

NashTech Training

## Hands on: Practical DevOps for DEV

SD2350 – Hồ Phước Trúc

21 August 2023

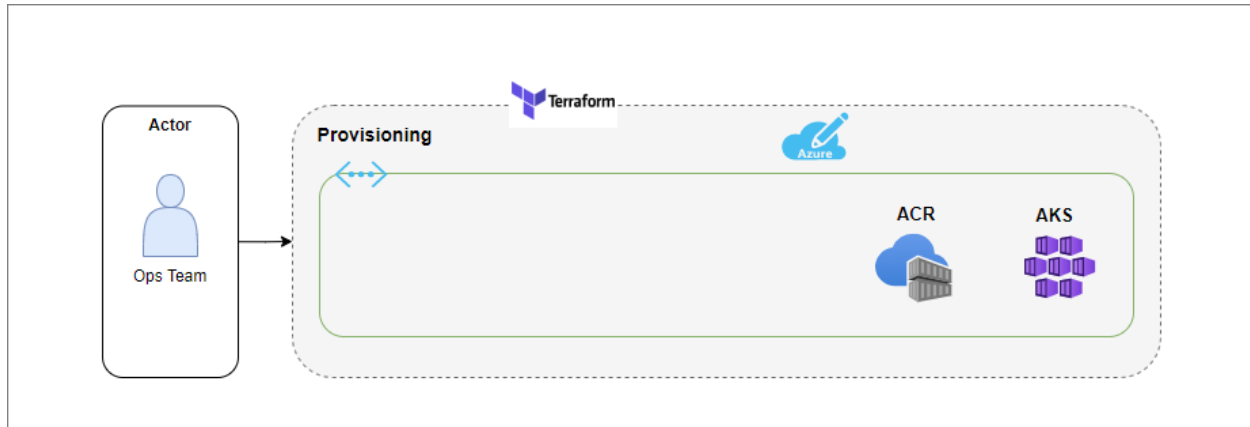
1	<b>Infrastructure as Code (IaC):</b> Provision Azure resources with Terraform .....	2
1.1	Authenticate using the Azure CLI .....	2
1.2	Prepare IaC project folders and variables .....	3
1.3	Script to create Virtual Network (VNET) and subnets using in the AKS cluster .....	4
1.4	Script to create Azure Container Registry (ACR) .....	5
1.5	Script to create Azure Kubernetes Service (AKS) .....	5
1.6	Run Terraform scripts .....	6
1.7	Verify resources created on Azure .....	7
2	<b>CI:</b> Setting up an azure pipeline, build and push Docker image to ACR .....	9
2.1	Setting up Azure Docker Registry service connection .....	10
2.2	Create new a CI pipeline and connect to MSA GitHub repository.....	11
2.3	Run CI build pipeline and verify results .....	13
3	<b>CD:</b> Setting up an azure pipeline and deploying applications on AKS .....	14
3.1	Create a Kubernetes namespace using for deployment .....	15
3.2	Setting up AKS service connection .....	16
3.3	Create new a CD pipeline and connect to Infrastructure GitHub repository.....	16
3.4	Run CD build pipeline and verify results.....	18
4	<b>GitOps:</b> Replace azure pipeline CD by Argo CD.....	19
4.1	Install Argo CD on AKS.....	20
4.2	Add new applications to Argo CD.....	22
4.3	Test deployment with Argo CD.....	24
5	<b>Monitoring:</b> Setup Prometheus and Grafana to monitor AKS resources .....	26
5.1	Create a namespace for monitoring .....	26
5.2	Install Prometheus and Grafana tools .....	27
5.3	Verify dashboard monitoring .....	28

# 1 Infrastructure as Code (IaC): Provision Azure resources with Terraform

Using Terraform to provision resources include VNET, ACR and AKS on Azure cloud.

GitHub repository: [https://github.com/hptruc/sd2350\\_azure\\_infrastructure](https://github.com/hptruc/sd2350_azure_infrastructure)

Architecture overview:

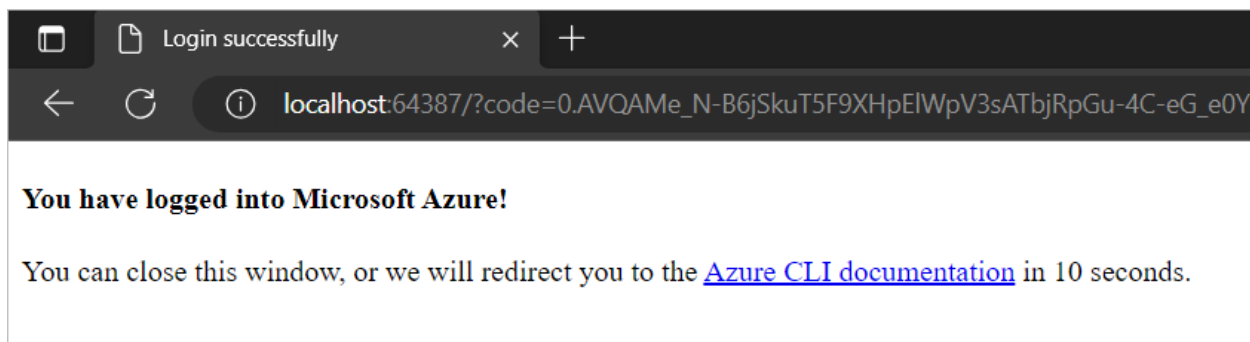


## 1.1 Authenticate using the Azure CLI

- In your terminal, use the Azure CLI tool to setup your account permissions locally.

```
Windows PowerShell
PS C:\Users\truchop> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue
the login in the web browser. If no web browser is available or if the web browser fails to open, use device code flow w
ith 'az login --use-device-code'.
```

- Your browser will open and prompt you to enter your Azure login credentials.
- After successful authentication, your terminal will display your subscription information.



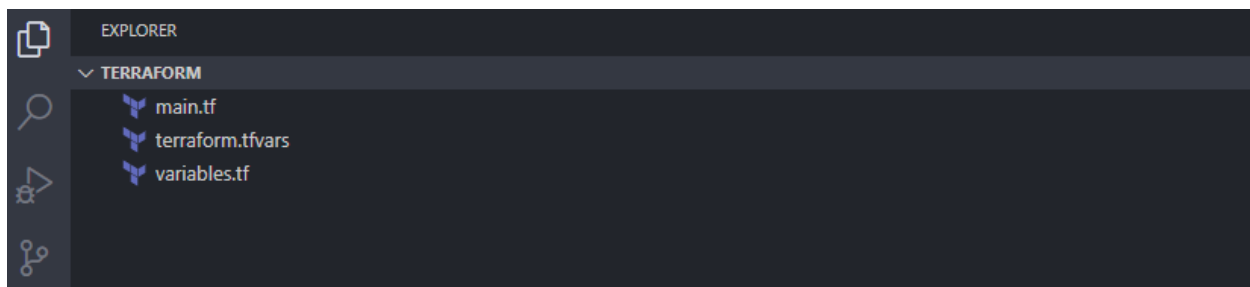
```
Windows PowerShell
PS C:\Users\truchop> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue
the login in the web browser. If no web browser is available or if the web browser fails to open, use device code flow w
ith 'az login --use-device-code'.
The following tenants don't contain accessible subscriptions. Use 'az login --allow-no-subscriptions' to have tenant lev
el access.
[REDACTED] 'Default Directory'
The following tenants require Multi-Factor Authentication (MFA). Use 'az login --tenant TENANT_ID' to explicitly login t
o a tenant.
[REDACTED] 'Default Directory'
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "[REDACTED]",
    "id": "[REDACTED]",
    "isDefault": false,
    "managedByTenants": [],
    "name": "Visual Studio Enterprise",
    "state": "Enabled",
    "tenantId": "[REDACTED]",
    "user": {
      "name": "TruchoPhuoc@[REDACTED].com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "[REDACTED]",
    "id": "[REDACTED]",
    "isDefault": true,
```

- Once you have chosen the account subscription ID, set the account with the Azure CLI.

```
Windows PowerShell
PS C:\Users\truchop> az account set --subscription "c7130152-[REDACTED]"
```

## 1.2 Prepare IaC project folders and variables

- Create a folder to store IaC source code, e.g. Terraform
- In this folder, create 3 files as below



- In **main.tf** file, add terraform script to configure the Azure provider (Azure resource manager – Azure RM)

```

main.tf > ...
1  # Configure the Azure provider
2  terraform {
3      required_providers {
4          azurerm = {
5              source = "hashicorp/azurerm"
6              version = "~> 3.0.2"
7          }
8      }
9
10     required_version = ">= 1.1.0"
11 }
12
13 provider "azurerm" {
14     features {}
15 }

```

- In **variables.tfvars** file, define terraform variables as below

```

terraform.tfvars > ...
1  resource_group_name = "practice-devops-resource"
2  location             = "southeastasia"
3
4  vnet_name            = "aks-vnet"
5  vnet_address_space = ["10.1.0.0/16"]
6
7  subnet_name          = "aks-subnet"
8  subnet_address_prefixes = ["10.1.0.0/24"]
9
10 acr_name = "hptacr"
11 acr_sku  = "Standard"
12
13 aks_name                = "aks-cluster-service"
14 aks_dns_prefix          = "aks-dns"
15 aks_version             = "1.26.6"
16 aks_sku                 = "Free"
17 aks_node_pool_name      = "aksagentpool"
18 aks_node_pool_count     = 1
19 aks_node_pool_size      = "Standard_D2_v2"
20 aks_node_pool_max_count = 2
21 aks_node_pool_min_count = 1
22 aks_rbac_enable         = true
23 aks_network_plugin      = "azure"
24 aks_network_policy      = "azure"
25
26 role_assignment_name = "AcrPull"

```

- Notes: variable name and type defined in **terraform.tf** file

### 1.3 Script to create Virtual Network (VNET) and subnets using in the AKS cluster

- Open the **main.tf** file and add script to create a virtual network (VNET) and a subnet using in AKS cluster.

```

main.tf > ...
21
22 # Create a virtual network (VNET)
23 resource "azurerm_virtual_network" "aks_vnet" {
24     name                = var.vnet_name
25     location            = azurerm_resource_group.rg.location
26     resource_group_name = azurerm_resource_group.rg.name
27     address_space       = var.vnet_address_space
28 }
29
30 # Create a subnet for AKS
31 resource "azurerm_subnet" "aks_subnet" {
32     name                = var.subnet_name
33     resource_group_name = azurerm_resource_group.rg.name
34     virtual_network_name = azurerm_virtual_network.aks_vnet.name
35     address_prefixes     = var.subnet_address_prefixes
36 }

```

- Notes: reference to **variables.tfvars** file to get value of variables

## 1.4 Script to create Azure Container Registry (ACR)

- At the end of **main.tf** file, add script below to create ACR resource

```

38 # Create an ACR
39 resource "azurerm_container_registry" "acr" {
40     name                = var.acr_name
41     resource_group_name = azurerm_resource_group.rg.name
42     location            = azurerm_resource_group.rg.location
43     sku                 = var.acr_sku
44     admin_enabled       = true
45 }

```

- Notes: reference to **variables.tfvars** file to get value of variables

## 1.5 Script to create Azure Kubernetes Service (AKS)

- At the end of **main.tf** file, add script to create AKS resource and grant permission to AKS can pull images form container registry (ACR).
- Grant permission for AKS

```

main.tf > ...
81 # Grant permission for AKS pull image from Container Registry (ACR)
82 resource "azurerm_role_assignment" "aks_role" {
83     scope                = azurerm_container_registry.acr.id
84     role_definition_name = "AcrPull"
85     principal_id         = azurerm_kubernetes_cluster.aks.kubelet_identity[0].object_id
86
87     depends_on = [azurerm_kubernetes_cluster.aks]
88 }
89

