OC_PIZA DOPRR

OC-Pizza

Oc-Pizza Application web

Dossier d'exploitation

Version 1.0.0

Auteur
Diego Patino
Chef de Project

TABLE DES MATIÈRES

1 - Versions

Auteur	Date	Description	Version
Diego Patino	11/12/19	Création du document	1.0.0
Diego Patino	11/12/19	Documentation complete	1.0.1

2 - Introduction

2.1 - Objet du document

Le document suivant a pour objectif d'expliquer de manière très claire et efficace , lLes étapes de déploiement, démarrage , et maintenance de l'application web OC-Pizzas, ce document et adresse au administrateurs des systèmes , ainsi à l'ensemble de l'équipe de développeurs.

2.2 - Références

Pour de plus amples informations, se référer :

1. **OC-PIZZAS** - Dossier de conception technique : Dossier de conception technique de l'application
2. **OC-PIZZAS** - Dossier de conception fonctionnelle

3 - Pré-requis

3.1 - Système

3.1.1 - Serveur de Base de données

Serveur de base de données sera un serveur mysql.

Il sera déployé sur un conteneur Docker,

Le schéma pour la base OC PIZZA, contenant le stable suivantes :

- table_clients,
- table_order,
- table_store,
- table_employees.

les données seront stockées physiquement sur des volumes virtuelles, utilisant la technologie docker volumes.

3.1.1.1 - Caractéristiques techniques

L'image du conteneur docker du serveur de bases de donnes sera celle avec le tag :

mysql :5.7

La documentation est disponible sur l'adresse suivante :

https://hub.docker.com/_/mysql?tab=description

3.1.2 - Serveur Web

Le serveur contenant l'application front sera une serveur nodejs a l'intérieur de un conteneur docker.

3.1.2.1 - Caractéristiques techniques

L'image du conteneur docker su serveur web pour l'application web sera celle avec le tag :

node :9-alpine

La documentation est disponible sur l'adresse suivante :

https://hub.docker.com/_/node?tab=description

3.1.3 - Serveurs des Api

Les Diferents Api de l'application sont developes en dotnet-core, chaque api sera disponible dans une image docker repectivement :

- oc_pizzas/user-api.
- oc_pizzas/products-api.
- oc_pizzas/order-api.
- oc_pizzas/store-api.
- oc_pizzas/employees-api.
- oc_pizzas/stock-api.
- oc_pizzas/recipes-api.

La documentation des api est disponible sur :

oc_pizzas-api ;

La norme utilse pour les c'est open api-3, la documentation est disponible sur

https://www.openapis.org/

l'editeur choisit cest swagger,la documentation est disponible sur :

https://swagger.io/tools/open-source/

3.2 - Bases de données

Pour la genereation du serveur de base de donnes, utilise le fichier disponible dans le repertoire

\Application\Services\Database\MySql\Dockerfile.MySql

Pour la generation des bases de donnes , on utilise entity framework, au lancement de chaque api,les bases des donness respectives seront generes. Chaque microservice posede un fichier

{µ_SERVICE_NAME}dbsetup.sql .

ce fichier se charge de creer la configuration de la base dans le serveur mysql.

La documentation de entity framework est disponilbe sur :

Présentation d'Entity Framework

l'insertion de donnes dans la base se fait via des seeders, disponibles au niveau de chaque project .

La documentation descriptif de ce procesus est dispoible sur le lien suivant

https://en.wikipedia.org/wiki/Database_seeding

3.3 - Web-services

Les web services suivants doivent être accessibles et à jour :

- https://oc_pizzas.fr/oc_pizzas/user-api.
- https://oc_pizzas.fr/oc_pizzas/products-api.
- https://oc_pizzas.fr/oc_pizzas/order-api.
- https://oc_pizzas.fr/oc_pizzas/store-api.
- https://oc_pizzas.fr/oc_pizzas/employees-api.
- https://oc_pizzas.fr/oc_pizzas/stock-api.
- https://oc_pizzas.fr/oc_pizzas/recipes-api.

