OC PIZZA

Nouveau Système Informatique d'OC Pizza

Dossier de conception technique

Version 1.0

Auteur
Emmanuel Plumas
Développeur Java Junior

TABLE DES MATIERES

1 - VERSIONS

Auteur	Date	Description	Version
Emmanuel Plumas	21/09/2020	Création du document	1.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier de conception technique de l'application « OC Pizza ».
Il a pour objectif de définir et spécifier l'architecture, les différents éléments, leurs interactions et leur déploiement, du nouveau système informatique de l'ensemble des pizzerias de la société « OC Pizza ».
Les éléments du présent dossier découlent :

- du dossier de conception fonctionnelle de l'application.

2.2 - Références

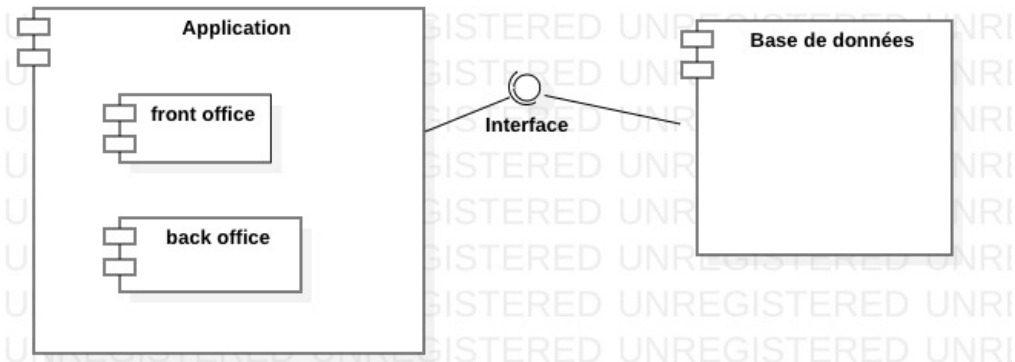
Pour de plus amples informations, se référer également aux éléments suivants :

1. **P8DOCPizza_01_fonctionnelle** : Dossier de conception fonctionnelle de l'application

3 - ARCHITECTURE TECHNIQUE

3.1 - Description de l'architecture

Diagramme de composants



La solution technique proposée est constituée d'une application qui inclura un back office et un front office. L'application communiquera avec une base de données qui contiendra toutes les données nécessaires au bon fonctionnement de la solution.

Le diagramme de composants décrit ici les composants fonctionnels.
La solution technique se veut être le reflet de l'application. L'application étant simple, la solution technique l'est également.

