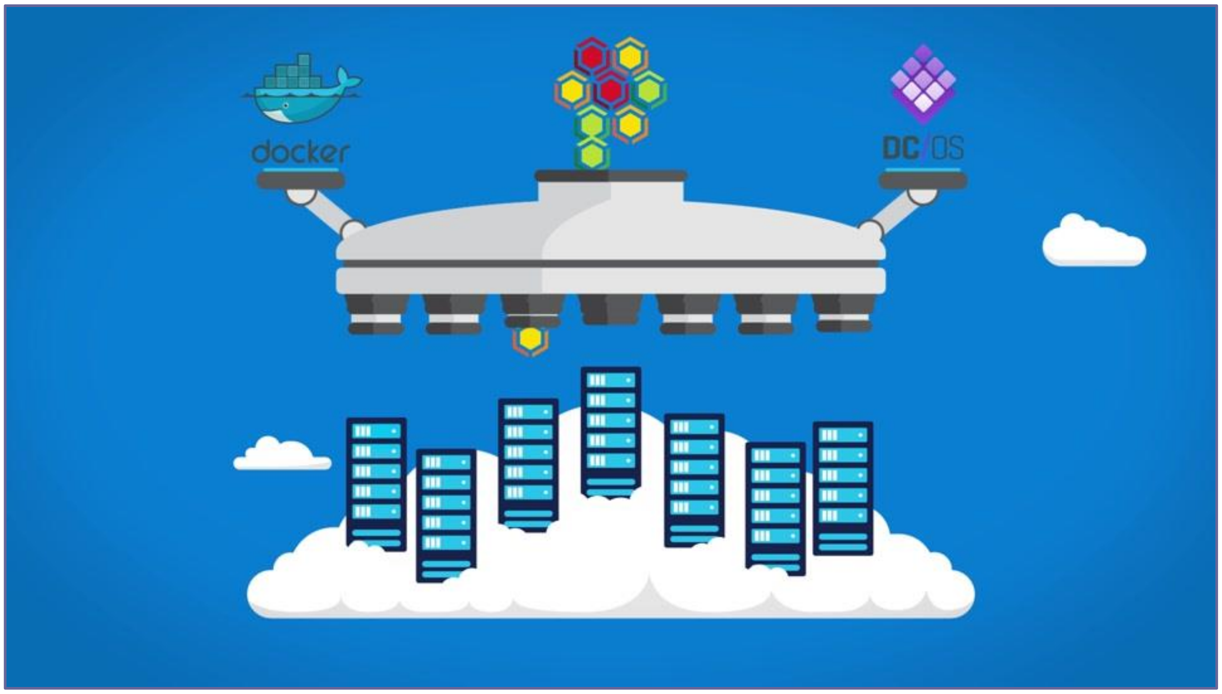


Azure Developer Series

Migrating a legacy ASP.NET 2-tier application
to Azure using Container Services

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer
PDTIT and 007FFFLearning.com

@pdtit @007FFFLearning

Version: October 2018 – 2.0

Contents

Migrating a legacy ASP.NET 2-tiered application to Azure using Container Services - Hands-On-Labs step-by-step

Abstract and Learning Objectives

This workshop enables anyone to learn, understand and build a Proof of Concept, in performing a multi-tiered legacy ASP.NET web application using Microsoft SQL Server database, platform migration to Azure public cloud, leveraging on different Azure Platform Azure A Service (PaaS) and Azure Container Services.

After an introductory module on cloud app migration strategies and patterns, students get introduced to the basics of automating Azure resources deployments using Visual Studio and Azure Resource Manager (ARM) templates. Next, attendees will learn about Microsoft SQL database migration to SQL Azure PaaS, as well as deploying and migrating Azure Web Apps.

After these foundational platform components, the workshop will totally focus on the core concepts and advantages of using containers for running web apps, based on Docker, Azure Container Registry (ACR), Azure Container Instance (ACI), as well as how to enable container cloud-scale using Azure Container Services (ACS) with Kubernetes and Azure Kubernetes Service (AKS).

The focus of the workshop is having a Hands-On-Labs experience, by going through the following exercises and tasks:

- Deploying a 2-tier Azure Virtual Machine (Webserver and SQL database Server) using ARM-template automation with Visual Studio 2017;
- Migrating a legacy SQL 2012 database to Azure SQL PaaS (Lift & Shift);
- Migrating a legacy ASP.NET web application to Azure Web Apps (Lift & Shift);
- Containerizing a legacy ASP.NET web application using Docker;
- Running Azure Container Instance (ACI) from an Azure Container Registry (ACR) image;
- Deploy and run Azure Container Services (ACS) with Kubernetes;
- Deploy and run Azure Kubernetes Services (AKS);
- Managing and Monitoring Azure Container Services (ACS) and Azure Kubernetes Services (AKS);

Hands-On-Lab Scenario

The baseline of the hands-on-lab scenario is starting from an 8-year-old legacy ASP.NET application, developed around 2010, currently offering a product catalog browsing web page, pulling the product catalog list from a legacy Microsoft SQL Server 2012 database, running on dedicated Windows Server 2012 R2 Virtual Machines. (This could be seen as a subset of a larger application landscape within your organization, think of manufacturing database information, HR solutions, e-commerce platforms,… and alike). Imagine this application is running in our on-premises datacenter, where you want to perform an "application digital migration" to Azure Public cloud. You will use several Azure cloud-native services and solutions, ranging from Virtual Machines, Platform Services and different Container Solutions on Azure.

Requirements

Naming Conventions:

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word **"[SUFFIX]"** as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Azure Subscription:

Participants need a "pay-as-you-go", MSDN or other paid Azure subscription

a) Azure Trial subscriptions won't work
b) In one of the Azure Container Services tasks, you are required to create an Azure AD Service Principal, wich typically requires an Azure subscription owner to log in to create this object. If you don't have the owner right in your Azure subscription, you could ask another person to execute this step for you.
c) The Azure subscription must allow you to run enough cores, used by the baseline Virtual Machines, but also later on in the tasks when deploying the Azure Container Services, where ACS agent and master machines are getting set up. If you follow the instructions as written out in the lab guide, you need 12 cores.
d) If you run this lab setup in your personal or corporate Azure payable subscription, using the configuration as described in the lab guide, the estimated Azure consumption costs for running the setups during the 2 days of the workshop is $20.

Other requirements:

Participants need a local client machine, running a recent Operating System, allowing them to:

- browse to https://portal.azure.com from a most-recent browser;
- establish a secured Remote Desktop (RDP) session to a lab-jumpVM running Windows Server 2016;

Alternative Approach:

Where the lab scenario assumes all exercises will be performed from within the lab-jumpVM, (since several tools will be installed on the lab-jumpVM or are already installed by default), participants could also execute (most, if not all…) steps from their local client machine.

The following tools are being used throughout the lab exercises:

- Visual Studio 2017 community edition (updated to latest version)
- Docker for Windows (updated to latest version)
- Azure CLI 2.0 (updated to latest version)
- Kubernetes CLI (updated to latest version)

Make sure you have these tools installed prior to the workshop, if you are not using the lab-jumpVM. You should also have full administrator rights on your machine to execute certain steps within using these tools.

Final Remarks:

VERY IMPORTANT: You should be typing all of the commands as they appear in the guide, except where explicitly stated in this document. Do not try to copy and paste from Word to your command windows or other documents where you are instructed to enter information shown in this document. There can be issues with Copy and Paste from Word or PDF that result in errors, execution of instructions, or creation of file content.

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word "[SUFFIX]" as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Lab 5: Running Azure Container Instance (ACI) from an Azure Container Registry (ACR) image

What you will learn

In this lab, you start from creating a new Azure Container Registry resource. Next, after authenticating to this ACR, you learn how to tag and push the Docker image you created in the previous lab to an Azure Container Registry. Lastly, you run an Azure Container Instance, based on the Docker image that is stored in the Azure Container Registry.

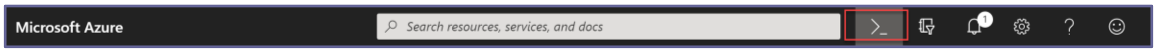
Time Estimate

This lab should take about an hour to complete.

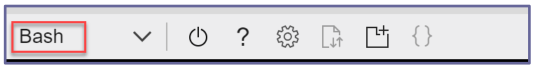
Task 1: Creating an Azure Container Registry using Azure CloudShell

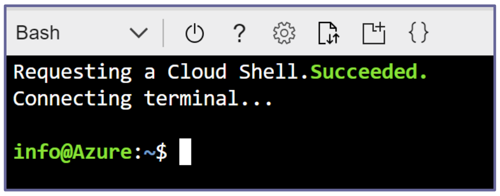
As we now validated our Docker image is running successfully in a container, we can migrate this image to an Azure Container Registry (ACR). This will allow to reuse this image for future Azure Container solutions we will be talking about.

1. Log on to the Azure Portal, http://portal.azure.com, with your Azure admin credentials. From here, Open Cloud Shell



2. Follow the configuration steps if this is the first time you launched Cloud Shell. In the Environment, make sure you choose Bash.

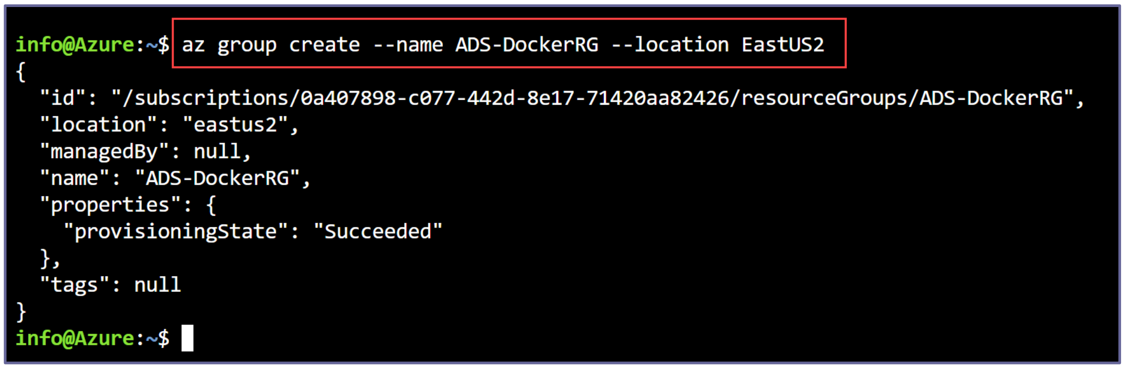




```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

info@Azure:~$
```

3. Execute the following Azure CLI commands, to create a new Azure Resource Group:

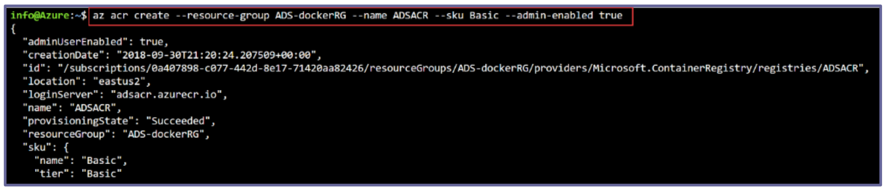
```
az group create --name [SUFFIX]-dockerRG --location EastUS2
```



```
info@Azure:~$ az group create --name ADS-DockerRG --location EastUS2
{
  "id": "/subscriptions/0a407898-c077-442d-8e17-71420aa82426/resourceGroups/ADS-DockerRG",
  "location": "eastus2",
  "managedBy": null,
  "name": "ADS-DockerRG",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
info@Azure:~$
```

