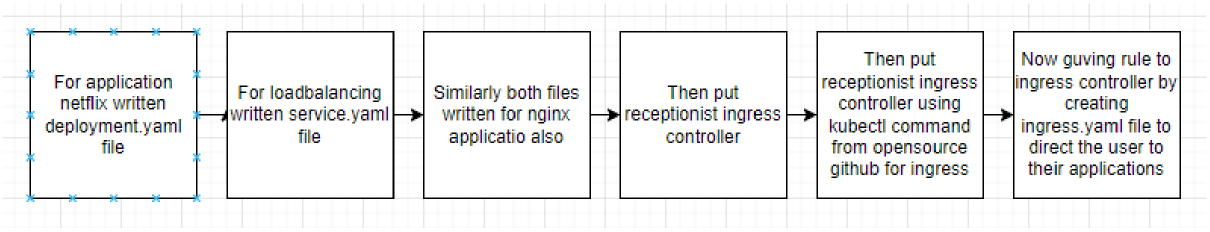
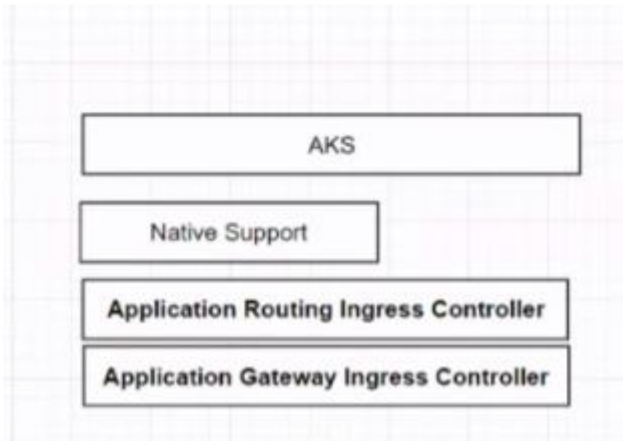
AGENDA – INGRESS

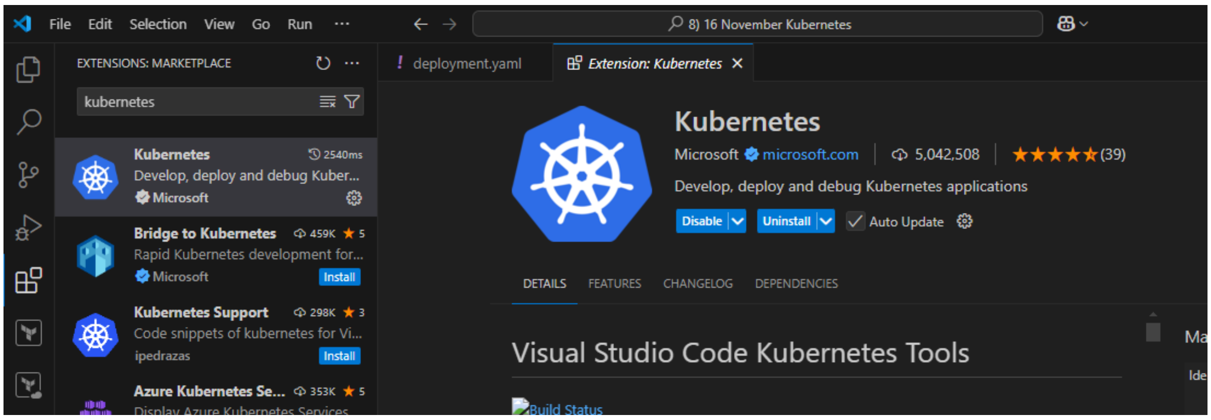

For application netflix written deployment.yaml file
For loadbalancing written service.yaml file
Similarly both files written for nginx applicatio also
Then put receptionist ingress controller
Then put receptionist ingress controller using kubectl command from opensource github for ingress
Now guving rule to ingress controller by creating ingress.yaml file to direct the user to their applications

1) Ingress controller works as a receptionist which actually routes the nginx calls to nginx and nelflix calls to Netflix. Ingress is assigned public ip which actually expose to users to access all applications from one single ingress ip

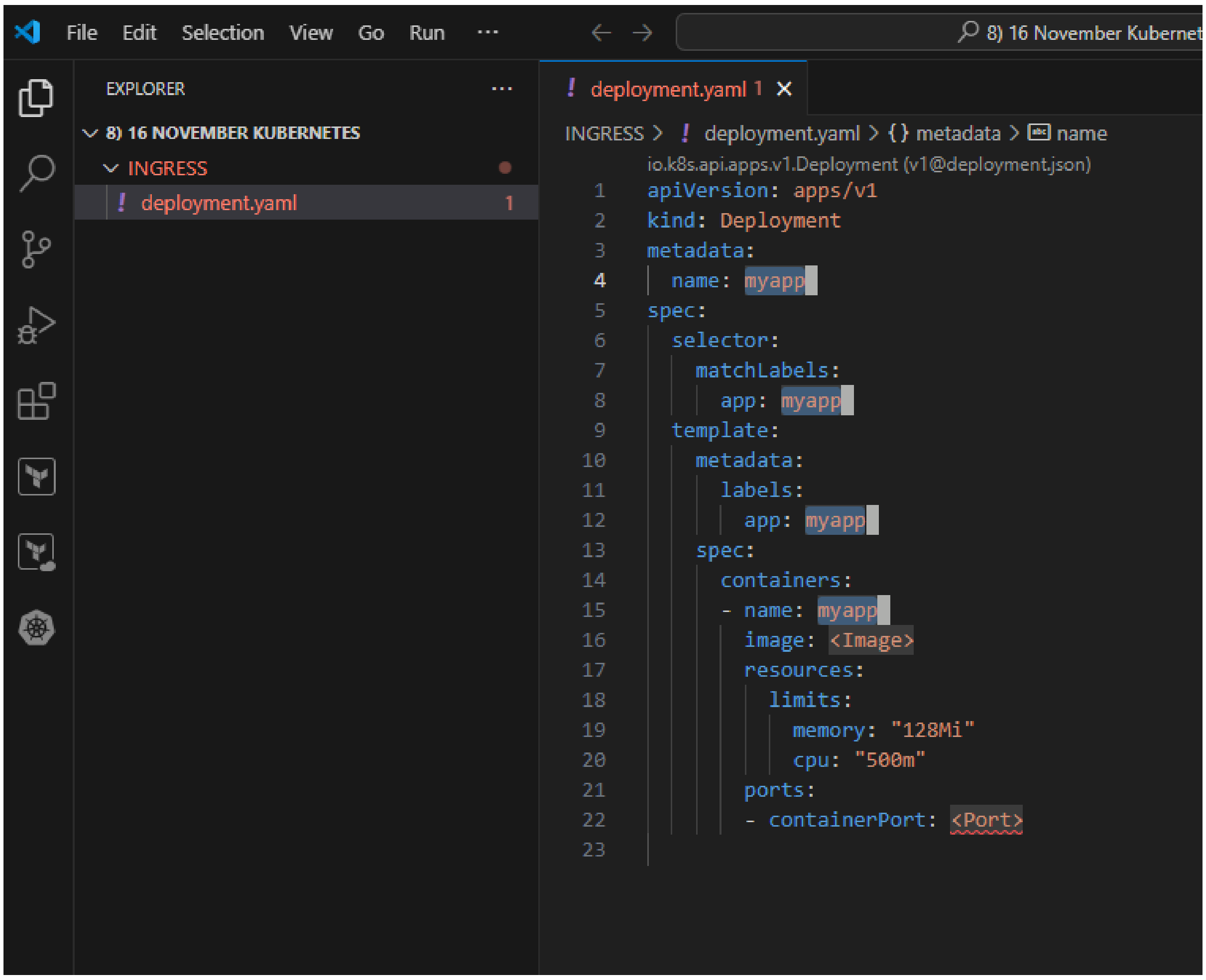

AKS
Native Support
Application Routing Ingress Controller
Application Gateway Ingress Controller

2) Create folder “8) 16 November Kubernetes” and create folder “ingress” and create file “netflix-deployment.yaml”.

3) Now enable k8s extension in vscode

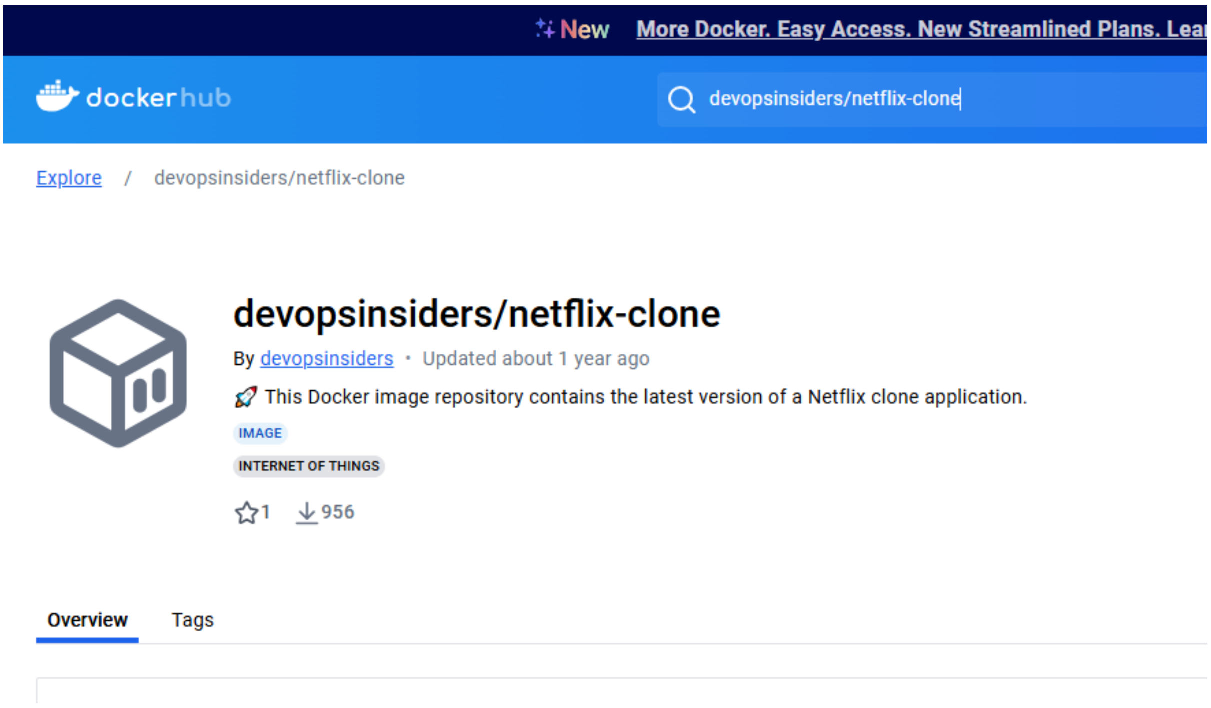

File Edit Selection View Go Run ...
8) 16 November Kubernetes
EXTENSIONS: MARKETPLACE
kubernetes
Kubernetes 2540ms
Develop, deploy and debug Kuber...
Microsoft
Bridge to Kubernetes 459K 5
Rapid Kubernetes development for...
Microsoft
Install
Kubernetes Support 298K 3
Code snippets of kubernetes for Vi...
ipedrazas
Install
Azure Kubernetes Se... 353K 5
deployment.yaml
Extension: Kubernetes
Kubernetes
Microsoft microsoft.com 5,042,508 (39)
Develop, deploy and debug Kubernetes applications
Disable
Uninstall
Auto Update
DETAILS
FEATURES
CHANGELOG
DEPENDENCIES
Visual Studio Code Kubernetes Tools
Build Status

4) write de and we will get deployment code


File Edit Selection View Go Run ···
8) 16 November Kubernet
EXPLORER
8) 16 NOVEMBER KUBERNETES
INGRESS
deployment.yaml 1
INGRESS > deployment.yaml > {} metadata > name
io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp
spec:
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      containers:
      - name: myapp
        image: <Image>
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: <Port>
```

5) Search below


New More Docker. Easy Access. New Streamlined Plans. Lea
docker hub
devopsinsiders/netflix-clone
Explore / devopsinsiders/netflix-clone

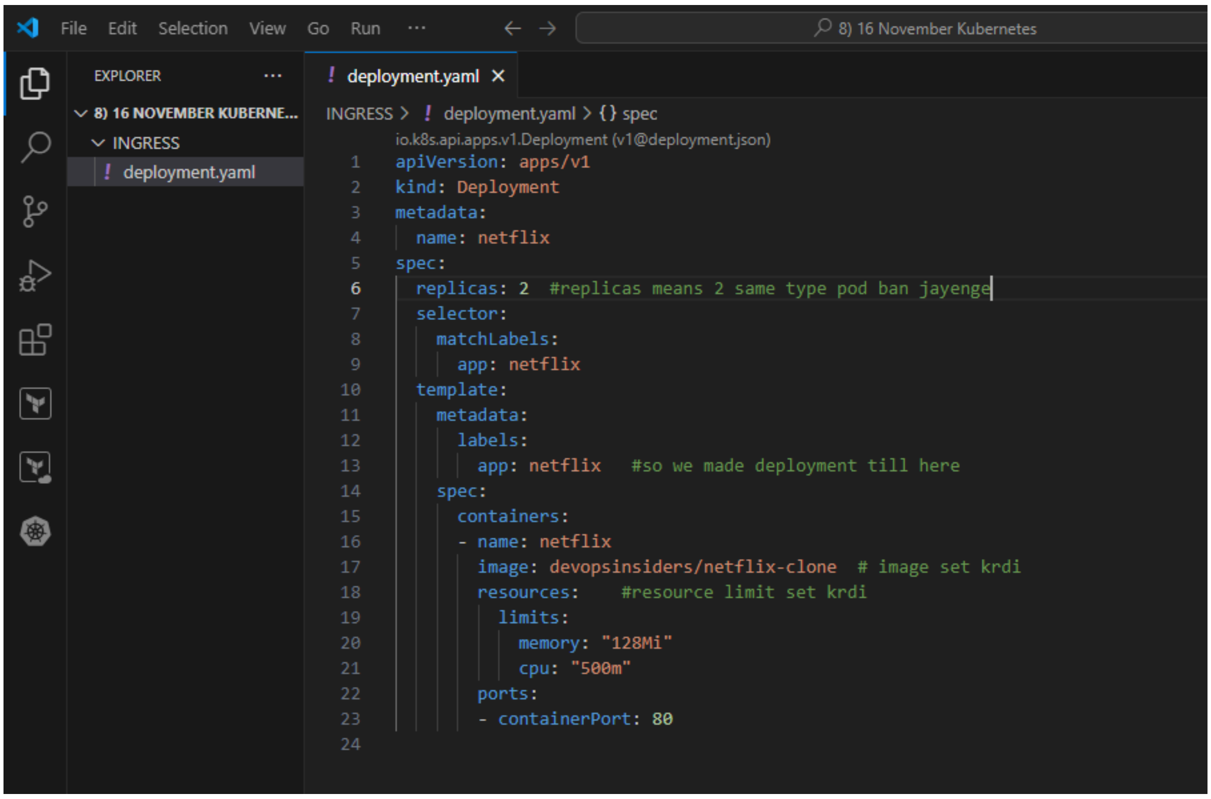
devopsinsiders/netflix-clone

By devopsinsiders • Updated about 1 year ago

🚀 This Docker image repository contains the latest version of a Netflix clone application.

IMAGE
INTERNET OF THINGS
☆1 ↓956
Overview
Tags

6) so deployment.yaml file is as follows

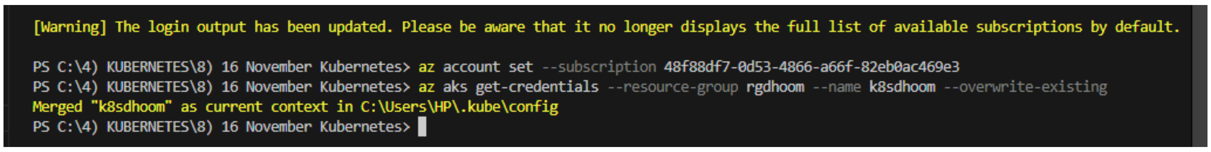

File Edit Selection View Go Run ···
8) 16 November Kubernetes
EXPLORER
8) 16 NOVEMBER KUBERNE...
INGRESS
deployment.yaml
INGRESS > deployment.yaml > {} spec
io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: netflix
spec:
  replicas: 2  #replicas means 2 same type pod ban jayenge
  selector:
    matchLabels:
      app: netflix
  template:
    metadata:
      labels:
        app: netflix   #so we made deployment till here
    spec:
      containers:
      - name: netflix
        image: devopsinsiders/netflix-clone  # image set krdi
        resources:    #resource limit set krdi
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 80
```

