|  |
| --- |
| **OC PIZZA**  **Documentez votre système de gestion de pizzeria**  Dossier de conception technique  Version 1.0 |
| **Auteur**  AUBRUN Eric  *Analalyste-programmeur* |

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 - Application Web 5

3.1.1 - Composant Authentification 6

3.1.2 - Composant Gestion du stock 7

3.1.3 - Composant Cycle de vie de commande 7

3.1.4 - Les API (Application Programming Interface) 8

3.1.4.1 - L’API Géolocalisation 8

3.1.4.2 - L’API SMS 8

3.1.4.3 - L’API Paiement en ligne 8

3.1.4.4 - L’API Mail 8

4 - Architecture de Déploiement 9

4.1 - Serveur de Base de données 9

4.2 - Serveur XXX 9

5 - Architecture logicielle 10

5.1 - Principes généraux 10

5.1.1 - Les couches 10

5.1.2 - Les modules 10

5.1.3 - Structure des sources 10

5.2 - Application Web 11

5.3 - Application Xxx 11

6 - Points particuliers 12

6.1 - Gestion des logs 12

6.2 - Fichiers de configuration 12

6.2.1 - Application web 12

6.2.1.1 - Datasources 12

6.2.1.2 - Fichier xxx.yyy 12

6.2.2 - Application Xxx 12

6.3 - Ressources 12

6.4 - Environnement de développement 12

6.5 - Procédure de packaging / livraison 12

6.6 - XXX 12

7 - Glossaire 14

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| Eric AUBRUN | 01/05/2021 | Création du document | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier de conception technique de l'application OC PIZZA. Il est destiné à l’attention des développeurs, mainteneurs et de l’équipe technique d’OC PIZZA.

L’objectif de l’analyse technique étant de lister les contraintes spécifiques dont les développeurs vont devoir tenir compte pour coder l’application, le présent document présentera les langages et les conventions de développement, les conventions de développement, l’architecture logicielle et de déploiement ainsi que les logs, le monitoring propres à l’application.

Les éléments du présent dossier découlent des documents suivants :

* Mise en place d’un nouveau système informatique pour l’ensemble des pizzerias du groupe OC Pizza
* Concevez la solution technique d’un système de gestion de pizzeria

## Références

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

1. **PDOCPizza\_02\_fonctionnelle.pdf – 1.0**: Le Dossier de conception technique de l'application
2. **PDOCPizza\_03\_exploitation.pdf – 1.0** : Le Dossier d'exploitation de l’application
3. **PDOCPizza\_04\_livraison.pdf – 1.0 :** Le PV de livraison finale

