Helm

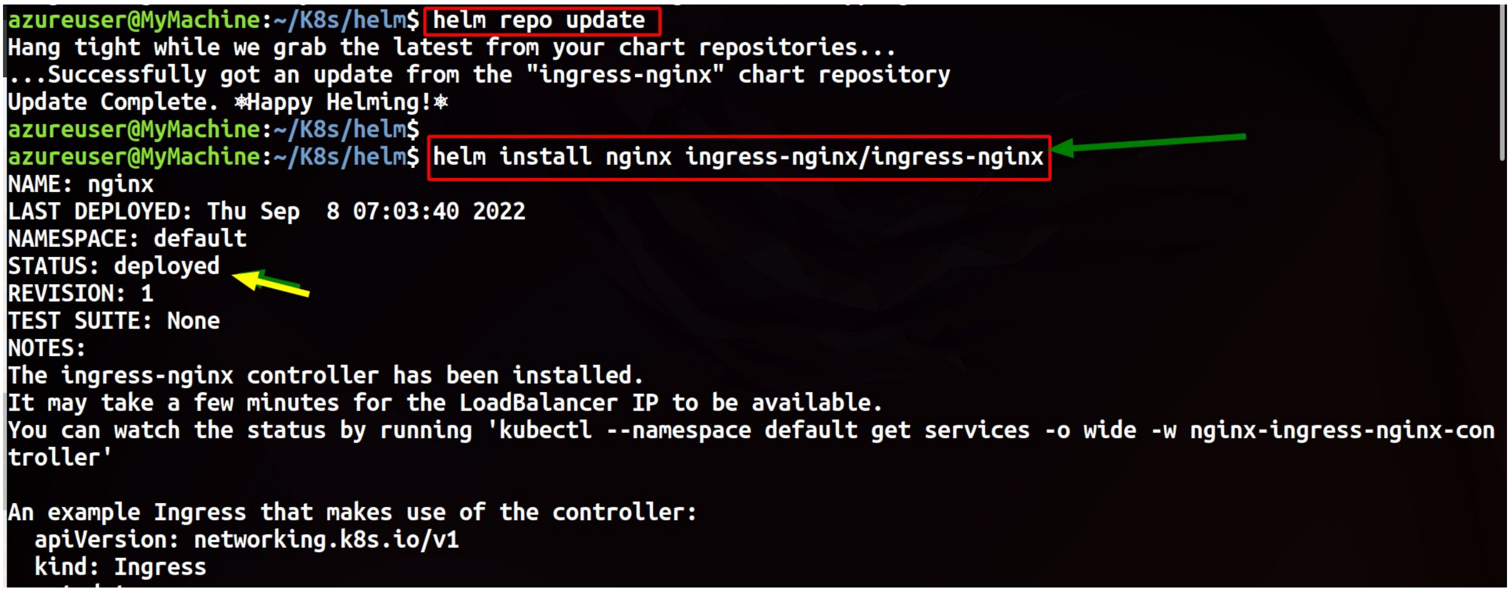
>> Install Package installer [helm]
>> Install ingress controller [nginx] using helm
>> Create a Deployment for sBoot app
>> Create an ingress rule for the sBoot endpoint
>> Verify using ingress ctrl port and ingress ctrl logs

Step1: >> Install Package installer [helm]

```
$  curl https://baltocdn.com/helm/signing.asc | sudo apt-key add -
$ curl https://baltocdn.com/helm/signing.asc | sudo apt-key add -
$ sudo apt-get install apt-transport-https --yes
$ echo "deb https://baltocdn.com/helm/stable/debian/ all main" | sudo tee /etc/apt/sources.list.d/helm-stable-debian.list
$ sudo apt-get update
$ sudo apt-get install helm
$ helm -version
```

Step2:>> Install ingress controller [nginx] using helm

```
$ helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
$ helm repo update
$ helm install nginx ingress-nginx/ingress-nginx
```



```
azureuser@MyMachine:~/K8s/helm$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "ingress-nginx" chart repository
Update Complete. ⎈Happy Helming!⎈
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$ helm install nginx ingress-nginx/ingress-nginx
NAME: nginx
LAST DEPLOYED: Thu Sep  8 07:03:40 2022
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The ingress-nginx controller has been installed.
It may take a few minutes for the LoadBalancer IP to be available.
You can watch the status by running 'kubectl --namespace default get services -o wide -w nginx-ingress-nginx-con
troller'

An example Ingress that makes use of the controller:
  apiVersion: networking.k8s.io/v1
  kind: Ingress
```

