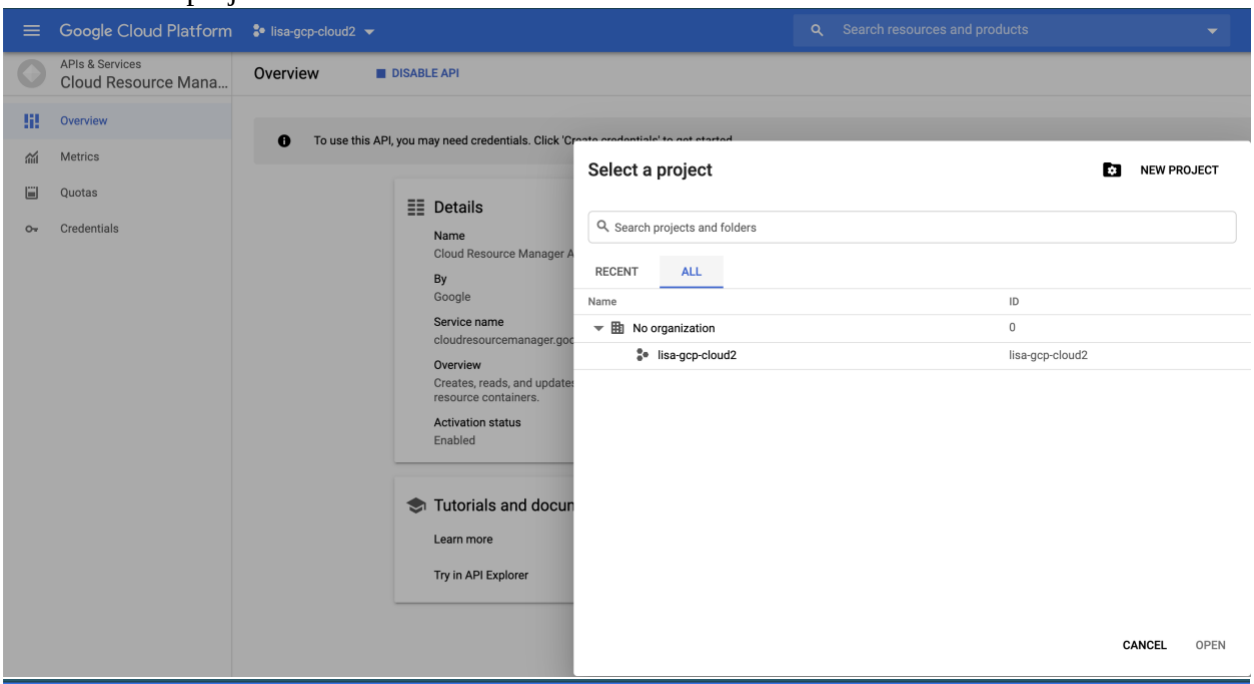
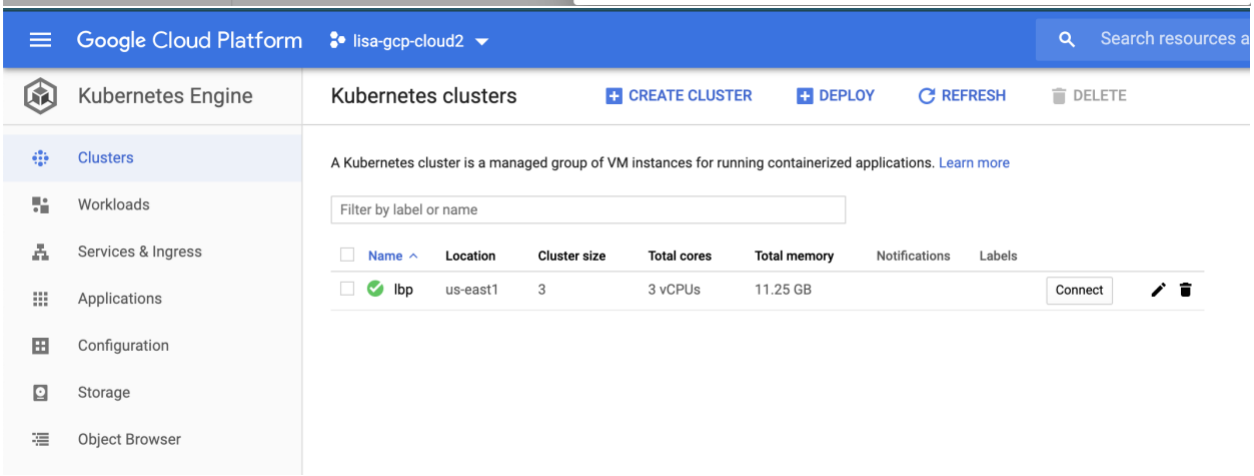
Practical Assignment

