

Kazakh-British Technical University



**Cloud Computing
Assingment3**

**Student: Khanfiyeva Elnara
Professor: Serek Azamat**

2024

Table of Contents

Introduction.....	3
1. Identity and Security Management.....	4
2. Google Kubernetes Engine (GKE).....	11
3. App Engine and Cloud Functions.....	18
Results.....	25
Conclusion.....	25
References.....	25

Introduction

Today's cloud computing requires the development of efficient, scalable, and secure applications. By keeping developers at the heart of the process and avoiding the need for them to plan complex infrastructure extensively, Google Cloud Platform (GCP) provides a set of services that may be programmed to achieve these objectives. Key GCP components are examined in this document, including Google Kubernetes Engine (GKE), App Engine, and Google Cloud Functions. To ensure that users and services have the appropriate right of access to resources, these two leverage concepts like Identity and Access Management (IAM), role-based access control, encryption, and cloud login.

Using the well-known container orchestration platform Kubernetes, which gives developers control over a microservices environment, GKE facilitates the deployment and scaling of containerized applications. App Engine is a PaaS service that lets users launch web apps without having to worry about infrastructure management. It handles size adjustments and monitoring. However, Cloud Functions, which work best with small-scale, horizontal, and microservices, operate on a serverless principle in which a collection of functions are run in response to events such as file uploads, HTTP calls, and database updates. All of these elements work together to ensure scalability and flexibility through auto scaling, load balancing, and self-tension adjustments; security and governance with the latest tools; and better developer productivity due to the abstraction of much of the underlying infrastructure.

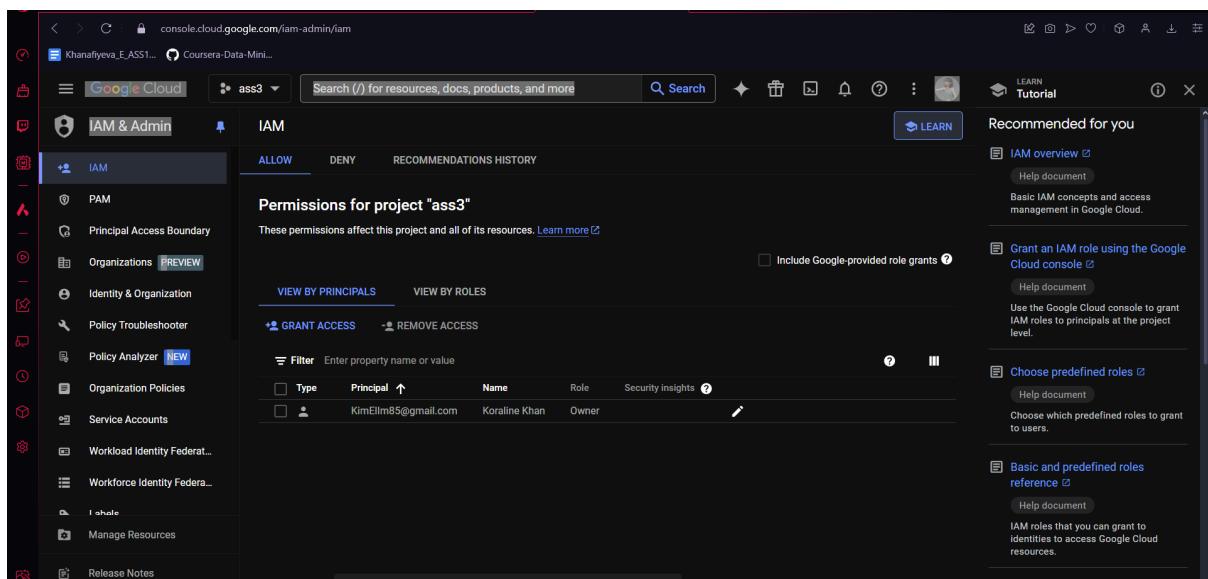
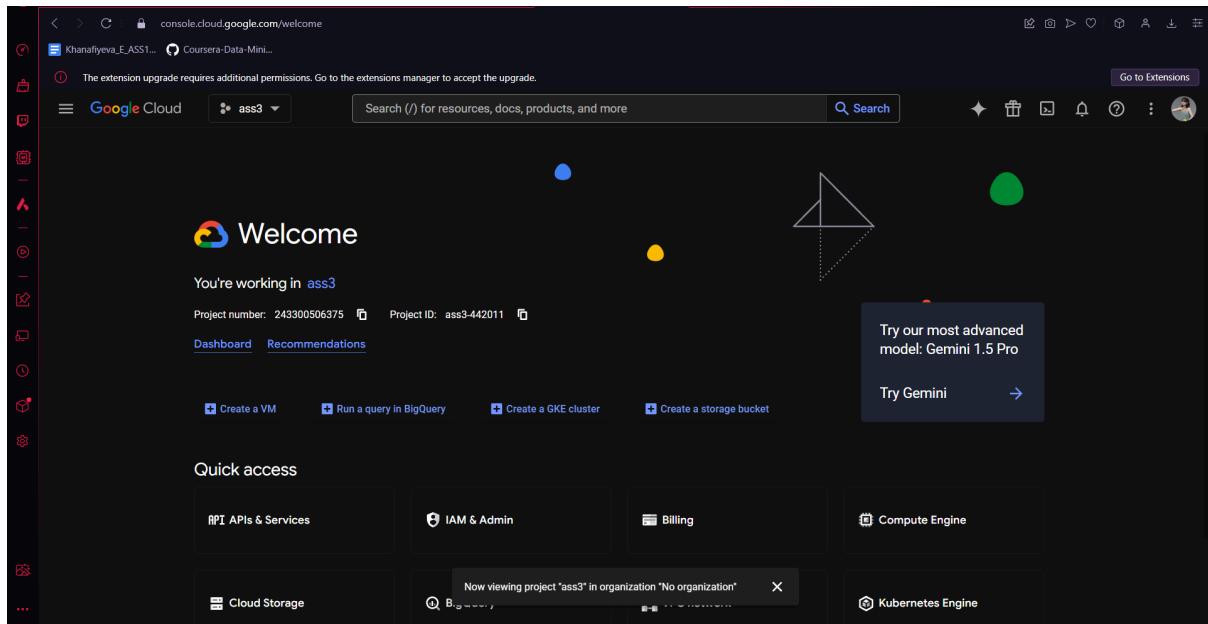
Additionally, managed services like App Engine and GKE, as well as serverless techniques like Cloud Functions, are cost-effective, pay only for what you use, consume model, and decrease operations. With the help of these GCP services, developers and businesses can create scalable, competitive, and secure cloud-based solutions that help them fulfill the ever-evolving economic demands of a technologically advanced market.

1. Identity and Security Management

Exercise 1: Setting Up IAM Roles

- Create a new Google Cloud project.
- Set up Identity and Access Management (IAM) roles for different team members (e.g., Viewer, Editor, Owner).
- Assign specific roles to users and document the permissions associated with each role.

First I go to the Google Cloud Console, and create the project fist.

