Cloud Computing

Introduction to Kubernetes

Exercise Three - Scale up your App (3%)

Task 1: I deleted the previous minikube container by running the command: **Minikube delete –all** as you can see in the image below.

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>minikube delete --all
* Deleting "minikube" in docker ...
* Removing C:\Users\waleed\.minikube\machines\minikube ...
* Removed all traces of the "minikube" cluster.
* Successfully deleted all profiles
```

I then started a new minikube cluster with two nodes using the following command:

Minikube start –nodes 2

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>minikube start --nodes 2
* minikube v1.27.0 on Microsoft Windows 10 Pro 10.0.19043 Build 19043
! Kubernetes 1.25.0 has a known issue with resolv.conf. minikube is using a workaround that should work for most use cases.
! For more information, see: https://github.com/kubernetes/kubernetes/issues/112135
* Automatically selected the docker driver
* Using Docker Desktop driver with root privileges
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.25.0 on Docker 20.10.17 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass

* Starting worker node minikube-m02 in cluster minikube
* Pulling base image ...
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Found network options:
  - NO_PROXY=192.168.49.2
  - no_proxy=192.168.49.2
* Preparing Kubernetes v1.25.0 on Docker 20.10.17 ...
  - env NO_PROXY=192.168.49.2
* Verifying Kubernetes components...
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

I then checked if the we had 2 nodes available using the command: **Kubectl get nodes**

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl get nodes
NAME           STATUS   ROLES           AGE     VERSION
minikube       Ready    control-plane   6m53s   v1.25.0
minikube-m02   Ready    <none>          5m51s   v1.25.0
```

Here we can see there are two nodes that are ready.

Task 2:

Deployment: Declarative updates for Pods and ReplicaSets are offered by a deployment, So, a Kubernetes deployment instructs Kubernetes on how to produce new instances of the pods that house containerized applications or alter existing ones. Deployments enable the controlled rollout of updated code, the efficient scaling of the number of replica pods, and the ability to revert to a previous deployment version when necessary.

1**. kind:** lets you run Kubernetes on your local computer. This tool requires that you have Docker installed and configured.

2. **replicas:** ensures that a desired number of Pods with a matching label selector are available and operational. So, if replicas are set to 2, it will ensure there are 2 pods.

3. **replicas.strategy:** strategy is any technique employed to successfully launch a new version of the software solution they provide.

4. **spec.template.spec.affinity:** affinity is a set of rules used by the scheduler to determine where a pod can be placed. The rules are defined using custom labels on nodes and label selectors specified in pods. Node affinity allows a pod to specify an affinity (or anti-affinity) towards a group of nodes it can be placed on.

5**. spec.template.spec.containers:** creates a container.

Task 3: To make a new deployment to Kubernetes using the aforementioned configuration file. I used the command: kubectl apply –f hello-deployment.yaml

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl apply -f hello-deployment.yaml
deployment.apps/hello created
```

Now we ran the command: **kubectl get pods -o wide**

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE   IP           NODE           NOMINATED NODE   READINESS GATES
hello-7fb88bf7b6-r7dzq   1/1     Running   0          4m    10.244.1.2   minikube-m02   <none>           <none>
hello-7fb88bf7b6-xdwbg   1/1     Running   0          4m    10.244.0.3   minikube       <none>           <none>
```

Here will see that the two pods are running to two different nodes. But we want to change that and assign the two pods to one specific node so we follow the steps below.

(a) Delete the previous deployment and make sure no other pod is running

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl delete deploy hello
deployment.apps "hello" deleted
```

I then run the command kubectl get pods –o wide to check if we have any pods running

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl get pods -o wide
No resources found in default namespace.
```

(b) You should associate the preferred node with a label. You can do so using the command **kubectl label nodes minikube-m02 disktype=secondnode.**

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl label nodes minikube-m02 disktype=secondnode
node/minikube-m02 labeled
```

I set the minikube-m02 label to “secondnode”.

(c) View the new label using the command: **kubectl get nodes –-show-labels**

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl get nodes --show-labels
NAME           STATUS   ROLES           AGE    VERSION   LABELS
minikube       Ready    control-plane   116m   v1.25.0   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=minikube,kubern
7842a7d34b6767bbdac2b,minikube.k8s.io/name=minikube,minikube.k8s.io/primary=true,minikube.k8s.io/updated_at=2022_10_12T18_25_40_0700,minikube.k8s.io/version=v1.27.0,node-role.kub
xternal-load-balancers=
minikube-m02   Ready    <none>          115m   v1.25.0   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,disktype=secondnode,kubernetes.io/arch=amd64,kubernetes.io/host
```

(d) Create a new configuration file with the name "hello–deployment_updated.yaml" Copy inside it the code from "hello–deployment.yaml", and make adjustments to associate these two pods with the node that has been received the new label. I used the website to help with adjustments [assign-pod-node]

So, the adjustments I made to the hello-deployment.yaml file was to change the podAntiAffinity to nodeAffinity which allows us to constrain which nodes your Pod can be scheduled on based on node labels. I also deleted

```
      - labelSelector:
            topologyKey: "kubernetes.io/hostname"
```

And replaced label selector with nodeSelectorTerms and then added in new matchExpressions

```
key: disktype
operator: In
values:
- secondnode
```

I then ran the command kubectl apply –f hello-deployment_updated.yaml and created the deployment and then checked for pods which are now set to one node this is because in the code we used secondnode as the key which was the label for the node minikube-m02.

```
C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl apply -f hello-deployment_updated.yaml
deployment.apps/hello created

C:\Users\waleed\Desktop\PRACTICAL3_waleed_wazir_19396951\ex3>kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE     IP           NODE           NOMINATED NODE   READINESS GATES
hello-6f9c4986d8-6bntp   1/1     Running   0          4m32s   10.244.1.3   minikube-m02   <none>           <none>
hello-6f9c4986d8-wff2h   1/1     Running   0          4m32s   10.244.1.4   minikube-m02   <none>           <none>
```