Lesson 08 Demo 03

Connecting to an AKS Cluster from Azure Cloud Shell

Objective: To connect to an AKS cluster from Azure cloud shell to enhance the security and provide a consistent and accessible environment for AKS cluster related tasks

Tools required: Azure management tools

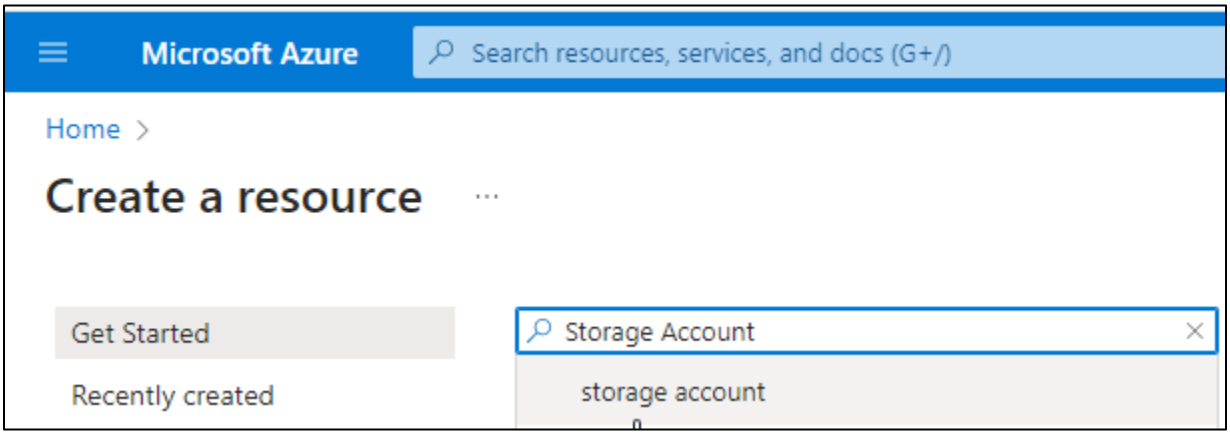
Prerequisites: An AKS cluster should already be set up (refer to the steps provided in Lesson 08, Demo 01 for guidance).

Steps to be followed:

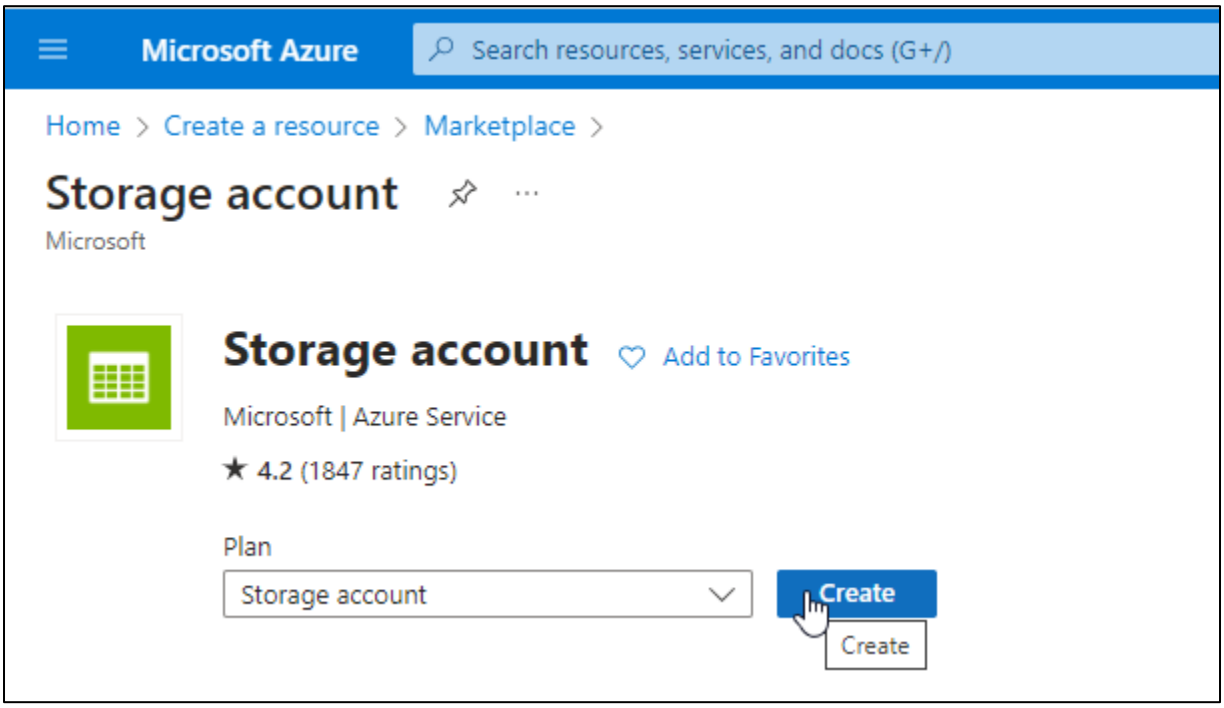
1. Create a storage account
2. Create a file share for the storage account
3. Set up the Azure Cloud Shell
4. Create a deployment and roll out an update for it

