

Workshop GCP IZZI

1. Prerrequisitos (Instalar)

- 1.1. Python 3.10
- 1.2. Git local
- 1.3. Docker
- 1.4. Visual Code
- 1.5. SDK Cloud (<https://cloud.google.com/sdk/docs/install-sdk#windows>)
- 1.6. Manejador de base de datos DBeaver o Workbench.
- 1.7. Lista de permisos requeridos por usuario
 - Acceso a repositorios. Github
 - Cloud run admin
 - Invoker
 - Cloud build viewer
 - Compute engine admin.
 - ServiceAccountUser

2. Técnicas de aprendizaje (Teoria)

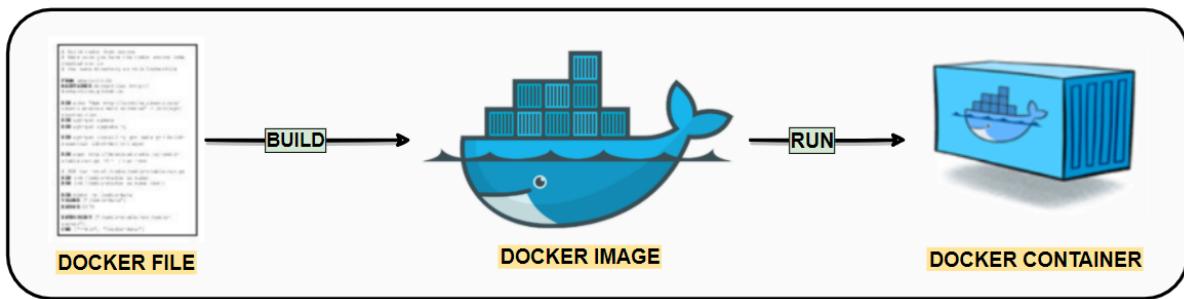
- 2.1. Aprendizaje por frecuencia espaciada.
- 2.2. Aprendizaje por recompensa.
- 2.3. Aprendizaje por asociación.
- 2.4. Efecto Dunning Krager.

3. Práctica básica (Flask, Docker y Cloud run)

3.1. Docker

Docker es una plataforma creada con el fin de desarrollar, implementar y ejecutar aplicaciones dentro de contenedores. Lo cual permite a los desarrolladores realizar el empaquetado de nuestras aplicaciones junto a sus correspondientes dependencias dentro de una unidades estandarizadas conocidas bajo el término de contenedores de software.

Aunque existen distintos tipos de implementaciones de contenedores, el más famoso motor para crear contenedores de software es Docker.



Proceso de creación de un contenedor en Docker

- **Dockerfile:** es el documento de texto sobre el que podemos agrupar una serie de comandos con el fin que se ejecuten todos a la vez evitando así tener que ejecutarlos uno a uno manualmente con el fin de que el proceso de crear una imagen de Docker sea mucho más rápido y más eficiente.
- **Docker image:** una imagen de Docker, contiene las librerías, junto al código de la aplicación que contiene todo lo necesario para ejecutar nuestra aplicación.
- **Container:** es una imagen de Docker cuando empieza a funcionar, es decir, cuando cobra vida.

■ Tabla de comandos principales en Docker

Descripción	Comando
Conocer la versión de docker	<code>docker --version</code>
Crear una imagen de Docker	<code>docker build -t nombre-app</code>
Descargar imagen existente en DockerHub	<code>docker pull httpd</code>
Listar las imágenes del sistema	<code>docker images/ docker images ls</code>
Crear un contenedor y correrlo	<code>docker run -it -d httpd</code>
Enumerar todos los contenedores	<code>docker ps (con -a incluye los finalizados)</code>

Muestra los contenedores que se ejecutan	<code>docker container ls</code>
Mostrar información de los estados de los contenedores	<code>docker container stats</code>
Mostrar logs de salida de un contenedor	<code>docker container logs</code> <code>id-container</code>
Mostrar información detallada de un contenedor	<code>docker container inspect</code> <code>id-container</code>
Pone en pausa o reanuda un contenedor que se encuentra corriendo	<code>docker container pause/unpause</code> <code>id-container</code>
Detiene contenedor	<code>docker container stop</code> <code>id-container</code>
Reinicia contenedor	<code>docker container restart</code> <code>id-container</code>
Inicia un contenedor creado	<code>docker container start</code> <code>id-container</code>
Elimina un contenedor	<code>docker container rm id-container</code>
Elimina la imagen	<code>docker rmi id-imagen</code>
Renombrar un contenedor	<code>docker container rename</code> <code>id-container</code>
Ejecutar un comando dentro del contenedor	<code>docker exec id-container comando</code>
Entrar en la shell del contenedor	<code>docker exec -it id-container</code> <code>bash</code>
Listar los procesos que se ejecutan dentro de un contenedor	<code>docker container top</code> <code>id-container</code>
Detalla toda la ed en el cluster	<code>docker network ls</code>

Información detallada sobre el sistema, kernel, imágenes etc.	<code>docker info</code>
Copiar archivos desde contenedor al sistema operativo local	<code>docker cp</code> <code>id-container:/usr/local/apache2/logs/httpd.pid</code> <code>c:/home/geekflare/</code>
Mostrar historial de imágenes	<code>docker history nombre-app</code>
Crear y listar un volumen	<code>docker volume create/ls</code>

■ Correr aplicación en docker local y prueba de comandos

- `docker build -t hellorun .`
- `docker run -e PORT=8080 -p 8080:80 hellorun`

3.2. Cloud Run

[Cloud Run](#) es una plataforma de procesamiento administrada que le permite ejecutar contenedores sin estado que se pueden invocar a través de solicitudes HTTP. Cloud Run es una plataforma sin servidores que quita la complejidad de la administración de la infraestructura, para que pueda enfocarse en lo que más importa: compilar aplicaciones extraordinarias.

Cloud Run se basa en [Knative](#), por lo que puede ejecutar sus contenedores completamente administrados con Cloud Run o en el clúster de [Google Kubernetes Engine](#) con Cloud Run en GKE.

El objetivo de esta práctica lab es que compile una imagen de aplicación sencilla, alojarla en contenedores y la implemente en Cloud Run.

- Implementa un contenedor de muestra que responda a solicitudes web entrantes con esta [Guía de inicio rápido](#).

- ¿Quieres desarrollar desde la fuente? Implementa una aplicación de muestra en Cloud Run desde la fuente con esta [guía](#).
- Ejecuta migraciones de bases de datos, informes nocturnos o transformaciones de datos por lotes con [los trabajos de Cloud Run](#).

Características

- Admite cualquier lenguaje, objeto binario o biblioteca

Usa el lenguaje de programación que deseas, las bibliotecas de cualquier lenguaje o sistema operativo, o incluso tus propios objetos binarios.

- Aprovecha los estándares y los flujos de trabajo de los contenedores

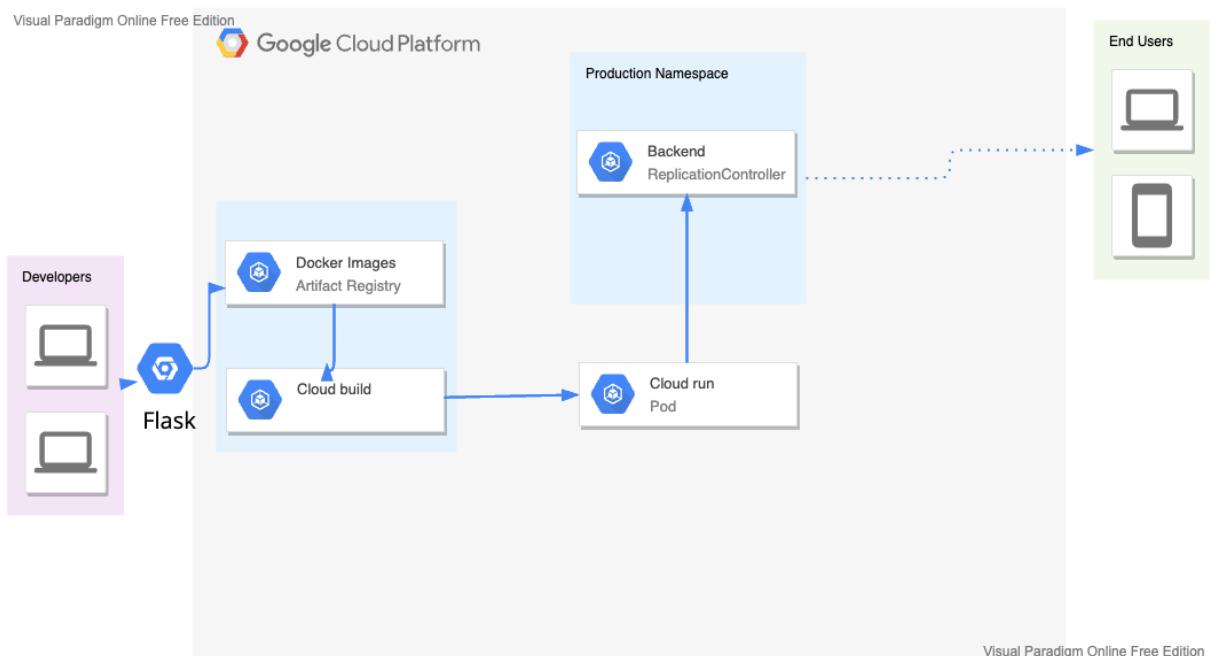
Los contenedores se convirtieron en un estándar para empaquetar e implementar el código y sus dependencias. Cloud Run funciona muy bien con el ecosistema de contenedores:

[Cloud Build](#), [Cloud Code](#), [Artifact Registry](#) y [Docker](#).

- Paga por lo que usas

Paga solo por el tiempo de ejecución de tu código, con una facturación que se redondea a la centena de milisegundos más cercana.

■ Imagen de arquitectura



3.3. Pre requisitos de configuración inicial

- Iniciar terminal en la máquina personal o iniciar en la consola de GCP.
- Verificar la cuenta en la consola

```
gcloud auth list
```

- Autenticar con la cuenta a utilizar

```
gcloud auth login
```

- Configurar región en la que vamos a trabajar

```
gcloud config set compute/region us-central1
```

- Si se desea activar la api cloud run desde consola, ejecutar siguiente comando

```
gcloud services enable run.googleapis.com
```

- Setear una variable de entorno para fijar la religión y proyecto en la que vamos a trabajar más adelante

```
LOCATION="us-central1" ó export LOCATION="us-central1"
```

```
GOOGLE_CLOUD_PROJECT="nombre-proyecto"
```

- Fijar proyecto en el que vamos a trabajar

```
gcloud config set project ${GOOGLE_CLOUD_PROJECT}
```

- Listamos los proyectos para tomar el número de proyecto.

```
$ gcloud projects list
```

- Configuramos la variable del número de proyecto (PROJECT_NUMBER)
GOOGLE_CLOUD_PROJECT_NUMBER="project-example-33"
- Confirmar la configuración
`gcloud config list`

Botón para abrir consola desde la consola

The screenshot shows the Google Cloud Platform dashboard for the project 'qwiklabs-gcp-02-885f6aec64b8'. At the top, there's a search bar and a menu icon. Below the header, there are tabs for 'DASHBOARD', 'ACTIVITY', and 'RECOMMENDATIONS'. On the right, there's a 'CUSTOMIZE' button. In the center, there's a 'Project info' section with details like Project name: 'qwiklabs-gcp-03-752778a02830', Project number: '219798113601', and Project ID: 'qwiklabs-gcp-03-752778a02830'. To the right of this is a 'Cloud Shell' terminal window titled '(qwiklabs-gcp-03-752778a02830)'. The terminal shows a session starting with 'Cloud Shell! Type "help" to get started.' and listing various environment variables and commands run.

