



# Azure Developer Series

Migrating a dotnetcore 2-tier application to Azure,  
using different architectures and DevOps best practices

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer  
PDTIT and 007FFFLearning.com

@pdtit

@007FFFLearning

Version: Sept 2019 – 1.0

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2019 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

# Contents

Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices - Hands-On-Labs step-by-step .....	4
Abstract and Learning Objectives .....	5
Requirements .....	6
Naming Conventions:.....	6
Azure Subscription:.....	6
Other requirements: .....	6
Alternative Approach: .....	6
Final Remarks:.....	7
Lab 8: Managing and Monitoring Azure Container Services (ACS) and Azure Kubernetes Services (AKS); .....	8
What you will learn .....	8
Time Estimate .....	8
Task 1: Enabling container scalability in Azure Kubernetes Services (AKS) .....	8
Task 2: Monitoring Azure Kubernetes Services in Azure .....	10
Task 3: Using the Kubernetes Dashboard in Azure Kubernetes Services .....	13
Task 4: Managing Kubernetes from Visual Studio Code.....	17
Summary.....	22
Summary.....	Error! Bookmark not defined.

# Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices - Hands-On-Labs step-by-step

You are part of an organization that is running a dotnetcore e-commerce platform application, using Windows Server infrastructure on-premises today, comprising a WebVM running Windows Server 2012 R2 with Internet Information Server (IIS) and a 2<sup>nd</sup> SQLVM running Windows Server 2012 R2 and SQL Server 2014.

The business has approved a migration of this business-critical workload to Azure, and you are nominated as the cloud solution architect for this project. No decision has been made yet on what the final architecture should or will look like. Your first task is building a Proof-of-Concept in your Azure environment, to test out the different architectures possible:

- Infrastructure as a Service (IAAS)
- Platform as a Service (PAAS)
- Containers as a Service (CaaS)

At the same time, your CIO wants to make use of this project to switch from a more traditional mode of operations, with barriers between IT sysadmin teams and Developer teams, to a DevOps way of working. Therefore, you are tasked to explore Azure DevOps and determine where CI/CD Pipelines can assist in optimizing the deployment and running operations of this e-commerce platform, especially when deploying updates to the application.

As you are new to the continuous changes in Azure, you want to make sure this process goes as smooth as possible, starting from the assessment to migration to day-to-day operations.

## Abstract and Learning Objectives

This workshop enables anyone to learn, understand and build a Proof of Concept, in performing a multi-tiered .Net Core web application using Microsoft SQL Server database, platform migration to Azure public cloud, leveraging on different Azure Infrastructure as a Service, Azure Platform as a Service (PaaS) and Azure Container offerings like Azure Container Instance (ACI) and Azure Kubernetes Services (AKS).

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 learn about the importance of performing proper assessments, and what tools Microsoft offers to help in this migration preparation phase. Once the application has been deployed on Azure Virtual Machines, students learn about Microsoft SQL database migration to SQL Azure PaaS, as well as deploying and migrating web applications to Azure Web Apps.

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

In the last part of the workshop, students get introduced to Azure DevOps, the new Microsoft Application Lifecycle environment, helping in building a CI/CD Pipeline to publish workloads using the DevOps principals and concepts, showing the integration with the rest of the already touched on Azure services like Azure Web Apps and Azure Kubernetes Services (AKS), closing the workshop with a module on overall Azure monitoring and operations and what tools Azure has available to assist your IT teams in this challenge.

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 2019;
- Publishing a .NET Core e-commerce application to an Azure Web Virtual Machine and SQL DB Virtual Machine;
- Performing a proper assessment of the as-is Web and SQL infrastructure using Microsoft Assessment Tools;
- Migrating a SQL 2014 database to Azure SQL PaaS (Lift & Shift);
- Migrating a .NET Core web application to Azure Web Apps (Lift & Shift);
- Containerizing a .NET Core web application using Docker, and pushing to Azure Container Registry (ACR);
- Running Azure Container Instance (ACI) and WebApp for Containers;
- Deploy and run Azure Azure Kubernetes Services (AKS);

- Deploying Azure DevOps and building a CI/CD Pipeline for the subject e-commerce application;
- Managing and Monitoring Azure Kubernetes Services (AKS);

## 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

- In one of the Azure Container Services tasks, you are required to create an Azure AD Service Principal, which 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.
- 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.
- 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); this could also be Visual Studio 2019 community edition - latest version
- Docker for Windows (updated to latest version)
- Azure CLI 2.0 (updated to latest version)
- Kubernetes CLI (updated to latest version)
- SimplCommerce Open Source e-commerce platform example (<http://www.simplcommerce.com>)

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 7: Managing and Monitoring Azure Kubernetes Services (AKS);

### What you will learn

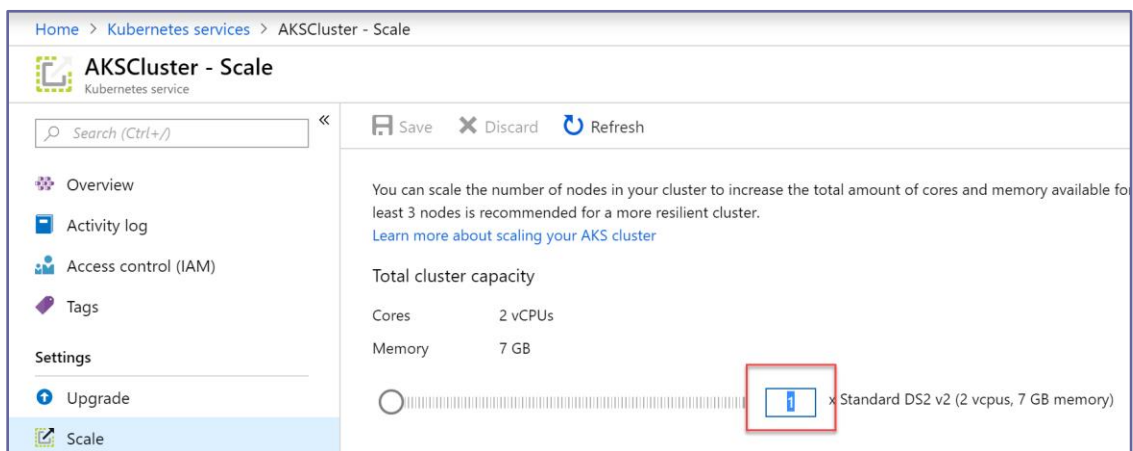
In this next lab of this workshop, we will focus on common operations, related to AKS. This includes enabling the basics of container scalability within the platform, as well as configuring the Azure built-in monitoring capabilities for these services, using Azure Monitor for ACS and Azure Application Insights for Kubernetes.