3.2 - Description de l'application Web

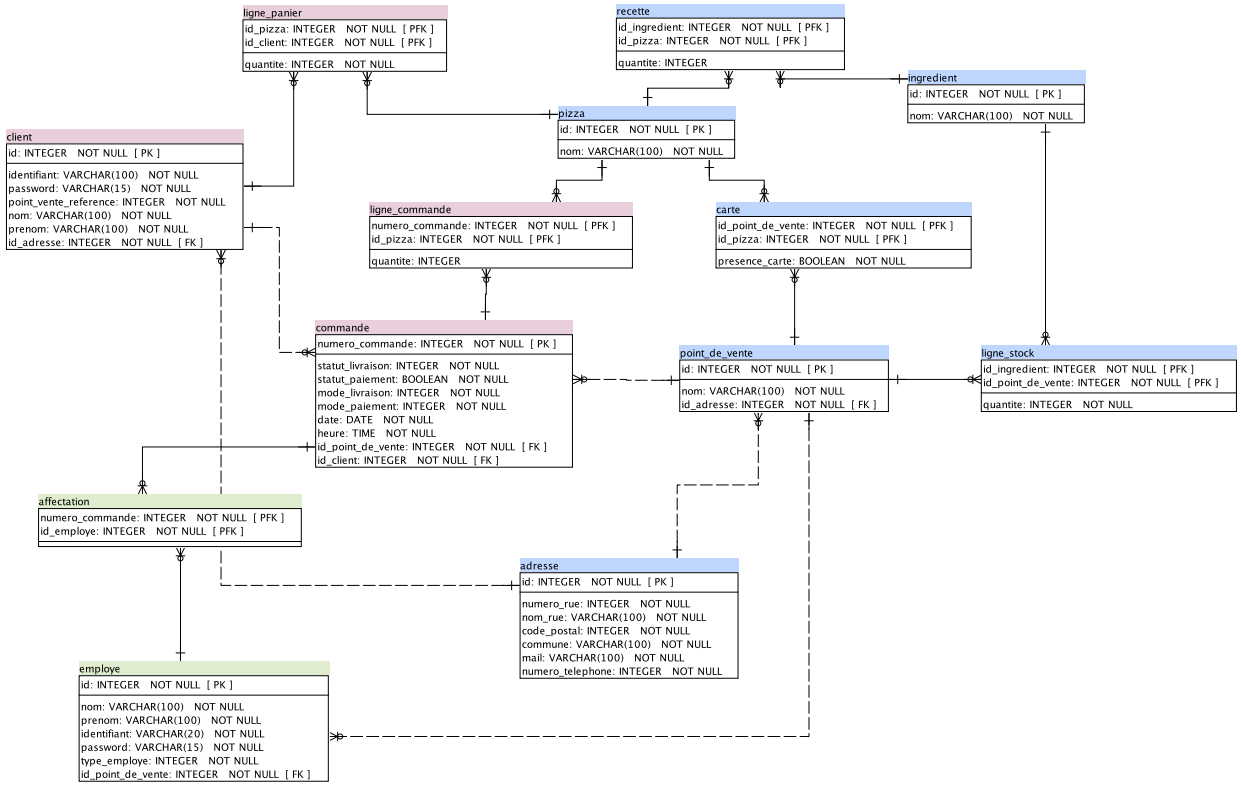
L'architecture retenue est une architecture monolithique. Les besoins du projet ne justifient pas plusieurs composants. Cette architecture garantira toutes les performances ainsi que l'évolutivité de la solution.
La pile logicielle est la suivante :

- Application JAVA (JDK version 1.8)
- Serveur d'application Tomcat 9.0
- Spring Boot 2.0

3.3 - La base de données

Les données de l'application seront persistées dans une base de données unique.
La base de données sera hébergée sur un serveur de Base de données PostgreSQL 11
La base de données sera implémentée à partir du Modèle Physique de Données ci-dessous :

