

Dossier de spécifications techniques

Projet : OC Pizza

Sommaire

- Contexte.....3
- 1. Description du domaine fonctionnel.....4-11
 - 1.1 introduction au modèle fonctionnel.....4
 - 1.2 Identification des classes.....5-10
 - 1.3 Introduction au Diagramme de classe.....10
 - 1.4 Diagramme de classe.....11
- 2. Les Composants.....12-20
 - 2.1 Introduction au diagramme de composants.....12
 - 2.2 Description des composants.....13-19
 - 2.3 Diagramme de composants.....20
- 3. Le Déploiement.....21-25
 - 3.1 Introduction au diagramme de déploiement.....21
 - 3.2 Description des nœuds.....22-24
 - 3.3 Diagramme de déploiement.....25
- 4. La Base de données.....26-35
 - 4.1 Introduction au Modèle Physique de Données.....26
 - 4.2 Description des tables.....27-34
 - 4.3 Modèle Physique de données.....35

CONTEXTE

« OC Pizza » est un jeune groupe de pizzeria en plein essor et spécialisé dans les pizzas livrées ou à emporter. Il compte déjà 5 points de vente et prévoit d'en ouvrir au moins 3 de plus d'ici la fin de l'année. Un des responsables du groupe a pris contact avec vous afin de mettre en place un système informatique, déployé dans toutes ses pizzerias et qui lui permettrait notamment :

- d'être plus efficace dans la gestion des commandes, de leur réception à leur livraison en passant par leur préparation ;
- de suivre en temps réel les commandes passées et en préparation ;
- de suivre en temps réel le stock d'ingrédients restants pour savoir quelles pizzas sont encore réalisables ;
- de proposer un site Internet pour que les clients puissent :

passer leurs commandes, en plus de la prise de commande par téléphone ou sur place
payer en ligne leur commande s'ils le souhaitent – sinon, ils paieront directement à la livraison

modifier ou annuler leur commande tant que celle-ci n'a pas été préparée

- de proposer un aide mémoire aux pizzaïolos indiquant la recette de chaque pizza
- d'informer ou notifier les clients sur l'état de leur commande

Le client a déjà fait une petite prospection et les logiciels existants qu'il a pu trouver ne lui conviennent pas.

1. Description du modèle fonctionnel

1.1 Introduction au modèle fonctionnel.

Pour modéliser un domaine fonctionnel nous avons besoin d'identifier les différentes entités nécessaires et les associations entre celle-ci.

Ex : Dans notre système nous avons des acteurs qui vont effectuer des actions sur une commande, donc nous allons créer une classe général Utilisateur et une classe spécialisée pour chaque rôle (Client, Commercial, Livreur et pizzaïolo) ainsi à l'aide d'une association entre un acteur et la classe commande chaque acteur pourra effectuer une action différente sur une ou plusieurs commandes en fonction des limites de l'association.

Identification des entités

Pour pouvoir identifier les entités nous allons avoir besoin de savoir qu'elle est l'utilité du système et qu'elles en sont ses utilisations. Le dossier de spécifications fonctionnelles répond à ces deux question grâce à l'identification des acteurs et aux descriptions des cas d'utilisations.

Identification des associations

Toujours à l'aide du dossier de spécifications fonctionnelles nous allons pouvoir identifier les Associations entre les différentes classes.

Pour poser les limites de chaque Association il faut connaître la nature de l'association entre les deux classes.

Ex : Un Client va pouvoir effectuer autant de commandes qu'il souhaite à partir de son compte mais une commande ne peut être effectuer que par un et seulement un client.

Donc nous allons avoir une Association un à plusieurs entre la classe Client et Commande.



Description

Dans la suite du dossier nous allons identifier chaque classe ainsi que leurs associations avec les autres classes du système et nous termineront par la modélisation complète du diagramme de classe.

1.2 Identification des classes.

1.2.1 Classe User.

Nom : User

Description : La classe User contient les informations de base d'un utilisateur.

Association

Address : Association plusieurs à un (0..* à 1) avec la classe Address qui va contenir l'adresse de l'utilisateur.

Authenticate : Association un à un (1 à 1) avec la classe Authenticate qui contient le mot de passe d'un utilisateur. La suppression d'un utilisateur engendre la suppression de son mot de passe.

Héritage

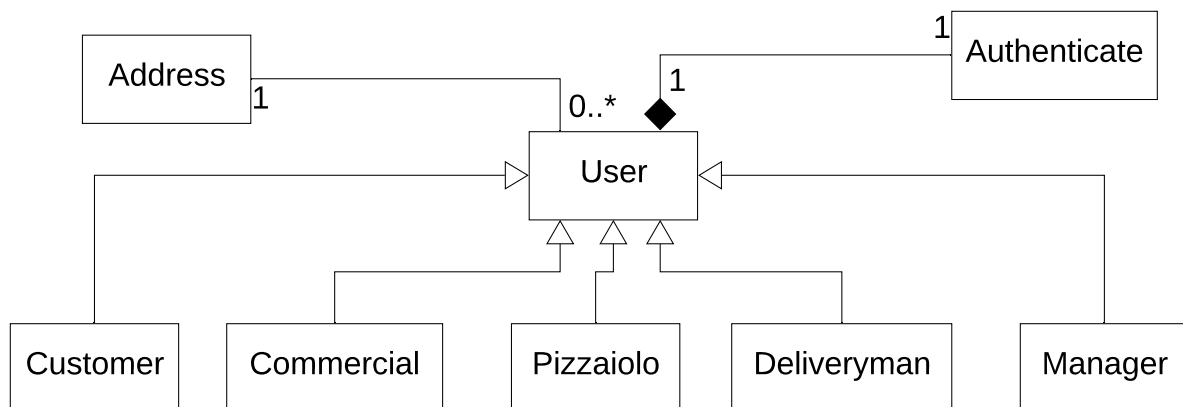
Customer : Utilisateur client de OC Pizza qui effectue des commandes.

Commercial : Employé qui gère les commandes des clients.

Deliveryman : Livreur qui livre les commandes aux clients.

Pizzaiolo : Le pizzaiolo prépare une ou plusieurs commandes.

Manager : Responsable qui gère la gestion d'un point de vente.



1.2.2 Classe Order.

Nom : Order

Description : La classe Order représente la commande d'un client.

Association

Customer : Client de la commande, un client peut avoir zéro ou plusieurs commandes mais une commande a seulement un client.

Commercial : Commerciale qui effectue une commande pour un client, un commercial peut avoir effectué zéro ou plusieurs commandes pour un client mais une commande est effectuée par seulement un commercial.

Deliveryman : Livreur de la commande, un livreur livre zéro ou plusieurs commandes mais une commande peut être livré par seulement un livreur.

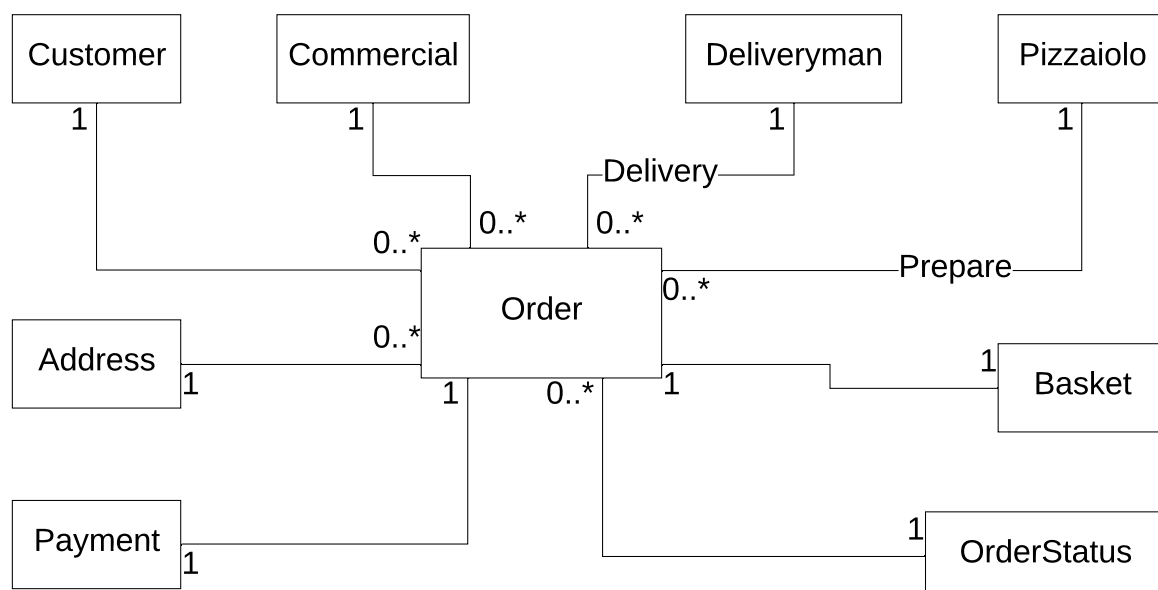
Pizzaiolo : Le pizzaiolo prépare zéro ou plusieurs commandes et une commande est préparée par un pizzaiolo.

Address : Adresse de livraison de la commande, une commande dispose d'une seule adresse de livraison.

Payment : Paiement de la commande, Le paiement de la commande peut être effectué seulement une fois et un paiement est effectué pour seulement une commande.

OrderStatus : état de la commande (Awaiting prepare, Under prepare, Awaiting delivery, Under delivery, Under modification, Canceled, Completed), un état dispose de zéro ou plusieurs commandes mais une commande peut avoir seulement un état.

Basket : Panier de la commande, une commande dispose d'un seul panier et un panier dispose d'une seule commande.



1.2.3 Classe Basket.

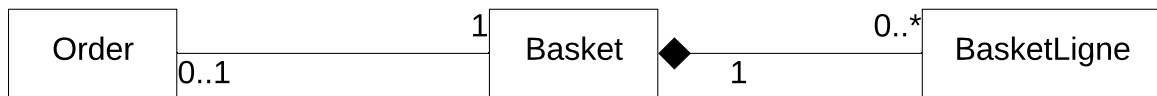
Nom : Basket

Description : La classe Basket représente le panier d'un client.

Association

Order : La commande du panier, une commande a un panier et un panier a zéro ou une commande.

BasketLigne : Une ligne du panier, un panier peut avoir zéro ou plusieurs lignes et une ligne a seulement un panier. La suppression d'un panier doit engendrer la suppression de toutes les lignes en Association avec celui-ci.



1.2.4 Classe BasketLigne.

Nom : BasketLigne

Description : La classe BasketLigne indique une pizza et sa quantité que l'on souhaite commander.

Association

Basket : Panier qui contient zéro ou plusieurs lignes, une ligne de panier dépend de seulement un panier.

Pizza : Produit que l'on souhaite commander, une ligne de panier contient seulement une pizza mais une pizza a zéro ou plusieurs lignes.



1.2.5 Classe Pizza.

Nom : Pizza

Description : La classe Pizza représente une pizza.

Association

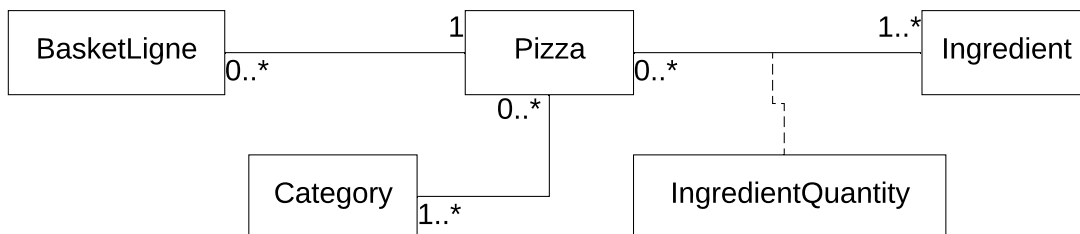
BasketLigne : Ligne de panier qui contient la pizza que le client veut acheter ainsi que sa quantité. Une pizza a zéro ou plusieurs lignes de panier et une ligne de panier dispose de seulement une pizza.

