Ingress Demonstration

- ✓ Path-Based Routing
- ✓ Host-Based Routing
- ✓ SSL Certificate

1

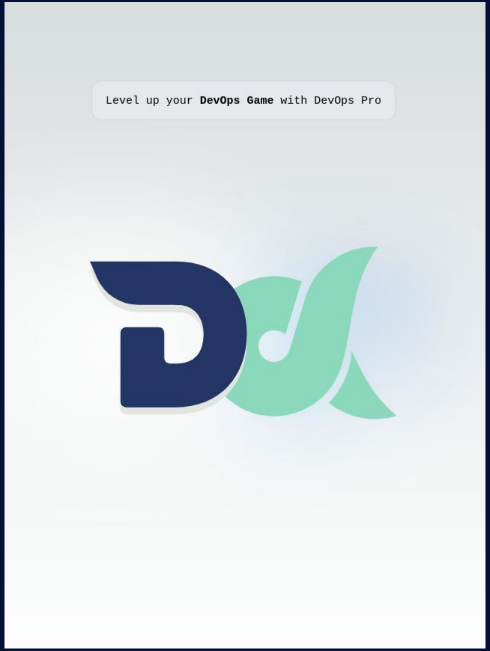
https://devopspro.in

DNS Record

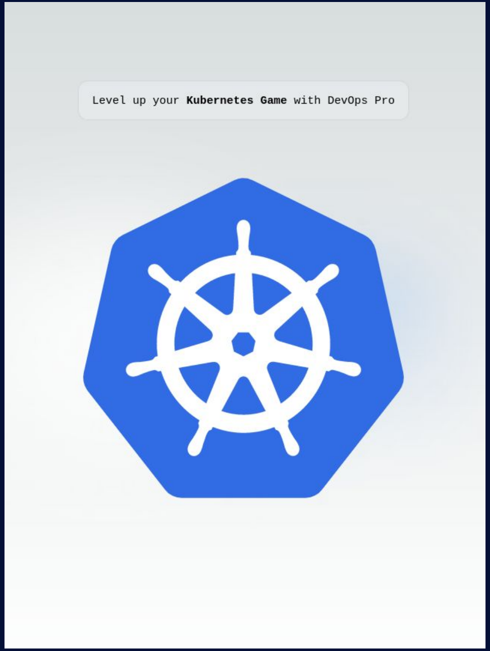
Type	Name	Value
A	example.devopspro.in	11.22.33.44
Type	Name	Value
A	sample-1.devopspro.in	11.22.33.44
Type	Name	Value
A	sample-2.devopspro.in	11.22.33.44

