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.

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 le module mod_wsgi intégré et chargé.
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.

```
### 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>
```

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 user_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.

En tant que **user_ocpizza** :

Vérifier la bonne installation de paquets suivant : *python3-pip – libapache2-mod-wsgi-py3 – postgresql-common-client – git*
En passant la commande suivante : **sudo *dpkg s « nom_du_paquet »***. Le status doit etre installed. Si ce n'est pas les cas passer la commande ; ***sudo apt-get install nom_du_paquet.***

Vérifier que python-virtualenv soit bien installé en passant la commande suivante : ***pip check virtualenv.*** Si rien ne s'affiche, installer virtualenv : ***pip install virtualenv***

Vérifier que le fichier ocpizza.conf se trouve bien dans le répertoire */etc/apache2/sites_available* en passant la commande : ***ls -l /etc/apache2/sites_available.*** S'il n'est pas présent le récupérer en tapant la commande suivante : ***sudo wget*** https://mega.nz/file/NZ5gkCwI#3V1senLiq-x3KquYvFrXZHVXSV_PyRNYENZ8eYrkd30 ***-O /etc/apache2/sites_available/ocpizza.conf***

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.

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. Pour d'éventuelles versions futures, aller sur github pour voir dans le fichier README.md si des modifications à la procédure de déploiement/mise à jour ont été effectuées.

Toutes les manipulations doivent se faire sous le compte **user-ocpizza**

4.2.1 - Mettre le site en mode maintenance(mise à jour)

Avant toute mise à jour, il faut mettre le site en mode maintenance. Pour cela, passer les commandes suivantes :

- **sudo *a2dissite ocpizza***
- ***sudo a2ensite maintenance***
- ***sudo /etc/init.d/apache2 restart***

4.2.2 - Déploiement des fichiers (déploiement et mise à jour)

Le déploiement est effectué par **git** :

Avant tout, il faut :
- Activer l'environnement virtuel de l'application en tapant la commande : ***source /usr/local/venv/bin/activate***.

Se mettre dans le répertoire */var*/www/ocpizza (le créer au besoin si déploiement) et taper la commande : g*it clone* https://github.com/pharos_consulting/oc_pizza.git → Le code utilisateur/password vous sera fourni à ce moment.

Rester dans le répertoire racine de l'application et installer les modules requis : ***pip install -r requirements.txt***.Il se peut qu'aucun nouveau module ne soit installé.

Toujours dans le répertoire racine, taper la commande *:* ***python manage.py collectstatic*.** Cela va mettre tous les fichiers statiques des répertoires de développement dans le répertoire de production.

Ensuite, il faut passer les migrations (mises à jour de la structure de la base de données). Pour cela, il faut passer les commandes suivantes :

- ***python manage.py makemigrations***
- ***python manage.py migrate***

Il se peut qu'il n'y ait aucune migration mais cela ne coûte rien de passer ces 2 commandes…

Enfin, il faut passer un script de mise à jour des données ou autre en tapant la commande :

- ***python manage.py update_ocpizza***

4.2.3 - Création d'un superuser (déploiement)

Passer la commande suivante : ***python manage.py create superuser.*** L'utilisateur/mot de passe entrés seront à reporter dans le PV de livraison.

4.2.4 - Variables d'environnement (déploiement et mise à jour)

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

Nom	Requis	Description et valeur
DJANGO_SETTINGS_MODULE	Oui	Positionne le fichier settings.py de production **Valeur :** prod_settings.py
SECRET_KEY	Oui	Nécessaire à la sécurité des échanges client/serveur **Valeur :** une suite aléatoire de 50 caractères
MAILGUN_APIKEY	Oui	Utilisée pour envoi mails monitoring espace disque Clé appartenant à Pharos Consulting

Ces variables sont initialisées dans le fichier /etc/environnement à la livraison du serveur.

4.2.5 - Vérifications (déploiement et mise à jour)

Pour vérifier la nouvelle version, Il est possible d'utiliser les tests unitaires déjà préparés. Pour cela, toujours dans le répertoire racine de l'application, taper la commande suivante :

- ***python manage.py test***

Il ne doit pas y avoir d'erreur.

4.2.6 - Remettre le site en mode production(déploiement et mise à jour)

Pour cela, passer les commandes suivantes :

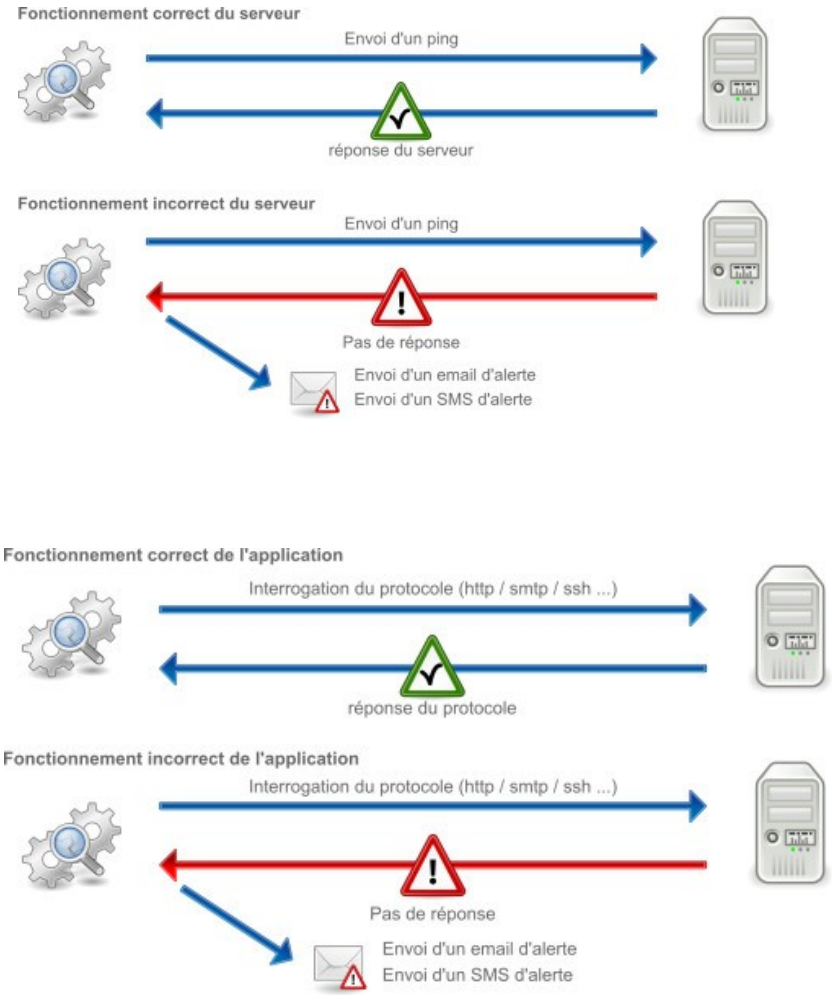
- ***a2dissite maintenance***
- ***a2ensite ocpizza***
- ***/etc/init.d/apache2 restart***

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.

```
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)
```

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 **user_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.

Vous pouvez à tout moment mettre l'application en indisponibilité et la remettre disponible (cf mise à jour).

6.3 - Fichiers logs

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

7 - 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 unxi permettant de lancer des taches à horaire fixe