Openshift-EX280-Version_4.5 **Duration**: 3 Hrs

Max.Mark:300
PassMark: 210

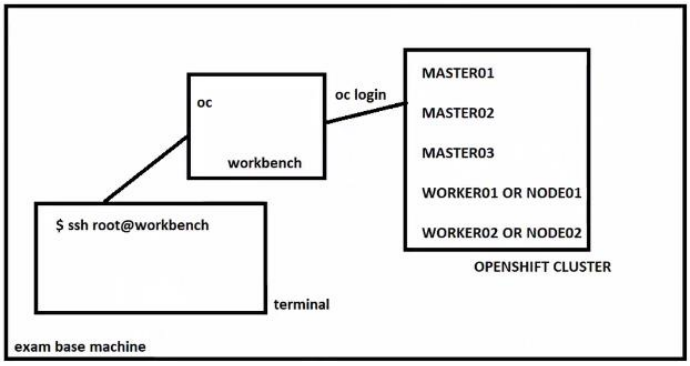
Initial Setup

S.NO	*DOMAIN NAME*	*IP ADDRESS*
1	Workbench.lab.example.com	172.25.250.11
2	Master.lab.example.com	172.25.250.12
3	Node1.lab.example.com	172.25.250.13
4	Node2.lab.example.com	172.25.250.14
5	Utility.lab.example.com	172.25.250.15
6	API Server URL	https://api.ocp4.example.com:6443

1. Wild-card domain for the cluster: apps.ocp4.example.com
2. Documentation about openshift can be accessed at the following url: *https://access.redhat.com/documentation/en-us/openshift_container_platform/4.5/*
3. Kubeadmin password will be available in the location as /root/kubeadmin.conf
4. Root password for login in to workbench VM will be provided in the exam itself

Question Outline

1. Configure the Identity Provider for the Openshift
2. Configure Cluster permissions
3. Configure Project permissions
4. Create Groups and configure permissions
5. Configure Quotas for the Project
6. Configure Limits for the Project
7. Deploy an Application
8. Configure and Deploy a secure route
9. Scale the Application manually
10. Configure Auto-scaling for an Application
11. Configure a Secret
12. Use the Secret value for Application Deployment
13. Configure an Service Account
14. Deploy an Application
15. Deploy an Application
16. Deploy an Application



Detailed Questions

1. Configure the Identity Provider for the Openshift
   - Create an Htpass Identity Provider with the name: htpass-ex280
   - Create the secret for Identity provider users: htpass-idp-ex280
   - Create the user account jobs with password deluges
   - Create the user account wozniak with password grannies
   - Create the user account collins with password culverins
   - Create the user account adlerin with password artiste
   - Create the user account armstrong with password spacesuits

Answer:

Step -1:

Log onto the workbench node and then execute the below commands.

#ssh root@workbench

Step -2:

Cat the ocp4.config file to know the password for the user kubeadmin

cat /usr/local/etc/ocp4.config

oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443

oc whoami

Step -3:

Install the htpasswd tools, since it's not installed by default

sudo yum install httpd-tools -y

htpasswd

Create a directory to store the user file

mkdir mypass

Create the list of users, but the -c will be used only when create a file afterword's its not required and we are going to append the users into the same file

htpasswd -c -B -b mypass/users jobs deluges

htpasswd -B -b mypass/users wozniak grannies

htpasswd -B -b mypass/users collins culverins

htpasswd -B -b mypass/users adlerin artiste

htpasswd -B -b mypass/users armstrong spacesuits

You can do the cat users file to verify but the password will be encrypted.

cat mypass/users

**** Option to correct if we made any mistake on htpasswd file creation.**

oc extract secret/**htpasswd-secret** -n openshift-config --to /tmp/ --confirm → where **htpasswd-secret** is secret name which we created earlier.

htpasswd -B -b /tmp/htpasswd anna newpassword → we can update the user password

htpasswd -B -b /tmp/htpasswd loria secret → we can create a new user

htpasswd -D /tmp/htpasswd armstring → we can delete the user as well

oc set data secret/htpasswd-secret --from-file htpasswd=/tmp/htpasswd -n openshift-config

→ finally we are updating the htpasswd secret to cluster.

