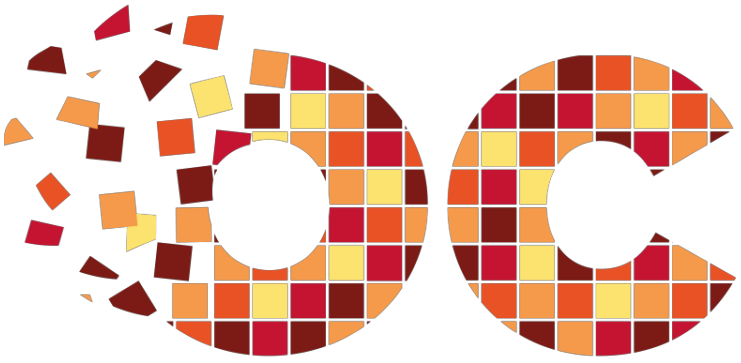
Dossier de Conception Technique

22/09/2018

version 1.0

Document établie par : Morgan Le Bihan



OPENCLASSROOMS

Table des Matières

1. Versions

Version	Date	Auteur	Pages Modifiées	Description
1.0	22/09/2018	MLB	Toutes	document établie à partir des besoins rédigés par OCPizza et du DCF.

	Nom et Qualité	Date et Visa
Auteur	MLB, analyste-programmeur	22/09/2018
Vérificateur		
Approbateur		

2. Introduction

2.1 Objet du document

Ce document présente la conception technique d'un système informatique pour le groupe OCPizza.

Le groupe OCPizza est spécialisé dans les pizzas livrées ou à emporter.

Par convention, ce projet de système informatique sera nommé 'système' dans le reste du document.

Ce document a pour objectif de :

- identifier les différents éléments composant le système à mettre en place et leurs interactions (diagramme de composants)
- décrire le déploiement des différents composants (diagramme de déploiement)
- représenter le modèle physique de données
- présenter les choix techniques

2.2 Références

Les diagrammes figurant dans ce document suivent la norme UML 2.5. Ces diagrammes sont créés avec draw.io.

Le présent document est mis en forme avec Pages.

Les fichiers sources sont disponibles dans le dossier "FichiersSources" joint à ce dossier.

Nous pourrons nous référer aux documents suivants pour des compléments d'information :

- Dossier de Conception Fontionnelle, (DCF)
- Dossier d'Exploitation

3. Architecture technique

3.1 Présentation

Les composants du système sont présentés en page suivante sous forme de diagramme de composants.

Les deux composants principaux du système répondent aux besoins d'OCPizza :

- un site web (CMS marchand) pour commander
- une application iOS pour préparer et livrer la commande

Ils sont arbitrairement représentés en bleu.

Un base de données vient compléter le système en apportant du liant aux deux composants principaux. La base de données permet la persistance des données collectées et manipulées.
Le site web et l'application iOS permettent la mise à jour des données en temps réel.

La base de données est arbitrairement représentée en jaune.