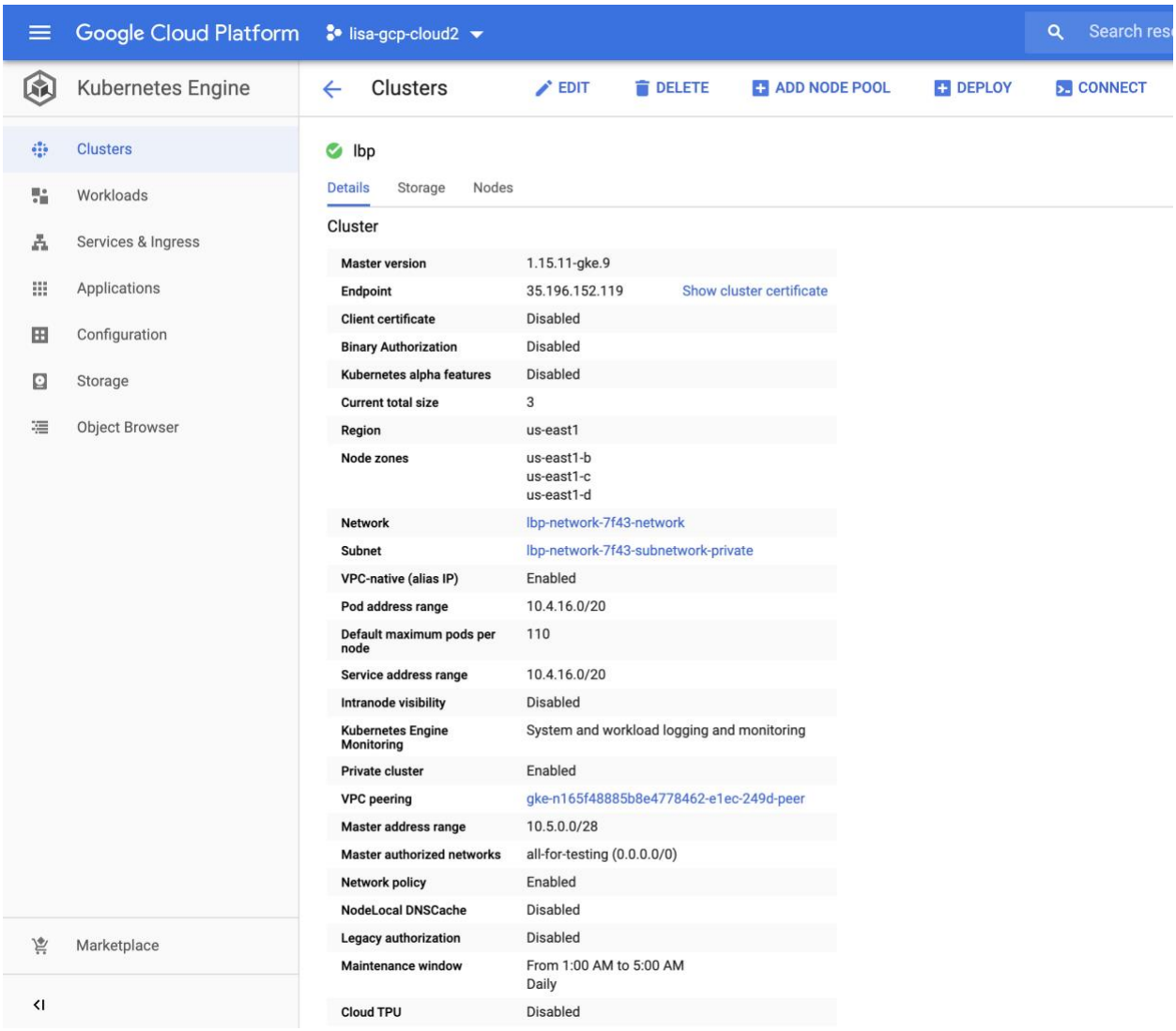
Requirements:

This assignment is designed to test your experience with infrastructure as code (IaC) and Kubernetes.
In the assignment, you are asked to deliver a script with an IaC script (deployment manager, terraform, API via ruby, etc) that will perform the following

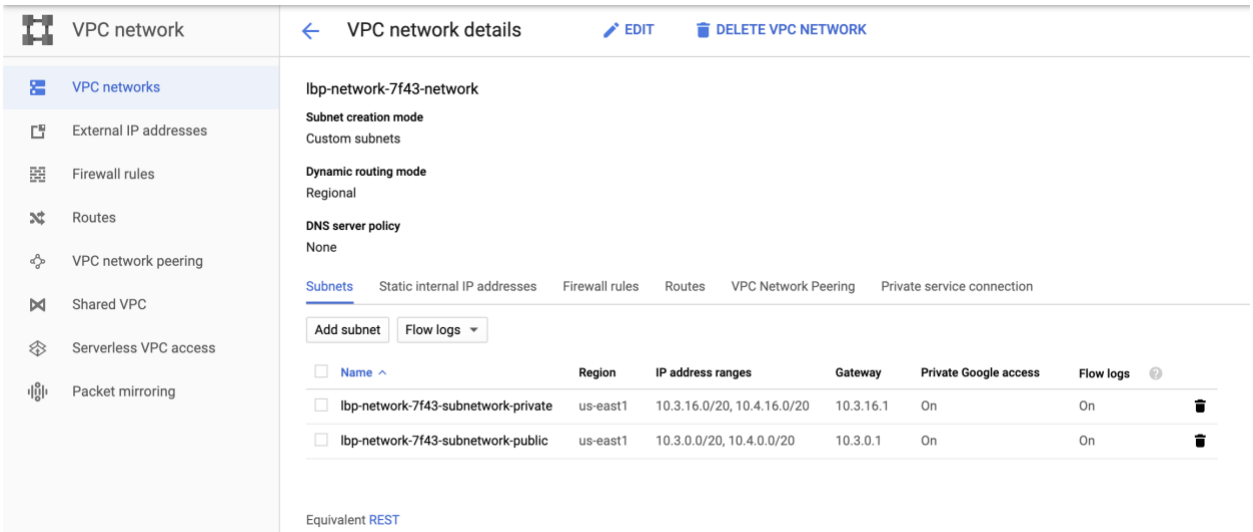
Tasks:

- Create a GCP project
- 

- 



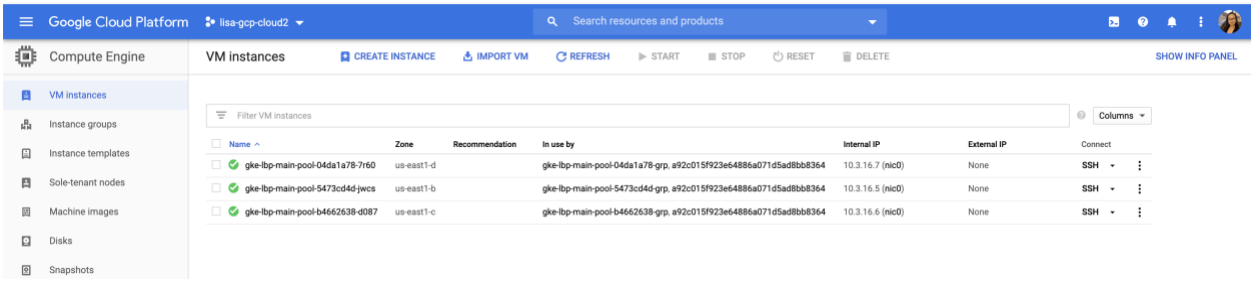
- Create a custom network



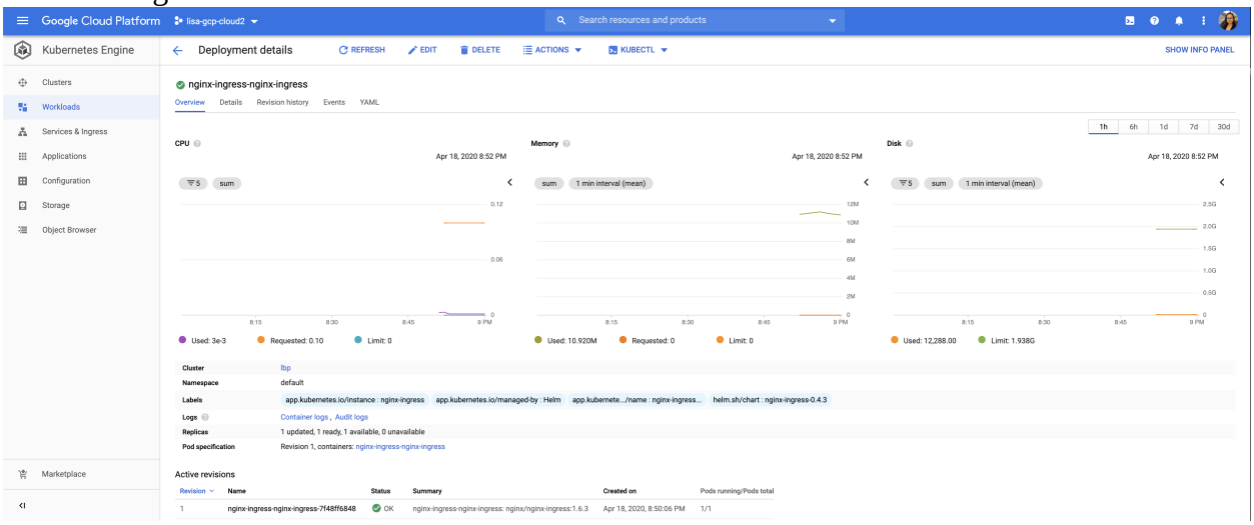
- Create a custom subnet *(e.g: us-east1)*

-

Create a regional GKE Cluster on 3 availability zones in the project that can be scaled up to 500 nodes. *(By default, regional clusters will create nodes across 3 zones in a region.*



- Create a Kubernetes deployment on the GKE cluster created to deploy the official Nginx Docker image



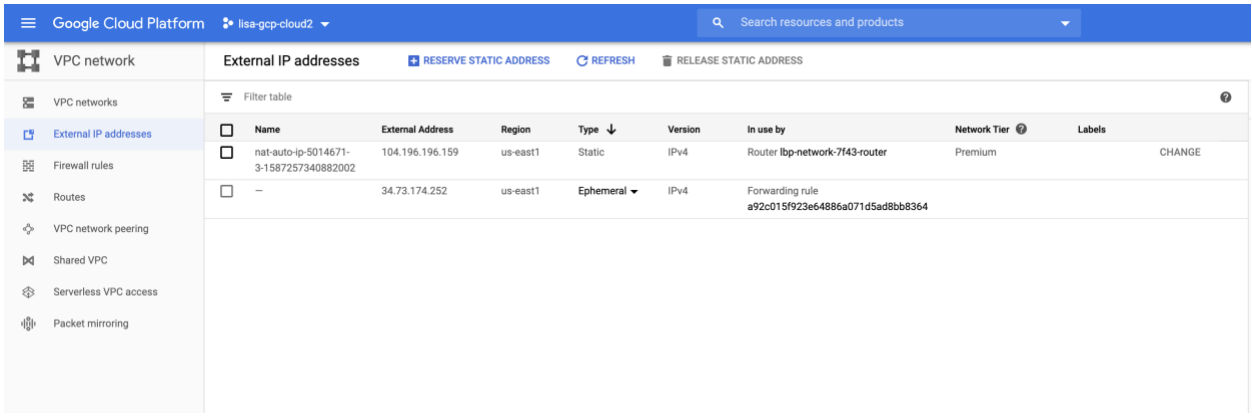
-
- configure external HTTP(S) load balancer to publish the Nginx deployment
  Reference:
- https://docs.nginx.com/nginx-ingress-controller/installation/installation-with-helm/

