



# Ingénierie informatique et big Data

## Mémoire de Master

STAGE DE FIN D'ÉTUDES

### Développement d'un Chatbot Médical Intelligent

Analyse des Symptômes, Recherche de  
Médecins et Informations sur les Médicaments

Intégration de Dialogflow et OpenAI pour l'Assistance Médicale  
Automatisée

DEFENDED ON: JULY 15, 2025

Supervised By **Pr. taoufik Fouad**

*Realised By:*

**RIAD JAWAD**

**Titre :** Conception d'un Chatbot Médical Intelligent pour l'Assistance Sanitaire : Analyse des Symptômes, Recherche de Médecins et Informations sur les Médicaments.

**Mots clés :** Chatbot médical, Santé, Symptômes, Assistance automatisée, Information médicale, Communication patient-système, Médecins, Médicaments.

**Résumé:** Actuellement, la technologie est un moyen extrêmement puissant d'aider le monde de la santé en ce qui concerne le développement des applications et la modernisation du système des réseaux. En d'autres termes, une avancée technologique adapte les services médicaux évite la surcharge des centres de traitement, le manque du personnel médical, et l'incapacité de la population d'accéder aux services de professeurs médicaux. C'est donc ici que notre idée, sous la forme de service dédié à la santé familiale enseignée par un chatbot de santé formule avec des informations sur la santé basées sur l'intelligence artificielle, se positionne.

Ce chatbot est conçu pour pouvoir communiquer avec un utilisateur de façon formelle, sans connaître de langage avancé. Grâce à la simple discussion avec la ou le chatbot à propos de ses symptômes, l'utilisateur est plus au courant de sa pathologie. Ce genre de technologie qui s'appelle "médecins virtuels" s'avère indispensable pour le cadre dans lequel sont placés les urgences et les hôpitaux, et quand nous faisons appel à l'intelligence artificielle pour mettre en rapport avec un professionnel de la santé et/ou quelqu'un spécialisé dans un domaine spécifique dont le village se trouve à telle distance en cas de besoin nous sommes en ligne avec le fou et l'ergot.

Cette combinaison sert exclusivement à la création d'une application e-Santé ingénieuse et

fortement opérationnelle surtout orientée survie mais accessible 24 heures sur 24. Elle emploie la prévention, le contrôle et la communication. Dans d'autres mots, ce chatbot n'est pas du tout un docteur mais il est une aide numérique essentielle pour certaines personnes car il est en mesure de leur conseiller sans tarder et de manière très précise au début de leur premier contact avec un patient. Cette façon de s'occuper de la santé est intelligente, en permettant de toucher un secteur large de la population et de responsabiliser les patients, ce qui peut dynamiser les systèmes de prestation du soin aux clients et aussi améliorer son efficacité.

Il assiste dans un cadre sémantique très spécifique, mais tout en étant combative, faisant également de l'intégration intelligente entre l'intelligence artificielle, la technologie de traitement du langage naturel, et la santé publique. L'intégration de ces trois éléments dans le déroulement de l'activité politique de santé est sans pareille et renverse la perception à propos des avatars de la parole manufacturée ce vu des marqueurs dans l'établissement des liens entre les acteurs augmentés de leurs services et clients tels Leslie assureurs et les patients. Nous unissons la vie et la technologie pour créer une santé numérique intégrée dans le quotidien de chacun, et pleinement acceptée dans une approche globale.

## Remerciements

---

Nous tenons à exprimer notre profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet de recherche.

Nous souhaitons tout particulièrement remercier notre encadrant, **Mr taoufik Fouad**, pour son soutien indéfectible, ses précieux conseils et son encadrement attentif tout au long de ce projet.

Ses orientations ont été déterminantes pour la réussite de cette recherche.

Nos remerciements s'adressent également à **Mr moumoun lahcen**, pour son accompagnement constant et ses conseils éclairés, qui ont été d'une aide inestimable tout au long de ce parcours académique.

À tous, merci pour votre précieuse contribution.



## Dédicace

---

Ces lettres je les cède à ma famille. Ils m'ont appris à ne jamais perdre confiance en soi. Ils m'ont convaincu que malgré tous les obstacles on peut toujours avancer. Tout ce qu'on doit avoir c'est une inépuisable volonté.

Je me suis toujours dit que je ne peux rien leur offrir. Cependant, ils m'ont fait sentir que juste le fait de vouloir apprendre et avancer dans sa vie. Juste le fait de vouloir un jour être là pour eux, même si j'y n'arrive pas. C'est déjà la paix pour eux, car ils savent que j'y arriverai. Ils me font comprendre que j'arriverai un jour à être mieux, et pas pour eux mais pour moi-même.

Ils sont la raison que j'apprends, j'écris, je rêve, je pense, j'espère et je désire. Merci père, mère.

# Contents

<b>Acknowledgements</b>	<b>2</b>
<b>Dédicace</b>	<b>4</b>
<b>List of Figures</b>	<b>6</b>
<b>1 spécification et analyse des besoins</b>	<b>9</b>
1.1 Etude de l'existant . . . . .	10
1.1.1 Description de l'existant . . . . .	10
1.1.2 Critique de l'existant . . . . .	10
1.1.3 Solution proposée . . . . .	11
1.2 Besoins Fonctionnels et Non Fonctionnels . . . . .	11
1.2.1 Besoins Fonctionnels . . . . .	11
1.2.2 Besoins Non Fonctionnels . . . . .	12
1.3 Cadre du Projet . . . . .	12
1.3.1 Présentation de l'application MedBot . . . . .	13
<b>2 Conception</b>	<b>15</b>
2.1 Conception Générale . . . . .	16
2.2 Conception détaillée . . . . .	18
2.2.1 Le Diagramme de classe . . . . .	18
2.3 Conclusion . . . . .	21
<b>3 Réalisation</b>	<b>23</b>
3.1 Introduction aux Technologies Utilisées . . . . .	25
3.2 Utilisation de Dialogflow pour l'intelligence conversationnelle . . . . .	25
3.3 Mise en place du backend avec Flask . . . . .	31
3.4 Page d'Accueil . . . . .	36
3.5 Navigation de l'application . . . . .	38
3.6 Conclusion sur l'intégration du chatbot médical . . . . .	42
<b>Bibliography</b>	<b>43</b>

# List of Figures

1.1	Schéma du cadre de projet de Application de MedBot . . . . .	13
2.1	Schéma du modèle de cycle de vie . . . . .	16
2.2	Architecture 3-tiers de l'AGSS . . . . .	17
2.3	Diagramme de classes du système de chatbot médical . . . . .	18
2.4	Diagramme de cas d'utilisation de l'utilisateur et du système . . . . .	20
2.5	Diagramme de séquence : Interaction avec le chatbot médical . . . . .	21
3.1	Carte Mentale : Exemple de cohérence visuelle . . . . .	26
3.2	Exemple d'intents configurés dans Dialogflow . . . . .	27
3.3	Exemple d'intents configurés dans Dialogflow . . . . .	28
3.4	Interface de création des intents dans Dialogflow . . . . .	29
3.5	Exemple d'entité nom_medicament avec variantes . . . . .	29
3.6	Exemple de script de scraping médecins utilisant Selenium et ChromeDriver . . . . .	30
3.7	Exemple de script de scraping médecins utilisant Selenium et ChromeDriver . . . . .	30
3.8	Structure JSON des données des médecins . . . . .	31
3.9	Structure JSON des données des médicaments . . . . .	31
3.10	Étapes pour mettre en place un projet Flask . . . . .	32
3.11	Créer un fichier app.py de base . . . . .	32
3.12	Connexion de Dialogflow à Flask via Ngrok . . . . .	33
3.13	les paramètres de connection supabase par flask . . . . .	34
3.14	la base de données Supabase . . . . .	35
3.15	l'application Frontend . . . . .	35
3.16	l'application Backend . . . . .	35
3.17	Page d'Accueil de l'application . . . . .	36
3.18	À propos solution médicale intelligente . . . . .	37
3.19	Aperçu des services proposés par l'application . . . . .	37
3.20	Formulaire de contact de l'application . . . . .	38
3.21	Modal de connexion et d'inscription des utilisateurs . . . . .	38
3.22	Vidéo démonstrative de l'application et des fonctionnalités du chatbot . . . . .	39
3.23	Conversation avec le chatbot médical - Exemple 1 . . . . .	39
3.24	Conversation avec le chatbot médical - Exemple 2 . . . . .	40

3.25 Conversation avec le chatbot médical - Exemple 3 . . . . .	40
3.26 Conversation avec le chatbot médical - Exemple 4 . . . . .	40
3.27 Conversation avec le chatbot médical - Exemple 5 . . . . .	40
3.28 Articles Médicaux avec la recherche de symptômes . . . . .	41
3.29 Articles Médicaux avec la recherche de symptômes en arabe . . . . .	41
3.30 Affichage d'informations sur les médicaments à partir du chatbot . . . . .	42
3.31 Interface de la section Discussion de l'application . . . . .	42



# Chapter 1 spécification et analyse des besoins

---

## Contenu du chapitre

1.1	Etude de l'existant . . . . .	10
1.1.1	Description de l'existant . . . . .	10
1.1.2	Critique de l'existant . . . . .	10
1.1.3	Solution proposée . . . . .	11
1.2	Besoins Fonctionnels et Non Fonctionnels . . . . .	11
1.2.1	Besoins Fonctionnels . . . . .	11
1.2.2	Besoins Non Fonctionnels . . . . .	12
1.3	Cadre du Projet . . . . .	12
1.3.1	Présentation de l'application MedBot . . . . .	13

---

## Abstract

---

Dans ce chapitre, nous mettons le sujet dans son cadre général. Standard la suite, nous abordons l'étude de l'existant du projet, suivie d'une étude pour dégager les contraintes à respecter pendant la réalisation de notre projet.

Ainsi, ce chapitre présente l'ensemble des besoins, qu'ils soient fonctionnels ou non fonctionnels, nécessaires à la réalisation du projet.

---

Au sein du présent chapitre, l'attention est portée sur la conception d'un chatbot associé au domaine médical. Cette solution a été spécifiquement développée en réponse aux besoins croissants d'une information claire, rapide et accessible pour des usagers souhaitant mieux appréhender leur santé. Le chatbot propose d'accompagner les usagers dans la compréhension de leurs symptômes, la recherche d'un professionnel de santé ou encore la consultation d'informations fiables sur les médicaments. Tous les besoins identifiés et les constats provenant de l'étude de l'existant ont été retenus afin d'orienter le développement de cette application.

## 1.1 Etude de l'existant

### 1.1.1 Description de l'existant

À l'heure actuelle, plusieurs plateformes numériques proposent des services d'information médicale à destination du grand public. Ces solutions en ligne offrent des fonctionnalités diversifiées permettant aux utilisateurs d'accéder à du contenu médical vulgarisé et fiable.

- **Consultation de Symptômes** : Des sites comme *WebMD* ou *Healthline* proposent des outils interactifs ("symptom checkers") permettant aux utilisateurs de décrire leurs symptômes et d'obtenir une liste de pathologies possibles à titre informatif.
- **Base de Données Médicale** : Ces plateformes offrent un accès structuré à des fiches descriptives sur les maladies, les traitements, les causes, les symptômes, les complications, ainsi que les options de prévention.
- **Informations sur les Médicaments** : Il est possible de consulter des bases de données de médicaments, incluant leur nom, posologie, effets secondaires, interactions et précautions d'usage.
- **Articles de Santé et Bien-être** : De nombreux articles mis à jour régulièrement couvrent des sujets liés à la nutrition, la santé mentale, les premiers secours, la prévention, et d'autres domaines connexes.
- **Accessibilité Multilingue** : Certaines plateformes, telles que *Doctissimo* en français ou *Healthline* en anglais, adaptent leur contenu à différents contextes culturels et linguistiques.

