100-Kubernetes Basic 2022

Status	
Assign	

Learn Kubernetes

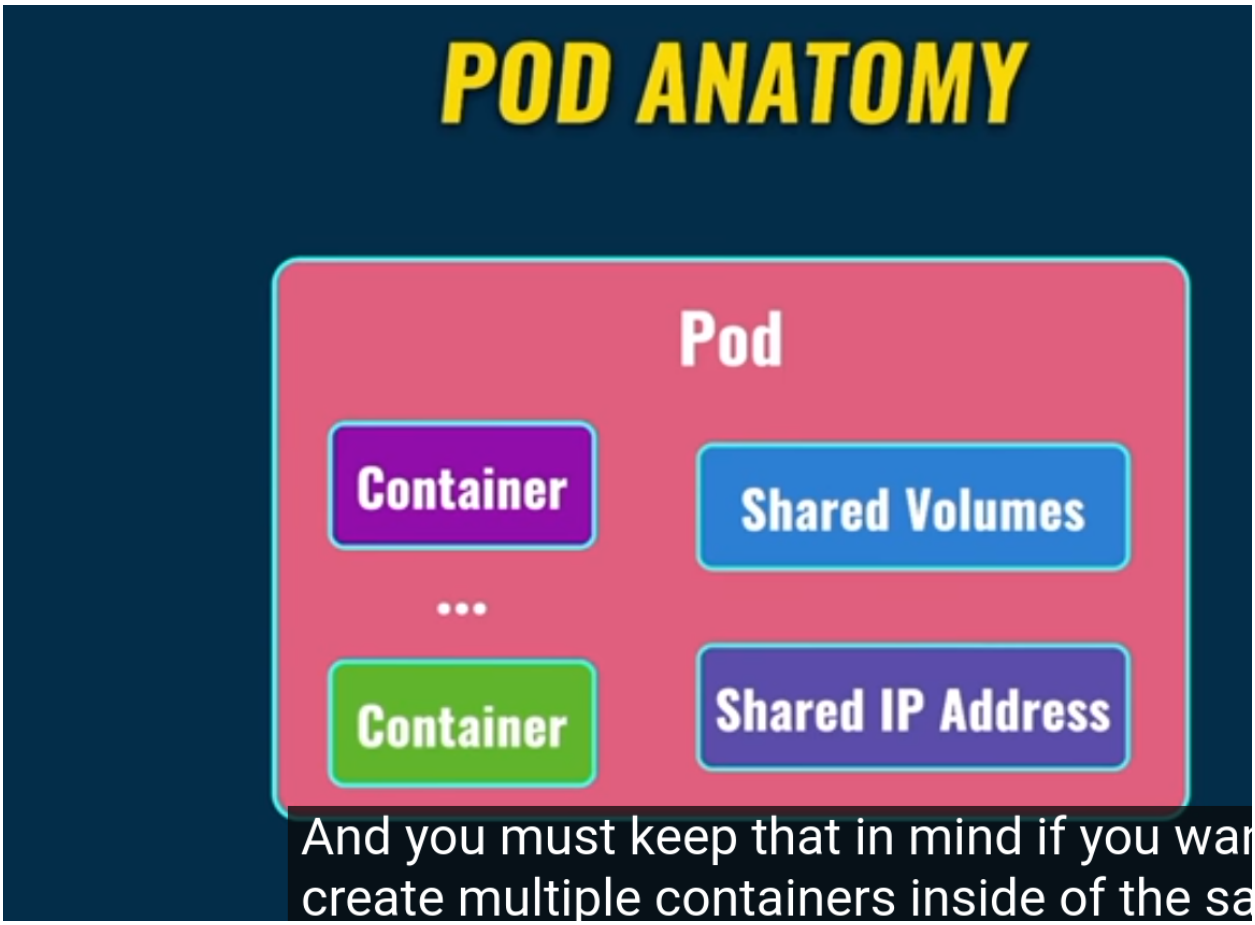
> KUBERNETES is container orchestration system

We can get the Following advantages

- Automatic deployment of the containerised applications across different servers
- Distribution of the load across multiple servers
- Auto-scaling of the deployed applications
- Monitoring and Health check of the containers
- Replacement of the failed containers

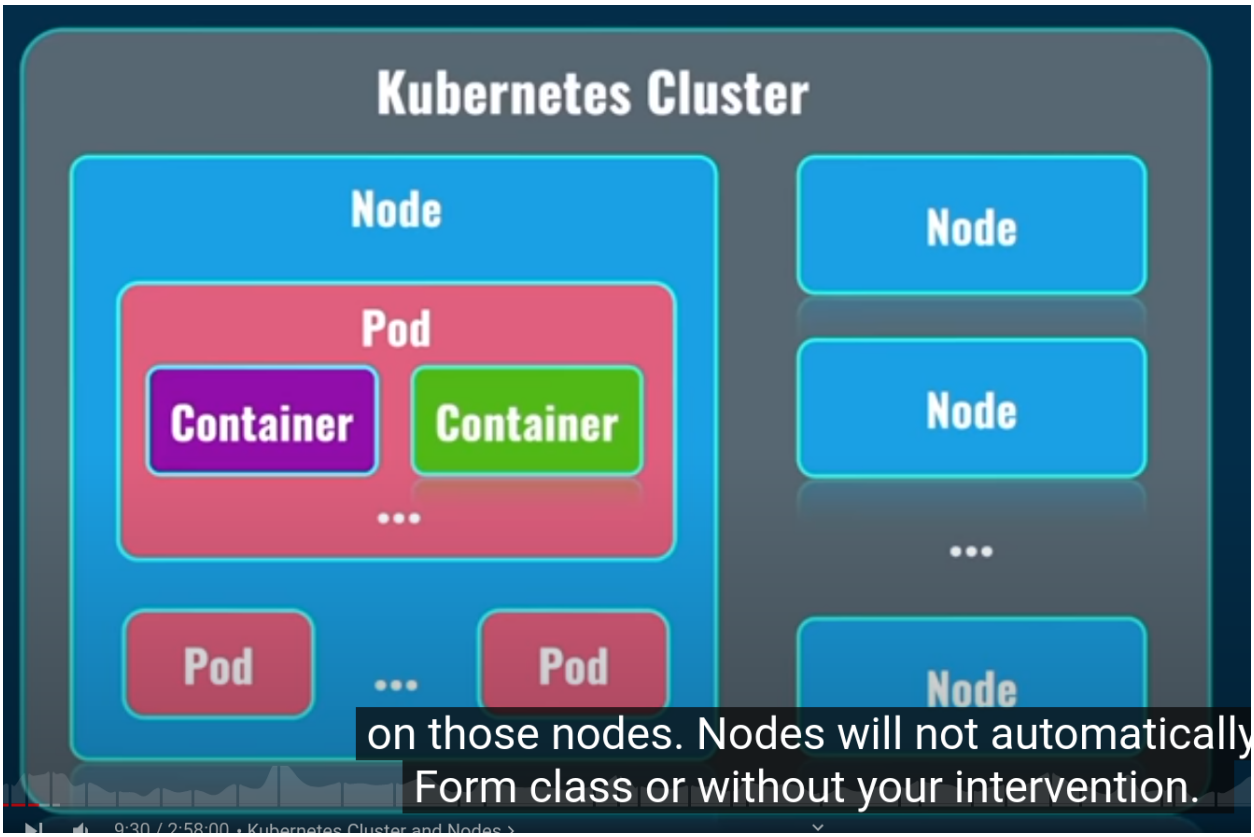
POD is the smallest unit in the Kubernetes World.

containers are present inside the POD. it can be one or more containers present in the POD.



