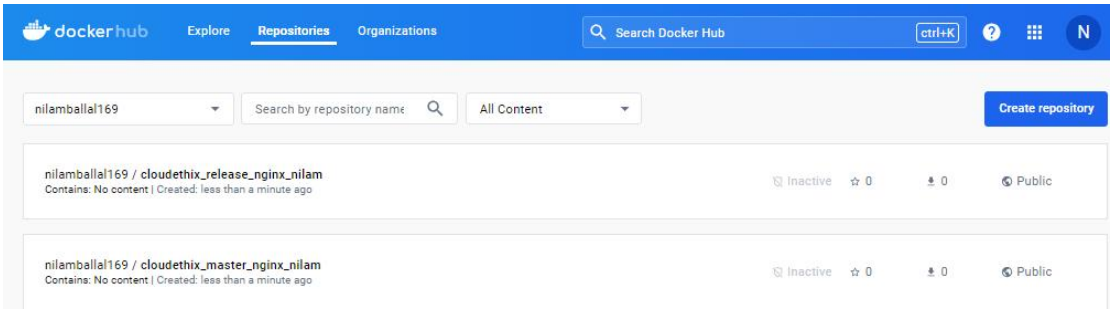
CLOUDETHIX

Que 1 →

● Create 2 Public Docker Hub registries named cloudethix_master_nginx_yourname & cloudethix_release_nginx_yourname.



● Clone below repository on your system.

https://github.com/zembutsu/docker-sample-nginx.git

```
root@Nilam:k8sCluster_kubeadm_terraform# git clone https://github.com/zembutsu/docker-sample-nginx.git
Cloning into 'docker-sample-nginx'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 22 (delta 7), reused 6 (delta 6), pack-reused 10
Receiving objects: 100% (22/22), done.
Resolving deltas: 100% (7/7), done.
```

```
root@Nilam:k8sCluster_kubeadm_terraform# cd docker-sample-nginx/
root@Nilam:docker-sample-nginx# ll
total 4
drwxrwxrwx 1 nilam nilam  512 Feb 21 11:53 ./
drwxrwxrwx 1 nilam nilam  512 Feb 21 11:53 ../
drwxrwxrwx 1 nilam nilam  512 Feb 21 11:53 .git/
-rwxrwxrwx 1 nilam nilam   95 Feb 21 11:53 Dockerfile*
-rwxrwxrwx 1 nilam nilam 1084 Feb 21 11:53 LICENSE*
-rwxrwxrwx 1 nilam nilam   73 Feb 21 11:53 README.md*
-rwxrwxrwx 1 nilam nilam  286 Feb 21 11:53 default.conf*
-rwxrwxrwx 1 nilam nilam  103 Feb 21 11:53 index.html*
```

●

Initialize a local repository & copy the code from above repo to your local repository in master branch and then create below branches.

release

main

hotfix

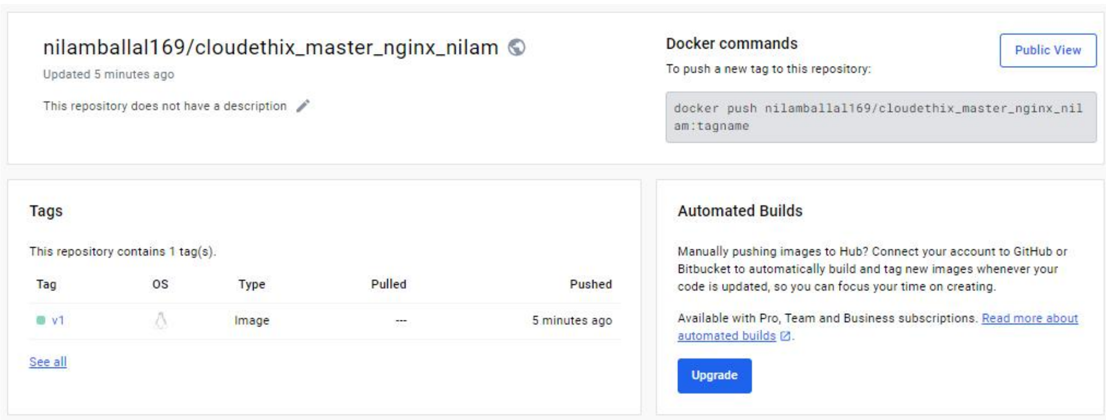
```
root@Nilam:docker-sample-nginx# git branch
* master
root@Nilam:docker-sample-nginx# git branch release
root@Nilam:docker-sample-nginx# git branch main
root@Nilam:docker-sample-nginx# git branch hotfix
root@Nilam:docker-sample-nginx# git branch
  hotfix
  main
* master
  release
root@Nilam:docker-sample-nginx#
```

● Once code is copied to local repository,from master branch update the index.html and add word "Cloudethix Master Branch Nginx" and build the docker image & add meaningful tags and push to Docker Hub registry cloudethix_master_nginx_yourname.

```
root@Nilam:docker-sample-nginx# cat index.html
 "Cloudethix Master Branch Nginx"
root@Nilam:docker-sample-nginx#
```

```
root@Nilam:docker-sample-nginx# docker image build -t nilamballal169/cloudethix_master_nginx_nilam:v1 .
[+] Building 30.1s (9/9) FINISHED                                    docker:default
root@Nilam:docker-sample-nginx#
```

```
root@Nilam:docker-sample-nginx# docker image push  nilamballal169/cloudethix_master_nginx_nilam:v1
The push refers to repository [docker.io/nilamballal169/cloudethix_master_nginx_nilam]
28c4e6a5149b: Pushed
2abf08b521cd: Pushed
667a247707f0: Mounted from library/nginx
d8527026595f: Mounted from library/nginx
2593b08e5428: Mounted from library/nginx
9909978d630d: Mounted from library/nginx
c5140fc719dd: Mounted from library/nginx
3137f8f0c641: Mounted from nilamballal169/k8s_project07
718db50a47c0: Mounted from library/nginx
aedc3bda2944: Mounted from library/nginx
v1: digest: sha256:7645a749e1b0d91bf905149c4f100a17c3944d2734839fdd57bc94df2ae95b26 size: 2403
root@Nilam:docker-sample-nginx#
```



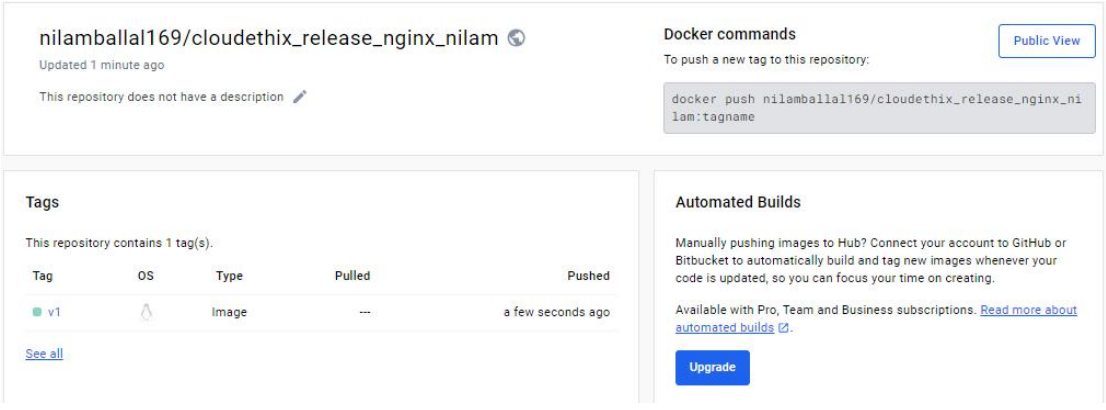
● Also from release branch update the index.html and add word "Cloudethix Release Branch Nginx" and build the docker image & add meaningful tags and push to Docker Hub registry cloudethix_release_nginx_yourname.

```
root@Nilam:docker-sample-nginx# git checkout release
M       index.html
Switched to branch 'release'
root@Nilam:docker-sample-nginx# git branch
  hotfix
  main
  master
* release
```

```
root@Nilam:docker-sample-nginx# vim index.html
root@Nilam:docker-sample-nginx# cat index.html
"Cloudethix Release Branch Nginx"
```

```
root@Nilam:docker-sample-nginx# docker image build -t nilamballal169/cloudethix_release_nginx_nilam:v1 .
[+] Building 12.9s (9/9) FINISHED                                                                    docker:default
```

```
root@Nilam:docker-sample-nginx# docker image push nilamballal169/cloudethix_release_nginx_nilam:v1
The push refers to repository [docker.io/nilamballal169/cloudethix_release_nginx_nilam]
3d5150f886cc: Pushed
2abf08b521cd: Mounted from nilamballal169/cloudethix_master_nginx_nilam
667a247707f0: Mounted from nilamballal169/cloudethix_master_nginx_nilam
d8527026595f: Mounted from nilamballal169/cloudethix_master_nginx_nilam
2593b08e5428: Mounted from nilamballal169/cloudethix_master_nginx_nilam
9909978d630d: Mounted from nilamballal169/cloudethix_master_nginx_nilam
c5140fc719dd: Mounted from nilamballal169/cloudethix_master_nginx_nilam
3137f8f0c641: Mounted from nilamballal169/cloudethix_master_nginx_nilam
718db50a47c0: Mounted from nilamballal169/cloudethix_master_nginx_nilam
aedc3bda2944: Mounted from nilamballal169/cloudethix_master_nginx_nilam
v1: digest: sha256:a6e6ce861ca80795a04407fbc43e74ab0aab7c8036eb22c2ce0c458134a96a4f size: 2403
root@Nilam:docker-sample-nginx#
```



● Once Images are copied to Docker hub registries, switch to the main branch.

```
root@Nilam:docker-sample-nginx#  git checkout main
M       index.html
Switched to branch 'main'
root@Nilam:docker-sample-nginx# git branch
  hotfix
* main
  master
  release
root@Nilam:docker-sample-nginx#
```

● In main branch create directory named kube/clusterIP & inside kube directory create file named master_pod.yaml with pod name master_nginx & with label master_nginx & add image that you have pushed in Docker Hub registry cloudethix_master_nginx_yourname.

```
root@Nilam:docker-sample-nginx# mkdir -p kube/clusterIP
root@Nilam:docker-sample-nginx# ll
total 4
drwxrwxrwx 1 nilam nilam  512 Feb 21 12:58 ./
drwxrwxrwx 1 nilam nilam  512 Feb 21 11:53 ../
drwxrwxrwx 1 nilam nilam  512 Feb 21 12:56 .git/
-rwxrwxrwx 1 nilam nilam   95 Feb 21 11:53 Dockerfile*
-rwxrwxrwx 1 nilam nilam 1084 Feb 21 11:53 LICENSE*
-rwxrwxrwx 1 nilam nilam   73 Feb 21 11:53 README.md*
-rwxrwxrwx 1 nilam nilam  286 Feb 21 11:53 default.conf*
-rwxrwxrwx 1 nilam nilam   34 Feb 21 12:51 index.html*
drwxrwxrwx 1 nilam nilam  512 Feb 21 12:58 kube/
```

```
root@Nilam:kube# touch master_pod.yaml
root@Nilam:kube# cat master_pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: master-nginx
  labels:
    name: master-nginx
spec:
  containers:
  - name: master-nginx
    image: nilamballal169/cloudethix_master_nginx_nilam:v1
    ports:
      - containerPort: 80
root@Nilam:kube#
```

● Also create a file release_pod.yaml with pod name release_nginx & with label release_nginx & add image that you have pushed in Docker Hub registry cloudethix_release_nginx_yourname.

```
root@Nilam:kube# cat release_pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: release_nginx
  labels:
    name: release_nginx
spec:
  containers:
  - name: release_nginx
    image: nilamballal169/cloudethix_release_nginx_nilam:v1
    ports:
      - containerPort: 80
root@Nilam:kube#
```

● Create a file called cluster_ip-service.yaml with service name cloudethix_clusterip and with Type clusterIP.
● Then, select the pod with label release_nginx in service.● Create all these three resources in your k8s cluster.

```
root@Nilam:kube# cat cluster_ip-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: cloudethix_clusterip
spec:
  selector:
    app: release_nginx
  ports:
  - port: 80
    targetPort: 80
    type: ClusterIP
root@Nilam:kube#
```

● Now, access master_nginx pod shell & curl the master_nginx pod & check the result.

```
root@Nilam:kube# kubectl exec -it master-nginx -- /bin/sh
/ # curl localhost
 "Cloudethix Master Branch Nginx"
/ #
```

● Also try to curl release_nginx pod with DNS name & check the result.

