Assignment 4, Cloud App Dev
Application Security Best Practices and Scaling Applications on Google Cloud

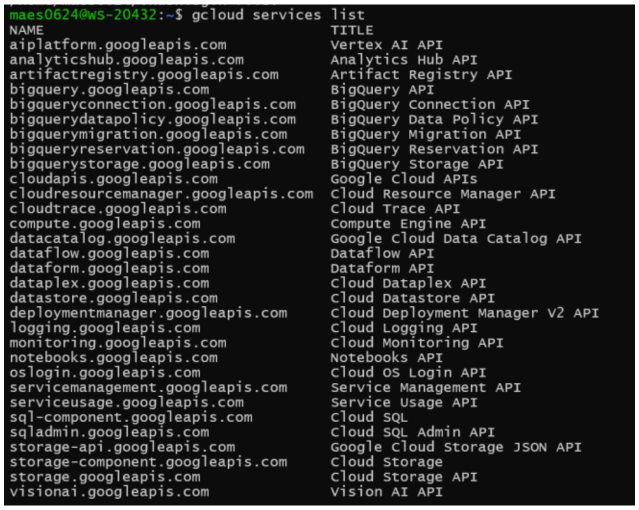
Yessimseit Manarbek

5.11.2024

1. **Executive Summary**
   As part of this assignment, we implemented security practices into our google cloud platform resources and systems. We also set up scalability options for our application running in kubernetes using the Googke kubernetes engine platform.
2. **Table of Contents**

3. **Introduction**
   Google Cloud is a cloud platform that provides computing power, services and platform-as-a-service services (Kubernetes Engine, App engine, Cloud storage). With GCP, we deploy our applications using all the ready-made features. One of the key factors in the operation of applications is their security and scalability. Google Cloud has taken care of everything, and it provides every opportunity to improve security and scalability.
   In this report, I wrote how we use Key management to create encryption keys, how we create Load balancers to work over HTTPS connections. We also adjusted the scalability of our application running in the kubernetes GKE cluster. We adjusted the horizontal and vertical scaling. We also set up monitoring and alert systems. We also figured out performance and cost optimization.
4. **Application Security Best Practices**
   - **Overview of Cloud Security**
     - Explanation of the shared responsibility model.

During this task we will try use best security practices. We will implement their into our Midterm To-Do task. This means for example we will encrypt data on Cloud storage bucket use encrypt key on Google key management.
In order to ensure encryption in our web application on the network, we will use the secure HTTPS protocol using an SSL certificate that we will create and upload to Google Cloud. We will set up traffic to our application via Load Balancer and set HTTPS.
Firstly we enable all needed to our API. I have included all the APIs I need, among all this list we will need to use Compute Engine, Cloud SQL, Cloud Storage



```
maes0624@ws-20432:~$ gcloud services list
NAME                                  TITLE
aiplatform.googleapis.com             Vertex AI API
analyticshub.googleapis.com           Analytics Hub API
artifactregistry.googleapis.com       Artifact Registry API
bigquery.googleapis.com               BigQuery API
bigqueryconnection.googleapis.com     BigQuery Connection API
bigquerydatapolicy.googleapis.com     BigQuery Data Policy API
bigquerymigration.googleapis.com      BigQuery Migration API
bigqueryreservation.googleapis.com    BigQuery Reservation API
bigquerystorage.googleapis.com        BigQuery Storage API
cloudapis.googleapis.com              Google Cloud APIs
cloudresourcemanager.googleapis.com   Cloud Resource Manager API
cloudtrace.googleapis.com             Cloud Trace API
compute.googleapis.com                Compute Engine API
datacatalog.googleapis.com            Google Cloud Data Catalog API
dataflow.googleapis.com               Dataflow API
dataform.googleapis.com               Dataform API
dataplex.googleapis.com               Cloud Dataplex API
datastore.googleapis.com              Cloud Datastore API
deploymentmanager.googleapis.com      Cloud Deployment Manager V2 API
logging.googleapis.com                Cloud Logging API
monitoring.googleapis.com             Cloud Monitoring API
notebooks.googleapis.com              Notebooks API
oslogin.googleapis.com                Cloud OS Login API
servicemanagement.googleapis.com      Service Management API
serviceusage.googleapis.com           Service Usage API
sql-component.googleapis.com          Cloud SQL
sqladmin.googleapis.com               Cloud SQL Admin API
storage-api.googleapis.com            Google Cloud Storage JSON API
storage-component.googleapis.com      Cloud Storage
storage.googleapis.com                Cloud Storage API
visionai.googleapis.com               Vision AI API
```

- **IAM Configuration**
  - Discussion of least privilege and conditions implemented.

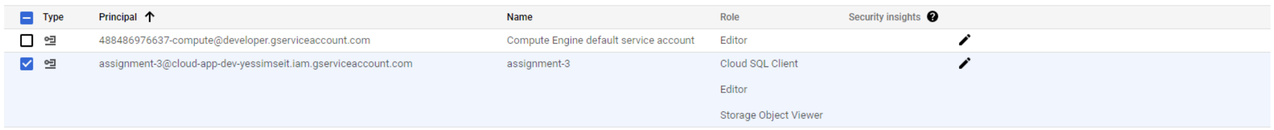
Here we will create a service account with the principle of minimum privileges. This means that we will provide only the minimum access that is needed for our application to work. For example, for changes or in order to upload data to Google Bucket, we only need the access level ‘Editor’ and we will not give our service account access above ‘editor’, for example, if we gave it ‘owner’

To create go Menu → IAM & Admin → Service Accounts. Put Create Service Account and give least privilege

For example:

To access Cloud Storage: Storage Object Viewer

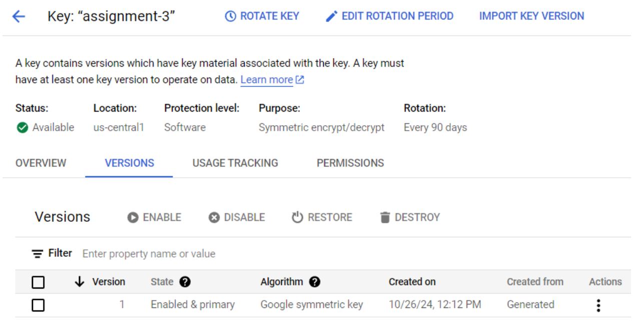
To access Cloud SQL: Cloud SQL client



Type	Principal	Name	Role	Security insights
☐	488486976637-compute@developer.gserviceaccount.com	Compute Engine default service account	Editor	
☑	assignment-3@cloud-app-dev-yessimseit.iam.gserviceaccount.com	assignment-3	Cloud SQL Client Editor Storage Object Viewer	

- **Data Protection**
  - Methods used for encrypting data at rest and in transit.

We will set up encryption for data at rest using Google Cloud KMS and Use HTTPS for data in transit and configure load balancers to enforce SSL.
Firstly On Google Cloud → Key management we create google symmetric encrypt key on us-central1 zone.