Ingredient : Représente chaque ingrédient d'une pizza. Une pizza a un ou plusieurs ingrédients et un ingrédient a zéro ou plusieurs pizzas.

Category : Catégorie d'une pizza (Cheeses, Meats, Fishes, Fruits). Une pizza a une ou plusieurs catégories et une catégorie a zéro ou plusieurs pizzas.

Interface

IngredientQuantity : Cette interface permet de représenter la quantité d'ingrédient qu'une pizza utilise.



1.2.6 Classe Ingredient.

Nom : Ingrédient

Description : La classe Ingrédient représente un ingrédient nécessaire à la fabrication d'une pizza.

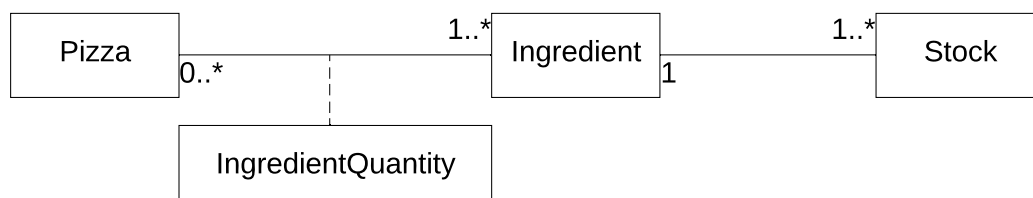
Association

Pizza : Pizza qui utilise des ingrédients, une pizza a un ou plusieurs ingrédients et un ingrédient a zéro ou plusieurs pizzas.

Stock : Le stock qui inventorie les ingrédients et leur quantité, un stock inventorie un ingrédient et un ingrédient est inventorié dans un ou plusieurs stocks.

Interface

IngredientQuantity : Cette interface permet de représenter la quantité d'ingrédient qu'une pizza utilise.



1.2.7 Classe Stock.

Nom : Stock

Description : La classe Stock représente un ingrédient et sa quantité disponible.

Association

Ingredient : Ingrédient stocké, un stock représente un ingrédient et un ingrédient est représenté dans un ou plusieurs stocks.

SaleStore : Point de vente du stock, un point de vente dispose de zéro ou plusieurs stocks et un stock dépend d'un point de vente.



1.2.8 Classe SaleStore.

Nom : SaleStore

Description : La classe SaleStore représente un point de vente du groupe OC Pizza.

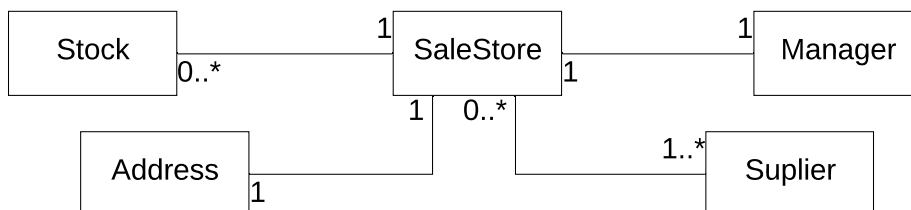
Association

Stock : Stock du point de vente, un point de vente dispose de zéro ou plusieurs stocks et un stock dépend d'un point de vente.

Address : Adresse du point de vente, un point de vente dispose d'une adresse et une adresse référence un point de vente.

Manager : Responsable du point de vente, un point de vente dispose d'un responsable et un responsable représente un point de vente.

Suplier : Fournisseur du point de vente, un point de vente dispose d'un ou plusieurs fournisseurs et un fournisseur fournit zéro ou plusieurs points de ventes.



1.2.9 Classe Suplier.

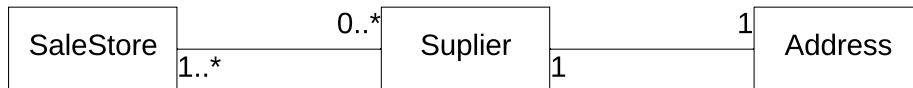
Classe : Suplier

Description : La classe Suplier représente un fournisseur du groupe OC Pizza.

Association

SaleStore : Client du fournisseur, un point de vente dispose d'un ou plusieurs fournisseurs et un fournisseur fournit zéro ou plusieurs points de ventes.

Address : Adresse du fournisseur, un fournisseur dispose d'une adresse et une adresse référence un fournisseur.



1.3 Introduction au diagramme de classe.

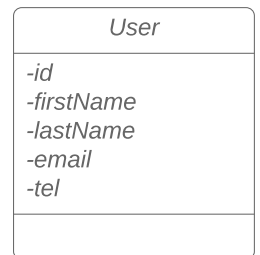
Le diagramme de classe est la représentation graphique du domaine fonctionnel.

Les classes et leurs associations entre elles sont représentée à l'aide du langage UML (Unified Modeling Language, traduisez "langage de modélisation objet unifié").

Les attributs

Une classe contient les attributs que l'on souhaite connaître.

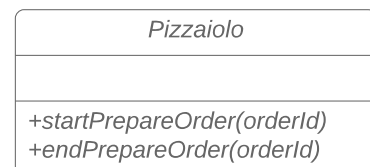
Ex : La classe Utilisateur contient l'id pour l'identification de l'utilisateur ainsi que son nom, prénom, mail et numéro de téléphone.



Les fonctions

Pour pouvoir indiquer les différentes fonctions d'une classe nous allons les représenter dans la partie inférieure de la classe.

Ex : Un pizzaiolo a pour but premier de préparer une commande, donc nous allons avoir besoin d'une fonction de préparation d'une commande ainsi qu'une seconde pour notifier les commandes prêtes pour la livraison.



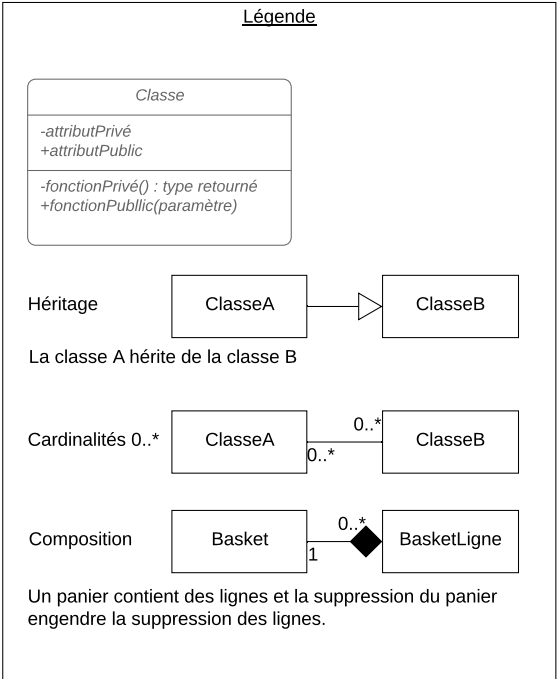
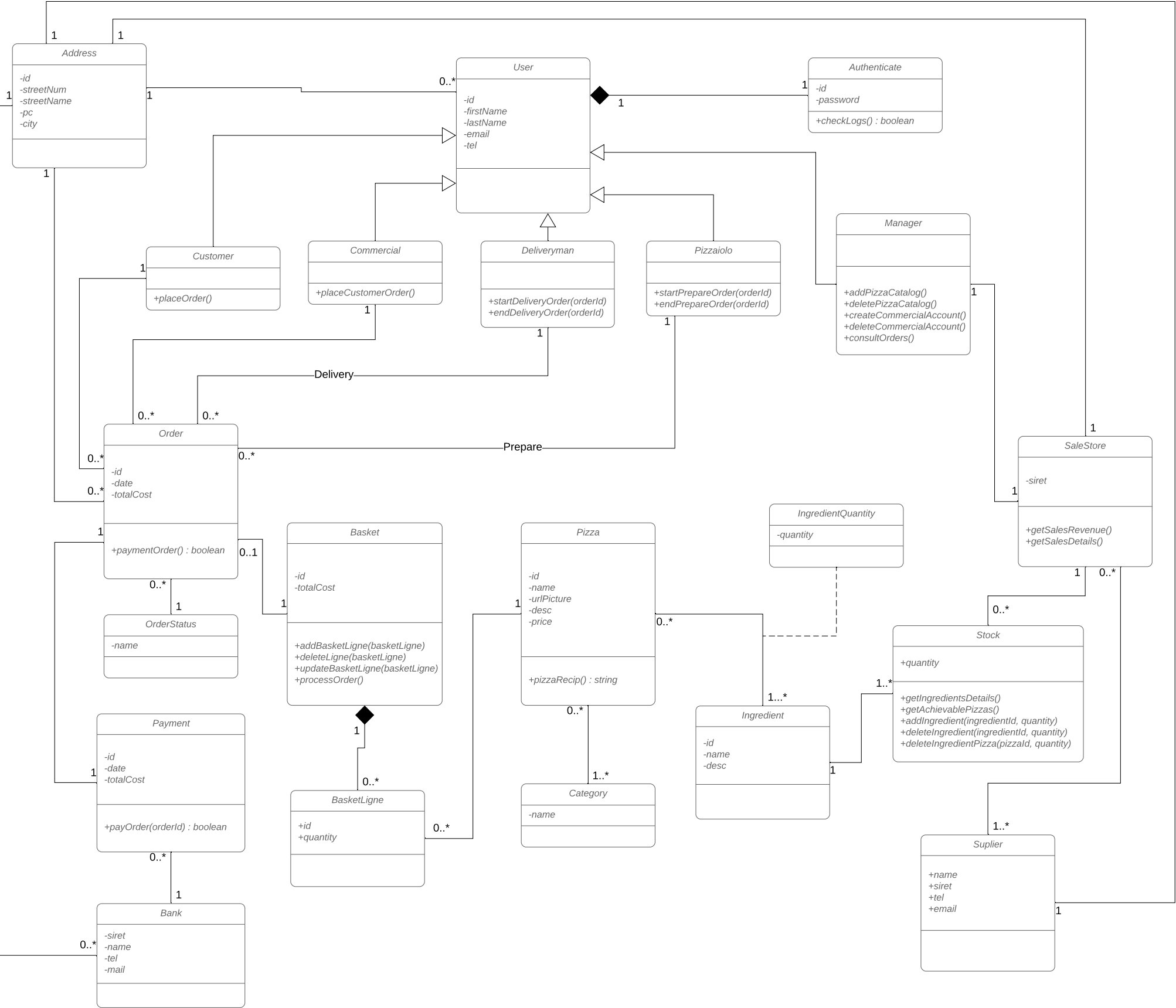
startPrepareOrder(orderId) : Change le statut de la commande pour "En préparation";

endPrepareOrder(orderId) : Change le statut de la commande pour "En attente de livraison";

Les associations

Les associations entre les classes suivent le même principe que les schémas précédent.

1.4 Diagramme de classe.

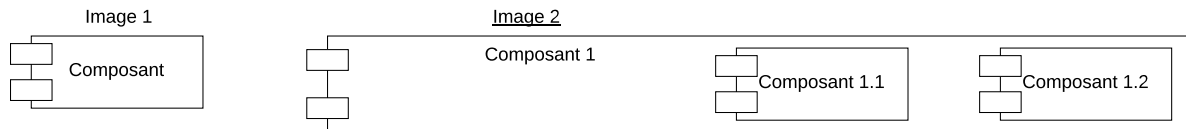


2. LES COMPOSANTS

2.1 Introduction au diagramme de composant.

Le diagramme de composant est un diagramme avec un niveau d'abstraction plus élevé que le diagramme de classe. Il permet de représenter les différentes parties d'un système, leurs dépendances entre celle-ci et identifier les composants réutilisables.

Les composants



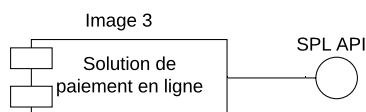
Un composant n'est rien de spécifique, il peut être de nature très différente d'un autre.

Ex de composant : Logiciel, API, back office, fichiers sources, librairies, base de données...

Il est représenté (Image 1) sous forme de rectangle avec deux autres rectangles sur la ligne verticale gauche.

Un composant peut être complexe et englober d'autres composants (Image 2).

