OC Pizza

Système informatique OC PIZZA

Dossier d'exploitation

Version 1.0

Auteur
Maryam Launois
Analyste-programmeur

Table des matières

1 VERSIONS

Auteur	Date	Description	Version
Maryam Launois	09/05/2020	Création du document	1.0

2 INTRODUCTION

2.1 Objet du document

Le présent document constitue le dossier d'exploitation du système informatique de gestion et de vente en ligne de pizza. L'application a pour but de centraliser toutes les informations concernant la gestion des catalogues, des achats, des commandes et leur statut, des livraisons, des comptes clients et des comptes employés.

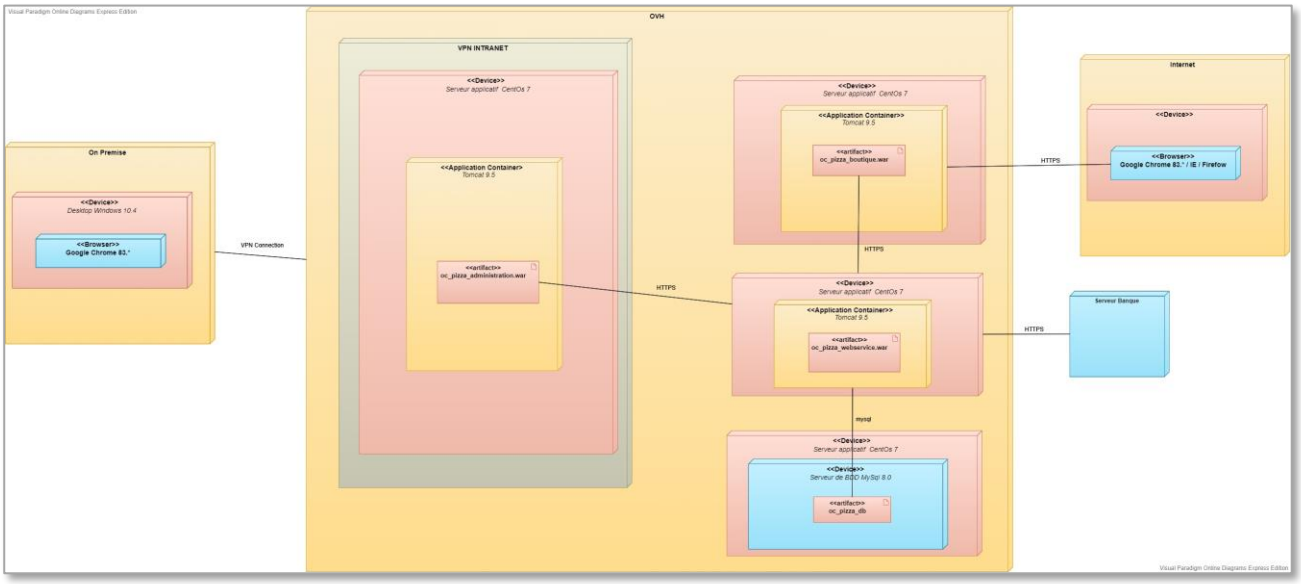
2.2 Références

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

1. **OC Pizza Système informatique** : Dossier de conception fonctionnelle
2. **OC Pizza Système informatique** : Dossier technique

3 PREREQUIS

3.1 Diagramme de déploiement



Le nouveau système informatique de OC pizza sera composé de :

- un site internet qui sera hébergé sur un serveur externe,
- une application intranet qui contiendra toute la logique métier avec un accès privé via vpn,
- et d'une api web qui sera hébergé en externe.

3.1- Système

3.1.1 Technologies requises

Pile technique :

• Système d'exploitation : **Centos 7**
• Application : **J2EE (JDK version 11) / HTML5 / CSS3 /Javascript**
• Serveur d'application : **Apache-Tomcat 9.5**
• Moteur de base de données : **Mysql 8.0**

3.1.2 Serveurs

Le système informatique du client sera composé :

- D'un site web que les clients pourront consulter et sur lequel ils pourront effectuer des commandes.
- Un site intranet pour la gestion des comptes employés ainsi que la gestion administrative.

- Un webservice qui permettra d'alimenter en ressources le site web et le site en interne.

