Groupe OC-Pizza

Projet OC-Pizza

Dossier de conception technique

Version 1.0

Auteur
Sébastien Ziegler
Etudiant/développeur

TABLE DES MATIÈRES

1. VERSIONS

Auteur	Date	Description	Version
Sébastien Ziegler	26/12/2019	Création du document	1.0

2.INTRODUCTION

2.1. Objet du document

Le présent document constitue le dossier de conception technique de l'application OC-Pizza.

Objectif du document est de fournir les spécifications technique du projet OC-Pizza, destiné aux développeurs, aux mainteneurs et à l'équipe technique du client.

2.2. Références

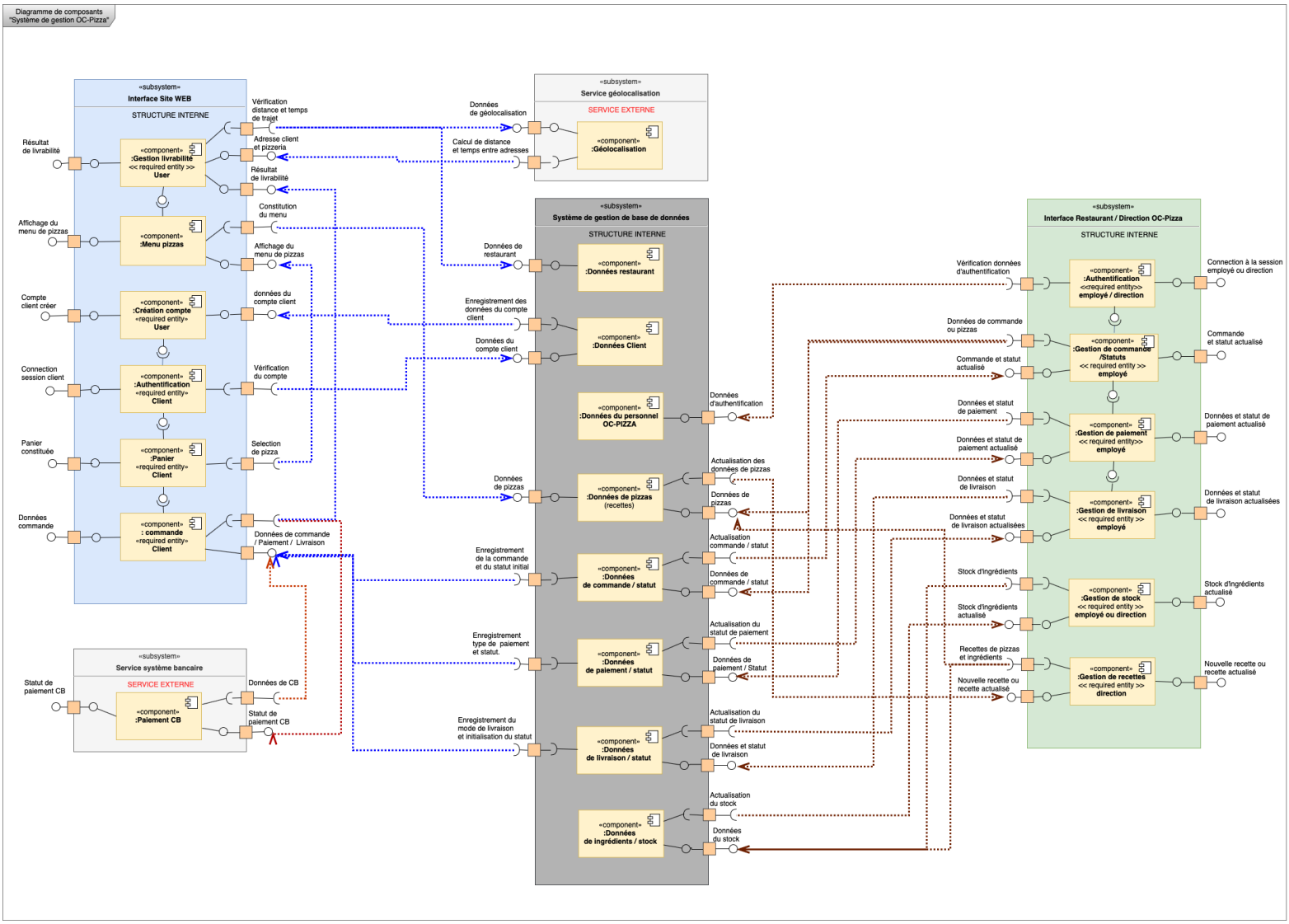
Pour de plus amples informations, se référer également aux éléments suivants:

