Mettre en production une application Python Flask sur Debian

Baptiste MEGEVAND

Table des matières

Prérequis et liens

1. Une VM Linux Debian
2. Droit d'administration sur la VM
3. Python 3.X d'installé
4. vim d'installé, sinon remplacer les commande « vi » par « nano »
5. Apache2 d'installé (sudo apt-get install apache2)

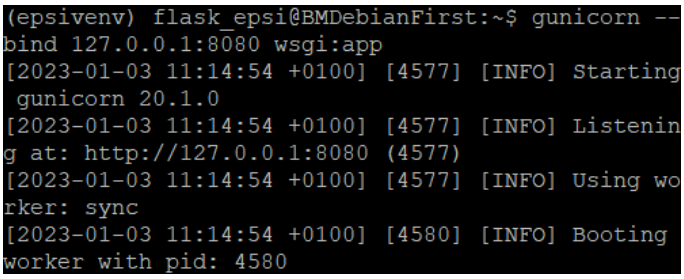
Lien utilisé pour construire le document : https://docs.gunicorn.org/en/stable/settings.html

1 - Configuration python et répertoire

sudo adduser flask_epsi --disabled-password	Créer un utilisateur de service qui permettra de déposer l'application, sans mot de passe pour qu'il ne puisse pas se connecter
sudo apt-get install python3-venv sudo pip3 install virtualenv	Sur Debian python 3 est installé par défaut, il faut juste ajouter le venv pour pouvoir cloisonner nos applications
sudo su flask_epsi cd mkdir logs	Se déplacer sur l'utilisateur nouvellement créé et ajout d'un répertoire pour les logs dans son home
python3 -m venv epsivenv source epsivenv/bin/activate pip install flask	Création du venv dans le home puis installation des modules requis, ici seulement flask pour notre exemple
vi app.py *from flask import Flask* *app = Flask(__name__)* *@app.route('/')* *def get_square():* *return f" Salut à tous c'est Fanta "* *if __name__ == '__main__':* *app.run(host='127.0.0.1', port=8080, debug=True)*	Création d'une petite application Flask répondant un objet HTML sur « / »
python app.py curl -i http://127.0.0.1:8080 HTTP/1.1 200 OK Server: Werkzeug/2.2.2 Python/3.7.3 Date: Tue, 03 Jan 2023 10:01:08 GMT Content-Type: text/html; charset=utf-8 Content-Length: 32 Connection: close Salut à tous c'est Fanta	Lancer l'application Sur un autre terminal utiliser curl pour tester si l'application répond Exemple de retour de la commande curl

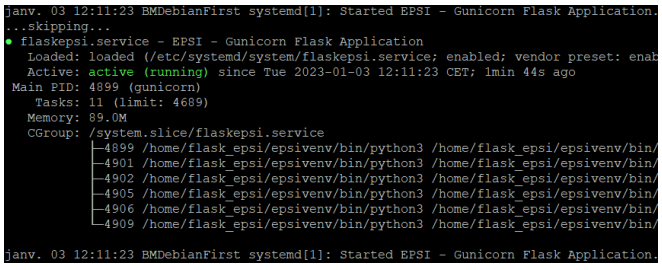
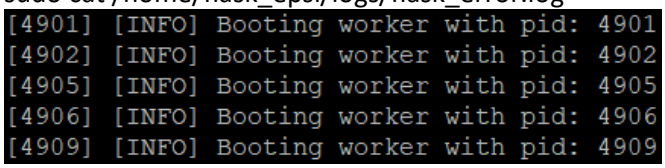
2 - Configurer le module python Gunicorn

source epsivenv/bin/activate pip install gunicorn	Si le venv n'est pas actif, l'activer Installer gunicorn dans le venv
vi wsgi.py *from app import app* *if __name__ == '__main__':* *app.run()*	Créer un fichier wsgi.py qui va récupérer l'application Flask créée précédemment et la run, à noter, il ne prendra pas en compte les configurations « serveur » (IP, port) réalisées dans le fichier app.py

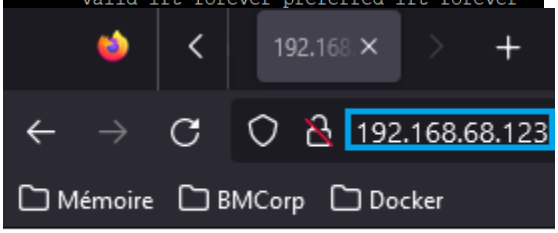
gunicorn --bind 127.0.0.1:8080 wsgi:app
 	
 
`(epsivenv) flask_epsi@BMDebianFirst:~$ gunicorn --bind 127.0.0.1:8080 wsgi:app`
`[2023-01-03 11:14:54 +0100] [4577] [INFO] Starting gunicorn 20.1.0`
`[2023-01-03 11:14:54 +0100] [4577] [INFO] Listening at: http://127.0.0.1:8080 (4577)`
`[2023-01-03 11:14:54 +0100] [4577] [INFO] Using worker: sync`
`[2023-01-03 11:14:54 +0100] [4580] [INFO] Booting worker with pid: 4580`
	Lancer l'application avec gunicorn
	
