

# Mise en place d'un système de prise de rendez-vous en ligne pour le Centre Pédiatrique de Meyrin

Rapport du projet d'intégration Abuzar Chitsaz SIE 2024

## 1. Introduction

### Concept du projet SIE :

Ce rapport présente la mise en œuvre d'un ERP avec un système de prise de rendez-vous en ligne conçu pour remplacer le processus manuel existant au Centre Pédiatrique de Meyrin (1). Ce nouveau système vise à simplifier la prise de rendez-vous et la rendre accessible à tout moment, à réduire la charge de travail administrative et à améliorer l'efficacité globale des opérations de la clinique.

### L'organisation et ses besoins :

Le Centre Pédiatrique de Meyrin nécessitait une méthode plus efficace et moins coûteuse pour gérer les rendez-vous. Le système manuel existant était fastidieux, sujet à des erreurs et chronophage tant pour le personnel que pour les patients.

### Informations initiales et stratégie de mise en œuvre :

J'ai commencé par analyser le processus de prise de rendez-vous existant afin d'identifier les inefficacités et les domaines à améliorer. L'objectif était de concevoir un système automatisé, accessible partout et à toutes les heures de la journée, réduisant le temps de réservation et s'intégrant de manière transparente à l'infrastructure informatique existante de la clinique.

## 2. Analyse

### Description du problème :

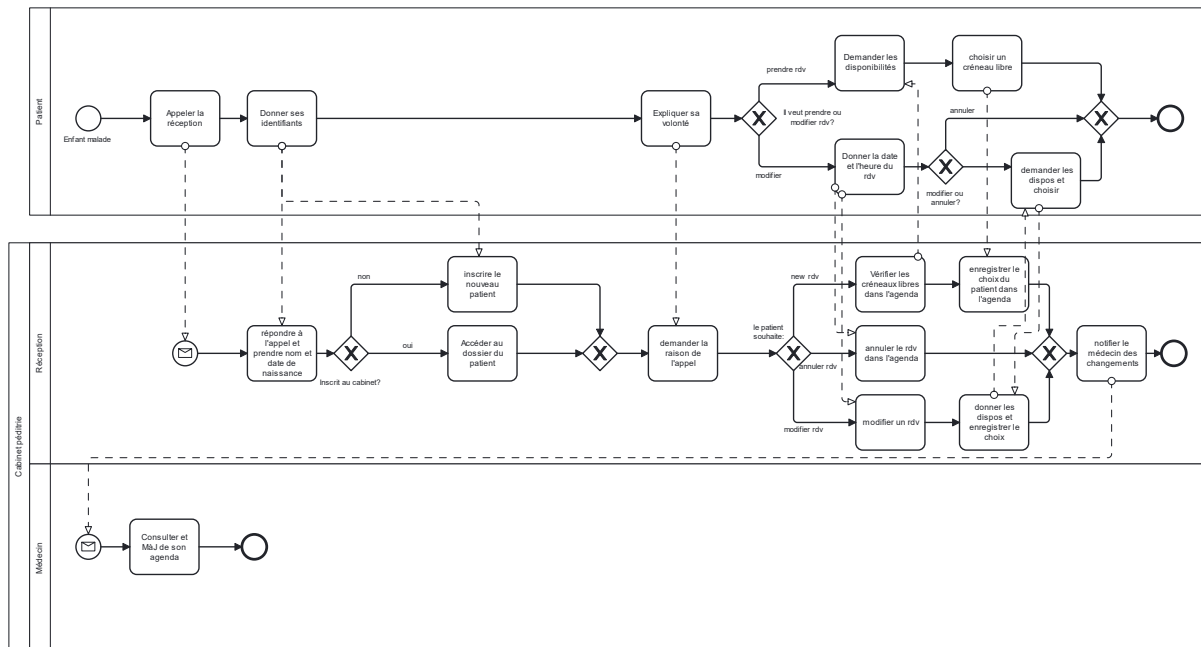
Les principaux problèmes du système actuel sont les heures limitées de prise de rendez-vous, l'obligation de la présence d'un salarié constamment à l'écoute, l'impossibilité de prendre plusieurs rendez-vous en même temps (limité à une ligne téléphonique), l'intervention humaine qui entraînait des erreurs fréquentes, le mécontentement des patients et un fardeau administratif excessif.