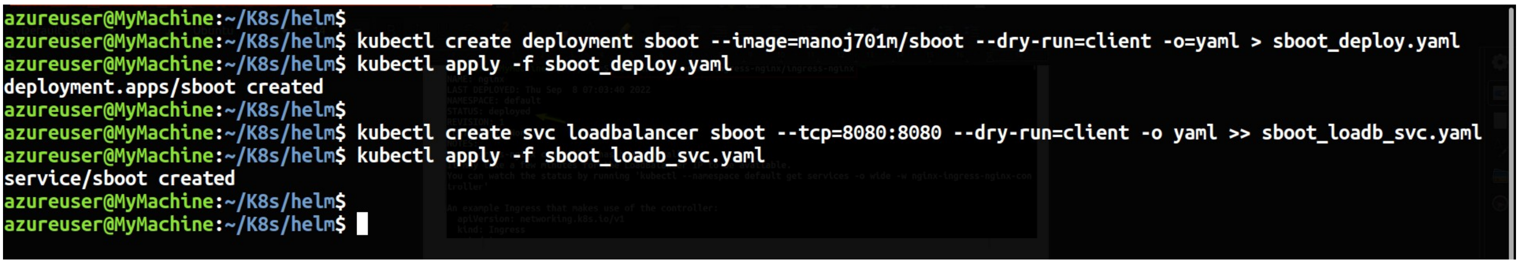
Step3: >> Create a Deployment for sBoot app
>> Create a loadbalancer Service

$ kubectl create deployment sboot --image=manoj701m/sboot --dry-run=client -o=yaml > sboot_deploy.yaml

$ kubectl apply -f sboot_deploy.yaml

$ kubectl create svc loadbalancer sboot --tcp=8080:8080 --dry-run=client -o yaml >> sboot_loadb_svc.yaml

$ kubectl apply -f sboot_loadb_svc.yaml

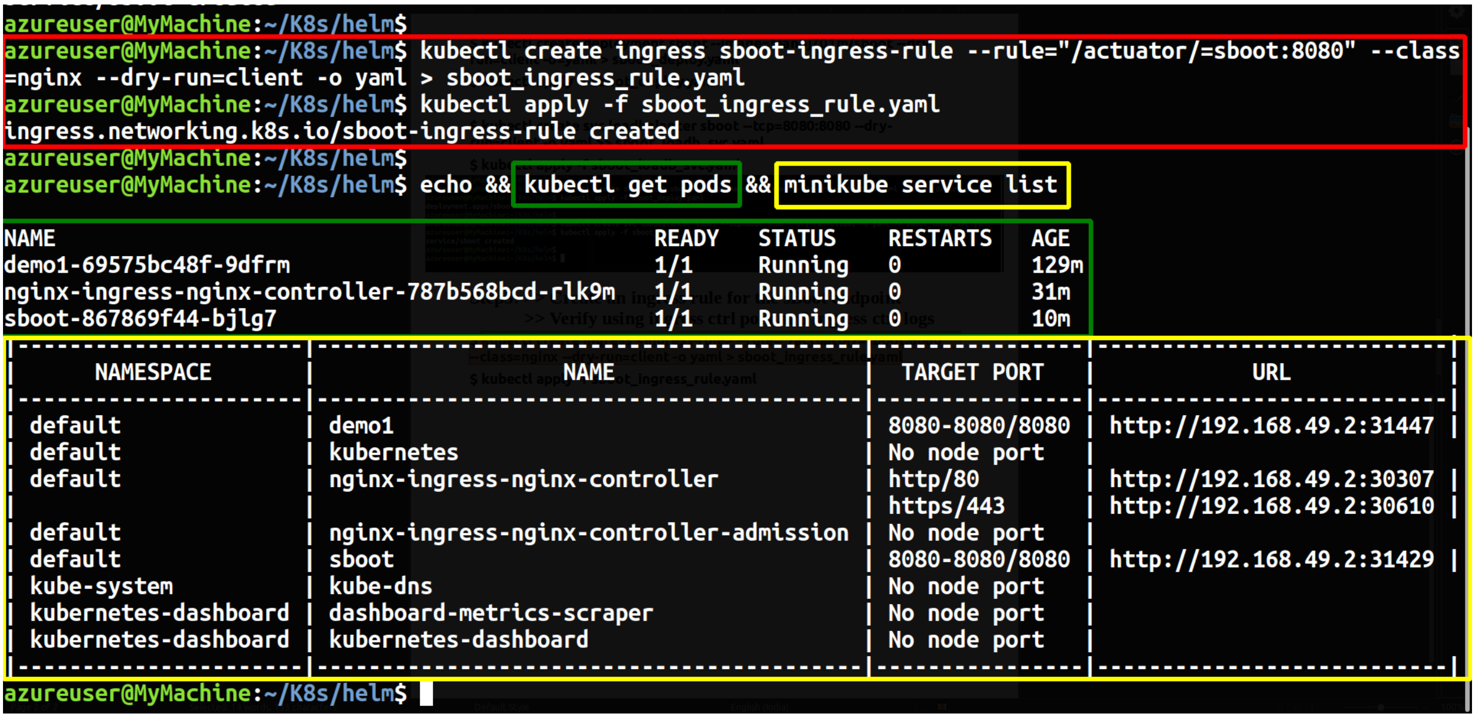


```
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$ kubectl create deployment sboot --image=manoj701m/sboot --dry-run=client -o=yaml > sboot_deploy.yaml
azureuser@MyMachine:~/K8s/helm$ kubectl apply -f sboot_deploy.yaml
deployment.apps/sboot created
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$ kubectl create svc loadbalancer sboot --tcp=8080:8080 --dry-run=client -o yaml >> sboot_loadb_svc.yaml
azureuser@MyMachine:~/K8s/helm$ kubectl apply -f sboot_loadb_svc.yaml
service/sboot created
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$
```

