REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

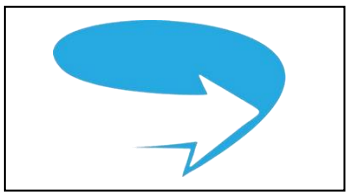
FACULTÉ DES MATHEMATIQUES ET DE l'INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de **Master**

Spécialité : **Génie Logiciel**

En vue de créer une startup

TAWJIH



Par :
Lardjane Imane
Moumene Manar

Sur le thème

Développement d'un Chatbot Intelligent pour l'optimisation de l'assistance Client dans les Organisations
Cas d'étude : Centre des Impôts de Tiaret

Soutenu publiquement le 19 / 06 / 2025 à Tiaret devant le jury composé de :

Mr BOUDAA Boudjemaa	MCA	Université de Tiaret	Président
Mr BEKKI Khadhir	MAA	Université de Tiaret	Encadrant
Mr OUARED Abdelkader	MCA	Université de Tiaret	Co-encadrant
Mme ZENATI Anissa	MAA	Université de Tiaret	Examinatrice
Mr DAOUD Mohamed Amine	MCB	Université de Tiaret	Représentant de l'incubateur
Mr BENAMARA Ahmed	Ingénieur en informatique - Directions des finances de Tiaret		Représentant du partenaire économique

2024-2025

Remerciements

Avant toute chose, nous remercions **Allah**, le Tout-Puissant, qui nous a accordé la force, la patience et la persévérance pour mener à bien ce travail. C'est grâce à Sa guidance que nous avons pu franchir cette étape importante de notre parcours.

Nous tenons à exprimer notre plus sincère gratitude à notre encadrant, **Mr. BEKKI Khadhir**, pour sa disponibilité, ses précieux conseils et son accompagnement tout au long de ce travail.

Nous remercions également notre co-encadrant, **Mr. OUARED Abdelkader**, pour son encadrement, ses orientations judicieuses et son soutien continu.

Nous tenons également à remercier chaleureusement les membres du jury, **Mr. BOUDAA Boudjemaa** et **Mme. ZENATI Anissa**, pour avoir accepté d'évaluer notre travail et pour l'intérêt qu'ils y ont porté.

Nos remerciements s'adressent aussi à l'ensemble des enseignants et personnels de l'université pour la qualité de leur encadrement tout au long de notre cursus.

Dédicace

À ma mère

Toi qui représentes la lumière de ma vie, le symbole de la tendresse infinie et du soutien inconditionnel.
Tes prières et ton amour m'ont accompagné tout au long de ce parcours.
Ce travail n'est qu'un modeste témoignage de ma gratitude et de mon profond amour.

À mon père

À toi qui n'as jamais cessé de te sacrifier pour mon avenir.
Ton courage, ta sagesse et ton dévouement m'ont toujours inspiré.
Que ce travail soit l'aboutissement des efforts que tu as semés pour moi.

À ma famille

À mes frères et sœurs, piliers de ma vie,
merci pour votre soutien, votre affection et votre présence constante.
Vous êtes une source de motivation et de sérénité dans mon cœur.

À mes amis

À tous ceux qui ont partagé ce chemin avec moi,
vos encouragements, vos sourires et votre amitié ont embelli mon parcours.
Merci d'avoir été là, dans les moments de doute comme dans les moments de joie.

Moumene Manar

Dédicace

Je dédie ce travail :

À moi-même, Pour avoir su garder la patience et la motivation malgré les moments difficiles. Ce travail est le fruit de ma détermination et de mon courage face aux obstacles.

À mes parents, Pour leur amour infini, leur soutien constant et leurs sacrifices silencieux. Sans vous, rien de tout cela n'aurait été possible. Merci d'avoir toujours cru en moi.

À mon frère et mes soeurs, Pour votre présence réconfortante, vos encouragements sincères et votre soutien tout au long de ce parcours. Vous avez été ma force dans les moments de doute.

À mes amies, Pour vos mots d'encouragement, votre écoute attentive et votre soutien indéfectible.

Lardjane Imane

Résumé

Dans les organisations, le service d'assistance aux clients est souvent surchargé par des demandes fréquentes et répétitives concernant les documents à fournir, les étapes de procédures, les horaires d'ouverture et fermeture, et les informations administratives en général. Cela entraîne des aller retours inutiles et des pertes de temps pour les clients et le personnel.

Ce projet propose la conception et le développement d'un chatbot intelligent pour automatiser et structurer l'assistance aux clients. Grâce aux techniques avancées de traitement du langage naturel (NLP), ce chatbot sera capable de comprendre les demandes des utilisateurs et de fournir des réponses claires, structurées et personnalisées. Les informations communiquées incluront la liste des documents à fournir, les étapes de procédures, les horaires de disponibilité des services, ainsi que des conseils adaptés au profil de l'utilisateur.

L'objectif est d'améliorer l'expérience utilisateur en réduisant le nombre d'interactions humaines pour des questions récurrentes tout en offrant une assistance rapide et efficace. Ce projet est inscrit dans le cadre d'une startup qui sert à différencier en offrant un service d'assistance client intelligent, optimisé et automatisé.

Mots clés : Chatbot, Assistance Client, Traitement du Langage Naturel (NLP), Automatisation, Intelligence Artificielle, Systèmes d'Information, Service Client.

Abstract

In organizations, customer support services are often overwhelmed by frequent and repetitive inquiries regarding required documents, procedural steps, opening and closing hours, and general administrative information. This results in unnecessary back-and-forth communication and wasted time for both customers and staff.

This project proposes the design and development of an intelligent chatbot to automate and structure customer support. Using advanced Natural Language Processing (NLP) techniques, the chatbot will be able to understand user requests and provide clear, structured, and personalized responses. The information provided will include the list of required documents, procedural steps, service availability hours, as well as advice tailored to the user's profile.

The goal is to improve the user experience by reducing the number of human interactions for recurring questions while offering fast and efficient support. This project is part of a startup that aims to differentiate itself by offering an intelligent, optimized, and automated customer support service.

Keywords : Chatbot, Customer Support, Natural Language Processing (NLP), Automation, Artificial Intelligence, Information Systems, Customer Service.

ملخص

في المؤسسات، غالبًا ما تكون خدمة الدعم للعملاء مثقلة بالطلبات المتكررة والمكررة، مثل الاستفسارات المتعلقة بالوثائق المطلوبة، وخطوات الإجراءات، ومواعيد العمل (الافتتاح والإغلاق)، والمعلومات الإدارية بشكل عام. هذا الوضع يؤدي إلى ذهاب وإياب غير ضروري، ويُسبب هدرًا في الوقت لكل من العملاء والموظفين.

يقترح هذا المشروع تصميم وتطوير روبوت دردشة ذكي يهدف إلى أتمتة وتنظيم خدمة الدعم للعملاء. وبالاعتماد على تقنيات متقدمة في معالجة اللغة الطبيعية (NLP)، سيكون هذا الروبوت قادرًا على فهم طلبات المستخدمين وتقديم إجابات واضحة، منظمة، ومخصصة حسب ملف المستخدم. تشمل المعلومات التي سيتم تقديمها: قائمة الوثائق المطلوبة، وخطوات الإجراءات، ومواعيد توفر الخدمات، بالإضافة إلى نصائح موجهة حسب حالة المستخدم.

الهدف من هذا المشروع هو تحسين تجربة المستخدم من خلال تقليل عدد التفاعلات البشرية في الأسئلة المتكررة، وتقديم دعم سريع وفعّال . يعد هذا المشروع جزءًا من شركة ناشئة تهدف إلى التميز من خلال تقديم خدمة دعم عملاء ذكية ومحسنة وآلية..

الكلمات المفتاحية :روبوت دردشة، دعم العملاء، معالجة اللغة الطبيعية(NLP) ، الأتمتة، الذكاء الاصطناعي، نظم المعلومات، خدمة العملاء.

TABLE DES MATIÈRES

Liste de Figures

Liste de Tables

Glossaire

NLP : Natural Language Processing (Traitement automatique du langage naturel).

LLM : Large Language Model (Grand Modèle de Langage).

IA : Intelligence Artificielle.

UI : User Interface (Interface Utilisateur).

UX : User Experience (Expérience Utilisateur).

TALN : Traitement Automatique du Langage Naturel.

RAG : Retrieval-Augmented Generation (Génération augmentée par récupération).

TF-IDF : Term Frequency-Inverse Document Frequency (Fréquence du terme - Fréquence inverse de document).

CNN : Convolutional Neural Network (Réseau de Neurones Convolutif).

RNN : Recurrent Neural Network (Réseau de Neurones Récurrent).

BERT : Bidirectional Encoder Representations from Transformers (Représentations bidirectionnelles encodées par transformeurs).

GPT : Generative Pre-trained Transformer (Transformeur génératif pré-entraîné).

POS : Part-of-Speech (Catégories grammaticales).

NER : Named Entity Recognition (Reconnaissance d'entités nommées).

NLU : Natural Language Understanding (Compréhension du langage naturel).

NLG : Natural Language Generation (Génération du langage naturel).

ASR : Automatic Speech Recognition (Reconnaissance automatique de la parole).

XLNet : Modèle de langage autoregressif basé sur permutation (successeur de BERT).

T5 : Text-to-Text Transfer Transformer (Modèle de conversion texte-à-texte).

CPU : Central Processing Unit (Processeur Central).

RAM : Random Access Memory (Mémoire vive).

GPU : Graphics Processing Unit (Processeur Graphique).

DM : Dialog Management (Gestion du dialogue).

RG : Réponse Générée.

API : Application Programming Interface (Interface de programmation d'applications).

CLM : Causal Language Modeling (Modélisation causale du langage).

MLM : Masked Language Modeling (Modélisation du langage masqué).

LORA : Low-Rank Adaptation (Adaptation à faible rang – technique de fine-tuning efficace).

QLORA : Quantized LORA (Version quantifiée de LoRA pour réduire la mémoire utilisée).

FAQ : Foire Aux Questions.

BMC : Canevas de Modèle d'Affaires.

Introduction Générale

I- Introduction Générale

I.1- Context et motivation

Les institutions gouvernementales sont souvent confrontées à des défis pour fournir des services aux citoyens de manière efficiente et rapide. Les demandes fréquentes des citoyens concernant les horaires d'ouverture, les documents à fournir, les procédures administratives ou d'autres informations essentielles sur les services proposés peuvent entraîner de longues files d'attente et une surcharge de travail pour les employés. Cela engendre non seulement des désagréments pour les citoyens, mais nuit également à l'efficacité globale de l'administration[30].

Dans ce cadre, un système automatisé s'appuyant sur un chatbot intelligent pourrait représenter une solution judicieuse pour faciliter et accélérer la gestion des requêtes répétitives. En utilisant des technologies de pointe telles que le traitement du langage naturel (NLP) et modèles de langage de grande taille (LLMs), ce chatbot est en mesure de comprendre et de répondre de façon appropriée aux requêtes des citoyens, qu'il s'agisse de questions liées à l'administration, de procédures à suivre ou même d'orientations sur la manière d'accéder à certains services spécifiques (e.g., [30]).

I.2- Problématique

À titre d'exemple typique, les contribuables qui interagissent avec l'administration fiscale posent souvent les mêmes questions récurrentes : « Quels documents dois-je fournir pour ma déclaration d'impôts ? », « Comment régulariser une situation fiscale ? », « *Quels sont les délais pour le paiement des taxes* ? ». Le volume important de ces questions génère une sur charge pour les agents fiscaux, qui doivent répondre individuellement à chaque requête, provoquant ainsi des délais d'attente prolongés, des erreurs potentielles dans les réponses fournies et une insatisfaction croissante des usagers. Par ailleurs, cette situation entraîne pour les usagers de nombreux allers-retours quotidiens entre les bureaux administratifs, à cause d'informations parfois incomplètes ou contradictoires, ce qui alourdit encore d'avantage les délais et la charge de travail. Cette situation nuit à l'efficacité du service public et accroît la pression sur les équipes chargées du service client. En interne, les employés des services de soutien sont souvent confrontés à des interruptions fréquentes de la part de leurs collègues, qui sollicitent leur aide pour des questions relatives aux procédures, à l'utilisation des outils ou à d'autres aspects techniques. Ces interruptions répétées nuisent à leur concentration et réduisent leur productivité. Cette surcharge engendre du stress, une fatigue mentale accrue et une baisse de motivation, affectant la qualité du travail et le bien-être au sein de l'organisation. L'absence d'un système centralisé et accessible pour répondre rapidement aux questions courantes oblige les collaborateurs à recourir constamment à l'aide

directe de leurs pairs, ce qui perturbe le fonctionnement global des services et augmente la charge collective de travail.

I.3- Notre contribution

Dans un contexte où les administrations publiques sont confrontées à une demande croissante de rapidité, de transparence et d'accessibilité, ce projet propose le développement d'un chatbot intelligent destiné à automatiser et optimiser l'assistance aux citoyens. Conçu comme un système hybride, ce chatbot combinera des approches basées sur des règles métier avec des techniques avancées d'intelligence artificielle (IA) pour offrir des réponses contextualisées, pertinentes et fiables.

L'enjeu central est de réduire la charge de travail des agents, souvent sollicités pour répondre à des questions répétitives, tout en améliorant la qualité du service offert aux usagers. En s'appuyant sur une interface intuitive et un moteur de traitement linguistique performant, le chatbot sera capable de comprendre les demandes, d'en analyser le contexte (statut administrative..etc.) et de fournir des réponses personnalisées, de manière autonome et instantanée.

Ce projet s'inscrit dans une logique d'innovation et la transition numérique demandé dans les axe de développement en Algérie au service de l'administration et poursuit l'objectif suivant :

Objectif : Développer un **chatbot intelligent** capable de traiter automatiquement et avec précision les demandes des *citoyens* et des *agents administratifs*.

Les actions menées pour atteindre notre objectif sont synthétisées comme suit :

1. Analyse des **besoins** d'information des clients des services publics (ex. : service financier).
2. Élaboration de **questions fréquentes** à partir des registres, des sites et des entretiens.
3. Création d'un **dataset** à partir de LLM avec des prompts, suivie par une vérification manuelle.
4. Développer un **modèle de langue** question/réponse.
5. Développer une **interface UI/UX** dédiée pour les citoyens, sous forme d'une application mobile.

Grâce à la disponibilité permanente du chatbot (24h/24, 7j/7), cette solution contribuera à améliorer l'accessibilité aux services publics, à optimiser l'expérience utilisateur et à renforcer la performance interne des administrations. De plus, le chatbot repose sur une capacité à fournir des réponses organisées et structurées, permettant ainsi un accompagnement clair et précis des usagers, qu'il s'agisse des documents à fournir, des étapes à suivre ou des coordonnées des services. Ce projet constitue ainsi un levier stratégique pour une transformation numérique efficace, durable et centrée sur l'utilisateur.

I.4- Organisation de mémoire

Après cette introduction générale , ce mémoire est structuré en plusieurs chapitres :

— **Chapitre 1 :** *Introduction à l'intelligence artificielle générative.* Ce chapitre introduit les concepts fondamentaux de l'IA générative, en particulier le traitement du langage naturel (NLP) et les grands modèles de langage (LLM).

— **Chapitre 2 :** *Création d'agents conversationnels intelligents.* Ce chapitre est consacré aux chatbots, en abordant leurs types, leurs modes de fonctionnement et leurs domaines d'application.

— **Chapitre 3 :** *Proposition d'un chatbot hybride pour support client .* Il présente notre proposition d'un chatbot hybride visant à améliorer l'efficacité du service client.

— **Chapitre 4 :** *Outils et technologies.* Ce chapitre est dédié à la présentation des outils techniques ayant permis le développement du système.

— **Conclusion et perspectives**

1

Concepts fondamentaux sur l'IA Générative

Table de sommaire

1.1 Introduction

L'IA générative désigne des modèles capables de produire du contenu original (texte, images, vidéos, etc.) à partir de vastes ensembles de données, en générant des prédictions émergentes et de nouvelles formes de présentation.

Le traitement du langage naturel (NLP) joue un rôle majeur dans le paysage de l'IA générative, permettant aux modèles de comprendre, d'analyser et de fournir un langage du monde réel avec une précision inébranlable. L'essor du traitement du langage naturel et de l'IA générative a conduit à l'essor des modèles linguistiques à long terme (LLM) , caractérisés par leur capacité à communiquer et à reproduire un modèle d'activité spécifique et cohérent.

En s'appuyant sur une large gamme de médias et d'algorithmes avancés, les programmes LLMs sont devenus des outils révolutionnaires dans certains domaines, établissant un processus décisionnel automatisé collectif. Comprendre l'interdépendance de l'IA générative, du traitement du langage naturel et de la maîtrise en linguistique est essentiel pour être conscient des avancées et des défis associés à ces technologies émergentes.

Dans ce chapitre, nous présenterons les principes fondamentaux et les méthodologies qui soutiennent ces technologies.

1.2 Traitement du langage naturel (NLP)

Definition 1.2.1 (Natural Language Processing (NLP))**.** *NLP est la discipline de l'IA qui traite L, le langage naturel, permettant à une machine M de* $\{comprendre, interpréter, générer\}$ *du texte ou de la parole.*

Le TLN ne se limite pas à la reconnaissance des mots , il explore les subtilités de la syntaxe, de la sémantique et de la pragmatique pour saisir l'ensemble du spectre de la communication humaine [29].

La figure 1.1 illustre le flux de travail d'un système de TALN, qui suit un processus en plusieurs étapes : à partir d'un texte brut, il applique un prétraitement, extrait des caractéristiques, puis utilise un modèle pour produire un résultat tel qu'une classification ou une traduction.

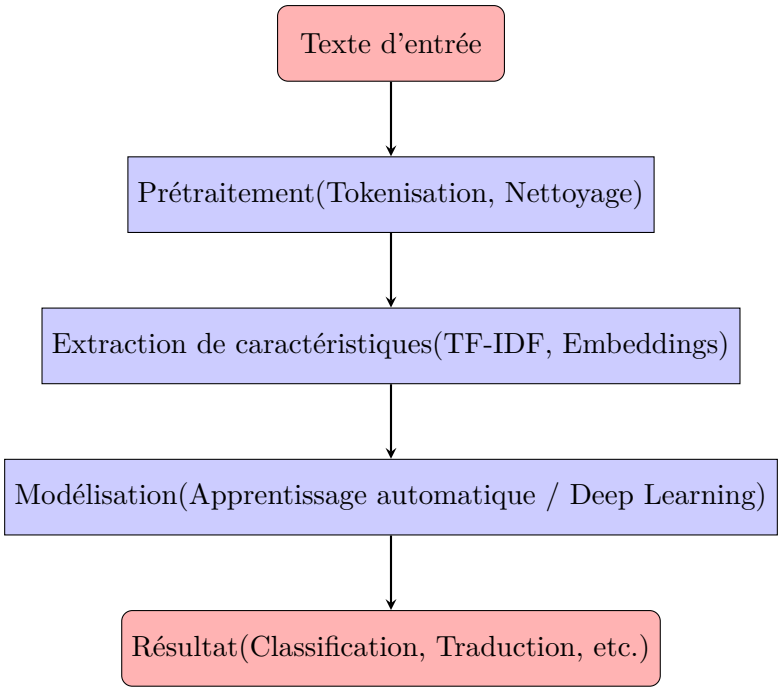


FIGURE 1.1 – Flux de travail du traitement automatique du langage naturel (TALN).

1.2.1 Historique du chatbot

Origine et premiers concepts L'idée de la traduction automatique remonte au 17 siècle, avec des philosophes comme Leibniz et Descartes, qui imaginaient des langages universels logiques.

1930s Premières demandes de brevets pour dispositifs de traduction, dont un dictionnaire bilingue automatisé par Georges Artsrouni.

1950 Test de Turing : évaluation de la capacité d'une machine à simuler un dialogue humain.

1950-1960 *Premières avancées* :

— 1954 : Traduction de 60 phrases russe-anglais par Georgetown-IBM.

— 1957 : Grammaire générative de Chomsky, fondation pour le TLN.

1960-1980 *Hivers de l'IA* :

— 1964 : Création de l'ALPAC (États-Unis).

— 1966 : le rapport ALPAC critique la traduction automatique, freinant le financement.

1980-2000 *Renaissance statistique* :

— Années 1980 : Modèles statistiques (sac de mots, n-grams).

— Fin 1980s : Premiers RNNs, limités par la puissance.

— Années 1990 : corpus annotés et machine learning (POS tagging, NER).

2000-présent *Ère du deep learning* :

— Années 2000 : Big Data et réseaux sociaux dynamisent le TLN.

— Années 2010 : architectures CNN, RNN, Transformers.

— 2017 : Transformer de Vaswani révolutionne avec BERT, GPT, etc.

— Aujourd'hui : chatbots et traducteurs neuronaux omniprésents [28] .Comme illustré dans la figure 1.2 [9] .

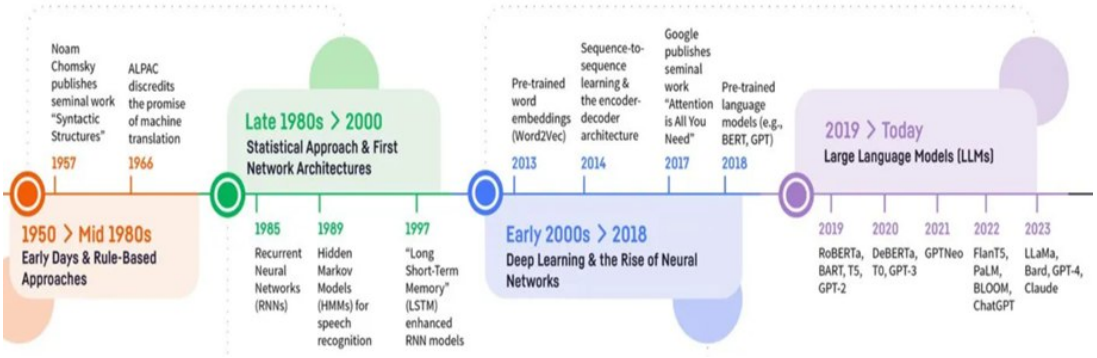


FIGURE 1.2 – L'histoire du traitement de langage naturel (NLP) .

1.2.2 Les composantes de traitement de Langage Naturel (NLP)

Le traitement du langage naturel donne aux machines la capacité de déchiffrer, comprendre et analyser le langage humain, une compétence vitale pour une multitude d'applications dans divers domaines, à l'instar du service à la clientèle, de la santé, des finances et de l'éducation.

Les trois éléments mentionnés constituent des éléments primordiaux du traitement de langage naturel :

— **Reconnaissance vocale** : conversion du discours en texte écrit.

— **Compréhension du langage naturel (NLU)** : aptitude d'un ordinateur à saisir le sens des mots.

— **Génération de langage naturel (NLG)** : création de contenu en langage naturel par un ordinateur.

Comme illustré dans l'algorithme 1, le processus NLP commence par la reconnaissance vocale, suivie de la compréhension et de la génération du langage naturel.

1.2.2.1 Compréhension du langage naturel (NLU)

L'aptitude à la compréhension du langage naturel permet aux ordinateurs de saisir le langage humain, qu'il soit sous forme écrite ou orale. C'est une procédure complexe comportant plusieurs couches d'examen :

- Analyse syntaxique : décompose le langage en ses éléments grammaticaux (phrases, clauses, syntagmes).

Algorithm 1 Système simplifié de traitement du langage naturel

Require: Signal audio ou texte x
Ensure: Réponse générée y
1: **if** x est audio **then**
2: $x \leftarrow \texttt{ReconnaissanceVocale}(x)$ // $\Rightarrow$
3: **end if**
4: $m \leftarrow \texttt{CompréhensionNL}(x)$ // compréhension : extraire sens, intentions
5: $y \leftarrow \texttt{GénérationNL}(m)$ // $\Rightarrow$
6: **return** y

- Analyse sémantique : cherche à saisir le sens du langage en déterminant le contexte, l'intonation et l'intention.

- Analyse pragmatique : porte un intérêt particulier au contexte culturel et social du langage (sarcasme, ironie, humour).

Exemple- *analyse d'une question financière.*

Question : *Quels sont les risques associés à un investissement dans les cryptomonnaies ?*

— **Analyse syntaxique** : identifier la structure de la phrase, le sujet (*les risques*), le verbe (*sont*), le complément (*associés à un investissement dans les cryptomonnaies*).
— **Analyse sémantique** : comprendre le sens de la question, notamment ce que signifie « risques »(volatilité, régulation, cybersécurité) et « investissement dans les cryptomonnaies ».
— **Analyse pragmatique** : interpréter l'intention de l'utilisateur (souhaite-t-il investir, ou simplement s'informer ?). et prendre en compte le contexte (marché actuel, tendances, actualité).

1.2.2.2 Génération du langage naturel (NLG)

La création de langage naturel (NLG) fait référence à l'utilisation d'algorithmes pour générer un langage semblable à celui de l'homme. Elle intègre diverses phases, résumées dans le tableau 1.1

Phase	**Description**
Organisation des phrases	Structuration de l'information de manière cohérente.
Réalisation de surface	Utilisation de la grammaire et du vocabulaire pour créer un texte cohérent.

Table 1.1 – Phases de la création de langage naturel (NLG).

Utilisations du NLG :

- Journalisme automatisé : création d'articles à partir de données organisées (ex. : résultats sportifs).

- Chatbots et assistants numériques : création de réponses aux demandes des utilisateurs[28].

1.2.2.3 Reconnaissance de la parole (ASR)

La technologie de reconnaissance de la parole (Automatic Speech Recognition, ASR) transforme le discours oral en texte écrit. Elle est utilisée dans des applications telles que les assistants numériques (Siri, Alexa) et les services de transcription en direct.

Le processus englobe les éléments suivants :

- Modélisation acoustique : étude des ondes sonores et transformation en formats numériques.

- Modélisation du langage : utilisation d'algorithmes statistiques pour analyser grammaticalement le langage parlé.

- Décodage : interprétation définitive du texte qui a été transcrit.

La technologie de reconnaissance vocale trouve son application dans divers secteurs tels que la santé, le support client et l'éducation[28].

La figure 1.3 présente le pipeline de NLP.

