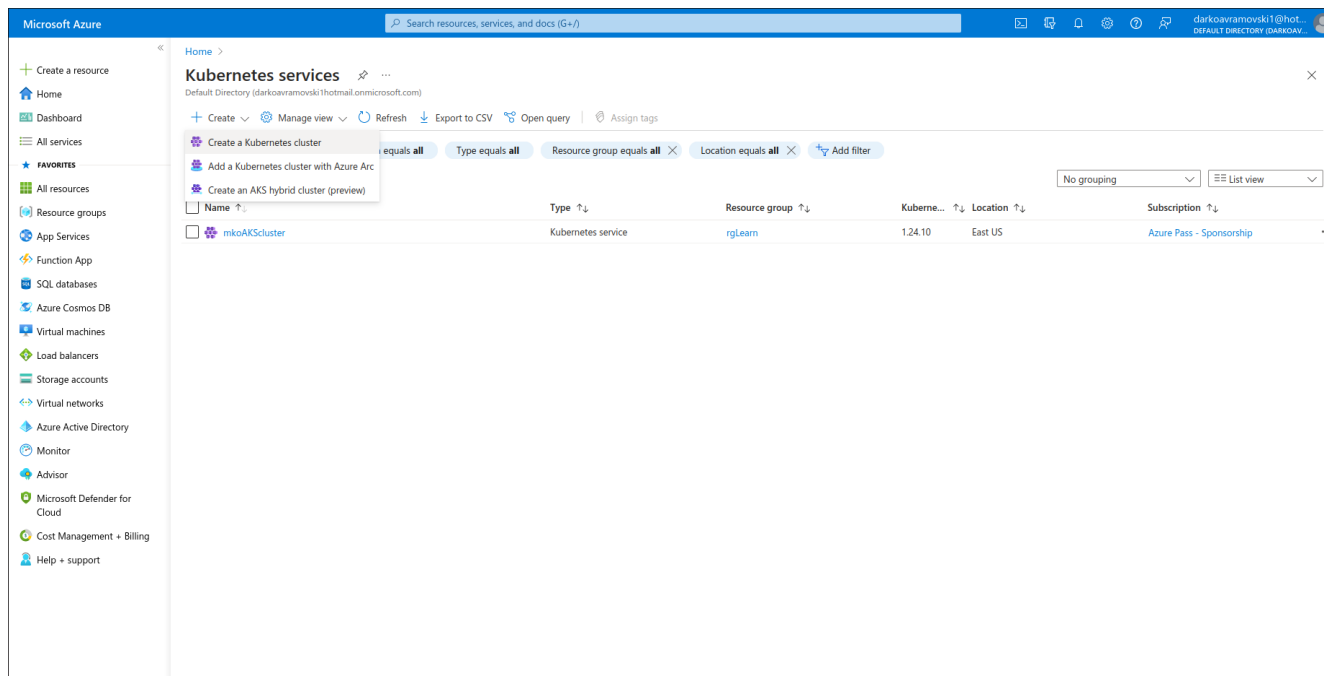
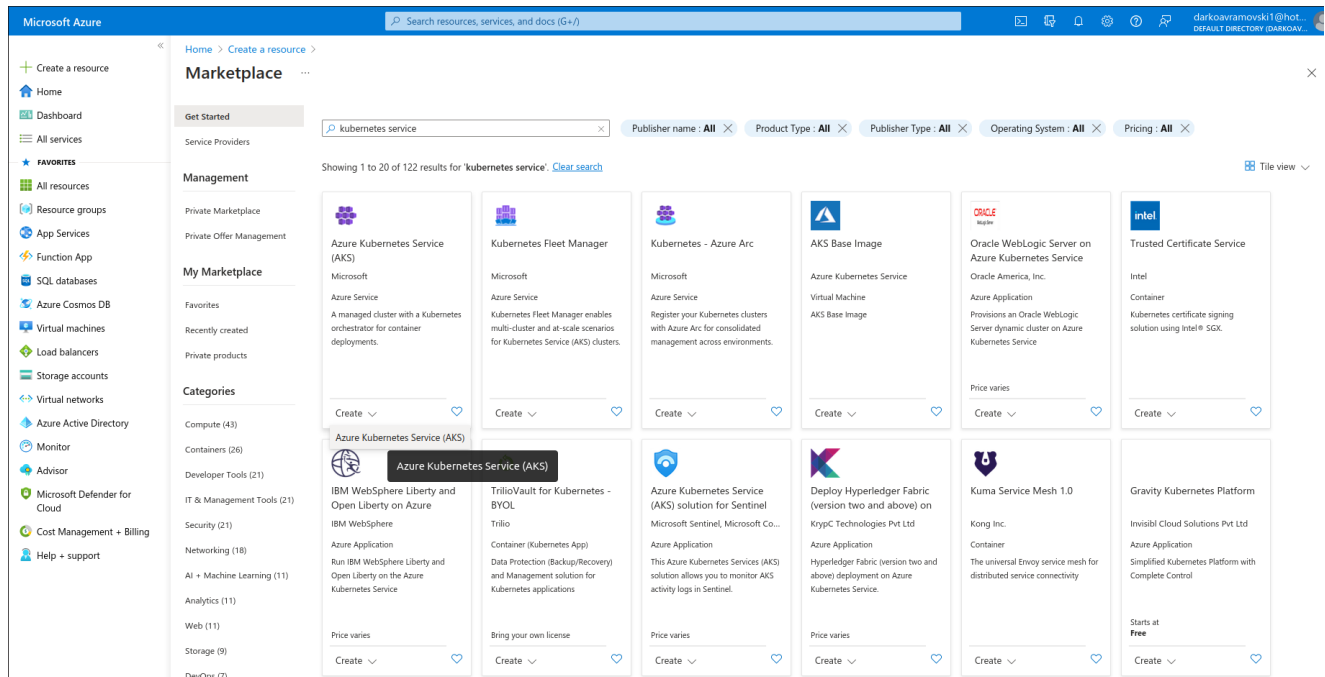


