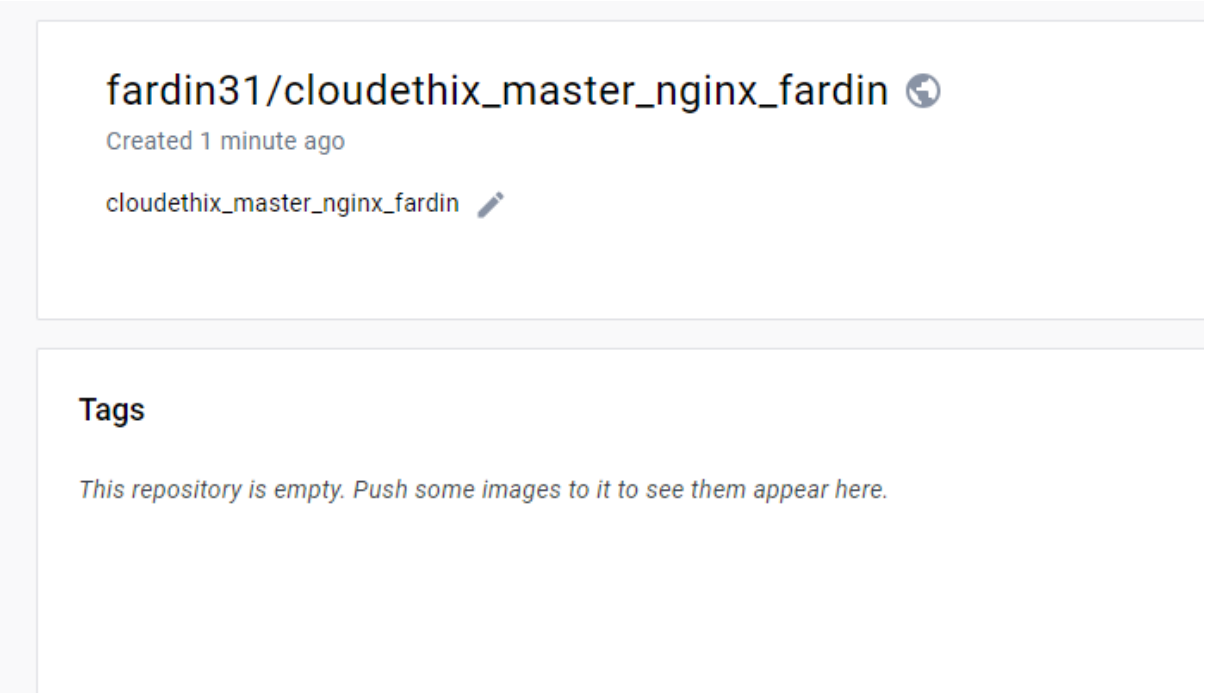
**CLOUDETHIX**

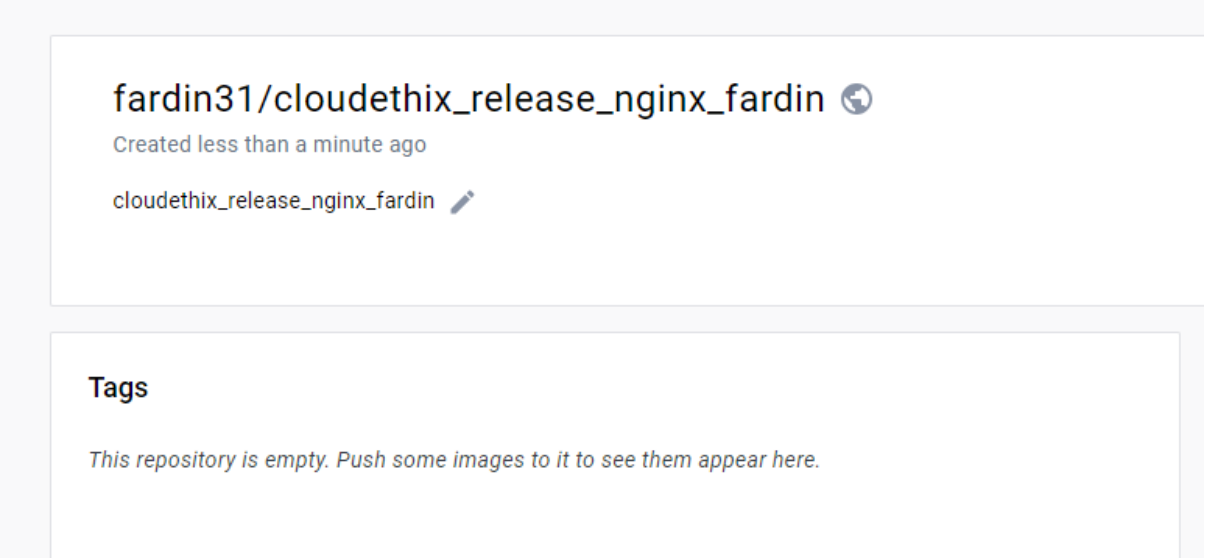
**Que 1 →**

**● Create 2 Public Docker Hub registries named**

**cloudethix_master_nginx_yourname & cloudethix_release_nginx_yourname.**

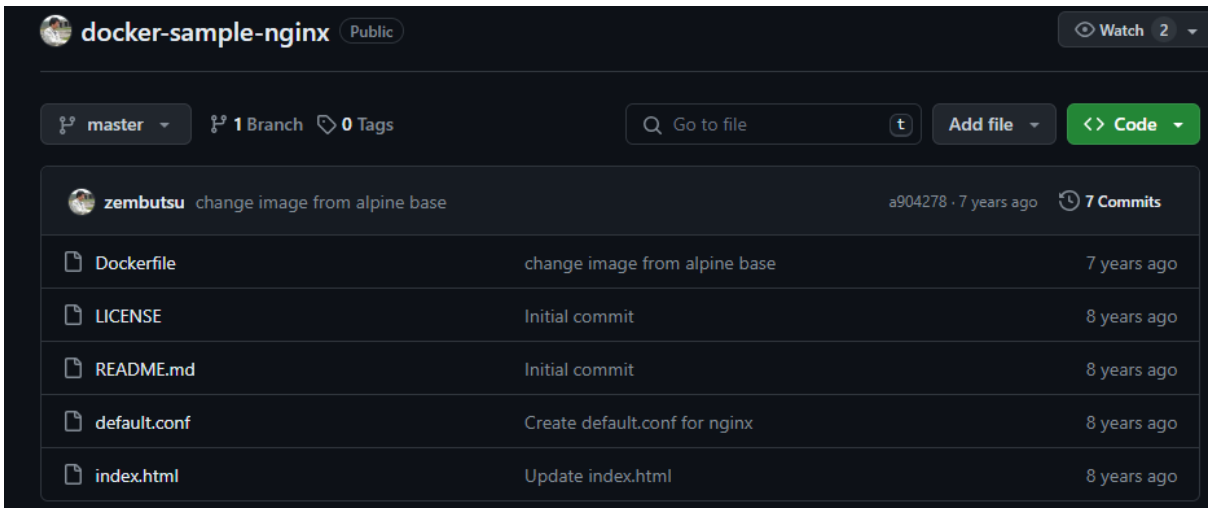
**ANS:-**



● **Clone below repository on your system.**

[https://github.com/zembutsu/docker-sample-nginx.git](https://github.com/zembutsu/docker-sample-nginx.git)




```
root@FARDIN:/mnt/f/k8s_Assignment# git clone git@github.com:zembutsu/docker-sample-nginx.git
Cloning into 'docker-sample-nginx'...
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
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@FARDIN:/mnt/f/k8s_Assignment# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21 12:26 ./
drwxrwxrwx 1 root root 512 Feb 21 12:25 ../
drwxrwxrwx 1 root root 512 Feb 21 12:26 docker-sample-nginx/
root@FARDIN:/mnt/f/k8s_Assignment#
```

● **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@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch release
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch main
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch hotfix
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch -l
  hotfix
  main
* master
  release
```

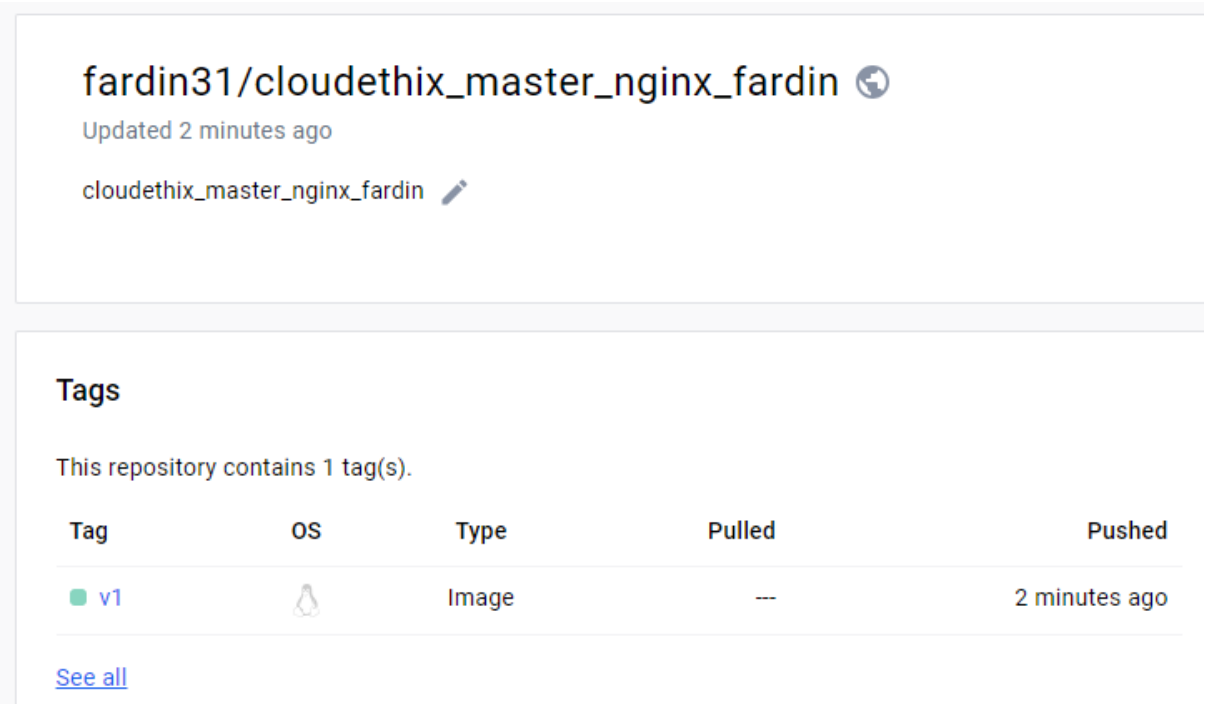
● **Once code is copied to local repository,from master branch update the index.html and add word "Cloudethix Master Branch Nginx" and build the docker image & add meaningful tags and push to Docker Hub registry cloudethix_master_nginx_yourname.**

