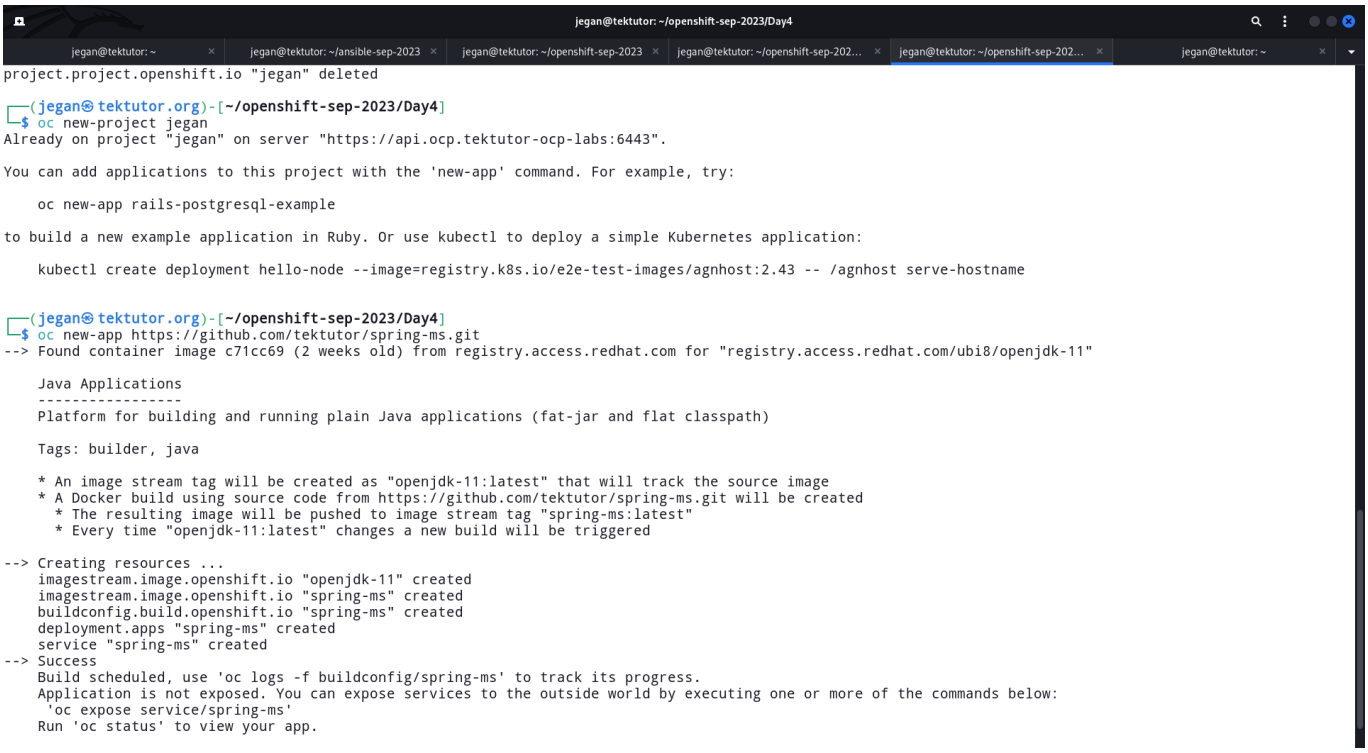
Day 4

Lab - Deploying an application using GitHub Repo source code using Docker strategy

```
oc project jegan
oc new-app https://github.com/tektutor/spring-ms.git
oc expose service/spring-ms
```

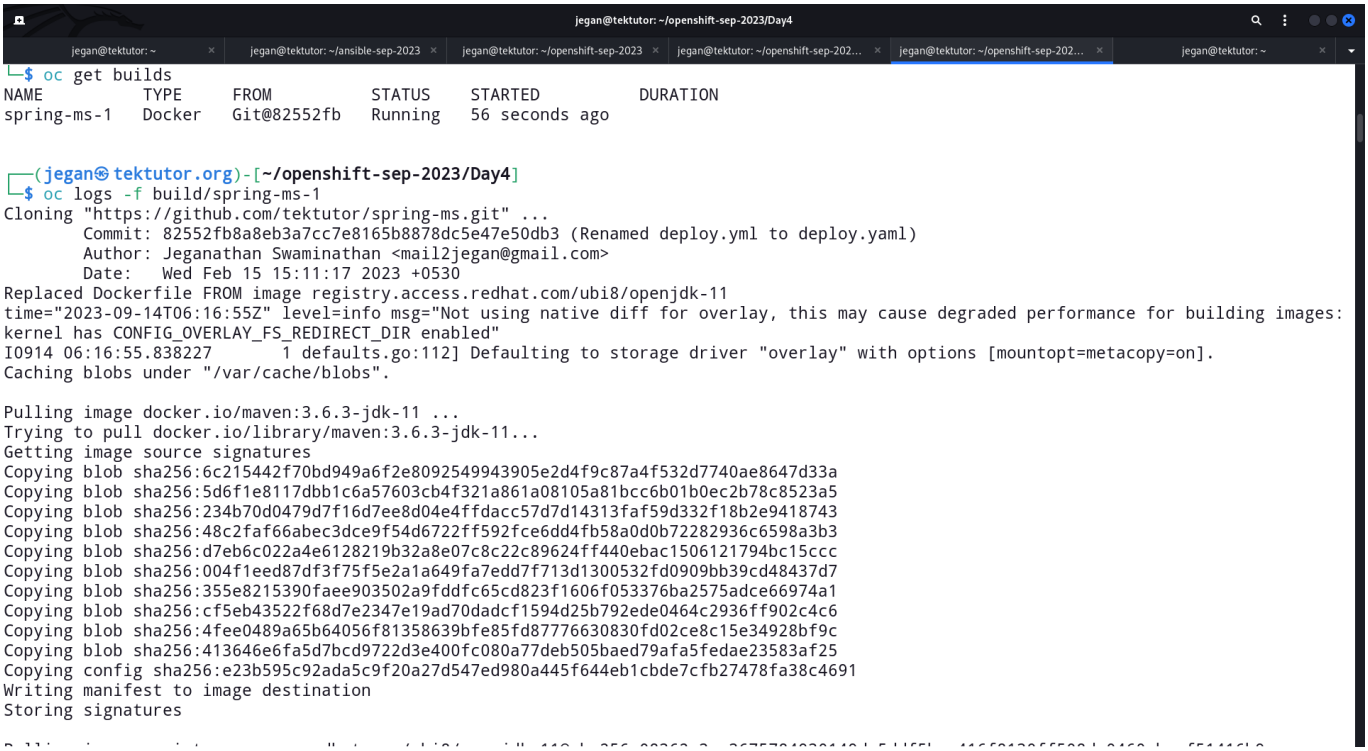
Expected output

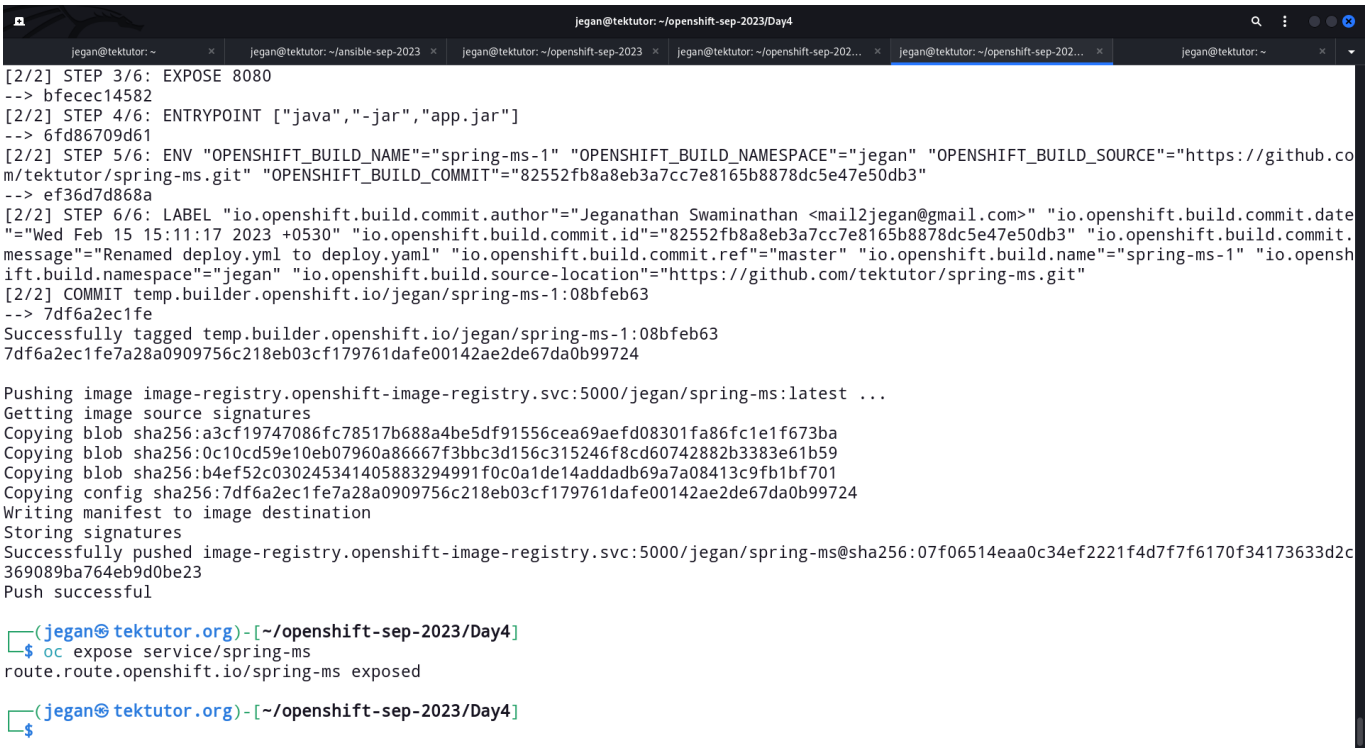


To check the build logs in command line, you can try this

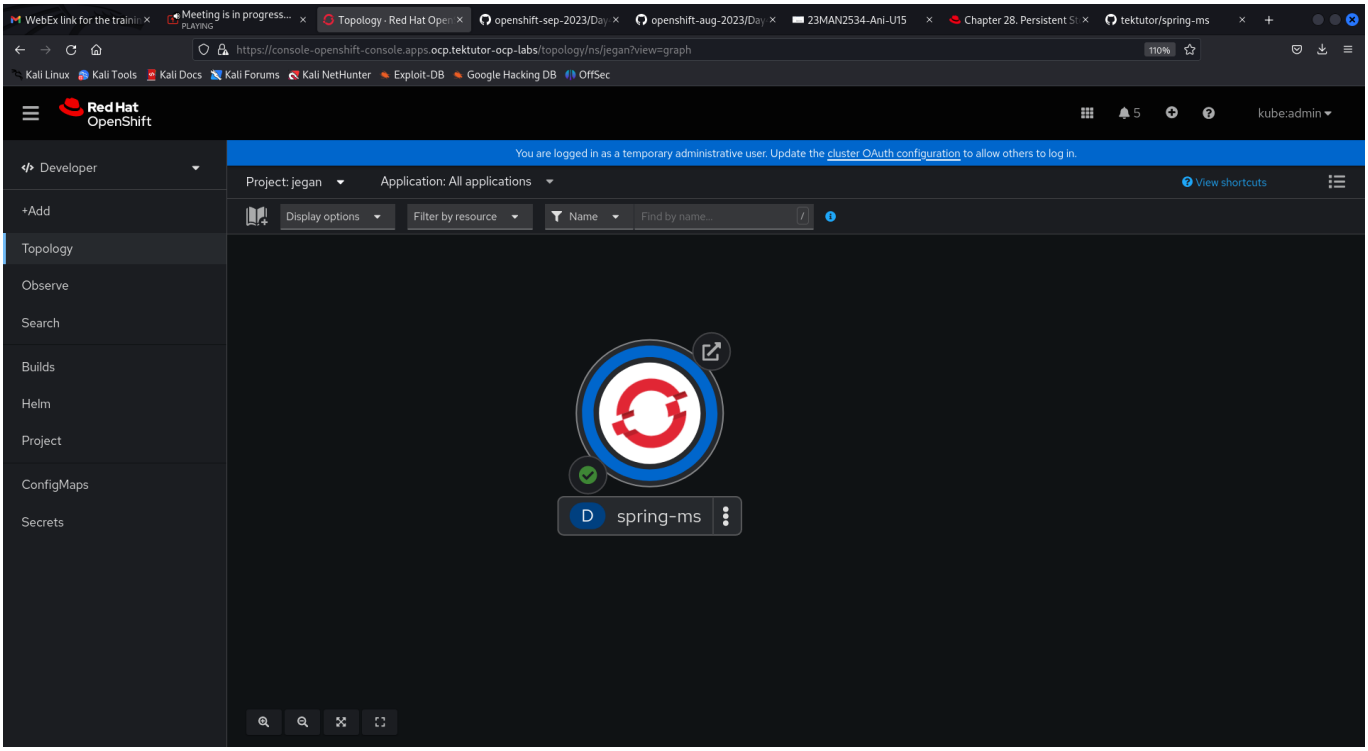
```
oc logs -f build/spring-ms-1
```

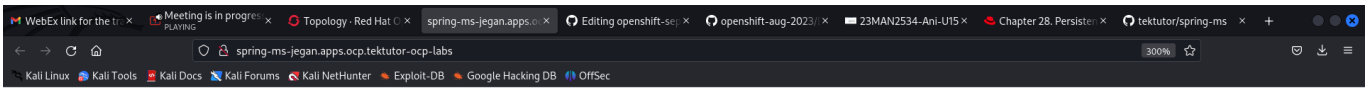
Expected output





You can check the Developer view Topology





Hello Microservice !