FIGURE 1.3 – Pipeline NLP réduit : Reconnaissance, Compréhension et Génération.

1.2.3 Niveaux de traitement du langage naturel (NLP)

Le traitement du langage naturel (NLP) repose sur une modélisation en niveaux de langage permettant d'expliquer son fonctionnement interne. Contrairement à l'approche séquentielle classique, où chaque niveau est traité de manière linéaire, les recherches en psycholinguistique montrent que ces niveaux interagissent de façon dynamique.

Un système NLP performant exploite donc simultanément plusieurs niveaux linguistiques pour une analyse plus précise et contextuelle.

1.2.3.1 Niveau de phonologie

Ce niveau concerne principalement la prononciation. Il s'agit de l'interprétation des sons de la parole par le biais des mots[27].

1.2.3.2 Niveau morphologique

Ce niveau traite des plus petites unités de sens, à savoir les préfixes et les suffixes. Par exemple, le mot *lapins* comporte deux morphèmes : *lapin* et *s*, ce dernier indiquant le pluriel[27].

1.2.3.3 Niveau lexical

Ce niveau implique l'examen des mots en termes de leur signification lexicographique et de leur catégorie grammaticale (POS). Il s'appuie sur le lexique, qui est une collection de lexèmes [27].

1.2.3.4 Niveau syntaxique

Ce niveau traite de la grammaire et de la structure des phrases. Il étudie les relations entre les mots. Le résultat du balisage POS est utilisé ici pour grouper les mots dans des phrases et des clauses [27].

1.2.3.5 Niveau sémantique

Ce niveau traite du sens des mots et des phrases. Deux approches sont couramment utilisées :

- **Analyse sémantique pilotée par la syntaxe.**
- **Grammaire sémantique :** elle étudie le sens des mots en lien avec leur structure grammaticale [27].

1.2.3.6 Niveau de discours

Ce niveau aborde la structure de divers types de textes. On distingue notamment :

- **La résolution d'anaphore**
- **La reconnaissance de la structure du discours/texte**

Les mots sont remplacés dans la résolution de l'anaphore[27].

1.2.3.7 Niveau pragmatique

Ce niveau traite de l'utilisation des connaissances du monde réel et de la compréhension de la manière dont celles-ci influencent le sens de ce qui est communiqué.

La pragmatique identifie le sens des mots et des phrases en fonction de la manière dont la langue est utilisée pour communiquer[27].

La table 1.2 illustre des examples des niveaux linguistiques.

Niveau	**Exemple**
Phonologie	Reconnaître oralement la phrase *"déclaration d'impôt"* malgré les variations d'accent.
Morphologie	Analyser que *"imposable"* se compose de *"im-"*, *"pos"* (racine), et *"-able"*.
Lexical	Comprendre que *"fiscal"* se rapporte aux impôts, pas à autre chose.
Syntaxe	Vérifier que la phrase *"Le contribuable soumet sa déclaration"* est grammaticalement correcte.
Sémantique	Interpréter que *"le montant dû"* signifie la somme d'argent à payer à l'administration fiscale.
Discours	Dans un échange, comprendre que *"ce document"* se réfère au formulaire d'impôt mentionné avant.
Pragmatique	Interpréter la question *"Pouvez-vous envoyer la déclaration avant vendredi ?"* comme une demande urgente.

TABLE 1.2 – Exemples des niveaux linguistiques appliqués au domaine des impôts.

1.2.4 Prétraitement des données

Le prétraitement des données est l'étape la plus essentielle de tout modèle de machine learning. La qualité du nettoyage et du prétraitement des données brutes joue un rôle majeur dans les performances du modèle.

De même, dans le cas du traitement de langage naturel, la toute première étape est le traitement de texte [52].

1.2.4.1 Prétraitement de texte

À l'aide de minuscules , Il est assez évident, d'après le nom lui-même, que nous essayons de convertir nos données textuelles en minuscules.

Mais pourquoi cette étape est-elle nécessaire ?

Lorsque nous avons une entrée de texte, comme un paragraphe, nous terminons les mots en minuscules et en majuscules. Cependant, les mêmes mots écrits dans des casses différentes sont considérés comme des entités différentes par l'ordinateur [52]

1.2.4.2 La tokenisation

La tokenisation est l'opération qui consiste à diviser une requête utilisateur en unités plus petites, appelées *tokens*.

Cette procédure s'appuie sur un ensemble de données d'entraînement et peut être effectuée en utilisant diverses méthodes, en fonction de l'approche de tokenisation choisie.

L'objectif principal de la tokenisation est d'éliminer les éléments superflus tels que les mots parasites, les symboles et les chiffres qui pourraient nuire à la pertinence de la requête de recherche de l'utilisateur[35], comme illustré dans l'algorithme 2.

Algorithm 2 Tokenization

Input : $t1$ = string
Output : Tokens in $t1$
$arr \leftarrow t1$.`split()`
for $i = 0$ to $arr.length - 1$ **do**
  print $arr[i]$
end for

1.2.4.3 Suppression des mots d'arrêt

Nous devons maintenant supprimer les mots d'arrêt qui sont une collection de mots qui apparaissent fréquemment dans n'importe quelle langue, mais qui n'ajoutent pas beaucoup de sens aux phrases[35], comme illustré dans l'algorithme 3.

Algorithm 3 Stop-words Removal

Input : User Search String
Output : List of correct words
import list of stop-words as *swd*
parse Input into words
for each word i in parsed words **do**
  for each stop-word j in *swd* **do**
    if word[i] == swd[j] **then**
      remove word[i]
    end if
  end for
end for
return correct words

1.2.4.4 Dérivation (Stemming)

Le terme *stemming* vise à déterminer le sens fondamental d'un mot. Cette étape consiste à séparer les affixes et autres éléments lexicaux des unités lexicales, ne conservant que la portion de base[35], comme illustré dans l'algorithme 4.

1.2.4.5 Lemmatisation

La lemmatisation ne permet pas toujours d'obtenir des mots qui font partie du vocabulaire de la langue. Elle donne souvent lieu à des mots qui n'ont aucun sens pour les utilisateurs. Afin de surmonter cet inconvénient, nous utiliserons le concept de lemmatisation[52],comme illustré dans l'algorithme 5.

Algorithm 4 Stemming

```
import stemmer
Input 11 = List of correct words
Output = Sentence with stemmed words
arr[] = parse(11)
for i = 1 to number of words in arr[] do
    if arr[i] == compound word then
        Stemmer.stem(arr[i])
    end if
end for
return query with stemmed words
```

Algorithm 5 Lemmatization

```
import lemmatizer
Input = Sentence of words
Output = Sentence with lemmatized words
arr[] = tokenize(Input)
for i = 1 to number of words in arr[] do
    arr[i] = lemmatizer.lemmatize(arr[i])
end for
return arr[]
```

1.2.5 Applications du traitement du langage naturel

Le traitement automatique du langage naturel (TALN) intervient dans un large éventail d'applications. La figure 1.4 illustre les usages les plus courants de cette technologie dans divers domaines.

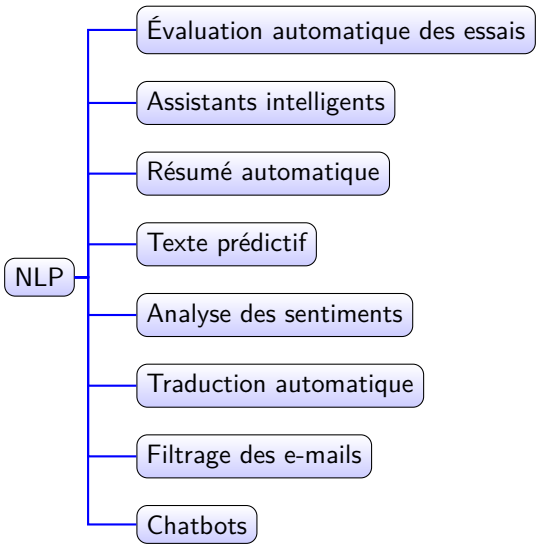


FIGURE 1.4 – Applications courantes du traitement automatique du langage naturel (NLP) dans divers secteurs.

— **Chatbots :** automatisent le service client en comprenant et répondant aux requêtes des utilisateurs [15].

— **Filtrage des e-mails :** trier et classer les e-mails pour détecter les spams et organiser les messages [15].

— **Traduction automatique :** facilite la traduction de documents techniques et d'assistance, aidant à surmonter les barrières linguistiques, bien que la compréhension fine des phrases reste complexe [6].

— **Analyse des sentiments :** identifier les émotions et opinions dans les textes pour améliorer les stratégies d'entreprise [15].

— **Texte prédictif :** anticiper les mots et corriger les erreurs pour accélérer et fluidifier l'écriture [15].

— **Résumé automatique :** extraire l'essentiel d'un texte pour en générer une version concise et compréhensible [15].

— **Assistants intelligents :** interpréter et exécuter des commandes vocales pour une interaction plus intuitive avec la technologie [15].

— **Évaluation automatique des essais :** analyser et noter objectivement les textes écrits, assurant une correction rapide, cohérente et sans biais [15].

1.2.6 Défis du traitement du langage naturel (NLP)

Le traitement du langage naturel a connu des progrès remarquables ces dernières années, mais il est encore confronté à plusieurs défis qui entravent son plein potentiel. Ces défis incluent :

— **Ambiguïté et polysémie :** difficulté à interpréter les multiples sens des mots selon le contexte [13].

— **Rareté et qualité des données :** manque de données annotées et irrégularité des données disponibles [13].

— **Contexte et compréhension :** complexité à saisir et modéliser le contexte précis dans les textes [13].

— **Variations linguistiques et multilinguisme :** obstacles liés aux dialectes, à l'argot et à la diversité linguistique [13].

— **Insuffisance des données spécifiques au domaine :** difficulté d'accès aux données labellisées pour secteurs spécialisés (santé, droit, finance) [13].

— **Compréhension et raisonnement sémantiques :** limites dans les inférences, déductions logiques et capture des nuances du langage [13].

— **Évolutivité et performance :** Défi de concevoir des modèles capables de traiter de grandes quantités de données tout en restant performants [13].

1.3 Grands modèles de langage (LLM)

Les grands modèles de langage (LLM) sont des réseaux de neurones qui ont été formés sur d'immenses volumes de données textuelles. Ils sont capables de prévoir le mot suivant dans une suite, de saisir le contexte et de produire un texte semblable à celui d'un humain. Ils reposent sur des méthodes d'apprentissage profond, en particulier des structures de transformateurs. Les grands modèles de langage possèdent des milliards de paramètres, ce qui les rend capables de comprendre et de générer un langage humain complexe[50].

Exemple de dialogue avec LLM : Dans cet exemple, nous avons demandé une explication sur la signification du nom de l'application **"Tawjih".**

Requête utilisateur : Que signifie "Tawjih" ?

Réponse : "Tawjih" est un mot arabe qui signifie "orientation" ou "guidance".

1.3.1 Évolution des systèmes de grands modèles de langage

Historique :

Les LLM sont une catégorie de modèle d'IA capable de traiter et de produire du texte en langue naturelle. Ces modèles sont généralement formés sur de vastes corpus textuels [50].

1950/1980 : Modèles à base de règles

— Utilisent des règles linguistiques écrites à la main.
— Capacité limitée à gérer la complexité du langage naturel.

1980/2000 : Renaissance statistique

— **Années 1980** : Introduction des modèles statistiques (sac de mots, n-grams).
— **Fin des années 1980** : Premiers RNNs, limités par la puissance de calcul.
— **Années 1990** : Développement de corpus annotés, POS tagging, NER.

2000/2015 : Big Data et deep learning

— **Années 2000** : Essor des réseaux sociaux et du Big Data.
— **2010** : Modèle RNNLM – meilleur contexte pour la génération de texte.
— **2015** : GNMT (Google) – Traduction neuronale à grande échelle.

2017/Aujourd'hui : Révolution des Transformers et LLMs

— **2017** : Introduction du Transformer.
— **2018** : GPT-1 (OpenAI), 117M de paramètres, basé sur Transformer.
— **2020** : GPT-3 – génération de texte cohérente, multitâche.
— **Aujourd'hui** : chatbots, traducteurs neuronaux et assistants vocaux omniprésents.

1.3.2 Fonctionnement des modèles LLM

Les LLMs reposent sur des modèles de transformateur, dévoilés dans l'article de 2017 « Attention Is All You Need ». Ces modèles mettent en oeuvre des mécanismes d'attention et d'auto-attention pour évaluer l'importance des mots dans une phrase, saisissant le contexte, le ton et la syntaxe. La formation implique le traitement de vastes corpus de texte et l'ajustement des paramètres du modèle pour prédire avec précision le prochain mot[11].

Example

Les *LLM* reposent sur des modèles de *transformers* qui utilisent des mécanismes d'*attention* pour comprendre le contexte d'une phrase. Par exemple, pour la séquence d'entrée :

"Le chat mange..."

le modèle prédit le mot suivant en évaluant les probabilités possibles et choisit souvent un mot comme *"une"*, *"la"* ou *"sa"*, afin de former une phrase cohérente.

Ainsi, en traitant mot par mot et en tenant compte du contexte global, le LLM génère un texte fluide et naturel.

1.3.3 L'architecture du transformateur

Les modèles de langage modernes se composent de plusieurs composants, souvent construits à partir de différents réseaux neuronaux, chacun conçu pour des tâches spécifiques. Lancée en 2017, l'architecture Transformer est devenue le choix dominant, révolutionnant le traitement du langage naturel (NLP) et s'étendant à d'autres domaines de l'IA [44].

L'architecture du transformateur est considérée comme la pierre angulaire des LLM. Elle est conçue pour les réseaux neuronaux afin de traiter efficacement les données séquentielles. Cette structure ne fait pas appel à des méthodes d'itération. Au contraire, elle s'appuie sur une approche axée sur l'attention pour déterminer les dépendances globales des entrées-sorties. Le modèle a la capacité de traiter des entrées de longueurs diverses et peut ajuster son focus en fonction de la longueur de la séquence. Par conséquent, elle est devenue le standard dans plusieurs domaines, remplaçant souvent les réseaux neuronaux récurrents ou convolutionnels plus complexes par une structure beaucoup plus performante. Pour cette raison, elle revêt une importance particulière pour les applications LLM [43].

La figure 1.5 illustre l'architecture de modèle transformateur [5].

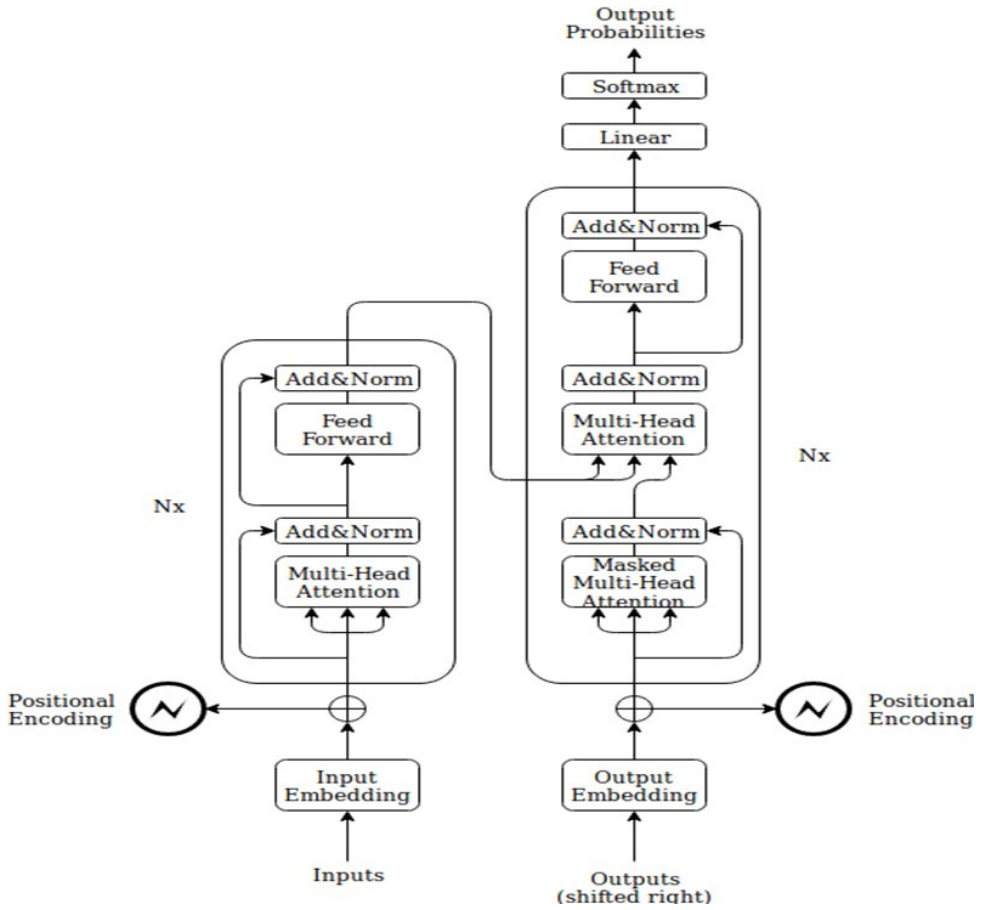


FIGURE 1.5 – L'architecture du modèle Transformer .

Le tableau 1.3 illustre les Principales composantes d'un modèle Transformer.

Composante	Description
1. Couche d'intégration	Transforme les jetons d'entrée en vecteurs numériques, représentant le sens sémantique et le contexte du texte. [17]
2. Codage de la position	Ajoute des informations sur la position de chaque mot dans la séquence, comblant l'absence de traitement séquentiel du *transformer*. [17]
3. Encodeur / Décodeur	L'encodeur extrait les éléments contextuels du texte d'entrée. Le décodeur génère des réponses en prédictant les mots suivants. [17]
4. Auto-attention	Permet au modèle de pondérer les mots selon leur importance dans le contexte. Cela améliore la compréhension des relations sémantiques globale . [17]
5. Réseaux de neurones récurrents	Appliquent des transformations supplémentaire sur les données issues de l'auto-attention, pour une compréhension plus fine du langage. [17]
6. Normalisation des couches & Connexions résiduelles	Stabilisent l'apprentissage, préviennent la disparition du gradient et facilitent l'entraînement de réseaux profonds. [17]

TABLE 1.3 – Principales composantes d'un modèle Transformer.

1.3.4 Exemples populaires de grands modèles de langage

1.3.4.1 GPT (Transformateur Génératif Pré-entraîné) :

Introduite par OpenAI en 2018, la série GPT représente une percée significative dans l'évolution des modèles de langage de grande envergure. GPT-1 a démontré l'utilité d'un pré-entraînement sur de grands ensembles de textes, suivi d'une spécialisation pour des missions précises. Les versions ultérieures, telles que GPT-2 et GPT-3, ont largement amplifié la taille et les performances du modèle, GPT-3 étant particulièrement remarquable pour ses aptitudes exceptionnelles en matière de génération et de compréhension du langage naturel [25].

1.3.4.2 BERT (Représentations d'Encodeurs Bidirectionnels de Transformers) :

Lancé par Google en 2018, BERT a transformé le traitement du langage naturel en proposant une méthode de pré-entraînement bidirectionnelle. À l'inverse des modèles antérieurs qui abordaient le texte de manière séquentielle, BERT faisait appel au masquage de mots et à la prévision de la phrase suivante pour saisir le contexte dans les deux sens. Grâce à cette méthode, il a pu exceller dans différentes activités de traitement du langage naturel, comme l'analyse de sentiments, la détection des entités nommées et la réponse aux interrogations [25].

1.3.4.3 XLNet :

Élaboré par des chercheurs de Google AI en 2019, XLNet a permis d'améliorer considérablement les modèles de langage en corrigeant les faiblesses des versions antérieures, comme la discontinuité du contexte et l'ordre des tokens. En optant pour une approche d'entraînement basée sur la permutation, il a pu saisir le contexte bidirectionnel tout en conservant les bénéfices du modèle de langage autorégressif. En adoptant cette méthode, XLNet a réalisé des performances exceptionnelles sur diverses tâches de référence, surpassant des modèles tels que BERT [25].

1.3.4.4 T5 (Transformateur de Transfert de Texte en Texte) :

Créé par Google AI en 2019, T5 a offert une perspective unifiée sur le traitement du langage naturel en convertissant toutes les tâches en transformations de texte à texte. À l'inverse des modèles traditionnels dédiés à des tâches précises, T5 est capable de réaliser une multitude de fonctions telles que la traduction, le résumé, la réponse aux interrogations et la catégorisation de texte en modifiant simplement la présentation des entrées et des sorties. Cette approche a facilité la simplification de l'architecture du modèle et son apprentissage, ce qui rend les modèles linguistiques plus performants et multifonctionnels. [25].

La table 1.4 illustre une comparaison des grands modèles de langage (LLM).

Critères	GPT	BERT	XLNet	T5
Année d'introduction	2018	2018	2019	2019
Institution	OpenAI	Google	Google AI	Google AI
Type d'apprentissage	Auto-régressif	Auto-encodeur	Auto-régressif permuté	Texte à texte
Pré-entraînement bidirectionnel	✗	✓	✓	✓
Masquage de mots	✗	✓	✗	✓
Contextualisation unifiée	✗	✗	✓	✓
Tâches multiples	✓	✓	✓	✓
Génération de texte	✓	✗	✓	✓
Utilisable comme assistant IA	✓	✓	✓	✓
Performances sur benchmarks	Haute	Très haute	Très haute	Très haute

TABLE 1.4 – Comparaison des grands modèles de langage (LLM).

Nous avons interrogé un LLM sur les modèles de langage open source qui ne consomment pas beaucoup de ressources, afin de sélectionner celui qui répond le mieux aux contraintes de CPU de nos machines.

Question : Quel LLM open source léger recommander pour un usage efficace sans forte consommation de ressources ?]

Réponse : Un des meilleurs choix actuels est **Mistral 7B** ou sa variante fine-tunée **Mixtral**. Ce modèle open source offre un bon compromis entre performance et légèreté. Il est optimisé pour tourner sur une machine locale avec des ressources limitées (GPU modeste ou même CPU avec quantization). D'autres alternatives incluent **LLaMA 2 7B**, ou encore **Phi-2** de Microsoft pour des environnements ultra-légers. Ces modèles permettent un déploiement local sans dépendance cloud, avec une qualité de génération adaptée à de nombreuses tâches NLP courantes.

La figure 1.6 représente un aperçu chronologique des grands modèles de langage (LLMs), des modèles multimodaux et des modèles scientifiques au cours des dernières années. Les LLMs dont les points de contrôle sont accessibles publiquement sont mis en évidence en vert [19].

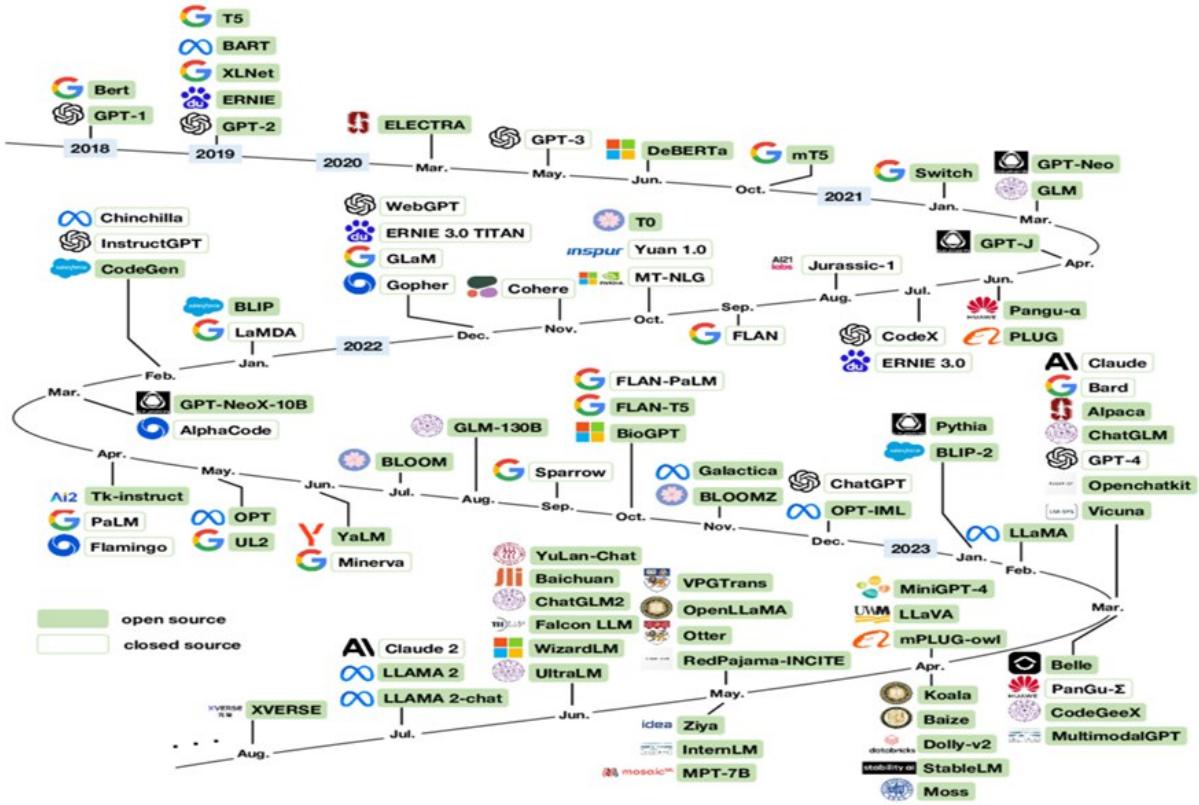


FIGURE 1.6 – Aperçu chronologique des grands modèles de langage (LLMs) .

Nous avons analysé les règles, recommandations et bonnes pratiques proposées par les développeurs de LLM open source, en nous basant sur leurs documentations officielles et sites comme Hugging Face [1]. Ces directives ont été structurées sous forme d'un arbre de décision, la figure 1.7 permettant de recommander le modèle le plus adapté selon les contraintes matérielles des machines (CPU, RAM, GPU).