Le webservice communiquera avec la base de données dont nous verrons les détails dans la partie 3.1.2.1.

3.1.2.1 Serveur web

Dans le tableau ci-dessous, vous trouverez le récapitulatif des différents serveurs qui seront utilisés et installés pour chacun des sites.

	Hébergement	**Serveur applicatif**	**Système d'exploitation**
Application métier	Interne VPN vers OVH	Tomcat 9.5	CentOs 7
Site web, front	OVH	Tomcat 9.5	CentOs 7
Api web	OVH	Tomcat 9.5	CentOs 7

3.1.2.1.1 Installation de Tomcat

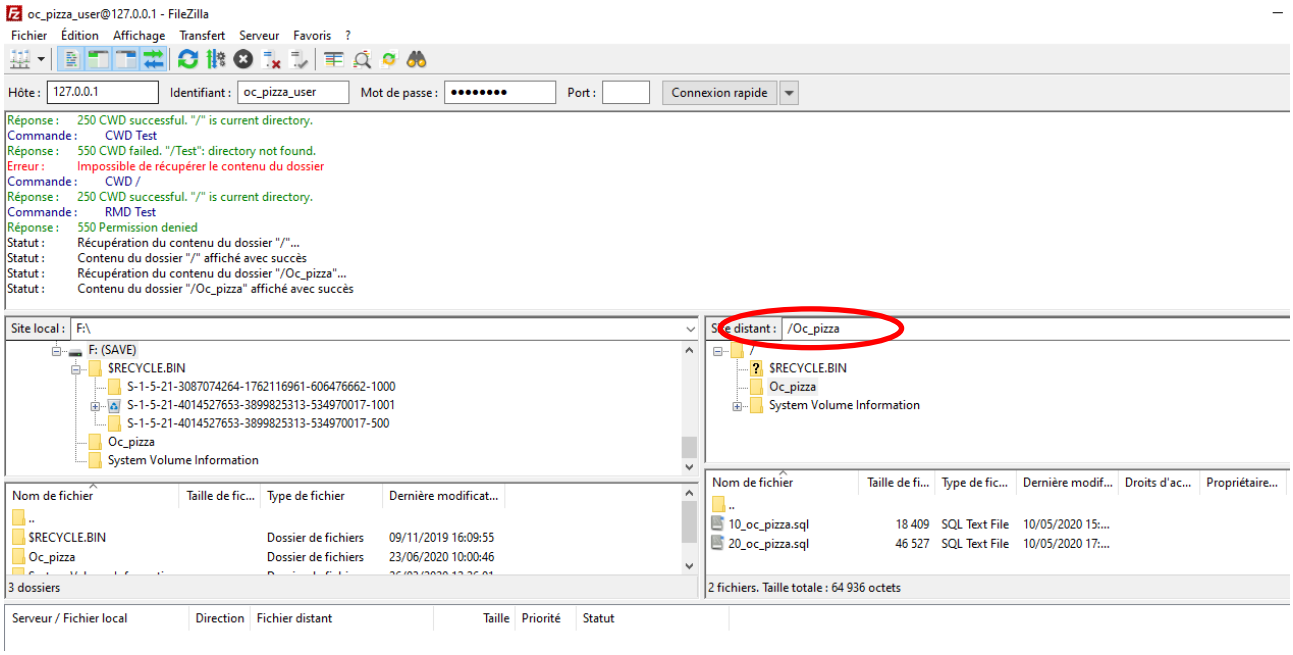
3.1.2.2 Base de données

Nom du schéma	oc_pizza_db
Moteur	Mysql 8.0
Hébergement	OVH
Serveur sur lequel la base est installée	CentOs 7
Port	3306
Script de création du schéma	10_oc_pizza.sql
Script d'insertion des données	20_oc_pizza.sql
Identifiant de connexion	Utilisateur ayant droit
Mot de passe de connexion	Utilisateur ayant droit

Remarque :

Les opérations de production s'effectueront avec un utilisateur ayant des droits de création et d'insertion.

