OC Pizza

Gestion de pizzeria - application web

Dossier d'exploitation

Version 1.1

Auteur
Jody Etienne
Analyste-programmeur

TABLE DES MATIÈRES

1 - Versions

Auteur	Date	Description	Version
Jody Etienne	30/09/2020	Création du document	1.1

2 - Introduction

2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application OC Pizza.
Nous nous intéressons aux besoins de nos clients tels que la gestion des utilisateurs, des commandes, des paiements à travers les spécifications techniques pour aboutir à un site de qualité qui répond aux besoins de OC Pizza.
Les éléments du présents dossiers découlent d'une prise de contact avec l'un des responsable du groupe.

Objectif :

Mise en place d'un nouveau système informatique pour l'ensemble des pizzerias du groupe OC Pizza.

2.2 - Références

Pour de plus amples informations, se référer également aux éléments suivants :

1. **DCF** : Dossier de conception fonctionnelle de l'application
2. **DCT** : Dossier de conception technique de l'application
3. **PVL** : Procès Verbal de livraison

3 - Pré-requis

3.1 - Système

3.1.1 - Serveur de Base de données

Le Serveur de Base de Données Relationnelle sera **PostgreSQL**. Il sera implanter lors de la préparation du serveur.

De plus, un utilisateur avec les pleins pouvoirs sera créé.

3.1.1.1 - Caractéristiques techniques

Emplacement du datacenter : Gravelines, France

Offre : **Comfort**

Processeur : **4 vCore**

Mémoire : **8 Go**

Stockage : **160 Go SSD NVMe**

Bande passante publique : **1 Gbit/s**

Image : **Ubuntu 20.04**

3.1.2 - Serveur Web

Serveur Virtuel Privé hébergeant l'application web : **VPS Comfort** chez OVH

3.1.2.1 - Caractéristiques techniques

Les mêmes que le serveur de base de données.

3.2 - Bases de données

L'ORM de Django est utilisé. Les bases de données et schémas suivants doivent être accessibles et à jour :

- **PostgreSQL :** version 13

3.3 - Web-services

Les web services suivants doivent être accessibles et à jour :

- **Django :** version 3.1
- **Nginx :** version 1.19.3
- **Gunicorn :** version 20.0.4

3.4 - Autres Ressources

- **Python** : version 3.8.6
- Inscription à un compte gratuit pour la gestion des paiement en ligne : **API Mollie**
- Abonnement au service de paiement en ligne : **Paypal**
- Compte avec facturation enregistrée sur Google Cloud Platform : **Geocoding API, Directions API**

4 - Procédure de déploiement

4.1 - Déploiement de l'Application Web

4.1.1 - Variables d'environnement

Créer un fichier .env dans le dossier racine de ocpizza et insérer ces variables :

```
DJANGO_SECRET_KEY=<ocpizza_django_secret_key>
DEBUG=<True>
DJANGO_ALLOWED_HOSTS=<IPv4_du_VPS>
DATABASE_ENGINE=<postgresql_psycopg2>
DATABASE_NAME=<ocpizza_prod_db>
DATABASE_USERNAME=<u_ocpizza>
DATABASE_PASSWORD=<ocpizza_prod_password>
DATABASE_HOST=<ocpizza_prod_host>
DATABASE_PORT=<ocpizza_prod_port>
DJANGO_LOGLEVEL=<info>
MOLLIE_API_KEY=<ocpizza_mollie_api_key>
GEOCODING_API_KEY=<ocpizza_geocoding_api_key>
DIRECTIONS_API_KEY=<ocpizza_directions_api_key>
```

Editer les valeurs suivantes entre les balises <...>

4.1.2 - Déploiement

>> Prérequis :

- Avoir réservé le VPS sur le site OVHcloud.
- Être connecté à l'espace client OVHcloud.
- Disposer des informations de connexion envoyées par e-mail après l'installation.

