|  |
| --- |
| **OC Pizza**  **Projet 08**  Dossier de conception technique  Version 1.0 |
| **Auteur**  Laurent Cordier  *Développeur d'application* |

Table des matières

1 - Versions 4

2 - Introduction 5

2.1 - Objet du document 5

2.2 - Références 5

3 - Architecture Technique 6

3.1 - Composant commun 6

3.1.1 - Package Sécurité 6

3.1.1.1 - Composant Contrôle-accès 6

3.2 - Application Web e-Commerce 6

3.2.1 - La pile logicielle 6

3.2.2 - Diagramme UML de Composants 6

3.2.3 - Composant CtrlAccès 7

3.2.4 - Composant panierAchat 7

3.2.5 - Composants gestionCompte 7

3.3 - Application Web e-Management 8

3.3.1 - La pile logicielle 8

3.3.2 - Diagramme UML de Composants 8

3.3.3 - Composant Supervision 8

3.3.4 - Composant GestionAccès 8

3.3.5 - Composant CtrlAccès 9

3.4 - Application Web e-Production 9

3.4.1 - La pile logicielle 9

3.4.2 - Diagramme UML de Composants 9

3.4.3 - Composant x 9

3.4.4 - Composant c 9

3.5 - Application Serveur 10

3.5.1 - La pile logicielle 10

3.5.1.1 - Diagramme UML de Composants 10

3.5.2 - Service Frontal-Web 10

3.5.3 - Service Persistence-Crud 10

3.5.4 - Service Paiement-Api 10

4 - Architecture de Déploiement 11

4.1 - Serveur de Base de données 11

4.2 - Serveur XXX 11

5 - Architecture logicielle 12

5.1 - Principes généraux 12

5.1.1 - Les couches 12

5.1.2 - Les modules 12

5.1.3 - Structure des sources 13

5.2 - Application Web 14

5.3 - Application Xxx 14

6 - Points particuliers 15

6.1 - Gestion des logs 15

6.2 - Fichiers de configuration 15

6.2.1 - Application web 15

6.2.1.1 - Datasources 15

6.2.1.2 - Fichier xxx.yyy 15

6.2.2 - Application Xxx 15

6.3 - Ressources 15

6.4 - Environnement de développement 15

6.5 - Procédure de packaging / livraison 15

6.6 - XXX 15

7 - Glossaire 17

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| Laurent Cordier | 19/12/2019 | Création du document | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier de conception technique de l'application OC Pizza portant sur la mise en place d’un nouveau système informatique pour l’ensemble des pizzerias du groupe.

## Références

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

1. **DCF – Projet08** : Dossier de conception fonctionnelle de l'application.
2. **DEX** **– Projet08** : Dossier d’exploitation.

# Architecture Technique

## Composant commun

### Package Sécurité

#### Composant Contrôle-accès

Description et rôle/objectif...

## Application Web e-Commerce

### La pile logicielle

La pile logicielle est la suivante :

* Angular : 8.2.14, Angular/Material : 8.2.3, Node.js : 12.13.0, Typescript : 3.5.3

### Diagramme UML de Composants

Une image contenant texte

Description générée automatiquement

### Composant e-Commerce

Le composant e-Commerce représente un module au sens Angular qui est servi par le composant UML node.js sur le port 4201.

Note : le fonctionnement d’une application Angular et du serveur node.js ne sont pas décrits ici.

### Composant ctrlAccès

Le composant ctrlAccès représente un service et un composant, au sens Angular. Ce composant utilise l’interface api-gestionSecu exposée par le service Frontal-Web. Il produit les fonctions suivantes :

* Masque de saisie pour s’identifier avec contrôle des champs obligatoires,
* Bouton d’action pour se déconnecter.

