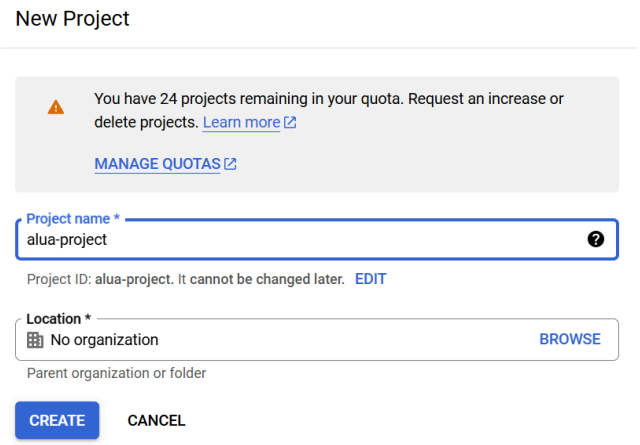
Assignment 3, Cloud Computing

1. Identity and Security Management

Exercise 1: Setting Up IAM Roles

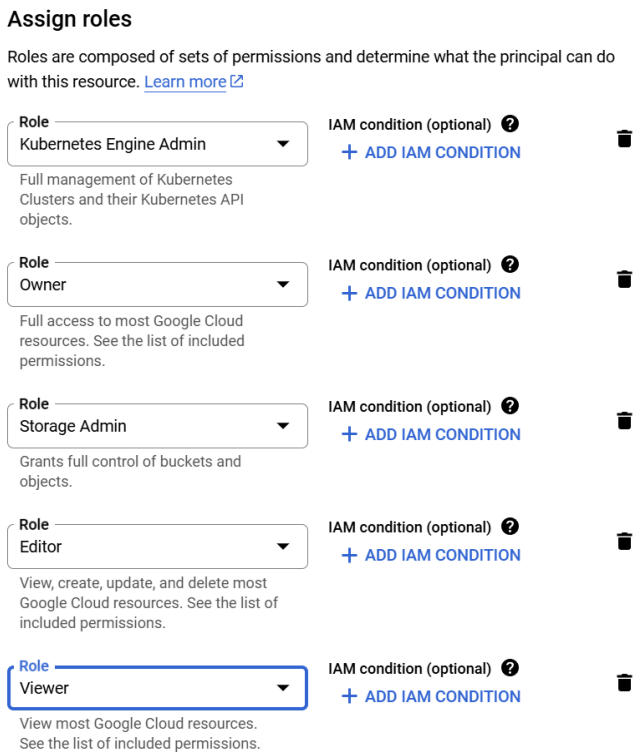
- *Create a new Google Cloud project.*

As the first step, I created a new project called ‘alua-project’.



- *Set up Identity and Access Management (IAM) roles for different team members (e.g., Viewer, Editor, Owner).*

As there are no other team members, I am assigning myself with these roles, such as Kubernetes Engine and Storage admins, and roles Viewer, Editor etc.



- *Assign specific roles to users and document the permissions associated with each role.*

A Viewer is usually a person that cannot make any changes in a project, but only ‘view’ what is happening. Monitoring logs, review performance and observing resource status.

Editor is a person who can make changes, usually developers.

Owner has full access over the project, can manage billing and access permissions.

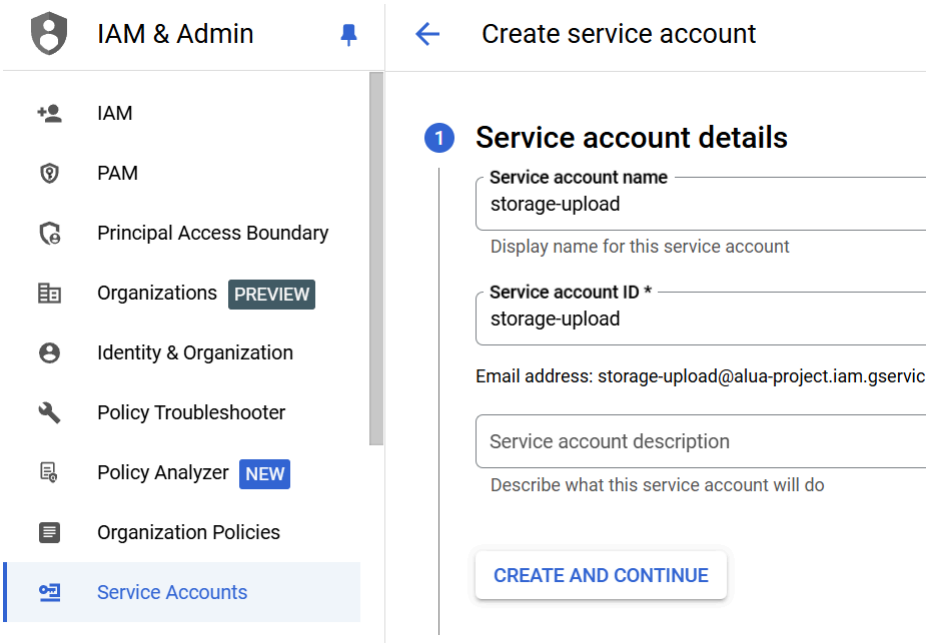
Kubernetes Engine Admin - allows full access to manage Kubernetes resources.

