OC Pizza

Système de gestion de pizzeria

Dossier d'exploitation

Version 1.0

Auteur
Adrien SIMON
Développeur

TABLE DES MATIÈRES

VERSIONS

Auteur	Date	Description	Version
Adrien SIMON	26/12/2020	Création du document	0.1
Adrien SIMON	26/12/2020	Remplissage de l'introduction	0.2
Adrien SIMON	27/12/2020	Remplissage des pré-requis	0.3
Adrien SIMON	27/12/2020	Remplissage des procédures de déploiement	0.4
Adrien SIMON	27/12/2020	Remplissage des procédures de démarrage / arrêt	0.5
Adrien SIMON	27/12/2020	Remplissage des procédures de mise à jour	0.6
Adrien SIMON	27/12/2020	Remplissage des procédures de supervision	0.7
Adrien SIMON	27/12/2020	Remplissage des procédures de sauvegarde	0.8
Adrien SIMON	27/12/2020	Nettoyage du document, version finale	1.0

INTRODUCTION

Objet du document

Le présent document constitue le dossier d'exploitation de l'application OC pizza...

Le but de ce document est de renseigner les informations et procédures nécessaires au fonctionnement de l'application, c'est à dire le serveur, le déploiement, la configuration etc

Références

Pour de plus amples informations, se référer :

- **Documentation_technique_adrien_simon.pdf** : Dossier de conception technique de l'application
- **Documentation_fonctionnelle_adrien_simon.pdf** : Dossier de documentation fonctionnelle de l'application

Pré-requis

Système

Serveur de Base de données

ip: 51.68.121.152

hébergeur: OVH

***utilisateur**: debian*

***mot de passe**: fourni dans la base keepass de l'entreprise*

Caractéristiques techniques:

- **Système d'exploitation**: *debian 10*
- ***vCores: 1***
- ***Mémoire: 2 Go***
- ***Stockage: 40 Go***

Serveur Web

ip: 51.68.122.183

hébergeur: OVH

Système d'exploitation: *debian 10*

***utilisateur**: debian*

***mot de passe**: fourni dans la base Keepass de l'entreprise*

Caractéristiques techniques:

- **Système d'exploitation**: *debian 10*
- ***vCores: 1***
- ***Mémoire: 2 Go***
- ***Stockage: 40 Go***

Bases de données

Les bases de données et schémas suivants doivent être accessibles et à jour :

- **mariaDB :** version 10.3.27
- **Schema :** OC_Pizza

Web-services

Les web services suivants doivent être accessibles et à jour :

- **Google maps :** version 3.43.3
- **Stripe :** version 2020-08-27

PROCÉDURE DE DÉPLOIEMENT

Déploiement des Batches

Artefacts

Les batches de l'application OC pizza sont construits sous la forme d'une archive ZIP contenant les répertoires :

- **bin** : les scripts SH de lancement des différents batches

Extraire l'archive **OC_pizza_batches.zip** dans le répertoire :

~/

Positionner les droits d'exécution sur les scripts SH de lancement des batches.

- **configure_nginx.sh**
- **configure_gunicorn.sh**

Lancer le script **configure_nginx.sh** puis suivez les instructions à l'écran, il créera la configuration de nginx en fonction des informations que vous lui donnerez.

Lancer le script **configure_gunicorn.sh** puis suivez les instructions à l'écran, il créera un service système qui permettra de lancer et arrêter votre application facilement avec systemctl.

Example de résultat de lancement des batches

Configuration de nginx:

```
$ sudo ./configure_nginx.sh

Quelle est l'ip / nom de domaine de votre serveur : 51.68.122.183

Dans quel dossier se trouvent vos fichiers statiques: /home/debian/P9_Adrien_Simon

Votre fichier de configuration nginx a été généré à l'emplacement
/etc/nginx/sites-enabled/oc_pizza
```

Configuration de gunicorn:

```
$ sudo ./configure_nginx.sh

Quel est votre nom d'utilisateur: debian

Dans quel dossier se trouve l'application: /home/debian/P9_Adrien_Simon

Les fichiers gunicorn.socket et gunicorn.service ont été générés à l'emplacement
/etc/systemd/system/
```