### Analyse du processus :

Le système actuel implique plusieurs étapes nécessitant une saisie et une confirmation manuelles, conduisant à des erreurs humaines potentielles et à des retards inutiles. Les principaux acteurs de ce processus comprennent le personnel de réception et les patients affectés par les limitations du système manuel et les professionnels médicaux.

### Modèle de processus AS-IS :

Le processus existant repose sur les appels téléphoniques et l'intervention humaine dans les échanges avec le système (l'agenda du cabinet). Cette méthode manque de flexibilité et ne prend pas en charge les mises à jour en temps réel, l'accès à distance et l'accès 24/24 7/7.



### Autres exigences :

En plus d'améliorer l'efficacité et de réduire les erreurs, le nouveau système devait soutenir la sécurité des données, être accessible sur divers appareils et permettre une gestion facile et la récupération des données de rendez-vous.

## 3. Proposition

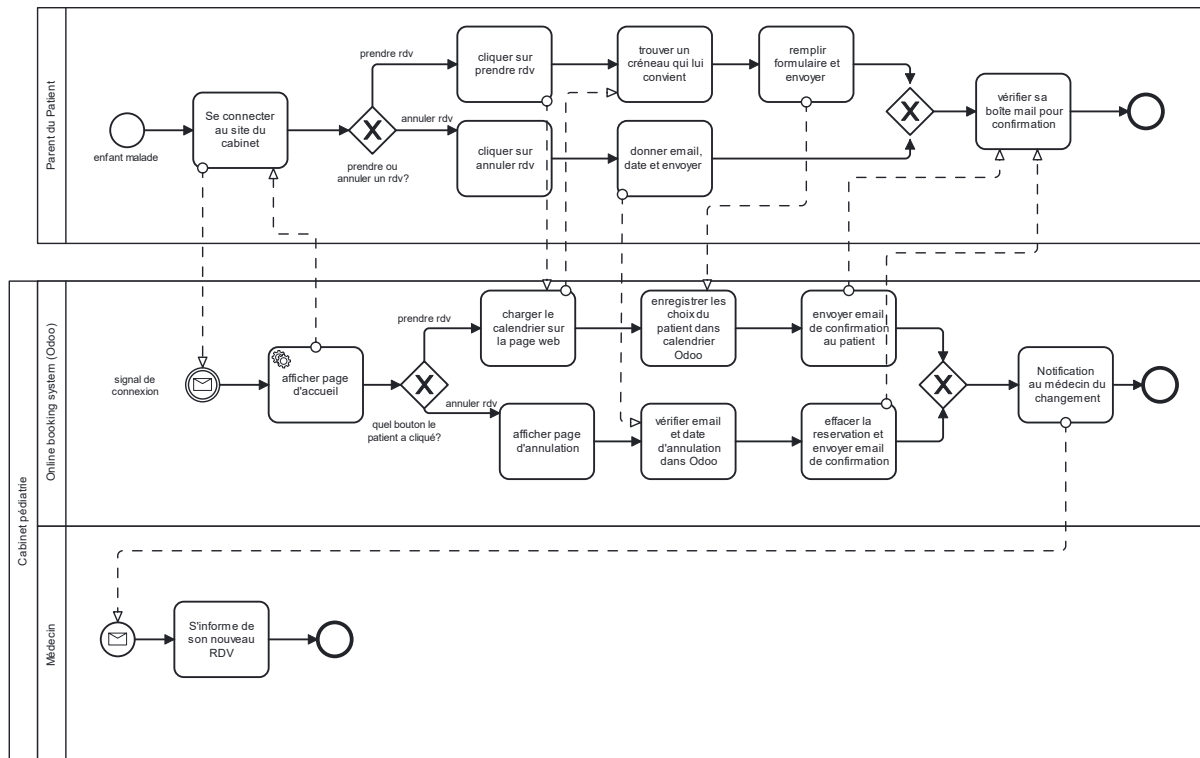
### Approche et solution :

Ma solution implique le développement d'un système de rendez-vous en ligne qui intègre Odoo ERP pour la gestion en backend et utilise Flask comme framework web pour l'interface frontend.

