EXPERIMENT 8

AIM: Working with Kubernetes (Single Node Cluster)

Steps to Complete:

Step 1 - Start Minikube

Minikube has been installed and configured in the environment. Check that it is properly installed, by running the *minikube version* command:

```
minikube version
```

Start the cluster, by running the *minikube start* command:

```
minikube start --wait=false
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual machine for you, and a Kubernetes cluster is now running in that VM.

Step 2 - Cluster Info

The cluster can be interacted with using the *kubectl* CLI. This is the main approach used for managing Kubernetes and the applications running on top of the cluster.

Details of the cluster and its health status can be discovered via

```
kubectl cluster-info
```

To view the nodes in the cluster using

```
kubectl get nodes
```

If the node is marked as **NotReady** then it is still starting the components.

This command shows all nodes that can be used to host our applications. Now we have only one node, and we can see that it's status is ready (it is ready to accept applications for deployment).

Step 3 - Deploy Containers

With a running Kubernetes cluster, containers can now be deployed.

Using `kubectl run`, it allows containers to be deployed onto the cluster –

```
kubectl create deployment first-deployment --image=katacoda/docker-http-server
```

The status of the deployment can be discovered via the running Pods –

```
kubectl get pods
```

Once the container is running it can be exposed via different networking options, depending on requirements. One possible solution is NodePort, that provides a dynamic port to a container.

```
kubectl expose deployment first-deployment --port=80 --type=NodePort
```

The command below finds the allocated port and executes a HTTP request.

```
export PORT=$(kubectl get svc first-deployment -o go-template='{{range.spec.ports}}{{if .nodePort}}{{.nodePort}}{{"\n"}}{{end}}{{end}}') echo "Accessing host01:$PORT" curl host01:$PORT
```

The result is the container that processed the request.

Step 4 - Dashboard

Enable the dashboard using Minikube with the command

```
minikube addons enable dashboard
```

Make the Kubernetes Dashboard available by deploying the following YAML definition. This should only be used on Katacoda.

```
kubectl apply -f /opt/kubernetes-dashboard.yaml
```

The Kubernetes dashboard allows you to view your applications in a UI. In this deployment, the dashboard has been made available on port *30000* but may take a while to start.

To see the progress of the Dashboard starting, watch the Pods within the *kube-system* namespace using

```
kubectl get pods -n kubernetes-dashboard -w
```

Once running, the URL to the dashboard is https://2886795282-30000-simba09b.environments.katacoda.com/