7) **az login**

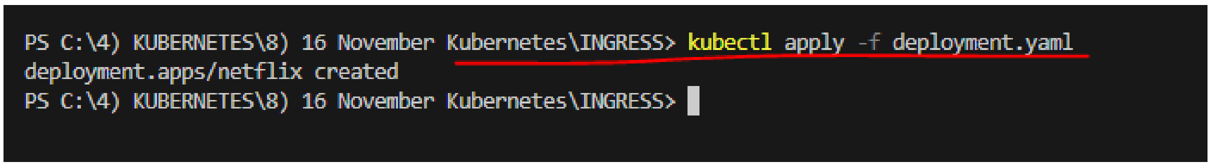
az account set --subscription 48f88df7-0d53-4866-a66f-82eb0ac469e3

az aks get-credentials --resource-group rgdhoom --name k8sdhoom --overwrite-existing



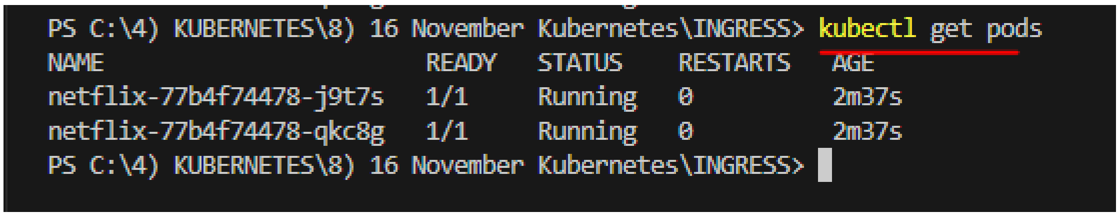
```
[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by default.

PS C:\4) KUBERNETES\8) 16 November Kubernetes> az account set --subscription 48f88df7-0d53-4866-a66f-82eb0ac469e3
PS C:\4) KUBERNETES\8) 16 November Kubernetes> az aks get-credentials --resource-group rgdhoom --name k8sdhoom --overwrite-existing
Merged "k8sdhoom" as current context in C:\Users\HP\.kube\config
PS C:\4) KUBERNETES\8) 16 November Kubernetes>
```

8) **kubectl apply -f deployment.yaml** = create deployment



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f deployment.yaml
deployment.apps/netflix created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

9) **kubectl get pods**

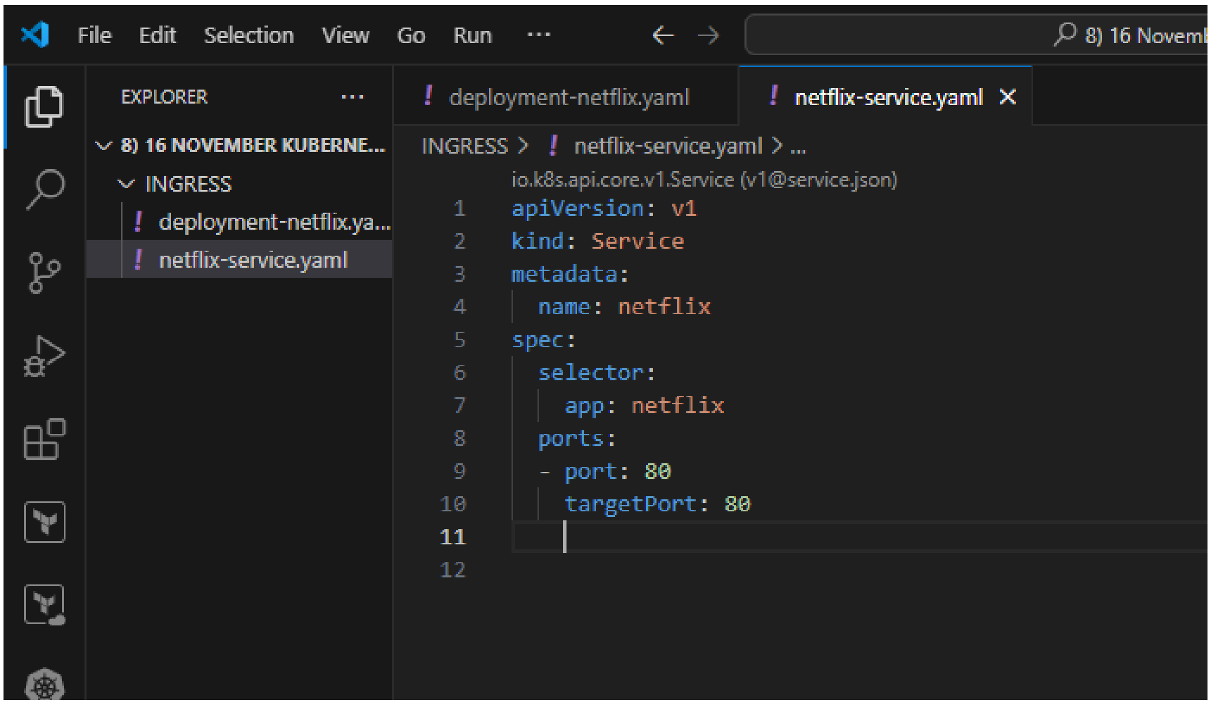


```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
netflix-77b4f74478-j9t7s   1/1     Running   0          2m37s
netflix-77b4f74478-qkc8g   1/1     Running   0          2m37s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

AGENDA – Making service or actually loadbalancer

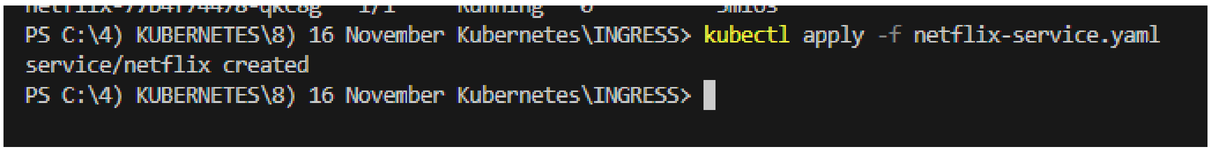
1) create “netflix-service.yaml”

2) Write ser so it will give service.yaml



```yaml
apiVersion: v1
kind: Service
metadata:
  name: netflix
spec:
  selector:
    app: netflix
  ports:
  - port: 80
    targetPort: 80
```

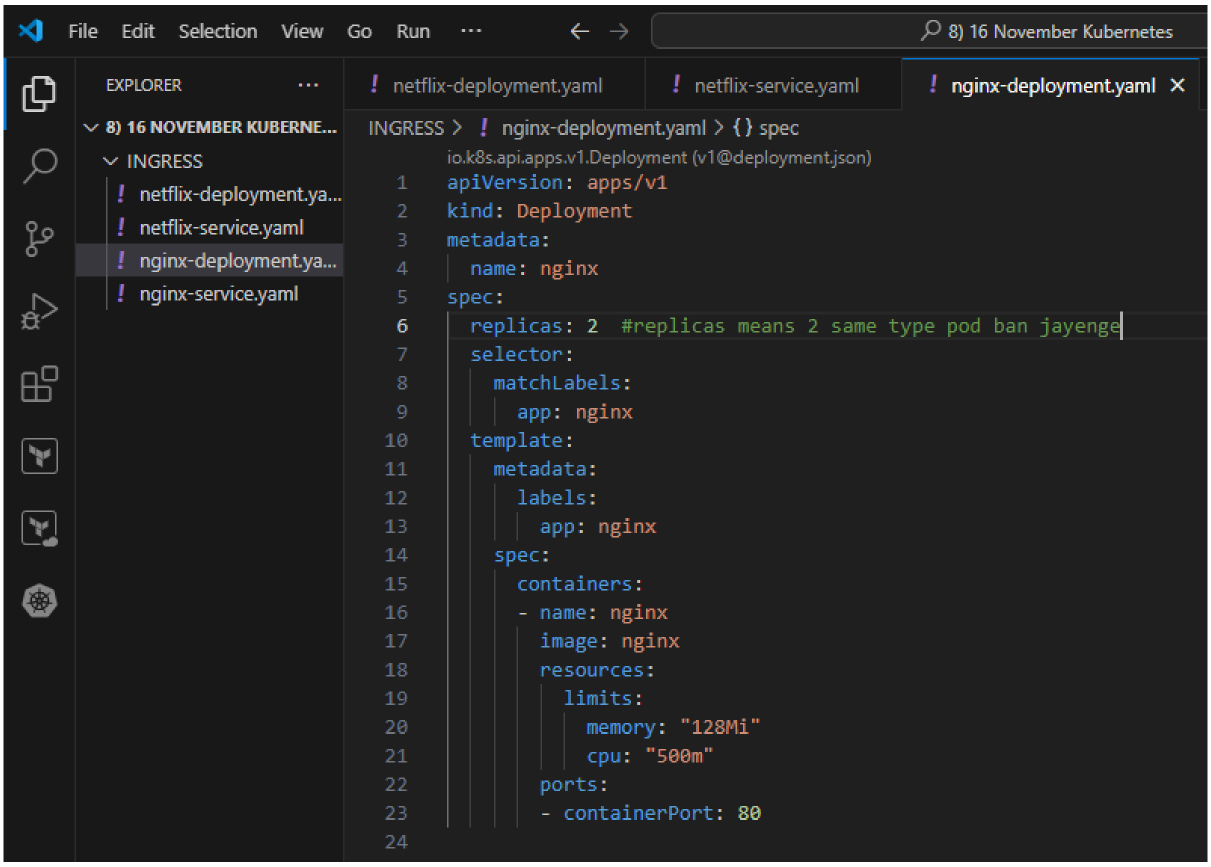
3) **kubectl apply -f netflix-service.yaml** = create service



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f netflix-service.yaml
service/netflix created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

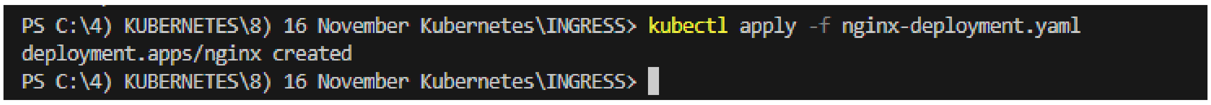
AGENDA – Create nginx deployment yaml file

1) Create nginx-deployment.yaml file


File Edit Selection View Go Run ···
8) 16 November Kubernetes
EXPLORER
8) 16 NOVEMBER KUBERNE...
INGRESS
netflix-deployment.ya...
netflix-service.yaml
nginx-deployment.ya...
nginx-service.yaml
netflix-deployment.yaml
netflix-service.yaml
nginx-deployment.yaml
INGRESS > nginx-deployment.yaml > { } spec
io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 2  #replicas means 2 same type pod ban jayenge
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 80
```

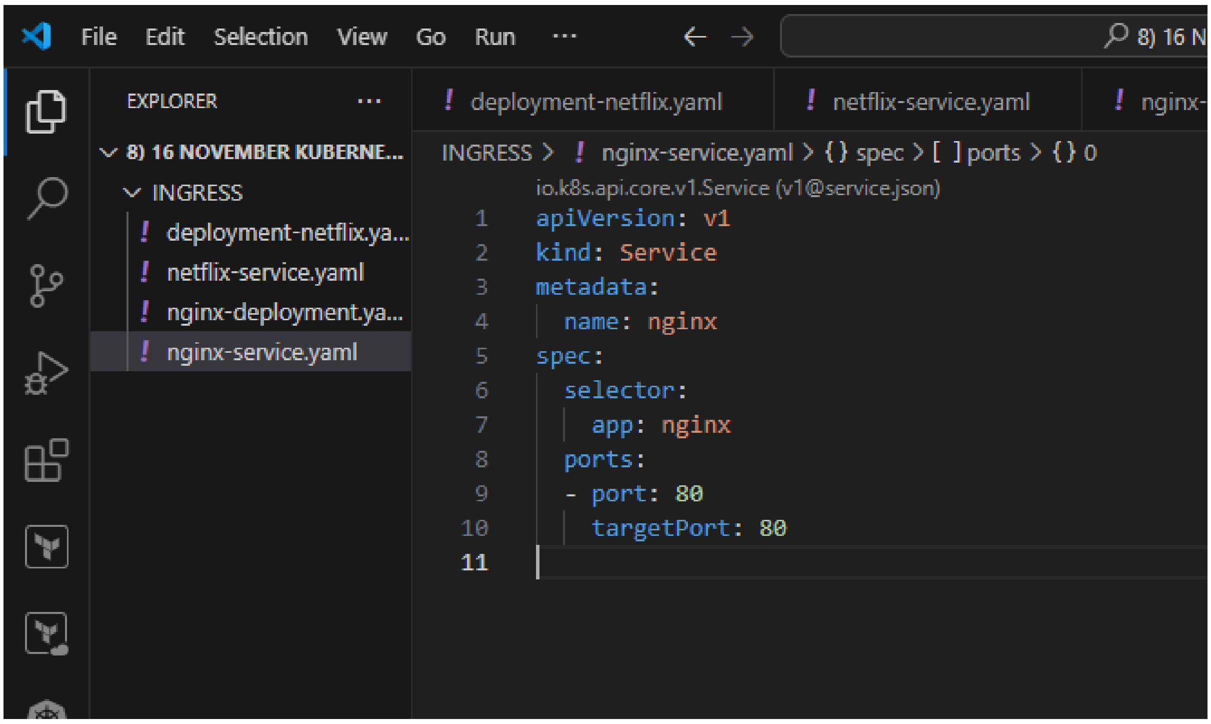
kubectl apply -f nginx-deployment.yaml



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

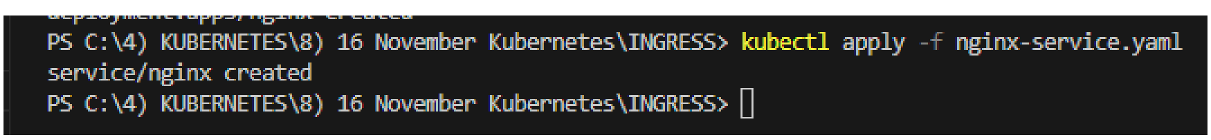
AGENDA – CREATE nginx-service

1) Create nginx-service.yaml file and write “ser”


File Edit Selection View Go Run ···
8) 16 N
EXPLORER
8) 16 NOVEMBER KUBERNE...
INGRESS
deployment-netflix.ya...
netflix-service.yaml
nginx-deployment.ya...
nginx-service.yaml
deployment-netflix.yaml
netflix-service.yaml
nginx-
INGRESS > nginx-service.yaml > {} spec > [ ] ports > {} 0
io.k8s.api.core.v1.Service (v1@service.json)

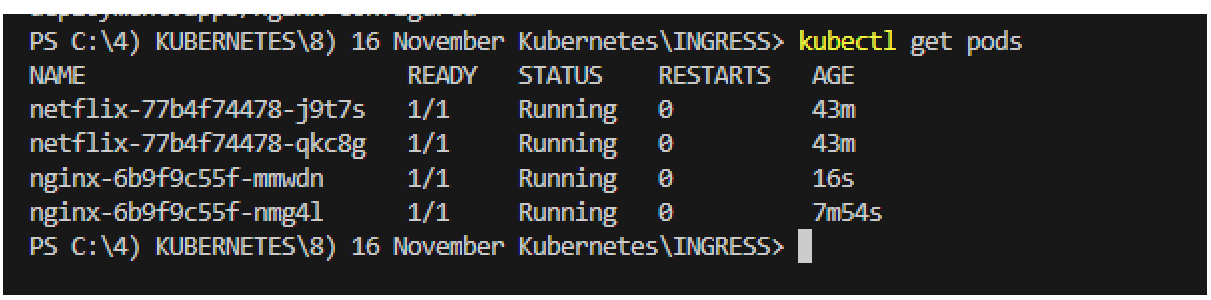
```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  selector:
    app: nginx
  ports:
  - port: 80
    targetPort: 80
```

kubectl apply -f nginx-service.yaml



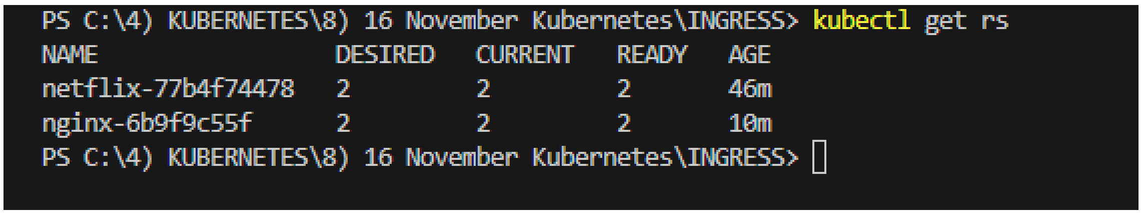
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f nginx-service.yaml
service/nginx created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```

