# OC Pizza

# Système informatique OC PIZZA

Dossier de conception technique

Version Version1.0

**Auteur**
Maryam Launois
*Analyste-programmeur*

# Table des matières


1 Versions ..... 4
2 Introduction ..... 5
2.1 Objet du document ..... 5
2.2 Références ..... 5
3 Architecture Technique ..... 6
3.1 Composants généraux ..... 6
3.1.1 Diagramme de composants ..... 6
3.1.2 Composant web app ..... 6
3.1.3 Composant BDD ..... 7
3.1.4 Composant Banque ..... 8
3.1.5 Composant Serveur web ..... 8
3.2 Application Web ..... 8
3.2.1 Modèle physique de données ..... 9
4 Architecture de Déploiement ..... 10
4.1 Diagramme de déploiement : ..... 10
5 Architecture logicielle ..... 11
5.1 Principes généraux ..... 11
5.1.1 Les couches ..... 11
5.1.2 Les modules ..... 11
5.1.3 Structure des sources ..... 11
6 Points particuliers ..... 13
6.1 Gestion des logs ..... 13
6.2 Fichiers de configuration ..... 13
6.3 Environnement de développement ..... 13
6.4 Procédure de packaging / livraison ..... 13

# 1 VERSIONS

| Auteur | Date | Description | Version |
|---|---|---|---|
| Maryam Launois | 11/05/20 | Création du document | 1.0 |

# 2 INTRODUCTION

## 2.1 Objet du document

Le présent document constitue le dossier de conception technique du système informatique de gestion centralisée des différents points de vente du client ainsi que du site de vente en ligne.

Objectif du document :

Mettre en place la conception technique de l'application du client OC Pizza.

## 2.2 Références

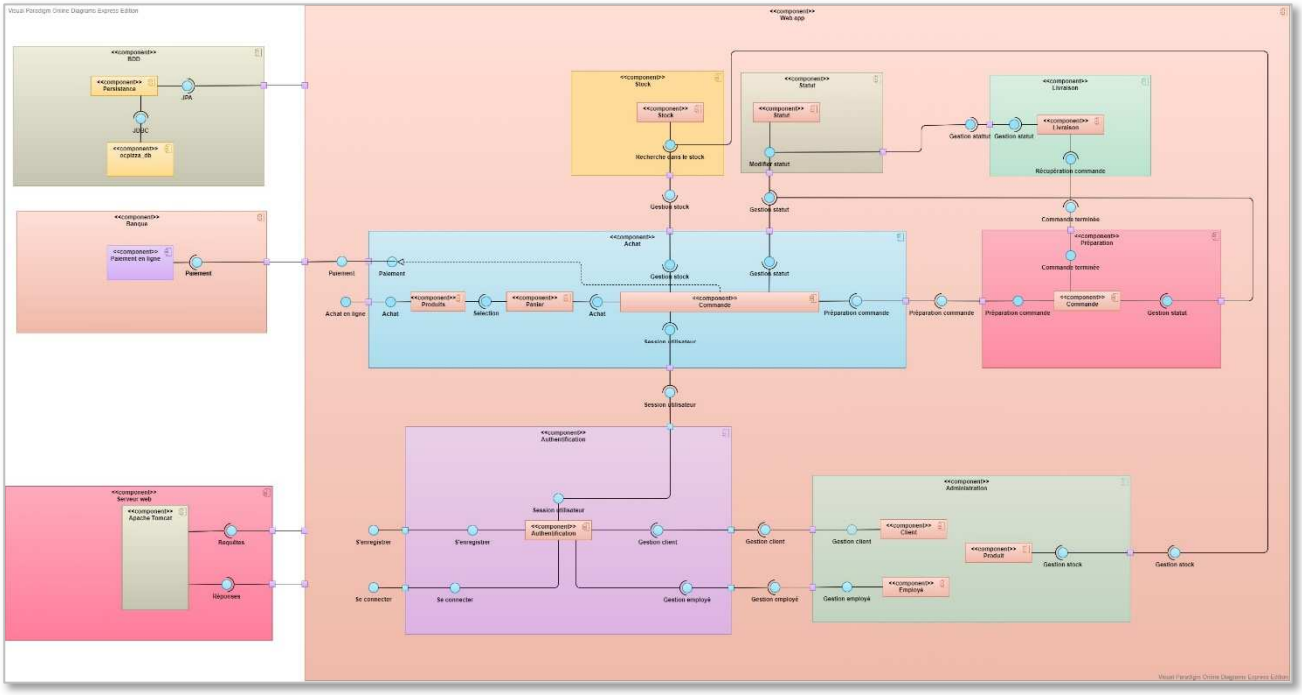
Pour de plus amples informations, se référer également aux éléments suivants :