```html
<html>
<body>
        <h1>Cloudethix Master Branch Nginx</h1>

</body>
</html>
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# docker build -t fardin31/cloudethix_master_nginx_fardin:v1 .
[+] Building 7.8s (9/9) FINISHED                                                                                docker:default
 => [internal] load build definition from Dockerfile                                                                        0.1s
 => => transferring dockerfile: 132B                                                                                        0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                                             2.8s
 => [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       4.4s
 => => resolve docker.io/library/nginx:alpine@sha256:6a2f8b28e45c4adea04ec207a251fd4a2df03ddc930f782af51e315ebc76e9a9       0.0s
 => => sha256:6913ed9ec8d009744018c1740879327fe2e085935b2cce7a234bf05347b670d7 11.74kB / 11.74kB                           0.0s
 => => sha256:619be1103602d98e1963557998c954c892b3872986c27365e9f651f5bc27cab8 3.40MB / 3.40MB                             1.1s
 => => sha256:018b9065ed0dfedff48bbd11f6014960bb496e71c395f772bfad123ab33a1800 1.90MB / 1.90MB                             1.5s
 => => sha256:6a2f8b28e45c4adea04ec207a251fd4a2df03ddc930f782af51e315ebc76e9a9 8.71kB / 8.71kB                             0.0s
 => => sha256:cb0953165f59b5cf2227ae979a49a2284956d997fad4ed7a338eebc6aef3e70b 2.50kB / 2.50kB                             0.0s
 => => sha256:c3ea3344e711fd7111dee02f17deebceb725ed1d0ee998f7fb472114dc1399ce 629B / 629B                                 1.2s
 => => extracting sha256:619be1103602d98e1963557998c954c892b3872986c27365e9f651f5bc27cab8                                  0.1s
 => => sha256:c7059f3102784cd05dc96fff74a52bce9fa50fea724ece08748507fa3455999b 956B / 956B                                 1.5s
 => => sha256:a101c9a82b88a3fa561030af162d98a130ca3bc0501b2e70594410dd426f2c9b 393B / 393B                                 1.7s
 => => extracting sha256:018b9065ed0dfedff48bbd11f6014960bb496e71c395f772bfad123ab33a1800                                  0.2s
 => => sha256:e1c681003a03fff277ecf90fccf526881bcc2e006c9e371b58f45680d54c1954 1.40kB / 1.40kB                             2.1s
 => => sha256:d6a456492aaa4c003389fec3da0939f31c505232fcf1925db314815a196c444f 1.21kB / 1.21kB                             1.9s
 => => sha256:a85ccd8c07bd7090e8a37ab878413b035a370e872367b145a0c0aaaaf60ccbdf 12.65MB / 12.65MB                           3.6s
 => => extracting sha256:c3ea3344e711fd7111dee02f17deebceb725ed1d0ee998f7fb472114dc1399ce                                  0.0s
 => => extracting sha256:c7059f3102784cd05dc96fff74a52bce9fa50fea724ece08748507fa3455999b                                  0.0s
 => => extracting sha256:a101c9a82b88a3fa561030af162d98a130ca3bc0501b2e70594410dd426f2c9b                                  0.0s
 => => extracting sha256:d6a456492aaa4c003389fec3da0939f31c505232fcf1925db314815a196c444f                                  0.0s
 => => extracting sha256:e1c681003a03fff277ecf90fccf526881bcc2e006c9e371b58f45680d54c1954                                  0.0s
 => => extracting sha256:a85ccd8c07bd7090e8a37ab878413b035a370e872367b145a0c0aaaaf60ccbdf                                  0.4s
 => [internal] load build context                                                                                           0.1s
 => => transferring context: 453B                                                                                           0.0s
 => [2/3] COPY default.conf /etc/nginx/conf.d/                                                                              0.2s
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# docker push fardin31/cloudethix_master_nginx_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_master_nginx_fardin]
22103b65cdcf: Pushed
7e89943c594a: 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 library/nginx
718db50a47c0: Mounted from library/nginx
aedc3bda2944: Mounted from library/nginx
v1: digest: sha256:149ebd9a8116a76137fe1a4bddc2ab900840db4edf5f762a2233dec9815c094a size: 2403
```




- **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@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch -l
  hotfix
  main
  master
* release
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# cat index.html
<html>
<body>
        <h1>Cloudethix Release Branch Nginx</h1>

</body>
</html>
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# docker build -t fardin31/cloudethix_release_nginx_fardin:v1 .
[+] Building 2.0s (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                                                               1.7s
 => [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: 161B                                                                                             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                                                                                                       0.1s
 => => writing image sha256:c36cebb14878d2d62c7668e8c6a9bda61583841bc6c7f42dcc69bbe001a140d9                                  0.0s
 => => naming to docker.io/fardin31/cloudethix_release_nginx_fardin:v1                                                        0.0s
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# docker push fardin31/cloudethix_release_nginx_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_release_nginx_fardin]
f01420174397: Pushed
7e89943c594a: Mounted from fardin31/cloudethix_master_nginx_fardin
667a247707f0: Mounted from fardin31/cloudethix_master_nginx_fardin
d8527026595f: Mounted from fardin31/cloudethix_master_nginx_fardin
2593b08e5428: Mounted from fardin31/cloudethix_master_nginx_fardin
9909978d630d: Mounted from fardin31/cloudethix_master_nginx_fardin
c5140fc719dd: Mounted from fardin31/cloudethix_master_nginx_fardin
3137f8f0c641: Mounted from fardin31/cloudethix_master_nginx_fardin
718db50a47c0: Mounted from fardin31/cloudethix_master_nginx_fardin
aedc3bda2944: Mounted from fardin31/cloudethix_master_nginx_fardin
v1: digest: sha256:564179db1ff0de004dadb345f60fc8b987b0ad118055a3aa55ede518cb974517 size: 2403
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx#
```

fardin31 / Repositories / cloudethix_release_nginx_fardin / General

General Tags Builds Collaborators Webhooks Settings

## fardin31/cloudethix_release_nginx_fardin

Updated 1 minute ago

cloudethix_release_nginx_fardin

### Tags

This repository contains 1 tag(s).

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| v1 | | Image | --- | a minute ago |

See all

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

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git checkout main
M       index.html
Switched to branch 'main'
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git branch -l
  hotfix
* main
  master
  release
root@FARDIN:/mnt/f/k8s_Assignment/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@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# ll
total 4
drwxrwxrwx 1 root root  512 Feb 21 16:05 ./
drwxrwxrwx 1 root root  512 Feb 21 12:26 ../
drwxrwxrwx 1 root root  512 Feb 21 16:09 .git/
-rwxrwxrwx 1 root root   95 Feb 21 12:26 Dockerfile*
-rwxrwxrwx 1 root root 1084 Feb 21 12:26 LICENSE*
-rwxrwxrwx 1 root root   73 Feb 21 12:26 README.md*
-rwxrwxrwx 1 root root  286 Feb 21 12:26 default.conf*
-rwxrwxrwx 1 root root   88 Feb 21 12:47 index.html*
drwxrwxrwx 1 root root  512 Feb 21 16:07 kube/
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# tree kube/
kube/
└── clusterIP
    ├── cluster_ip-service.yaml
    ├── master_pod.yaml
    └── release_pod.yaml

1 directory, 3 files
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# cat master_pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: master-nginx
  labels:
    name: master-nginx
spec:
  containers:
  - name: master-nginx-container
    image: fardin31/cloudethix_master_nginx_fardin:v1
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

● **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@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# cat release_pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: release-nginx
  labels:
    name: release-nginx
spec:
  containers:
  - name: release-nginx
    image: fardin31/cloudethix_release_nginx_fardin:v1
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

● **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.**

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/clusterIP# cat cluster_ip-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: cloudethix-clusterip
spec:
  selector:
    app: release-nginx
  ports:
  - port: 80
    targetPort: 80
```

● **Create all these three resources in your k8s cluster.**

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/clusterIP# kaf .
service/cloudethix-clusterip created
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/clusterIP# kgs
NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
cloudethix-clusterip    ClusterIP   10.106.204.223   <none>        80/TCP    5s
kubernetes              ClusterIP   10.96.0.1        <none>        443/TCP   98m
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# kaf .
pod/master-nginx created
pod/release-nginx created
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# kgp -o wide
NAME            READY   STATUS    RESTARTS   AGE   IP              NODE       NOMINATED NODE   READINESS GATES
master-nginx    1/1     Running   0          10s   10.111.158.65   worker-1   <none>           <none>
release-nginx   1/1     Running   0          9s    10.108.43.4     worker-0   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube#
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# k exec -it master-nginx -- /bin/sh
/ # curl localhost
<html>
<body>
        <h1>Cloudethix Master Branch Nginx</h1>

</body>
</html>
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# k exec -it release-nginx -- /bin/sh
/ # curl localhost
<html>
<body>
        <h1>Cloudethix Release Branch Nginx</h1>

</body>
</html>
```

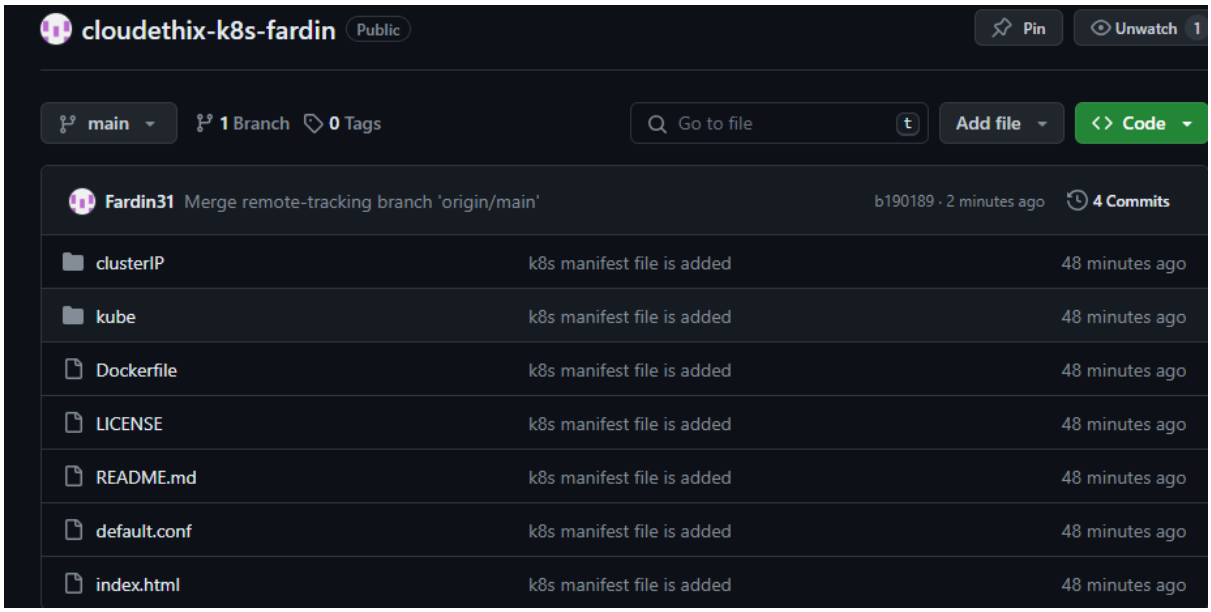
● **Then curl the clusterip service with its name and check the result.**

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/clusterIP# k exec -it release-nginx -- /bin/sh
/ # curl cloudethix-clusterip
<html>
<body>
        <h1>Cloudethix Release Branch Nginx</h1>

</body>
</html>
/ #
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx# git push -u origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (13/13), 2.17 KiB | 11.00 KiB/s, done.
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Fardin31/cloudethix-k8s-fardin.git
   68ab929..b190189  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```




- **Take all screenshots and create a well formatted document.**

**Que 2 →**

**ANS:-**

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

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# mkdir NodePort
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21 16:11 ./
drwxrwxrwx 1 root root 512 Feb 21 16:05 ../
drwxrwxrwx 1 root root 512 Feb 21 16:11 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**

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21 16:15 ./
drwxrwxrwx 1 root root 512 Feb 21 16:11 ../
-rwxrwxrwx 1 root root   0 Feb 21 16:12 backend-deployment.yaml*
-rwxrwxrwx 1 root root   0 Feb 21 16:13 backend-service.yaml*
-rwxrwxrwx 1 root root   0 Feb 21 16:15 frontend-NodePort-service.yaml*
-rwxrwxrwx 1 root root   0 Feb 21 16:14 frontend-deployment.yaml*
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/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
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort# cat backend-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: hello
spec:
  selector:
    app: hello
    tier: backend
  ports:
  - protocol: TCP
    port: 80
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/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
```

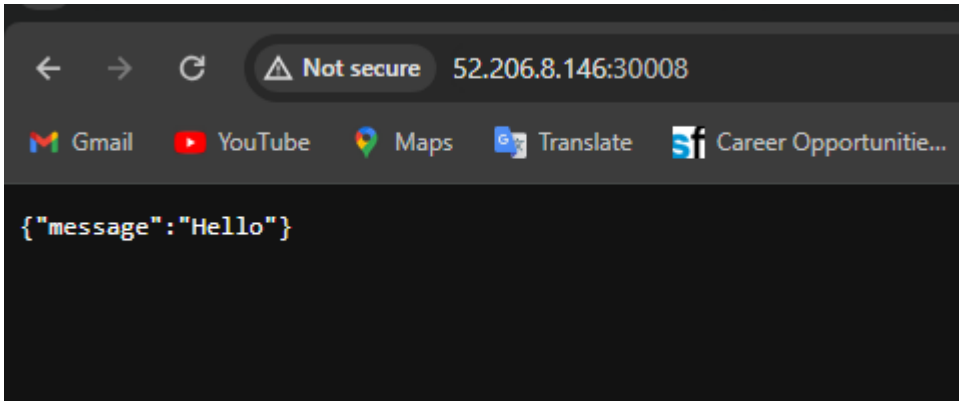
```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort# cat frontend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend
spec:
  selector:
    matchLabels:
      app: hello
      tier: frontend
      track: stable
  replicas: 1
  template:
    metadata:
      labels:
        app: hello
        tier: frontend
        track: stable
    spec:
      containers:
        - name: nginx
          image: "gcr.io/google-samples/hello-frontend:1.0"
          lifecycle:
            preStop:
              exec:
```

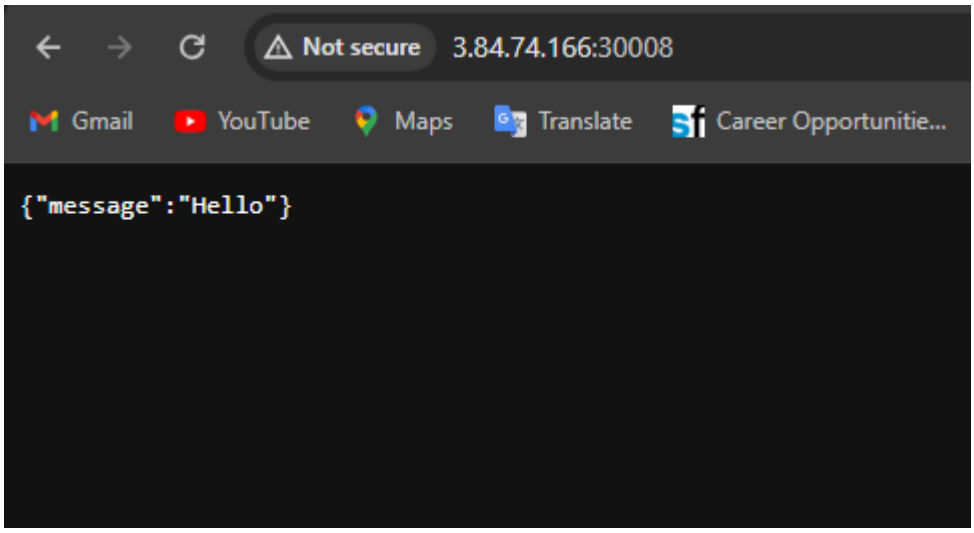
- **Once files are created , create all the resources in your k8s cluster.**

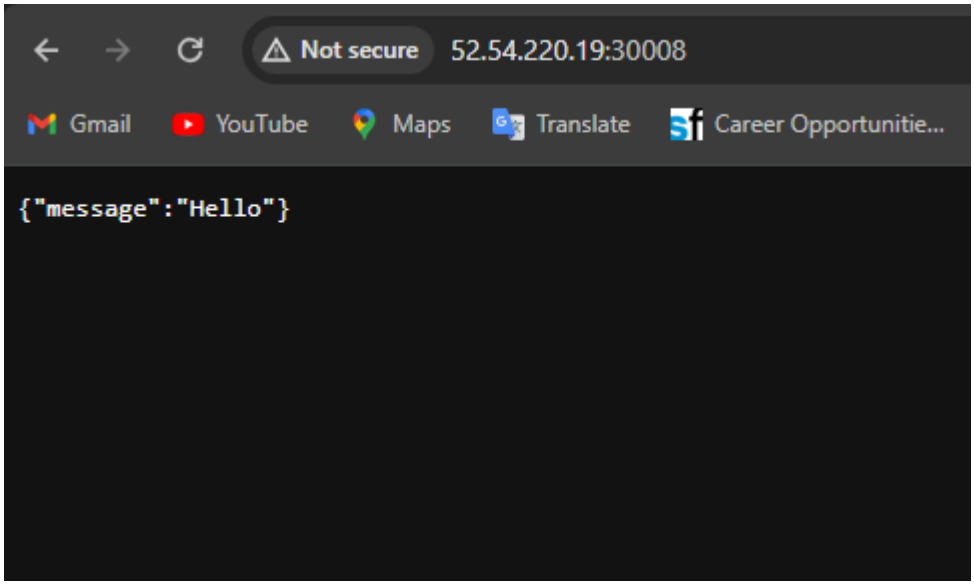
```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort# kgp -o wide
NAME                        READY   STATUS    RESTARTS   AGE     IP              NODE       NOMINATED NODE   READINESS GATES
backend-7f5b7998b9-2nwg4    1/1     Running   0          6m20s   10.111.158.74   worker-1   <none>           <none>
backend-7f5b7998b9-mddg7    1/1     Running   0          6m20s   10.108.43.6     worker-0   <none>           <none>
backend-7f5b7998b9-rngdz    1/1     Running   0          6m20s   10.111.158.73   worker-1   <none>           <none>
frontend-85c84f8b8b-tmgjz   1/1     Running   0          6m19s   10.111.158.75   worker-1   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort# kgs -o wide
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE     SELECTOR
frontend     NodePort    10.108.127.253   <none>        80:30008/TCP   4m30s   app=hello,tier=frontend
hello        ClusterIP   10.99.32.126     <none>        80/TCP         6m50s   app=hello,tier=backend
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP        5h25m   <none>
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube/NodePort#
```

- **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@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube# git commit -m "NodePort Derectory is added"
[main 7d29b9c] NodePort Derectory is added
 4 files changed, 75 insertions(+)
 create mode 100644 kube/NodePort/backend-deployment.yaml
 create mode 100644 kube/NodePort/backend-service.yaml
 create mode 100644 kube/NodePort/frontend-NodePort-service.yaml
 create mode 100644 kube/NodePort/frontend-deployment.yaml
root@FARDIN:/mnt/f/k8s_Assignment/docker-sample-nginx/kube#
```

