OC Pizza

Système de gestion de commandes

Dossier de conception technique

Version 2.0

Auteur
Bryan Ferreras-Roca
Analyste-programmeur

TABLE DES MATIERES

1 - VERSIONS

Auteur	Date	Description	Version
Bryan Ferreras-Roca	14/12/2021	Création du document	1.0
Bryan Ferreras-Roca	22/03/2022	Mise en conformité avec la trame d'entreprise	2.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier de conception technique du système de gestion des commandes.

2.1.1 - Objectif du document

Enumérer et détailler les composants techniques du système. Ce document est à destination des équipes de développeurs et opérationnels.

Les éléments du présent dossier découlent :

- du diagramme de classes
- du modèle physique de données
- du diagramme de composants
- du digramme de déploiement

2.2 - Références

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

1. Dossier de conception fonctionnelle
2. Dossier d'exploitation
3. PV de livraison

3 - Architecture Technique

3.1 - Composants généraux

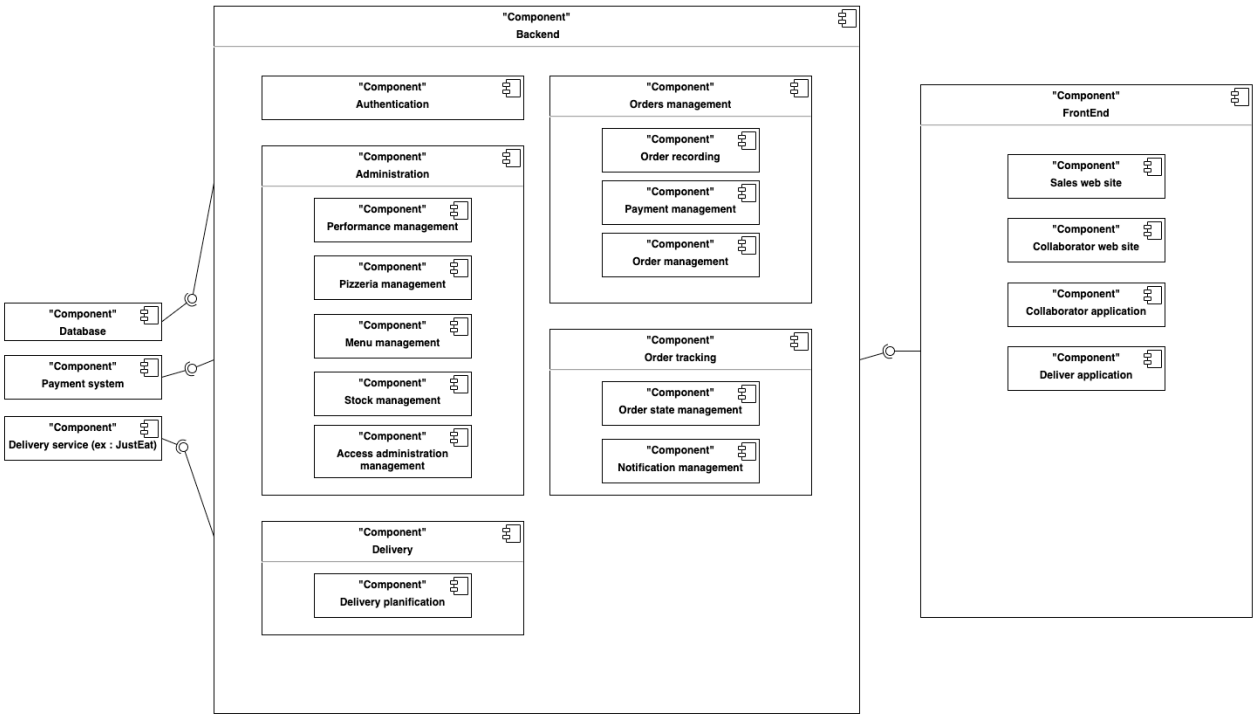
La pile logicielle est la suivante :

- Kotlin v 1.6.10
- Java v1.8
- Ubuntu Server v 20.04
- Apache v 2.4.41
- PostgreSQL 12
- Git v 2.25.1
- Github
- Rsync 3.1.3

3.1.1 - Diagramme de composants

L'architecture technique est découpée en plusieurs composants.