ONE container Per POD is the most common use case

Kubernetes consists of nodes , where each node represent a bare metal server or virtual servers.Nodes belong to same kubernetes cluster usually close to each other in order to perform all jobs more efficiently

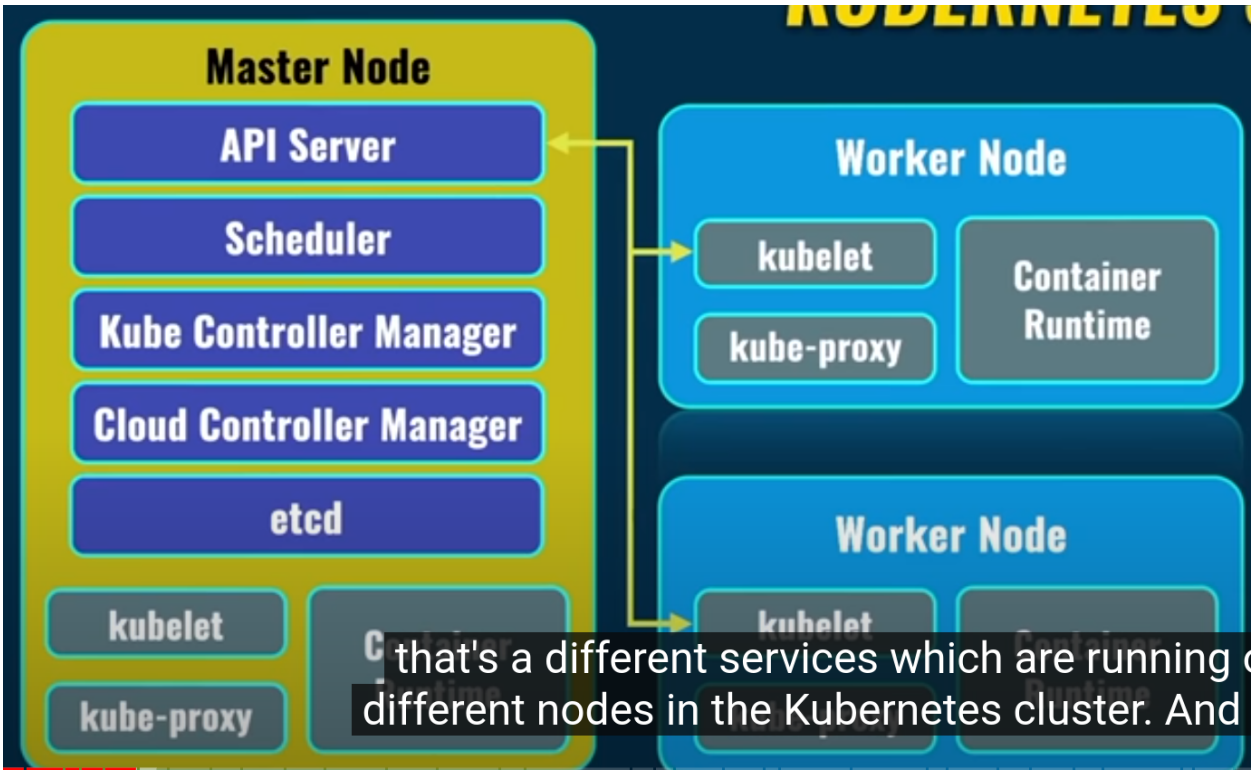


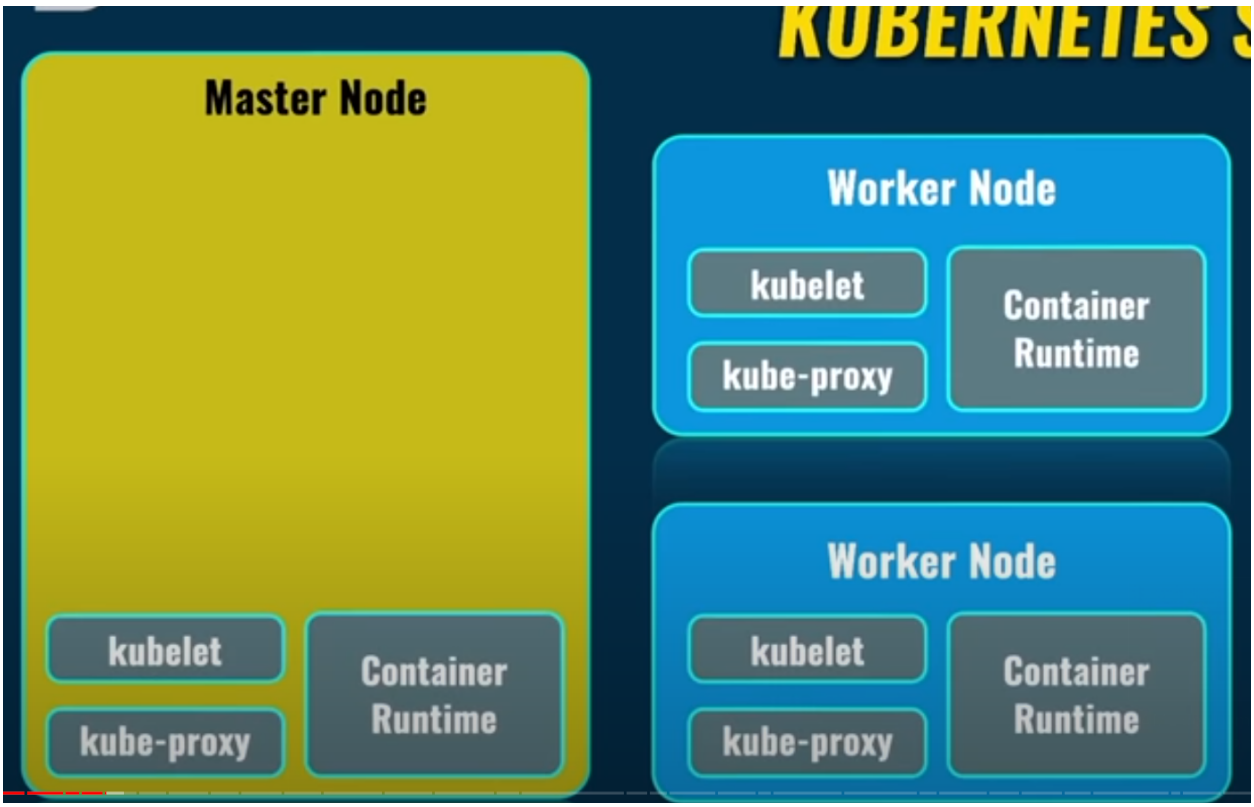
How Nodes are communicate with Each Other

IN k8s cluster There is a master node . Other nodes in the cluster is called worker nodes .

Master Node is actually control plane, and it doesn’t run your client applications.

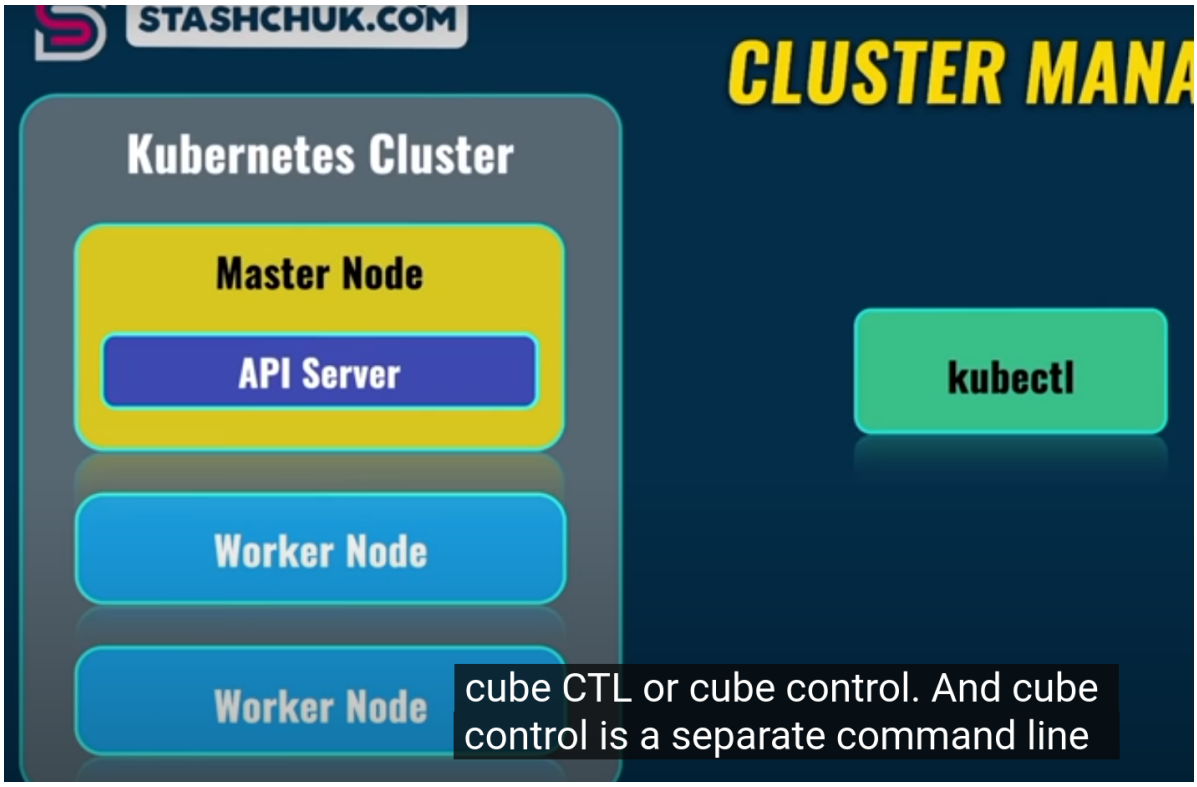
Kubernetes Services


Master Node
API Server
Scheduler
Kube Controller Manager
Cloud Controller Manager
etcd
kubelet
kube-proxy
Worker Node
kubelet
kube-proxy
Container Runtime
Worker Node
that's a different services which are running
different nodes in the Kubernetes cluster. And

