

Examen Final

Documentation

Herbert Rafael Reyes Portillo

201612114

El primer paso es crear el cluster de kubernetes

Google Cloud Platform g6-analysis Search products and resources

Create a Kubernetes cluster ADD NODE POOL REMOVE NODE POOL

Cluster basics

The new cluster will be created with the name, version, and in the location you specify here. After the cluster is created, name and location can't be changed.

To experiment with an affordable cluster, try **My first cluster** in the **Cluster set-up guides**

My first cluster
An affordable cluster to experiment with

NAME
cluster-1

Location type
☒ Zonal
☐ Regional

Zone
us-central1-c

☐ Specify default node locations
Current default: us-central1-c

Master version
Choose a release channel for automatic management of your cluster's version and upgrade cadence. Choose a static version for more direct management of your cluster's version. [Learn more.](#)

☒ Static version

Definimos la versión que deseamos para nuestro cluster

Master version

Choose a release channel for automatic management of your cluster's version and upgrade cadence. Choose a static version for more direct management of your cluster's version. [Learn more.](#)

☒ Static version
☐ Release channel

Static version
1.16.13-gke.401 (default)

Luego en default pool, seleccionamos el tamaño y cuantas instancias vamos a utilizar

Una vez creado el cluster, nos va a aparecer así

A Kubernetes cluster is a managed group of VM instances for running containerized applications. [Learn more](#)

Filter by label or name						
<input type="checkbox"/> Name ^	Location	Cluster size	Total cores	Total memory	Notifications	Labels
<input type="checkbox"/> final-sopes1	us-central1-c	3	6 vCPUs	6.00 GB	<button>Connect</button>	

Una vez creado el cluster, por medio de power Shell ingresamos a el:

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to g6-analysis.
Use "gcloud config set project [PROJECT ID]" to change to a different project.
herbertreyes13@cloudshell:~ (g6-analysis)$ gcloud container clusters get-credentials final-sopes1 --zone us-central1-c --project g6-analysis
Fetching cluster endpoint and auth data.
```

Ahora el siguiente paso es configurar nuestra aplicación.

El primer paso es crear el Dockerfile tanto para el front end como para el back end

```

app-final > Dockerfile > ...
1  # pull official base image
2  FROM node:13.12.0-alpine
3  EXPOSE 3000
4  # set working directory
5  WORKDIR /app
6
7  # add `/app/node_modules/.bin` to $PATH
8  ENV PATH /app/node_modules/.bin:$PATH
9
10 # install app dependencies
11 COPY package.json ./
12 COPY package-lock.json ./
13 RUN npm install --silent
14 RUN npm install react-scripts@3.4.1 -g --silent
15
16 # add app
17 COPY . ./
18 # start app
19 CMD ["npm", "start"]

```

Dockerfile de front end

```

Back > Dockerfile > ...
1  FROM python
2  WORKDIR /app
3  ADD . .
4  RUN pip3 install --upgrade pip
5  RUN pip3 install Flask
6  RUN pip3 install flask_cors
7  RUN pip3 install requests
8  RUN pip3 install mysqlclient
9  RUN pip3 install mysql-connector-python
10 EXPOSE 4200
11 ENTRYPOINT ["python", "index.py"]

```

Dockerfile backend

Luego procedemos a crear nuestro Docker-compose con la siguiente información

```
docker-compose.yml
1  version: '3'
2
3  services:
4    sample:
5      image: herbertreyes13j/finalfront:5.0
6      build:
7        context: app-final/.
8        dockerfile: Dockerfile
9      ports:
10       - 3000:3000
11      environment:
12       - CHOKIDAR_USEPOLLING=true
13      stdin_open: true
14      labels:
15       kompose.service.type: LoadBalancer
16      deploy:
17       replicas: 3
18
19    servidor1:
20      image: herbertreyes13j/finalback:5.0
21      build:
22        context: Back/.
23        dockerfile: Dockerfile
24      ports:
25       - "4200:4200"
26      network_mode: host
27      labels:
28       kompose.service.type: LoadBalancer
29      deploy:
30       replicas: 3
31
```

En donde básicamente le explicamos en que puertos vamos a correr todo, cuantas replicas deseamos, si trabajara como balanceador de carga.

El siguiente paso, es volver este Docker-compose a manera que Kubernetes pueda entenderlo, por lo que el siguiente paso consiste en convertir todos estos archivos al lenguaje de Kubernetes por medio de kompose. Únicamente debemos ingresar el comando “Kompose convert” y se generaran dichos archivos:

```
herbertreyes13@temporal-cloud: ~/sopes1_final - Google Chrome
ssh.cloud.google.com/projects/g6-analysis/zones/us-central1-a/instances/temporal-cloud?useAdminProxy=true&authuser=0&hl=en_US...
herbertreyes13@temporal-cloud:~/sopes1_final$ ls
BD Back Deploy README.md app-final docker-compose.yml
herbertreyes13@temporal-cloud:~/sopes1_final$ kompose convert
INFO Kubernetes file "sample-service.yaml" created
INFO Kubernetes file "servidor1-service.yaml" created
INFO Kubernetes file "sample-deployment.yaml" created
INFO Kubernetes file "servidor1-deployment.yaml" created
herbertreyes13@temporal-cloud:~/sopes1_final$
```

Ahora procedemos a crear nuestras imágenes de Docker y subirlas a Docker-hub. Lo primero es crear las imágenes

```
herbertreyes13@temporal-cloud: ~/sopes1_final - Google Chrome
ssh.cloud.google.com/projects/g6-analysis/zones/us-central1-a/instances/temporal-cloud?useAdminProxy=true&authuser=0&hl=en_US...
herbertreyes13@temporal-cloud:~/sopes1_final$ sudo docker build --pull --rm -f "app-final/Dockerfile" -t sopefinal
front:5.0 "app-final"
Sending build context to Docker daemon 779.3kB
Step 1/10 : FROM node:13.12.0-alpine
13.12.0-alpine: Pulling from library/node
aad63a933944: Pull complete
a00bd932208e: Pull complete
c57f2c59b937: Pull complete
f3446470f297: Pull complete
Digest: sha256:cc85e728fab3827ada20a181ba280cae1f8b625f256e2c86b9094d9bfe834766
Status: Downloaded newer image for node:13.12.0-alpine
----> 483343d6c5f5
Step 2/10 : EXPOSE 3000
----> Running in 16fb8b549eb5
Removing intermediate container 16fb8b549eb5
----> 50edb6c15bad
Step 3/10 : WORKDIR /app
----> Running in 3cb8bb836e1c
Removing intermediate container 3cb8bb836e1c
----> 82fa64a73547
Step 4/10 : ENV PATH /app/node_modules/.bin:$PATH
----> Running in e5119954eed8
Removing intermediate container e5119954eed8
----> 6473e04ce552
Step 5/10 : COPY package.json ./
----> 5c16bfff4052
Step 6/10 : COPY package-lock.json ./
----> 2dc3d6f3f0d2
Step 7/10 : RUN npm install --silent
----> Running in 8e538eab5af9
```