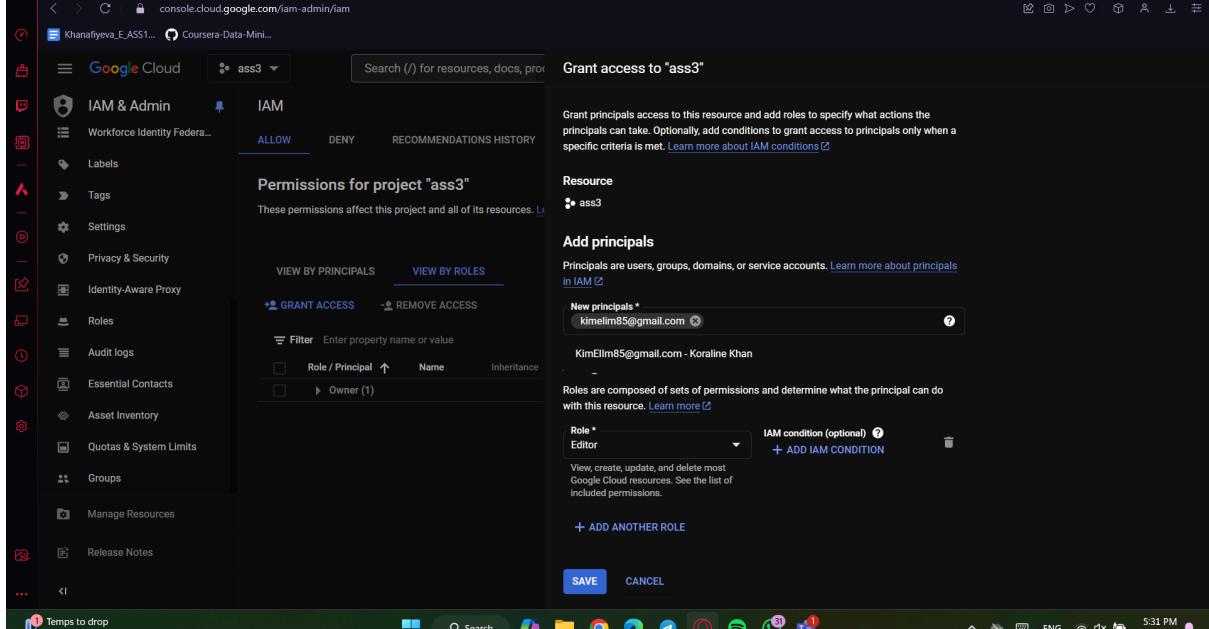


Now we create the IAM & Admin > IAM assign roles of Viewer, Editor, Owner to different users including my account.

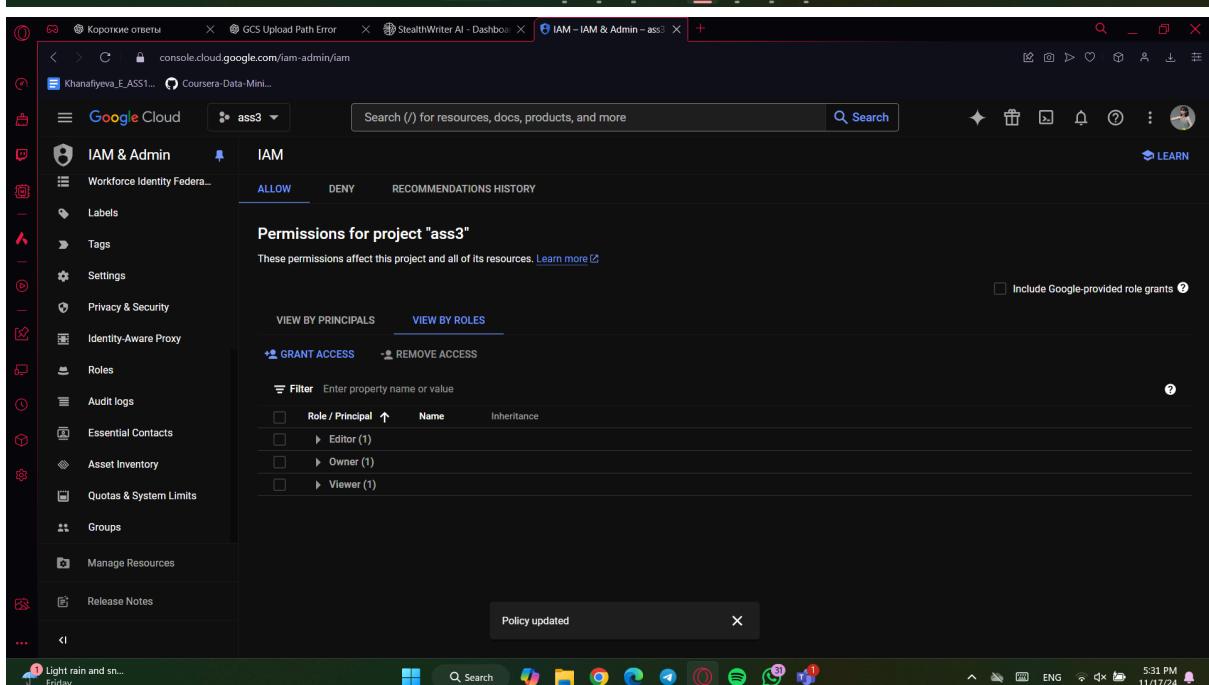
For the viewer the permissions usually are: resourcemanager.projects.get , resourcemanager.projects.list, compute.instances.get, storage.objects.get.

For editor is the create and delete permissions, and for the owner is the setting permissions policies, updating and deleting.

Expected Outcome: Users are assigned roles with specific permissions.



The screenshot shows the Google Cloud IAM interface for project "ass3". The left sidebar includes options like IAM & Admin, Workforce Identity Federation, Labels, Tags, Settings, Privacy & Security, Identity-Aware Proxy, Roles, Audit logs, Essential Contacts, Asset Inventory, Quotas & System Limits, Groups, Manage Resources, and Release Notes. The main area is titled "Grant access to "ass3"" and shows a list of principals under "Add principals". A single principal, "kimelini5@gmail.com", is listed with the role "Editor". There is also an "IAM condition (optional)" section. At the bottom, there are "SAVE" and "CANCEL" buttons.



The screenshot shows the "Permissions for project "ass3"" dialog. It lists the roles assigned to principals: "Editor (1)", "Owner (1)", and "Viewer (1)". A "Policy updated" message is displayed at the bottom. The left sidebar is identical to the first screenshot.

The screenshot shows the Google Cloud IAM & Admin interface. The left sidebar is titled 'IAM & Admin' and has a 'IAM' section selected. The main content area is titled 'Grant access to "ass3"'. It says 'Grant principals access to this resource and add roles to specify what actions the principals can take. Optionally, add conditions to grant access to principals only when a specific criteria is met.' Below this, it says 'Resource: ass3'. Under 'Add principals', there is a list with one item: 'New principals * khanafiyevainara2022@gmail.com'. Under 'Assign roles', there is a table with one row: 'Role / Principal ↑ Name' with 'Viewer' listed. At the bottom are 'SAVE' and 'CANCEL' buttons.

The screenshot shows the Google Cloud IAM & Admin interface. The left sidebar is titled 'IAM & Admin' and has a 'IAM' section selected. The main content area is titled 'Create role'. It shows a table with one row: 'ID * ke_role_some'. Under '+ ADD PERMISSIONS', it says 'No assigned permissions'. A modal window titled 'Add permissions' is open, showing a table of permissions. The first few rows are: 'Permission ↑ Status', 'accessapproval.requests.get Supported', 'accessapproval.requests.list Supported', 'accessapproval.serviceAccounts.get Supported', 'accessapproval.settings.get Supported', 'actions.agent.get Supported', 'actions.agentVersions.get Supported', 'actions.agentVersions.list Supported', 'firebase.projects.get Supported', 'resourcemanager.projects.get Supported', 'resourcemanager.projects.list Non-applicable'. At the bottom of the modal are 'CANCEL' and 'ADD' buttons.

