



Année universitaire :  
2024-2025

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ

صَدَقَ اللَّهُ الْعَظِيمَ

# Remerciements

*La gratitude constitue un pilier essentiel de toute réussite. Avant toute chose, nous exprimons nos profonds remerciements à **Dieu Tout-Puissant**, pour nous avoir dotés de la volonté, de la persévérance et de la sérénité nécessaires à l'accomplissement de ce travail dans les meilleures conditions.*

*Nos remerciements vont tout particulièrement à **M. Hassan Bouzahir**, chef de filière à l'ENSA d'Agadir, pour nous avoir offert l'opportunité d'effectuer ce stage et pour son suivi attentif tout au long de cette expérience.*

*Nous souhaitons également exprimer notre profonde reconnaissance à l'ensemble de l'équipe de **l'Agence d'Automatisation IA Maroc – AHDIGITAL AUTOMATION**, qui nous a accueillis et accompagnés lors de notre stage durant le mois de juillet 2025.*

*Nos remerciements les plus sincères vont à **M. Aboubaker Beggar**, pour sa confiance, son encadrement bienveillant et ses conseils avisés, qui nous ont permis d'évoluer tant sur le plan professionnel que personnel.*

*Nous remercions également toute l'équipe de l'agence pour leur accueil chaleureux, leur esprit collaboratif et leur disponibilité, qui ont rendu cette expérience particulièrement riche et formatrice.*

*Ce stage nous a permis de mettre en pratique nos acquis théoriques dans un environnement stimulant, de développer nos compétences et de confirmer nos choix professionnels.*

*Enfin, nous adressons nos remerciements à nos familles pour leur précieux soutien moral tout au long de cette aventure.*

# Table des acronymes

<b>CSS</b>	Cascading Style Sheets
<b>ENSA</b>	École Nationale des Sciences Appliquées
<b>Flask</b>	Micro-framework Python pour le développement web
<b>Google Sheets</b>	Service de feuille de calcul utilisé comme base de données
<b>HTML</b>	HyperText Markup Language
<b>IA</b>	Intelligence Artificielle
<b>JSON</b>	JavaScript Object Notation
<b>LLM</b>	Large Language Model (ex. Gemini)
<b>n8n</b>	Outil d'automatisation open-source pour orchestrer les work-flows
<b>SQL</b>	Structured Query Language
<b>UML</b>	Unified Modeling Language
<b>API</b>	Application Programming Interface
<b>IDE</b>	Integrated Development Environment

# Resumé

Dans le cadre de notre formation en sciences des données à l'ENSA d'Agadir, nous avons effectué un stage d'observation au sein de l'entreprise AHDigital, spécialisée dans le développement de solutions numériques innovantes. L'objectif de ce stage était la conception et la mise en œuvre d'une plateforme intelligente destinée à l'automatisation de la gestion des rendez-vous médicaux au sein d'une clinique.

Au cours de cette expérience, nous avons participé à l'ensemble des étapes du projet : l'analyse des besoins, la modélisation fonctionnelle, ainsi que le développement technique à l'aide de Flask pour le frontend et de n8n couplé à Google Sheets pour le backend. Ce stage nous a permis de consolider nos compétences techniques, de nous familiariser avec les méthodes de travail agiles et d'acquérir une meilleure compréhension des enjeux liés à la digitalisation dans le domaine de la santé.

# Abstract

As part of our data science engineering studies at ENSA Agadir, we carried out an internship at AHDigital, a company specializing in innovative digital solutions. The aim of this internship was to design and implement an intelligent platform dedicated to automating the medical appointment management process in a clinic.

Throughout this experience, we were involved in all stages of the project: requirements analysis, functional modeling, and technical development using Flask for the frontend and n8n integrated with Google Sheets for the backend. This internship enabled us to strengthen our technical skills, gain practical experience with agile development methods, and acquire a deeper understanding of the challenges and opportunities of digital transformation in the healthcare sector.

# Table des figures

1.1	Logo de l'entreprise <b>AHDIGITAL</b> . . . . .	10
1.2	Image de <b>Technopark d'Agadir</b> . . . . .	11
1.3	Organigramme de l'entreprise <b>AH Digital</b> . . . . .	12
2.1	Diagramme de Gantt du projet . . . . .	17
3.1	Flux linéaire : Utilisateurs ↔ Flask ↔ n8n ↔ Google Sheets . . . . .	20
3.2	Diagramme de cas d'utilisation – Patient, Personnel médical et Personnel administratif . . . . .	21
3.3	Diagramme de classes simplifié du système . . . . .	22
3.4	Diagramme de séquence – prise de rendez-vous par un patient . . . . .	23
4.1	Capture d'écran pour la Page d'accueil . . . . .	25
4.2	Capture d'écran pour la Page de Espace Patient . . . . .	25
4.3	Capture d'écran pour la Page Page de Formulaire de prise de Rendez-vous . . . . .	26
4.4	Capture d'écran pour le Workflow de Chatbot (n8n) . . . . .	28
4.5	Widget du chatbot sur la page d'accueil . . . . .	29
4.6	Capture d'écran pour le Workflow d'inscription patient (n8n) . . . . .	30
4.7	Capture d'écran pour le Workflow de connexion patient (n8n) . . . . .	30
4.8	Capture d'écran pour le Workflow de prise de rendez-vous (n8n) . . . . .	31
4.9	Capture d'écran pour le Workflow d'annulation de rendez-vous (n8n) . . . . .	31
4.10	Capture d'écran pour le Workflow de rappel automatique par email (n8n) . . . . .	32
4.11	Capture d'écran pour le Workflow de gestion des créneaux confirmés par le staff (n8n) . . . . .	32
4.12	Exemple d'email automatique de rappel de rendez-vous généré par n8n . . . . .	33
1	Exemple d'inteface : Capture d'écran de la page d'accueil avec le chatbot assistant . . . . .	37
2	Capture d'écran pour le Workflow d'inscription patient (n8n) . . . . .	38
3	Capture d'écran pour le Workflow de prise de rendez-vous (n8n) . . . . .	38
4	Capture d'écran Exemple de structure de la feuille Google Sheets . . . . .	40

# Table des matières

<b>Remerciements</b>	<b>3</b>
<b>Table des acronymes</b>	<b>4</b>
<b>Table des figures</b>	<b>6</b>
<b>Introduction</b>	<b>9</b>
<b>1 Présentation générale du stage</b>	<b>10</b>
1.1 Introduction . . . . .	10
1.1.1 Objectifs du stage . . . . .	10
1.2 Présentation de l'entreprise . . . . .	10
1.3 Localisation et zone d'activité . . . . .	11
1.4 Environnement de travail . . . . .	11
<b>2 Contexte général du projet</b>	<b>13</b>
2.1 Présentation du projet . . . . .	13
2.1.1 Titre du projet . . . . .	13
2.1.2 Contexte de la problématique . . . . .	13
2.1.3 Solution proposée et objectifs . . . . .	13
2.1.4 Parties prenantes Concernées . . . . .	14
2.2 Catalogue des Fonctionnalités et des Utilisateurs . . . . .	14
2.2.1 Fonctionnalités attendues . . . . .	14
2.2.2 Profils Utilisateurs . . . . .	15
2.3 Méthodologie de développement . . . . .	15
2.3.1 Choix de l'approche . . . . .	16
2.3.2 Étapes de réalisation . . . . .	16
2.3.3 Planning du projet . . . . .	16
2.3.4 Outils et technologies utilisés . . . . .	17
2.3.5 Résumé de la méthodologie . . . . .	17
2.4 Conclusion de la partie du chapitre 2 . . . . .	17
<b>3 Conception et architecture du projet</b>	<b>19</b>
3.1 Introduction . . . . .	19
3.2 Architecture générale du système . . . . .	19
3.3 Conception fonctionnelle . . . . .	20
3.3.1 Acteurs du système . . . . .	20
3.3.2 Cas d'utilisation . . . . .	20
3.3.3 Analyse . . . . .	21

---

3.4	Conception technique . . . . .	22
3.4.1	Modèle de données (diagramme de classes) . . . . .	22
3.4.2	Scénario type : prise de rendez-vous (diagramme de séquence) . . .	22
<b>4</b>	<b>Mise en œuvre technique du projet</b>	<b>24</b>
4.1	Développement du frontend (Flask + HTML/CSS) . . . . .	24
4.1.1	Pages principales . . . . .	24
4.1.2	Aspects techniques . . . . .	24
4.1.3	Captures d'écran . . . . .	24
4.2	Développement du backend (n8n + Google Sheets) . . . . .	26
4.3	Back-end du Chatbot (synthèse opérationnelle) . . . . .	26
4.3.1	Inscription des utilisateurs . . . . .	29
4.3.2	Authentification (connexion) . . . . .	30
4.3.3	Prise de rendez-vous . . . . .	30
4.3.4	Annulation de rendez-vous . . . . .	31
4.3.5	Notifications et rappels automatiques . . . . .	31
4.3.6	Gestion administrative des créneaux . . . . .	32
4.4	Tests et validation . . . . .	32
4.5	Conclusion Chapitre 4 . . . . .	34
<b>5</b>	<b>Conclusion et perspectives</b>	<b>35</b>
5.1	Synthèse . . . . .	35
5.2	Perspectives . . . . .	35
5.3	Conclusion générale . . . . .	36
	<b>Annexes</b>	<b>37</b>
	<b>Webographie</b>	<b>41</b>



# Introduction

Dans le cadre de ma formation en première année du cycle d'ingénieur en Sciences des Données, Big Data et Intelligence Artificielle à l'École Nationale des Sciences Appliquées d'Agadir, j'ai eu l'opportunité d'effectuer un stage d'application au sein de l'entreprise AHDigital, du 1er juillet au 31 juillet 2025.

Ce stage a constitué ma première immersion dans le monde professionnel de l'intelligence artificielle, un domaine en pleine évolution et au cœur de ma spécialisation. Il m'a permis de découvrir l'environnement de travail en entreprise, de mieux comprendre les enjeux actuels liés à l'IA et, surtout, de m'initier concrètement au fonctionnement des agents intelligents, qui occupent une place essentielle dans les systèmes modernes.

L'objectif principal de ce stage était la réalisation complète d'un projet de développement d'une plateforme intelligente de gestion et de prise de rendez-vous pour une clinique, depuis la phase de conception jusqu'à la mise en œuvre technique.

Les livrables finaux ont été produits sous la forme de fichiers HTML et CSS pour la partie frontend, et de fichiers JSON représentant les workflows automatisés développés avec n8n et intégrés à Google Sheets pour la partie backend.

Bien que ce travail a été encadré et supervisé par l'équipe technique d'AHDigital, qui a fourni des orientations stratégiques et des retours ponctuels. Ce stage m'a permis de mettre en pratique mes compétences techniques, de renforcer ma capacité à gérer un projet complet et d'approfondir mes connaissances dans la création de solutions intelligentes appliquées au domaine de la santé.

