|  |  |
| --- | --- |
| **OCPizza**  **Projet 8**  Dossier d'exploitation  Version 1.0 | |
| Auteur  **Nora Lilla Matyassi**  *Développeuse Junior* | |

Table des matières

[1 - Versions 4](#_Toc93966600)

[2 - Introduction 5](#_Toc93966601)

[2.1 - Objet du document 5](#_Toc93966602)

[2.2 - Références 5](#_Toc93966603)

[3 - Pré-requis 6](#_Toc93966604)

[3.1.1 - Serveur OC Pizza et OCPizzaDB 6](#_Toc93966605)

[3.1.2 - Heroku 6](#_Toc93966606)

[3.1.3 - App Store 7](#_Toc93966607)

[3.2 - Système 7](#_Toc93966608)

[3.2.1 - Serveur de Base de données 7](#_Toc93966609)

[3.2.2 - Caractéristiques techniques 7](#_Toc93966610)

[3.2.3 - Serveur OCPizza 7](#_Toc93966611)

[3.3 - Bases de données 8](#_Toc93966612)

[3.4 - Web-services 8](#_Toc93966613)

[3.5 - Autres Ressources 8](#_Toc93966614)

[4 - Procédure de déploiement 9](#_Toc93966615)

[4.1 - Déploiement sur App Store 9](#_Toc93966616)

[4.1.1 - Xcode 9](#_Toc93966617)

[4.1.2 - S’inscrire dans l’Apple Developer Program 10](#_Toc93966618)

[4.1.3 - App Store Connect 11](#_Toc93966619)

[4.1.4 - Préparer l’application à la publication 11](#_Toc93966620)

[4.1.5 - Tester l’application et fixer les bugs 12](#_Toc93966621)

[4.1.6 - Créer son App Store listing 14](#_Toc93966622)

[4.1.7 - Créer les captures d’écrans pour l’App Store 16](#_Toc93966623)

[4.2 - Déploiement de serveur OCPizza 17](#_Toc93966624)

[4.2.1 - Déploiement avec Heroku 17](#_Toc93966625)

[4.3 - Déploiement d’un serveur de base de données sur AWS 18](#_Toc93966626)

[4.3.1 - Démarrer un instance RDS avec PostgreSQL 18](#_Toc93966627)

[5 - Procédure de démarrage / arrêt 19](#_Toc93966628)

[5.1 - Base de données 19](#_Toc93966629)

[5.2 - Serveur OCPizza 19](#_Toc93966630)

[6 - Procédure de mise à jour 20](#_Toc93966631)

[6.1 - Base de données 20](#_Toc93966632)

[6.2 - Serveur OCPizza 20](#_Toc93966633)

[7 - Supervision/Monitoring 21](#_Toc93966634)

[7.1 - Supervision de l’application web 21](#_Toc93966635)

[8 - Procédure de sauvegarde et restauration 22](#_Toc93966636)

[9 - Glossaire 23](#_Toc93966637)

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| NLM | 20/01/2022 | Création du document | 1.0 |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier d’exploitation de l'application OCPizza

Le présent document décrit le déploiement et la maintenance de l’application.

## Références

Pour de plus amples informations, se référer :

1. **DCT - PDOCPizza\_01\_fonctionnelle :** Dossier de conception fonctionnelle de l'application
2. **DCT - PDOCPizza\_02\_technique :** Dossier de conception technique de l'application

# Pré-requis

### Serveur OC Pizza et OCPizzaDB

Avant de pouvoir déployer nos serveurs nous avons besoin d’intégrer le framework Vapor dans notre application.

Pour installer le Vapor Toolbox nous avons besoin de Homebrew et faire la commande suivante dans le Terminal :

brew install vapor/tap/vapor

Pour créer notre projet avec Vapor nous allons créer un nouveau projet (nous allons le cloner de github)

vapor new OCPizza –- template=OCPizza/OCPizza

cd OCPizza

Pour l’installer dans notre projet :

vapor build

Pour le générer dans Xcode :

vapor xcode -y

Xcode va démarrer. Ici à la place du package proposé dans Scheme nous allons choisir Run et MyMac. Ensuit cliquer sur Run.

Pour vérifier si notre serveur a bien démarré, nous allons ouvrir Safari et aller sur

<http://127.0.0.1:8080>

### Heroku

Nous allons créer un compte Heroku

<https://signup.heroku.com/>

### App Store

Pour déployer notre application sur l’App Store il est impérative d’être inscrit dans l’Apple Developer Program. ( <https://developer.apple.com/programs/> )

## Système

### Serveur de Base de données

Pour le serveur de base de données nous allons utiliser un plan Standard de Heroku pour PostgreSQL.

### Caractéristiques techniques

**Standard 2**

* nombre de lignes est illimité
* RAM de 8GB
* espace de stockage de 256GB
* 400 connexions
* rollback de 4 jours
* cryptage des données
* 200$ par mois

### Serveur OCPizza

Le serveur OC Pizza sera déployé sur Heroku. Nous avons choisi un package Standard & Performance qui réponde parfaitement aux besoins de notre application.

