**Note: Because my NT license burn out credit so I have to create another account to do the assignment. So, you can see some images with the difference resource name.**

I. General

There are 3 repos:

* sd2488\_azure\_infrastructure: This repo store Terraform source code which is used provisioning Azure resources
  + The Terraform code is structured by modular
  + In this repos is also stored azure-pipeline yaml file which will use to deploy Terraform on Azure DevOps
* sd2488\_msa: This repo store Microservices applications, that included:
  + Backend – writing by NodeJS
  + Fronend - writing by JS
  + MongoDB this is note use because I will write yaml file to deploy MongoDB selfhost DB on the AKS
  + In this repo is also include yaml files which is created azure pipeline to build the application to the container images and apply trivy to scan the container images. If the container images successfully and valid then it will be pushed to Azure container registry.
* sd2488\_devops: This repo store Helm script and manifest to describe how service will be deployed to k8s (AKS). The Helm script will help to deploy k8s manifest

II. Infrastructure as Code

1. Terraform modules.

After running Terraform script, it was generated 2 resources groups:

[rg-pisharp-dev-eus](https://portal.azure.com/#@namtranp.onmicrosoft.com/resource/subscriptions/449821ae-3c94-47aa-9291-cc860d356600/resourceGroups/rg-pisharp-dev-eus): This is resources group is included all the resources we need to create, it is managed by us.

A screenshot of a chat

Description automatically generated

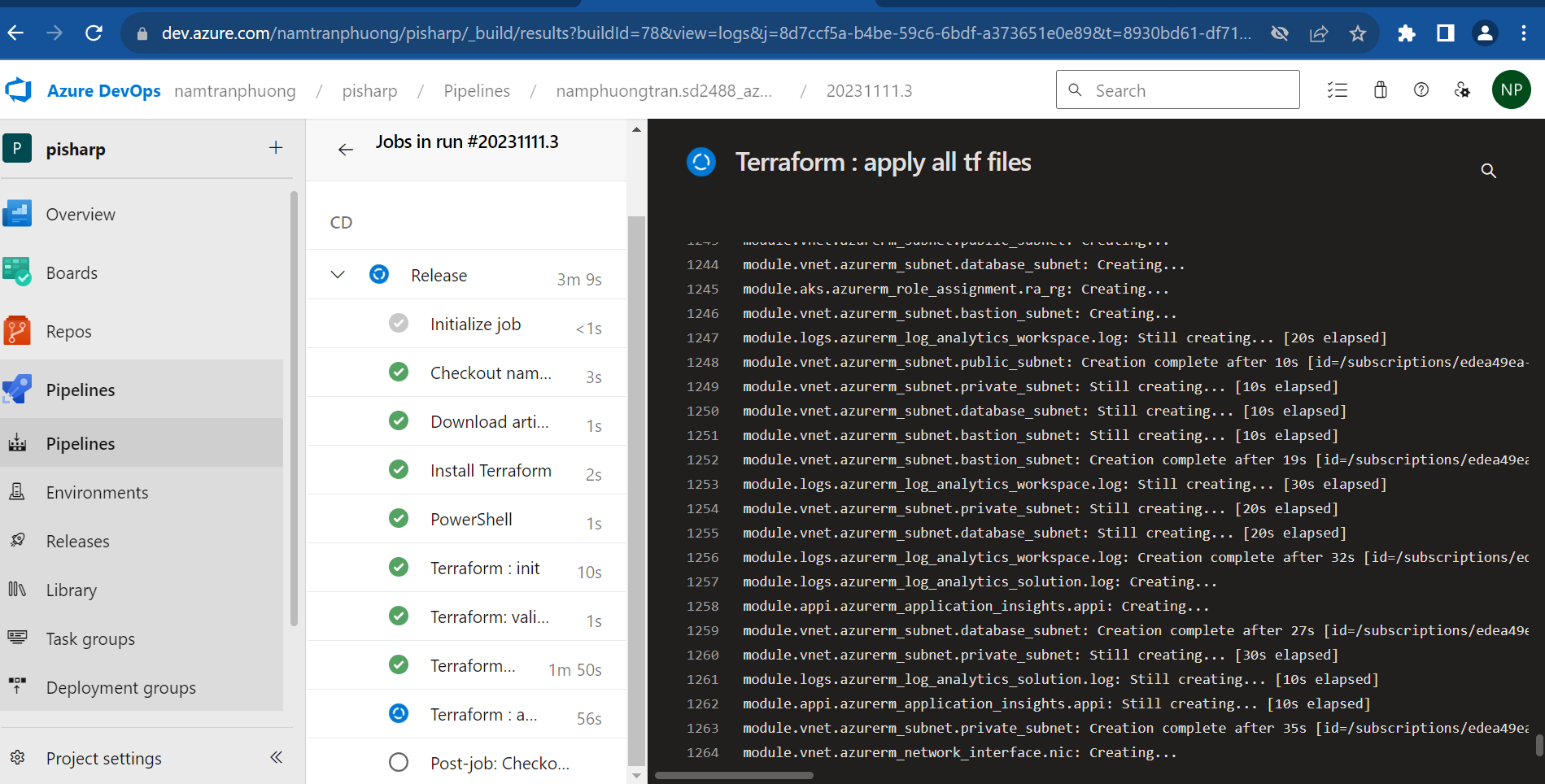
[rg-aks-managed-dev-eus](https://portal.azure.com/#@namtranp.onmicrosoft.com/resource/subscriptions/449821ae-3c94-47aa-9291-cc860d356600/resourceGroups/rg-aks-managed-dev-eus): This is resources group is included managed services which is controlled by Azure.

