

# OC Pizza

## Système de gestion de pizzerias

Dossier de conception technique

Version 1.0

**Auteur :**  
Maxime Point  
*Développeur2*

# TABLE DES MATIERES

<b>1 - Versions .....</b>	<b>3</b>
<b>2 - Introduction .....</b>	<b>4</b>
2.1 - Objet du document .....	4
2.2 - Références.....	4
<b>3 - Architecture Technique .....</b>	<b>5</b>
3.1 - Composants généraux.....	5
3.2 - Diagramme de classe.....	6
3.2.1 - La classe User .....	7
3.2.2 - La classe Order .....	8
3.2.3 - La classe Payment.....	9
3.2.4 - La classe Pizza .....	10
3.2.5 - La classe Ingrédient .....	11
3.2.6 - La classe Pizzeria .....	12
3.2.7 - La classe Address .....	13
3.2.8 - Autres classes et énumérations. ....	14
3.3 - Modèle physique de données .....	16
3.3.1 - La table User.....	17
3.3.2 - La table Order .....	18
3.3.3 - La table Payment .....	19
3.3.4 - La table d'association Order/Pizza.....	20
3.3.5 - La table Pizza.....	21
3.3.6 - La table d'association Pizza/Ingredient .....	22
3.3.7 - La table ingrédient.....	23
3.3.8 - La table d'association Pizzeria/Ingredient.....	24
3.3.9 - La table Pizzeria .....	25
3.3.10 - La table Address.....	26
3.4 - Script de la base de données mySQL.....	27
3.5 - Jeu de données .....	32
<b>4 - Architecture de déploiement .....</b>	<b>34</b>
4.1 - Hébergeur .....	34
4.2 - Serveur des bases de données.....	35
4.3 - Services Web .....	35
<b>5 - Architecture logicielle .....</b>	<b>36</b>
5.1 - Principes généraux.....	36
5.2 - Structure .....	36
<b>6 - Points particuliers .....</b>	<b>37</b>
6.1 - Gestions des logs .....	37
6.2 - Fichiers de configuration .....	37
6.3 - Procédure de packaging / livraison .....	37

# 1 - VERSIONS

Auteur	Date	Description	Version
M.Point	07/10/2022	Création du document. Établi par rapport au dossier des spécifications fonctionnelles et aux échanges avec l'équipe d'IT.	1.0

## 2 - INTRODUCTION

### 2.1 - Objet du document

Le présent document constitue le dossier de conception technique du système informatique de gestion des pizzerias du groupe OC Pizza.

L'objectif du document est d'aider les développeurs pour la conception et la maintenance du système informatique. Vous trouverez dans ce dossier, le diagramme de classe, le modèle physique de données ainsi que le script SQL pour la réalisation de la base de données.

Les éléments du présent dossier découlent :

- Du dossier de conception fonctionnelle,
- Des différents échanges avec l'équipe d'IT Consulting & Development.

### 2.2 - Références

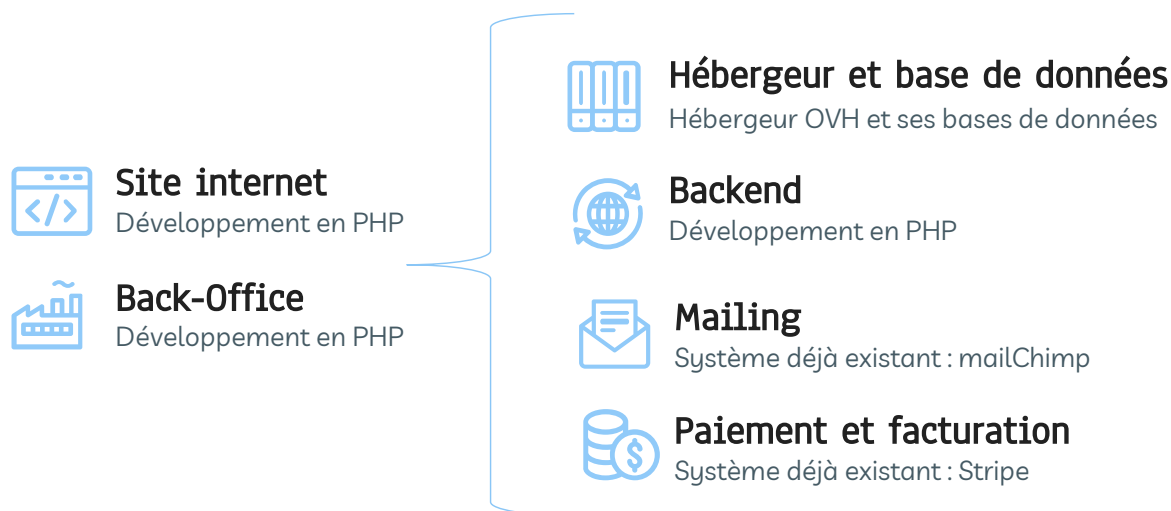
Pour de plus amples informations, se référer également aux éléments suivants :

1. **DCF** : Dossier de conception fonctionnelle de l'application.
2. **DE** : Dossier d'exploitation

