|  |
| --- |
| **Front Desk Services**  **Front Desk V2**  Dossier de conception fonctionnelle  Version 1.0 |
| **Auteur**  Gaëtan GROND  *Developpeur full stack/Fondateur Front Desk* |

Table des matières

1 -Versions 3

2 -Introduction 4

2.1 -Objet du document 4

2.2 -Références 4

2.3 -Besoin du client 4

2.3.1 -Contexte 4

2.3.2 -Enjeux et Objectifs 4

3 -Description générale de la solution 5

3.1 -Les principe de fonctionnement 5

3.2 -Les acteurs 5

3.3 -Les cas d’utilisation généraux 5

4 -Le domaine fonctionnel 6

4.1 -Référentiel 6

4.1.1 -Règles de gestion 6

4.2 -Package X 6

5 -Les workflows 7

5.1 -Le workflow XXX 7

6 -Application Web 8

6.1 -Les acteurs 8

6.2 -Les cas d’utilisation 8

6.3 -Les règles de gestion générales 8

6.4 -Le workflow XXX 8

7 -Composant XXX 9

8 -Glossaire 10

# Versions

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Description | Version |
| Gaëtan | 21/01/21 | Création du document | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Objet du document

Le présent document constitue le dossier de conception fonctionnelle de l'application pour la nouvelle version de la plateforme Front Desk

L’objectif du document est de définir le périmètre fonctionnel de cette application.

Il vise à informer aussi bien le client (Front Desk) que des potentiels investisseurs et collaborateurs sur le projet Front Desk V2

## Références

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

1. **Front Desk** : Dossier de conception technique de l'application
2. **Front Desk** : Cahier des charges

## Besoin du client

### Contexte

Front desk est une application web qui a vu le jour en janvier 2020, sa popularité auprès des différents corps de métier de l’hôtellerie a mis en lumière certains problèmes d’évolution et de conception, ce projet vise à créer une application web évolutive et répondant aux standards techniques et fonctionnelles du moment.

Front desk est entièrement développé, maintenu et imaginé par un ancien hôtelier.

Fort de 7 années d’expérience dans l’hôtellerie en France et à l’étranger, dans différents groupes hôteliers, du midscale et economy, Gaëtan GROND a pu définir les besoins du métier.

### Enjeux et Objectifs

Front desk est une application web (SPA\*) qui souhaite moderniser le quotidien des hôtels en France et dans le monde en fournissant des outils digitaux faciles à prendre en main et répondant aux besoins des hôteliers.

# Description générale de la solution

## Les principe de fonctionnement

L’application proposée est une application web SPA, aussi appelé application monolithique.

L’application a un design responsive pour être consultée aussi bien sur un écran d’ordinateur, un écran de smartphone ou bien encore une tablette.

Nous proposons de réaliser la partie back-end de cette application en utilisant le langage **Python\*** tout en s’appuyant sur le framework **Django\***.

Pour la réalisation de la partie front-end, nous proposons d’utiliser le langage **Javascript\*** à l’aide du framework **React\***.

Pour ce qui est de la base de données, nous proposons l’utilisation d’une base de données **PostgreSQL\***.

\* Voir le glossaire en fin de document

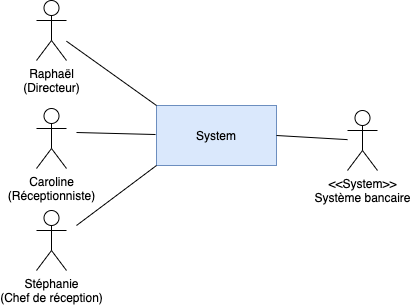
## Les acteurs

Nous allons définir ici trois persona. Les persona sont des personnages imaginaires utilisés

Pour représenter les différents profils d’utilisateurs amenés à se servir du système (Front Desk). Leur prénom permettra de les identifier dans les scénarios et le parcours utilisateur

Une image contenant texte

Description générée automatiquementUne image contenant texte, complet, personne, homme

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

## Les acteurs

Acteurs principaux :

### Raphaël (Directeur)

Raphaël est un directeur d'hôtels de 40ans avec une bonne maitrise des outils digitaux, il souhaite utiliser Front Desk pour gérer ses propriétés et son équipe de manière efficace et moderne.