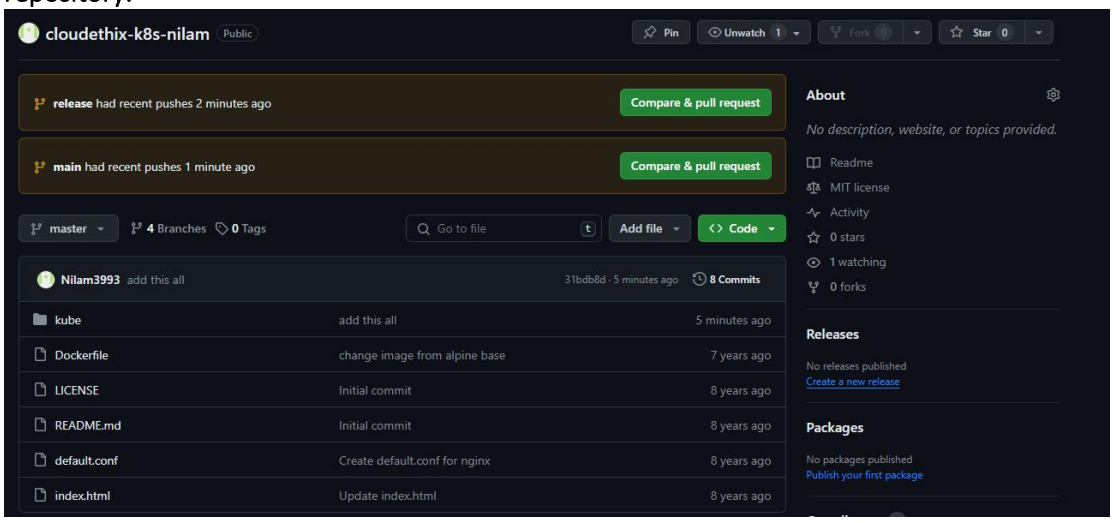
```
root@Nilam:kube# kubectl exec -it release-nginx -- /bin/sh
/ # curl localhost
"Cloudethix Release Branch Nginx"
/ #
```

● Then curl the clusterip service with its name and check the result.

```
root@Nilam:kube# kubectl exec -it release-nginx -- /bin/sh
/ # curl localhost
"Cloudethix Release Branch Nginx"
/ # curl cloudethix-clusterip
"Cloudethix Release Branch Nginx"
/ # exit
root@Nilam:kube#
```

```
root@Nilam:kube# kubectl exec -it master-nginx -- /bin/sh
/ # curl localhost
 "Cloudethix Master Branch Nginx"
/ # curl cloudethix-clusterip
"Cloudethix Release Branch Nginx"
/ # exit
root@Nilam:kube#
```

● Finally, create a GITHUB remote repository named cloudethix-k8s-yourname and push all the branches to the remote repository.



Que 2 →

● In the main branch of your local repository create a directory kube/NodePort.

● Create below files from below url. Please make sure you will create NodePort service with port 30008 instead of loadbalancer.
https://kubernetes.io/docs/tasks/access-application-cluster/connecting-frontend-backend/.

backend-deployment.yaml
backend-service.yaml
frontend-deployment.yaml
frontend-NodePort-service.yaml

● Once files are created , create all the resources in your k8s cluster.

● Access all public ips with port 30008 in the browser and then check the result.

● Finally, push all the latest code to the remote repository.

```
root@Nilam:NodePort# pwd
/mnt/c/Users/manis/Music/Project07/k8sCluster_kubeadm_terraform/docker-sample-nginx/kube/NodePort
root@Nilam:NodePort# ll
total 0
drwxrwxrwx 1 nilam nilam 512 Feb 21 17:05 ./
drwxrwxrwx 1 nilam nilam 512 Feb 21 16:53 ../
```

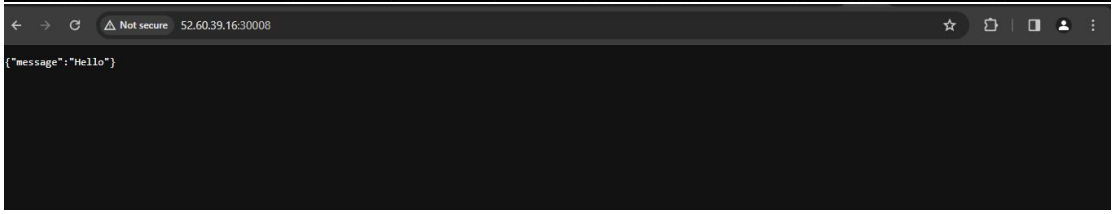
```
root@Nilam:NodePort# cat backend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
spec:
  selector:
    matchLabels:
      app: hello
      tier: backend
      track: stable
  replicas: 3
  template:
    metadata:
      labels:
        app: hello
        tier: backend
        track: stable
    spec:
      containers:
        - name: hello
          image: "gcr.io/google-samples/hello-go-gke:1.0"
          ports:
            - name: http
              containerPort: 80root@Nilam:NodePort#
```

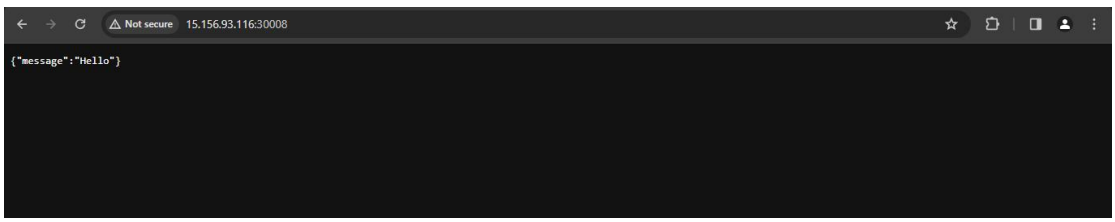
```
root@Nilam:NodePort# cat backend-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: hello
spec:
  selector:
    app: hello
    tier: backend
  ports:
  - protocol: TCP
    port: 80
    targetPort: httproot@Nilam:NodePort#
```

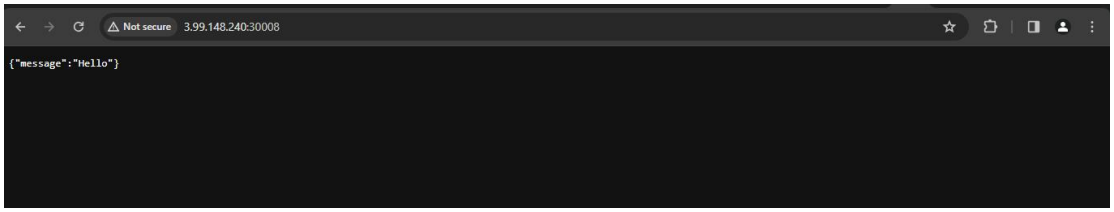
```
...root@Nilam:NodePort# cat frontend-NodePort-service.yaml
---
apiVersion: v1
kind: Service
metadata:
  name: frontend
spec:
  selector:
    app: hello
    tier: frontend
  ports:
  - protocol: "TCP"
    port: 80
    targetPort: 80
    nodePort: 30008
  type: NodePort
root@Nilam:NodePort#
```

```
root@Nilam:NodePort# cat frontend-deployment.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend
spec:
  selector:
    matchLabels:
      app: hello
      tier: frontend
      track: stable
  replicas: 1
  template:
    metadata:
      labels:
        app: hello
        tier: frontend
        track: stable
    spec:
      containers:
        - name: nginx
          image: "gcr.io/google-samples/hello-frontend:1.0"
          lifecycle:
            preStop:
              exec:
                command: ["/usr/sbin/nginx","-s","quit"]
...root@Nilam:NodePort#
```

```
root@Nilam:NodePort# kubectl apply -f .
deployment.apps/backend created
service/hello created
service/frontend created
deployment.apps/frontend created
root@Nilam:NodePort# kgp -o wide
NAME                        READY   STATUS    RESTARTS   AGE    IP              NODE       NOMINATED NODE   READINESS GATES
backend-7f5b7998b9-kbcgm    1/1     Running   0          6s     10.108.43.27    worker-0   <none>           <none>
backend-7f5b7998b9-pr4cc    1/1     Running   0          6s     10.108.43.26    worker-0   <none>           <none>
backend-7f5b7998b9-rc4ms    1/1     Running   0          6s     10.111.158.78   worker-1   <none>           <none>
frontend-85c84f8b8b-gf6pc   1/1     Running   0          4s     10.111.158.79   worker-1   <none>           <none>
master-nginx                1/1     Running   0          3h28m  10.108.43.19    worker-0   <none>           <none>
```







Que 3 →

●

Create any 2 pods and assign them to different worker nodes with nodeName property

```
root@Nilam:Ques03# cat pod01.yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-01
  labels:
    name: myapp-01
spec:
  containers:
  - name: myapp-01
    image: nilamballal169/cloudethix_master_nginx_nilam:v1
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

```
root@Nilam:Ques03# cat pod02.yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-02
  labels:
    name: myapp-02
spec:
  containers:
  - name: myapp-02
    image: nilamballal169/cloudethix_release_nginx_nilam:v1
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80root@Nilam:Ques03#
```

```
root@Nilam:Ques03# kubectl apply -f .
pod/myapp-01 created
pod/myapp-02 created
root@Nilam:Ques03# kgp
NAME                          READY   STATUS            RESTARTS   AGE
backend-7f5b7998b9-kbcgm      1/1     Running           0          9m50s
backend-7f5b7998b9-pr4cc      1/1     Running           0          9m50s
backend-7f5b7998b9-rc4ms      1/1     Running           0          9m50s
frontend-85c84f8b8b-gf6pc     1/1     Running           0          9m48s
master-nginx                  1/1     Running           0          3h37m
myapp-01                      1/1     Running           0          3s
myapp-02                      1/1     Running           0          3s
```

Que 4 →

● Label both worker nodes such as worker-0 node as cloudethix-k8s-00 & worker-1 node as cloudethix-k8s-01.

● Once nodes are labeled, create pod00.yaml file and schedule the pod on worker-0 node with nodeSelector property. Also create one more file named pod01.yaml & schedule the pod on worker-1 node.

```
root@Nilam:Ques04# cat pod01.yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod00
  labels:
    name: pod00
spec:
  containers:
  - name: container00
    image: nilamballal169/cloudethix_master_nginx_nilam:v1
  nodeSelector:
    worker: cloudethix-k8s-00


root@Nilam:Ques04#
```

```
root@Nilam:Ques04# cat pod02.yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod01
  labels:
    name: pod01
spec:
  containers:
  - name: container01
    image: nilamballal169/cloudethix_release_nginx_nilam:v1
  nodeSelector:
    worker: cloudethix-k8s-01


root@Nilam:Ques04#
```

Que 5 →

-

Clone the below repo locally & create DaemonSet from directory DaemonSet101.

https://github.com/collabnix/kubelabs

```
root@Nilam:Ques05# git clone https://github.com/collabnix/kubelabs
Cloning into 'kubelabs'...
remote: Enumerating objects: 12313, done.
remote: Counting objects: 100% (1160/1160), done.
remote: Compressing objects: 100% (483/483), done.
remote: Total 12313 (delta 696), reused 1049 (delta 642), pack-reused 11153
Receiving objects: 100% (12313/12313), 61.41 MiB | 7.36 MiB/s, done.
Resolving deltas: 100% (3277/3277), done.
Updating files: 100% (7329/7329), done.
root@Nilam:Ques05# ll
total 0
drwxrwxrwx 1 nilam nilam 512 Feb 21 19:42 ./
drwxrwxrwx 1 nilam nilam 512 Feb 21 19:42 ../
drwxrwxrwx 1 nilam nilam 512 Feb 21 19:45 kubelabs/
root@Nilam:Ques05# cd kubelabs/
```

```
root@Nilam:kubelabs# cd DaemonSet101
root@Nilam:DaemonSet101# ll
total 8
drwxrwxrwx 1 nilam nilam  512 Feb 21 19:45 ./
drwxrwxrwx 1 nilam nilam  512 Feb 21 19:45 ../
-rwxrwxrwx 1 nilam nilam 7040 Feb 21 19:45 README.md*
-rwxrwxrwx 1 nilam nilam  394 Feb 21 19:45 daemonset.yml*
root@Nilam:DaemonSet101# cat daemonset.yml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: prometheus-daemonset
spec:
  selector:
    matchLabels:
      tier: monitoring
      name: prometheus-exporter
  template:
    metadata:
      labels:
        tier: monitoring
        name: prometheus-exporter
    spec:
      containers:
      - name: prometheus
        image: prom/node-exporter
        ports:
        - containerPort: 80
```

```
pod00                              0/1     Pending             0          28m
pod01                              0/1     Pending             0          28m
prometheus-daemonset-7xsxq         1/1     Running             0          7s
prometheus-daemonset-zd558         1/1     Running             0          7s
```

Que 6 →

● Create a static pod with name cloudethix-static in your k8s cluster. Refer below link.

https://kubernetes.io/docs/tasks/configure-pod-container/static-pod/

```
root@Nilam:Ques06# cat static-web.yaml
apiVersion: v1
kind: Pod
metadata:
  name: cloudethix-static
  labels:
    role: myrole
spec:
  containers:
    - name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
          protocol: TCProot@Nilam:Ques06#
```

