

Microsoft ADC Cybersecurity Skilling Program

Week 8 Assignment 15

Student Name: DEBORAH BINYANYA NYATICHI

Student ID: ADC-CSS02-25051

Introduction

In this lab, we focus on **Configuring and Securing Azure Container Registry (ACR) and Azure Kubernetes Service (AKS)**, which are fundamental components for deploying and managing containerized applications in the cloud. As part of the AZ-500: Microsoft Azure Security Technologies course, this lab provides hands-on experience in building secure and scalable container environments on Azure.

The lab begins by using a **Dockerfile to build a container image**, which is then stored in **Azure Container Registry**—a private registry for managing container images. We then move on to configuring **Azure Kubernetes Service (AKS)** to orchestrate and run containerized workloads. Emphasis is placed on securing access to both the registry and the Kubernetes cluster, ensuring applications can be accessed securely from within the cluster (internal access) and by external users (external access).

By completing this lab, we gain practical knowledge in securing the container lifecycle from image creation to deployment, along with applying Azure-native security best practices to protect containerized applications in production environments.

Tasks Completed

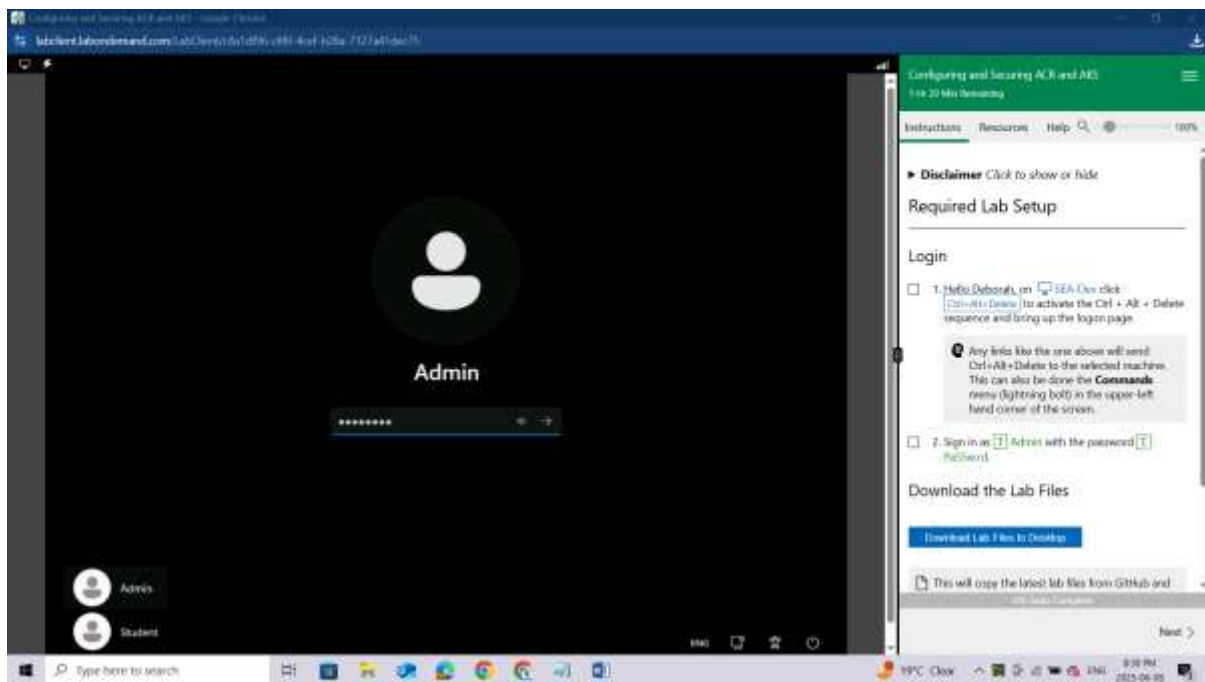
Required Lab Setup

Login

Hello Deborah, on [SEA-Dev](#) click [Ctrl+Alt+Delete](#) to activate the Ctrl + Alt + Delete sequence and bring up the logon page.

*Any links like the one above will send Ctrl+Alt+Delete to the selected machine. This can also be done the **Commands** menu (lightning bolt) in the upper-left hand corner of the screen.*

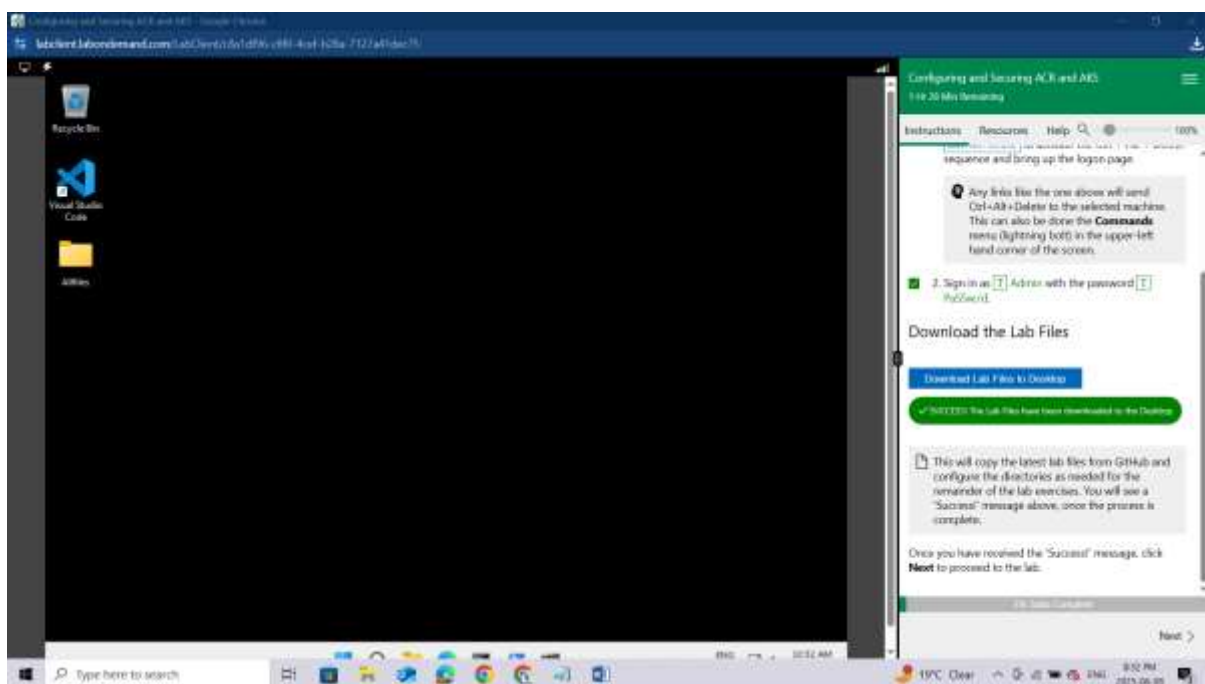
Sign in as **Admin** with the password **Pa55w.rd**.



Download the Lab Files

This will copy the latest lab files from GitHub and configure the directories as needed for the remainder of the lab exercises. You will see a 'Success!' message above, once the process is complete.

Once you have received the 'Success!' message, click **Next** to proceed to the lab.



For Task 3 steps 1-8 you will need to use this command in a Bash session within the Cloud Shell pane to create the Azure Kubernetes Service Cluster

```
Bash
az aks create --name MyKubernetesCluster --resource-group AZ500LAB09 --location eastus --no-ssh-key --node-vm-size Standard_D2s_v3 --nodepool-name agentpool --
```

```
node-count 1 --vm-set-type AvailabilitySet --network-plugin azure --dns-name-prefix MyKubernetesCluster-dns
```

Before running Task 4 step 2 you will need to manually create a virtual network. In the Search Resources, Services, and Docs text box at the top of the Azure portal page, type Virtual networks and press the Enter key. Click the Create button, and once that opens use the AZ500LAB09 Resource Group and name it AZ500LAB09-vnet. Once this is done you can use the script in Task 4 step 2.

Lab 04: Configuring and Securing ACR and AKS

Student lab manual

Lab scenario

You have been asked to deploy a proof of concept with Azure Container Registry and Azure Kubernetes Service. Specifically, the proof of concept should demonstrate:

- Using Dockerfile to build an image.
- Using Azure Container Registry to store images.
- Configuring an Azure Kubernetes Service.
- Securing and accessing container applications both internally and externally.

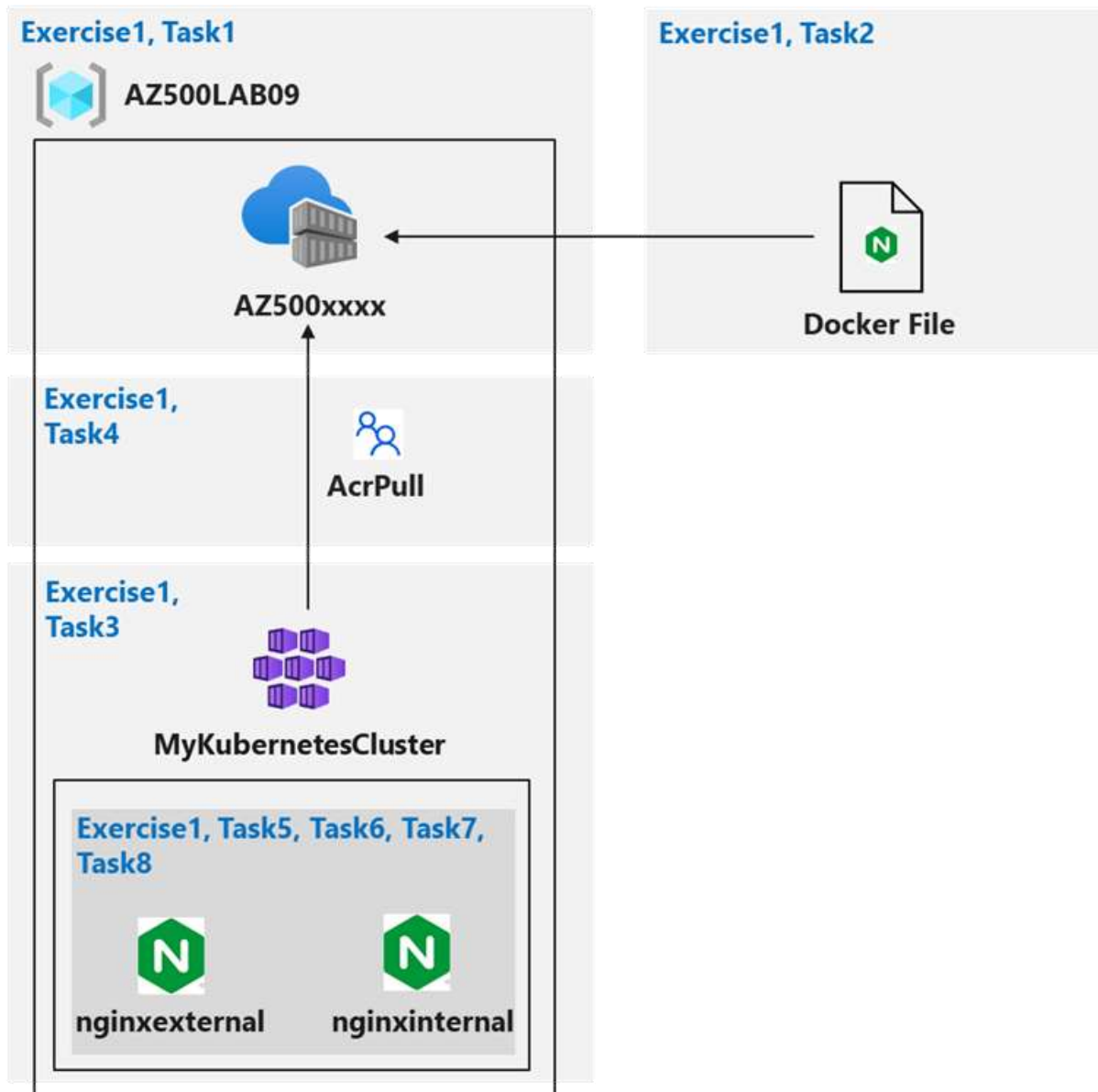
For all the resources in this lab, we are using the **East US** region. Verify with your instructor this is the region to use for class.

Lab objectives

In this lab, you will complete the following exercise:

- Exercise 1: Configuring and Securing ACR and AKS

Configuring and Securing ACR and AKS diagram



Instructions

Lab files:

- \Allfiles\Labs\09\nginxexternal.yaml
- \Allfiles\Labs\09\nginxinternal.yaml

Exercise 1: Configuring and Securing ACR and AKS

For all the resources in this lab, we are using the **East (US)** region. Verify with your instructor this is region to use for you class.

In this exercise, you will complete the following tasks:

- Task 1: Create an Azure Container Registry
- Task 2: Create a Dockerfile, build a container and push it to Azure Container Registry
- Task 3: Create an Azure Kubernetes Service cluster
- Task 4: Grant the AKS cluster permissions to access the ACR

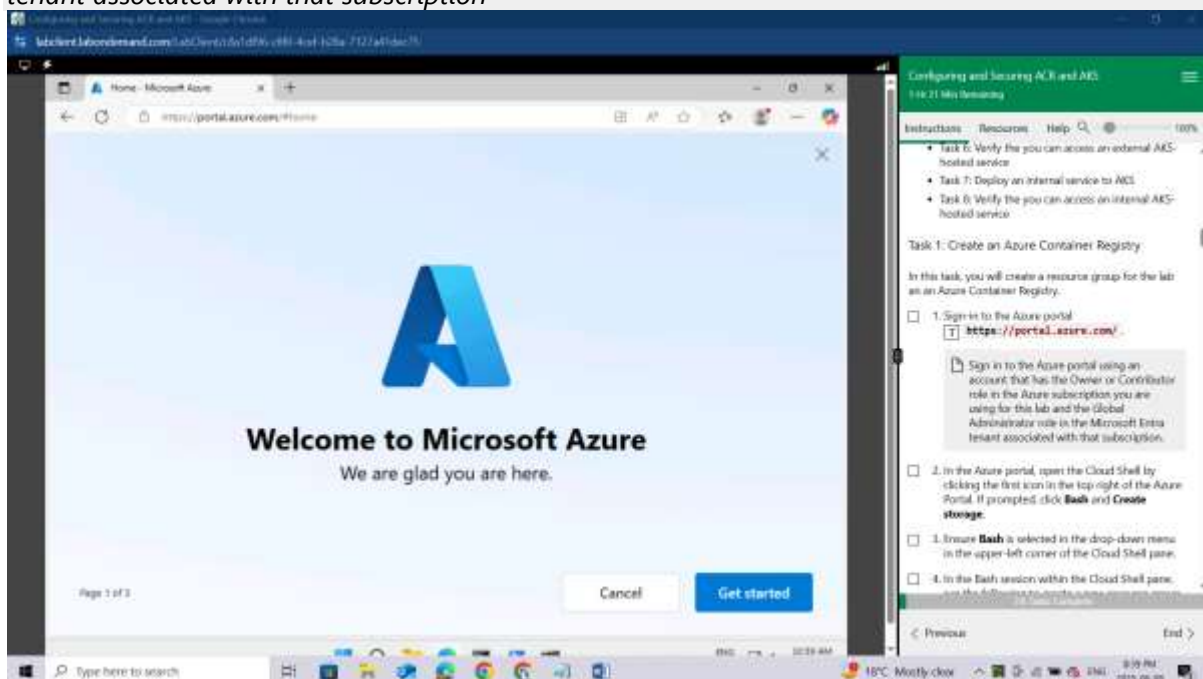
- Task 5: Deploy an external service to AKS
- Task 6: Verify the you can access an external AKS-hosted service
- Task 7: Deploy an internal service to AKS
- Task 8: Verify the you can access an internal AKS-hosted service

Task 1: Create an Azure Container Registry

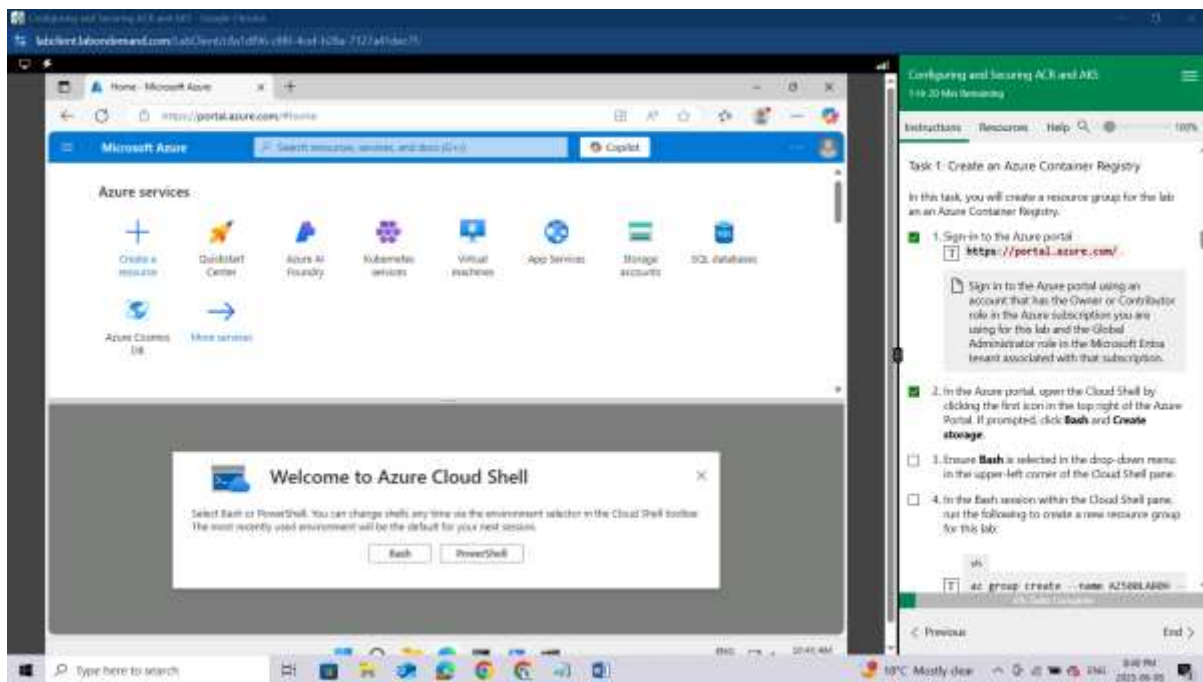
In this task, you will create a resource group for the lab an Azure Container Registry.

Sign-in to the Azure portal <https://portal.azure.com/>.

Sign in to the Azure portal using an account that has the Owner or Contributor role in the Azure subscription you are using for this lab and the Global Administrator role in the Microsoft Entra tenant associated with that subscription



In the Azure portal, open the Cloud Shell by clicking the first icon in the top right of the Azure Portal. If prompted, click **Bash** and **Create storage**.