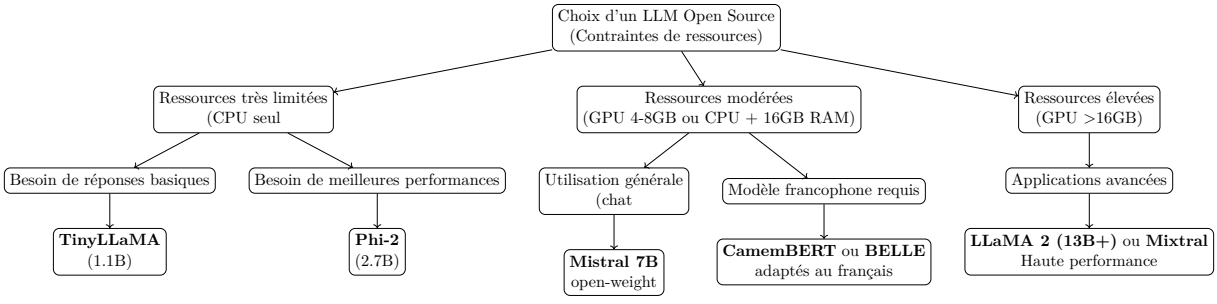


FIGURE 1.7 – Arbre de décision pour le choix d'un LLM open source selon les contraintes de ressources matérielles.

1.

1.3.5 Génération augmentée par récupération (RAG)

La génération augmentée par récupération (RAG) est une technique d'IA intelligente qui combine deux outils puissants : la recherche d'informations et la génération de texte. Imaginez un système capable de rechercher des faits ou des données pertinents, puis d'utiliser ces informations pour créer des réponses claires, précises et détaillées. La RAG est utilisée dans les chatbots, les assistants virtuels et d'autres outils d'IA pour fournir des réponses plus pertinentes et plus éclairées. C'est comme avoir un assistant extrêmement intelligent capable de dénicher et d'utiliser les informations appropriées [12]. La figure 1.8 présente l'architecture de (RAG) [31].

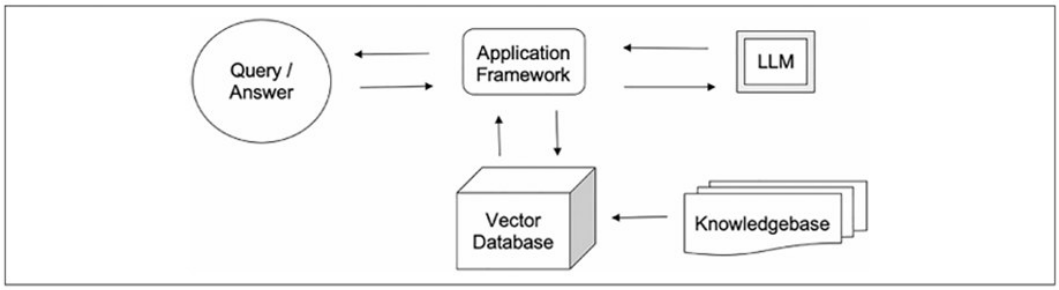


FIGURE 1.8 – Architecture de (RAG).

1.3.5.1 Type de génération augmentée par récupération

Le paradigme de recherche de la génération augmentée par récupération (RAG) est en constante évolution. Cette partie présente principalement son développement, que nous classons en trois catégories : RAG naïf, RAG avancé et RAG modulaire. Bien que le RAG naïf ait été une solution rentable et ait surpassé les modèles de langage natifs, il présentait encore de nombreuses limitations.

L'apparition du RAG avancé et du RAG modulaire vise à corriger ces insuffisances spécifiques et à améliorer ses performances.

Le paradigme de recherche RAG naïf représente la méthodologie initiale qui a gagné en notoriété peu après l'adoption généralisée de ChatGPT. Il comprend un processus traditionnel : indexation, recherche et génération [20].

Le RAG naïf

Le *RAG naïf* est souvent résumé comme un cadre en deux étapes : « **Récupérer** »–« **Lire** ».il repose sur trois phases principales : l'indexation, la récupération, puis la génération.

— **Indexation :** le processus d'extraction des données depuis la source et de création d'un index est généralement réalisé hors ligne. Il comprend notamment la génération d'embeddings (représentations vectorielles) à l'aide d'un modèle de codage – dont la taille des paramètres ne doit pas être excessive pour rester efficace. Une fois les embeddings obtenus, un index est construit : il associe les segments du corpus d'origine aux vecteurs correspondants sous forme de paires clé-valeur, facilitant ainsi les recherches ultérieures rapides et fréquentes [20].

— **Récupération :** à partir de la requête de l'utilisateur, le même modèle d'encodage que pour l'indexation est utilisé pour transformer la requête en vecteur. On calcule alors la similarité entre ce vecteur et ceux des documents indexés. Les K blocs de texte les plus similaires sont sélectionnés comme contexte pertinent pour la génération de la réponse [20].

— **Génération :** la requête et les documents récupérés sont combinés dans une nouvelle invite (*prompt*). Le grand modèle de langage est alors chargé de produire une réponse basée sur ce contexte. Selon les besoins de l'application, il peut être configuré pour répondre uniquement à partir des documents fournis, ou en utilisant également ses connaissances internes. En cas de dialogue multi-tours, l'historique des échanges peut également être inclus dans l'invite [20].

Le RAG avancé est apparu comme un nouveau paradigme avec des améliorations spécifiques pour résoudre certaines des lacunes du paradigme RAG naïf à la lumière des développements actuels dans le domaine du RAG.

Le RAG avancé

Le *RAG avancé* a émergé comme un nouveau paradigme visant à pallier certaines des limitations du *RAG naïf*, en s'appuyant sur les progrès récents dans le domaine de la génération augmentée par récupération.

Selon une étude récente, les techniques sophistiquées du RAG avancé peuvent être classées en trois catégories principales : les optimisations **pré-récupération**, **pendant la récupération** et **post-récupération**.

— **Pré-récupération :** cette étape prépare les données et les requêtes afin d'optimiser la récupération de l'information. Elle inclut des méthodes destinées à guider le récupérateur dans l'identification de documents pertinents, notamment à travers des mécanismes améliorés de recherche et d'évaluation [33].

— **Récupération :** plusieurs stratégies avancées sont déployées ici pour extraire les informations les plus pertinentes. Parmi celles-ci, on trouve le résumé spécifique à un domaine, la récupération de sous-graphes, l'intégration de la récupération avec des mécanismes de raisonnement ainsi que la distillation de l'attention [33].

— **Post-récupération :** cette phase vise à affiner les résultats récupérés afin d'en améliorer la pertinence. Des techniques telles que le reclassement, le filtrage, la distillation des connaissances, la sélection itérative de candidats et la classification de paires de séquences sont couramment utilisées [33].

Ces optimisations rendent le RAG avancé plus efficace, précis et pertinent pour des applications complexes. La figure 1.9 illustre la comparaison entre le RAG naif et le RAG avancé [33] .

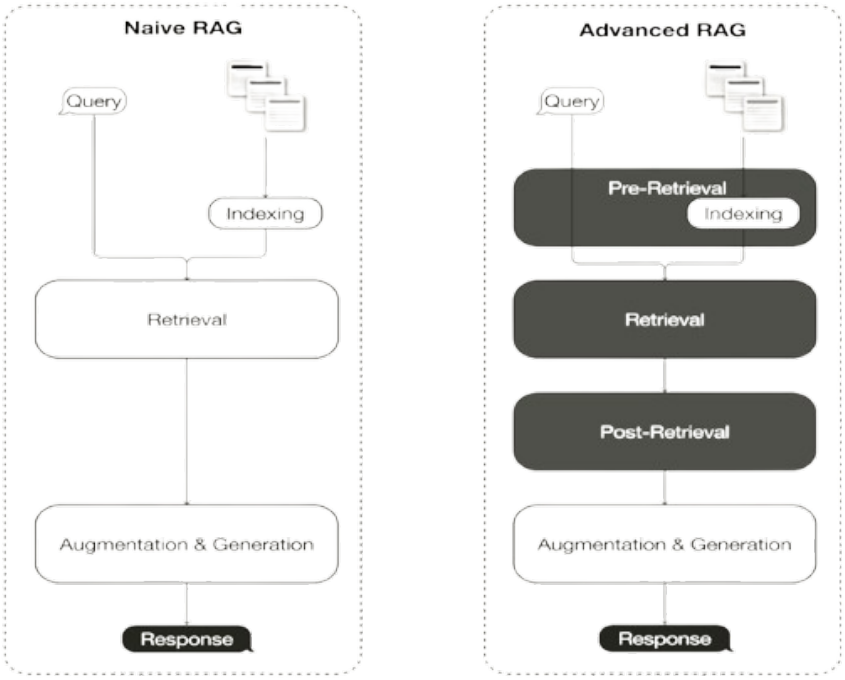


FIGURE 1.9 – Comparaison entre le RAG naïf et le RAG avancé .

Le RAG modulaire

Le *RAG modulaire* dépasse le cadre traditionnel du *RAG naïf* en introduisant une plus grande diversité et flexibilité dans le processus de traitement des données. Il intègre plusieurs techniques avancées, telles que l'ajout de modules de recherche spécialisés permettant d'affiner la récupération d'informations, ainsi que le réglage fin (*fine-tuning*) des modèles pour en améliorer la précision [20].

Face à certains défis spécifiques, de nouvelles structures modulaires et approches itératives ont également vu le jour, optimisant d'avantage les performances du système.

Aujourd'hui, le RAG modulaire tend à s'imposer comme le paradigme dominant. Il permet soit l'organisation d'un pipeline séquencé, soit un apprentissage de bout en bout sur plusieurs modules interconnectés, selon les besoins des cas d'usage [20].

1.3.6 Entraînement des grands modèles de Langage (LLMs)

L'entraînement des LLM suit plusieurs étapes essentielles pour assurer son efficacité. Il commence par la collecte et le traitement d'un large ensemble de données textuelles issues de diverses sources (livres, articles, sites Web, etc.), qui serviront de base d'apprentissage. Le modèle est ensuite formé selon une approche d'apprentissage non supervisé, où il apprend à prédire le mot suivant en fonction du contexte précédent, un processus connu sous le nom de modélisation du langage. Les LLM reposent sur des architectures neuronales avancées, telles que les Transformers, qui leur permettent d'analyser les relations complexes et les dépendances linguistiques. L'objectif est d'optimiser leurs paramètres afin d'améliorer la pertinence et la cohérence du texte généré. Cette

optimisation repose sur la descente de gradient stochastique (SGD) et ses variantes, associées à la rétropropagation, qui ajuste progressivement les poids du modèle en fonction des erreurs détectées [22].

La figure 1.10 présente le processus d'entrainement des grands modèles de langage.

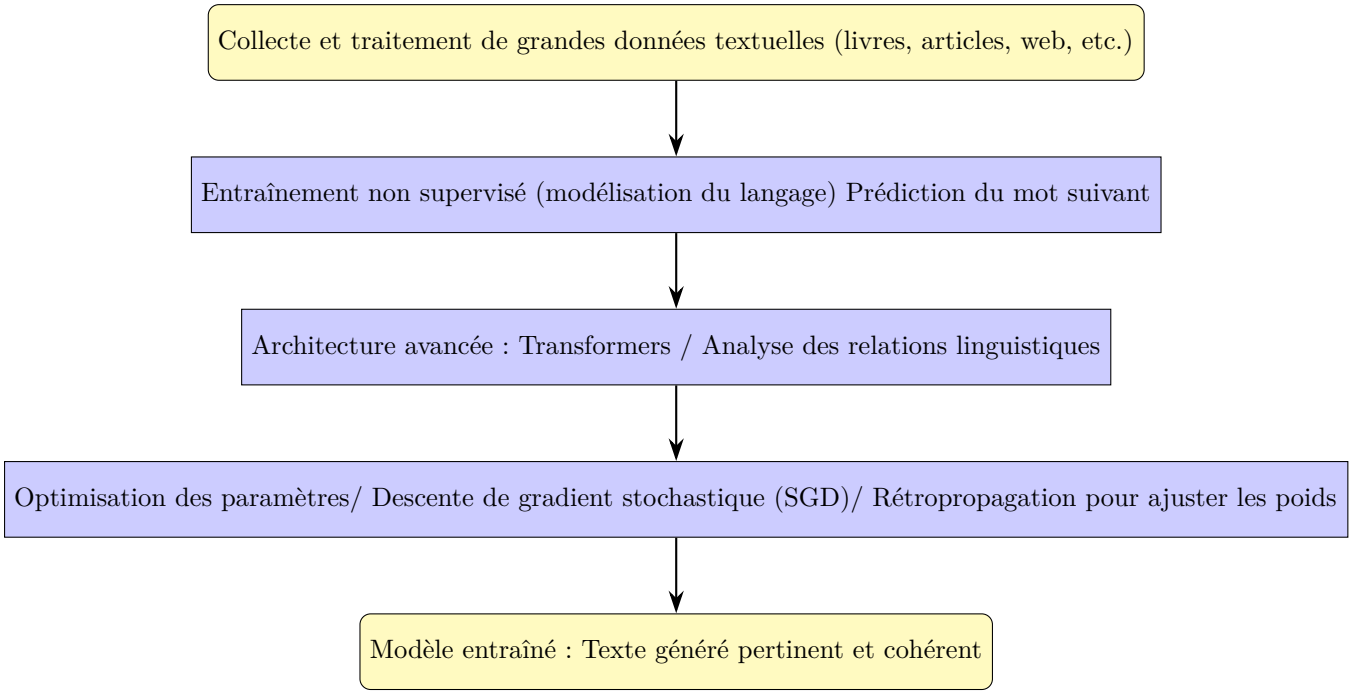


FIGURE 1.10 – Processus d'entraînement des grands modèles de langage (LLMs).

1.3.7 Adjustment fin (Fine Tuning)

Le réglage fin est une procédure qui utilise un modèle déjà entraîné comme point de départ. Cela implique de réaliser une formation complémentaire sur un ensemble de données plus limité et spécifique à un domaine pour optimiser les performances du modèle sur des missions spécifiques. Cette méthode exploite les savoirs préexistants du modèle, minimisant de ce fait la demande en données et en capacité de traitement. Par le biais du transfert des modèles et des caractéristiques acquis du modèle préalablement entraîné vers de nouvelles missions, l'affinage augmente la précision et l'efficacité tout en réduisant au minimum les ressources requises. Ce modèle est fréquemment employé dans le traitement du langage naturel pour des applications comme la classification de texte, l'étude de sentiments et les systèmes de questions-réponses [36]. La table 1.5 présente un exemple de réponse généré avant et aprés le (fine-tuning).

Before fine-tuning	**After fine-tuning**
You must provide personal and financial documents for the tax declaration.	*For your tax declaration in Algeria, you must provide :* *- Your national identity card* *- Proof of income (pay slips, rental income, etc.)* *- Certificates of deductible expenses* *- Any document specific to your situation (e.g., certificate of tax residence)*

TABLE 1.5 – Exemple de réponse générée avant et après adjustment fin (fine-tuning).

1.3.8 Applications de grands modèles de langage

Les grands modèles de langage (LLM) ont trouvé des applications dans un large éventail de domaines, dépassant les limites traditionnelles de ce que l'apprentissage automatique est traditionnellement connu pour faire. Leur capacité exceptionnelle à comprendre et à générer du texte de type humain a donné lieu à des cas d'utilisation innovants dans plusieurs secteurs. Les diverses applications des grands modèles de langage sont présentées dans la figure 1.11 [51].

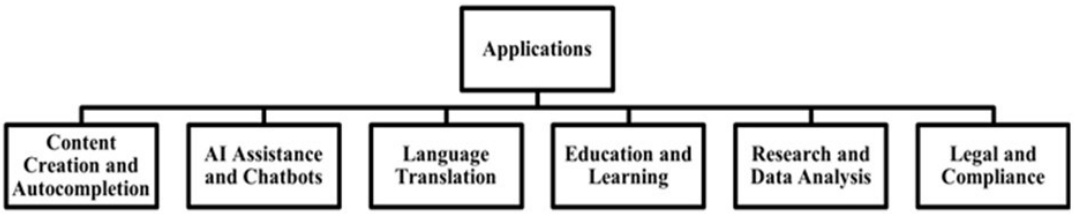


FIGURE 1.11 – Quelques applications des (LLMs) .

— **Génération de contenu et auto-complétions** : création d'articles, de poésie, de code, et amélioration de la complétion automatique dans les IDE.
— **Agents conversationnels** : développement de chatbots intelligents capables de tenir des conversations naturelles, utilisés dans le service client, l'assistance en santé mentale, etc.
— **Traduction automatique** : amélioration des traductions en prenant en compte le contexte, les expressions idiomatiques et les références culturelles (ex. : M2M-100 de Facebook AI).
— **Instruction et formation** : soutien à l'apprentissage personnalisé à travers la génération de questions, l'explication de notions complexes et un retour individualisé.
— **Recherche et analyse de données** : résumé d'articles scientifiques, extraction d'informations pertinentes, génération d'hypothèses.
— **Droit et conformité** : analyse de documents juridiques, révision de contrats et vérification de conformité réglementaire.

[51]

1.3.9 Défis des grands modèles de langage (LLMs)

Les LLM sont rapidement passés d'une absence d'existence à une présence omniprésente dans le domaine de l'apprentissage automatique en quelques années seulement. Leur extraordinaire capacité à générer du texte qui ressemble à celui d'un humain a attiré une attention considérable et des applications dans de nombreux domaines. Cependant, cette augmentation soudaine de ces dépendances technologiques à plus fort impact a également révélé de nombreux défis et préoccupations. Dans cette discussion, nous examinerons les défis importants liés aux LLM.

- **Contenu toxique :** les LLMs peuvent générer des propos haineux ou nuisibles, amplifiant les biais présents dans les données d'apprentissage [38].
- **Hallucination :** production d'informations erronées ou fictives, dues à des contradictions ou à des incohérences entre données pré-entraînées et nouvelles données [38].
- **Biais :** reproduction de stéréotypes sociaux liés à la race, au genre, à la religion, etc. [38].
- **Complexité et échelle des données :** les vastes corpus d'entraînement sont difficiles à maîtriser, ce qui soulève des questions de qualité et de biais [43].
- **Accès open-source et langues peu représentées :** peu de modèles accessibles au public, avec une prédominance de l'anglais, négligeant de nombreuses langues à faibles ressources [38].
- **Sensibilité à la tokenisation :** la décomposition du texte en tokens peut fortement influencer la compréhension et la génération, rendant le modèle sensible à de petites variations d'entrée [43].

1.4 Conclusion

L'intelligence artificielle générative a transformé la façon dont les machines comprennent et interagissent avec le langage humain. Ce chapitre s'est penché sur deux éléments cruciaux : le traitement du langage naturel (NLP) et les grands modèles de langage (LLM) Le traitement du langage naturel déchiffre, interprète et produit du texte à travers diverses étapes et composantes, avec des usages variés tels que la traduction automatique et l'analyse de sentiments. Les modèles de langage de grande envergure, grâce à des structures avancées, génèrent des réponses précises et contextuelles. Nous avons aussi traité de la génération augmentée par récupération (RAG), qui améliore la précision des modèles en utilisant des bases de données externes, ainsi que de l'ajustement fin (fine-tuning), une méthode qui adapte un modèle préformé à des besoins spécifiques pour optimiser sa performance. Dans le chapitre suivant, nous explorerons les agents conversationnels intelligents (chatbots).

2

Construction d'agents conversationnels intelligents

Table de sommaire

2.1 Introduction

Les progrès récents dans le domaine de l'intelligence artificielle (IA), en particulier concernant le traitement du langage naturel (NLP), ont mené à la naissance de chatbots intelligents, souvent désignés comme tels. Ces systèmes ont progressé de modèles basés sur des règles simples à des solutions élaborées basées sur l'intelligence artificielle, transformant radicalement les interactions entre l'homme et la technologie. Les chatbots, qui gagnent en popularité dans différents domaines tels que la finance et le service clientèle, sont capables d'améliorer l'expérience utilisateur en offrant une aide efficace et instantanée. Leur aptitude à saisir et à gérer le langage humain avec une exactitude remarquable en fait des instruments indispensables pour perfectionner l'assistance à la clientèle et enrichir les activités commerciales.

Dans ce chapitre, nous exposerons les principes fondamentaux et les méthodologies qui constituent la base du développement des chatbots.

2.2 Chatbots

Un chatbot est défini formellement comme suit :,

Définition 1 : Le chat est un moyen de communication qui facilite l'envoi de messages écrits et audio à travers des réseaux informatiques, mobiles ou encore Internet. Bien que cette interaction se fasse habituellement entre personnes, elle peut aussi concerner un programme automatisé destiné à accomplir des tâches précises en fonction des données d'entrée fournies, connu sous le nom de *chatbot*. On appelle également ces programmes des chatterbots, imbots, talkbots, systèmes de dialogue par chat ou agents conversationnels.

Au cours des dernières années, les chatbots ont démontré leur efficacité et performance en tant qu'applications. Ils ont la capacité d'examiner et de gérer les dialogues humains, qu'ils soient sous forme écrite ou verbale, afin de communiquer avec les utilisateurs via des outils numériques. En raison de leur développement rapide, ils sont aujourd'hui largement utilisés dans divers domaines, qu'ils soient du secteur public ou privé.

Les chatbots ne se limitent pas à répondre aux questions des utilisateurs, ils ont également la capacité d'analyser ces requêtes et, dans certaines situations, de prendre des décisions basées sur les données traitées. De surcroît, ils possèdent la faculté d'apprendre à partir des échanges avec les utilisateurs, ce qui améliore leurs performances progressivement[4].

On identifie essentiellement deux catégories de chatbots :

- **Les chatbots interactifs :** couramment déployés sur les plateformes web et les médias sociaux, ils utilisent des technologies de traitement du langage naturel (NLP) pour analyser les entrées textuelles et fournir les réponses les plus appropriées.
- **Les chatbots rationnels :** ils consignent les interrogations et demandes des utilisateurs tout en s'appuyant sur des bases de données externes pour formuler leurs réponses.

De plus, un chatbot incarné propose une interaction plus immersive en présentant une apparence humaine et en imitant un échange naturel, ce qui rend l'interaction plus lisse et authentique.

Les chatbots sont surtout employés pour dispenser une aide, recueillir des renseignements et recevoir des réponses immédiates du service client, contribuant ainsi à l'amélioration de l'expérience utilisateur et de la disponibilité des services [4] .

2.2.1 Historique de chatbots

1950 : alan Turing formule le *test de Turing* (« Les machines peuvent-elles penser ? »), marquant l'essor du concept de chatbot.

1966 : création d'**ELIZA** par Joseph Weizenbaum, un programme simulant un psychothérapeute en reformulant les déclarations de l'utilisateur sous forme de questions. Basé sur la correspondance de modèles et des règles simples, il réussit à convaincre certains utilisateurs malgré ses limites.

1972 : PARRY, développé par Kenneth Colby, introduit une personnalité simulée inspirée d'un patient atteint de schizophrénie paranoïde. Il constitue une avancée significative par rapport à ELIZA.

1995 : lancement de **ALICE** (Artificial Linguistic Internet Computer Entity), un chatbot utilisant le langage de balisage AIML pour structurer ses réponses. Il remporte le prix Loebner en 2000, 2001 et 2004.

2001 : SmarterChild, chatbot intégré aux services de messagerie (AIM, MSN), démocratise l'usage des agents conversationnels auprès du grand public.

Depuis 2010 : apparition des **assistants virtuels intelligents** : **Siri** (Apple), **Cortana** (Microsoft), **Alexa** (Amazon), **Google Assistant**, **IBM Watson**, représentant le sommet actuel des technologies conversationnelles [2].

La figure 2.1 montre une brève chronologie des chatbots depuis leur première apparition dans les années 1960 jusqu'aux modèles de langage utilisés aujourd'hui [7].

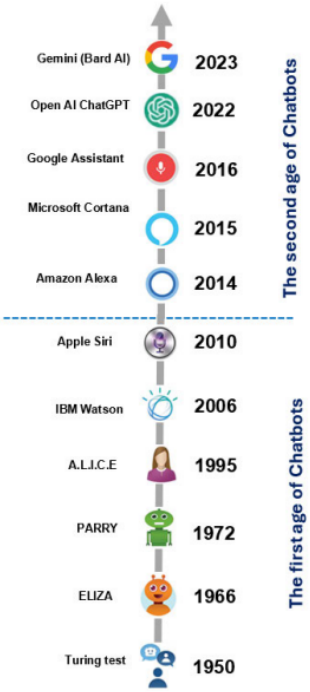


FIGURE 2.1 – Brève chronologie des chatbots .

2.2.2 Architecture

Dans cette section, nous présentons une conception architecturale générale et analysons en détail les éléments clés de chaque composant. L'architecture d'un chatbot repose généralement sur cinq composants principaux : une interface utilisateur, un module de compréhension du langage naturel (NLU), un module de gestion du dialogue (DM), un backend et un module de génération de réponses (RG). Comme illustré dans la figure 2.2, cette architecture représente le flux d'interaction entre l'utilisateur et le chatbot, depuis la saisie de la requête jusqu'à la génération de la réponse, en passant par l'analyse du langage et la gestion du dialogue [2].

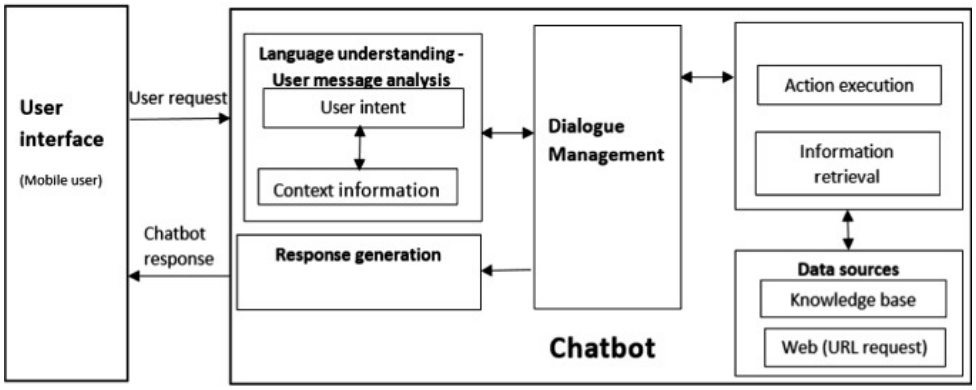


FIGURE 2.2 – Architecture générale du chatbot .