A screenshot of a computer

Description automatically generated

1. Setup pipeline to deploy Terraform.

Grant permission for the pipeline can run https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/permissions?view=azure-devops



1. Setup service connection for Azure container registry

A screenshot of a computer

Description automatically generated

1. Setup azure pipeline to build the images and apply Trivy to scan container images

A screenshot of a computer program

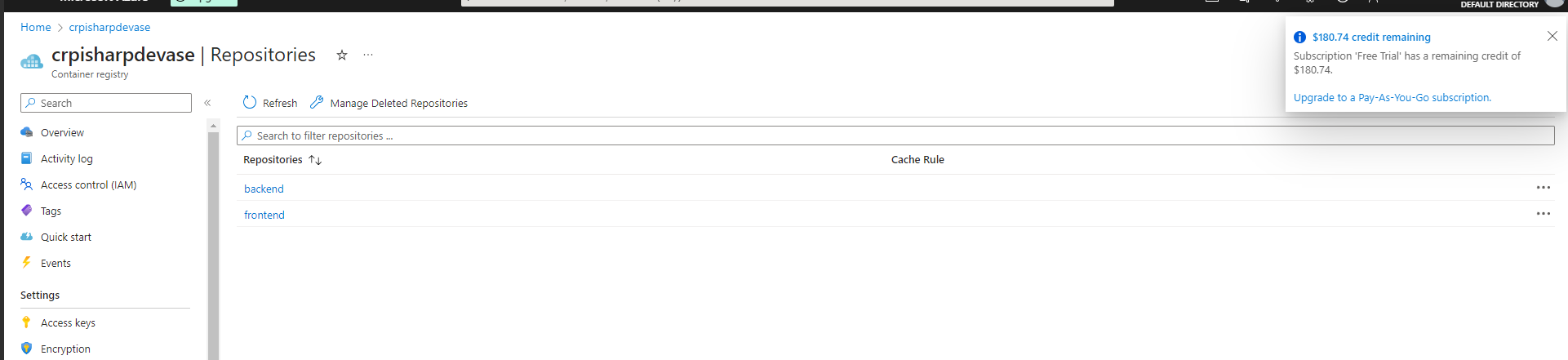
Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