2

Next JS Applications



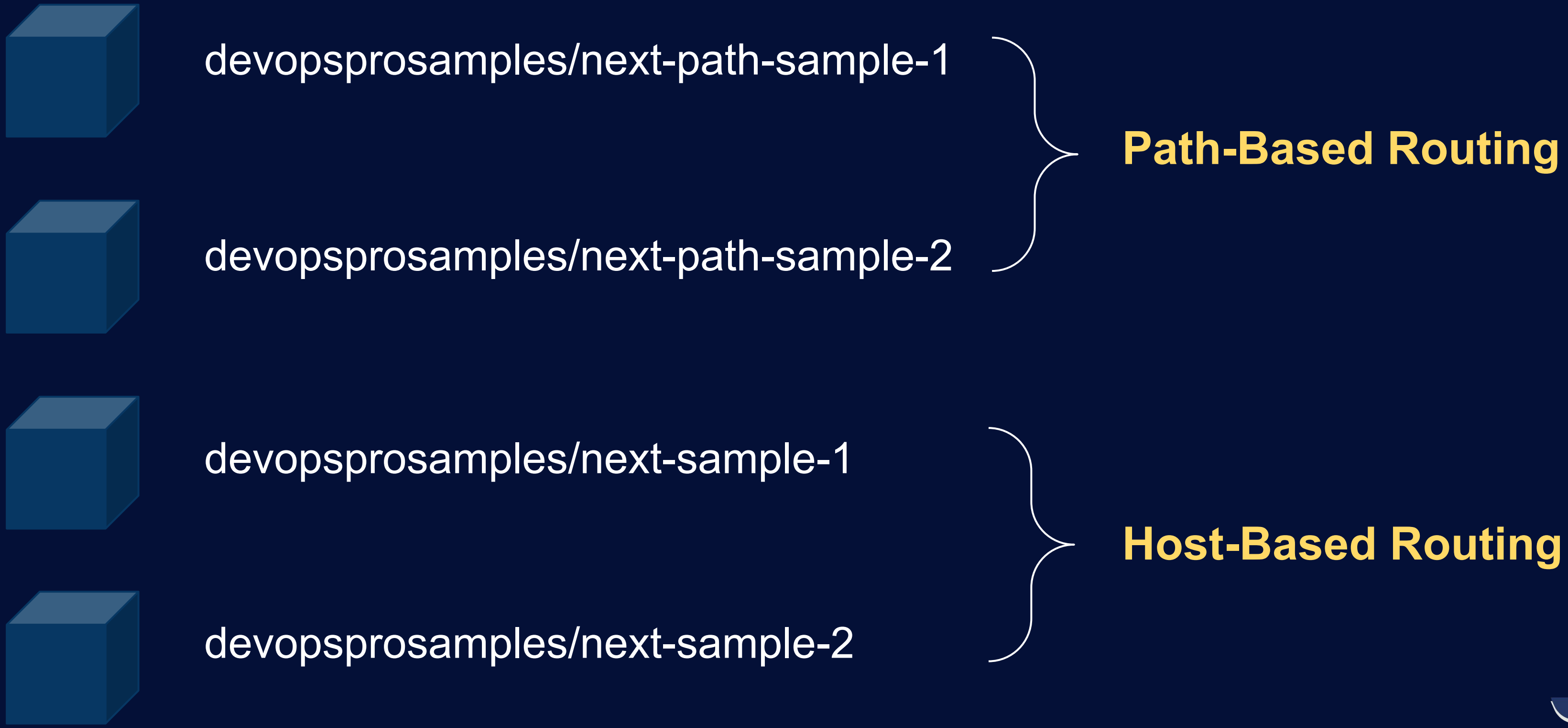
Level up your **DevOps Game** with DevOps Pro