Version en ligne



3.1.2 - Détail des composants backend

1. Composant « Authentification »

L'authentification permet de gérer les connexions/inscriptions des clients et collaborateurs.

2. Composant « Performance management »

Le suivi du chiffre d'affaires, des marges et des avis est effectué par ce composant.

3. Composant Pizzeria management

La gestion des pizzerias est effectuée par ce composant.

Celui-ci gère la création/modifications/suppressions des pizzerias.

4. Composant Menu management

La gestion des plats et des recettes est effectuée ici.

5. Composant Stock management

Toute la gestion du stock se fait ici. On peut ajouter des produits, les modifier et les supprimer. Une option d'alerte de faible quantité est activable et paramétrable. La sortie du stock est automatiquement effectuée par le système lors des ventes ; ceci afin de garantir une expérience utilisateur (client/collaborateur) optimale.

6. Composant Access administration management

La gestion des comptes client/collaborateur est gérée par ce composant. Les clients peuvent créer/modifier/supprimer leur compte, tandis que les collaborateurs peuvent, en fonction de leur niveau d'accès, créer/modifier/supprimer des comptes client/collaborateur.

7. Composant Order recording

Ce composant permet de saisir les commandes des clients, que ce soit par téléphone ou sur place. Les commandes passées par les clients via le site web sont automatiquement enregistrées dans le système.

8. Composant Payment management

Les collaborateurs peuvent enregistrer les règlements via ce composant. Ceci afin que le système mette à jour automatiquement le statut d'une commande.

9. Composant Order management

Ce composant permet la gestion des commandes. Grâce à celui-ci, une grande souplesse est apportée sur la gestion des commandes. Une commande peut être modifiée ou supprimée à tout moment et peu importe sa nature (saisi par les collaborateurs ou par le client via le site web).

10. Composant Order state management

Le système met automatiquement à jour le statut d'une commande lors de son cycle de vie. Les collaborateurs ont également la possibilité de le modifier manuellement.

11. Composant Notification management

Le système peut notifier le client de l'évolution de sa commande grâce aux différents statuts. Les collaborateurs peuvent déclencher manuellement une notification qui sera basée sur le statut actuel de la commande.

12. Composant Database

La base de données contient l'ensemble des informations concernant les clients, les collaborateurs, les pizzerias, les stocks et les différents éléments indirectements nécessaires au bon fonctionnement du système (ex : statut d'une commande…). Le modèle physique de données (cf modèle physique de données) stipule l'ensemble des données stockées dans la base et leur type.

13. Composant Payment system

Ce composant concerne les clients passant leur commande via le site web. La gestion de leur règlement est alors assurée par un composant externe (fourni par la banque). Les clients utilisent le composant externe et celui-ci communique avec le backend.

14. Composant Delivery service

Des services de livraisons font partie de la vitrine de l'enseigne. Les services de livraisons fournissent une API afin que le backend puisse communiquer avec celui-ci (ex : enregistrement de commande, mise à jour dynamique du menu…).

3.1.3 - Détail des composants frontend

15. Composant Sales web site

Il représente le site internet via lequel les clients accèdent au menu et passent commandes.

16. Composant Collaborator web site

Il représente le site internet via lequel les collaborateurs interagissent avec le système afin d'y effectuer toutes les possibilités offertes par les différents composants.

