OC_PIZA DOPRR

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 - DockerCompose

Ce dossier contient les fichier nescesaires pour la generation des images Dokcer , pour le lancement de l'application

3.1.1.1 - DockerCompose/env

Fichier de configuration pour le lancement de l'application sur les diferent environements.

Env, prod ,preprod

3.1.2 - Application

Dossier contenant Le Code source nescesaire pour generer les webservice, les pages html du site, ainsi que les dossier des script de build, run, et deploiment de l'application

3.1.2.1 - Services

Tous les classes c# des webservice du backend ce trouvent ici

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 repertoire src

Deja demontre dans les project precedants

4 - Architecture de Déploiement

Deja demontre dans les project precedants

4.1 - Serveur de Base de données

Description

Caractéristiques techniques (ex: Serveur Linux Debian Jessie + PostgreSQL 9.6...)

Informations importantes / points particuliers

4.2 - Serveur XXX

...

5 - Architecture logicielle

5.1 - Principes généraux

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

- **npm :** gestionaire 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 microservices:

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

pour l'application web

- une couche **view** responsible de l'affichage de donnes
- un couche **model** pour representer les donnes
- une couche **controller** pour fair ela pasarelle entre le model el la vue
- une couche **services** responsible de gerer la comunication avec le serveur

5.1.2 - Structure des sources

La structuration des répertoires du projet suit la logique suivante :

pour les serveur de microservices :

5.2 - Application Web

...

Si besoin, diagramme UML de composants pour monter les différents modules et leur inter-dépendances

5.3 - Application Xxx

...

6 - Points particuliers

6.1 - Gestion des logs

...

6.2 - Fichiers de configuration

6.2.1 - Application web

...

6.2.1.1 - Datasources

...

6.2.1.2 - Fichier xxx.yyy

...

6.2.2 - Application Xxx

...

6.3 - Ressources

...

6.4 - Environnement de développement

6.5 - Procédure de packaging / livraison

6.6 - XXX

...

7 - GLOSSAIRE