Les Interfaces

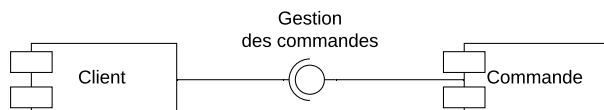


Un composant peut fournir des interfaces afin de permettre à d'autres composants de communiquer avec lui.

Ex : une solution de paiement en ligne fournit une API (interface) pour être utilisée par d'autres programmes.

L'interface est graphiquement représentée par un cercle relié au composant (Image 3).

Les dépendances



Un composant peut dépendre d'une interface pour pouvoir fonctionner

Ex : Le composant Client a pour but premier de passer des commandes donc il dépend de l'interface du composant Commande.

La représentation graphique d'une dépendance est un demi-cercle relié au composant (image 4).

2.2 Description des composants.

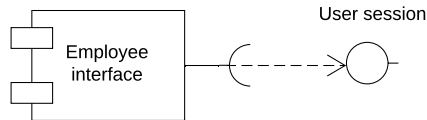
2.2.1 Composant Employee interface.

Nom : Employee interface

Description : Le composant Employee interface représente l'interface d'un employé du groupe OC Pizza.

Dépendances

User session : interface permettant l'authentification d'un employé.



2.2.2 Composant Customer interface.

Nom : Customer interface

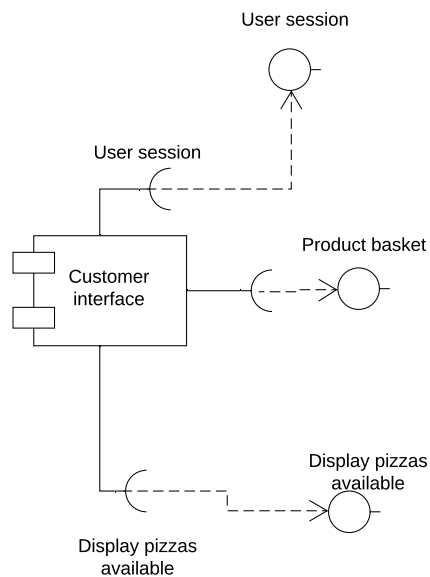
Description : Interface d'un client ou d'un visiteur (client non authentifié).

Dépendances

User session : interface permettant l'authentification d'un client.

Product basket : Panier de l'utilisateur rattacher à sa session.

Display pizzas available : permet l'affichage des pizzas disponibles en fonction des stocks.



2.2.1 Composant WebStore.

Nom : WebStore

Description : Le composant WebStore représente la logique de l'interface web de l'application OC Pizza.

Interfaces

User session : interface permettant l'authentification d'un utilisateur.

Product basket : Panier de l'utilisateur rattacher à sa session.

Display pizzas available : interface d'affichage des pizzas disponibles en fonction des stocks.

Dépendances

Manage connection : Gestion du compte utilisateur après l'authentification.

Create order : Démarrer une commande.

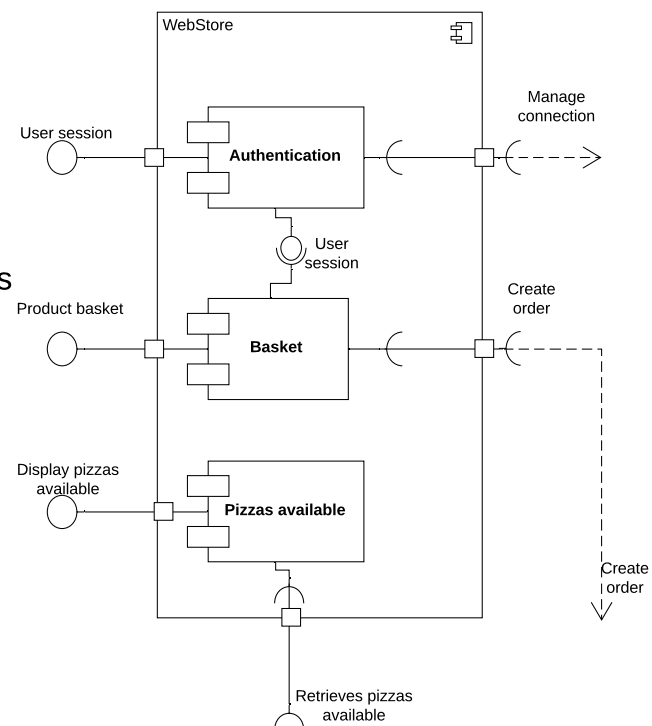
Retrieves pizzas available : récupérer les pizzas disponibles en fonction des stocks.

Sous composants

Authentification : Gestion de l'authentification des utilisateurs sur le système.

Basket : Panier d'un utilisateur relia à sa session.

Pizzas available : Pizzas disponible à la vente.



2.2.2 Composant Authentification.

Nom : Authentification

Description : Le composant gère l'authentification d'un utilisateur sur le système.

Interfaces

User session : interface permettant l'authentification d'un utilisateur.

Dépendances

Manage connection : Gestion du compte utilisateur après l'authentification.

2.2.3 Composant Basket.

Nom : Basket

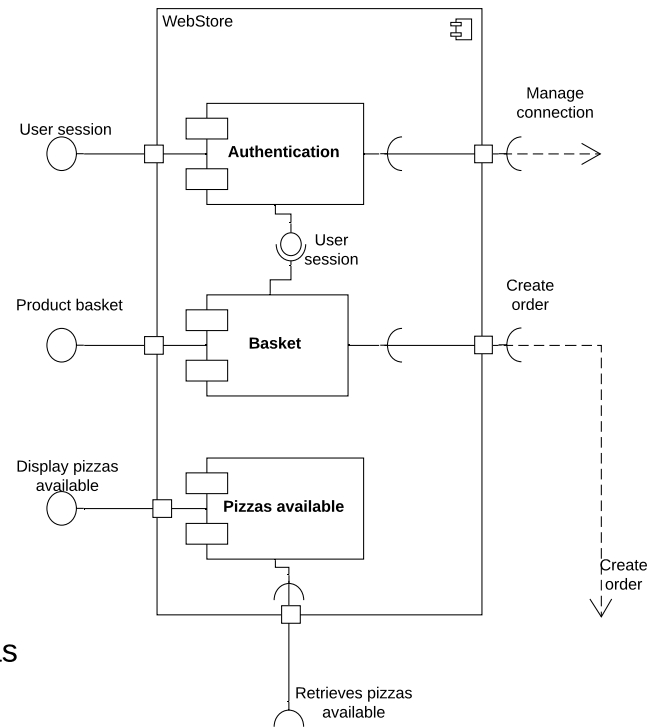
Description : Le composant "Basket" représente un panier contenant les produits d'un utilisateur.

Interfaces

Product basket : Panier de l'utilisateur rattacher à sa session.

Dépendances

Create order : Démarrer une commande à partir du panier.



2.2.4 Composant Pizza available.

Nom : Pizza available

Description : Le composant "Pizza available" affiche les pizzas disponibles à la production en fonction des stocks.

Interfaces

Display pizzas available : interface d'affichage des pizzas disponibles en fonction des stocks.

Dépendances

Retrieves pizzas available : récupérer les pizzas disponibles en fonction des stocks.

2.2.5 Composant Pizza Accounting.

Nom : Accounting

Description : Le composant Accounting représente la partie comptabilité de l'application.

Interfaces

Manage connection : interface de gestion du compte utilisateur.

Create order : Créer une commande.

Dépendances

Manage payments : Paiement d'une commande terminée.

Manage stock : Gestion du stock.

Sous composants

Account : Compte d'un utilisateur.

Customer : Client et visiteur (client non authentifié).

Commercial : Commercial passant commande pour un client.

Pizzaiolo : Pizzaiolo effectuant la préparation d'une commande.

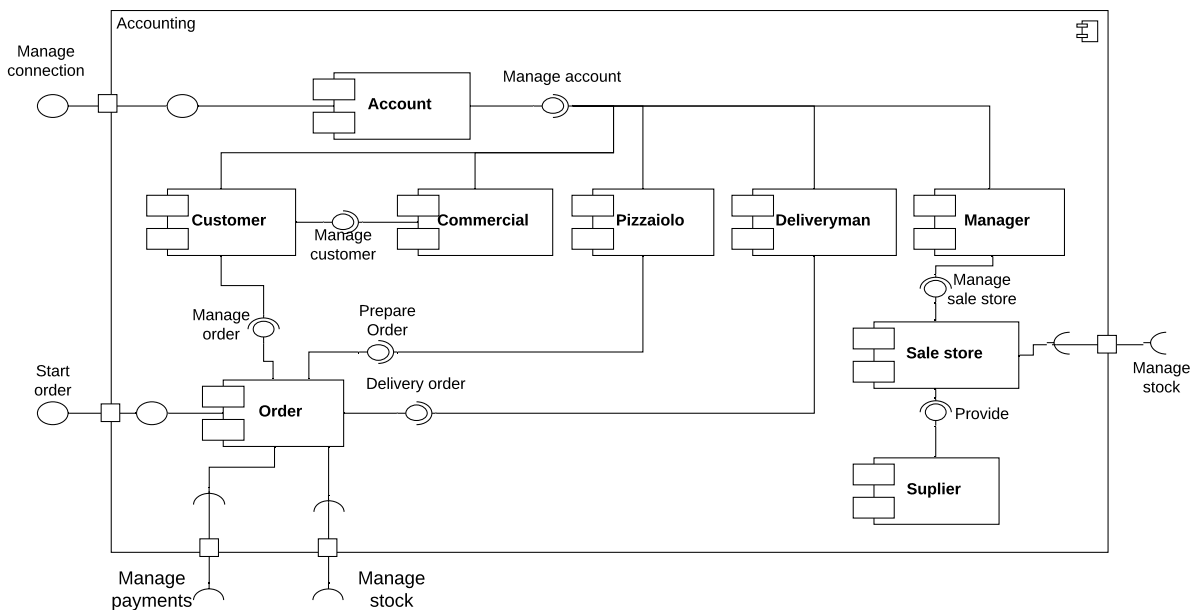
Deliveryman : Livreur livrant les commandes clients.

Manager : Responsable d'un point de vente du groupe OC Pizza.

Order : Commande client.

Sale store : Point de vente du groupe OC Pizza.

Suplier : Fournisseur du point de vente du groupe OC Pizza.



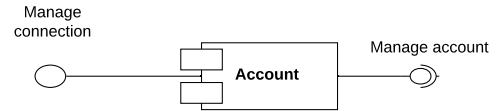
2.2.6 Composant Account.

Nom : Account.

Description : Compte d'un utilisateur.

Interfaces

Manage account : Gestion du compte utilisateur.



2.2.7 Composant Customer.

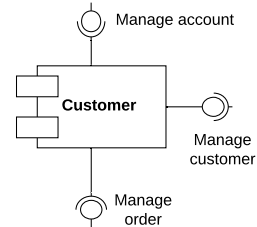
Nom : Customer.

Description : Utilisateur client pouvant effectuer des commandes, il représente aussi un visiteur (client non authentifié).

Interfaces

Manage order : Gestion d'une commande (passer, modifier, annuler une commande).

Manage customer : Permet à un employé de passer une commande sur un compte client.



Dépendances

Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.

2.2.8 Composant Commercial.

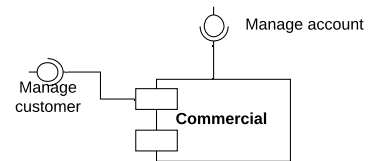
Nom : Commercial

Description : Commercial passant commande pour un client.

Dépendances

Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.

Manage customer : Permet de passer commande sur un compte client.



2.2.9 Composant Pizzaiolo.

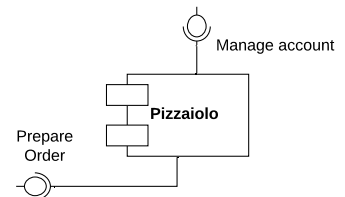
Nom : Pizzaiolo.

Description : Pizzaiolo effectuant la préparation d'une commande.

Dépendances

Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.

Prepare order: Préparation d'une commande client.