# Architecture Technique

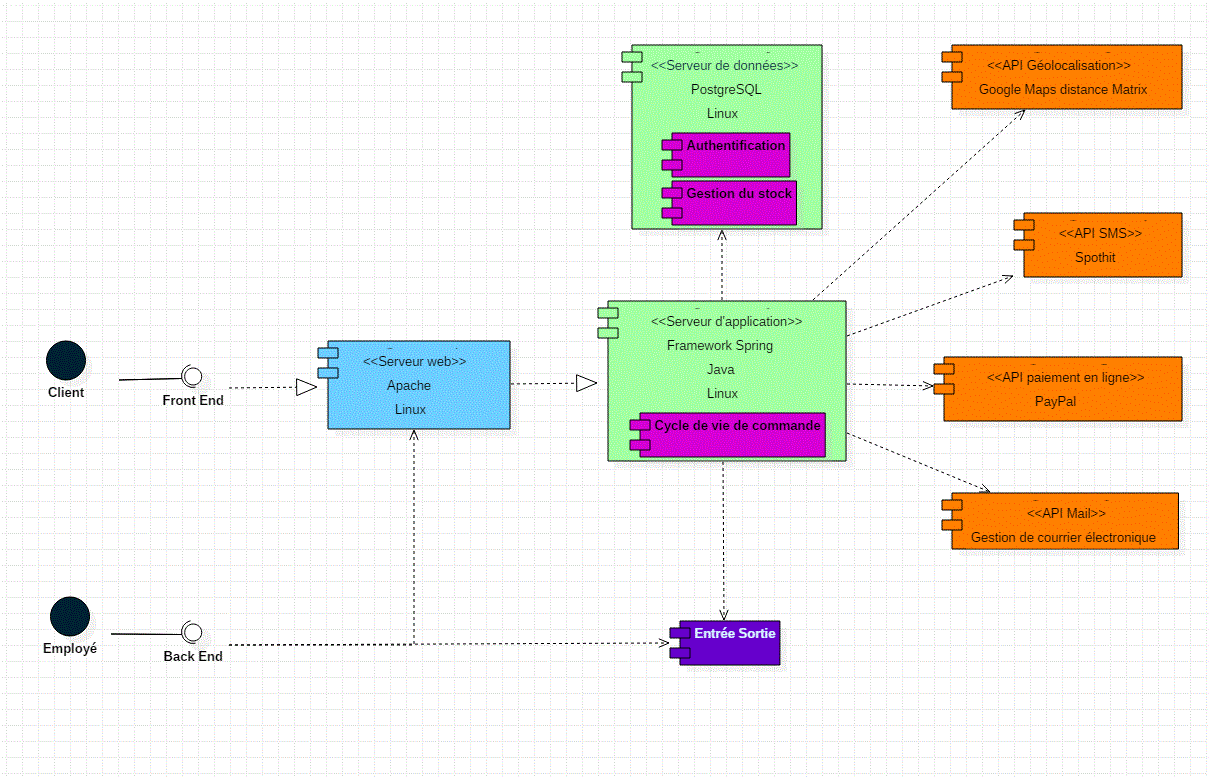
Nous proposons d’architecturer le produit en deux applications :

* Une application et son serveur traitant spécifiquement le front-end :« *Nous faisons le choix de développer le front-end en ayant recours au Framework Angular[[1]](#footnote-1). Cet outil sera accompagné et complété de Framework CSS, de librairies de composants Angular ainsi que de librairies JavaScript facilitant le développement en général.* »[[2]](#footnote-2)
* Une seconde application installée sur un autre serveur, Angular pilotant la communication entre les deux applications à partir de requêtes HTTP : « *Nous trouverons, côté back-end, une base de données contenant les informations relatives aux utilisateurs et aux produits proposés à la vente par OC Pizza. Nous utiliserons une API d’authentification afin de sécuriser les informations d’accès. Une API/web services permettra de gérer les paiements bancaires et enfin, dans le but d’améliorer les performances de l’application, nous pourrons exécuter côté serveur, Angular Universal.*»[[3]](#footnote-3)

## Application Web

La pile logicielle est la suivante :

* Pour la partie back-end :
* PostgreSQL installé sur un serveur de données sous Linux, gérant l’authentification ainsi que le stock de produits et ingrédients, nécessaires à la confection des pizzas.
* Le Framework Spring installé sur un serveur d’application tournant sous Linux, développé en Java, gérant plus particulièrement les différentes phases du cycle de vie d’une commande.
* Pour la partie front-end :
* Le Framework Angular installé sur un serveur Web Apache, lui aussi sous Linux, accompagné d’un Framework CSS comme Bootstrap, permettant notamment d’obtenir un effet « Responsive Web Design »[[4]](#footnote-4), des librairies de composants Angular comme Ng-Bootstrap, en mesure par exemple de construire des fenêtres modales, des librairies générant des graphiques de type « camemberts » comme Chart.Js ou encore Lodash, une bibliothèque Javascript contenant des fonctions permettant par exemple de tester et manipuler des valeurs, créer des fonctions composites ou utiles pour des tableaux, objets ou chaînes de texte.



