OC Pizza

oc_pizza

Dossier de conception technique

Version 1.2

Auteur
Nathan
Développeur

Table des matières

1 - Versions

Auteur	Date	Description	Version
Nathan	27/06/20	Création du document	v1.0
Nathan	12/07/20	Ajout de dernières sections	v1.1
Nathan	13/07/20	Ajustement du format	v1.2

2 - Introduction

2.1 - Objet du document

Le présent document constitue le dossier de conception technique de l'application OC Pizza qui sera développer prochainement

Objectif du document est de décrire en détails les différents élément de la conception technique

Les éléments du présents dossiers découlent :

- du domaine fonctionnel
- de l'architecture technique
- de l'architecture de déploiement
- de l'architecture Logiciel

2.2 - Références

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

1. **DCF – OC Pizza** : Dossier de conception fonctionnelle de l'application
2. **DCT – OC Pizza** : Dossier d'exploitation de l'application

2.3 - Besoin du Client

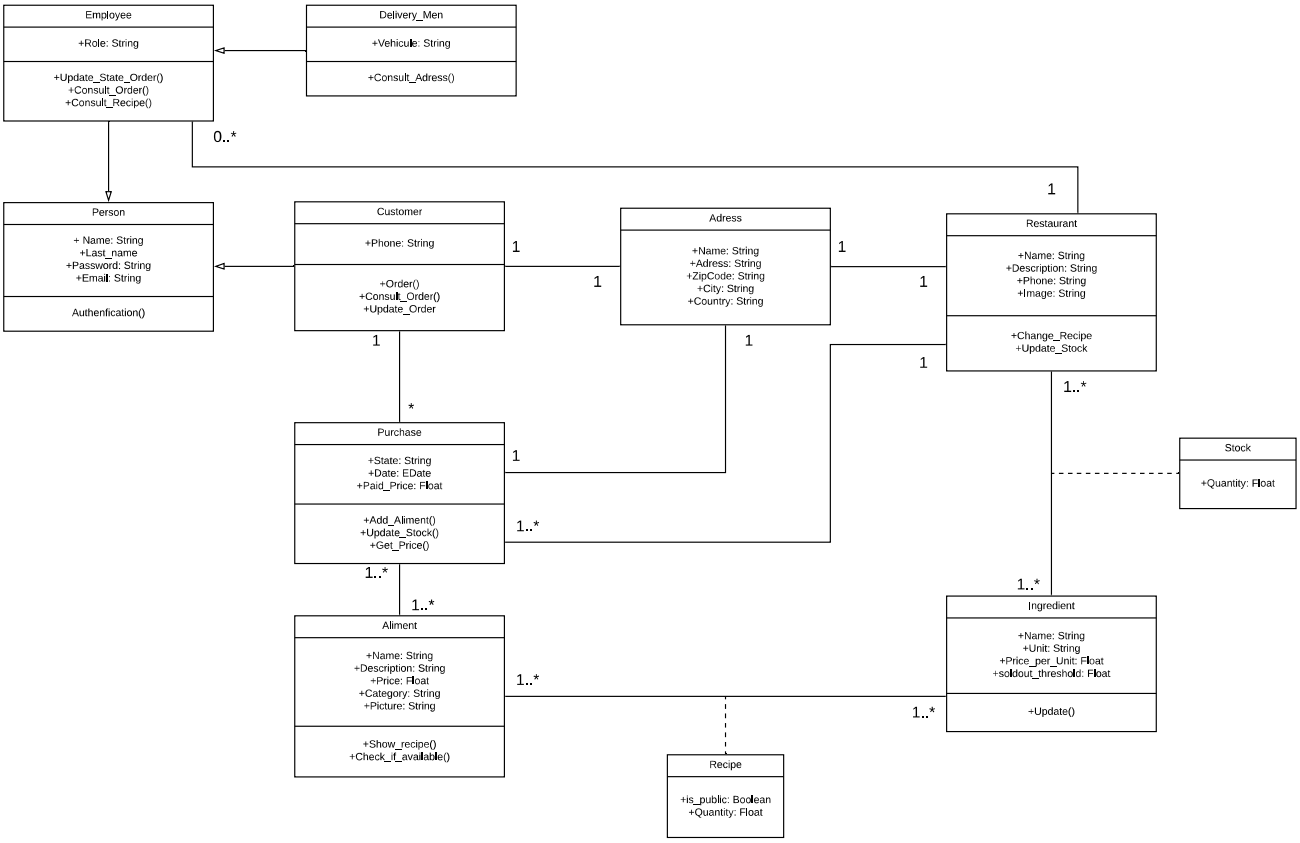
« OC Pizza » est un jeune groupe de pizzeria en plein essor et spécialisé dans les pizzas livrées ou à emporter. Il compte déjà 5 points de vente et prévoit d'en ouvrir au moins 3 de plus d'ici la fin de l'année. Un des responsables du groupe a pris contact avec vous afin de mettre en place un système informatique, déployé dans toutes ses pizzerias et qui lui permettrait notamment :d'être plus efficace dans la gestion des commandes, de leur réception à leur livraison en passant par leur préparation ;