The screenshot shows the Google Cloud IAM & Admin interface. On the left, there's a sidebar with various services like IAM, PAM, Principal Access Boundary, and Organizations. The main area is titled 'Create role' under 'General Availability'. A table lists '9 assigned permissions' with columns for 'Permission' and 'Status'. Most permissions are 'Supported'. A note at the bottom says 'Some permissions might be associated with and checked by third parties. These permissions contain the third party's service and domain name in the permission prefix.' A modal window at the bottom right says '9 permissions added'.

This screenshot shows the same 'Create role' interface as above, but with custom role details filled in. The 'Title' field is 'ke_role_some', 'ID' is 'ke_role_some', and 'Role launch stage' is 'General Availability'. The rest of the interface is identical to the first screenshot, showing assigned permissions and a confirmation message.

Exercise 2: Service Accounts

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

The service account is a special kind of account used by an application or compute workload, rather than a person, we will use it to authenticate a script for accessing Google Cloud

Storage.

The screenshot shows the Google Cloud IAM & Admin Service Accounts page. The sidebar on the left is titled 'IAM & Admin' and includes options like IAM, PAM, Principal Access Boundary, Organizations (PREVIEW), Identity & Organization, Policy Troubleshooter, Policy Analyzer (NEW), Organization Policies, Service Accounts (selected), Workload Identity Federation, and Workforce Identity Federation. The main content area is titled 'Service accounts for project "ass3"' and contains a table with one row:

Email	Status	Name	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
service_some_acc@ass3-442011.iam.gserviceaccount.com	Enabled	service_some_acc	No keys			103352487663382532558	⋮

Create a service account in IAM console. I will assign the role of **Storage Object Admin** to the service account, to have an access from my console to a storage.

I will generate JSON key of service account in the console. and download it.

The screenshot shows the Google Cloud IAM & Admin Service Accounts - Details page for the service account 'service_some_acc'. The sidebar is identical to the previous screenshot. The main content area has tabs for DETAILS, PERMISSIONS, KEYS (selected), METRICS, and LOGS. The KEYS tab displays a warning about service account keys and a note about Google automatically disabling service account keys. It also has sections for adding a new key pair and block service account key creation. A table at the bottom shows no rows displayed.

```

from google.cloud import storage

def upload_to_bucket(bucket_name, source_file_name,
destination_blob_name):
    storage_client =
storage.Client.from_service_account_json('C:/Users/elnus/Downloads/as
s3-442011-a63ede4088c4.json')
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(destination_blob_name)
    blob.upload_from_filename(source_file_name)
    print(f"File {source_file_name} uploaded to
{destination_blob_name}.")
upload_to_bucket('my_buckpipp',
'C:/Users/elnus/Downloads/ass3-442011-a63ede4088c4.json', 'key.json')

```

This script will upload the file to the bucket.

Here is bucket created :

The screenshot shows the Google Cloud Storage 'Bucket details' page for 'my_buckpipp'. The left sidebar has 'Cloud Storage' selected under 'Buckets'. The main area shows the bucket's configuration: Location (us), Storage class (Standard), Public access (Subject to object ACLs), and Protection (None). Below this is the 'OBJECTS' tab, which displays a table with columns: Buckets > my_buckpipp, Filter, Create folder, Upload, Transfer data, Other services, and a sorting/filtering options dropdown. A modal window titled 'Sorting and filtering options' is open, showing instructions: 'You can now sort and filter objects and folders by any value. Continue to filter by name prefix only or select 'Sort and filter' from the filtering menu.' At the bottom right of the main area, a success message says 'Created bucket my_buckpipp'.

```

C:\Users\elnus\Downloads>python script.py
File C:/Users/elnus/Downloads/ass3-442011-a63ede4088c4.json uploaded to key.json.
C:\Users\elnus\Downloads>

```

The screenshot shows the Google Cloud Storage 'Bucket details' page for 'my_buckpipp'. The left sidebar has 'Cloud Storage' selected under 'Buckets'. The main area shows a single object named 'key.json' with details: Size 2.3 KB, Type application/json, Created Nov 17, 2024, 6:04:20 PM, Standard storage class, Last modified Nov 17, 2024, 6:04:20 PM, and Public access set to 'Not public'. A 'Policy updated' message is visible at the bottom.

I ran the script and the key got uploaded to the bucket.

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

I go to IAM & Admin > Organization Policies and then apply a policy disabling the public IP, and there for I can't access it from the outside now.

The screenshot shows the Google Cloud IAM & Admin 'Organizations' page. The left sidebar has 'Organization Policies' selected under 'PREVIEW'. A message states 'Page not viewable for projects. This feature requires an organization'. Below this, instructions for setting up a Google Cloud organization are provided, along with a 'GO TO THE CHECKLIST' button. A message at the bottom indicates 'Now viewing project "ass3" in organization "No organization"'.

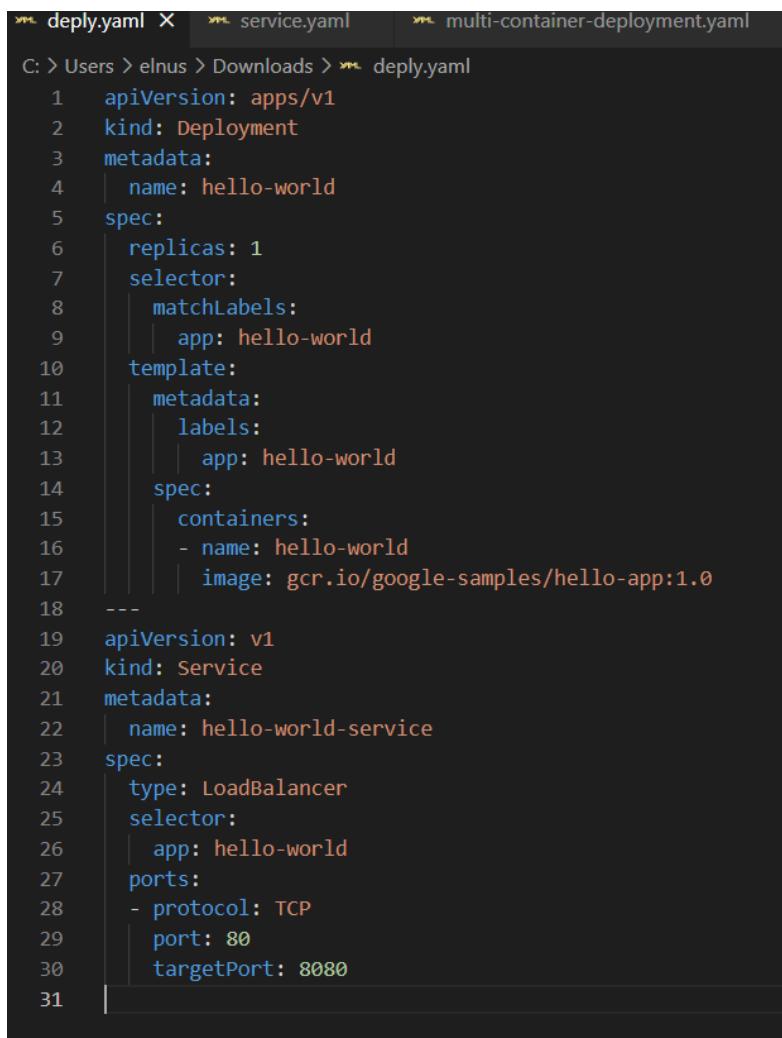
2. Google Kubernetes Engine (GKE)

Exercise 4: Deploying a Simple Application

- Set up a GKE cluster using the Google Cloud Console or gcloud command line.
- Deploy a simple containerized application (e.g., a Hello World app) to the cluster and expose it via a LoadBalancer service.

