Assignment 4 - B11-5A1E

Manoj Kumar M

BE CSE, Batch P

College of Engineering, Guindy, Anna University

**Assignment Kubernetes / Docker**

1.Pull an Image from docker hub and run it in docker playground.

2.Create a docker file for the jobportal application and deploy it in Docker desktop application.

3.Create an IBM container registry and deploy helloworld app or jobportalapp.

4.Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport

**1.Pull an Image from docker hub and run it in docker playground.**

```
[node1] (local) root@192.168.0.18 ~
$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:faa03e786c97f07ef34423fccceeec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
[node1] (local) root@192.168.0.18 ~
$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

## 2.Create a docker file for the jobportal application and deploy it in Docker desktop application.

```
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ sudo bash start.sh
Sending build context to Docker daemon  9.728kB
Step 1/6 : FROM tiangolo/uwsgi-nginx-flask:python3.6-alpine3.7
 ---> cdec3b0d8f20
Step 2/6 : RUN apk --update add bash nano
 ---> Using cache
 ---> b993777b9b18
Step 3/6 : ENV STATIC_URL /static
 ---> Using cache
 ---> 4d6b973b9dd8
Step 4/6 : ENV STATIC_PATH /var/www/app/static
 ---> Using cache
 ---> 47a61ea49469
Step 5/6 : COPY ./requirements.txt /var/www/requirements.txt
 ---> 37520174bf86
Step 6/6 : RUN pip install -r /var/www/requirements.txt
 ---> Running in ad23fd567b37
Collecting Flask==1.0.2 (from -r /var/www/requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/7f/e7/08578774ed4536d3242b14dacb4696386634607af824ea997202cd0edb4b/Flask-1.0.2-py2.py3-none-an
.whl (91kB)
Requirement already satisfied: click>=5.1 in /usr/local/lib/python3.6/site-packages (from Flask==1.0.2->-r /var/www/requirements.txt (line 1)) (7.1.
)
Requirement already satisfied: itsdangerous>=0.24 in /usr/local/lib/python3.6/site-packages (from Flask==1.0.2->-r /var/www/requirements.txt (line 1
) (1.1.0)
Requirement already satisfied: Werkzeug>=0.14 in /usr/local/lib/python3.6/site-packages (from Flask==1.0.2->-r /var/www/requirements.txt (line 1)) (
.0.1)
Requirement already satisfied: Jinja2>=2.10 in /usr/local/lib/python3.6/site-packages (from Flask==1.0.2->-r /var/www/requirements.txt (line 1)) (2.
1.2)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.6/site-packages (from Jinja2>=2.10->Flask==1.0.2->-r /var/www/requirements
txt (line 1)) (1.1.1)
Installing collected packages: Flask
  Found existing installation: Flask 1.1.2
    Uninstalling Flask-1.1.2:
      Successfully uninstalled Flask-1.1.2
Successfully installed Flask-1.0.2
You are using pip version 19.0.1, however version 21.3.1 is available.
```

```
Successfully installed Flask-1.0.2
You are using pip version 19.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Removing intermediate container ad23fd567b37
 ---> be567968b051
Successfully built be567968b051
Successfully tagged docker.test:latest
20d258523ca16e5cb557e42ea0d51984717890b7d4ed32dc9813f16448f8946a
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS          PORTS                              NAMES
20d258523ca1   docker.test   "/entrypoint.sh /sta…"   49 seconds ago   Up 48 seconds   443/tcp, 0.0.0.0:56733->80/tcp   docker.test
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ls -R
.:
Dockerfile        __pycache__       app               main.py           requirements.txt  start.sh          supervisord.pid   uwsgi.ini

./__pycache__:
main.cpython-36.pyc

./app:
__init__.py  __pycache__  static      templates    views.py

./app/__pycache__:
__init__.cpython-36.pyc  views.cpython-36.pyc

./app/static:

./app/templates:
```

## 3.Create an IBM container registry and deploy helloworld app or jobportalapp.