- de suivre en temps réel les commandes passées et en préparation ;
- de suivre en temps réel le stock d'ingrédients restants pour savoir quelles pizzas sont encore réalisables ;
- de proposer un site Internet pour que les clients puissent :
  - passer leurs commandes, en plus de la prise de commande par téléphone ou sur place
  - payer en ligne leur commande s'ils le souhaitent – sinon, ils paieront directement à la livraison
- modifier ou annuler leur commande tant que celle-ci n'a pas été préparée
- de proposer un aide mémoire aux pizzaiolos indiquant la recette de chaque pizza
- d'informer ou notifier les clients sur l'état de leur commande

3 - Le domaine fonctionnel

3.1 - Référentiel

Diagramme de classe de l'application OC Pizza décrivant chaques classe et leurs liens.
Ces classes correspondent au modèle de Base de donnée de l'application.



3.1.1 - Règles de gestion

3.1.1.1 - Association Customer/Adress

Definition des classes :

- Customer : Représente un Client de OC Pizza.
- Adress : Représente une adresse détaillée.

Relation des classes :

Relation de type Un à Un

- Un Customer peut contenir Une Adress
- Une Adress peut correspondre à Un Customer

3.1.1.2 - Association Customer/Purchase

Definition des classes :

- Customer : Représente un Client de OC Pizza.
- Purchase : Représente une commande d'un client

Relation des classes :

Relation de type Un à Plusieurs

- Un Customer peut contenir Plusieurs Purchase
- Une Purchase peut correspondre à Un Customer

3.1.1.3 - Association Restaurant/Adress

Definition des classes :

- Restaurant : Représente un Restaurant appartenant à OC Pizza
- Adress : Représente une adresse détaillée.

Relation des classes :

Relation de type Un à Un

- Un Restaurant peut contenir Une Adress
- Une Adress peut correspondre à Un Restaurant

3.1.1.4 - Association Restaurant/Employee

Definition des classes :

- Restaurant : Représente un Restaurant appartenant à OC Pizza
- Employee : Représente un Employée d'un Restaurant

Relation des classes :

Relation de type Un à Plusieurs

- Un Restaurant peut contenir Plusieurs Employee.
- Un Employee peut appartenir à Un Restaurant

3.1.1.5 - Association Restaurant/Ingredient

Definition des classes :

- Restaurant : Représente un Restaurant appartenant à OC Pizza
- Ingredient : Représente un Ingrédient pour réalisé un Produit

Relation des classes :

Relation de type Plusieurs à Plusieurs

- Un Restaurant peut contenir Plusieurs Ingredient.
- Un Ingredient peut appartenir à Plusieurs Restaurant