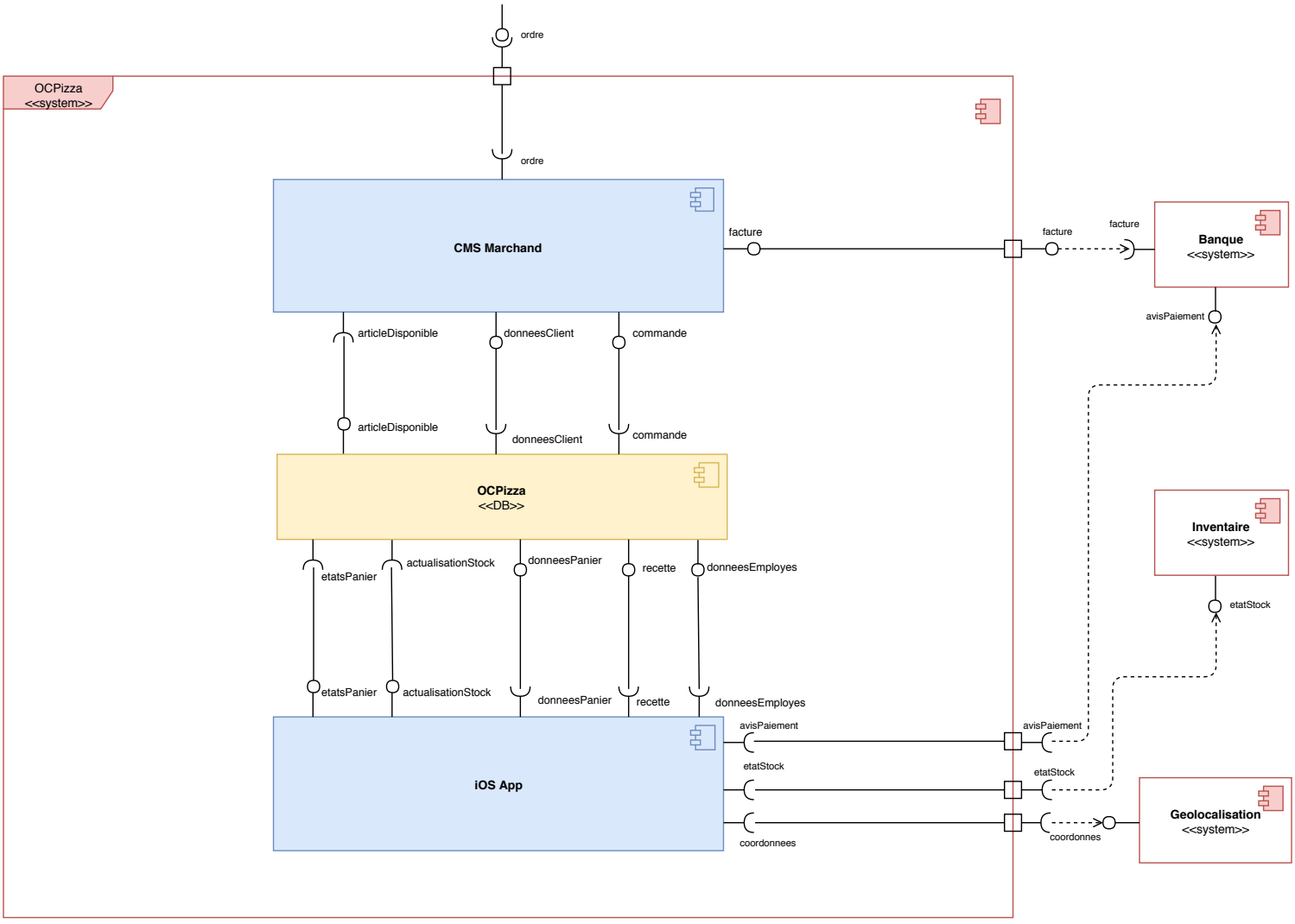
—

Page suivante, un système (<<system>>) est arbitrairement représenté cerclé de rouge.

Les systèmes Banque, Inventaire et Géolocalisation sont extérieurs au système. Ils interagissent avec lui. Ils lui apportent les données nécessaires au bon fonctionnement de ses composants. Des informations complémentaires sont disponibles au paragraphe Systèmes externes.

Le dernier composant extérieur au système est le client, manifesté ici sous la forme d'un ordre fourni au système.

3.2 Diagramme de Composants



4. Architecture de déploiement

4.1 Présentation

Le déploiement du système est présenté page suivante sous forme de diagramme de déploiement.

Il est convenu que :

