Refer confluence for more details : <https://confluence.forge.avaya.com/display/IXOUTREACH/ELK+Stack+with+OBaaS+Services#ELKStackwithOBaaSServices-Step1-Namespace>

**Detailed and exact steps to deploy ELK, please refer attached yaml**

1. **Create Namespace:**

kubectl create ns elk-logger

1. **Create Elasticsearch Certificates, refer 'elasticCertificate.yaml'**

kubectl apply -f elasticCertificate.yaml -n elk-logger

1. **Create Elastic Search**
   1. helm repo add elastic https://helm.elastic.co
   2. helm repo update
   3. refer 'elasticsearch.yaml' [ **Note**: confirm ‘tests’ is disabled in yaml]
   4. helm install elasticsearch elastic/elasticsearch --namespace elk-logger -f ./elasticsearch.yaml
   5. Output of above command will be like below

$ helm install elasticsearch elastic/elasticsearch --namespace elk-logger -f ./elasticsearch.yaml

NAME: elasticsearch

LAST DEPLOYED: Fri Sep 22 07:16:24 2023

NAMESPACE: elk-logger

STATUS: deployed

REVISION: 1

NOTES:

1. Watch all cluster members come up.

$ kubectl get pods --namespace=elk-logger -l app=elasticsearch-master -w

2. Retrieve elastic user's password.

$ kubectl get secrets --namespace=elk-logger elasticsearch-master-credentials -ojsonpath='{.data.password}' | base64 -d

3. Test cluster health using Helm test.

$ helm --namespace=elk-logger test elasticsearch

* 1. Get and store the password for future use and logging in as administrator using below command
  2. kubectl get secrets --namespace=elk-logger elasticsearch-master-credentials -ojsonpath='{.data.password}' | base64 -d
  3. output will be like below
  4. MHciIv41cBx4rJyd

1. **Generate 2 encryption keys locally to create secret for Kibana**
   1. encryptionkey1=$(docker run --rm [docker.elastic.co/kibana/kibana:8.4.3](http://docker.elastic.co/kibana/kibana:8.4.3) /bin/sh -c "< /dev/urandom tr -dc \_A-Za-z0-9 | head -c50")
   2. echo $encryptionkey1
   3. YrngGtiX5nHGyp1Tacl4Lb8DR\_gZH3MP8\_tO8Sux4EFYWBbZ59
   4. encryptionkey2=$(docker run --rm [docker.elastic.co/kibana/kibana:8.4.3](http://docker.elastic.co/kibana/kibana:8.4.3) /bin/sh -c "< /dev/urandom tr -dc \_A-Za-z0-9 | head -c50")
   5. echo $encryptionkey2
   6. S3SbX1bEV\_ZcwKWyQrMuInPhZTEKcXxONXcdkYUztE0G5VFBwR
2. **Using above generated encryption key create secret**
   1. Use above generated encryption key to generate secret in below format
   2. kubectl create secret --namespace=elk-logger generic kibana-keys --from-literal=secEncKey=<encryptionkey1> --from-literal=repEncKey=<encryptionkey2>
   3. for e.g. kubectl create secret --namespace=elk-logger generic kibana-keys --from-literal=secEncKey=YrngGtiX5nHGyp1Tacl4Lb8DR\_gZH3MP8\_tO8Sux4EFYWBbZ59 --from-literal=repEncKey=S3SbX1bEV\_ZcwKWyQrMuInPhZTEKcXxONXcdkYUztE0G5VFBwR
3. **Create Kibana certificate**
   1. Refer ‘kibanaCertificate.yml’
   2. kubectl apply -f kibanaCertificate.yml -n elk-logger
4. **Deploy Kibana**
   1. Refer ‘kibana.yaml’
   2. Remove ‘xpack.fleet.registryUrl:’
   3. helm install kibana elastic/kibana --namespace elk-logger -f ./kibana.yaml
   4. Output will be like below
      1. $ helm install kibana elastic/kibana --namespace elk-logger -f ./kibana.yaml
      2. NAME: kibana
      3. LAST DEPLOYED: Fri Sep 22 18:10:50 2023
      4. NAMESPACE: elk-logger
      5. STATUS: deployed
      6. REVISION: 1
      7. TEST SUITE: None
      8. NOTES:
      9. 1. Watch all containers come up.
      10. $ kubectl get pods --namespace=elk-logger -l release=kibana -w
      11. 2. Retrieve the elastic user's password.
      12. $ kubectl get secrets --namespace=elk-logger elasticsearch-master-credentials -ojsonpath='{.data.password}' | base64 -d
      13. 3. Retrieve the kibana service account token.
      14. $ kubectl get secrets --namespace=elk-logger kibana-kibana-es-token -ojsonpath='{.data.token}' | base64 -d
5. Deploy Kafka
   1. helm repo add rhcharts <https://ricardo-aires.github.io/helm-charts/>
   2. helm repo update
   3. helm upgrade --install elk-kafka rhcharts/kafka -f ./kafkavalues.yaml -n elk-logger
6. Deploy logstash
   1. Refer yaml file ‘logstash.yaml’, [use kafka, do not use Redis]
   2. Deploy logstash using below command
   3. helm install logstash elastic/logstash --namespace elk-logger -f ./logstash.yaml
   4. helm upgrade logstash elastic/logstash --namespace elk-logger -f ./logstash.yaml
7. Deploy filebeat
   1. Run below command to copy secret ‘elasticsearch-master-credentials’ to namespace kube-system, which will be used by filebeat

