Ingress controller

1. Create a daphne deployment with yaml file

Sudo nano bdittodaphnedeployment.yaml

Kubectl apply –f bdittodaphnedeployment.yaml –-namespace biditto-dev

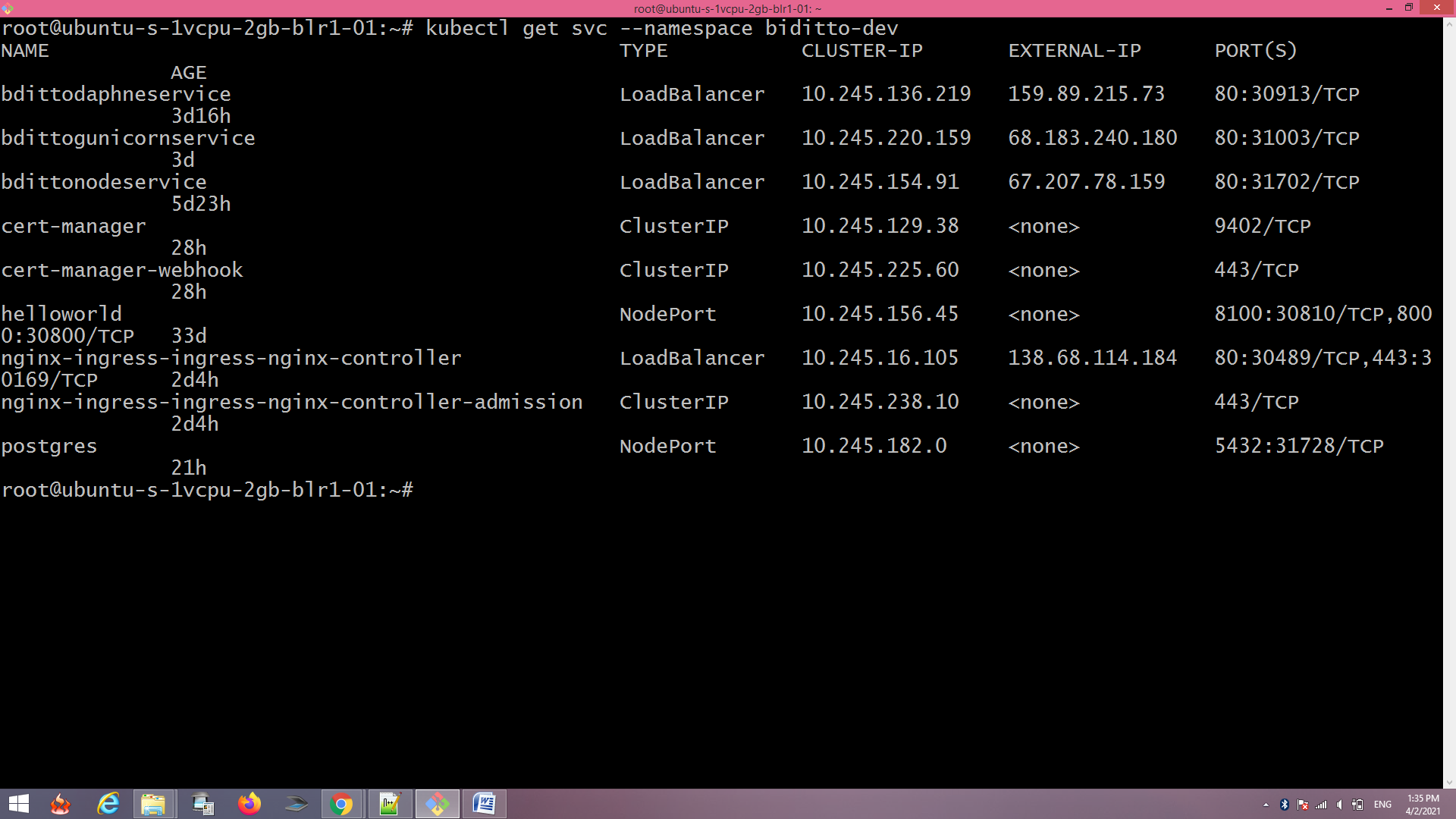
1. Create a gunicorn deployment with yaml file

Sudo nano bdittogunicorndeployment.yaml

Kubectl apply -f bdittogunicorndeployment.yaml –-namespace biditto-dev

After applying both the yaml files pods,services and deployment are up and running

We can see bdittodaphneservice and bdittogunicornservice up and running with having external ip



