OC Pizza

Système informatique

Gestion centralisée d'un réseau de pizzerias

Document de spécifications techniques

TABLE DES MATIERES

Introduction

Ce document présente les spécifications techniques du groupe de pizzeria OC Pizza. Groupe spécialisé dans la vente de pizza livrées ou à emporter. A ce jour il est constitué de cinq points de ventes et un minimum de trois points de vente supplémentaires vont ouvrir avant la fin de l'année.

Le commanditaire souhaite un système informatique plus performant, plus adapté aux besoins du groupe en pleine évolution et qui gère de manière centralisée toutes les points de vente.

Le système doit être plus efficaces dans la gestion et le traitement des commandes.
Le système doit proposer un suivis en temps réel des commandes et du stock d'ingrédients.
Le client doit pouvoir passer une commande en ligne, via un appel téléphonique ou sur un point de vente.
Le règlement doit pouvoir se faire en ligne ou à la réception de la commande.
Le client doit pouvoir suivre ses commandes en lignes, les modifier ou les annuler.
Le système doit également proposer aux pizzaiolos les différentes recettes nécessaires à la confection des pizzas.

Ce document détails les différents composants du domaine fonctionnel. Ainsi que les différentes relations qui lient ses composants.

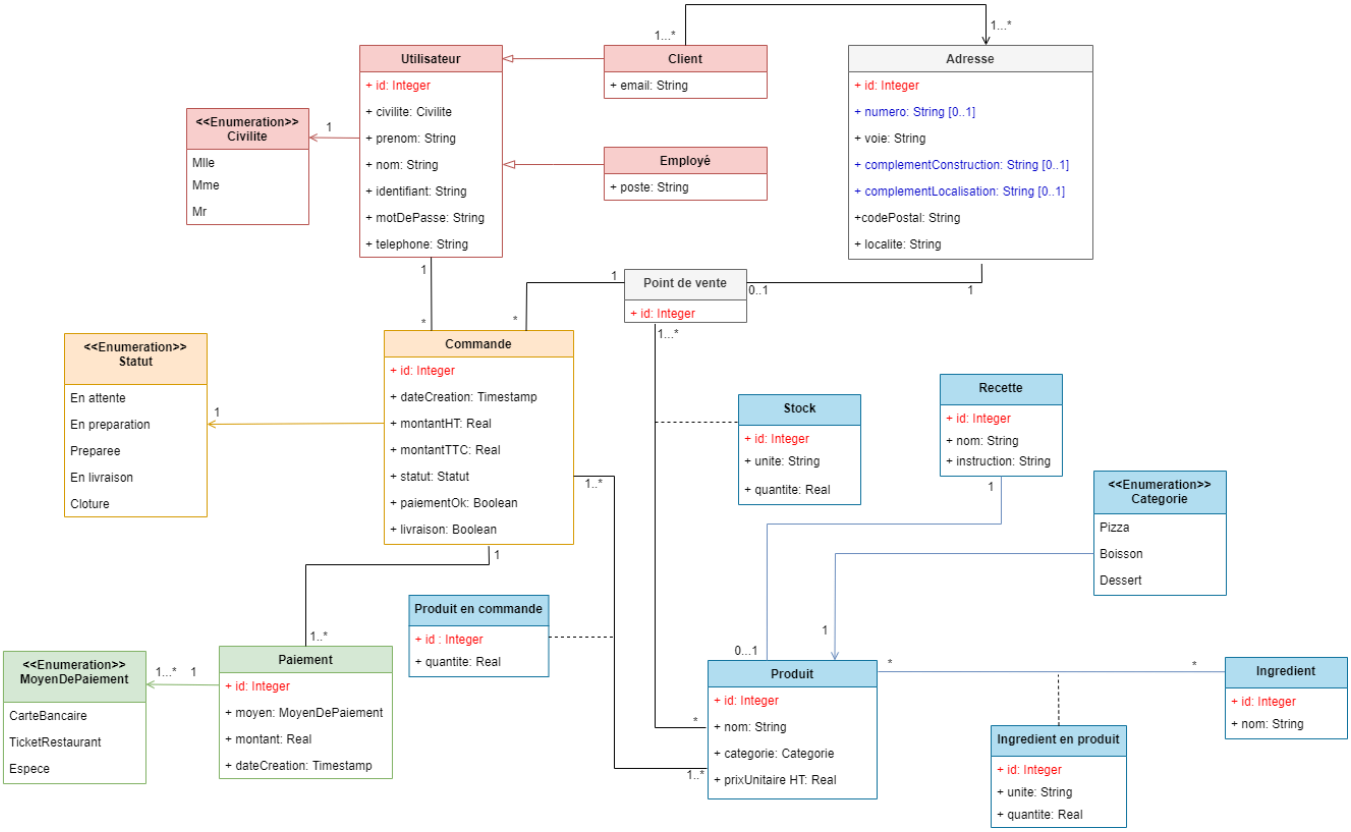
Le domaine fonctionnel

Description

Le domaine fonctionnel d' OC Pizza définit l'ensemble des classes utiles au système. Le domaine est modélisé via le langage UML et un diagramme de classe qui met en évidence les relations entre les différentes classes. Ces classes vont servir de support à la création du modèle physique de données qui sera utilisé pour créer la base de données qui va regrouper les informations du système que l'on veut enregistrer.