Luego de abrir la consola, se verá de esta forma

This screenshot shows the Google Cloud Platform dashboard for the same project. The 'Cloud Shell' terminal window is now open and displays a message: 'Click here to see details about your Cloud Shell session and usage quota'. A 'Got It!' button is visible at the bottom of this message box. The rest of the dashboard interface remains the same, with the 'Project info' section and other service links on the left.

- Desplegar una aplicación python desde código fuente.

- Abrir código python xxx.py
- Ejecutar comando
`$ gcloud run deploy`
\$(Si deseas apuntar la carpeta actual) `gcloud run deploy --source .`
\$(Especificar el nombre de la imagen y de la app) `gcloud run deploy hello-nombre --image us-central1-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/cloud-run-source-deploy/hello-img`
 - Preguntará por el nombre de la app, elegir un nombre.
 - Solicitará activar apis necesarias, responder Si a todo
 - run.googleapis.com

- artifactregistry.googleapis.com
- Preguntará en qué región se va desplegar, escribir [26] us-central1
- Al final solicitará confirmar el despliegue si todos los pasos fueron exitosos. Responder sí para continuar.
- Preguntará si desea activar el despliegue sin autenticación del servicio. Si

Debería finalizar con una respuesta de la siguiente manera

```
Operation "operations/acat.p2-284769178/57-485f8237-cfe0-4fa8-954c-f352365cc76a" finished successfully.
Deploying from source requires an Artifact Registry Docker repository to store built containers. A repository named
[cloud-run-source-deploy] in region [us-central1] will be created.

Do you want to continue (Y/n)? y

This command is equivalent to running `gcloud builds submit --tag [IMAGE] /Users/jhoannatera/Desktop/PROYECTOS/GCP/work_shop_izzi/practic_01/helloworld_from_code` and
`gcloud run deploy helloworldfromcode --image [IMAGE]`

Allow unauthenticated invocations to [helloworldfromcode] (y/N)? y

Building using Dockerfile and deploying container to Cloud Run service [helloworldfromcode] in project [innate-bonfire-361004] region [us-central1]
OK Building and deploying new service... Done.
OK Creating Container Repository...
OK Uploading sources...
OK Building Container... Logs are available at [https://console.cloud.google.com/cloud-build/builds/1c396db7-0931-4a86-b125-d229c
3dfb2e97?project=284769178757].
OK Creating Revision...
OK Routing traffic...
OK Setting IAM Policy...
Done.
Service [helloworldfromcode] revision [helloworldfromcode-00001-zaw] has been deployed and is serving 100 percent of traffic.
Service URL: https://helloworldfromcode-te7xy2moeq-uc.a.run.app
```

- Al final de la respuesta se encontrará la url del servicio desplegado, a continuación abrirlo en el navegador.

3.4. Cloud Build

Compila, prueba e implementa en nuestra plataforma de CI/CD sin servidores.

- Compila software rápidamente en todos los lenguajes de programación, incluidos Java, Go, Node.js y muchos más.
- Elige entre [15 tipos de máquinas](#) y ejecuta cientos de compilaciones simultáneas por grupo.
- Se implementa en varios entornos, como en las VM, Kubernetes, Firebase o sin servidores.
- Accede a flujos de trabajo de CI/CD alojados en la nube y completamente administrados dentro de [tu red privada](#).
- Mantén tus datos en reposo dentro de una región geográfica o una ubicación específica con residencia de datos

Compilaciones extremadamente rápidas

Accede a las máquinas conectadas a través de la red global de Google con el fin de reducir de forma significativa el tiempo de compilación. Ejecuta compilaciones en VMs con alta

capacidad de CPU o almacena en caché el código fuente, las imágenes y otras dependencias para incrementar aún más la velocidad de compilación.

- Automatiza tus implementaciones

Crea canalizaciones como parte de los pasos de compilación a fin de automatizar las implementaciones. Implementa con integraciones incorporadas de [Google Kubernetes Engine](#), [App Engine](#), [Cloud Functions](#) y [Firebase](#). Usa Spinnaker con Cloud Build para crear y ejecutar canalizaciones complejas.

- Compatibilidad con múltiples nubes

Implementa en múltiples nubes como parte de tu canalización de CI/CD. Cloud Build incluye imágenes de compilador que cuentan con lenguajes y herramientas instalados. Asimismo, las tareas alojadas en contenedores de Cloud Build son completamente portátiles en diferentes nubes.

- Confirma la implementación en minutos

Pasar de la solicitud de extracción a la compilación, la prueba y la implementación es muy sencillo. Configura activadores para compilar, probar o implementar automáticamente el código fuente cuando envías cambios a GitHub, [Cloud Source Repositories](#) o un repositorio de Bitbucket.

- Privacidad inigualable

Ejecuta compilaciones en una infraestructura protegida con la seguridad de Google Cloud. Activa flujos de trabajo de CI/CD completamente administrados desde repositorios privados de código fuente alojados en redes privadas, incluido GitHub Enterprise.

- [Aloje su aplicación en contenedores y súbelala a Artifact Registry](#)
- [Compilar la imagen en gcp en la ruta central](#)

```
gcloud builds submit --tag  
gcr.io/$GOOGLE_CLOUD_PROJECT/helloworld-img
```

Listar las imágenes en container registry desde consola

```
$ gcloud container images list
```

- Configurar docker para correr imágenes que se encuentran en container registry.

```
$ gcloud auth configure-docker
```

- Para ejecutar y probar la aplicación de manera local desde Cloud Shell, inicie con el siguiente comando estándar de docker

```
$ docker run -d -p 8080:8080
```

```
gcr.io/${GOOGLE_CLOUD_PROJECT}/helloworld-img
```

The screenshot shows the Google Cloud Container Registry interface. At the top, there's a navigation bar with 'Google Cloud' and 'My First Project'. Below it, a sidebar has 'Container registry' selected, with 'Images' and 'Settings' options. The main area is titled 'My First Project' and shows a table of images. The table has columns for 'Name', 'Hostname', and 'Visibility'. One row is visible, showing 'helloworld-img' as the name, 'gcr.io' as the hostname, and 'Private' as the visibility level.

Name	Hostname	Visibility
helloworld-img	gcr.io	Private

- Desplegar desde la imagen compilada anteriormente

```
$ gcloud run deploy --image  
gcr.io/$GOOGLE_CLOUD_PROJECT/helloworld-img --allow-unauthenticated  
--region=$LOCATION
```

- Desplegar servicio desde la interfaz cloud run gcp de una imagen existente..
- Seleccionar la imagen
- Mantener parámetros por defecto y continuar.

Seleccione la imagen, coloque el nombre del servicio y al final marcar no autenticado.

Cloud Run

Create service

A service exposes a unique endpoint and automatically scales the underlying infrastructure to handle incoming requests. Service name and region cannot be changed later.

- Deploy one revision from an existing container image

Container image URL —

gcr.io/innate-bonfire-361004/helloworld-img@sha256:d2032166ff55 · SELECT

TEST WITH A SAMPLE CONTAINER

Should listen for HTTP requests on \$PORT and not rely on local state.[How to build a container](#)

- Continuously deploy new revisions from a source repository

Service name * —

helloworld-img

Region * —

us-central1 (Iowa)

[How to pick a region?](#)

CPU allocation and pricing

- CPU is only allocated during request processing

You are charged per request and only when the container instance processes a request.

- CPU is always allocated

You are charged for the entire lifecycle of the container instance.

Now view

- CPU is always allocated
You are charged for the entire lifecycle of the container instance.

Auto-scaling ?

Minimum number of instances * <input type="text" value="0"/>	Maximum number of instances <input type="text" value="100"/>
---	---

Set to one to reduce cold starts. [Learn more](#)

Ingress ?

- Allow all traffic
 Allow internal traffic and traffic from Cloud Load Balancing
 Allow internal traffic only

Authentication * ?

- Allow unauthenticated invocations
Tick this if you are creating a public API or website.
 Require authentication
Manage authorised users with Cloud IAM.

Container, connections, security

CREATE

CANCEL

- Compilar y enviar una imagen mediante Docker a GCP.
 - Compilamos la imagen con docker.
 - Vamos a la carpeta donde se encuentra el código python y ejecutamos el siguiente comando.

```
$ gcloud builds submit --tag us-central1-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/cloud-run-source-deploy/hello-docker
```
 - Si aún no configuras Docker para Google Cloud CLI para autenticar solicitudes Container Registry, hazlo con el siguiente comando.

```
$ gcloud auth configure-docker
```
 - A continuación enviamos la imagen del contenedor a Container Registry. Para ello ejecutamos el siguiente comando.

```
gcloud builds submit --tag
us-central1-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/cloud-run-source-deploy/hello-docker
```

- Ir al servicio Artifact Registry y entrar a cloud-run-source-deploy/ y confirmar que se encuentre la imagen.

The screenshot shows the Google Cloud Artifact Registry interface. The top navigation bar includes 'Google Cloud', 'My First Project', and a search bar. Below the navigation is a sidebar with 'Artifact Registry', 'Repositories' (which is selected), and 'Settings'. The main content area shows a breadcrumb path: 'us-central1-docker.pkg.dev > innate-bonfire-361004 > cloud-run-source-deploy'. A 'Filter' input field is present. A table lists the image 'hello-docker' with columns 'Name', 'Created', and 'Updated'. The 'hello-docker' row has a timestamp of '5 minutes ago' for both created and updated.

Name	Created	Updated
hello-docker	5 minutes ago	5 minutes ago

- Ir a cloud run y crear un servicio por medio de la interfaz de cloud run (click a Create service).
- Seleccione la imagen creada por docker y enviada a artifact registry y seleccione hello-docker version latest.

The screenshot shows the 'Create service' dialog for Google Cloud Cloud Run. It includes fields for 'Container image URL' and 'Service name'. Below these are sections for 'TEST WITH A SAMPLE CONTAINER' and 'Continuously deploy new revisions from a source repository'. To the right, a 'Cloud Run pricing' table shows tiers: Free tier (First 180,000 vCPU-seconds/month, First 360,000 GiB-seconds/month, 2 million requests/month) and Paid tiers (with a link to 'Check paid tiers details'). On the far right, there are 'CONTAINER REGISTRY' and 'ARTIFACT REGISTRY' tabs, with 'ARTIFACT REGISTRY' currently selected. A dropdown menu under 'ARTIFACT REGISTRY' shows a project structure: 'Project: innate-bonfire-361004' with a 'CHANGE' button, followed by 'us-central1-docker.pkg.dev/innate-bonfire-361004/cloud-run-source-deploy' and its contents: 'hello-docker' with a 'latest' tag, and revisions '8bad1cbed9' and 'nuevo2', all updated '6 minutes ago'. Buttons for 'SELECT' and 'CANCEL' are at the bottom of the dropdown.

