OC-Pizza

Oc-Pizza Application web

Dossier de conception technique

Version 1.0.0

Auteur
Diego Patino
chef de Project

TABLE DES MATIÈRES

1 - VERSIONS

Auteur	Date	Description	Version
Diego Patino	11/01/19	Création du document	1.0.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier de conception technique de l'application Oc-Pizzas

Objectif du document c'est de documenter de maniere detaille toutes les besoin techniques du project Oc-pizza, ainsi que de clairifier tous les point importants pour les developpeur travaillant sur le project.

2.2 - Références

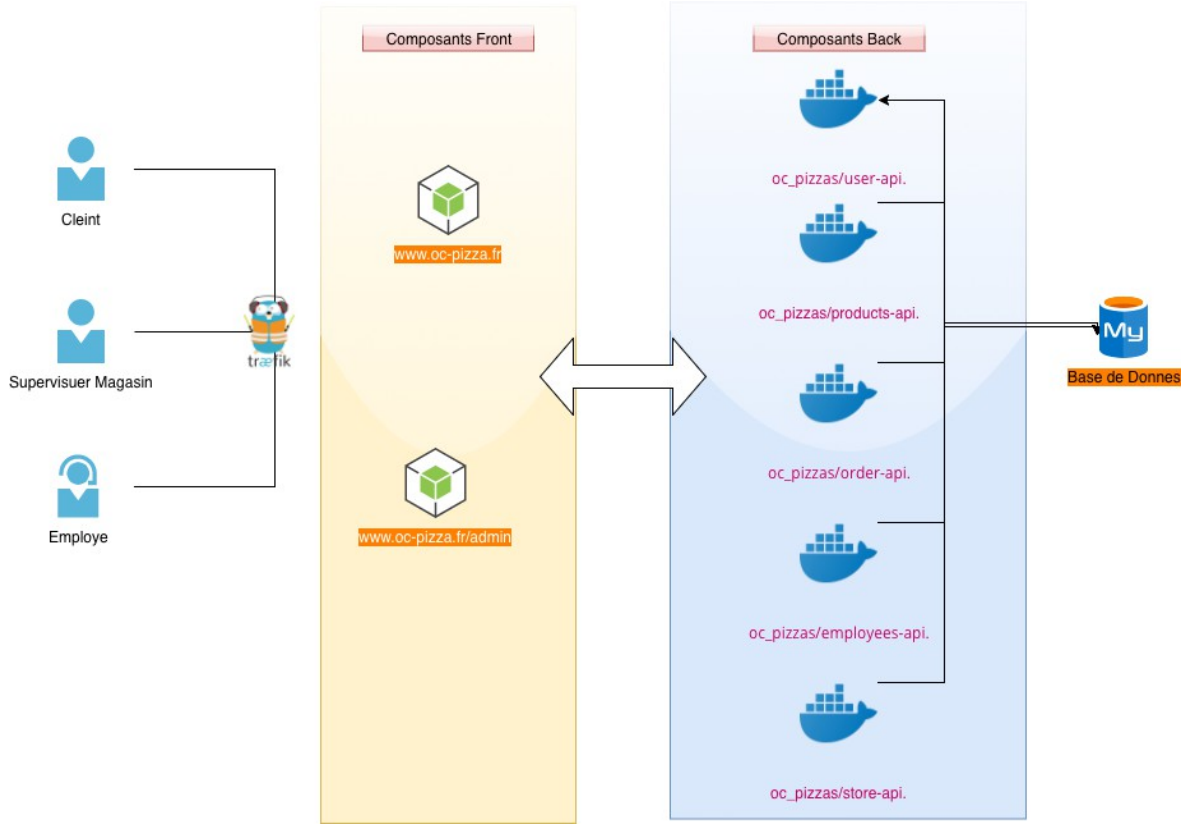
Pour de plus amples informations, se référer également aux éléments suivants:

1. **oc_pizzas-Dossier_de_conception_fonctionnelle** : Dossier de conception fonctionnelle de l'application
2. oc_pizzas-Dossier_d_exploitation Dossier de exploitation de l'application

3 - Architecture Technique

3.1 - Composants généraux

3.1.1 - Diagramme



Les composant principaux de l'architecture sont les serveur dédiés a héberger les interfaces graphiques et les serveurs des Api, un proxy traefik fait le pont entre les différents urls. Tous les composants du backend sur déployés sur des conteneurs docker. Et le serveur des bases de donnes c'est un serveur MySql.

3.1.2 - DockerCompose

Ce dossier contient les fichier nécessaires pour la génération des images Docker , pour le lancement de l'application

3.1.2.1 - DockerCompose/env

Fichier de configuration pour le lancement de l'application sur les différent environnements.

Env, prod ,preprod

3.1.3 - Application

Dossier contenant Le Code source nécessaire pour générer les web-service, les pages html du site, ainsi que les dossier des script de building, demarrage, et reploiement de l'application

3.1.3.1 - Services

3.1.3.2 - oc_pizzas/user-api.