4. Followed by another Azure CLI command to **create the Azure Container Registry**:

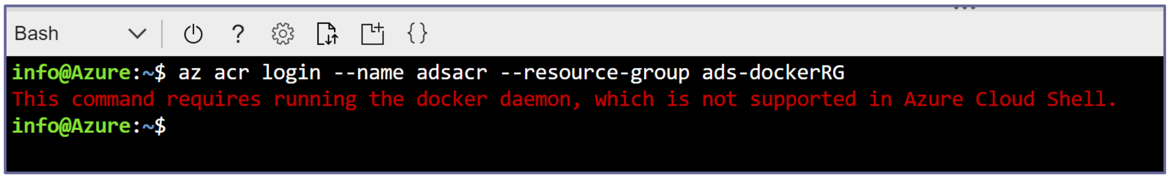
   ```
   az acr create --resource-group [Suffix]-dockerRG --name
   [SUFFIX]ACR --sku Basic --admin-enabled true
   ```



```
info@Azure:~$ az acr create --resource-group ADS-dockerRG --name ADSACR --sku Basic --admin-enabled true
{
  "adminUserEnabled": true,
  "creationDate": "2018-09-30T21:20:24.207509+00:00",
  "id": "/subscriptions/0a407898-c077-442d-8e17-71420aa82426/resourceGroups/ADS-dockerRG/providers/Microsoft.ContainerRegistry/registries/ADSACR",
  "location": "eastus2",
  "loginServer": "adsacr.azurecr.io",
  "name": "ADSACR",
  "provisioningState": "Succeeded",
  "resourceGroup": "ADS-dockerRG",
  "sku": {
    "name": "Basic",
    "tier": "Basic"
```

5. The next involves connecting to the Azure Container Registry we just created, and pushing our Docker image into it. This relies on the following command:

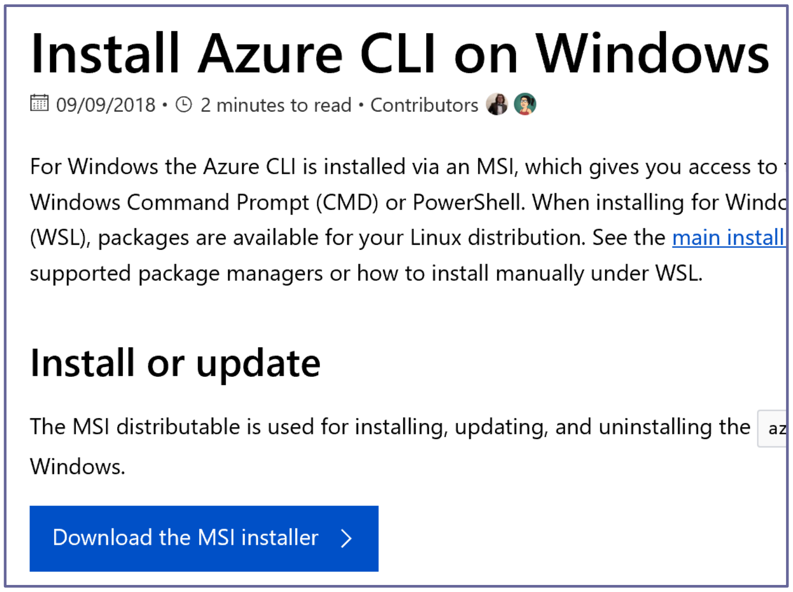
   ```
   az acr login –name [SUFFIX]ACR –resource-group [SUFFIX]-dockerRG
   ```



```
info@Azure:~$ az acr login --name adsacr --resource-group ads-dockerRG
This command requires running the docker daemon, which is not supported in Azure Cloud Shell.
info@Azure:~$
```

6. This means, we have to execute the remaining commands from our local lab-jumpVM, instead of the Azure Cloud Shell. We should install the Azure CLI for Windows using the following link: https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-windows?view=azure-cli-latest

7. From the appearing web page, **scroll down** to the **Download MSI installer** button**, and click it.**


Install Azure CLI on Windows

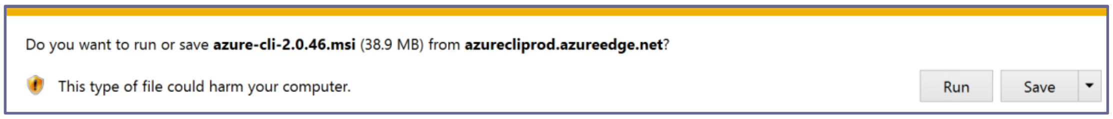
09/09/2018 • 2 minutes to read • Contributors

For Windows the Azure CLI is installed via an MSI, which gives you access to
Windows Command Prompt (CMD) or PowerShell. When installing for Windo
(WSL), packages are available for your Linux distribution. See the main install
supported package managers or how to install manually under WSL.

Install or update

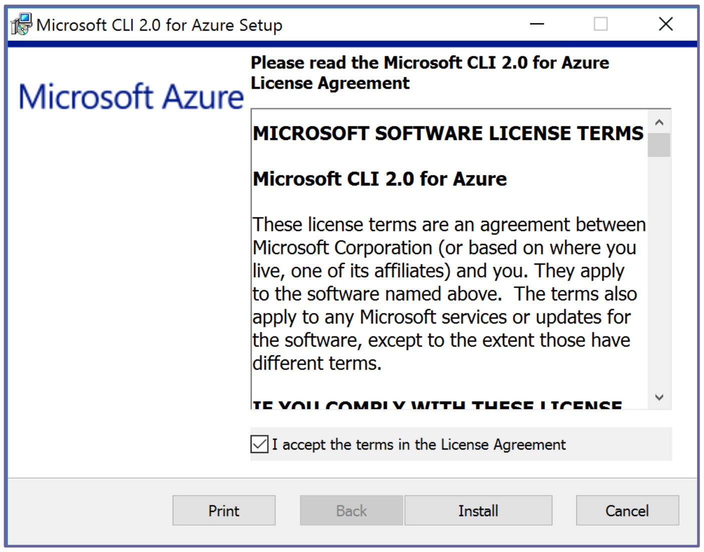
The MSI distributable is used for installing, updating, and uninstalling the `az`
Windows.

8. At the **download prompt**, **choose SAVE**; once the file is downloaded, select **RUN**


Do you want to run or save **azure-cli-2.0.46.msi** (38.9 MB) from **azurecliprod.azureedge.net**?

This type of file could harm your computer.

9. The Microsoft CLI 2.0 for Azure Setup Installer launches


Microsoft CLI 2.0 for Azure Setup

Microsoft Azure

Please read the Microsoft CLI 2.0 for Azure License Agreement

MICROSOFT SOFTWARE LICENSE TERMS

Microsoft CLI 2.0 for Azure

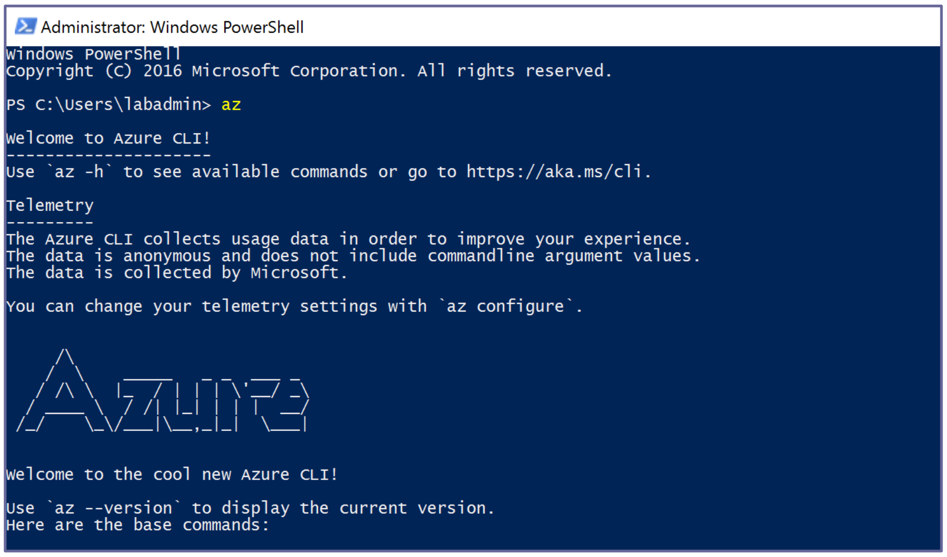
These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. They apply to the software named above. The terms also apply to any Microsoft services or updates for the software, except to the extent those have different terms.

☑ I accept the terms in the License Agreement

10. **Press the Install button** to continue. Wait for the installation to **finish** successfully.

11. To **validate** the Azure CLI is installed fine, **open a new PowerShell window,** and initiate the following command:

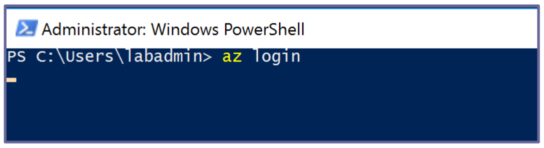
az



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\labadmin> az

Welcome to Azure CLI!
---------------------
Use `az -h` to see available commands or go to https://aka.ms/cli.

Telemetry
---------
The Azure CLI collects usage data in order to improve your experience.
The data is anonymous and does not include commandline argument values.
The data is collected by Microsoft.

You can change your telemetry settings with `az configure`.

     /\
    /  \    _____   _ _  ___ _
   / /\ \  |_  / | | | \'__/ _ \
  / ____ \  / /| |_| | | |  __/
 /_/    \_\/___|\__,_|_|  \___|


Welcome to the cool new Azure CLI!