■ Desplegar en cloud run por trabajo (jobs)

A diferencia de un servicio de Cloud Run, que escucha y entrega solicitudes, un trabajo de Cloud Run solo ejecuta sus tareas y se cierra cuando finaliza. Un trabajo no escucha ni entrega solicitudes, y no puede aceptar parámetros arbitrarios en la ejecución.

Puedes estructurar un trabajo como una sola tarea o como varias tareas independientes (hasta 10,000 tareas) que se pueden ejecutar en paralelo. Cada tarea ejecuta una instancia de contenedor y se puede configurar para que se reintente en caso de falla. Cada tarea conoce su índice, que se almacena en la variable de entorno CLOUD_RUN_TASK_INDEX. Si procesas datos en paralelo, el

código es responsable de determinar qué tarea controla qué subconjunto de los datos.

Puedes establecer tiempos de espera para las tareas y especificar la cantidad de reintentos en caso de falla de la tarea. Si alguna tarea excede la cantidad máxima de reintentos, esa tarea se marca como *con errores* y el trabajo se marca como *con errores* una vez que se ejecuten todas las tareas.

De forma predeterminada, cada tarea se ejecuta durante 10 minutos como máximo: puedes cambiar esto a un tiempo más corto o a un máximo de hasta 1 hora, si [cambias la configuración del tiempo de espera de la tarea](#). No hay un tiempo de espera explícito para la ejecución de un trabajo: después de que se completaron todas las tareas, la ejecución del trabajo está completa.

Los trabajos usan el [entorno de ejecución de segunda generación](#).

Aquí aprenderemos a crear un trabajo de Cloud Run simple, empaquetarlo en una imagen de contenedor, subir esta imagen de contenedor a Container Registry y luego implementar Cloud Run.

- Compilamos el código en una imagen y enviamos al registro.
- Usamos BuildPack para compilar el contenedor (Buildpack es un compilador de código fuente)
`$ gcloud builds submit --pack image=gcr.io/${PROJECT_ID}/logger-job-nombre`
- A continuación creamos el trabajo con el contenedor que acabas de compilar, con el siguiente comando.

```
$ gcloud beta run jobs create job-quickstart-nombre \
  --image gcr.io/${PROJECT_ID}/logger-job-nombre \
  --tasks 50 \
  --set-env-vars SLEEP_MS=10000 \
  --set-env-vars FAIL_RATE=0.5 \
  --max-retries 5 \
  --region ${LOCATION}
```

- Ahora ejecutamos el trabajo en Cloud Run.
- Para ejecutar el trabajo que acabas de crear, ejecuta el siguiente paso y al solicitar la región selecciona us-central1(26) (puedes elegir otra si gustas)
`$ gcloud beta run jobs execute job-quickstart-nombre`
- Para ver detalles de la ejecución puedes correr el siguiente comando.
`$ gcloud beta run jobs executions describe job-quickstart-66pm5`
- Puedes ejecutar el trabajo de forma local para comprobar funcionamiento.

```
docker run --rm -e FAIL_RATE=0.9 -e SLEEP_MS=1000  
gcr.io/PROJECT_ID/logger-job
```

- Compilación y desplegar con Cloud Build desde una configuración yaml.

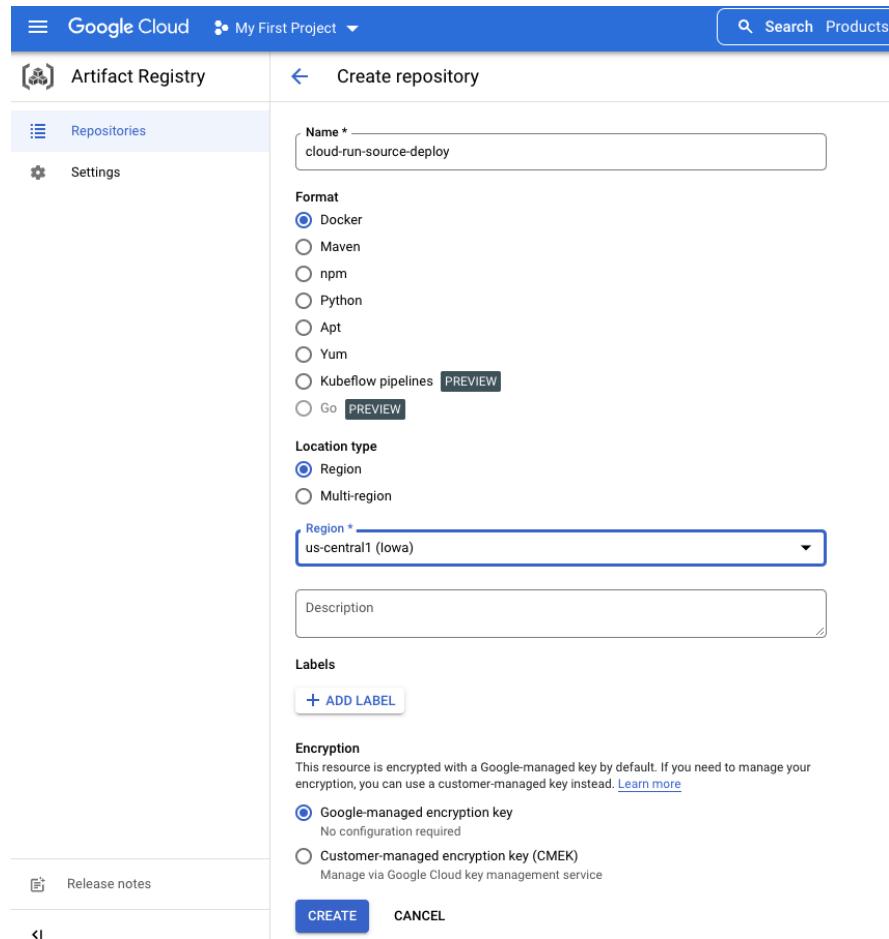
En esta sección, se explica un ejemplo de archivo de configuración de compilación para una app de Python. Tiene pasos de compilación para instalar requisitos, agregar pruebas de unidades y, luego de pasar las pruebas, compilar e implementar la app.

- Vamos a trabajar con la carpeta /practic_01/python-example-flask/
- De ser necesario aplicamos permisos necesarios a nuestra cuenta

```
# Grant the Cloud Run Admin role to the Cloud Build service account  
gcloud projects add-iam-policy-binding  
$GOOGLE_CLOUD_PROJECT \  
--member  
"serviceAccount:$GOOGLE_CLOUD_PROJECT_NUMBER@cloudbui  
ld.gserviceaccount.com" \  
--role roles/run.admin
```

```
# Grant the IAM Service Account User role to the Cloud Build service  
account on the Cloud Run runtime service account  
gcloud iam service-accounts add-iam-policy-binding \  
$GOOGLE_CLOUD_PROJECT_NUMBER-compute@developer.gser  
viceaccount.com \  
--member="serviceAccount:$GOOGLE_CLOUD_PROJECT_NUMBE  
R@cloudbuild.gserviceaccount.com" \  
--role="roles/iam.serviceAccountUser"
```

- Crear repositorio en Artifact Registry en caso que se no cree por sí solo



- Para realizar todas las tareas configuradas en el archivo de cloudbuild es necesario correr el siguiente comando.

```
$ gcloud builds submit --region=us-central1 --config cloudbuild.yaml  
--substitutions=_ARTIFACT_REGISTRY_REPO="cloud-run-source-deploy",_BUCKET_NAME="${GOOGLE_CLOUD_PROJECT}_cloudbuild",_PROJECT_ID="${GOOGLE_CLOUD_PROJECT}",_SHORT_SHA="mi-nombre".
```
- A continuación se podrá ver el servicio desplegado en cloud run y ver el log de la ejecución en cloud build.

Cloud Build

Successful: 86774d39

Started on 1 Sept 2022, 01:55:34

Steps Duration

Step	Duration
Build summary	00:02:15
5 Steps	
0: python pip install -r requirements.txt --user	00:00:27
1: python python -m pytest --junitxml=jhoan_test...	00:00:01
2: gcr.io/cloud-builders/docker build -t us-central1-docker.pkg.dev/innate...	00:00:10
3: gcr.io/cloud-builders/docker push us-central1-docker.pkg.dev/innate...	00:00:02
4: google/cloud-sdk gcloud run deploy helloworld-jhoan --image...	00:01:15

BUILD LOG

```

1 starting build "86774d39-8e12-463a-a4d5-0b03e34a17ad"
2
3 FETCHSOURCE
4 Fetching storage object: gs://innate-bonfire-361004_cloudbuild/source/1662015332.491501-4889056b2917
5 Copying gs://innate-bonfire-361004_cloudbuild/source/1662015332.491501-4889056b2917
6 / [0 files] 0.0 B/ 2.4 KiB
7 / [1 files] 2.4 KiB/ 2.4 KiB
8 Operation completed over 1 objects/2.4 KiB.
9 BUILD
10 Starting Step #0
11 Step #0: Pulling image: python
12 Step #0: Using default tag: latest
13 Step #0: latest: Pulling from library/python
14 Step #0: 1671565c8c0f: Pulling fs layer
15 Step #0: 3e94d13e55e7: Pulling fs layer
16 Step #0: f9a7c752e685: Pulling fs layer
17 Step #0: 53ad07279cd1: Pulling fs layer
18 Step #0: d6b983117533: Pulling fs layer
19 Step #0: d8892d5deded5: Pulling fs layer
20 Step #0: c71acf637d59: Pulling fs layer
21 Step #0: 864a10b3c7c84: Pulling fs layer
22 Step #0: 4334b2fe8293: Pulling fs layer
23 Step #0: 53ad07279cd1: Waiting
24 Step #0: d6b983117533: Waiting
25 Step #0: d8892d5deded5: Waiting
26 Step #0: c71acf637d59: Waiting
27 Step #0: 864a10b3c7c84: Waiting

```

Cloud Run

Services

CREATE SERVICE **MANAGE CUSTOM DOMAINS** **COPY** **DELETE**

Name	Req/sec	Region	Authentication	Ingress	Last deployed	Deployed by
helloworld-jhoan	0	us-central1	Allow unauthenticated	All	4 minutes ago	Cloud Build
helloworld-latest	0	us-central1	Allow unauthenticated	All	15 minutes ago	Cloud Build

■ Trabajo de limpieza

■ Eliminar imagen compilada

```
$ gcloud container images delete
```

```
gcr.io/$GOOGLE_CLOUD_PROJECT/helloworld-img
```

■ Para borrar el servicio de Cloud Run, use el siguiente comando

```
$ gcloud beta run services delete helloworld-img-ser
```

4. Practica 02 (Django, Sql cloud, Secrets, Cloud Build, Cloud run)

Desplegar aplicación con Cloud SQL

Objetivos de esta práctica donde se realicen las siguientes tareas

- Crear y conectar una base de datos de Cloud SQL
- Crea y usa los valores secretos de Secret Manager.
- Implementar una app de  en Cloud Run
- Alojar archivos estáticos en Cloud Storage
- Usar Cloud Build para automatizar la implementación

