

OC Pizza

Système de gestion de pizzerias

Dossier de conception technique

Version 1.0

Auteur

Laurie Compain
analyste-programmeur

TABLE DES MATIERES

1 - Versions.....	4
2 - Introduction	5
2.1 - Objet du document	5
2.2 - Références	5
3 - Architecture Technique	6
3.1 - Composants généraux.....	6
3.1.1 - Module WebStore	7
3.1.1.1 - Composant SearchEngine	7
3.1.1.2 - Composant Shopping Cart.....	7
3.1.2 - Module Outlet Management	7
3.1.2.1 - Composant Stock	7
3.1.2.2 - Composant Orders.....	7
3.1.3 - Module Accounting	7
3.1.3.1 - Composant Authentication	7
3.1.3.2 - Composant Users	7
3.1.3.3 - Composant Account.....	7
3.1.4 - Module Bank	8
3.1.4.1 - Composant Paiement.....	8
3.2 - API Web	8
3.2.1 - Modèle physique de données.....	8
3.3 - Web application	9
4 - Architecture de Déploiement.....	10
4.1 - Serveur de Base de données.....	10
4.1.1 - Caractéristiques techniques	10
4.2 - Serveur dédié hébergé API Web	11
4.2.1 - Caractéristiques techniques	11
4.3 - Serveur dédié hébergé Web application.....	11
4.3.1 - Caractéristiques techniques	11
5 - Architecture logicielle	12
5.1 - Principes généraux.....	12
5.1.1 - Les couches	12
5.1.2 - Structure des sources	12
6 - Points particuliers	14
6.1 - Gestion des logs.....	14
6.2 - Fichiers de configuration.....	14
6.2.1 - API Web	14
6.2.1.1 - Datasources	14
6.2.2 - Web application.....	14

6.3 - Ressources.....	14
6.4 - Environnement de développement.....	14
6.5 - Procédure de packaging / livraison.....	14
7 - Glossaire	16

1 - VERSIONS

Auteur	Date	Description	Version
Laurie	30/08/2021	Création du document	1.0

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier de conception technique de l'application de gestion d'OC Pizza.

Ce document a pour objectif de décrire le plus fidèlement et le plus clairement possible :

- Les composants propres à la solution ainsi que les composants extérieurs et l'ensemble des interactions
- Le déploiement de la solution proposée
- L'architecture logicielle
- Le Modèle Physique de Données

2.2 - Références

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

1. **PGESTpizza_01_dossier_spe_fonctionnelles**: Dossier de conception fonctionnelle de l'application
2. **PGESTpizza_03_Dossier_d_exploitation** : Dossier d'exploitation de l'application

3 - ARCHITECTURE TECHNIQUE

3.1 - Composants généraux

Le diagramme de composants décrit le système modélisé sous forme de composants réutilisables et met en évidence leurs relations de dépendance.