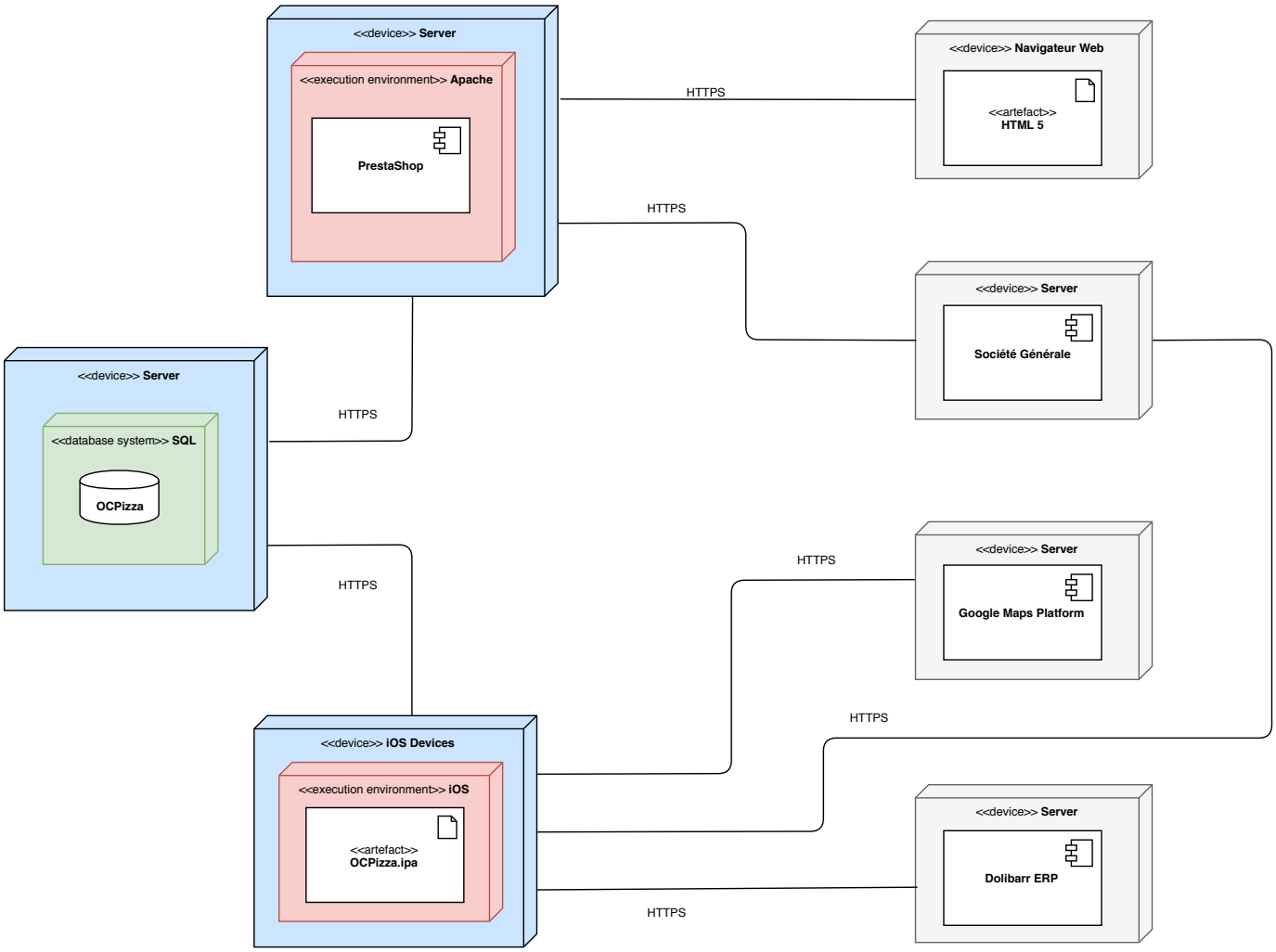
1. le site web est PrestaShop, déployé sur un serveur Apache
2. l'application iOS est déployé sur :
   - iPad WiFi 32Go pour les pizzaiolos et le management opérationnel
   - iPhone 8 64Go pour les livreurs
3. un serveur de base de données est configuré pour un système de gestion de base de données relationnelles MySQL
4. la banque est la Société Générale
5. les données de géolocalisation sont fournies par Google Maps Platform
6. les données de stock sont fournies par le progiciel de gestion intégré (PGI, en anglais Enterprise Resource Planning, ERP) Dolibarr ERP

Chaque entrée de cette liste représente un nœud du déploiement du système. La communication entre ces nœuds est conforme au protocole sécurisé HTTPS.

—

n.b. : la section 6 du document développe les choix de déploiement.

4.2 Diagramme de Déploiement



5. Modèle Physique de Données

5.1 Présentation

Le modèle physique de données (MPD) est représenté graphiquement page suivante.

Il a été conçu à l'aide de MySQL Workbench pour un système de gestion de base de données relationnelle MySQL. Nous nous sommes appuyés sur le diagramme de classes pour la conception des tables :

