Dossier de conception fonctionnelle

Version 1.1

David Barat
Dev. Python

TABLE DES MATIÈRES

1. VERSIONS

Auteur	Date	Description	Version
DBA	20/05/2020	Création du document	1.0
DBA	12/06/2020	Modifications suite soutenance	1.1

2.INTRODUCTION

2.1.Objet du document

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

Les éléments du présents dossiers découlent :

- du document de description du besoin client (OC-Pizza-description-de-notre-besoin.pdf)

3.LE DOMAINE FONCTIONNEL

3.1. Référentiel

Voici ci-dessous le diagramme de classe.

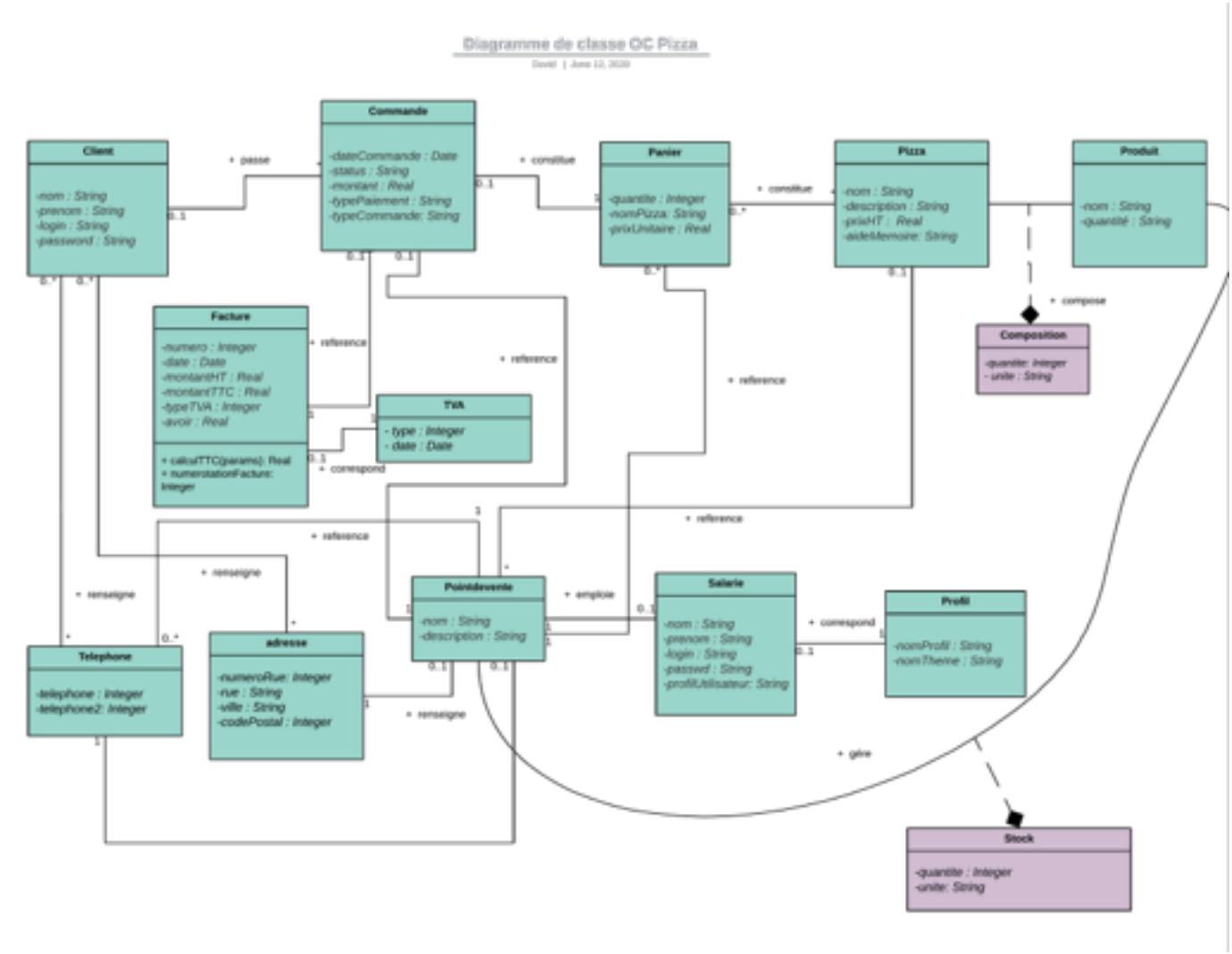

fig.1 Diagramme de classe OCPizza

Dans notre domaine fonctionnel, deux classes de composition ont été créées :

- une pour les stocks,
- une pour la composition des pizzas.

