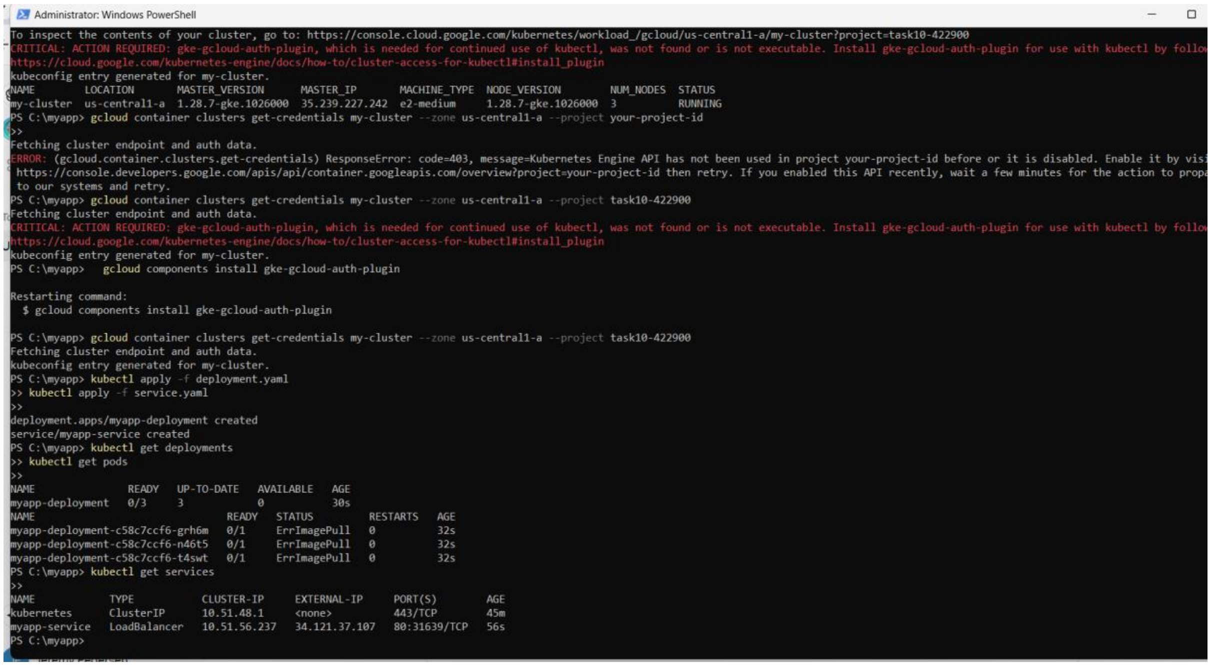
Additional to Task 10.1 (got there in the end …)