```
root@Nilam:Ques06# kubectl create -f .
pod/cloudethix-static created
```

```
root@Nilam:Ques06# kgp
NAME                                READY   STATUS            RESTARTS   AGE
backend-7f5b7998b9-kbcgm            1/1     Running           0          72m
backend-7f5b7998b9-pr4cc            1/1     Running           0          72m
backend-7f5b7998b9-rc4ms            1/1     Running           0          72m
cloudethix-static                   1/1     Running           0          12s
```

Que 7 →

- **Install Kubectx & kubens in your k8s cluster.**

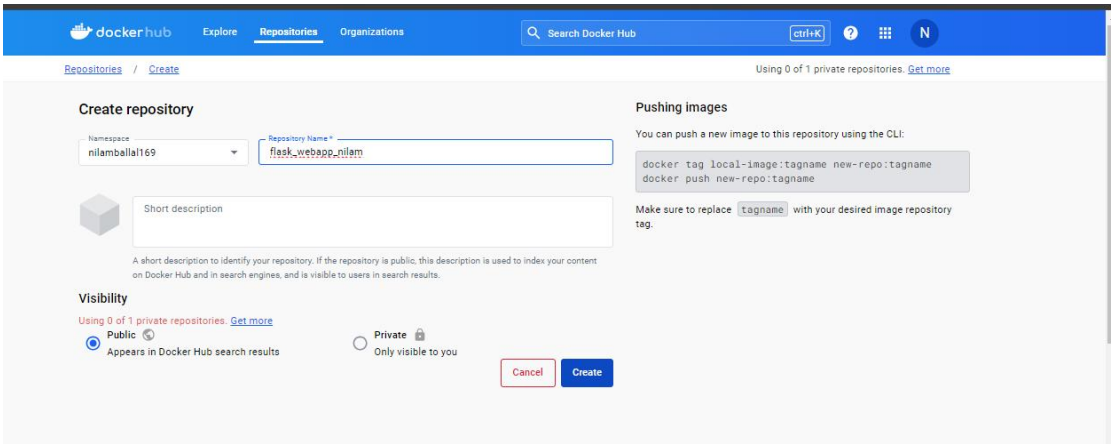
```
root@Nilam:~# git clone https://github.com/ahmetb/kubectx.git ~/.kubectx
Cloning into '/root/.kubectx'...
remote: Enumerating objects: 1502, done.
remote: Counting objects: 100% (452/452), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 1502 (delta 390), reused 355 (delta 353), pack-reused 1050
Receiving objects: 100% (1502/1502), 912.88 KiB | 378.00 KiB/s, done.
Resolving deltas: 100% (876/876), done.
```

```
root@Nilam:~# echo 'export PATH=$PATH:~/.kubectx' >> ~/.bashrc
root@Nilam:~# echo 'source ~/.kubectx/completion/kubens.bash' >> ~/.bashrc
root@Nilam:~# echo 'source ~/.kubectx/completion/kubectx.bash' >> ~/.bashrc
root@Nilam:~# source ~/.bashrc
root@Nilam:~#
```

```
root@Nilam:simpleapp# kubectl create namespace my-namespace
namespace/my-namespace created
root@Nilam:simpleapp#
root@Nilam:simpleapp# kubens
default
kube-node-lease
kube-public
kube-system
my-namespace
project01
root@Nilam:simpleapp# kubens my-namespace
Context "kubernetes-admin@kubernetes" modified.
Active namespace is "my-namespace".
root@Nilam:simpleapp# kubens
default
kube-node-lease
kube-public
kube-system
my-namespace
project01
root@Nilam:simpleapp#
```

Que 8 →

● Create 1 Public Docker Hub registry named flask_webapp_yourname.



● Clone below repository on your system.
https://github.com/mmumshad/simple-webapp-docker.git

```
root@Nilam:k8sCluster_kubeadm_terraform# git clone https://github.com/mmumshad/simple-webapp-docker.git
Cloning into 'simple-webapp-docker'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 14 (delta 3), reused 2 (delta 2), pack-reused 7
Receiving objects: 100% (14/14), done.
Resolving deltas: 100% (3/3), done.
root@Nilam:k8sCluster_kubeadm_terraform# ll
```

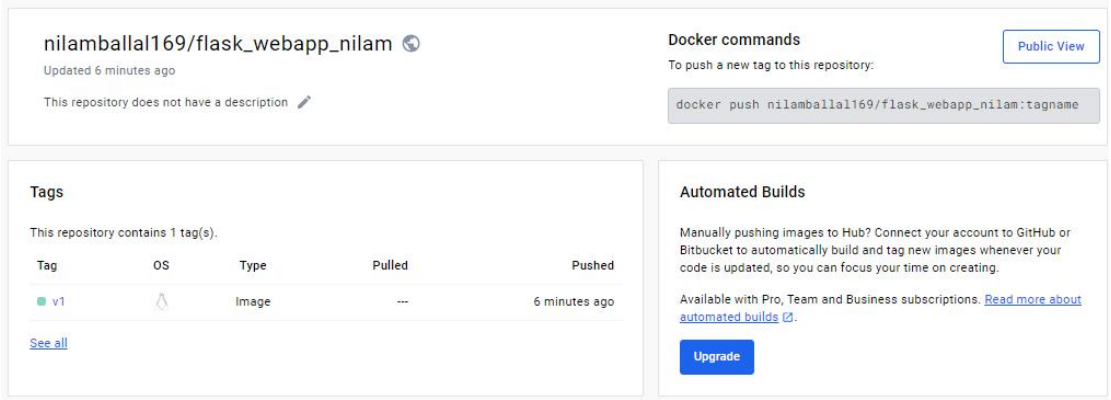
● Initialize a local repository & copy the code from above repo to your local repository in your working branch.

```
root@Nilam:simple-webapp-docker# git branch
* master
root@Nilam:simple-webapp-docker#
root@Nilam:simple-webapp-docker#
```

● Once code is copied to the local repository, build the docker image & add meaningful tags with version 1 and push to Docker Hub registry.

```
root@Nilam:simple-webapp-docker# docker image build -t nilamballal169/flask_webapp_nilam:v1 .
[+] Building 13.3s (10/10) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                           0.0s
 => => transferring dockerfile: 233B                                                                           0.0s
 => [internal] load metadata for docker.io/library/ubuntu:20.04                                               11.1s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                  0.0s
 => [internal] load .dockerignore                                                                              0.0s
 => => transferring context: 2B                                                                                0.0s
 => [1/4] FROM docker.io/library/ubuntu:20.04@sha256:bb1c41682108d7040f74a103022816041c50d7b0c89e0d706a74b4e548616e54   0.0s
 => [internal] load build context                                                                              0.0s
 => => transferring context: 208B                                                                              0.0s
 => CACHED [2/4] RUN apt-get update && apt-get install -y python3 python3-pip                                  0.0s
 => CACHED [3/4] RUN pip install flask                                                                         0.0s
 => CACHED [4/4] COPY app.py /opt/                                                                             0.0s
 => exporting to image                                                                                         0.0s
 => => exporting layers                                                                                        0.0s
 => => writing image sha256:493ec2070c58cea5d640e3961f042184d2880cb23faf91136729a6d4b3eaf05b                   0.0s
 => => naming to docker.io/nilamballal169/flask_webapp_nilam:v1                                                0.0s
```

```
root@Nilam:simple-webapp-docker# docker image push  nilamballal169/flask_webapp_nilam:v1
The push refers to repository [docker.io/nilamballal169/flask_webapp_nilam]
5433c09d294c: Layer already exists
b6813bc056ca: Layer already exists
dd76f60b2ee0: Layer already exists
28da0445c449: Layer already exists
v1: digest: sha256:c48ada368373db67e5ad8bd6cc50e5935b4609fb2b4e2e4d56d84fbbe4fbe0d5 size: 1160
```



● Once Images are pushed to Docker hub registries, create a directory named kube. Inside the kube directory create deployement.yaml file with 3 replication , labels app: flask-webapp , containerPort: 8080 and add the image that you have pushed in Docker Hub registry.

```
root@Nilam:Kube# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-webapp
spec:
  replicas: 3
  selector:
    matchLabels:
      app: flask-webapp
  template:
    metadata:
      labels:
        app: flask-webapp
    spec:
      containers:
      - name: flask-webapp
        image: nilamballal169/flask_webapp_nilam:v1
        ports:
        - containerPort: 8080

root@Nilam:Kube#
```

● Create one service.yaml file with type nodeport & select flask-webapp with port 8080 & targetPort 8080 with any nodePort between range 30000-32768.

```
root@Nilam:Kube# cat srv.yaml
apiVersion: v1
kind: Service
metadata:
  name: flask-webapp-service
spec:
  selector:
   app: flask-webapp
  ports:
  - port: 8080
    targetPort: 8080
    nodePort: 30011
  type: NodePort
root@Nilam:Kube#
```

```
root@Nilam:kube# kubectl apply -f .
deployment.apps/flask-webapp created
service/flask-webapp-service created
```

```
root@Nilam:Kube# kgp
NAME                            READY   STATUS    RESTARTS   AGE
flask-webapp-67648d4968-l4rpd   1/1     Running   0          88s
flask-webapp-67648d4968-vxhfp   1/1     Running   0          88s
flask-webapp-67648d4968-w7d48   1/1     Running   0          88s
root@Nilam:Kube#
```

● Once a service is created try accessing the web page in the browser as below. (30011 is nodeport mentioned in service.yaml). Meanwhile open app.py from your code to understand paths & output.
http://master_ip:30011/

Welcome!

I am good, how about you?

http://master_ip:30011/how are you

● Now , update the app.py from your code and add below route above if __name__ == "__main__" line

@app.route('/Who are you')

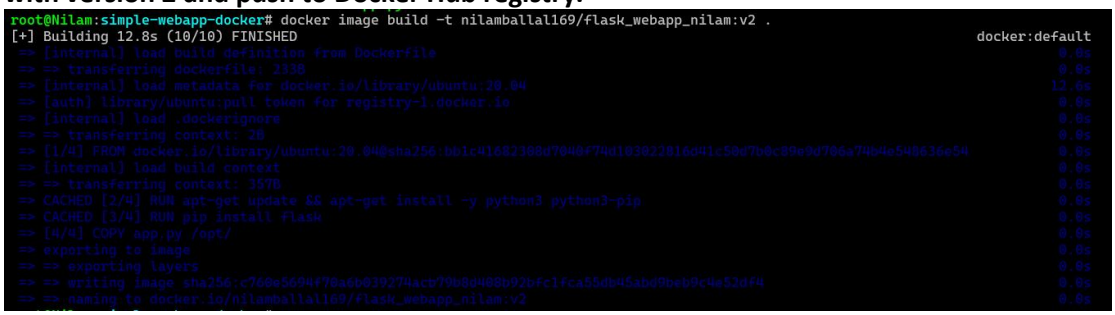
def cloudethix():

return 'Yes, I am cloudethix, and You !!!'

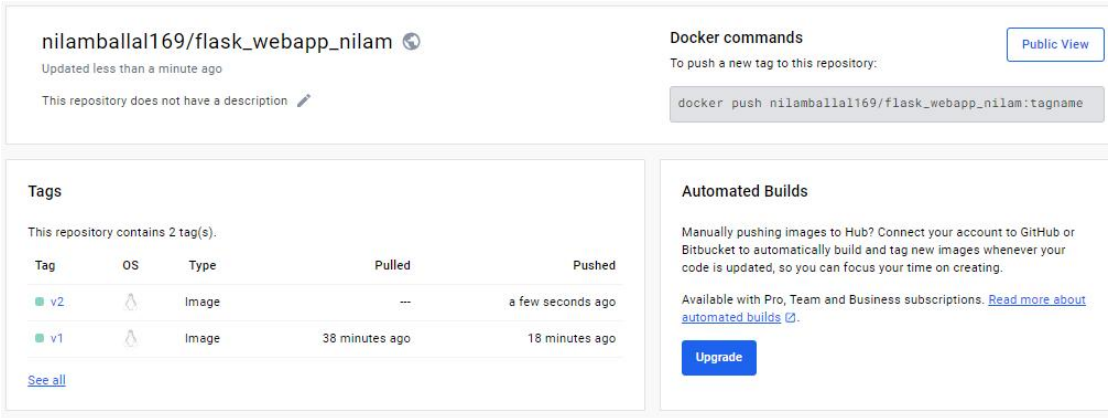
```
root@Nilam:simple-webapp-docker# cat app.py
import os
from flask import Flask
app = Flask(__name__)

@app.route("/")
def main():
    return "Welcome!"

@app.route('/how are you')
def hello():
    return 'I am good, how about you?'

if __name__ == "__main__":
    app.run()
root@Nilam:simple-webapp-docker#
```

```
root@Nilam:simple-webapp-docker# cat app.py
import os
from flask import Flask
app = Flask(__name__)

@app.route("/")
def main():
    return "Welcome!"

@app.route('/how are you')
def hello():
    return 'I am good, how about you?'

@app.route('/Who are you')
def cloudethix():
    return 'Yes, I am cloudethix, and You !!!'

if __name__ == "__main__":
    app.run()
root@Nilam:simple-webapp-docker#
```

● Once the file is updated , rebuild the docker image & add meaningful tags with version 2 and push to Docker Hub registry.