3.1.1.6 - Association Restaurant/Purchase

Definition des classes :

- Restaurant : Représente un Restaurant appartenant à OC Pizza
- Purchase : Représente une commande d'un client

Relation des classes :

Relation de type Un à Plusieurs

- Un Restaurant peut contenir Plusieurs Purchase.
- Un Purchase peut appartenir à Un Restaurant

3.1.1.7 - Association Adress/Purchase

Definition des classes :

- Adress : Représente une adresse détaillée.
- Purchase : Représente une commande d'un client

Relation des classes :

Relation de type Un à Un

- Une Purchase peut contenir Une Adress.
- Une Adress peut appartenir à Une Purchase.

3.1.1.8 - Association Aliment/Purchase

Definition des classes :

- Aliment : Représente un aliment commandable (Pizza, Boisson, Dessert)
- Purchase : Représente une commande d'un client

Relation des classes :

Relation de type Plusieurs à Plusieurs

- Une Purchase peut contenir Plusieurs Aliment.
- Un Aliment peut appartenir à Plusieurs Purchase.

3.1.1.9 - Association Aliment/Ingredient

Definition des classes :

- Aliment : Représente un aliment commandable (Pizza, Boisson, Dessert)
- Ingredient : Représente un Ingrédient pour réalisé un Produit

Relation des classes :

Relation de type Plusieurs à Plusieurs

- Un Aliment peut contenir Plusieurs Ingredient.
- Un Ingredient peut appartenir à Plusieurs Aliment.

3.1.1.10 - Person et ces classes Enfants

La classe Person est la classe parents de plusieurs classes qui hérite de ces attributs et de ces méthodes.

Definition des classes :

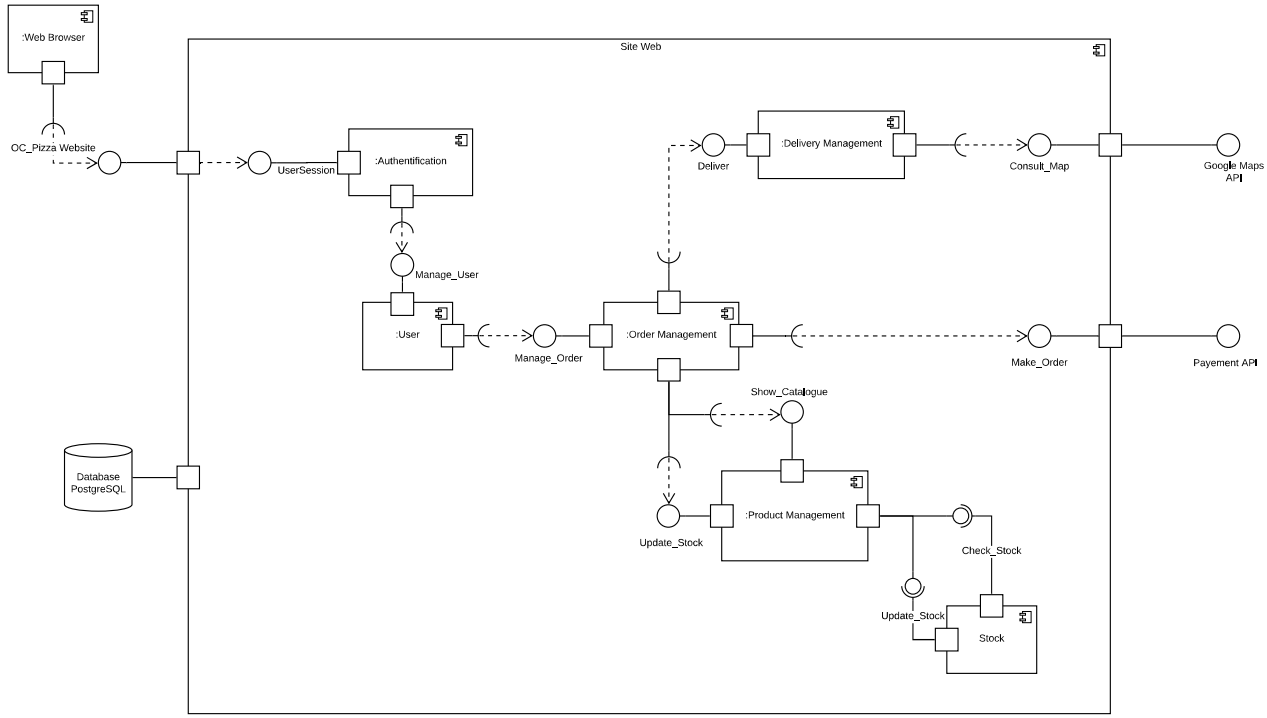
- Person : Représente tout les utilisateurs du site.
- Customer : Représente tout les clients.
- Employee : Représente tout les employée d'OC Pizza.
- Delivery_Men : Réprésente tout les employé du Restaurant qui sont livreurs.