Step -4:
Create the secret and configure it to openshift-config namespace.
oc create secret generic htpass-idp-ex280 --from-file htpasswd=mypass/users -n openshift-config
oc get secret htpass-idp-ex280 -n openshift-config
oc get secret htpass-idp-ex280 -o yaml -n openshift-config → to verify whether the secret is created.
oc get oauth cluster -o yaml → to see any if any providers already configured or not and the spec {} seems empty.

We need to download the yaml script from the product document to do the oauth cluster configuration
open the product link and → (ctrl+f) configure→ authentication & authorization → chapter 4 → section 4.1.5 or (**ctrl+f = smaple HTPasswd CR**).
sample HTpasswd CR

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
spec:
  identityProviders:
  - name: my_htpasswd_provider 1
    mappingMethod: claim 2
    type: HTPasswd
    htpasswd:
      fileData:
        name: htpass-secret 3
```

cat > oauth.yaml → *past the above yaml code into oauth.yaml and press ctrl+d to save it*
vim oauth.yaml → to change the name of the identity provider and the secrete name as well as per the question.

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
spec:
  identityProviders:
  - name: htpass-ex280 1
    mappingMethod: claim 2
    type: HTPasswd
    htpasswd:
      fileData:
        name: htpass-idp-ex280 3
```

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
spec:
  identityProviders:
  - name: htpass-ex280
    mappingMethod: claim
    type: HTPasswd
    htpasswd:
      fileData:
        name: htpass-idp-ex280
```

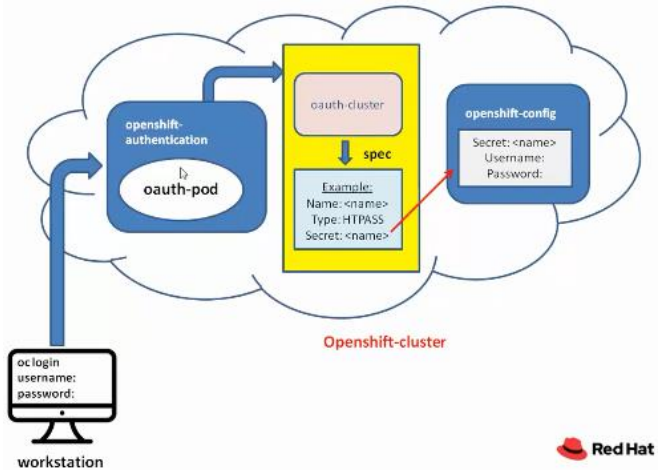
oc get oauth cluster -o yaml
Before you are going to replace the configuration, the openshift-authentication pods should be watch since after the replace the pods will redeploying, so open new tab and do watch the pods by below command.
watch oc get pods -n openshift- *authentication*
oc replace -f oauth.yaml

```
Every 2.0s: oc get pods -n openshift-authentication                    workstation.lab.example.com: Sat Apr 17 10:09:31 2021