Use `az --version` to display the current version.
Here are the base commands:
```

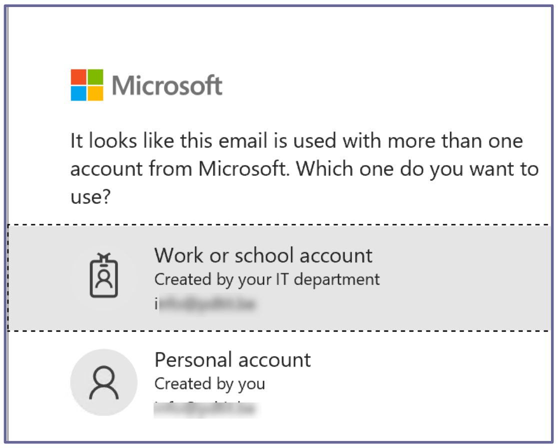
12. This confirms Azure CLI 2.0 is running as expected. We can continue with our Azure Container Registry creation process. But first, we need to "authenticate" our session to Azure, by running the following command:

az login



```
Administrator: Windows PowerShell
PS C:\Users\labadmin> az login
_
```

13. This opens your internet browsers, and prompts for your Azure admin credentials:



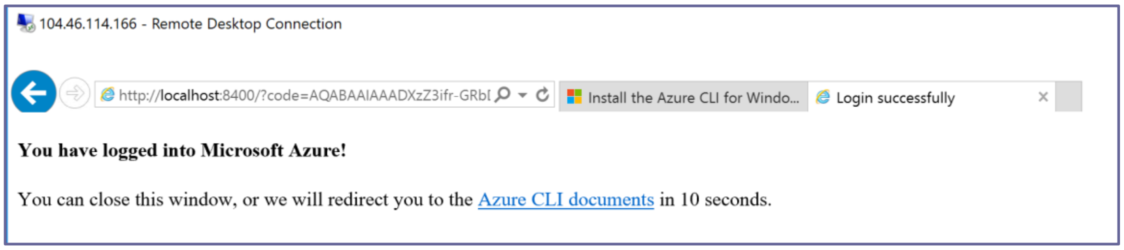
Microsoft

It looks like this email is used with more than one account from Microsoft. Which one do you want to use?

Work or school account
Created by your IT department
i[redacted]

Personal account
Created by you
[redacted]

14. After successful login, the following information is displayed:

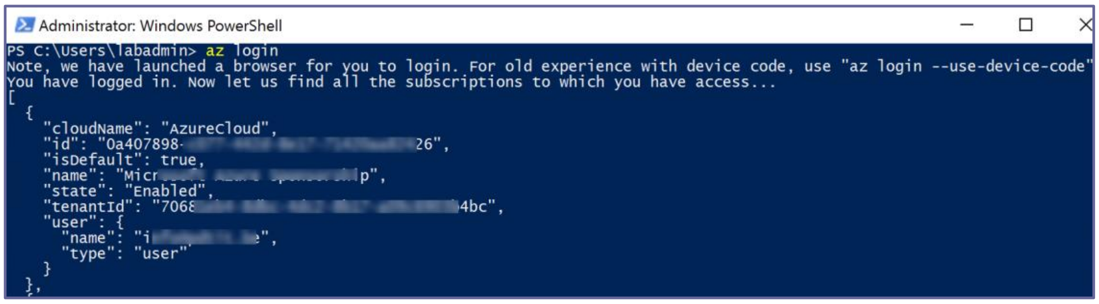

104.46.114.166 - Remote Desktop Connection
http://localhost:8400/?code=AQABAAIAAADXzZ3ifr-GRbI
Install the Azure CLI for Windo...
Login successfully

You have logged into Microsoft Azure!

You can close this window, or we will redirect you to the Azure CLI documents in 10 seconds.

15. You can close the internet browser.

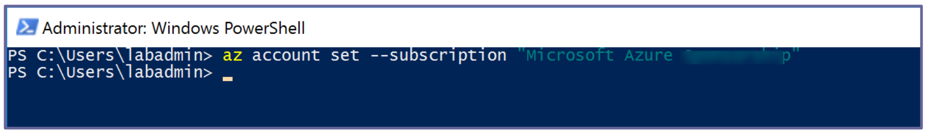
16. When you go back to the PowerShell window, it will show you the JSON output of your Azure subscription, related to this Azure admin user:



```
PS C:\Users\labadmin> az login
Note, we have launched a browser for you to login. For old experience with device code, use "az login --use-device-code"
You have logged in. Now let us find all the subscriptions to which you have access...
[
  {
    "cloudName": "AzureCloud",
    "id": "0a407898-[redacted]26",
    "isDefault": true,
    "name": "Micr[redacted]p",
    "state": "Enabled",
    "tenantId": "7068[redacted]4bc",
    "user": {
      "name": "i[redacted]e",
      "type": "user"
    }
  },
```

17. If you should have multiple Azure subscriptions linked to the same Azure admin credentials, run the following AZ CLI command to guarantee you are working in the correct subscription:

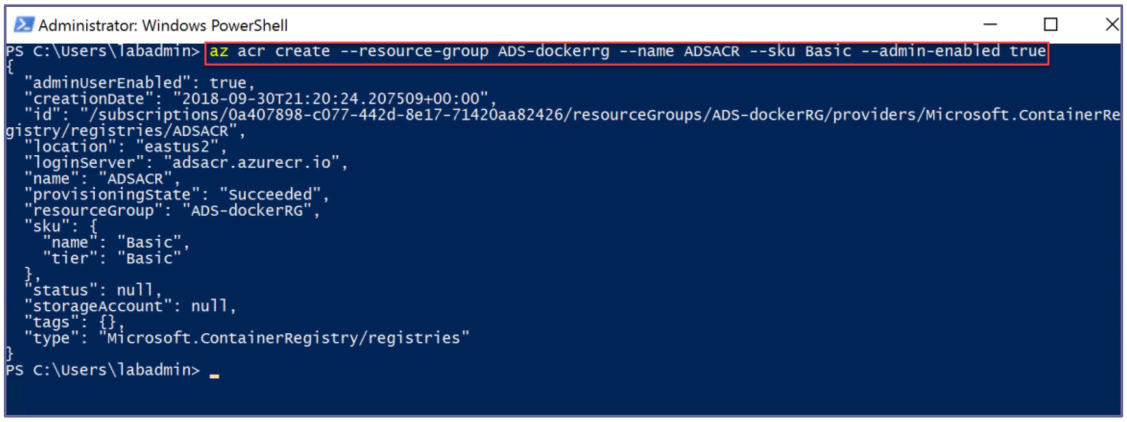
```
az account set --subscription "your subscription name here"
```



```
PS C:\Users\labadmin> az account set --subscription "Microsoft Azure [redacted]p"
PS C:\Users\labadmin> _
```

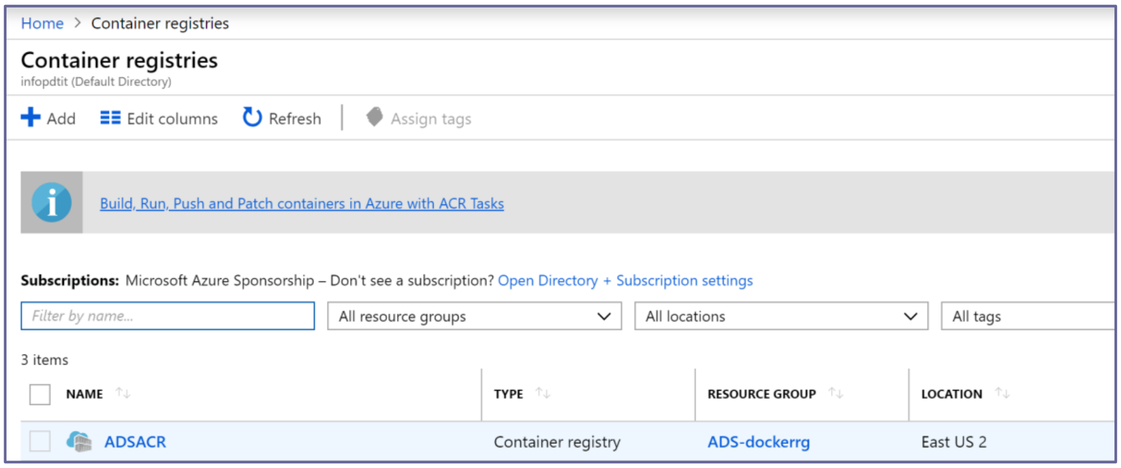
18. Let's try to redo our Azure Container Registry process, by executing the following command:

az acr create --resource-group [SUFFIX]-DockerRG --name [SUFFIX]ACR --sku Basic --admin-enabled true



