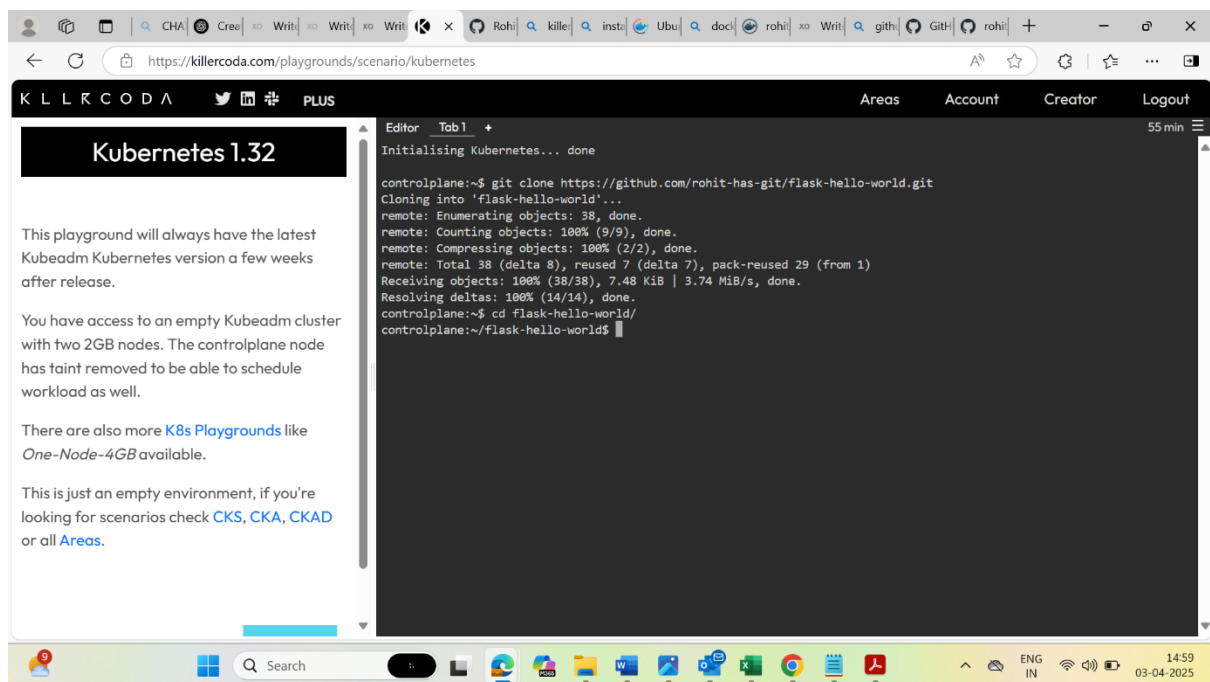
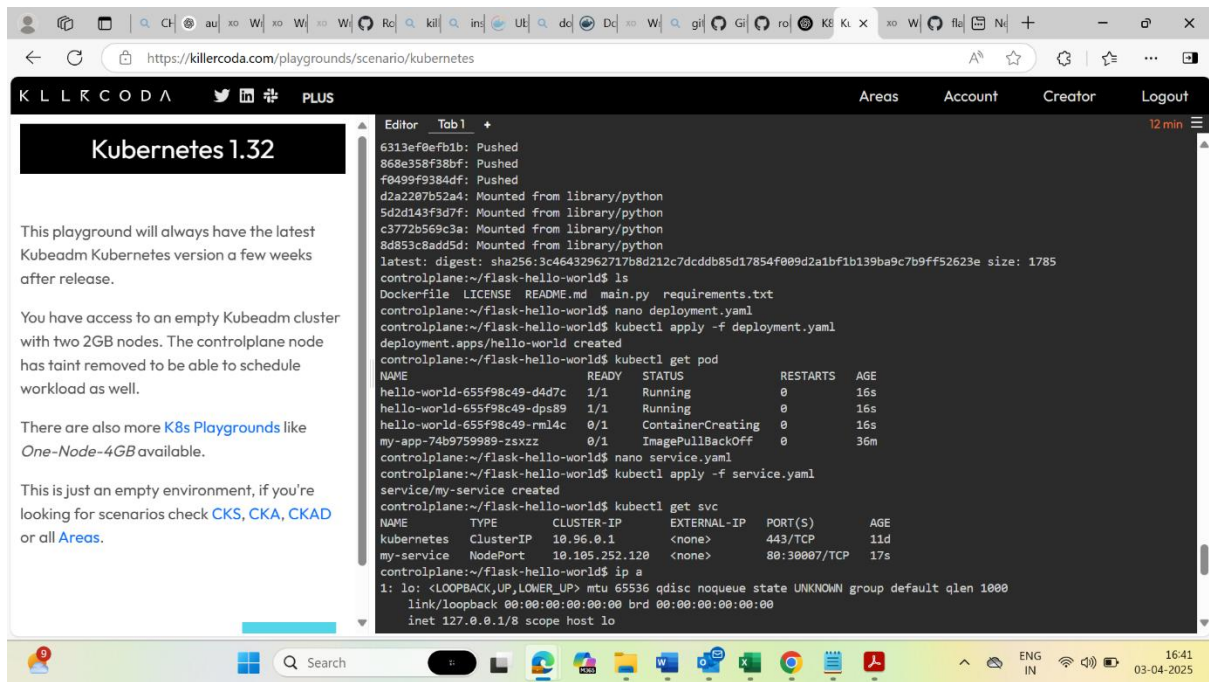


