

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 account.
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”.

```
Welcome to the Google Cloud CLI! Run "gcloud -h" to get the list of available commands.
---
C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud auth login
Your browser has been opened to visit:

    https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=32555940559.apps.googleusercontent.com&redirect_uri=http%3A%2F%2Flocalhost%3A8085%2F&scope=openid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fappengine.admin+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fsqlservice.login+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcompute+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Faccounts.reauth&state=woUWU6JddW6byW2I1zSU9aBhAf9H4U&access_type=offline&code_challenge=xsDa_005g4Iyx_QTgH4bzXzZ8PZVCgT6K0q4kQYyTk0&code_challenge_method=S256

You are now logged in as [s222610226@deakin.edu.au].
Your current project is [sit323-25t1-duong-102550c]. You can change this setting by running:
$ gcloud config set project PROJECT_ID
```

```
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.

Repositories						
+ Create repository Edit repository Delete Setup instructions						
<div><div>Filter</div><div>Enter property name or value</div></div>						
<input type="checkbox"/>	Name ↑	Format	Type	Location	Scanning (Disabled) ⓘ	Description
<input type="checkbox"/>	calculator-repo	Docker	Standard	australia-southeast2 (Melbourne)	Disabled	Reposito... ▼
<input type="checkbox"/>	task10-1p-calculator	Docker	Standard	australia-southeast2 (Melbourne)	Disabled	Reposito... ▼

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>”

```

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 (main)
$ docker push australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/task10-1p-calculator/hayden2310/task10p-calculator:v1
The push refers to repository [australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/task10-1p-calculator/hayden2310/task10p-calculator]
0b5ed06fa922: Pushed
ca687e11f9f3: Pushed
8b7a0e30c9b0: Pushed
e1006b8f30f8: Pushed
4ef4dc20f76: Pushed
438e733c9a0c: Pushed
f7bcf5aff5f1: Pushed
2f3ebdcf6a27: Pushed
bf9c09fb6f3a: Pushed
fcb8c0ae5d6: Pushed
8ce3e08e661a: Pushed
247ffb7158d: Pushed
v1: digest: sha256:2670478eea2183f55fda342ee26efbfc574ad9d9d439bb4e01280b40f193c65 size: 2839

```

[A] Artifact Registry / Project: sit323-25t1-duong-102550c / Location: australia-southeast2 / Repository: task10-1p-calculator

Repositories Settings

Images for task10-1p-calculator Delete Edit repository Setup instructions Copy path Refresh

Repository Details

Format Docker
Type Standard
Show more

Filter Enter property name or value

<input type="checkbox"/>	Name ↑	Connection	Created	Updated
<input type="checkbox"/>	hayden2310/task10p-calculator	—	20 minutes ago	20 minutes ago

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”

Google Cloud sit323-25t1-duong-102550c Search (/) for resources, docs, products, and more

VPC Network / VPC networks

VPC networks Create VPC network Refresh Learn

Networks in current project Subnets in current project

Get started with real-time analytics
Use Network Intelligence Center for comprehensive monitoring and troubleshooting. Learn more

- Visualize your network resources
- Diagnose and prevent connectivity issues
- View packet loss and latency metrics
- Keep your firewall rules strict and efficient

Try now Remind me later

SMTP port 25 allowed in this project. Learn more

VPC networks

Filter Enter property name or value

Name ↑	Subnets	MTU	Mode	IPv6 ULA range	Gateways	Firewall rules	Global dynamic routing	Network profile
gke-network	5	1460	Auto			0	Off	

Google Cloud sit323-25t1-duong-102550c Search (/) for resources, docs, products, and more

VPC Network / VPC networks

VPC networks Create VPC network Refresh Learn

Networks in current project Subnets in current project

Select the VPC networks for which you want to view subnets. If no networks are selected, the table shows the subnets in the current project.

VPC networks

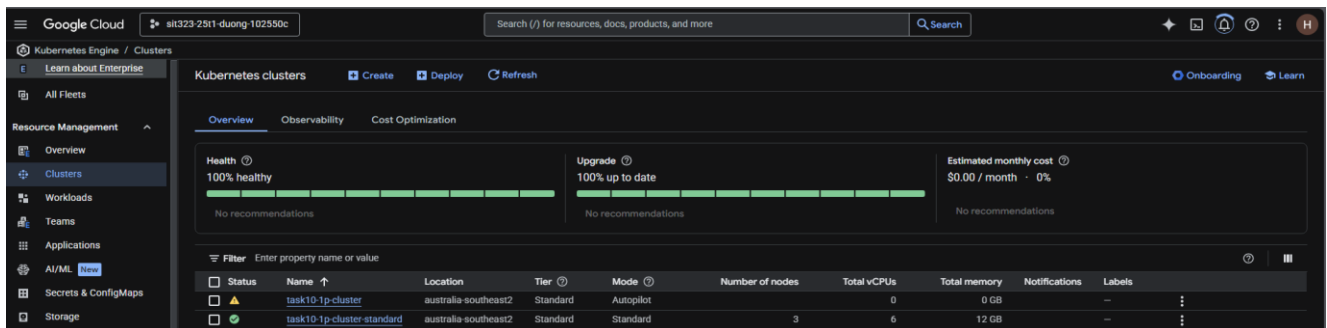
Subnets Manage flow logs

Filter Enter property name or value

<input type="checkbox"/>	Name	Region	VPC network ↑	IP stack type	Primary IPv4 range	Secondary IPv4 ranges	IPv6 ranges	Gateways	Flow logs
<input type="checkbox"/>	gke-network	australia-southeast1	gke-network	IPv4	10.152.0.0/20			10.152.0.1	Off
<input type="checkbox"/>	gke-network	australia-southeast2	gke-network	IPv4	10.192.0.0/20			10.192.0.1	Off
<input type="checkbox"/>	gke-network	us-central1	gke-network	IPv4	10.128.0.0/20			10.128.0.1	Off
<input type="checkbox"/>	gke-network	us-west1	gke-network	IPv4	10.138.0.0/20			10.138.0.1	Off
<input type="checkbox"/>	gke-network	us-west2	gke-network	IPv4	10.168.0.0/20			10.168.0.1	Off

2. Enable Kubernetes Engine API on Google Cloud Platform.
3. Create a cluster with the following:

- Switch to: “Standard Cluster”
- Name: “task10-1p-cluster-standard”
- Region: “australia-southeast2”
- Cluster tier: “Standard tier”
- In “default-pool”: Number of nodes = “1”
- 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-standard --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 X
k8s > deployment.yaml > ...
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: task10-1p-calculator-deployment
5    labels:
6      app: task10-1p-calculator
7  spec:
8    replicas: 1
9    selector:
10     matchLabels:
11       app: task10-1p-calculator
12   template:
13     metadata:
14       labels:
15         app: task10-1p-calculator
16     spec:
17       containers:
18         - name: task10-1p-calculator-container
19           image: hayden2310/task10p-calculator:v1
20           ports:
21             - containerPort: 3040