>> Connecter l'utilisateur :

```
ssh nom_d_utilisateur@IPv4_du_VPS
```

Étant donné que vous êtes maintenant connecté avec les privilèges root (un utilisateur sudo),

vous pouvez entrer des commandes pour effectuer des tâches administratives. Il est recommandé de modifier votre mot de passe au préalable :

```
$ sudo passwd
New password:
Retype new password:
passwd: password updated successfully
```

À noter que les mots de passe ne sont pas affichés. Basculez ensuite vers l'utilisateur "root" et définissez votre mot de passe admin :

```
$ sudo su -
# passwd
New password:
Retype new password:
passwd: password updated successfully
```

>> Mise à jour des dépendances de Ubuntu :

```
$ sudo apt -y update
# list upgradable
$ sudo apt -y upgrade
```

>> Installer les dépendances pour utiliser PostgreSQL avec Python/Django :

```
$ sudo apt -y install build-essential libpq-dev python3-dev
```

>> Installer le serveur PostgreSQL :

```
$ sudo apt -y install postgresql postgresql-contrib
```

>> Installer Nginx, qui sera utilisé pour servir des assets static (css, js, images) et aussi pour exécuter l'application Django derrière un serveur proxy :

```
$ sudo apt -y install nginx
```

>> Supervisor démarrera le serveur d'application et le gérera en cas de panne ou de redémarrage du serveur :

```
$ sudo apt -y install supervisor
```

>> Activez et démarrez Supervisor :

```
$ sudo systemctl enable supervisor
$ sudo systemctl start supervisor
```

>> Installer Pipenv, 'environnement virtuel Python :

```
$ sudo apt -y install pipenv
```

>> Configurer la base de donnée PostgreSQL :

```
$ sudo - postgres
```

>> Créer un utilisateur de la base de données ocpizza et la base de données d'application :

```
$ createuser u_ocpizza
$ createdb ocpizza_prod_db --owner u_ocpizza
$ psql -c "ALTER USER u_ocpizza WITH PASSWORD 'ocpizza_prod_passwd'
```

Avertissement : veillez à choisir un mot de passe sécurisé. 'ocpizza_prod_passwd' est une solution de simplicité.

>> Retourner à l'utilisateur root :

```
$ exit
```

>> Créer un nouvel utilisateur :

```
$ adduser ocpizza
```

>> Configuration du nouvel utilisateur :

```
Adding user `ocpizza' ...
Adding new group `ocpizza' (1000) ...
Adding new user `ocpizza' (1000) with group `urban' ...
Creating home directory `/home/ocpizza' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
```

```
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for ocpizza
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
```

>> Ajouter l'utilisateur ocpizza à la liste des sudoers :

```
$ gpasswd -a ocpizza sudo
```

>> Basculer vers l'utilisateur nouvellement créé :

```
$ su - ocpizza
```

>> Activer Pipenv, l'environnement virtuelle Python :

```
$ pipenv shell
```

>> Cloner le repository Github de ocpizza :

```
$ git clone https://github.com/etiennody/ocpizza.git && cd ocpizza
```

>> Installer les dépendances :

```
$ pipenv install
```

>> Configurer les identifiants de connexion à la base de données dans le fichier settings.py :

```
...
DATABASES = {
   "default": {
      "ENGINE": f"django.db.backends.{os.environ.get("DATABASE_ENGINE", "sqlite3")}",
      "NAME": os.environ.get("DATABASE_NAME", "polls"),
```

```
        “USER”: os.environ.get(“DATABASE_USER”, “myprojectuser”),
        “PASSWORD”: os.environ.get(“DATABASE_PASSWORD”, “password”),
        “HOST”: os.environ.get(“DATABASE_HOST”, “localhost”),
        “PORT”: os.environ.get(“DATABASE_PORT”, “5432”),
    }
}
...
```

>> Migrer la base de données :

```
$ python3 manage.py migrate
```

>> Collecter les fichiers static :

```
$ python3 manage.py collectstatic
```

>> Tester le serveur de développement si tout fonctionne jusqu’à maintenant :

```
$ python3 manage.py runserver IPv4_du_VPS:8000
```

>> Pour quitter le serveur Django :

```
CTRL-C
```

>> Installer Gunicorn

```
$ pipenv install gunicorn
```

>> Créer un fichier nommé gunicorn_start à l’intérieur du dossier /bin :

```
$ vim bin/gunicorn_start
```

>> Ajouter les informations suivantes :

home/ocpizza/bin/gunicorn_start

```
#!/bin/bash


NAME="ocpizza"
```

```
DIR=/home/ocpizza/ocpizza
USER= ocpizza
GROUP= ocpizza
WORKERS=3
BIND=unix:/home/ocpizza/run/gunicorn.sock
DJANGO_SETTINGS_MODULE=ocpizza.settings
DJANGO_WSGI_MODULE=ocpizza.wsgi
LOG_LEVEL=error

# On active l'environnement virtuel
cd $DIR
pipenv shell

export DJANGO_SETTINGS_MODULE=$DJANGO_SETTINGS_MODULE
export PYTHONPATH=$DIR:$PYTHONPATH

exec ../bin/gunicorn ${DJANGO_WSGI_MODULE}:application \
 --name $NAME \
 --workers $WORKERS \
 --user=$USER \
 --group=$GROUP \
 --bind=$BIND \
 --log-level=$LOG_LEVEL \
 --log-file=-
```