L'architecture d'un chatbot se compose de plusieurs modules essentiels :

— **Interface utilisateur** : assure la communication via des canaux textuels ou vocaux.

— **Reconnaissance automatique de la parole (ASR)** : convertit l'audio en texte pour les assistants vocaux.
— **Compréhension du langage naturel (NLU)** : analyse la requête utilisateur pour identifier les intentions et extraire les entités clés.
— **Gestionnaire de dialogue (DM)** : maintient le contexte conversationnel et coordonne les réponses.
— **Backend** : fournit les données nécessaires à la réponse.
— **Génération de réponse (RG)** : élabore une réponse pertinente et naturelle à transmettre à l'utilisateur.

Ce pipeline permet une interaction fluide et cohérente entre l'humain et la machine [24].

2.2.3 Classification des chatbots

Les chatbots peuvent être classés selon plusieurs critères : le mode d'interaction, l'approche d'implémentation, l'objectif et le domaine de connaissance. comme illustré dans la figure 2.3 [10].

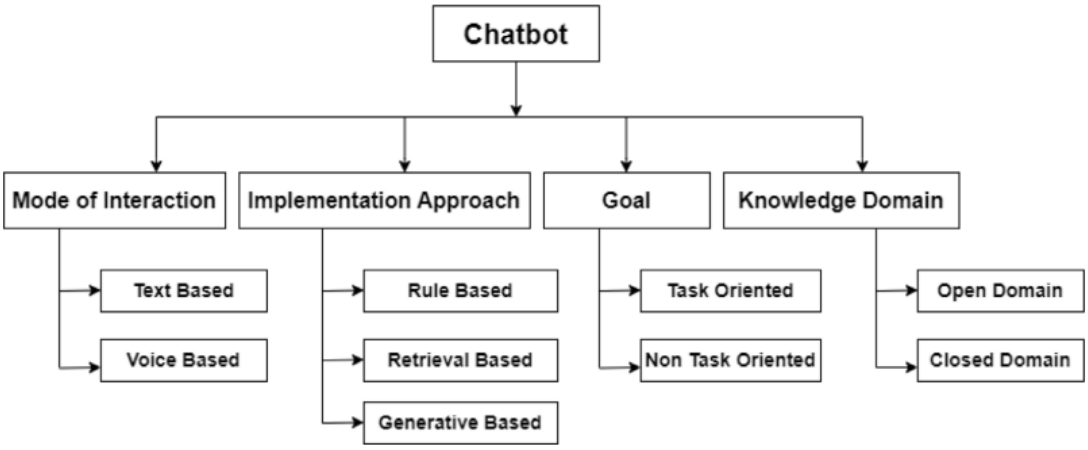


FIGURE 2.3 – Classification des Chatbots .

2.2.3.1 Mode d'Interaction

Les chatbots peuvent être classés selon leur mode d'interaction en deux grandes catégories : les chatbots textuels et les chatbots vocaux.

— **Chatbots textuels** : conçus pour gérer des messages écrits, ils sont particulièrement adaptés aux PC, appareils mobiles et applications de messagerie. Ils conviennent aux interactions complexes nécessitant du raisonnement ou des vérifications, car l'utilisateur peut suivre et contrôler le déroulement du dialogue [41].
— **Chatbots vocaux** : à l'image d'Alexa ou Siri, ces agents dialoguent par la voix. Ils offrent un fonctionnement mains libres, ce qui les rend utiles pour des tâches telles que le contrôle d'appareils domestiques intelligents ou la lecture de musique, facilitant ainsi le multitâche [41].

2.2.3.2 Objectif

Les chatbots se divisent en deux catégories principales : orientés tâches et non orientés tâches.

- **Les chatbots orientés tâches** : ces chatbots excellent dans l'accomplissement de tâches spécifiques, mais disposent de peu de connaissances générales. Ils suivent un flux de conversation prédéfini et nécessitent des données d'entraînement personnalisées [41].
- **Les chatbots non orientés tâches** : ces systèmes visent à établir une conversation naturelle. Ils peuvent être :
  - **Génératifs** : capables de créer de nouvelles réponses.
  - **Basés sur la récupération** : sélectionnent une réponse parmi un ensemble de réponses prédéfinies.

  Ces chatbots requièrent une grande quantité de données d'entraînement et sont moins performants dans l'exécution de tâches précises [41].

  Deux sous-types de chatbots non orientés tâches peuvent être distingués :
  - **Orienté information** : similaire à une FAQ, il récupère des informations pré-enregistrées [41].
  - **Orienté conversation** : il vise à maintenir un dialogue continu et engageant avec l'utilisateur [41].

2.2.3.3 Domaine de Connaissance

Les chatbots peuvent être classés selon l'étendue de leur domaine de connaissance :

- **Domaine ouvert** : capables de répondre à une grande variété de sujets sans restrictions spécifiques [24].
- **Domaine fermé** : spécialisés dans un domaine précis et limités aux questions relatives à ce domaine particulier [24].

2.2.3.4 Approche d'implémentation

Les approches d'implémentation des chatbots se divisent en trois grandes catégories :

- **Approche basée sur des règles** : ces chatbots utilisent des ensembles de règles prédéfinies pour sélectionner une réponse en fonction de la forme lexicale du texte saisi. Leur base de connaissances est codée manuellement et structurée à l'aide de modèles conversationnels. Plus la bibliothèque de règles est riche, plus le chatbot peut gérer de variations de saisie [41].
- **Approche basée sur la récupération** : ces modèles consultent des bases de données ou des API pour extraire des réponses à partir d'un ensemble prédéfini. Ils s'appuient sur des méthodes simples de correspondance de modèles ou sur des techniques plus avancées d'apprentissage automatique. Les réponses ne sont pas générées, mais choisies parmi des options existantes à l'aide d'heuristiques [2].

— **Approche basée sur la génération** : cette approche utilise des algorithmes d'apprentissage profond pour générer dynamiquement des réponses à partir du contexte des conversations. Elle se distingue par sa capacité à produire des réponses plus naturelles et adaptées, rapprochant ainsi les chatbots du comportement humain [1].

La table 2.1 illustré la comparaison des approches d'implémentation des chatbots.

Critère	**Basée sur des règles**	**Basée sur la récupération**	**Basée sur la génération**
Principe	Ensemble de règles prédéfinies, sélection de réponse selon la forme lexicale	Recherche dans bases de données ou API, sélection de réponses existantes via heuristiques ou apprentissage	Génération dynamique de réponses via apprentissage profond
Base de connaissances	Codée manuellement et structurée en modèles conversationnels	Ensemble de réponses prédéfinies accessibles via API ou bases	Modèles d'IA entraînés sur de larges corpus
Avantages	Contrôle précis sur les réponses, fiabilité pour les cas courants	Plus flexible que les règles seules, accès à un large éventail de réponses	Réponses naturelles et adaptées, capacité à gérer des conversations complexes.
Limites	Faible flexibilité, difficulté avec les requêtes non anticipées	Pas de génération de nouvelles réponses, dépendance à la qualité des données	Nécessite beaucoup de données, risques d'incohérences et d'erreurs.
Exemples d'usage	FAQ simples, support technique basique	Chatbots avec accès à des bases clients ou produits	Assistants vocaux, chatbots conversationnels avancés

TABLE 2.1 – Comparaison des approches d'implémentation des chatbots.

2.2.4 Applications de chatbots

— **Environnements éducatifs :** Les chatbots offrent un soutien personnalisé et du contenu pédagogique, aidant à compenser la diminution d'aide des enseignants dans les établissements d'enseignement supérieur, ce qui peut réduire les abandons [1].

— **Service client :** Ils améliorent l'expérience client en fournissant un service disponible 24h/24, réduisant les temps d'attente et proposant des réponses immédiates, même hors des heures de bureau [1].

— **Santé :** Les chatbots fournissent des informations personnalisées, aident au diagnostic, recommandent des traitements et soutiennent les thérapies ainsi que le suivi d'activité

physique [1].

— **Robotique :** l'interface en langage naturel des chatbots sert à commander des robots physiques, comme le robot autonome *KAMRO* [1].

— **Cas d'utilisation industrielle :** ils sont largement adoptés dans des secteurs comme la banque, l'agroalimentaire, la logistique, pour automatiser des tâches et améliorer les services [1].

En plus de cela, ils sont intégrés dans des assistants virtuels mobiles (comme Siri), des systèmes de tutorat intelligents, des environnements domestiques intelligents, ainsi que des systèmes de navigation et de contrôle embarqués dans les véhicules.

2.2.5 Les défis de chatbots

— **Maintenir l'interaction du système :** Un système interactif doit réagir rapidement et efficacement aux entrées des utilisateurs pour garder leur intérêt et assurer une expérience fluide, en fournissant des retours rapides et en réduisant les temps de réponse [41].

— **Confidentialité des données :** La protection des données sensibles est primordiale. Il est essentiel d'appliquer des mesures de sécurité robustes et d'assurer transparence et conformité légale dans la collecte et l'utilisation des données utilisateur [41].

— **Utilisation des méthodes NLU avec prise en compte de la sémantique :** les techniques de compréhension du langage naturel doivent interpréter correctement le contexte et le sens pour saisir l'intention des utilisateurs et fournir des réponses appropriées, favorisant ainsi des interactions naturelles et efficaces [41].

— **Difficulté de formation :** Former les utilisateurs à un nouveau système peut être complexe, notamment à cause de la complexité du système, du manque de matériel de formation ou des connaissances requises, ce qui peut freiner l'adoption et provoquer de la frustration [41].

2.3 Travaux connexes

Avec la croissance de l'intelligence artificielle, d'internet et des sites de réseautage social, les chatbots ont suscité beaucoup d'intérêt[34]. L'utilisation généralisée des chatbots a provoqué une révolution significative dans le domaine en constante évolution du service à la clientèle, modifiant radicalement la nature des interactions avec les utilisateurs. Les chatbots sont omniprésents et apparaissent dans un large éventail de contextes, y compris le marketing, le service à la clientèle, les soins de santé, l'apprentissage en ligne et le commerce électronique [54] .Cette section passe en revue les travaux de recherche axés sur le développement de chatbots intelligents pour améliorer le service d'assistance client dans les organisations.

Ils participent de façon importante à l'amélioration de l'expérience client en offrant une aide efficace et immédiate disponible 24h/24. Ils suscitent également une attention croissante dans divers domaines en raison de leur capacité à automatiser les demandes récurrentes, à améliorer l'efficacité des services et à accroître la satisfaction des utilisateurs [48].

Les solutions basées sur l'IA sont conçues pour comprendre, anticiper et répondre aux besoins des clients avec une précision et une rapidité exceptionnelles. Cela permet aux entreprises d'offrir des expériences hautement personnalisées et de personnaliser les services en fonction des préférences individuelles[3].

De plus, les chatbots une qualité de réponse constante , grâce à leur disponibilité constante, ils diminuent également les temps d'attente . L'atout le plus marquant est que les chatbots allègent la charge administrative des institutions publiques, en libérant les employés des tâches routinières[8].

L'IA peut créer une réflexion claire, des analyses intelligentes et l'automatisation des processus. Les chatbots alimentés par l'IA ont montré leur potentiel dans le soutien aux clients. Par conséquent, cela semble être un outil spécial que les entreprises peuvent utiliser pour gagner du temps [14].

Les chatbots transforment le service client en améliorant l'expérience utilisateur et l'efficacité opérationnelle. Trente pour cent des entreprises, y compris celles du secteur bancaire et de l'assurance, ont adopté cette technologie, selon Gartner. La numérisation a entraîné une hausse de leur adoption en Norvège [56].

Les chatbots dépassent les attentes des clients en matière de réactivité et de facilité et offrent un accès rapide à l'information et un service personnalisé Telmi, le chatbot de Telenor qui gère des millions d'intentions et rationalise les transactions, est un exemple remarquable. L'informatique conversationnelle s'est développée depuis les années 1960 avec ELIZA , ce qui a permis d'améliorer le service à la clientèle [56].

L'automatisation des réponses aux questions fréquentes permet de réduire considérablement le temps d'attente des clients et d'accroître leur satisfaction. Les résultats démontrent que les chatbots hybrides, basés à la fois sur des règles et sur l'intelligence artificielle, surpassent les modèles reposant uniquement sur l'un ou l'autre de ces facteurs[18].

Dans le cadre de ce projet, il est donc pertinent de s'interroger sur les approches actuellement employées dans les chatbots d'assistance client. La majorité des solutions simples utilisent encore des chatbots uniquement basés sur des règles, limitant ainsi leur capacité à comprendre des requêtes complexes ou non anticipées. À l'inverse, les chatbots entièrement basés sur l'intelligence artificielle peuvent souffrir d'un manque de contrôle sur la cohérence des réponses et nécessitent un volume important de données d'entraînement. Une étude d'Uzoka souligne que les chatbots alimentés par l'IA peuvent réduire les délais de réponse et les coûts opérationnels tout en améliorant la satisfaction client, mais nécessitent une approche équilibrée pour gérer les interactions complexes [53].n'ont pas reçu suffisamment d'attention dans les études antérieures. Établir une telle relation est essentielle pour créer des modèles de chatbots capables de gérer efficacement le langage naturel.

Notre chatbot hybride combine règles prédéfinies et intelligence artificielle pour offrir des réponses précises et flexibles. Il supporte plusieurs langues, dont l'arabe, l'arabizi et l'anglais, assurant une interaction naturelle pour un public varié. Conçu pour un domaine spécifique, il optimise l'expérience utilisateur tout en déchargeant les équipes support. Peu de chatbots exis-

tants intègrent cette approche multilingue hybride, surtout avec le support de l'arabe, ce qui rend notre solution innovante et adaptée aux besoins actuels. Un besoin réel encore peu couvert dans la littérature et les applications courantes.

2.4 Conclusion

Ce chapitre aborde l'histoire et la pertinence actuelle des chatbots, ces agents conversationnels intelligents qui utilisent l'intelligence artificielle pour reproduire la parole humaine. Il explore leurs types, leur architecture générale et leur évolution, tout en abordant des questions importantes.

Nous présenterons nos suggestions de solutions dans le prochaine chapitre.

3

Chatbot hybride pour support client

Table de sommaire

3.1 Introduction

Face aux limites des systèmes classiques de gestion des demandes administratives, notre projet propose une solution innovante combinant des règles prédéfinies et des modèles de langage pré-entraînés, notamment T5. Cette approche vise à améliorer l'efficacité de l'assistance aux citoyens en automatisant les réponses, personnalisant les interactions et guidant l'utilisateur dans ses démarches.

Notre solution repose sur trois axes principaux :

— Une base de connaissances organisée, comprenant intentions, exemples et réponses administratives.

— Un générateur de texte intelligent capable de répondre à des questions complexes, même sans correspondance directe dans la base.

— Une réponse structurée détaillée, incluant informations sur les services, étapes, documents, contacts et dates, facilitant la compréhension et l'action de l'utilisateur.

Ce chatbot intelligent s'appuie sur le traitement automatique du langage naturel (NLP), l'apprentissage machine et des bases relationnelles. Il est conçu pour comprendre les demandes des utilisateurs, extraire les informations essentielles et fournir des réponses adaptées dans le contexte administratif algérien, offrant ainsi une aide instantanée, personnalisée et automatisée.

3.2 Exemple de motivation

Dans cette section, nous présentons notre exemple motivant.

3.2.1 Scénario

Dans le contexte actuel, les services fiscaux sont confrontés à une charge de travail accrue en raison de l'augmentation continue des demandes et de la complexité des procédures. Le personnel a du mal à répondre efficacement aux questions récurrentes, ce qui entraîne des retards, de la frustration et une qualité de service réduite. De plus, l'absence de communication claire et de système centralisé entraîne une augmentation des malentendus entre les utilisateurs et la direction. Cette situation affecte non seulement les citoyens en quête d'informations fiables, mais aussi les salariés, qui sont confrontés à une pression accrue et à un risque d'épuisement professionnel. Compte tenu de cette réalité, l'adoption d'un système intelligent capable de traiter automatiquement les demandes conjointes est devenue une nécessité pour améliorer la rapidité de réponse, la précision et l'efficacité des services fiscaux. La figure 3.1 illustré la Frustration des usagers et la surcharge des employés dans un processus administratif.

FIGURE 3.1 – Example de Frustration des usagers et la surcharge des employés dans un processus administratif .

Prenons l'exemple de Mehdi, un homme d'affaires qui souhaite enregistrer sa société et obtenir un numéro d'identification fiscale. Après une longue attente au guichet, il a reçu des informations incomplètes sur les documents à soumettre. Ses efforts furent vains : il dut revenir plusieurs fois, découvrant à chaque fois qu'il devait fournir de nouveaux documents ou des modifications à son emploi du temps qui n'avaient pas été signalées. Ces escarmouches inutiles sont le résultat direct de procédures peu claires et d'une mauvaise communication.

Du côté du personnel, Sarah, représentante du service client, fait face à une lourde charge de travail au quotidien. Vous recevez constamment des questions récurrentes sur les procédures d'enregistrement fiscal ou la délivrance de certificats, et vous travaillez sous pression. Ce contexte stressant affecte sa concentration, l'amenant parfois à donner des réponses inexactes ou incomplètes.

Ce déséquilibre affecte également la communication interne. Ahmed, un employé informatique, est constamment interrompu par ses collègues qui lui posent des questions : « Comment dois-je effectuer cette procédure ? » Où puis-je trouver un tel modèle ? Ces interruptions constantes, que ce soit par téléphone, minent sa productivité et génèrent un stress croissant. Ce climat oppressant la même poussé à envisager sérieusement de quitter son poste.

La Figure 3.2 illustré la perte de temps subie par les citoyens et les employés du département des impôts et des finances, mettant en évidence les inefficacités dans la prestation de services.

Les citoyens sont confrontés à de longues files d'attente et à des informations contradictoires au sein du département des impôts et des finances, ce qui provoque de la frustration, tandis que la charge de travail excessive du personnel réduit la qualité du service.

Cas 1 – Mehdi (Homme d'affaires) :

— Visite le bureau **3 fois/semaine**.

— Attend **45 minutes** à chaque fois.

— Le service prend **15 minutes**.

— **Temps total perdu :** $3 \times 60 = 180$ minutes (3 heures).

Cas 2 – Sarah (Service client) :

— Poignées **50 requêtes/jour**. requête prend **5 minutes**.

— **Durée totale :** $50 \times 5 = 250$ minutes (4.2 heures) tous les jours.

FIGURE 3.2 – Exemple de perte de temps dans les services fiscaux et financiers.

Ce cycle de frustration, tant pour les citoyens que pour les employés, met en évidence la nécessité d'une solution plus efficace et plus accessible pour améliorer la prestation de services et la communication au sein de l'administration fiscale.

1. **C1 :** Longues files d'attente et perte de temps pour les citoyens.
2. **C2 :** Informations incomplètes ou contradictoires nécessitant plusieurs visites.
3. **C3 :** Un personnel surchargé qui gère des requêtes répétitives, source de stress
4. **C4 :** Mauvaise communication entre les citoyens et les autorités fiscales.

Dans le but de permettre aux usagers de tirer pleinement parti des services disponibles et de répondre à leurs besoins de manière rapide et efficace, il devient indispensable de mettre en place une solution de communication accessible à tout moment, 24 heures sur 24, 7 jours sur 7. L'adoption de systèmes intelligents et interactifs permettrait non seulement d'assurer un soutien continu, mais aussi de réduire considérablement les allers-retours inutiles, d'économiser du temps et d'optimiser l'efficacité des démarches administratives, tant pour les citoyens que pour le personnel administratif.

3.2.2 Notre proposition

Nous proposons Tawjih, un chatbot intelligent et hybride alliant les approches basées sur des règles et celles fondées sur l'intelligence artificielle, conçu pour répondre de manière rapide, précise et contextuelle aux préoccupations des citoyens et des agents concernant les services fiscaux.

Tawjih est capable de traiter efficacement les questions récurrentes liées aux procédures administratives, aux documents exigés, aux délais à respecter ainsi qu'aux informations de contact. Grâce à sa capacité à comprendre l'intention des utilisateurs et à tirer parti du contexte des échanges précédents, il fournit des réponses personnalisées, cohérentes et pertinentes.

Ce système innovant permet de désengorger les services d'assistance, de réduire la charge de travail liée aux tâches répétitives, et de limiter les déplacements inutiles des citoyens vers les centres fiscaux.

En plus de son interface conviviale, Tawjih intègre un tableau de bord interactif destiné aux agents de l'administration fiscale, leur permettant de suivre en temps réel les requêtes des usagers et d'accéder à des données utiles pour améliorer la qualité du service public.

Par son efficacité et sa flexibilité, Tawjih représente une avancée concrète vers une administration plus moderne, plus accessible et centrée sur les besoins des usagers.

3.2.2.1 Services de chatbot hybrides

Notre solution est un chatbot IA contextuel conçu pour aider les citoyens à répondre à leurs questions fiscales. Il traite les questions fréquentes telles que les délais, les documents requis et les étapes de service grâce à une logique basée sur des règles, et utilise l'IA et la compréhension du contexte pour les demandes plus complexes. En se connectant à une base de connaissances structurée, le chatbot fournit des réponses précises et personnalisées, garantissant un accès rapide et efficace aux informations fiscales.

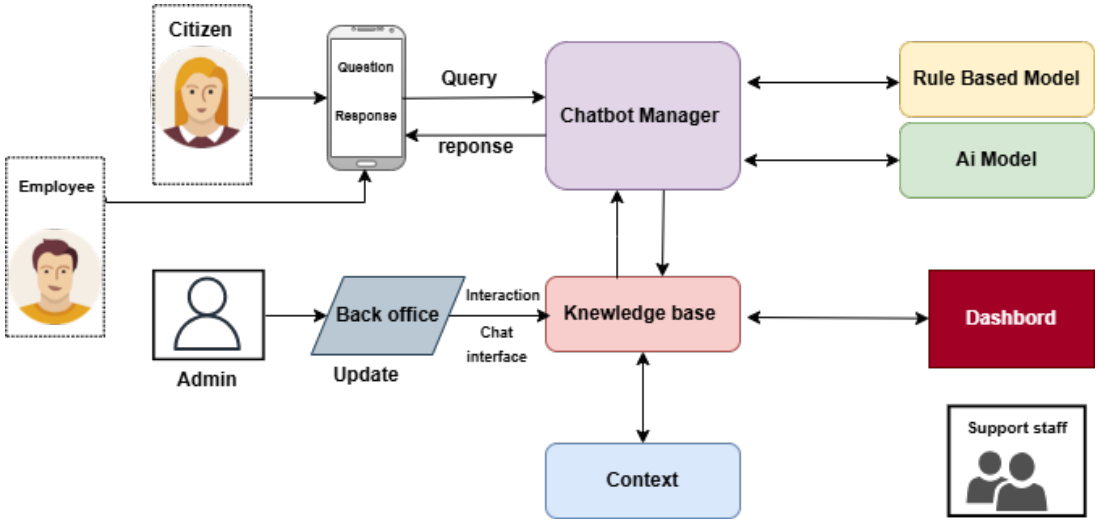


FIGURE 3.3 – Aperçu de notre proposition.

Comme le montre la figure **??**, le processus démarre lorsque l'utilisateur envoie une requête au chatbot via l'application de messagerie **Tawjih**. Le gestionnaire de chatbot, chargé de gérer et de traiter les requêtes entrantes, reçoit le message de l'utilisateur et le transmet au modèle basé sur des règles ou au modèle d'IA. Ces modèles génèrent une réponse et consultent la base de connaissances, qui contient toutes les informations à jour sur le contexte actuel, gérée par BackOfficeAdmin. La réponse est ensuite renvoyée au gestionnaire de chatbot, qui la formate et la transmet à l'utilisateur sous forme de réponse précise.

Le tableau de bord du support surveille la fréquence et les tendances des requêtes des utilisateurs afin d'améliorer la qualité et la réactivité du service.

Le système de chatbot hybride comprend les services principaux suivants :

Gestionnaire de chatbot : gère et achemine les messages entrants.

Générateur de réponses : utilise à la fois des modèles basés sur des règles et d'IA pour créer des réponses.

Base de connaissances : stocke toutes les informations contextuelles et mises à jour.

BackOfficeAdmin : permet aux administrateurs de gérer et de mettre à jour les données contextuelles.

Tableau de bord du support : permet au personnel de support de suivre et d'analyser les demandes des utilisateurs.

3.3 Analyse de domaine

Cette phase constitue une étape essentielle dans la mise en oeuvre du projet. Elle se décline en quatre volets principaux : la collecte des données, leur analyse, leur classification, et leur intégration. Comme le montre la figure 3.4.

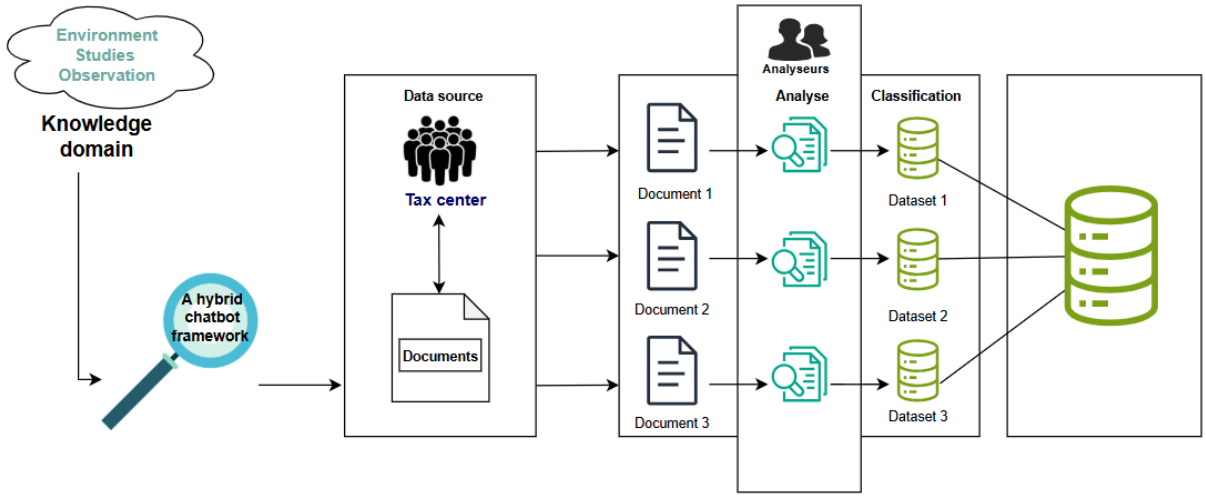


FIGURE 3.4 – Analyse de domaine.