Step4: >> Create an ingress rule for the sBoot endpoint

$ kubectl create ingress <ingress-rule> --rule="/actuator/=sboot:8080" --class=nginx --dry-run=client -o yaml > sboot_ingress_rule.yaml

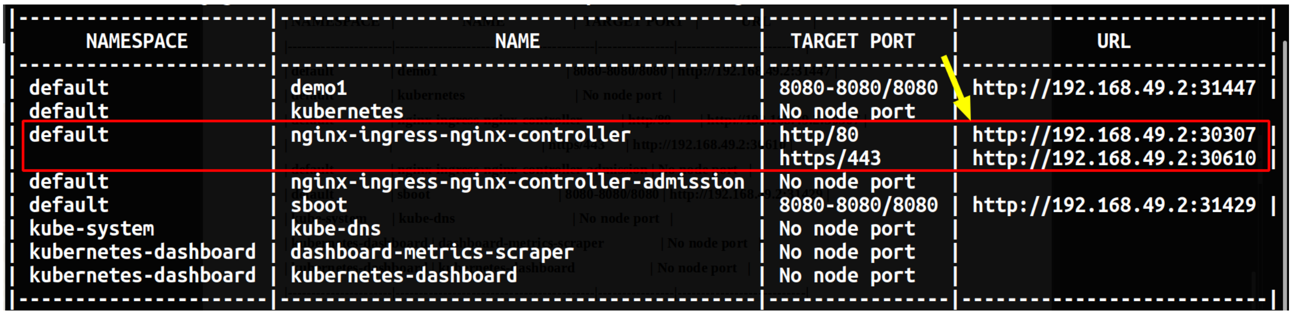
$ kubectl apply -f sboot_ingress_rule.yaml



```
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$ kubectl create ingress sboot-ingress-rule --rule="/actuator/=sboot:8080" --class
=nginx --dry-run=client -o yaml > sboot_ingress_rule.yaml
azureuser@MyMachine:~/K8s/helm$ kubectl apply -f sboot_ingress_rule.yaml
ingress.networking.k8s.io/sboot-ingress-rule created
azureuser@MyMachine:~/K8s/helm$
azureuser@MyMachine:~/K8s/helm$ echo && kubectl get pods && minikube service list

NAME                                             READY   STATUS    RESTARTS   AGE
demo1-69575bc48f-9dfrm                           1/1     Running   0          129m
nginx-ingress-nginx-controller-787b568bcd-rlk9m  1/1     Running   0          31m
sboot-867869f44-bjlg7                            1/1     Running   0          10m
|----------------------|------------------------------------------|----------------|---------------------------|
|      NAMESPACE       |                   NAME                   |  TARGET PORT   |            URL            |
|----------------------|------------------------------------------|----------------|---------------------------|
| default              | demo1                                    | 8080-8080/8080 | http://192.168.49.2:31447 |
| default              | kubernetes                               | No node port   |
| default              | nginx-ingress-nginx-controller           | http/80        | http://192.168.49.2:30307 |
|                      |                                          | https/443      | http://192.168.49.2:30610 |
| default              | nginx-ingress-nginx-controller-admission | No node port   |
| default              | sboot                                    | 8080-8080/8080 | http://192.168.49.2:31429 |
| kube-system          | kube-dns                                 | No node port   |
| kubernetes-dashboard | dashboard-metrics-scraper                | No node port   |
| kubernetes-dashboard | kubernetes-dashboard                     | No node port   |
|----------------------|------------------------------------------|----------------|---------------------------|
azureuser@MyMachine:~/K8s/helm$
```