## 3 - ARCHITECTURE TECHNIQUE

### 3.1 - Composants généraux

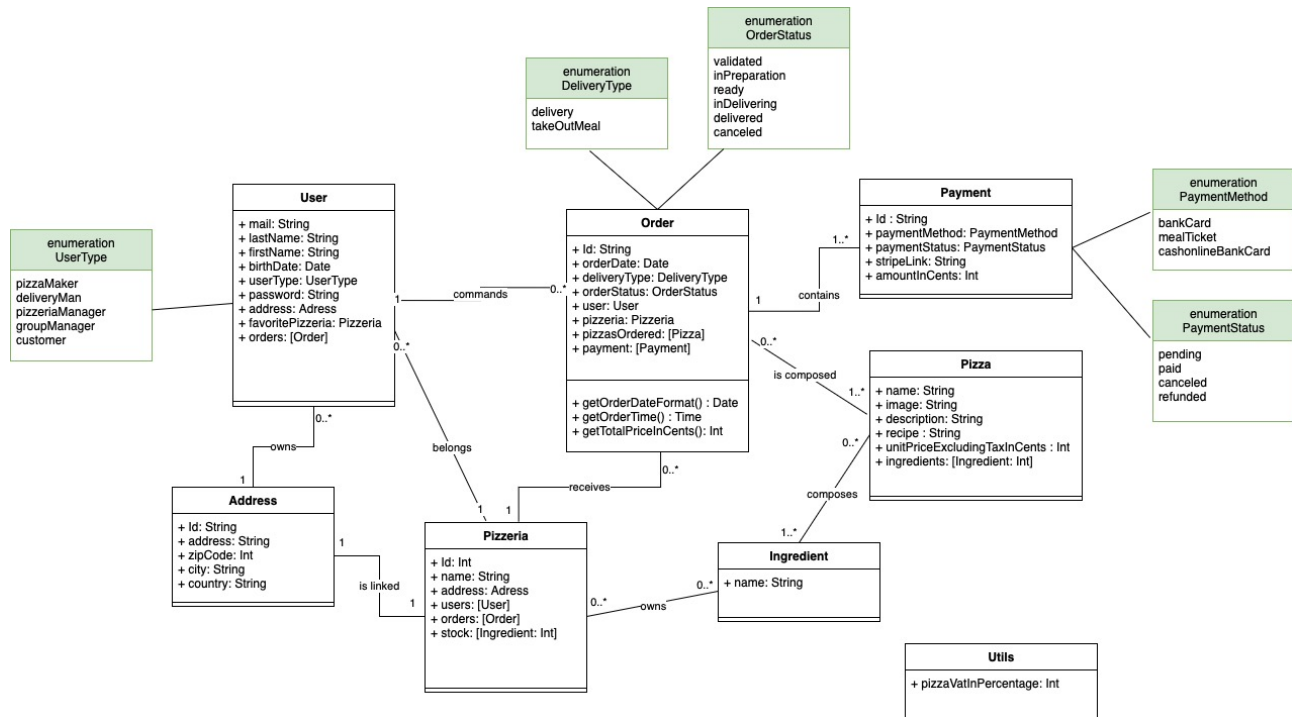
Suite aux différents échanges que nous avons pu avoir avec OC Pizza, ainsi qu'à la conception du dossier de spécifications fonctionnelles, vous pouvez retrouver ci-dessous les composants généraux qui seront utilisés pour la réalisation de ce système informatique.



- Un site internet de type webApp accessible au client, sur smartphone, tablette ou ordinateur. Ce site sera développé par notre équipe en PHP.
- Un back-office de type webApp également accessible aux employés, sur smartphone, tablette ou ordinateur. Ce back-office sera également développé par notre équipe en PHP.
- Un hébergeur : Nous utiliserons l'hébergeur OVH.
- Une base de donnée : Nous créerons une base de données MySQL sur l'hébergeur OVH.
- Un back-end : cela va faire le lien entre la base de données, le site internet et le backoffice. Ce back-end sera également développé par notre équipe en PHP.
- Un système de paiement et de facturation. Pour une question de compatibilité, de sécurité, et afin de réduire le coût et le temps de développement, nous avons opté pour utiliser un système déjà existant que nous aurons plus qu'à implementer sur notre système. Nous utiliserons le système « Stripe », qui est un système reconnu mondialement.
- Un système de mailing automatique. Ici également, dans le but de réduire le coût et le temps de développement, nous utiliserons un système déjà existant qui est le système « MailChimp », système également reconnu mondialement.

## 3.2 - Diagramme de classe

Vous trouverez ci-dessous un diagramme de classe UML afin de visualiser les classes qui se trouveront dans le code du système informatique. Ce diagramme de classe a été construit à partir de notre modèle de domaine et plus particulièrement à partir des descriptions des cas d'utilisation (use cases) se trouvant dans le dossier de conception fonctionnelle.



Vous trouverez dans les parties suivantes, le descriptif de chaque classe.

### 3.2.1 - La classe User

Cette classe regroupe les attributs d'un « User » qui correspond à un utilisateur du système informatique d'OC Pizza. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

USER	Attribut	Type	Description
	mail	String	Le mail de l'utilisateur
	lastName	String	Le nom de l'utilisateur
	firstName	String	Le prénom de l'utilisateur
	birthDate	Date	La date de naissance de l'utilisateur
	userType	UserType	Le type d'utilisateur (enumération comprenant : pizzaMaker, deliveryMan, pizzeriaManager, groupManager, customer)
	password	String	Le mot de passe de l'utilisateur
	address	Address	L'adresse de l'utilisateur
	favoritePizzeria	Pizzeria	La pizzeria favorite de l'utilisateur
	orders	[Order]	Toutes les commandes de l'utilisateur sous forme de tableau

Associations de la classe :

- 1 à plusieurs, avec la classe « Order ». Un utilisateur pourra effectivement réaliser 0 ou plusieurs commandes, mais une commande sera forcément attribuée à un seul utilisateur.
- 1 à plusieurs, avec la classe « Address ». Un utilisateur aura une seule adresse, mais une adresse pourra résider 0 ou plusieurs utilisateurs.
- 1 à plusieurs, avec la classe « Pizzeria ». Un utilisateur devra avoir une pizzeria favorite, mais une pizzeria pourra être la pizzeria favorite de plusieurs utilisateurs.