-
    - setup HTTPS (TLS) between client and load balancer (a self-signed certificate is sufficient)

-
```
bash-3.2#
bash-3.2# kubectl get nodes -o wide
NAME                                STATUS   ROLES    AGE     VERSION          INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                             KERNEL-VERSION   CONTAINER-RUNTIME
gke-lbp-main-pool-04da1a78-7r60   Ready    <none>   2m54s   v1.15.11-gke.9   10.3.16.7                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
gke-lbp-main-pool-5473cd4d-jwcs   Ready    <none>   2m56s   v1.15.11-gke.9   10.3.16.5                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
gke-lbp-main-pool-b4662638-d087   Ready    <none>   2m55s   v1.15.11-gke.9   10.3.16.6                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
bash-3.2# kubectl get service
NAME                                TYPE           CLUSTER-IP    EXTERNAL-IP     PORT(S)                      AGE
kubernetes                          ClusterIP      10.4.16.1     <none>          443/TCP                      11m
nginx-ingress-nginx-ingress         LoadBalancer   10.4.23.186   34.73.174.252   80:31443/TCP,443:31389/TCP   5m57s
bash-3.2# kubectl get secret

NAME                                             TYPE                                  DATA   AGE
default-token-dpr9b                              kubernetes.io/service-account-token   3      11m
nginx-ingress-nginx-ingress-default-server-secret   Opaque                                2      6m9s
nginx-ingress-nginx-ingress-token-kg26s          kubernetes.io/service-account-token   3      6m9s
sh.helm.release.v1.nginx-ingress.v1              helm.sh/release.v1                    1      6m10s
bash-3.2#
bash-3.2#
bash-3.2#
bash-3.2# kubectl describe secret nginx-ingress-nginx-ingress-default-server-secret
Name:         nginx-ingress-nginx-ingress-default-server-secret
Namespace:    default
Labels:       app.kubernetes.io/instance=nginx-ingress
              app.kubernetes.io/managed-by=Helm
              app.kubernetes.io/name=nginx-ingress-nginx-ingress
              helm.sh/chart=nginx-ingress-0.4.3
Annotations:  <none>

Type:  Opaque

Data
====
tls.key:  1679 bytes
tls.crt:  1013 bytes
bash-3.2#
```
  - use a static IP address
  - 

- script should accept the following variables (When you run #terrafrom plan and #terraform apply, please input project name, region name and cluster name)
  - GCP project name
  - GCP region name
  - GKE cluster name
- script should output the following attributes
  - Kubernetes API endpoint
  - curl http://localhost:8080/api/v1/namespaces/default/endpoints
  - the external IP that the nginx is exposed at.

```
bash-3.2# kubectl get secret
NAME                                                TYPE                                  DATA   AGE
default-token-6c6bx                                 kubernetes.io/service-account-token   3      15m
nginx-ingress-nginx-ingress-default-server-secret   Opaque                                2      9m37s
nginx-ingress-nginx-ingress-token-tg29l             kubernetes.io/service-account-token   3      9m37s
sh.helm.release.v1.nginx-ingress.v1                 helm.sh/release.v1                    1      9m38s
bash-3.2# kubectl describe secret nginx-ingress-nginx-ingress-default-server-secret
Name:         nginx-ingress-nginx-ingress-default-server-secret
Namespace:    default
Labels:       app.kubernetes.io/instance=nginx-ingress
              app.kubernetes.io/managed-by=Helm
              app.kubernetes.io/name=nginx-ingress-nginx-ingress
              helm.sh/chart=nginx-ingress-0.4.3
Annotations:  <none>

Type:  Opaque

Data
====
tls.crt:  1013 bytes
tls.key:  1679 bytes
```

Delivery:

- Copy your work into a zipped folder that is named QT_FirstName_Lastname.zip
- The folder should contain all your work and the documentation
- In the documentation explain what you did and how to run your code
- We are not going to test a running environment; just review the code you provide.

Note: If there some caveats or concerns about your approach please tell about them. Given the exercise, not all things can be perfect so we would love to hear your thoughts on how to make this production-ready if given time and resources.