Pour le développement du chatbot intelligent dédié au domaine fiscal, nous nous sommes appuyés sur des sources fiables, incluant des agents de l'administration fiscale à différents niveaux (cadres, agents d'accueil, etc.), ainsi que sur des documents officiels et des informations extraites du site officiel de l'administration.

Ces dossiers ont ensuite été traités et classés selon différents domaines tels que le paiement, les impôts généraux, l'enregistrement et l'identification.

Enfin, elles ont été intégrées dans une base de données unique et consolidée.

3.4 Architecture générale du système

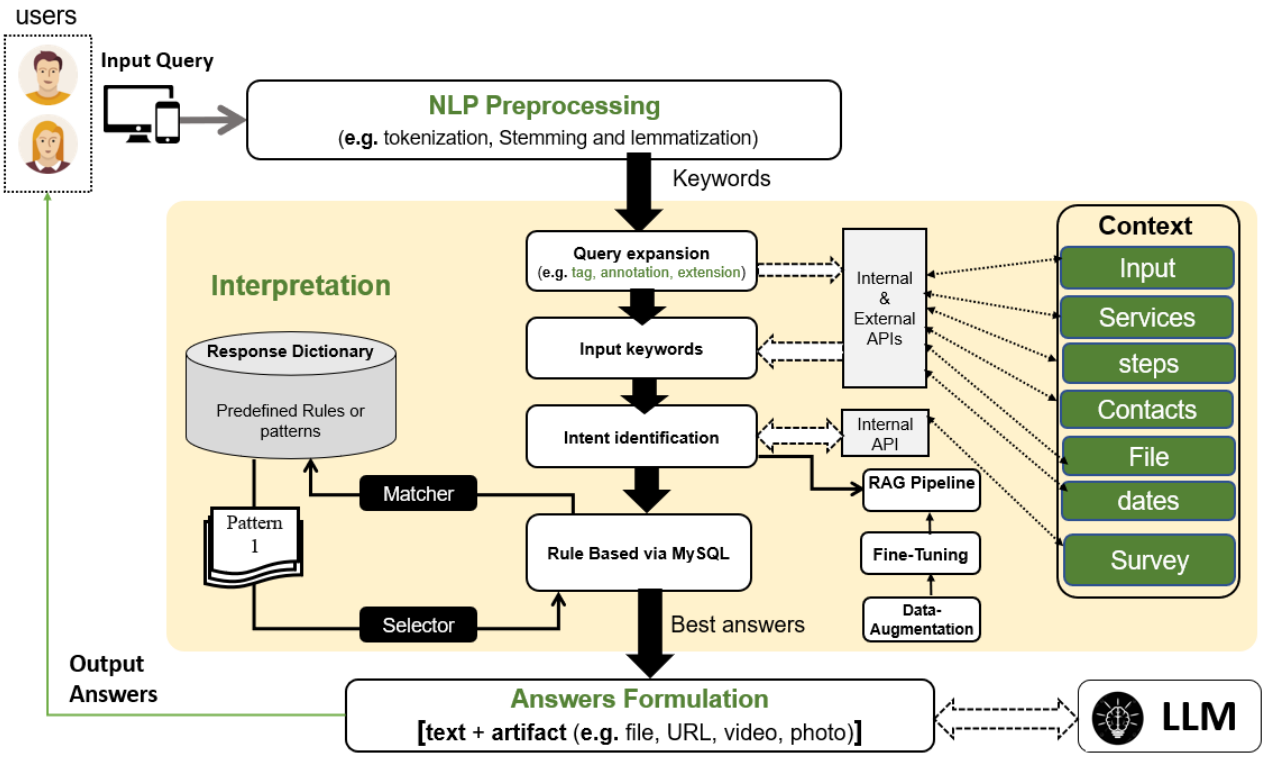


FIGURE 3.5 – Architecture proposée pour l'interprétation des requêtes utilisateurs dans le chatbot.

L'architecture fonctionnelle du chatbot intelligent repose sur une approche hybride, combinant des règles prédéfinies et des modèles d'intelligence artificielle avancés (LLM). La requête de l'utilisateur, saisie via une interface, subit un prétraitement NLP afin d'extraire les mots-clés essentiels et d'interpréter l'intention grâce à une analyse enrichie. Celle-ci peut être complétée par des API apportant des données contextuelles supplémentaires, comme illustré dans la figure 3.5 . La gestion du contexte tient compte de plusieurs éléments : services, démarches, documents requis, contacts, échéances et retours utilisateurs, pour personnaliser la réponse. Si une réponse adéquate existe dans la base de règles, elle est renvoyée. Sinon, un pipeline RAG est déclenché, combinant la recherche documentaire et la génération de réponse par un modèle IA adapté.

Enfin, la réponse est formatée (texte, documents, liens, etc.) avant d'être transmise à l'utilisateur. Cette architecture assure une assistance automatisée, fluide et adaptée à divers besoins administratifs.

3.5 Paramètres contextuels et intentions

Les intentions des utilisateurs peuvent être organisées en :

Classes d'intention et leurs catégories, sont illustrées dans les figures 3.6 et 3.7. Nous affirmons que toutes les intentions des citoyens appartiennent à ces catégories, ce qui facilite l'identification correcte du contexte.

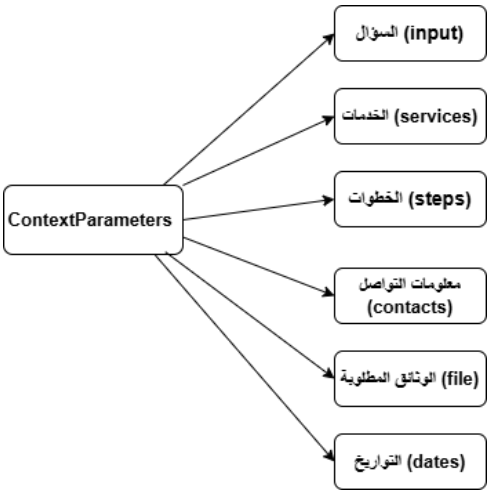


FIGURE 3.6 – Paramètres de contexte de demandes des utilisateurs.

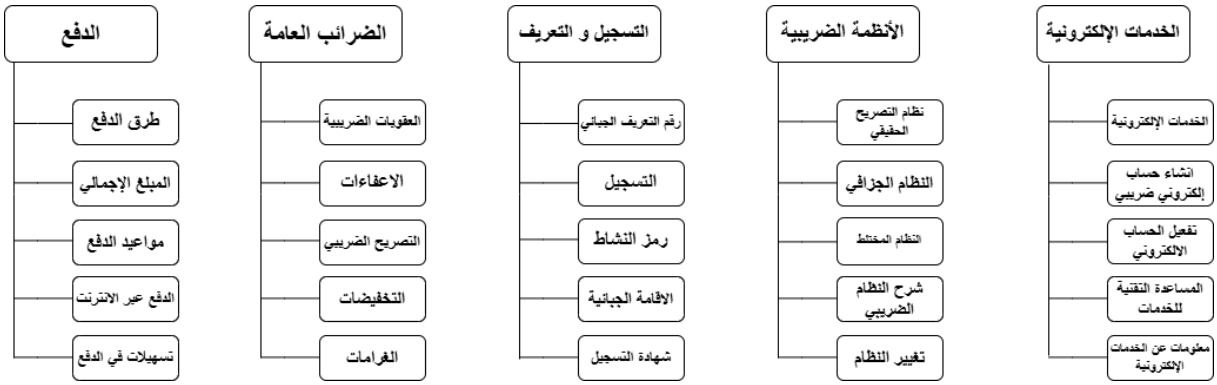


FIGURE 3.7 – classes d'intentions et leurs catégories

Dans ce cas, nous avons identifié les classes d'intention : paiement, impôts généraux, enregistrement et identification, activités économiques, systèmes fiscaux, services électroniques.

3.6 Formalisation mathématique du chatbot

Pour formaliser le fonctionnement de notre chatbot intelligent, nous proposons une approche mathématique qui modélise les différentes étapes de traitement des requêtes utilisateurs.

3.6.1 Modélisation de la requête utilisateur

Soit $q \in \mathcal{Q}$ une requête exprimée en langage naturel, où $\mathcal{Q}$ désigne l'ensemble des requêtes possibles. Le système commence par analyser q afin d'en identifier l'intention principale.

$$f : \mathcal{Q} \to \mathcal{I}$$

$$f(q) = i$$

où I est l'ensemble des intentions définies, et i est l'intention associée à la requête q.

3.6.2 Recherche d'information dans la base de données

Une fois l'intention identifiée, le chatbot accède à une base de connaissances relationnelle K (ex. MySQL) afin d'extraire les informations nécessaires à la génération de la réponse.

$$g : \mathcal{I} \times \mathcal{K} \to \mathcal{R}$$

$$g(i, \mathcal{K}) = r$$

où R est l'ensemble des réponses possibles, et r est la réponse extraite directement à partir des données structurées.

3.6.3 Génération via LLM (modèle de langage)

Si la base de données ne contient pas d'information pertinente, le système utilise un modèle de langage (LLM) pour produire une réponse contextuelle basée sur le contenu de la requête et le contexte c :

$$\hat{r} = \arg\max_{r \in \mathcal{R}} P(r \mid q, c)$$

où P est la probabilité estimée par le LLM que la réponse r soit la plus appropriée à la requête q.

3.6.4 Approche hybride

Notre approche combine deux méthodes complémentaires :

1/ Approche déterministe (basée sur des règles) :

$$\text{Si } (i, k) \Rightarrow r$$

2/ Approche probabiliste (basée sur l'IA) :

$$r = \text{LLM}(q, \mathcal{K})$$

Cette hybridation permet d'assurer des réponses à la fois précises (via les règles) et intelligentes (via le LLM), offrant ainsi une interaction fluide et pertinente avec l'utilisateur.

3.7 Organisation conceptuelle du système

la figure 3.8 montre l'élément central du système, le **modèle racine du chatbot**, est `ChatbotAgent`, qui est personnalisé à l'aide d'un ensemble de paramètres (*configuration*) définis par des propriétés `name` et `value`. Il traite une séquence de paires `question/response`, où chaque question correspond à une `Intent` détectée à partir d'une requête utilisateur (`InputQuery`), et chaque

intention est liée à une ou plusieurs réponses contenant des ressources structurées telles que du `text`, des `images` ou des `vidéos`.

La classe `Intent` représente l'objectif de l'utilisateur et est associée à la classe `Entity`, grâce à `MatchingIntent`, qui est responsable de la détection d'intention, soit par un appariement de mots-clés fondé sur des règles, soit par des vérifications de similarité basées sur l'intelligence artificielle. Une `Intent` capture ainsi le but de l'utilisateur et est associée à une ou plusieurs instances de la classe `Entity`, extraites à l'aide de différentes méthodes de `NLPProcessing`, telles que la reconnaissance de mots-clés, la tokenisation, la reconnaissance d'entités nommées ou la similarité sémantique basée sur des embeddings. L'association entre `Intent` et `Entity` est gérée par le composant `MatchingIntent`, qui peut utiliser soit une logique déterministe fondée sur des règles, soit des techniques d'intelligence artificielle exploitant la similarité vectorielle, des modèles de langage pré-entraînés ou des algorithmes de classification fine-tunés afin d'identifier avec précision le mapping pertinent dans un contexte donné.

Les entités, représentées par des instances de la classe `Entity`, sont enrichies à l'aide du `Context` courant (instance de la classe `Context`), ce qui permet au système d'identifier et de mapper avec précision les intentions les plus pertinentes. La classe `Context` généralise différents éléments liés à la situation du citoyen.

Pour améliorer le raisonnement et offrir des services personnalisés, le système intègre une **architecture de connaissance duale**. Il commence par consulter une base de règles manuelles, associées à des intentions et à des contextes précis. Si aucune règle ne correspond à la situation, il bascule automatiquement vers l'approche `AI-Based` à travers un pipeline dédié de récupération et de génération.

Au centre de ce pipeline se trouve le composant `Retriever`, chargé de rechercher les documents, réponses ou étapes de service pertinents à partir de la `KnewledgeBase`. Le `Retriever` applique une stratégie de récupération (`retrievalStrategy`) pour extraire les contenus les plus sémantiquement proches, tels que des tableaux (`dates, contact` , `contact details`, `required documents`,`service` , `steps`). Ces éléments récupérés sont ensuite transmis au `LLMEngine`, éventuellement connecté à un `FineTuneModel`, qui génère une réponse appropriée en tenant compte du `context` actuel et de l'`intent` détecté. Cette architecture permet une *génération augmentée par récupération* (`retrieval-augmented generation`), garantissant des réponses à la fois sensibles au contexte et fondées sur des faits fiables.

La classe `Survey` est utilisée pour collecter des entrées structurées supplémentaires auprès du citoyen. Elle contient un ensemble d'éléments `QuestionOrder`, chacun avec un `name`, une `option` et une `evaluation`. Ces questions peuvent être posées explicitement ou déduites automatiquement au fil de l'analyse continue du modèle de chatbot. Le chatbot stocke les valeurs des options sous forme `true`, `false` ou `undefined`, ce qui permet d'affiner la personnalisation et de recommander des services adaptés.

En alignant `MatchingInputQuery`, `MatchingIntent` et `MatchingContext`, le chatbot est capable de produire des réponses intelligentes, adaptatives et sensibles au contexte, permettant ainsi au citoyen d'accéder efficacement aux services administratifs et fiscaux.

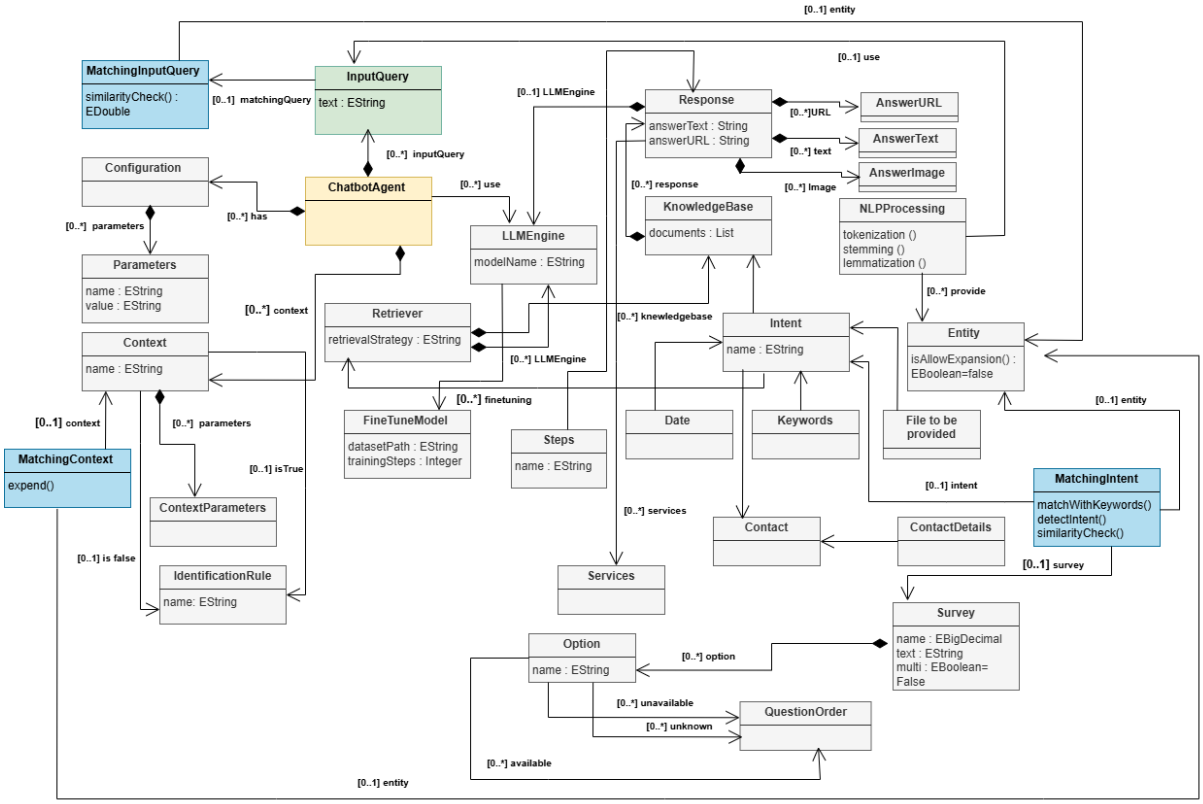


FIGURE 3.8 – Organisation conceptuelle du système de chatbot

3.8 Processus d'exécution et augmentation des données

Afin d'assurer une réponse optimale aux requêtes des utilisateurs, notre chatbot intelligent suit un processus d'exécution structuré, intégrant des techniques de prétraitement, d'analyse, de recherche d'information et de génération de réponse. Par ailleurs, l'utilisation de l'augmentation des données (*Data Augmentation*) permet d'améliorer la robustesse et la couverture du modèle, notamment pour des intentions peu représentées. La table 3.1 illustre ce processus global.

Étape	Composant	Description
1	Prétraitement (NLP Processing)	Tokenisation, lemmatisation, suppression des stopwords pour préparer la requête utilisateur.
2	Détection d'intention	Identification de l'intention via un classificateur ou des règles (intents).
3	Extraction d'entités	Récupération des mots-clés (ex : type de document, lieu, etc.) dans la requête.
4	Recherche dans la base de données	Si une réponse existe, elle est extraite à partir des tables relationnelles (MySQL).
5	Génération de réponse (LLM)	Si aucune réponse directe n'est trouvée, un modèle de langage est utilisé pour produire une réponse contextuelle.
6	Réponse enrichie	La réponse finale peut inclure des fichiers, liens ou images selon le contexte.

TABLE 3.1 – Étapes principales du processus d'exécution du chatbot

3.8.1 Rôle de l'augmentation des données (Data Augmentation)

Afin de renforcer la performance du système, notamment dans la reconnaissance d'intentions rares ou mal formulées, nous avons recours à des techniques de data augmentation :

- Paraphrasage automatique : génération de variantes syntaxiques de la même question.

- Traduction aller-retour (Back Translation) : traduction de la phrase vers une autre langue puis retour au français pour créer des versions alternatives.

- Synonymie contextuelle : substitution de mots-clés par leurs synonymes à l'aide de modèles NLP.

Ces techniques permettent d'enrichir le jeu d'entraînement, d'améliorer la généralisation du modèle, et de mieux couvrir les diverses formulations des usagers.

3.9 Entraînement du modèle et amélioration des performances

3.9.1 Importance de l'entrainement du modèle

Afin d'augmenter la précision du modèle et d'améliorer sa capacité à interagir de manière intelligente avec les utilisateurs, nous avons procédé à un *réglage fin* (fine-tuning) du modèle de base (LLM) en utilisant les données augmentées. Cette étape vise à spécialiser le modèle selon le domaine administratif algérien et les services publics associés.

3.9.2 Schéma du processus d'entraînement

La figure 3.9 présente un schéma de l'utilisation des données augmentées pour l'entraînement du modèle, afin d'enrichir la diversité des exemples et d'améliorer sa capacité de généralisation, en particulier pour les intentions rares.

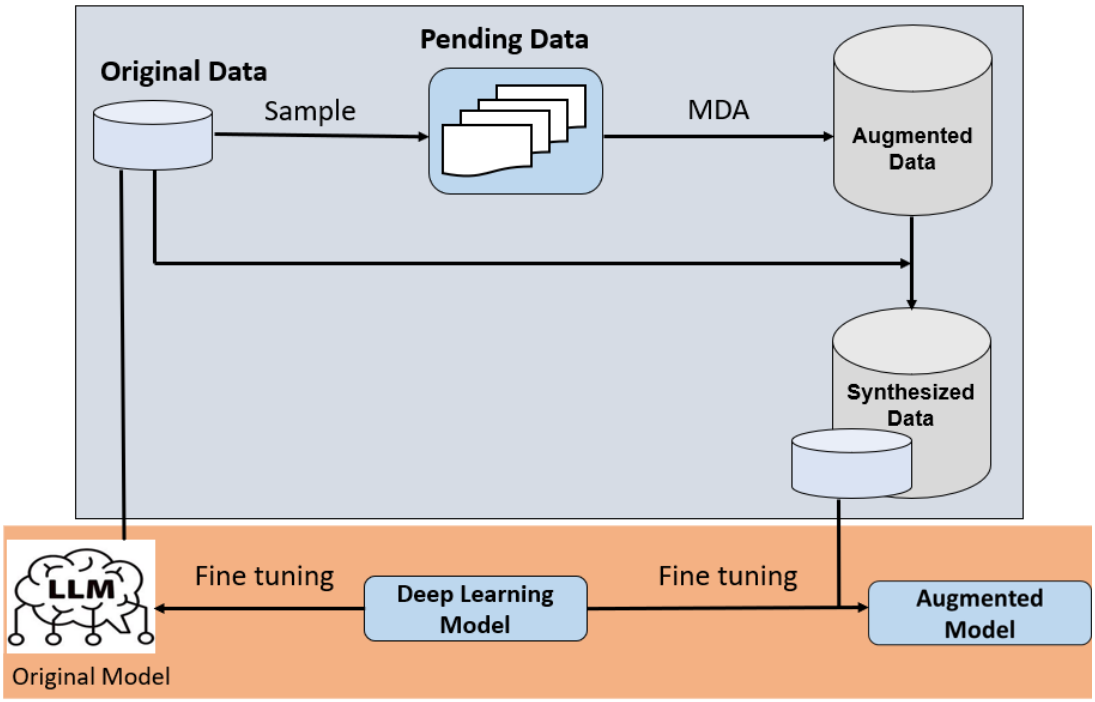


FIGURE 3.9 – Schéma illustrant l'utilisation des données augmentées pour l'entraînement du modèle.

Le schéma ci-dessus décrit comment les données originales sont prélevées et traitées à l'aide de techniques d'augmentation (telles que MDA). Les données augmentées (*Augmented Data*) et synthétisées (*Synthesized Data*) sont ensuite utilisées pour entraîner un modèle de deep learning. Ce processus permet d'affiner le modèle de langage de base (LLM) et de produire un modèle final amélioré, plus adapté aux formulations spécifiques des utilisateurs.

3.9.2.1 Contraintes techniques rencontrées

Même si cette étape est essentielle, nous avons rencontré de nombreux obstacles techniques, à savoir :

— **Insuffisance de moyens matériels :** la formation des modèles à grande échelle exige des GPU puissants et une mémoire conséquente, ce qui a restreint notre possibilité de réaliser plusieurs phases d'entraînement.

— **Complexité de l'environnement de développement :** la mise en place de l'environnement d'apprentissage (PyTorch, CUDA, Transformers, etc.) a nécessité du temps et une infrastructure spécifique.

— **Durée de traitement :** l'entraînement a pris du temps en raison du nombre important de données et des dimensions du modèle employé.

En dépit de ces restrictions, la formation nous a permis :

— D'améliorer notablement le niveau des réponses du chatbot.

— D'étendre le spectre des intentions détectées.

— Diminuer la quantité de réponses standardisées ou hors sujet.

3.10 Structure de la réponse

Afin de fournir des réponses précises, contextuelles et utiles aux utilisateurs, notre chatbot adopte une structure de réponse enrichie et bien organisée, comme illustré dans la figure 3.10.

FIGURE 3.10 – Structure organisée de la réponse générée par le chatbot.

3.10.1 Le format structuré retenu

Plusieurs raisons ont conduit à la sélection de ce format structuré :

— **Offrir une réponse centrale précise et sans détour** à la question soulevée.

— **Inclure des informations supplémentaires** sans nécessiter une nouvelle demande de l'utilisateur.

— **Accompagner l'utilisateur** dans les procédures administratives à entreprendre.

— **Proposer une expérience utilisateur enrichie** par des liens, des dates et des contacts pertinents.

La table 3.2 présente une description détaillée des composants de la réponse.

Composante	Description
`Réponse`	Le texte principal de la réponse fourni par le chatbot.
`Services`	Liens utiles vers les services administratifs ou informations de géolocalisation (ex. Google Maps).
`Étapes`	Étapes concrètes que l'utilisateur doit suivre pour compléter la procédure.
`Dossier`	Liste des documents requis pour déposer un dossier administratif.
`Contacts`	Coordonnées de contact (numéro de téléphone, email, adresse, etc.).
`Dates`	Dates ou délais importants liés à la démarche.

Table 3.2 – Description des composantes de la réponse.

3.10.2 Avantages de cette structure

— **Clarté de présentation :** l'information est présentée de manière claire et structurée, facilitant la compréhension par l'utilisateur.

— **Intégration multiplateforme :** la structure est facilement intégrable dans une interface mobile ou web, garantissant une expérience utilisateur homogène.

— **Scalabilité :** le format est extensible, permettant l'ajout futur de nouvelles composantes (comme des vidéos, des formulaires ou des services intelligents) sans avoir à modifier l'architecture initiale.

3.11 Architecture du modèle T5

Le modèle **T5 (Text-to-Text Transfer Transformer)** repose sur l'architecture Transformer classique, en adoptant une approche unifiée où toute tâche de NLP est reformulée comme une tâche de transformation de texte en texte. Comme illustré dans la figure 3.11 .