### Conception du processus :

Le nouveau design du processus inclut des fonctionnalités telles que la prise de rendez-vous en ligne, les confirmations automatiques par email, et la capacité d'annuler ou de reprogrammer les rendez-vous en ligne. Cela réduit la nécessité de planification basée sur les appels téléphoniques et minimise la charge de travail de la réception tout en améliorant l'accessibilité.

Le BPMN du nouveau processus (TOBE) et le suivant :



Il est important d'ajouter que dans le schéma BPMN TOBE, la fonction de notifier le médecin par mail est incluse mais ne figure pas dans le code car j'ignore leur méthode de notification mais s'il s'agit d'email, c'est exactement le même procédé et fonction que pour le client, il suffit juste d'ajouter le message et de renseigner le mail. Je n'ai aussi pas intégré le processus de modification à des fins de simplicités et de clarté du schéma car l'utilisateur peut en quelques cliques annuler son ancien rendez-vous et prendre le nouveau ce qui revient à modifier son rendez-vous.

### Implémentation :

J'ai utilisé Python pour la logique côté serveur, Flask, HTML, CSS, Bootstrap pour le frontend, et connecté le système à Odoo ERP en utilisant son API pour la gestion des données de rendez-vous et contact avec la bibliothèque OdooRPC de Python.

### Description du Processus de Code

#### 1) Configuration initiale :

- **Connexion à Odoo** : J'établis une connexion au serveur Odoo en utilisant **odoorpc.ODOO**. Cette connexion est essentielle pour interagir avec la base de données ERP où les informations sur les rendez-vous et les contacts sont stockées.
- **Configuration du serveur de messagerie** : Utilisation de Flask-Mail configuré avec un SMTP fictif pour envoyer des confirmations de rendez-vous et d'annulation par email aux utilisateurs.

#### 2) Gestion des rendez-vous :

- **API pour récupérer les événements** : Une route Flask **/api/events** utilise Odoo pour récupérer les rendez-vous existants, les convertit en fuseau horaire local avec **pytz**, et les renvoie au frontend en format JSON.

- **Soumission de rendez-vous** : La route **/submit** gère la soumission des formulaires de rendez-vous. Elle vérifie la disponibilité, crée des rendez-vous dans Odoo, et envoie des emails de confirmation.
- **Annulation de rendez-vous** : La route **/cancel-appointment** permet aux utilisateurs d'annuler des rendez-vous via l'interface utilisateur, avec la suppression correspondante dans Odoo et l'envoi d'un email de confirmation d'annulation.

### 3) Fonctions auxiliaires :

- **find\_or\_create\_contact** : Cette fonction recherche un contact existant dans Odoo ou crée un nouveau contact si nécessaire. Elle est cruciale pour lier les rendez-vous aux données des utilisateurs.
- **send\_cancellation\_email** : Fonction dédiée à l'envoi d'emails de confirmation d'annulation avec l'utilisation de Flask-Mail.

Les requêtes au serveur web et les mises à jour du calendrier se font grâce à la méthode Ajax.

Je fais appel à la bibliothèque PyTZ de Python qui permet la manipulation des fuseaux horaires afin de synchroniser l'heure de la localisation du client avec celui du serveur et cabinet.

Avec Flask je lance et contrôle les 3 pages HTML du site (avec un esthétique basique qui peuvent facilement être améliorés).

J'utilise flask\_mail configuré avec un SMTP fictif pour envoyer les emails automatiques de confirmation de rendez-vous et d'annulation aux utilisateurs.

J'ai essayé de diminuer le temps de chargement de la page du calendrier (index.html) avec flask\_caching, la fonction setTimeout et l'attribut defer dans le javascript sans grand succès car cela est dû essentiellement à la vitesse d'échange avec Odoo sur un serveur local mais je les ai laissés car dans un déploiement réel, ils peuvent optimiser le déroulement du programme et améliorer l'expérience utilisateur.