Le domaine fonctionnel est constitué de l'ensemble des composants suivants : ceux de la partie « Utilisateur », ceux de la partie « Produit », de la partie « Commande » et de la partie « Paiement ». Il contient aussi deux composants généraux : un composant « Adresse » et un deuxième «Point de vente ».

Diagramme de classe d'OC Pizza



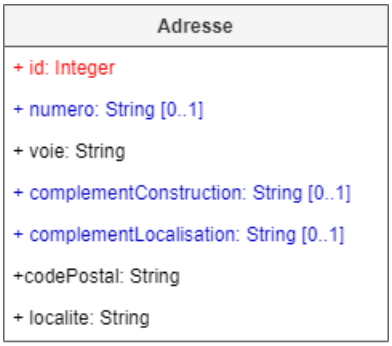
Les composants généraux

Adresse

Cette classe est composée des attributs des adresses des clients, des adresses des points de vente et des adresses des commandes.

Un même client peut avoir plusieurs adresses de livraison (domicile, travail..) c'est une relation one-to-many. Une même adresse peut correspondre à différents clients notamment dans le cas où, dans une famille, plusieurs membre de cette famille possède un compte client dans le système (relation one-to-many).

Les champs à renseigner sont les champs qui correspondent aux informations d'une adresse postale classique : le **numéro** et le nom de la **voie**, un complément **d'information** sur la **construction** (cage d'escalier, numéro de bâtiment, code d'accès…) et la **localisation** (nom de résidence…) si nécessaire. Le **code postal** et le nom de la **localité**.



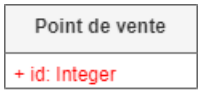
Il existe une relation one-to-one avec la classe **Point de vente** : chaque point de vente possède une adresse.

Point de vente

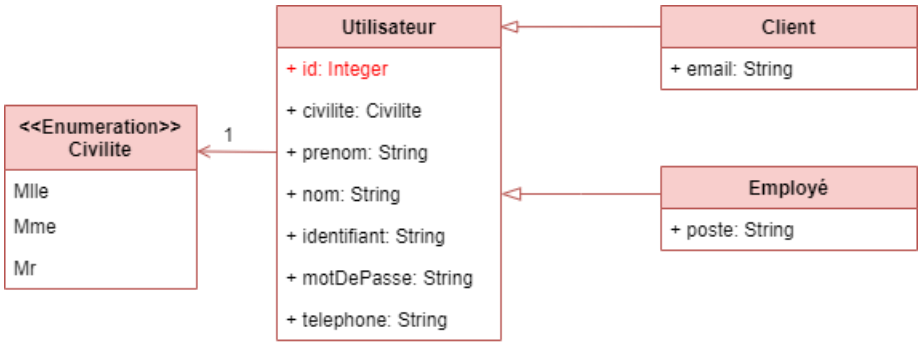
La classe **Point de vente** représente les différentes boutiques du groupe. L'attribut **id** permet de différencier chacun d'entre eux. Cette classe est associées aux classe **Adresse**, **Commande**, **Stock** et **Produit**. Chaque point de vente est associé à un stock.