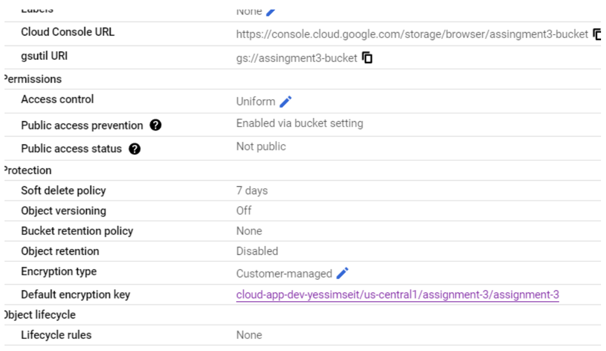
Key: “assignment-3”

A key contains versions which have key material associated with the key. A key must have at least one key version to operate on data. Learn more

Status:	Location:	Protection level:	Purpose:	Rotation:
Available	us-central1	Software	Symmetric encrypt/decrypt	Every 90 days

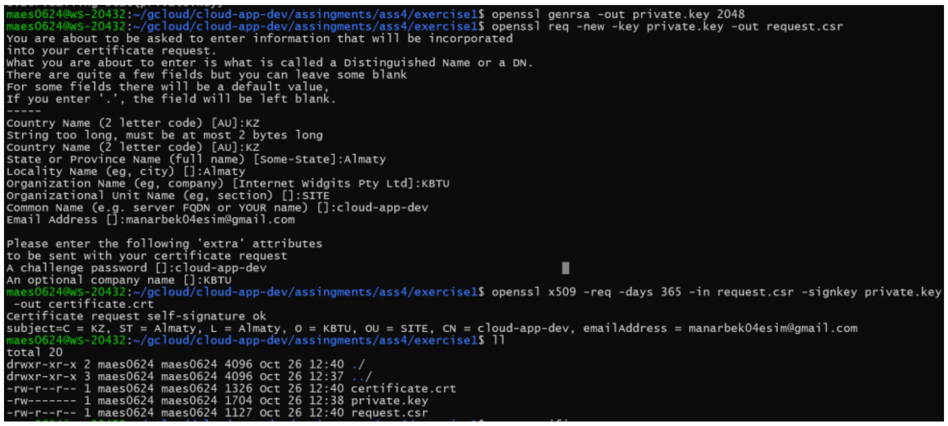
	Version	State	Algorithm	Created on	Created from	Actions
☐	1	Enabled & primary	Google symmetric key	10/26/24, 12:12 PM	Generated	⋮

Then we implemeny this encrypt key to out google bucket for encrypt data



Labels	None
Cloud Console URL	https://console.cloud.google.com/storage/browser/assingment3-bucket
gsutil URI	gs://assingment3-bucket
ermissions	
Access control	Uniform
Public access prevention	Enabled via bucket setting
Public access status	Not public
rotection	
Soft delete policy	7 days
Object versioning	Off
Bucket retention policy	None
Object retention	Disabled
Encryption type	Customer-managed
Default encryption key	cloud-app-dev-yessimseit/us-central1/assignment-3/assignment-3
bject lifecycle	
Lifecycle rules	None

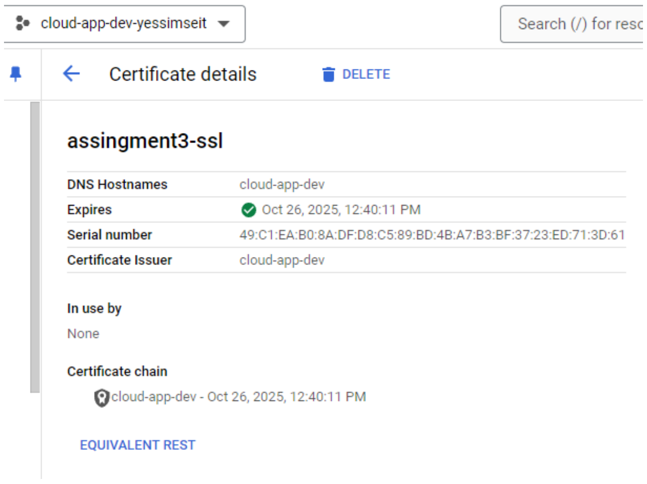
Next using openssl we generate certificate with encryption key



```
maes0624@ws-20432:~/gcloud/cloud-app-dev/assingments/ass4/exercise1$ openssl genrsa -out private.key 2048
maes0624@ws-20432:~/gcloud/cloud-app-dev/assingments/ass4/exercise1$ openssl req -new -key private.key -out request.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:KZ
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:KZ
State or Province Name (full name) [Some-State]:Almaty
Locality Name (eg, city) []:Almaty
Organization Name (eg, company) [Internet Widgits Pty Ltd]:KBTU
Organizational Unit Name (eg, section) []:SITE
Common Name (e.g. server FQDN or YOUR name) []:cloud-app-dev
Email Address []:manarbek04esim@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:cloud-app-dev
An optional company name []:KBTU
maes0624@ws-20432:~/gcloud/cloud-app-dev/assingments/ass4/exercise1$ openssl x509 -req -days 365 -in request.csr -signkey private.key
 -out certificate.crt
Certificate request self-signature ok
subject=C = KZ, ST = Almaty, L = Almaty, O = KBTU, OU = SITE, CN = cloud-app-dev, emailAddress = manarbek04esim@gmail.com
maes0624@ws-20432:~/gcloud/cloud-app-dev/assingments/ass4/exercise1$ ll
total 20
drwxr-xr-x 2 maes0624 maes0624 4096 Oct 26 12:40 ./
drwxr-xr-x 3 maes0624 maes0624 4096 Oct 26 12:37 ../
-rw-r--r-- 1 maes0624 maes0624 1326 Oct 26 12:40 certificate.crt
-rw------- 1 maes0624 maes0624 1704 Oct 26 12:38 private.key
-rw-r--r-- 1 maes0624 maes0624 1127 Oct 26 12:40 request.csr
```

Then we upload this certificate to certificate manager on google cloud. We will use this certificate when create load balancer with HTTPS connection to be secure.



assingment3-ssl

DNS Hostnames	cloud-app-dev
Expires	Oct 26, 2025, 12:40:11 PM
Serial number	49:C1:EA:B0:8A:DF:D8:C5:89:BD:4B:A7:B3:BF:37:23:ED:71:3D:61
Certificate Issuer	cloud-app-dev

In use by

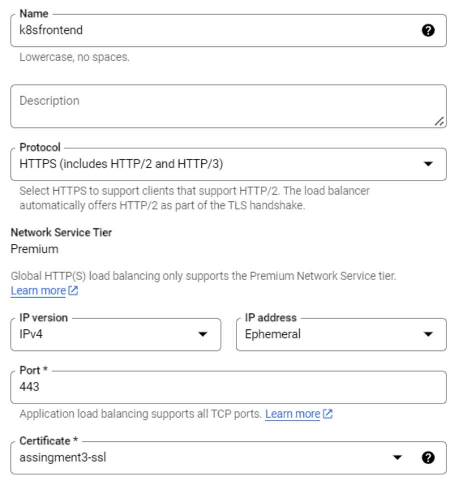
None

Certificate chain

cloud-app-dev - Oct 26, 2025, 12:40:11 PM

Next go to Network Services → Load Balancing and create Load Balancer. I Will create load balancer to application uploaded on Google Kubernetes

Frontend Configuration


Name
k8sfrontend
Lowercase, no spaces.

Description

Protocol
HTTPS (includes HTTP/2 and HTTP/3)
Select HTTPS to support clients that support HTTP/2. The load balancer automatically offers HTTP/2 as part of the TLS handshake.

Network Service Tier
Premium

Global HTTP(S) load balancing only supports the Premium Network Service tier.
Learn more

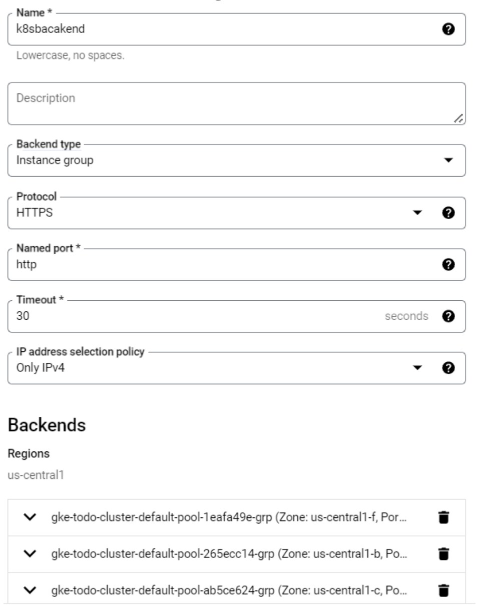
IP version
IPv4

IP address
Ephemeral

Port *
443
Application load balancing supports all TCP ports. Learn more

Certificate *
assingment3-ssl

Backend Configuration


Name *
k8sbacakend
Lowercase, no spaces.