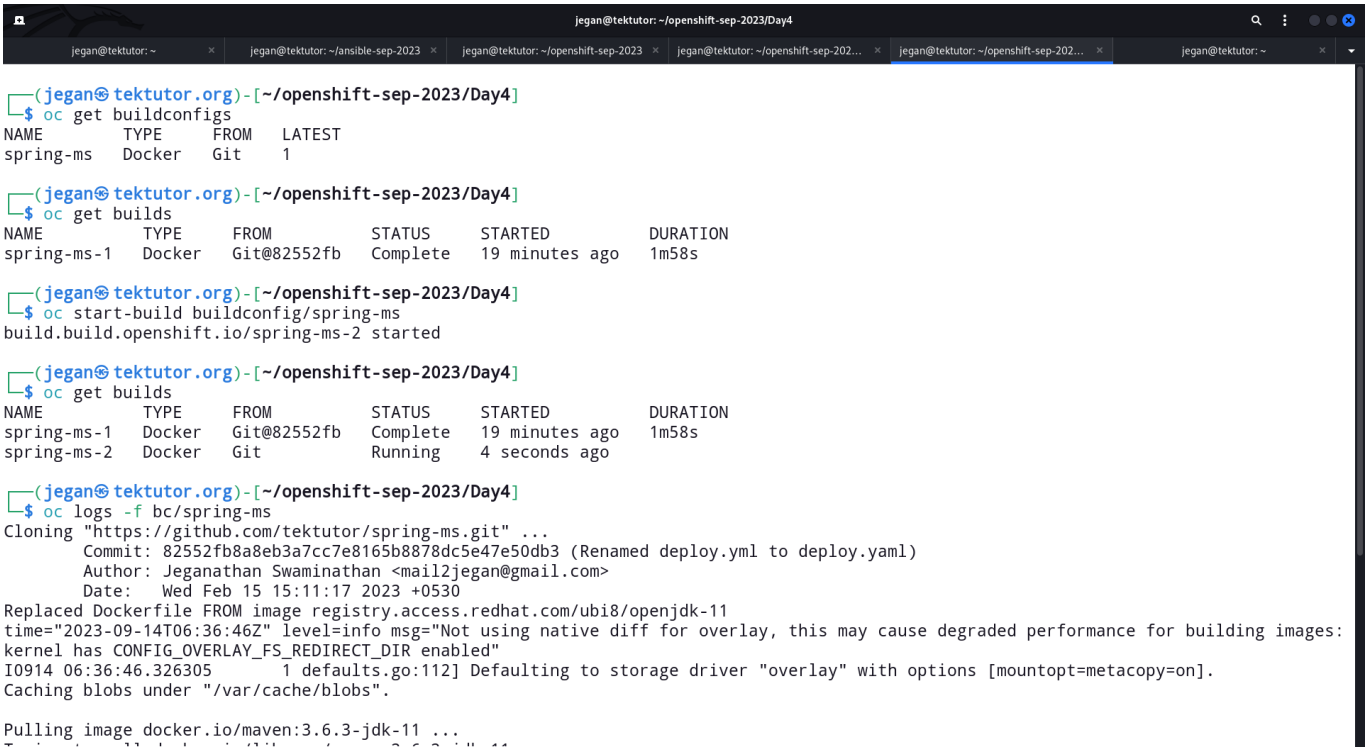
Lab - Starting a build from command-line using buildconfig

```
oc get buildconfigs
oc get buildconfig
oc get bc
```

Start a build from buildconfig

```
oc start-build bc/spring-ms
oc logs -f bc/spring-ms
```

Expected output



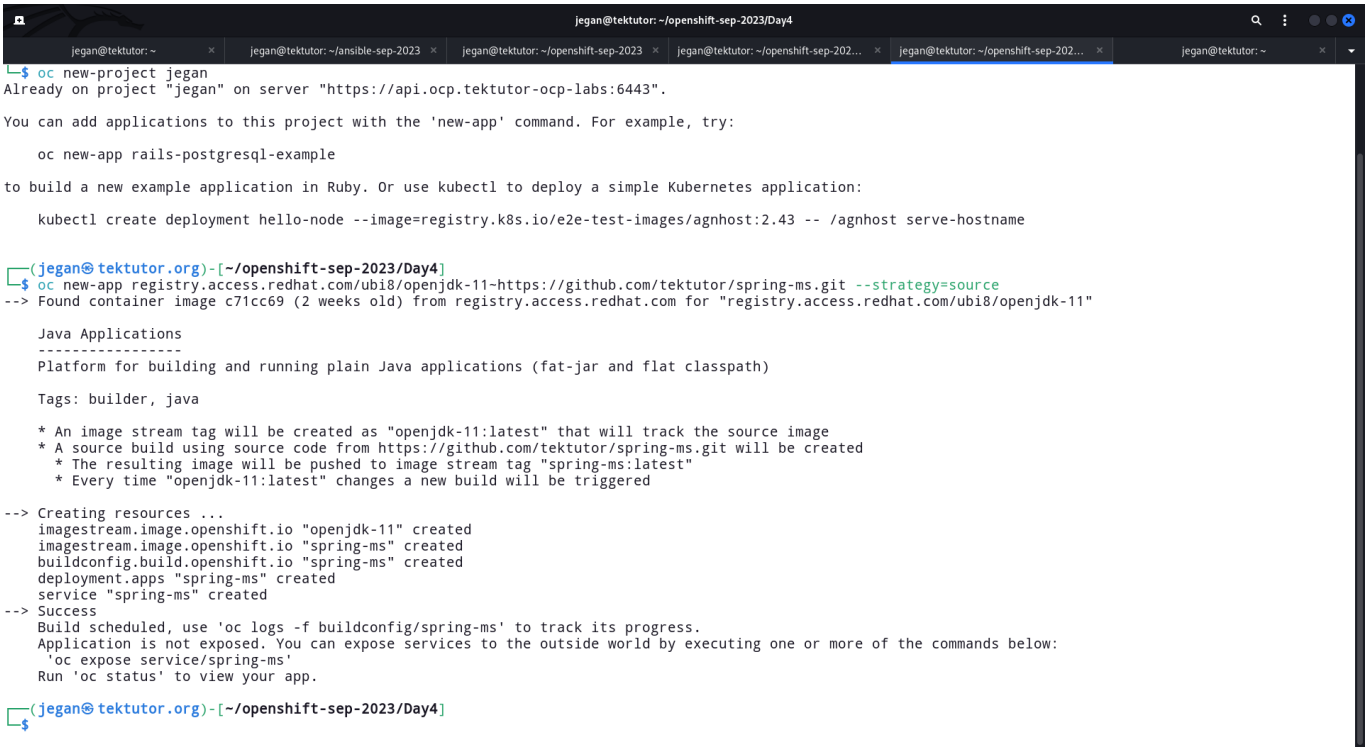
Create a route for spring-ms application deployment

```
oc get svc
oc expose svc/spring-ms
```

Lab - Deploying our custom application into OpenShift using source strategy

```
oc delete project/jegan
oc new-project jegan
oc new-app registry.access.redhat.com/ubi8/openjdk-
11~https://github.com/tektutor/spring-ms.git --strategy=source
```