I go to the Google Console and Kubernetes Engine > Clusters > Create Cluster.

After creating a cluster I create the yaml file for an image for K8s.



```
deploy.yaml X  service.yaml  multi-container-deployment.yaml
C: > Users > elnus > Downloads > deploy.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: hello-world
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: hello-world
10   template:
11     metadata:
12       labels:
13         app: hello-world
14     spec:
15       containers:
16         - name: hello-world
17           image: gcr.io/google-samples/hello-app:1.0
18   ---
19  apiVersion: v1
20  kind: Service
21  metadata:
22    name: hello-world-service
23  spec:
24    type: LoadBalancer
25    selector:
26      app: hello-world
27    ports:
28      - protocol: TCP
29        port: 80
30        targetPort: 8080
31  |
```

I apply the file to k8s by kubectl apply -f deploy.yaml

gcloud init

gcloud container clusters create smth --zone us-central1-a

gcloud container clusters get-credentials cluster --zone us-central1-a

```

Command Prompt
hello-world-55897d9698-p92sn  0/1    Pending   0          54s
C:\Users\elnus\Downloads>kubectl scale deployment deploy --replicas=4
error: no objects passed to scale
C:\Users\elnus\Downloads>kubectl get pods
NAME        READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-p92sn   1/1     Running   0          2m23s
C:\Users\elnus\Downloads>kubectl scale deployment deploy --replicas=4
error: no objects passed to scale
C:\Users\elnus\Downloads>kubectl scale deployment deploy --replicas=4
error: no objects passed to scale
C:\Users\elnus\Downloads>kubectl get deployments
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
hello-world   1/1      1           1          3m45s
C:\Users\elnus\Downloads>kubectl scale deployment hello-world --replicas=4
deployment.apps/hello-world scaled
C:\Users\elnus\Downloads>kubectl apply -f service.yaml
service/hello-world-service unchanged
C:\Users\elnus\Downloads>kubectl get service hello-world-service
NAME        TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
hello-world-service   LoadBalancer   34.118.236.250   35.226.82.110   80:31714/TCP   5m57s
C:\Users\elnus\Downloads>

```

kubectl get pods

I create a file named service.yaml to expose my application using a LoadBalancer:

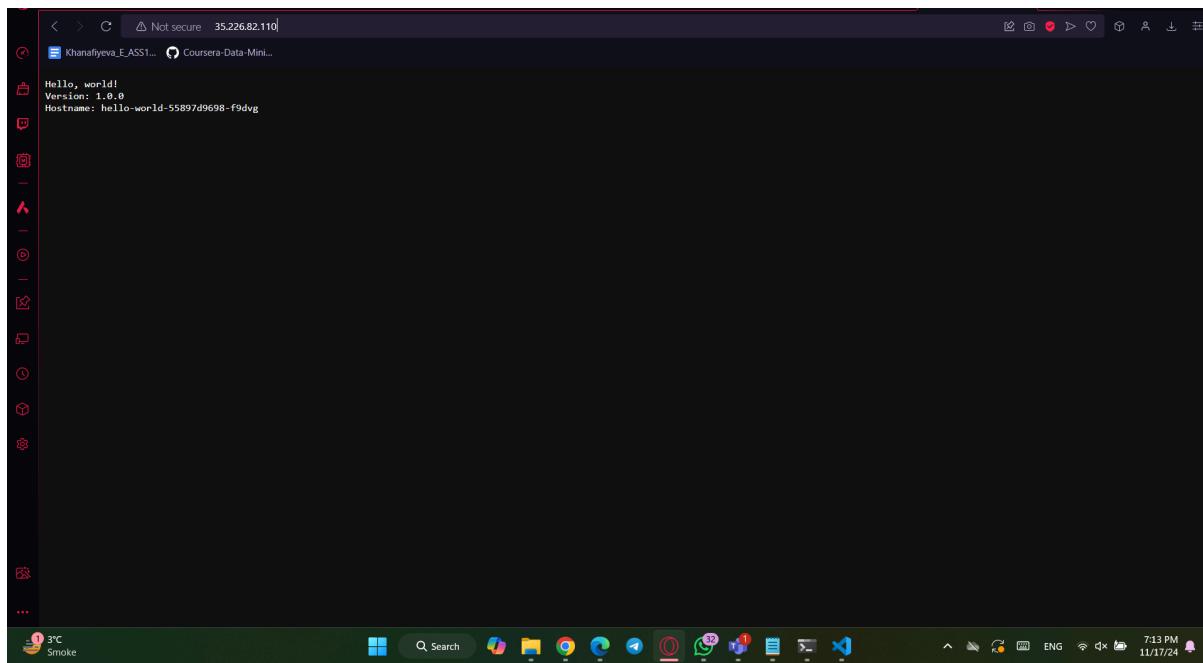
```

deploy.yaml  service.yaml  multi-container
C: > Users > elnus > Downloads > service.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: hello-world-service
5  spec:
6    selector:
7      app: hello-world
8    ports:
9      - protocol: TCP
10     port: 80
11     targetPort: 8080
12     type: LoadBalancer
13

```

kubectl apply -f service.yaml

kubectl get service service



As expected it's available.

Exercise 5: Managing Pods and Deployments

- Create a Deployment for a multi-container application using Kubernetes YAML files.
- Scale the Deployment to manage the number of replicas and update the application with a new container image.

We scale the deployment that we have by kubectl scale deployment hello-world --replicas=3.

```
multi-container-deployment.yaml
C: > Users > elnus > Downloads > multi-container-deployment.yaml
 1  apiVersion: apps/v1
 2  kind: Deployment
 3  metadata:
 4    name: multi-container-deployment
 5  spec:
 6    replicas: 2
 7    selector:
 8      matchLabels:
 9        app: multi-container
10    template:
11      metadata:
12        labels:
13          app: multi-container
14      spec:
15        containers:
16          - name: frontend
17            image: gcr.io/google-samples/hello-app:1.0
18            ports:
19              - containerPort: 8080
20          - name: backend
21            image: gcr.io/google-samples/hello-backend:1.0
22            ports:
23              - containerPort: 9090
```

We make another yaml file:

This defines the 2 containers here, so I apply it to deploy: **kubectl apply -f multi-container-deployment.yaml** and **kubectl get pods**.

