Github Link:
https://github.com/HaydenDuong/SIT323_Cloud_Native_Application_Development/tree/main/Coding%20Tasks/10.1P

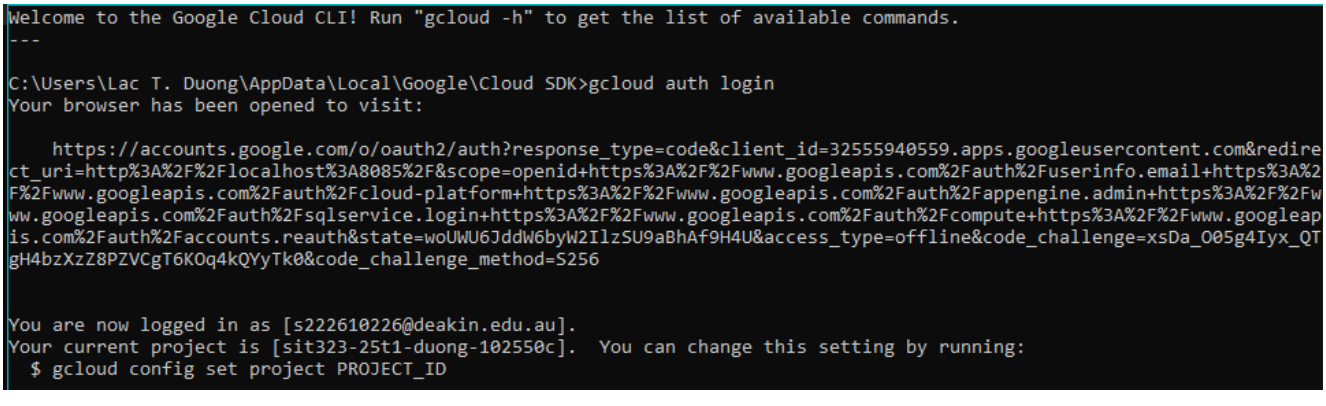
A/ Tools:

- Node.js
- Docker
- Kubernetes
- Google Cloud Platform Services
    - Artifact Registry Repository
    - VPC Network
    - Google Kubernetes Engine Cluster
    - Logging – Log Explorer

B/ Steps

Accessing the assigned Google Cloud Project by the following steps:

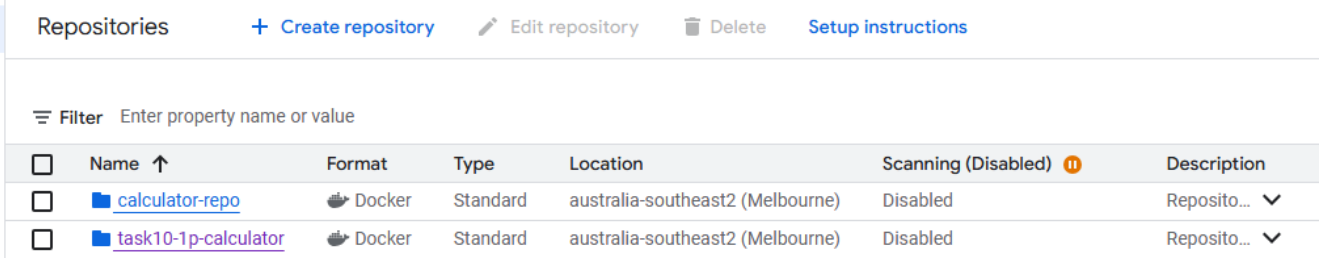
1. Log in Google Cloud via Google Cloud SDK Shell through command “gcloud auth login”.
2. Enter Deakin Email account in the pop-up Google login page.
3. Enter command “gcloud projects list” to check which projects are assigned to the current login accound.
4. Type in “gcloud config set project <project_id>”, in this case, “sit323-25t1-duong-10255c” to switch to that project.
5. (Optional) Re-check the current project by “gcloud config get-value project”.



```
C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud projects list
PROJECT_ID                  NAME                        PROJECT_NUMBER
des-webana-datadictionary   Des WebAna DataDictionary   1038693078675
sit323-25t1-duong-102550c   sit323-25t1-duong-102550c   87055954794

C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud config get-value project
sit323-25t1-duong-102550c
```

Create an Artifact Registry Repository (AR Repo) on Google Cloud Platform:

1. Type in Search Box for "Artifact Repository" and enable this API.
2. Choose "Create repository" with the following selections:
    a. Name: "task10-1p-calculator"
    b. Format: "Docker"
    c. Mode: "Standard"
    d. Location Type: "Region"
        i. Region: "australia-southeast2 (Melbourne)"
    e. Leave other selections as their initial state and click "Create" button.