Step 1: Create a storage account

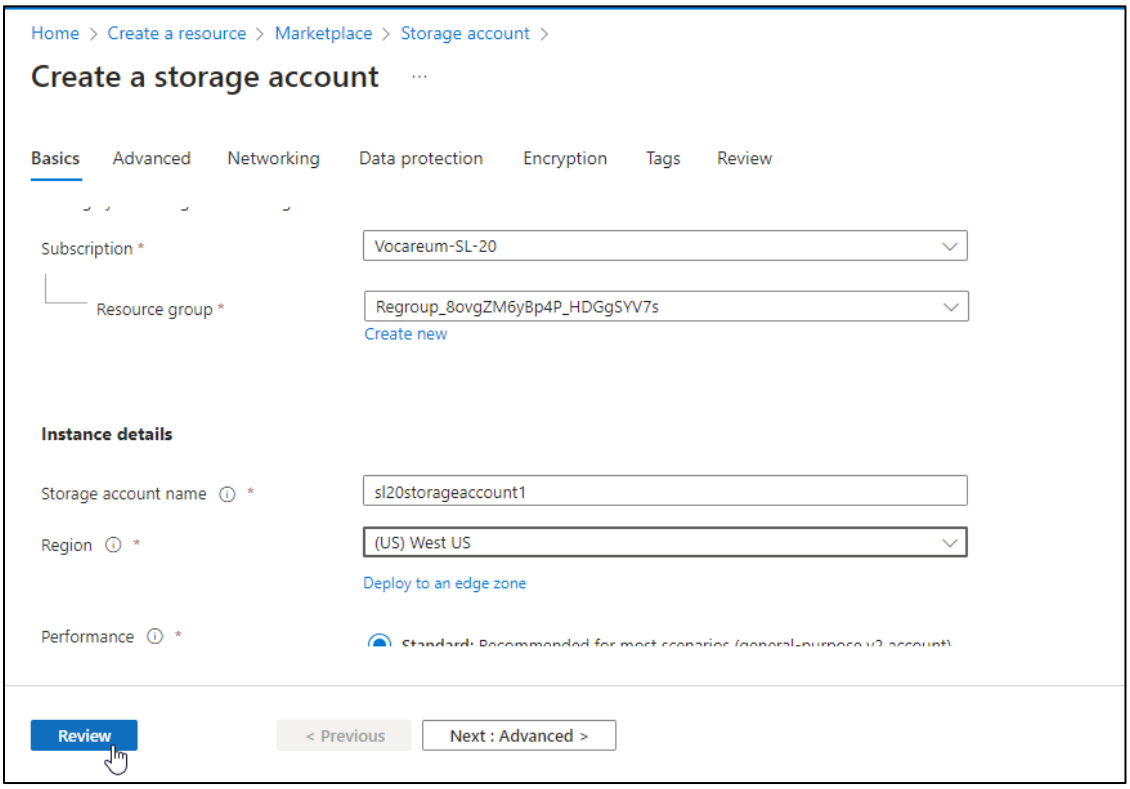
1.1 Navigate to the Azure portal, and search for and select **storage account**



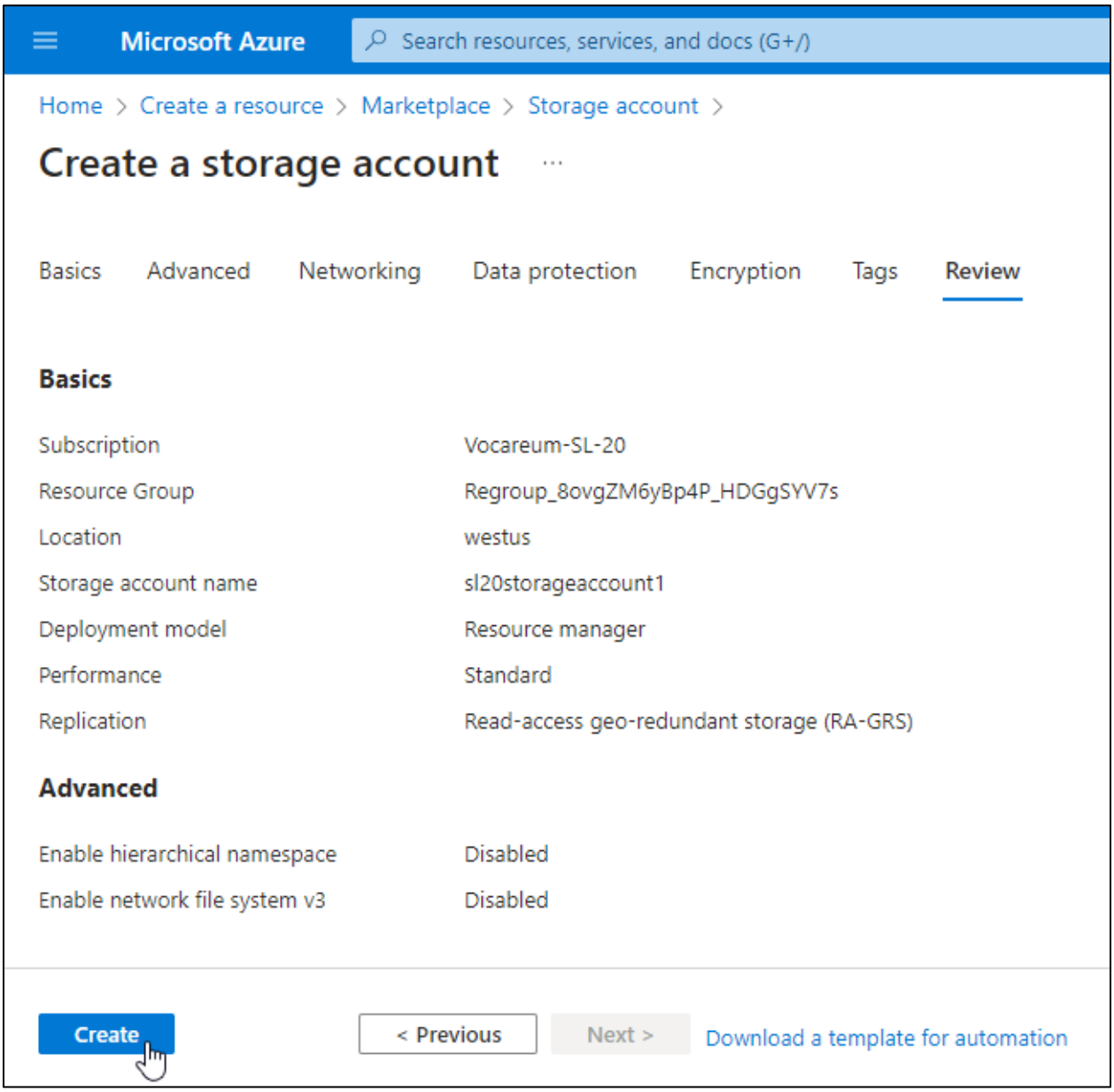
1.2 On the storage account page, click on **Create**

