



**ROI Training**

# **Lab:**

## **Deploying StatefulSets**

# Introduction

- In this lab, you will deploy a simple StatefulSet demo
  - The events app case study will not be used for this lab
    - But you can leave it running as is

# Verify Minikube Is Running

- From a codespaces terminal, check if Minikube is still running with:  
`minikube status`
- If it is stopped, start it again with:  
`minikube start`

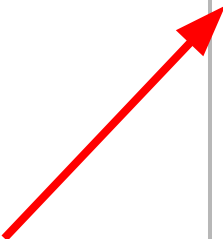
# Open the StatefulSet Example

- Change into the statefulset-demo folder  
`cd /workspaces/eventsapp/statefulset-demo/`
  - This folder was created when pulling the Git repo earlier in the course
- Open the statefulset-demo.yaml file in the editor and answer the questions on the following slide

# Investigate the YAML

- What is the name of the StatefulSet?
  - How many replicas will be created?
  - What is the mount path of the volumes?
  - What is the name of the volume claim?
- 
- Notice how the service and StatefulSet are in the same YAML
    - You can separate multiple objects in the same yaml with - - -

```
kind: Service
apiVersion: v1
metadata:
  name: statefulset-demo-service
spec:
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: LoadBalancer
- - -
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: statefulset-demo
.....
```



# Creating the StatefulSet

- Deploy the StatefulSet to the cluster:  
`kubectl apply -f statefulset-demo.yaml`
- View the pods and persistent volumes:  
`kubectl get pods`  
`kubectl get pvc`
- The pods will start up one at a time. Keep executing the previous commands until you have three pods and three volumes.
  - Or you can use the `-w` option to watch the pods or pvcs

# Storing State

- Execute into the statefulset-demo-1 pod  
`kubectl exec -it statefulset-demo-1 -- /bin/bash`
- Create a file on the persistent volume with the following commands:  
`cd /var/www/html/`  
`echo "this is a test file created on $(date)" > testfile`  
`cat testfile`  
`exit`

# Verify Data Is Persistent After Deleting Pod

- Delete the statefulset-demo-1 pod:  
`kubectl delete pod statefulset-demo-1`
  - The pod will be replaced by a new pod with the same name and the same volume will be mounted back to the new pod
- Verify the pod has been recreated and is running:  
`kubectl get pods`
- Execute into the statefulset-demo-1 pod and verify the data is still there:  
`kubectl exec -it statefulset-demo-1 -- /bin/bash`  
`cd /var/www/html/`  
`cat testfile`  
`exit`



# Verify Data Is Persistent After Deleting Pod (continued)

- Notice how even after deleting a pod, the new pod is the exact same name
  - The exec command was the same as prior to deleting the pod
  - That is one of the advantages of StatefulSets - consistent pod names

# If You Have More Time

- Feel free to experiment with the StatefulSet more if you like
- For example:
  - Try deleting the entire StatefulSet
    - This will delete all pods but not the PVCs
  - Then apply the `statefulset-demo.yaml` again
    - This will recreate the pods with the same names and remount the correct volumes
  - Execute into the `statefulset-demo-1` pod and the file will still be there

# Clean Up

- Delete the StatefulSet and PVCs with the following commands:

```
kubectl delete statefulset statefulset-demo
```

```
kubectl delete pvc hello-web-disk-statefulset-demo-0
```

```
kubectl delete pvc hello-web-disk-statefulset-demo-1
```

```
kubectl delete pvc hello-web-disk-statefulset-demo-2
```

```
kubectl delete svc statefulset-demo-service
```

# Success

- **Congratulations!** You have successfully used a StatefulSet
  - Experimented with StatefulSets to provide persistent storage to pods