NAME                               READY   STATUS        RESTARTS   AGE
oauth-openshift-54b5b94f99-2gp8s   1/1     Running       0          9s
oauth-openshift-54b5b94f99-lstqm   1/1     Running       0          18s
oauth-openshift-68799984ff-m2mjf   1/1     Terminating   0          254d
oauth-openshift-68799984ff-pfrv7   1/1     Terminating   0          254d
```

oc get oauth cluster -o yaml → to verify the configuration on spec {} part.
Step -5:
Verify the user's logon

oc login -u jobs -p deluges
oc whoami
oc login -u armstrong -p spacesuits
oc whoami
oc login -u wozniak -p grannies
oc whoami
oc login -u collins -p culverins
oc whoami
oc login -u adlerin -p artiste
oc whoami
**finally log back as kubeadmin to proceed further next question.*
oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443



2. Configure Cluster permissions
   - User jobs is able to modify the cluster
   - Wozniak is able to create project
   - Amstrong cannot create projects
   - Wozniak cannot modify the cluster
   - Remove the kubeadmin user from the cluster → only on exam not on lab practices

Answers:

Step - 1: *to make the jobs user as cluster admin*
oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443
oc whoami
oc get clusterrolebinnding -o wide |grep jobs → before verify jobs use having any roles
oc adm policy add-cluster-role-to-user cluster-admin jobs
oc get clusterrolebinnding -o wide |grep jobs → after verify the jobs user become a cluster admin

step – 2&3:
oc login -u wozniak -p grannies → *verify the user having ability to create a project*
oc login -u armstrong -p spacesuits → *verify the user having ability to create a project*

By default, all the users having ability to create a project, since because the group self-provisioner, hence we are going to remove the group from all users.

oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443

oc describe clusterrolebinding self-provisioner → to get the group name as system:authenticated:oauth
oc adm policy remove-cluster-role-from-group self-provisioner system:authenticated:oauth
Note: self-provisioner is roles and while removing the group it shows the error msg.

**Verification*
oc login -u wozniak -p grannies → *after removal of the group the users shouldn't able to create project, but this use should have project creation rights as per question requirement,*
oc login -u armstrong -p spacesuits → *after removal of the group the users shouldn't able to create project*

Again, we are going to add that roles into required user as ***wozniak***
oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443

oc adm policy add-cluster-role-to-user self-provisioner wozniak
oc describe clusterrolebinding self-provisioner → verify it again and only wozniak having the rights

**Verification*
oc login -u wozniak -p grannies → *after this user alone into group he should be able to create project*
oc login -u armstrong -p spacesuits → *this user shouldn't able to create project*

step – 4: the user wozniak doesn't have permission to modify the cluster
oc login -u wozniak -p grannies
oc get nodes → it'll showing an error since the user wozniak doesn't have any cluster level permission.

step – 5:
**verify both kubeadmin and jobs user are having cluster rights.*
oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443
oc login -u jobs -p deluges
oc get nodes

Get the project name and delete the kubeadmin user secret file, so that user can't be logon again.
oc get project | grep kube-system → to get the project / namespace.
oc delete secret kubeadmin -n kube-system

**verify the kubeadmin login will show the errors (not authorized).*
oc login -u kubeadmin -p nIMBQ-ZoYZ5-UyS5D-oGTjF https://api.ocp4.example.com:6443

3. Configure Project permissions
    a. Create following projects
        i. apollo
        ii. titan
        iii. gemini
        iv. bluebook
        v. apache
    b. User armstong is admin for the Apollo and Titan project
    c. User Collins is able to view the Apollo project

Answers:

Step- 1:

oc whoami --> we should get the user jobs since it's the only cluster admin user and we already deleted the kubeadmin user.
oc new-project apollo
oc project → verify currently on which project.
oc new-project titan
oc new-project gemini
oc new-project bluebook
oc new-project apache
oc get projects | grep 'apollo\|titan\|gemini\|bluebook\|apache'
→ verify to all the above projects are created.

Step- 2&3:

oc policy add-role-to-user admin armstrong -n apollo
oc policy add-role-to-user admin armstrong -n titan
oc policy add-role-to-user view collins -n apollo

**verification after added the roles to the users.*
oc get rolebinding -o wide -n apollo
oc get rolebinding -o wide -n titan

4. Create Groups and configure permissions
    a. Create a group called commander and user wozniak is the member of this group
    b. Create a group called pilot and user adlerin is the member of this group
    c. The commander group members are able to edit the Apollo and Titan project
    d. The pilot group members are able to view Apollo project but not edit it.

Answers:

Step – 1:

oc adm groups new commander
oc adm groups add-users commander wozniak
oc get groups → verify the group are created and user added into it

Step – 2:

oc adm groups new pilot
oc adm groups add-users pilot adlerin
oc get groups → verify the groups created and user added into it

Step – 3:

oc policy add-role-to-group edit commander -n apollo
oc policy add-role-to-group edit commander -n titan
oc policy add-role-to-group view pilot -n apollo

**verify the project and roles details for the above groups.*
oc get rolebinding -o wide -n apollo
oc get rolebinding -o wide -n titan

5. Configure Quotas for the Project

Create ResourceQuota in manhattan project named ex280-quota

a. The amount of memory consumed across all containers may not exceed 1Gi
b. The amount of CPU across all containers may not exceed 2 full cores.
c. The maximum number of replication controllers does not exceed 3
d. The maximum number of pods does not exceed 3
e. The maximum number of services does not exceed 6

Answers:

Note: if the question they mentioned at least for some values to resources then we need to user the **requests** instead of **limits**.

Step – 1:

oc project manhattan
oc create quota ex280-quota --hard limits.memory=1Gi,limits.cpu=2,replicationcontrollers=3,pods=3,services=6

oc describe resourcequotas ex280-quota → verify the quota details

oc delete resourcequotas ex280-quota → if we made any mistake on the above quota creation, we can user this command deletes it and recreate it.

6. Configure Limits for the Project

Create a Limit Range in the bluebook project name ex280-limits

a. The amount of memory consumed by a single pod is between 100Mi and 300Mi
b. The amount of cpu consumed by a single pod is between 10m and 500m
c. The amount of cpu consumed by a single container is between 10m and 500m with a default request value of 100m
d. The amount of memory consumed by a single container is between 100Mi and 300Mi with a default request value of 100Mi

Answers:

Step - 1:

oc project bluebook
We need to download the yaml file form the product documentation,
open the document → develop → Nodes → ctrl+f (search limit range) or 6.3 → 6.3.2 → create limit range.
Need to copy till copy as mentioned as point # 5.