La relation avec la classe **Commande** est de type one-to-many. Il y a aucune ou plusieurs commandes passées dans un point de vente. Et chaque commande est reliée à un seule point-de-vente.

Idem pour la relation avec la classe **Produit** (cardinalité 0 ..*) , il peut y avoir aucun ou une multitude de produits sur un point de vente.



Les composants de la partie « utilisateur »



Utilisateur

La classe **Utilisateur** est la classe mère qui est associée aux classes filles **Client** et **Employé.** Cette classe regroupe les attributs communs à chaque utilisateur. On y trouve sa **civilité**, son **nom**, son **prénom**, son **identifiant**, son **mot de passe** et son numéro de **téléphone**.

Civilité

La classe **Civilité** est une énumération qui propose les choix possibles concernant l'utilisateur : **Mlle**, **Mme** ou **Mr**.

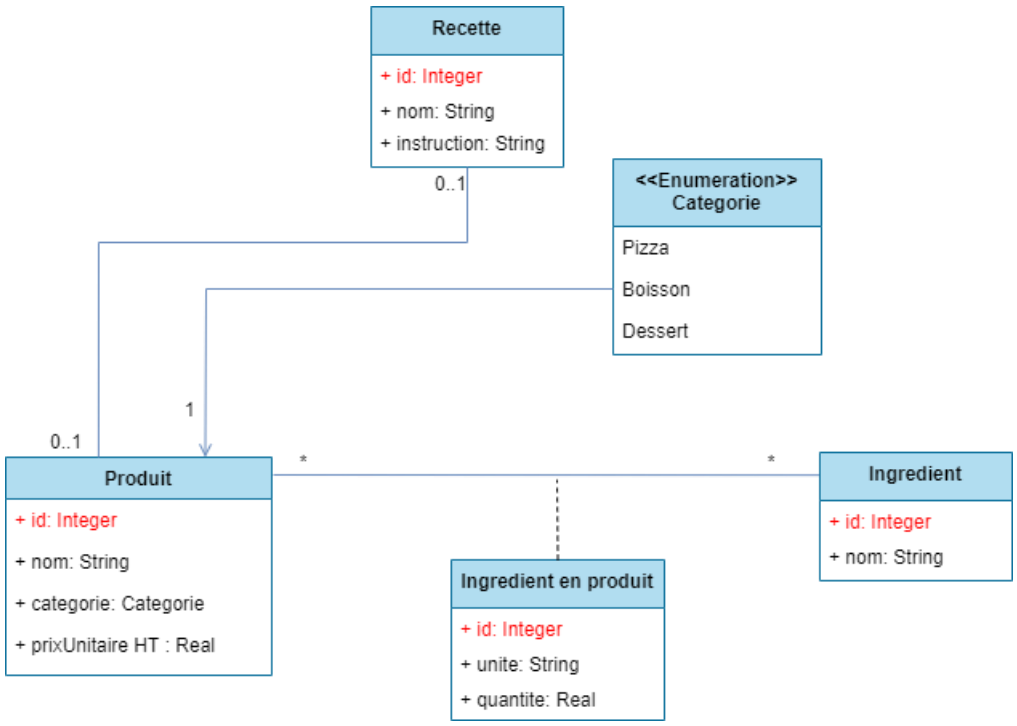
Client

La classe **Client** contient le champ **email** qui est utiliser pour faire parvenir au client les informations relatives aux commandes effectuées.

Employé

La classe **Employé** précise le poste qu'occupe celui-ci dans l'entreprise pour lui affecter des droits d'accès dans le système.

Les composants de la partie « produit »



Produit

Chaque produit appartient à une catégorie. Une recette peut être associé à aucun ou un seul produit et réciproquement. Un produit à un attribut **id** qui sert d'identifiant unique, un attribut **nom** et un **prix unitaire HT**.

