Docker and Kubectl setup commands

Saturday, 10 June 2023 11:47 am

Project A

1. Create a new account on dockerhub. My account details are:
   Username: sama863nz

2. Clone the django project from **github** on **Ubuntu linux**

3. Create an **image** in docker
   Docker build -t sama863nz/django-todo:latest .

   Where
   t: tag
   sama863nz: docker account
   Django-todo: is the name of the image, which I have chosen
   Latest: because I want to tag it as latest
   . Is the destination folder

4. Create a docker **container** for the above image
   Docker run -d -p 8000:8000 sama863nz/django-todo:latest

   Where
   -p is the port which is chosen as 8000:8000

5. To test the port allocated
   Ss -tuln

6. Test if docker is running on Ubuntu linux server
   Curl -L http://44.211.81.73:8000
   44.211.81.73 (IP address public)

   Error: the connection was not getting established to the above IP address.
   Solution: create an inbound rule in "security groups" for RDP. If RDP does not work then use "all traffic".

7. Push the docker image to docker hub
   Docker login (give your username & password as sama863nz and S@njeev863 respectively)

   Docker push sama863nz/django-todo

8. Create pods: **vim sam-pod.yaml**
   Create a .yaml file as sam-pod.yaml and copy paste the code structure from internet (got to internet search and type pod kubernetes and copy the code)

   ```
   apiVersion: v1
   kind: Pod
   metadata:
     name: nginx  (replace with todo-pod)
   spec:
     containers:
     - name: nginx  (replace with todo-app)
       image: nginx:1.14.2    (replace with docker pull sama863nz/django-todo:latest image name from docker hub)
       ports:
       - containerPort: 80  (replace with 8000)
   ```

9. Unassign port 8000 from the docker container. For this do
   Docker ps (to figure out container ID)
   Docker kill <container ID>

   Kubectl apply -f sam-pod.yaml (this will create the pod. So kubectl created the pod)

10. Create deployment file on Kubernetes
    Vim sam-deployment.yaml

    ```
    apiVersion: apps/v1
    kind: Deployment
    metadata:
      name: todo-deployment
      labels:
        app: todo-app
    spec:
      replicas: 3
    ```

```
  selector:
    matchLabels:
      app: todo-app
  template:
    metadata:
      labels:
        app: todo-app
    spec:
      containers:
      - name: todo-app
        image: sama863nz/django-todo:latest
        ports:
        - containerPort: 8000
```

vim editor shortcuts:

:set paste
Your code
:set nopaste

Esc then ggVG (to select all lines) then delete

To cancel or exit without saving
Esc + : + q!

To check the system usage in docker
Docker system df

To reset
Docker system prune -a

Use nano editor instead of vim

Before exiting remove paste and nopaste lines from the vim editor

11. Run the deployment file
    Kubectl apply -f sam-deployment.yaml (currently getting error in pod)

11. Test if the pods are ready and available
    Kubectl get deployments
    Kubectl get pods

12. Test to connect the pod which is running the image
    Kubectl get pods -o wide (this will give the IP address of the pod)
    Curl -L http://<ip address from above>

    The above curl will not be able to connect because the pod is inside the kubernetes minikube cluster. For this, a service file needs to be created to communicate from outside to the pod:

    Search for "service kubernetes"
    Go to "NodePort " and copy the code and paste it in a new file by name sam-service.yaml in ubuntu

```
apiVersion: v1
kind: Service
metadata:
  name: todo-service
spec:
  type: NodePort
  selector:
    app: todo-app
  ports:
    # By default and for convenience, the `targetPort` is set to the same value as the `port` field.
   - port: 80
     targetPort: 8000
     # Optional field
     # By default and for convenience, the Kubernetes control plane will allocate a port from a range (default: 30000-32767)
     nodePort: 30007
```

Kubectl appy -f sam-service.yaml

Kubectl get service (to test the service). This will give the internal "Cluster_IP"

Now, we need minikube (kubernetes cluster) IP which we can get by command

Minikube service <service name from above> --url
e.g. minikube service todo-service --url

Call this from outside:
Curl -L http://<ip address from above>

Auto-Healing test:

Kubectl get pods
Kubectl delete pod <NAME> (to delete a pod)
Kubectl get pods (to test)

Note a new pod has been auto created when an existing pod has been deleted

Host IP allocation:

goto file /etc/hosts
Edit this file with the above url from curl

Sudo nano hosts

e.g. in the last line (allhosts) mention the new url as 192.168.49.2 sam-todo-app.com (no http or port)

To run the deployed app:
Curl -L http://sam-todo-app.com:30007