Exercise: PodsPods are the smallest, most basic deployable objects in Kubernetes. A Pod represents a single instance of a running process in your cluster. Pods contain one or more containers, such as Docker containers. Although you want deploy pods directly (static pods), knowledge for defining pods manifest files will be used for defining more complex Kubernetes resources like Controllers.

## Practice1: Simple pods operations

Login to your Azure portal and crete AKS cluster, navigate to Kubernetes services / Create Cluster



Microsoft Azure

Search resources, services, and docs (G+)

Home > Kubernetes services >

## Create Kubernetes cluster

Basics Node pools Access Networking Integrations Advanced Tags Review + create

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment, making it quick and easy to deploy and manage containerized applications without container orchestration expertise. It also eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand, without taking your applications offline. [Learn more](#)

**Project details**

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* Azure Pass - Sponsorship

Resource group \* (New) Resource group [Create new](#)

**Cluster details**

Cluster preset configuration Standard (\$\$)  
To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time. [Learn more and compare presets](#)

Kubernetes cluster name \*

Region \* (Europe) West Europe

Availability zones Zones 1,2,3  
High availability is recommended for standard configuration.







AKS pricing tier Standard

Kubernetes version \* 1.24.9 (default)

Automatic upgrade Enabled with patch (recommended)

[Review + create](#) < Previous Next : Node pools > [Feedback](#)

When we created the cluster we can connect to the cluster, click Connect new windows will popup on the right side of the screen there we have details how to connect to our cluster



darkoavramovski1@hot...  
DEFAULT DIRECTORY (DARKOAV...

## Connect to mkoAKScluster ×

Connect to your cluster using command line tooling to interact directly with cluster using kubectl, the command line tool for Kubernetes. Kubectl is available within the Azure Cloud Shell by default and can also be installed locally. [Learn more](#)

1. [Open Cloud Shell](#) or the Azure CLI
2. Run the following commands

```
az account set --subscription df86697d-88bc-4474-899b-64b5dfd1d8cf
```

```
az aks get-credentials --resource-group rgLearn --name mkoAKScluster
```