2.2.10 Composant Deliveryman.

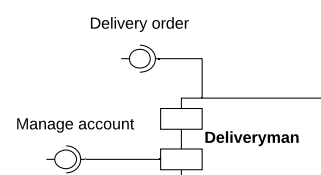
Nom : Deliveryman.

Description : Livreur livrant les commandes clients.

Dépendances

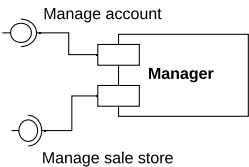
Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.

Delivery order: Livraison de la commande d'un client.



2.2.11 Composant Manager.

Nom : Manager.
Description : Responsable d'un point de vente du groupe OC Pizza.

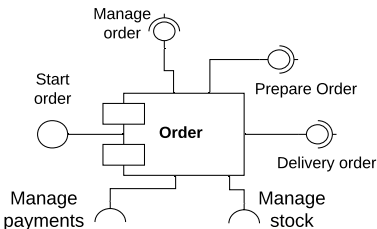


Dépendances
Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.
Manage sale store : Gestion d'un point de vente du groupe OC Pizza.

2.2.12 Composant Order.

Nom : Order.
Description : Commande client.

Interfaces
Manage order : Gestion d'une commande (passer, modifier, annuler une commande).
Delivery order: Livraison de la commande d'un client.
Prepare order: Préparation d'une commande client.
Create order: Créer une commande client.

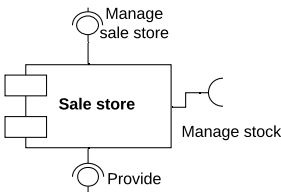


Dépendances
Manage account : Permet de modifier les informations du profil utilisateur et de gérer son compte.
Manage payments : Paiement d'une commande terminée.
Manage Stock : Modifie les stocks en fonction des commandes passées.

2.2.13 Composant Sale store.

Nom : Sale store.
Description : Point de vente du groupe OC Pizza

Interfaces
Manage sale store : Gestion d'un point de vente du groupe OC Pizza.

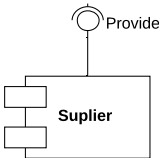


Dépendances
Manage stock : Gestion du stock (ajout, suppression de stock).
Provide : Gestion des fournisseurs.

2.2.14 Composant Suplier.

Composant : Suplier.
Description : Fournisseur du point de vente du groupe OC Pizza

Interfaces
Provide: Approvisionnement du point de vente.



2.2.15 Composant Bank.

Nom : Bank

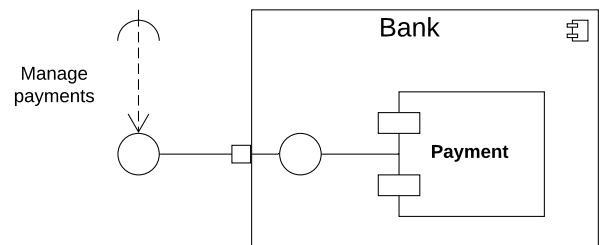
Description : Le composant Bank permet le paiement des commandes. Il représente les solutions de paiement en ligne tel que PayPal et Stripe.

Interfaces

Manage payments : Gère le paiement des commandes par carte bancaire.

Sous composant :

Payment : interface de paiement d'une commande.



2.2.16 Composant Payment.

Nom : Payment

Description : interface de paiement des solutions PayPal et Stripe.

Interfaces

Manage payments : Gère le paiement des commandes par carte bancaire.

2.2.17 Composant Warehouse.

Nom : Warehous

Description : Le composant Wharehouse représente la partie entrepôt d'un point de vente.

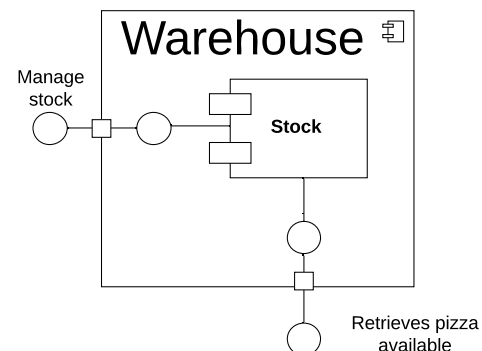
Interfaces

Manage stock : Gestion du stock (ajout, suppression d'ingrédients).

Retrieves pizzas available : Répertoire les pizzas disponibles en fonction du stock.

Sous composant :

Stock : stock d'un ingrédient nécessaire à la préparation d'une pizza.



2.2.18 Composant Stock.

Nom : stock

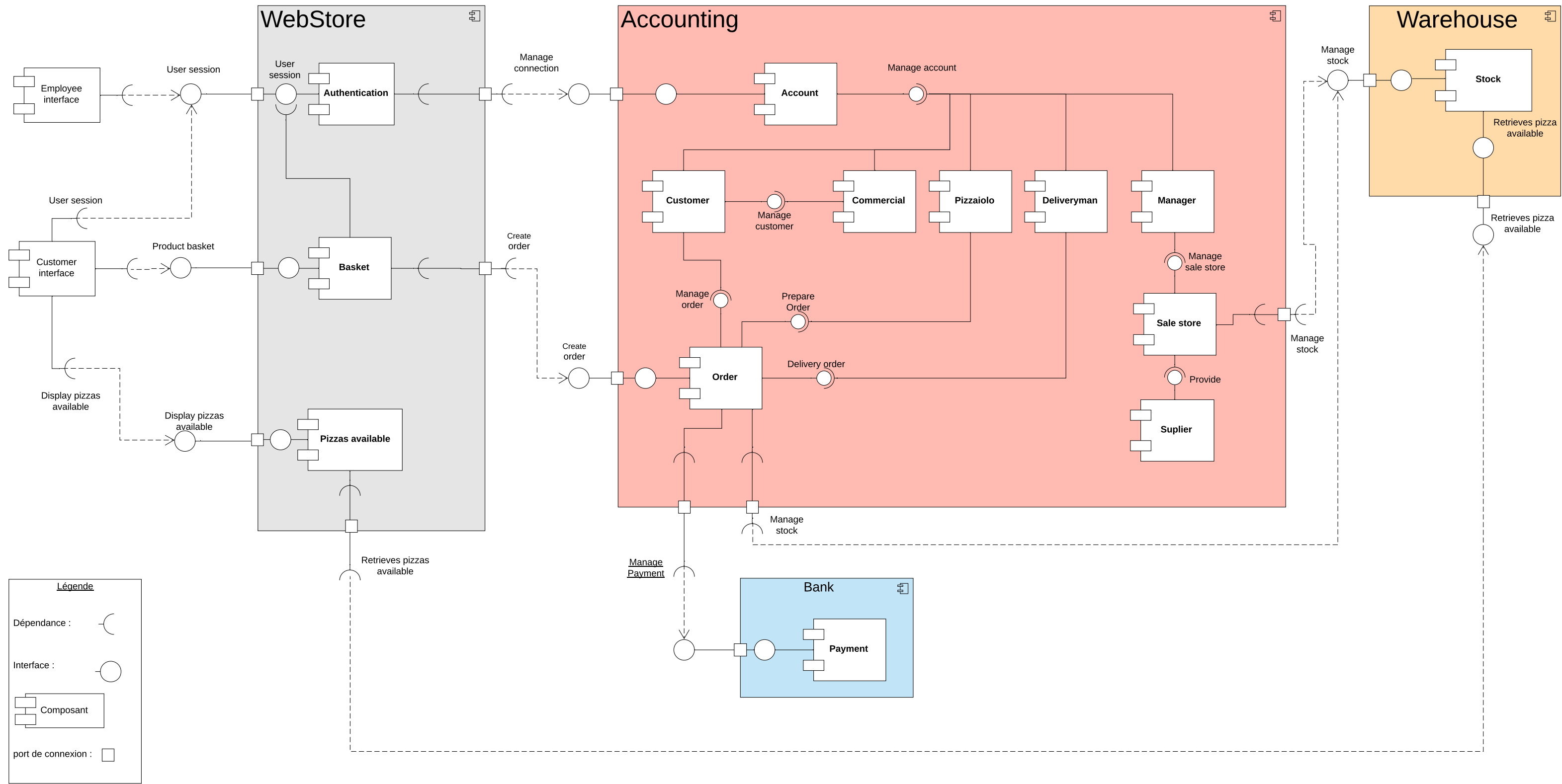
Description : stock d'un ingrédient nécessaire à la préparation d'une pizza.

Interfaces

Manage stock : Gestion du stock (ajout, suppression d'ingrédients).

Retrieves pizzas available : Répertoire les pizzas disponibles en fonction du stock.

2.3 Diagramme de composant.



3. LE DÉPLOIEMENT

3.1 Introduction au diagramme de déploiement.

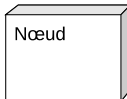
Le diagramme de déploiement décrit le déploiement physique des composants du système. Il permet de montrer le déploiement des éléments logiciels par les éléments matériels et la nature des connexions entre les différents éléments matériels.

Les éléments composants le diagramme de déploiement

Le diagramme de déploiement reprend les éléments du diagramme de composant ainsi que :

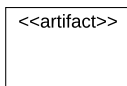
Les Nœuds

Ressource matériel ou logiciel du système représenté par un parallélépipède rectangle ou cube. Un nœud peut contenir des éléments (nœud, composant, artefact) et être relié par des associations.



Les Artefacts

Tous types d'éléments développés par le logiciel, symbolisés par un rectangle et le mot artefact. Ils mettent en œuvre des composants qui sont créés ou utilisés pendant le déploiement.



Les associations

Les associations entre les nœuds sont stéréotypées par le type de connexion ou technique de partage d'information.

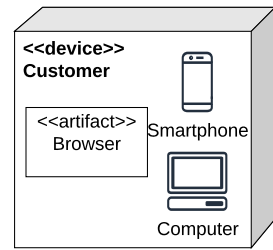
Ex : tcp/ip, http, https, API...

3.2 Description des nœuds.

3.2.1 Nœud Customer.

Le nœud Customer représente les appareils permettant l'accès à l'interface utilisateur du site web OC Pizza par connexion HTTPS.

Les types d'appareils par lesquels un utilisateur accède à l'interface du site OC Pizza peuvent être : un Smartphone, une tablette, un ordinateur et un ordinateur portable.

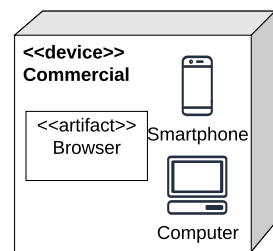


Artefact : L'artefact Browser est le logiciel avec lequel ils accèdent à l'interface du site web (Firefox, Chrome, Opera, Yahoo...).

3.2.2 Nœud Commercial.

Le nœud Commercial représente les appareils permettant l'accès à l'interface employé du site web OC Pizza par connexion HTTPS.

Les types d'appareils par lesquels un utilisateur accède à l'interface du site OC Pizza peuvent être : un Smartphone, une tablette, un ordinateur et un ordinateur portable.

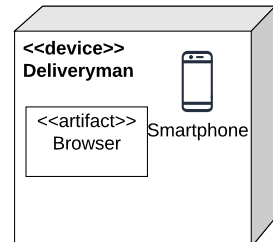


Artefact : L'artefact Browser est le logiciel avec lequel ils accèdent à l'interface du site web (Firefox, Chrome, Opera, Yahoo...).

3.2.3 Nœud Deliveryman.

Le nœud Commercial représente les appareils permettant l'accès à l'interface employé du site web OC Pizza par connexion HTTPS.

Les types d'appareils par lesquels un utilisateur accède à l'interface du site OC Pizza peuvent être : un Smartphone, une tablette, un ordinateur et un ordinateur portable.

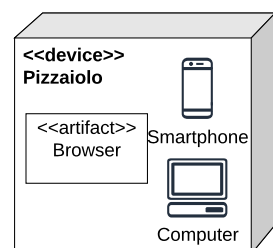


Artefact : L'artefact Browser est le logiciel avec lequel ils accèdent à l'interface du site web (Firefox, Chrome, Opera, Yahoo...).

3.2.4 Nœud Pizzaiolo.

Le nœud Commercial représente les appareils permettant l'accès à l'interface employé du site web OC Pizza par connexion HTTPS.