kubectl get secret elasticsearch-master-credentials -n elk-logger -o yaml \

| sed s/"namespace: elk-logger"/"namespace: kube-system"/\

| kubectl apply -n kube-system -f -

* 1. Deploy filebeat using below command.
  2. kubectl apply -f filebeat-kubernetes.yaml

1. Deploy Metric Beat
   1. helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
   2. helm repo update
   3. helm install kube-state-metrics prometheus-community/kube-state-metrics
   4. Refer ‘metricbeat-kubernetes.yaml’, few modifications required, like aks, subscription details and deploy it
   5. Kubectl apply -f metricbeat-kubernetes.yaml
2. Deploy HeartBeat[Observability >> Uptime > Monitors]
   1. Refer ‘heartbeat-kubenetes.yaml’
   2. $ k apply -f heartbeat-kubenetes.yaml
   3. Output will look like below.
   4. configmap/heartbeat-deployment-config configured
   5. deployment.apps/heartbeat configured
   6. clusterrolebinding.rbac.authorization.k8s.io/heartbeat unchanged
   7. rolebinding.rbac.authorization.k8s.io/heartbeat unchanged
   8. rolebinding.rbac.authorization.k8s.io/heartbeat-kubeadm-config unchanged
   9. clusterrole.rbac.authorization.k8s.io/heartbeat unchanged
   10. role.rbac.authorization.k8s.io/heartbeat unchanged
   11. role.rbac.authorization.k8s.io/heartbeat-kubeadm-config unchanged
   12. serviceaccount/heartbeat unchanged

**NOTE**: In case ‘elk-kafka’ pods failed to start with ‘crashloopbackerror’ and it’s logs showing below log entries please run script ‘deploykafka.sh’

*>> "[2023-09-29 13:03:22,862] ERROR Failed to clean up log for \_\_consumer\_offsets-49 in dir /var/lib/kafka/data due to IOException (kafka.server.LogDirFailureChannel)  
>> java.nio.file.FileSystemException: /var/lib/kafka/data/\_\_consumer\_offsets-49/00000000000000000000.log.cleaned: Operation not permitted"*

Refer below URL’s to access AT1 and AT2 kibana to access pod logs, metrics and hearbeat details.

<https://obaas-kibana.at1.avaya-obaas.com:5601/> obaas\_user/Kibana@123$

<https://obaas-kibana.at2.avaya-obaas.com:5601/> obaas\_user/Kibana@123$

Access Kibana: <https://20.10.229.21:5601/> :

<https://obaas-kibana.at1.avaya-obaas.com:5601/login?next=%2F> User/pwd: elastic/MHciIv41cBx4rJyd

<https://obaas-kibana.at1.avaya-obaas.com:5601/login?next=%2F> User/pwd: elastic/QZcHMiUJr8HOJMR6

How to get logs and stats: <https://confluence.forge.avaya.com/display/IXOUTREACH/ELK%3A+View+logs+and+metrics>