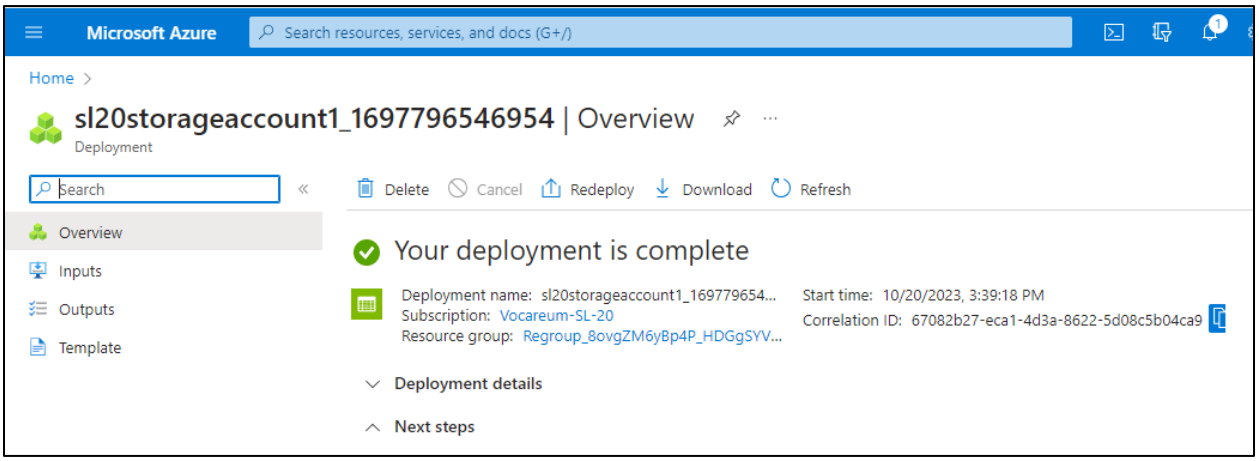


1.3 Enter the details for the storage account, and click on **Review**



1.4 Review all the settings and click on **Create**

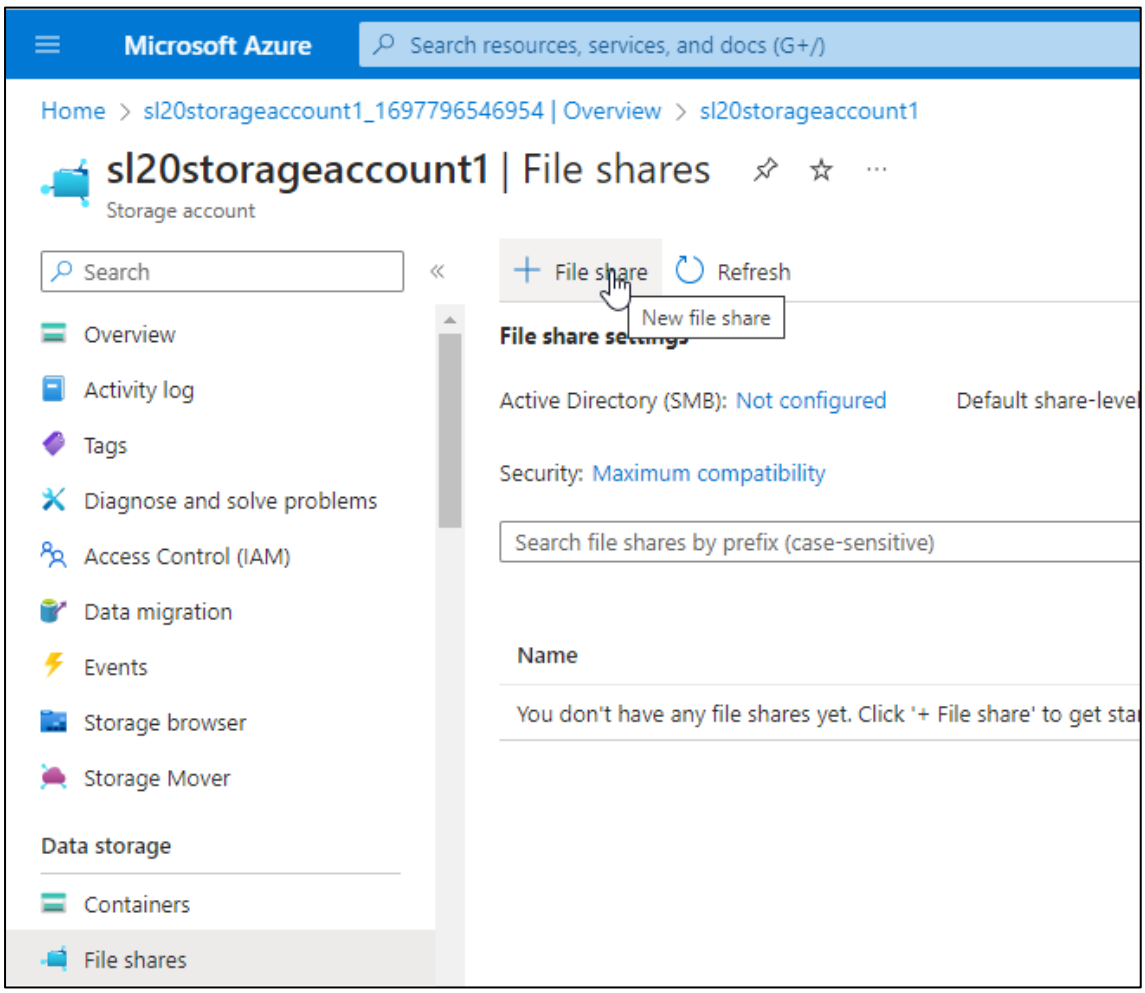




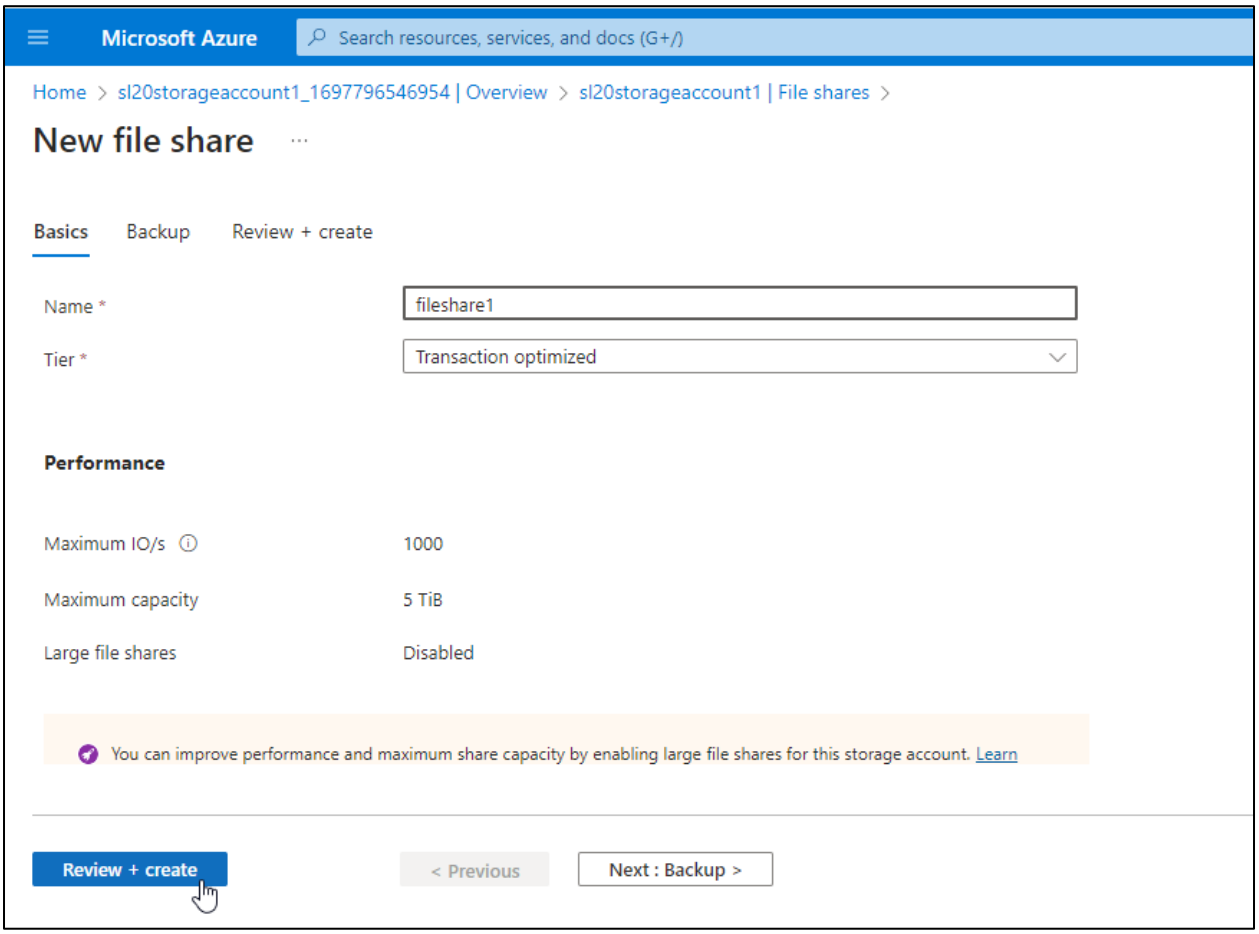
The storage account is successfully created.

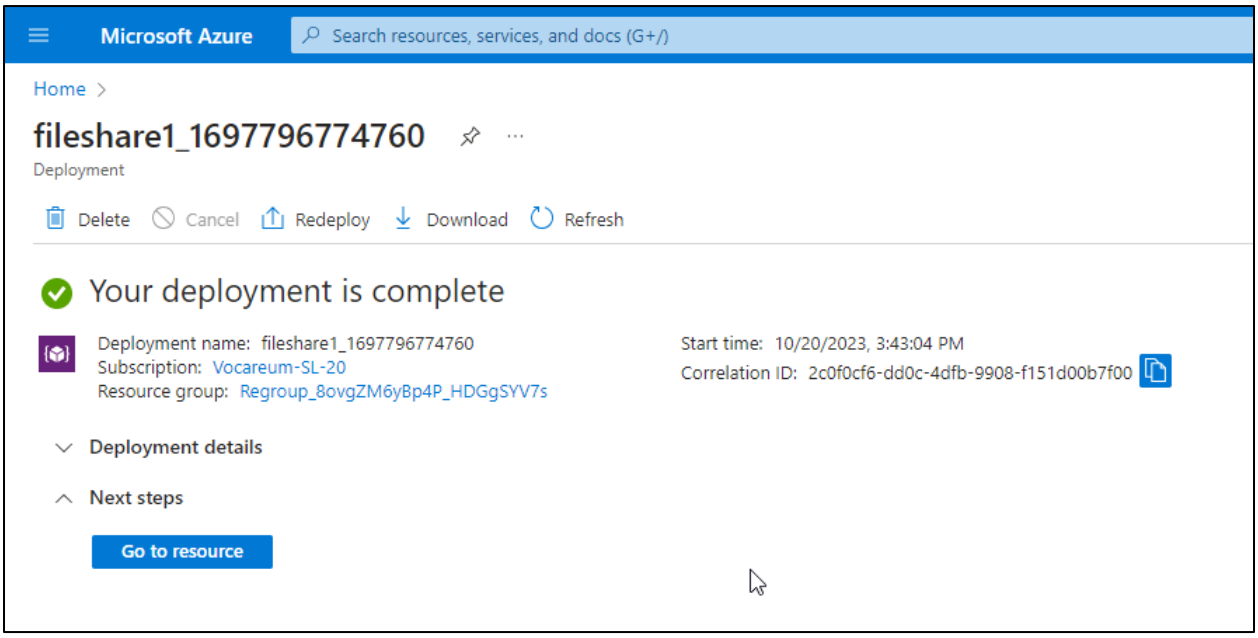