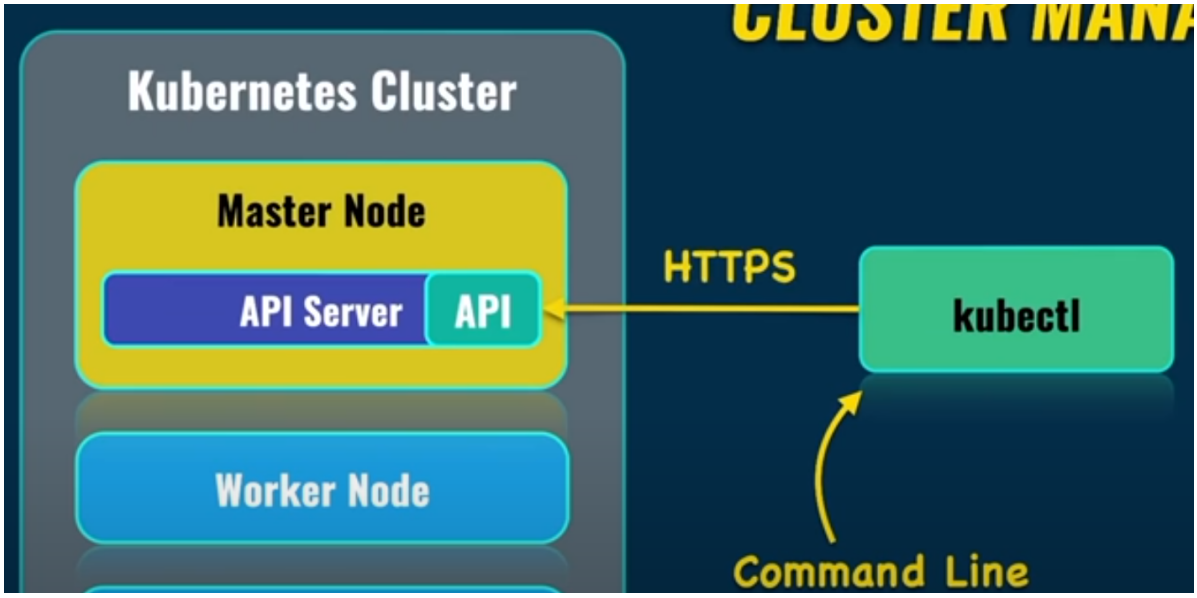

Master Node
kubelet
kube-proxy
Container Runtime
Worker Node
kubelet
kube-proxy
Container Runtime
Worker Node
kubelet
kube-proxy
Container Runtime

- kubelet

    - This service will communicate with API SERVER in the master node
    - API Server is the main point of communication between different nodes in the kubernetes world
- kube-proxy
    - Its responsible for network communication inside of each node and between nodes.
- container runTime
    - we are running our application inside the container runtime, one most common run time is Docker ,
- Scheduler
    - it is responsible for planning and distribution of load between different nodes in the cluster
- Kube controller manager
    - it controls everything in the cluster .
    - It controls what happens on each node in the cluster
- Cloud Controller Manager
    - its responsibilities is to interact with cloud services where we run our kubernetes cluster
- etcd
    - This service actually stores all logs related to operation of entire k8s cluster
- DNS
    - Its running in master and responsible for name resolution in the kubernetes cluster



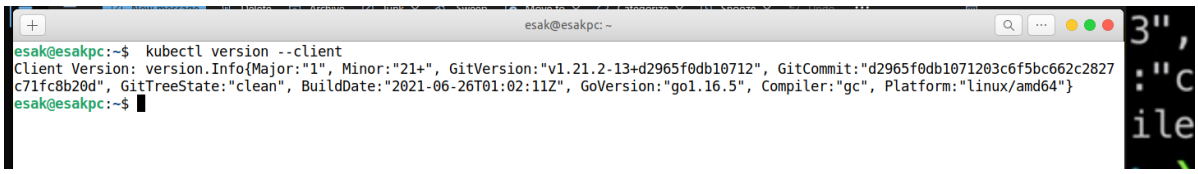
Using API Server we can control the Entire K8s cluster . How we can achieve it ? we use kubectl.

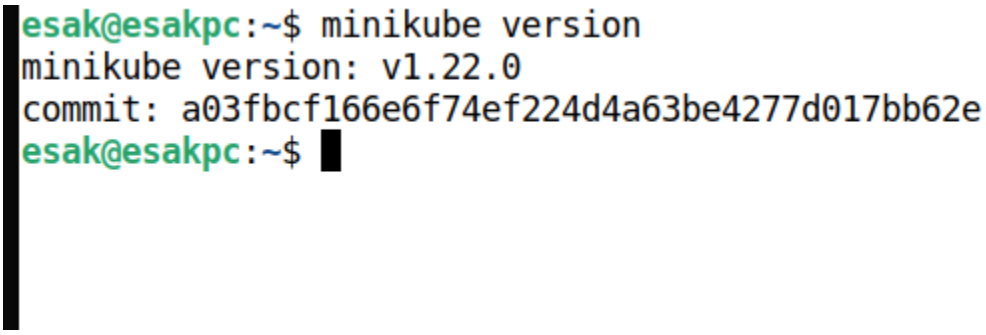


kubectl is a command line tool . which allow us to connect to k8s Cluster and manage it remotely. This communication is happening via restapi over HTTPS

Local Installation

we can minikube to create a single node kubernetes cluster





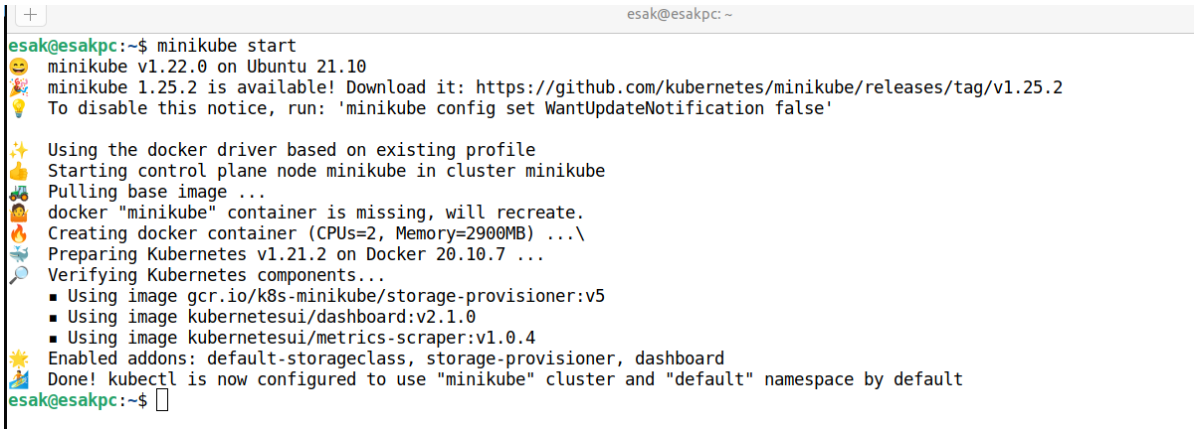
Lets Start a Cluster

Find status

```
esak@esakpc:~$ minikube status
E0602 23:23:20.222059  106235 status.go:258] status error: host: state: unknown state "minikube": docker container inspect minikube --
format={{.State.Status}}: exit status 1
stdout:


stderr:
Error: No such container: minikube
E0602 23:23:20.222095  106235 status.go:261] The "minikube" host does not exist!
minikube
type: Control Plane
host: Nonexistent
kubelet: Nonexistent
apiserver: Nonexistent
kubeconfig: Nonexistent

esak@esakpc:~$
```

