**3. Review Ingress CPR, Deploy and Verify**

--- **note** – in this lesion, we will discuss about ALB ingress context path-based routing.

**Create ALB Ingress Context path-based Routing Kubernetes manifest**

--- **04-ALB-Ingress-ContextPath-Based-Routing.yml**

# Annotations Reference: https://kubernetes-sigs.github.io/aws-load-balancer-controller/latest/guide/ingress/annotations/

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: ingress-cpr-demo # the name of my service.

  annotations:

    # Load Balancer Name

    alb.ingress.kubernetes.io/load-balancer-name: cpr-ingress

    # Ingress Core Settings

    #kubernetes.io/ingress.class: "alb" (OLD INGRESS CLASS NOTATION - STILL WORKS BUT RECOMMENDED TO USE IngressClass Resource)

    alb.ingress.kubernetes.io/scheme: internet-facing

    # Health Check Settings

    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP

    alb.ingress.kubernetes.io/healthcheck-port: traffic-port

    #Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    alb.ingress.kubernetes.io/healthcheck-interval-seconds: '15'

    alb.ingress.kubernetes.io/healthcheck-timeout-seconds: '5'

    alb.ingress.kubernetes.io/success-codes: '200'

    alb.ingress.kubernetes.io/healthy-threshold-count: '2'

    alb.ingress.kubernetes.io/unhealthy-threshold-count: '2'

spec:

  ingressClassName: my-aws-ingress-class   # Ingress Class

  rules:

    - http:

        paths:

          - path: /app1 # when the request comes to /app1 then it will redirected to app1-nginx-nodeport-service.

            pathType: Prefix

            backend:

              service:

                name: app1-nginx-nodeport-service

                port:

                  number: 80

          - path: /app2

            pathType: Prefix

            backend:

              service:

                name: app2-nginx-nodeport-service

                port:

                  number: 80

          - path: /

            pathType: Prefix

            backend:

              service:

                name: app3-nginx-nodeport-service

                port:

                  number: 80

# Important Note-1: In path based routing order is very important, if we are going to use  "/\*", try to use it at the end of all rules.

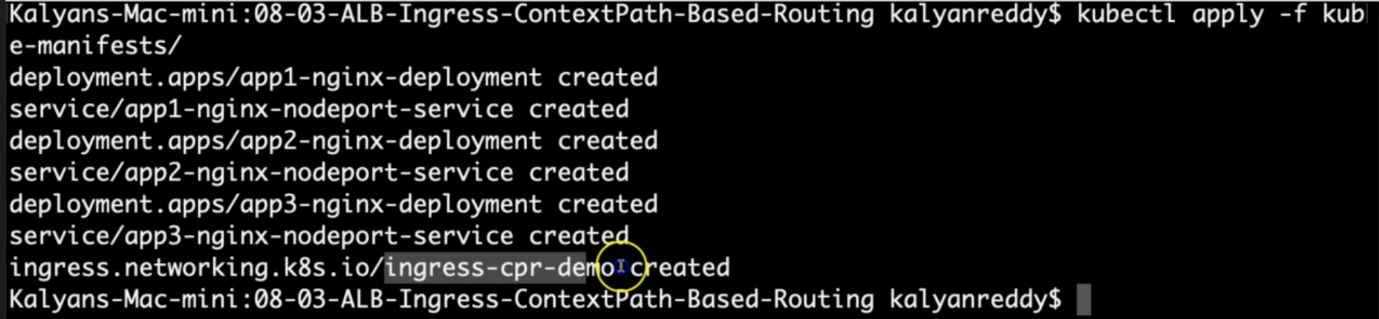
# 1. If  "spec.ingressClassName: my-aws-ingress-class" not specified, will reference default ingress class on this kubernetes cluster

# 2. Default Ingress class is nothing but for which ingress class we have the annotation `ingressclass.kubernetes.io/is-default-class: "true"`