```
root@Nilam:simple-webapp-docker# docker image push  nilamballal169/flask_webapp_nilam:v2
The push refers to repository [docker.io/nilamballal169/flask_webapp_nilam]
f87be2ea6175: Pushed
b6813bc056ca: Layer already exists
dd76f60b2ee0: Layer already exists
28da0445c449: Layer already exists
v2: digest: sha256:34dec574029f9893800172103d65383eecddfa0879648324efa39ead21faed49 size: 1160
root@Nilam:simple-webapp-docker#
```



● Now we have the latest docker image in repo, It's time to roll out a new image. Roll out the new Image with all three ways one by one.

1. With kubectl set command

2. With kubectl edit deployment

3. With deployment.yaml file modification. **● Run the # kubectl rollout command to check status and history.**

● Note:- Once above step 1 is done , run # kubectl rollout undo deployment

```
root@Nilam:Kube# kubectl set image deployment flask-webapp flask-webapp=nilamballal169/flask_webapp_nilam:v2 -
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/flask-webapp image updated
root@Nilam:Kube#
root@Nilam:Kube# kubectl edit deployment flask-webapp --record
Flag --record has been deprecated, --record will be removed in the future
Edit cancelled, no changes made.
root@Nilam:Kube# kubectl edit deployment flask-webapp --record
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/flask-webapp edited
root@Nilam:Kube#
root@Nilam:Kube# kubectl rollout status deployment flask-webapp
Waiting for deployment "flask-webapp" rollout to finish: 1 out of 3 new replicas have been updated...
^Croot@Nilam:Kube# kubectedit deployment flask-webapp --recordpp
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/flask-webapp edited
root@Nilam:Kube# kubectl rollout status deployment flask-webapp
deployment "flask-webapp" successfully rolled out
root@Nilam:Kube#
root@Nilam:Kube# kubectl rollout pause deployment flask-webapp
deployment.apps/flask-webapp paused
root@Nilam:Kube#
root@Nilam:Kube# kubectl rollout resume deployment flask-webapp
deployment.apps/flask-webapp resumed
```

```
root@Nilam:Kube# kubectl rollout history deployment flask-webapp
deployment.apps/flask-webapp
REVISION  CHANGE-CAUSE
1         <none>
3         kubectl edit deployment flask-webapp --record=true
4         kubectl edit deployment flask-webapp --record=true

root@Nilam:Kube#
root@Nilam:Kube#
root@Nilam:Kube# kubectl rollout undo deployment flask-webapp --to-revision=3
deployment.apps/flask-webapp rolled back
root@Nilam:Kube# kubectl rollout history deployment flask-webapp
deployment.apps/flask-webapp
REVISION  CHANGE-CAUSE
1         <none>
4         kubectl edit deployment flask-webapp --record=true
5         kubectl edit deployment flask-webapp --record=true
```

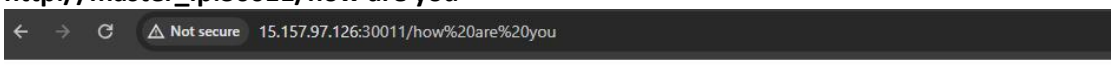
command to rollback the change and then try a second way of rollout.

● In the browser run all three routes & notice the changes.

http://master_ip:30011/

Welcome!

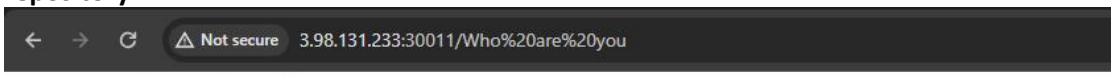
http://master_ip:30011/how are you



I am good, how about you?

http://master_ip:30011/Who are you

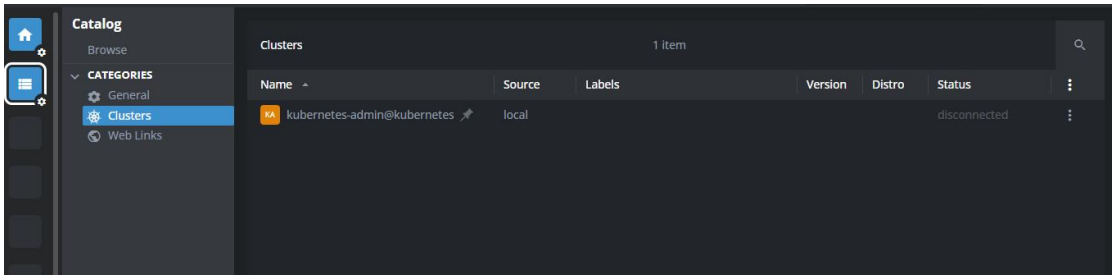
● Once done with all above steps , commit all the changes to the remote repository.



Yes, I am cloudethix, and You !!!

Que 9 →

- Download and install Lens & access your k8s cluster from Lens.
- Create nginx Pod and Nodeport service. Check the Pod logs from Lens.
- Check the service from lens. Also login to the pod shell using the lens.
- Take snaps and delete the resources that you have just created.




```
root@Nilam:Ques09# cat nginxpod.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

```
root@Nilam:Ques09# cat srv.yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: NodePort

root@Nilam:Ques09#
```

```
root@Nilam:Ques09# kubectl apply -f .
deployment.apps/nginx-deployment created
service/nginx-service created
root@Nilam:Ques09# kgp
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7c79c4bf97-bfmzr   1/1     Running   0          3s
root@Nilam:Ques09#
```

Name	Namespace	Contai...	CPU	Memory	Restarts	Controlled By	Node	QoS	Age	Status
nginx-d	project01	■	N/A	N/A	0	ReplicaSet	worker-1	BestEffort	5m15s	Running

Terminal | Pod nginx-deployment-7c79c4bf97-bfmzr

Owner: ReplicaSet nginx-deployment-7c79c4... Pod: nginx-deployment-7c79c4bf97-bfmzr Container: nginx Search...

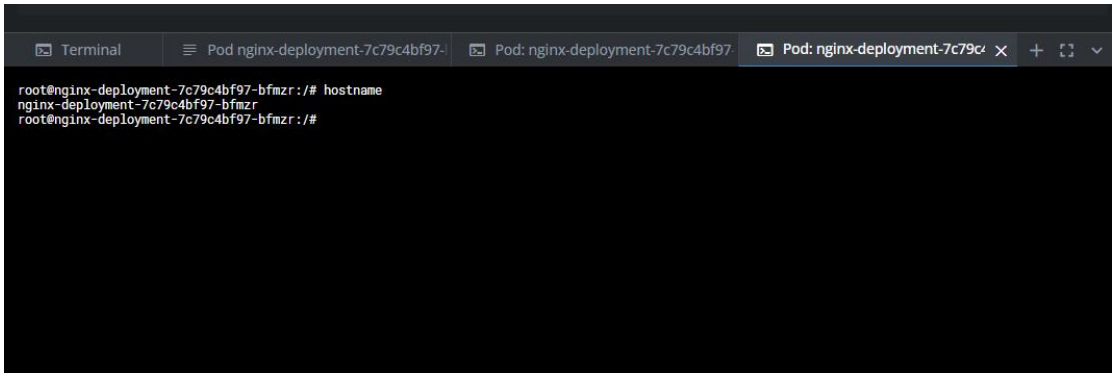
```
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/02/22 07:27:45 [notice] 1#1: using the "epoll" event method
2024/02/22 07:27:45 [notice] 1#1: nginx/1.25.4
2024/02/22 07:27:45 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/02/22 07:27:45 [notice] 1#1: OS: Linux 5.4.0-1103-aws
2024/02/22 07:27:45 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1024:4096
2024/02/22 07:27:45 [notice] 1#1: start worker processes
2024/02/22 07:27:45 [notice] 1#1: start worker process 29
```

Properties

Created	7m 26s ago 2024-02-22T12:57:44+05:30
Name	nginx-service
Namespace	project01
Annotations	kubectl.kubernetes.io/last-applied-configuration={"apiVersion":"v1...
Selector	app=nginx
Type	NodePort
Session Affinity	None

Connection

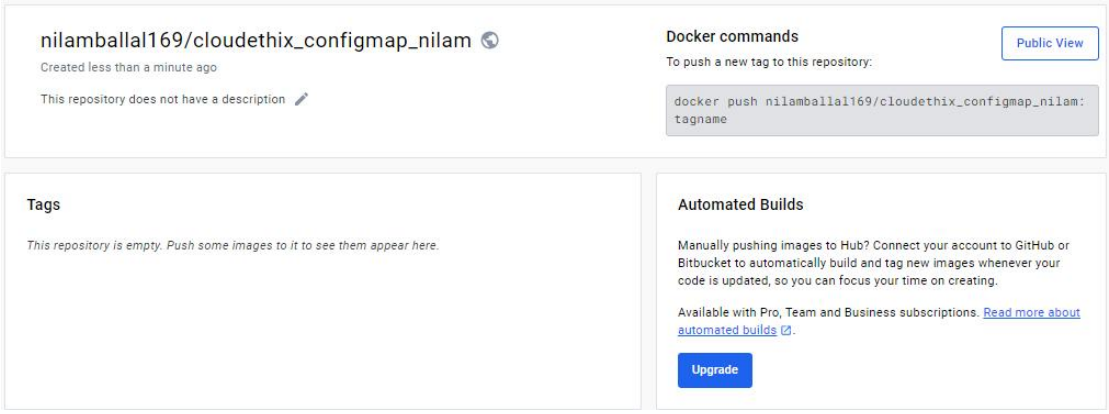
Cluster IP	10.104.153.86
Cluster IPs	10.104.153.86



==========

Que 10 →

● Create 1 Public Docker Hub registry named cloudethix_configmap_yourname.



● Clone below repository on your system.
https://github.com/zembutsu/docker-sample-nginx.git

```
root@Nilam:Ques10# git clone https://github.com/zembutsu/docker-sample-nginx.git
Cloning into 'docker-sample-nginx'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 22 (delta 7), reused 6 (delta 6), pack-reused 10
Receiving objects: 100% (22/22), done.
Resolving deltas: 100% (7/7), done.
root@Nilam:Ques10# cd docker-sample-nginx/
root@Nilam:docker-sample-nginx# ll
total 4
drwxrwxrwx 1 nilam nilam  512 Feb 22 13:17 ./
drwxrwxrwx 1 nilam nilam  512 Feb 22 13:17 ../
drwxrwxrwx 1 nilam nilam  512 Feb 22 13:17 .git/
-rwxrwxrwx 1 nilam nilam   95 Feb 22 13:17 Dockerfile*
-rwxrwxrwx 1 nilam nilam 1084 Feb 22 13:17 LICENSE*
-rwxrwxrwx 1 nilam nilam   73 Feb 22 13:17 README.md*
-rwxrwxrwx 1 nilam nilam  286 Feb 22 13:17 default.conf*
-rwxrwxrwx 1 nilam nilam  103 Feb 22 13:17 index.html*
root@Nilam:docker-sample-nginx#
```

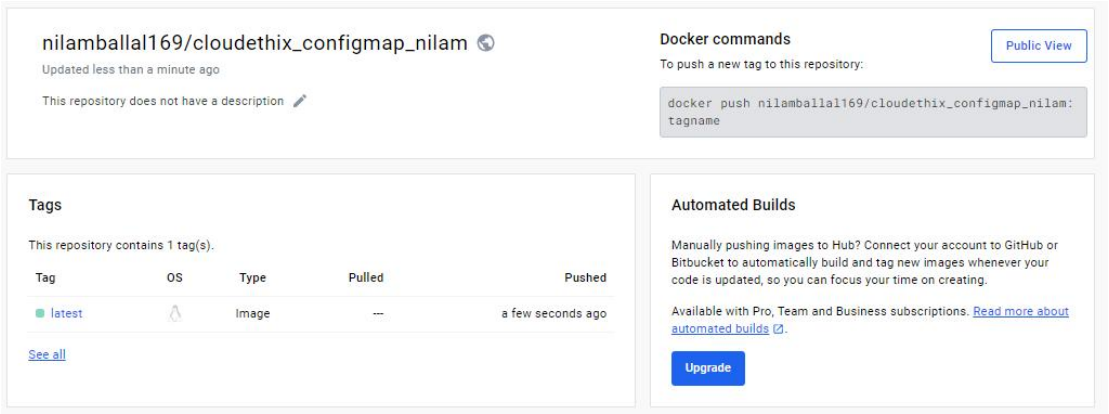
● Initialize a local repository & copy the code from above repo to your local repository in the working branch.

```
root@Nilam:docker-sample-nginx# git branch
* master
root@Nilam:docker-sample-nginx#
```

● Once code is copied , build a docker image from docker file and add meaningful tags and push to docker hub repository.

```
root@Nilam:docker-sample-nginx# docker image build -t nilamballal169/cloudethix_configmap_nilam:latest .
[+] Building 24.8s (9/9) FINISHED                                                                                docker:default
 => [internal] load build definition from Dockerfile                                                                      0.0s
 => => transferring dockerfile: 132B                                                                                      0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                                          24.5s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                              0.0s
 => [internal] load .dockerignore                                                                                         0.0s
 => => transferring context: 2B                                                                                           0.0s
 => [1/3] FROM docker.io/library/nginx:alpine@sha256:6a2f8b28e45c4adea04ec207a251fd4a2df03ddc930f782af51e315ebc76e9a9     0.0s
 => [internal] load build context                                                                                         0.0s
 => => transferring context: 469B                                                                                         0.0s
 => CACHED [2/3] COPY default.conf /etc/nginx/conf.d/                                                                     0.0s
 => [3/3] COPY index.html /usr/share/nginx/html/                                                                          0.1s
 => exporting to image                                                                                                    0.1s
 => => exporting layers                                                                                                   0.0s
 => => writing image sha256:b05a17e9109c5c3f940dff2f8c8aaf58c5fd1ce8400e9fb5a4de4d89e8304aeb                              0.0s
 => => naming to docker.io/nilamballal169/cloudethix_configmap_nilam:latest                                               0.0s
root@Nilam:docker-sample-nginx#
```

```
root@Nilam:docker-sample-nginx# docker image push  nilamballal169/cloudethix_configmap_nilam:latest
The push refers to repository [docker.io/nilamballal169/cloudethix_configmap_nilam]
35323508ddee: Pushed
2abf08b521cd: Mounted from nilamballal169/cloudethix_release_nginx_nilam
667a247707f0: Mounted from nilamballal169/cloudethix_release_nginx_nilam
d8527026595f: Mounted from nilamballal169/cloudethix_release_nginx_nilam
2593b08e5428: Mounted from nilamballal169/cloudethix_release_nginx_nilam
9909978d630d: Mounted from nilamballal169/cloudethix_release_nginx_nilam
c5140fc719dd: Mounted from nilamballal169/cloudethix_release_nginx_nilam
3137f8f0c641: Mounted from nilamballal169/cloudethix_release_nginx_nilam
718db50a47c0: Mounted from nilamballal169/cloudethix_release_nginx_nilam
aedc3bda2944: Mounted from nilamballal169/cloudethix_release_nginx_nilam
latest: digest: sha256:85d75b350d9c87563875194ec8f95aab8036e57e48a556056955182f4e888d38 size: 2403
root@Nilam:docker-sample-nginx#
```