#### Caractéristiques techniques

**Standard 2X**

* RAM de 1GB
* déployable de git
* patchs OS automatisé
* logs unifiés
* types de processus illimités
* toujours en marche
* custom domains
* SSL gratuit sur les custom domains
* mise en échelle horizontale
* preboot
* des métriques détaillés
* 50$ par mois.

## Bases de données

Les bases de données et schémas suivants doivent être accessibles et à jour :

* OCpizzaDB : version 1.0

## Web-services

Les web services suivants doivent être accessibles et à jour :

* … : version XXX

# Procédure de déploiement

## Déploiement sur App Store

### ***Xcode***

Avant de publier une application sur l’App Store il est nécessaire de la finaliser dans Xcode.

* 1. Faire un double clic sur le nom du projet
  2. Cliquer sur « Target »
  3. Dans le tab « Générale » il faut incrémenter le numéro de version et le numéro de build
  4. Changer l’appareil en générique en passant par Produit > Destination > iOS appareil
  5. Vérifier si le mode de build est en mode production
  6. Créer une image de l’application en passant par Produit > Archiver
  7. L’archivage terminé un fenêtre Archive va apparait dans laquelle il va falloir choisir la version actuelle et ensuit cliquer sur « Télécharger sur l’App Store »

### S’inscrire dans l’Apple Developer Program

Il coute 99$ par ans. Il permette de publier des applications dans l’App Store, de créer des extensions Safari et de les lister dans la galerie des extensions, accéder aux versions beta des softwares Apple, utiliser TestFlight, profiter d’App analyitcs.

A noter, si la suscription n’est pas renouvelée, l’application est retirée de l’App Store.

Graphical user interface

Description automatically generated

### App Store Connect

Après l’inscription dans l’Apple Developer Program, nous avons accès à App Store Connect qui nous permette de profiter des avantages listés auparavant.

Graphical user interface, application

Description automatically generated

### Préparer l’application à la publication

Avant de publier l’application sur l’App Store il faut vérifier si elle est conforme aux règles de l’App Store. Elle peut passer le processus de certification de l’application si elle est conforme à 100%.

Pour prendre connaissance de l’intégralité des règles veuillez référer au lien suivant.

<https://developer.apple.com/app-store/review/guidelines/>

Graphical user interface, text, application, Word, email

Description automatically generated

### Tester l’application et fixer les bugs

Il est nécessaire de tester l’application dans des conditions différents que dans Xcode. Pour réaliser ces tests nous allons se servir de TestFlight qui nous est accessible par l’Apple Developer Program. Ici nous pouvons faire tester nos applications par nos équipes, et des personnes indépendants pour recevoir du feedback.

<https://developer.apple.com/testflight/>

Graphical user interface, application

Description automatically generated

### Créer son App Store listing

* + 1. Se connecter à App Store Connect :

Pour publier une application il faut Cliquer sur « My Apps » et sélectionner l’option « + ».

Graphical user interface, application, email

Description automatically generated

2. Remplir les cases concernant l’application (nom et catégorie)

Graphical user interface, application, email

Description automatically generated

3. Donner l’URL de la politique sur la protection des renseignements personnels

Graphical user interface, text, application

Description automatically generated

4. Donner le prix de l’application

Graphical user interface, text, application, email, Teams

Description automatically generated

5. Choisir la date de publication de l’application

Graphical user interface, text, application, email

Description automatically generated

### Créer les captures d’écrans pour l’App Store

Pour publier une application sur l’App Store il faut créer des captures d’écrans pour chaque type d’écran si l’application est disponible sur des appareilles de tailles différents (iPhone / iPad / iWatch)

Graphical user interface, application

Description automatically generated

## Déploiement de serveur OCPizza

### Déploiement avec Heroku

Pour commencer nous allons installer Heroku CLI avec la commande suivante dans le Terminal

brew tap heroku/brew && brew install heroku

Nous avons déjà notre compte Heroku, donc nous allons continuer avec la commande suivante toujours dans le Terminal

heroku login -i

Avant de déployer avec Heroku il faut être certain d’avoir initialiser un dépôt local et d’avoir fait un commit de l’application dessus. Si ce n’est pas fait voici un exemple :

Text

Description automatically generated

Pour l’instant le code est suivi dans le dépôt local, mais n’a pas encore été poussé sur le serveur.

Afin de déployer l’application sur le serveur il faut pousser le code sur un dépôt spécial distant dédié de Heroku. Cette commande va créer une application vide sur Heroku, avec un dépôt Git associé, qui va être le dépôt distant par default à notre dépôt local.