La relation entre la classe **Produit** et la classe **Ingrédient** est de type many-to-many. Un ingrédient peut constituer aucun, un ou plusieurs produit et inversement un produit peut être constitué d'aucun, un ou plusieurs ingrédients.

Recette

Une recette a un **id** unique, un **nom** qui facilite sa recherche en cas de modification, et un champ **instruction** qui contient le texte qui liste les opérations à suivre.

Ingrédient

Un ingrédient à un **id** unique et un **nom**. On peut avoir aucun ou plusieurs ingrédients par produit. Et un ingrédient peut composer aucun ou plusieurs produits.

Ingrédient en produit

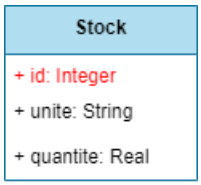
Cette classe est une classe d'association. Elle ajoute un champ **id**, un champ **unité** et un champ **quantité** à l'association entre les tables **Produit** et **Ingrédient**. Cela indique la quantité d'ingrédient qui compose le produit. C'est une relation many-to-many.

Catégorie

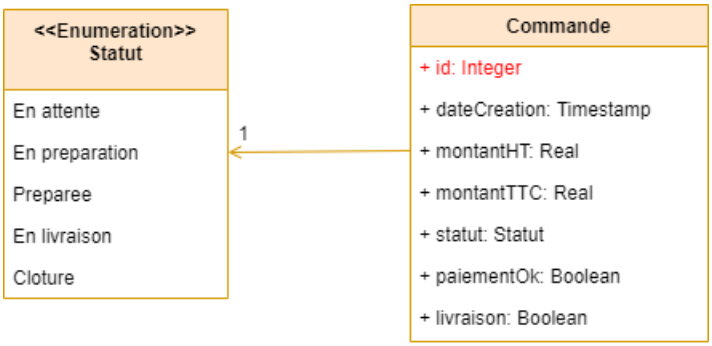
Cette classe permet de préciser si le produit est une **pizza**, une **boisson** ou un **dessert**.

Stock

C'est une classe d'association qui permet d'ajouter les attributs **unité** et **quantité** à l'association entre les tables **Produit** et **Point de vente**. C'est une relation many-to-many.



Les composants de la partie « commande »



Commande

La classe **Commande** permet d'enregistrer la **date** et **l'heure** de création d'une commande, les **montants HT** et **TTC** et le **statut** de celle-ci. On précise aussi si la commande est **réglée** et s'il s'agit d'une commande qui doit être **livrée** ou à retirer sur place.

Il existe une relation entre la classe **Commande** et la classe **Utilisateur**. Une commande correspond à un utilisateur et un utilisateur peut passer aucune, une ou une infinité de commande.

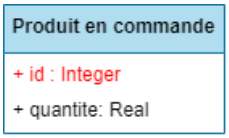
Une deuxième relation se déroule avec la classe **Paiement**. Une relation de type one-to-many. A chaque commande correspond un ou plusieurs paiements.

La relation avec le composant **Point de vente** a déjà été évoqué précédemment

Statut

Cette classe précise l'état de la commande si celle-ci est **en attente**, **en préparation**, **préparée**, **en livraison** ou **clôturée**.

Produit en commande



C'est une classe d'association. Elle ajoute un champ **id** et un champ **quantité** à l'association entre les tables **Produit** et **Commande**. Elle précise la quantité d'un type de produit qui compose une commande.

Les composants de la partie « paiement »

Paiement

Ce composant contient un champ **id**, un champ **moyen de paiement**, le **montant** à payer, une **date** et une **heure** de création. Un paiement peut être effectué avec différents moyen de paiement. La relation entre la classe **Moyen de paiement** et la classe **Paiement** est de type one-to-many.