Then we scale the app: **kubectl scale deployment multi-container-deployment --replicas=4**

kubectl get pods

```
C:\Users\elnus\Downloads>kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
hello-world   1/1       1           1          3m45s

C:\Users\elnus\Downloads>kubectl scale deployment hello-world --replicas=4
deployment.apps/hello-world scaled

C:\Users\elnus\Downloads>kubectl apply -f service.yaml
service/hello-world-service unchanged

C:\Users\elnus\Downloads>kubectl get service hello-world-service
NAME        TYPE   CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
hello-world-service   LoadBalancer   34.118.236.250  35.226.82.110  80:31714/TCP  5m57s

C:\Users\elnus\Downloads>kubectl apply -f multi-container-deployment.yaml
Warning: autopilot-default-resources-mutator:Autopilot updated Deployment default/multi-container-deployment: defaulted unspecified 'cpu' resource for containers [frontend, backend] (see http://g.co/gke/autopilot-defaults).
deployment.apps/multi-container-deployment created

C:\Users\elnus\Downloads>kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dv9g   1/1     Running   0          3m44s
hello-world-55897d9698-p92sn   1/1     Running   0          7m49s
hello-world-55897d9698-smwx6   0/1     Pending   0          3m44s
hello-world-55897d9698-vfk9k   1/1     Running   0          3m44s
multi-container-deployment-668699d5b7-7277p  0/2     Pending   0          12s
multi-container-deployment-668699d5b7-ppp9m  0/2     Pending   0          12s

C:\Users\elnus\Downloads>
```

```
C:\Users\elnus\Downloads>kubectl get service hello-world-service
NAME        TYPE   CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
hello-world-service   LoadBalancer   34.118.236.250  35.226.82.110  80:31714/TCP  5m57s

C:\Users\elnus\Downloads>kubectl apply -f multi-container-deployment.yaml
Warning: autopilot-default-resources-mutator:Autopilot updated Deployment default/multi-container-deployment: defaulted unspecified 'cpu' resource for containers [frontend, backend] (see http://g.co/gke/autopilot-defaults).
deployment.apps/multi-container-deployment created

C:\Users\elnus\Downloads>kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dv9g   1/1     Running   0          3m44s
hello-world-55897d9698-p92sn   1/1     Running   0          7m49s
hello-world-55897d9698-smwx6   0/1     Pending   0          3m44s
hello-world-55897d9698-vfk9k   1/1     Running   0          3m44s
multi-container-deployment-668699d5b7-7277p  0/2     Pending   0          12s
multi-container-deployment-668699d5b7-ppp9m  0/2     Pending   0          12s

C:\Users\elnus\Downloads>kubectl scale deployment multi-container-deployment --replicas=4
deployment.apps/multi-container-deployment scaled

C:\Users\elnus\Downloads>kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dv9g   1/1     Running   0          4m8s
hello-world-55897d9698-p92sn   1/1     Running   0          8m13s
hello-world-55897d9698-smwx6   0/1     Pending   0          4m8s
hello-world-55897d9698-vfk9k   1/1     Running   0          4m8s
multi-container-deployment-668699d5b7-7277p  0/2     Pending   0          36s
multi-container-deployment-668699d5b7-jxd5p  0/2     Pending   0          4s
multi-container-deployment-668699d5b7-ppp9m  0/2     Pending   0          36s
multi-container-deployment-668699d5b7-q5x6r  0/2     Pending   0          3s

C:\Users\elnus\Downloads>
```

```

C:\Users\elnus\Downloads>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dvg      1/1    Running   0          3m44s
hello-world-55897d9698-p92sn      1/1    Running   0          7m49s
hello-world-55897d9698-smwx6      0/1    Pending   0          3m44s
hello-world-55897d9698-vfk9k      1/1    Running   0          3m44s
multi-container-deployment-668699d5b7-7277p  0/2    Pending   0          12s
multi-container-deployment-668699d5b7-ppp9m  0/2    Pending   0          12s

C:\Users\elnus\Downloads>kubectl scale deployment multi-container-deployment --replicas=4
deployment.apps/multi-container-deployment scaled

C:\Users\elnus\Downloads>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dvg      1/1    Running   0          4m8s
hello-world-55897d9698-p92sn      1/1    Running   0          8m13s
hello-world-55897d9698-smwx6      0/1    Pending   0          4m8s
hello-world-55897d9698-vfk9k      1/1    Running   0          4m8s
multi-container-deployment-668699d5b7-7277p  0/2    Pending   0          36s
multi-container-deployment-668699d5b7-jxd5p  0/2    Pending   0          4s
multi-container-deployment-668699d5b7-ppp9m  0/2    Pending   0          36s
multi-container-deployment-668699d5b7-q5x6r  0/2    Pending   0          3s

C:\Users\elnus\Downloads>kubectl set image deployment/multi-container-deployment frontend=gcr.io/google-samples/hello-app:2.0 backend=gcr.io/google-samples/hello-backend:2.0
deployment.apps/multi-container-deployment image updated

C:\Users\elnus\Downloads>kubectl rollout status deployment/multi-container-deployment
Waiting for deployment "multi-container-deployment" rollout to finish: 2 out of 4 new replicas have been updated...

```

Exercise 6: ConfigMaps and Secrets

- Implement ConfigMaps and Secrets in your GKE application.
- Use a ConfigMap to pass configuration data and a Secret to manage sensitive information (e.g., API keys).

```

C:\Users\elnus\Downloads>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dvg      1/1    Running   0          3m44s
hello-world-55897d9698-p92sn      1/1    Running   0          7m49s
hello-world-55897d9698-smwx6      0/1    Pending   0          3m44s
hello-world-55897d9698-vfk9k      1/1    Running   0          3m44s
multi-container-deployment-668699d5b7-7277p  0/2    Pending   0          12s
multi-container-deployment-668699d5b7-ppp9m  0/2    Pending   0          12s

C:\Users\elnus\Downloads>kubectl scale deployment multi-container-deployment --replicas=4
deployment.apps/multi-container-deployment scaled

C:\Users\elnus\Downloads>kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
hello-world-55897d9698-f9dvg      1/1    Running   0          4m8s
hello-world-55897d9698-p92sn      1/1    Running   0          8m13s
hello-world-55897d9698-smwx6      0/1    Pending   0          4m8s
hello-world-55897d9698-vfk9k      1/1    Running   0          4m8s
multi-container-deployment-668699d5b7-7277p  0/2    Pending   0          36s
multi-container-deployment-668699d5b7-jxd5p  0/2    Pending   0          4s
multi-container-deployment-668699d5b7-ppp9m  0/2    Pending   0          36s
multi-container-deployment-668699d5b7-q5x6r  0/2    Pending   0          3s

C:\Users\elnus\Downloads>kubectl set image deployment/multi-container-deployment frontend=gcr.io/google-samples/hello-app:2.0 backend=gcr.io/google-samples/hello-backend:2.0
deployment.apps/multi-container-deployment image updated

C:\Users\elnus\Downloads>kubectl rollout status deployment/multi-container-deployment
Waiting for deployment "multi-container-deployment" rollout to finish: 2 out of 4 new replicas have been updated...

C:\Users\elnus\Downloads>
C:\Users\elnus\Downloads>
C:\Users\elnus\Downloads>
C:\Users\elnus\Downloads>
C:\Users\elnus\Downloads>kubectl rollout status deployment/multi-container-deployment
Waiting for deployment "multi-container-deployment" rollout to finish: 2 out of 4 new replicas have been updated...

C:\Users\elnus\Downloads>kubectl apply -f configmap.yaml
configmap/app-config created

C:\Users\elnus\Downloads>kubectl get configmap app-config -o yaml