```

- Terraform script create the AKS resource

```
main.tf > resource "azurerm_kubernetes_cluster" "aks" > network_profile > network_policy
46 # Create an AKS cluster
47 resource "azurerm_kubernetes_cluster" "aks" {
48     name                = var.aks_name
49     location            = azurerm_resource_group.rg.location
50     resource_group_name = azurerm_resource_group.rg.name
51     dns_prefix          = var.aks_dns_prefix
52     kubernetes_version  = var.aks_version
53     sku_tier            = var.aks_sku
54
55     role_based_access_control_enabled = var.aks_rbac_enable
56
57     default_node_pool {
58         name                = var.aks_node_pool_name
59         node_count          = var.aks_node_pool_count
60         vm_size             = var.aks_node_pool_size
61         vnet_subnet_id      = azurerm_subnet.aks_subnet.id
62         enable_auto_scaling = true
63         min_count           = var.aks_node_pool_min_count
64         max_count           = var.aks_node_pool_max_count
65     }
66
67     network_profile {
68         network_plugin = var.aks_network_plugin
69         network_policy = var.aks_network_policy
70     }
71
72     identity {
73         type = "SystemAssigned"
74     }
75
76     depends_on = [azurerm_subnet.aks_subnet]
77 }
```

- Notes: reference to **variables.tfvars** file to get value of variables

## 1.6 Run Terraform scripts

- In the root folder, open a Terminal and type command > **terraform init**

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS D:\Others\Practical DevOps\Source\Infrastructure\Terraform> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~> 3.0.2"...
- Installing hashicorp/azurerm v3.0.2...
- Installed hashicorp/azurerm v3.0.2 (signed by HashiCorp)
```

- Then, type command > **terraform plan**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\Others\Practical DevOps\Source\Infrastructure\Terraform> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
```

Plan: 6 to add, 0 to change, 0 to destroy.

- Type command > terraform apply

```
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

azure_rm_resource_group.rg: Creating...
azure_rm_resource_group.rg: Creation complete after 1s [id=/subscriptions/c713.../resourceGroups/prac
azure_rm_virtual_network.aks_vnet: Creating...
azure_rm_container_registry.acr: Creating...
azure_rm_virtual_network.aks_vnet: Creation complete after 4s [id=/subscriptions/c713.../resourceGrou
azure_rm_subnet.aks_subnet: Creating...
azure_rm_subnet.aks_subnet: Creation complete after 4s [id=/subscriptions/c713.../resourceGroups/prac
azure_rm_kubernetes_cluster.aks: Creating...
azure_rm_container_registry.acr: Still creating... [10s elapsed]
azure_rm_container_registry.acr: Still creating... [20s elapsed]
azure_rm_container_registry.acr: Creation complete after 26s [id=/subscriptions/c71.../resourceGroup
```

1.7 Verify resources created on Azure

- After resources create successfully, login Azure portal to verify resources created
- Resource group

practice-devops-resource

Resource group

CreateManage viewDelete resource groupRefreshExport to CSVOpen queryAssign tags

EssentialsJSON View

ResourcesRecommendations

Filter for any field...Type equals allLocation equals allAdd filter

Showing 1 to 3 of 3 records.Show hidden typesNo groupingList view

Name	Type	Location
aks-cluster-service	Kubernetes service	Southeast Asia
aks-vnet	Virtual network	Southeast Asia
hptacr	Container registry	Southeast Asia



- Azure container registry (ACR)

The screenshot shows the Azure portal interface for the 'hptacr' Container registry resource. The breadcrumb navigation is 'Home > Resource groups > practice-devops-resource > hptacr'. The resource is identified as a 'Container registry'. Below the header, there are links for 'Move' and 'Delete'. The 'Essentials' section displays the following details:

Resource group (move)	practice-devops-resource	Login server	hptacr.azurecr.io
Location	Southeast Asia	Creation date	8/20/2023, 1:20 AM GMT+7
Subscription (move)	Visual Studio Enterprise	Provisioning state	Succeeded
Subscription ID	c7	Pricing plan	Standard

- Virtual network and subnets

The screenshot shows the Azure portal interface for the 'aks-vnet' Virtual network resource. The breadcrumb navigation is 'Home > Resource groups > practice-devops-resource > aks-vnet'. The resource is identified as a 'Virtual network'. Below the header, there are links for 'Move', 'Delete', 'Refresh', and 'Give feedback'. The 'Essentials' section displays the following details:

Resource group (move)	practice-devops-resource	Address space	: 10.1.0.0/16
Location (move)	: Southeast Asia	DNS servers	: Azure provided DNS service
Subscription (move)	: Visual Studio Enterprise	Flow timeout	: Configure
Subscription ID	: c7	BGP community string	: Configure
		Virtual network ID	: 9eb1

The screenshot shows the 'Subnets' view for the 'aks-vnet' Virtual network. The breadcrumb navigation is 'Home > Resource groups > practice-devops-resource > aks-vnet > Subnets'. The interface includes a search bar for subnets and a table listing the available subnets.

Name ↑↓	IPv4 ↑↓	IPv6 ↑↓	Available IPs ↑↓	Delegated to ↑↓	Security group ↑↓	Route table ↑↓
aks-subnet	10.1.0.0/24	-	222	-	-	-

- Azure Kubernetes service (AKS)

Microsoft Azure Search resources, services, and docs (G+)

Home > Resource groups > practice-devops-resource >

## aks-cluster-service

Kubernetes service

» + Create Connect Start Stop Delete Refresh Open in mobile Give feedback

Essentials JSON View

Get started **Properties** Monitoring Capabilities (4) Recommendations Tutorials

**Kubernetes services**

Encryption type Encryption at-rest with a platform-managed key

Virtual node pools Not enabled

**Node pools**

Node pools 1 node pool

Kubernetes versions 1.26.6

Node sizes Standard\_D2\_v2

**Configuration**

Kubernetes version 1.26.6

Auto Upgrade Type Disabled

Authentication and Authorization Local accounts with Kubernetes RBAC

Local accounts Enabled

**Networking**

API server address aks-dns-2l4lx42b.hcp.southeastasia.azmk8s.io

Network type (plugin) Azure CNI

Pod CIDR -

Service CIDR 10.0.0.0/16

DNS service IP 10.0.0.10

Docker bridge CIDR -

Network Policy Azure

Load balancer Standard

HTTP application routing Not enabled

Private cluster Not enabled

Authorized IP ranges Not enabled

Application Gateway ingress controller Not enabled

**Node pools** Nodes

Node pools provide space for applications to run. Node pools of different types can be added to the cluster to handle a variety of workloads, existing node pools can be scaled and upgraded, or node pools that are no longer needed can be deleted. Each node pool will contain nodes backed by virtual machines. [Learn more about node pools](#)

Node pool ↓	Provisioning state ⓘ	Power state ⓘ	Node count ⓘ	Mode	Kubernetes version	Node size	Operating system
aksagentpool	Succeeded	Running	✓ 1/1 ready	System	1.26.6	Standard_D2_v2	Linux

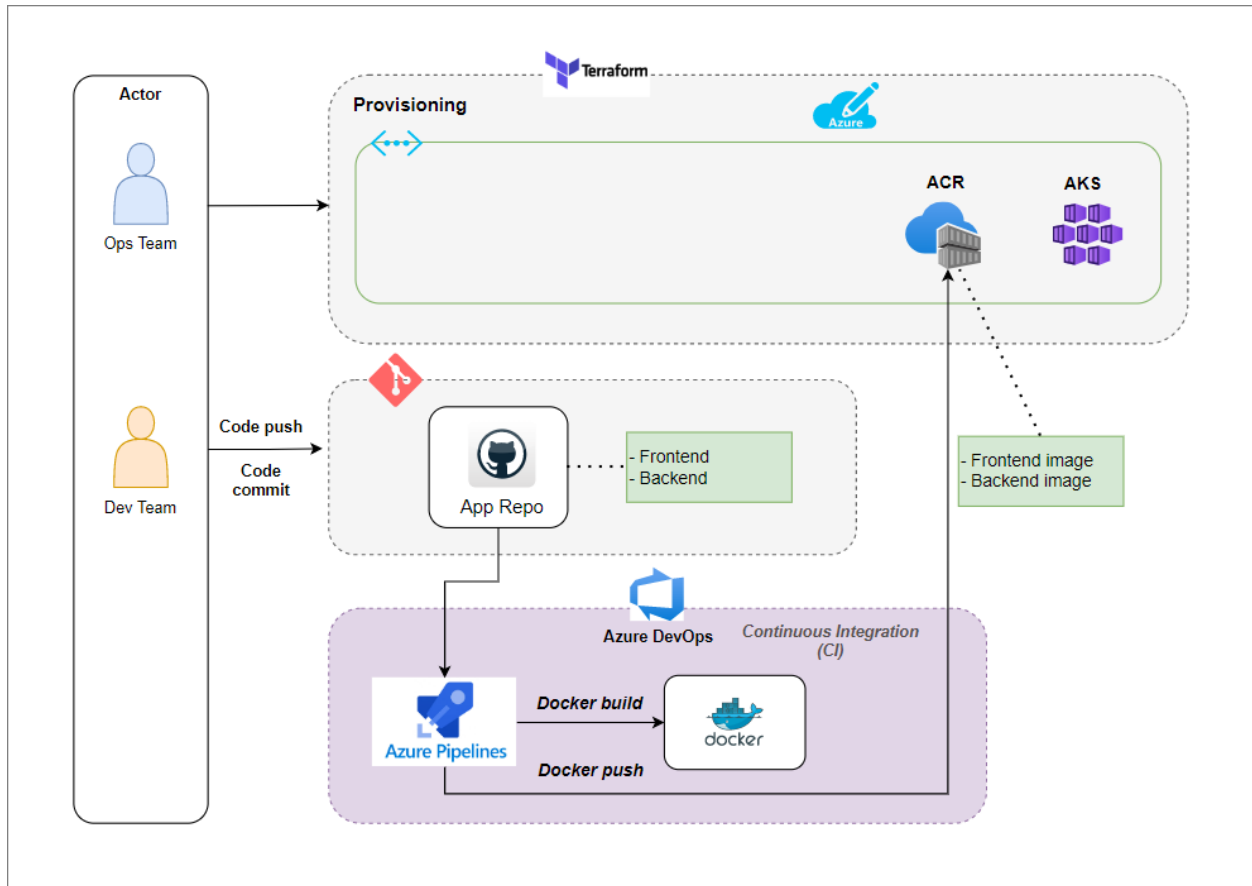
## 2 CI: Setting up an azure pipeline, build and push Docker image to ACR

GitHub repository: [https://github.com/hptruc/sd2350\\_msa](https://github.com/hptruc/sd2350_msa)

Precondition:

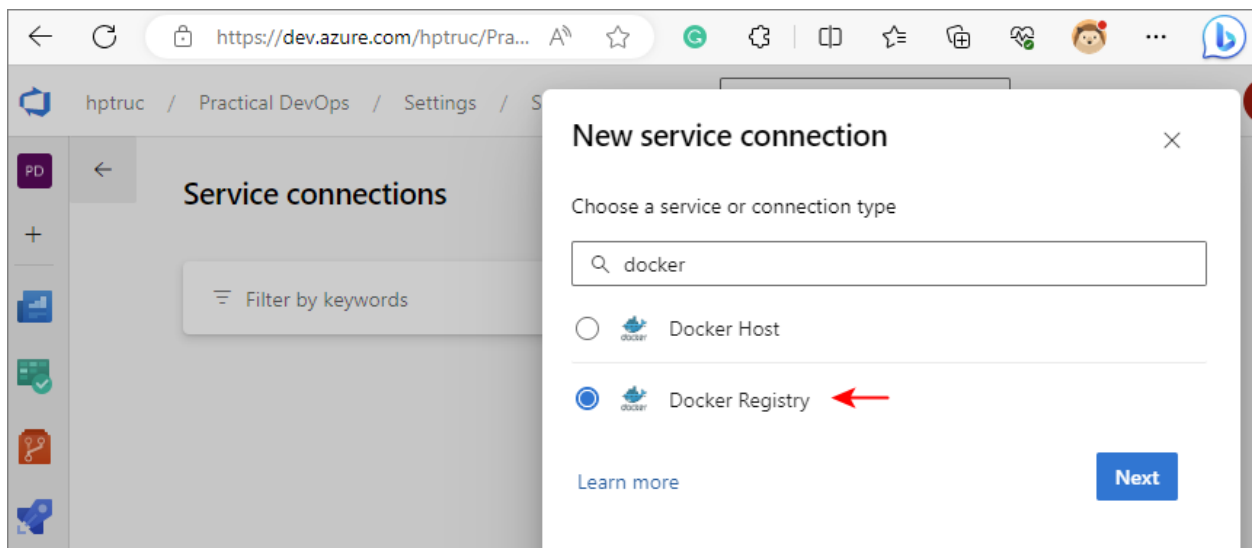
- Create an Azure DevOps instance and request to Microsoft ([fill form](#)) to get a Microsoft-hosted agents for build pipelines.

