OC Pizza

Projet08

Dossier d'exploitation

Version 1.0

Auteur
Laurent Cordier
Développeur d'application

TABLE DES MATIERES

1 - VERSIONS

Auteur	Date	Description	Version
Laurent Cordier	24/12/2019	Création du document	0.1
Laurent Cordier	01/01/2020	Version livrable	1.0

2 - INTRODUCTION

2.1 - Objet du document

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

2.2 - Références

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

- **DCF – Projet08** : Dossier de conception fonctionnelle de l'application.
- **DCT – Projet08** : Dossier de conception technique de l'application.

3 - PREREQUIS

3.1 - Domain internet et DNS Dynamique

La mise en œuvre du DDNS, par exemple en décrivant le paramétrage de la box du FAI, ainsi que la location d'un domaine/sous-domaine auprès d'un registraire de nom de domaine comme « Google domains » avec paramétrage du DDNS , est hors périmètre.

Pour la suite du document, il est acquis :

- OC Pizza a enregistré les sous-domaines suivants auprès d'un registraire de son choix:
  - « ocpizza.fr »,
  - « ocpizza.management.fr »,
  - « ocpizza.prodcution.fr ».
- OC Pizza est abonné à un fournisseur de service Dyn DNS,
- Que le prestataire du FAI a réalisé la prestation de paramétrage de la « box »

Sur cette base :

- L'application e-commerce est disponible via l'URL :
  - http://www.ocpizza.fr
    - qui doit être convertit (NAT) en http://<add serveur interne> :4201/
- L'application e-management est disponible via l'URL :
  - http://www.ocpizza.management.fr/
    - qui doit être convertit (NAT) en http://<add serveur interne> :4202/

- L'application e-production est disponible via l'URL :
    - http://www.ocpizza.production.fr/
        - qui doit être convertit (NAT) en http://<add serveur interne> :4203/

3.2 - Système

3.2.1 - Système de gestion de base de données

Le SGBD PostgreSQL 12.1doit être installé et actif en tant que service Windows avec démarrage automatique. L'outil pgAdmin V4 doit être installé et fonctionnel.

3.2.2 - Serveur OC Pizza

OC Pizza exploite un serveur sous Windows server 2019.

3.3 - Base de données

Le répertoire c:\bdData doit exister et avoir les droits d'accès en rwx pour le compte « Système local ».

La base de données db_projet08 doit être accessible et à jour.

Le mot de passe requis pour la connexion au SGBD est celui choisi lors de son installation.

Le mot de passe requis pour la connexion à la base de connées db_projet08 est celui transmis pas courrier séparé par « IT Consulting » au DBA de « OC Pizza ».

3.4 - Serveur Web & applications

Le serveur Web node.js doit être installé dans sa version v12.14.0. LTS Le package est téléchargeable ici : https://nodejs.org/fr/ .

Enfin, les webapp suivantes doivent être accessibles et à jour (version dans le pv de livraison) :

- e-commerce
- e-production
- e-management

3.5 - Containers (services web)

Les containers suivants, sous forme de fichier JAR, doivent être accessibles et à jour (version dans le pv de livraison) :

- Service-FrontalWeb
- Service-Crud

- Service-intfPaiement

3.5.1.1 - Serveur Web

Le serveur WEB est Tomcat, embarqué dans chaque fichier JAR – produit avec Spring Boot.

3.5.1.2 - Prérequis techniques

- Java en version 11.0.4 doit être installé.
- Git bash en version 2.24.0.windows.2 doit être installé.

3.6 - Variables d'environnement

3.6.1 - Path

Il faut modifier le PATH général comme suit (valeur usuelle : C:\Program Files)

  - Pour les binaires JAVA : C:\Program Files\Java\jdk11.0.4\bin;
- Pour les binaires et utilitaires PostgreSQL :
  - C:\Program Files\PostgreSql\bin ;
  - C:\Program Files\PostgreSql\scripts;
- Pour Git Bash : C:\Program Files\Git\bin;
- Pour le serveur web node.js : C:\Program Files\nodejs\;

3.6.2 - Java_home

  - Pour java_home : C:\Program Files\Java\jdk-11.0.4

4 - PROCEDURE DE DEPLOIEMENT

4.1 - Déploiement de la base de données

4.1.1 - Préparation après livraison

L'état général doit être :

- PostgreSQL est lancé.
- Les webapp sont stoppées.
- Les services web sont stoppés.

Le livrable contient un fichier db.zip qu'il faut décompresser en respectant les chemins relatifs.

La structure est la suivante :

- ./sql_uml/prep.sql
- ./sql_uml/create_tbl.sql

4.1.2 - Déploiement (schémas, tables)

Exécuter dans l'ordre les commandes suivantes

- Sous pgsql,
  - exécuter le script ./sql_uml/prep.sql .
- Sous pgadmin,
  - attribuer le password xxxx au role rl_ocPizza (xxxx : fourni par IT Consulting par courrier séparé),
  - choisir la base db_ocPizza en vous y connectant avec le mot de passe identique au role rl_ocPizza,
  - enfin, exécuter sql_uml/create_tbl.sql

4.2 - Déploiement des services web

4.2.1 - Préparation après livraison

Le livrable contient un fichier servicesweb.zip qu'il faut décompresser en respectant les chemins relatifs.