### 1.1.2 Critique de l'existant

Les plateformes actuelles d'information médicale, bien qu'utiles, présentent plusieurs limitations :

- **Interaction limitée** : L'absence d'un système conversationnel empêche les utilisateurs de poser des questions en langage naturel et d'obtenir des réponses personnalisées en temps réel.

- 
- **Pas de prise en compte du contexte utilisateur** : Les informations proposées ne s'adaptent pas à la situation spécifique de chaque utilisateur (âge, antécédents, symptômes croisés, etc.).
  - **Navigation manuelle** : L'utilisateur doit chercher activement l'information, ce qui peut être difficile pour des personnes non familières avec les termes médicaux ou en situation d'urgence.
  - **Pas d'accompagnement guidé** : Aucune fonctionnalité n'assure un suivi pas à pas ou ne propose d'orientation vers un professionnel en fonction des réponses fournies par l'utilisateur.

### 1.1.3 Solution proposée

Face aux limites identifiées dans les plateformes actuelles d'information médicale, nous proposons la conception d'un chatbot médical intelligent. Ce dernier vise à offrir une assistance automatisée, accessible et interactive à tout utilisateur en quête d'informations de santé fiables.

- **Interaction conversationnelle** : Le chatbot permet à l'utilisateur de poser ses questions en langage naturel. Il peut décrire ses symptômes, demander des informations sur un médicament ou solliciter une aide pour trouver un professionnel de santé.
- **Réponses personnalisées** : En fonction des données saisies (symptômes, âge, sexe, localisation, etc.), le système adapte ses réponses pour fournir un contenu pertinent et contextualisé.
- **Orientation vers des professionnels** : Le chatbot peut suggérer, à titre informatif, des types de médecins à consulter selon les symptômes mentionnés, et indiquer des praticiens disponibles dans une ville donnée.
- **Consultation simplifiée des médicaments** : Il permet d'accéder rapidement à la description, l'usage, les effets secondaires et les précautions concernant un médicament donné.
- **Disponibilité continue** : Accessible 24h/24 et 7j/7, ce système offre un premier niveau d'assistance médicale, particulièrement utile dans les zones à faible accès aux soins ou en dehors des heures d'ouverture des centres de santé.

## 1.2 Besoins Fonctionnels et Non Fonctionnels

Dans cette section du chapitre, nous nous intéressons aux besoins des utilisateurs à travers les spécifications fonctionnelles et non fonctionnelles afin de développer une application de qualité conforme aux attentes du domaine de la santé numérique.

### 1.2.1 Besoins Fonctionnels

Les besoins fonctionnels se divisent en plusieurs catégories importantes :

---

a) **Interaction utilisateur – chatbot**

- **Saisie libre des symptômes** : L'utilisateur peut décrire ses symptômes en langage naturel pour recevoir des suggestions informatives sur d'éventuelles affections.
- **Réponses automatisées** : Le chatbot analyse les demandes et fournit des réponses médicales générales adaptées à la situation exprimée.

b) **Consultation d'informations médicales**

- **Recherche de médicaments** : L'utilisateur peut consulter la fiche descriptive d'un médicament (nom, usage, effets, précautions).
- **Requête sur des maladies** : Accès à des informations vulgarisées sur les maladies, leurs symptômes, et leur traitement.

c) **Orientation vers un professionnel de santé**

- **Recherche de médecins** : Possibilité d'indiquer une spécialité et une ville pour recevoir une liste de médecins correspondants.
- **Filtrage par localisation** : L'utilisateur peut restreindre sa recherche à une région géographique donnée.

d) **Expérience conversationnelle continue**

- **Maintien du contexte** : Le chatbot conserve le fil de la conversation afin de permettre un dialogue fluide et cohérent.
- **Personnalisation des échanges** : Les réponses tiennent compte des informations précédemment fournies par l'utilisateur (âge, sexe, historique).

### 1.2.2 Besoins Non Fonctionnels

Les besoins non fonctionnels incluent les aspects suivants :

- **Performance** : Assurer que l'application reste rapide et réactive.
- **Sécurité** : Protéger les données des utilisateurs contre les accès non autorisés.
- **Accessibilité** : Rendre l'application utilisable par tous, y compris les personnes handicapées.
- **Scalabilité** : Permettre à l'application de gérer une augmentation du nombre d'utilisateurs et de données.
- **Ergonomie et bon IHM** : L'application doit être facile à utiliser avec peu d'effort de la part de l'utilisateur.
- **Efficacité** : Permettre l'accomplissement des tâches avec un minimum de manipulations.

## 1.3 Cadre du Projet

Dans cette section, nous décrivons le cadre du projet de développement d'une application web conversationnelle dédiée à la santé, intégrant un chatbot médical. Cette application a pour vocation d'offrir un accès simplifié à des informations médicales fiables, de guider les utilisateurs

---

dans la compréhension de leurs symptômes, et de faciliter l'orientation vers des professionnels de santé.

### 1.3.1 Présentation de l'application MedBot

L'application web \*\*MedBot\*\* a été conçue pour répondre aux besoins suivants :

– **Assistance médicale via chatbot :**

- Permet aux utilisateurs de décrire leurs symptômes en langage naturel.
- Fournit des réponses informatives et des suggestions de pathologies possibles.
- Oriente l'utilisateur vers des professionnels de santé selon la spécialité et la localisation.

– **Consultation de médicaments :**

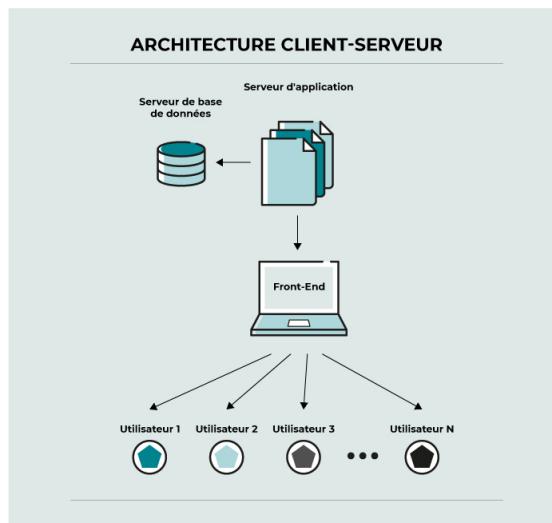
- Permet de rechercher un médicament par nom ou symptôme associé.
- Affiche la composition, les usages, les effets secondaires et les précautions d'usage.

– **Recherche d'articles médicaux :**

- Met à disposition une base d'articles vulgarisés sur divers sujets de santé.
- Permet à l'utilisateur de rechercher des contenus éducatifs selon ses besoins.

– **Espace de discussion entre utilisateurs :**

- Intègre un système de chat communautaire pour échanger sur des questions de santé.
- Favorise l'entraide, le partage d'expériences et la diffusion d'informations utiles.



**Figure 1.1:** Schéma du cadre de projet de Application de MedBot

---

### 1.3.2 Conclusion

Ce chapitre a présenté le cadre général de l'application « MedBot », en mettant en lumière ses objectifs, ses fonctionnalités principales, ainsi que les besoins fonctionnels et non fonctionnels qui ont guidé sa conception. L'application vise à offrir une assistance médicale accessible, rapide et fiable à travers un chatbot intelligent capable de dialoguer avec les utilisateurs en langage naturel. Les différentes sections de l'application permettent de rechercher des symptômes, consulter des informations sur des médicaments, trouver des médecins selon leur spécialité et leur localisation, ainsi que lire des articles vulgarisés. Un espace d'échange entre utilisateurs complète la plateforme afin d'encourager le partage d'expériences et la diffusion d'informations utiles. Cette base fonctionnelle constitue le socle sur lequel s'appuiera la modélisation technique du projet dans les chapitres suivants.

## Chapter 2 Conception

---

### Chapter Contents

2.1	Conception Générale . . . . .	16
2.1.1	Le Modèle de Cycle de Vie . . . . .	16
2.1.2	Méthodologie adoptée . . . . .	16
2.1.3	Concept et architecture de l'AGSS (Application de MedBot) . . . . .	17
2.2	Conception détaillée . . . . .	18
2.2.1	Le Diagramme de classe . . . . .	18
2.2.1	Le Diagramme de classe . . . . .	18
2.2.2	Les diagrammes de cas d'utilisation . . . . .	19
2.2.3	Diagramme de séquence . . . . .	20
2.3	Conclusion . . . . .	21

---

### Abstract

---

La phase de conception constitue un élément crucial dans le cycle de vie de développement d'une application. Elle a pour but de transformer le modèle conceptuel du système, qui a été élaboré lors de l'étape précédente d'analyse des besoins, en modèles détaillés de l'architecture du système. Cette étape est essentielle pour clarifier et organiser les spécifications fonctionnelles et non fonctionnelles en une structure cohérente qui guide le développement ultérieur.

En somme, la phase de conception vise à élaborer une feuille de route précise pour le développement, en s'assurant que toutes les exigences sont couvertes et que l'architecture du système est optimisée pour répondre aux besoins identifiés lors de l'analyse.

## 2.1 Conception Générale

### 2.1.1 Le Modèle de Cycle de Vie

Le modèle de cycle de vie d'une application est un cadre structuré qui décrit les étapes et les processus nécessaires pour développer une application depuis sa conception jusqu'à sa mise en œuvre et son maintien. Ce modèle assure que chaque phase du développement est systématiquement abordée et que les exigences du projet sont satisfaites de manière efficace et cohérente.

#### a. Présentation du Modèle

La présentation du modèle de cycle de vie décrit comment chaque phase du développement est abordée, en commençant par la conception initiale jusqu'à la mise en œuvre et le support continu. Ce modèle offre une vue d'ensemble des processus impliqués et assure une approche structurée pour le développement d'applications.

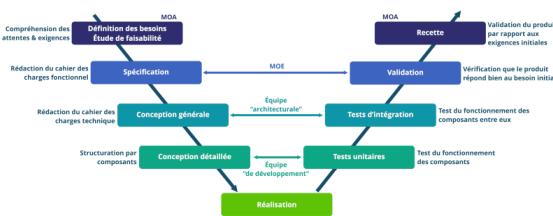


Figure 2.1: Schéma du modèle de cycle de vie

#### a. Description du modèle

La présentation du modèle de cycle de vie décrit comment chaque phase du développement est abordée, en commençant par la conception initiale jusqu'à la mise en œuvre et le support continu. Ce modèle offre une vue d'ensemble des processus impliqués et assure une approche structurée pour le développement d'applications.

### 2.1.2 Méthodologie adoptée

Pour maximiser la compréhension et assurer la bonne exécution du projet, nous avons opté pour la **méthodologie** de développement **RUP** (\*Rational Unified Process\*). Ce choix repose sur le fait que cette approche est un standard reconnu, offrant une **conception** détaillée qui prend en compte chaque **aspect** de la conception, y compris les systèmes existants. Le **RUP** se distingue

---

par sa capacité à offrir une vue d'ensemble sur le **système d'information** dans son ensemble, ce qui le rend particulièrement adapté aux projets initiés à partir de zéro.