● **Take all screenshots and create a well formatted document.**

**Que 3 →**

**ANS:-**

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

```
root@FARDIN:/mnt/f/k8s_Assignment/03_pod.yaml# cat  worker-0.yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-nginx
  labels:
    name: my-nginx
spec:
  nodeName: worker-0
  containers:
  - name: my-nginx
    image: nginx
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

```
root@FARDIN:/mnt/f/k8s_Assignment/03_pod.yaml# cat  worker-1.yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-http
  labels:
    name: my-http
spec:
  nodeName: worker-1
  containers:
  - name: my-http
    image: httpd
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

```
root@FARDIN:/mnt/f/k8s_Assignment/03_pod.yaml# kgp -o wide
NAME       READY   STATUS    RESTARTS   AGE   IP              NODE       NOMINATED NODE   READINESS GATES
my-http    1/1     Running   0          96s   10.111.158.76   worker-1   <none>           <none>
my-nginx   1/1     Running   0          97s   10.108.43.7     worker-0   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/03_pod.yaml#
```

**Que 4 →**

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

```
root@FARDIN:/mnt/f/k8s_Assignment# kubectl label nodes worker-0 customName=cloudethix-k8s-00
node/worker-0 labeled
```

```
root@FARDIN:/mnt/f/k8s_Assignment# kubectl label nodes worker-1 customName=cloudethix-k8s-01
node/worker-1 labeled
```

**● 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@FARDIN:/mnt/f/k8s_Assignment/04_pod.yaml# cat pod00.yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-nginx
  labels:
    name: my-nginx
spec:
  containers:
  - name: my-nginx
    image: nginx
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
  nodeSelector:
    cloudethix-k8s-00: "true"
```

```
root@FARDIN:/mnt/f/k8s_Assignment/04_pod.yaml# cat pod01.yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-http
  labels:
    name: my-http
spec:
  containers:
  - name: my-http
    image: httpd
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
  nodeSelector:
    cloudethix-k8s-01: "true"
```

```
root@FARDIN:/mnt/f/k8s_Assignment/04_pod.yaml# kgp -o wide
NAME       READY   STATUS    RESTARTS   AGE     IP              NODE       NOMINATED NODE   READINESS GATES
my-http    1/1     Running   0          2m13s   10.111.158.78   worker-1   <none>           <none>
my-nginx   1/1     Running   0          2m35s   10.108.43.9     worker-0   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/04_pod.yaml#
```

**Que 5 →**

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

**[https://github.com/collabnix/kubelabs](https://github.com/collabnix/kubelabs)**

```
root@FARDIN:/mnt/f/k8s_Assignment# git clone git@github.com:collabnix/kubelabs.git
Cloning into 'kubelabs'...
remote: Enumerating objects: 12313, done.
remote: Counting objects: 100% (1164/1164), done.
remote: Compressing objects: 100% (486/486), done.
remote: Total 12313 (delta 699), reused 1050 (delta 643), pack-reused 11149
Receiving objects: 100% (12313/12313), 61.41 MiB | 1.83 MiB/s, done.
Resolving deltas: 100% (3275/3275), done.
Updating files: 100% (7329/7329), done.
root@FARDIN:/mnt/f/k8s_Assignment# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21 17:47 ./
drwxrwxrwx 1 root root 512 Feb 21 12:25 ../
drwxrwxrwx 1 root root 512 Feb 21 17:22 03_pod.yaml/
drwxrwxrwx 1 root root 512 Feb 21 17:30 04_pod.yaml/
drwxrwxrwx 1 root root 512 Feb 21 16:05 docker-sample-nginx/
drwxrwxrwx 1 root root 512 Feb 21 17:49 kubelabs/
```

```
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# ll
total 8
drwxrwxrwx 1 root root  512 Feb 21 17:49 ./
drwxrwxrwx 1 root root  512 Feb 21 17:49 ../
-rwxrwxrwx 1 root root 7040 Feb 21 17:49 README.md*
-rwxrwxrwx 1 root root  394 Feb 21 17:49 daemonset.yml*
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kaf .
daemonset.apps/prometheus-daemonset created
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kubectl get daemonsets
NAME                  DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
prometheus-daemonset  2         2         2       2            2           <none>          44s
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101#
```

**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@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# ssh ubuntu@52.54.220.19
The authenticity of host '52.54.220.19 (52.54.220.19)' can't be established.
ED25519 key fingerprint is SHA256:4ws58QSmc0vJvz8Azv1BeW1XCuaXHXvVuWUrnhhd44Q.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '52.54.220.19' (ED25519) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1103-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Wed Feb 21 12:49:01 UTC 2024

  System load:  0.08               Users logged in:        0
  Usage of /:   51.6% of 7.57GB    IP address for eth0:    172.31.90.108
  Memory usage: 26%                IP address for docker0: 172.17.0.1
  Swap usage:   0%                 IP address for tunl0:   10.108.43.0
  Processes:    131


Expanded Security Maintenance for Infrastructure is not enabled.

10 updates can be applied immediately.
2 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

89 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04
```

```
root@ip-172-31-90-108:/etc/kubernetes/manifests# cat static-web.yaml
apiVersion: v1
kind: Pod
metadata:
  name: static-web
  labels:
    role: myrole
spec:
  containers:
    - name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
          protocol: TCP
```

```
root@ip-172-31-90-108:/etc/kubernetes/manifests# systemctl restart kubelet
root@ip-172-31-90-108:/etc/kubernetes/manifests# systemctl status  kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
  Drop-In: /etc/systemd/system/kubelet.service.d
           └─10-kubeadm.conf
   Active: active (running) since Wed 2024-02-21 12:56:14 UTC; 19s ago
     Docs: https://kubernetes.io/docs/home/
 Main PID: 2404 (kubelet)
    Tasks: 10 (limit: 2342)
   CGroup: /system.slice/kubelet.service
           └─2404 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=

Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824753    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824783    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824807    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824831    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824877    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.824942    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.825006    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.825034    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.825066    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
Feb 21 12:56:15 ip-172-31-90-108 kubelet[2404]: I0221 12:56:15.825127    2404 reconciler_common.go:258] "operationExecutor.VerifyControllerAttachedV
```

```
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kgp
NAME                          READY   STATUS    RESTARTS   AGE
prometheus-daemonset-j6vl8    1/1     Running   0          24m
prometheus-daemonset-k4cln    1/1     Running   0          24m
static-web-worker-0           1/1     Running   0          5m20s
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# k delete po static-web-worker-0
pod "static-web-worker-0" deleted
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kgp
NAME                          READY   STATUS    RESTARTS   AGE
prometheus-daemonset-j6vl8    1/1     Running   0          25m
prometheus-daemonset-k4cln    1/1     Running   0          25m
static-web-worker-0           1/1     Running   0          5s
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101#
```

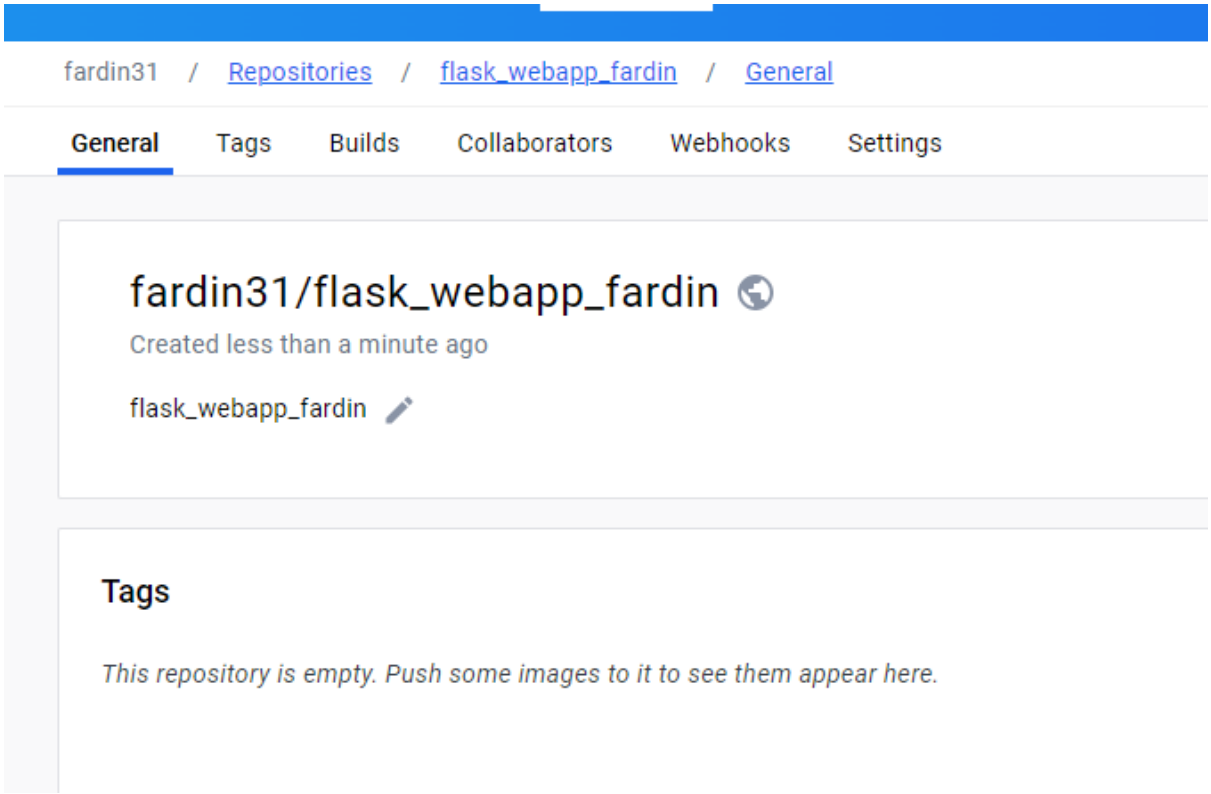
**Que 7 →**

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

```
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# sudo git clone https://github.com/ahmetb/kubectx /opt/kubectx
Cloning into '/opt/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 | 4.11 MiB/s, done.
Resolving deltas: 100% (876/876), done.
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# sudo ln -s /opt/kubectx/kubectx /usr/local/bin/kubectx
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# sudo ln -s /opt/kubectx/kubens /usr/local/bin/kubens
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kubens
default
kube-node-lease
kube-public
kube-system
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101# kubectx
kubernetes-admin@kubernetes
root@FARDIN:/mnt/f/k8s_Assignment/kubelabs/DaemonSet101#
```

**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](https://github.com/mmumshad/simple-webapp-docker.git)

```
root@FARDIN:/mnt/f/k8s_Assignment# git clone git@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.
```