**Deploy all manifests and test**

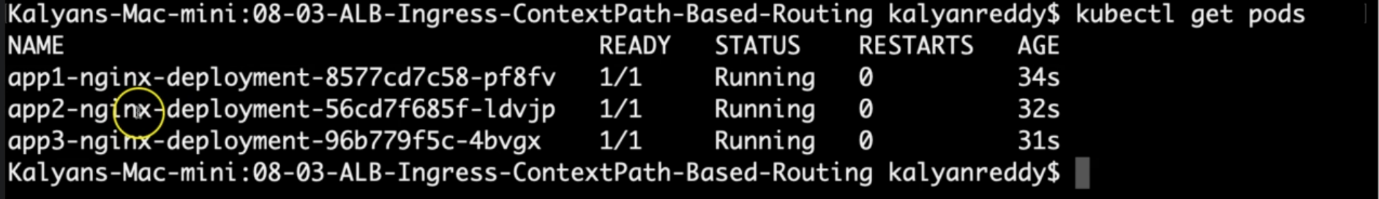
**# Deploy Kubernetes manifests**

--- **kubectl apply -f kube-manifests/**



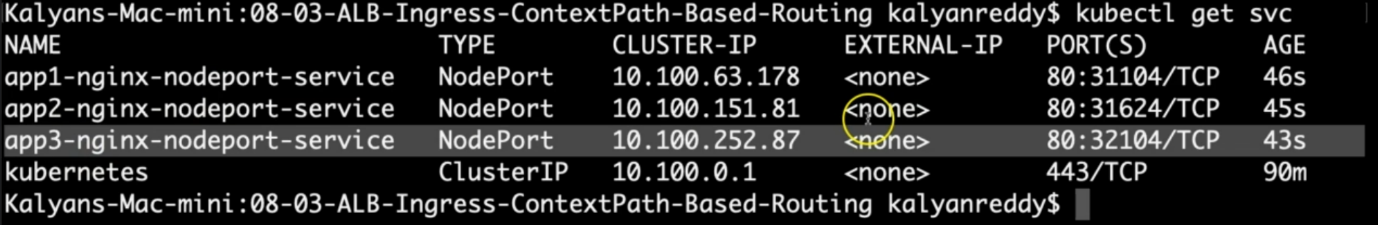
**# List Pods**

--- **kubectl get pods**



**# List Services**

--- **kubectl get svc**



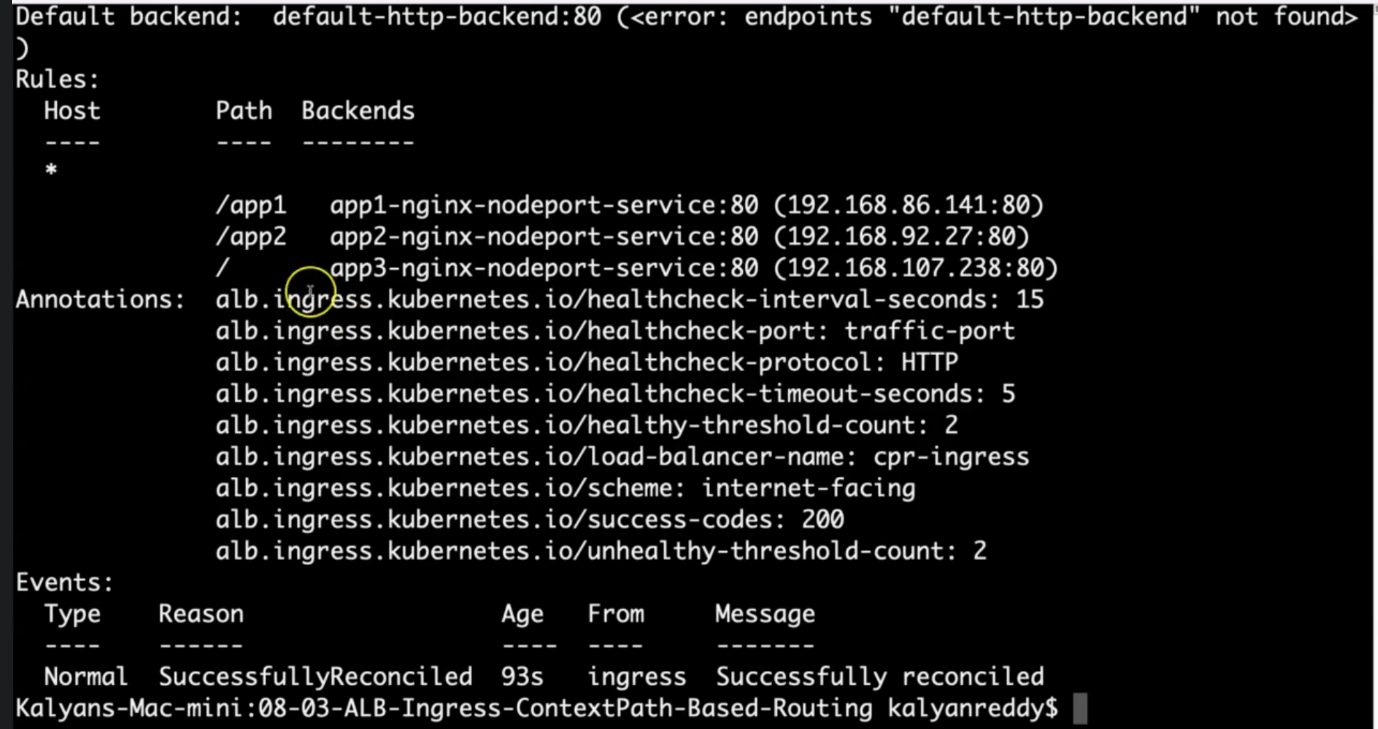
**# List Ingress Load Balancers**

--- **kubectl get ingress**



**# Describe Ingress and view Rules**

--- **kubectl describe ingress ingress-cpr-demo**