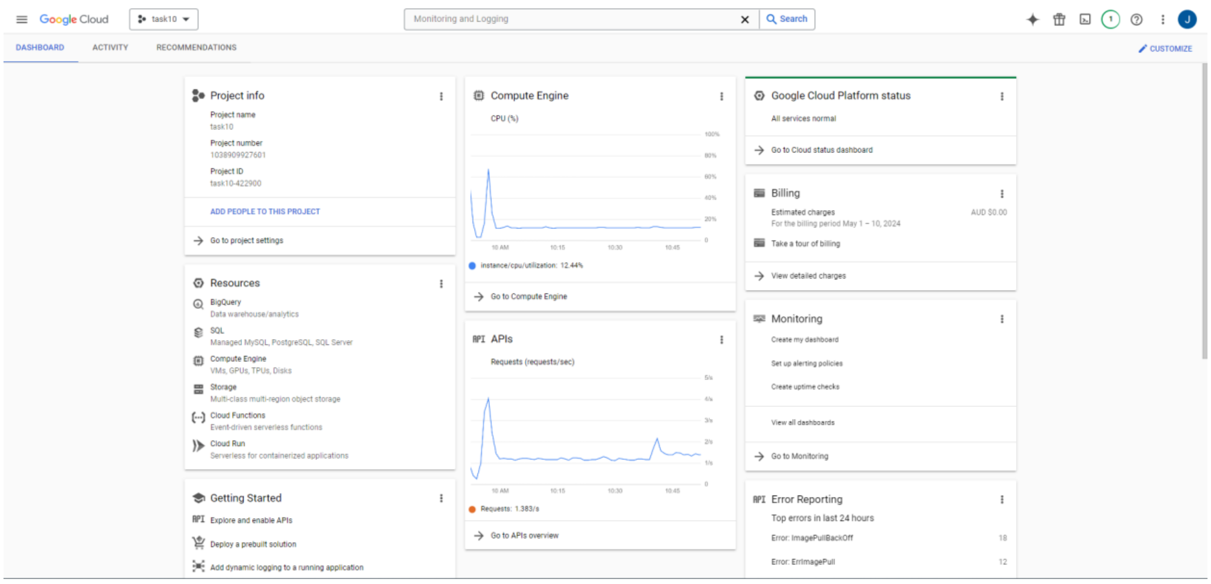


```
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-central1-a/my-cluster?project=task10-422900
CRITICAL: ACTION REQUIRED: gke-gcloud-auth-plugin, which is needed for continued use of kubectl, was not found or is not executable. Install gke-gcloud-auth-plugin for use with kubectl by follo
https://cloud.google.com/kubernetes-engine/docs/how-to/cluster-access-for-kubectl#install_plugin
kubeconfig entry generated for my-cluster.
NAME        LOCATION       MASTER_VERSION      MASTER_IP       MACHINE_TYPE  NODE_VERSION        NUM_NODES  STATUS
my-cluster  us-central1-a  1.28.7-gke.1026000  35.239.227.242  e2-medium     1.28.7-gke.1026000  3          RUNNING
PS C:\myapp> gcloud container clusters get-credentials my-cluster --zone us-central1-a --project your-project-id
>>
Fetching cluster endpoint and auth data.
ERROR: (gcloud.container.clusters.get-credentials) ResponseError: code=403, message=Kubernetes Engine API has not been used in project your-project-id before or it is disabled. Enable it by visi
 https://console.developers.google.com/apis/api/container.googleapis.com/overview?project=your-project-id then retry. If you enabled this API recently, wait a few minutes for the action to propa
 to our systems and retry.
PS C:\myapp> gcloud container clusters get-credentials my-cluster --zone us-central1-a --project task10-422900
Fetching cluster endpoint and auth data.
CRITICAL: ACTION REQUIRED: gke-gcloud-auth-plugin, which is needed for continued use of kubectl, was not found or is not executable. Install gke-gcloud-auth-plugin for use with kubectl by follo
https://cloud.google.com/kubernetes-engine/docs/how-to/cluster-access-for-kubectl#install_plugin
kubeconfig entry generated for my-cluster.
PS C:\myapp>   gcloud components install gke-gcloud-auth-plugin

Restarting command:
  $ gcloud components install gke-gcloud-auth-plugin

PS C:\myapp> gcloud container clusters get-credentials my-cluster --zone us-central1-a --project task10-422900
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster.
PS C:\myapp> kubectl apply -f deployment.yaml
>> kubectl apply -f service.yaml
>>
deployment.apps/myapp-deployment created
service/myapp-service created
PS C:\myapp> kubectl get deployments
>> kubectl get pods
>>
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
myapp-deployment   0/3     3            0           30s
NAME                               READY   STATUS         RESTARTS   AGE
myapp-deployment-c58c7ccf6-grh6m   0/1     ErrImagePull   0          32s
myapp-deployment-c58c7ccf6-n46t5   0/1     ErrImagePull   0          32s
myapp-deployment-c58c7ccf6-t4swt   0/1     ErrImagePull   0          32s
PS C:\myapp> kubectl get services
>>
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
kubernetes      ClusterIP      10.51.48.1     <none>          443/TCP        45m
myapp-service   LoadBalancer   10.51.56.237   34.121.37.107   80:31639/TCP   56s
PS C:\myapp>
```

Overview



Project info

Project name
task10

Project number
1038909927601

Project ID
task10-422900

ADD PEOPLE TO THIS PROJECT

Go to project settings

Compute Engine

CPU (%)

● instance/cpu/utilization: 12.44%

Go to Compute Engine

Google Cloud Platform status

All services normal

Go to Cloud status dashboard

Billing

Estimated charges AUD $0.00
For the billing period May 1 – 10, 2024

Take a tour of billing

View detailed charges

Resources

BigQuery
Data warehouse/analytics

SQL
Managed MySQL, PostgreSQL, SQL Server

Compute Engine
VMs, GPUs, TPUs, Disks

Storage
Multi-class multi-region object storage

Cloud Functions
Event-driven serverless functions

Cloud Run
Serverless for containerized applications

APIs

Requests (requests/sec)