La classe **stock** va permettre d'avoir une vision claire des stocks par points de vente et non

plus par point de vente.

La classe pizza permet de centraliser les informations de composition d'une pizza, afin de ne pas multiplier les données.

Afin de répondre aux besoins exprimés par les dirigeants d'***OCPizza***, j'ai crée une table **point de vente** qui contient les informations du point de vente et j'ai crée une foreign key sur l'id du point de vente que j'ai inséré dans les tables stock, commande, facture afin d'avoir une vision par point de vente. L'attribut **status** pour les commandes a été créée afin de donner une vision claire des commandes pour les livreurs mais pour les clients.

Création d'une table de paramétrage pour la TVA en cas de changement de celle-ci.

Création d'une table profil afin de différencier l'interface utilisateur suivant le profil de connexion.

Pour les mots de passe j'ai utilisé le sha2 afin de les « d'hasher », l'attribut password est stocké en char(64). En effet le sha1 et le md5 sont plus facilement « crackable ».

Concernant la table facture, afin de laisser la possibilité au client de gérer leur numérotation de facture, la colonne numéro de facture a été créée en primary key sans « auto increment ». La numérotation sera traitée via une méthode, ce sujet sera traité ultérieurement.

3.1.1. Règles de gestion

Tableau récapitulatif des multiplicités:

Relation entre les classes	attribut sur lequel porte la relation	multiplicité
classe facture et classe commande	numero_facture	un à un
classe TVA et classe facture	id_TVA	un à un
classe commande et classe panier	id_commande	un à un
classe pizza et classe composition	id_pizza	un à un
classe pizza et classe panier	id_pizza	un à plusieurs
classe client et classe adresse	id_client	un à plusieurs
classe client et classe telephone	id_client	un à plusieurs
classe lient et classe commande	id_client	un à plusieurs
classe produit et classe composition	id_produit	un à plusieurs
classe produit et classe stock	id_produit	un à un
classe point_de_vente et classe telephone	id_point_de_vente	un à plusieurs

Relation entre les classes	attribut sur lequel porte la relation	multiplicité
classe point_de_vente et classe adresse	id_point_de_vente	un à un
classe point_de_vente et classe stock	id_point_de_vente	un à plusieurs
classe point_de_vente et classe salarie	id_point_de_vente	un à un
classe point_de_vente et classe facture	id_point_de_vente	un à un
classe point_de_vente et classe commande	id_point_de_vente	un à un
classe profil_utilisateur et classe salarie	id_profil	un à un

A noter que la clé primaire a été très utilisée et ce afin de permettre aux responsables d'***OCPizza*** d'avoir une vision par point de vente mais aussi de manière globale; et ce que ce soit en terme de chiffre d'affaires, de stocks, de commandes, de factures.

4. ARCHITECTURE TECHNIQUE

4.1. Application Web OCPizza

La pile logicielle est la suivante :

- Application **Flask** / **Python 3.6,**
- Serveur web **NGinx 1.19,**
- Serveur WSGI **Gunicorn 20.0.4,**
- Mysql **8.0.18**

