Despliegue de un proyecto de Django en Google Compute Engine

Google Compute Engine

El producto IaaS de Google Cloud Platform, permite gestionar máquinas virtuales donde tú defines el software que se instala, del sistema operativo en adelante. Da una máxima flexibilidad y permite gestionar la alta disponibilidad con elementos como redes privadas.

Las capacidades de Google Compute Engine incluyen:

- ❏ **Cómputo:** Inicien máquinas virtuales Linux bajo demanda. VMs con 1, 2, 4 y 8 núcleos virtuales están disponibles con 3.75 GB en RAM por núcleo virtual.
- ❏ **Almacenamiento:** Almacena datos en disco local en nuestro nuevo dispositivo de bloque persistente, o en nuestro almacén de objetos a escala de Internet, Google Cloud Storage.
- ❏ **Red:** Conecten sus máquinas virtuales entre sí con nuestra tecnología de red de alto rendimiento para formar poderosos grupos de cómputo y administrar la conectividad a Internet con firewalls configurables.
- ❏ **Herramientas:** Configuran y controlan sus máquinas virtuales a través de una herramienta de línea de comandos con scripts o una interfaz de usuario web. O creen su propio sistema de gestión dinámica utilizando nuestra API.

Django es un framework de desarrollo de aplicaciones web eficiente, versátil y dinámica en evolución. Cuando Django inicialmente ganó popularidad, la configuración recomendada para la ejecución de aplicaciones de Django se basa en Apache con mod_wsgi. El arte de ejecutar Django avanzada y en estos días la configuración recomendada es más eficiente y flexible, pero también más compleja e incluye herramientas tales como:

- virtualenv
- Gunicorn
- Supervisord
- Nginx

Nginx es un servidor web/proxy inverso ligero de alto rendimiento y un proxy para protocolos de correo electrónico (IMAP/POP3). Es software libre y de código abierto, licenciado bajo la Licencia BSD simplificada. Es multiplataforma, por lo que corre en sistemas tipo Unix (GNU/Linux, BSD, Solaris, Mac OS X, etc.) y Windows. El sistema es usado por una larga lista de sitios web conocidos, como: WordPress, Hulu, GitHub, Ohloh, SourceForge, TorrentReactor y partes de Facebook (como el servidor de descarga de archivos zip pesados).

Gunicorn 'Green Unicorn' es un servidor HTTP de Python WSGI para UNIX. Es un trabajador modelo pre-tenedor. El servidor Gunicorn es ampliamente compatible con varios frameworks web, simplemente implementado, bajo consumo de recursos del servidor, y bastante rápida.

Virtualenv o entorno virtual de Python es un ambiente creado con el objetivo de aislar recursos como librerías y entorno de ejecución, del sistema principal o de otros entornos virtuales. Lo anterior significa que en el mismo sistema, máquina o computadora, es posible tener instaladas múltiples versiones de una misma librería sin crear ningún tipo de conflicto.

Supervisor es un sistema cliente / servidor que permite a sus usuarios a monitorear y controlar una serie de procesos en sistemas operativos tipo UNIX.