● Requests: 1.383/s

Go to APIs overview

Monitoring

Create my dashboard

Set up alerting policies

Create uptime checks

View all dashboards

Go to Monitoring

Getting Started

Explore and enable APIs

Deploy a prebuilt solution

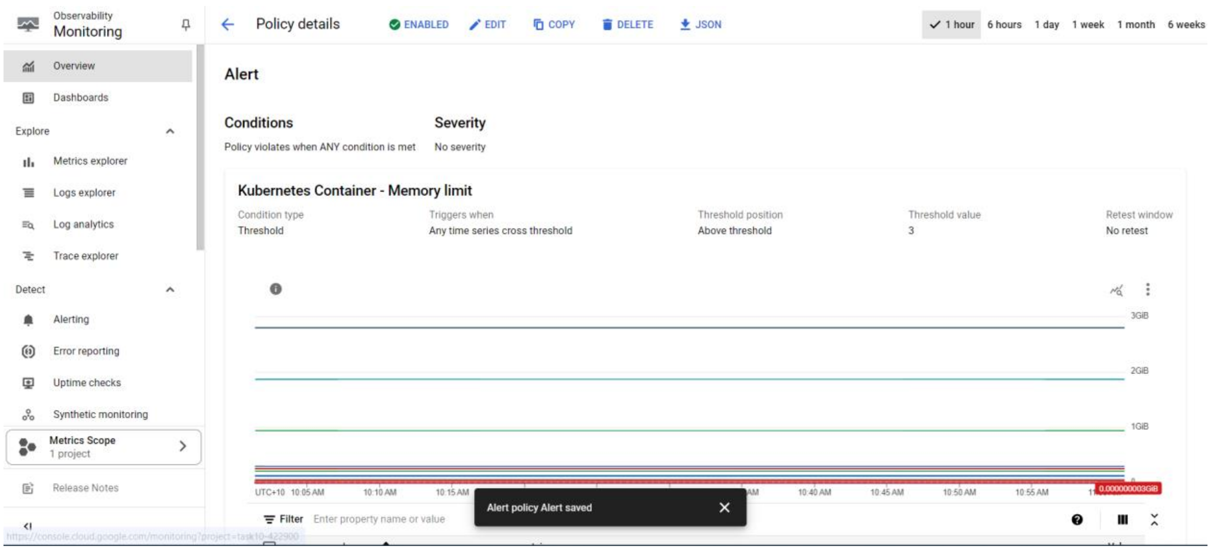
Add dynamic logging to a running application