- noms des colonnes
- types des données
- relation entre les tables

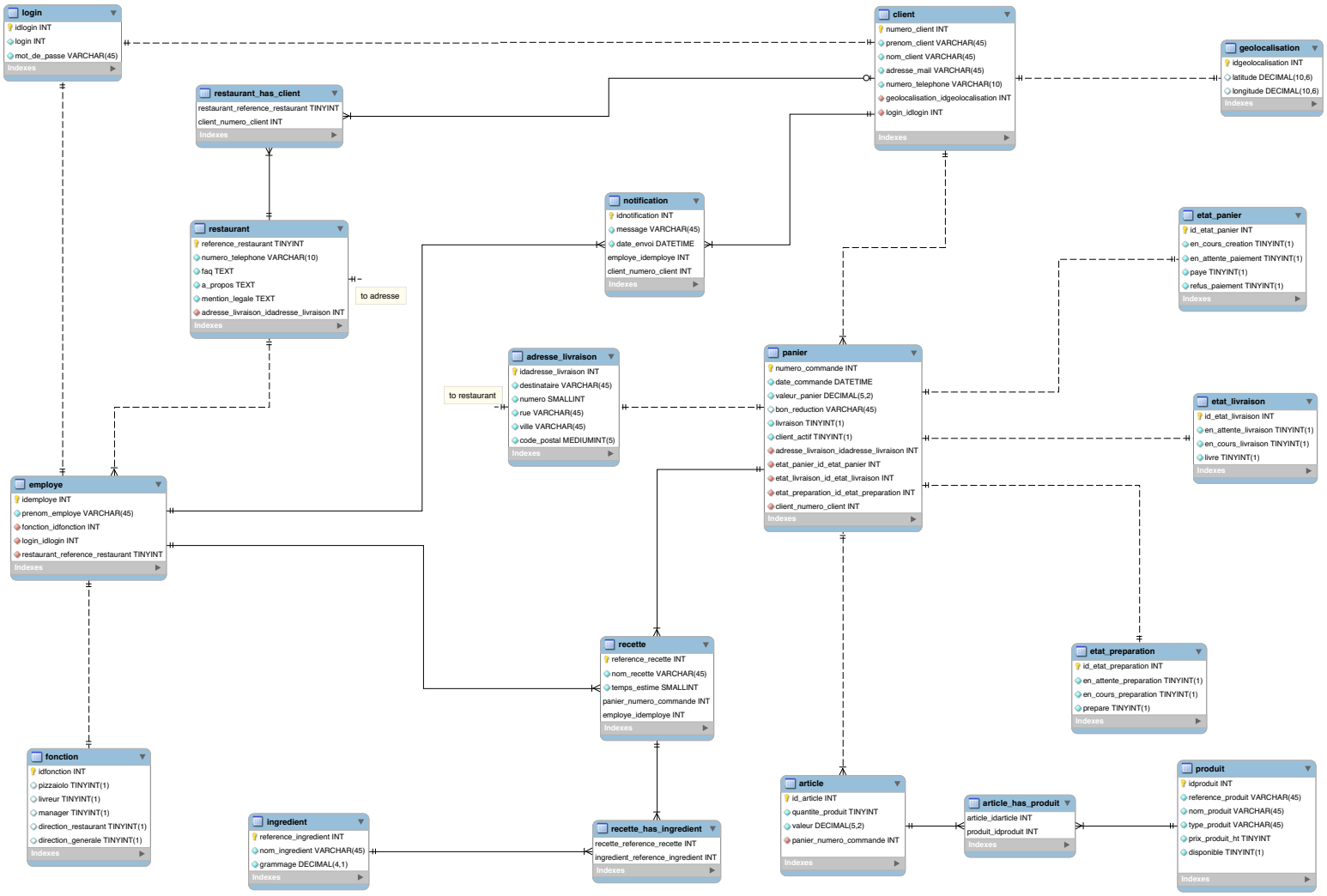
—

n.b.

Le script compressé est joint au présent document.

Le diagramme de Classe est présenté et commenté dans le DCF.

5.2 Modèle Physique de Données MySQL



6. Choix Techniques

6.1 Présentation

6.1.1 Application iOS

Il a été convenu d'utiliser les appareils mobiles Apple (iDevices) pour la gestion des commandes reçues, de leurs préparations et de leurs livraisons parce qu'il est nécessaire d'utiliser des outils robustes et fiables dans un environnement professionnel. Par ailleurs le système de distribution de l'application iOS proposé par Apple est simple à mettre en œuvre et bien connu des développeurs. Ce processus fiable et efficace facilite l'utilisation, la maintenance et l'évolution de l'application.

Les outils de développement décrit dans la section 6.3 sont communément utilisé par les développeurs. Leur connaissance et leur maîtrise de ces outils permettent des gains de productivité.

Nous utilisons le système de flux de développement git flow parce qu'il rationalise le processus de développement collaboratif de nouvelles fonctionnalités et sécurise la version de l'application en production.

Les outils collaboratifs mis en place forment un ensemble cohérent et s'intégrant parfaitement les uns avec les autres. Ces outils sont également bien adaptés à la méthodologie de développement adoptée. Cela a pour avantage de fluidifier tous les processus de travail et de communication, d'être réactif et efficace, et in fine de gagner en productivité.

6.1.2 Site Web

Le CMS marchand PrestaShop a été choisi pour les raisons suivantes :

C'est un système

- personnalisable. Il permet donc de répondre aux besoins formels et fonctionnels d'OCPizza.
- simple à mettre en œuvre et simple à maintenir. Cela représente des gains de productivité.
- simple d'utilisation et dont la communauté est nombreuse et active. Cela offre à OCPizza une autonomie de gestion.

- modulaire. Cela permet de faire évoluer facilement le site web en fonction de l'évolution des besoins d'OCPizza. L'offre en modules couvre un très large spectre des besoins liés à la vente en ligne. Cela permet également des gains de productivité.