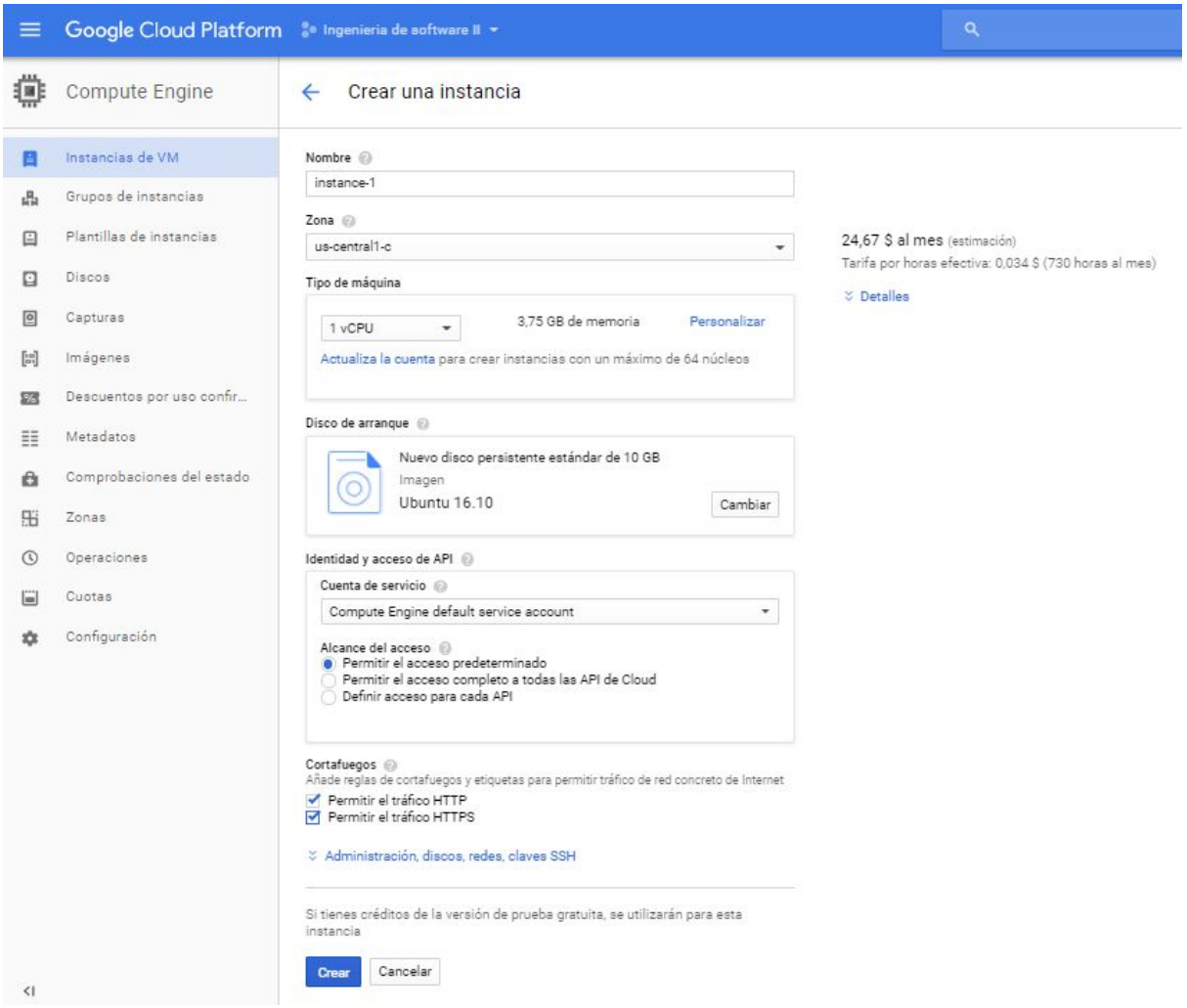
Prerrequisitos

Para poner en producción una aplicación web desarrollada con el framework django (Lenguaje de programación python), en google compute engine necesitamos tener en cuenta para lo cual se requiere hacer lo siguiente:

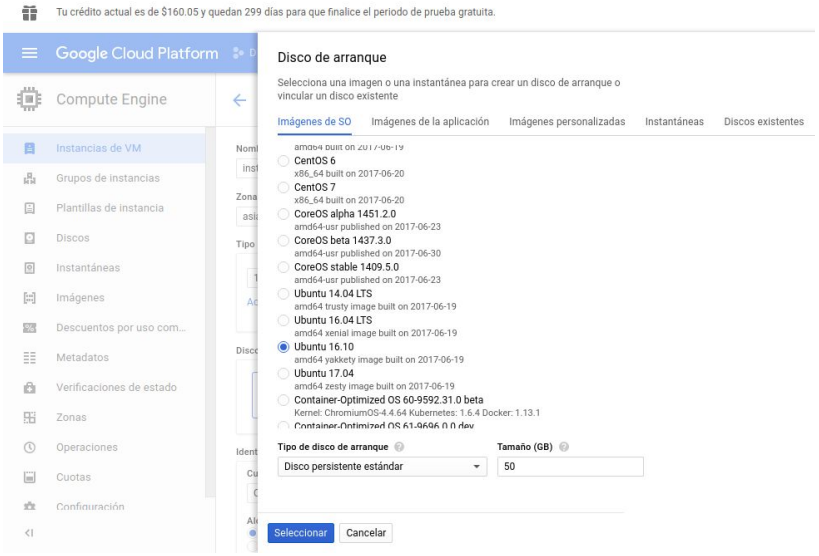
1. Tener activa una cuenta en para los servicios de google(Google Cloud Platform).
2. Tener un proyecto(Ejem:Ingeniería-de-software)