1. **OC Pizza Système informatique** : Dossier de conception fonctionnelle
2. **OC Pizza Système informatique** : Dossier d'exploitation

# 3 ARCHITECTURE TECHNIQUE

## 3.1 Composants généraux

### 3.1.1 Diagramme de composants



### 3.1.2 Composant web app

#### 3.1.2.1 Sous composant Authentification :

Il gère la connexion et la déconnexion à l'application web mais aussi la création d'une session utilisateur.

#### 3.1.2.2 Sous composant Achat :

C'est à ce niveau, que sera gérée toute la partie achat en ligne. On y trouve les sous composants suivants :

- produits, pour la consultation des produits en ligne,
- panier,
- commande qui va être transféré au sous composant préparation.

#### *3.1.2.3 Sous composant Préparation :*

Il prend en compte toute la partie préparation de commande avec la gestion des statuts.

#### *3.1.2.4 Sous composant Livraison :*

Une fois la préparation de commande terminée, le composant livraison va récupérer la commande et finaliser le processus de livraison.

#### *3.1.2.5 Sous composant Statut :*

Ce sous composant va gérer tous les statuts des commandes.

#### *3.1.2.6 Sous composant Stock :*

Il est relié au sous composant **Achat** et au sous composant **Administration.** En effet, lors d'un achat nous consommons des produits dont il faudra mettre à jour le stock. Le sous composant **Administration**, lui, se charge d'afficher ou de remplir le stock.

#### *3.1.2.7 Sous composant Administration :*

Il est défini par les sous composants suivants :

- **Client**, pour la gestion des comptes clients,
- **Produit**, pour la gestion des produits en stock,
- **Employé**, pour la gestion des comptes employé.

### *3.1.3 Composant BDD*

#### *3.1.3.1 Sous composant Persistance :*

Il contient le sous composant **Persistance** qui va fournir le service JPA requis par les différents sous composants de l'application web. Ce service permettra la persistance des données récupérées depuis les différents sous composants du composant application web vers la BDD.

#### *3.1.3.2 Sous composant ocpizza_db :*

Il fournit le service *JDBC* pour permettre la communication du sous composant **Persistance** avec la BDD.

### *3.1.4 Composant Banque*

#### *3.1.4.1 Sous composant paiement en ligne :*

Il fournit le service *paiement* qui va permettre au composant **Application web** de payer une commande en ligne.

