Create the K8s EKS,further you have to do the deployment of the Nginx application and access the application outside the cluster.

Sir am tried to do in EKS but not getting output so i tried in local

```
root@ip-172-31-32-86:/home/ubuntu# kubectl get namespace
NAME                   STATUS   AGE
class                  Active   27m
default                Active   30m
kube-node-lease        Active   30m
kube-public            Active   30m
kube-system            Active   30m
kubernetes-dashboard   Active   23m
root@ip-172-31-32-86:/home/ubuntu# kubectl get deployments -n class
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
team    1/1     1            1           27m
root@ip-172-31-32-86:/home/ubuntu# kubectl get pods -n class
NAME                    READY   STATUS    RESTARTS   AGE
team-749bd8dcb9-f77pj   1/1     Running   0          27m
root@ip-172-31-32-86:/home/ubuntu# kubectl get service -n class
NAME    TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
team    LoadBalancer   10.98.85.248   <pending>     80:31792/TCP   18m
root@ip-172-31-32-86:/home/ubuntu# minikube service team -n class
|-----------|------|-------------|---------------------------|
| NAMESPACE | NAME | TARGET PORT |            URL            |
|-----------|------|-------------|---------------------------|
| class     | team |          80 | http://192.168.49.2:31792 |
|-----------|------|-------------|---------------------------|
* Opening service class/team in default browser...
  http://192.168.49.2:31792
```

<u>Create the K8s in windows instead of EKS,further you have to do the deployment of the Nginx application and access the application outside the cluster</u>

STEP 1 - check minikube status then create the namespace and create deployment by pulling the image from docker hub

```
PS P:\Aws Devops\Sample codes> minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

PS P:\Aws Devops\Sample codes> kubectl create namespace team
namespace/team created
PS P:\Aws Devops\Sample codes> kubectl get namespace
NAME                   STATUS   AGE
default                Active   3d1h
guvi                   Active   3h53m
kube-node-lease        Active   3d1h
kube-public            Active   3d1h
kube-system            Active   3d1h
kubernetes-dashboard   Active   3d1h
team                   Active   12s
PS P:\Aws Devops\Sample codes> kubctl create deployent class --image=nginx -n team
```

STEP 2 - set the replicates and run the pods check in dashboard

```
PS P:\Aws Devops\Sample codes> kubectl scale deployment class --replicas=2 -n team
deployment.apps/class scaled
PS P:\Aws Devops\Sample codes> kubectl get pods -n team
NAME                 READY   STATUS    RESTARTS   AGE
class-cc4ff767f-gsvv9   1/1     Running   0          21s
class-cc4ff767f-k8x85   1/1     Running   0          77s
PS P:\Aws Devops\Sample codes> minikube dashboard
🤔  Verifying dashboard health ...
🚀  Launching proxy ...
🤔  Verifying proxy health ...
🎉  Opening http://127.0.0.1:56983/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
```

STEP 3 - expose the deployment outside the world

```
PS P:\Aws Devops\Sample codes> kubectl expose deployment class --port=80 --type=LoadBalancer -n team
service/class exposed
PS P:\Aws Devops\Sample codes> kubectl get services -n team
NAME    TYPE          CLUSTER-IP      EXTERNAL-IP   PORT(S)       AGE
class   LoadBalancer   10.111.130.170   <pending>     80:31643/TCP   18s
```

STEP 4 - run the minikube service to deploy deployment its stateless

```
PS P:\Aws Devops\Sample codes> minikube service class -n
Error: flag needs an argument: 'n' in -n
See 'minikube service --help' for usage.
PS P:\Aws Devops\Sample codes> minikube service class -n team
|-----------|-------|-------------|--------------------------|
| NAMESPACE | NAME  | TARGET PORT |           URL            |
|-----------|-------|-------------|--------------------------|
| team      | class |          80 | http://192.168.49.2:31643 |
|-----------|-------|-------------|--------------------------|
🏃  Starting tunnel for service class.
|-----------|-------|-------------|------------------------|
| NAMESPACE | NAME  | TARGET PORT |          URL           |
|-----------|-------|-------------|------------------------|
| team      | class |             | http://127.0.0.1:57050 |
|-----------|-------|-------------|------------------------|
🎉  Opening service team/class in default browser...
❗  Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

STEP 5 - now automatically u taken to the browser to view our deployment