Ce rapport présente, dans un premier temps, l'entreprise AHDigital, son domaine d'activité et son organisation. Il détaille ensuite les missions qui m'ont été confiées, les connaissances acquises, ainsi que les outils et approches utilisés. Enfin, un bilan global vient clore ce document, en mettant en lumière les apports de ce stage sur les plans personnel et professionnel.

# 1. Présentation générale du stage

## 1.1 Introduction

### 1.1.1 Objectifs du stage

#### Objectif principal :

Mettre en pratique mes compétences techniques en développement web et en automatisation pour réaliser une plateforme intelligente de gestion et de prise de rendez-vous pour une clinique.

#### Objectifs spécifiques :

- Concevoir l'architecture de la plateforme (frontend + backend).
- Développer les interfaces en HTML/CSS.
- Créer et configurer les workflows d'automatisation avec n8n.
- Intégrer le stockage et la gestion des données via Google Sheets.
- Tester et valider les fonctionnalités mises en place.

## 1.2 Présentation de l'entreprise



FIGURE 1.1 – Logo de l'entreprise **AHDIGITAL**

Fondée en mai 2024 à Agadir, **AH Digital** est une entreprise marocaine innovante, spécialisée dans le développement de solutions digitales, de sites web et dans l'automatisation via des agents intelligents (*agents AI*). Dès ses débuts, elle a su allier expertise

---

technique et esprit d’innovation pour offrir des services numériques de haute qualité, aussi bien aux entreprises locales qu’internationales.

Au fil des années, l’entreprise a élargi son champ d’expertise pour mieux accompagner la transformation digitale des entreprises. Aujourd’hui, **AH Digital** se positionne comme une startup dynamique basée au *Technoparc d’Agadir*, en pleine croissance, et reconnue comme un acteur régional majeur dans le domaine du développement web et des solutions SaaS.

Parmi ses produits phares figure la plateforme SaaS « AH DIGITAL », une solution tout-en-un spécialement conçue pour les agences marketing. Cette plateforme intègre des fonctionnalités essentielles telles que :

- La gestion de la relation client (CRM) ;
- La création de funnels marketing ;
- La gestion de campagnes multicanales ;
- Un calendrier intégré ;
- Un constructeur de landing pages personnalisables.

Grâce à cette solution flexible, modulaire et évolutive, **AH Digital** vise à fournir aux agences marketing un outil puissant pour gérer leurs opérations, optimiser leur efficacité et renforcer leur présence numérique.

## 1.3 Localisation et zone d’activité



FIGURE 1.2 – Image de **Technoparc d’Agadir**

L’entreprise **AH Digital** est implantée au *Technoparc d’Agadir*, un espace dédié aux startups technologiques offrant un cadre de travail moderne, collaboratif et propice à l’innovation. Située dans un bureau lumineux et bien aménagé (**Bureau TA113 – Premier étage**), l’équipe évolue dans un environnement stimulant, favorisant la créativité, l’échange et la concentration.

## 1.4 Environnement de travail

L’environnement de travail chez AH Digital repose sur une structure hiérarchique claire et une forte culture de collaboration. L’organisation interne comprend plusieurs pôles :

- **Direction générale (CEO)** : définit la stratégie globale de l’entreprise,
- **Direction technique (CTO)** : pilote la vision technologique et supervise l’équipe de développement,

- **Équipe de développement** : conçoit, implémente et teste les solutions logicielles,
- **Équipe d'automatisation** : développe des outils intelligents d'optimisation des processus,
- **Équipe marketing** : conçoit et met en œuvre les campagnes de communication,
- **Direction des opérations (COO)** : assure la coordination quotidienne et la bonne exécution des projets,
- **Équipes support et consulting** : accompagnent les clients et assurent une assistance technique personnalisée.

Cette organisation assure une synergie entre les différents pôles, favorisant une culture de l'innovation et de la qualité dans l'ensemble des projets.

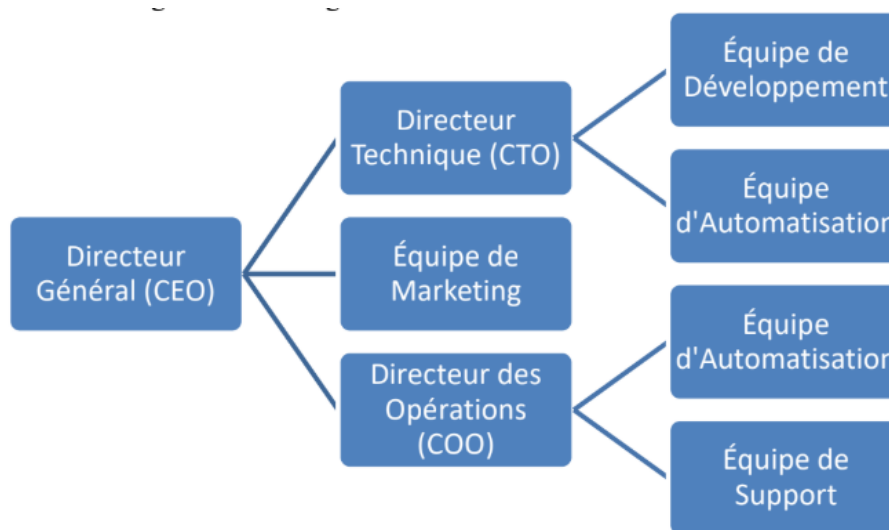


FIGURE 1.3 – Organigramme de l'entreprise **AH Digital**

## Fiche signalétique de l'entreprise

Nom de l'entreprise	AH Digital
Date de création	Mai 2024
Statut juridique	SARL
Adresse	Technopark Agadir, Bureau TA113, Route Nationale N°105
Secteurs d'activité	Développement web, Automatisation intelligente, Formation no-code/IA
Email	automation@ahdigital.tech
Site web	<a href="http://www.ahdigital.tech">http://www.ahdigital.tech</a>
Téléphone	+212 636-308953

Tableau 1 fiche signalétique de l'organisation

## 2. Contexte général du projet

### 2.1 Présentation du projet

#### 2.1.1 Titre du projet

Développement d'un **Système Intelligent d'Orientation et de Prise de Rendez-vous** pour une **Clinique intelligente**.

#### 2.1.2 Contexte de la problématique

Dans le fonctionnement actuel de la clinique, la prise de rendez-vous est effectuée de manière entièrement manuelle. Cette méthode, bien que simple, entraîne plusieurs difficultés :

- Des délais importants dans le traitement des demandes.
- Des erreurs fréquentes dans la planification des consultations.
- Une mauvaise gestion des flux de patients, notamment lors des périodes de forte affluence.
- Un manque de traçabilité et de centralisation des informations.

Face à ces limitations, il est devenu nécessaire de moderniser et d'automatiser l'ensemble du parcours patient afin de réduire les délais, améliorer la précision des plannings et offrir une meilleure expérience aux usagers.

#### 2.1.3 Solution proposée et objectifs

Le projet vise à concevoir et déployer une **plateforme numérique intelligente** permettant d'automatiser la prise de rendez-vous et la gestion des dossiers patients. Cette solution repose sur :

- Un **frontend** développé en **HTML/CSS** pour offrir une interface utilisateur claire et intuitive.
- Un **backend** automatisé avec **n8n**, chargé de traiter les demandes, vérifier les disponibilités et gérer les actions automatiques.
- Un système de stockage et de gestion des données centralisé dans **Google Sheets**, facilitant le suivi et la consultation des informations.

Les objectifs principaux sont :

- Permettre au patient de soumettre ses informations personnelles et son besoin médical.
- Analyser automatiquement la demande afin de proposer le service ou le médecin adapté.
- Vérifier les disponibilités en temps réel dans le planning médical.
- Réserver automatiquement un créneau optimal.

---

### 2.1.4 Parties prenantes Concernées

Le projet implique plusieurs parties prenantes clés, chacune ayant un rôle spécifique dans la réussite du développement et du déploiement de la plateforme :

**Client clinique** Commanditaire du projet, exprime les besoins, valide les livrables et assure l'intégration au sein de la structure médicale.

**Agence d'automatisation et de développement (nous)** Responsable de la conception, du développement, des tests, de l'intégration et du support technique.

**Personnel médical et administratif du clinique** Utilisateurs finaux de la plateforme (médecins, secrétaires, infirmiers), contribuent à la définition des besoins et valident l'ergonomie.

**Patients** Utilisateurs indirects, bénéficient de l'orientation intelligente, de la prise de rendez-vous, du suivi et de l'accès à leurs informations.

## 2.2 Catalogue des Fonctionnalités et des Utilisateurs

### 2.2.1 Fonctionnalités attendues

#### Module Patient

- Création de compte ou connexion.
- Formulaire de prise de rendez-vous avec description des symptômes.
- Orientation automatique vers le service et le médecin adapté.
- Affichage des infos du médecin associé lors de rdv.
- Historique des rendez-vous.
- Annulation des rendez-vous à venir.

#### Module Médecin

- Création de compte ou connexion .
- Gestion des disponibilités (ajout, modification, suppression).
- Consultation des dossiers patients.
- Ajout de traitements et de remarques pour le suivi.

#### Module Administrateur

- Connexion sécurisée.
- Statistiques sur le nombre de patients, médecins et personnels administratifs.
- Gestion des comptes (création, modification, suppression).

#### Module Assistant IA (Chatbot)

##### Objectif

- Accueillir l'utilisateur et l'orienter vers l'action adéquate : prendre un rendez-vous, voir/annuler ses rendez-vous, se connecter/s'inscrire, accéder aux espaces Médecin/Admin, consulter les informations & contacts.
- Répondre brièvement à des questions générales (FAQ) et fournir des conseils de santé non-diagnostiques, avec prudence.

- 
- Réduire le nombre de clics et le taux d'abandon.

### Portée (V1)

- Disponible sur la page d'accueil (extension possible à d'autres pages ensuite).
- Public : visiteurs, patients, médecins et administrateurs (connectés ou non).

### Limites

- Aucun diagnostic médical. En cas de symptômes graves, afficher un message d'alerte et orienter vers les urgences/contacts.
- Aucune collecte de données de santé dans le chatbot.

### Déclencheurs & Interface

- Bouton flottant « » en bas à droite, ouverture d'un panneau de conversation.
- À l'ouverture : message de bienvenue + raccourcis « Espace Patient », « Espace Médecin », « Espace Admin », « Infos & Contact ».