IaaS – GCP Introduction

Step 0: Clone code

git clone https://github.com/LIUBOPENG/GCP.git
cd GCP

Or:

- # unzip QT_Bopeng_Liu.zip

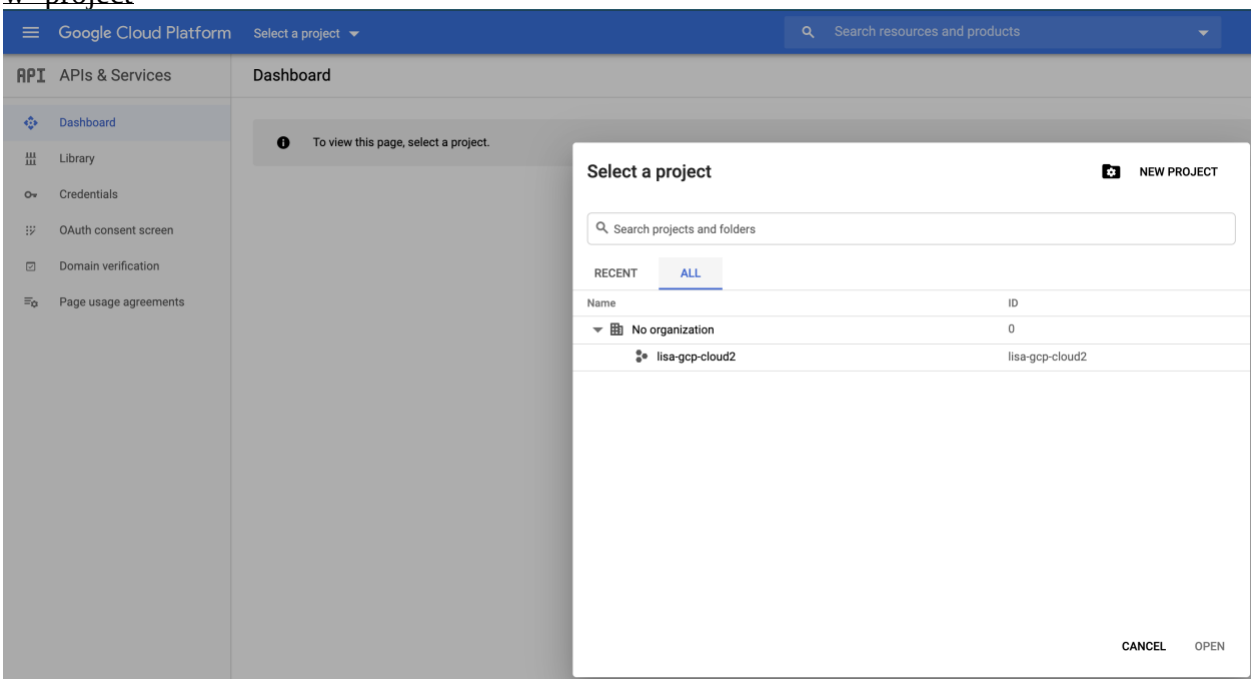
Step 1: Create project

All the steps are included in 001-create-project.sh

```
bash-3.2# vim 001-create-project.sh
bash-3.2# ./001-create-project.sh
Please input project ID:
lisa-gcp-cloud2
Project ID: lisa-gcp-cloud2
Create in progress for [https://cloudresourcemanager.googleapis.com/v1/projects/lisa-gcp-cloud2].
Waiting for [operations/cp.6327301690345238724] to finish...done.
Enabling service [cloudapis.googleapis.com] on project [lisa-gcp-cloud2]...
Operation "operations/acf.a6772f26-bd52-4d14-ac31-99a491e638bc" finished successfully.
bash-3.2#
```

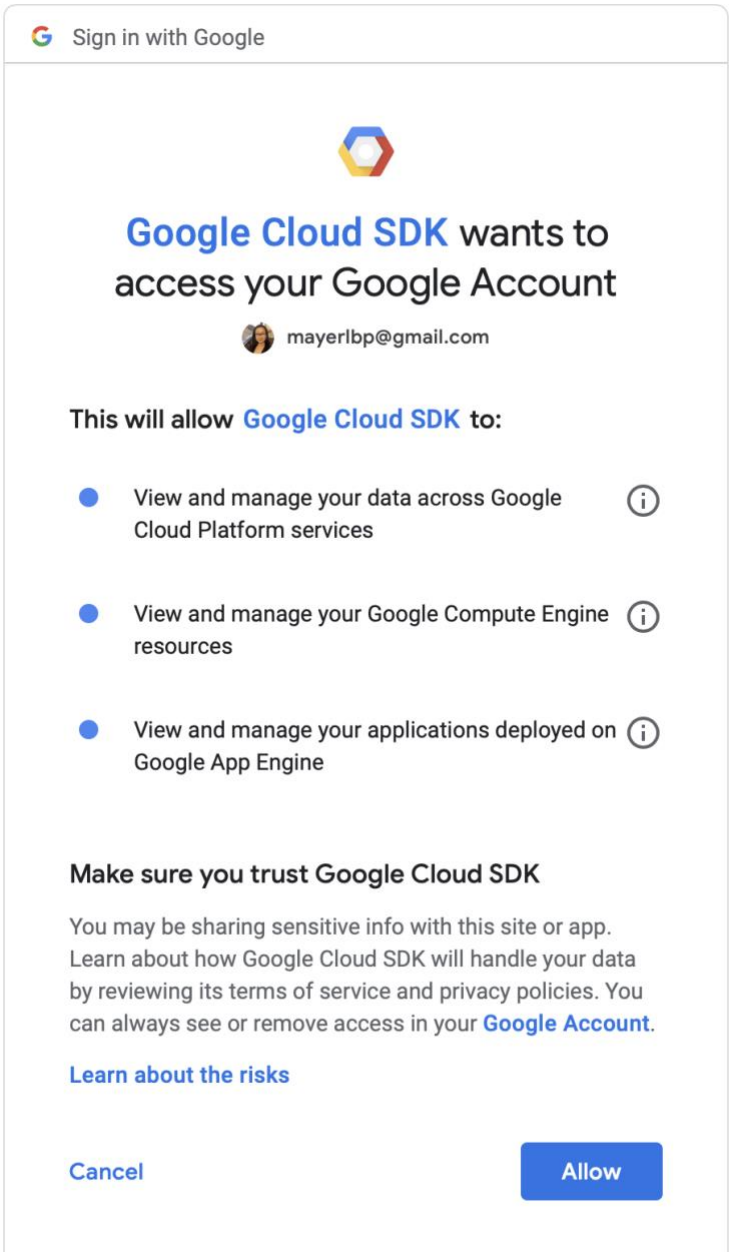
Check new project name in the dashboard:

https://console.cloud.google.com/projectselector2/apis/dashboard?authuser=1&supportedpurview=project



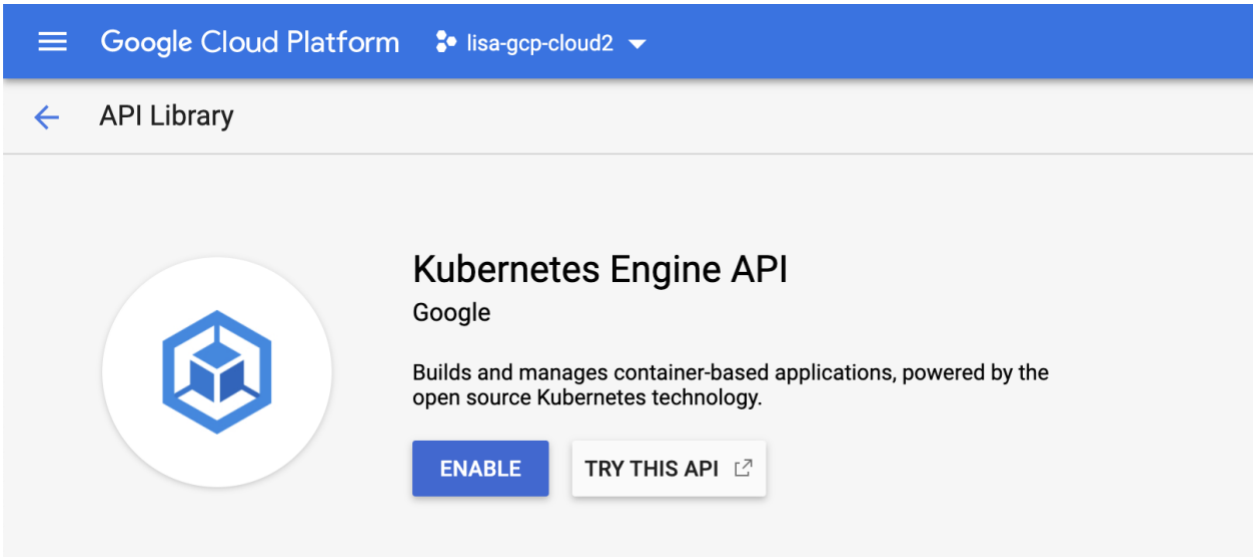
Step2:
#gcloud init

Visit the URL in the above picture and give access to certain user.

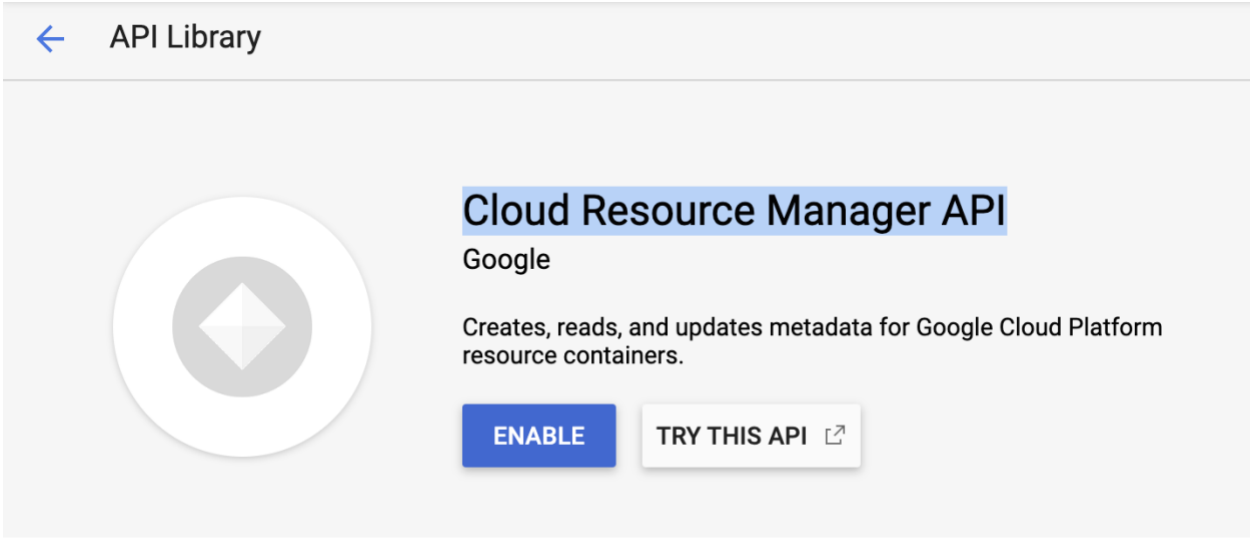


Step3: Enable advanced features

- Enable Kubernetes Engine API



- Enable Cloud Resource Manager API



Step 4: Create Credentials (Please run the following command and replace your username)