Les serveurs sont hébergés par Gandi pour les raisons suivantes :

- service fiable
- partenaire de longue date
- assistance technique irréprochable
- offre compétitive

6.2 Environnement de Développement

6.2.1 Système d'Exploitation

macOS 10.14

6.2.2 Serveur Local

MAMP 4.5

Apache 2.2.34

PHP 7.2.1

MySQL 5.6.38 (moteur InnoDB)

phpMyAdmin 4.7.7

(une liste complète des composants est accessible dans les FAQ de la page d'accueil de MAMP, à l'adresse localhost:8888)

6.2.3 Création du Modèle Physique de Données

MySQL Workbench 6.3

6.2.4 Editeur

Atom 1.29.0

6.3 Application iOS

6.3.1 Outils de Développement

Nous utilisons les dernières versions disponibles des outils de développement.
A ce jour :

Langage
Apple Swift version 4.2

Système d'exploitation
iOS 12

Gestion de version (version control system)
git

Gestion des flux de développement (branch management system)
git flow

Outil de développement collaboratif
Bitbucket

Outil de gestion du travail collaboratif
Trello
Slack

IDE
Xcode 10

Gestion des dépendances
Cocoa pods

Tests unitaires
XCTest

Issue tracker
Bitbucket

Intégration et Déploiement Continus
Jenkins, Fastlane

6.3.2 Gestion du Déploiement

6.3.2.1 Déploiement des Appareils Mobiles

Une vue d'ensemble de l'utilisation en entreprise d'appareils mobiles est présentée dans le document 'Managing Devices and Corporate Data on iOS'.

—

Nous nous référons au document 'iOS Deployment Reference' relatif au déploiement en entreprise des appareils mobiles Apple. Ce document nous informe sur les nécessités d'infrastructures WiFi et réseau. Il nous renseigne sur l'intégration de gestionnaire de parc d'appareils mobiles (Mobile Device Management, MDM).

Les étapes du déploiement sont décrites dans le document 'iOS Deployment Overview for Business'.

Afin de rationaliser l'utilisation des appareils, il est convenu d'utiliser le système suivant :

- modèle de déploiement : propriété de l'entreprise (organization-owned)
- modèle de distribution : non-personnalisé (nonpersonalized)
- modèle de sécurité : supervisé (supervised)

Le modèle de distribution non-personnalisé permet l'échange d'un même iPhone entre livreurs. Le modèle de sécurité supervisé offre plus de flexibilité sur les données accessibles par l'appareil et sur l'intégration à un MDM.

—

Le MDM retenu est hexnode mdm. Nous pouvons nous référer à ce document ou ce document pour un comparatif des alternatives à cette application.

Le choix s'est arrêté sur 'hexnode mdm' avec la solution "entreprise" pour les raisons suivantes :

- tarifs compétitifs et transparents (à fonctionnalités équivalentes)
- interface de gestion intuitive
- excellence de l'assistance technique
- intégration directe et homogène avec VPP
- système évoluant avec les besoins de l'entreprise (politique de gestion et taille du parc d'appareils)

L'ensemble des fonctionnalités est fourni par ce document.
Les étapes d'intégration au MDM sont fournies par ce document.

Document de référence MDM fournit par Apple pour le responsable IT d'OCPizza : cliquer ici.
Documents de référence MDM fournit par Apple pour les développeurs : cliquer ici et ici.

6.3.2.2 Déploiement de l'Application iOS

L'application iOS est distribué dans le cadre du système VPP (Volume Purchase Program).
Cela permet de :

- gérer privativement le déploiement de l'application OCPizza sur les appareils de l'entreprise
- gérer les mises à jours de façon simple, automatique et transparente
- rationaliser la gestion des applications sur appareils mobiles (Mobile Application Management, MAM), c'est-à-dire gérer à partir d'un nœud unique l'ensemble des applications iOS qu'OCPizza juge nécessaire de déployer (en plus de l'application B2B OCPizza — les employés communiquent avec Slack par exemple).

Nous suivons les étapes d'intégration au VPP. Elles sont décrites ici. Le système VPP s'intègre parfaitement au système hexnode mdm.
Le paragraphe "Purchase and distribute custom apps " du Guide Apple Business Manager décrit le fonctionnement du service fournit par Apple pour le déploiement d'application B2B.