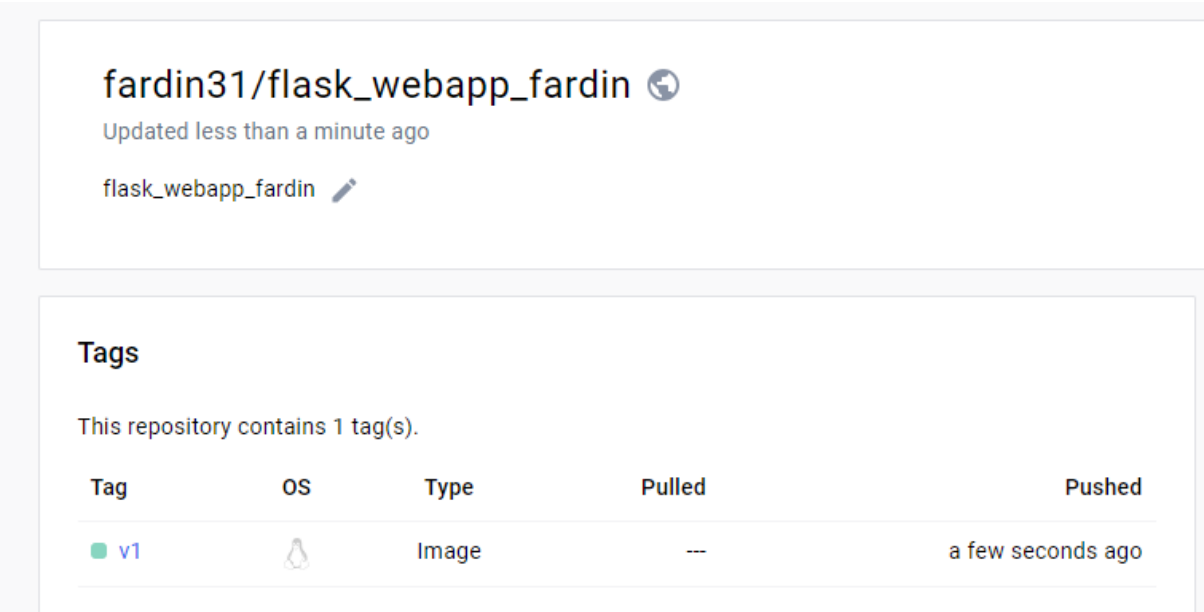
- **Initialize a local repository & copy the code from above repo to your local repository in your working branch.**

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21  2024 ./
drwxrwxrwx 1 root root 512 Feb 21  2024 ../
drwxrwxrwx 1 root root 512 Feb 21  2024 .git/
-rwxrwxrwx 1 root root 194 Feb 21  2024 Dockerfile*
-rwxrwxrwx 1 root root 229 Feb 21  2024 app.py*
```

- **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@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# docker build -t fardin31/flask_webapp_fardin:v1 .
[+] Building 113.5s (10/10) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                            0.1s
 => => transferring dockerfile: 233B                                                                            0.0s
 => [internal] load metadata for docker.io/library/ubuntu:20.04                                                36.3s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                   0.0s
 => [internal] load .dockerignore                                                                               0.1s
 => => transferring context: 2B                                                                                 0.0s
 => [1/4] FROM docker.io/library/ubuntu:20.04@sha256:bb1c41682308d7040f74d103022816d41c50d7b0c89e9d706a74b4e548636e54  4.6s
 => => resolve docker.io/library/ubuntu:20.04@sha256:bb1c41682308d7040f74d103022816d41c50d7b0c89e9d706a74b4e548636e54  0.0s
 => => sha256:bb1c41682308d7040f74d103022816d41c50d7b0c89e9d706a74b4e548636e54 1.13kB / 1.13kB                  0.0s
 => => sha256:a4fab1802f08df089c4b2e0a1c8f1a06f573bd1775687d07fef4076d3a2e4900 424B / 424B                      0.0s
 => => sha256:18ca3f4297e795532c0d053ba443d392d5d316ee83ddee0de27f1e742a7db273 2.30kB / 2.30kB                  0.0s
 => => sha256:8ee0874247356ecb5ea92128219660506b139dcb6cc45dcab84d98b3c6485061 27.51MB / 27.51MB                3.0s
 => => extracting sha256:8ee0874247356ecb5ea92128219660506b139dcb6cc45dcab84d98b3c6485061                        1.2s
 => [internal] load build context                                                                               0.1s
 => => transferring context: 264B                                                                               0.0s
 => [2/4] RUN apt-get update && apt-get install -y python3 python3-pip                                         66.4s
 => [3/4] RUN pip install flask                                                                                 4.2s
 => [4/4] COPY app.py /opt/                                                                                     0.1s
 => exporting to image                                                                                          1.7s
 => => exporting layers                                                                                         1.7s
 => => writing image sha256:8e9e6957fb9eee548f9b5072e6332e4a4a502061b0f78e1e8ce10c9a1520b8fe                    0.0s
 => => naming to docker.io/fardin31/flask_webapp_fardin:v1                                                      0.0s
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# docker push fardin31/flask_webapp_fardin:v1
The push refers to repository [docker.io/fardin31/flask_webapp_fardin]
9285060951a7: Pushed
a017666f6a77: Pushed
d406adb21506: Pushed
28da0445c449: Mounted from library/ubuntu
v1: digest: sha256:21adab0e39b4af1d6e79b4b20b20ae6c4fad8aaa0889fcb05c559694f5453d83 size: 1160
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker#
```



● **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@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# mkdir kube
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# ll
total 0
drwxrwxrwx 1 root root 512 Feb 21  2024 ./
drwxrwxrwx 1 root root 512 Feb 21 19:04 ../
drwxrwxrwx 1 root root 512 Feb 21  2024 .git/
-rwxrwxrwx 1 root root 194 Feb 21 19:04 Dockerfile*
-rwxrwxrwx 1 root root 229 Feb 21 19:04 app.py*
drwxrwxrwx 1 root root 512 Feb 21  2024 kube/
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# cat deployement.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-webapp
spec:
  selector:
    matchLabels:
      app: flask-webapp
  replicas: 3
  template:
    metadata:
      labels:
        app: flask-webapp
    spec:
      containers:
      - name: flask-webapp
        image: fardin31/flask_webapp_fardin:v1
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 8080
```

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

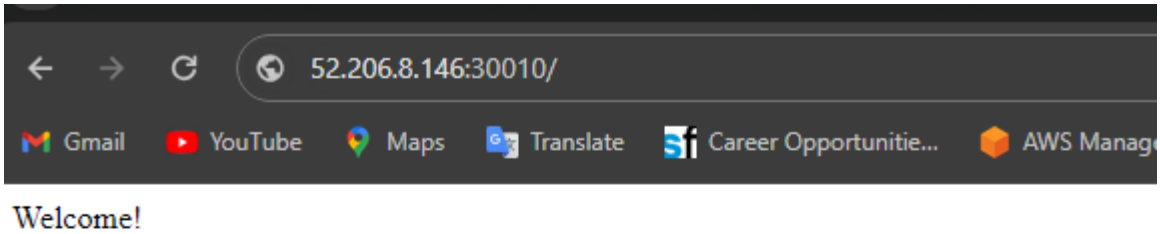
```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: flask-webapp-service
spec:
  selector:
    app: flask-webapp
  ports:
  - port: 8080
    targetPort: 8080
    nodePort: 30010
  type: NodePort
```

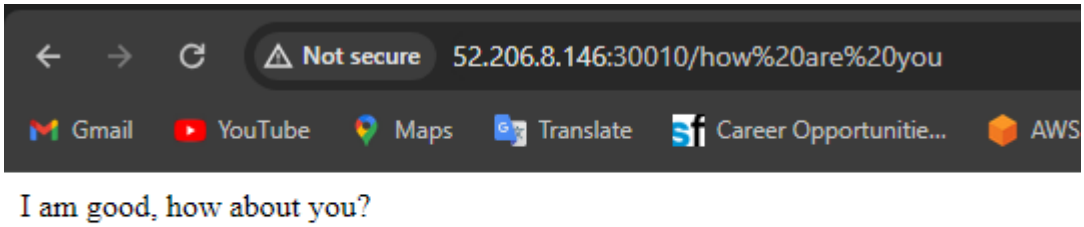
```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# kgp -o wide
NAME                          READY   STATUS    RESTARTS   AGE   IP               NODE       NOMINATED NODE   READINESS GATES
flask-webapp-7658bbd7b-6jtgb  1/1     Running   0          27s   10.111.158.81    worker-1   <none>           <none>
flask-webapp-7658bbd7b-jhv9z  1/1     Running   0          27s   10.108.43.13     worker-0   <none>           <none>
flask-webapp-7658bbd7b-z5h2b  1/1     Running   0          27s   10.111.158.82    worker-1   <none>           <none>
static-web-worker-0           1/1     Running   0          70m   10.108.43.11     worker-0   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# kgs
NAME                  TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
flask-webapp-service  NodePort    10.99.48.242   <none>        8080:30010/TCP   34s
kubernetes            ClusterIP   10.96.0.1      <none>        443/TCP          8h
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/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/**

**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 !!!'**

```python
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()
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# docker build -t fardin31/flask_webapp_fardin:v2 .
[+] Building 2.3s (10/10) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                          0.1s
 => => transferring dockerfile: 233B                                                                                          0.0s
 => [internal] load metadata for docker.io/library/ubuntu:20.04                                                               1.9s
 => [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:bb1c41682308d7040f74d103022816d41c50d7b0c89e9d706a74b4e548636e54          0.0s
 => [internal] load build context                                                                                             0.1s
 => => transferring context: 274B                                                                                             0.1s
 => 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
 => [4/4] COPY app.py /opt/                                                                                                   0.1s
 => exporting to image                                                                                                        0.0s
 => => exporting layers                                                                                                       0.0s
 => => writing image sha256:195230efe75874c1e9a174e8959b88371056990ee65d0ad5aaaafb21f463d940                                  0.0s
 => => naming to docker.io/fardin31/flask_webapp_fardin:v2                                                                    0.0s
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# docker push fardin31/flask_webapp_fardin:v2
The push refers to repository [docker.io/fardin31/flask_webapp_fardin]
986713e50806: Pushed
a017666f6a77: Layer already exists
d406adb21506: Layer already exists
28da0445c449: Layer already exists
v2: digest: sha256:8a67894458ec3f9d2e1f7a144b3fb05b2c86a0171bcd1eb32cfdbf98ee3145c8 size: 1160
root@FARDIN:/mnt/f/k8s_Assignment/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**

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# kubectl set image deployment/flask-webapp  flask-webapp=fardin31/flask_w
ebapp_fardin:v1 --record
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/flask-webapp image updated
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# kubectl set image deployment/flask-webapp  flask-webapp=fardin31/flask_w
ebapp_fardin:v2 --record
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/flask-webapp image updated
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube#
```

2. **With kubectl edit deployment**

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# kubectl rollout undo deployment flask-webapp
deployment.apps/flask-webapp rolled back
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# k edit deployment flask-webapp
deployment.apps/flask-webapp edited
```

```yaml
spec:
  progressDeadlineSeconds: 600
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: flask-webapp
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: flask-webapp
    spec:
      containers:
      - image: fardin31/flask_webapp_fardin:v2
        imagePullPolicy: IfNotPresent
        name: flask-webapp
        ports:
        - containerPort: 8080
          protocol: TCP
        resources: {}
```

3. **With deployment.yaml file modification.**

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-webapp
spec:
  selector:
    matchLabels:
      app: flask-webapp
  replicas: 3
  template:
    metadata:
      labels:
        app: flask-webapp
    spec:
      containers:
      - name: flask-webapp
        image: fardin31/flask_webapp_fardin:v2
        ports:
        - containerPort: 8080
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube#
```

● **Run the # kubectl rollout command to check status and history.**

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube# k rollout history  deployment  flask-webapp
deployment.apps/flask-webapp
REVISION  CHANGE-CAUSE
3         kubectl set image deployment/flask-webapp flask-webapp=fardin31/flask_webapp_fardin:v2 --record=true
4         kubectl set image deployment/flask-webapp flask-webapp=fardin31/flask_webapp_fardin:v1 --record=true

root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker/kube#
```

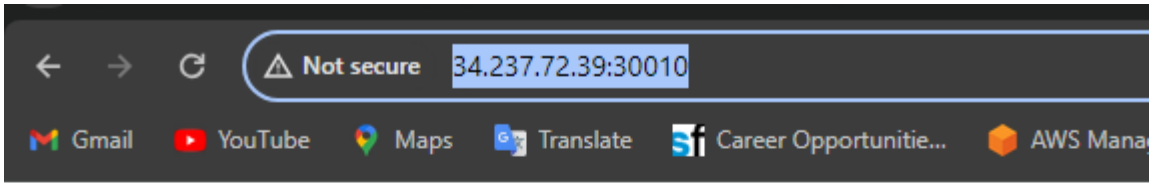
● **Note:- Once above step 1 is done , run # kubectl rollout undo deployment command to rollback the change and then try a second way of rollout.**

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

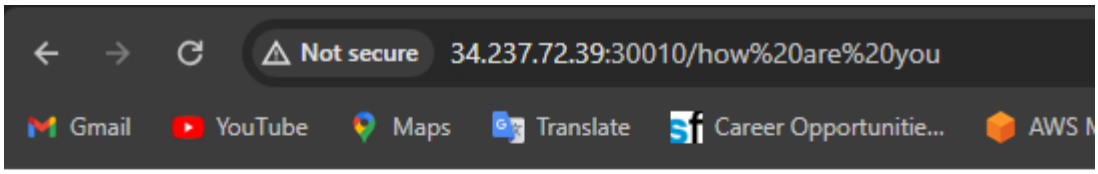
**http://master_ip:30011/**

**http://master_ip:30011/how are you**

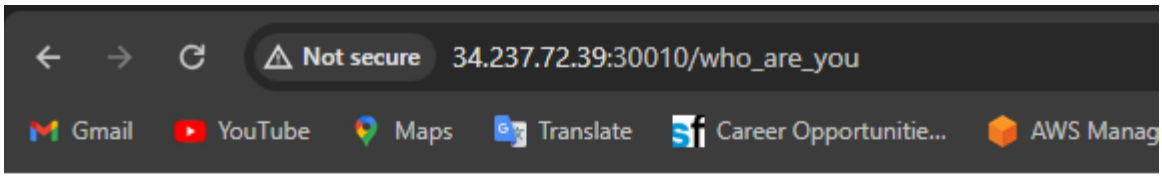
**http://master_ip:30011/Who are you**




Welcome!




I am good, how about you?




Yes, I am cloudethix, and You !!!

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