>> Rendre le fichier gunicorn_start exécutable :

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

>> Créer un dossier run pour le ficher de socket unix :

```
$ mkdir run
```

>> Configuration de Supervisor pour prendre soin du lancement du serveur de gunicorn :

>> Créer un dossier nommé logs :

```
$ mkdir logs
```

>> Créer un fichier pour journaliser les erreurs d'application :

```
$ touch logs/gunicorn-errors.log
```

>> Créer un fichier de configuration pour Supervisor :

```
$ sudo vim /etc/supervisor/conf.d/ocpizza.conf
```

/etc/supervisor/conf.d/ocpizza.conf

```
[program:ocpizza]
command=/home/ocpizza/bin/gunicorn_start
user= ocpizza
autostart=true
autorestart=true
redirect_stderr=true
stdout_logfile=/home/ocpizza/logs/gunicorn-error.log
```

>> Relire le fichier de configuration de Supervisor et rendre le programme accessible :

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

>> Ajouter un nouveau fichier de configuration pour nginx nommé ocpizza dans /etc/nginx/sites_available/ et insérer ces informations:

```
$ sudo vim /etc/nginx/sites-available/ocpizza
```

/etc/nginx/sites-available/ocpizza

```
upstream app_server {
   server unix:/home/ocpizza/run/gunicorn.sock fail_timeout=0;
}

server {
```

```
    listen 80;

    # add here the ip address of your server
    # or a domain pointing to that ip (like example.com or www.example.com)
    server_name www.ocpizza.fr ocpizza.fr;

    keepalive_timeout 5;
    client_max_body_size 4G;

    access_log /home/ocpizza/logs/nginx-access.log;
    error_log /home/ocpizza/logs/nginx-error.log;

    location /static/ {
        alias /home/ocpizza/static/;
    }

    # checks for static file, if not found proxy to app
    location / {
        try_files $uri @proxy_to_app;
    }

    location @proxy_to_app {
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_set_header Host $http_host;
      proxy_redirect off;
      proxy_pass http://app_server;
    }
}
```

>> Créer un lien symbolique vers le dossier sites-enabled :

```
$ sudo ln -s /etc/nginx/sites-available/ocpizza /etc/nginx/sites-enabled/ocpizza
```

>> Enlever le site par défaut de Nginx :

```
$ sudo rm /etc/nginx/sites-enabled/default
```

>> Relancer Nginx :

```
$ sudo service nginx restart
```

>> Tester au final en redémarrant la machine pour vérifier si tout se relance automatiquement :

```
$ sudo reboot
```

Pour activer le HTTPS, nous utiliserons Let'sEncrypt et Cerbot :

```
# ajout du dépôt de cerbot
$ sudo add-apt-repository ppa:certbot/certbot
$ sudo apt update

# installation de certbot
sudo apt-get install python-certbot-nginx

# lancement de cerbot
sudo certbot --nginx -d www.ocpizza.com  -d ocpizza.com
```

>> Pour le renouvellement des certificats SSL:

```
$ sudo cerbot renew --dry-run
```

5 - Procédure de démarrage / arrêt

5.1 - Base de données

Démarrage :

```
$ sudo service postgresql start
```

Arrêt :

```
$ sudo service postgresql stop
```

5.2 - Application web

Démarrage :

```
$ sudo service nginx start
```

Arrêt :

```
$ sudo service nginx stop
```

6 - Procédure de mise à jour

6.1 - Base de données

```
$ ssh ocpizza@IPv4_du_VPS

$ pipenv shell
$ cd ocpizza
$ git pull origin master
$ python3 manage.py migrate
$ exit
```

6.2 - Application web

```
$ ssh ocpizza@IPv4_du_VPS

$ pipenv shell
$ cd ocpizza
$ git pull origin master
$ python3 manage.py collectstatic
$ sudo supervisorctl restart ocpizza
$ exit
```

7 - Supervision/Monitoring

7.1 - Supervision de l'application web

Sentry est un tableau de bord qui vous permet de visualiser ce qui se passe dans l'application Django. La configuration sera effectuée de sorte à remonter les logs d'erreurs et d'évènements de l'application.