1. **DCF - Projet OC-Pizza** : Dossier de conception fonctionnelle de l'application
2. **DE - Projet OC-Pizza** : Dossier d'exploitation.
3. **PVR - Projet OC-Pizza** : Procès verbal de réception.

3.Architecture Technique

3.1. Composants

3.1.1 Diagramme UML de composants



3.1.2 Description des sous systèmes

Le système est composé de quatre **sous-systèmes interne** au groupe OC-Pizza, comprenant:

1. L'interface Site WEB, pour la visualisation du menu de pizzas, la création de compte, la constitution du panier ainsi que le passage de commande.

2. Le "Système de gestion de base de données" (SGDB), pour l'enregistrement des données client, des données des acteurs OC-Pizza, des données de pizzas (recettes, description, photo etc…), des données de stock d'ingrédient ainsi que des données et des statuts de commande, de paiement et de livraison.

3. L'interface Restaurant / Direction OC-Pizza, pour la gestion des commandes et d'administration est destiné aux acteurs d'OC-Pizza, leur permettant de s'authentifier, de gérer les commandes (création de commande comptoir, gestion des commandes web), de gérer les paiements, les livraisons, les stocks, les recettes ainsi que les statuts, selon le poste occuper par les acteurs.

Le système est également composé de **deux sous système externe** au groupe OC-Pizza:

1. Le service de géolocalisation (Google Map par exemple) qui va permettre de géolocaliser l'adresse du client et déterminer le temps de trajet nécessaire pour la livraison.

2. Le service système bancaire qui va vérifier la validité des informations de paiement.

3.1.3 Composants de l'interface "Site web"

a. Le composant "**Gestion de livrabilité**" permet de tester si l'utilisateur est livrable. Il requiert le composant "Géolocalisation" auquel il envoie l'adresse saisie par l'utilisateur ainsi que les adresses de pizzerias afin de déterminer les distances et temps de trajets et retournera le résultat de livrabilité.

b. Le composant "**Menu Pizzas**" permet à l'utilisateur de consulter le menu des pizzas. Il requiert le composant "gestion de livrabilité" afin de vérifier si le client est livrable avant de lui permettre de consulter la carte et requiert le composant "données de pizzas" du SGDB afin d'afficher le menu. Le composant renvoi l'affichage du menu.

c. Le composant "**Création compte**" permet à l'utilisateur d'entrée les informations et de créer son compte. Il requiert l'entité "user" (utilisateur) pour entrer les informations de compte (Nom, adresse mail, mot de passe etc…). Ces informations seront ensuite enregistrées par le composant "données client" su SGDB. Le composant renvoi le statut de création de compte.

d. Le composant "**Authentification**" permet au client de s'authentifier au compte client. Il requiert que la création du compte soit préalablement effectuée par le composant "création de compte". Ce composant requiert l'entité client ainsi que le composant "données client" du SGDB, pour respectivement, entrer les informations

d’authentification et les comparer aux informations enregistrées dans la base de données. Le composant renvoi le statut de connexion.

e. Le composant "**Panier**" permet au client de constituer un panier de commande. Il requiert le composant "authentification client"pour vérifier que le client est authentifié, il requiert également le composant "Menu" pour faire son choix parmi parmi les pizzas proposées au menu. Le composant mémorise le panier constitué dans les données de session utilisateur avant l’enregistrement de la commande.

f. Le composant "**Commande**" permet à l'entité "client" de saisir les informations de paiement, de livraison. Il enregistre ensuite ces informations ainsi que le panier préalablement constitué dans la base de données. Il requiert donc le composant "panier" pour la commande du client, il requiert également le composant "gestion livrabilité", pour vérifier si l’adresse de livraison entrée par le client est livrable et le composant "paiement" pour vérifier et valider les informations de paiement dans le cas ou le client aurait choisi le mode de paiement "En ligne".