Architecture overview:



## 2.1 Setting up Azure Docker Registry service connection

- Login to Azure DevOps instance and create new a project, e.g. Practical DevOps
- Go to Project settings page > Service connections > New service connection
- Select Docker Registry and click on Next button



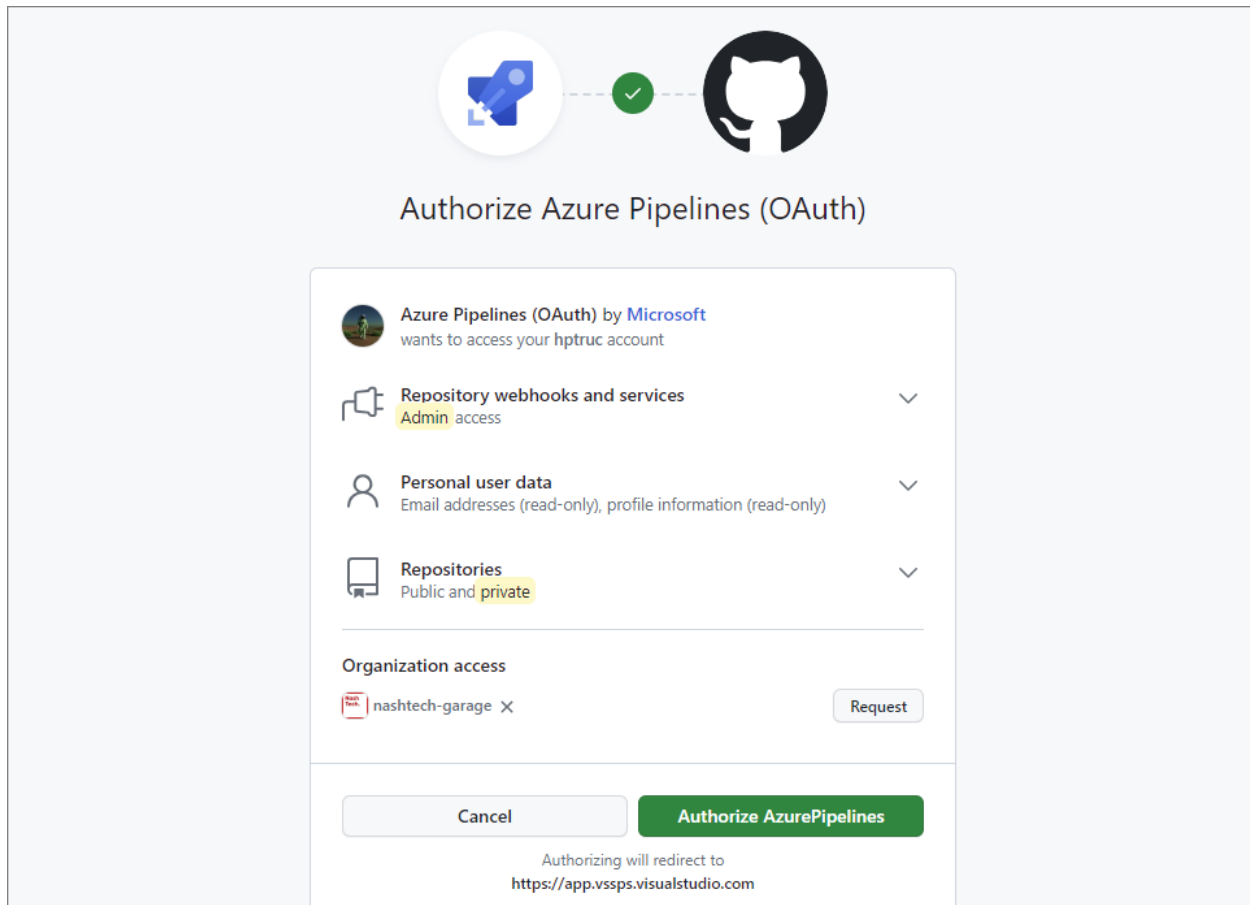
- Select **Others** option and fill required information

- The “**Azure container registry login server**” field: Link of ACR instance.

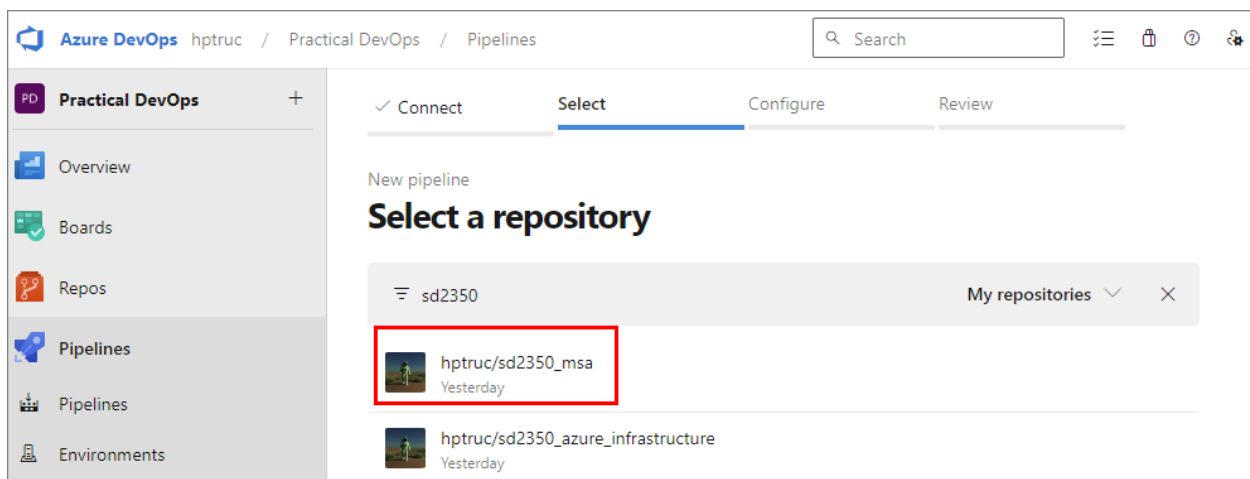
## 2.2 Create new a CI pipeline and connect to MSA GitHub repository

- Click on Pipelines item from left side bar > New pipeline
- Select GitHub method

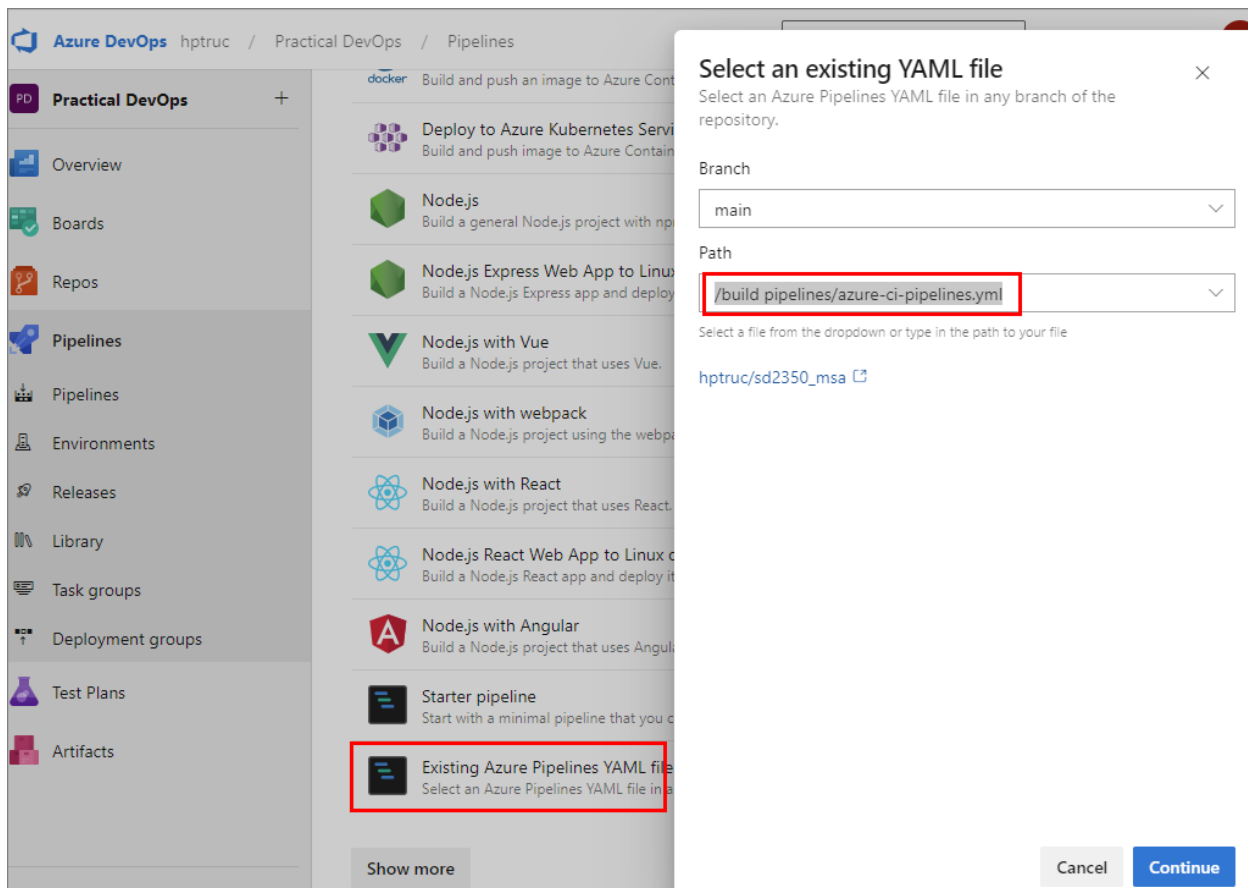
- Azure DevOps will redirect to GitHub authorize page



- Click on **Authorize Pipelines** button
- After authorize successfully, at the Azure DevOps page, select MSA repository