Step 2: Create a file share for the storage account

2.1 Navigate to the storage account, click **File shares** under the Data storage section, and click on **+ File share**



2.2 Provide a name to the file share and click on **Review + create**

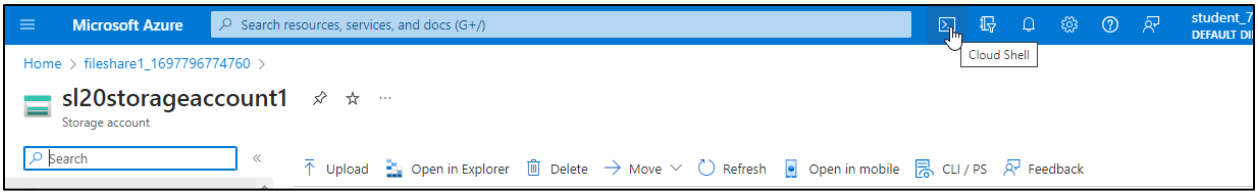




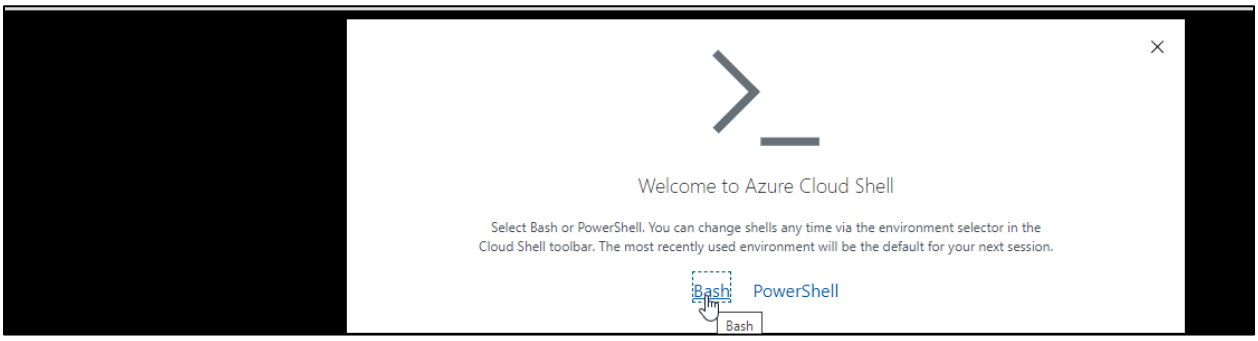
The file share is successfully created.