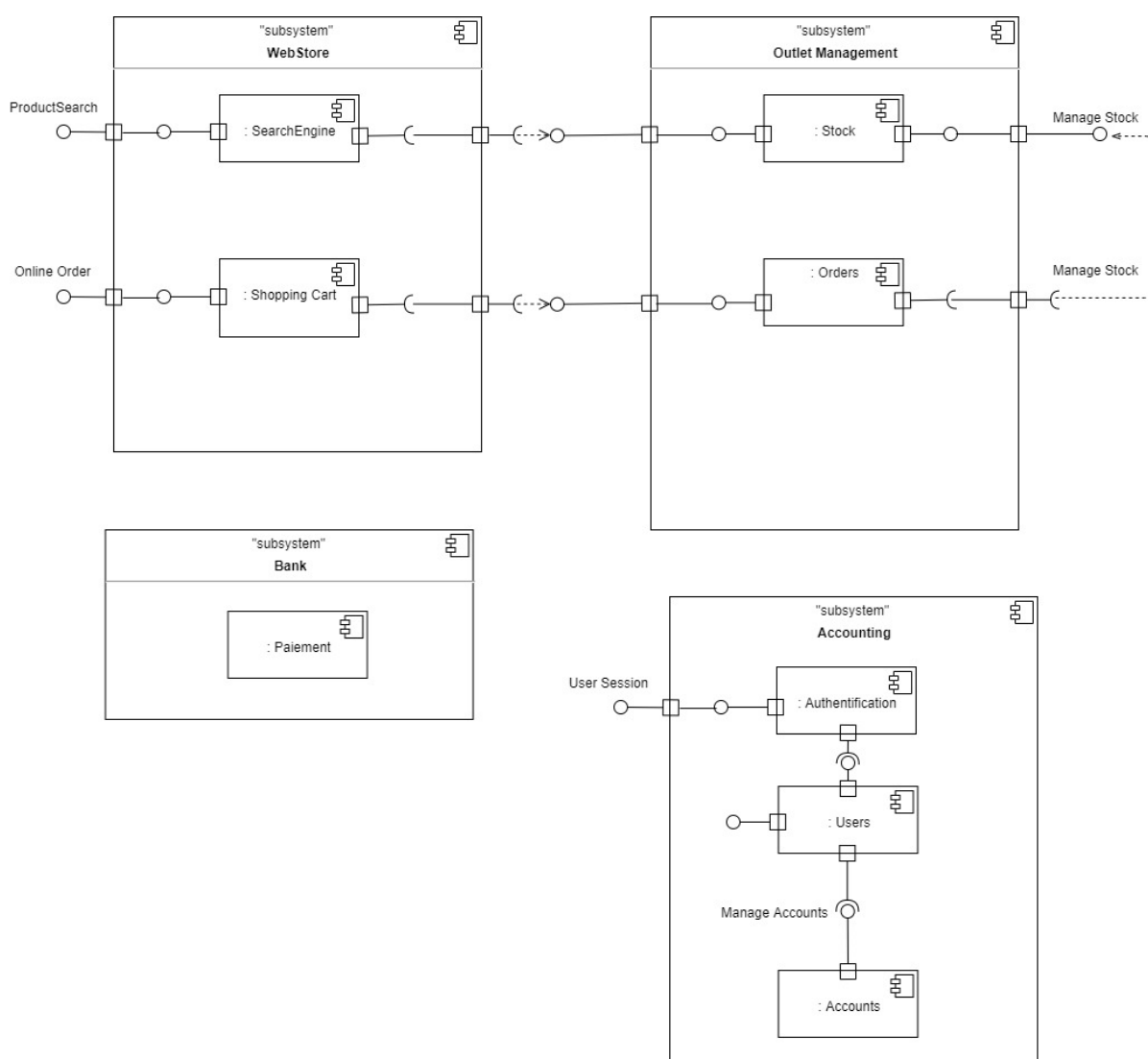


Figure 1 – Diagramme de composants

3.1.1 - Module WebStore

3.1.1.1 - Composant SearchEngine

Rôle : Recherche des produits proposés

Description : Permet d'afficher les produits, leurs informations

3.1.1.2 - Composant Shopping Cart

Rôle : Stocke les produits dans l'attente de la commande client

Description : Affiche au client les produits et leurs informations en attente de commande. Informe également le client sur le montant total de sa commande.

3.1.2 - Module Outlet Management

3.1.2.1 - Composant Stock

Rôle : Répertoire le stock des ingrédients et des pizzas

Description : Permet de connaître le nom des ingrédients et la quantité et ainsi savoir quelles pizzas peut être fabriquée

3.1.2.2 - Composant Orders

Rôle : Répertoire les commandes depuis leur validation par le client jusqu'à la livraison

Description : Permet d'afficher toutes les commandes en cours.

3.1.3 - Module Accounting

3.1.3.1 - Composant Authentication

Rôle : Authentification des utilisateurs

Description : Permet de connaître le type d'utilisateur et les droits qu'il possède

3.1.3.2 - Composant Users

Ce composant permet la gestion des utilisateurs

3.1.3.3 - Composant Account

Couplé au composant users, il permet la gestion des comptes utilisateur.