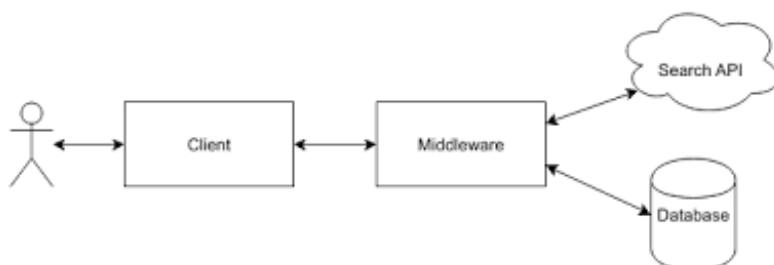
Cette méthodologie repose sur le langage de modélisation **UML** (\*Unified Modeling Language\*), que nous avons choisi pour guider tout notre travail. En réalité, **UML** n'est pas une méthode ou un processus, mais bien un **langage** de modélisation. **UML** se caractérise par une **notation** très précise et des règles grammaticales spécifiques qui permettent de créer des modèles de **logiciels**.

L'**UML** propose un ensemble varié d'éléments de **notation graphique**, permettant de modéliser divers aspects tels que les **classes**, les **composants**, les **nœuds**, les **activités**, le **workflow**, les **cas d'utilisation**, les **objets**, et les **états**, ainsi que les relations entre ces éléments. De plus, **UML** permet de personnaliser ces modèles à travers des **extensions personnelles** sous forme d'éléments stéréotypés.

### 2.1.3 Concept et architecture de l'AGSS (Application de MedBot)

Le projet consiste à concevoir une application web avec une architecture 3-tiers. Cette architecture est conçue pour séparer les responsabilités et faciliter la gestion des différentes couches de l'application. L'architecture 3-tiers est une approche bien établie pour le développement d'applications web, assurant la modularité, la scalabilité et la sécurité du système.

- **Client** : Il s'agit de l'interface utilisateur où les utilisateurs interagissent avec l'application via un navigateur web. Cette couche est responsable de l'affichage des données et de la capture des interactions de l'utilisateur.
- **Middleware (Serveur d'application)** : Cette couche sert d'intermédiaire entre le client et le serveur de données. Elle gère la logique métier de l'application, traite les requêtes des utilisateurs, et communique avec la base de données pour récupérer ou sauvegarder des informations.
- **Serveur de données** : Il s'agit de la base de données où toutes les informations nécessaires à l'application sont stockées. Cette couche est responsable de la gestion des données, y compris leur stockage, récupération, et manipulation.



**Figure 2.2:** Architecture 3-tiers de l'AGSS

## 2.2 Conception détaillée

### 2.2.1 Le Diagramme de classe

Le diagramme de classe est un outil fondamental de la modélisation orientée objet. Il permet de représenter de manière structurée les entités principales de l'application ainsi que les relations entre elles. Dans le cadre du développement de notre chatbot médical, ce diagramme illustre les différentes classes impliquées dans le système, telles que les utilisateurs, les conversations, les symptômes, les médicaments, et les médecins. Il facilite la visualisation de la structure de la base de données, des attributs, des méthodes associées à chaque classe, ainsi que des liens d'héritage ou d'association. Ce modèle constitue une étape essentielle pour assurer une conception cohérente, réutilisable et maintenable de l'application avant son implémentation technique.

#### 1. Diagramme de classes du système de chatbot médical

- **AuthUser** : Représente l'utilisateur principal du système.
  - Attributs : `id`, `email`, `display_name`, `phone`
  - Méthodes : `login()`, `logout()`, `resetPassword()`, `updateProfile()`
- **ProfileUser** : Stocke les informations de profil spécifiques de l'utilisateur.
  - Attributs : `user_id`, `full_name_ar`, `image_url`
  - Méthodes : `getProfile()`, `updateImage()`, `updateFullName()`
- **SessionUser** : Gère les sessions de connexion des utilisateurs.
  - Attributs : `user_id`, `session_id`
  - Méthodes : `createSession()`, `updateSession()`, `getSession()`
- **GlobalMessage** : Gère les messages envoyés par les utilisateurs à l'échelle globale (ex. forum).
  - Attributs : `id`, `user_id`, `message`, `created_at`
  - Méthodes : `sendMessage()`, `getAllMessages()`, `getMessagesByUser()`
- **ChatbotInteraction** : Stocke les interactions entre l'utilisateur et le chatbot.
  - Attributs : `id`, `user_id`, `question`, `response`, `created_at`
  - Méthodes : `askQuestion()`, `storeInteraction()`, `getHistoryByUser()`

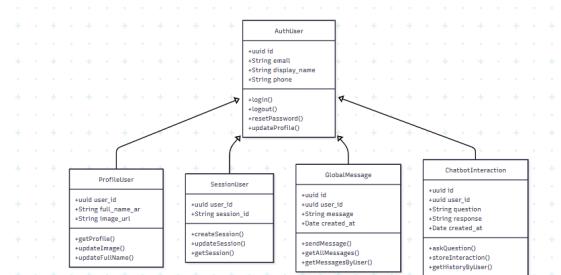


Figure 2.3: Diagramme de classes du système de chatbot médical

---

## 2.2.2 Les diagrammes de cas d'utilisation

L'une des étapes cruciales dans la conception d'un système est la définition des cas d'utilisation fondamentaux. L'objectif principal de cette étape est d'identifier les principaux cas d'utilisation. Dans cette partie, nous nous concentrerons sur la réalisation des diagrammes de cas d'utilisation. Ces diagrammes décrivent de manière précise les besoins du client final et spécifient le comportement attendu du système à développer. En général, un diagramme de cas d'utilisation modélise un service rendu par le système.

### a. Présentation des acteurs

Dans cette section, nous présentons les principaux acteurs impliqués dans le système et leurs rôles respectifs. Deux acteurs principaux sont identifiés : l'utilisateur et le système.

→ **Utilisateur :**

- Se connecter ou créer un compte
- Rechercher un symptôme via une barre de recherche
- Lire un article médical lié à un symptôme
- Poser une question au chatbot
- Recevoir une réponse médicale automatisée
- Rechercher un médicament (usage, effets, posologie)
- Rechercher un médecin par spécialité et localisation
- Accéder à un espace de discussion communautaire

→ **Système (Chatbot médical) :**

- Gérer l'authentification et les sessions utilisateur
- Traiter les requêtes en langage naturel
- Extraire et analyser les entités médicales (symptômes, médicaments, spécialités)
- Générer des réponses personnalisées à partir de la base de données
- Afficher les résultats de recherche (articles, médecins, médicaments)
- Assurer la disponibilité continue du service

### a. Description du cas d'utilisation

Dans cette section, nous présentons les principaux acteurs impliqués dans le système et leurs rôles respectifs.

### b. Diagramme de cas d'utilisation

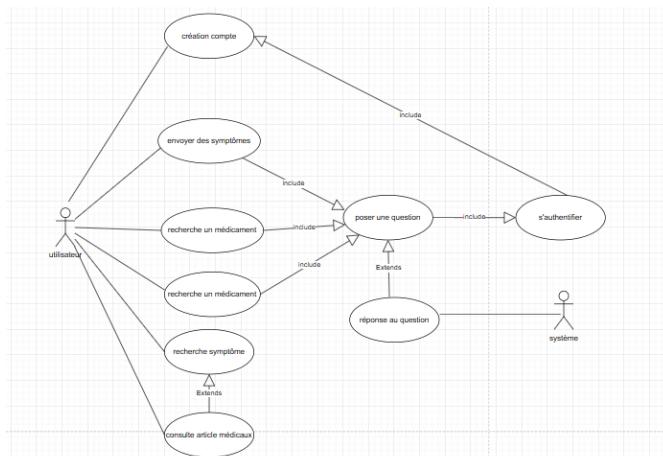
Le diagramme ci-dessous illustre les cas d'utilisation spécifiques à l'utilisateur dans le cadre de l'application de chatbot médical. Ce diagramme offre une vue d'ensemble des interactions entre l'utilisateur et le système, mettant en évidence les différentes fonctionnalités accessibles telles que la recherche de symptômes, la consultation d'articles médicaux, la recherche de médicaments ou de médecins, ainsi que l'interaction directe avec le chatbot pour obtenir des réponses personnalisées. Le système, représenté par le moteur du chatbot, prend en charge le traitement des requêtes, l'analyse sémantique et la génération des réponses à partir de la

---

base de connaissances médicale.

### 1. Diagramme de cas d'utilisation - Utilisateur et Système

- L'utilisateur peut interagir avec le système via différentes fonctionnalités : poser une question, envoyer des symptômes, rechercher un médicament, consulter des articles médicaux, etc.
- La fonctionnalité "poser une question" est centrale et inclut ou étend plusieurs autres cas d'utilisation selon le contexte.
- L'accès aux fonctionnalités principales nécessite l'authentification de l'utilisateur, qui inclut la possibilité de créer un compte.
- Le système est responsable de traiter les requêtes et de générer des réponses adaptées.



**Figure 2.4:** Diagramme de cas d'utilisation de l'utilisateur et du système

### 2.2.3 Diagramme de séquence

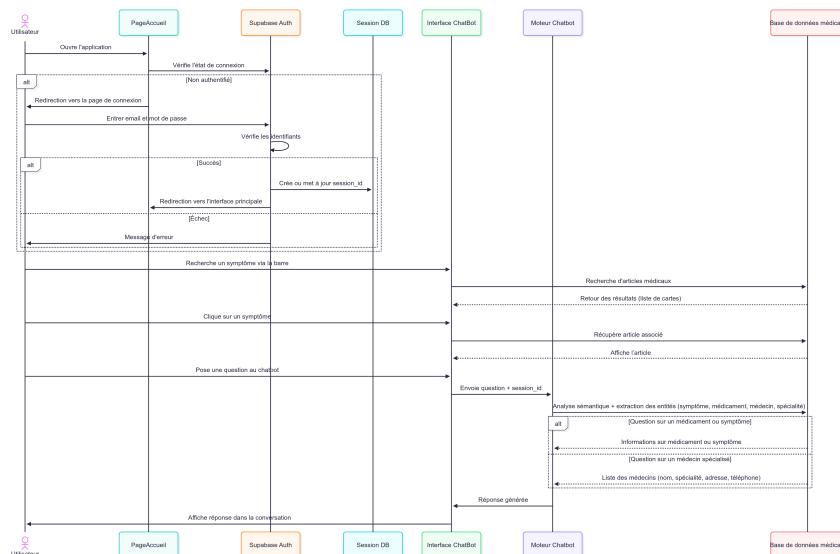
Le diagramme de séquence est un diagramme comportemental de l'UML permettant de modéliser les échanges de messages entre les acteurs (utilisateurs, systèmes) et les composants du système selon une chronologie. Il est particulièrement utile pour illustrer comment les objets interagissent dans un scénario particulier du système, tout en mettant en évidence l'ordre temporel des messages.

#### 1. Diagramme de séquence : Interaction utilisateur avec le chatbot médical

Ce diagramme représente le déroulement typique d'une session d'interaction entre un utilisateur et le système, depuis la saisie d'une question dans l'interface jusqu'à la génération

d'une réponse pertinente par le chatbot, incluant l'accès aux données médicales (symptômes, médicaments ou médecins).

- L'utilisateur saisit une question via l'interface.
- L'interface envoie cette question avec le `session_id` au cœur du chatbot.
- Le chatbot analyse la requête, extrait les entités clés, et interroge la base de données si nécessaire.
- Les informations extraites (ex. détails sur un symptôme ou un médicament) sont utilisées pour formuler une réponse.
- La réponse est affichée à l'utilisateur, éventuellement enrichie avec l'adresse ou le numéro d'un médecin pertinent.