### 3.2.2 - La classe Order

Cette classe regroupe les attributs d'une « Order » qui correspond à une commande d'un client. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Order	Attribut	Type	Description
	id	String	L'identifiant de la commande
	orderDate	Date	La date de la commande
	deliveryType	DeliveryType	Le type de livraison (enumération comprenant : delivery, takeOutMeal)
	orderStatus	OrderStatus	Le statut de la commande (validated, inPreparation, ready, inDelivering, delivered, canceled)
	user	User	L'utilisateur qui a passé la commande
	pizzeria	Pizzeria	La pizzeria où a été effectuée la commande
	pizzasOrdered	[Pizza]	Le contenu de la commande sous forme de tableau de pizza
	payment	[Payment]	Les paiements de la commande sous forme de tableau de paiement

Associations de la classe :

- 1 à plusieurs, avec la classe « User ». Un utilisateur pourra effectivement réaliser 0 ou plusieurs commandes, mais une commande sera forcément attribuée à un seul utilisateur.
- 1 à plusieurs, avec la classe « Payment ». Une commande aura un ou plusieurs paiements, mais un paiement sera forcément attribué à une seule commande.
- plusieurs à plusieurs, avec la classe « Pizza ». Une commande pourra contenir plusieurs pizzas, et une pizza pourra être présente dans 0 ou plusieurs commandes.
- 1 à plusieurs, avec la classe « Pizzeria ». Une commande pourra être liée à une seule pizzeria, mais une pizzeria pourra recevoir 0 ou plusieurs commandes.



### 3.2.3 - La classe *Payment*

Cette classe regroupe les attributs d'un « Payment » qui correspond à un paiement d'une commande. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Payment	Attribut	Type	Description
	id	String	L'identifiant du paiement
	paymentMethod	PaymentMethod	Le type de paiement (enumération comprenant : bankCard, mealTicket, cashonlineBankCard)
	paymentStatus	PaymentStatus	Le statut du paiement (enumération comprenant : pending, paid, canceled, refunded)
	stripeLink	String	Le lien de la plateforme Stripe correspondant au paiement
	amountInCents	Int	Le montant de la commande en centimes

Associations de la classe :

- 1 à plusieurs, avec la classe « Order ». Une commande aura un ou plusieurs paiements, mais un paiement sera forcément attribué à une seule commande.

### 3.2.4 - La classe Pizza

Cette classe regroupe les attributs d'une « Pizza » qui correspond à une pizza disponible chez OC Pizza. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Pizza	Attribut	Type	Description
	name	String	Le nom de la pizza
	image	String	Le lien de la photo de la pizza
	description	String	La description de la pizza
	recipe	String	La recette de la pizza
	unitPriceExcludingTaxInCents	Int	Le montant de la pizza en centimes
	ingredients	[Ingredient: Int]	Les ingrédients avec leur quantité qui composent cette pizza

Associations de la classe :

- plusieurs à plusieurs, avec la classe « Order ». Une commande pourra contenir plusieurs pizzas, et une pizza pourra être présente dans 0 ou plusieurs commandes.
- plusieurs à plusieurs, avec la classe « Ingredient ». Une pizza pourra contenir plusieurs aliments, et un aliment pourra être présent dans 0 ou plusieurs pizzas.

### 3.2.5 - La classe *Ingrédient*

Cette classe regroupe les attributs d'un « Ingrédient » qui correspond à un ingrédient pouvant être utilisé dans les pizzas d'OC Pizza. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Ingrédients	Attribut	Type	Description
	name	String	Le nom de l'ingrédient

Associations de la classe :

- plusieurs à plusieurs, avec la classe « Pizza ». Un ingrédient pourra être présent dans 0 ou plusieurs pizzas, et une pizza pourra être composée avec 1 ou plusieurs ingrédients.
- plusieurs à plusieurs, avec la classe « Pizzeria ». Une pizzeria pourra avoir dans son stock plusieurs aliments, et un aliment pourra être présent dans 0 ou plusieurs stocks de pizzeria.

### 3.2.6 - La classe Pizzeria

Cette classe regroupe les attributs d'une « Pizzeria » qui correspond à une pizzeria du groupe OC Pizza. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Pizzeria	Attribut	Type	Description
	id	Int	L'identifiant de la pizzeria
	name	String	Le nom de la pizzeria
	address	Adress	L'adresse de la pizzeria
	users	[User]	Les utilisateurs ayant cette pizzeria en favorite, sous forme de tableau de User
	orders	[Order]	Les commandes de cette pizzeria, sous forme de tableau de Order
	stock	[Ingrédient : Int]	Le stock d'ingrédients de la pizzeria, sous forme de dictionnaire avec comme clés l'ingrédient, et en valeur la quantité.

Associations de la classe :

- 1 à plusieurs, avec la classe « Pizzeria ». Un utilisateur devra avoir une pizzeria favorite, mais une pizzeria pourra être la pizzeria favorite de plusieurs utilisateurs.
- 1 à plusieurs, avec la classe « Pizzeria ». Une commande pourra être liée à une seule pizzeria, mais une pizzeria pourra recevoir 0 ou plusieurs commandes.
- plusieurs à plusieurs, avec la classe « Pizzeria ». Une pizzeria pourra avoir dans son stock plusieurs aliments, et un aliment pourra être présent dans 0 ou plusieurs stocks de pizzeria.
- un à un : une pizzeria pourra avoir une seule adresse, et une adresse pourra être liée qu'à une seule pizzeria.

### 3.2.7 - La classe Address

Cette classe regroupe les attributs d'un « Payment » qui correspond à un paiement d'une commande. Vous trouverez ci-dessous la liste des attributs présents dans cette classe.

Address	Attribut	Type	Description
	id	String	L'identifiant de l'adresse
	address	String	L'adresse, c'est-à-dire le numéro et le nom de la rue.
	zipCode	Int	Le code postale de l'adresse
	city	String	La ville de l'adresse
	country	String	Le pays de l'adresse

Associations de la classe :

- un à un : une pizzeria pourra avoir une seule adresse, et une adresse pourra être liée qu'à une seule pizzeria.
- 1 à plusieurs, avec la classe « Address ». Un utilisateur aura une seule adresse, mais une adresse pourra résider 0 ou plusieurs utilisateurs.

### 3.2.8 - Autres classes et énumérations.

D'autres classes, comme la class Utils, ainsi que des énumérations sont présents dans notre diagramme de classes . Vous trouverez ci-dessous la liste de ces classes et de ces énumérations, avec leurs attributs.