3.1.4 Composants de l’interface "Employé / Direction OC-Pizza"

a. Le composant "**Authentification**" permet aux acteurs OC-Pizza (employés ou direction) de s’authentifier a leurs comptes pour accéder à leurs interfaces avec les autorisations définies en fonction du poste occupé dans la société. Ce composant requiert l’entité client ainsi que le composant "données du personnel" du SGDB, pour respectivement, entrer les informations d’authentification et les comparer aux informations enregistrées dans la base de données. Le composant renvoi le statut de connexion.

b. Le composant "**Gestion commande/ Statut**" permet aux acteurs d’OC-Pizza, selon leurs attributions définies par leur poste, de créer une commande, de supprimer une commande, de modifier une commande, de visualiser une commande, visualiser une recette et d’en modifier le statut. Le composant requiert les composants "Authentification", "données de pizza" et "données de commande / statut" et met à jour le statut de commande dans la base de données si le statut est modifié par un acteur.

c. Le composant "**Gestion de paiement / Statut**" permet aux acteurs d’OC-Pizza, selon leurs attributions défini par leur poste, de modifier le statut et données de paiement d’une commande. Le composant requiert les composants "données de commande" et "données de paiement" du SGDB afin de visualiser la commande et son statut de paiement et met à jour le statut de paiement et date de paiement dans la base de données si le statut est modifié.

d. Le composant "**Gestion de livraison / Statut**" permet aux acteurs d’OC-Pizza (livreur), selon leurs attributions défini par leur poste, de modifier le statut et les données de livraison. Ce composant requiert les composants "donnée de paiement" afin de visualiser si la commande est réglée ou encore à encaisser. Ce composant met à jour le statut de livraison dans la base de données si le statut est modifié et enregistre la date de livraison si le statut est en phase finale. (statut "livré")

e. Le composant "**Gestion de stock**" permet aux acteurs d’OC-Pizza (employé ou direction), selon les attributions définies par leur poste, de visualiser les ingrédients encore disponibles en stock et d’ajuster les stocks si nécessaire. Ce composant requiert

le composant "données d'ingrédient / stock" du SGBD et enregistre les données actualisées dans la base de données.

f. Le composant "**Gestion de recette**" permet à la direction d'OC-Pizza de créer ou mettre à jour une recette de pizza. Ce composant requiert les composants "données de pizza" et "données d'ingrédients / stock" et enregistre les nouvelles recettes ou recettes de pizzas actualisé dans la base de données.

3.1.5 Composants interface "Système de gestion de base de données"

a. Le composant "**Données client**" enregistre les données du compte l'utilisateur créer par le biais du composant "Création de compte" et renvoie des données du client au composant "Authentification" lorsque l'utilisateur s'authentifie.

b. Le composant "**Données du personnel OC-Pizza**" renvoie des données de l'acteur d'OC-Pizza demandé par le composant "Authentification" lorsque l'acteur s'authentifie.

c. Le composant "**Données de pizzas**" enregistre les nouvelles recettes de pizzas ainsi que les mises à jour de recettes existante et renvoie les données demandées par les composants "Menu pizza", "Gestion de commande / Statut" et "Gestion de recettes".

d. Le composant "**Données de commande**" enregistre les données et mise à jour de la commande ainsi que le statut et la mise à jour du statut de commande. Le composant renvoie les données de commande enregistrées et le statut, demandées par le composant "Gestion de commande / Statut".

e. Le composant "**Données de paiement /Statut**" enregistre les données de paiement ainsi que le statut et la mise à jour du statut de la commande. Le composant renvoie les données de paiement enregistrées ainsi que le statut, demandées par le composant "Gestion de paiement / Statut".

f. Le composant "**Données de livraison /Statut**" enregistre les données de livraison ainsi que le statut et la mise à jour du statut de la commande. Le composant renvoie les données de paiement enregistrées ainsi que le statut, demandés par le composant "Gestion de paiement / Statut".

g. Le composant "**Données ingrédients / Stock**" enregistre les données de stock actualisées par les acteurs OC-Pizza qui disposent de l'autorisations de modifier les stocks (Pizzaiolo et Direction) et retourne les données de stock au composant "Gestion de stock" pour mettre à jour les données.