### Critères d'acceptation (V1)

1. Saisie « je veux un rdv » → bouton « Prendre un rendez-vous » qui ouvre `/patientform`.
2. Saisie « voir/annuler mes rendez-vous » → bouton « Voir/annuler mes rendez-vous » vers `/dashboard`.
3. Saisie « se connecter / s'inscrire (patient) » → bouton « S'inscrire / Se connecter » vers `/login` et `/register`.
4. Saisie « médecin » → boutons « Créer un compte » (`/registermed`) et « Se connecter » (`/loginmedecin`).
5. Saisie « admin/staff » → boutons « Créer un compte » (`/registeradmin`) et « Se connecter » (`/loginadmin`).
6. Saisie « infos / contact / horaires / adresse » → bouton « Infos & Contact ».

## 2.2.2 Profils Utilisateurs

- **Patient** : accès au formulaire de rendez-vous et à l'historique.
- **Médecin** : gestion des disponibilités et des dossiers patients.
- **Administrateur** : supervision et gestion des comptes et statistiques.

## 2.3 Méthodologie de développement

Pour mener à bien le développement de la plateforme intelligente de gestion et de prise de rendez-vous, une approche méthodologique claire a été adoptée, combinant organisation structurée et exécution itérative.

---

### 2.3.1 Choix de l'approche

Le projet a été réalisé sans mise en place formelle d'un cadre Agile complet, mais en s'inspirant des principes de développement itératif. Cette méthode a permis de progresser par étapes, en validant progressivement chaque composant avant de passer au suivant. Les choix technologiques ont été guidés par les objectifs du cahier des charges et par la recherche d'une solution simple, efficace et facilement déployable.

### 2.3.2 Étapes de réalisation

Le développement s'est articulé autour des étapes suivantes :

1. **Analyse des besoins et rédaction du cahier des charges** : identification des fonctionnalités essentielles et des profils utilisateurs.
2. **Conception fonctionnelle** : définition de l'architecture générale et des interactions entre les modules (frontend, backend, stockage).
3. **Développement du frontend** :
  - Création des interfaces en **HTML/CSS**, avec une attention portée à l'ergonomie et à la simplicité d'utilisation.
  - Mise en place des formulaires pour la prise de rendez-vous et la gestion des informations patient.
4. **Mise en place du backend** :
  - Utilisation de **n8n** pour automatiser les workflows : traitement des demandes, vérification des disponibilités et envoi de notifications.
  - Intégration avec **Google Sheets** pour le stockage et la gestion centralisée des données.
5. **Tests et validations** :
  - Vérification du bon fonctionnement de chaque fonctionnalité.
  - Simulation de scénarios réels pour évaluer la fiabilité et la réactivité du système.
6. **Ajustements et améliorations** :
  - Correction des anomalies détectées lors des tests.
  - Optimisation des workflows et amélioration de l'interface utilisateur.

### 2.3.3 Planning du projet

Afin de mieux organiser les différentes étapes de réalisation, un planning a été défini et représenté sous forme d'un diagramme de Gantt (Figure ??). Celui-ci illustre la répartition des tâches tout au long de la période du stage (01 juillet au 31 juillet 2025).



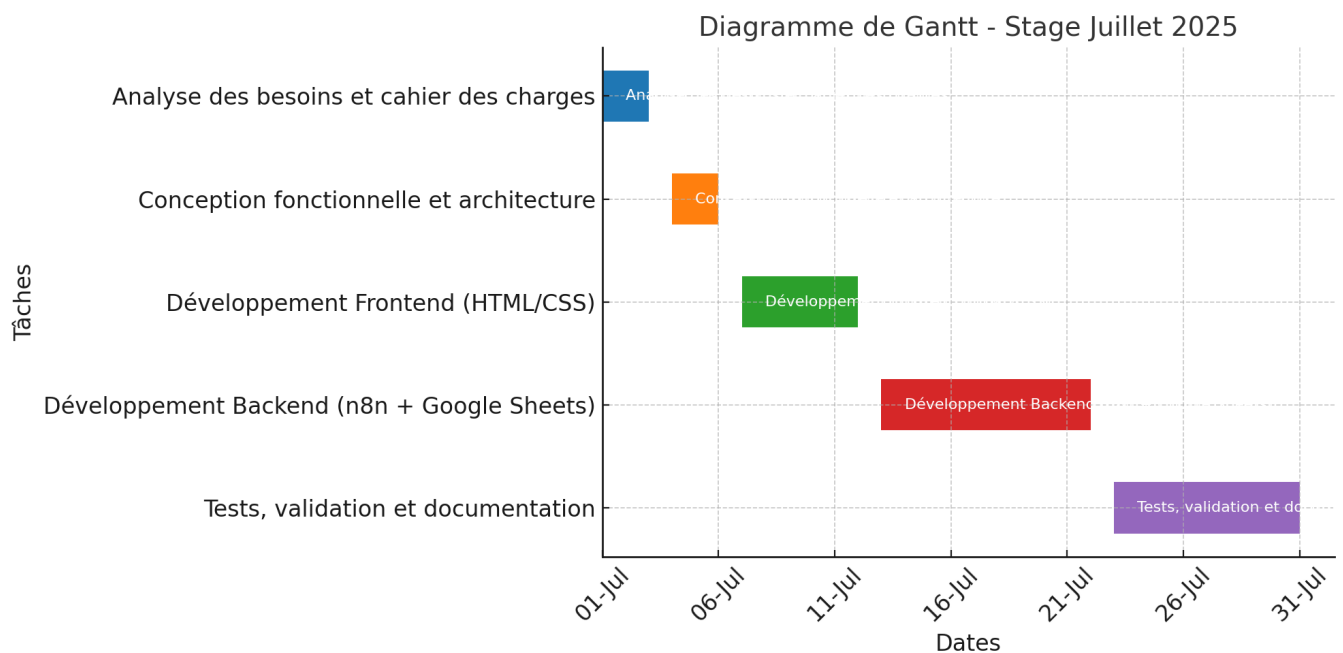


FIGURE 2.1 – Diagramme de Gantt du projet

### 2.3.4 Outils et technologies utilisés

- **Frontend** : HTML, CSS.
- **Backend et automatisation** : n8n.
- **Base de données / stockage** : Google Sheets.
- **Communication et tests** : Navigateur web pour validation des interfaces et vérification des données.

### 2.3.5 Résumé de la méthodologie

En résumé, la méthodologie adoptée a permis de structurer efficacement le projet, en suivant une logique claire : analyse, conception, développement, intégration puis validation. Cette approche progressive et itérative a assuré la conformité du produit final aux objectifs initiaux, tout en respectant les contraintes de temps et de ressources fixées par le stage.

## 2.4 Conclusion de la partie du chapitre 2

Dans ce chapitre, nous avons présenté le cadre général du projet en mettant en évidence la problématique initiale rencontrée par la clinique, à savoir la gestion manuelle des rendez-vous et ses limites. Nous avons ensuite défini la solution proposée, à travers le développement d'une plateforme numérique intégrant un frontend en **HTML/CSS**, un backend automatisé via **n8n**, et un stockage centralisé sur **Google Sheets**. Les fonctionnalités clés et les profils utilisateurs ont été détaillés, afin de mieux cerner les besoins auxquels répond la solution.

---

Enfin, une méthodologie de développement claire a été exposée, incluant le choix de l'approche, les étapes de réalisation, le planning du projet ainsi que les outils technologiques utilisés. Cette démarche structurée constitue la base solide sur laquelle s'appuie la suite du travail, qui portera sur la conception et l'architecture technique de la plateforme.

## 3. Conception et architecture du projet

### 3.1 Introduction

La conception constitue une étape essentielle dans le cycle de développement logiciel. Elle permet de traduire les besoins fonctionnels identifiés lors des phases d'analyse en une architecture technique claire et structurée. L'objectif est de définir l'organisation générale du système, les interactions entre ses différents composants ainsi que les flux d'information qui circulent entre eux.

Dans le cadre de ce projet, la conception a porté sur la mise en place d'une plateforme intelligente de gestion des rendez-vous pour une clinique. Cette étape a permis de déterminer l'architecture globale de la solution, de modéliser les interactions des différents profils utilisateurs (patients, personnel médical et administratif, administrateur), et de spécifier les choix techniques retenus pour le frontend, le backend et le stockage des données.

Ainsi, ce chapitre présente l'architecture générale du système, la conception fonctionnelle à travers les cas d'utilisation, puis la conception technique détaillant les modules principaux de la plateforme.

### 3.2 Architecture générale du système

L'architecture générale de la plateforme repose sur une organisation en trois couches principales :

- **Frontend (Interface utilisateur)** : développé avec le framework **Flask (Python)**, qui génère et sert les pages **HTML/CSS**. Cette couche constitue le point d'accès principal des utilisateurs (patients, personnel médical et administratif) à la plateforme. Elle comprend notamment la page d'accueil, l'espace patient, le formulaire de rendez-vous et le tableau de bord.
- **Backend (Automatisation et logique métier)** : assuré par la plateforme **n8n**, il orchestre les processus métiers : enregistrement des patients, création et annulation de rendez-vous, notifications automatiques et génération de factures. Flask communique avec les workflows n8n via des appels API (webhooks), jouant ainsi un rôle d'intermédiaire entre l'interface utilisateur et l'automatisation des traitements.
- **Stockage des données** : basé sur **Google Sheets**, utilisé comme base de données centralisée pour conserver les informations relatives aux patients, aux rendez-vous et aux services de la clinique.

Le schéma de l'architecture globale illustre les interactions entre ces composants. Un patient saisit ses informations via l'interface frontend servie par Flask, qui les transmet ensuite au backend n8n. Les données sont traitées et stockées dans Google Sheets, puis

---

une réponse est générée et renvoyée à l'utilisateur sous forme de confirmation ou de notification.

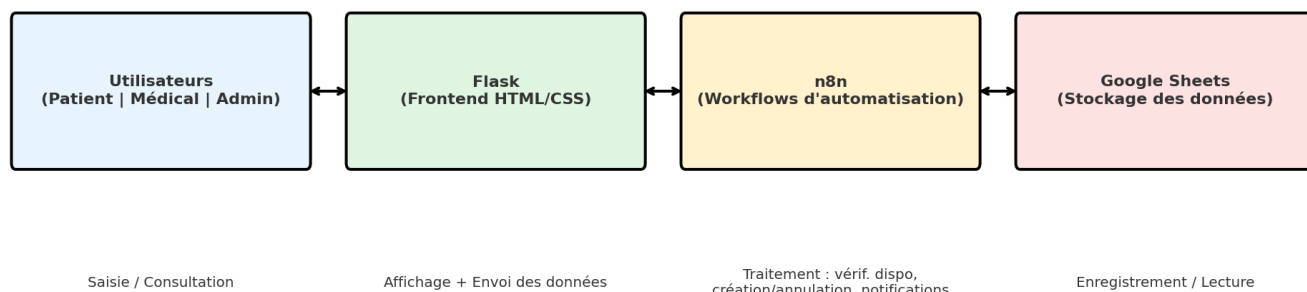


FIGURE 3.1 – Flux linéaire : Utilisateurs ↔ Flask ↔ n8n ↔ Google Sheets

## 3.3 Conception fonctionnelle

La conception fonctionnelle vise à identifier et à décrire les principaux acteurs du système ainsi que les interactions qu'ils entretiennent avec la plateforme. Cette étape permet de définir clairement les fonctionnalités offertes par le système et de préciser le rôle de chaque utilisateur.