3. Grant the push-pull image from AR Repo to Docker by type in command:
   "gcloud auth configure-docker <Region_selected>-docker.pkg.dev"

4. Build a Docker image for the application through Dockerfile with its name matching the criteria for pushing to Docker Hub:

   "docker build <docker-hub_username>/<image_name>:<version>"

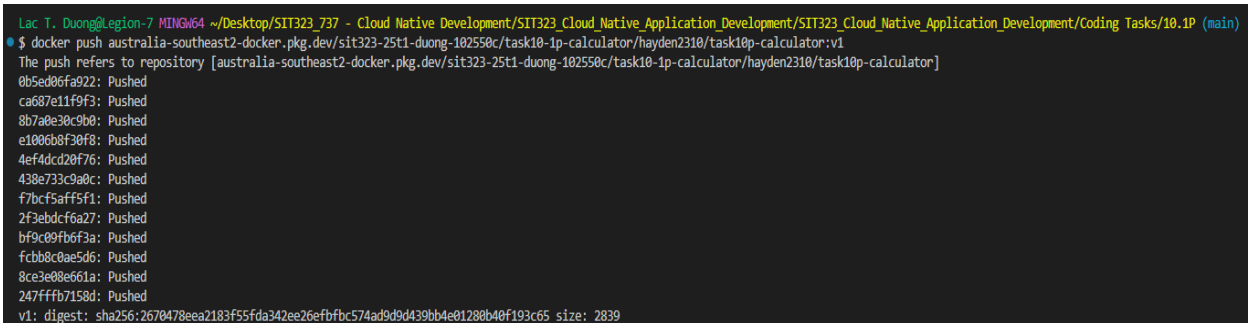
   "docker build hayden2310/task10-p-calculator:v1"

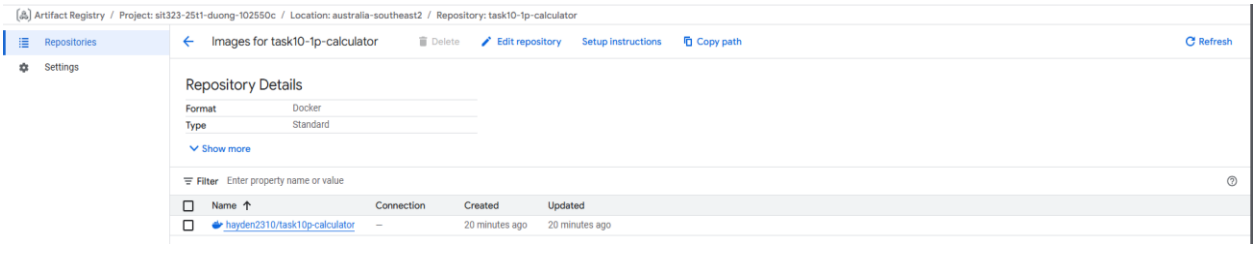
5. Push this image to Docker Hub through command: "docker push <image_name>:<version>

6. Create a new image from this image through the following command:

   "docker tag <image_name> <region_selected>-docker.pkg.dev/<project_id>/<AR_Repo_name>/<image_name>:<tag>"

