Deploying C++ application on GKE

Overview

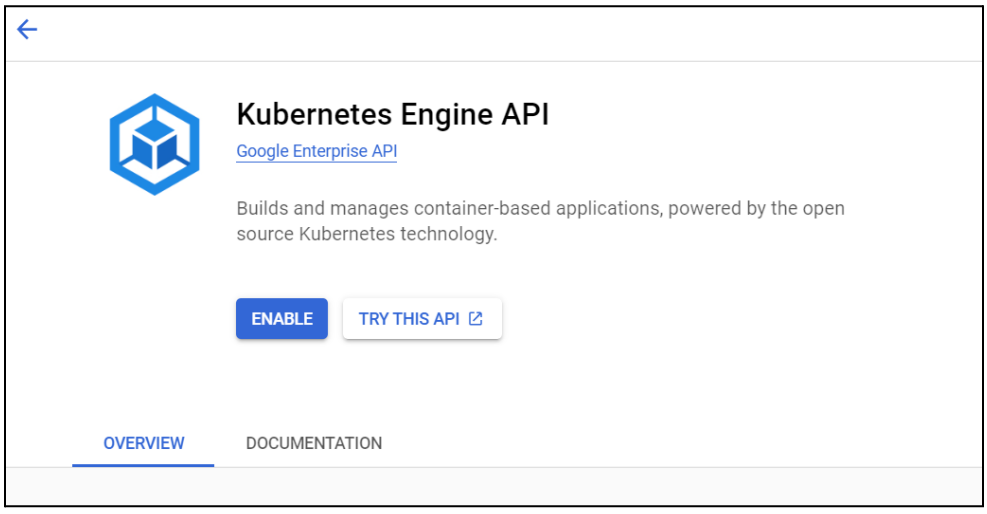
Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure. The GKE environment consists of multiple machines (specifically, Compute Engine instances) grouped together to form a cluster.

In this hands-on lab, you'll create Google Kubernetes Engine (GKE) cluster using the Google Cloud Console.

Prerequisites

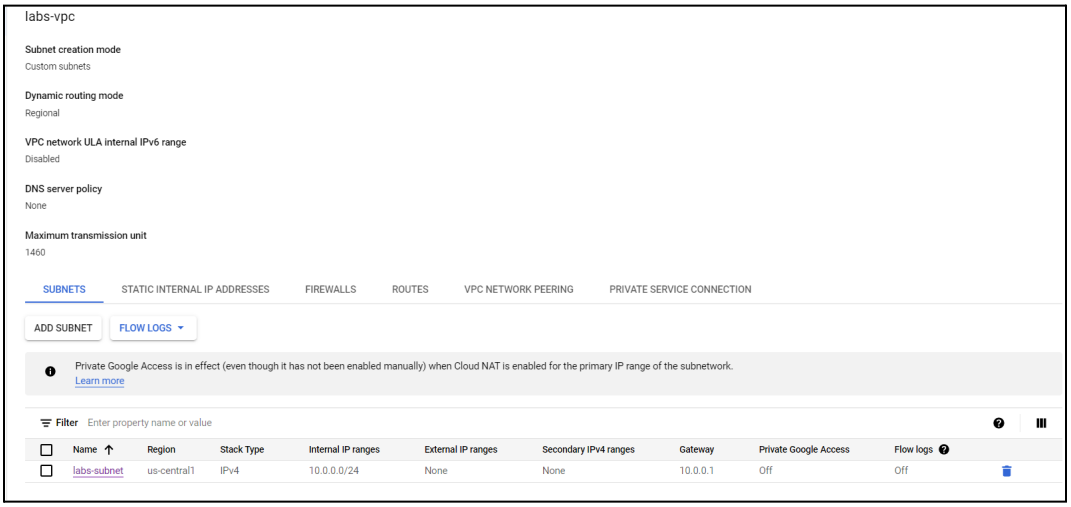
To create a Google Kubernetes Engine (GKE) cluster in GCP following resources such as secondary IP ranges for Nodes and Pods, service account and firewall rule for health checks is required to be provisioned

Enable the Kubernetes Engine API - Kubernetes Engine API

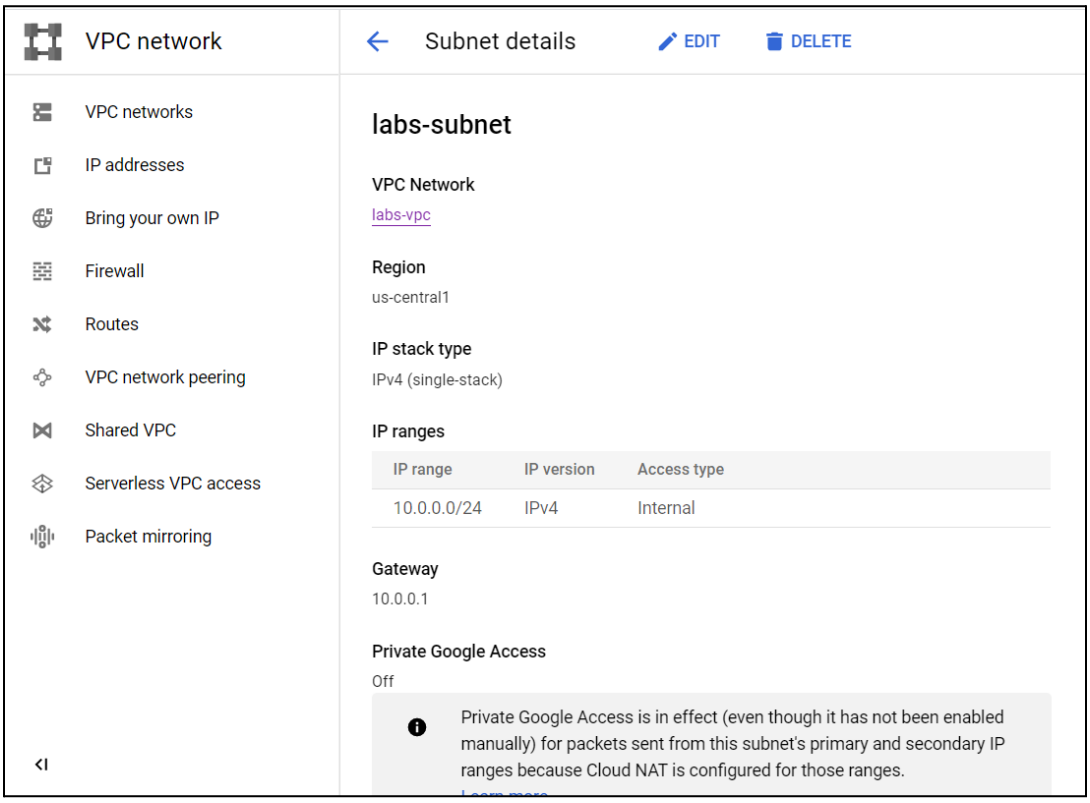


1. Create a **Secondary IP Ranges** for Pods and Service

- In the Cloud Console, on the **Navigation menu** (☰), click **VPC network** > **VPC networks**.
- Search VPC created previously i.e **labs-vpc**, click on **labs-vpc**.

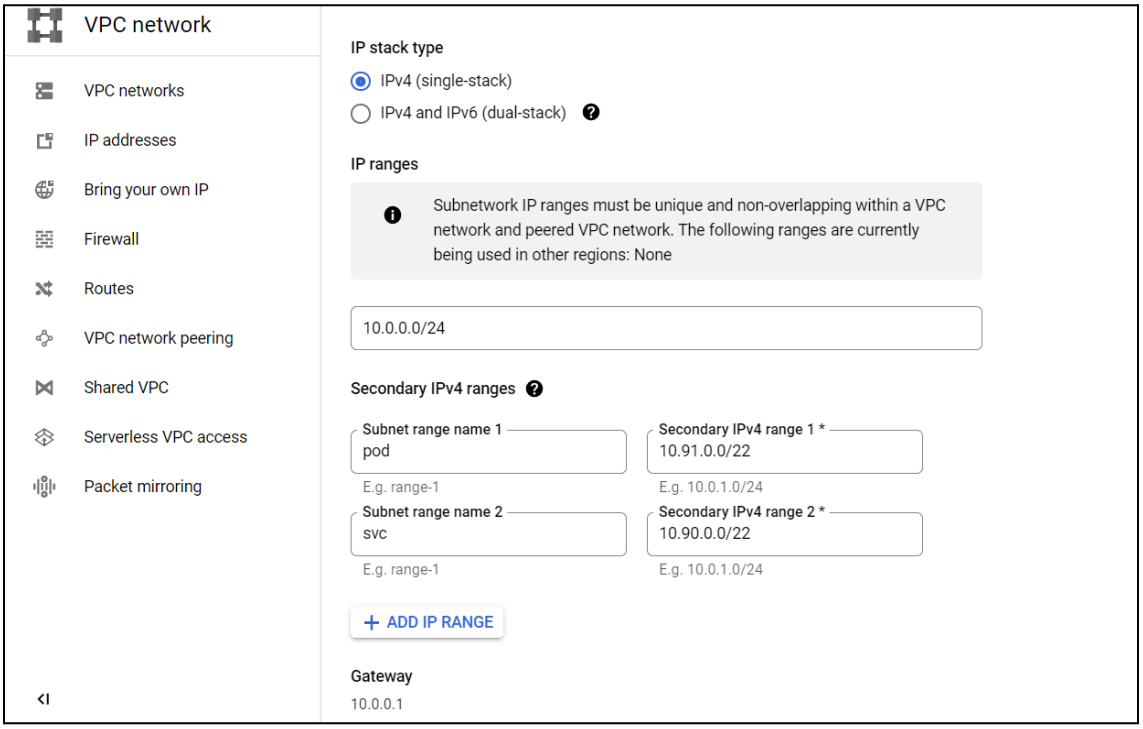


- In the subnet section click on **labs-subnet**, to add Secondary ranges.

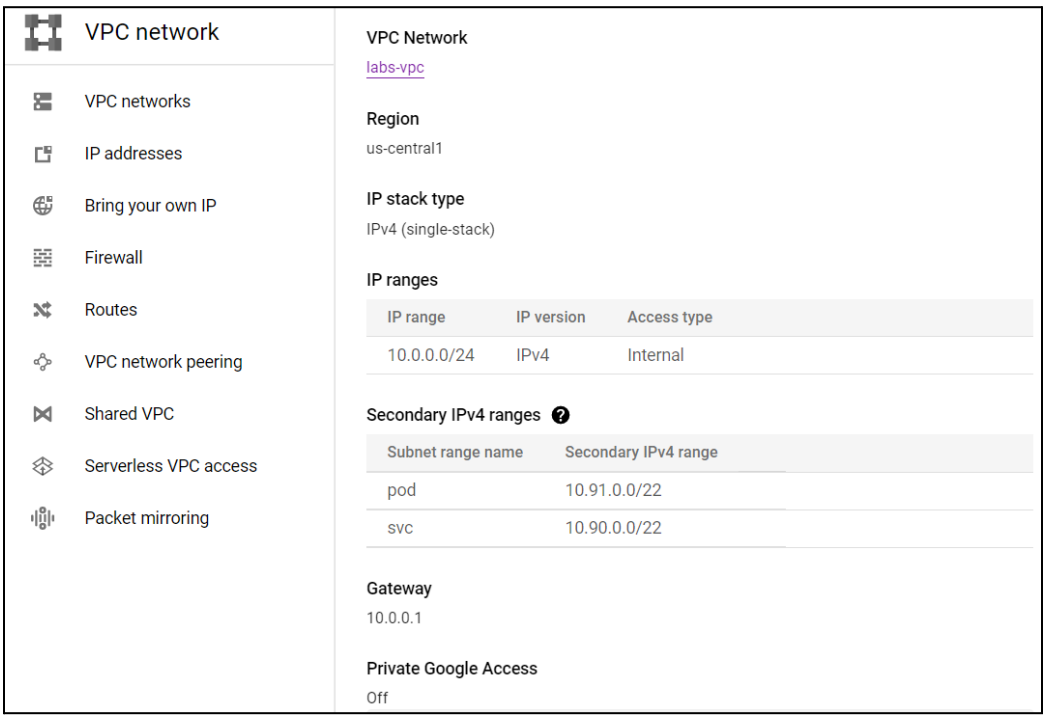


- Click on **EDIT**, to add secondary ranges.
- In **Secondary IPv4 ranges**, click on **ADD IP RANGE.**

Add **subnet range name 1** as pod and **Secondary IPv4 range 1** as 10.91.0.0/22.
Add **subnet range name 2** as svc and **Secondary IPv4 range 2** as 10.90.0.0/22.



- Click **SAVE** to add the secondary ranges. The secondary ranges for pods and service are created.



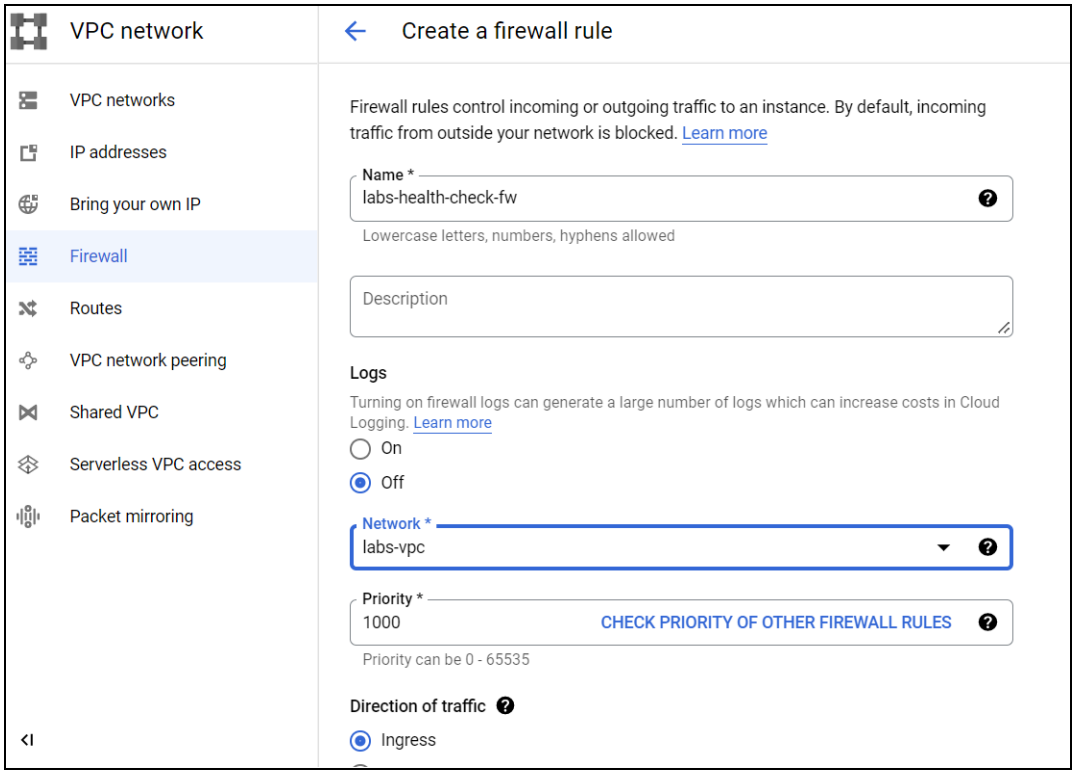
2. Create **Health check Firewall Rule**

You need to create ingress firewall rules applicable to all VMs being load balanced to allow traffic from health check prober IP ranges.