Description

Backend type
Instance group

Protocol
HTTPS

Named port *
http

Timeout *
30 seconds

IP address selection policy
Only IPv4

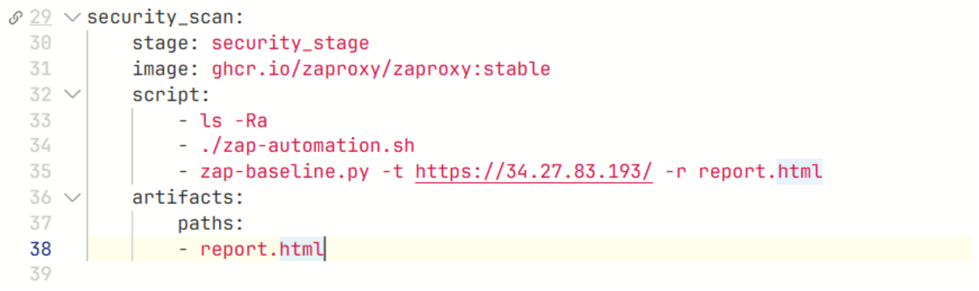
Backends

Regions
us-central1

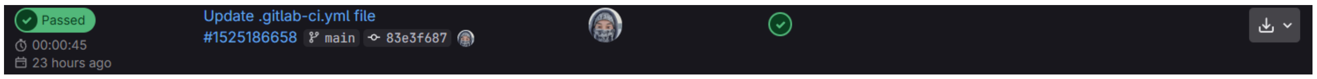
gke-todo-cluster-default-pool-1eafa49e-grp (Zone: us-central1-f, Por...
gke-todo-cluster-default-pool-265ecc14-grp (Zone: us-central1-b, Po...
gke-todo-cluster-default-pool-ab5ce624-grp (Zone: us-central1-c, Po...

- ### Security Testing
  - Tools integrated into the CI/CD pipeline and findings from the vulnerability assessment.

In this task we need integrate a security scanning tool (e.g., Snyk, OWASP ZAP) into your CI/CD pipeline. I will use Gitlab platform to run gitlab pipeline. We will use OWASP ZAP tool.


```
security_scan:
    stage: security_stage
    image: ghcr.io/zaproxy/zaproxy:stable
    script:
        - ls -Ra
        - ./zap-automation.sh
        - zap-baseline.py -t https://34.27.83.193/ -r report.html
    artifacts:
        paths:
        - report.html
```

By running an NMAP security scan of our application in pipeline, we will save the report in html and look at the vulnerability.


Passed
00:00:45
23 hours ago
Update .gitlab-ci.yml file
#1525186658 main 83e3f687

After downloading the report, we analyze all the detected problems and the presence of vulnerabilities.

Risk Level	Number of Alerts
High	0
Medium	2
Low	3
Informational	1
False Positives:	0

Alerts

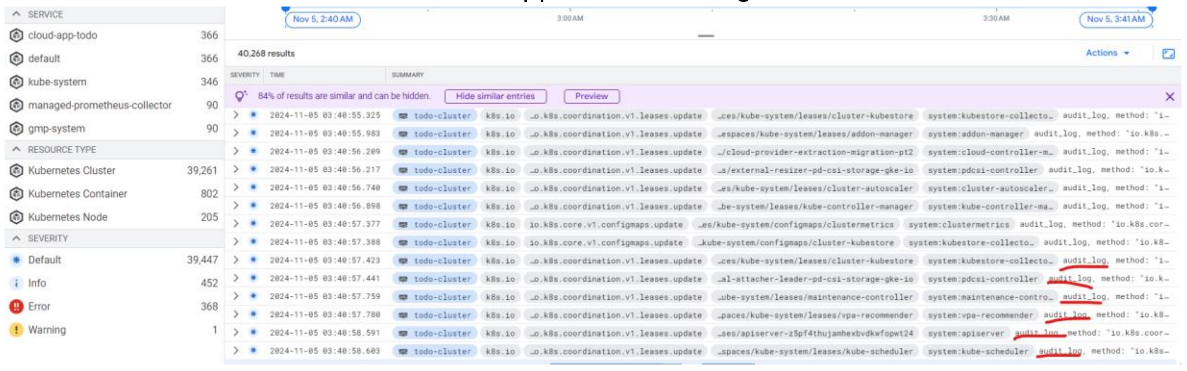
Name	Risk Level	Number of Instances
Content Security Policy (CSP) Header Not Set	Medium	4
Missing Anti-clickjacking Header	Medium	2
Permissions Policy Header Not Set	Low	4
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	4
X-Content-Type-Options Header Missing	Low	2
Storable and Cacheable Content	Informational	4

The absence of the Content Security Policy (CSP) header is the level of risk: Average
You need to configure the server and add the Content-Security-Policy header. You need to specify trusted sources for scripts and images in order to limit the loading of unsafe resources.
Lack of Anti-clickjacking header - Risk level: Average
There is no protection against clickjacking attacks when attackers can embed an invisible iframe into our site. To solve the problem, you need to use the X-Frame-Options header with the value DENY/SAMEORIGIN to limit the ability to load your site into an IFRAME from third-party resources.
No header Permissions Policy - Risk Level: Low
This header controls the browser's access to various functions such as the camera and microphone.
The server version is visible in the "Server" header - Risk level: Low.
The server discloses information about the server version, which can help attackers identify vulnerabilities related to a specific software version.

Since my application is a TO-do List, I am doing this task as part of my experience working with pipeline and owasp zap. Ideally, in pipeline, security verification is an important step before deploying an application in a production environment. As part of the CI process, we assemble the application and check its security. After we receive the report and fix all the vulnerabilities, we must restart the scan to make sure that our application is safe. Then we can deploy our application to the production environment.

- **Monitoring and Logging**
    - Overview of monitoring solutions and security alerts set up.

During this taks we enable Audit logs. Go to Logs Explorer To make sure that our log collection system is working perfectly. As you can see, logging is working fine, our latest logs are linked to our to-do kubernetes cluster where we have our To-DO List application running



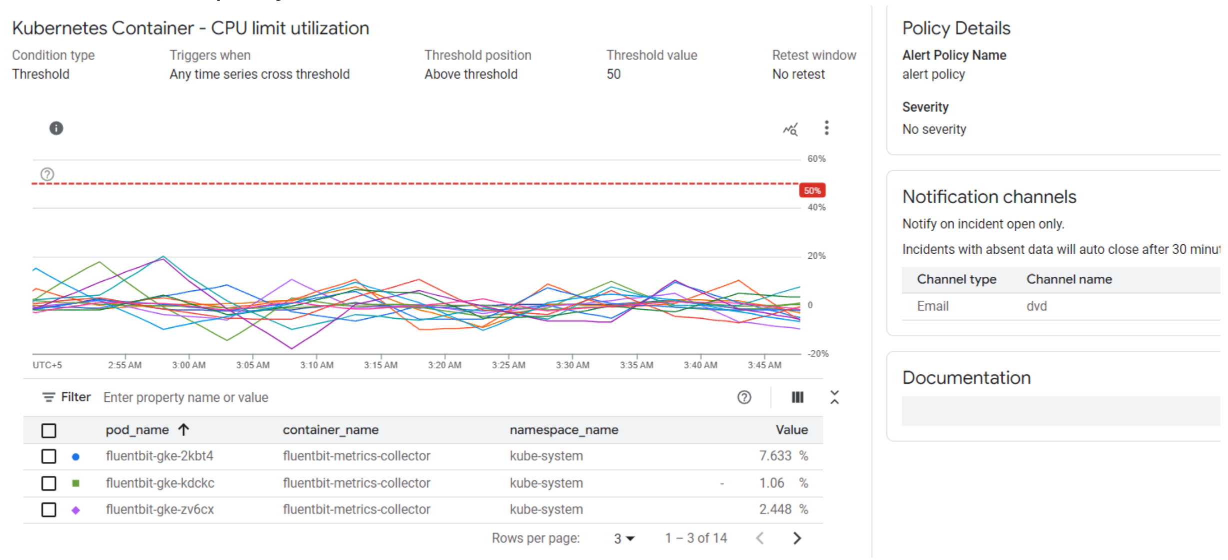
cloud-app-todo	366
default	366
kube-system	346
managed-prometheus-collector	90
gmp-system	90
Kubernetes Cluster	39,261
Kubernetes Container	802
Kubernetes Node	205
Default	39,447
Info	452
Error	368
Warning	1

