|  |
| --- |
| **OC PIZZA**  **Projet 8 : Documentez votre système de gestion de pizzeria**  Dossier d'exploitation  Version 1.0 |
| **Auteur**  AUBRUN Éric  *Analyste-programmeur* |

Table des matières

1 - Versions 3

2 - Introduction 4

2.1 - Objet du document 4

2.2 - Références 4

3 - Prérequis 5

3.1 - Système 5

3.1.1 - Serveur de base de données 5

3.1.2 - Serveur Web 5

3.1.3 - Serveur Batch et API 5

3.2 - Bases de données 5

3.3 - Web-services 6

3.3.1 - PAYMILL, API de paiement en ligne intégrant des bibliothèques Java 6

3.3.2 - JDBC (Java DataBase Connectivity) API d’accès aux bases de données relationnelles 6

3.3.3 - SMS API d’envoi de SMS en java via un SDK 6

3.3.4 - JMM : API de géolocalisation « Java MaxMind » 6

3.3.5 - JAAS (Java Authentication and Authorization Service) : API de gestion de l’authentification. 6

3.3.6 - JavaMail API pour la gestion de courrier électronique. 7

4 - Procédure de déploiement 8

4.1 - Déploiement des Batches 8

4.1.1 - Artefacts 8

4.1.2 - Variables d'environnement 8

4.1.3 - Configuration 8

4.1.3.1 - Vérifications 8

4.2 - Déploiement de l'Application Web 10

4.2.1 - Artefacts 10

4.2.2 - DataSources 10

4.2.3 - Vérifications 10

5 - Procédure de démarrage / arrêt 11

5.1 - Base de données 11

5.2 - Batches 11

5.3 - Application web 11

6 - Procédure de mise à jour 12

6.1 - Base de données 12

6.2 - Batches 12

6.3 - Application web 12

7 - Supervision/Monitoring 13

7.1 - Supervision de l’application web 13

8 - Procédure de sauvegarde et restauration 14

9 - Glossaire 15

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| Eric AUBRUN | 10/05/2021 | Création du document | 1.0 |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier d’exploitation de l'application OC PIZZA.

Ce document décrit les spécifications techniques du déploiement de cette application.

## Références

Pour de plus amples informations, se référer :

1. **PDOCPizza\_01\_fonctionnelle.pdf – 1.0:** Le Dossier de conception technique de l’application
2. **PDOCPizza\_02\_technique.pdf – 1.0** : Le Dossier de conception technique de l'application
3. **PDOCPizza\_04\_livraison.pdf – 1.0:** Le PV de livraison finale

# Prérequis

## Système

### Serveur de base de données

Le serveur de base de données est basé sur PostgreSQL 12.5. Il est hébergé sur un serveur de type linux Debian.

### Serveur Web

Le serveur est hébergé́ chez le fournisseur de l’application. Il héberge, la base de données, le service web.

L’ensemble des images (jpg) et des différentes ressources utilisées par le client sont hébergés sur ce serveur.

Ces services sont hébergés sur des serveurs Apache Tomcat.

### Serveur Batch et API

Le serveur de batch et de l’API accueille l’application permettant de commander automatiquement les produits et ingrédients. Il lisse sur chaque point de vente les différents niveaux de stock permettant la poursuite de l’activité sans interruption.

## Bases de données

La base de données doit être accessible lors du chargement du web service. La base de données est intitulée OCPizza.

Paramètres de connexion à la BDD :

url=jdbc:postgresql://localhost:5432/OCPizza

username=postgres

password=postgres

Si vous modifiez ces paramètres, vous devez modifier le fichier application.properties du service web.

## Web-services

Le fonctionnement de l’application nécessite que les web-services suivants soit activés et opérationnels :

### PAYMILL, API de paiement en ligne intégrant des bibliothèques Java

Un pont JavaScript va permettre de récupérer les informations de paiement de manière sécurisée.

