OC Pizza

Projet 9

Dossier d'exploitation

Version 1.0

Auteur

Yohan Solon

Développeur

TABLE DES MATIERES

1 - VERSIONS

Auteur	Date	Description	Version
Yohan solon	04/05/2018	Création du document	1.0

2 - Introduction

2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application Oc Pizza

Objectif du document fournir à l'équipe technique de OC Pizza une documentation de prise en main de l'application.

2.2 - Références

Pour de plus amples informations, se référer :

1. **DCT - projet 9** : Dossier de conception technique de l'application

3 - PRE-REQUIS

3.1 - Système

3.1.1 - Serveur de Base de données

Serveur de base de données hébergeant le/les schémas/base Postgresql

3.1.1.1 - Caractéristiques techniques

PostgreSQL version	9.6.7

3.1.2 - Serveur Web

Serveur physique ou virtuel hébergeant l'application web.

Serveur dédié hébergé par kimsuffi

Sytème d'exploitation Debian GNU/Linux 9 (stretch)

nginx version: nginx/1.10.3

4 - PROCEDURE DE DEPLOIEMENT

4.1 - Déploiement de l'Application Web

4.1.1 - Environnement de l'application web

4.1.1.1 - Variables d'environnement

Le serveur d'application doit être exécuté avec la variable d'environnement suivante définie au démarrage. Elle est nécessaire afin de récupérer le répertoire contenant les fichiers de configuration de l'application :

```
DJANGO_SETTINGS_MODULE="master.settings.production"
```

INFO : il ne faut pas mettre de « / » à la fin de la valeur de la variable et ne pas utiliser d'espace dans le chemin.

4.1.2 - Répertoire de configuration applicatif

Le répertoire de configuration applicatif doit être créé sur le système de fichier et définit de la façon suivante :

```
$home_application/master/settings
```

Le fichier de configuration de production :

production.py

4.1.2.1 - Fichier production.py

```
from master.settings import *

# Database
# https://docs.djangoproject.com/en/2.0/ref/settings/#databases
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': <database name>, # Nom de la base de données
        'USER': <database username>, # Nom de l'utilisateur de la base de données
        'PASSWORD': <database password>, # Mot de passe de l'utilisateur de la base de
données$ path_app/manage.py migrate --settings=settings.production


        'HOST': 'localhost',
        #'PORT': '5432',
        #'OPTIONS': {'sslmode': 'require'},
    }
}
#S2dgzxeQG4PMUKQN

DEBUG = False

TEMPLATE_DEBUG = False # debug template deactivate


SECRET_KEY = get_env_variable('SECRET_KEY',<générer une clef secret>)

ALLOWED_HOSTS = ['localhost',<nom.de.domaine>] # authorized hostname
```

4.1.3 - DataSources

Les accès aux bases de données doivent se configurer à l'aide des fichiers production.py

Le fichier de drivers **postgresql (postgresql-9.6.x.)** doit être installer grâce à la commande :

```
sudo apt-get install postgresql
```

4.1.4 - Vérifications

Afin de vérifier le bon déploiement de l'application, faire ceci

$ path_app/env/bin/gunicorn master.wsgi:application

La page correspondant au site sera accessible depuis :

http://ip.serveur:8000

Une fois la vérification réalisée faite Ctrl+c pour quitter le serveur gunicorn

5 - PROCEDURE DE DEMARRAGE / ARRET

5.1 - Base de données

Démarrage du serveur postgresql

$ /etc/init.d/postgresql start

Arrêt du serveur postgresql

$ /etc/init.d/postgresql stop

5.2 - Application web

Démarrage de l'application web

$ path_app/env/bin/gunicorn master.wsgi:application

Arrêt de l'application web

Ctrl+c

6 - PROCEDURE DE MISE A JOUR

6.1 - Base de données

Mise à jour de l'architecture de la base de données depuis les migrations.

$ source path_app/bin/env/bin/activate

$ path_app/manage.py makemigrations --settings=settings.production

$ path_app/manage.py migrate --settings=settings.production

Commande pour le chargement de la mise à jour des données de la base.

$ path_app/manage.py loaddata path/to/dump/file/dump.json --settings=settings.production

Attention pensée à faire une sauvegarde de la base de données avant tout changement.

6.2 - Application web

Depuis la racine de l'application

$ git pull

Restart gunicorn

7 - SUPERVISION/MONITORING

7.1 - Supervision de l'application web

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

Se rendre dans /etc/supervisor/conf.d/ et créer un fichier ocpizza-app-gunicorn.conf.

$ cd /etc/supervisor/conf.d/

$ nano ocpizza-app-gunicorn.conf

Copier le code suivant :

[program:pb-app-gunicorn]

command = /<application path>/env/bin/gunicorn -b 127.0.0.1:8002 master.wsgi:application

user = pb

directory = <application path>

autostart = true

autorestart = true

environment =

DJANGO_SETTINGS_MODULE="master.settings.production"

Puis ctrl+x valider les modifications Y|O entrer

$ supervisorctl reread

$ supervisorctl update

$ supervisorctl status

7.2 - Monitering

Dans l'objectif de suivre l'évolution du projet et de parrer à toute erreur due au code il sera conseiller de metre en place et configure sentry , un outil de monitoring du code.

Sentry permet de détecter les erreurs des pages et d'avertir l'équipe technique par mail d'une erreur lorsque celle-ci est détectée sur une page.

8 - PROCEDURE DE SAUVEGARDE ET RESTAURATION

Afin de prévenir de tout dysfonctionnement de l'application il est recommandé de mettre en place une sauvegarde journalière avec une conservation sur 3 jours et une sauvegarde mensuelle.

La commande pour réaliser un dump de la base de données depuis django est la suivante :

$ path_app/manage.py dumpdata path/to/dump/file/db.json --settings=settings.production

Dans le cas d'une restauration de sauvegarde la commande est la suivante :

$ path_app/manage.py loaddata path/to/dump/file/db.json --settings=settings.production