OC Pizza

OCPizzApp

Dossier d'exploitation

Version 1.0

Auteur
Martin Gaucher
Analyste développeur

TABLE DES MATIÈRES

1 - Versions

Auteur	Date	Description	Version
Martin Gaucher	18/02/2020	Création du document	1.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application OCPizza. L'objectif de ce document est de détailler ce qu'il faut faire pour que l'application soit installée correctement et sereinement.

2.2 - Références

Pour de plus amples informations, se référer:

1. **MG - DCT – 1.1** : Dossier de conception technique de l'application
2. **MG – FO - 1.1 :** Dossier de conception fonctionnel de l'application

3 - Pré-requis

3.1 - Système

3.1.1 - Serveur Web

La mise en place du serveur web sera assurée par gunicorn.

3.1.2 - Serveur de Batches

Il y a un serveur de batch présent sur Héroku. On peut y lancer des commandes pour pouvoir ajouter, supprimer ou bien modifier des fichiers.

3.2 - Bases de données

Le serveur de base de données sera hébergé sur Heroku. On utilisera la dernière version de PSQL(12.2).

3.3 - Web-services

L'API (Application Program Interface) Google Maps devra être fonctionnelle. Pour s'assurer de sa bonne marche, il faudra vérifier que ces éléments soient bien configurés:

- avoir un accès à la Google Search Console en ayant entré ses données bancaires (sinon on ne nous donne pas accès à l'application).
- possèder une clé API pour faire le lien entre notre compte Google et l'application.
- veiller à ne pas dépasser le nombre de requêtes auquel nous avons souscrit au risque que l'application ne fonctionne plus.

4 - Procédure de déploiement

4.1 - Configuration des fichiers

4.1.1 - Artefacts

Les batches de l'application OCPizza sont construits sous la forme d'une archive ZIP contenant les répertoires :

- **ocpizzapp**: Contient tous les fichiers liés à la configuration de l'application.
- **ventes**: Ce dossier contient l'application qui gère les ventes.
- **production**: Ce dossier contient l'application qui gére la production.
- **templates**: Les templates utilisés seront stockés dans ce dossier.
- **static**: Ce dossier contient les assets (js, css, …)
- **requirement.txt**: Ce fichier contient toutes les dépendances pythons nécessaires au projet.
- **Procfile**: Fichier de configuration nécessaire pour faire tourner le serveur web gunicorn

4.1.2 - Variables d'environnement

Voici les variables d'environnement reconnues par les batches de l'application OCPizzApp:

Nom	Obligatoire	Description
ENV_MODE	Oui	Précise si l'application est en mode Production ou Développement
GMAP_API_KEY	Oui	Stock la clé API nécessaire au bon fonctionnement de GMAP
SECRET_KEY	Oui	Clé secrète utiliser par Django pour rendre l'application unique.

Pour définir les variables d'environnement, il y a deux manières:

La première est de passer par l'interface d'Heroku dans les configurations de l'application.

La seconde manière est de les définir par le biais d'Heroku CLI.

4.1.3 - Configuration

Voici les différents fichiers de configuration:

4.1.3.1 - Fichier requirements.txt

Ce fichier contient toutes les dépendances python nécessaires pour faire fonctionner le projet.

4.1.3.2 - Fichier Procfile

Ce fichier est à configurer afin de pouvoir faire fonctionner le serveur web (gunicorn).

4.1.3.3 - Fichier settings.py

C’est le fichier de configuration de l’application Django.

4.1.4 - Vérifications

Afin de vérifier le bon déploiement des batches, on pourrait imaginer une commande Django personnalisée qui testerait que tout est bien en place.

4.2 - Déploiement de l'Application Web

4.2.1 - Déploiement de la base de donnée

Pour que la base de donnée soit bien à jour, il faut effectuer la migration Django associée. Pour se faire, il suffit d'effectuer cette commande lorsque le serveur est en marche pour vérifier les migrations en cours:

```
heroku run ./manage.py makemigrations
```

```
heroku run ./manage.py migrate
```

Si la migration s'est bien déroulée, un message est affiché. Si l'inverse se produit il faudra contacter le développeur en charge de l'installation de la base de données.

4.2.2 - Application web

Pour mettre en place l'application web, il faudra bien s'assurer que la liaison entre notre projet local et le serveur d'Heroku est bien assurée. Pour cela il faut vérifier si git est bien relié entre les deux instances. Une fois que cette vérification est faite et que Git est fonctionnel, il faudra exécuter la commande suivante pour envoyer tout le code sur le serveur:

```
git push heroku master
```

4.2.3 - Serveur web

Le serveur web devra être configuré via le fichier de configuration appelé Procfile. Sans ce dernier le serveur web ne pourra pas tourner. Il doit contenir cette ligne:

```
web:gunicorn ocpizza.run:app
```

Il faut aussi que gunicorn soit présent dans les requirements.txt afin qu'il soit installé et que tout marche pour le mieux.

5 - Procédure de démarrage / arrêt

5.1 - Démarrage du serveur

Après avoir suivi les étapes précédentes de la configuration des fichiers de l'application, il faudra effectuer cette commande dans heroku CLI:

```
heroku ps:scale web=1
```

5.2 - Arrêt du serveur

Pour que l'application s'arrête, il suffit d'effectuer cette commande:

```
heroku ps:scale web=0
```

6 - Procédure de sauvegarde et restauration

6.1 - Base de données

6.1.1 - Importer une base de données

Nous devons d'abord compresser la base de donnée dans le bon format. Pour se faire, il faut utiliser la commande pg_dump. Voici la commande à effectuer pour compresser et récupérer la base local au bon format:

```
PGPASSWORD=<mot-de-passe> pg_dump -Fc –no-acl h localhost U <utilisateur> <nom-de-la-base> > <nom-de-la-base>.dump
```

Ensuite nous devons l'importer dans Heroku. Il faut d'abord créer une URL signée comme suit:

```
aws s3 presign s3://<mon-adresse>/<mon-objet>
```

Puis importer la base de donnée dans Heroku:

```
heroku pg:backups:restore '<mon-url-signé>' <URL-BASE-DE-DONNÉE>
```

6.1.2 - Effectuer une récupération de la base de donnée

Pour effectuer une sauvegarde de la base, on peut lancer ces deux commandes sur Heroku:

```
heroku pg:backups:capture
```

Cette commande permet de figer et de sauvegarder la base à un instant T.

Puis pour la télécharger et la conserver, il faut effectuer la commande suivante:

```
heroku pg:backups:download
```

7 - SUPERVISION/MONITORING

7.1 - Supervision de l’application web

On peut surveiller que l’application fonctionne bien depuis Heroku. L’application propose un service “Metrics” qui est payant. Depuis le tableau de bord d’Heroku on pourra surveiller les charges du serveur et le trafic.

Si une erreur ou un dysfonctionnement surviennent, on peut vérifier les logs du serveur avec la commande suivante (depuis heroku CLI):

```
heroku logs
```

8 - MAINTENANCE

8.1 - Mettre l’application en mode maintenance

Pour mettre l’application en mode maintenance il faut effectuer la commande suivante:

```
heroku maintenance:on
```

8.2 - Remettre l’application en ligne

Pour remettre l’application en ligne après une maintenance tapez la commande suivante:

```
heroku maintenance:off
```

8.3 - Vérifier si l’application est en maintenance

Pour vérifier si l’application est en mode maintenance ou non, il suffit de taper la commande suivante:

```
heroku maintenance
```

Cela affichera en console si l’application est en maintenance (on) ou non (off).