● Once Images are pushed to Docker hub registries, create a directory named kube. Inside the kube directory create deployement.yaml file with 3replication , labels app: frontend-webapp , containerPort: 80 and add the image that you have pushed in Docker Hub registry.

```
root@Nilam:kube# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend-webapp
spec:
  replicas: 3
  selector:
    matchLabels:
      app: frontend-webapp
  template:
    metadata:
      labels:
        app: frontend-webapp
    spec:
      containers:
      - name: frontend-webapp
        image: nilamballal169/cloudethix_configmap_nilam:latest
        ports:
        - containerPort: 80

root@Nilam:kube#
```

● Create one service.yaml file with type nodeport & select frontend-webapp pod with port 80 & targetPort 80 with any nodePort between range 30000-32768.

```
root@Nilam:kube# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: frontend-webapp-svc
spec:
  selector:
    app: frontend-webapp
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: NodePort
  nodePort: 30011

root@Nilam:kube#
```

● Once the service is created try accessing the web page in the browser as below. Notice the changes & take the snap.

```
root@Nilam:kube# kubectl apply -f .
deployment.apps/frontend-webapp created
service/frontend-webapp-service created
root@Nilam:kube#
root@Nilam:kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
frontend-webapp-64458ddb5f-cjvnj    1/1     Running   0          7s
frontend-webapp-64458ddb5f-mrvw5    1/1     Running   0          7s
frontend-webapp-64458ddb5f-x49vg    1/1     Running   0          7s
nginx-deployment-7c79c4bf97-bfmzr   1/1     Running   0          40m
root@Nilam:kube#
```

Host: frontend-webapp-64458ddb5f-cjvnj

Version: 1.1

● Now create a configmap.yaml file with below data & delete the deployment that you have created.

<html>
<body>
<h1> I am Cloudethix Team, Are you ?!! </h1>
Version: 1.1
</body>
</html>

```
root@Nilam:kube# cat configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: frontend-webapp-configmap
data:
  index.html: |
    <html>
    <body>
    <h1> I am Cloudethix Team, Are you ?!! </h1>
    Version: 1.1
    </body>
    </html>


root@Nilam:kube#
```

```
root@Nilam:kube# kgp
NAME                              READY   STATUS    RESTARTS   AGE
frontend-webapp-64458ddb5f-dv2xw  1/1     Running   0          4s
frontend-webapp-64458ddb5f-m779p  1/1     Running   0          4s
frontend-webapp-64458ddb5f-rh57l  1/1     Running   0          4s
nginx-deployment-7c79c4bf97-bfmzr 1/1     Running   0          57m
root@Nilam:kube#
root@Nilam:kube#
root@Nilam:kube# kubectl delete deployment frontend-webapp
deployment.apps "frontend-webapp" deleted
root@Nilam:kube# kgp
NAME                              READY   STATUS    RESTARTS   AGE
nginx-deployment-7c79c4bf97-bfmzr 1/1     Running   0          57m
root@Nilam:kube#
```

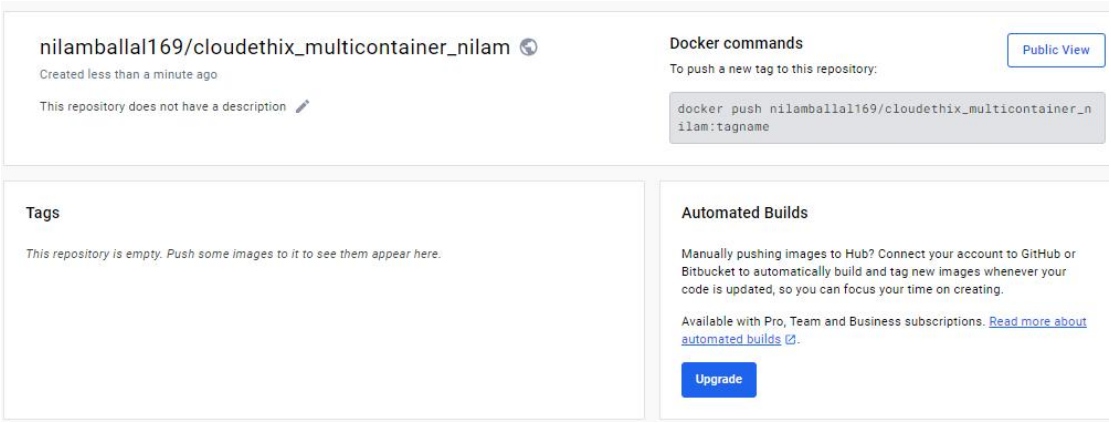
● Then update the same deployment.yaml file and mount configmap as volume on container using volumeMounts with mountPath /usr/share/nginx/html/

```
root@Nilam:kube# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend-webapp
spec:
  replicas: 3
  selector:
    matchLabels:
      app: frontend-webapp
  template:
    metadata:
      labels:
        app: frontend-webapp
    spec:
      containers:
      - name: frontend-webapp
        image: nilamballal169/cloudethix_configmap_nilam:latest
        ports:
        - containerPort: 80
        volumeMounts:
        - name: config-volume
          mountPath: /usr/share/nginx/html
      volumes:
      - name: config-volume
        configMap:
          name: frontend-webapp-configmap
root@Nilam:kube#
```

● Now it's time to create configmap & deployment. Once created , try to access the webpage in the browser & confirm that the index page is the same as we have in configmap.

Que 11 →

● Create 1 Public Docker Hub registry named cloudethix_multicontainer_yourname.



● Clone below repository on your system.
https://github.com/janakiramm/Kubernetes-multi-container-pod.git

```
root@Nilam:Ques11# git clone https://github.com/janakiramm/Kubernetes-multi-container-pod.git
Cloning into 'Kubernetes-multi-container-pod'...
remote: Enumerating objects: 51, done.
remote: Total 51 (delta 0), reused 0 (delta 0), pack-reused 51
Receiving objects: 100% (51/51), 88.14 KiB | 3.26 MiB/s, done.
Resolving deltas: 100% (21/21), done.
root@Nilam:Ques11# ll
total 0
drwxrwxrwx 1 nilam nilam 512 Feb 22 14:10 ./
drwxrwxrwx 1 nilam nilam 512 Feb 22 14:10 ../
drwxrwxrwx 1 nilam nilam 512 Feb 22 14:11 Kubernetes-multi-container-pod/
root@Nilam:Ques11# cd Kubernetes-multi-container-pod/
root@Nilam:Kubernetes-multi-container-pod# ll
total 120
drwxrwxrwx 1 nilam nilam    512 Feb 22 14:11 ./
drwxrwxrwx 1 nilam nilam    512 Feb 22 14:10 ../
drwxrwxrwx 1 nilam nilam    512 Feb 22 14:11 .git/
-rwxrwxrwx 1 nilam nilam      9 Feb 22 14:11 .gitignore*
drwxrwxrwx 1 nilam nilam    512 Feb 22 14:11 Build/
drwxrwxrwx 1 nilam nilam    512 Feb 22 14:11 Deploy/
-rwxrwxrwx 1 nilam nilam   2550 Feb 22 14:11 README.md*
-rwxrwxrwx 1 nilam nilam 116003 Feb 22 14:11 multi-container-pod.png*
root@Nilam:Kubernetes-multi-container-pod#
```

● Initialize a local repository & copy the code from above repo to your local repository in any of your working branches.

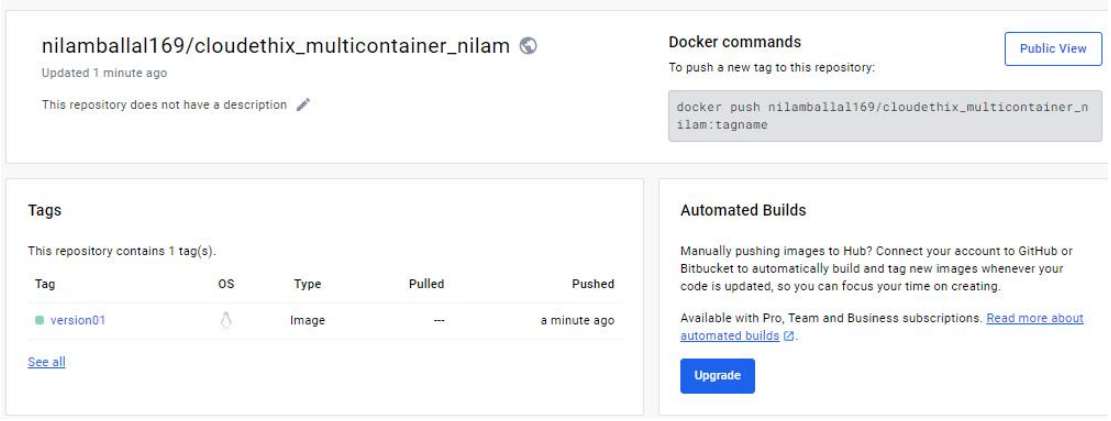
```
root@Nilam:Kubernetes-multi-container-pod# git branch
* master
root@Nilam:Kubernetes-multi-container-pod#
root@Nilam:Kubernetes-multi-container-pod#
root@Nilam:Kubernetes-multi-container-pod#
```

● Once code is copied , go to the Build directory and build docker image from docker file and add meaningful tags and push to docker hub repository.

```
root@Nilam:Kubernetes-multi-container-pod# cd Build/
root@Nilam:Build# ll
total 4
drwxrwxrwx 1 nilam nilam  512 Feb 22 14:11 ./
drwxrwxrwx 1 nilam nilam  512 Feb 22 14:11 ../
-rwxrwxrwx 1 nilam nilam   62 Feb 22 14:11 Dockerfile*
-rwxrwxrwx 1 nilam nilam 1607 Feb 22 14:11 app.py*
-rwxrwxrwx 1 nilam nilam  242 Feb 22 14:11 docker-compose.yml*
-rwxrwxrwx 1 nilam nilam   24 Feb 22 14:11 requirements.txt*
root@Nilam:Build#
```

```
root@Nilam:Build# docker image build --no-cache -t nilamballal169/cloudethix_multicontainer_nilam:version01 .
[+] Building 18.2s (10/10) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                          0.0s
 => => transferring dockerfile: 99B                                                                                           0.0s
 => [internal] load metadata for docker.io/library/python:2.7-onbuild                                                        12.6s
 => [auth] library/python:pull token for registry-1.docker.io                                                                 0.0s
 => [internal] load .dockerignore                                                                                             0.0s
 => => transferring context: 2B                                                                                               0.0s
 => CACHED [1/1] FROM docker.io/library/python:2.7-onbuild@sha256:5af88e1d011bf7e8405e83013712d09f91be1a39e2ede092808fc53e0a0ce1  0.0s
 => [internal] load build context                                                                                             0.0s
 => => transferring context: 150B                                                                                             0.0s
 => [2/1] COPY requirements.txt /usr/src/app/                                                                                 0.1s
 => [3/1] RUN pip install --no-cache-dir -r requirements.txt                                                                  5.2s
 => [4/1] COPY . /usr/src/app                                                                                                 0.0s
 => exporting to image                                                                                                        0.1s
 => => exporting layers                                                                                                       0.1s
 => => writing image sha256:d8b7520le98e9d2e981e017b1989c6585130b4efe5fb9a4c6833598409c8bb92b                                  0.0s
 => => naming to docker.io/nilamballal169/cloudethix_multicontainer_nilam:version01                                           0.0s
root@Nilam:Build#
```

```
root@Nilam:Build# docker image push  nilamballal169/cloudethix_multicontainer_nilam:version01
The push refers to repository [docker.io/nilamballal169/cloudethix_multicontainer_nilam]
5df2fd5def35: Layer already exists
7496c1803c9e: Pushed
73e0e5f7af46: Pushed
3e397f5b8357: Layer already exists
e257add70b4b: Layer already exists
ce7e990ce056: Layer already exists
633d23790c1d: Layer already exists
d071a18d9802: Layer already exists
8451f9fe0016: Layer already exists
858cd8541f7e: Layer already exists
a42d312a03bb: Layer already exists
dd1eb1fd7e08: Layer already exists
version01: digest: sha256:7f9919a7e09d30187eb4a9d1b9b20aa57f36586fba1e70576e9ca19ff6a0f3f6 size: 2844
root@Nilam:Build#
root@Nilam:Build#
```



```
root@Nilam:Kubernetes-multi-container-pod# cd Deploy/
root@Nilam:Deploy# ll
total 1
drwxrwxrwx 1 nilam nilam 512 Feb 22 14:11 ./
drwxrwxrwx 1 nilam nilam 512 Feb 22 14:11 ../
-rwxrwxrwx 1 nilam nilam 325 Feb 22 14:11 db-pod.yml*
-rwxrwxrwx 1 nilam nilam 203 Feb 22 14:11 db-svc.yml*
-rwxrwxrwx 1 nilam nilam 467 Feb 22 14:11 web-pod-1.yml*
-rwxrwxrwx 1 nilam nilam 467 Feb 22 14:11 web-pod-2.yml*
-rwxrwxrwx 1 nilam nilam 644 Feb 22 14:11 web-rc.yml*
-rwxrwxrwx 1 nilam nilam 218 Feb 22 14:11 web-svc.yml*
root@Nilam:Deploy#
```