Storage Admin - grants full access to manage Cloud Storage resources.

Exercise 2: Service Accounts

- Create a service account that can access Google Cloud Storage.

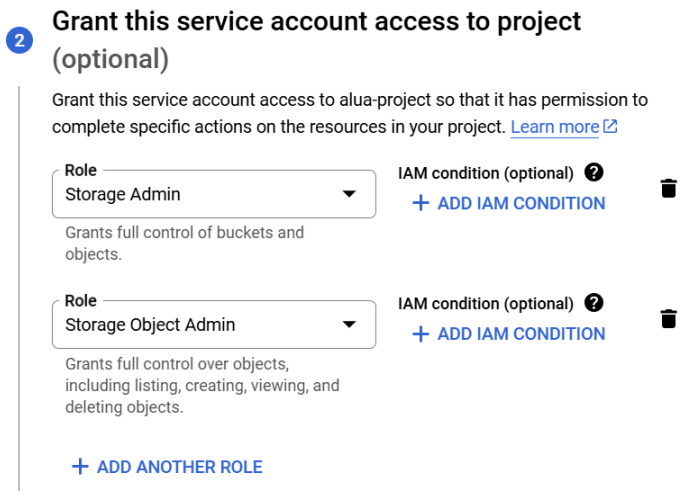
In the ‘IAM & Admin section’’ I choose ‘Service Accounts’ and create a new service account.



Granting this account with roles, such as Storage and Storage Object Admin.

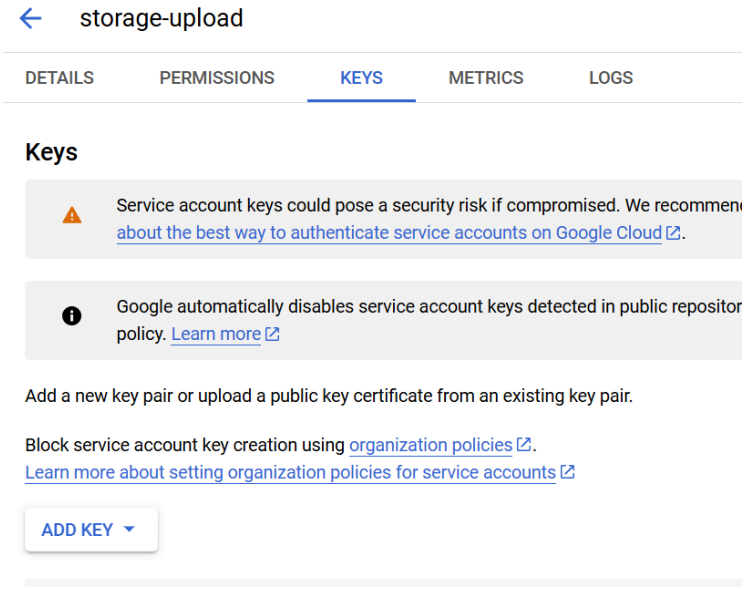
Their definitions can be seen in the picture.