4 - Architecture Technique

4.1 - Application Web

La pile logicielle est la suivante :

- Application **Python 3.8**
- Serveur d'application **Gunicorn 20.0.4**
- Serveur web **Nginx 1.18**
- Serveur de base de donnée **PostgreSQL 9.5**



4.1.1 - Composants Web Browser

Le composants Web Browser est celui que tout les utilisateurs vont acceder notre site web.

4.1.2 - Composants Authentification

Le composants Authentification est celui qui permettra à l'utilisateur à se connecté à son compte. Les accès et services seront différent dépendament du type d'utilisateur.

Ce composants proposera l'interface du compte d'utilisateur

4.1.3 - Composants User

Le composants User est celui qui permettra à tout les utilisateurs d'acceder au service qui lui sont proposé selon son type.

Ce composants proposera l'interface de la gestion des commande.

4.1.4 - Composants Order Management

Le composants Order Management est celui qui permettra aux utilisateurs d'acceder au commande qui leur sont lié, de les modifié si cela est possible et de faire certaines actions.

Ce composants proposera l'interface de livraison, également celle de la conception d'une commande. Mais aussi l'interface permettant d'avoir accès au produit disponible dans le stock et de mettre à jour l'inventaire.

4.1.5 - Composants Delivery Management

Le composants Delivery Management est celui qui permettra à l'utilisateur d'acceder à la livraison en cours et pour les livreurs de gérer leur livraison, d'acceder au informations nécessaires et de mettre à jour l'état de la commande.

Ce composants proposera l'interface de livraison en cours ainsi que la carte fournit par l'API

4.1.6 - Composants Product Management

Le composants Product Management est celui qui permettra à certains utilisateurs et au système d'acceder au stock afin de la mettre à jour. Mais Il permettra aussi à tout les utilisateurs d'acceder au produit disponible dans le Restaurant.

Ce composants se connectera au Stock avec les fonctionnalité “Check_Stock” et “Update_Stock”.

4.1.7 - Composants Stock

Le composants Stock est celui qui permettra au système d'acceder et de mettre à jour l'inventaire du Restaurant.

Ce composants sera accessible via les fonctionnalité “Check_Stock” et “Update_Stock”.

4.1.8 - Composants Database

Le composants Database est celui qui permettra à tout notre site web à ce connecté à notre base de donnée PostgreSQL.