●

Now go to the deploy directory and notice the files.● Here, web-pod-1.yml file will create the pod with two containers (Multi
container). Take a note of lables , name of containers and ports. Also,
please make sure you will update the python container image that you have
pushed to your docker registry.

```
root@Nilam:Deploy# cat web-pod-1.yml
apiVersion: v1
kind: Pod
metadata:
  name: web1
  labels:
    name: web
    app: demo
spec:
  containers:
    - name: redis
      image: redis
      ports:
        - containerPort: 6379
          name: redis
          protocol: TCP
    - name: python
      image: nilamballal169/cloudethix_multicontainer_nilam:version01
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          name: http
          protocol: TCP
root@Nilam:Deploy#
```

```
root@Nilam:Deploy# kubectl apply -f web-pod-1.yml
pod/web1 created
root@Nilam:Deploy# kgp
NAME   READY   STATUS    RESTARTS   AGE
web1   2/2     Running   0          5s
```

● Now, open web-svc.yml file and notice service Type , selectors & targetPort.
Apply the file.

```
root@Nilam:Deploy# cat web-svc.yml
apiVersion: v1
kind: Service
metadata:
  name: web
  labels:
    name: web
    app: demo
spec:
  selector:
    name: web
  type: NodePort
  ports:
   - port: 80
     name: http
     targetPort: 5000
     protocol: TCProot@Nilam:Deploy#
```

● Now open db-pod.yml & notice the lables , name , Image, containerPort and apply the file.

```
root@Nilam:Deploy# cat db-pod.yml
apiVersion: "v1"
kind: Pod
metadata:
  name: mysql
  labels:
    name: mysql
    app: demo
spec:
  containers:
    - name: mysql
      image: mysql:5.7.25
      ports:
        - containerPort: 3306
          protocol: TCP
      env:
        -
          name: "MYSQL_ROOT_PASSWORD"
          value: "password"

root@Nilam:Deploy#
```

● Now open the db-svc.yml file and notice service Type , selectors & targetPort. Apply the file.

```
root@Nilam:Deploy# cat db-svc.yml
apiVersion: v1
kind: Service
metadata:
  name: mysql
  labels:
    name: mysql
    app: demo
spec:
  ports:
  - port: 3306
    name: mysql
    targetPort: 3306
  selector:
    name: mysql
    app: demoroot@Nilam:Deploy#
```

● Once above files are applied , Check that the Pods and Services are created using command line or lens.

```
root@Nilam:Deploy# kubectl apply -f db-pod.yml
pod/mysql created
root@Nilam:Deploy# kubectl apply -f db-svc.yml
service/mysql created
root@Nilam:Deploy# kubectl apply -f web-pod-1.yml
pod/web1 created
root@Nilam:Deploy# kubectl apply -f web-svc.yml
service/web created
root@Nilam:Deploy# kgp
NAME    READY   STATUS    RESTARTS   AGE
mysql   1/1     Running   0          29s
web1    2/2     Running   0          10s
root@Nilam:Deploy#
root@Nilam:Deploy#
root@Nilam:Deploy# kubectl get pod
NAME    READY   STATUS    RESTARTS   AGE
mysql   1/1     Running   0          55s
web1    2/2     Running   0          36s
root@Nilam:Deploy# kubectl get service
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP        6h37m
mysql        ClusterIP   10.100.136.9   <none>        3306/TCP       54s
web          NodePort    10.97.89.137   <none>        80:30012/TCP   34s
root@Nilam:Deploy#
```

● Now , from the command line run below urls & notice the changes.

 # curl http://$NODE_IP:$NODE_PORT/init

```
root@Nilam:Deploy#
root@Nilam:Deploy#
root@Nilam:Deploy# export NODE_IP="15.157.97.126"
root@Nilam:Deploy# export NODE_PORT="30012"
```

```
root@Nilam:Deploy# curl http://$NODE_IP:$NODE_PORT/init
DB Init doneroot@Nilam:Deploy#
root@Nilam:Deploy# curl http://$NODE_IP:$NODE_PORT/init
DB Init doneroot@Nilam:Deploy#
```

Initialize the database with sample schema

● Now it's time to Insert some sample data. Make sure you will use correct $NODE_IP:$NODE_PORT

 # curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"John Doe"}' http://$NODE_IP:$NODE_PORT/users/add

 # curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "2", "user":"Jane Doe"}' http://$NODE_IP:$NODE_PORT/users/add

 # curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "3", "user":"Bill Colls"}' http://$NODE_IP:$NODE_PORT/users/add

 # curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "4", "user":"Mike Taylor"}' http://$NODE_IP:$NODE_PORT/users/add

```
root@Nilam:Deploy# curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"John Doe"}' http://$NODE_IP:$NODE_POR
T/users/add
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "2", "user":"Jane Doe"}' http://$NODE_IP:$NODE_PORT/users/add
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "3", "user":"Bill Collins"}' http://$NODE_IP:$NODE_PORT/users/add
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "4", "user":"Mike Taylor"}' http://$NODE_IP:$NODE_PORT/users/add
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 5
Server: Werkzeug/1.0.1 Python/2.7.15
Date: Thu, 22 Feb 2024 12:48:11 GMT

Addedcurl: (7) Failed to connect to 15.157.97.126 port 30012 after 219 ms: Connection refused
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 5
Server: Werkzeug/1.0.1 Python/2.7.15
Date: Thu, 22 Feb 2024 12:48:11 GMT

AddedHTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 5
Server: Werkzeug/1.0.1 Python/2.7.15
Date: Thu, 22 Feb 2024 12:48:12 GMT
```

● Now access the data that we have added to database using below command.

 # curl http://$NODE_IP:$NODE_PORT/users/1●

```
root@Nilam:Deploy#
root@Nilam:Deploy# curl http://$NODE_IP:$NODE_PORT/users/1
John Doeroot@Nilam:Deploy#
```

The second time you access the data, it appends '(c)' indicating that it is pulled from the Redis cache.

```
root@Nilam:Deploy# kubectl create -f web-rc.yml
replicationcontroller/web created
root@Nilam:Deploy#
root@Nilam:Deploy# kubectl apply -f web-rc.yml
Warning: resource replicationcontrollers/web is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is requ
ired by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or
 kubectl apply. The missing annotation will be patched automatically.
replicationcontroller/web configured
```

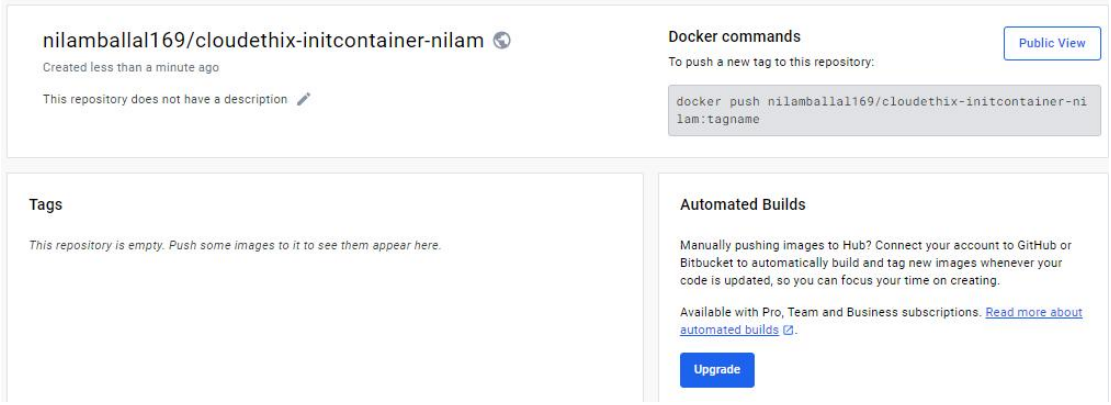
```
root@Nilam:Deploy# kgp
NAME        READY   STATUS    RESTARTS   AGE
mysql       1/1     Running   0          30m
web-5hbw2   2/2     Running   0          26s
web-6zdd5   2/2     Running   0          26s
web-99dz8   2/2     Running   0          26s
web-bwd2p   2/2     Running   0          26s
web-cm59b   2/2     Running   0          26s
web-fcmjz   2/2     Running   0          26s
web-l9zj8   2/2     Running   0          26s
web-n86x8   2/2     Running   0          26s
web-tw6s2   2/2     Running   0          26s
web1        2/2     Running   0          30m
root@Nilam:Deploy# curl http://$NODE_IP:$NODE_PORT/users/1
John Doeroot@Nilam:Deploy#
```

● Also, try to access mysql shell i.e db pod & run select * from the users table.
check app.py for DB related information.

```
root@Nilam:Deploy# kubectl exec -it mysql -- /bin/sh
# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 24
Server version: 5.7.25 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use USERDB
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from users;
+------+--------------+
| ID   | USER         |
+------+--------------+
|    1 | John Doe     |
|    3 | Bill Collins |
|    4 | Mike Taylor  |
+------+--------------+
3 rows in set (0.00 sec)

mysql> exit
```

Que 12 →

● Create 1 Public Docker Hub registry named cloudethix_Initcontainer_yourname.



● Clone below repository on your system.
https://github.com/janakiramm/simpleapp.git

```
root@Nilam:Ques12# git clone https://github.com/janakiramm/simpleapp.git
Cloning into 'simpleapp'...
remote: Enumerating objects: 47, done.
remote: Total 47 (delta 0), reused 0 (delta 0), pack-reused 47
Receiving objects: 100% (47/47), 8.20 KiB | 365.00 KiB/s, done.
Resolving deltas: 100% (9/9), done.
root@Nilam:Ques12# ll
total 0
drwxrwxrwx 1 nilam nilam 512 Feb 22  2024 ./
drwxrwxrwx 1 nilam nilam 512 Feb 22  2024 ../
drwxrwxrwx 1 nilam nilam 512 Feb 22  2024 simpleapp/
```

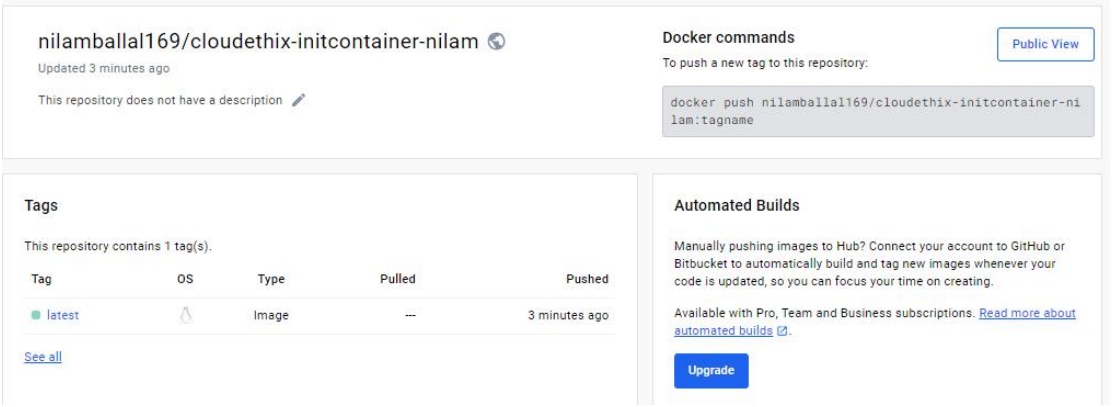
● Initialize a local repository & copy the code from above repo to your local repository in any of your working branch.

```
root@Nilam:simpleapp# git branch
* master
root@Nilam:simpleapp#
```

● Once code is copied , go to the Build directory and build docker image from docker file and add meaningful tags and push to docker hub repository.

```
root@Nilam:simpleapp# docker image build -t nilamballal169/cloudethix-initcontainer-nilam:latest .
[+] Building 2.2s (9/9) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                          0.1s
 => => transferring dockerfile: 122B                                                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                               1.9s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [internal] load build context                                                                             0.0s
 => => transferring context: 785B                                                                             0.0s
 => CACHED [1/3] FROM docker.io/library/nginx:latest@sha256:c26ae7472d624ba1fafd296e73cecc4f93f853088e6a9c13c0d52f6ca5865107  0.0s
 => [2/3] COPY wrapper.sh /                                                                                   0.1s
 => [3/3] COPY html /usr/share/nginx/html                                                                     0.0s
 => exporting to image                                                                                        0.1s
 => => exporting layers                                                                                       0.0s
 => => writing image sha256:89163089870c467de1954eb119a2390e4d57a33f875eb8d3fb38dea804ladd6                   0.0s
 => => naming to docker.io/nilamballal169/cloudethix-initcontainer-nilam:latest                               0.0s
root@Nilam:simpleapp#
```

```
root@Nilam:simpleapp#
root@Nilam:simpleapp# docker image push  nilamballal169/cloudethix-initcontainer-nilam:latest
The push refers to repository [docker.io/nilamballal169/cloudethix-initcontainer-nilam]
0019d3382407: Pushed
db8e783f58ca: Pushed
61a7fb4dabcd: Mounted from nilamballal169/k8s_project07
bcc6856722b7: Mounted from nilamballal169/k8s_project07
188d128a188c: Mounted from nilamballal169/k8s_project07
7d52a4114c36: Mounted from nilamballal169/k8s_project07
3137f8f0c641: Mounted from nilamballal169/cloudethix_configmap_nilam
84619992a45b: Mounted from nilamballal169/k8s_project07
ceb365432eec: Mounted from nilamballal169/k8s_project07
latest: digest: sha256:fc81cc7f5c50e7ebede32ce5867fc9f88a9066538efdc0c0469ecabe0f28494a size: 2192
root@Nilam:simpleapp#
```