Service chargé de la gestion des comptes utilisateur present sur ll'application oc-pizzas.fr ;

trois type de comptes sont disponibles :

- Client
- Manager
- Enploye

3.1.3.3 - oc_pizzas/products-api.

Service chargé de la liste de produits disponibles pour les chois des utilisateur, ce service géré, les différents associations entres le produits,les prix des produit

3.1.3.4 - oc_pizzas/order-api.

Service chargé de la gestion des commandes, prise de commande et mise a jour de commande

3.1.3.5 - oc_pizzas/store-api.

Service chargé de la gestion du payement des commandes.

3.1.3.6 - oc_pizzas/employees-api.

Service charge de la gestion des employés, disponibilité des employés, nombre de commandes prises en charge

3.2 - Application Web

La pile logicielle est la suivante :

- Application Web AngularJs dans le dossier « Application/Views/Oc-pizza »
- Serveur d'application NodeJs dans le répertoire « src »

l'application web est compose de 4 modules

- web-ui/live
- web-ui /user
- web-ui/commande
- web-ui/catalog

3.2.1 - web-ui/live

Ce module permet de consulter les statut de toues les commandes en cours. Cette liste peut être filtré par magasin, préparateur et statut

3.2.2 - web-ui /user

Ce module géré toutes le interface , de login , creation de compte, gestion de compte ;

3.2.3 - web-ui/commande

Ce module géré la gestion de prise de command. Il est disponible en mode connecte ou déconnecte ;

3.2.4 - web-ui/catalog

Ce module géré l'affichage des produit du catalogue ; il possédé deux sous modules :

3.2.4.1 - web-ui-catalog/admin

Géré l'affichage du stock des ingrédients, géré aussi l'affichage d'aide mémoire pour les employés,

3.2.4.2 - web-ui-catalog/catalog

Géré l'interface présenté aux utilisateur pour pouvoir choisir les produit désirés

4 - Architecture de Déploiement

4.1 - Serveur de Base de données

Le serveur de bases de donnes est un serveur MySql, ce serveur es deploiye sur un conteneur docker.

Le contenu du fichier d'initialisation du serveur est le suivant :

```
CREATE USER IF NOT EXISTS 'doprr'@'%' IDENTIFIED BY 'eltiempo!';
```

Le contenu du fichier destine a la création du conteneur docker est le suivant :

```
FROM mysql:5.7
ENV MYSQL_ROOT_PASSWORD=ElTiempCambia

WORKDIR /src
COPY Services/Database/MySql/0-init-users.sql /docker-entrypoint-initdb.d
COPY Services/Meats/MeatsDbSetup.sql /docker-entrypoint-initdb.d
COPY Services/Recipes/RecipesDbSetup.sql /docker-entrypoint-initdb.d
```

5 - ARCHITECTURE LOGICIELLE

5.1 - Principes généraux

Les sources et versions du projet sont gérées par **Git**, les dépendances et le packaging par :

- **npm :** gestionnaire de package pour les modules NodeJs
- **docker et docker-compose** Docker est un logiciel libre permettant facilement de lancer des applications dans des conteneurs logiciels
- **dotnet :** NET Core est un cadriciel Libre et Open Source pour les systèmes d'exploitation Windows, macOS et Linux. Il comprend CoreCLR, un environnement d'exécution complet de CLR, la machine virtuelle qui gère l'exécution des programmes .NET.

5.1.1 - Les couches

L'architecture applicative est la suivante :

pour les micro-services:

- une couche **business** : responsable de la logique métier du composant
- une couche **model** : implémentation du modèle des objets métiers
- une couche **DomainAdapters** responsable de l'accès au donnes
- une couche **Migrations responsable** de la gestion de schémas de bases de donnes