La structure est la suivante :

- ./Service-FrontalWeb/service-web.jar
- ./Service-Crud/service-crud.jar
- ./Service-intfPaiement/service-intfpaiement.jar

4.2.2 - Déploiement

Ouvrir un terminal PowerShell avec droits admin dans chacun des répertoires. Puis exécuter dans chaque terminal :

- **jar -xf myapp.jar**

4.3 - Déploiement des applications Web

4.3.1 - Préparation après livraison

Le livrable contient un fichier webapp.zip qu'il faut décompresser en respectant les chemins relatifs.

La structure est la suivante :

- ./e-commerce/webapp
- ./e-production/webapp
- ./e-management/webapp

Note :

- Ces répertoires de chaque webapp contiennent chacun les fichiers nécessaires à l'exécution de la webapp. Leur contenu est créé avec ng build --prod. Cette commande produit un répertoire ./dist renommé selon les noms ci-dessus, destiné à la production.
- Il est donc normal de trouver des fichiers JavaScript et non pas TypeScript/Angular.

4.3.2 - Déploiement

Lors du premier déploiement, ouvrir un terminal powershell avec les droits admin.

Puis exécuter la commande pour un premier déploiement (inutile pour les mises à jour) :

- **npm install http-server -g**

5 - PROCEDURE DE DEMARRAGE / ARRET

5.1 - Base de données

PostgreSQL est installé en tant que service Windows en mode démarrage auto. La ruche HKEY_LOCAL_MACHINE contient :

- SYSTEM\CurrentControlSet\Services\postgresql-x64-12
    - DisplayName : postgresql-x64-12 - PostgreSQL Server 12
    - Start : 2

5.1.1 - Démarrage

Exécuter l'application services.msc. Dans la fenêtre qui s'ouvre, repérer la ligne PostgreSql. Cliquez pour ouvrir la fenêtre pop-up des options. Cliquez sur démarrer puis ok.
Le service doit être dans l'état « started ».

5.1.2 - Arrêt

Exécuter l'application services.msc. Dans la fenêtre qui s'ouvre, repérer la ligne PostgreSql. Cliquez pour ouvrir la fenêtre pop-up des options. Cliquez sur arréter puis ok.
Le service doit être dans l'état « stopped ».

5.2 - Services Web

5.2.1 - Rappel de la structure

La structure est la suivante :

- ./Service-FrontalWeb/service-web.jar
- ./Service-Crud/service-crud.jar
- ./Service-intfPaiement/service-intfpaiement.jar

5.2.2 - Démarrage

Ouvrir un terminal PowerShell avec droits admin dans chacun des répertoires. Puis exécuter dans chaque terminal :

- **java org.springframework.boot.loader.JarLauncher**

La dernière ligne affichée doit indiquer que le service a démarré.

5.2.3 - Arrêt

CTR-C dans la fenêtre terminal du service à arrêter.

5.2.4 - Pour le contrôle et la vérification

Dans un navigateur, appelez les url :

http://localhost:9201/actuator/health/ pour vérifier le bon fonctionnement du service-FrontWeb

http://localhost:9202/actuator/health/ pour vérifier le bon fonctionnement du service-IntfPaiement

http://localhost:9203/actuator/health/ pour vérifier le bon fonctionnement du service-Crud

Le navigateur affiche :

- `{"status":"UP"}` lorsque la santé du service est normal
- `{"status":"DOWN"}` lorsque le service n'est pas opérationnel.
- En cas d'erreur 404, le service est arrêté ou défaillant.

En cas d'anomalie, ouvrir un redmine et joindre le log du service défaillant.

5.3 - Applications web

5.3.1 - Rappel de la structure

La structure des répertoires est la suivante :

- ./e-commerce
- ./e-production

- ./e-management

5.3.2 - Démarrage

Ouvrir un terminal PowerShell avec droits admin dans chacun des répertoires. Puis exécuter dans chaque terminal :

- **http-server ./webapp -p xxxx où** xxxx est le port d'écoute de la webapp (cf DCT)

5.3.3 - Arrêt

CTR-C dans la fenêtre terminal de la webapp à arrêter.

5.3.4 - Pour le contrôle et la vérification

- Ouvrir un navigateur chromium.
- Choisir paramètres/outils supplémentaires/outils de développement.
- Dans la fenêtre d'outils de développement, choisir l'onglet console.
- Saisir l'adresse web de la webapp à vérifier.

La fenêtre d'accueil de la webapp à vérifier s'affiche ou le navigateur indique un code d'erreur respectant les standards.

La console de l'outils de développement affiche les traces applicatives en cas d'erreur.

Si nécessaire, copier l'écran, ouvrir un redmine et joindre la copie d'écran.