3.1.4 - Module Bank

3.1.4.1 - Composant Paiement

Le composant « paiement » est un composant extérieur au système, il permet d'accepter les paiements par carte bancaire.

Différentes solutions existent comme des fournisseurs tels que PayPal et Stripe mais vous pouvez opter pour une solution proposée par une banque traditionnelle comme par exemple E-transaction du Crédit Agricole. Toutes ces solutions ont comme point commun de fournir un module simple à intégrer à une application web ou à un applicatif

3.2 - API Web

La pile logicielle est la suivante :

- Application J2EE (JDK version 16)
- Serveur d'application Tomcat
- Base de données PostgreSQL
- Framework Spring

3.2.1 - Modèle physique de données

4 - Afin modéliser notre base de données, nous établissons un modèle physique de données (MPD). Celui-ci nous permet de mettre en évidence les relations entre les différentes tables

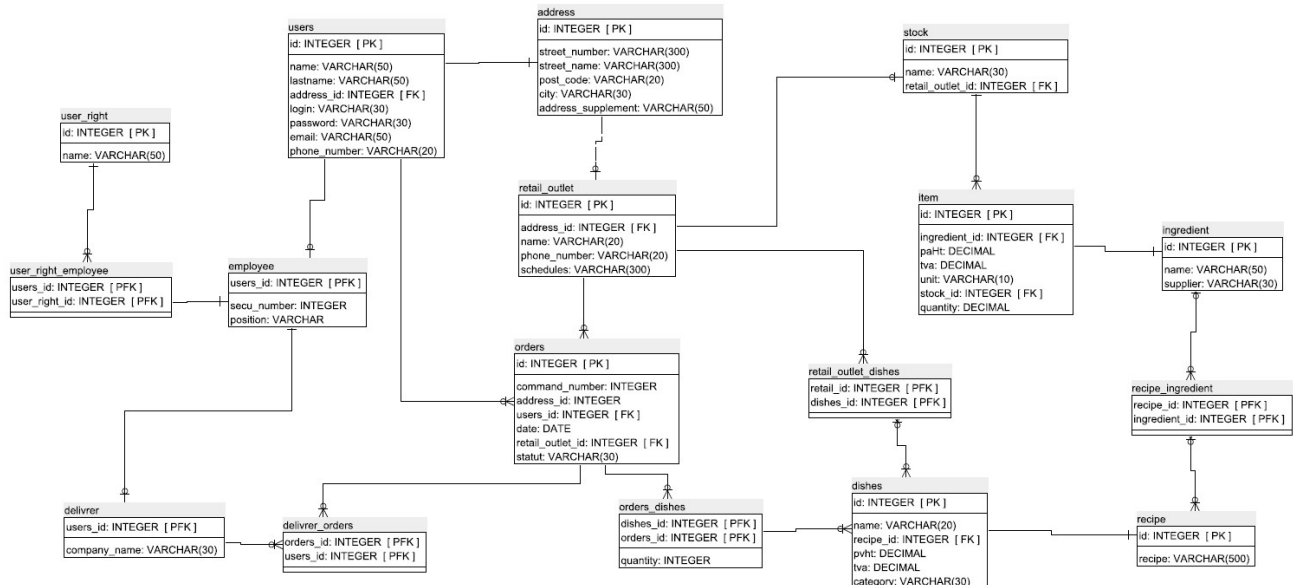


Figure 2 – Modèle physique de données

3.3 - Web application

La pile logicielle est la suivante :

- Application J2EE (JDK version 16)
- Serveur d'application Tomcat
- Framework Angular