2) **kubectl get pods**



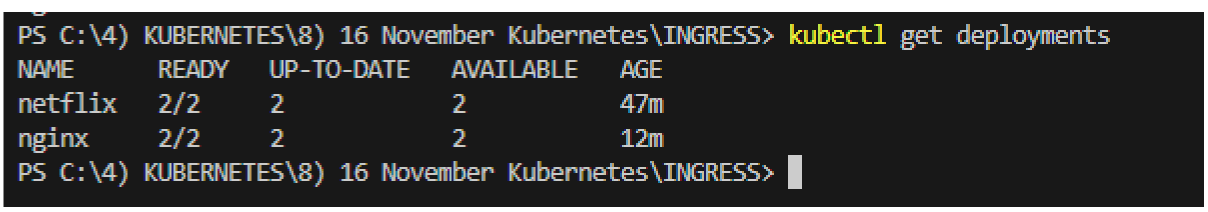
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
netflix-77b4f74478-j9t7s      1/1     Running   0          43m
netflix-77b4f74478-qkc8g      1/1     Running   0          43m
nginx-6b9f9c55f-mmwdn         1/1     Running   0          16s
nginx-6b9f9c55f-nmg4l         1/1     Running   0          7m54s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```

3) **kubectl get rs**



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get rs
NAME                 DESIRED   CURRENT   READY   AGE
netflix-77b4f74478   2         2         2       46m
nginx-6b9f9c55f      2         2         2       10m
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```

4) **kubectl get deployments**



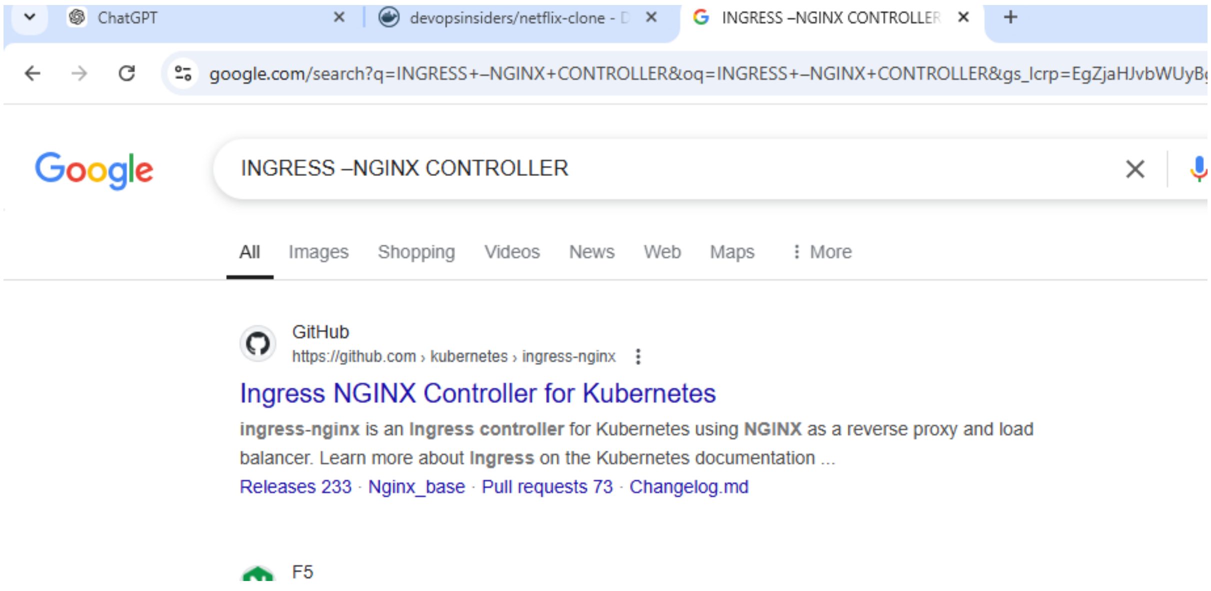
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
netflix   2/2     2            2           47m
nginx     2/2     2            2           12m
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```

++

AGENDA – Install ingress

1) SEARCH = INGRESS –NGINX CONTROLLER

https://github.com/kubernetes/ingress-nginx


ChatGPT
devopsinsiders/netflix-clone - D
INGRESS –NGINX CONTROLLER
google.com/search?q=INGRESS+–NGINX+CONTROLLER&oq=INGRESS+–NGINX+CONTROLLER&gs_lcrp=EgZjaHJvbWUyBg
INGRESS –NGINX CONTROLLER
All
Images
Shopping
Videos
News
Web
Maps
More

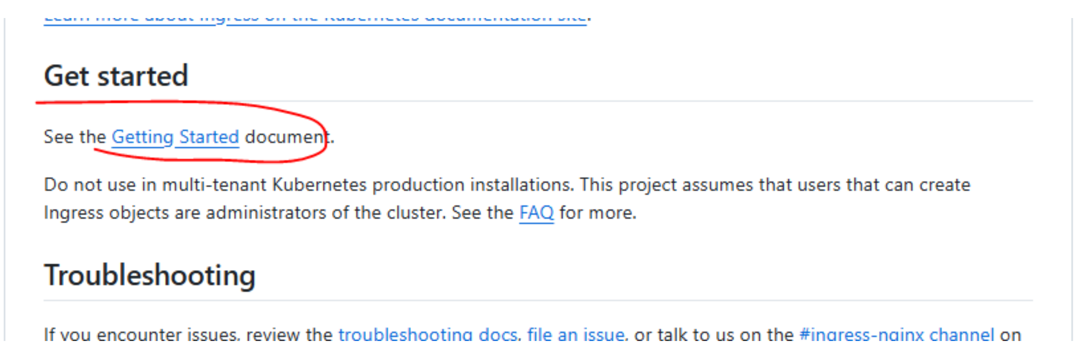
GitHub
https://github.com › kubernetes › ingress-nginx

Ingress NGINX Controller for Kubernetes

ingress-nginx is an **Ingress controller** for Kubernetes using **NGINX** as a reverse proxy and load balancer. Learn more about **Ingress** on the Kubernetes documentation ...

Releases 233 · Nginx_base · Pull requests 73 · Changelog.md

F5



Get started

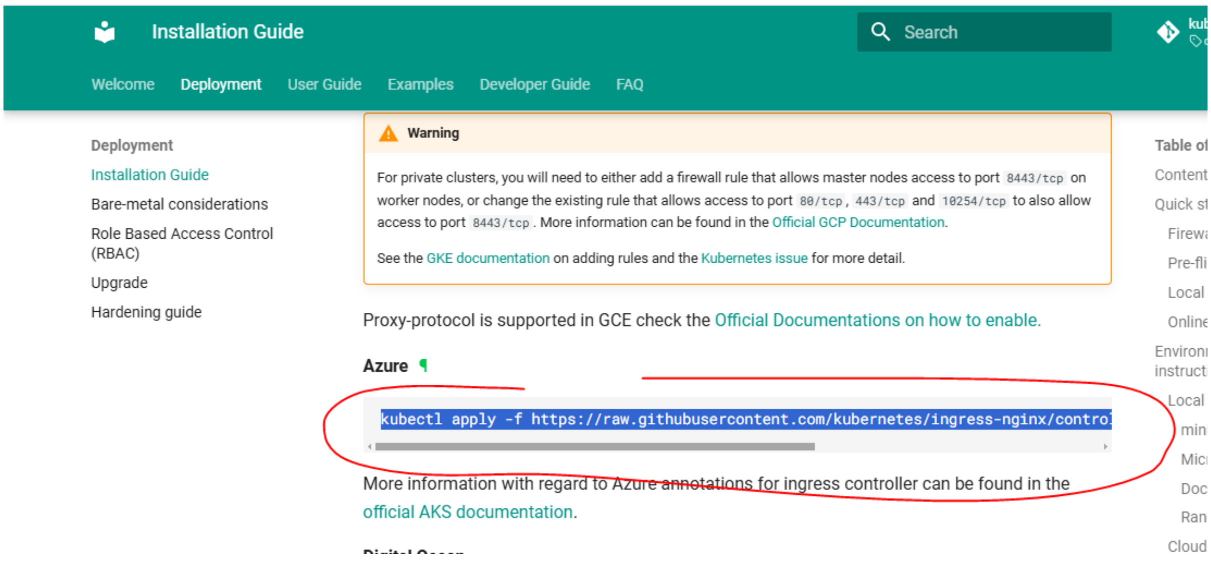
See the Getting Started document.

Do not use in multi-tenant Kubernetes production installations. This project assumes that users that can create Ingress objects are administrators of the cluster. See the FAQ for more.

Troubleshooting

If you encounter issues, review the troubleshooting docs, file an issue, or talk to us on the #ingress-nginx channel on

2) go to azure section



Warning

For private clusters, you will need to either add a firewall rule that allows master nodes access to port `8443/tcp` on worker nodes, or change the existing rule that allows access to port `80/tcp`, `443/tcp` and `10254/tcp` to also allow access to port `8443/tcp`. More information can be found in the Official GCP Documentation.

See the GKE documentation on adding rules and the Kubernetes issue for more detail.

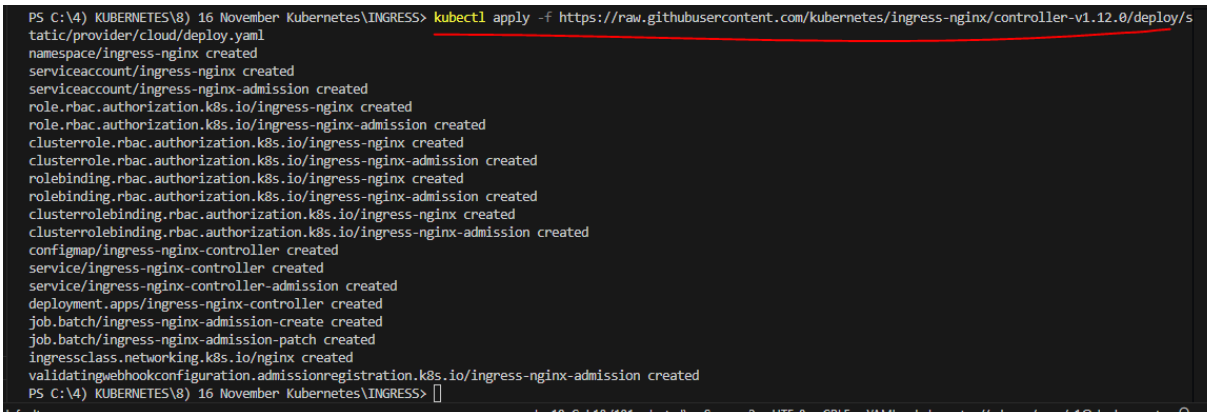
Proxy-protocol is supported in GCE check the Official Documentations on how to enable.

Azure

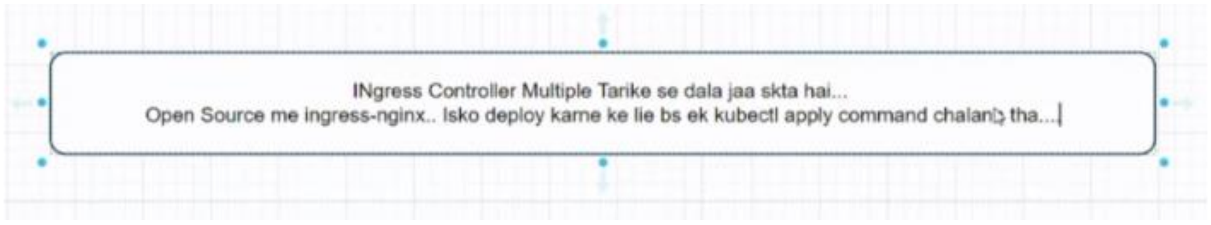
```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/contro
```

More information with regard to Azure annotations for ingress controller can be found in the official AKS documentation.

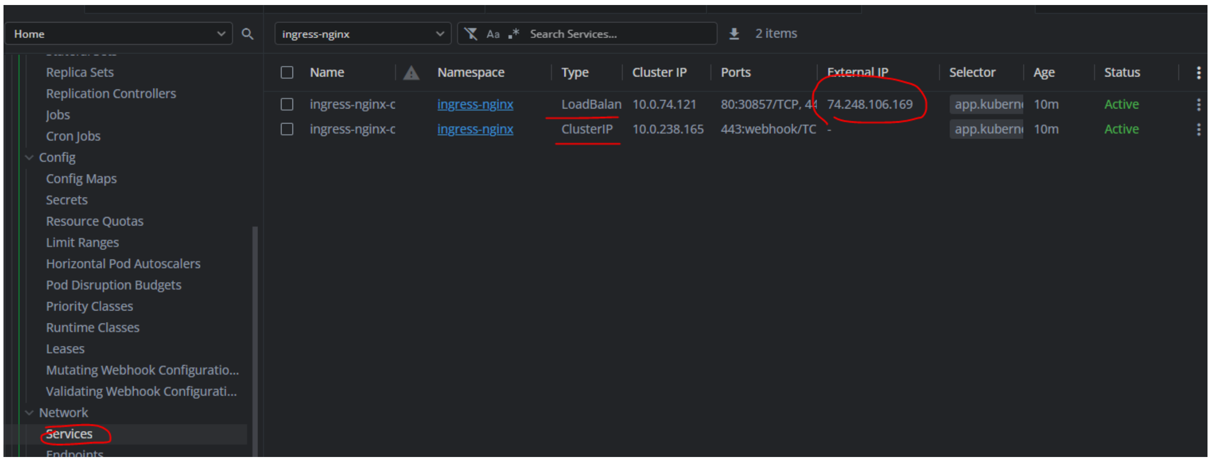
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.12.0/deploy/static/provider/cloud/deploy.yaml



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.12.0/deploy/s
tatic/provider/cloud/deploy.yaml
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
serviceaccount/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
configmap/ingress-nginx-controller created
service/ingress-nginx-controller created
service/ingress-nginx-controller-admission created
deployment.apps/ingress-nginx-controller created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ingressclass.networking.k8s.io/nginx created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```



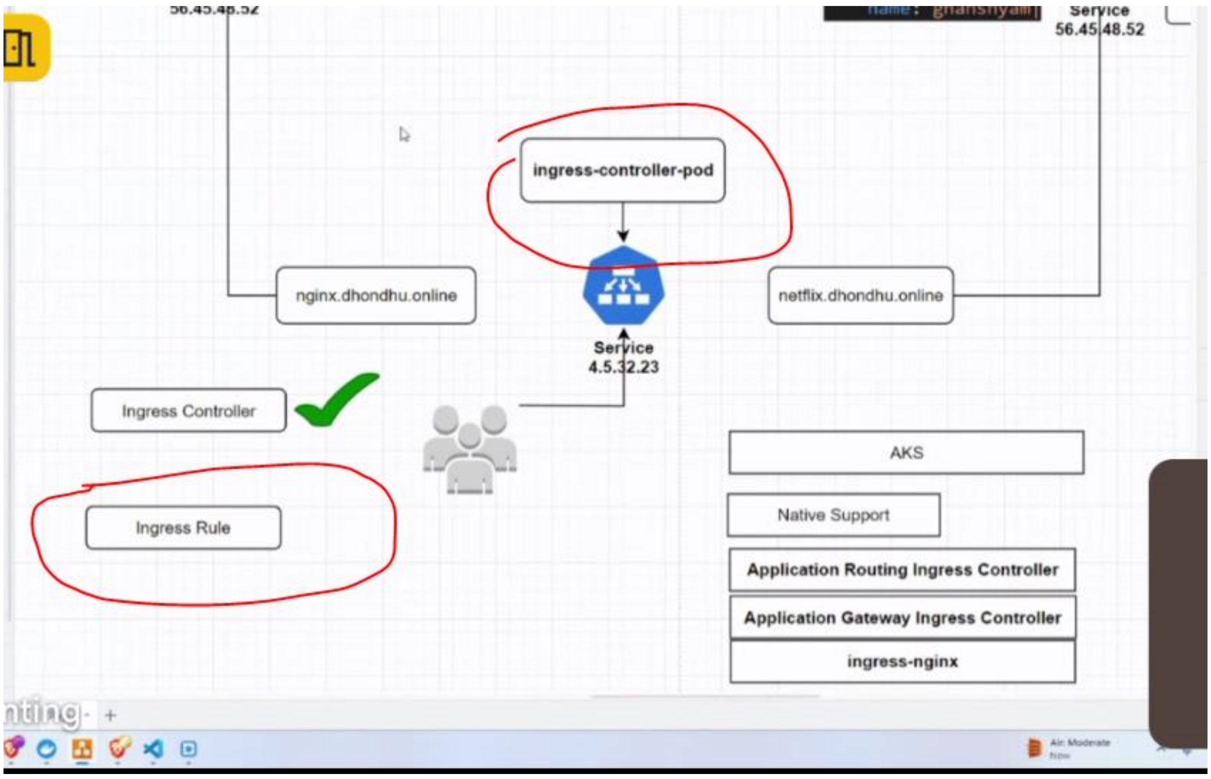
INgress Controller Multiple Tarike se dala jaa skta hai...
Open Source me ingress-nginx.. Isko deploy karne ke lie bs ek kubectl apply command chalana tha...



