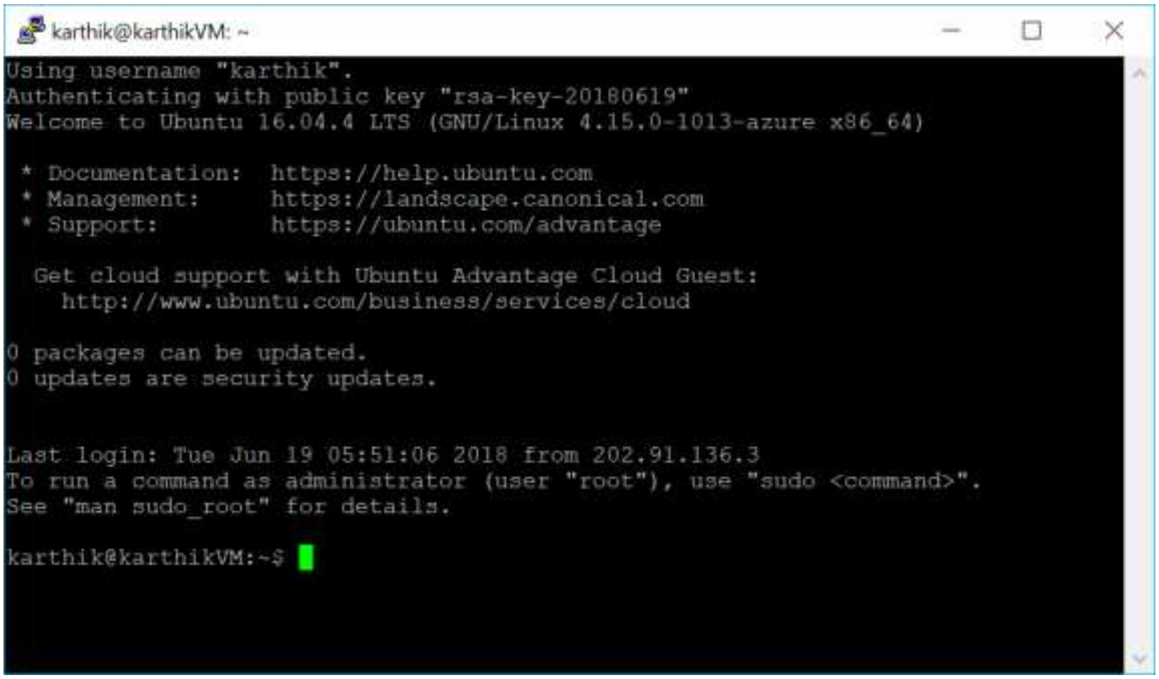
☐ Allow agent forwarding

☐ Allow attempted changes of username in SSH-2

Private key file for authentication: D:\Users\karmuthu\Desktop\linux-vm.ppk

Step 13: Before selecting Open, click Connection > SSH > Auth tab. Browse to and select your PuTTY private key (.ppk file) which we have save in the system.

Step 14: Click open button and accept the putty security alert to open the Linux VM.



```
Using username "karthik".
Authenticating with public key "rsa-key-20180619"
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-1013-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.


Last login: Tue Jun 19 05:51:06 2018 from 202.91.136.3
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

karthik@karthikVM:~$
```

Step 15: To see your VM in action, install the NGINX web server. To update package sources and install the latest NGINX package, run the following commands from your SSH session.

```
# update packages
sudo apt-get -y update

# install NGINX
sudo apt-get -y install nginx
```

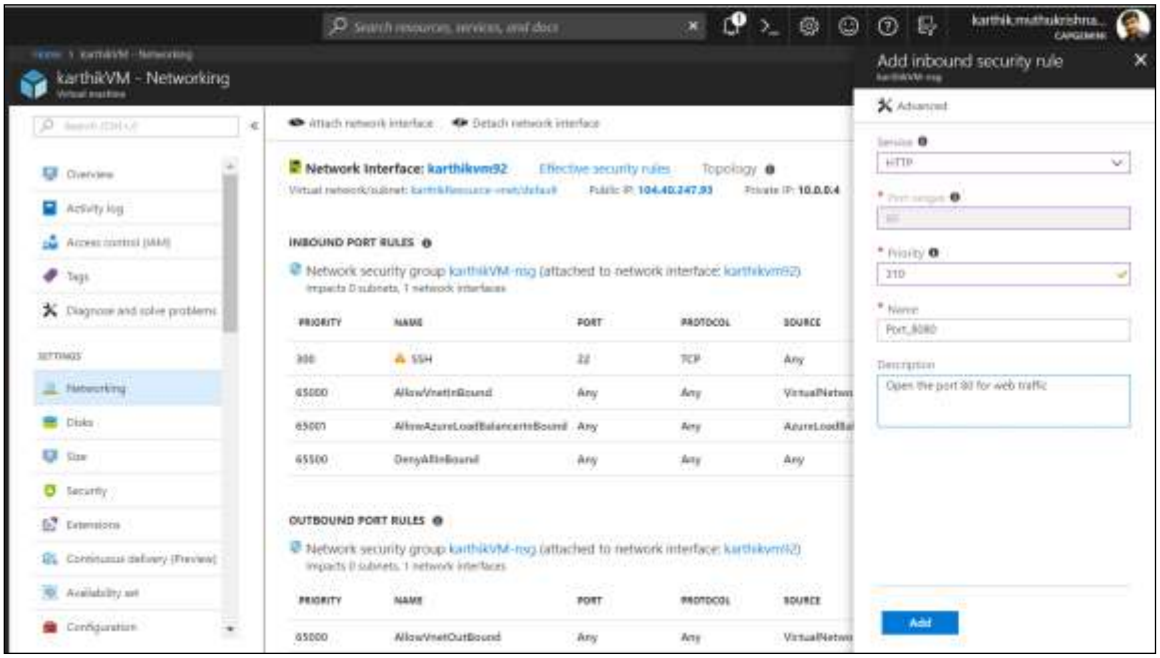
When done, exit the SSH session and return to the VM properties in the Azure portal.

Step 16: On the VM overview page, select Networking.

Step 17: The list of existing inbound and outbound rules are shown. Choose to Add inbound port rule.

Step 18: Select the Basic option across the top, then choose HTTP from the list of available services. Port 80, a priority, and name, are provided for you to Open the port 80 for web traffic.

Step 19: To create the rule, select Add.



Network Interface: karthikvm92

Virtual network/subnet: karthikResource-vnet/default Public IP: 104.40.247.93 Private IP: 10.0.0.4

INBOUND PORT RULES

Network security group karthikVM-nsg (attached to network interface: karthikvm92)
Impacts 0 subnets, 1 network interfaces

PRIORITY	NAME	PORT	PROTOCOL	SOURCE
300	⚠ SSH	22	TCP	Any
65000	AllowVnetInBound	Any	Any	VirtualNetw
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBal
65500	DenyAllInBound	Any	Any	Any

OUTBOUND PORT RULES

Network security group karthikVM-nsg (attached to network interface: karthikvm92)
Impacts 0 subnets, 1 network interfaces

PRIORITY	NAME	PORT	PROTOCOL	SOURCE
65000	AllowVnetOutBound	Any	Any	VirtualNetw

Add inbound security rule
karthikVM-nsg

Service: HTTP

* Port ranges: 80

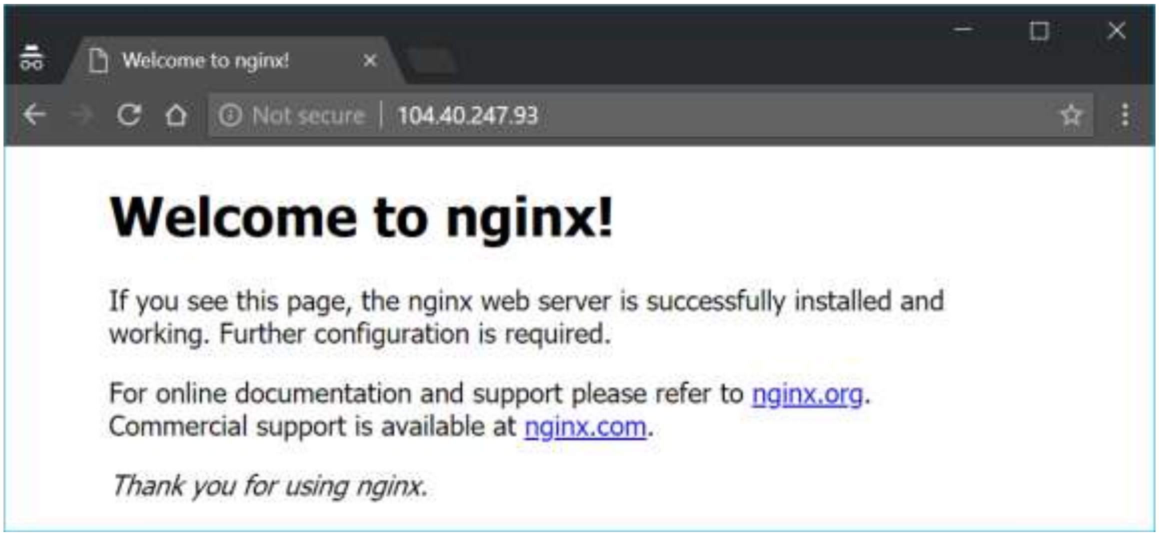
* Priority: 310

* Name: Port_8080

Description: Open the port 80 for web traffic

Step 20: With NGINX installed and port 80 open to your VM, the web server can now be accessed from the internet. Open a web browser, and enter the public IP address of the VM.

Note: public IP address can be found on the VM overview page, or at the top of the Networking page where you add the inbound port rule.



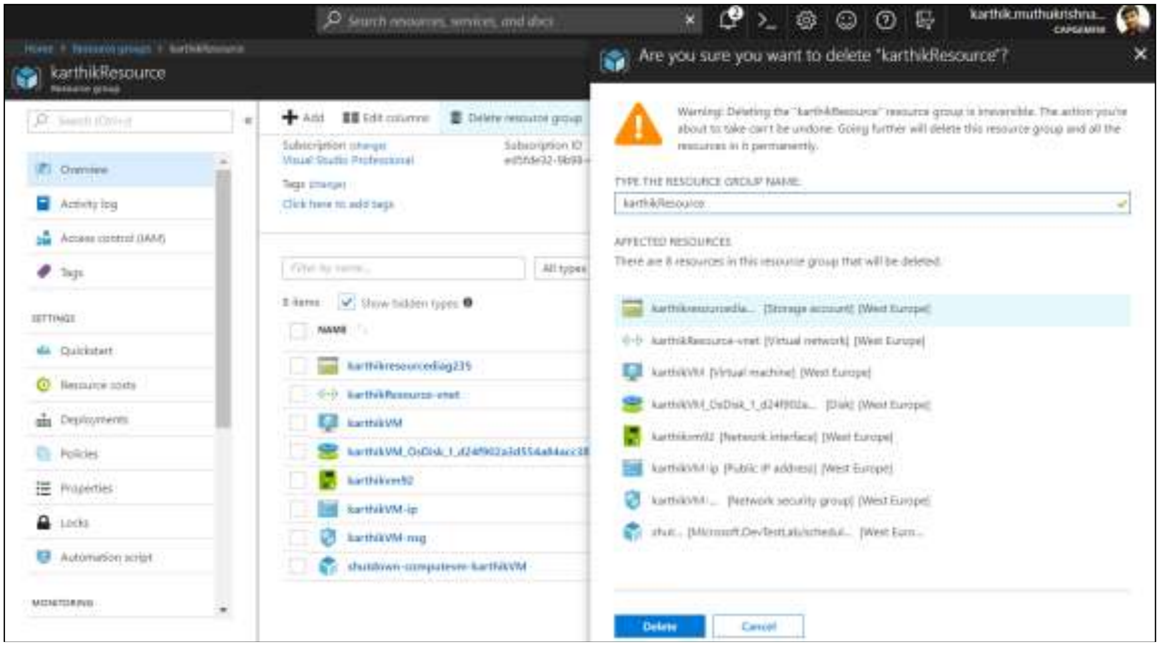
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Step 21: When no longer needed, you can delete the resource group, virtual machine, and all related resources. To do so, select the resource group for the virtual machine, select Delete, then confirm the name of the resource group to delete so that we can save money.



Are you sure you want to delete "karthikResource"?

Warning: Deleting the "karthikResource" resource group is irreversible. The action you're about to take can't be undone. Going further will delete this resource group and all the resources in it permanently.

TYPE THE RESOURCE GROUP NAME:

karthikResource ✓

AFFECTED RESOURCES

There are 8 resources in this resource group that will be deleted.

- karthikresourcediа... [Storage account] [West Europe]
- karthikResource-vnet [Virtual network] [West Europe]
- karthikVM [Virtual machine] [West Europe]
- karthikVM_OsDisk_1_d24f902a... [Disk] [West Europe]
- karthikvm92 [Network interface] [West Europe]
- karthikVM-ip [Public IP address] [West Europe]
- karthikVM-... [Network security group] [West Europe]
- shut... [Microsoft.DevTestLab/schedul... [West Euro...

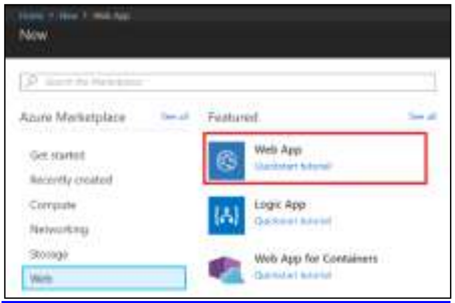
Lab 2. Creating Website using Azure portal

Goals	Create and deploy a basic HTML Site using Azure Portal
Time	60 Mins

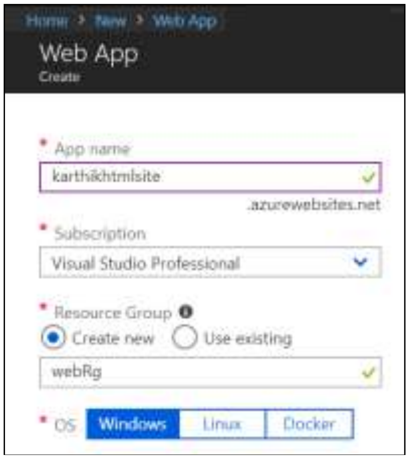
Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Choose Create a resource in the upper left-hand corner of the Azure portal.

Step 3: Select Web from Azure Marketplace then select Web App



Step 4: Create a Web App with a unique App name and create a new resource group



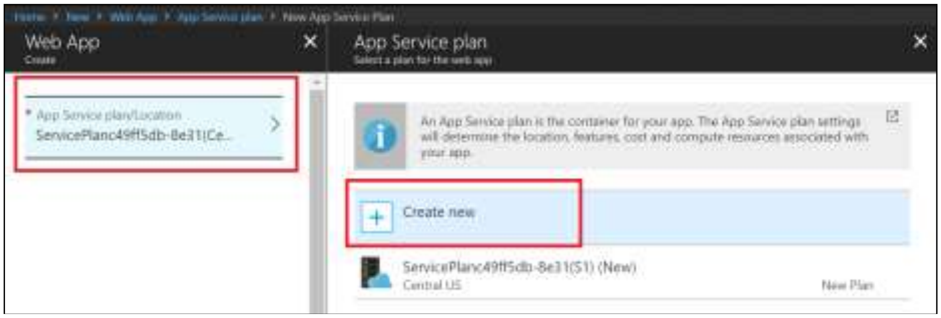
* App name
karthikhtmlsite
.azurewebsites.net

* Subscription
Visual Studio Professional

* Resource Group
● Create new ○ Use existing
webRg

* OS
Windows | Linux | Docker

Step 5: Select App Service plan/Location and create a new App Service Plan



Home > New > Web App > App Service plan > New App Service Plan

Web App
Create

* App Service plan/Location
ServicePlanc49ff5db-8e31(Ce...

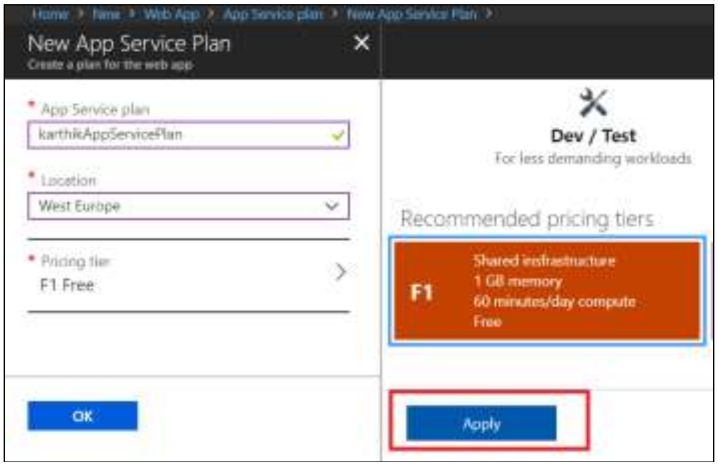
App Service plan
Select a plan for the web app

An App Service plan is the container for your app. The App Service plan settings will determine the location, features, cost and compute resources associated with your app.

Create new

ServicePlanc49ff5db-8e31(S1) (New)
Central US
New Plan

Step 6: Select App Service plan/Location and create a new App Service Plan by applying F1 Free Pricing tier from Dev / Test section and click create the Web App.



Home > New > Web App > App Service plan > New App Service Plan >

New App Service Plan
Create a plan for the web app

* App Service plan
karthikAppServicePlan

* Location
West Europe

* Pricing tier
F1 Free

OK

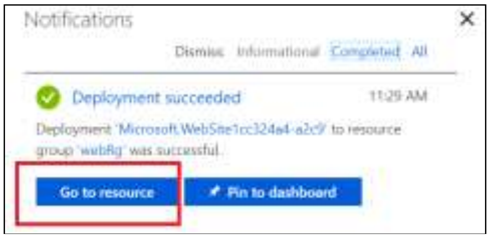
Dev / Test
For less demanding workloads

Recommended pricing tiers

F1
Shared infrastructure
1 GB memory
60 minutes/day compute
Free

Apply

Step 7: Click Goto Resourse once the deployment succeded from the notification message



Notifications

Dismiss Informational Completed All

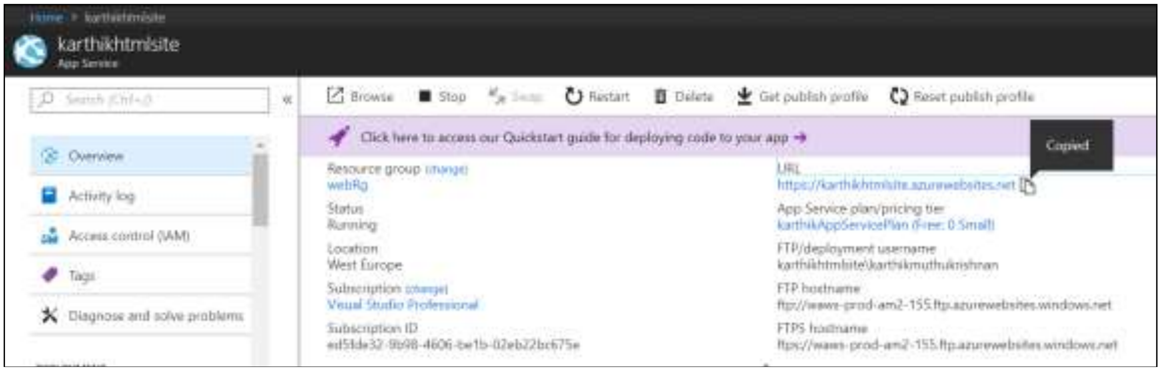
Deployment succeeded 11:29 AM

Deployment 'Microsoft.WebSite1cc324a4-a2c9' to resource group 'webRg' was successful.

Go to resource

Pin to dashboard

Step 8: Copy the URL of newly created Web app from the Web app overview page



Resource group (change)
webRg

Status
Running

Location
West Europe

Subscription (change)
Visual Studio Professional

Subscription ID
ed5fde32-9b98-4606-be1b-02eb22bc675e

URL
https://karthikhtmlsite.azurewebsites.net

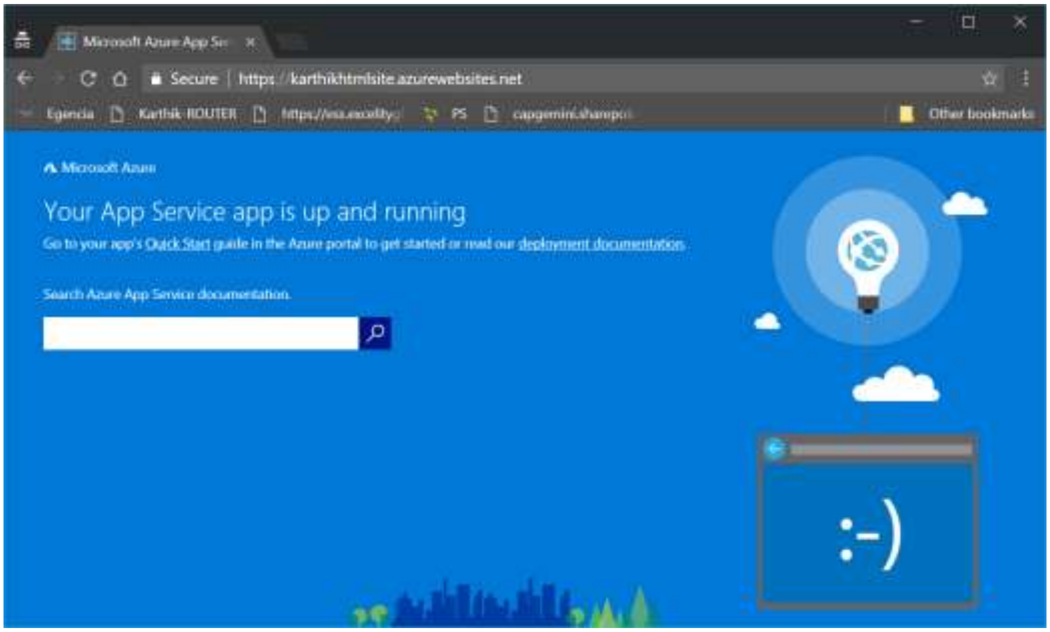
App Service plan/pricing tier
karthikAppServicePlan (Free: 0 Small)

FTP/deployment username
karthikhtmlsite\karthikmuthukrishnan

FTP hostname
ftp://waws-prod-am2-155.ftp.azurewebsites.windows.net

FTPS hostname
ftps://waws-prod-am2-155.ftp.azurewebsites.windows.net

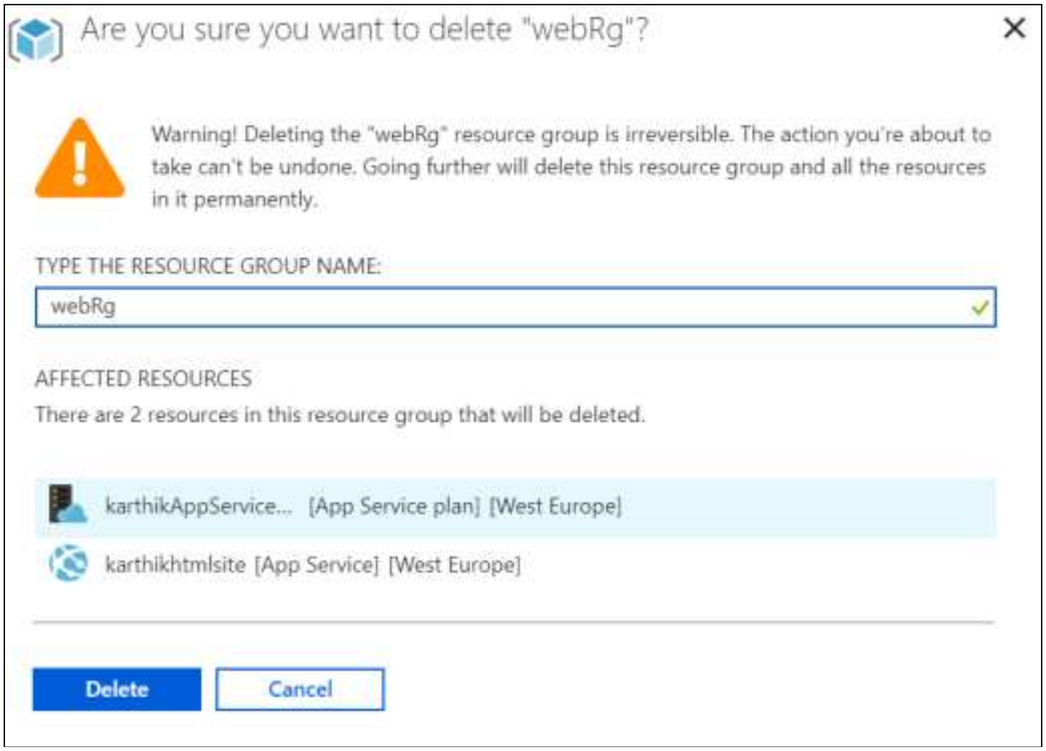
Step 9: Open the browser and paste the URL to access the web app we have created



Your App Service app is up and running

Go to your app's Quick Start guide in the Azure portal to get started or read our deployment documentation.

Step 10: Delete the Resource group WebRg to delete the App Service Plan and App Service



Are you sure you want to delete "webRg"?

Warning! Deleting the "webRg" resource group is irreversible. The action you're about to take can't be undone. Going further will delete this resource group and all the resources in it permanently.

TYPE THE RESOURCE GROUP NAME:

webRg ✓

AFFECTED RESOURCES

There are 2 resources in this resource group that will be deleted.

karthikAppService... [App Service plan] [West Europe]

karthikhtmlsite [App Service] [West Europe]

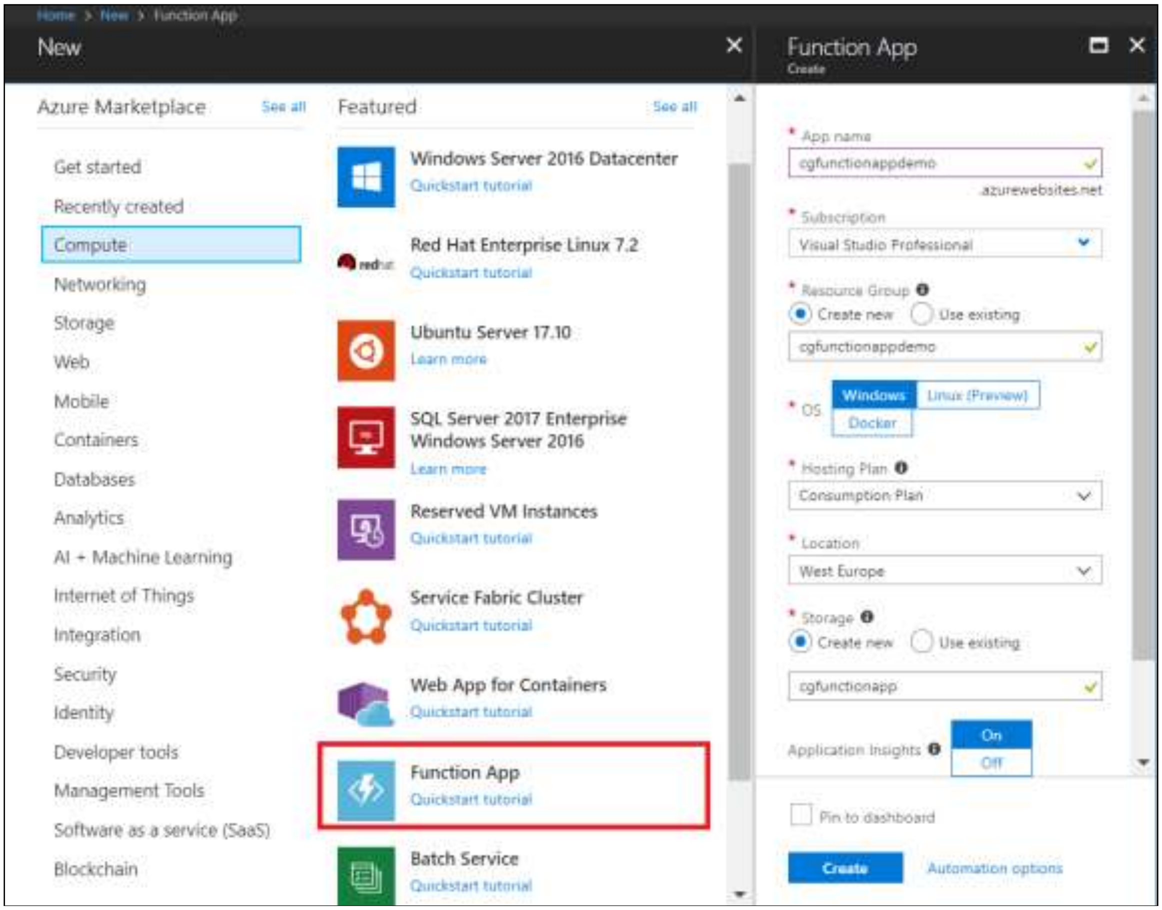
Lab 3. Creating a Function App in the Azure portal

Goals	Use Functions to create a "hello world" function in the Azure portal.
Time	40 Mins

Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Choose Create a resource in the upper left-hand corner of the Azure portal.

Step 3: Select Compute > Function App and use the function app settings as specified in the below image



* App name: cgfunctionappdemo .azurewebsites.net

* Subscription: Visual Studio Professional

* Resource Group: ● Create new ○ Use existing

cgfunctionappdemo

* OS: Windows | Linux (Preview) | Docker

* Hosting Plan: Consumption Plan

* Location: West Europe

* Storage: ● Create new ○ Use existing

cgfunctionapp

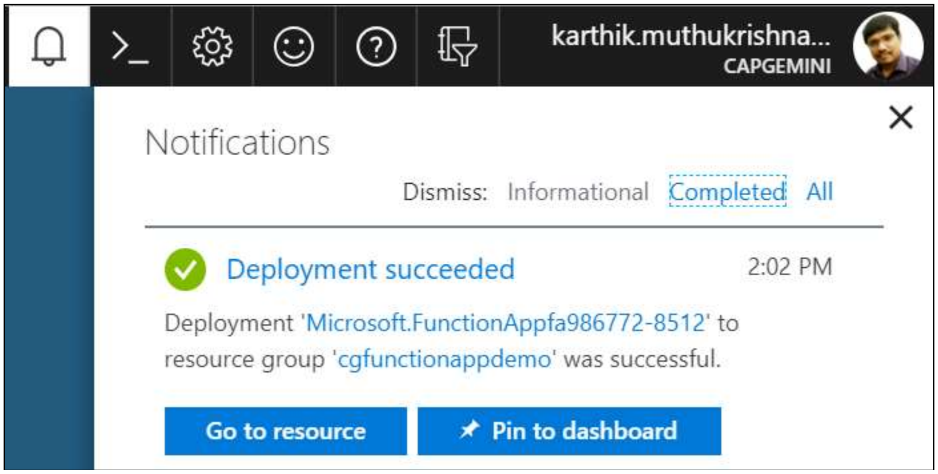
Application Insights: On | Off

□ Pin to dashboard

Create   Automation options

Step 4: Select Create to provision and deploy the function app

Step 5: Select the Notification icon in the upper-right corner of the portal and watch for the Deployment succeeded message.

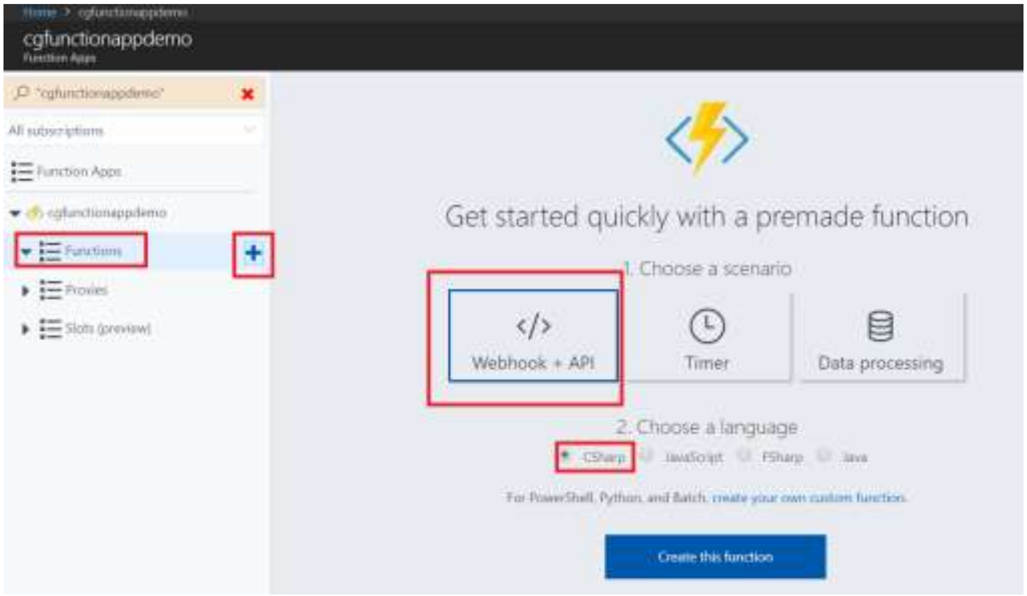


Deployment succeeded 2:02 PM

Deployment 'Microsoft.FunctionAppfa986772-8512' to resource group 'cgfunctionappdemo' was successful.

