|  |
| --- |
| GroupPizza  AppliPizzeria  Dossier de conception technique  Version 1.0 |
| **Auteur**  R.Yoann  Développeur |

**Table des matières**

**1 -Versions 3**

**2 -Introduction 4**

2.1 -Objet du document 4

2.2 -Références 4

**3 -Architecture Technique 5**

3.1 -Composants généraux 5

*3.1.1 -Package A 5*

3.1.1.1 -Composant X 5

3.1.1.2 -Composant Y 5

*3.1.2 -Package B 5*

3.1.2.1 -Composant Z 5

3.2 -Application Web 5

*3.2.1 -Composants X 5*

*3.2.2 -Composants Y et Z 5*

3.3 -Application XXX... 5

**4 -Architecture de Déploiement 6**

4.1 -Serveur de Base de données 6

4.2 -Serveur XXX 6

**5 -Architecture logicielle 7**

5.1 -Principes généraux 7

*5.1.1 -Les couches 7*

*5.1.2 -Les modules 7*

*5.1.3 -Structure des sources 7*

5.2 -Application Web 8

5.3 -Application Xxx 8

**6 -Points particuliers 9**

6.1 -Gestion des logs 9

6.2 -Fichiers de configuration 9

*6.2.1 -Application web 9*

6.2.1.1 -Datasources 9

6.2.1.2 -Fichier xxx.yyy 9

*6.2.2 -Application Xxx 9*

6.3 -Ressources 9

6.4 -Environnement de développement 9

6.5 -Procédure de packaging / livraison 9

6.6 -XXX 9

**7 -Glossaire 10**

* Versions

|  |  |  |  |
| --- | --- | --- | --- |
| **Auteur** | **Date** | **Description** | **Version** |
| R.Yoann | 17/07/2019 | Création du document | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* Introduction
* Objet du document

Le présent document constitue le dossier de conception technique de l'application pizzeria

Objectif du document est de décrire l'architecture du logiciel en passant par des diagrammes de déploiement et composant.

Les éléments du présents dossiers découlent :

* Un diagramme de composant
* Un diagramme de déploiement
* Les couches de l'applicatio
* Architecture Technique
* Composants généraux



* *Package "Boutique web"*
* ***Composant "Moteur de recherche"***

Permet la recherche de produits dans le catalogue des pizzerias, consulte le stock disponible de la pizzeria pour informer le client si le produit est disponible.

* ***Composant "Panier"***

Gère la commande du client, le client peut passer une commande avec plusieurs articles.

* ***Composant "Authentification"***

Gère le compte client. Le client doit posséder un compte pour remplir son panier et effectuer une commande

* *Package "Stock"*
* ***Composant "Inventaire"***

Gère le stock d'une pizzeria. Informe si les produits sont disponibles ou non.

* *Package "Commande"*
* ***Composant "Achat"***

Gère l'achat du client. Demande confirmation auprès de la banque du paiement et informe le stock d'un retrait d'une list de produits.

* *Package "Banque"*
* ***Composant "Paiement"***

Gère le paiment du client pour sa commande. Informe la commande sur le status de la transaction.

* Application Web

La pile logicielle est la suivante :

* Application **J2EE** (JDK version 1.8)
* Serveur d'application **Tomcat 9.0.19**
* Architecture de Déploiement



L'application sera déployé sur un serveur Debian 9, la base de données fonctionnera aussi sur celui-ci.

Le serveur Java est déployé via Tomcat.

La base de données est fonctionne avec PostgreSQL 11.

La communication avec l'application web se fait en HTTPS pour sécuriser les clients, elle se fait aussi en JSON avec des méthodes jQuery pour éviter un rechargement de la page.

Pour le paiement, l'application est mise en communication avec une page de la banque en HTTPS.

* Serveur de Base de données

**Debian**

**Caractéristiques techniques :** Debian 9

* Serveur

**Debian**

**Caractéristiques techniques :** Debian 9

* Architecture logicielle
* Principes généraux

Les sources et versions du projet sont gérées par **Git**, les dépendances et le packaging par **Apache Maven**, **Java/javascript** comme languages utilisées. Framework : **Spring/JQuery/Struts 2**

* *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
* une couche **batch** : implémente les batchs de l'application
* une couche **consumer** : implémentation des interactions avec des services extérieurs
* une couche **technical** :
* une couche **webapp** :
* *Les modules*



* *Structure des sources*

La structuration des répertoires du projet suit la logique suivante :

pizzeria  
 ├─ *pom.xml*  
 ├─ <batch>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ ├─ test  
 │ │ ├─ java  
 │ │ └─ resources

│ ├─ assembly

│ │

│ └─ data

│ └─ conf   
 ├─ <business>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources

├─ <consumer>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources  
 ├─ <webapp>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ ├─ ressources

│ │ └─ webapp  
 │ └─ test  
 │ ├─ java  
 │ ├─ ressources

│ └─ webapp  
 │   
 ├─ <model>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources  
 ├─ <technical>  
 │ ├─ *pom.xml*  
 │ └─ src  
 │ ├─ main  
 │ │ ├─ java  
 │ │ └─ resources  
 │ └─ test  
 │ ├─ java  
 │ └─ ressources  
 └─ src  
 └─ lib

* Points particuliers
* Gestion des logs

**Log4j2 :** Trace écrite dans le fichier logInfo.txt

**Niveau des logs :** WARN

* Fichiers de configuration
* *Application web*

struts.xml

web.xml

* ***Datasources***

src/main/webapp/META-INF/context.xml

* *Batch*

src/assembly/archive-deploy.xml

src/data/conf/config.properties

* ***Datasources***

src/data/conf/pizzeria-db.properties

* Ressources

**readme.md**

* Environnement de développement

**IntelliJ IDEA**

* Procédure de packaging / livraison

**Apache Maven**

* Glossaire

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