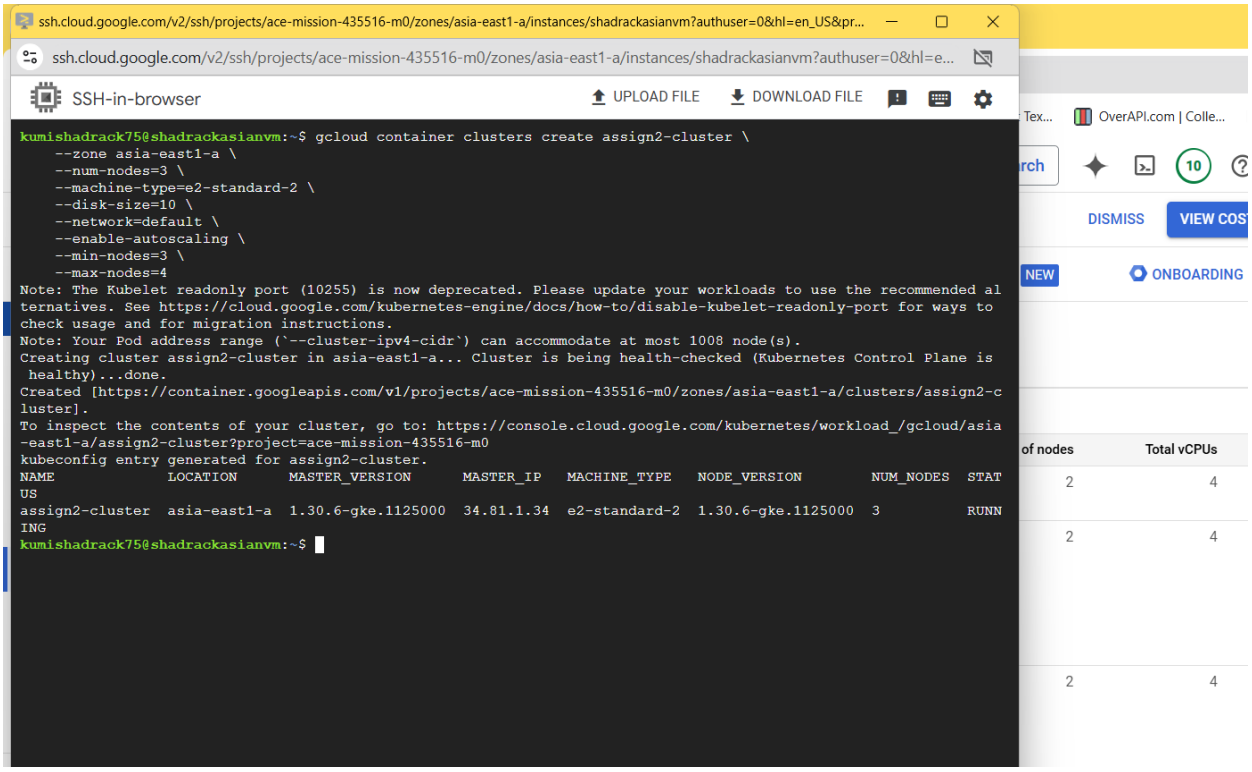
Set Up the GKE Cluster:

- Create a Google Cloud Project.
- Enable the **Google Kubernetes Engine API**.
- Create a GKE cluster:

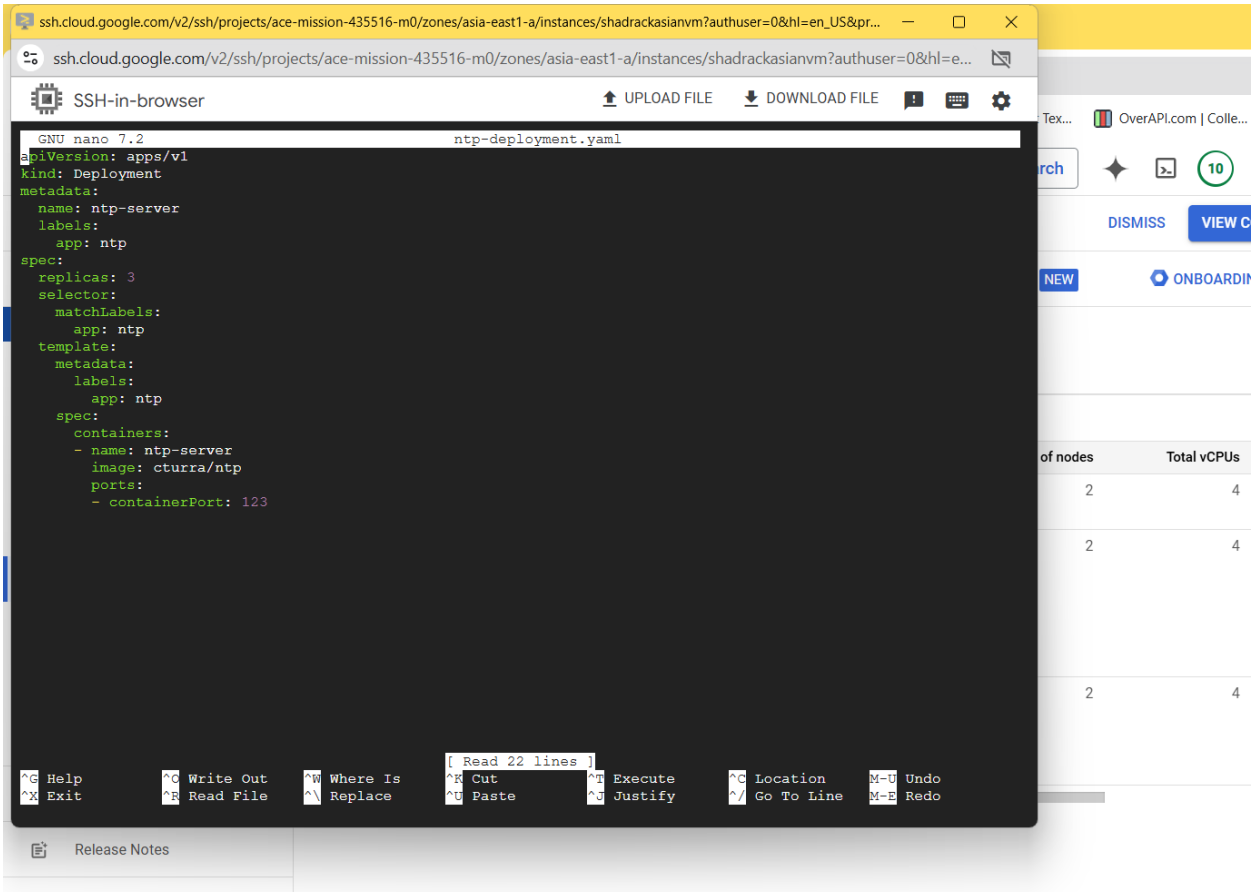
```
gcloud container clusters create assign2-cluster \
--zone asia-east1-a \
  --num-nodes=3 \
  --machine-type=e2-standard-2 \
  --disk-size=10 \
  --network=default \
  --enable-autoscaling \
  --min-nodes=3 \
  --max-nodes=4
```