3.1.3 Serveur de Fichiers

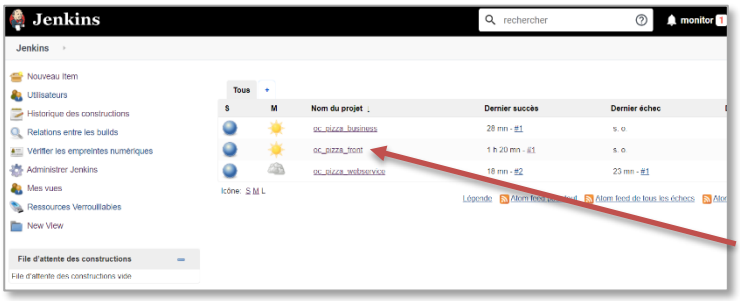


Pour la récupération des scripts sql :

- Se connecter au client Filezilla avec l'utilisateur oc_pizza_user et son mot de passe. (utilisateur ayant les droits)
- Se rendre dans le dossier distant /**Oc_pizza**.
- Récupérer le script de création de la base de données **10_oc_pizza.sql** et le script d'insertion de données **20_oc_pizza.sql**.

4 PROCEDURE DE DEPLOIEMENT

4.1 Construction de l'application avec Jenkins

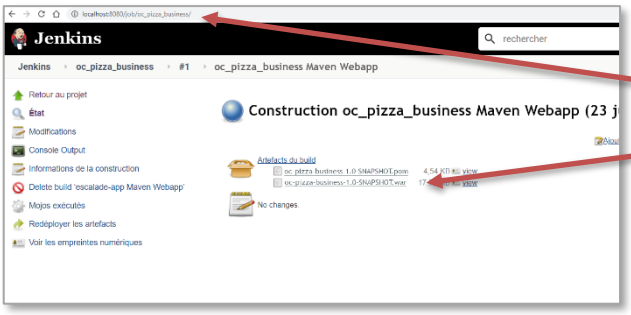


L'application se compose :

- d'une application métier,
- d'une application front et
- d'un webservice.

Des jobs ont été créés pour chacune des applications.

4.1.1 Job de l'application métier

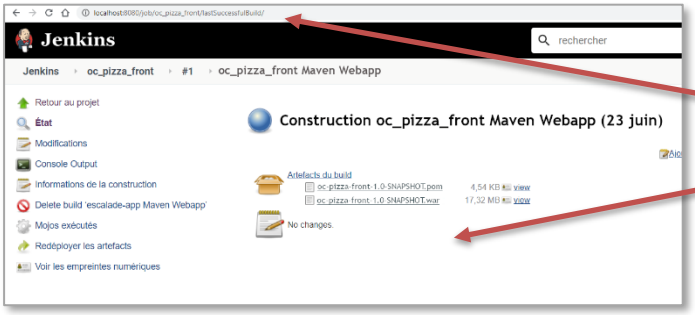


Capture d'un job réussi de l'application métier :

- Url pour accéder au job
- Artefacts correspondants.

http://localhost:8080/job/oc_pizza_business/

4.1.2 Job de l'application front

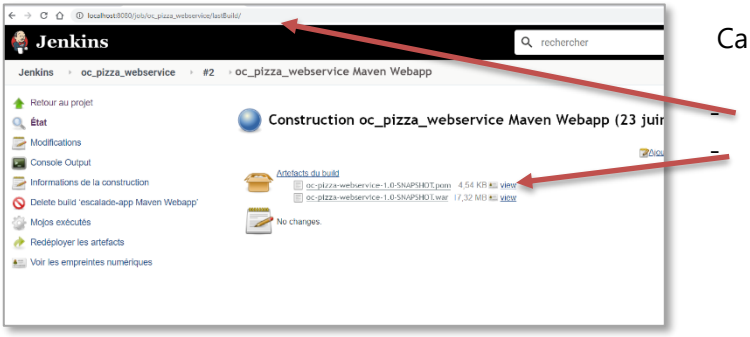


Capture d'un job réussi de l'application métier :

- Url pour accéder au job
- Artefacts correspondants.

http://localhost:8080/job/oc_pizza_front/

4.1.3 Job du webservice



Capture d'un job réussi de l'application métier :

- Url pour accéder au job
- Artefacts correspondants.