Name	Namespace	Type	Cluster IP	Ports	External IP	Selector	Age	Status
ingress-nginx-c	ingress-nginx	LoadBalan	10.0.74.121	80:30857/TCP, 44	74.248.106.169	app.kuberne	10m	Active
ingress-nginx-c	ingress-nginx	ClusterIP	10.0.238.165	443:webhook/TC	-	app.kuberne	10m	Active

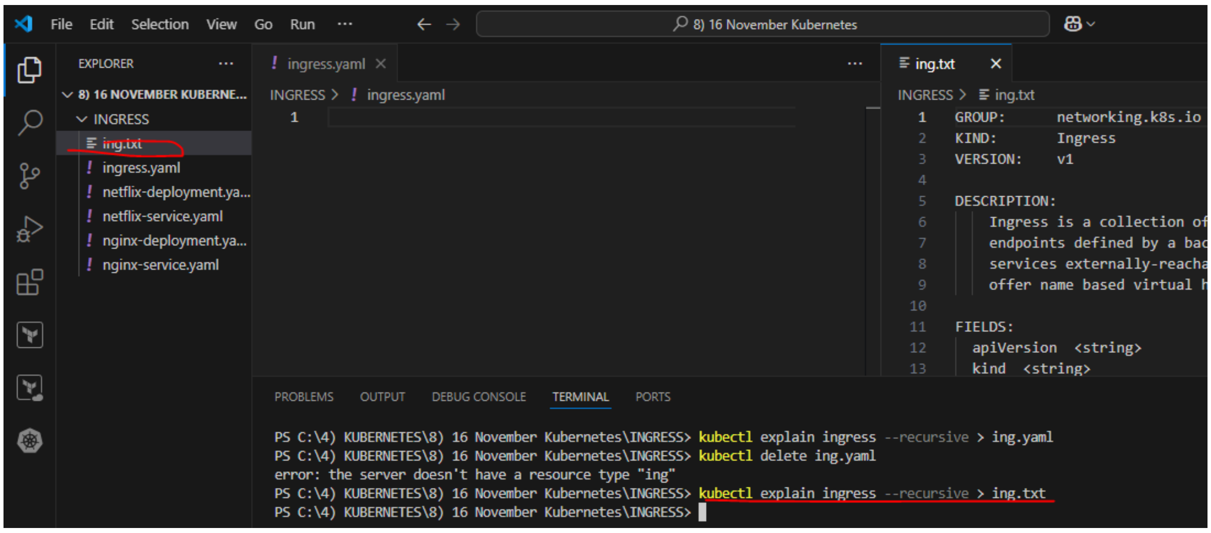
++

AGENDA – Create rules in ingress



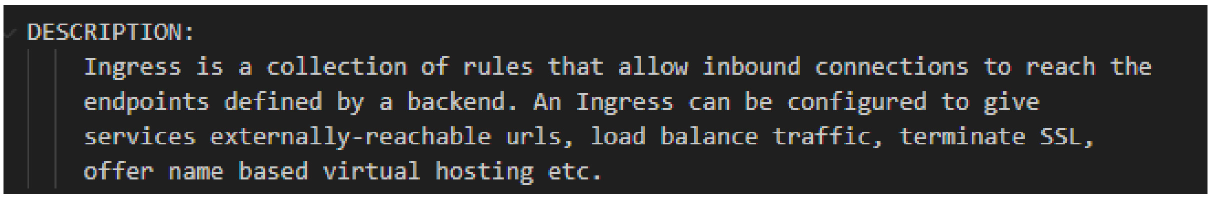
1) create Netflix-ingress.yaml

kubectl explain ingress --recursive > ing.txt



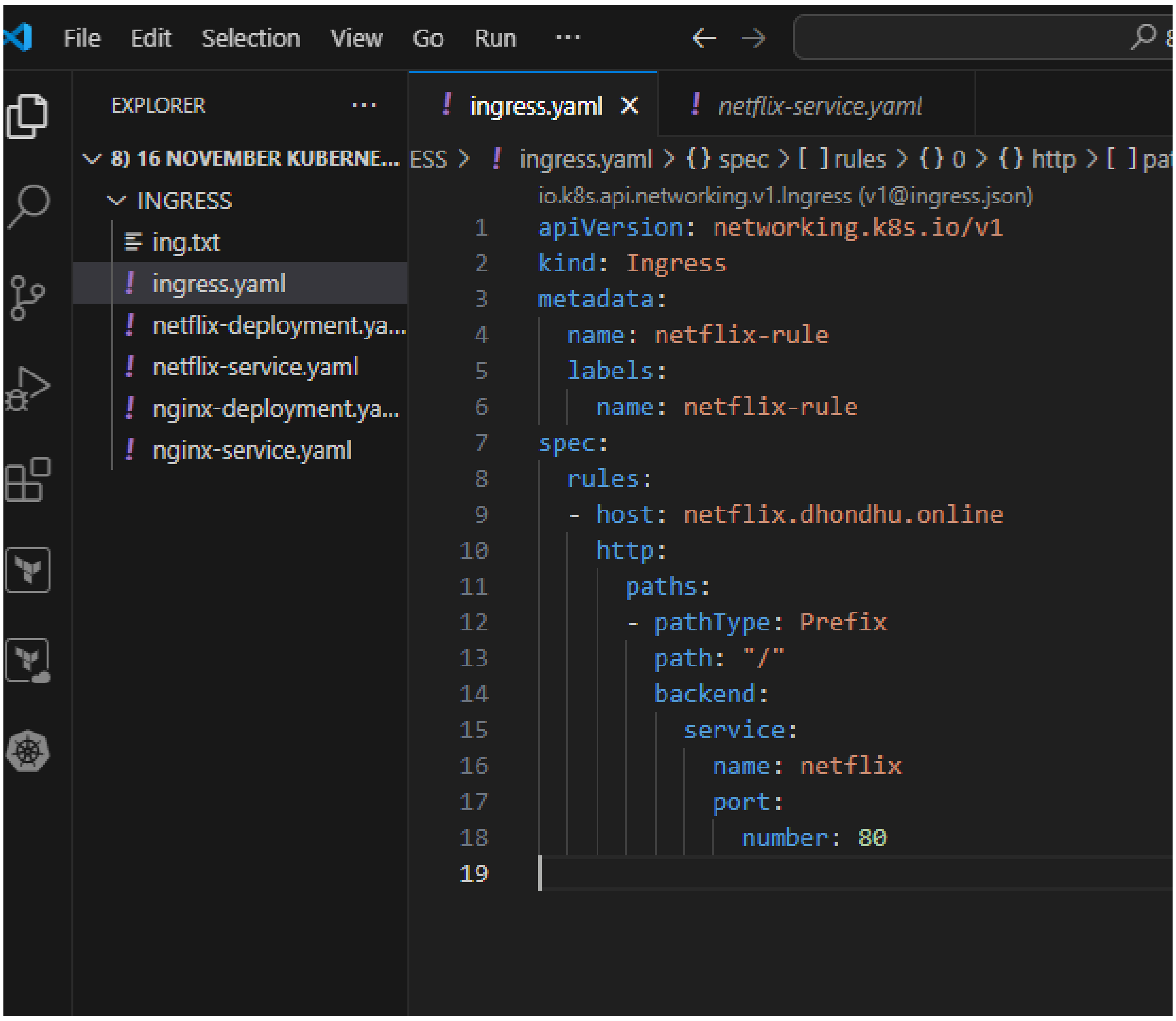
```
GROUP:      networking.k8s.io
KIND:       Ingress
VERSION:    v1

DESCRIPTION:
    Ingress is a collection of
    endpoints defined by a bac
    services externally-reacha
    offer name based virtual h

FIELDS:
  apiVersion  <string>
  kind  <string>
```

```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl explain ingress --recursive > ing.yaml
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl delete ing.yaml
error: the server doesn't have a resource type "ing"
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl explain ingress --recursive > ing.txt
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> 
```



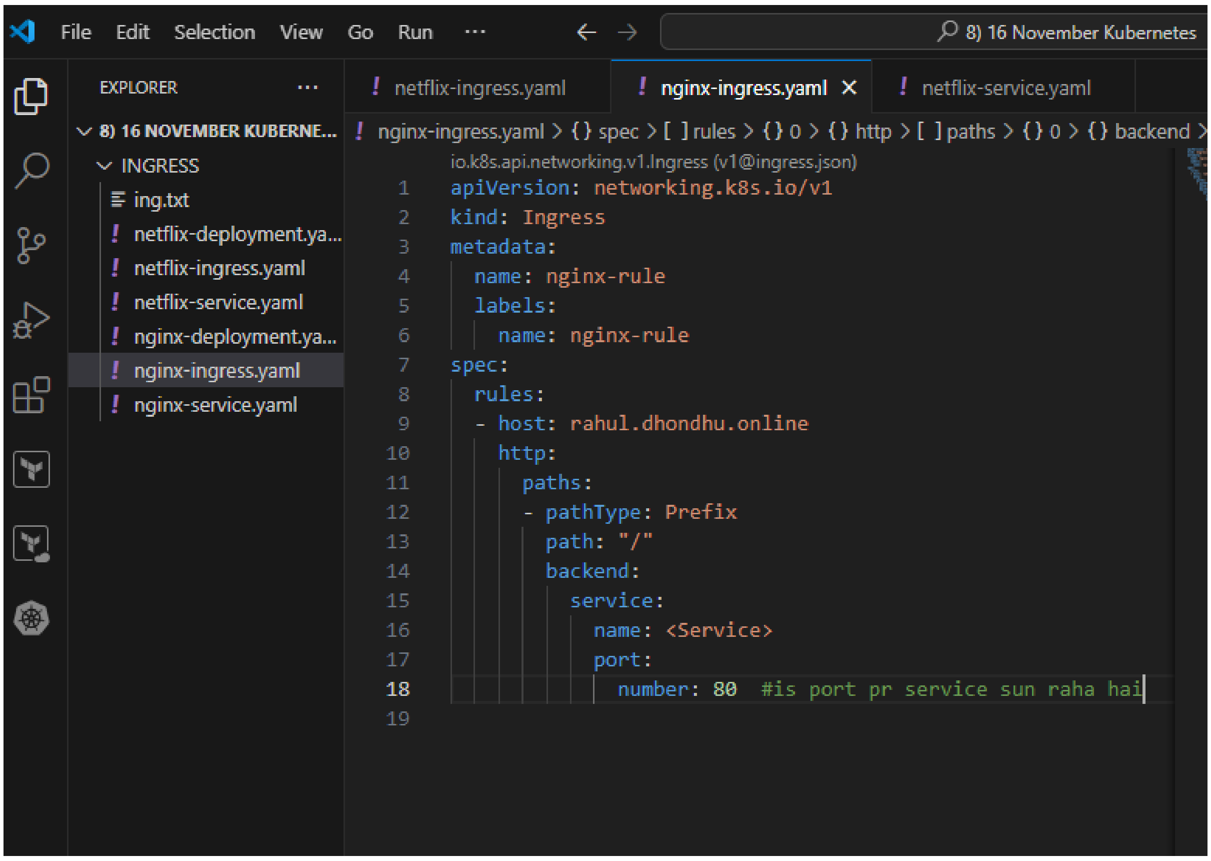
```
DESCRIPTION:
    Ingress is a collection of rules that allow inbound connections to reach the
    endpoints defined by a backend. An Ingress can be configured to give
    services externally-reachable urls, load balance traffic, terminate SSL,
    offer name based virtual hosting etc.
```

2) ingress.yaml rule file is below



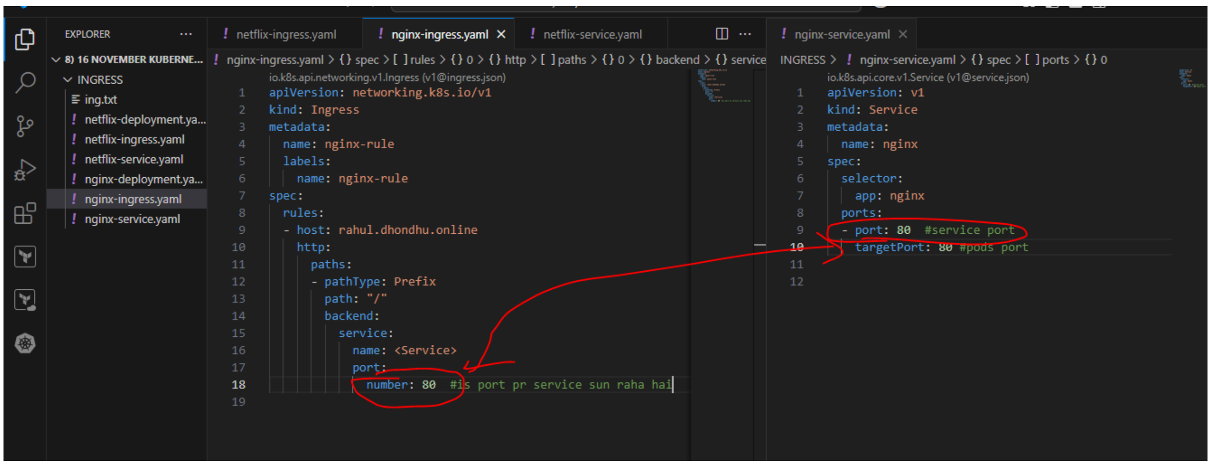
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: netflix-rule
  labels:
    name: netflix-rule
spec:
  rules:
  - host: netflix.dhondhu.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: netflix
            port:
              number: 80

```

3) Similarly create “nginx-ingress.yaml” file



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: nginx-rule
  labels:
    name: nginx-rule
spec:
  rules:
  - host: rahul.dhondhu.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: <Service>
            port:
              number: 80  #is port pr service sun raha hai
```