Componentes a utilizar:

- [Cloud SQL](#)
- [Cloud Storage](#)
- [Cloud Run](#)
- [Cloud Build](#)
- [Container Registry](#)
- [Secret Manager](#)

Requisitos:

- Habilitar facturación
- Cloud run
- Cloud SQL
- Cloud Build
- Secret Manager
- Compute Engine
- Instalar Google Cloud CLI

Permisos necesarios (en caso que no se tenga owner)

- Administrador Cloud SQL
- Administrador Cloud Run
- Administrador de Secret Manager

A. En el repo descargado, abrir la carpeta practica02/django/

```
$ cd practica02/django/ (o en visual code dar click derecho a carpeta y luego "open terminal")
```

B. Actualizar pip e instalar dependencias

```
$ pip install --upgrade pip  
$ pip install -r requirements.txt
```

C. Descarga e instala proxy de cloud sql

WIndows: https://dl.google.com/cloudsql/cloud_sql_proxy_x64.exe

Mac: curl -o cloud_sql_proxy

https://dl.google.com/cloudsql/cloud_sql_proxy.darwin.arm64

D. Crear y configurar una instancia Cloud SQL usando PostgreSQL.

```
$ gcloud sql instances create pg-test \
    --project $GOOGLE_CLOUD_PROJECT \
    --database-version POSTGRES_13 \
    --tier db-f1-micro \
    --region $LOCATION
```

Nota: En caso de tardar y aún no se crea y falla el comando anterior, correr el siguiente comando para esperar hasta que termine de crearse la instancia.

```
$ gcloud beta sql operations wait --project $GOOGLE_CLOUD_PROJECT
1442fdda-fb57-4050-aecb-936800000032
```

Nota: Este último número lo genera el comando cuando falla por tiempo de espera.

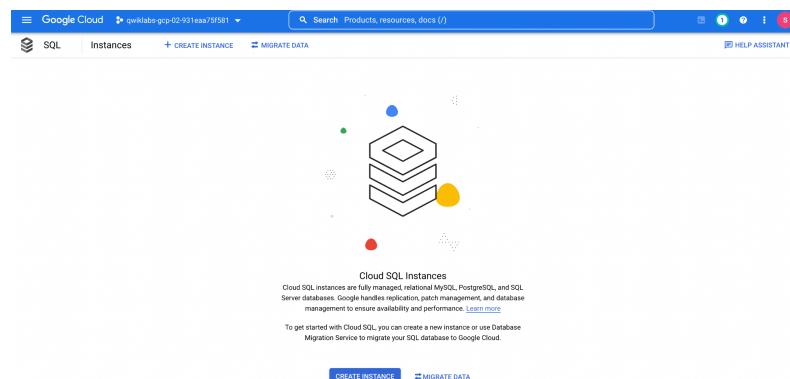
E. Cuando termine el comando anterior, creamos una base de datos postgresql

```
$ gcloud sql databases create db-nombre \
    --instance pg-test
```

F. Creamos un usuario

```
$ gcloud sql users create user-izzi --instance pg-test --password test123
```

Si desea crear la instancia de la base de datos por interfaz sin utilizar comandos, aquí se encuentran los pasos necesarios.



Google Cloud Search Products, resources, docs (/)

SQL Create an instance HELP ASSISTANT

Choose your database engine

MySQL Versions: 8.0, 5.7, 5.6 Choose MySQL

PostgreSQL Versions: 14, 13, 12, 11, 10, 9.6 Choose PostgreSQL

SQL Server Versions: 2019, 2017 Choose SQL Server

Want more context on the Cloud SQL database engines? [Learn more](#)

Google Cloud Search

Create a MySQL instance

Instance info

Instance ID * blog-db
Use lowercase letters, numbers, and hyphens. Start with a letter.

Password * GENERATE
Set a password for the root user. [Learn more](#)

No password

PASSWORD POLICY

Database version * MySQL 8.0

Choose a configuration to start with

These suggested configurations will pre-fill this form as a starting point for creating an instance. You can customize as needed later.

Production You've made changes that override the selected configuration. **RESET**
Optimized for the most critical workloads. Highly available, performant, and durable.

Development Performant but not highly available, while reducing cost by provisioning less compute and storage.

Google Cloud qwiklabs-gcp-02-931eaa75f581 Search

[Create a MySQL instance](#)

[CONFIGURATION DETAILS](#)

Choose region and zonal availability

For better performance, keep your data close to the services that need it. Region is permanent, while zone can be changed any time.

You have the location org policy in affect that is restricting your regions.

Region
us-central1 (Iowa)

Zonal availability

Single zone
In case of outage, no failover. Not recommended for production.

Multiple zones (Highly available)
Automatic failover to another zone within your selected region. Recommended for production instances. Increases cost.

[SPECIFY ZONES](#)

Customize your instance

You can also customize instance configurations later

[SHOW CONFIGURATION OPTIONS](#)

[CREATE INSTANCE](#) [CANCEL](#)

Google Cloud qwiklabs-gcp-02-931eaa75f581 Search

SQL

PRIMARY INSTANCE

- [Overview](#) (Selected)
- [Connections](#)
- [Users](#)
- [Databases](#)
- [Backups](#)
- [Replicas](#)
- [Operations](#)

All instances > blog-db

blog-db MySQL 8.0

Instance is being created. This may take a few minutes. While this operation is running, you may continue to view information about the instance.

Chart — CPU utilization

No data is available for the selected time frame.

1 hour 6 hours 1 day 7

Connect to this instance

Connection name: qwiklabs-gcp-02-931eaa75f581:us-central1:blog-db

Need help connecting?

Configuration

vCPUs	Memory
4	26 GB

Click here to see details

Creating blog-db

CLOUD SHELL Terminal (qwiklabs-gcp-02-931eaa75f581) + -

Google Cloud qwiklabs-gcp-02-931eaa75f581

Search Products, resources, docs (/)

SQL	Users								
<p>PRIMARY INSTANCE</p> <ul style="list-style-type: none">OverviewConnectionsUsersDatabasesBackupsReplicasOperations	<p>All instances > blog-db</p> <p>blog-db</p> <p>MySQL 8.0</p> <p>User accounts enable users and applications to connect to your instance. Learn more</p> <p>ADD USER ACCOUNT</p> <table border="1"><thead><tr><th>User name</th><th>Host name</th><th>Authentication</th><th>Password status</th></tr></thead><tbody><tr><td>root</td><td>% (any host)</td><td>Built-in</td><td>N/A</td></tr></tbody></table>	User name	Host name	Authentication	Password status	root	% (any host)	Built-in	N/A
User name	Host name	Authentication	Password status						
root	% (any host)	Built-in	N/A						

Search Products, resources, docs (/)

Add a user account to instance blog-db

Choose how to authenticate
You can manage access to this instance using Cloud IAM or MySQL built-in authentication. [Learn more](#)

Built-in authentication
Creates a new username and password specific to this instance. User account will have `cloudsqlsuperuser` root access, but you can customize that later as needed. [Learn more](#)

User name *

Password (Optional)

PASSWORD POLICY

Host name ?
 Allow any host (%)
 Restrict host by IP address or address range

Users created with built-in authentication have the same privileges as the `root` user. [Learn more](#)

Cloud IAM
Associates an existing IAM principal with this user account. Must have a role providing instance-level access assigned to connect.

ADD CANCEL

Google Cloud qwiklabs-gcp-02-931eaa75f581

Search Products, resources, docs (/)

SQL	Users												
<p>PRIMARY INSTANCE</p> <ul style="list-style-type: none">OverviewConnectionsUsersDatabasesBackupsReplicasOperations	<p>All instances > blog-db</p> <p>blog-db</p> <p>MySQL 8.0</p> <p>User accounts enable users and applications to connect to your instance. Learn more</p> <p>ADD USER ACCOUNT</p> <table border="1"><thead><tr><th>User name</th><th>Host name</th><th>Authentication</th><th>Password status</th></tr></thead><tbody><tr><td>blogdbuser</td><td>% (any host)</td><td>Built-in</td><td>N/A</td></tr><tr><td>root</td><td>% (any host)</td><td>Built-in</td><td>N/A</td></tr></tbody></table>	User name	Host name	Authentication	Password status	blogdbuser	% (any host)	Built-in	N/A	root	% (any host)	Built-in	N/A
User name	Host name	Authentication	Password status										
blogdbuser	% (any host)	Built-in	N/A										
root	% (any host)	Built-in	N/A										

Google Cloud qwiklabs-gcp-02-931ea75f581 Search Products, resources, docs (/)

SQL	Connections
PRIMARY INSTANCE	
Overview	All instances > blog-db
Connections	blog-db
Users	MySQL 8.0
Databases	NETWORKING SECURITY CONNECTIVITY TESTS
Backups	Choose how you want your source to connect to this instance, then define which networks are authorized to connect. Learn more
Replicas	You can use the Cloud SQL Proxy for extra security with either option. Learn more
Operations	
Release Notes	

Connections

Choose how you want your source to connect to this instance, then define which networks are authorized to connect. [Learn more](#)

You can use the Cloud SQL Proxy for extra security with either option. [Learn more](#)

Instance IP assignment

Private IP

Assigns an internal, Google-hosted VPC IP address. Requires additional APIs and permissions. Can't be disabled once enabled. [Learn more](#)

Public IP

Assigns an external, internet-accessible IP address. Requires using an authorized network or the Cloud SQL Proxy to connect to this instance. [Learn more](#)

Authorized networks

You can specify CIDR ranges to allow IP addresses in those ranges to access your instance. [Learn more](#)

You have not authorized any external networks to connect to your Cloud SQL instance. External applications can still connect to the instance through the Cloud SQL Proxy. [Learn more](#)

[ADD NETWORK](#)

App Engine authorization

All apps in this project are authorized by default. You can use [Cloud IAM](#) to authorize apps in other projects. [Learn more](#)

[SAVE](#)

[DISCARD CHANGES](#)

The screenshot shows the Google Cloud SQL Connections page. On the left, there's a sidebar with options like Overview, Connections (which is selected), Users, Databases, Backups, Replicas, and Operations. The main area is titled 'Connections' and contains a sub-section 'Authorized networks'. It says: 'Assigns an external, internet-accessible IP address. Requires using an authorized network or the Cloud SQL Proxy to connect to this instance.' Below this is a 'New network' dialog box. Inside the dialog, the 'Name' field is set to 'web front end'. Under 'Network', the value '34.69.104.120/32' is entered. There are 'CANCEL' and 'DONE' buttons at the bottom right of the dialog. Below the dialog, there's a button labeled 'ADD NETWORK'. At the bottom of the main page, there's an 'App Engine authorization' section with a note about default app authorization and a 'SAVE' button.

G. Configuramos un bucket de Cloud Storage para almacenar elementos estáticos.

```
$ gsutil mb -l $LOCATION gs://${GOOGLE_CLOUD_PROJECT}_bucket
```

Algunos datos en lugar de colocarlos directamente en el código fuente, se usa Secret Manager para almacenar esta información de forma segura.