Microsoft Azure

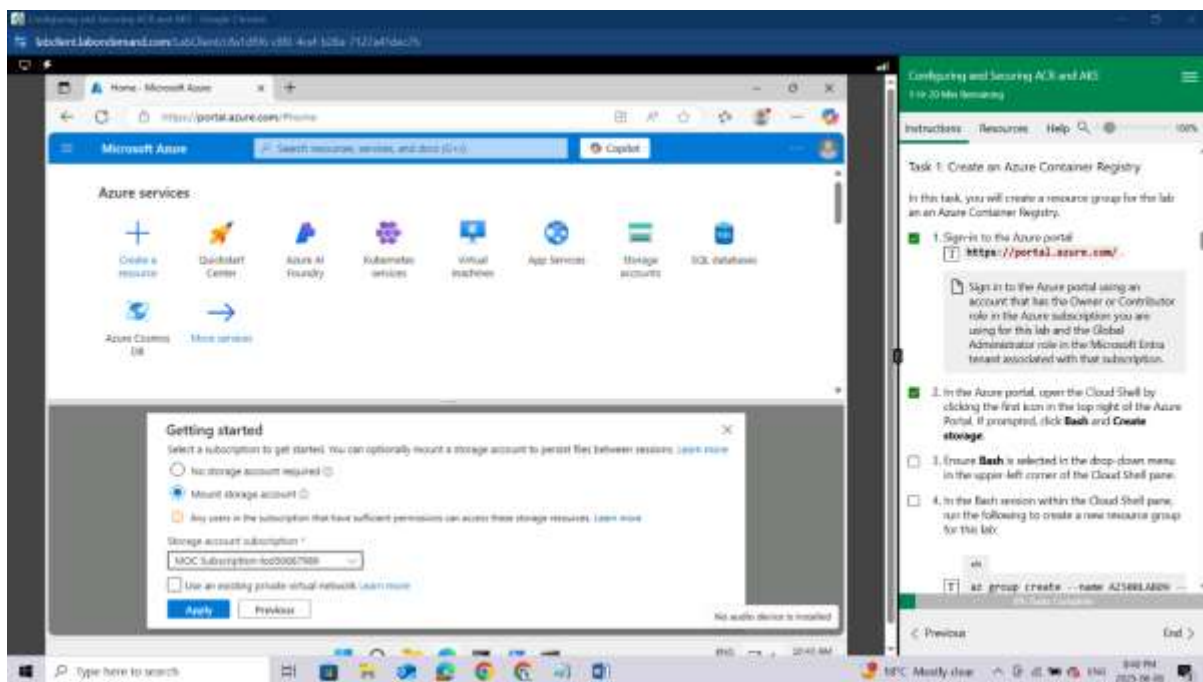
Search resources, services, and docs (0/10)

Azure services:

- Create a resource
- Dashboard
- Azure AI Foundry
- Subscriptions
- Virtual machines
- App Services
- Storage accounts
- SQL databases
- Azure Cosmos DB
- More services

Welcome to Azure Cloud Shell

Select Bash or PowerShell. You can change shells any time via the environment selector in the Cloud Shell toolbar. The most recently used environment will be the default for your next session.



Microsoft Azure

Search resources, services, and docs (0/10)

Azure services:

- Create a resource
- Dashboard
- Azure AI Foundry
- Subscriptions
- Virtual machines
- App Services
- Storage accounts
- SQL databases
- Azure Cosmos DB
- More services

Getting started

Select a subscription to get started. You can optionally mount a storage account to persist files between sessions. [Learn more](#)

☐ No storage account required

☒ Mount storage account

Any users in the subscription that have sufficient permissions can access these storage resources. [Learn more](#)

Storage account subscription *

☐ Use an existing private virtual network. [Learn more](#)

No audio device is installed

Configuring and Securing ACI and AKS

1 of 20 Min Remaining

Instructions Resources Help 100%

Task 1: Create an Azure Container Registry

In this task, you will create a resource group for the lab and an Azure Container Registry.

1. Sign in to the Azure portal <https://portal.azure.com/>.
Sign in to the Azure portal using an account that has the Owner or Contributor role in the Azure subscription you are using for this lab and the Global Administrator role in the Microsoft Entra tenant associated with that subscription.
2. In the Azure portal, open the Cloud Shell by clicking the first icon in the top right of the Azure Portal. If prompted, click **Back** and **Create storage**.
3. Ensure **Bash** is selected in the drop-down menu in the upper-left corner of the Cloud Shell pane.
4. In the Bash session within the Cloud Shell pane, run the following to create a new resource group for this lab:

```
az group create --name AZ588LAB01
```


Configuring and Securing ACI and AKS

1 of 10 Modules

Instructions: Resources Help 100%

Task 1: Create an Azure Container Registry

In this task, you will create a resource group for the lab on an Azure Container Registry.

1. Sign in to the Azure portal: <https://portal.azure.com/>.

Sign in to the Azure portal using an account that has the Owner or Contributor role in the Azure subscription you are using for this lab and the Global Administrator role in the Microsoft Entra tenant associated with that subscription.

2. In the Azure portal, open the Cloud Shell by clicking the first icon in the top right of the Azure Portal. If prompted, click **Back** and **Create storage**.
3. Ensure **Bash** is selected in the drop-down menu in the upper left corner of the Cloud Shell pane.
4. In the Bash session within the Cloud Shell pane, run the following to create a new resource group for this lab:

```
az group create --name AZ1001LAB01
```

Previous End

Home - Microsoft Azure

https://portal.azure.com/Phone

Microsoft Azure Search resources, services, and docs (0/0)

Azure services:

Create a resource Dashboard Azure AI Foundry Subscriptions Virtual machines App Services Storage accounts SQL databases Azure Cosmos DB More services

Mount storage account

Azure Cloud Shell requires a storage account with Azure file shares to persist files. Select an option below to mount a storage account. [Learn more](#)

☐ Select existing storage account

☐ We will create a storage account for you

☒ I want to create a storage account

Next Previous

Configuring and Securing ACI and AKS

2 of 10 Modules

Instructions: Resources Help 100%

3. Ensure **Bash** is selected in the drop-down menu in the upper left corner of the Cloud Shell pane.
4. In the Bash session within the Cloud Shell pane, run the following to create a new resource group for this lab:

```
az group create --name AZ1001LAB01
```

5. In the Bash session within the Cloud Shell pane, run the following to verify the resource group was created:

```
az group list --query "[?name=='AZ1001LAB01']"
```

6. In the Bash session within the Cloud Shell pane, run the following to create a new Azure Container Registry (ACR) instance. (The name of the ACR must be globally unique):

```
az acr create --resource-group AZ1001LAB01
```

7. In the Bash session within the Cloud Shell pane, run the following to confirm that the new ACR:

Home - Microsoft Azure

https://portal.azure.com/Phone

Microsoft Azure Search resources, services, and docs (0/0)

Azure services:

Create a resource Dashboard Azure AI Foundry Subscriptions Virtual machines App Services Storage accounts SQL databases Azure Cosmos DB More services

Create storage account

Subscription: MDC Subscription-8d388d7985

Resource group: ResourceGroup

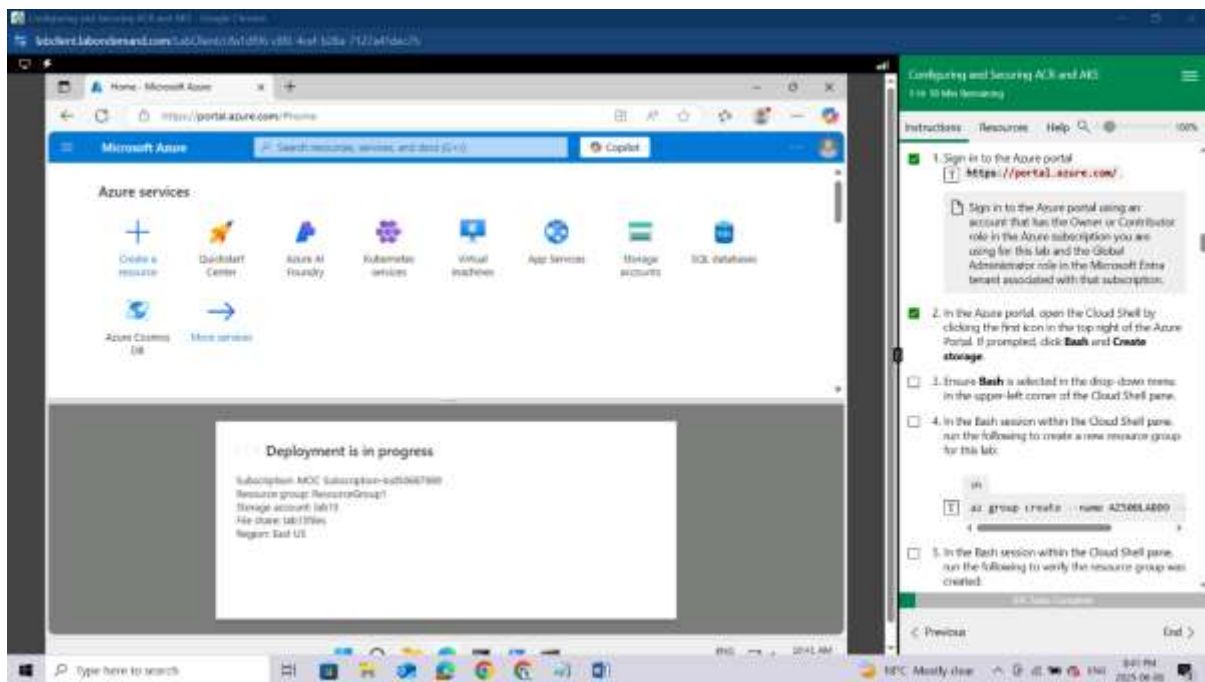
Region: (US) East US

Storage account name: lab1001lab01

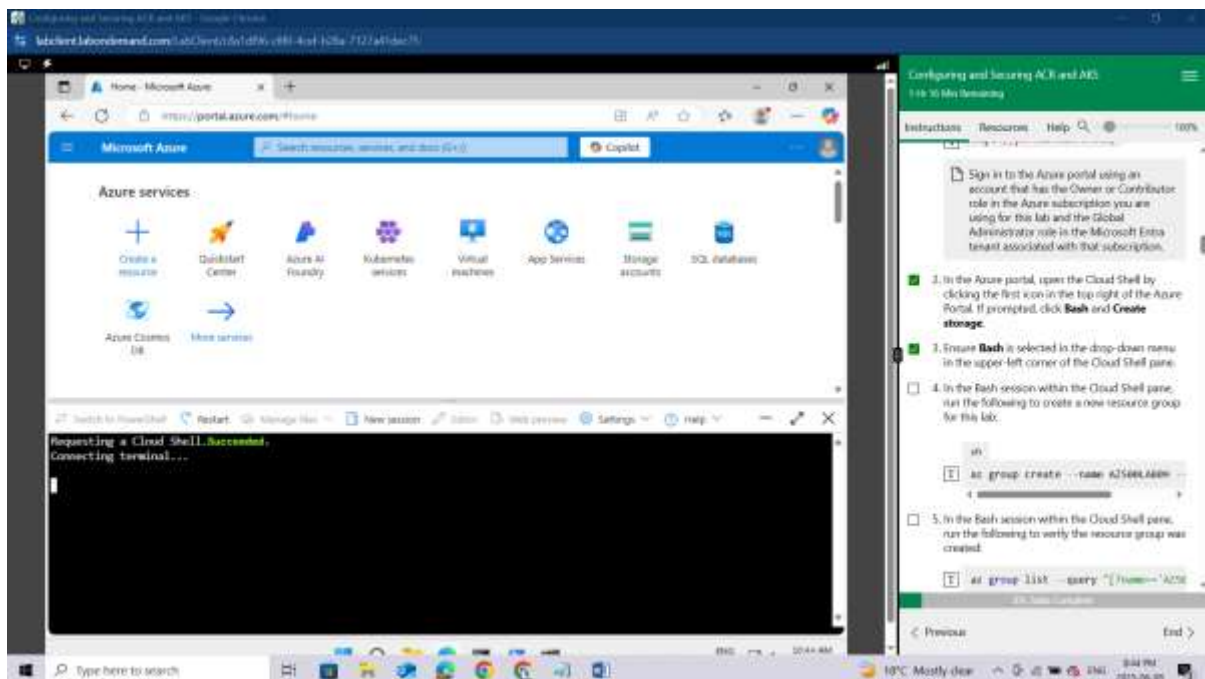
File share: lab1001lab01

Create a resource group

Create Previous

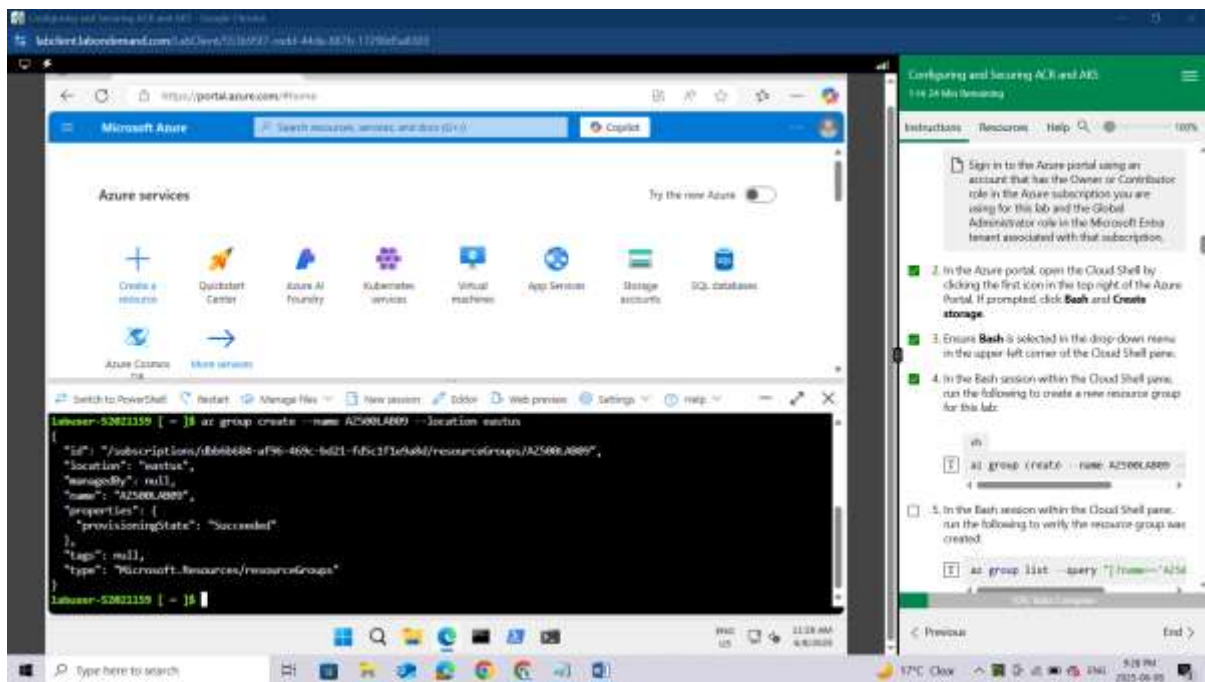


Ensure **Bash** is selected in the drop-down menu in the upper-left corner of the Cloud Shell pane.



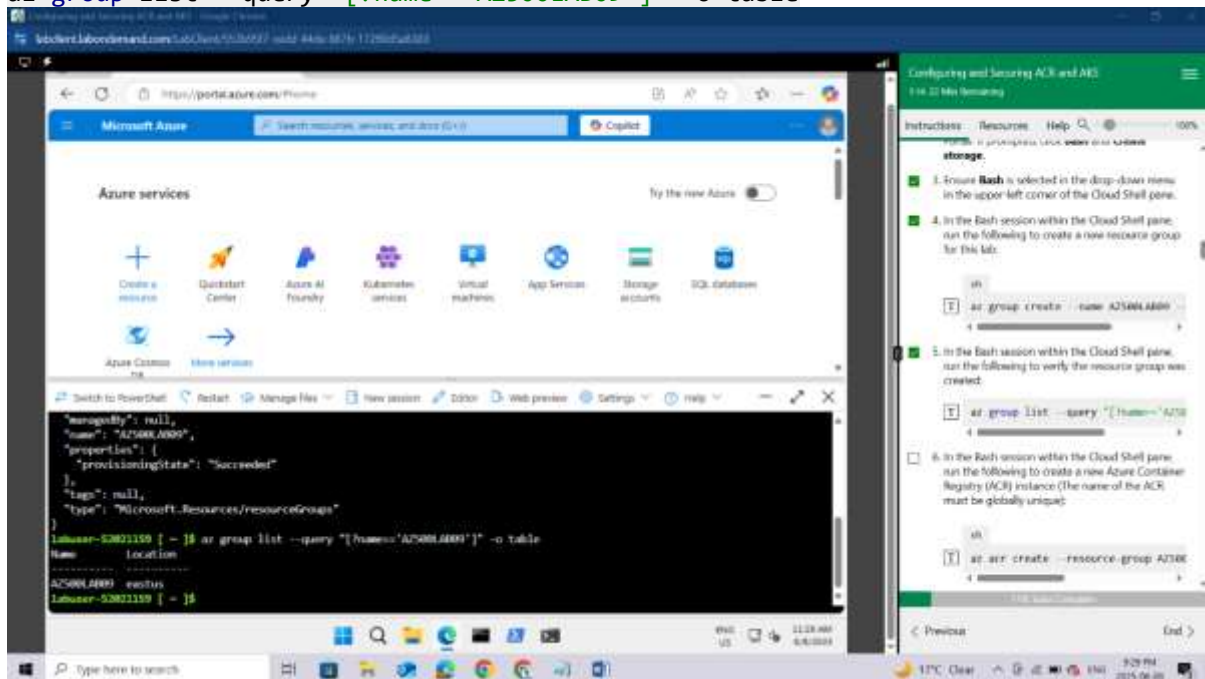
In the Bash session within the Cloud Shell pane, run the following to create a new resource group for this lab:

```
sh
az group create --name AZ500LAB09 --location eastus
```

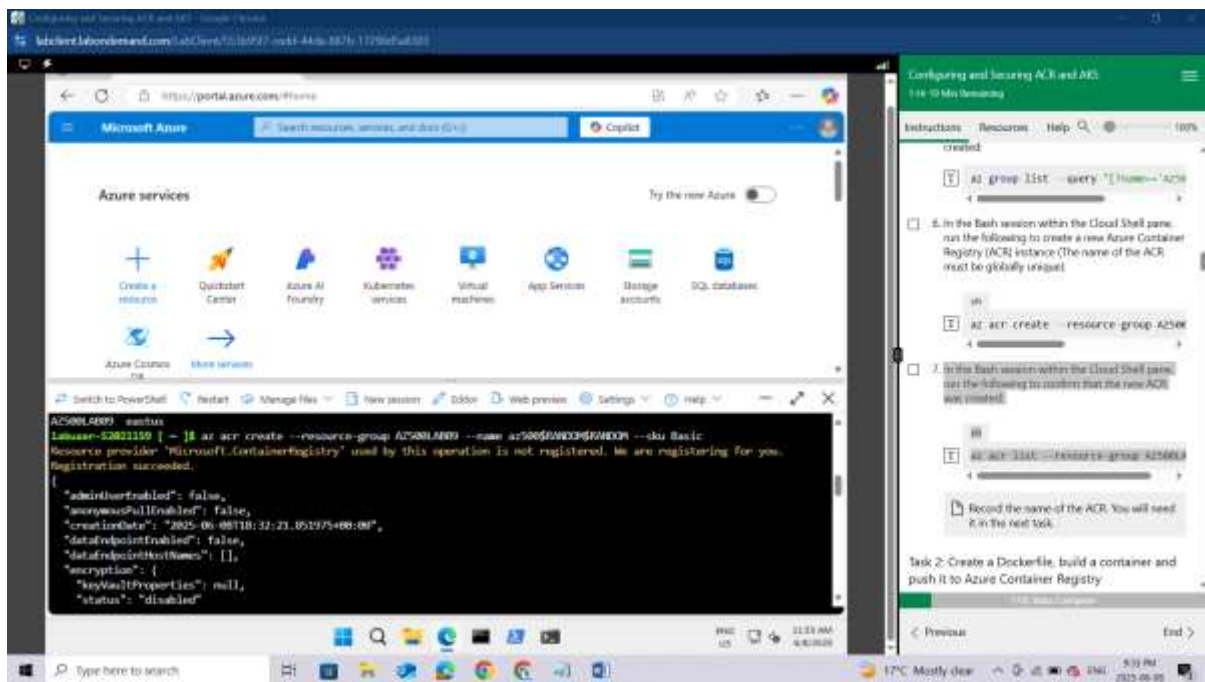
In the Bash session within the Cloud Shell pane, run the following to verify the resource group was created:

```
az group list --query "[?name=='AZ500LAB09']" -o table
```