Pasos a seguir en para crear nuestro entorno:

1. Compute Engine>Instancias VM



2. Crear Instancia
    a. Creamos una máquina de acuerdo a nuestras requerimientos.



Tener en cuenta las reglas de Firewall

3. Abrimos conexion por ssh
4. Hacemos una configuracion basica:
    a. Para poder conectarnos por ssh

```
sudo apt-get update
sudo apt-get upgrade
sudo adduser yuselenin #creando usuario
sudo gpasswd -a yuselenin sudo #agregamos al grupos de administradores
```

Al final del archivo añadimos:

```
/etc/ssh/sshd_config
```

```
...
Match User yuselenin
PasswordAuthentication yes
```

Reiniciamos para aplicar los últimos cambios.

```
sudo service ssh restart
```

    b. Configuración extra(mejoramos nuestra terminal)

```
sudo apt-get install zsh
sh -c "$(wget
https://raw.githubusercontent.com/robbyrussell/oh-my-zsh/master/tools/install.sh
-O -)"
```

Configuramos nuestro entorno de despliegue:

1. Creamos usuario con privilegios limitados

Por seguridad las aplicaciones web deben ejecutarse como usuarios del sistema con privilegios limitados. Creamos un usuario para cada aplicación aplicación, asignado a un grupo denominado webapps.

```
sudo groupadd --system webapps
```

Por cada aplicación:

```
sudo useradd --system --gid webapps --shell /bin/bash --home /var/www/clivet
clivet_user
```

2. Virtualenv

Virtualenv es una herramienta que le permite crear entornos Python independientes en un sistema. Esto le permite ejecutar aplicaciones con diferentes conjuntos de requisitos simultáneamente (por ejemplo, uno basado en Django 1.5, otro basado en 1.6)

Instalamos

```
sudo apt install python-virtualenv
```

a. Configuración inicial

Creamos y activar un entorno para nuestra aplicación

```
sudo mkdir -p /var/www/clivet/
sudo chown clivet_user /var/www/clivet/
sudo su - clivet_user
cd /var/www/clivet/
virtualenv .
source bin/activate
```

Dentro de nuestro entorno configuramos nuestra aplicación

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

Permitir a otros usuarios escribir acceso al directorio de la aplicación

```
sudo chown -R clivet_user:webapps /var/www/clivet
sudo chmod -R g+w /var/www/clivet
```

3. Gunicorn

En la producción no usaremos el servidor de desarrollo único de Django, sino un servidor de aplicaciones dedicado llamado gunicorn.
Ejecutamos dentro de nuestro entorno virtual

```
pip install gunicorn
```

Para que se más usable guardamos en el archivo

```
/var/www/clivet/bin/gunicorn_start
```

```
#!/bin/bash
NAME="clivet"                                   # Nombre de la aplicacion
DJANGODIR=/var/www/clivet/clivet             # Directorio del proyecto
SOCKFILE=/var/www/clivet/run/gunicorn.sock  # comunicacion via socket
USER=clivet_user                                          # usuario
GROUP=webapps                                      # grupo
NUM_WORKERS=3                                      # Cuantos procesos
DJANGO_SETTINGS_MODULE=clivet.config.local             # archivo de conf del
proy.
DJANGO_WSGI_MODULE=config.wsgi                      # archivo de WSGI del proy.

echo "Iniciando $NAME como `whoami`"

# Activate the virtual environment
cd $DJANGODIR
source ../bin/activate
export DJANGO_SETTINGS_MODULE=$DJANGO_SETTINGS_MODULE
export PYTHONPATH=$DJANGODIR:$PYTHONPATH

# Create the run directory if it doesn't exist
RUNDIR=$(dirname $SOCKFILE)
test -d $RUNDIR || mkdir -p $RUNDIR

# Start your Django Unicorn
# Programs meant to be run under supervisor should not daemonize themselves (do
not use --daemon)
exec ../bin/gunicorn ${DJANGO_WSGI_MODULE}:application \
  --name $NAME \
  --workers $NUM_WORKERS \
  --user=$USER --group=$GROUP \
  --bind=unix:$SOCKFILE \
  --log-level=debug \
  --log-file=-
```

Indicamos que el archivo gunicorn_start sea ejecutable.

```
sudo chmod u+x bin/gunicorn_start
```

Como regla general, establezca a los trabajadores (NUM_WORKERS) de acuerdo con la siguiente fórmula: 2 * CPUs + 1. La idea es que en cualquier momento la mitad de sus workers estarán ocupados haciendo I/O. Para una sola máquina de la CPU le daría 3.