```
PS C:\Users\labadmin> az acr create --resource-group ADS-dockerrg --name ADSACR --sku Basic --admin-enabled true
{
  "adminUserEnabled": true,
  "creationDate": "2018-09-30T21:20:24.207509+00:00",
  "id": "/subscriptions/0a407898-c077-442d-8e17-71420aa82426/resourceGroups/ADS-dockerRG/providers/Microsoft.ContainerRe
gistry/registries/ADSACR",
  "location": "eastus2",
  "loginServer": "adsacr.azurecr.io",
  "name": "ADSACR",
  "provisioningState": "Succeeded",
  "resourceGroup": "ADS-dockerRG",
  "sku": {
    "name": "Basic",
    "tier": "Basic"
  },
  "status": null,
  "storageAccount": null,
  "tags": {},
  "type": "Microsoft.ContainerRegistry/registries"
}
PS C:\Users\labadmin> _
```

19. While the JSON output is here, you can also validate from the Azure Portal


Home > Container registries
Container registries
infopdtit (Default Directory)
Add
Edit columns
Refresh
Assign tags
Build, Run, Push and Patch containers in Azure with ACR Tasks

Subscriptions: Microsoft Azure Sponsorship – Don't see a subscription? Open Directory + Subscription settings

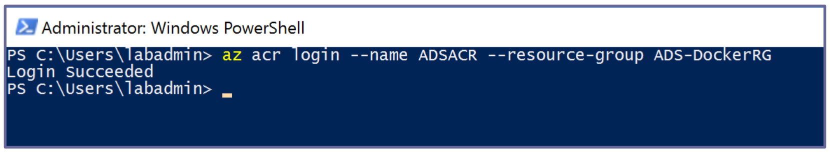
Filter by name... | All resource groups | All locations | All tags

3 items

NAME	TYPE	RESOURCE GROUP	LOCATION
ADSACR	Container registry	ADS-dockerrg	East US 2

20. Next, we need to authenticate to the Azure Container Registry itself, using the following command:

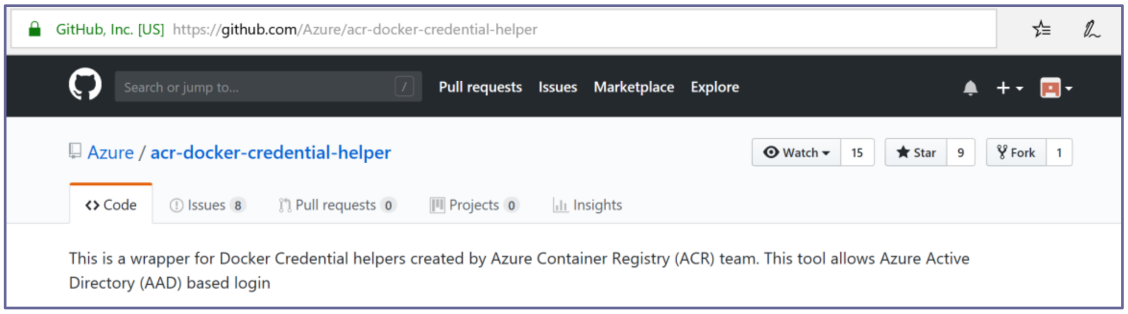
```
az acr login --name ADSACR --resource-group ADS-DockerRG
```



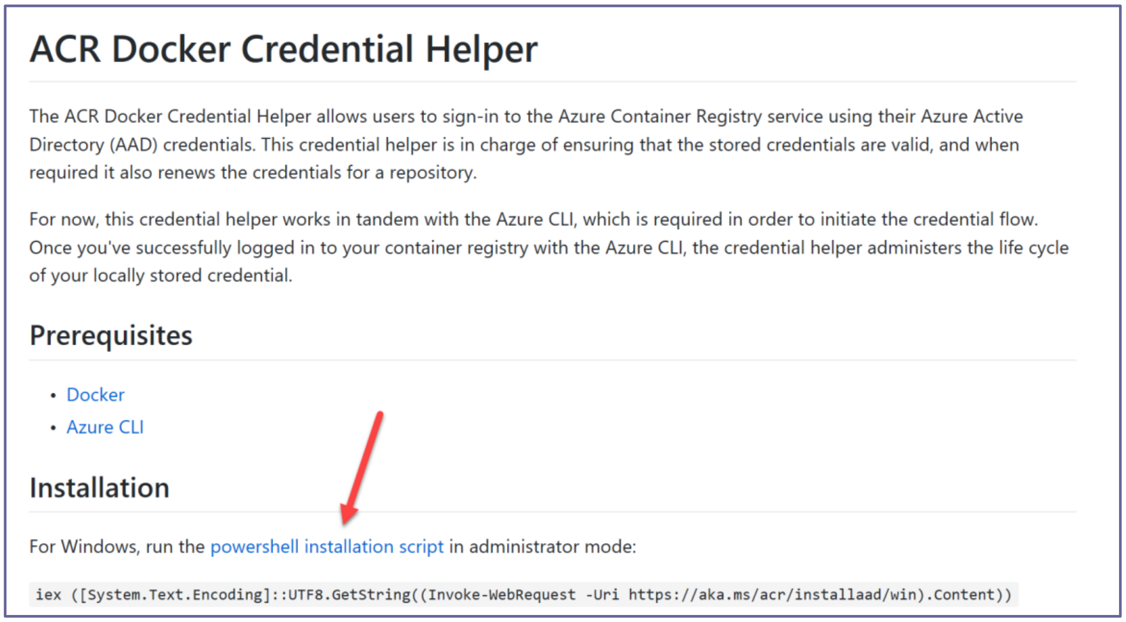
```
PS C:\Users\labadmin> az acr login --name ADSACR --resource-group ADS-DockerRG
Login Succeeded
PS C:\Users\labadmin> _
```

21. Note, ONLY if the above command should fail, it is most probably related to an issue with the Windows Credential Store. The work-around for now is running another PowerShell that is available on GitHub, allowing to install the ACR-Docker-Credential-Helper:

https://github.com/Azure/acr-docker-credential-helper



This is a wrapper for Docker Credential helpers created by Azure Container Registry (ACR) team. This tool allows Azure Active Directory (AAD) based login



ACR Docker Credential Helper

The ACR Docker Credential Helper allows users to sign-in to the Azure Container Registry service using their Azure Active Directory (AAD) credentials. This credential helper is in charge of ensuring that the stored credentials are valid, and when required it also renews the credentials for a repository.

For now, this credential helper works in tandem with the Azure CLI, which is required in order to initiate the credential flow. Once you've successfully logged in to your container registry with the Azure CLI, the credential helper administers the life cycle of your locally stored credential.

Prerequisites

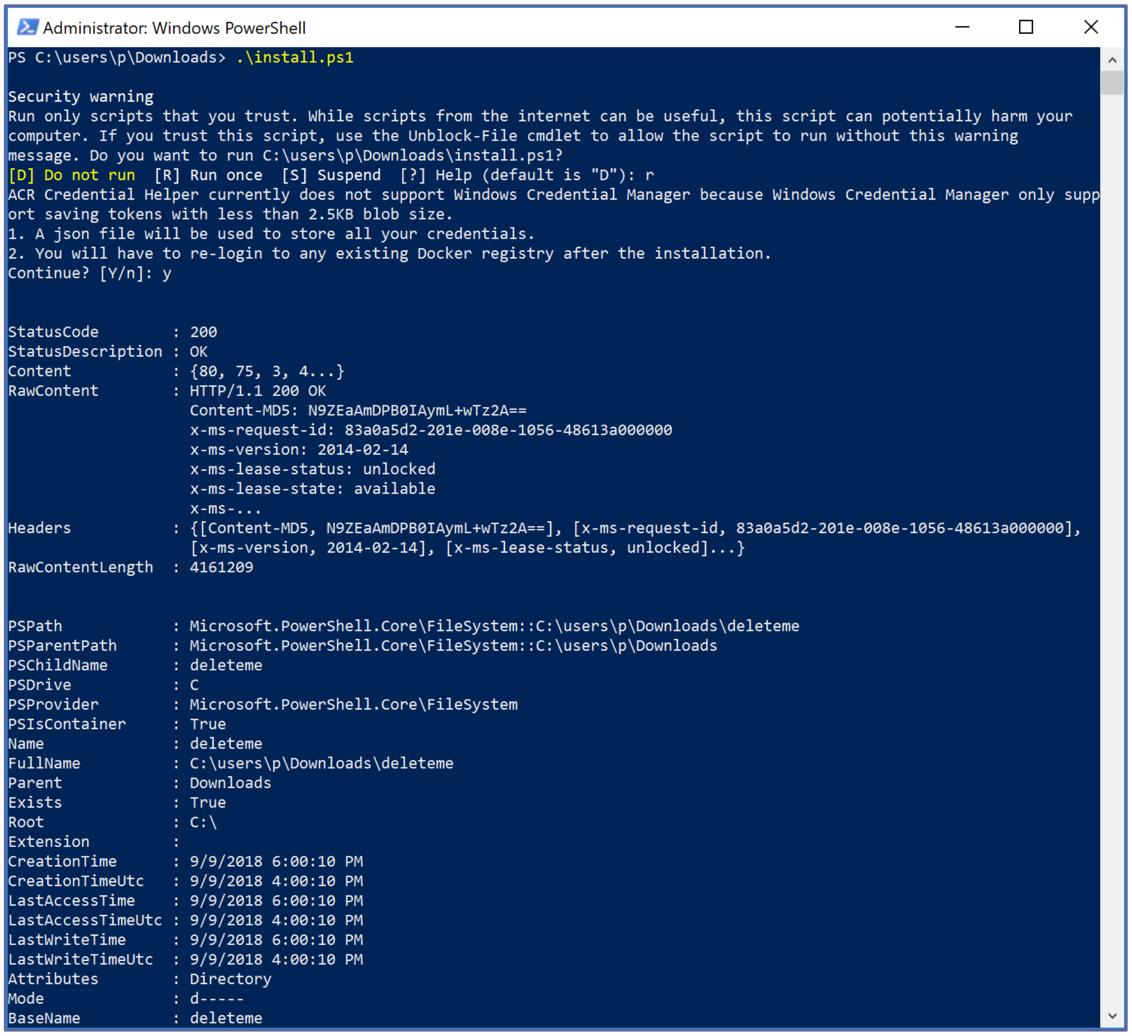
- Docker
- Azure CLI

Installation

For Windows, run the powershell installation script in administrator mode:

```
iex ([System.Text.Encoding]::UTF8.GetString((Invoke-WebRequest -Uri https://aka.ms/acr/installaad/win).Content))
```

Download the install.ps1 from the link (with the red arrow)



```
PS C:\users\p\Downloads> .\install.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be useful, this script can potentially harm your
computer. If you trust this script, use the Unblock-File cmdlet to allow the script to run without this warning
message. Do you want to run C:\users\p\Downloads\install.ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): r
ACR Credential Helper currently does not support Windows Credential Manager because Windows Credential Manager only supp
ort saving tokens with less than 2.5KB blob size.
1. A json file will be used to store all your credentials.
2. You will have to re-login to any existing Docker registry after the installation.
Continue? [Y/n]: y


StatusCode        : 200
StatusDescription : OK
Content           : {80, 75, 3, 4...}
RawContent        : HTTP/1.1 200 OK
                    Content-MD5: N9ZEaAmDPB0IAymL+wTz2A==
                    x-ms-request-id: 83a0a5d2-201e-008e-1056-48613a000000
                    x-ms-version: 2014-02-14
                    x-ms-lease-status: unlocked
                    x-ms-lease-state: available
                    x-ms-...
Headers           : {[Content-MD5, N9ZEaAmDPB0IAymL+wTz2A==], [x-ms-request-id, 83a0a5d2-201e-008e-1056-48613a000000],
                    [x-ms-version, 2014-02-14], [x-ms-lease-status, unlocked]...}
RawContentLength  : 4161209


PSPath            : Microsoft.PowerShell.Core\FileSystem::C:\users\p\Downloads\deleteme
PSParentPath      : Microsoft.PowerShell.Core\FileSystem::C:\users\p\Downloads
PSChildName       : deleteme
PSDrive           : C
PSProvider        : Microsoft.PowerShell.Core\FileSystem
PSIsContainer     : True
Name              : deleteme
FullName          : C:\users\p\Downloads\deleteme
Parent            : Downloads
Exists            : True
Root              : C:\
Extension         :
CreationTime      : 9/9/2018 6:00:10 PM
CreationTimeUtc   : 9/9/2018 4:00:10 PM
LastAccessTime    : 9/9/2018 6:00:10 PM
LastAccessTimeUtc : 9/9/2018 4:00:10 PM
LastWriteTime     : 9/9/2018 6:00:10 PM
LastWriteTimeUtc  : 9/9/2018 4:00:10 PM
Attributes        : Directory
Mode              : d-----
BaseName          : deleteme
```

22. And try to authenticate again to the Azure Container Registry.

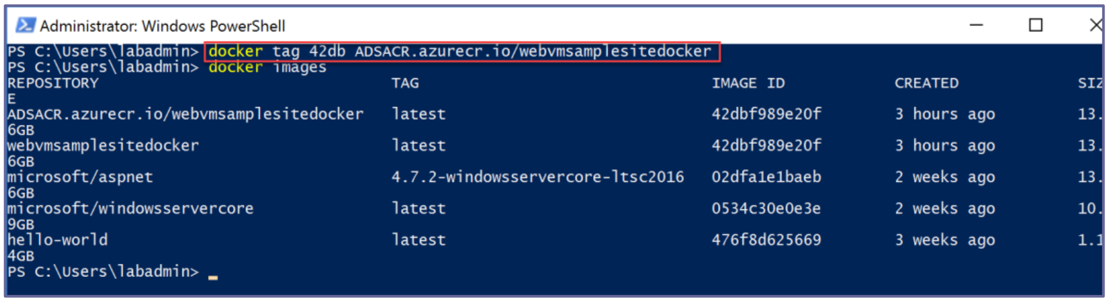
Task 2: Pushing a Docker image into an Azure Container Registry

1. As we now have connectivity towards the ACR, we can push our Docker image to it. There is however a dependency that the name of our Docker image has the name of the Azure Container Registry in it. So we first need to update the Docker image tag for our Docker image, by executing the following command:

   `docker images`   (To get the image ID number)

   `docker tag 42db [SUFFIX]ACR.azurecr.io/webvmsamplesitedocker`

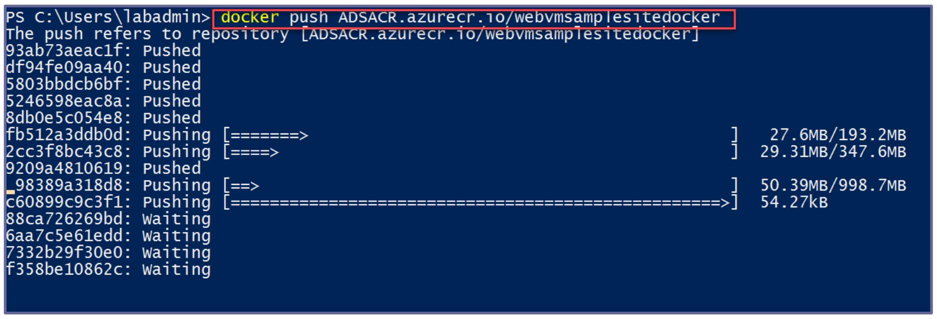
   `docker images`   (To validate the "new" image)