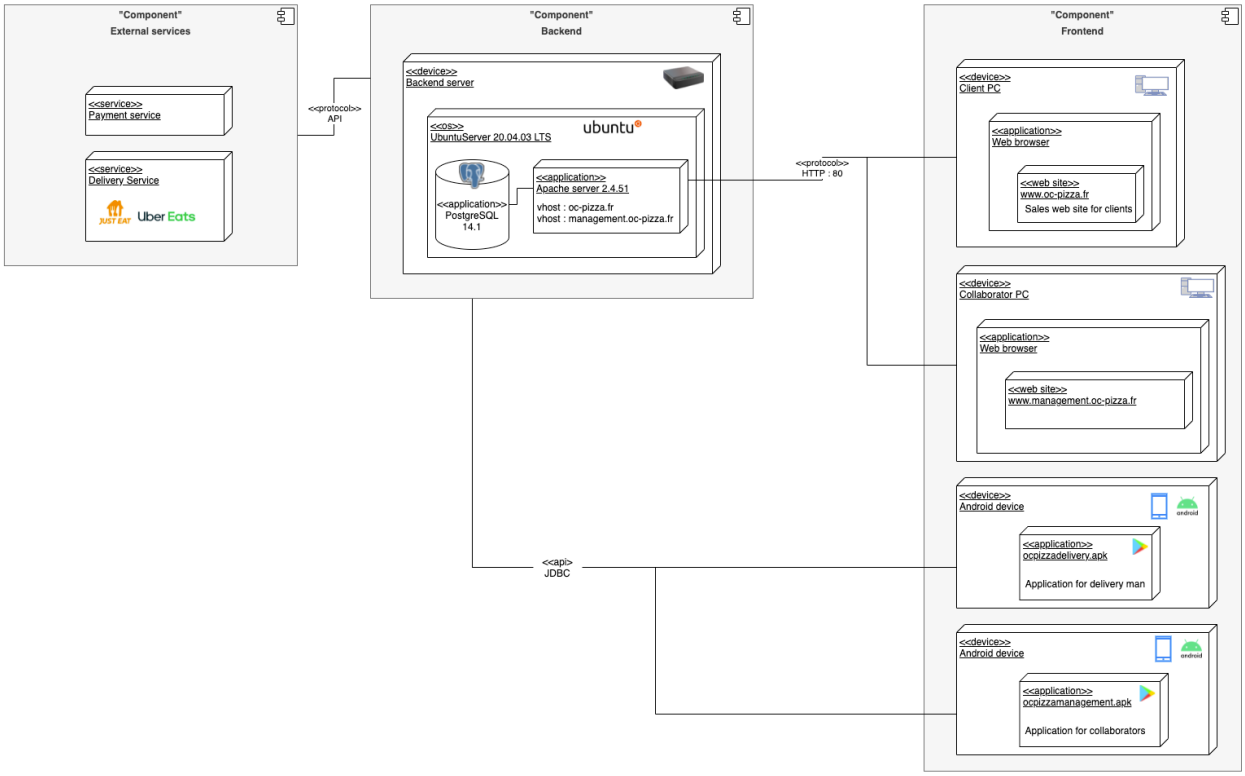
17. Composant Deliver application

Il représente l'application utilisée par les livreurs. Grâce à celle-ci, tout le processus de livraison est géré par l'application (de la notification d'une livraison programmée à l'encaissement du client…).

4 - Architecture de Deploiement

4.1 - Diagramme de déploiement

Version en ligne



4.2 - Applications mobile/tablette

Les applications sont responsives. Une copie locale est automatiquement conservée afin de garantir une HA (High Availability), dans le cas d'une défaillance des réseaux, par exemple. L'API minimum d'Android est la V21 Lollipop.

4.3 - Serveur

Le serveur Ubuntu gère le service Web (Apache) et la base de données (PostgreSQL). Afin d'optimiser les ressources, le même serveur Web gère le site de vente pour les clients et le site d'administration pour les collaborateurs via des virtual hosts.

5 - ARCHITECTURE LOGICIELLE

5.1 - Principes généraux

Les sources et versions du projet sont gérées par **Git** et hébergées **GitHub**. Les dépendances et le packaging sont eux gérés par **Gradle.**

5.1.1 - Les couches

L'architecture applicative choisit est la clean architecture.
Celle-ci vise à réduire les dépendances de la logique métier avec les services (APIs, librairies...) et ceci afin de maintenir un code stable, testable et scalable.