**DIAGRAMME 1 : DIAGRAMME DE COMPOSANTS**

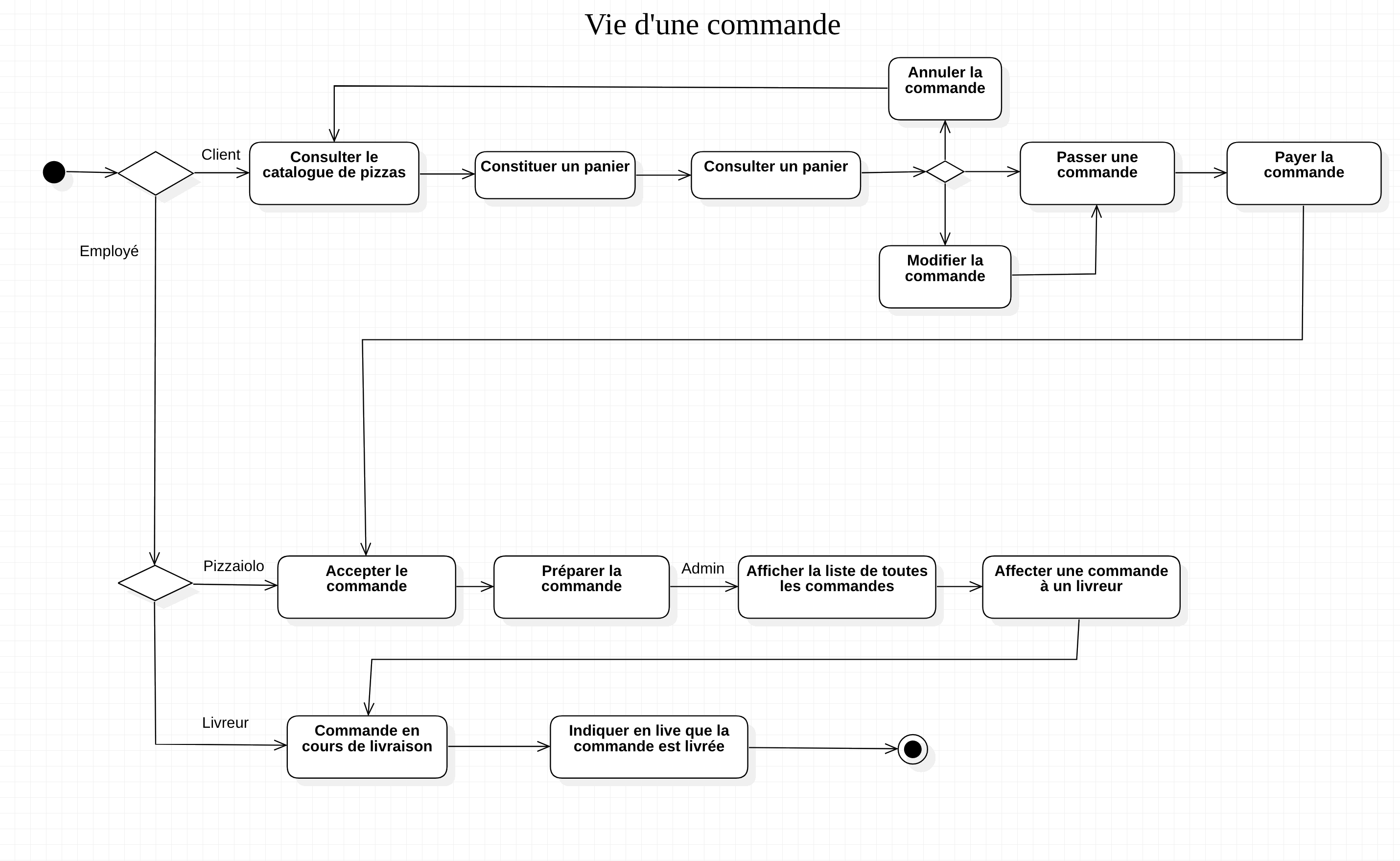
### Composant Authentification

Ce composant permet de sécuriser l’accès à l’application. Le client comme l’employé a accès à certaines fonctionnalités, en proportion des droits qui lui sont en amont attribués. On pourrait élever le niveau de sécurité en couplant la base de données avec Spring Security ce qui permettrait d’atténuer les risques de failles de type Cross-Site Scripting (XSS)[[5]](#footnote-5) et Cross Site Request Forgery (XSRF en français)[[6]](#footnote-6)

### Composant Gestion du stock

La gestion du stock est un élément clé dans l’offre commerciale d’OC Pizza. On pourrait éventuellement proposer en supplément une application de type Batch qui permettrait de récupérer la liste des ingrédients et des produits dont le stock propre se situerait en dessous d’un plancher prédéfini. Cette information permettrait à l’administrateur d’effectuer une commande auprès de ses fournisseurs en proportion des stocks d’ingrédients et produits insuffisants pour la poursuite de l’activité de l’entreprise.