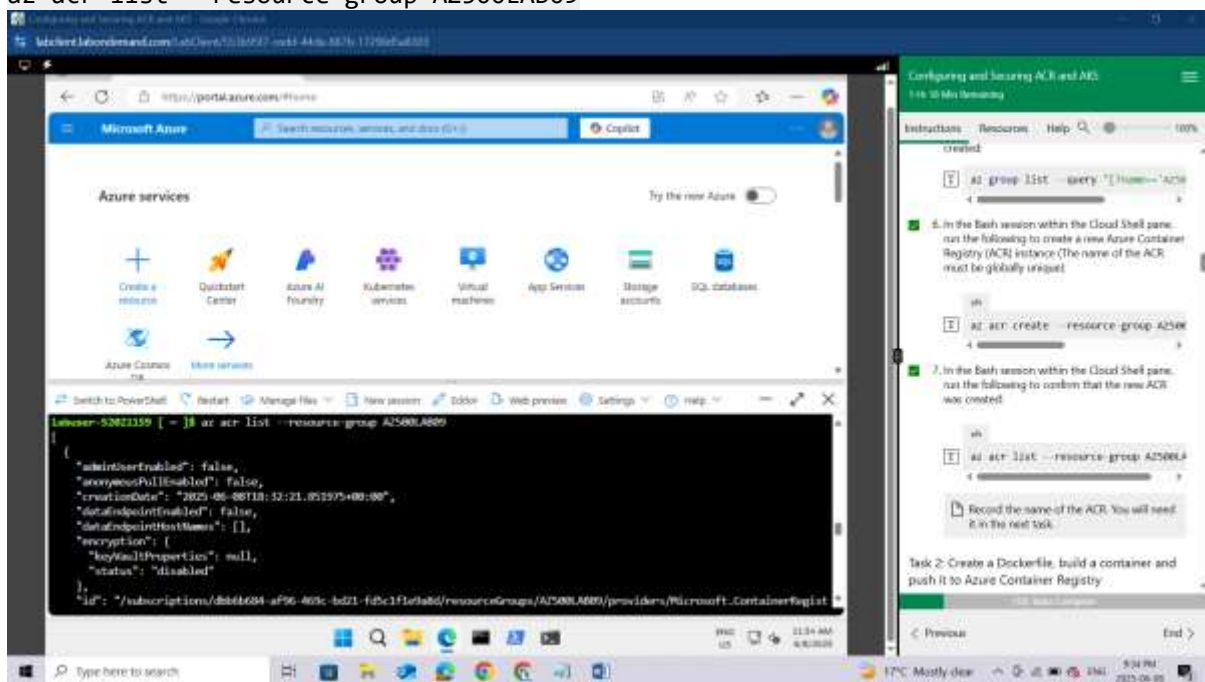
In the Bash session within the Cloud Shell pane, run the following to create a new Azure Container Registry (ACR) instance (The name of the ACR must be globally unique):

```
sh
az acr create --resource-group AZ500LAB09 --
```



In the Bash session within the Cloud Shell pane, run the following to confirm that the new ACR was created:

```
sh
az acr list --resource-group AZ500LAB09
```



Task 2: Create a Dockerfile, build a container and push it to Azure Container Registry

In this task, you will create a Dockerfile, build an image from the Dockerfile, and deploy the image to the ACR.

In the Bash session within the Cloud Shell pane, run the following to create a Dockerfile to create an Nginx-based image:

```
sh
echo FROM nginx > Dockerfile
```

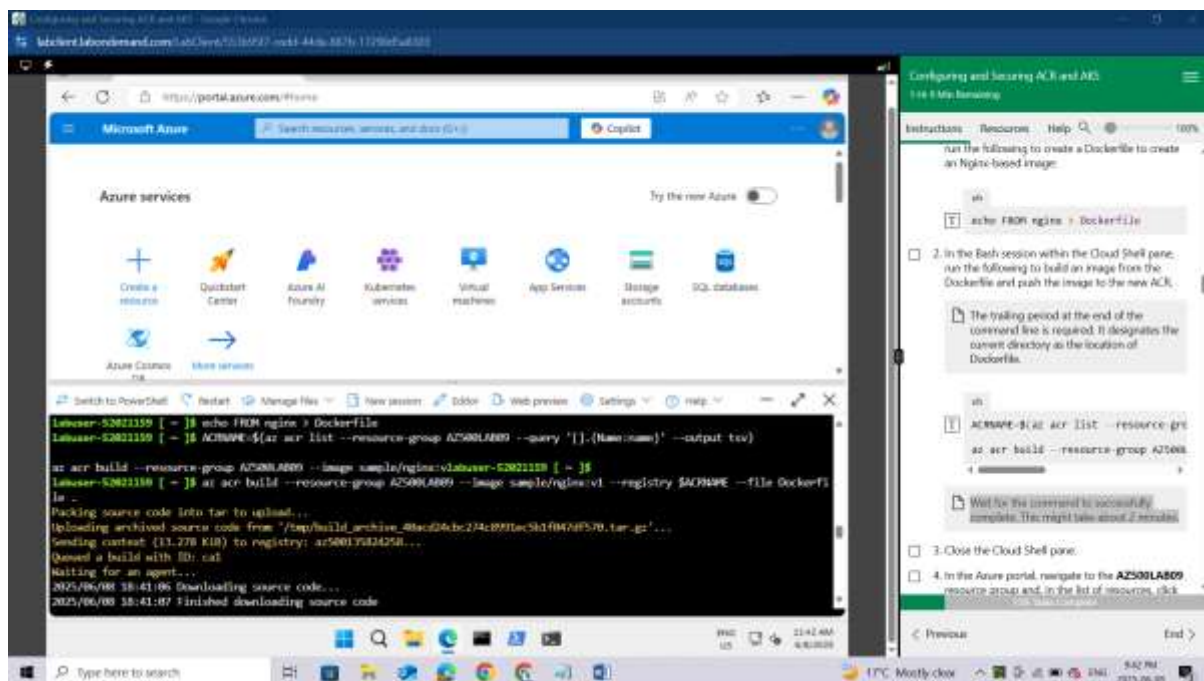
In the Bash session within the Cloud Shell pane, run the following to build an image from the Dockerfile and push the image to the new ACR.

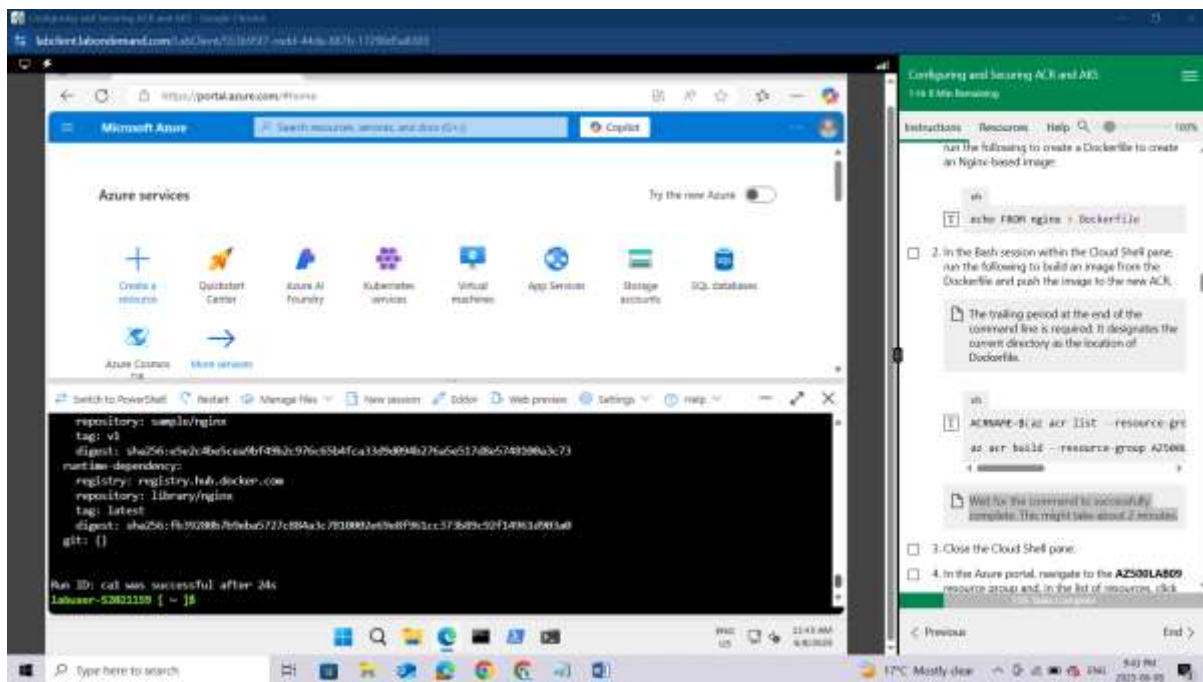
The trailing period at the end of the command line is required. It designates the current directory as the location of Dockerfile.

```
sh
ACRNAME=$(az acr list --resource-group AZ500LAB09 --query '[].{Name:name}' --
output tsv)
```

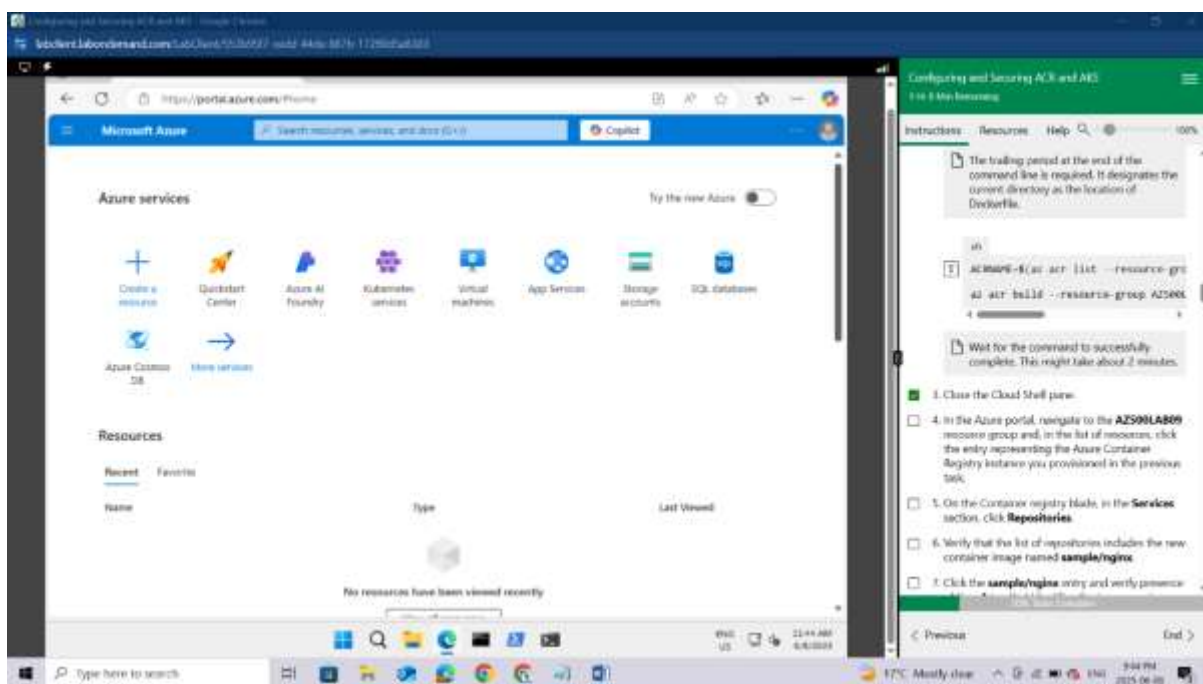
```
az acr build --resource-group AZ500LAB09 --image sample/nginx:v1 --registry
$ACRNAME --file Dockerfile .
```

Wait for the command to successfully complete. This might take about 2 minutes.





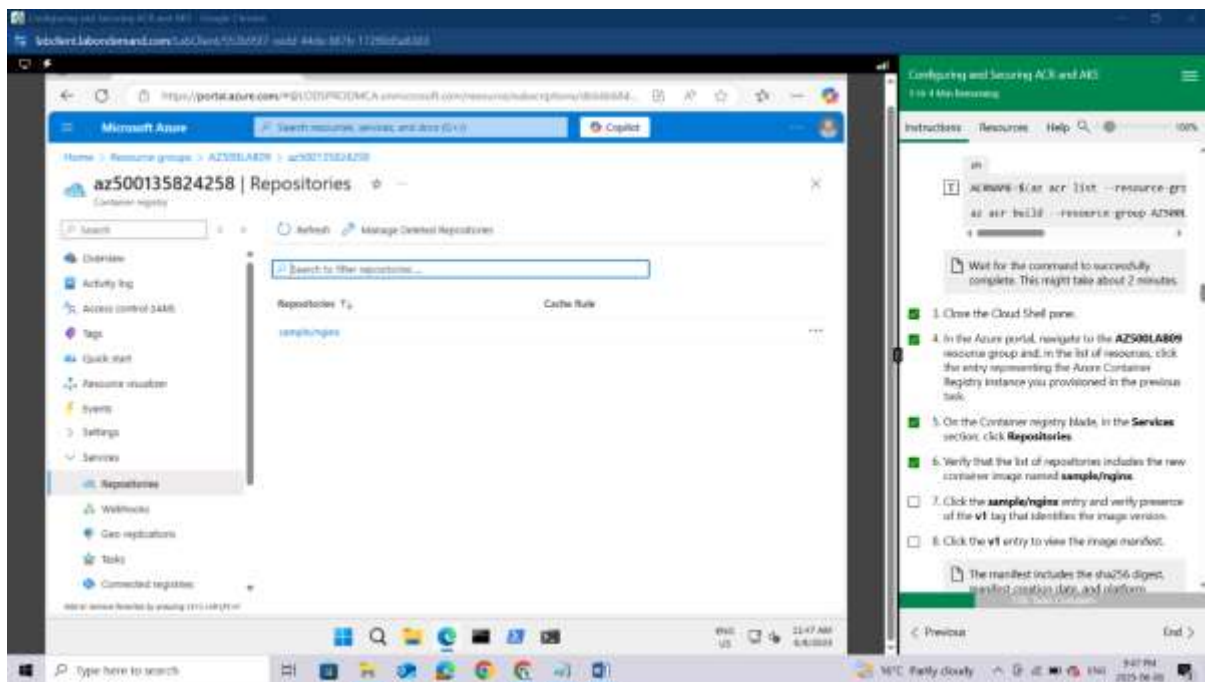
Close the Cloud Shell pane.



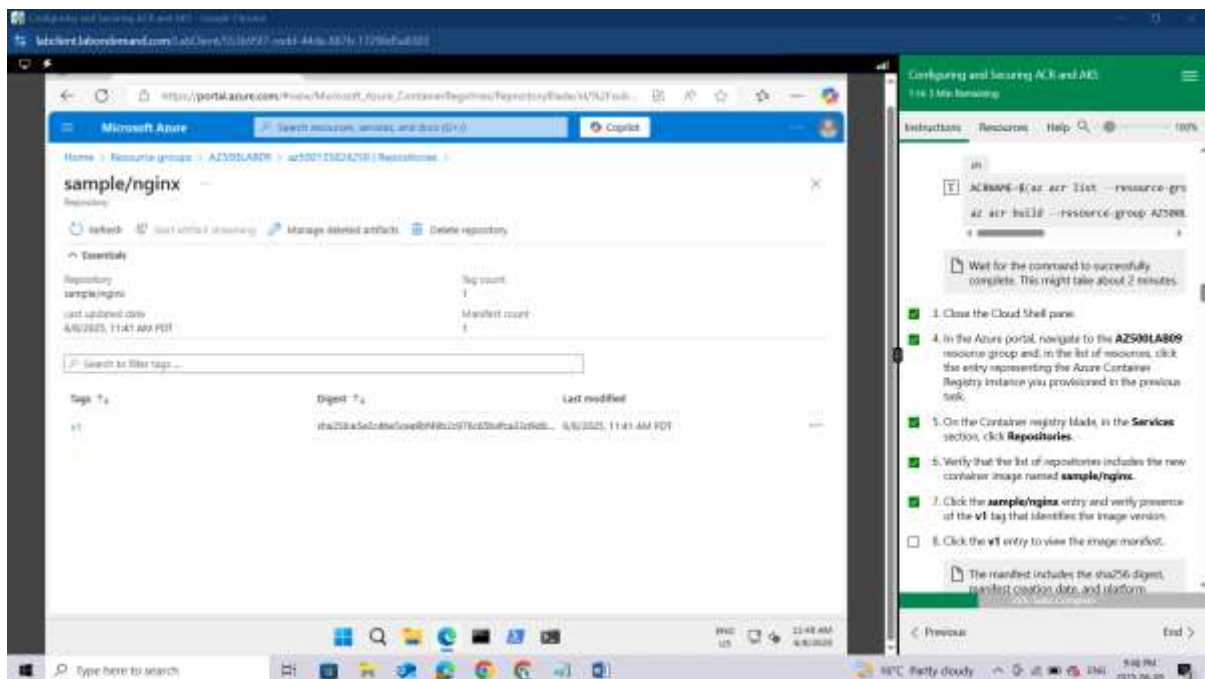
In the Azure portal, navigate to the **AZ500LAB09** resource group and,

The screenshot displays the Microsoft Azure portal interface. The main window shows the 'AZ500LAB09' resource group. Under the 'Resources' tab, a table lists the resources within the group. One resource is visible: 'az50003024256', identified as a 'Container Registry' in the 'East US' location. The table has columns for 'Name', 'Type', and 'Location'. To the right of the portal, a terminal window is open, showing the execution of Azure CLI commands: 'az acr list' and 'az acr build'. Below the terminal, a list of steps is visible, including 'Close the Cloud Shell pane', 'In the Azure portal, navigate to the AZ500LAB09 resource group...', 'On the Container registry blade, in the Services section, click Repositories', 'Verify that the list of repositories includes the new container image named sample/nginx', and 'Click the sample/nginx entry and verify presence...'. The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 17°C Mostly clear, 9:48 PM, 7/2/2024.

Verify that the list of repositories includes the new container image named **sample/nginx**.

