

# DOSSIER D'EXPLOITATION



## **Projet 9** **« Documentez votre système de gestion de pizzeria »**

(Version 1.0)

**Auteur**  
LEKISHVILI Rati  
Analyste-programmeur

## Table des matières

<b>1.Versions</b>	<b>3</b>
<b>2.Introduction</b>	<b>4</b>
2.1.Objet du document	4
2.2.Références	4
<b>3.Pré-requis</b>	<b>5</b>
3.1.Système	5
3.1.1.Serveur de Base de données	5
3.1.1.1.Caractéristiques techniques	5
3.1.2.Serveur Web	5
3.1.2.1.Caractéristiques techniques	5
3.1.3.Serveur de Fichiers	5
3.1.3.1.Caractéristiques techniques	5
3.2.Web-services	5
<b>4.Procédure de déploiement</b>	<b>6</b>
4.1.Déploiement de l'application web	6
4.1.1.Déploiement sur Digital Ocean	6
4.1.2.Environnement de l'application web	7
4.1.3.Fichiers de configuration	7
4.1.4.Vérifications	7
<b>5.Procédure de démarrage / arrêt</b>	<b>8</b>
5.1.Base de données PostgreSQL	8
5.2.Application web directement depuis Digital Ocean	8
5.3.Serveur web Nginx	8
5.4.Serveur d'application Gunicorn via Supervisor	9
<b>6.Procédure de mise à jour</b>	<b>10</b>
6.1.Base de données	10
6.2.Application web	10
<b>7.Supervision/Monitoring</b>	<b>11</b>
7.1.Supervision de l'application web	11
7.2.Supervision du serveur (Droplet)	11
<b>8.Procédure de sauvegarde et restauration</b>	<b>12</b>
<b>9.Glossaire</b>	<b>13</b>

# 1. VERSIONS

Auteur	Date	Description	Version
LEKISHVILI Rati	25/08/2021	Création du document	1.0

## 2.INTRODUCTION

### 2.1.Objet du document

Le présent document constitue le dossier d'exploitation de l'application du groupe OC PIZZA.

Objectif du document est de fournir à l'équipe d'exploitation les informations nécessaires pour assurer une exploitation en règle du système et pouvoir réagir de manière appropriée si un problème surgit.

### 2.2.Références

Pour de plus amples informations, se référer au :

1. **Dossier de Spécifications Fonctionnelles - 1.0** : Dossier de spécifications fonctionnelles de l'application.
2. **Dossier de Spécifications Techniques - 1.0** : Dossier de spécifications techniques de l'application.
3. **Procès Verbal de livraison - 1.0**

## 3. PRÉ-REQUIS

### 3.1. Système

L'application sera hébergée sur un **Droplet** réservé sur le serveur **Digital Ocean**.

Ce dernier met à disposition un serveur Linux (Ubuntu 20.04) pour le déploiement de l'application.

Pour une question de sécurité, en plus de l'utilisateur « **root** » (par défaut), un nouvel utilisateur avec des privilèges normaux devra être créé, afin de gérer le déploiement de l'application.

Un Firewall devra être configuré afin d'autoriser les connexions en **ssh** en plus des connexions en **http** au Droplet.

#### 3.1.1. Serveur de Base de données

La base de données sera nommée **oc\_pizza** et hébergée sur le serveur lié à Digital Ocean.

##### 3.1.1.1. Caractéristiques techniques

On utilisera le SGBDR **PostgreSQL** : 13.4 dernière version.

#### 3.1.2. Serveur Web

**NGINX** sera configuré et utilisé comme serveur web pour notre application. Ce dernier servira les fichiers statiques et dirigera le trafic vers l'application Django.

##### 3.1.2.1. Caractéristiques techniques

Version recommandée : 1.20.1 à ce jour.

#### 3.1.3. Serveur de Fichiers

**GUNICORN** est un serveur http Python pour Unix et utilise les spécifications WSGI afin de permettre la communication entre le serveur web et l'application Django. Il sera configuré et utilisé autant que serveur d'application.

##### 3.1.3.1. Caractéristiques techniques

Commande pour lancer le serveur de production Unicorn : `unicorn oc_pizza.wsgi:application`

Version recommandée : 20.1.0 à ce jour.