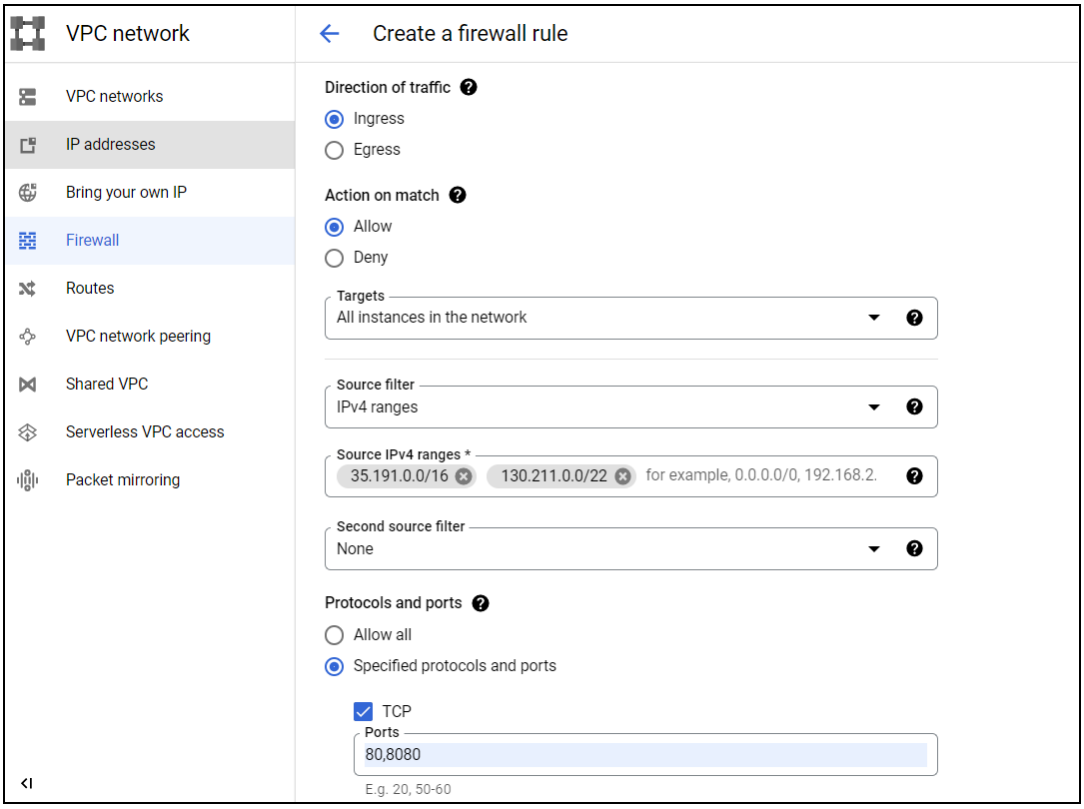
- In the Cloud Console, on the **Navigation menu** (☰), click **VPC network** > **Firewall.**
- To create a new Health check Firewall rule, click **CREATE FIREWALL RULE.**

There are many parameters you can configure when creating a new firewall rule

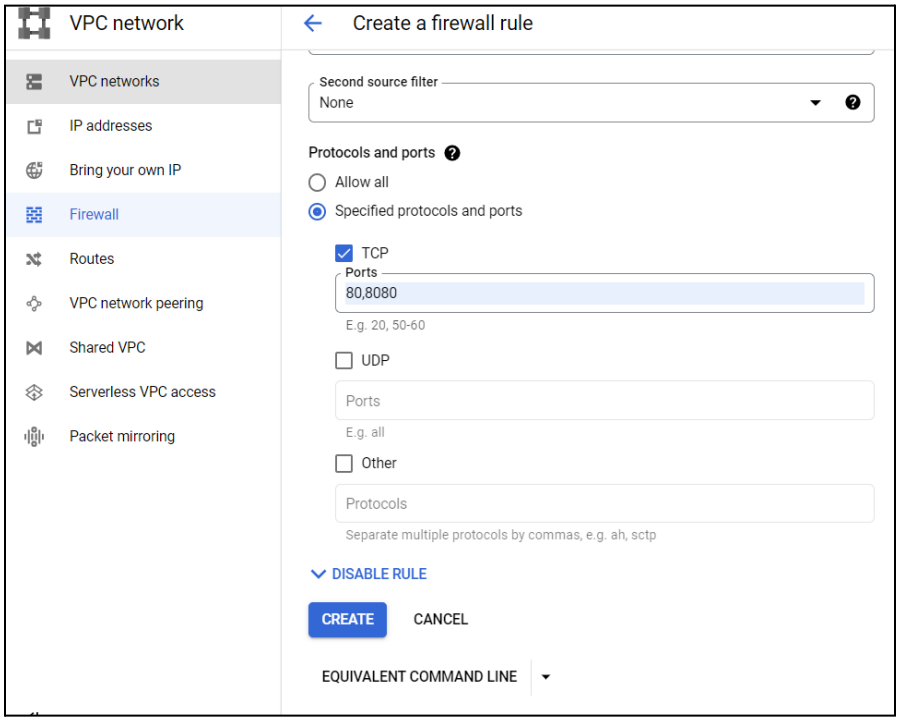
- Enter firewall **Name** as it is a mandatory field.
- Description can be left blank as it is an optional field.
- Select the VPC **Network** created previously i.e. *labs-vpc* to attach the firewall to that network.



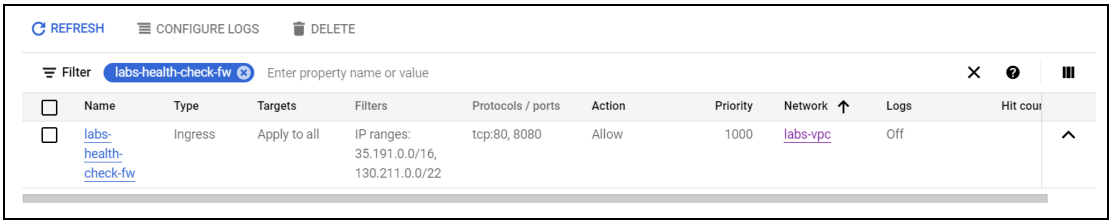
- Select Direction of traffic as **Ingress** by selecting the checkbox.
- Select Action on match as **Allow** by selecting the checkbox.
- In the Target field, select **All instances in the network**.
- In the Source filter, select IPv4 ranges.
- Enter the Source IPv4 range **35.191.0.0/16 & 130.211.0.0/22**.
- In Protocols and Ports, enable **TCP** protocol with port 80 for HTTP and 8080.



- Click on **CREATE** to create a Health check firewall.



- The IAP firewall named labs-health-check-fw is created.



3. Create Service Account for GKE

A service account is identified by its email address, which is unique to the account. Before creating the service account following the below steps

- Enable the IAM API - IAM API

Required roles for your IAM account.

To get the permissions that you need to manage service accounts, ask your administrator to grant you the following IAM roles on the project:

To view and create service accounts:

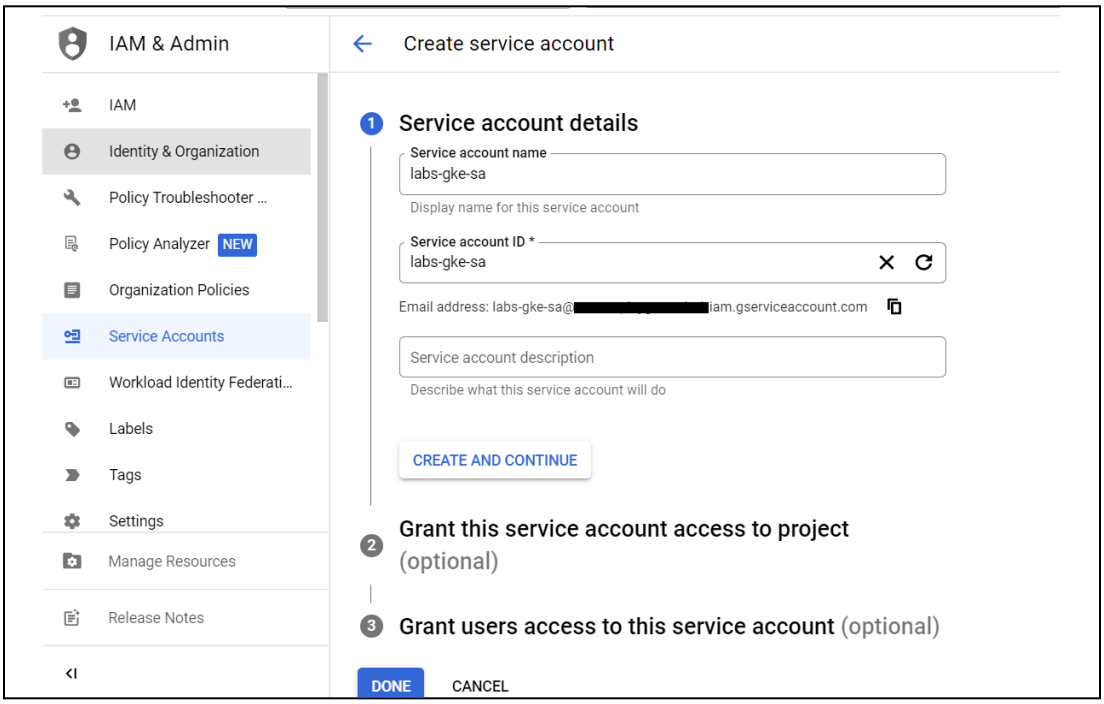
Create Service Accounts (roles/iam.serviceAccountCreator)

- In the Cloud Console, on the **Navigation menu** (☰), click **IAM & Admin** > **Firewall**.
- To create a new custom service account, click **CREATE SERVICE ACCOUNT**.

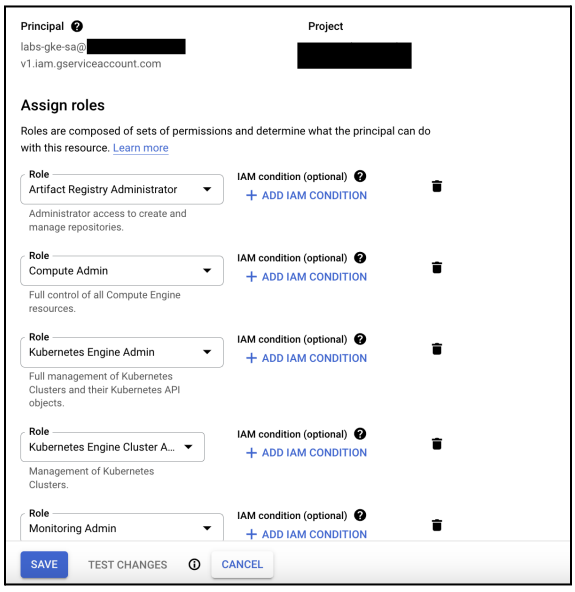
There are many parameters you can configure when creating a new firewall rule

- Enter **Service Account Name** as it is a mandatory field.
- Service account ID is auto populated with service account name.
- Description can be kept blank as it is an optional field.

- Click **CREATE AND CONTINUE** to create the custom service account.



- Other fields are optional and can be skipped. The custom service account name labs-gke-sa@project-id.iam.gserviceaccount.com is created
- Ask your administrator to grant **Storage Admin, Kubernetes Engine Cluster Admin, Kubernetes Engine Admin, Compute Admin and Artifact Admin** IAM roles on the created custom service account
- Navigate to IAM, click **Grant ACCESS**.
- Enter the created service account name in **New Principals**.
- In Role, Select **Storage Admin, Kubernetes Engine Cluster Admin and Artifact Admin** from the dropdown.

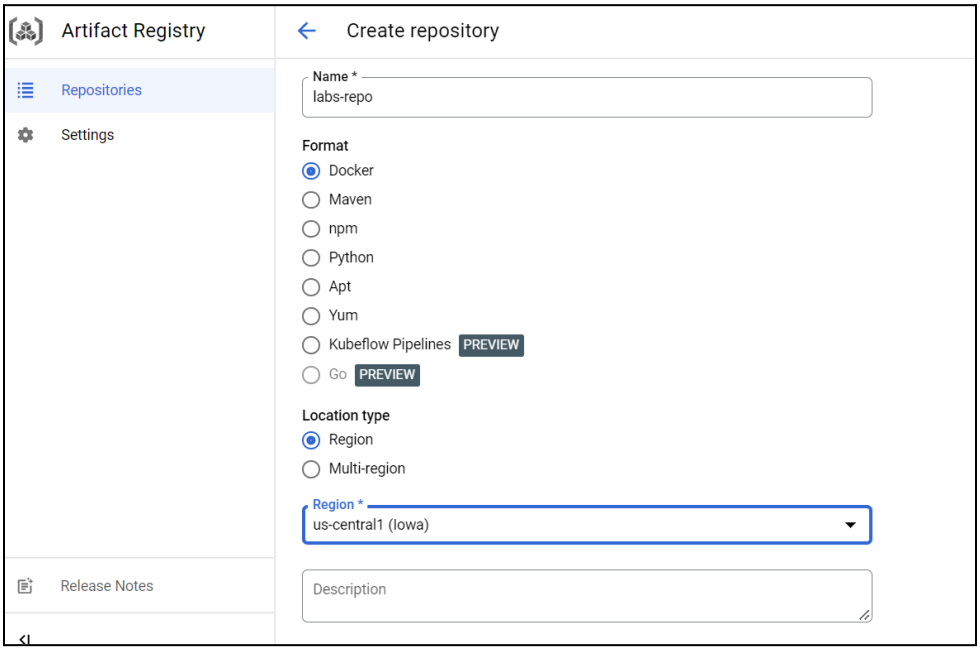


4. Create Artifact Repository

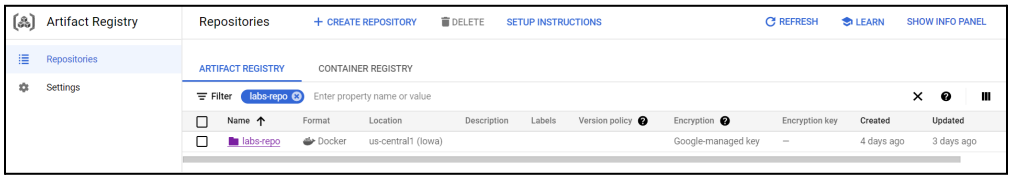
- In the Cloud Console, on the **Navigation menu** (☰), click **Artifact Registry** > **Repositories**.
- To create a new Artifact repository, click **CREATE ARTIFACT REPOSITORY**.

  There are many parameters you can configure when creating a new Artifact Repository.

- Enter **Repository Name** as it is a mandatory field.
- Select format as **DOCKER**.
- Select Location type as Region and from the region drop down select **us-central1**.
- Description can be kept blank as it is an optional field.
- For encryption, let it be default google managed encryption keys.
- Click **CREATE**, to create a new artifact repository.



- The repository is created named **labs-repo**.

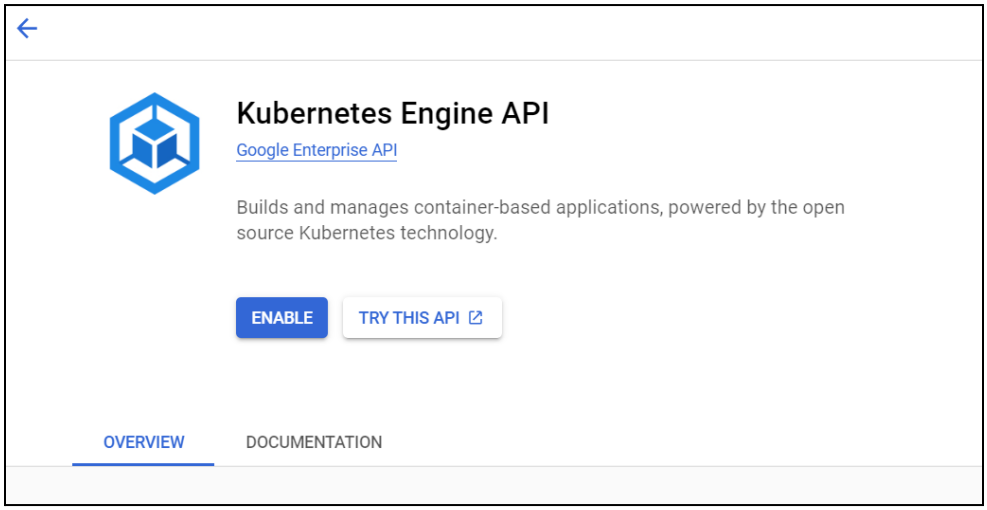


5. Install Docker

Install Docker based on your Operating System using this link.

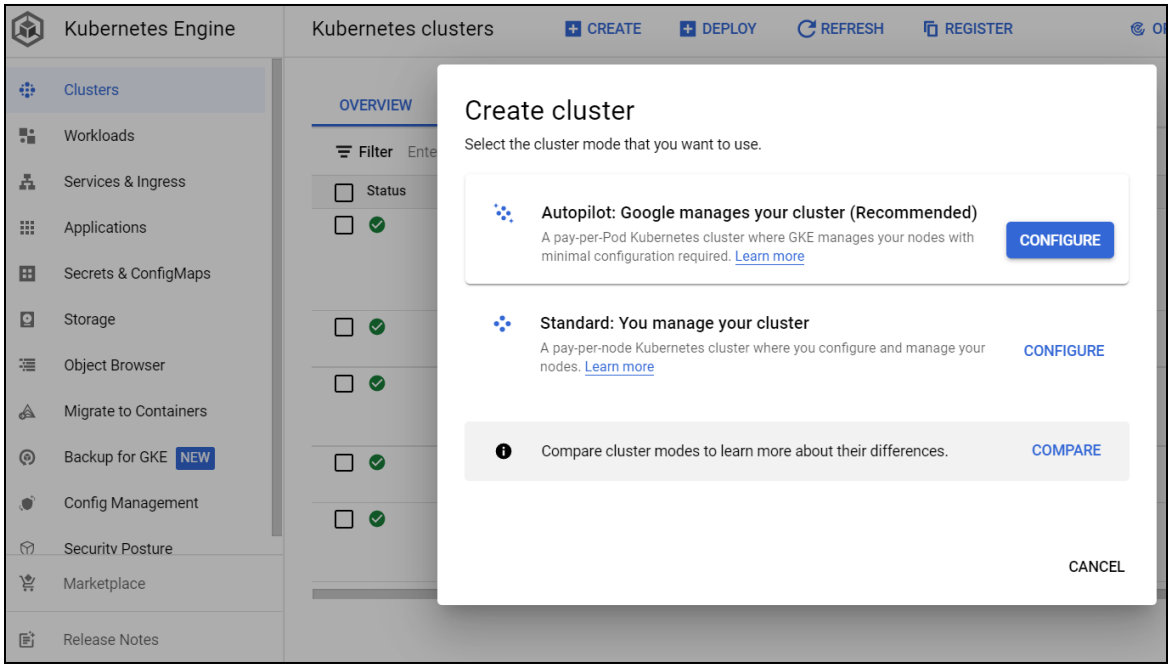
Setup for Google Kubernetes Engine

Enable the Kubernetes Engine API - [Kubernetes Engine API]