Moyen de Paiement

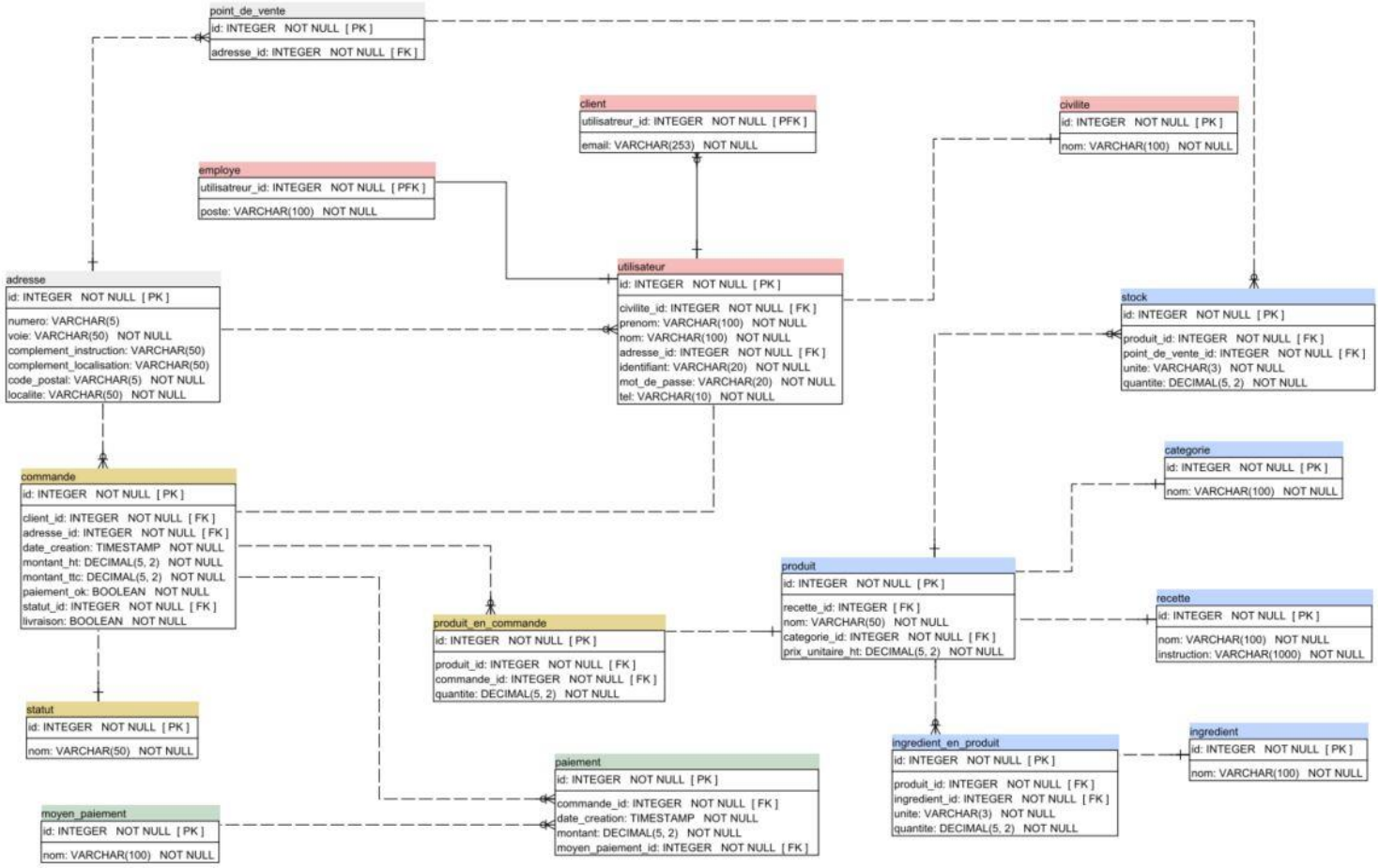
Cette classe contient les champs des différents moyen de paiement : **carte bancaire**, **ticket restaurant** et **espèces**.

Les différents types de données utilisées

Boolean	pour les valeurs booléenne
Integer	pour les entiers relatifs
Real	pour les nombres réels
String	pour les chaînes de caractères
Timestamp	pour une date et l'heure

Le model physique de données est la dernière étape de l'analyse du système. Il modélise la base de données relationnelle d'OC Pizza. Il décrit les tables et les différents liens entre elles. On trouve les types de données des colonnes qui composent chaque table ainsi que les clés primaires et étrangères.