On pourra ajouter des évènements tels que :

- L'inscription d'un nouvel utilisateur
- La désinscription d'un utilisateur
- L'utilisation de l'aide-mémoire des recettes
- Les paiements en ligne validés
- Les livraisons validées
- Les ingrédients en rupture de stock

Prérequis :

- Avoir créé et configuré un compte Sentry.io pour l'application Django ocpizza.
- Être connecté à l'espace client Sentry.io.

>> Ouvrir le lien suivant :

https://sentry.io/{nom_utilisateur}/ocpizza

7.2 - Monitoring de l'application web

Dans l'interface d'administration d'OVHCloud dans l'onglet « Monitoring », vous avez accès à des informations.

Vous pouvez surveiller :

- L'utilisation du processeur.
- L'utilisation de la mémoire virtuelle.
- L'utilisation du trafic.

>> Ouvrir le lien suivant :

https://www.ovh.com/manager/dedicated/#/iaas/vps/{nom_vps}/monitoring

8 - Procédure de sauvegarde et restauration

Une sauvegarde du VPS (hors disques additionnels) est planifiée quotidiennement, exportée puis répliquée trois fois avant d'être disponible dans votre espace client.

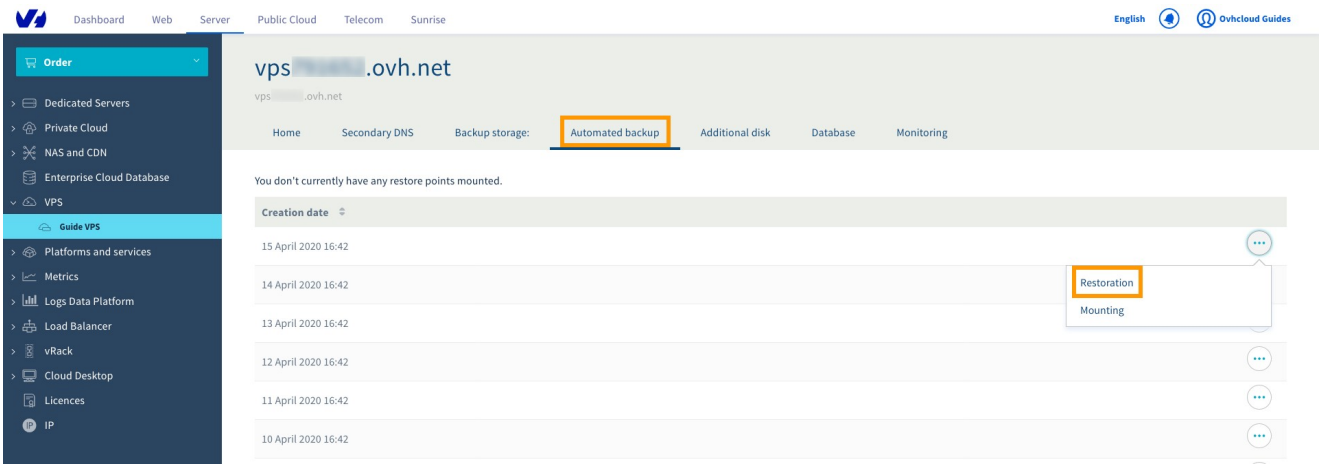
Prérequis :

- Être connecté à l'espace client OVHcloud.
- Un VPS OVHcloud déjà configuré.
- Un accès administrateur (root) en SSH à votre VPS (facultatif).

Connectez-vous à votre espace client OVHcloud, partie Server et sélectionnez votre serveur dans colonne de gauche sous la partie VPS .

>> Restaurer une sauvegarde à partir de l'espace client OVHcloud

Sélectionnez votre VPS puis cliquez sur l'onglet Backup automatisé dans le menu horizontal. Un maximum de 15 sauvegardes quotidiennes seront disponibles. Cliquez sur ... à droite de la sauvegarde à restaurer et sélectionnez Restauration.



