

# OC Pizzeria

## Gestion de commande

Dossier d'exploitation

Version 2.0

**Auteur**  
Jean baptiste Servais  
*Developpeur*

# 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
3.2 -Bases de données.....	5
3.3 -Web-services.....	6
<b>4 -Procédure de déploiement.....</b>	<b>7</b>
4.1 -Déploiement de l'application.....	7
4.1.1 -Les éléments composant l'application.....	7
4.1.2 -Variables d'environnement.....	8
4.1.3 -Configuration.....	8
4.1.3.1 -Déploiement.....	9
4.1.3.2 -Création de l'application.....	9
4.1.4 -Vérifications.....	10
4.2 -Déploiement de la base de donnée.....	11
4.2.1 -Vérification.....	11
<b>5 -Procédure de démarrage / arrêt.....</b>	<b>12</b>
5.1 -Maintenance HEROKU.....	12
5.2 -Démarrage/arret avec heroku CLI.....	12
<b>6 -Procédure de mise à jour.....</b>	<b>13</b>
6.1 -Heroku CLI maintenance.....	13
<b>7 -Supervision/Monitorin.....</b>	<b>14</b>
<b>8 -Procédure de sauvegarde et restauration.....</b>	<b>15</b>
<b>9 -Glossaire.....</b>	<b>16</b>

# 1 - VERSIONS

Auteur	Date	Description	Version
Jean baptiste Servais	03/04/19	Création du document	01/02/00

## 2 - INTRODUCTION

### 2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application: "gestion de commande d'OcPizzeria".

L'objectif du document est de constituer un dossier informatif sur les outils que nous allons utiliser. Que soit au niveau de la base de donnée (Postgresql) mais aussi du serveur, du déploiement de l'application (sur Heroku) et du framework (Django).

### 2.2 - Références

Pour de plus amples informations, se référer :

1. **DCT – 2.0** : Dossier de conception technique de l'application
2. **DCF – 2.0** : Dossier de conception fonctionnelle de l'application