6 - PROCEDURE DE MISE A JOUR

6.1 - Base de données

Se reporter au & 4.1 -Déploiement de la base de données

6.2 - Services Web

Se reporter au & 4.2 -Déploiement des services web1 -

6.3 - Applications web

Se reporter au & 4.3 - Déploiement des applications Web

7 - SUPERVISION/MONITORING

7.1 - Supervision des services web

Ouvrir un terminal Git Bash dans chacun des répertoires. Puis exécuter dans chaque terminal :

- Tail -f <nomdufichierjar>.log

Au démarrage, la dernière ligne doit indiquer que le service a démarré.

En production, le service enregistre son activité.
Les cas suivants sont anormaux et doit déclencher un redmine – y joindre le log :

- Une ligne qui contient :
  - le libellé ERROR ou WARN,
  - un stack trace,
  - ou une absence de mise à jour régulière,
- Le terminal où a été lancé le service montre que le service est arrêté (retour à la ligne de commande du shell).

8 - SAUVEGARDE ET RESTAURATION

8.1 - Périmètre

L'exploitation d'une base de données et en particulier du SBGD PostgreSQL est hors périmètre de ce document.

Les informations qui concernent les sauvegardes et restaurations ainsi que le détail de leur mise en œuvre sont présentées ici : https://doc.postgresql.fr/9.5/continuous-archiving.html

Note :

- Le fichier `recovery.conf n'existe` plus en version 12.
- Le fichier `postgresql.conf` contient les informations précédemment contenues dans l'ancien fichier `recovery.conf`.

Les paragraphes suivants présentent une commande de sauvegarde et une méthode de restauration.

8.2 - commande de sauvegarde

La commande présentée correspond à une sauvegarde physique au fil de l'eau à l'aide de l'outil, interactif ou non, pg_basebackup fourni avec PostgreSQL.

Le répertoire « c:/dataSauveNAS/repSauve » est créé par l'administrateur système qui se charge de

la sauvegarde journalière sur support externe. La nature du support externe (Cloud, NAS, robot de DON, …) est hors périmètre de ce document.

La commande dans sa version interactive est :

pg_basebackup

--pgdata= c:/dataSauveNAS/repSauve/ « jjmmyyyy »

--format=tar

--write-recovery-conf

--xlog-method=stream

--progress

--host=localhost

--port= 5432

--username=rl_projet08

--password

où jjmmyyyy est à valoriser avec la date de la sauvegarde journalière. Le password a été transmis par document séparé par IT Consulting lors de la livraison.

Le répertoire c:/dataSauveNAS/repSauve /« jjmmyyyy » contient les fichiers base.tar, pg_wal pour les journaux de transaction et un fichier xxxx.tar qui correspond au tablespace des tables de la base de données db_projet08.

Le fichier `Postgresql.conf` se trouve dans le fichier base.tar de la sauvegarde.

8.3 - Procédure de restauration

1. Arrêter le serveur s'il est en cours d'exécution.
2. Si la place nécessaire est disponible, copier le répertoire complet de données du cluster et tous les *tablespaces* dans un emplacement temporaire en prévision d'un éventuel besoin ultérieur. Cette précaution nécessite qu'un espace suffisant sur le système soit disponible pour contenir deux copies de la base de données existante. S'il n'y a pas assez de place disponible, il faut au minimum copier le contenu du sous-répertoire `pg_xlog` du répertoire des données du cluster car il peut contenir des journaux qui n'ont pas été archivés avant l'arrêt du serveur.
3. Effacer tous les fichiers et sous-répertoires existant sous le répertoire des données du cluster et sous les répertoires racines des *tablespaces*.
4. Restaurer les fichiers de la base de données à partir de la sauvegarde des fichiers. Il faut veiller à ce qu'ils soient restaurés avec le bon propriétaire (l'utilisateur système de la base de données, et non pas `root` !) et avec les bons droits. Si des *tablespaces* sont utilisés, il faut s'assurer que les liens symboliques dans `pg_tblspc/` ont été correctement restaurés.

5. Supprimer tout fichier présent dans `pg_xlog/` ; ils proviennent de la sauvegarde et sont du coup probablement obsolètes. Si `pg_xlog/` n'a pas été archivé, il suffit de recréer ce répertoire en faisant attention à le créer en tant que lien symbolique, si c'était le cas auparavant.
6. Si des fichiers de segment WAL non archivés ont été sauvegardés dans l'étape 2, les copier dans `pg_xlog/`. Il est préférable de les copier plutôt que de les déplacer afin qu'une version non modifiée de ces fichiers soit toujours disponible si un problème survient et qu'il faille recommencer.
7. Modifier le fichier de commandes postgresql.conf avec restore_command = 'copy " c:\\dataSauveNAS/\\repSauve \\« jjmmyyyy »\\pg_wal.tar" "%p"'
8. Démarrer le serveur. Le serveur se trouve alors en mode récupération et commence la lecture des fichiers WAL archivés dont il a besoin. A la fin du processus de récupération, le serveur modifie postgresql.conf en `commentant la ligne` restore_command (pour éviter de retourner accidentellement en mode de récupération), puis passe en mode de fonctionnement normal.

9 - GLOSSAIRE

Libelle	Définition
DBMS	Système de base de données
SGBD	Système de base de données
Tomcat Embedded	Inclusion du serveur d'application dans un fichier archive format JAR