Level up your **Kubernetes Game** with DevOps Pro

3

Docker images



devopsprosamples/next-path-sample-1

devopsprosamples/next-path-sample-2

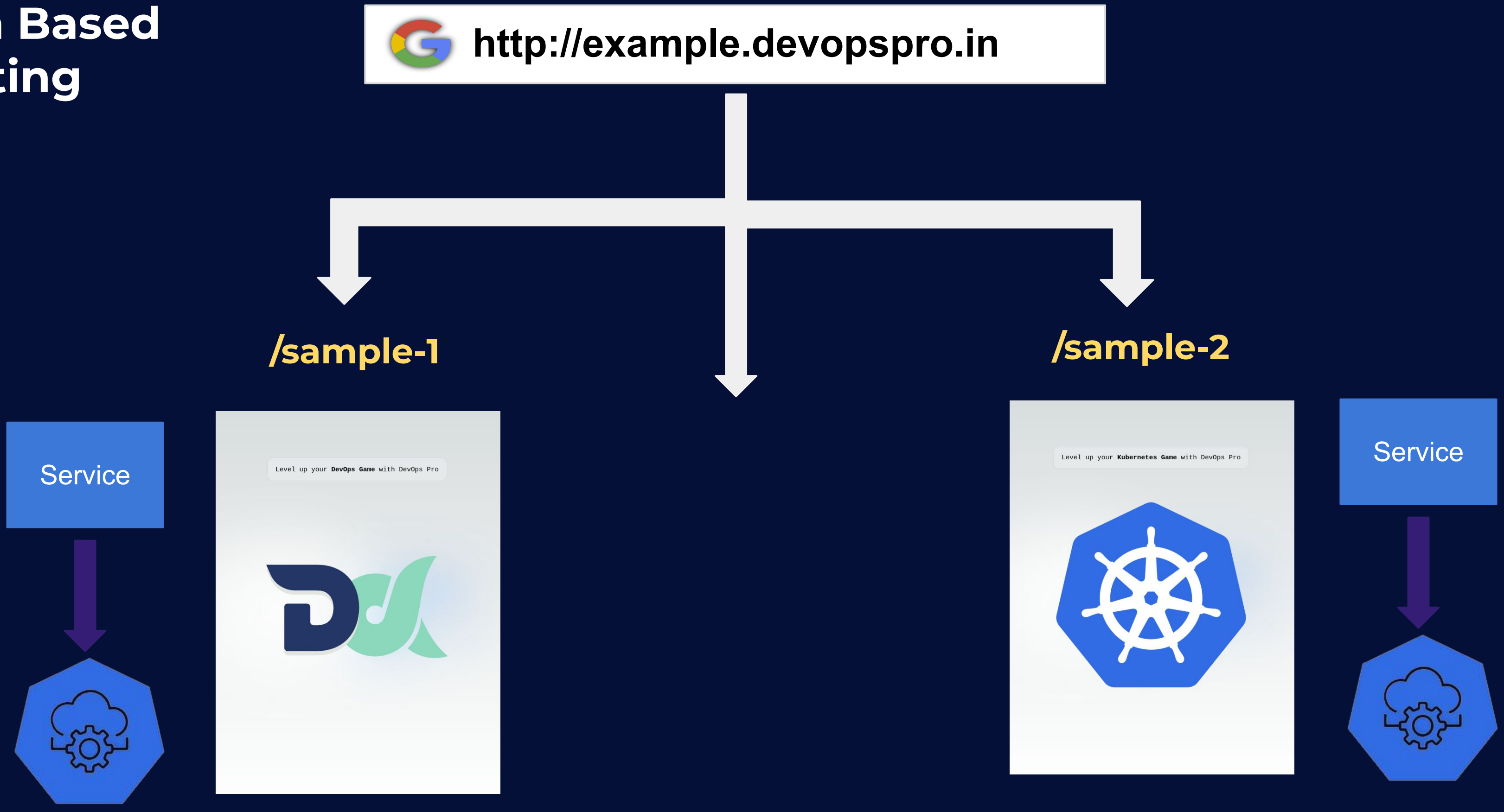
Path-Based Routing

devopsprosamples/next-sample-1

devopsprosamples/next-sample-2

Host-Based Routing

Path Based Routing



http://example.devopspro.in

/sample-1

/sample-2

Service

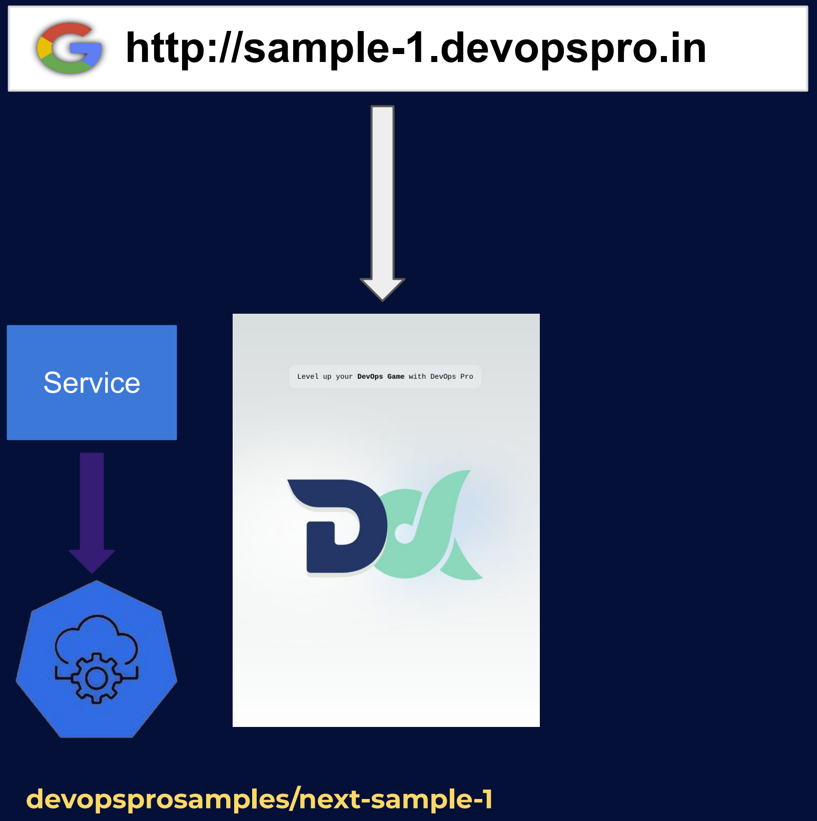
Service

Level up your DevOps Game with DevOps Pro
Level up your Kubernetes Game with DevOps Pro