```

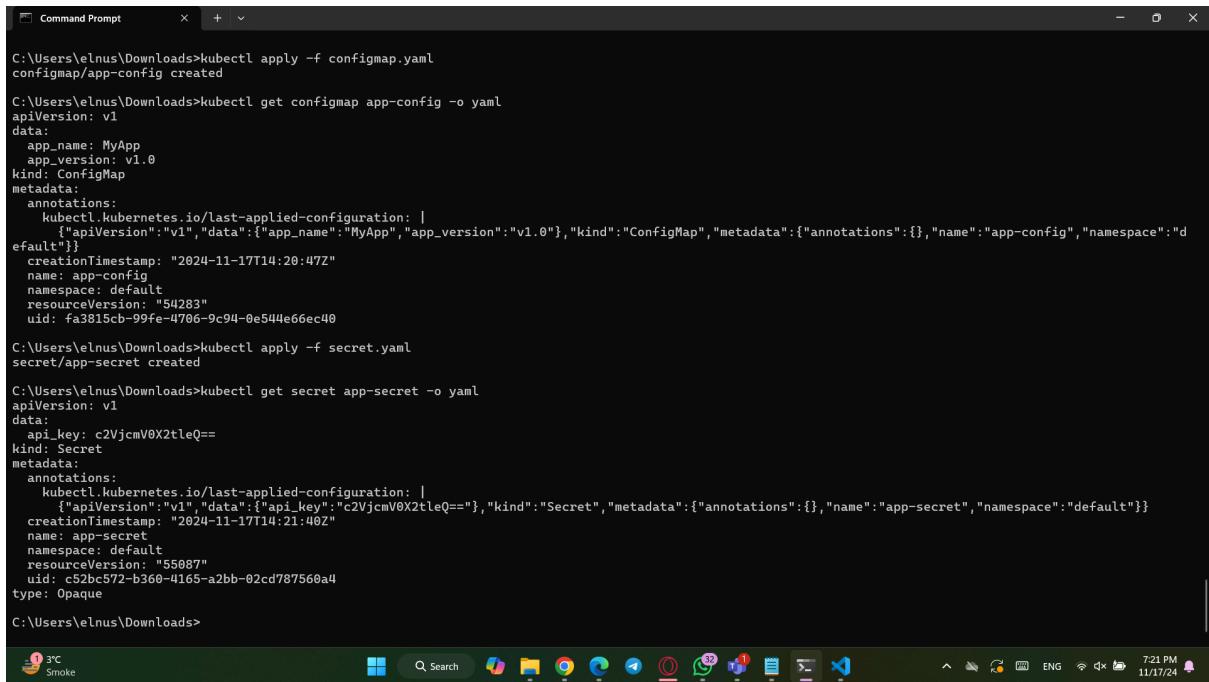
Here I apply config maps, first i create yaml file:

```
C:\> Users > elnus > Downloads > configmap.yaml
```

```
1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: app-config
5  data:
6    app_name: "MyApp"
7    app_version: "v1.0"
8
```

```
kubectl apply -f configmap.yaml
```

```
kubectl get configmap app-config -o yaml
```



```
C:\Users\elnus\Downloads>kubectl apply -f configmap.yaml
configmap/app-config created

C:\Users\elnus\Downloads>kubectl get configmap app-config -o yaml
apiVersion: v1
data:
  app_name: MyApp
  app_version: v1.0
kind: ConfigMap
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"app_name":"MyApp","app_version":"v1.0"},"kind":"ConfigMap","metadata":{"annotations":{},"name":"app-config","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:20:47Z"
  name: app-config
  namespace: default
  resourceVersion: "54283"
  uid: fa3815cb-99fe-4706-9c94-0e544e66ec40

C:\Users\elnus\Downloads>kubectl apply -f secret.yaml
secret/app-secret created

C:\Users\elnus\Downloads>kubectl get secret app-secret -o yaml
apiVersion: v1
data:
  api_key: c2VjcmV0X2tleQ==
kind: Secret
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"api_key":"c2VjcmV0X2tleQ=="},"kind":"Secret","metadata":{"annotations":{},"name":"app-secret","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:21:40Z"
  name: app-secret
  namespace: default
  resourceVersion: "55087"
  uid: c52bc572-b360-4165-a2bb-02cd787560a4
type: Opaque
```

Then i also create the secret yaml file:

```
apiVersion: v1 kind: Secret metadata: name: app-secret data: api_key: c2VjcmV0X2tleQ==
```

Then we apply it again to the k8s: **kubectl apply -f secret.yaml**

```
Command Prompt

C:\Users\elnus\Downloads>kubectl apply -f configmap.yaml
configmap/app-config created

C:\Users\elnus\Downloads>kubectl get configmap app-config -o yaml
apiVersion: v1
data:
  app_name: MyApp
  app_version: v1.0
kind: ConfigMap
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"app_name":"MyApp","app_version":"v1.0"},"kind":"ConfigMap","metadata":{"annotations":{},"name":"app-config","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:20:47Z"
  name: app-config
  namespace: default
  resourceVersion: "54283"
  uid: fa3815cb-99fe-4706-9c94-0e544e66ec40
secret:
  type: Opaque

C:\Users\elnus\Downloads>kubectl apply -f secret.yaml
secret/app-secret created

C:\Users\elnus\Downloads>kubectl get secret app-secret -o yaml
apiVersion: v1
data:
  api_key: c2VjcmV0X2tleQ==
kind: Secret
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"api_key":"c2VjcmV0X2tleQ=="},"kind":"Secret","metadata":{"annotations":{},"name":"app-secret","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:21:40Z"
  name: app-secret
  namespace: default
  resourceVersion: "55087"
  uid: c52bc572-b360-4165-a2bb-02cd787560a4
type: Opaque

C:\Users\elnus\Downloads>
```

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

I create the simple web app here with as before flask) :

```
mkdir my-flask-app
```

```
cd my-flask-app
```

Requirements.txt with flask version, of course.

In the app.py I write dome basic stuff:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'

if __name__ == '__main__':
    app.run()
```

ThenI configure the app.yaml file with:

```
C: > Users > elnus > Downloads > my-flask-app > app.yaml
1   runtime: python39
2   entrypoint: gunicorn -b :$PORT app:app
3
4   env_variables:
5     FLASK_ENV: 'production'
6
```

Then let's deploy it to the GAE, so open gcloud cli and run : **gcloud init**, then **gcloud app deploy** , **gcloud app browse**

```
C:\Users\elnus\Downloads>kubectl get configmap app-config -o yaml
apiVersion: v1
data:
  app_name: MyApp
  app_version: v1.0
kind: ConfigMap
metadata:
  annotations:
    kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"app_name":"MyApp","app_version":"v1.0"},"kind":"ConfigMap","metadata":{"annotations":{},"name":"app-config","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:20:47Z"
  name: app-config
  namespace: default
  resourceVersion: "54283"
  uid: fa3815cb-99fe-4706-9c94-0e544e66ec40
C:\Users\elnus\Downloads>kubectl apply -f secret.yaml
secret/app-secret created