	Attribut	Type	Description
Utils	pizzaVatinPercentage	Int	Taxe en % à appliquer pour chaque pizza en fonction du pays.

	Attribut	Description
Enumeration UserType	pizzaMaker	Un utilisateur de type pizzaiolo
	deliveryMan	Un utilisateur de type livreur
	pizzeriaManager	Un utilisateur de type manager d'une pizzeria d'OC Pizza
	groupManager	Un utilisateur de type manager du groupe OC Pizza
	customer	Un utilisateur de type client

	Attribut	Description
Enumeration DeliveryType	delivery	Une livraison de type « livraison »
	takeOutMeal	Une livraison de type « à emporter »

	Attribut	Description
Enumeration OrderStatus	validated	Une commande avec le statut validée
	inPreparation	Une commande avec le statut en préparation
	ready	Une commande avec le statut prête
	inDelivering	Une commande avec le statut en livraison
	delivered	Une commande avec le statut délivrée
	canceled	Une commande avec le statut annulée






Enumeration PaymentMethod	Attribut	Description
	bankCard	Une méthode de paiement de type Carte Bancaire
	mealTicket	Une méthode de paiement de type ticket restaurant
	cashonlineBankCard	Une méthode de paiement de type carte bancaire en ligne

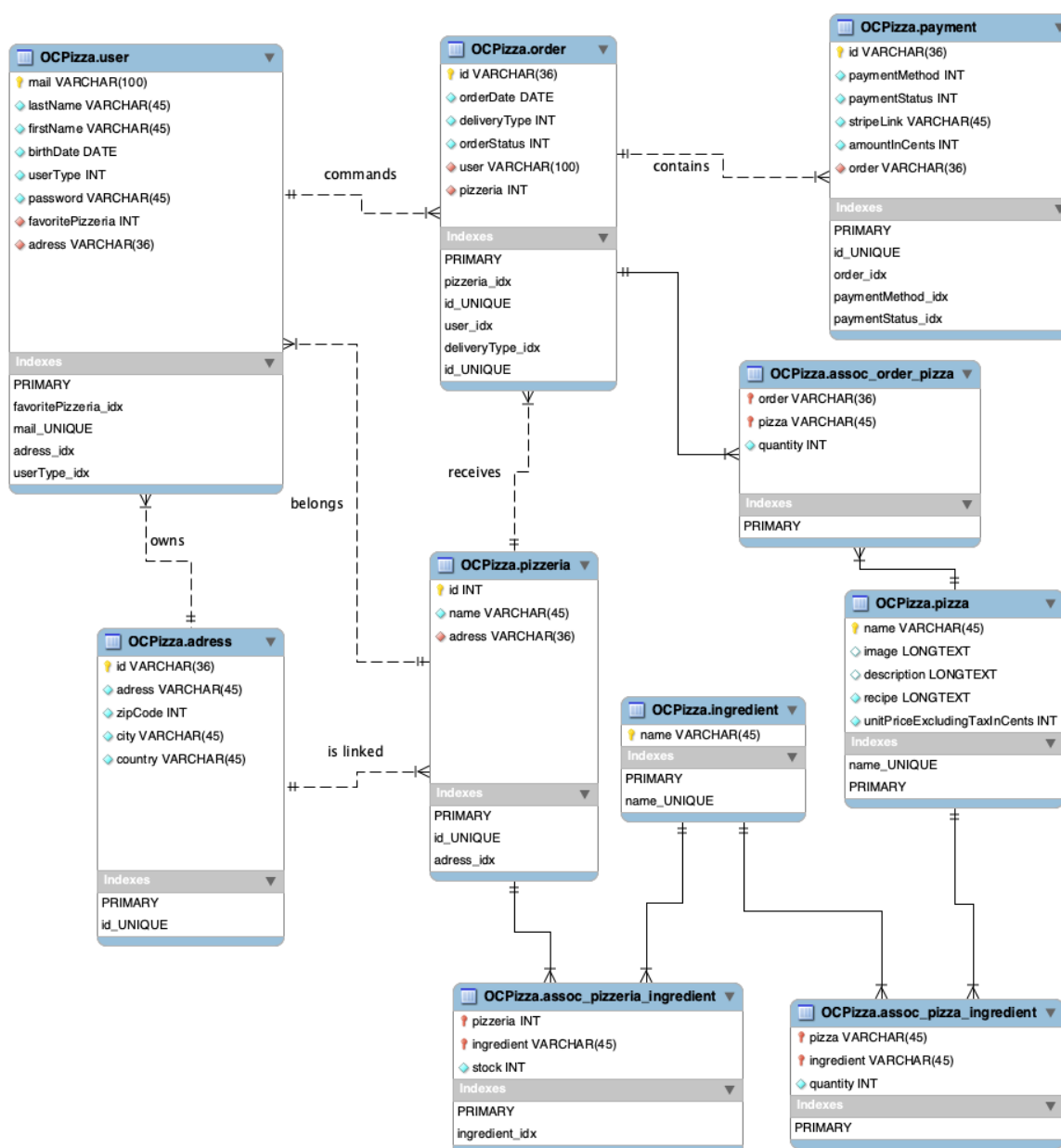
Enumeration PaymentStatus	Attribut	Description
	pending	Un paiement avec le statut en attente
	paid	Un paiement avec le statut payé
	canceled	Un paiement avec le statut annulé
	refunded	Un paiement avec le statut remboursé

### 3.3 - Modèle physique de données

Vous trouverez ci-dessous le diagramme illustrant le modèle physique de données. Celui-ci a été conçu à partir du diagramme de classe. Il permet d'avoir une représentation de la structure de la base de données du système informatique d'OC Pizza.

Pour faciliter la compréhension de ce diagramme, vous trouverez ci-dessous une légende.