Les types d'appareils par lesquels un utilisateur accède à l'interface du site OC Pizza peuvent être : un Smartphone ou une tablette.

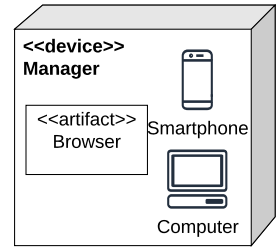


Artefact : L'artefact Browser est le logiciel avec lequel ils accèdent à l'interface du site web (Firefox, Chrome, Opera, Yahoo...).

3.2.5 Nœud Manager.

Le nœud Commercial représente les appareils permettant l'accès à l'interface employé du site web OC Pizza par connexion HTTPS.

Les types d'appareils par lesquels un utilisateur accède à l'interface du site OC Pizza peuvent être : un Smartphone, une tablette, un ordinateur et un ordinateur portable.



Artefact : L'artefact Browser est le logiciel avec lequel ils accèdent à l'interface du site web (Firefox, Chrome, Opera, Yahoo...).

3.2.6 Nœud Heroku.

Heroku est une plateforme permettant de déployer des applications sur le cloud AWS (Amazon Web Service) et ainsi bénéficier du savoir-faire d'Amazon.

Il offre une interface d'utilisation simple et performante à l'aide de l'interface en ligne de commande heroku (Heroku CLI) et utilise un tarif flexible en fonction des performances souhaitées.

Sous nœud

Django : Framework de développement de l'application OC Pizza.

Postgresql database : base de données du projet.

3.2.7 Nœud Django.

Django est un framework web puissant de haut niveau permettant de créer des applications web en Python. Il permet une réalisation rapide, sécurisée et simple de maintenance.

Le nœud Django contient les composants de l'application web à savoir la partie magasin, comptabilité et stock. Il utilise les services des nœuds Stripe, PayPal par le biais d'une API.

Les composants

Customer interface : interface web utilisée par l'utilisateur client ou visiteur.

Employee interface : interface utilisateur utilisée par tous les employés du groupe OC Pizza.

WebStore : Partie magasin permettant la gestion de l'authentification ainsi que l'affichage du panier et la gestion du catalogue pour un client.

Accounting : Comptabilité du groupe OC Pizza il gère la gestion des comptes, des commandes, des points de ventes ainsi que des fournisseurs.

Wharehous : Partie entrepôt d'un point de vente permettant la gestion des stocks.

3.2.8 Nœud Stripe.

Stripe est une solution de traitement de paiement en ligne permettant à l'aide de son API de gérer le paiement des commandes de façon sécuriser.

Composant

Payment : API du traitement de paiement en ligne de Stripe.

3.2.9 Nœud PayPal.

PayPal est une solution de paiement en ligne qui accepte tous type de paiement en ligne (carte, compte bancaire, ou solde PayPal) mis à part la crypto-monnaies.

Composant

Payment : API du traitement de paiement en ligne de PayPal.

3.2.10 Nœud Postgresql database.

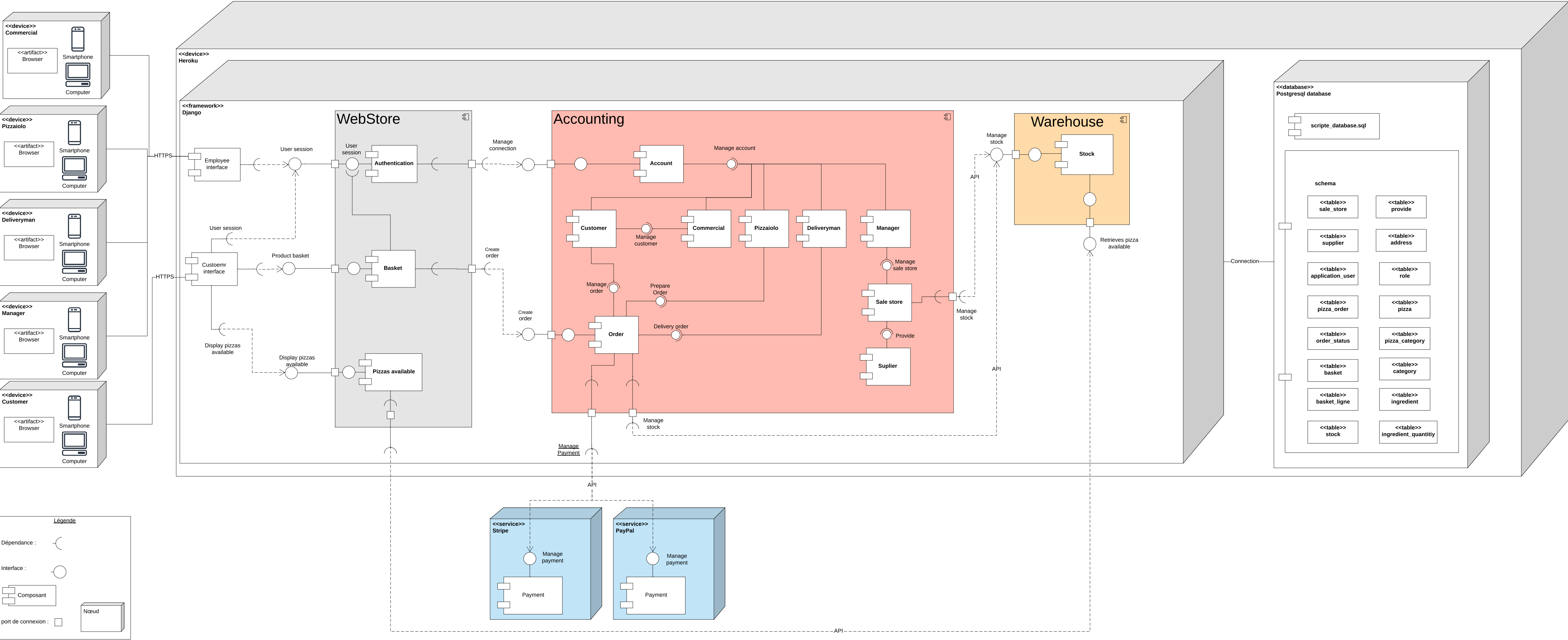
Postgresql est un système de gestion de base de données relationnel open source. C'est un SGBR puissant et il est pris en charge par Heroku.

Composant

script_database.sql : Scripte de création de la base de données.

Schema : schéma contenant les tables de la base de données.

3.3 Diagramme de déploiement.



4. LA BASE DE DONNÉES

4.1 Introduction au Modèle Physique de Données.

Le modèle physique de données représente la structure logique d'une base de données, les Associations et les contraintes qui déterminent comment les données peuvent être stockées et accessibles.

Le modèle physique de données est créé à partir du digramme de classe précédemment présenter.

Création du MPD

Chaque table du MPD reprend une classe du diagramme de classe et les colonnes d'une classe son ajouter à la table. On indique pour chaque colonne le type de données ainsi que la restriction du type de données.

EX : pour une chaîne de caractère on peut restreindre le nombre de caractère : "name : VARCHAR(30)", le nom est donc restreint à 30 caractère.

Clé primaire :

Une table contient toujours une ou plusieurs colonnes identifiées comme Clé primaire et permet d'identifier de manière unique un enregistrement dans une table. La Clé primaire est représentée par la paire PK entre crochet [PK].

Clé étrangère :

Une clés étrangère permet de gérer la association entre deux tables. Elle fait référence à la Clé primaire d'une autre table. La clés étrangère est représenter par la paire FK entre crochet [FK].

Les associations :

Les association entre les tables son créé en fonction des associations entre les classes du diagramme de classe.

Pour les associations de type un à un il est possible de fusionner les classes entre elles mais cela ne doit pas nuire à la lisibilité des données et aux performances.

Légende :

Table :

| | |
|-------------------------------------------------|--|
| pizza_order | |
| id: INTEGER NOT NULL [PK] | |
| order_status_name: VARCHAR(300) NOT NULL [FK] | |
| date_time: TIMESTAMP NOT NULL | |
| total_cost: DECIMAL NOT NULL | |
| basket_id: INTEGER NOT NULL [FK] | |
| user_id: INTEGER NOT NULL [FK] | |
| address_id: INTEGER NOT NULL [FK] | |

Clé primaire : [PK]

Clés étrangère : [FK]

Cardinalités des associations :

1 = ———+ zéro ou plusieurs = ———○<=

0 ou 1 = ———○+ un ou plusieurs = ———<=

4.2 Description des tables.

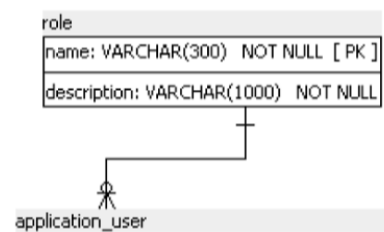
4.2.1 Table role.

Nom : role

Description : La classe role représente le rôle d'un utilisateur. C'est le rôle de l'utilisateur qui va indiquer à l'application le type d'interface à afficher et autoriser certaines actions.

Clé primaire

name : le nom du rôle est unique, il peut donc être utilisé comme Clé primaire de la classe role.



| application_user | | | | | | | |
|------------------|------------|-----------|------------|-----------------------|------------|-----------------|----------------|
| id [PK] | first_name | last_name | tel | email | password | address_id [FK] | role_name [FK] |
| 1 | Mario | bross | 0711111111 | mario.bros@gmail.com | MarioBros | 1 | Customer |
| 2 | Donkey | Kong | 0700000000 | donkey.kong@gmail.com | DonkeyKong | 2 | Commercial |

| role | |
|------------|------------------------------------------|
| name [PK] | description |
| Customer | User who orders pizzas. |
| Commercial | User who orders pizzas for the customer. |
| Pizzaiolo | User who prepares orders. |

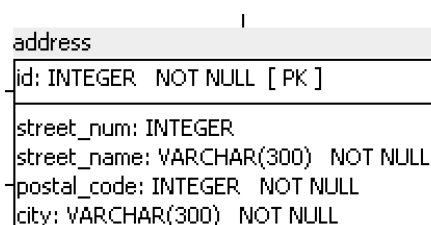
4.2.2 Table address.

Nom : address

Description : La classe address contient les adresses de tous les utilisateurs, points de ventes, fournisseurs et commandes.

Clé primaire

id : la colonne id permet d'identifier une adresse de manière unique dans la classe address.



| application_user | | | | | | | |
|------------------|------------|-----------|------------|-----------------------|------------|-----------------|----------------|
| id [PK] | first_name | last_name | tel | email | password | address_id [FK] | role_name [FK] |
| 1 | Mario | bross | 0711111111 | mario.bros@gmail.com | MarioBros | 1 | Customer |
| 2 | Donkey | Kong | 0700000000 | donkey.kong@gmail.com | DonkeyKong | 2 | Commercial |

| address | | | | |
|---------|------------|-----------------------------|-------------|-------|
| id [PK] | street_num | street_name | postal_code | city |
| 1 | 1 | Rue aus Ours | 76000 | ROUEN |
| 2 | 9 | Rue Guillaume le Conquérant | 76000 | ROUEN |

4.2.3 Table application_user.

Nom : application_user

Description : La classe application_user représente un utilisateur de l'application, elle renseigne ses informations.

Clé primaire

id : id est la Clé primaire de cette classe elle permet d'identifier un utilisateur de manière unique.