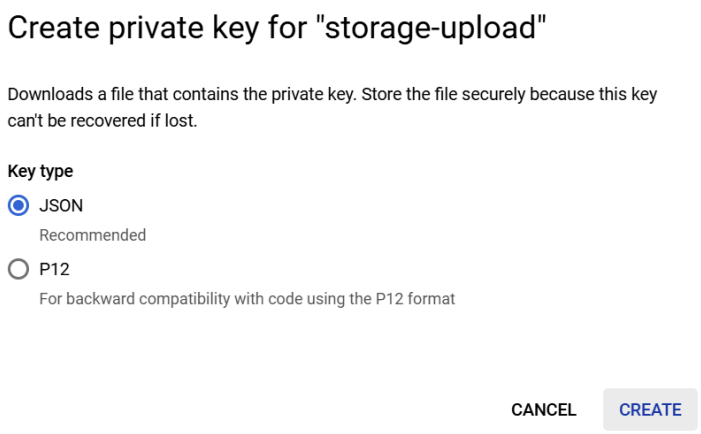
Storage Admin has full access over buckets and objects, Object Admin has a full access only over Objects within buckets.



- Generate and download a key for this service account, and use it to authenticate a Python script that uploads a file to a Cloud Storage bucket.

So in the 'Keys' sections I press a button to create new keys in the JSON format. They are used to authenticate the service account in a python script for secure access to Storage.





First of all, I created a bucket called ‘alua-bucket’ for testing, and will upload a file here.

So this is a simple python code that helps to upload files to Cloud Storage.

As a credential I am choosing this JSON file that I previously downloaded.

Then in a **file_path** I am defining a file that I want to upload into Google Storage (my local file).

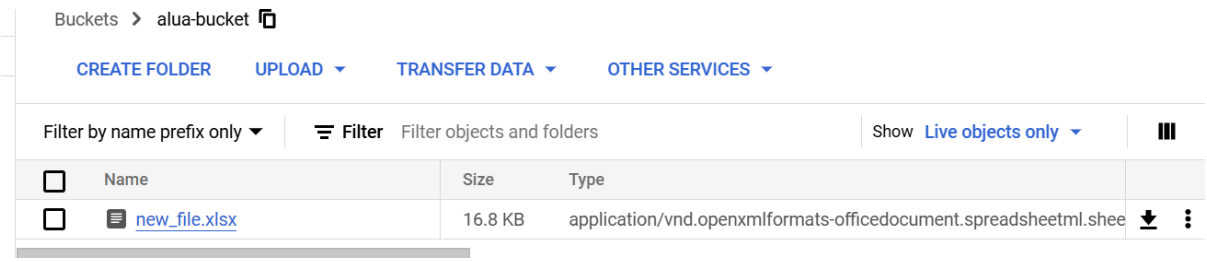
And **destination_blob_name** will be a new name of an uploaded file.

```
app.py > ...
from google.cloud import storage
import os

os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "C:/Users/onaye/Downloads/alua-project-eae0bd536ff8.json"


client = storage.Client()


bucket_name = "alua-bucket"
file_path = "C:/Users/onaye/Downloads/grades.xlsx"
destination_blob_name = "new_file.xlsx"

bucket = client.bucket(bucket_name)
blob = bucket.blob(destination_blob_name)
blob.upload_from_filename(file_path)

print(f"File {file_path} uploaded to {destination_blob_name}.")
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\onaye\Downloads\cloud_assign3>
& C:/Users/onaye/AppData/Local/Programs/Python/Python311/python.exe c:/Users/onaye/Downloads/cloud_assign3/app.py
File C:/Users/onaye/Downloads/grades.xlsx uploaded to new_file.xlsx.
```

Run code above, and then open my bucket to see whether the file was downloaded.

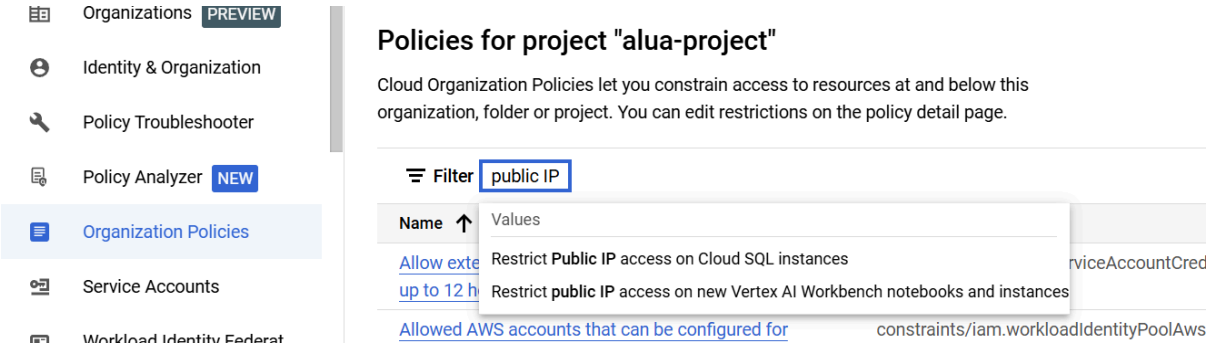
And as it can be seen, the excel file is uploaded with a new name.