Question 1: Containerize and Push the Application

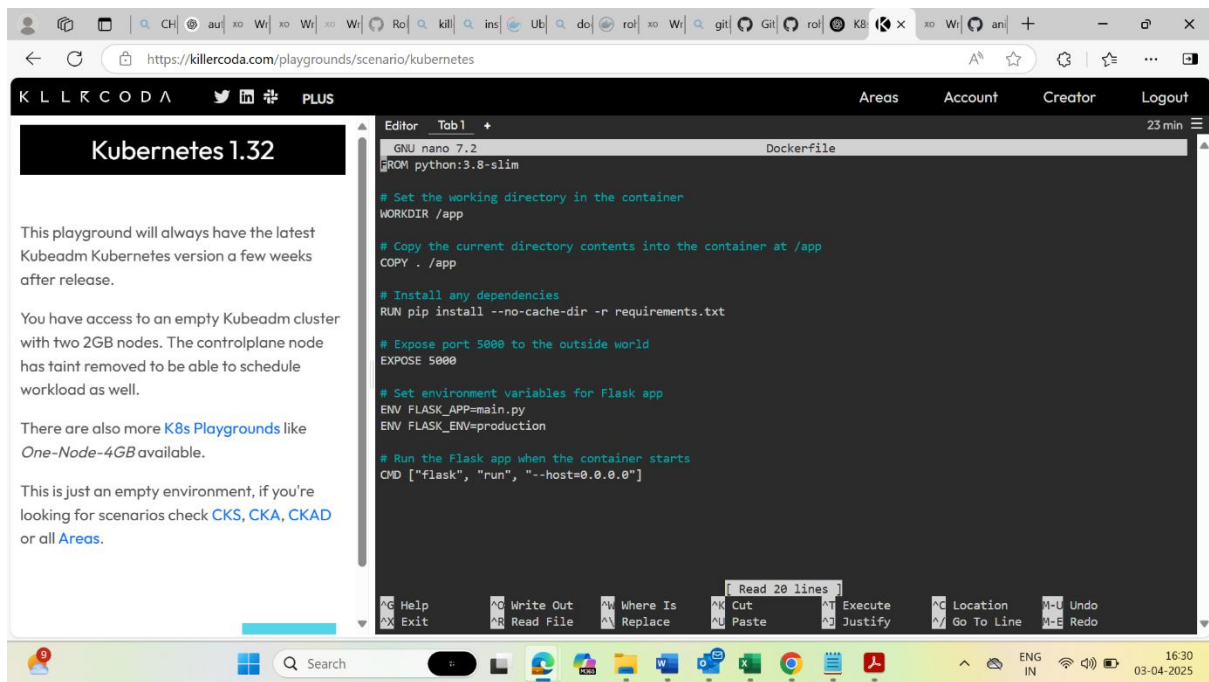
- Fork and clone a sample application repository (Java, Node.js, or Python preferred).
- Package the application as required (e.g., JAR, ZIP, etc.).
- Write a Dockerfile to containerize the application.
- Build the Docker image and push it to DockerHub.