# 3 - PRÉ-REQUIS

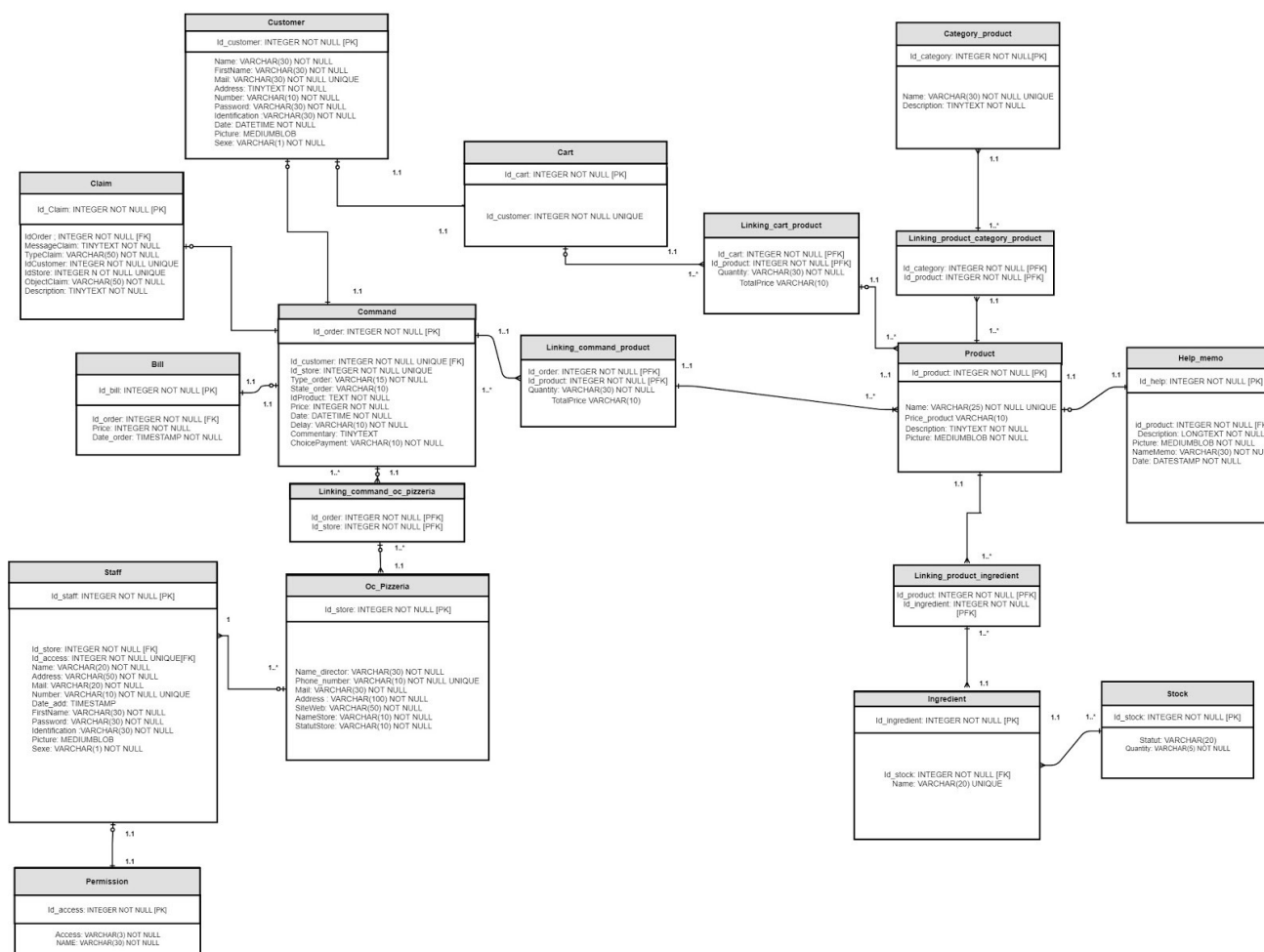
## 3.1 - Système

### 3.1.1 - Serveur de Base de données

Nous allons nous servir d'une base de donnée iWeb déployée avec un système de gestion de base de donnée (SGBD) de type PostgreSQL que nous déploierons sur Heroku.

## 3.2 - Bases de données

Voici le schéma de la base de donnée. La base de donnée est à jour :



### 3.3 - Web-services

Nous utiliserons différents web services qui sont accessibles et à jour :

- Paypal pour le paiement en ligne,
- Bootstrap (thème de template payant) pour le design de l'application.

# 4 - PROCÉDURE DE DÉPLOIEMENT

## 4.1 - Déploiement de l'application

### 4.1.1 - Les éléments composant l'application

Afin de pouvoir déployer notre application sur Heroku il faut au préalable:

- construire un fichier `.gitignore.txt` car lors du push tous les éléments de ce fichier ne seront pas pris en compte.
- Un fichier `Procfile` sans extension contenant web: `"unicorn plateforme.wsgi --log-file -"`.
- Un fichier `requirements.txt` contenant toutes les librairies qui ont été utilisées dans l'ensemble de l'application (pour cela il suffit de faire un `pip freeze > requirements.txt`).
- Un `runtime.txt` contenant la version de python que nous utilisons (version 3-7-1 en effet, cela va renseigner à heroku le langage qui est utilisé).
- Un plan de test au format `.txt` (assurant ainsi le bon fonctionnement des pages web, des fonctions utilisées et de nos requêtes api).
- Un dossier `"staticfiles"` (copie du fichier static) contenant toutes les images, css, js, thème bootstrap ...
- Un dossier `"templates"` contenant nos templates (nos pages web).
- Le dossier contenant l'application `"Pizzeria"` (contenant le fichier `setting_production.py` pour la production et settings pour le développement)
- Le dossier contenant l'application Users.
- Le dossier contenant l'application Cart.
- Le dossier contenant l'application Claim.
- Le dossier contenant l'application Order.
- Le dossier contenant l'application Store.
- Le dossier contenant l'application Stock.
- Le jeu de donnée postgresql pour la database.