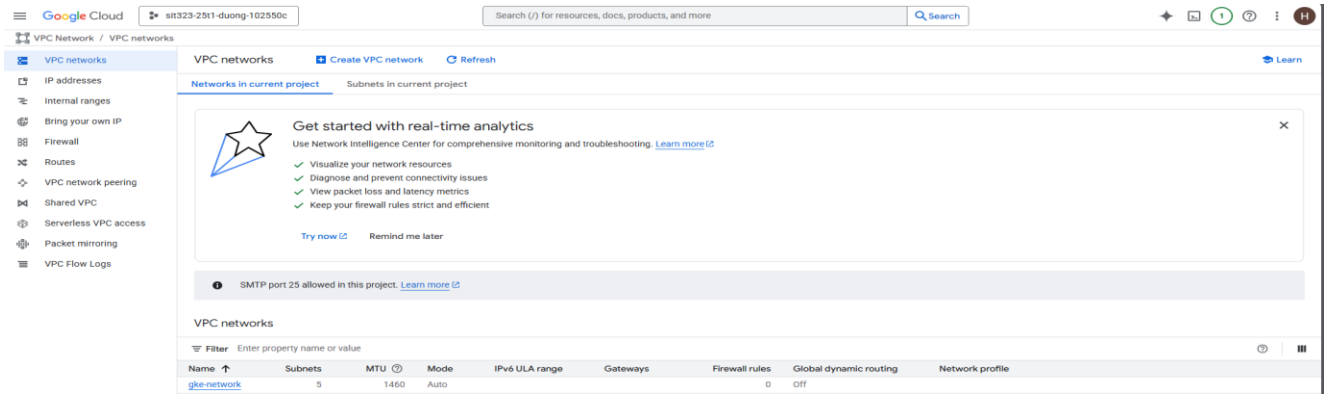
7. This newly created image is now push-able to AR Repo as its name has meet the requirement through command “docker push <image_name>

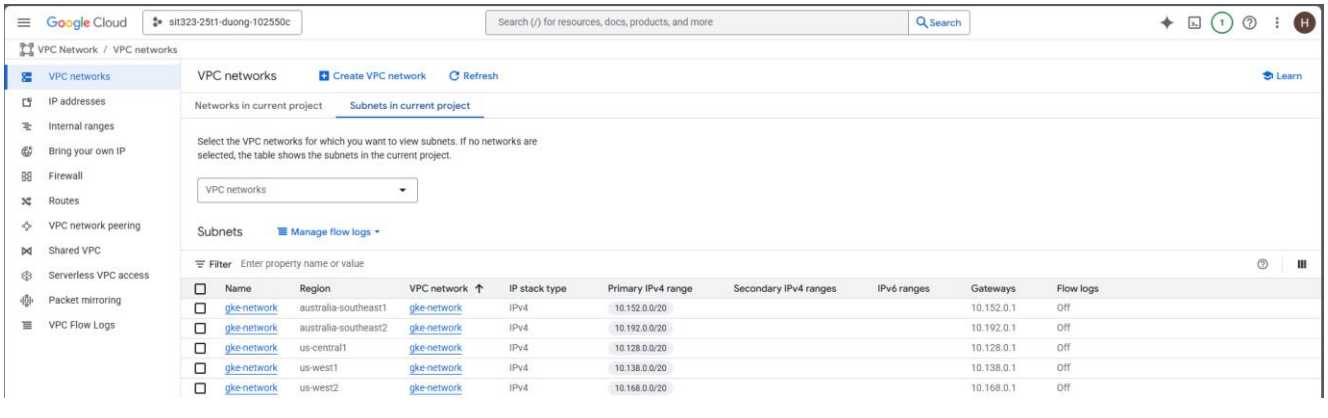




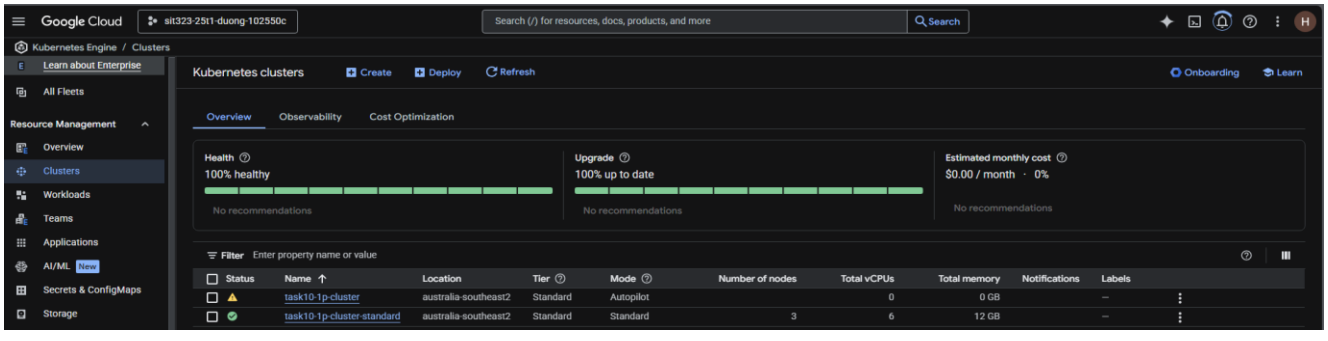
Create Google Kubernetes Engine (GKE) Cluster