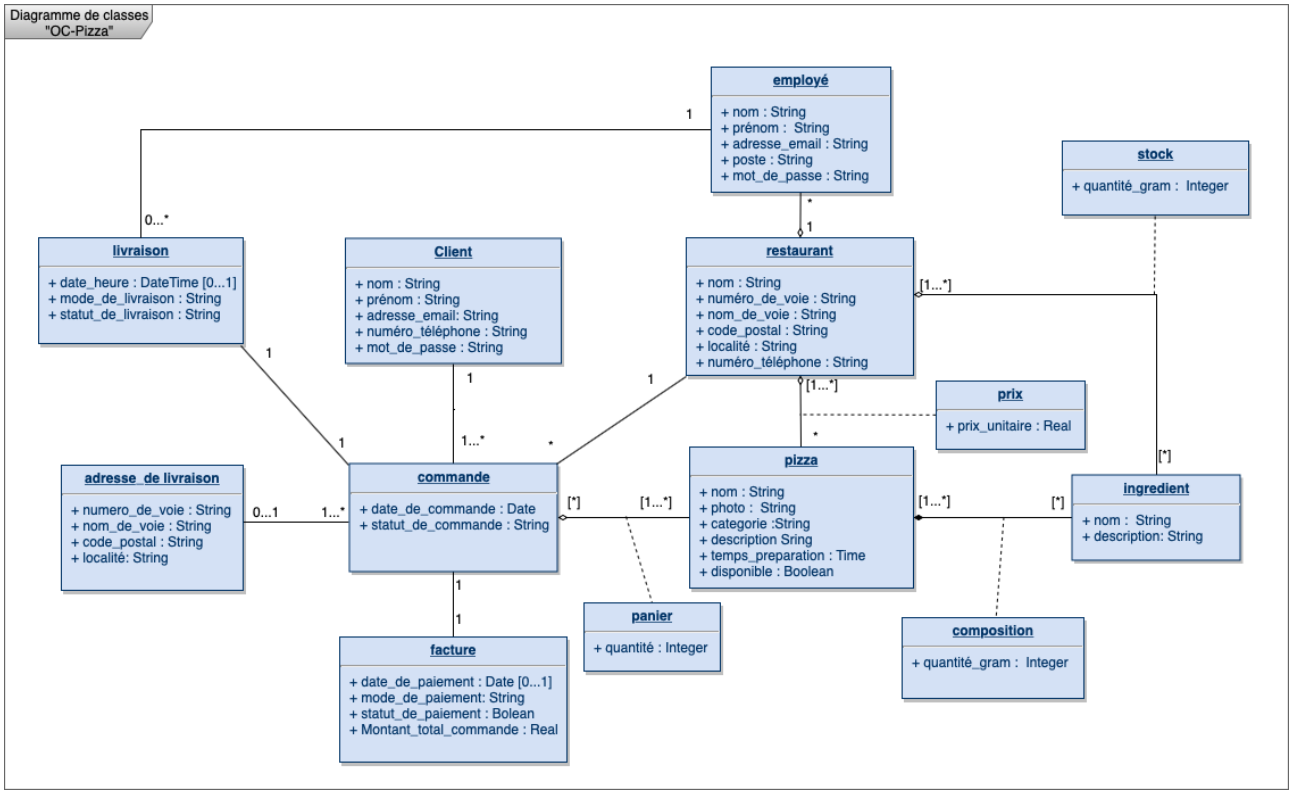
3.1.6 Composants de l'interface "Système banquaire"

a. Le composant "**Paiement CB**" vérifie les informations de paiement lors de la validation de commande si le client à valider un paiement "En ligne" et entré ses données de CB. Ce composant retourne le type de paiement et le statut au composant "données de paiement / Statut".