### Time Estimate

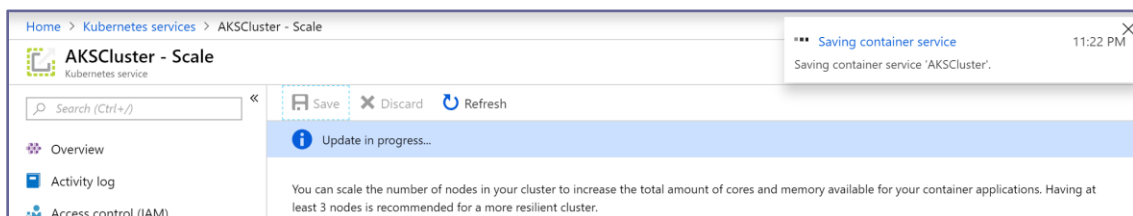
This lab shouldn't take longer than 60 minutes.

### Task 1: Enabling container scalability in Azure Kubernetes Services (AKS)

1. AKS provides some nice integration in the Azure Portal, for example on how to **scale out** your Kubernetes Service. **From the Azure Portal**, browse to your **Azure Kubernetes Service**. In the detailed blade, **go to settings | scale**



2. Change the single node configuration to 2, by updating the number of moving the bar.
3. Save the changes.





4. You can achieve the same by using the `kubect`l commandline syntax:

```
az aks scale --resource-group=[SUFFIX]AKSRG --name  
=[SUFFIX]AKSCluster --node-count 3
```

Note the command takes a couple of minutes to complete, without having impact on the already running pods. The result is published in the JSON output.

```
PS C:\DockerImage1> az aks scale --resource-group=NativeAKSRG --name=NativeAKSCluster --node-count 3
{
  "aadProfile": null,
  "addonProfiles": {
    "omsagent": {
      "config": {
        "logAnalyticsworkspaceResourceID": "/subscriptions/0a407898-c077-442d-8e17-71420aa82426/resource
sourcegroup-eus/providers/microsoft.operationalinsights/workspaces/defaultworkspace-0a407898-c077-442d-8
-eus"
      },
      "enabled": true
    }
  },
  "agentPoolProfiles": [
    {
      "count": 3,
      "maxPods": 110,
      "name": "nodepool1",
      "osDiskSizeGb": null,
      "osType": "Linux",
      "storageProfile": "ManagedDisks",
      "vmSize": "Standard_DS2_v2",
      "vnetSubnetId": null
    }
  ],
  "dnsPrefix": "NativeAKSC-NativeAKSRG-0a4078",
  "enableRBac": true,
  "fqdn": "nativeaksc-nativeaksg-0a4078-1e03fdff.hcp.eastus2.azmk8s.io",
  "id": "/subscriptions/0a407898-c077-442d-8e17-71420aa82426/resourcegroups/NativeAKSRG/providers/Micros
vice/managedclusters/NativeAKSCluster",
  "kubernetesVersion": "1.9.9",
  "linuxProfile": {
    "adminUsername": "azureuser",
    "ssh": {
      "publicKeys": [
        {
          "keyData": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDCB/2TX8SOL5o2yBjEgagcwQIvXPPO4Mt+u3NkF+6hCM
cbcRo3Ta5jSnLZ4VBFoxcf1kckfkbFPdItYia71iB9d0dE6617mWgK7XmvJqcdx+Xl5wGzj1RwwFegRiZv/Ah14PcRN1HyOBQAS1vwch
6ro8V1MtygLyxvRupyjgeb0KS+rP9wog4Fk6Dk9ojBZbz1XQAaHs0R4rgKVSZoyKkabjyDOOfNDDA1U6brr0mozbockKnG7mnHEZ4wpt
MgLGFIqd81koyz8Jem3AcsY2Ijm6NcAbtqPbFbaf"
        }
      ]
    }
  }
}
```

5. Another **"scale"** option, is not scaling the number of Kubernetes Nodes, but scaling the actual pods, using another update in the previously configured `simplaks.yml` file. This is done by running the following command:

```
kubectl scale --replicas=3 -f <path to yml file>
```

Which in this scenario spins up 3 instances of the Drupal container.