Start the Cluster

Single Node Cluster Started



Now login to the minikube Docker in the interactive Mode and we can find the Docker containers inside of it. The services or jorgan we have seen are

available/running in the docker container

```
root@minikube:/# docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS         PORTS     NAMES
7bc2d517e462   6e38f40d628d           "/storage-provisioner"   6 minutes ago   Up 5 minutes             k8s_storage-provisioner_stora
ge-provisioner_kube-system_34cda185-7468-41d4-90a0-51584dc16989_7
53c78cedb76f   86262685d9ab           "/metrics-sidecar"       6 minutes ago   Up 5 minutes             k8s_dashboard-metrics-scraper
_dashboard-metrics-scraper-7976b667d4-r5kc9_kubernetes-dashboard_3c1595b3-416f-4e42-a960-df2dec5325d8_4
3c65ebeaa15a   296a6d5035e2           "/coredns -conf /etc…"   6 minutes ago   Up 5 minutes             k8s_coredns_coredns-558bd4d5d
b-hd8kp_kube-system_831bbd28-d5bd-4d6e-9e88-e3ebe4792b0a_4
3bb154946558   9a07b5b4bfac           "/dashboard --insecu…"   6 minutes ago   Up 5 minutes             k8s_kubernetes-dashboard_kube
rnetes-dashboard-6fcdf4f6d-qstxh_kubernetes-dashboard_15cb65c8-0b0f-4a56-b0ce-a038b648651d_5
acbbee2e8b6b   a6ebd1c1ad98           "/usr/local/bin/kube…"   6 minutes ago   Up 6 minutes             k8s_kube-proxy_kube-proxy-4hd
df_kube-system_c4b82593-da35-419d-8232-face2c82a798_4
dda99e66c67a   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_storage-provisioner_k
ube-system_34cda185-7468-41d4-90a0-51584dc16989_4
ed4aa7092bca   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_kubernetes-dashboard-
6fcdf4f6d-qstxh_kubernetes-dashboard_15cb65c8-0b0f-4a56-b0ce-a038b648651d_4
13d926af5f87   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_kube-proxy-4hddf_kube
-system_c4b82593-da35-419d-8232-face2c82a798_4
5a1dbe0bd64d   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_dashboard-metrics-scr
aper-7976b667d4-r5kc9_kubernetes-dashboard_3c1595b3-416f-4e42-a960-df2dec5325d8_4
83c067fe14bb   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_coredns-558bd4d5db-hd
8kp_kube-system_831bbd28-d5bd-4d6e-9e88-e3ebe4792b0a_4
07179f7b5354   0369cf4303ff           "etcd --advertise-cl…"   6 minutes ago   Up 6 minutes             k8s_etcd_etcd-minikube_kube-s
ystem_be5cbc7ffcadbd4ffc776526843ee514_4
90acd702c082   k8s.gcr.io/pause:3.4.1 "/pause"                 6 minutes ago   Up 6 minutes             k8s_POD_etcd-minikube_kube-sy
stem_be5cbc7ffcadbd4ffc776526843ee514_4
e3d5d89651d2   ae24db9aa2cc           "kube-controller-man…"   6 minutes ago   Up 6 minutes             k8s_kube-controller-manager_k
ube-controller-manager-minikube_kube-system_a5754fbaabd2854e0e0cdce8400679ea_4
c3859547e4b7   106ff58d4308           "kube-apiserver --ad…"   6 minutes ago   Up 6 minutes             k8s_kube-apiserver_kube-apise
rver-minikube_kube-system_cefbe66f503bf010430ec3521cf31be8_4
3615da90ed83   f917b8c8f55b           "kube-scheduler --au…"   6 minutes ago   Up 6 minutes             k8s_kube-scheduler_kube-sched
uler-minikube_kube-system_a2acd1bccd50fd7790183537181f658e_4
```

Running kubectl commands from the HOST Machine

we are using kubectl to control the k8s Cluster

```
esak@esakpc: ~
esak@esakpc:~$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
esak@esakpc:~$
```

Kubectl will always create a single Node

This Single Node will act as the master and worker Node

```
esak@esakpc:~$ kubectl get nodes
NAME       STATUS   ROLES                  AGE    VERSION
minikube   Ready    control-plane,master   287d   v1.21.2
esak@esakpc:~$
```

Namespaces are used in kubernetes to group different resources and configuration objects

```
esak@esakpc:~$ kubectl get pods
No resources found in default namespace.
esak@esakpc:~$ kubectl get namespaces
NAME                   STATUS   AGE
default                Active   287d
kube-node-lease        Active   287d
kube-public            Active   287d
kube-system            Active   287d
kubernetes-dashboard   Active   287d
esak@esakpc:~$
```

To Get PODS from different namesapce

```
esak@esakpc:~$ kubectl get pods --namespace=kube-system
NAME                               READY   STATUS    RESTARTS   AGE
coredns-558bd4d5db-hd8kp           1/1     Running   4          287d
etcd-minikube                      1/1     Running   4          287d
kube-apiserver-minikube            1/1     Running   4          287d
kube-controller-manager-minikube   1/1     Running   4          287d
kube-proxy-4hddf                   1/1     Running   4          287d
kube-scheduler-minikube            1/1     Running   4          287d
storage-provisioner                1/1     Running   7          287d
esak@esakpc:~$
```

To stop the Cluster

```
storage-provisioner                    1/1     Running   7
esak@esakpc:~$ minikube stop
✋  Stopping node "minikube"  ...
🛑  Powering off "minikube" via SSH ...
🛑  1 nodes stopped.
esak@esakpc:~$
```

Create First POD

```
esak@esakpc:~$ kubectl run nginx --image=nginx
pod/nginx created
esak@esakpc:~$ kubectl get pods
NAME    READY   STATUS              RESTARTS   AGE
nginx   0/1     ContainerCreating   0          7s
esak@esakpc:~$ kubectl get pods
NAME    READY   STATUS              RESTARTS   AGE
nginx   0/1     ContainerCreating   0          15s
esak@esakpc:~$ kubectl get pods
NAME    READY   STATUS              RESTARTS   AGE
nginx   0/1     ContainerCreating   0          28s
esak@esakpc:~$ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          53s
esak@esakpc:~$
```

GET information about the pod NGINX

```
esak@esakpc:~$ kubectl describe pod nginx
Name:         nginx
Namespace:    default
Priority:     0
Node:         minikube/192.168.49.2
Start Time:   Fri, 03 Jun 2022 13:48:46 +0530
Labels:       run=nginx
Annotations:  <none>
Status:       Running
IP:           172.17.0.5
IPs:
  IP:  172.17.0.5
Containers:
  nginx:
    Container ID:   docker://861f9b8c9414812c6373f3cae3c1233959da41f2f1c39be3ffdfeaec60781234
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:2bcabc23b45489fb0885d69a06ba1d648aeda973fae7bb981bafbb884165e514
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Fri, 03 Jun 2022 13:49:20 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-pkf4h (ro)
Conditions:
  Type              Status
```

In order to connect to ports we have to use services in kubernetes

we can see 2 nginx containers one container is our nginx and another one is called pause container .This pause container is created to keel the namespace of the PORT

```
root@minikube: /
root@minikube:/# docker ps  | grep nginx
861f9b8c9414   nginx                  "/docker-entrypoint.…"   11 minutes ago      Up 11 minutes               k8s_nginx_nginx_default_d1dbbb28-a1e5-4c3b-952c-d326a7f80e02_0
aef5861f4aad   k8s.gcr.io/pause:3.4.1   "/pause"                 12 minutes ago      Up 12 minutes               k8s_POD_nginx_default_d1dbbb28-a1e5-4c3b-952c-d326a7f80e02_0
root@minikube:/#
```

connecting to the containers

Login to the container and get the ip address

```
root@minikube:/# docker exec -it 861f9b8c9414 /bin/bash
root@nginx:/# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@nginx:/# hostname
nginx
root@nginx:/# hostname -i
172.17.0.5
root@nginx:/# curl 172.17.0.5
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@nginx:/#
```

Get IP from our Machine

```
esak@esakpc:~$ kubectl get pods -o wide
NAME    READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
nginx   1/1     Running   0          17m   172.17.0.5   minikube   <none>           <none>
esak@esakpc:~$
```

lets conenct to the above url from the host Machine

```
esak@esakpc:~$ curl 172.17.0.5
curl: (7) Failed to connect to 172.17.0.5 port 80: No route to host
esak@esakpc:~$
```

We can delete the pods

```
esak@esakpc:~$ kubectl delete pod nginx
pod "nginx" deleted
esak@esakpc:~$ kubectl get pods -o wide
No resources found in default namespace.
esak@esakpc:~$
```

Deployment is responsible for creation of the actual ports . All ports inside the deployment are same.