4) And below ports are same



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: nginx-rule
  labels:
    name: nginx-rule
spec:
  rules:
  - host: rahul.dhondhu.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: <Service>
            port:
              number: 80  #is port pr service sun raha hai
```

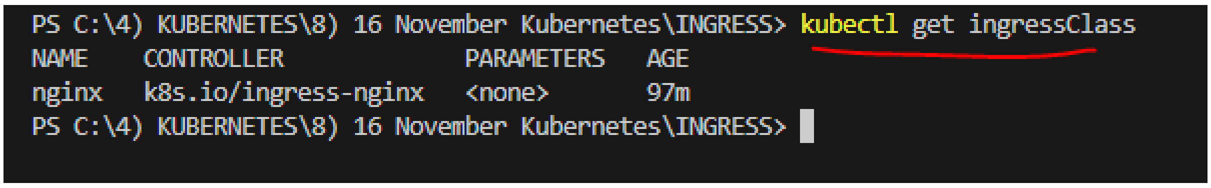
```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  selector:
    app: nginx
  ports:
  - port: 80  #service port
    targetPort: 80 #pods port
```

++

AGENDA – ingressClassName

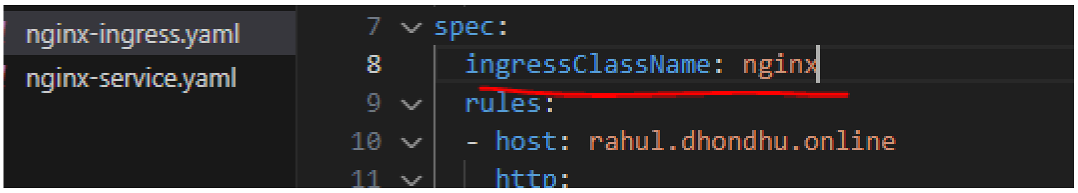
यदि आपके क्लस्टर में एक से अधिक Ingress कंट्रोलर्स हैं (जैसे NGINX Ingress Controller, Traefik, या कोई और कस्टम कंट्रोलर), तो आप Ingress Class Name का उपयोग करके यह specify कर सकते हैं कि कौन सा कंट्रोलर इस विशेष Ingress रिसोर्स को प्रोसेस करेगा

1) **kubectl get ingressClass**



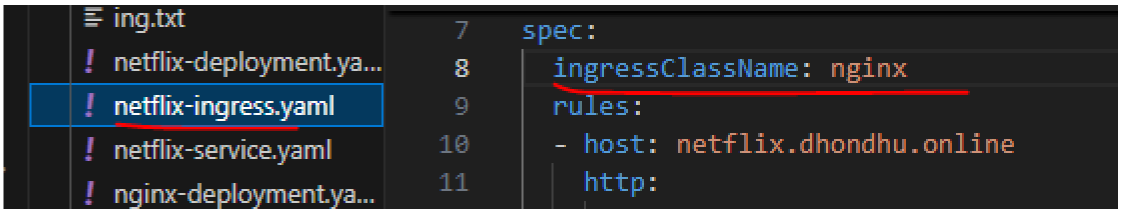
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get ingressClass
NAME    CONTROLLER             PARAMETERS   AGE
nginx   k8s.io/ingress-nginx   <none>       97m
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

2) Go to **nginx-ingress.yaml** and mention "ingressClassName: nginx"



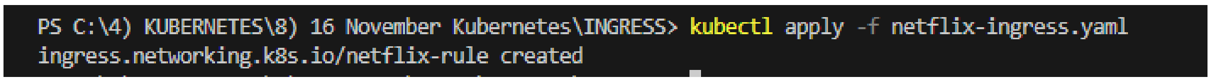
```
spec:
  ingressClassName: nginx
  rules:
  - host: rahul.dhondhu.online
    http:
```

3) Similarly go to **netflix-ingress.yaml** and mention "ingressClassName: nginx"



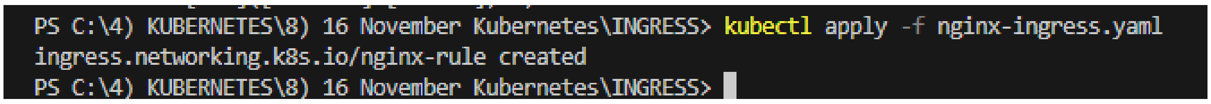
```
spec:
  ingressClassName: nginx
  rules:
  - host: netflix.dhondhu.online
    http:
```

4) **kubectl apply -f netflix-ingress.yaml**



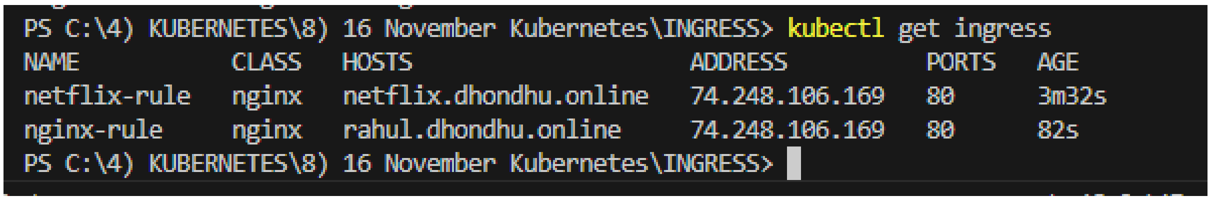
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f netflix-ingress.yaml
ingress.networking.k8s.io/netflix-rule created
```

5) **kubectl apply -f nginx-ingress.yaml**



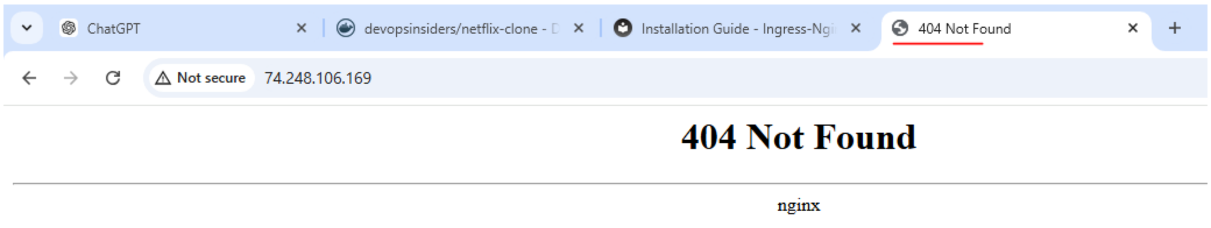
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f nginx-ingress.yaml
ingress.networking.k8s.io/nginx-rule created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

6) **kubectl get ingress** = ingress rules we got



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get ingress
NAME           CLASS   HOSTS                    ADDRESS          PORTS   AGE
netflix-rule   nginx   netflix.dhondhu.online   74.248.106.169   80      3m32s
nginx-rule     nginx   rahul.dhondhu.online     74.248.106.169   80      82s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

7) let run ip = 74.248.106.169 which will not run



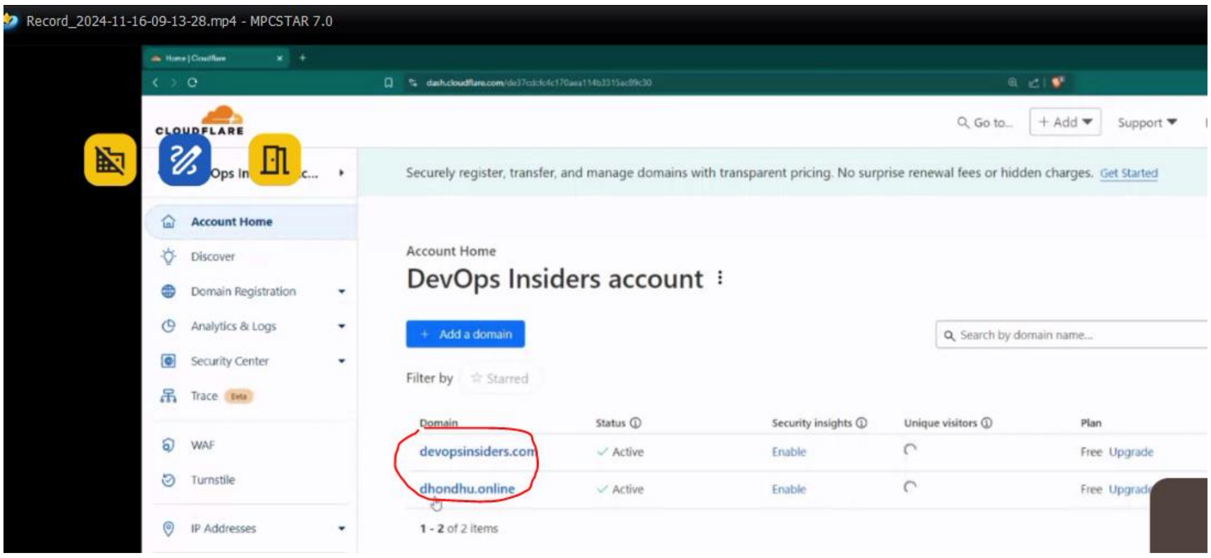
ChatGPT
devopsinsiders/netflix-clone - D
Installation Guide - Ingress-Ng
404 Not Found
Not secure
74.248.106.169

404 Not Found

nginx

8) Now go to cloudflare = 2.16

9) go to do domain = dhondhu.online -> DNS -> Add record -> give details -> save


Record_2024-11-16-09-13-28.mp4 - MPCSTAR 7.0

Account Home

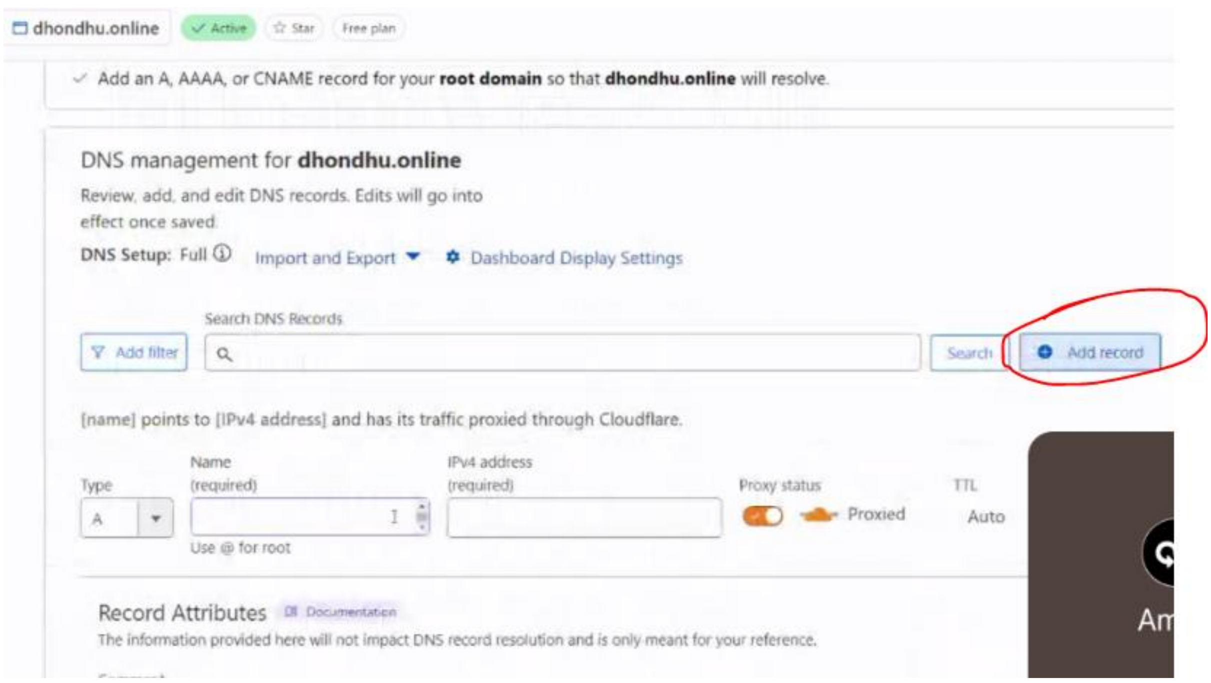
DevOps Insiders account

Securely register, transfer, and manage domains with transparent pricing. No surprise renewal fees or hidden charges. Get Started

Filter by ☆ Starred

Domain	Status	Security insights	Unique visitors	Plan
devopsinsiders.com	✓ Active	Enable		Free Upgrade
dhondhu.online	✓ Active	Enable		Free Upgrad

1 - 2 of 2 items


dhondhu.online ✓ Active ☆ Star Free plan

✓ Add an A, AAAA, or CNAME record for your **root domain** so that **dhondhu.online** will resolve.

DNS management for **dhondhu.online**

Review, add, and edit DNS records. Edits will go into effect once saved.

DNS Setup: Full ⓘ Import and Export ▾ Dashboard Display Settings

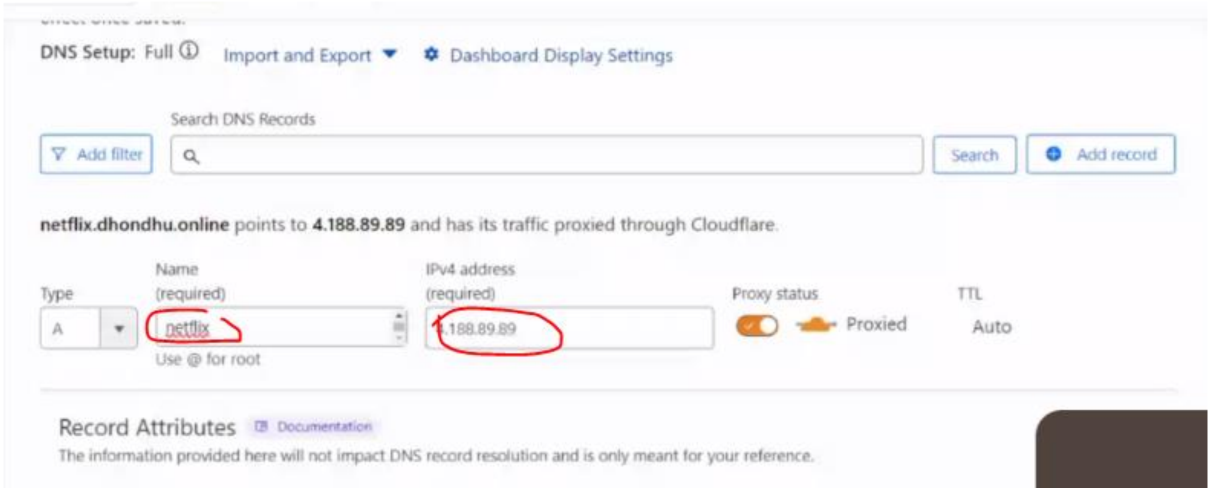
[name] points to [IPv4 address] and has its traffic proxied through Cloudflare.

Type	Name (required)	IPv4 address (required)	Proxy status	TTL
A			Proxied	Auto

Use @ for root

Record Attributes Documentation

The information provided here will not impact DNS record resolution and is only meant for your reference.


DNS Setup: Full ⓘ Import and Export ▾ Dashboard Display Settings

netflix.dhondhu.online points to **4.188.89.89** and has its traffic proxied through Cloudflare.

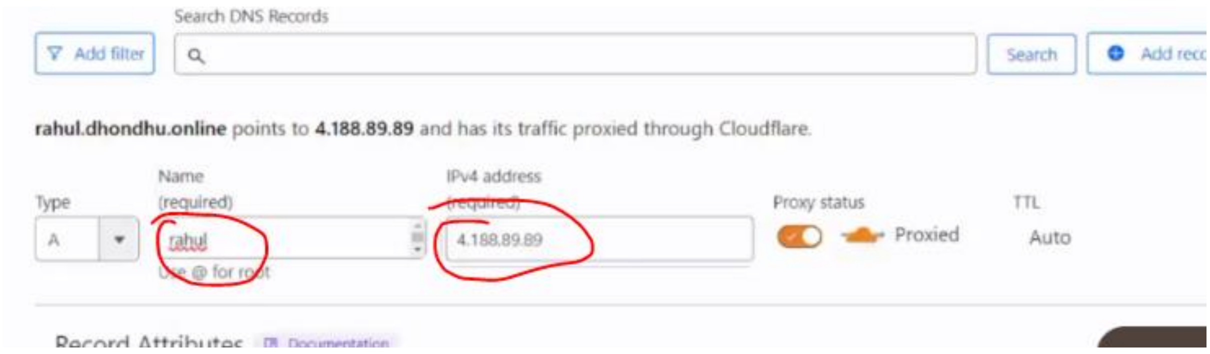
Type	Name (required)	IPv4 address (required)	Proxy status	TTL
A	netflix	4.188.89.89	Proxied	Auto

Use @ for root

Record Attributes Documentation

The information provided here will not impact DNS record resolution and is only meant for your reference.