3.1.7 Composants de l'interface "Système de géolocalisation"

a. Le composant "**Géolocalisation**" reçois les adresses source et destination (adresse client et adresses de pizzeria), puis retourne les données de distance et de temps de trajet entre les adresses au composant "Gestion de livrabilité".

3.2.La base de données.



3.2.1 Description générale des relations.

Tous les employés travaillent dans l'un des restaurants d'OC-Pizza, ils devront être authentifiés pour accéder à leur interface, qui, selon le poste occupé, leur permet de gérer une commande (ajout/suppression/modification), de mettre à jour les statuts de commande, de paiement et de livraison, de mettre à jour les stocks ou créer/modifier une recette de pizza . Certains employés seront affectés aux livraisons notamment les livreurs pour les livraisons à domicile ainsi que le réceptionniste pour les livraisons au comptoir.

Chaque restaurant OC-Pizza dispose d'un stock de plusieurs ingrédients, leur permettant de composer plusieurs pizzas. Chaque restaurant a plusieurs recettes (pizza/composition/ingrédient) de pizza, dont certains ingrédients peuvent être communs à plusieurs recettes et dont le prix peut varier en fonction du restaurant.

Après la création d'un compte et l'authentification, un utilisateur devient un client.
Un client peut passer une ou plusieurs commandes qui sont affectés au restaurant qu'il aura préalablement sélectionné. Le panier, d'une ou de plusieurs pizzas, que le client aura constitué, sera affecté à la commande. Une commande pourra également contenir une adresse de livraison si le client opte pour le mode de livraison "A domicile". Chaque commande a une livraison, soit à domicile, soit au comptoir, dont le statut, ainsi que la date et heure sont mises à jour après la livraison. Chaque commande a également une facture dont le montant total de la commande et le statut sont initialisés lors de l'enregistrement de la commande et dont le statut et la date de paiement seront mis à jour après le paiement.

3.2.2 Les règles de gestion.

1. Un **employé** est employé dans un **restaurant**.
   Un **restaurant** emploi plusieurs **employé**s.

2. Un **restaurant** stock plusieurs **ingrédient**s.
   Un **ingrédient** est stocké dans un ou plusieurs **restaurant**s.
   La relation entre les tables "**restaurant**" et "**ingrédient**" est une association de type "agrégation"

3. Un **restaurant** propose plusieurs **pizza**s.
   Une **pizza** peut être proposée dans un ou plusieurs **restaurant**s.

4. Une **pizza** est composé de plusieurs **ingrédient**s.
   Un **ingrédient** est un composant d'une ou plusieurs **pizza**s.
   La relation entre les tables "**pizza**" et "**ingrédient**" est une association de type "composition"

5. Une **commande** est constitué d'une ou plusieurs **pizza**s.
   Une **pizza** peut constituer une ou plusieurs **commande**s.

6. Un **restaurant** traite plusieurs **commande**s.
   Une **commande** est traité par un **restaurant**.

7. Une **commande** est passé par un **client**.
   Une **client** peut passer une ou plusieurs **commande**s.

8. Une **commande** nécessite une **livraison** (A domicile ou au comptoir).
   Une **livraison** est nécessaire pour une **commande.**

9. Une **commande** génère une **facture.**
   Une **facture** est généré par une **commande.**

10. Une **adresse_de_livraison** est associée à une **commande**.
    Une **commande** peut être associée à une **adresse_de_livraison** si la livraison est "A domicile".

11. Un **employé** livre une **livraison**.
    Une **livraison** est livrée par un **employé** (le livreur ou le réceptionniste).

3.3. Applications Backend

La pile logicielle est la suivante :

- **Python >= 3.8**
- **Django v.2.2 LTS**
- **MySQL >= 8.0**
- Serveur d'application **Gunicorn v.20.0.4**
- **Supervisor >= 4.0.1**
- **New Relic (Infrastructure Monitoring)**
- **SDK Sentry (Application Monitoring)**

Les applications telles que Django ou Gunicorn tournent dans un environnement virtuel (virtualenv) python3.