### *3.1.5 Composant Serveur web*

#### *3.1.5.1 Sous composant Apache Tomcat :*

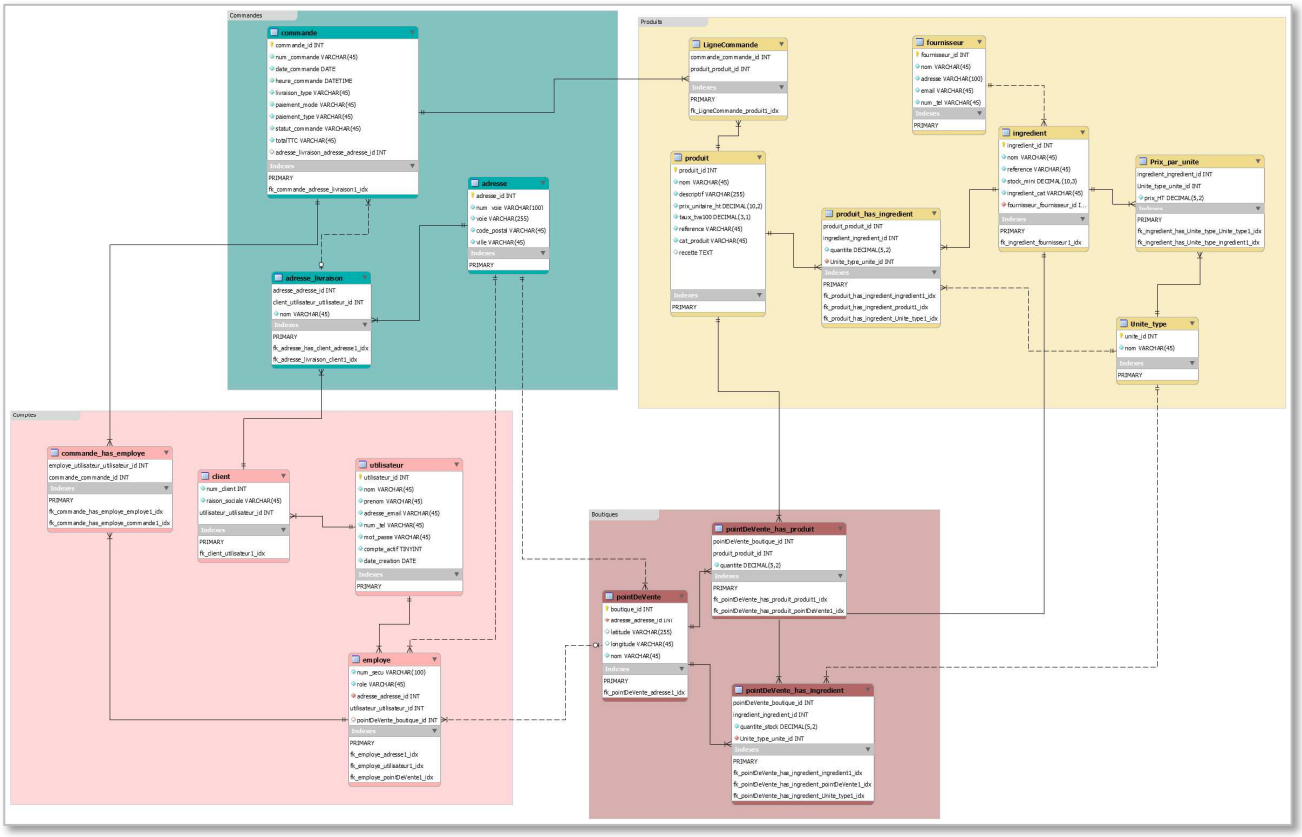
Il fournit le service ***Requêtes*** auquel se connecte l'interface du composant **Application web** pour envoyer une requête au serveur.
Il possède également une interface ***Réponses***, qui se connecte sur le service ***Réponses*** du composant **Application web** pour lui transférer la réponse à une requête.

## 3.2 Application Web

La pile technologique est la suivante :

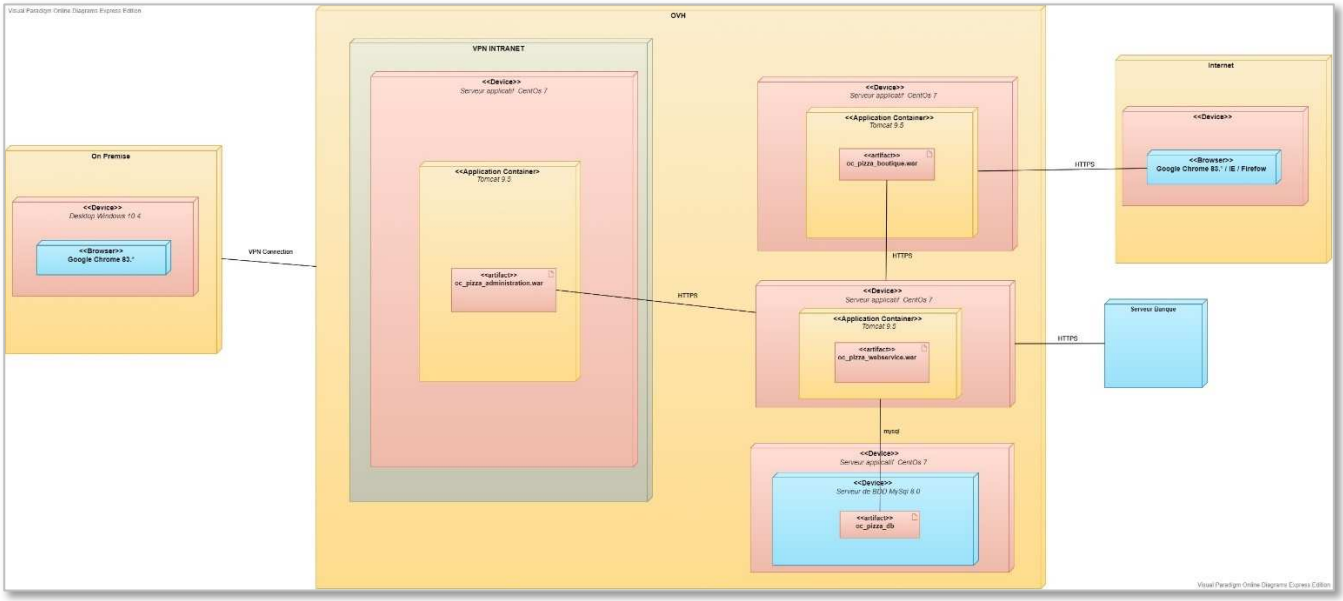
- Application **J2EE (JDK version 11)**
- Base de données **Mysql 8.0**
- Serveur d'application **Tomcat 9.5**