- une couche **data** : implémentation des données et apis
- une couche **domain** : définition des interfaces et des usescases
- une couche **presentation** : responsable des views

5.1.2 - Les modules

Etant donné l'architecture choisit, il y a 3 modules, 1 par couche :

- un module **data**
- un module **domain**
- un module **presentation**

5.1.3 - Structure des sources

La structuration des répertoires du projet suit la logique suivante :

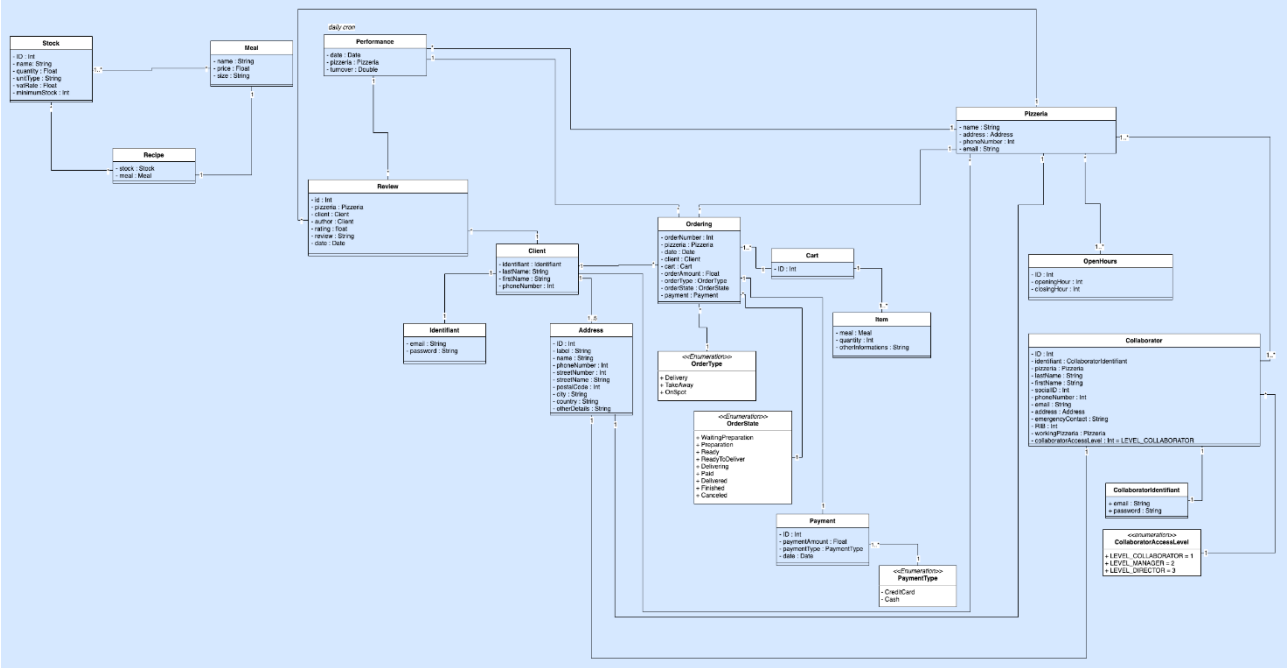
```
racine

|
|-<presentation>
|   |-src
|     |-----main
|           |-----java
|           |      |-----di
|           |      |-----uistates
|           |      |-----viewmodels
|           |      |-----views
|           |      |-----App
|           |-----test
|-<domain>
|   |-src
|     |-----main
|           |-----java
|                  |-----di
|                  |-----entities
|                  |-----repositories
|                  |-----response
|                  |-----usescases
|-<data>
|   |-src
|     |-----main
|           |-----java
|           |      |-----di
|           |      |-----apis
|           |      |-----datasources
|           |      |-----entities
|           |      |-----db
|           |      |-----repositories
|           |-----test
```

5.1.4 - Diagramme de classes

Grâce au diagramme de classe, les différentes entités et leurs relations sont identifiées.