Cloud Run y Cloud Build interactúan con los Secrets mediante las cuentas de servicio respectivas. Las cuentas de servicio se identifican con una dirección de correo electrónico que contiene el número de proyecto.

H. Creamos un archivo de entorno .env para escribir la configuración del bucket y otros datos.

```
echo
"DATABASE_URL=postgres://DB_USERNAME:PASSWORD@//cloudsql/${GOOGLE_CLOUD_PROJECT}:$LOCATION:pg-test/DATABASE_NAME" > .env

echo GS_BUCKET_NAME=${GOOGLE_CLOUD_PROJECT}_bucket >> .env
```

```
echo SECRET_KEY=$(cat /dev/urandom | LC_ALL=C tr -dc '[[:alpha:]]' | fold -w 50 | head -n1) >> .env
```

- I. Crear un secreto nuevo de django_settings, con los valores del archivo .env

```
$ gcloud secrets create django_settings --data-file .env
```

Nota: confirmamos que el secreto se guardó correctamente.

```
$ gcloud secrets describe django_settings
$ gcloud secrets versions access latest --secret django_settings
```

- J. Obtenemos el valor del número del proyecto en la variable PROJECTNUM

```
export PROJECTNUM=$(gcloud projects describe ${GOOGLE_CLOUD_PROJECT} \
--format='value(projectNumber)')
```

- K. Otorgamos permiso al secreto en la cuenta de servicio de Cloud Run

```
gcloud secrets add-iam-policy-binding django_settings --member \
serviceAccount:${PROJECTNUM}-compute@developer.gserviceaccount.com \
--role roles/secretmanager.secretAccessor
```

- L. De la misma forma agregamos permisos al servicio de la cuenta de cloud build

```
gcloud secrets add-iam-policy-binding django_settings \
--member serviceAccount:${PROJECTNUM}@cloudbuild.gserviceaccount.com \
--role roles/secretmanager.secretAccessor
```

- M. Creamos un secreto superuser_password a partir de una contraseña generada de forma aleatoria

```
echo -n "$(cat /dev/urandom | LC_ALL=C tr -dc '[[:alpha:]]' | fold -w 30 | head -n1)" | gcloud secrets create superuser_password --data-file -
```

Otorgamos acceso al secreto a Cloud Build

```
gcloud secrets add-iam-policy-binding superuser_password \
--member serviceAccount:${PROJECTNUM}@cloudbuild.gserviceaccount.com \
--role roles/secretmanager.secretAccessor
```

- N. A fin de que Cloud Build aplique las migraciones de la base de datos, debes otorgar permisos para que Cloud Build acceda a Cloud SQL.

```
gcloud projects add-iam-policy-binding ${GOOGLE_CLOUD_PROJECT} \
    --member serviceAccount:${PROJECTNUM}@cloudbuild.gserviceaccount.com \
    --role roles/cloudsql.client
```

- O. Ejecutamos el proxy de cloud sql para podernos conectar a la base de datos

```
./cloud_sql_proxy \
--instances="${GOOGLE_CLOUD_PROJECT}:us-central1:pg-test=tcp:5432"
```

ahora puedes ejecutar la app en tu computadora. Esta configuración permite el desarrollo local y la aplicación de migraciones de bases de datos. Ten en cuenta que las migraciones de bases de datos también se aplican en Cloud Build, pero necesitarás esta configuración local para makemigrations.

- P. Seteamos una variable de entorno para configurar la app para usar o no el proxy de cloud sql.

```
$ export USE_CLOUD_SQL_AUTH_PROXY=true
```

- Q. Ahora realizamos las migraciones en la base de datos

```
python manage.py makemigrations
python manage.py makemigrations polls
python manage.py migrate
python manage.py collectstatic
```

Nota: Para correr collectstatic el usuario logueado en la consola debe tener acceso a Cloud Storage Object.

- R. Ahora corremos la aplicación localmente

```
python manage.py runserver
```

- S. Ahora puede revisarlo en el navegador

<http://localhost:8000>

<http://localhost:8000/admin/>

<http://localhost:8000/test/>

- T. Vamos a compilar la imagen del backend django utilizando cloudmigrate.yaml como flujo de despliegue.

```
gcloud builds submit --config cloudmigrate.yaml \
    --substitutions _INSTANCE_NAME=pg-test,_REGION=us-central1
```

- U. Ahora desplegamos la imagen creada anteriormente

```
gcloud run deploy polls-service-jhoan \
    --platform managed \
```

```
--region ${LOCATION} \
--image gcr.io/${GOOGLE_CLOUD_PROJECT}/polls-service-jhoan \
--add-cloudsql-instances ${GOOGLE_CLOUD_PROJECT}:${LOCATION}:pg-test \
--allow-unauthenticated
```

- V. Configuramos en la consola la variable de la URL de la app desplegada en el servicio de cloud run.

Ahora que se conoce la URL del servicio, actualízala para configurar este valor como una variable de entorno

```
SERVICE_URL=$(gcloud run services describe polls-service-jhoan --platform managed \
--region $LOCATION --format "value(status.url)")
```

```
gcloud run services update polls-service-jhoan \
--platform managed \
--region $LOCATION \
--set-env-vars CLOUDRUN_SERVICE_URL=$SERVICE_URL
```

- W. Confirma que la variable DEBUG de mysite/settings.py esté configurada en False. Esto evitará que el usuario vea páginas de error detalladas
- X. Cambiar la contraseña del usuario postgres de la base de datos a una conocida.
- Y. Iniciamos una conexión en la instancia SQL

```
gcloud sql connect pg-test --user postgres
```

Creamos un usuario con menos privilegios, al crearse manualmente se crea con menos permisos.

```
CREATE USER "user-jhoan" WITH PASSWORD '123';
```

Damos privilegios al usuario pero solo con la base de datos creada

```
GRANT ALL PRIVILEGES ON DATABASE "db-izzi-jn" TO "user-jhoan";
```

Salimos de psql con \q

- Z. De forma predeterminada, este servicio se implementa con la [cuenta de servicio de procesamiento predeterminada](#). Sin embargo, en algunos casos, usar la cuenta de servicio predeterminada puede proporcionar demasiados permisos. Si deseas ser más restrictivo, debes crear tu propia cuenta de servicio y asignar sólo los permisos que requiere tu servicio. Los permisos

necesarios pueden variar de un servicio a otro, según los recursos que use un servicio en particular.

Las funciones mínimas que requiere este servicio son las siguientes:

- a. Invocador de Cloud Run
- b. Cliente de Cloud SQL
- c. Administrador de almacenamiento, en el bucket de medios
- d. Usuario con acceso a Secret Manager, en el secreto de configuración de `SECRET`. (El servicio no necesita acceder al secreto de administrador de `SECRET`).

AA. Para crear una cuenta de servicio con los permisos necesarios y asignarla al servicio, ejecute el siguiente comando

```
gcloud iam service-accounts create pollss-service-account  
  
SERVICE_ACCOUNT=polls-service-account@${GOOGLE_CLOUD_PROJECT}.iam.gserviceac  
count.com
```

```
# Cloud Run Invoker  
gcloud projects add-iam-policy-binding ${GOOGLE_CLOUD_PROJECT} \  
--member serviceAccount:${SERVICE_ACCOUNT} \  
--role roles/run.invoker
```

```
# Cloud SQL Client  
gcloud projects add-iam-policy-binding ${GOOGLE_CLOUD_PROJECT} \  
--member serviceAccount:${SERVICE_ACCOUNT} \  
--role roles/cloudsql.client
```

```
# Storage Admin, on the media bucket  
gsutil iam ch \  
serviceAccount:${SERVICE_ACCOUNT}:roles/storage.objectAdmin \  
gs://MEDIA_BUCKET
```

```
# Secret Accessor, on the Django settings secret.  
gcloud secrets add-iam-policy-binding django_settings \  
--member serviceAccount:${SERVICE_ACCOUNT} \  
--role roles/secretmanager.secretAccessor
```

BB. Asocia el servicio con la cuenta de servicio nueva para implementarla

```
gcloud run services update polls-service-jhoan \
--platform managed \
--region $LOCATION \
--service-account ${SERVICE_ACCOUNT}
```

Automatización con Cloud Build

Habilitar la API de Cloud Build

https://console.cloud.google.com/flows/enableapi?apiid=cloudbuild.googleapis.com&edirect=https://cloud.google.com/build/docs/automating-builds/create-manage-triggers&_ga=2.227724701.1825221963.1662346924-1415742975.1661315684

Debes tener la función de **editor de Cloud Build**

(roles/cloudbuild.builds.editor) en tu proyecto para crear activadores.

- a. Abrir Cloud Build en el menú principal y dar click al submenu Activadores (Trigger)

The screenshot shows the Google Cloud Platform interface for Cloud Build. The top navigation bar includes 'Google Cloud' and 'My First Project'. A search bar is at the top right. Below the navigation is a blue header bar with the 'Cloud Build' logo, 'Triggers', '+ CREATE TRIGGER', 'CONNECT REPOSITORY', and 'MANAGE REPOSITORIES' buttons. On the left, a sidebar lists 'Dashboard', 'History', 'Triggers' (which is selected and highlighted in blue), and 'Settings'. The main content area shows a table of triggers. The table has columns for Name, Description, Repository, Event, Build configuration, and Status. Two triggers are listed: 'eon-hoot' and 'trigger-front'. Both triggers point to the repository 'jnatera/eonhoot_users' and 'jnatera/front-auto-cloudbuild' respectively. The 'Event' column shows 'Push to branch' for both. The 'Build configuration' column shows 'cloudbuild.yaml' for both. The 'Status' column shows 'Enabled' for both. A filter input field 'Enter property name or value' is also visible.