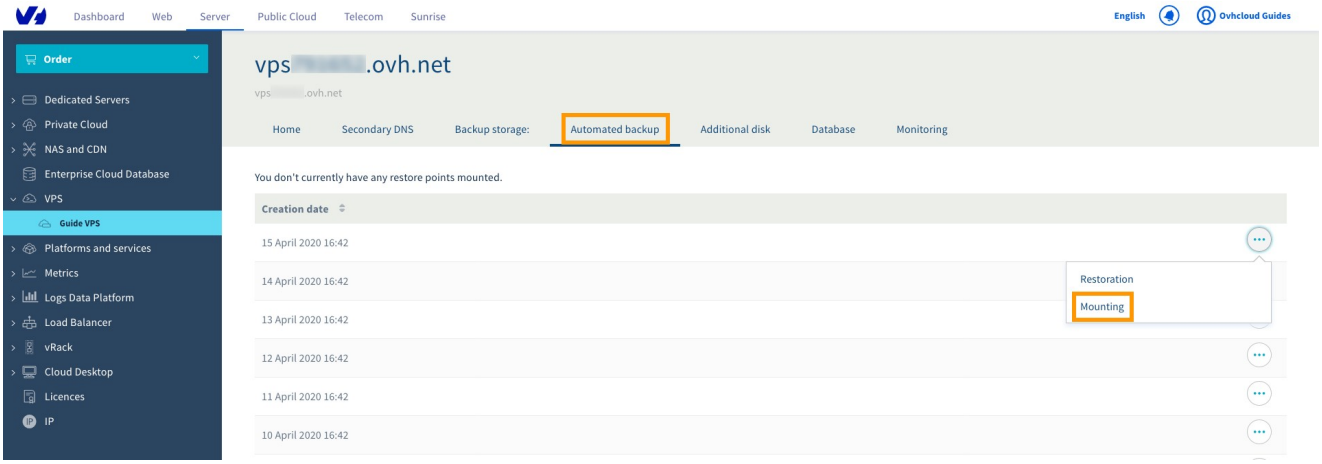
Si vous avez récemment modifié votre mot de passe root, assurez-vous de cocher l'option « Modifier le mot de passe root lors de la restauration » dans la fenêtre contextuelle pour conserver votre mot de passe root actuel et cliquez sur Confirmer. Vous recevrez un e-mail dès que la restauration sera terminée. Celle-ci peut prendre un certain temps, selon l'espace disque utilisé.

>> Monter et accéder à une image de sauvegarde

Cette option permet d'accéder aux données de sauvegarde au cas où vous ne souhaitez pas remplacer complètement votre service existant par la restauration.

> Étape 1 : depuis l'espace client

Cliquez sur ... à droite de la sauvegarde souhaitée et sélectionnez Montage :



Une fois le processus terminé, vous recevrez un e-mail. Vous pourrez alors vous connecter à votre VPS et ajouter la partition où se trouve votre sauvegarde.

> Étape 2 : en SSH

Connectez-vous au VPS ocpizza en SSH :

```
$ ssh nom_d_utilisateur@IPv4_du_VPS
```

Vous pouvez utiliser la commande suivante pour vérifier le nom du nouveau périphérique connecté :

```
$ lsblk
```

Voici un exemple de résultat de cette commande :

```
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda       8:0    0   25G  0 disk
├─sda1    8:1    0 24.9G  0 part /
├─sda14   8:14   0    4M  0 part
└─sda15   8:15   0  106M  0 part
```

```
sdb      8:16   0   25G  0 disk
├─sdb1   8:17   0 24.9G  0 part
├─sdb14  8:30   0    4M  0 part
└─sdb15  8:31   0  106M  0 part /boot/efi
sdc      8:32   0   50G  0 disk
```

Dans cet exemple, la partition contenant votre système de fichiers de sauvegarde est nommée “sdb1”. Créez à présent un répertoire pour cette partition et définissez-le comme point de montage :

```
$ mkdir -p /mnt/restore
$ mount /dev/sdb1 /mnt/restore
```

Vous pouvez maintenant basculer vers ce dossier et accéder aux données de sauvegarde.

9 - Glossaire

Monitoring	Surveillance à l'aide d'un moniteur.
Certificat SSL	Fichier de données qui lie une clé cryptographique aux informations d'une organisation.
SSH	Secure Shell (SSH) est à la fois un programme informatique et un protocole de communication sécurisé.
Vim	Editeur de texte en ligne de commande.

Sources :

https://docs.ovh.com/fr/vps/autobackup-vps/

http://www.vulgairedev.fr/blog/article/deploiement-django

https://medium.com/@panzelva/deploying-django-website-to-vps-with-uwsgi-and-nginx-on-ubuntu-18-04-db9b387ad19d

https://simpleisbetterthancomplex.com/tutorial/2016/10/14/how-to-deploy-to-digital-ocean.html

https://www.digitalocean.com/community/tutorials/how-to-deploy-a-local-django-app-to-a-vps

https://docs.ovh.com/fr/vps/debuter-avec-vps/