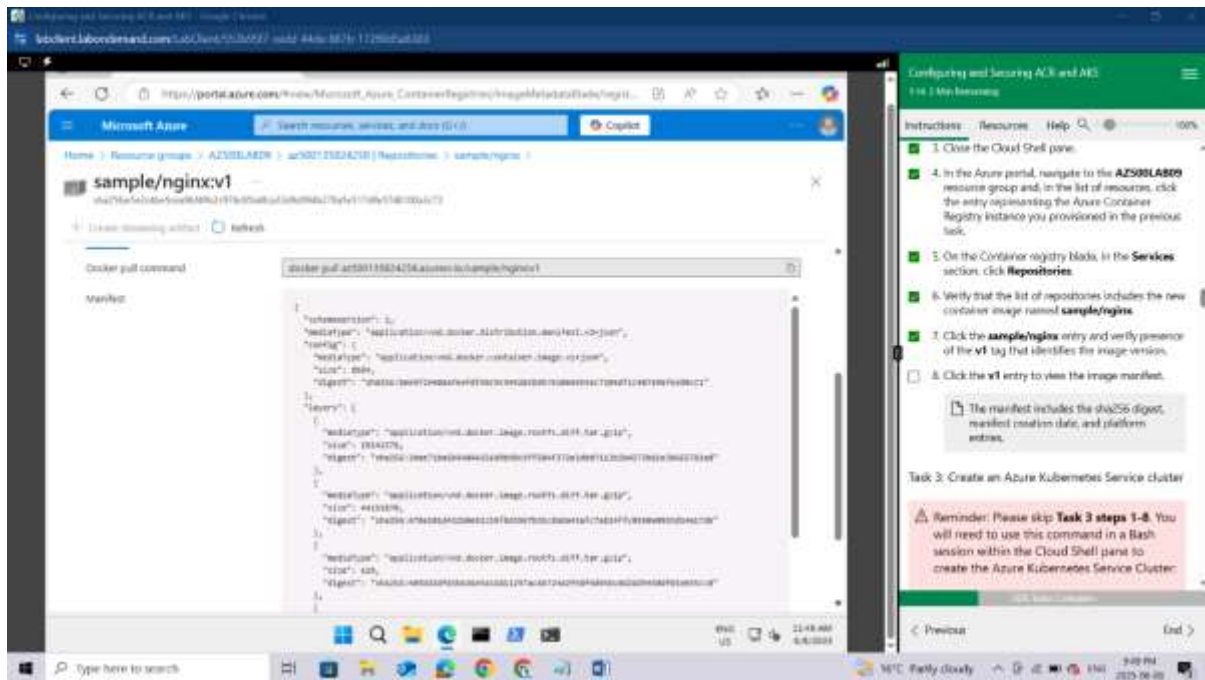


Click the **sample/nginx** entry and verify presence of the **v1** tag that identifies the image version.



Click the **v1** entry to view the image manifest.

The manifest includes the sha256 digest, manifest creation date, and platform entries.



Task 3: Create an Azure Kubernetes Service cluster

Reminder: Please skip Task 3 steps 1-8. You will need to use this command in a Bash session within the Cloud Shell pane to create the Azure Kubernetes Service Cluster:

```
Bash
az aks create --name MyKubernetesCluster --resource-group AZ500LAB09 --location eastus --no-ssh-key --node-vm-size Standard_D2s_v3 --nodepool-name agentpool --node-count 1 --vm-set-type AvailabilitySet --network-plugin azure --dns-name-prefix MyKubernetesCluster-dns
```

In this task, you will create an Azure Kubernetes service and review the deployed resources.

In the Azure portal, in the **Search resources, services, and docs** text box at the top of the Azure portal page, type **Kubernetes services** and press the **Enter** key.

On the **Kubernetes services** blade, click **+ Create** and, in the drop-down menu, click **+ Create a Kubernetes cluster**

On the **Basics** tab of the **Create Kubernetes cluster** blade, select **Cluster preset configuration**, select **Dev/Test (\$)**. Now specify the following settings (leave others with their default values):

Setting	Value
Subscription	the name of the Azure subscription you are using in this lab
Resource group	AZ500LAB09
Kubernetes cluster name	MyKubernetesCluster

Setting	Value
Region	(US) East US
Availability zones	None
Scale method	Manual
Node count	1

Click **Next: Node Pools** > and, on the **Node Pools** tab of the **Create Kubernetes cluster** blade, specify the following settings (leave others with their default values):

Setting	Value
Enable virtual nodes	cleared checkbox

Click **Next: Access** >, on the **Access** tab of the **Create Kubernetes cluster** blade, accept the defaults, and click **Next: Networking** >.

On the **Networking** tab of the **Create Kubernetes cluster** blade, specify the following settings (leave others with their default values):

Setting	Value
Network configuration	Azure CNI Overlay
DNS name prefix	Leave the default value

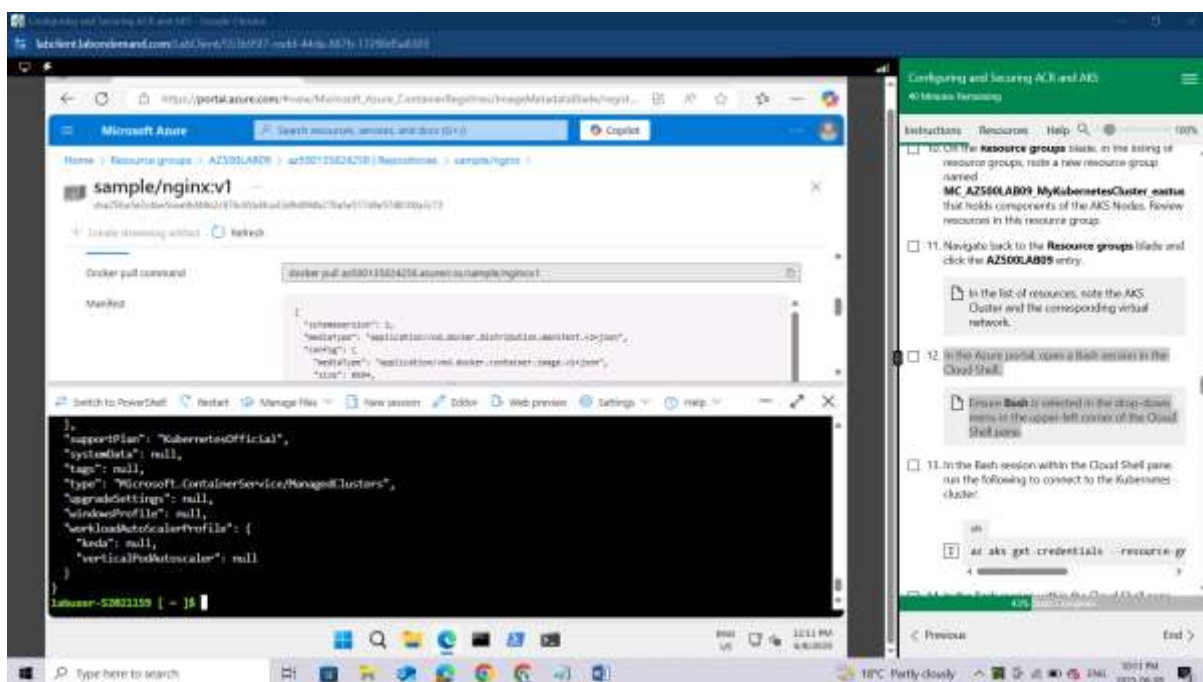
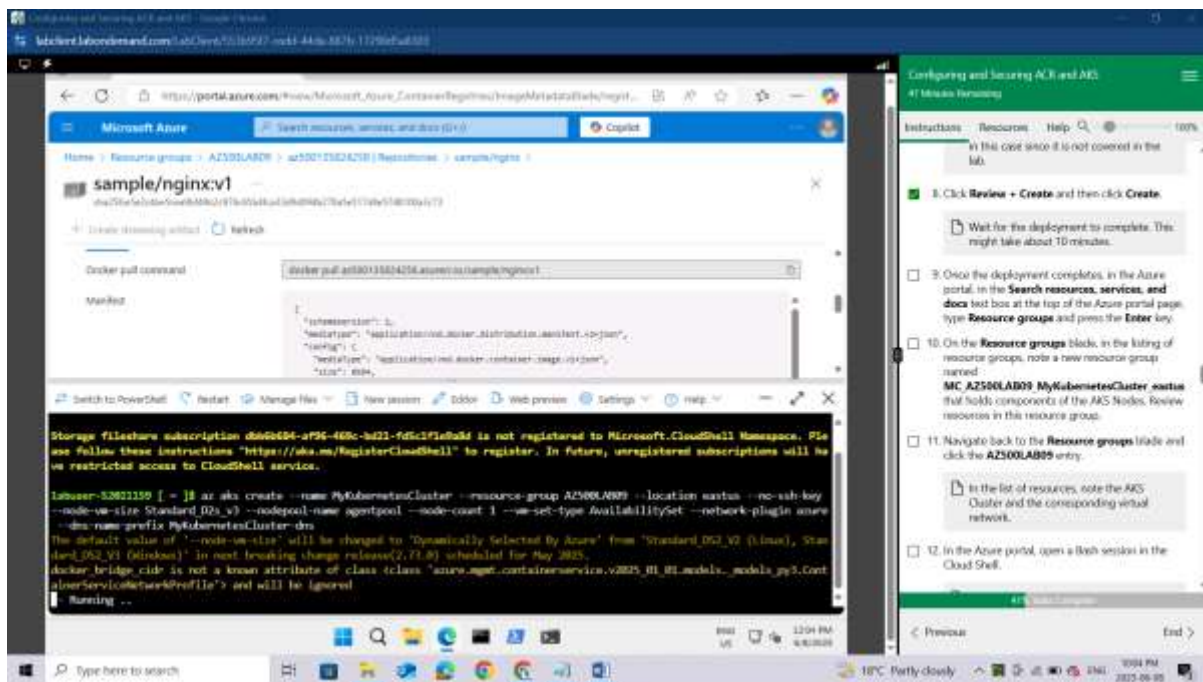
AKS can be configured as a private cluster. This assigns a private IP to the API server to ensure network traffic between your API server and your node pools remains on the private network only. For more information, visit <https://docs.microsoft.com/en-us/azure/aks/private-clusters> page.

Click **Next: Integrations** > and, on the **Integrations** tab of the **Create Kubernetes cluster** blade, set **Container monitoring** to **Disabled**.

In production scenarios, you would want to enable monitoring. Monitoring is disabled in this case since it is not covered in the lab.

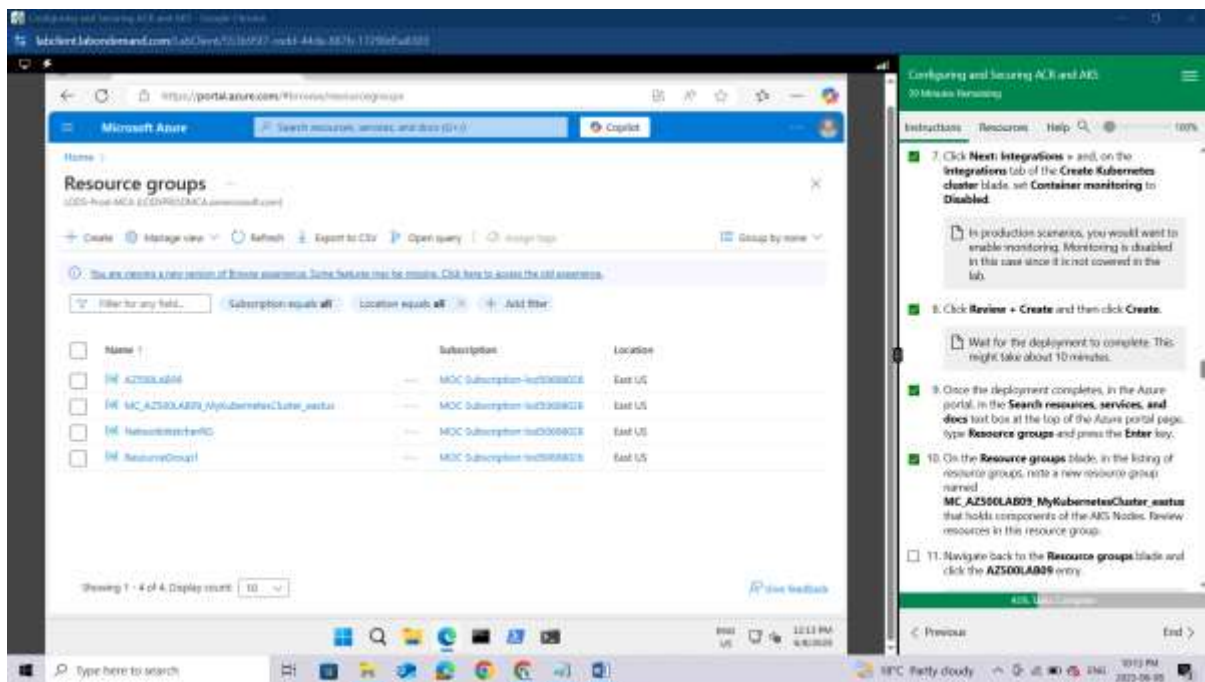
Click **Review + Create** and then click **Create**.

Wait for the deployment to complete. This might take about 10 minutes.

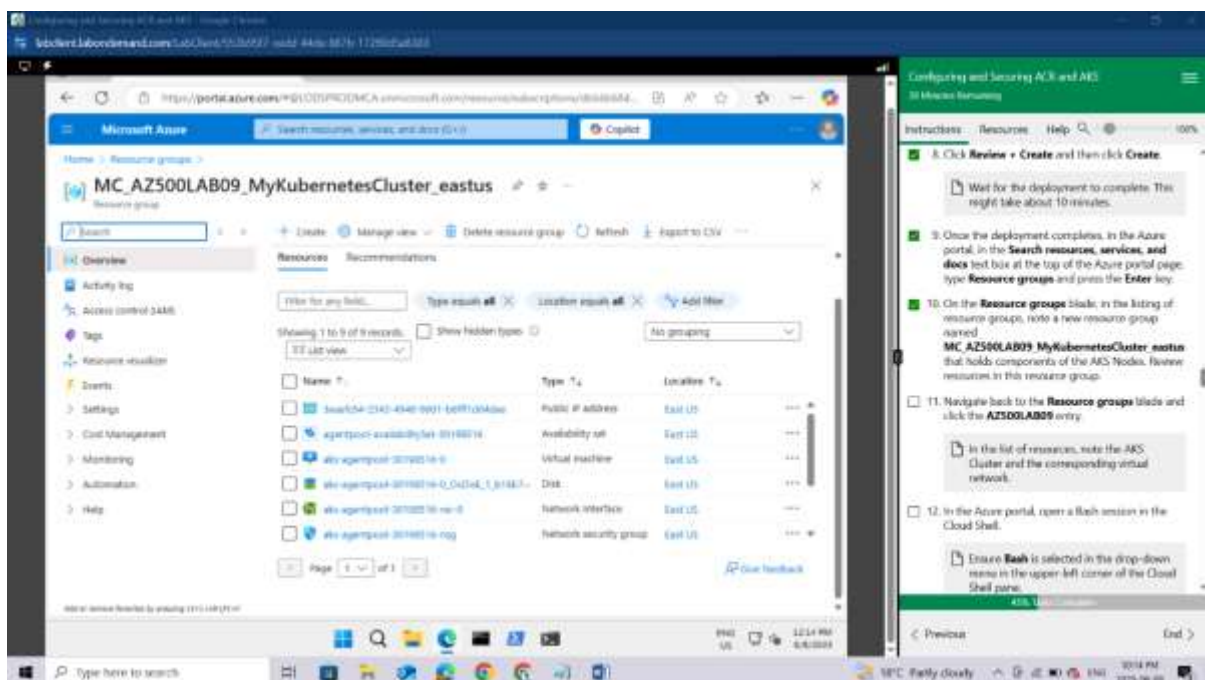


Once the deployment completes, in the Azure portal, in the **Search resources, services, and docs** text box at the top of the Azure portal page, type **Resource groups** and press the **Enter** key.

On the **Resource groups** blade, in the listing of resource groups, note a new resource group named **MC_AZ500LAB09_MyKubernetesCluster_eastus** that holds components of the AKS Nodes.

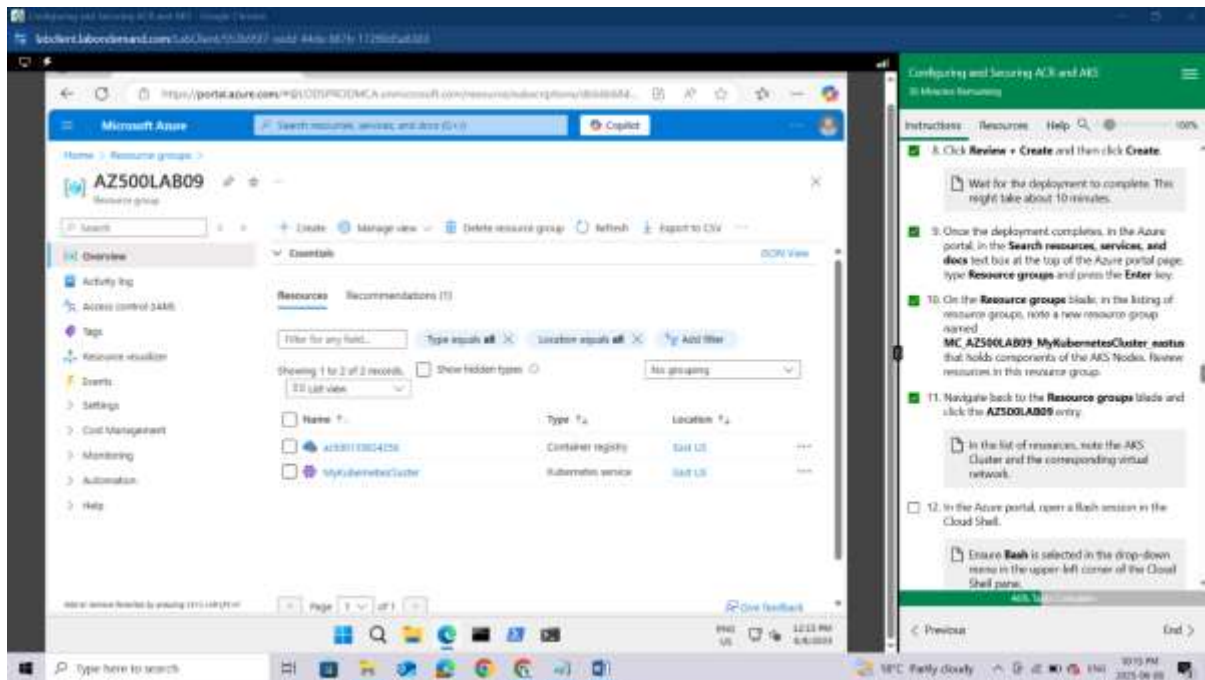


Review resources in this resource group.



Navigate back to the **Resource groups** blade and click the **AZ500LAB09** entry.

In the list of resources, note the AKS Cluster and the corresponding virtual network.



In the Azure portal, open a Bash session in the Cloud Shell.

Ensure **Bash** is selected in the drop-down menu in the upper-left corner of the Cloud Shell pane.