```
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker# git push origin release
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 1.04 KiB | 10.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'release' on GitHub by visiting:
remote:      https://github.com/Fardin31/cloudethix-k8s-fardin/pull/new/release
remote:
To github.com:Fardin31/cloudethix-k8s-fardin.git
 * [new branch]      release -> release
root@FARDIN:/mnt/f/k8s_Assignment/simple-webapp-docker#
```

● **Capture the snap and prepare a well formatted document.**

**Que 9 →**

● **Download and install Lens & access your k8s cluster from Lens.**

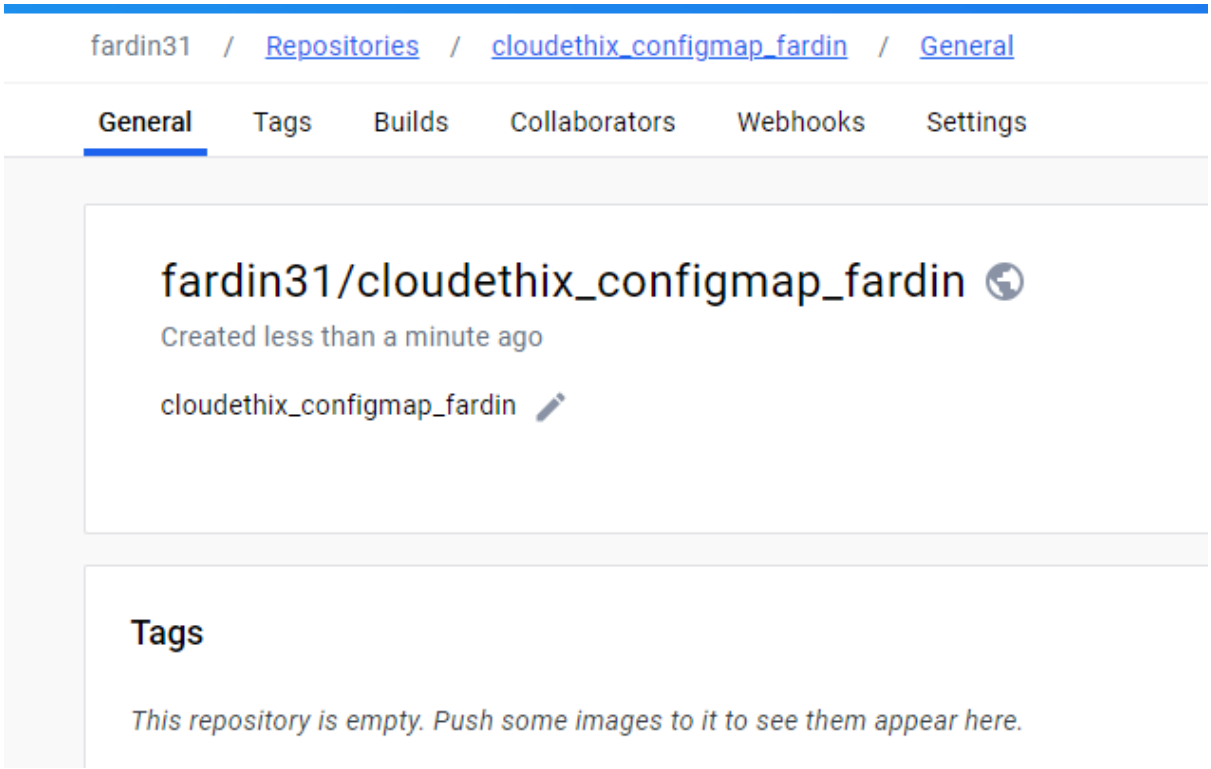
● **Create nginx Pod and Nodeport service. Check the Pod logs from Lens.**

● **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.**

**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**

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

```
root@FARDIN:/mnt/f/k8s_task# git clone git@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@FARDIN:/mnt/f/k8s_task# ll
total 0
drwxrwxrwx 1 root root 512 Feb 22 11:51 ./
drwxrwxrwx 1 root root 512 Feb 22 11:51 ../
drwxrwxrwx 1 root root 512 Feb 22 11:51 docker-sample-nginx/
root@FARDIN:/mnt/f/k8s_task# cd docker-sample-nginx/
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx# ll
total 4
drwxrwxrwx 1 root root  512 Feb 22 11:51 ./
drwxrwxrwx 1 root root  512 Feb 22 11:51 ../
drwxrwxrwx 1 root root  512 Feb 22 11:51 .git/
-rwxrwxrwx 1 root root   95 Feb 22 11:51 Dockerfile*
-rwxrwxrwx 1 root root 1084 Feb 22 11:51 LICENSE*
-rwxrwxrwx 1 root root   73 Feb 22 11:51 README.md*
-rwxrwxrwx 1 root root  286 Feb 22 11:51 default.conf*
-rwxrwxrwx 1 root root  103 Feb 22 11:51 index.html*
```

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

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx# docker build  -t  fardin31/cloudethix_configmap_fardin:v1 .
[+] Building 2.4s (9/9) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 132B
 => [internal] load metadata for docker.io/library/nginx:alpine
 => [auth] library/nginx:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build context
 => => transferring context: 469B
 => [1/3] FROM docker.io/library/nginx:alpine@sha256:6a2f8b28e45c4adea04ec207a251fd4a2df03ddc930f782af51e315ebc76e9a9
 => CACHED [2/3] COPY default.conf /etc/nginx/conf.d/
 => [3/3] COPY index.html /usr/share/nginx/html/
 => exporting to image
 => => exporting layers
 => => writing image sha256:4990f65d8042de32be1b614a6eeb02960b85ff8318cd1b7b044d425a07c7c3ba
 => => naming to docker.io/fardin31/cloudethix_configmap_fardin:v1
```

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx# docker push fardin31/cloudethix_configmap_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_configmap_fardin]
bcce38d52a67: Pushed
7e89943c594a: Mounted from fardin31/cloudethix_release_nginx_fardin
667a247707f0: Mounted from fardin31/cloudethix_release_nginx_fardin
d8527026595f: Mounted from fardin31/cloudethix_release_nginx_fardin
2593b08e5428: Mounted from fardin31/cloudethix_release_nginx_fardin
9909978d630d: Mounted from fardin31/cloudethix_release_nginx_fardin
c5140fc719dd: Mounted from fardin31/cloudethix_release_nginx_fardin
3137f8f0c641: Mounted from fardin31/cloudethix_release_nginx_fardin
718db50a47c0: Mounted from fardin31/cloudethix_release_nginx_fardin
aedc3bda2944: Mounted from fardin31/cloudethix_release_nginx_fardin
v1: digest: sha256:89ee7f0497f92bc6a37b5ec4a0675760eed48ee078967e9b360c357dac647cf7 size: 2403
```

**● 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: frontend-webapp , containerPort: 80 and add the**

**image that you have pushed in Docker Hub registry.**

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx# ll
total 4
drwxrwxrwx 1 root root  512 Feb 22 11:56 ./
drwxrwxrwx 1 root root  512 Feb 22 11:51 ../
drwxrwxrwx 1 root root  512 Feb 22 11:52 .git/
-rwxrwxrwx 1 root root   95 Feb 22 11:51 Dockerfile*
-rwxrwxrwx 1 root root 1084 Feb 22 11:51 LICENSE*
-rwxrwxrwx 1 root root   73 Feb 22 11:51 README.md*
-rwxrwxrwx 1 root root  286 Feb 22 11:51 default.conf*
-rwxrwxrwx 1 root root  103 Feb 22 11:51 index.html*
drwxrwxrwx 1 root root  512 Feb 22 11:56 kube/
```

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend-webapp
spec:
  selector:
    matchLabels:
      app: frontend-webapp
  replicas: 3
  template:
    metadata:
      labels:
        app: frontend-webapp
    spec:
      containers:
      - name: frontend-webapp
        image: fardin31/cloudethix_configmap_fardin:v1
        ports:
        - containerPort: 80
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube#
```

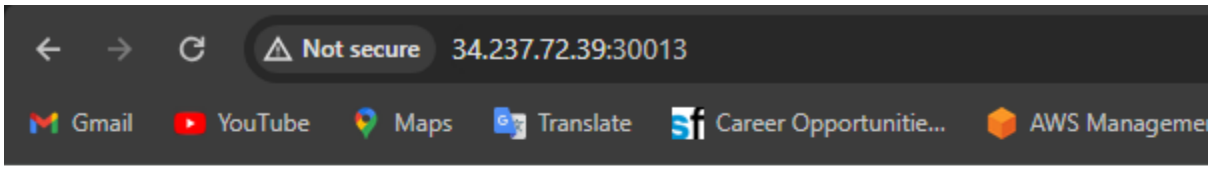
**● Create one service.yaml file with type nodeport & select frontend-webapp pod with port 80 & targetPort 80 with any nodePort between range 30000-32768.**

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: frontend-webapp
  ports:
  - port: 80
    targetPort: 80
    NodePort : 30013
  type: NodePort
```

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

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
frontend-webapp-78b8bf5df-48rxb     1/1     Running   0          94s
frontend-webapp-78b8bf5df-6xfj8     1/1     Running   0          94s
frontend-webapp-78b8bf5df-zl846     1/1     Running   0          94s
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# ll
```

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# kgs
NAME               TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
frontend-service   NodePort    10.96.45.244   <none>        80:30013/TCP   5s
kubernetes         ClusterIP   10.96.0.1      <none>        443/TCP        94m
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube#
```




# Host: frontend-webapp-78b8bf5df-6xfj8

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@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# cat configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-data
data:
  index.html: |
    <html>
    <body>
    <h1> I am Cloudethix Team, Are you ?!! </h1>
    Version: 1.1
    </body>
    </html>
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube#
```

● **Then update the same deployment.yaml file and mount configmap as**

**volume on container using volumeMounts with mountPath**

**/usr/share/nginx/html/**

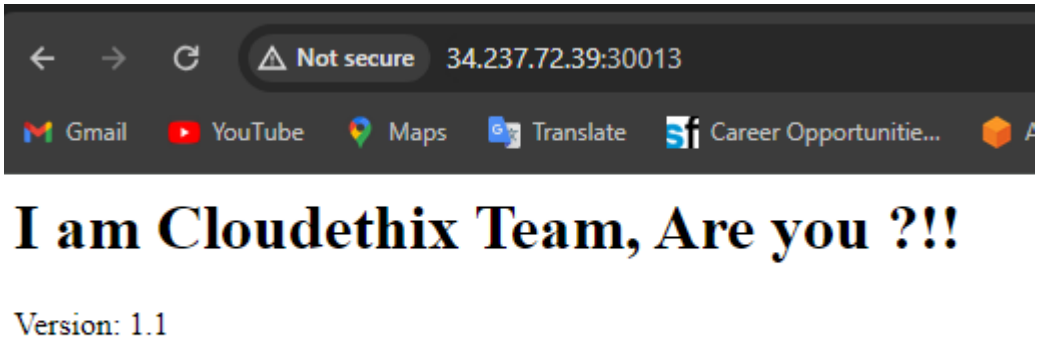
```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend-webapp
spec:
  selector:
    matchLabels:
      app: frontend-webapp
  replicas: 3
  template:
    metadata:
      labels:
        app: frontend-webapp
    spec:
      containers:
      - name: frontend-webapp
        image: fardin31/cloudethix_configmap_fardin:v1
        ports:
        - containerPort: 80
        volumeMounts:
        - name: config-volume
          mountPath: /usr/share/nginx/html/
      volumes:
      - name: config-volume
        configMap:
          name: my-data
```

● **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.**

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# kaf .
configmap/my-data created
deployment.apps/frontend-webapp created
service/frontend-service created
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
frontend-webapp-755b55d485-6x98h    1/1     Running   0          5s
frontend-webapp-755b55d485-rw8jc    1/1     Running   0          5s
frontend-webapp-755b55d485-z24sf    1/1     Running   0          5s
```

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# k get cm
NAME                DATA   AGE
kube-root-ca.crt    1      106m
my-data             1      47s
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube#
```

```
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube# kgs
NAME                TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
frontend-service    NodePort    10.96.157.55   <none>        80:30013/TCP   73s
kubernetes          ClusterIP   10.96.0.1      <none>        443/TCP        106m
root@FARDIN:/mnt/f/k8s_task/docker-sample-nginx/kube#
```




**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](https://github.com/janakiramm/Kubernetes-multi-container-pod.git)