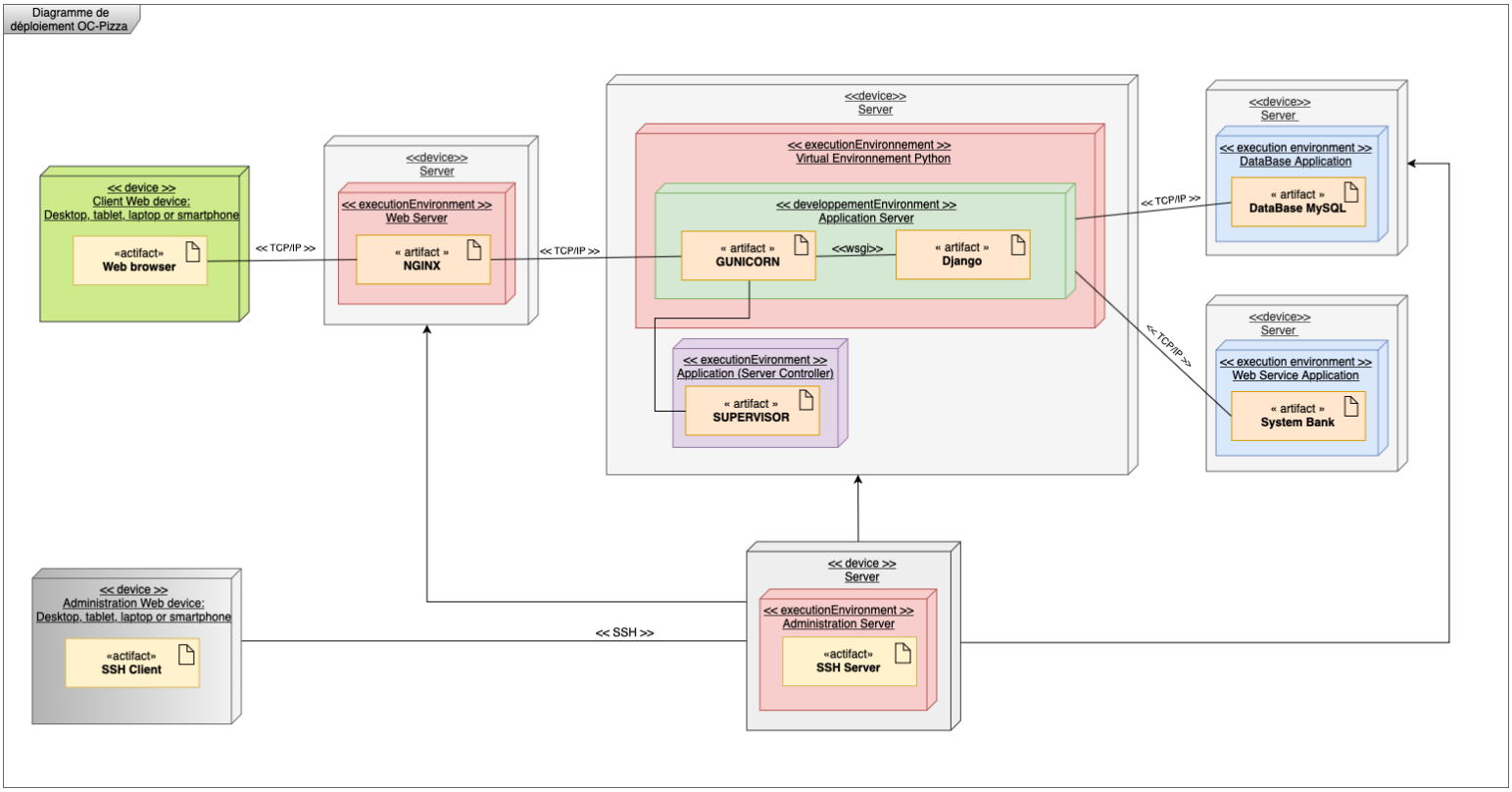
3.4. Applications Frontend

La pile logicielle est la suivante:

- **JQuery >= v3.4.0**
- **Bootstrap >= v4.4.0**
- **ReactJS >= v16.12.0**

4.Architecture de Déploiement

Diagramme UML de déploiement



Le système sera constituer de 4 serveurs logiciels installer sur un serveur physique, internes au système OC-Pizza et fera appel à un serveur externe pour le traitement des données bancaires.