-  Clé primaire + clé étrangère
-  Clé primaire
-  Attribut Not Null
-  Attribut
-  Clé étrangère Not Null



Vous trouverez dans les parties suivantes, le détail de chaque table de cette base de données.



### 3.3.1 - La table User

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « User ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

User	Attribut	Type	Caractéristiques	Description
	mail	VARCHAR(100)	Not Null, Clé primaire, Unique	Le mail de l'utilisateur
	lastName	VARCHAR(45)	Not Null	Le nom de l'utilisateur
	firstName	VARCHAR(45)	Not Null	Le prénom de l'utilisateur
	birthDate	Date	Not Null	La date de naissance de l'utilisateur
	userType	Int	Not Null	Le type d'utilisateur (de type Int, mais sera lié par la suite dans le code à une énumération)
	password	VARCHAR(45)	Not Null	Le mot de passe de l'utilisateur
	favoritePizzeria	Int	Not Null, clé étrangère	La pizzeria favorite de l'utilisateur
	address	VARCHAR(36)	Not Null, clé étrangère	L'adresse de l'utilisateur

### 3.3.2 - La table Order

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Order ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Order	Attribut	Type	Caractéristiques	Description
	id	VARCHAR(36)	Not Null, Clé primaire, Unique	L'identifiant de la commande
	orderDate	Date	Not Null	La date de la commande
	deliveryType	Int	Not Null	Le type de livraison (de type Int, mais sera lié par la suite dans le code à une énumération)
	orderStatus	Int	Not Null	Le statut de la commande (de type Int, mais sera lié par la suite dans le code à une énumération)
	user	VARCHAR(100)	Not Null, clé étrangère	L'utilisateur qui a passé la commande
	pizzeria	Int	Not Null, clé étrangère	La pizzeria où a été effectuée la commande

### 3.3.3 - La table Payment

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Payment ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Payment	Attribut	Type	Caractéristiques	Description
	id	VARCHAR(36)	Not Null, Clé primaire, Unique	L'identifiant du paiement
	paymentMethod	Int	Not Null	Le type de paiement (de type Int, mais sera lié par la suite dans le code à une énumération)
	paymentStatus	Int	Not Null	Le statut du paiement (de type Int, mais sera lié par la suite dans le code à une énumération)
	stripeLink	VARCHAR(45)	Not Null	Le lien de la plateforme Stripe correspondant au paiement
	amountInCents	Int	Not Null	Le montant de la commande en centimes
	order	VARCHAR(36)	Not Null, clé étrangère	La commande qui est reliée à ce paiement

### 3.3.4 - La table d'association Order/Pizza

Cette table Order/Pizza est une table d'association, car nous avons dans ce cas une cardinalité plusieurs-à-plusieurs entre nos deux objets Pizza et Order. Elle est forcément composée d'au moins 2 clés étrangères, référençant chacune l'un des 2 objets. . Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

association Order/Pizza	Attribut	Type	Caractéristiques	Description
	order	VARCHAR(36)	Not Null, clé étrangère	La commande associée à cet pizza
	pizza	VARCHAR(45)	Not Null, clé étrangère	La pizza associée à cette commande
	quantity	Int	Not Null	La quantité de cet pizza qu'il y a dans cette commande

### 3.3.5 - La table Pizza

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Pizza ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Pizza	Attribut	Type	Caractéristiques	Description
	name	VARCHAR(36)	Not Null, Clé primaire, Unique	Le nom de la pizza
	image	LONGTEXT		Le lien de la photo de la pizza
	description	LONGTEXT		La description de la pizza
	recipe	LONGTEXT	Not Null	La recette de la pizza
	unitPriceExcludingTaxInCents	Int	Not Null	Le montant de la pizza en centimes

### 3.3.6 - La table d'association Pizza/Ingredient

Cette table Pizza/Ingredient est une table d'association, car nous avons dans ce cas une cardinalité plusieurs-à-plusieurs entre nos deux objets Pizza et Ingredient. Elle est forcément composée d'au moins 2 clés étrangères, référençant chacune l'un des 2 objets. . Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

association Pizza/Ingredient	Attribut	Type	Caractéristiques	Description
	pizza	VARCHAR(45)	Not Null, clé étrangère	La pizza associée à cet ingredient
	ingredient	VARCHAR(45)	Not Null, clé étrangère	L'ingrédient associé à cette pizza
	quantity	Int	Not Null	La quantité de cet ingredient qu'il y a dans cette pizza

### 3.3.7 - La table *ingrédient*

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Ingredient ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Ingredient	Attribut	Type	Caractéristiques	Description
	name	VARCHAR(45)	Not Null, Clé primaire, Unique	Le nom d'ingrédient

### 3.3.8 - La table d'association Pizzeria/Ingredient

Cette table Pizzeria/Ingredient est une table d'association, car nous avons dans ce cas une cardinalité plusieurs-à-plusieurs entre nos deux objets Pizzeria et Ingredient. Elle est forcément composée d'au moins 2 clés étrangères, référençant chacune l'un des 2 objets. . Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

association Pizzeria/Ingredient	Attribut	Type	Caractéristiques	Description
	pizzeria	Int	Not Null, clé étrangère	La pizzeria ayant en stock cet ingredient
	ingredient	VARCHAR(45)	Not Null, clé étrangère	L'ingrédient stocké dans cette pizzeria
	stock	Int	Not Null	Le quantité de cet ingredient qu'il y a dans le stock de cette pizzeria



### 3.3.9 - La table Pizzeria

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Pizzeria ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Pizzeria	Attribut	Type	Caractéristiques	Description
	id	Int	Not Null, Clé primaire, Unique, Auto_Increment	L'identifiant de la pizzeria
	name	VARCHAR(45)	Not Null	Le nom de la pizzeria
	address	VARCHAR(36)	Not Null, clé étrangère	L'adresse de la pizzeria

### 3.3.10 - La table Address

Cette table possède des colonnes qui correspondent à la plupart des attributs de la classe « Address ». Vous trouverez ci-dessous la liste des colonnes présents dans cette table avec leurs caractéristiques.