Clés étrangères

address_id : fait référence à la classe address et représente l'adresse de l'utilisateur.

role_name : fait référence à la classe role et indique le rôle d'un utilisateur (Customer, Commercial, Deliveryman, Pizzaiolo, Manager).

| application_user | |
|-----------------------------------------|--|
| id: INTEGER NOT NULL [PK] | |
| first_name: VARCHAR(38) NOT NULL | |
| last_name: VARCHAR(38) NOT NULL | |
| tel: VARCHAR(10) NOT NULL | |
| email: VARCHAR(300) NOT NULL | |
| password: VARCHAR(300) NOT NULL | |
| address_id: INTEGER NOT NULL [FK] | |
| role_name: VARCHAR(300) NOT NULL [FK] | |

| application_user | | | | | | | |
|------------------|------------|-----------|------------|-----------------------|------------|-----------------|----------------|
| id [PK] | first_name | last_name | tel | email | password | address_id [FK] | role_name [FK] |
| 1 | Mario | bross | 0711111111 | mario.bros@gmail.com | MarioBros | 1 | Customer |
| 2 | Donkey | Kong | 0700000000 | donkey.kong@gmail.com | DonkeyKong | 2 | Commercial |

| address | | | | |
|---------|------------|-----------------------------|-------------|-------|
| id [PK] | street_num | street_name | postal_code | city |
| 1 | 1 | Rue aus Ours | 76000 | ROUEN |
| 2 | 9 | Rue Guillaume le Conquérant | 76000 | ROUEN |

| role | |
|------------|------------------------------------------|
| name [PK] | description |
| Customer | User who orders pizzas. |
| Commercial | User who orders pizzas for the customer. |

4.2.4 Table order_status.

Nom : order_status

Description : order_status indique le statut d'une commande, elle permet d'indiquer aux différents acteurs de la préparation d'une commande les actions qu'ils doivent effectués.

Ex : "Awaiting prepare" indique au pizzaiolo qu'il doit préparer la commande.

Clé primaire

name : le nom de l'état d'une commande étant unique il est utilisé comme Clé primaire.

| order_status | |
|------------------------------------|--|
| name: VARCHAR(300) NOT NULL [PK] | |

| pizza_order | | | | | | | |
|-------------|------------------|------------------------|------------|----------------|--------------|-----------------|--|
| id [PK] | date_time | order_status_name [FK] | total_cost | basket_id [FK] | user_id [FK] | address_id [FK] | |
| 1 | 2019-11-10 18:05 | Awaiting prepare | 80.5 | 5 | 5 | 1 | |
| 2 | 2019-11-12 18:05 | Completed | 12 | 6 | 6 | 1 | |

| order_status |
|--------------------|
| name [PK] |
| Awaiting prepare |
| Under prepare |
| Awaiting delivery |
| Under delivery |
| Under modification |
| Canceled |
| Completed |

4.2.5 Table sale_store.

Nom : sale_store

Description : Représente un point de vente du groupe OC Pizza.

Clé primaire

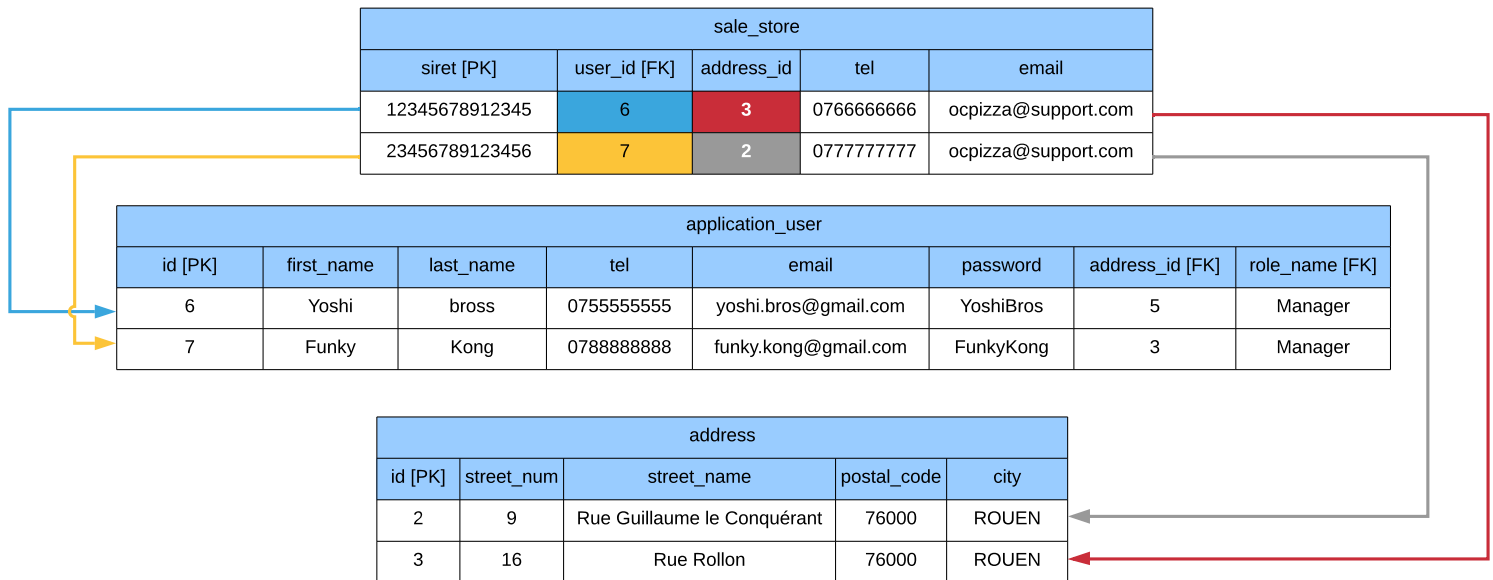
siret : Un siret permet d'identifier un établissement d'une entreprise de manière unique et peut donc être utilisé comme Clé primaire.

Clés étrangères

user_id : fait référence au manager qui gère le point de vente.

address_id : indique l'id de l'adresse du point de vente dans la classe address.

| sale_store | |
|-------------------------------------|--|
| siret: BIGINT NOT NULL [PK] | |
| user_id: INTEGER NOT NULL [FK] | |
| address_id: INTEGER NOT NULL [FK] | |
| tel: VARCHAR(10) NOT NULL | |
| email: VARCHAR(300) NOT NULL | |



4.2.6 Table suplier.

Nom : suplier

Description : Représente un fournisseur du groupe OC Pizza.

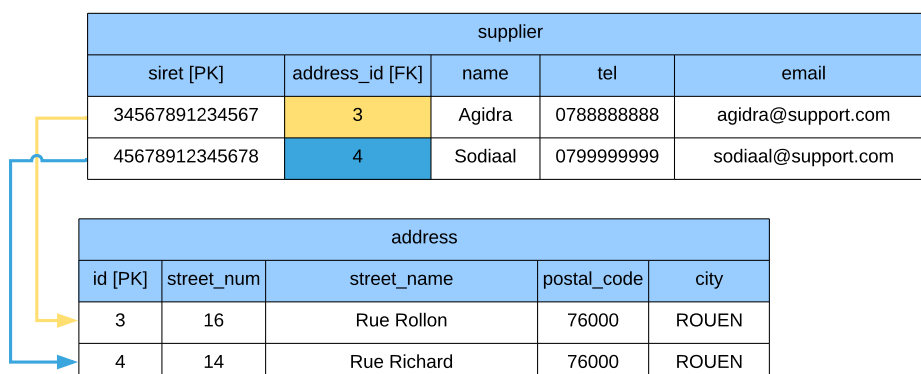
Clé primaire

siret : Un siret permet d'identifier un établissement d'une entreprise de manière unique et peut donc être utilisé comme clé primaire.

Clés étrangères

address_id : indique l'id de l'adresse du fournisseur dans la classe address.

| suplier | |
|-------------------------------------|--|
| siret: BIGINT NOT NULL [PK] | |
| address_id: INTEGER NOT NULL [FK] | |
| name: VARCHAR(100) NOT NULL | |
| tel: VARCHAR(10) NOT NULL | |
| email: VARCHAR(300) NOT NULL | |



4.2.7 Table provide.

Nom : provide

Description : La classe provide remplace l'association plusieurs à plusieurs entre la classe supplier et sale_store et permet de lier un point de vente avec ses fournisseurs.

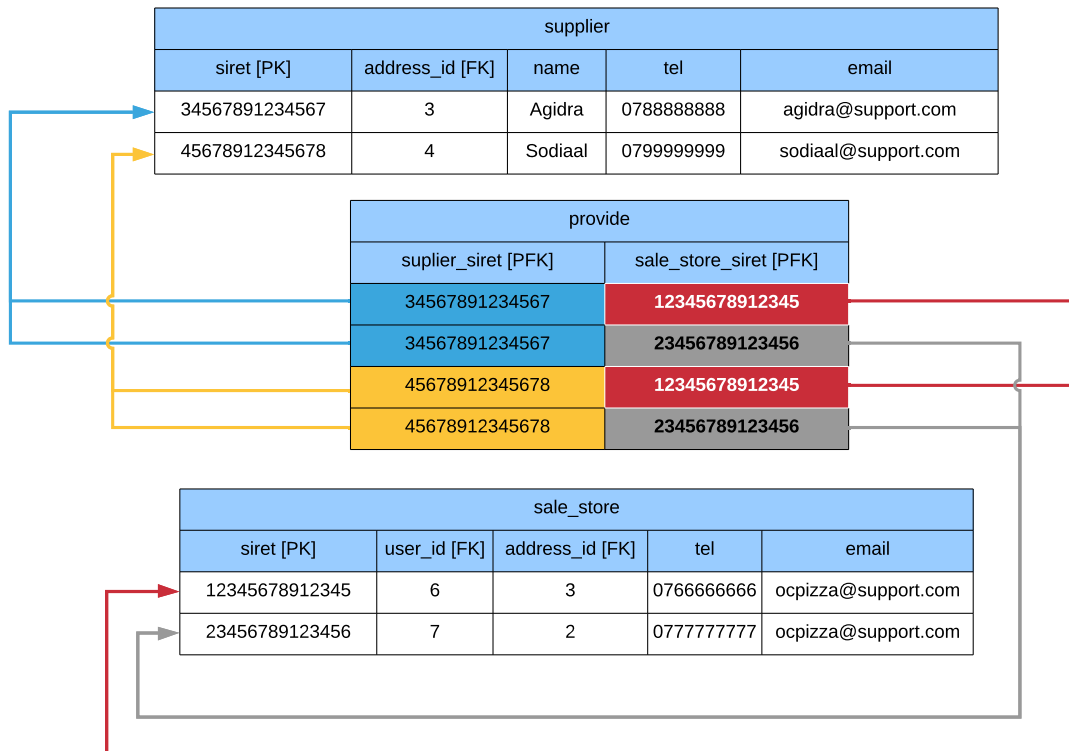
| provide | |
|--------------------------------------------|--|
| supplier_siret: VARCHAR NOT NULL [PFK] | |
| sale_store_siret: VARCHAR NOT NULL [PFK] | |

Clés primaires et étrangères

la classe provide dispose d'une paire de clés primaires qui sont elles-mêmes des clés étrangères.

supplier_siret : Fait référence à la clé primaire de la classe supplier.

sale_store_siret : Fait référence à la clé primaire de la classe sale_store.



4.2.8 Table ingredient.

Nom : ingredient

Description : Représente un ingrédient nécessaire à la préparation d'une pizza.

| ingredient | |
|-----------------------------|--|
| id: INTEGER NOT NULL [PK] | |
| name: VARCHAR(100) NOT NULL | |
| description: VARCHAR(300) | |

Clé primaire

id : Chaque ingrédient est représenté par un entier unique.

| ingredient | | |
|------------|-------|---------------------|
| id [PK] | name | description |
| 1 | Flour | Flour T45. |
| 2 | Yeast | Fresh baking yeast. |
| 3 | Salt | Fine salt. |
| 4 | Sugar | Fine sugar. |

4.2.9 Table stock.

Nom : stock

Description : Représente un stock d'ingrédient d'un point de vente du groupe OC Pizza.