6. You can validate the creation using `kubectl get pods` and `kubectl get services --watch`

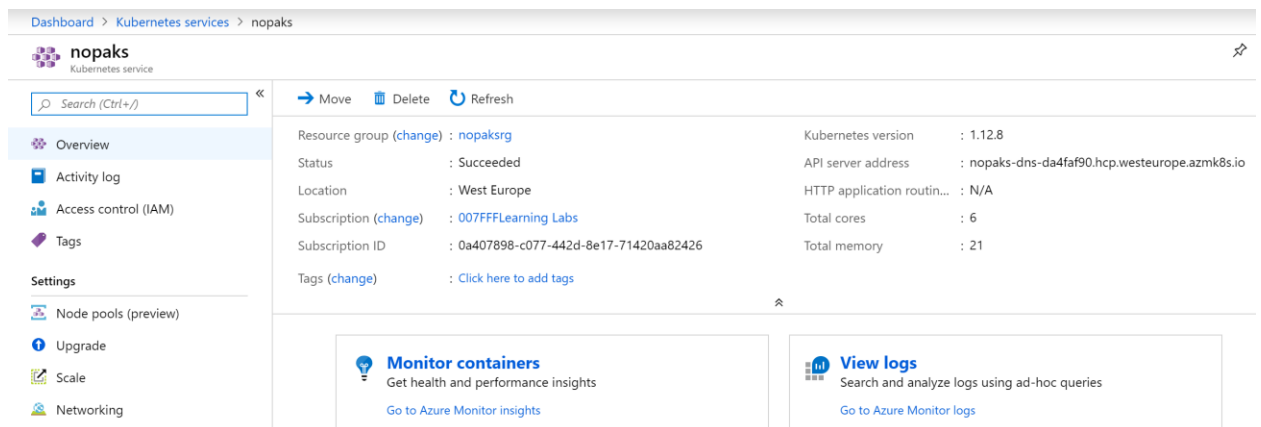
```
Administrator: Windows PowerShell
PS C:\Users\labadmin> cd \
PS C:\> cd .\DockerImage1\
PS C:\DockerImage1> kubectl scale --replicas=3 -f .\kubernetes3.yml
deployment.apps "drupalcntr" scaled
error: scaling the resource failed with: could not fetch the scale for services drupalcntr
e requested resource
PS C:\DockerImage1> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
adsakssample-6d7c8cf5cd-9brgr       0/1     ImagePullBackoff    0          22h
aksshellworld-64dbbb7cf8-vfgnc      1/1     Running          1          21h
dockerwebvmsample-79947845f6-jwr7p  0/1     ImagePullBackoff    0          22h
dockerwebvmsample2-77cd55c9bd-kkcjh  0/1     ImagePullBackoff    0          22h
drupalcntr-5fff4774bf-h8bx2         1/1     Running          0          20s
drupalcntr-5fff4774bf-hdm6g         1/1     Running          0          20s
drupalcntr-5fff4774bf-zm8lk         1/1     Running          0          50m
newadsakssample-6486f76985-p4r42    0/1     ImagePullBackoff    0          22h
newdockerwebvmsample-54dff974d-qpv4  0/1     ImagePullBackoff    0          22h
ubuntucont-6f555d84d8-xs7v1        0/1     CrashLoopBackoff    15         54m
PS C:\DockerImage1> kubectl get services --watch
NAME      TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
adsakssample  LoadBalancer  10.0.212.10  104.209.177.162  80:30156/TCP     22h
aksshellworld  LoadBalancer  10.0.164.32  137.116.72.252  80:31558/TCP     21h
drupalcntr    LoadBalancer  10.0.74.211  104.46.117.95   80:30750/TCP     50m
kubernetes    ClusterIP      10.0.0.1     <none>         443/TCP          22h
newadsakssample  LoadBalancer  10.0.56.37   104.209.180.231  80:32692/TCP     22h
ubuntucont     LoadBalancer  10.0.254.169 104.210.11.189  80:31412/TCP     55m
```

7. This completes the task on learning different scaling methods in AKS.

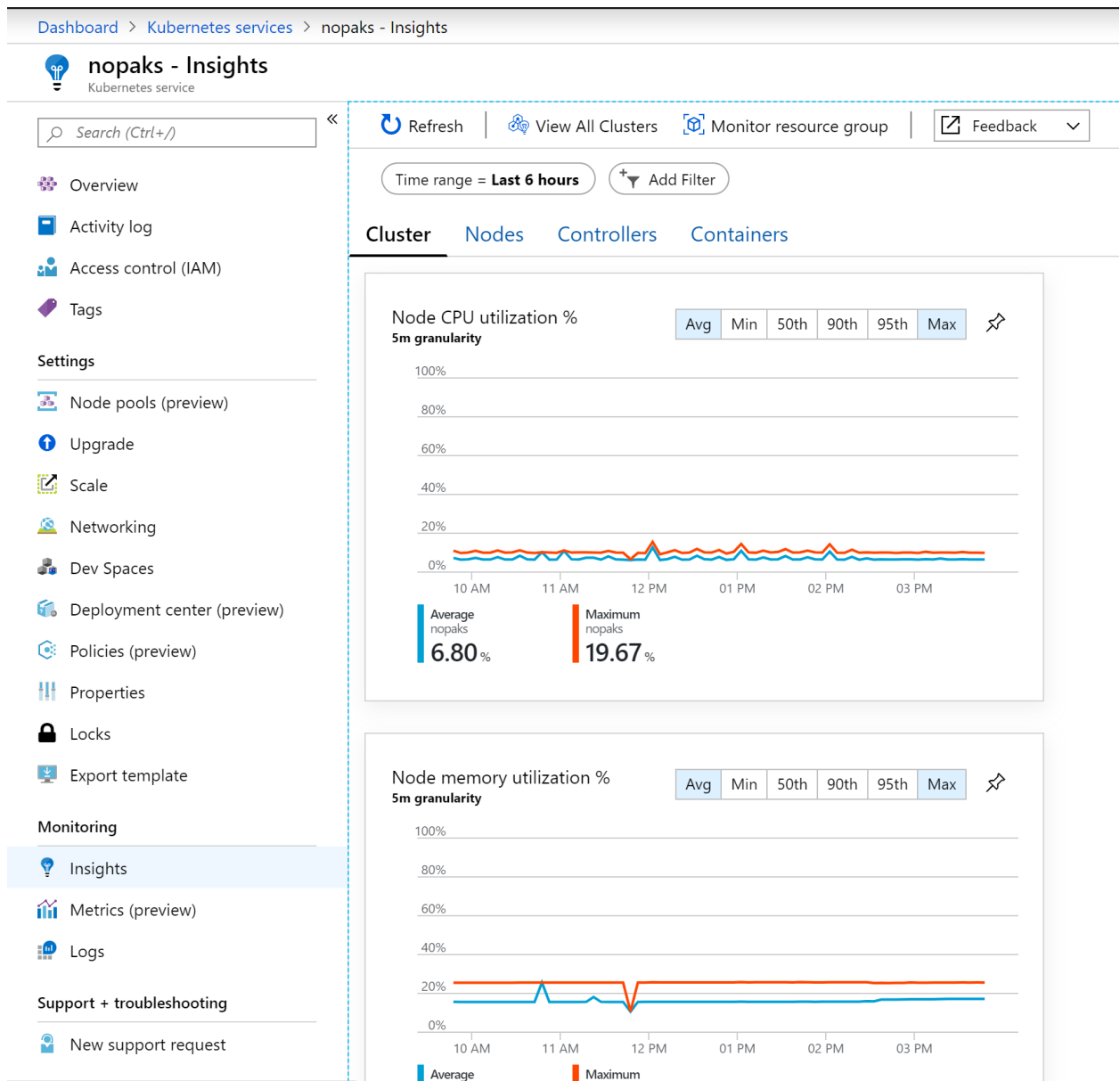
## Task 2: Monitoring Azure Kubernetes Services in Azure

Azure provides a nice integration (Insights) between standard Azure monitoring capabilities and the AKS services.