Lets create deployment

deployment will automatically create a POD. This POD was automatically managed by the deployment

```
esak@esakpc:~$ kubectl create deployment nginxdeployment --image=nginx
deployment.apps/nginxdeployment created
esak@esakpc:~$ kubectl get deployments
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
nginxdeployment   1/1     1            1           11s
esak@esakpc:~$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginxdeployment-6fcdbc7d77-z7gp8   1/1     Running   0          21s
esak@esakpc:~$
```

Get More info about the deployment

ports and deployments are separate objects we have to assign specific ports to particular deployments. PORT was created automatically and it also had the same selector as “app=nginxdeployment”

```
esak@esakpc:~$ kubectl describe deployment nginxdeployment
Name:                   nginxdeployment
Namespace:              default
CreationTimestamp:      Fri, 03 Jun 2022 14:21:47 +0530
Labels:                 app=nginxdeployment
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginxdeployment
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginxdeployment
  Containers:
   nginx:
    Image:        nginx
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
```

Scale the number of pods

we have increased the pods count to 5

```
esak@esakpc:~$ kubectl scale deployment nginxdeployment --replicas=5
deployment.apps/nginxdeployment scaled
esak@esakpc:~$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginxdeployment-6fcdbc7d77-2n95n   1/1     Running   0          23s
nginxdeployment-6fcdbc7d77-8zwvd   1/1     Running   0          23s
nginxdeployment-6fcdbc7d77-f2bjw   1/1     Running   0          23s
nginxdeployment-6fcdbc7d77-gc2gn   1/1     Running   0          23s
nginxdeployment-6fcdbc7d77-z7gp8   1/1     Running   0          15m
esak@esakpc:~$
```

```
esak@esakpc:~$ kubectl get pods -o wide
NAME                               READY   STATUS    RESTARTS   AGE     IP           NODE       NOMINATED NODE   READINESS GATES
nginxdeployment-6fcdbc7d77-2n95n   1/1     Running   0          2m17s   172.17.0.7   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-8zwvd   1/1     Running   0          2m17s   172.17.0.6   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-f2bjw   1/1     Running   0          2m17s   172.17.0.8   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-gc2gn   1/1     Running   0          2m17s   172.17.0.9   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-z7gp8   1/1     Running   0          17m     172.17.0.5   minikube   <none>           <none>
esak@esakpc:~$
```

Lets scale down

we scale down to 3 nodes and 2 more nodes are terminated

```
esak@esakpc:~$ kubectl scale deployment nginxdeployment --replicas=3
deployment.apps/nginxdeployment scaled
esak@esakpc:~$ kubectl get pods -o wide
NAME                               READY   STATUS        RESTARTS   AGE     IP           NODE       NOMINATED NODE   READINESS GATES
nginxdeployment-6fcdbc7d77-2n95n   1/1     Running       0          3m46s   172.17.0.7   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-8zwvd   1/1     Running       0          3m46s   172.17.0.6   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-f2bjw   0/1     Terminating   0          3m46s   172.17.0.8   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-gc2gn   0/1     Terminating   0          3m46s   172.17.0.9   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-z7gp8   1/1     Running       0          18m     172.17.0.5   minikube   <none>           <none>
esak@esakpc:~$ kubectl get pods -o wide
NAME                               READY   STATUS    RESTARTS   AGE     IP           NODE       NOMINATED NODE   READINESS GATES
nginxdeployment-6fcdbc7d77-2n95n   1/1     Running   0          3m57s   172.17.0.7   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-8zwvd   1/1     Running   0          3m57s   172.17.0.6   minikube   <none>           <none>
nginxdeployment-6fcdbc7d77-z7gp8   1/1     Running   0          18m     172.17.0.5   minikube   <none>           <none>
esak@esakpc:~$
```

Services

it allow us to connect to specific deployments using specific IP addresses. There are different options available you could create so called cluster IP , these IP are created

and assigned to specific deployments . We can connect to specific deployments only inside cluster.

There were options to create external IP address to open deployment to outside World , it is possible to expose specific deployments to the IP address of the node or use a load balancer

lets create a cluster IP

Nginx container usually runs at port 80

we want to expose port 8080 to the outside world

```
esak@esakpc:~$ kubectl expose deployment nginxdeployment --port=8080 --target-port=80
service/nginxdeployment exposed
esak@esakpc:~$
```

list the service

```
esak@esakpc:~$ kubectl get services
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
kubernetes        ClusterIP   10.96.0.1        <none>        443/TCP    288d
nginxdeployment   ClusterIP   10.100.205.128   <none>        8080/TCP   85s
esak@esakpc:~$
```

CLUSTERIP is not available for outside of the kubernetes cluster

we can access pods with in the node

we login to the minikube node and we execute the curl command

When we are inside the kubernetes cluster and we can access this IP and get results . This result was provided by any one of the pods in the deployments. we have 3 pods in total

```
esak@esakpc:~$ docker exec -it ad234bc48e57 /bin/bash
root@minikube:/# curl 10.100.205.128:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@minikube:/#
```

Get details about the service

```
esak@esakpc:~$ kubectl get svc
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
kubernetes        ClusterIP   10.96.0.1        <none>        443/TCP    288d
nginxdeployment   ClusterIP   10.100.205.128   <none>        8080/TCP   10m
esak@esakpc:~$ kubectl describe svc nginxdeployment
Name:              nginxdeployment
Namespace:         default
Labels:            app=nginxdeployment
Annotations:       <none>
Selector:          app=nginxdeployment
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                10.100.205.128
IPs:               10.100.205.128
Port:              <unset>  8080/TCP
TargetPort:        80/TCP
Endpoints:         172.17.0.5:80,172.17.0.6:80,172.17.0.7:80
Session Affinity:  None
Events:            <none>
esak@esakpc:~$
```

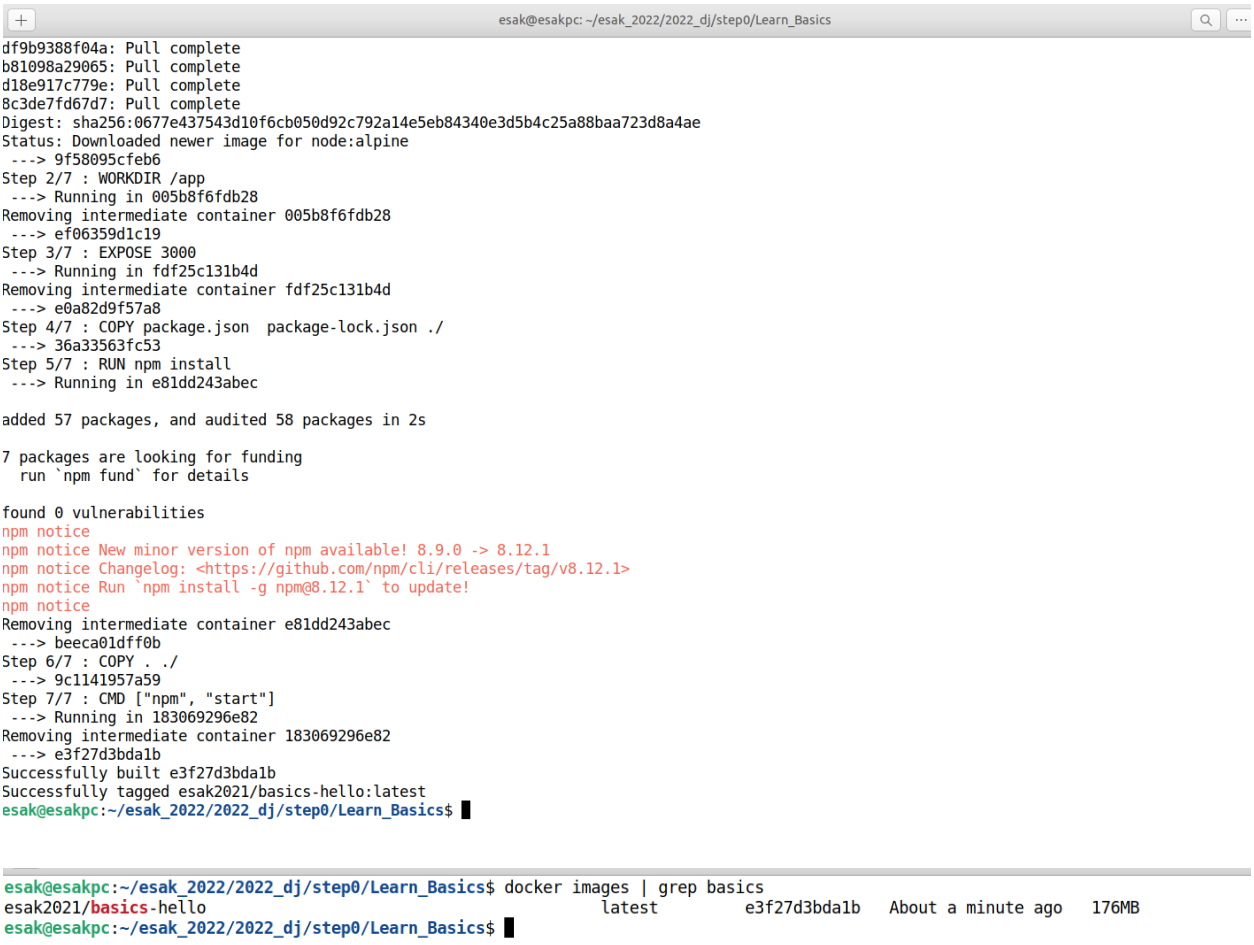
Lets create a simple Application using Docker