[GithubLink](#)

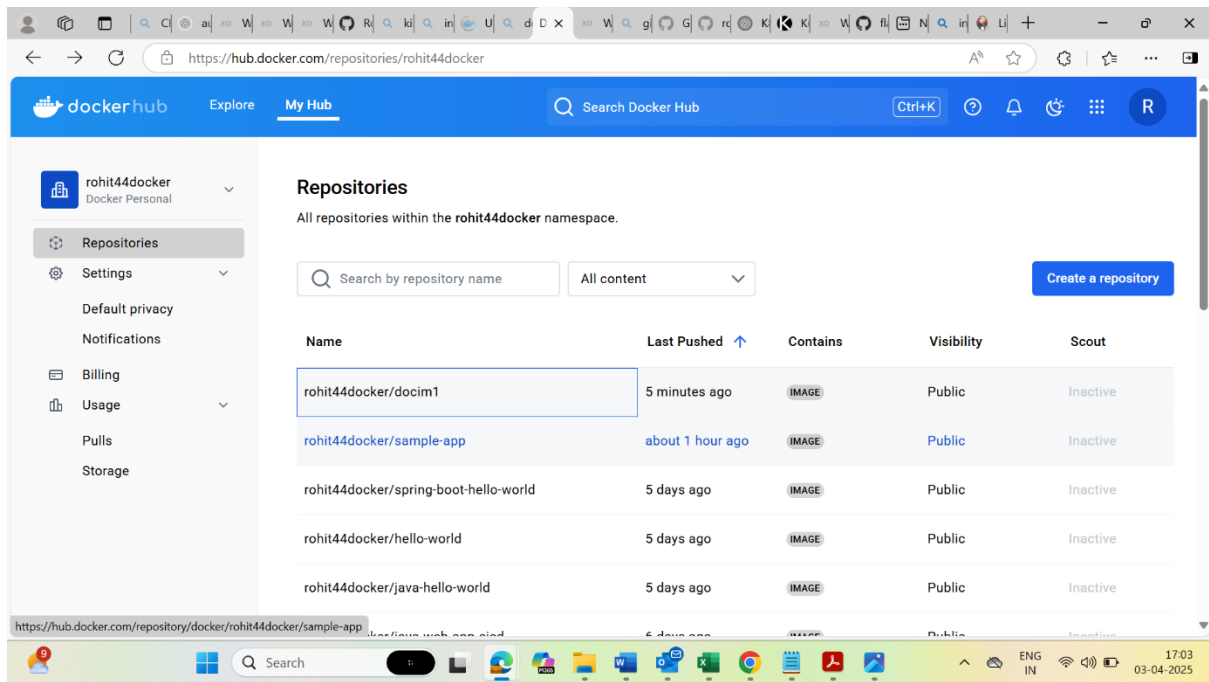




This is Dockerfile



This is Docker Image



Docker Image link below :

[Link](#)

Question 2: Kubernetes Deployment and Service Exposure

- Create Kubernetes `deployment.yaml` to deploy your Docker image.
- Create `service.yaml` to expose it using NodePort.
- Apply both YAMLS using `kubectl`, and verify deployment.
- Use `curl` with public IP and node port to access the service.