1. Create a VPC network through command in Google Cloud Shell: “gcloud compute networks create gke-network –subnet-mode=auto”





2. Enable Kubernetes Engine API on Google Cloud Platform.
3. Create a cluster with the following:
    a. Switch to: “Standard Cluster”
    b. Name: “task10-1p-cluster-standard”
    c. Region: “australia-southeast2”
    d. Cluster tier: “Standard tier”
    e. In “default-pool”: Number of nodes = “1”
    f. Choose “Create” button



4. Enter this command into Google Cloud Shell:

gcloud container clusters get-credentials task10-1p-cluster-standard --region australia-southeast2 --project sit323-25t1-duong-102550c

```
C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud container clusters get-credentials task10-1p-cluster-standar
d --region australia-southeast2 --project sit323-25t1-duong-102550c
Fetching cluster endpoint and auth data.
kubeconfig entry generated for task10-1p-cluster-standard.
```

5. Check status:

```
$ kubectl config current-context
gke_sit323-25t1-duong-102550c_australia-southeast2_task10-1p-cluster-standard

Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_
323_Cloud_Native_Application_Development/Coding Tasks/10.1P/k8s (main)
$ kubectl get nodes
NAME                                                  STATUS   ROLES    AGE     VERSION
gke-task10-1p-cluster-st-default-pool-a704cec7-2m8j   Ready    <none>   2m11s   v1.32.2-gke.1297002
gke-task10-1p-cluster-st-default-pool-d7138cdc-h11m   Ready    <none>   2m10s   v1.32.2-gke.1297002
gke-task10-1p-cluster-st-default-pool-e9b57b92-vrdl   Ready    <none>   2m11s   v1.32.2-gke.1297002
```

Create Deployment.yaml & Service.yaml for the application:

1. Deployment.yaml:

deployment.yaml 1, U

k8s > deployment.yaml > ...

io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: task10-1p-calculator-deployment
  labels:
    app: task10-1p-calculator
spec:
  replicas: 1
  selector:
    matchLabels:
      app: task10-1p-calculator
  template:
    metadata:
      labels:
        app: task10-1p-calculator
    spec:
      containers:
      - name: task10-1p-calculator-container
        image: hayden2310/task10p-calculator:v1
        ports:
        - containerPort: 3040
```

2. Service.yaml:

service.yaml U

k8s > service.yaml > { } spec > type

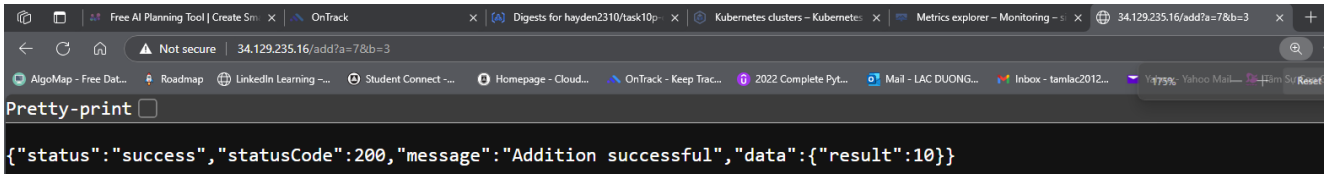
io.k8s.api.core.v1.Service (v1@service.json)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: task10-1p-calculator-service
  labels:
    app: task10-1p-calculator
spec:
  selector:
    app: task10-1p-calculator
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3040
  type: LoadBalancer
```