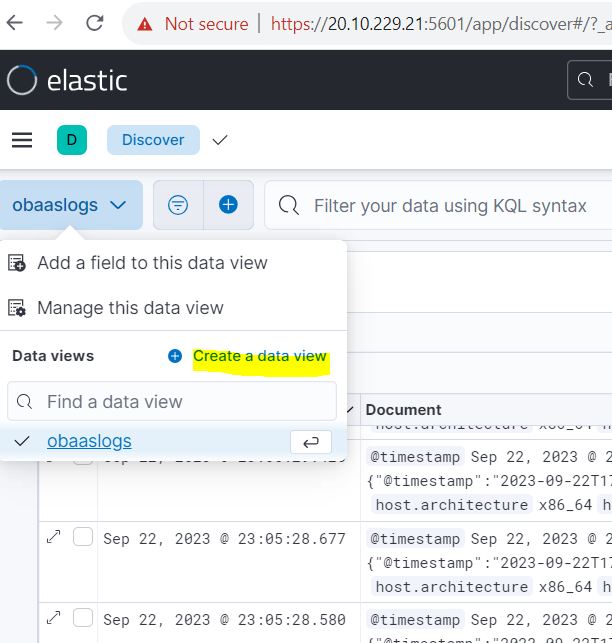
Can retrieve username and password for kibana using below commands

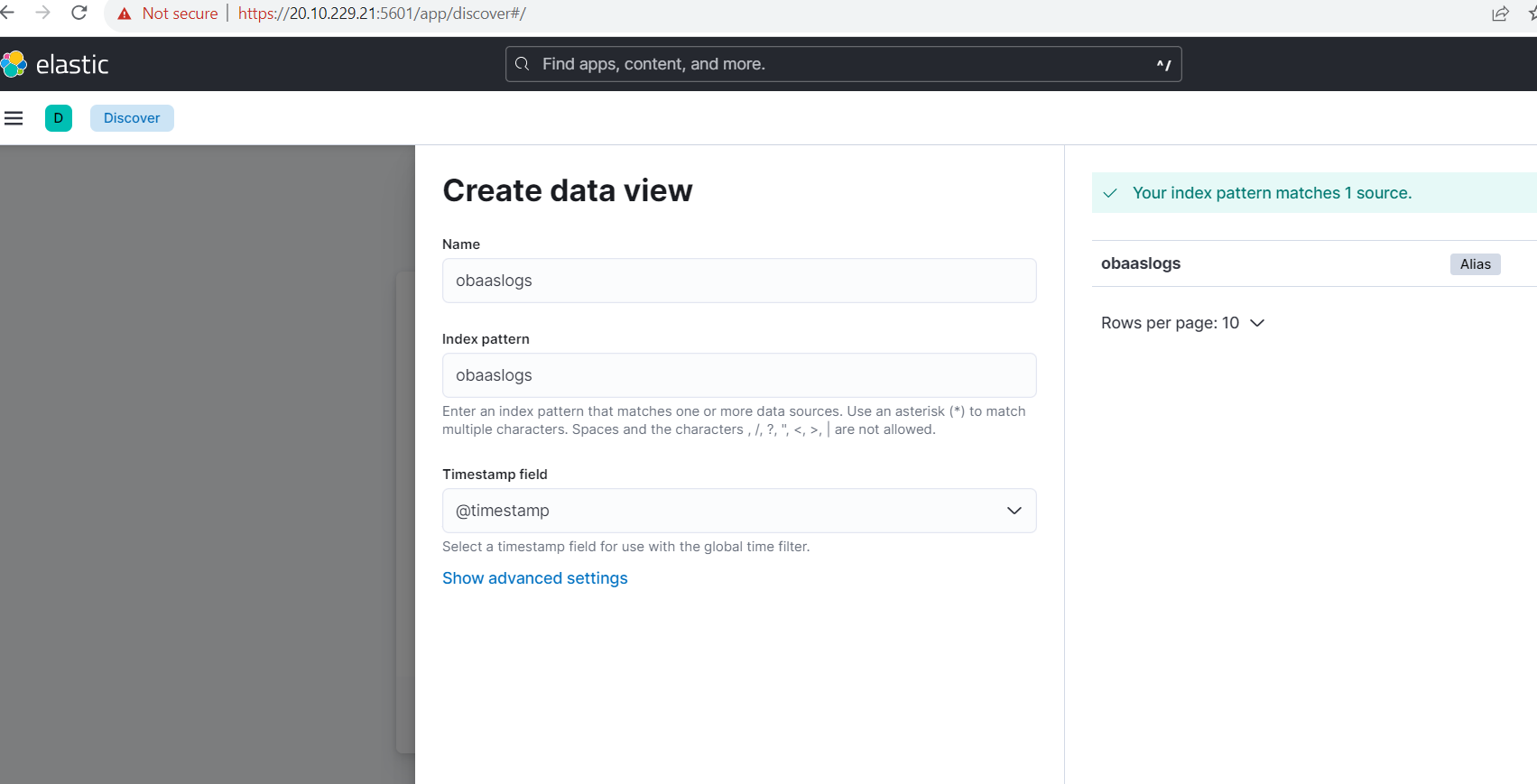
kubectl get secrets --namespace=elk-logger elasticsearch-master-credentials -ojsonpath='{.data.username}' | base64 -d