```
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud login
API endpoint: https://cloud.ibm.com

Email> 2019103033@student.annauniv.edu

Password>
Authenticating...
OK

Targeted account ManojKumar Manivannan's Account (0ede53fab64747acaa49ebd6eefecd5a)


Select a region (or press enter to skip):
1. au-syd
2. in-che
3. jp-osa
4. jp-tok
5. kr-seo
6. eu-de
7. eu-gb
8. ca-tor
9. us-south
10. us-east
11. br-sao
Enter a number> 1
Targeted region au-syd


API endpoint:      https://cloud.ibm.com
Region:            au-syd
User:              2019103033@student.annauniv.edu
Account:           ManojKumar Manivannan's Account (0ede53fab64747acaa49ebd6eefecd5a)
Resource group:    No resource group targeted, use 'ibmcloud target -g RESOURCE_GROUP'
CF API endpoint:
Org:
Space:
```

```
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud plugin install container-registry -r 'IBM Cloud'
Looking up 'container-registry' from repository 'IBM Cloud'...
Plug-in 'container-registry[cr] 1.0.2' found in repository 'IBM Cloud'
Attempting to download the binary file...
 11.69 MiB / 11.69 MiB [=============================================================================================================] 100.00% 0s
12255232 bytes downloaded
Installing binary...
OK
Plug-in 'container-registry 1.0.2' was successfully installed into /root/.bluemix/plugins/container-registry. Use 'ibmcloud plugin show container-reg
istry' to show its details.
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud cr region-set global
The region is set to 'global', the registry is 'icr.io'.

OK
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud cr namespace add 2019103033
FAILED
'namespace' is not a registered command. Check your list of installed plug-ins. See 'ibmcloud cr help'.

[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud cr namespace-add 2019103033
No resource group is targeted. Therefore, the default resource group for the account ('Default') is targeted.

Adding namespace '2019103033' in resource group 'Default' for account ManojKumar Manivannan's Account in registry icr.io...

Successfully added namespace '2019103033'

OK
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud cr login
Logging 'docker' in to 'icr.io'...
Logged in to 'icr.io'.

OK
```

```
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker tag hello-world icr.io/2019103033/test:latest
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker push icr.io/2019103033/test:latest
The push refers to repository [icr.io/2019103033/test]
e07ee1baac5f: Pushed
latest: digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4 size: 525
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ ibmcloud cr image-list
Listing images...

Repository               Tag      Digest         Namespace    Created      Size     Security status
icr.io/2019103033/test   latest   f54a58bc1aac   2019103033   1 year ago   2.5 kB   -

OK
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker pull ic
icr.io/2019103033/test         icr.io/2019103033/test:latest
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker pull icr.io/2019103033/test
Using default tag: latest
latest: Pulling from 2019103033/test
Digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4
Status: Image is up to date for icr.io/2019103033/test:latest
icr.io/2019103033/test:latest
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker images
REPOSITORY                  TAG                 IMAGE ID       CREATED          SIZE
docker.test                 latest              be567968b051   26 minutes ago   198MB
<none>                      <none>              262949140 97d  46 minutes ago   198MB
icr.io/2019103033/test      latest              feb5d9fea6a5   13 months ago    13.3kB
hello-world                 latest              feb5d9fea6a5   13 months ago    13.3kB
tiangolo/uwsgi-nginx-flask  python3.6-alpine3.7  cdec3b0d8f20   2 years ago      189MB
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker run
docker.test                                hello-world:latest                          tiangolo/uwsgi-nginx-flask
docker.test:latest                         icr.io/2019103033/test                      tiangolo/uwsgi-nginx-flask:python3.6-alpine3.7
```

```
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker run icr
icr.io/2019103033/test          icr.io/2019103033/test:latest
[node1] (local) root@192.168.0.18 /var/www/TestApp
$ docker run icr.io/2019103033/test