```
root@FARDIN:/mnt/f/k8s_Assignment# 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 | 1.40 MiB/s, done.
Resolving deltas: 100% (21/21), done.
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod# git branch -l
* master
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod# ll
total 120
drwxrwxrwx 1 root root    512 Feb 22 12:47 ./
drwxrwxrwx 1 root root    512 Feb 22 12:45 ../
drwxrwxrwx 1 root root    512 Feb 22 12:48 .git/
-rwxrwxrwx 1 root root      9 Feb 22 12:45 .gitignore*
drwxrwxrwx 1 root root    512 Feb 22 12:45 Build/
drwxrwxrwx 1 root root    512 Feb 22 12:45 Deploy/
-rwxrwxrwx 1 root root   2550 Feb 22 12:45 README.md*
-rwxrwxrwx 1 root root 116003 Feb 22 12:45 multi-container-pod.png*
root@FARDIN:/mnt/f/k8s_Assignment/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@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Build# docker build  -t  fardin31/cloudethix_multicontainer_fardin:v1 .
[+] Building 78.0s (10/10) FINISHED                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                                0.1s
 => => transferring dockerfile: 99B                                                                                                 0.0s
 => [internal] load metadata for docker.io/library/python:2.7-onbuild                                                              36.0s
 => [auth] library/python:pull token for registry-1.docker.io                                                                       0.0s
 => [internal] load .dockerignore                                                                                                   0.0s
 => => transferring context: 2B                                                                                                     0.0s
 => [1/1] FROM docker.io/library/python:2.7-onbuild@sha256:5af88e1d011bf7e845e83813712d9f91be1a39e2ede092008fc53e0a0ce1333b        34.4s
 => => resolve docker.io/library/python:2.7-onbuild@sha256:5af88e1d011bf7e845e83813712d9f91be1a39e2ede092008fc53e0a0ce1333b         0.1s
 => => sha256:3f246dd60a17017c0e05a8d8e358bac601370ed9ea82b1e2fadb8f88b4bbdd12 8.12kB / 8.12kB                                      0.0s
 => => sha256:d660b1f15b9bfb8142f50b518156f2d364d9642fe05854538b060498e2f7928d 54.25MB / 54.25MB                                   10.1s
 => => sha256:46dde23c37b3419122bb597461c1a48bdea1842aaae7dbe728dfa20a9aabe11b 17.54MB / 17.54MB                                    7.5s
 => => sha256:5af88e1d011bf7e845e83813712d9f91be1a39e2ede092008fc53e0a0ce1333b 2.37kB / 2.37kB                                      0.0s
 => => sha256:204402305218 3aff91c02c9437f0b1b3d527269807d00d6aef79f00d1f5e3dd9 2.22kB / 2.22kB                                      0.0s
 => => sha256:6ebaeb0745895220f609d4aa703e4563c39de239a2d00b85bece23a3ca3ac735 43.30MB / 43.30MB                                    17.5s
 => => sha256:e7428f935583e84197ae834885e62b69922f1ce7e8672a3746295555b3853fc7 131.10MB / 131.10MB                                  26.8s
 => => extracting sha256:d660b1f15b9bfb8142f50b518156f2d364d9642fe05854538b060498e2f7928d                                           4.4s
 => => sha256:0c3de61682aa7de56035b5e9a9f3c37dfc01736dcc728369502d0c57940fc7b9 5.75MB / 5.75MB                                     12.2s
 => => sha256:56f10ddf117374aecfff5440019a5f90d8c774aa561ea25b4d3bb288a6e2ed34 14.59MB / 14.59MB                                    16.9s
 => => extracting sha256:46dde23c37b3419122bb597461c1a48bdea1842aaae7dbe728dfa20a9aabe11b                                           0.7s
 => => sha256:4473537c621dafa8c2f6fedf1c3735b9587755d22e4beacca0a68fd9f20f7ec3 1.78MB / 1.78MB                                      18.1s
 => => extracting sha256:6ebaeb0745895220f609d4aa703e4563c39de239a2d00b85bece23a3ca3ac735                                           2.7s
 => => sha256:3106f7df3d1c759cc33c5d6c1aaf84c2c43913a748c801bf42f0154e6166a512 3.66MB / 3.66MB                                      19.8s
 => => sha256:3de1c6ceef68d9f0dc989175d5e3735bc22846f22b31a211db5bd6009885f38a 130B / 130B                                          18.4s
 => => extracting sha256:e7428f935583e84197ae834885e62b69922f1ce7e8672a3746295555b3853fc7                                           5.3s
 => => extracting sha256:0c3de61682aa7de56035b5e9a9f3c37dfc01736dcc728369502d0c57940fc7b9                                           0.3s
 => => extracting sha256:56f10ddf117374aecfff5440019a5f90d8c774aa561ea25b4d3bb288a6e2ed34                                           0.6s
 => => extracting sha256:4473537c621dafa8c2f6fedf1c3735b9587755d22e4beacca0a68fd9f20f7ec3                                           0.3s
 => => extracting sha256:3106f7df3d1c759cc33c5d6c1aaf84c2c43913a748c801bf42f0154e6166a512                                           0.3s
 => => extracting sha256:3de1c6ceef68d9f0dc989175d5e3735bc22846f22b31a211db5bd6009885f38a                                           0.0s
 => [internal] load build context                                                                                                   0.1s
 => => transferring context: 2.10kB                                                                                                 0.0s
 => [2/1] COPY requirements.txt /usr/src/app/                                                                                       0.6s
 => [3/1] RUN pip install --no-cache-dir -r requirements.txt                                                                        6.5s
```

● **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@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/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: fardin31/cloudethix_multicontainer_fardin:v1
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          name: http
          protocol: TCP                    root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Dep
```

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

**Apply the file.**

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/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
```

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

**apply the file.**

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/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"
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/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
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# kgp
NAME    READY   STATUS    RESTARTS   AGE
mysql   1/1     Running   0          84s
web1    2/2     Running   0          54s
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# kgs
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        170m
mysql        ClusterIP   10.98.28.227    <none>        3306/TCP       72s
web          NodePort    10.100.159.91   <none>        80:31962/TCP   8s
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy#
```

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

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

**Initialize the database with sample schema**

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# export NODE_IP=34.237.72.39
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# export NODE_PORT=31962
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# kgs
```

● **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@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl http://$NODE_IP:$NODE_PORT/init
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"John Doe"}' http
://$NODE_IP:$NODE_PORT/users/addRT/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 08:05:48 GMT
```

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "2", "user":"Jane Doe"}' http
://$NODE_IP:$NODE_PORT/users/adds/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 08:06:05 GMT

root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "3", "user":"Bill Collins"}'
http://$NODE_IP:$NODE_PORT/users/adds/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 08:06:19 GMT

root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "4", "user":"Mike Taylor"}' h
ttp://$NODE_IP:$NODE_PORT/users/adds/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 08:06:29 GMT
```

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

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

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -w "\n" http://$NODE_IP:$NODE_PORT/users/1
John Doe
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy#
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy# curl -w "\n" http://$NODE_IP:$NODE_PORT/users/1
John Doe(c)
root@FARDIN:/mnt/f/k8s_Assignment/Kubernetes-multi-container-pod/Deploy#
```

● **Also, try to access mysql shell i.e db pod & run select * from the users table.**

**check app.py for DB related information.**

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| USERDB             |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

```
mysql> USE USERDB;
Database changed
mysql> SHOW TABLES;
+------------------+
| Tables_in_USERDB |
+------------------+
| users            |
+------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT * FROM users;
+------+--------------+
| ID   | USER         |
+------+--------------+
|    1 | John Doe     |
|    2 | Jane Doe     |
|    3 | Bill Collins |
|    4 | Mike Taylor  |
+------+--------------+
4 rows in set (0.00 sec)

mysql>
```

● **Prepare proper documentation in brief & write start to end flow. Refer below**

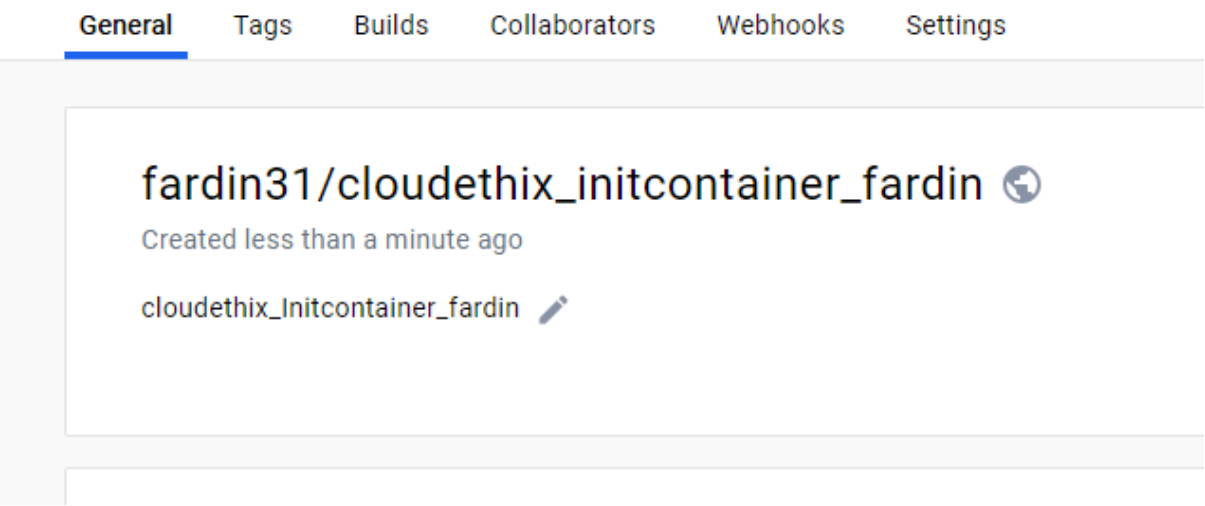
**link if you face any issues.**

**https://github.com/janakiramm/Kubernetes-multi-container-pod**

**Que 12 →**

● **Create 1 Public Docker Hub registry named**

**cloudethix_Initcontainer_yourname.**




● **Clone below repository on your system.**

[https://github.com/janakiramm/simpleapp.git](https://github.com/janakiramm/simpleapp.git)

```
root@FARDIN:/mnt/f/k8s_Assignment# 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 | 137.00 KiB/s, done.
Resolving deltas: 100% (9/9), done.
root@FARDIN:/mnt/f/k8s_Assignment#
```