Model physique de données d'Oc Pizza



Adresse

La table **adresse** regroupe les informations des adresses des clients et des points de vente. Sa clé primaire est **id** et est auto incrémentée. Seul les colonnes **numéro**, **complément d'instruction** et **complément de localisation** acceptent NULL comme valeur.

Point de vente

La colonne **id** est la clé primaire et est auto incrémentée. La colonne **adresse_id** est une clé étrangère qui a pour valeur la clé primaire de la colonne **id** de la table **Adresse** qui lui correspond.

Utilisateur

Son **id** est la clé primaire et est auto incrémentée. Chacune de ses colonnes ne peut être NULL. La colonne **civilite_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Civilité**. Et la colonne **adresse_id** est une clé étrangère qui a pour valeur la clé primaire de la colonne **id** de la table **Adresse** qui lui correspond.

Civilité

Son **id** est la clé primaire et est auto incrémentée. Chacune de ses colonnes ne peut être NULL.

Client et employé

Les colonnes **utilisateur_id** sont des clés étrangères qui se réfèrent à l'**id** correspondant de la table **utilisateur**. Chacune des colonnes ne peut être NULL.

Produit

Son **id** est la clé primaire et est auto incrémentée. Chacune de ses colonnes ne peut être NULL sauf la colonne **recette_id** car un produit n'est pas obligatoirement rattaché à une recette. La colonne **recette_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Recette**. Et la colonne **categorie_id** est une clé étrangère qui a pour valeur la clé primaire de la colonne **id** de la table **Catégorie** qui lui correspond.

Catégorie, recette et ingrédient

Les colonnes **id** sont les clés primaires et sont auto incrémentées. Chacune des colonnes ne peut être NULL.

Stock

Son **id** est la clé primaire et est auto incrémentée. La colonne **produit_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Produit**. La colonne **point_de_vente_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Point de vente**. Chacune des colonnes ne peut être NULL.

Ingrédient en produit

Son **id** est la clé primaire et est auto incrémentée. La colonne **produit_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Produit**. La colonne **ingrédient_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Ingrédient**. Chacune des colonnes ne peut être NULL.

Commande

Son **id** est la clé primaire et est auto incrémentée. La colonne **client_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Utilisateur**. La colonne **adresse_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Recette**. La colonne **statut_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Statut**. Chacune des colonnes ne peut être NULL.

Produit en commande

Son **id** est la clé primaire et est auto incrémentée. La colonne **produit_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Produit**. La colonne **commande_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Commande**. Chacune des colonnes ne peut être NULL.

Paiement

Son **id** est la clé primaire et est auto incrémentée. La colonne **commande_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Commande**. La colonne **moyen_de_paiement_id** est une clé étrangère qui a pour valeur l'**id** correspondant de la table **Moyen de paiement**. Chacune des colonnes ne peut être NULL.

Statut et moyen de paiement

Les colonnes **id** sont les clés primaires et sont auto incrémentées. Chacune des colonnes ne peut être NULL.