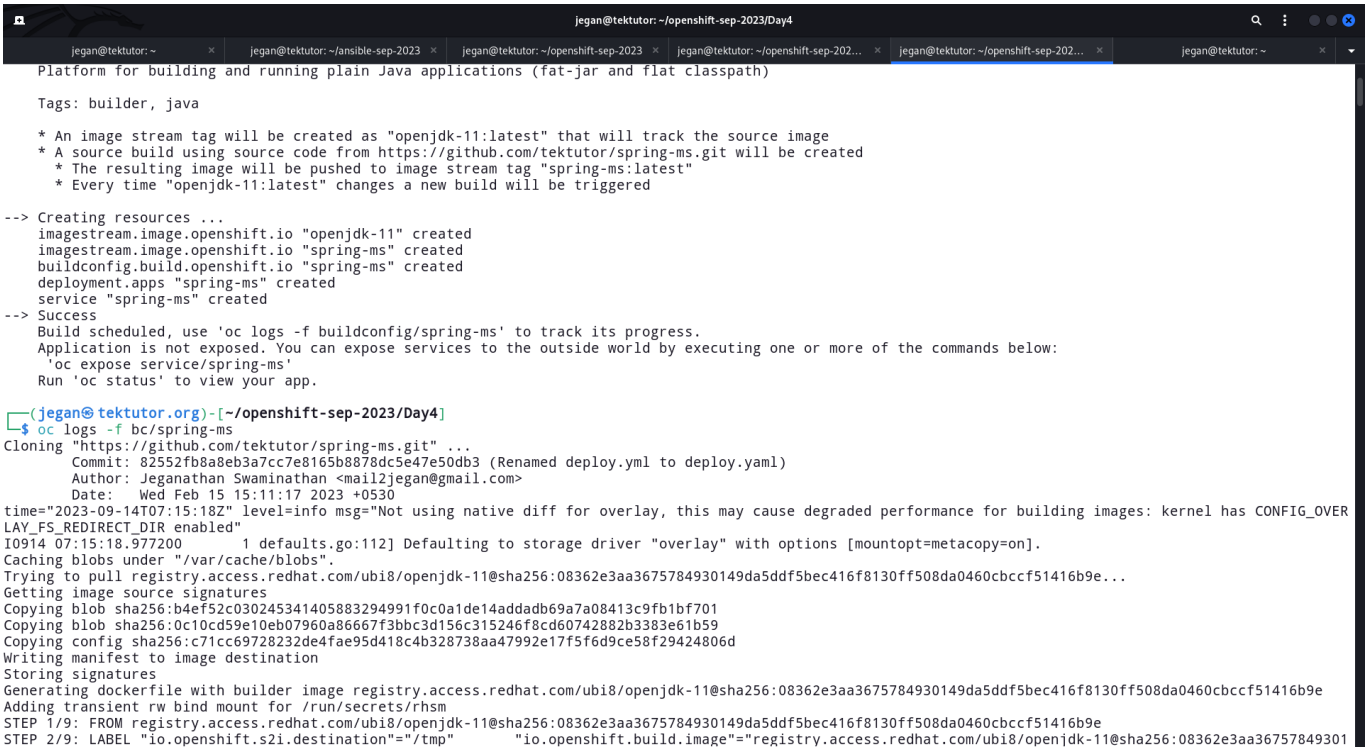
Expected output

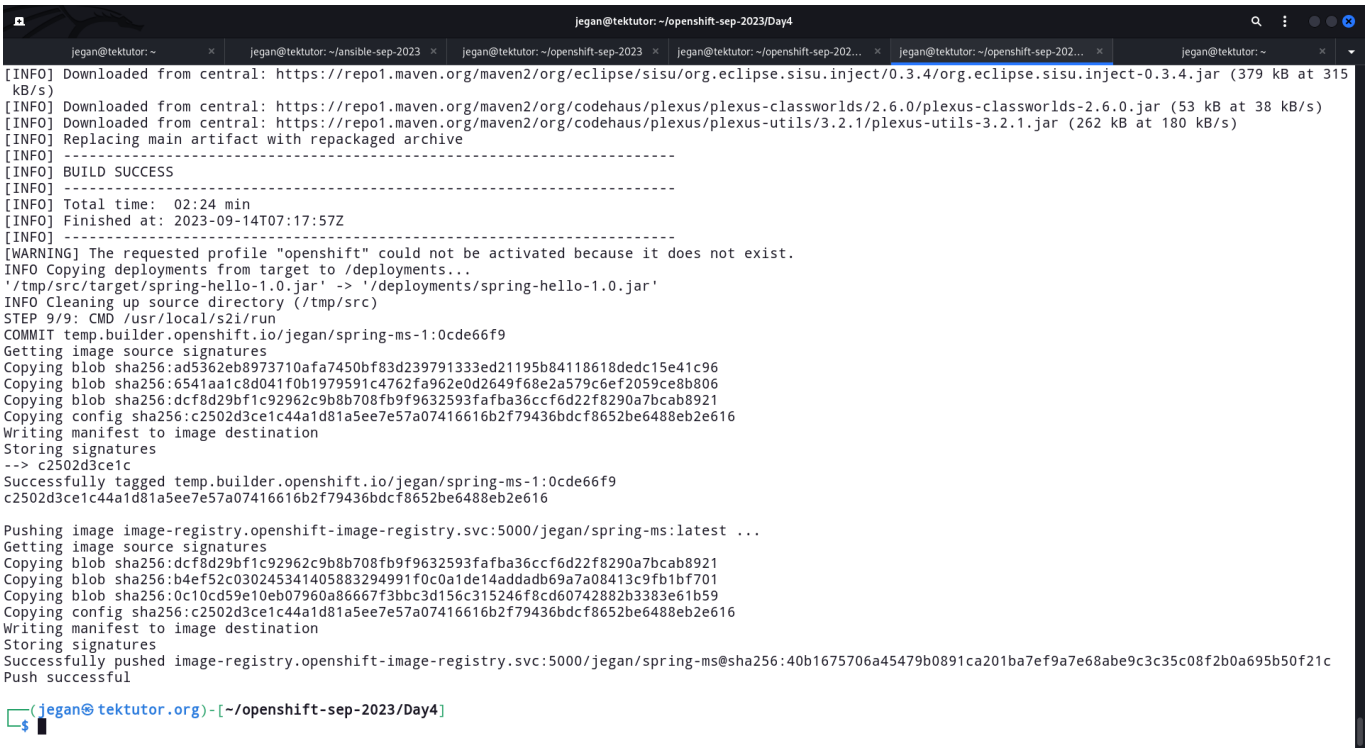


Checking the build log

```
oc logs -f bc/spring-ms
```

Expected output

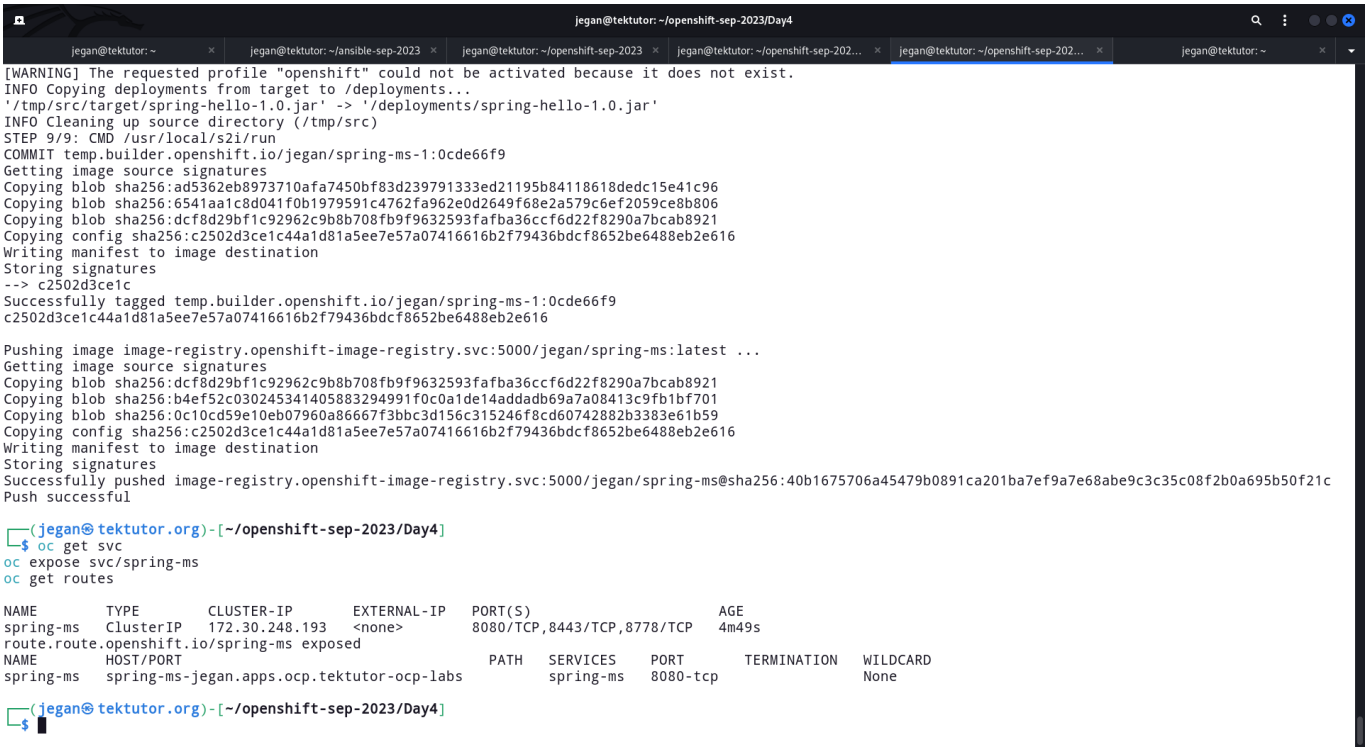




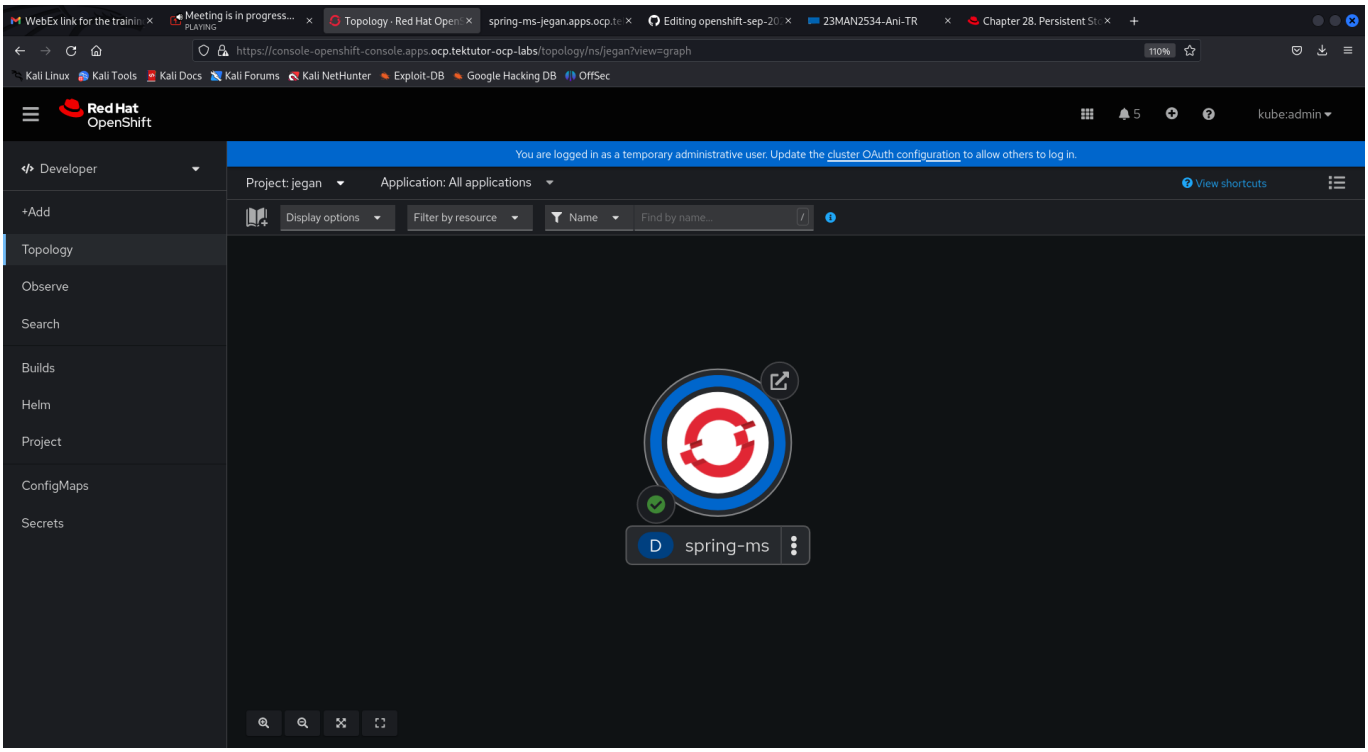
Let's create a route for the spring-ms service

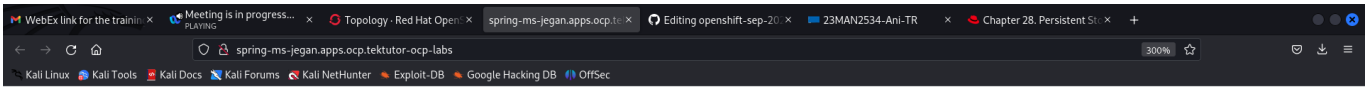
```
oc get svc
oc expose svc/spring-ms
oc get routes
```

Expected output



Now you may access the route from Developer context Topology from your web browser on the CentOS Lab machine.



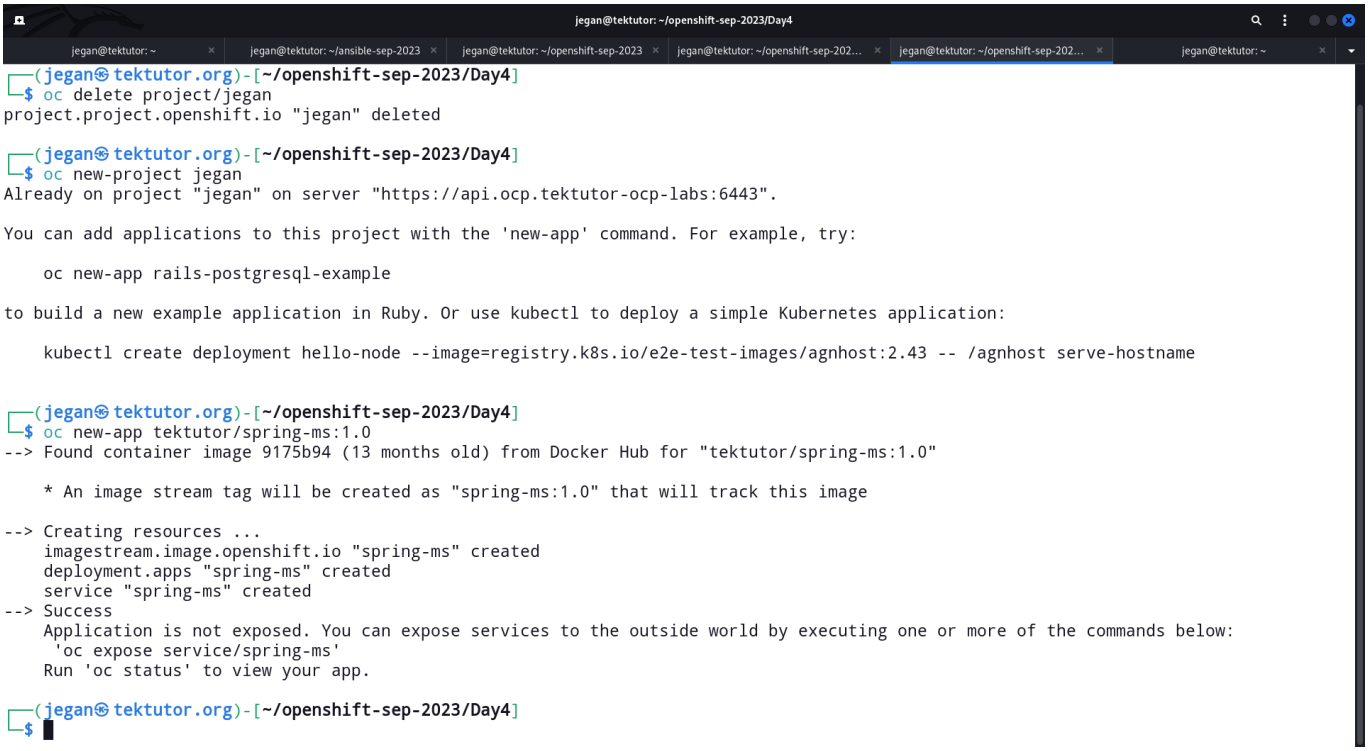


Hello Microservice !