	Exemple de retour si le lancement a fonctionné

vi config.py *import multiprocessing* *workers = multiprocessing.cpu_count() * 2 + 1* *bind = 'unix:flaskepsi.sock'* *umask = 0o007* *reload = True* *#logging* *accesslog = '/home/flask_epsi/logs/flask_access.log'* *errorlog = '/home/flask_epsi/logs/flask_error.log'*	Créer un fichier config.py pour les configurations du gunicorn Ici dans 'bind' nous créons un socket unix pour que l'application ne réponde qu'à notre machine linux, mais il est également possible de bind simplement une IP d'écoute (ex : 127.0.0.1) Faire pointer les logs vers le répertoire créé précédemment
Ctrl + C deactivate Ctrl + D	Pour stopper gunicorn Pour quitter le venv Pour retourner vers l'utilisateur avec les droits sudo

3 - Configuration du service Linux

sudo usermod flask_epsi -aG www-data	Ajouter l'utilisateur de service au groupe apache2 (www-data pour Debian, peu changer suivant les distributions)
sudo vi /etc/systemd/system/flaskepsi.service *[Unit]* *Description=EPSI - Gunicorn Flask Application* *After=network.target* *[Service]* *User=flask_epsi* *Group=www-data* *WorkingDirectory=/home/flask_epsi/* *Environment="PATH=/home/flask_epsi/epsivenv/bin"* *ExecStart=/home/flask_epsi/epsivenv/bin/gunicorn --config config.py wsgi:app* *[Install]* *WantedBy=multi-user.target*	Création du fichier service qui servira au lancement de l'application
sudo systemctl start flaskepsi.service sudo systemctl enable flaskepsi.service	Démarrage du service Activation du service
sudo systemctl status flaskepsi.service  	Si tout se passe bien, le statut est en « active (running) »
sudo cat /home/flask_epsi/logs/flask_error.log  	Dans nos logs, nous pouvons voir que les workers gunicorn démarre bien, il peut y avoir des erreurs d'overload, il faut alors réduire le nombre de workers

4 – Configuration du reverse proxy sur Apache2

sudo systemctl enable apache2.service	S'il n'est pas actif au démarrage, activer apache2
sudo mkdir /home/logs	Création d'un répertoire logs pour récupérer les logs apaches
sudo vi /etc/apache2/sites-available/flaskepsi.conf *<VirtualHost *:80>* *ServerAdmin root@BMDebianFirst* *ErrorLog /home/logs/flaskepsi-error.log* *CustomLog /home/logs/flaskepsi-access.log combined* *<Location />* *ProxyPass unix:/home/flask_epsi/flaskepsi.sock|http://127.0.0.1/* *ProxyPassReverse unix:/home/flask_epsi/flaskepsi.sock|http://127.0.0.1/* *</Location>* *</VirtualHost>*	Créer le site apache2 avec la configuration reverse proxy sur notre socket créé à l'étape précédente
sudo a2enmod proxy sudo a2enmod proxy_http sudo a2enmod ssl	Activation des modules apache nécessaire au reverse proxy Module SSL optionnel
sudo a2ensite flaskepsi	Activation du site que l'on vient de créer
baptistem@BMDebianFirst:/home/logs$ ip a s 1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc no 000 link/loopback 00:00:00:00:00:00 brd 00:00:00 inet 127.0.0.1/8 scope host lo valid_lft forever preferred_lft forever inet6 ::1/128 scope host valid_lft forever preferred_lft forever 2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu fault qlen 1000 link/ether 08:00:27:6b:93:69 brd ff:ff:ff:ff inet 192.168.68.123/24 brd 192.168.68.255 sc valid_lft 7078sec preferred_lft 7078sec inet6 fe80::a00:27ff:fe6b:9369/64 scope link valid_lft forever preferred_lft forever  Salut à tous c'est Fanta	Afin de vérifier que tout fonctionne correctement taper sur l'IP de la machine virtuelle Si ce n'est pas le cas observer les logs apache et flask pour comprendre et corriger les erreurs