Step 3: Set up the Azure Cloud Shell

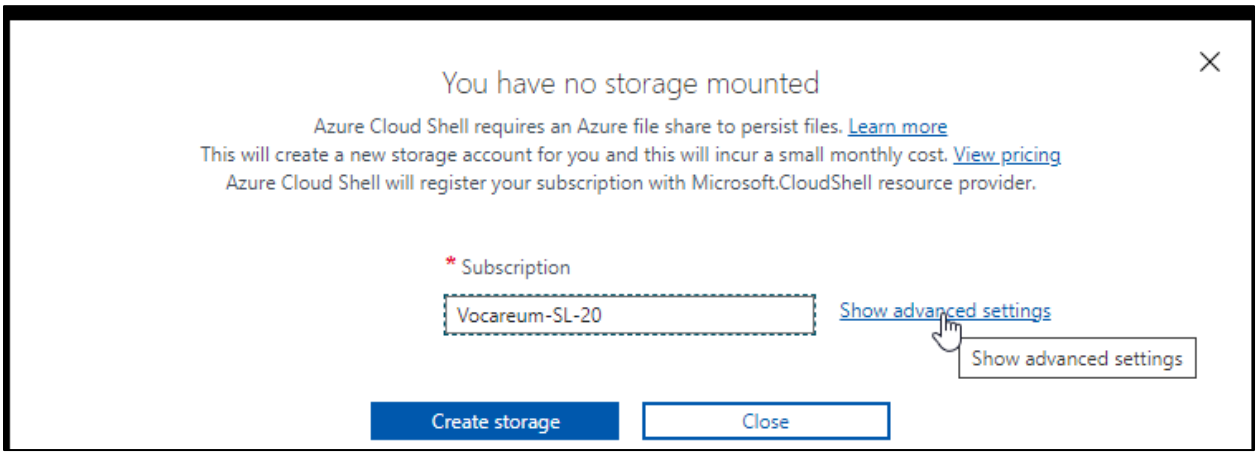
3.1 Click on the **Cloud Shell** icon from the top navigation bar



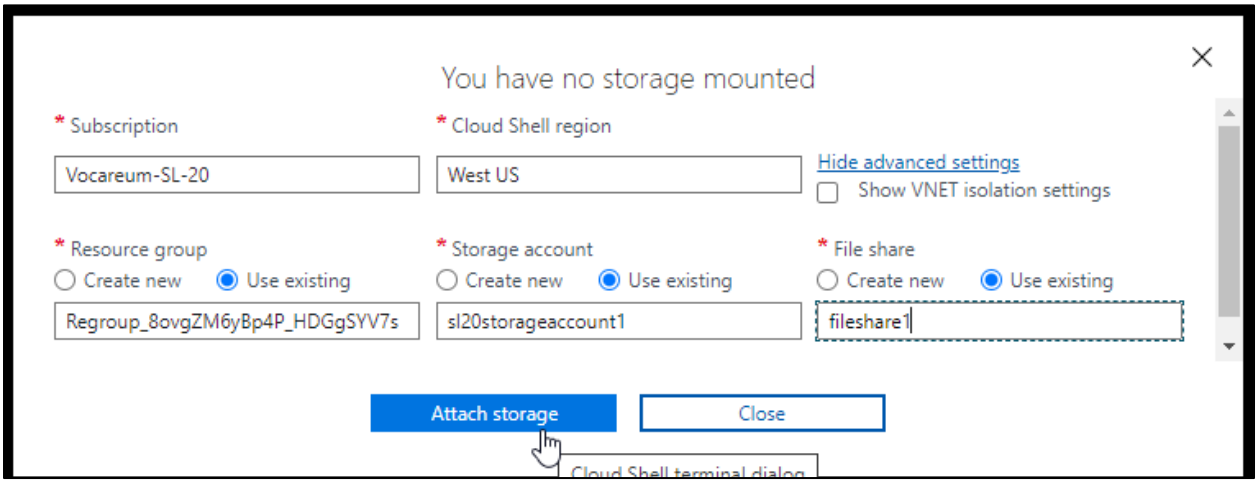
3.2 On the Azure Cloud Shell window, click on **Bash**



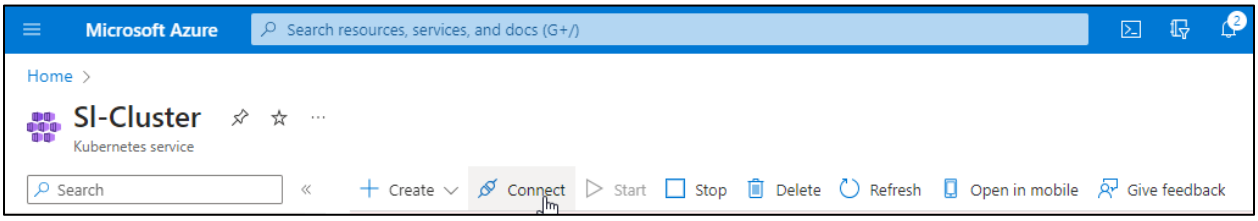
3.3 On the next screen, click on **Show advanced settings**