Exercise 3: Organization Policies

- Explore organization policies by applying a restriction (e.g., disabling the creation of public IP addresses) at the organization or project level.

In the ‘Organization Policies’ section, I am searching for something like public IP. In order to disable public IP creation for resources for security improvement.



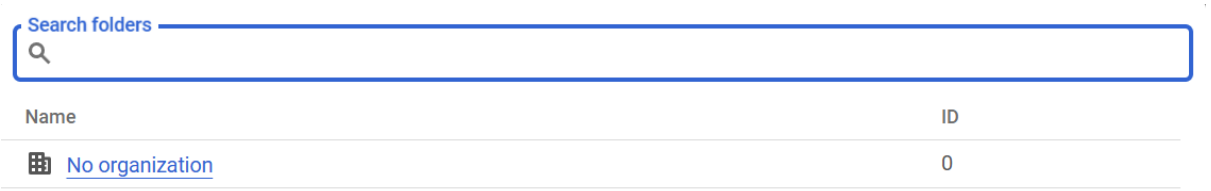
- Document the steps to enforce and view the organization policy settings.

Encountered an error while applying policies, as the project was not associated with any organization.

‘You need to select a resource under an organization to manage policies.’

Facing error like this:

So when I was creating a project, I didn't have any possibility to choose an organization.



But policy can be applied by the way I described earlier in the first step.

---

2. Google Kubernetes Engine (GKE)

Exercise 4: Deploying a Simple Application

- Set up a GKE cluster using the Google Cloud Console or `gcloud` command line.

In google Google Cloud SDK Shell I will create a cluster with command line

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud container clusters create hello-world-cluster --num-nodes=2 --zone=us-cen
tral1-a
Note: The Kubelet readonly port (10255) is now deprecated. Please update your workloads to use the recommended alternati
ves. See https://cloud.google.com/kubernetes-engine/docs/how-to/disable-kubelet-readonly-port for ways to check usage an
d for migration instructions.
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster hello-world-cluster in us-central1-a... Cluster is being configured...⠹
```

Status	Name	Location	Number of nodes	Total vCPUs	Total memory
✓	hello-world-cluster	us-central1-a	2	4	8 GB

Installing this plugin as an administrator

```
gcloud components install gke-gcloud-auth-plugin
```

```
╔═══ Downloading: gke-gcloud-auth-plugin ═══╗
╔═══ Downloading: gke-gcloud-auth-plugin (Platform Specific) ═══╗
╔═══ Installing: gke-gcloud-auth-plugin ═══╗
╔═══ Installing: gke-gcloud-auth-plugin (Platform Specific) ═══╗
```

- Deploy a simple containerized application (e.g., a Hello World app) to the cluster and expose it via a LoadBalancer service.

Deployed an app using .yaml file

```
kubectl apply -f
C:\Users\onaye\Downloads\cloud_assign3_2\hello-world-deployment.yaml
```

app.py | Dockerfile | ! hello-world-deployment.yaml 1

! hello-world-deployment.yaml > {} spec > [ ] ports > {} 0

io.k8s.api.core.v1.Service (v1@service.json) | io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world
```

```
C:\Program Files (x86)\Google\Cloud SDK>kubectl apply -f C:\Users\onaye\Downloads\cloud_assign3_2\hello-world-deployment
.yaml
deployment.apps/hello-world created
service/hello-world-service created
```

Confirmed the service with the command below, obtaining an external IP address to access the app

```
kubectl get services
```

34.55.150.188

```
C:\Program Files (x86)\Google\Cloud SDK>kubectl get services
NAME                  TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)        AGE
hello-world-service   LoadBalancer   34.118.234.60   34.55.150.188   80:30184/TCP   115s
kubernetes            ClusterIP      34.118.224.1    <none>          443/TCP        26m
```

Exercise 5: Managing Pods and Deployments

- Create a Deployment for a multi-container application using Kubernetes YAML files.

