

DESAFIO # 6: Mejore la implementación de la API realizando un despliegue que use contenedores (valide las distintas opciones que le brinda su nube).

Considere una prueba de consumo a la API implementando o activando algún front de acceso para ejecutar la invocación a la view/query/report.

## Desarrollo:

Para la realización del presente punto se tuvieron en cuenta 3 diferentes archivos:

- Dockerfile: archivo base para correr Docker desktop y generar el contenedor que posteriormente será llevado a la nube de GCS.
- Requirements.txt: archivo base que contiene las dependencias utilizadas por Docker y por Google SDK para el correcto funcionamiento y despliegue en nube, las dependencias son:

```
Flask  
google-cloud-bigquery  
flask-ngrok  
gunicorn
```

- index.html: contiene la realización del punto 6, mediante la creación de un código html junto con un script de javascript, que realiza el llamado fetch a la URL producto de toda la configuración Docker - SDK, para así cumplir con el despliegue del desafío.

## Procedimiento:

1. Se usó Docker desktop para habilitar el entorno de contenedores y configurar el acceso a la Shell de Google o Google SDK
2. Se crean los archivos Dockerfile y requirements.txt, necesarios para generar el contenedor
3. Mediante gitbash se configuran los comandos para correr y desplegar la configuración que nos dará el enlace al servicio, el cual es: <https://mi-api-flask-krzpjsiaia-uc.a.run.app/report>
4. Se crea el archivo index.html para crear el front y hacer la solicitud fetch mediante javascript, para así comunicar dicho front con el funcionamiento del aplicativo, y por medio de flask se pone a correr.

## Anexos (pruebas de operatividad en los diferentes softwares)

### Git bash

```
Byron@DESKTOP-FIMIK6H MINGW64 ~/desktop/Entrevista (main)
$ docker build -t mi-api-flask .
[+] Building 14.2s (11/11) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 650B
=> [internal] load metadata for docker.io/library/python:3.9-slim
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/6] FROM docker.io/library/python:3.9-slim@sha256:aa7b73608abcfb021247bbb4c111435234a0459298a6da610681097a54ca2c2a
=> [internal] load build context
=> => transferring context: 1.46kB
=> CACHED [2/6] WORKDIR /app
=> [3/6] COPY requirements.txt ./
=> [4/6] RUN pip install --no-cache-dir -r requirements.txt
=> [5/6] RUN pip install gunicorn
=> [6/6] COPY . .
=> exporting to image
=> => exporting layers
=> => writing image sha256:9a2bd31cb27b771665f0eb60f2bbb82bf6053e556a26c87db092d6f2c883a060
=> => naming to docker.io/library/mi-api-flask

What's Next?
  View a summary of image vulnerabilities and recommendations - docker scout quickview

Byron@DESKTOP-FIMIK6H MINGW64 ~/desktop/Entrevista (main)
$ docker tag mi-api-flask gcr.io/planar-abbey-417215/mi-api-flask

Byron@DESKTOP-FIMIK6H MINGW64 ~/desktop/Entrevista (main)
$ docker push gcr.io/planar-abbey-417215/mi-api-flask
Using default tag: latest
The push refers to repository [gcr.io/planar-abbey-417215/mi-api-flask]
f24ff778eca1: Pushed
b7874bf76ec4: Pushed
00287f20a4e9: Pushed
925c17220ff3: Pushed
a57a35eb69b5: Layer already exists
4a7ac3585b06: Layer already exists
6be461d39d4d: Layer already exists
d91aa0e727e2: Layer already exists
c8f253aef560: Layer already exists
a483da8ab3e9: Layer already exists
latest: digest: sha256:88e515d2f243b8bc48e2a8e40498ee3934e561e3ed3b46c6ed3af9b91395c25c size: 2414

Byron@DESKTOP-FIMIK6H MINGW64 ~/desktop/Entrevista (main)
$ gcloud run deploy mi-api-flask \
  --image gcr.io/planar-abbey-417215/mi-api-flask \
  --platform managed \
  --region us-central1 \
  --allow-unauthenticated
Deploying container to Cloud Run service [mi-api-flask] in project [planar-abbey-417215] region [us-central1]
OK Deploying... Done.
  OK Creating Revision...
  OK Routing traffic...
  OK Setting IAM Policy...
Done.
Service [mi-api-flask] revision [mi-api-flask-00002-81m] has been deployed and is serving 100 percent of traffic.
Service URL: https://mi-api-flask-krzpj5iaia-uc.a.run.app
```