Step 6: Select Go to resource to view your new function app

Step 7: To create an HTTP triggered function expand new function app, then click the + button next to Functions.

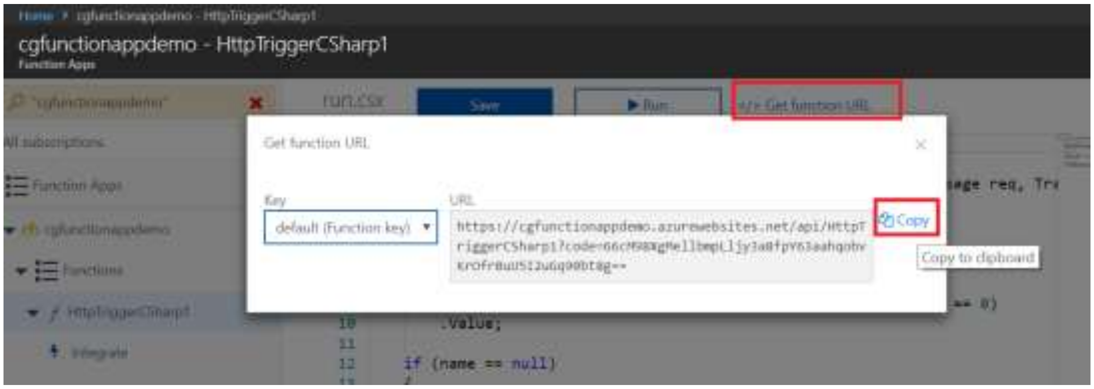


For PowerShell, Python, and Batch, create your own custom function.

Step 8: Select WebHook + API, choose a language for your function, and click Create this function.

A function is created in your chosen language C# using the template for an HTTP triggered function.

Step 9: In your new function, click </> Get function URL at the top right, select default (Function key), and then click Copy.



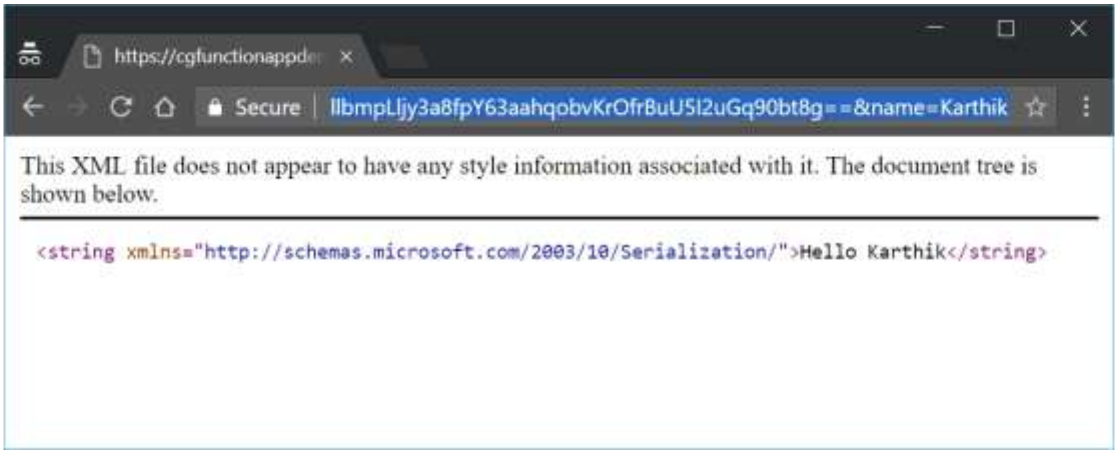
Get function URL

Key: default (Function key)

URL: https://cgfunctionappdemo.azurewebsites.net/api/HttpTriggerCSharp1?code=66cM98XgMe1lbmpLljy3a8fpY63aahqobvKrOfrBuU5I2uGq90bt8g==

Step 10: Paste the function URL into your browser's address bar. Add the query string value &name=<yourname> to the end of this URL and press the Enter key on your keyboard to execute the request.

The following image shows the response in the Chrome browser (other browsers like Edge browser may display it in JSON)

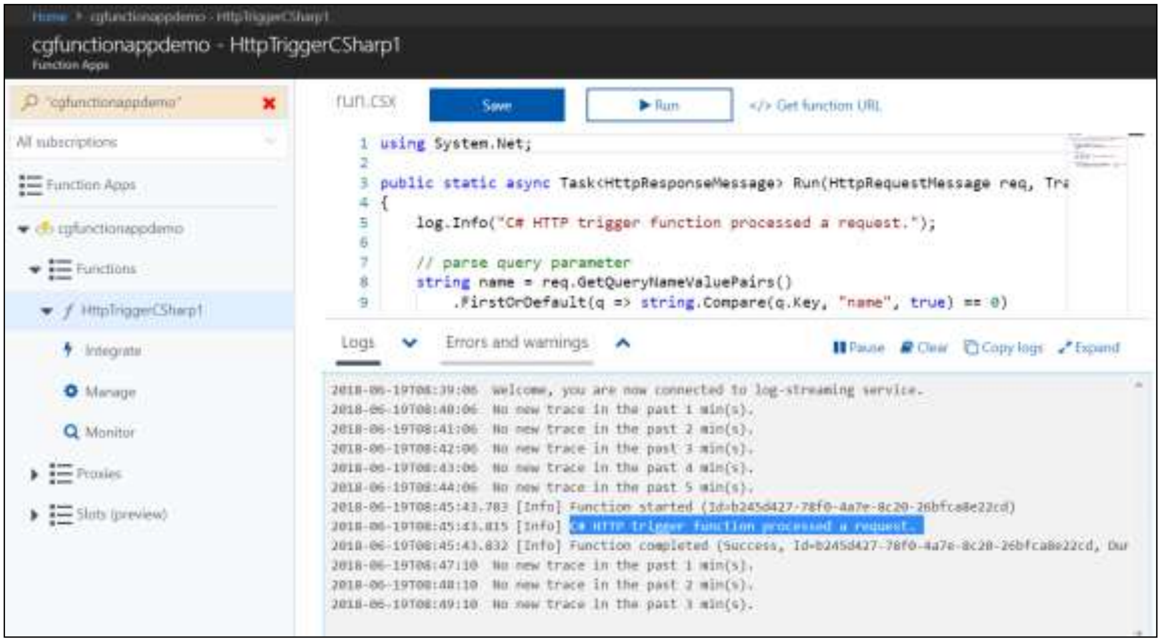


llbmpLljy3a8fpY63aahqobvKrOfrBuU5I2uGq90bt8g==&name=Karthik

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<string xmlns="http://schemas.microsoft.com/2003/10/Serialization/">Hello Karthik</string>
```

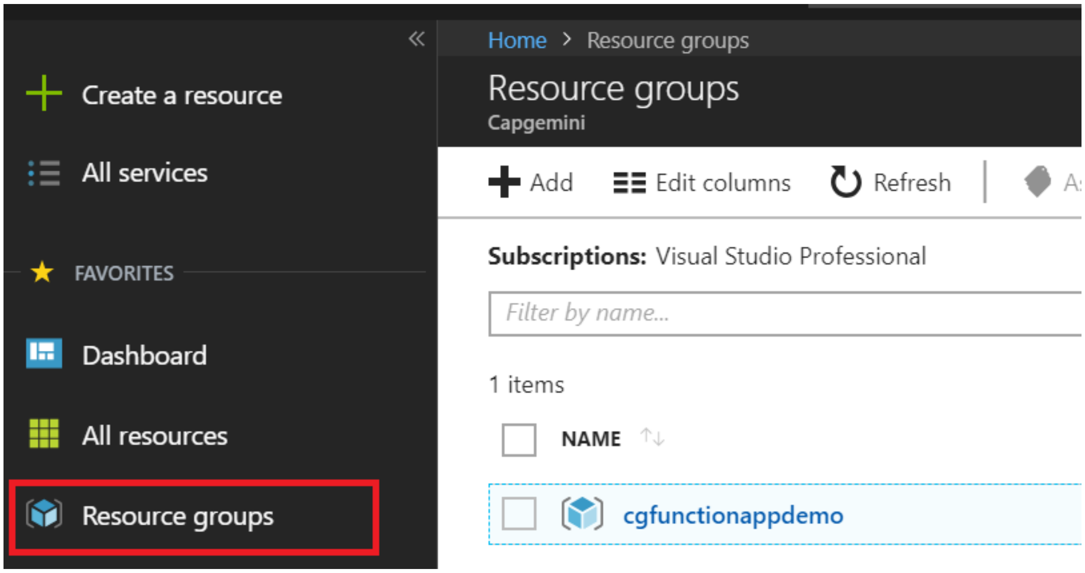
Step 11: When your function runs, trace information is written to the logs. To see the trace output from the previous execution, return to your function in the portal and click the arrow at the bottom of the screen to expand the Logs.



```
using System.Net;

public static async Task<HttpResponseMessage> Run(HttpRequestMessage req, Tra
{
    log.Info("C# HTTP trigger function processed a request.");

    // parse query parameter
    string name = req.GetQueryNameValuePairs()
        .FirstOrDefault(q => string.Compare(q.Key, "name", true) == 0)
```

```
2018-06-19T08:39:06  Welcome, you are now connected to log-streaming service.
2018-06-19T08:40:06  No new trace in the past 1 min(s).
2018-06-19T08:41:06  No new trace in the past 2 min(s).
2018-06-19T08:42:06  No new trace in the past 3 min(s).
2018-06-19T08:43:06  No new trace in the past 4 min(s).
2018-06-19T08:44:06  No new trace in the past 5 min(s).
2018-06-19T08:45:43.783 [Info] Function started (Id=b245d427-78f0-4a7e-8c20-26bfca8e22cd)
2018-06-19T08:45:43.815 [Info] C# HTTP trigger function processed a request.
2018-06-19T08:45:43.832 [Info] Function completed (Success, Id=b245d427-78f0-4a7e-8c20-26bfca8e22cd, Dur
2018-06-19T08:47:10  No new trace in the past 1 min(s).
2018-06-19T08:48:10  No new trace in the past 2 min(s).
2018-06-19T08:49:10  No new trace in the past 3 min(s).
```

Step 12: In the Azure portal, go to the Resource group page.

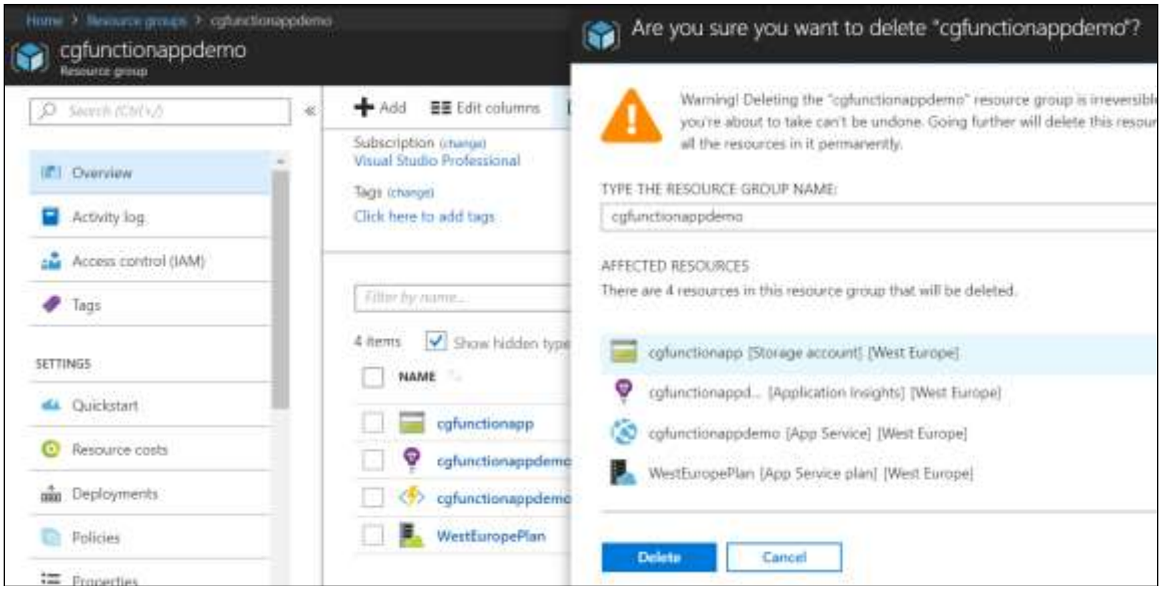


Subscriptions: Visual Studio Professional

1 items

□	NAME
□	cgfunctionappdemo

Step 13: In the Resource group page, review the list of included resources, and verify that they are the ones you want to delete



Are you sure you want to delete "cgfunctionappdemo"?

Warning! Deleting the "cgfunctionappdemo" resource group is irreversibl you're about to take can't be undone. Going further will delete this resour all the resources in it permanently.

TYPE THE RESOURCE GROUP NAME: cgfunctionappdemo

AFFECTED RESOURCES

There are 4 resources in this resource group that will be deleted.

cgfunctionapp [Storage account] [West Europe]
cgfunctionappd... [Application Insights] [West Europe]
cgfunctionappdemo [App Service] [West Europe]
WestEuropePlan [App Service plan] [West Europe]

Step 14: Select Delete resource group

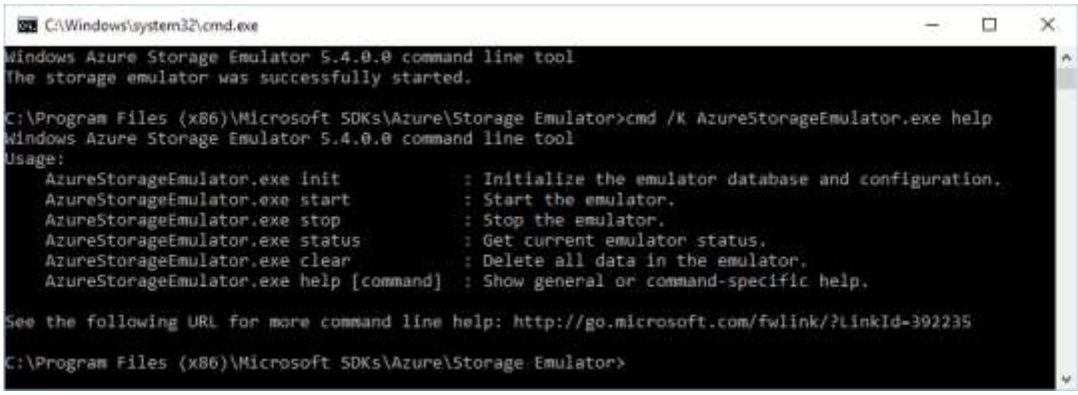
Lab 4. Creating a Container and Upload a Blob

Goals	Create a container and upload a blob using Azure SDK in Storage Emulator
Time	60 Mins

Step 1: Select the Start button or press the Windows key

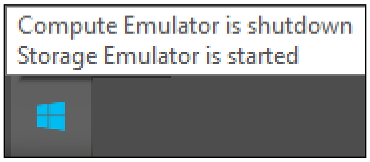
Step 2: Begin typing **Azure Storage Emulator**

Step 3: Select the emulator from the list of displayed applications to start it.



```
Windows Azure Storage Emulator 5.4.0.0 command line tool
The storage emulator was successfully started.

C:\Program Files (x86)\Microsoft SDKs\Azure\Storage Emulator>cmd /K AzureStorageEmulator.exe help
Windows Azure Storage Emulator 5.4.0.0 command line tool
Usage:
    AzureStorageEmulator.exe init             : Initialize the emulator database and configuration.
    AzureStorageEmulator.exe start            : Start the emulator.
    AzureStorageEmulator.exe stop             : Stop the emulator.
    AzureStorageEmulator.exe status           : Get current emulator status.
    AzureStorageEmulator.exe clear            : Delete all data in the emulator.
    AzureStorageEmulator.exe help [command]   : Show general or command-specific help.

See the following URL for more command line help: http://go.microsoft.com/fwlink/?LinkId=392235

C:\Program Files (x86)\Microsoft SDKs\Azure\Storage Emulator>
```

Step 4: Ensure that you'll see an icon in the Windows taskbar notification area while the emulator is running



Compute Emulator is shutdown
Storage Emulator is started

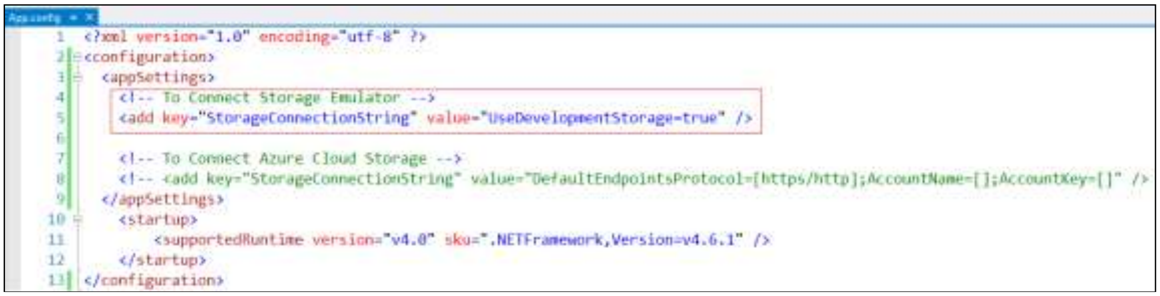
Step 5: Open Visual Studio 2017 and create C# Console application project named **AzureBlobStorage**

Step 6: Add the Nuget package **WindowsAzure.Storage (8.7.0)** to the solution

Step 7: Add reference to **System.Configuration.dll** to access configuration details from **App.Config** file

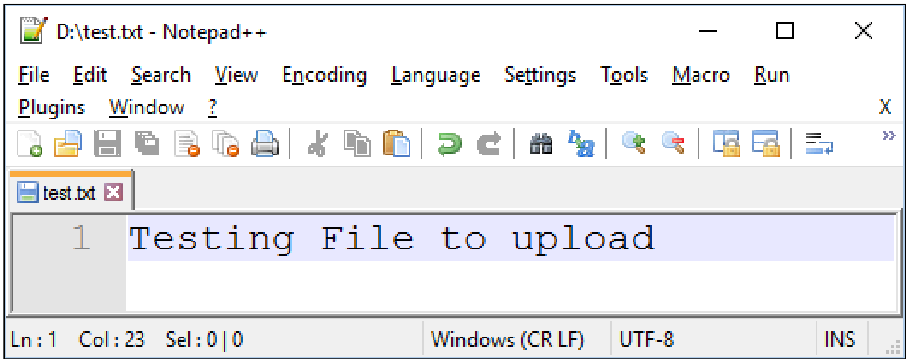
Step 8: To connect to the storage emulator from application, configure a connection string in application's configuration file

```
<add key="StorageConnectionString" value="UseDevelopmentStorage=true" />
```



```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!-- To Connect Storage Emulator -->
    <add key="StorageConnectionString" value="UseDevelopmentStorage=true" />

    <!-- To Connect Azure Cloud Storage -->
    <!-- <add key="StorageConnectionString" value="DefaultEndpointsProtocol=[https/http];AccountName=[];AccountKey=[]" />
  </appSettings>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
    </startup>
</configuration>
```

Step 9: Create file named **test.txt** in d:\



D:\test.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run
Plugins Window ?
X
test.txt

```
Testing File to upload
```

Ln : 1 Col : 23 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS

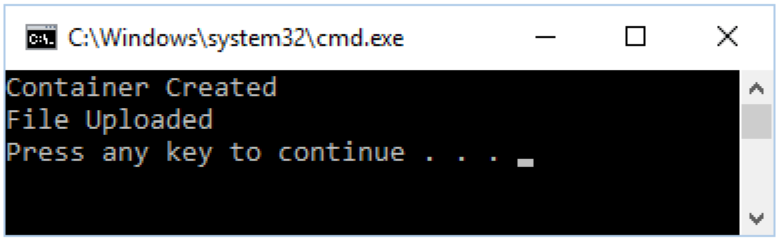
Step 10: Add the following namespaces in program.cs

```
using System;
using System.Configuration;
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Blob;
```

Step 11: Add the following code snippet in program.cs to create a container named **my-file** and upload a file under a blob named **test-blob**

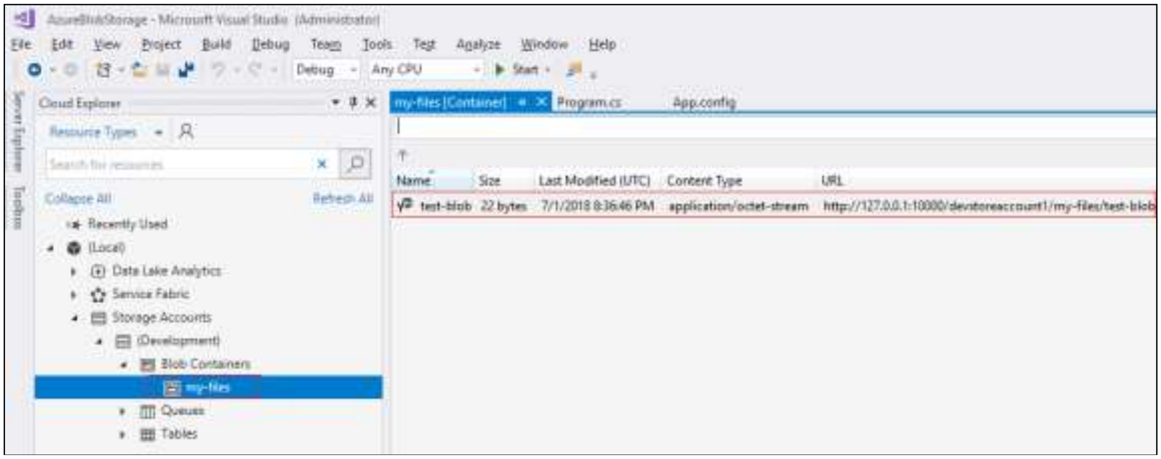
```
namespace AzureBlobStorage
{
   class Program
   {
      static void Main(string[] args)
      {
         try
         {
            CloudStorageAccount storageAccount =
CloudStorageAccount.Parse(ConfigurationManager.AppSettings["StorageConnectionString"]);

            CloudBlobClient blobClient = storageAccount.CreateCloudBlobClient();

            CloudBlobContainer container = blobClient.GetContainerReference("my-files");

            container.CreateIfNotExists();

            Console.WriteLine("Container Created");

            CloudBlockBlob blob = container.GetBlockBlobReference("test-blob");

            blob.UploadFromFile(@"d:\test.txt");

            Console.WriteLine("File Uploaded");
         }
         catch (StorageException ex)
         {
            Console.WriteLine(ex.Message);
         }
      }
   }
}
```

Step 11: Run the program CTRL + F5



```
Container Created
File Uploaded
Press any key to continue . . .
```

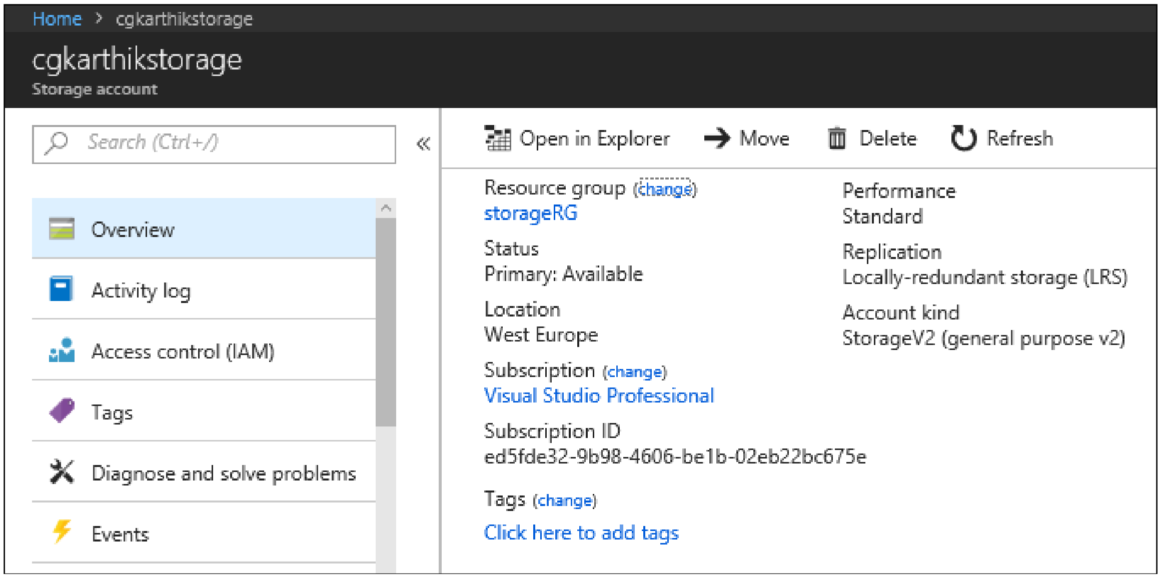
Step 12: Click View → Cloud Explorer and refresh the Blob containers to see our container named my-files and open the container to see the blob with the file that we have uploaded.



Name	Size	Last Modified (UTC)	Content Type	URL
test-blob	22 bytes	7/1/2018 8:36:46 PM	application/octet-stream	http://127.0.0.1:10000/devstoreaccount1/my-files/test-blob

Stretched Assignment:

Step 1: Create a storage account (change it with a unique name) as given below



cgkarthikstorage
Storage account

Resource group (change)
storageRG

Status
Primary: Available

Location
West Europe

Subscription (change)
Visual Studio Professional

Subscription ID
ed5fde32-9b98-4606-be1b-02eb22bc675e

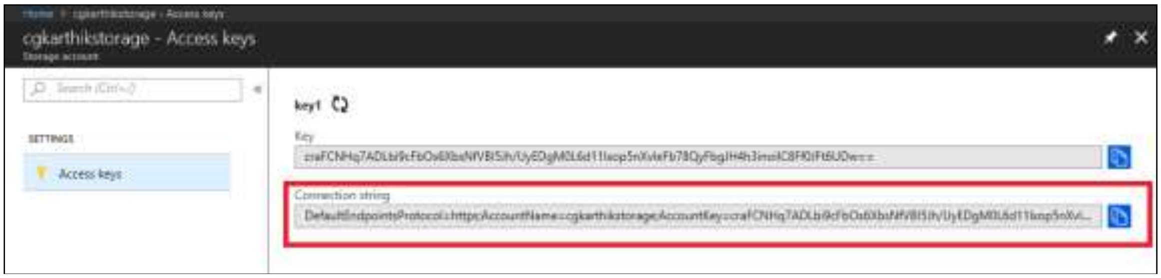
Tags (change)
Click here to add tags

Performance
Standard

Replication
Locally-redundant storage (LRS)

Account kind
StorageV2 (general purpose v2)

Step 2: Replace the appSettings by copying the Connection string value from Access Keys and run the program to create the blob in Azure Cloud Storage.

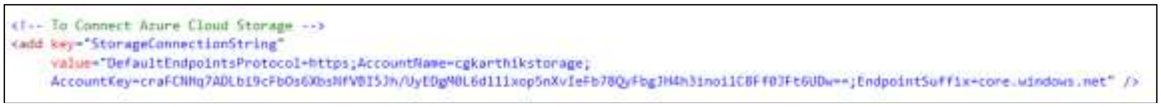


cgkarthikstorage - Access keys
Storage account

key1

Key
craFCNHq7ADLbi9cFbOs6XbsNfVBI5Jh/UyEDgM0L6d11xop5nXvIeFb78QyFbgJH4h3inoi1C8Ff0JFt6UDw==

Connection string
DefaultEndpointsProtocol=https;AccountName=cgkarthikstorage;AccountKey=craFCNHq7ADLbi9cFbOs6XbsNfVBI5Jh/UyEDgM0L6d11xop5nXvI...



```
<!-- To Connect Azure Cloud Storage -->
<add key="StorageConnectionString"
     value="DefaultEndpointsProtocol=https;AccountName=cgkarthikstorage;
     AccountKey=craFCNHq7ADLbi9cFbOs6XbsNfVBI5Jh/UyEDgM0L6d11xop5nXvIeFb78QyFbgJH4h3inoi1C8Ff0JFt6UDw==;EndpointSuffix=core.windows.net" />
```

Lab 5. CRUD Operation in Azure Table Storage using C#

Goals	Perform CRUD Operation in Azure Table Storage using Azure SDK in Storage Emulator through C# Console Application
Time	90 Mins

Step 1: Select the Start button or press the Windows key

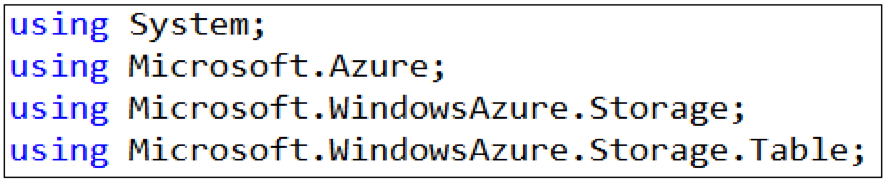
Step 2: Begin typing Azure Storage Emulator

Step 3: Select the emulator from the list of displayed applications to start it.

Step 4: Open Visual Studio 2017 and create C# Console application project named **AzureTableStorage**

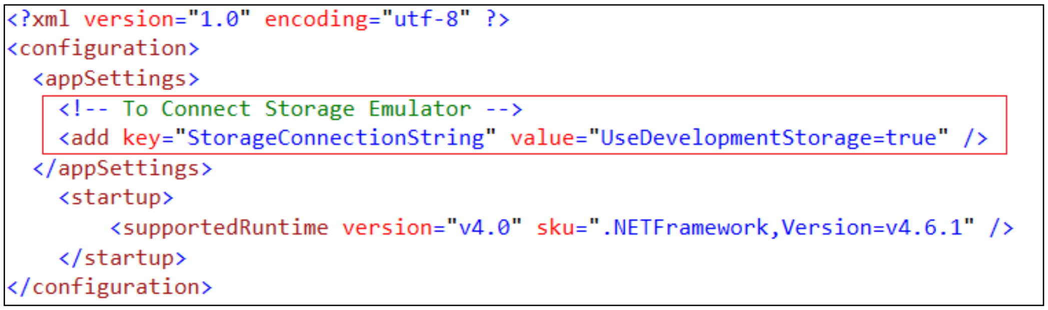
Step 5: Add the Nuget package **WindowsAzure.Storage (9.3.0/8.7.0) and Microsoft.WindowsAzure.ConfigurationManager** (3.2.3) to the solution

Step 6: Add the following namespaces in **program.cs**



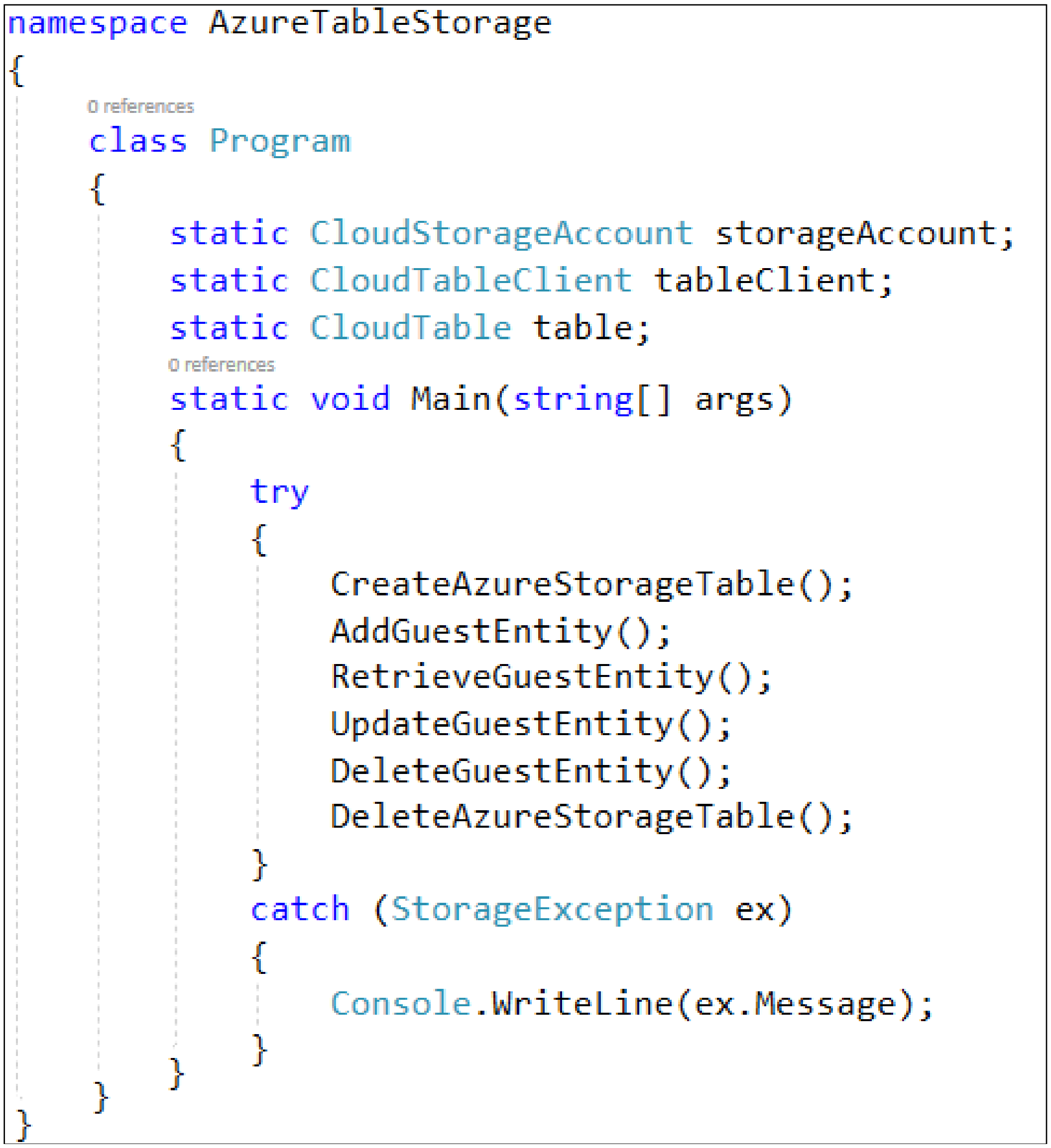
```
using System;
using Microsoft.Azure;
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Table;
```

Step 7: To connect to the storage emulator from application, configure a connection string in application's configuration file



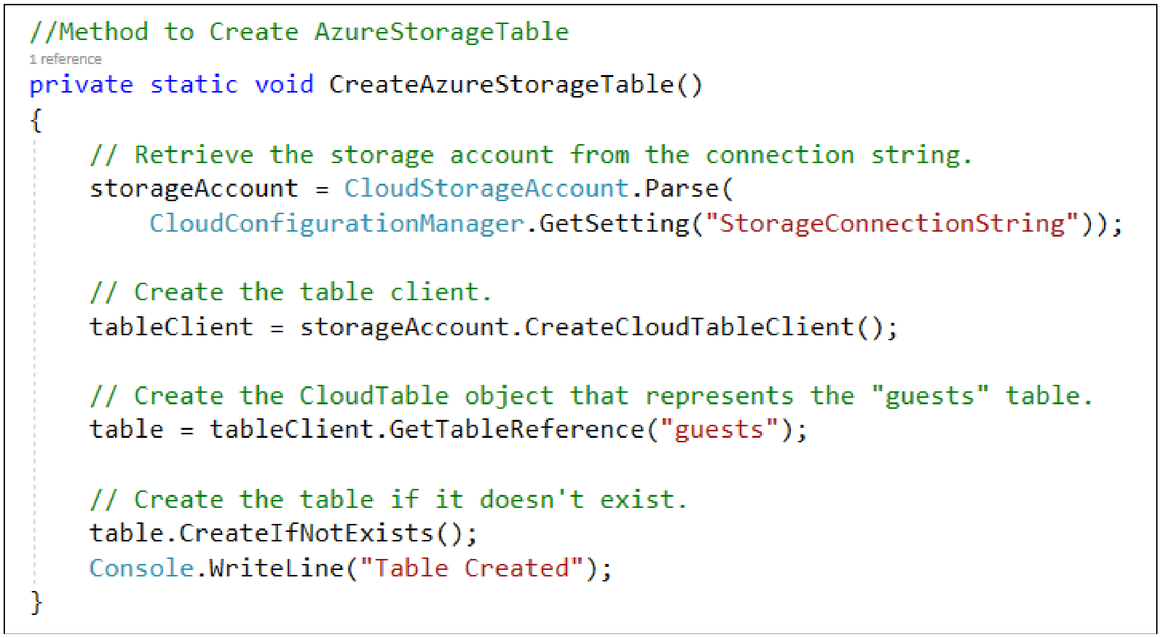
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!-- To Connect Storage Emulator -->
    <add key="StorageConnectionString" value="UseDevelopmentStorage=true" />
  </appSettings>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
    </startup>
</configuration>
```