### 3.2. Web-services

-> Paiements en ligne : <https://stripe.com>

## 4. PROCÉDURE DE DÉPLOIEMENT

### 4.1. Déploiement de l'application web

#### 4.1.1. Déploiement sur Digital Ocean

1) Se connecter au Droplet avec ssh en ligne de commande :

-> \$ ssh utilisateur@adresse\_IP\_du\_droplet

2) Cloner le projet depuis GitHub :

-> (depuis le répertoire du projet) \$ git clone + lien du projet sur GitHub.

3) Installer un environnement virtuel et l'activer :

-> \$ pip install virtualenv

-> \$ source env/bin/activate

4) Peupler la base de données :

A. Effectuer les migrations :

-> \$ Python manage.py migrate

B. Pour remplir la base de données en utilisant l'api d'Open Food Facts, utiliser la commande django personnalisée créée :

-> \$ python manage.py commands

\* « **commands** » est le nom de la commande personnalisée qui peuplera la base de données avec les informations renvoyées par Open Food Facts.

5) Installer et configurer Nginx :

-> \$ pip install nginx

5) Installer Gunicorn :

-> \$ pip install gunicorn

6) Installer et configurer Supervisor :

-> \$ pip install supervisor

### 4.1.2. Environnement de l'application web

Voici les variables d'environnement de l'application en production :

Nom	Obligatoire	Description
ENV	Oui	Définit s'il s'agit de l'environnement de production
SECRET_KEY	Oui	Clé secrète de l'application
DJANGO_SETTING_MODULE	Oui	Contient le chemin vers le fichier de configuration de la base de données en production (production.py)

### 4.1.3. Fichiers de configuration

Voici les différents fichiers de configuration :

\* Nom du répertoire du projet : `oc_pizza`

- `oc_pizza/settings/__init__.py` : fichier de configuration de l'application.
- `oc_pizza/settings/production.py` : informations sur la base de données à utiliser en production (hérite de `settings/__init__.py`).
- `/etc/nginx/sites-enabled/oc_pizza` : fichier de configuration de Nginx.
- `oc_pizza/wsgi.py` : fichier de configuration pour lancer le serveur de production Gunicorn.
- `/etc/supervisor/conf.d/oc_pizza-gunicorn.conf` : fichier de configuration de Supervisor.
- `travis.yml` : fichier de configuration de Travis-CI.
- `travis.py` : modèle de base de données pour Travis-CI, afin qu'il puisse effectuer les tests dans son environnement.

### 4.1.4. Vérifications

Afin de vérifier le bon déploiement de l'application, vérifier le nom de domaine en production dans un navigateur quelconque : [oc\\_pizza.fr](http://oc_pizza.fr) (par exemple).

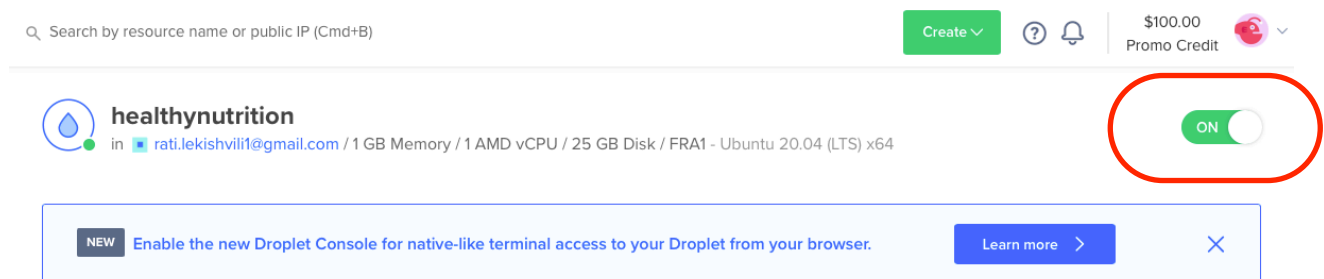
## 5. PROCÉDURE DE DÉMARRAGE / ARRÊT

### 5.1. Base de données PostgreSQL

- 1) Démarrer : **\$ sudo service postgresql start**
- 2) Vérifier le statut : **\$ sudo service postgresql status**
- 3) Arrêter : **\$ sudo service postgresql stop**
- 4) L'emplacement du fichier de configuration en cas de problème : **\$ sudo ps | grep postgres**

### 5.2. Application web directement depuis Digital Ocean

En se rendant sur le Droplet, directement accessible depuis l'interface utilisateur de Digital Ocean : <https://www.digitalocean.com>, il est possible d'arrêter ou de démarrer l'application à l'aide d'un bouton.