4. Supervisor

Usaremos esto para que se inicie con el sistema y que pueda reiniciarse automáticamente si existe alguna razón inesperada.

```
sudo apt install supervisor
```

Supervisor busca a los archivos creados en la carpeta para ejecutarlos:

```
/etc/supervisor/conf.d
```

Necesitamos poner nuestro script dentro de esa carpeta

```
/etc/supervisor/conf.d/clivet.conf
```

```
[program:clivet]
command = /var/www/clivet/bin/gunicorn_start
user = clivet_user
stdout_logfile = /var/www/clivet/logs/gunicorn_supervisor.log
redirect_stderr = true
environment=LANG=en_US.UTF-8,LC_ALL=en_US.UTF-8
```

Creamos los siguientes archivos que pusimos en el archivo clivet.conf para control de los logs:

```
mkdir -p /var/www/clivet/logs/
touch /var/www/clivet/logs/gunicorn_supervisor.log
```

Luego pedimos que vuelva a leer la carpeta.

```
sudo supervisorctl reread
sudo supervisorctl update
```

Verificamos el estado

```
sudo supervisorctl status clivet
sudo supervisorctl start clivet
```

5. Nginx

Usaremos nginx como un servidor proxy para nuestros archivos estáticos del proyecto:

```
sudo apt install nginx
sudo service nginx start
```

Creamos una configuración de servidor virtual Nginx para Django

Cada servidor virtual Nginx debe ser descrito por un archivo en el directorio /etc/nginx/sites-available. Seleccione los sitios que desea habilitar mediante enlaces simbólicos a los del directorio /etc/nginx/sites-enabled.
Cree un nuevo archivo de configuración del servidor nginx para su aplicación Django que se ejecuta en example.com en /etc/nginx/sites-available/clivet. El archivo debe contener algo en las siguientes líneas

```
clivet
```

```
upstream clivet_app_server {
  # fail_timeout=0 means we always retry an upstream even if it failed
  # to return a good HTTP response (in case the Unicorn master nukes a
  # single worker for timing out).

  server unix:/var/www/clivet/run/gunicorn.sock fail_timeout=0;
}
server {
    listen   80;
    server_name example.com;
    client_max_body_size 4G;
    access_log /var/www/clivet/logs/nginx-access.log;
    error_log /var/www/logs/clivet/logs/nginx-error.log;

    location /static/ {
        alias   /var/www/clivet/clivet/static/;
    }

    location /media/ {
        alias   /var/www/clivet/clivet/media/;
    }

    location / {
        # an HTTP header important enough to have its own Wikipedia entry:
        #   http://en.wikipedia.org/wiki/X-Forwarded-For
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

        # enable this if and only if you use HTTPS, this helps Rack
        # set the proper protocol for doing redirects:
        # proxy_set_header X-Forwarded-Proto https;

        # pass the Host: header from the client right along so redirects
        # can be set properly within the Rack application
        proxy_set_header Host $http_host;

        # we don't want nginx trying to do something clever with
        # redirects, we set the Host: header above already.
        proxy_redirect off;

        # set "proxy_buffering off" *only* for Rainbows! when doing
        # Comet/long-poll stuff.  It's also safe to set if you're
        # using only serving fast clients with Unicorn + nginx.
        # Otherwise you _want_ nginx to buffer responses to slow
        # clients, really.
        # proxy_buffering off;
```

```
        # Try to serve static files from nginx, no point in making an
        # *application* server like Unicorn/Rainbows! serve static files.
        if (!-f $request_filename) {
            proxy_pass http://clivet_app_server;
            break;
        }
    }

    # Error pages
    error_page 500 502 503 504 /500.html;
    location = /500.html {
        root /var/www/clivet/clivet/static/;
    }
}
```