$ heroku create

Pour vérifier si nous avons bien créer un dépôt distant au nom de « heroku » :

$ git remote -v

Par défaut tous les dépôts sont créés sous le nom « heroku ». Pour renommer ce dépôt :

$ git remote rename heroku OCPizzaHeroku

Pour deployer notre application nous allons utiliser le command push pour pousser le code du branche main ou master de notre dépôt local à notre dépôt OCPizzaHeroku :

$ git push OcPizzaHeroku main

Pour sortir du processus de push il suffit de faire Ctrl + C, le processus va continuer dans l’arrière-plan, il n’arrête pas le processus de build ou le déploiement.

## Déploiement d’un serveur de base de données sur Heroku

### Démarrer un instance Heroku Postgres

Pour commencer le déploiement de notre base de données nous allons provisionner un plan Heroku en précisant le nom du plan et la version de PostgreSQL désirée

$ heroku addons : create heroku-postgresql:standard-2 : standard --version=12

Pendant le processus de provisionnement un DATABASE\_URL est ajouté à la configuration de notre application. Le DATABASE\_URL contient l’URL que l’application utilise pour accéder à la base de données.

Quand le processus est terminé une base de donnée PostgreSQL vide est provisionnée.

Heroku recommande d’utiliser l’application Postgres pour assurer la parité entre les différents environnements.

<https://postgresapp.com/>

Après avoir installé l’application Postgres il faut exporter la variable de DATABASE\_URL pour que notre application puisse s’y connecter.

$ export DATABASE\_URL=postgres://$(whoami)

Le fichier de configuration DATABASE\_URL d’une application désigne la base de données prioritaire sur Heroku Postgres (il peut y en avoir plusieurs) Pour des applications avec une seule base de données, son URL est assignée d’une manière automatique.

Pour établir une session avec la base de données distant il suffit de faire la commande :

$ heroku pg:psql

Pour transférer des données de notre base de données local :

$ heroku pg:push <mylocaldb> HEROKU\_POSTGRESQL\_<DBCOLOR> --app OCPizza

Pour consulter d’autres commandes CLI consulter ce lien :

<https://devcenter.heroku.com/articles/managing-heroku-postgres-using-cli#pg-push-and-pg-pull>

Pour établir la connexion entre notre application et notre base de données, nous allons dans le directory de l’app OCPizza dans le Terminal et nous exécutons la commande suivante :

$ heroku config

A partir de ce moment-là nous avons la possibilité d’accéder et de manipuler notre base de donné dans Xcode.

Il faut toujours faire des commits les modifications.

# Procédure de démarrage / arrêt

## Base de données

A partir du moment où notre base de données est déployée sur Heroku, elle est démarrée. Il n’existe pas un arrêt proprement dit, nous avons la possibilité de la redémarrer en cas de besoin ou de la supprimer.

## Serveur OCPizza

Dans le cas des serveurs Heroku il est possible d’arrêter leur fonctionnement pour cause de maintenance. Un des possibilités est de passer par le Terminal et faire la commande suivante :

$ heroku maintenance:on

Maintenance mode enabled.

Pour quitter le mode maintenance :

$ heroku maintenance:off

Maintenance mode disabled.

Ou aller sur leur site et activer/désactiver le mode maintenance dans le panel de l’admin.

# Procédure de mise à jour

## Base de données

Dès que nous apportons des modifications à notre base de données il faut les sauvegarder réaliser les commits Git ensuite pousser les changements sur notre dépôt de base de données.

## Serveur OCPizza

Pour réaliser des mises à jour, il faut pousser la nouvelle version sur Hiroku et publier la nouvelle version sur l’App Store avec App Store Connect en mettant à jour les captures d’écran et publier une description détaillée des changements.

# Supervision/Monitoring

## Supervision de l’application web

Pour superviser notre application il suffit de se connecter comme admin sur le site de Heroku aller sur le Dashbord et nous pouvons regarder les métriques.

Le monitoring de l’application est possible via l’App Store Connect pour les metrics et les rapports de ventes, mais également dans Xcode pour les crashes.

# Procédure de sauvegarde et restauration

## Application OCPizza

Comme nous avons choisi de travailler avec Vapor et Heroku, tout est constamment sauvegardé sur Git. Avec le contrôle de version il est toujours possible de retourner sur une version précédente en cas de problème.

Pour voir les versions publiées de l’application :

$ heroku releases

Pour avoir les informations sur une version en particulier :

$ heroku releases:info v24

Pour retourner à une version précédente :

$ heroku rollback « numero de version »

## Base de données OCPizzaDB

Pour créer des backups de notre base de données :

$ heroku pg:backups:capture –<name of the app>

Le nombre des backups possible à garder est de 25. Si nous avons atteint les 25 backups, la commande de capture va automatiquement sauvegarder sur le backup le plus ancien.

Il est également possible de faire des backups automatiques.

Pour restaurer un backup :

heroku pg:backups:restore sushi::<backup id> DATABASE\_URL --<name of the app>-staging

# Glossaire

|  |  |
| --- | --- |
| Multi-AZ | Zone de disponibilité mulitple |
| **RDS** | Service de base de données relationnelle géré |
| **EC2** | Amazon Elastic Compute Cloud, qui offre une capacité de calcul évolutive dans le cloud Amazon Web Services |