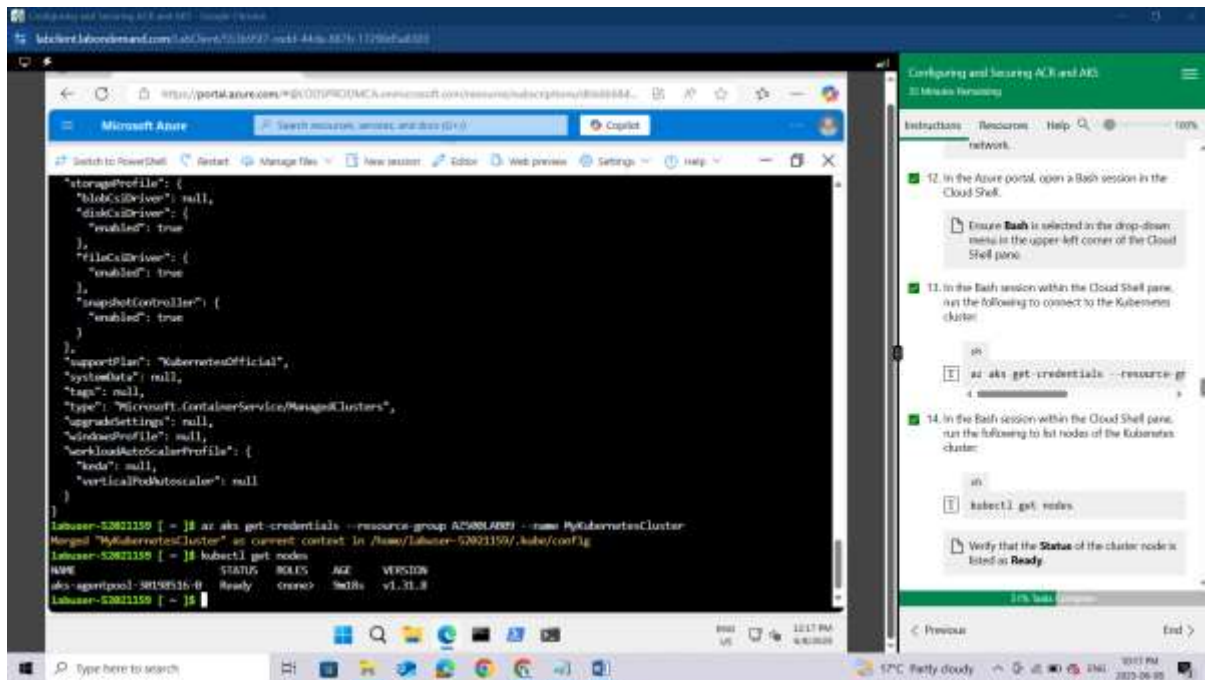
In the Bash session within the Cloud Shell pane, run the following to connect to the Kubernetes cluster:

```
sh
az aks get-credentials --resource-group AZ500LAB09 --name MyKubernetesCluster
```

In the Bash session within the Cloud Shell pane, run the following to list nodes of the Kubernetes cluster:

```
sh
kubectl get nodes
```

Verify that the **Status** of the cluster node is listed as **Ready**.



Task 4: Grant the AKS cluster permissions to access the ACR and manage its virtual network

Reminder: Before running Task 4 step 2 you will need to manually create a virtual network. In the Search Resources, Services, and Docs text box at the top of the Azure portal page, type Virtual networks and press the Enter key. Click the Create button, and once that opens use the AZ500LAB09 Resource Group and name it AZ500LAB09-vnet. Once this is done you can use the script in Task 4 step 2.

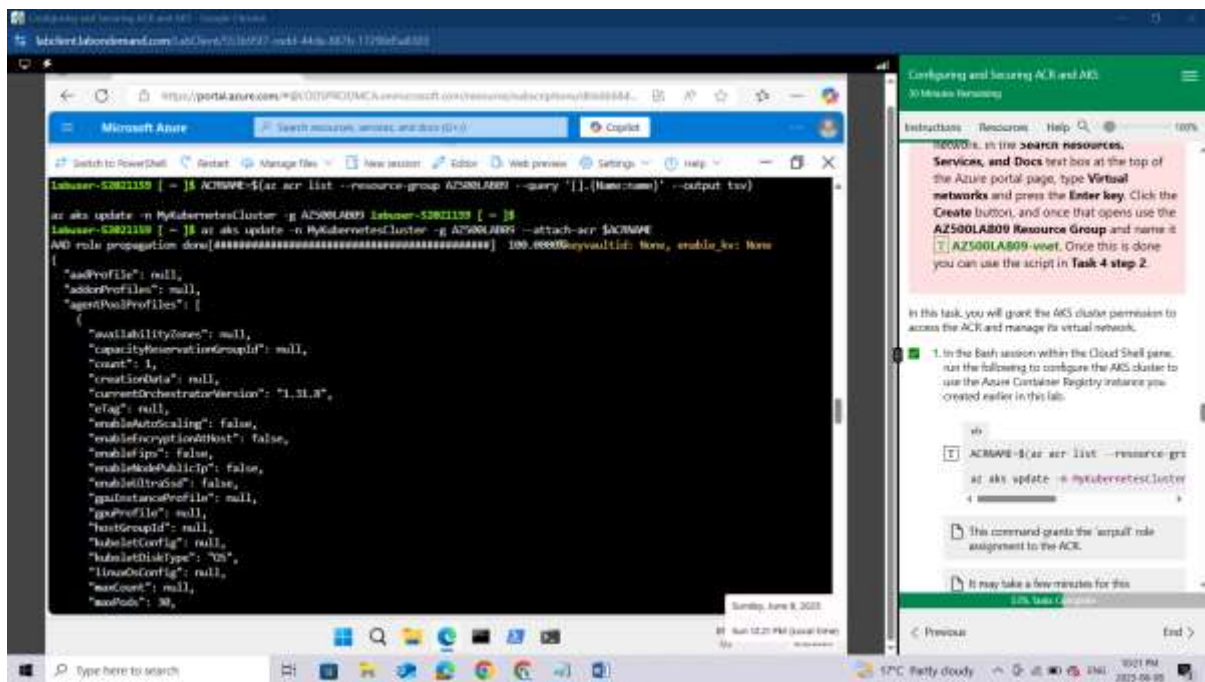
In this task, you will grant the AKS cluster permission to access the ACR and manage its virtual network.

In the Bash session within the Cloud Shell pane, run the following to configure the AKS cluster to use the Azure Container Registry instance you created earlier in this lab.

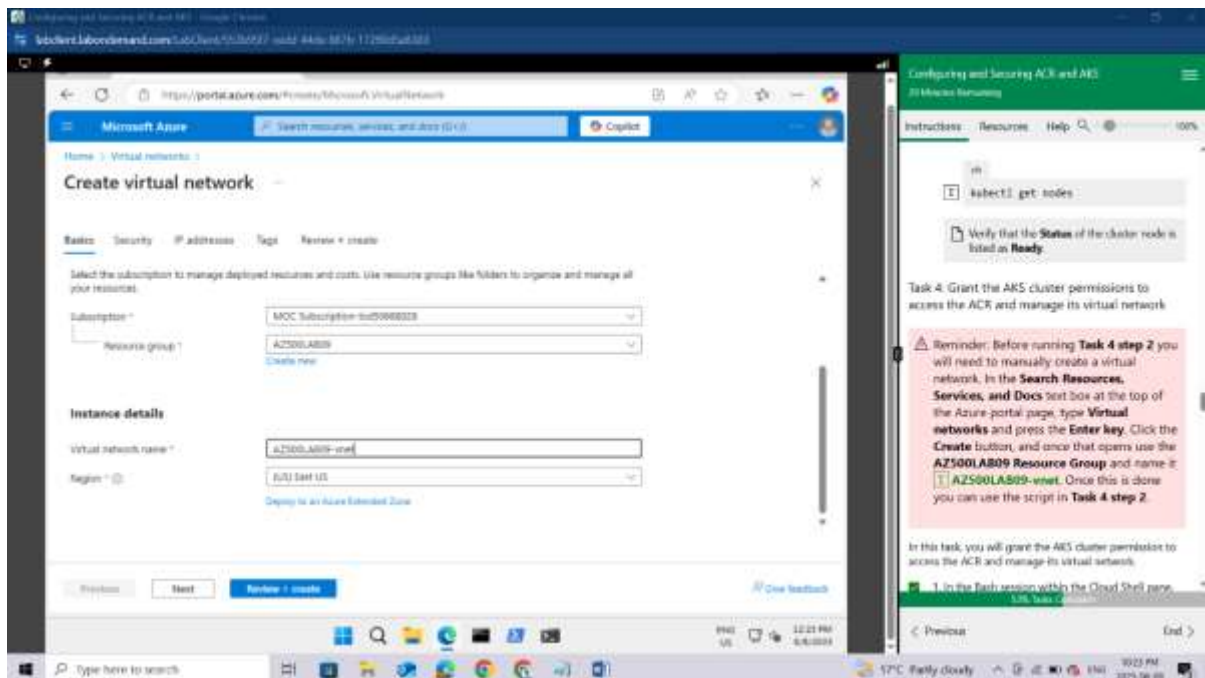
```
sh
ACRNAME=$(az acr list --resource-group AZ500LAB09 --query '[].{Name:name}' --
output tsv)
```

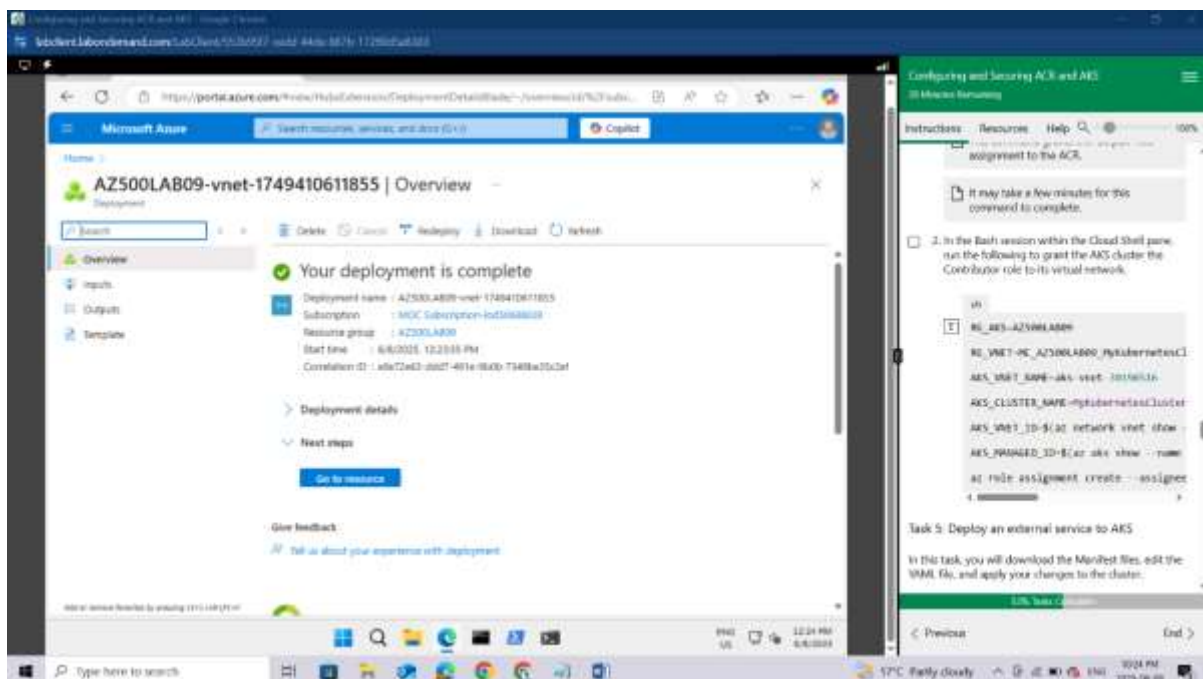
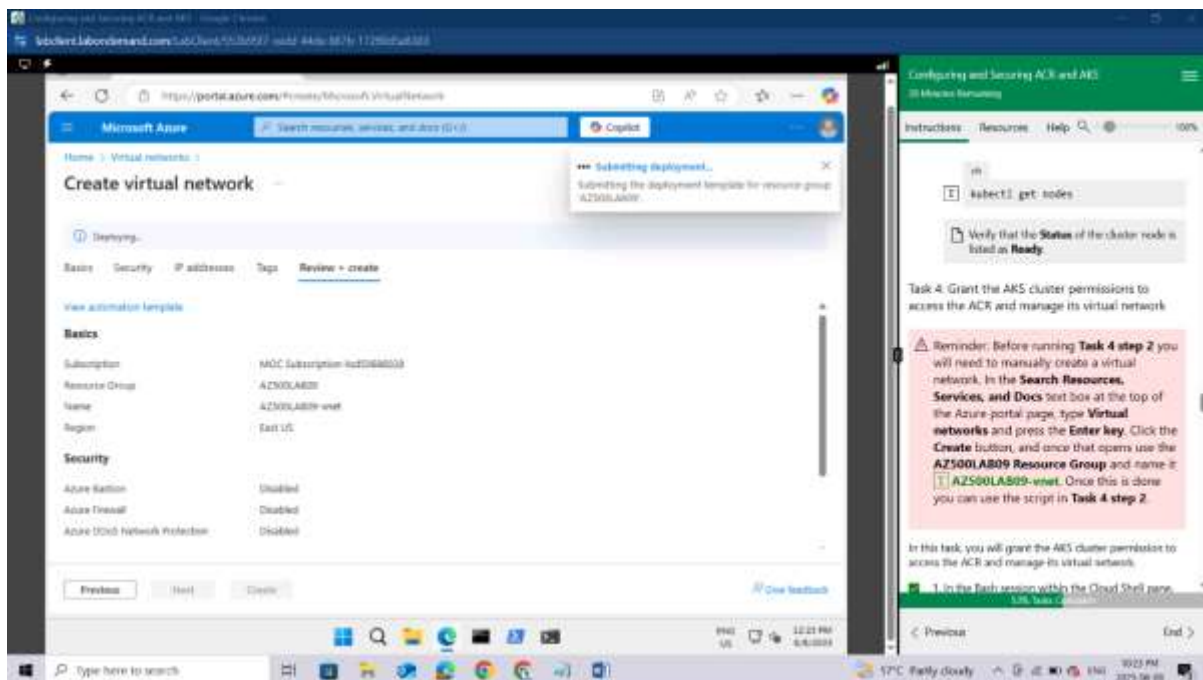
```
az aks update -n MyKubernetesCluster -g AZ500LAB09 --attach-acr $ACRNAME
```

*This command grants the 'acrpull' role assignment to the ACR.
It may take a few minutes for this command to complete.*



Before running Task 4 step 2 you will need to manually create a virtual network. In the Search Resources, Services, and Docs text box at the top of the Azure portal page, type Virtual networks and press the Enter key. Click the Create button, and once that opens use the AZ500LAB09 Resource Group and name it AZ500LAB09-vnet. Once this is done you can use the script in Task 4 step 2.





In the Bash session within the Cloud Shell pane, run the following to grant the AKS cluster the Contributor role to its virtual network.

```
sh
```

```
RG_AKS=AZ500LAB09
```

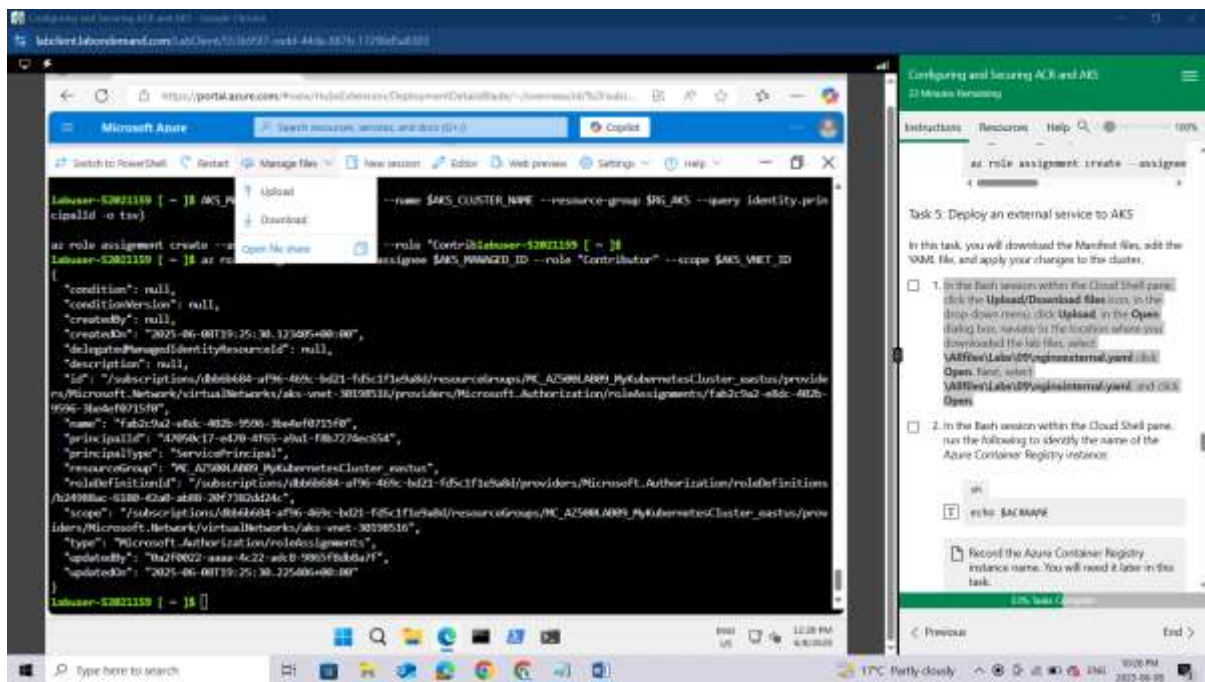
```
RG_VNET=MC_AZ500LAB09_MyKubernetesCluster_eastus
```

```
AKS_VNET_NAME=aks-vnet-30198516
```

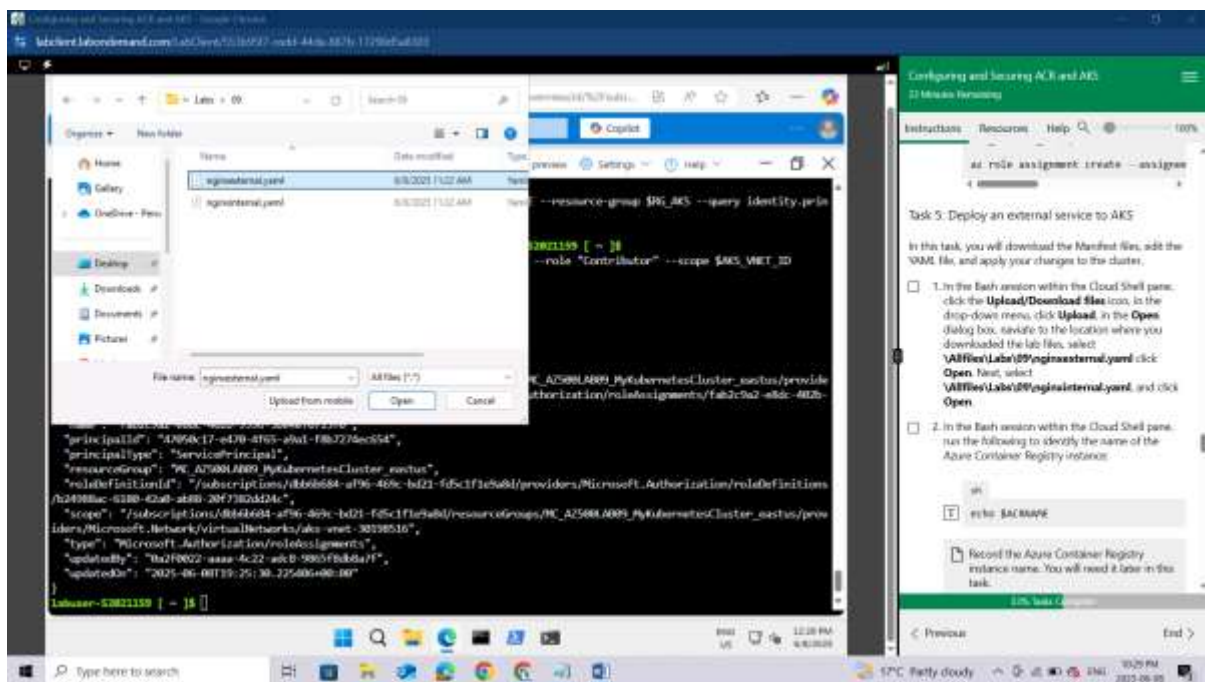
```
AKS_CLUSTER_NAME=MyKubernetesCluster
```

