**<u>Lab Activity 1 - Label Selectors</u>**

Deploy two pods:

One pod with a label of “Tier = Web”

```
kubectl run web --image=nginx --labels="Tier=Web"

Or

apiVersion: v1
kind: Pod
metadata:
  labels:
    Tier: App
  name: app
spec:
  containers:
  - image: nginx
    name: app
```

One pod with a label of “Tier = App”

```
kubectl run app --image=nginx --labels="Tier=App"

Or

apiVersion: v1
kind: Pod
metadata:
  labels:
    Tier: Web
  name: web
spec:
  containers:
  - image: nginx
    name: web
```

Verify the labels are applied.

```
kubectl get pods --show-labels

NAME                   READY   STATUS    RESTARTS   AGE   LABELS
app        1/1     Running   0          24s   Tier=App
Web          1/1     Running   0          71s   Tier=Web
```

**<u>Lab Activity 2 - Daemonsets</u>**

Deploy a Daemonset that leverages the nginx image

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
 name: nginx-ds
spec:
 selector:
   matchLabels:
     name: nginx-ds
 template:
   metadata:
     labels:
       name: nginx-ds
   spec:
     containers:
     - name: nginx-ds
       image: nginx
```

Verify the daemonset has been created successfully

```
kubectl get daemonset
NAME       DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
```

```
nginx-ds   2         2         2       2            2           <none>          7s
```

**<u>Lab Activity 3 - Resource Limits</u>**

Create a new namespace called “tenant-b-100mi”

```
kubectl create ns tenant-b-100mi
namespace/tenant-b-100mi created
```

Create a memory limit of 100Mi for this namespace

```
apiVersion: v1
kind: LimitRange
metadata:
  name: mem-limit-range
spec:
  limits:
  - max:
      memory: 100Mi
    type: Container
```

Create a pod with a memory request of 150Mi, ensure the limit has been set by verifying you

get a error message.

```
apiVersion: v1
kind: Pod
metadata:
  name: default-mem-demo
spec:
  containers:
  - name: default-mem-demo-ctr
    image: nginx
    resources:
      requests:
        memory: 150Mi

The Pod "default-mem-demo" is invalid: spec.containers[0].resources.requests: Invalid value: "150Mi": must be less than or equal to memory limit
```

**<u>Lab Activity 4 - Multiple Schedulers</u>**

Assume another scheduler “custom-scheduler” has been created in your environment. Configure a pod to use this scheduler.

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-web
  labels:
 role: web
spec:
  containers:
  - name: nginx
 image: nginx
  schedulerName: custom-scheduler
```

Validate the pod is using this scheduler.

```
kubectl get pod nginx-web -o yaml | grep schedulerName

{"apiVersion":"v1","kind":"Pod","metadata":{"annotations":{},"labels":{"role":"web"},"name":"nginx-web","namespace":"default"},"spec":{"containers":[{"image":"nginx","name":"nginx"}],"schedulerName":"custom-scheduler"}}

schedulerName: custom-scheduler
```

**<u>Lab Activity 5 - Schedule Pod without a scheduler</u>**

One one of the worker nodes:

Create a the directory /etc/staticpods

```
mkdir /etc/staticpods
```

Create a pod manifest file in this directory

```
apiVersion: v1
kind: Pod
metadata:
  name: staticpod
spec:
  containers:
  - name: staticpod
 image: nginx
```

Configure the kubelet service on this worker node to create pods from /etc/staticpods

```
● kubelet.service - kubelet: The Kubernetes Node Agent
  Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
 Drop-In: /etc/systemd/system/kubelet.service.d
          └─10-kubeadm.conf
  Active: active (running) since Mon 2019-04-29 12:07:52 UTC; 2min 50s ago
    Docs: https://kubernetes.io/docs/home/https://kubernetes.io/docs/home/
Main PID: 8099 (kubelet)
   Tasks: 16 (limit: 2320)
  CGroup: /system.slice/kubelet.service
          └─8099 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/config.yaml

Nano /var/lib/kubelet/config.yaml

Add the following line

staticPodPath: /etc/staticpods

sudo systemctl daemon-reload
sudo systemctl restart kubelet
```

**<u>Lab Activity 6 - Display Scheduler Events</u>**

Create a pod manifest file using the nginx image which will create a pod called “nginx-web” (Alternatively do this via kubectl run)

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

Extract the events from the cluster, particularly those pertaining to scheduling to find where this pod was scheduled to.

```
kubectl describe pod nginx-7db9fccd9b-xw6qd
```

Extract the logs from the pod running the default scheduler, or from the respective file if running as a deamon service on your master node.

```
kubectl get pods -n kube-system | grep sch
kube-scheduler-k8s-master-03         1/1  Running   0       3h1m

kubectl logs kube-scheduler-k8s-master-03 -n kube-system
```

**<u>Lab Activity 7 - Know how to configure the Kubernetes Scheduler</u>**

Configure the Kube-Scheduler by adding --logtostderr=true to the existing configuration.

```
sudo cat /etc/kubernetes/manifests/kube-scheduler.yaml

spec:
  containers:
  - command:
 - kube-scheduler
 - --bind-address=127.0.0.1
 - --kubeconfig=/etc/kubernetes/scheduler.conf
 - --leader-elect=true
 - --logtostderr=true
```

**Lab Activity 8 - Taints**

Add taint a node

```
kubectl taint nodes node-01 available=no:NoExecute
```

Pod manifest with tolerations

```
apiVersion: v1
kind: Pod
metadata:
  name: activity-8
spec:
  containers:
  - name: activity-8-ctr
    image: nginx
  tolerations:
  - key: "available"
    operator: "Equal"
    value: "no"
    effect: "NoExecute"
```

Observe behaviour by applying manifest and using `kubectl get pods -o wide` to view placement of new pod or existance of existing pod.

Pod manifest with tolerationSeconds

```
apiVersion: v1
kind: Pod
metadata:
  name: activity-8
spec:
  containers:
  - name: activity-8-ctr
    image: nginx
  tolerations:
  - key: "key"
    operator: "Equal"
    value: "value"
    effect: "NoExecute"
    tolerationSeconds: 15
```