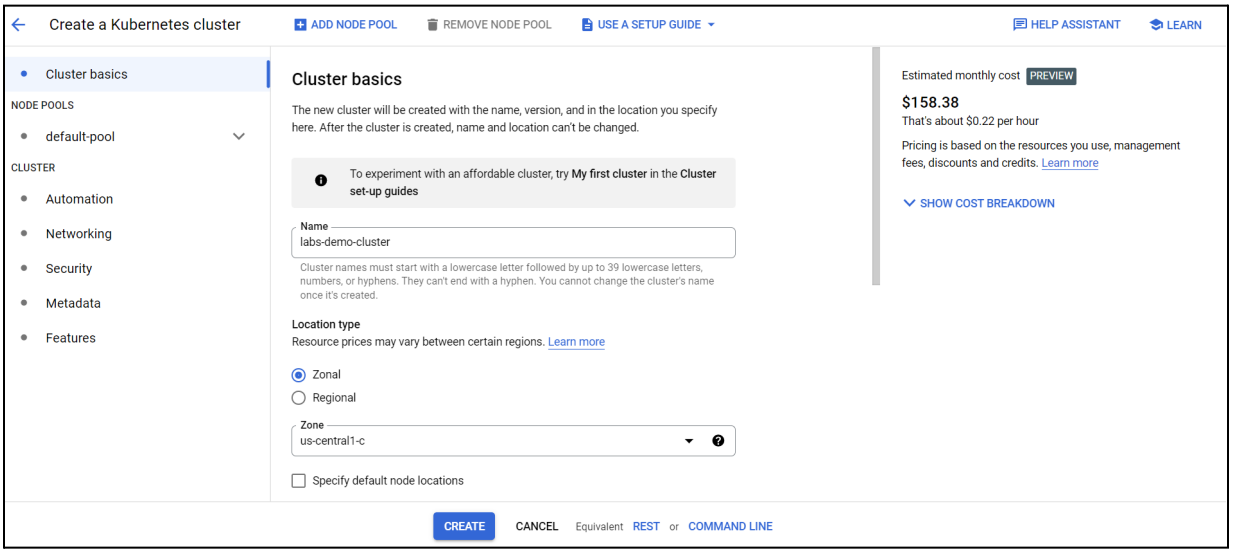


Create Google Kubernetes Engine Cluster

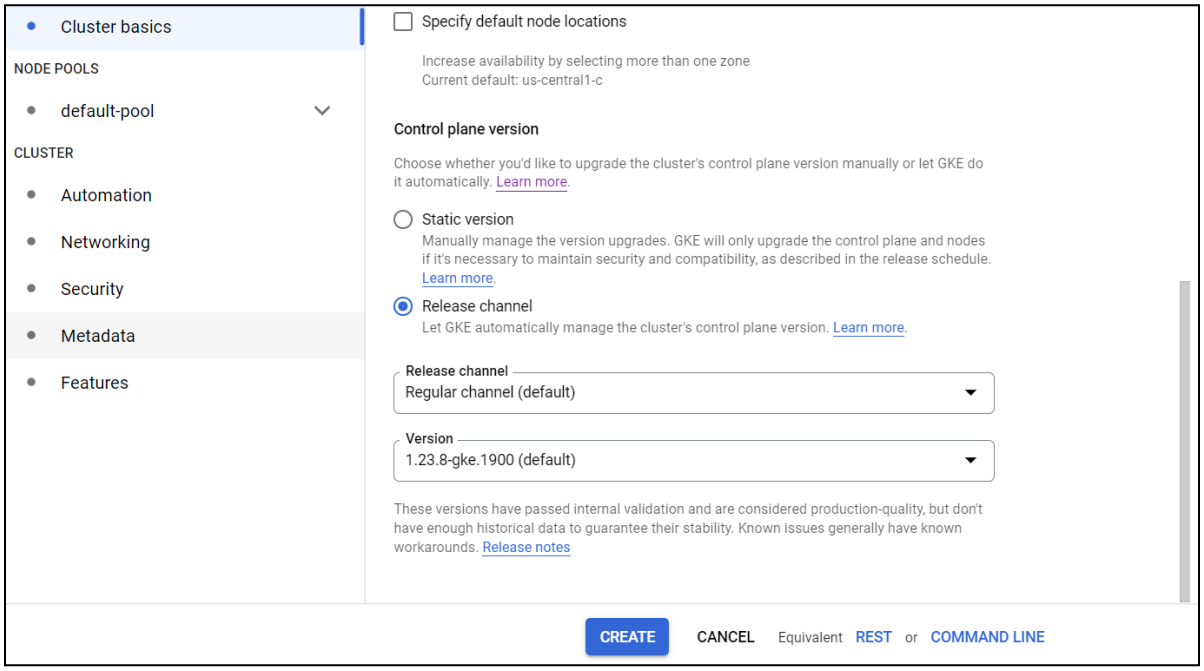
- In the Cloud Console, on the **Navigation menu** (☰), click **Kubernetes Engine** > **Cluster.**
- To create a new cluster, click **CREATE**.
- A popup window will appear to select the cluster mode, select **Standard** mode.



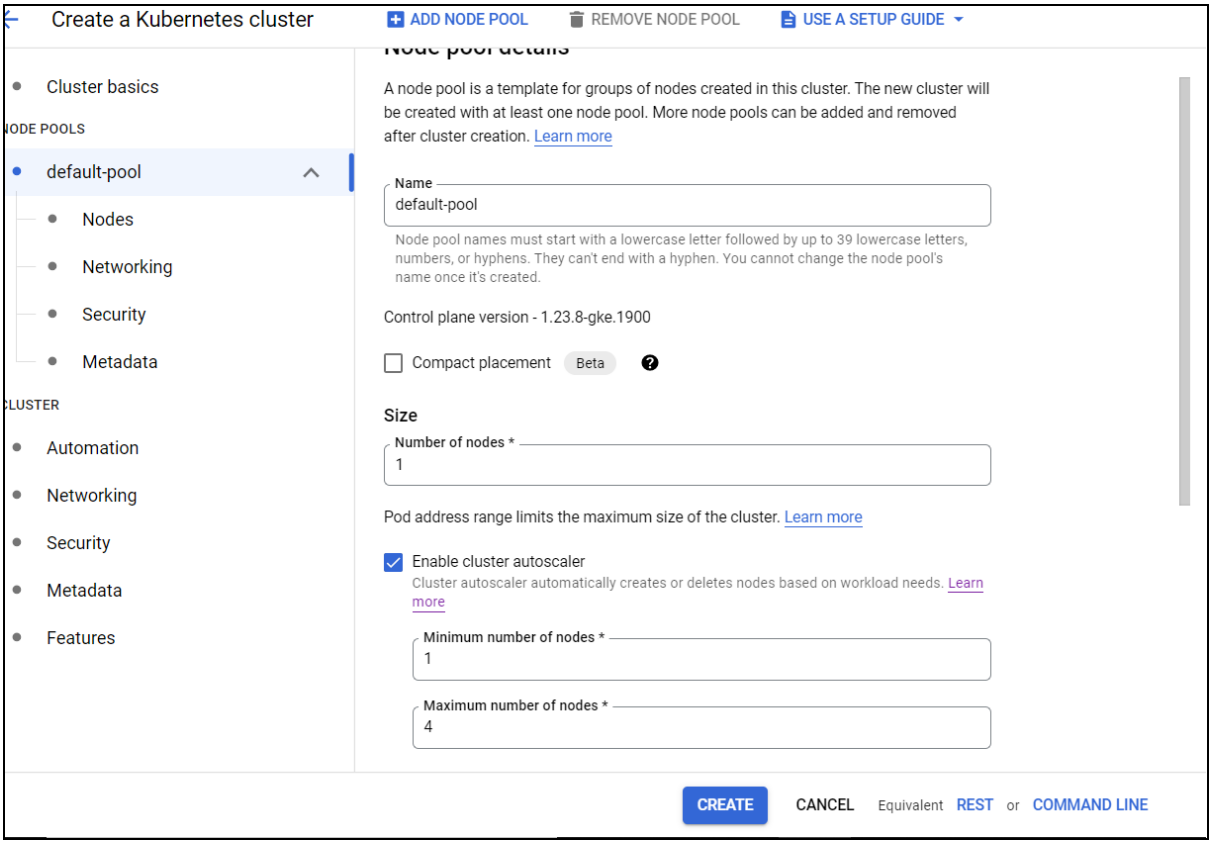
- On Cluster basics, Enter the **Name** of cluster. Select location type as **Zonal** and in zone select **us-central1** from dropdown.



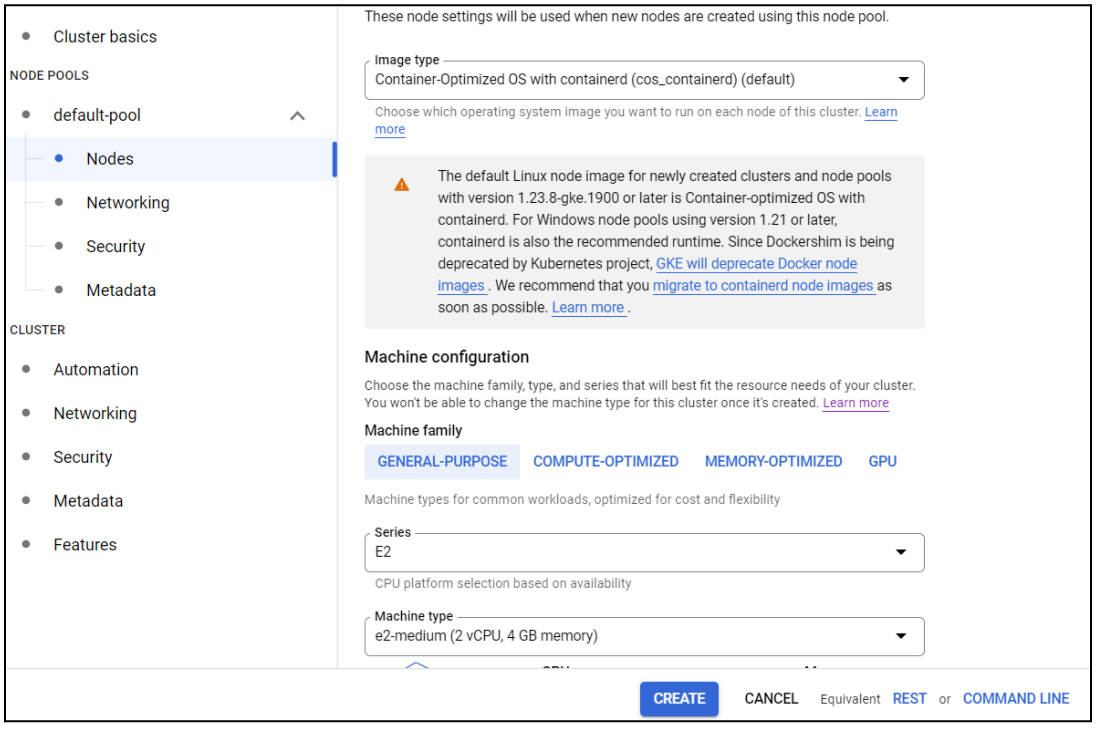
- Rest of the fields can be left with default values.



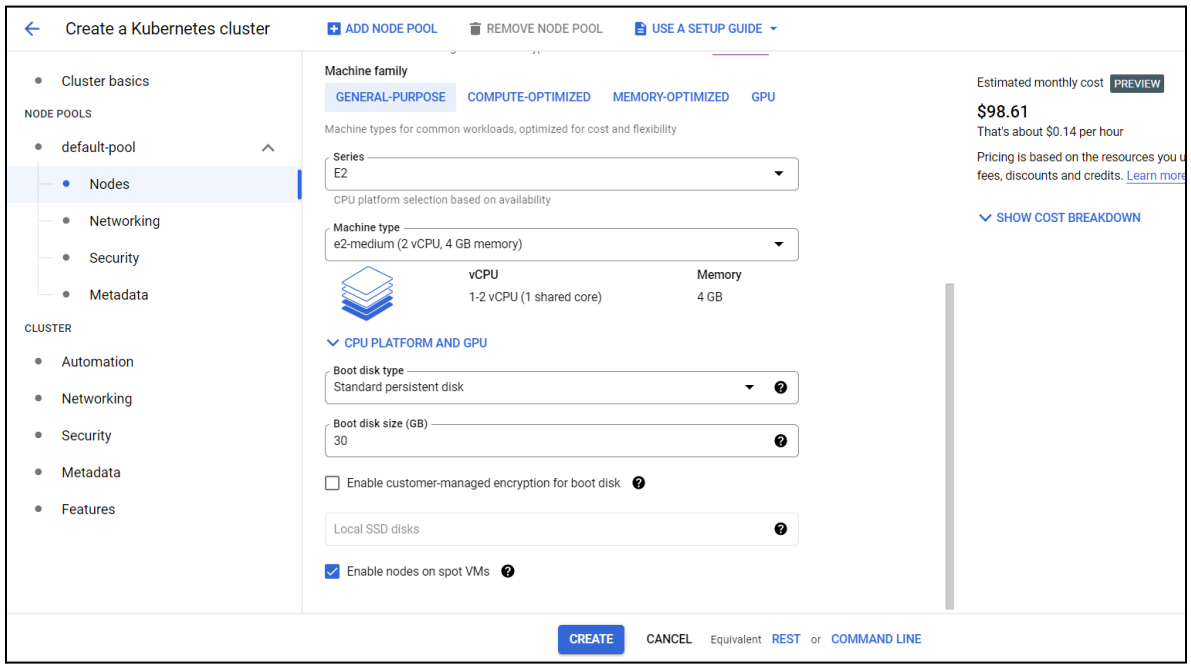
- Click on default-pool. In Size, Enter **Number of nodes** to **1**. Select the checkbox **Enable cluster autoscaler** for nodes autoscaling. Enter **minimum number of nodes** to **1** and **maximum number of nodes** to **4**.



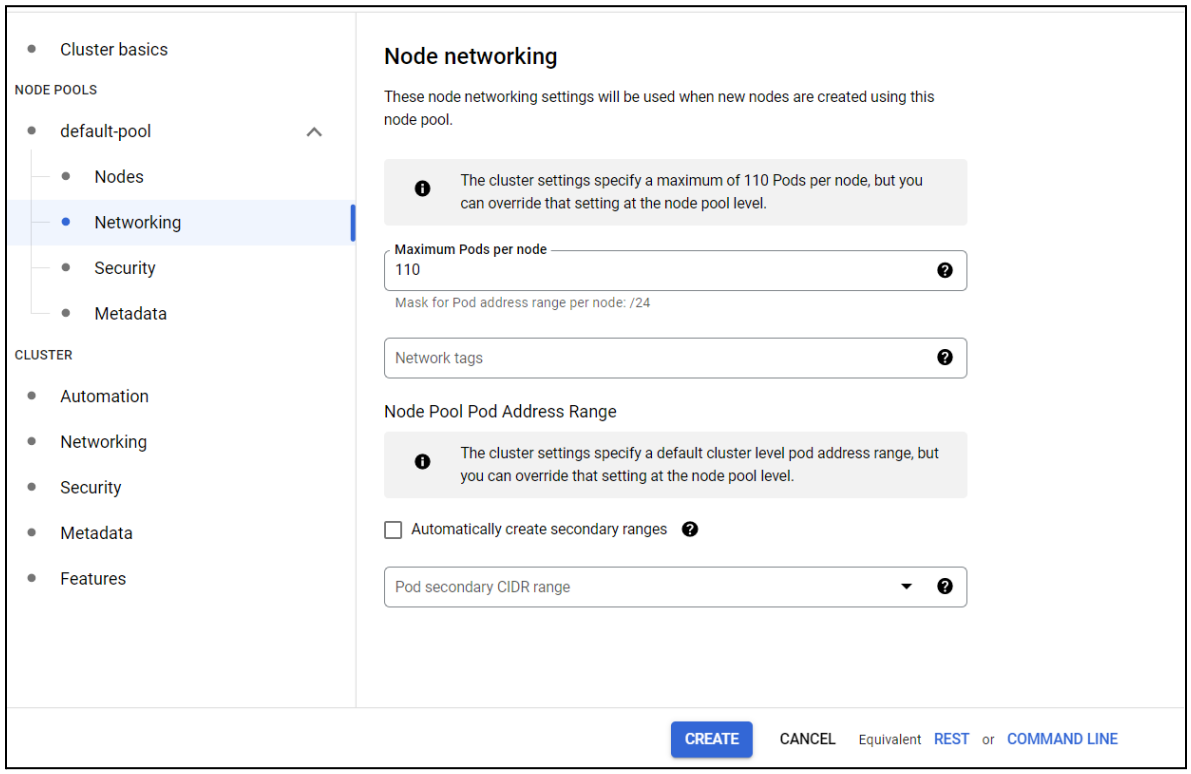
- In the default-pool sub section, Click on **Nodes**. Select the machine type for your cluster.