Step 8: Add the following code snippet in **program.cs** and generate the methods



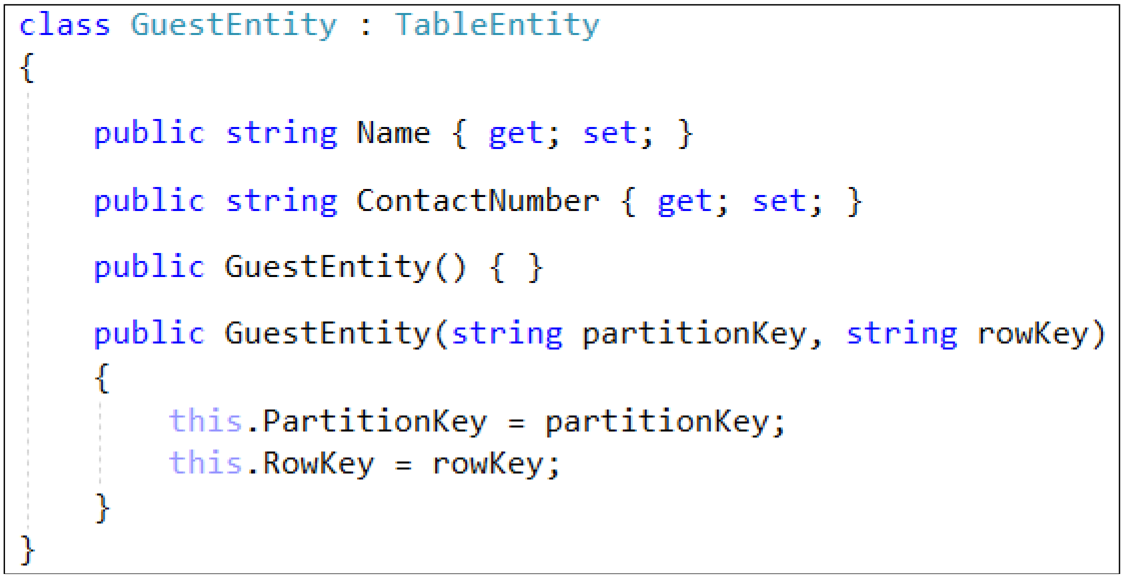
```
namespace AzureTableStorage
{
    0 references
    class Program
    {
        static CloudStorageAccount storageAccount;
        static CloudTableClient tableClient;
        static CloudTable table;
        0 references
        static void Main(string[] args)
        {
            try
            {
                CreateAzureStorageTable();
                AddGuestEntity();
                RetrieveGuestEntity();
                UpdateGuestEntity();
                DeleteGuestEntity();
                DeleteAzureStorageTable();
            }
            catch (StorageException ex)
            {
                Console.WriteLine(ex.Message);
            }
        }
    }
}
```

Step 9: Add the following code snippet in to create azure table storage



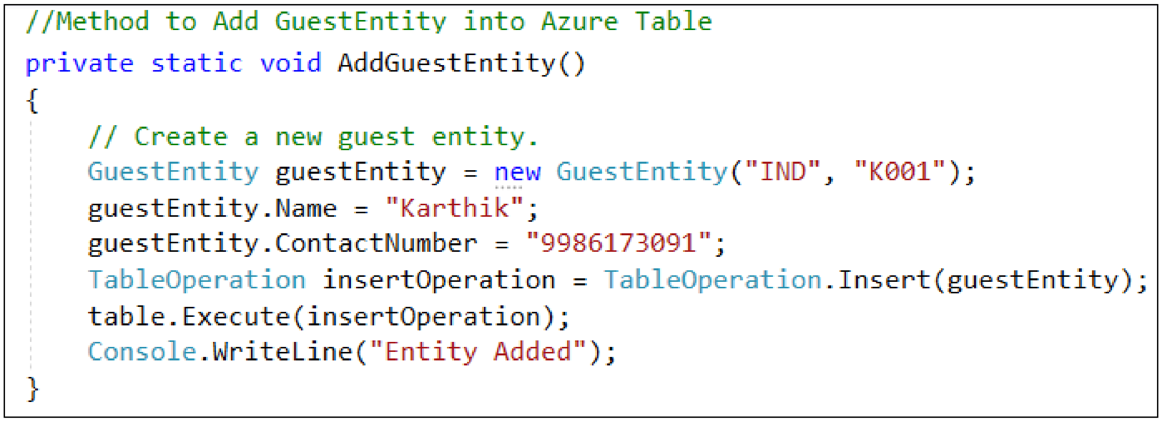
```
//Method to Create AzureStorageTable
private static void CreateAzureStorageTable()
{
    // Retrieve the storage account from the connection string.
    storageAccount = CloudStorageAccount.Parse(
        CloudConfigurationManager.GetSetting("StorageConnectionString"));

    // Create the table client.
    tableClient = storageAccount.CreateCloudTableClient();

    // Create the CloudTable object that represents the "guests" table.
    table = tableClient.GetTableReference("guests");

    // Create the table if it doesn't exist.
    table.CreateIfNotExists();
    Console.WriteLine("Table Created");
}
```

Step 10: Add the following code snippet to create **GuestEntity** derived from **TableEntity**



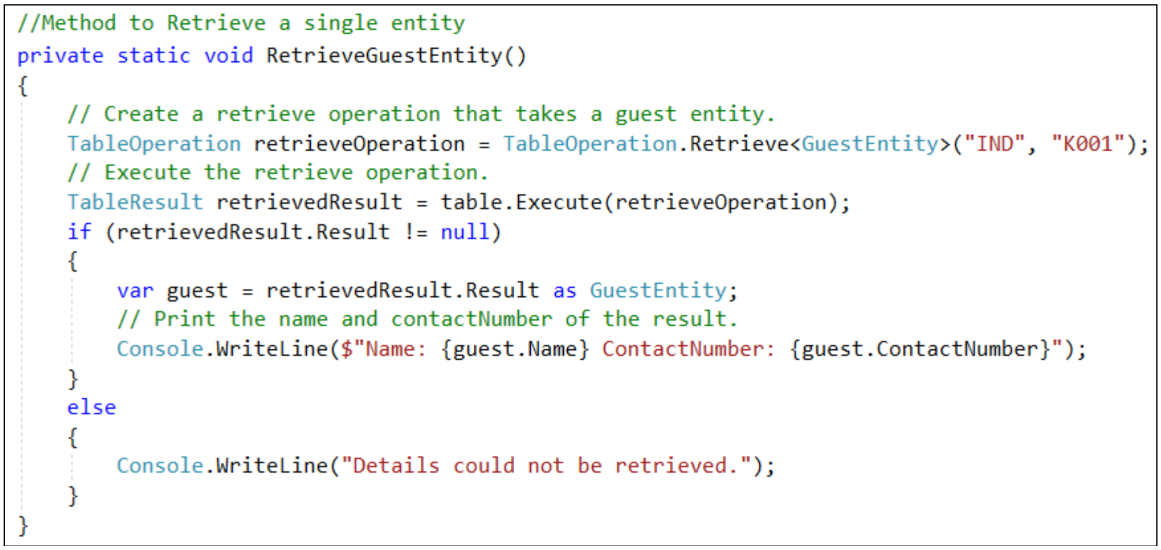
```
class GuestEntity : TableEntity
{

    public string Name { get; set; }

    public string ContactNumber { get; set; }

    public GuestEntity() { }

    public GuestEntity(string partitionKey, string rowKey)
    {
        this.PartitionKey = partitionKey;
        this.RowKey = rowKey;
    }
}
```

Step 11: Add the following code snippet to add Guest Entity into azure table



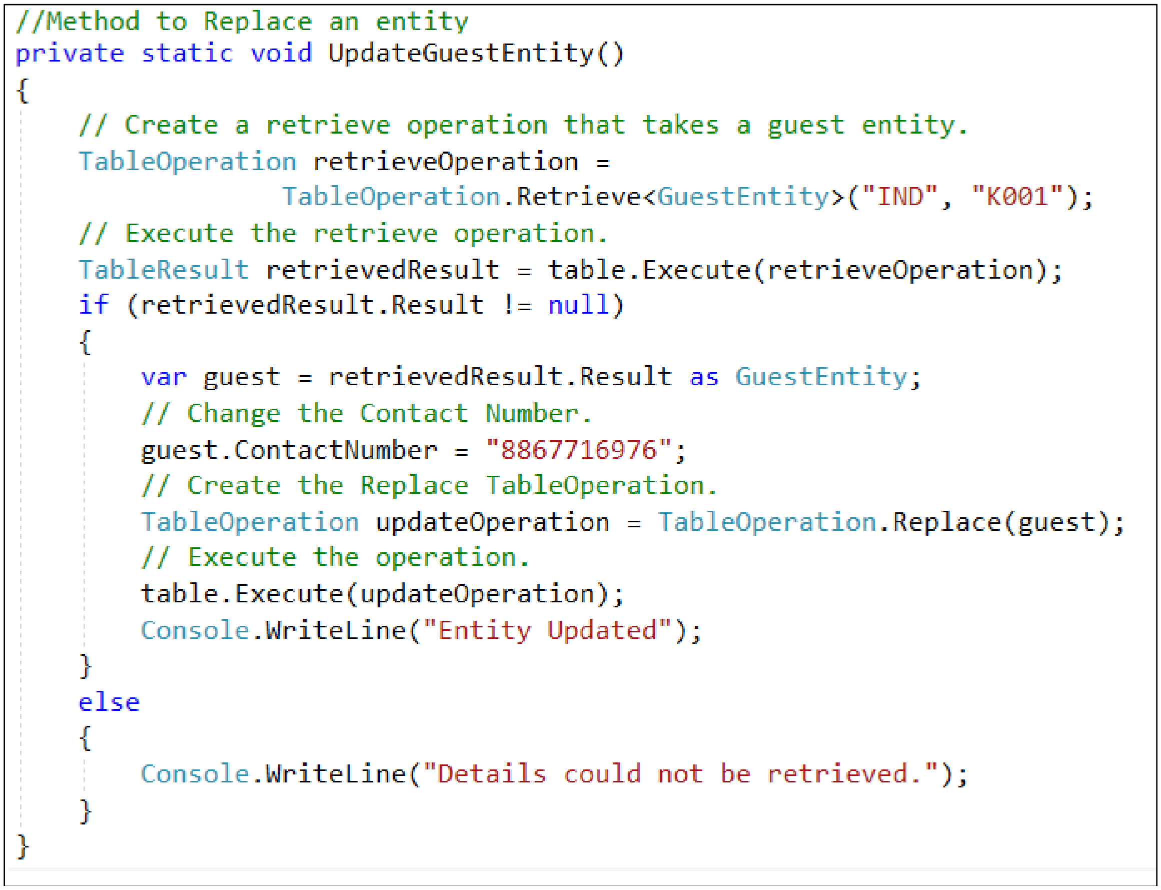
```
//Method to Add GuestEntity into Azure Table
private static void AddGuestEntity()
{
    // Create a new guest entity.
    GuestEntity guestEntity = new GuestEntity("IND", "K001");
    guestEntity.Name = "Karthik";
    guestEntity.ContactNumber = "9986173091";
    TableOperation insertOperation = TableOperation.Insert(guestEntity);
    table.Execute(insertOperation);
    Console.WriteLine("Entity Added");
}
```

Step 12: Add the following code snippet to retrieve specific Guest Entity from azure table



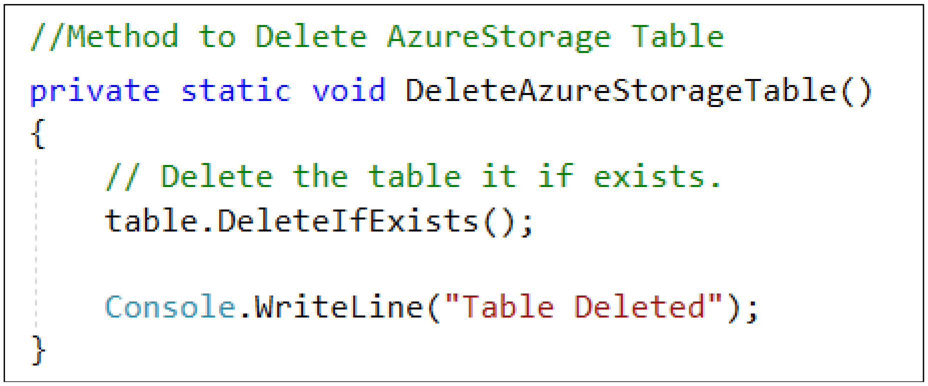
```
//Method to Retrieve a single entity
private static void RetrieveGuestEntity()
{
    // Create a retrieve operation that takes a guest entity.
    TableOperation retrieveOperation = TableOperation.Retrieve<GuestEntity>("IND", "K001");
    // Execute the retrieve operation.
    TableResult retrievedResult = table.Execute(retrieveOperation);
    if (retrievedResult.Result != null)
    {
        var guest = retrievedResult.Result as GuestEntity;
        // Print the name and contactNumber of the result.
        Console.WriteLine($"Name: {guest.Name} ContactNumber: {guest.ContactNumber}");
    }
    else
    {
        Console.WriteLine("Details could not be retrieved.");
    }
}
```

Step 13: Add the following code snippet to retrieve specific Guest Entity from azure table



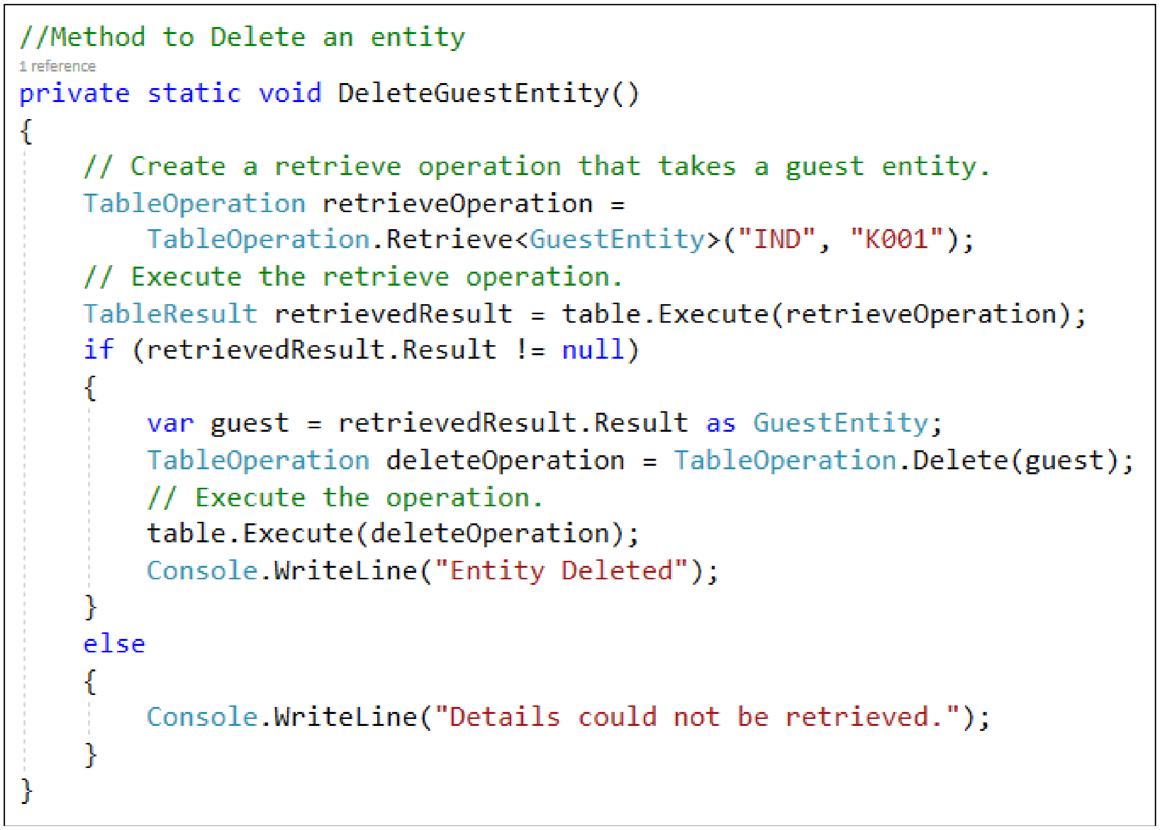
```
//Method to Replace an entity
private static void UpdateGuestEntity()
{
    // Create a retrieve operation that takes a guest entity.
    TableOperation retrieveOperation =
                 TableOperation.Retrieve<GuestEntity>("IND", "K001");
    // Execute the retrieve operation.
    TableResult retrievedResult = table.Execute(retrieveOperation);
    if (retrievedResult.Result != null)
    {
        var guest = retrievedResult.Result as GuestEntity;
        // Change the Contact Number.
        guest.ContactNumber = "8867716976";
        // Create the Replace TableOperation.
        TableOperation updateOperation = TableOperation.Replace(guest);
        // Execute the operation.
        table.Execute(updateOperation);
        Console.WriteLine("Entity Updated");
    }
    else
    {
        Console.WriteLine("Details could not be retrieved.");
    }
}
```

Step 14: Add the following code snippet to delete azure table



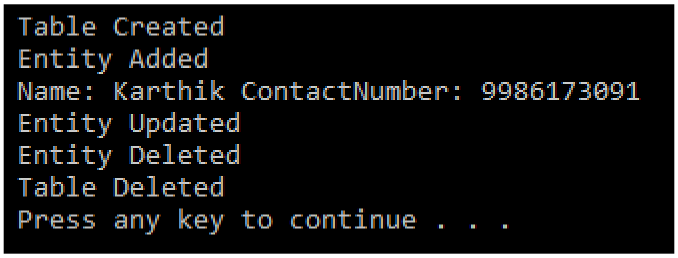
```
//Method to Delete AzureStorage Table
private static void DeleteAzureStorageTable()
{
    // Delete the table it if exists.
    table.DeleteIfExists();

    Console.WriteLine("Table Deleted");
}
```

Step 15: Add the following code snippet to delete a specific entity from azure table



```
//Method to Delete an entity
1 reference
private static void DeleteGuestEntity()
{
    // Create a retrieve operation that takes a guest entity.
    TableOperation retrieveOperation =
        TableOperation.Retrieve<GuestEntity>("IND", "K001");
    // Execute the retrieve operation.
    TableResult retrievedResult = table.Execute(retrieveOperation);
    if (retrievedResult.Result != null)
    {
        var guest = retrievedResult.Result as GuestEntity;
        TableOperation deleteOperation = TableOperation.Delete(guest);
        // Execute the operation.
        table.Execute(deleteOperation);
        Console.WriteLine("Entity Deleted");
    }
    else
    {
        Console.WriteLine("Details could not be retrieved.");
    }
}
```

Step 16: Run the Program CTRL + F5 to see the results



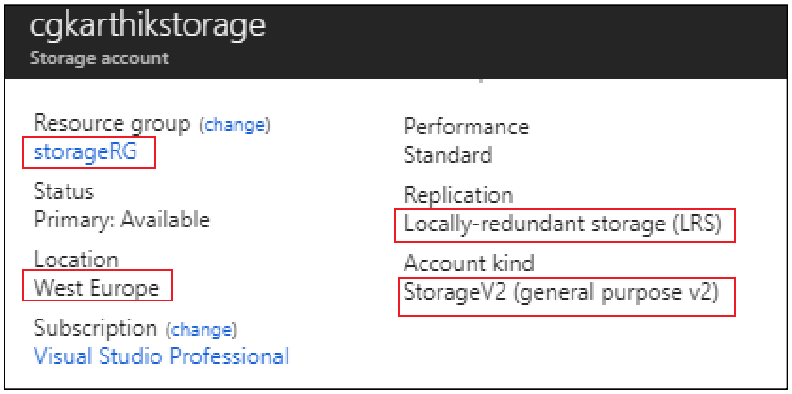
```
Table Created
Entity Added
Name: Karthik ContactNumber: 9986173091
Entity Updated
Entity Deleted
Table Deleted
Press any key to continue . . .
```

Lab 6. Creating a Logic App

Goals	Create a Logic App using Logic App Designer in Azure portal which triggers an action when a new message is added to queue
Time	60 Mins

Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Create a Storage Account as shown below



cgkarthikstorage
Storage account

Resource group (change)
storageRG

Status
Primary: Available

Location
West Europe

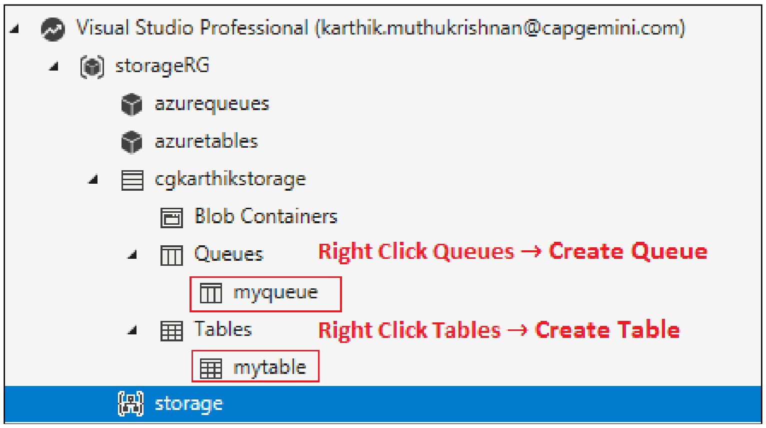
Subscription (change)
Visual Studio Professional

Performance
Standard

Replication
Locally-redundant storage (LRS)

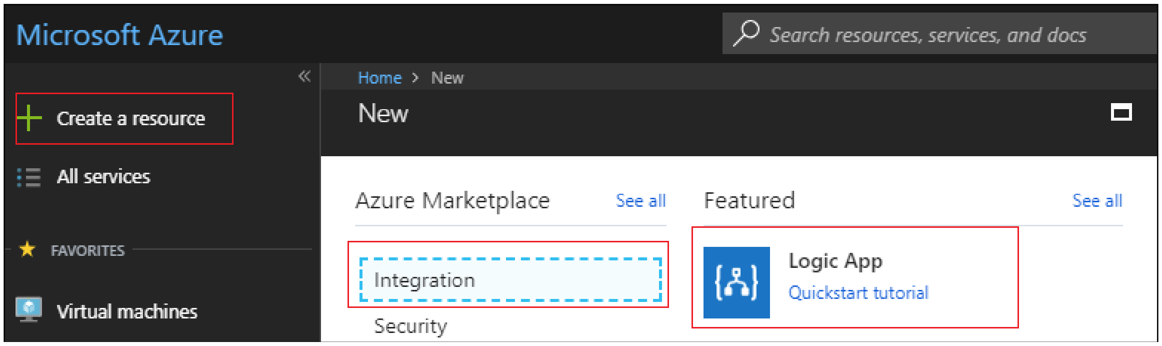
Account kind
StorageV2 (general purpose v2)

Step 3: Open Visual Studio 2017 → View → Cloud Explorer and create Queue named **myqueue** and table named **mytable** under your subscriptions



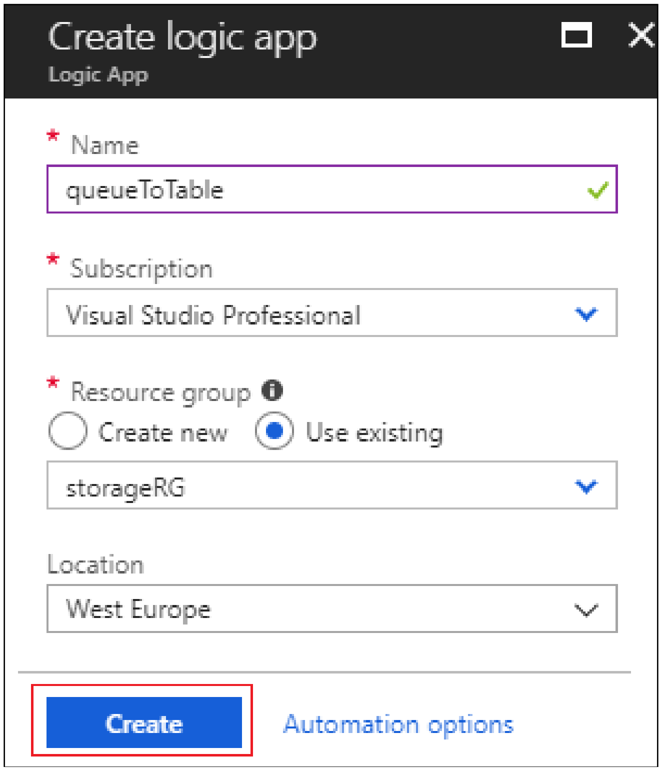
```
Visual Studio Professional (karthik.muthukrishnan@capgemini.com)
  storageRG
    azurequeues
    azuretables
    cgkarthikstorage
      Blob Containers
      Queues        Right Click Queues → Create Queue
        myqueue
      Tables        Right Click Tables → Create Table
        mytable
    storage
```

Step 4: Click Create a resource → Integration → Logic App to create a logic App



Step 5: Click Create a resource → Integration → Logic App to create a logic App

Step 6: Create a Logic App named **queueToTable** as shown below



* Name: queueToTable

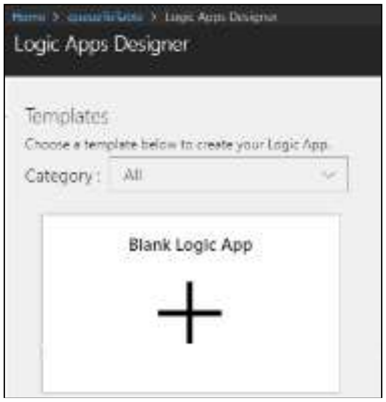
* Subscription: Visual Studio Professional

* Resource group: ○ Create new ● Use existing

storageRG

Location: West Europe

Step 7: Once the deployment succeeded, click **Go to Resources** to open the Logic Apps Designer and select Blank Logic App to open the designer



Home > [illegible] > Logic Apps Designer

Logic Apps Designer

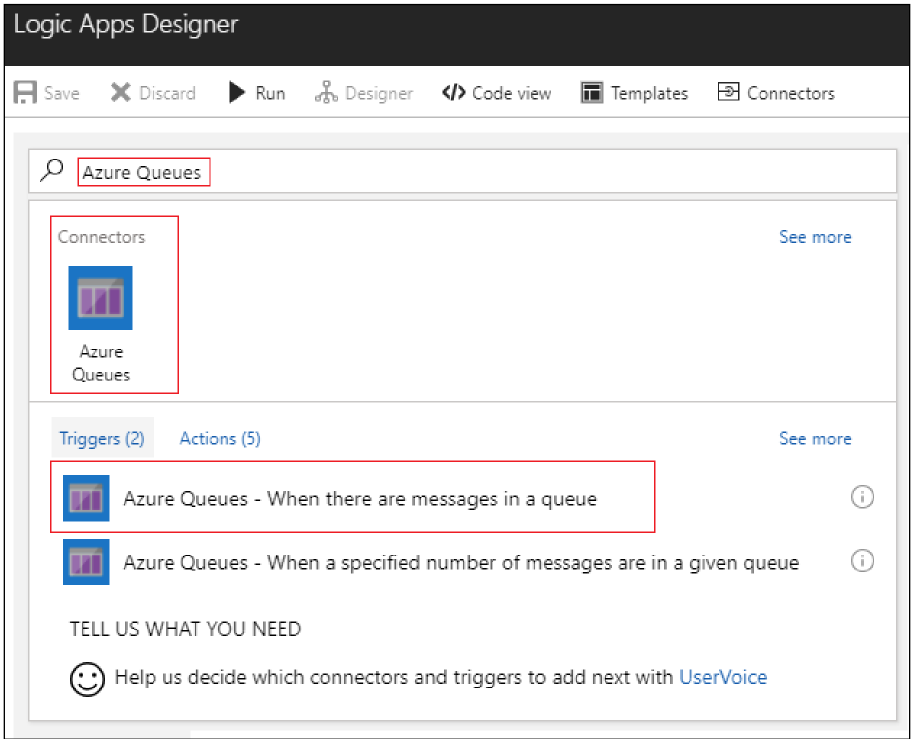
Templates

Choose a template below to create your Logic App

Category : All

Blank Logic App

Step 8: Search for the connectors **Azure Queues** and select the trigger **When there are messages in a queue**



Logic Apps Designer

Azure Queues

Connectors

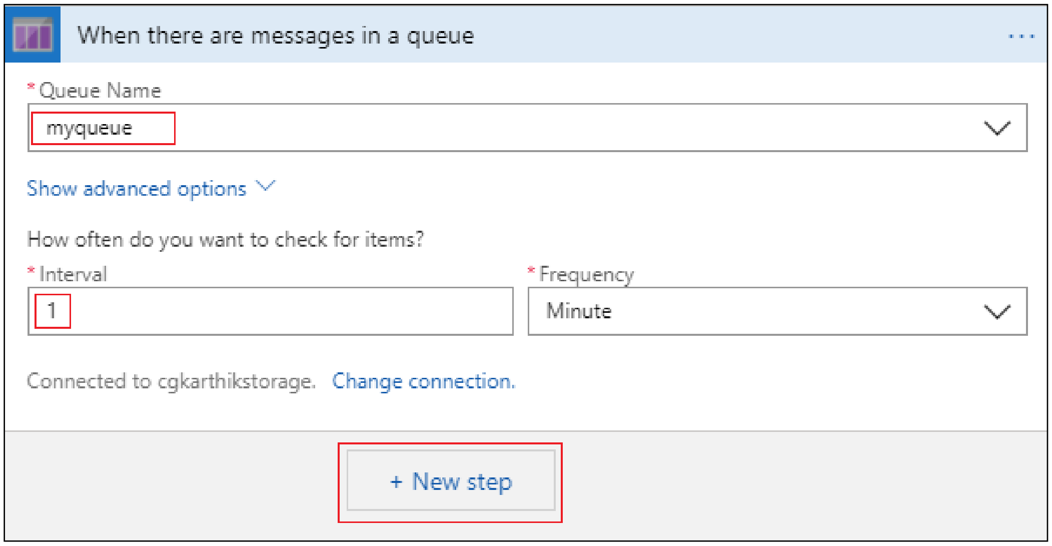
Azure Queues - When there are messages in a queue

Azure Queues - When a specified number of messages are in a given queue

TELL US WHAT YOU NEED

Help us decide which connectors and triggers to add next with UserVoice

Step 9: Select the queue that we have created through cloud explorer named **myqueue** which trigger the action for every 1 minute.