Les différents types de données utilisées

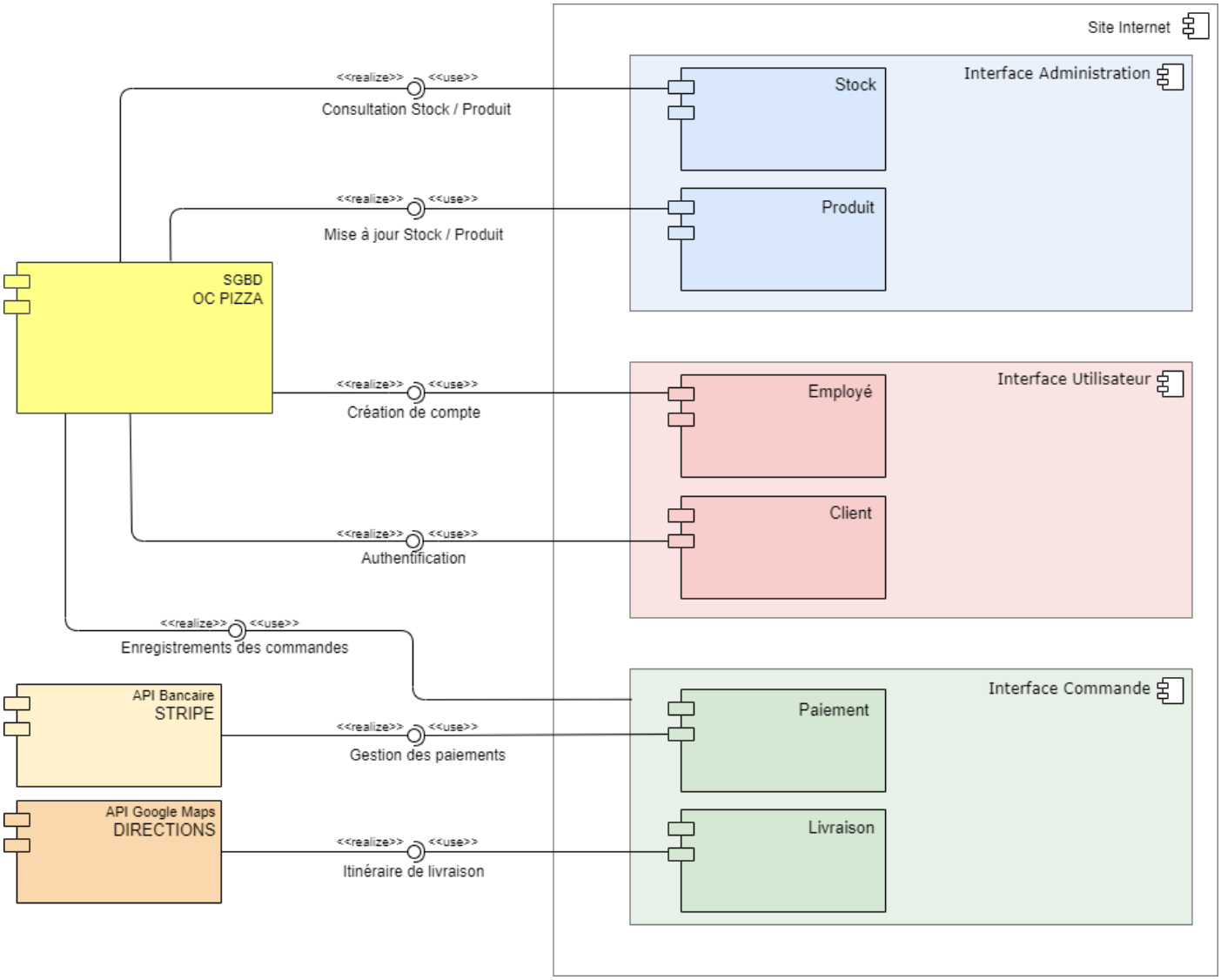
BOOLEAN	pour les valeurs booléenne
INTEGER	pour les entiers relatifs (taille de 4 octets)
DECIMAL(p,s)	pour les nombres décimaux (precision, scale)
VARCHAR(x)	Taille de texte variable (x caractères maximum)
TIMESTAMP	pour une date et l'heure

Les composants externes

En UML, un composant est un élément logiciel remplaçable et réutilisable qui fournit ou reçoit un service bien précis. Les composants fournissent des services via des interfaces.
On distingue deux types d'interfaces : les interfaces requises qui fournissent un service au composant et dont le composant a besoin pour fonctionner. Et les interfaces fournies pour lesquels le composant fourni lui-même un service.

Le diagramme de composant

Le diagramme de composants fait partie des diagrammes structuraux d'UML. Il permet de représenter les différents éléments logiciels (composants) du système et leurs dépendances (relations qui les lient).