Lab - Deploying custom appling using Docker Hub custom image

```
oc delete project/jegan
oc new-project jegan
oc new-app tektutor/spring-ms:1.0
```

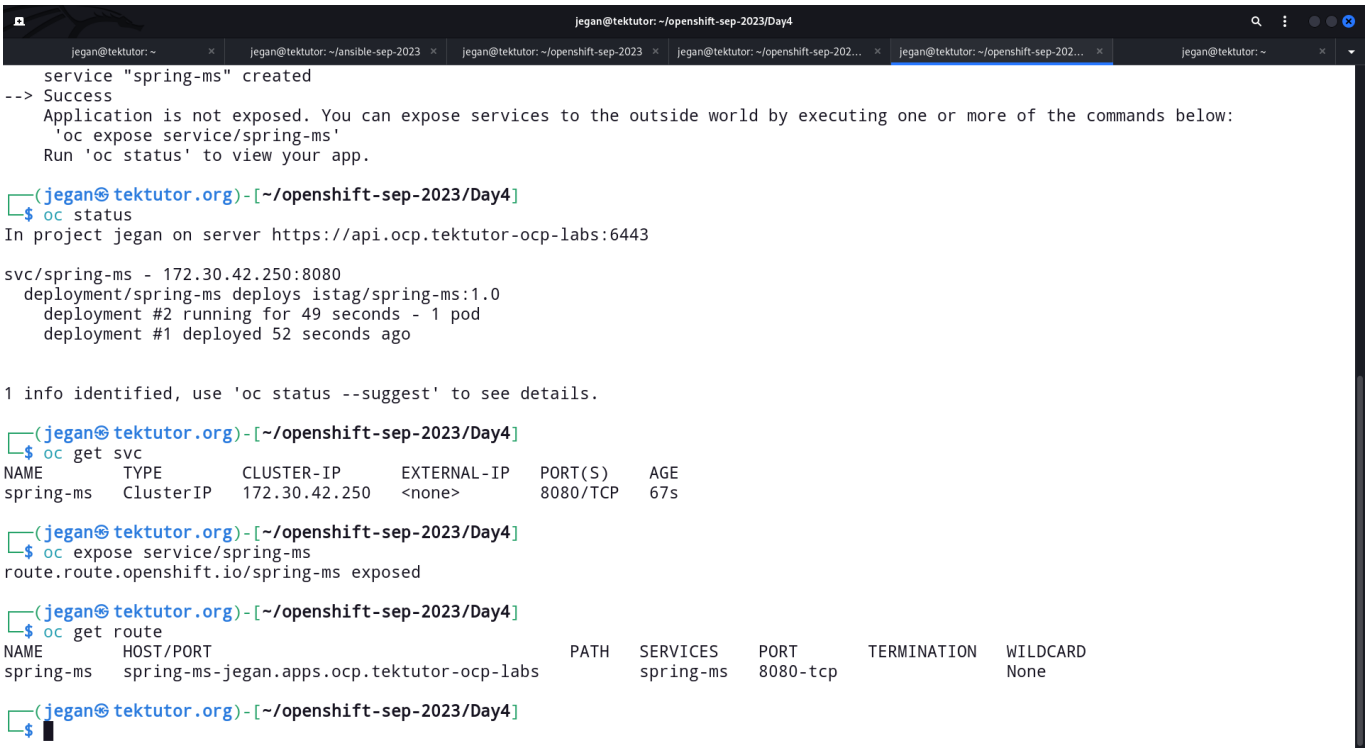
Expected output



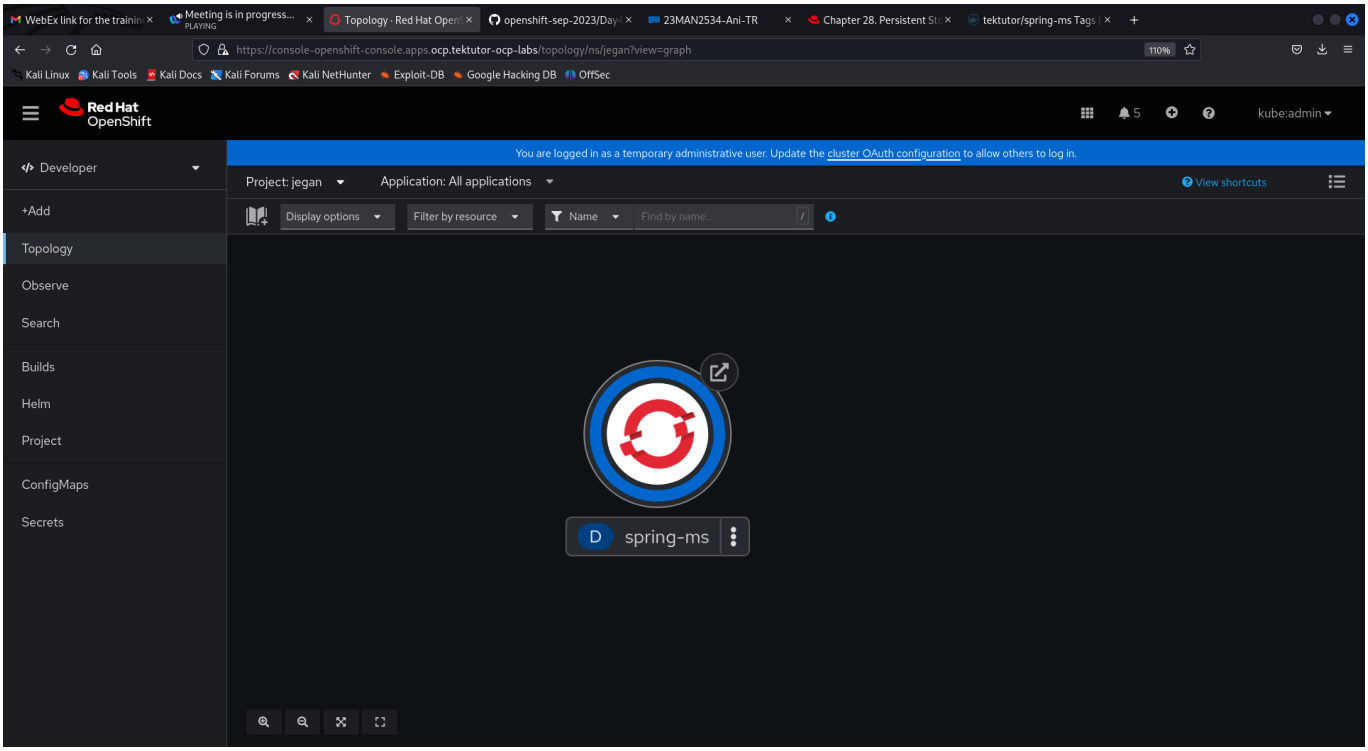
Let us check the deployment status

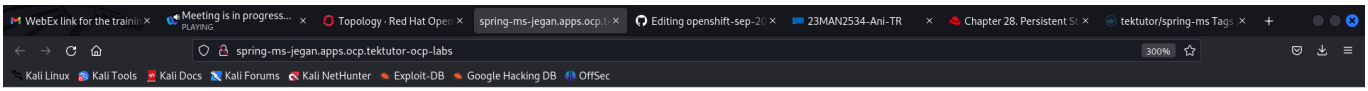
```
oc status
oc get svc
oc expose svc/spring-ms
oc get route
```

Expected output



Now you may access the application using its route url from the OpenShift webconsole





Greetings from Spring Boot!

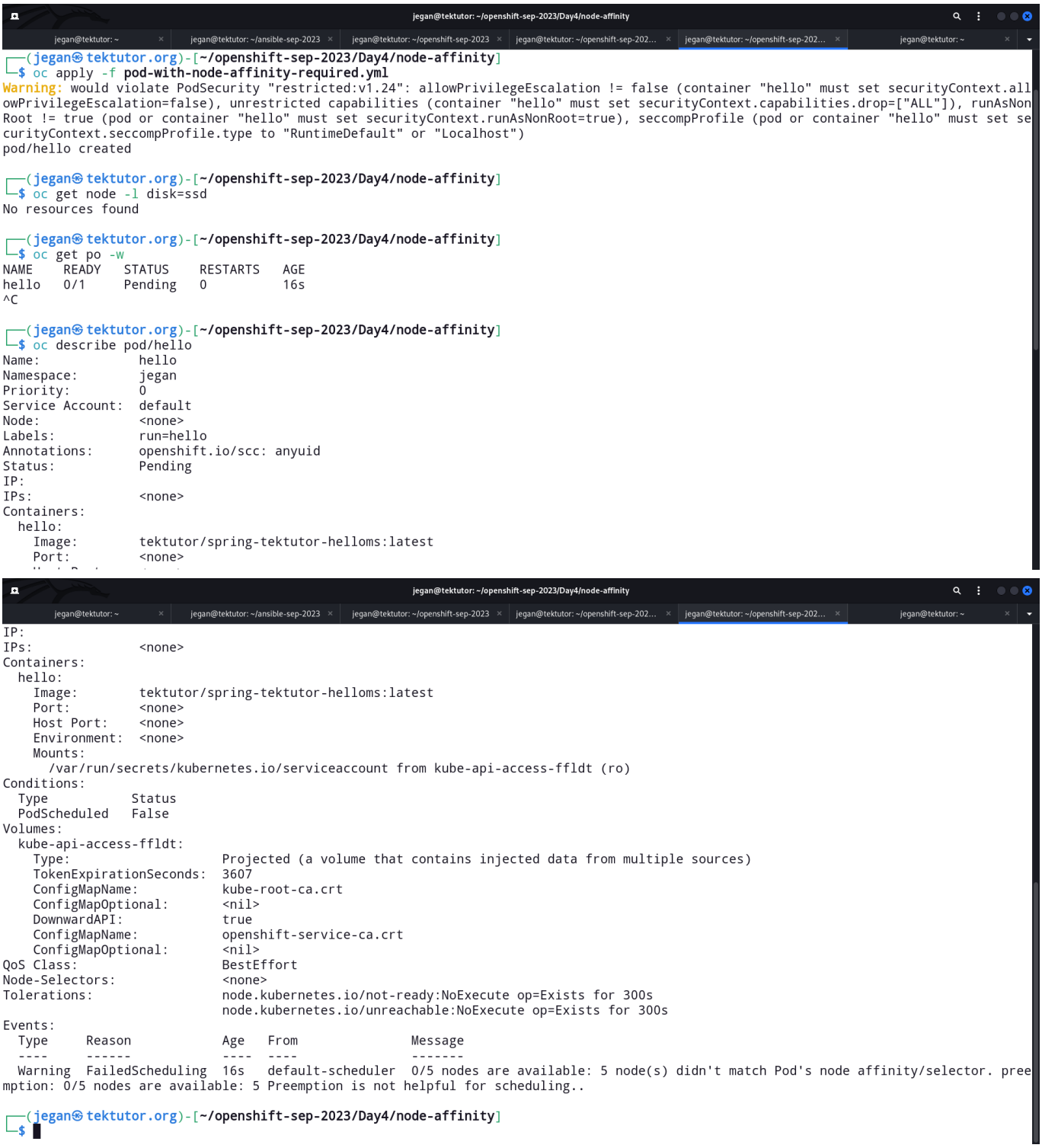
Lab - Node Affinity Required criteria

When creating the pod, we have added some criteria for the default-scheduler to follow. When the criteria is required, the scheduler will look for nodes that has a label matching "ssd" type disk, in case it isn't able to find a node that has label disk=ssd then the Pod will not be deployed.

```
cd ~/openshift-sep-2023
git pull
cd Day4/node-affinity
oc apply -f pod-with-node-affinity-required.yml
```

```
oc get po
oc describe pod/hello
```