Name	Description	Repository	Event	Build configuration	Status
eon-hoot	-	jnatera/eonhoot_users	Push to branch	cloudbuild.yaml	Enabled
trigger-front	-	jnatera/front-auto-cloudbuild	Push to branch	cloudbuild.yaml	Enabled

- b. Luego dar click en Connect Repository y seleccione el proveedor de código Github

The screenshot shows the 'Connect repository' dialog from the Google Cloud Platform. On the left, there's a sidebar with 'Google Cloud Platform.', 'sources, docs (/)', 'REPOSITORY', and 'MANAGE REPOSITORIES'. Below that is a table with two rows:

Event	Build configuration
Push to branch	cloudbuild.yaml
Push to branch	cloudbuild.yaml

The main area is titled 'Connect repository' with a 'Region: us-central1' dropdown. It has a step-by-step guide:

- Select source**
 - GitHub (Cloud Build GitHub App)
Build source code in response to pull requests and pushes.
 - GitHub Enterprise
Build source code hosted on premises in response to pull requests and pushes.
 - Bitbucket server
Build source code hosted on premises in response to pull requests and pushes.
 - Bitbucket data centre
Build source code hosted on premises in response to pull requests and pushes.
 - Bitbucket Cloud (mirrored) BETA
Build source code in response to pushes, mirrored through Cloud Source Repositories.
- Show more**
- Authenticate**
- Select repository**
- Create a trigger (optional)**

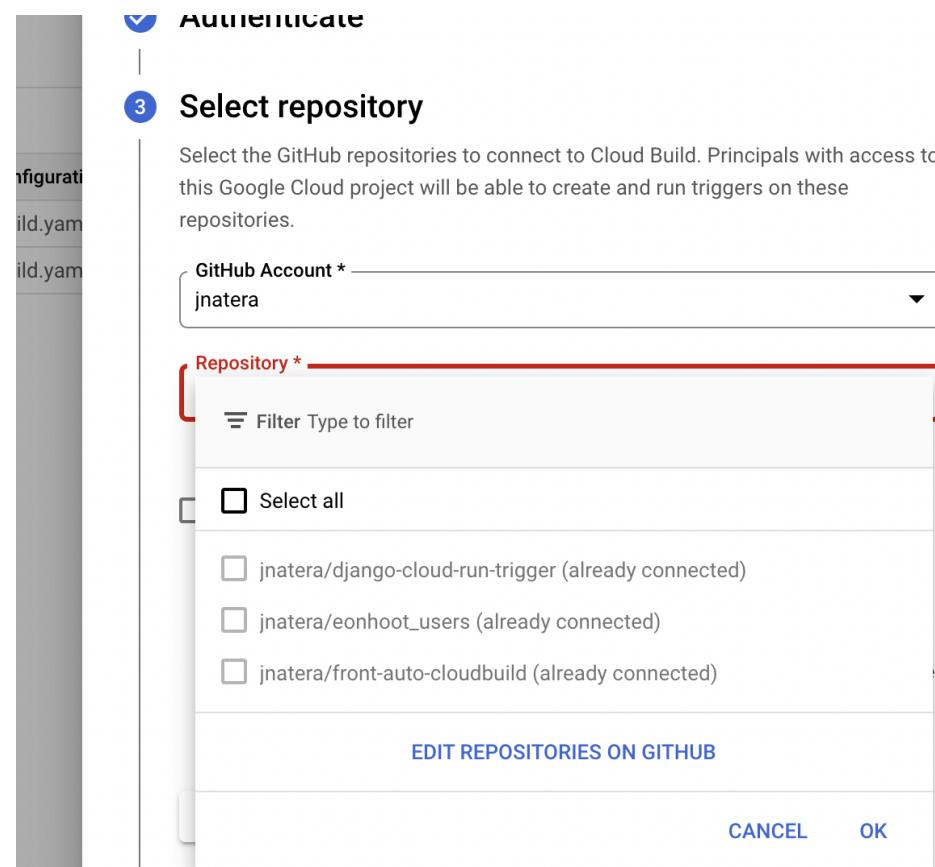
A 'CONTINUE' button is visible at the bottom of the first step.

c. Autenticarse y seleccionar el repositorio

<https://github.com/jnatera/front-auto-cloudbuild.git>

The screenshot shows the GitHub App settings page for the Google Cloud Build app. On the left, there's a sidebar with various account and repository management options like Public profile, Account, Appearance, Accessibility, Notifications, Planning and plans, Emails, Password and authentication, SSH and GPG keys, Organizations, Moderation, Planning, and automation, Repositories, Packages, and GitHub Copilot. The main content area displays the app's details: "Google Cloud Build" with a logo, installed 10 days ago, developed by GoogleCloudBuild, and a link to https://cloud.google.com/cloud-build/. It describes Google Cloud Build as letting you create fast, consistent, reliable builds across all languages, automatically building containers or non-container artifacts on commits to your GitHub repository. Below this, under "Permissions", it lists "Read access to code, issues, metadata, and pull requests" and "Read and write access to checks and commit statuses". A "Repository access" section shows that "All repositories" is selected, applying to all current and future repositories owned by the resource owner, including public repositories (read-only). A "Danger zone" button is visible at the bottom.

- d. Seleccionar el repositorio configurado y presionar Create trigger.



4 Create a trigger (optional)

- e. Dar click a Create Trigger para crear un activador hacia un despliegue configurado en un repositorio conectado anteriormente.

The screenshot shows the Google Cloud Build dashboard under the "My First Project" project. The "Triggers" tab is selected. On the right, there is a "+ CREATE TRIGGER" button. Below it, a "Region" dropdown is set to "us-central1". A "Filter" input field is present at the bottom right. The left sidebar lists "Cloud Build", "Dashboard", "History", and "Triggers".

- f. Colocar nombre del activador y marcar push branch para que se dispare el despliegue que tenemos en el repositorio indicado al final de la imagen (Source).

≡ Google Cloud • My First Project ▾

Search Products, resources, docs (/)

Cloud Build Create trigger

Dashboard History Triggers Settings

Name * Must be unique within the project's region

Region *

Description

Tags ?

Event

Repository event that invokes trigger

Push to a branch

Push new tag

Pull request
Not available for Cloud Source Repositories

Or in response to

Manual invocation

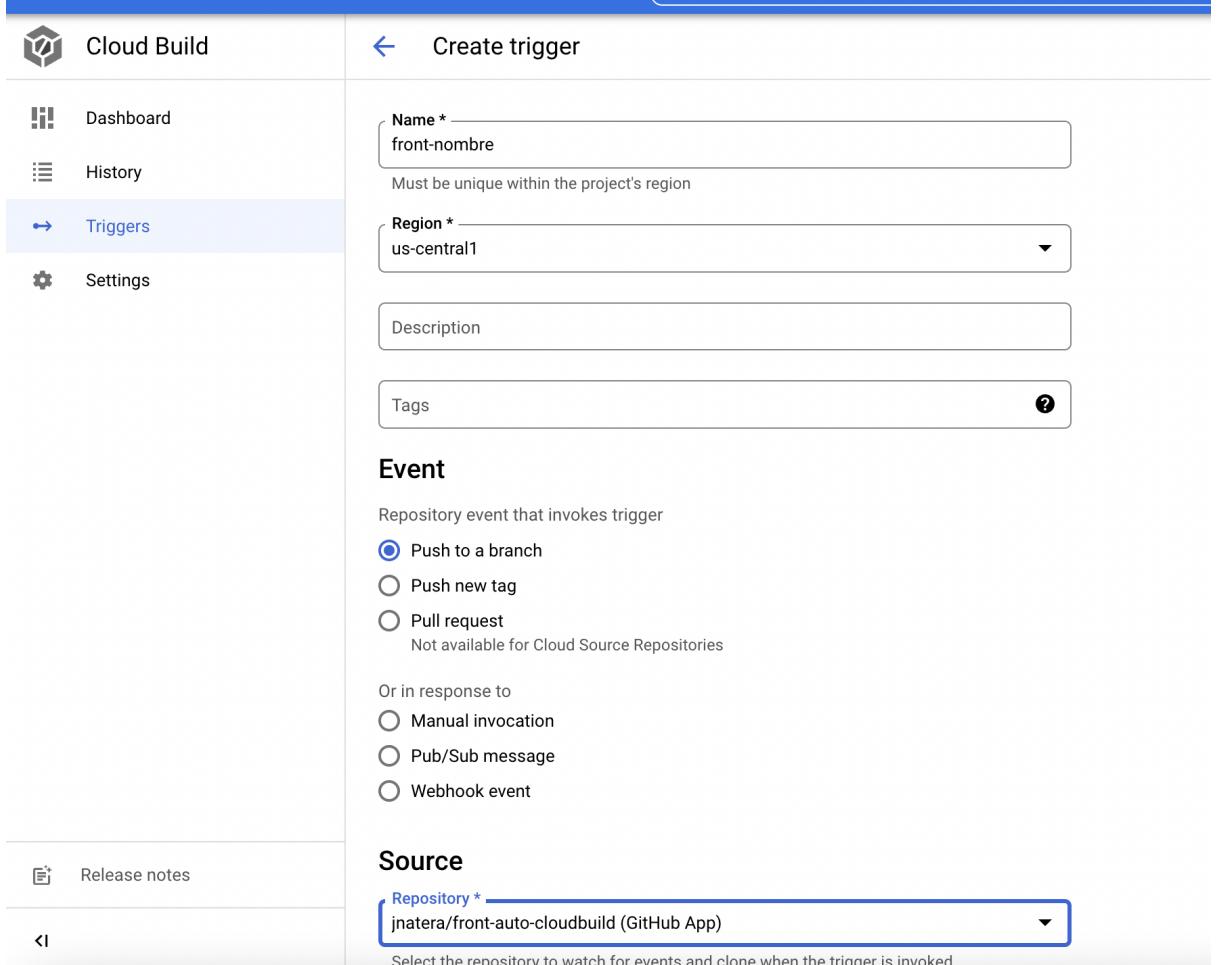
Pub/Sub message

Webhook event

Source

Repository *

Select the repository to watch for events and clone when the trigger is invoked



- g. Escribe el nombre de la rama que se usará para disparar el despliegue en Branch y Service Account Email selecciona x-compute@developer como se muestra en la imagen y luego Crear.

The screenshot shows the Google Cloud Platform interface for Cloud Build. On the left, there's a sidebar with navigation links: Cloud Build (selected), Dashboard, History, Triggers (selected), and Settings. The main content area is titled 'Create trigger' and includes the following fields:

- Branch ***: ^main\$
- Invert Regex**: Unchecked
- Matches the branch: main**
- SHOW INCLUDED AND IGNORED FILES FILTERS**: A link to expand file filter settings.
- Configuration**:
 - Type**: Autodetected (radio button selected)
 - Cloud Build configuration file (YAML or JSON)
 - Dockerfile
 - Buildpacks
- Location**:
 - Repository**: jnatera/front-auto-cloudbuild (GitHub App) (radio button selected)
 - Inline: Write inline YAML
- Advanced**:
 - Approval**: Unchecked
 - Build logs**: Unchecked
 - Service account**: Trigger a build with the following service account [Learn more](#). The email field contains 284769178757-compute@developer.gserviceaccount.com.
- Release notes**: A link at the bottom of the sidebar.

- h. Luego realiza algún cambio en el código y ejecuta comando de git push y puedes ir a History para revisar cómo va la ejecución del despliegue.

The screenshot shows the Google Cloud Build interface for a project named 'My First Project'. The main view displays a successful build summary with the ID 'd2045bf4', which started on '14 Sept 2022, 00:38:19'. The build log section shows a detailed timeline of events from scanning up to the final step. The execution details section provides a summary of the build steps.

Steps	Duration
Build summary 3 Steps	00:05:27
0: build image build -t gcr.io/innate-bonfire-361004/as...	00:04:54
1: build push push gcr.io/innate-bonfire-361004/ass...	00:00:06
2: build deploy run deploy eonhook --image=gcr.io/inha...	00:00:17

BUILD LOG

- Scanned up to 14/09/2022, 00:43:22.854 CDT Step #1 -
- 2022-09-14 00:43:22.854 CDT Step #1 -
- 2022-09-14 00:43:22.903 CDT Step #1 -
- 2022-09-14 00:43:22.903 CDT Step #1 -
- 2022-09-14 00:43:22.946 CDT Step #1 -
- 2022-09-14 00:43:23.109 CDT Step #1 -
- 2022-09-14 00:43:23.109 CDT Step #1 -
- 2022-09-14 00:43:23.835 CDT Step #1 -
- 2022-09-14 00:43:23.859 CDT Step #1 -
- 2022-09-14 00:43:24.325 CDT Step #1 -

EXECUTION DETAILS

■ Tarea para la casa.