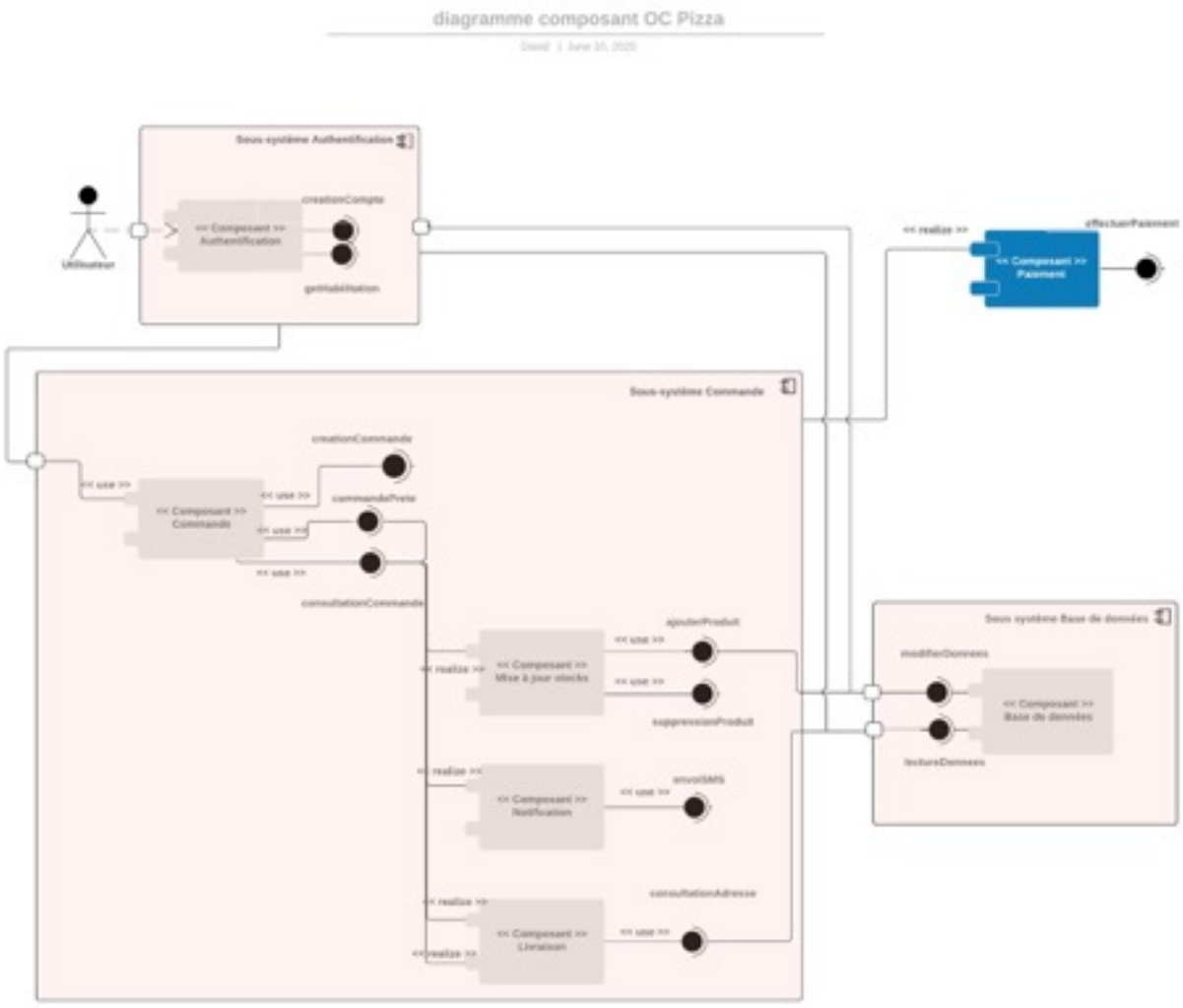

fig.2 Diagramme de composant OCPizza

4.1.1.Package Authentification

4.1.1.1.Composant Authentification

Le composant authentification offre deux interfaces :

- creationCompte qui permet à l'utilisateur de créer son compte à partir d'un formulaire,
- getHabilitation qui permet à la session de l'utilisateur de récupérer son profil (droit de navigation, menu supplémentaire)

4.1.2.Package Gestion Commande

4.1.2.1.Composant Commande

Le composant Commande offre quatre interfaces :

- *creationCommande* qui permet à l'utilisateur que ce soit un client ou un salarié d'***OCPizza*** de créer une commande à partir d'une liste de pizza. Cette interface est un pré-requis pour le composant externe **Paiement**.
- *consultationCommande* qui permet de visualiser une commande passée quelque soit son statut.
- *miseAjourStatutCommande* qui permet de mettre à jour le statut d'une commande.
- *commandePrete* qui est un déclencheur pour plusieurs composant. Cette interface est un pré-requis pour les composants **Mise à jour Stocks**, **Notification** et **Livraison**.

4.1.2.2.Composant Notification

Le composant Notification offre une interface:

- *envoiSMS* qui permet à l'utilisateur que ce soit un client ou un salarié d'***OCPizza*** de créer une commande à partir d'une liste de pizza. Cette interface est un pré-requis pour le composant externe **Paiement**.

4.1.2.3.Composant Livraison

Le composant Livraison offre une interface:

- *consultationAdresse* qui permet aux livreurs de consulter l'adresse du client une fois la commande prête.

4.1.2.Package Gestion Commerciale

4.1.2.1.Composant Mise à jour stock

Le composant Mise à jour stock offre deux interfaces:

- *ajouterProduit* qui permet à un utilisateur d'ajouter un produit dans la table qui référence tous les produits, cette interface est aussi utilisée par l'application ***OCPizza***,
- *suppressionProduit* qui permet à un utilisateur de supprimer un produit dans la table qui référence tous les produits, cette interface est aussi utilisée par l'application ***OCPizza***.

4.1.3.Package Base de données

Le composant Base de données offre deux interfaces:

- *modifierDonnees* qui offre la possibilité à l'application ***OCPizza*** de modifier un enregistrement dans la base de données.
- *lectureDonnees* qui offre la possibilité à l'application ***OCPizza*** de modifier un enregistrement dans la base de données.