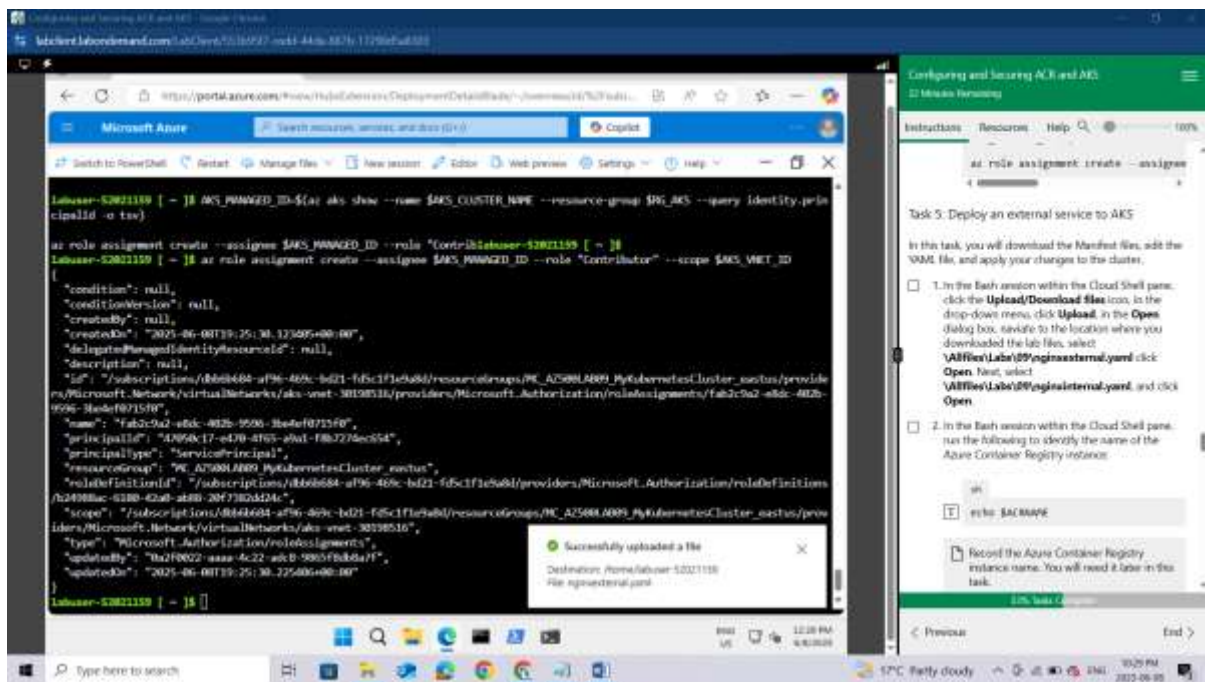
```
AKS_VNET_ID=$(az network vnet show --name $AKS_VNET_NAME --resource-group $RG_VNET --query id -o tsv)
```

In the Bash session within the Cloud Shell pane, click the **Upload/Download files** icon, in the drop-down menu, click **Upload**,

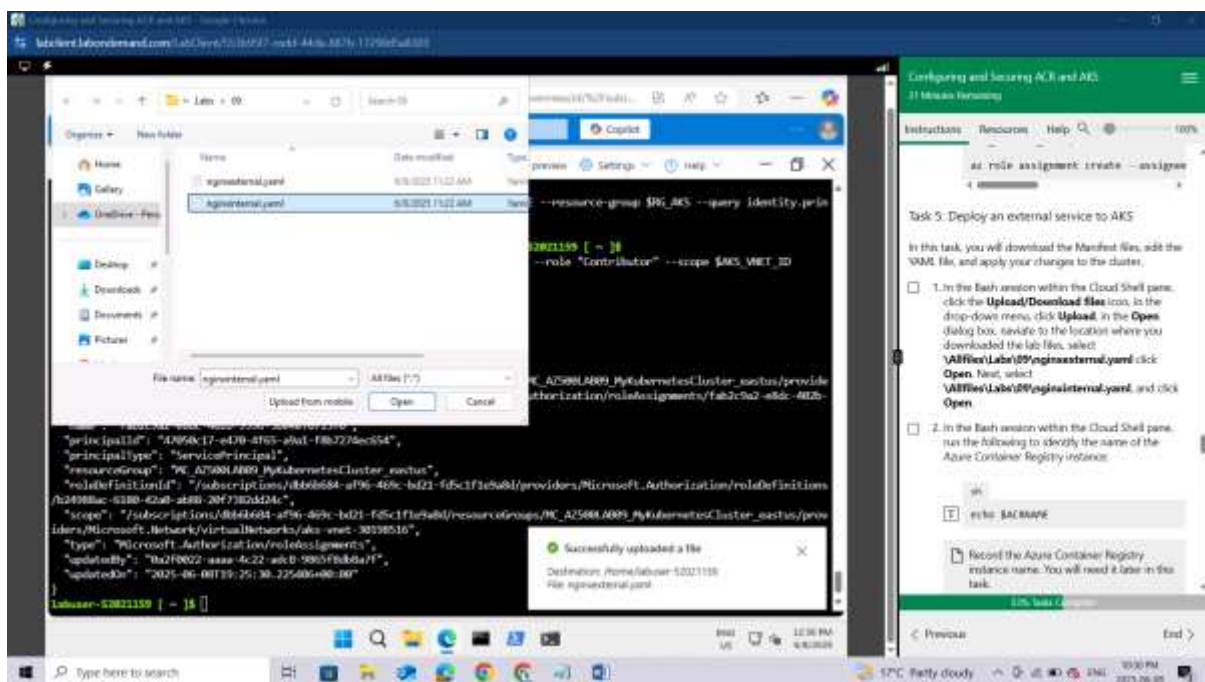


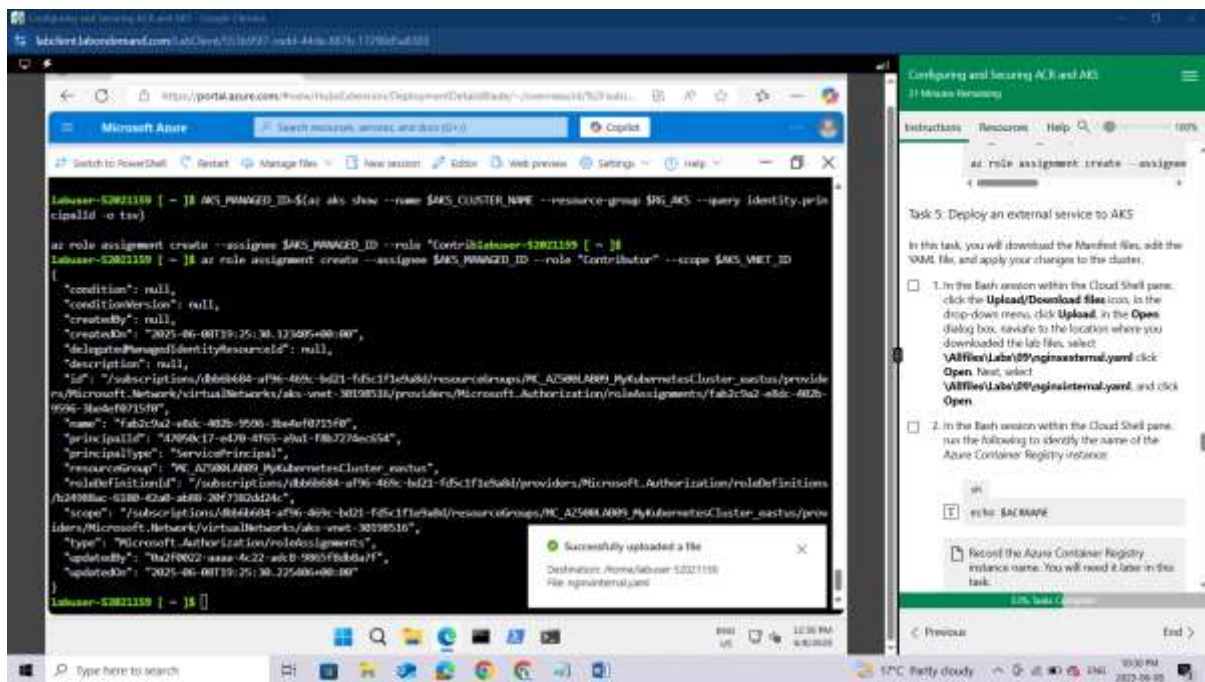
in the **Open** dialog box, navigate to the location where you downloaded the lab files, select **\Allfiles\Labs\09\nginxexternal.yaml** click **Open**.





Next, select **\\Allfiles\Labs\09\nginxinternal.yaml**, and click **Open**.

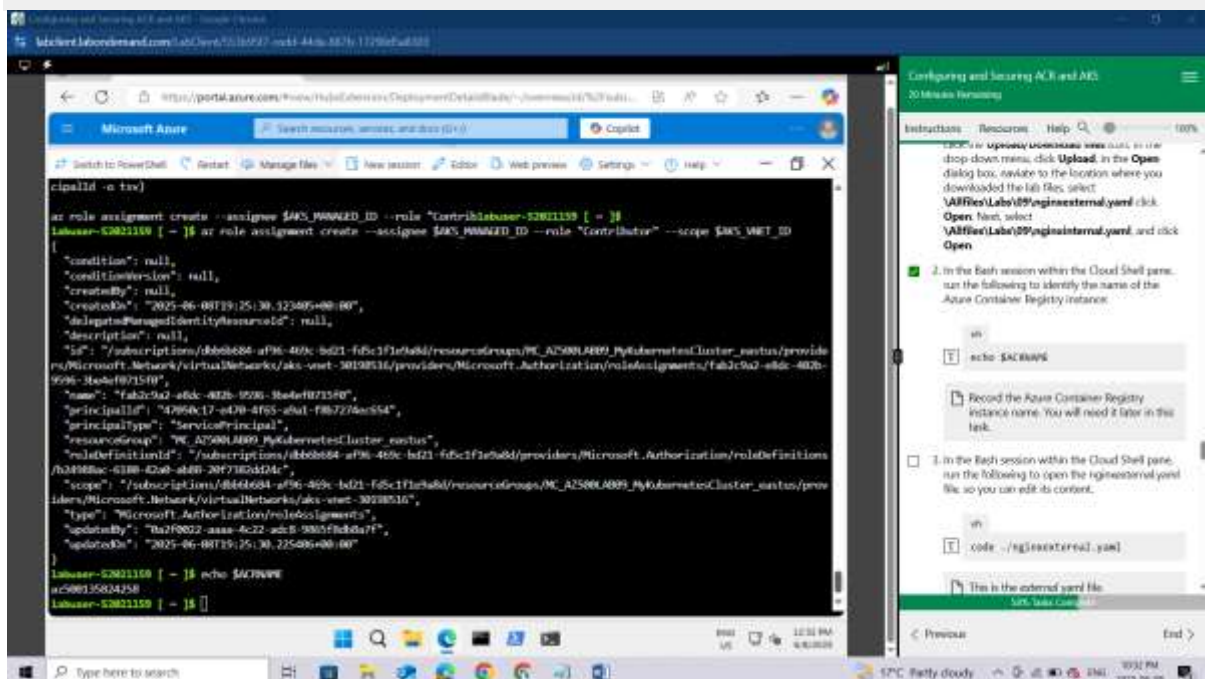




In the Bash session within the Cloud Shell pane, run the following to identify the name of the Azure Container Registry instance:

```
sh
echo $ACRNAME
```

Record the Azure Container Registry instance name. You will need it later in this task.

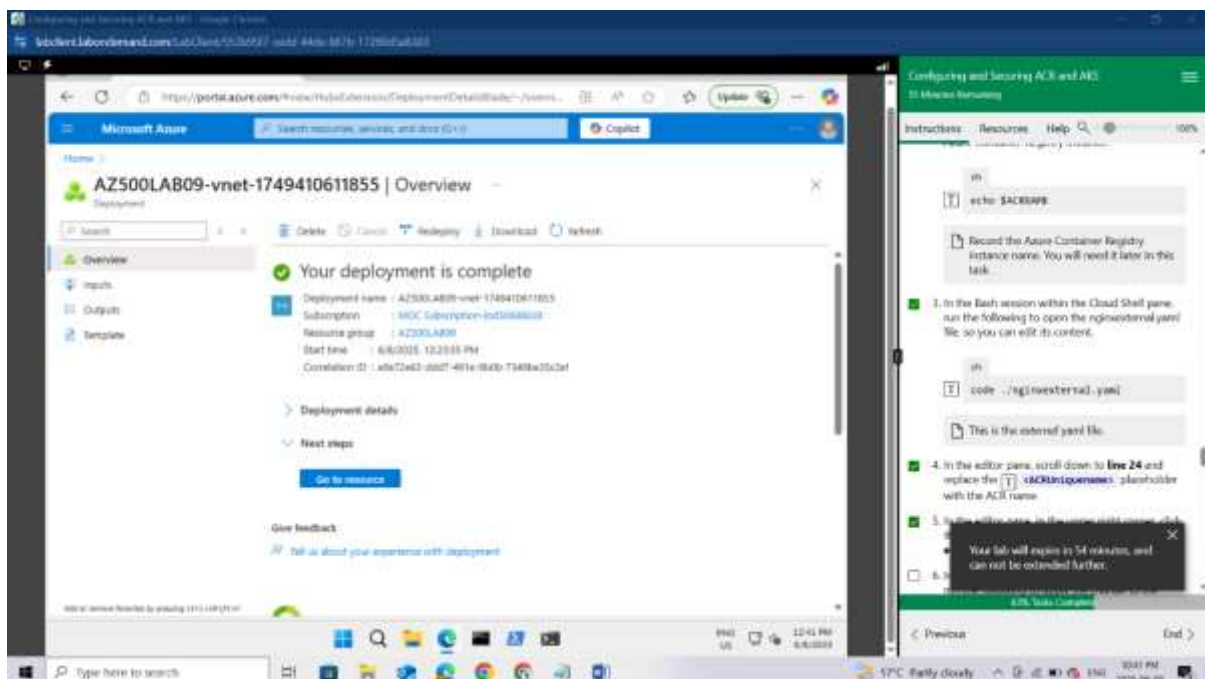
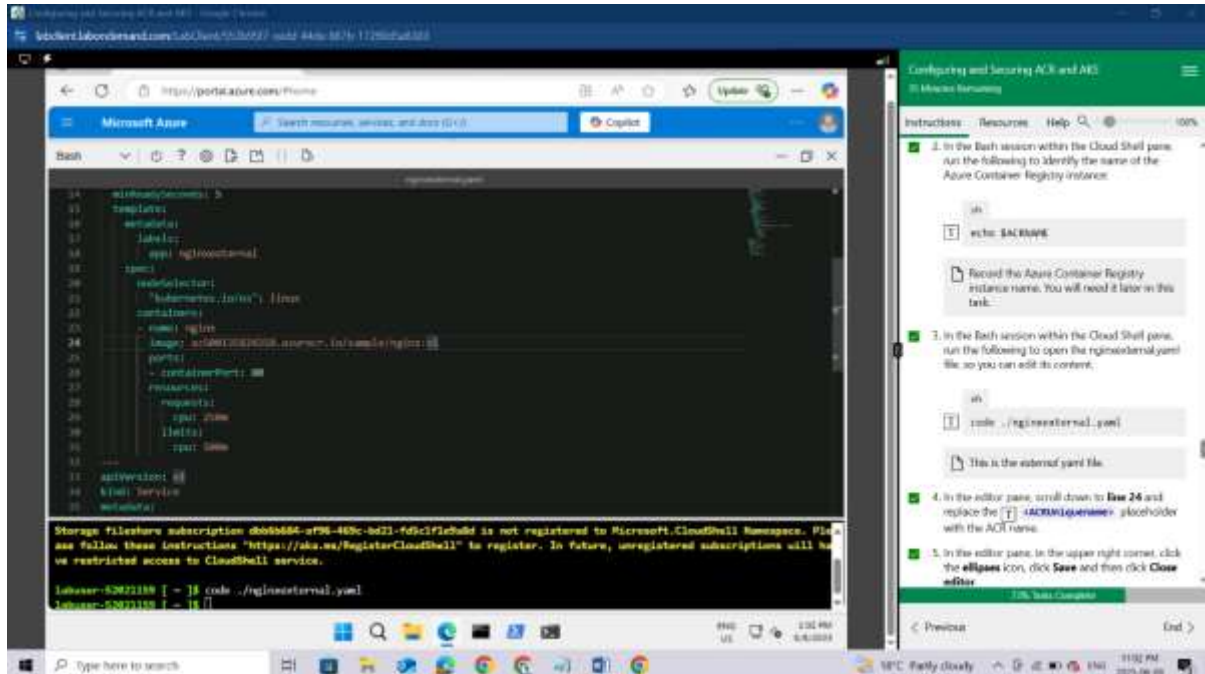


In the Bash session within the Cloud Shell pane, run the following to open the nginxexternal.yaml file, so you can edit its content.

```
sh
code ./nginxexternal.yaml
```

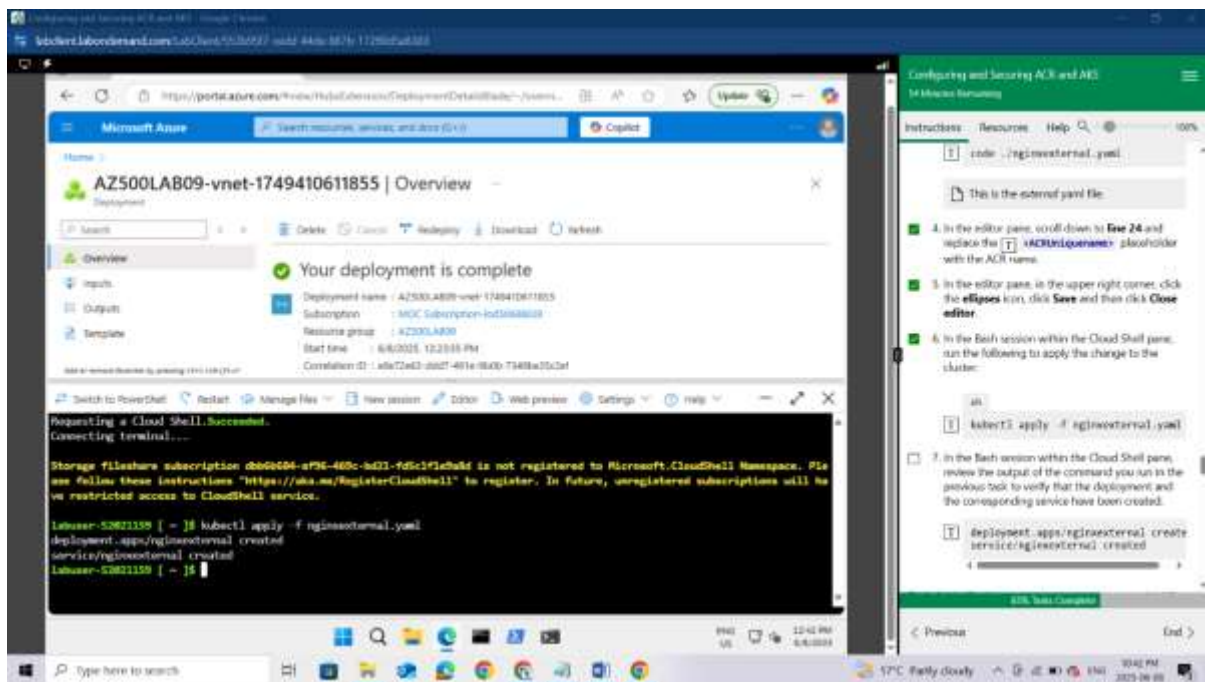

This is the *external* yaml file.