```
apiVersion: "v1"
kind: "LimitRange"
metadata:
  name: "resource-limits" 1
spec:
  limits:
    - type: "Pod" 2
      max:
        cpu: "2"
        memory: "1Gi"
      min:
        cpu: "200m"
        memory: "6Mi"
    - type: "Container" 3
```

```
      max:
        cpu: "2"
        memory: "1Gi"
      min:
        cpu: "100m"
        memory: "4Mi"
      default: 4
        cpu: "300m"
        memory: "200Mi"
      defaultRequest: 5
        cpu: "200m"
        memory: "100Mi"
```

cat > limits.yaml → past the above codes and press ctrl +d to save it.
vim limits.yaml
We need to remove the default part and keep the defaultRequest part on container, since the question they requested only the default requesters alone.

```
apiVersion: "v1"
kind: "LimitRange"
metadata:
  name: " ex280-limits  " 1
spec:
  limits:
    - type: "Pod" 2
      max:
        cpu: "500m"
        memory: "300Mi"
      min:
        cpu: "10m"
        memory: "100Mi"
    - type: "Container" 3
      max:
        cpu: "500m"
        memory: "300Mi"
      min:
        cpu: "10m"
        memory: "100Mi"
      defaultRequest: 5
        cpu: "100m"
        memory: "100Mi"
```

oc project → verify the current project as bluebook

Create the limitrange
oc create -f limits.yaml

**verify the limitrnge*
oc describe limitranges
###oc delete limitranges ex280-limits → if we did any mistake, we can delete the limit range and create it by using the command.

7. Deploy an Application

Deploy an application a called rocky in bluewills project

a. The application should be reachable from the following url:
http://rocky.apps.ocp4.example.com

b. You should get valid Output

Note:

From question #7 onwards all are scenario-based questions and we need to follow the layered approach to do the troubleshooting → like application layer → service layer→ route layer → finally client

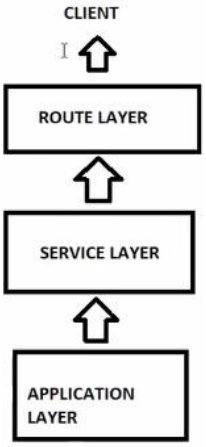
Note: not for exam, only on Lab:

cd Downloads/Scenario_v1.0/
sh startSCENARIO.sh
cd

Answers:

oc project bluewills
oc get all
oc get pod → pod is in pending state
oc logs rocky-564dc5fd7c-v8tv2 → logs will not help us since the pod is not yet deployed
oc describe pod rocky-564dc5fd7c-v8tv2
oc get events
oc get nodes

oc describe node master01 | grep Taints → it will show you node=worker:noschedule
oc describe node master02 | grep Taints → it will show you node=worker:noschedule
oc describe node master03 | grep Taints → it will show you node=worker:noschedule
Do watch window before remove the taint condition by running the watch window
watch oc get pod
oc describe node master02 | grep Taints
oc adm taint node master01 node-
→ node can be any keyword like compute/ something, so whatever we need to use that keyword
**verification*
oc describe node master01 | grep Taints
oc describe node master02 | grep Taints
oc describe node master03 | grep Taints
oc get nodes
oc adm taint node master02 node- → it will remove taint condition
oc get nodes → will show you what are the nodes are master / workers



**We need to do on only on worker nodes not in the master
#oc adm taint node worker01 node- (in exam)
#oc adm taint node worker02 node- (in exam)
**After remove the taint on application layer still the issue existing. We may need to lookout the service layer.*
oc get service
oc describe service rocky
**We have look at the endpoints: <>, if it had IP address then the service layer seems ok. We may look at the router layer.*

if you run the below command, it'll show you the labels which is for the deployment and the same in-service selector as well. Then only the service will work and we can't add the endpoint ip address by manually.

oc describe service rocky | grep Selector

oc describe pod rocky-564dc5fd7c-v8tv2 | grep Label → only if the service not shown the endpoints details, then only we need to check these 2 points.

oc get route → *the web url is wrong we have to fix it by removing the route and add it again.*

oc delete route rocky

oc get route → verify the route is deleted

oc get service → to see the service name

oc expose service rocky --hostname=rocky.apps.ocp4.example.com → *this is not secure app so we can expose to create a route as a non-secure.*

oc get route → verify the right route is created.

**verify the web url link on browser to see the application reachable.*

8. Configure and Deploy a secure route

Deploy an application called oxcart securely in the project called area51

a. The application has self-signed certificate available at

"/C=US/ST=NC/L=Raleigh/O=RedHat/OU=RHT/CN=oxcart.apps.ocp4.example.com "

b. The application should be reachable at the following url
https://oxcart.apps.ocp4.example.com

c. Application produces a valid Output

Note: which methods we are going to choose.

1. We are going to use the edge termination → choosing this one.
2. Passthrough termination we have put the certificate inside the application, in the question they are mentioned the location of placing the certifications.
3. Re-Encryption required 2 certificates, but in the question, they mentioned only one. So, this also not applicable

Answer:

oc project area51

oc get pods

oc get route → it'll not shown any termination type

oc delete route oxcart → since we are going to create new secure route using the certificate

oc get route