1. From the Azure Portal, browse to Azure Kubernetes Services, and select your AKS Service.



2. Selecting the AKS Cluster Service object, will open the detailed blade for this service. Here, select Monitoring Containers



3. This opens the Container monitoring blade details.
4. On top of the detailed blade, **Select Nodes**. This shows a more detailed view of the different AKS Nodes within that running cluster.

Dashboard > Kubernetes services > nopaks - Insights

**nopaks - Insights**  
Kubernetes service

Search (Ctrl+/)

Overview  
Activity log  
Access control (IAM)  
Tags

Settings  
Node pools (preview)  
Upgrade  
Scale  
Networking  
Dev Spaces  
Deployment center (preview)

Refresh | View All Clusters | Monitor resource group | Feedback

Time range = Last 6 hours | Add Filter | View Workbooks

Cluster Nodes Controllers Containers | Forums | Learn more

Search by name... Metric: CPU Usage (millicores) | Min Avg 50th 90th 95th Max

3 items

NAME	STATUS	95TH %	95TH	CONTAIN...	UPTIME	CONTROLLER	TREND 95TH % (1 BAR = 15M)
aks-agentp...	O...	11%	227 mc	12	21 ho...	-	
aks-agentp...	O...	6%	129 mc	7	21 ho...	-	
aks-agentp...	O...	6%	114 mc	8	21 ho...	-	

aks-agentp...  
Node  
View live data(preview)  
View in analytics

Node Name  
aks-agentpool-41570145-0  
Status  
Ready  
Cluster Name  
nopaks  
Kubelet Version  
v1.12.8

- Next, select **Controllers** in the top menu. This opens a more detailed view of the running AKS controllers. Highlight the **drupal controller**, and open its details. Here, you can nicely see the 3 scaled pods, with some details on performance for each, as well as uptime/running time.

Dashboard > Kubernetes services > nopaks - Insights

**nopaks - Insights**  
Kubernetes service

Search (Ctrl+/)

Overview  
Activity log  
Access control (IAM)  
Tags

Settings  
Node pools (preview)  
Upgrade  
Scale  
Networking  
Dev Spaces  
Deployment center (preview)  
Policies (preview)  
Properties  
Locks

Refresh | View All Clusters | Monitor resource group | Feedback

Time range = Last 6 hours | Add Filter | View Workbooks

Cluster Nodes Controllers Containers | Forums | Learn more

Search by name... Metric: CPU Usage (millicores) | Min Avg 50th 90th 95th Max

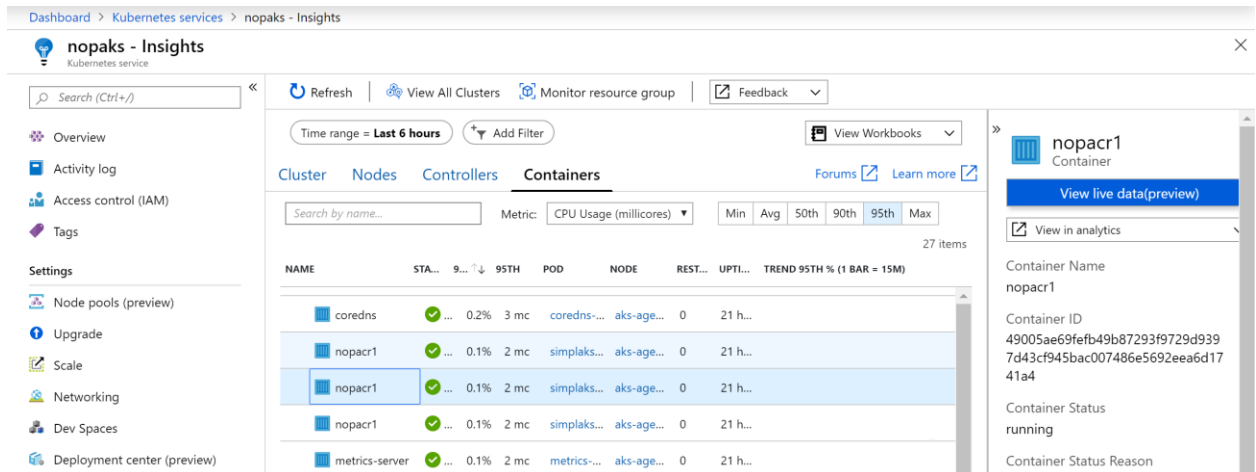
18 items

NAME	STATUS	95TH %	95TH	CONT...	RES...	UPT...	NODE	TREND 95TH % (1 BAR = 15M)
omsagent-rs-...	1	7%	11 mc	1	0	21 ...	-	
omsagent (Da...	3	5%	8 mc	3	0	21 ...	-	
tunnelfront-7...	1	5%	94 mc	1	0	21 ...	-	
traefik-597b7...	1	2%	4 mc	1	0	14 ...	-	
heapster-567...	1	2%	1 mc	2	0	21 ...	-	
kubernetes-d...	1	0.5%	0.5 ...	1	0	21 ...	-	
kube-proxy (D...	3	0.3%	5 mc	3	0	21 ...	-	

omsagent-rs-...  
Controller  
View live data(preview)  
View in analytics

Controller Name  
omsagent-rs-557d6ff466  
Namespace  
kube-system  
Controller Kind  
ReplicaSet  
Pod Count  
1  
Container Count  
1  
Service Name  
-

- Lastly, select **Containers**, which will provide a more detailed view on the actual running containers (and PODs).



for instance, we can see the NOPACR1 (the Azure Container Registry I'm using), offering 3 POD instances of the simplaks container as we defined earlier using the Kubernetes Yaml file.

