OC PIZZA

PIZZAFRAPP

Dossier d'exploitation

Version 1.0

Auteur
Stanimir Dimitrov

OC PIZZA

SOLUTIONS

TABLE DES MATIERES

1 - VERSIONS

Auteur	Date	Description	Version
SD	15/09/2019	Création du document	1.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application PIZZFRAPP. L'objectif du document est de fournir à l'équipe technique de la société OC PIZZA les informations essentielles pour une bonne utilisation de l'application, ainsi que les instructions à suivre pour le déploiement et la maintenance de celle-ci.

2.2 - Références

Pour de plus amples informations, se référer :

1. P9 - DCT – 1.1: Dossier de conception technique de l'application
2. P9 - DCF – 1.1 : Dossier de conception fonctionnelle de l'application.

3 - PRÉ-REQUIS

3.1 - Système

L'application web PIZZFRAPP est hébergé sur la plateforme Heroku .
L'application est liée au nom de domaine : pizzfrapp.fr

3.2 - Bases de données

Le SGBD utilisé par l'application est PostgreSQL dans sa version 11.5. La base de données est hébergée sur les serveurs de Heroku.

3.3 - Web-services

Le fonctionnement de l'application nécessite que les web-services suivants soit opérationnels : □ Google Maps API : L'utilisation de cette API nécessite une clé d'identification. La clé utilisée par PIZZFRAPP est la suivante : ********. Le service étant payant étant donné le nombre de requêtes à cette API nécessaire pour le bon fonctionnement de PIZZFRAPP, il est nécessaire de veillez à ce que les règlements soient bien effectués sous peine de dysfonctionnement de l'application.

4 - PROCÉDURE DE DÉPLOIEMENT

4.1 - Déploiement de l'application web

4.1.1 - Composition de l'application web

L'application PIZZFRAPP est construite sous la forme d'une archive ZIP contenant les répertoires suivants :

- **PIZZFRAPP** : Contient les fichiers de configuration de l'application et de Django.
- **Procfile** : Contient les instructions pour démarrer l'application
- **requirements.txt** : contient les librairies nécessaires pour que l'application fonctionne.

- **Static :** Contient les fichiers .CSS et .JS de l'application.
- **Templates :** Contient les fichiers HTML de l'application.
- **Ventes :** Contient les fichiers liés au package Ventes de l'application.
- **Production :** Contient les fichiers liés au package Production de l'application.
- **Docs :** Contient la documentation de l'application.

Le fichier **Procfile** contient les instructions exécutées par Heroku pour démarrer l'application.
Le contenu de ce fichier doit être le suivant pour que l'application fonctionne :

```
web: gunicorn pizzfrapp.runapp
```

4.1.2 - Variables d'environnement

L'application nécessite la configuration des variables d'environnement suivantes :

Nom	Valeur	Obligatoire	Description
ENV	**PRODUCTION**	Oui	Indique à l'application qu'il faut utiliser la configuration de production et non de développement
SECRET_KEY	**Samo-Adra1924**	Oui	Nécessaire au démarrage de Django. *Cette clé est la clé de production et ne doit figurer nulle part ailleurs que sur le serveur de production.*

4.1.3 Configuration

Voici les différents fichiers de configuration :

- **procfile** : fichier contenant les commandes utilisées par Heroku pour le déploiement de l'application.
- **requirements.txt** : fichier contenant la liste des librairies à installer sur le serveur Heroku pour que l'application fonctionne correctement.
- **pizzafrpp/pizzfrapp/settings.py** : fichier de configuration l'application Django.

4.1.4 - Déploiement

Heroku propose différentes méthodes pour déployer une application. Nous conseillons cependant d'effectuer un déploiement en ligne de commande en utilisant Heroku CLI

Les étapes présentées ci-dessous nécessite que Heroku CLI soit installé, que le projet soit suivi avec GitHub et il faut être sur la branche Master du projet.

4.1.4.1 - Création de l'application :

Une fois Heroku cli installé la connexion au compte OC-PIZZA effectuée, il faut se rendre à la racine du projet de l'application et entrer la commande suivante :

```
$ heroku create pizzfrapp
```

4.1.4.2 - Configuration des variables d'environnement :

Les variables d'environnement se définissent via la ligne de commande de la façon suivante :

```
$ heroku config :set ENV='PRODUCTION'
$ heroku config :set SECRET_KEY= '**********'
```

Pour vérifier que les variables ont bien été configurées, utilisez la commande :

```
$ heroku config
```

4.1.4.3 - Envoyer l'application sur le serveur

Une fois les étapes précédentes réalisées, il suffit d'effectuer un Push sur le serveur :

```
$ git push heroku master
```

4.1.5 - Vérifications

Une fois le déploiement terminé, les lignes suivantes doivent apparaître dans le terminal :

```
remote :       https://pizzfrapp.fr/ deployed to Heroku
remote: Verifying deployement done.
```

Pour s'assurer du bon déploiement, il faut ensuite de se connecter à l'adresse www.pizzapp.fr afin de vérifier que l'application est bien en ligne.

4.2 - Déploiement de la base de données

Le déploiement de la base de données sur la plateforme Heroku nécessite que l'application ait été déployée comme présenté précédemment.

De la même manière que pour le déploiement de l'application, celui de la base de données est présenté ici en utilisant la ligne de commande et Heroku CLI.

4.2.1 - Lancement des migrations

Pour importer les données, il est nécessaire de créer et configurer les tables de la base de données via la commande suivante :

```
$ heroku run python manage.py migrate
```

4.2.2 - Création d'un superutilisateur

Une fois la base de données créée, il faut ajouter un super utilisateur via la commande :

```
$heroku run python manage.py createsuperuser
```

4.2.3 - Import des données

Il est possible d'importer les données de la version de développement de l'application sur le serveur de production. Pour cela, il faut créer un dump de la base de données de développement, puis l'envoyer l'importer sur le serveur d'heroku.

- Création d'un dump de la base de données :

```
$ ./manage.py dumpdata pizzfrapp > pizzfrapp/dumps/pizzfrapp.json
```

- Import des données dans la base :

```
$ heroku run python manage.py loaddata pizzapp/dumps/pizzfrapp.json
```

4.2.4 - Vérifications

La vérification de l'état de la base de données peut se faire directement via le site de Heroku dans la rubrique Ressources du projet.

Il faut ensuite se rendre dans la partie Add-ons puis cliquer sur Heroku Postgres : Database .

On retrouve dans cet espace les informations sur la base de données telles que : son état (active ou non), le nombre de ligne ou encore l'espace utilisé.

5-PROCEDURE DE DEMARRAGE / ARRET

La gestion de l'application peut se faire de deux manières, soit depuis le site de Heroku soit en ligne de commande après avoir téléchargé l'interface heroku-cli :

5.1 - Démarrage / arrêt sur le site heroku.com

Une fois déployée sur Heroku, l'application est par défaut en ligne.

Afin de désactiver l'application il faut se rendre dans la rubrique Settings, puis changer le statut Maintenance Mode à ON.

Il suffira ensuite de faire la démarche inverse afin de rétablir l'application.

5.2 - Démarrage / arrêt en ligne de commande

Afin de démarrer ou arrêter l'application via la ligne de commande, il faut utiliser les commandes suivantes :

Mise en marche de l'application :

```
$ heroku run
```

Arrêt de l'application :

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

6-PROCEDURE DE MISE A JOUR

La mise à jour de l'application nécessite de passer celle-ci en mode maintenance.

La mise en mode maintenance peut s'effectuer directement via le tableau de bord heroku soit directement en ligne de commande via la commande suivante :

```
$ heroku maintenance :on
```

Lorsque le mode maintenance est activé, un message s'affiche lors de la connexion à l'application et les utilisateurs n'ont plus accès à celle-ci.

Une fois la mise à jour de l'application effectuée, l'application peut être réactivée comme ceci :

```
$ heroku maintenance :off
```

7-SUPERVISION/MONITORING

La plateforme Heroku fournit des outils de monitoring dans la rubrique Metrics de l'application.

Plus d'informations sont disponibles dans la documentation de Heroku, à l'adresse suivante :
https://devcenter.heroku.com/categories/monitoring-metrics

8-PROCEDURE DE SAUVEGARDE ET RESTAURATION

Heroku propose ses fonctionnalités pour le processus de sauvegarde et restauration de l'application qui passe par sauvegarde de la Base des données.

8.1 - Sauvegarde de la base de données

```
$ heroku pg:backups:capture --app pizzfrapp
```

8.2 - Restauration de la base de données

```
$ heroku pg:backups:restore <nom sauvegarde> DATABASE_URL --app pizzfrapp
```

9-GLOSSAIRE

SGBD	Système de gestion de base de données
DJANGO	Framework permettant la réalisation d'application web en Python.