```
1. edge termination
2. passthrough termination
3. reencryption ""
```

mkdir cert

cd cert/

Generate Private Key

openssl genrsa -out oxcart.key 2048

ls

Generate CSR

openssl req -new -key=oxcart.key -out oxcart.csr -subj="/C=US/ST=NC/L=Raleigh/O=RedHat/OU=RHT/CN=oxcart.apps.ocp4.example.com"

ls

Generate Certificate

openssl x509 -req -days 365 -signkey oxcart.key -in oxcart.csr -out oxcart.crt

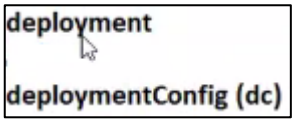
ls

oc get service → to get the service name and based on that we are going to create secure route.

oc create route edge --service=oxcart --cert=oxcart.crt --key=oxcart.key
--hostname=oxcart.apps.ocp4.example.com
oc get route
verify the web url via browser whether the link is working and it'll produce the output, make sure* ***https://<>*

9. Scale the Application manually

Scale an application called hydra in the project called lerna
The hydra application should be scaled to five times



Note: Type of deployment is 2 different type as showing as in picture. →

Answers:

oc project lerna
oc get pods
oc get all | grep deploy → we need to know whether they deployed as deployment / deployment-config.

oc scale --replicas=5 deployment.apps/hydra

10. Configure Autoscaling for an Application

Configure an autoscaling for the scala application in the project gru with following specification

a. Minimum number of replicas: 6
b. Maximum number of replicas: 40
c. Threshold CPU-Percentage: 60
d. Application resource of CPU Request: 25m
e. Application limits of CPU Limits: 100m

Answer:

oc project gru
oc get pods
oc get all | grep deploy → to get the deployment name.
oc autoscale --min=6 --max=40 --cpu-percent=60 deployment.apps/scala
oc get hpa → verify the autoscal
oc describe deployment scala | grep Limits → to verify if is there any limits already created
oc set resources --requests cpu=25m --limits cpu=100m deployment.apps/scala
oc describe deployment scala | grep -A3 Limits → verify the limits are created.
**verification*
 # oc get pods → it should show 6 pods

11. Configure a Secret

Configure a secret in the math project and the name of secret should be magic.
The secret should have following key value pairs
Decoder_Ring: ASDA142hfh-gfrhhueo-erfdk345v

Answer:

oc project math
oc create secret generic magic --from-literal Decoder_Ring=ASDA142hfh-gfrhhueo-erfdk345v
→ we need to use the = not :
oc get secret magic -o yaml → verify the secret on yaml output.

12. Use the Secret value for Application Deployment

Configure the environmental variable for the application called qed in the math project so that it uses the secret "magic"

After configuring the environmental value for the application, it should stop producing the following output

"App is not configured properly"

Answer:

oc project
oc get pods
oc get route
oc get all | grep deploy
oc describe deployment qed | grep Environment
curl qed.apps.ocp4.example.com
oc get secret magic
oc set env --from secret/magic deployment.apps/qed
oc describe deployment qed | grep -A1 Environment
curl qed.apps.ocp4.example.com

13. Configure a Service Account

- Create a service account called ex-280-sa in the project called apples
- This service account should able to run application with any user id.

Answer:

oc project apples
oc create serviceaccount ex-280-sa
oc get serviceaccounts ex-280-sa
oc adm policy add-scc-to-user anyuid -z ex-280-sa → SCC – means **s**ecure **c**ontact **c**onstrains.
oc get clusterrolebindings -o wide | grep ex-280-sa

14. Deploy an Application

Deploy an application called oranges in the project called apples

a. This application should use the service account ex-280-sa
b. The Application should produce a valid output

Answer:

oc project
oc get pods
oc logs oranges-bc578f98d-j7nhj