```
PS C:\Users\labadmin> docker tag 42db ADSACR.azurecr.io/webvmsamplesitedocker
PS C:\Users\labadmin> docker images
REPOSITORY                                   TAG                                IMAGE ID       CREATED       SIZ
E
ADSACR.azurecr.io/webvmsamplesitedocker      latest                             42dbf989e20f   3 hours ago   13.
6GB
webvmsamplesitedocker                        latest                             42dbf989e20f   3 hours ago   13.
6GB
microsoft/aspnet                             4.7.2-windowsservercore-ltsc2016   02dfa1e1baeb   2 weeks ago   13.
6GB
microsoft/windowsservercore                  latest                             0534c30e0e3e   2 weeks ago   10.
9GB
hello-world                                  latest                             476f8d625669   3 weeks ago   1.1
4GB
PS C:\Users\labadmin> _
```

2. Execute the following command to upload this image to the Azure Container Registry:

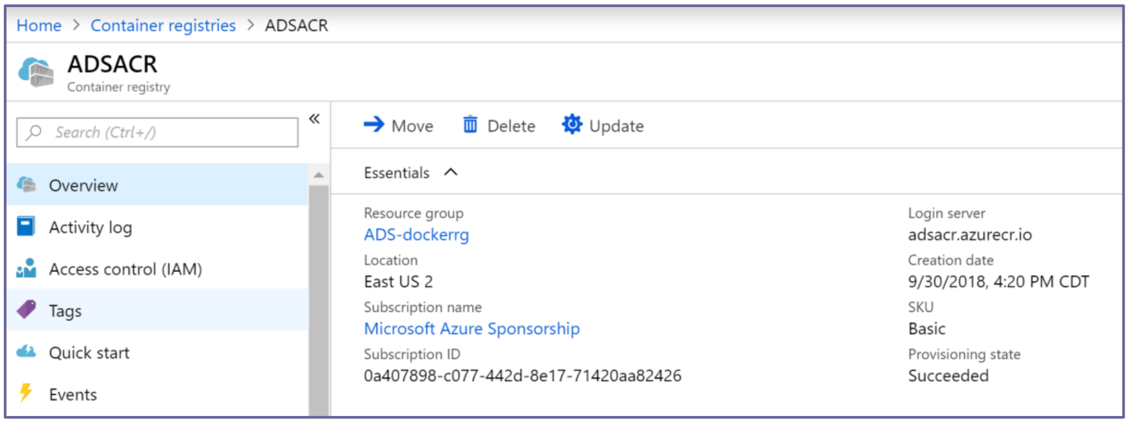
```
docker push [SUFFIX]ACR.azurecr.io/webvmsamplesitedocker
```