### Composant Cycle de vie de commande

Pour rappel :

**DIAGRAMME 1 : DIAGRAMME D’ACTIVITÉ DE LA VIE D’UNE COMMANDE CHEZ OC PIZZA[[7]](#footnote-7)**

Ce composant devra être doté des 4 opérations de base sur les données persistantes en base de données (Create (créer) - Read (lire) - Update (mettre à jour) - Delete (supprimer)). Ce pattern permettra au client de LIRE/Consulter le catalogue de pizzas, de CRÉER/Passer une commande, de METTRE À JOUR/Modifier cette commande ou de la SUPPRIMER.

### Les API (Application Programming Interface)

« *Une API, ou* [*interface de programmation d'application*](https://www.redhat.com/fr/topics/api)*, est un ensemble de définitions et de protocoles qui facilite la création et l'intégration de logiciels d'applications. Elle est parfois considérée comme un contrat entre un fournisseur d'informations et un utilisateur d'informations, qui permet de définir le contenu demandé au consommateur (l'appel) et le contenu demandé au producteur (la réponse). Par exemple, l'API d'un service de météo peut demander à l'utilisateur de fournir un code postal et au producteur de renvoyer une réponse en deux parties : la première concernant la température maximale et la seconde la température minimale. »* Source : <https://www.redhat.com/fr/topics/api/what-is-a-rest-api>.

#### L’API Géolocalisation

Cette API est construite de telle sorte que lorsque l’utilisateur effectue une recherche de points de vente d’OC Pizza, le résultat affiche ceux en proximité de l’adresse de livraison du client.

#### L’API SMS

Cette API est accessible uniquement pour l’administrateur. Elle va lui permettre d’affecter une livraison à un livreur dès lors qu’une commande est prête.

#### L’API Paiement en ligne

Cette API va principalement vérifier la solvabilité du client afin que le paiement puisse être entièrement effectué.

#### L’API Mail

L’API Mail génère des mails pour notifier le client du suivi de sa commande. Le premier mail confirme la validation du paiement, le second informe que la commande est en préparation et donc ne peut plus être ni modifiée ni annulée et le troisième mail confirme la réception de la commande.

# 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 crées 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

…

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

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

1. Source : https://www.angularchef.com/recette/2/ [↑](#footnote-ref-1)
2. Extrait de : « Mise en place d’un nouveau système informatique pour l’ensemble des pizzerias du groupe OC Pizza », page 39 [↑](#footnote-ref-2)
3. Ibid. page 39 [↑](#footnote-ref-3)
4. Le Responsive Design ou plus précisément le Responsive Web Design (RWD) est une technique de conception d’interface digitale qui fait en sorte que l’affichage d’une quelconque page d’un site s’adapte de façon automatique à la taille de l’écran du terminal qui le lit. Source : <https://www.seo.fr/definition/responsive-design#:~:text=Le%20Responsive%20Design%20ou%20plus,du%20terminal%20qui%20le%20lit>. [↑](#footnote-ref-4)
5. Le cross-site scripting (abrégé en XSS) est un type de faille de sécurité de sites Internet. Des scripts malveillants sont introduits (le terme « injecté » est habituellement utilisé) dans des sites Web, afin de pouvoir attaquer les systèmes des utilisateurs. Source : <https://www.ionos.fr/digitalguide/sites-internet/developpement-web/quest-ce-que-le-xss-cross-site-scripting/> [↑](#footnote-ref-5)
6. CSRF : le Cross Site Request Forgery (XSRF en français) est un mode d’escroquerie courant sur Internet. Les criminels prennent le contrôle d'une session autorisée par l’utilisateur (Session Riding) et peuvent ainsi exécuter des actions malveillantes. Celles-ci passent par le biais de requêtes HTTP. Source : <https://www.ionos.fr/digitalguide/serveur/securite/cross-site-request-forgery/> [↑](#footnote-ref-6)
7. Extrait de : « Mise en place d’un nouveau système informatique pour l’ensemble des pizzerias du groupe OC Pizza », page 5 [↑](#footnote-ref-7)