6.4 Site Web

CMS marchand

Prestashop 1.7 ou supérieur
Pour une liste des fonctionnalités : cliquer ici
Ressources et documentations exhaustives : cliquer ici

Guide pour les développeurs : cliquer ici
Code source : cliquer ici
Image Docker : cliquer ici

Configuration requise

Apache 2.0 ou version supérieur
PHP 5.4 ou version supérieur

MySQL 5.0 ou version supérieur

6.5 Systèmes externes

Banque

Société Générale, banque actuellement partenaire d'OCPizza

Inventaire

Dolibarr ERP, outil d'inventaire des stocks actuellement utilisé par OCPizza

Le flux des données entre Dolibarr et l'application OCPizza est au format JSON. L'API REST de Dolibarr est documenté à cette adresse

Géolocalisation

Google Maps Platform

Maps API, SDK, tutoriels : cliquer ici

Hébergement des serveurs

Gandi

Gandi CLI

6.6 Contraintes

6.6.1 Applications iOS

- Le code a un historique de commit cohérent et suit les règles d'écriture proposées par Linus Torvald et exposées ici
- Le code est écrit en anglais : commentaires, propriétés, méthodes…
- Le code suit l'API Design Guideline fournit par Apple
- Le projet ne contient aucun warning ni erreur
- L'application iOS est structurée selon le modèle d'architecture MVC (Model-View-Controller)
- L'application contient des tests unitaires qui couvrent la majorité de la logique du code
- L'application est en mode portait pour iPhone et iPad
- L'application supporte iOS 11 et les versions supérieures

6.6.2 Site Web

- Le code a un historique de commit cohérent et suit les règles d'écriture proposées par Linus Torvald
- Le site est responsive
- Le design de l'interface client est en 'Flat Design' et respecte la charte graphique d'OCPizza
- Le site est compatible avec les navigateurs :
  - Firefox 61.0.2 ou supérieur
  - Chrome 68.0.3440.106 ou supérieur
  - Safari 11.1.2 ou supérieur
  - Opera 54 ou supérieur

7. Conclusion

7.1 Choix techniques

Afin de répondre aux besoins exprimés par OCPizza, nous mettons en place :

- une **application iOS** pour
  - la gestion des commandes en temps réels,
  - leurs préparations
  - leurs livraisons
- un site marchand **PrestaShop** pour
  - la gestion des commandes
  - la gestion des clients
- une **base de données relationnelle MySQL** pour
  - structurer les données mises en jeu par le système
  - la gestion des données en temps réels
  - la gestion de leurs persistances

7.2 Méthodologie de Développement

La méthodologie de développement agile utilisée est SCRUM. Les sprints sont sur trois semaines et incluent une revue de code.

7.3 Equipe de développement

Le développement du système est assuré par :

- un Product Owner

- un designer UX/UI
- un développeur front-end
- un développeur back-end
- trois développeurs iOS

7.4 Maintenance

La maintenance corrective est offerte sur une durée limitée à 12 mois. Cette durée est renouvelable et court à partir de la livraison de la version 1.0 du système.

La maintenance évolutive (ajout de nouvelles fonctionnalités) est facturée.

7.5 Livraison

L'échéance pour la livraison d'une première version du système est trois mois après acceptation du devis.

Une formation pour l'utilisation du site PrestaShop et de l'application iOS est inclue. La formation dure une demie journée pour chacun de ces composants.