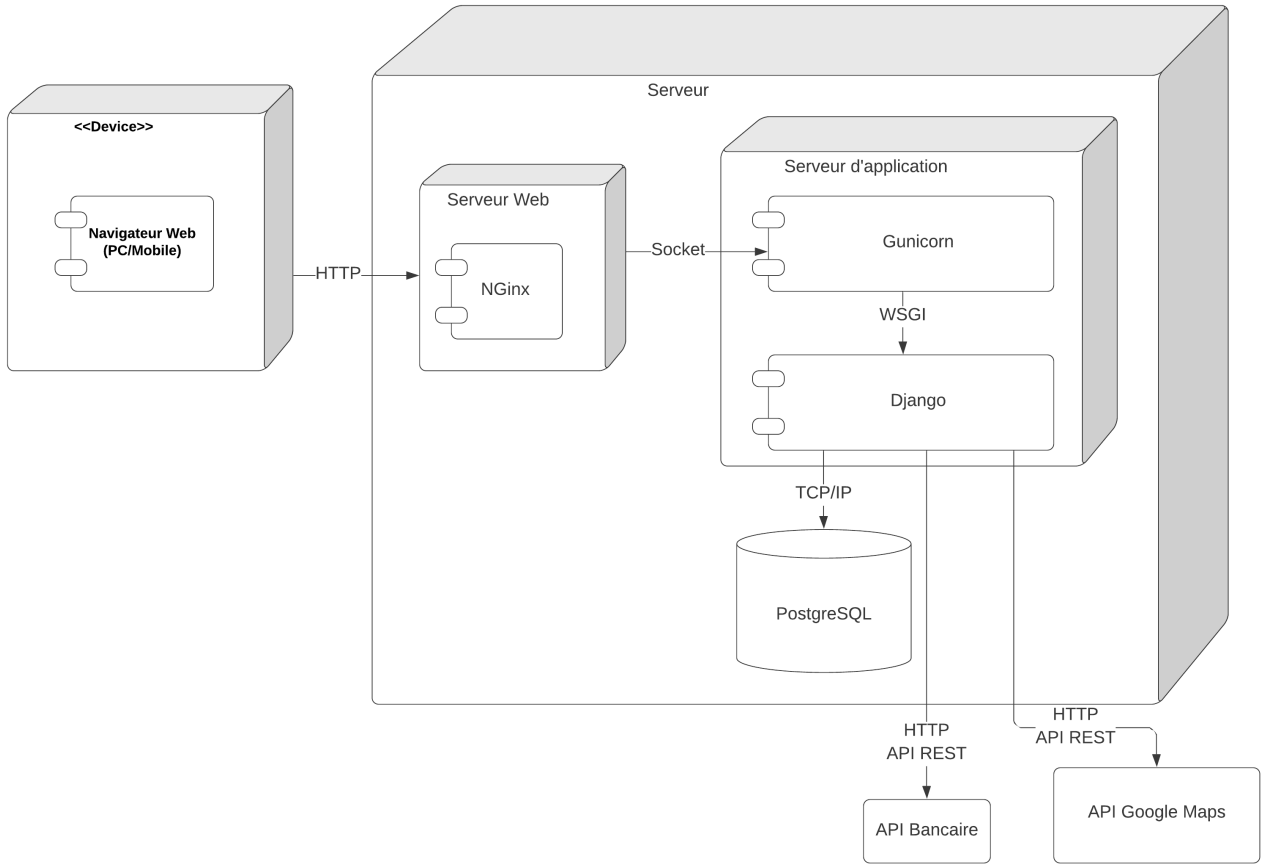
4.1.9 - Composants extérieurs : APIs

Nous avons deux composants extérieurs qui sont tout deux lié à une API.

- L'API de Google Map qui nous permettra d'afficher une carte et réaliser des chemins.
- L'API de Payement qui permettra à l'utilisateur d'effectué son paiement en toute sécurité.

5 - Architecture de Déploiement

Diagramme de Déploiement de l'application OC Pizza décrivant les différents composant qui constituera le serveur et leurs connections.



5.1 - Serveur de Base de données

Le serveur de Base de données communique via le protocole TCP/IP avec le serveur d'application. Ce serveur stock toutes les données lié à notre application de façon structurée.