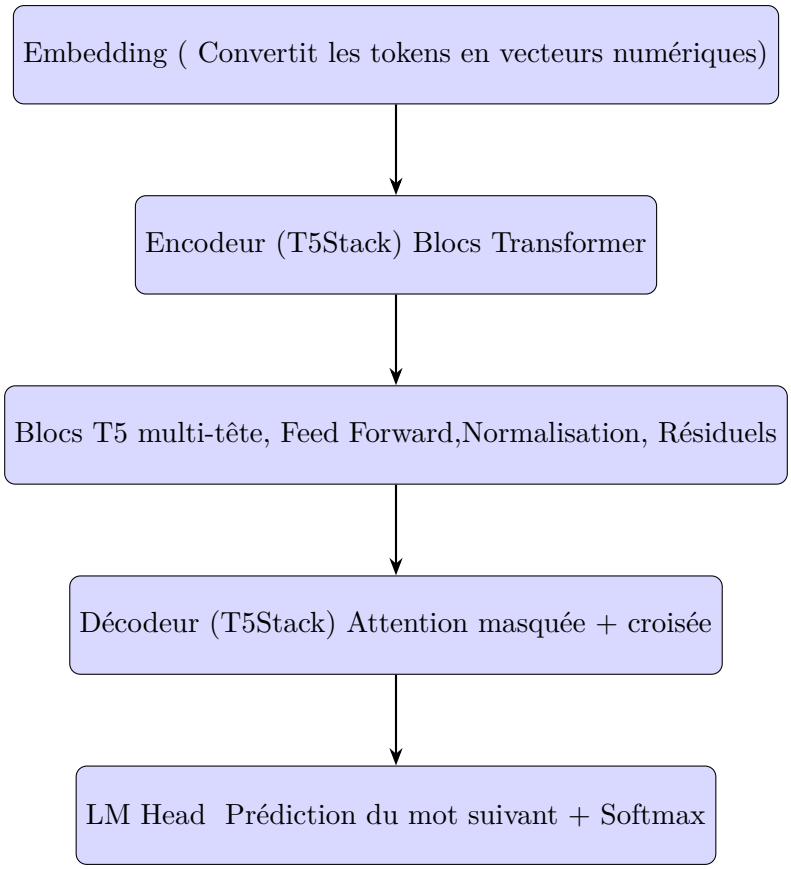


FIGURE 3.11 – Flux de travail de l'architecture du modèle T5

— **Tokeniseur** : utilise SentencePiece pour convertir le texte brut en jetons compréhensibles par le modèle. Chaque tâche est formulée sous forme de texte (entrée et sortie).
— **Couche d'embedding** : transforme chaque jeton en vecteur dense. Ces vecteurs sont utilisés comme entrées pour l'encodeur et le décodeur.
— **Blocs T5** : unités de base du modèle comprenant l'attention multi-tête, les couches feed-forward, la normalisation et les connexions résiduelles.
— **Encodeur (T5Stack)** : composé de plusieurs blocs Transformer. Il traite l'entrée et extrait des représentations contextuelles riches.
— **Décodeur (T5Stack)** : génère le texte de sortie en se basant sur les représentations de l'encodeur. Il intègre l'attention masquée et l'attention croisée.
— **Partage des poids** : les poids sont partagés entre l'encodeur et le décodeur, ce qui réduit la taille du modèle et améliore l'efficacité.
— **Tête de langage (LM Head)** : tête de prédiction qui applique une fonction softmax pour générer le mot suivant à partir des sorties du décodeur.
— **Objectif d'apprentissage** : utilise la technique de « corruption de span » où certaines parties du texte sont masquées, et le modèle doit les reconstruire.

3.12 Conclusion

La solution proposée dans ce travail repose sur la conception et l'implémentation d'un chatbot intelligent basé sur le modèle T5, capable de générer automatiquement des réponses précises à partir des données disponibles. Le système combine des approches classiques basées sur des règles et des techniques avancées de traitement du langage naturel (LLM), permettant une réponse efficace même aux questions absentes de la base de données.

De plus, les réponses sont organisées sous une structure riche et bien définie, incluant des champs tels que : les étapes à suivre, les documents requis, les dates importantes, les coordonnées, ainsi que des liens utiles vers les services concernés. Cette architecture améliore l'intégration future du chatbot dans des interfaces web ou mobile, et assure une gestion fluide et évolutive des contenus administratifs.

4

Tawjih : notre preuve de concept et outils

Table de sommaire

4.1 Introduction

Grâce aux avancées en intelligence artificielle et en traitement du langage naturel, Tawjih, un chatbot intelligent, a été développé pour offrir une assistance personnalisée et interactive aux usagers dans le domaine fiscal et administratif. Il vise à améliorer l'expérience des utilisateurs en fournissant des réponses précises, des étapes claires et un accompagnement adapté, répondant ainsi aux défis rencontrés dans les services publics traditionnels.

Pour assurer la fiabilité et l'efficacité de Tawjih, une phase de preuve de concept a été intégrée au processus de développement. Ce chapitre se concentre sur l'aspect pratique de la mise en oeuvre de notre assistant intelligent. Il présente en détail les différentes étapes qui ont permis de passer de la conception initiale à une solution fonctionnelle et opérationnelle.

Nous débuterons par une présentation des technologies, langages de programmation et bibliothèques utilisés. Puis, nous décrirons le processus de développement du modèle intelligent, comprenant la préparation de la base de données, l'entraînement du modèle, sa phase de test, ainsi que son intégration dans une interface utilisateur intuitive via une application mobile.

Enfin, nous expliquerons l'architecture de déploiement choisie, qui garantit un accès fluide et performant à l'assistant, tout en assurant son évolutivité et sa capacité à répondre efficacement aux besoins des utilisateurs.

4.2 Technologie utilisée

Dans cette section, nous présentons l'environnement de programmation utilisé pour développer notre système . Cela inclut le langage de programmation et une présentation générale des bibliothèques utilisées.

4.2.1 Langage de programmation et bibliothèques utilisées

Pour le développement du modèle de notre chatbot, nous avons opté pour le langage de programmation **Python** , reconnu pour sa richesse en bibliothèques dédiées à l'intelligence artificielle et au traitement du langage naturel. Les principales bibliothèques utilisées dans ce projet sont présentées dans la table 4.1.

Bibliothèque	**Rôle**
Flask	Création d'API web simples et rapides
Transformers	Intégration de modèles NLP pré-entraînés
Sentence Transformers	Représentation sémantique des phrases
Deep Translator	Traduction automatique multilingue
FAISS	Recherche vectorielle rapide
PyTorch	Apprentissage profond et NLP
NumPy	Calcul scientifique et tableaux
Pandas	Traitement des données tabulaires
Scikit-learn	Prétraitement et évaluation des modèles
JSON	Échange de données formatées
Modules pers.	Fonctions internes : nettoyage, formats, etc.

TABLE 4.1 – bibliothèques utilisées.

— **Json :** la bibliothèque JSON est un module intégré à Python qui simplifie l'utilisation des données JSON (JavaScript Object Notation). Elle offre des fonctionnalités pour encoder les structures de données Python au format JSON et décoder les chaînes JSON en objets Python [42, 45].

— **PyTorch** : bibliothèque open-source d'apprentissage profond développée par *Facebook AI Research (FAIR)*. Conçue pour Python, elle est largement utilisée en recherche et en industrie dans les domaines de la vision par ordinateur, du traitement automatique du langage naturel et de l'apprentissage par renforcement. PyTorch est particulièrement appréciée pour sa flexibilité et son mode de calcul dynamique, qui facilitent le prototypage rapide et l'expérimentation[37].

— **Numpy :**il s'agit d'une bibliothèque d'apprentissage automatique populaire qui prend en charge les matrices volumineuses et les données multidimensionnelles. Elle intègre des fonctions mathématiques facilitant les calculs. [21].

— **Transformers :** la bibliothèque Transformers, créée par Hugging Face, offre la possibilité d'exploiter des modèles de traitement du langage naturel (NLP) préalablement entraînés comme BERT, GPT, T5, etc. Elle prend en charge diverses tâches telles que la classification de texte, la traduction et la création de texte [55].

— **Deep Translator :** est une bibliothèque Python qui offre la possibilité de traduire du texte en se servant de divers fournisseurs tels que Google Translate, DeepL, Microsoft Translator, et d'autres. Elle offre une interface conviviale pour réaliser des traductions multilingues [47].

— **FAISS :** est un outil élaboré par Facebook AI Research, conçu pour réaliser des recherches rapides et performantes de vecteurs similaires dans d'importantes bases de données. Elle est conçue pour être performante tant sur le CPU que sur le GPU [26].

— **Pandas :** est une librairie libre pour le traitement et l'analyse de données à travers Python. Elle propose des structures de données adaptables telles que le DataFrame pour la manipulation des données sous forme de tableau [32].

— **Scikit-learn :** est une bibliothèque Python libre et open source dédiée à l'apprentissage automatique. Elle offre une vaste sélection d'algorithmes de classification, de régression et de regroupement via une API à la fois simple et uniforme [39].
— **Sentence Transformers :** est une librairie destinée à générer des représentations vectorielles denses de phrases (embeddings) qui saisissent leur sens sémantique. Elle facilite des comparaisons efficaces de phrases, ce qui est pratique pour des missions telles que la recherche sémantique, le regroupement ou la détection de similitudes textuelles [46].
— **Flask :** est un framework web minimaliste pour Python qui facilite la création d'applications web légères. Il est élaboré pour être facile à manipuler tout en offrant la flexibilité nécessaire pour développer des applications sophistiquées. Flask adhère à la philosophie du « microframework » : il n'impose pas une structure d'architecture particulière et offre au programmeur le choix de façonner son application comme bon lui semble [40].

4.3 Notre mise en oeuvre

Dans cette section, notre code source est organisé en trois modules principaux : (i) le modèle, (ii) l'apprentissage et (iii) les tests permettant de déterminer l'interface de fracture. Nous allons présenter les principales fonctionnalités et logiques implémentées dans chacun de ces modules.

4.3.1 Schéma de la base de données

La table 4.2 présente les Technologies de base de données utilisées.

Technologie	**Rôle**
MySQL	Stockage des données structurées
MySQL Connector (Python)	Communication entre l'application Python et la base de données

Table 4.2 – Technologies de base de données utilisées.

Notre base de données relationnelle est conçue pour stocker et organiser les différentes composantes nécessaires au bon fonctionnement de l'assistant intelligent. Elle contient les tables suivantes :

— **L'intention** : contient les intentions des utilisateurs.
— **Mots-clés** : mots-clés associés à chaque intention pour faciliter l'appariement.
— **Réponse** : stocke les réponses textuelles associées aux intentions.
— **Étapes** : étapes détaillées à suivre pour accomplir une démarche.
— **Services** : liens ou informations sur les services administratifs concernés.
— **Documents à fournir** : liste des documents à fournir pour chaque démarche.
— **Contacts** : identifiants des entités de contact pour chaque service.
— **Détails de contact** : informations de contact (numéro, email, adresse, etc.).

— **Dates** : dates et délais importants relatifs à chaque demande.

Comme illustré dans la figure 4.1

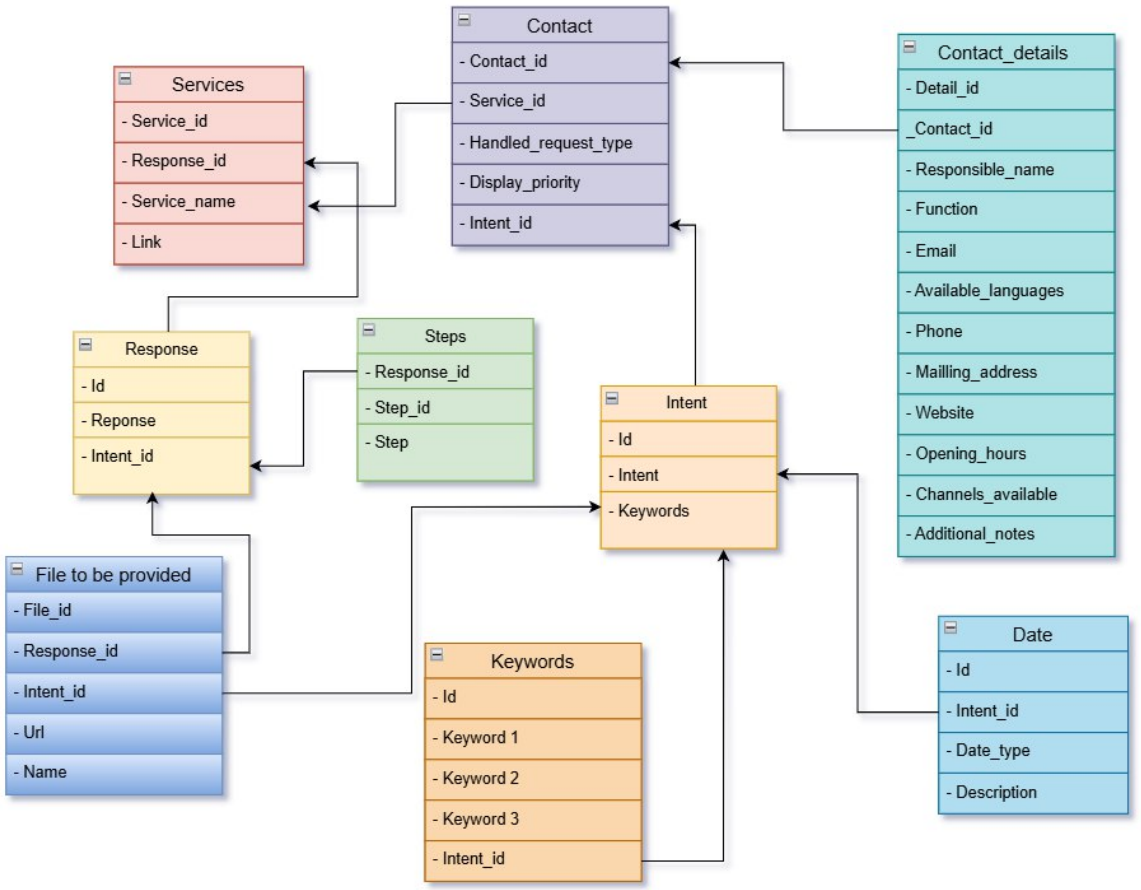


FIGURE 4.1 – Schéma de la base de données relationnelle de notre assistant.

4.3.2 Ensemble de données (dataset)

La base d'exemples, stockée en JSON, contient les champs suivants : réponse, services, étapes, dossier, contacts et dates. Elle sert aux modules de prétraitement et d'entraînement. la figure 4.2 illustré la structure d'un enregistrement.

```
"input": "هل يمكنني خصم تكاليف تجديدات المبنى التجاري؟",
"response": "نعم، يمكنك خصم تكاليف تجديدات المبنى التجاري إذا كانت تهدف إلى الحفاظ على المبنى أو تحسينه بشكل يتناسب مع استخدامه في العمل.",
"services": [
    "https://maps.google.com/?q=Direction+des+Impôts+Tiaret"
],
"steps": [
    "تقديم الفواتير الخاصة بالتجديد",
    "إدراجها في الاستمارة الضريبية المناسبة"
],
"dossier": [
    "فواتير التجديد",
    "مستندات الملكية"
],
"contacts": [
    "0214567890"
],
"dates": [
    "31/12/2024"
]
```


FIGURE 4.2 – Un extrait de code JSON correspondant à l'ensemble de données (dataset).

4.3.2.1 Augmentation des données

L'augmentation des données consiste à générer de nouveaux exemples de données pour l'entraînement d'un modèle.

L'augmentation des données est une technique utile pour fournir davantage d'informations à partir de moins de données[23].la figure 4.3 présente le flux de travail de traitement des données [49].

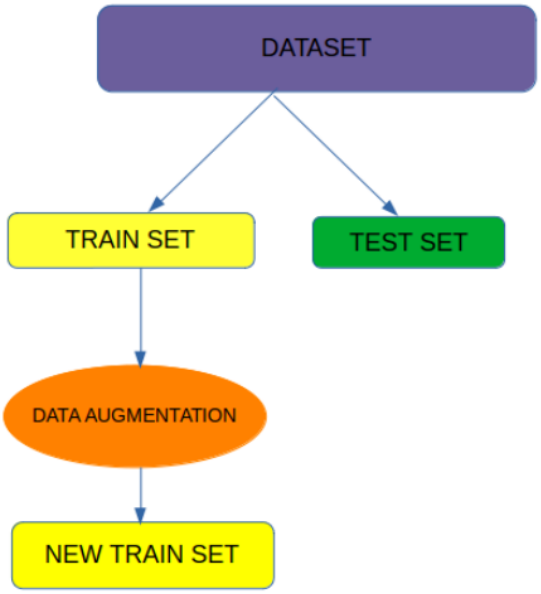


FIGURE 4.3 – Flux de travail de traitement des données.

Exemple 1. Augmentation de données Considérez la question suivante :

Question de base :

Q1 : هل يمكنني خصم مصاريف تجديد المبنى التجاري ؟

La technique utilisée est :

- Ajout d'exemples bruités (Noisy Data Generation)

- Q2- : تكاليف يمكنني اتلجاري؟ تجديدات هل المبنى خصم
- Q3- : تجديدتا المبنى خصم يكمنني تاكليف الجتاري؟ هل
- Q4- : التجاري؟ يمكنني تحسينات تكاليف خصم هل المنشأة
- Q5- : هل يمكنني خصم تكاليف تجديدات المنبى التجاري
- Q6- : هل يمكنني خصم تكاليف تجدديات المبنى التجاري
- Q7- : تجديدات المبنى هل خصم التجاري؟ تكاليف يمكنن
- Q8- : تكاليف تجيددات المبنى خصم التجاري؟ هل يمكننين
- Q9- : تكاليف تدجيدات خصم التجاري؟ المنشأة هل يمكنني

4.3.3 Comparaison expérimentale des modèles de langage

Nous avons évalué trois modèles de langage dans le cadre de notre étude : **LLaMA 3**, **Mistral 7B** et **T5**. Les résultats expérimentaux sont résumés comme suit :

— **LLaMA 3 :** a démontré d'excellentes capacités pour traiter des requêtes complexes et maintenir le contexte à long terme en anglais. Toutefois, ses performances en arabe se sont révélées inconstantes. De plus, ce modèle nécessite des ressources computationnelles importantes (notamment en termes de GPU et de mémoire vive), ce qui limite son usage dans des environnements en temps réel ou sur mobile.

— **Mistral 7B :** s'est distingué par une rapidité de réponse supérieure à celle de LLaMA 3, tout en offrant de bonnes performances en anglais. Néanmoins, il souffre du même défaut concernant le traitement de l'arabe, ce qui constitue un inconvénient majeur dans le contexte d'un chatbot destiné aux citoyens algériens.

— **T5 :** a surpassé les deux autres modèles en compréhension de la langue arabe. Il présente également une meilleure flexibilité pour l'ajustement fin (*fine-tuning*) et se montre plus adapté aux environnements à faibles ressources, ce qui en fait un candidat privilégié pour un déploiement local et réaliste.

4.3.4 Modèle de langage utilisé

Nous avons opté pour le modèle **T5 (Text-to-Text Transfer Transformer)** de Google. Ce modèle est particulièrement adapté aux tâches de génération de texte et de réponse automatique. Il a été choisi pour sa capacité à :

— Comprendre les requêtes formulées en langage naturel.

— Générer des réponses complètes, claires et bien structurées.

— Être facilement ajustable (fine-tuned) sur des jeux de données spécifiques.

La figure 4.4 représente l'architecture d'un chatbot intelligent basé sur le modèle de langage **T5**. Le processus commence par une phase de prétraitement où les données sont transformées en paires clé-valeur, avec un découpage et un encapsulage des textes. Ensuite, ces textes sont convertis en vecteurs grâce à la technique d'incorporation vectorielle (*embeddings*) afin de fournir une représentation compacte de l'information. Lorsqu'un utilisateur envoie une question, elle est convertie en vecteur puis comparée à la base de données indexée en utilisant la recherche de similarité, ce qui permet d'extraire la réponse la plus pertinente. Cette méthodologie garantit des réponses précises et cohérentes avec le contexte de l'utilisateur.

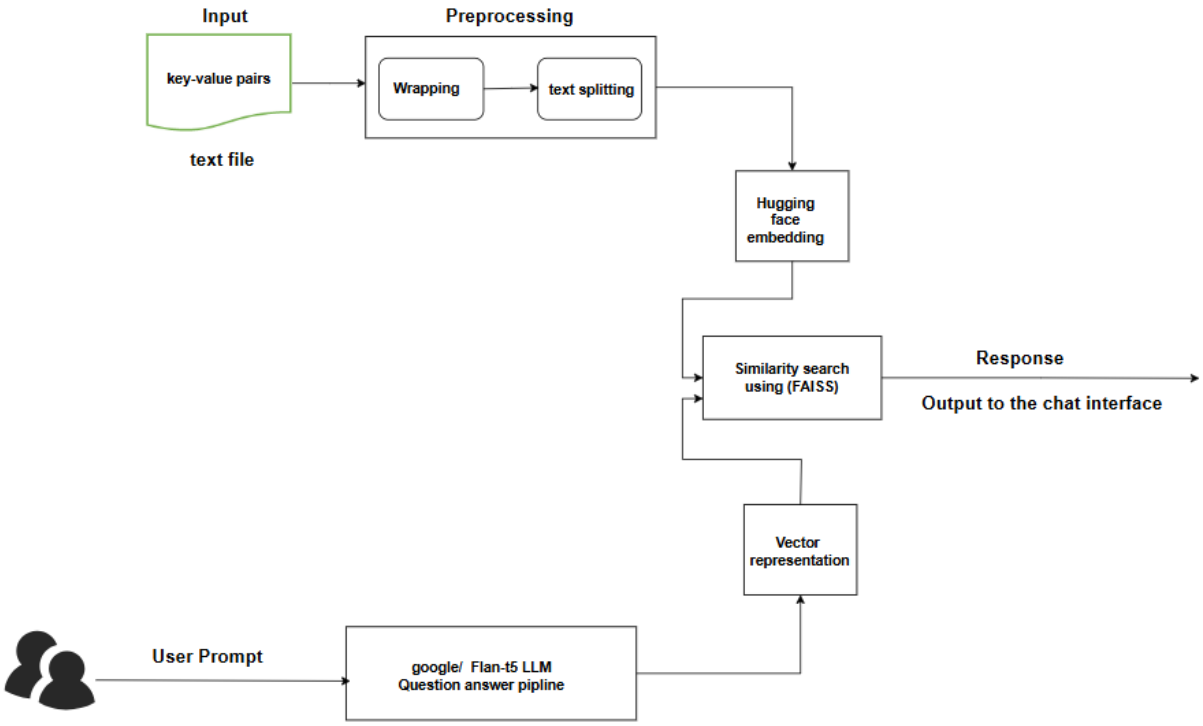


FIGURE 4.4 – Architecture pour la mise en œuvre d'un chatbot utilisant LLM.

4.3.5 Modèle Flan-T5 : code et intégration

La figure 4.5 présente le code de notre modèle d'IA :

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
tokenizer = AutoTokenizer.from_pretrained("google/flan-t5-base")
model = AutoModelForSeq2SeqLM.from_pretrained("google/flan-t5-base").to(device)
```

FIGURE 4.5 – Un extrait de code Python correspondant au modèle AI (Flan-T5-base) .

— **Préparation du modèle Flan-T5 :** ce code permet de préparer un modèle de génération de texte basé sur l'architecture `Flan-T5` pour qu'il soit utilisé dans une application comme un chatbot intelligent. Il effectue trois actions principales :

— **Importation des composants nécessaires :**

  - Le `tokenizer` sert à convertir le texte en une séquence de nombres que le modèle peut comprendre.
  - Le `modèle` (Flan-T5) est un réseau de neurones capable de comprendre un texte et de générer une réponse pertinente.
- **Chargement du modèle pré-entraîné :**
  - Le modèle `Flan-T5-base` est téléchargé depuis la plateforme Hugging Face.
  - Il a été déjà entraîné sur une grande quantité de données pour effectuer des tâches de type « question-réponse », « traduction », « résumé », etc.
- **Déploiement sur l'appareil disponible (CPU ou GPU) :**
  - Le modèle est transféré vers l'unité de calcul appropriée pour des performances optimales, notamment en cas d'utilisation sur GPU.

la figure 4.6 présente le code qui fait partie de l'implémentation du chatbot.