When there are messages in a queue

* Queue Name: myqueue

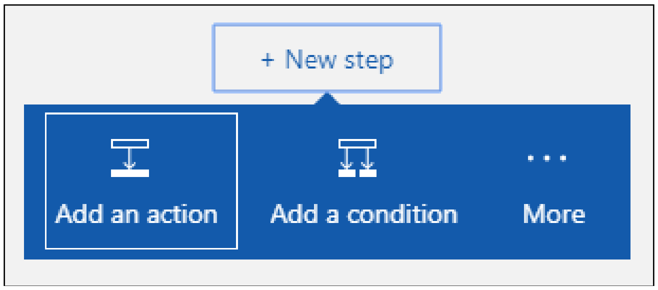
How often do you want to check for items?

* Interval: 1

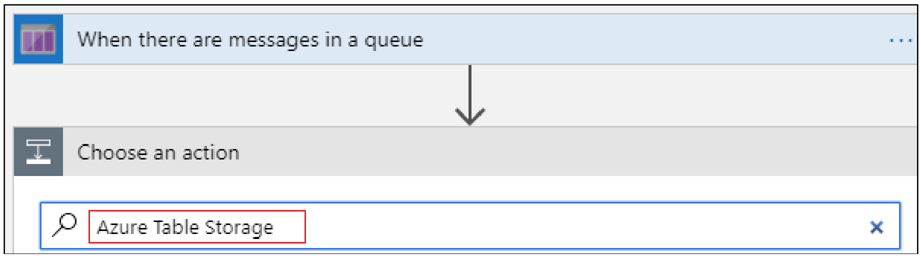
* Frequency: Minute

Connected to cgkarthikstorage. Change connection.

Step 10: Click New Step to Add an action



Step 11: Search for Azure Table Storage in the new action

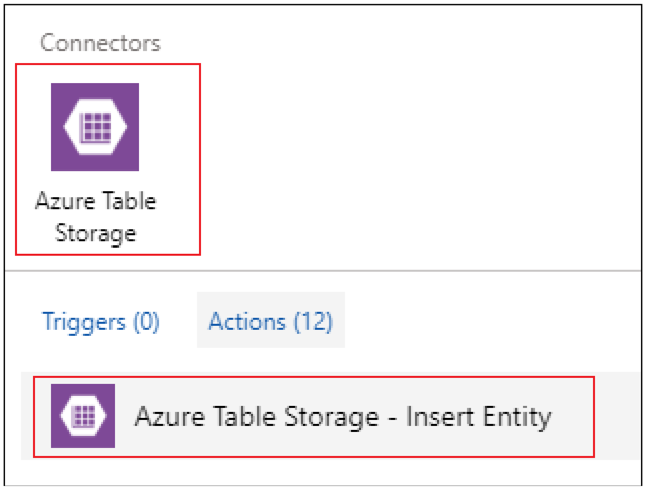


When there are messages in a queue

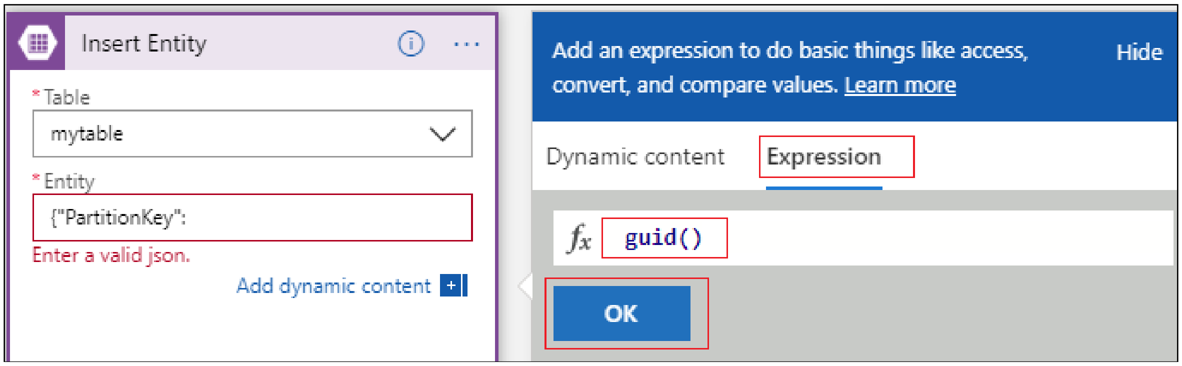
Choose an action

Azure Table Storage

Step 12: Select the connector **Azure Table Storage then** select the action **Azure Table Storage – Insert Entity**



Step 13: Select the table **myTable** and create the entity using Dynamic content and Expression as given below



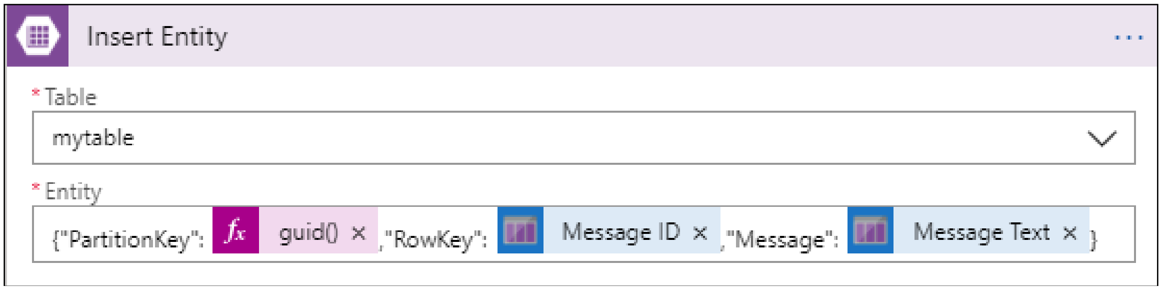
* Table: mytable

* Entity: {"PartitionKey":

Enter a valid json.

Add an expression to do basic things like access, convert, and compare values. Learn more

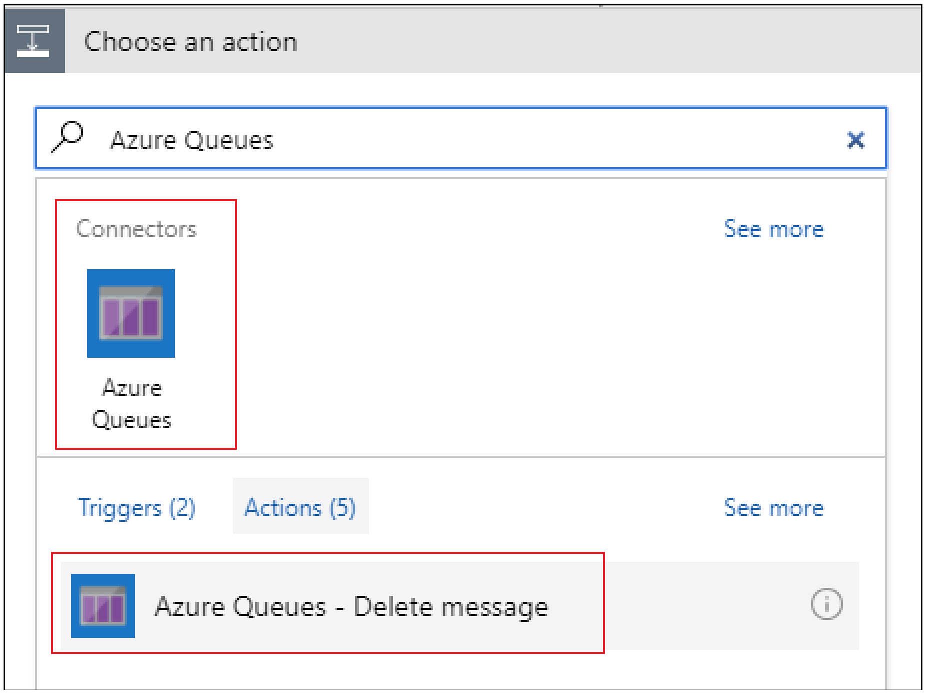
fx guid()



* Table: mytable

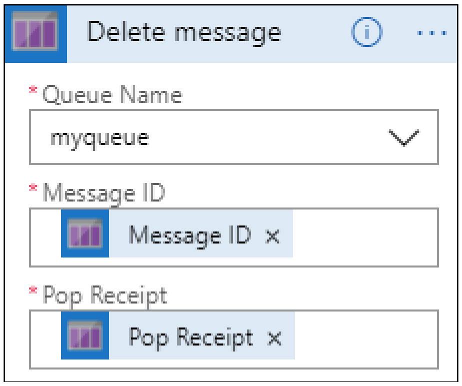
* Entity: {"PartitionKey": *fx* guid() ,"RowKey": Message ID ,"Message": Message Text }

Step 13: Click **New Step** again and Select **Add an Action** to choose and action **Azure Queues – Delete message**


Choose an action

Azure Queues

Step 14: Complete the action to delete the message from queue after inserting it in the Azure table

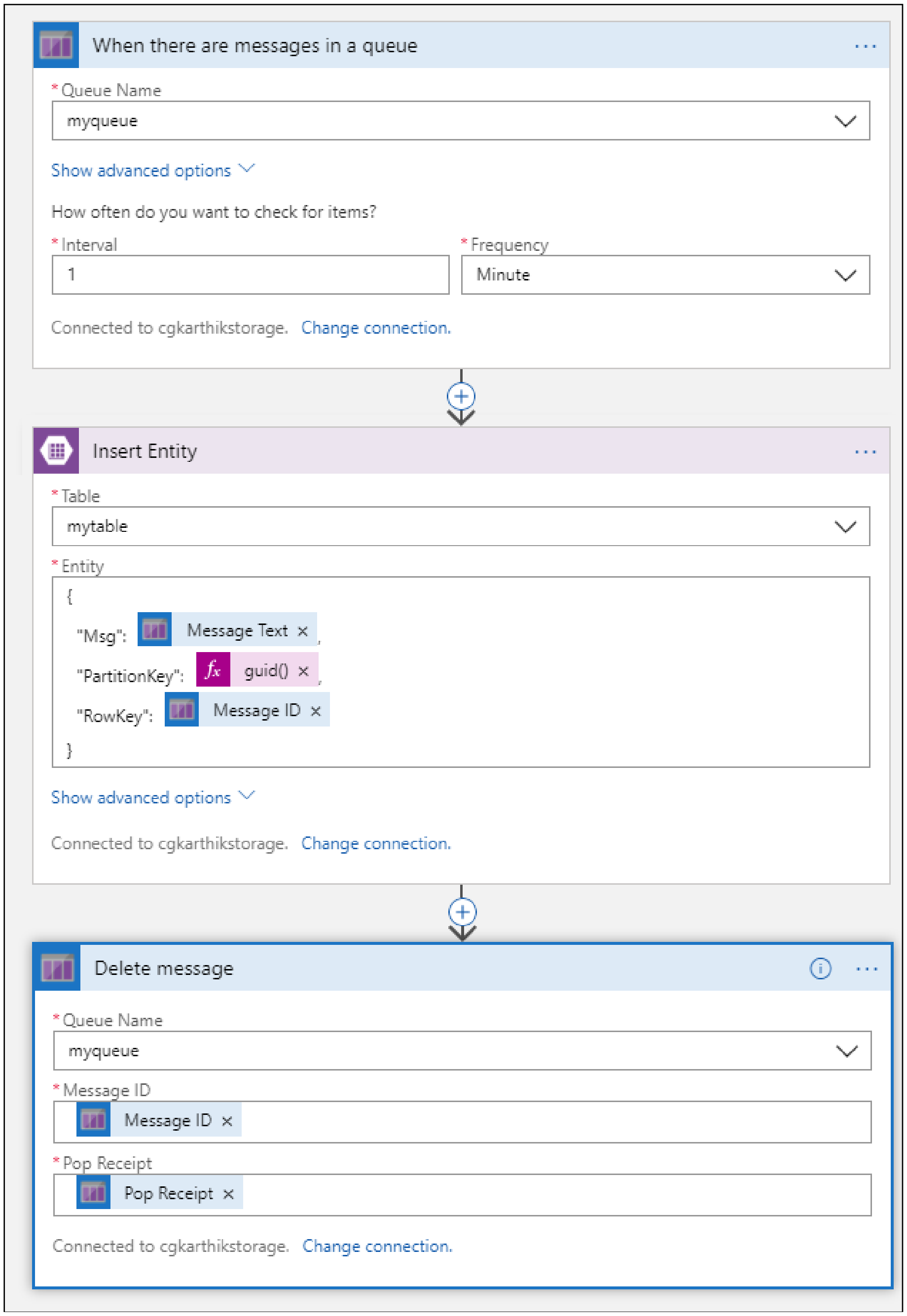

Delete message

* Queue Name: myqueue

* Message ID: Message ID

* Pop Receipt: Pop Receipt

Step 15: Save the Logic App. Completed design looks like the image given below



When there are messages in a queue

*Queue Name: myqueue

Show advanced options

How often do you want to check for items?

*Interval: 1

*Frequency: Minute

Connected to cgkarthikstorage. Change connection.

Insert Entity

*Table: mytable

*Entity:

```
{
  "Msg": Message Text ×,
  "PartitionKey": guid() ×,
  "RowKey": Message ID ×
}
```

Show advanced options

Connected to cgkarthikstorage. Change connection.

Delete message

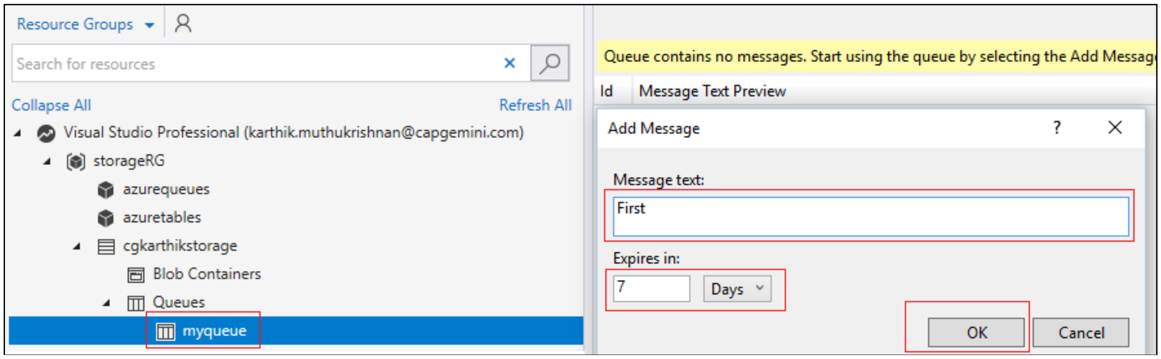
*Queue Name: myqueue

*Message ID: Message ID ×

*Pop Receipt: Pop Receipt ×

Connected to cgkarthikstorage. Change connection.

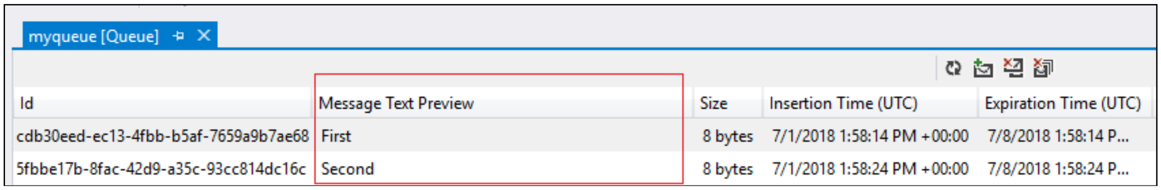
Step 16: Open Cloud Explorer and Add Messages which expires in 7 days



Queue contains no messages. Start using the queue by selecting the Add Messag

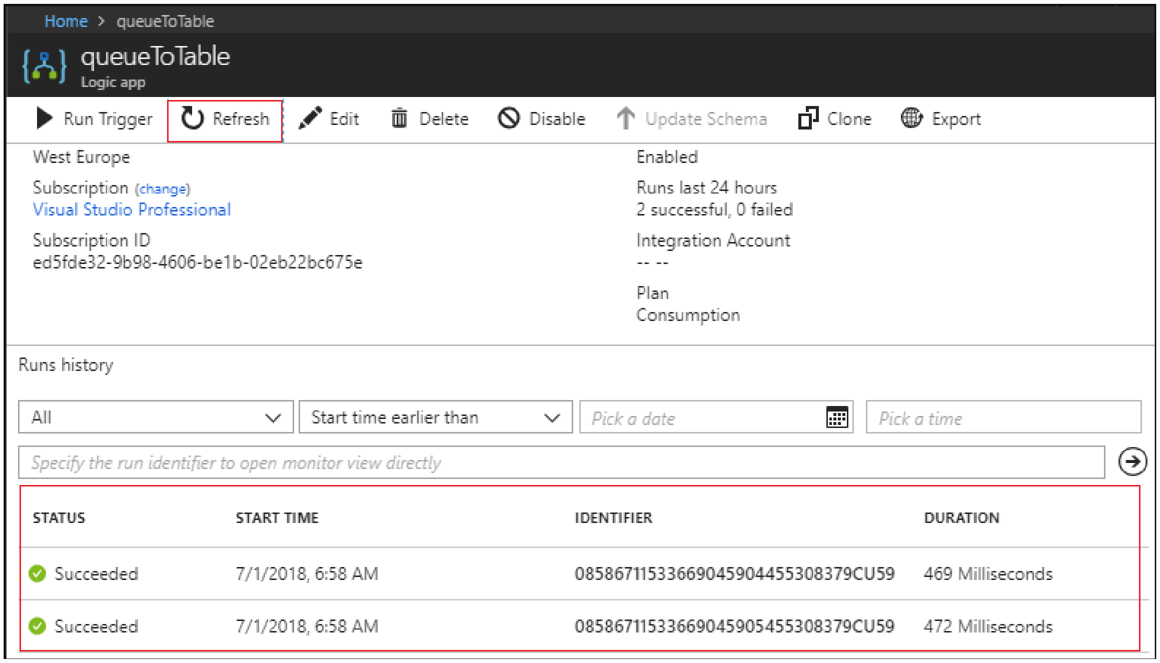
Message text: First

Expires in: 7 Days



Id	Message Text Preview	Size	Insertion Time (UTC)	Expiration Time (UTC)
cdb30eed-ec13-4fbb-b5af-7659a9b7ae68	First	8 bytes	7/1/2018 1:58:14 PM +00:00	7/8/2018 1:58:14 P...
5fbbe17b-8fac-42d9-a35c-93cc814dc16c	Second	8 bytes	7/1/2018 1:58:24 PM +00:00	7/8/2018 1:58:24 P...

Step 17: Logic App triggers the Action when new message is added to queue



West Europe

Subscription (change)
Visual Studio Professional

Subscription ID
ed5fde32-9b98-4606-be1b-02eb22bc675e

Enabled

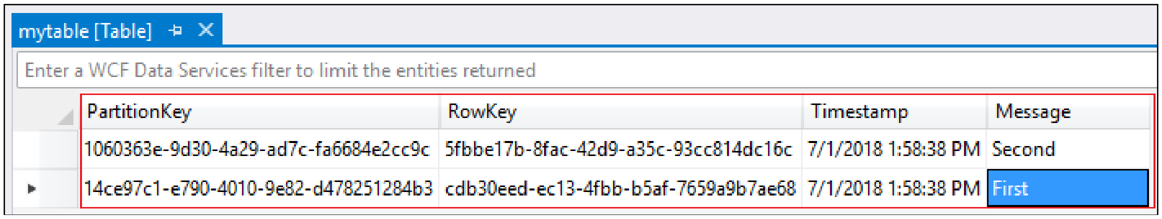
Runs last 24 hours
2 successful, 0 failed

Integration Account
-- --

Plan
Consumption

STATUS	START TIME	IDENTIFIER	DURATION
✓ Succeeded	7/1/2018, 6:58 AM	08586711533669045904455308379CU59	469 Milliseconds
✓ Succeeded	7/1/2018, 6:58 AM	08586711533669045905455308379CU59	472 Milliseconds

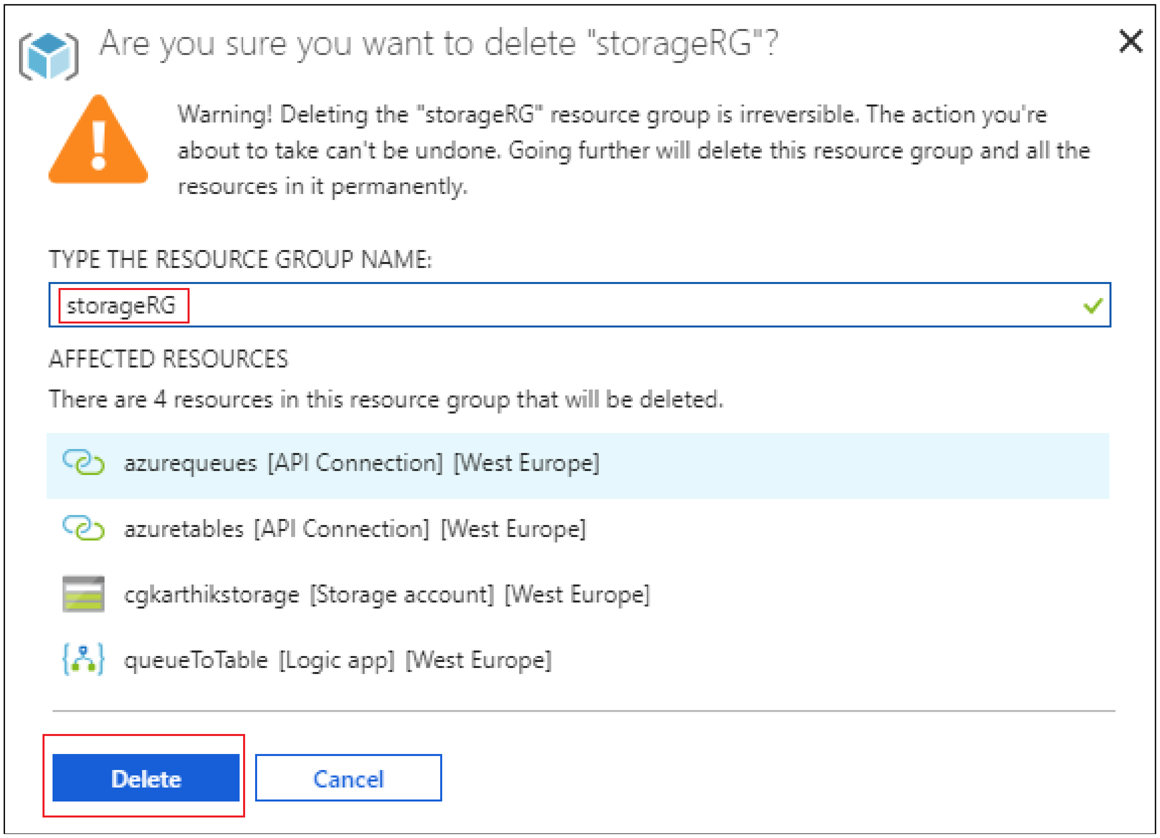
Step 18: As a result, it deletes the message from the queue and add the message in Azure Table Storage



Enter a WCF Data Services filter to limit the entities returned

PartitionKey	RowKey	Timestamp	Message
1060363e-9d30-4a29-ad7c-fa6684e2cc9c	5fbbe17b-8fac-42d9-a35c-93cc814dc16c	7/1/2018 1:58:38 PM	Second
14ce97c1-e790-4010-9e82-d478251284b3	cdb30eed-ec13-4fbb-b5af-7659a9b7ae68	7/1/2018 1:58:38 PM	First

Step 19: Delete the Resource Group



Are you sure you want to delete "storageRG"?

Warning! Deleting the "storageRG" resource group is irreversible. The action you're about to take can't be undone. Going further will delete this resource group and all the resources in it permanently.

TYPE THE RESOURCE GROUP NAME:

storageRG

AFFECTED RESOURCES

There are 4 resources in this resource group that will be deleted.

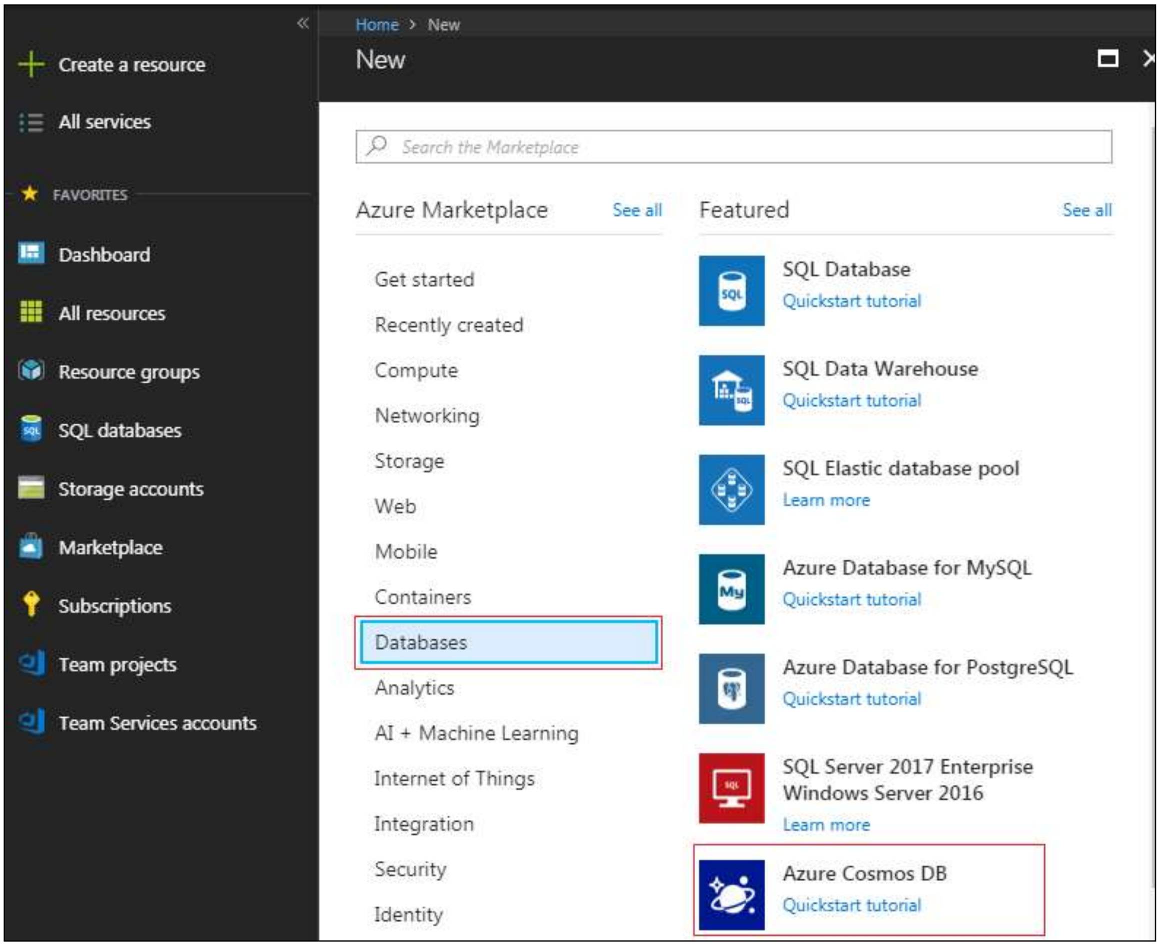
- azurequeues [API Connection] [West Europe]
- azuretables [API Connection] [West Europe]
- cgkarthikstorage [Storage account] [West Europe]
- queueToTable [Logic app] [West Europe]

Lab 7. Build a .NET web app with Azure Cosmos DB

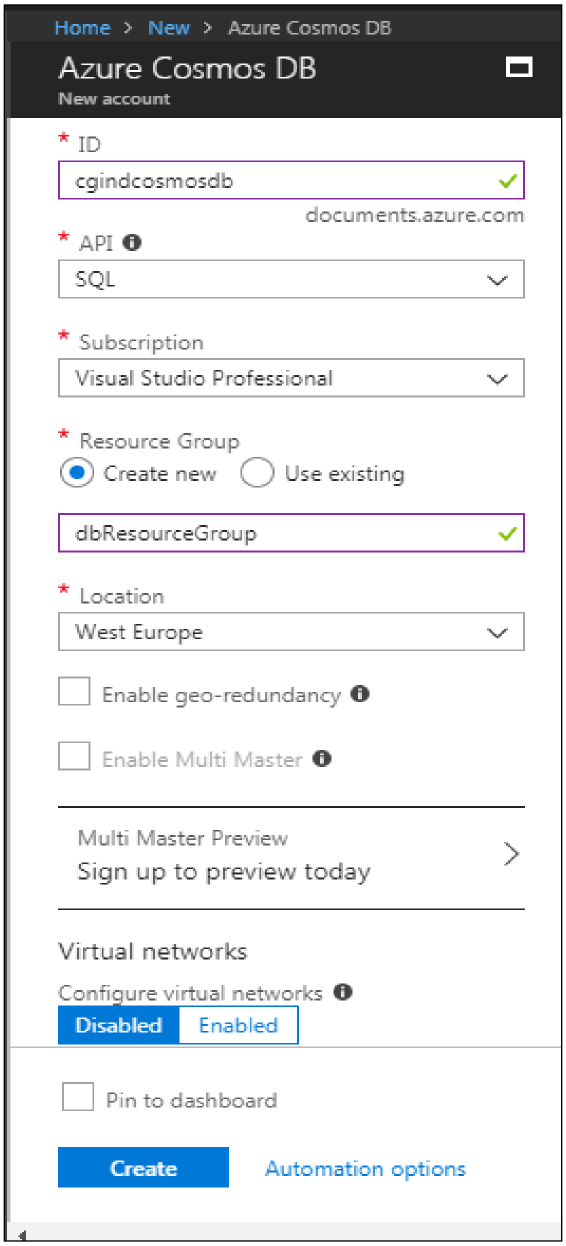
Goals	Create an Azure Cosmos DB SQL API account, document database, and collection using the Azure portal then build and deploy a todo list web app built on the SQL .NET API
Time	60 Mins

Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Click Create a resource > Databases > Azure Cosmos DB.



Step 3: In the New account page, enter the settings for the new Azure Cosmos DB account.



Home > New > Azure Cosmos DB

Azure Cosmos DB

New account

* ID

cgindcosmosdb ✓

documents.azure.com

* API

SQL

* Subscription

Visual Studio Professional

* Resource Group

● Create new ○ Use existing

dbResourceGroup ✓

* Location

West Europe

□ Enable geo-redundancy

□ Enable Multi Master

Multi Master Preview

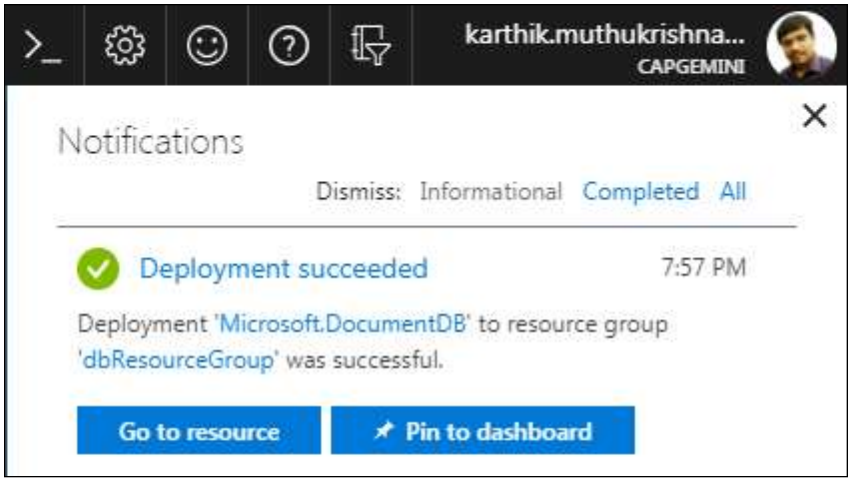
Sign up to preview today

Virtual networks

Configure virtual networks

□ Pin to dashboard

Step 4: Select the Notification icon in the upper-right corner of the portal and watch for the Deployment succeeded message



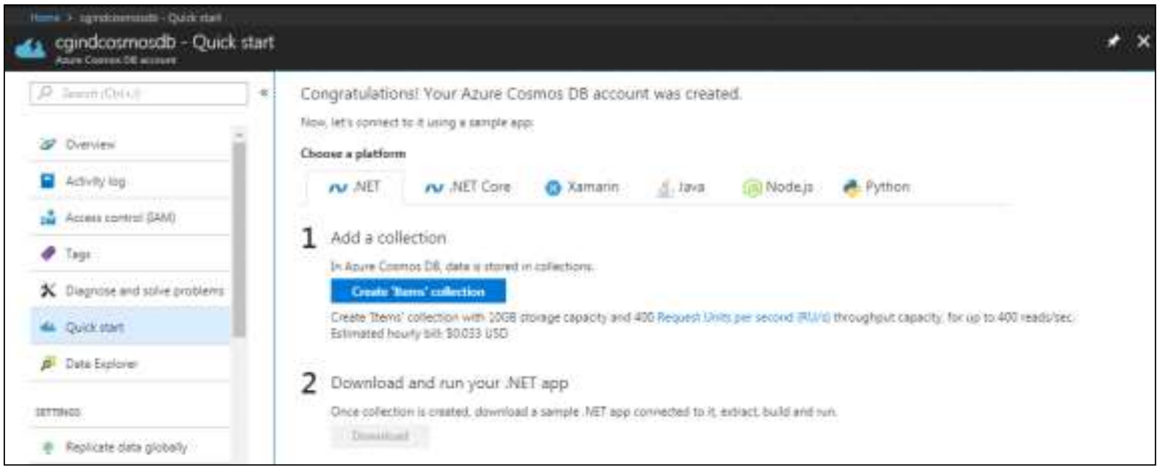
Notifications

Dismiss: Informational Completed All

Deployment succeeded 7:57 PM

Deployment 'Microsoft.DocumentDB' to resource group 'dbResourceGroup' was successful.