http://localhost:8080/job/oc_pizza_webservice/

4.2 Récupération du livrable depuis Nexus

4.2.1 Exécuter Nexus Manager

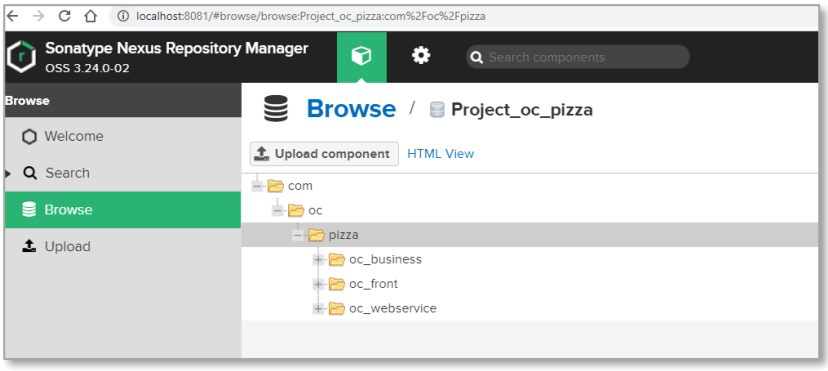
Tout d'abord lancer Nexus Manager depuis votre ordinateur :

- Se rendre dans le dossier : **C:_nexus-3.24.0-02\nexus-3.24.0-02\bin**
- Ouvrir un invité de commande
- Taper la commande **nexus.exe /run**

Une fois nexus.exe lancé, ouvrir votre navigateur et se rendre à l'adresse suivante, **http://localhost:8081/** pour afficher le tableau de bord de Nexus manager.

Il faudra se diriger vers un administrateur pour récupérer les identifiants et mot de passe pour se connecter à Nexus Manager.

4.2.2 Récupération des livrables



Les livrables sont composés :

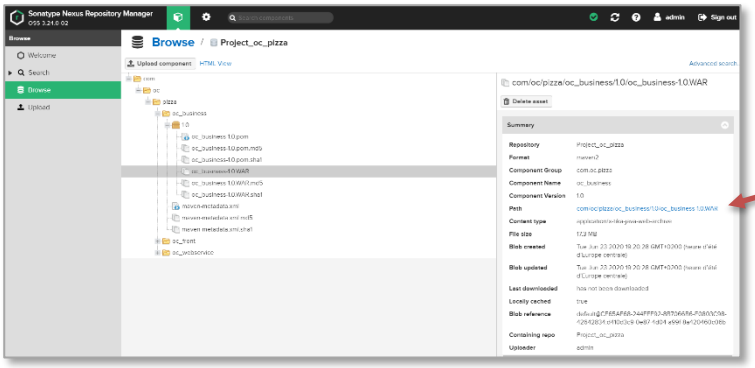
- d'une archive **oc_pizza_front.1.0.war** qui se trouve dans le dossier ***com/oc/pizza/oc_front***

- d'une archive **oc_pizza_business.1.0.war** qui se trouve dans le dossier **com/oc/pizza/oc_business**

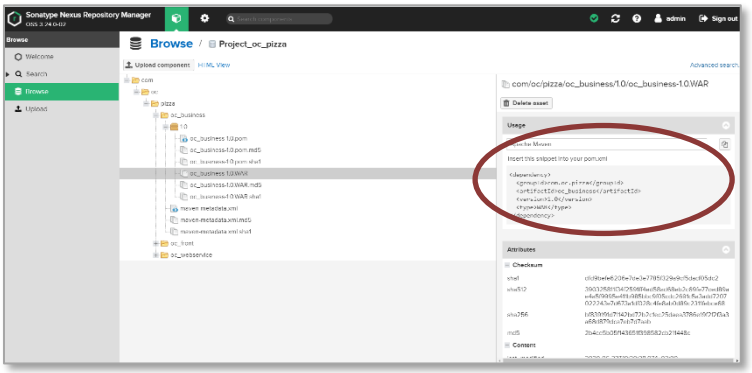
- d'une archive **oc_pizza_webservice1.0.war** qui se trouve dans le dossier **com/oc/pizza/oc_webservice**