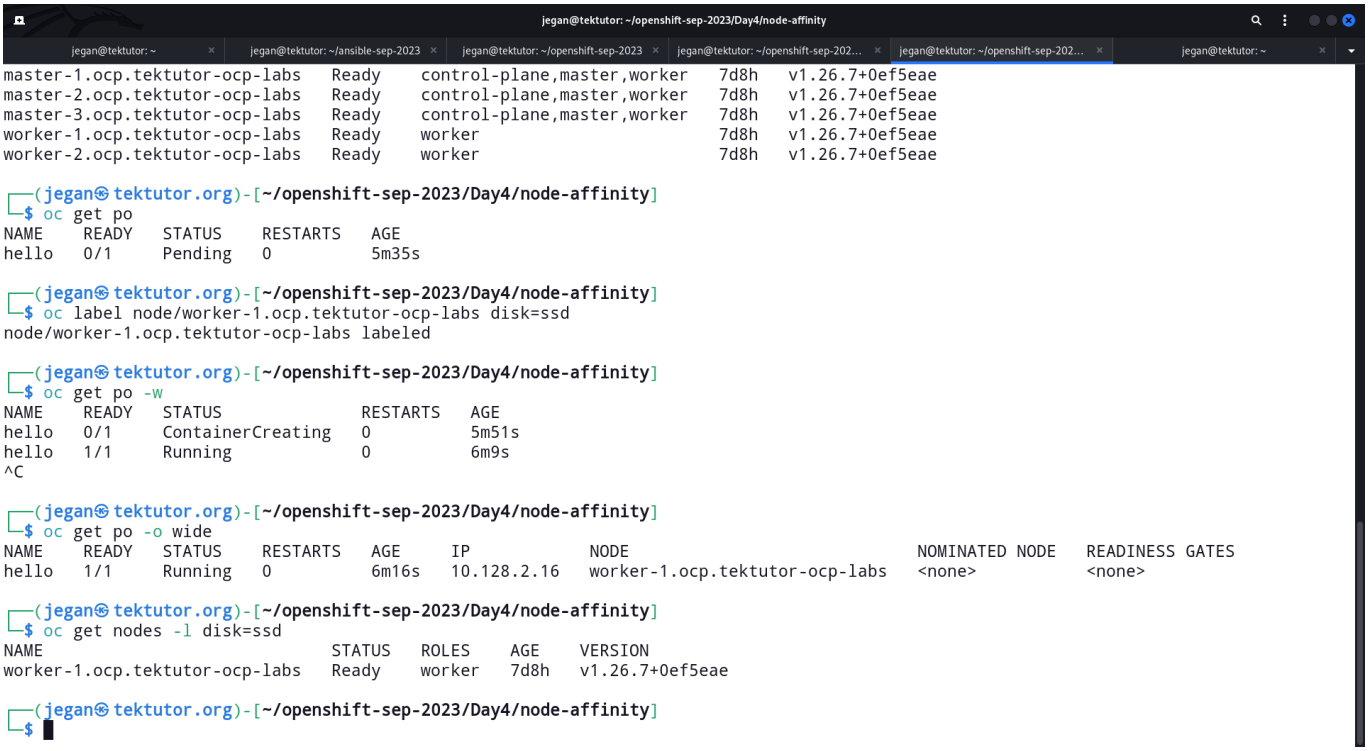
Expected output



Now let's add the label to worker-1 node and see the Pod getting deployed into worker-1.

```
oc label node/worker-1.ocp.tektutor-ocp-labs disk=ssd
oc get po -w
oc get po -o wide
oc get nodes -l disk=ssd
```

Expected output



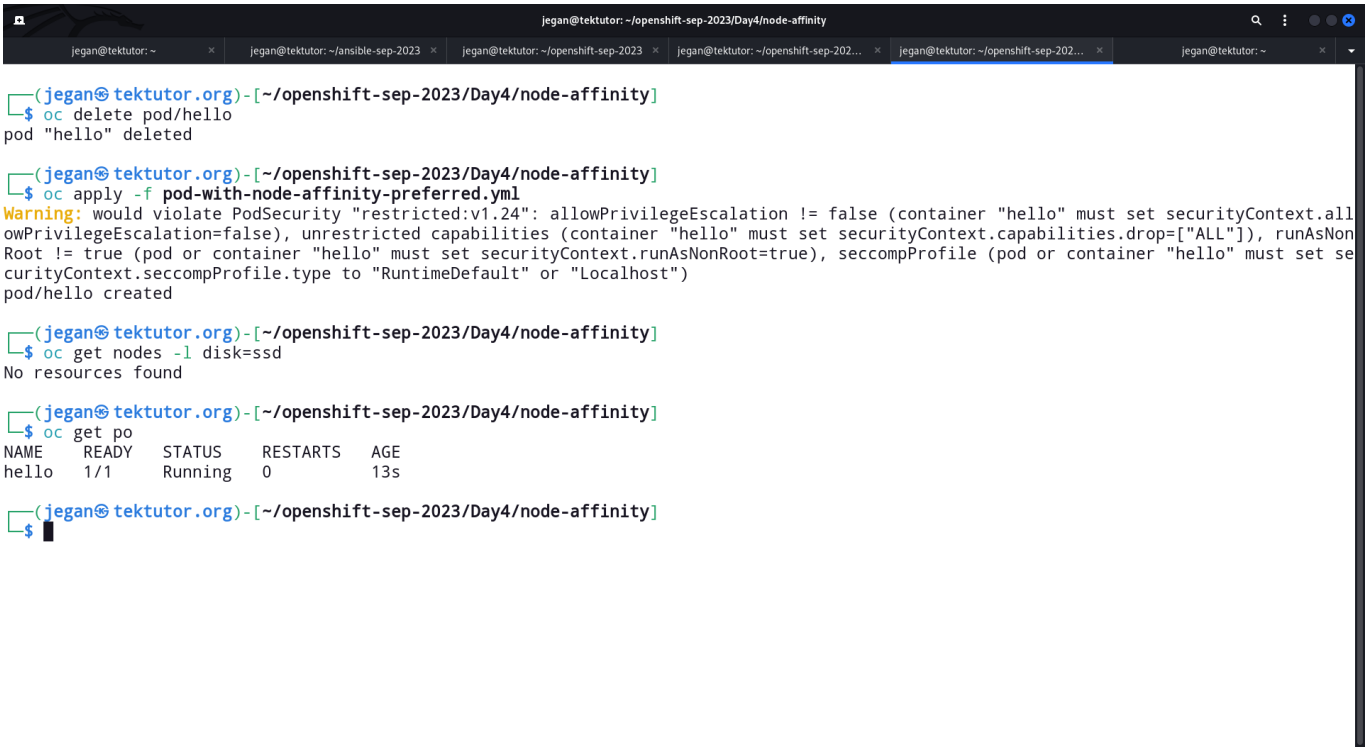
Lab - Pod manifest has a preferred criteria i.e node has label disk=ssd

How the node affinity works in case of Preferred ?

- The scheduler will search for nodes that has label disk=ssd, if it is able to find a node that has the label then the Pod will be scheduler there
- If the scheduler isn't able to find such a node, the scheduler will deploy it on any node as per scheduler's choice

```
cd ~/openshift-sep-2023
git pull
oc delete pod/hello
cd Day4/node-affinity
oc apply -f pod-with-node-affinity-preferred.yml
oc get nodes -l disk=ssd
oc get po
```

Expected output



Info - Docker Network Model

Info - OpenShift/Kubernetes Network Model

Post Test Link

https://app.mymapit.in/code4/tiny/yTefEw

Feedback link :

https://tcheck.co/HgqK27

Installing OpenShift on your laptop

https://developers.redhat.com/products/openshift-local/overview

You need to login to your RedHat account to download crc compressed file and pullsecret files.

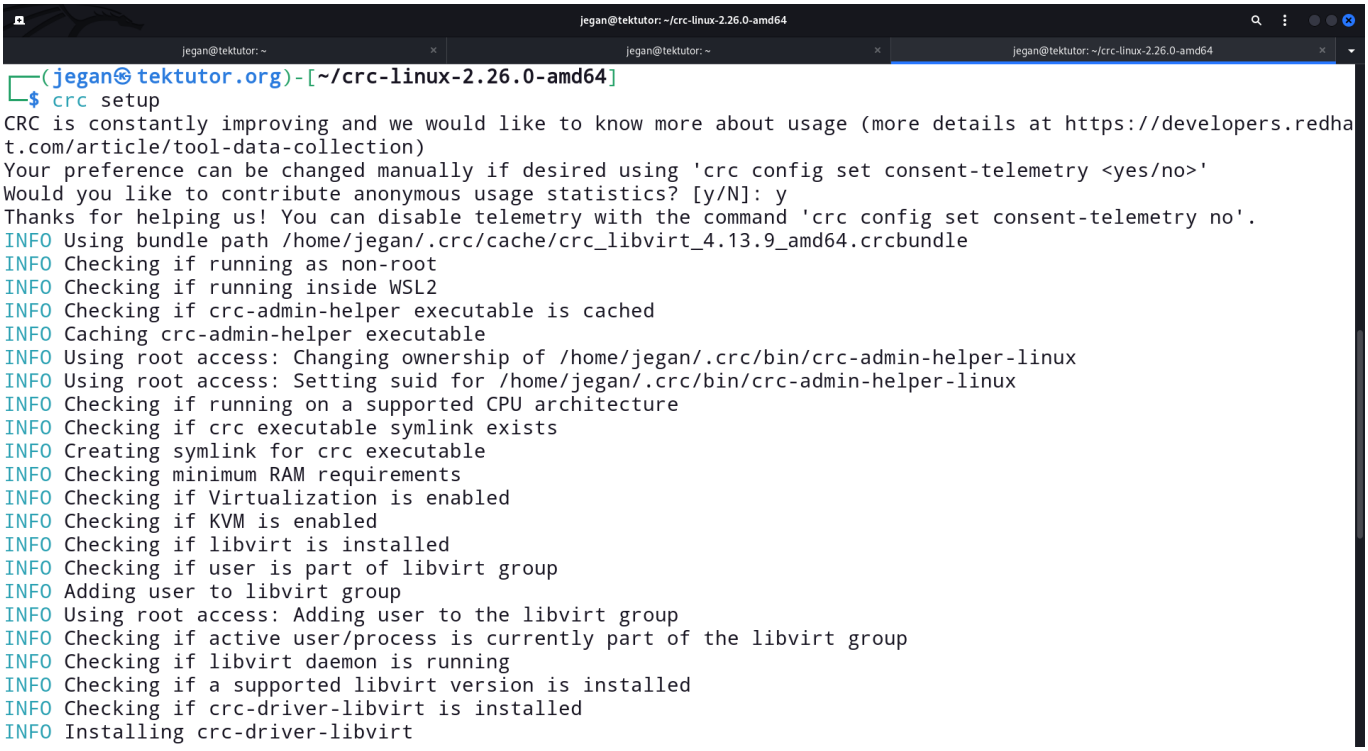
You need to extract it

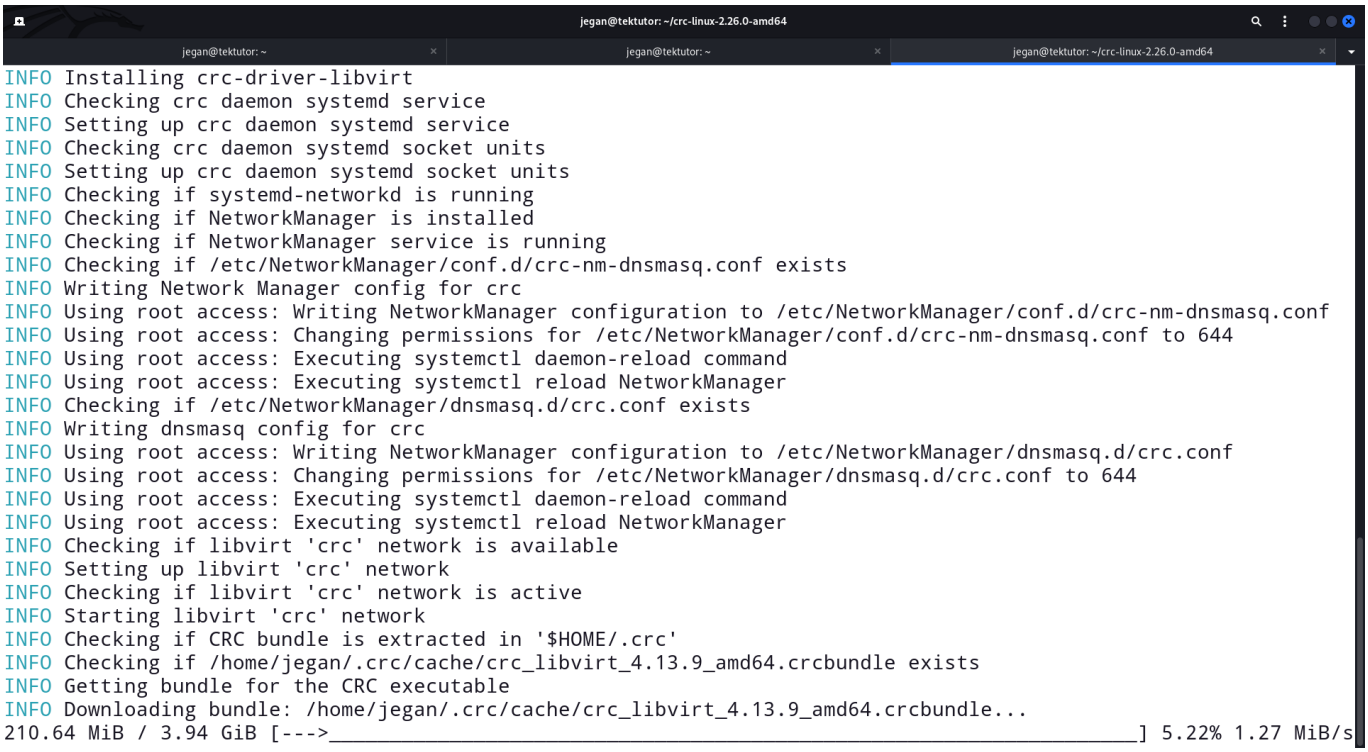
```
cd ~/Downloads
mv crc-linux-amd64.tar.xz ..
cd ..
tar xvf crc-linux-amd64.tar.xz
cd crc-linux-amd64.tar.xz
sudo cp crc /usr/bin
```