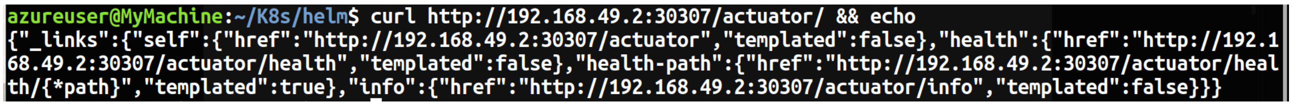
Step5: >> Verify using ingress ctrl port and ingress ctrl logs



```
|----------------------|-------------------------------------|--------------|---------------------------|
|      NAMESPACE       |                NAME                 | TARGET PORT  |            URL            |
|----------------------|-------------------------------------|--------------|---------------------------|
| default              | demo1                               | 8080-8080/8080 | http://192.168.49.2:31447 |
| default              | kubernetes                          | No node port |
| default              | nginx-ingress-nginx-controller      | http/80      | http://192.168.49.2:30307 |
|                      |                                     | https/443    | http://192.168.49.2:30610 |
| default              | nginx-ingress-nginx-controller-admission | No node port |
| default              | sboot                               | 8080-8080/8080 | http://192.168.49.2:31429 |
| kube-system          | kube-dns                            | No node port |
| kubernetes-dashboard | dashboard-metrics-scraper           | No node port |
| kubernetes-dashboard | kubernetes-dashboard                | No node port |
|----------------------|-------------------------------------|--------------|---------------------------|
```

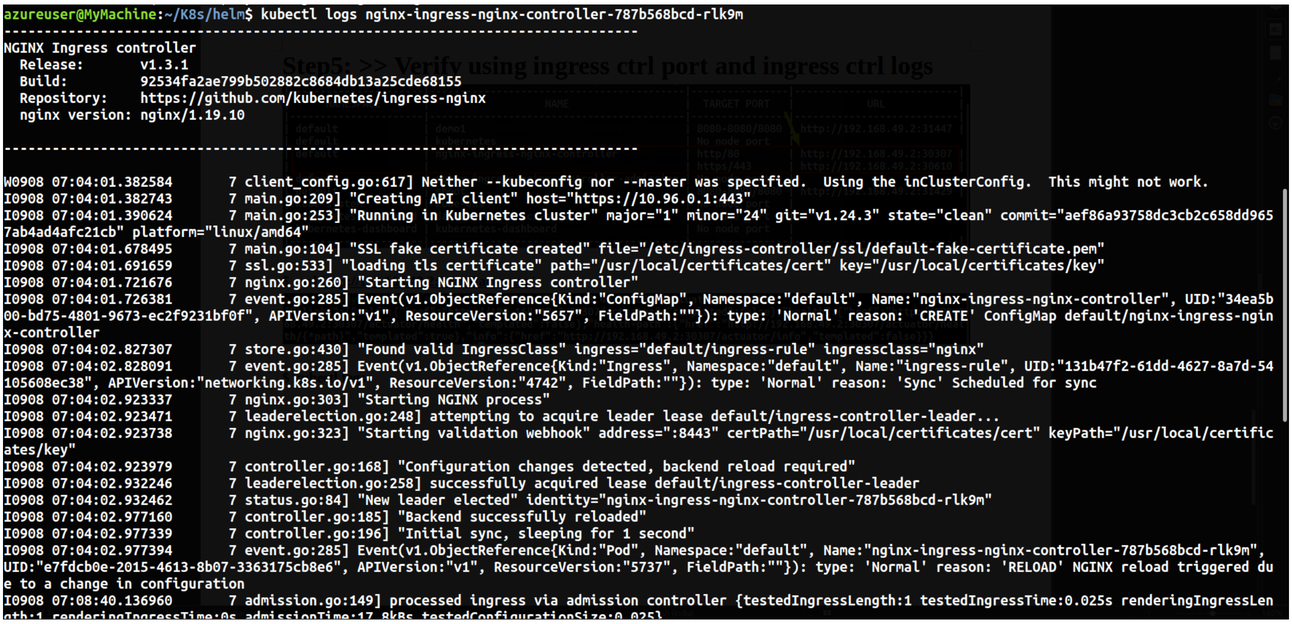
$ curl http://192.168.49.2:30307/actuator/