Configuring ingress

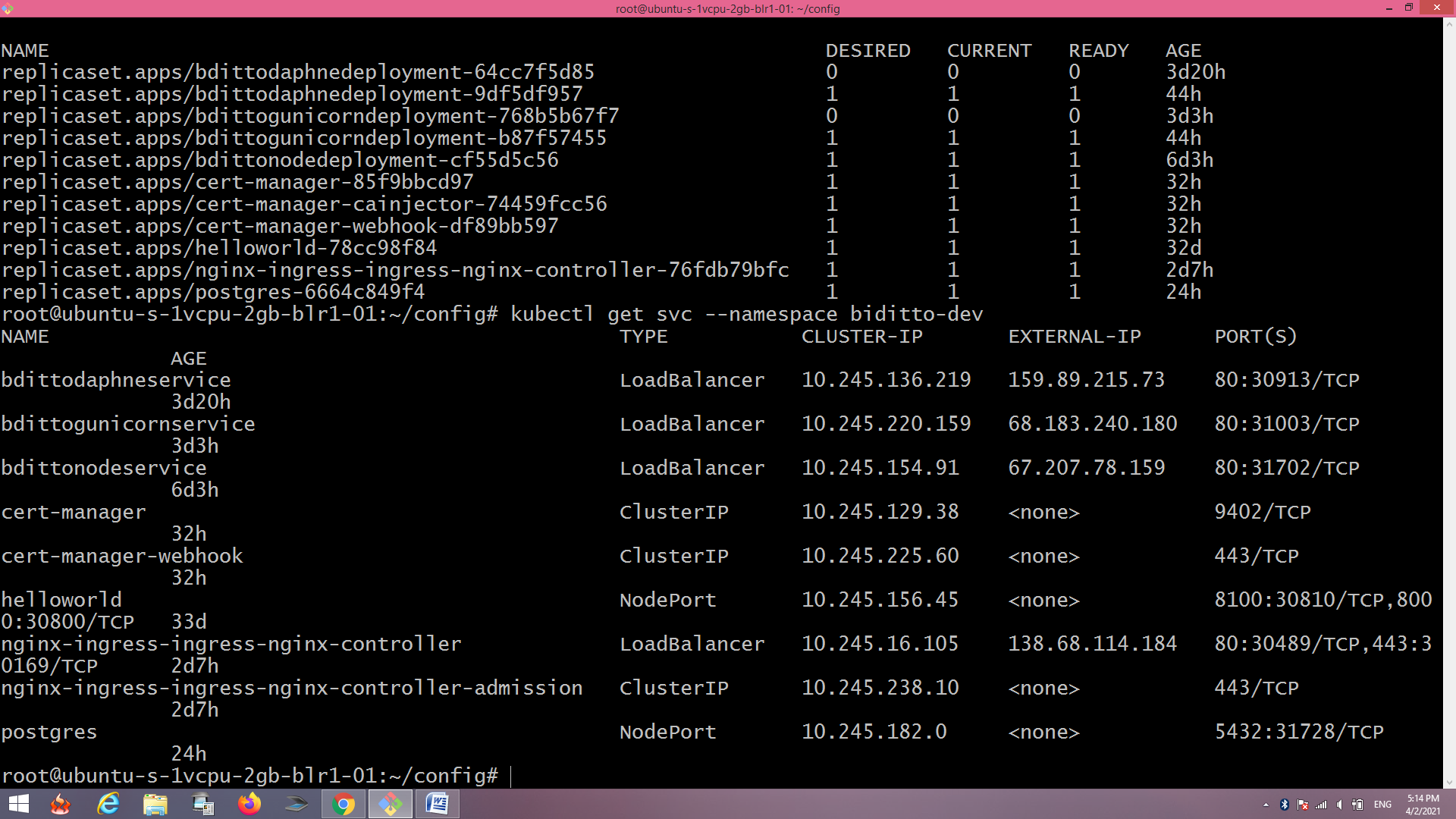
helm repo add ingress-nginx <https://kubernetes.github.io/ingress-nginx>

helm repo update

helm install nginx-ingress ingress-nginx/ingress-nginx –namespace biditto-dev --set controller.publishService.enabled=true

kubectl --namespace biditto-dev get services -o wide -w nginx-ingress-ingress-nginx-controller

Next, you’ll need to ensure that your two domains are pointed to the Load Balancer via A records. This is done through your DNS provider. To configure your DNS records on DigitalOcean, see [How to Manage DNS Records](https://www.digitalocean.com/docs/networking/dns/how-to/manage-records/).



In our case our ingress loadbalancer ip 138.68.114.184

We have to add A record entry in dns

With both domain l

create the ingress with

sudo nano hello-kubernetes-ingress.yaml

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: hello-kubernetes-ingress

annotations:

kubernetes.io/ingress.class: nginx

spec:

rules:

- host: "async.dev.bditto.com"

http:

paths:

- pathType: Prefix

path: "/"

backend:

service:

name: bdittodaphneservice

port:

number: 80

- host: "http.dev.bditto.com"

http:

paths:

- pathType: Prefix

path: "/"

backend:

service:

name: bdittogunicornservice

port:

number: 80

kubectl apply -f hello-kubernetes-ingress.yaml –namespace biditto-dev

now both domains are working

http.dev.bditto.com for gunicorn

async.dev.bditto.com for daphne

1. Configuring SSL for both the domain

kubectl create namespace cert-manager

helm repo add jetstack <https://charts.jetstack.io>

helm repo update

helm install cert-manager jetstack/cert-manager --namespace bditto-dev --version v1.2.0 --set installCRDs=true

sudo nano production\_issuer.yaml

apiVersion: cert-manager.io/v1

kind: ClusterIssuer

metadata:

name: letsencrypt-prod

spec:

acme:

# Email address used for ACME registration

email: your\_email\_address

server: https://acme-v02.api.letsencrypt.org/directory

privateKeySecretRef:

# Name of a secret used to store the ACME account private key

name: letsencrypt-prod-private-key

# Add a single challenge solver, HTTP01 using nginx

solvers:

- http01:

ingress:

class: nginx

kubectl apply -f production\_issuer.yaml --namespace bditto-dev

sudo nano hello-kubernetes-ingress.yaml

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: hello-kubernetes-ingress

annotations:

kubernetes.io/ingress.class: nginx

cert-manager.io/cluster-issuer: letsencrypt-prod

spec:

tls:

- hosts:

- async.dev.bditto.com

- http.dev.bditto.com

secretName: hello-kubernetes-tls

rules:

- host: " async.dev.bditto.com "

http:

paths:

- pathType: Prefix

path: "/"

backend:

service:

name: bdittodaphneservice

port:

number: 80

- host: " http.dev.bditto.com "

http:

paths:

- pathType: Prefix

path: "/"

backend:

service:

name: bdittogunicornservice

port:

number: 80

kubectl apply -f hello-kubernetes-ingress.yaml –namespace bditto-dev

kubectl describe certificate hello-kubernetes-tls –namespace bditto-dev

you can check both domain are working with https