```
PS C:\Users\labadmin> docker push ADSACR.azurecr.io/webvmsamplesitedocker
The push refers to repository [ADSACR.azurecr.io/webvmsamplesitedocker]
93ab73aeac1f: Pushed
df94fe09aa40: Pushed
5803bbdcb6bf: Pushed
5246598eac8a: Pushed
8db0e5c054e8: Pushed
fb512a3ddb0d: Pushing [=======>                                            ]   27.6MB/193.2MB
2cc3f8bc43c8: Pushing [====>                                               ]  29.31MB/347.6MB
9209a4810619: Pushed
98389a318d8: Pushing [==>                                                  ]  50.39MB/998.7MB
c60899c9c3f1: Pushing [==================================================>]  54.27kB
88ca726269bd: Waiting
6aa7c5e61edd: Waiting
7332b29f30e0: Waiting
f358be10862c: Waiting
```

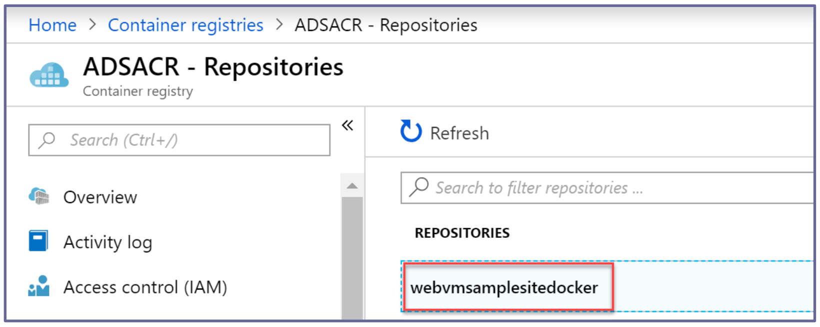
Note: Since this image is about 13GB in size, this initial upload process might take some time.

3. From the Azure Portal | All Services | Azure Container Registries | select the ACR you created earlier.

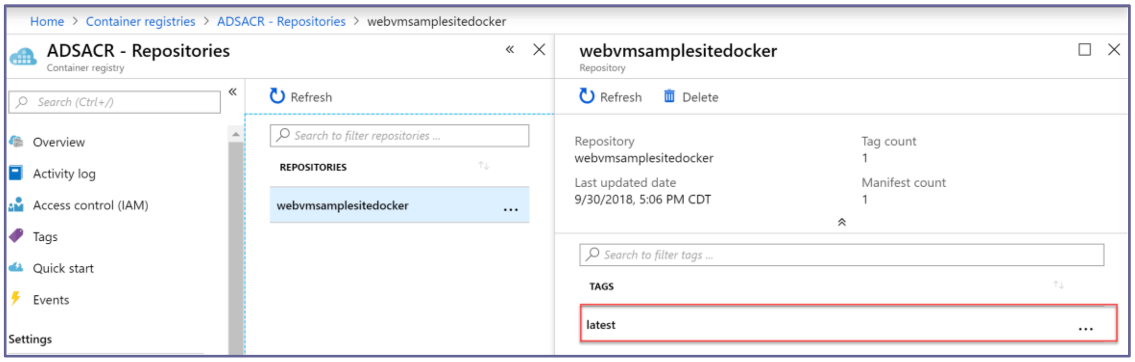


Resource group
ADS-dockerrg
Location
East US 2
Subscription name
Microsoft Azure Sponsorship
Subscription ID
0a407898-c077-442d-8e17-71420aa82426
Login server
adsacr.azurecr.io
Creation date
9/30/2018, 4:20 PM CDT
SKU
Basic
Provisioning state
Succeeded

4. In the blade menu to the left under the **Services** section, **click Repositories.** Notice the Docker image "webvmsamplesitedocker" is stored in here.

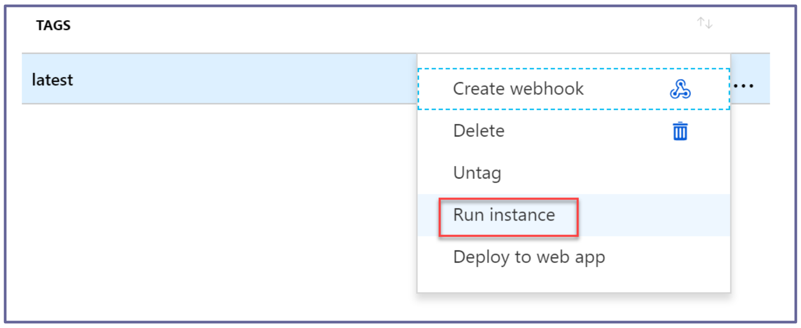


5. Click the webvmsamplesitedocker repository, which opens the specific details for this image, exposing its version:



Task 3: Running an Azure Container Instance from a Docker image in Azure Container Registry

1. Click the … next to latest, and choose Run Instance

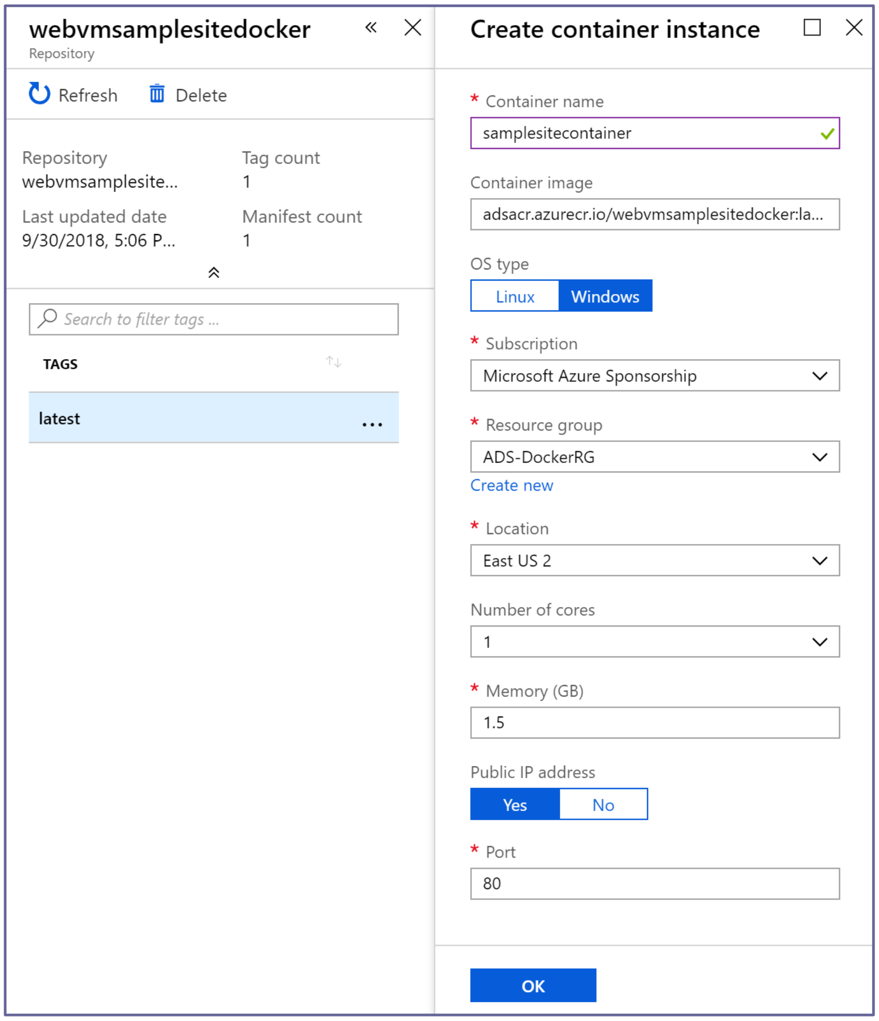


2. This opens the Create Container Instance blade. Complete the parameter fields using the following information:

- Container Name	samplesitecontainer
- OS-type	Windows

- Subscription	your Azure Subscription
- Resource Group	select [Suffix]-DockerRG as Resource Group
- Location	should be picked up from the Resource Group

Leave all other settings unchanged (1 core, 1,5GB memory, Public IP address YES and Port 80)



webvmsamplesitedocker
Repository

Refresh   Delete

Repository: webvmsamplesite...
Tag count: 1
Last updated date: 9/30/2018, 5:06 P...
Manifest count: 1

Search to filter tags ...

TAGS
latest

Create container instance

* Container name: samplesitecontainer

Container image: adsacr.azurecr.io/webvmsamplesitedocker:la...

OS type: Linux [✓ Windows]

* Subscription: Microsoft Azure Sponsorship

* Resource group: ADS-DockerRG
Create new

* Location: East US 2

Number of cores: 1

* Memory (GB): 1.5

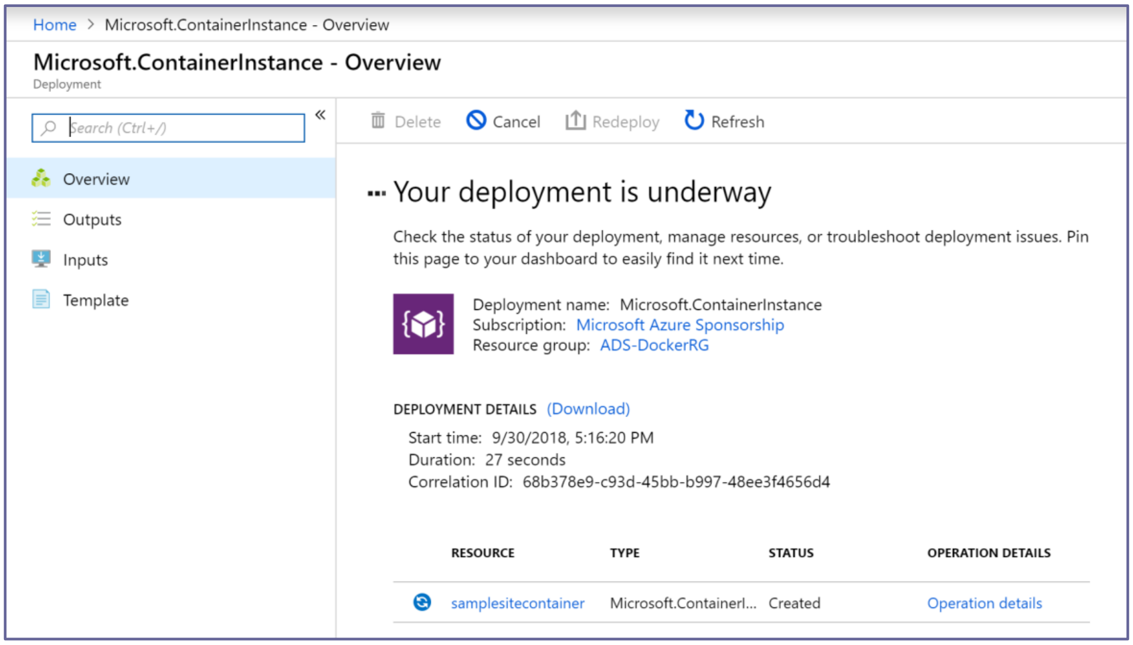
Public IP address: [✓ Yes] No

* Port: 80

OK

3. Press OK to have the Container Instance created.

4. You can follow the process from the notification area


Home > Microsoft.ContainerInstance - Overview

Microsoft.ContainerInstance - Overview

Deployment

Your deployment is underway

Check the status of your deployment, manage resources, or troubleshoot deployment issues. Pin this page to your dashboard to easily find it next time.

Deployment name: Microsoft.ContainerInstance
Subscription: Microsoft Azure Sponsorship
Resource group: ADS-DockerRG

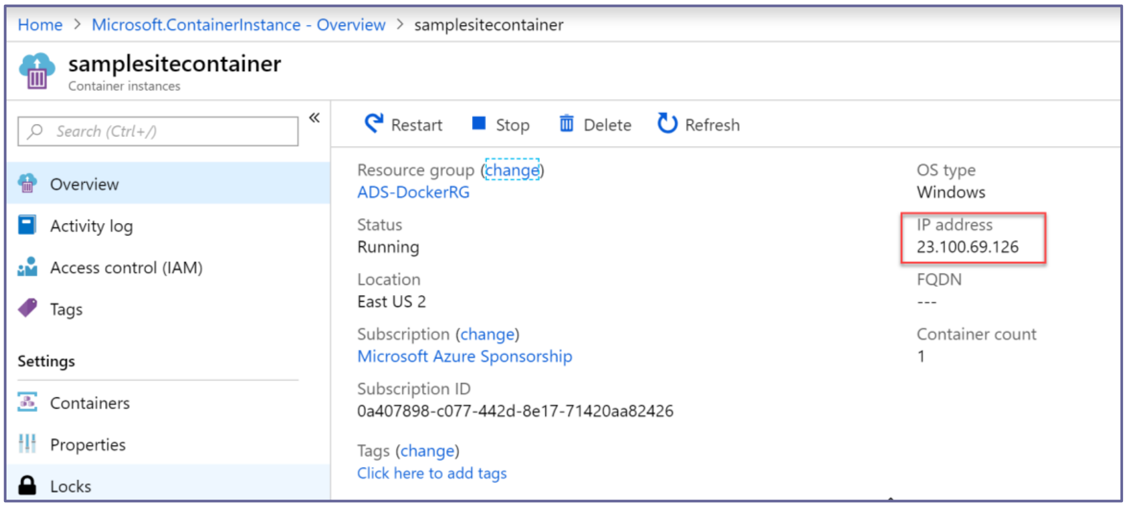
DEPLOYMENT DETAILS (Download)

Start time: 9/30/2018, 5:16:20 PM
Duration: 27 seconds
Correlation ID: 68b378e9-c93d-45bb-b997-48ee3f4656d4

RESOURCE	TYPE	STATUS	OPERATION DETAILS
samplesitecontainer	Microsoft.Containerl...	Created	Operation details

5. Wait for the deployment process to complete successfully.

6. Once the deployment is finished, **open the Azure Container Instance** in the portal (All Services | Container Instances), and **browse to the ACI "samplesitecontainer"** that just got created.


Home > Microsoft.ContainerInstance - Overview > samplesitecontainer

samplesitecontainer

Container instances

Resource group (change)
ADS-DockerRG

Status
Running

Location
East US 2

Subscription (change)
Microsoft Azure Sponsorship

Subscription ID
0a407898-c077-442d-8e17-71420aa82426

Tags (change)
Click here to add tags

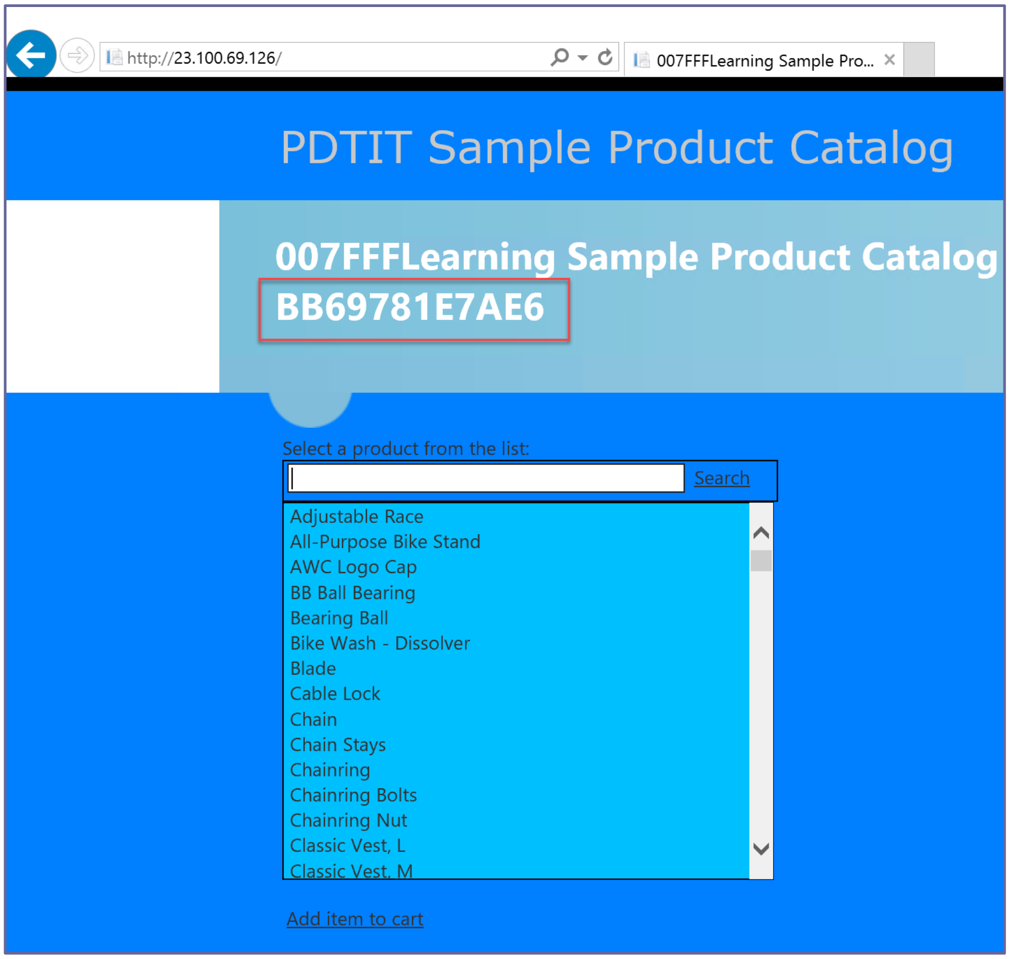
OS type
Windows

IP address
23.100.69.126

FQDN

Container count
1

7. **Copy the IP address** for this Azure Container Instance, or directly browse to it from your internet browser


http://23.100.69.126/

007FFFLearning Sample Pro...

PDTIT Sample Product Catalog

007FFFLearning Sample Product Catalog

BB69781E7AE6

Select a product from the list:

Search

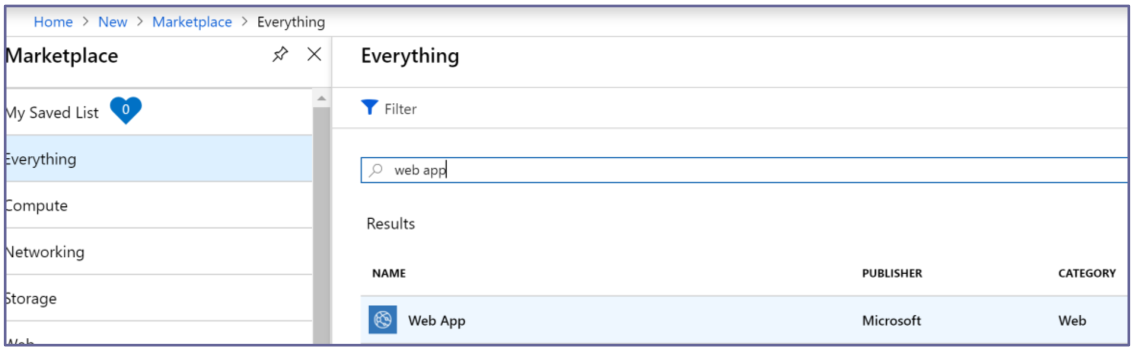
Adjustable Race
All-Purpose Bike Stand
AWC Logo Cap
BB Ball Bearing
Bearing Ball
Bike Wash - Dissolver
Blade
Cable Lock
Chain
Chain Stays
Chainring
Chainring Bolts
Chainring Nut
Classic Vest, L
Classic Vest, M

Add item to cart

8. The sample website starts successfully, and again has connectivity to the underlying SQL Azure database. Notice the name of the Azure Container Instance is visible too.