### Composant panierAchat

Le composant panierAchat représente un service et un composant, au sens Angular. Il utilise les apis

* api-Catalogue,
* api-Paiement,
* api-gestionStock,
* api-commande,

exposées par le service Frontal-Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Confirmation de commande en affichant les boutons d’action de confirmation ou d’annulation.
* Choisir le mode paiement en proposant la liste déroulante :
  + Par chèque à la livraison,
  + Par CB.
* Pouvoir annuler la Commande tant qu’elle n’est pas en production en activant le bouton d’action « annulCmde ».
* Consulter le catalogue Groupe OC Pizza en affichant la liste des recettes disponibles avec case de saisie de nombre de pizza en regard de chaque ligne.
* Choisir une pizzeria en affichant la liste des pizzerias du groupe avec un radio bouton en regard de chaque ligne reçue.
* Créer un Panier d’achat en affichant un bouton dont l’activation place les recettes dont le nombre de pizzas est >0 dans le panier d’achat.

### Composant gestionCompte

Le composant gestionCompte représente un service et un composant, au sens Angular. Il utilise l’api api-gestionSecu exposée par le service Frontal-Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Créer un compte client en proposant un masque de saisie des informations requises pour la création d’un compte client.
* Gérer son compte client en affichant les informations modifiables du compte client en mode « modification autorisée » ainsi que les boutons d’action de confirmation ou annulation.

## Application Web e-Management

### La pile logicielle

La pile logicielle est la suivante :

* Angular : 8.2.14, Angular/Material : 8.2.3, Node.js : 12.13.0, Typescript : 3.5.3

### Diagramme UML de Composants

Une image contenant capture d’écran

Description générée automatiquement

### Composant e-Management

Le composant e-Management représente un module au sens Angular qui est servi par le composant UML node.js sur le port 4202.

Note : le fonctionnement d’une application Angular et du serveur node.js ne sont pas décrits ici.

### Composant Supervision

Le composant Supervision représente un service et un composant, au sens Angular. Il utilise l’api api-Supervision exposée par le service Frontal-Web. Il produit les fonctions suivantes :

* Visualiser les indicateurs d’activité de la pizzeria en affichant les données sous forme de tableau, issues du journal des commandes.
* Choisir une pizzeria pour en consulter l’activité en affichant la liste des pizzerias – disponible pour acteur Direction uniquement.
* Visualiser les indicateurs d’activité du groupe en affichant les données sous forme de tableau, issues de la consolidation des journaux des commandes des pizzerias.

### Composant GestionAccès

Le composant Supervision représente un service et un composant, au sens Angular. Il utilise l’api api-gestionSecu exposée par le service Frontal-Web pour transmettre ou recevoir les données. Il produit la fonction suivante :

* Demander ouverture/renouvellement compte collaborateur en proposant le masque de saisie des informations requises pour la création d’un compte collaborateur.

### Composant CtrlAccès

Le composant CtrlAccès représente un service et un composant, au sens Angular. Il utilise l’api api-gestionSecu exposées par le service Frontal- Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Masque de saisie pour s’identifier avec contrôle des champs obligatoires
* Bouton d’action pour se déconnecter

## Application Web e-Production

### La pile logicielle

La pile logicielle est la suivante :

* Angular : 8.2.14, Angular/Material : 8.2.3, Node.js : 12.13.0, Typescript : 3.5.3

### Diagramme UML de Composants

Une image contenant capture d’écran

Description générée automatiquement

### Composant e-Production

Le composant e-Production représente un module au sens Angular qui est servi par le composant UML node.js sur le port 4203.

Note : le fonctionnement d’une application Angular et du serveur node.js ne sont pas décrits ici.

### Composant ProductionPizza

Le composant ProductionPizza représente un service et un composant, au sens Angular. Il utilise les apis :

* api-aideEnLigne,
* api-fluxProduction,

exposées par le service Frontal- Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Liste des pizzas à livrer sous forme de tableau avec bouton radio en regard de chaque ligne,
* Liste des pizzas à produire sous forme de tableau avec bouton radio en regard de chaque ligne,

### Composant CtrlAccès

Le composant CtrlAccès représente un service et un composant, au sens Angular. Il utilise l’api api-gestionSecu exposées par le service Frontal- Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Masque de saisie pour s’identifier avec contrôle des champs obligatoires
* Bouton d’action pour se déconnecter

### Composant GestionStock

Le composant GestionStock représente un service et un composant, au sens Angular. Il utilise l’api api-GestionStock exposées par le service Frontal- Web pour transmettre ou recevoir les données. Il produit les fonctions suivantes :

* Liste des ingrédients de quantité inférieur au seuil avec liste déroulante de fournisseur et champ de saisie de quantité libellé avec l’unité (Kg, litre…)
* Bouton d’action pour confirmer l’envoi des bons de commandes.