Step 5: Click Go to resource and from the overview page click QuickStart which to display Congratulations! Your Azure Cosmos DB account was created page.



Congratulations! Your Azure Cosmos DB account was created.

Now, let's connect to it using a sample app:

Choose a platform

1 Add a collection

In Azure Cosmos DB, data is stored in collections.

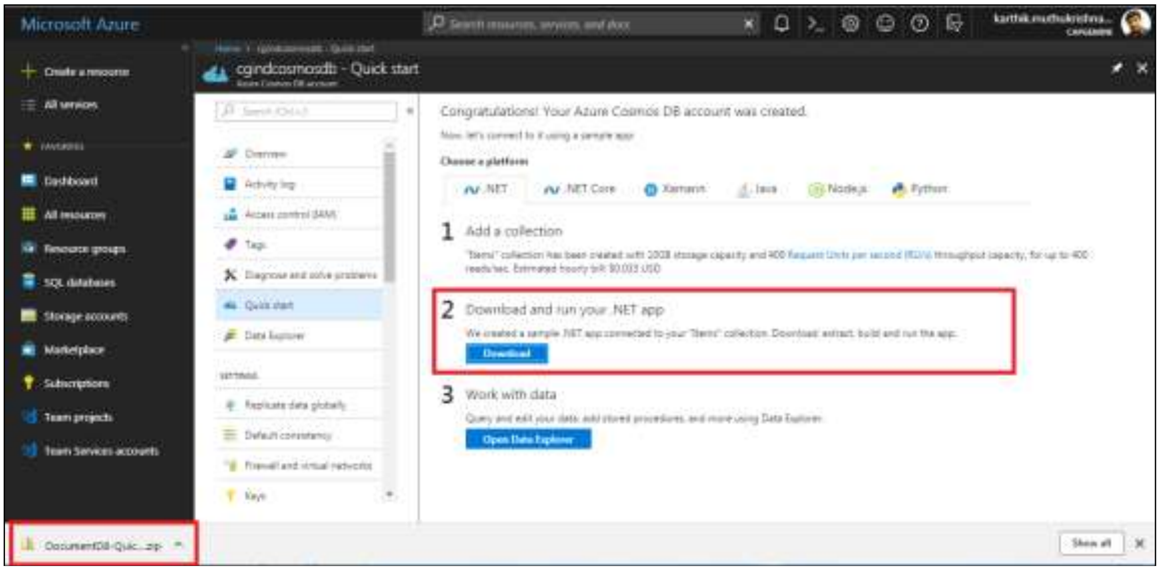
Create 'Items' collection with 10GB storage capacity and 400 Request Units per second (RU/s) throughput capacity, for up to 400 reads/sec.
Estimated hourly bill: $0.033 USD

2 Download and run your .NET app

Once collection is created, download a sample .NET app connected to it, extract, build and run.

Step 6: Click Create 'Items' collection to create a sample database named Tasks with a collection named Items

Step 7: Download the sample .NET app provided by Azure which gets connected to your "Items" collection.



Congratulations! Your Azure Cosmos DB account was created.

Now, let's connect to it using a sample app

Choose a platform

1 Add a collection

"Items" collection has been created with 10GB storage capacity and 400 Request Units per second (RU/s) throughput capacity, for up to 400 reads/sec. Estimated hourly bill: $0.033 USD

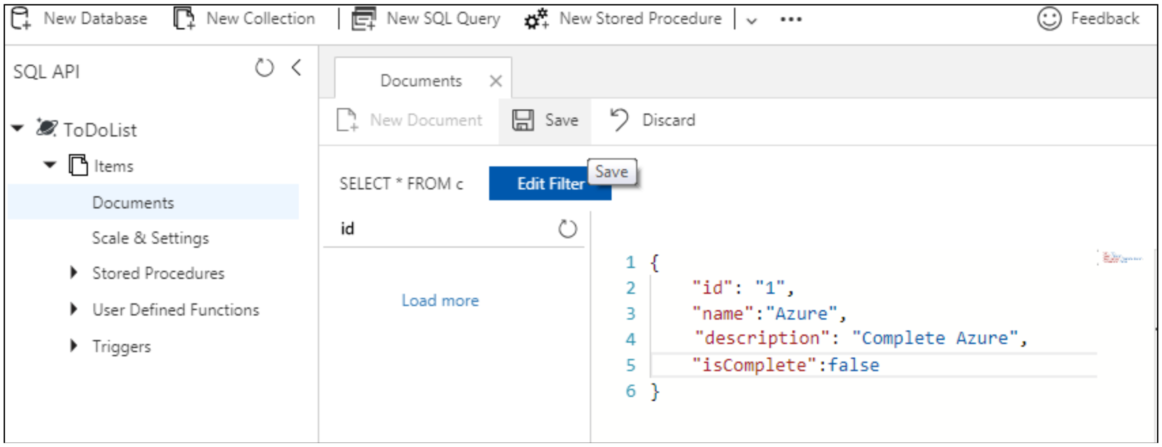
2 Download and run your .NET app

We created a sample .NET app connected to your "Items" collection. Download, extract, build and run the app.

3 Work with data

Query and edit your data, add stored procedures, and more using Data Explorer.

Step 8: Click Open Data Explorer and now add a document to the collection with the following structure and save it.



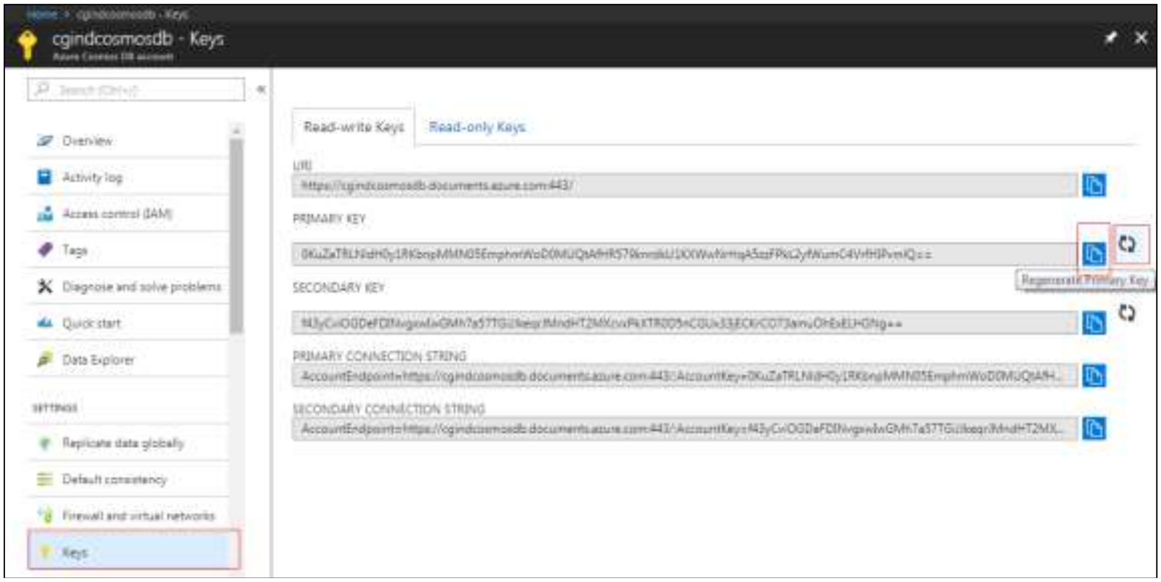
SELECT * FROM c

```
{
    "id": "1",
    "name":"Azure",
    "description": "Complete Azure",
    "isComplete":false
}
```

Step 9: Open the downloaded todo solution file in Visual Studio and build the project to install the nuget packages and compile the project

Step 10: In the Azure portal, in your Azure Cosmos DB account, in the left navigation click Keys, and then click Read-write Keys.

Step 11: Click Regenerate Primary Key and copy the PRIMARY KEY value from the portal and make it the value of the authKey in web.config. You've now updated your app with all the info it needs to communicate with Azure Cosmos DB.



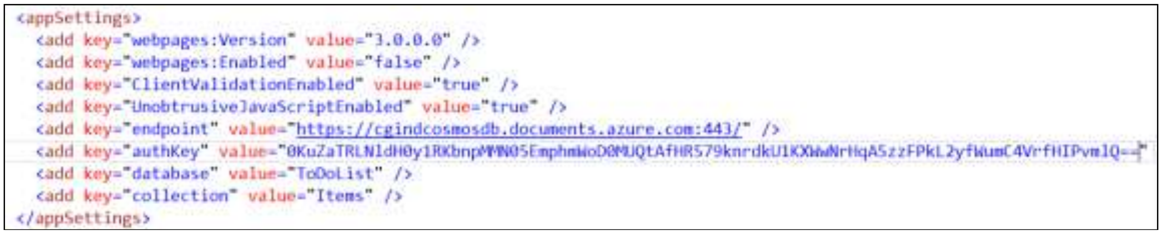
URI
https://cgindcosmosdb.documents.azure.com:443/

PRIMARY KEY
0KuZaTRLNldH0y1RKbnpMMN05EmphmWoD0MUQtAfHR579knrdkU1KXWwNrHqA5zzFPkL2yfWumC4VrfHIPvmlQ==

SECONDARY KEY
f43yCvOGDeFDINvgswlwGMh7a57TGUkeqrMndHT2MXcvPkXTR0D5nC0Uk33ECK/C0T3amuOhExELHGNg==

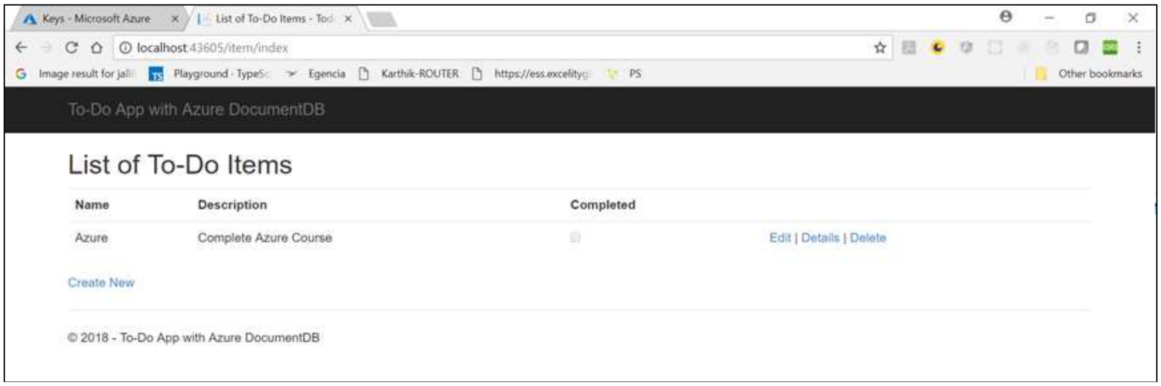
PRIMARY CONNECTION STRING
AccountEndpoint=https://cgindcosmosdb.documents.azure.com:443/;AccountKey=0KuZaTRLNldH0y1RKbnpMMN05EmphmWoD0MUQtAfH...

SECONDARY CONNECTION STRING
AccountEndpoint=https://cgindcosmosdb.documents.azure.com:443/;AccountKey=f43yCvOGDeFDINvgswlwGMh7a57TGUkeqrMndHT2MX...



```xml
<appSettings>
  <add key="webpages:Version" value="3.0.0.0" />
  <add key="webpages:Enabled" value="false" />
  <add key="ClientValidationEnabled" value="true" />
  <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  <add key="endpoint" value="https://cgindcosmosdb.documents.azure.com:443/" />
  <add key="authKey" value="0KuZaTRLNldH0y1RKbnpMMN05EmphmWoD0MUQtAfHR579knrdkU1KXWwNrHqA5zzFPkL2yfWumC4VrfHIPvmlQ=="
  <add key="database" value="ToDoList" />
  <add key="collection" value="Items" />
</appSettings>
```

Step 12: Click CTRL + F5 to run the application. Your app displays in your browser.



To-Do App with Azure DocumentDB

List of To-Do Items

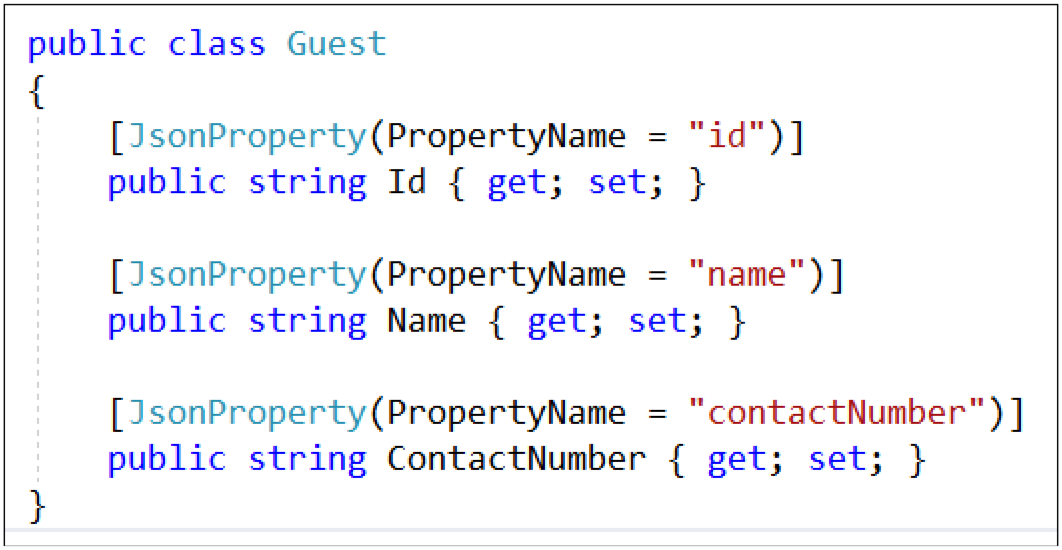
Name	Description	Completed	
Azure	Complete Azure Course	☐	Edit | Details | Delete

Create New

© 2018 - To-Do App with Azure DocumentDB

Stretched Assignment:

Change the Program to work with the entity given below instead of Item



```
public class Guest
{
    [JsonProperty(PropertyName = "id")]
    public string Id { get; set; }

    [JsonProperty(PropertyName = "name")]
    public string Name { get; set; }

    [JsonProperty(PropertyName = "contactNumber")]
    public string ContactNumber { get; set; }
}
```

Lab 8. Azure Management Client using C# Console App

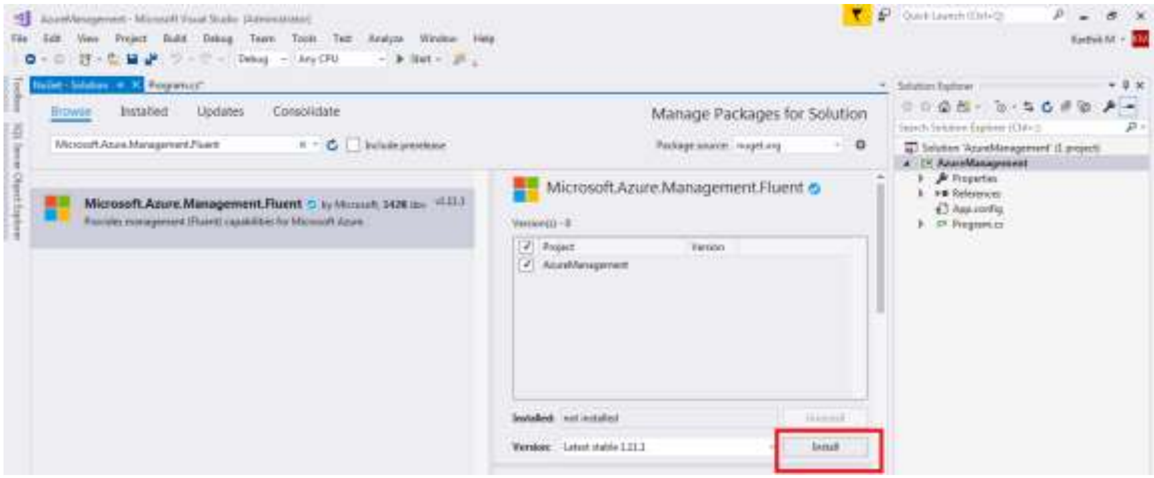
Goals	Create Resource group using Azue SDK through C# Console Application
Time	120 Mins

Step 1: In Visual Studio, click File > New > Project.

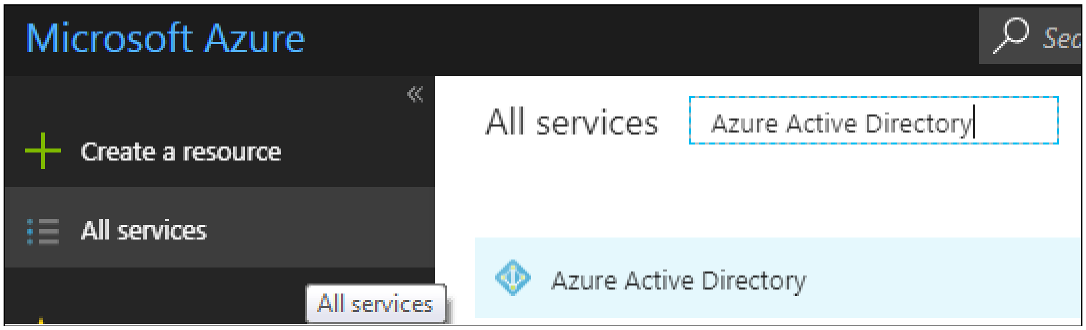
Step 2: In Templates > Visual C#, select Console App (.NET Framework), enter **AzureManagement** for the name of the project, select the location of the project, and then click OK.

Step 3: Click Tools > Nuget Package Manager, and then click Package Manager Console.

Step 4: Click Tools > Nuget Package Manager, and then click Manage Nuget Packages for Solution Browse and install the nuget package **Microsoft.Azure.Management.Fluent**


AzureManagement - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug Any CPU Start
NuGet - Solution Program.cs*
Browse Installed Updates Consolidate
Manage Packages for Solution
Microsoft.Azure.Management.Fluent
Include prerelease
Package source: nuget.org
Microsoft.Azure.Management.Fluent by Microsoft, 142K downloads v1.21.1
Provides management (Fluent) capabilities for Microsoft Azure
Microsoft.Azure.Management.Fluent
Version(s) - 0
Project Version
AzureManagement
Installed: not installed
Uninstall
Version: Latest stable 1.21.1
Install
Solution Explorer
Solution 'AzureManagement' (1 project)
AzureManagement
Properties
References
App.config
Program.cs

Step 5: Open Azure portal and search for Azure Active Directory

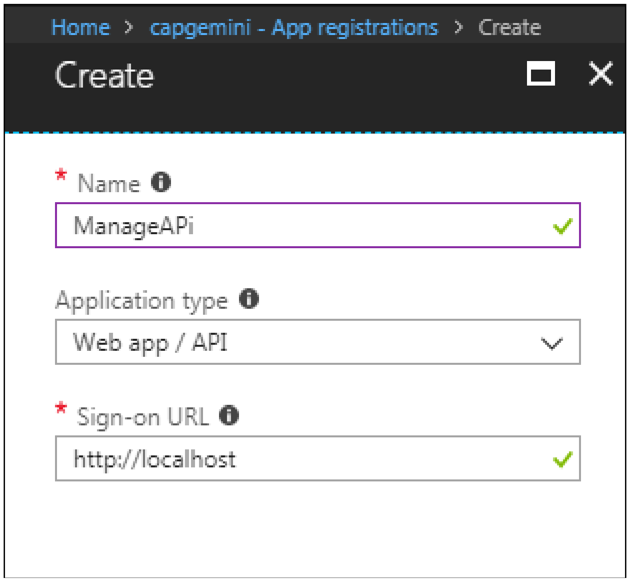


All services: Azure Active Directory

Azure Active Directory

Step 6: Click App registrations, Click the option: New application registration

Step 7: Provide a name and URL for the application. Select Web app / API for the type of application you want to create. You cannot create credentials for a Native application; therefore, that type does not work for an automated application. After setting the values, select Create



Home > capgemini - App registrations > Create

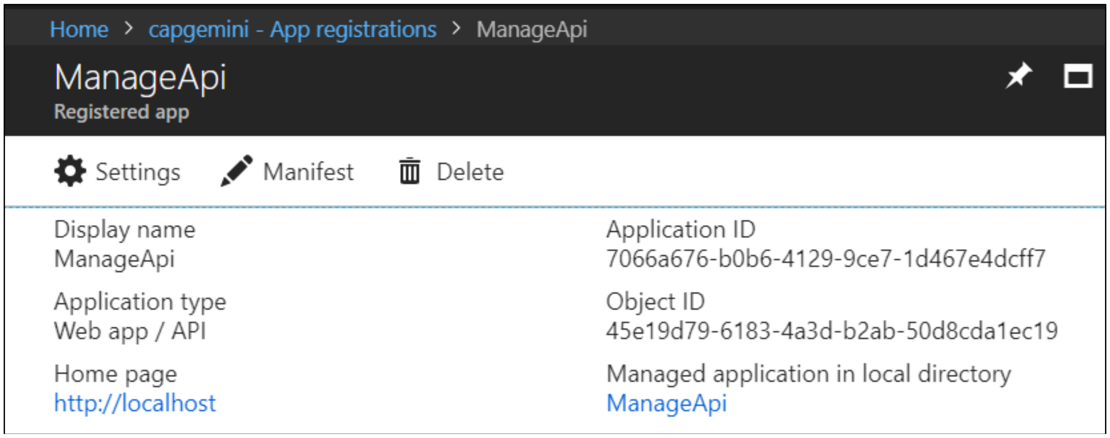
Create

* Name: ManageAPi

Application type: Web app / API

* Sign-on URL: http://localhost

Step 8: App gets registered



ManageApi
Registered app

Display name
ManageApi

Application type
Web app / API

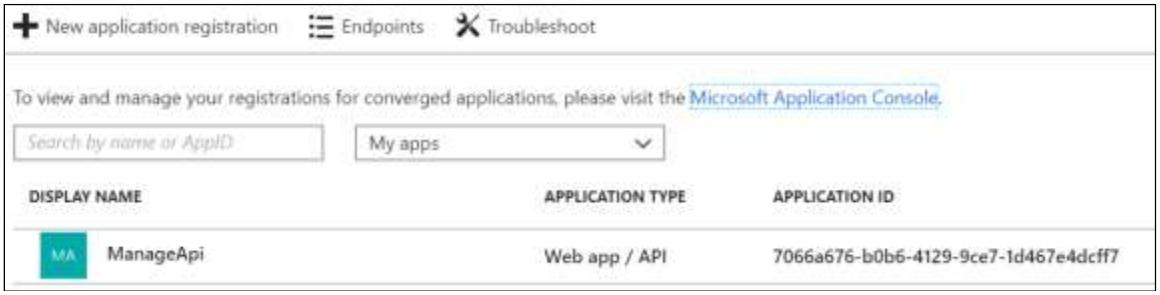
Home page
http://localhost

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Object ID
45e19d79-6183-4a3d-b2ab-50d8cda1ec19

Managed application in local directory
ManageApi

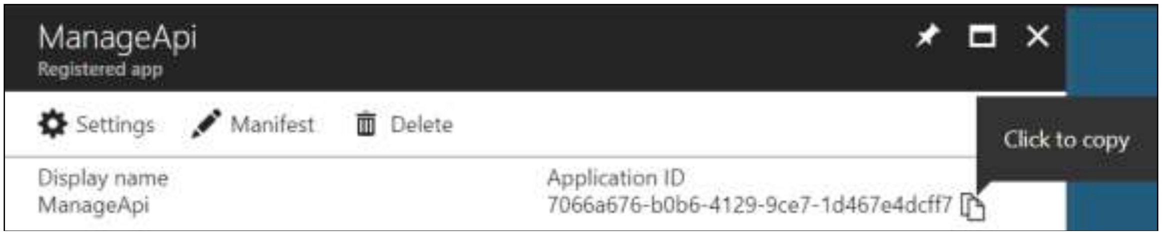
Step 9: When programmatically logging in, you need the ID for your application and an authentication key. To get those values from App registrations in Azure Active Directory, select your application.



To view and manage your registrations for converged applications, please visit the Microsoft Application Console.

DISPLAY NAME	APPLICATION TYPE	APPLICATION ID
ManageApi	Web app / API	7066a676-b0b6-4129-9ce7-1d467e4dcff7

Step 10: Copy the Application ID

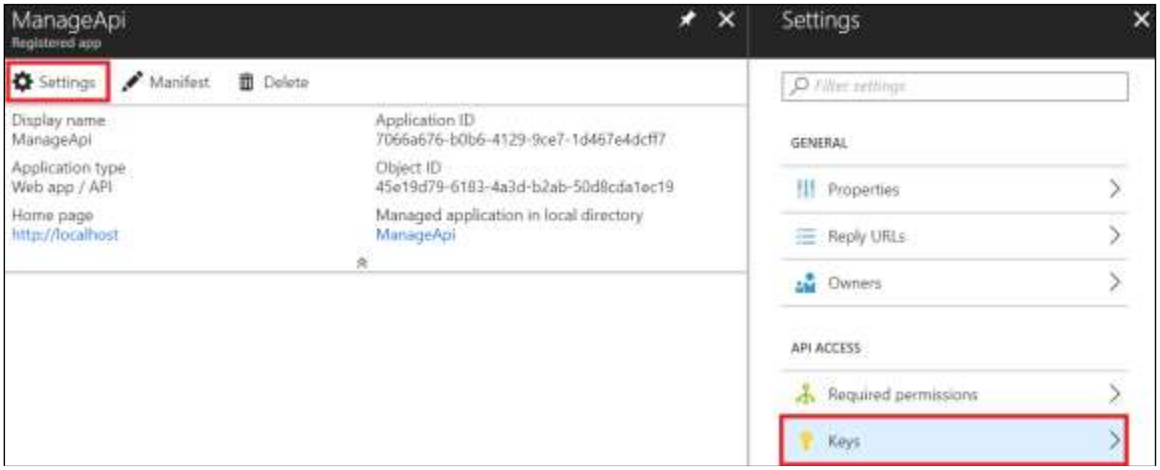


ManageApi
Registered app

Display name
ManageApi

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Step 11: To generate an authentication key, select Setting and then to generate an authentication key, select Keys.



Display name
ManageApi

Application type
Web app / API

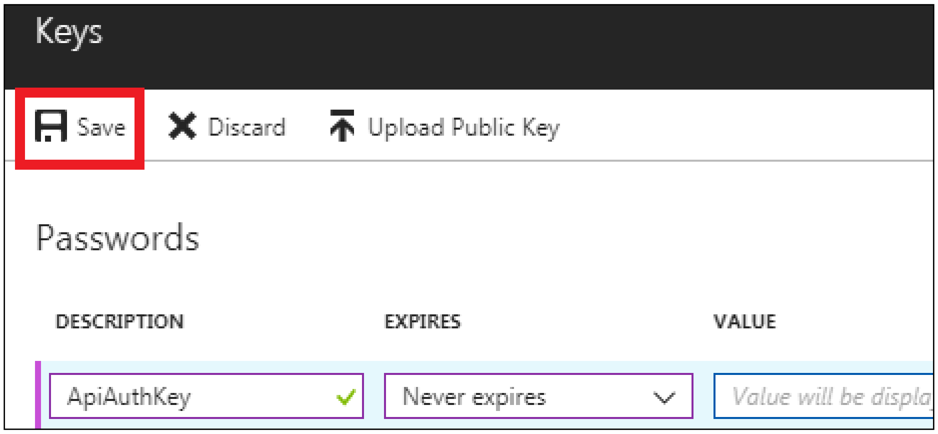
Home page
http://localhost

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Object ID
45e19d79-6183-4a3d-b2ab-50d8cda1ec19

Managed application in local directory
ManageApi

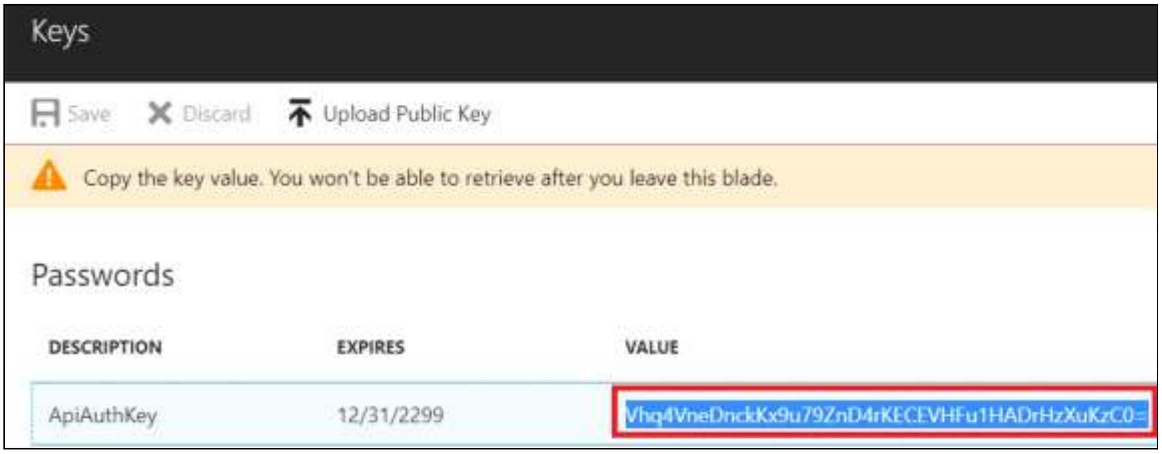
Step 12: Provide a description of the key, and a duration for the key. When done, select Save.



DESCRIPTION	EXPIRES	VALUE
ApiAuthKey	Never expires	Value will be displa

After saving the key, the value of the key is displayed. Copy this value because you are not able to retrieve the key later. You provide the key value with the application ID to log in as the application. Store the key value where your application can retrieve it.

Step 13: Copy the value for future use



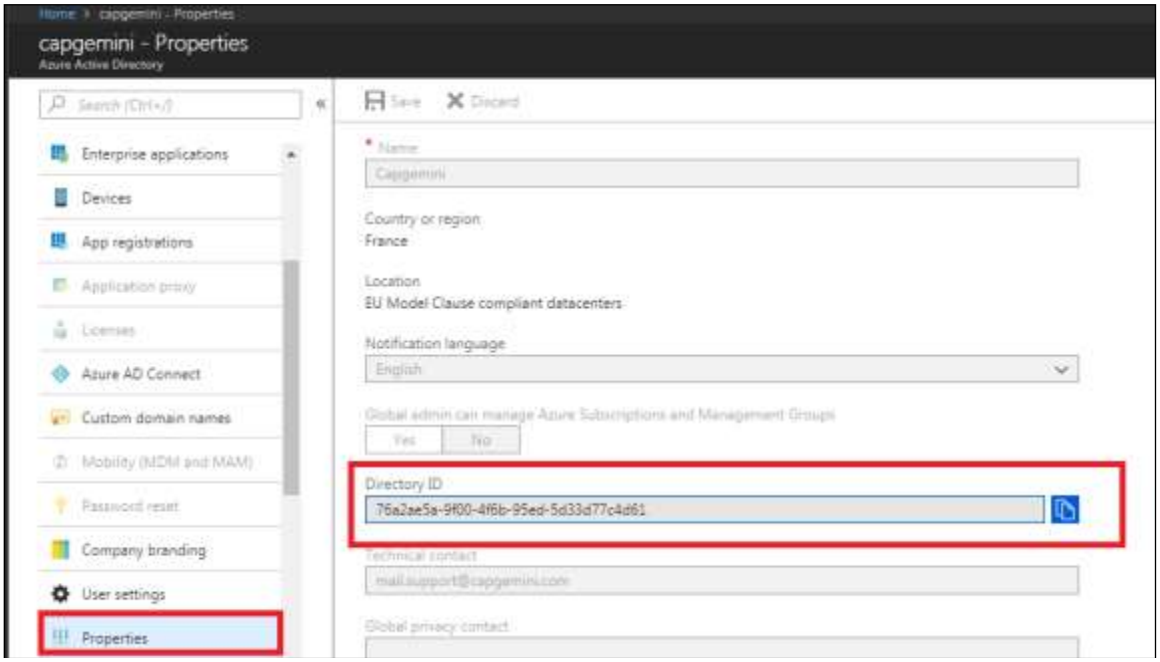
Keys

Copy the key value. You won't be able to retrieve after you leave this blade.