III. Setup AKS cluster

1. Access to AKS cluster.

Installing kubectl client, kubelogin

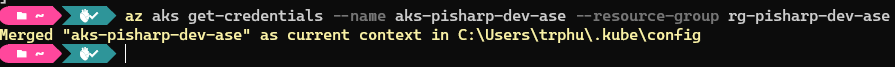
Then

Using az cli to login to Azure

az login

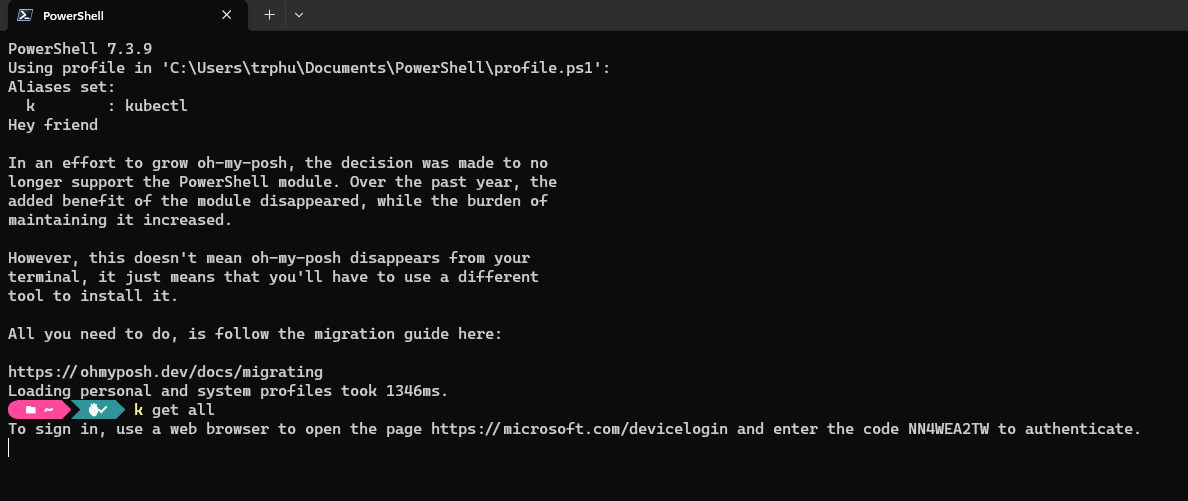
After provisioning the Infrastructure, I use Azure CLI to get credentials of the AKS cluster

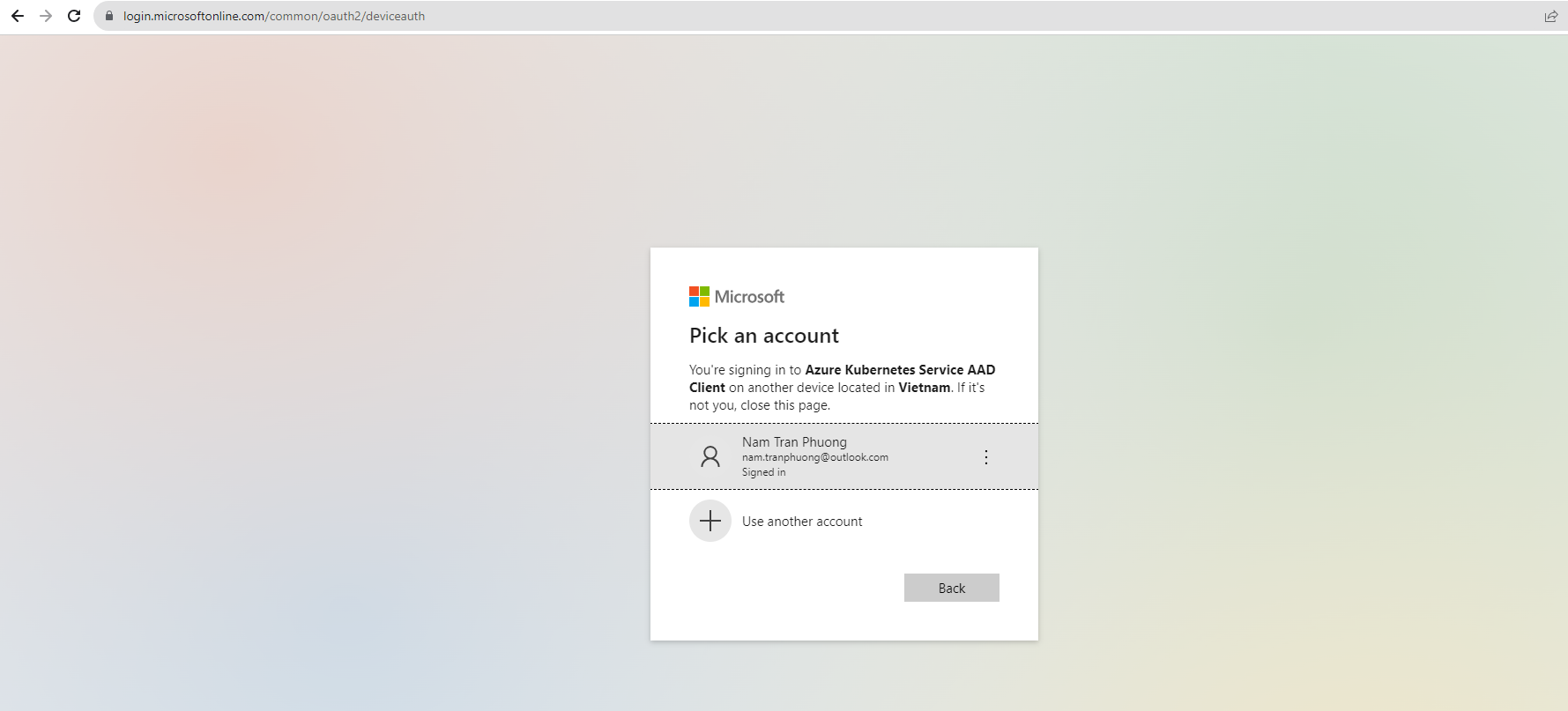
az aks get-credentials --name aks-pisharp-dev-ase --resource-group rg-pisharp-dev-ase