**Figure 2.5:** Diagramme de séquence : Interaction avec le chatbot médical

## 2.3 Conclusion

Ce chapitre se concentre principalement sur l'étude de la conception de l'architecture du système, qui est une étape essentielle pour garantir le bon déroulement de la réalisation du projet. La conception de l'architecture joue un rôle crucial en définissant la structure globale du système, en identifiant les principaux composants et en établissant les relations entre ces différents éléments. En offrant une vue d'ensemble claire et détaillée de la manière dont les différentes parties du système interagissent, cette phase permet de minimiser les ambiguïtés et les incohérences qui pourraient surgir lors du développement.

Dans le cadre de notre projet, l'étude de l'architecture du système couvre plusieurs

aspects clés. Tout d'abord, elle s'attache à définir les composants fondamentaux du système, tels que les modules de gestion des utilisateurs, la gestion des stages, et le suivi des tâches hebdomadaires. Ensuite, elle détaille les interactions entre ces différents modules, afin de s'assurer que chaque composant fonctionne de manière harmonieuse et efficace au sein de l'ensemble.

L'approche architecturale choisie facilite également l'identification et la priorisation des tâches principales, comme la gestion des profils des utilisateurs, l'attribution et le suivi des tâches aux stagiaires, la gestion des rapports de stage et la validation des stages. Ces tâches constituent le cœur fonctionnel du système, et leur conception bien étudiée permet d'assurer une base solide pour le développement et l'intégration ultérieure des fonctionnalités.

En mettant l'accent sur la conception de l'architecture, ce chapitre illustre comment une planification soigneuse et une analyse approfondie de la structure du système peuvent simplifier la phase de réalisation. Une architecture bien conçue facilite non seulement le développement de chaque composant de manière modulaire, mais elle permet également de s'adapter aux évolutions futures et aux besoins émergents, assurant ainsi la pérennité et la flexibilité du système.

# Chapter 3 Réalisation

---

## Chapter Contents

3.1	Introduction aux Technologies Utilisées . . . . .	25
3.1.1	React.js . . . . .	25
3.2	Utilisation de Dialogflow pour l'intelligence conversationnelle . . . . .	25
3.2.1	Présentation de Dialogflow : . . . . .	26
3.2.2	Rôle dans les systèmes de chatbot intelligents : . . . . .	26
3.2.3	Versions disponibles : Dialogflow ES vs CX : . . . . .	26
3.2.4	Pourquoi le choix de Dialogflow pour ce projet ? . . . . .	26
3.2.5	Structure d'un agent Dialogflow . . . . .	27
3.2.6	Contextes conversationnels . . . . .	27
3.2.7	Création des intents . . . . .	27
3.2.8	Définition des paramètres et entités . . . . .	28
3.2.9	Base de données pour les réponses dynamiques . . . . .	30
3.3	Mise en place du backend avec Flask . . . . .	31
3.3.1	Pourquoi utiliser Flask ? . . . . .	32
3.3.2	Étapes pour créer un projet Flask : . . . . .	32
3.3.3	Connexion entre Dialogflow et le backend Flask via Ngrok . . . . .	32
3.3.4	Connexion du backend Flask à la base de données Supabase . . . . .	33
3.3.5	Illustration de la Structure Technologique . . . . .	34
3.4	Page d'Accueil . . . . .	36
3.4.1	Page d'Accueil . . . . .	36
3.4.2	À propos de notre solution . . . . .	36
3.4.3	Services . . . . .	37
3.4.4	Contact . . . . .	37
3.4.5	Connexion et inscription des utilisateurs . . . . .	38
3.5	Navigation de l'application . . . . .	38
3.5.1	Accueil . . . . .	39
3.5.2	Chatbot médical . . . . .	39
3.5.3	Articles Médicaux . . . . .	41
3.5.4	Médicaments . . . . .	41
3.5.5	Discussion . . . . .	42
3.6	Conclusion sur l'intégration du chatbot médical . . . . .	42

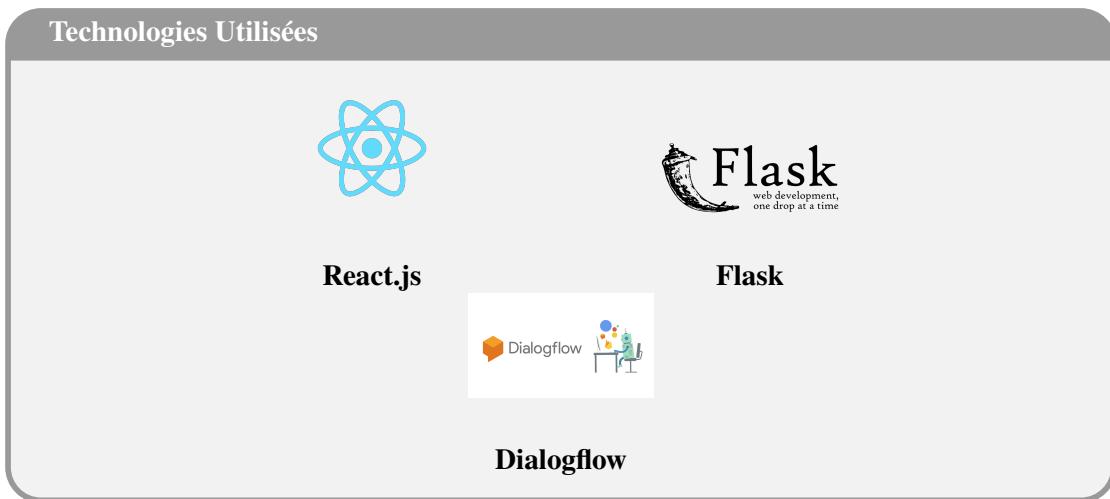
---

## **Abstract**

---

La partie de réalisation est une étape cruciale réservée au développement concret du système étudié. Elle constitue le cœur du projet, où les concepts théoriques et les spécifications fonctionnelles sont transformés en une application opérationnelle. Cette phase implique l'utilisation de technologies adaptées et la mise en œuvre de fonctionnalités clés pour répondre aux besoins définis, garantissant ainsi la réussite et la robustesse de l'ensemble du système.

### 3.1 Introduction aux Technologies Utilisées



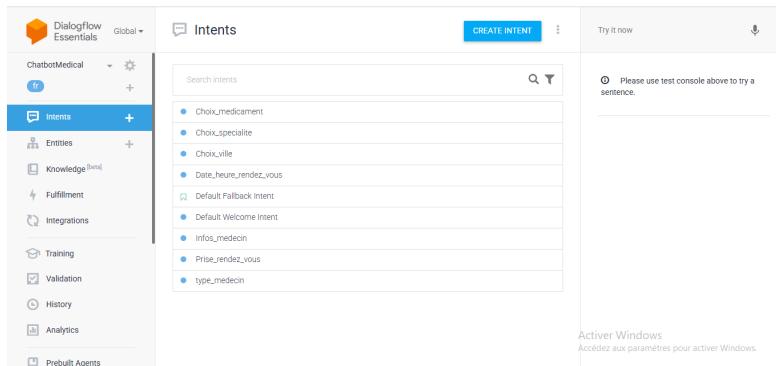
#### 3.1.1 React.js

React.js est une bibliothèque JavaScript dédiée à la création d'interfaces utilisateur. Elle est particulièrement adaptée pour le développement de composants front-end modernes et dynamiques. Voici quelques-uns de ses avantages clés :

- **Gestion efficace de l'état** : React utilise un système de gestion d'état (comme `useState` et `useReducer`) permettant de manipuler l'état de l'application de manière performante et organisée.
- **Composants réutilisables** : La logique de développement orientée composants permet de créer des éléments modulaires et réutilisables à travers l'application, facilitant ainsi la maintenance et l'évolution du code.
- **Interfaces utilisateur interactives et réactives** : Grâce à son approche basée sur le DOM virtuel, React assure des mises à jour rapides et optimisées de l'interface utilisateur, offrant une expérience fluide à l'utilisateur.
- **Écosystème riche** : React dispose d'une vaste communauté et d'un écosystème de bibliothèques et d'outils (comme React Router, Redux) qui simplifient la gestion du routage, de l'état global, et d'autres fonctionnalités avancées.

### 3.2 Utilisation de Dialogflow pour l'intelligence conversationnelle

Cette section présente en détail l'intégration de la plateforme Dialogflow dans notre système, en expliquant son fonctionnement, ses avantages, ainsi que sa connexion avec notre backend Flask via Ngrok.



**Figure 3.1:** Carte Mentale : Exemple de cohérence visuelle

### 3.2.1 Présentation de Dialogflow :

Dialogflow est une plateforme développée par Google permettant de concevoir des interfaces de conversation homme-machine, appelées aussi agents conversationnels ou "chatbots". Elle repose sur des modèles de traitement du langage naturel (NLP) afin de comprendre les requêtes des utilisateurs et d'y répondre de manière contextuelle.

### 3.2.2 Rôle dans les systèmes de chatbot intelligents :

Dans notre projet, Dialogflow joue un rôle central pour comprendre les questions de l'utilisateur, identifier les intentions (intents) et extraire les entités (paramètres) clés comme le nom d'un médicament, un symptôme ou une spécialité médicale. Cela permet au système de répondre avec précision ou de déclencher des actions via le backend.

### 3.2.3 Versions disponibles : Dialogflow ES vs CX :

Dialogflow existe en deux versions :

- **Dialogflow ES (Essentials)** : plus simple, adapté aux projets avec une logique conversationnelle linéaire.
- **Dialogflow CX (Customer Experience)** : plus avancé, orienté entreprise, adapté aux projets à plusieurs flux complexes.

Dans ce projet, nous avons utilisé **Dialogflow ES** pour sa simplicité d'intégration avec un backend Flask.

### 3.2.4 Pourquoi le choix de Dialogflow pour ce projet ?

Dialogflow a été choisi pour plusieurs raisons :

- Il offre une interface intuitive pour créer des intents et des entités.
- Il permet une intégration fluide avec un backend personnalisé via un webhook HTTP.

- Il intègre un puissant moteur NLP pour comprendre les requêtes variées des utilisateurs, même avec des synonymes ou des fautes mineures.

Ce choix a facilité le développement rapide d'un agent intelligent capable de répondre aux besoins spécifiques du domaine médical.

### 3.2.5 Structure d'un agent Dialogflow

L'architecture de base d'un agent Dialogflow repose sur les composants suivants :

- Intents** : Représentent les objectifs ou intentions de l'utilisateur.
- Entités** : Permettent d'extraire des informations spécifiques à partir des phrases.
- Contextes** : Servent à maintenir l'état de la conversation pour garantir une continuité logique des échanges.