devopsprosamples/next-path-sample-1

devopsprosamples/next-path-sample-2

Host Based Routing

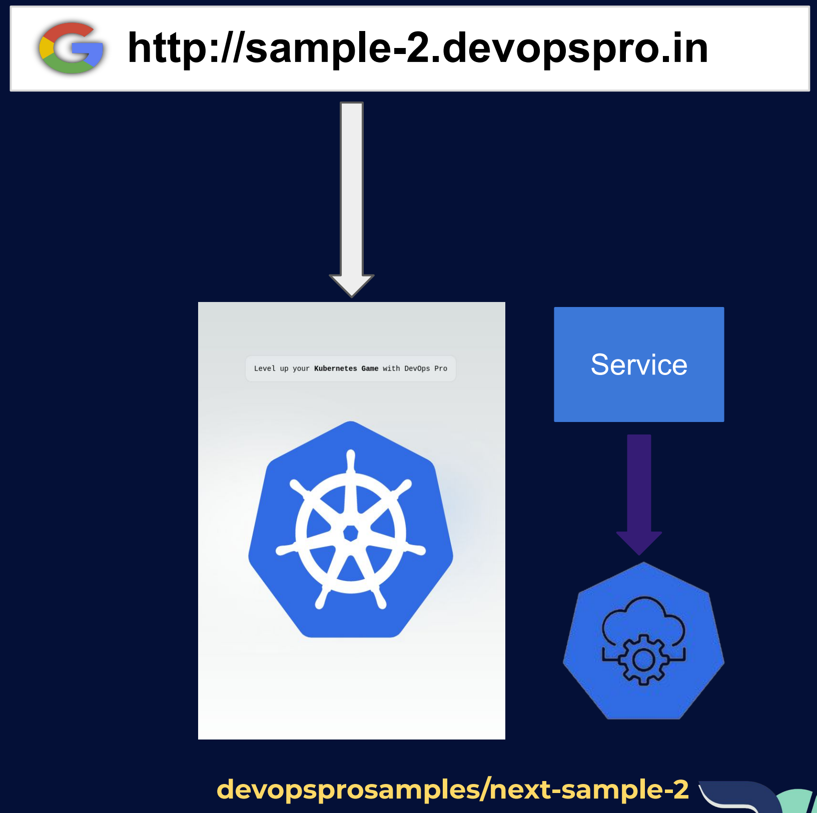


http://sample-1.devopspro.in

Service

Level up your **DevOps Game** with DevOps Pro

devopsprosamples/next-sample-1



http://sample-2.devopspro.in

Level up your **Kubernetes Game** with DevOps Pro

Service

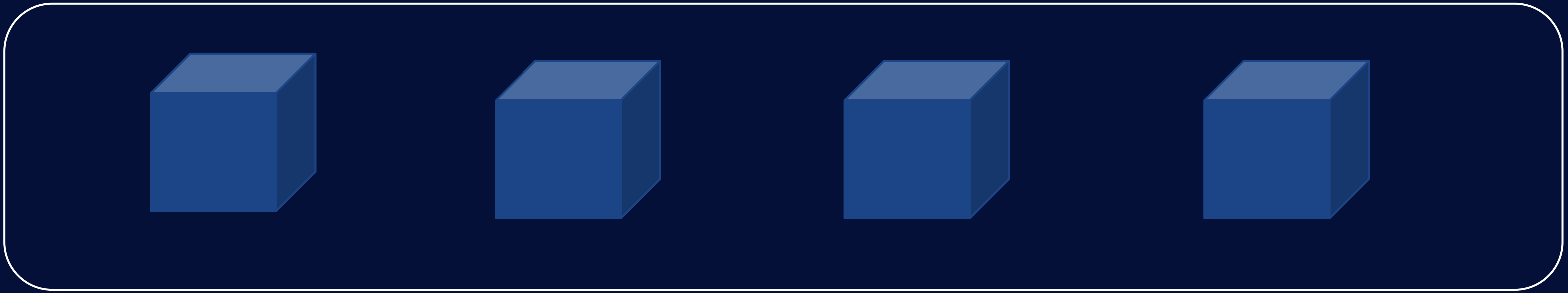
devopsprosamples/next-sample-2

4

Ingress Controller

Nginx Ingress Controller

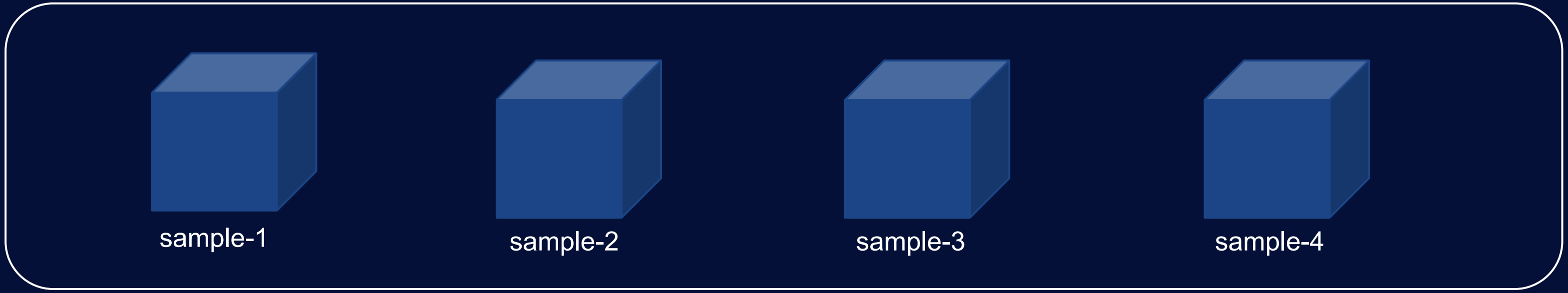
```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/kind/deploy.yaml
```



5

Deployment