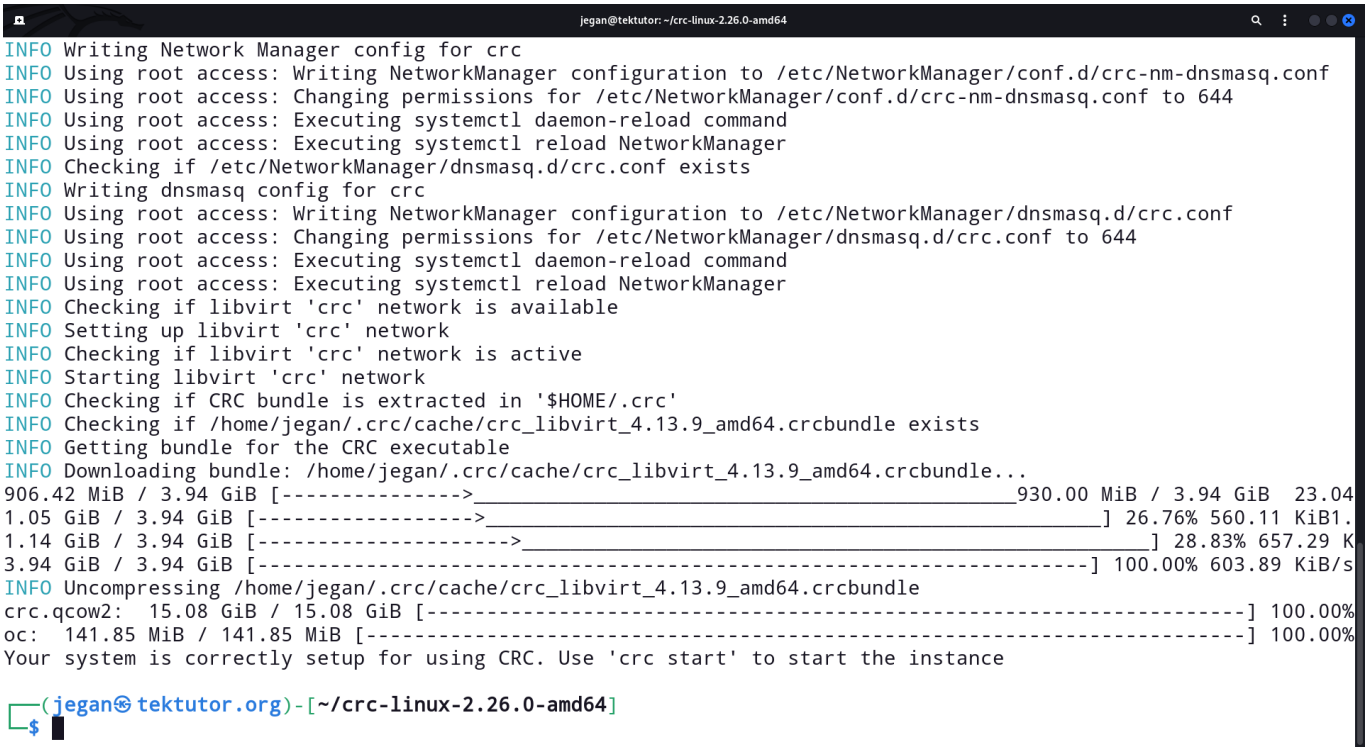
Now you can start the Code Ready Container Setup

```
crc setup
```

Expected output



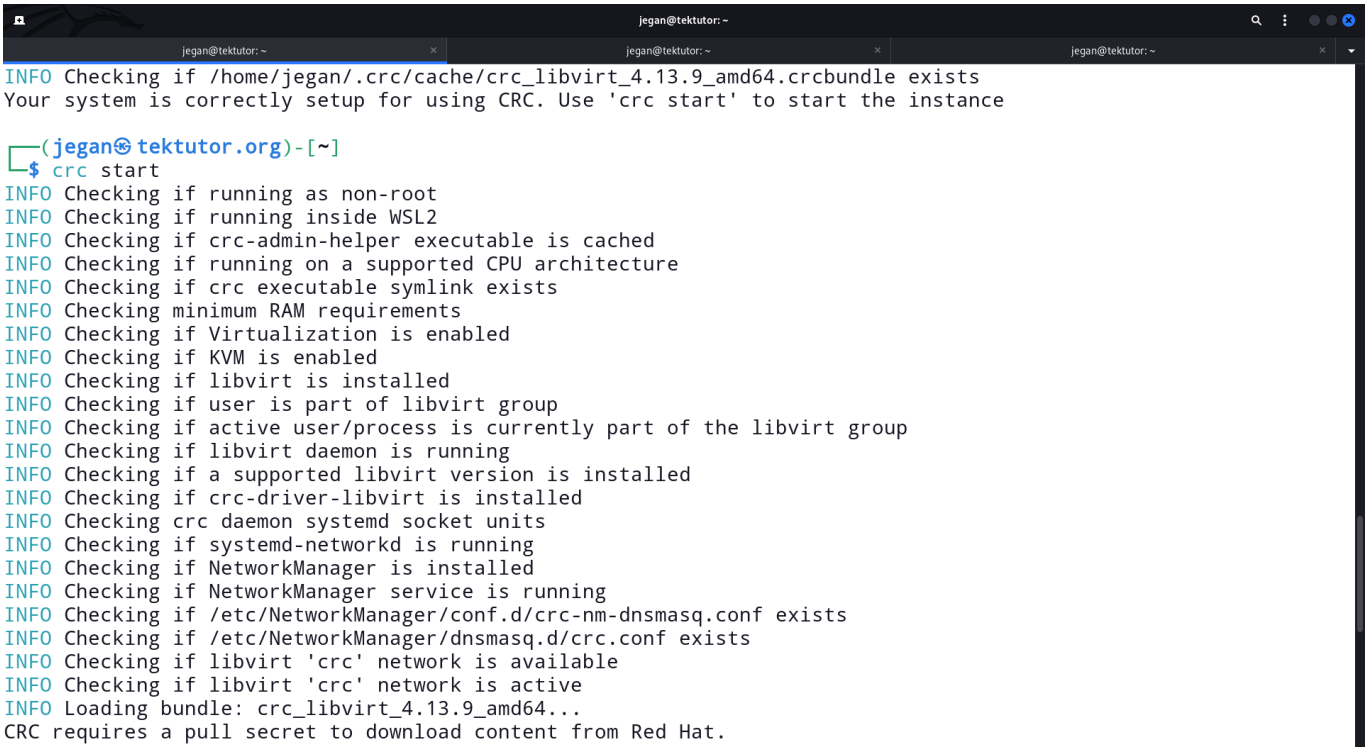




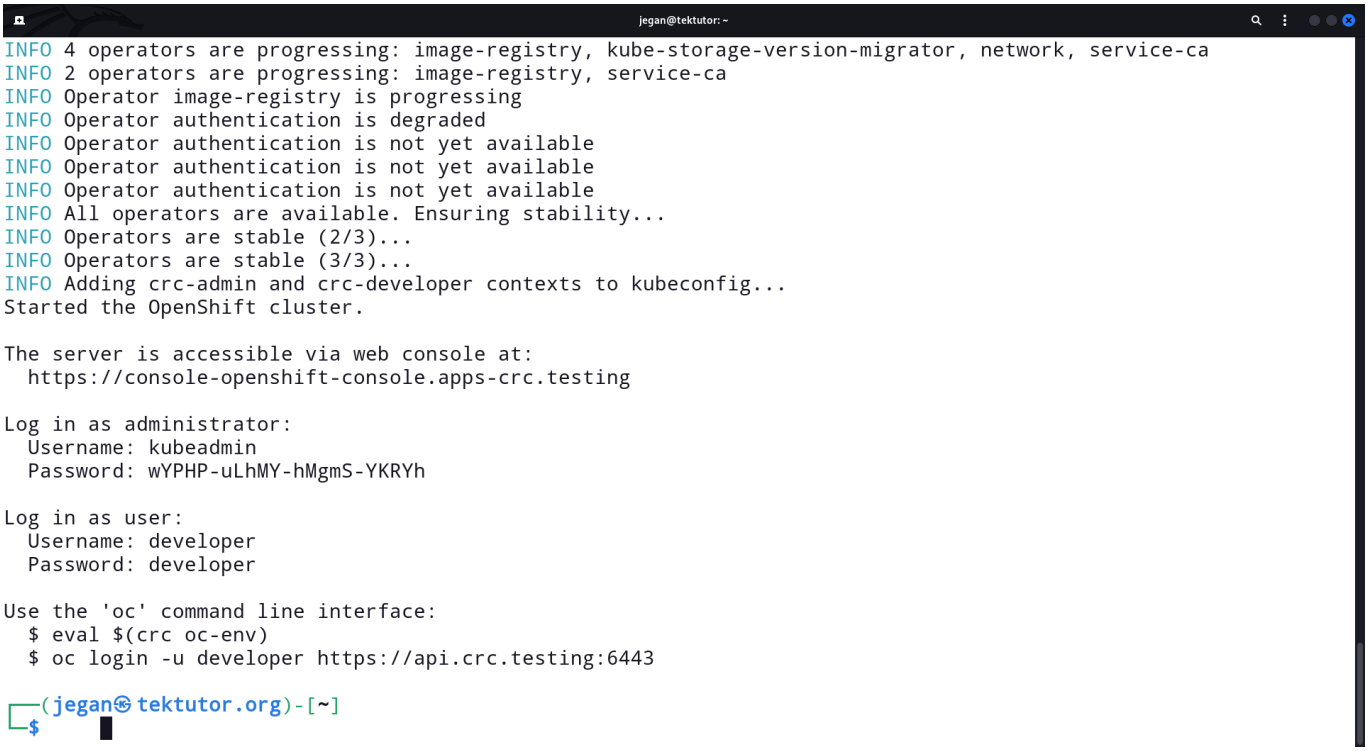
Now you need to start the CRC as shown below

```
crc start
```