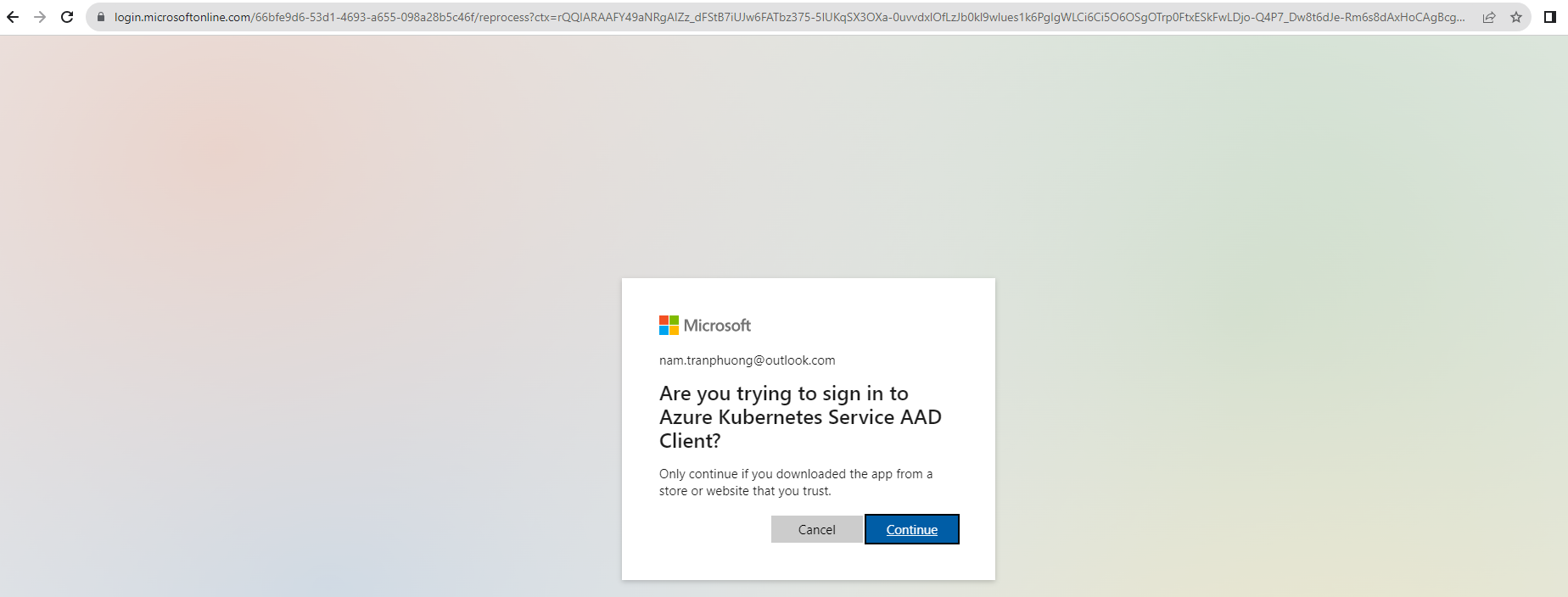


Please be noted that, for the first time, it requires to authenticate and authorize. So, when typing the command as below

k get all (k- shortcut of kubectl which I have already setup before)