**Illustration de l'interface des intents utilisés :**

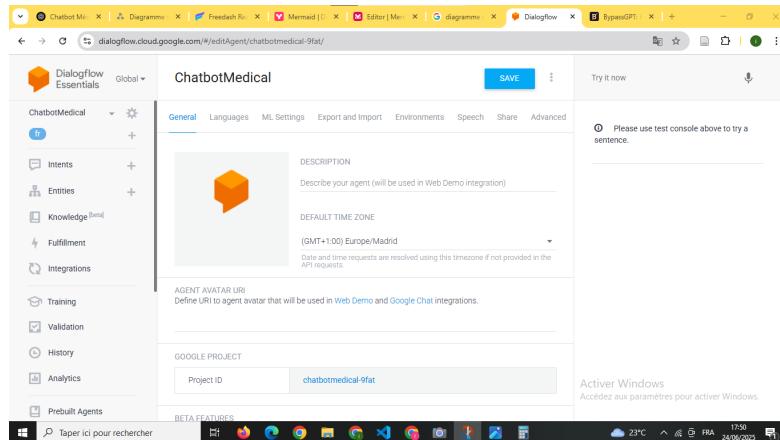


Figure 3.2: Exemple d'intents configurés dans Dialogflow

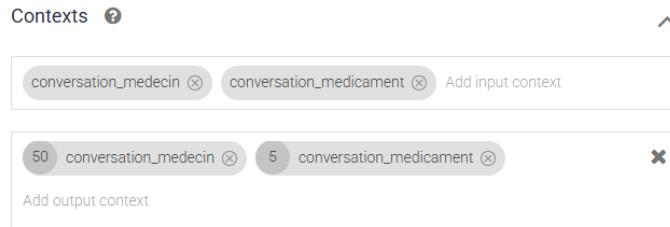
### 3.2.6 Contextes conversationnels

Les contextes sont utilisés pour suivre l'état d'une conversation et adapter les réponses selon la progression de l'échange. Voici deux contextes essentiels définis dans notre projet :

- conversation\_médecin** : Conserve les informations liées aux demandes concernant les médecins (ex. spécialité, ville).
- conversation\_médicament** : Conserve le contexte lorsqu'un médicament est mentionné pour obtenir sa composition, son prix ou ses indications.

### 3.2.7 Crédit des intents

Un **intent** (ou intention) dans Dialogflow représente un objectif spécifique de l'utilisateur, que ce soit une question ou une action. Il permet de capter une intention à partir de phrases formulées de différentes manières par l'utilisateur.



**Figure 3.3:** Exemple d'intents configurés dans Dialogflow

Chaque intent est constitué de trois parties principales :

- **Phrases d'entraînement** : Ce sont des exemples de formulations possibles que l'utilisateur pourrait dire (ex : « Je cherche un neurologue à Rabat », « Quel est le prix de Paracétamol ? »).
- **Paramètres / Entités** : Permettent d'extraire des informations clés de la phrase (comme @symptome, @nom\_medicament, @ville, etc.).
- **Réponse** : Il peut s'agir d'un texte statique, d'une réponse générée dynamiquement via un Webhook, ou d'une redirection vers une autre action.

#### **Exemples de phrases d'entraînement :**

- « Donne-moi les effets secondaires du Doliprane »
- « Est-ce que je dois consulter un spécialiste pour ce symptôme ? »
- « Je veux voir un médecin à Casablanca »

#### **Gestion des réponses par défaut :**

Quand aucun paramètre n'est détecté ou quand la requête est floue, Dialogflow peut activer une réponse par défaut ou passer le relais au Webhook pour une meilleure interprétation via GPT ou traitement métier.

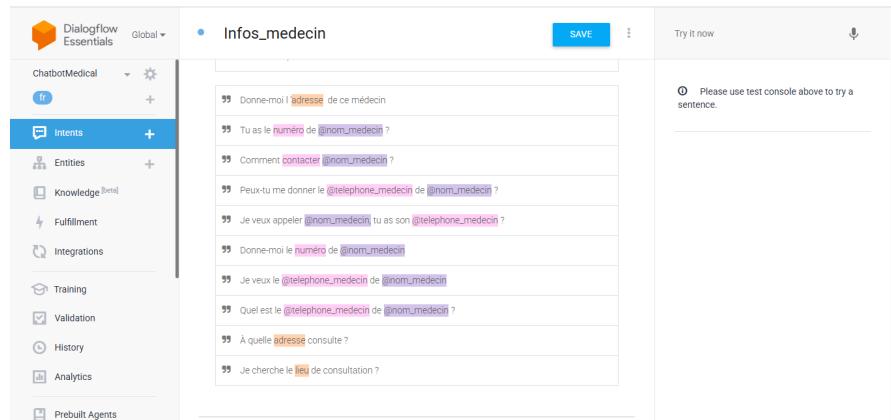
#### **Lien avec la logique métier :**

Chaque intent peut être relié à une fonction Python dans le backend Flask, selon le nom de l'intent (ex : Infos\_medecin → handle\_infos\_medecin()), assurant un mappage clair entre le langage naturel et les actions techniques.

#### **Illustration de la création des intents dans Dialogflow :**

### **3.2.8 Définition des paramètres et entités**

Les paramètres et entités jouent un rôle central dans la compréhension des requêtes utilisateur par l'agent Dialogflow. Ils permettent d'extraire des informations précises à partir d'une phrase naturelle.



**Figure 3.4:** Interface de création des intents dans Dialogflow

## Qu'est-ce qu'un paramètre ?

Un paramètre représente une donnée extraite automatiquement par l'agent à partir d'un mot ou d'une expression, grâce à la détection d'une entité associée. Par exemple, dans « Donne-moi les effets du *Doliprane* », le mot *Doliprane* sera reconnu comme un paramètre appartenant à l'entité @nom\_medicament.

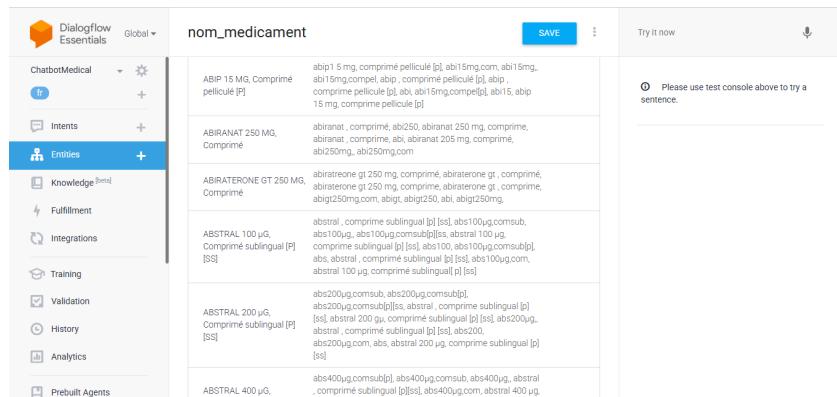
## **Création d'entités personnalisées**

Dialogflow permet la création d'entités personnalisées pour détecter des types de données spécifiques à une application. Par exemple, on peut créer une entité @nom\_medicament contenant les noms des médicaments courants utilisés dans le projet.

## Ajout de synonymes pour une meilleure compréhension

Chaque valeur d'une entité peut être enrichie de synonymes afin d'augmenter la flexibilité de reconnaissance. Par exemple, l'entité @nom\_medicament peut contenir la valeur *Paracétamol* avec les synonymes : *Doliprane, Efferalgan*, etc.

**Exemple : entité nom\_medicament avec valeurs et variantes**



**Figure 3.5:** Exemple d'entité nom\_medicament avec variantes

### 3.2.9 Base de données pour les réponses dynamiques

Certaines réponses du chatbot, notamment celles concernant les informations sur les médecins et les médicaments, nécessitent l'utilisation d'une base de données structurée et à jour. Pour alimenter ces données, nous avons développé un script de **web scraping** utilisant Selenium et ChromeDriver.

#### Exemple de code de scraping des données :

```
# ☑ 1. Fonction pour générer dynamiquement l'URL
def generate_dababoc_url(speciality_id: str, city_id: str, country_code: str = "MA") -> str:
    return (
        f"https://www.dababoc.com/search?"
        f"&country={country_code}"
        f"&search[doctor_speciality_id]={speciality_id}"
        f"&search[type]=false"
        f"&search[booking_type]=0"
        f"&search[city_id]={city_id}"
        f"&button=par"
    )

# ☑ 2. Configuration Selenium ChromeDriver
chromedriver_path = r"E:\chromedriver-win64\chromedriver.exe" # À adapter selon ton PC

options = Options()
# options.add_argument("--headless") # Activer si tu veux sans interface graphique
options.add_argument("--no-sandbox")
options.add_argument("--disable-popup-blocking")
options.add_experimental_option("prefs", {
    "protocol_handler.excluded_schemes": {"tel": False}
})

driver = webdriver.Chrome(service=Service(chromedriver_path), options=options)

# ☑ 3. Paramètres (spécialité : Cardiologue, ville : Kénitra)
speciality_id = "51d6e1f4ef96750d4d00000d"
city_id = "5225c16122814f1518000005"

url = generate_dababoc_url(speciality_id, city_id)
driver.get(url)
```

Figure 3.6: Exemple de script de scraping médecins utilisant Selenium et ChromeDriver

```
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options

# ☑ Chemin vers chromedriver.exe
chromedriver_path = r"E:\chromedriver-win64\chromedriver-win64\chromedriver.exe"

# ☑ Configuration du navigateur Chrome (en mode invisible)
options = Options()
#options.add_argument("--headless") # Enlève cette ligne si tu veux voir la fenêtre
driver = webdriver.Chrome(service=Service(chromedriver_path), options=options)

# ☑ Base URL par lettre
base_url = "https://medicament.ma/listing-des-medicaments/?lettre={}"

# ☑ Créer le dossier de sortie
output_folder = "medicaments_csv"
os.makedirs(output_folder, exist_ok=True)

# ☑ Fonction pour extraire les liens des médicaments depuis la page
def get_med_links():
    meds = []
    items = driver.find_elements(By.CSS_SELECTOR, "table.table tbody tr td a")
    for item in items:
        try:
            nom = item.find_element(By.CLASS_NAME, "details").text.split("\n")[0].strip()
            desc = item.find_element(By.CLASS_NAME, "small").text.strip()
            href = item.get_attribute("href")
            meds.append((nom, desc, href))
        except:
            continue
    return meds
```

Figure 3.7: Exemple de script de scraping médecins utilisant Selenium et ChromeDriver

---

### Illustration de la structure de données JSON pour les médecins :

```
 "Dr Hakkou El Mehdi": {
    "specialite": "neurologue",
    "ville": "Rabat",
    "description": "Neurologue à Rabat",
    "adresse": "Adresse inconnue",
    "telephone": 212655712955
},
"Professeur Boutaleb": {
    "specialite": "neurologue",
    "ville": "Rabat",
    "description": "Neurologue à Rabat",
    "adresse": "Residence El Ghafiki, 10 belle résidence A, avenue abderahman",
    "telephone": 212537774073
},
"Dr Oudrhiri Mohammed yassaad": {
    "specialite": "neurologue",
    "ville": "Rabat",
    "description": "Neurologue à Rabat",
    "adresse": "Hôpital des Spécialités de Rabat, Avenue Hafiane Cherkaoui BP6220 Rabat Instituts",
    "telephone": 212666132472
},
"Dr Bouchra Benaaboud": {
    "specialite": "neurologue",
    "ville": "Rabat",
    "description": "Neurologue à Rabat",
    "adresse": "Rés Oum El Kheir 1 & 2 App10 2 ème Etage Av Mohamed V Tabriquet",
    "telephone": 212537861086
},
```

**Figure 3.8:** Structure JSON des données des médecins

### Illustration de la structure de données JSON pour les médicaments :

```
{
    "ABIP 15 MG, Comprimé pelliculé [P)": {
        "description": "Boîte de 20 - PPV: 268.00 dhs",
        "lien": "https://medicament.ma/medicament/abip-15-mg-comprime-pellicule/",
        "présentation": "Boîte de 20",
        "dosage": "15 MG",
        "distributeur": "",
        "composition": "Aripiprazole",
        "classe_therapeutique": "Neuroleptique atypique",
        "statut": "Commercialisé",
        "code_atc": "N05AX12",
        "ppv": "268.00 dhs",
        "prix_hospitalier": "168.00 dhs",
        "tableau": "A",
        "indications": "L'aripiprazole est indiqué dans le traitement de la schizophrénie chez les adultes et enfants."
    },
    "ABIRANAT 250 MG, Comprimé": {
        "description": "Boîte d'un flacon de 120 - PPV: 10411.00 dhs",
        "lien": "https://medicament.ma/medicament/abiranat-250-mg-comprime/",
        "présentation": "Boîte d'un flacon de 120",
        "dosage": "250 MG",
        "distributeur": "",
        "composition": "Abiraterone",
        "classe_therapeutique": "Endocrinothérapie, autres antagonistes hormonaux et agents apparentés",
        "statut": "Commercialisé",
        "code_atc": "L02BX03",
        "ppv": "10411.00 dhs",
        "prix_hospitalier": "10215.00 dhs",
        "tableau": "A",
        "indications": "Indiqué en association avec la prednisone ou la prednisolone dans :n- le traitement de l'hyperplasie prostatique bénigne"
    }
},
```

**Figure 3.9:** Structure JSON des données des médicaments

Ces données alimentent la base de connaissances du chatbot pour fournir des réponses précises et fiables aux utilisateurs concernant les professionnels de santé et les médicaments.

