Assignments
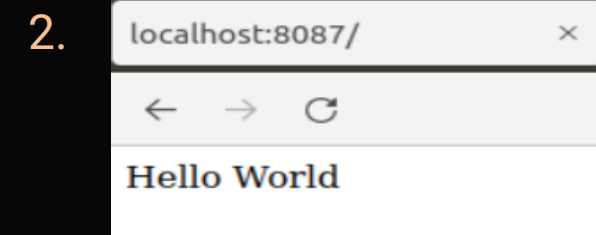
Navdeep Kaur

Docker Lab1 Solution

1. docker run --name python-c –d –p 8087:5000   navjoy220161/python_flask:1.0.0

2.
```
localhost:8087/          ×
←   →   C
Hello World
```

3.
```
training@ip-172-31-37-38:~/docker_fundamentals/python-flask-demo$ docker ps
CONTAINER ID    IMAGE                           COMMAND              CREATED
NAMES
3a63c0b19c9e    navjoy220161/python_flask:1.0.0  "python main.py"      4 seconds ago
```

   docker exec –it 3a63c0b19c9e sh

```
# ls
Dockerfile  main.py  pod-defination.yaml  requirements.txt
# rm main.py
```

4. docker stop 3a63c0b19c9e
   docker start 3a63c0b19c9e

5.
```
localhost:8087/          ×
Unable to connect
```
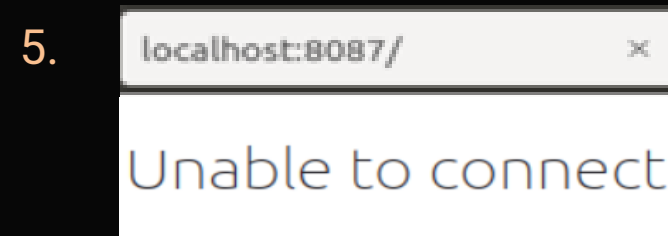
6. docker logs 3a63c0b19c9e
   docker ps -a

7. docker rm –f 3a63c0b19c9e

8. docker run --name python-c –d –p 8087:5000   navjoy220161/python_flask:1.0.0

9. docker cp main.py <container_id>:/python-flask/

Docker Lab2 Solution

cd docker_fundamentals/python-flask-demo
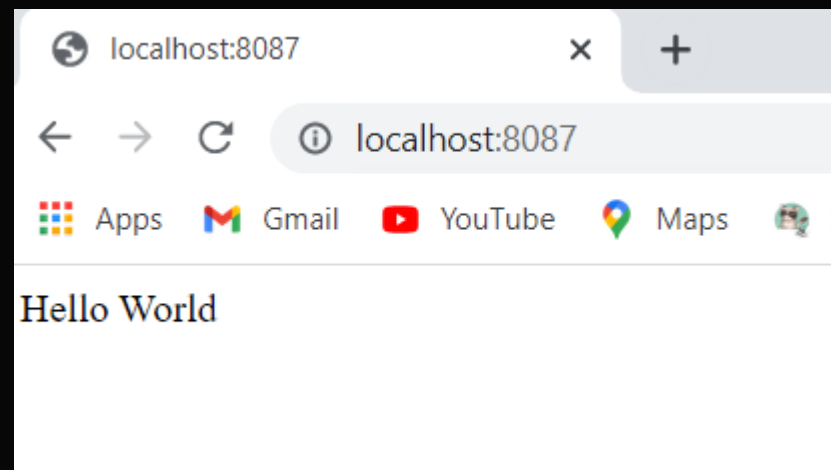
docker build –t navjoy220161/python-flask:1.0.0 .

docker login

docker push navjoy220161/python-flask:1.0.0

docker rmi navjoy220161/python-flask:1.0.0

docker run –p 8087:5000 –d --name python-flask navjoy220161/python-flask:1.0.0

Pods Solution

1. Create a new pod with the nginx image and name nginx-pod without using a yaml file.

```
kubectl run nginx-pod --image nginx
```

2. Check solution in kubernetes_solutions\pod\2.yaml

4. There is some issue in the above Pod. Fix the issue in above pod and apply the changes.

```
kubectl get pod

NAME          READY    STATUS             RESTARTS    AGE
httpd-pod     0/1      ImagePullBackOff   0           33s
```

```
kubectl get pod httpd-pod -o yaml > httpd.yaml
```

```
kubectl delete pod httpd-pod
```