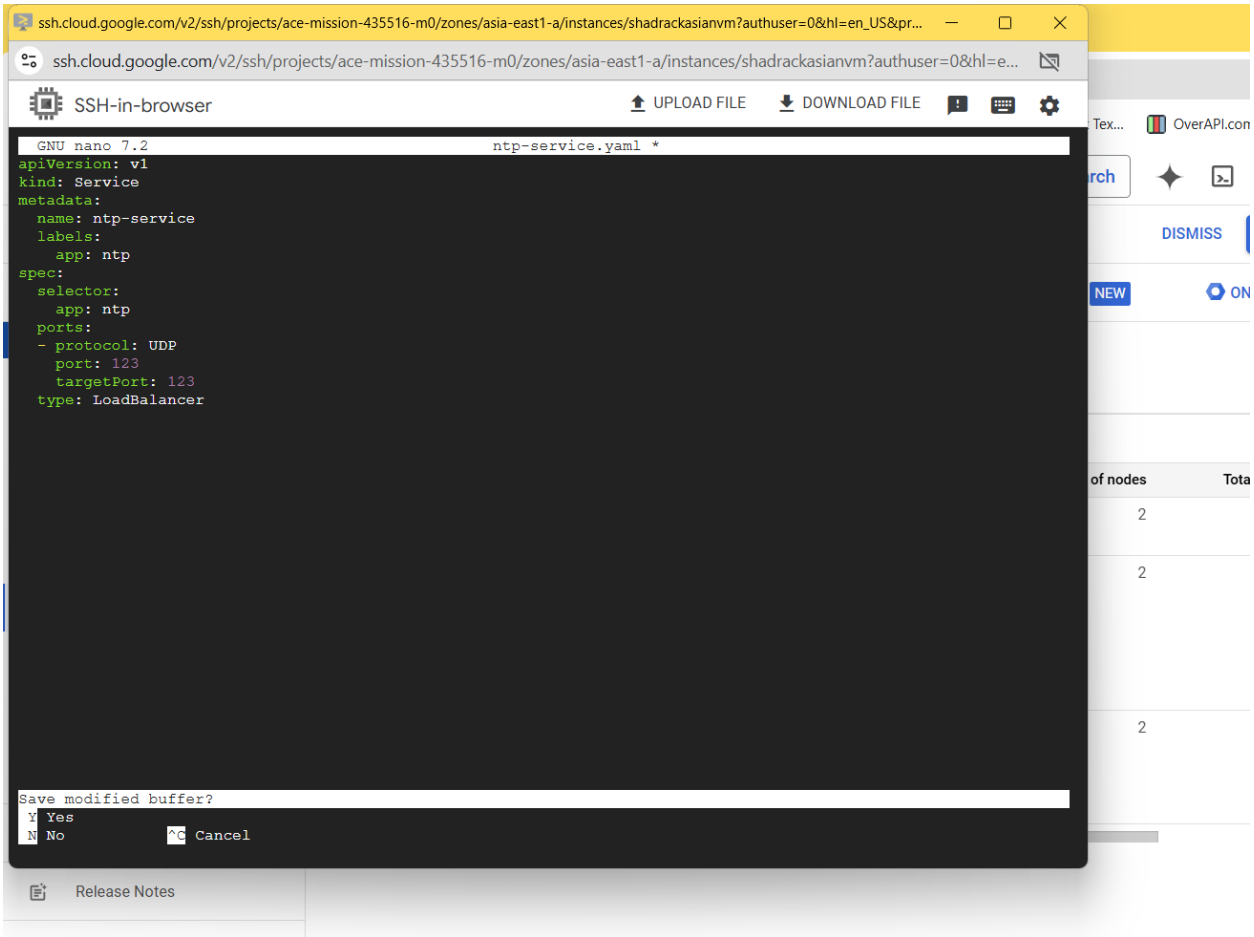
Ensure the **VPC Network** and **firewall rules** are set up to allow external access to the LoadBalancer service

Deploy NTP Server Containers:

- Select a suitable NTP container image (e.g., cturra/ntp).
- Write a Kubernetes Deployment manifest with three replicas of the NTP server:



Create a LoadBalancer service to expose the NTP servers:




```
NAME              LOCATION       MASTER_VERSION       MASTER_IP    MACHINE_TYPE    NODE_VER
US
assign2-cluster   asia-east1-a   1.30.6-gke.1125000   34.81.1.34   e2-standard-2   1.30.6-g
ING
kumishadrack75@shadrackasianvm:~$ nano ntp-deployment.yaml
kumishadrack75@shadrackasianvm:~$ nano ntp-deployment.yaml
kumishadrack75@shadrackasianvm:~$ nano ntp-service.yaml
kumishadrack75@shadrackasianvm:~$
```

Configure the NTP Server:

- Customize the container's configuration if the image supports it, using ConfigMaps or environment variables.

- Restrict access by applying Kubernetes Network Policies.

Deploy the Manifests:

- Apply the manifests using kubectl

```
ING
kumishadrack75@shadrackasianvm:~$ nano ntp-deployment.yaml
kumishadrack75@shadrackasianvm:~$ nano ntp-deployment.yaml
kumishadrack75@shadrackasianvm:~$ nano ntp-service.yaml
kumishadrack75@shadrackasianvm:~$ kubectl apply -f ntp-deployment.yaml
kubectl apply -f ntp-service.yaml
deployment.apps/ntp-server created
service/ntp-service created
kumishadrack75@shadrackasianvm:~$
```

Test the Cluster:

Retrieve the external IP of the LoadBalancer:

```
kumishadrack75@shadrackasianvm:~$ kubectl apply -f ntp-deployment.yaml
kubectl apply -f ntp-service.yaml
deployment.apps/ntp-server created
service/ntp-service created
kumishadrack75@shadrackasianvm:~$ kubectl get service ntp-service
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)         AGE
ntp-service   LoadBalancer   34.118.230.209   <pending>     123:32683/UDP   35s
kumishadrack75@shadrackasianvm:~$
```

Network

Linux

Use ntpdate or chronyc to test synchronization:

```
kumishadrack75@shadrackasianvm:~$ kubectl apply -f ntp-deployment.yaml
kubectl apply -f ntp-service.yaml
deployment.apps/ntp-server created
service/ntp-service created
kumishadrack75@shadrackasianvm:~$ kubectl get service ntp-service
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP    PORT(S)         AGE
ntp-service   LoadBalancer   34.118.230.209   <pending>      123:32683/UDP   35s
kumishadrack75@shadrackasianvm:~$ ntpdate -q 34.118.230.209
-bash: ntpdate: command not found
kumishadrack75@shadrackasianvm:~$ kubectl get service ntp-service
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
ntp-service   LoadBalancer   34.118.230.209   34.81.163.131   123:32683/UDP   2m1s
kumishadrack75@shadrackasianvm:~$ kubectl describe service ntp-service
Name:                     ntp-service
Namespace:                default
Labels:                   app=ntp
Annotations:              cloud.google.com/neg: {"ingress":true}
Selector:                 app=ntp
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       34.118.230.209
IPs:                      34.118.230.209
LoadBalancer Ingress:     34.81.163.131
Port:                     <unset>  123/UDP
TargetPort:               123/UDP
NodePort:                 <unset>  32683/UDP
Endpoints:                10.72.0.12:123,10.72.1.5:123,10.72.2.6:123
Session Affinity:         None
External Traffic Policy:  Cluster
Events:
  Type    Reason                Age    From                Message
  ----    ------                ----   ----                -------
  Normal  EnsuringLoadBalancer  2m12s  service-controller  Ensuring load balancer
  Normal  EnsuredLoadBalancer   81s    service-controller  Ensured load balancer
kumishadrack75@shadrackasianvm:~$
```

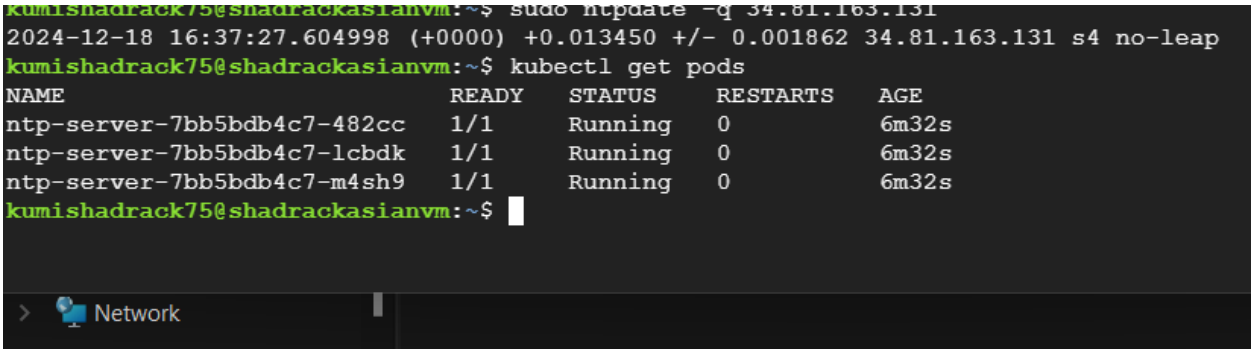
```
kumishadrack75@shadrackasianvm:~$ sudo ntpdate -q 34.81.163.131
2024-12-18 16:37:27.604998 (+0000) +0.013450 +/- 0.001862 34.81.163.131 s4 no-leap
kumishadrack75@shadrackasianvm:~$
```