gcloud iam service-accounts create **lisa-gcp-cloud2**
gcloud projects add-iam-policy-binding **lisa-gcp-cloud2** --member "serviceAccount:**lisa-gcp-cloud2@lisa-gcp-cloud2**.iam.gserviceaccount.com" --role "roles/owner"
#gcloud iam service-accounts keys create **lisa-gcp-cloud2.json** --iam-account **lisa-gcp-cloud2@lisa-gcp-cloud2**.iam.gserviceaccount.com
#export GOOGLE_APPLICATION_CREDENTIAL="/Users/mayerlbp/GCP-2/3/terraform-google-gke/**lisa-gcp-cloud2.json**"

```
bash-3.2# gcloud iam service-accounts create lisa-gcp-cloud2
Created service account [lisa-gcp-cloud2].
bash-3.2# gcloud projects add-iam-policy-binding lisa-bopeng-cloud2 --member "serviceAccount:lisa-bopeng-cloud2@lisa-bopeng-cloud2.iam.gserviceaccount.com" --role "roles/owner"
ERROR: (gcloud.projects.add-iam-policy-binding) User [mayerlbp@gmail.com] does not have permission to access project [lisa-bopeng-cloud2:getIamPolicy] (or it may not exist): Th
ve permission
bash-3.2# gcloud projects add-iam-policy-binding lisa-bopeng-cloud2 --member "serviceAccount:lisa-gcp-cloud2@lisa-gcp-cloud2.iam.gserviceaccount.com" --role "roles/owner"
ERROR: (gcloud.projects.add-iam-policy-binding) User [mayerlbp@gmail.com] does not have permission to access project [lisa-bopeng-cloud2:getIamPolicy] (or it may not exist): Th
ve permission
bash-3.2# gcloud projects add-iam-policy-binding lisa-gcp-cloud2 --member "serviceAccount:lisa-gcp-cloud2@lisa-gcp-cloud2.iam.gserviceaccount.com" --role "roles/owner"
Updated IAM policy for project [lisa-gcp-cloud2].
bindings:
- members:
  - serviceAccount:service-41837938635@compute-system.iam.gserviceaccount.com
  role: roles/compute.serviceAgent
- members:
  - serviceAccount:service-41837938635@container-engine-robot.iam.gserviceaccount.com
  role: roles/container.serviceAgent
- members:
  - serviceAccount:41837938635-compute@developer.gserviceaccount.com
  - serviceAccount:41837938635@cloudservices.gserviceaccount.com
  - serviceAccount:service-41837938635@containerregistry.iam.gserviceaccount.com
  role: roles/editor
- members:
  - serviceAccount:lisa-gcp-cloud2@lisa-gcp-cloud2.iam.gserviceaccount.com
  - user:mayerlbp@gmail.com
  role: roles/owner
etag: BwWjmedHQys=
version: 1
bash-3.2#
```

Step 5: Input GKE name, region, project id when you run:

#terraform init
#terraform plan

```
  # module.vpc_network.module.network_firewall.google_compute_firewall.public_allow_all_inbound will be created
  + resource "google_compute_firewall" "public_allow_all_inbound" {
      + creation_timestamp = (known after apply)
      + destination_ranges = (known after apply)
      + direction          = "INGRESS"
      + id                 = (known after apply)
      + name               = (known after apply)
      + network            = (known after apply)
      + priority           = 1000
      + project            = "lisa-gcp-cloud2"
      + self_link          = (known after apply)
      + source_ranges      = [
          + "0.0.0.0/0",
        ]
      + target_tags        = [
          + "public",
        ]

      + allow {
          + ports    = []
          + protocol = "all"
        }
    }

Plan: 19 to add, 0 to change, 0 to destroy.

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.

bash-3.2#
```

Step 6: Input GKE name, region, project id when you run:

#terraform apply

```
bash-3.2# ls -al
total 40
drwxr-xr-x  7 root  staff    224 18 Apr 20:24 .
drwxr-xr-x  3 root  staff     96 18 Apr 16:48 ..
drwxr-xr-x  4 root  staff    128 18 Apr 16:50 .terraform
-rwxr-xr-x  1 root  staff    127 18 Apr 20:21 001-create-project.sh
-rw-r--r--  1 root  staff  10328 18 Apr 16:50 main.tf
drwxr-xr-x  4 root  staff    128 18 Apr 16:48 modules
-rw-r--r--  1 root  staff   3145 18 Apr 18:01 variables.tf
bash-3.2#
bash-3.2# terraform init
Initializing modules...

Initializing the backend...

Initializing provider plugins...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.null: version = "~> 2.1"
* provider.random: version = "~> 2.2"
* provider.template: version = "~> 2.1"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
bash-3.2#
```

