Procédure d'Installation et de Déploiement - Application Web Django

Table des Matières

Table des Matières 1

1. Présentation de la Solution 2

2. Architecture Technique 2

3. Prérequis et Infrastructure 3

4. Procédure d'Installation 4

5. Configuration et Déploiement 5

6.Utilisation du serveur 7

Conclusion 7

1. Présentation de la Solution

1.1 Objectif

Cette procédure détaille l'installation et le déploiement d'une application web Django dans un environnement de production virtualisé, avec la migration d'une base de données SQLite existante vers une base de données vierge MariaDB.

1.2 Portée

- Migration d'une base de données SQLite vers le serveur de base de données MariaDB
- Déploiement sur infrastructure de machines virtualisées (VirtualBox)
- Configuration de production avec les services web Gunicorn et Nginx
- Sécurisation et optimisation des performances

2. Architecture Technique

2.1 Vue d'Ensemble

L'architecture repose sur deux machines virtuelles distinctes :

- **App Server** : Serveur d'application (Django + Gunicorn + Nginx)
- **DB Server** : Serveur de base de données (MariaDB)

2.2 Configuration Réseau

- **App Server** :
  - Interface NAT : 10.0.2.15 (accès Internet - DHCP)
  - Interface Bridge : 192.168.10.1 (communication inter-VM - Statique)
- **MariaDB Server** :
  - Interface Bridge : 192.168.10.2 (serveur Mariadb)
  - Passerelle : 192.168.10.1

2.3 Stack Technologique

- **Frontend** : Django Templates

- **Backend** : Django Framework
- **Serveur WSGI** : Gunicorn
- **Serveur Web** : Nginx
- **Base de données** : MariaDB
- **Virtualisation** : VirtualBox

3. Prérequis et Infrastructure

3.1 Matériel Requis

- 2 machines virtuelles avec au minimum :
  - **App Server** :

```
service NetworkManager stop
nano /etc/network/interfaces

# Interface NAT (enp0s3)

auto enp0s3
iface enp0s3 inet dhcp

# Interface accès par pont (enp0s8)

auto enp0s8
iface enp0s8 inet static
   address 192.168.10.1
   netmask 255.255.255.0
```

  - **DB Server** :

```
service NetworkManager stop
nano /etc/network/interfaces

# Interface accès par pont (enp0s3)

auto enp0s3
iface enp0s3 inet static
   address 192.168.10.1
   netmask 255.255.255.0

       gateway 192.168.10.2
```

3.2 Logiciels Requis

```
# App Server - Serveur d'application
sudo apt update && sudo apt install python3 python3-pip python3-venv nginx git

# DB Server - Serveur de base de données
sudo apt update && sudo apt install mariadb-server
```

4. Procédure d'Installation

4.1 Configuration du Serveur MariaDB (DB Server)

4.1.1 Installation et Sécurisation

```
# Installation
sudo apt install mariadb-server -y
```

4.1.2 Configuration des Accès Distants

```
# Modification du fichier de configuration
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

# Modifier la ligne :
bind-address = 0.0.0.0

# On rajoute :
port = 3306

# Redémarrage du service
sudo systemctl restart mariadb
```

4.1.3 Création de la Base et de l'Utilisateur

```
mysql -u root -p

CREATE DATABASE nom_base CHARACTER SET utf8mb4;
CREATE USER 'username'@'% (ou @IP de la VM2)' IDENTIFIED BY 'mot__passe';
# '%' permettra d'autoriser l'utilisateur depuis n'importe quelle machine
GRANT ALL PRIVILEGES ON nom_base.* TO 'toto'@'%';
FLUSH PRIVILEGES;
EXIT;
```

4.2 Configuration du Serveur d'Application (App Server)

4.2.1 Création de l'Environnement Python

```
# Création du répertoire projet
mkdir ~/Myproject
cd ~/Myproject

# Création et activation de l'environnement virtuel
python3 -m venv venv
source venv/bin/activate
```

```
# Installation des dépendances
pip install django gunicorn mysqlclient reportlab pillow
```

4.2.2 Configuration Django

Modifier le fichier `settings.py` :

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql', # remplacer ‘.sqlite’ par ‘.mysql’
        'NAME': 'nom_base',
        'USER': 'toto',
        'PASSWORD': 'mot_de_passe',
        'HOST': '192.168.10.2',  # localisation de la BDD, ici notre VM MariaDB
        'PORT': '3306',    # port ‘mysql’ par défaut
    }
}

# Configuration des fichiers statiques
STATIC_URL = '/static/'    # l’affichage dans l’URL
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')  #dossier créé dans l’arborescence

# Configuration de production
DEBUG = False
ALLOWED_HOSTS = ['10.0.2.15', '192.168.10.1']
```

5. Configuration et Déploiement

5.1 Migration des Données

5.1.1 Sauvegarde SQLite

```
# Depuis l'ancienne configuration
python manage.py dumpdata > backup_data.json
```

5.1.2 Migration vers MariaDB

```
# Avec la nouvelle configuration MariaDB
python manage.py makemigrations
python manage.py migrate
python manage.py loaddata backup_data.json
```

5.2 Configuration de Gunicorn

5.2.1 Test Manuel

```
cd /home/toto/Myproject/myproject
gunicorn --bind 10.0.2.15:8000 nom_application.wsgi:application
```

5.2.2 Service Systemd

Créer `/etc/systemd/system/gunicorn.service` :

```
[Unit]
Description=Gunicorn daemon for Django
Requires=gunicorn.socket
After=network.target

[Service]
User=username
Group=www-data
WorkingDirectory=/home/username/Myproject/myproject
ExecStart=/home/username/Myproject/venv/bin/gunicorn
          --workers 3
          --bind unix:/home/toto/Administration/administration.sock
          nom_application.wsgi:application

[Install]
WantedBy=multi-user.target
sudo systemctl daemon-reload
sudo systemctl enable gunicorn
sudo systemctl start gunicorn
```

5.3 Configuration de Nginx

5.3.1 Configuration du Site

Créer `/etc/nginx/sites-available/nom_application` :

```
server {
    listen 80;
    server_name 10.0.2.15;

    location /static/ {
        alias /home/username/Myproject/myproject/staticfiles/;
    }
```

```
    location / {
        include proxy_params;
        proxy_pass http://unix:/run/gunicorn.socket;
    }
}
```

5.3.2 Activation du Site

```
sudo ln -s /etc/nginx/sites-available/django_app /etc/nginx/sites-enabled/
sudo nginx -t       # vérification de la syntaxe, sortie attendu : 'test is successful'
sudo systemctl restart nginx
```

5.4 Collecte des Fichiers Statiques

```
python manage.py collectstatic
sudo chown -R www-data:www-data /home/username/Myproject/myproject/staticfiles/
```

6.Utilisation du serveur

```
# dans un navigateur local
http://<@_ip_VM_AppServer>       # ici http://10.0.2.15
```

Conclusion

Cette procédure garantit un déploiement robuste et sécurisé d'une application Django en production. L'architecture à deux niveaux (application/base de données) offre une séparation claire des responsabilités et facilite la maintenance.

Points clés à retenir :

- Configuration réseau Bridge pour la communication inter-VM
- Sécurisation des accès MariaDB
- Séparation des environnements (développement/production)
- Monitoring continu des services