[ip address is minikube's ip | To get minikube ip >> $ minikube ip]



```
azureuser@MyMachine:~/K8s/helm$ curl http://192.168.49.2:30307/actuator/ && echo
{"_links":{"self":{"href":"http://192.168.49.2:30307/actuator","templated":false},"health":{"href":"http://192.1
68.49.2:30307/actuator/health","templated":false},"health-path":{"href":"http://192.168.49.2:30307/actuator/heal
th/{*path}","templated":true},"info":{"href":"http://192.168.49.2:30307/actuator/info","templated":false}}}
```

$ kubectl logs nginx-ingress-nginx-controller-787b568bcd-rlk9m



```
azureuser@MyMachine:~/K8s/helm$ kubectl logs nginx-ingress-nginx-controller-787b568bcd-rlk9m
-------------------------------------------------------------------------------
NGINX Ingress controller
  Release:       v1.3.1
  Build:         92534fa2ae799b502882c8684db13a25cde68155
  Repository:    https://github.com/kubernetes/ingress-nginx
  nginx version: nginx/1.19.10

-------------------------------------------------------------------------------

W0908 07:04:01.382584       7 client_config.go:617] Neither --kubeconfig nor --master was specified.  Using the inClusterConfig.  This might not work.
I0908 07:04:01.382743       7 main.go:209] "Creating API client" host="https://10.96.0.1:443"
I0908 07:04:01.390624       7 main.go:253] "Running in Kubernetes cluster" major="1" minor="24" git="v1.24.3" state="clean" commit="aef86a93758dc3cb2c658dd965
7ab4ad4afc21cb" platform="linux/amd64"
I0908 07:04:01.678495       7 main.go:104] "SSL fake certificate created" file="/etc/ingress-controller/ssl/default-fake-certificate.pem"
I0908 07:04:01.691659       7 ssl.go:533] "loading tls certificate" path="/usr/local/certificates/cert" key="/usr/local/certificates/key"
I0908 07:04:01.721676       7 nginx.go:260] "Starting NGINX Ingress controller"
I0908 07:04:01.726381       7 event.go:285] Event(v1.ObjectReference{Kind:"ConfigMap", Namespace:"default", Name:"nginx-ingress-nginx-controller", UID:"34ea5b
00-bd75-4801-9673-ec2f9231bf0f", APIVersion:"v1", ResourceVersion:"5657", FieldPath:""}): type: 'Normal' reason: 'CREATE' ConfigMap default/nginx-ingress-ngin
x-controller
I0908 07:04:02.827307       7 store.go:430] "Found valid IngressClass" ingress="default/ingress-rule" ingressclass="nginx"
I0908 07:04:02.828091       7 event.go:285] Event(v1.ObjectReference{Kind:"Ingress", Namespace:"default", Name:"ingress-rule", UID:"131b47f2-61dd-4627-8a7d-54
105608ec38", APIVersion:"networking.k8s.io/v1", ResourceVersion:"4742", FieldPath:""}): type: 'Normal' reason: 'Sync' Scheduled for sync
I0908 07:04:02.923337       7 nginx.go:303] "Starting NGINX process"
I0908 07:04:02.923471       7 leaderelection.go:248] attempting to acquire leader lease default/ingress-controller-leader...
I0908 07:04:02.923738       7 nginx.go:323] "Starting validation webhook" address=":8443" certPath="/usr/local/certificates/cert" keyPath="/usr/local/certific
ates/key"
I0908 07:04:02.923979       7 controller.go:168] "Configuration changes detected, backend reload required"
I0908 07:04:02.932246       7 leaderelection.go:258] successfully acquired lease default/ingress-controller-leader
I0908 07:04:02.932462       7 status.go:84] "New leader elected" identity="nginx-ingress-nginx-controller-787b568bcd-rlk9m"
I0908 07:04:02.977160       7 controller.go:185] "Backend successfully reloaded"
I0908 07:04:02.977339       7 controller.go:196] "Initial sync, sleeping for 1 second"
I0908 07:04:02.977394       7 event.go:285] Event(v1.ObjectReference{Kind:"Pod", Namespace:"default", Name:"nginx-ingress-nginx-controller-787b568bcd-rlk9m",
UID:"e7fdcb0e-2015-4613-8b07-3363175cb8e6", APIVersion:"v1", ResourceVersion:"5737", FieldPath:""}): type: 'Normal' reason: 'RELOAD' NGINX reload triggered du
e to a change in configuration
I0908 07:08:40.136960       7 admission.go:149] processed ingress via admission controller {testedIngressLength:1 testedIngressTime:0.025s renderingIngressLen
gth:1 renderingIngressTime:0s admissionTime:17.8kBs testedConfigurationSize:0.025}
```