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

## 4.Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport

```
[node1] (local) root@192.168.0.13 ~
$ ibmcloud ks clusters
OK
Name            ID                     State    Created          Workers   Location   Version       Resource Group Name   Provider
mycluster-free  cdmqakpf08kf4pv50g80   normal   47 minutes ago   1         mil01      1.24.7_1542   Default               classic
[node1] (local) root@192.168.0.13 ~
$ export CLUSTER_NAME=mycluster-free
[node1] (local) root@192.168.0.13 ~
$ ibmcloud ks cluster config --cluster $CLUSTER_NAME
OK
The configuration for mycluster-free was downloaded successfully.

Added context for mycluster-free to the current kubeconfig file.
You can now execute 'kubectl' commands against your cluster. For example, run 'kubectl get nodes'.
If you are accessing the cluster for the first time, 'kubectl' commands might fail for a few seconds while RBAC synchronizes.
[node1] (local) root@192.168.0.13 ~
$ kubectl get namespace
NAME              STATUS   AGE
default           Active   41m
ibm-cert-store    Active   30m
ibm-operators     Active   38m
ibm-system        Active   41m
kube-node-lease   Active   41m
kube-public       Active   41m
kube-system       Active   41m
```

```
[node1] (local) root@192.168.0.13 ~
$ git clone https://github.com/IBM/guestbook.git
Cloning into 'guestbook'...
remote: Enumerating objects: 448, done.
remote: Total 448 (delta 0), reused 0 (delta 0), pack-reused 448
Receiving objects: 100% (448/448), 205.42 KiB | 11.41 MiB/s, done.
Resolving deltas: 100% (264/264), done.
[node1] (local) root@192.168.0.13 ~
$ git clone https://github.com/IBM/kube101.git
Cloning into 'kube101'...
remote: Enumerating objects: 678, done.
remote: Counting objects: 100% (188/188), done.
remote: Compressing objects: 100% (135/135), done.
remote: Total 678 (delta 58), reused 125 (delta 27), pack-reused 490
Receiving objects: 100% (678/678), 2.83 MiB | 23.17 MiB/s, done.
Resolving deltas: 100% (298/298), done.
[node1] (local) root@192.168.0.13 ~
$ kubectl create deployment guestbook --image=ibmcom/guestbook:v1
deployment.apps/guestbook created
[node1] (local) root@192.168.0.13 ~
$ kubectl get pods
NAME                         READY   STATUS    RESTARTS   AGE
guestbook-7c568567b4-9rhjh   1/1     Running   0          37s
```

```
[node1] (local) root@192.168.0.13 ~
$ ibmcloud ks clusters
OK
Name              ID                     State    Created          Workers   Location
  Version         Resource Group Name    Provider
mycluster-free    cdmqakpf08kf4pv50g80   normal   47 minutes ago   1         mil01
  1.24.7_1542     Default                classic
[node1] (local) root@192.168.0.13 ~
$ export CLUSTER_NAME=mycluster-free
[node1] (local) root@192.168.0.13 ~
$ ibmcloud ks cluster config --cluster $CLUSTER_NAME
OK
The configuration for mycluster-free was downloaded successfully.

Added context for mycluster-free to the current kubeconfig file.
You can now execute 'kubectl' commands against your cluster. For example, run 'kubect
l get nodes'.
If you are accessing the cluster for the first time, 'kubectl' commands might fail fo
r a few seconds while RBAC synchronizes.
[node1] (local) root@192.168.0.13 ~
$ kubectl get namespace
NAME              STATUS   AGE
[node1] (local) root@192.168.0.13 ~
$ kubectl expose deployment guestbook --type="NodePort" --port=3000
service/guestbook exposed
[node1] (local) root@192.168.0.13 ~
$ kubectl get service guestbook
NAME        TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
guestbook   NodePort   172.21.173.140   <none>        3000:32585/TCP   11s
[node1] (local) root@192.168.0.13 ~
$ kubectl get nodes -o wide
NAME             STATUS   ROLES    AGE   VERSION       INTERNAL-IP      EXTERNAL-IP
  OS-IMAGE             KERNEL-VERSION       CONTAINER-RUNTIME
10.144.180.124   Ready    <none>   40m   v1.24.7+IKS   10.144.180.124   169.51.194.2
  Ubuntu 18.04.6 LTS   4.15.0-194-generic   containerd://1.6.8
```

```
[node1] (local) root@192.168.0.13 ~
$ kubectl expose deployment guestbook --type="NodePort" --port=3000
service/guestbook exposed
[node1] (local) root@192.168.0.13 ~
$ kubectl get service guestbook
NAME        TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
guestbook   NodePort   172.21.173.140   <none>        3000:32585/TCP   11s
[node1] (local) root@192.168.0.13 ~
$ kubectl get nodes -o wide
NAME             STATUS   ROLES    AGE   VERSION       INTERNAL-IP      EXTERNAL-IP    OS-IMAGE             KERNEL-VERSION       CONTAINER-RUNTIME
10.144.180.124   Ready    <none>   40m   v1.24.7+IKS   10.144.180.124   169.51.194.2   Ubuntu 18.04.6 LTS   4.15.0-194-generic   containerd://1.6.8
[node1] (local) root@192.168.0.13 ~
$
```