--- **note** – successfully reconciled, there is no error logs also.

**# Verify AWS Load Balancer Controller logs**

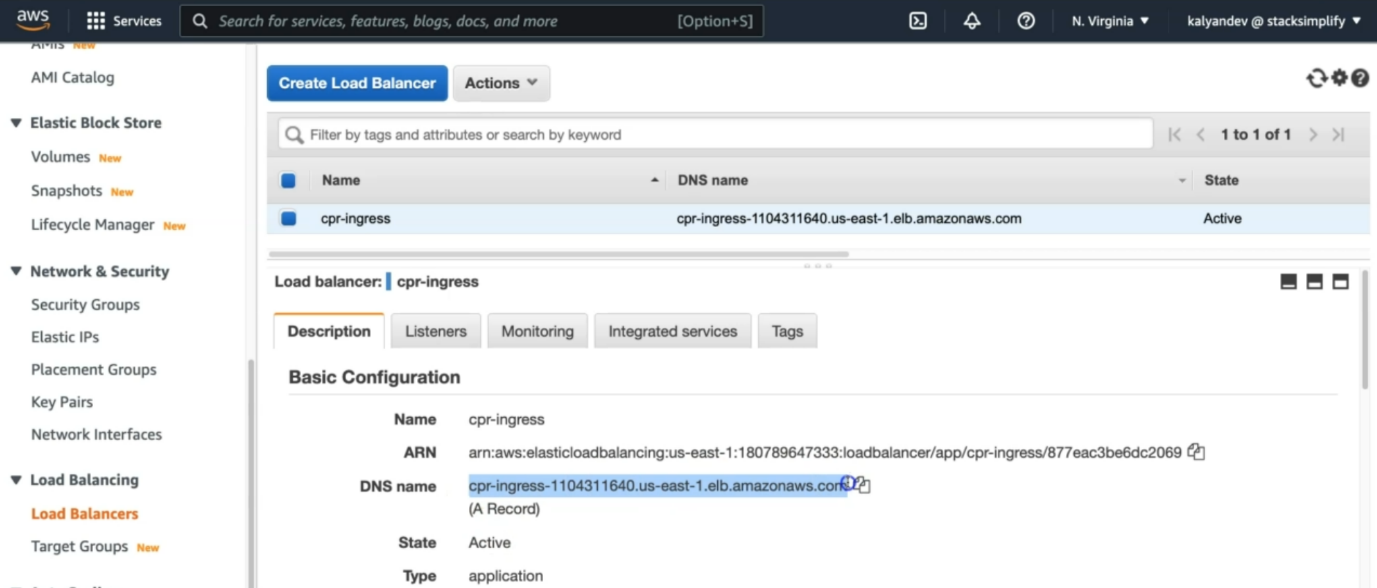
--- **kubectl -n kube-system get pods**

--- **kubectl -n kube-system logs -f aws-load-balancer-controller-794b7844dd-8hk7n**

--- **note** – if your ingress load balancer is not working then verify the aws load balancer pod log. The aws load balancer pod is nothing but ingress load balancer.