Deployed a multi-container app by defining 2 replicas in the .yaml file

```
kubectl apply -f
C:\Users\onaye\Downloads\cloud_assign3_2\multi-container-deployment.yaml
```

app.py | Dockerfile | ! hello-world-deployment.yaml 1 | ! multi-container-deployment.yaml 1 ×

! multi-container-deployment.yaml > {} spec > {} template > {} spec > [ ] containers > {} 1 > [ ] ports

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

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: multi-container-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: multi-container-app
  template:
    metadata:
```

```
C:\Program Files (x86)\Google\Cloud SDK>kubectl apply -f C:\Users\onaye\Downloads\cloud_assign3_2\multi-container-deploy
ment.yaml
deployment.apps/multi-container-app created
```

- Scale the Deployment to manage the number of replicas and update the application with a new container image.

Here I am changing the number of replicas with 3 instead of 2 with the scale command.

kubectl scale deployment multi-container-app --replicas=3

```
C:\Program Files (x86)\Google\Cloud SDK>kubectl scale deployment multi-container-app --replicas=3
deployment.apps/multi-container-app scaled
```

Exercise 6: ConfigMaps and Secrets

- Implement ConfigMaps and Secrets in your GKE application.

Created a ConfigMap to pass configuration data.

```
kubectl apply -f
C:\Users\onaye\Downloads\cloud_assign3_2\app-config.yaml
```

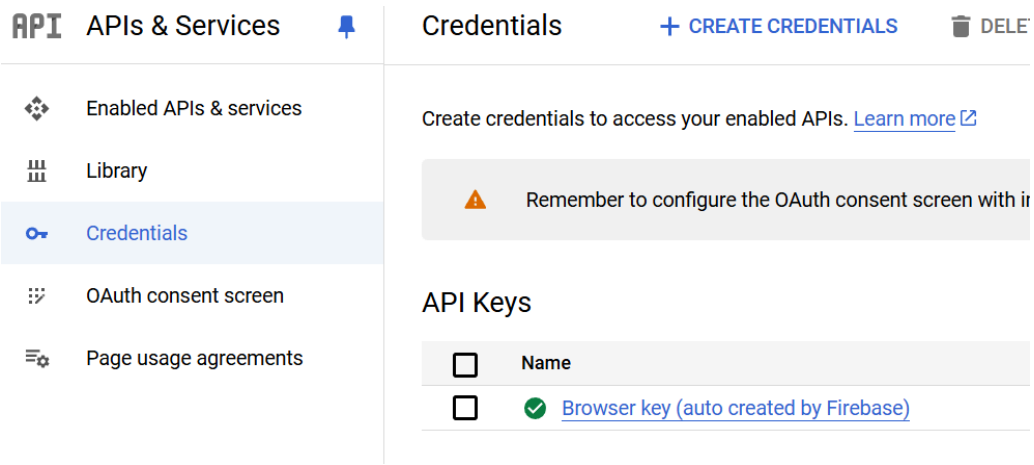
```yaml
! app-config.yaml > {} data
    App config Spotify Backstage - Spotify's Backstage App configuration for all plugins (app-config.json)
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  DATABASE_HOST: db-service
  DATABASE_PORT: "5432"
```

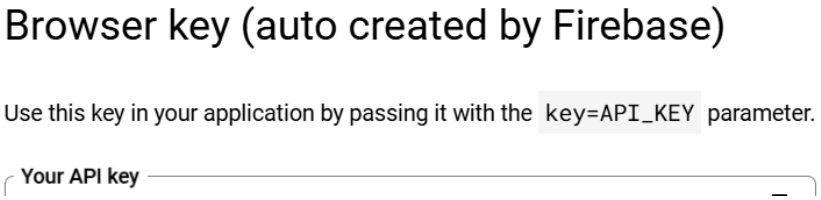
```
C:\Program Files (x86)\Google\Cloud SDK>kubectl apply -f C:\Users\onaye\Downloads\cloud_assign3_2\app-config.yaml
configmap/app-config created
```

- Use a ConfigMap to pass configuration data and a Secret to manage sensitive information (e.g., API keys).

```
app.py    Dockerfile    ! app-secret.yaml ×
! app-secret.yaml > {} data
    io.k8s.api.core.v1.Secret (v1@secret.json)
apiVersion: v1
kind: Secret
metadata:
  name: app-secret
type: Opaque
data:
  API_KEY: <base64-encoded-key>
```

In API_KEY I need to write an encoded key in base 24





I encode an API_KEY in base64 with the following command in PowerShell.

First of all, it converts a string into a byte array, and this array is converted into a base64 string.