Cette configuration assure que le système est solide, évolutif et facile à maintenir.

### Déploiement :

Le système devrait être déployé sur un serveur cloud pour une haute disponibilité et fiabilité avec l'installation de Odoo et l'activation des modules de base (et gratuit) Contact et Calendar.

Dans la section du commentaire « Paramètres de connexion » Il faut adapter les paramètres de connexion à l'url du serveur, la base de données et les identifiants pour se connecter. Le port de connexion au serveur Odoo devra être remplacé (sauf s'il utilise 8069). Après avoir défini le SMTP qu'on souhaite utiliser pour envoyer les emails, le SMTP actuel qui sert au développement devra être modifié avec les nouveaux url, port, username et password dans la section « app.config ».

Si l'emplacement de l'utilisation se limite au canton de Genève, il n'y a pas besoin de changer le fuseau horaire mais si nous envisageons d'exporter ce système à d'autre destination, la fonction pytz.timezone('Europe/Paris') devra être adapté au nouveau fuseau horaire.

Tous ces changements sont à effectuer aux 30 premières lignes du code Python.

## 4. Conclusion

### Évaluation de la solution :

Le nouveau système de prise de rendez-vous améliore considérablement le processus de réservation en le rendant plus rapide, plus fiable, accessible partout et en tout temps. Il réduit les erreurs administratives et améliore la satisfaction des patients en fournissant des confirmations ou annulations de réservation immédiates. Il permet au cabinet de supprimer le poste de réceptionniste ou de consacrer cet acteur à un autre rôle plus utile dans les tâches de l'établissement.

### Rétrospective du projet :

Bien que le projet ait été un défi, notamment au niveau de la synchronisation des 3 différents fuseaux horaires du système et le temps de chargement du calendrier dû aux centaines d'événements fictifs déjà existants dans le calendrier de la démo d'Odoo, il a été couronné de succès. Le système a bien passé tous les tests utilisateurs sans faille et peut être proposé au centre pédiatrique de Meyrin avec une petite retouche sur l'esthétique du frontend pour laquelle je n'ai pas trouvé assez de temps pour l'élaborer davantage.

### Prochaines étapes :

La sécurité et l'accès pour la création de compte du prototype doit être amélioré. Dans l'état actuel n'importe qui peut créer autant de contacts basé sur des faux emails qu'il veut sans vérification et restriction ce qui pourrait saturer le système en cas de mauvaise intention.

Il est essentiel de mettre un accent sur le renforcement de la sécurité et la gestion des accès pour la création de comptes. Actuellement, le prototype permet la création de contacts sans restrictions ni vérification des emails, ce qui pourrait potentiellement saturer le système si utilisé à mauvais escient. Pour pallier ce risque, j'envisage de mettre en place un processus de vérification des adresses email afin de confirmer l'authenticité des utilisateurs avant qu'ils ne puissent effectuer des réservations. De plus, des mesures restrictives seront examinées pour limiter le nombre de comptes qu'un utilisateur peut créer et pour détecter et prévenir les comportements abusifs. Ces améliorations aideront à préserver l'intégrité du système et à garantir une utilisation sécurisée et efficace pour tous les utilisateurs.

Il serait intéressant qu'avec plus de temps et de ressource, le dossier médical des patients soit intégré dans le système Odoo afin d'associer le dossier à chaque rendez-vous et de le présenter au médecin à l'heure de la consultation.

La surveillance continue et la collecte de retours aideront à affiner davantage le système. Les améliorations futures pourraient inclure le développement d'une application mobile et l'intégration de fonctionnalités supplémentaires telles que les rappels de rendez-vous par SMS.

## 5. Annexes

### Détails d'installation et de configuration :

Suivre les étapes de la section Déploiement, en cas de problèmes contacter le développeur [abuzar.chitsaz@etu.unige.ch](mailto:abuzar.chitsaz@etu.unige.ch)

- 1) <https://cpm.officemed.ch/je-veux-un-rendez-vous-medical-cpm/>