|  |
| --- |
| **OC PIZZA**  ProJet 9  Dossier de conception technique  Version 1.0 |
| **Auteur**  Yann Hamdi |

Table des matières

1 -Versions 3

2 -Introduction 4

2.1 -Objet du document 4

2.2 -Références 4

**3- Le domaine fonctionnel**……………………………………………………………………………………………………………………………**5**

3.1-Référentiel…………………………………………………………………………………………………………………………………….**5**

3.2- Diagramme de classe, explication………………………………………………………………………………………………..6

4 -Architecture Technique 7

4.1 -Application Web 7

4.1.1 -Composants clients 8

4.1.2 -Composants produits 8

4.1.3- Composants établissements…………………………………………………………………………………………… ……..8

4.1.4- Composants commandes………………………………………………………………………………………………………..8

4.1.5-Composants types de paiements……………………………………………………………………………………………..8

4.1.6-Composants employés……………………………………………………………………………………………………………..8

4.1.7-Composants pizzas…………………………………………………………………………………………………………………..8

4.1.8-Composants stock……………………………………………………………………………………………………………………8

5 -Architecture de Déploiement 9

5.1 -Serveur de Base de données 10

5.2 -Serveur XXX 10

6 -Architecture logicielle………………………………………………………………………………………………………………………..11

6.1 -Principes généraux………………………………………………………………………………………………………………………**…11**

6.1.1 -Les couches 11

6.1.2 -Les modules 11

6.1.3 -Structure des sources 11

7 -Points particuliers 13

7.1 -Gestion des logs 13

7.2 -Fichiers de configuration 13

7.2.1 -Application web 13

7.3 -Ressources 13

7.4 -Environnement de développement 13

7.5 -Procédure de packaging / livraison 13

8-Glossaire…………………………………………………………………………………………………………………………………………..140

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| Yann Hamdi | 21/02/2021 | Création du document | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier de conception fonctionnelle de l'application OC pizza

Objectif du document est de présenter les besoins de l’utilisateur de décrire la solution qui va être implémentée pour répondre à ces besoins.

Les éléments du présent dossier découlent :

• de l’entretien réalisé avec le dirigeant de la société OC pizza du 18/09/2018.

• De l’analyse des besoins suite à cet entretien effectué par l’équipe IT EXPERTS

## Références

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

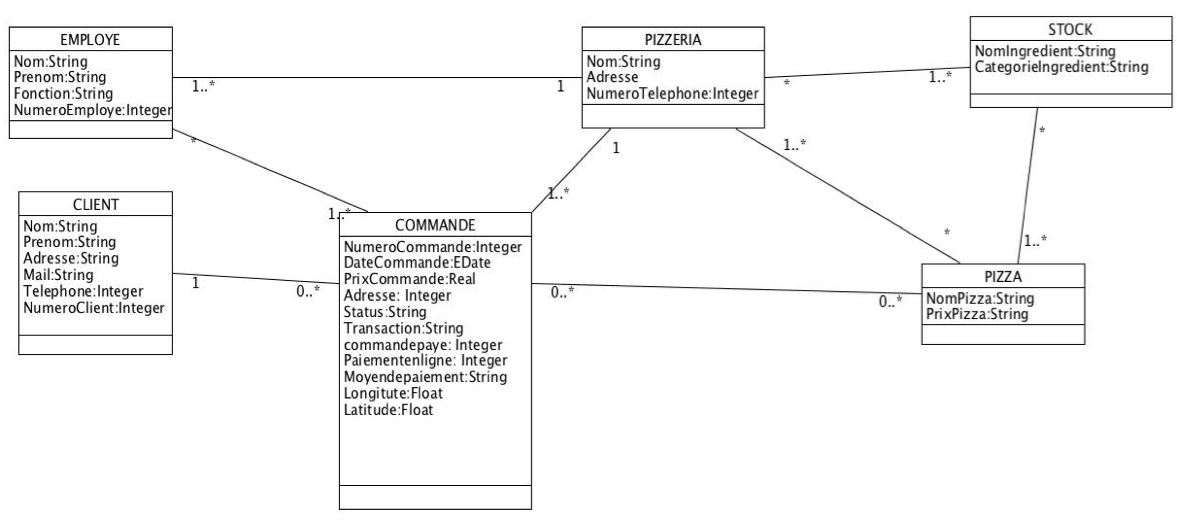
1. **DCT – 1.0** : Dossier de conception fonctionnelle
2. **DE – 1.0** : Dossier d’exploitation de l’application

# Le domaine fonctionnel

## Référentiel

Cette partie du documents va presenter l’organisation et l’utilisation des données.