## Google SDK

```
Seleccionar C:\Windows\SYSTEM32\cmd.exe

Created a default .boto configuration file at [C:\Users\Byron\.boto]. See this file and
[https://cloud.google.com/storage/docs/gsutil/commands/config] for more
information about configuring Google Cloud Storage.
Your Google Cloud SDK is configured and ready to use!

* Commands that require authentication will use johnbyronh321@gmail.com by default
* Commands will reference project `planar-abbey-417215` 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 SDK like arg files and output formatting
* Run `gcloud cheat-sheet` to see a roster of go-to `gcloud` commands.

C:\Users\Byron\AppData\Local\Google\Cloud SDK>gcloud services enable cloudbuild.googleapis.com
Operation "operations/acf.p2-881666349076-c09f01f5-0ee5-445e-968f-01762a75ddb5" finished successfully.

C:\Users\Byron\AppData\Local\Google\Cloud SDK>gcloud services enable run.googleapis.com
Operation "operations/acf.p2-881666349076-426a9bf7-69f2-49f1-8c10-dd492f9bd973" finished successfully.

C:\Users\Byron\AppData\Local\Google\Cloud SDK>
C:\Users\Byron\AppData\Local\Google\Cloud SDK>docker tag mi-api-flask gcr.io/tu-id-de-proyecto/mi-api-flask
"docker" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Users\Byron\AppData\Local\Google\Cloud SDK>gcloud services enable cloudbuild.googleapis.com
C:\Users\Byron\AppData\Local\Google\Cloud SDK>gcloud services enable run.googleapis.com
```

## Google Cloud Run

The screenshot shows the Google Cloud Run console for a service named 'mi-api-flask'. The interface includes a navigation bar with 'Cloud Run' and 'Detalles del servicio'. Below this, there's a section for the service details, including the URL 'https://mi-api-flask-krzpsiaia-uc.a.run.app/' and the region 'us-central1'. The 'Registros' (Logs) tab is selected, displaying a table of log entries. The table has columns for 'Gravedad' (Severity), 'Marca de tiempo' (Timestamp), and 'Resumen' (Summary). The logs show various events, including startup probes, errors, and successful deployments.

Gravedad	Marca de tiempo	Resumen
>	2024-03-15 02:37:12.031 EDT	Default STARTUP TCP probe succeeded after 1 attempt for container "mi-api-flask-1" on port 8080.
>	2024-03-15 02:37:12.045 EDT	[2024-03-15 06:37:12 +0000] [2] [ERROR] Worker (pid:3) exited with code 3
>	2024-03-15 02:37:12.046 EDT	[2024-03-15 06:37:12 +0000] [2] [ERROR] Shutting down: Master
>	2024-03-15 02:37:12.046 EDT	[2024-03-15 06:37:12 +0000] [2] [ERROR] Reason: Worker failed to boot.
>	2024-03-15 02:37:12.590 EDT	Container called exit(3).
>	2024-03-15 02:37:12.752 EDT	GET 500 0 0 0 ms Chrome 122 https://mi-api-flask-krzpsiaia-uc.a.run.app/favicon.ico
>	2024-03-15 02:38:46.847 EDT	GET 503 955 0 1 s Chrome 122 https://mi-api-flask-krzpsiaia-uc.a.run.app/
>	2024-03-15 02:38:46.462 EDT	[2024-03-15 06:38:46 +0000] [2] [INFO] Starting unicorn 21.2.0
>	2024-03-15 02:38:46.463 EDT	[2024-03-15 06:38:46 +0000] [2] [INFO] Listening at: http://0.0.0.0:8080 (2)
>	2024-03-15 02:38:46.463 EDT	[2024-03-15 06:38:46 +0000] [2] [INFO] Using worker: sync
>	2024-03-15 02:38:46.479 EDT	[2024-03-15 06:38:46 +0000] [3] [INFO] Booting worker with pid: 3
>	2024-03-15 02:38:46.485 EDT	[2024-03-15 06:38:46 +0000] [3] [ERROR] Exception in worker process

Conclusión final : Después de realizar todas las configuraciones previamente mencionadas, correr Index.html otorgará el resultado deseado para la resolución del desafío.