kubectl get secrets --namespace=elk-logger elasticsearch-master-credentials -ojsonpath='{.data.password}' | base64 -d

Don’t use redis cache, prefer Apache kafka.

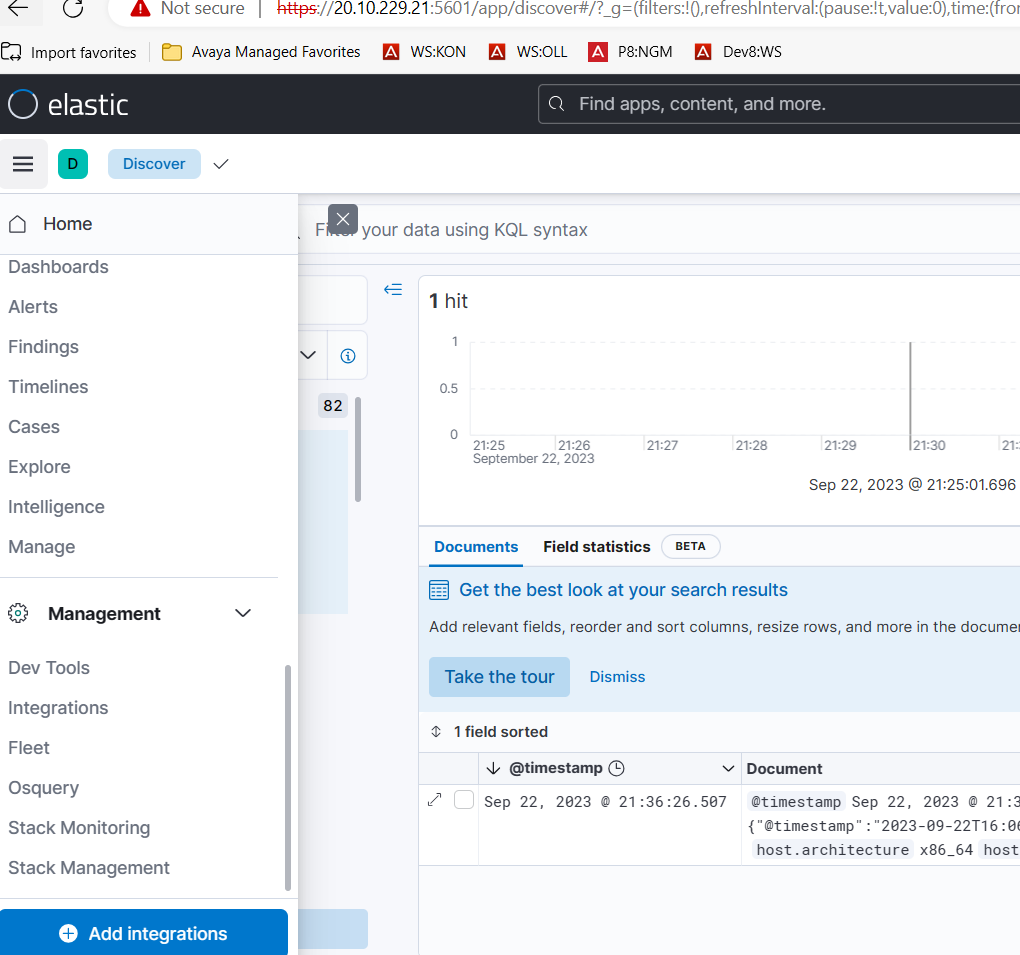
**How to create Data View?: Refer below two screenshots.**