Address	Attribut	Type	Caractéristiques	Description
	id	VARCHAR(36)	Not Null, Clé primaire, Unique	L'identifiant de l'adresse
	address	VARCHAR(45)	Not Null	L'adresse, c'est-à-dire le numéro et le nom de la rue.
	zipCode	Int	Not Null	Le code postale de l'adresse
	City	VARCHAR(45)	Not Null	La ville de l'adresse
	country	VARCHAR(45)	Not Null	Le pays de l'adresse

### 3.4 - Script de la base de données mySQL

Suite à la réalisation du modèle physique de données, nous avons pu generer un script mySQL permettant de créer la base de données vierge pour l'integrer à notre serveur. Vous pouvez retrouver ce script ci-dessous.

```
-- MySQL Script generated by MySQL Workbench
-- Sun Oct9 20:00:27 2022
-- Model: New ModelVersion: 1.0
-- MySQL Workbench Forward Engineering

--
-----

-- Schema OCPizza
--
-----

-- Création de la base de données
CREATE SCHEMA IF NOT EXISTS `OCPizza` DEFAULT CHARACTER SET utf8 ;
USE `OCPizza` ;

--
-----

-- Table `OCPizza`.`adress`
--
-----

CREATE TABLE IF NOT EXISTS `OCPizza`.`adress` (
  `id` VARCHAR(36) NOT NULL,
  `adress` VARCHAR(45) NOT NULL,
  `zipCode` INT NOT NULL,
  `city` VARCHAR(45) NOT NULL,
  `country` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE3` (`id` ASC))
ENGINE = InnoDB;

--
-----

-- Table `OCPizza`.`pizzeria`
--
-----

CREATE TABLE IF NOT EXISTS `OCPizza`.`pizzeria` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `adress` VARCHAR(36) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE4` (`id` ASC),
  INDEX `adress_idx` (`adress` ASC),
  CONSTRAINT `adress1`
    FOREIGN KEY (`adress`)
    REFERENCES `OCPizza`.`adress` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

```
ENGINE = InnoDB;
```

```
-- -----  
-- Table `OCPizza`.`user`  
-- -----
```

```
CREATE TABLE IF NOT EXISTS `OCPizza`.`user` (  
  `mail` VARCHAR(100) NOT NULL,  
  `lastName` VARCHAR(45) NOT NULL,  
  `firstName` VARCHAR(45) NOT NULL,  
  `birthDate` DATE NOT NULL,  
  `userType` INT NOT NULL,  
  `password` VARCHAR(45) NOT NULL,  
  `favoritePizzeria` INT NOT NULL,  
  `adress` VARCHAR(36) NOT NULL,  
  PRIMARY KEY (`mail`),  
  INDEX `favoritePizzeria_idx` (`favoritePizzeria` ASC),  
  UNIQUE INDEX `mail_UNIQUE` (`mail` ASC),  
  INDEX `adress_idx` (`adress` ASC),  
  INDEX `userType_idx` (`userType` ASC),  
  CONSTRAINT `pizzeria`  
  FOREIGN KEY (`favoritePizzeria`)  
  REFERENCES `OCPizza`.`pizzeria` (`id`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION,  
  CONSTRAINT `adress2`  
  FOREIGN KEY (`adress`)  
  REFERENCES `OCPizza`.`adress` (`id`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

```
-- -----  
-- Table `OCPizza`.`order`  
-- -----
```

```
CREATE TABLE IF NOT EXISTS `OCPizza`.`order` (  
  `id` VARCHAR(36) NOT NULL,  
  `orderDate` DATE NOT NULL,  
  `deliveryType` INT NOT NULL,  
  `orderStatus` INT NOT NULL,  
  `user` VARCHAR(100) NOT NULL,  
  `pizzeria` INT NOT NULL,  
  PRIMARY KEY (`id`),  
  INDEX `pizzeria_idx` (`pizzeria` ASC),  
  INDEX `user_idx` (`user` ASC),  
  INDEX `deliveryType_idx` (`deliveryType` ASC),  
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),  
  CONSTRAINT `pizzeria2`  
  FOREIGN KEY (`pizzeria`)
```

```
REFERENCES `OCPizza`.`pizzeria` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `user`
FOREIGN KEY (`user`)
REFERENCES `OCPizza`.`user` (`mail`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`ingredient`
-- -----

CREATE TABLE IF NOT EXISTS `OCPizza`.`ingredient` (
`name` VARCHAR(45) NOT NULL,
PRIMARY KEY (`name`),
UNIQUE INDEX `name_UNIQUE` (`name` ASC))
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`assoc_pizzeria_ingredient`
-- -----

CREATE TABLE IF NOT EXISTS `OCPizza`.`assoc_pizzeria_ingredient` (
`pizzeria` INT NOT NULL,
`ingredient` VARCHAR(45) NOT NULL,
`stock` INT NOT NULL,
PRIMARY KEY (`pizzeria`, `ingredient`),
INDEX `ingredient_idx` (`ingredient` ASC),
CONSTRAINT `pizzeria3`
FOREIGN KEY (`pizzeria`)
REFERENCES `OCPizza`.`pizzeria` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `ingredient`
FOREIGN KEY (`ingredient`)
REFERENCES `OCPizza`.`ingredient` (`name`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`pizza`
-- -----

-- DROP TABLE IF EXISTS `OCPizza`.`pizza`;
CREATE TABLE IF NOT EXISTS `OCPizza`.`pizza` (
`name` VARCHAR(45) NOT NULL,
`image` LONGTEXT NULL,
`description` LONGTEXT NULL,
```

```
`recipe` LONGTEXT NOT NULL,
`unitPriceExcludingTaxInCents` INT NOT NULL,
UNIQUE INDEX `name_UNIQUE` (`name` ASC),
PRIMARY KEY (`name`)
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`assoc_pizza_ingredient`
-- -----

CREATE TABLE IF NOT EXISTS `OCPizza`.`assoc_pizza_ingredient` (
`pizza` VARCHAR(45) NOT NULL,
`ingredient` VARCHAR(45) NOT NULL,
`quantity` INT NOT NULL,
PRIMARY KEY (`pizza`, `ingredient`),
CONSTRAINT `pizza`
FOREIGN KEY (`pizza`)
REFERENCES `OCPizza`.`pizza` (`name`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `ingredient2`
FOREIGN KEY (`ingredient`)
REFERENCES `OCPizza`.`ingredient` (`name`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`assoc_order_pizza`
-- -----

CREATE TABLE IF NOT EXISTS `OCPizza`.`assoc_order_pizza` (
`order` VARCHAR(36) NOT NULL,
`pizza` VARCHAR(45) NOT NULL,
`quantity` INT NOT NULL,
PRIMARY KEY (`order`, `pizza`),
CONSTRAINT `order`
FOREIGN KEY (`order`)
REFERENCES `OCPizza`.`order` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `pizza2`
FOREIGN KEY (`pizza`)
REFERENCES `OCPizza`.`pizza` (`name`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `OCPizza`.`payment`
-- -----
```

```
-----  
CREATE TABLE IF NOT EXISTS `OCPizza`.`payment` (  
  `id` VARCHAR(36) NOT NULL,  
  `paymentMethod` INT NOT NULL,  
  `paymentStatus` INT NOT NULL,  
  `stripeLink` VARCHAR(45) NOT NULL,  
  `amountInCents` INT NOT NULL,  
  `order` VARCHAR(36) NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE INDEX `id_UNIQUE2` (`id` ASC),  
  INDEX `order_idx` (`order` ASC),  
  INDEX `paymentMethod_idx` (`paymentMethod` ASC),  
  INDEX `paymentStatus_idx` (`paymentStatus` ASC),  
  CONSTRAINT `order2`  
  FOREIGN KEY (`order`)  
  REFERENCES `OCPizza`.`order` (`id`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

## 3.5 - Jeu de données

Suite à la création de la base de données, nous avons commencé à implémenter des données dans celle-ci afin de tester son fonctionnement. Vous trouverez le script ci-dessous permettant d'ajouter les premières données dans notre base de données MySQL.