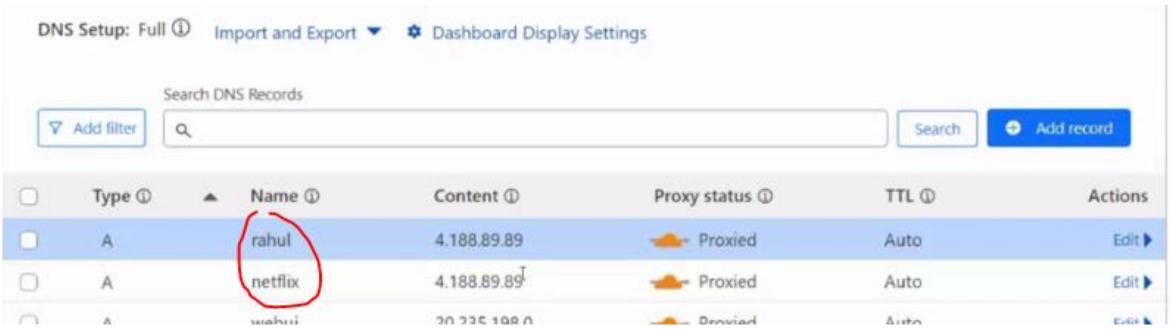
10) similarly adding another domain then save



Search DNS Records

rahul.dhondhu.online points to 4.188.89.89 and has its traffic proxied through Cloudflare.

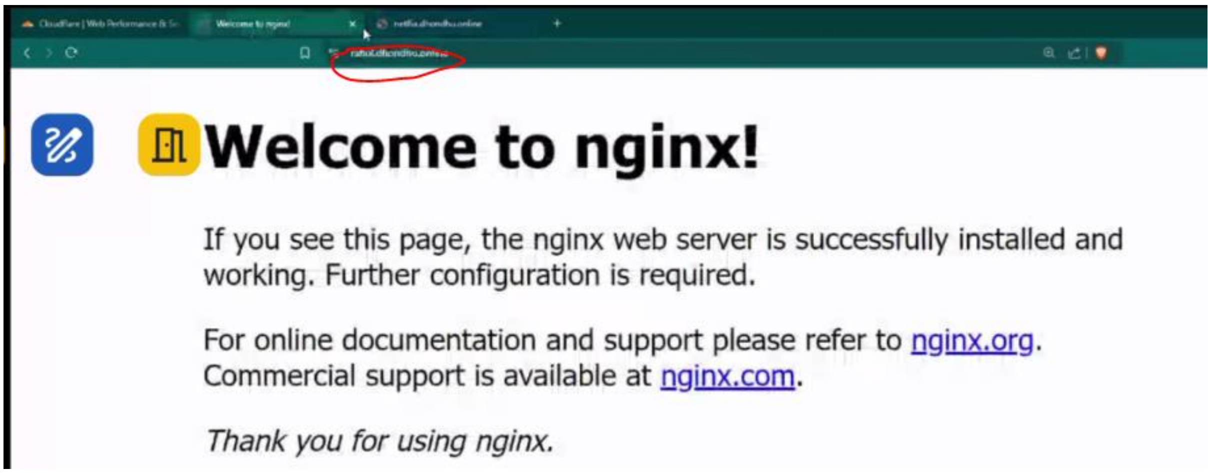
Type	Name (required)	IPv4 address (required)	Proxy status	TTL
A	rahul	4.188.89.89	Proxied	Auto



DNS Setup: Full

Type	Name	Content	Proxy status	TTL	Actions
A	rahul	4.188.89.89	Proxied	Auto	Edit
A	netflix	4.188.89.89	Proxied	Auto	Edit
A	webui	20.235.198.0	Proxied	Auto	Edit

11) **rahul.dhondhu.online**



rahul.dhondhu.online

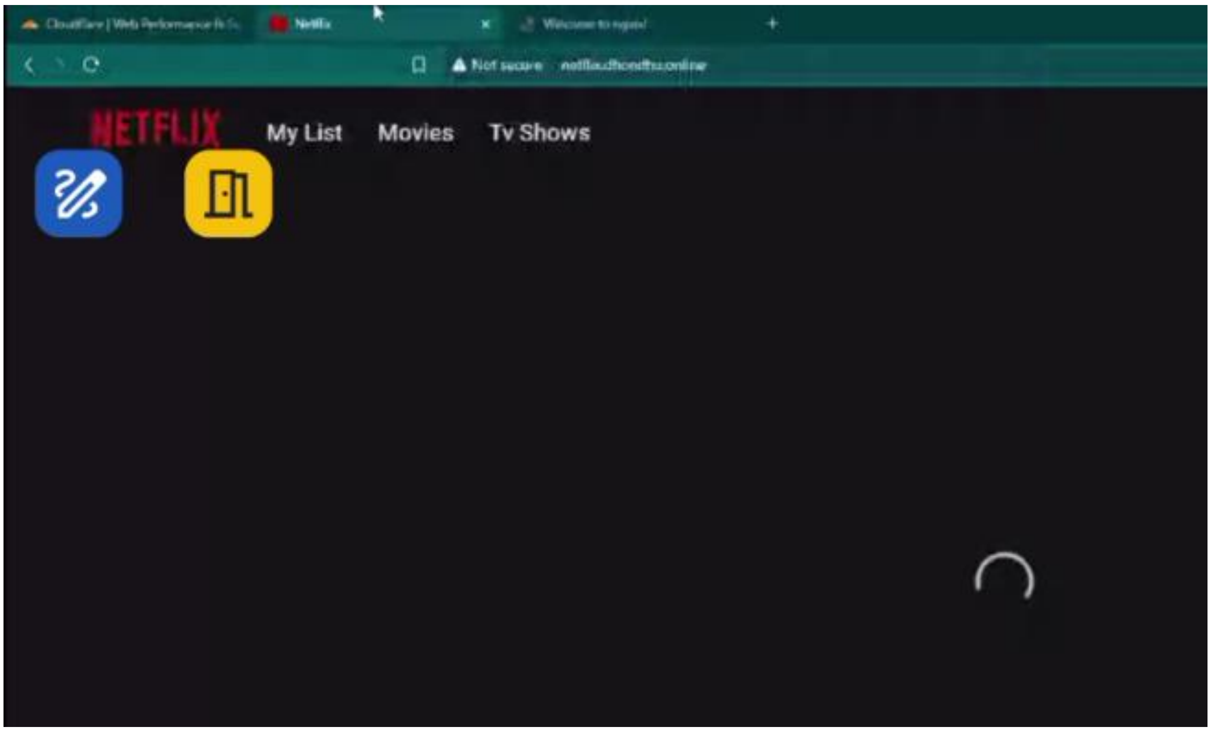
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

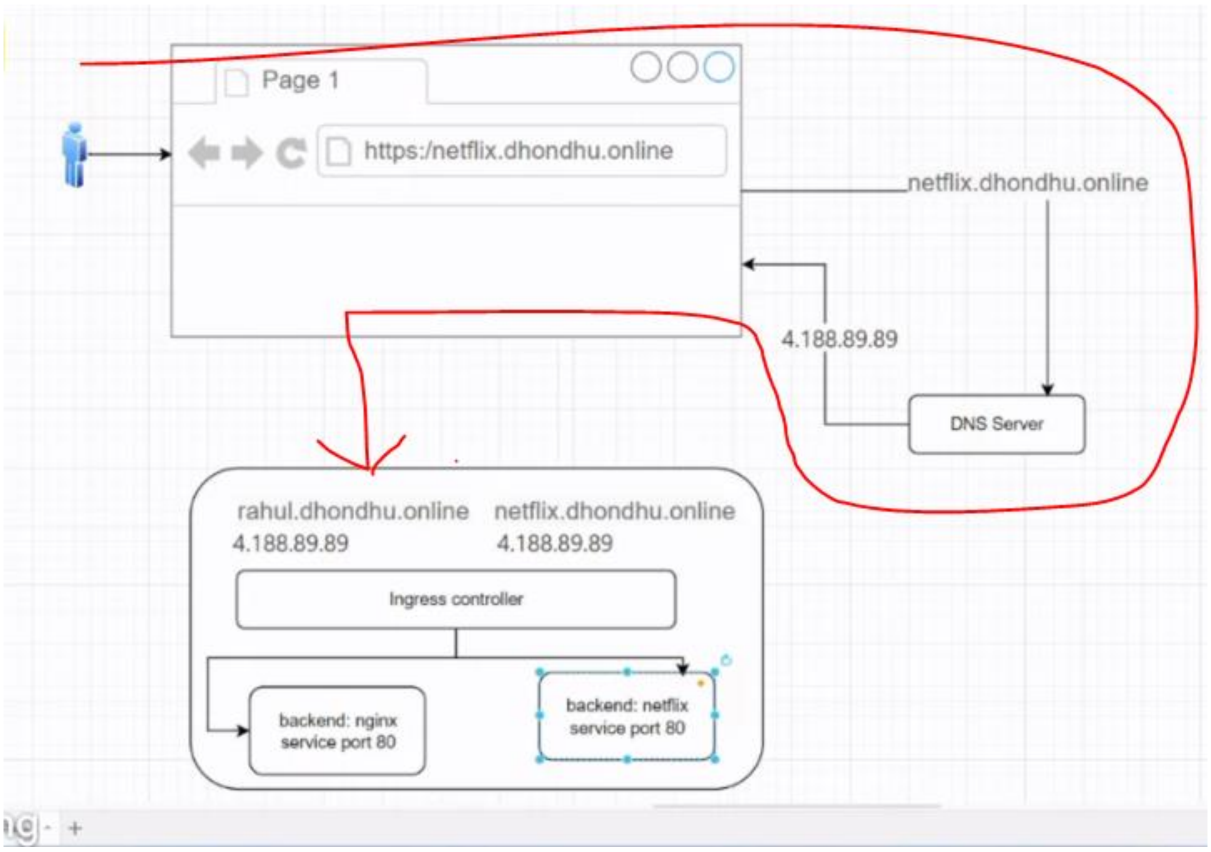
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

12) **netflix.dhondhu.online**

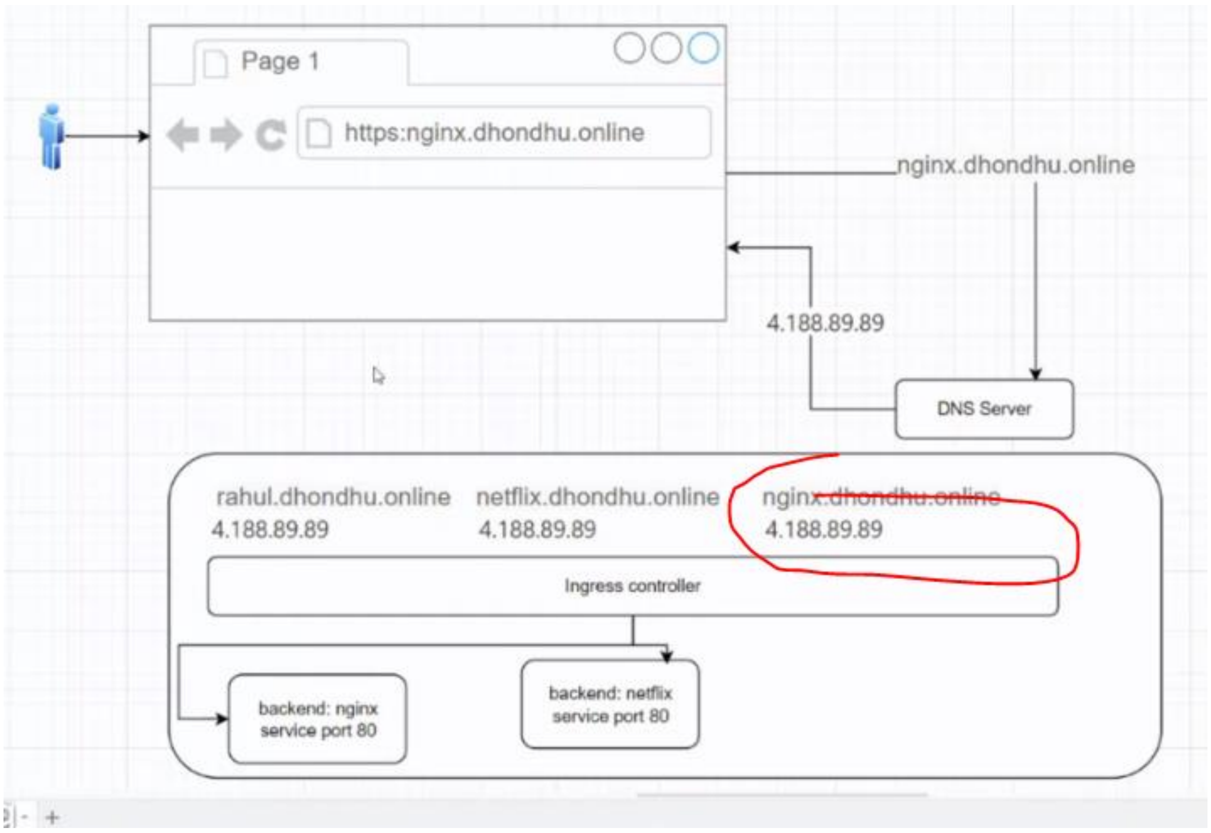


13) so the whole flow of ingress is



14) what is cloud flare? = works on tunnels

15) when 404 not found error comes = when we have not made any rule

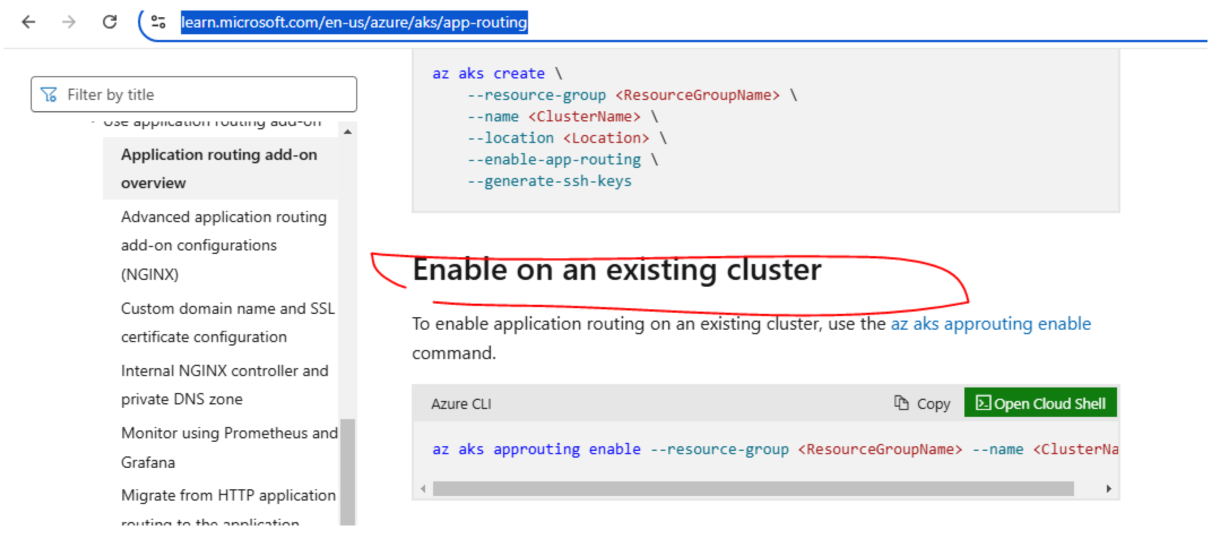


++

AGENDA – SOME ADVANCED TOPICS

1) SEARCH = application routing ingress controller aks

https://learn.microsoft.com/en-us/azure/aks/app-routing


learn.microsoft.com/en-us/azure/aks/app-routing
Filter by title
Use application routing add-on
Application routing add-on overview
Advanced application routing add-on configurations (NGINX)
Custom domain name and SSL certificate configuration
Internal NGINX controller and private DNS zone
Monitor using Prometheus and Grafana
Migrate from HTTP application routing to the application
Azure CLI
Copy
Open Cloud Shell