### Caroline (Réceptionniste)

Caroline est une jeune réceptionniste d'hôtel de 26ans qui souhaite utiliser Front Desk pour digitaliser son shift, elle a une culture digitale avancée

### Stéphanie (Chef de reception)

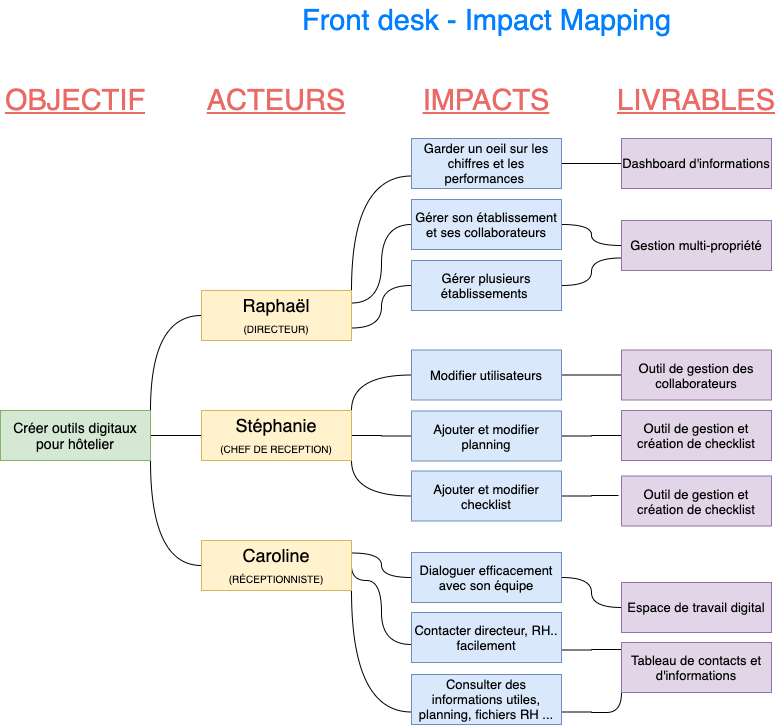
Stéphanie est une chef de réception de 34ans qui souhaite utiliser Front Desk pour gérer ses équipes efficacement, elle a une culture digitale avancée

Acteurs secondaires :

* **Vérification de paiement :** Système permettant de vérifier les paiements des utilisateurs pour l’abonnement si utilise

## Impact mapping

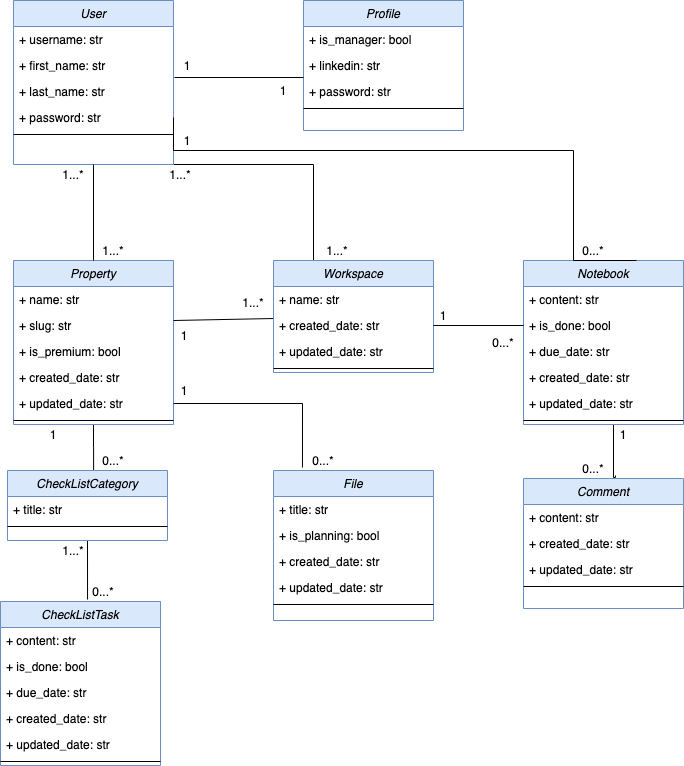
L'impact mapping permet de visualiser les différents objectifs des différents acteurs