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

● **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@FARDIN:/mnt/f/k8s_Assignment/simpleapp# docker build  -t  fardin31/cloudethix_initcontainer_fardin:v1 .
[+] Building 23.6s (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                                                              14.0s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                                  0.0s
 => [internal] load .dockerignore                                                                                             0.1s
 => => transferring context: 2B                                                                                               0.0s
 => [1/3] FROM docker.io/library/nginx:latest@sha256:c26ae7472d624ba1fafd296e73cecc4f93f853088e6a9c13c0d52f6ca5865107         8.7s
 => => resolve docker.io/library/nginx:latest@sha256:c26ae7472d624ba1fafd296e73cecc4f93f853088e6a9c13c0d52f6ca5865107         0.1s
 => => sha256:05aa73005987caaed48ea8213696b0df761ccd600d2c53fc0a1a97a180301d71 2.29kB / 2.29kB                                0.0s
 => => sha256:e1caac4eb9d2ec24aa3618e5992208321a92492aef5fef5eb9e470895f771c56 29.12MB / 29.12MB                              5.2s
 => => sha256:c26ae7472d624ba1fafd296e73cecc4f93f853088e6a9c13c0d52f6ca5865107 9.85kB / 9.85kB                                0.0s
 => => sha256:e4720093a3c1381245b53a5a51b417963b3c4472d3f47fc301930a4f3b17666a 7.04kB / 7.04kB                                0.0s
 => => sha256:88f6f236f401ac07aa5309d8ade2b0c9d24b9f526bd4e73311bf5c1787cfd49c 41.39MB / 41.39MB                              7.1s
 => => sha256:c3ea3344e711fd7111dee02f17deebceb725ed1d0ee998f7fb472114dc1399ce 629B / 629B                                    0.3s
 => => sha256:cc1bb4345a3a849289cfb3e2471c096f374423ec1ef74766137b9de546498612 957B / 957B                                    1.0s
 => => sha256:da8fa4352481b358fc60d40ee20d92da64124d2cf40511564 0d45980339f47e5 394B / 394B                                  1.4s
 => => sha256:c7f80e9cdab20387cd09e3c47121ef0eb531043cf0aca1a52aab659de3ccb704 1.21kB / 1.21kB                                1.8s
 => => sha256:18a869624cb60aaa916942dc71c22b194a078dcbbb9b8f54d40512eba55f70b8 1.40kB / 1.40kB                                2.2s
 => => extracting sha256:e1caac4eb9d2ec24aa3618e5992208321a92492aef5fef5eb9e470895f771c56                                     1.8s
 => => extracting sha256:88f6f236f401ac07aa5309d8ade2b0c9d24b9f526bd4e73311bf5c1787cfd49c                                     0.9s
 => => extracting sha256:c3ea3344e711fd7111dee02f17deebceb725ed1d0ee998f7fb472114dc1399ce                                     0.0s
 => => extracting sha256:cc1bb4345a3a849289cfb3e2471c096f374423ec1ef74766137b9de546498612                                     0.0s
 => => extracting sha256:da8fa4352481b358fc60d40ee20d92da64124d2cf40511564 0d45980339f47e5                                    0.0s
 => => extracting sha256:c7f80e9cdab20387cd09e3c47121ef0eb531043cf0aca1a52aab659de3ccb704                                     0.0s
 => => extracting sha256:18a869624cb60aaa916942dc71c22b194a078dcbbb9b8f54d40512eba55f70b8                                     0.0s
 => [internal] load build context                                                                                             0.1s
 => => transferring context: 785B                                                                                             0.0s
 => [2/3] COPY wrapper.sh /                                                                                                   0.5s
 => [3/3] COPY html /usr/share/nginx/html                                                                                     0.1s
 => exporting to image                                                                                                        0.1s
 => => exporting layers                                                                                                       0.1s
 => => writing image sha256:ade0d5957aa56703c51e10a02b82ab1571f2469458fcfb93a295ae1233efe523                                  0.0s
 => => naming to docker.io/fardin31/cloudethix_initcontainer_fardin:v1                                                        0.0s
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp# docker push  fardin31/cloudethix_initcontainer_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_initcontainer_fardin]
657e3bc647f2: Pushed
15f958e209b9: Pushed
61a7fb4dabcd: Mounted from library/nginx
bcc6856722b7: Mounted from library/nginx
188d128a188c: Mounted from library/nginx
7d52a4114c36: Mounted from library/nginx
3137f8f0c641: Mounted from fardin31/cloudethix_configmap_fardin
84619992a45b: Mounted from library/nginx
ceb365432eec: Mounted from library/nginx
v1: digest: sha256:ff81e9ca57807604587f3b318452a57e8e979a69656182d451e1f9089d86116a size: 2192
root@FARDIN:/mnt/f/k8s_Assignment/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@FARDIN:/mnt/f/k8s_Assignment/simpleapp# mkdir kube
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp# ll
total 0
drwxrwxrwx 1 root root 512 Feb 22 14:15 ./
drwxrwxrwx 1 root root 512 Feb 22 14:07 ../
drwxrwxrwx 1 root root 512 Feb 22 14:10 .git/
-rwxrwxrwx 1 root root  85 Feb 22 14:07 Dockerfile*
drwxrwxrwx 1 root root 512 Feb 22 14:07 html/
drwxrwxrwx 1 root root 512 Feb 22 14:15 kube/
-rwxrwxrwx 1 root root  69 Feb 22 14:07 wrapper.sh*
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# cat deployment.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: fardin31/cloudethix_initcontainer_fardin:v1
        ports:
        - containerPort: 80
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube#
```

**● Create one service.yaml file with type nodeport & select simpleapp-webapp**

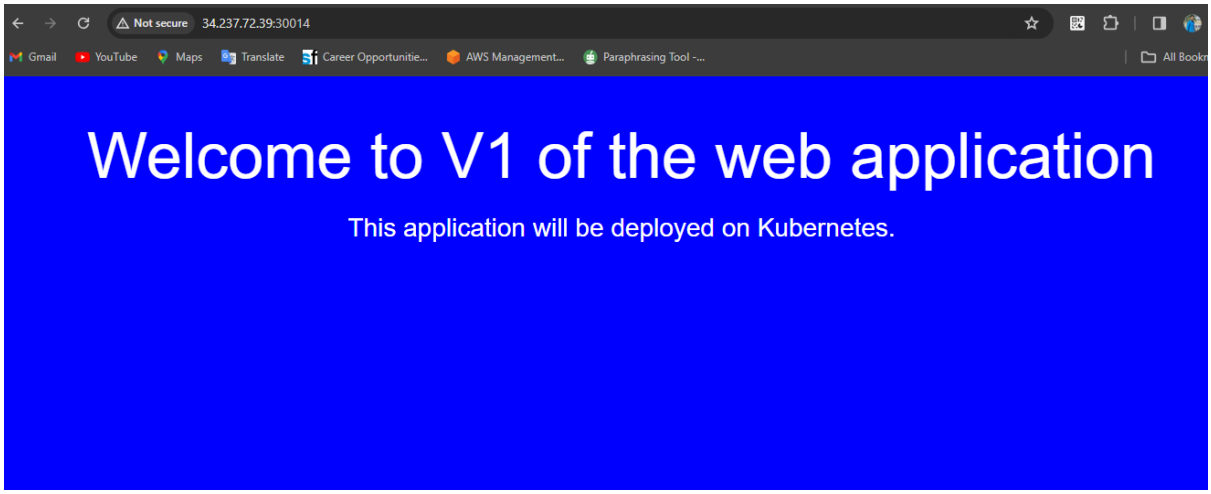
**pod with port 80 & targetPort 80 with any nodePort between range**

**30000-32768.**

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: web-service
spec:
  selector:
    app: simpleapp-webapp
  ports:
  - port: 80
    targetPort: 80
    nodePort: 30014
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kaf .
deployment.apps/simpleapp-webapp created
service/web-service created
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgp -o wide
NAME                              READY   STATUS    RESTARTS   AGE   IP               NODE       NOMINATED NODE   READINESS GATES
simpleapp-webapp-774f65887d-dm52s   1/1     Running   0          7s    10.111.158.101   worker-1   <none>           <none>
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgs
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes    ClusterIP   10.96.0.1       <none>        443/TCP        3h48m
web-service   NodePort    10.98.228.223   <none>        80:30014/TCP   18s
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/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@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# k delete -f .
deployment.apps "simpleapp-webapp" deleted
service "web-service" deleted
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgp
No resources found in default namespace.
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/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@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# cat deployment.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: fardin31/cloudethix_initcontainer_fardin:v2
        ports:
        - containerPort: 80
        volumeMounts:
        - name: html-volume
          mountPath: /usr/share/nginx/html
        - name: workdir
          mountPath: /work-dir
      volumes:
      - name: html-volume
        emptyDir: {}
      - name: workdir
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgp
NAME                                READY   STATUS            RESTARTS   AGE
simpleapp-webapp-68d586b89b-sjqq7   0/1     PodInitializing   0          3s
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgp
NAME                                READY   STATUS    RESTARTS   AGE
simpleapp-webapp-68d586b89b-sjqq7   1/1     Running   0          7s
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube#
```

```
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube# kgs
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes    ClusterIP   10.96.0.1       <none>        443/TCP        3h58m
web-service   NodePort    10.111.53.145   <none>        80:30014/TCP   24s
root@FARDIN:/mnt/f/k8s_Assignment/simpleapp/kube#
```

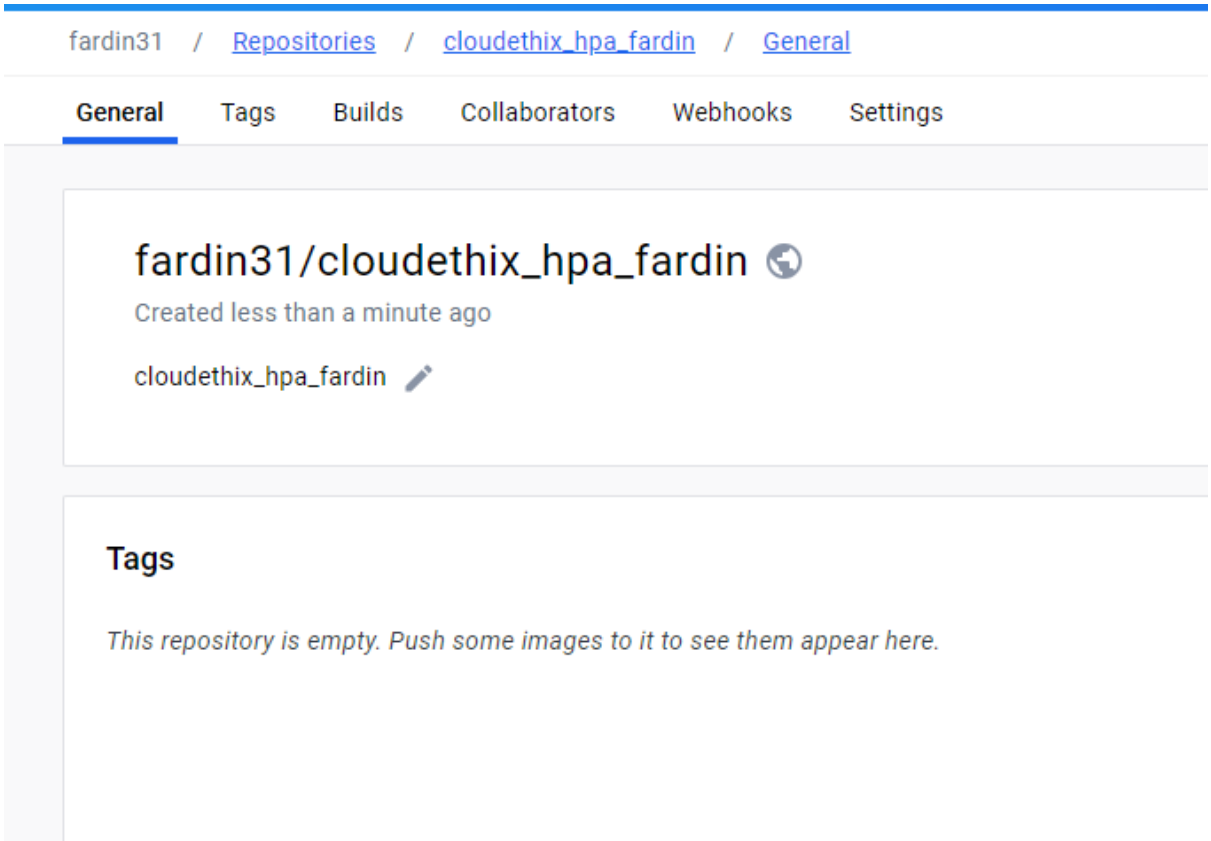
● **Once the deployment.yaml file is ready, create the deployment & access the**

**page in the browser and notice the changes.**


● **Prepare a well formatted document and write your understanding step by step**

**Que 13 →**

● **Create 1 Public Docker Hub registry named cloudethix_hpa_yourname.**




- **Clone below repository on your system.**

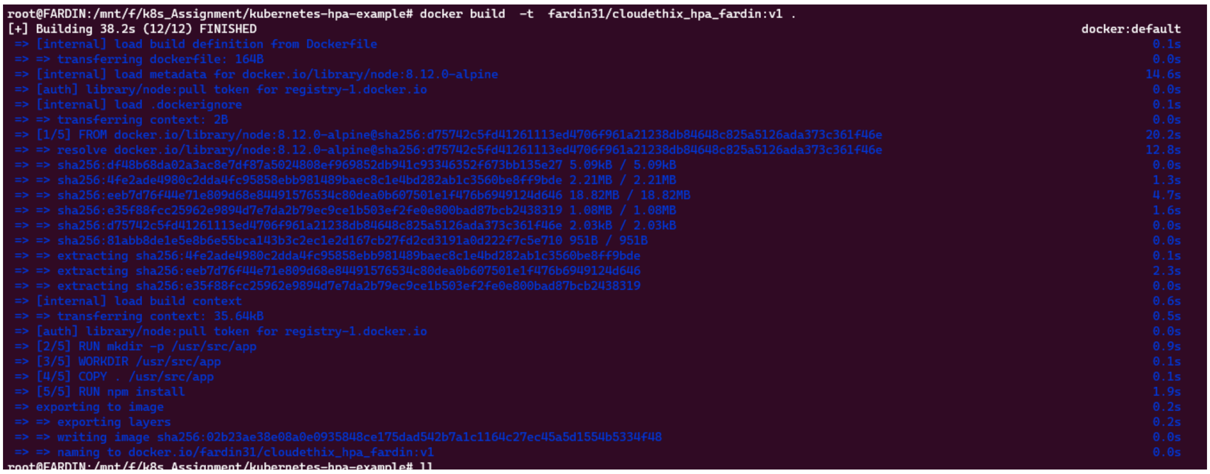
[https://github.com/vivekamin/kubernetes-hpa-example.git](https://github.com/vivekamin/kubernetes-hpa-example.git)

```
root@FARDIN:/mnt/f/k8s_Assignment# 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@FARDIN:/mnt/f/k8s_Assignment#
```

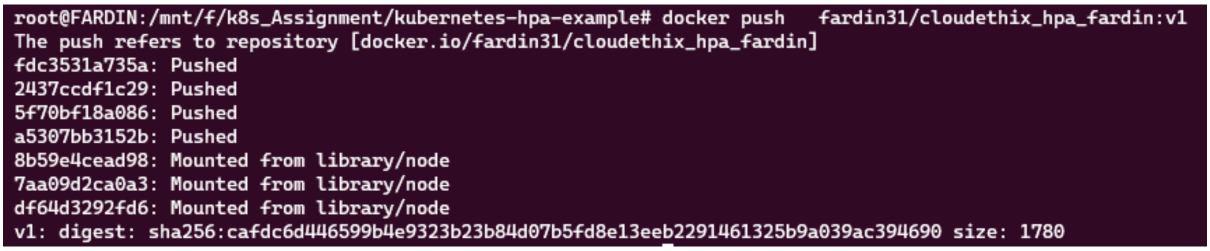
```
root@FARDIN:/mnt/f/k8s_Assignment# cd kubernetes-hpa-example/
root@FARDIN:/mnt/f/k8s_Assignment/kubernetes-hpa-example# ll
total 4
drwxrwxrwx 1 root root  512 Feb 22  2024 ./
drwxrwxrwx 1 root root  512 Feb 22  2024 ../
drwxrwxrwx 1 root root  512 Feb 22  2024 .git/
-rwxrwxrwx 1 root root  127 Feb 22  2024 Dockerfile*
-rwxrwxrwx 1 root root 2788 Feb 22  2024 README.md*
drwxrwxrwx 1 root root  512 Feb 22  2024 k8s/
-rwxrwxrwx 1 root root  272 Feb 22  2024 package.json*
drwxrwxrwx 1 root root  512 Feb 22  2024 src/
root@FARDIN:/mnt/f/k8s_Assignment/kubernetes-hpa-example#
```