Vous trouverez tous les livrables à l'adresse suivante :
http://localhost:8081/#browse/browse:Project_oc_pizza:com%2Foc%2Fpizza

4.2.2.1 Oc_pizza_business

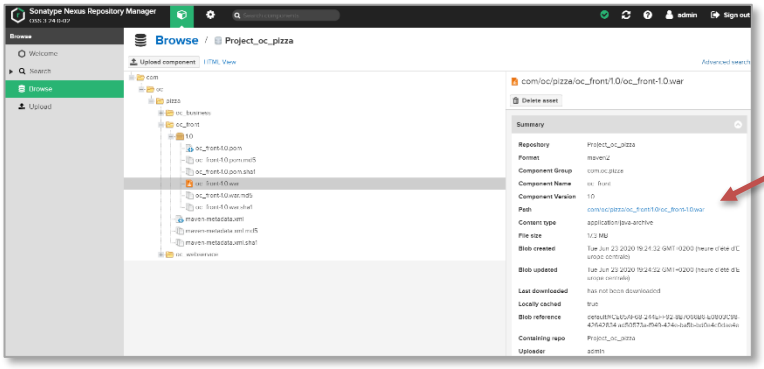


Cliquez sur le lien pour récupérer le war correspondant à l'application métier.

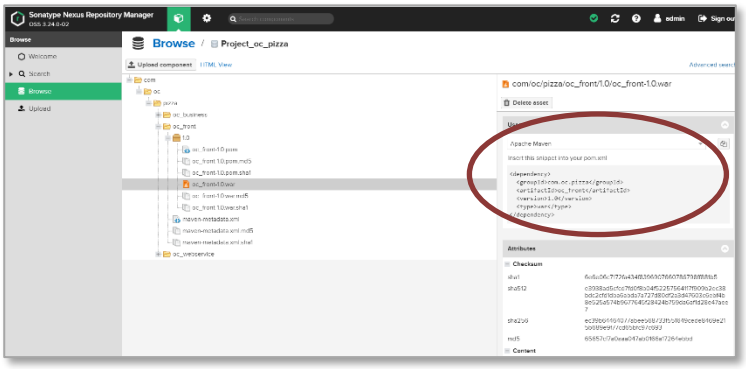


Il y a aussi la possibilité de récupérer la dépendance Maven de l'application métier.

4.2.2.2 Oc_pizza_front

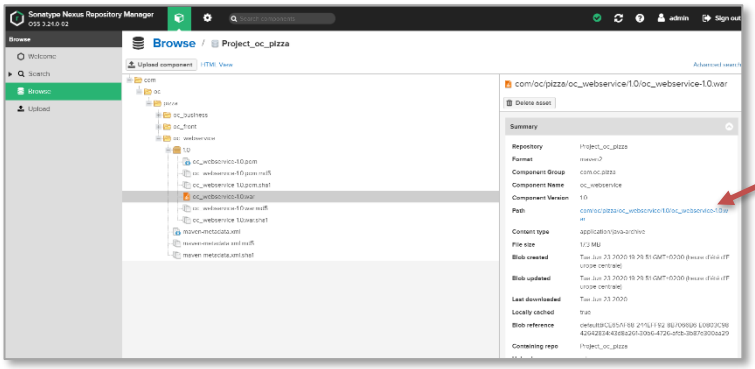


Cliquez sur le lien pour récupérer le war correspondant à l'application front.

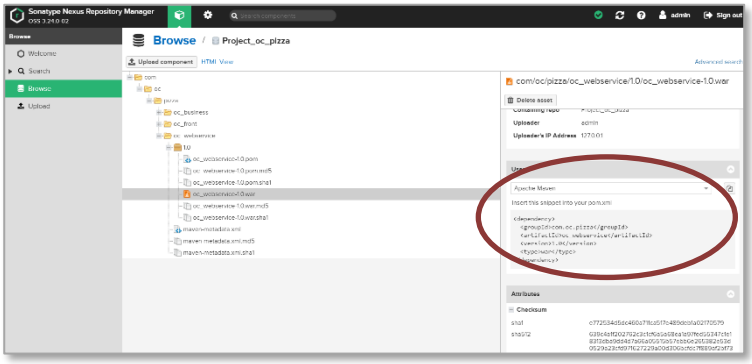


Dépendance Maven de l'application front.