In the editor pane, scroll down to **line 24** and replace the `<ACRUniquename>` placeholder with the ACR name. In the editor pane, press CTRL + S to **Save** and then **Close editor**.



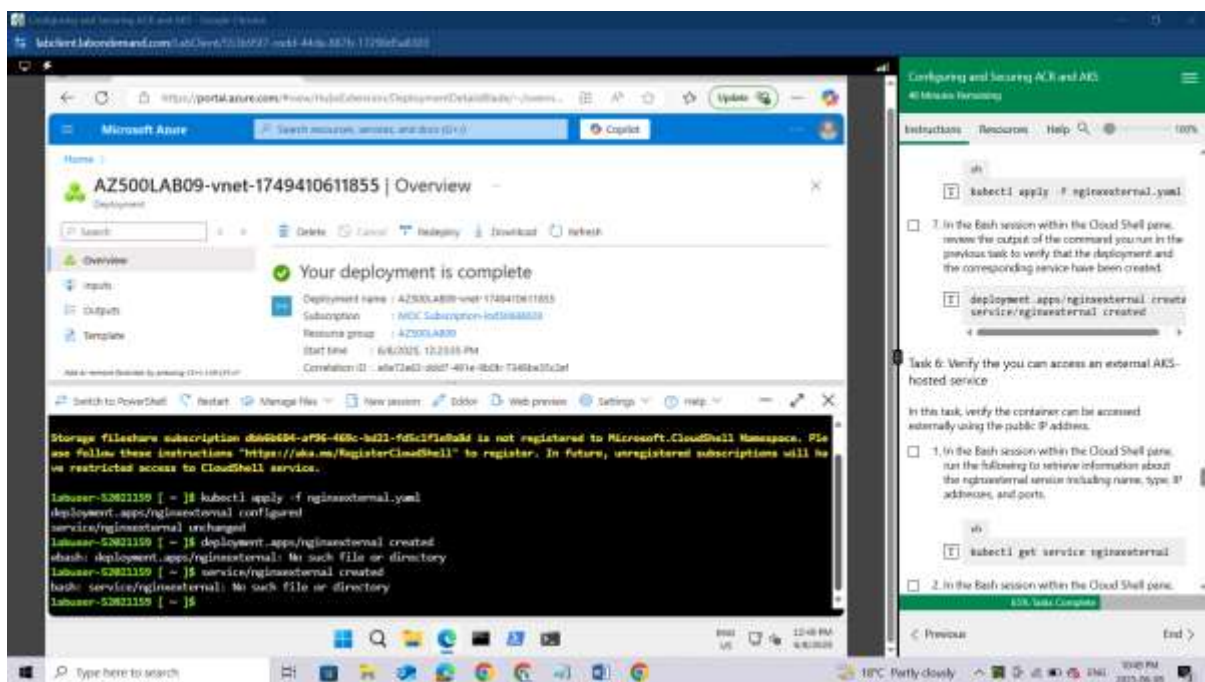
In the Bash session within the Cloud Shell pane, run the following to apply the change to the cluster:

```
sh
kubectl apply -f nginxexternal.yaml
```



In the Bash session within the Cloud Shell pane, review the output of the command you run in the previous task to verify that the deployment and the corresponding service have been created.

deployment.apps/nginxexternal created
service/nginxexternal created



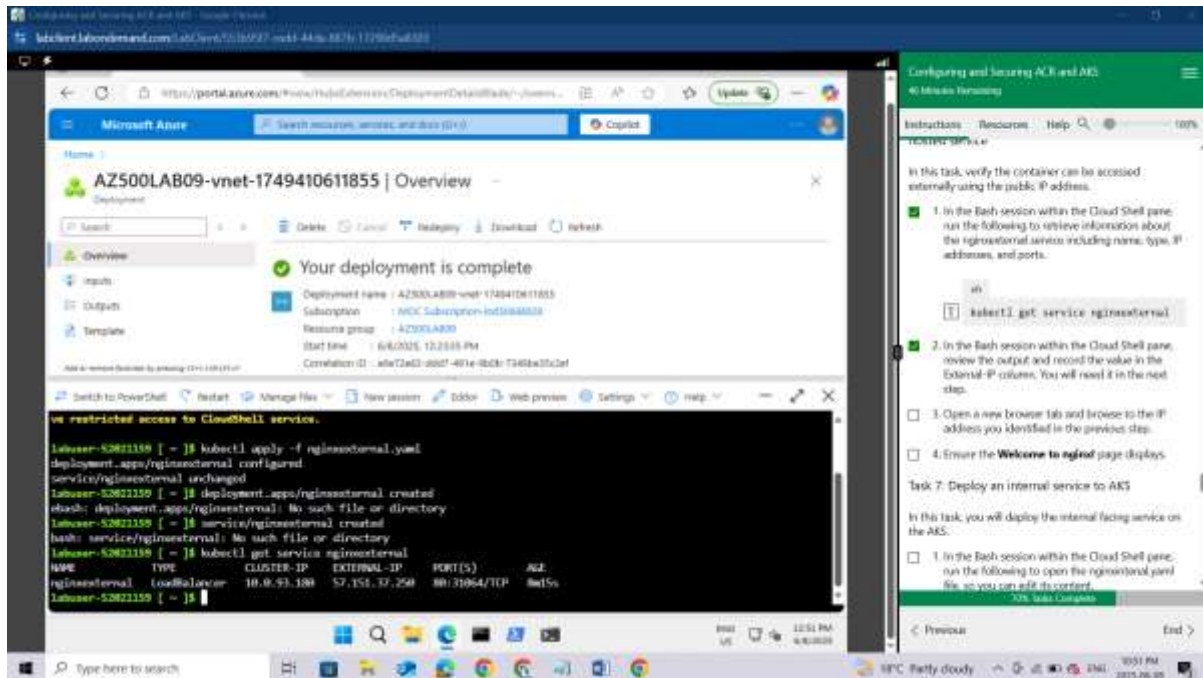
Task 6: Verify the you can access an external AKS-hosted service

In this task, verify the container can be accessed externally using the public IP address.

In the Bash session within the Cloud Shell pane, run the following to retrieve information about the nginxexternal service including name, type, IP addresses, and ports.

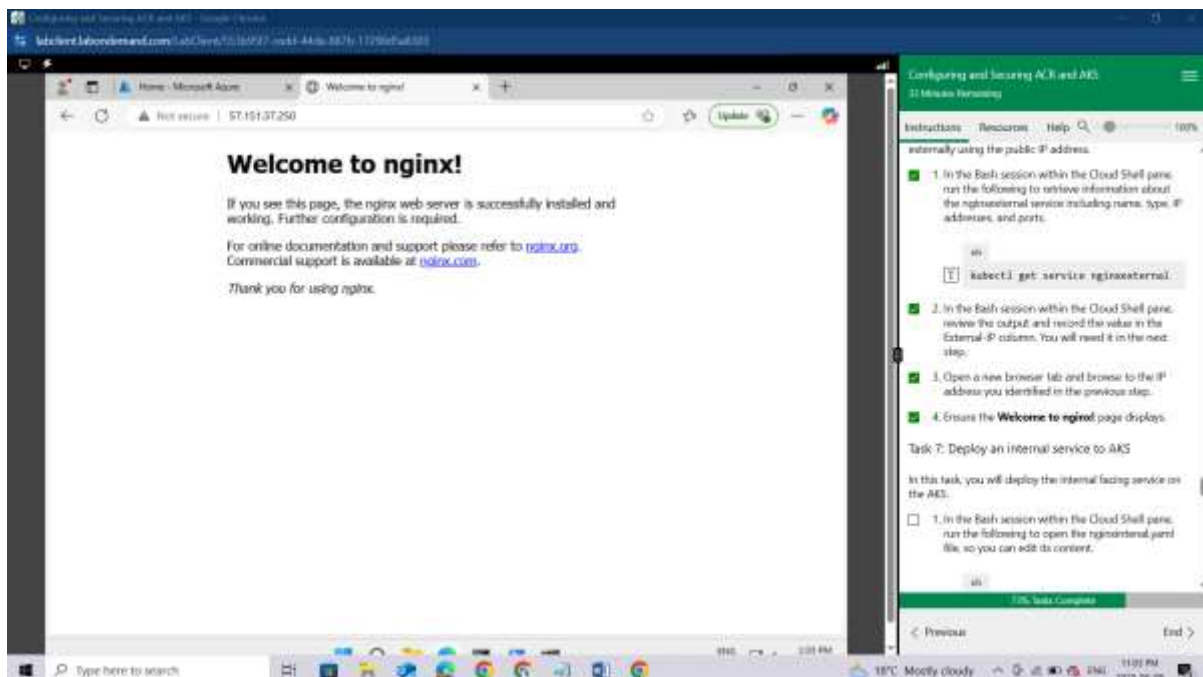
```
sh
kubectl get service nginxexternal
```

In the Bash session within the Cloud Shell pane, review the output and record the value in the **External-IP** column. You will need it in the next step.



Open a new browser tab and browse to the IP address you identified in the previous step.

Ensure the **Welcome to nginx!** page displays.



Task 7: Deploy an internal service to AKS

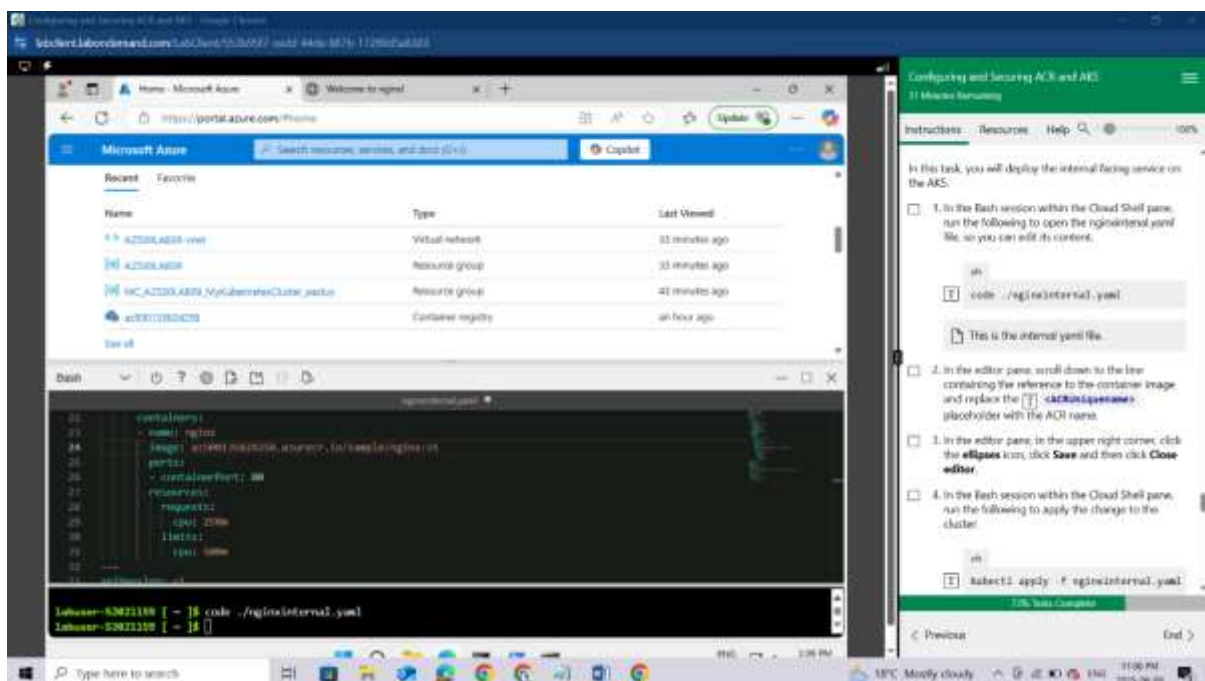
In this task, you will deploy the internal facing service on the AKS.

In the Bash session within the Cloud Shell pane, run the following to open the `nginxinternal.yaml` file, so you can edit its content.

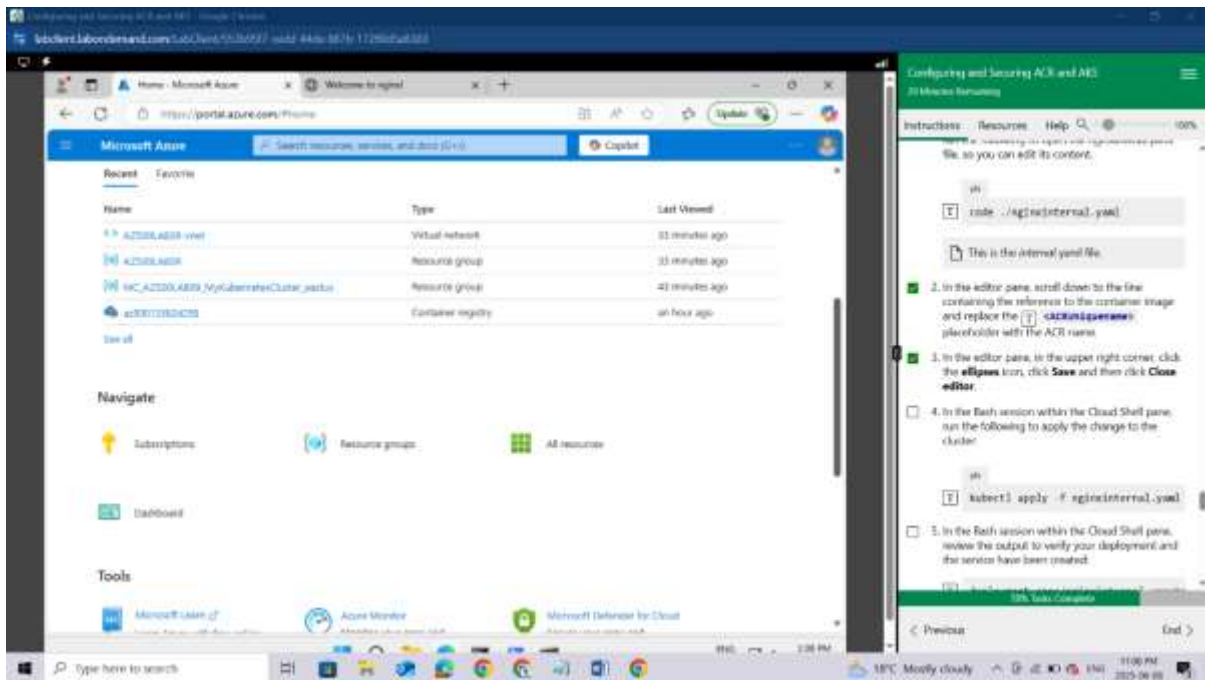
```
sh
code ./nginxinternal.yaml
```

This is the internal yaml file.

In the editor pane, scroll down to the line containing the reference to the container image and replace the `<ACRUniquename>` placeholder with the ACR name.



In the editor pane, click CTRL+S to **Save** and then click **Close editor**.



In the Bash session within the Cloud Shell pane, run the following to apply the change to the cluster:

```
sh
kubectl apply -f nginxinternal.yaml
```

In the Bash session within the Cloud Shell pane, review the output to verify your deployment and the service have been created:

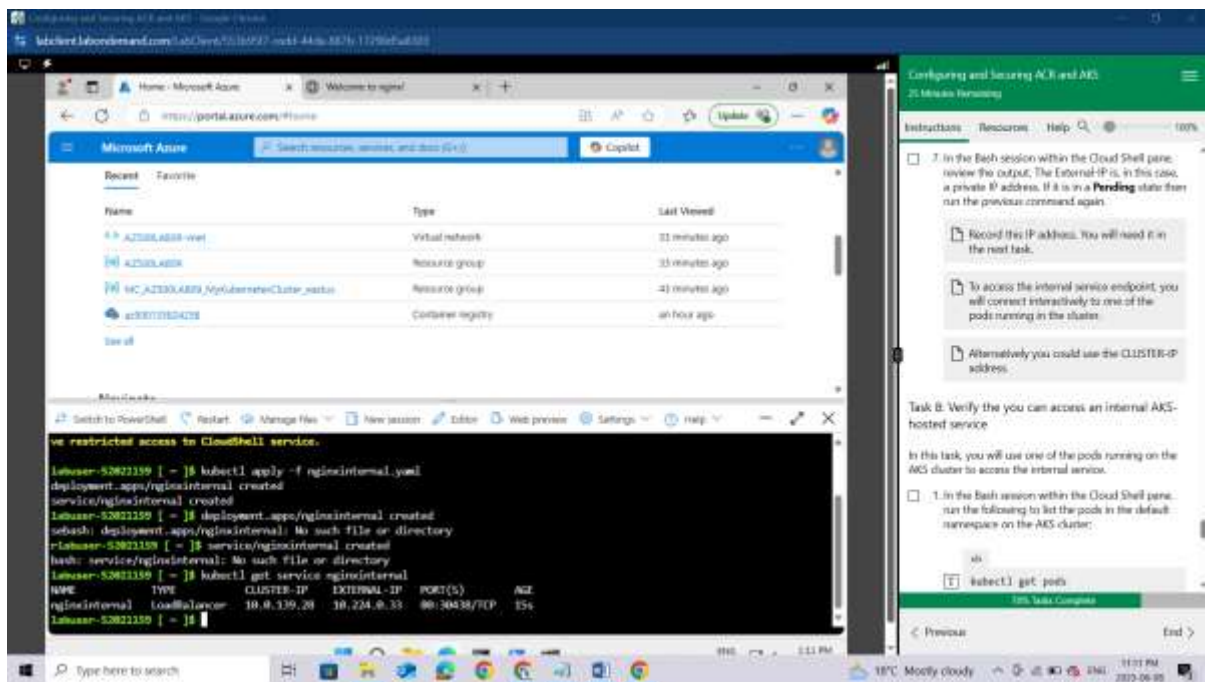
```
deployment.apps/nginxinternal created
service/nginxinternal created
```

In the Bash session within the Cloud Shell pane, run the following to retrieve information about the nginxinternal service including name, type, IP addresses, and ports.

```
sh
kubectl get service nginxinternal
```

In the Bash session within the Cloud Shell pane, review the output. The External-IP is, in this case, a private IP address. If it is in a **Pending** state then run the previous command again.