### 3.3.1 Acteurs du système

Trois profils d'utilisateurs interagissent avec la plateforme :

- **Patient** : utilise la plateforme pour créer un compte, s'orienter vers le bon service, réserver, modifier ou annuler un rendez-vous, et consulter son tableau de bord.
- **Personnel médical** : gère les consultations (validation et ajout de notes), consulte son planning, met à jour ses disponibilités et accède aux archives patients.
- **Personnel administratif** : supervise les rendez-vous, consulte les statistiques globales et gère les comptes des patients et des médecins.

### 3.3.2 Cas d'utilisation

Les interactions entre les acteurs et le système sont représentées sous forme de cas d'utilisation. Chaque acteur dispose d'un ensemble d'actions spécifiques qui traduisent ses besoins fonctionnels.

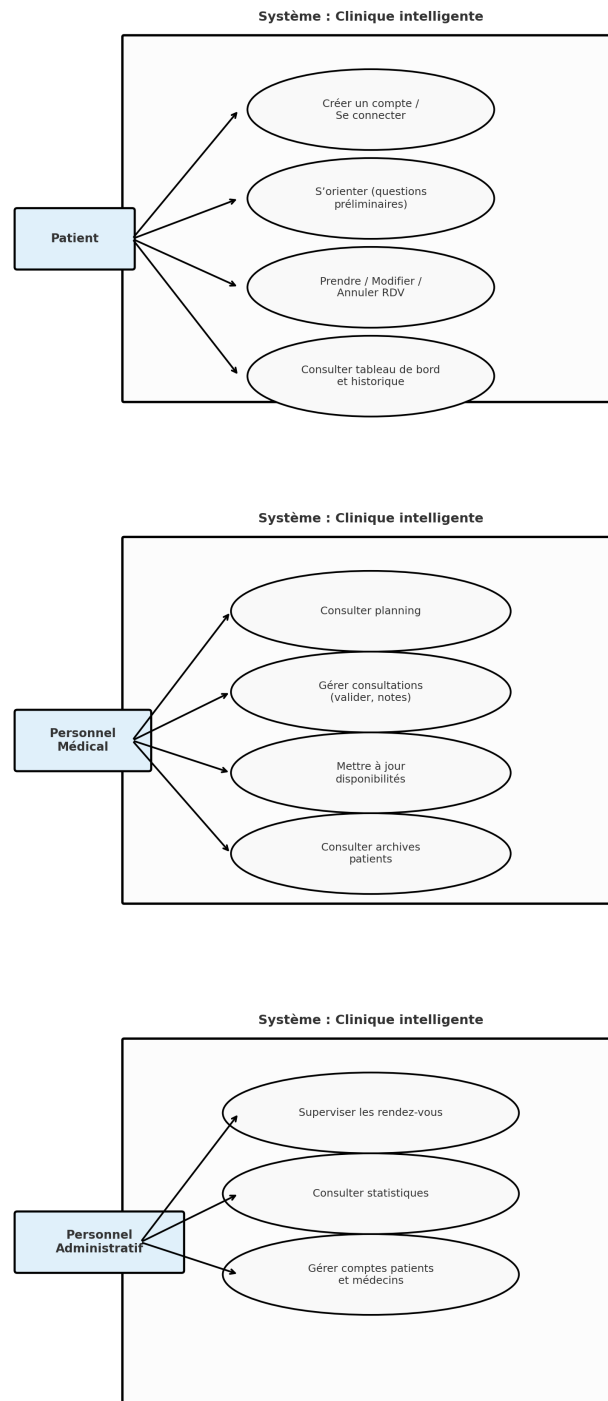


FIGURE 3.2 – Diagramme de cas d'utilisation – Patient, Personnel médical et Personnel administratif

### 3.3.3 Analyse

Ces cas d'utilisation mettent en évidence les fonctionnalités essentielles du système et définissent les responsabilités de chaque acteur. Ils constituent une base solide pour la conception technique en assurant une couverture complète des besoins fonctionnels de la

---

clinique intelligente.

## 3.4 Conception technique

### 3.4.1 Modèle de données (diagramme de classes)

Le modèle de données s'appuie sur cinq entités principales (*Patient*, *Medecin*, *Secrétaire*, *CreneauDisponible*, *RendezVous*). Chaque rendez-vous est associé à un patient et à un médecin, et est réservé à partir d'un créneau disponible identifié par l'`id_medecin` et la date/heure.

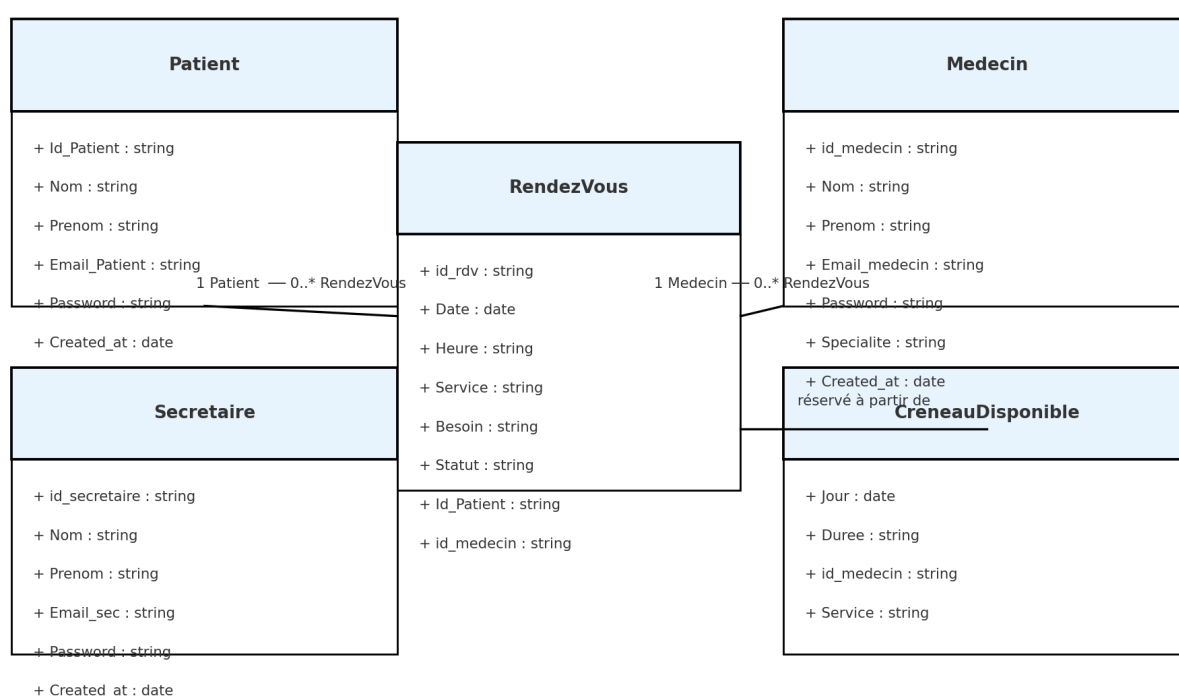


FIGURE 3.3 – Diagramme de classes simplifié du système

### 3.4.2 Scénario type : prise de rendez-vous (diagramme de séquence)

La Figure ?? détaille le flux d'interaction lors d'une prise de rendez-vous par un patient : le formulaire est soumis via **Flask**, transmis à **n8n** (webhook), qui interroge **Google Sheets** pour vérifier la disponibilité, écrit le rendez-vous confirmé, puis renvoie une réponse JSON affichée à l'utilisateur.

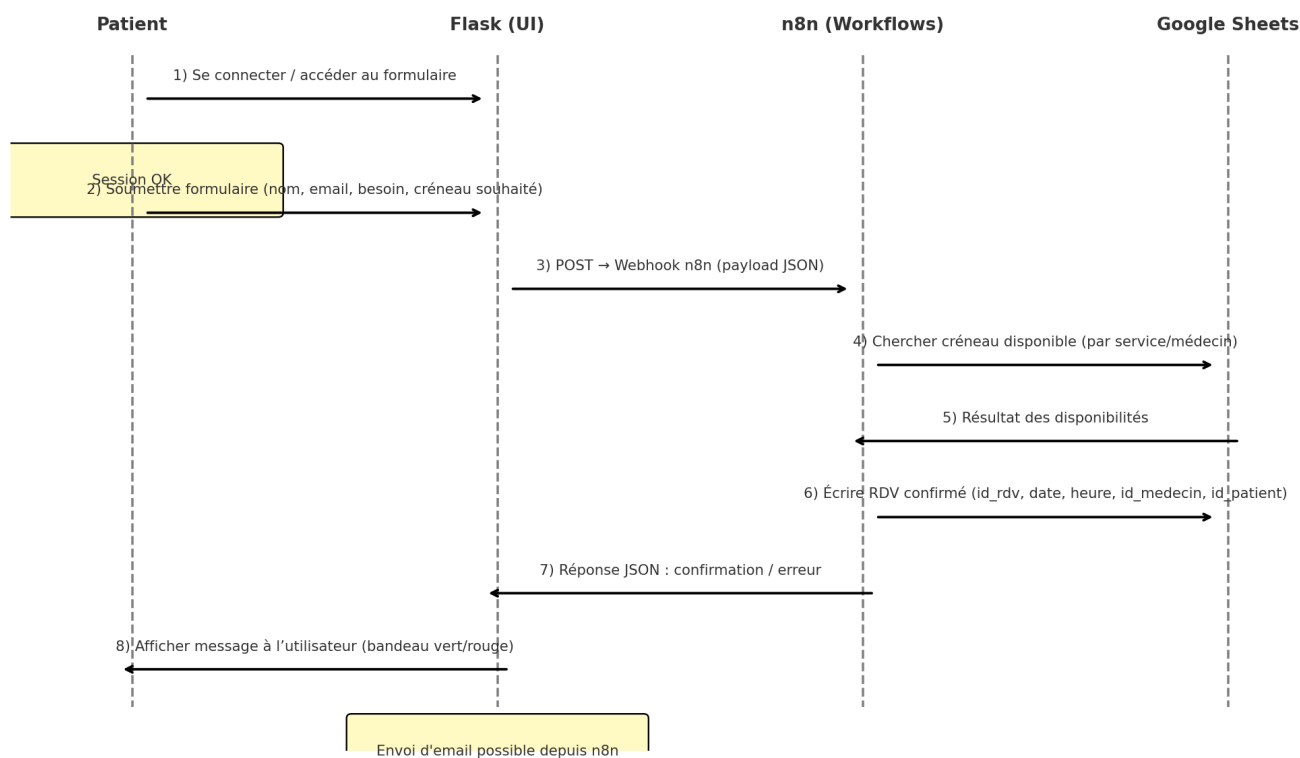


FIGURE 3.4 – Diagramme de séquence – prise de rendez-vous par un patient

## 4. Mise en œuvre technique du projet

### 4.1 Développement du frontend (Flask + HTML/CSS)

Le développement de la partie frontend a été assuré à l'aide du framework **Flask**, qui permet de générer des pages dynamiques en Python, tout en intégrant du code **HTML/CSS** pour le rendu visuel. L'objectif était de concevoir une interface simple, ergonomique et cohérente pour l'ensemble des utilisateurs de la plateforme.