### Sample commands

Once you have run the command above to connect to the cluster, you can run any kubectl commands. Here are a few examples of useful commands you can try.

```
# List all deployments in all namespaces
kubectl get deployments --all-namespaces=true
```

```
# List all deployments in a specific namespace
# Format :kubectl get deployments --namespace <namespace-name>
kubectl get deployments --namespace kube-system
```

```
# List details about a specific deployment
# Format :kubectl describe deployment <deployment-name> --namespace
<namespace-name>
kubectl describe deployment my-dep --namespace kube-system
```

```
# List pods using a specific label
```

PROF

Next open PowerShell and Run the following commands.

```
az account set --subscription df86697d-88bc-4474-899b-64b5dfd1d8cf
az aks get-credentials --resource-group rgLearn --name mkoAKScluster
```

Microsoft Azure

Home > mkoAKScluster

Overview

Essentials

Resource group: [rgLearn](#)

Status: Succeeded (Running)

Location: East US

Subscription: [Azure Pass - Sponsorship](#)

Subscriptions ID: df86697d-88bc-4474-899b-64b5dfd1d8cf

Tags: [\(edit\)](#) [Click here to add tags](#)

Kubernetes resources

Get started Properties Monitoring Capabilities (3) 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: 1.24.10

Node sizes: Standard\_B4ms

Connect to mkoAKScluster

Connect to your cluster using command line tooling to interact directly with cluster using kubectl, the command line tool for Kubernetes. Kubectl is available within the Azure Cloud Shell by default and can also be installed locally. [Learn more?](#)

1. Open Cloud Shell or the Azure CLI

2. Run the following commands

az account set --subscription df86697d-88bc-4474-899b-64b5dfd1d8cf

az aks get-credentials --resource-group rgLearn --name mkoAKScluster

Sample commands

Once you have run the command above to connect to the cluster, you can run any kubectl commands. Here are a few examples of useful commands you can try.

# List all deployments in all namespaces

kubectl get deployments --all-namespaces=true

# List all deployments in a specific namespace

# Format: kubectl get deployments --namespace <namespace-name>

kubectl get deployments --namespace kube-system

# List details about a specific deployment

# Format: kubectl describe deployment <deployment-name> --namespace <namespace-name>

kubectl describe deployment my-dep --namespace kube-system

# List pods using a specific label

PowerShell

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI

Type "help" to learn about Cloud Shell

MOTD: Azure Cloud Shell now includes Predictive IntelliSense! Learn more: <https://aka.ms/CloudShell/IntelliSense>

VERBOSE: Authenticating to Azure ...

VERBOSE: Building your Azure drive ...

PS /home/darko> az account set --subscription df86697d-88bc-4474-899b-64b5dfd1d8cf

PS /home/darko> az account set --subscription df86697d-88bc-4474-899b-64b5dfd1d8cf

check for active pods run command

## kubectl get pods

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI

Type "help" to learn about Cloud Shell

MOTD: Azure Cloud Shell now includes Predictive IntelliSense! Learn more: <https://aka.ms/CloudShell/IntelliSense>

VERBOSE: Authenticating to Azure ...

VERBOSE: Building your Azure drive ...

PS /home/darko> kubectl get pods

No resources found in default namespace.

PS /home/darko> |

We dont see any active pods

Now check all namespaces. Run .

## kubectl get pods --all-namespaces

PowerShell

Requesting a Cloud Shell.Succeeded.

Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI

Type "help" to learn about Cloud Shell

MOTD: Azure Cloud Shell now includes Predictive IntelliSense! Learn more: <https://aka.ms/CloudShell/IntelliSense>

VERBOSE: Authenticating to Azure ...

VERBOSE: Building your Azure drive ...

PS /home/darko> kubectl get pods

No resources found in default namespace.