Après etude, nous obtenons le diagramme de classe suivant :



## Diagramme de classe, explication

Nous avons en premier lieu la classe « EMPLOYE » qui représentera tous les employés des pizzérias, cette classe est liée avec la classe pizzeria avec une relation 1..\* et 1 puisqu’un employé ne peut travailler quand dans une seule pizzeria par contre une pizzeria peut avoir 1 ou plusieurs employés qui y travaille.

Il y a également une relation entre la classe employé et la classe commande avec une relation \*, 1..\* puisqu’il y peut y avoir plusieurs employés qui travaillent sur la même commande (livreur et pizzaiolo par exemple) et un employé peut avoir travaillé sur plusieurs commandes.

Nous avons ensuite la classe **« PIZZERIA »** qui représente les différentes pizzerias du groupe. Cette classe est reliée à la classe **« EMPLOYE »**  comme expliqué plus haut puis nous avons d’autre relation.  
Elle est ensuite reliée au stock afin d’avoir un suivi du stock des différentes pizzerias, elle est reliée par une relation \*, 1..\* puisque que le même aliment du stock peut être présent dans plusieurs pizzeria à la fois et une pizzeria a plusieurs aliments du stock.

Ensuite, nous avons une relation entre la classe **« Pizzeria »** et la classe « Commande » avec une relation 1, 1..\* puisqu’une commande ne peut appartenir qu’a une seule pizzeria et une pizzeria peut avoir plusieurs commandes.

Puis également une relation entre la classe **« PIZZA»** et la classe  
**« PIZZERIA»** ce qui correspondrait au menu par pizzeria, nous avons une relation 1..\*, \* puisque qu’une pizza peut appartenir à toutes les pizzerias et qu’une pizzeria peut avoir plusieurs pizzas au menu.

La classe « **STOCK**» correspond au stock disponible par pizzeria, elle joue le rôle de suivi du stock des aliments.  
Elle est reliée à la classe **« PIZZERIA »** comme déjà expliqué précédemment puis à la classe **« PIZZA »** puisque qu’une pizza est composé de différents ingrédients disponible dans le stock, nous avons une relation de cardinalité \*, 1..\*, effectivement un ingrédient du stock peut être présent sur plusieurs pizzas et une pizza peut avoir différents aliments par sorte de pizza.

La classe **« PIZZA »** correspond aux différentes pizzas disponibles au menu et cela par pizzeria. Elle est liée à la classe **« PIZZA »** et la classe **« STOCK »** (déjà mentionnée dans la classe pizza et la classe stock)  
Elle est également liée à la classe **« COMMANDE »** puisqu’une commande va être composée d’aucune ou plusieurs pizzas d’où la relation 0..\*, 0..\*. Et une pizza peut faire partie d’aucune commande ou bien de plusieurs commandes.

La classe **« COMMANDE »** représente le détail de l’ensemble commandes présentes pour chaque pizzeria, la relation entre la classe  
**« PIZZERIA »** et la classe **« COMMANDE »** a déjà été traitée dans la classe **« PIZZERIA ».**Nous avons une relation entre la classe **« COMMANDE »** et la classe  
**« CLIENT »** et cette relation est 1, 0..\*, évidemment, une commande ne peut appartenir qu’a un seul client à la fois tandis qu’un client peut avoir aucune commande à plusieurs commandes.

## Application XXX...

# Architecture de Déploiement

Diagramme UML de déploiement

Explication / commentaires si nécessaires...

## Serveur de Base de données

Description

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

Informations importantes / points particuliers

## Serveur XXX

...

# Architecture logicielle

## Principes généraux

Les sources et versions du projet sont gérées par **Git**, les dépendances et le packaging par **Apache Maven / Grunt / ...**.

...

### Les couches

L'architecture applicative est la suivante :

* unecouche **business** : responsable de la logique métier du composant
* unecouche **model** : implémentation du modèle des objets métiers
* …
  + …
* ...

### Les modules

Ex: modules Maven dans le cas d’application multi-module...

### Structure des sources

La structuration des répertoires du projet suit la logique suivante :

* les répertoires sources sont crées de façon à respecter la philosophie Maven (à savoir : « convention plutôt que configuration »)

racine  
 ├─ *pom.xml*  
 ├─ <moduleX>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ resources  
 ├─ <moduleY>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources  
 └─ src  
 └─ lib

* ...

## Application Web

…

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

## Application Xxx

…

# Points particuliers

## Gestion des logs

…

## Fichiers de configuration

### Application web

...

#### Datasources

...

#### Fichier xxx.yyy

...

### Application Xxx

...

## Ressources

...

## Environnement de développement

## Procédure de packaging / livraison

## XXX

…

# Glossaire

|  |  |
| --- | --- |
|  |  |
|  |  |