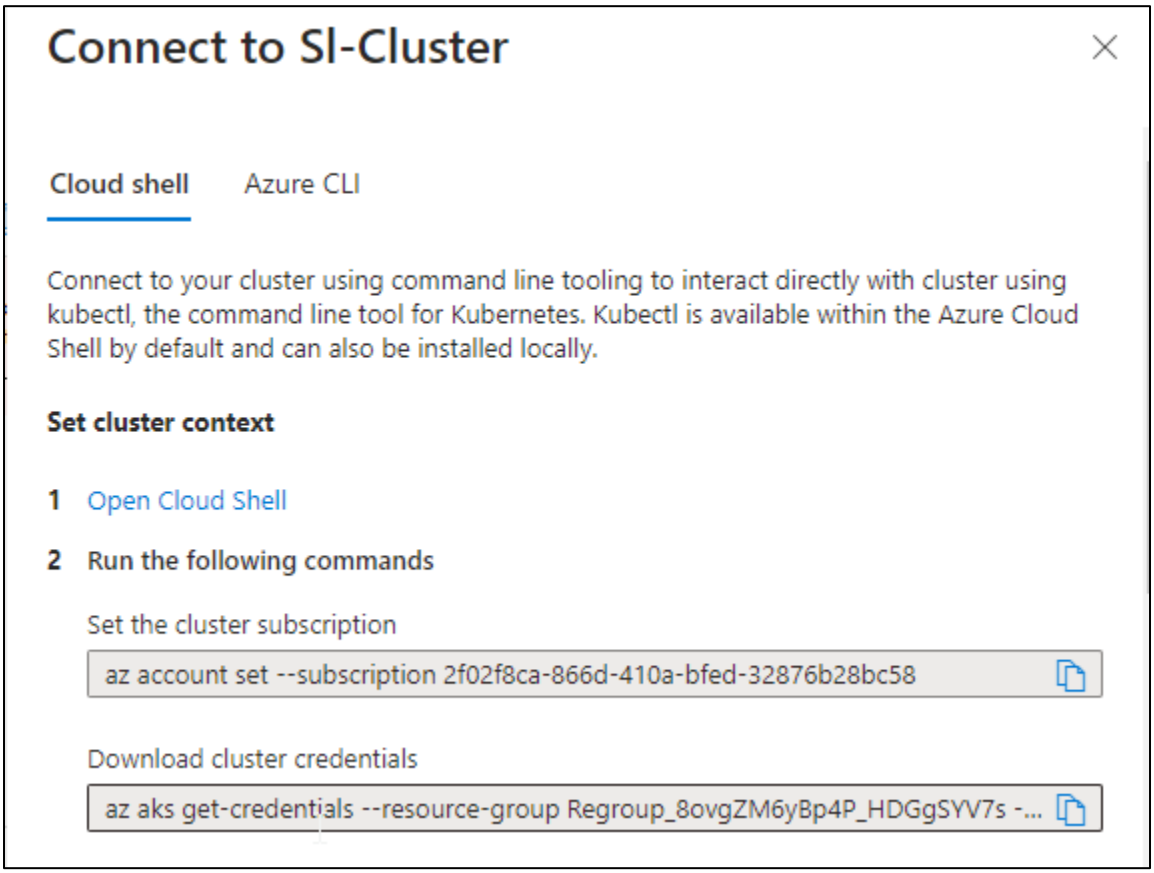
3.4 Fill in the required details as shown in the screenshot and click on **Attach storage**



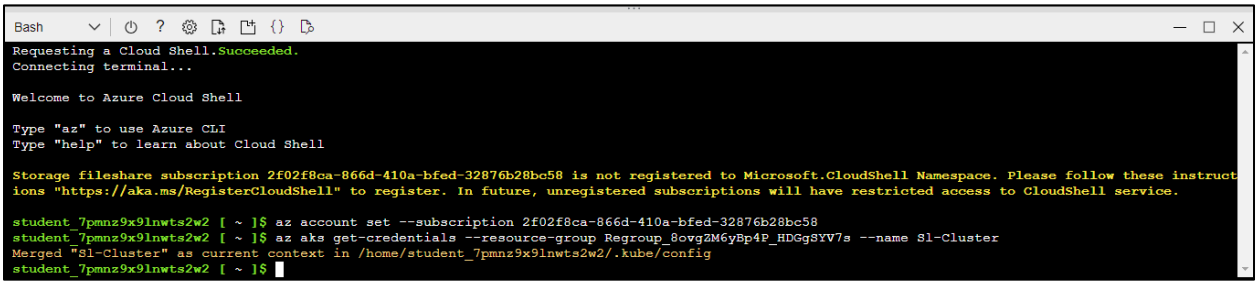
3.5 Navigate to the cluster that you have created, and click on the **Connect** button



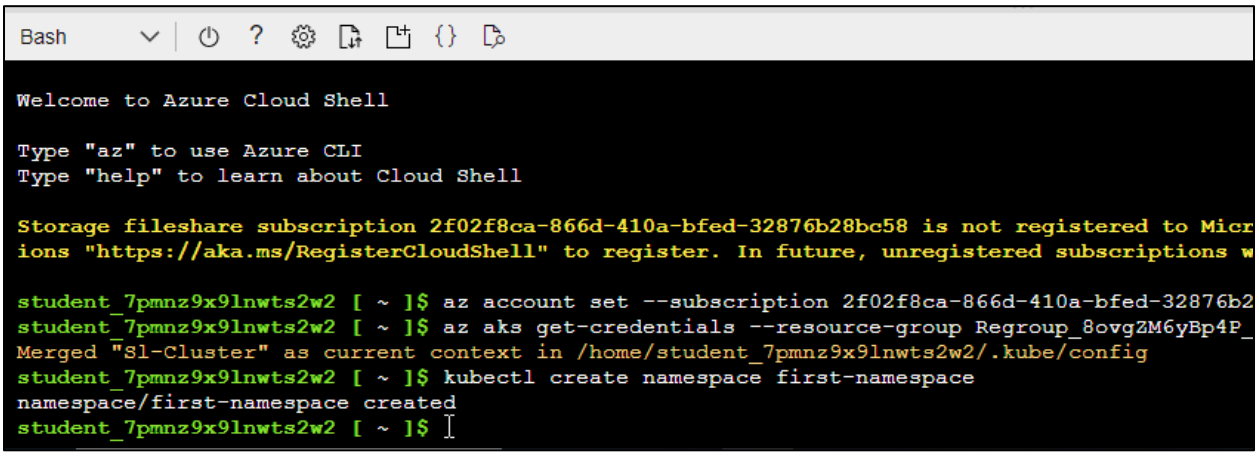
3.6 Copy the commands that you get on clicking on the Connect button