PS /home/darko> kubectl get pods --all-namespaces

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	azure-ip-masq-agent-4c9tw	1/1	Running	0	81m
kube-system	azure-ip-masq-agent-jbbzl	1/1	Running	0	81m
kube-system	cloud-node-manager-8vxhc	1/1	Running	0	81m
kube-system	cloud-node-manager-c4c6z	1/1	Running	0	81m
kube-system	coredns-59b6bf8b4f-dgrq2	1/1	Running	0	80m
kube-system	coredns-59b6bf8b4f-wld9z	1/1	Running	0	81m
kube-system	coredns-autoscaler-5f9cb57949-4j2q9	1/1	Running	0	81m
kube-system	csi-azuredisk-node-thfw	3/3	Running	0	81m
kube-system	csi-azuredisk-node-znztw	3/3	Running	0	81m
kube-system	csi-azurefile-node-bv7wd	3/3	Running	0	81m
kube-system	csi-azurefile-node-wq6bf	3/3	Running	0	81m
kube-system	connectivity-agent-584ccb9c4-28mpc	1/1	Running	0	59m
kube-system	connectivity-agent-584ccb9c4-phbx5	1/1	Running	0	59m
kube-system	kube-proxy-n2zvt	1/1	Running	0	81m
kube-system	kube-proxy-th7ff	1/1	Running	0	81m
kube-system	metrics-server-8655f897d8-9gxb5	2/2	Running	0	80m
kube-system	metrics-server-8655f897d8-dtkhl	2/2	Running	0	80m

PS /home/darko> |

How many pods do you see? Who deployed these pods? Why are they deployed?

Now let`s deploy the first pod imperative approach.

```
kubectl run nginx --image=nginx
```

Now we can check for active pods. run command **kubectl get pods**

```
kubectl get pods*
```

```
kubectl get pods*
```

Now we can see logs from the pod we created in powershell type

```
kubectl ngnix logs
```

```
kubectl ngnix logs
```

To view the resources consumption **kubectl top pod nginx**

```
kubectl top pod nginx
```

```
kubectl top pod nginx
```

You can check the status of the nodes and list all pods of the kube-system namespace as follows:

```
kubectl get pods -o wide
```

```
kubectl get pods -o wide
```

More detailed info about run **kubectrl describe pod nginx**


PROF

```
kubectl describe pod nginx
```

```
PowerShell
Labels:      run=nginx
Annotations: <none>
Status:      Running
IP:          10.244.1.3
IPs:         IP: 10.244.1.3
Containers:
  nginx:
    Container ID:   containerd://17ed6605baeda14e3ff8e23b5f6a9c889c924127dc99b568192068eb0ba3b10
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:2ab30d6ac53580a6db8b657abf0f68d75360ff5cc1670a85acb5bd85ba1b19c0
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Tue, 04 Apr 2023 10:54:42 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-zmbn7 (ro)
Conditions:
  Type              Status
  Initialized        True
  Ready              True
  ContainersReady    True
  PodScheduled       True
Volumes:
  kube-api-access-zmbn7:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:    kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:      true
  QoS Class:         BestEffort
  Node-Selectors:     <none>
  Tolerations:        node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                      node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason      Age   From          Message
  ----     -
  Normal   Scheduled   18m   default-scheduler   Successfully assigned default/nginx to aks-agentpool-11812048-vmss000001
  Normal   Pulling     18m   kubelet          Pulling image "nginx"
  Normal   Pulled      18m   kubelet          Successfully pulled image "nginx" in 3.004499223s
  Normal   Created     18m   kubelet          Created container nginx
  Normal   Started     18m   kubelet          Started container nginx
PS /home/darko>
```

To delete the pod that i created run **kubectl delete pod nginx**

```
kubectl delete pods nginx
```

 Create cluster k8

Let's find the image used on one of the corednspods under the kube-system namespace

```
kubectl get pods --namespace kube-system
```

PROF

```
PS /home/darko> kubectl get pods --namespace kube-system
NAME                                READY   STATUS    RESTARTS   AGE
azure-ip-masq-agent-4c9tw           1/1     Running   0           134m
azure-ip-masq-agent-jbbzl           1/1     Running   0           134m
cloud-node-manager-8vvhc             1/1     Running   0           134m
cloud-node-manager-c4c6z             1/1     Running   0           134m
coredns-59b6bf8b4f-dgrq2             1/1     Running   0           133m
coredns-59b6bf8b4f-wl09z             1/1     Running   0           134m
coredns-autoscaler-5f9cb57949-4j2q9  1/1     Running   0           134m
csi-azuredisk-node-thfkx             3/3     Running   0           134m
csi-azuredisk-node-znztw             3/3     Running   0           134m
csi-azurefile-node-bv7wd             3/3     Running   0           134m
csi-azurefile-node-wgdxf             3/3     Running   0           134m
connectivity-agent-584ccb9c4-28mpc   1/1     Running   0           112m
connectivity-agent-584ccb9c4-phbx5   1/1     Running   0           112m
kube-proxy-n2zvt                     1/1     Running   0           134m
kube-proxy-th7ff                     1/1     Running   0           134m
metrics-server-8655f897d8-9gxb5      2/2     Running   0           133m
metrics-server-8655f897d8-dtkhl      2/2     Running   0           133m
PS /home/darko>
```