Annexe

Table des matières

1. Script SQL

Reproduction du script de création de la base de donnée. Le script est compressé et joint à ce document.

```sql
-- MySQL Script generated by MySQL Workbench
-- Fri Aug 17 12:29:29 2018
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

------

-- Schema ocpizza

------

DROP DATABASE IF EXISTS `ocpizza`

------

-- Schema ocpizza

------

CREATE SCHEMA IF NOT EXISTS `ocpizza` DEFAULT CHARACTER SET utf8 ;
USE `ocpizza` ;

------

-- Table `ocpizza`.`geolocalisation`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`geolocalisation` (
  `idgeolocalisation` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `latitude` DECIMAL(10,6) NULL,
  `longitude` DECIMAL(10,6) NULL,
  PRIMARY KEY (`idgeolocalisation`))
ENGINE = InnoDB;

------

-- Table `ocpizza`.`client`

------
```

```
45
46  CREATE TABLE IF NOT EXISTS `ocpizza`.`client` (
47    `numero_client` INT UNSIGNED NOT NULL AUTO_INCREMENT,
48    `prenom_client` VARCHAR(45) NOT NULL,
49    `nom_client` VARCHAR(45) NOT NULL,
50    `adresse_mail` VARCHAR(45) NOT NULL,
51    `numero_telephone` VARCHAR(10) NOT NULL,
52    `mot_de_passe` VARCHAR(45) NOT NULL,
53    `geolocalisation_idgeolocalisation` INT UNSIGNED NULL,
54    PRIMARY KEY (`numero_client`),
55    INDEX `fk_client_geolocalisation_idx`
    (`geolocalisation_idgeolocalisation` ASC),
56    UNIQUE INDEX `adresse_mail_UNIQUE` (`adresse_mail` ASC),
57    UNIQUE INDEX `numero_telephone_UNIQUE` (`numero_telephone` ASC),
58    CONSTRAINT `fk_client_geolocalisation`
59      FOREIGN KEY (`geolocalisation_idgeolocalisation`)
60      REFERENCES `ocpizza`.`geolocalisation` (`idgeolocalisation`)
61      ON DELETE NO ACTION
62      ON UPDATE NO ACTION)
63  ENGINE = InnoDB;
64
65  ------
66
67  -- Table `ocpizza`.`restaurant`
68
69  ------
70
71  CREATE TABLE IF NOT EXISTS `ocpizza`.`restaurant` (
72    `reference_restaurant` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
73    `numero_telephone` VARCHAR(10) NOT NULL,
74    `faq` TEXT NOT NULL,
75    `a_propos` TEXT NOT NULL,
76    `mention_legale` TEXT NOT NULL,
77    PRIMARY KEY (`reference_restaurant`))
78  ENGINE = InnoDB;
79
80  ------
81
82  -- Table `ocpizza`.`adresse`
83
84  ------
85
86  CREATE TABLE IF NOT EXISTS `ocpizza`.`adresse` (
87    `idadresse` INT UNSIGNED NOT NULL AUTO_INCREMENT,
88    `destinataire` VARCHAR(45) NOT NULL,
89    `numero` SMALLINT UNSIGNED NOT NULL,
90    `rue` VARCHAR(45) NOT NULL,
91    `ville` VARCHAR(45) NOT NULL,
92    `code_postal` MEDIUMINT(5) UNSIGNED NOT NULL,
```

```
  PRIMARY KEY (`idadresse`))
ENGINE = InnoDB;

------

-- Table `ocpizza`.`produit`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`produit` (
  `idproduit` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `reference_produit` VARCHAR(45) NOT NULL,
  `nom_produit` VARCHAR(45) NOT NULL,
  `type_produit` VARCHAR(45) NOT NULL,
  `prix_produit_ht` TINYINT UNSIGNED NOT NULL,
  `disponible` TINYINT(1) UNSIGNED NOT NULL,
  PRIMARY KEY (`idproduit`))
ENGINE = InnoDB;

------

-- Table `ocpizza`.`commande`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`commande` (
  `numero_commande` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `date_commande` DATETIME NOT NULL,
  `valeur_panier` DECIMAL(5,2) UNSIGNED NOT NULL,
  `bon_reduction` VARCHAR(45) NULL,
  `livraison` TINYINT(1) UNSIGNED NOT NULL,
  `client_actif` TINYINT(1) UNSIGNED NOT NULL,
  `etat_creation` TINYINT(1) UNSIGNED NOT NULL,
  `etat_paiement` TINYINT(1) UNSIGNED NOT NULL,
  `etat_preparation` TINYINT(1) UNSIGNED NOT NULL,
  `etat_livraison` TINYINT(1) UNSIGNED NOT NULL,
  `client_numero_client` INT UNSIGNED NOT NULL,
  `adresse_idadresse` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`numero_commande`),
  INDEX `fk_panier_client1_idx` (`client_numero_client` ASC),
  INDEX `fk_commande_adresse1_idx` (`adresse_idadresse` ASC),
  CONSTRAINT `fk_panier_client1`
    FOREIGN KEY (`client_numero_client`)
    REFERENCES `ocpizza`.`client` (`numero_client`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_commande_adresse1`
    FOREIGN KEY (`adresse_idadresse`)
    REFERENCES `ocpizza`.`adresse` (`idadresse`)
```

```
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`article`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`article` (
  `id_article` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `quantite_produit` TINYINT UNSIGNED NOT NULL,
  `valeur` DECIMAL(5,2) UNSIGNED NOT NULL,
  `commande_numero_commande` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`id_article`),
  INDEX `fk_article_commande1_idx` (`commande_numero_commande` ASC),
  CONSTRAINT `fk_article_commande1`
    FOREIGN KEY (`commande_numero_commande`)
    REFERENCES `ocpizza`.`commande` (`numero_commande`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`ingredient`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`ingredient` (
  `reference_ingredient` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `nom_ingredient` VARCHAR(45) NOT NULL,
  `grammage` DECIMAL(4,1) UNSIGNED NOT NULL,
  PRIMARY KEY (`reference_ingredient`))