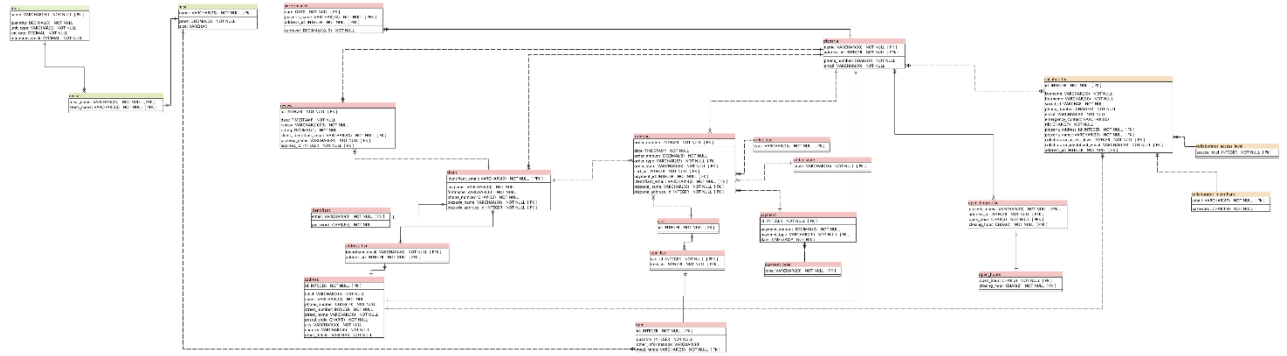
Version en ligne



5.2 - Base de données

La base de de données est de type PostgreSQL 12. Son schéma est conçu grâce au diagramme de classes ci-dessus. Ainsi en découle le modèle physique de données suivant :

Version en ligne



6 - POINTS PARTICULIERS

6.1 - Fichiers de configuration

6.1.1 - Application web

18. VHost oc-pizza.com

```
<VirtualHost *:80>
        ServerAdmin webmaster@oc-pizza.com
        DocumentRoot /var/www/ocpizza/OC-Pizza/
        ServerName oc-pizza.com
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

19. VHost management.oc-pizza.com

```
<VirtualHost *:80>
        ServerAdmin webmaster@oc-pizza.com
        DocumentRoot /var/www/ocpizza-management/OC-Pizza-Management/
        ServerName management.oc-pizza.com
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

20. Script de création de la base de données

Le fichier « script.sql » contient les requêtes pour :

1. Créer l'utilisateur
2. Attribuer les droit à l'utilisateur
3. Créer la base de données
4. Ajouter les tables et leurs relations

21. Commande CRON de backup

Le script « configure_backup.sh » paramètre le CRON qui est exécuté tous les jours à 3h00.