Edit httpd.yaml

```
spec:
  containers:
  - image: httpd
    imagePullPolicy: Always
    name: httpd-pod
```

```
kubectl create -f .\httpd.yaml
```

Pods Solution

5. Get the logs of container mongodb-container

```
kubectl logs myweb-app -c mongodb-container
```

Go inside container httpd-container and run the command whoami

```
kubectl exec -it myweb-app -c httpd-container -- sh
```

ReplicaSet Solutions

1. Check solution in kubernetes_solutions\replicaset\1.yaml

2. Delete one of the pod created by replicaset and check the pod status

```
kubectl delete pod nginx-rs-5wmsd
```

```
kubectl get pod
NAME              READY   STATUS     RESTARTS   AGE
nginx-rs-4w958    1/1     Running    0          21s
nginx-rs-tnwpm    1/1     Running    0          105s
nginx-rs-xthg7    1/1     Running    0          105s
```

3. There is an existing replicaset named web-rs running in your machine. First check how many replicas it is running right now and scale it up to run 6 replicas

```
kubectl scale rs myweb-rs --replicas 6
```

4. There is an existing replicaset named loadbalancer-rs running in your machine. Pod started by this replicas are in error state, fix the issue and make the pod in running state.

```
kubectl edit rs nginx-rs
```

```
kubectl delete pod nginx-rs-gj6hj nginx-rs-2bbsl nginx-rs-cvlql
```

Deployment Solutions

1. Check solution in kubernetes_solutions\deployment\1.yaml

2. Edit the deployment and change its update Strategy to "Recreate"

```
> kubectl edit deployment nginx-deployment
```

```
strategy:
    type: Recreate
template:
    metadata:
        creationTimestamp: null
```

3. In above deployment , set the image to nginx-junk and check the status

```
kubectl set image deployment nginx-deployment   nginx-container=nginx-junk --record

> kubectl rollout status   deployment nginx-deployment

> kubectl get pod
```

```
NAME                              READY   STATUS          RESTARTS   AGE
nginx-deployment-68494758b4-2rwqt  0/1     ErrImagePull        0        116s
nginx-deployment-68494758b4-6dlbg  0/1     ImagePullBackOff    0        116s
nginx-deployment-68494758b4-s86fn  0/1     ImagePullBackOff    0        116s
```
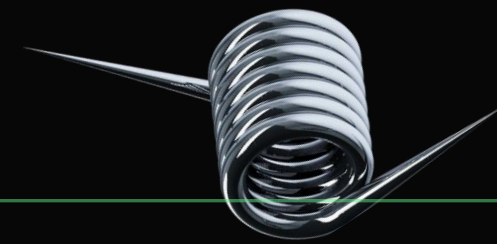
4. Now check the history of all rollouts

```
kubectl rollout history deployment nginx-deployment
```

5. Now rollback to previous version and check status

```
kubectl rollout undo   deployment nginx-deployment
```
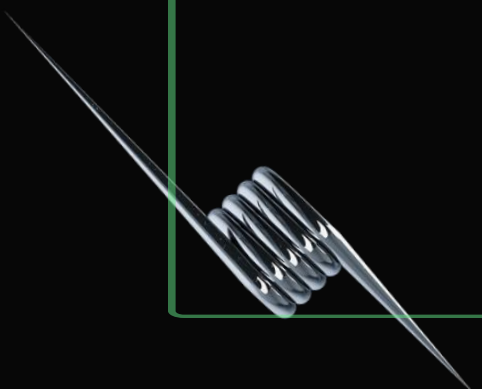
ConfigMap Solutions

1. Create a configmap for a gaming app using command with below properties
   name: gameconfigmap -> enemies=aliens, lives=3

```
kubectl create configmap gameconfigmap --from-literal=enemies=aliens --from-literal=lives=3
```

2. Check solution in kubernetes_solutions\configmap\2.yaml

3. Check solution in kubernetes_solutions\configmap\3.yaml

Secret Solution

1. Create a secret for the same gaming app using command with below properties. name: dbsecret => db_host=mongoDB , db_user=root,db_pass=mongo123

```
kubectl create secret generic dbsecret --from-literal=db_host=mongoDB --from-literal=db_user=root
--from-literal=db_pass=mongo123
```

2. Check the solution in kubernetes_solutions\secret\2.yaml

3. Check the solution in kubernetes_solutions\secret\3.yaml

4. Check the solution in kubernetes_solutions\secret\4.yaml