5. ARCHITECTURE DE DÉPLOIEMENT

Voici ci-dessous le diagramme de déploiement :

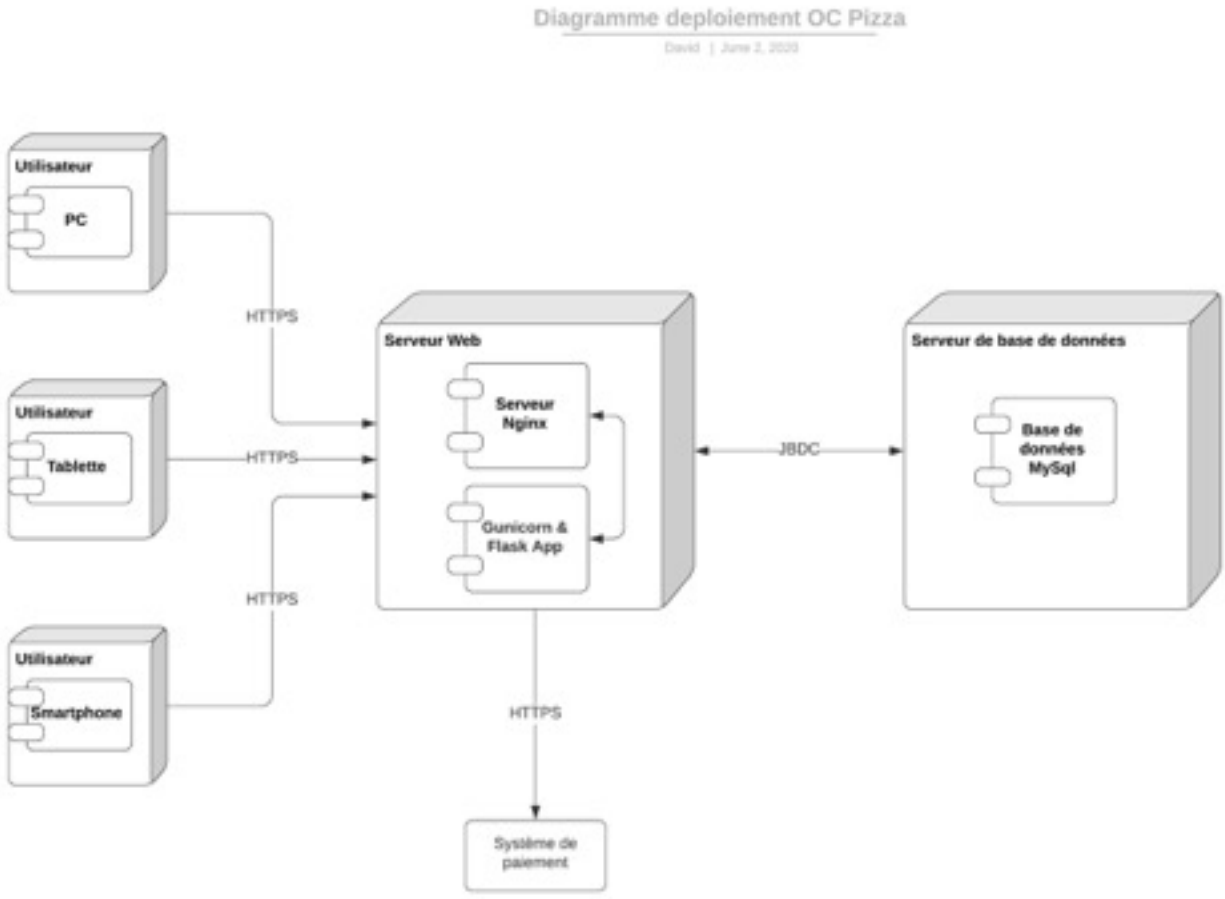

fig.3 Diagramme de déploiement OCPizza

L'architecture proposée à ***OCPizza*** est une architecture deux tiers dont voici les deux éléments :

5.1.Serveur de Base de données

Le serveur de base de données va héberger une base de données Mysql. Ce serveur est un serveur Linux distribution Debian 10 (Buster).

Produits installés :

- Mysql Community Server

  Le character setting utilisé est UTF8 et la collation utf8mb4_0900_ai_ci.

5.2.Serveur Web

Le serveur web va héberger le serveur web nginx, le serveur d'application Gunicorn ainsi que le code python (flask). Ce serveur est aussi un serveur Linux distribution Debian 10 (Buster).

Produits installés :

- Nginx,
- Python,
- Flask,
- Gunicorn.

7.GLOSSAIRE