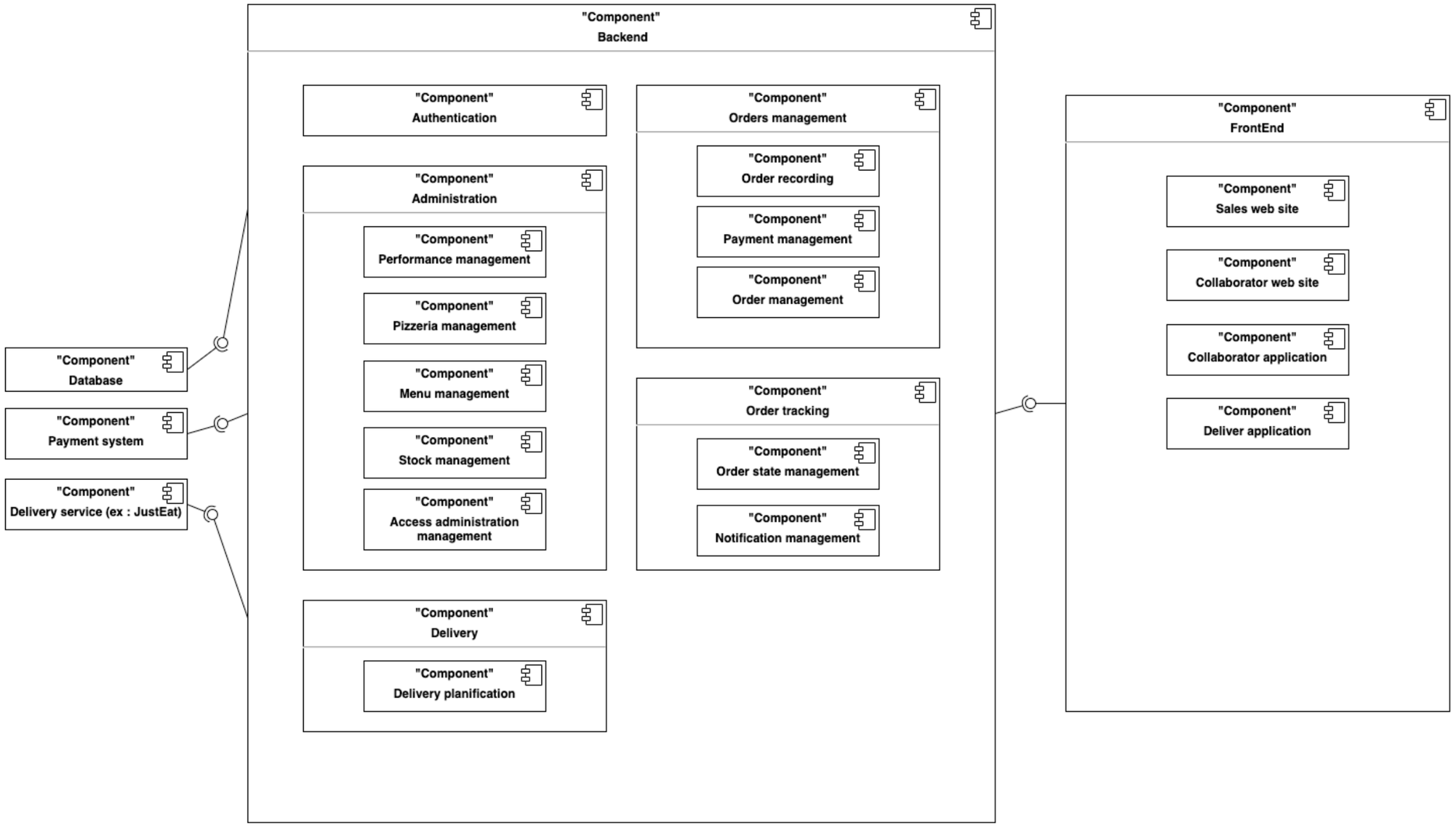
```
sudo rsync -arR --delete --files-from=/home/oc/.files_to_backup.txt / /home/oc/backup/
```

Les dossiers suivants sont automatiquement sauvegardés :