Once again list all pods under all namespaces.

```
kubectl get pods --all-namespaces
```

Note one of the coredns pods. Now run `kubectl describe pod -n kube-system`. Replace the place holder with noted name.

```
kubectl describe pod coredns-59b6bf8b4f-dgrg2 --namespace kube-system
```

## Practice2: Working with pod manifest files

Now it is time to deploy pod using manifest file (declarative approach). Copy the following code block on your local computer in a file called `redis.yaml`:

Connect to your cluster, and upload the file that you created `redis.yaml` and runn **`kubectl create -f redis.yaml`** fix the erros that you got in `redis.yaml` and run again the same command.

A screenshot of a code editor with a dark theme. The editor shows a file named 'redis.yaml' with the following content:

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: static-web
5    labels:
6      role: myrole
7  spec:
8    containers:
9      - name: redis
10        image: redis123
11
```

The editor has tabs at the top: 'redis.yaml' (active), 'README.md DevOps Lab 3 - Git & GitHub', and another 'README.md DevOps Lab'. The file path 'home > ds > Desktop > ! redis.yaml' is visible in the top left.

```
apiVersion: v1
kind: Pod
metadata:
  name: static-web
  labels:
    role: myrole
spec:
  containers:
    - name: redis
      image: redis123
```

The pod is not running

```
MOTD: Azure Cloud Shell now includes Predictive IntelliSense! Learn more: https://aka.ms/CloudShell/IntelliSense

VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/darko> kubectl create -f redis.yaml
Error from server (AlreadyExists): error when creating "redis.yaml": pods "static-web" already exists
PS /home/darko> kubectl get pods
NAME          READY   STATUS             RESTARTS   AGE
static-web    0/1     ImagePullBackOff   0           23m
PS /home/darko>
```

Check the events associated with this pod. Run the `kubectl describe pod static-web` command. What are the events showing? Why your pod is not running?

we get following error

***pull access denied, repository does not exist or may require authorization: server message: insufficient\_scope: authorization failed***

```
Type      Reason      Age      From      Message
----      -
Warning   FailedScheduling 26m      default-scheduler 0/2 nodes are available: 1 node(s) had untolerated taint {node.cloudprovider.kubernetes.io/uninitialized: true}, 1 node(s) had untolerated taint {node.kubernetes.io/network-unavailable: }, preemption: 0/2 nodes are available: 2 Preemption is not helpful for scheduling.
Normal    Scheduled     25m      default-scheduler  Successfully assigned default/static-web to aks-agentpool-11812048-vmss000002
Normal    Pulling       24m (x4 over 25m) kubelet          Pulling image "redis123"
Warning   Failed        24m (x4 over 25m) kubelet          Failed to pull image "redis123": rpc error: code = Unknown desc = failed to pull and unpack image "docker.io/library/redis123:latest": pull access denied, repository does not exist or may require authorization: server message: insufficient_scope: authorization failed
Warning   Failed        24m (x4 over 25m) kubelet          Error: ErrImagePull
Warning   Failed        23m (x6 over 25m) kubelet          Error: ImagePullBackOff
Normal    BackOff       28s (x107 over 25m) kubelet          Back-off pulling image "redis123"
PS /home/darko> kubectl delete pod static-web
pod "static-web" deleted
PS /home/darko>
```

Find the correct image (check the Docker hub page) and correct it in the manifest. Locate the image information and put the correct image name. Redeploy the pod (first run `kubectl delete pod static-web` to delete the pod, then run `kubectl create` once again).

Fix the errors in `redis.yaml` file upload and run again **`kubectl create -f redis.yaml`**

```
! redis.yaml x ① README.md m
home > ds > Desktop > ! redis.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: static-web
5    labels:
6      role: myrole
7  spec:
8    containers:
9      - name: redis
10      image: redis:5.0.4
11
12
```

```
PS /home/darko> kubectl create -f redis.yaml
pod/static-web created
PS /home/darko> kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
static-web    1/1     Running   0           9s
PS /home/darko>
```

Now you can delete the pod. Try to delete it using the `kubectl delete -f redis.yaml`.



