OC-Pizza

Pizza-Application

Dossier d'exploitation

Version 1.0

Auteur
Le Magorou Jean-Martial

Table des matières

Pharos-consulting
www.pharosinc.fr

55 rue du Faubourg St-Honoré – 75 008 Paris – infos@pharosinc.fr

SARL au capital de 1 000,00 € enregistrée au RCS de Paris – SIREN 999 999 999 – Code APE : 6202A

1 - Versions

Auteur	Date	Description	Version
JMLM	04/08/2020	Création du document	V 0.1
JMLM	05/08/2020	Fin + correction	V 0.2

2 - Introduction

2.1 - Objet du document

Le présent document constitue le volet “dossier d’exploitation” de la réponse à l’appel d’offre qu’a fait la société OC-Pizza afin de remplacer son Système d’Information actuel.

Ce document a pour objectif de présenter les différentes informations techniques permettant de comprendre et de refaire la mise en production de l’application OC-Pizza. Il doit permettre à toute personne ayant le bagage technique suffisant de maintenir, arrêter et redémarrer l’application

2.2 - Références

Pour de plus amples informations, se référer :

1. **P9 - Dossier de conception fonctionnelle** : Dossier de conception fonctionnelle de l’application
2. **P9 - Dossier de conception technique** : Dossier de conception technique de l’application
3. **P9 - PV Livraison** : PV de livraison de l’application

3 - Pré-requis

3.1 - Système

3.1.1 - Serveur

Le serveur hébergeant l’application est serveur VPS de la plate-forme OVH. Ce serveur est sous le système d ‘exploitation LINUX/Ubuntu. Pharos consulting dispose d’une matrice pré-paramétrée et sécurisée de ce système pour ses clients et l’utilisera pour OC-Pizza.

L’application sera accessible via l’adresse internet www.oc-pizza.fr. Des alias DNS seront créés et les noms de domaines acquis afin de pouvoir répondre à www.ocpizza.fr www.oc-pizza.com et www.ocpizza.com.

L’application sera sous un conteneur Docker. Cela permet entre autre :

- Une gestion des mises à jour et du déploiement simplifiée
- Une administration de l’application simplifiée et standardisée

3.1.1.1 - Caractéristiques techniques

Le serveur est un serveur Ubuntu 20,04 installé depuis une matrice Pharos consulting. Les mises à jour systèmes sont automatiquement installées et un mail au correspondant OC-Pizza sera automatiquement envoyé à chaque mise à jour. Les versions d'apache, docker… sont les versions standards d'Ubuntu et bénéficient ainsi des mises à jour automatiques. Les mises à jour du logiciel OC-Pizza sont traitées en 4 plus bas dans ce document.

Le serveur mis en place est déjà préparé, installé et configuré. Toutes les commandes d'installation du serveur sont mises dans un souci de documentation et n'ont pas besoin d'être repassées.

L'installation du serveur est donc une installation minimum avec en plus les paquets suivants :
- sudo
- net-tools
- openssh-server
- unattended-upgrade
- python3-pip
- docker
- docker-compose

S'il était besopinPour chaque paquet : Taper la commande suivante : ***apt-get -y install nom_du_paquet***

Le module python suivant est requis pour la surveillance de l'espace disque : psutil. Il doit etre installé en tapant la commande suivante : ***pip3 install psutil.***

Tout les reste (serveur web, connexion base de données…) est directement mis dans le conteneur Docker et ne nécessite donc pas d'installation lourde sur le serveur

3.1.2 - Serveur de Base de données

Le serveur de base de données est un serveur postgreSQL. Ce serveur est hébergé sur la plateforme Cloud-database d'OVH. Cela évite les taches d'administration et de sauvegarde de la base. Les paramètres de connexion sont déjà paramétrés dans le fichier prod_settings.py.

3.1.3 - Serveur Web

Le serveur WEB est un serveur Apache avec les modules WSGI et SSL intégrés et chargés.
Les fichiers statiques sont gérés par le module WhiteNoise et le serveur Apache directement.
Cette configuration est largement apte à supporter la charge présente et future du site.