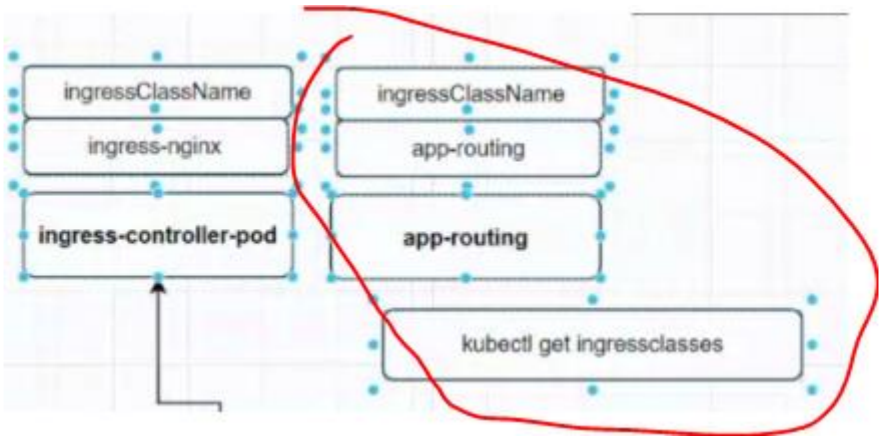
```
az aks create \
    --resource-group <ResourceGroupName> \
    --name <ClusterName> \
    --location <Location> \
    --enable-app-routing \
    --generate-ssh-keys
```

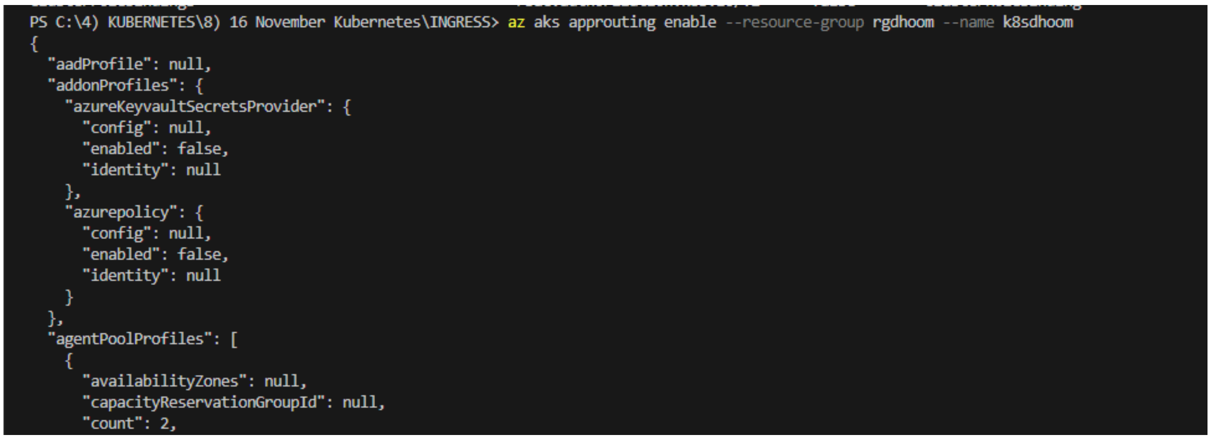
Enable on an existing cluster

To enable application routing on an existing cluster, use the az aks approuting enable command.

```
az aks approuting enable --resource-group <ResourceGroupName> --name <ClusterNa
```

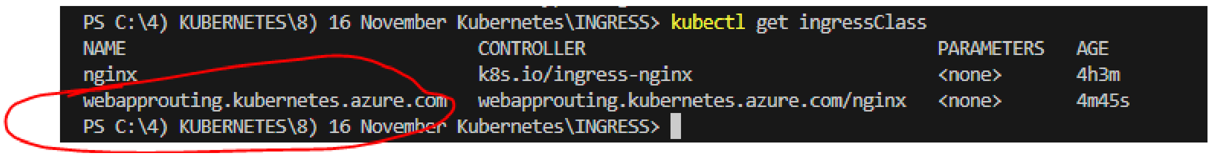
2) **az aks approuting enable --resource-group rgdhoom** --name k8sdhoom = is command se humare cluster me automatically ek aur ingress controller aakr baith jayega



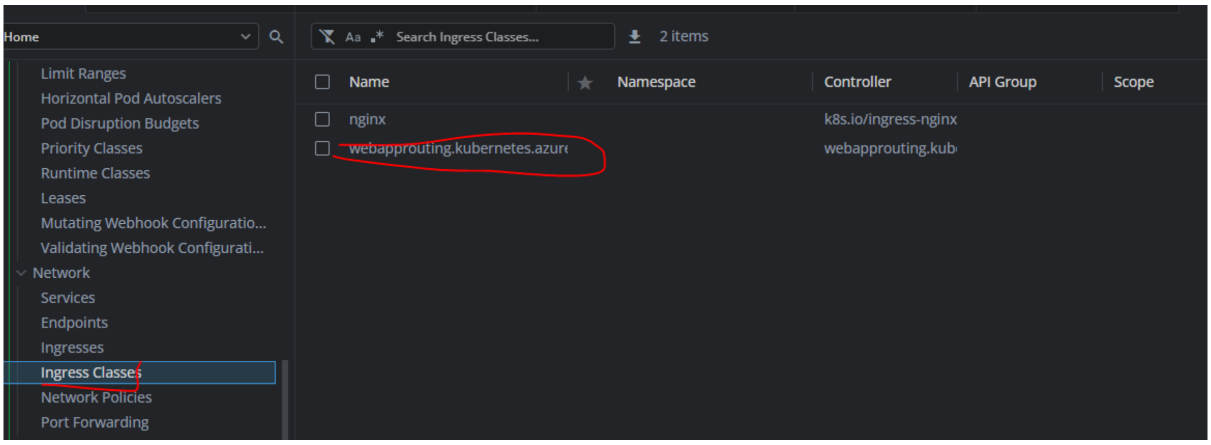


```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> az aks approuting enable --resource-group rgdhoom --name k8sdhoom
{
  "aadProfile": null,
  "addonProfiles": {
    "azureKeyvaultSecretsProvider": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "azurepolicy": {
      "config": null,
      "enabled": false,
      "identity": null
    }
  },
  "agentPoolProfiles": [
    {
      "availabilityZones": null,
      "capacityReservationGroupId": null,
      "count": 2,
```

3) kubectl get ingressClass



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get ingressClass
NAME                                CONTROLLER                                    PARAMETERS   AGE
nginx                               k8s.io/ingress-nginx                          <none>       4h3m
webapprouting.kubernetes.azure.com  webapprouting.kubernetes.azure.com/nginx      <none>       4m45s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

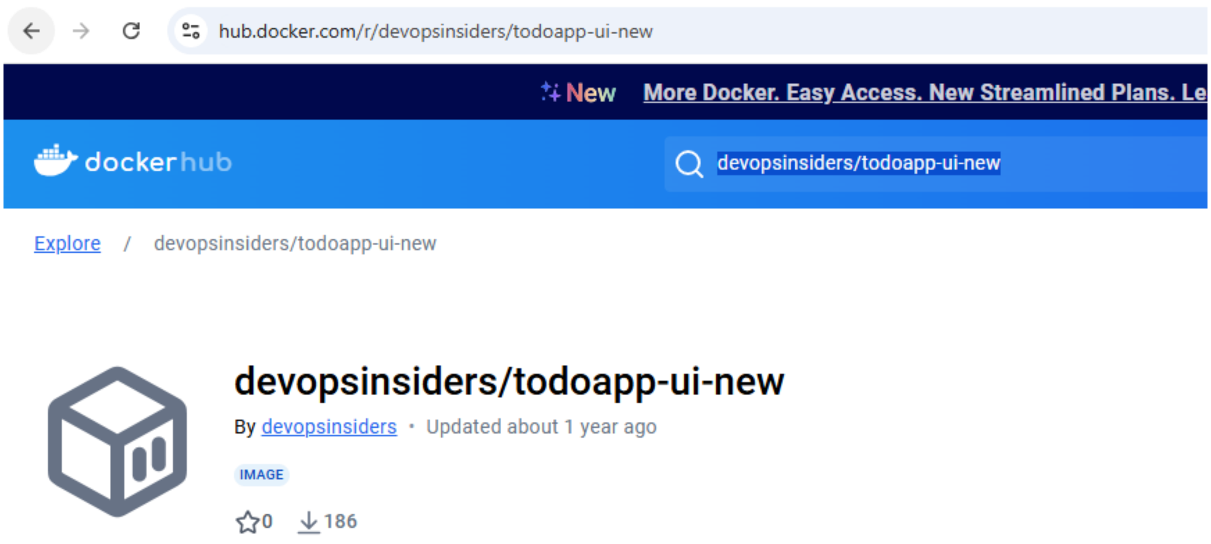


Name	Namespace	Controller	API Group	Scope
nginx		k8s.io/ingress-nginx		
webapprouting.kubernetes.azure		webapprouting.kub		