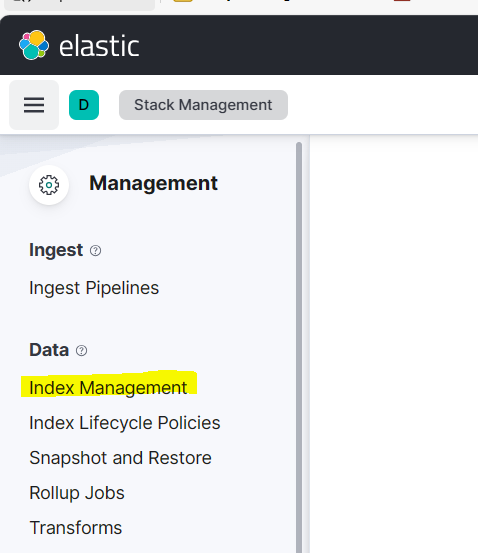


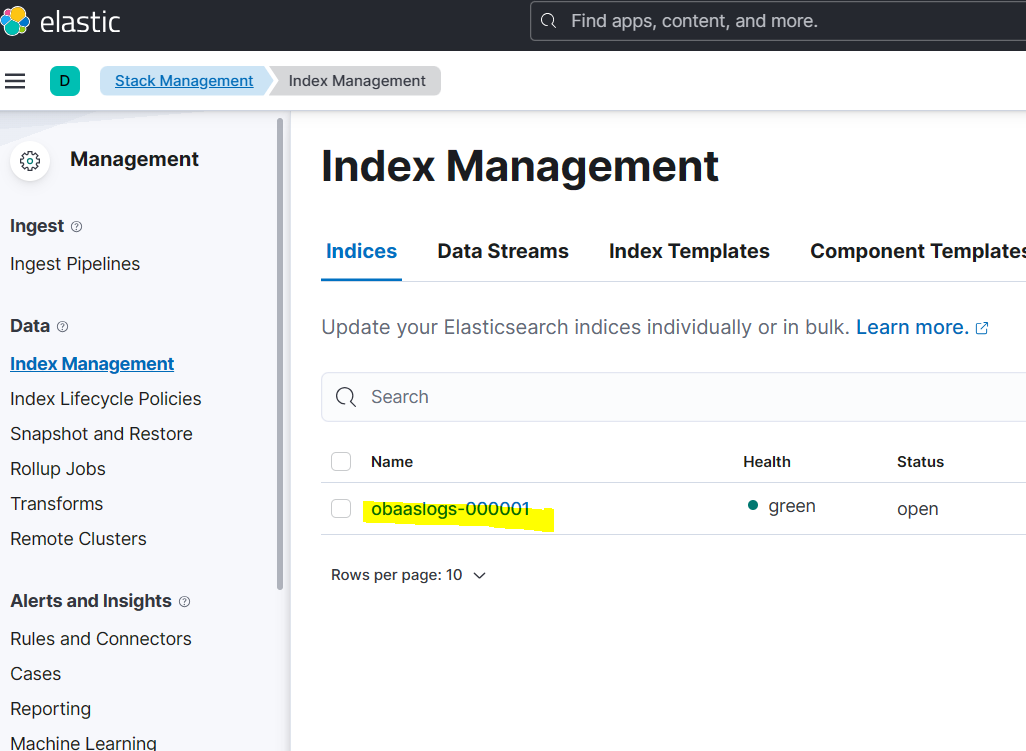


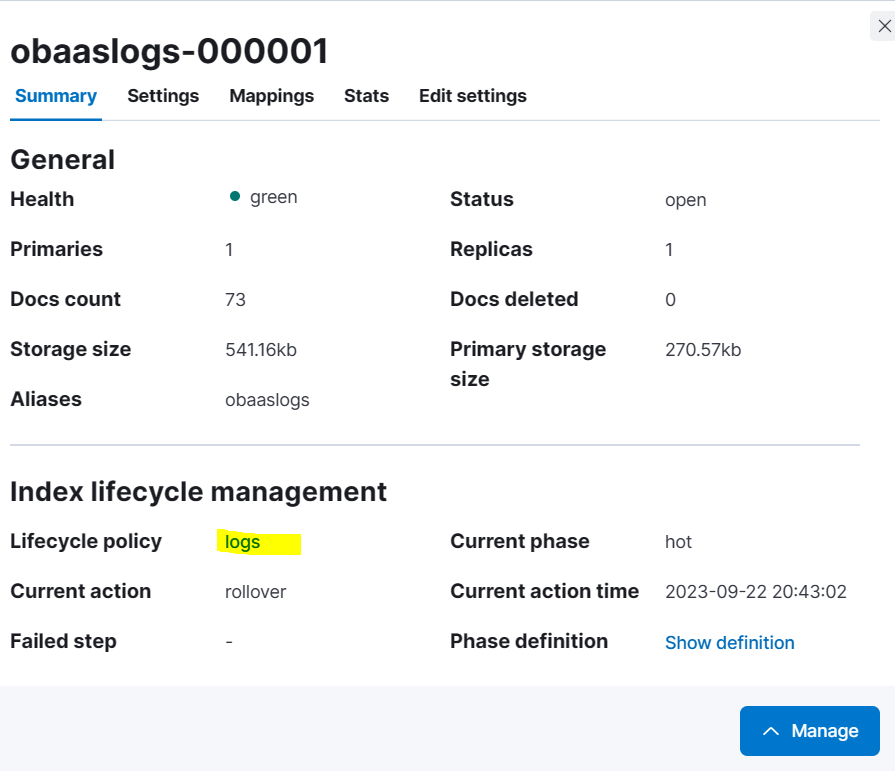
**How to create Index?**

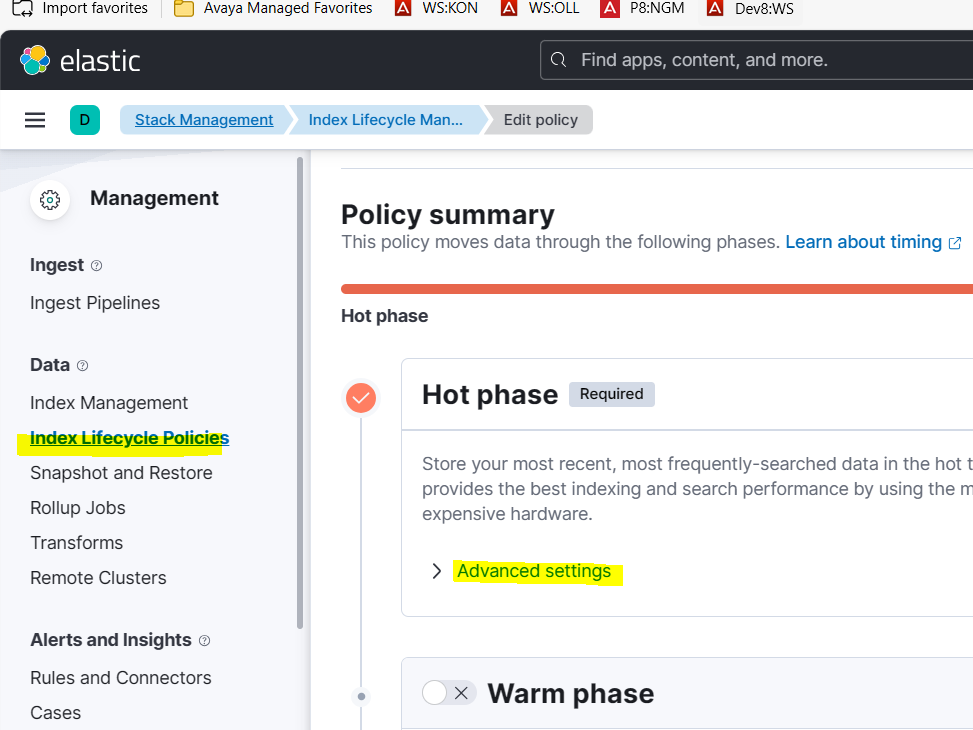
1. Click on Management >> Stack Management >> Index Management
2. Click on ‘obaaslogs-000001 >> ‘logs’
3. Enable ‘Delete data after this phase’
4. Click on ‘Advance Settings’
5. Enable ‘Roll Over’
6. Maximum primary shard size : 20 gigabytes
7. Maximum Days: 3 Days
8. Maximum Index Size : 20 gigabytes
9. Move data into phase when: 5 Hours
10. Click ‘Save Policy’s