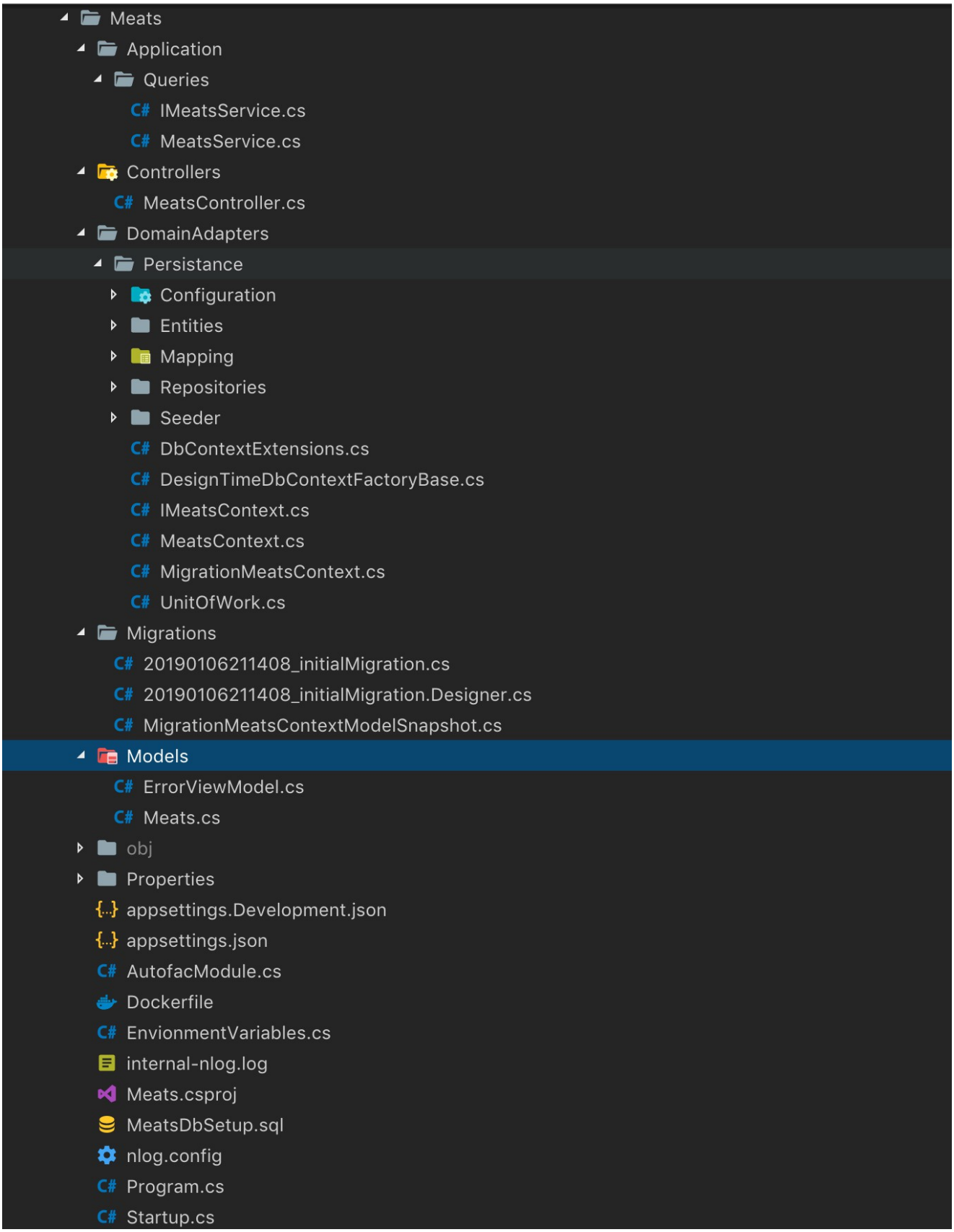
pour l'application web

- une couche **view** responsable de l'affichage de donnes
- un couche **model** pour représenter les donnes
- une couche **controller** pour faire la passerelle entre le model et la vue
- une couche **services** responsable de gérer la communication avec le serveur

5.1.2 - Structure des sources

La structuration des répertoires du projet suit la logique suivante :

pour les serveurs de micro-services :



pour l’application web :

6 - POINTS PARTICULIERS

6.1 - Gestion des logs

Les logs de microservices sont geres grace au framework nlog ; chaque log est disponible sur les dossier /var/logs/oc-pizza/ de son conteneur docker ; les conteneurs dockers sont configures de sorte que le dossier oc-pizza, corresponde a un point de montage vers un disque virtuel docker.

Les format de sortie de logs ,cest un format standard qui reponds a la plupart des critères dans le web en ce qui concerna l'analyses et lecture des logs.

6.2 - Fichiers de configuration

6.2.1 - Application web

Le fichier /oc-pizzas/Application/Views/src/config.js, continet la configuration pou rler serveur nodejs qui heberge le site web.

6.2.1.1 - Data-sources, Application

Dans le répertoire Application/DockerCompose se trouvent le fichier qui donnent accès aux différents paramétrés de configuration des micro-services.

Chaque micro-service pose un fichier de configuration pour sa base de donnes, ainsi que un fichier de configuration pour la génération de son image docker.

6.3 - Procédure de packaging / livraison

1. Démarrer une console bash;
2. Se placer au niveau du répertoire ./Application. Lancer la commande :
3. ./build.sh

   ceci aura pour effet la génération des images docker nécessaires au fonctionnement de l'application.
4. Une fois la génération des images terminé. avec la commande

   ./push.sh les images sont mise a jour sur le serveur des gestion d'images docker
5. Se connecter sur le serveur de production en ssh avec le compte administrateur.
6. Lancer le script ./run.sh. Celui -ci effectue un pull des dernières images docker disponibles sur le repository, et ensuite redémarre l'application

6.4 - Environnement de développement

Pour démarrer l’application en mode développement :

1. Démarrer une console bash;
2. Se placer au niveau du répertoire ./Application. Lancer la commande :
3. ./build.sh
4. ceci aura pour effet la génération des images docker nécessaires au fonctionnement de l'application.
5. le script ./run.sh démarre l’application en local
6. l’url d’accès c’est sera http://localhost/

7 - GLOSSAIRE

cadriciel	désigne un ensemble cohérent de composants logicielsstructurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture)
repository	En informatique, un dépôt ou référentiel (de l'anglais **repository**) est un stockage centralisé et organisé de données