4 - ARCHITECTURE DE DEPLOIEMENT

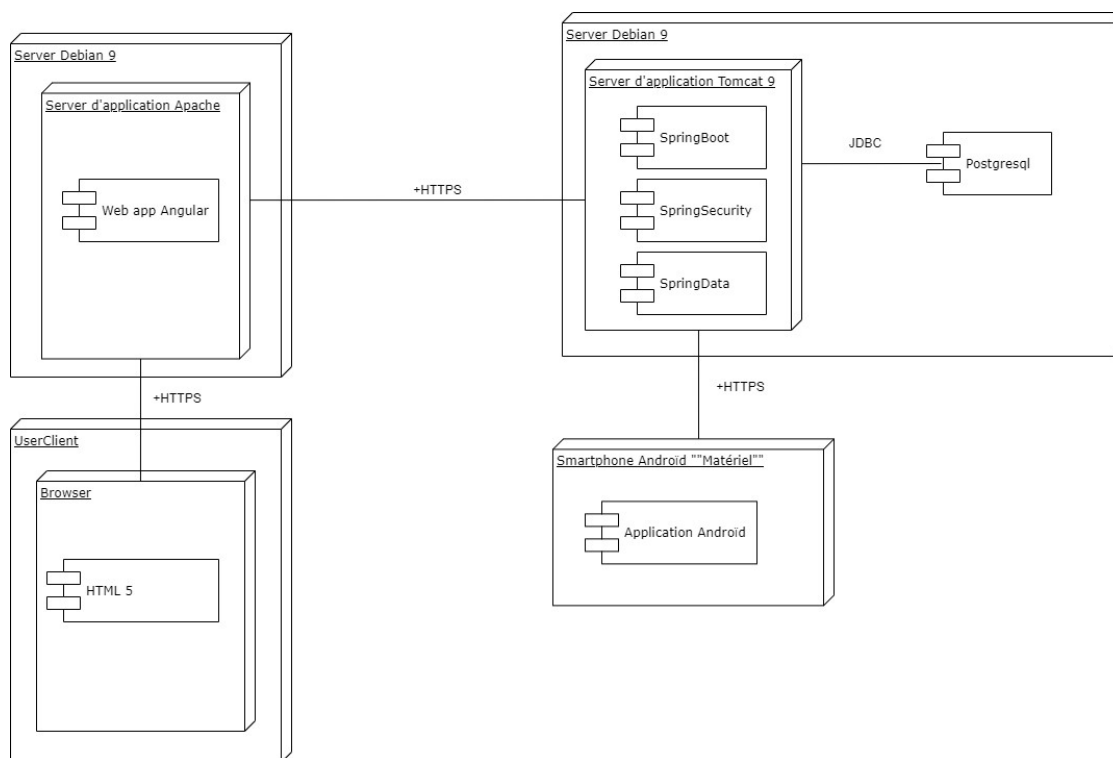


Figure 3 – Diagramme de déploiement

4.1 - Serveur de Base de données

La base de données sera déployée sur le serveur dédié où sera aussi installé le serveur d'application Apache Tomcat pour l'API Web

4.1.1 - Caractéristiques techniques

- OS : Debian 9
- SGBD : PostgreSQL 12

4.2 - Serveur dédié hébergé API Web

Le déploiement des applications web nécessitent un serveur dédié capable de prendre en charge un fort trafic, c'est pourquoi un serveur dédié sera loué auprès d'OVH.

4.2.1 - Caractéristiques techniques

- OS : Debian 9
- Serveur d'application : Apache Tomcat 9.0.26
- SGBD : PostgreSQL 12

4.3 - Serveur dédié hébergé Web application

Un deuxième server dédié sera loué auprès de ovh pour déployer la partie Frontend de notre application

4.3.1 - Caractéristiques techniques

- OS : Debian 9
- Serveur d'application : Apache Tomcat 9.0.26

5 - ARCHITECTURE LOGICIELLE

5.1 - Principes généraux

Les sources et versions du projet sont gérées par Git, les dépendances et le packaging par Apache Maven.

Le développement de l'application est un projet basé sur les Framework Spring Boot et Angular. Il est divisé en deux parties :

- web application basée sur le Framework Angular ;
- Une API Web en REST basée sur le Framework Spring Boot.