oc get all | grep deploy
oc set serviceaccount deployment.apps/oranges ex-280-sa
oc get route → after getting the url we have verified whether is working or not via browser. (it'll not)
oc get service → hence we need to check the service layer level everything seems ok or not, found that the service is not having endpoints. Because there is a label miss match.

oc describe service oranges
oc get pods -o wide
oc describe service oranges | grep Endpoints
oc describe service oranges | grep Selector → to see the selector label details
oc describe pod oranges-7849dcbd68-fm8zf | grep Labels → to see the label details, here is the miss match.
oc edit service oranges → edit the selector name as "oranges" → under the spec we can able to see the selector and deployment label should be "oranges" and save the file :wq
**verify the selector, labels & endpoints are ok.*
oc describe service oranges | grep Selector
oc describe pod oranges-7849dcbd68-fm8zf | grep Labels
oc describe service oranges | grep Endpoints
oc get pods -o wide
**verify the web url on browser to ensure the content/app is visible*

15. Deploy an Application

Deploy an application called voyager in the project path-finder

a. Don't add any new configuration
b. Application should produce a valid output

Answer:

Note: the issue is router and node selector, since because it's using the ingress controller, so many times you are deleting the router it'll not delete means it'll create a another one immediately. Hence, we need to edit the ingress controller for the modifications.

oc project path-finder
oc get pods → pod is a pending state.
oc get events → we'll get an error didn't match the node selectors.
oc describe pod voyager-5b7bf5599-qf4z9 | grep Node-Selector → to get the node selector details
oc get nodes --show-labels → to see the nodes labels
oc get all | grep deploy
oc edit deployment.apps/voyager → modify the labels on deployment config goto end spec { } → template → spec { } → containers → nodeselector.
oc get route → to get the url
curl <url > → it'll not reachable and showing naming resolution issue, since the name resolution FQDN is wrong on the url.
oc get service
oc describe service voyager → verified the services are showing proper endpoints so nothing from that service layer
oc get route
oc describe route voyager-p2fvb → requested host: is wrong values and it'll suggest the right FQDN
oc get route -n apples → verify the previous project routes.
oc get route
oc delete route voyager-p2fvb
oc get route → even you deleted it'll re-appear, since it's created through the controller ingress
oc get ingress
oc edit ingress voyager → goto spec { } → rules: → -host : <change the right hostname>
oc get route

curl voyager.apps.ocp4.example.com
**Other projects they were not created the ingress controller. The below commands for only verification purpose only.*
oc project apples
oc get ingress

16. Deploy an Application

Deploy an application called mercury in the project atlas

a. Don't add any new configuration
b. Application should produce a valid output

Answer:
oc project atlas
oc get pods
oc get events → the deployment is failed because of the insufficient memory.
oc get all | deploy
oc get all | grep deploy
oc describe deployment mercury | grep -A1 Requests → its requesting 80Gi of memory, its not limit and its just requesting. So, we can change it smaller value to start the application based on trial-and-error method. Since the goal is to bring the application active.
oc describe node master01 → to see the resource details on the node.
oc describe deployment mercury | grep -A1 Requests
oc set resources --requests memory=256Mi deployment mercury → watch oc get pods
oc describe node master01
oc get route
curl mercury.apps.ocp4.example.com