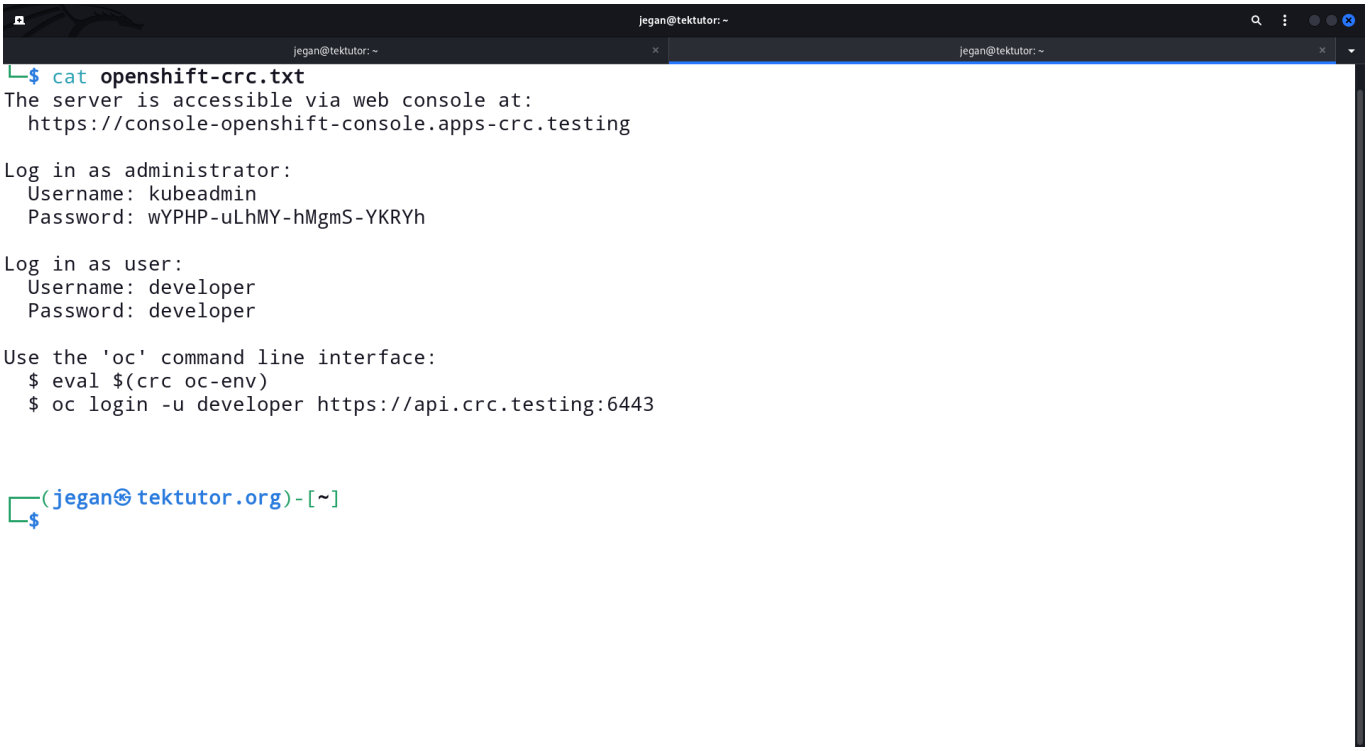
Expected output




jegan@tektutor: ~
INFO Loading bundle: crc_libvirt_4.13.9_amd64...
CRC requires a pull secret to download content from Red Hat.
You can copy it from the Pull Secret section of https://console.redhat.com/openshift/create/local.
? Please enter the pull secret
INFO Creating CRC VM for OpenShift 4.13.9...
INFO Generating new SSH key pair...
INFO Generating new password for the kubeadmin user
INFO Starting CRC VM for openshift 4.13.9...
INFO CRC instance is running with IP 192.168.130.11
INFO CRC VM is running
INFO Updating authorized keys...
INFO Configuring shared directories
INFO Check internal and public DNS query...
INFO Check DNS query from host...
INFO Verifying validity of the kubelet certificates...
INFO Starting kubelet service
INFO Waiting for kube-apiserver availability... [takes around 2min]
INFO Adding user's pull secret to the cluster...
INFO Updating SSH key to machine config resource...
INFO Waiting until the user's pull secret is written to the instance disk...
INFO Changing the password for the kubeadmin user
INFO Updating cluster ID...
INFO Updating root CA cert to admin-kubeconfig-client-ca configmap...
INFO Starting openshift instance... [waiting for the cluster to stabilize]
INFO 8 operators are progressing: dns, image-registry, ingress, kube-storage-version-migrator, network, ...
INFO 6 operators are progressing: image-registry, ingress, kube-storage-version-migrator, network, openshift-con
troller-manager, ...
INFO 5 operators are progressing: image-registry, ingress, kube-storage-version-migrator, network, service-ca
INFO 5 operators are progressing: image-registry, ingress, kube-storage-version-migrator, network, servi
ce-ca


jegan@tektutor: ~
INFO 4 operators are progressing: image-registry, kube-storage-version-migrator, network, service-ca
INFO 2 operators are progressing: image-registry, service-ca
INFO Operator image-registry is progressing
INFO Operator authentication is degraded
INFO Operator authentication is not yet available
INFO Operator authentication is not yet available
INFO Operator authentication is not yet available
INFO All operators are available. Ensuring stability...
INFO Operators are stable (2/3)...
INFO Operators are stable (3/3)...
INFO Adding crc-admin and crc-developer contexts to kubeconfig...
Started the OpenShift cluster.
The server is accessible via web console at:
https://console-openshift-console.apps-crc.testing
Log in as administrator:
Username: kubeadmin
Password: wYPHP-uLhMY-hMgmS-YKRYh
Log in as user:
Username: developer
Password: developer
Use the 'oc' command line interface:
$ eval $(crc oc-env)
$ oc login -u developer https://api.crc.testing:6443
(jegan㉿tektutor.org)-[~]
$

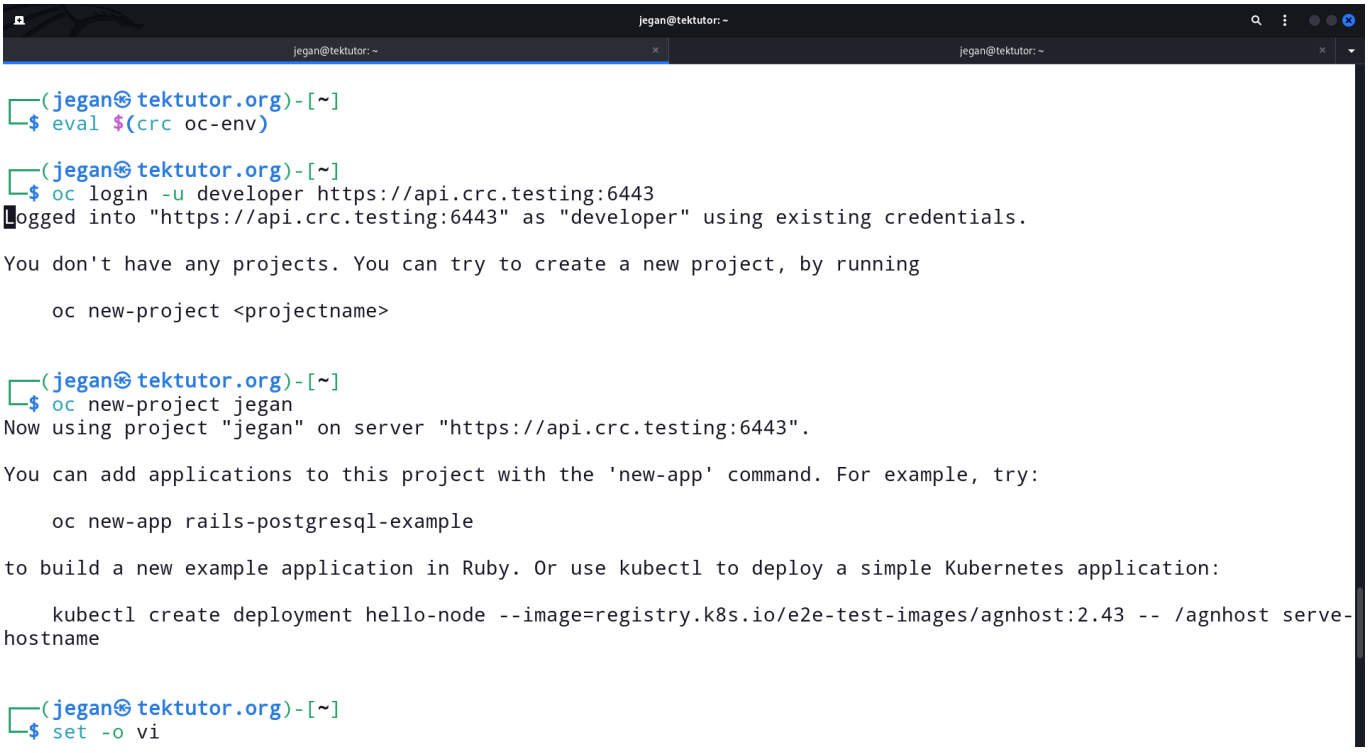
Don't forget to save the Red Hat OpenShift login credentials, in my case I saved it as shown below



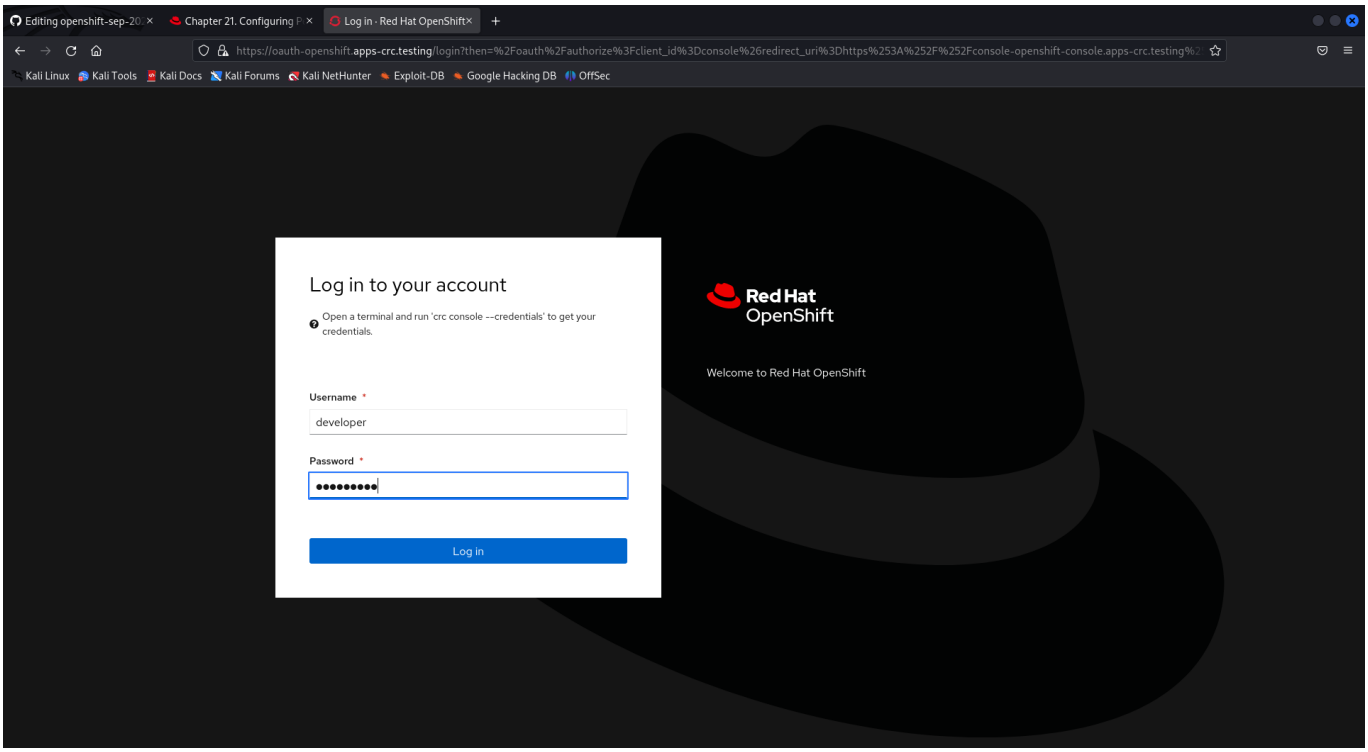
Login to Red Hat OpenShift via command-line and create a project

```
oc login -u developer https://api.crc.testing:6443
```