```
$encoded =
[Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes("my_pass
word"))
$encoded
```

```
PS C:\Users\onaye> $encoded = [Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes("AIz
TdRLR9o7CUYfFE"))
PS C:\Users\onaye> $encoded
QUl6YVN5QkV0cllTREU1dGQ5U3VwU2hpZlRkUkxSOW83Q1VZZkZF
PS C:\Users\onaye>
```

So I changed in .yaml file API_KEY and run:

```
C:\Program Files (x86)\Google\Cloud SDK>kubectl apply -f C:\Users\onaye\Downloads\cloud_assign3_2\app-secret.yaml
secret/app-secret created
```

3. App Engine and Cloud Functions

Exercise 7: Deploying an App on App Engine

- Create a simple web application (e.g., a Flask or Node.js app) and deploy it to Google App Engine.

Creating flask-app directory, and Installing Flask and env.

```
C:\Users\onaye>mkdir flask-app

C:\Users\onaye>cd flask-app

C:\Users\onaye\flask-app>python -m venv venv

C:\Users\onaye\flask-app>.\venv\Scripts\activate

(venv) C:\Users\onaye\flask-app>pip install Flask
Collecting Flask
  Using cached flask-3.0.3-py3-none-any.whl (101 kB)
Collecting Werkzeug>=3.0.0
  Downloading werkzeug-3.1.2-py3-none-any.whl (224 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 224.4/224.4 kB
```

- Configure app.yaml for the deployment, specifying runtime and service settings.

Defining app.yaml file with the parameters below.

```
Welcome    app.yaml    requirements.txt
app.yaml > {} env_variables
1   runtime: python39
2   entrypoint: gunicorn -b :$PORT main:app
3
4   instance_class: F2
5   env_variables:
6     KEY: value  #
7
```

Running main.py file.

```
(venv) C:\Users\onaye\flask-app>python main.py
 * Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 503-231-298
```

Exercise 8: Using Cloud Functions

- Write a Cloud Function that triggers on a specific event (e.g., an object creation in Cloud Storage) and performs a task (e.g., sending a notification).

So when a new object will be created. The message below, such as 'New file uploaded' will appear.

```
cloud-function > main.py > ...
import json
import google.cloud.storage as storage

def send_notification(data, context):
    bucket_name = data['alua-bucket']
    file_name = data['file1']

    print(f'New file uploaded: {file_name} in bucket: {bucket_name}')
```

- Deploy the function and test it by uploading a file to the designated Cloud Storage bucket.

gcloud functions deploy send_notification --runtime python39 --trigger-resource alua-bucket --trigger-event google.storage.object.finalize

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud functions deploy send_notification --runtime python39 --trigger-resource
alua-bucket --trigger-event google.storage.object.finalize
API [cloudfunctions.googleapis.com] not enabled on project [alua-project]. Would you like to enable and retry (this will
 take a few minutes)? (y/N)?  y

Enabling service [cloudfunctions.googleapis.com] on project [alua-project]...
Operation "operations/acf.p2-838239515081-03f4f738-9ce5-44d0-96a3-4ed72d5e2015" finished successfully.
```

`runtime python39`: Specifies that the Cloud Function uses Python 3.9.

`trigger-resource alua-bucket`: The Cloud Function will trigger when an object is created or updated in the `alua-bucket`.

`trigger-event google.storage.object.finalize`: This sets the event to trigger when an object is uploaded or finalized in the bucket.

But later an error appeared. An error indicated that permission denied, and service account doesn't exist.

```
ERROR: (gcloud.functions.deploy) ResponseError: status=[400], code=[Ok], message=[Validation failed for trigger projects
/alua-project/locations/us-central1/triggers/send-notification-336675: Invalid resource state for "": Permission denied
while using the Eventarc Service Agent. If you recently started to use Eventarc, it may take a few minutes before all ne
cessary permissions are propagated to the Service Agent. Otherwise, verify that it has Eventarc Service Agent role.]
```

So the error indicates that the service account doesn't exist. And with the command below, I will try to find the correct Eventarc Service Account for my project, trying to list all service accounts associated with my project.

```
gcloud services enable eventarc.googleapis.com
```

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud services enable eventarc.googleapis.com
```