## Application Serveur

### La pile logicielle

La pile logicielle est la suivante :

* SpringBoot 2.2.0 Release
* JDK version 11.04,
* PostgreSQL 12.1,
* [Tomcat Embedded](https://mvnrepository.com/artifact/org.apache.tomcat.embed/tomcat-embed-core) [9.0.27](https://mvnrepository.com/artifact/org.apache.tomcat.embed/tomcat-embed-core/9.0.27)
* Hibernate 5.4.6 Final
* Spring Security 5.2.0 Release

#### Diagramme UML de Composants

Une image contenant capture d’écran

Description générée automatiquement

### Composant Service-FrontalWeb

Le composant Service-FrontalWeb représente un service hébergé par le gestionnaire de container Tomcat Embedded, au sens Spring Boot.

Il consomme les apis :

* En lien avec le S.I Existant que sont : api\_LstPizzeria, api\_LstRecettes et api\_CtrlAccessInterne pour recevoir ou transmettre les informations relatives à :
  + Liste des pizzerias,
  + Liste des recettes du groupe,
  + Ouverture/Fermeture session de travail des collaborateurs.
* api\_GestSystPaiement pour transmettre les données nécessaires au paiement d’une commande au service d’interface entre le système de paiement et l’application OCPizza.
* api\_CptClient (dénomination simplifiée) qui se décompose en
  + api\_CptClientGet pour obtenir les informations Client dont le courriel et/ou le nom est passé en attribut de la requête.
  + api\_CptClientPost pour enregistrer un nouveau compte client ou modifier un existant.
* api\_Stock (dénomination simplifiée) qui se décompose en
  + api\_StockGet pour obtenir la liste des ingrédients par pizzeria avec leur quantité en stock
  + api\_StockPut pour mettre à jour le stock d’un ingrédient, par pizzeria.
* api\_FluxProd (dénomination simplifiée) qui se décompose en
  + api\_FluxProdGet pour obtenir la liste des commandes par pizzerias ainsi que leur état courant,
  + api\_FluxProdPost pour mettre à jour la liste des commandes par pizzerias,
* api\_Supervision pour obtenir le contenu du journal des commandes,
  + par pizzeria si l’attribut idPizz est valorisé,
  + pour le groupe, en consolidant l’ensemble du journal.

Il réalise les apis :

TODO

### Composant Service-Crud

Description et rôle/objectif...

### Composant Package Security

Description et rôle/objectif...

### Composant S.I existant

### Composant Service-IntfPaiement

Description et rôle/objectif...

# Architecture de Déploiement

Diagramme UML de déploiement

Explication / commentaires si nécessaires...

## Serveur de Base de données

Description

Caractéristiques techniques (ex: Serveur Linux Debian Jessie + PostgreSQL 9.6…)

Informations importantes / points particuliers

## Serveur XXX

...

# Architecture logicielle

## Principes généraux

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

...

### Les couches

L'architecture applicative est la suivante :

* unecouche **business** : responsable de la logique métier du composant
* unecouche **model** : implémentation du modèle des objets métiers
* …
  + …
* ...

### Les modules

Ex: modules Maven dans le cas d’application multi-module...

### 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 (à savoir : « convention plutôt que configuration »)

racine  
 ├─ *pom.xml*  
 ├─ <moduleX>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ resources  
 ├─ <moduleY>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java

│ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources  
 └─ src  
 └─ lib

* ...

## Application Web

Uml component dans l’optique archi logicielle (mes services vu sous forme uml avec interdependances)

Si besoin, diagramme UML de composants pour monter les différents modules et leur inter-dépendances

## Application Xxx

# Points particuliers

## Gestion des logs

…

## Fichiers de configuration

### Application web

...

#### Datasources

...

#### Fichier xxx.yyy

...

### Application Xxx

...

## Ressources

...

## Environnement de développement

## Procédure de packaging / livraison

## XXX

…

# Glossaire

|  |  |
| --- | --- |
|  |  |
|  |  |