```

herbertreyes13@temporal-cloud:~$ cd sopest1_final
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker build --pull --rm -f "Back/Dockerfile" -t sopestfinalback:
5.0 "Back"
Sending build context to Docker daemon  5.12kB
Step 1/11 : FROM python
latest: Pulling from library/python
e4c3d3e4f7b0: Extracting [----->] 22.02MB/50.4MB
101c41d0463b: Download complete
8275efcd805f: Download complete
751620502a7a: Download complete
0a5e725150a2: Downloading [----->] 165.7MB/192.3MB
397d8a5694db: Download complete
b1d09d0eaabcb: Download complete
475299e7c7f3: Download complete
ade3eeefc571: Download complete

```

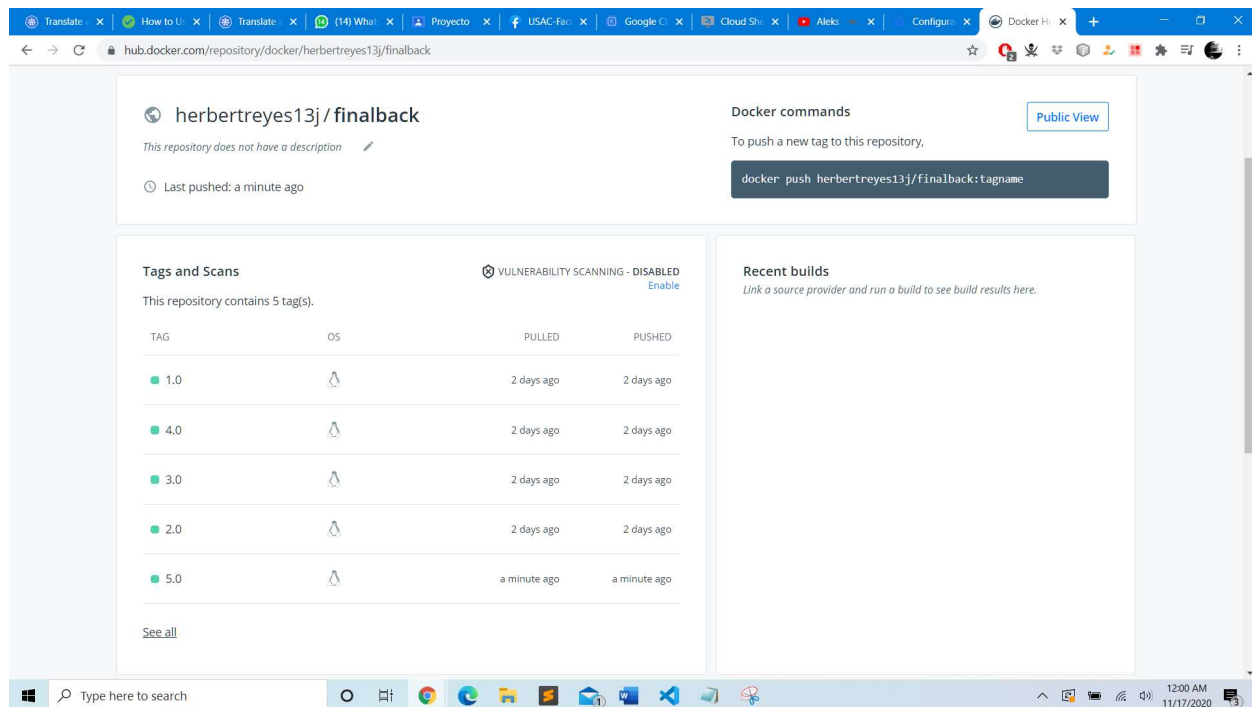
Ahora procedemos a agregarles una etiqueta a nuestras imágenes para subirlas a la nube. El primer paso es colocarle la etiqueta con la que lo subiremos a Docker hub. Una vez tenga la etiqueta, hacemos un Docker push para que se suba a Docker hub.