40,268 results

```
2024-11-05 03:40:55.325  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …ces/kube-system/leases/cluster-kubestore  system:kubestore-collecto…  audit_log, method: "i…
2024-11-05 03:40:55.983  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …espaces/kube-system/leases/addon-manager  system:addon-manager  audit_log, method: "io.k8s.…
2024-11-05 03:40:56.209  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …/cloud-provider-extraction-migration-pt2  system:cloud-controller-m…  audit_log, method: "i…
2024-11-05 03:40:56.217  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …s/external-resizer-pd-csi-storage-gke-io  system:pdcsi-controller  audit_log, method: "io.k…
2024-11-05 03:40:56.740  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …es/kube-system/leases/cluster-autoscaler  system:cluster-autoscaler…  audit_log, method: "i…
2024-11-05 03:40:56.898  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …be-system/leases/kube-controller-manager  system:kube-controller-ma…  audit_log, method: "i…
2024-11-05 03:40:57.377  todo-cluster  k8s.io  io.k8s.core.v1.configmaps.update  …es/kube-system/configmaps/clustermetrics  system:clustermetrics  audit_log, method: "io.k8s.cor…
2024-11-05 03:40:57.388  todo-cluster  k8s.io  io.k8s.core.v1.configmaps.update  …kube-system/configmaps/cluster-kubestore  system:kubestore-collecto…  audit_log, method: "io.k8…
2024-11-05 03:40:57.423  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …ces/kube-system/leases/cluster-kubestore  system:kubestore-collecto…  audit_log, method: "i…
2024-11-05 03:40:57.441  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …al-attacher-leader-pd-csi-storage-gke-io  system:pdcsi-controller  audit_log, method: "io.k…
2024-11-05 03:40:57.759  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …ube-system/leases/maintenance-controller  system:maintenance-contro…  audit_log, method: "i…
2024-11-05 03:40:57.780  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …paces/kube-system/leases/vpa-recommender  system:vpa-recommender  audit_log, method: "io.k8…
2024-11-05 03:40:58.591  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …ses/apiserver-z5pf4thujamhexbvdkwfopwt24  system:apiserver  audit_log, method: "io.k8s.coor…
2024-11-05 03:40:58.603  todo-cluster  k8s.io  …o.k8s.coordination.v1.leases.update  …spaces/kube-system/leases/kube-scheduler  system:kube-scheduler  audit_log, method: "io.k8s…
```

Next **Set Up Alerts Using Google Cloud Monitoring**

We create Alert policy – Kubernetes container CPU limit utilization.



Kubernetes Container - CPU limit utilization

Condition type	Triggers when	Threshold position	Threshold value	Retest window
Threshold	Any time series cross threshold	Above threshold	50	No retest

Filter Enter property name or value

	pod_name ↑	container_name	namespace_name	Value
☐ ●	fluentbit-gke-2kbt4	fluentbit-metrics-collector	kube-system	7.633 %
☐ ■	fluentbit-gke-kdckc	fluentbit-metrics-collector	kube-system	1.06 %
☐ ◆	fluentbit-gke-zv6cx	fluentbit-metrics-collector	kube-system	2.448 %

Rows per page: 3 1 – 3 of 14

Policy Details

Alert Policy Name
alert policy

Severity
No severity

Notification channels

Notify on incident open only.

Incidents with absent data will auto close after 30 minut

Channel type	Channel name
Email	dvd

Documentation

This means that if in any container in the cluster the percentage of CPU utilization is greater than or equal to 50 percent of the desired value, we received a notification about it in the dvd channel. The mail is set up in the dvd channel manarbek04esim@gmail.com . It is to this email that we will receive a notification. Alerts help us with the fact that if our system is overloaded, we receive a notification and can quickly respond to it.

- **Incident Response**
  - Description of the incident response plan and simulation results.

We need to create an individual incident response plan. We need to do 5 main stages of incident response.
1) Identification - you need to set up incident detection and analysis.
2) Stopping the spread of the threat
3) Elimination of the root cause of the incident
4) System Recovery
5) Analysis and improvement of the plan

We need to form the roles and responsibilities of who will be responsible for the response. It is necessary to determine which resources are the most significant and which are not. You need to set up monitoring and alerts. You need to make a backup, for example, Cloud storage. And of course, document the whole process to analyze all this and try to fix all the vulnerabilities.

After creating a plan, you can arrange a simulation of the incident. For example
, an unauthorized access attempt. At the identification stage, it is necessary to stimulate notification of suspicious login activity from an unknown IP address. Next, you need to restrict access to systems that could be affected by the threat. Next, you need to eliminate and fix vulnerabilities. At the recovery stage, you need to restart everything, change passwords, change encryption, and certificates.

5. **Scaling Applications on Google Cloud**
   - **Overview of Scalability**
     - Importance of scalability in cloud architecture.

Scalability in applications running in cloud/ Kubernetes clusters helps applications cope with high workloads during peak application usage. And, accordingly, on the contrary, it makes it possible to reduce

resources with a decrease in demand for applications. Together with the approach of cloud platforms such as GCP, a business can pay exactly as much as is required for its application to work: In normal times, maintain resources in average values for stable operation of the application, in times of high loads (for example, Black Friday) pay for a large number of resources used to withstand the load and in times of low load, on the contrary, reduce costs for resources and not to overpay.
As part of this assignment, I decided to use my ToDo List application from midterm. We will consider the option of deploying the application using Google Kubernetes Engine.
As part of scaling in the kubernetes cluster, we need to understand that we are talking about **'Pod'**. Our applications are running in the pod inside the container.
Pod resources include: memory and CPU, as well as the number of Pods, which in kubernetes is designated as **'replicas**'. All these resources affect the performance of the system.

- **Application Design**
    - Description of the web application created and the architecture used.

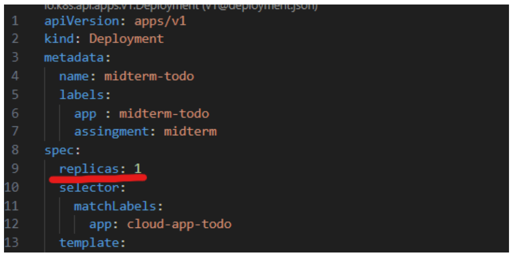
In this assignment, I will use my To-Do List application written in python. We can say that this is a monolithic application. I built a Docker image of my application and uploaded it to google artifactory. Next, I will deploy my application in kubernetes Google Kubernetes Engine. My application works via HTTP methods.

- **Scaling Methods**

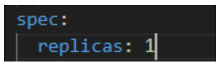
We will consider scaling methods within the of the pod in the kubernetes cluster:
Horizontal – when the number of replicas of the pod changes (the number of instances increases and decreases)
Vertical – when the power of the pods increases: that is, by adding the values of cpu and memory.
I will be modifying the deployment manifest deployment.yaml changing resource parameters demonstrating vertical and horizontal scaling.