3.7 Navigate back to the bash window and paste the copied commands

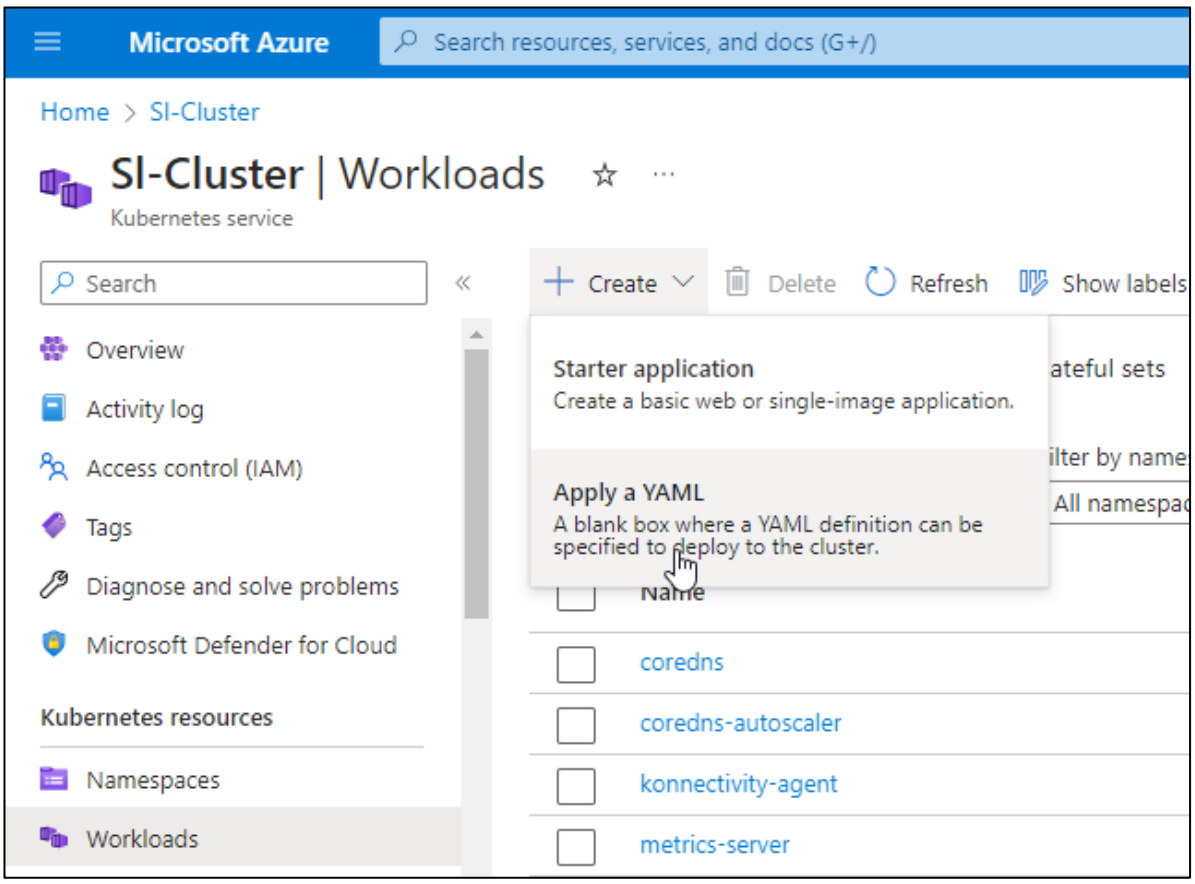


3.8 Run the command **kubectl create namespace first-namespace** to create a namespace



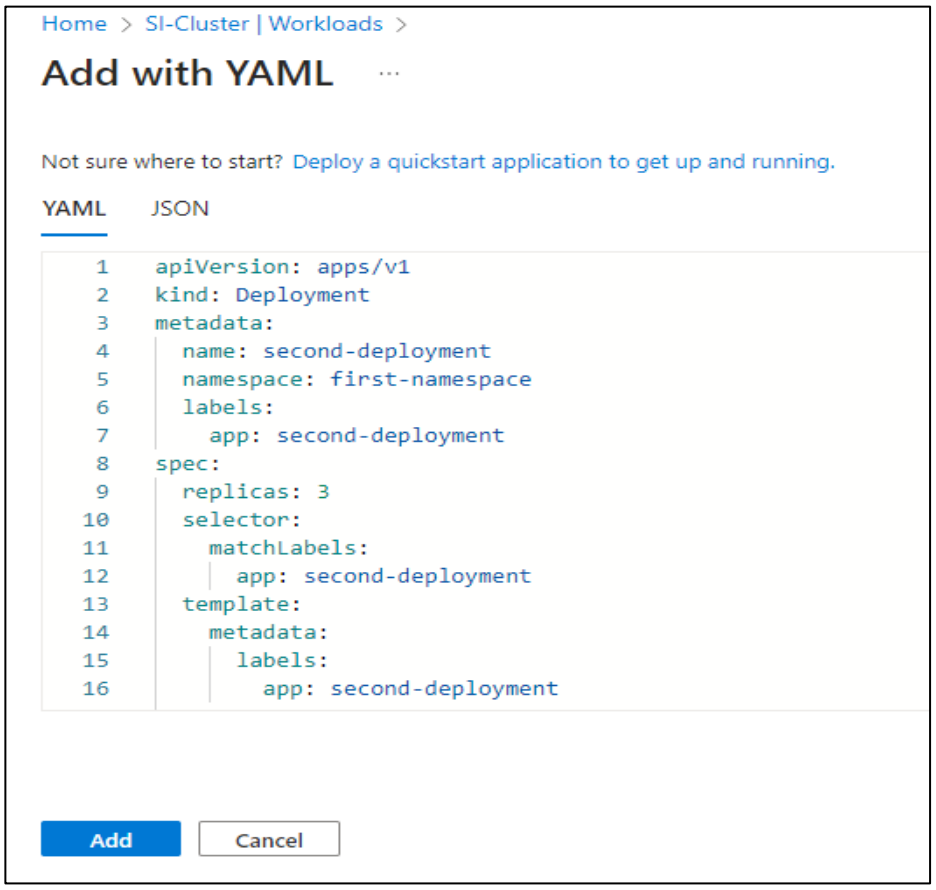
Step 4: Create a deployment and roll out an update for it