4.2.2.3 *Oc_pizza_webservice*



Cliquez sur le lien pour récupérer le war correspondant au webservice.



Dépendance Maven du webservice.

4.3 Déploiement de l'Application Web

4.3.1 Artefacts

Le projet Oc_pizza se compose de 3 applications qui sont packagées sous format WAR. Vous trouverez ces artefacts dans le dépôt Nexus comme indiqué un peu plus haut.

4.3.2 Déploiement

Les 3 fichiers WAR, oc_pizza_front-1.0.war, oc_pizza_business-1.0.war et oc_pizza_webservice-1.0.war doivent être déployés sur le serveur Tomcat préalablement installé sur le serveur physique.

4.3.3 Environnement de l'application web

4.3.3.1 Variables d'environnement

Le serveur d'application JOnAS doit être exécuté avec la variable d'environnement suivante définie au démarrage. Elle est nécessaire afin de récupérer le répertoire contenant les fichiers de configuration de l'application :

-Dcom.ocpizza.apps.conf=$home_application_conf_directory

INFO : il ne faut pas mettre de « / » à la fin de la valeur de la variable et ne pas utiliser d'espace dans le chemin.

4.3.4 Répertoire de configuration applicatif

Le répertoire de configuration applicatif doit être créé sur le système de fichier et définit de la façon suivante :

$home_application_conf_directory/applicationX

… fichiers de configuration… :

- …

4.3.4.1 Fichier xxx.yyy

…

4.3.5 DataSources

Les accès aux bases de données doivent se configurer à l'aide des fichiers…

Le fichier de drivers **postgresql (postgresql-9.2.x.)** doit être déposé dans le répertoire :