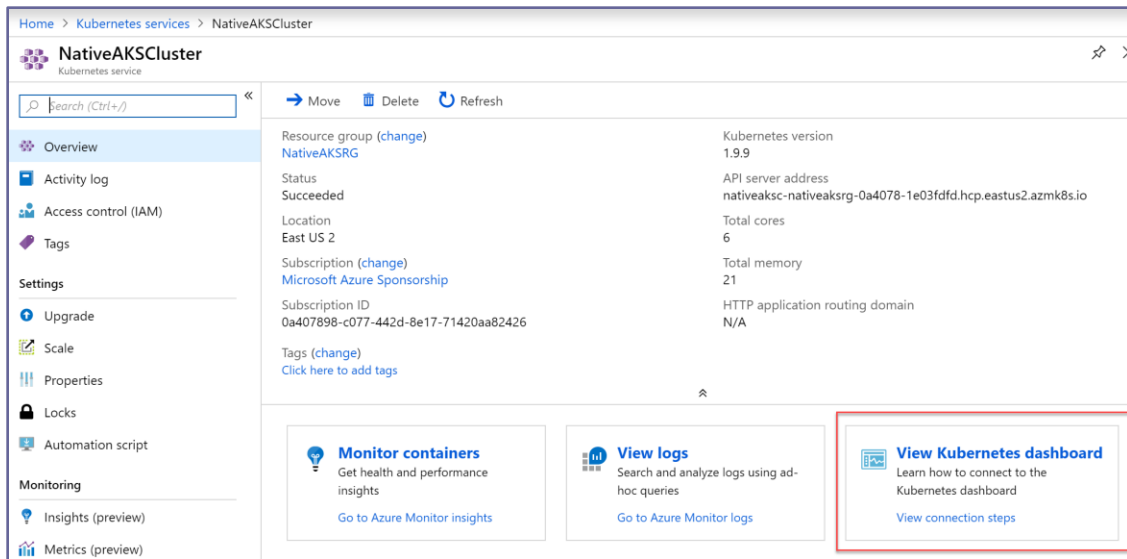
7. This concludes this part of the task.

### Task 3: Using the Kubernetes Dashboard in Azure Kubernetes Services

1. If you are familiar with other Azure Monitoring tools (like Azure Monitor, Application Insights,...) you will notice how hard Microsoft has been trying to provide a similar monitoring experience for you.

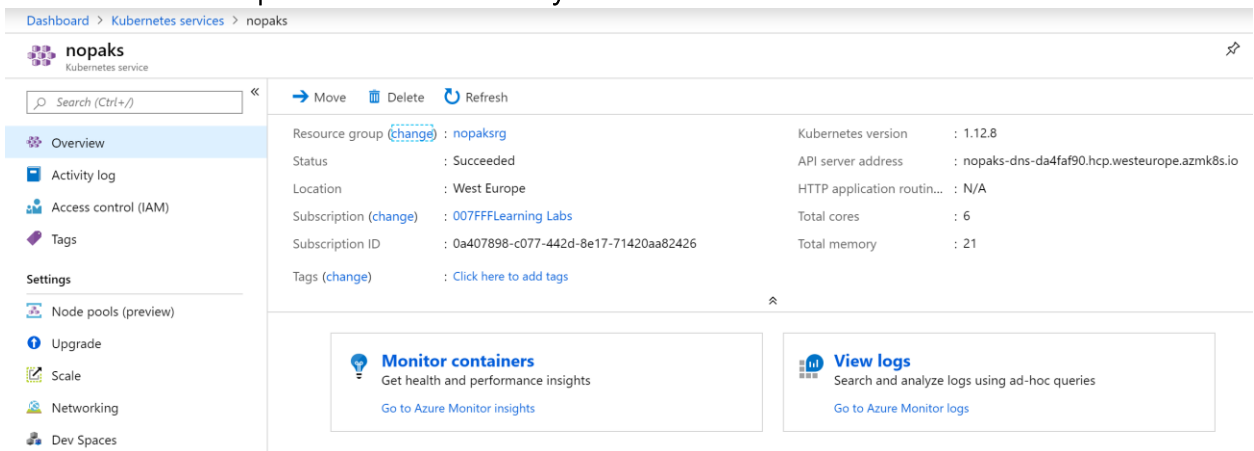
This is well excepted in the market, but for customers who run Kubernetes in a multi-cloud (public and hybrid) strategy, will most probably rely on the core built-in Kubernetes dashboard, instead of using the specific AKS-flavored one.

For a long time, the Azure Portal had an option to start this Kubernetes dashboard, but it is not showing up in the portal anymore.



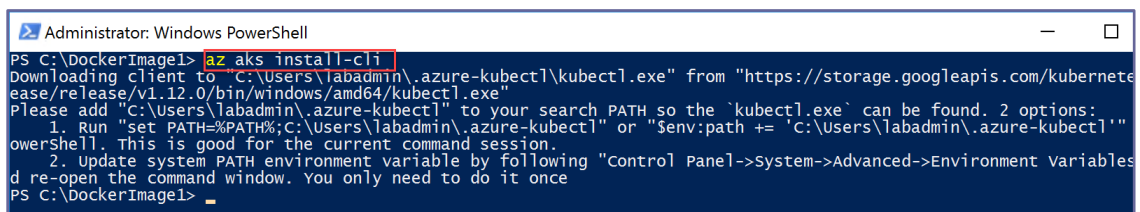
this is what the AKS portal blade showed until 2 months ago...

This is what the AKS portal blade shows today



2. The good news is however, that you can still use the built-in Kubernetes dashboard, by using AZ AKS BROWSE.
3. Let's run the following commands one by one, in PowerShell on our lab-jumpVM:

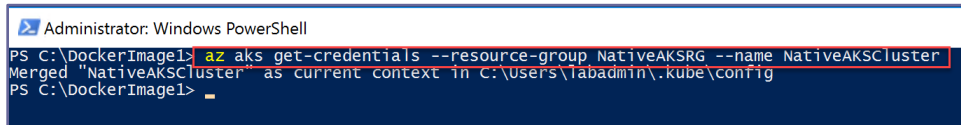
```
az aks install-cli
```



Which makes sure we have the latest version of the Azure CLI for managing Kubernetes installed.

- Next, run the following command to authenticate against the AKS Cluster:

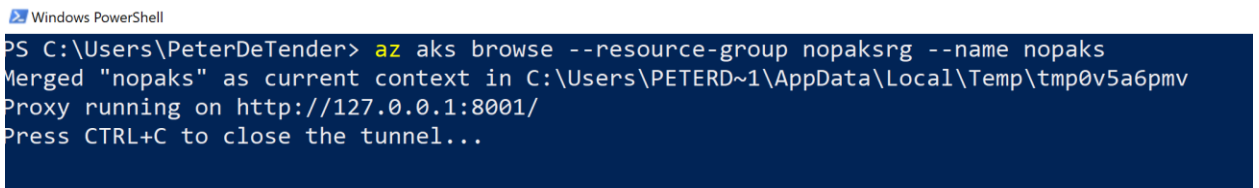
```
az aks get-credentials --resource-group [SUFFIX]AKSRG --name [SUFFIX]AKSCluster
```



```
Administrator: Windows PowerShell
PS C:\DockerImage1> az aks get-credentials --resource-group NativeAKSRG --name NativeAKSCluster
Merged "NativeAKSCluster" as current context in C:\Users\labadmin\.kube\config
PS C:\DockerImage1>
```