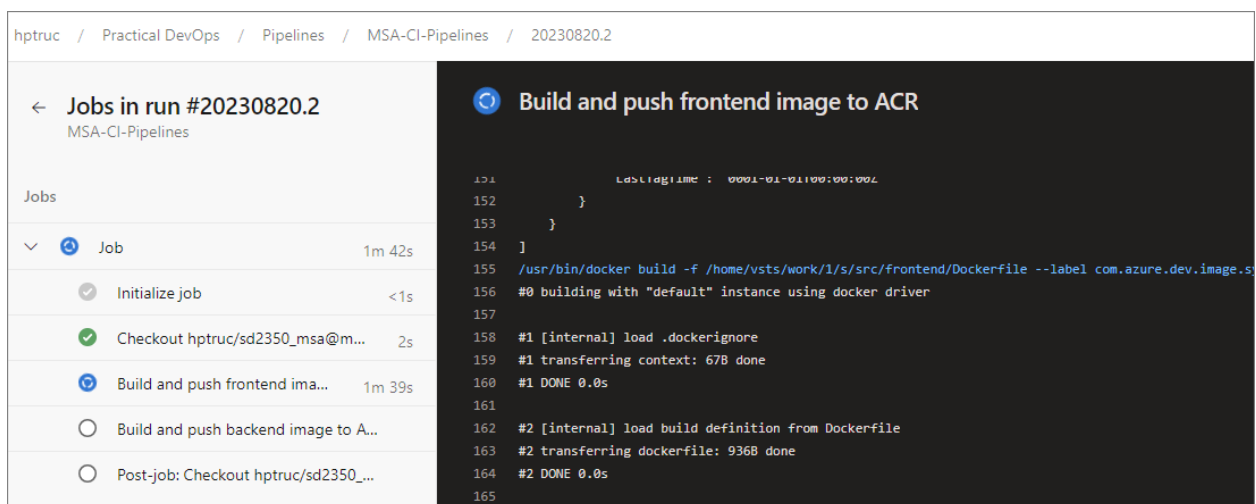
- Approve permission from GitHub page to access to this repository
- The next, in tab **Configure** > select item **Existing Azure Pipelines YAML file**
- In the select file popup > select ci pipeline file in Path dropdown



- Verify **ACR connection** to make sure that it work correctly.
- Save pipeline with name **MSA-CI-Pipelines**

## 2.3 Run CI build pipeline and verify results

- Select pipeline **MSA-CI-Pipelines** and click on run button to start build pipeline






- Waiting for 3-5 minutes
- Check results from console to make sure that build pipeline complete without any error.

- Login to Azure portal to verify images
- Frontend image

[Home](#) >
[hptacr | Repositories](#) >

# frontend

Repository

 Refresh
 Manage deleted artifacts
 Delete repository

---

## ^ Essentials


Repository : frontend

Tag count : 2



Last updated date : 8/20/2023, 5:41 PM GMT+7

Manifest count : 2

---

 Search to filter tags ...

---




Tags 	Digest 	Last modified
190	sha256:6dc7dbe0e48d2e3f953ba49ade1a3033fe5f622ec4c9e5...	8/20/2023, 5:41 PM GMT+7

- Backend image

[Home](#) >
[hptacr](#) |
[Repositories](#) >

## backend

Repository

 Refresh
 Manage deleted artifacts
 Delete repository

^ Essentials

Repository : backend

Tag count : 2

Last updated date : 8/20/2023, 5:42 PM GMT+7

Manifest count : 2

Tags ↑↓	Digest ↑↓	Last modified
<a href="#">190</a>	sha256:9d925664a1b16a82a58a815c51b789f8c53f5f539490f5...	8/20/2023, 5:42 PM GMT+7

- Complete Azure CI build pipeline.

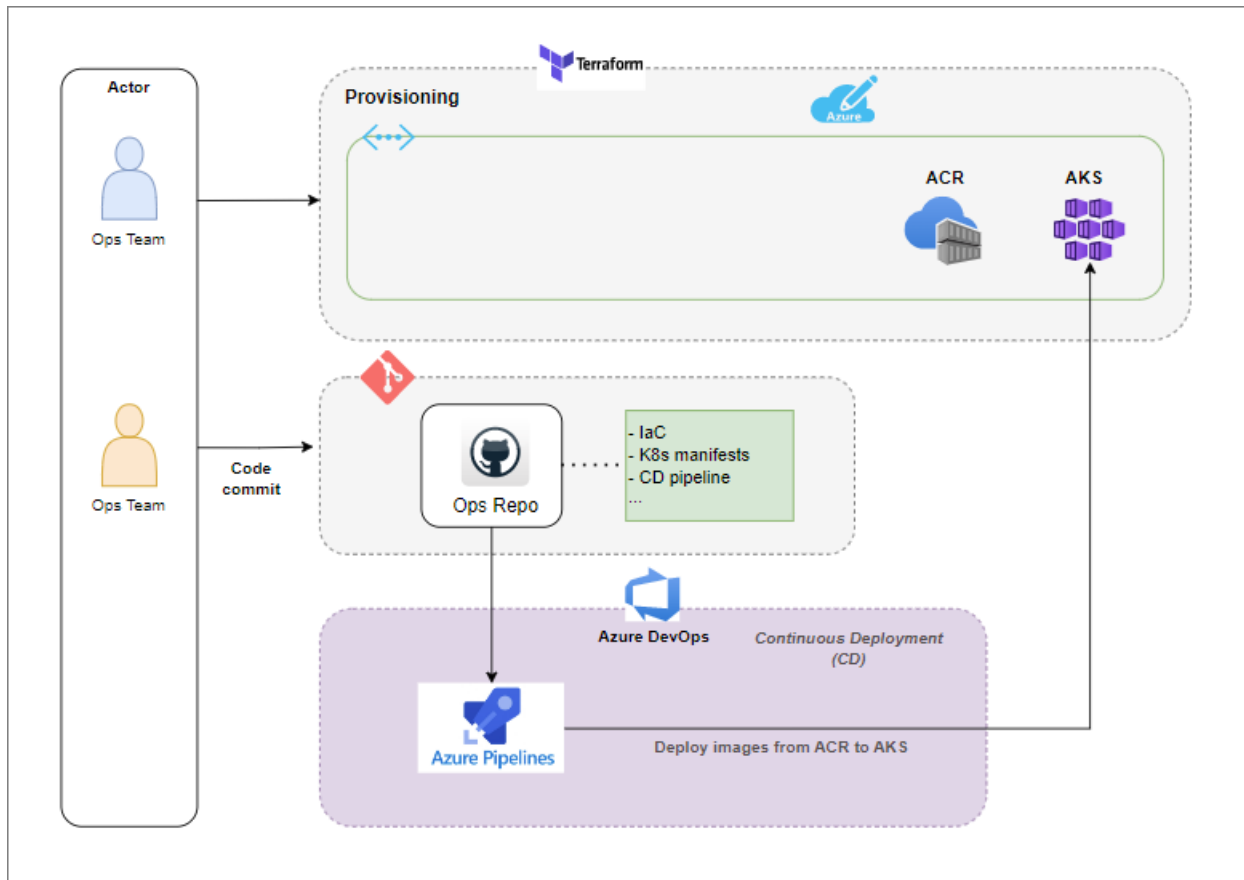
### 3 CD: Setting up an azure pipeline and deploying applications on AKS

GitHub repository: [https://github.com/hptruc/sd2350\\_azure\\_infrastructure](https://github.com/hptruc/sd2350_azure_infrastructure)

Precondition:

- CI build pipeline executed to push MSA images to Azure Container Registry

Architecture overview:



### 3.1 Create a Kubernetes namespace using for deployment

- Open terminal and type this command to create a namespace
- Namespace: **aks-ns**

```
Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe create namespace aks-ns
```

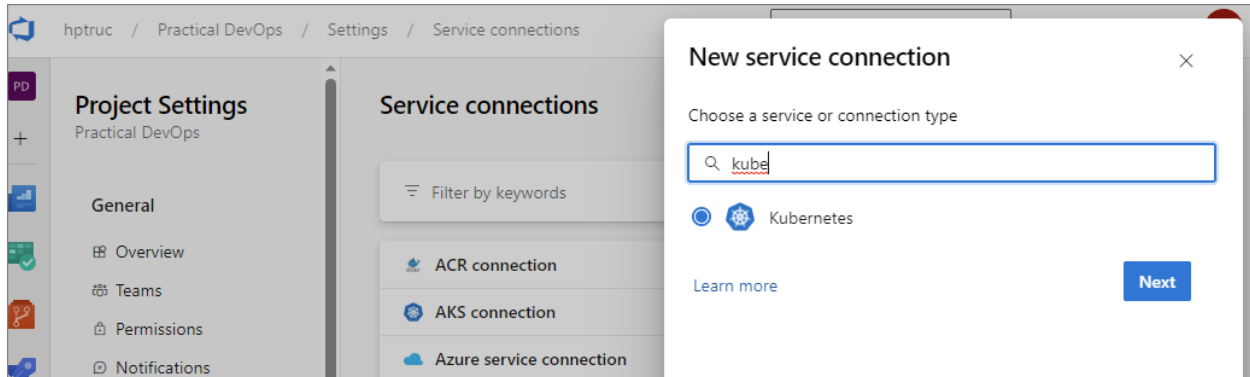
- Verify namespace created

```
Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get namespace
NAME      STATUS   AGE
aks-ns    Active   3h4m
```



## 3.2 Setting up AKS service connection

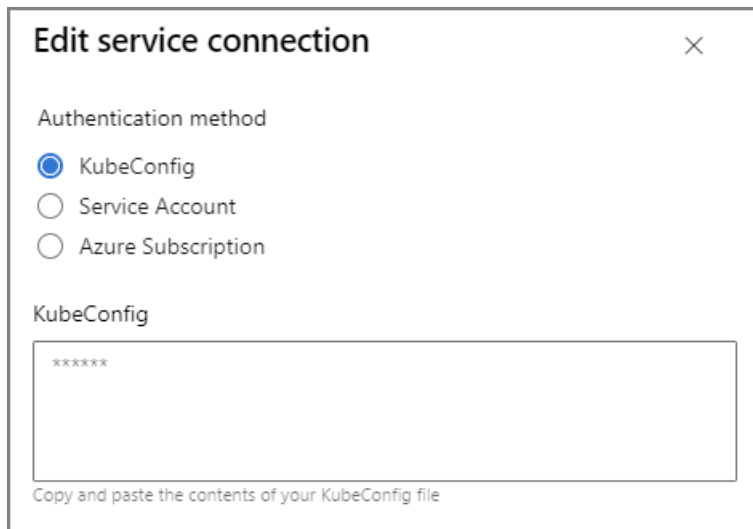
- Login to Azure DevOps instance > access **Practical DevOps** project
- Go to Project settings page > Service connections > New service connection
- Select **Kubernetes** and click on Next button



- You can choose one of three options depending on the Azure account you are using. In this demo, I choose the option KubeConfig.
- Open terminal and type this command to set Kubernetes credential to configure file