4.1. Serveur web NGINX (Proxy inverse)

Le système utilise le serveur Nginx qui est un serveur web HTTP proxy inverse, qui va rediriger toutes les requêtes entrante, effectué par le client via son navigateur web sur l'adresse ip public, vers l'adresse ip privé du serveur de production Gunicorn et renverra le résultat de la requête au navigateur web du client. Nginx se servira directement des fichiers statiques sans passer par le serveur d'application Gunicorn. La communication se fait par protocole TCP/IP. Nginx est installé sur le serveur physique de l'hébergeur Gandi.

4.2. Serveur d'application Gunicorn

Gunicorn est un serveur d'application wsgi de production qui s'exécute dans l' environnement python3 et qui peux recevoir les requêtes HTTP et renvoyer le résultat directement depuis une adresse IP local . Gunicorn exécute le code de l'application Django par le biais de l'interface wsgi (Web Server Gateway Interface). Il reçoit des requêtes du serveur Nginx et lui retourne les résultats. Gunicorn est installé dans l'environnement virtuel python 3, installé sur le serveur physique de l'hébergeur Gandi.

4.3.L'environnement de développement Django

Django est l'environnement d'exécution dans lequel se trouve le code python de l'application qui est interprétée et exécutée par le serveur d'application Gunicorn. Django fonctionne également dans l' environnement virtuel python3 , installé sur le serveur physique de l'hébergeur Gandi.

4.4. L'application de surveillance Supervisor.

Supervisor est une application de surveillance de processus. Il surveille les processus du serveur Gunicorn afin de s'assurer que celui-ci est bien en fonctionnement. Lorsque le processus Supervisor est démarré, celui-ci démarre automatiquement le serveur Gunicorn s'il est à l'arrêt ou tente de le redémarrage si le démarrage à échoué. Supervisor est installé sur le serveur physique de l'hébergeur Gandi

4.5. Le serveur de base de données MySQL (SGBD)

Le système utilise MySQL, qui est le serveur de base de données qui reçoit des requêtes SQL du serveur d'application et qui lui retourne les résultats des requêtes. Le serveur MySQL est installer sur le serveur physique de l'hébergeur Gandi.

4.6. Le serveur d'administration SSH (Secure Shell)

Le serveur SSH doit également être installé et configurer sur le serveur physique. Celui-ci va permettre aux administrateurs systèmes de configurer le serveur à distance en mode console par le protocole SSH de communication sécurisée en utilisant une application cliente ssh. Le serveur ssh est également installé sur le serveur physique de l'hébergeur Gandi.

4.7. L’infrastructure

Le système est déployé sur un serveur cloud chez l’hébergeur Gandi et dispose du système d’exploitation Linux Debian 10 (Buster) stable.

5.ARCHITECTURE LOGICIELLE

5.1.Principes généraux

Les sources et versions du projet sont gérées par **Github**. L'intégration continu sera gérer par Travis-ci.

5.1.1.Les applications du projet.

L'architecture applicative est similaire à celle défini par les packages.

Le projet est donc composé des applications suivante:

- L'application "**gestion_client**".
- L'application "**gestion_pizzeria**".
- L'application "**administration_groupe**".

Ces applications sont développées dans l'**environnement de développement** "**Django**" qui est installé dans un **environnement virtuel** "**Python 3**" avec tous les packages nécessaires au fonctionnement des applications (les "**requirements**").

Le serveur d'application "**Gunicorn**" est également installé dans l'**environnement virtuel python**.

5.1.2.Les applications du serveur.

Les serveurs logiciels, décrits dans la partie déploiement, sont installé sur le serveur physique sous Linux Debian 10 de l'hébergeur Gandi, auquel s'ajoute l'environnement virtuel, qui contient toute la partie applicative cité ci-dessus, ainsi que le protocole de communication sécurisé SSL.

L'architecture logiciel installé sur serveur Debian 10 est donc la suivante.