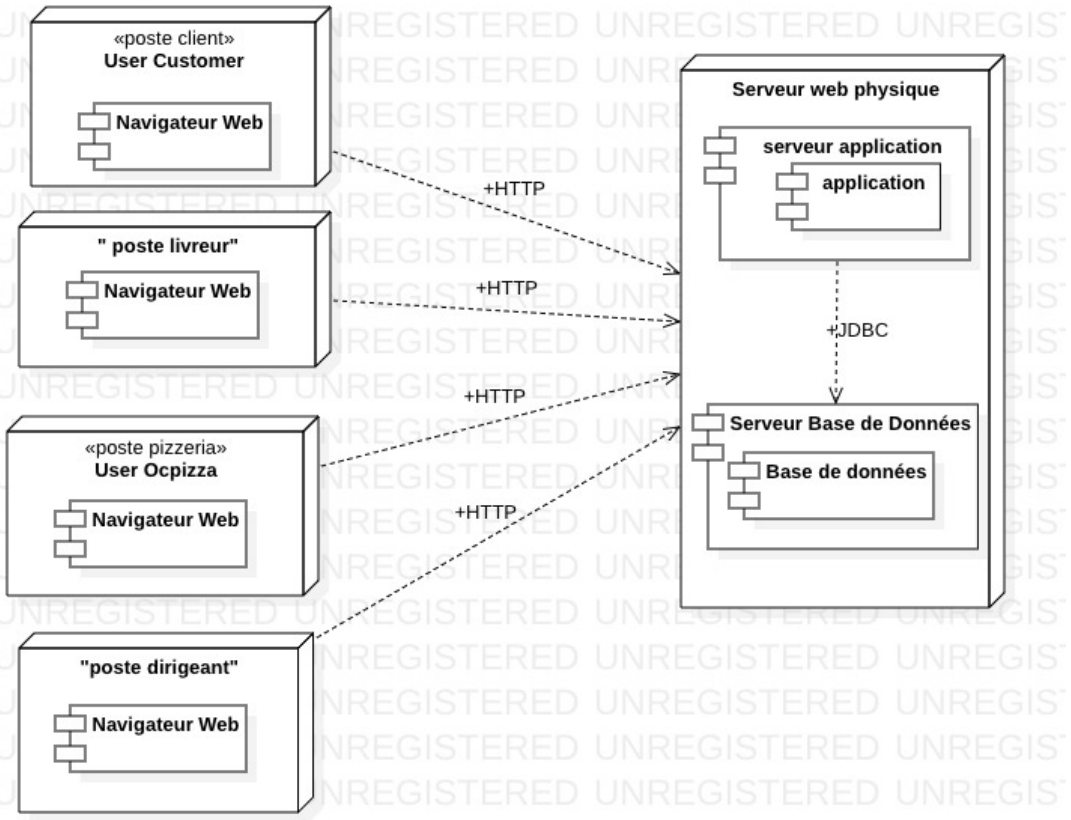
Modèle Physique de Données



4 - ARCHITECTURE DE DÉPLOIEMENT

Le déploiement global de l'application sera fait en respectant le diagramme de déploiement suivant :

Diagramme de déploiement



L'application ainsi que la base de données seront installés sur un serveur web. Le serveur d'application, qui contiendra l'application, sera implémenté sur le serveur web, tout comme le serveur de base de données, qui accueillera la base de données.

C'est l'application qui sollicitera la base de données (Création, Lecture, modification ou suppression de données).

L'application sera elle-même sollicitée par les différents postes qui auront accès à l'application via le navigateur web de leur poste. Quatre types de poste seront déployés :

- (Le poste Client)
- Le poste Livreur
- Le poste Utilisateur / Client
- Le poste Dirigeant

4.1 - Serveur de base de données

Serveur de base de données : PostgreSQL 11

4.2 - Serveur Web

Serveur d'application : Tomcat 9.0

5 - ARCHITECTURE LOGICIELLE

Versionning et sources : Les sources et versions du projet seront gérées par Git et seront hébergées sur un repo GitHub : **https://github.com/eplms/p8JavaOcPizza.git**

Gestion des dépendances et packaging : Les dépendances, ainsi que le packaging seront gérés par Apache Maven.

Architecture et modularité : L'application sera développée en respectant une Architecture en couche et consistera en un projet monomodule Maven.

Design pattern mis en œuvre :
- MVC
- Inversion de contrôle

Responsive : Le caractère responsive de l'application sera géré grâce à Bootstrap

Les frameworks utilisés
- Spring Boot
- Spring Data JPA
- Spring MVC
- Spring Boot Actuator
- Spring Security
- Thymeleaf
- Bootstrap

Création du projet : Spring Initialzr incluant les dépendances suivantes
- Spring Web
- Spring Data JPA
- Spring Security
- Spring Boot Actuator
- Thymeleaf
- PostgreSQL Driver

5.1 - Les couches

L'architecture applicative est la suivante :
- Couches **model** : couche gérant l'ensemble des objets qui seront persistés en Base de données. (implémentation du modèle des objets métiers). Chaque objet métier sera mappé avec la base de données grâce aux annotations de Spring Data Jpa.
- Couche **controller** : couche responsable du traitement des requêtes provenant des utilisateurs.
- Couche **service :** couche responsable de la logique métier de l'application
- Couche **repository** : couche responsable de l'accès aux données dans la Base de données.
- Couche **security** : couche responsable de la mise en œuvre de la sécurité du site.
- Couche **configuration** : couche responsable de la configuration de la sécurité.

Chacune de ces couches sera contenues dans un package du projet qui lui sera propre.