3.1.3.1 - Caractéristiques techniques

La configuration est faite avec un virtualhost dont la configuration est ci-dessous.

Le fichier de configuration apache est dans les annexes.

L'utilisateur qui accédera en HTTP au site sera automatiquement routé vers le site en HTTPS. Cela sera transparent pour lui mais permet de sécuriser complètement les échanges entre le navigateur de l'utilisateur et le serveur.

3.2 - Web-services

Les web services suivants doivent être accessibles et à jour :

- Google Maps API : Cette API permet l'affichage de la map sur le site ainsi que la géolocalisation du client. Elle nécessite une clé pour fonctionner.
- Braintree : Cette API sert aux règlements directs sur site ainsi qu'a l'encaissement à la livraison. Elle nécessite une clé pour fonctionner.
- Nexmo : Cette API sert à l'envoi de SMS afin de prévenir le client de la fin de préparation de sa commande. Elle nécessite une clé pour fonctionner.

Toutes les clés sont stockées sur le serveur sous forme de variable d'environnement dans la configuration Apache (cf virtualhost). Le module mod_env permettant de les récupérer dans les programmes étant activé de base.

3.3 - Autres Ressources

Un compte utilisateur **ocpizza** a été créé sur le serveur. La clé ssh vous a été fournie et un logiciel permettant de s'y connecter installé sur les ordinateurs désignés. La procédure d'installation de ce logiciel gratuit vous a été fournie. Ce compte a été mis dans le groupe sudo, c'est-à-dire qu'en précédant les commandes du préfixe sudo, il dispose des autorisations du compte root du serveur. Cela est surtout utile pour docker qui demande systématiquement les droits root pour exécuter les commandes.

Toutes les installations systèmes nécessaires au bon fonctionnement du logiciel OC-Pizza ont été faites avant la mise à disponibilité du serveur.

Comme il est dit plus haut, le site fonctionne en HTTPS. Les fichiers nécessaires au SSL/TLS se trouvent dans le répertoire */etc/*ssl/ocpizza. La clé SSL/TLS que nous avons acquise est valable 36 mois. Il conviendra de la renouveler alors. Pour cela vous pourrez utiliser le fichier ocpizza.key existant, mais vous devrez régénérer une demande : Se mettre dans le répertoire */etc/*ssl/ocpizza, puis entrez la commande :
- ***sudo openssl req -new -key ocpizza.key -out ocpizza.csr.*** Il faut alors répondre aux questions puis se rendre sur le site de la société SSLS par exemple : https://www.ssls.com/ pour y acquérir une nouvelle clé. Cela sera un fichier avec l'extension **crt**. Il faudra alors récupérer ce fichier, le renommer en ocpizza.crt et le copier dans le répertoire */etc/*ssl/ocpizza du serveur.

3.4 - Le conteneur Docker

Le conteneur Docker contient donc le serveur web et l'application OC-Pizza. Le fichier de configuration (Dockerfile) qui permet de le générer est dans les annexes. Les fichiers logs ainsi que le fichier de configuration du serveur web et les clés SSL n'ont pas été intégrés à l'image. En effet, en cas de souci sur le conteneur, les logs restent accessibles. C'est aussi le cas pour une modification de la configuration du serveur web. Pour ce qui est des clés SSL cela permet leur changement sans régénérer une image Docker.

Le lancement de l'image Docker se fait via Docker-compose qui apporte une plus grande souplesse d'utilisation ainsi que la possibilité de récupérer facilement la dernière version de

l'application OC-Pizza. Pour ce faire, pour toute nouvelle version, Pharos déposera une nouvelle image de l'application sur le site de Docker-Hub, la nuit suivante (04:00), le serveur ira de lui-même chercher la nouvelle version et la mettra en ligne.

Le script de mise à jour et la crontab sont en annexe. La procédure est traitée plus en avant dans ce document.

4 - Procédure de déploiement et mise à jour

4.1 - Arborescence de l'application

```
ocpizza
├── app_admin     (3 applis : app_admin – app_livraisons – app_commandes)
│   ├── admin.py
│   ├── apps.py
│   ├── forms.py
│   ├── __init__.py
│   ├── migrations
│   ├── models.py
│   ├── static                          → fichiers statiques de développement
│   │   └── app_admin
│   │       ├── css
│   │       │   └── style.css
│   │       ├── img
│   │       │   └── image.jpeg
│   │       └── js
│   │           └── script.js
│   ├── templates
│   │   └── app_admin
│   │       ├── home.html
│   │       └── mentions.html
│   ├── urls.py
│   └── views.py
├── manage.py
├── ocpizza
│   ├── asgi.py
│   ├── __init__.py
│   ├── settings.py          → fichier de configuration de développement
│   ├── prod_settings.py     → fichier de configuration de production
–  │   ├── urls.py
│   └── wsgi.py
├── README.md
├── requirements.txt         → fichier de configuration des paquets
├── staticfiles                    → Fichiers statiques de production
├── static                         → fichiers statiques de développement
│   └── img
│       ├── favicon.png
│       └── logo
├── templates
│   ├── errors
│   │   └── errors.html
│   └── layouts
│       ├── base.html
│       └── partials
```

```
├── _back.html
├── _footer.html
├── _nav.html
├── _paginator.html
└── _scripts.html
```

Cette arborescence est l'arborescence en début de développement et évoluera au fur et à mesure de l'avancée du projet.

4.2 - Déploiement ou mise à jour de l'application

La procédure de déploiement/mise à jour ci-dessous est valable pour la version 1.0 d'OC-Pizza.

Aucune manipulation n'est normalement nécessaire. Une mise à jour automatique du logiciel OC-Pizza est effectuée tous les jours à 04:00 via un script envoyé par crontab.

4.2.1 - Script de déploiement/mise à jour.

Le même script peut être utilisé pour le déploiement et les mises à jour.

- Dans un 1er temps, le script arrête l'application OC-Pizza et met à la place la page d'indisponibilité.
- Ensuite le script supprime toutes les versions de l'application OC-Pizza sur le disque du serveur, puis récupère la version de l'application OC-Pizza depuis le serveur hub.docker.com.
- Enfin le script retire la page d'indisponibilité puis redémarre l'application OC-Pizza.

Le script dure actuellement entre 3 et 4 minutes.

4.2.2 - Déploiement des fichiers

Comme dit ci-dessus : AUCUNE manipulation spéciale n'est nécessaire à la mise à jour normale de l'application. Une mise à jour quotidienne est effectuée. Cette mise à jour permet aussi de vider les éventuels caches ou autres qui se seraient créés durant la journée. En cas de nécessité (correction de bugs par exemple), pour lancer manuellement la mise à jour :
- Se connecter au serveur avec le compte **ocpizza** et taper la commande : ***sudo ./maj_ocpizza.sh***

Le programme de mise à jour dure environ 3 à 4 mn et se trouve en annexe.

ATTENTION : Le site OC-Pizza n'est plus accessible durant l'exécution du script de mise à jour

4.2.3 - Variables d'environnement

Voici les variables d'environnement pour la bonne exécution de l'application **OC-Pizza:**

Nom	Requis	Description et valeur	Conteneur/ serveur
DJANGO_SETTINGS_MODULE	Oui	Positionne le fichier settings.py de production **Valeur :** prod_settings.py	Conteneur

SECRET_KEY	Oui	Nécessaire à la sécurité des échanges client/serveur **Valeur :** une suite aléatoire de 50 caractères	Conteneur
MAILGUN_APIKEY	Oui	Utilisée pour envoi mails monitoring espace disque Clé appartenant à Pharos Consulting	Serveur

Ces variables sont initialisées dans le fichier /etc/environnement pour les variables « Serveur » et dans le fichier de génération du conteneur pour les autres.

4.2.4 - Vérifications

La commande sudo docker ps doit vous renvoyer une ligne similaire :

La commande suivante : ***sudo docker exec -ti ocpizza /usr/bin/python3 manage.py test*** va executer une série de tests qui doivent se passer sans erreur.

Enfin, vous devez aussi pouvoir vous connecter au site.

4.2.5 - Mise en indisponibilité et inverse manuellement

Il est possible de mettre le site OC-Pizza en indisponibilité (page prévue à cet effet).
Pour cela 2 scripts ont été réalisés.

Pour mettre en indisponibilité : Sous le compte **ocpizza**, taper la commande : ***sudo homeocpizza/docker_indispo.sh***

Pour remettre en disponibilité : Sous le compte **ocpizza**, taper la commande : ***sudo homeocpizza/docker_dispo.sh***

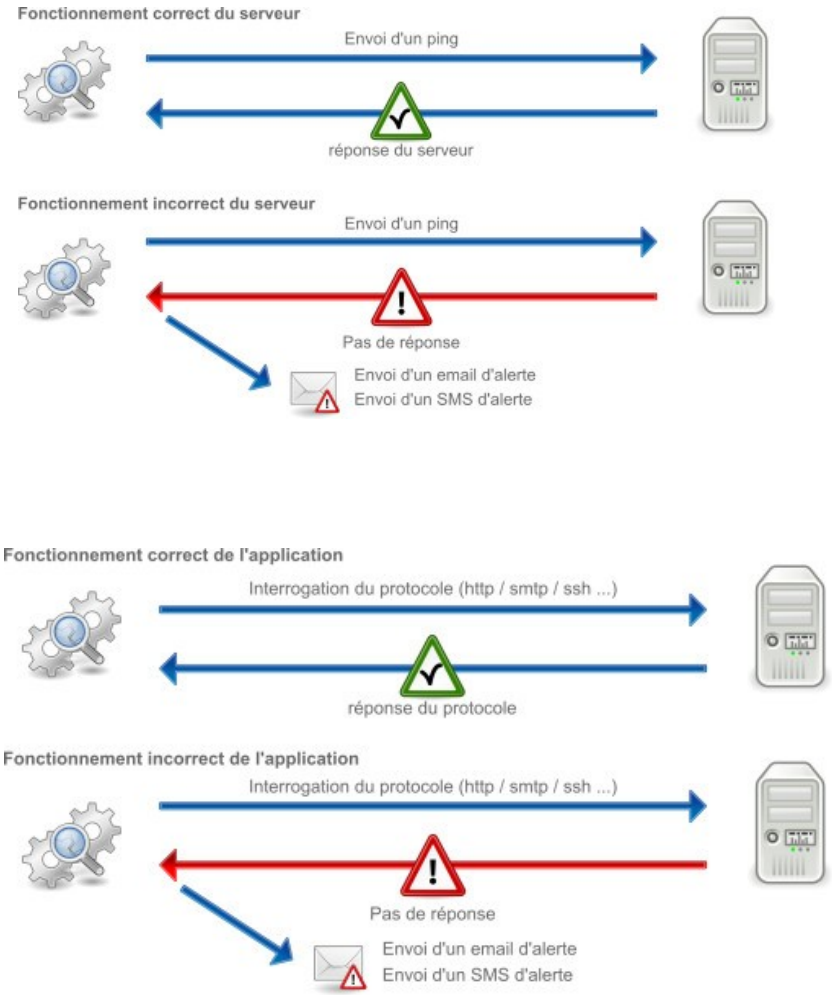
Le fait de lancer plusieurs fois l'une ou l'autre des commandes ne gêne en rien !

5 - *SUPERVISION*

5.1 - Supervision par la plate-forme OVH

La supervision de l'application se fait au travers du tableau de bord d'OVH. Ce tableau de bord dispose d'outils de monitoring qui scrutent certains services du serveur nécessaires à la bonne exécution de l'application. En cas de défaillance de l'un d'entre eux, un mail envoyé à l'administrateur du site.

Le schéma ci-dessous vous présente le mode de fonctionnement pour le protocole ICMP (ping) qui lui va plutôt informer sur le côté « up and down » du serveur, alors que le 2^{d} schéma montre la surveillance des applicatifs standards (site web par exemple).



La base de données étant une base de donnée sur le cloud d'OVH il n'y a pas de monitoring à effectuer.

5.2 - Monitoring espace disque

OVH ne gère pas les alertes concernant les espaces disques des serveurs VPS.
Une procédure a été créée pour cela. Elle se trouve dans le répertoire */root/* et se nomme check_disk_space.py. Cette procédure est lancée tous les matins à 07:00 par la crontab suivante :***0 7 * * * python3 /root/ check_disk_space.py***

Cette procédure fait partie de la matrice de serveur Pharos consulting. La clé API nécessaire est mise dans les variables d'environnement. Cette clé est la propriété de Pharos Consulting.

La procédure a été mise dans les annexes.

6 - Divers

6.1 - Sauvegarde et restauration

Le fait que la base de données soit sur l'espace Cloud d'OVH aucune procédure de sauvegarde/restauration spécifique n'a été prévue. En effet, la base est automatiquement sauvegardée par OVH quotidiennement. Toutefois, il est possible d'exécuter une sauvegarde ponctuelle en tapant la commande suivante : Toujours sous l'utilisateur **ocpizza → *pg_dump ocpizza_db -F c -f /var/tmp/ocpizz_db.dmp***

La restoration d'une base devant rester un événement exceptionnel et le risque de perte irrémédiable ou de corruption de données étant important, aucune procédure n'a été prévue. Nous vous conseillons fortement de nous contacter avant d'entamer une restoration quelconque.

6.2 - Procédure de démarrage / arrêt

L'arrêt du serveur n'est normalement jamais nécessaire. Toutefois, le tableau de bord OVH permet d'arrêter ou de redémarrer le serveur. Le redémarrage de l'application est automatique. La dernière ligne du fichier de configuration de Docker-compose (restart:always) permet de redémarrer le conteneur dès que le démon composer ne le trouve pas

Il est possible de mettre le site en indisponibilité en exécutant la commande suivante :
sudo /home/ocpizza/docker_indispo.sh

La remise en disponibilité se fait par la commande suivante :
sudo /home/ocpizza/docker_dispo.sh

Les 2 scripts se trouvent en annexe

6.3 - Fichiers logs

Les fichiers logs se trouvent dans le répertoire : /var/logs/docker/apache2 et se nomment access-ocpizza.log et error-ocpizza.log

7 - Annexes

7.1 - Fichier ocpizza.conf → Configuration apache

partie WSGI → Globale
WSGIDaemonProcess mondom.com python-home=/usr/local/venv python-path=/var/www/ocpizza home=/var/www/ocpizza

WSGIProcessGroup ocpizza.fr

WSGIScriptAlias / /var/www/ocpizza/ocpizza/wsgi.py process-group=ocpizza.fr

```
<VirtualHost *:80>                                          → partie en HTTP
        ServerName www.ocpizza.fr www.ocpizza.com www.oc-pizza.fr www.oc-pizza.com
        ServerAlias ocpizza.fr
        ServerAdmin webmaster@ocpizza.fr
        Redirect permanent / https://www.ocpizza.fr → redirection permanente vers le site en HTTPS
</VirtualHost>
<VirtualHost *:443>                                         → partie HTTPS
        SSLEngine  on
        SSLCertificateFile          /etc/ssl/ocpizza/ocpizza.crt
        SSLCertificateKeyFile       /etc/ssl/ocpizza/ocpizza.key
        ServerName www.ocpizza.fr www.ocpizza.com www.oc-pizza.fr www.oc-pizza.com
        ServerAlias ocpizza.fr
        ServerAdmin webmaster@modom.com
        Alias /static/ /var/www/ocpizza/staticfiles/
        <Directory /var/www/ocpizza/staticfiles>
                Require all granted
        </Directory>
        <Directory /var/www/ocpizza>
        Require all granted
        </Directory>
        <Directory /var/www/ocpizza/ocpizza>
        Require all granted
        </Directory>
        LogLevel warn
        ErrorLog    ${APACHE_LOG_DIR}/error-ocpizza.log
        CustomLog   ${APACHE_LOG_DIR}/access-ocpizza.log combined
</VirtualHost>
```

7.2 - Procedure de surveillance de l'espace disque du serveur.

```
import psutil
import requests
import os

def send_simple_message(mountpoint,percent):
        text = f"Attention ! l'espace disque {mountpoint} du serveur web OC-Pizza est plein à {percent}% !"
        api_key = os.getenv("MAILGUN_APIKEY")
        return requests.post(
                "https://api.mailgun.net/v3/sandbox1f42285ff9e446fa9e90d34287cd8fee.mailgun.org/messages",
                auth=("api", api_key),
                data={"from": "Serveur OC_PIZZA <webmaster@oc-pizza.fr>",
                        "to": ["webmaster@oc-pizza.fr"],
                        "subject": "Espace disk server OC-Pizza",
```

```
                "text": text})

disks = psutil.disk_partitions()
for disk in disks:
        freespace = psutil.disk_usage(disk.mountpoint).percent
        if freespace > 80:
                send_simple_message(disk.mountpoint,freespace)
```

7.3 - Scripts mise à dispo-indispo du site

```
#! /bin/bash
echo "docker-compose stop"
docker-compose stop
echo "lancement site d attente"
docker run -dit --name ocp -p 80:80 -v /home/ocpizza/siteattente/:/usr/local/apache2/htdocs/ httpd:2.4
```

```
#! /bin/bash
echo "arret site d attente"
docker stop ocp
echo "docker-compose up -d"
docker-compose up -d
```

7.4 - Configurations Docker

Dockerfile → Création du conteneur

```
FROM ubuntu:20.04

ENV TZ=Europe/Paris
ENV SECRET_KEY ")m0s)r&_gl3mkc^6-*5i44-2q@cf_^^^$&&(r6th4o@)7euy=s"
ENV DJANGO_SETTINGS_MODULE "prod_settings.py"

RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
RUN apt-get update && apt-get -y install apache2 libapache2-mod-wsgi-py3 python3 python3-pip postgresql-client
RUN mkdir /var/www/ocpizza
COPY ./website/ocpizza-Docker/ /var/www/ocpizza/
WORKDIR /var/www/ocpizza
RUN a2enmod ssl
RUN pip3 install --upgrade pip &&  pip3 install -r /var/www/ocpizza/requirements.txt
RUN python3 manage.py collectstatic && python3 manage.py makemigrations && python3 manage.py migrate

EXPOSE 80 443

CMD ["apache2ctl","-D","FOREGROUND"]
```

Docker-compose.yml → Execution du conteneur

```
version: "3"
services:
  ocpizza_service:
    image: jmlm74/ocpizza:latest
    ports:
      - "80:80" # host:container
      - "443:443"
    volumes:
```

```
  - "/var/log/docker:/var/log/apache2"                    # logfiles
  - "/home/ocpizza/ssl:/etc/ssl"                          # ssl cert
  - "/home/ocpizza/conf:/etc/apache2/sites-enabled"       # apache conf
hostname: "www"
domainname: "ocpizza.com"
container_name: "ocpizza"
restart: always
```

7.5 - Crontab

La crontab est celle de l'utilisateur root

8 - Glossaire

WSGI	La Web Server Gateway Interface (WSGI) est une spécification qui définit une interface entre des serveurs et des applications web pour le langage python.
Virtualhost	Dispositif permettant à un serveur web d'héberger plusieurs sites web
SSL/TLS	Protocoles de sécurisation des échanges en réseau et notamment par le réseau Internet. Permet de crypter les échanges entre 2 machines.
Crontab	Outil Unix permettant de lancer des taches à horaire fixe
Docker	Logiciel permettant de lancer des applications dans des conteneurs logiciels.