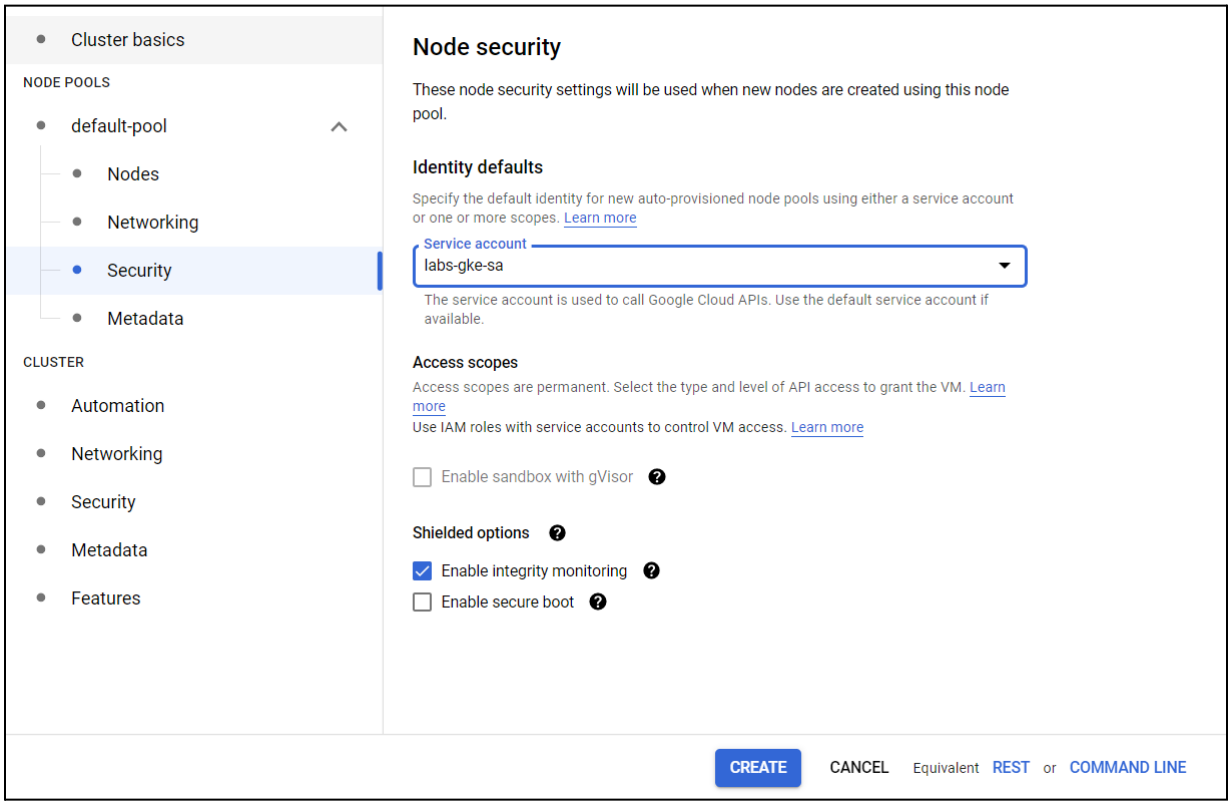
- Select **Series** as **E2** machine type as **e2-medium**. In Boot disk size enter **30 GB**. Boot disk size can be decided based on application requirements.
- Select the checkbox **Enable nodes on spot VMs**, spot VM's are available for only 24 hours post that they are terminated. By enabling the spot VM option, the total cost of the cluster is reduced from $158.94 to $98.61.



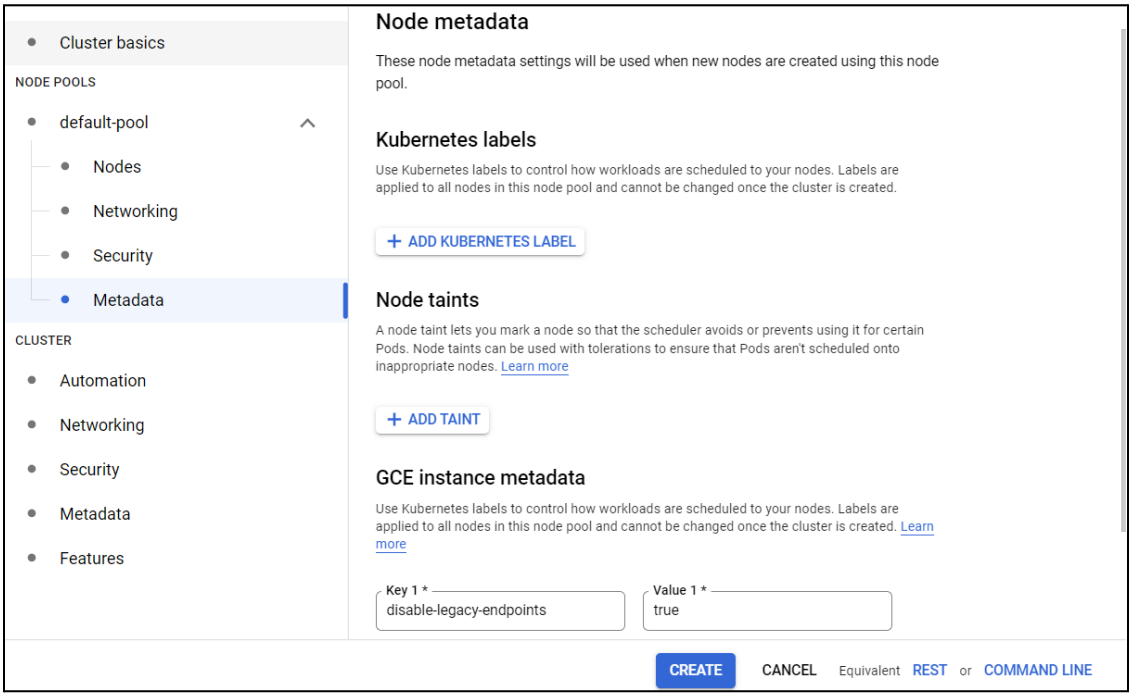
- In **Networking**, leave with default values.



- In **Security**, In Identity defaults select the service account created previously from the dropdown i.e. **labs-gke-sa**.

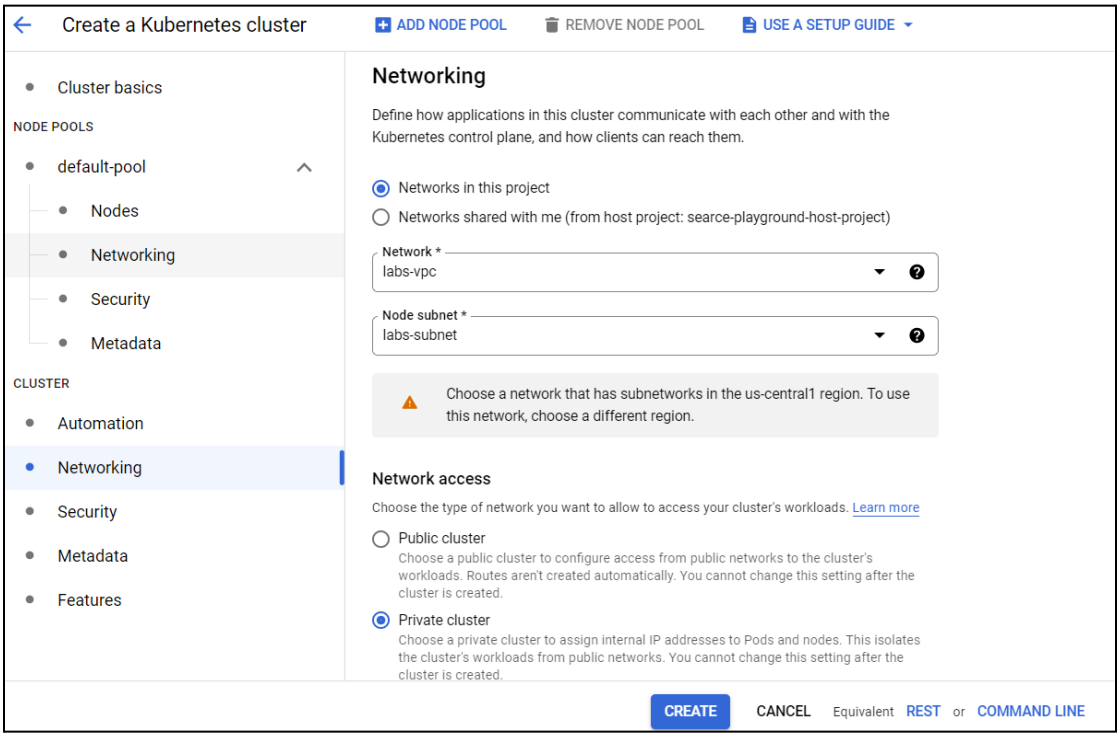


- In **Metadata**, leave with default values.

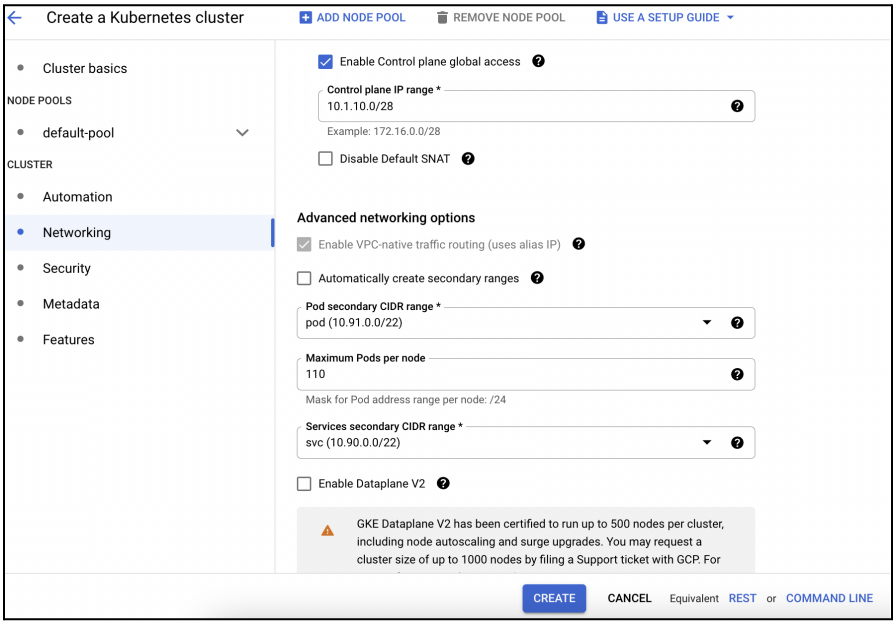


- In **Networking**, Select checkbox networks in this project. In **Network**, select the network created previously **labs-vpc** from the dropdown. In the Node subnet,

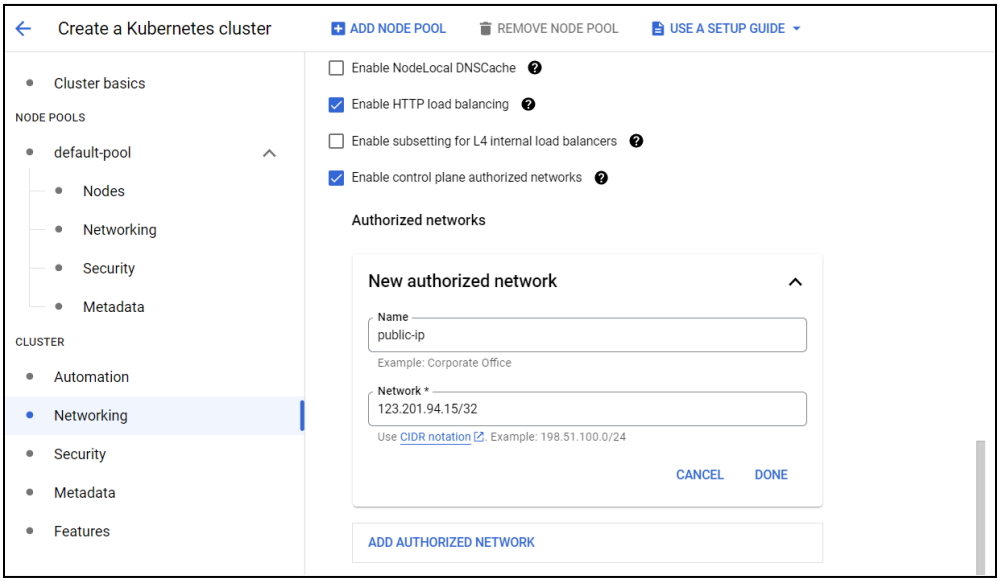
select the subnet created previously **labs-subnet**. In **Network access**, select the checkbox **Private Cluster**.



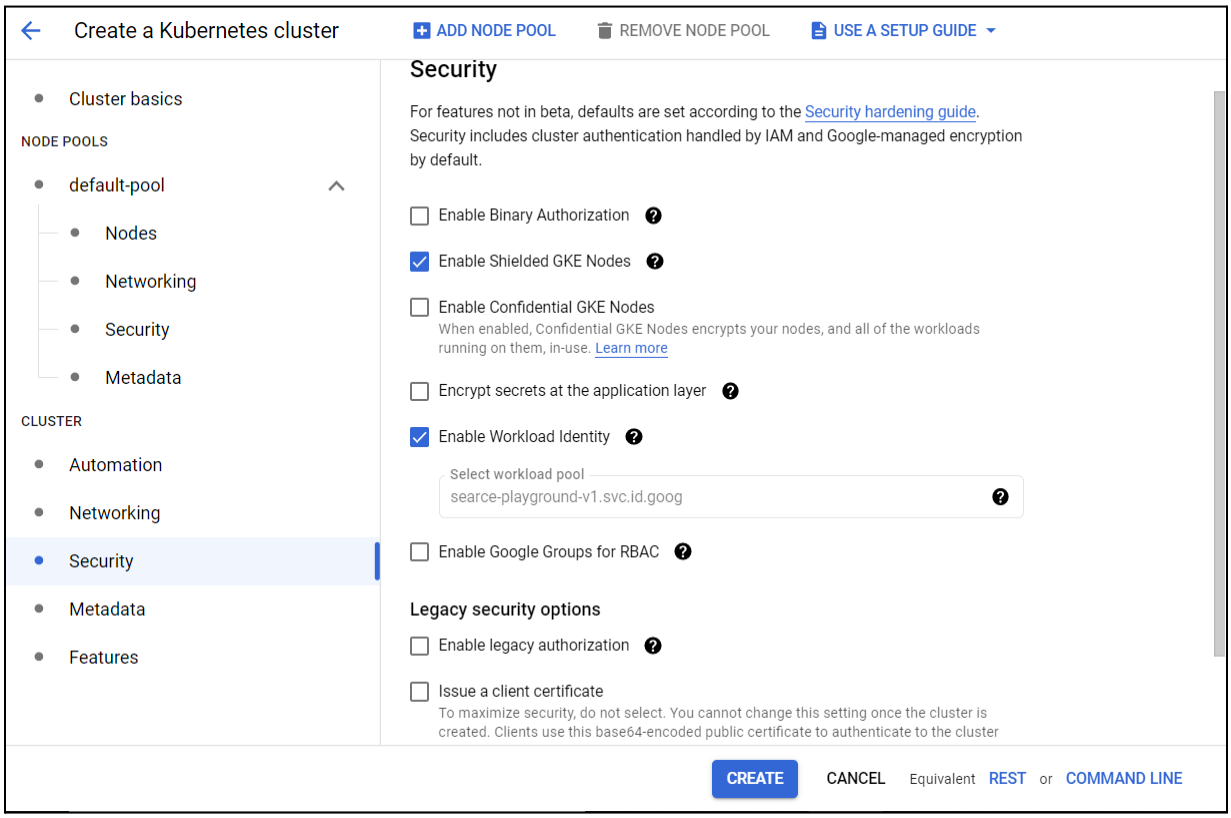
- Enable the checkbox**, Access control plane using its external IP address** and **Enable Control plane global access**. In control plane IP range enter **10.1.10.0/28**. Disable checkbox **Automatically create Secondary ranges** & select pod and service. Rest of the fields are auto populated.



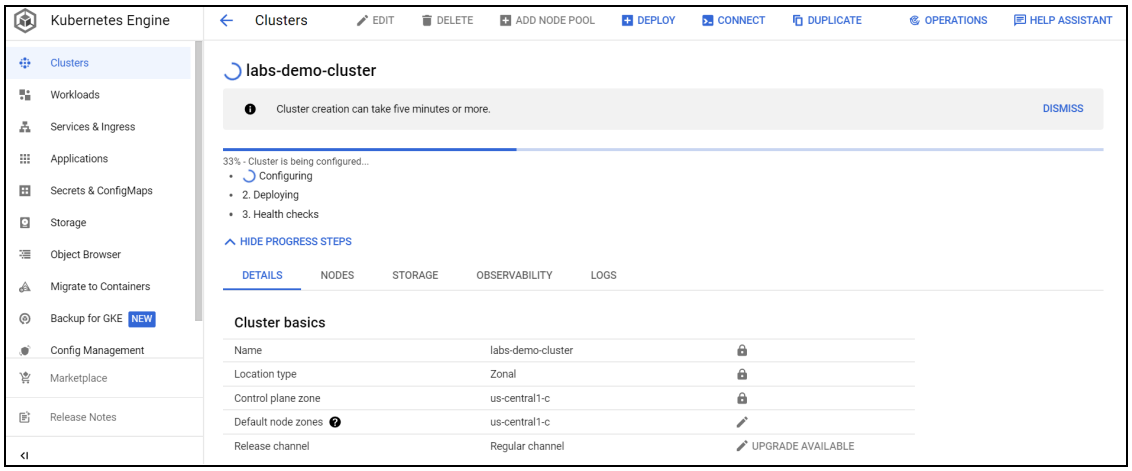
- Select the checkbox, enable **control plane authorized network** to connect to your cluster from local machine or vm. In Authorized Networks, Enter **Name** and In Network enter your **public-ip/32**. Click on **DONE**.

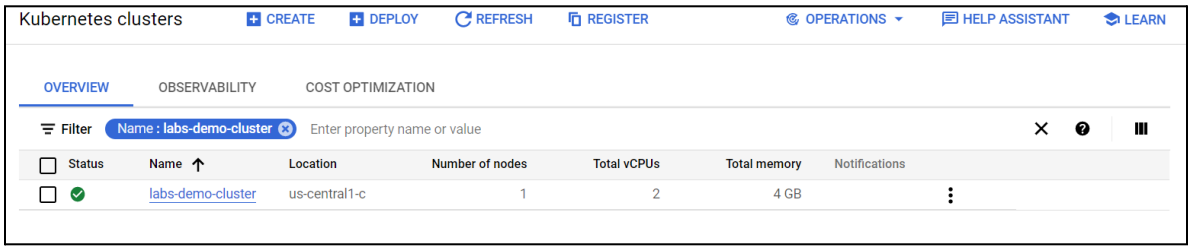


- In Security, Select the checkbox **Enable Workload Identity**.