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```
6313ef0efb1b: Pushed
868e358f38bf: Pushed
f0499f9384df: Pushed
d2a2207b52a4: Mounted from library/python
5d2d143f3d7f: Mounted from library/python
c3772b569c3a: Mounted from library/python
8d853c8add5d: Mounted from library/python
latest: digest: sha256:3c46432962717b8d21c7dcddb85d17854f009d2a1bf1b139ba9c7b9ff52623e size: 1785
controlplane:~/flask-hello-world$ ls
controlplane:~/flask-hello-world$ nano deployment.yaml
controlplane:~/flask-hello-world$ kubectl apply -f deployment.yaml
deployment.apps/hello-world created
controlplane:~/flask-hello-world$ kubectl get pod
NAME                                READY   STATUS    RESTARTS   AGE
hello-world-655f98c49-d4d7c         1/1     Running   0           16s
hello-world-655f98c49-dps89         1/1     Running   0           16s
hello-world-655f98c49-rm14c         0/1     ContainerCreating   0           16s
my-app-74b9759989-zsxzz             0/1     ImagePullBackOff    0           36m
controlplane:~/flask-hello-world$ nano service.yaml
controlplane:~/flask-hello-world$ kubectl apply -f service.yaml
service/my-service created
controlplane:~/flask-hello-world$ kubectl get svc
NAME         TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1     <none>        443/TCP          11d
my-service   NodePort    10.105.252.120 <none>        80:30007/TCP     17s
controlplane:~/flask-hello-world$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
```

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```
controlplane:~/flask-hello-world$ nano service.yaml
controlplane:~/flask-hello-world$ nano deployment.yaml
controlplane:~/flask-hello-world$ ls
controlplane:~/flask-hello-world$ nano Dockerfile
controlplane:~/flask-hello-world$ kubectl get po,svc
NAME                                READY   STATUS    RESTARTS   AGE
pod/hello-world-655f98c49-d4d7c     1/1     Running   0           8m22s
pod/hello-world-655f98c49-dps89     1/1     Running   0           8m22s
pod/hello-world-655f98c49-rm14c     1/1     Running   0           8m22s
pod/my-app-74b9759989-zsxzz         0/1     ImagePullBackOff    0           44m

NAME         TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
service/kubernetes   ClusterIP   10.96.0.1     <none>        443/TCP          11d
service/my-service   NodePort    10.105.252.120 <none>        80:30007/TCP     6m25s
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/my-service
error: no objects passed to scale deployments.apps "my-service" not found
controlplane:~/flask-hello-world$ kubectl get deploy
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
hello-world  3/3     3            3           9m47s
my-app       0/1     1            0           45m
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/hello-world
deployment.apps/hello-world scaled
controlplane:~/flask-hello-world$ kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
hello-world-655f98c49-d4d7c         1/1     Running   0           10m
hello-world-655f98c49-dps89         1/1     Running   0           10m
hello-world-655f98c49-rm14c         1/1     Running   0           10m
my-app-74b9759989-zsxzz             0/1     ImagePullBackOff    0           46m
controlplane:~/flask-hello-world$
```

Curl

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```

valid_lft 86310626sec preferred_lft 75521426sec
inet6 fe80::a944:43fe:e7f1:c11a/64 scope link
valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1454 qdisc noqueue state DOWN group default
link/ether 02:42:45:9b:2f:5c brd ff:ff:ff:ff:ff:ff
inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
valid_lft forever preferred_lft forever
inet6 fe80::42:45ff:fe9b:2f5c/64 scope link
valid_lft forever preferred_lft forever
4: flannel.1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state UNKNOWN group default
link/ether aa:db:47:a8:78:b1 brd ff:ff:ff:ff:ff:ff
inet 192.168.0.0/32 brd 192.168.0.0 scope global flannel.1
valid_lft forever preferred_lft forever
inet6 fe80::a8db:47ff:fea8:78b1/64 scope link
valid_lft forever preferred_lft forever
7: cali0eb6f59d2cf@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
link/ether ee:ee:ee:ee:ee:ee brd ff:ff:ff:ff:ff:ff link-netns cni-2cd3ec58-284f-acf1-a11f-6e21cc9a6af9
inet6 fe80::ecee:eeff:feee:eeee/64 scope link
valid_lft forever preferred_lft forever
8: cali052f3e6c0aa@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
link/ether ee:ee:ee:ee:ee:ee brd ff:ff:ff:ff:ff:ff link-netns cni-c8710970-cad3-70d9-e81c-e6d02d30730e
inet6 fe80::ecee:eeff:feee:eeee/64 scope link
valid_lft forever preferred_lft forever
29: cali0eb219be0fe@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
link/ether ee:ee:ee:ee:ee:ee brd ff:ff:ff:ff:ff:ff link-netns cni-b3422e48-00d8-1361-b352-9403e2c79045
inet6 fe80::ecee:eeff:feee:eeee/64 scope link
valid_lft forever preferred_lft forever
controlplane:~/flask-hello-world$ ^C
controlplane:~/flask-hello-world$ curl 172.30.1.2:30007/
Hello, World!controlplane:~/flask-hello-world$

```

Question 3: Scale the Deployment

- Scale the application to 3 replicas using `kubectl scale`.
- Validate scaling with `kubectl get pods`.
- Scale it back down to 1 replica.

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```

controlplane:~/flask-hello-world$ nano service.yaml
controlplane:~/flask-hello-world$ nano deployment.yaml
controlplane:~/flask-hello-world$ ls
Dockerfile LICENSE README.md deployment.yaml main.py requirements.txt service.yaml
controlplane:~/flask-hello-world$ nano Dockerfile
controlplane:~/flask-hello-world$ kubectl get po,svc
NAME READY STATUS RESTARTS AGE
pod/hello-world-655f98c49-d4d7c 1/1 Running 0 8m22s
pod/hello-world-655f98c49-dps89 1/1 Running 0 8m22s
pod/hello-world-655f98c49-rm14c 1/1 Running 0 8m22s
pod/my-app-74b9759989-zsxzz 0/1 ImagePullBackOff 0 44m

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 11d
service/my-service NodePort 10.105.252.120 <none> 80:30007/TCP 6m25s
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/my-service
error: no objects passed to scale deployments.apps "my-service" not found
controlplane:~/flask-hello-world$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
hello-world 3/3 3 3 9m47s
my-app 0/1 1 0 45m
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/hello-world
deployment.apps/hello-world scaled
controlplane:~/flask-hello-world$ kubectl get po
NAME READY STATUS RESTARTS AGE
hello-world-655f98c49-d4d7c 1/1 Running 0 10m
hello-world-655f98c49-dps89 1/1 Running 0 10m
hello-world-655f98c49-rm14c 1/1 Running 0 10m
my-app-74b9759989-zsxzz 0/1 ImagePullBackOff 0 46m
controlplane:~/flask-hello-world$