$home_server/lib/ext

…

4.3.6 Ressources

…

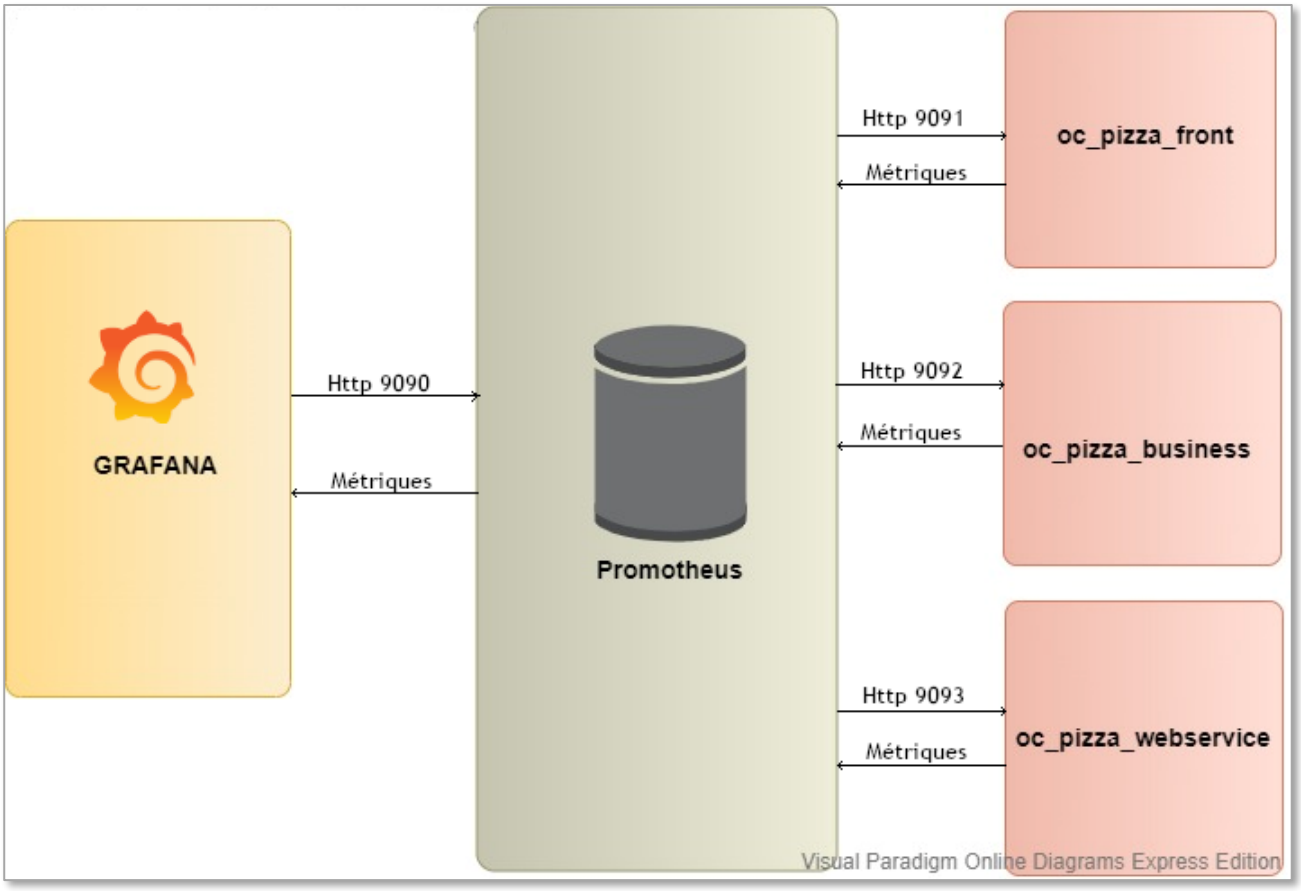
4.3.7 Vérifications

Afin de vérifier le bon déploiement de l'application, faire ceci…

5 SUPERVISION/MONITORING

5.1 Supervision de l'application web

5.1.1 Monitoring système



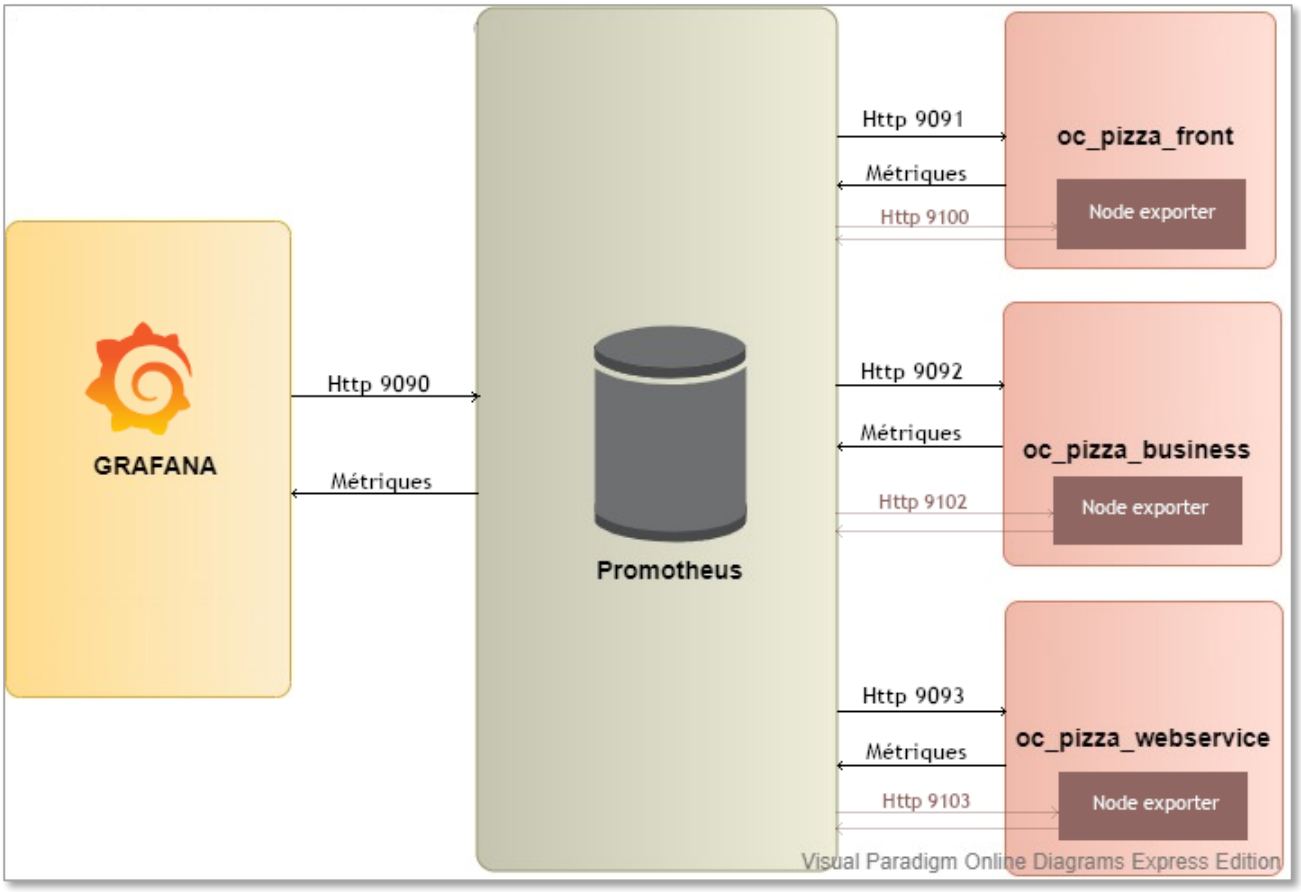
Afin de surveiller le bon fonctionnement de notre application et de prévenir une panne, un système de monitoring a été mis en place à l'aide des outils **GRAFANA** et **PROMOTHEUS**.