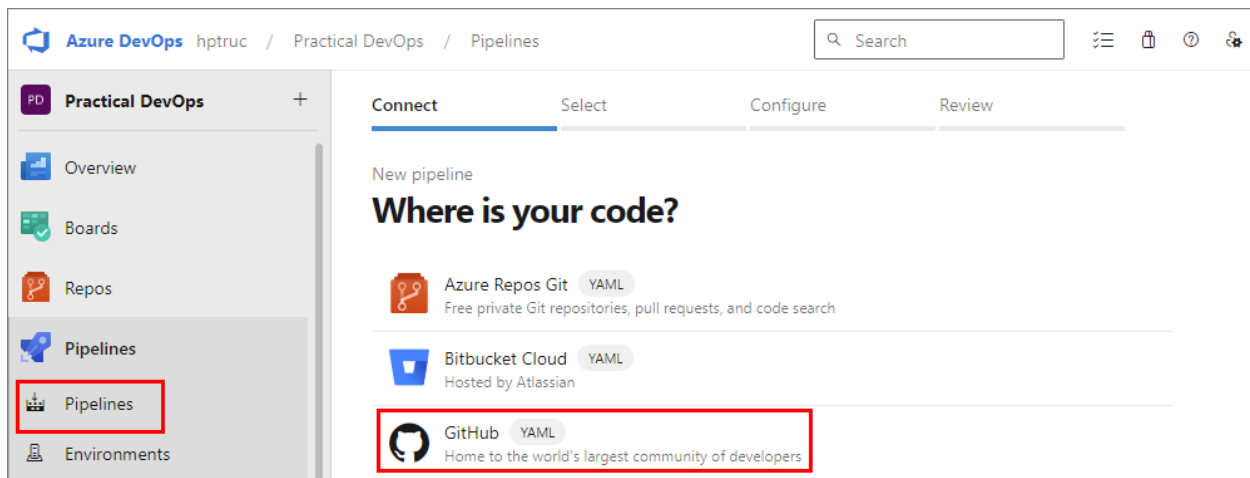
```
Windows PowerShell
PS C:\Users\truchop> az aks get-credentials --resource-group practice-devops-resource --name aks-cluster-service
Merged "aks-cluster-service" as current context in C:\Users\truchop\.kube\config
PS C:\Users\truchop> |
```

- Go to configure file with path in console (C:\Users\...) )
- Copy all content of configure file and paste into textbox KubeConfig
- Click on verify button and save service connection with name **AKS connection**



## 3.3 Create new a CD pipeline and connect to Infrastructure GitHub repository

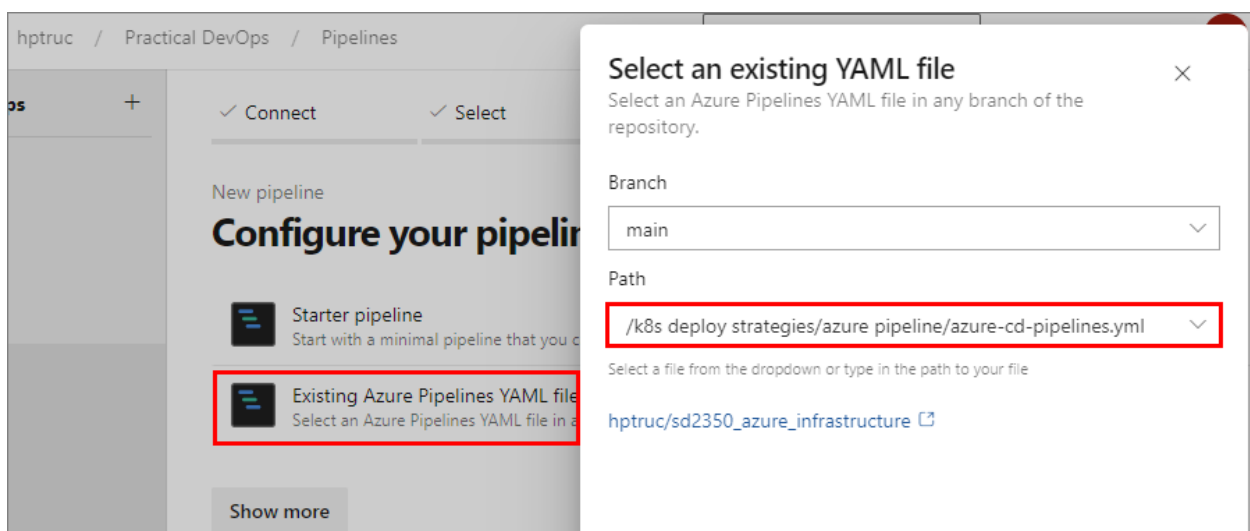
- Click on Pipelines item from left side bar > New pipeline
- Select GitHub method



- In the Azure DevOps page, select infrastructure repository



- Approve permission from GitHub page to access to this repository
- The next, in tab Configure > select **Existing Azure Pipelines YAML file** item
- In the select file popup > select cd pipeline file in Path dropdown



- Save pipeline with name **MSA-CD-Pipelines**

### 3.4 Run CD build pipeline and verify results

- Select pipeline **MSA-CD-Pipelines** and click on run to start build pipeline

← Jobs in run #20230820.27

MSA-CD-Pipelines

Deploy MSA source

>	✓ Deploy Database	8s
>	✓ Deploy Frontend	8s
>	✓ Deploy Backend	6s

Application Ingress

☐ NGINX ingress controller
 ☐ Application Ingress

✓ Deploy Database

```

1 Pool: Azure Pipelines
2 Image: ubuntu-latest
3 Queued: Just now [manage_parallel_jobs]
4 Agent: Hosted Agent
5 Started: Just now
6 Duration: 8s
7
8 The agent request is already running or has already completed.
9 ▶ Job preparation parameters
10 Job live console data:
11 Starting: Deploy Database
12 Async Command Start: DetectDockerContainer
13 Async Command End: DetectDockerContainer
14 Async Command Start: DetectDockerContainer
15 Async Command End: DetectDockerContainer
16 Finishing: Deploy Database

```

- Waiting for 3-5 minutes
- Check result from console logs to make sure build pipeline complete without any error

← Jobs in run #20230820.27

MSA-CD-Pipelines

Deploy MSA source

>	✓ Deploy Database	8s
>	✓ Deploy Frontend	8s
>	✓ Deploy Backend	6s

Application Ingress

☐ NGINX ingress controller
 ☐ Application Ingress

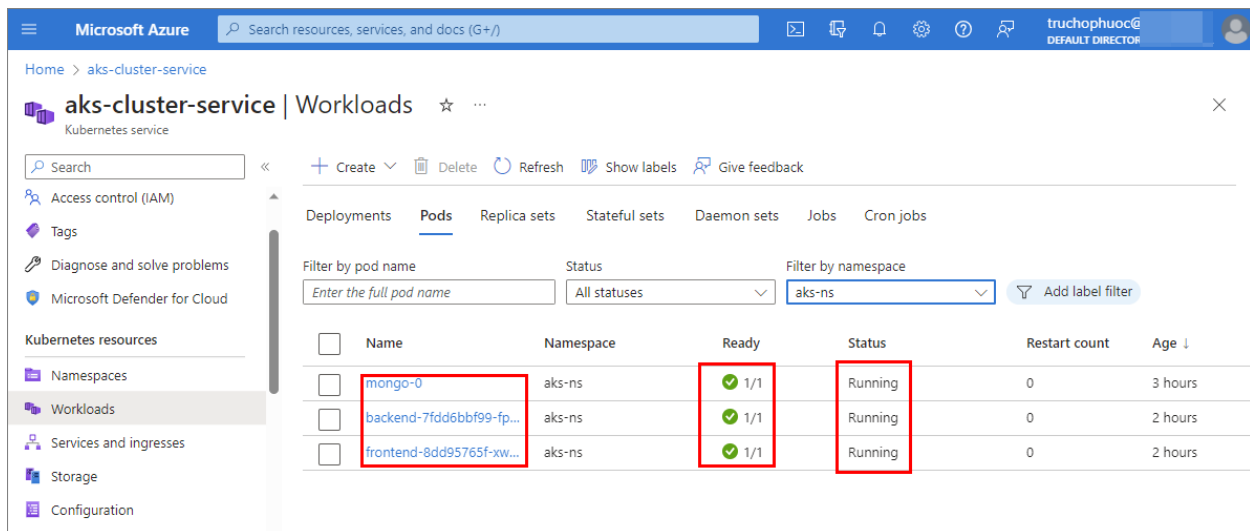
✓ Deploy Database

```

1 Pool: Azure Pipelines
2 Image: ubuntu-latest
3 Queued: Just now [manage_parallel_jobs]
4 Agent: Hosted Agent
5 Started: Just now
6 Duration: 8s
7
8 The agent request is already running or has already completed.
9 ▶ Job preparation parameters
10 Job live console data:
11 Starting: Deploy Database
12 Async Command Start: DetectDockerContainer
13 Async Command End: DetectDockerContainer
14 Async Command Start: DetectDockerContainer
15 Async Command End: DetectDockerContainer
16 Finishing: Deploy Database