Passwords

DESCRIPTION	EXPIRES	VALUE
ApiAuthKey	12/31/2299	Vhq4VneDnckKx9u79ZnD4rKECEVHFu1HADrHzXuKzC0=

Step 14: To get the Tenant Id select Azure Active Directory > Properties for your Azure AD tenant. Copy the Directory ID. This value is your tenant ID



* Name
Capgemini

Country or region
France

Location
EU Model Clause compliant datacenters

Notification language
English

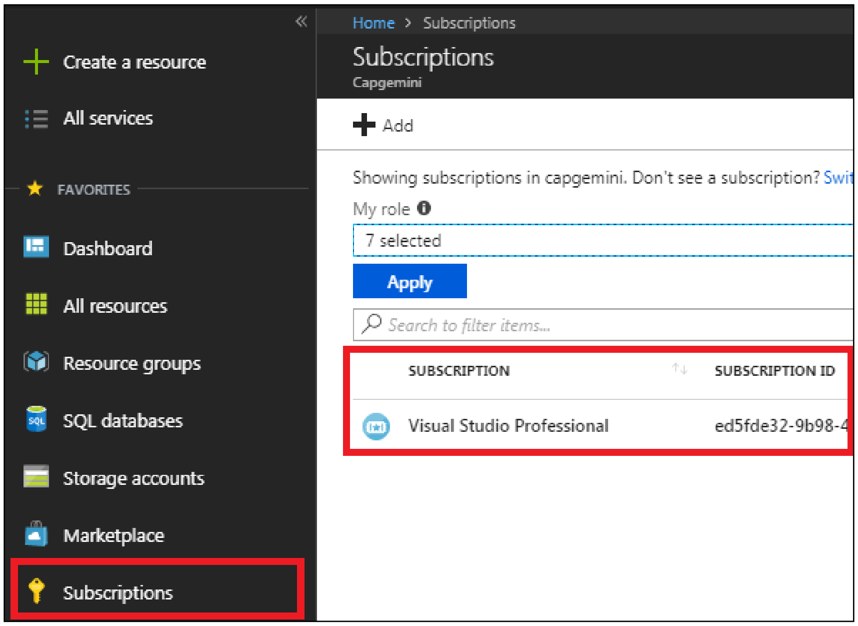
Global admin can manage Azure Subscriptions and Management Groups
Yes No

Directory ID
76a2ae5a-9f00-4f6b-95ed-5d33d77c4d61

Technical contact
mail.support@capgemini.com

Global privacy contact

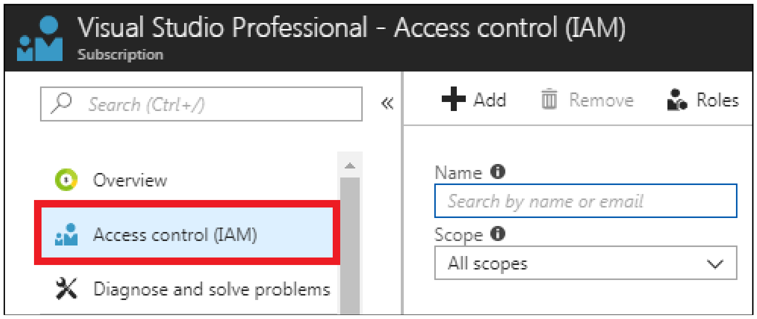
Step 15: To Assign application to a role. Navigate to the level of scope you wish to assign the application to. For example, to assign a role at the subscription scope, select Subscriptions(Go to All Services and search for Subscription). You could instead select a resource group or resource.



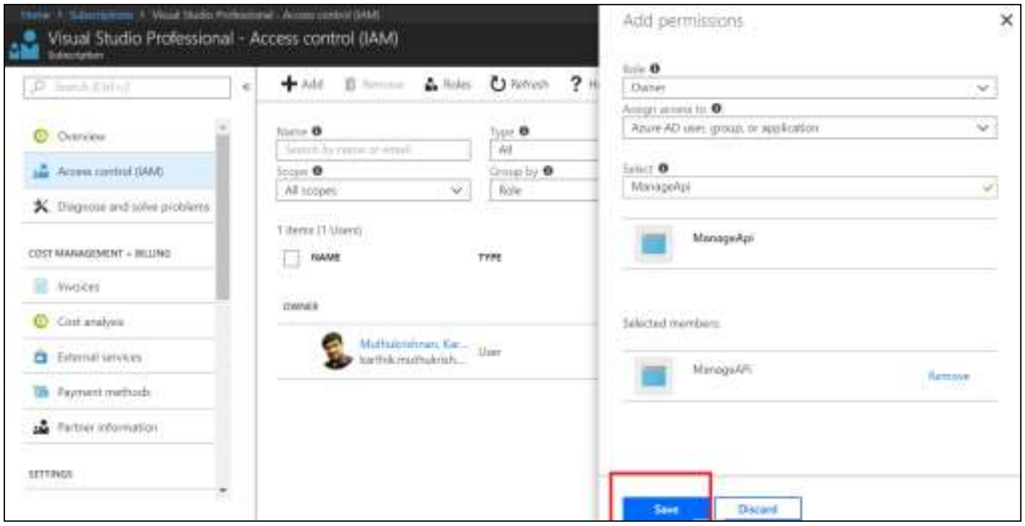
Showing subscriptions in capgemini. Don't see a subscription? Swi

SUBSCRIPTION	SUBSCRIPTION ID
Visual Studio Professional	ed5fde32-9b98-4

Step 16: Select the particular subscription (resource group or resource) to assign the application to and select Access Control (IAM).



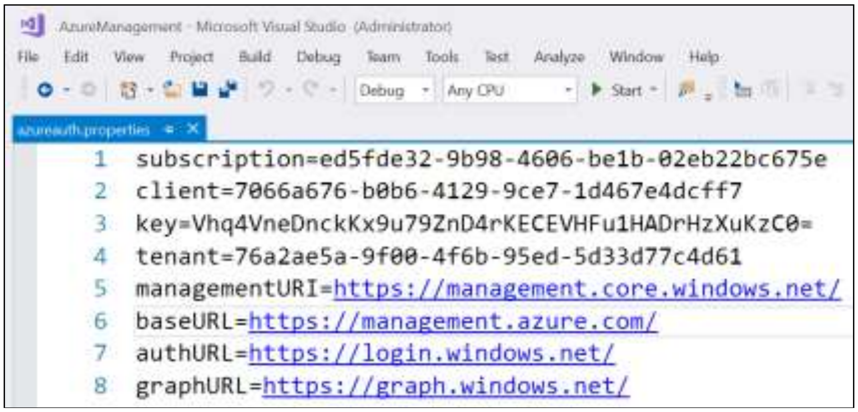
Step 17: Select Add > Select the role you wish to assign to the application. The following image shows the Reader role. By default, Azure Active Directory applications aren't displayed in the available options. To find your application, you must provide the name of it in the search field. Select it.



Role: Owner
Assign access to: Azure AD user, group, or application
Select: ManageApi
ManageApi
Selected members:
ManageAPI

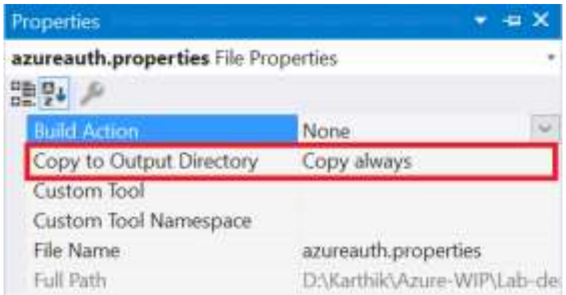
Step 18: Select Save to finish assigning the role. You see your application in the list of users assigned to a role for that scope.

Step 19: In Solution Explorer, right-click myDotnetProject > Add > New Item, and then select Text File in Visual C# Items. Name the file azureauth.properties, and then click Add. Add these authorization properties:


AzureManagement - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug
Any CPU
Start
azureauth.properties

```
subscription=ed5fde32-9b98-4606-be1b-02eb22bc675e
client=7066a676-b0b6-4129-9ce7-1d467e4dcff7
key=Vhq4VneDnckKx9u79ZnD4rKECEVHFu1HADrHzXuKzC0=
tenant=76a2ae5a-9f00-4f6b-95ed-5d33d77c4d61
managementURI=https://management.core.windows.net/
baseURL=https://management.azure.com/
authURL=https://login.windows.net/
graphURL=https://graph.windows.net/
```

Step 20: Right click the azureauth.properties and change the Copy to Output Directory to **Copy always**

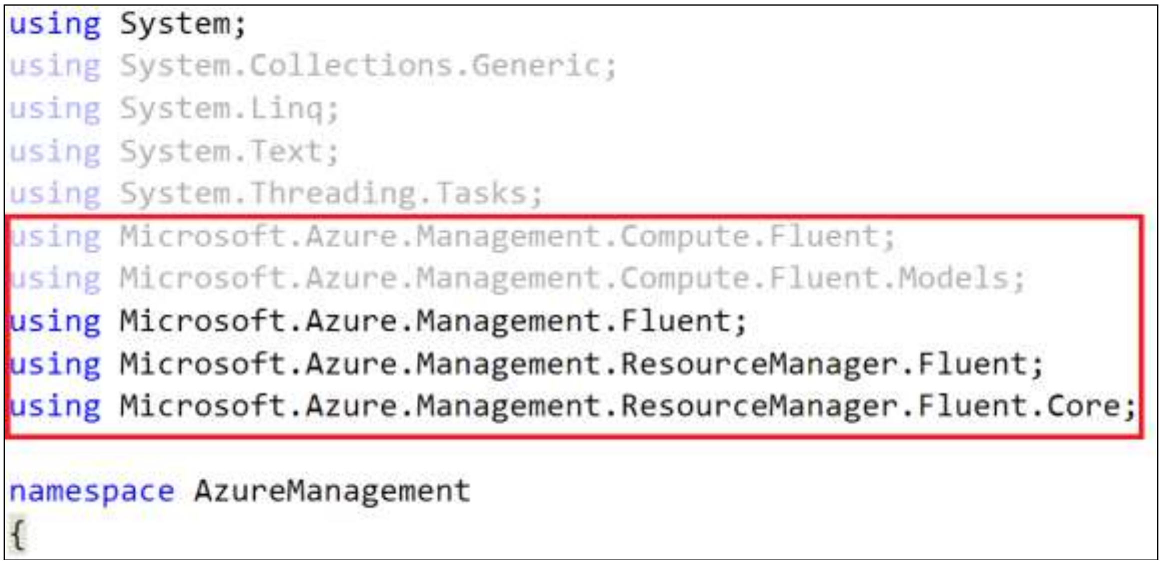


azureauth.properties File Properties

Build Action	None
Copy to Output Directory	Copy always
Custom Tool	
Custom Tool Namespace	
File Name	azureauth.properties
Full Path	D:\Karthik\Azure-WIP\Lab-de

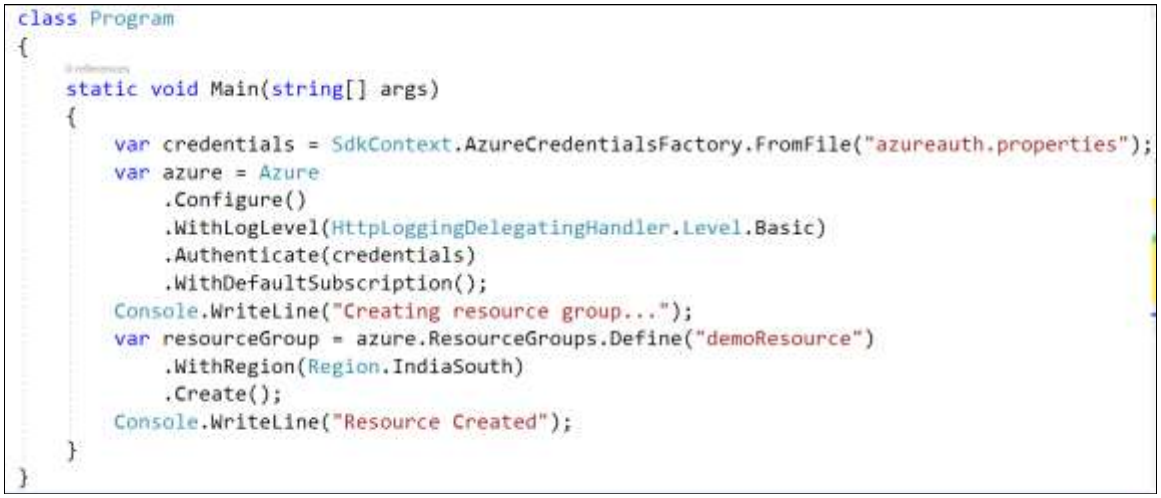
Step 21: Replace <subscription-id> with your subscription identifier, <application-id> with the Active Directory application identifier, <authentication-key> with the application key, and <tenant-id> with the tenant identifier. Save the azureauth.properties file.

Step 22: Open the Program.cs file for the project that you created, and then add these using statements to the existing statements at top of the file:



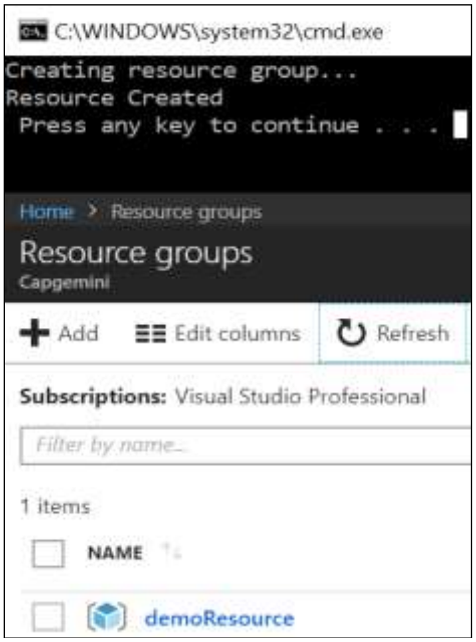
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.Azure.Management.Compute.Fluent;
using Microsoft.Azure.Management.Compute.Fluent.Models;
using Microsoft.Azure.Management.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent.Core;

namespace AzureManagement
{
```

Step 22: To create the management client and adding a Resource, add this code to the Main method:



```
class Program
{
    static void Main(string[] args)
    {
        var credentials = SdkContext.AzureCredentialsFactory.FromFile("azureauth.properties");
        var azure = Azure
            .Configure()
            .WithLogLevel(HttpLoggingDelegatingHandler.Level.Basic)
            .Authenticate(credentials)
            .WithDefaultSubscription();
        Console.WriteLine("Creating resource group...");
        var resourceGroup = azure.ResourceGroups.Define("demoResource")
            .WithRegion(Region.IndiaSouth)
            .Create();
        Console.WriteLine("Resource Created");
    }
}
```

Step 23: Build and Run the Program. It will create Resource group in Azure portal



```
Creating resource group...
Resource Created
 Press any key to continue . . .
```

Lab 9. Use RBAC to manage access with the REST API

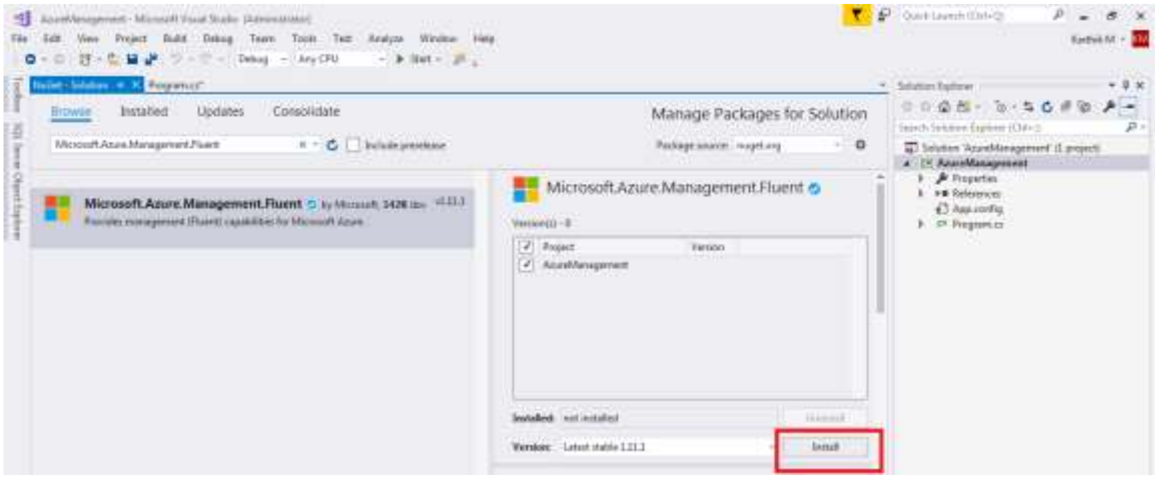
Goals	Create Resource group using Azue SDK through C# Console Application
Time	120 Mins

Step 1: In Visual Studio, click File > New > Project.

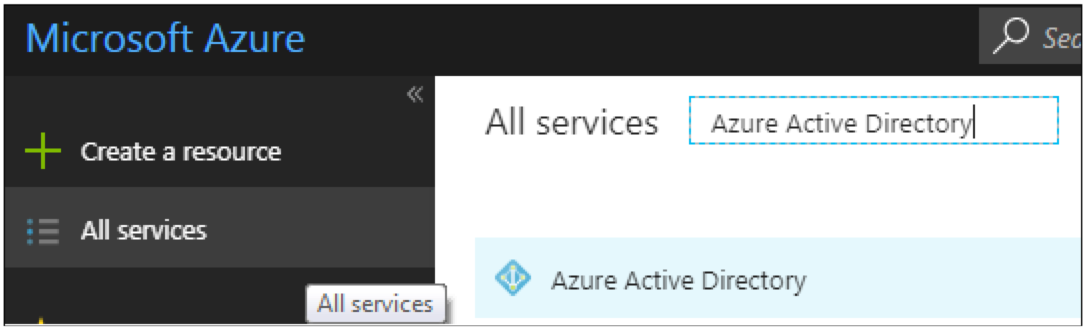
Step 2: In Templates > Visual C#, select Console App (.NET Framework), enter **AzureManagement** for the name of the project, select the location of the project, and then click OK.

Step 3: Click Tools > Nuget Package Manager, and then click Package Manager Console.

Step 4: Click Tools > Nuget Package Manager, and then click Manage Nuget Packages for Solution Browse and install the nuget package **Microsoft.Azure.Management.Fluent**

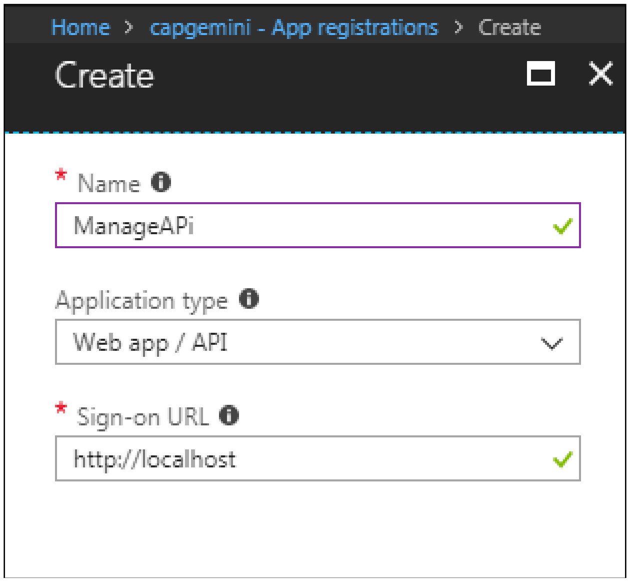

File
Edit
View
Project
Build
Debug
Team
Tools
Test
Analyze
Window
Help
Debug
Any CPU
Start
Browse
Installed
Updates
Consolidate
Manage Packages for Solution
Microsoft.Azure.Management.Fluent
Include prerelease
Package source: nuget.org
Microsoft.Azure.Management.Fluent by Microsoft
Provides management (Fluent) capabilities for Microsoft Azure
Microsoft.Azure.Management.Fluent
Project
Version
AzureManagement
Installed: not installed
Uninstall
Version: Latest stable 1.21.1
Install
Solution Explorer
Solution 'AzureManagement' (1 project)
AzureManagement
Properties
References
App.config
Program.cs

Step 5: Open Azure portal and search for Azure Active Directory



Step 6: Click App registrations

Step 7: Provide a name and URL for the application. Select Web app / API for the type of application you want to create. You cannot create credentials for a Native application; therefore, that type does not work for an automated application. After setting the values, select Create


Home > capgemini - App registrations > Create

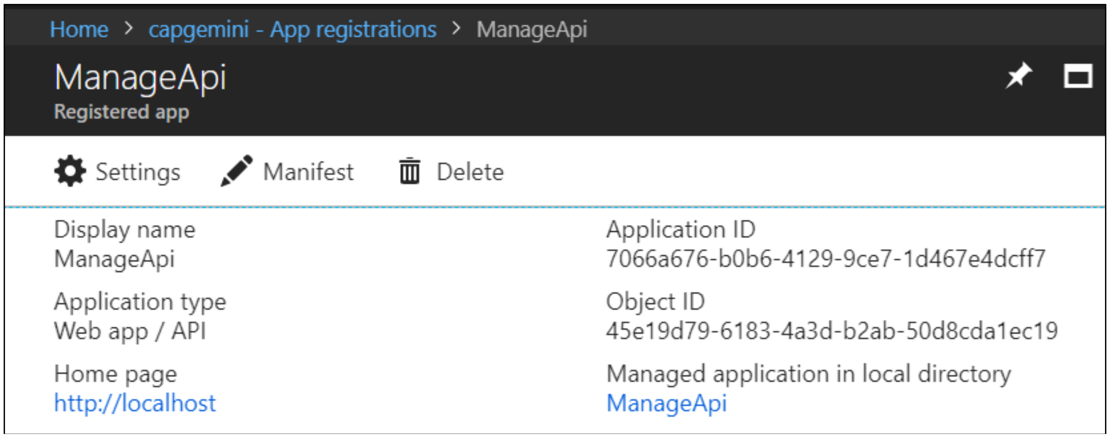
Create

* Name: ManageAPi

Application type: Web app / API

* Sign-on URL: http://localhost

Step 8: App gets registered



ManageApi
Registered app

Display name
ManageApi

Application type
Web app / API

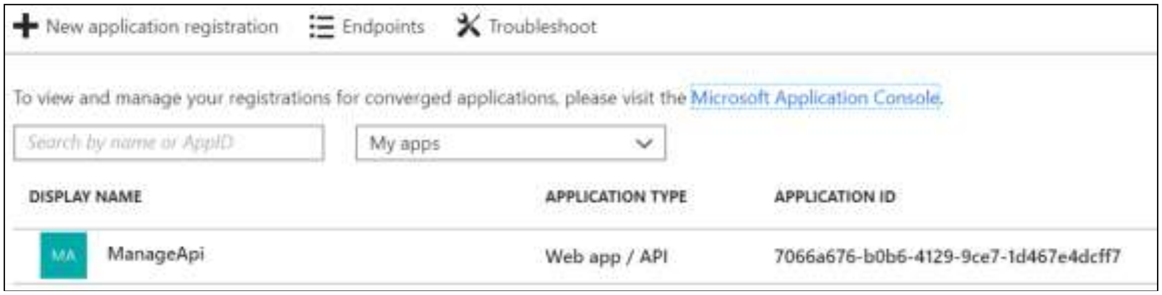
Home page
http://localhost

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Object ID
45e19d79-6183-4a3d-b2ab-50d8cda1ec19

Managed application in local directory
ManageApi

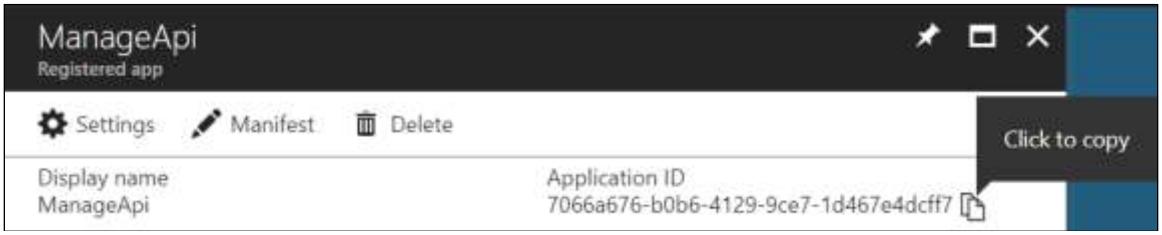
Step 9: When programmatically logging in, you need the ID for your application and an authentication key. To get those values from App registrations in Azure Active Directory, select your application.



To view and manage your registrations for converged applications, please visit the Microsoft Application Console.

DISPLAY NAME	APPLICATION TYPE	APPLICATION ID
ManageApi	Web app / API	7066a676-b0b6-4129-9ce7-1d467e4dcff7

Step 10: Copy the Application ID

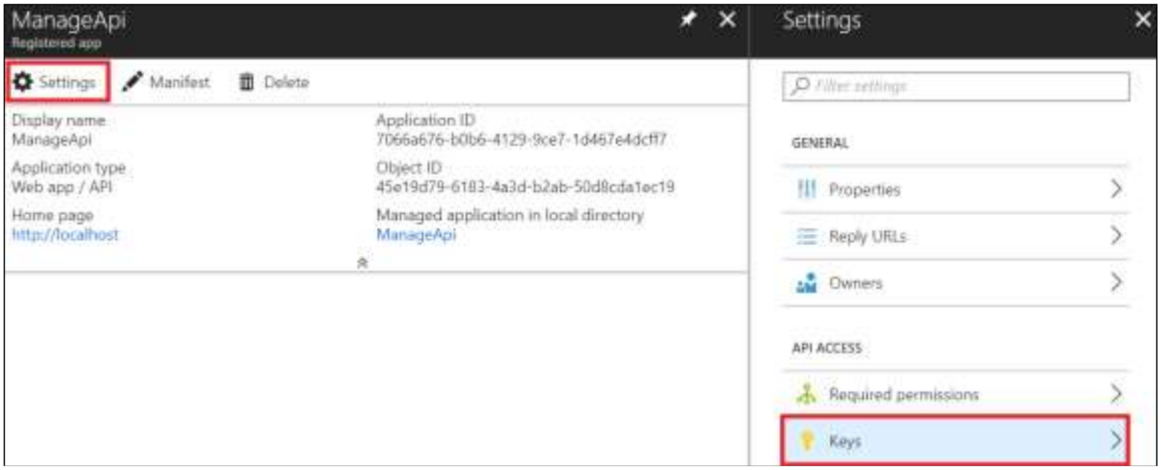


ManageApi
Registered app

Display name
ManageApi

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Step 11: To generate an authentication key, select Setting and then to generate an authentication key, select Keys.



Display name
ManageApi

Application type
Web app / API

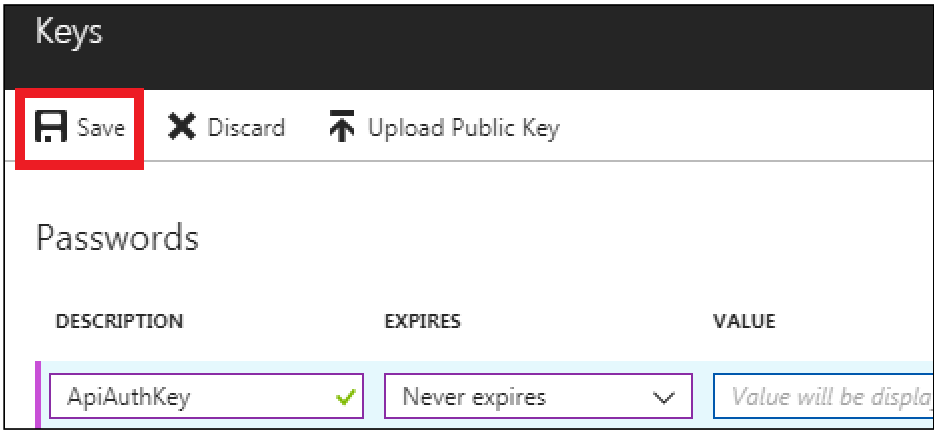
Home page
http://localhost

Application ID
7066a676-b0b6-4129-9ce7-1d467e4dcff7

Object ID
45e19d79-6183-4a3d-b2ab-50d8cda1ec19

Managed application in local directory
ManageApi

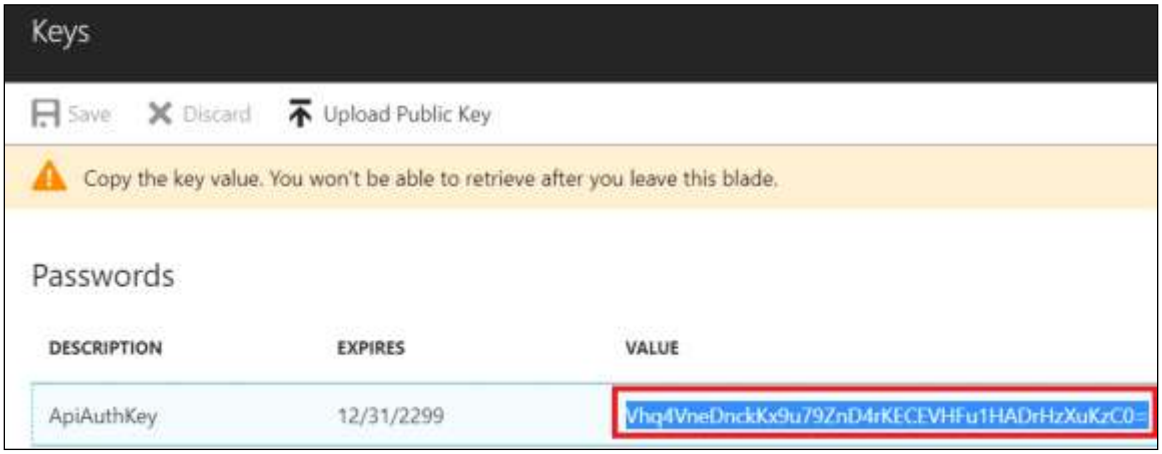
Step 12: Provide a description of the key, and a duration for the key. When done, select Save.



DESCRIPTION	EXPIRES	VALUE
ApiAuthKey	Never expires	Value will be displa

After saving the key, the value of the key is displayed. Copy this value because you are not able to retrieve the key later. You provide the key value with the application ID to log in as the application. Store the key value where your application can retrieve it.

Step 13: Copy the value for future use

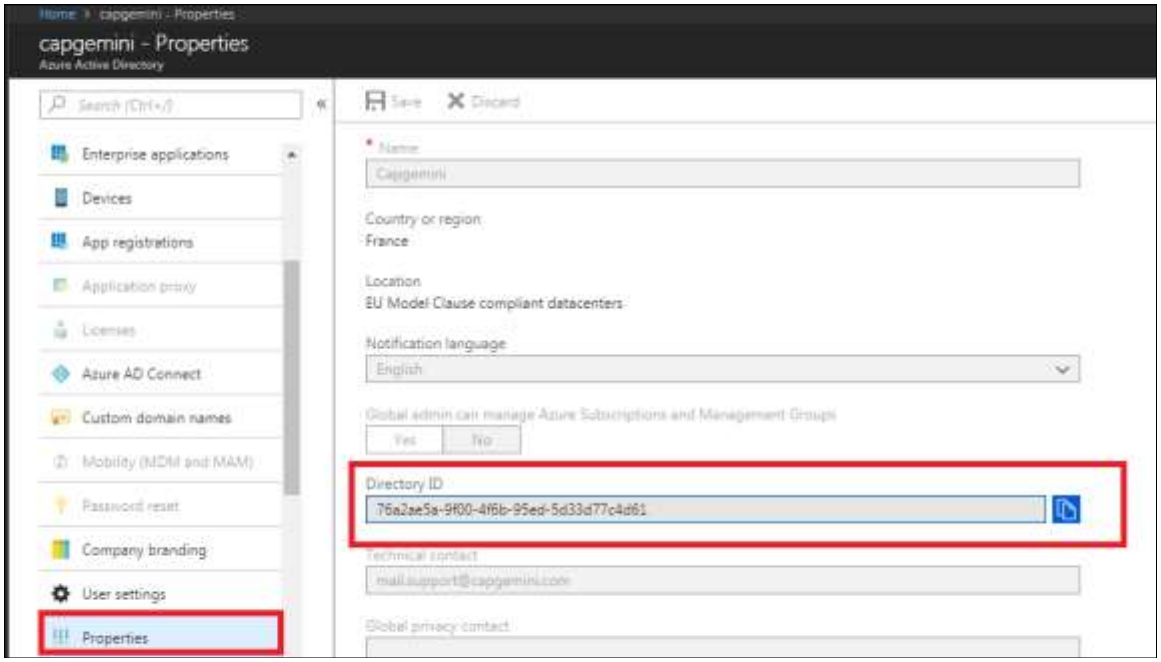


Copy the key value. You won't be able to retrieve after you leave this blade.

Passwords

DESCRIPTION	EXPIRES	VALUE
ApiAuthKey	12/31/2299	Vhq4VneDnckKx9u79ZnD4rKECEVHFu1HADrHzXuKzC0=

Step 14: To get the Tenant Id select Azure Active Directory > Properties for your Azure AD tenant. Copy the Directory ID. This value is your tenant ID



* Name: Capgemini

Country or region: France

Location: EU Model Clause compliant datacenters

Notification language: English

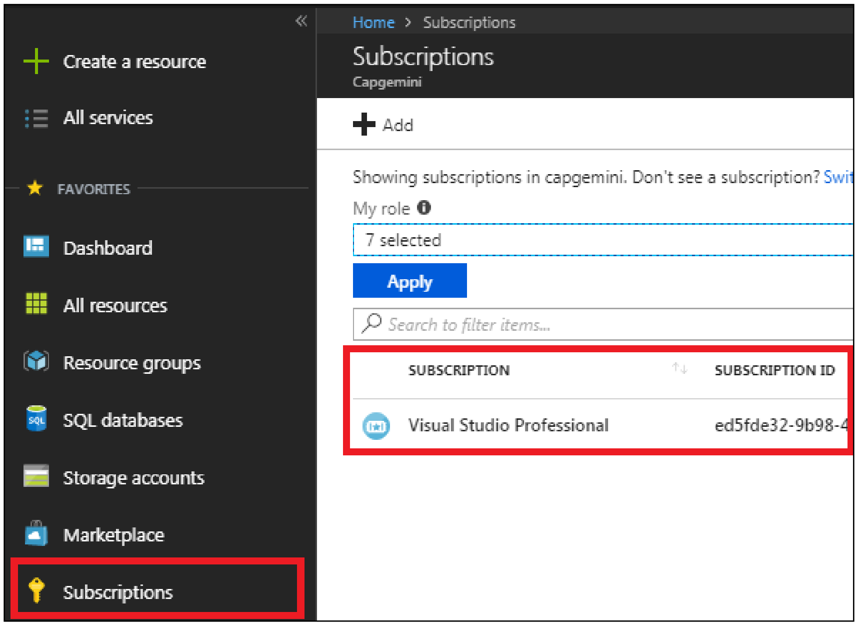
Global admin can manage Azure Subscriptions and Management Groups: Yes [✓ No]

Directory ID: 76a2ae5a-9f00-4f6b-95ed-5d33d77c4d61

Technical contact: mail.support@capgemini.com

Global privacy contact

Step 15: To Assign application to a role. Navigate to the level of scope you wish to assign the application to. For example, to assign a role at the subscription scope, select Subscriptions. You could instead select a resource group or resource.

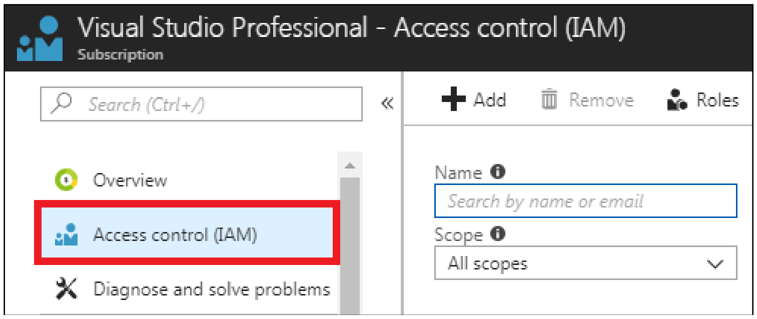


Showing subscriptions in capgemini. Don't see a subscription? Swit

My role: 7 selected

SUBSCRIPTION	SUBSCRIPTION ID
Visual Studio Professional	ed5fde32-9b98-4

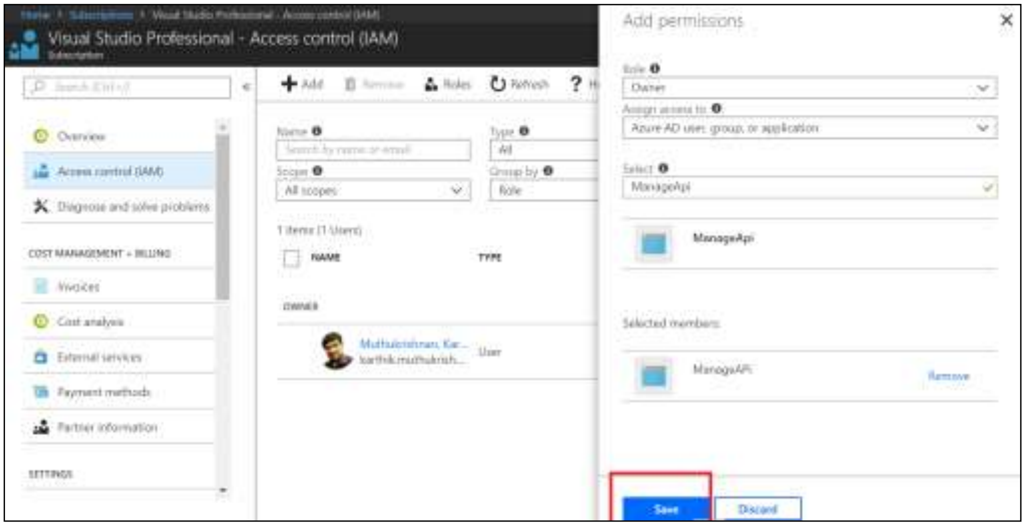
Step 16: Select the particular subscription (resource group or resource) to assign the application to and select Access Control (IAM).


Visual Studio Professional - Access control (IAM)
Subscription
Search (Ctrl+/)
«
Overview
Access control (IAM)
Diagnose and solve problems
Add
Remove
Roles

Name: Search by name or email

Scope: All scopes

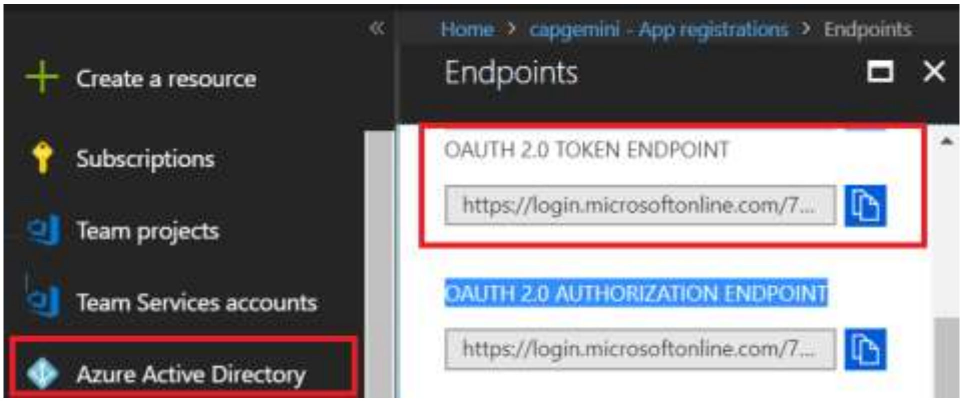
Step 17: Select Add > Select the role you wish to assign to the application. The following image shows the Reader role. By default, Azure Active Directory applications aren't displayed in the available options. To find your application, you must provide the name of it in the search field. Select it.



Step 18: Select Save to finish assigning the role. You see your application in the list of users assigned to a role for that scope.

Step 19: You can redeem the code for an access token to the desired resource, by sending a POST request to the /token endpoint

You can get the OAuth 2.0 authorization endpoint for your tenant by selecting Azure Active Directory > App registrations > Endpoints in the Azure portal. Copy the OAUTH 2.0 TOKEN ENDPOINT



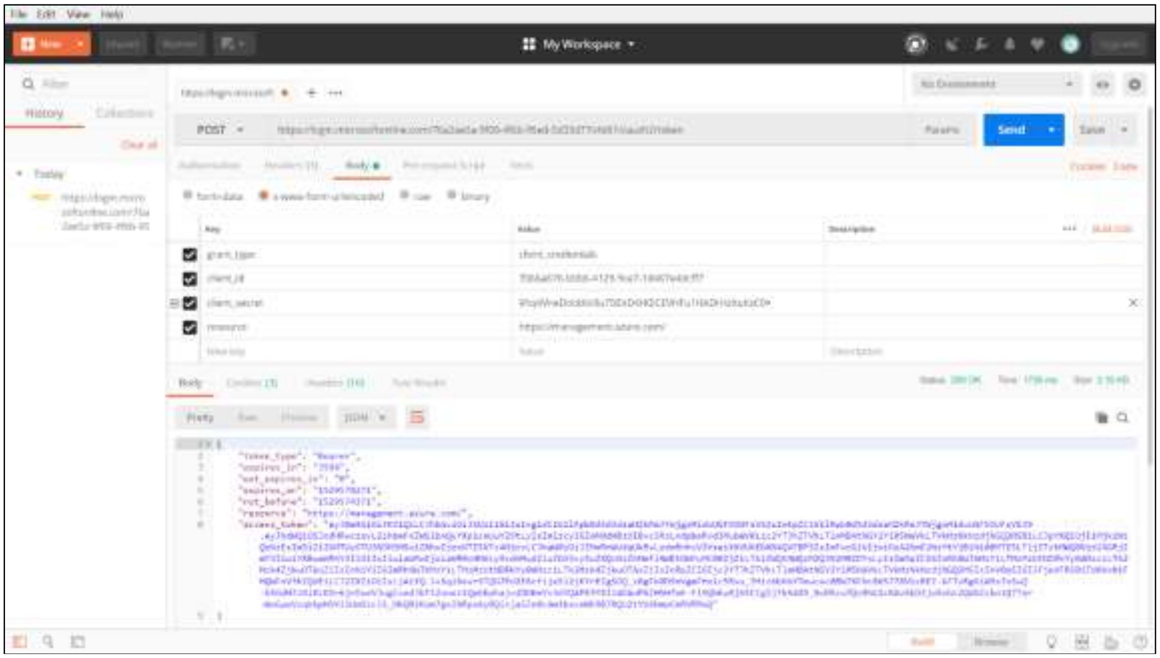
Create a resource
Subscriptions
Team projects
Team Services accounts
Azure Active Directory
Home > capgemini - App registrations > Endpoints
Endpoints
OAUTH 2.0 TOKEN ENDPOINT
https://login.microsoftonline.com/7...
OAUTH 2.0 AUTHORIZATION ENDPOINT
https://login.microsoftonline.com/7...

Step 22: Install Postman for Windows or install Postman chrome plugin from web store

Step 23: Send a POST request through Postman to get the Bearer access_token Token

POST:
https://login.microsoftonline.com/76a2ae5a-9f00-4f6b-95ed5d33d77c4d61/oauth2/token
Content-Type: application/x-www-form-urlencoded

Request Body:
grant_type=client_credentials
client_id=7066a676-b0b6-4129-9ce7-1d467e4dcff7&
client_secret=Vhq4VneDnckKx9u79ZnD4rKECEVHFu1HADrHzXuKzC0=
resource=https://management.azure.com/



Key	Value
☑ grant_type	client_credentials
☑ client_id	7066a676-b0b6-4129-9ce7-1d467e4dcff7
☑ client_secret	Vhq4VneDnckKx9u79ZnD4rKECEVHFu1HADrHzXuKzC0=
☑ resource	https://management.azure.com/