● Once Images are pushed to Docker hub registries, create a directory named kube. Inside the kube directory create deployement.yaml file with 3 replication , label app: simpleapp-webapp , containerPort: 80 and add the image that you have pushed in Docker Hub registry.

```
root@Nilam:kube# cat deployement.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: simpleapp-webapp
spec:
  selector:
    matchLabels:
      app: simpleapp-webapp
  template:
    metadata:
      labels:
        app: simpleapp-webapp
    spec:
      containers:
      - name: simpleapp-webapp
        image: nilamballal169/cloudethix_initcontainer-nilam:latest
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 80
root@Nilam:kube#
```

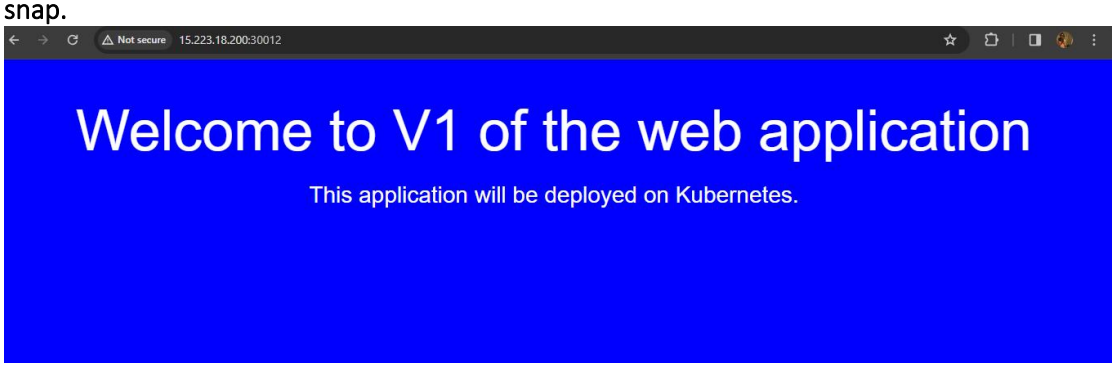
● Create one service.yaml file with type nodeport & select simpleapp-webapp pod with port 80 & targetPort 80 with any nodePort between range 30000-32768.

```
root@Nilam:kube# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: simpleapp-webapp
spec:
  selector:
    app: simpleapp-webapp
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
    nodePort: 30012
  type: NodePort


root@Nilam:kube#
```

```
root@Nilam:kube# kubectl apply -f .
deployment.apps/simpleapp-webapp created
service/simpleapp-webapp created
```

```
root@Nilam:kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
simpleapp-webapp-5fb8775796-njkjc   1/1     Running   0          50s
root@Nilam:kube#
```

● Open the webpage in the browser and notice the changes and capture the snap.



● Then delete the deployment that you have just created.

```
root@Nilam:kube# kubectl delete -f deployement.yaml
deployment.apps "simpleapp-webapp" deleted
root@Nilam:kube#
```

● Update the deployment.yaml file and add volumeMounts with mountPath /usr/share/nginx/html from emptyDir: {} volume.

● Once above changes are added, add initContainers block with below parameters. Also add volumeMounts for Init Container with mountPath "/work-dir" from emptyDir: {} volume.

initContainers:
- name: install
image: busybox:1.28
command:
- wget
- "-O"
- "/work-dir/index.html"
- http://info.cern.ch
volumeMounts:
- name: workdir
mountPath: "/work-dir"

● Add volumes with emptyDir: {} in deployment.yaml file.

```
root@Nilam:kube# cat deployement.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: simpleapp-webapp
spec:
  selector:
    matchLabels:
      app: simpleapp-webapp
  template:
    metadata:
      labels:
        app: simpleapp-webapp
    spec:
      initContainers:
      - name: install
        image: busybox:1.28
        command:
        - wget
        - "-O"
        - "/work-dir/index.html"
        - http://info.cern.ch
        volumeMounts:
        - name: workdir
          mountPath: "/work-dir"
      containers:
      - name: simpleapp-webapp
        image: nilamballal169/cloudethix_initcontainer-nilam:version01
        ports:
        - containerPort: 80
        volumeMounts:
        - name: html-volume
          mountPath: /usr/share/nginx/html
        - name: workdir
          mountPath: /work-dir
      volumes:
      - name: html-volume
        emptyDir: {}
      - name: workdir
        emptyDir: {}
root@Nilam:kube#
```

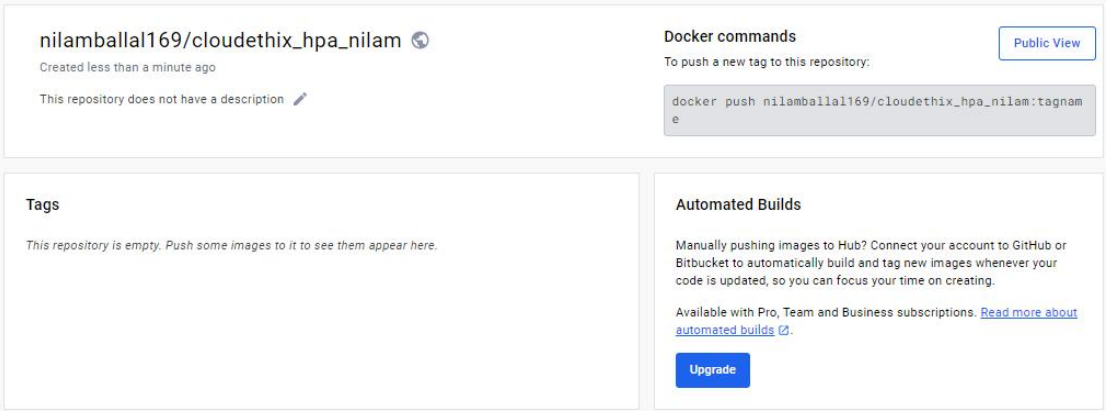
```
root@Nilam:kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
simpleapp-webapp-79848d4568-dp528   1/1     Running   0          91s
root@Nilam:kube# kgs
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes        ClusterIP   10.96.0.1        <none>        443/TCP        63m
simpleapp-webapp  NodePort    10.105.180.194   <none>        80:30013/TCP   45m
root@Nilam:kube#
```

● Once the deployment.yaml file is ready, create the deployment & access the page in the browser and notice the changes.

==========

Que 13 →

● Create 1 Public Docker Hub registry named cloudethix_hpa_yourname.



● Clone below repository on your system.

https://github.com/vivekamin/kubernetes-hpa-example.git

```
root@Nilam:Ques13# git clone https://github.com/vivekamin/kubernetes-hpa-example.git
Cloning into 'kubernetes-hpa-example'...
remote: Enumerating objects: 26, done.
remote: Total 26 (delta 0), reused 0 (delta 0), pack-reused 26
Receiving objects: 100% (26/26), done.
Resolving deltas: 100% (9/9), done.
root@Nilam:Ques13# ll
total 0
drwxrwxrwx 1 nilam nilam 512 Feb 23 12:27 ./
drwxrwxrwx 1 nilam nilam 512 Feb 23 12:26 ../
drwxrwxrwx 1 nilam nilam 512 Feb 23 12:27 kubernetes-hpa-example/
root@Nilam:Ques13# cd k
-bash: cd: k: No such file or directory
root@Nilam:Ques13# cd kubernetes-hpa-example/
root@Nilam:kubernetes-hpa-example#
```

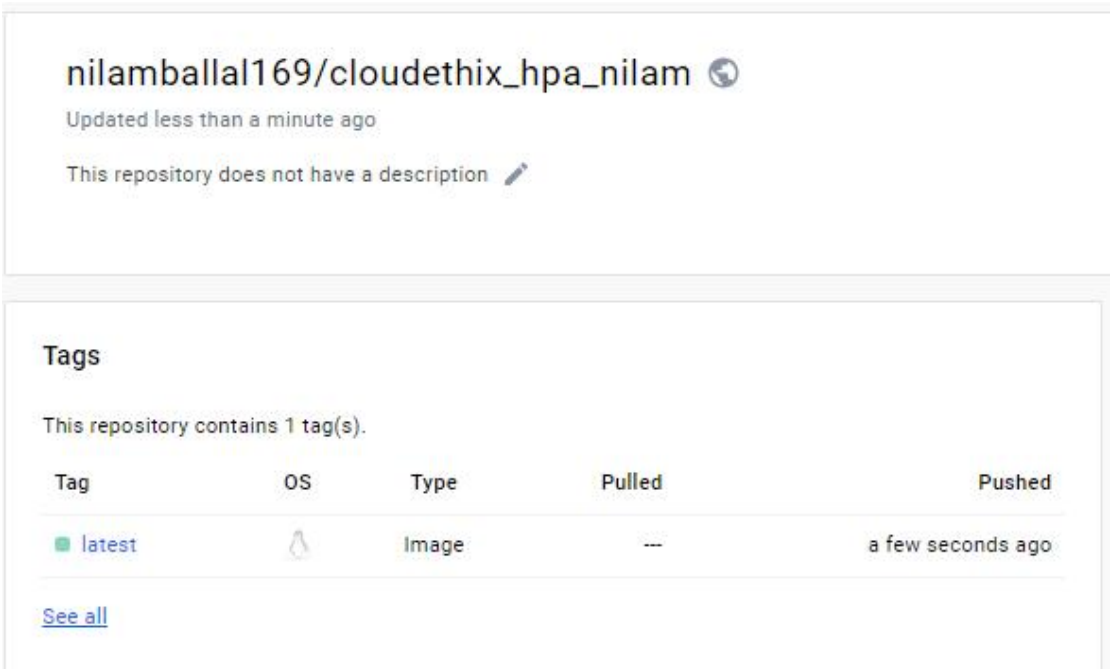
● Initialize a local repository & copy the code from above repo to your local repository in any of your working branch.

```
root@Nilam:kubernetes-hpa-example#  git config --global --add safe.directory /mnt/c/Users/manis/Music/Question12/k8sCluster_kubeadm_terraform/Ques13
/kubernetes-hpa-example
root@Nilam:kubernetes-hpa-example#
root@Nilam:kubernetes-hpa-example#
root@Nilam:kubernetes-hpa-example# git branch
* master
root@Nilam:kubernetes-hpa-example#
```

● Once code is copied , build a docker image from the docker file and add meaningful tags and push to the docker hub repository.

```
root@Nilam:kubernetes-hpa-example# docker image build -t nilamballal169/cloudethix_hpa_nilam:latest .
[+] Building 19.8s (11/11) FINISHED                                                         docker:default
 => [internal] load build definition from Dockerfile                                                  0.0s
 => => transferring dockerfile: 164B                                                                  0.0s
 => [internal] load metadata for docker.io/library/node:8.12.0-alpine                                14.0s
 => [auth] library/node:pull token for registry-1.docker.io                                           0.0s
 => [internal] load .dockerignore                                                                     0.0s
 => => transferring context: 2B                                                                       0.0s
 => [1/6] FROM docker.io/library/node:8.12.0-alpine@sha256:[illegible]                                3.6s
 => => resolve docker.io/library/node:8.12.0-alpine@sha256:[illegible]                                0.0s
 => => sha256:[illegible] 2.03kB / 2.03kB                                                             0.0s
 => => sha256:[illegible] 951B / 951B                                                                 0.0s
 => => sha256:[illegible] 5.09kB / 5.09kB                                                             0.0s
 => => sha256:[illegible] 2.21MB / 2.21MB                                                             1.4s
 => => sha256:[illegible] 18.82MB / 18.82MB                                                           2.3s
 => => sha256:[illegible] 1.08MB / 1.08MB                                                             0.9s
 => => extracting sha256:[illegible]                                                                  0.1s
 => => extracting sha256:[illegible]                                                                  1.2s
 => => extracting sha256:[illegible]                                                                  0.0s
 => [internal] load build context                                                                     0.2s
 => => transferring context: 38.34kB                                                                  0.2s
 => [2/6] RUN mkdir -p /usr/src/app                                                                   0.7s
 => [3/6] WORKDIR /usr/src/app                                                                        0.0s
 => [4/6] COPY . /usr/src/app                                                                         0.0s
 => [5/6] RUN npm install                                                                             1.3s
 => exporting to image                                                                                0.1s
 => => exporting layers                                                                               0.0s
 => => writing image sha256:[illegible]                                                               0.0s
 => => naming to docker.io/nilamballal169/cloudethix_hpa_nilam:latest                                 0.0s
root@Nilam:kubernetes-hpa-example#
```

```
root@Nilam:kubernetes-hpa-example# docker image push nilamballal169/cloudethix_hpa_nilam:latest
The push refers to repository [docker.io/nilamballal169/cloudethix_hpa_nilam]
2617299ccfcd: Pushed
014713deb4da: Pushed
5f70bf18a086: Mounted from nilamballal169/project-haproxy
7993d3ccdf24: Pushed
8b59e4cead98: Mounted from library/node
7aa09d2ca0a3: Mounted from library/node
df64d3292fd6: Mounted from library/node
latest: digest: sha256:886df1940f07163eb7018c063f89e0c72b18cab32748a445e5d11a8ab16b6f71 size: 1781
root@Nilam:kubernetes-hpa-example#
```



● Once the image is pushed, go to k8s directory and update deployment.yaml file with image name from your repo. And then create it.

```
root@Nilam:k8s# cat deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: node-example
  namespace: project-01
spec:
  replicas: 1
  selector:
    matchLabels:
      app: node-example
  template:
    metadata:
      labels:
        app: node-example
    spec:
      containers:
      - name: node-container
        image: nilamballal169/cloudethix_hpa_nilam:latest
        imagePullPolicy: Always
        ports:
        - containerPort: 3000
        resources:
          limits:
            cpu: "0.5"
          requests:
            cpu: "0.25"
root@Nilam:k8s#
```

● Open service.yml and change the type to nodePort and apply the same.

```
root@Nilam:k8s# cat service.yml
apiVersion: v1
kind: Service
metadata:
  name: hpa-service
  namespace: project-01
spec:
  type: NodePort
  selector:
    app: node-example
  ports:
  - protocol: TCP
    port: 3000
    targetPort: 3000
    nodePort: 30017
root@Nilam:k8s#
```

- Open the HPA.yaml file, notice it and then apply the same.

```
root@Nilam:k8s# cat hpa.yml
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: node-example
  namespace: default
spec:
  maxReplicas: 4
  minReplicas: 1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: node-example
  targetCPUUtilizationPercentage: 1
root@Nilam:k8s#
```

```
root@Nilam:k8s# kgp
NAME                            READY   STATUS    RESTARTS   AGE
node-example-64cf4c5c5d-zvwks   1/1     Running   0          85s
root@Nilam:k8s# kgs
NAME          TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hpa-service   NodePort   10.108.145.66   <none>        3000:30017/TCP   2m36s
root@Nilam:k8s#
```

● Open the browser, and access the webpage.
● Now it's time to test the HPA working with the below command.