- Le serveur proxy inverse **NGINX** .
- Le serveur de base de données **MySQL** (mysql-server).
- **Python 3** et l'environnement virtuel **Virtualenv** (pipenv).
- Le serveur d'administration **SSH** (open-ssh).
- Le protocole sécurisé **SSL** (open-ssl).
- Le service de monitoring de l'infrastructure **New Relic**.

6.POINTS PARTICULIERS

6.1.Gestion des logs (monitoring)

Pour la partie infrastructure, les logs seront gérer via le service web New Relic.

Pour la partie applicative (logiciel), les logs seront gérer via le service web sentry.io.

6.2.Fichiers de configuration

Le fichier de configuration de production "product.py" contient les variables d'environnement de production et les données d'accès au serveur MySQL. Il se trouve dans le répertoire "ressources/fichiers_configurations/" du repository Github et doit être copié sur le serveur de production dans le répertoire "settings" de l'application principal du projet et être rempli avec les paramètres spécifiques.

Pour des raisons de sécurité, ce fichier n'est pas traqué par Git et ne sera pas sur Github s'il est copié dans le répertoire "setttings".

Les paramètres de ce fichier doit être complété directement sur le serveur avec les paramètres spécifiques au projet.

production.py

```
SECRET_KEY = 'Votre clé secrête' # Remplacez par votre clé secrète d'application
DEBUG = False
ALLOWED_HOSTS = ['oc-pizza.fr', 'www.oc-pizza.fr'] # Adresses autorisées à se connecter

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql', # Adaptateur MySQL
        'NAME': 'nom de la base de donnée', # Remplacer par le nom de notre base de données créée.
        'USER': 'nom utilisateur', # Remplacez par votre nom d'utilisateur de la base de données.
        'PASSWORD': 'mot de passe', # Remplacez par le mot de passe d'accès à la base de données.
        'HOST': '',
        'PORT': '5432',
    }
}


sentry_sdk.init(
    dsn="https://clé DNS de sentry", # Remplacez par la clé DNS fourni par Sentry
    integrations=[DjangoIntegration()],
    send_default_pii=True
)
```

6.2.1.Applications web

Les applications sont installé depuis le repository Github.

La procédure d'installation complète est décrite dans le fichier README.md du repository.

6.2.2.Datasources

Le fichier au format SQL "recettes.sql" avec les recettes de pizzas mise au catalogue dans les différents points de vente est disponible dans le répertoire "ressources/fichiers_configurations/" du repository Github et doit-être intégré à la base de données MySQL après migration des tables.

6.3.Ressources

Afin de fonctionner dans les meilleurs conditions et de permettre de nombreuses connections simultanées au site OC-Pizza aux heures de grande affluence (heure de repas), nous recommandons un serveur qui dispose au minimum de, 8Gb de RAM, d'un processeur CPU 4 coeurs et d'un espace de stockage de 25Gb. L'hébergeur Gandi donne la possibilité d'augmenter ou diminuer les resources en fonction des besoins réels sans migration.

6.4.Nom de domaine et SSL

Le nom de domaine "oc-pizza.fr" est acheté chez l'hébergeur / registrar Gandi.
Afin bénéficier d'une connexion web "https" sécurisé et certifié, un certificat signé est demandé auprès de l'AC (Autorité de Certification) Gandi, pour le nom de domaine "oc-pizza.fr" et ses sous domaines (ex: "admin-groupe.oc-pizza.fr").
Le protocole de communication SSL (open-ssl), est également installé sur le serveur.

6.5.Environnement de développement

L'environnement de développement est Django version 2.2 LTS (Long-Term support) pour permettre une pérennité du support jusqu'en 2022.

6.6.Procédure de packaging / livraison

Le projet est livré par le biais d'un repository Github avec accès privé pour le client OC-Pizza.
Un dossier "ressources" est disponible à la racine du repository Github avec la documentations et les fichiers de configuration tel que le fichier de configuration de production (production.py), les fichiers de configuration du serveur Nginx (oc-pizza) et de Supervisor (oc_pizza.conf), le fichier de création de la base de données et des recettes de pizza (creation_OC_Pizza_db_recettes.backup.sql).