### 3.2.1 Modèle physique de données



# 4 ARCHITECTURE DE DEPLOIEMENT

## 4.1 Diagramme de déploiement :



# 5 ARCHITECTURE LOGICIELLE

## 5.1 Principes généraux

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

### *5.1.1 Les couches*

L'architecture applicative est la suivante :

- une couche **business** : responsable de la logique métier du composant
- une couche **model** : implémentation du modèle des objets métiers
- une couche **consumer** : responsable de la persistance en BDD
- une couche **webapp** : responsable du transfert des données à la vue, contient les controllers et les conteneurs de vue.

### *5.1.2 Les modules*

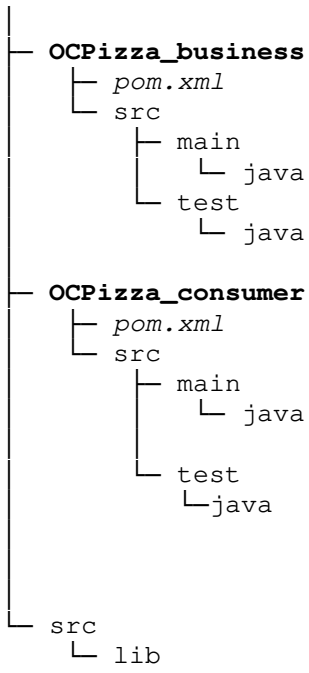
Il s'agira d'un projet Maven multi modules, un module par couche.

### *5.1.3 Structure des sources*

La structuration des répertoires du projet suit la logique suivante :

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

```
OC Pizza
  ├─ pom.xml
  ├─ OCPizza_webapp
  │    ├─ pom.xml
  │    └─ src
  │         ├─ main
  │         │    ├─ java
  │         │    └─ resources
  │         └─ test
  │              ├─ java
  │              └─ resources
  │
  ├─ OCPizza_model
  │    ├─ pom.xml
  │    └─ src
  │         ├─ main
  │         │    └─ java
  │         │
  │         └─ test
  │              └─java
  │
```


OCPizza_business
pom.xml
src
main
java
test
java
OCPizza_consumer
pom.xml
src
main
java
test
java
src
lib

# 6 POINTS PARTICULIERS

## 6.1 Gestion des logs

La gestion des logs se fera dans un fichier log.

Librairie utilisée : Log4j2

## 6.2 Fichiers de configuration

L'application sera développée à l'aide du Framework Spring Boot. Toutes les informations nécessaires à la connexion à la base de données se trouveront dans le fichier **configuration.properties**.

## 6.3 Environnement de développement

L'application a été développée avec **IntelliJ Ultimate 2020.1**

## 6.4 Procédure de packaging / livraison

Le projet est packagé à l'aide de la commande mvn clean package sous forme de war dans l'outil Jenkins. Les livrables sont récupérables depuis le dépôt Nexus. (voir dossier d'exploitation*)