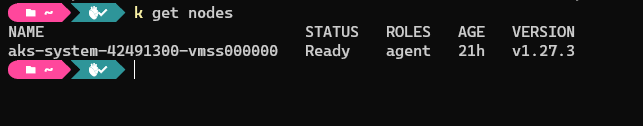




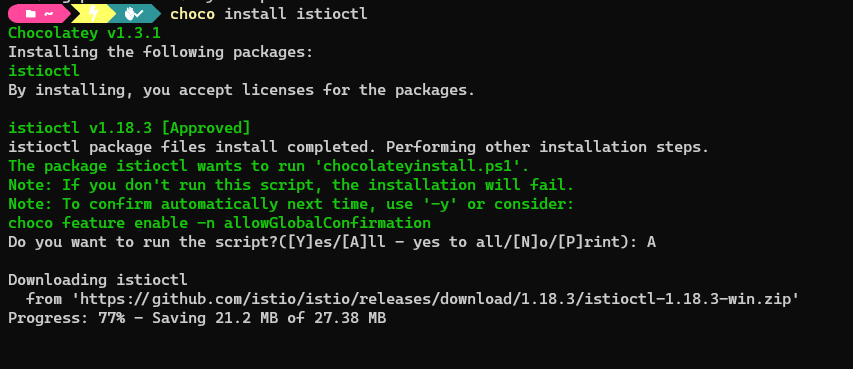




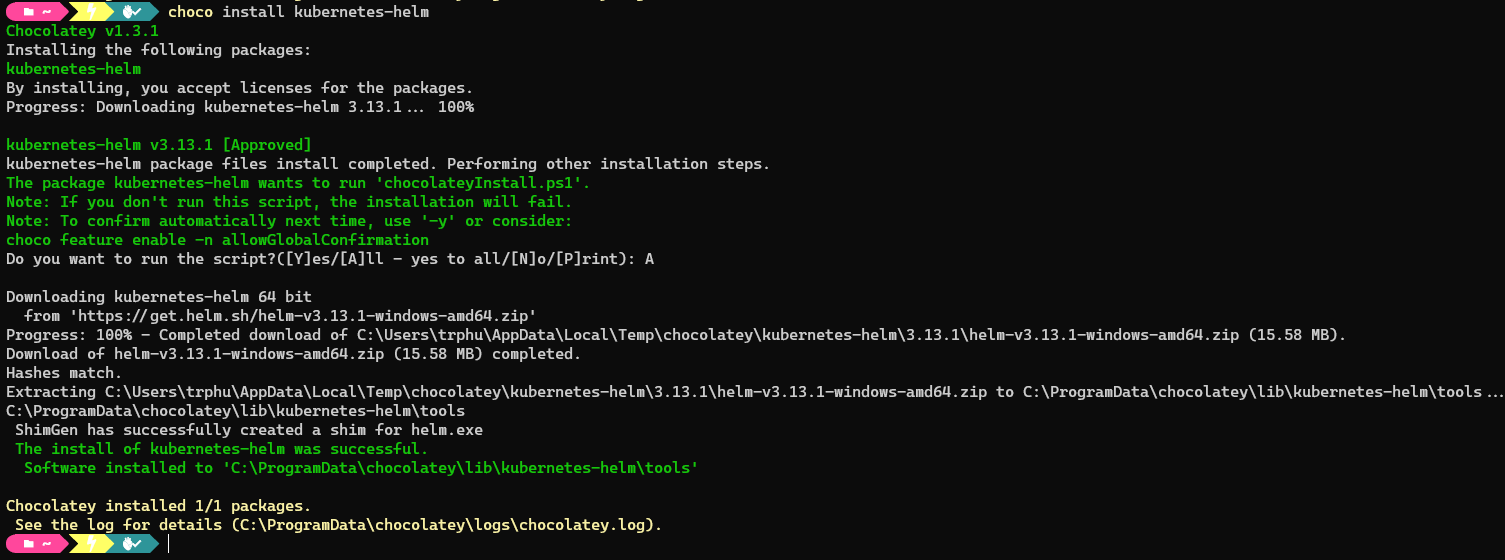
k get nodes



1. Setup Istio and Helm.
   1. Install Chocolatey
      1. Using PowerShell as Administrator
      2. Get-ExecutionPolicy
      3. Set-ExecutionPolicy AllSigned or Set-ExecutionPolicy Bypass -Scope Process
      4. Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
   2. Install IstioCtl
      1. choco install istioctl



* 1. Install Helm
     1. choco install kubernetes-helm

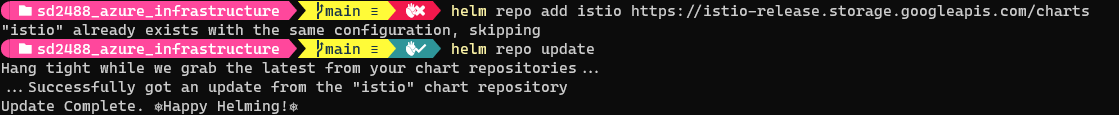


1. Install Istio

Configure the Helm repository:

helm repo add istio https://istio-release.storage.googleapis.com/charts

helm repo update



k get namespace -L istio-injection

k create namespace istio-system

A screen shot of a computer

Description automatically generated

A black screen with white text and pink and yellow arrows

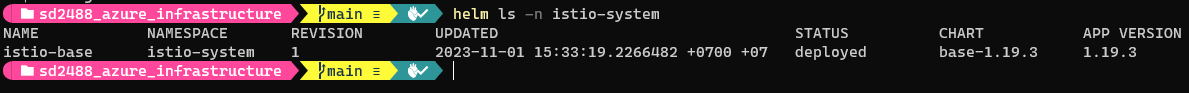
Description automatically generated

helm install istio-base istio/base -n istio-system --set defaultRevision=default

A screen shot of a computer

Description automatically generated

helm ls -n istio-system

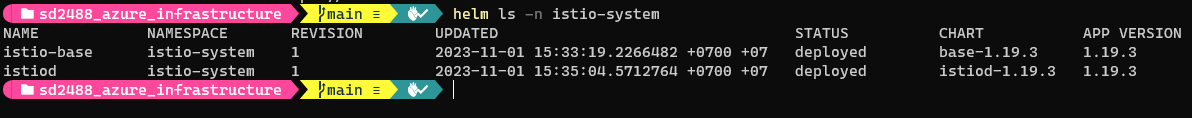


helm install istiod istio/istiod -n istio-system –wait