**Verify Application Load Balancer on AWS Management Console\*\***

--- **Verify Load Balancer**

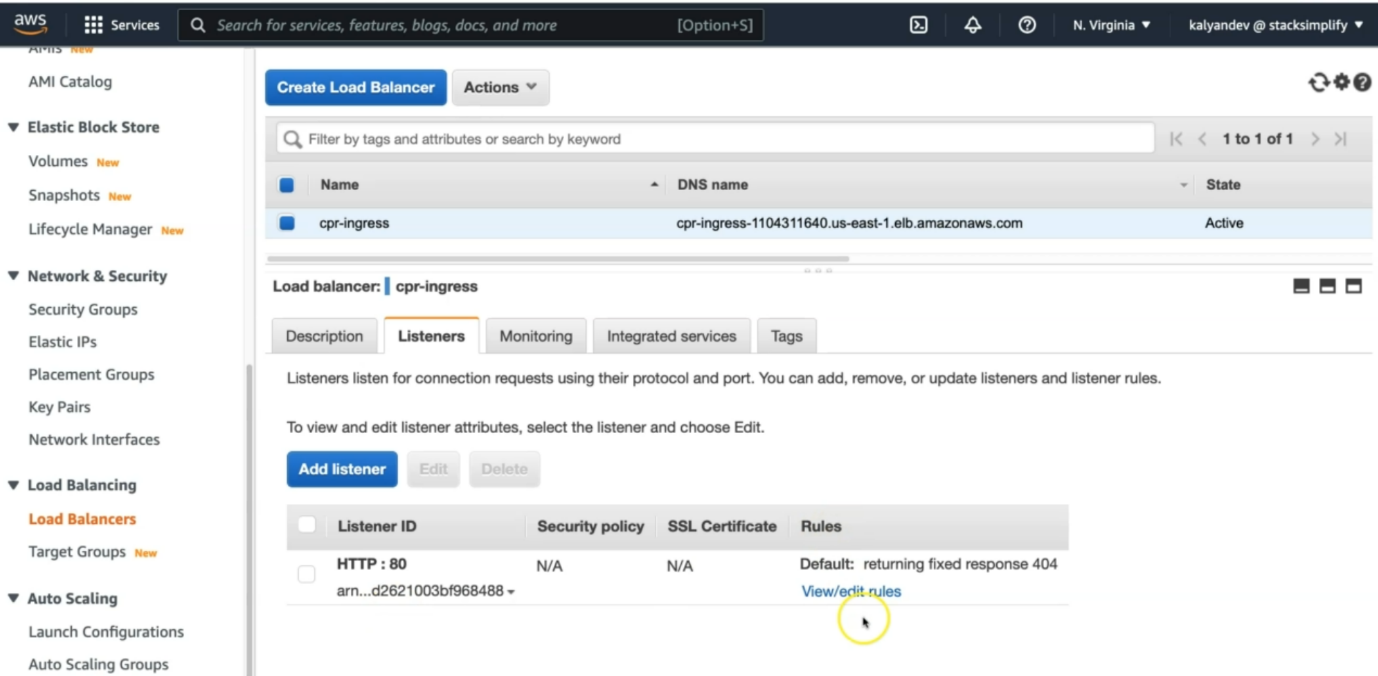


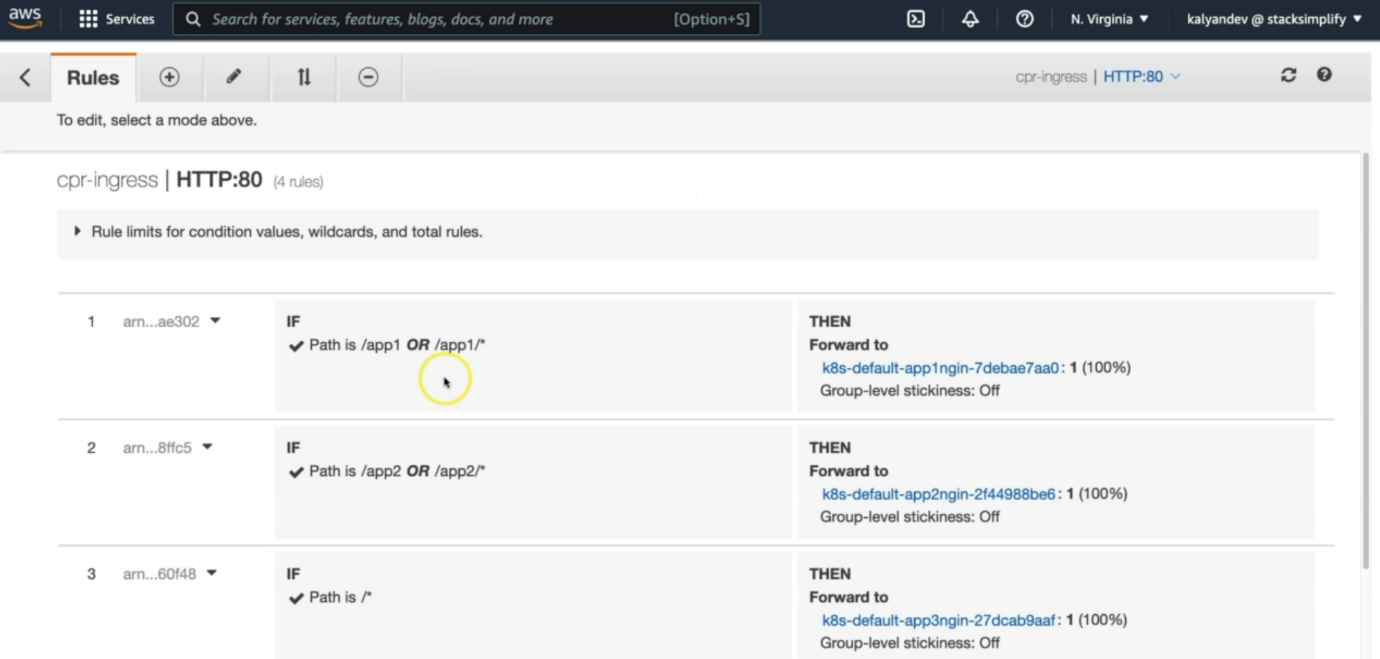
--- this is the load balancer, we created and you can also see the load balancer dns name here.