3. Apply these two YAML files and check their status:

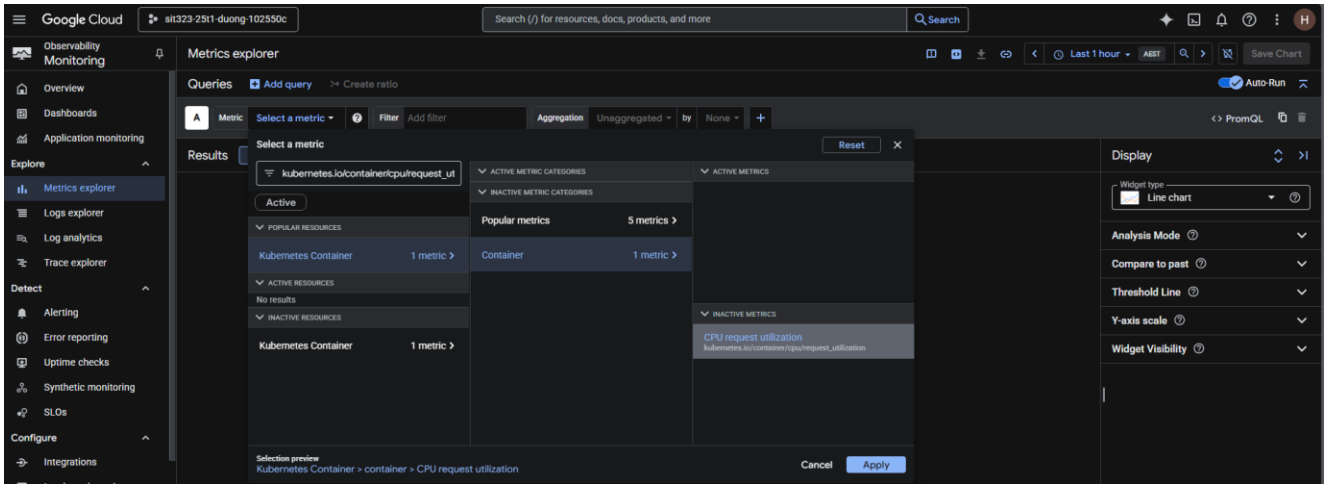
```
Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/SIT323_Cloud_Native_Application_Development/Coding Tasks/10.1P/k8s (main)
$ kubectl get deployment
NAME                              READY   UP-TO-DATE   AVAILABLE   AGE
task10-1p-calculator-deployment   1/1     1            1           5m12s

Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/SIT323_Cloud_Native_Application_Development/Coding Tasks/10.1P/k8s (main)
$ kubectl get pods
NAME                                               READY   STATUS    RESTARTS   AGE
task10-1p-calculator-deployment-7d8c977d6f-g8k4v   1/1     Running   0          2m16s

Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/SIT323_Cloud_Native_Application_Development/Coding Tasks/10.1P/k8s (main)
$ kubectl get svc
NAME                         TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)        AGE
kubernetes                   ClusterIP      34.118.224.1    <none>          443/TCP        28m
task10-1p-calculator-service LoadBalancer   34.118.233.103  34.129.235.16   80:31415/TCP   21m
```

4. Testing the web application through the external ip

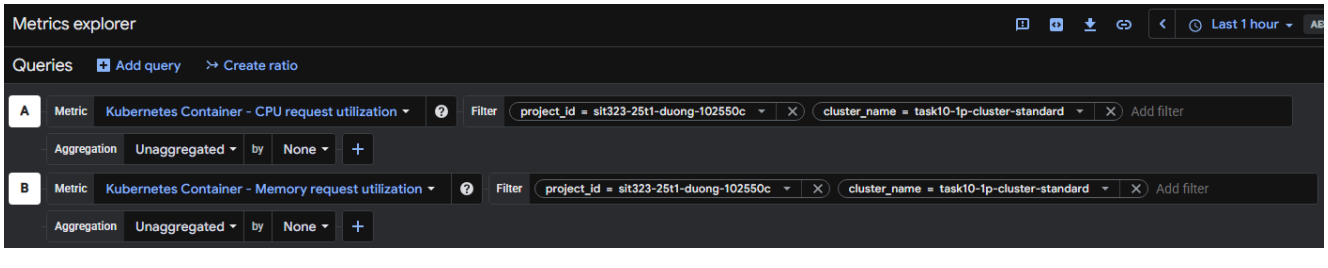


Setup the Monitoring in GCP

- Have to unclicked “Active” for these two metrics to be found:
  - kubernetes.io/container/cpu/request_utilization
  - kubernetes.io/container/memory/request_utilization