## 4.1.2 - Variables d'environnement

Voici les variables d'environnement de l'application Gestion de commande OcPizzeria :

Nom	Obligatoire	Description
Settings	oui	Pour signaler que nous sommes en développement et non en production car en production nous nous servons de setting_Production.py
Secret_Key	Oui	Secret_key Django nécessaire aux hashes
Environnement virtuel	Oui	Sécurise l'environnement du développeur et du développement avec virtualenv ou venv.
Database_url	Oui	Url menant à la base de donnée
Pip	Oui	Qui nous permet d'installer les librairies nécessaires pour l'application.

## 4.1.3 - Configuration

Voici les différents fichiers de configuration :

- **Procfile** : contient : "gunicorn plateforme.wsgi --log-file -". Grâce à ce fichier Heroku utilise le serveur web gunicorn.
- **requirements.txt** : contient toutes les librairies que nous avons utilisées. Ce fichier dit à Heroku quoi installer mais renseigne aussi les autres membres de l'équipe sur les librairies utilisées.
- **setting\_prod.py**: settings pour la production. Signal à Heroku la nouvelle database, le nouvelle host ect.

A noter qu'il suffit d'importer le fichier de développement settings.py et à y apporter les modifications dans settings\_prod.py.

N'oubliez pas de mettre à debug = True → mode développement, False production.



#### 4.1.3.1 - Déploiement

Le déploiement sur Heroku nécessite: *les éléments composants l'application web* en 4.1.1.

Heroku propose différentes manières de déployer l'application. La première est avec heroku CLI (à installer) et l'autre avec Github (GIT). Nous avons choisi de prendre la technique d'heroku CLI bien sur nous sommes ouvert à l'autre proposition.

Pour cela il faut:

installer Heroku CLI <https://devcenter.heroku.com/articles/heroku-cli> afin de pouvoir interagir avec le cmd,

puis de se login à partir du cmd (ctrl+r → cmd) après s'être inscrit à Heroku sur <https://signup.heroku.com>.

Une fois login activez votre environnement virtuel.

Clonez votre dossier avec heroku 'git:clone -a application\_name' (toujours sur le cmd);

positionnez-vous dans votre application cd application\_name.

On ajoute ! git add .,

on commit ! git commit -am "make it better" et on push ! git push heroku master.

*Instructions issuent d'Heroku*

#### 4.1.3.2 - Création de l'application

Pour pouvoir faire notre application sous Django il faut par exemple:

se diriger vers le cmd ctrl + r → cmd tapez entrée.

Créez un environnement virtuel avec pip (vérifiez que vous l'avez installé lors du téléchargement de python sinon réinstallez ou modifiez python en cochant pip).

Installez virtualenv avec pip install virtualenv.

Puis tapez virtualenv nom\_app (ici nous créons notre environnement virtuel).

Activez l'environnement virtuel !

Cd Script (dans notre environnement virtuel) active cd la racine,

installez Django ! Pip install django.

Créez votre projet avec django-admin startproject nom\_application puis configurez vos urls et faites un petit coup de manage.py makemigrations, puis manage.py migrate.

Créez un superuser avec manage.py createsuperuser puis faites vos applications avec manage.py startapp nom\_application.

#### 4.1.4 - Vérifications

Afin de vérifier le bon déploiement de l'application allez sur Heroku et cliquez sur openapp. Si la page affiche un message comme log-rail tapez dans le cmd. Cela vous décrit les erreurs. Réparez les ! S'il n'y a pas de bug l'application apparaît. Attention au fichier Procfile, P majuscule, sans extension. Installez bel et bien gunicorn.

Sinon pour une première visualisation lors de votre git push heroku master regardez attentivement les différentes lignes. Cela peut être une première indication des erreurs. A la fin normalement il devrait y avoir écrit **remote: Verifying deploy...done**. Cela est un bon signe pour un bon déploiement.