```
gcloud iam service-accounts list --project alua-project
```

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud iam service-accounts list --project alua-project
DISPLAY NAME                            EMAIL                                                   DISABLED
Compute Engine default service account  838239515081-compute@developer.gserviceaccount.com      False
App Engine default service account      alua-project@appspot.gserviceaccount.com                False
firebase-adminsdk                       firebase-adminsdk-6tkrp@alua-project.iam.gserviceaccount.com  False
storage-upload                          storage-upload@alua-project.iam.gserviceaccount.com     False
```

So when I got the correct service account, i'm rerunning the command with correct email address.

```
gcloud projects add-iam-policy-binding alua-project
--member="serviceAccount:alua-project@appspot.gserviceaccount.com"
--role="roles/eventarc.eventReceiver"
```

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud projects add-iam-policy-binding alua-project --member="serviceAccount:al
a-project@appspot.gserviceaccount.com" --role="roles/eventarc.eventReceiver"
Updated IAM policy for project [alua-project].
bindings:
- members:
  - serviceAccount:service-838239515081@gcp-gae-service.iam.gserviceaccount.com
  role: roles/appengine.serviceAgent
```

Rerun this command:

```
gcloud functions deploy send_notification --runtime python39
--trigger-resource alua-bucket --trigger-event
google.storage.object.finalize --source
C:\Users\onaye\flask-app\cloud-function
```

```
C:\Program Files (x86)\Google\Cloud SDK>gcloud functions deploy send_notification --runtime python39 --trigger-resource
alua-bucket --trigger-event google.storage.object.finalize --source C:\Users\onaye\flask-app\cloud-function
As of this Cloud SDK release, new functions will be deployed as 2nd gen  functions by default. This is equivalent to cur
rently deploying new  with the --gen2 flag. Existing 1st gen functions will not be impacted and will continue to deploy
as 1st gen functions.
You can disable this behavior by explicitly specifying the --no-gen2 flag or by setting the functions/gen2 config proper
ty to 'off'.
To learn more about the differences between 1st gen and 2nd gen functions, visit:
https://cloud.google.com/functions/docs/concepts/version-comparison
Preparing function...done.
⠟ Deploying function...
  . [Build]
  . [Service]
  . [Trigger]
  . [ArtifactRegistry]
  . [Healthcheck]
  . [Triggercheck]
```

Exercise 9: Monitoring and Logging

- Set up monitoring and logging for your App Engine application and Cloud Functions.

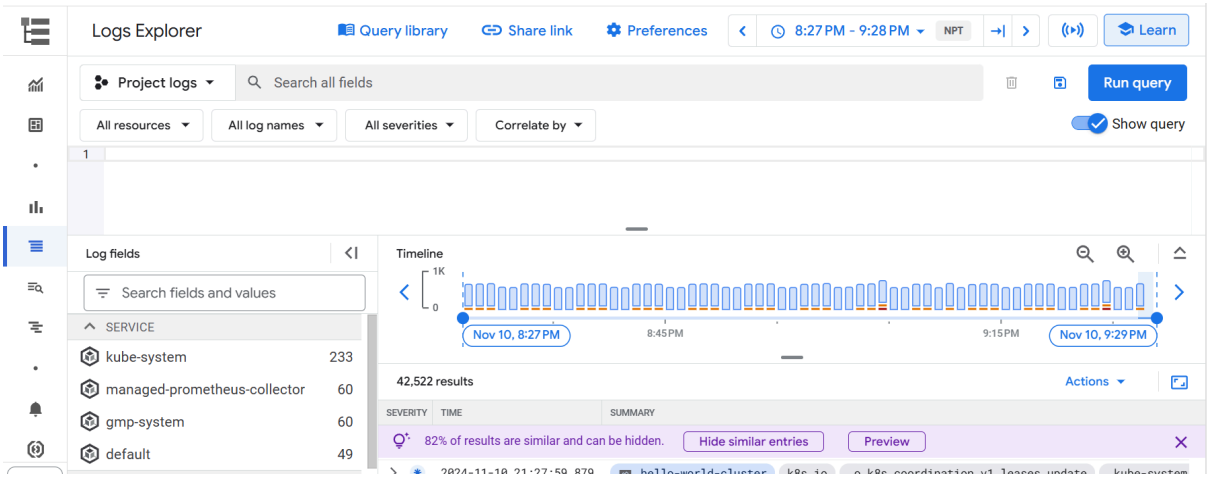
installing Google Cloud and Monitoring agents using:

```
curl -sSO https://dl.google.com/cloudagents/install-logging-agent.sh
sudo bash install-logging-agent.sh
```

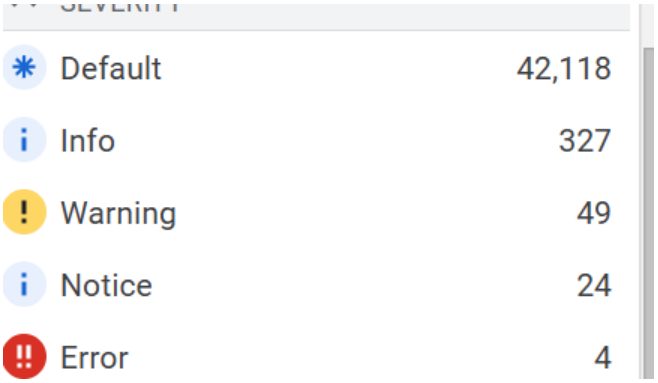
```
(venv) onayeva_alua@instance-alua:~/flask-app$ curl -sSO https://dl.google.com/cloudagents/i
nstall-logging-agent.sh
(venv) onayeva_alua@instance-alua:~/flask-app$ sudo bash install-logging-agent.sh
============================================================================
Starting installation of google-fluentd
============================================================================

Installing agent for Debian or Ubuntu.
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8
)).
OK
Ign:1 http://packages.cloud.google.com/apt google-cloud-logging-bookworm InRelease
Err:2 http://packages.cloud.google.com/apt google-cloud-logging-bookworm Release
  404  Not Found [IP: 173.194.206.101 80]
Reading package lists... Done
E: The repository 'http://packages.cloud.google.com/apt google-cloud-logging-bookworm Releas
e' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by defa
ult.
N: See apt-secure(8) manpage for repository creation and user configuration details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package google-fluentd
```

- Use Google Cloud's monitoring tools to analyze the performance and logs of your deployed services.

In the Monitoring section I'm choosing Logs Explorer.



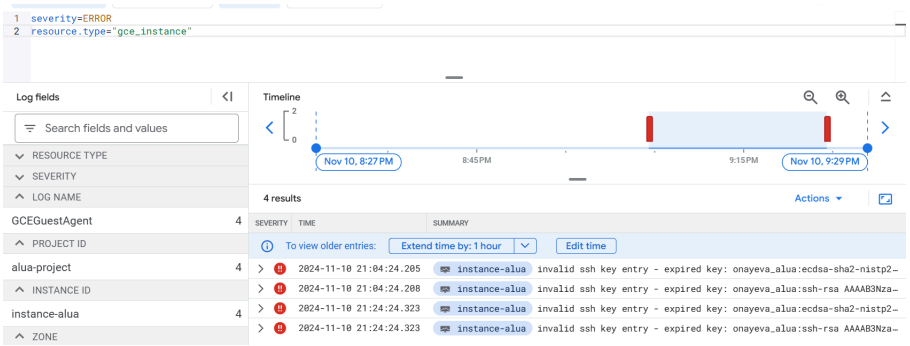
I can see log entries for errors, warnings and etc.



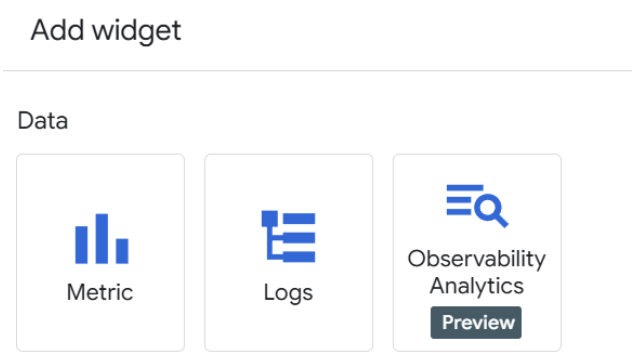
resource.type='gae_app' is used for App Engine

resource.type='cloud_function' is used for Cloud Function logs'

Defining errors like this:



In **Monitoring** > **Dashboard**, created a dashboard with widgets for visualizing logs and performance metrics for App Engine and Cloud Functions, specifying parameters to track.



I choose logs, and later I choose parameters. So dashboards can be created like that.

I can choose any other metrics and also graph types, such as bar graphs, line plots and etc.