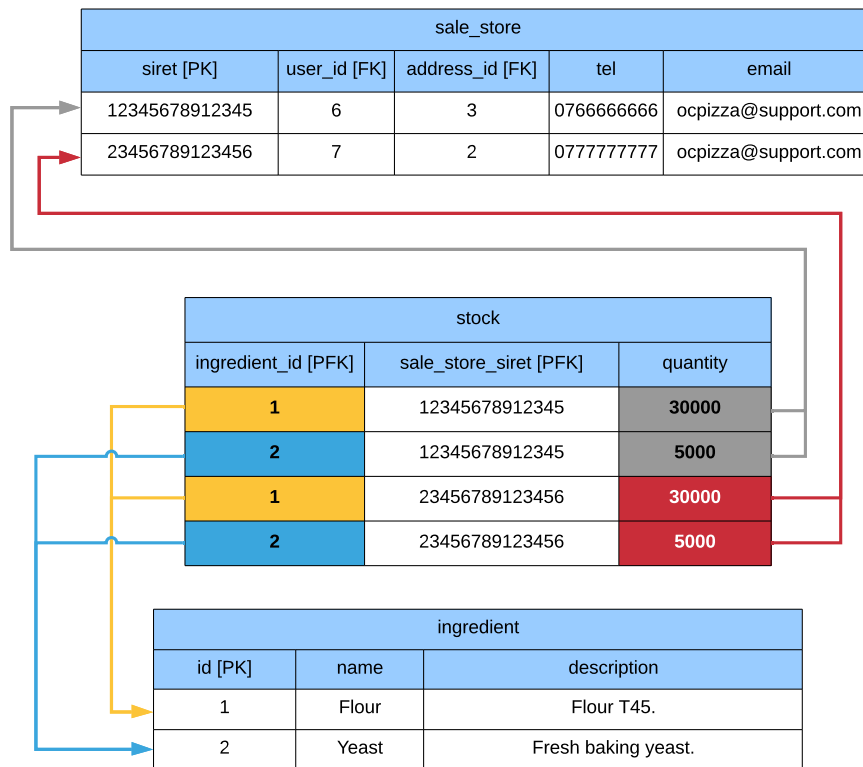
| stock | |
|--------------------------------------------|--|
| ingredient_id: INTEGER NOT NULL [PFK] | |
| sale_store_siret: VARCHAR NOT NULL [PFK] | |
| quantity: DECIMAL | |

Clés primaires et étrangères

la classe stock dispose d'une paire de clés primaires qui sont elles-mêmes des clés étrangères.

ingredient_id : Fait référence à l'identifiant d'un ingrédient dans la classe ingredient.

sale_store_siret : Fait référence au SIRET d'un point de vente du groupe OC Pizza.



4.2.10 Table pizza.

Nom : pizza

Description : la classe pizza représente une pizza vendue par le groupe OC Pizza.

Clé primaire

id : la clé primaire est représentée par un entier unique.

| pizza | |
|-------------------------------------|--|
| id: INTEGER NOT NULL [PK] | |
| name: VARCHAR(100) NOT NULL | |
| url_picture: VARCHAR(300) | |
| recipe: VARCHAR(1000) NOT NULL | |
| description: VARCHAR(1000) NOT NULL | |
| price: NUMERIC(5, 2) NOT NULL | |

| pizza | | | | | |
|---------|------------------|---------------------|------------------------|-----------------------|-------|
| id [PK] | name | url_picture | recipe | description | price |
| 1 | Honey goat | https://url.picture | Préchauffer le four... | Pizza au fromage... | 10.5 |
| 2 | Pizza 4 Fromages | https://url.picture | Préchauffer le four... | Pizza aux 4 fromages. | 12 |

4.2.11 Table ingredient_quantity.

Nom : ingredient_quantity

Description : la classe ingredient_quantity permet d'indiquer la quantité d'un ingrédient contenu dans une pizza.

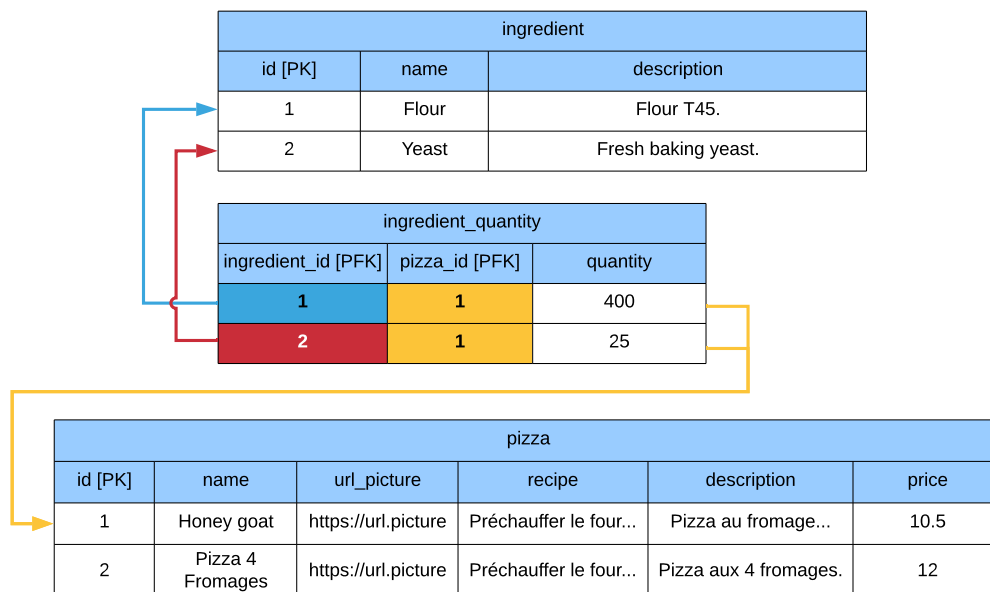
| ingredient_quantity | |
|------------------------|------------------|
| ingredient_id: INTEGER | NOT NULL [PFK] |
| pizza_id: INTEGER | NOT NULL [PFK] |
| quantity: DECIMAL | NOT NULL |

Clé primaire

la classe ingredient_quantity dispose d'une paire de clés primaires qui sont elles-mêmes des clés étrangères.

ingredient_id : fait référence à l'identifiant d'un ingrédient.

pizza_id : fait référence à l'identifiant d'une pizza.



4.2.12 Table category.

Nom : category

Description : Représente la catégorie d'une pizza (Cheeses, Meats, Fishes, Fruits).

| category | |
|--------------------|-----------------|
| name: VARCHAR(300) | NOT NULL [PK] |

Clé primaire

name : le nom d'une catégorie étant unique il représente la clé primaire.

| category |
|-----------|
| name [PK] |
| Cheeses |
| Meats |
| Fishes |
| Fruits |

4.2.13 Table pizza_category.

Nom : pizza_category

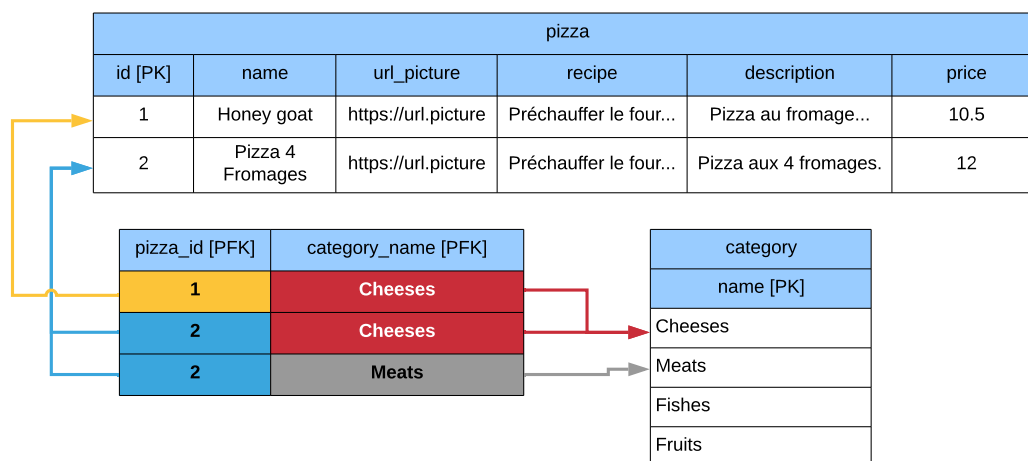
Description : La classe pizza_category remplace l'association plusieurs à plusieurs entre les classe pizza et category.

| pizza_category | |
|----------------------------------------------|--|
| pizza_id: INTEGER NOT NULL [PFK] | |
| category_name: VARCHAR(300) NOT NULL [PFK] | |

Clé primaire

la classe pizza_catégorie dispose d'une paire de clés primaires qui sont elles-mêmes des clés étrangères.

name : le nom d'une catégorie étant unique il représente la clé primaire.



4.2.14 Table basket.

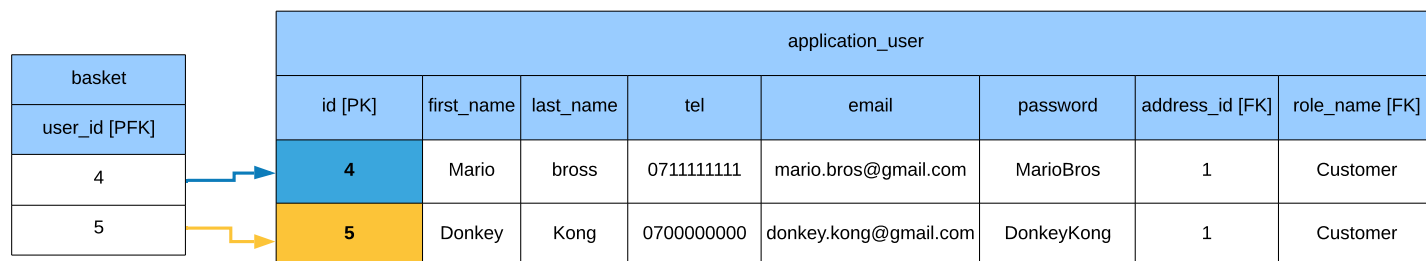
Nom : basket

Description : Représente un panier de produits d'un utilisateur.

| basket | |
|-----------------------------------|--|
| user_id: INTEGER NOT NULL [PFK] | |

Clé primaire et étrangère

user_id : Fait référence à l'identifiant d'un utilisateur.



4.2.15 Table basket_ligne.

Nom : basket_ligne

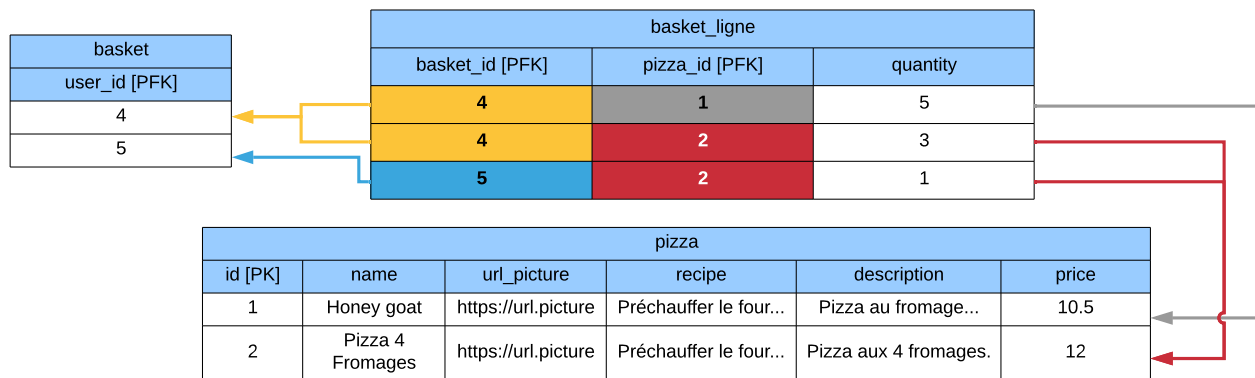
Description : Représente une ligne du panier de produits d'un utilisateur. Elle contient la quantité de pizza qu'un utilisateur souhaite commander.

| basket_ligne | |
|------------------------------------|--|
| id: INTEGER NOT NULL [PK] | |
| basket_id: INTEGER NOT NULL [FK] | |
| pizza_id: INTEGER NOT NULL [FK] | |
| quantity: INTEGER NOT NULL | |

Clés primaires et étrangères

basket_id : Fait référence à l'identifiant unique d'un panier.

pizza_id : Fait référence à l'identifiant unique d'une pizza.



4.2.16 Table pizza_order.

Nom : pizza_order

Description : Représente la commande d'un client.

Clé primaire

id : Identifiant unique d'une commande.