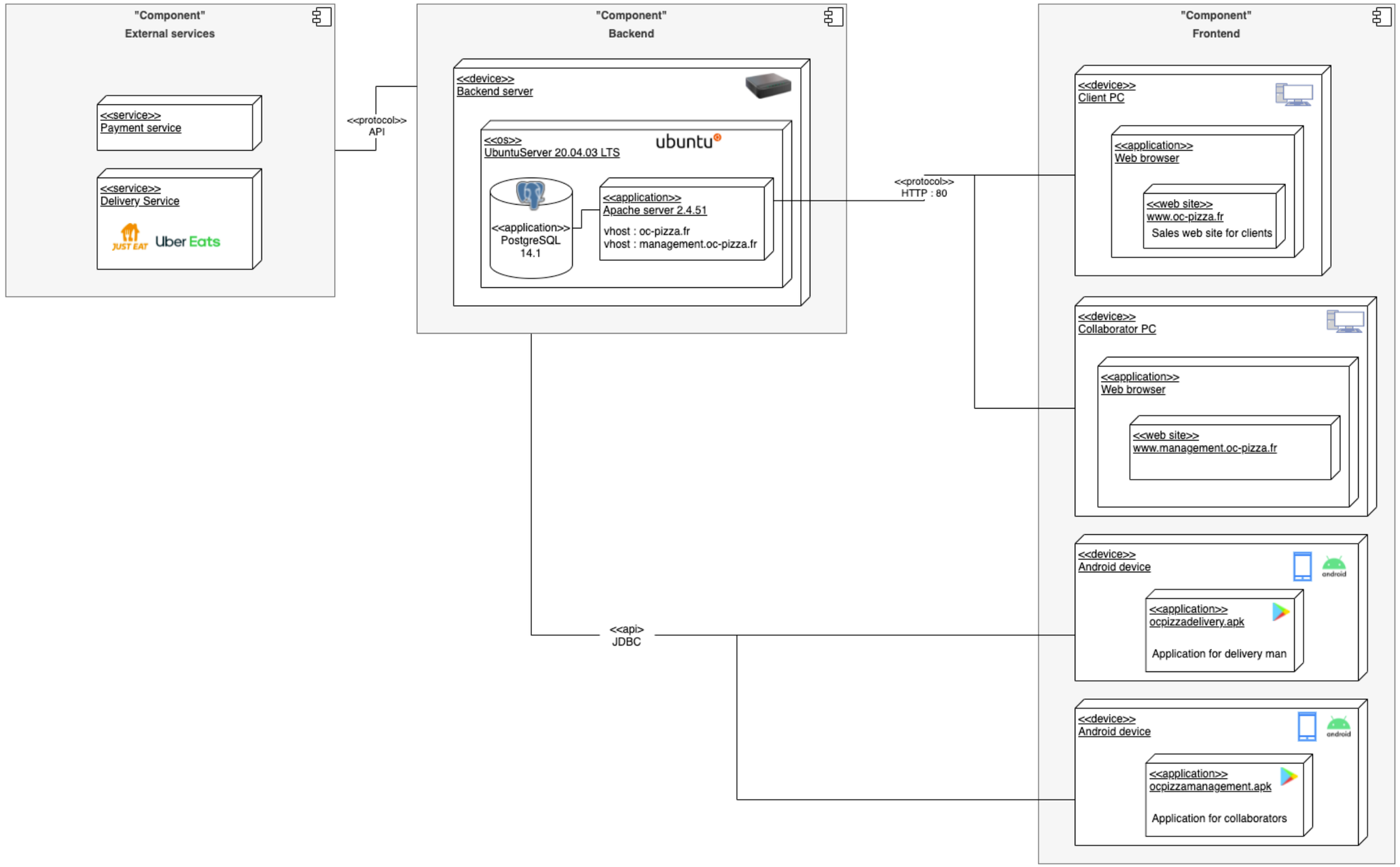
- /var/www/ocpizza
- /var/www/ocpizza-management
- /etc/apache2/sites-available/ocpizza.conf
- /etc/apache2/sites-available/ocpizza-management.conf
- /var/lib/postgresql/12/main