Horizontal scaling

In `default', the replica values are 1, by changing the values from 1 for example to 2 or more, we increase the power of our application, since now our application will work with 2x power

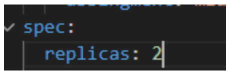


```
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
apiVersion: apps/v1
kind: Deployment
metadata:
  name: midterm-todo
  labels:
    app : midterm-todo
    assingment: midterm
spec:
  replicas: 1
  selector:
    matchLabels:
      app: cloud-app-todo
  template:
```



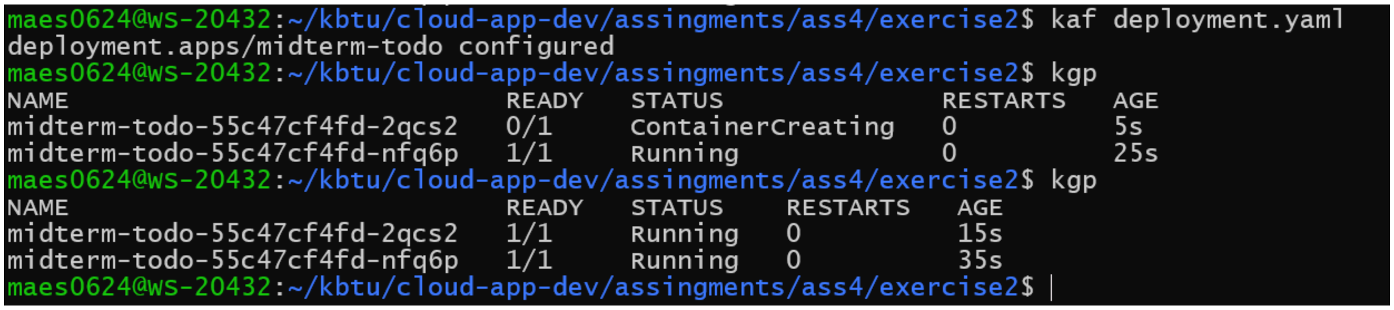
```
spec:
  replicas: 1
```

->



```
spec:
  replicas: 2
```

Now we are applying a new deployment and we can make sure that our change has been applied.
Using the kubectl get pods command we can see that the number of pods has increased

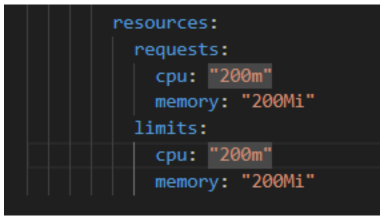


```
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf deployment.yaml
deployment.apps/midterm-todo configured
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kgp
NAME                            READY   STATUS              RESTARTS   AGE
midterm-todo-55c47cf4fd-2qcs2   0/1     ContainerCreating   0          5s
midterm-todo-55c47cf4fd-nfq6p   1/1     Running             0          25s
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kgp
NAME                            READY   STATUS    RESTARTS   AGE
midterm-todo-55c47cf4fd-2qcs2   1/1     Running   0          15s
midterm-todo-55c47cf4fd-nfq6p   1/1     Running   0          35s
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ |
```

We have successfully implemented horizontal scaling

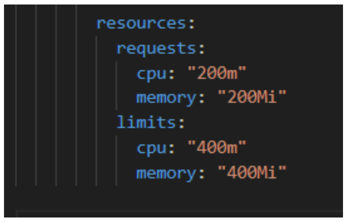
Vertical scaling:

I have assigned the base resources to the Pod. If we want to scale our application, we need to increase the capacity of the Pod in which the container is running, thus our application will be able to withstand a heavy load. To do this, we will increase all resource limits indicators, thus our application will have more and more resources in stock.



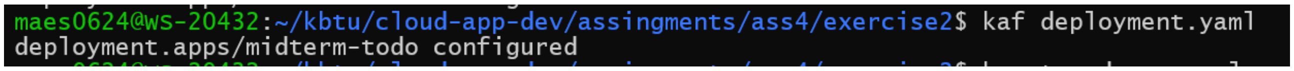
```
resources:
  requests:
    cpu: "200m"
    memory: "200Mi"
  limits:
    cpu: "200m"
    memory: "200Mi"
```

→



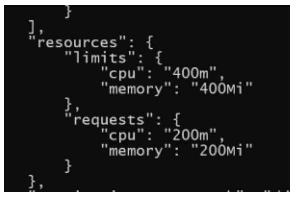
```
resources:
  requests:
    cpu: "200m"
    memory: "200Mi"
  limits:
    cpu: "400m"
    memory: "400Mi"
```

Applying our deployment



```
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf deployment.yaml
deployment.apps/midterm-todo configured
```

How you see new configuration applied:

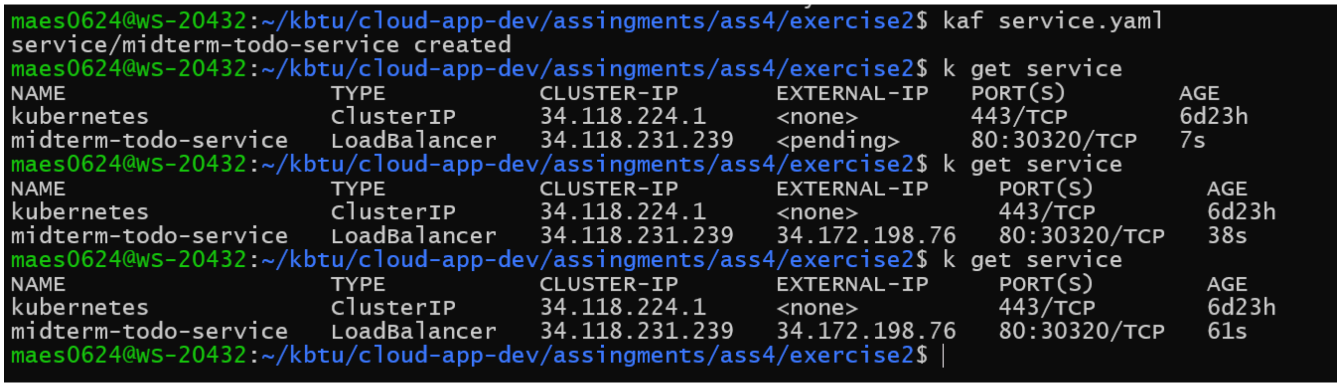


```
    }
  ],
  "resources": {
      "limits": {
          "cpu": "400m",
          "memory": "400Mi"
      },
      "requests": {
          "cpu": "200m",
          "memory": "200Mi"
      }
  },
```

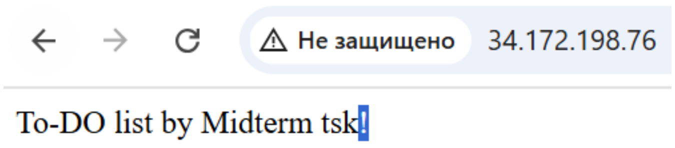
Comparison:
Horizontal scaling is more preferable as it is best suited for applications that require high availability and scalability, such as web applications with changing traffic. For vertical scaling, we needed a reboot, whereas for horizontal scaling, we only wait until the new pod starts working, after which kubernetes itself would begin to balance the load on all working pod. I would say that this type of scaling is more suitable for stateless applications.

- **Load Balancing**
    - Configuration details for Google Cloud Load Balancing and health checks.