```

2. Service.yaml:

```

service.yaml U X
k8s > service.yaml > {} spec > type
io.k8s.api.core.v1.Service (v1@service.json)
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: task10-1p-calculator-service
5    labels:
6      app: task10-1p-calculator
7  spec:
8    selector:
9      app: task10-1p-calculator
10   ports:
11     - protocol: TCP
12       port: 80
13       targetPort: 3040
14   type: LoadBalancer

```

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

```

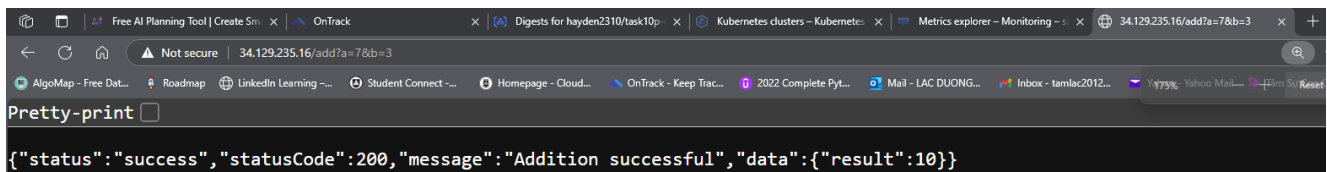
Lac T. Duong@legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/SIT323_Cloud_Native_Application_Development/Coding_Tasks/10.1P/k8s (main)
$ kubectl get deployment
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
task10-1p-calculator-deployment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