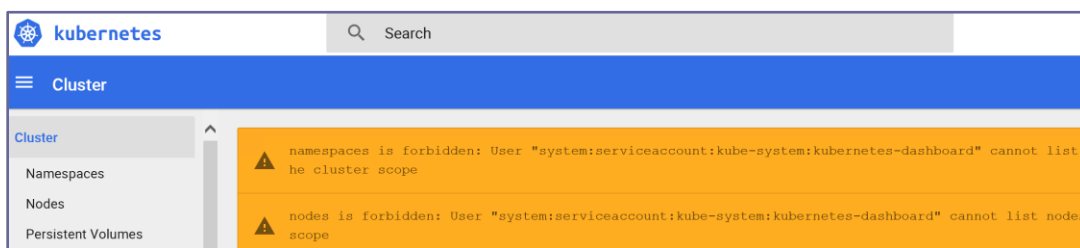
- Lastly, run the AZ AKS Browse command, which will call the Kubernetes dashboard in the browser:

```
az aks browse --resource-group [SUFFIX]AKSRG --name [SUFFIX]AKSCluster
```



```
Windows PowerShell
PS C:\Users\PeterDeTender> az aks browse --resource-group nopaksrg --name nopaks
Merged "nopaks" as current context in C:\Users\PETERD~1\AppData\Local\Temp\tmp0v5a6pmv
Proxy running on http://127.0.0.1:8001/
Press CTRL+C to close the tunnel...
```

- This opens your internet browser, and shows the Kubernetes Dashboard.



- Notice the error messages, saying your serviceaccount cannot list the cluster scope or nodes.
- This is related to a known "issue" / feature 😊, related to the fact our **Azure Kubernetes Service** is managed by RBAC. We need to tell the Kubernetes dashboard built-in service account to "trust/allow" RBAC, by **setting a clusterrolebinding**, using the following command:

```
kubectl create clusterrolebinding kubernetes-dashboard --clusterrole=cluster-admin --serviceaccount=kube-
```

system:kubernetes-dashboard

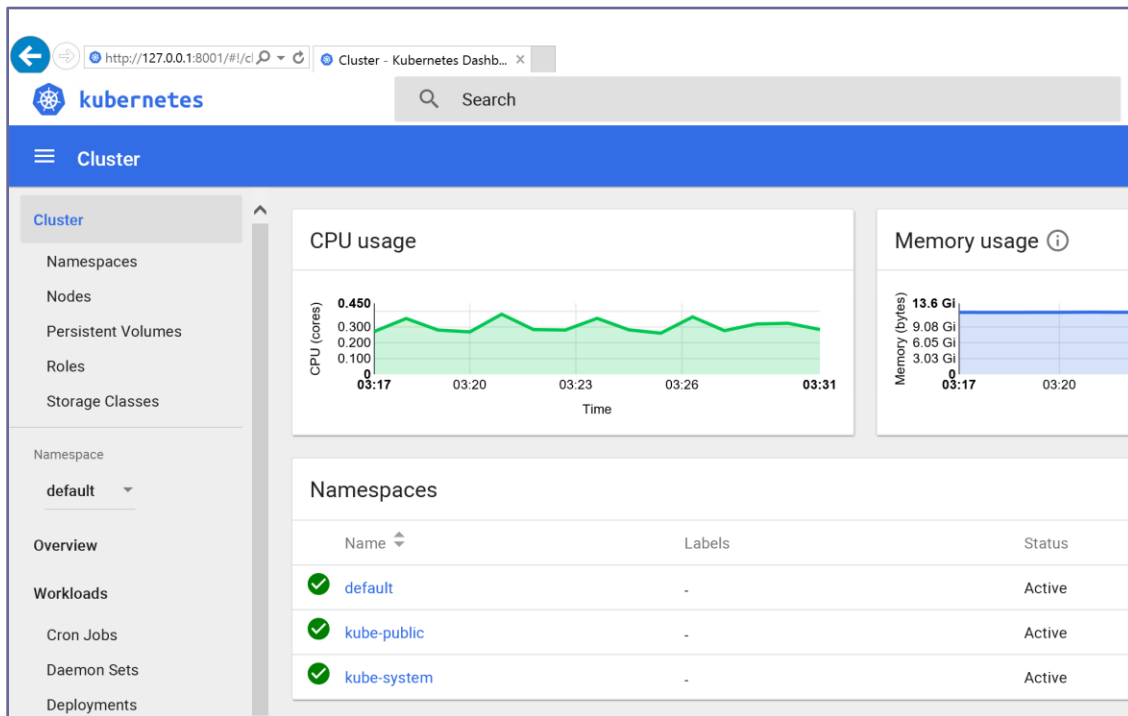
```
Administrator: Windows PowerShell
PS C:\DockerImage1> kubectl create clusterrolebinding kubernetes-dashboard --clusterrole=cluster-admin --serviceaccount=kube-system:kubernetes-dashboard
clusterrolebinding.rbac.authorization.k8s.io "kubernetes-dashboard" created
PS C:\DockerImage1>
```

9. If we then run our "az aks browse..." command again:

```
az aks browse --resource-group [SUFFIX]AKSRG --name
[SUFFIX]AKSCluster
```

```
Administrator: Windows PowerShell
PS C:\DockerImage1> az aks browse --resource-group NativeAKSRG --name NativeAKSCluster
Merged "NativeAKSCluster" as current context in C:\Users\labadmin\AppData\Local\Temp\tmp166y_hoh
Proxy running on http://127.0.0.1:8001/
Press CTRL+C to close the tunnel...
Forwarding from 127.0.0.1:8001 -> 9090
Forwarding from [::1]:8001 -> 9090
Handling connection for 8001
```

The Kubernetes dashboard will load successfully now:



10. From here, you can click around and get equally detailed views on the AKS Cluster, click-through on Nodes, Pods,... and more detailed status information on each of these resources.