Firstly we apply our load-balancer



```
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf service.yaml
service/midterm-todo-service created
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get service
NAME                   TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes             ClusterIP      34.118.224.1     <none>        443/TCP        6d23h
midterm-todo-service   LoadBalancer   34.118.231.239   <pending>     80:30320/TCP   7s
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get service
NAME                   TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)        AGE
kubernetes             ClusterIP      34.118.224.1     <none>          443/TCP        6d23h
midterm-todo-service   LoadBalancer   34.118.231.239   34.172.198.76   80:30320/TCP   38s
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get service
NAME                   TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)        AGE
kubernetes             ClusterIP      34.118.224.1     <none>          443/TCP        6d23h
midterm-todo-service   LoadBalancer   34.118.231.239   34.172.198.76   80:30320/TCP   61s
maes0624@WS-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ |
```

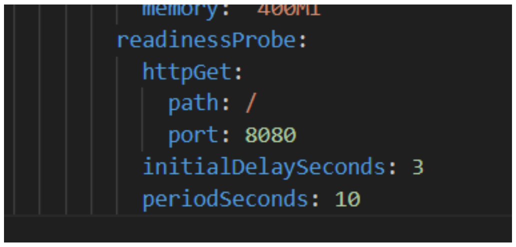


Не защищено 34.172.198.76

To-DO list by Midterm tsk!

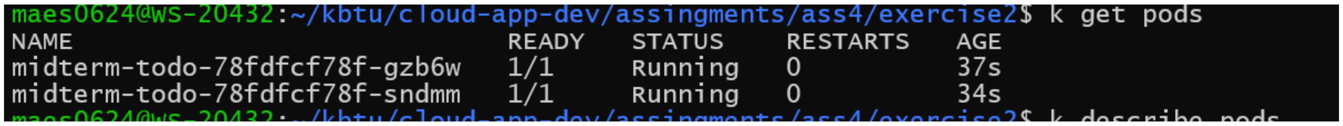
How you see our application working

Now we want to make sure that the load balancer redirects the load only to those Pods that are considered alive and working. To do this, we need to add a Readiness Probe. In GKE, this thing will check the readiness of the Pod and if it is ready, it allows you to contact our server. This way we can implement the ‘Configure health checks to ensure traffic is routed only to healthy instances' policy. To do this, we will change the Deployment.yaml file.



```
          memory: "400Mi"
        readinessProbe:
          httpGet:
            path: /
            port: 8080
          initialDelaySeconds: 3
          periodSeconds: 10
```

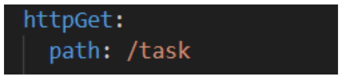
After add readinessProbe we can deploy deployment file to update pods.



```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get pods
NAME                           READY   STATUS    RESTARTS   AGE
midterm-todo-78fdfcf78f-gzb6w  1/1     Running   0          37s
midterm-todo-78fdfcf78f-sndmm  1/1     Running   0          34s
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k describe pods
```

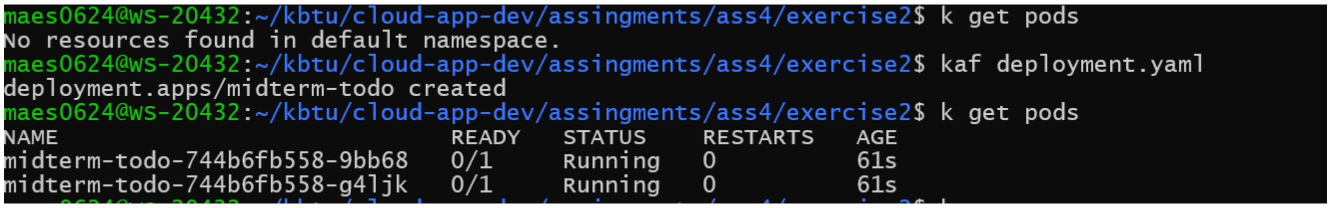
How you see our 2 pods ready and worked.

Now I want show your that health check working, I will change httpGet/path to wrong and you will see that pod not ready.



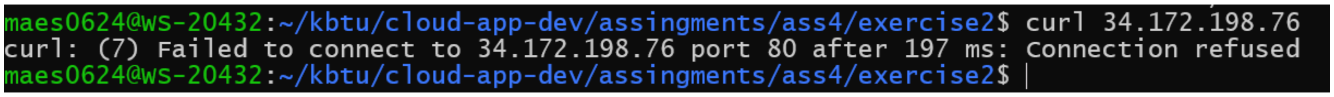
```
  httpGet:
    path: /task
```

On http path /task we We don't have anything. Because right http path **is /tasks**



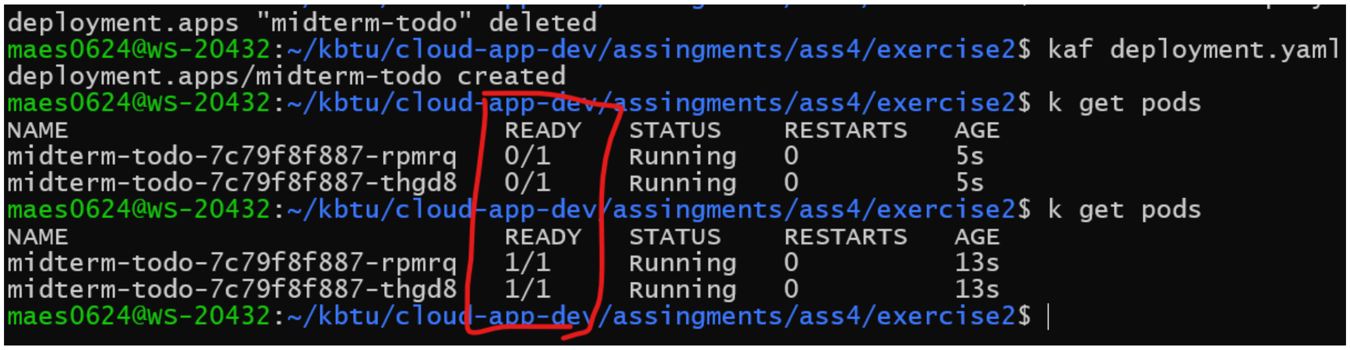
```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get pods
No resources found in default namespace.
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf deployment.yaml
deployment.apps/midterm-todo created
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get pods
NAME                            READY   STATUS    RESTARTS   AGE
midterm-todo-744b6fb558-9bb68   0/1     Running   0          61s
midterm-todo-744b6fb558-g4ljk   0/1     Running   0          61s
```

How you see our pods running, but they are not ready. So if you make curl to load balancer we don’t have anything



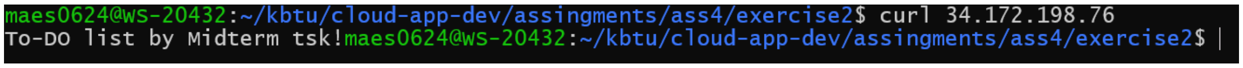
```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ curl 34.172.198.76
curl: (7) Failed to connect to 34.172.198.76 port 80 after 197 ms: Connection refused
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ |
```

Now I show what we will have if I write right http path - /tasks



```
deployment.apps "midterm-todo" deleted
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf deployment.yaml
deployment.apps/midterm-todo created
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get pods
NAME                            READY   STATUS    RESTARTS   AGE
midterm-todo-7c79f8f887-rpmrq   0/1     Running   0          5s
midterm-todo-7c79f8f887-thgd8   0/1     Running   0          5s
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ k get pods
NAME                            READY   STATUS    RESTARTS   AGE
midterm-todo-7c79f8f887-rpmrq   1/1     Running   0          13s
midterm-todo-7c79f8f887-thgd8   1/1     Running   0          13s
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ |
```

How you see after 10 sec readinessProbe check that pod ready and our pods ready and load-balancer start redirect to working pods.