### 3.3 Mise en place du backend avec Flask

**Flask** est un micro-framework web léger écrit en Python. Il permet de créer des API web rapidement et efficacement, tout en gardant une structure simple et modulaire.

---

### 3.3.1 Pourquoi utiliser Flask ?

- Léger, flexible et facile à configurer
- Intégration facile avec des librairies Python (comme celles pour l'IA ou la base de données)
- Idéal pour créer des API REST utilisées par les chatbots ou les applications mobiles/web

### 3.3.2 Étapes pour créer un projet Flask :

```
C:\Users\hp>pip install flask
```

Figure 3.10: Étapes pour mettre en place un projet Flask

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return "Bienvenue sur le backend Flask"

if __name__ == '__main__':
    app.run(port=5000, debug=True)
```

Figure 3.11: Créer un fichier app.py de base

Dans la prochaine section, nous expliquerons comment ce backend local peut être exposé à Dialogflow à l'aide de l'outil **Ngrok**.

### 3.3.3 Connexion entre Dialogflow et le backend Flask via Ngrok

Pour connecter notre serveur Flask local à Dialogflow, nous avons utilisé **Ngrok**, un outil permettant d'exposer un port local (comme 5000) à internet via une URL sécurisée.

- **Ngrok** joue le rôle d'intermédiaire entre Dialogflow (en ligne) et le serveur Flask (en local).
- Il génère une URL publique temporaire pointant vers localhost:5000.

**Commandes à utiliser :**

```
ngrok http 5000
```

Cette commande fournit une URL comme : <https://b7a2-154-124-XX-XX.ngrok.io>

#### Configuration côté Dialogflow :

1. Aller dans l'onglet **Fulfillment**.
2. Activer Webhook.
3. Coller l'URL Ngrok suivie du chemin défini dans app.py, par exemple :  
<https://550a-196-89-141-28.ngrok-free.app/webhook>
4. Sauvegarder les modifications.

#### Côté Flask, la route webhook ressemble à ceci :

```
@app.route('/webhook', methods=['POST'])
def webhook():

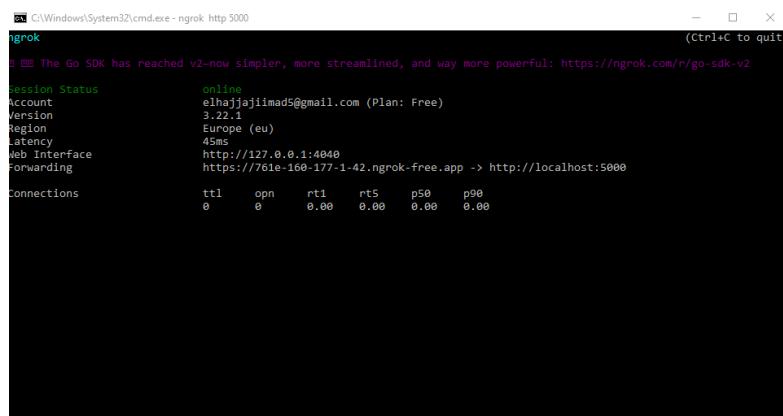
    req = request.get_json()

    intent = req['queryResult']['intent']['displayName']

    if intent == "Infos_medecin":

        # Logique mtier
        return jsonify({"fulfillmentText": "Voici les infos du medecin..."})
```

#### Illustration de la liaison Flask Dialogflow via Ngrok :



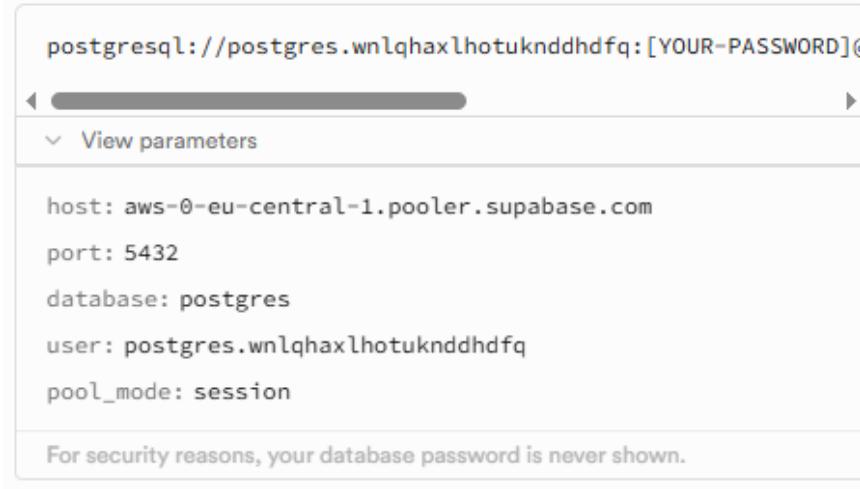
```
PS C:\Windows\System32\cmd.exe - ngrok http 5000
ngrok
000 The Go SDK has reached v2-now simpler, more streamlined, and way more powerful: https://ngrok.com/r/go-sdk-v2
Session Status          online
Account                 elha@jajimad5@gmail.com (Plan: Free)
Version                3.22.1
Region                 Europe (eu)
Latency                45ms
Web Interface          http://127.0.0.1:4040
Forwarding             https://61e-160-177-1-42.ngrok-free.app -> http://localhost:5000

Connections            ttl     opn     rt1     rt5     p50     p90
                        0       0       0.00   0.00   0.00   0.00
```

Figure 3.12: Connexion de Dialogflow à Flask via Ngrok

### 3.3.4 Connexion du backend Flask à la base de données Supabase

Pour gérer le stockage des utilisateurs, des sessions et des historiques de questions/réponses, nous avons choisi d'utiliser **Supabase**, une plateforme open-source équivalente à Firebase, basée sur **PostgreSQL**.



**Figure 3.13:** les paramètres de connection supabase par flask

### Pourquoi Supabase ?

- Facile à connecter avec Flask via des requêtes HTTP (RESTful).
- Interface graphique pour visualiser les données.
- API instantanée sécurisée via des clés (API Keys).
- Authentification intégrée et gestion des utilisateurs.

### Connexion avec la bibliothèque requests :