```
{
    "token_type": "Bearer",
    "expires_in": "3599",
    "ext_expires_in": "0",
    "expires_on": "[illegible]",
    "not_before": "[illegible]",
    "resource": "https://management.azure.com/",
    "access_token": "[illegible]"
}
```

Step 24: Azure AD returns an access token upon a successful response. To minimize network calls from the client application and their associated latency, the client application should cache access tokens for the token lifetime that is specified in the OAuth 2.0 response.

Response Body:

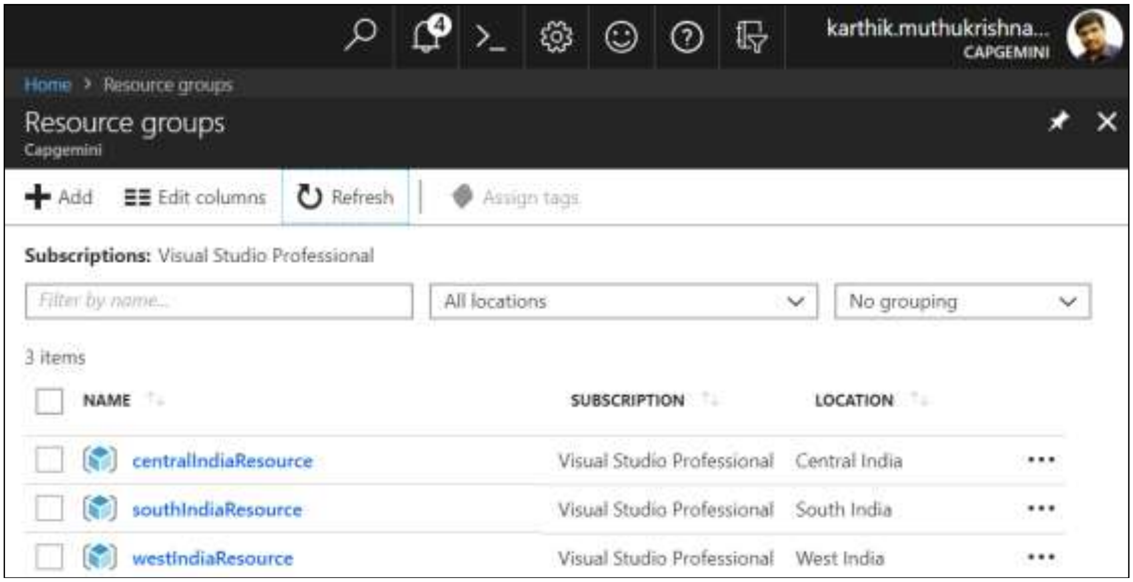
```
{
   "token_type": "Bearer",
   "expires_in": "3600",
   "ext_expires_in": "0",
   "expires_on": "1529579766",
   "not_before": "1529575866",
   "resource": "https://management.azure.com/",
   "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6IlRpb0d5d3dsaHZkRmJY
WjgxM1dwUGF5OUFsVSIsImtpZCI6IlRpb0d5d3dsaHZkRmJYWjgxM1dwUG
F5OUFsVSJ9.eyJhdWQiOiJodHRwczovL21hbmFnZW1lbnQuYXp1cmUuY29t
LyIsImlzcyI6Imh0dHBzOi8vc3RzLndpbmRvd3MubmV0Lzc2YTJhZTVhLTlm
MDAtNGY2Yi05NWVkLTVkMzNkNzdjNGQ2MS8iLCJpYXQiOjE1Mjk1NzU
4NjYsIm5iZiI6MTUyOTU3NTg2NiwiZXhwIjoxNTI5NTc5NzY2LCJhaW8iOiJ
ZMmRnWUZqdzY2WHlTK2Zka3lhSXJKVmNjK1JoTFFBPSIsImFwcGlkIjoiN
zA2NmE2NzYtYjBiNi00MTI5LTljZTctMWQ0NjdlNGRjZmY3IiwiYXBwaWR
hY3IiOiIxIiwiaWRwIjoiaHR0cHM6Ly9zdHMud2luZG93cy5uZXQvNzZhMmFl
NWEtOWYwMC00ZjZiLTk1ZWQtNWQzM2Q3N2M0ZDYxLyIsIm9pZCI6Im
RhNWJhMzY1LTMzMzUtNDRkYy04Mzc1LTk2Mzk4ZjkwOTAxZiIsInN1YiI6
ImRhNWJhMzY1LTMzMzUtNDRkYy04Mzc1LTk2Mzk4ZjkwOTAxZiIsInRpZ
CI6Ijc2YTJhZTVhLTlmMDAtNGY2Yi05NWVkLTVkMzNkNzdjNGQ2MSIsIn
V0aSI6ImNzSnFadC1hQUUyT0hIcU5JUkJnQUEiLCJ2ZXIiOiIxLjAifQ.Yf4na
VPMXCIkLIHCLgRHFykGiDnWo--_w9ecoRoUXqH_OGL20-
WecBfllHl9BFdOfAemoMHSe3f-
U9vN3Ldd5oTZ0urEhq86b1_MQXGvFa6DypvzGshwhWs7dqmXRwssswQDA
ylp41m5MiNAjL5oCZII4oLehbDgyASWor6rYdH-K-ECLWl0BPC-
HlAD9ns4i3x8goEYBlXUy9FKGOA1i3MffhDYMyLSBTu8L153RHe2WPVwg
c3sHXaCe9D1YIA-
4tGE3fmBZkjoF6qXt6efskCZKyMMbNXV6fdC6FsFT8Qq0HhSGIzziHaxswem
l4Z9bAR5SYSD8nss0KZlmjLzCw"
}
```

Step 25: Now that you've successfully acquired an access_token, you can use the token in requests to Web APIs, by including it in the Authorization header

Sample Request:

```
GET /data HTTP/1.1
Host: service.contoso.com
Authorization: Bearer <access_token>
```

Step 26: Create some 3 Resource Groups using Azure Portal



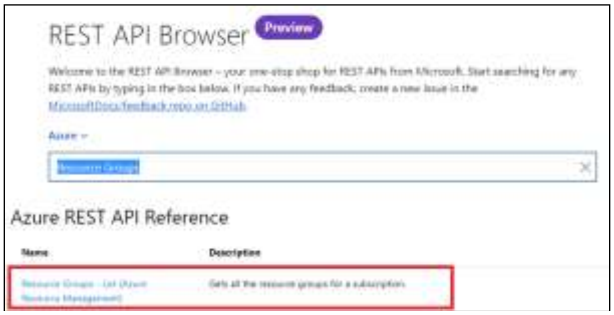
Resource groups
Capgemini

Subscriptions: Visual Studio Professional

3 items

□	NAME	SUBSCRIPTION	LOCATION	
□	centralIndiaResource	Visual Studio Professional	Central India	•••
□	southIndiaResource	Visual Studio Professional	South India	•••
□	westIndiaResource	Visual Studio Professional	West India	•••

Step 27: Open Azure REST API Browser - Azure REST API Reference using the following URL : https://docs.microsoft.com/en-us/rest/api/?view=Azure and search for **Resource Groups**



REST API Browser

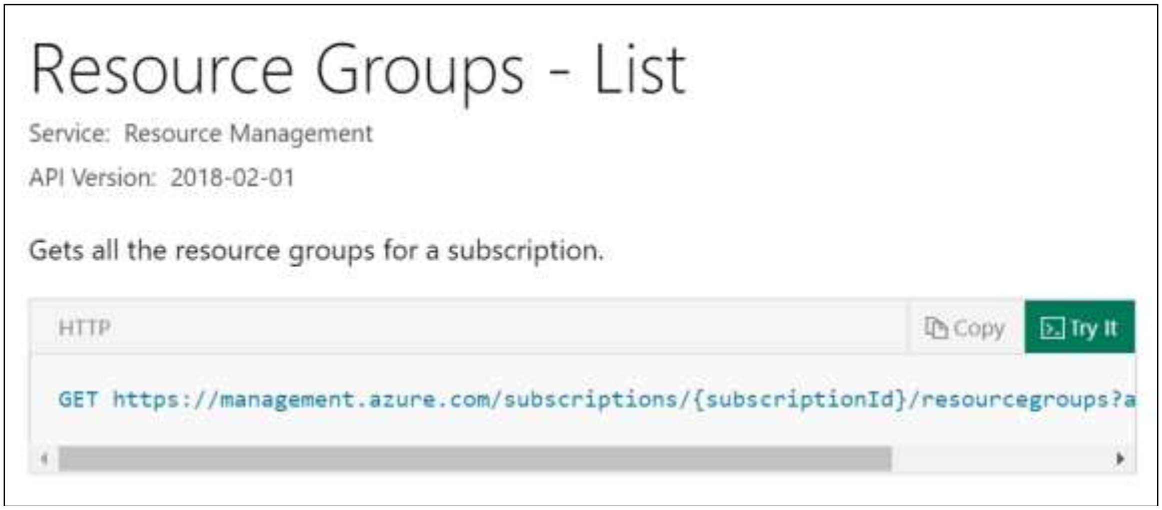
Welcome to the REST API Browser – your one-stop shop for REST APIs from Microsoft. Start searching for any REST APIs by typing in the box below. If you have any feedback, create a new issue in the MicrosoftDocs/feedback repo on GitHub.

Resource Groups

Azure REST API Reference

Name	Description
Resource Groups - List (Azure Resource Management)	Gets all the resource groups for a subscription.

Step 28: Open the Link to get the API Definition to get all the resource groups for a subscription



Resource Groups - List

Service: Resource Management

API Version: 2018-02-01

Gets all the resource groups for a subscription.

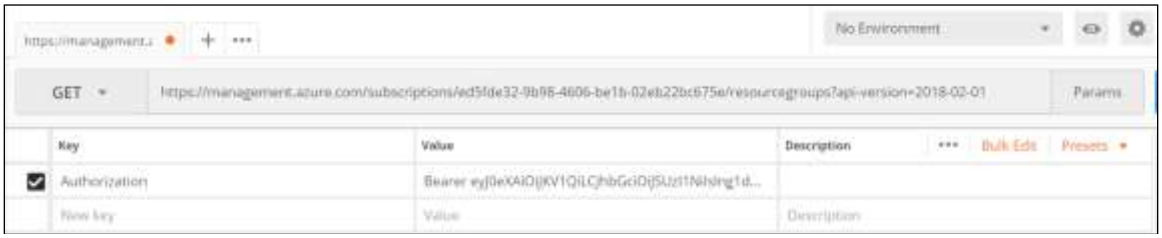
```
GET https://management.azure.com/subscriptions/{subscriptionId}/resourcegroups?a
```

Step 29: Provide a GET Request by replacing the {subscriptionId} with your subscriptionid and the Access token in the Request Authorization Header as given below in Postman:

HTTP Request

GET: http:// management.azure.com/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourcegroups?api-version=2018-02-01

Request Header:
Authorization: Bearer eyJ0eXA*** 4m05Llw

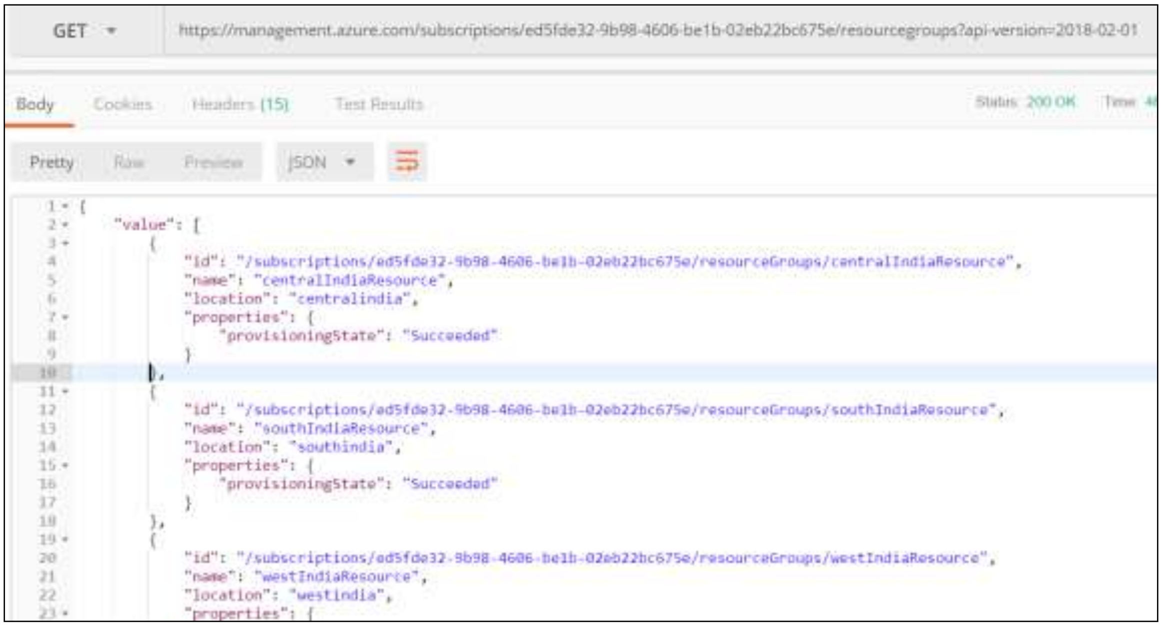


GET https://management.azure.com/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourcegroups?api-version=2018-02-01

	Key	Value	Description
☑	Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1d...	
	New key	Value	Description

Step 30: As a HTTP Response we can see all the Resource groups of our subscription

Http Response

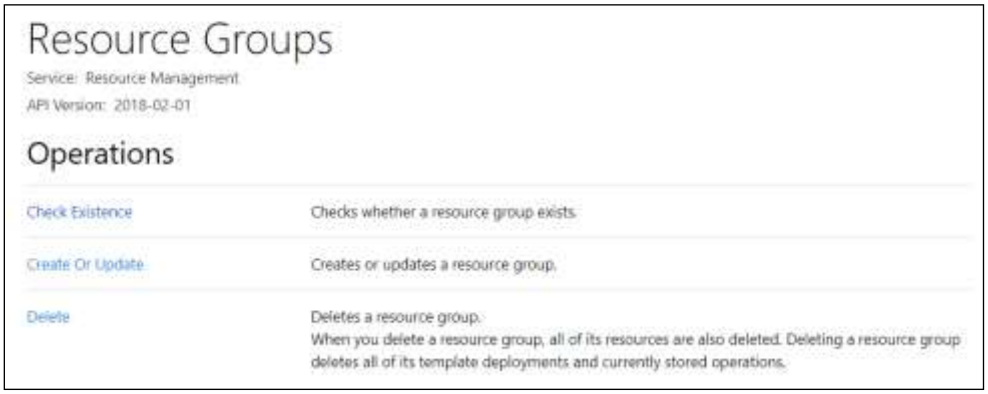


https://management.azure.com/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourcegroups?api-version=2018-02-01