Cree un enlace simbólico en la carpeta habilitada para sitios:

```
sudo ln -s /etc/nginx/sites-available/clivet /etc/nginx/sites-enabled/clivet
sudo service nginx restart #Reiniciar nginx
```

6. Estructura final de los archivos

```
(clivet) ➜  clivet tree
.
├── bin
│   ├── activate
│   ├── activate.csh
│   ├── activate.fish
│   ├── activate_this.py
│   ├── gunicorn
│   ├── gunicorn_paster
│   ├── gunicorn_start
├── clivet
│   ├── apps
│   │   ├── ...
│   ├── config
│   │   ├── settings
│   │   │   ├── base.py
```

```
|   |   |   ├── local.py
|   |   |   ├── production.py
|   |   |   └── test.py
|   |   ├── urls.py
|   |   └── wsgi.py
|   ├── db.sqlite3
|   ├── manage.py
|   ├── requirements
|   |   ├── base.txt
|   |   ├── local.txt
|   |   ├── production.txt
|   |   └── test.txt
|   ├── static
|   |   ├── ...
├── include
|   └── ...
├── lib [error opening dir]
├── logs
|   ├── gunicorn-error.log
|   ├── gunicorn_supervisor.log
|   ├── nginx-access.log
|   └── nginx-error.log
├── run
|   └── gunicorn.sock
└── share
    └── ..
```

7. Comandos más usados para mantenimiento

```
sudo supervisorctl status clivet
sudo supervisorctl restart clivet
sudo service nginx restart
sudo service nginx status
```

*Como esta configuración es una tare repetitiva se creó un script automatizado que hace todo el proceso en uno solo, podemos encontrar en la siguiente repositorio: https://github.com/yuselenin/clivet_config donde el archivo principal es el siguiente:

script_clivet.sh

```
echo "Iniciando $NAME como `whoami`"
#================================================================================
PROJECT='clivet'
GIT_REP='https://gitlab.com/wannabe/CLIVET.git'
USER='clivet_user'
#================================================================================
apt update
apt upgrade
echo "=============Users============"
groupadd --system webapps
useradd --system --gid webapps --shell /bin/bash --home /var/www/$PROJECT $USER
echo "=============Virtual============"
apt install python-virtualenv
mkdir -p /var/www/$PROJECT
cd /var/www/$PROJECT
git clone $GIT_REP $PROJECT
mkdir -p /var/www/$PROJECT/$PROJECT/temp/logs
chown $USER /var/www/$PROJECT/
cd /var/www/$PROJECT/
su - $USER
#write this comands
https://github.com/yuselenin/clivet_config/blob/master/configutation_in_user.sh
chown -R $USER:webapps /var/www/$PROJECT
chmod -R g+w /var/www/$PROJECT
echo "=============Gunicorn============"
cd /var/www/$PROJECT
chown -R $USER:users /var/www/$PROJECT
chmod u+x bin/gunicorn_start
wget
https://raw.githubusercontent.com/yuselenin/clivet_config/master/gunicorn_start
-P /var/www/$PROJECT/bin/
echo "=============supervisor============"
apt install supervisor
wget
https://raw.githubusercontent.com/yuselenin/clivet_config/master/clivet.conf -P
/etc/supervisor/conf.d/
mkdir -p /var/www/$PROJECT/logs/
touch /var/www/$PROJECT/logs/gunicorn_supervisor.log
chown -R $USER:webapps /var/www/$PROJECT
supervisorctl reread
supervisorctl update
supervisorctl status $PROJECT
# supervisorctl restart $PROJECT
echo "=============nginx============"
apt install nginx
```

```
echo "STATUS"
service nginx start
wget https://raw.githubusercontent.com/yuselenin/clivet_config/master/clivet -P
/etc/nginx/sites-available/
ln -s /etc/nginx/sites-available/$PROJECT /etc/nginx/sites-enabled/$PROJECT
service nginx restart
```

8. Referencias

- Nginx:
  - https://www.nginx.com/resources/wiki/
- Supervisor:
  - http://supervisord.org/
- Gunicorn:
  - http://gunicorn.org
- Configurations:
  - http://docs.gunicorn.org/en/stable/deploy.html
  - https://gist.github.com/Atem18/4696071