#### 4.1.1 Pages principales

- Le frontend comprend plusieurs pages qui structurent la navigation des utilisateurs :
- **Page d'accueil** : propose l'accès aux différents espaces (Patient, Médecin, Administratif).
  - **Page de connexion (Login)** : permet à un patient déjà inscrit de s'authentifier.
  - **Page d'inscription (Register)** : formulaire d'enregistrement pour créer un nouveau compte patient.
  - **Espace patient** : interface regroupant deux accès : la prise de rendez-vous et le tableau de bord.
  - **Formulaire de rendez-vous** : permet de saisir les informations personnelles et le besoin médical.
  - **Dashboard patient** : tableau de bord affichant les rendez-vous confirmés, avec possibilité d'annulation.

#### 4.1.2 Aspects techniques

Chaque page est construite en HTML pour la structure et en CSS pour la mise en forme, tandis que Flask assure la gestion des routes et la liaison avec les workflows n8n via des requêtes HTTP. Ainsi, les formulaires envoyés depuis le frontend sont redirigés vers les webhooks n8n, et les réponses JSON retournées sont affichées sous forme de messages de confirmation (barre verte) ou d'erreur (barre rouge).

#### 4.1.3 Captures d'écran

Les figures suivantes présentent quelques interfaces clés de la plateforme :

- Page d'accueil .
- Page de connexion et d'inscription.
- Formulaire patient.





FIGURE 4.1 – Capture d'écran pour la Page d'accueil

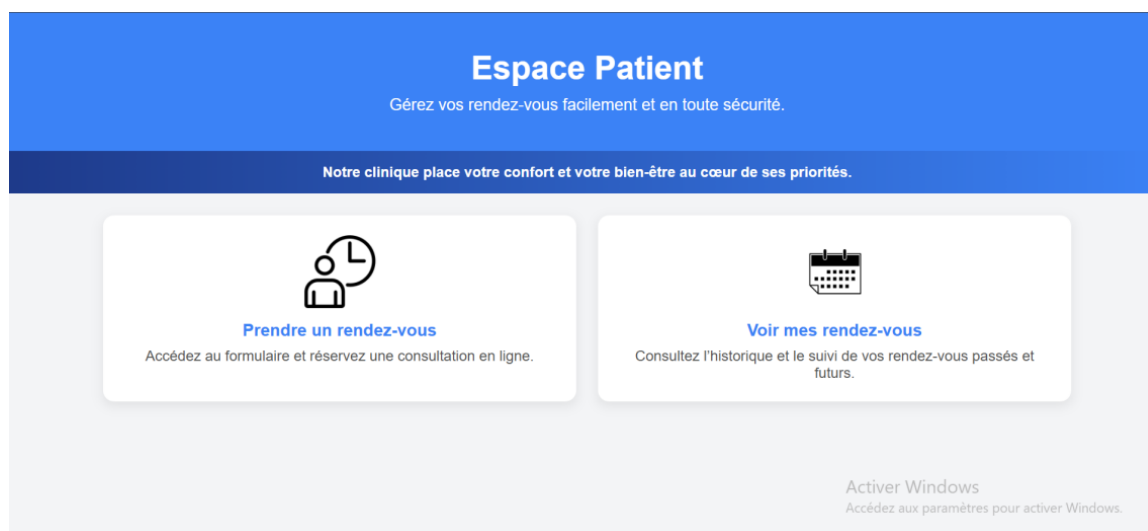


FIGURE 4.2 – Capture d'écran pour la Page de Espace Patient

Merci de compléter vos informations pour réserver votre consultation.

Réservez Maintenant Votre Rendez-vous

NomPrénom

DupontMarie

CINÂge

AB12345628

Email (Gmail)

exemple@gmail.com

Type d'assurance

-- Choisir --

Besoin

Décrivez brièvement votre besoin médical...

Envoyer

Activer Windows  
Accédez aux paramètres pour activer Windows.

Activer Windows  
Accédez aux paramètres pour activer Windows.

FIGURE 4.3 – Capture d’écran pour la Page Page de Formulaire de prise de Rendez-vous

## 4.2 Développement du backend (n8n + Google Sheets)

La logique métier et l’automatisation des processus ont été réalisées grâce à la plateforme **n8n**, qui permet de concevoir des workflows visuels interconnectés avec **Google Sheets** servant de base de données centralisée. Chaque fonctionnalité clé de la plateforme (inscription, connexion, gestion des rendez-vous, notifications) a été traduite par un workflow dédié.

## 4.3 Back-end du Chatbot (synthèse opérationnelle)

### Architecture minimale

- **Front** (Flask inchangé) : `chat.js` envoie la question au Webhook n8n et rend la réponse.
- **Orchestration** : `n8n` → Webhook (POST) → LLM (Gemini) → Parser structuré → Réponse.

- 
- **Environnements** : Test /webhook-test/chatbot-router (exécution manuelle) et Prod /webhook/chatbot-router (workflow activé).

## Contrat d'API (contrôle d'interface)

### Requête

```
POST /webhook(-test)/chatbot-router
Content-Type: application/json
{ "question": "je veux un rdv" }
```

### Réponse

```
{ "text": "Pour réserver un rendez-vous, suivez le lien ci-dessous.",
  "choices": [ { "label": "Prendre un rendez-vous", "intent": "patient_rdv" } ] }
```

*Remarque* : selon la configuration du parser n8n, la charge utile peut être encapsulée dans output.. Le front accepte les deux.

## Règles métier dans le prompt (résumé)

- Sortie **100% JSON** : clés `text` (FR,  $\leq 240$  car.) et `choices[]` (0–4 éléments {label, intent}).
- Routage direct : *rdv/book* → `patient_rdv`; *voir/annuler* → `patient_dashboard`; *login/inscription patient* → `patient_auth`; *médecin* → `med_menu`; *admin/staff* → `admin_menu`; *contact/horaires/adresse* → `infos`.
- Santé : conseils généraux, pas de diagnostic ; message d'alerte uniquement si symptômes graves.
- Ambiguïtés : proposer 2–3 choix maximum, libellés clairs.

## Aperçu résumé du code du chatbot (JavaScript)

Le script `static/js/chat.js` ajoute un widget de chat côté client :

- **UI** : crée le bouton flottant « » et le panneau (header, corps, footer).
- **Intégration n8n** : envoie `question` au webhook (`N8N_URL`) et affiche le JSON {text, choices}.
- **Fallback** : si n8n/LLM ne répond pas, des règles locales orientent vers RDV, Dashboard, Infos.
- **Sécurité** : aucune clé API dans le front ; intents filtrés via une liste blanche.

### Extrait minimal (appel n8n et rendu) :

```
1 const N8N_URL='http://localhost:5678/webhook-test/chatbot-router
  ';
2 async function askN8N(question){
3   const r = await fetch(N8N_URL,{
4     method:'POST',headers:{'Content-Type':'application/json'},
5     body: JSON.stringify({question})
6   });
7   if(!r.ok) throw new Error('n8n error');
8   return r.json();
9 }
```

```

10 function renderFromN8N(data){
11     const p = data.output ? data.output : data;           // parser
12     output
13     const text = typeof p.text==='string' ? p.text : '';
14     const choices = Array.isArray(p.choices) ? p.choices : [];
15     if(!text) throw new Error('bad payload');
16     bot(text, choices);                                   // affiche
17     bulle + boutons
18 }

```

### Workflow n8n (éléments de configuration)

1. **Webhook (POST)** : path chatbot-router, body JSON.
2. **Google Gemini** : gemini-1.5-flash, prompt conforme aux règles ci-dessus.
3. **Structured Output Parser** : impose le schéma {text, choices[] }.
4. **Respond to Webhook** : , *First Incoming Item* :

Access-Control-Allow-Origin: \*  
 Access-Control-Allow-Methods: POST, OPTIONS  
 Access-Control-Allow-Headers: Content-Type

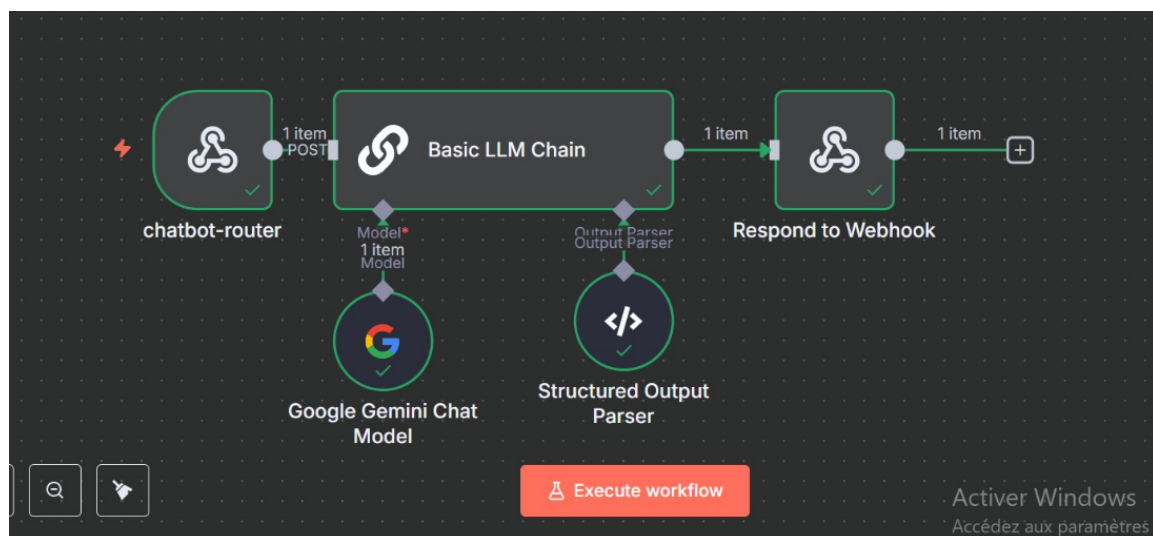


FIGURE 4.4 – Capture d’écran pour le Workflow de Chatbot (n8n)

### Sécurité et conformité

- Clés API uniquement côté n8n ; aucune clé exposée au navigateur.
- Pas de données de santé dans les prompts ; pas de diagnostic.
- Transparence : mention de l’usage d’un assistant IA et limites.