Create Docker Image

```
FROM node:alpine
WORKDIR /app
EXPOSE 3000
COPY package.json  package-lock.json ./
RUN npm install
COPY . ./
CMD ["npm", "start"]
```

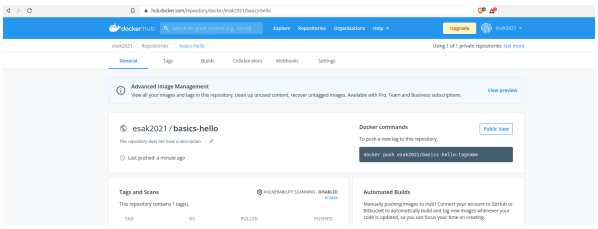
Create the Docker image Locally

```
docker build . -t esak2021/basics-hello
```



Lets push the image to docker hub

```
docker login
docker push esak2021/basics-hello
```




```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/esak/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ docker push esak2021/basics-hello
Using default tag: latest
The push refers to repository [docker.io/esak2021/basics-hello]
e7b21e1e9041: Pushed
8b565512a71a: Pushed
cce58091a875: Pushed
8cfce1e49867: Pushed
d01c433a71f9: Mounted from library/node
92ff8a47a573: Mounted from library/node
b158cee4c019: Mounted from library/node
4fc242d58285: Mounted from library/node
latest: digest: sha256:7aac1131ed7d0040dcd40a7f868788d202f6ea2b94f7daf4a3ba622628e342ce size: 1992
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Lets create a deployment from the image

```
esak@esakpc:~$ kubectl create deployment basics-hello-deploy --image=esak2021/basics-hello
deployment.apps/basics-hello-deploy created
esak@esakpc:~$ kubectl get pods
NAME                                  READY   STATUS              RESTARTS   AGE
basics-hello-deploy-646657f494-hkzp8  0/1     ContainerCreating   0          7s
esak@esakpc:~$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
basics-hello-deploy-646657f494-hkzp8  1/1     Running   0          56s
esak@esakpc:~$
```

Lets create the cluster IP for this deployment

```
esak@esakpc:~$ kubectl expose deployment basics-hello-deploy --port=3000
service/basics-hello-deploy exposed
esak@esakpc:~$
```

Lets check the cluster ip and access it from the k8 cluster

```
kubectl get svc
docker ps
docker exec -it <minikube_container_id> /bin/bash

#Since we are inside the single node k8s
#we are issuing a curl command

curl 10.111.60.166:3000
```

```
esak@esakpc:~$ kubectl get svc
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
basics-hello-deploy  ClusterIP   10.111.60.166   <none>        3000/TCP   106s
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP    291d
esak@esakpc:~$ docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED        STATUS                          PORTS
                                                                                                                  NAMES
ad234bc48e57   gcr.io/k8s-minikube/kicbase:v0.0.25   "/usr/local/bin/entr…"   3 days ago     Up 53 minutes                   127.0.0.1:49157->22/tcp, 127
.0.0.1:49156->2376/tcp, 127.0.0.1:49155->5000/tcp, 127.0.0.1:49154->8443/tcp, 127.0.0.1:49153->32443/tcp   minikube
34401e40b307   mongo-express                      "tini -- /docker-ent…"   2 months ago   Restarting (0) 30 seconds ago
                                                                                                                  techworld-js-docker-demo-app_mongo-express_1
esak@esakpc:~$ docker exec -it ad234bc48e57 /bin/bash
root@minikube:/# curl 10.111.60.166:3000
Hello from the basics-hello-deploy-646657f494-hkzp8root@minikube:/#
```

Lets increase the number of pods in the deployment

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
basics-hello-deploy-646657f494-hkzp8   1/1     Running   0          9m23s
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get deploy
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
basics-hello-deploy   1/1     1            1           22m
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl scale deploy basics-hello-deploy --replicas=4
deployment.apps/basics-hello-deploy scaled
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get deploy
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
basics-hello-deploy   1/4     4            1           22m
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Now the Pods increased to 4

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
basics-hello-deploy-646657f494-hkzp8   1/1     Running   0          23m
basics-hello-deploy-646657f494-hqfb8   1/1     Running   0          71s
basics-hello-deploy-646657f494-jpnnz   1/1     Running   0          71s
basics-hello-deploy-646657f494-zdqxp   1/1     Running   0          71s
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Lets Check the Results

```
root@minikube:/# curl 10.111.60.166:3000; echo
Hello from the basics-hello-deploy-646657f494-hkzp8
root@minikube:/# curl 10.111.60.166:3000; echo
Hello from the basics-hello-deploy-646657f494-jpnnz
root@minikube:/# curl 10.111.60.166:3000; echo
Hello from the basics-hello-deploy-646657f494-hqfb8
root@minikube:/# curl 10.111.60.166:3000; echo
Hello from the basics-hello-deploy-646657f494-zdqxp
root@minikube:/#
```

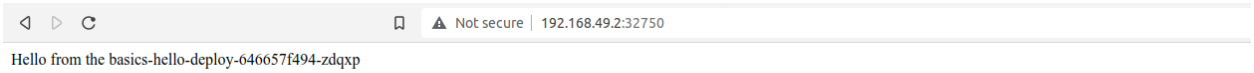
Create a Node Port so that we can expose the port to Browsers

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl delete svc basics-hello-deploy
service "basics-hello-deploy" deleted
```

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl expose deployment basics-hello-deploy --type=NodePort --port=3000
service/basics-hello-deploy exposed
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get svc
NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
basics-hello-deploy   NodePort    10.104.133.108   <none>        3000:32750/TCP   11s
kubernetes           ClusterIP   10.96.0.1        <none>        443/TCP          291d
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

```
To Get IP address
docker inspect -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' ad234bc48e57
```

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ docker inspect -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' ad234bc48e57
192.168.49.2
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```



Hello from the basics-hello-deploy-646657f494-zdqxp

```
esak@esakpc:~$ kubectl get svc
NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
basics-hello-deploy   NodePort    10.104.133.108   <none>        3000:32750/TCP   54m
kubernetes           ClusterIP   10.96.0.1        <none>        443/TCP          291d
esak@esakpc:~$ minikube service basics-hello-deploy
|-----------|---------------------|-------------|---------------------------|
| NAMESPACE |        NAME         | TARGET PORT |            URL            |
|-----------|---------------------|-------------|---------------------------|
| default   | basics-hello-deploy |        3000 | http://192.168.49.2:32750 |
|-----------|---------------------|-------------|---------------------------|
🎉  Opening service default/basics-hello-deploy in default browser...
esak@esakpc:~$ Opening in existing browser session.
libva error: vaGetDriverNameByIndex() failed with unknown libva error, driver_name = (null)
```

Lets Create a Load balancer service

```
kubectl expose deployment basics-hello-deploy --type-LoadBalancer --port=3000
```

```
esak@esakpc:~$ kubectl expose deployment basics-hello-deploy --type=LoadBalancer --port=3000
service/basics-hello-deploy exposed
esak@esakpc:~$ kubectl get svc
NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
basics-hello-deploy  LoadBalancer   10.109.123.135   <pending>     3000:32639/TCP   6s
kubernetes           ClusterIP      10.96.0.1        <none>        443/TCP          291d
esak@esakpc:~$
```