```
tokenizer = AutoTokenizer.from_pretrained("google/flan-t5-base")
model = AutoModelForSeq2SeqLM.from_pretrained("google/flan-t5-base").to(device)
q_embedding = embedder.encode([question], convert_to_numpy=True)
D, I = index.search(q_embedding, k=1)
context_en = GoogleTranslator(source='auto', target='en').translate(
    ↪ context_answer)
question_en = GoogleTranslator(source='auto', target='en').translate(question)
input_text = f"question: {question_en} context: {context_en}"
inputs = tokenizer(input_text, return_tensors="pt", padding=True, truncation=
    ↪ True).to(device)
output = model.generate(**inputs, max_length=100)
answer_en = tokenizer.decode(output[0], skip_special_tokens=True)
answer_ar = GoogleTranslator(source='en', target='ar').translate(answer_en).
    ↪ strip()
```

FIGURE 4.6 – Un extrait de code Python correspondant au code du chatbot.

ce bloc de code met en oeuvre un pipeline complet de génération de réponse pour un chatbot intelligent en s'appuyant sur un modèle de type Flan-T5.

- **Chargement des composants du modèle**
  Les deux premières lignes chargent le tokenizer et le modèle préentraîné Flan-T5-base à partir de la plateforme Hugging Face.
- **Vectorisation de la question**
  La question posée par l'utilisateur est transformée en un vecteur d'embedding grâce à un encodeur (comme SentenceTransformer).
- **Recherche dans la base de connaissances (index Faiss)**
  Le vecteur de la question est comparé aux vecteurs du contexte sauvegardé pour trouver la réponse la plus pertinente.
- **Traduction automatique (vers l'anglais)**
  La question et le contexte trouvé sont traduits automatiquement vers l'anglais.

— **Préparation du texte d'entrée**
Les deux parties (question + contexte) sont fusionnées dans une structure attendue par le modèle.

— **Tokenisation et mise en forme du batch**
Le texte est tokenisé, tronqué si trop long, puis converti en tenseurs pour le modèle.

— **Génération de la réponse par le modèle**
Le modèle génère une réponse en anglais à partir des tokens d'entrée.

— **Décodage de la réponse**
La réponse générée est convertie de tokens en texte brut.

— **Traduction de la réponse en arabe**
Enfin, la réponse anglaise est traduite automatiquement en arabe pour la rendre compréhensible par l'utilisateur.

4.3.6 Évaluation du chatbot basé sur des règles

La figure 4.7 illustré le graphique linéaire de l'évaluation des performances d'un chatbot fonctionnant selon une approche à base de règles.

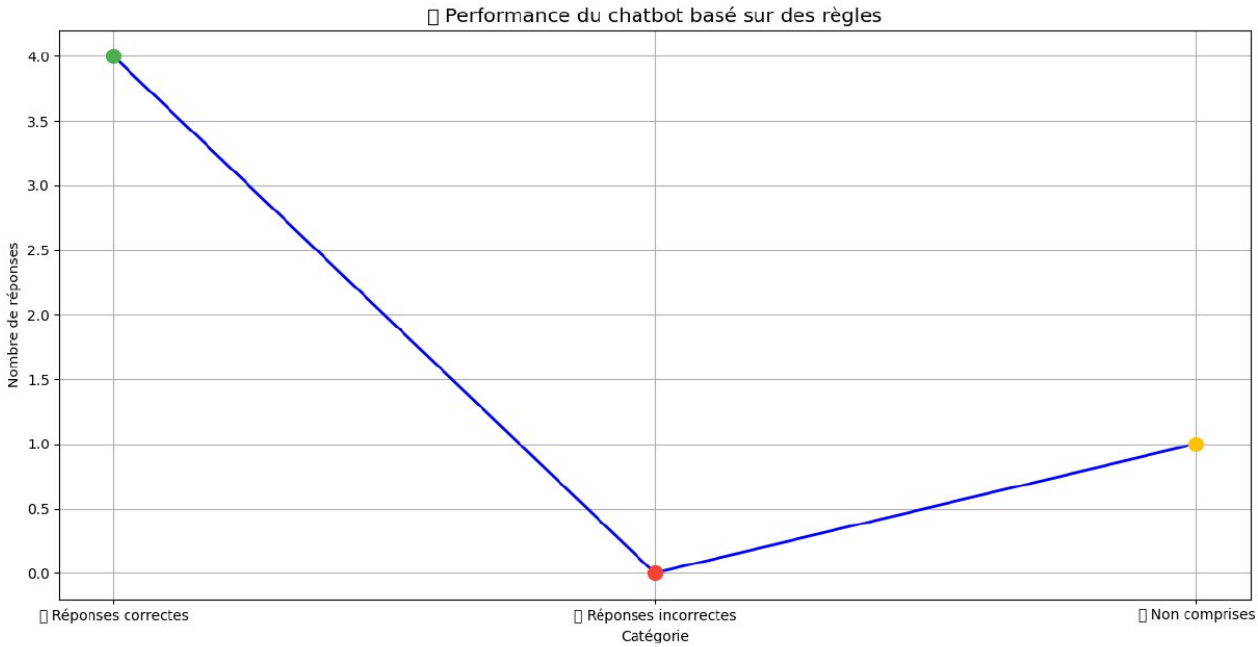


FIGURE 4.7 – Performance du chatbot basé sur des règles.

Les résultats sont répartis en trois catégories :

— **Réponses correctes** : le chatbot a fourni **4 réponses précises** correspondant aux attentes des utilisateurs.

— **Réponses incorrectes** : aucune réponse n'a été jugée incorrecte (**0 réponse**).

— **Réponses non comprises** : **1 requête** n'a pas été comprise ni traitée par le système.

Ces résultats montrent que le chatbot basé sur des règles est capable de fournir des réponses fiables lorsqu'une règle correspond bien à l'intention de l'utilisateur. L'absence de réponses incorrectes est un point positif, soulignant la rigueur du système. Toutefois, la présence d'une requête non comprise révèle une limite inhérente à ce type d'approche, notamment en ce qui concerne la flexibilité face à des formulations imprévues.

4.3.7 Résultats du chatbot basé sur un modèle LLM

la figure 4.8 illustré les performances du chatbot intelligent reposant sur un modèle de langage de grande taille (LLM), dans le traitement d'un ensemble de requêtes utilisateurs simulées.

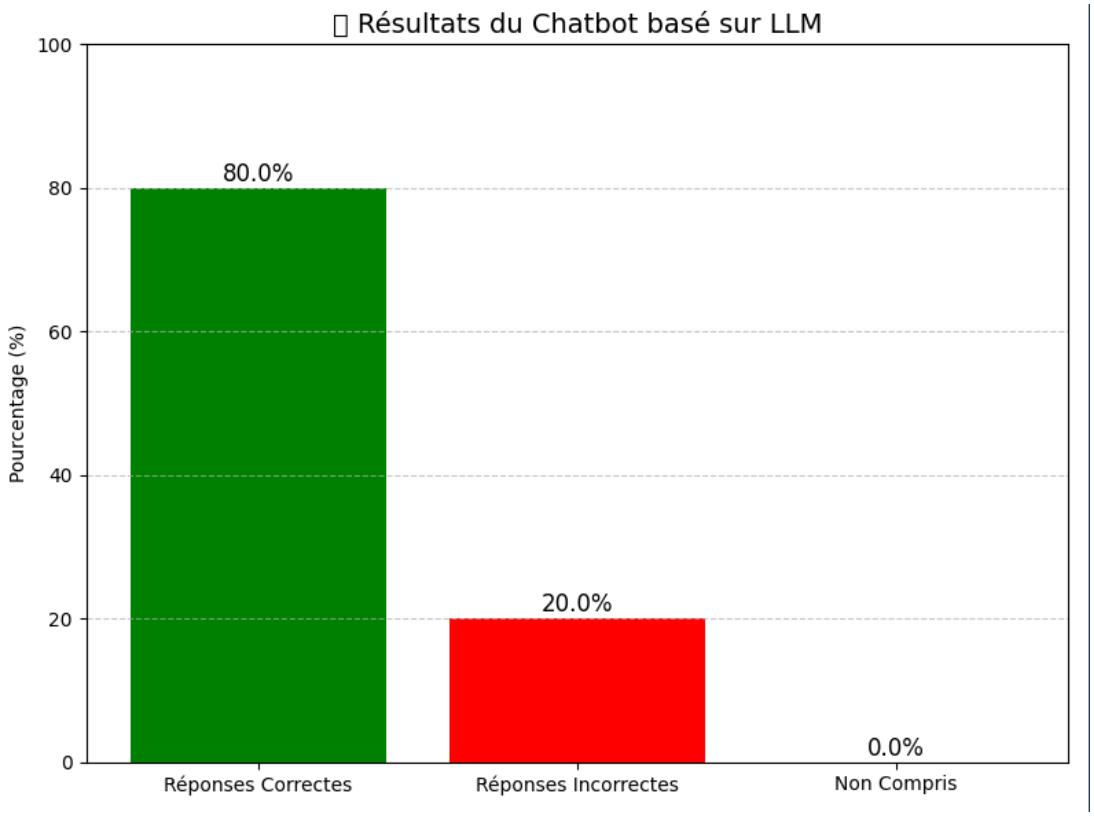


FIGURE 4.8 – Résultats du chatbot basé sur un LLM.

Les pourcentages de réponses générées par le chatbot sont répartis en trois catégories principales :

- **Réponses correctes** : 80% des requêtes ont reçu des réponses jugées correctes et pertinentes.
- **Réponses incorrectes** : 20% des réponses n'étaient pas adaptées ou contenaient des erreurs d'interprétation.
- **Requêtes non comprises** : Aucune requête n'a été laissée sans réponse ou non comprise (0%).

Le modèle LLM montre une forte capacité de compréhension et de génération de réponses pertinentes dans un contexte fiscal. Contrairement au chatbot à base de règles, ce modèle n'a laissé aucune requête sans réponse.

4.4 Processus d'entraînement du modèle

Le processus d'entraînement du modèle T5 se déroule en plusieurs étapes clés, comme illustré dans la figure 4.9 [16].

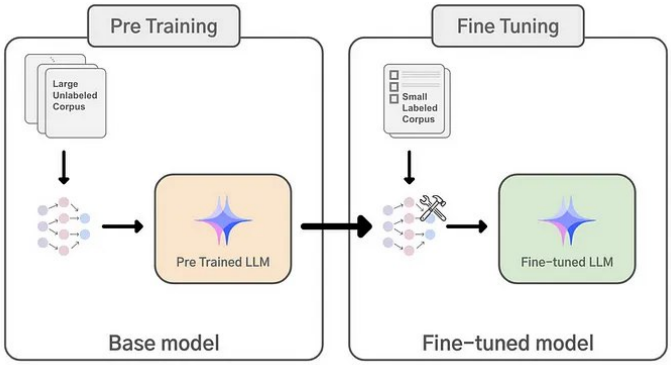


FIGURE 4.9 – Phases d'entraînement du modèle LLM : pre-training, fine-tuning .

Ce processus garantit Le développement d'un modèle de langage de grande taille (LLM) suit généralement deux grandes étapes : le **pré-entraînement** et le **adjustment fin (fine tuning)**.

4.4.1 Pré-entraînement

Lors de cette phase, les modèles sont exposés à d'immenses corpus textuels afin d'apprendre les structures linguistiques. Deux principales approches sont utilisées :

— Le **modèle de langage causal (CLM)** : comme GPT ou LLaMA, où le modèle prédit le mot suivant dans une séquence.

— Le **modèle de langage masqué (MLM)** : comme BERT, où le modèle apprend à prédire des mots masqués dans une phrase.

Cependant, les modèles pré-entraînés sont généralistes et nécessitent un ajustement pour des tâches spécifiques.

4.4.2 Adjustment fin (Fine-tuning)

Permet d'adapter le modèle à une tâche ou un domaine particulier en ajustant ses paramètres pour minimiser une fonction de perte spécifique. On distingue plusieurs types :

— **Instruction fine-tuning** : pour des modèles de type assistant conversationnel.

— **Adaptation au domaine** : pour des secteurs comme la médecine, le droit, etc.

— **Tâches spécifiques** : classification de texte, traduction, résumé, etc.

Récemment, des techniques comme **LoRA (Low-Rank Adaptation)** et **QLoRA** ont permis de rendre ce processus plus efficace, en gelant les poids du modèle et en n'entraînant que de petites matrices ajoutées. LoRA permet, par exemple, de réduire jusqu'à 10 000 fois le nombre de paramètres entraînables tout en maintenant des performances équivalentes au fine-tuning complet.

4.4.3 Exigences et outils de adjustment fin (fine-tuning)

Le fine-tuning de modèles comme LLaMA 7B ou Mistral 7B nécessite entre 150 à 195 Go de mémoire GPU, disponibles via des plateformes cloud comme AWS ou Google Colab. La qualité des données est cruciale : elles doivent suivre le format attendu par le modèle.

— **LLaMA/Mistral** : `[INST] instruction [/INST]réponse.`
— **OpenAI** : structure JSON avec les rôles `system`, `user`, `assistant`
— **T5** : `instruction: <prompt> answer: <response>`

Des outils *low-code* comme **Axolotl** facilitent grandement le fine-tuning, en offrant des configurations prêtes à l'emploi pour LoRA et QLoRA, même avec un minimum de connaissances techniques.

4.5 l'entraînement du modèle LLM T5

Notre projet est divisé en plusieurs fichiers Python, chacun jouant un rôle clair dans le processus d'entraînement et d'utilisation du modèle IA.

4.5.1 Paramètres globaux (config.py)

Définit les chemins, le nom du modèle, les paramètres d'entraînement. Comme illustré dans la figure 4.10

```
model_name = "google/flan-t5-base"
dataset_path = "augmented_dataset.json"
num_train_epochs = 4
batch_size = 4
```

FIGURE 4.10 – Extrait de configuration définissant le modèle T5.

4.5.2 Chargement des données (loader.py)

Charge les données depuis un fichier JSON et les convertit en dataset Hugging Face. Comme illustré dans la figure 4.11

```
with open(dataset_path, "r") as f:
    raw_data = json.load(f)
dataset = Dataset.from_list(raw_data)
```

FIGURE 4.11 – Chargement des données.

4.5.3 Prétraitement (preprocessing.py)

Prépare les données pour l'entraînement en les tokenisant, comme illustré dans la figure 4.12

```
input = tokenizer("        : " + ex["input"], ...)
target = tokenizer(json.dumps(ex["response"]), ...)
```

FIGURE 4.12 – Tokenisation des entrées et sorties pour le prétraitement.

4.5.4 Entraînement du modèle (train.py)

Entraîne le modèle T5 avec les données préparées. Comme illustré dans la figure 4.13

```
model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
trainer = Trainer(model=model, ...)
trainer.train()
```

FIGURE 4.13 – Entraînement du modèle T5.

4.5.5 Sauvegarde (save_model.py)

Sauvegarde le modèle et le tokenizer pour une utilisation future. Comme illustré dans la figure 4.14

```
model.save_pretrained(save_dir)
tokenizer.save_pretrained(save_dir)
```

FIGURE 4.14 – Sauvegarde du modèle.

4.5.6 Test simple (test_model.py)

Teste le modèle avec une question pour vérifier son comportement. Comme illustré dans la figure 4.15

```
inputs = tokenizer("        : ...", return_tensors="pt")
output = model.generate(**inputs)
print(tokenizer.decode(output[0]))
```

FIGURE 4.15 – Test du modèle avec une question d'exemple.

4.5.7 Exécution automatique (main.py)

Lance l'ensemble du pipeline depuis le chargement jusqu'au test. Comme illustré dans la figure 4.16

```
from loader import load_dataset
from preprocessing import preprocess
from train import train_model
from save_model import save
```

```
from test_model import test_model

def main():
    data = load_dataset()
    tokenized = data.map(preprocess)
    model = train_model(tokenized)
    save(model)
    test_model(model)