## 4.2 - Déploiement de la base de donnée

Direction Heroku ! Il faut aller dans Ressource depuis votre dashboard depuis votre application. Dans les Add-ons installez Heroku Postgresql. Afin de pouvoir vérifier vos données vous pouvez vous servir de psycopg2 (librairie python, pip install psycopg2) ;

ou vous servir de pgAdmin4 (installez le depuis internet <https://www.postgresql.com>, n'oubliez pas le mot de passe, sinon servez vous du root avec psql (set PATH=%PATH%; (copie de l'url du dossier BIN) postgresql -u root -p, tapez le mot de passe.

Sinon servez vous de pgadmin4 configurez un serveur et rentrez les informations\* issus de Heroku (cd prochain paragraphe)).

\*En prenant les informations depuis : cliquez sur heroku Postgresql → réglages → voir les références et prendre les informations: hôte, base de donnée, utilisateur, port et mot de passe.

Pour Django en production allez dans vos settings et dans DATABASE mettez DATABASE['default'] = dj\_database\_url.config().

Sinon pour le mode développement faites une base de donnée locale.

Il va falloir faire une migration avec Heroku pour les databases et l'orm de django.

Faites depuis le cmd **manage.pyheroku shell**.

Vérifiez les migrations avec heroku run python manage.py makemigrations. Puis migrate. Afin de voir les migrations allez dans le dossier de l'application, ouvrez le dossier migrations, normalement pour la première migrations le fichier s'appelle 001.xxx.py

Créez un superutilisateur avec Heroku run python manage.py createsuperuser.

### 4.2.1 - Vérification

Sur Heroku allez dans ressource cliquez sur heroku postgresql (dans les Addons-on) une nouvelle page s'ouvre et vous pouvez vérifier l'état de la database. Encore une fois vous pouvez visualiser votre database avec psycopg2 ou pgadmin4.



# 5 - PROCÉDURE DE DÉMARRAGE / ARRÊT

## 5.1 - Maintenance HEROKU

Depuis Heroku nous pouvons mettre le site en maintenance.

Direction Heroku, settings tout en bas vous pouvez mettre la maintenance en mode ON. Par défaut elle est en off.

## 5.2 - Démarrage/arrêt avec Heroku CLI

Allez dans votre cmd. Tapez Heroku run pour activé l'application.

Pour la désactiver tapez **heroku ps:scale web=0**.

Pour la réactiver tapez **heroku ps:scale web=1**.

## 6 - PROCÉDURE DE MISE À JOUR

Afin de pouvoir faire une mise à jour il faut mettre le site en maintenance.

### 6.1 - Heroku CLI maintenance

Vous pouvez faire votre maintenance directement sur Heroku cité plus haut en 5.1 ou avec client en tapant les lignes suivantes:

Heroku maintenance:on

ou Heroku maintenance:off

## 7 - SUPERVISION/MONITORIN

Heroku dispose d'outils de monitoring. Allez sur Heroku dans Metrics. Cependant ajouter des métrics coûte 7\$. Pour de plus amples informations direction:

"<https://devcenter.heroku.com/articles/metrics>"

## 8 - PROCÉDURE DE SAUVEGARDE ET RESTAURATION

Afin de pouvoir sauvegarder et faire des restaurations vous pouvez aller sur Heroku, dans "activité". Vous pouvez revenir à une sauvegarde antérieure en cliquant sur revenir ici.

Sinon vous pouvez, avec le cmd sauvegarder la base de donnée avec:

```
heroku pg:backups:capture --app nom_application
```

Afin de pouvoir restaurer la base de donnée faites:

```
heroku pg:backups:restore<nom_sauvegarde> DATABASE_URL --app non_application
```



## 9 - GLOSSAIRE

Python	Langage de programmation
Base de donnée	Permet de pouvoir stocker des informations
SGBD	Système de gestion de base de donnée
Iweb	Base de donnée hébergée par le web
Django	framework
framework	Infrastructure logicielle
Heroku	Plateforme de déploiement