```

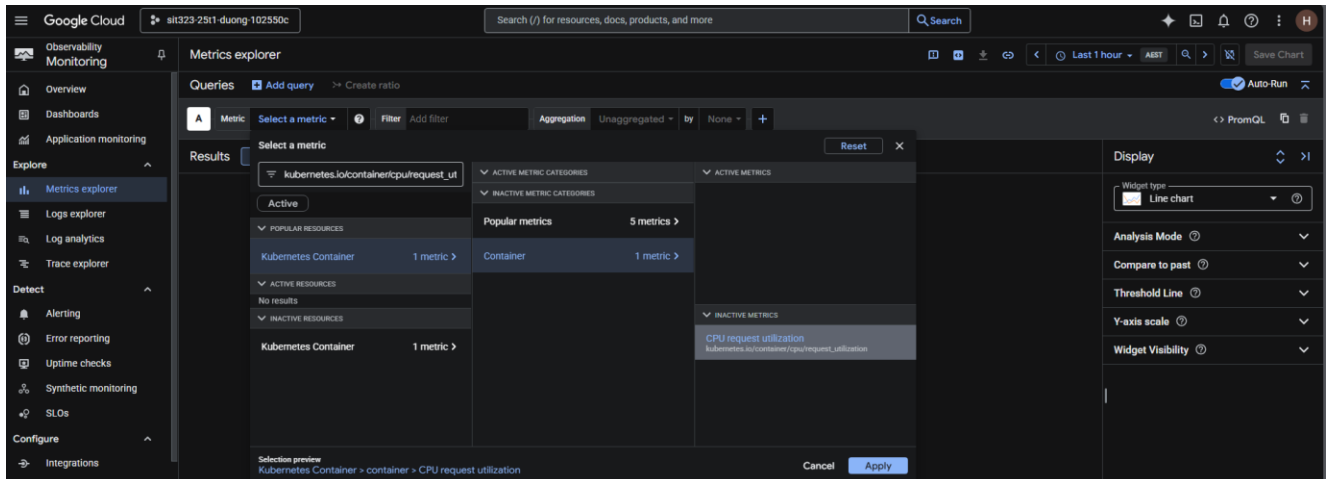
4. Testing the web application through the external ip



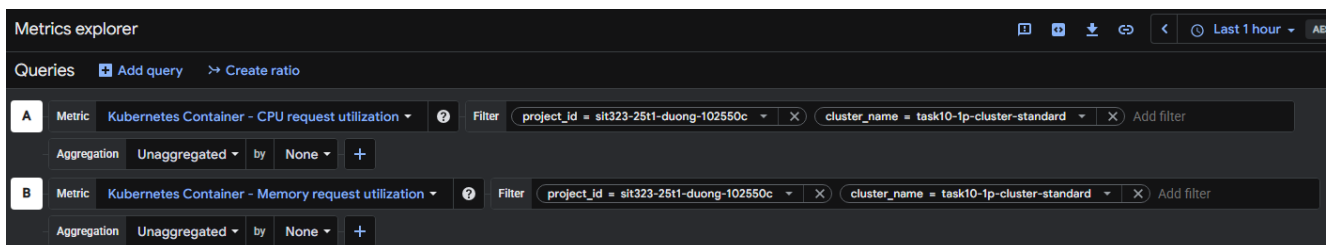
The screenshot shows a web browser window with the address bar displaying `34.129.235.16/add?a=7&b=3`. The page content shows a JSON response: `{"status": "success", "statusCode": 200, "message": "Addition successful", "data": {"result": 10}}`. The browser's developer tools are open, showing the response in the console.