7 - Annexes

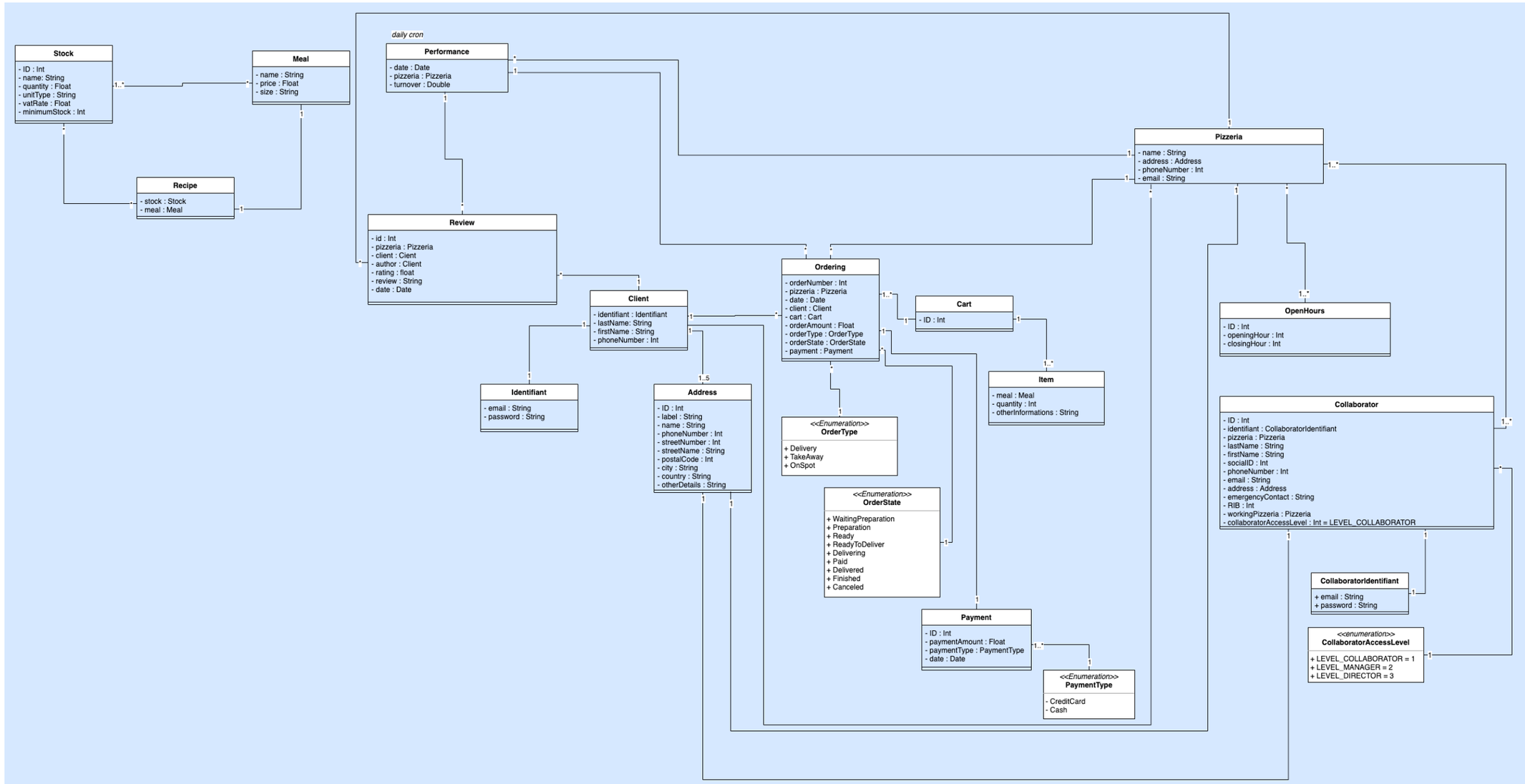
7.1 - Diagramme de composant


"Component"
Backend
"Component"
Authentication
"Component"
Administration
"Component"
Performance management
"Component"
Pizzeria management
"Component"
Menu management
"Component"
Stock management
"Component"
Access administration management
"Component"
Orders management
"Component"
Order recording
"Component"
Payment management
"Component"
Order management
"Component"
Order tracking
"Component"
Order state management
"Component"
Notification management
"Component"
Delivery
"Component"
Delivery planification
"Component"
Database
"Component"
Payment system
"Component"
Delivery service (ex : JustEat)
"Component"
FrontEnd
"Component"
Sales web site
"Component"
Collaborator web site
"Component"
Collaborator application
"Component"
Deliver application

7.2 - Diagramme de déploiement


"Component"
External services
<<service>>
Payment service
<<service>>
Delivery Service
JUST EAT
Uber Eats
<<protocol>>
API
"Component"
Backend
<<device>>
Backend server
<<os>>
UbuntuServer 20.04.03 LTS
ubuntu
<<application>>
PostgreSQL
14.1
<<application>>
Apache server 2.4.51
vhost : oc-pizza.fr
vhost : management.oc-pizza.fr
<<protocol>>
HTTP : 80
<<api>
JDBC
"Component"
Frontend
<<device>>
Client PC
<<application>>
Web browser
<<web site>>
www.oc-pizza.fr
Sales web site for clients
<<device>>
Collaborator PC
<<application>>
Web browser
<<web site>>
www.management.oc-pizza.fr
<<device>>
Android device
android
<<application>>
ocpizzadelivery.apk
Application for delivery man
<<device>>
Android device
android
<<application>>
ocpizzamanagement.apk
Application for collaborators

7.3 - Diagramme de classes



7.4 - Modèle physique de données