Caractéristiques techniques :

- Linux Debian 10.3
- PostgreSQL 9.5

Ce serveur sera gérer par l'ORM de l'application Django.

5.2 - Serveur d'application

Gunicorn est un serveur HTTP Python pour Unix qui utilise les spécifications WSGI. Gunicorn est basé sur des sous-processus créés à l'avance. Celui-ci est tout à fait adapté pour la gestion de notre application Django, puisque Django est supporté nativement par Gunicorn

5.3 - Serveur Web

Nginx est un serveur web qui va gérer tout trafic arrivant sur notre serveur. Celui-ci est tout particulièrement puissant grace à son système asynchrone qui va nous permettre des performances élevé avec une charge et une consomations de mémoire particulièrement faible.

6 - Architecture logicielle

6.1 - Principes généraux

Les sources et versions du projet sont gérées par **Git**, les dépendances par **PyPi** et le packaging par **apt-get**

6.1.1 - Les couches

L'architecture applicative est la suivante :

- une couche **model** : implémentation du modèle des objets métiers
- Une couche **template** : représente les gabarits des données renvoyées
- Une couche vue : donnes accès au modèle et au template adapté à la requête

6.1.2 - Structure des sources

La structuration des répertoires du projet suit la logique suivante :

```
racine
├─ venv
├─ requirements.txt
├─ manage.py
├─ <project>_project
│   ├─ urls.py
│   ├─ wsgi.py
│   └─ settigs
│       ├─ __init__.py
│       ├─ productions.py
│       └─ travis.py
├─ <app>
│   ├─ __init__.py
│   ├─ admin.py
│   ├─ apps.py
│   ├─ models.py
│   ├─ urls.py
│   ├─ forms
│   ├─ migrations
│   ├─ static
│   ├─ templates
│   │   └─ index.html
│   ├─ tests
│   └─ views
└─ staticfiles
```

7 - Points particuliers

7.1 - Gestion des logs

Gestion des logs par sentry.io, un lien sera créé lors du déploiement. (Voir Dossier d'exploitation)

7.2 - Fichiers de configuration

7.2.1 - Application web

7.2.1.1 - Fichier productions.py

Localisation : /home/<utilisateur>/oc_pizza/oc_pizza_project/settings/

Il établit les différents modules additionnels au projet tel que le support de Raven qui permet d'envoyer les informations à sentry.io pour le logging ou encore l'authentification à la base de données.

7.2.1.2 - Fichier oc_pizza (Nginx)

Localisation : /etc/nginx/sites-available/oc_pizza

Il permet de dicter le comportement de Nginx vis-à-vis du projet en fonction des ressources demandées. Un fichier présente dans l'arborescence « static » de notre projet sera renvoyé directement par Nginx sans passer par notre application Django afin de gagner en performance.

7.2.1.3 - Fichier oc_pizza-gunicorn.conf (Supervisor)

Localisation : /etc/supervisor/conf.d/oc_pizza-gunicorn.conf

Il permet de définir les paramètres et le contexte dans lequel Supervisor va exécuter notre application. Il a pour but de maintenir notre application en fonctionnement quoi qu'il arrive dans un environnement de production.

7.3 - Environnement de développement

7.3.1.1 - VirtualEnv

Notre application utilise VirtualEnv afin de concentrer l'ensemble des librairies (gérer par Pypi) nécessaire à notre application dans son seul dossier.

8 - Glossaire

API	En informatique, une **interface de programmation d'application** ou **interface de programmation applicative** est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.
Logs	Les **logs** désigne l'enregistrement séquentiel dans un fichier ou une base de données **de tous les événements affectant un processus particulier.**
ORM	Un **mapping objet-relationnel** est un type de programme informatique qui se place en interface entre un programme applicatif et une base de données relationnelle pour simuler une base de données orientée objet
WSGI	La **Web Server Gateway Interface** (WSGI) est une spécification qui définit une interface entre des serveurs et des applications web pour le langage Python.