Expected output



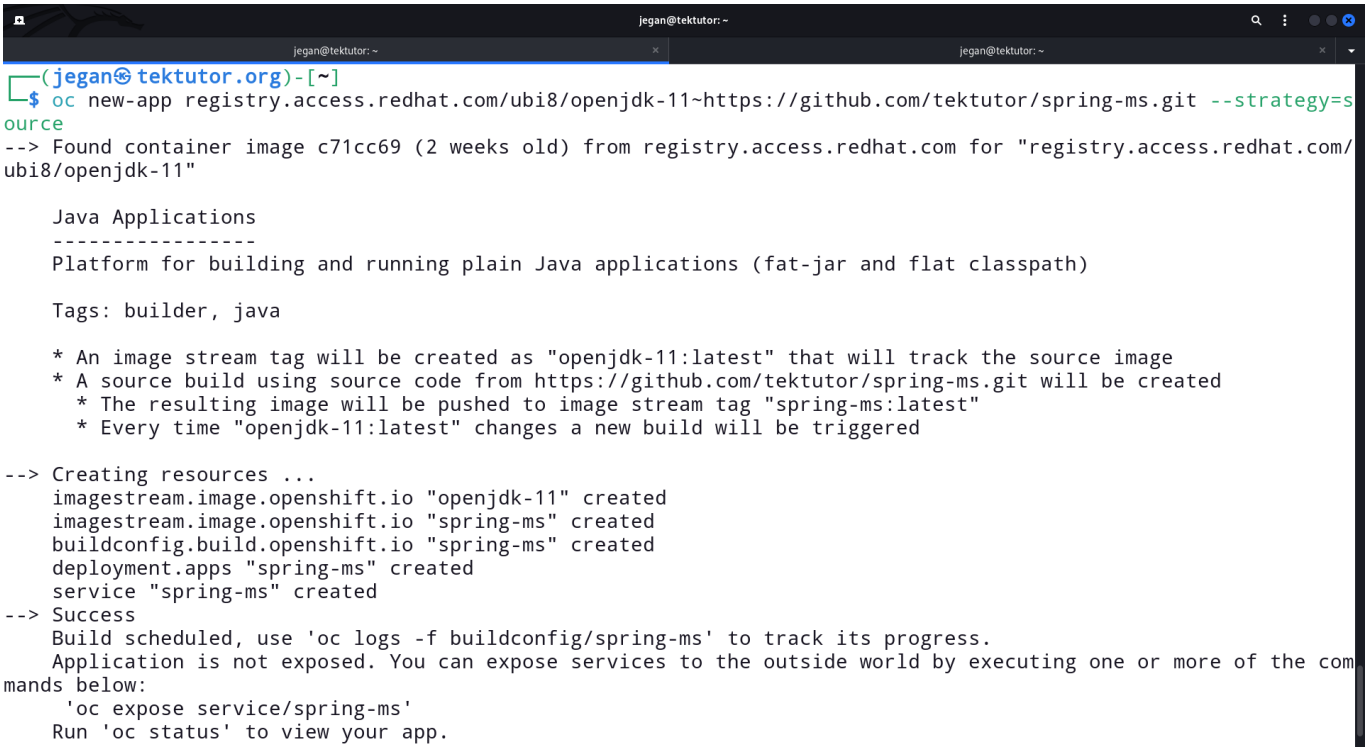
You can access the webconsole at the below URL in your web browser with developer as user and developer as password



Deploy an application to check if your cluster is working properly

```
oc new-app registry.access.redhat.com/ubi8/openjdk-11~https://github.com/tektutor/spring-ms.git --strategy=source
```

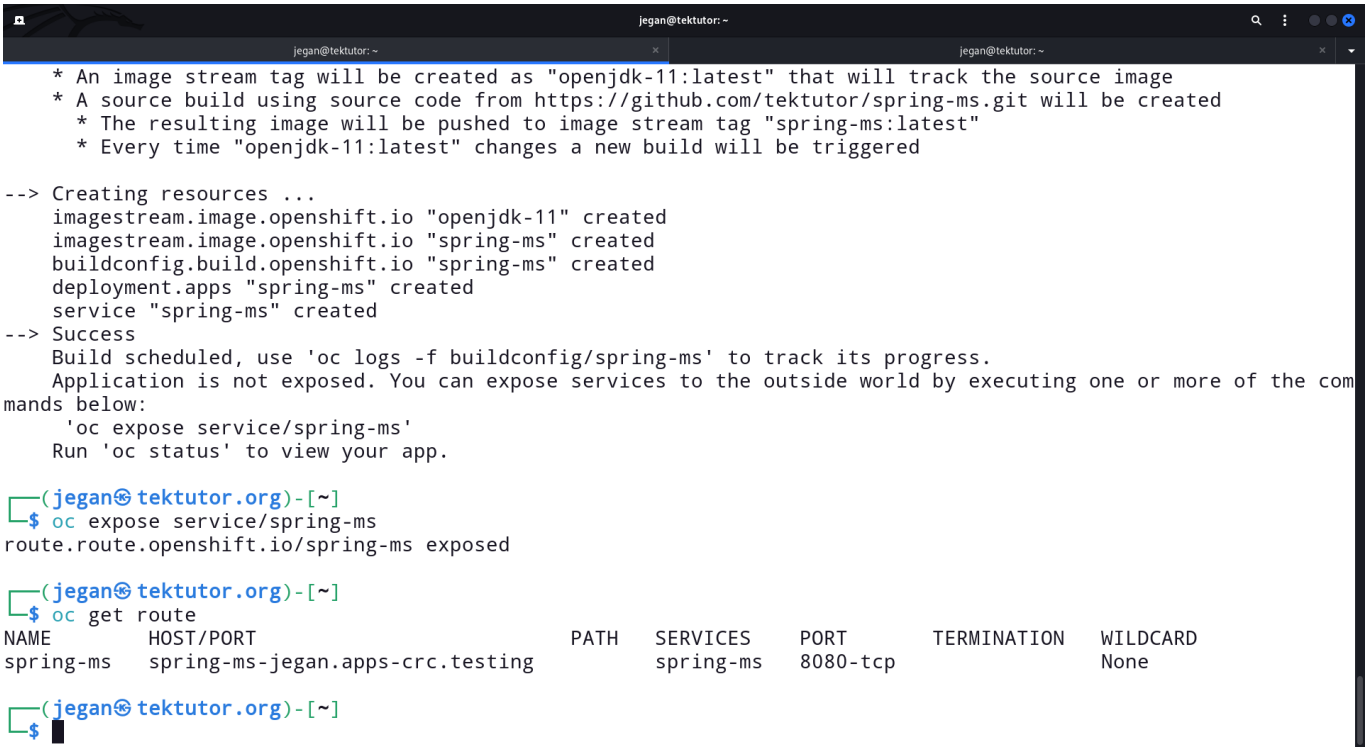
Expected output



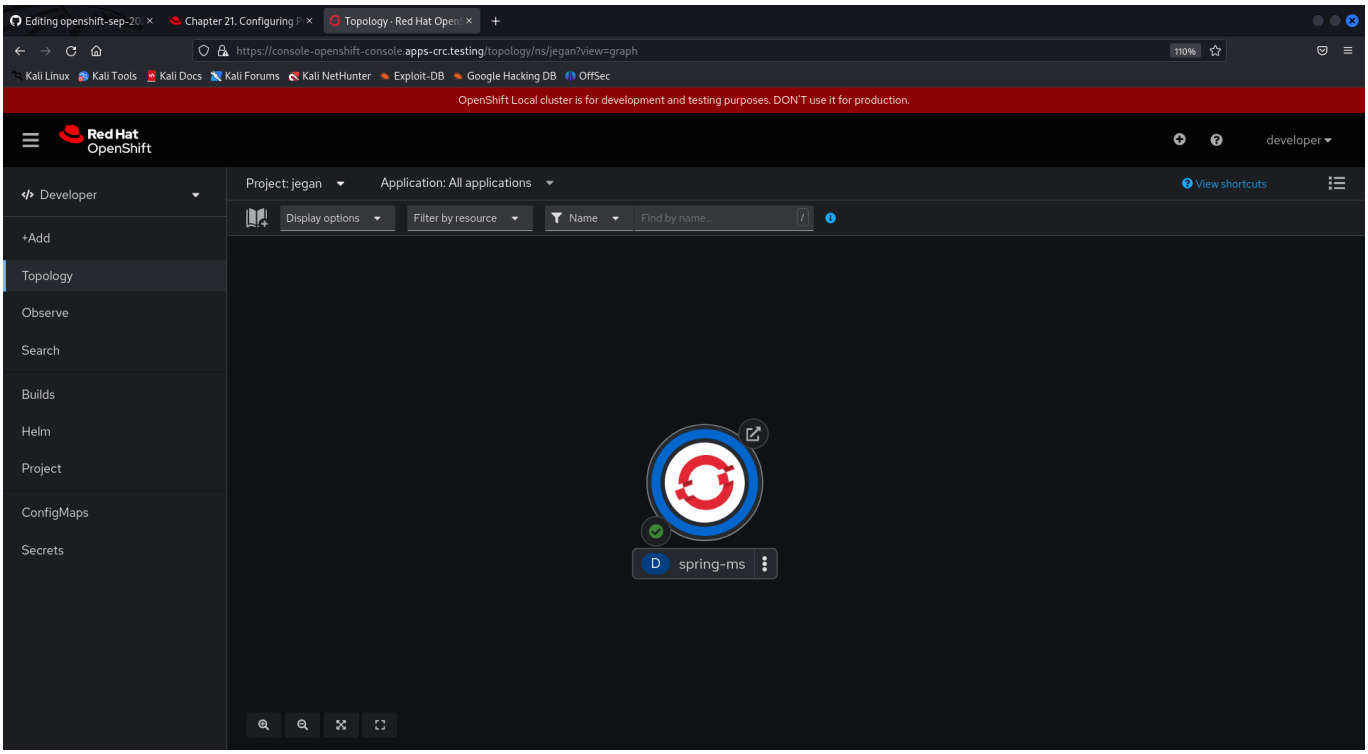
Create a route to access the deployed application

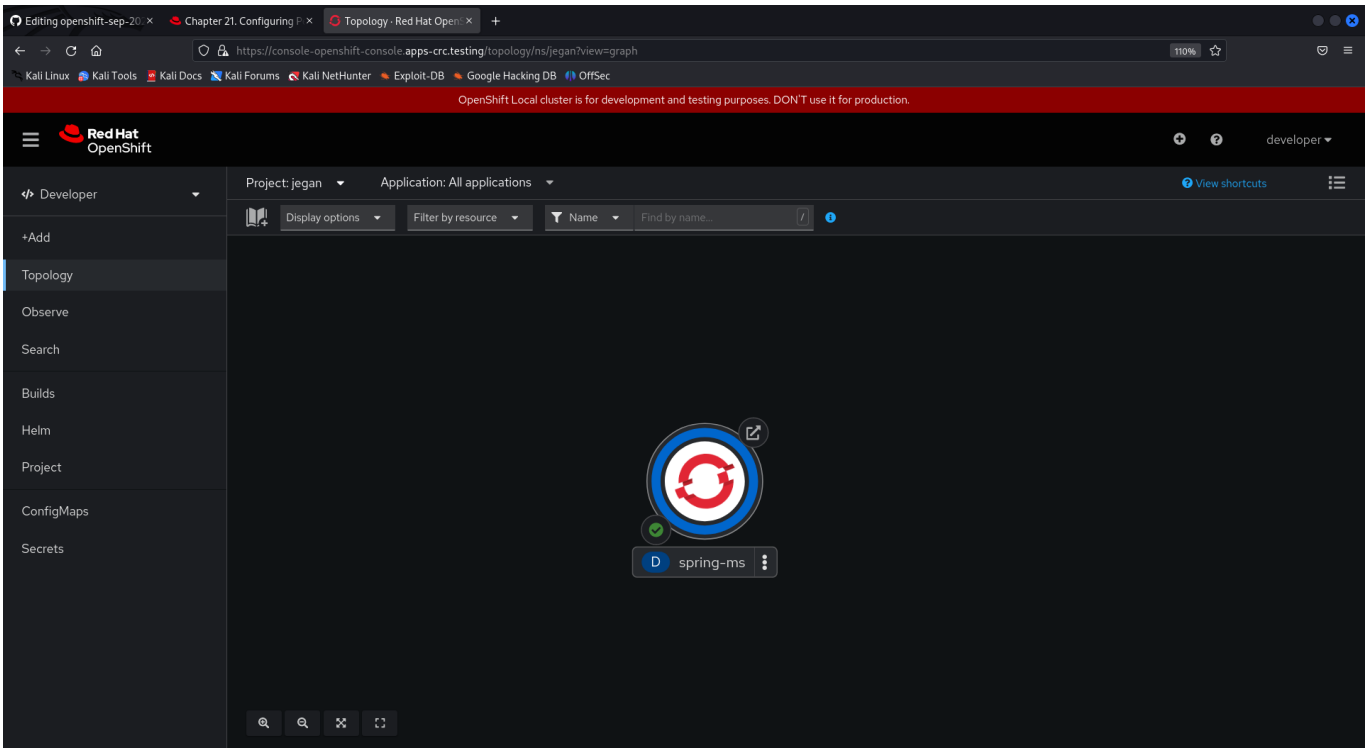
```
oc expose service/spring-ms
```

Expected ouput



Now head-over to your Red Hat OpenShift webconsole to access the application route





Congratulations!, your Red Hat OpenShift local setup is ready to practice ...