Vérifications

Vérifier que le fichier de configuration de nginx est bien présent et est au format suivant:

```
$ less /etc/nginx/sites-enabled/oc_pizza

server {
    listen 80;
    server_name 51.68.122.183;

    location = /favicon.ico { access_log off; log_not_found off; }
    location /static/ {
        root /home/debian/P9_Adrien_Simon;
    }

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

Vérifier que le proxy fonctionne bien quand on met le server_name dans un navigateur, on tombera soit sur une page nginx ou soit sur l'application si tout a bien été lancé.

Vérifier que gunicorn tourne bien avec **`sudo systemctl status gunicorn`**, si il tourne bien vous devriez voir un texte en vert avec marqué active (running)

Déploiement de l'Application Web

Environnement de l'application web

Variables d'environnement

La configuration de l'application devrait se faire automatiquement grâce au script gunicorn (avant de lancer le service il clone l'application et fait la configuration).

Si vous avez un problème vérifiez que le dossier de l'application a bien été créé et qu'il contient bien un fichier **.env** contenant:

- SECRET_KEY="clé secrète django"

- ENVIRONMENT=PROD

- MYSQL_PASSWORD="mdp de la bdd"

DataSources

La base de données devrait déjà être créée et configurée sur le serveur de base de données, si ce n'est pas le cas lancez la commande:

sudo apt install mariadb-server=1:10.3.27-0+deb10u1

Ensuite, ouvrez le fichier **/etc/mysql/mariadb.conf.d/50-server.cnf** avec l'éditeur de texte de votre choix et changez la ligne **bind: localhost** par **bind: 0.0.0.0** ce qui correspond a toutes les interfaces réseau, pour pouvoir permettre l'accès de l'extérieur.

Après avoir fait ça ouvrez le port 3306 au serveur applicatif avec la commande:

sudo iptables -A INPUT -i eth0 -s 51.68.122.183 -p tcp --destination-port 3306 -j ACCEPT

Pour créer les tables exécuter depuis le serveur de l'application:

- python manage.py makemigrations

- python manage.py migrate

Procédure de démarrage / arrêt

Base de données

Démarrer la base de données: sudo service start mariadb

Arrêter la base de données: sudo service stop mariadb

Application web

Démarrer l'application: sudo systemctl start gunicorn

Arréter l'application: sudo systemctl stop gunicorn

Procédure de mise à jour

Base de données

- sudo apt update

- sudo apt install mariadb-server

Application web

- cd P9_Adrien_Simon

- git pull origin main

- source venv/bin/activate

- pip install -r requirements.txt

- python manage.py makemigrations

- python manage.py migrate

- sudo systemctl restart gunicorn

Supervision/Monitoring

Supervision de l'application web

Afin de tester que l'application web est toujours fonctionnelles, faire ceci:

- Aller vérifier qu'on peut bien y accéder depuis le navigateur

Si ce n'est pas le cas:

- Vérifier que l'application tourne toujours avec **sudo systemctl status gunicorn**

- Vérifier les logs de l'application avec la commande: **sudo journalctl -u gunicorn**

Monitoring des serveurs

Pour vérifier que les serveurs tournent sans souci on utilisera les données de monitoring d'OVH qui donnent toutes les informations dont on a besoin.

Procédure de sauvegarde et restauration

Pour procéder aux sauvegardes des serveurs nous utiliserons REOBack

Pour celà nous aurons configuré le fichier **/etc/reoback/settings.conf** tel que:

```
remotebackup = 1
```

```
rbackuptype = FTP
remotehost = serveur de sauvegarde
remotepath = /OC_pizza_backups/
ftpuser = BackupUser
ftppasswd = BackupPassword
```

Ainsi que le fichier **etc/reoback/files.conf** tel que:

File: home

/home/debian

File: nginx

/etc/nginx/sites_available

File: gunicorn_socket

/etc/systemd/system/gunicorn.socket

File: gunicorn_service

/etc/systemd/system/gunicorn.service

Le backup sera lancé par cette tâche cron tous les jours à minuit:

0 0 * * * /etc/reoback/run_reoback.sh

Pour restaurer ces fichiers nous récupérerons simplement ceux-ci du backup et les remettront à leur emplacement