```

herbertreyes13@temporal-cloud:~/sopest1_final - Google Chrome
ssh.cloud.google.com/projects/g6-analisis/zones/us-central1-a/instances/temporal-cloud?useAdminProxy=true&authuser=0&hl=en_US&projectNumber=857789487621
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker images
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://v2Fvar%2Frun%2Fdocker.sock/v1.40/images/json: dial unix /var/run/docker.sock: connect: permission denied
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
sopestfinalback      5.0                8fa485854f05       49 seconds ago     905MB
<none>               <none>             2dc3d6f3f0d2       3 minutes ago      115MB
python               latest             768307c0b962       13 days ago        886MB
node                 13.12.0-alpine     483343d6c5f5       7 months ago       114MB
herbertreyes13@temporal-cloud:~/sopest1_final$ docker tag 8fa485854f05 herbertreyes13j/finalback:5.0
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://v2Fvar%2Frun%2Fdocker.sock/v1.40/images/8fa485854f05/tag?repo=herbertreyes13j%2Ffinalback&tag=5.0: dial unix /var/run/docker.sock: connect: permission denied
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker tag 8fa485854f05 herbertreyes13j/finalback:5.0
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
herbertreyes13j/finalback  5.0                8fa485854f05       About a minute ago  905MB
sopestfinalback         5.0                8fa485854f05       About a minute ago  905MB
<none>                 <none>             2dc3d6f3f0d2       4 minutes ago      115MB
python                 latest             768307c0b962       13 days ago        886MB
node                   13.12.0-alpine     483343d6c5f5       7 months ago       114MB
herbertreyes13@temporal-cloud:~/sopest1_final$ sudo docker push herbertreyes13j/finalback
The push refers to repository [docker.io/herbertreyes13j/finalback]
0c48d6fd4d9a: Pushed
79197a1777f9: Pushed
3a29afe171cc: Pushed
247d049e40f9: Pushed
201864cc179d: Pushed
dc99b07bd533: Pushed
ebc0df41118c: Pushed
8f2a3ef28f14: Pushed
641e75738194: Layer already exists
c4fc3bde6e3cb: Layer already exists
5f03d9827383: Layer already exists
6f7043721c9b: Layer already exists
a933681cf349: Layer already exists
f49420b92dc8: Layer already exists
fc342cfe6c03: Layer already exists
630e4f1da707: Layer already exists
9780f6d83e45: Layer already exists
5.0: digest: sha256:cd94fcee60d4e150a2a9a1b18e51fb8b793546953caad63fb4db425781ae09 size: 3896
herbertreyes13@temporal-cloud:~/sopest1_final$

```

Y tendremos nuestra imagen subida a dockerhub, con la versión que le hemos puesto



Finalmente procedemos a kubernetes, a hacer deploy a los archivos que generamos anteriormente

```
herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$ kubectl apply -f Deploy
deployment.apps/sample configured
service/sample configured
deployment.apps/servidor1 configured
service/servidor1 configured
herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$
```

Y finalmente podremos ver los pods

```
herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$ kubectl get deployment,svc,pods
NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/sample 3/3 1 3 2d3h
deployment.apps/servidor1 3/3 3 3 2d3h

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 10.4.0.1 <none> 443/TCP 2d5h
service/sample LoadBalancer 10.4.10.60 34.123.155.182 3000:30070/TCP 2d3h
service/servidor1 LoadBalancer 10.4.10.178 35.202.58.137 4200:30002/TCP 2d3h

NAME READY STATUS RESTARTS AGE
pod/sample-6c677d76f4-9mpkh 0/1 ContainerCreating 0 29s
pod/sample-78bddd9f-2v8mc 1/1 Running 0 47h
pod/sample-78bddd9f-7h9qx 1/1 Running 0 47h
pod/sample-78bddd9f-nw28d 1/1 Running 0 47h
pod/servidor1-7966cbc6b4-g7cpr 1/1 Running 0 29s
pod/servidor1-7966cbc6b4-mxzh 1/1 Running 0 18s
pod/servidor1-7966cbc6b4-pg2c6 1/1 Running 0 23s
pod/servidor1-d5f6959-8rgzt 1/1 Terminating 0 29s
herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$
```

Vemos como se eliminan ciertos pods y otros se han ido creando


```

herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$ kubectl get deployment,svc,pods
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/sample              3/3     3             3           2d3h
deployment.apps/servidor1           3/3     3             3           2d3h

NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/kubernetes                  ClusterIP           10.4.0.1         <none>           443/TCP          2d5h
service/sample                      LoadBalancer       10.4.10.60       34.123.155.182   3000:30070/TCP   2d3h
service/servidor1                   LoadBalancer       10.4.10.178      35.202.58.137    4200:30002/TCP   2d3h

NAME                                READY   STATUS      RESTARTS   AGE
pod/sample-6c677d76f4-968fp         1/1     Running     0          72s
pod/sample-6c677d76f4-9mpkh         1/1     Running     0          111s
pod/sample-6c677d76f4-ds7qr         1/1     Running     0          37s
pod/sample-78bddd9f-2v8mc            0/1     Terminating 0          47h
pod/servidor1-7966cbc6b4-g7cpr       1/1     Running     0          111s
pod/servidor1-7966cbc6b4-mxzh       1/1     Running     0          100s
pod/servidor1-7966cbc6b4-pg2c6      1/1     Running     0          105s
herbertreyes13@cloudshell:~/sopes1_final (g6-analisis)$

```

Y finalmente veremos nuestra aplicación creada