ENGINE = InnoDB;

------

-- Table `ocpizza`.`employe`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`employe` (
  `idemploye` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `prenom_employe` VARCHAR(45) NOT NULL,
  `fonction` TINYINT(2) ZEROFILL NOT NULL,
  `restaurant_reference_restaurant` TINYINT UNSIGNED NOT NULL,
  PRIMARY KEY (`idemploye`),
```

```
  INDEX `fk_employe_restaurant1_idx`
(`restaurant_reference_restaurant` ASC),
  CONSTRAINT `fk_employe_restaurant1`
    FOREIGN KEY (`restaurant_reference_restaurant`)
    REFERENCES `ocpizza`.`restaurant` (`reference_restaurant`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`recette`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`recette` (
  `reference_recette` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `nom_recette` VARCHAR(45) NOT NULL,
  `temps_estime` SMALLINT UNSIGNED NOT NULL,
  `employe_idemploye` INT UNSIGNED NOT NULL,
  `commande_numero_commande` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`reference_recette`, `employe_idemploye`,
`commande_numero_commande`),
  INDEX `fk_recette_employe1_idx` (`employe_idemploye` ASC),
  INDEX `fk_recette_commande1_idx` (`commande_numero_commande` ASC),
  CONSTRAINT `fk_recette_employe1`
    FOREIGN KEY (`employe_idemploye`)
    REFERENCES `ocpizza`.`employe` (`idemploye`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_recette_commande1`
    FOREIGN KEY (`commande_numero_commande`)
    REFERENCES `ocpizza`.`commande` (`numero_commande`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`notification`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`notification` (
  `idnotification` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `message` VARCHAR(45) NOT NULL,
  `date_envoi` DATETIME NOT NULL,
  `employe_idemploye` INT UNSIGNED NOT NULL,
  `client_numero_client` INT UNSIGNED NOT NULL,
```

```
  PRIMARY KEY (`idnotification`, `employe_idemploye`,
`client_numero_client`),
  INDEX `fk_notification_employe1_idx` (`employe_idemploye` ASC),
  INDEX `fk_notification_client1_idx` (`client_numero_client` ASC),
  CONSTRAINT `fk_notification_employe1`
    FOREIGN KEY (`employe_idemploye`)
    REFERENCES `ocpizza`.`employe` (`idemploye`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_notification_client1`
    FOREIGN KEY (`client_numero_client`)
    REFERENCES `ocpizza`.`client` (`numero_client`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`restaurant_has_client`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`restaurant_has_client` (
  `restaurant_reference_restaurant` TINYINT UNSIGNED NOT NULL,
  `client_numero_client` INT UNSIGNED NULL,
  PRIMARY KEY (`restaurant_reference_restaurant`,
`client_numero_client`),
  INDEX `fk_restaurant_has_client_client1_idx` (`client_numero_client`
ASC),
  INDEX `fk_restaurant_has_client_restaurant1_idx`
(`restaurant_reference_restaurant` ASC),
  CONSTRAINT `fk_restaurant_has_client_restaurant1`
    FOREIGN KEY (`restaurant_reference_restaurant`)
    REFERENCES `ocpizza`.`restaurant` (`reference_restaurant`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_restaurant_has_client_client1`
    FOREIGN KEY (`client_numero_client`)
    REFERENCES `ocpizza`.`client` (`numero_client`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`recette_has_ingredient`

------
```

```
CREATE TABLE IF NOT EXISTS `ocpizza`.`recette_has_ingredient` (
  `recette_reference_recette` INT UNSIGNED NOT NULL,
  `ingredient_reference_ingredient` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`recette_reference_recette`,
`ingredient_reference_ingredient`),
  INDEX `fk_recette_has_ingredient_ingredient1_idx`
(`ingredient_reference_ingredient` ASC),
  INDEX `fk_recette_has_ingredient_recette1_idx`
(`recette_reference_recette` ASC),
  CONSTRAINT `fk_recette_has_ingredient_recette1`
    FOREIGN KEY (`recette_reference_recette`)
    REFERENCES `ocpizza`.`recette` (`reference_recette`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_recette_has_ingredient_ingredient1`
    FOREIGN KEY (`ingredient_reference_ingredient`)
    REFERENCES `ocpizza`.`ingredient` (`reference_ingredient`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

------

-- Table `ocpizza`.`article_has_produit`

------

CREATE TABLE IF NOT EXISTS `ocpizza`.`article_has_produit` (
  `article_idarticle` INT UNSIGNED NOT NULL,
  `produit_idproduit` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`article_idarticle`, `produit_idproduit`),
  INDEX `fk_ligne_article_has_article_ligne_article1_idx`
(`article_idarticle` ASC),
  INDEX `fk_ligne_article_has_article_article1_idx`
(`produit_idproduit` ASC),
  CONSTRAINT `fk_ligne_article_has_article_ligne_article1`
    FOREIGN KEY (`article_idarticle`)
    REFERENCES `ocpizza`.`article` (`id_article`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ligne_article_has_article_article1`
    FOREIGN KEY (`produit_idproduit`)
    REFERENCES `ocpizza`.`produit` (`idproduit`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
```

```
327 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```