++

AGENDA – TODO UI

1) go to docker hub = devopsinsiders/todoapp-ui-new



hub.docker.com/r/devopsinsiders/todoapp-ui-new

New More Docker. Easy Access. New Streamlined Plans. Le

docker hub devopsinsiders/todoapp-ui-new

Explore / devopsinsiders/todoapp-ui-new

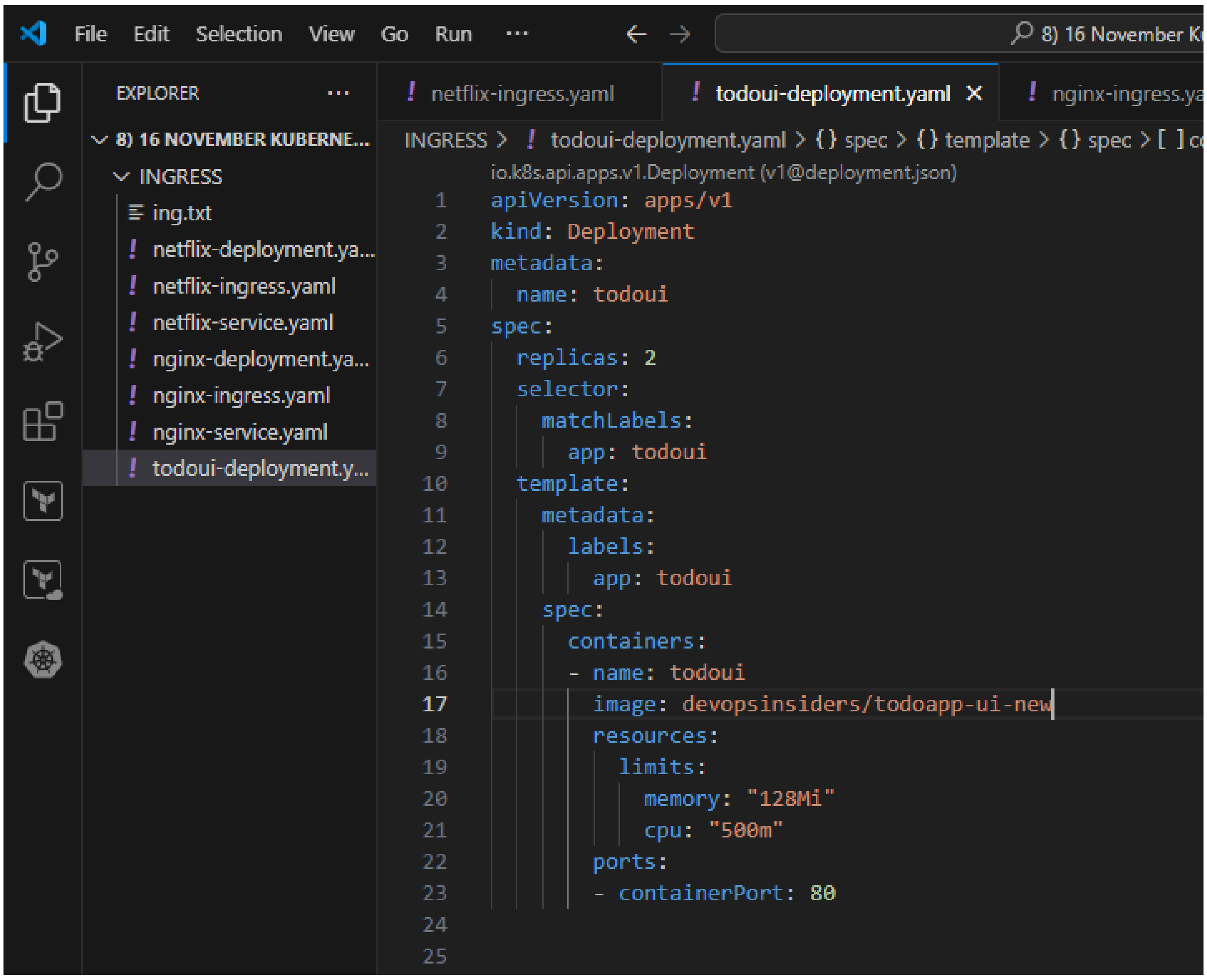
devopsinsiders/todoapp-ui-new

By devopsinsiders · Updated about 1 year ago

IMAGE

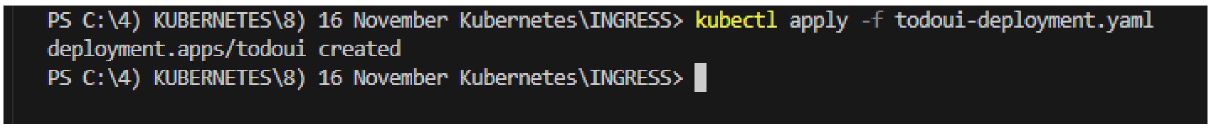
0 186

2) **create todoui-deployment.yaml**



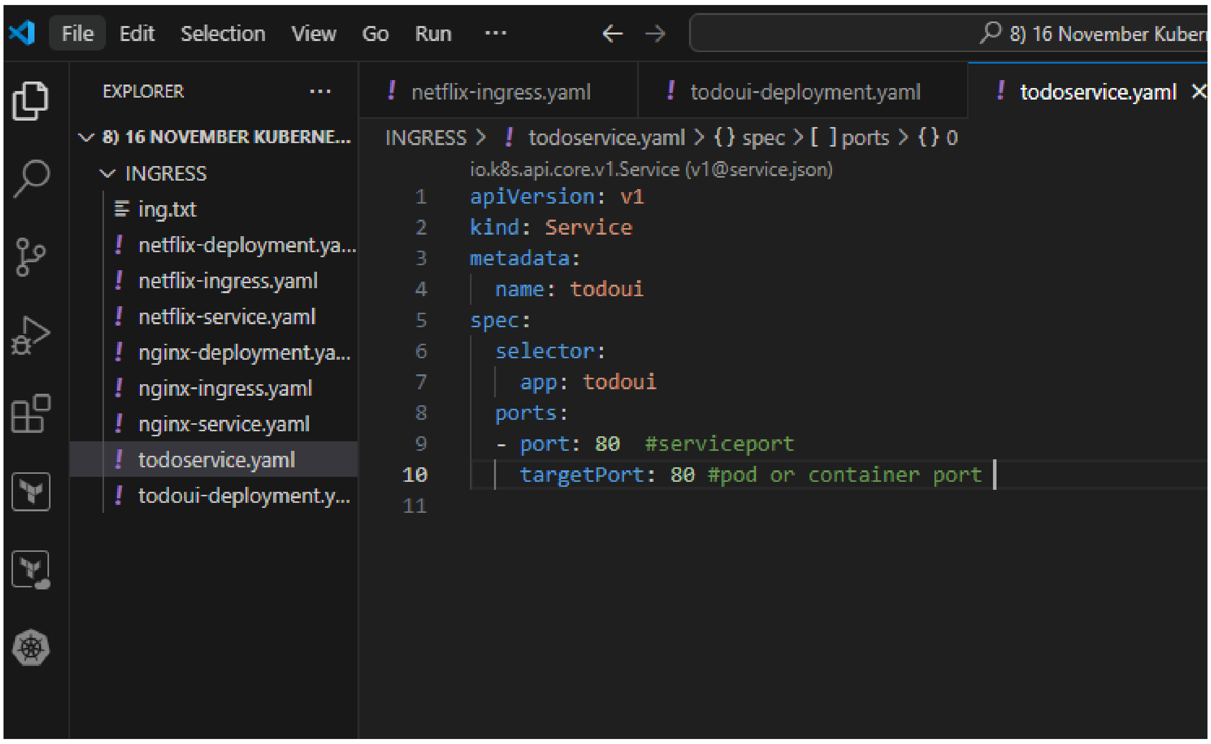
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: todoui
spec:
  replicas: 2
  selector:
    matchLabels:
      app: todoui
  template:
    metadata:
      labels:
        app: todoui
    spec:
      containers:
      - name: todoui
        image: devopsinsiders/todoapp-ui-new
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 80
```

kubectl apply -f todoui-deployment.yaml



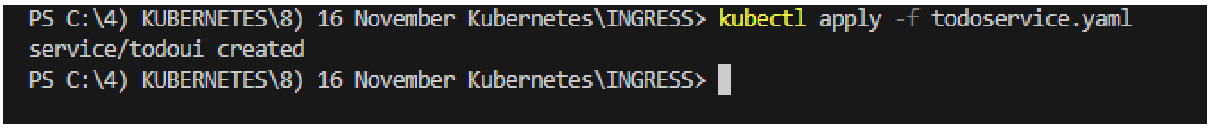
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f todoui-deployment.yaml
deployment.apps/todoui created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

3) **create todoservice.yaml**



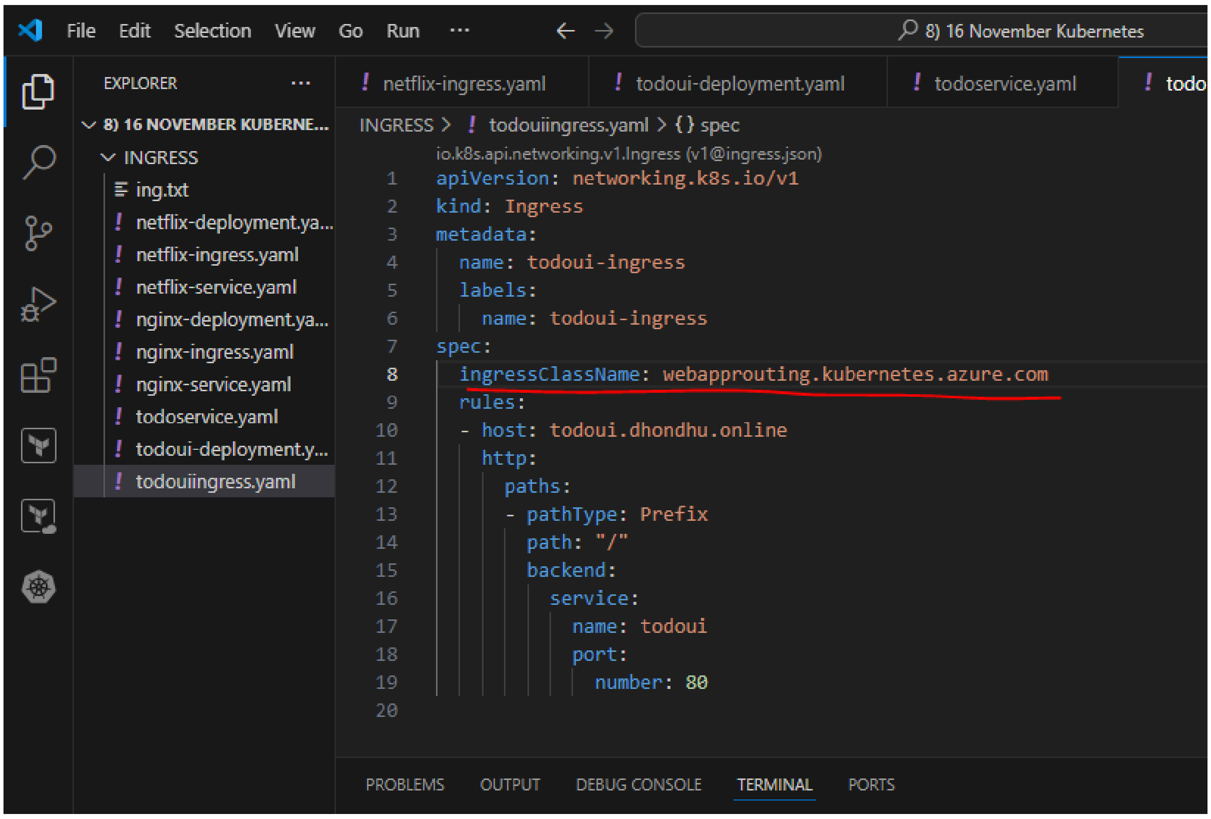
```yaml
apiVersion: v1
kind: Service
metadata:
  name: todoui
spec:
  selector:
    app: todoui
  ports:
  - port: 80  #serviceport
    targetPort: 80 #pod or container port
```

kubectl apply -f todoservice.yaml



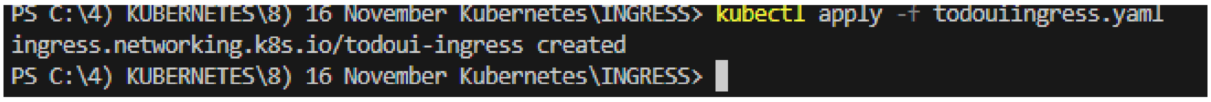
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f todoservice.yaml
service/todoui created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

4) **create todouiingress.yaml**



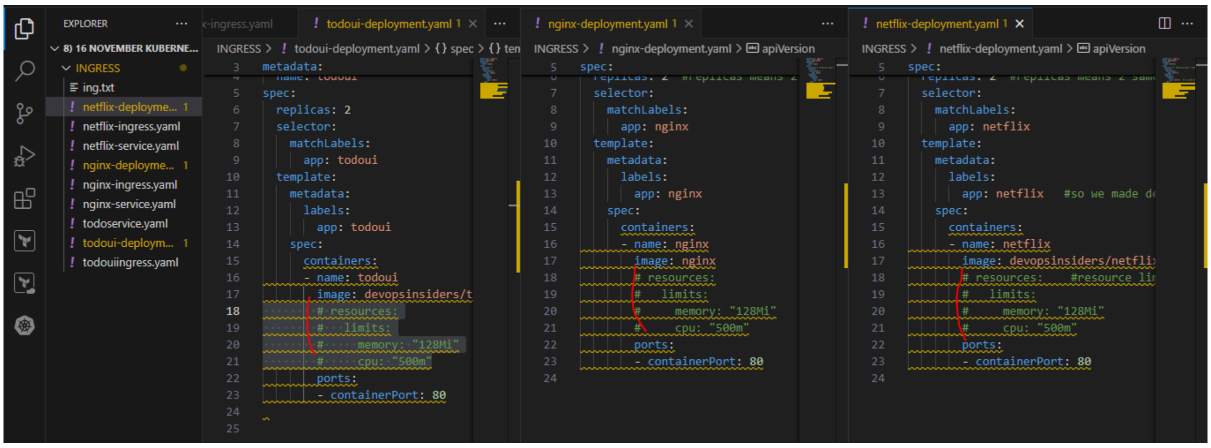
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: todoui-ingress
  labels:
    name: todoui-ingress
spec:
  ingressClassName: webapprouting.kubernetes.azure.com
  rules:
  - host: todoui.dhondhu.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: todoui
            port:
              number: 80
```

<span style="color:red">kubectl apply -f todouiingress.yaml</span>



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f todouiingress.yaml
ingress.networking.k8s.io/todoui-ingress created
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

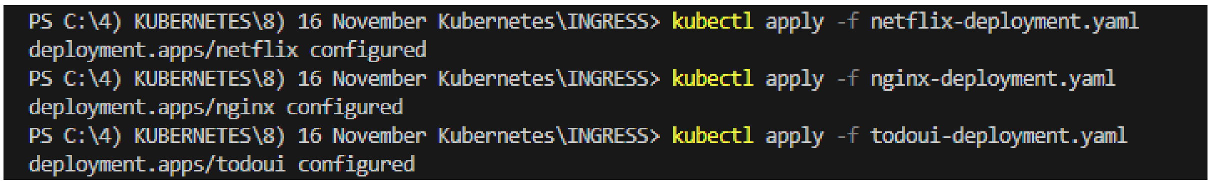
5) remove resource limits



```yaml
metadata:
  name: todoui
spec:
  replicas: 2
  selector:
    matchLabels:
      app: todoui
  template:
    metadata:
      labels:
        app: todoui
    spec:
      containers:
      - name: todoui
        image: devopsinsiders/t
        # resources:
        #   limits:
        #     memory: "128Mi"
        #     cpu: "500m"
        ports:
        - containerPort: 80
```

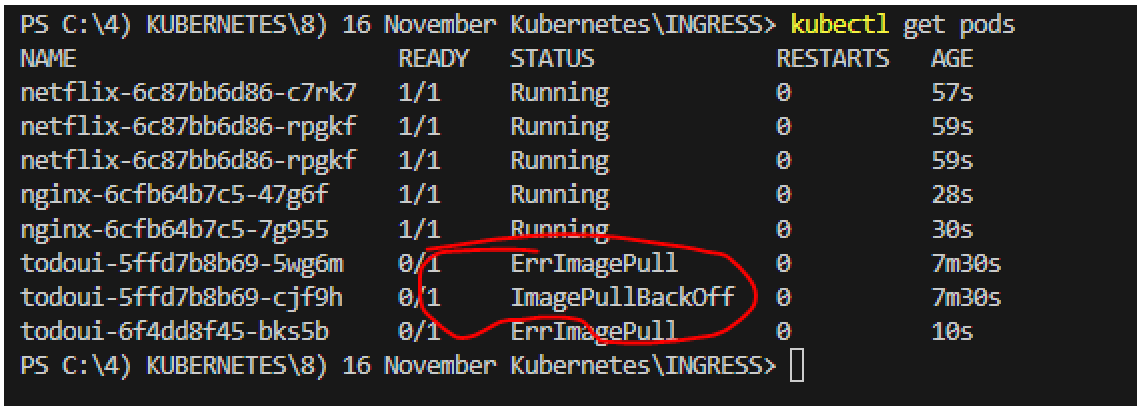
```yaml
spec:
  replicas: 2  #replicas means 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        # resources:
        #   limits:
        #     memory: "128Mi"
        #     cpu: "500m"
        ports:
        - containerPort: 80
```

```yaml
spec:
  replicas: 2  #replicas means 2 same
  selector:
    matchLabels:
      app: netflix
  template:
    metadata:
      labels:
        app: netflix   #so we made de
    spec:
      containers:
      - name: netflix
        image: devopsinsiders/netflix
        # resources:    #resource lin
        #   limits:
        #     memory: "128Mi"
        #     cpu: "500m"
        ports:
        - containerPort: 80
```



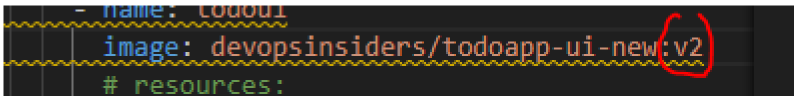
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f netflix-deployment.yaml
deployment.apps/netflix configured
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx configured
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl apply -f todoui-deployment.yaml
deployment.apps/todoui configured
```

6)



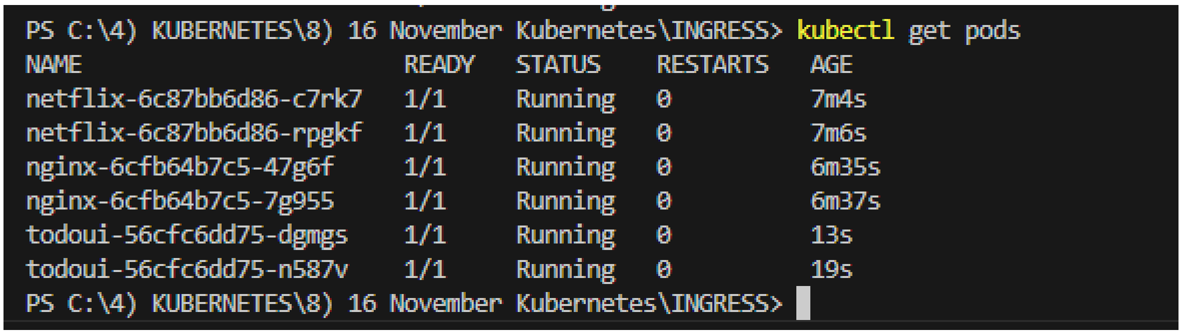
```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get pods
NAME                          READY   STATUS             RESTARTS   AGE
netflix-6c87bb6d86-c7rk7      1/1     Running            0          57s
netflix-6c87bb6d86-rpgkf      1/1     Running            0          59s
netflix-6c87bb6d86-rpgkf      1/1     Running            0          59s
nginx-6cfb64b7c5-47g6f        1/1     Running            0          28s
nginx-6cfb64b7c5-7g955        1/1     Running            0          30s
todoui-5ffd7b8b69-5wg6m       0/1     ErrImagePull       0          7m30s
todoui-5ffd7b8b69-cjf9h       0/1     ImagePullBackOff   0          7m30s
todoui-6f4dd8f45-bks5b        0/1     ErrImagePull       0          10s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

This error coming due to tag issue



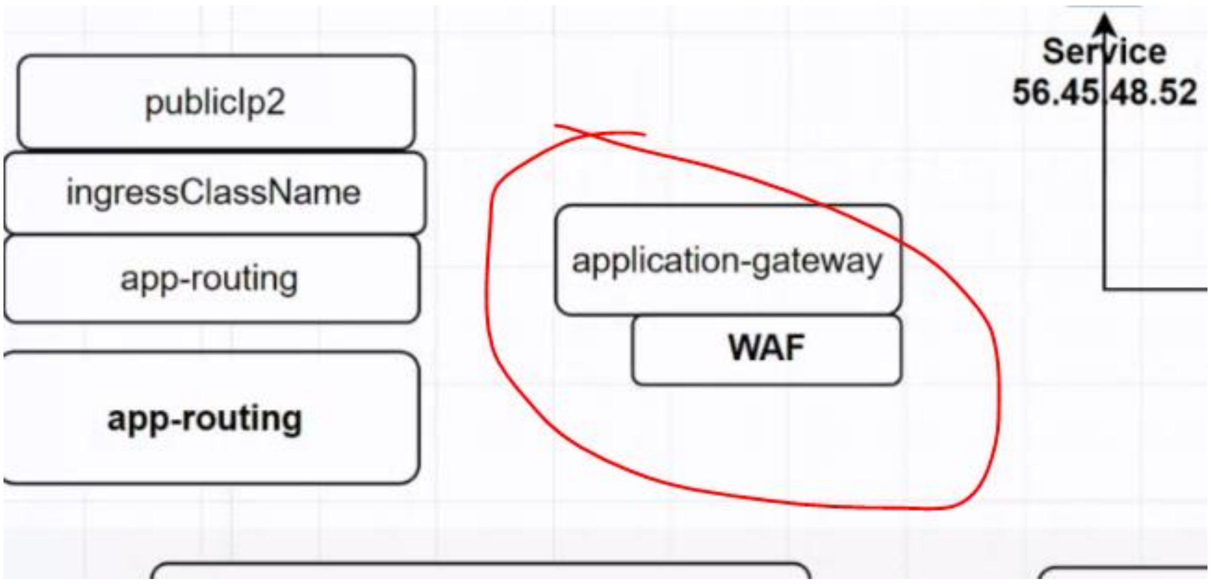
```
      - name: todoui
        image: devopsinsiders/todoapp-ui-new:v2
        # resources:
```

Now issue solved



```
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS> kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
netflix-6c87bb6d86-c7rk7      1/1     Running   0          7m4s
netflix-6c87bb6d86-rpgkf      1/1     Running   0          7m6s
nginx-6cfb64b7c5-47g6f        1/1     Running   0          6m35s
nginx-6cfb64b7c5-7g955        1/1     Running   0          6m37s
todoui-56cfc6dd75-dgmgs       1/1     Running   0          13s
todoui-56cfc6dd75-n587v       1/1     Running   0          19s
PS C:\4) KUBERNETES\8) 16 November Kubernetes\INGRESS>
```

7)



This one is difficult ingress which needs different configuration cluster

++