```
import requests

def get_user_by_email(email):
    url = f"{SUPABASE_URL}/rest/v1/users?email={email}"
    headers = {
        "apikey": SUPABASE_KEY,
        "Authorization": f"Bearer {SUPABASE_KEY}"
    }
    response = requests.get(url, headers=headers)
    return response.json()
```

### Illustration de l'architecture : Supabase

#### 3.3.5 Illustration de la Structure Technologique

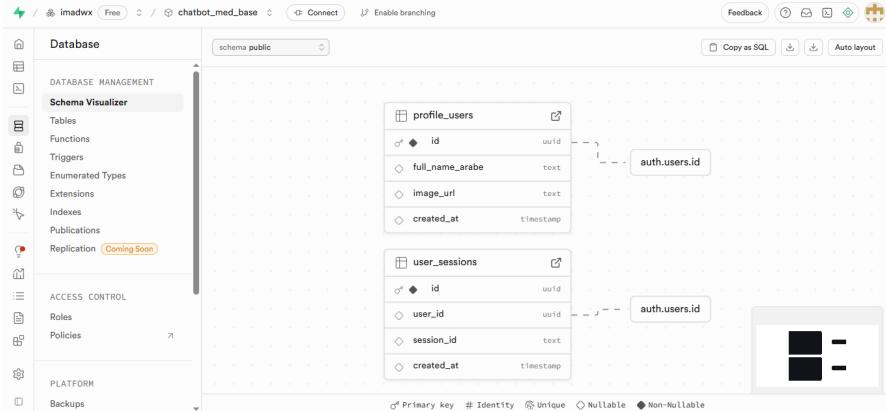


Figure 3.14: la base de données Supabase

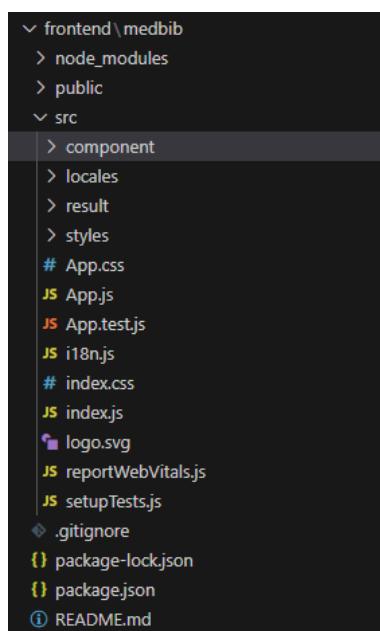


Figure 3.15: l'application Frontend

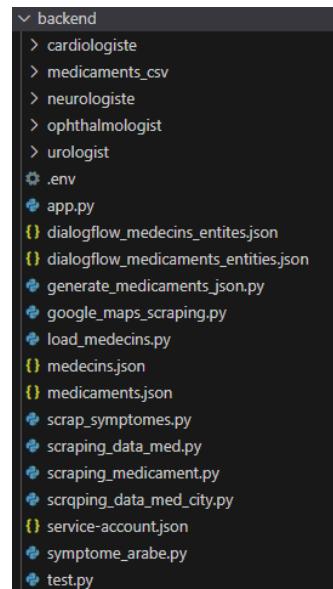


Figure 3.16: l'application Backend

Les deux applications composant notre système sont soigneusement structurées pour répondre à des besoins distincts mais complémentaires. La première application, dédiée à l’interface utilisateur, est construite avec React.js et se concentre sur l’expérience utilisateur. Elle est conçue pour offrir une interface intuitive et réactive, facilitant la navigation et l’interaction des utilisateurs avec le système. Cette application gère l’affichage des données, la soumission des formulaires et l’interaction directe avec les utilisateurs finaux.

La seconde application, développée en Django, représente la partie administration et backend du système. Elle s’occupe de la gestion des données, de l’authentification des utilisateurs et de l’exécution des opérations métier. Ce backend robuste est responsable de la validation des données, de la gestion des utilisateurs et des tâches, ainsi que de la communication avec la base de données.

Pour assurer une communication fluide entre ces deux applications, nous utilisons des API bien définies. Ces API permettent d’envoyer et de recevoir des données entre le frontend et le

---

backend, assurant ainsi une synchronisation efficace et une intégrité des données à travers le système. Le frontend envoie des requêtes HTTP au backend pour récupérer ou soumettre des informations, tandis que le backend traite ces requêtes, met à jour la base de données et renvoie les réponses appropriées.

Cette architecture en deux parties garantit non seulement une séparation claire des responsabilités mais aussi une flexibilité et une évolutivité dans le développement et la maintenance du système.

## 3.4 Page d'Accueil

### 3.4.1 Page d'Accueil

La page d'accueil de notre application est conçue pour offrir une vue d'ensemble intuitive et engageante aux utilisateurs. Elle sert de point d'entrée principal vers les différentes fonctionnalités et sections de l'application. Voici une représentation visuelle de la page d'accueil :

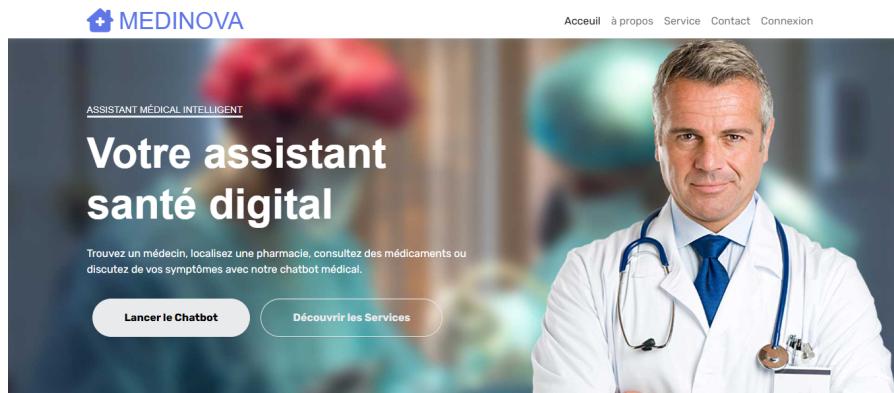
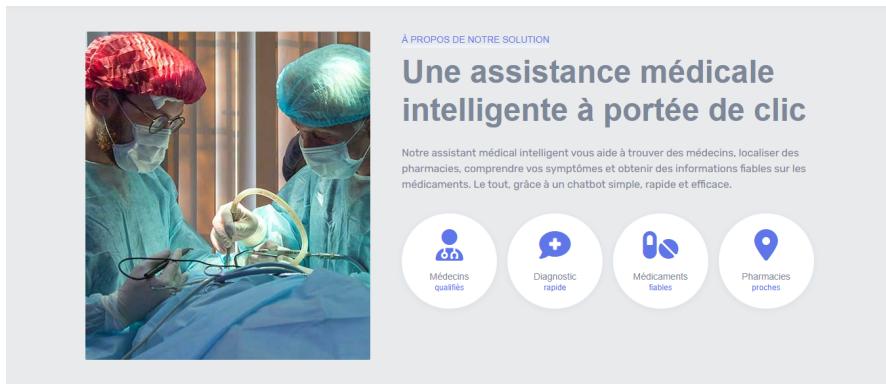


Figure 3.17: Page d'Accueil de l'application

### 3.4.2 À propos de notre solution

Notre solution propose une assistance médicale intelligente permettant aux utilisateurs de trouver des médecins qualifiés, localiser des pharmacies proches, comprendre leurs symptômes et obtenir des informations fiables sur les médicaments. Le tout grâce à un chatbot simple, rapide et efficace qui centralise ces fonctionnalités dans une interface conviviale.

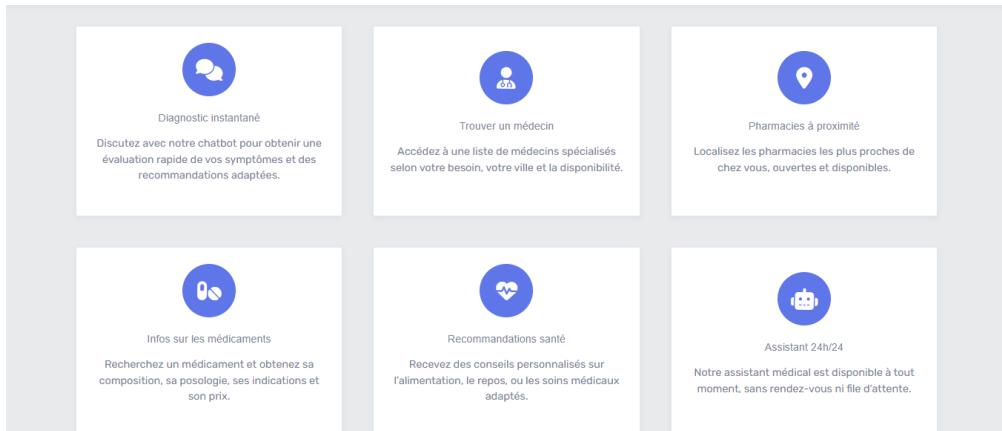


**Figure 3.18:** À propos solution médicale intelligente

### 3.4.3 Services

Notre application propose plusieurs services essentiels pour accompagner les utilisateurs dans leur parcours de santé. Parmi ces services :

- Diagnostic instantané des symptômes via un chatbot intelligent
- Recherche et informations détaillées sur les médicaments
- Trouver un médecin selon la spécialité, la ville et la disponibilité
- Localisation des pharmacies proches
- Recommandations santé personnalisées
- Un assistant médical disponible 24h/24



**Figure 3.19:** Aperçu des services proposés par l'application

### 3.4.4 Contact

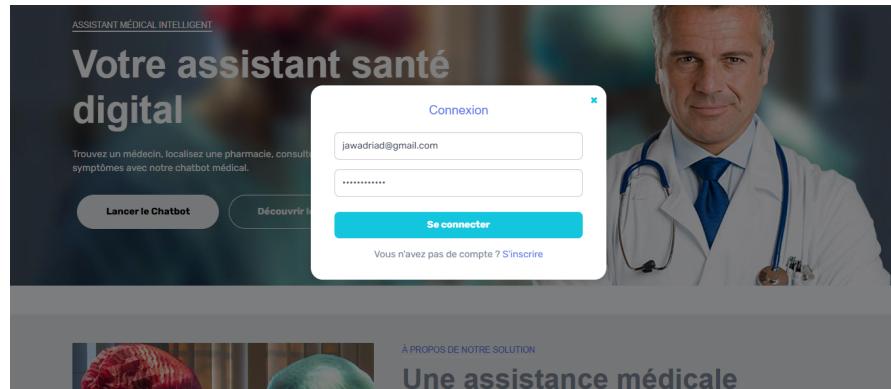
La page de contact permet aux utilisateurs de nous envoyer des messages, poser des questions ou soumettre des retours concernant l'application. Elle comprend un formulaire simple (nom, email, message) pour faciliter la prise de contact.

**Figure 3.20:** Formulaire de contact de l’application

### 3.4.5 Connexion et inscription des utilisateurs

L’application propose un modal dédié permettant aux utilisateurs de se connecter ou de s’inscrire facilement. Ce modal assure une authentification sécurisée et rapide, offrant une expérience utilisateur fluide dès la première utilisation.

- **Connexion** : les utilisateurs existants peuvent saisir leur email et mot de passe pour accéder aux services proposés.
- **Inscription** : les nouveaux utilisateurs peuvent créer un compte en fournissant les informations nécessaires (nom, email, mot de passe).



**Figure 3.21:** Modal de connexion et d’inscription des utilisateurs

## 3.5 Navigation de l’application

Cette section décrit la structure de navigation de l’application, plus particulièrement les pages accessibles depuis la barre latérale gauche (*left sidebar*). L’utilisateur peut accéder facilement aux différentes fonctionnalités proposées.

### 3.5.1 Accueil

La page d'accueil de l'application présente un aperçu global du système et contient une vidéo démonstrative qui explique le fonctionnement de l'application ainsi que les principales fonctionnalités offertes par le chatbot médical. Cette vidéo permet aux utilisateurs de comprendre rapidement comment interagir avec le bot, rechercher des informations médicales ou poser des questions sur leurs symptômes.

