In this lab, you will learn how to connect to an existing AKS cluster in order to manage it and deploy your applications to it.

You can connect to both Kubernetes and Azure Kubernetes Services (AKS) clusters using a command line tool called kubectl.

Learning Objectives
In this lab, you will learn how to do the following:

Log in to the Azure CLI
Install the kubectl tool
Connect to an existing Azure Kubernetes Services (AKS) cluster
Clean up the AKS cluster

Login to azure
Az login -u $username -p $password

```
$ az aks list --query "[].{Name: name}"
[]
$
```

Create cluster .
Get version of Kubernetes supported.
$ az aks get-versions --location eastus

```
Terminal +
Get the versions available for creating a managed Kubernetes cluster

az account list
Get a list of subscriptions for the logged in account. (autogenerated)

az account set --subscription mysubscription
Set a subscription to be the current active subscription. (autogenerated)

https://docs.microsoft.com/en-US/cli/azure/aks#az_aks_get_versions
Read more about the command in reference docs
$ az aks get-versions --location eastus
{
  "id": "/subscriptions/fb326778-1e8f-455c-ac0b-f2e7b5a7c03f/providers/Microsoft.ContainerService/locations/eastus/orchestrators",
  "name": "default",
  "orchestrators": [
    {
      "default": null,
      "isPreview": null,
      "orchestratorType": "Kubernetes",
      "orchestratorVersion": "1.23.12",
      "upgrades": [
        {
          "isPreview": null,
          "orchestratorType": "Kubernetes",
          "orchestratorVersion": "1.23.15"
        },
        {
          "isPreview": null,
          "orchestratorType": "Kubernetes",
          "orchestratorVersion": "1.24.6"
        },
        {
          "isPreview": null,
          "orchestratorType": "Kubernetes",
          "orchestratorVersion": "1.24.9"
        }
      ]
    },
    {
      "default": null,
      "isPreview": null,
      "orchestratorType": "Kubernetes",
      "orchestratorVersion": "1.23.15",
      "upgrades": [
        {
```

Create cluster :
az aks create --resource-group $resource --name $aksName --kubernetes-version 1.24.9

```
Terminal
  "type": "Microsoft.ContainerService/locations/orchestrators"
}
$ az aks create --resource-group $resource --name $aksName --kubernetes-version 1.24.9

{
  "aadProfile": null,
  "addonProfiles": null,
  "agentPoolProfiles": [
    {
      "availabilityZones": null,
      "count": 3,
      "creationData": null,
      "currentOrchestratorVersion": "1.24.9",
      "enableAutoScaling": false,
      "enableEncryptionAtHost": false,
      "enableFips": false,
      "enableNodePublicIp": false,
      "enableUltraSsd": false,
      "gpuInstanceProfile": null,
      "hostGroupId": null,
      "kubeletConfig": null,
      "kubeletDiskType": "OS",
      "linuxOsConfig": null,
      "maxCount": null,
      "maxPods": 110,
      "minCount": null,
      "mode": "System",
      "name": "nodepool1",
      "nodeImageVersion": "AKSUbuntu-1804gen2containerd-202303.13.0",
      "nodeLabels": null,
      "nodePublicIpPrefixId": null,
      "nodeTaints": null,
      "orchestratorVersion": "1.24.9",
      "osDiskSizeGb": 128,
      "osDiskType": "Managed",
      "osSku": "Ubuntu",
      "osType": "Linux",
      "podSubnetId": null,
      "powerState": {
        "code": "Running"
      },
      "provisioningState": "Succeeded",
      "proximityPlacementGroupId": null,
      "scaleDownMode": null,
      "scaleSetEvictionPolicy": null,
```

```
$
$ az aks list --query "[].{Name: name}"
[
  {
    "Name": "aks124138602"
  }
]
$
```

Introducing the kubectl Tool
According to the Kubernetes documentation, "the Kubernetes command-line tool, kubectl, allows you to run commands against Kubernetes clusters. You can use kubectl to deploy applications, inspect and manage cluster resources, and view logs."

kubectl works both with Kubernetes clusters and Azure Kubernetes Services (AKS) clusters.

Our lab environment in based on Linux, so we will install the Linux version. Azure CLI gives you a convenient command to install kubectl.

Use the following CLI command to install kubectl on your testing environment:

az aks install-cli

This command will download the kubectl files and install the tool on your environment.

```
$ az aks install-cli
Downloading client to "/usr/local/bin/kubectl" from "https://storage.googleapis.com/kubernetes-release/release/v1.26.3/bin/linux/amd64/kubectl"
Please ensure that /usr/local/bin is in your search PATH, so the `kubectl` command can be found.
Downloading client to "/tmp/tmphewbgq9t/kubelogin.zip" from "https://github.com/Azure/kubelogin/releases/download/v0.0.28/kubelogin.zip"
Please ensure that /usr/local/bin is in your search PATH, so the `kubelogin` command can be found.
$
```

In the next step, we will connect to our AKS cluster using kubectl
Connecting to Our AKS Cluster
Now it's time to connect to our AKS cluster. I order to do so, you need to first get the cluster credentials stored on your testing environment and then use the credentials to run kubectl command. The credentials are saved/merged into the .kube/config file so then kubectl can use them.

First, let's get the AKS credentials using this command:

az aks get-credentials --resource-group $resource --name $aksName

```
$ az aks get-credentials --resource-group $resource --name $aksName
Merged "aks124138602" as current context in /root/.kube/config
$
```

Now, use the following command to make sure kubectl can connect to AKS. This command returns the list of nodes in your cluster and will only work if the correct credentials are available:
Confirm that you can see a list of AKS nodes in the command output. This list includes the following node properties:

NAME: node name
STATUS: provisioning status, for example READY
ROLES: shows if the node is a master node
AGE: the node age from when it was provisioned
VERSION: Kubernetes version, for example 1.24.0

Now let's clean up the cluster in the next step.
Deleting the AKS Cluster
The following command will clean up the new AKS cluster:

az aks delete --name $aksName --resource-group $resource

```
$ az aks delete --name $aksName --resource-group $resource
Are you sure you want to perform this operation? (y/n): y
$
```

Now use the list command again to get all the AKS clusters in your temporary Azure subscription:

```
$ az aks list --query "[].{Name: name}"
[]
$
```