A screenshot of a computer

Description automatically generated

helm ls -n istio-system



helm status istiod -n istio-system

A screenshot of a computer

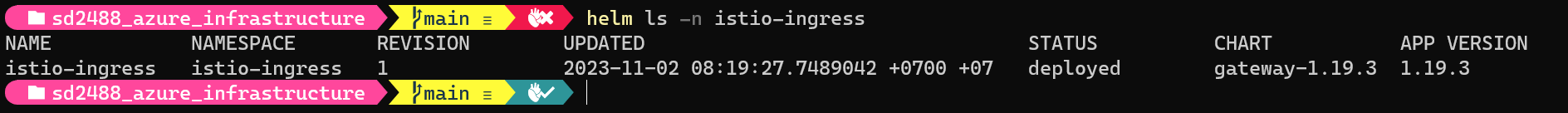
Description automatically generated

k create namespace istio-ingress

helm install istio-ingress istio/gateway -n istio-ingress –wait

A screenshot of a computer program

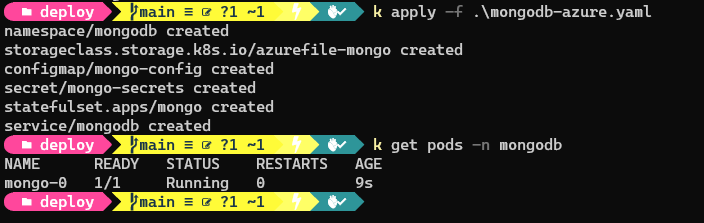
Description automatically generated



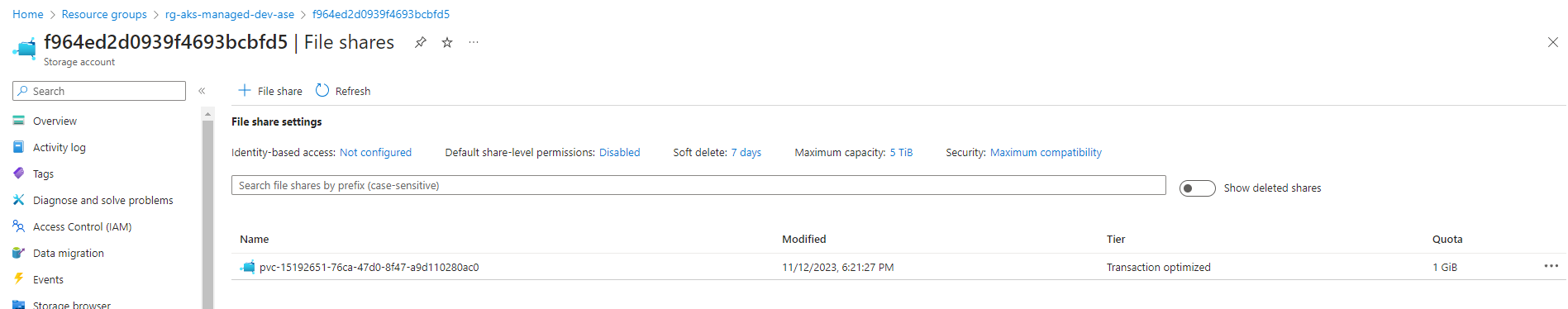
k get all -n istio-system

III. Deploy the k8s manifest application to AKS

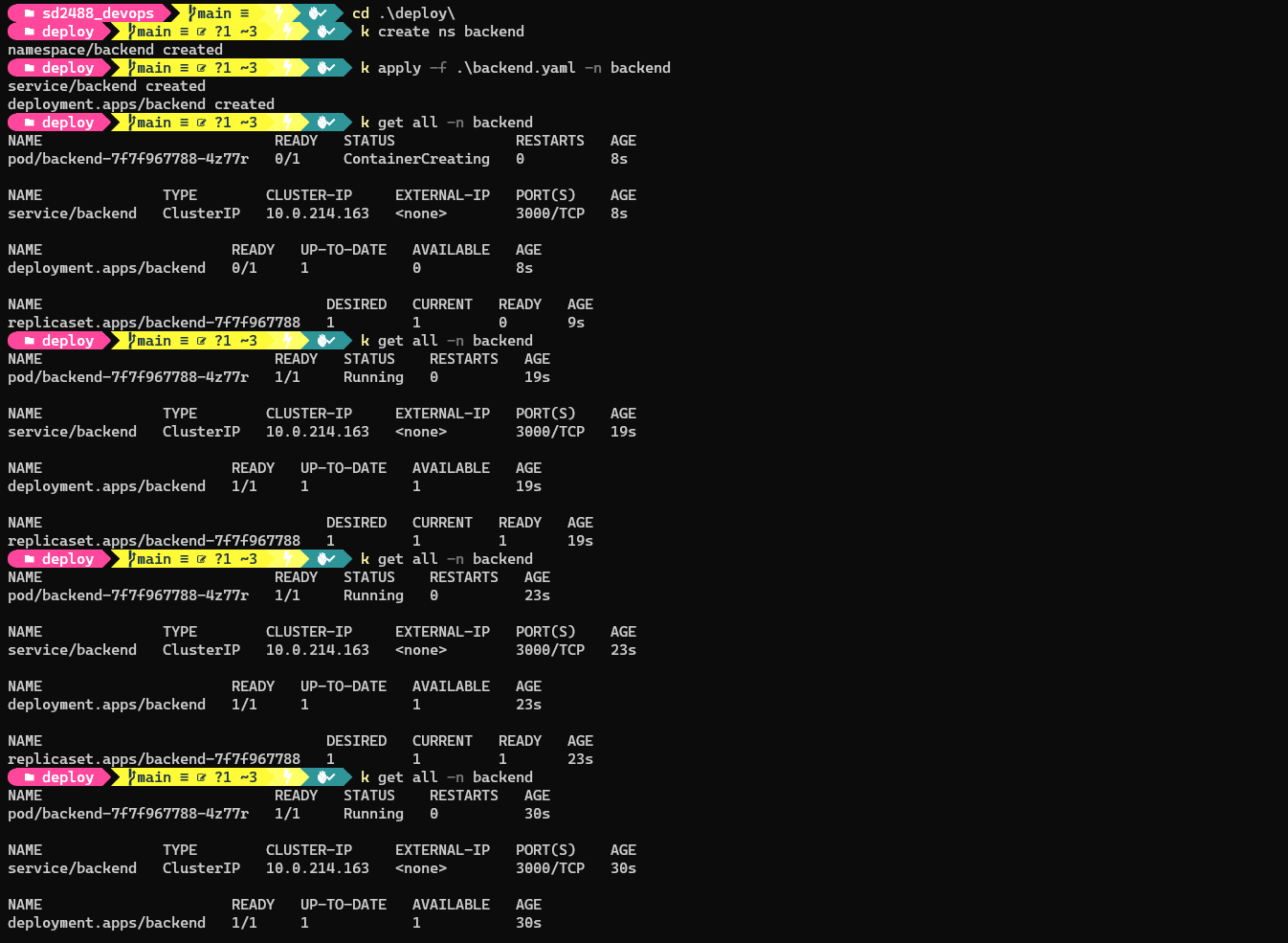
1. Deploy MongoDB