```

- Login to Azure portal to verify images



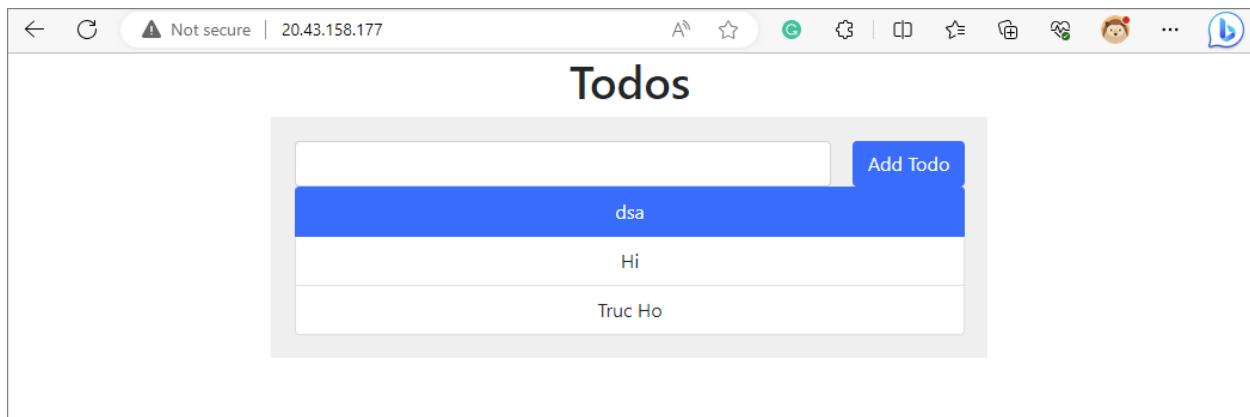
- Open terminal and type this command

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get -n aks-ns ingress -o wide
NAME      CLASS  HOSTS      ADDRESS      PORTS      AGE
ingress   nginx  *          20.43.158.177 80         130m
PS C:\Users\truchop>

```

- Copy ingress IP ADDRESS and paste to browser to go to web page



- Complete Azure CD deployment pipeline.

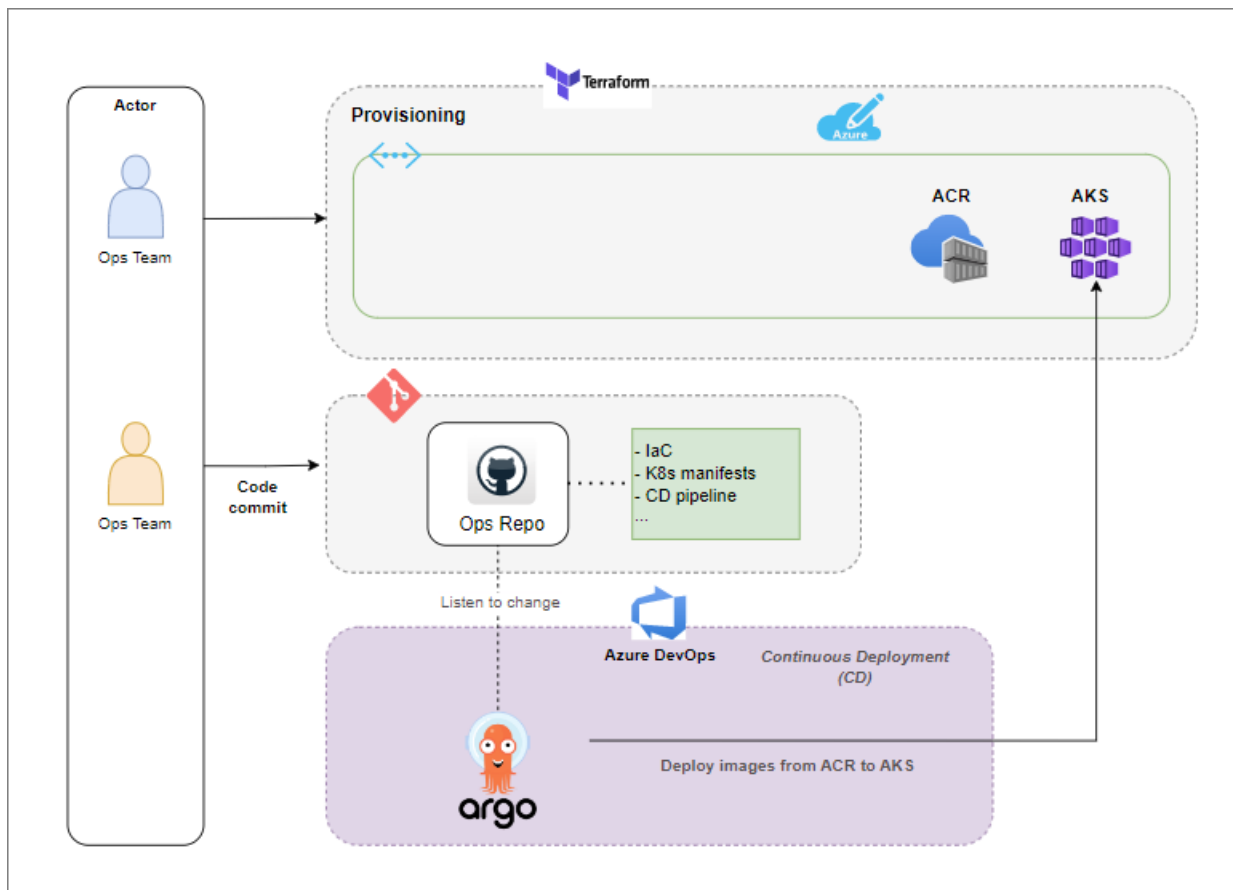
## 4 GitOps: Replace azure pipeline CD by Argo CD

GitHub repository:

Precondition:

- CI build pipeline executed to push MSA images to Azure Container Registry.

Architecture overview:



## 4.1 Install Argo CD on AKS

- Open terminal and create a namespace for Argo CD
- Namespace: argocd

```
Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe create namespace argocd
namespace/argocd created
```

- Verify namespace created

```
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get namespace
NAME          STATUS   AGE
aks-ns        Active   3h53m
argocd        Active   8s
default       Active   17h
ingress-nginx Active   126m
kube-node-lease Active   17h
kube-public   Active   17h
kube-system   Active   17h
```

- Install Argo CD by script: <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

```

Windows PowerShell
kube-public      Active  17h
kube-system      Active  17h
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
customresourcedefinition.apiextensions.k8s.io/applications.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/applicationsets.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/appprojects.argoproj.io created
serviceaccount/argocd-application-controller created
serviceaccount/argocd-applicationset-controller created
serviceaccount/argocd-dex-server created
serviceaccount/argocd-notifications-controller created
serviceaccount/argocd-redis created
serviceaccount/argocd-repo-server created
serviceaccount/argocd-server created
role.rbac.authorization.k8s.io/argocd-application-controller created
role.rbac.authorization.k8s.io/argocd-applicationset-controller created
role.rbac.authorization.k8s.io/argocd-dex-server created
role.rbac.authorization.k8s.io/argocd-notifications-controller created
role.rbac.authorization.k8s.io/argocd-server created
clusterrole.rbac.authorization.k8s.io/argocd-application-controller created
clusterrole.rbac.authorization.k8s.io/argocd-server created

```

- Verify Argo CD installed successfully

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get all -n argocd
NAME                                     READY   STATUS             RESTARTS   AGE
pod/argocd-application-controller-0      0/1     ContainerCreating   0           12s
pod/argocd-applicationset-controller-5787d44dff-dwffn  1/1     Running             0           13s
pod/argocd-dex-server-858cfd495f-ncb68    0/1     Init:0/1            0           13s
pod/argocd-notifications-controller-5d889fdf74-sf5xn  0/1     ContainerCreating   0           13s
pod/argocd-redis-7d8d46cc7f-cqrck        0/1     ContainerCreating   0           13s
pod/argocd-repo-server-7b6d785784-2vgqm   0/1     Init:0/1            0           13s
pod/argocd-server-67f667d48c-6cdts       0/1     ContainerCreating   0           12s

```

- Get raw password (base 64) login to Argo CD

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}"
emQ0ckVXYTR3RVRwRDN0SQ==
PS C:\Users\truchop>

```

- Decode password by command

```

Windows PowerShell
PS C:\Users\truchop> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("emQ0ckVXYTR3RVRwRDN0SQ=="))
zd4rEWa4wETpD3tI
PS C:\Users\truchop>

```

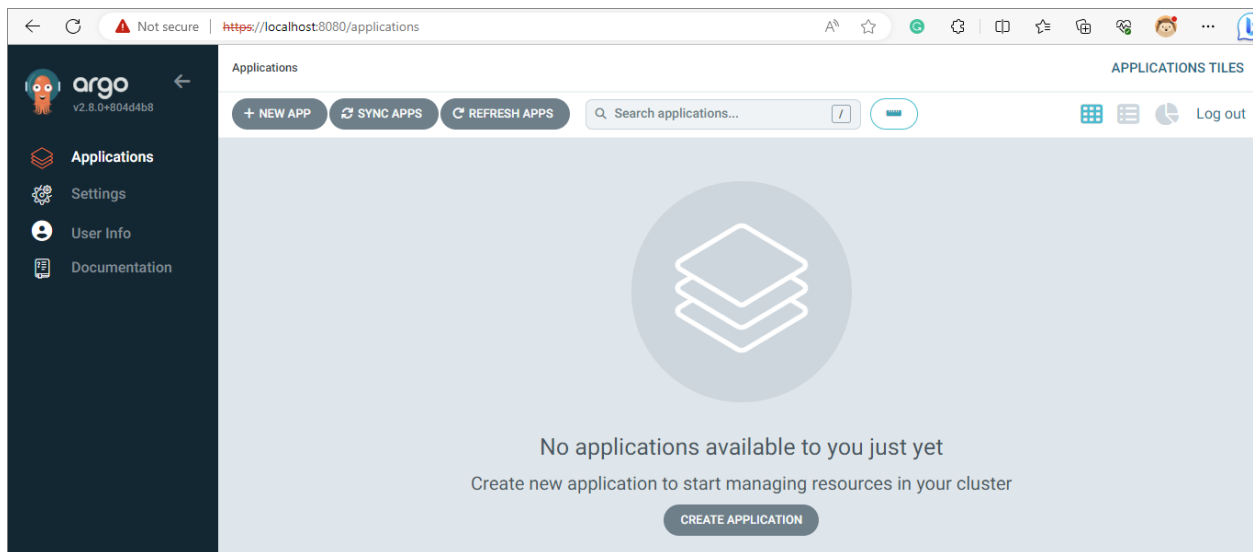
- Finally, use port forwarding to connect to the `argocd-server` service in AKS cluster.

```

Windows PowerShell
zd4rEWa4wETpD3tI
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe port-forward svc/argocd-server -n argocd 8080:443
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080

```

- Open browser and go to address: <http://localhost:8080> with credential
  - o Username: admin
  - o Password: { decoded in previous step }



## 4.2 Add new applications to Argo CD

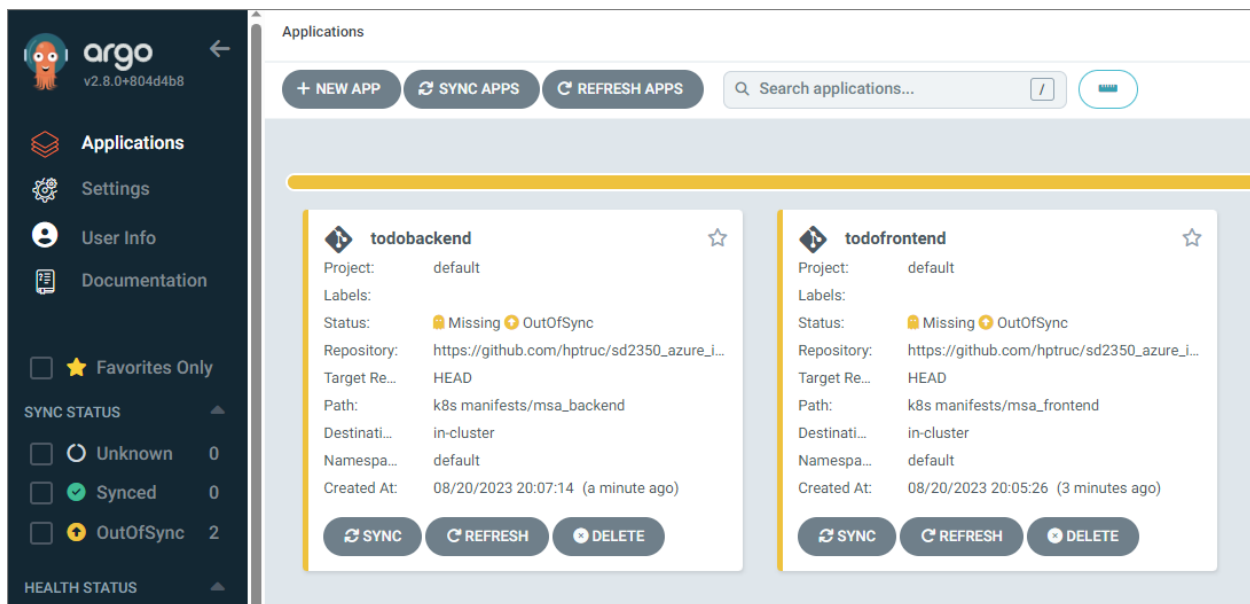
- In Argo CD portal, click on New App button to create new application
- For demo purpose, I will create 2 applications Frontend and Backend (skip database)
- Frontend

Field Name	Value
Application Nam	todofrontend
Project Name	default
Sync Policy	Manual
Repository URL	<a href="https://github.com/hptruc/sd2350_azure_infrastructure.git">https://github.com/hptruc/sd2350_azure_infrastructure.git</a>
Path	k8s manifests/msa_frontend
Cluster URL	<a href="https://kubernetes.default.svc">https://kubernetes.default.svc</a>
Namespace	default

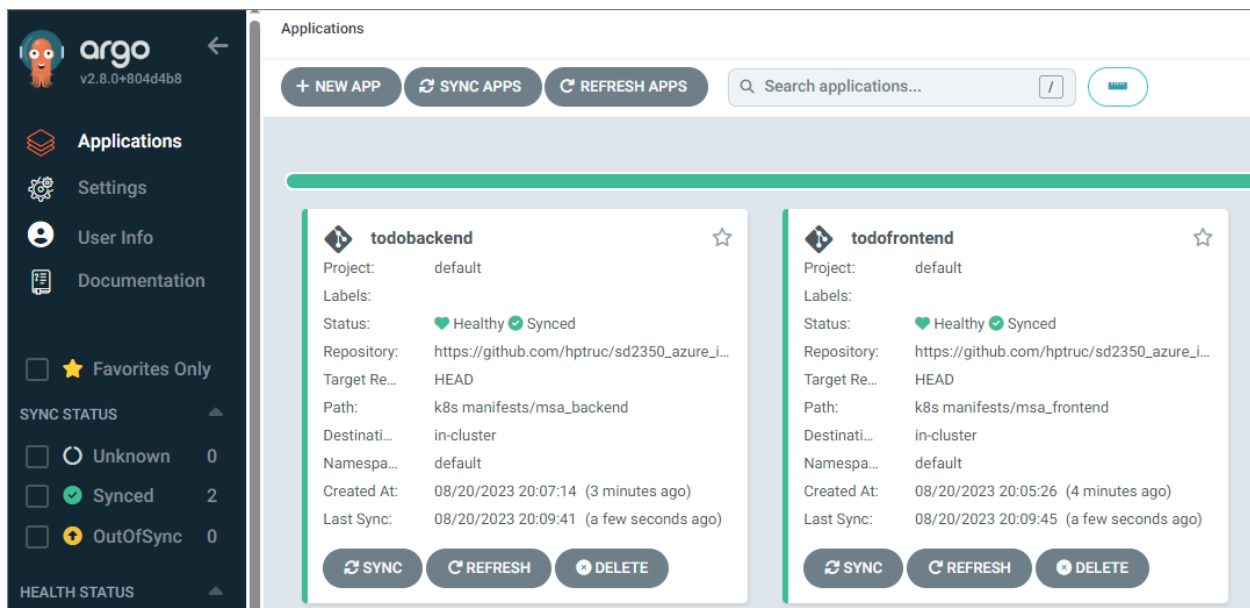
- Backend

Field Name	Value
Application Nam	todobackend
Project Name	default
Sync Policy	Manual
Repository URL	<a href="https://github.com/hptruc/sd2350_azure_infrastructure.git">https://github.com/hptruc/sd2350_azure_infrastructure.git</a>
Path	k8s manifests/msa_backend
Cluster URL	<a href="https://kubernetes.default.svc">https://kubernetes.default.svc</a>
Namespace	default

- Click on Create button to create application
- Initially, you will notice your application sync status is **OutOfSync** and your health status is **Missing**

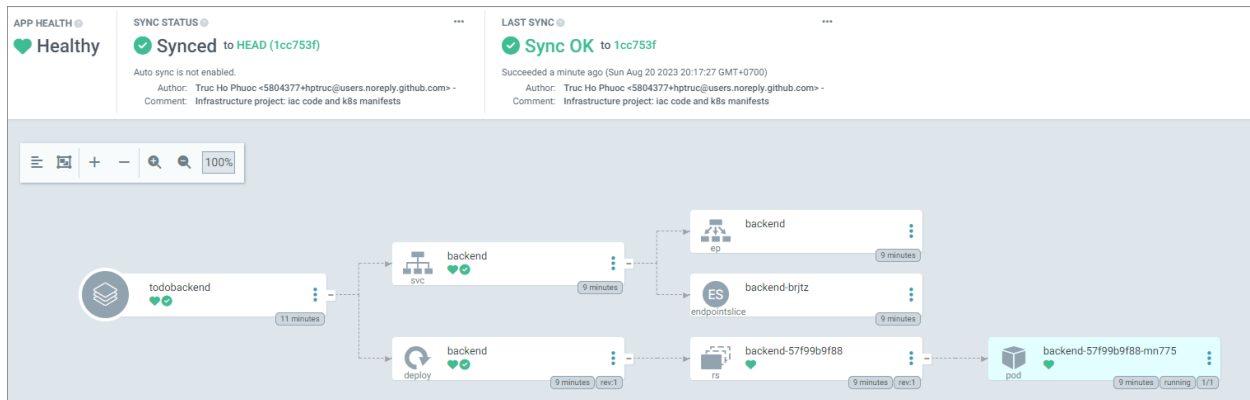


- Click on SYNC button > SYNCHRONIZE



- Verify status of applications after sync





### 4.3 Test deployment with Argo CD

- Open MSA frontend source code and change something then commit change to MSA repo to build pipeline trigger create new image on ACR
- I will change as below