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

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



```
root@FARDIN:/mnt/f/k8s_Assignment/kubernetes-hpa-example# docker build  -t  fardin31/cloudethix_hpa_fardin:v1 .
[+] Building 38.2s (12/12) FINISHED                                                                                      docker:default
 => [internal] load build definition from Dockerfile                                                                               0.1s
 => => transferring dockerfile: 164B                                                                                               0.0s
 => [internal] load metadata for docker.io/library/node:8.12.0-alpine                                                             14.6s
 => [auth] library/node:pull token for registry-1.docker.io                                                                        0.0s
 => [internal] load .dockerignore                                                                                                  0.1s
 => => transferring context: 2B                                                                                                    0.0s
 => [1/5] FROM docker.io/library/node:8.12.0-alpine@sha256:d75742c5fd41261113ed4706f961a21238db84648c825a5126ada373c361f46e       20.2s
 => => resolve docker.io/library/node:8.12.0-alpine@sha256:d75742c5fd41261113ed4706f961a21238db84648c825a5126ada373c361f46e       12.8s
 => => sha256:df48b68da02a3ac8e7df87a5024808ef969852db941c93346352f673bb135e27 5.09kB / 5.09kB                                     0.0s
 => => sha256:4fe2ade4980c2dda4fc95858ebb981489baec8c1e4bd282ab1c3560be8ff9bde 2.21MB / 2.21MB                                     1.3s
 => => sha256:eeb7d76f44e71e809d68e84491576534c80dea0b607501e1f476b6949124d646 18.82MB / 18.82MB                                   4.7s
 => => sha256:e35f88fcc25962e9894d7e7da2b79ec9ce1b503ef2fe0e800bad87bcb2438319 1.08MB / 1.08MB                                     1.6s
 => => sha256:d75742c5fd41261113ed4706f961a21238db84648c825a5126ada373c361f46e 2.03kB / 2.03kB                                     0.0s
 => => sha256:81abb8de1e5e8b6e55bca143b3c2ec1e2d167cb27fd2cd3191a0d222f7c5e710 951B / 951B                                         0.0s
 => => extracting sha256:4fe2ade4980c2dda4fc95858ebb981489baec8c1e4bd282ab1c3560be8ff9bde                                          0.1s
 => => extracting sha256:eeb7d76f44e71e809d68e84491576534c80dea0b607501e1f476b6949124d646                                          2.3s
 => => extracting sha256:e35f88fcc25962e9894d7e7da2b79ec9ce1b503ef2fe0e800bad87bcb2438319                                          0.0s
 => [internal] load build context                                                                                                  0.6s
 => => transferring context: 35.64kB                                                                                               0.5s
 => [auth] library/node:pull token for registry-1.docker.io                                                                        0.0s
 => [2/5] RUN mkdir -p /usr/src/app                                                                                                0.9s
 => [3/5] WORKDIR /usr/src/app                                                                                                     0.1s
 => [4/5] COPY . /usr/src/app                                                                                                      0.1s
 => [5/5] RUN npm install                                                                                                          1.9s
 => exporting to image                                                                                                             0.2s
 => => exporting layers                                                                                                            0.2s
 => => writing image sha256:02b23ae38e08a0e0935848ce175dad542b7a1c1164c27ec45a5d1554b5334f48                                       0.0s
 => => naming to docker.io/fardin31/cloudethix_hpa_fardin:v1                                                                       0.0s
root@FARDIN:/mnt/f/k8s_Assignment/kubernetes-hpa-example# ll
```



```
root@FARDIN:/mnt/f/k8s_Assignment/kubernetes-hpa-example# docker push   fardin31/cloudethix_hpa_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_hpa_fardin]
fdc3531a735a: Pushed
2437ccdf1c29: Pushed
5f70bf18a086: Pushed
a5307bb3152b: Pushed
8b59e4cead98: Mounted from library/node
7aa09d2ca0a3: Mounted from library/node
df64d3292fd6: Mounted from library/node
v1: digest: sha256:cafdc6d446599b4e9323b23b84d07b5fd8e13eeb2291461325b9a039ac394690 size: 1780
```

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

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: node-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: node-example
  template:
    metadata:
      labels:
        app: node-example
    spec:
      containers:
      - name: node-example
        image: fardin31/cloudethix_hpa_fardin:v1
        imagePullPolicy: Always
        ports:
        - containerPort: 3000
        resources:
            limits:
              cpu: "0.5"
            requests:
              cpu: "0.25"
deployment.yml (END)
```

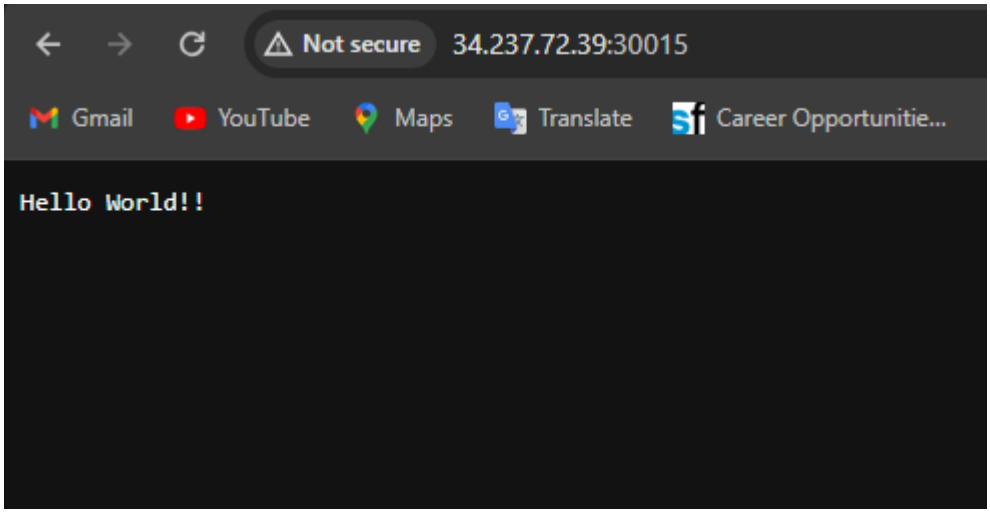
- **Open service.yml and change the type to nodePort and apply the same.**

```yaml
apiVersion: v1
kind: Service
metadata:
  name: hpa-service
  namespace: default
  labels:
    app: node-example
spec:
  selector:
    app: node-example
  ports:
  - port: 3000
    protocol: TCP
    targetPort: 3000
    nodePort: 30015
  type: NodePort
service.yml (END)
```

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

```yaml
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
hpa.yml (END)
```

- **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-**
**ttp://NODE_PORT_SERVICE_NAME; done"**

```
root@FARDIN:/mnt/f/k8sCluster_kubeadm_terraform# kubectl run -i --tty load-generator-04 --rm --image=busybox --restart=Never -- /bin/
sh -c "while sleep 0.01; do wget -q -O- http://node-example: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!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
Hello World!!
```

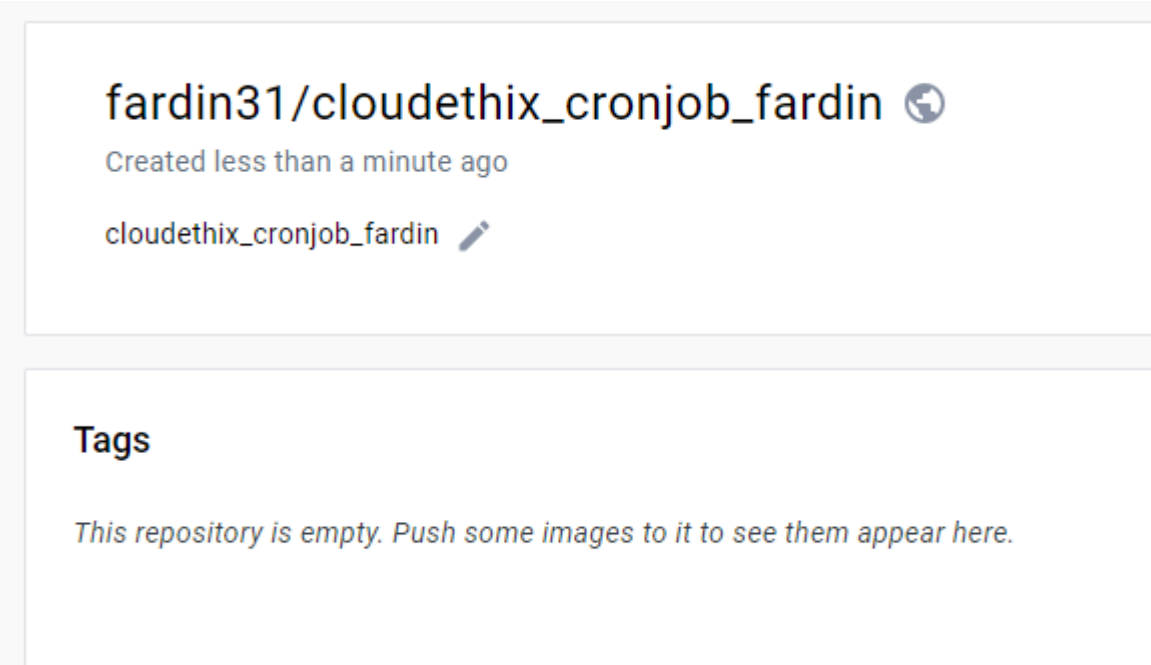
● **Check the HPA from kubectl command and also check if the deployment is scaling up.**

```
^C^Croot@FARDIN:/mnt/f/k8sCluster_kubeadm_terraform# kubectl get hpa
NAME           REFERENCE                 TARGETS        MINPODS   MAXPODS   REPLICAS   AGE
node-example   Deployment/node-example   <unknown>/1%   1         4         1          12m
root@FARDIN:/mnt/f/k8sCluster_kubeadm_terraform#
```

● **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.**

```
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"]
Dockerfile (END)
```

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

helloworld.py (END)
```

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

```
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# mv phythoncronjob.yaml phythoncronjob.yml
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# docker build  -t  fardin31/cloudethix_cronjob_fardin:v1 .
[+] Building 21.1s (11/11) FINISHED                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                                0.1s
 => => transferring dockerfile: 387B                                                                                                0.0s
 => [internal] load metadata for docker.io/library/python:3.7-alpine                                                               13.9s
 => [auth] library/python:pull token for registry-1.docker.io                                                                       0.0s
 => [internal] load .dockerignore                                                                                                   0.1s
 => => transferring context: 2B                                                                                                     0.0s
 => [1/5] FROM docker.io/library/python:3.7-alpine@sha256:f3d31c8677d03f0b3c724446077f229a6ce9d3ac430f5c08cd7dff00292048c3          5.2s
 => => resolve docker.io/library/python:3.7-alpine@sha256:f3d31c8677d03f0b3c724446077f229a6ce9d3ac430f5c08cd7dff00292048c3          0.0s
 => => sha256:4819c95424fc4a94767c9329b02238ebcce0bc682384cb671379bc1fb8a12b55 10.94MB / 10.94MB                                    2.1s
 => => sha256:f3d31c8677d03f0b3c724446077f229a6ce9d3ac430f5c08cd7dff00292048c3 1.65kB / 1.65kB                                      0.0s
 => => sha256:e6da3ee9bb64dd12b98fa609487f112fe1e365522e6e8345309db15c22a80a51 1.37kB / 1.37kB                                      0.0s
 => => sha256:1bac8ae77e4af0b868b62a75115616a20e025e0451eeed05d94a4cfc4523e58a 6.87kB / 6.87kB                                      0.0s
 => => sha256:96526aa774ef0126ad0fe9e9a95764c5fc37f409ab9e97021e7b4775d82bf6fa 3.40MB / 3.40MB                                      2.8s
 => => sha256:9875af95546db78168a6761b7fa205ed1cd0c153cd89356c1512e551c12b2d5c 622.29kB / 622.29kB                                  1.1s
 => => sha256:148762f75a1f92cc9857e9c488bf95d5aac61e9905ec47a7408025b2dd5c3b7a 240B / 240B                                          1.8s
 => => sha256:ea1518237b3753b3fe40ee773d77651704178d9baa72ae5012e13a992cfa6c63 2.85MB / 2.85MB                                      2.5s
 => => extracting sha256:96526aa774ef0126ad0fe9e9a95764c5fc37f409ab9e97021e7b4775d82bf6fa                                           0.2s
 => => extracting sha256:9875af95546db78168a6761b7fa205ed1cd0c153cd89356c1512e551c12b2d5c                                           0.5s
 => => extracting sha256:4819c95424fc4a94767c9329b02238ebcce0bc682384cb671379bc1fb8a12b55                                           0.8s
 => => extracting sha256:148762f75a1f92cc9857e9c488bf95d5aac61e9905ec47a7408025b2dd5c3b7a                                           0.0s
 => => extracting sha256:ea1518237b3753b3fe40ee773d77651704178d9baa72ae5012e13a992cfa6c63                                           0.3s
 => [internal] load build context                                                                                                   0.1s
 => => transferring context: 180B                                                                                                   0.0s
 => [2/5] RUN addgroup -S appgroup && adduser -S appuser -G appgroup                                                                0.8s
 => [3/5] WORKDIR /app                                                                                                              0.1s
 => [4/5] COPY helloworld.py /app/helloworld.py                                                                                     0.1s
 => [5/5] RUN chmod +x /app/helloworld.py                                                                                           0.5s
 => exporting to image                                                                                                              0.1s
 => => exporting layers                                                                                                             0.1s
```

```
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# docker push   fardin31/cloudethix_cronjob_fardin:v1
The push refers to repository [docker.io/fardin31/cloudethix_cronjob_fardin]
3cce0db51e21: Pushed
f1a19d69b7e1: Pushed
44c310ec3460: Pushed
23f7a6144d20: 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
v1: digest: sha256:352d66a3791c255ec2bc8585937f67efb49b669c2f8516661a9090186792b656 size: 2196
root@FARDIN:/mnt/f/k8s_Assignment/cron_job#
```

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

```yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: python-helloworld
spec:
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: python-helloworld
            image: fardin31/cloudethix_cronjob_fardin:v1
            command: ["/app/helloworld.py"]
          restartPolicy: OnFailure
```

phythoncronjob.yml (END)

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

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

```
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# kubectl get jobs
NAME                          COMPLETIONS   DURATION   AGE
python-helloworld-28476796    1/1           3s         2m29s
python-helloworld-28476797    1/1           4s         89s
python-helloworld-28476798    1/1           3s         29s
```

```
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# kgp
NAME                                READY   STATUS      RESTARTS   AGE
python-helloworld-28476803-6pwgz    0/1     Completed   0          2m29s
python-helloworld-28476804-dn7x2    0/1     Completed   0          89s
python-helloworld-28476805-bmt9h    0/1     Completed   0          29s
```

```
root@FARDIN:/mnt/f/k8s_Assignment/cron_job# k logs python-helloworld-28476803-6pwgz
Welcome to the Cloudethix World
Today is
2024-02-22 13:23:01.008093
root@FARDIN:/mnt/f/k8s_Assignment/cron_job#
```

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

**● 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
```