Error Reporting

Top errors in last 24 hours

Error: ImagePullBackOff	18
Error: ErrImagePull	12

Alerts



Alert

Conditions
Policy violates when ANY condition is met

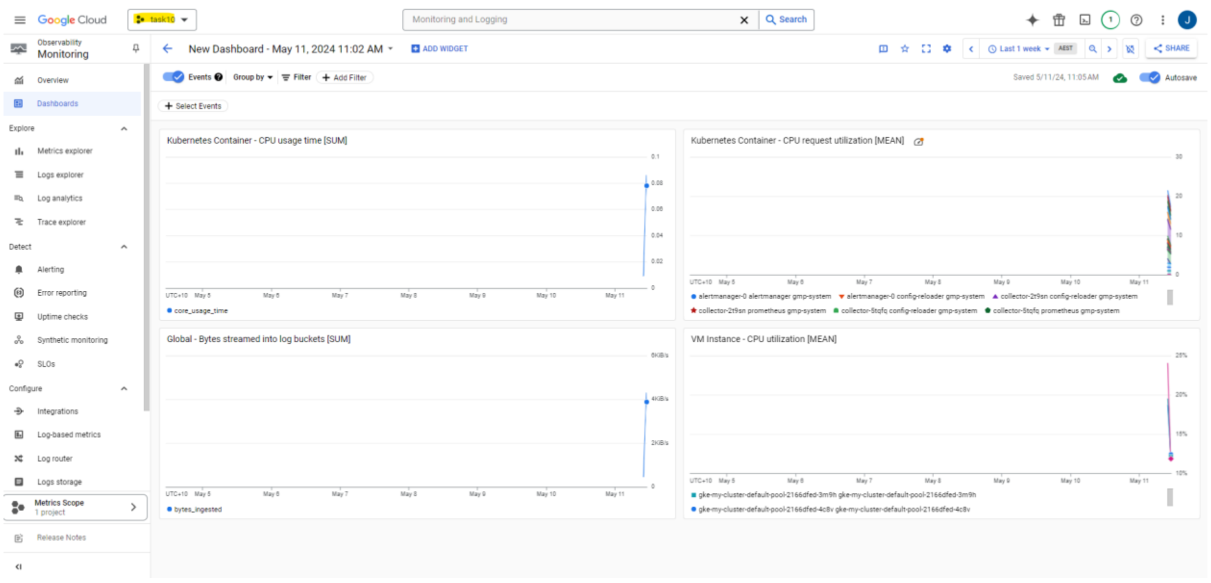
Severity
No severity

Kubernetes Container - Memory limit

Condition type	Triggers when	Threshold position	Threshold value	Retest window
Threshold	Any time series cross threshold	Above threshold	3	No retest

Alert policy Alert saved

Monitoring in Google Cloud


Google Cloud
task10
Monitoring and Logging
Search
Observability
Monitoring
Overview
Dashboards
Explore
Metrics explorer
Logs explorer
Log analytics
Trace explorer
Detect
Alerting
Error reporting
Uptime checks
Synthetic monitoring
SLOs
Configure
Integrations
Log-based metrics
Log router
Logs storage
Metrics Scope
1 project
Release Notes
New Dashboard - May 11, 2024 11:02 AM
ADD WIDGET
Last 1 week
AEST
SHARE
Events
Group by
Filter
Add Filter
Saved 5/11/24, 11:05 AM
Autosave
Select Events
Kubernetes Container - CPU usage time [SUM]
core_usage_time
Kubernetes Container - CPU request utilization [MEAN]
alertmanager-0 alertmanager gmp-system
alertmanager-0 config-reloader gmp-system
collector-2t9sn config-reloader gmp-system
collector-2t9sn prometheus gmp-system
collector-5tqfq config-reloader gmp-system
collector-5tqfq prometheus gmp-system
Global - Bytes streamed into log buckets [SUM]
bytes_ingested
VM Instance - CPU utilization [MEAN]
gke-my-cluster-default-pool-2166dfed-3m9h gke-my-cluster-default-pool-2166dfed-3m9h
gke-my-cluster-default-pool-2166dfed-4c8v gke-my-cluster-default-pool-2166dfed-4c8v

Development and Local Testing:

1. **Developed a simple Node.js application** that runs a primary web server using Express.js. This server responds with "Hello World!" to HTTP requests.
2. **Containerized the Node.js application** by creating a Docker file. This Docker file defines the environment in which the Node.js app runs, including the base image, working directory, dependencies installation, and the command to start the application.
3. **Tested the Docker container locally** to ensure the application runs correctly inside a container.

Kubernetes Deployment on GCP:

1. **Created Kubernetes deployment and service configurations**:
    - The deployment configuration (deployment.yaml) specifies the application's desired state, including the Docker image to use and the number of replicas.
    - The service configuration (service.yaml) defines how the application pods are accessed through a Load Balancer service, which exposes the application to the internet.
2. **Deployed the application to a GCP Kubernetes cluster**:
    - Configured the **gcloud** and **kubectl** command-line tools to interact with GCP and the Kubernetes cluster.
    - Built and pushed the Docker image to the Google Container Registry (GCR).
    - Applied the Kubernetes configurations to launch the application in the cluster.
3. **Verified the deployment** by checking the status of deployments and services in the Kubernetes cluster, ensuring the application is running and accessible via an external IP.

Monitoring and Logging:

1. **Set up Google Cloud's operations suite** for monitoring and logging:
    - Enabled and configured Google Cloud Monitoring and Logging for the project.
    - Deployed monitoring agents in the Kubernetes cluster to collect detailed metrics and logs.
    - Configured dashboards for real-time monitoring of the application and infrastructure health.
    - Set up alerts about critical conditions that might impact the application.