Côté applications, nous avons installé le client Spring **PROMETHEUS** qui permettra de remonter toutes les informations concernant les différentes instances, et de ce fait, surveiller l'état de notre application.

PROMETHEUS est installé sur un serveur, il va envoyer des requêtes http à intervalles réguliers sur nos différentes applications, pour récupérer des métriques et les stocker (Voir schéma ci-dessus).

On pourra exploiter ces métriques sur **GRAFANA** qui sera installé sur un autre serveur. **GRAFANA** présente un tableau de bord qui contiendra les métriques récupérées qu'il affichera sous forme de graphe.

Tous les ports à utiliser pour appeler les différents services sont indiqués sur le schéma.



Si les métriques fournis par le client ne suffisent pas, on peut implémenter nos propres collectors. Mais comme nous n'avons pas forcément accès au code de l'application (côté développeur), une solution serait d'utiliser les **exporters** de **PROMETHEUS**. Il s'agit d'un programme qui permet de récupérer des métriques systèmes d'une source et de les exposer au format **PROMETHEUS**.

Une fois tout cela configuré, il va falloir mettre en place un tableau de bord pour les métriques systèmes qu'on a récupérés. On peut récupérer un tableau de bord **Node Exporter** avec des graphes déjà configurés et l'importer sur notre outil.

https://www.youtube.com/watch?v=gbg-iPvn3C4

5.1.2 Monitoring applicatif

Il s'agit là, de consulter des logs fonctionnels, c'est-à-dire les logs qui concernent l'application et que l'on retrouve dans les fichiers logs.

Afin de récupérer et surveiller les informations remontées par les **Loggers**, nous allons utiliser des outils de gestion de logs. Il s'agit de la suite ELK.

Cette suite se compose :

- D'un tableau de bord, **KIBANA**, qui permettra d'afficher les logs,
- D'un serveur de base de données de logs, **Logstash**,
- De **Elasticsearch**, outil qui permettra de faire les requêtes depuis le tableau de bord vers la base de données de logs **Logstach**.

Pour envoyer les logs fonctionnels vers le serveur **Logstash**, il va falloir utiliser un **Appender** spécifique à **Logstash** pour indiquer à l'application que c'est vers ce serveur que les logs devront être envoyés.

6 PROCEDURE DE DEMARRAGE / ARRET

6.1 Base de données

6.2 Application web

7 PROCEDURE DE MISE A JOUR

7.1 Base de données

8 PROCEDURE DE SAUVEGARDE ET RESTAURATION

En cas de panne de l'application, appliquer les consignes dans l'ordre qui suit :

- Redémarrage de l'application
- Faire une recovery sur dernière sauvegarde de la base de données.