C:\Users\elnus\Downloads>kubectl get secret app-secret -o yaml
apiVersion: v1
data:
  api_key: c2VjcmV0X2tleQ==
kind: Secret
metadata:
  annotations:
    kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","data":{"api_key":"c2VjcmV0X2tleQ=="},"kind":"Secret","metadata":{"annotations":{},"name":"app-secret","namespace":"default"}}
  creationTimestamp: "2024-11-17T14:21:40Z"
  name: app-secret
  namespace: default
  resourceVersion: "55087"
  uid: c52bc572-b360-4165-a2bb-02cd787560a4
type: Opaque

C:\Users\elnus\Downloads>mkdir my-flask-app
C:\Users\elnus\Downloads>cd my-flask-app
C:\Users\elnus\Downloads\my-flask-app>
```

```
[39] australia-southeast1-c
[40] australia-southeast1-a
[41] southamerica-east1-b
[42] southamerica-east1-c
[43] southamerica-east1-a
[44] africa-south1-a
[45] africa-south1-b
[46] africa-south1-c
[47] asia-east2-a
[48] asia-east2-b
[49] asia-east2-c
[50] asia-northeast2-a
Did not print [75] options.
Too many options [125]. Enter "list" at prompt to print choices fully.
Please enter numeric choice or text value (must exactly match list item): 27
Your project default Compute Engine zone has been set to [asia-east1-a].
You can change it by running [gcloud config set compute/zone NAME].
Your project default Compute Engine region has been set to [asia-east1].
You can change it by running [gcloud config set compute/region NAME].
The Google Cloud CLI is configured and ready to use!
* Commands that require authentication will use kimelelim85@gmail.com by default
* Commands will reference project `ass3-4d2011` by default
* Compute Engine commands will use region `asia-east1` by default
* Compute Engine commands will use zone `asia-east1-a` by default
Run 'gcloud help config' to learn how to change individual settings
This gcloud configuration is called [default]. You can create additional configurations if you work with multiple accounts and/or projects.
Run 'gcloud topic configurations' to learn more.
Some things to try next:
* Run 'gcloud --help' to see the Cloud Platform services you can interact with. And run 'gcloud help COMMAND' to get help on any gcloud command.
* Run 'gcloud topic --help' to learn about advanced features of the CLI like arg files and output formatting
* Run 'gcloud cheat-sheet' to see a roster of go-to 'gcloud' commands.
C:\Users\elnus\Downloads\my-flask-app>gcloud app browse|
```

```

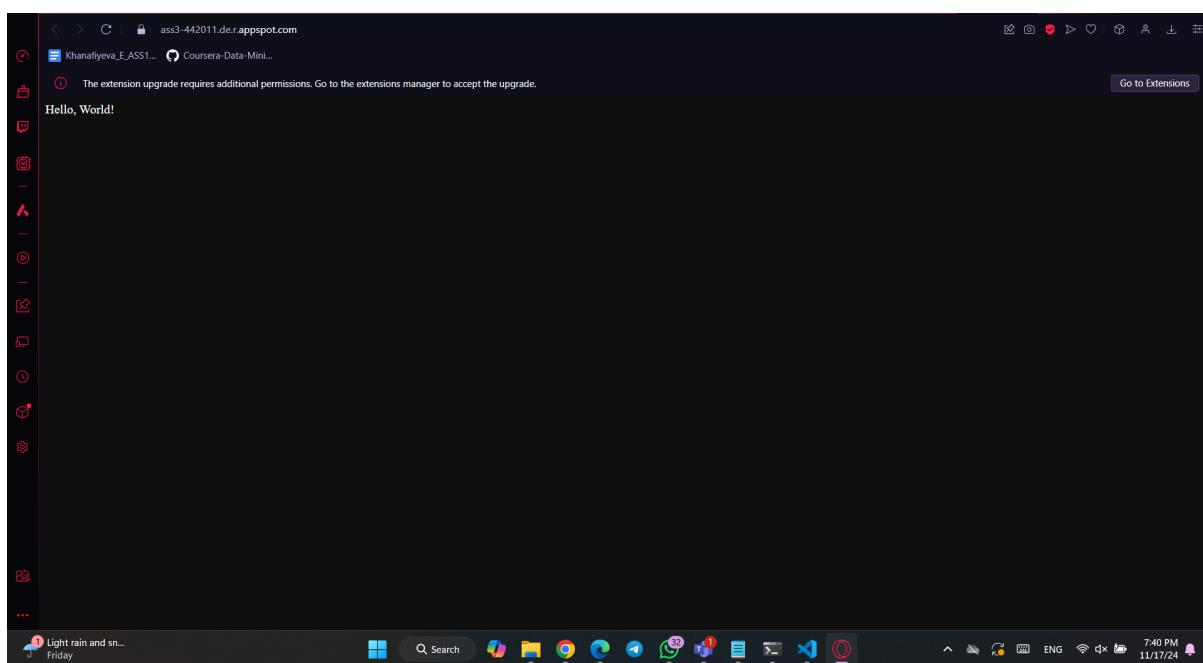
[10] europe-central2 (supports standard and flexible)
[11] europe-west (supports standard and flexible and search_api)
[12] europe-west2 (supports standard and flexible and search_api)
[13] europe-west3 (supports standard and flexible and search_api)
[14] europe-west6 (supports standard and flexible and search_api)
[15] northamerica-northeast1 (supports standard and flexible and search_api)
[16] southamerica-east1 (supports standard and flexible and search_api)
[17] us-central (supports standard and flexible and search_api)
[18] us-east1 (supports standard and flexible and search_api)
[19] us-east4 (supports standard and flexible and search_api)
[20] us-west1 (supports standard and flexible)
[21] us-west2 (supports standard and flexible and search_api)
[22] us-west3 (supports standard and flexible and search_api)
[23] us-west4 (supports standard and flexible and search_api)
[24] cancel
Please enter your numeric choice: 1

Creating App Engine application in project [ass3-442011] and region [asia-east1]...done.
Success! The app is now created. Please use 'gcloud app deploy' to deploy your first app.

C:\Users\elnus\Downloads\my-flask-app>gcloud app deploy
Services to deploy:
descriptor: [c:\Users\elnus\Downloads\my-flask-app\app.yaml]
source: [c:\Users\elnus\Downloads\my-flask-app]
target project: [ass3-442011]
target service: [default]
target version: [20241117t192811]
target url: [https://ass3-442011.de.r.appspot.com]
target service account: [ass3-442011@appspot.gserviceaccount.com]

Do you want to continue (Y/n)? y
Beginning deployment of service [default]...
Created .gcloudignore file. See 'gcloud topic gcloudignore' for details.
#####
#= Uploading 2 files to Google Cloud Storage      =
#####
File upload done.

```



Then I can see it from the browser.

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

I install the firebase sdk to my project, thne i create the index.js :

