HPA - Horizontal Pod Autoscaling

API server - ?

Rollout - ?

Helm charts - ? Install ingress in helm charts ?

Stateful set vs deployemts ?

Gateway api in k8s vs ingress ?

Service mesh ?

Secrets

kubectl expose deployment nginx-pvc-deployment --type=NodePort --port=80

kubectl expose deployment nginx-pvc-deployment --type=NodePort --port=80

 is a command used in Kubernetes to create a NodePort service that exposes your nginx-pvc-deployment.

This command will:

1. Look up the deployment named nginx-pvc-deployment.
2. Use the selector of that deployment to select the pods running your nginx application.
3. Create a new Kubernetes service of type NodePort.
4. Configure the service to listen on port 80.
5. Route traffic from port 80 on the service to the pods matching the deployment's selector.

###PS C:\Users\Gnana raj\kubernetes-kind\storage> kubectl port-forward service/nginx-pvc-deployment 8080:80

Error from server (NotFound): services "nginx-pvc-deployment" not found

PS C:\Users\Gnana raj\kubernetes-kind\storage> kubectl get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 18m

PS C:\Users\Gnana raj\kubernetes-kind\storage> kubectl expose deployment nginx-pvc-deployment --type=NodePort --port=80

service/nginx-pvc-deployment exposed

PS C:\Users\Gnana raj\kubernetes-kind\storage> kubectl get svc nginx-pvc-deployment

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

nginx-pvc-deployment NodePort 10.96.170.116 <none> 80:30395/TCP 53s###

When you run wget <http://localhost:30395> from *inside* the debug-pod, localhost (or 127.0.0.1) refers to the **network interface of the debug-pod itself**.

Your Nginx service (nginx-pvc-deployment) is listening on its own Pod's IP address on port 80, and the NodePort service (which has IP 10.96.170.116 and port 80:30395/TCP) provides a way to access that service.

So, from inside your debug-pod, localhost:30395 means "try to connect to my *own* Pod on port 30395", which doesn't have Nginx running on it, nor is port 30395 open on the debug pod. That's why you get "Connection refused."

# From inside the debug-pod

/ # wget <http://nginx-pvc-deployment:80>

kubectl run -it --rm --restart=Never debug-pod --image=busybox -- sh

**#for testing, creating a temporary pod**

**When we create a KIND cluster add extraport mapping becaz container runs in a separate network and laptop runs in a separate network so to bypass it you need a same port mapping so the container port and host port are same**

**When we do localhost:port then it should take us to the service that has the port.**

**apiVersion: kind.x-k8s.io/v1alpha4**

**kind: Cluster**

**nodes:**

**- role: control-plane**

**extraPortMappings:**

**- containerPort: 30000**

**hostPort: 30000**

**listenAddress: "0.0.0.0" # Optional, defaults to "0.0.0.0"**

**protocol: tcp # Optional, defaults to tcp**

**Here the when I do localhost:30000 traffic takes me to the service that also has 30000 port, becaz I created a cluster with 30000:30000**

**# app-a-nodeport-service.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: app-a-nodeport-service # Renamed to avoid conflict and be clear**

**spec:**

**type: NodePort # <--- Change this**

**selector:**

**app: app-a**

**ports:**

**- protocol: TCP**

**port: 80**

**targetPort: 5678**

**nodePort: 30000 # <--- Optional: specify a port, otherwise K8s assigns one**

**docker inspect -f '{{.NetworkSettings.Networks.kind.IPAddress}}' my-first-cluster-control-plane  
#to Get KIND Node IP**

**kubectl exec -it <pod-name> -- /bin/bash**

**Go inside a container**

**Kubectl auth can-I create pods###checks the permission for actions**

**O/p; yes/no**

**Kubectl auth can-I create pods --as="system:serviceaccount:default:test-sa"**

**O/p; yes/no ###checks the permission for actions as a service account(test-sa**