```
-- Table OCPizza`.`adress`
INSERT INTO `OCPizza`.`adress` (`id`, `adress`, `zipCode`, `city`, `country`)
VALUES
('1A', '10 rue mallifaud', 38100, 'Grenoble', 'France'),
('1B', '5 rue saint thierry', 51100, 'Reims', 'France'),
('1C', '15 rue saint-jean', 51100, 'Reims', 'France');

-- Table OCPizza`.`pizzeria`
INSERT INTO `OCPizza`.`pizzeria` (`name`, `adress`)
VALUES ('OC Pizza Reims', '1B'), ('OC Pizza Grenoble', '1A'), ('OC Pizza Reims', '1C');

-- Table OCPizza`.`user`
INSERT INTO `OCPizza`.`user` (`mail`, `lastName`, `firstName`, `birthDate`, `userType`, `password`, `favoritePizzeria`, `adress`)
VALUES ('martin.dupont@hotmail.fr', 'Dupont', 'Martin', '1992-11-06', 1, '061192', 1, '1A');

-- Table OCPizza`.`order`
INSERT INTO `OCPizza`.`order` (`id`, `orderDate`, `deliveryType`, `orderStatus`, `user`, `pizzeria`)
VALUES ('1A', '2022-10-14', 1, 1, 'martin.dupont@hotmail.fr', 1);

-- Table OCPizza`.`ingredient`
INSERT INTO `OCPizza`.`ingredient` (`name`) VALUES ('Tomate'), ('Oignons'), ('Pate à pizza'), ('Poivron Rouge'), ('Chorizo');

-- Table OCPizza`.`assoc_pizzeria_ingredient`
INSERT INTO `OCPizza`.`assoc_pizzeria_ingredient` (`pizzeria`, `ingredient`, `stock`)
VALUES ((SELECT min(id) FROM `OCPizza`.`pizzeria` WHERE `name` = 'OC Pizza Reims'), 'Tomate', 10);

-- Table OCPizza`.`pizza`
INSERT INTO `OCPizza`.`pizza` (`name`, `image`, `description`, `recipe`, `unitPriceExcludingTaxInCents`)
VALUES
('La calzone', 'https://assets.afcdn.com/recipe/20161130/7916_w1024h1024c1cx2808cy1872.jpg',
'Pizza incontournable',
'1- Préparer la pate',
1200);

-- Table OCPizza`.`assoc_pizza_ingredient`
INSERT INTO `OCPizza`.`assoc_pizza_ingredient` (`pizza`, `ingredient`, `quantity`)
```



```
VALUES ('La calzone', 'Tomate', 1);
```

```
-- Table OCPizza`.`assoc_order_pizza
```

```
INSERT INTO `OCPizza`.`assoc_order_pizza` (`order`, `pizza`, `quantity`)
```

```
VALUES ('1A', 'La calzone', 2);
```

```
-- Table OCPizza`.`payment
```

```
INSERT INTO `OCPizza`.`payment` (`id`, `paymentMethod`, `paymentStatus`, `stripeLink`, `amountInCents`, `order`)
```