### 5.3. Serveur web Nginx

- 1) Démarrer : **\$ sudo service nginx start**
- 2) Vérifier le statut : **\$ sudo service nginx status**
- 3) Arrêter : **\$ sudo service nginx stop**
- 4) Redémarrer : **\$ sudo service nginx reload**



## 5.4. Serveur d'application Gunicorn via Supervisor

- 1) Démarrer : **\$ sudo supervisorctl start oc\_pizza-gunicorn**
- 2) Vérifier le statut : **\$ sudo supervisorctl status oc\_pizza-gunicorn**
- 3) Arrêter : **\$ sudo supervisorctl stop oc\_pizza-gunicorn**
- 4) Redémarrer :
  - 1) **\$ sudo supervisorctl reread oc\_pizza-gunicorn**
  - 2) **\$ sudo supervisorctl update oc\_pizza-gunicorn**

## 6. PROCÉDURE DE MISE À JOUR

### 6.1. Base de données

La mise à jour de la base de données s'effectue automatiquement une fois par semaine (tous les vendredis à 17h). Une tâche cron a été programmée pour exécuter la commande personnalisée django : **python manage.py commands**.

```
#  
# m h dom mon dow  command  
  
0 17 * * 5 /home/rati/P10_healthy_nutrition/env/bin/python3 /home/rati/P10_healthy_nutrition/manage.py commands >> /home/rati/P10_healthy_nutrition/cron.log  
#  
#
```

Si vous souhaitez lancer cette commande manuellement c'est aussi possible. Vous devez vous placer dans le répertoire du projet : « **oc\_pizza** » et exécuter cette même commande : **python manage.py commands**.

\* « **commands** » est le nom de la commande personnalisée qui peuplera la base de données avec les informations renvoyées par Open Food Facts.

### 6.2. Application web

Mettre à jour l'application web sans notre aval est une pratique que nous déconseillons, car elle peut se révéler invalide. Déterminer la cause sera certainement plus difficile pour notre équipe de développeur. Tout changement non compris dans le contrat initial peut occasionner des frais supplémentaires.

# 7.SUPERVISION/MONITORING

## 7.1.Supervision de l'application web

Afin de tester que l'application web est toujours fonctionnelle, Sentry sera chargé de la surveillance de cette dernière. Sentry est un tableau de bord qui permet de visualiser ce qui se passe dans l'application, c'est à dire d'afficher les logs. Mais, il fournit aussi des détails concernant le navigateur de l'utilisateur, l'url ...

Un compte sera créé (ainsi que des alertes), afin de surveiller l'application en production. Nous serons averti par mail sur d'éventuels erreurs survenues.

## 7.2.Supervision du serveur (Droplet)

Digital Ocean offre la possibilité de surveiller le bon fonctionnement du serveur, en créant des alertes depuis l'interface utilisateur, rubrique « Monitoring ».

Deux alertes seront créées :





- 1) La première surveillera que le processeur (CPU) ne soit pas surchargé à plus de 70%. Les vérifications seront répétées toutes les 30 minutes.
- 2) La deuxième alerte surveillera que l'utilisation de la mémoire vive ne soit pas trop importante et ne dépasse pas 70%. Les vérifications seront répétées toutes les 30 minutes.

\* Comme sur exemple ci-dessous :

### Monitoring

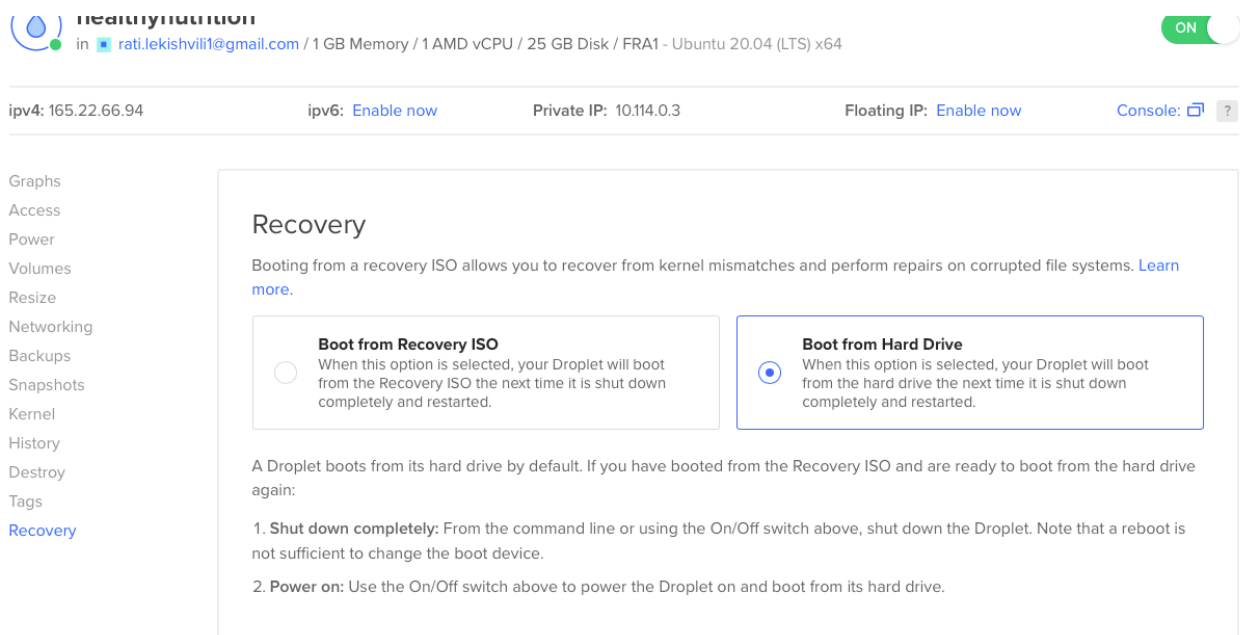
#### Alert Policies

[Setup instructions](#)
[Create alert policy](#)

Name	Applied to	
 <b>CPU is running high on healthy nutrition</b> CPU is above 70% for 30 min	 healthynutrition	<a href="#">More ▾</a>
 <b>Memory Utilization is running high on healthy nutrition</b> Memory Utilization is above 70% for 30 min	 healthynutrition	<a href="#">More ▾</a>

## 8. PROCÉDURE DE SAUVEGARDE ET RESTAURATION

Digital Ocean permet d'assurer la sauvegarde et la restauration des données à l'aide de : « Recovery ISO » et la « recovery console ».



The screenshot shows the DigitalOcean management console for a Droplet named 'healthynutrition'. The top bar includes the Droplet name, email 'rati.lekishvili1@gmail.com', and specifications: 1 GB Memory / 1 AMD vCPU / 25 GB Disk / FRA1 - Ubuntu 20.04 (LTS) x64. A green 'ON' switch is visible. Below the bar, network settings are listed: ipv4: 165.22.66.94, ipv6: Enable now, Private IP: 10.114.0.3, Floating IP: Enable now, and Console: [icon] [?]. A left sidebar contains navigation links: Graphs, Access, Power, Volumes, Resize, Networking, Backups, Snapshots, Kernel, History, Destroy, Tags, and Recovery (highlighted in blue). The main content area is titled 'Recovery' and explains that booting from a recovery ISO allows recovery from kernel mismatches and repairs on corrupted file systems. It offers two options: 'Boot from Recovery ISO' (unselected) and 'Boot from Hard Drive' (selected with a blue dot). Below these options, instructions are provided: 'A Droplet boots from its hard drive by default. If you have booted from the Recovery ISO and are ready to boot from the hard drive again: 1. Shut down completely: From the command line or using the On/Off switch above, shut down the Droplet. Note that a reboot is not sufficient to change the boot device. 2. Power on: Use the On/Off switch above to power the Droplet on and boot from its hard drive.'

Chaque cas étant particulier, toutes les étapes sont détaillées dans la documentation fournie par Digital Ocean : <https://docs.digitalocean.com/products/droplets/resources/recovery-iso/>

## 9. GLOSSAIRE

<b>Droplet</b>	Désigne un espace serveur dans le jargon de Digital Ocean.
<b>Firewall</b>	Un pare-feu gérant l'accès au Droplet.
<b>SGBDR</b>	Système de Gestion de Base de Données Relationnelles.
<b>SSH</b>	Protocole de communication sécurisée.
<b>Travis-CI</b>	Service d'intégration continue.