### Tests d’acceptation back-end

- **Parcours** : “prendre un rdv”, “voir/annuler RDV”, “médecin”, “admin login”, “infos/horaires”.

- 
- **Robustesse** : question vide, très longue, caractères spéciaux ; coupure n8n/LLM ⇒ fallback.
  - **Validation** : JSON strict, intents dans la liste autorisée, liens valides.

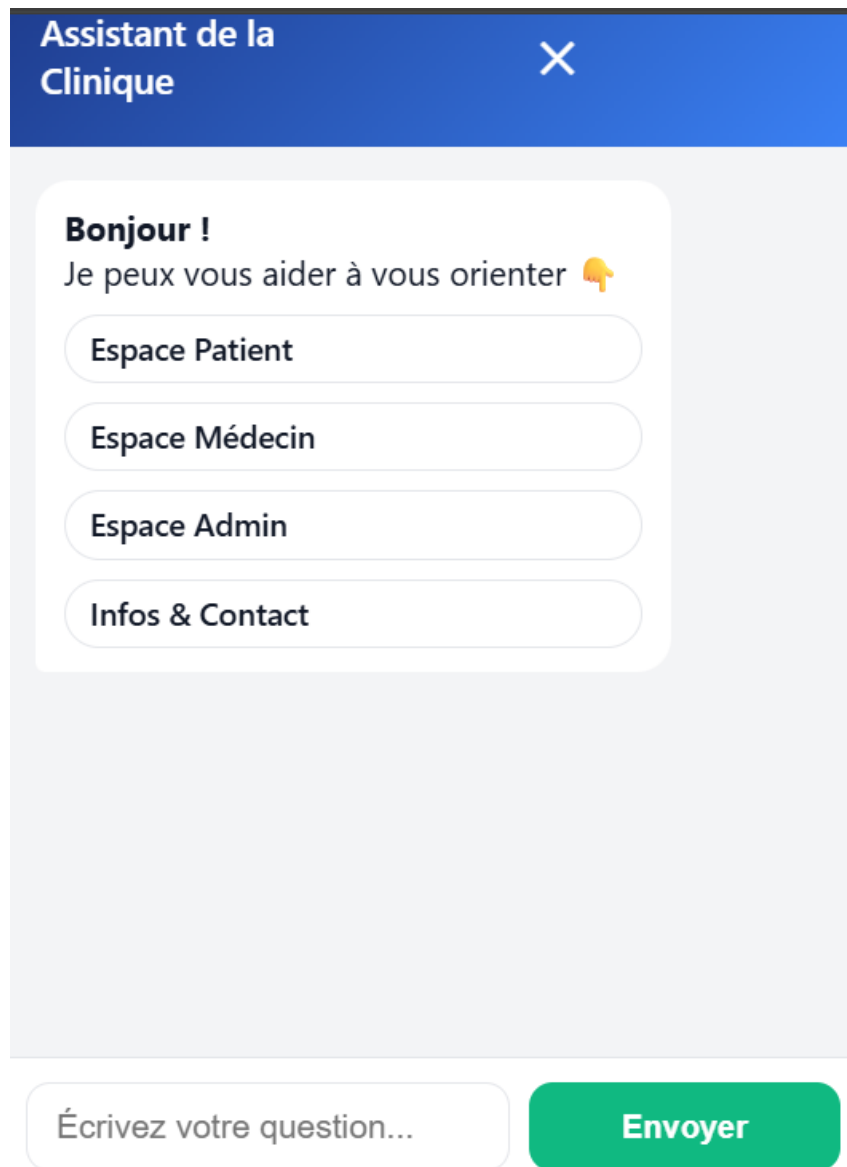


FIGURE 4.5 – Widget du chatbot sur la page d’accueil

### 4.3.1 Inscription des utilisateurs

L’inscription d’un patient (et par analogie d’un médecin ou d’un secrétaire) repose sur un workflow qui reçoit les données du formulaire via un webhook, génère automatiquement un identifiant unique (`id_patient`), puis enregistre les informations dans la feuille `patients` de Google Sheets. Le workflow se termine par une réponse JSON envoyée à Flask confirmant ou non la réussite de l’inscription.

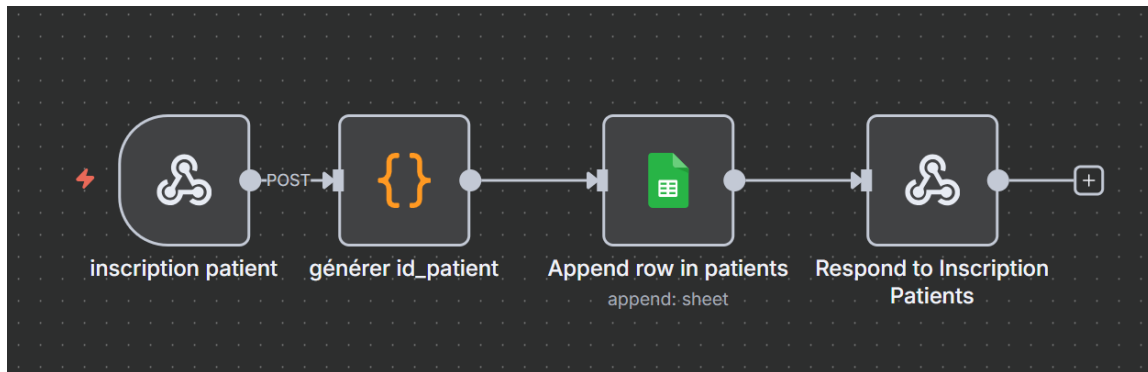


FIGURE 4.6 – Capture d’écran pour le Workflow d’inscription patient (n8n)

### 4.3.2 Authentification (connexion)

La connexion des utilisateurs repose sur la vérification des identifiants. Le workflow commence par la réception de l’email et du mot de passe via un webhook. Ensuite, une recherche est effectuée dans la feuille correspondante de Google Sheets. Si l’email est trouvé, le mot de passe est comparé à celui stocké. Le workflow retourne alors une réponse de succès ou d’erreur (*connexion réussie*, *email introuvable*, ou *mot de passe incorrect*).

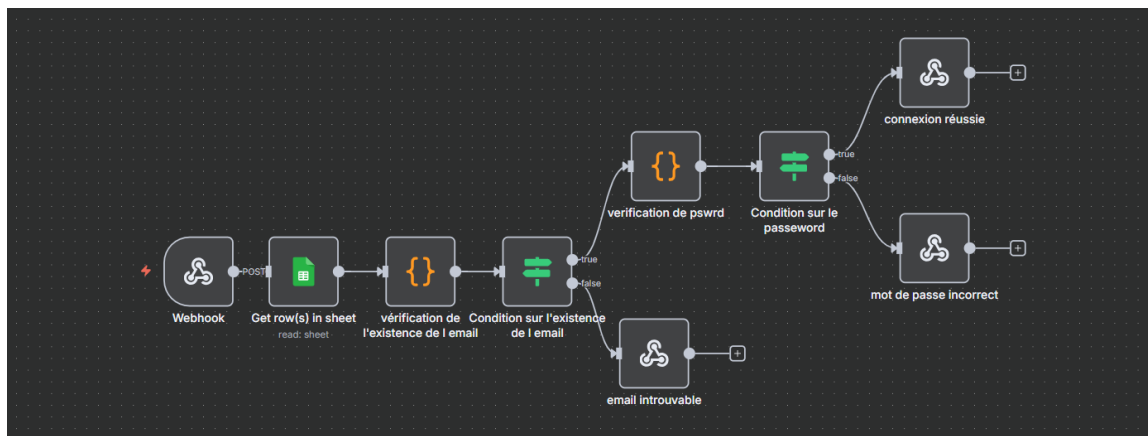


FIGURE 4.7 – Capture d’écran pour le Workflow de connexion patient (n8n)

### 4.3.3 Prise de rendez-vous

La réservation d’un rendez-vous est l’un des workflows les plus importants. Après réception du formulaire patient (nom, email, besoin médical), un module d’IA (**Google Gemini**) oriente la demande vers le bon service/médecin. Un identifiant de rendez-vous (*id\_rdv*) est généré, puis les créneaux disponibles sont vérifiés dans Google Sheets. Si un créneau est trouvé, le rendez-vous est ajouté dans l’onglet **créneaux confirmés** et un retour JSON contenant les détails de la réservation est envoyé à Flask.

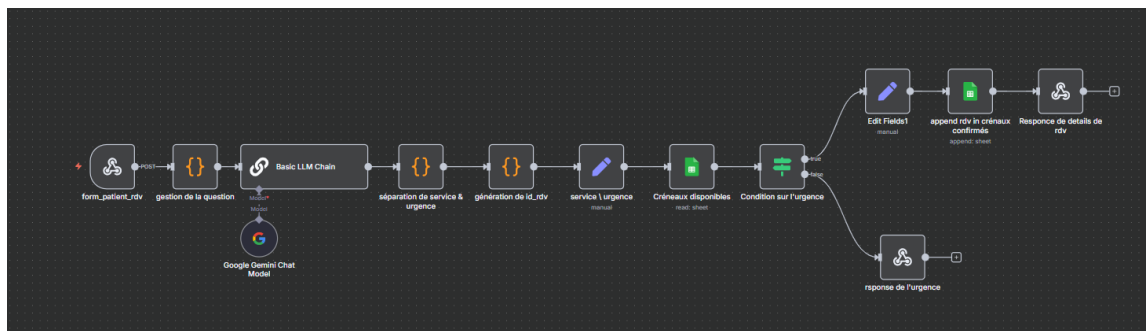


FIGURE 4.8 – Capture d’écran pour le Workflow de prise de rendez-vous (n8n)

### 4.3.4 Annulation de rendez-vous

Pour annuler un rendez-vous, le patient clique sur le bouton d’annulation dans le tableau de bord. Flask envoie alors une requête au workflow d’annulation qui recherche le rendez-vous via son `id_rdv` dans la feuille `créneaux confirmés`. Si l’entrée existe, son statut est modifié en « *annulé* » et la confirmation est renvoyée à Flask. Sinon, une erreur est retournée.