--- **kubectl get ingress**



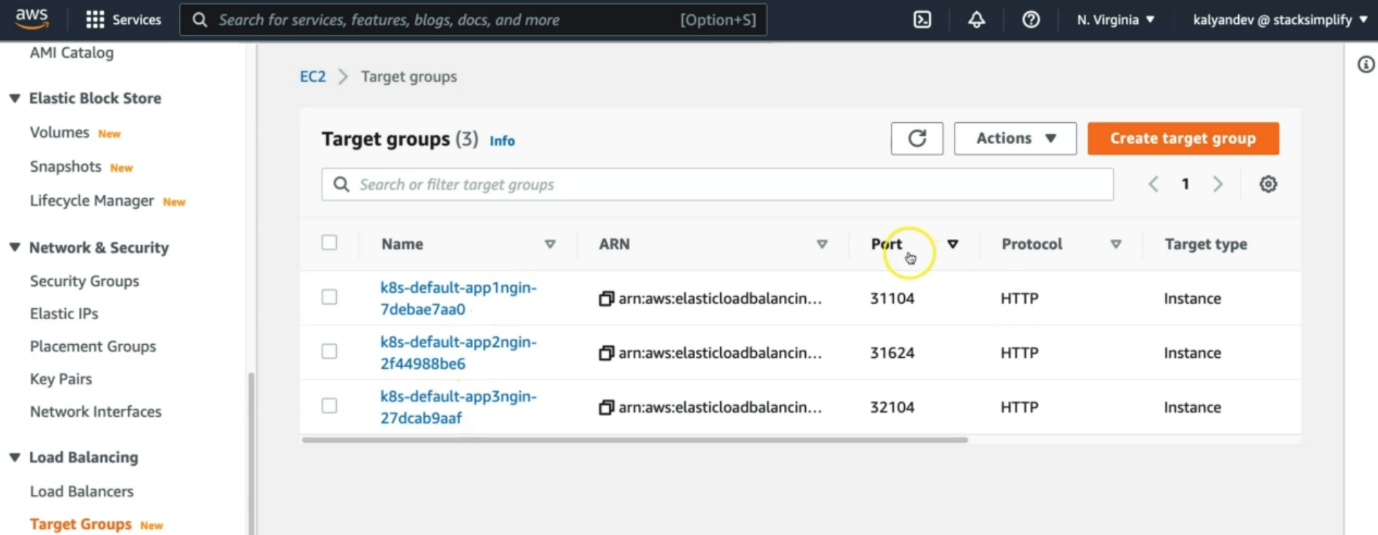
--- **note** - the ingress address is matching with load balancer DNS name. click on listeners.