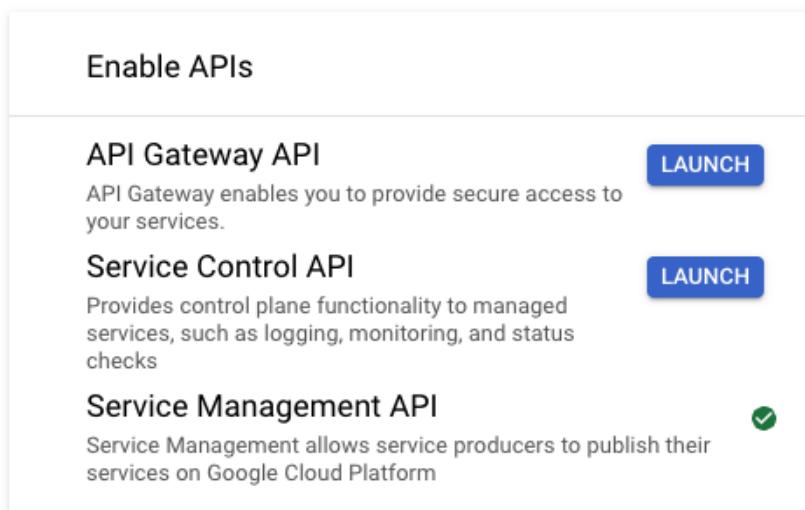
Realizar las siguientes tareas:

- Ubicar la carpeta => /practic_03/app-test en el repo de la práctica
- Compilar la aplicación en angular localmente y modificar si gusta.
- Compilar la imagen con docker localmente
- Correrlo con docker en el puerto 8080
- Luego compilar la imagen en la nube con gcloud build
- Desplegar la imagen en cloud run desde el comando deploy
- Comprobar que se visualice la aplicación web en la url del servicio cloud run.
- Intentar desplegar por configuración yaml.

5. Práctica 03 (Django, Sql cloud, Cloud Run, Secrets, api gateway)

5.1. Generar api gateway

- Habilitamos API Gateway API, Service Control API y Service Management API



- Otra forma de habilitar las apis directamente desde consola son con los siguientes comandos.

```
gcloud services enable apigateway.googleapis.com  
gcloud services enable servicemanagement.googleapis.com  
gcloud services enable servicecontrol.googleapis.com
```

- Ahora creamos una api de api gateway

```
gcloud api-gateway apis create api-zzi-nombre  
--project=$GOOGLE_CLOUD_PROJECT
```

- Vamos a comprobar detalles de la api

```
gcloud api-gateway apis describe api-zzi-nombre  
--project=$GOOGLE_CLOUD_PROJECT
```

- Crear configuración de API

Antes de que se pueda usar API Gateway para administrar el tráfico al backend de tu API implementada, se necesita una configuración de API.

Puedes crear una configuración de API con una especificación de OpenAPI que contenga anotaciones especializadas para definir el comportamiento deseado de API Gateway.

En la siguiente ubicación se encuentra el archivo de configuración para esta práctica.

```
/practic_02/openapi_yaml.yml
```

```
gcloud api-gateway api-configs create my-config-nombre \
--api=api-zzi-nombre --openapi-spec=openapi.yaml \
--project=$GOOGLE_CLOUD_PROJECT
--backend-service-account=${PROJECTNUM}-compute@developer.gserviceaccount.com
```

Esta operación puede tardar varios minutos en completarse, ya que la configuración de API se propaga a los sistemas posteriores. La creación de una configuración de API compleja podría tardar hasta diez minutos en completarse de forma correcta.

- **Crear una puerta de enlace**

Ahora implementa la configuración de la API en una puerta de enlace. La implementación de una configuración de API en una puerta de enlace define una URL externa que los clientes de la API pueden usar para acceder a tu API.

```
gcloud api-gateway gateways create gw-izzi-nombre \
--api=api-zzi-nombre --api-config=my-config-nombre \
--location=$LOCATION --project=$GOOGLE_CLOUD_PROJECT
```

- Al finalizar el proceso anterior, ejecutamos el siguiente comando para ver detalles del gateway.

```
gcloud api-gateway gateways describe gw-izzi-nombre \
--location=$LOCATION --project=$GOOGLE_CLOUD_PROJECT
```

- **Proteger acceso por medio de clave de API**

- Buscar el nombre del servicio administrado, ver la siguiente imagen

The screenshot shows the Google Cloud Platform interface for managing APIs. At the top, there's a blue header bar with the Google Cloud logo, the project name 'My First Project', and a search bar. Below the header, the main title is 'APIs'. There are three buttons: '+ CREATE GATEWAY', 'REFRESH', and a refresh icon. A table lists the managed services. The first row shows a black dot icon, the name 'api-zzi' (underlined in purple), 'Managed service', 'Created' (12/09/2022, 01:35), and 'Labels'. To the right of the table is a vertical ellipsis menu.

	Name	Managed service	Created	Labels
●	api-zzi	api-zzi-110jczhzwaku1.apigateway.innate-bonfire-361004.cloud.goog	12/09/2022, 01:35	

- Ejecutar siguiente comando tomando el valor mencionado anteriormente

```
gcloud services enable
MANAGED_SERVICE.apigateway.$GOOGLE_CLOUD_PROJECT.cloud.goog
```

- Modifica la especificación de OpenAPI que se usa para crear la configuración de tu API a fin de incluir instrucciones a fin de aplicar una política de seguridad de validación de la clave de API en todo el tráfico.

La securityDefinition configura tu API para que requiera una clave de API que se pasa como un parámetro de búsqueda llamado key cuando se solicita acceso a todas las rutas definidas en la especificación. En el archivo yml al final agregar lo siguiente.

```
securityDefinitions:  
    # This section configures basic authentication with an API key.  
    api_key:  
        type: "apiKey"  
        name: "key"  
        in: "query"
```

- Crea una configuración de API nueva con la especificación de OpenAPI modificada mediante el siguiente comando

```
gcloud api-gateway api-configs create my-config-nombre2 \  
--api=api-zzi-nombre --openapi-spec=openapi.yml \  
--project=$GOOGLE_CLOUD_PROJECT  
--backend-auth-service-account=${PROJECTNUM}-compute@developer.gserviceaccount.com
```

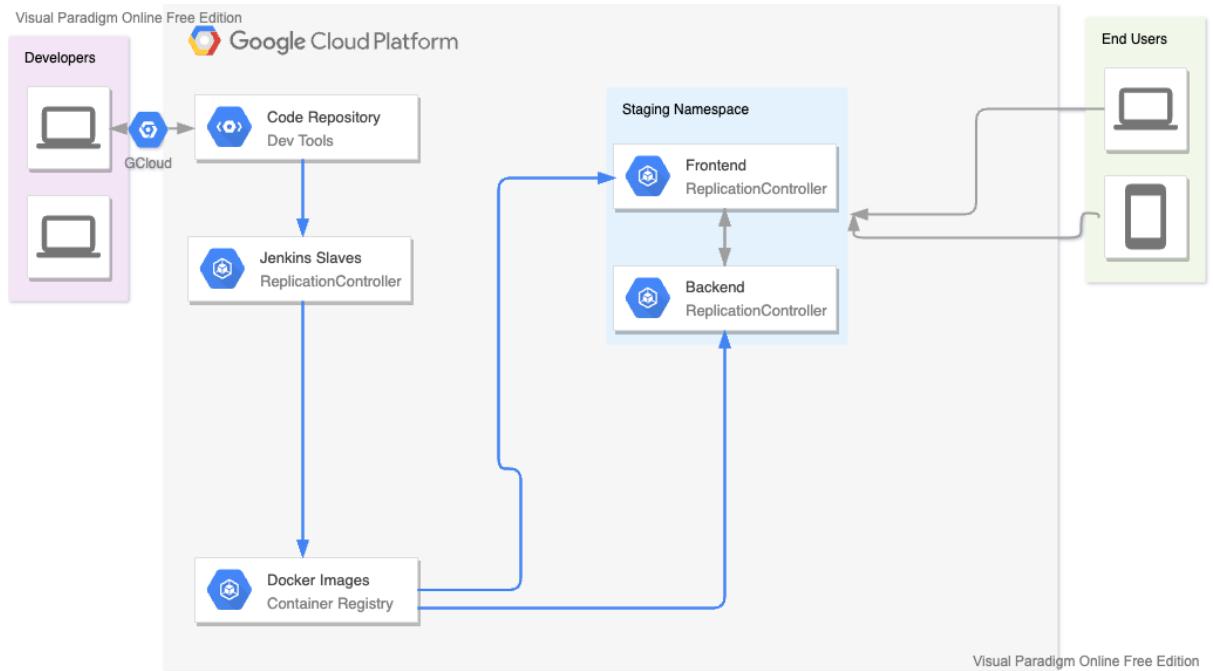
- Ejecutamos el siguiente comando para actualizar nuestra puerta de enlace existente

```
gcloud api-gateway gateways update gw-izzi-nombre \  
--api=api-zzi-nombre --api-config=my-config-key-nombre \  
--location=$LOCATION --project=$GOOGLE_CLOUD_PROJECT
```



6. Práctica 04 (Jenkins)

Arquitectura



6.1. Instalar y configurar jenkins

Screenshot of the Google Cloud Marketplace search results for "jenkins bitnami". The search bar shows "jenkins bitnami". The results page shows "jenkins bitnami" by Bitnami, with a description: "Jenkins is an open source automation server that helps you automate the building, testing, and deployment of any project across multiple platforms. Jenkins helps to avoid breaking changes so that you can save time and ensure the delivery of high-quality software. Its web-based management provides an easy way to manage and test your applications before taking them to production. This image bundles the latest versions of Jenkins and its dependencies." A file named "lapDesktop.msi" is visible in the top right corner of the browser window.

≡ Google Cloud • My First Project ▾

←

Jenkins packaged by Bitnami



Bitnami

Up-to-date, secure, and ready to run.

[LAUNCH](#) [VIEW PAST DEPLOYMENTS](#)

[OVERVIEW](#) [PRICING](#) [SUPPORT](#)

Overview

Jenkins is an open source automation server that helps you automate the building, testing, and deployment of any project across multiple platforms. Jenkins helps to avoid breaking changes so that you can save time and ensure the delivery of high-quality software. Its web interface provides an easy way to manage and test your applications before taking them to production. This image bundles the latest versions of community-recommended plugins, including pipelines and Git integration. This open source solution is packaged by Bitnami. Learn how to install, configure, and manage it at [docs.bitnami.com](#). For deployment issues, reach out our support team or join the [Bitnami Jenkins user group](#).

Additional details

Runs on: Google Compute Engine
Type: [Virtual machines](#), Single VM
Last updated: 19/09/2022
Category: [Developer tools](#)
Version: 2.346.3-2-r02
Operating System: Debian 11
Package contents: Java 11.0.15-1 . dosu 1.14.0 . Jenkins Swar

≡ Google Cloud • My First Project ▾

New Jenkins packaged by Bitnami deployment

Enable required APIs

The following APIs are required to deploy a VM product from Marketplace

Compute Engine API ⓘ	<input checked="" type="checkbox"/> Enabled
Cloud Deployment Manager V2 API ⓘ	<input checked="" type="checkbox"/> Enabled
Cloud Runtime Configuration API ⓘ	<input type="checkbox"/> Not enabled

[ENABLE](#) [SEND FEEDBACK](#)

New Jenkins packaged by Bitnami deployment

Deployment name *

Zone ▼ ?