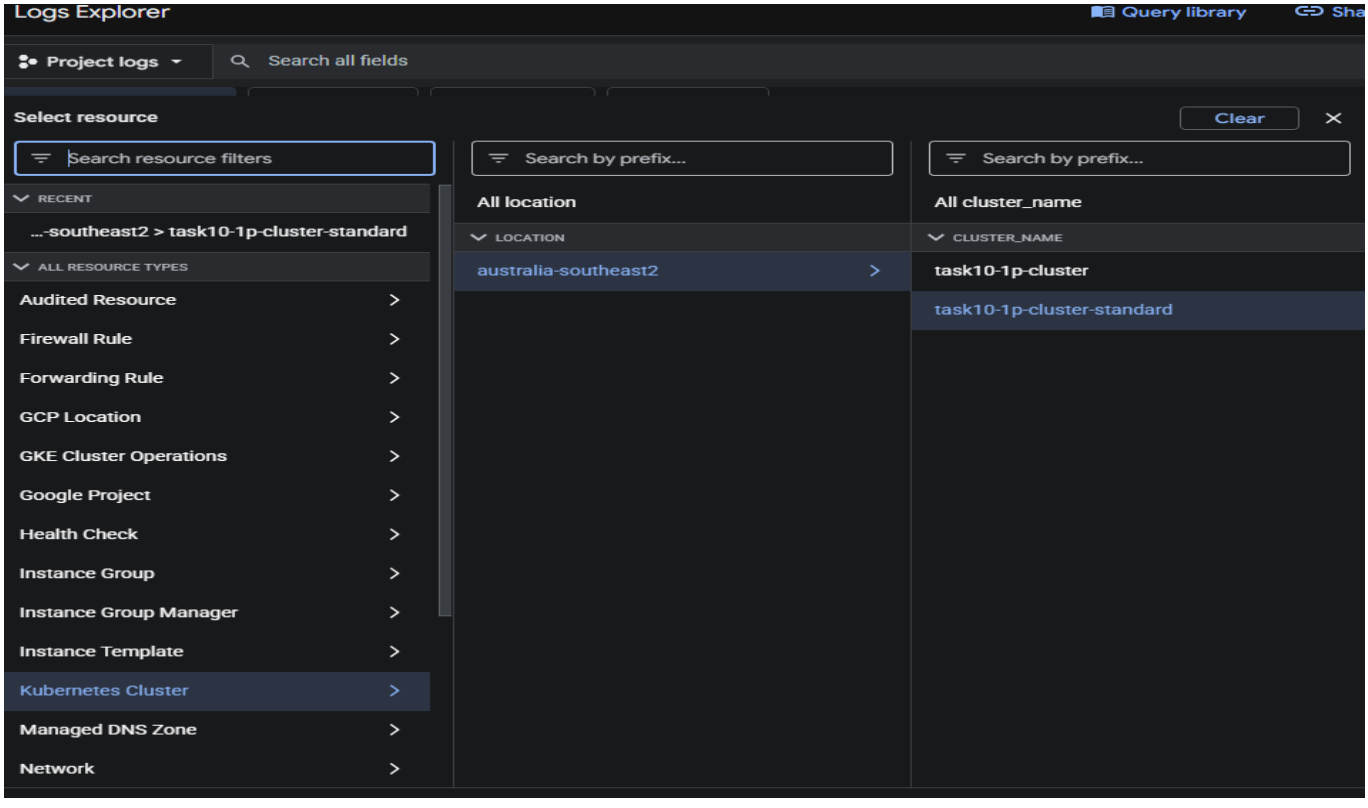


- For both metric, add as following images

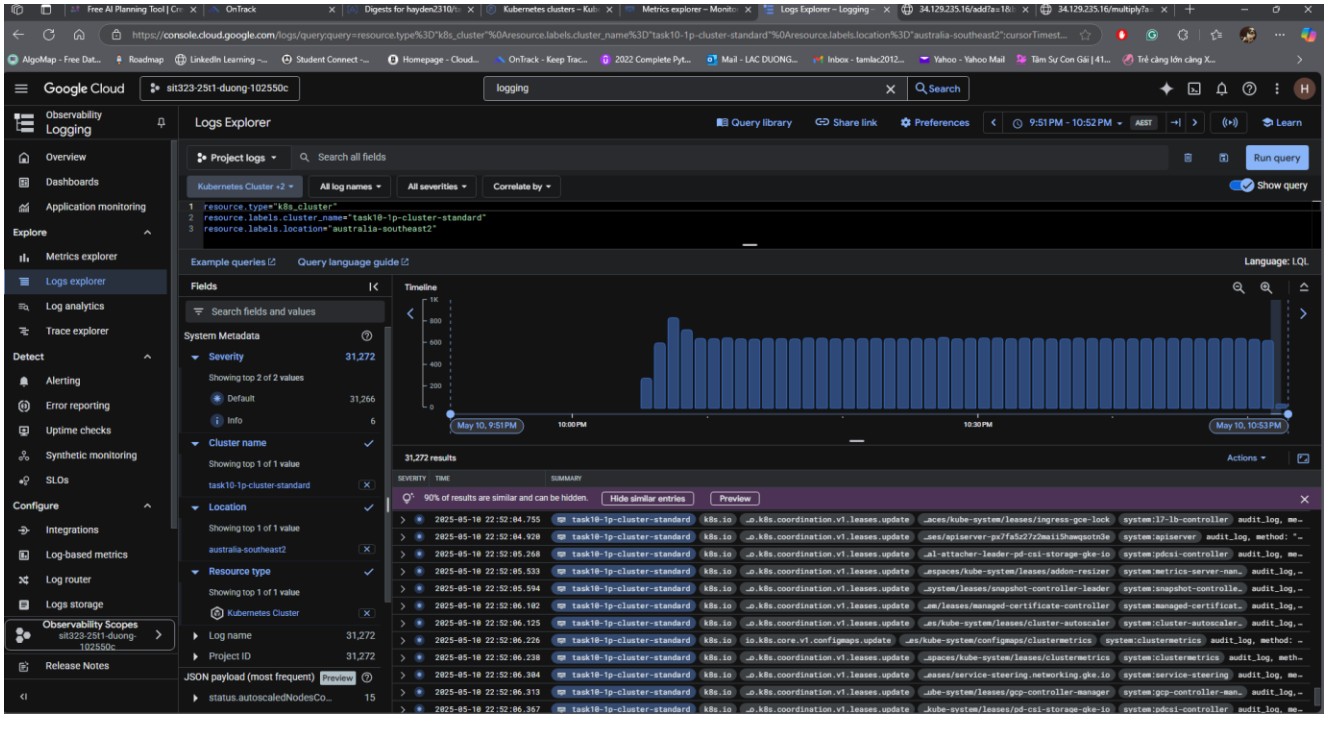


- Navigate to “Logging”-“Logs explorer”

- Apply the following:



Result:



C/ Challenges

- There was no network available for the creation of GKE cluster.
    - Manually created a VPC network called “gke network” by using command “gcloud compute networks create gke-network –subnet-mode=auto”
    - With this network, GKE cluster can now use it for create.
- Autopilot GKE cluster did not generate any node to be used for deployment with k8s.
    - Had to switched to Standard Cluster Mode and and set the number of nodes to “1”
    - 3 nodes were created and now be able to used for deployment
- Artifact Registry Repository does not allow image pulling from it to the local for deployment
    - Had to pull image from Docker Hub through fixing the image name in deployment.yaml
- From Microsoft Team, I found that some students are experiencing with no-shown metric on GCP platform, as result, I decided to use results generated from kubectl to substituate through command “kubectl top nodes” and “kubectl top pods” to display the percentage usage of CPU & Memory, as well as, the unit they are using in “cores” and “bytes”, correspondingly.