Namespace							
default							
Overview							
Workloads							
Cron Jobs							
Daemon Sets							
Deployments							
Jobs							
Pods							
Replica Sets							

✓	drupalcntr-5fff4774bf-h8b:	aks-nodepool1-20062427-0	Running	0	43 minutes	▲	0
✓	drupalcntr-5fff4774bf-hdr:	aks-nodepool1-20062427-0	Running	0	43 minutes	▲	0
✓	drupalcntr-5fff4774bf-zm8	aks-nodepool1-20062427-0	Running	0	an hour	▲	0
!	ubuntucont-6f555d84d8-x:	aks-nodepool1-20062427-0	Waiting: CrashLoop	24	an hour	-	
	Back-off restarting failed container						
✓	akshelloworld-64dbbb7cf8	aks-nodepool1-20062427-0	Running	1	22 hours		0
!	newadsakssample-6486f7	aks-nodepool1-20062427-0	Waiting: ImagePullE	0	23 hours	-	
	Error: ImagePullBackOff						
		aks-nodepool1-					

Note: the failed deployments from my testing when building this lab guide are also included, containing a nice descriptive reason for the failed status of each pod.

11. This completes this part of the task.

## Task 4: Managing Kubernetes from Visual Studio Code

Besides the Azure built-in monitoring tools in the previous task, or the provided “Kubernetes-specific” dashboard, one can also manage the AKS cluster using Visual Studio Code.

1. If Visual Studio Code is not installed on your machine yet, run the install from [code.visualstudio.com](https://code.visualstudio.com)

https://code.visualstudio.com

Visual Studio Code Docs Updates

Version 1.33

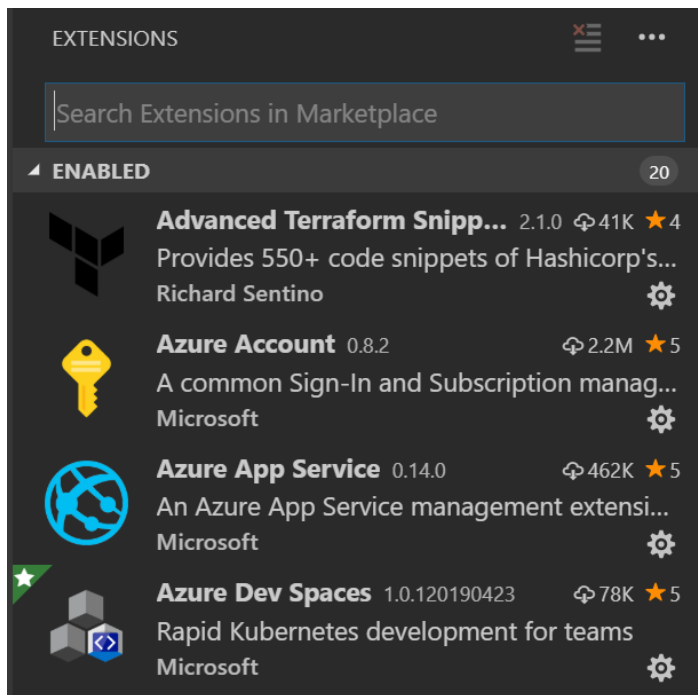
# Code editing. Redefined.

Free. Built on open source. Runs everywhere.

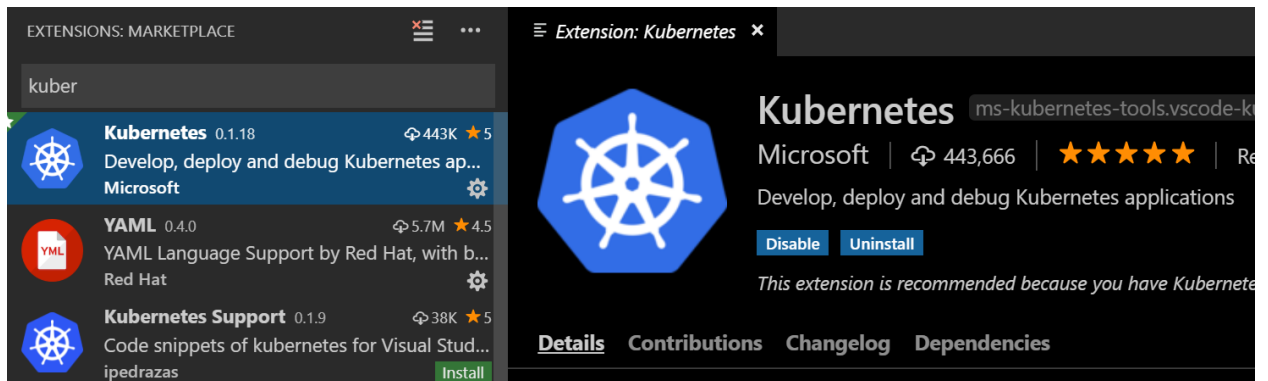
**Download for Windows**  
Stable Build

		Stable	Insiders
<b>macOS</b>	Package	↓	↓
<b>Windows x64</b>	User Installer	↓	↓
<b>Linux x64</b>	.deb	↓	↓
	.rpm	↓	↓
<a href="#">Other downloads</a>			

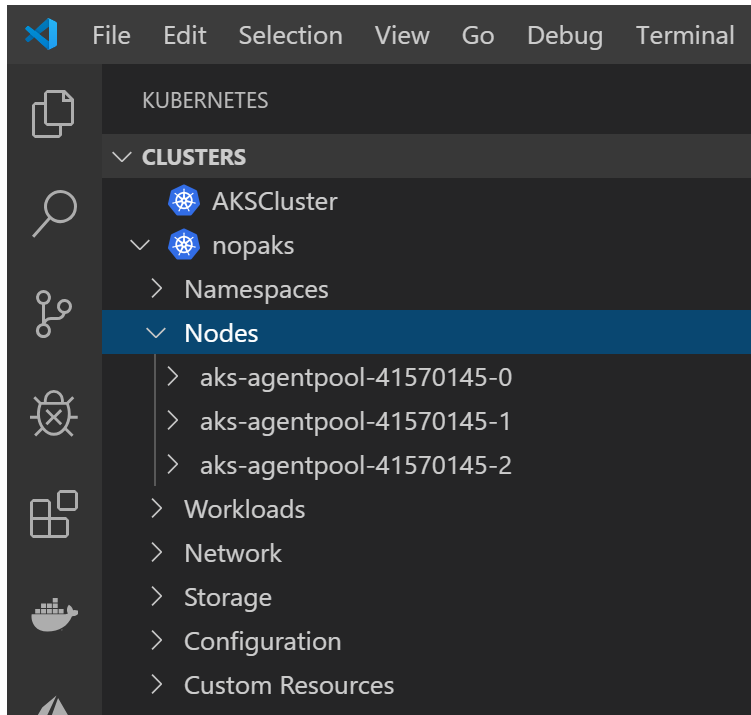
2. Once Visual Studio is installed, from the menu, go to **File / Preferences / Extensions**. This shows a list of community and 3<sup>rd</sup> party vendor provided extensions.