Clés étrangères

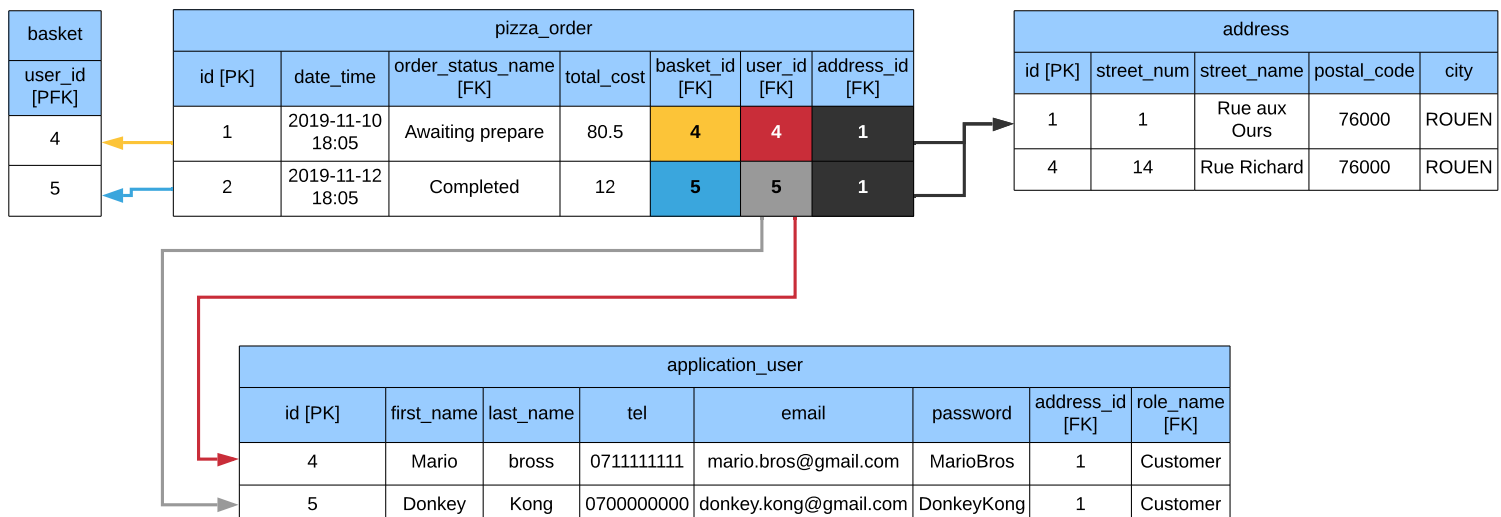
order_status_name : fait référence à la classe "order_status" et indique l'état de la commande.

basket_id : fait référence à l'identifiant de la classe "basket" et indique les pizzas commandé par le client.

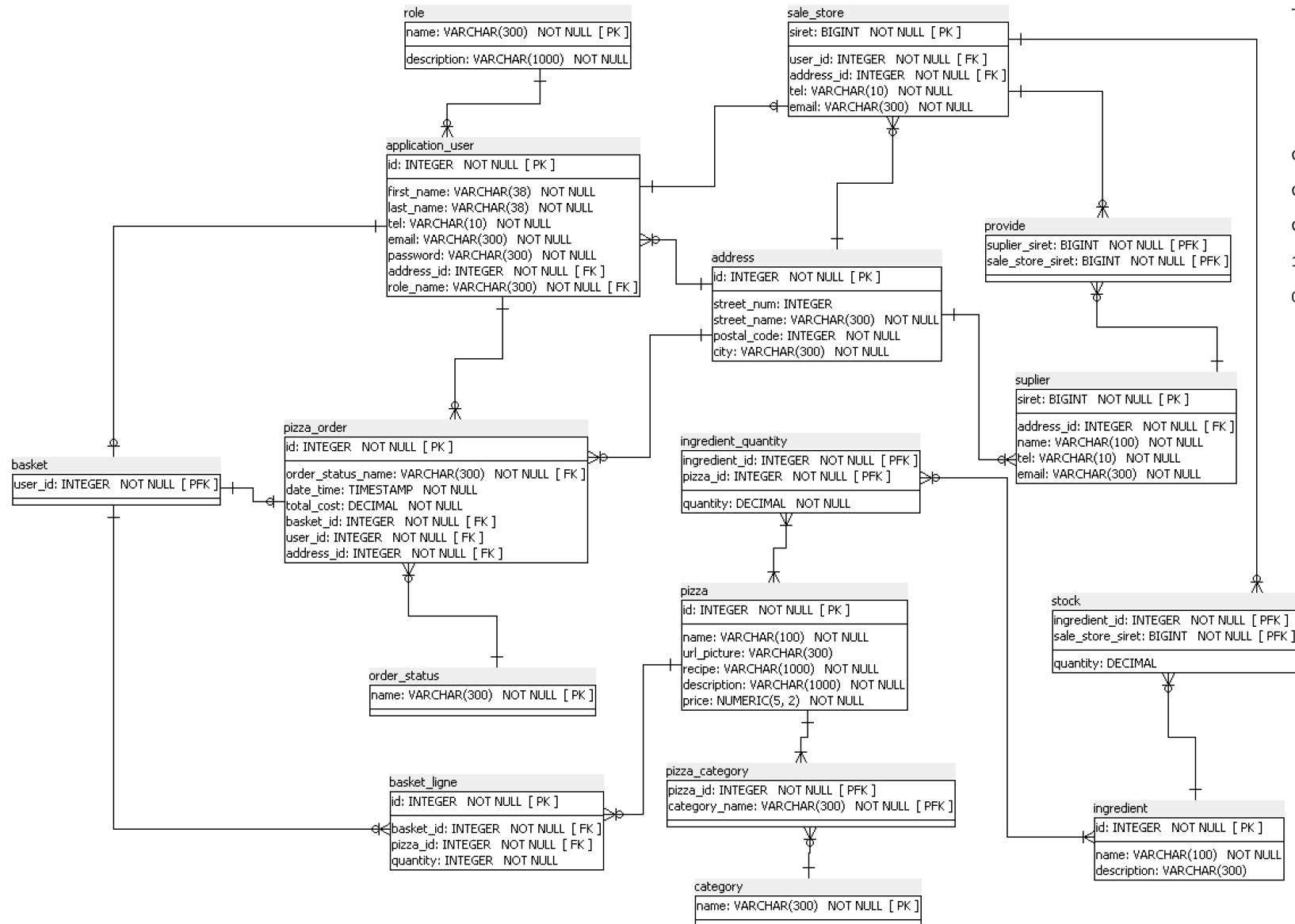
user_id : fait référence à un identifiant unique de la classe "application_user" pour indiquer l'utilisateur qui a passer la commande.

address_id : fait référence à l'identifiant de la classe "address" et indique l'adresse de livraison de la commande.

| pizza_order | |
|-------------------------------------------------|--|
| id: INTEGER NOT NULL [PK] | |
| order_status_name: VARCHAR(300) NOT NULL [FK] | |
| date_time: TIMESTAMP NOT NULL | |
| total_cost: DECIMAL NOT NULL | |
| basket_id: INTEGER NOT NULL [FK] | |
| user_id: INTEGER NOT NULL [FK] | |
| address_id: INTEGER NOT NULL [FK] | |



4.3 Modèle Physique de Données.



Légende :

Table :

```

pizza_order
id: INTEGER NOT NULL [ PK ]
order_status_name: VARCHAR(300) NOT NULL [ FK ]
date_time: TIMESTAMPT NOT NULL
total_cost: DECIMAL NOT NULL
basket_id: INTEGER NOT NULL [ FK ]
user_id: INTEGER NOT NULL [ FK ]
address_id: INTEGER NOT NULL [ FK ]


```

Clé primaire : [PK]

Clés étrangère : [FK]

Cardinalités des associations :

1 = zéro ou plusieurs =

0 ou 1 = 

zéro ou plusieurs =

un ou plusieurs =

Table des matières

| | |
|--------------------------------------------------------------|----|
| Contexte..... | 3 |
| 1. DESCRIPTION DU DOMAINE FONCTIONNEL | |
| 1.1 Identification des classes et de leurs associations..... | 4 |
| 1.2 Identification des classes..... | 5 |
| 1.2.1 Classe User..... | 5 |
| 1.2.2 Classe Order..... | 6 |
| 1.2.3 Classe Basket..... | 7 |
| 1.2.4 Classe BasketLigne..... | 7 |
| 1.2.5 Classe Pizza..... | 8 |
| 1.2.6 Classe Ingredient..... | 8 |
| 1.2.7 Classe Stock..... | 9 |
| 1.2.8 Classe SaleStore..... | 9 |
| 1.2.9 Classe Suplier..... | 10 |
| 1.3 Introduction au diagramme de classe..... | 10 |
| 1.4 Diagramme de classe..... | 11 |
| 2. LES COMPOSANTS | |
| 2.1 Introduction au diagramme de composant..... | 12 |
| 2.2 Description des composants..... | 13 |
| 2.2.1 Composant Employee interface..... | 13 |
| 2.2.2 Composant Customer interfcae..... | 13 |
| 2.2.3 Composant WebStore..... | 14 |
| 2.2.4 Composant Authentification..... | 14 |
| 2.2.5 Composant Basket..... | 15 |
| 2.2.6 Composant Pizzas available..... | 15 |
| 2.2.7 Composant accounting..... | 16 |

Table des matières

| | |
|---------------------------------------------------|----|
| 2.2.8 Composant Account..... | 17 |
| 2.2.9 Composant Customer..... | 1 |
| 2.2.10 Composant Commercial..... | 17 |
| 2.2.11 Composant Pizzaiolo..... | 17 |
| 2.2.12 Composant Deliveryman..... | 17 |
| 2.2.13 Composant Manager..... | 18 |
| 2.2.14 Composant Order..... | 18 |
| 2.2.15 Composant Sale store..... | 18 |
| 2.2.16 Composant Suplier..... | 18 |
| 2.2.17 Composant Bank..... | 19 |
| 2.2.18 Composant Payment..... | 19 |
| 2.2.19 Composant Wharehouse..... | 19 |
| 2.2.20 Composant Stock..... | 19 |
| 2.3 Diagramme de composants..... | 20 |
| 3. LE DÉPLOIEMENT | |
| 3.1 Introduction au diagramme de déploiement..... | 21 |
| 3.2 Description des nœuds..... | 22 |
| 3.2.1 Nœud Customer..... | 22 |
| 3.2.2 Nœud Commercial..... | 22 |
| 3.2.3 Nœud Pizzaiolo..... | 22 |
| 3.2.4 Nœud Deliveryman..... | 22 |
| 3.2.5 Nœud Manager..... | 23 |
| 3.2.6 Nœud Heroku..... | 23 |
| 3.2.7 Nœud Django..... | 23 |
| 3.2.8 Nœud Stripe..... | 24 |
| 3.2.9 Nœud PayPal..... | 24 |

Table des matières

| | |
|-----------------------------------------------------|----|
| 3.2.10 Nœud Postgresql database..... | 24 |
| 3.3 Diagramme de déploiement..... | 25 |
| 4. LA BASE DE DONNÉES | |
| 4.1 Introduction au modèle physique de données..... | 26 |
| 4.2 Description des tables..... | 27 |
| 4.2.1 Table role..... | 27 |
| 4.2.2 Table address..... | 27 |
| 4.2.3 Table application_user..... | 28 |
| 4.2.4 Table order_status..... | 28 |
| 4.2.5 Table sale_store..... | 29 |
| 4.2.6 Table supplier..... | 29 |
| 4.2.7 Table provide..... | 30 |
| 4.2.8 Table ingredient..... | 30 |
| 4.2.9 Table stock..... | 31 |
| 4.2.10 Table pizza..... | 31 |
| 4.2.11 Table ingredient_quantity..... | 32 |
| 4.2.12 Table category..... | 32 |
| 4.2.13 Table pizza_category..... | 33 |
| 4.2.14 Table basket..... | 33 |
| 4.2.15 Table basket_ligne..... | 34 |
| 4.2.16 Table pizza_order..... | 34 |
| 4.3 Modèle physique de données..... | 35 |

FIN