# Le domaine fonctionnel

## Diagramme de classe

Diagramme UML de classes



**Relations entre les classes :**

Le but des descriptions ci-après est d'expliquer les liaisons entre les différentes classes du diagramme. Il est important de noter que pour une uniformisation avec le modèle physique de données, les noms des classes ainsi que leurs attributs sont en anglais

### Définitions des classes:

Définitions des classes :

**User :** Représente un utilisateur inscrit sur le site

**Profile:** Représente le profile d’un utilisateur

**Property:** Représente un établissement (ex : Hôtel)

**Workspace :** Représente un espace de travail (ex : Pour communiquer entre collaborateurs)

**Notebook:** Représente un message d’un utilisateur

**Comment:** Représente un commentaire lié à un notebook

**CheckListCategory:** Représente une categorie pour une checklist (ex : Matin, Nuit, Fin de mois..)

**CheckListTask:** Représente une tâche liée à une checklistcategory

### Relations entre les classes:

Nous allons définir ci-dessous les relations entre classe et les multiplicités, pour en savoir plus sur les classes, veuillez-vous référer à la partie **4.1.1**

**User – Profile:**

Un utilisateur est associé à un profil et un profil est associé à utilisateur

**User – Property:**

Un utilisateur est associé à une propriété et un utilisateur est associé à une ou plusieurs propriétés.

**User – Workspace:**

Un utilisateur est associé à un ou plusieurs espaces de travail (workspace) et un espace de travail est associé à un ou plusieurs utilisateurs

**User – Notebook:**

Un utilisateur est associé à un ou aucun message et un message est associé à un utilisateur

**User – Comment:**

Un utilisateur est associé à un ou aucun commentaire et un commentaire est associé à un utilisateur

**Property – Workspace:**

Une propriété est associé à un ou plusieurs espaces de travail (Workspace) et un espace de travail est associé à une propriété

**Property – CheckListCategory:**

Une propriété est associé à un ou plusieurs check list et une checklist est associé à une propriété

**Property – File:**

Une propriété est associé à un ou plusieurs fichiers et un fichier est associé à une propriété

**CheckListCategory - CheckListTasks:**

Une check list category est associé à aucun ou plusieurs check list tasks et une checklist task est associé à une check list category

**Notebook – Workspace:**

Un notebook est associé à un workspace et un workspace est associé à aucun ou plusieurs notebook.

**Notebook – Comment:**

Un notebook est associé à aucun ou plusieurs commentaires et un commentaire est associé à un notebook

## Composants du systéme

Ces composants ci-dessous représentes des applications selon la terminologie représenté par le Framework Django :