Trois composants externes interviennent dans le système :

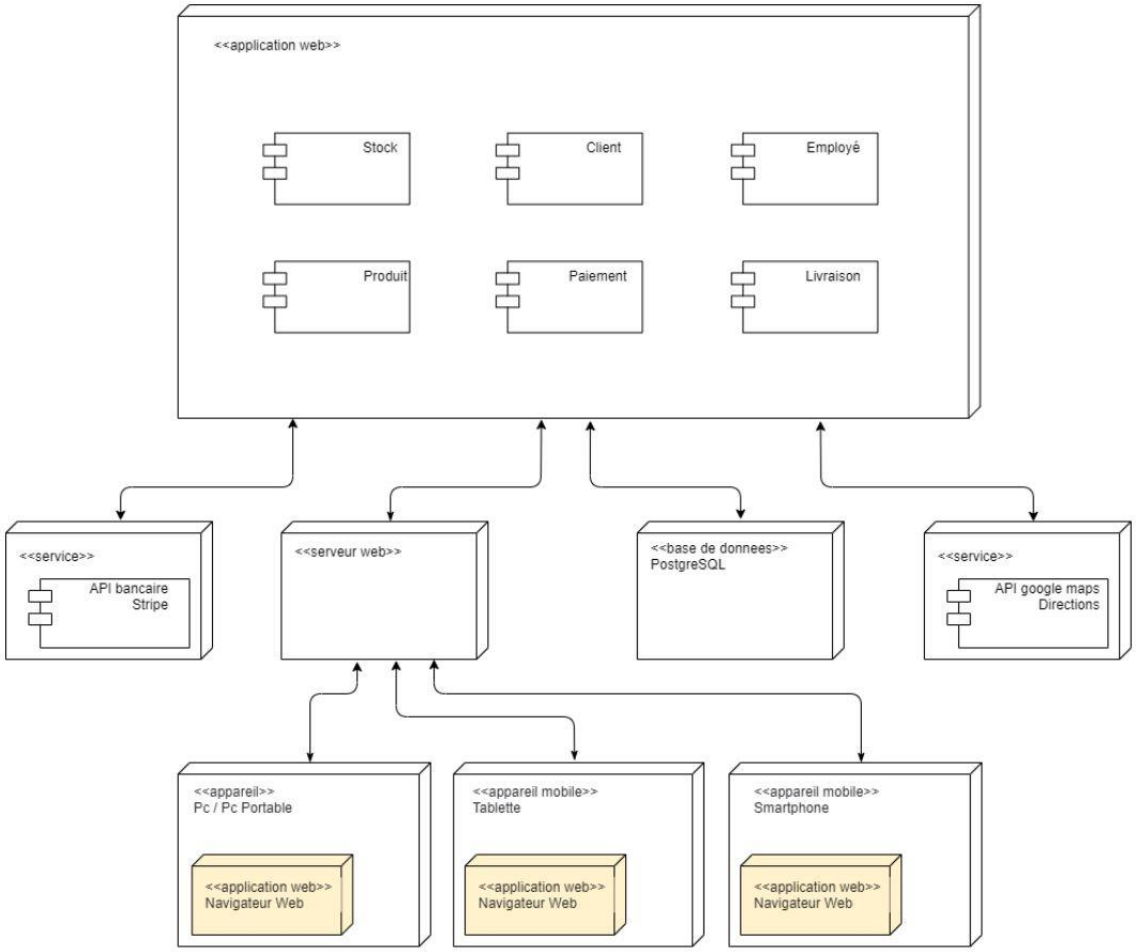
L'API Directions de Google Maps est le composant utilisé pour calculer le parcours et l'itinéraire que le livreur doit emprunter pour les livraisons à domicile.

L'API STRIPE est le système bancaire utilisé pour que les clients puissent régler leurs commandes de manière sécurisée.

La base de données OC Pizza va enregistrer et sauvegarder les informations concernant les commandes, les règlements, les comptes utilisateurs et les informations relatives au stock de produit et d'ingrédients.

Le diagramme de déploiement

Le diagramme de déploiement décrit le déploiement physique des informations générées par le logiciel sur des composants matériels. Les nœuds (les boîtes en trois dimensions) représentent les composants du système qu'ils soient logiciels ou matériels.



L'application web est reliée au service bancaire, au service Directions de Google Maps et à la base de données PostgreSQL et au serveur web.

Les utilisateurs accèdent au système à partir d'un Smartphone, d'une tablette, d'un pc ou d'un pc portable.