Informe Experto IT + DEVOPS

Creación de Aplicación

La aplicación se creo siguiendo TDD con Python y Flask
El código tiene el siguiente esquema

~~~
 |-app.py
 |-test_app.py
 |-rest
    |- __init__.py
 |-instance
    |- __init__.py
    |- config.py
 |- .env
~~~

- Para instalar los requermimientos se usa:
  pip install -r requirements.txt
- Para ejecutar las pruebas unitarias se utiliza el comando:
  pyton test_app.py
- Para ejecutar el aplicativo el comando:
  python app.py

Se puede usar flask o gunicorn para demonizar el microservicio.

flask run (flask necesita de la variable de entorno FLASK_APP)
gunicorn -b 0.0.0.0:5000 app:app

Nota: Al ser una aplicación muy pequeña no se obtuvo un reporte de covertura.

La aplicación y todo el contenido esta versionado en:

https://github.com/Wolfant/DShello.git

Dockerizar la Aplicación

La aplicación fue almacenada en un contenedor, para ellos se utilizo el archivo
Dockerfile. el cual se encuentra en la raiz del proyecto.
El archivo se lo versiono en: https://github.com/Wolfant/Devsu-app
Con lo cual se subio al registry publico de docker hub.
se lo puede obtener con el comando:

docker pull wolfant/devsu-app

Adicional se genero un contenedor del proyecto el cual se lo puede obtener:

docker pull wolfant/dshello

Bono

Las imágenes del balanceador y del aplicativo se subieron a un registry privado en Google Cloud

Los pasos para subir fueron:

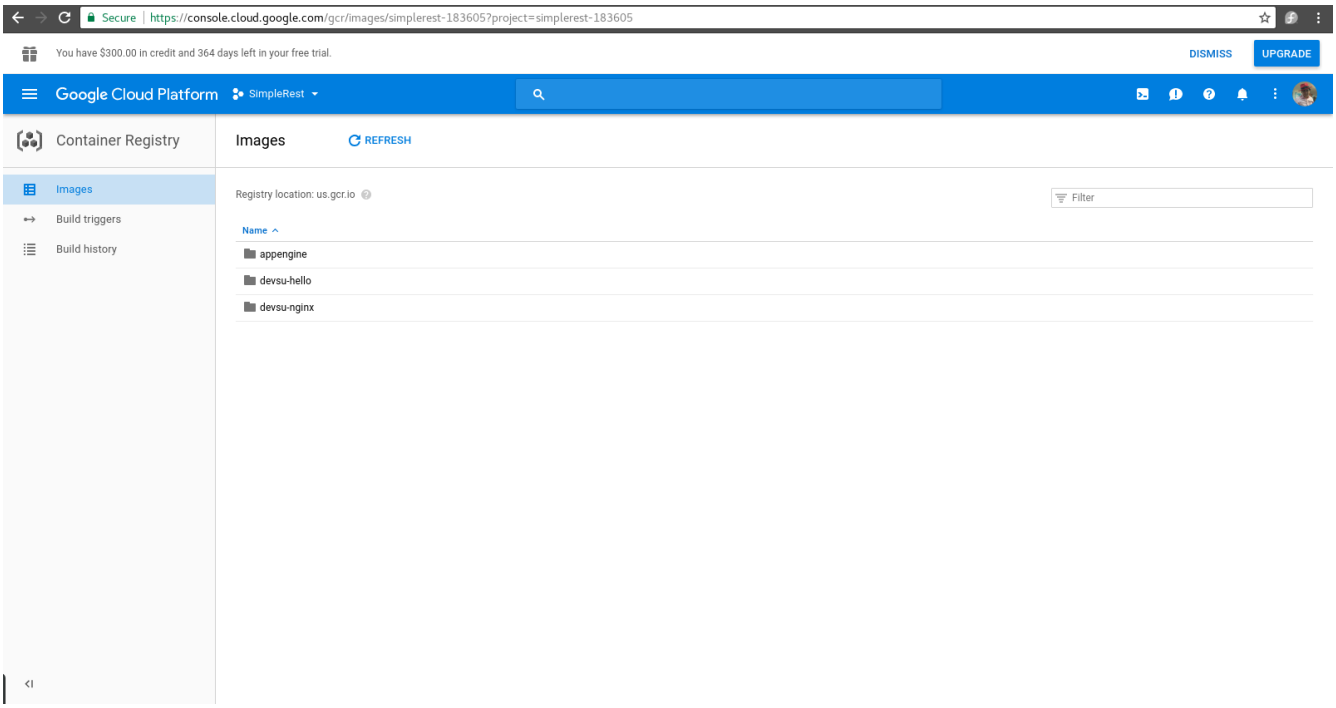
- En el la consola de google cloud:
  Se activa Tools -> Container Registry

- Se generan las imagenes:
  sudo docker build -t "wolfant/devsu-nginx" .
  sudo docker build -t "wolfant/devsu-app" .

- Se crean los tags para las imagenes
  sudo docker tag wolfant/devsu-app 'us.gcr.io/simplerest-183605/devsu-hello'
  sudo docker tag wolfant/devsu-nginx 'us.gcr.io/simplerest-183605/devsu-nginx'

- Se sube a google cloud las imagenes

  sudo gcloud auth login
  sudo gcloud docker -- push us.gcr.io/simplerest-183605/devsu-nginx
  sudo gcloud docker -- push us.gcr.io/simplerest-183605/devsu-hello



Balanceo de Aplicación

Se Creo un contenedor con nginx como balanceador de carga.
El archivo de configuración de nginx se encuentra en Docker/nginx/nginx.conf
adicional en esta carpeta esta la creación del contenedor en Docker/nginx/Dockerfile

Para la composición de toda la solución se crea un docker-compose.yml el cual está en:

Docker/docker-comose.yml

- Para compilar la composición se ejecuta el comando:

sudo docker-compose build

- Para subir la composición se ejecuta el comando:

  sudo docker-compose up

  El cual subira toda la arquitectura, como pruebas se puede revisar en local
  http://localhost:8080/hello/pepe

Pipiline de Integración continua

Para la integración continua se uso Travis-ci.org, ya que permite realizar Pipeline as code, lo cual permite versionar el pipeline. el pipeline se encuentra en el archivo .travis.yml
El Pipeline es basico se ejecuta al hacer cambios en el brach master.
los pasos del pipeline son:

1. Genera un contenedor
2. Instala los requerimientos para corre el app
3. Ejecuta las pruebas unitarias
4. Despliega la aplicacion en heroku

https://devsu-tst.herokuapp.com/hello/{name}

Bono: Se despliega la aplicación en Google Cloud APP Engine

Se debe crear el archivo de descripción para google cloud, app.yaml
Se encuentra versionado en el repositorio en la carpeta principal.
Para hacer el despliegue automático en Google Cloud:

Con un Githook en el branch master se debe ejecutar el comando,

gcloud app deploy app.yaml --project simplerest-183605

Este comando se lo debe instalar en un contenedor o en computador con Goole Cloud SDK
Google cloud permite tener varios entornos como testing, stagin y prod.

https://simplerest-183605.appspot.com

[Test_app]

https://8080-dot-3171646-dot-devshell.appspot.com/hello/pepe

[Prod]
https://simplerest-183605.appspot.com/hello/example