- kubectl create deploy sample-1 --image=devopsprosamples/next-path-sample-1
- kubectl create deploy sample-2 --image=devopsprosamples/next-path-sample-2
- kubectl create deploy sample-3 --image=devopsprosamples/next-sample-1
- kubectl create deploy sample-4 --image=devopsprosamples/next-sample-2



6

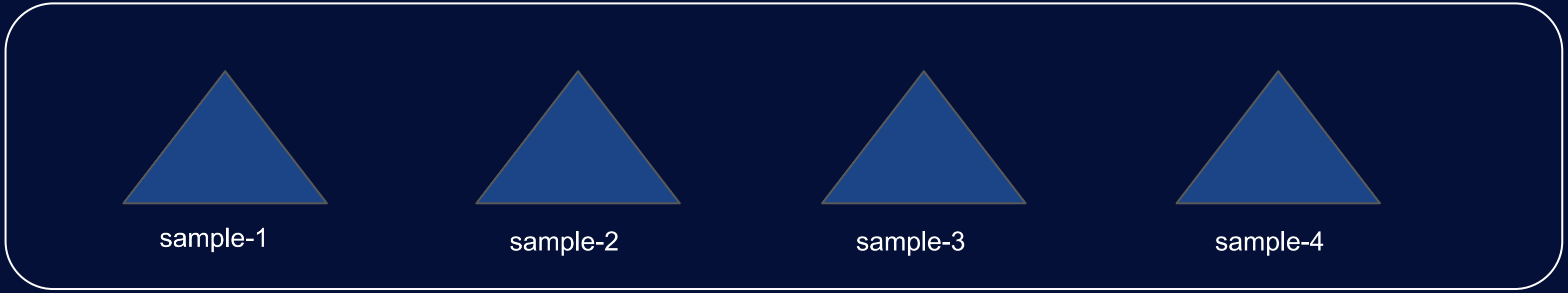
Service

```
kubectl expose deploy sample-1 --type=ClusterIP --port=3000
```

```
kubectl expose deploy sample-2 --type=ClusterIP --port=3000
```

```
kubectl expose deploy sample-3 --type=ClusterIP --port=3000
```

```
kubectl expose deploy sample-4 --type=ClusterIP --port=3000
```