--- In Listeners Tab, click on View/Edit Rules under Rules

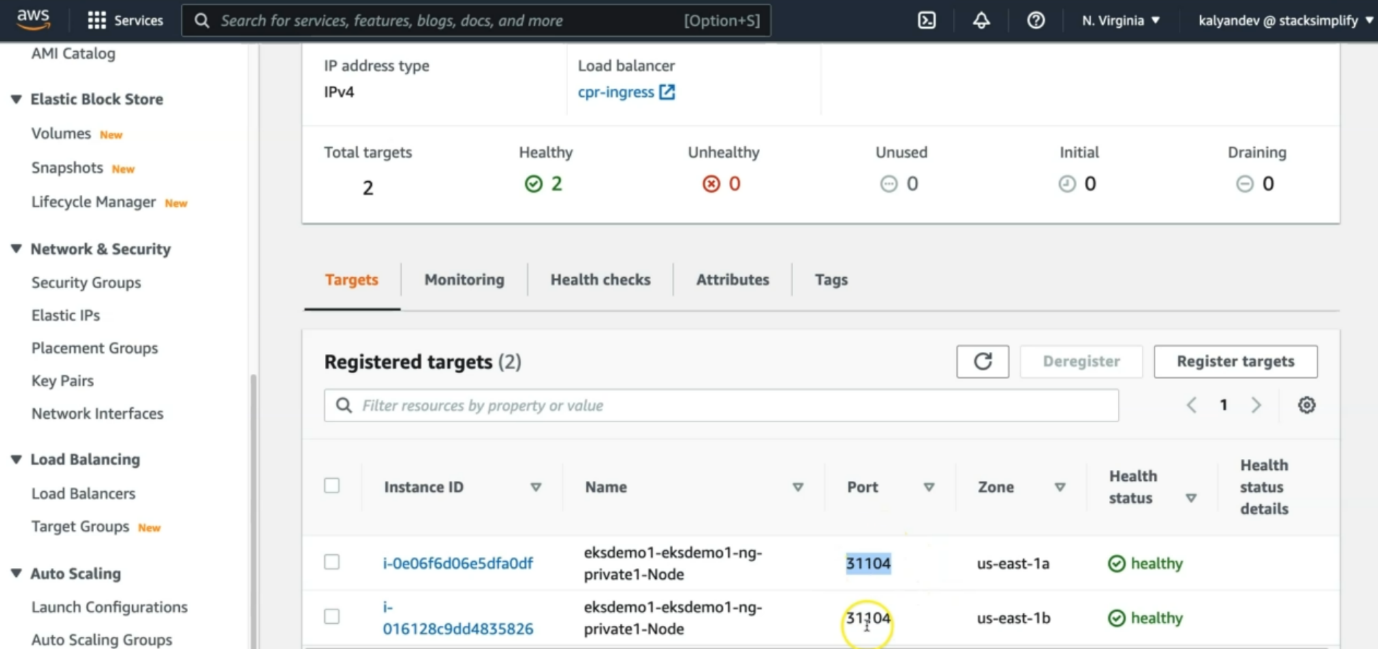


--- **Verify Target Groups**

1. Group D Details
2. Targets: Ensure they are healthy
3. Verify Health check path
4. Verify all 3 targets are healthy)



--- **note** – you have total 3 target groups, there are app1nginx, app2nginx, app3nginx and all are using nodeport service. Click on any target group.



--- we have 2 worker nodes are registered and the worker nodes are healthy. If you want then you can check health check path.