```
VALUES ('1A', 1, 2, 'https://stripe.com/54651464656', 2400, '1A');
```

## 4 - ARCHITECTURE DE DEPLOIEMENT

### 4.1 - Hébergeur

Le système (comprenant site internet, back-office et back-end) sera déployé sur les serveurs d'OVH, entreprise française reconnue ayant des serveurs sécurisés et performants. Comme convenu lors des différents échanges avec le client, nous avons opté pour la formule d'OVH « Hébergement Performance » (détails et tarifs actuels à retrouver sur l'image ci-dessous) avec les options :

- « Anycast DNS » et « CDN Basic » pour accélérer les performances du système,
- « SSL » pour avoir la certification du nom de domaine, le protocole https activé et avoir un site sécurisé.

#### Liez un hébergement web à votre nom de domaine

[Passer cette étape](#)

Trouvez l'hébergement web qui répond à vos besoins parmi notre gamme de solutions. Elles vous offrent simplicité, sécurité et performances.

##### Hébergez votre site web personnel, votre blog ou votre CV en ligne

###### Perso

**3,29 €** HT /mois

soit 39,48 €/an

[Ajouter cette offre d'hébergement](#)

Idéal pour héberger votre site web personnel, votre blog ou votre CV en ligne

Parfait pour 5 sites web

100 Go HDD stockage

Puissance ● ○ ○ ○ ○

Optimisé pour 

##### Héberger un site web professionnel avec des adresses e-mail

###### Pro

**6,59 €** HT /mois

soit 79,08 €/an

[Ajouter cette offre d'hébergement](#)

Tout ce dont vous avez besoin pour héberger un site web professionnel, adresses e-mail incluses

Parfait pour 10 sites web

250 Go HDD stockage

Puissance ● ● ● ○ ○

Optimisé pour  

##### Présentez votre entreprise en ligne avec un hébergement évolutif

###### Performance

**10,99 €** HT /mois

soit 131,88 €/an

1 vCore - 2 Go RAM

[Supprimer cette offre d'hébergement](#)

Hébergement haute performance et évolutif pour mettre en valeur votre entreprise en ligne

Parfait pour 500 sites web

500 Go HDD stockage

Puissance ● ● ● ● ●

Optimisé pour    

##### Votre sélection

Noms de domaine

**oc-pizza.com**
**0,00 €**

Gratuit la première année

Option

**Anycast DNS**
**1,09 €**

Option

**DNSSEC (DNS sécurisé)**
**Inclus**

Hébergement

**Performance**
**131,88 €**

Spécifications

**1 vCore - 2 Go RAM**

Option

**SSL Let's Encrypt**
**Inclus**

Option

**CDN Basic**
**Inclus**

Prix HT /1ère année

**132,97 €**

20% TVA

**26,59 €**
**Total**

5 produits

**159,56 €**
**TTC**
[Continuer la commande →](#)

## 4.2 - Serveur des bases de données

La formule d'hébergement d'OVH pour laquelle nous avons optée, dispose d'un serveur de bases de données privé lié à l'hébergement. Nous utiliserons donc ces serveurs pour les bases de données du système. Nous avons également choisi la technologie MySQL pour la base de données, qui est une technologie libre, gratuite, reconnue et totalement adaptée au projet d'OC Pizza.

## 4.3 - Services Web

Comme indiqué précédemment, nous utiliserons dans le système 2 web services :

- Stripe : un service de facturation et de paiement qui sera déployé dans le système par notre équipe grâce à la documentation officielle de Stripe (<https://stripe.com/docs>), via l'intégration de lignes de codes dans nos fichiers sources.
- MailChimp : un service de campagne de mailing et de mailing automatique qui sera également déployé dans le système par notre équipe grâce à la documentation officielle (<https://mailchimp.com/developer/>), via l'intégration de lignes de codes dans nos fichiers sources.

## 5 - ARCHITECTURE LOGICIELLE

### 5.1 - Principes généraux

La gestion de versions du système sera réalisée par le logiciel Git.

L'architecture du système respectera quant à lui le design pattern MVC (model, view, controller), ce qui signifie que les principaux composants du système seront séparés comme suit :

- Les « models » seront les éléments responsables de la logique du système, de la gestion des données et de leur persistance.
- Les « views » seront les éléments responsables des données et des pages à afficher à l'utilisateur.
- Les « controllers » seront les éléments qui feront le lien entre les « models » et les « views ». Ils traiteront et interpréteront les données des « models » pour les préparer à l'affichage dans les « views ». Ils s'occuperont également de recevoir les données entrées par l'utilisateur et de les communiquer aux modèles

### 5.2 - Structure

Il est à noter que le système sera conçu en plusieurs modules ayant des fonctions distinctes. Nous pourrions par exemple retrouver dans le système :

- Un module « gestion de compte », comprenant tous les fichiers sources consacrés à l'espace membre.
- Un module « Paiement », comprenant tous les fichiers sources consacrés aux paiements, aux encaissements, et à la facturation.
- Un module « Gestion des stocks », comprenant tous les fichiers sources permettant de gérer les stocks des aliments dans chaque pizzeria.
- Un module « statistique », comprenant tous les fichiers sources permettant au gérant des pizzerias et du groupe OC Pizza de consulter les statistiques des différents points de ventes (nombre de commandes, CA ...).

## 6 - POINTS PARTICULIERS

### 6.1 - Gestions des logs

Le stockage, la consultation des logs et la consultation des statistiques sont possibles via le tableau de bord d'OVH.

Après connexion à l'espace client OVH Cloud, cliquez sur l'hébergeur web, puis sur l'onglet Statistiques et logs, enfin sélectionnez la catégorie statistiques ou logs en fonction de ce que vous souhaitez consulter.

Pour plus d'informations, consultez la documentation officielle d'OVH sur ce sujet :

- <https://docs.ovh.com/fr/hosting/mutualise-consulter-les-statistiques-et-les-logs-de-mon-site/>

### 6.2 - Fichiers de configuration

Les fichiers de configuration seront à retrouver dans le dossier config.

Nous ajouterons dans ce dossier les clés API des services comme Stripe et MailChimp. Ces clés seront différentes dans un environnement de production ou de développement.

### 6.3 - Procédure de packaging / livraison

Avec l'accord du client, le système sera déployé dans sa totalité et sera complètement fonctionnel, lors de la livraison.

La livraison des fichiers de code source, des scripts de bases de données, ainsi que la documentation se feront via mails. Ces mails contiendront des liens de dépôt GitHub.

De plus, l'intégralité des identifiants et des mots de passe pouvant être utiles pour accéder aux espaces clients des différents services seront également fournis via mail.