Task 4: Deploy a Container using Azure Web Apps

1. From the Azure Portal | Create New Resource | Web App.


Home > New > Marketplace > Everything

Marketplace

My Saved List 0
Everything
Compute
Networking
Storage

Everything

Filter

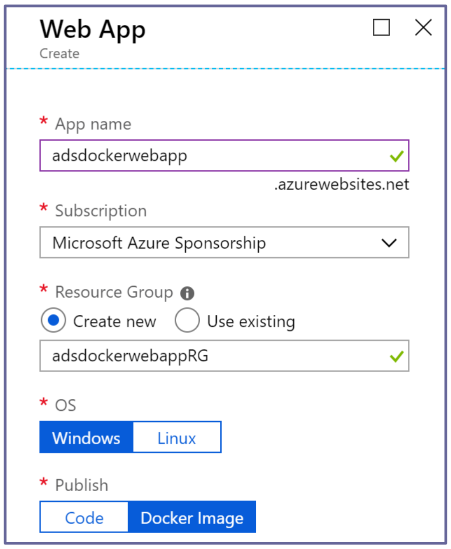
web app

Results

NAME	PUBLISHER	CATEGORY
Web App	Microsoft	Web

2. Press the Create button to open the Web App blade. Complete the required parameters as follows:

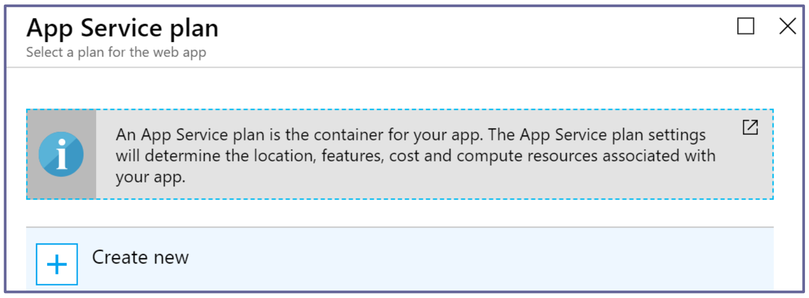
- App Name:	[SUFFIX]dockerwebapp.azurewebsites.net
- Resource Group:	Create New | [SUFFIX]dockerwebappRG
- OS:	Windows
- Publish:	Docker Image



Web App
Create

* App name: adsdockerwebapp .azurewebsites.net
* Subscription: Microsoft Azure Sponsorship
* Resource Group: [✓ Create new] Use existing
adsdockerwebappRG
* OS: [✓ Windows] Linux
* Publish: Code [✓ Docker Image]

3. For the Service Plan parameter, click Create New



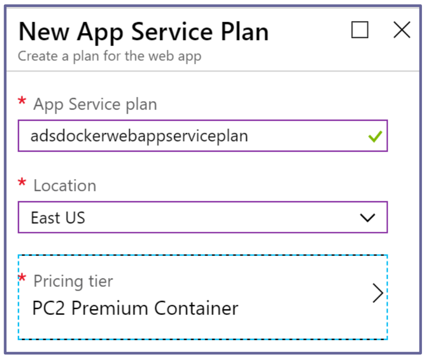
App Service plan
Select a plan for the web app

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

4. Complete the required parameters for the App Service Plan as follows:

- App Service Plan:	[SUFFIX]dockerwebappserviceplan
- Location:	East US
- Pricing Tier:	Select the PC2 Premium Container plan



New App Service Plan
Create a plan for the web app

* App Service plan: adsdockerwebappserviceplan

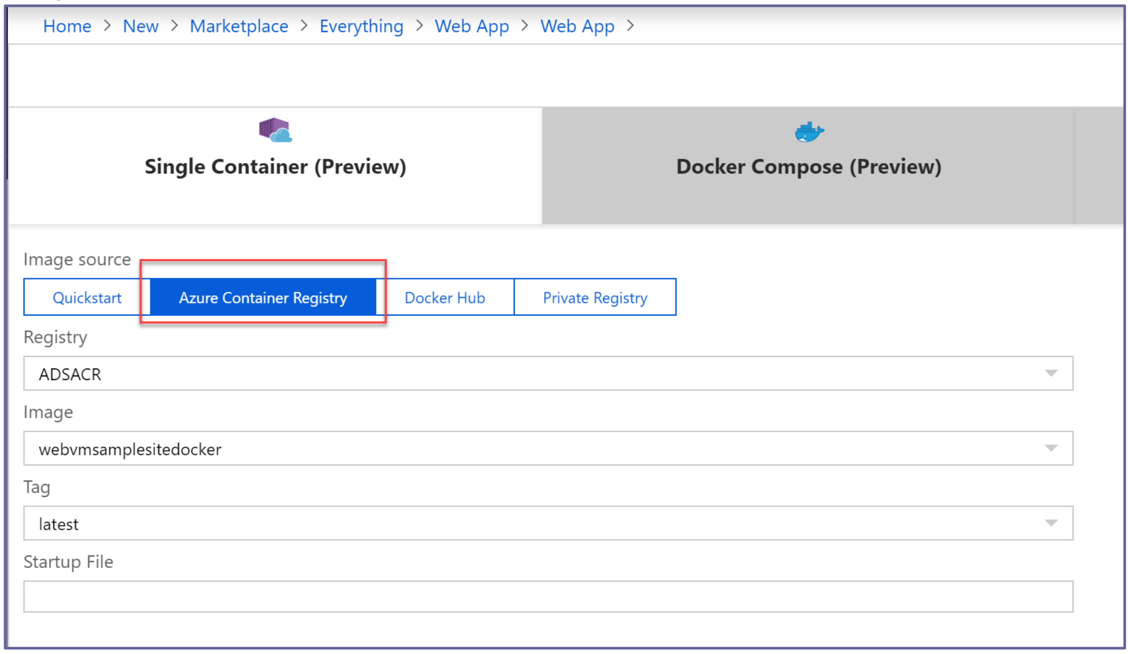
* Location: East US

* Pricing tier: PC2 Premium Container

And confirm the plan with **OK.**

5. **Completing the "Configure Container"** parameter opens the detailed blade, where you **make the following selections**:

- **Single Container (Preview)**	
- **Image Source**	Azure Container Registry
- **Registry**	[SUFFIX]ACR
- **image**	webvmsamplesitedocker
- **Tag**	latest



Home > New > Marketplace > Everything > Web App > Web App >
Single Container (Preview)
Docker Compose (Preview)
Quickstart
Azure Container Registry
Docker Hub
Private Registry

Image source: Azure Container Registry

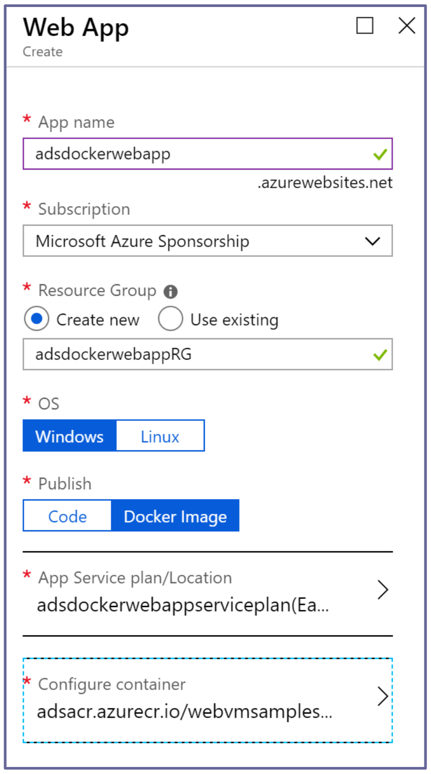
Registry: ADSACR

Image: webvmsamplesitedocker

Tag: latest

Startup File:

6. **Confirm the creation** by pressing the **Apply** button. Your container web app settings are now like this:


Web App
Create

* App name
adsdockerwebapp
.azurewebsites.net

* Subscription
Microsoft Azure Sponsorship

* Resource Group
Create new [✓] Use existing
adsdockerwebappRG

* OS
[✓ Windows] Linux

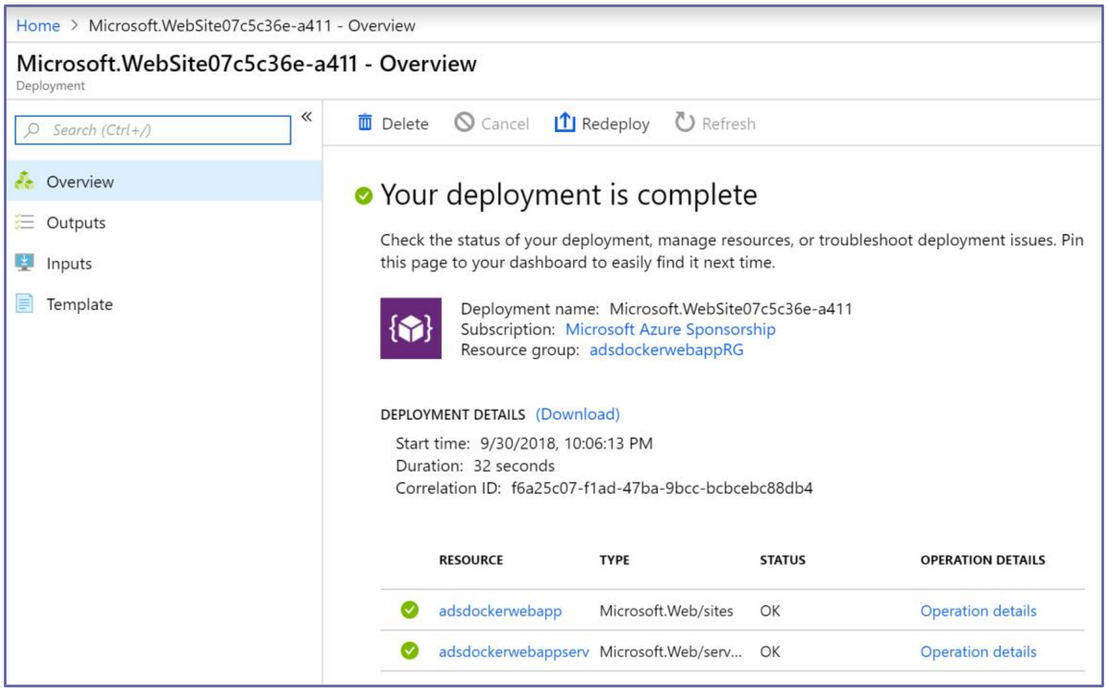
* Publish
Code [✓ Docker Image]

* App Service plan/Location
adsdockerwebappserviceplan(Ea...

* Configure container
adsacr.azurecr.io/webvmsamples...

7. **Press** the **Create** button to start the deployment of the Azure Web App for Containers.

8. **Follow-up** on the deployment from the notification area.



Microsoft.WebSite07c5c36e-a411 - Overview
Deployment

Your deployment is complete

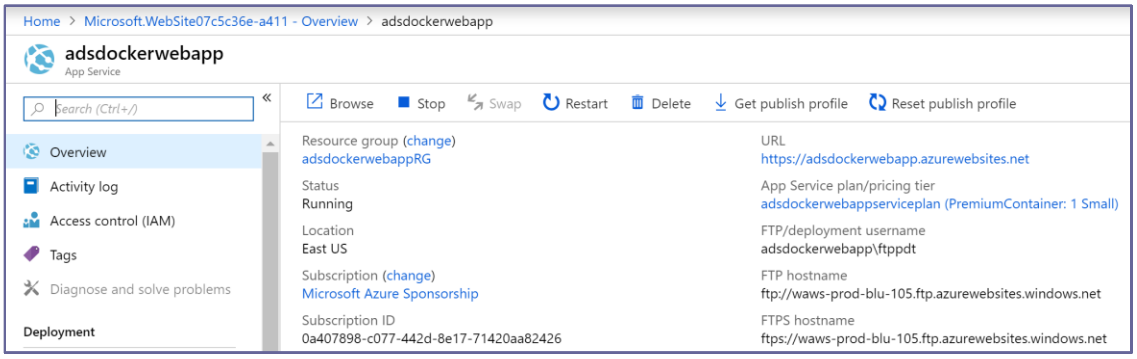
Check the status of your deployment, manage resources, or troubleshoot deployment issues. Pin this page to your dashboard to easily find it next time.

Deployment name: Microsoft.WebSite07c5c36e-a411
Subscription: Microsoft Azure Sponsorship
Resource group: adsdockerwebappRG

DEPLOYMENT DETAILS (Download)
Start time: 9/30/2018, 10:06:13 PM
Duration: 32 seconds
Correlation ID: f6a25c07-f1ad-47ba-9bcc-bcbcebc88db4

	RESOURCE	TYPE	STATUS	OPERATION DETAILS
✓	adsdockerwebapp	Microsoft.Web/sites	OK	Operation details
✓	adsdockerwebappserv	Microsoft.Web/serv...	OK	Operation details

9. **Once deployed**, browse to the **[SUFFIX]dockerwebapp Azure Resource**, which opens the detailed blade:



Resource group (change)
adsdockerwebappRG

Status
Running

Location
East US

Subscription (change)
Microsoft Azure Sponsorship

Subscription ID
0a407898-c077-442d-8e17-71420aa82426

URL
https://adsdockerwebapp.azurewebsites.net

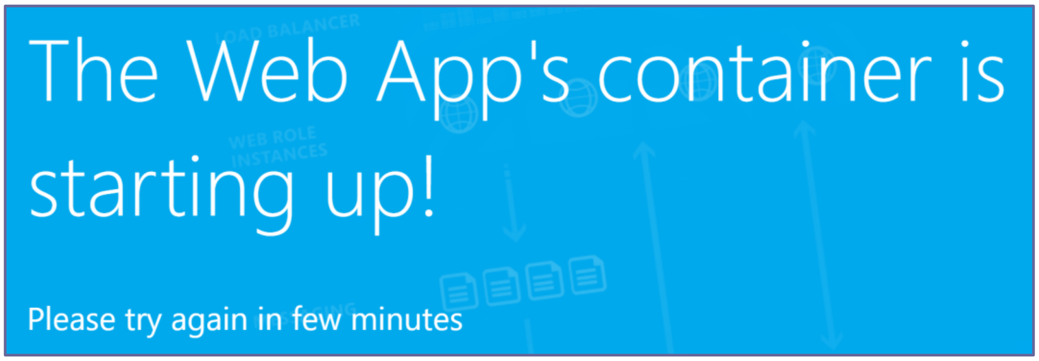
App Service plan/pricing tier
adsdockerwebappserviceplan (PremiumContainer: 1 Small)

FTP/deployment username
adsdockerwebapp\ftppdt

FTP hostname
ftp://waws-prod-blu-105.ftp.azurewebsites.windows.net

FTPS hostname
ftps://waws-prod-blu-105.ftp.azurewebsites.windows.net

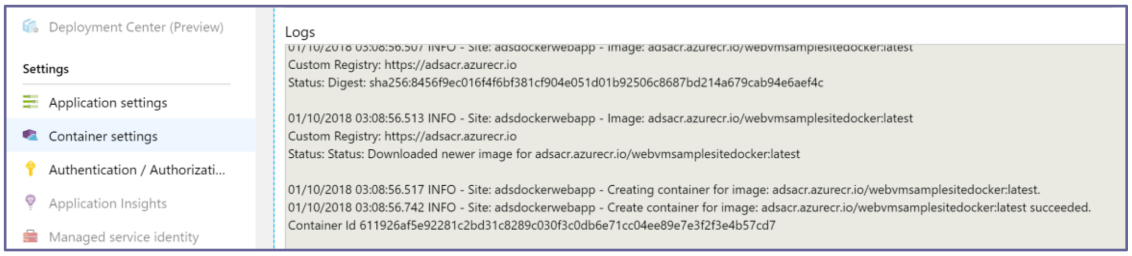
10. **Copy** the URL and paste it in your internet browser. Note the message about the container starting up:



The Web App's container is starting up!

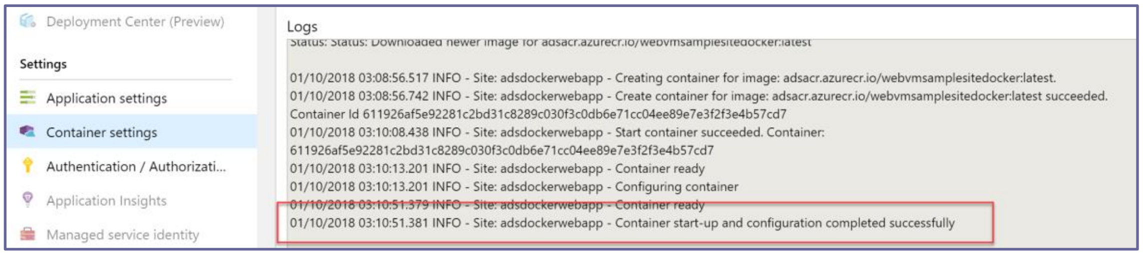
Please try again in few minutes

11. **Go back** to the Azure Portal, which still has your Azure Web App for Containers open; here, **browse** to **Settings | Container Settings** and **look at the LOGs section.** This shows the different steps undergoing to get the container running.



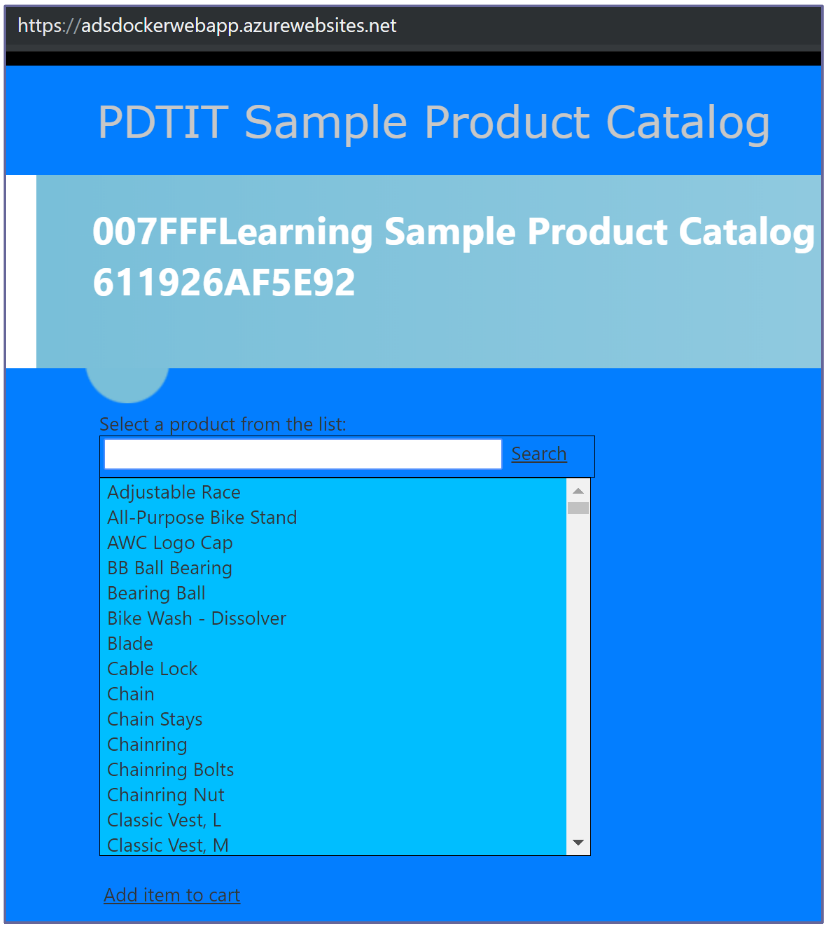
```
01/10/2018 03:08:56.507 INFO - Site: adsdockerwebapp - Image: adsacr.azurecr.io/webvmsamplesitedocker:latest
Custom Registry: https://adsacr.azurecr.io
Status: Digest: sha256:8456f9ec016f4f6bf381cf904e051d01b92506c8687bd214a679cab94e6aef4c