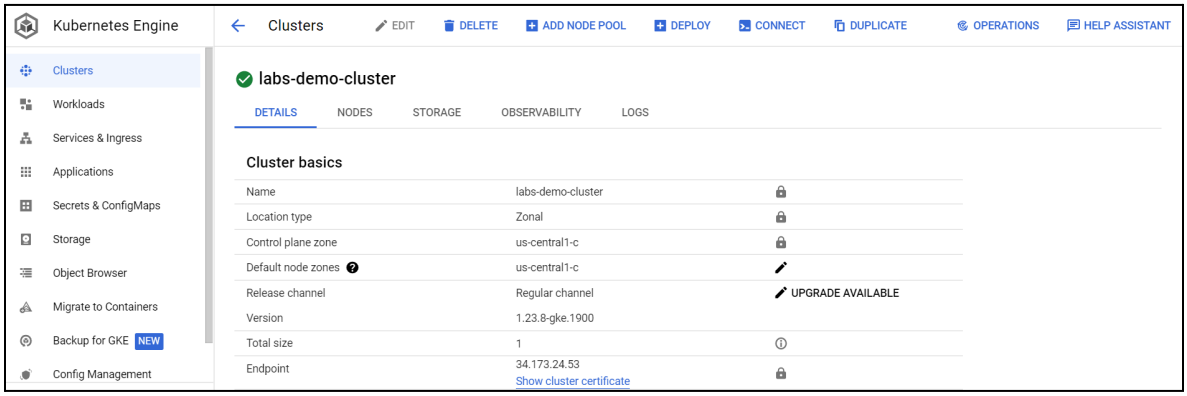


- Metadata and Features, leave with default values.
- Cluster creation may take from 5-10 mins. Cluster named labs-demo-cluster is created successfully.

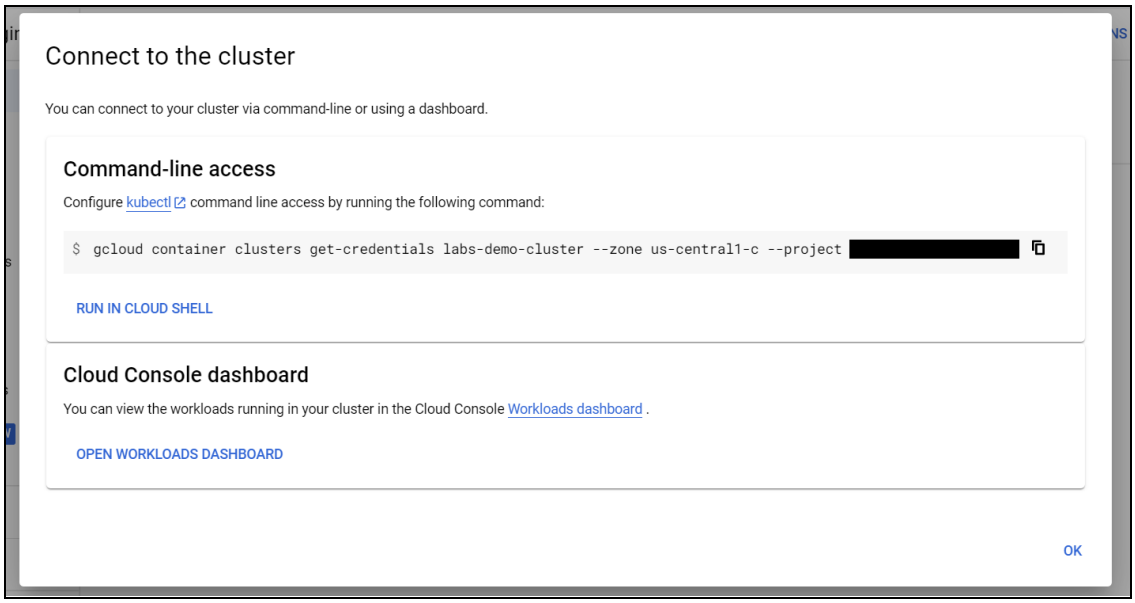




- To connect to cluster, click on cluster name, Click on **CONNECT**



- A popup will appear with the gcloud command to connect to your cluster. Copy the command.



- SSH into your VM created previously (in Lab1), since cluster and VM are in the same network, VM can connect with an internal ip address. Use the above copied command with **--internal-ip** at the end. Install kubectl on the VM. Follow the documentation - kubectl. Post installation of kubectl run command **kubectl get nodes** to verify nodes details.

```
root@labs-demo-vm:~# gcloud container clusters get-credentials labs-demo-cluster --zone us-central1-c --project
    <project-id>          --internal-ip
Fetching cluster endpoint and auth data.
CRITICAL: ACTION REQUIRED: gke-gcloud-auth-plugin, which is needed for continued use of kubectl, was not found o
r is not executable. Install gke-gcloud-auth-plugin for use with kubectl by following https://cloud.google.com/b
log/products/containers-kubernetes/kubectl-auth-changes-in-gke
kubeconfig entry generated for labs-demo-cluster.
root@labs-demo-vm:~# kubectl get nodes
W1114 09:41:40.595756    2558 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                                                STATUS   ROLES    AGE   VERSION
gke-labs-demo-cluster-default-pool-5814bf37-9tfw   Ready    <none>   20m   v1.23.8-gke.1900
root@labs-demo-vm:~#
```

Build Docker Image and Push it to Artifact Repository

- So far, we are able to connect to Cluster. Now it's time to start the deployment of the application and validate node & pod autoscaling.
- Login to Github and clone the following repository using command:
  git clone https://github.com/doketanmay98/labs-sample-app.git
- Folder named labs-sample-app will be created in the present working directory(PWD).
- The repository consists of C++ application codes and Dockerfile to containerize the application.

```
root@labs-demo-vm:~# ll | grep labs
drwxr-xr-x  6 root root     4096 Nov 10 21:10 labs-sample-app/
root@labs-demo-vm:~#
```

- Change directory to Web-application using cd **labs-sample-app/Web-application**. This folder contains C++ web applications and Dockerfile.

```
root@labs-demo-vm:~/labs-sample-app/Web-application# ls
Dockerfile  WebApp
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

- Create a docker image locally using the following command

cd labs-sample-app/Web-application/

docker build -t web-app .

docker images

```
root@labs-demo-vm:~/labs-sample-app/Web-application# docker build -t web-app .
Sending build context to Docker daemon  18.94kB
Step 1/7 : FROM amytabb/docker_ubuntu16_essentials
latest: Pulling from amytabb/docker_ubuntu16_essentials
8ee29e426c26: Already exists
6e83b260b73b: Already exists
e26b65fd1143: Already exists
40dca07f8222: Already exists
b420ae9e10b3: Already exists
dfaf13193fe3: Already exists
Digest: sha256:aa1aee49d49b6d641a81afeedb252befe55fbbce6a6badc50e61cf649dc9e8e6
Status: Downloaded newer image for amytabb/docker_ubuntu16_essentials:latest
 ---> 13f6d277d91b
Step 2/7 : COPY WebApp /WebApp
 ---> 59a40c51e465
Step 3/7 : WORKDIR /WebApp/
 ---> Running in e743d9d5cf88
Removing intermediate container e743d9d5cf88
 ---> b734564ff59e
Step 4/7 : RUN g++ -o WebApp WebApp.c
 ---> Running in 96e053187698
Removing intermediate container 96e053187698
 ---> e85a58f61128
Step 5/7 : CMD ["./WebApp"]
 ---> Running in 65430e7cdd47
Removing intermediate container 65430e7cdd47
 ---> 10f0b6358638
Step 6/7 : FROM nginx
latest: Pulling from library/nginx
e9995326b091: Already exists
71689475aec2: Already exists
f88a23025338: Already exists
0df440342e26: Already exists
eef26ceb3309: Already exists
8e3ed6a9e43a: Already exists
Digest: sha256:943c25b4b66b332184d5ba6bb18234273551593016c0e0ae906bab111548239f
Status: Downloaded newer image for nginx:latest
 ---> 76c69feac34e
Step 7/7 : COPY WebApp/index.html /usr/share/nginx/html/index.html
 ---> d788554c740f
Successfully built d788554c740f
Successfully tagged web-app:latest
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

```
root@labs-demo-vm:~/labs-sample-app/Web-application# docker images
REPOSITORY                              TAG       IMAGE ID       CREATED              SIZE
web-app                                 latest    d788554c740f   About a minute ago   142MB
<none>                                  <none>    10f0b6358638   About a minute ago   807MB
nginx                                   latest    76c69feac34e   2 weeks ago          142MB
amytabb/docker_ubuntu16_essentials      latest    13f6d277d91b   4 years ago          807MB
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

- Run locally to validate the image working using the following command.

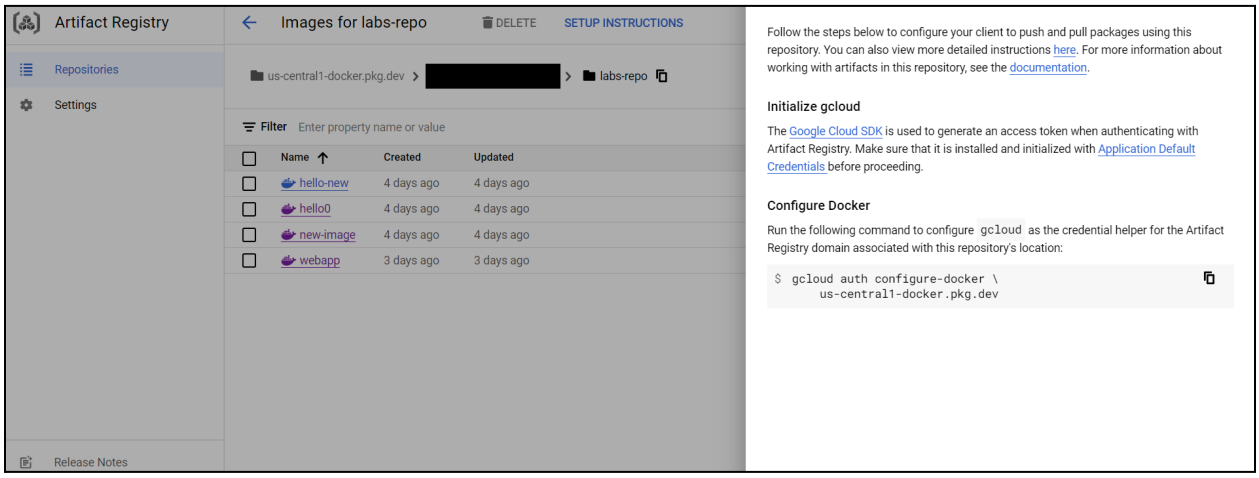
docker run -itd -p 8080:80 web-app:latest

curl localhost:8080

```
root@labs-demo-vm:~/labs-sample-app/Web-application# docker run -itd -p 8080:80 web-app:latest
57f21d35f11169b4f983d6ab6ae71fe729a661ddcccf2e1ef0cb6b2390bb81b9
root@labs-demo-vm:~/labs-sample-app/Web-application# docker ps -a
CONTAINER ID   IMAGE           COMMAND                  CREATED          STATUS         PORTS
                 NAMES
57f21d35f111   web-app:latest   "/docker-entrypoint.…"   4 seconds ago    Up 3 seconds   0.0.0.0:8080->80/tcp, :
::8080->80/tcp   nervous_booth
2d0e44020ad4   web-app:latest   "/docker-entrypoint.…"   13 seconds ago   Created
                 happy_williamson
root@labs-demo-vm:~/labs-sample-app/Web-application# curl localhost:8080
Files can be tricky, but it is fun enough!root@labs-demo-vm:~/labs-sample-app/Web-application#
```

- Authorize with artifact credentials. Navigate to your Artifact Repository. Click on **SETUP INSTRUCTIONS**.

**gcloud auth configure-docker **
us-central1-docker.pkg.dev




```
root@labs-demo-vm:~/labs-sample-app/Web-application# gcloud auth configure-docker \
>     us-central1-docker.pkg.dev
WARNING: Your config file at [/root/.docker/config.json] contains these credential helper entries:

{
  "credHelpers": {
    "us-central1-docker.pkg.dev": "gcloud"
  }
}
Adding credentials for: us-central1-docker.pkg.dev
gcloud credential helpers already registered correctly.
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

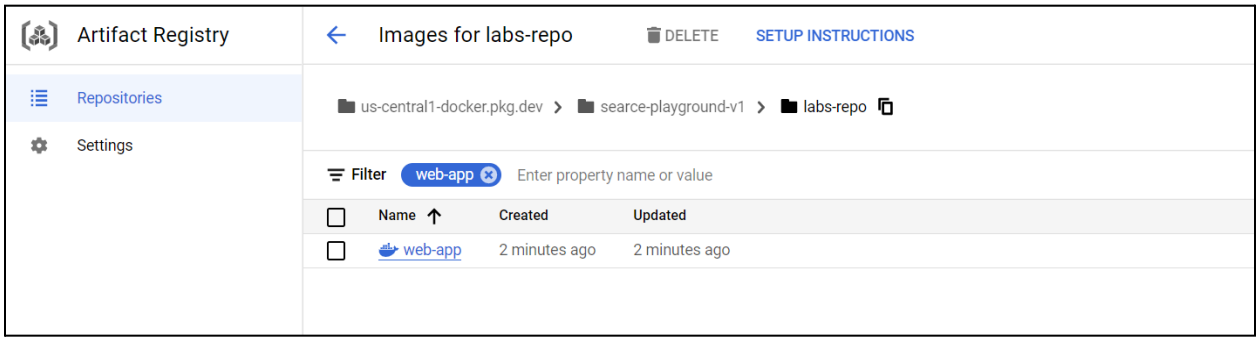
- Tag the locally built image with artifact details using the following command.
  docker tag web-app us-central1-docker.pkg.dev/<project-id>/labs-repo/web-app

```
root@labs-demo-vm:~/labs-sample-app/Web-application# docker images
REPOSITORY                          TAG       IMAGE ID       CREATED          SIZE
web-app                             latest    d788554c740f   49 minutes ago   142MB
<none>                              <none>    10f0b6358638   49 minutes ago   807MB
nginx                               latest    76c69feac34e   2 weeks ago      142MB
amytabb/docker_ubuntu16_essentials  latest    13f6d277d91b   4 years ago      807MB
root@labs-demo-vm:~/labs-sample-app/Web-application# docker tag web-app us-central1-docker.pkg.dev/searce-playground-v1/labs-repo/web-app
root@labs-demo-vm:~/labs-sample-app/Web-application# docker images
REPOSITORY                                                      TAG       IMAGE ID       CREATED          SIZE
web-app                                                         latest    d788554c740f   52 minutes ago   142MB
us-central1-docker.pkg.dev/searce-playground-v1/labs-repo/web-app   latest    d788554c740f   52 minutes ago   142MB
<none>                                                          <none>    10f0b6358638   52 minutes ago   807MB
nginx                                                           latest    76c69feac34e   2 weeks ago      142MB
amytabb/docker_ubuntu16_essentials                              latest    13f6d277d91b   4 years ago      807MB
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

- Push the image to the Artifact repository.
  docker push us-central1-docker.pkg.dev/<project-id>/labs-repo/web-app:latest

```
root@labs-demo-vm:~/labs-sample-app/Web-application# docker push us-central1-docker.pkg.dev/searce-playground-v1/labs-repo/web-ap
Using default tag: latest
The push refers to repository [us-central1-docker.pkg.dev/searce-playground-v1/labs-repo/web-app]
55fdad469d63: Pushed
a2e59a79fae0: Layer already exists
4091cd312f19: Layer already exists
9e7119c28877: Layer already exists
2280b348f4d6: Layer already exists
e74d0d8d2def: Layer already exists
a12586ed027f: Layer already exists
latest: digest: sha256:0d3b243aa1864c97f40f15b705194d42a597d3e22d54315c3750d34ccece3caa size: 1777
root@labs-demo-vm:~/labs-sample-app/Web-application#
```

- Verify the pushed image in the artifact repository.



Deploying C++ Web Application on GKE Cluster

- Change directory to gke using cd **/labs-sample-app/gke**. This folder contains Kubernetes deployment, service and Horizontal Pod Autoscaling manifest files.
  cd /labs-sample-app/gke.
- Edit the deployment.yaml file. Insert your artifact repository url in the **image** parameter to pull the image.
  vi deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp
spec:
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      containers:
      - name: myapp
        image: us-central1-docker.pkg.dev/ <project-id> /labs-repo/web-app:latest
        resources:
          requests:
            memory: "128Mi"
            cpu: "100m"
          limits:
            memory: "128Mi"
            cpu: "100m"
        ports:
        - containerPort: 80
```

Press **escape** and type **:wq** to save the changes.

- Deploy the application using the following command

```
#deploy the application
kubectl apply -f deployment.yaml

#check the status of pods
kubectl get pods
```

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f deployment.yaml
W1114 11:13:15.846520    4287 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
deployment.apps/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get pods
W1114 11:13:27.270393    4308 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                     READY   STATUS              RESTARTS   AGE
myapp-6df67d5d8b-2wqlb   0/1     ContainerCreating   0          7s
root@labs-demo-vm:~/labs-sample-app/gke#
```

Wait till Pod status becomes Running.

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get pods
W1114 11:14:31.273515    4317 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                     READY   STATUS    RESTARTS   AGE
myapp-6df67d5d8b-2wqlb   1/1     Running   0          71s
root@labs-demo-vm:~/labs-sample-app/gke#
```

- Expose the service as a load balancer service to view the application.