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`

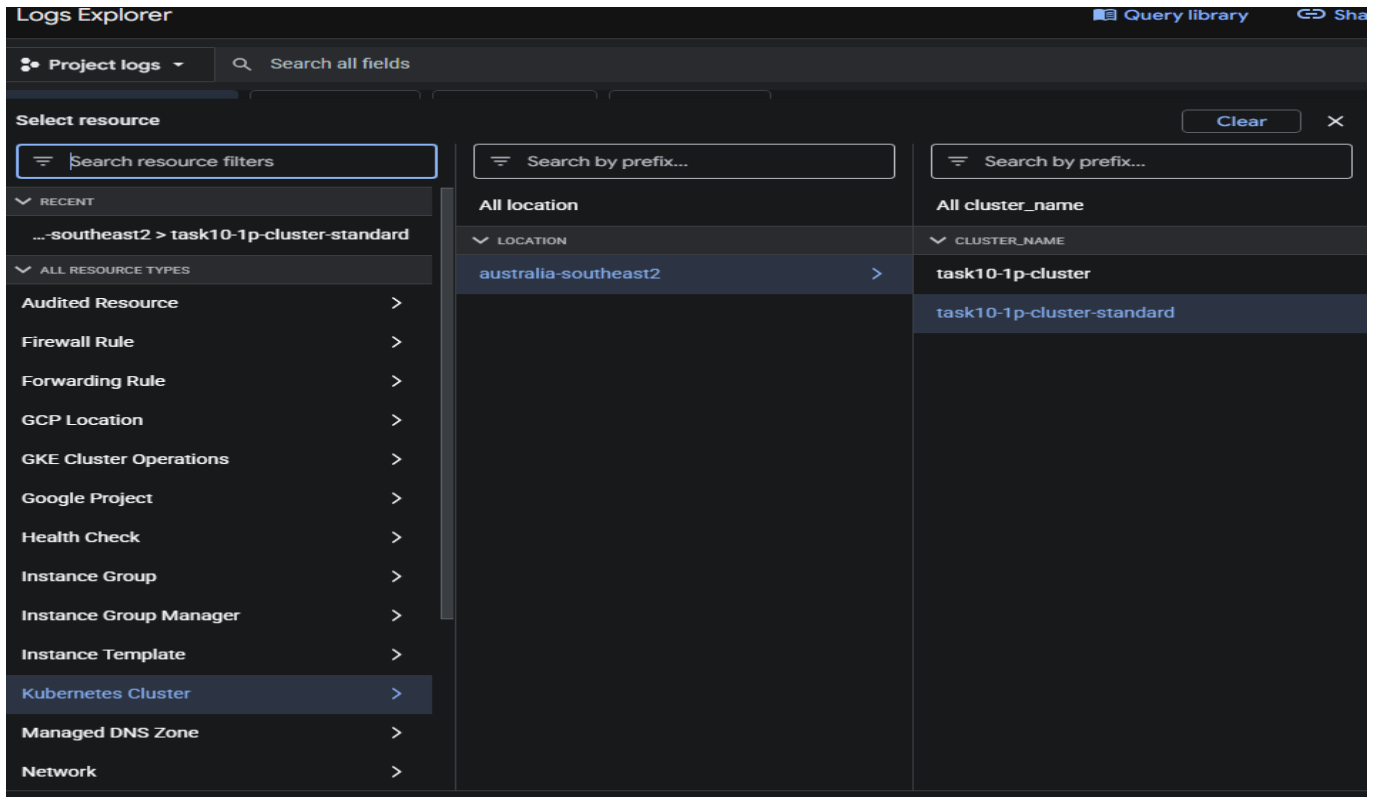


- For both metric, add as following images

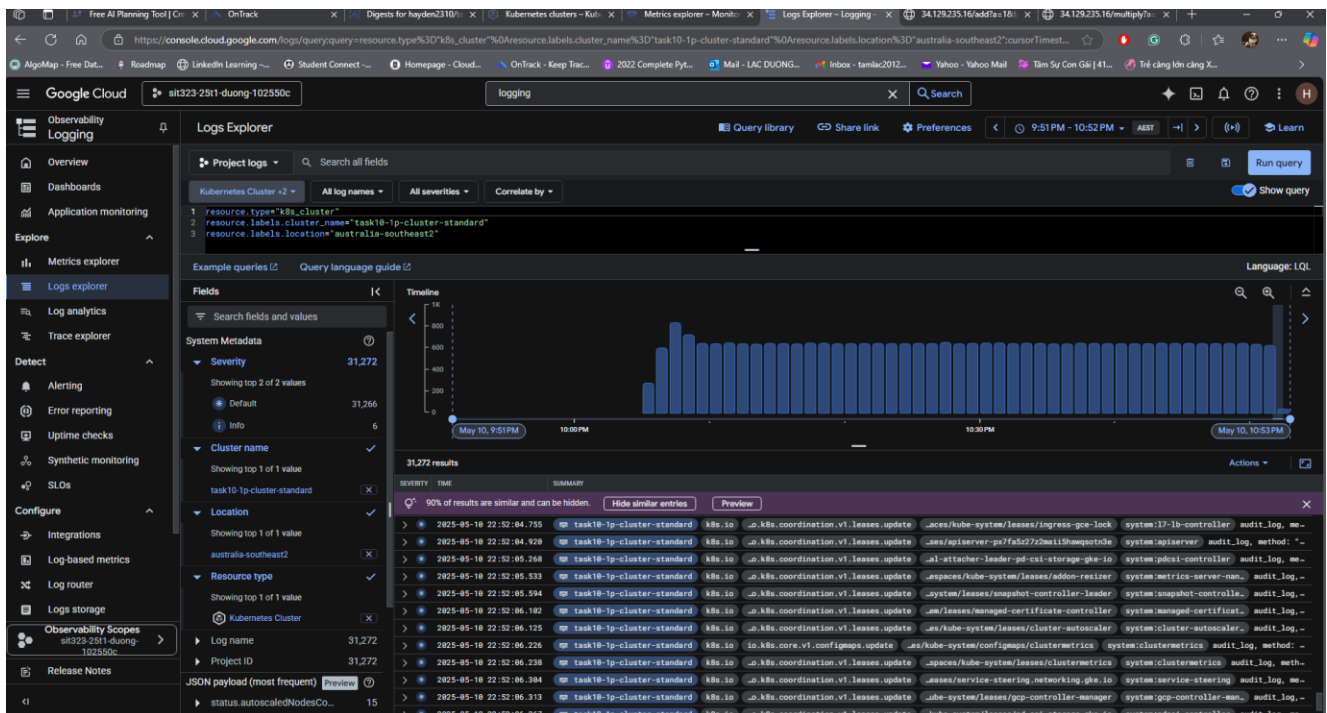


- Navigate to “Logging”-“Logs explorer”

- 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