Step 7: Set up Kubernetes Proxy:
#kubectl proxy &

```
bash-3.2# kubectl proxy &
[2] 28788
bash-3.2# Starting to serve on 127.0.0.1:8001
```

Step 8: Get endpoints of GKE:
#curl http://localhost:8001/api/v1/namespaces/default/endpoints

```
bash-3.2# curl http://localhost:8001/api/v1/namespaces/default/endpoints
{
  "kind": "EndpointsList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/namespaces/default/endpoints",
    "resourceVersion": "7918"
  },
  "items": [
    {
      "metadata": {
        "name": "kubernetes",
        "namespace": "default",
        "selfLink": "/api/v1/namespaces/default/endpoints/kubernetes",
        "uid": "68da7b3c-172a-42f0-b31b-5ad34eb01c21",
        "resourceVersion": "147",
        "creationTimestamp": "2020-04-18T20:55:20Z"
      },
      "subsets": [
        {
          "addresses": [
            {
              "ip": "10.5.0.2"
            }
          ],
          "ports": [
            {
              "name": "https",
              "port": 443,
              "protocol": "TCP"
            }
          ]
        }
      ]
    },
    {
      "metadata": {
        "name": "nginx-ingress-nginx-ingress",
        "namespace": "default",
        "selfLink": "/api/v1/namespaces/default/endpoints/nginx-ingress-nginx-ingress",
        "uid": "27bf85d2-42dd-4ac1-b1e9-bdc1be5aa892",
        "resourceVersion": "4730",
        "creationTimestamp": "2020-04-18T21:01:19Z",
        "labels": {
          "app.kubernetes.io/instance": "nginx-ingress",
          "app.kubernetes.io/managed-by": "Helm",
          "app.kubernetes.io/name": "nginx-ingress-nginx-ingress",
          "helm.sh/chart": "nginx-ingress-0.4.3"
        }
      },
      "subsets": [
        {
          "addresses": [
            {
              "ip": "10.4.17.5",
              "nodeName": "gke-example-cluster-main-pool-e3bfb679-n08j",
              "targetRef": {
                "kind": "Pod",
                "namespace": "default",
                "name": "nginx-ingress-nginx-ingress-7f48ff6848-cvbfj",
                "uid": "1816e001-457b-461a-8eef-37179c61138a",
                "resourceVersion": "4204"
              }
            }
          ],
          "ports": [
            {
              "name": "http",
              "port": 80,
              "protocol": "TCP"
            },
            {
              "name": "https",
              "port": 443,
              "protocol": "TCP"
            }
          ]
        }
      ]
    }
  ]
```

Step 8: Get the external IP of nginx is exposed at:

#kubectl get service

```
bash-3.2#
bash-3.2# kubectl get nodes -o wide
NAME                                STATUS   ROLES    AGE     VERSION          INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                             KERNEL-VERSION   CONTAINER-RUNTIME
gke-lbp-main-pool-04da1a78-7r60   Ready    <none>   2m54s   v1.15.11-gke.9   10.3.16.7                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
gke-lbp-main-pool-5473cd4d-jwcs   Ready    <none>   2m56s   v1.15.11-gke.9   10.3.16.5                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
gke-lbp-main-pool-b4662638-d087   Ready    <none>   2m55s   v1.15.11-gke.9   10.3.16.6                   Container-Optimized OS from Google   4.19.104+        docker://19.3.1
bash-3.2# kubectl get service
NAME                       TYPE           CLUSTER-IP    EXTERNAL-IP     PORT(S)                      AGE
kubernetes                 ClusterIP      10.4.16.1     <none>          443/TCP                      11m
nginx-ingress-nginx-ingress   LoadBalancer   10.4.23.186   34.73.174.252   80:31443/TCP,443:31389/TCP   5m57s
bash-3.2# kubectl get secret

NAME                                          TYPE                                  DATA   AGE
default-token-dpr9b                           kubernetes.io/service-account-token   3      11m
nginx-ingress-nginx-ingress-default-server-secret   Opaque                                2      6m9s
nginx-ingress-nginx-ingress-token-kg26s       kubernetes.io/service-account-token   3      6m9s
sh.helm.release.v1.nginx-ingress.v1           helm.sh/release.v1                    1      6m10s
bash-3.2#
bash-3.2#
bash-3.2#
bash-3.2# kubectl describe secret nginx-ingress-nginx-ingress-default-server-secret
Name:         nginx-ingress-nginx-ingress-default-server-secret
Namespace:    default
Labels:       app.kubernetes.io/instance=nginx-ingress
              app.kubernetes.io/managed-by=Helm
              app.kubernetes.io/name=nginx-ingress-nginx-ingress
              helm.sh/chart=nginx-ingress-0.4.3
Annotations:  <none>

Type:  Opaque

Data
====
tls.key:  1679 bytes
tls.crt:  1013 bytes
bash-3.2#
```

The external IP of nginx service can be got by command:

#kubectl get service | grep nginx | awk '{print $4}'
For example: (Test result in another GKE cluster, so IP is different)

```
bash-3.2# kubectl get service | grep nginx |awk '{print $4}'
35.237.232.18
```

Test result:
Nginx service works.