01/10/2018 03:08:56.513 INFO - Site: adsdockerwebapp - Image: adsacr.azurecr.io/webvmsamplesitedocker:latest
Custom Registry: https://adsacr.azurecr.io
Status: Status: Downloaded newer image for adsacr.azurecr.io/webvmsamplesitedocker:latest

01/10/2018 03:08:56.517 INFO - Site: adsdockerwebapp - Creating container for image: adsacr.azurecr.io/webvmsamplesitedocker:latest.
01/10/2018 03:08:56.742 INFO - Site: adsdockerwebapp - Create container for image: adsacr.azurecr.io/webvmsamplesitedocker:latest succeeded.
Container Id 611926af5e92281c2bd31c8289c030f3c0db6e71cc04ee89e7e3f2f3e4b57cd7
```

12. **Wait** for the Logs output mentioning the container is started and configuration completed successfully.



```
Status: Status: Downloaded newer image for adsacr.azurecr.io/webvmsamplesitedocker:latest

01/10/2018 03:08:56.517 INFO - Site: adsdockerwebapp - Creating container for image: adsacr.azurecr.io/webvmsamplesitedocker:latest.
01/10/2018 03:08:56.742 INFO - Site: adsdockerwebapp - Create container for image: adsacr.azurecr.io/webvmsamplesitedocker:latest succeeded.
Container Id 611926af5e92281c2bd31c8289c030f3c0db6e71cc04ee89e7e3f2f3e4b57cd7
01/10/2018 03:10:08.438 INFO - Site: adsdockerwebapp - Start container succeeded. Container:
611926af5e92281c2bd31c8289c030f3c0db6e71cc04ee89e7e3f2f3e4b57cd7
01/10/2018 03:10:13.201 INFO - Site: adsdockerwebapp - Container ready
01/10/2018 03:10:13.201 INFO - Site: adsdockerwebapp - Configuring container
01/10/2018 03:10:51.379 INFO - Site: adsdockerwebapp - Container ready
01/10/2018 03:10:51.381 INFO - Site: adsdockerwebapp - Container start-up and configuration completed successfully
```

13. If you **go back to your browser window** with the "container starting message" and **refresh it, it opens up the c**ontainerized web application as expected.


https://adsdockerwebapp.azurewebsites.net
PDTIT Sample Product Catalog
007FFFLearning Sample Product Catalog
611926AF5E92
Select a product from the list:
Search
Add item to cart

Adjustable Race
All-Purpose Bike Stand
AWC Logo Cap
BB Ball Bearing
Bearing Ball
Bike Wash - Dissolver
Blade
Cable Lock
Chain
Chain Stays
Chainring
Chainring Bolts
Chainring Nut
Classic Vest, L
Classic Vest, M

14. This completes this task.

Summary

In this lab, you created an Azure Container Registry in Azure, pushed a local Docker image to the Azure Container Registry, and learned how to run an Azure Container Instance from this Docker image. You also learned how to deploy an Azure Web App for Containers for running your Docker image as an Azure Web App.

Migrating a legacy ASP.NET 2-tier application to Azure using Container Services

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer
PDTIT and 007FFFLearning.com

@pdtit @007FFFLearning

Version: October 2018 – 2.0