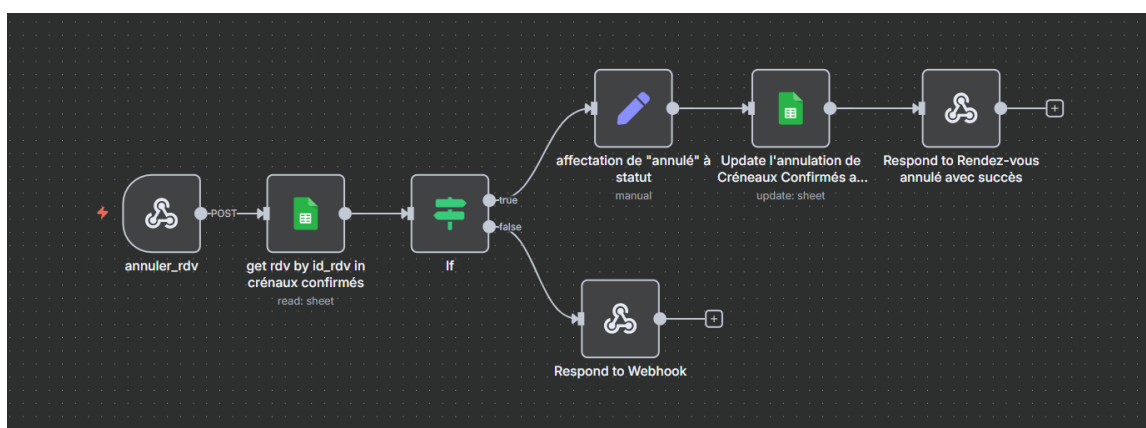


FIGURE 4.9 – Capture d’écran pour le Workflow d’annulation de rendez-vous (n8n)

### 4.3.5 Notifications et rappels automatiques

Afin de réduire le taux d’absentéisme, un workflow planifié envoie automatiquement des emails de rappel aux patients avant leurs rendez-vous. Le workflow lit la feuille `créneaux confirmés`, sélectionne les rendez-vous proches, crée un message personnalisé, puis l’envoie via l’API Gmail. Chaque rappel est enregistré dans Google Sheets afin d’assurer un suivi.

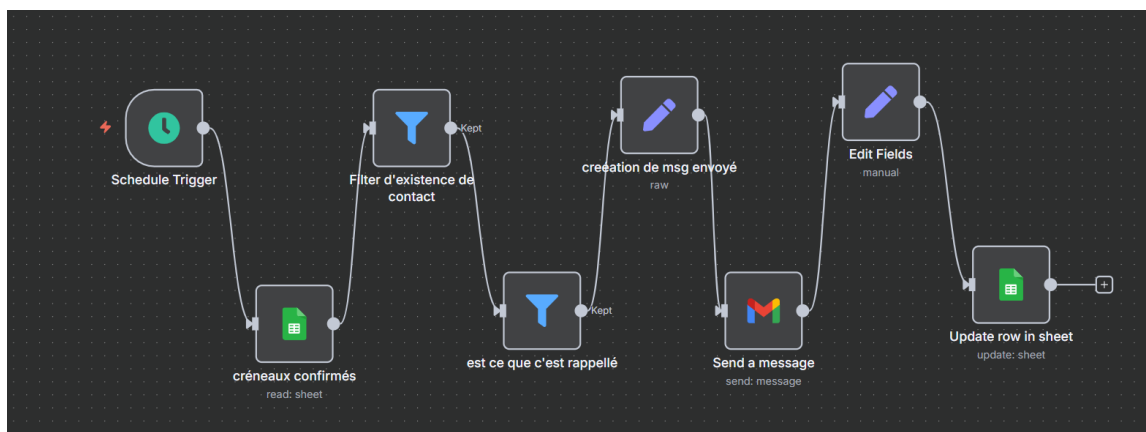


FIGURE 4.10 – Capture d’écran pour le Workflow de rappel automatique par email (n8n)

### 4.3.6 Gestion administrative des créneaux

Le personnel administratif dispose également de workflows pour gérer les créneaux confirmés. Il peut consulter, modifier ou réaffecter un rendez-vous directement dans la feuille dédiée. Les changements sont immédiatement pris en compte par n8n et reflétés dans le dashboard patient.

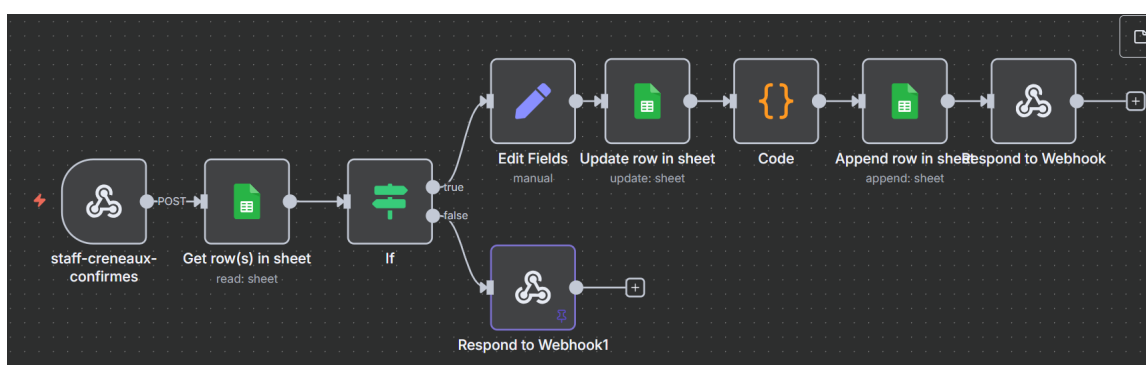


FIGURE 4.11 – Capture d’écran pour le Workflow de gestion des créneaux confirmés par le staff (n8n)

## 4.4 Tests et validation

Après le développement du frontend et la mise en place des workflows n8n, une phase de tests a été réalisée afin de vérifier le bon fonctionnement des principales fonctionnalités. Ces tests ont permis de s’assurer que la plateforme répond aux besoins définis dans le cahier des charges.

### Tests d’inscription et de connexion

Les tests effectués sur le module d’inscription ont montré que :

- un patient peut créer un compte avec succès, les informations étant bien enregistrées dans la feuille `patients` ;
- un message de confirmation est correctement affiché dans l’interface Flask ;



- 
- les workflows d’inscription médecin et secrétaire fonctionnent de manière analogue. Concernant la connexion, les scénarios suivants ont été validés :
  - Connexion réussie avec email et mot de passe corrects ;
  - Message d’erreur explicite si l’email n’existe pas dans Google Sheets ;
  - Message d’erreur en cas de mot de passe incorrect.

### Tests de prise de rendez-vous

Le workflow de réservation a été testé en simulant différents cas :

- Création d’un rendez-vous avec un créneau disponible : l’entrée apparaît bien dans la feuille **créneaux confirmés** et une confirmation est affichée côté patient.
- Demande d’un créneau déjà pris : le système refuse la réservation et retourne un message d’erreur.
- Orientation vers le bon service : l’IA Gemini a correctement redirigé les besoins vers la spécialité correspondante.

### Tests d’annulation

Le module d’annulation a été validé en :

- Annulant un rendez-vous existant : le statut passe bien à « *annulé* » dans Google Sheets ;
- Tentant d’annuler un rendez-vous inexistant : le workflow retourne un message d’erreur géré par Flask.

### Tests des notifications

Enfin, les tests de notifications planifiées ont permis de confirmer que :

- un email de rappel est envoyé avant la date de chaque rendez-vous confirmé ;
- les rappels envoyés sont correctement marqués dans Google Sheets ;
- en cas de patient inexistant ou email invalide, le message n’est pas envoyé et une erreur est loggée.

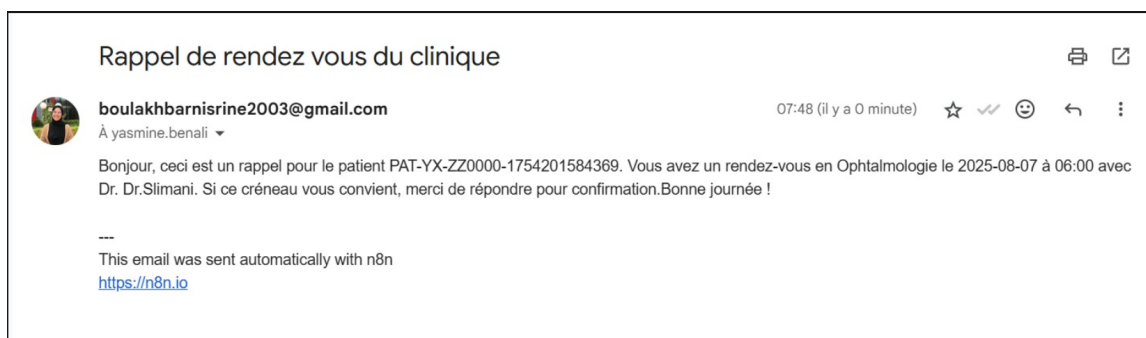


FIGURE 4.12 – Exemple d’email automatique de rappel de rendez-vous généré par n8n

### Validation globale

L’ensemble des fonctionnalités a été validé par une série de scénarios utilisateurs (patients, médecins, secrétaires). Les résultats confirment la conformité du système par rapport aux objectifs initiaux : fluidité de l’inscription, fiabilité des rendez-vous, et réduction du risque d’erreurs humaines dans la gestion des créneaux.

---

## 4.5 Conclusion Chapitre 4

Ce chapitre a présenté en détail la réalisation technique du projet. Nous avons décrit le développement du **frontend**, basé sur Flask et des interfaces HTML/CSS, qui offre une navigation claire et intuitive aux différents utilisateurs. Nous avons ensuite détaillé la mise en œuvre du **backend**, construit autour de la plateforme n8n et de Google Sheets, permettant d'automatiser l'ensemble des processus : inscription, connexion, prise de rendez-vous, annulation, notifications et gestion administrative.

Enfin, une phase de **tests et validation** a permis de vérifier le bon fonctionnement des modules développés, confirmant la fiabilité et la conformité de la solution par rapport aux besoins exprimés dans le cahier des charges. Ainsi, la plateforme obtenue constitue une première version opérationnelle de clinique intelligente, capable de gérer efficacement les flux de rendez-vous et d'améliorer l'expérience des patients comme du personnel médical et administratif.

## 5. Conclusion et perspectives

### 5.1 Synthèse

Ce travail de stage avait pour objectif principal de concevoir et de développer une **plateforme intelligente de gestion et de prise de rendez-vous pour une clinique**, en s'appuyant sur des technologies modernes et accessibles.

Le projet a suivi une méthodologie progressive :

- une phase de **conception fonctionnelle**, comprenant l'analyse des besoins, l'identification des acteurs et la modélisation des cas d'utilisation ;
- une phase de **conception technique**, avec l'élaboration de l'architecture du système, le diagramme de classes et le diagramme de séquence ;
- une phase de **réalisation**, marquée par le développement du frontend en Flask/HTML/CSS et la mise en place de workflows n8n interconnectés avec Google Sheets ;
- enfin, une phase de **tests et validation**, qui a confirmé la conformité de la solution aux objectifs définis dans le cahier des charges.

Les résultats obtenus démontrent la faisabilité et l'efficacité d'une telle solution : l'inscription et la connexion des utilisateurs fonctionnent correctement, la gestion des rendez-vous est automatisée, et des rappels sont envoyés pour réduire le risque d'oubli. L'intégration d'un module d'IA (Gemini) constitue un atout supplémentaire pour l'orientation médicale.