```
Deploying function (may take a while - up to 2 minutes).../
For Cloud Build Logs, visit: https://console.cloud.google.com/cloud-build/builds;region=us-central1/a48ef055-4476-4966-9e2f-1f240da41af8?project=24330050637
5
Deploying function (may take a while - up to 2 minutes)...done.
automaticUpdatePolicy: {}
availableMemoryMb: 256
buildId: a48ef055-4476-4966-9e2f-1f240da41af8
buildName: projects/243300506375/locations/us-central1/builds/a48ef055-4476-4966-9e2f-1f240da41af8
buildServiceAccount: projects/ass3-442011/serviceAccounts/243300506375-compute@developer.gserviceaccount.com
dockerRegistry: ARTIFACT_REGISTRY
entryPoint: notify
eventTrigger:
  eventType: google.storage.object.finalize
  failurePolicy: {}
  resource: projects/_/buckets/newbuvkypip
  service: storage.googleapis.com
ingressSettings: ALLOW_ALL
labels:
  deployment-tool: cli-gcloud
maxInstances: 3000
name: projects/ass3-442011/locations/us-central1/functions/notify
runtime: python39
serviceAccountEmail: ass3-442011@appspot.gserviceaccount.com
sourceUploadUrl: https://storage.googleapis.com/uploads-592202699764.us-central1.cloudfunctions.appspot.com/207918e6-a810-489c-9f24-aec1b02cf4bd.zip
status: ACTIVE
timeout: 60s
updateTime: '2024-11-17T15:25:55.062Z'
versionId: '2'

C:\Users\elnus\Downloads\my-cloud-function>gsutil cp index.js gs://newbuvkypip
Copying file://index.js [Content-Type=application/javascript]...
- [1 files] 171.0 B/ 171.0 B
Operation completed over 1 objects/171.0 B.

C:\Users\elnus\Downloads\my-cloud-function>gcloud functions logs read notify
LEVEL NAME EXECUTION_ID TIME_UTC LOG
D     notify iy36v3sflxu5 2024-11-17 15:29:38.548 Function execution took 14 ms, finished with status: 'ok'
D     notify iy36v3sflxu5 2024-11-17 15:29:38.546 File index.js uploaded to newbuvkypip.
D     notify iy36v3sflxu5 2024-11-17 15:29:38.534 Function execution started

C:\Users\elnus\Downloads\my-cloud-function>

C:\Users\elnus\Downloads\my-flask-app> cd .
C:\Users\elnus\Downloads\my-flask-app>cd /
C:\>cd C:\Users\elnus\Downloads
C:\Users\elnus\Downloads>mkdir my-cloud-function
C:\Users\elnus\Downloads>cd my-cloud-function
C:\Users\elnus\Downloads\my-cloud-function>npm init -y
Wrote to C:\Users\elnus\Downloads\my-cloud-function\package.json:

{
  "name": "my-cloud-function",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}

C:\Users\elnus\Downloads\my-cloud-function>npm install @google-cloud/storage
added 69 packages, and audited 70 packages in 7s
8 packages are looking for funding
  run 'npm fund' for details
found 0 vulnerabilities

C:\Users\elnus\Downloads\my-cloud-function>
C:\Users\elnus\Downloads\my-cloud-function>
```

```
uirement index X main.py - +  
File Edit View  
exports.notify = (event, context) => {  
    const file = event.name;  
    const bucket = event.bucket;  
    console.log(`File ${file} uploaded to bucket ${bucket}`);  
};
```

```
File Edit View  
import functions_framework  
@functions_framework.cloud_event  
def notify(event):  
    bucket = event.data["bucket"]  
    name = event.data["name"]  
    print(f"File {name} uploaded to {bucket}.")
```

And main py :

This is made for notifying.

Then I deploy it to the cloud function:

```
gcloud functions deploy notify --runtime python39 --trigger-resource buck --trigger-event google.storage.object.finalize
```

Then we upload the file here by gsutil cp file.txt gs://buck

gcloud functions logs read notify

```
C:\Users\elnus\Downloads\my-cloud-function>gsutil cp index.js gs://newbuvkpipl  
Copying file://index.js [Content-Type=application/javascript]...  
- [1 files][ 171.0 B/ 171.0 B]  
Operation completed over 1 objects/171.0 B.  
  
C:\Users\elnus\Downloads\my-cloud-function>gcloud functions logs read notify  
LEVEL NAME EXECUTION_ID TIME_UTC LOG  
D     notify iy36v3sf1xu5 2024-11-17 15:29:38.548 Function execution took 14 ms, finished with status: 'ok'  
D     notify iy36v3sf1xu5 2024-11-17 15:29:38.546 File index.js uploaded to newbuvkpipl.  
D     notify iy36v3sf1xu5 2024-11-17 15:29:38.534 Function execution started  
  
C:\Users\elnus\Downloads\my-cloud-function>
```

Exercise 9: Monitoring and Logging

- Set up monitoring and logging for your App Engine application and Cloud Functions.
- Use Google Cloud's monitoring tools to analyze the performance and logs of your deployed services.

To setup the monitoring i make custom dashboard here:

The screenshot shows two tabs open in a browser window:

- Monitoring Dashboard:** A "New Dashboard - Nov 17, 2024 8:31 PM" tab. It features a "Untitled Logs Panel" with a log entry list and a chart titled "Global - Bytes streamed into log buckets [SUM]". The chart shows data from 9:40 PM to 10:20 PM, with values fluctuating between 2KB/s and 8KB/s.
- Logs Explorer:** A "Logs Explorer" tab showing logs for a "hello-world" application. The timeline shows log entries from Nov 17, 8:31 PM to 9:11 PM. One entry is highlighted: "2024-11-17 21:35:06.447 UTC+5:45 [6208.93] hello-world 2024/11/17 15:26:58.678592353Z duration=PT1H". The log details pane shows nested fields like insertId and labels.

Results

Key Google Cloud services, such as App Engine, Cloud Functions, and monitoring tools, were made practical through the exercises.

App.yaml was used to establish runtime and service configurations for a basic Flask application that was uploaded to Google App Engine for Exercise 7. A misconfigured app.yaml was the cause of the initial deployment issues; this was fixed by fixing the syntax and defining the runtime and entry point. The application was successfully tested after deployment, showcasing how App Engine makes web application hosting easier.

Creating a Cloud Function that was triggered by a Cloud Storage event was the task for Exercise 8. The purpose of the function was to notify the user when a file upload was detected. A bucket location issue that was managed by making sure both resources were in the same place as the function's trigger region and a permissions issue that was overcome by giving the Cloud Storage service account the necessary roles/pubsub.publisher IAM role were among the challenges. These challenges demonstrated how crucial exact IAM role configuration and regional alignment are for event-driven systems.

Both App Engine and Cloud Functions monitoring and logging were set up in Exercise 9. Cloud Monitoring was used to track metrics like error rates and response times, and alerting rules were set up. Alert levels had to be changed during testing in order to represent actual application performance. These tools shown their worth in preserving the performance visibility and health of applications.

Conclusion

The adaptability and integration of Google Cloud services in contemporary cloud architecture were highlighted by this set of exercises. The need for appropriately allocated IAM roles was demonstrated by the importance of identity and security management in maintaining seamless operations and preventing deployment problems. Cloud Functions enabled serverless, event-driven operations with great flexibility, while App Engine worked well for developing straightforward, scalable applications with no overhead. Proactive troubleshooting was made possible by monitoring and logging technologies, which also offered insights into application performance. When combined, these services demonstrated Google Cloud's resilience and versatility for creating cloud-native apps that are scalable, safe, and effective.

References

Google Cloud Documentation

https://cloud.google.com/docs/?gad_source=1&gclid=CjwKCAiAxea5BhBeEiwAh4t5K13qhD9zANEVnnnjlcqXosy7PHAVeQk4fJOhOJ3oSmUwkOQzVERHRoCtf0QAvD_BwE&gclsrc=aw.ds

Cloud Functions Overview <https://cloud.google.com/functions/docs>