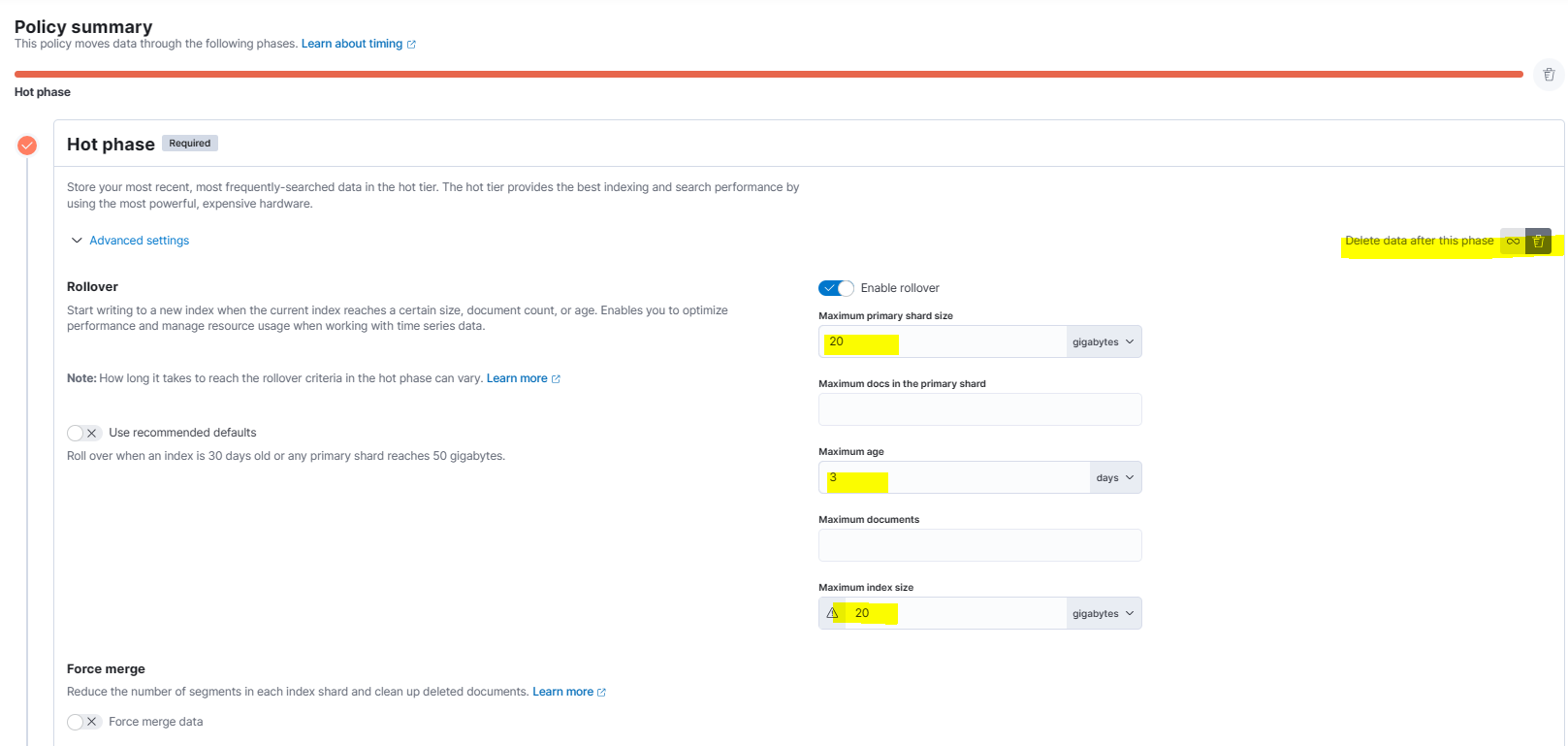


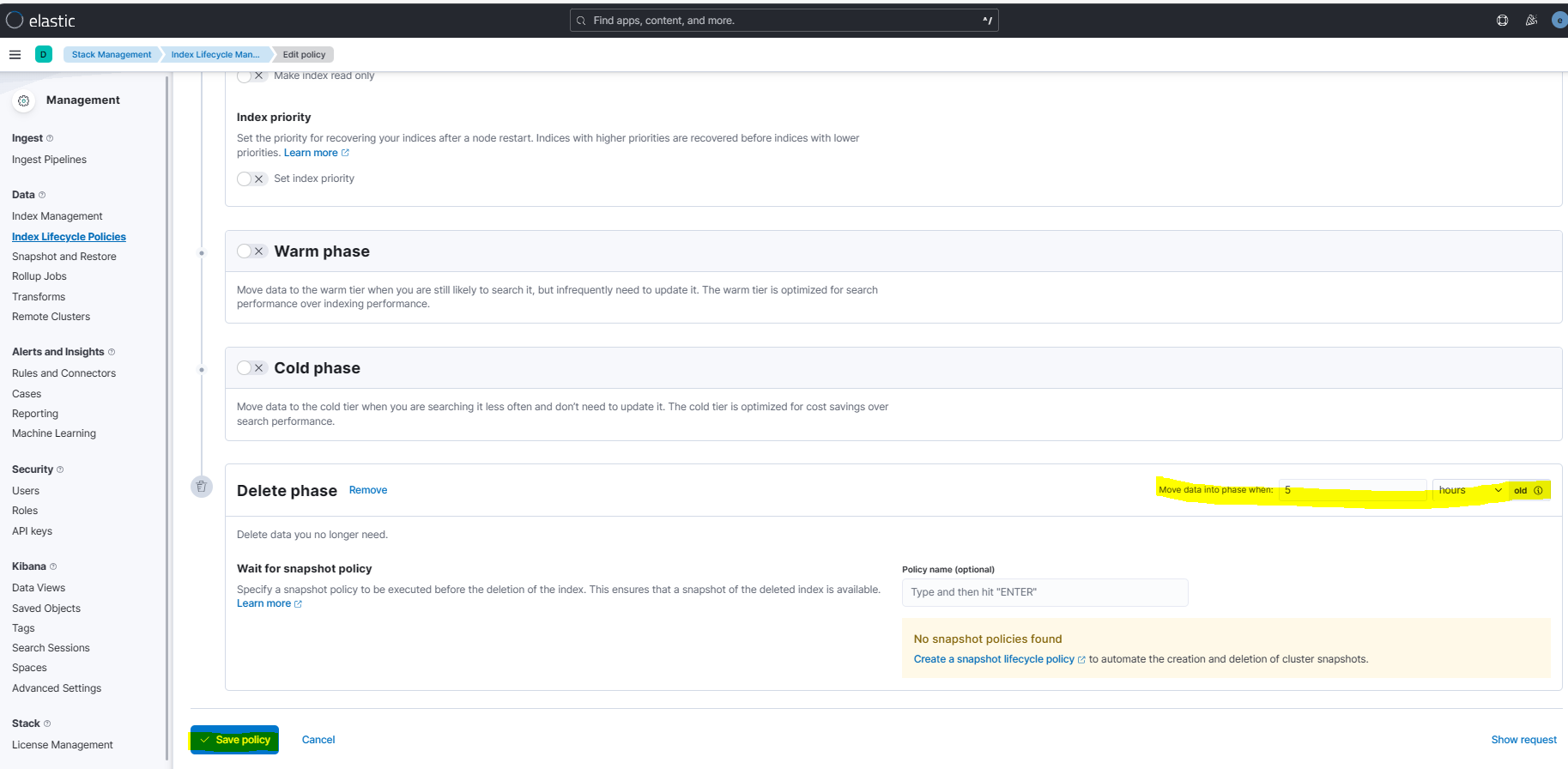












Cedric ELK: <https://148.147.173.62:31000/#/login> admin / admin123