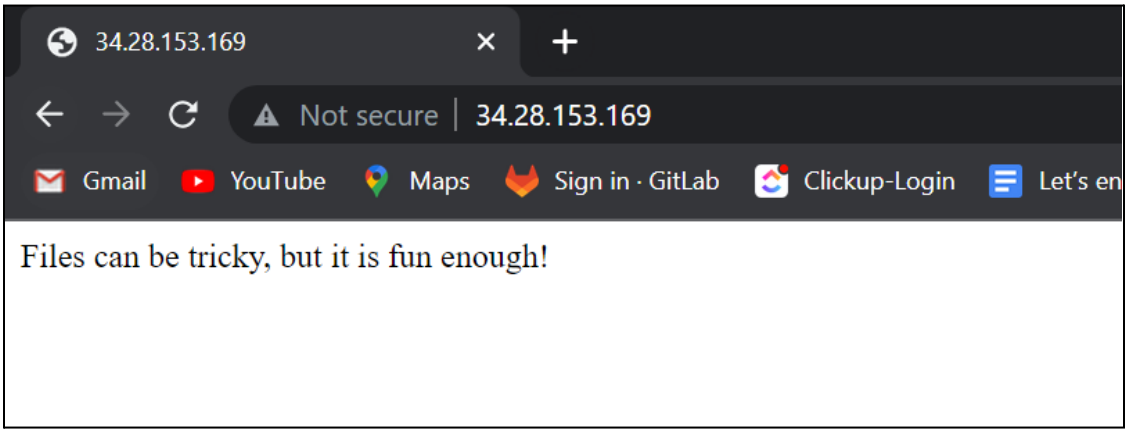
```
#deploy load balancer service
kubectl apply -f service.yaml

#check the status of service
kubectl get svc
```

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f service.yaml
W1114 11:16:57.659652    4336 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
service/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get svc
W1114 11:17:10.501661    4344 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP      10.91.0.1     <none>        443/TCP        118m
myapp        LoadBalancer   10.91.3.17    <pending>     80:30170/TCP   12s
root@labs-demo-vm:~/labs-sample-app/gke#
```

Service will create an external load balancer, wait till it assigns a public ip.

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get svc
W1114 11:17:47.163593    4352 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP     PORT(S)        AGE
kubernetes   ClusterIP      10.91.0.1     <none>          443/TCP        118m
myapp        LoadBalancer   10.91.3.17    34.28.153.169   80:30170/TCP   49s
root@labs-demo-vm:~/labs-sample-app/gke#
```

Once the external-ip is assigned to service, use the external ip and open browser to view the application. Application is successfully deployed on GKE.



AutoScaling in Google Kubernetes Engine

Overview

When demand is high, the cluster autoscaler adds nodes to the node pool. When demand is low, the cluster autoscaler scales back down to a minimum size that you designate. This can increase the availability of your workloads when you need it, while controlling costs.

Cluster autoscaler increases or decreases the size of the node pool automatically by adding or removing virtual machine (VM) instances in the underlying Compute Engine Managed Instance Group (MIG) for the node pool.

When you first deploy your workload to a Kubernetes cluster, you may not be sure about its resource requirements and how those requirements might change depending on usage patterns, external dependencies, or other factors. Horizontal Pod autoscaling

helps to ensure that your workload functions consistently in different situations, and allows you to control costs by only paying for extra capacity when you need it.

Setup for Autoscaling

- The previously created cluster already has Cluster autoscaler enabled to automatically scale the number of nodes.
- For Pods Autoscaling, Horizontal Pod Autoscaler Kubernetes object manifest file needs to be deployed.
- Change directory to gke using cd **/labs-sample-app/gke**. This folder contains Kubernetes deployment, service and Horizontal Pod Autoscaling manifest files.

  cd /labs-sample-app/gke.
- Deploy the HPA using the following command

```
#deploy the HPA object
kubectl apply -f hpa.yaml

#check the status of hpa
kubectl get hpa
```

The following HPA deployment scales pods from 1 to 6 if CPU utilization goes above 5%. If load on application is greater than the threshold value, it triggers pods to autoscale.

```
root@labs-demo-vm:~/labs-sample-app/gke# cat hpa.yaml
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: myapp
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: myapp
  minReplicas: 1
  maxReplicas: 6
  targetCPUUtilizationPercentage: 5
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f hpa.yaml
W1114 11:35:12.088688    4435 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
horizontalpodautoscaler.autoscaling/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get hpa
W1114 11:35:16.131737    4440 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME    REFERENCE          TARGETS       MINPODS   MAXPODS   REPLICAS   AGE
myapp   Deployment/myapp   <unknown>/5%   1         6         0          4s
root@labs-demo-vm:~/labs-sample-app/gke#
```

- To validate the load testing install the Apache Benchmark tool. Follow the documentation - Apache Benchmark.

```
root@labs-demo-vm:~/labs-sample-app/gke# ab -V
This is ApacheBench, Version 2.3 <$Revision: 1807734 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

root@labs-demo-vm:~/labs-sample-app/gke#
```

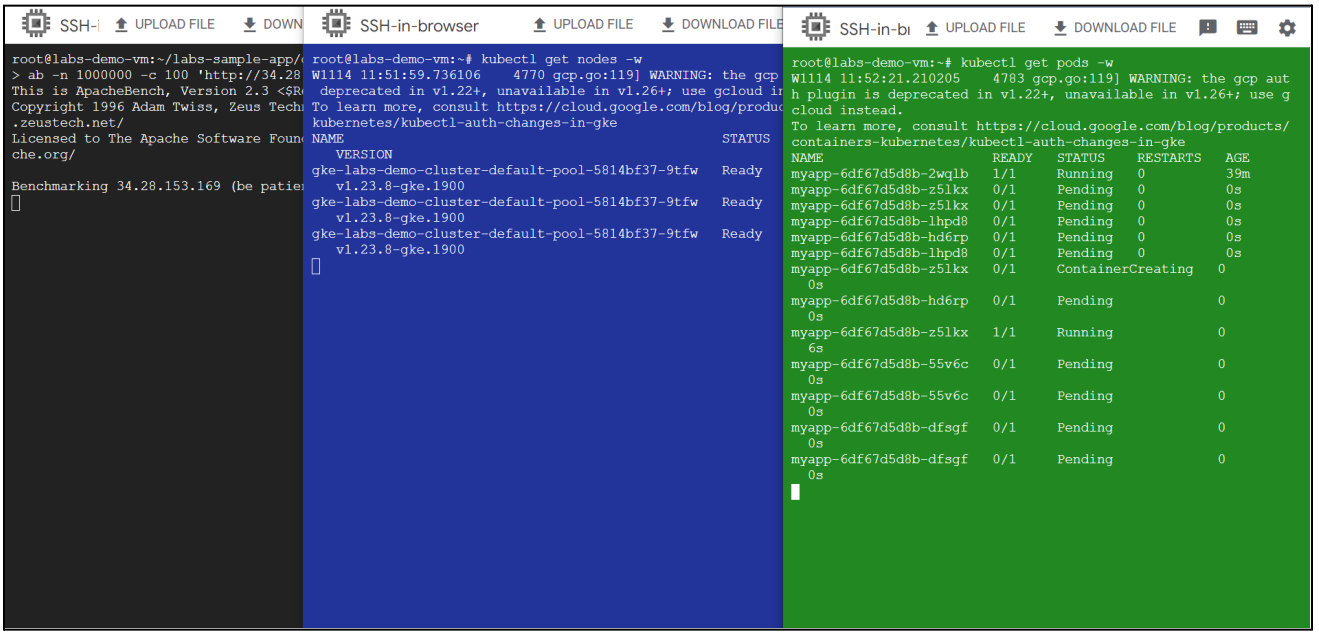
- Open 2 more tabs of VM, one to view pod autoscaling and other to view node autoscaling.
  Before autoscaling: Watch the nodes & Nodes
  Terminal 2: kubectl get nodes -w
  Terminal 3: kubectl get pods -w

  For autoscaling : Get the external IP of your service. Replace it with your service IP address

  Terminal 1 (In black): Enter **ab -n 10000 -c 100 'http://<svc-external-ip>/'**
  -n = number of request
  -c = number of concurrent request

  Terminal 2 (In blue): View Node autoscaling
  The number of nodes autoscaled from 1 to 3. Maximum number of nodes will be autoscaled is 4 depending on the traffic/load.

  Terminal 3 (In green): View Pod Autoscaling
  The number of pods autoscaled from 1 to 6. Many pods will get created but only 6 will be in active/running state as per the HPA definitions.



  Post Autoscaling:
  Nodes and Pods count will decrease to its initial value pre autoscaling.

AutoScaling in Google Kubernetes Engine based on Pub/Sub Triggers

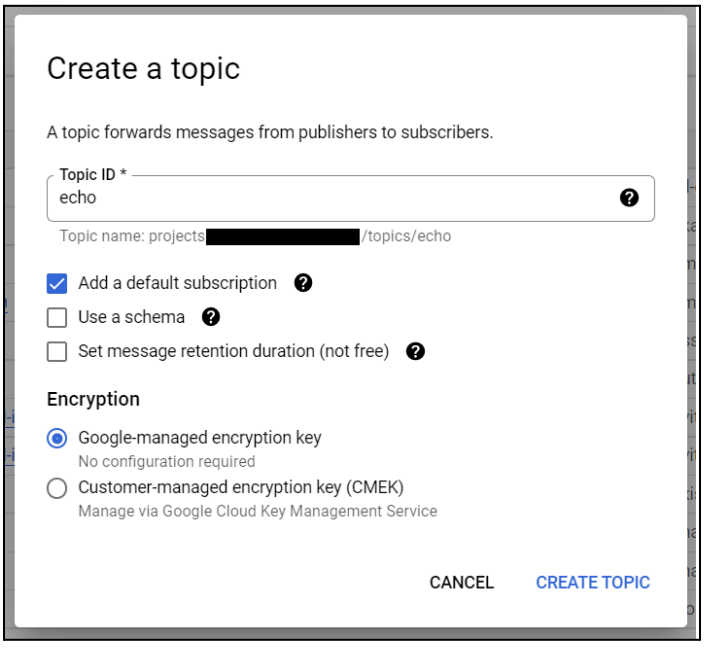
Overview

There are certain use cases where scaling horizontally based on cpu usage does not really work well. Let's say you have a consumer worker pool running in Kubernetes. The consumers are pulling messages from a PubSub topic. When the queue is filling up we want more workers to process the messages quickly. On the other hand, when the queue is empty, we don't want to pay for a big worker pool that sits idle. With PubSub Stackdriver metrics adapter running on GKE we can easily autoscale our worker pool for minimum latency and maximum cost-effectivity.

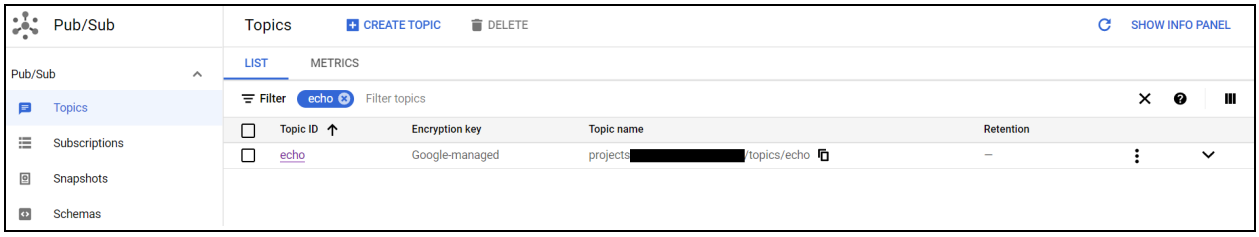
This lab shows autoscaling based on number of undelivered messages in a Cloud Pub/Sub subscription,

Setup for Pub/Sub Metrics based Autoscaling

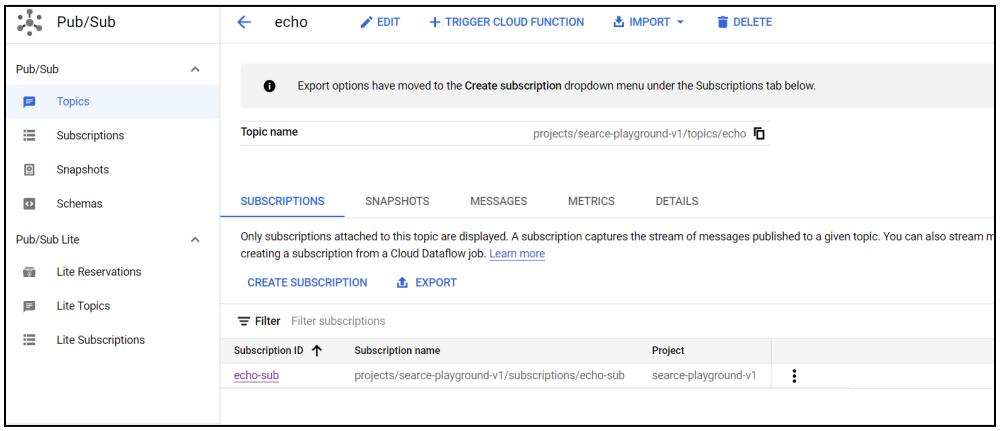
- In the Cloud Console, on the **Navigation menu** (☰), click **Pub/Sub** > **Topics.**
- To create a new topic, click **CREATE TOPIC**.
- A dialogue box will appear. Enter **Topic ID** as it is a mandatory field. Leave other fields with default values. Click on **CREATE TOPIC** to create a new topic.



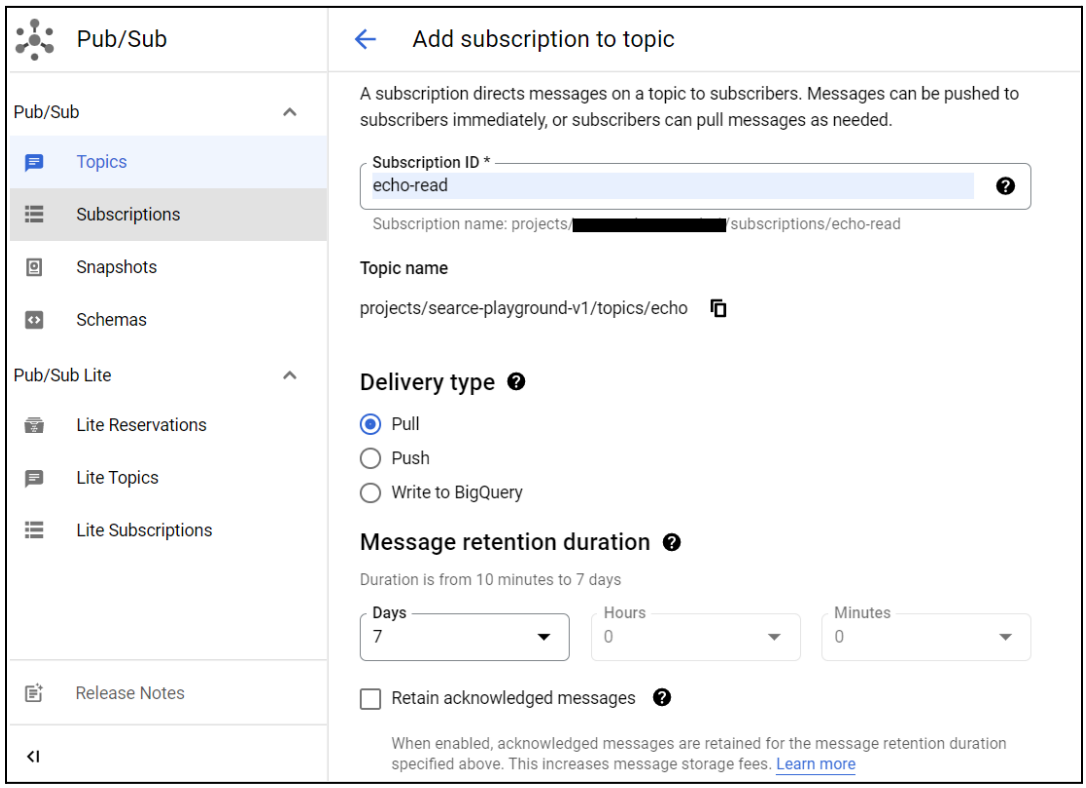
Topic named echo is created.