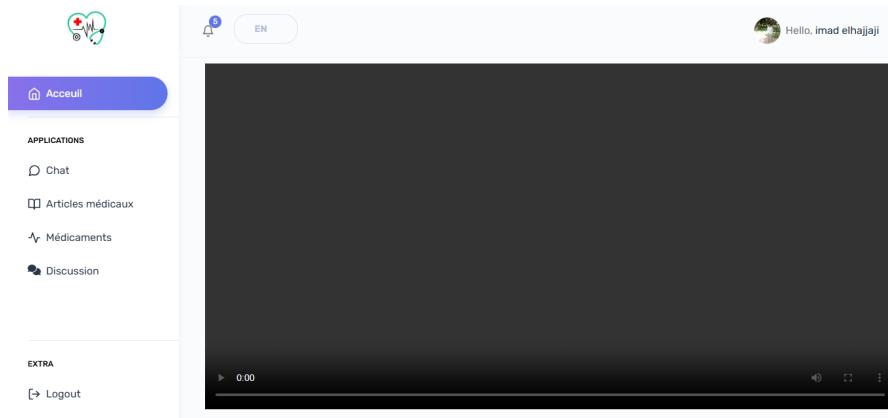


Figure 3.22: Vidéo démonstrative de l'application et des fonctionnalités du chatbot

### 3.5.2 Chatbot médical

Cette section de l'application permet aux utilisateurs d'interagir directement avec le chatbot médical. L'utilisateur peut poser des questions concernant ses symptômes, rechercher des médecins ou des informations sur des médicaments, et recevoir des réponses personnalisées générées par le système.

Le chatbot guide la conversation, extrait les paramètres importants (ville, spécialité, nom de médicament, etc.) et fournit des recommandations ou des résultats pertinents.

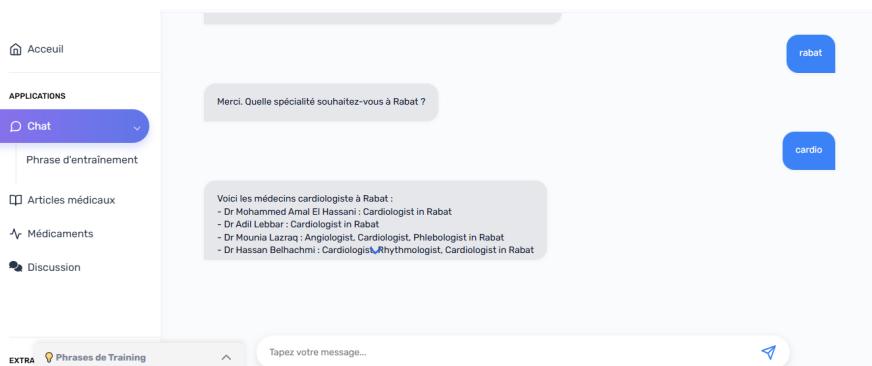
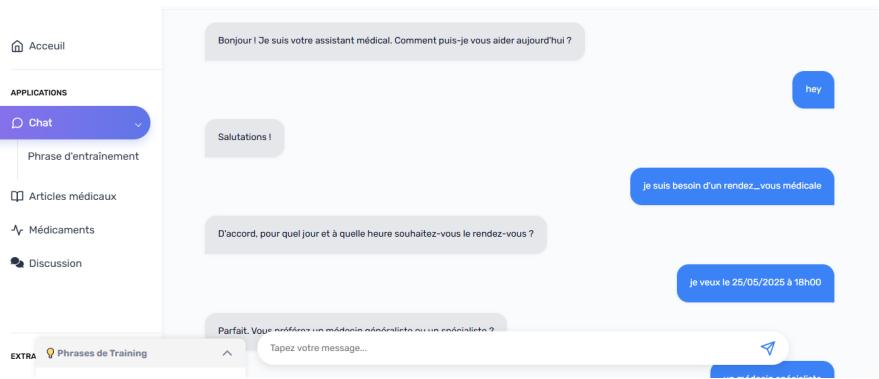
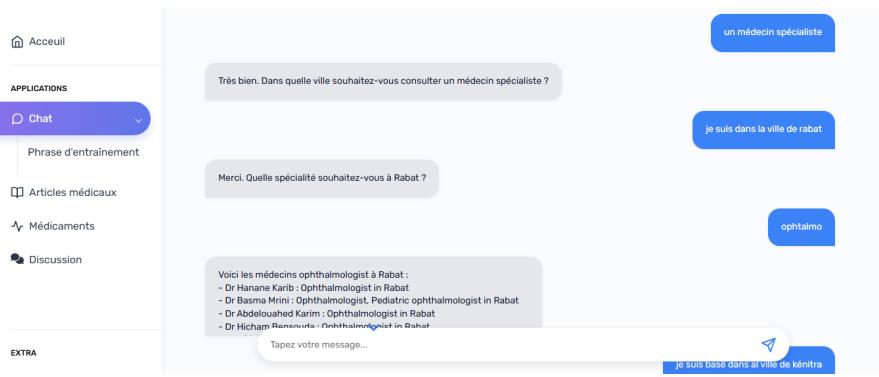


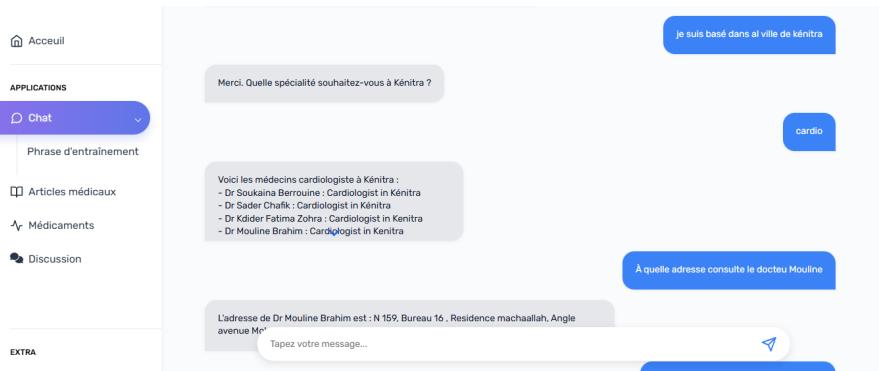
Figure 3.23: Conversation avec le chatbot médical - Exemple 1



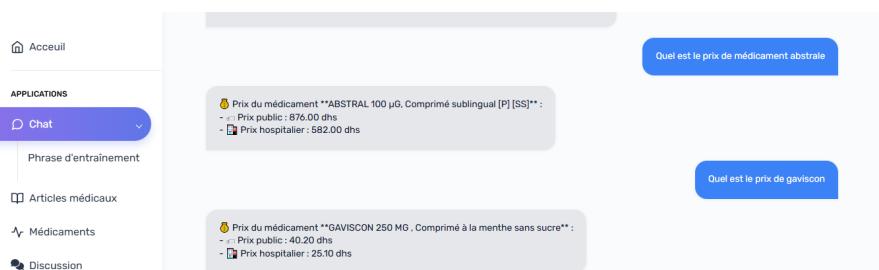
**Figure 3.24:** Conversation avec le chatbot médical - Exemple 2



**Figure 3.25:** Conversation avec le chatbot médical - Exemple 3



**Figure 3.26:** Conversation avec le chatbot médical - Exemple 4



**Figure 3.27:** Conversation avec le chatbot médical - Exemple 5

### 3.5.3 Articles Médicaux

Cette section permet aux utilisateurs de rechercher des informations médicales fiables sur divers symptômes. Ils peuvent saisir un mot-clé dans la barre de recherche pour filtrer rapidement les résultats, puis consulter des articles détaillés pour mieux comprendre leurs symptômes et les démarches à suivre.

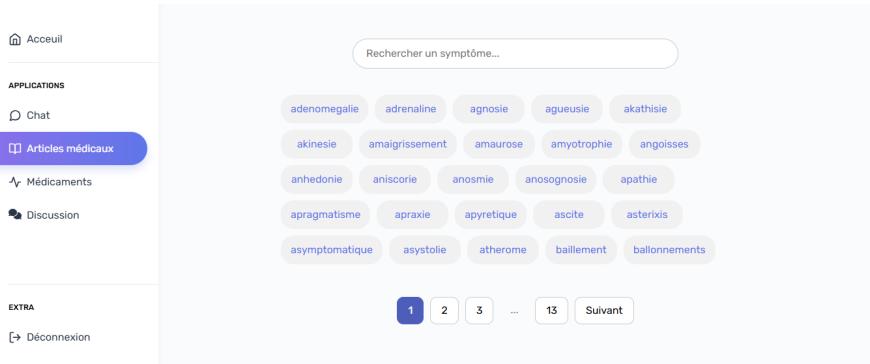


Figure 3.28: Articles Médicaux avec la recherche de symptômes

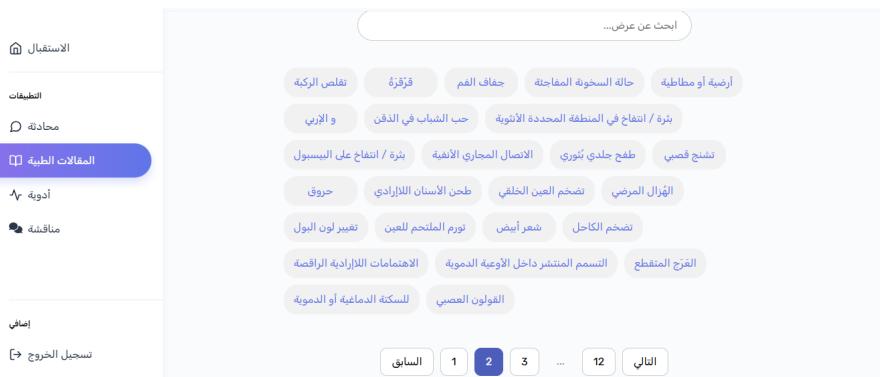


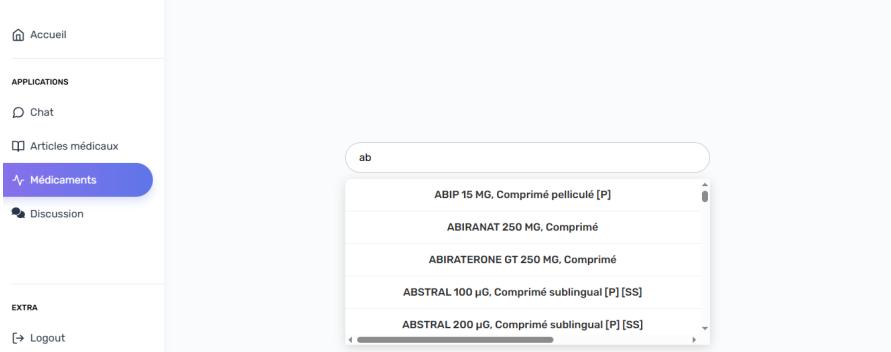
Figure 3.29: Articles Médicaux avec la recherche de symptômes en arabe

### 3.5.4 Médicaments

Cette section offre à l'utilisateur la possibilité de rechercher un médicament par son nom. L'application fournit alors des informations détaillées telles que :

- La composition du médicament
- Sa posologie
- Les indications thérapeutiques
- Les prix (public et hospitalier)

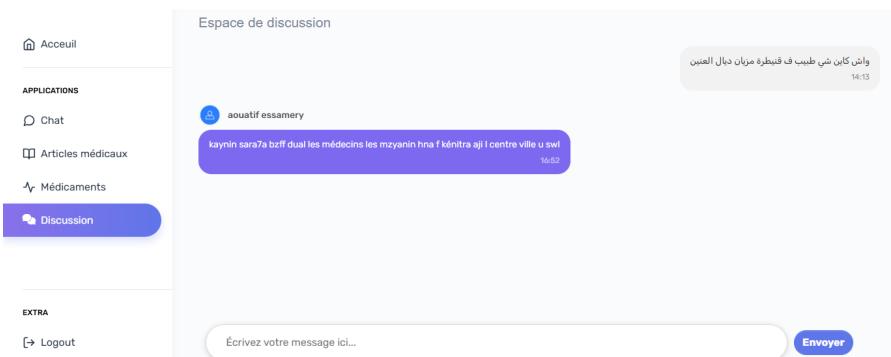
L'utilisateur peut ainsi accéder facilement à des données médicales précises pour mieux comprendre l'usage du médicament.



**Figure 3.30:** Affichage d'informations sur les médicaments à partir du chatbot

### 3.5.5 Discussion

Cette section permet aux utilisateurs d'échanger et de discuter librement de leurs préoccupations médicales. Ils peuvent poser des questions, partager leurs expériences ou demander des conseils auprès de la communauté ou des professionnels, tout en conservant l'historique de leurs conversations.



**Figure 3.31:** Interface de la section Discussion de l'application

## 3.6 Conclusion sur l'intégration du chatbot médical

L'intégration du chatbot médical dans notre application représente une avancée majeure pour améliorer l'accès aux informations de santé, l'orientation vers les professionnels et la compréhension des symptômes par les utilisateurs. Ce projet permet d'offrir une assistance rapide, continue et personnalisée, grâce à une interface conviviale et à un système intelligent de traitement des requêtes. À l'avenir, l'amélioration continue des performances du chatbot et l'enrichissement de la base de données permettront de répondre de manière encore plus précise et fiable aux besoins des utilisateurs.

## Bibliography

---

- [1] **Bird, Steven, Ewan Klein, and Edward Loper**, “Natural Language Processing with Python,” *O'Reilly Media*, 2009. Référence classique pour le NLP avec Python et NLTK.
- [2] **FastAPI Community**, “FastAPI: High performance, easy to learn, fast to code, ready for production,” *FastAPI Official Documentation*, 2023. Framework moderne pour créer des API backend, alternative légère à Flask.
- [3] **Google Cloud**, “Dialogflow CX Documentation,” *Google Cloud Documentation*, 2023. Documentation officielle pour la création d'agents conversationnels avec Dialogflow.
- [4] **Lewis, Mike, Myle Ott, Jingfei Du et al.**, “Natural Language Processing with Transformers,” *O'Reilly Media*, 2020. Introduction aux Transformers et leur application dans les chatbots.
- [5] **Mitchell, Ryan**, “Data Scraping with Python and Selenium,” *O'Reilly Media*, 2021. Guide détaillé pour l'extraction de données web avec Selenium et Python.
- [6] **Mnih, Volodymyr, Koray Kavukcuoglu, David Silver et al.**, “Human-level control through deep reinforcement learning,” *Nature*, 2015, 518 (7540), 529–533. Présentation des réseaux de neurones profonds pour l'apprentissage par renforcement.
- [7] **Ng, Andrew**, “Machine Learning Yearning,” *DeepLearning.ai*, 2018. Guide pratique pour construire et itérer des systèmes d'apprentissage automatique.
- [8] **Raj, Sumit**, “Building Chatbots with Python,” *Apress*, 2019. Introduction complète au développement de chatbots avec Python et NLTK.
- [9] **SeleniumHQ**, “Selenium with Python Documentation,” *Selenium Official Documentation*, 2023. Référence pour l'automatisation et le scraping web avec Selenium.
- [10] **Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio et al.**, “Deep Learning for Chatbots,” *arXiv preprint arXiv:1605.05744*, 2016. Travail académique sur les techniques d'apprentissage profond appliquées aux chatbots.