```
{
    "value": [
        {
            "id": "/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourceGroups/centralIndiaResource",
            "name": "centralIndiaResource",
            "location": "centralindia",
            "properties": {
                "provisioningState": "Succeeded"
            }
        },
        {
            "id": "/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourceGroups/southIndiaResource",
            "name": "southIndiaResource",
            "location": "southindia",
            "properties": {
                "provisioningState": "Succeeded"
            }
        },
        {
            "id": "/subscriptions/ed5fde32-9b98-4606-be1b-02eb22bc675e/resourceGroups/westIndiaResource",
            "name": "westIndiaResource",
            "location": "westindia",
            "properties": {
```

Stretched Assignments:

Try the following Resource Groups operations in the same way by referring the API Definitions from Azure REST API Reference



Resource Groups

Service: Resource Management

API Version: 2018-02-01

Operations

Check Existence	Checks whether a resource group exists.
Create Or Update	Creates or updates a resource group.
Delete	Deletes a resource group. When you delete a resource group, all of its resources are also deleted. Deleting a resource group deletes all of its template deployments and currently stored operations.

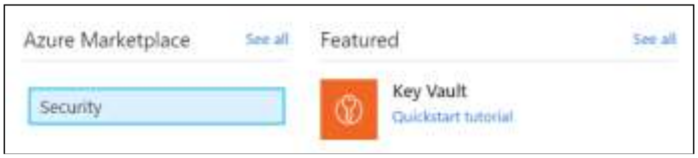
Lab 10. Working with Key Vault

Goals	Create a Key Vault and store a secret in it and access the secret value in C# console application
Time	60 Mins

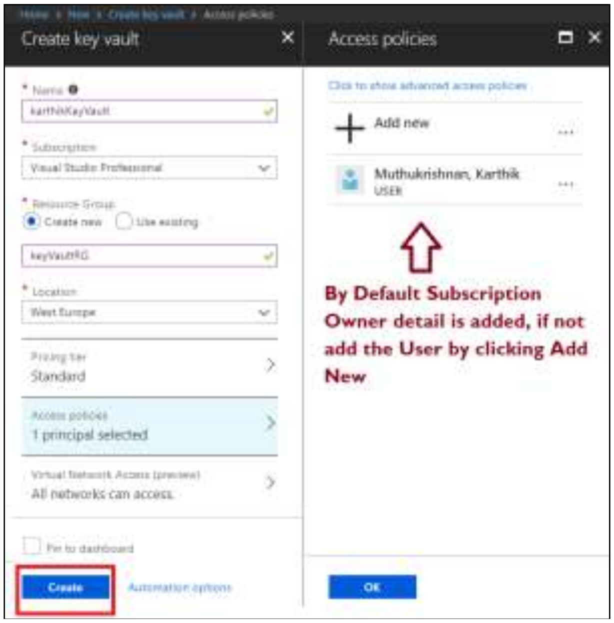
Step 1: Log in to the Azure portal at http://portal.azure.com

Step 2: Choose Create a resource in the upper left-hand corner of the Azure portal.

Step 3: Click Security in Marketplace and select Key Vault



Step 4: Create Key Vault with a unique Key Vault Name and Create / Assign a Resource Group and click create


Home > New > Create key vault > Access policies
Create key vault
Access policies
Click to show advanced access policies
Add new
Create
Automation options
OK

* Name: karthikKeyVault

* Subscription: Visual Studio Professional

* Resource Group: ● Create new ○ Use existing

keyVaultRG

* Location: West Europe

Pricing tier: Standard

Access policies: 1 principal selected

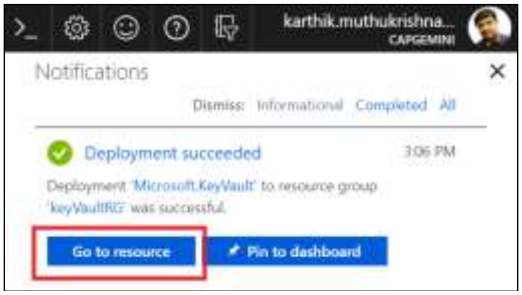
Virtual Network Access (preview): All networks can access.

□ Pin to dashboard

Muthukrishnan, Karthik
USER

By Default Subscription Owner detail is added, if not add the User by clicking Add New

Step 5: Click Goto Resource once deployment succeeded



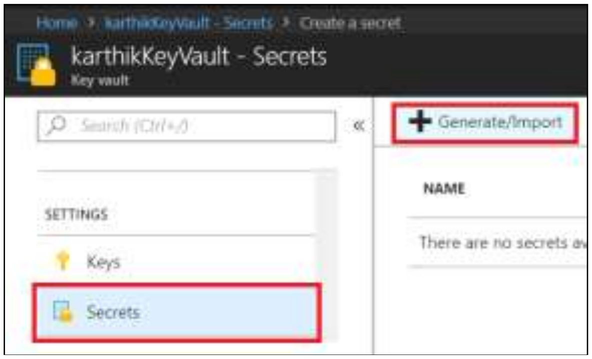
Notifications

Dismiss: Informational Completed All

Deployment succeeded 3:06 PM

Deployment 'Microsoft.KeyVault' to resource group 'keyVaultRG' was successful.

Step 6: To create secret click **Secret** under the settings and then click **Generate / Import**

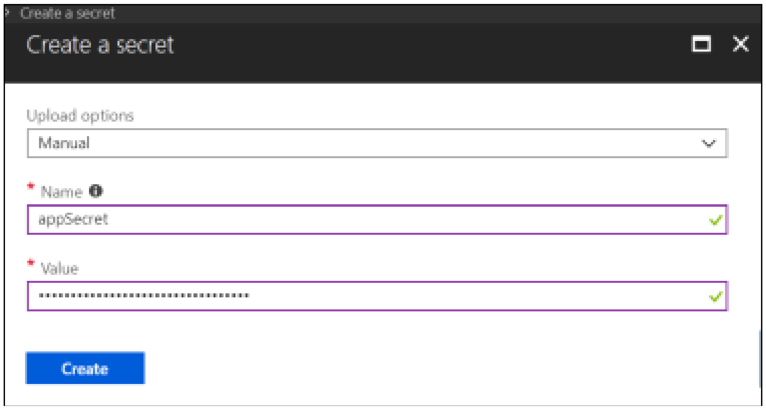


karthikKeyVault - Secrets

Key vault

There are no secrets av

Step 7: Create a Secret with the name **appSecret** and value as **Secret created in Azure Key Vault** given below

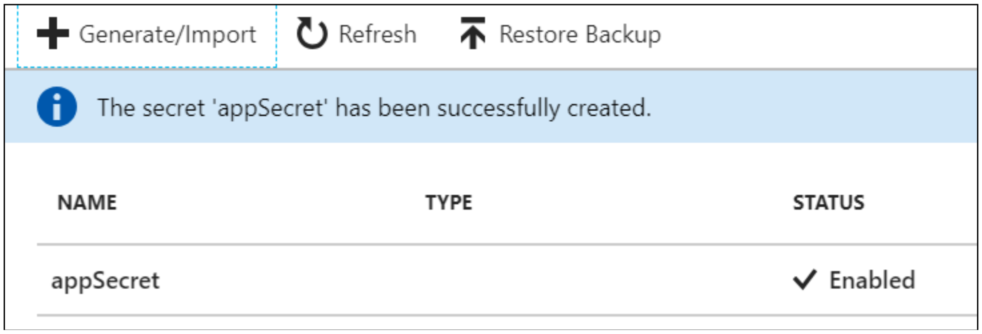


Create a secret

Upload options: Manual

* Name: appSecret

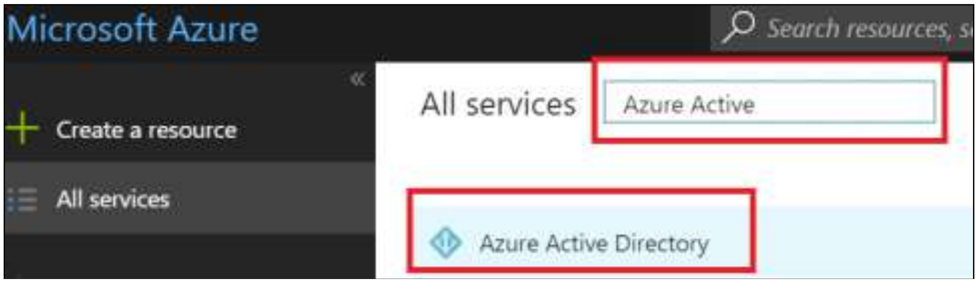
* Value: ••••••••••••••••••••••••••••••••••••



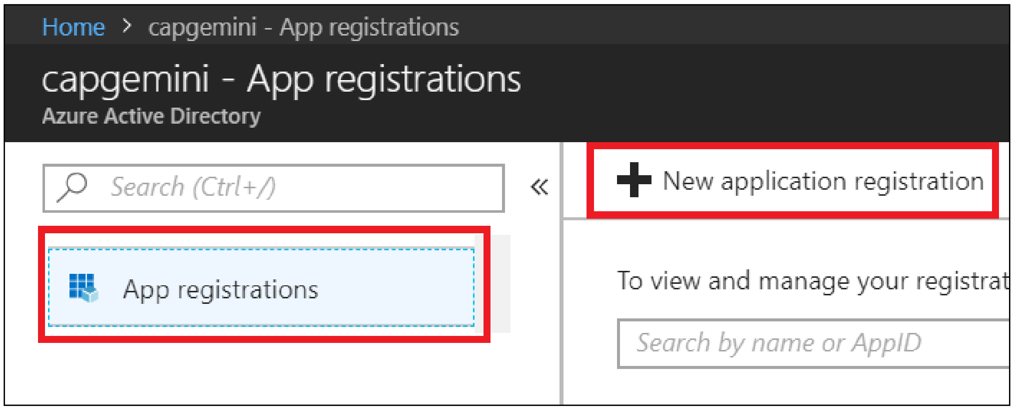
The secret 'appSecret' has been successfully created.

NAME	TYPE	STATUS
appSecret		✓ Enabled

Step 8: Open Azure Active Directory by clicking All services to register an app to access the secret created in Azure Vault

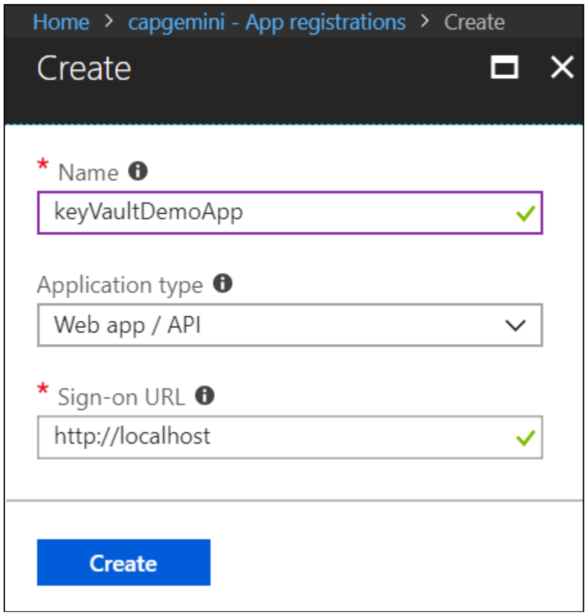


Step 9: Click App Registrations under Manage Section and click New application registration.


Home > capgemini - App registrations
capgemini - App registrations
Azure Active Directory
Search (Ctrl+/)
«
New application registration
App registrations
Search by name or AppID

To view and manage your registrat

Step 10: Create App with the name **keyVaultDemoApp** and select the application type as **Web app / API** and provide the Sign-on URL as **http://localhost** and click create

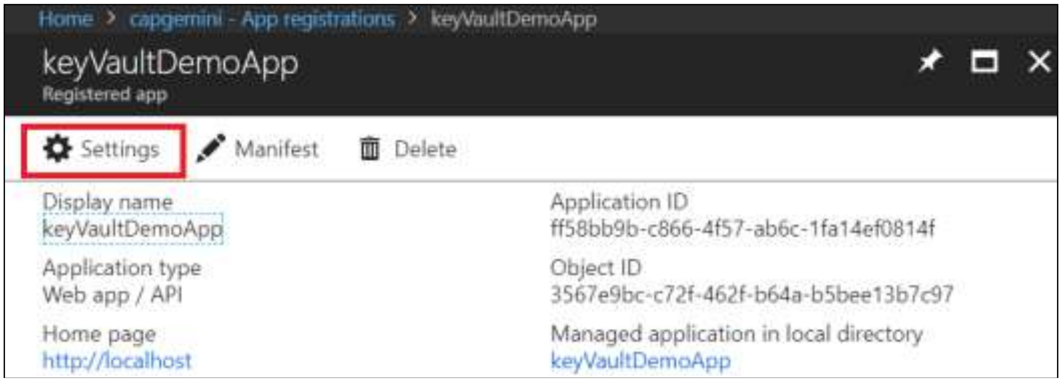


* Name: keyVaultDemoApp

Application type: Web app / API

* Sign-on URL: http://localhost

Step 11: Create App with the name **keyVaultDemoApp** and select the application type as **Web app / API** and provide the Sign-on URL as **http://localhost** and click create



Display name: keyVaultDemoApp

Application type: Web app / API

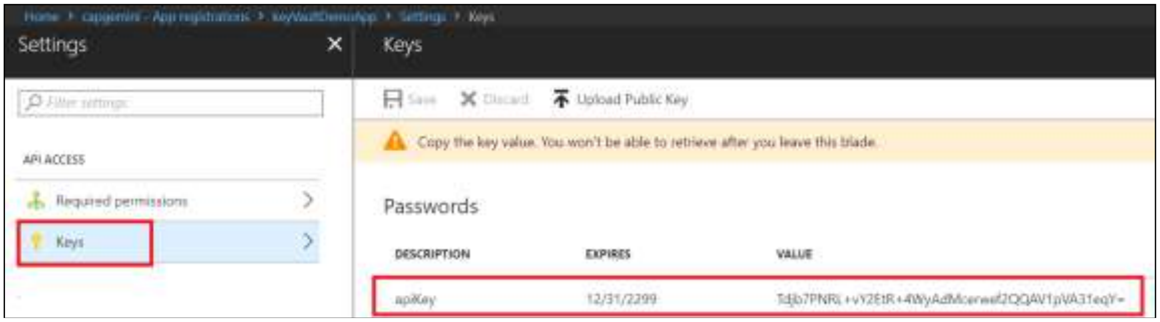
Home page: http://localhost

Application ID: ff58bb9b-c866-4f57-ab6c-1fa14ef0814f

Object ID: 3567e9bc-c72f-462f-b64a-b5bee13b7c97

Managed application in local directory: keyVaultDemoApp

Step 12: Click Settings from the Registered app and then click **Keys** under **API ACCESS** and save key with value **apiKey** and set Expires to **Never expires**



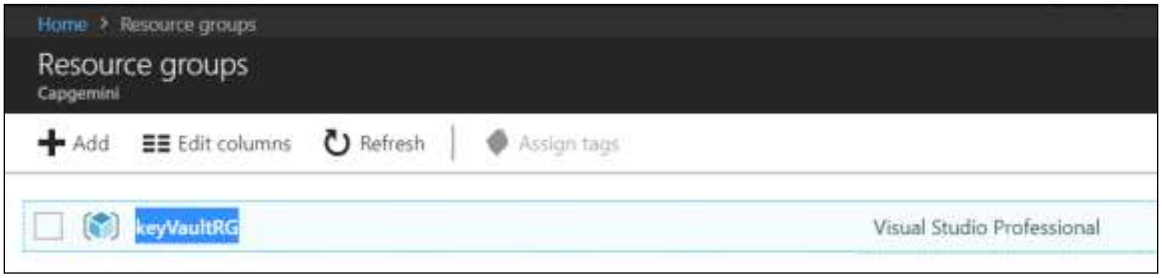
Copy the key value. You won't be able to retrieve after you leave this blade.

Passwords

DESCRIPTION	EXPIRES	VALUE
apiKey	12/31/2299	Tdjb7PNRL+vY2EtR+4WyAdMcerwef2QQAV1pVA31eqY=

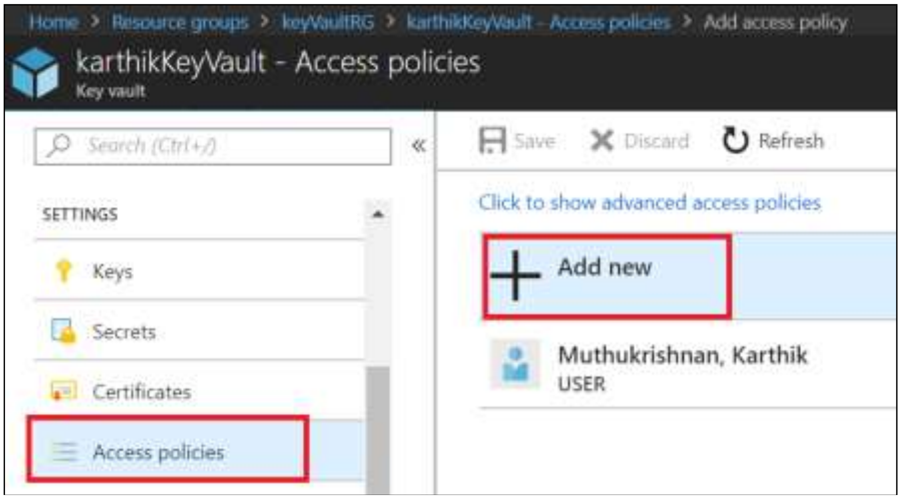
Step 13: Copy the generated Key value in notepad

Step 14: Open the Resource Group **keyVaultRG**



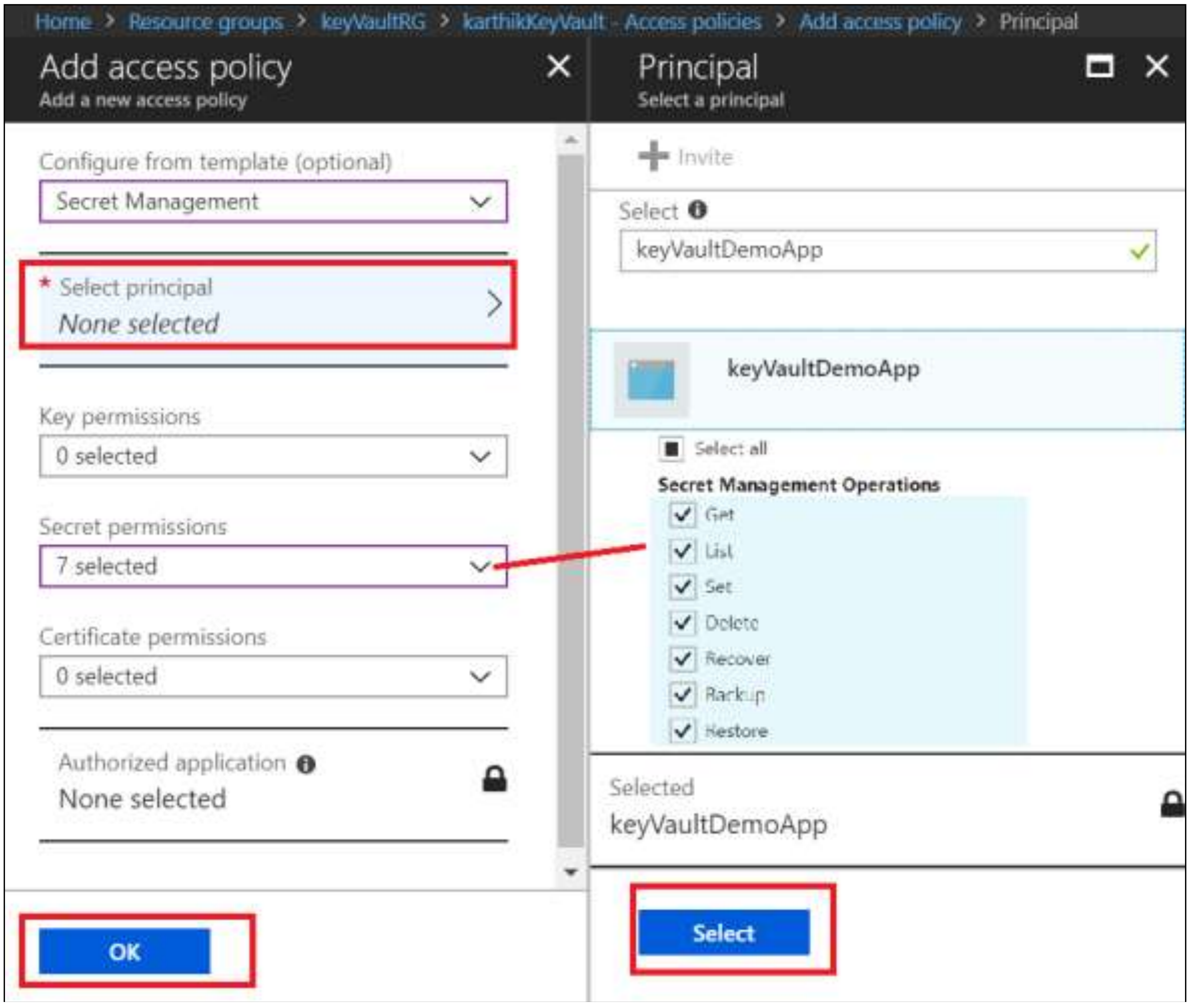
□ keyVaultRG Visual Studio Professional

Step 15: Open the KeyVault (**karthikKeyVault**) then click Access Policies under Settings and click Add new link to add the app


Home > Resource groups > keyVaultRG > karthikKeyVault - Access policies > Add access policy
karthikKeyVault - Access policies
Key vault
Search (Ctrl+/)
SETTINGS
Keys
Secrets
Certificates
Access policies
Save
Discard
Refresh
Click to show advanced access policies
Add new

Muthukrishnan, Karthik
USER

Step 16: Open the KeyVault (**karthikKeyVault**) then click Access Policies under Settings and click **Add new** link to add the app by selecting the required secret permission and click ok


Home > Resource groups > keyVaultRG > karthikKeyVault - Access policies > Add access policy > Principal

Add access policy

Add a new access policy

Configure from template (optional): Secret Management

* Select principal: None selected

Key permissions: 0 selected

Secret permissions: 7 selected

Certificate permissions: 0 selected

Authorized application: None selected

OK

Principal

Select a principal

Invite

Select: keyVaultDemoApp

keyVaultDemoApp

■ Select all

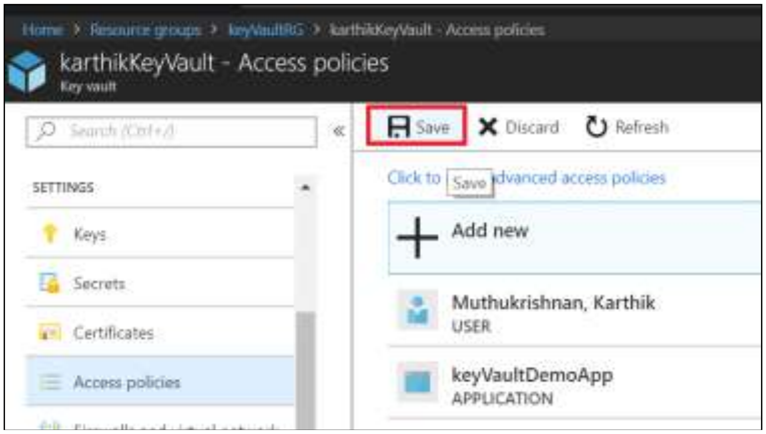
Secret Management Operations

- ☑ Get
- ☑ List
- ☑ Set
- ☑ Delete
- ☑ Recover
- ☑ Backup
- ☑ Restore

Selected: keyVaultDemoApp

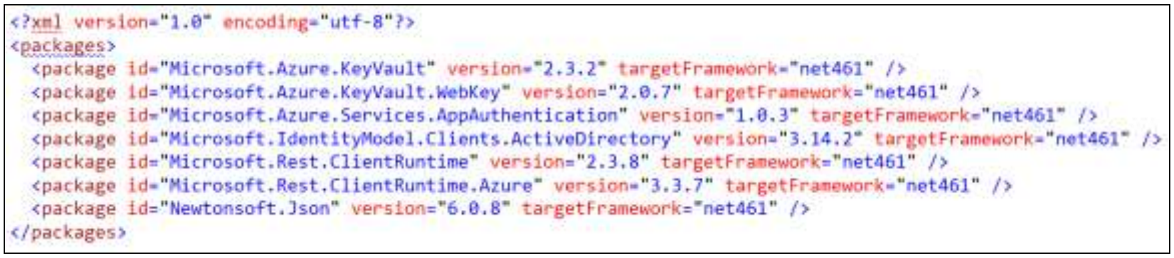
Select

Step 17: Click Save to update the KeyVault



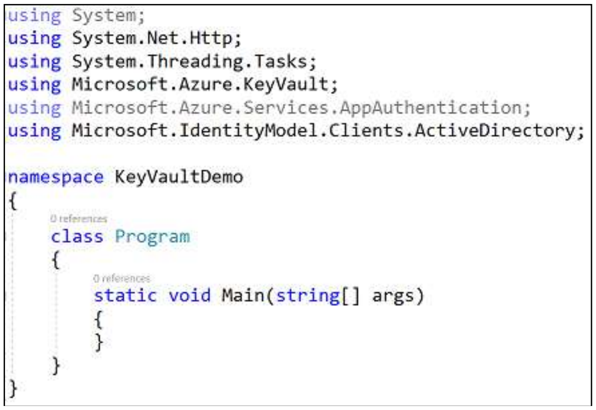
Step 18: Create New console application named **KeyVaultDemo** and add the following Nuget Packages and ensure the package.config file is updated as shown in the image given below

- **Microsoft.Azure.KeyVault**
- **Microsoft.Azure.Services.AppAuthentication**



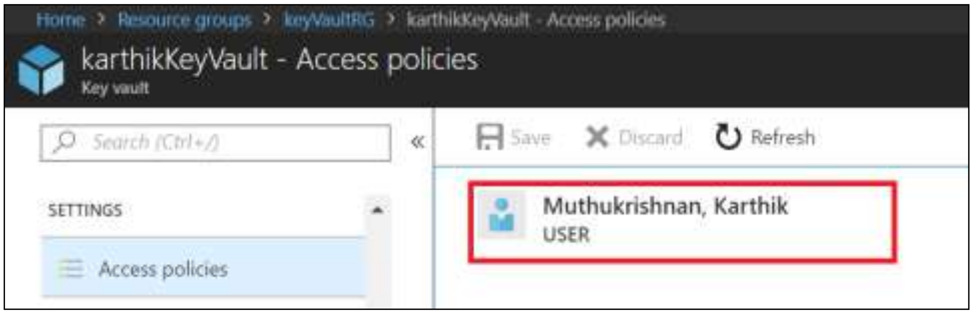
```xml
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="Microsoft.Azure.KeyVault" version="2.3.2" targetFramework="net461" />
  <package id="Microsoft.Azure.KeyVault.WebKey" version="2.0.7" targetFramework="net461" />
  <package id="Microsoft.Azure.Services.AppAuthentication" version="1.0.3" targetFramework="net461" />
  <package id="Microsoft.IdentityModel.Clients.ActiveDirectory" version="3.14.2" targetFramework="net461" />
  <package id="Microsoft.Rest.ClientRuntime" version="2.3.8" targetFramework="net461" />
  <package id="Microsoft.Rest.ClientRuntime.Azure" version="3.3.7" targetFramework="net461" />
  <package id="Newtonsoft.Json" version="6.0.8" targetFramework="net461" />
</packages>
```

Step 19: Add the following Namespaces in **Program.cs**

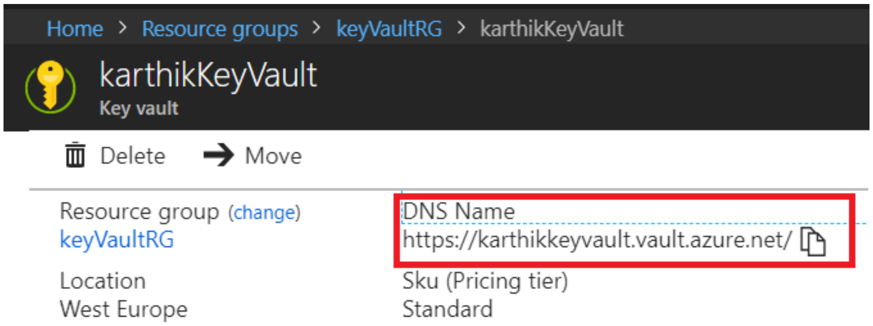


```csharp
using System;
using System.Net.Http;
using System.Threading.Tasks;
using Microsoft.Azure.KeyVault;
using Microsoft.Azure.Services.AppAuthentication;
using Microsoft.IdentityModel.Clients.ActiveDirectory;

namespace KeyVaultDemo
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
        }
    }
}
```

Step 20: To Access the secret key using the user security principle added in **karthikKeyVault** (log in the system as Karthik Muthukrishnan) copy the DNS Name for Vault URL add the following code in **program.cs**


Home > Resource groups > keyVaultRG > karthikKeyVault - Access policies
karthikKeyVault - Access policies
Key vault
Search (Ctrl+/)
«
Save
Discard
Refresh
SETTINGS
Access policies

Muthukrishnan, Karthik
USER



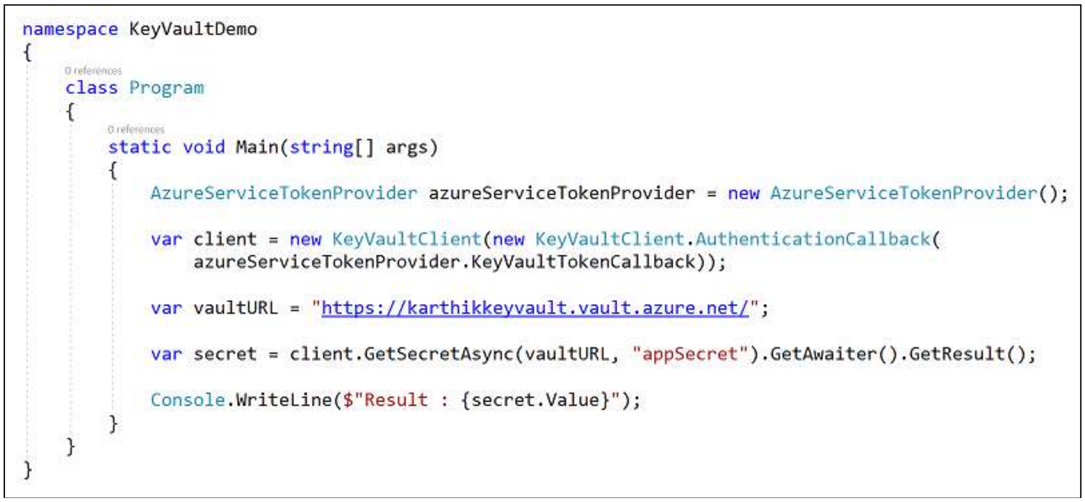
Resource group (change)
keyVaultRG

DNS Name
https://karthikkeyvault.vault.azure.net/

Location
West Europe

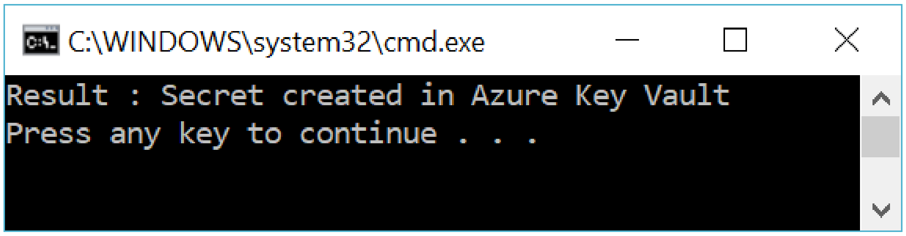
Sku (Pricing tier)
Standard

Program.cs



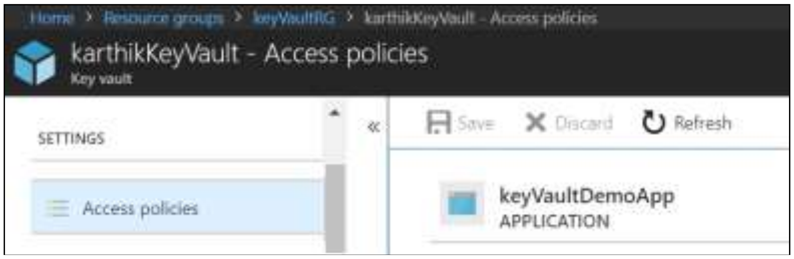
```
namespace KeyVaultDemo
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            AzureServiceTokenProvider azureServiceTokenProvider = new AzureServiceTokenProvider();

            var client = new KeyVaultClient(new KeyVaultClient.AuthenticationCallback(
                azureServiceTokenProvider.KeyVaultTokenCallback));

            var vaultURL = "https://karthikkeyvault.vault.azure.net/";

            var secret = client.GetSecretAsync(vaultURL, "appSecret").GetAwaiter().GetResult();

            Console.WriteLine($"Result : {secret.Value}");
        }
    }
}
```

Step 21: Run the Program CTRL + F5 to see the secret saved in the Key Vault

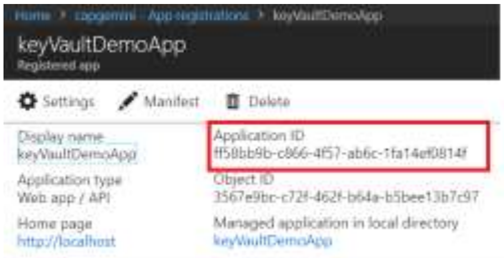


```
Result : Secret created in Azure Key Vault
Press any key to continue . . .
```

Step 22: To access the secret through app security principal


Home > Resource groups > keyVaultRG > karthikKeyVault - Access policies
karthikKeyVault - Access policies
Key vault
SETTINGS
Access policies
Save
Discard
Refresh
keyVaultDemoApp
APPLICATION

Copy the keyVaultDemoApp Application Id from Azure Active Directory App Registration and the key we generated and add the code snippet shown below in program.cs



Display name
keyVaultDemoApp

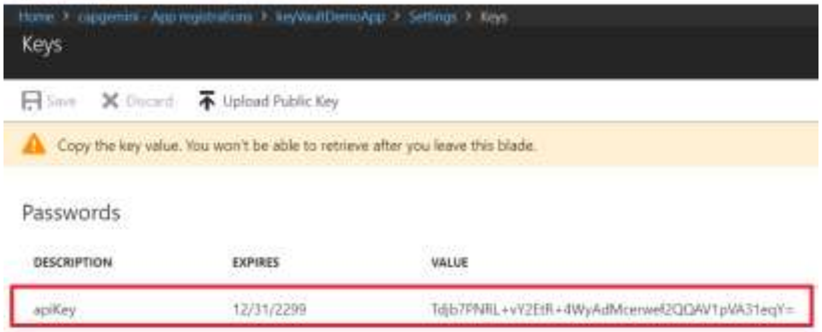
Application ID
ff58bb9b-c866-4f57-ab6c-1fa14ef0814f

Application type
Web app / API

Object ID
3567e9bc-c72f-462f-b64a-b5bee13b7c97

Home page
http://localhost

Managed application in local directory
keyVaultDemoApp

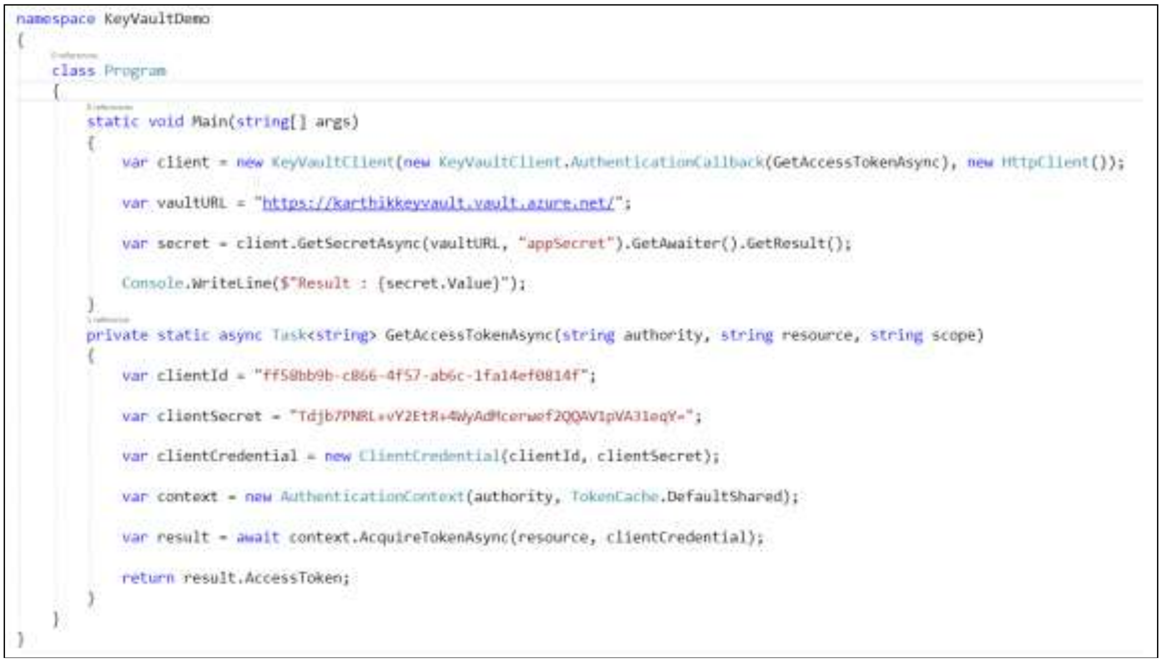

Home > capgemini - App registrations > keyVaultDemoApp > Settings > Keys
Keys
Save
Discard
Upload Public Key

Copy the key value. You won't be able to retrieve after you leave this blade.

Passwords

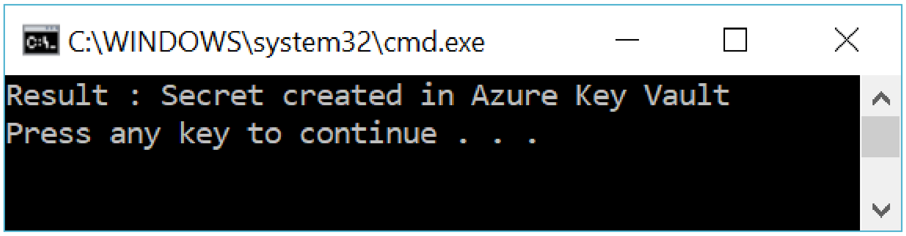
DESCRIPTION	EXPIRES	VALUE
apiKey	12/31/2299	Tdjb7PNRL+vY2EtR+4WyAdMcerwe82QQAV1pVA31eqY=

Program.cs



```
namespace KeyVaultDemo
{
    class Program
    {
        static void Main(string[] args)
        {
            var client = new KeyVaultClient(new KeyVaultClient.AuthenticationCallback(GetAccessTokenAsync), new HttpClient());

            var vaultURL = "https://karthikkeyvault.vault.azure.net/";

            var secret = client.GetSecretAsync(vaultURL, "appSecret").GetAwaiter().GetResult();

            Console.WriteLine($"Result : {secret.Value}");
        }
        private static async Task<string> GetAccessTokenAsync(string authority, string resource, string scope)
        {
            var clientId = "ff58bb9b-c866-4f57-ab6c-1fa14ef0814f";

            var clientSecret = "Tdjb7PNRL+vY2EtR+4WyAdMcerwef2QQAV1pVA31eqY=";

            var clientCredential = new ClientCredential(clientId, clientSecret);

            var context = new AuthenticationContext(authority, TokenCache.DefaultShared);

            var result = await context.AcquireTokenAsync(resource, clientCredential);

            return result.AccessToken;
        }
    }
}
```

Step 22: Run the Program CTRL + F5 to see the secret saved in the Key Vault



```
Result : Secret created in Azure Key Vault
Press any key to continue . . .
```