ROITRAINING
MAXIMIZE YOUR TRAINING INVESTMENT

Cheat Sheet: Kubernetes Deployment

Starting a Kubernetes Cluster

- In this activity, you will be using Minikube to run your Kubernetes cluster
  - Minikube is already installed in the Cloud Shell class environment
- If you are not already, log back in to the class environment
  - https://cloudshell.roitraining.com
- In the Cloud Shell terminal, start the Kubernetes cluster with:

```
minikube start
```

  - If you are prompted to **Authorize Cloud Shell**, click **Authorize**
  - It will take 1-2 minutes to start the cluster

Starting a Kubernetes Cluster (continued)

- Once the cluster is started, verify the status with:
  minikube status
  - You should see the following:

```
$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

- Kubernetes is now ready

K8s YAMLs

- In the Cloud Shell terminal, run the following commands:

```
cd ~
mkdir kube
cd sources
cp events-app-yaml.zip ~/kube
cd ~/kube
unzip events-app-yaml.zip
```

- View the files in the **kube** folder
    - You should see four yaml files
    - Investigate the four yaml files and answer the questions on the next slide

K8s YAMLs (continued)

- What is the name of the external service load balancer?
- What port number does the external service operate on?
- What is the name of the internal service cluster IP?
- What port number does the internal service operate on?

K8s YAMLs (continued)

- Edit the `externaldeployment.yaml` file and change the image URL to match your external image URL in Docker Hub
  - Be sure to include the version number after the `:`
- Edit the `internaldeployment.yaml` file and change the image URL to match your internal image URL in Docker Hub
  - Be sure to include the version number after the `:`
- Apply all four yamls with the following commands:

```
kubectl apply -f externalservice.yaml
kubectl apply -f externaldeployment.yaml
kubectl apply -f internaldeployment.yaml
kubectl apply -f internalservice.yaml
```

Watch the Pods Create

- Run the following command to watch the pods create:

  ```
  kubectl get pods -w
  ```

  - The **`-w`** puts the command into watch mode which causes it to update the output in real time

- Leave that command running and continue with the lab

Testing the Application

- If Minikube was running on a local system, you could test the app locally
  - Since you are running it on a remote class environment, you must create a port forward to be able to test the app
- Open a new Cloud Shell terminal tab and run the following command:

```
minikube tunnel & kubectl port-forward service/demo-ui-service 8080:80
```

  - This is a blocking command—you will not get the prompt back
  - Be sure to leave it running
- Switch back to the other terminal tab and ensure both pods are running
- Test your services with the **Preview on port 8080**

Investigating the Services

- Switch back to the terminal tab that is running the watch pods
- Press CTRL+C to stop the watch command
- List the Kubernetes services with:

```
kubectl get svc
```

How to Stop the Port Forward

- **This slide is just informational, no action is required**
  - For now, you can leave your port forward running
- The following can be used to stop the port forward if needed:
  - Stop the Minikube tunnel:

    ```
    pkill -f "minikube tunnel"
    ```

  - Switch to the terminal tab running the port forward and press CTRL+C to stop it
- The port forward would not be needed if running a local Minikube or a Kubernetes cluster that is available on the internet

Success!

- **Congratulations!** You have successfully deployed the case study to Kubernetes.

If you have time

Updating Your Git Repo

- If you have time, add all new files to your Git repo
  - You may need to copy them into the correct folder
  - Add, commit, and push them
  - Don’t forget your Kubernetes yaml and Dockerfiles

If you have time

Pod Autoscaling

- Here is an example of a Horizontal Pod Autoscaler
  - Try modifying this file for your external service and apply it

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
    name: my-autoscaler
spec:
    scaleTargetRef:
        apiVersion: apps/v1
        kind: Deployment
        name: my-deployment
    minReplicas: 2
    maxReplicas: 5
    targetCPUUtilizationPercentage: 60
```