### JDBC (Java DataBase Connectivity) API d’accès aux bases de données relationnelles

Toutes les classes de JDBC sont dans le package java.sql. Il est donc nécessaire de l’importer dans tous les programmes devant l’utiliser (import java.sql. \* ;).

### SMS API d’envoi de SMS en java via un SDK

Cette API est constituée de 3 méthodes : une méthode de réception de SMS, une méthode de création de SMS et une méthode batch d’envoi de SMS.

### JMM : API de géolocalisation « Java MaxMind »

Un import dans le POM parent ainsi qu’un chargement dans le Main de l’application permet d’avoir accès à la géolocalisation.

### JAAS (Java Authentication and Authorization Service) : API de gestion de l’authentification.

Cette API donne accès aux deux fonctionnalités d’authentification et d’autorisation de manière sécurisée.

### JavaMail API pour la gestion de courrier électronique.

Cette API permet de déployer la fonctionnalité d’envoi de mail aux clients de sorte que ceux-ci puissent avoir accès aux niveaux d’avancée de leurs commandes. Nous pouvons, si nous le souhaitons, injecter cette API dans un batch dans le but d’automatiser l’envoie de mail.

# Procédure de déploiement

## Déploiement des Batches

### Artefacts

Les batches de l’application OCPizza sont construits sous la forme d'une archive ZIP contenant les répertoires :

* **bin** : les scripts SH de lancement des différents batches et les fichiers jar
* **conf** : les fichiers de configuration

Il est nécessaire d’extraire l'archive **OCPizza.zip** et de positionner les droits d'exécution sur les scripts SH de lancement des batches.

### Variables d'environnement

Ces applications sont développées à l’aide du langage java. Voici les variables d'environnement reconnues par les batches de l’application OCPizza :

|  |  |  |
| --- | --- | --- |
| Nom | Obligatoire | Description |
| JAVA\_HOME | non | Répertoire racine de l’installation de l’application |

Ces services sont développés à l’aide de java 1.8. Votre version de java doit donc être au minimum la version 1.8.

### Configuration

Voici les différents fichiers de configuration :

* **Log4j.xml :** fichier de configuration des logs pour l’application Angular
* **application.properties :** fichier de configuration contenant notamment l’adresse du web service.

#### Vérifications

Afin de vérifier le bon déploiement du système, un batch de test permet de tester le bon fonctionnement de la base de données ainsi que du service web et de l’application web. Ces procédures de test peuvent être initiées à l’aide de Maven. Ils sont automatiquement exécutés dans le cadre du packaging du war.

## Déploiement de l'Application Web

### Artefacts

Zip

L’application se compose de deux artefacts. L’un attaché au service web (ServiceWeb.war) et un client web (SoEat.war).

Une fois la version finale de l’application validée, les applications peuvent être packagées à l’aide de Maven. Les applications sont développées à l’aide de github et sont disponible à l’adresse suivante : https://www.github/PJ8. La branche master de l’application est la version disponible de l’application.

La commande « mvn clean install » permet de packager le web service et le client struts sous forme de war.

### DataSources

Les accès aux bases de données doivent se configurer à l'aide du fichier application.properties Le fichier de drivers postgresql (postgresql-10) doit être déposé́ dans le répertoire :

$home\_server/lib/ext

Maven permet d’automatiser la gestion de ces ressources.

### Vérifications

Les vérifications sont opérées par les tests lors de la compilation de l’application à l’aide de Maven. Un rapport vous permet de savoir si les tests ont été exécutés correctement ou non.

# Procédure de démarrage / arrêt

## Base de données

## Batches

## Application web

# Procédure de mise à jour

## Base de données

## Batches

## Application web

# Supervision/Monitoring

## Supervision de l’application web

Afin de tester que l’application web est toujours fonctionnelles, faire ceci…

# Procédure de sauvegarde et restauration

# Glossaire

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