5.1.1 - Les couches

L'architecture applicative pour l'API web est la suivante :

- Une couche service : responsable de la logique métier du composant
- Une couche model : implémentation du modèle des objets métiers
- Une couche consumer : qui correspond à la couche d'accès aux données
- Une couche exposition : Gère les requêtes client. Point d'entrée de l'application
- Une couche security : Gère l'authentification des utilisateurs et la vérification des droits utilisateurs.

L'architecture applicative pour la web application est la suivante :

- Une couche service : responsable de la logique métier du composant
- Une couche model : implémentation du modèle des objets métiers
- Une couche consumer : qui correspond à la couche d'accès aux données (consomme les données de l'api web)
- Une couche controller : Gère les requêtes client et délègue le processus d'affichage aux vues qui sont dans la partie webapp de l'application

5.1.2 - Structure des sources

La structuration des répertoires du projet suit la logique suivante :

- les répertoires sources sont créés de façon à respecter la philosophie Maven

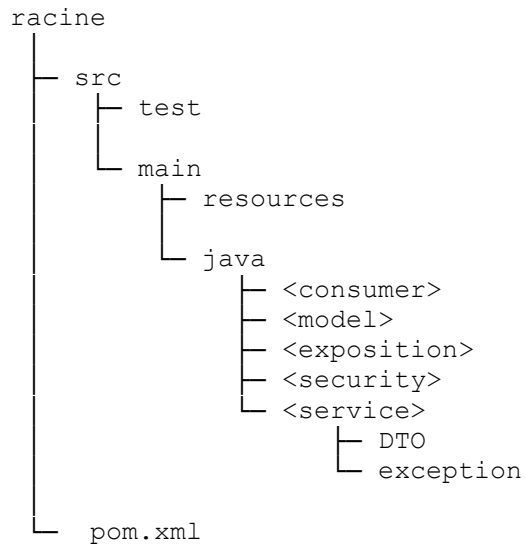


Figure 4 – Structure API Web

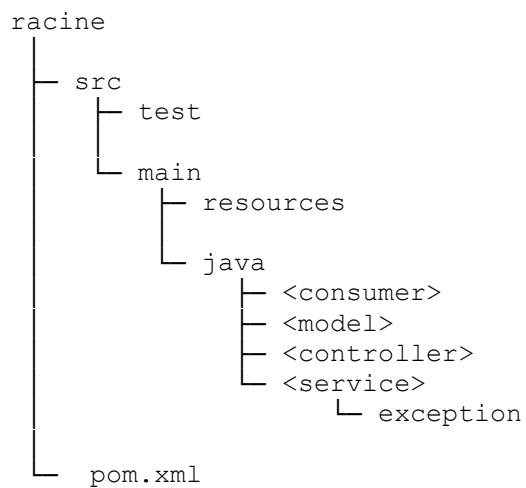


Figure 5– Structure web application

6 - POINTS PARTICULIERS

6.1 - Gestion des logs

La gestion des logs est assurée à l'aide de Log4j.

6.2 - Fichiers de configuration

6.2.1 - API Web

Le fichier application.yaml configure le webservice Spring. La base de données y est configurée.

6.2.1.1 - Datasources

Le fichier dump_db.sql permet de charger, au lancement de l'API un jeu de données.

6.2.2 - Web application

Au démarrage de l'application un fichier JSON se charge. Ce fichier décrit les paramètres du fichier « environment.ts ».

6.3 - Ressources

Les ressources sont stockées sur le serveur du client Web. Le client web accède à cet espace de stockage afin de charger ces différentes ressources.

6.4 - Environnement de développement

Ce Projet a été développé à partir des environnements Spring Boot 2.4.5, Angular et PostgreSQL 12.5 pour la base de données.

Afin de soigner l'affichage et que l'application soit responsive, la bibliothèque de composants de Bootstrap sera utilisée.

6.5 - Procédure de packaging / livraison

L'application sera packagée dans un fichier .war

Pour les scripts utilisés pour la création de la base de données, ils seront livrés dans un zip.



7 - GLOSSAIRE