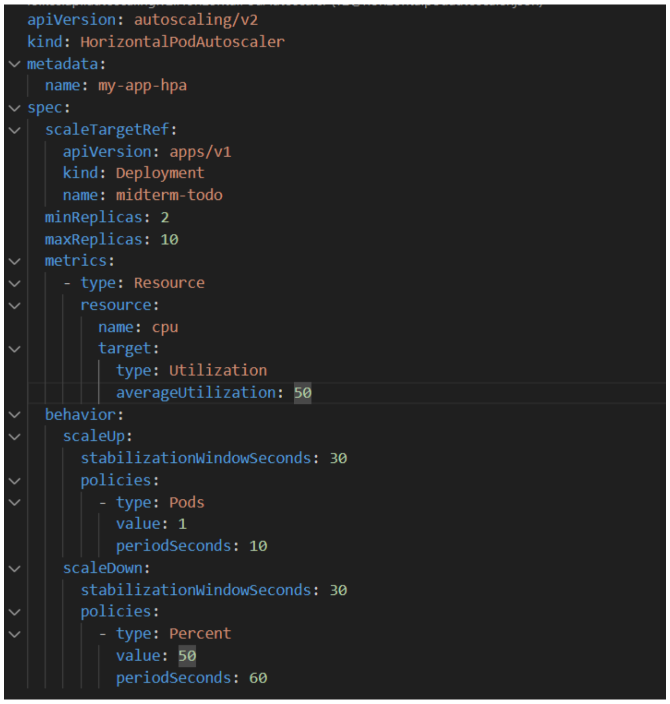


```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ curl 34.172.198.76
To-DO list by Midterm tsk!maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ |
```

- **Auto-Scaling Implementation**

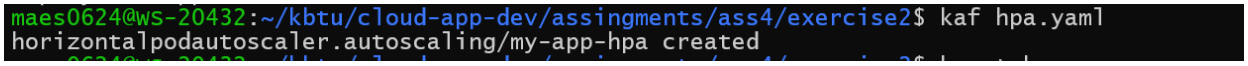
- Explanation of the auto-scaling policies established.

In this task, we will implement automatic horizontal scaling in the google kubernetes engine. As you may remember, we have already done horizontal scaling, but now we will do ‘Auto-Scaling implementation’.
To do this, we create an hpa.yaml file that includes the HorizontalPodAutoscaler manifest.
HPA will receive the metrics of the used CPU from the ‘metrics-server’ and if our condition is met, kubernetes will start adding the number of replicas of our pod. We will also add ‘Scale down’ and 'Scale up' behavior.
Let's take a look at our final file



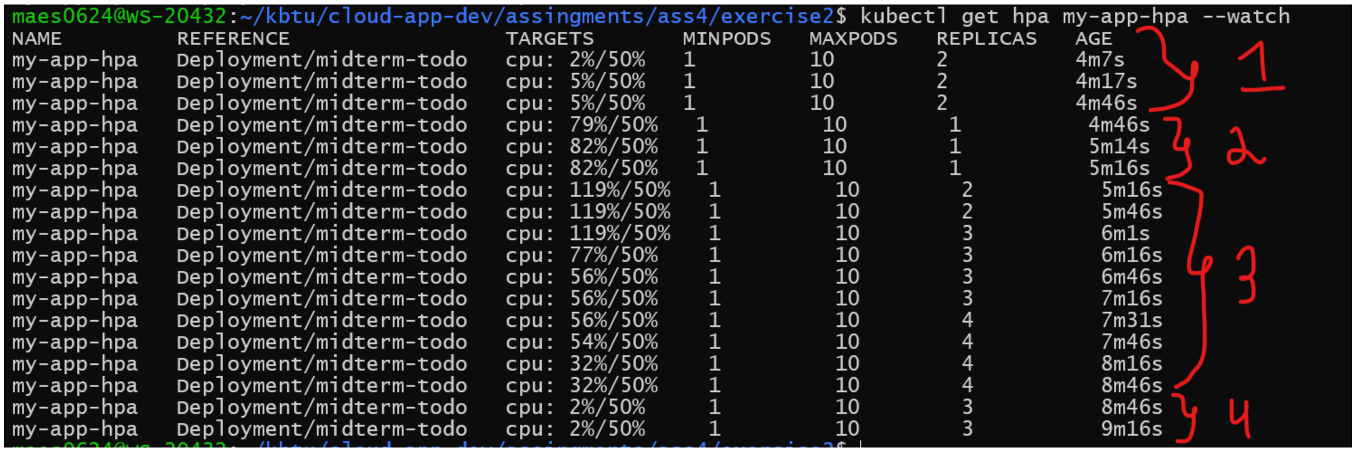
```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: my-app-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: midterm-todo
  minReplicas: 2
  maxReplicas: 10
  metrics:
    - type: Resource
      resource:
        name: cpu
        target:
          type: Utilization
          averageUtilization: 50
  behavior:
    scaleUp:
      stabilizationWindowSeconds: 30
      policies:
        - type: Pods
          value: 1
          periodSeconds: 10
    scaleDown:
      stabilizationWindowSeconds: 30
      policies:
        - type: Percent
          value: 50
          periodSeconds: 60
```

Here we have specified that the HPA controls our ‘midterm-todo’ deployment. We have specified the number of replicas: minimum 2 and maximum 10. We also announced the cpu.utilization condition: 50 percent. This means that if we specified 200 CPUs, then our system will track changes from 50 percent, that is, from 100m.
We also specified ‘behavior’, here we described the behavior when auto-scaling down or up. For example: To increase, we took a stabilization window of 30 seconds, told him to increase the number of pods by 1 unit within 10 seconds.
Next, we need to apply the manifest file via the command kubectl apply hpa.yaml and check that’s worked



```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kaf hpa.yaml
horizontalpodautoscaler.autoscaling/my-app-hpa created
```

To demonstrate the work of HPA, I have put an artificial load on my application, and we will observe the changes of horizontalPodAutoScaler live.
Using the kubectl get hpa my-app-hpa --watch command



```
maes0624@ws-20432:~/kbtu/cloud-app-dev/assingments/ass4/exercise2$ kubectl get hpa my-app-hpa --watch
NAME         REFERENCE                  TARGETS       MINPODS    MAXPODS    REPLICAS    AGE
my-app-hpa   Deployment/midterm-todo    cpu: 2%/50%    1          10         2           4m7s
my-app-hpa   Deployment/midterm-todo    cpu: 5%/50%    1          10         2           4m17s
my-app-hpa   Deployment/midterm-todo    cpu: 5%/50%    1          10         2           4m46s
my-app-hpa   Deployment/midterm-todo    cpu: 79%/50%    1          10         1           4m46s
my-app-hpa   Deployment/midterm-todo    cpu: 82%/50%    1          10         1           5m14s
my-app-hpa   Deployment/midterm-todo    cpu: 82%/50%    1          10         1           5m16s
my-app-hpa   Deployment/midterm-todo    cpu: 119%/50%    1          10         2           5m16s
my-app-hpa   Deployment/midterm-todo    cpu: 119%/50%    1          10         2           5m46s
my-app-hpa   Deployment/midterm-todo    cpu: 119%/50%    1          10         3           6m1s
my-app-hpa   Deployment/midterm-todo    cpu: 77%/50%     1          10         3           6m16s
my-app-hpa   Deployment/midterm-todo    cpu: 56%/50%     1          10         3           6m46s
my-app-hpa   Deployment/midterm-todo    cpu: 56%/50%     1          10         3           7m16s
my-app-hpa   Deployment/midterm-todo    cpu: 56%/50%     1          10         4           7m31s
my-app-hpa   Deployment/midterm-todo    cpu: 54%/50%     1          10         4           7m46s
my-app-hpa   Deployment/midterm-todo    cpu: 32%/50%     1          10         4           8m16s
my-app-hpa   Deployment/midterm-todo    cpu: 32%/50%     1          10         4           8m46s
my-app-hpa   Deployment/midterm-todo    cpu: 2%/50%      1          10         3           8m46s
my-app-hpa   Deployment/midterm-todo    cpu: 2%/50%      1          10         3           9m16s
```

1) Init time, after init HPA see that cpu show less than 50 percent and will change replicas from 2 to 1. Also at this moment I will just start my artificial load.
2) HPA put replicas to 1 and start to see that the application starts using more than 100m CPU.
3) The HPA begins to respond to the increased load, having visited the number of Pod from 1 to 2, then from 2 to 3, from 3 to 4. As you can see, gradually 4 replicas working at a power of 400m CPU or more begin to cope with the load. Therefore, it can be assumed that the HPA will stop at 4 replicas.
   At this point, I stopped the artificial load
4) After I lowered the load, the application began to consume less and less CPU, so HPA began to reduce the number of replicas.