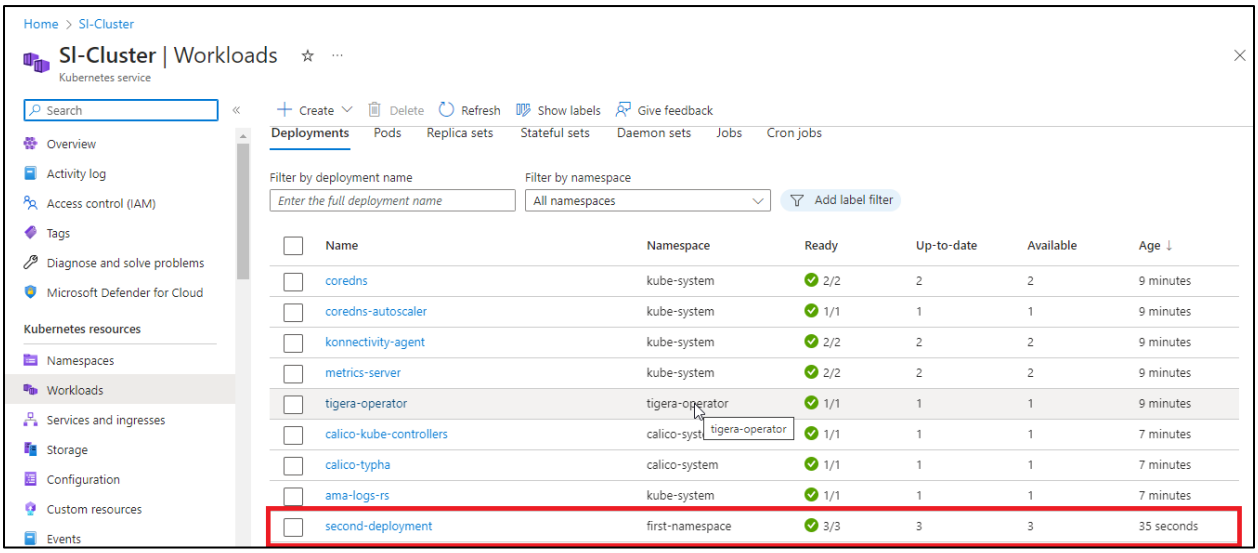
4.1 Navigate to the cluster, select **Workloads** under Kubernetes resources section, click on the **Deployments** tab, click on **Create**, and choose **Apply a YAML** option



4.2 Add the following code in the YAML section, and click on **Add**

```
apiVersion: apps/v1
kind: Deployment
metadata:
name: second-deployment
namespace: first-namespace
labels:
app: second-deployment
spec:
replicas: 3
selector:
matchLabels:
app: second-deployment
template:
metadata:
labels:
app: second-deployment
spec:
containers:
- name: nginx-container
image: nginx:1.14.2
ports:
- containerPort: 80
```

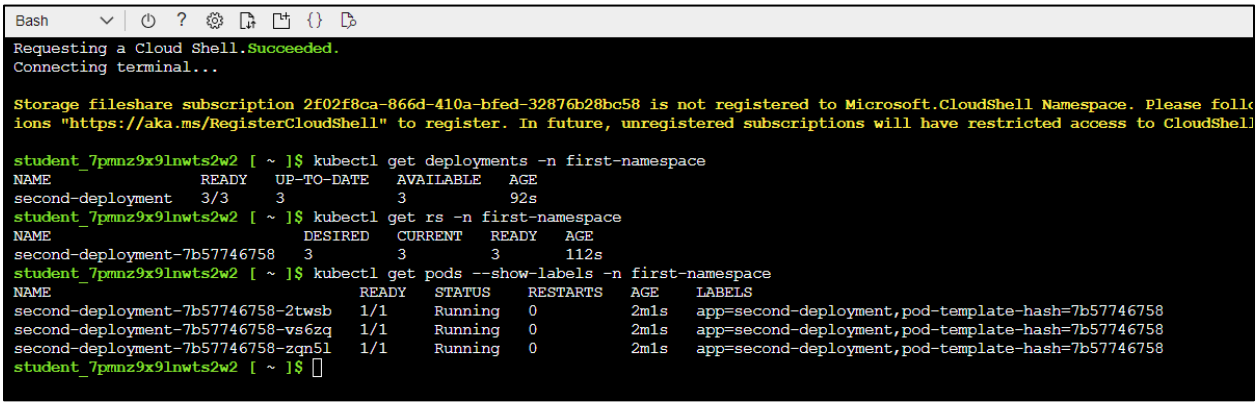




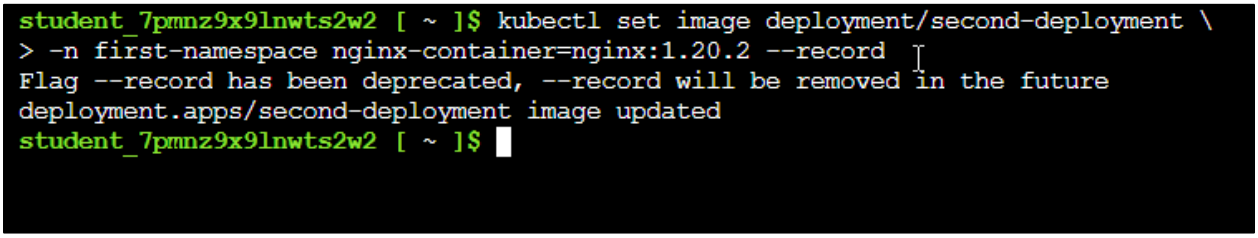
The deployment named **second-deployment** is successfully created.

4.3 Run the following commands to check the newly created deployment:

kubectl get deployments -n first-namespace
kubectl get rs -n first-namespace
kubectl get pods --show-labels -n first-namespace



4.4 Run the following commands to update the **Nginx image version** to **1.20.2** in the second-deployment:



4.5 Run the following commands to check the rollout status of the deployment and then describe the updated deployment:

kubectl rollout status deployment/second-deployment -n first-namespace

kubectl describe deployments second-deployment -n first-namespace

```
student_7pmnz9x9lnwts2w2 [ ~ ]$ kubectl rollout status deployment/second-deployment -n first-namespace
deployment "second-deployment" successfully rolled out
student_7pmnz9x9lnwts2w2 [ ~ ]$
```

```
Bash
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=second-deployment
  Containers:
   nginx-container:
    Image:        nginx:1.20.2
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  second-deployment-7b57746758 (0/0 replicas created)
NewReplicaSet:   second-deployment-547859cc5c (3/3 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  10m   deployment-controller  Scaled up replica set second-deployment-7b57746758 to 3
  Normal  ScalingReplicaSet  103s  deployment-controller  Scaled up replica set second-deployment-547859cc5c to 1
  Normal  ScalingReplicaSet  97s   deployment-controller  Scaled down replica set second-deployment-7b57746758 to 2 from 3
  Normal  ScalingReplicaSet  97s   deployment-controller  Scaled up replica set second-deployment-547859cc5c to 2 from 1
  Normal  ScalingReplicaSet  91s   deployment-controller  Scaled down replica set second-deployment-7b57746758 to 1 from 2
  Normal  ScalingReplicaSet  91s   deployment-controller  Scaled up replica set second-deployment-547859cc5c to 3 from 2
  Normal  ScalingReplicaSet  89s   deployment-controller  Scaled down replica set second-deployment-7b57746758 to 0 from 1
student_7pmnz9x9lnwts2w2 [ ~ ]$
```

Terminal container button

You can observe that the image version is **1.20.2**.

By following these steps, you have successfully connected to an AKS cluster from the Azure Cloud Shell.