```

37
38 render() {
39   return (
40     <div className="App container">
41       <div className="container-fluid">
42         <div className="row">
43           <div className="col-xs-12 col-sm-8 col-md-8">
44-            <h1>Todos</h1>
44+            <h1>Todos (Change for test Argo CD)</h1>
45           <div className="todo-app">
46             <AddTodo handleAddTodo={this.handleAddTodo}>
47               <TodoList todos={this.state.todos} />
48             </div>
49           </div>
50         </div>
51       </div>

```

- Commit code change
- CI trigger build

The screenshot shows a GitHub Actions workflow run titled '#20230820.3 • Test argo cd detect change' under the repository 'MSA-CI-Pipelines'. The 'Summary' section indicates it was 'Triggered by hptruc'. Below this, a table provides details about the repository and version, the time started and elapsed, and related work items and artifacts. The 'Jobs' section shows a single job named 'Job' with a status of 'Queued', which is highlighted by a red box.

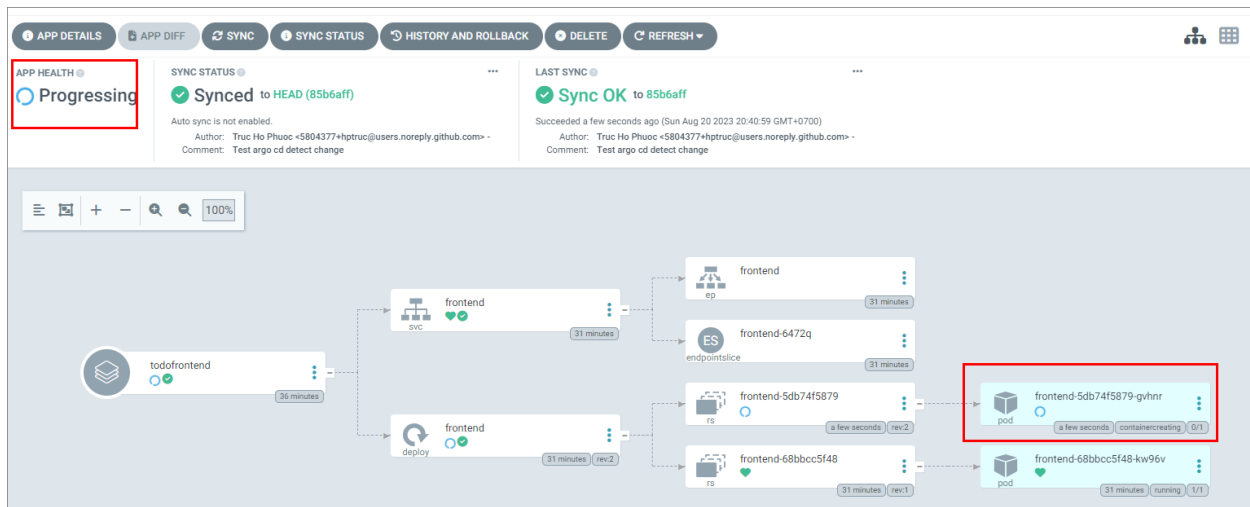
- Login to Azure portal to check new image tag

<b>frontend</b> ...		
Repository		
<a href="#">Refresh</a> <a href="#">Manage deleted artifacts</a> <a href="#">Delete repository</a>		
^ Essentials		
Repository	: frontend	Tag count : 3
Last updated date	: 8/20/2023, 8:35 PM GMT+7	Manifest count : 3
<input type="text" value="Search to filter tags ..."/>		
Tags ↑↓	Digest ↑↓	Last modified
195	sha256:b2f0fe96bfccdf7c875f60eb658026b9022b3324d77624a4cbd...	8/20/2023, 8:35 PM GMT+7
190	sha256:6dc7dbe0e48d2e3f953ba49ade1a3033fe5f622ec4c9e54a5de6c...	8/20/2023, 5:41 PM GMT+7
160	sha256:e4c23ec8fae8b20341885c7e7c64d23eb2291869114fabb5dad1...	8/20/2023, 6:50 AM GMT+7

- New image tag: 195
- Update K8s manifest of frontend to get image by new tag 195

<pre> labels:   app: backend spec:   containers:     - name: backend       image: hptacr.azurecr.io/backend:190       imagePullPolicy: Always       ports:         - containerPort: 3000 </pre>	→	<pre> labels:   app: backend spec:   containers:     - name: backend       image: hptacr.azurecr.io/backend:195       imagePullPolicy: Always       ports:         - containerPort: 3000 </pre>
<pre> 24 template: 25   metadata: 26     labels: 27       app: frontend 28   spec: 29     containers: 30       - name: frontend 31       image: hptacr.azurecr.io/frontend:190 32       imagePullPolicy: Always 33       ports: 34         - containerPort: 3000 35 36 </pre>	→	<pre> 24 template: 25   metadata: 26     labels: 27       app: frontend 28   spec: 29     containers: 30       - name: frontend 31       image: hptacr.azurecr.io/frontend:195 32       imagePullPolicy: Always 33       ports: 34         - containerPort: 3000 35 36 </pre>

- Commit code change
- Login to Argo CD dashboard, status of repo changed to **OutOfSync**
- Click on SYNC button
- Argo CD triggered deploy to AKS automatically

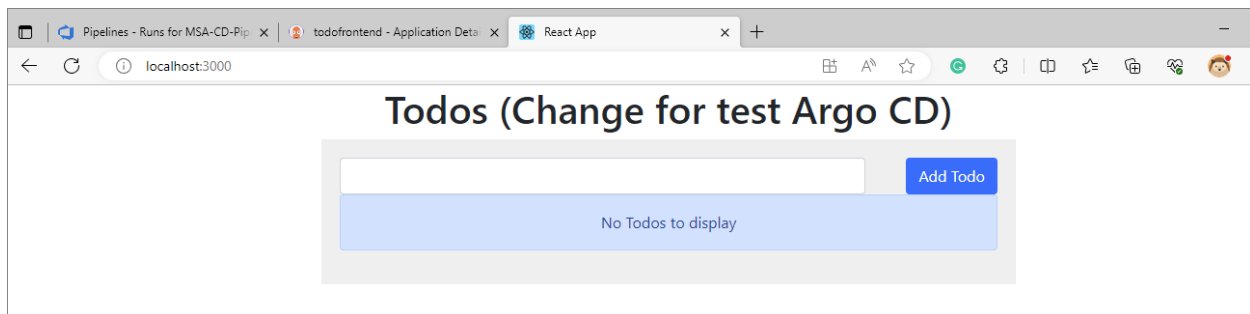


- Forward port to localhost and try access to AKS

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe port-forward service/frontend 3000:3000
Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::1]:3000 -> 3000

```



- Argo CD deployed new image successfully.