```

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```
controlplane:~/flask-hello-world$ kubectl scale --replicas=1 deployment/hello-world
deployment.apps/hello-world scaled
controlplane:~/flask-hello-world$ kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
hello-world-655f98c49-d4d7c        1/1     Terminating    0          11m
hello-world-655f98c49-dps89        1/1     Terminating    0          11m
hello-world-655f98c49-rml4c        1/1     Running         0          11m
my-app-74b9759989-zsxzz            0/1     ImagePullBackOff 0          46m
controlplane:~/flask-hello-world$
```

Question 4: Automate Deployment with Jenkins

Installation of jenkins

Kubernetes 1.32

This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.

You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.

There are also more [K8s Playgrounds](#) like [One-Node-4GB](#) available.

This is just an empty environment, if you're looking for scenarios check [CKS](#), [CKA](#), [CKAD](#) or all [Areas](#).

```
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
controlplane:~$ systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-04-03 11:48:50 UTC; 2min 19s ago
     Main PID: 24267 (java)
       Tasks: 38 (limit: 2320)
      Memory: 301.6M (peak: 398.5M)
         CPU: 22.430s
      CGroup: /system.slice/jenkins.service
              └─24267 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins

Apr 03 11:48:40 controlplane jenkins[24267]: 9806f647e14a45c28d8410b4bf050418
Apr 03 11:48:40 controlplane jenkins[24267]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Apr 03 11:48:40 controlplane jenkins[24267]: *****
Apr 03 11:48:40 controlplane jenkins[24267]: *****
Apr 03 11:48:40 controlplane jenkins[24267]: *****
Apr 03 11:48:50 controlplane jenkins[24267]: 2025-04-03 11:48:50.358+0000 [id=31] INFO jenkins.InitReap
Apr 03 11:48:50 controlplane jenkins[24267]: 2025-04-03 11:48:50.387+0000 [id=23] INFO hudson.lifecycle
Apr 03 11:48:50 controlplane systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Apr 03 11:48:51 controlplane jenkins[24267]: 2025-04-03 11:48:51.448+0000 [id=46] INFO h.m.DownloadSer
Apr 03 11:48:51 controlplane jenkins[24267]: 2025-04-03 11:48:51.452+0000 [id=46] INFO hudson.util.Retp
lines 1-20/20 (END)
```