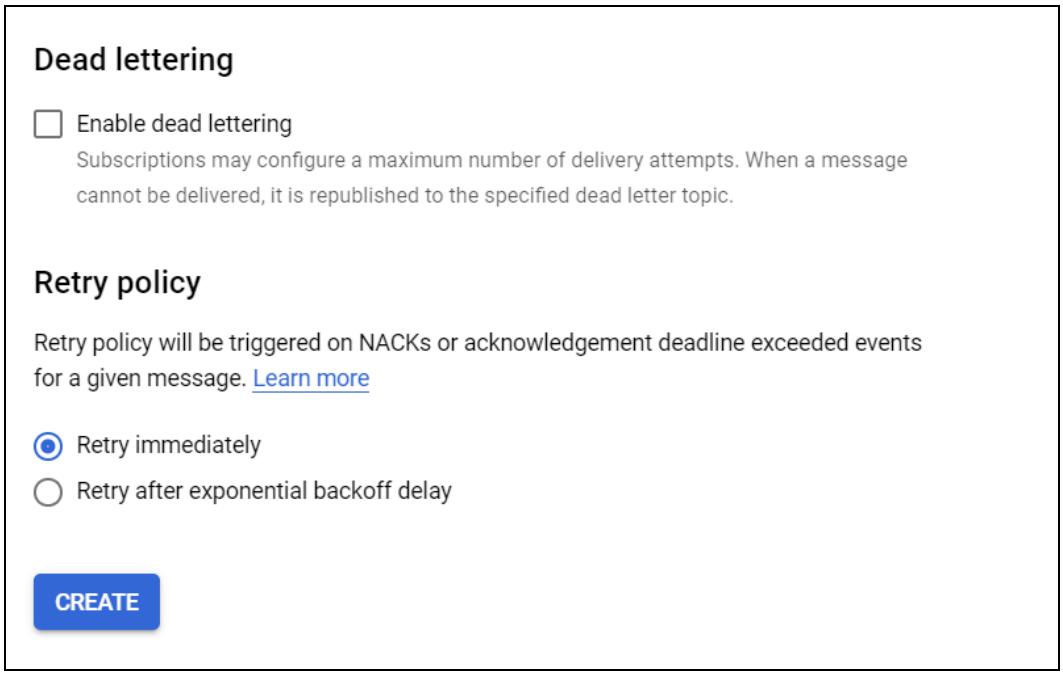
- Create a custom subscription. Click on topic, In Subscription it will contain the default subscription which was created with topic creation.



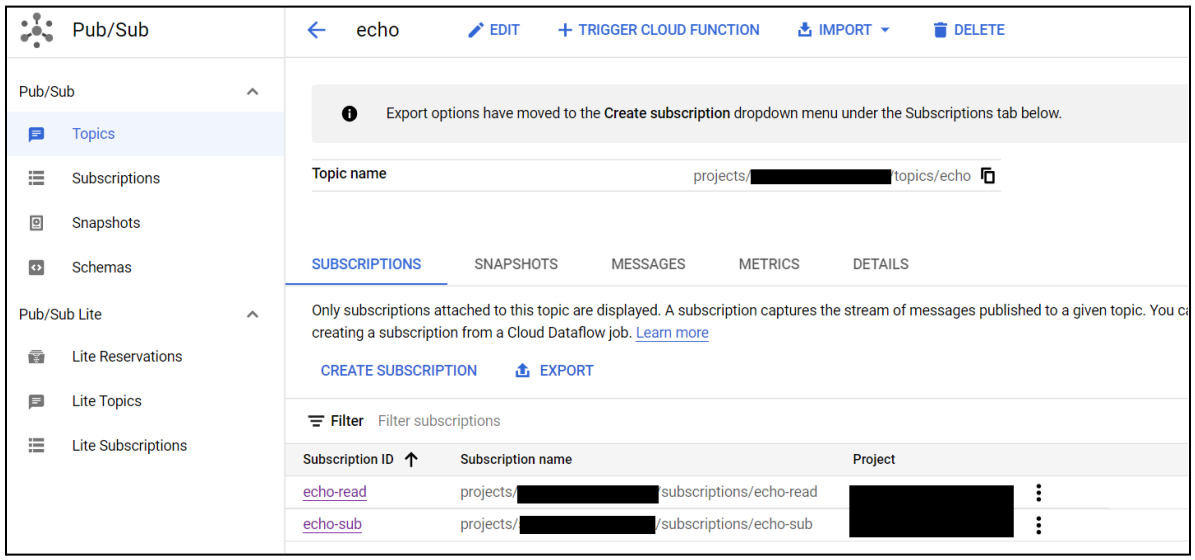
- Click on **CREATE SUBSCRIPTION**, to create a custom subscription.
- Enter Subscription ID as it is a mandatory field. Select Delivery type as **Pull**.



- Keep the rest of the fields with default values and click on **CREATE**.



- Subscription named **echo-read** is created.



- Deploy the deployment.yaml and service.yaml from the cloned repository. Follow ***Deploying C++ Web Application on GKE*** described above.

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f deployment.yaml
W1116 10:03:00.790901   25901 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
deployment.apps/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f service.yaml
W1116 10:03:11.658616   25908 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
service/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get pods
W1116 10:03:18.182939   25913 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                     READY   STATUS    RESTARTS   AGE
myapp-6df67d5d8b-pjqkc   1/1     Running   0          16s
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get svc
W1116 10:03:21.590736   25917 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP      10.90.0.1     <none>        443/TCP        9m28s
myapp        LoadBalancer   10.90.3.49    <pending>     80:32564/TCP   9s
root@labs-demo-vm:~/labs-sample-app/gke#
```

- Delete the previously created HPA object by using the following command.

```
kubectl delete -f hpa.yaml

Or

kubectl get hpa
kubectl delete <hpa-name>
```

- Create a custom service account which will be required for stackdriver authorization. Follow the below commands

  Note: Add your Project ID in variable $GCP_PROJECT_ID.
  $GCP_PROJECT_ID = <project-id>

```
#create service account
gcloud iam service-accounts create custom-metrics-sd-adapter --project
"$GCP_PROJECT_ID"

#add roles to created service account
gcloud projects add-iam-policy-binding "$GCP_PROJECT_ID" \
  --member
"serviceAccount:custom-metrics-sd-adapter@$GCP_PROJECT_ID.iam.gservice
account.com" \
  --role "roles/monitoring.editor"
```

```
root@labs-demo-vm:~/labs-sample-app/gke# gcloud iam service-accounts create custom-metrics-sd-adapter --project
"searce-playground-v1"
Created service account [custom-metrics-sd-adapter].
root@labs-demo-vm:~/labs-sample-app/gke# gcloud projects add-iam-policy-binding "searce-playground-v1" \
>   --member "serviceAccount:custom-metrics-sd-adapter@searce-playground-v1.iam.gserviceaccount.com" \
>   --role "roles/monitoring.editor"
 [1] EXPRESSION=request.time < timestamp("2022-03-16T19:34:18.912Z"), TITLE=expire_10_hours
 [2] EXPRESSION=request.time < timestamp("2022-03-21T20:44:35.924Z"), TITLE=expire_access_12_hours
 [3] EXPRESSION=request.time < timestamp("2022-04-01T20:52:43.232Z"), TITLE=expire_access_6_hours
 [4] EXPRESSION=request.time < timestamp("2022-05-09T20:33:53.852Z"), TITLE=expire_access_6_hours
 [5] EXPRESSION=request.time < timestamp("2022-05-26T19:39:42.851Z"), TITLE=expire_in_8_hours
 [6] EXPRESSION=request.time < timestamp("2022-05-26T19:40:05.950Z"), TITLE=expire_in_8_hours
 [7] EXPRESSION=request.time < timestamp("2022-10-17T15:30:00.000Z"), TITLE=expire_after_8_hours
 [8] EXPRESSION=request.time < timestamp("2022-10-17T18:28:28.149Z"), TITLE=expire_access_8_hours
 [9] EXPRESSION=request.time < timestamp("2022-11-11T16:30:00.000Z"), TITLE=valid till 11th nov
 [10] EXPRESSION=request.time.getDayOfWeek() <= 6, TITLE=A week access
 [11] None
 [12] Specify a new condition
The policy contains bindings with conditions, so specifying a condition is required when adding a binding.
Please specify a condition.:  10
```

- To deploy a stackdriver adapter, various IAM roles and bindings are created. To deploy stackdriver IAM roles follow the following steps.

```
cd /labs-sample-app/gke/stackdriver

#deploy stackdriver iam-roles-bindings
kubectl apply -f iam-adapter.yaml
```

Few resources are created and deployed by using the above command.

```
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver# kubectl apply -f iam-adapter.yaml
W1116 10:18:48.018201   26140 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
namespace/custom-metrics created
serviceaccount/custom-metrics-stackdriver-adapter created
clusterrolebinding.rbac.authorization.k8s.io/custom-metrics:system:auth-delegator created
rolebinding.rbac.authorization.k8s.io/custom-metrics-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/custom-metrics-resource-reader created
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver#
```

- Impersonating & Binding the Google Managed Service account with Kubernetes service account using the following commands. Enter your project ID.

```
#iam binding
gcloud iam service-accounts add-iam-policy-binding \
  --role roles/iam.workloadIdentityUser \
  --member
"serviceAccount:$GCP_PROJECT_ID.svc.id.goog[custom-metrics/custom-metri
cs-stackdriver-adapter]" \
  "custom-metrics-sd-adapter@$GCP_PROJECT_ID.iam.gserviceaccount.com"

#annotating k8s service account

kubectl annotate serviceaccount custom-metrics-stackdriver-adapter \

"iam.gke.io/gcp-service-account=custom-metrics-sd-adapter@$GCP_PROJECT

_ID.iam.gserviceaccount.com" \

  --namespace custom-metrics
```

```
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver# gcloud iam service-accounts add-iam-policy-binding \
>   --role roles/iam.workloadIdentityUser \
>   --member "serviceAccount:searce-playground-v1.svc.id.goog[custom-metrics/custom-metrics-stackdriver-adapter]
" \
>   "custom-metrics-sd-adapter@searce-playground-v1.iam.gserviceaccount.com"
Updated IAM policy for serviceAccount [custom-metrics-sd-adapter@searce-playground-v1.iam.gserviceaccount.com].
bindings:
- members:
  - serviceAccount:searce-playground-v1.svc.id.goog[custom-metrics/custom-metrics-stackdriver-adapter]
  role: roles/iam.workloadIdentityUser
etag: BwXtk95fyZA=
version: 1
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver# kubectl annotate serviceaccount custom-metrics-stackdriver-
adapter \
>   "iam.gke.io/gcp-service-account=custom-metrics-sd-adapter@searce-playground-v1.iam.gserviceaccount.com" \
>   --namespace custom-metrics
W1116 10:23:56.865843   26207 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
serviceaccount/custom-metrics-stackdriver-adapter annotated
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver#
```

- Deploying the stackdriver adapter deployment using the following command.

```
cd /labs-sample-app/gke/stackdriver

#deploy stackdriver deployment
kubectl apply -f deploy-adapter.yaml

#get all objects deployed
kubectl get all -n custom-metrics
```

```
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver# kubectl apply -f deploy-adapter.yaml
W1116 10:25:26.168779   26213 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
deployment.apps/custom-metrics-stackdriver-adapter created
service/custom-metrics-stackdriver-adapter created
apiservice.apiregistration.k8s.io/v1beta1.custom.metrics.k8s.io created
apiservice.apiregistration.k8s.io/v1beta2.custom.metrics.k8s.io created
apiservice.apiregistration.k8s.io/v1beta1.external.metrics.k8s.io created
Warning: resource clusterroles/external-metrics-reader is missing the kubectl.kubernetes.io/last-applied-configu
ration annotation which is required by kubectl apply. kubectl apply should only be used on resources created dec
laratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched autom
atically.
clusterrole.rbac.authorization.k8s.io/external-metrics-reader configured
clusterrolebinding.rbac.authorization.k8s.io/external-metrics-reader created
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver# kubectl get all -n custom-metrics
W1116 10:25:39.962379   26218 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/custom-metrics-stackdriver-adapter-5885cc597f-jsftr   0/1     Pending   0          14s

NAME                                          TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)   AGE
service/custom-metrics-stackdriver-adapter    ClusterIP   10.90.3.138   <none>        443/TCP   14s

NAME                                            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/custom-metrics-stackdriver-adapter   0/1     1            0           14s

NAME                                                       DESIRED   CURRENT   READY   AGE
replicaset.apps/custom-metrics-stackdriver-adapter-5885cc597f   1         1         0       14s
root@labs-demo-vm:~/labs-sample-app/gke/stackdriver#
```

- Stackdriver adapter is successfully deployed. It's time to deploy Pub/sub **number of undelivered/unacknowledged** Metric based Horizontal Pod Autoscaler using the following command.

```
cd /labs-sample-app/gke
```

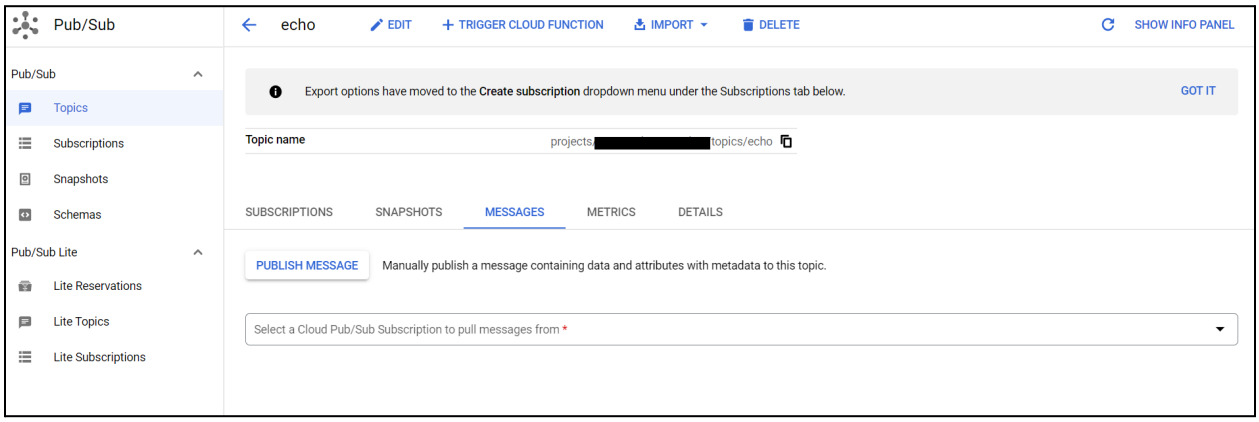
kubectl apply -f pubsub-hpa.yaml

kubectl get hpa

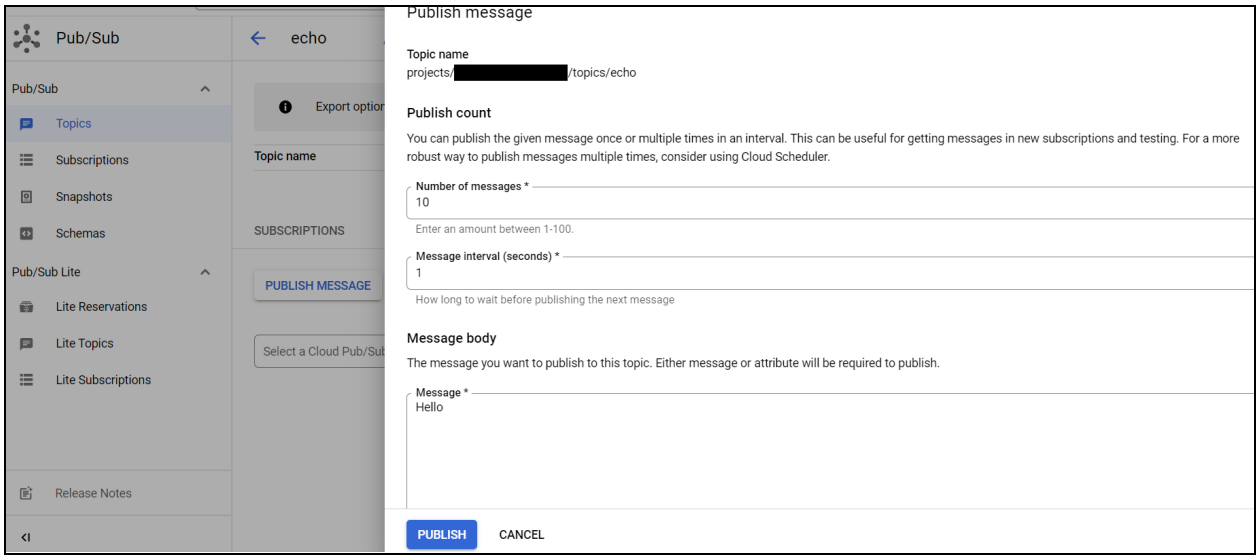
We'll autoscale between 1–5 replicas, based on external metric ***pubsub.googleapis.com|subscription|num_undelivered_messages*** from our echo-read subscription.
Target value is 2 undelivered messages.