If we check the external Ip , it shows like pending , but in the real World we get a IP from the cloud Provider

```
esak@esakpc:~$ kubectl expose deployment basics-hello-deploy --type=LoadBalancer --port=3000
service/basics-hello-deploy exposed
esak@esakpc:~$ kubectl get svc
NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
basics-hello-deploy  LoadBalancer   10.109.123.135   <pending>     3000:32639/TCP   6s
kubernetes           ClusterIP      10.96.0.1        <none>        443/TCP          291d
esak@esakpc:~$ minikube service basics-hello-deploy --url
http://192.168.49.2:32639
esak@esakpc:~$
```

Lets Do a Rolling Update

Lets update the Code

```
import os from 'os'
import express from 'express'


const app = express()
const PORT=3000

app.get('/' , (req, res) => {
    const helloMessage = `VERSION2::Hello from the ${os.hostname()}`
    console.log(helloMessage)
    res.send(helloMessage)
```

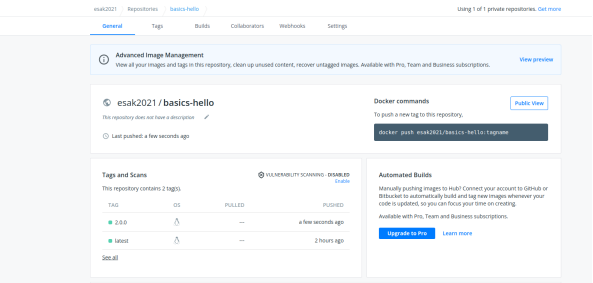
```
})

app.listen(PORT, () => {
    console.log(`Webserver is listening at the PORT ${PORT}`)
})
```

Build docker images

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ docker build -t esak2021/basics-hello:2.0.0 .
Sending build context to Docker daemon  44.03kB
Step 1/7 : FROM node:alpine
 ---> 9f58095cfeb6
Step 2/7 : WORKDIR /app
 ---> Using cache
 ---> ef06359d1c19
Step 3/7 : EXPOSE 3000
 ---> Using cache
 ---> e0a82d9f57a8
Step 4/7 : COPY package.json  package-lock.json ./
 ---> Using cache
 ---> 36a33563fc53
Step 5/7 : RUN npm install
 ---> Using cache
 ---> beeca01dff0b
Step 6/7 : COPY . ./
 ---> c109a42a68b9
Step 7/7 : CMD ["npm", "start"]
 ---> Running in 94c746c322d0
Removing intermediate container 94c746c322d0
 ---> 43ec7d98c207
Successfully built 43ec7d98c207
Successfully tagged esak2021/basics-hello:2.0.0
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ docker push esak2021/basics-hello:2.0.0
The push refers to repository [docker.io/esak2021/basics-hello]
896b70ef31e0: Pushed
8b565512a71a: Layer already exists
cce58091a875: Layer already exists
8cfce1e49867: Layer already exists
d01c433a71f9: Layer already exists
92ff8a47a573: Layer already exists
b158cee4c019: Layer already exists
4fc242d58285: Layer already exists
2.0.0: digest: sha256:bf14082af2d92df56d95565c733936bb0495da286befa5059afebc85a61c8eeb size: 1992
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Checking in dockerhub



Lets do the rolling update

```
kubectl set image deployment basics-hello-deploy basics-hello=esak2021/basics-hello:2.0.0
deployment basics-hello-deploy  ---> Deployment Name
```

```
 basics-hello  == Pods Nmae
esak2021/basics-hello:2.0.0   ===> New Image Name
```

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl set image deployment basics-hello-deploy basics-hello=esak2021/basics-hello:2.0.0
deployment.apps/basics-hello-deploy image updated
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

```
kubectl rollout status deploy basics-hello-deploy
```

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl rollout status deploy basics-hello-deploy
deployment "basics-hello-deploy" successfully rolled out
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl rollout status deploy basics-hello-deploy
Waiting for deployment "basics-hello-deploy" rollout to finish: 2 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 2 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 2 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 3 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 3 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 3 out of 4 new replicas have been updated...
Waiting for deployment "basics-hello-deploy" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "basics-hello-deploy" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "basics-hello-deploy" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "basics-hello-deploy" rollout to finish: 3 of 4 updated replicas are available...
deployment "basics-hello-deploy" successfully rolled out
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Lets Delete a PODS

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get pods
NAME                                 READY   STATUS    RESTARTS   AGE
basics-hello-deploy-578884f678-cxtq4   1/1     Running   0          23m
basics-hello-deploy-578884f678-lgbmz   1/1     Running   0          23m
basics-hello-deploy-578884f678-mmm2w   1/1     Running   0          23m
basics-hello-deploy-578884f678-zsh9l   1/1     Running   0          23m
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl delete pod basics-hello-deploy-578884f678-mmm2w
pod "basics-hello-deploy-578884f678-mmm2w" deleted
```

Once the POD is deleted another POD will be created automatically .Since the desired count is 4 , it try to keep the count

```
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
basics-hello-deploy-578884f678-cxtq4   1/1     Running   0          23m
basics-hello-deploy-578884f678-lgbmz   1/1     Running   0          23m
basics-hello-deploy-578884f678-mmm2w   1/1     Running   0          23m
basics-hello-deploy-578884f678-zsh9l   1/1     Running   0          23m
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl delete pod basics-hello-deploy-578884f678-mmm2w
pod "basics-hello-deploy-578884f678-mmm2w" deleted
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
basics-hello-deploy-578884f678-cxtq4   1/1     Running   0          24m
basics-hello-deploy-578884f678-lgbmz   1/1     Running   0          24m
basics-hello-deploy-578884f678-vppt7   1/1     Running   0          32s
basics-hello-deploy-578884f678-zsh9l   1/1     Running   0          24m
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```

Launch Kubernetes Dashboard

```
minikube dashboard
```

To delete all the resources

```
kubectl delete all --all
```

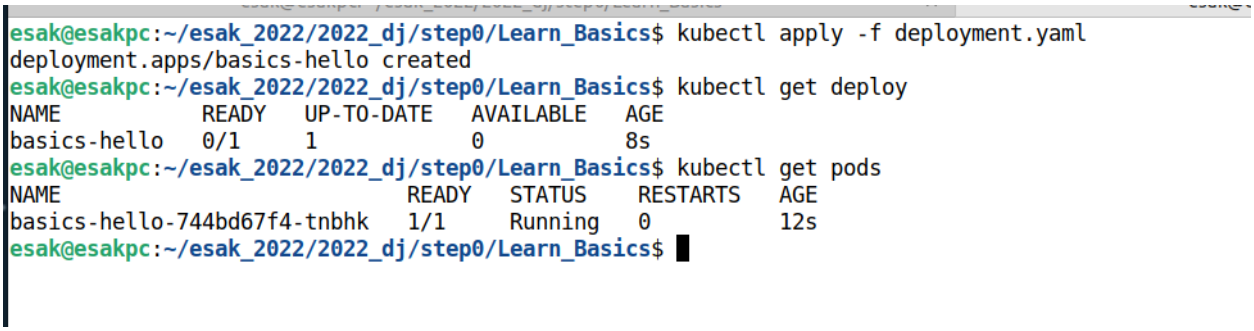
Using YAML Files

Create the YAML File

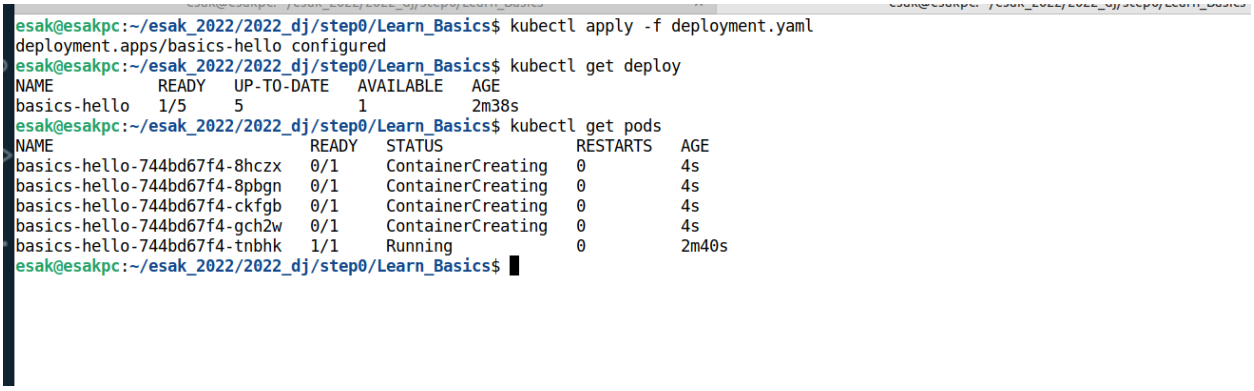
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: basics-hello
spec:
  replicas: 5
  selector:
    matchLabels:
      app: basics-hello
  template:
    metadata:
      labels:
```