3. In the Search Extensions in MarketPlace, type "Kubernetes".



4. Click **Install** and wait for the extension to get installed successfully. You will see a shortcut to it in the left menu sidebar. **Click on it.** Out of your Azure subscription ID and Azure admin account credentials, it will list all Kubernetes clusters it recognizes.

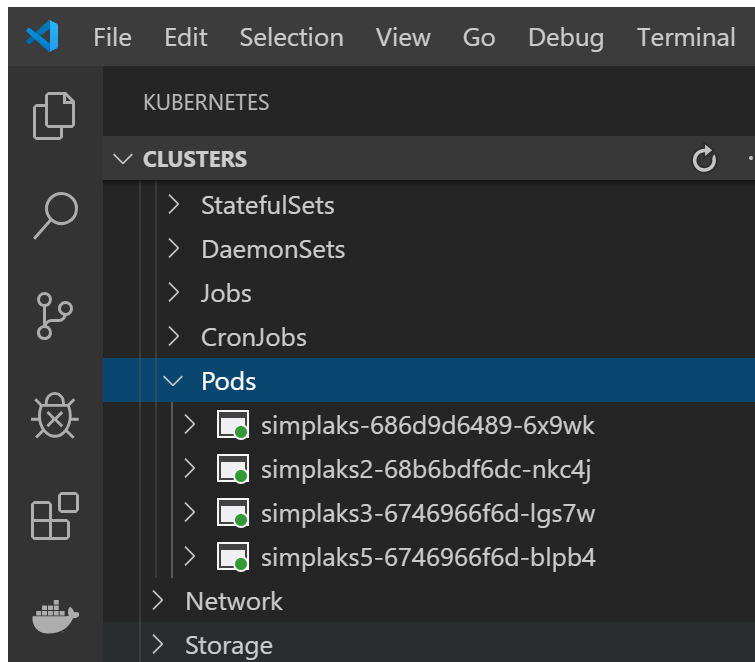


5. Note: if no AKS cluster is showing up here, Open PowerShell or Azure CLI, and run the following commands:

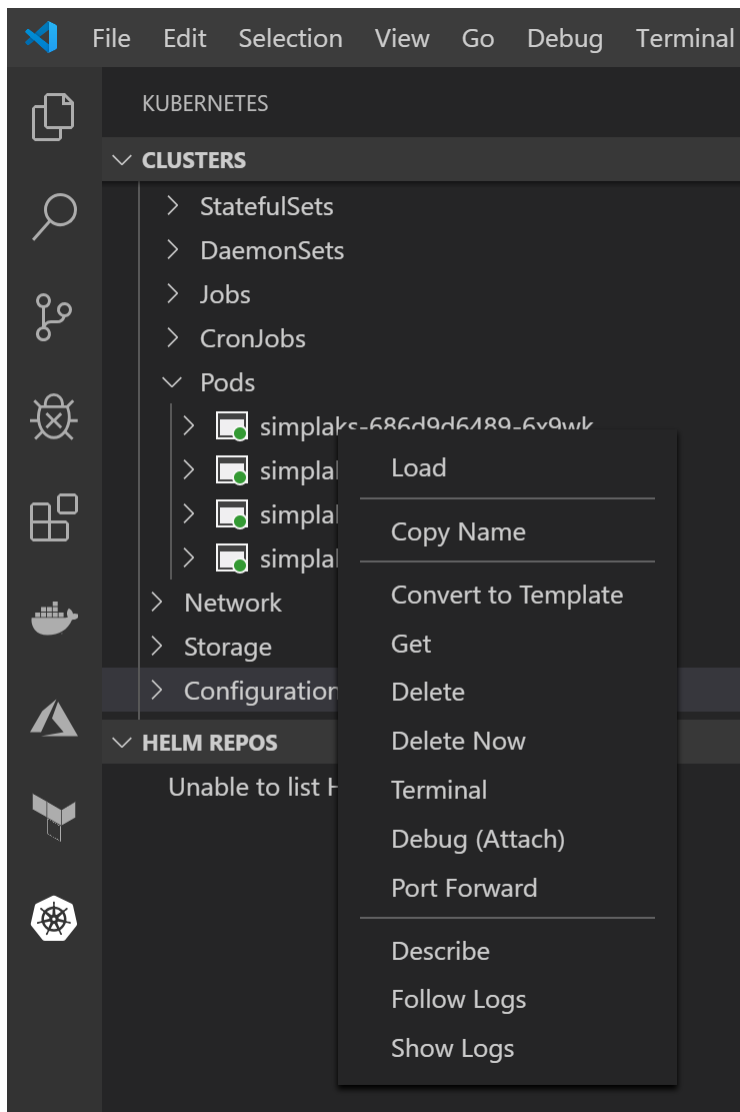
- az account login (this authenticates your session with Azure)
- az aks get-credentials --resource-group <your AKS RG> --name <your AKS Cluster Name>

```
PS C:\DockerImage1> az aks get-credentials --resource-group devopspdt813-rg --name devopspdt813
Merged "devopspdt813" as current context in C:\Users\PeterDeTender\.kube\config
```

6. **Refresh** the Visual Studio Code window, by select Kubernetes again. This time, your AKS Cluster should show up fine.
7. **You can browse** through the different core Kubernetes cluster components, like Nodes, Storage, Configuration and more.



8. And even perform some management tasks for the items, by right-clicking on one:



9. This completes the lab exercise.

## Summary

In this lab, you learned the basic admin tasks about scaling Azure Kubernetes Services, using both the Azure Portal and Kubectl command line. Next, you became familiar with the Azure Monitor capabilities of Kubernetes monitoring, as well how the built-in standard Kubernetes dashboard can be used besides the Azure monitoring capabilities. Last, you deployed the Kubernetes extension in Visual Studio Code, and performed some basic operations against the AKS cluster from there.