Ce stage a également permis de renforcer mes compétences techniques (Flask, n8n, Google Sheets, automatisation) et de développer des aptitudes en gestion de projet, autonomie et résolution de problèmes.

### 5.2 Perspectives

Bien que le projet ait atteint ses objectifs initiaux, plusieurs améliorations et extensions peuvent être envisagées :

- **Migration vers une base de données robuste** : remplacer Google Sheets par une base relationnelle (MySQL, PostgreSQL) ou NoSQL pour améliorer la performance et la sécurité.
- **Renforcement de la sécurité** : implémenter le chiffrement des mots de passe, utiliser des protocoles sécurisés (OAuth2, JWT) et ajouter une gestion des rôles plus fine.
- **Extension des canaux de notification** : intégrer l'envoi de rappels par SMS ou WhatsApp en plus des emails pour une meilleure accessibilité.
- **Dossier médical numérique** : permettre aux médecins de générer et partager des prescriptions, ordonnances et comptes rendus via la plateforme.

- 
- **Interface plus riche et responsive** : améliorer l'expérience utilisateur avec un design plus moderne et compatible avec tous les supports (PC, tablette, mobile).
  - **Exploitation avancée de l'IA** : intégrer un chatbot médical plus complet, capable de réaliser un pré-diagnostic ou de guider le patient avec plus de précision.

## 5.3 Conclusion générale

En définitive, ce stage nous a permis d'allier **connaissances théoriques et application pratique**, tout en développant des compétences solides en développement logiciel et en gestion de projet. Il nous a donné l'opportunité de découvrir concrètement le fonctionnement d'une entreprise innovante comme AHDIGITAL, et de confirmer notre intérêt pour le domaine de l'intelligence artificielle appliquée au secteur de la santé.

Cette expérience constitue ainsi un tremplin pour nos futures études et projets, en consolidant, notre capacité d'analyse ainsi notre motivation à m'investir dans des solutions technologiques à fort impact sociétal.

# Annexes techniques

## Introduction

Les annexes qui suivent présentent des éléments complémentaires illustrant la mise en œuvre technique du projet. Elles regroupent des captures d'écran des interfaces utilisateur, des extraits de workflows n8n et des portions de code Flask, afin d'apporter des preuves concrètes du travail réalisé au cours du stage.

## Annexe A : Interfaces de la plateforme

- **Page de connexion et d'inscription** : permet aux patients de créer un compte ou de se connecter afin d'accéder à l'espace patient.
- **Assitant d'orientation (chatbot)** : affiche la liste des espaces valables et oriente l'utilisateur selon son besoin et sa question .
- **Espace patient** : offre deux options principales : prendre un rendez-vous ou consulter l'historique des rendez-vous.
- **Formulaire de rendez-vous** : interface où le patient saisit ses informations et son besoin médical avant soumission.
- **Dashboard patient** : affiche la liste des rendez-vous confirmés, avec possibilité d'annulation en un clic.
- **Notification email** : exemple de rappel automatique envoyé avant la consultation.



FIGURE 1 – Exemple d'interface : Capture d'écran de la page d'accueil avec le chatbot assistant

---

## Annexe B : Workflows n8n

Plusieurs workflows ont été développés dans n8n pour automatiser les processus métier :

- **Workflow d'inscription patient** : reçoit les données du formulaire et enregistre un nouveau patient avec un identifiant unique dans Google Sheets.

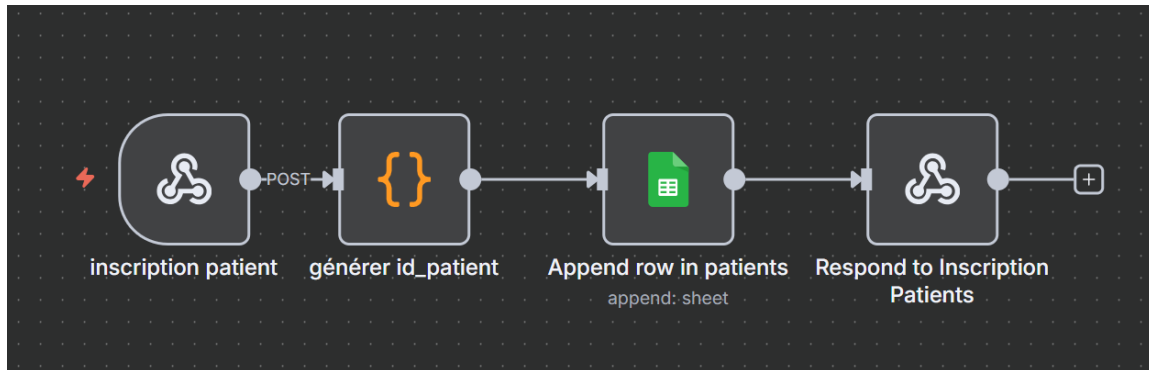


FIGURE 2 – Capture d'écran pour le Workflow d'inscription patient (n8n)

- **Workflow de connexion patient** : vérifie l'email et le mot de passe fournis, puis renvoie une réponse de succès ou d'erreur.
- **Workflow de prise de rendez-vous** : oriente le besoin du patient vers un médecin via IA Gemini, génère un ID de rendez-vous et l'ajoute dans Google Sheets.

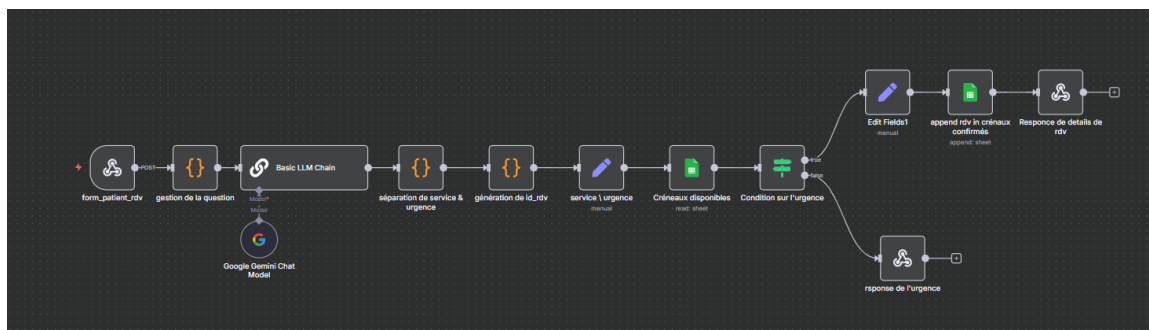


FIGURE 3 – Capture d'écran pour le Workflow de prise de rendez-vous (n8n)

- **Workflow d'annulation** : recherche le rendez-vous à partir de son identifiant et met à jour son statut en "annulé".
- **Workflow de rappels automatiques** : envoi automatique d'emails de rappel avant le rendez-vous afin de limiter les absences.

## Annexe C : Extrait de code Flask

Cette annexe présente un extrait représentatif du fichier `app.py`, qui illustre la manière dont l'application Flask interagit avec les workflows n8n. Il s'agit ici de la route `/register`, qui reçoit les données saisies par un patient dans le formulaire d'inscription, les transmet au webhook n8n via une requête POST, puis affiche le message de confirmation ou d'erreur retourné par le backend.

---

Ci-dessous un extrait représentatif du fichier `app.py`, montrant la route `/register` qui relie le formulaire d'inscription au webhook n8n.

```
1 # --- Inscription patient ---
2 @app.route("/register", methods=["GET", "POST"])
3 def register():
4     if request.method == "POST":
5         nom = (request.form.get("nom") or "").strip()
6         prenom = (request.form.get("prenom") or "").strip()
7         email = (request.form.get("email") or "").strip().lower()
8         password = (request.form.get("password") or "").strip()
9         try:
10             resp = requests.post(
11                 WEBHOOKS["register_patient"],
12                 json={"nom": nom, "prenom": prenom, "email": email, "password": password},
13                 timeout=REQUEST_TIMEOUT,
14             )
15             print("n8n register patient:", resp.status_code, resp.text)
16             data = safe_json(resp)
17             if resp.ok and success_register(data):
18                 flash(data.get("message", "Inscription réussie."), "success")
19                 if data.get("id_patient"):
20                     session["id_patient"] = data["id_patient"]
21                     session["user_email"] = email
22                     session["role"] = "patient"
23                     return redirect(url_for("patient_space"))
24             elif resp.ok:
25                 flash(data.get("message", "Inscription impossible."), "error")
26             else:
27                 flash("Erreur lors de l'inscription.", "error")
28             except Exception as e:
29                 print("Erreur n8n (register patient):", e)
30                 flash("Erreur de connexion au serveur d'inscription.", "error")
31             return redirect(url_for("register"))
32             return render_template("register.html")
```

Listing 1 – Extrait du fichier

## Annexe D : Table Google Sheets

Cette annexe présente la structure des différentes feuilles : Patients, Médecins, Secrétaires, RendezVous et Créneaux. Elle permet de visualiser la manière dont les informations sont organisées et reliées entre elles.

Exemple de structure de la feuille Google Sheets utilisée comme base de données :

- **Patients** : Id\_Patient, Nom, Prénom, Email\_Patient, Password, Created\_at
- **Médecins** : id\_medecin, Nom, Prénom, Email\_medecin, Password, Spécialité, Created\_at
- **Secrétaires** : id\_secretaire, Nom, Prénom, Email\_sec, Password, Created\_at
- **RendezVous** : id\_rdv, Id\_Patient, id\_medecin, Service, Date, Heure, Statut
- **Créneaux disponibles** : Jour, Durée, id\_medecin, Service

L6											
	A	B	C	D	E	F	G	H	I	J	
1	DATE	HEURE	Id_rdv	id_medecin	SERVICE	gmail patient	Besoin	rappel automatique	Statut	disponible	
2											

	A	B	C	D	E	F
1	Id_Patient	Nom	Prenom	Email_Patient	Password	Created_at
2						

A1						
	A	B	C	D	E	F
1	id_medecin	Nom	Prenom	Email_medecin	Passesword	Created_at

FIGURE 4 – Capture d’écran Exemple de structure de la feuille Google Sheets



# Webographie

- Documentation officielle de Flask : <https://flask.palletsprojects.com/>
- Documentation officielle de n8n : <https://docs.n8n.io/>
- Google Developers – Documentation Google Sheets API : <https://developers.google.com/sheets/api>
- LangChain – Documentation Gemini et LLMs : <https://python.langchain.com/>
- Docker – Documentation officielle : <https://docs.docker.com/>
- Overleaf – Ressources LaTeX pour rédaction académique : <https://www.overleaf.com/learn>
- W3Schools – Tutoriels HTML et CSS : <https://www.w3schools.com/>
- Stack Overflow – Forum de développeurs pour dépannage et bonnes pratiques : <https://stackoverflow.com/>