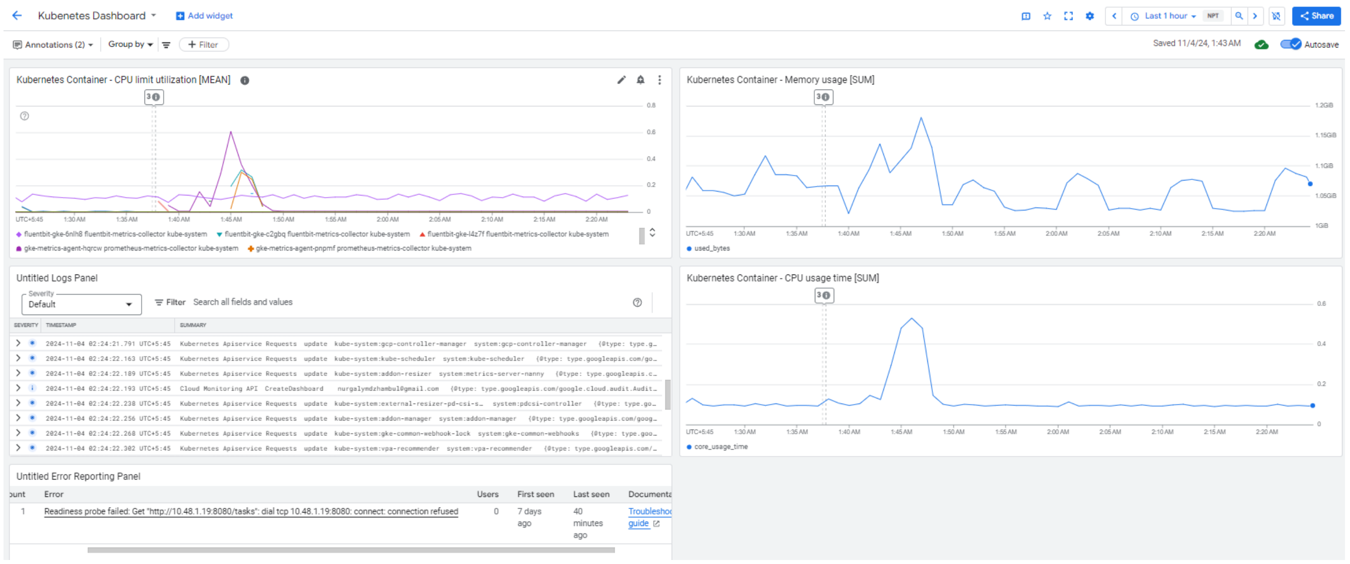
I have successfully implemented horizontal auto-scaling in GKE

- **Performance Monitoring**
  - Metrics tracked and dashboards created

I enable monitoring API and create kubernetes named dashboard that shows Kubernetes Container – Cpu limits utilization, memory usage, CPU usage.

One of the main is Logs panel that shows all logs. Here we see How GKE working, all cli commands and others.

And we have 1 error reporting portal. How you see there error related with Health Check Load balancer task when I show work of readinessProbe.


Kubenetes Dashboard
Add widget
Last 1 hour
Share
Annotations (2)
Group by
Filter
Saved 11/4/24, 1:43 AM
Autosave
Kubernetes Container - CPU limit utilization [MEAN]
fluentbit-gke-6nlh8 fluentbit-metrics-collector kube-system
fluentbit-gke-c2gbq fluentbit-metrics-collector kube-system
fluentbit-gke-l4z7f fluentbit-metrics-collector kube-system
gke-metrics-agent-hqrcw prometheus-metrics-collector kube-system
gke-metrics-agent-pnpmf prometheus-metrics-collector kube-system
Kubernetes Container - Memory usage [SUM]
used_bytes
Kubernetes Container - CPU usage time [SUM]
core_usage_time
Untitled Logs Panel
Severity
Default
Filter
Search all fields and values
Untitled Error Reporting Panel

```
SEVERITY TIMESTAMP                          SUMMARY
2024-11-04 02:24:21.791 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:gcp-controller-manager  system:gcp-controller-manager  {@type: type.g…
2024-11-04 02:24:22.163 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:kube-scheduler  system:kube-scheduler  {@type: type.googleapis.com/go…
2024-11-04 02:24:22.189 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:addon-resizer  system:metrics-server-nanny  {@type: type.googleapis.c…
2024-11-04 02:24:22.193 UTC+5:45  Cloud Monitoring API  CreateDashboard  nurgalymdzhambul@gmail.com  {@type: type.googleapis.com/google.cloud.audit.Audit…
2024-11-04 02:24:22.238 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:external-resizer-pd-csi-s…  system:pdcsi-controller  {@type: type.go…
2024-11-04 02:24:22.256 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:addon-manager  system:addon-manager  {@type: type.googleapis.com/goog…
2024-11-04 02:24:22.268 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:gke-common-webhook-lock  system:gke-common-webhooks  {@type: type.goo…
2024-11-04 02:24:22.302 UTC+5:45  Kubernetes Apiservice Requests  update  kube-system:vpa-recommender  system:vpa-recommender  {@type: type.googleapis.com/…
```

ount	Error	Users	First seen	Last seen	Documenta
1	Readiness probe failed: Get "http://10.48.1.19:8080/tasks": dial tcp 10.48.1.19:8080: connect: connection refused	0	7 days ago	40 minutes ago	Troublesho guide

- **Cost Optimization Strategies**
  - Recommendations made and their impact on cost savings.

After a detailed analysis, I can suggest a couple of changes to make the deployment of our application much cheaper.
First, when starting a kubernetes cluster, 3 nodes for 3 zones are created by default. We can reduce their number to 1 by specifying only one zone to run the cluster, thereby reducing the payment by 3 times.
Secondly, the type of virtual machines on which the kubernetes cluster is running is e2-meduim.For our kubernetes cluster with 1 node, e2-small will be enough. Thus, we will again reduce the cost of payment. Also, for more detailed resource values, we could do more detailed tests through Jmeter to find out the exact indicators of the resource limit. So we could reduce the cost again.
Thirdly, we could set up automatic node scaling in the kubernetes cluster, even if our application needed more power, GCP launched another node to maintain the required amount of resources. Or vice versa, if the load was too small, it would destroy one of the nodes.
The most important thing that remains unchanged is the payment for external IP addresses for our load balancer. To reduce the cost for future use, we could implement the Ingress nginx controller.

6. Conclusion

It is very important to apply methods to ensure safety. Access control, data encryption and other methods help to prevent unauthorized access, protect confidential data and comply with all the latest security requirements. This way your applications will be in demand due to its reliability. And using monitoring tools and setting up alerts will allow you to quickly identify threats. Also, even if you get into a bad situation, following the incident response plan, you will be able to react quickly and solve all the problems.
By applying horizontal and vertical scaling strategies, you will ensure the stable operation of the application. And by making all this automatic, you will optimally distribute power and manage costs.

7. Recommendations

- It is necessary to implement data encryption practices using key management.
- If we create a Load balancer, then we need to configure HTTPS SSL certificates so that our data is encrypted.
- You need to set up autoscaling to optimize costs and performance.
- You need to set up alert and alert systems to quickly respond to incidents.
- You need to set up a recovery plan.

8. References

1) https://cloud.google.com/?hl=ru
2) https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/
3) https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/
4) https://cloud.google.com/load-balancing?hl=ru\

9. Appendices

https://github.com/Esimseitm/cloud-app-dev - github repo with all files, code.