Ingress Resource

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: example-ingress
spec:
 rules:
 - host: "example.devopspro.in"
   http:
     paths:
     - pathType: Prefix
       path: /sample-1
       backend:
         service:
           name: sample-1
           port:
             number: 3000
```

```yaml
 - host: "sample-2.devopspro.in"
   http:
     paths:
     - pathType: Prefix
       path: "/"mple-2
       backend:
         service:
           name: sample-4
           port:
             number: 3000
```

8

SSL Termination

🔒 https://example.devopspro.in

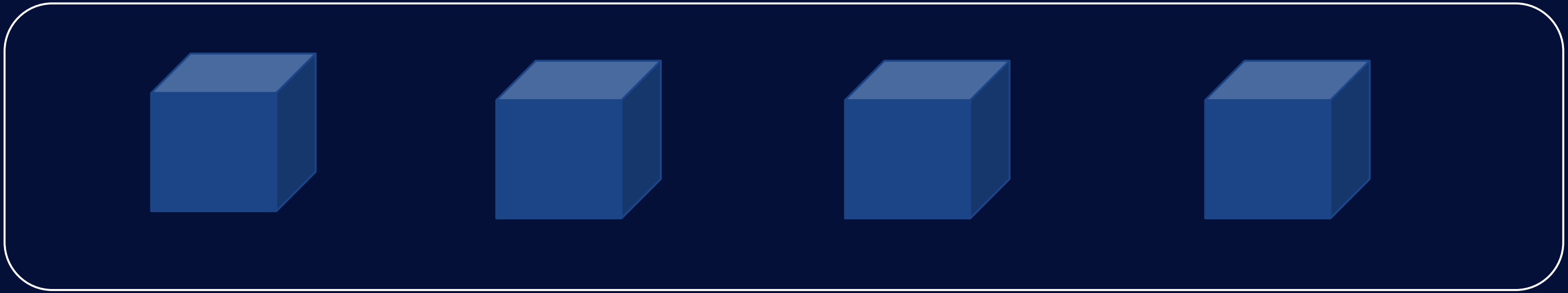
🔒 https://sample-2.devopspro.in

🔒 https://sample-1.devopspro.in

Step 1:-

Installing Cert Manager

```
kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.7.1/cert-manager.yaml
```



Step 2:-

Creating Issuer

Issuers, and ClusterIssuers, are Kubernetes resources that represent certificate authorities (CAs) that are able to generate signed certificates

staging_issuer.yaml

```yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-staging
  namespace: cert-manager
spec:
  acme:
    server: https://acme-staging-v02.api.letsencrypt.org/directory
    email: user@gmail.com
    privateKeySecretRef:
      name: letsencrypt-staging
    solvers:
    - http01:
        ingress:
          class:  nginx
```

Step 3:-

Updating Ingress Resource

ingress-resource.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: example-ingress
 annotations:
   cert-manager.io/cluster-issuer: "letsencrypt-staging"
   kubernetes.io/ingress.class: "nginx"
spec:
 tls:
 - hosts:
     - <your-host>
   secretName: tls-secret
 rules:
 .
 .
 .
```

Step 4:-

Creating Production Issuer

prod_issuer.yaml

```yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
  namespace: cert-manager
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: user@gmail.com
    privateKeySecretRef:
      name: letsencrypt-prod
    solvers:
    - http01:
        ingress:
          class:  nginx
```

Step 5:-

Updating Ingress Resource

ingress-resource.yaml

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: example-ingress
 annotations:
   cert-manager.io/cluster-issuer: "letsencrypt-prod"
   kubernetes.io/ingress.class: "nginx"
spec:
 tls:
 - hosts:
     - <your-host>
   secretName: tls-secret
 rules:
 .
 .
 .
```