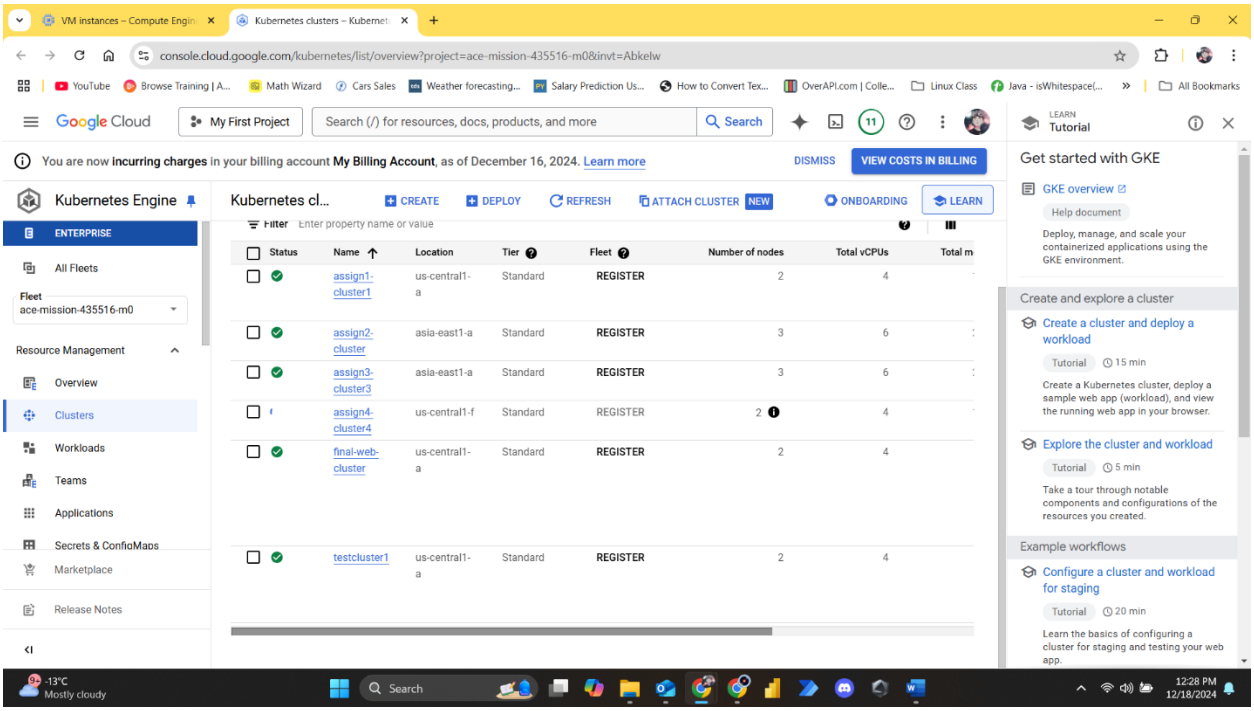
Simulate a failure by deleting a pod:

Monitor and Document:

- Enable **Google Cloud Monitoring** for insights into cluster health.
- Use **kubectl logs** and monitoring dashboards to validate pod health.

GET NODES :





SUMMARY

Step 1: Created a GKE Cluster

I started by creating a GKE cluster with three nodes using the gcloud CLI. I ensured the cluster was in the us-central1 region and used e2-medium machine types for the nodes. I also confirmed that the VPC network and firewall rules allowed external access to the LoadBalancer.

Step 2: Wrote and Applied Kubernetes Manifests

I created two Kubernetes manifest files:

1. **Deployment (ntp-deployment.yaml)**: This defined a deployment with three replicas of an NTP server container using the cturra/ntp image.
2. **Service (ntp-service.yaml)**: This exposed the deployment as a LoadBalancer service on UDP port 123.

I applied these manifests using kubectl commands:

kubectl apply -f ntp-deployment.yaml

kubectl apply -f ntp-service.yaml

Step 3: Checked LoadBalancer Service
After deploying the service, I ran kubectl get service ntp-service to retrieve its details. Initially, the EXTERNAL-IP field showed <pending>, indicating the LoadBalancer was still provisioning.

Step 4: Troubleshooting the LoadBalancer
To debug the issue, I used:

kubectl describe service ntp-service

Step 5: Installed ntpdate for Testing
On my client machine, I noticed that ntpdate was not installed when I attempted to test the NTP server. To resolve this, I installed ntpdate using:

sudo apt update

sudo apt install ntpdate -y

Step 6: Tested the NTP Server
Once the LoadBalancer's external IP became available, I used the following command to test the NTP server's functionality:

ntpdate -q <EXTERNAL-IP>

This allowed me to verify accurate time synchronization.