Machine type

Machine family

GENERAL-PURPOSE COMPUTE-OPTIMISED MEMORY-OPTIMISED

Machine types for common workloads, optimised for cost and flexibility

Series ▼

Powered by Intel Skylake CPU platform or one of its predecessors

Machine type ▼



vCPU	Memory
1	3.75 GB

Boot Disk

Boot disk type * ▼ ?



Jenkins packaged b

Product provided by Bitnami

Bitnami Jenkins Usage Fee

Bitnami does not charge a usage fee.

Infrastructure fee

VM instance: 1 vCPU + 3.75 GB memory (n1)

Standard persistent disk: 10 GB

Sustained-use discount ?

Estimated monthly total

All products are priced in USD and charged Billing Account. The price for this month is 20 MXN

Price estimates based on 30-day, 24-h resources in the selected region. The calculation may not reflect all Google created or consumed by this product (consumption). Bitnami may be able to monthly GCP IaaS consumption.

New Jenkins packaged by Bitnami deployment

Firewall

Add tags and firewall rules to allow specific network traffic from the Internet



Creating certain firewall rules may expose your instance to the Internet.
Please check if the rules that you are creating are aligned with your security preferences. [Learn more](#)

Allow HTTP traffic from the Internet

Source IP ranges for HTTP traffic 

Allow HTTPS traffic from the Internet

Source IP ranges for HTTPS traffic 

I accept the [GCP Marketplace Terms of Service](#) and [Bitnami Terms of Service](#).

DEPLOY

Terms of Service

By deploying the application, you agree to the [Bitnami terms of service](#). These terms apply to the use of the software and its related documentation. You must review these terms before proceeding. By accepting these terms, you acknowledge that you have read and understood them and that you are bound by their provisions.

By using this product, you agree to the terms of service. The software is licensed to you under the Bitnami terms of service. You are not allowed to share the software with others or use it for commercial purposes without prior written consent from Bitnami.

Google is providing the software as-is. The service will be provided on an "as-is" basis. Google is not responsible for any damages or losses resulting from the use of the software.

Google Cloud My First Project

jenkins-izzi-nombre

STOP DELETE

jenkins-izzi-nombre is being deployed

VIEW DETAILS

jenkins-packaged-by-jinja

- jenkins-packaged-by-vm-tmpl vm_instance.py
- jenkins-izzi-nombre-vm vm instance
- generated-password-0 password.py
- software-status software_status.py
- jenkins-izzi-nombre-config config
- jenkins-izzi-nombre-software config waiter
- software-status-script software_status_script.py
- jenkins-izzi-nombre-tcp-80 firewall
- jenkins-izzi-nombre-tcp-443 firewall

Jenkins packaged by Bitnami
Solution provided by Bitnami

Site address	Pending
Admin user	user
Admin password	L36ctcnmHnc3 (Temporary)
Instance	jenkins-izzi-nombre-vm
Instance zone	us-central1-a
Instance machine type	n1-standard-1

MORE ABOUT THE SOFTWARE

Get started with Jenkins packaged by Bitnami

You will be able to use Jenkins packaged by Bitnami after

Documentation

- Getting Started

Support

Bitnami provides technical support for installation and set up center. Go to Bitnami support

Site address

<http://34.123.48.237/jenkins>

Admin user

user

Admin password (Temporary)

L36ctcnmHnc3

Instance

[jenkins-izzi-nombre-vm](#)

Instance zone

us-central1-a

Instance machine type

n1-standard-1



Welcome to Jenkins!

Username

Contraseña

Keep me signed in

Sign in

VM instances [CREATE INSTANCE](#) [IMPORT VM](#) [REFRESH](#) [OPERATIONS](#) [HELP ASSISTANT](#) [SHOW INFO PANEL](#) [LEARN](#)

[INSTANCES](#) [INSTANCE SCHEDULES](#)

VM instances are highly configurable virtual machines for running workloads on Google infrastructure. [Learn more](#)

<input type="checkbox"/>	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
<input checked="" type="checkbox"/>	Running	jenkins-izzi-nombre-vm	us-central1-a			10.128.0.6 (nic0)	34.123.48.237 (nic0)	SSH

Related actions [^ HIDE](#)

[Explore Actifio GO](#) [View billing report](#) [Monitor VMs](#) [Explore VM logs](#)



OPERATIONS ▾

HELP ASSISTANT

SHOW INFO PANEL

LEARN

?

≡

by	Internal IP	External IP	Connect
	10.128.0.6 (nic0)	34.123.48.237 (nic0)	SSH ▾

Monitor VMs
View outlier VMs across metrics like CPU and network

Start/Resume

Stop

Suspend

Reset

Delete

View network details

Create new machine image

View logs

View monitoring

▲ HIDE

/M

≡ Google Cloud • My First Project ▾

Search Products, resources, docs (/)

Compute Engine

Focal machines

- VM instances
- Instance templates
- Sole-tenant nodes
- Machine images
- TPUs
- Committed-use discounts
- Migrate to Virtual Machin...

Storage

- Disks
- Snapshots
- Images

Instance groups

- Marketplace
- Release notes

Edit jenkins-izzi-nombre-vm instance

Block project-wide SSH keys

When ticked, project-wide SSH keys cannot access this instance. [Learn more](#)

+ ADD ITEM

Identity and API access ?

Service accounts ?

Service account

Compute Engine default service account

Requires the Service Account User role (roles/iam.serviceAccountUser) to be set for users who want to access VMs with this service account. [Learn more](#)

Access scopes ?

Allow default access

Allow full access to all Cloud APIs

Set access for each API

Management

Availability policies

VM provisioning model

Standard

Choose 'Spot' to get a discounted, pre-emptible VM. Otherwise, stick to 'Standard'. [Learn more](#)

Instance stopped

X

SAVE CANCEL



Jenkins

Panel de Control >

+ Nueva Tarea

Personas

Historial de trabajos

Administrador Jenkins

Mis vistas

New View

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

1 Inactivo

2 Inactivo

Administración

Parece que la co

Building on the b

Jenkins URL is e
as BUILD_URL.

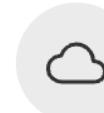
Please provide a

System Config



Configur

Configur



Adminis

Añadir, b

tures like email notifications, PR status update, and environment variables such

[Dismiss](#)

Tool Configuration

Tools, their locations and
installers.



Administrar Plugins

Añadir, borrar, desactivar y activar plugins
que extienden la funcionalidad de Jenkins.

The screenshot shows the Jenkins Plugin Manager interface. At the top, there's a navigation bar with the Jenkins logo, a search bar containing 'búsqueda', and several status icons (a question mark, a bell with '2', a shield with '1', and a refresh symbol). Below the navigation bar, the page title is 'Gestor de plugins'. Underneath, there are links to 'Volver al Panel de control', 'Administrar Jenkins', and 'Centro de actualizaciones'. A search bar contains the text 'gcloud'. The main content area is titled 'Plugin Manager' and shows a search result for 'GCloud SDK 0.0.3'. The result includes a checkbox, two descriptive tabs ('Plugins que añaden tareas relacionadas con la ejecución' and 'Plugins relacionados con la forma de ejecutar trabajos'), and a timestamp 'Hace 2 Años'. A note below states 'GCloud SDK Plugin allows the invocation of the gcloud CLI as a job step.' At the bottom, there are three buttons: 'Install without restart', 'Download now and install after restart', and 'Comprobar ahora'. A note indicates 'Update information obtained: 25 Min ago'.



Jenkins

Panel de Control > Update Center

↑ Volver al Panel de control

Administrar Jenkins

Administrar 'plugins'

on.



Global Tool Configuration

Configure tools, their locations and automatic installers.

 Jenkins

Panel de Control > Global Tool Configuration

[↑ Back to Dashboard](#)

[Manage Jenkins](#)

Global Tool Configuration

Maven Configuration

Default settings provider

Use default maven settings

Default global settings provider

Use default maven global settings

JDK

instalaciones de JDK

Listado de instalaciones de JDK en este sistema

[Añadir JDK](#)

Maven

[Save](#) [Apply](#)

 Jenkins

Panel de Control >

+ Nueva Tarea

00 Personas

Todo >

Enter an item name

» Required field

 **Crear un proyecto de estilo libre**
Esta es la característica principal de Jenkins, la de ejecutar el proyecto combinando cualquier tipo de repositorio de software (SCM) con cualquier modo de construcción o ejecución (make, ant, mvn, rake, script ...). Por tanto se podrá tanto compilar y empaquetar software, como ejecutar cualquier proceso que requiera monitorización.

OK

Products, r...

Project

resources. Learn

per gserviceaccount.com

account.com

Principal	Project
284769178757-compute@developer.gserviceaccount.com	My First Project

Role **Editor** **Condition** **+ ADD CONDITION** 

View, create, update and delete most Google Cloud resources. See the list of included permissions.

Role **Compute Instance Admin (v1)** **Condition** **+ ADD CONDITION** 

Full control of Compute Engine instances, instance groups, disks, snapshots and images. Read access to all Compute Engine networking resources.

Role **Compute Network Admin** **Condition** **+ ADD CONDITION** 

Full control of Compute Engine networking resources.

Role **Service Account User** **Condition** **+ ADD CONDITION** 

Run operations as the service account.

+ ADD ANOTHER ROLE

SAVE **TEST CHANGES**  **CANCEL**

Panel de Control > pipeline_slopez >

General Branch Sources Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance

Health metrics Properties Pipeline Libraries

Disable

(No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

Branch Sources

GitHub

Credentials ? 

- none -  **Repository HTTPS URL** ? 

Credentials are recommended

Repository HTTPS URL ? 

https://github.com/hansel/cloudrun-jenkins/blob/main/Jenkinsfile



Repository Scan - Deprecated Visualization

Behaviours

[Plain text] **Visualizar** 

Disable

(No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

Branch Sources

Add source ▲

Git

GitHub  Single repository & branch

Subversion

http://jenkins/pipeline_slopez/configure#

Branch Sources

GitHub

Credentials ? 

- none -  Add 

Credentials are recommended

Repository HTTPS URL  **Repository HTTPS URL** ?

Repository Scan - Deprecated Visualization

Behaviours



Panel de Control → pipeline_slopez → main →

 Up
 Status
 Changes
 Construir ahora 
 Ver Configuración
 Full Stage View
 GitHub
 Pipeline Syntax

Branch main
Full project name: pipeline_slopez/main

 Recent Changes

Stage View

Declarative: Checkout SCM	Run gcloud	cloud run deploy
Average stage times: 2s	4s	14s
#2 May 20 10:19 No Changes	305ms	2s
#1 May 20 10:14 No Changes	4s	6s
		2s failed

 Q. Filter builds...
 #2 20 may, 2022 15:19 
 #1 20 may, 2022 15:14
 Atom feed Para Todos