- Vues : Templates thymeleaf en charge de l'affichage des pages en front

5.2 - Le module

L'ensemble de ces couches sera contenu dans un seul et même module **Maven.**

5.3 - Structure des sources

La structuration des répertoires du projet sera la suivante :

```
racine
 ├─ pom.xml
 └─ src
     ├─ main
     │   ├ java
     │   │  └ package principal
     │   │       ├ P8JavaOcPizzaApplication
     │   │       │ package controller
     │   │       │ package model
     │   │       │ package service
     │   │       │ package repository
     │   │       │ package security
     │   │       └ package configuration
     │   │
     │   └ resources
     │        ├─ doc
     │        ├─ templates
     │        │      html1
     │        │      html2
     │        │      ..............
     │        ├  application.properties
     │        └  log4j2.xml
     │
     └─ test
         ├─ java
         └─ ressources
```

6 - POINTS PARTICULIERS

6.1 - Gestion des logs

La gestion des logs se fera grâce à **log4j2**.

Configuration :
Le fichier de configuration des logs ***log4j2.xml*** sera placé dans le dossier **resources** en respectant l'arborescence ci-dessous :

```
racine
 ├pom.xml
 └src
    ├─ main
    │   ├ java
    │   │  └ package principal
    │   │       ├ P8JavaOcPizzaApplication
    │   │       │ package controller
    │   │       │ package model
    │   │       │ package service
    │   │       │ package repository
    │   │       │ package security
    │   │       └ package configuration
    │   │
    │   └ resources
    │       ├─ doc
    │       ├─ templates
    │       │     html1
    │       │     html2
    │       │     ..............
    │       ├  application.properties
    │       └  log4j2.xml
    │
```

Les niveaux de logs retenus :
Parmi les niveaux de logs disponibles (DEBUG, INFO, WARN, ERROR, FATAL), les seuls utilisés seront :

- INFO
- ERROR : Au delà du précédent, il permettra de tracer les sources des erreurs.

Contenu :
Le message de log contiendra :

- Le jour
- L'heure
- Message propre à l'erreur qui l'a déclenché

Appenders :
Les logs seront sauvegardés quotidiennement dans un nouveau fichier **ocpizza** selon l'arborescence suivante

```
homedir
  └ logs
    └ ocpizza
```

6.2 - Fichiers de configuration

Le fichier de configuration est le fichier : ***application.properties***

Il sera placé dans le dossier **resources** en respectant l'arborescence ci-dessous :

```
racine
  ├pom.xml
  └src
     ├─ main
     │   ├ java
     │   │  └ package principal
     │   │       ├ P8JavaOcPizzaApplication
     │   │       │ package controller
     │   │       │ package model
     │   │       │ package service
     │   │       │ package repository
     │   │       │ package security
     │   │       └ package configuration
     │   │
     │   └ resources
     │       ├─ doc
     │       ├─ templates
     │       │      html1
     │       │      html2
     │       │      ..............
     │       ├  application.properties
     │       └  log4j2.xml
```

Le fichier de configuration contiendra les éléments suivants :

- **Données de connexion nécessaire à la base de données :**

```
spring.datasource.url=jdbc:postgresql://localhost:5432/db_p8_ocpizza
spring.datasource.username= db_p8_ocpizza
spring.datasource.password= passwordBaseDeDonnées(A completer)
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Le nom de la base de données ainsi que son mot de passe sont à définir et confidentielles.
Dans un premier temps, une base de données de test sera utilisée

- **Mise à jour de la base de données dans postgreSql en fonction du contenu des entities java**

```
spring.jpa.hibernate.ddl-auto=update
```

Le port auquel l'application sera disponible sera le port par défaut (localhost :8080)

6.3 - Environnement de développement

L'environnement de développement retenu est :

- Ide : Eclipse
- Java 8

- Maven 3.6.3

6.4 - Procédure de packaging/livraison

- Récupération le fichier jar dans le repository qui accueille le clone : **https://github.com/eplms/p8JavaOcPizza.git**
- Saisir la commande :

```
mvn package
```