--- even though you created only pod but your node port service is spread over 2 worker nodes. That is why the 2 worker nodes are registered. If your node port service is spread over 10 worker nodes then those 10 worker nodes will be registered under this target group.

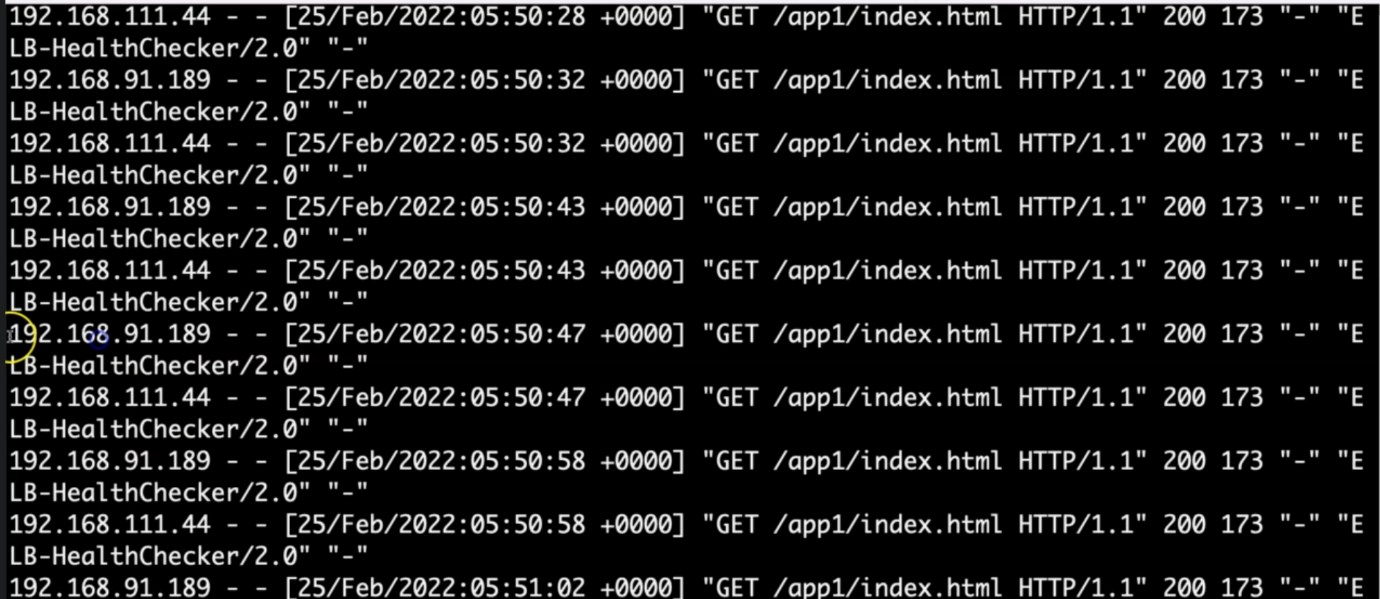
**# Access Application**

--- **http://<ALB-DNS-URL>/app1/index.html**

--- **http://<ALB-DNS-URL>/app2/index.html**

--- **http://<ALB-DNS-URL>/**

--- **kubectl logs -f app1-nginx** -



--- **note** – the health check path is also using the same thing.