## 5 Monitoring: Setup Prometheus and Grafana to monitor AKS resources

### 5.1 Create a namespace for monitoring

- Open terminal and create a namespace with name **monitoring**

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe create namespace monitoring
namespace/monitoring created

```

- Verify namespace created

```

Windows PowerShell
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get namespaces
NAME                STATUS    AGE
aks-ns              Active    6h47m
argocd              Active    174m
default             Active    20h
ingress-nginx       Active    5h
kube-node-lease     Active    20h
kube-public         Active    20h
kube-system         Active    20h
monitoring          Active    42m
PS C:\Users\truchop>

```

## 5.2 Install Prometheus and Grafana tools

- Install helm charts

```

PS C:\Users\truchop> C:\helm\helm.exe repo add prometheus-community https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories
PS C:\Users\truchop> C:\helm\helm.exe repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "prometheus-community" chart repository
Update Complete. *Happy Helming!*

```

- Install Prometheus tool

```

PS C:\Users\truchop> C:\helm\helm.exe install prometheus prometheus-community/kube-prometheus-stack --namespace monitoring
NAME: prometheus
LAST DEPLOYED: Sun Aug 20 21:20:24 2023
NAMESPACE: monitoring
STATUS: deployed
REVISION: 1
NOTES:
kube-prometheus-stack has been installed. Check its status by running:
  kubectl --namespace monitoring get pods -l "release=prometheus"

```

- Verify Prometheus & Grafana installed

```

PS C:\Users\truchop> C:\Kubernetes\kubectl.exe get all -n monitoring
NAME                                READY    STATUS    RESTARTS    AGE
pod/alertmanager-prometheus-kube-prometheus-alertmanager-0  2/2      Running   0            2m13s
pod/prometheus-grafana-7db8f7d857-g66rf  3/3      Running   0            2m23s
pod/prometheus-kube-prometheus-operator-795b9759b8-q9rfc  1/1      Running   0            2m23s
pod/prometheus-kube-state-metrics-6df4697c45-f8zjn  1/1      Running   0            2m23s
pod/prometheus-prometheus-kube-prometheus-prometheus-0  2/2      Running   0            2m12s
pod/prometheus-prometheus-node-exporter-wzcng  1/1      Running   0            83s
pod/prometheus-prometheus-node-exporter-x4rkh  1/1      Running   0            2m24s

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)
AGE
service/alertmanager-operated      ClusterIP      None           <none>          9093/TCP,9094/TCP,9094/UDP  2m13s
service/prometheus-grafana         ClusterIP      10.0.43.80    <none>          80/TCP
2m24s
service/prometheus-kube-prometheus-alertmanager  ClusterIP      10.0.125.255  <none>          9093/TCP,8080/TCP
2m24s
2m24s
service/prometheus-kube-prometheus-prometheus  ClusterIP      10.0.89.102   <none>          9090/TCP,8080/TCP
2m24s
service/prometheus-kube-state-metrics  ClusterIP      10.0.32.137   <none>          8080/TCP
2m24s
service/prometheus-operated          ClusterIP      None           <none>          9090/TCP
2m12s
service/prometheus-prometheus-node-exporter  ClusterIP      10.0.45.81    <none>          9100/TCP
2m24s

AGE
os=linux  2m24s

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/prometheus-grafana  1/1      1              1            2m24s
deployment.apps/prometheus-kube-prometheus-operator  1/1      1              1            2m24s
deployment.apps/prometheus-kube-state-metrics  1/1      1              1            2m24s

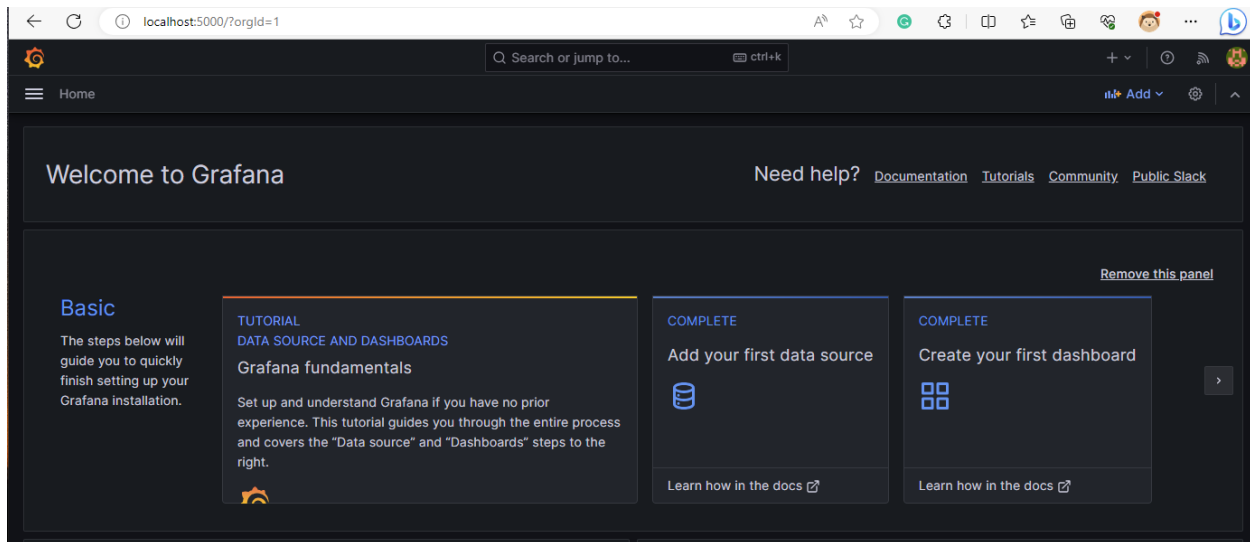
```

- Forward port to localhost to access dashboard

```
PS C:\Users\truchop> C:\Kubernetes\kubectl.exe port-forward svc/prometheus-grafana -n monitoring 5000:80
Forwarding from 127.0.0.1:5000 -> 3000
Forwarding from [::1]:5000 -> 3000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
```

### 5.3 Verify dashboard monitoring

- Open browser and paste link: <http://localhost:5000>
- Credential default: admin/prom-operator



- Go to Grafana [home page](#) and select a type dashboard to import (by ID). For example, I select this dashboard: Kubernetes Pod Metrics. ID 747

## Kubernetes Pod Metrics

Monitors Kubernetes cluster using Prometheus. Shows overall cluster CPU / Memory / Filesystem usage as well as individual pod, containers, systemd services statistics. Uses cAdvisor metrics only.

**Overview** Revisions Reviews

Uses Kube state metrics agent to expose cluster level metrics about individual pods and deployments.  
<https://github.com/kubernetes/kube-state-metrics>

Sign up for Grafana Cloud ?

Get up and running in minutes with the **Grafana Cloud free tier**, which includes free forever 10k metrics, 50GB logs, 50GB traces, 500 VUh, and more.

[Create free account](#)

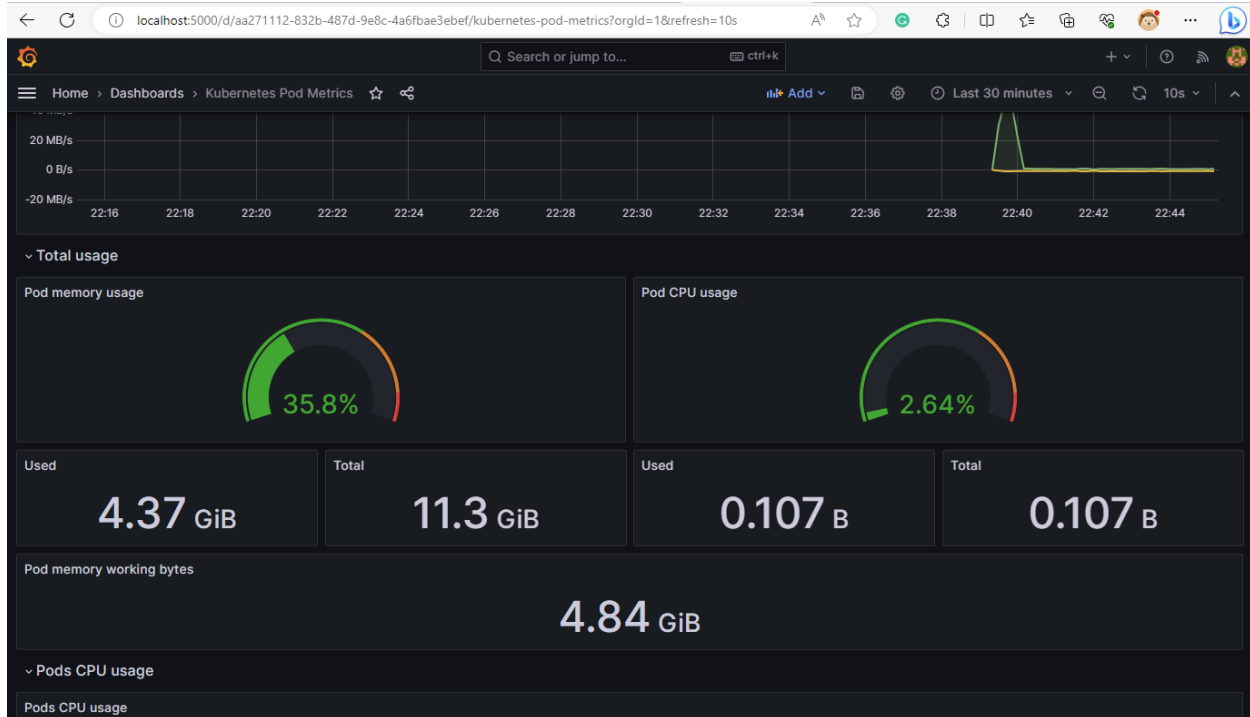
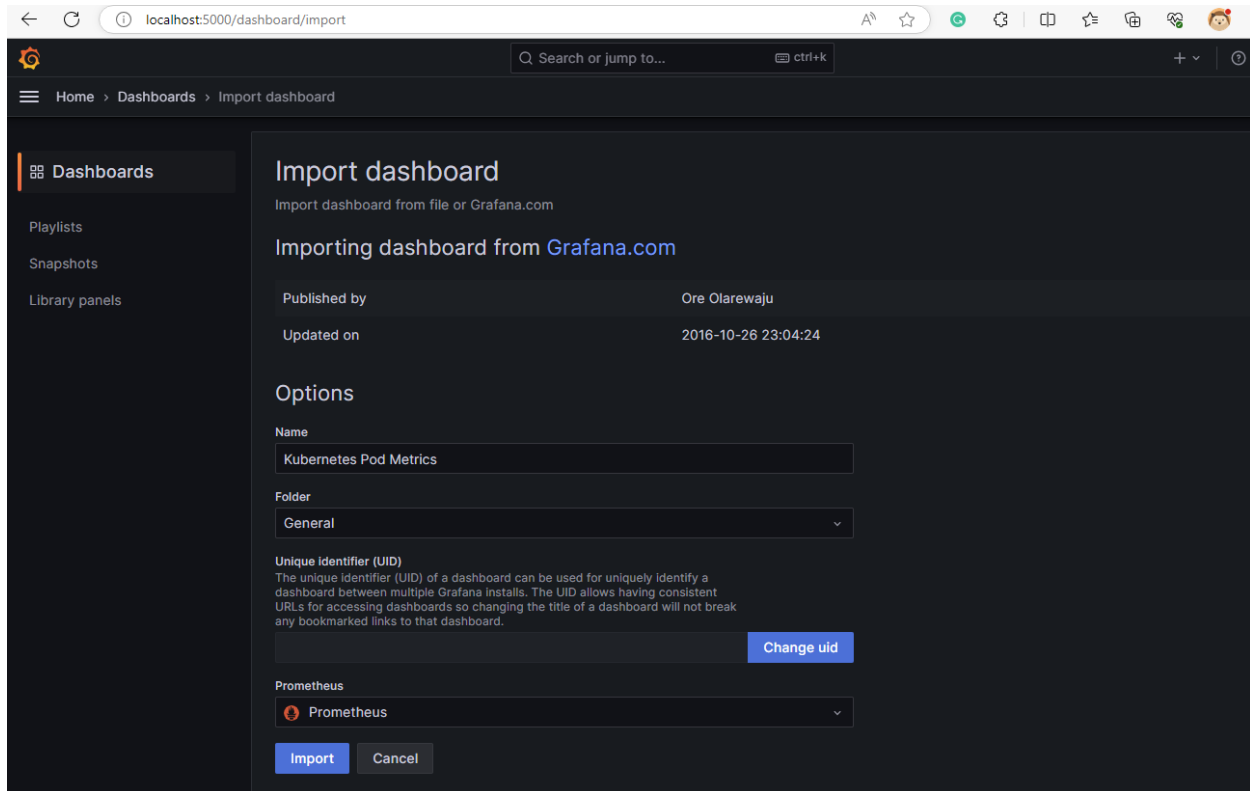
**Get this dashboard**

**Data source:**  
Prometheus 1.0.0

**Dependencies:**  
grafana 3.1.0 Graph (old) Singlestat Text

**Import the dashboard template:**  
[Copy ID to clipboard](#)

- Import to our dashboard



--- End documents ---