```
        app: basics-hello
    spec:
      containers:
      - name: basics-hello
        image: esak2021/basics-hello:2.0.0
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 3000
```

Apply the chnages to minikube

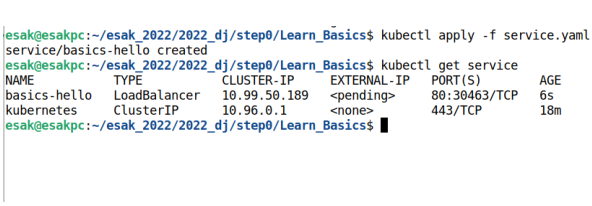


Add replicas as 5 and rerun the apply command



Create a Load Balancer service using yaml

```
apiVersion: v1
kind: Service
metadata:
  name: basics-hello
spec:
  type: "LoadBalancer"
```



```
  selector:
    app: basics-hello
  ports:
  - port: 80
    targetPort: 3000
```

```
minikube service basics-hello
will open the url in the browse
```

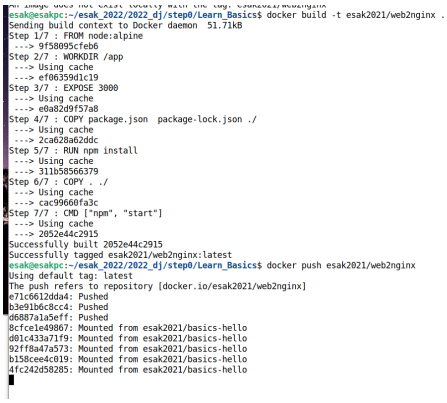
Create another endpoints

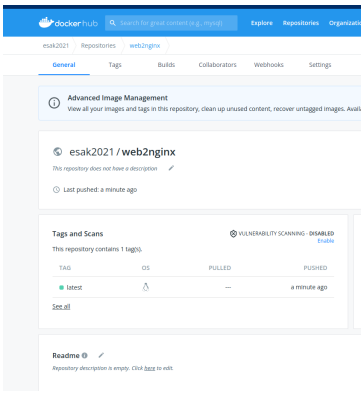
Lets create a ngix service and we try to communicate with that service from our code

```
import os from 'os'
import express from 'express'
import fetch from 'node-fetch'

const app = express()
const PORT=3000

app.get('/' , (req, res) => {
    const helloMessage = `VERSION2::Hello from the ${os.hostname()}`
    console.log(helloMessage)
    res.send(helloMessage)
})

app.get("/nginx", async (req, res) => {
    const url ="http://nginx"
    const response = await fetch(url)
    const body = await response.text()
    res.send(body)
})
app.listen(PORT, () => {
    console.log(`Webserver is listening at the PORT ${PORT}`)
})
```

Build Docker image

```
docker build -t esak2021/web2nginx .
docker push esak2021/web2nginx
```





Lets create the web2nginx

```
apiVersion: v1
kind: Service
metadata:
  name: web2nginx
spec:
  type: LoadBalancer
  selector:
    app: web2nginx
  ports:
  - port: 3333
    targetPort: 3000

---

apiVersion: apps/v1
kind: Deployment
metadata:
  name: web2nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web2nginx
  template:
    metadata:
      labels:
        app: web2nginx
    spec:
      containers:
      - name: web2nginx
        image: esak2021/web2nginx
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
```

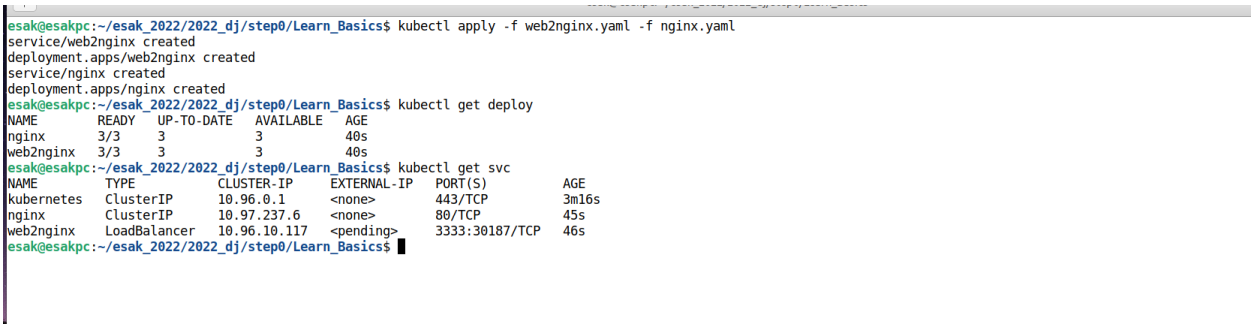
Create a nginx deployment File

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  selector:
    app: nginx
  ports:
  - port: 80

---

apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
```

```
        ports:
        - containerPort: 3000
```

```
        ports:
        - containerPort: 80
```

Lets Deploy Both of these Files

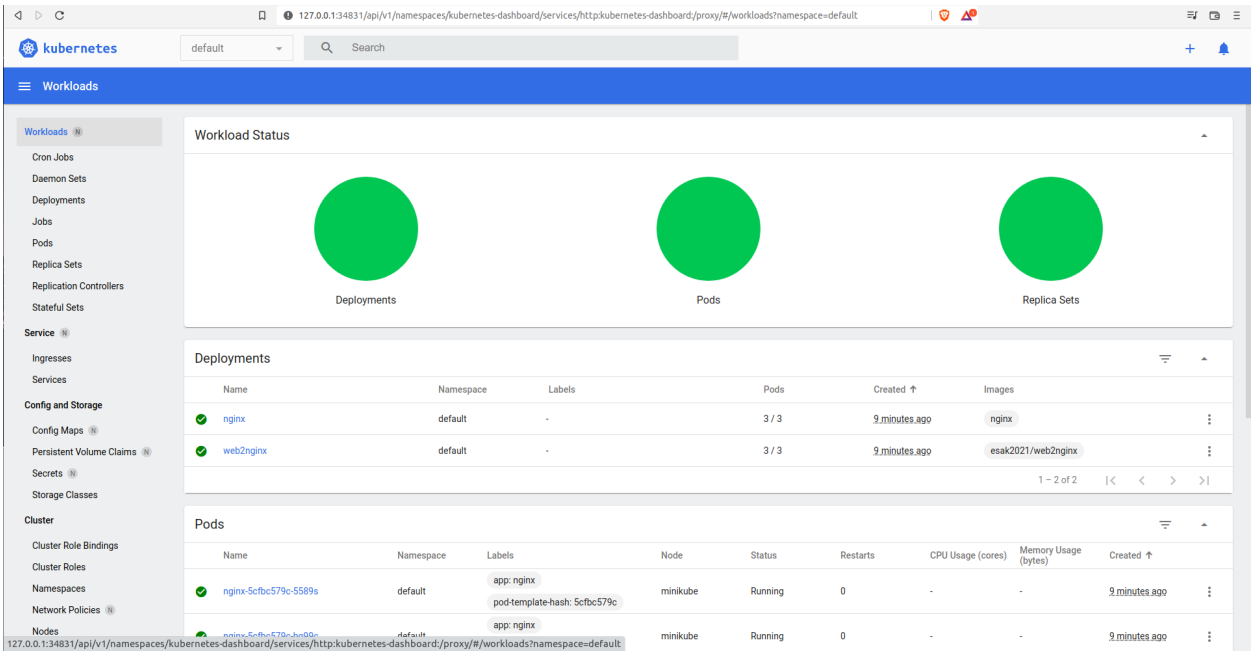


Once everything is deployed lets open up the Load Balancer

when we navigate to the /nginx path we are communicating with nginx server

Mini kube provides simple dashboard to view the details

```
minikube dashboard
```



Delete the Resources

```
esak@esakpc: ~/esak_2022/2022_dj/step0/Learn_Basics
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$ kubectl delete -f web2nginx.yaml -f nginx.yaml
service "web2nginx" deleted
deployment.apps "web2nginx" deleted
service "nginx" deleted
deployment.apps "nginx" deleted
esak@esakpc:~/esak_2022/2022_dj/step0/Learn_Basics$
```