```
root@labs-demo-vm:~/labs-sample-app/gke# kubectl apply -f pubsub-hpa.yaml
W1116 11:21:16.111308   26740 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
Warning: autoscaling/v2beta1 HorizontalPodAutoscaler is deprecated in v1.22+, unavailable in v1.25+; use autosca
ling/v2 HorizontalPodAutoscaler
horizontalpodautoscaler.autoscaling/myapp created
root@labs-demo-vm:~/labs-sample-app/gke# kubectl get hpa
W1116 11:21:33.437378   26746 gcp.go:119] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v
1.26+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME    REFERENCE          TARGETS            MINPODS   MAXPODS   REPLICAS   AGE
myapp   Deployment/myapp   <unknown>/2 (avg)   1         6         0          16s
root@labs-demo-vm:~/labs-sample-app/gke#
```

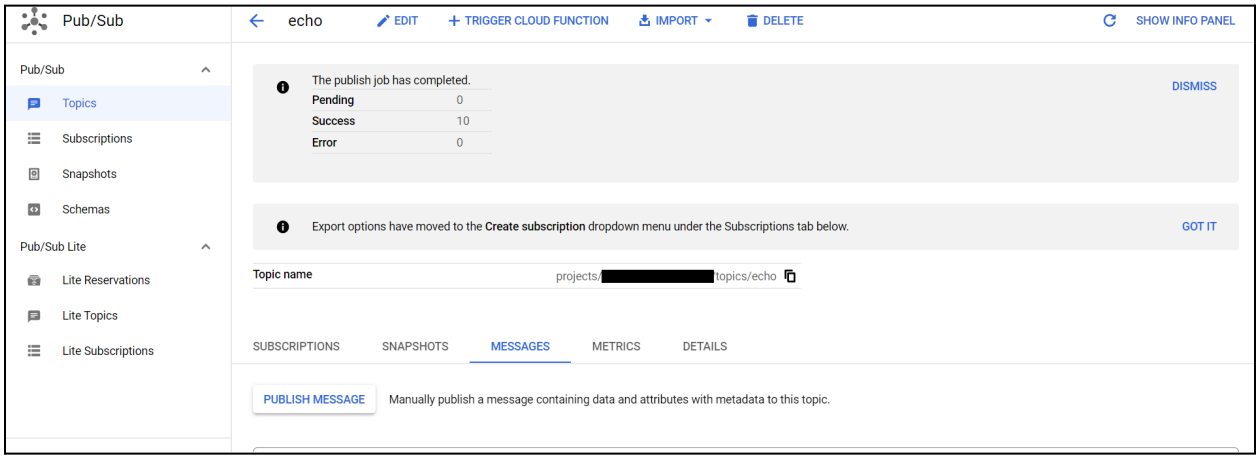
- Navigate to pub/sub topic created previously named **echo**. Open the topic, go to the **MESSAGES** section. Here we will publish some messages. Click on **PUBLISH MESSAGE**.



- A dialogue box will appear to publish the message. Lets publish 10 messages.
  Enter **Number of messages** to **10.**
  In the **Message body** enter your message like **hello**.
  Finally, click on **PUBLISH** to publish the messages.



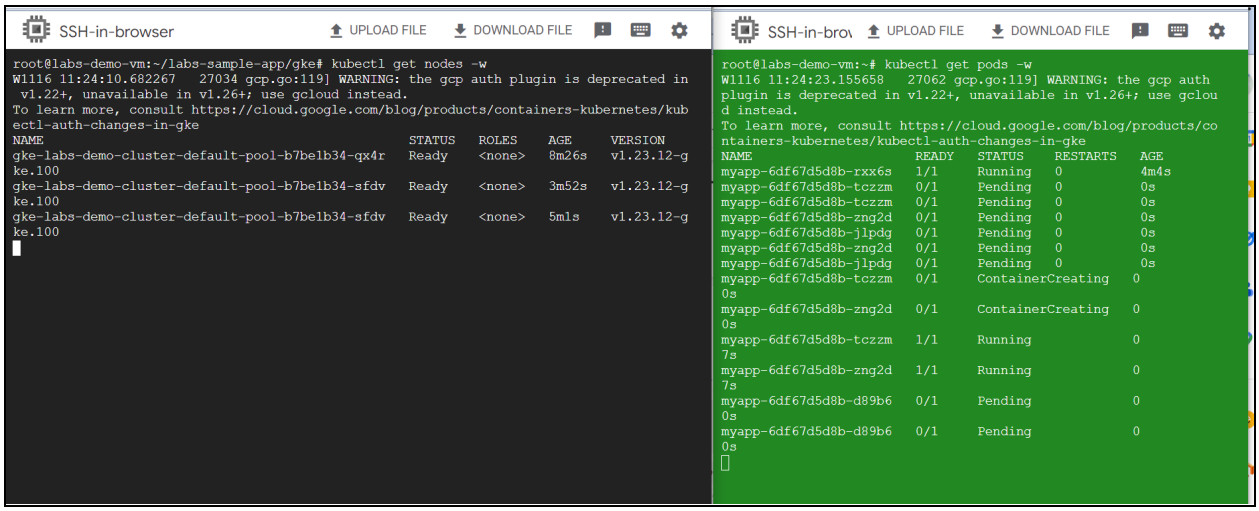
- It will start publishing the messages.



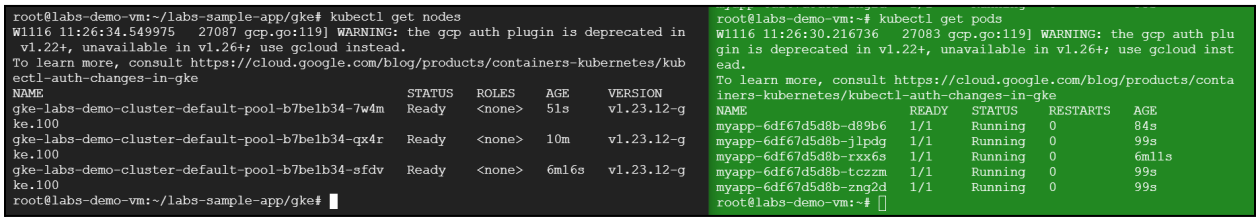
- As there are 10 undelivered/unacknowledged messages in queue, as per Horizontal Pod Autoscaler it will trigger to scale more nodes. Open 2 terminals to view Node & Pod autoscaling.

  Terminal 1(Black): Node autoscaling
  The number of nodes autoscaled from 1 to 3.

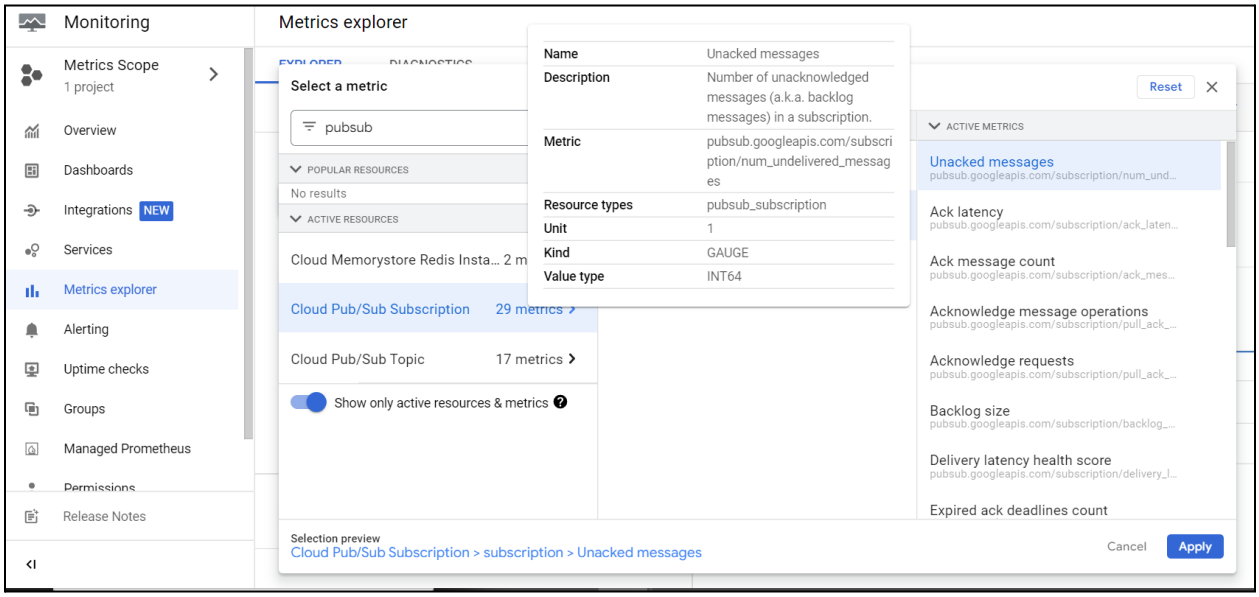
  Terminal 2(Green): Pod Autoscaling
  The number of pods autoscaled from 1 to 5. Many pods will get created but only 5 will be in active/running state as per the HPA definitions.



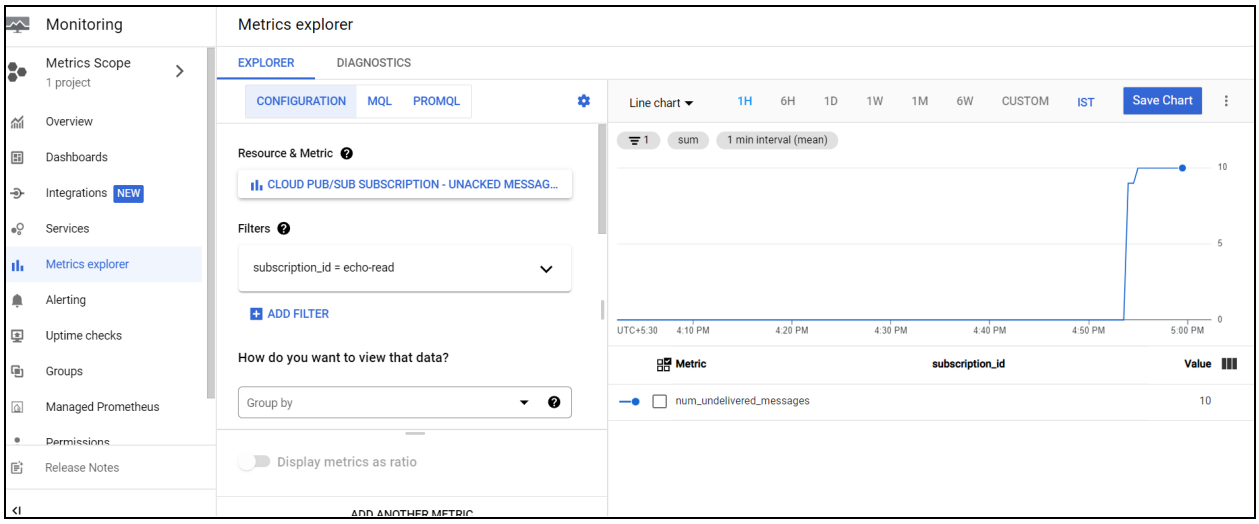
- Nodes and Pods autoscaled Successfully based on number of undelivered/unacknowledged messages in Pub/Sub.



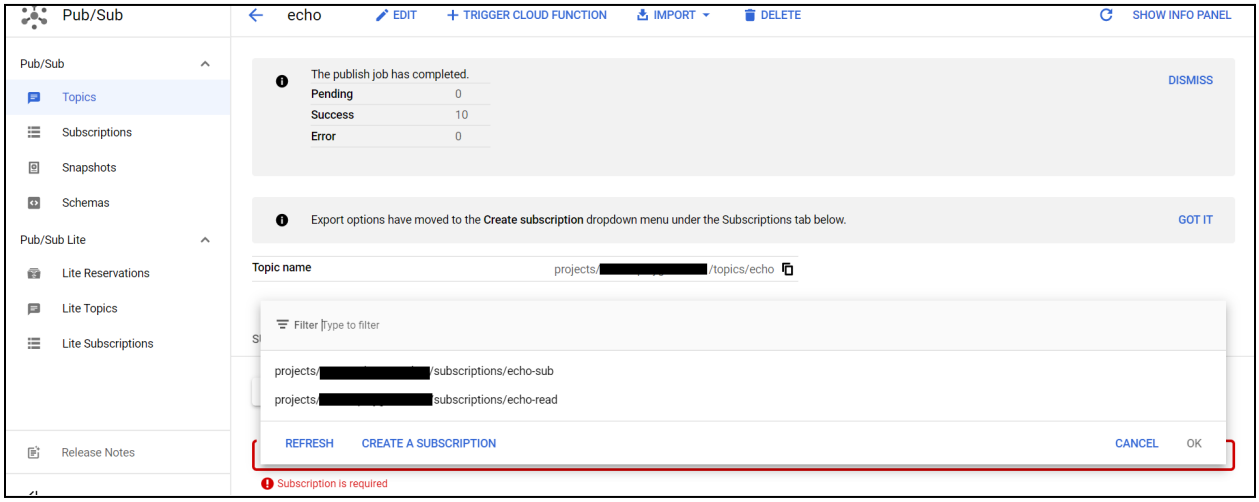
- Open a new tab. In the Cloud Console, on the **Navigation menu** (☰), click **Monitoring** > **Metric Explorer**.
- Click on Resource & Metric, search Pub/Sub. In Pub/Sub search for **Unacked messages** metrics and click on **Apply**.

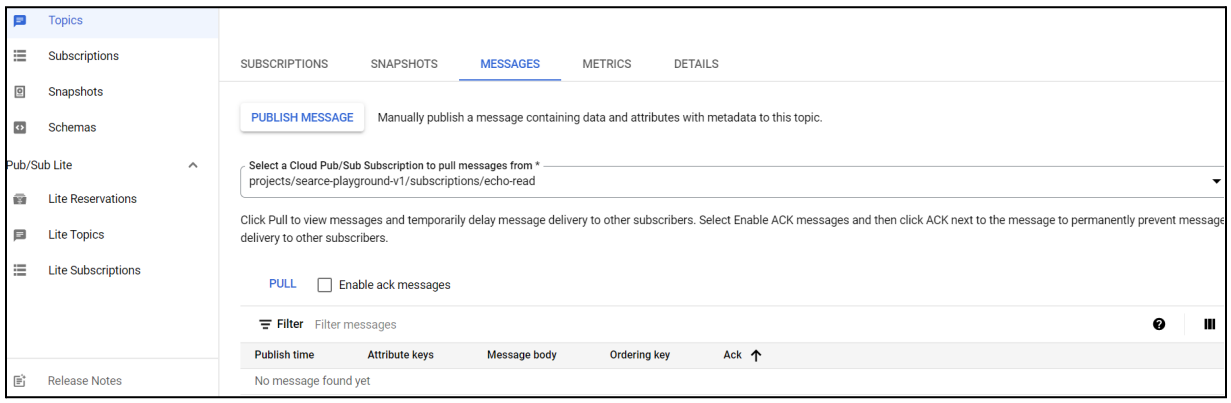


- It will show no of undelivered/unacknowledged messages in Pub/Sub. As we have published 10 messages the graph shows the same 10 messages.



- Let's acknowledge the messages, navigate to Pub/Sub topic. Go in **MESSAGES**, select the subscription **echo-read**.

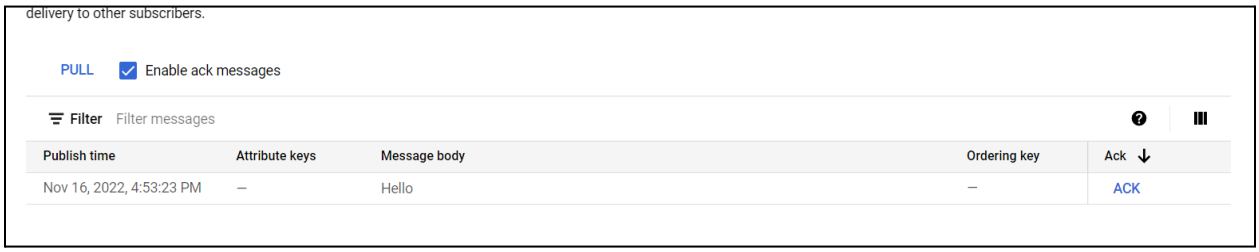




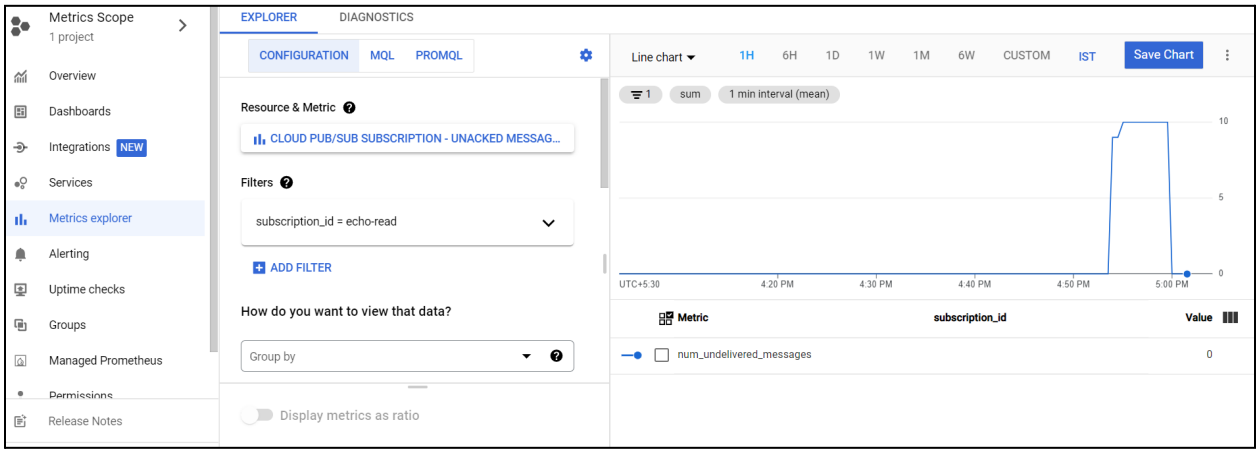
- As it is a Pull based subscription, Click on **PULL** & select the checkbox **Enable ack messages**. It will list all the published messages with the **ACK** field at the end.

Note : ACK will show as a deadline exceed in some second so complete the ACK step ASAP

- Click on **ACK** of each message which is at the end of row or last column. As soon as you click on **ack** messages will disappear from the list.



- Visit the monitoring tab which shows the details above unacknowledged messages. Refresh the page to view the dip in unacknowledged messages from 10 to 0.



- As there are no messages left unacknowledged HPA will be triggered to scale down the pods as well as nodes.

Conclusion

In this lab, we deployed C++ web application on the GKE cluster. Autoscaled the Nodes & pods of application by using Horizontal Pod Autoscaler based on CPU utilization. Also, Autoscaled the Nodes & pods of application by using Horizontal Pod Autoscaler based on Pub/Sub trigger using external Pub/Sub metric.

References:

- Google Kubernetes Engine - GKE
- Pub/Sub in GCP - Pub/Sub
- Overview about Kubernetes - K8s