```
kubectl delete -f redis.yaml.
```

Your next task is to create and test nginx pod definition. Your definition should use the nginx official image, should use label named app with value frontend and should publish port 80. Make sure you complete this task because we will use this template in our next Labs. Your nginx pod should be running without any issues.

```
VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/darko> kubectl describe deployment frontend
Name: frontend
Namespace: default
CreationTimestamp: Wed, 05 Apr 2023 09:14:30 +0000
Labels: <none>
Annotations: deployment.kubernetes.io/revision: 1
Selector: app=nginx
Replicas: 2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType: RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels: app=nginx
  Containers:
    nginx:
      Image: nginx:1.14.2
      Port: 80/TCP
      Host Port: 0/TCP
      Environment: <none>
      Mounts: <none>
      Volumes: <none>
  Conditions:
    Type           Status Reason
    ----           -
    Available       True  MinimumReplicasAvailable
    Progressing     True  NewReplicaSetAvailable
  OldReplicaSets: <none>
  NewReplicaSet: frontend-6595874d85 (2/2 replicas created)
Events:
  Type    Reason             Age    From                      Message
  ----    -
  Normal  ScalingReplicaSet  108s   deployment-controller     Scaled up replica set frontend-6595874d85 to 2
PS /home/darko> 
```

### Practice3: Multi-container pods

Once finished you can try to create multi-container pod definition. Your multi-container pod should use redis and nginx containers with port 6379 and 80 published respectively. Label name should be app with value web.

Note that in reality there is no sense to put the redis and nginx under the same pod but it can be done for the purpose of learning.

```
redis.yaml  README.md  ! nginx.yaml x
me > ds > Desktop > ! nginx.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: webapp
5    labels:
6      name: app
7  spec:
8    containers:
9      - name: nginx-container
10        image: nginx:1.14.2
11        ports:
12          - containerPort: 80
13            hostPort: 8080
14      - name: redis
15        image: redis:5.0.4
16
17
```

```
kubectl create -f nginx.yaml
```

```
PowerShell
/usr/bin/mv: cannot stat 'nginx.yaml': No such file or directory
PS /home/darko> kubectl create -f nginx.yaml
error: unable to decode "nginx.yaml": json: cannot unmarshal string into Go struct field ObjectMeta.metadata.labels of type map[string]string
PS /home/darko> rm nginx.yaml
PS /home/darko> kubectl create -f nginx.yaml
error: error validating "nginx.yaml": error validating data: ValidationError: Terminal container button labels: invalid type for io.k8s.apimachinery.pkg.apis.meta.v1.ObjectMeta.labels: got "string", expected
if you choose to ignore these errors, turn validation off with --validate=false
PS /home/darko> rm nginx.yaml
PS /home/darko> ls
cloudrive  Microsoft  redis.yaml
PS /home/darko> kubectl create -f nginx.yaml
pod/webapp created
PS /home/darko>
```

PROF

## Verify deployments

```
error: the server doesn't have a resource type 'deployments'
PS /home/darko> kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-deployment 2/2 2 2 48m
PS /home/darko>
```

```
NAME                                READY STATUS RESTARTS AGE
nginx-deployment-6595874d85-6mt6l  1/1   Running 0      25m
nginx-deployment-6595874d85-ncxrm  1/1   Running 0      25m
Show Applications
PS /home/darko>
```

Delete all the pods under the default namespace.

Don't delete any of the manifest files you have created so far.

## Practice4: Probes

First we will create and test liveness probe with exec test.

Create a file named probes\_exec.yaml with following content: