DOSSIER D'EXPLOITATION

OC PIZZA

Auteur
Elodie Meunier
Développeur

TABLE DES MATIÈRES

1. VERSIONS

Auteur	Date	Description	Version
Elodie	11/02/2021	Création du document	1

2. INTRODUCTION

2.1 OBJET DU DOCUMENT

Le présent document constitue le dossier d'exploitation de l'application web OC Pizza développée par l'entreprise EdenConception. Ce document vise à décrire les étapes afin de mettre en production l'application et l'a maintenir.

2.2 RÉFÉRENCES

Pour de plus amples informations, se référer aux éléments suivants :

1. DCT : Dossier de conception technique de l'application
2. DCF : Dossier de conception fonctionnelle de l'application

3. PRÉ-REQUIS

3.1 SYSTÈME

3.1.1 Serveur web

Le serveur web utilisé pour l'application est Gunicorn, un serveur HTTP Python très utilisé.

3.1.2 Hébergement

L'application web sera hébergée sur le service de déploiement d'application web Heroku. Le nom de domaine choisi par le client est ocpizza.com et sera défini dans les configurations d'Heroku.

3.2 BASES DE DONNÉES

Serveur de base de données

Le serveur de base de données utilisé sera PostgreSQL (12.5) fourni avec Heroku.

Le web-service utilisé pour l'application doit être accessible et à jour. Le voici :

- PayPal API : Service de paiement en ligne.

4. PROCÉDURE DE DÉPLOIEMENT

4.1 VARIABLES D'ENVIRONNEMENT

Découvrez ci-dessous les différentes variables d'environnement qu'il faudra configurer sur le service de déploiement, Heroku.

Variable	Valeur	Description
ENV	PROD	Permet de cibler la base de données de production ainsi que la bonne clé secrète pour le lancement de l'app.
SECRET_KEY	%_uznehy!@e-(g5%	Clé secrète de production. Permet l'exécution du serveur.
PAYPAL_KEY	%_uznehy!@e-(g5%pay	Clé api PayPal. Permet l'utilisation de l'API

Pour ajouter une variable d'environnement à la configuration d'Heroku :

```
1 $ heroku config:set ENV=PROD
```

Découvrez les étapes de configuration Heroku afin d'effectuer correctement le déploiement de l'application.

- ### 4.2.1 Création d'un fichier Procfile

  Vous devrez créer un fichier Procfile (sans extension) et y insérer le contenu suivant :

```
1 web gunicorn ocapp.wsgi
```

  Vous devrez donc installer dans les dépendances, le serveur web Gunicorn :

```
$ pip install gunicorn
```

- ### 4.2.2 Création d'une nouvelle application Heroku

  Si ce n'est déjà fait, il faudra créer une application Heroku. Pour ce faire, tapez la commande suivante :

```
$ heroku create app_name
```

- ### 4.2.3 Création d'une base de données

  Grâce à Heroku, nous pouvons créer aisément une base de données. N'oubliez pas d'importer psycopg2 :

```
$ heroku addons:add heroku-postgresql:hobby-dev
$ pip install psycopg2
```

- ### 4.2.4 Envoi des données sur Heroku

  Il ne vous reste plus qu'à pousser vos développements puisqu'Heroku se base sur Git :

```
$ git add -p
$ git commit -m "[ADD] - pushing to prod"
$ git push heroku master
```

5. PROCÉDURE DE DÉMARRAGE / ARRÊT

Pour arrêter votre application, il suffit de réduire les dynos web à 0 ce qui mettra tout votre processus http hors ligne.

```
$ heroku ps:scale web=0
Scaling web processes… done, now running 0
```

Et pour redémarrer l'application, il vous suffira de remettre les dynos web à 1 :

```
$ heroku ps:scale web=1
Scaling web processes… done, now running 0
```

6. PROCÉDURE DE MISE À JOUR

6.1 APPLICATION WEB

Heroku propose dans son interface d'administration la possibilité de mettre votre application en maintenance. Ceci vous permettra d'effectuer toutes les mises à jours que vous le souhaitez avant de désactiver le mode « Maintenance ».

6.2 BASE DE DONNÉES

Avant de mettre une base de données en maintenance, il est conseillé de mettre en maintenance l'application lié (cf. 5.1).
Voici comment lancer et stopper une maintenance manuellement avec Heroku CLI ci-dessous :

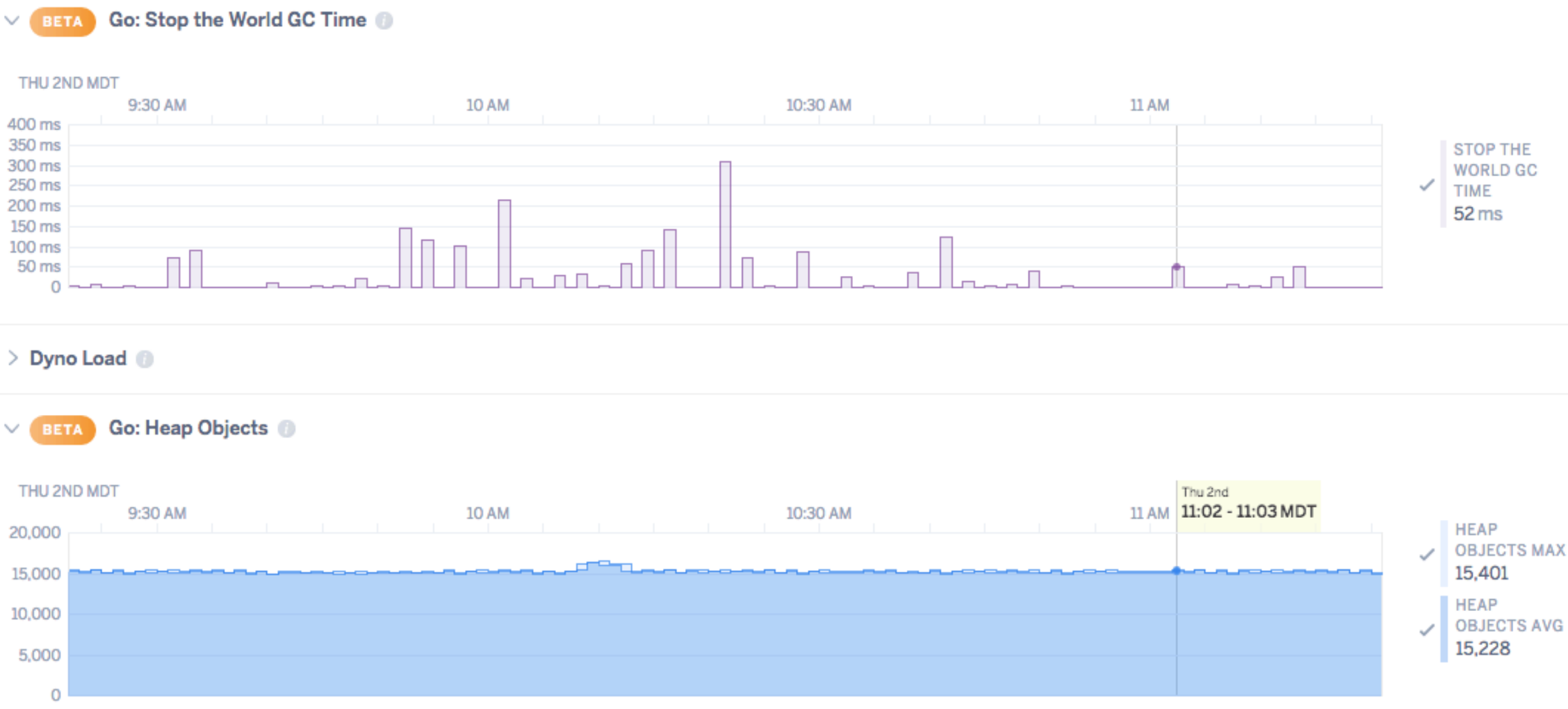
```
$ heroku maintenance:on -a ocapp
Enabling maintenance mode for ⬢ ocapp… done

$ heroku pg:maintenance:run DATABASE -a ocapp
Starting maintenance for postgresql-clean-29349... done

$ heroku maintenance:off -a ocapp
Disabling maintenance mode for ⬢ ocapp... done
```

7. SUPERVISION / MONITORING

Heroku propose un service de monitoring que l'on peut retrouver depuis l'interface d'administration de l'application créée.
Exemple de monitoring :



8. PROCÉDURE DE SAUVEGARDE / RESTAURATION

8.1 CRÉER UNE SAUVEGARDE

Chaque base de données Heroku Postgres de niveau professionnel est livrée avec un mécanisme de protection continue en coulisse en cas de sinistres et possède un système de sauvegarde en option pouvant être controlé.

Par défaut, pg:backups fonctionne sur votre base de données principale, identifiée par la variable de configuration DATABASE_URL.

```
$ heroku pg:backups:capture --app ocapp
Hit Ctrl-C at any time to stop watching progress; the backup will
continue running. Stop a running backup with heroku pg:backups:cancel.

HEROKU_POSTGRESQL_BLACK (DATABASE_URL)  ----backup--->  b251

Running... done
```

Si vous avez plusieurs bases de données sur votre application, vous pouvez choisir celle à sauvegarder en spécifiant le nom de la base de données.

```
$ heroku pg:backups:capture HEROKU_POSTGRESQL_PINK
Hit Ctrl-C at any time to stop watching progress; the backup will
continue running. Stop a running backup with heroku pg:backups:cancel.

HEROKU_POSTGRESQL_PINK  ----backup--->  b252

Running... done
```

Vous pouvez utiliser l'indicateur --verbose pour afficher les journaux au fur et à mesure de la progression de votre sauvegarde. Si vous devez arrêter une sauvegarde pour une raison quelconque, utilisez la commande cancel :

```
$ heroku pg:backups:cancel

Canceled backup b252
```

8.2 PLANIFICATION DES SAUVEGARDES

En plus des sauvegardes déclenchées manuellement, vous pouvez planifier des sauvegardes automatiques régulières. Ceux-ci seront exécutés quotidiennement sur la base de données spécifiée.

```
$ heroku pg:backups:schedule DATABASE_URL --at '02:00 America/Los_Angeles' --app ocapp
```

L'option –at utilise une horloge de 24 heures pour indiquer l'heure de la journée à laquelle vous souhaitez que la sauvegarde soit effectuée. Pour arrêter les sauvegardes régulières, utilisez unschedule :

```
$ heroku pg:backups:unschedule DATABASE_URL --app ocapp
```

8.3 RESTAURATION

Pour restaurer une sauvegarde, utilisez la commande restore :

```
$ heroku pg:backups:restore b101 DATABASE_URL --app ocapp
```

Cela restaurera l'ID de sauvegarde B101 sur l'URL de base de données spécifiée dans l'application. Remarque : vous pouvez omettre l'ID de sauvegarde et la base de données cible vers laquelle restaurer la dernière sauvegarde DATABASE_URL, sinon l'ID de sauvegarde et la base de données cible doivent être fournis.

FIN