**User:**

Composant pour gérer les utilisateurs, leur profil et leur information

**Property:**

Composant pour gérer un établissement, ses utilisateurs et différents outils

**File:**

Composant pour gérer les fichiers d’un établissement

**Workspace:**

Composant pour dialoguer et travailler à plusieurs

**LogBook:**

Composant pour gérer les taches de maintenance

**Api:**

Composant pour gérer l’API de l’application

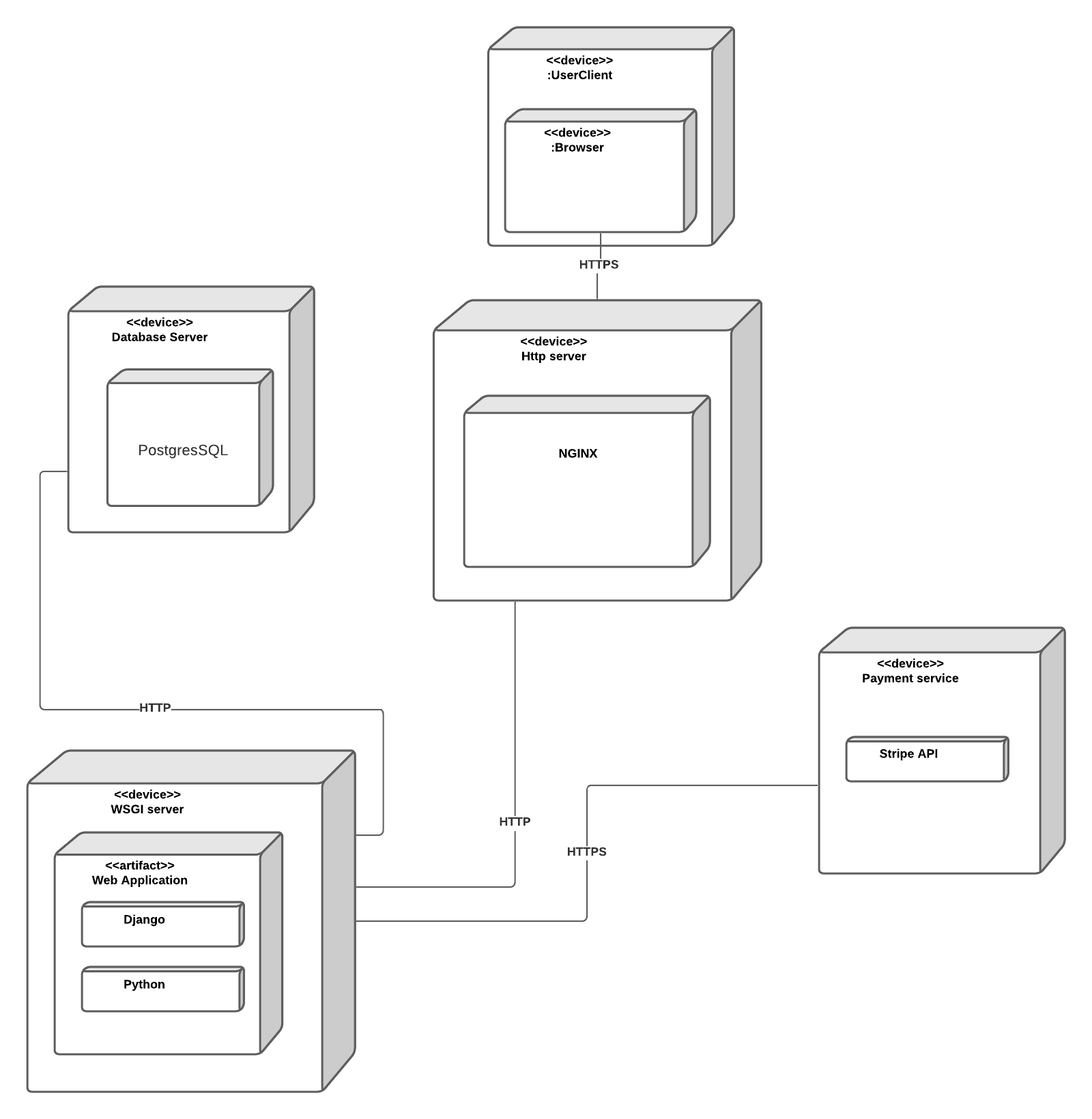
## Diagramme de déploiement

Le solution Front desk sera déployé sur un serveur physique contenant les composants et technologies suivantes :

• Les utilisateur interagiront avec un serveur **HTTP** utilisant la technologie **NGINX**

• L'application utilisera la Framework **Django** et le langage **Python** et sera déployé sur un serveur **WSGI** partageant le serveur **NGINX**

• Le troisième serveur sera la base de données relationnelles utilisant **Postgres** **SQL** qui communiquera avec le serveur **WSGI** et le serveur **http**



# Glossaire

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| **Cas d’utilisation** | Unité représentant une fonctionnalité d’un système visible de l’extérieur (exemple : se connecter) |
| **Diagramme de Cas d’utilisation** | Diagramme mettant en œuvre les relations entre les acteurs d’un système et les différents cas d’utilisation le composant. |
| **Diagramme de Packages** | Diagramme mettant en œuvre les relations entre les acteurs d’un système et les différents packages le composant. |
| **Package** | Ensemble de cas d’utilisation (exemple : interface client) |
| **Persona** | Utilisateur fictif utilisé dans la démarche de conception |
| **Scénario d’exception** | Scénario alternatif d’un cas d’utilisation |
| **Scénario nominal** | Scénario principal d’un cas d’utilisation |
| **Workflow** | Parcours d’un utilisateur, pour un cas d’utilisation donné, ici représenté sous forme d’un diagramme |