 # kubectl run -i --tty load-generator --rm --image=busybox --restart=Never -- /bin/sh -c "while sleep 0.01; do wget -q -O http://NODE_PORT_SERVICE_NAME; done"

```
root@Nilam:k8s# kubectl run -i --tty load-generator-04 --rm --image=busybox --restart=Never -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://hpa-service:3000; done"
If you don't see a command prompt, try pressing enter.
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
```

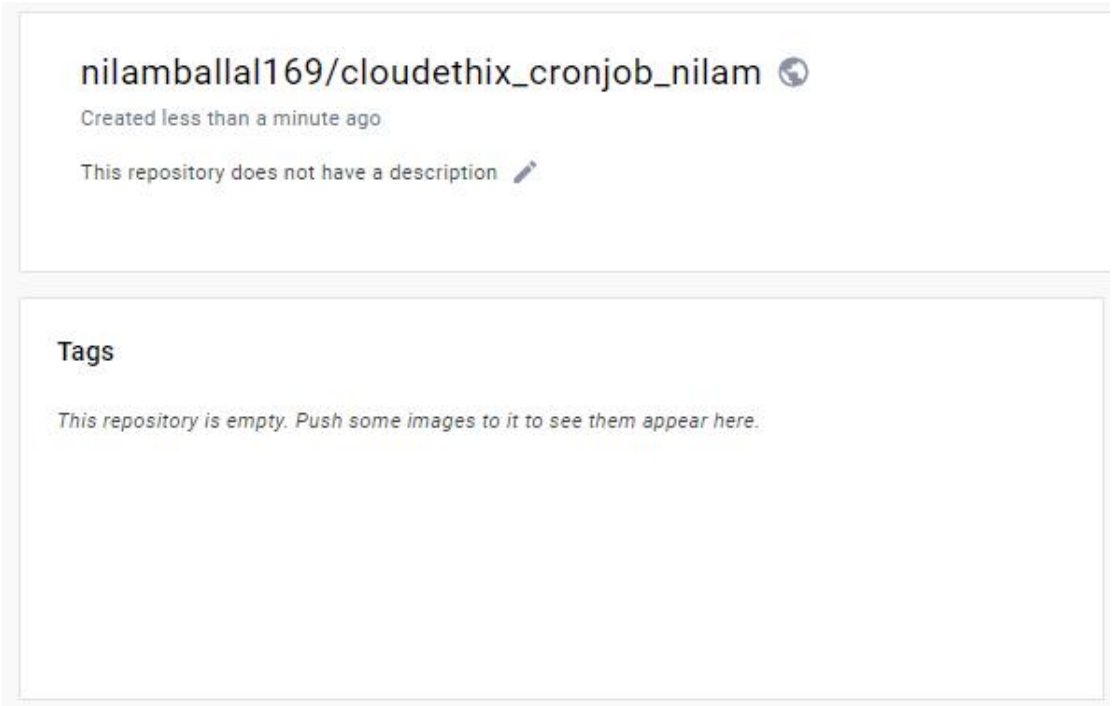
● Check the HPA from kubectl command and also check if the deployment is scaling up.

```
root@Nilam:k8s# kubectl get deployment node-example -n project-01
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
node-example   1/1     1            1           15m
```

● Take the snap , prepare a well formatted doc and write your understanding.

Que 14 →

● Create 1 Public Docker Hub registry named cloudethix_cronjob_yourname.



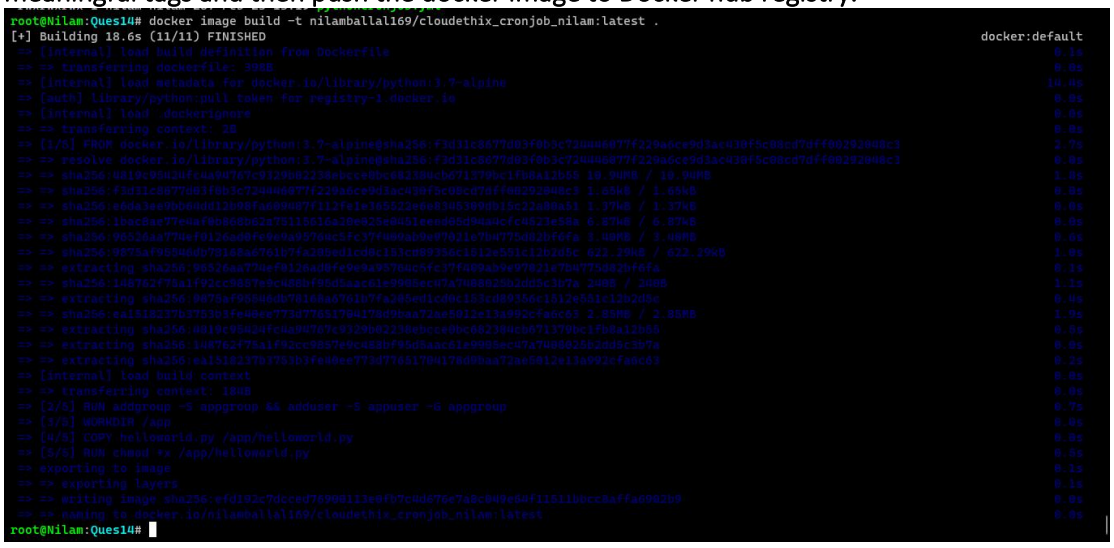
● Initialize a local repository & copy below code (three files) to your local repository in any of your working branch.

```
root@Nilam:Ques14# cat Dockerfile
FROM python:3.7-alpine
#add user group and ass user to that group
RUN addgroup -S appgroup && adduser -S appuser -G appgroup
#creates work dir
WORKDIR /app
#copy python script to the container folder app
COPY helloworld.py /app/helloworld.py
RUN chmod +x /app/helloworld.py
#user is appuser
USER appuser
ENTRYPOINT ["python", "/app/helloworld.py"]
```

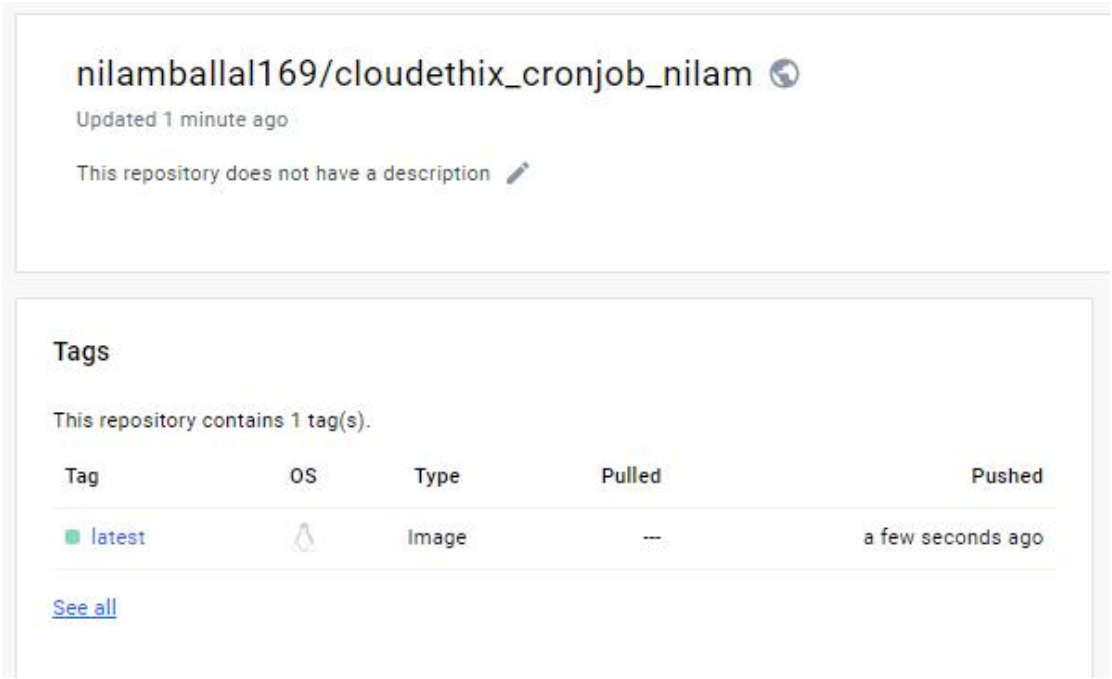
```
root@Nilam:Ques14# cat helloworld.py
!/usr/local/bin/python3
import datetime
x = datetime.datetime.now()
print("Welcome to the Cloudethix World")
print("Today is")
print(x)
root@Nilam:Ques14#
```

```
root@Nilam:Ques14# cat pythoncronjob.yml
apiVersion: batch/v1
kind: CronJob
metadata:
name: python-helloworld
spec:
schedule: "*/1 * * * *""
jobTemplate:
spec:
template:
spec:
containers:
- name: python-helloworld
image: python-helloworld
command: [/app/helloworld.py]
restartPolicy: OnFailure
root@Nilam:Ques14#
```

● Once code is copied, build the docker image from Dockerfile , add meaningful tags and then push the docker image to Docker hub registry.




```
root@Nilam:Ques14# docker image push  nilamballal169/cloudethix_cronjob_nilam:latest
The push refers to repository [docker.io/nilamballal169/cloudethix_cronjob_nilam]
b09c898ff569: Pushed
8ad0a76e3d32: Pushed
4fa8e8fc0c1f: Pushed
895091e6000a: Pushed
ae2ed3079163: Mounted from library/python
aa3a591fc84e: Mounted from library/python
7f29b11ef9dd: Mounted from library/python
a1c2f058ec5f: Mounted from library/python
cc2447e1835a: Mounted from library/python
latest: digest: sha256:49f82816b67767edeee9b3860f2f4eb2104563ccabe85c168a9ecc460a24573b size: 2197
root@Nilam:Ques14#
```



● Now update the pythoncronjob.yml file to change the image name that you have just pushed to docker hub registry.

```
root@Nilam:Ques14# cat pythoncronjob.yml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: python-helloworld
spec:
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: python-helloworld
            image: nilamballal169/cloudethix_cronjob_nilam:latest
            command: ["/app/helloworld.py"]
          restartPolicy: OnFailure

root@Nilam:Ques14#
```

```
root@Nilam:Ques14# kubectl create -f pythoncronjob.yml
cronjob.batch/python-helloworld created
```

● Now create a cron job using pythoncronjob.yml file. Check with kubectl command if the cron job is created.

```
root@Nilam:Ques14# kubectl get cronjobs
NAME                SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
python-helloworld   */1 * * * *   False     1        50s             53s
root@Nilam:Ques14#
```

● Check the Job name which is created by cronjob from command line or lens.

● Then check the pod logs which are created by the job and capture the output.

```
root@Nilam:Ques14# kubectl get jobs
NAME                          COMPLETIONS   DURATION   AGE
python-helloworld-28477932    0/1           108s       108s
python-helloworld-28477933    0/1           48s        48s
root@Nilam:Ques14#
```

- Prepare well formatted documents and write your understanding.

 # vim helloworld.py#!/usr/local/bin/python3
import datetime
x = datetime.datetime.now()
print("Welcome to the Cloudethix World")
print("Today is")
print(x)

 # vim Dockerfile
FROM python:3.7-alpine
#add user group and ass user to that group
RUN addgroup -S appgroup && adduser -S appuser -G appgroup
#creates work dir
WORKDIR /app
#copy python script to the container folder app
COPY helloworld.py /app/helloworld.py
RUN chmod +x /app/helloworld.py
#user is appuser
USER appuser
ENTRYPOINT ["python", "/app/helloworld.py"]
 # vim pythoncronjob.yml
apiVersion: batch/v1
kind: CronJob
metadata:
name: python-helloworld
spec:
schedule: "/1 * * * *""*
jobTemplate:
spec:
template:
spec:
containers:
- name: python-helloworld
image: python-helloworld
command: [/app/helloworld.py]
restartPolicy: OnFailure