4 - Procédure de déploiement

4.1 - Déploiement des api, Application Web, serveur des basse des donnes

4.1.1 - Artefacts

Les diferents services de l'application oc-pizzas sont construites sous la forme d'images docker :

- oc_pizzas.fr/oc_pizzas/user-api:latest
- oc_pizzas.fr/oc_pizzas/products-apii:latest
- oc_pizzas.fr/oc_pizzas/order-api:latest
- oc_pizzas.fr/oc_pizzas/store-api:latest
- oc_pizzas.fr/oc_pizzas/employees-api:latest
- oc_pizzas.fr/oc_pizzas/stock-api:latest
- oc_pizzas.fr/oc_pizzas/recipes-api:latest

Le fichier :

buid.sh

disponible dans le project au nivau du repertoire Application, permet de generer les images docker nescesaires pour demarrer l'application,.

la commande **docker images** permet de verifier la liste des images generees.

Le fichier :

push.sh

disponible dans le project au nivau du repertoire Application, permet de demarrer l'application en local.

4.1.2 - Variables d'environnement

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

Nom	Obligatoire	Description
APP_HOST	oui	Url du site utilise par le serveur roxy

Les variables d'environement de chaque service sont decrites dans les parametres respectives au project. Les valuers sont definies dans le fichier :

Application\DockerCompose\env\docker-compose.{environement}.yml

4.1.3 - Configuration

Voici les différents fichiers de configuration :

- .env: fichier de configuration du host

4.1.4 - Vérifications

Afin de vérifier le bon déploiement des microservices, se placer au niveau du repertoire Application, et lancer la command:

dotnet test --filter HEALT-CHECK

cette commande permet de verifier que tous le microservices sont en marche et deployés sur le serveur.

Afin de verifier le site web la commande

curl -I http://www.oc-pizza.com

doit repondre avec le code 200.

si les test **HEALT-CHECK** on reussi, onpeut deduire que le serveur de base de donnes est disponible.

4.1.5 - DataSources

Les accès aux bases de données doivent se configurer à l'aide des fichiers dispoibles au niveau de chauqe project d'api, ces fichier se trouvent sous la forme :

{µ_SERVICE_NAME}dbsetup.sql

ces fichier sont deployes automatiquement losr de la generation du conteneur de la abses de donnes.

$home_server/lib/ext

5 - Procédure de démarrage / arrêt

5.1 - Base de données

Sur le serveur de Production, lance la commande :

docker ps

recuperer l'id du conteneur docker qui utilise limage pour le serveur de base de donnes.

Avec la commande :

docker stop {id_conteneur}

le serveur de base de donnes sera arreter

5.2 - Api,Application web

Sur le serveur de Production, lance la commande :

docker ps

copier les fichier : **Application/scripts/stop.sh** sur le servuer de production.

Changer les droits sur le fichier avec la commande

chmod 7777 ./stop.sh

Lancer lefichier avec la commande:

./stop

Tous les conteneur pour les api , ainsi que le conteneur pour l'application web seront arretes.

6 - Procédure de mise à jour

6.1 - Base de données

Se placer au niveau du repertoire :

\Application

lancer la commande :

./scripts/build.sh

une fois les images docker generes.

Lancer la commande:

./scripts/push.sh

ceci mettra a jour les image docker dans le repo docker prive.

Une fois les image mise a jour sur le repo docker .

Copier le fichier :

\Application\scripts\run.sh

sur les serveur production; se connecter au serveur , et lancer le script.

7 - Procédure de sauvegarde et restauration

Pour les backup de la base de donnes; il faut se connecter au serveur de production.

1. Sur le serveur de Production, lancer la commande :

   docker ps

2. recuperer l'id du conteneur docker qui utilise limage pour le serveur de base de donnes. Avec la commande :

   docker stop {id_conteneur}

3. executer la commande :

   docker exec -it {id_conteneur} /sh

   ceci ouvira un sheel au niveau du conteneur doker.

4. Lance la commande

   mysqldump --all-databases > dump.sql

5. recuperer les fichier dump.sql et le sauvegarder.

Pour restaurer les donnes, repeter les etapes 1,2,3;

ensuite il fau tcopier le fichier de sauvegar sur le conteneur dokcer. Et lace la commande :

mysql> source dump.sql

8 - Supervision

8.1 - Supervision de l’application web

Se reporter au poitn 4.1.4