*Record this IP address. You will need it in the next task.
To access the internal service endpoint, you will connect interactively to one of the pods running in the cluster.
Alternatively you could use the CLUSTER-IP address.*



Task 8: Verify the you can access an internal AKS-hosted service

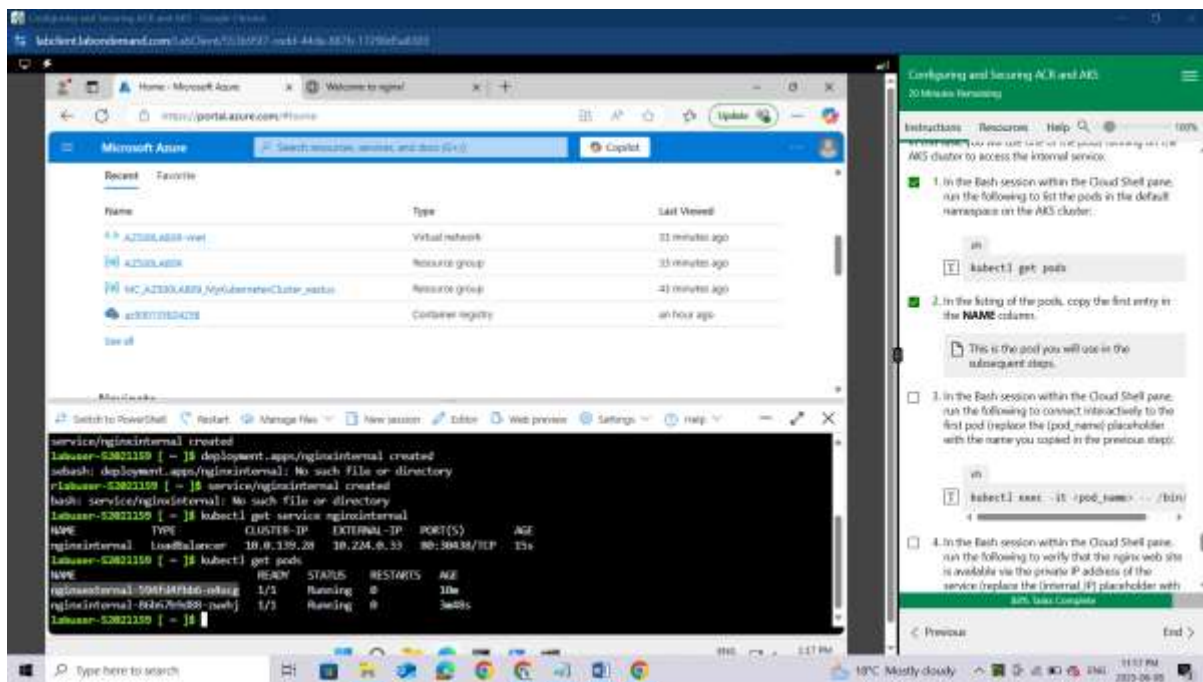
In this task, you will use one of the pods running on the AKS cluster to access the internal service.

In the Bash session within the Cloud Shell pane, run the following to list the pods in the default namespace on the AKS cluster:

```
sh
kubectl get pods
```

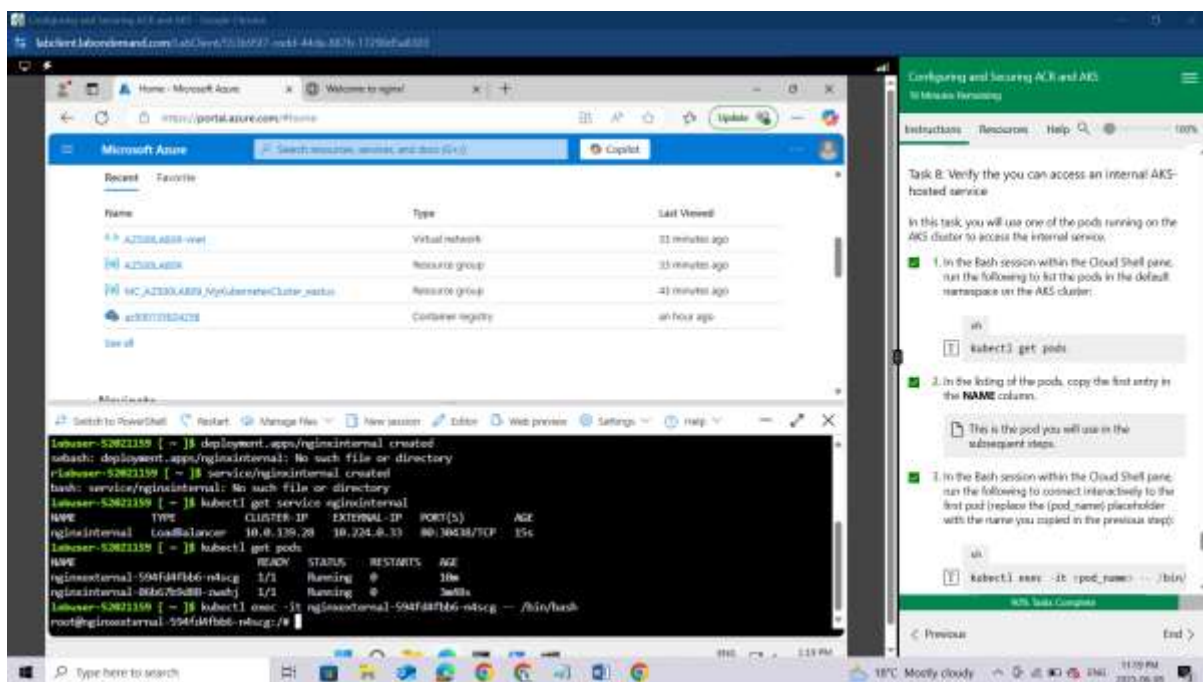
In the listing of the pods, copy the first entry in the **NAME** column.

This is the pod you will use in the subsequent steps.



In the Bash session within the Cloud Shell pane, run the following to connect interactively to the first pod (replace the (pod_name) placeholder with the name you copied in the previous step):

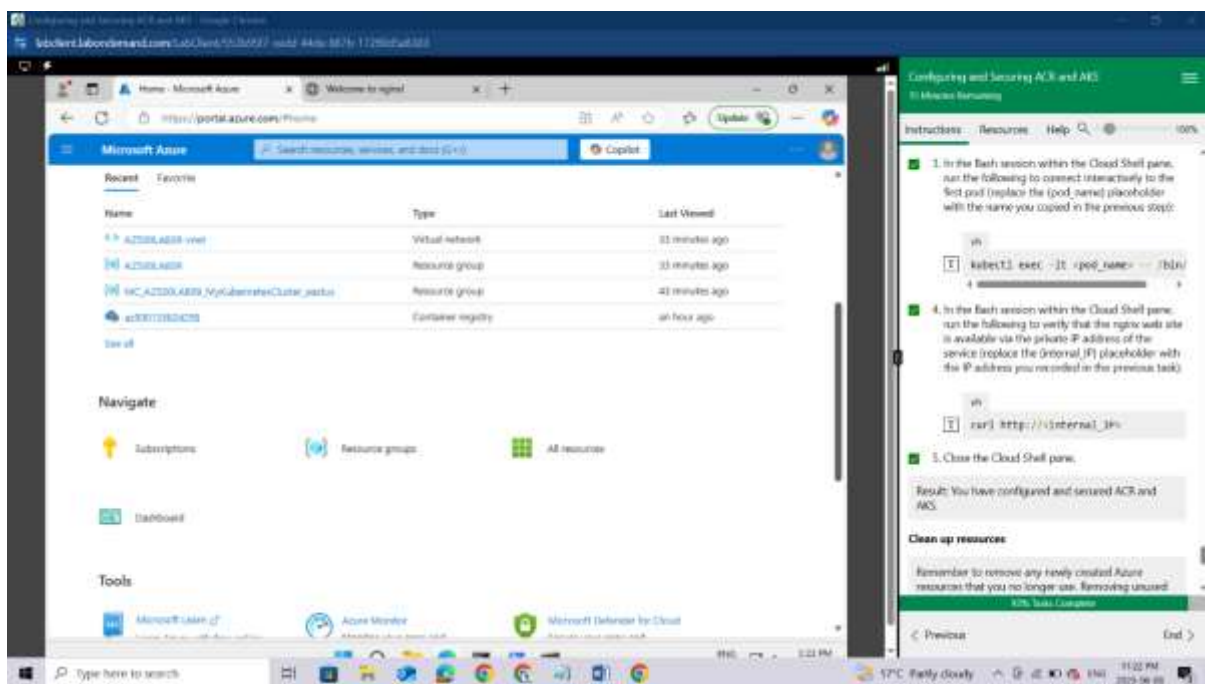
```
sh
kubectl exec -it <pod_name> -- /bin/bash
```



In the Bash session within the Cloud Shell pane, run the following to verify that the nginx web site is available via the private IP address of the service (replace the (internal_IP) placeholder with the IP address you recorded in the previous task):

```
sh
curl http://<internal_IP>
```

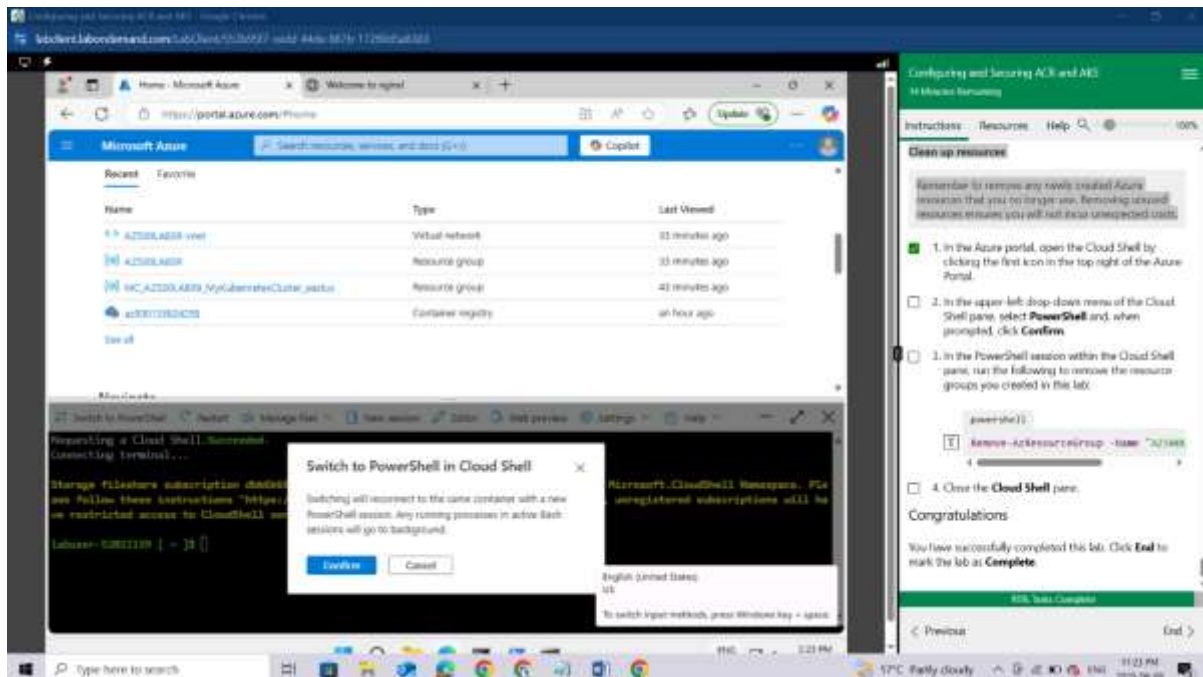
Result: You have configured and secured ACR and AKS.



Remember to remove any newly created Azure resources that you no longer use. Removing unused resources ensures you will not incur unexpected costs.

In the Azure portal, open the Cloud Shell by clicking the first icon in the top right of the Azure Portal.

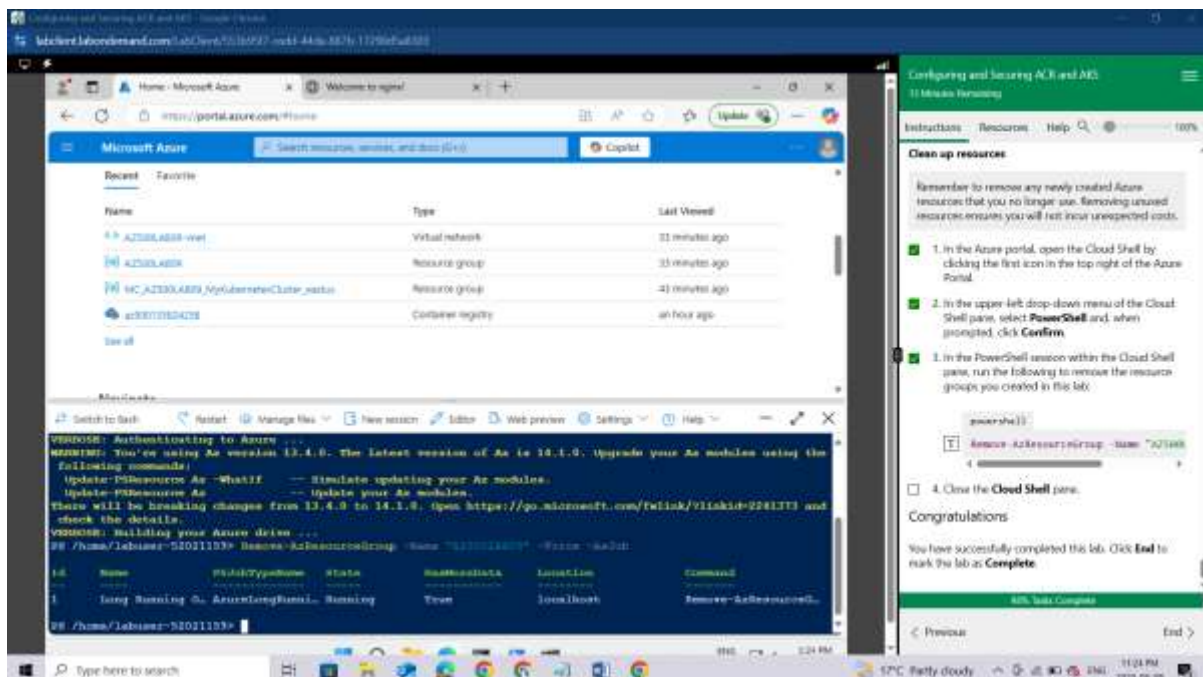
In the upper-left drop-down menu of the Cloud Shell pane, select **PowerShell** and, when prompted, click **Confirm**.



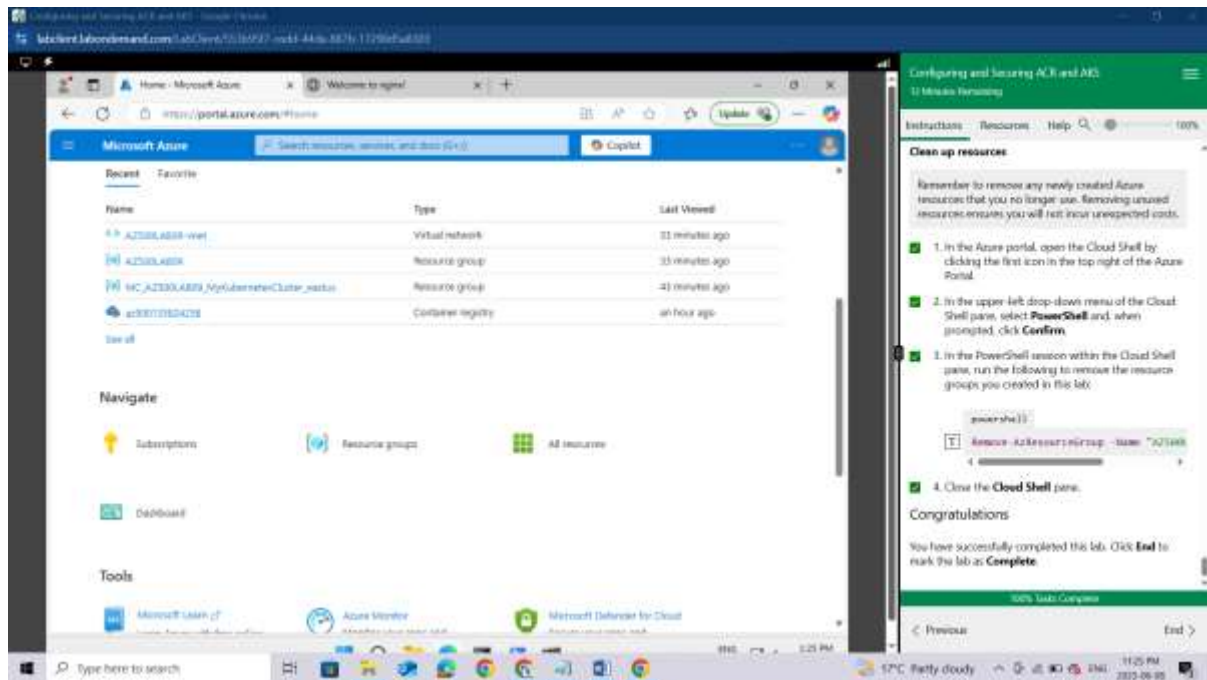
In the PowerShell session within the Cloud Shell pane, run the following to remove the resource groups you created in this lab:

```
powershell
```

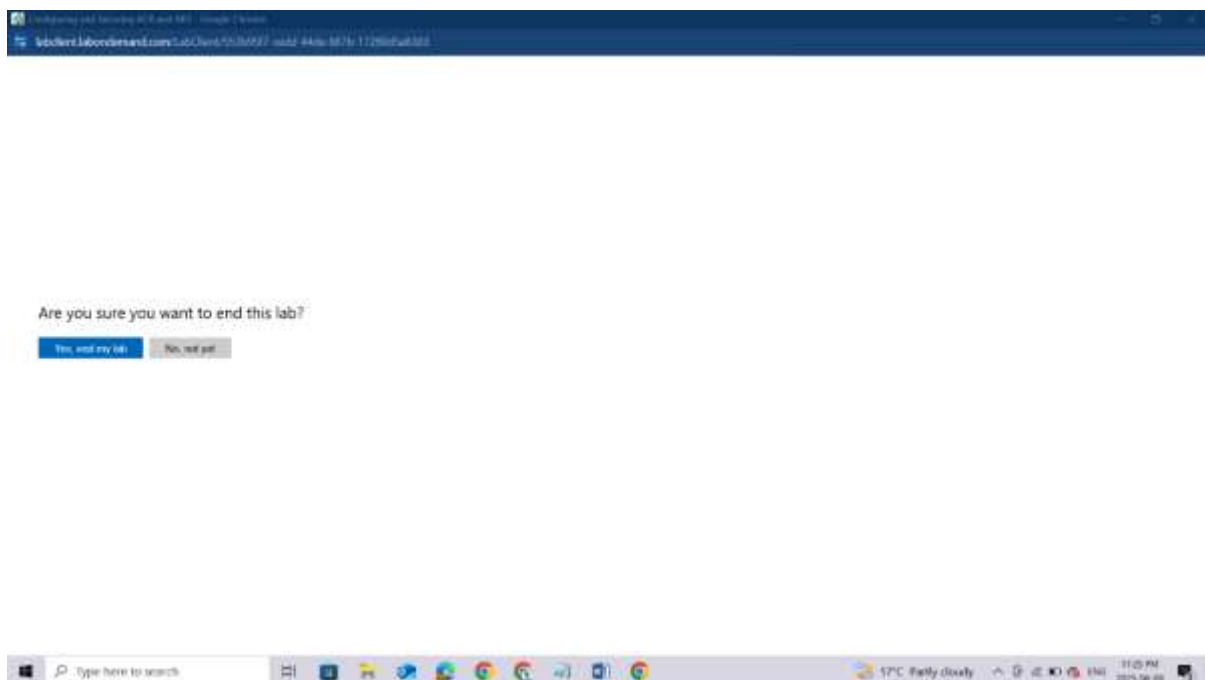
```
Remove-AzResourceGroup -Name "AZ500LAB09" -Force -AsJob
```



Close the **Cloud Shell** pane.



Congratulations, You have successfully completed this lab. Click **End** to mark the lab as **Complete**.



Conclusion

This lab provided practical insight into securing and managing containerized applications using **Azure Container Registry (ACR)** and **Azure Kubernetes Service (AKS)**. From building a Docker image and storing it in ACR to deploying and securing workloads in AKS, each step reinforced key concepts in container security and orchestration. By implementing both internal and external access controls, I learnt how to safeguard container applications against unauthorized access while maintaining functionality and scalability. The hands-on experience gained through this lab is essential for understanding real-world security challenges and best practices when working with containers in

Azure. Overall, this lab strengthened my ability to securely build, store, and deploy containers using Microsoft Azure, aligning with the core objectives of the AZ-500 certification.