   1. Apply yaml file



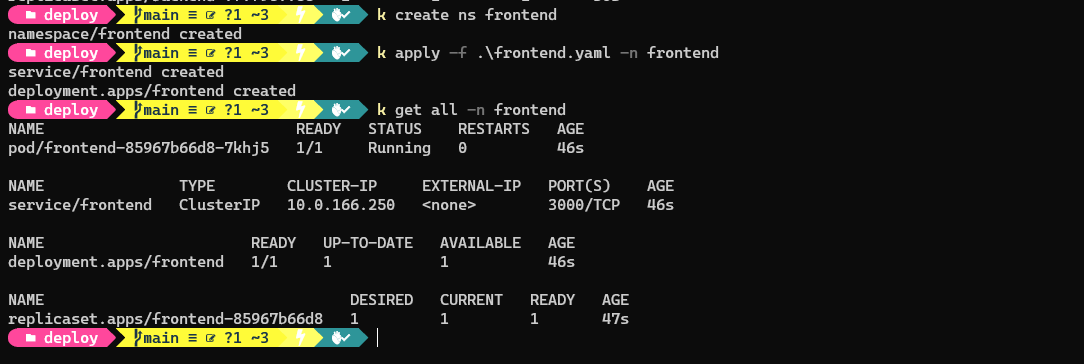
After deploying MongoDB successfully, we can check from Azure portal



1. Deploy Backend App

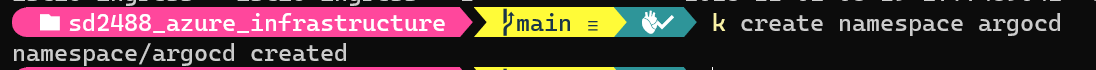


1. Deploy Frontend App



IV GitOps at Scale

k create namespace argocd

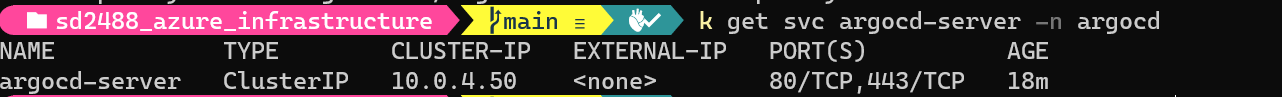


k apply -n argocd -f <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

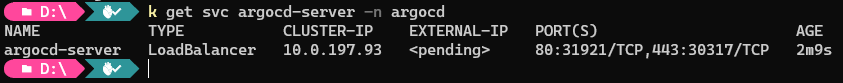
A screenshot of a computer program

Description automatically generated

k get svc argocd-server -n argocd



k edit svc argocd-server -n argocd



A screenshot of a computer

Description automatically generated