if __name__ == "__main__":
    main()
```

FIGURE 4.16 – Exécution automatique.

4.6 Évaluation du modèle

Pour évaluer la qualité des réponses générées par notre modèle, nous avons utilisé trois métriques :

- Accuracy : pour mesurer la correspondance globale entre les réponses attendues et générées.

- BLEU : pour évaluer la similarité entre le texte généré et la réponse de référence.

- Edit Distance : pour calculer le nombre de modifications nécessaires pour transformer une réponse générée en réponse correcte.

4.7 Test du modèle

4.7.1 Répartition des résultats du modèle LLM (T5)

Nous avons évalué les performances du modèle de langage T5 à travers une analyse de la répartition des réponses générées. La figure 4.17 présente cette répartition sous forme de diagramme circulaire.

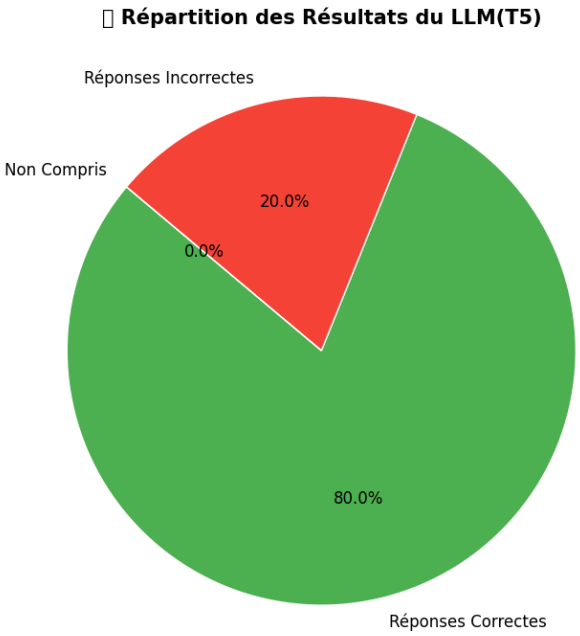


FIGURE 4.17 – Diagramme circulaire montrant la performance du modèle LLM (T5).

— **Réponses correctes (80%)** : la majorité des réponses fournies par le modèle sont exactes, ce qui indique une bonne performance générale.
— **Réponses incorrectes (20%)** : une partie des réponses contiennent des erreurs ou sont inexactes.
— **Non compris (0%)** : toutes les questions ont été comprises par le modèle, sans cas d'échec de compréhension.

Cela démontre que le modèle T5 possède une capacité de traitement du langage naturel efficace dans ce contexte expérimental.

4.7.2 Évaluation préliminaire du modèle avant l'entraînement

Nous avons évalué les performances initiales du chatbot avant toute phase d'entraînement. La figure 4.18 présente les performances du chatbot avant entraînement, évaluées à l'aide du taux de similarité (SequenceMatcher) et du BLEU score.

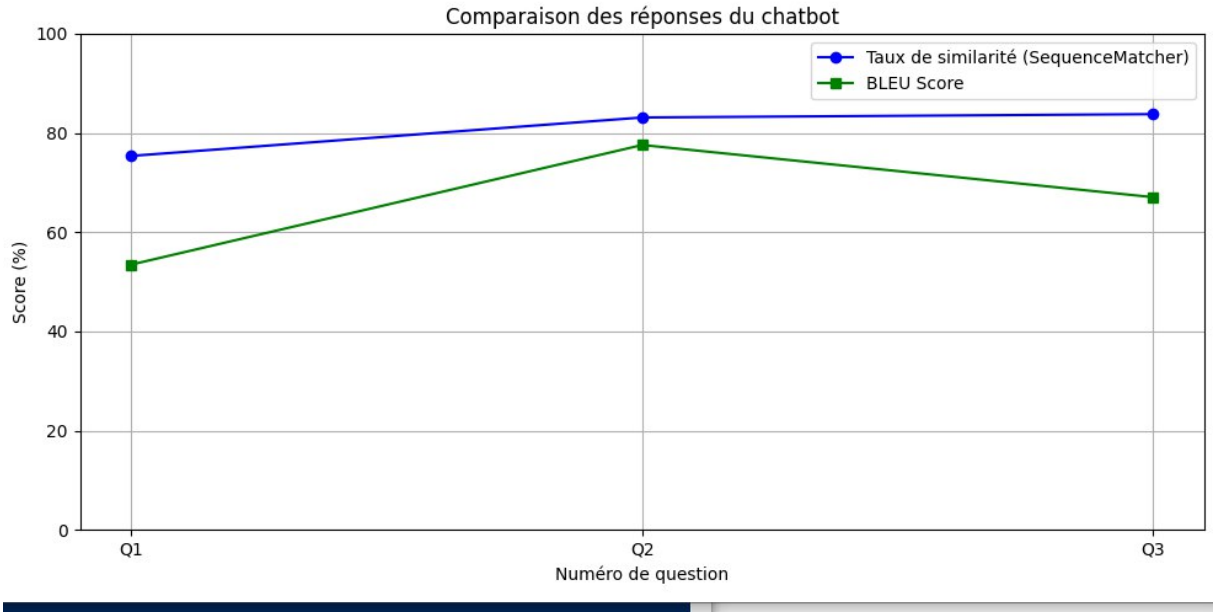


FIGURE 4.18 – Résultat de la réponse du chatbot avant l'entraînement.

Les résultats montrent que le taux de similarité est relativement stable autour de (75-80 %) , tandis que le BLEU score varie davantage selon les questions (de 55 à 75 %), indiquant une précision lexicale inégale. Cette figure sert de point de référence pour évaluer l'amélioration après entraînement.

Ce constat met en évidence que le chatbot, bien qu'ayant une certaine cohérence textuelle, nécessite un entraînement approfondi pour améliorer la qualité sémantique et lexicale de ses réponses. Cela justifie l'étape suivante de fine-tuning afin d'obtenir des réponses plus pertinentes et adaptées.

4.7.3 Évaluation des réponses de notre chatbot

La figure 4.19 présente un tableau affichant des réponses structurées, enrichies par des données contextuelles (date, contact, service...) et accompagnées d'une évaluation de leur exactitude.

.

Question	La réponse organisée	Évaluation
هل يمكنني خصم تكاليف تجديدات المبنى التجاري؟	نعم، يمكنك خصم تكاليف تجديدات المبنى التجاري إذا كانت تهدف إلى الحفاظ على المبنى أو تحسينه بشكل يتناسب مع استخدامه في العمل. **التاريخ**: 31/12/2024 **المصلحة**: Direction des Impôts Tiaret **الوثائق**: فواتير التجديد، مستندات الملكية **الاتصال**: 0214567890 **الخطوات**: تقديم الفواتير، إدراجها في الاستمارة الضريبية	صحيح
كيف أقدّم إقراري الضريبي عبر الإنترنت؟	يمكنك تقديم إقرارك الضريبي عبر الإنترنت باستخدام برنامج IRS Free File. **التاريخ**: 01/04/2025 **المصلحة**: Direction des Impôts Tiaret **الوثائق**: رقم الضمان الاجتماعي، مستندات الدخل **الاتصال**: 0211234567 **الخطوات**: الدخول إلى بوابة IRS، إنشاء حساب، ملء البيانات، إرسال الإقرار	صحيح
هل يجب دفع ضرائب على دخل الإيجار؟	نعم، يجب عليك دفع ضرائب على دخل الإيجار، ويجب الإبلاغ عنها في إقرارك الضريبي. **التاريخ**: 10/07/2025 **المصلحة**: Direction des Impôts Tiaret **الوثائق**: إيصالات دخل الإيجار، تفاصيل العقار **الاتصال**: 0256789012 **الخطوات**: حصر دخل الإيجار، حساب الضرائب، دفعها ضمن الإقرار	صحيح

FIGURE 4.19 – Des réponses structurées générées par notre assistant intelligent et leur évaluation.

Notre assistant génère des réponses structurées, pertinentes et claires pour les démarches fiscales. L'évaluation combinant méthodes automatiques et validations humaines confirme sa fiabilité, tout en soulignant quelques limites sur des cas complexes. Cette analyse valide notre approche hybride et oriente les améliorations futures.

4.7.4 Analyse de la précision par intention

Nous avons effectué un test pour évaluer la précision du modèle dans la classification des différentes intentions au sein du chatbot fiscal.

À partir de la figure 4.20 , nous constatons que la performance du modèle varie clairement selon l'intention.

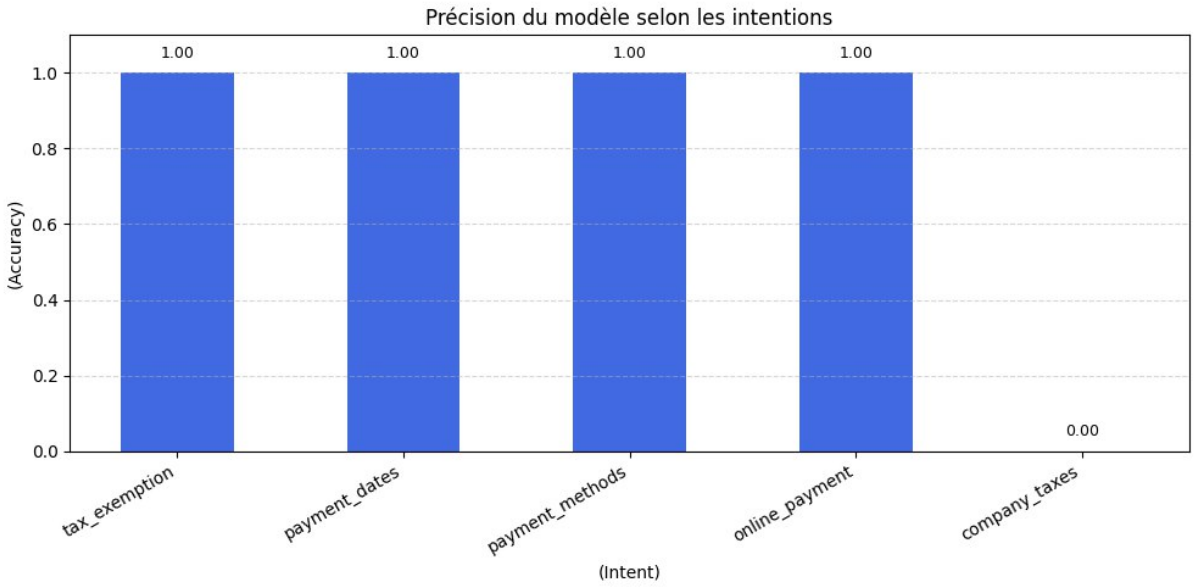


FIGURE 4.20 – Taux de précision par intention.

Pour les intentions :

- **Exonération fiscale** (tax_exemption).
- **Dates de paiement** (payment_dates).
- **Méthodes de paiement** (payment_methods).
- **Paiement en ligne** (online_payment).

Le modèle a atteint une **précision de 1,00**, ce qui montre que nos intentions sont **bien représentées** dans le jeu de données d'entraînement, avec des exemples **suffisamment nombreux, variés et bien formulés**.

En revanche, l'intention **impôts des entreprises** (company_taxes) affiche une **précision de 0,00**, ce qui signifie que le modèle **n'a jamais réussi à l'identifier correctement**.

Le modèle montre des performances inégales selon les intentions : il réussit à bien classifier celles qui sont bien représentées dans les données d'entraînement, mais échoue totalement sur les intentions sous-représentées.

4.7.5 Évaluation des performances en fonction des stratégies de réponse

Afin de comparer les performances des différentes stratégies de génération de réponse, nous avons mesuré les temps de réponse moyens (en secondes) pour quatre approches distinctes de chatbot. La figure 4.21 illustré clairement cette comparaison, en mettant en évidence les écarts de latence entre chaque stratégie.

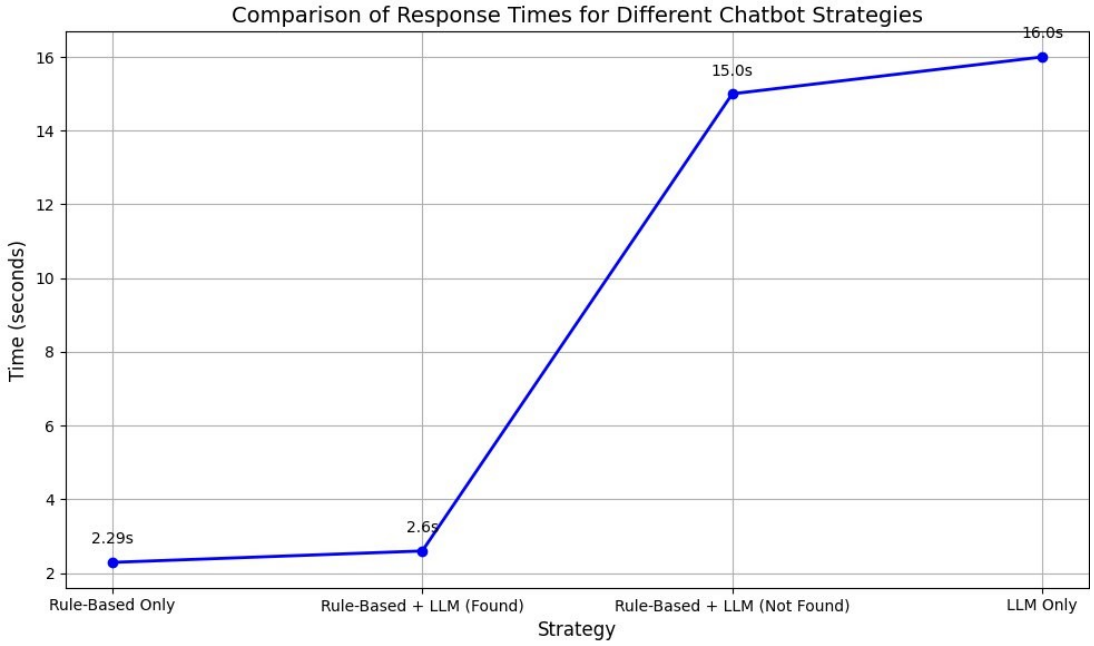


FIGURE 4.21 – Comparaison des temps de réponse des différentes stratégies de chatbot.

La table 4.3 présente les temps de réponse moyens obtenus pour chaque stratégie, mettant en évidence les différences de latence entre les approches évaluées.

Stratégie	**Temps de réponse (s)**
Basé uniquement sur des règles	2,29
Règles (réponse trouvée) + LLM	2,60
Règles (réponse non trouvée) + LLM	15,00
Basé uniquement sur un modèle de langage	16,00

TABLE 4.3 – Temps de réponse moyen selon les stratégies de chatbot.

Différentes stratégies d'interaction entre les règles et le modèle de langage ont été évaluées. Les résultats montrent des variations notables en termes de latence :

— **Basé uniquement sur des règles** : stratégie la plus rapide avec un temps de réponse moyen de **2,29 secondes**. Elle est adaptée aux requêtes simples et fréquentes bien couvertes par la base de règles.

— **Règles (réponse trouvée) + LLM** : le moteur à règles identifie une réponse, qui est ensuite validée ou enrichie par le modèle de langage. Cette validation légère augmente légèrement le temps de réponse à **2,60 secondes**.

— **Règles (réponse non trouvée) + LLM** : si aucune réponse n'est trouvée via les règles, le modèle de langage prend entièrement en charge la génération de la réponse. Cela entraîne une latence importante, avec un temps moyen de **15,00 secondes**.

— **Basé uniquement sur un modèle de langage** : toute la réponse est générée par le modèle de langage, ce qui en fait la stratégie la plus lente avec un temps de réponse moyen de **16,00 secondes**.

Les résultats montrent que plus on dépend du LLM, plus le temps de réponse augmente. Les systèmes à base de règles sont rapides mais limités en intelligence, tandis que les LLM sont plus intelligents mais plus lents. La meilleure solution est donc un système hybride, qui combine la rapidité des règles avec la puissance du LLM pour garantir performance et efficacité.

4.7.6 Analyse des performances selon le niveau des questions

Nous avons testé les performances de notre chatbot en fonction de trois niveaux de complexité des questions : facile, moyen et difficile. La Figure 4.22 illustré deux indicateurs clés de cette évaluation : le taux de couverture et la précision des réponses.

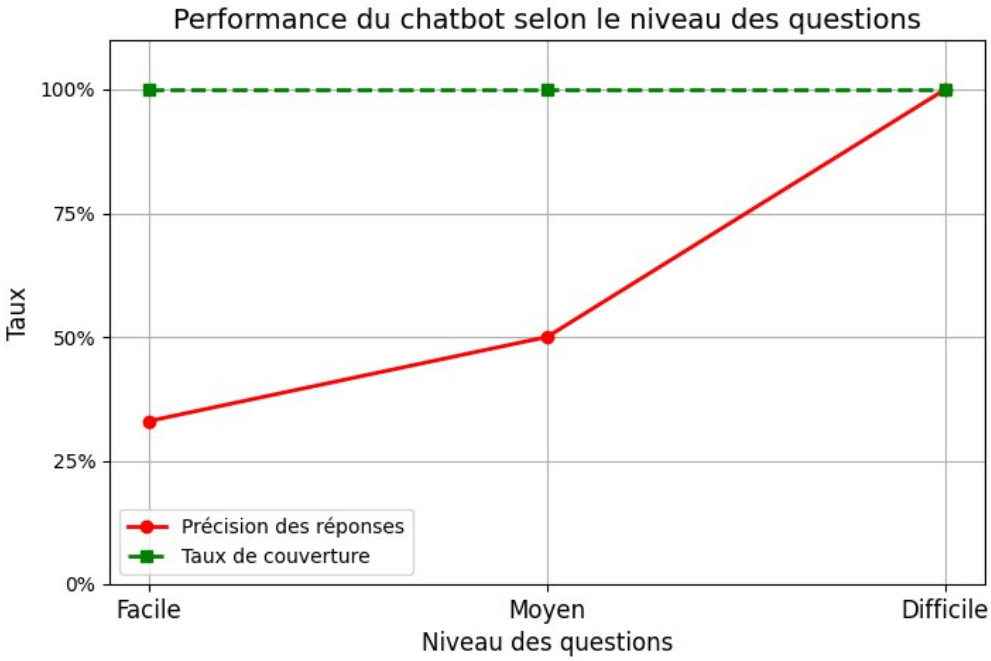


FIGURE 4.22 – Performance du chatbot selon le niveau de difficulté des questions.

Comme l'indique la figure, le taux de couverture reste constant à 100 % pour tous les niveaux, ce qui signifie que le chatbot est capable de fournir une réponse à toutes les questions posées, quel que soit leur niveau de difficulté.

En revanche, la précision des réponses varie en fonction du niveau de complexité. Elle est relativement faible pour les questions faciles (33 %), s'améliore pour les questions de niveau moyen (50 %), et atteint un taux optimal de (100 %) pour les questions difficiles.

Cette évolution peut s'expliquer par le fait que les questions difficiles sont généralement mieux formulées, plus structurées, et donc plus compréhensibles pour le modèle, ce qui améliore sa capacité à fournir une réponse correcte.

4.7.7 Progression de l'entraînement

Nous avons procédé à l'entraînement du modèle sur une période de 8 jours, répartie sur 4 époques (epochs). La figure 4.23 illustré la progression cumulative de l'entraînement au fil du temps.

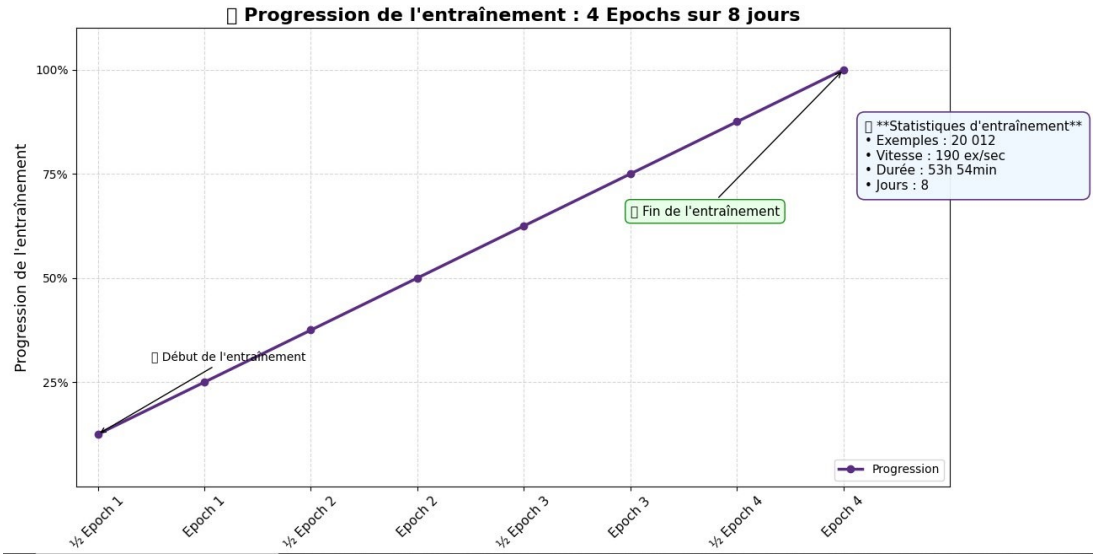


FIGURE 4.23 – Progression de l'entraînement : 4 époques sur 8 jours.

La courbe montre une montée régulière du taux de progression, passant de 12,5 % à 100 %. Cela reflète un déroulement fluide et maîtrisé des différentes étapes de l'entraînement, sans interruption ni ralentissement notable.

En complément, la figure contient un encadré synthétique présentant les statistiques générales de l'entraînement, à savoir :

Nombre total d'exemples traités : 20 012

Vitesse d'exécution : 190 exemples/seconde

Durée totale : 53 heures et 54 minutes

Nombre de jours : 8

Ces résultats témoignent d'un cadre d'exécution stable et optimisé, permettant d'obtenir une montée en compétence progressive du modèle, dans un délai maîtrisé.

4.8 UI/UX de notre outil d'assistant

Dans cette section, nous expliquerons le processus de conception UI/UX utilisé pour développer notre outil d'assistance.

4.8.1 Technologie utilisée

— **Conception UI/UX :** nous avons utilisé *Figma*, un outil de conception vectorielle basé sur le cloud. Accessible depuis n'importe quel navigateur, Figma permet de concevoir, de

prototyper et de collaborer en temps réel. Il est suffisamment intuitif pour être utilisé même par des non-spécialistes, facilitant ainsi le partage et l'exploitation des maquettes.

— **Développement mobile :** l'application a été développée avec *Flutter* au sein de l'environnement *Android Studio*.

Flutter : est un kit de développement d'interface utilisateur (UI) open source, créé par Google, qui permet de concevoir des applications multiplateformes (Android, iOS, Web, Linux, macOS, etc.) à partir d'un seul et même code source.

4.8.2 Processus de conception d'applications mobiles

La figure 4.24 représente le Processus de conception d'applications mobiles.

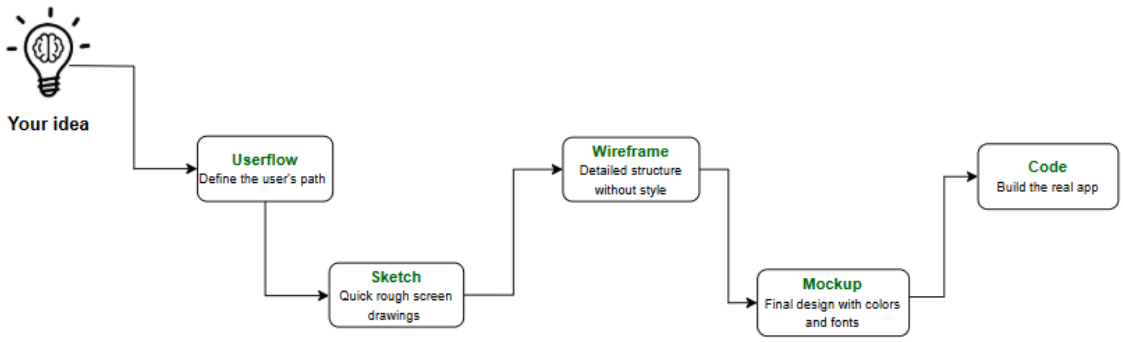


FIGURE 4.24 – Processus de conception d'applications mobiles.

4.8.2.1 Flux d'utilisateurs (User flow)

Le flux d'utilisateurs est un schéma qui décrit l'ensemble des étapes qu'un utilisateur suit pour interagir avec un produit numérique. Il permet aux concepteurs de visualiser le parcours complet de l'utilisateur afin de concevoir des interfaces fluides, intuitives et centrées sur ses besoins.

Nous avons modélisé le flux d'utilisateurs de notre application mobile pour illustrer les principales interactions et transitions entre les différentes vues. La figure 4.25 présente ce flux, depuis l'écran d'accueil jusqu'aux fonctionnalités avancées comme la messagerie, l'historique des conversations et les paramètres de l'application.

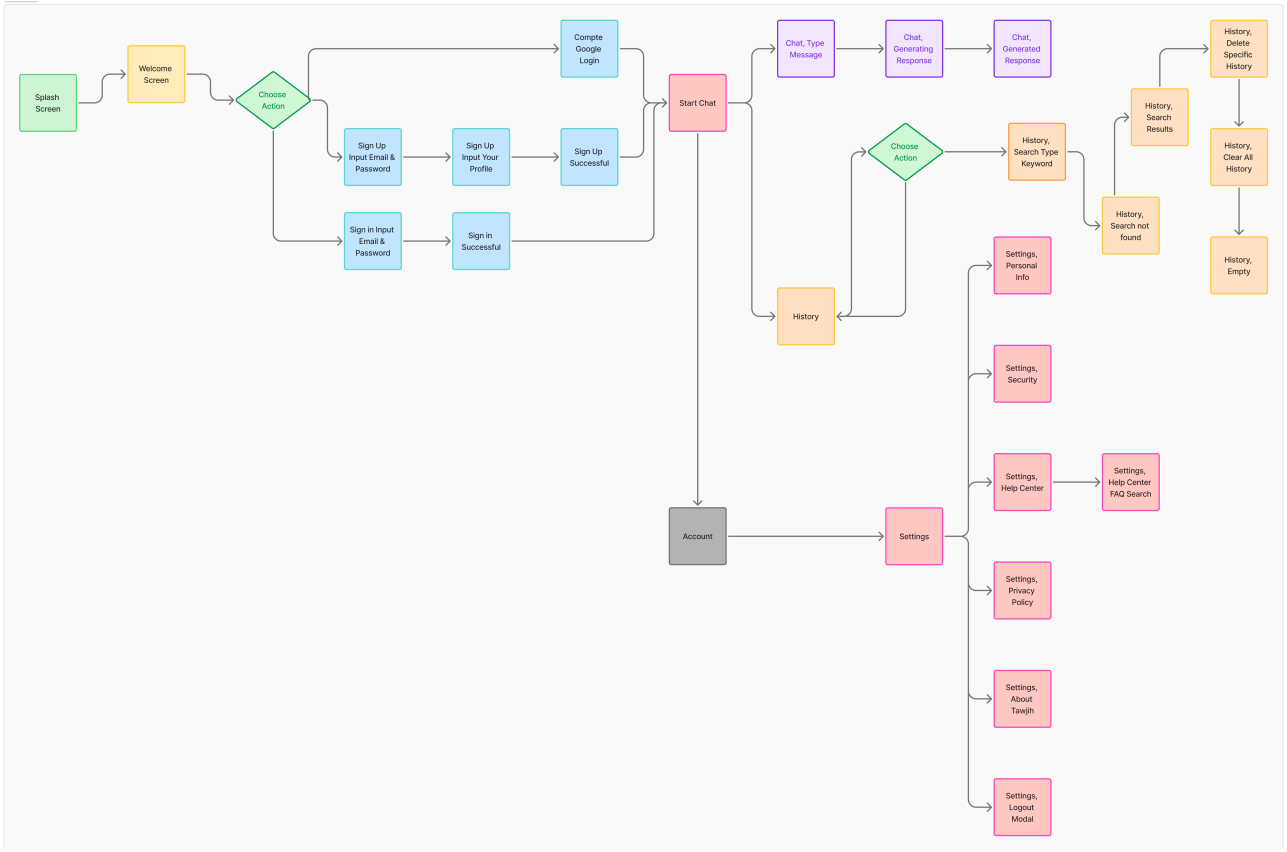


FIGURE 4.25 – flux d'utilisateurs de notre application (user flow).

4.8.2.2 Esquisse (Sketching)

Le terme « esquisse » dans la conception d'applications décrit la phase préliminaire de production d'esquisses brutes et de faible fidélité pour recueillir l'inspiration et les idées de mise en page de votre application mobile. La figure 4.26 illustré l'esquisse réalisée pour notre application mobile.

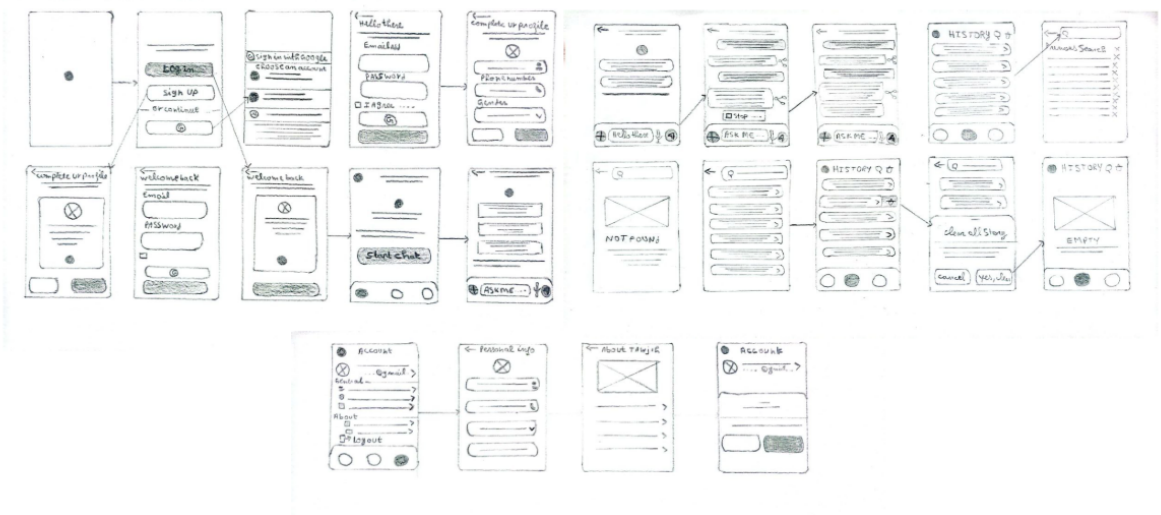


FIGURE 4.26 – Esquisse de notre application mobile (Sketching).

4.8.2.3 Maquette filaire (Wireframing)

Un Manquette filaire (wireframe) est une représentation visuelle simplifiée d'une interface utilisateur, utilisée pour planifier la structure, la disposition des éléments et la navigation d'un site web ou d'une application, sans se soucier de l'aspect graphique final.

Il sert à organiser les contenus et fonctionnalités de manière claire, afin de tester et valider l'ergonomie avant le design complet et le développement. La figure 4.27 illustré le maquette filaire réalisée pour notre application.

FIGURE 4.27 – Maquette filaire de notre application (Wireframing).

4.8.2.4 Scénarimages (Storyboard)

Un scénarimage (storyboard) est un plan visuel qui décrit la structure et les éléments clés d'une expérience planifiée.

Il utilise une séquence d'illustrations, de croquis ou d'écrans pour prévisualiser le déroulement des événements ou des interactions. Cette approche permet aux équipes de recueillir des retours précoces, d'apporter des améliorations et d'harmoniser les idées avant de passer à la phase finale de développement ou de production. La figure 4.28 illustré un exemple de scénarimage réalisé pour notre projet.

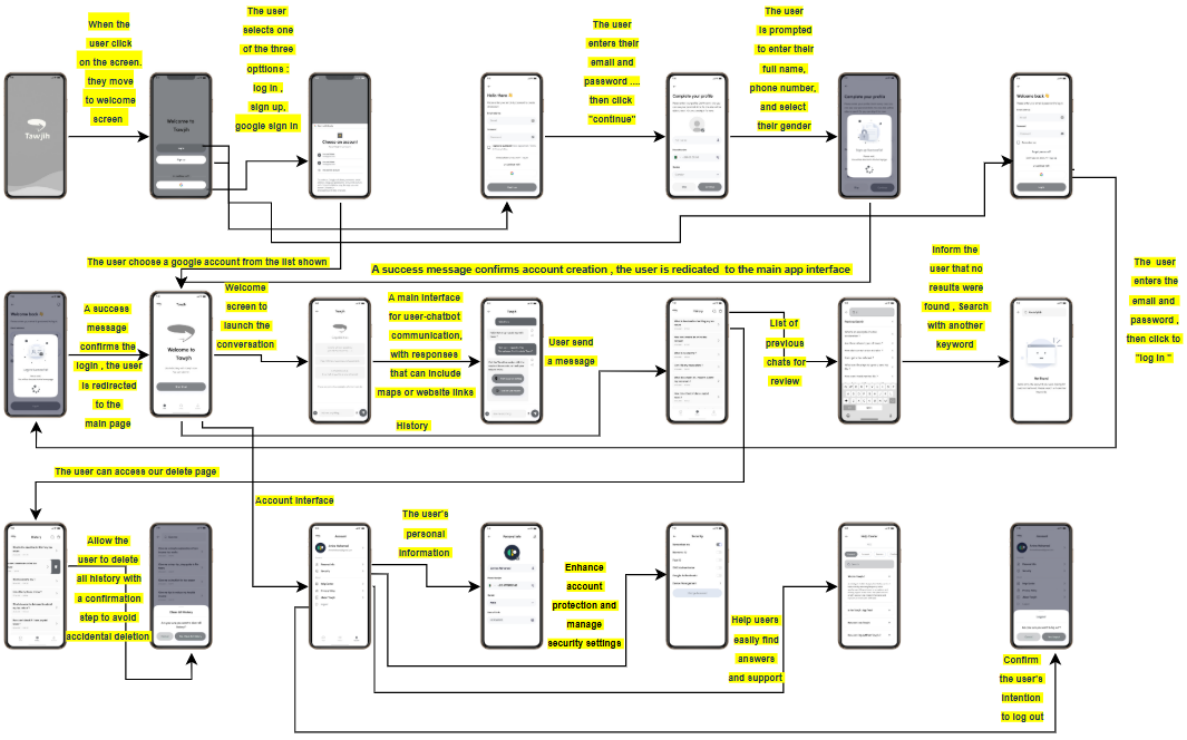


FIGURE 4.28 – scénarimages de notre application (Storyboards)

4.8.3 Aperçus des interfaces utilisateur

Dans cette section, nous allons définir les interfaces de notre application mobile.

— **Écran de démarrage (Splash Screen) :** il s'agit de l'écran d'accueil de l'application, qui présente uniquement le logo de *Tawjih*.

— **Écran de bienvenue (Welcome Screen) :** l'utilisateur accède à l'écran de bienvenue, où il peut choisir une action : se connecter ou créer un compte.

Comme illustré dans la figure 4.29.

FIGURE 4.29 – L'interface graphique principale de Tawjih et l'écran de bienvenue.

Inscription et connexion (Sign Up and Login) : Cette section permet à l'utilisateur de créer un compte ou de se connecter à un compte existant.

— **Compte Google Login :** Option alternative pour se connecter via un compte Google.

Comme illustré dans la figure 4.30.

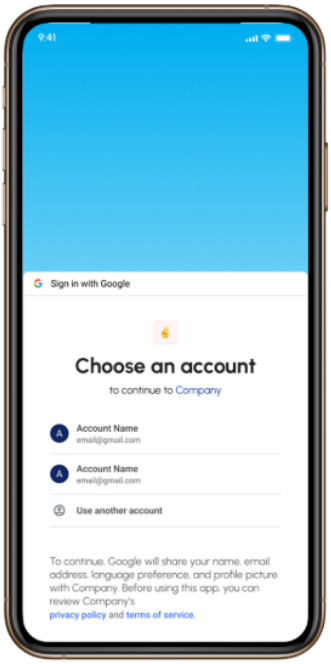


FIGURE 4.30 – Compte Google Login.

— **Inscription – Saisissez votre adresse e-mail et votre mot de passe (Sign Up –**

Input Email and Password) : L'utilisateur saisit son adresse e-mail et son mot de passe pour commencer l'inscription.

— **Inscrivez-vous – Saisissez votre profil (Sign Up – Input Your Profile) :** Il renseigne ses informations personnelles (nom, prénom, etc.).

— **Inscription réussie (Sign Up Successful) :** L'inscription est réussie, l'utilisateur est redirigé vers l'écran de chat.

Comme illustré dans la figure 4.31.

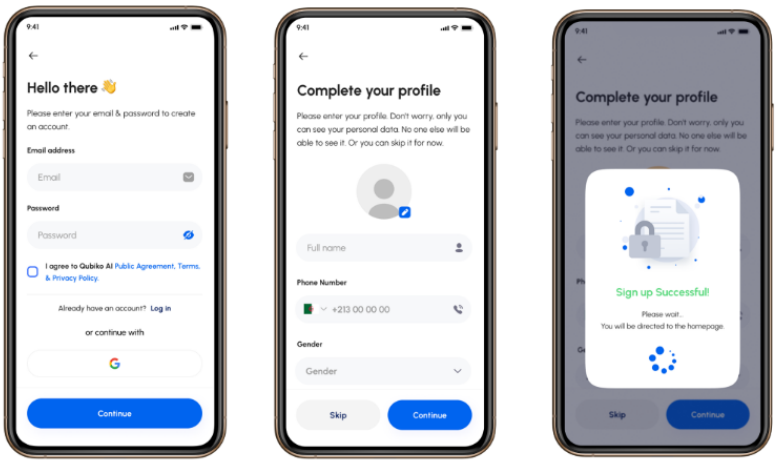


FIGURE 4.31 – Les étapes de l'inscription.

— **Connexion – Saisissez votre adresse e-mail et votre mot de passe (Sign In – Input Email and Password) :** L'utilisateur entre ses identifiants pour se connecter à un compte existant.

— **Connexion réussie (Sign In Successful) :** Une fois la connexion réussie, il accède à la page de chat.

Comme illustré dans la figure 4.32.

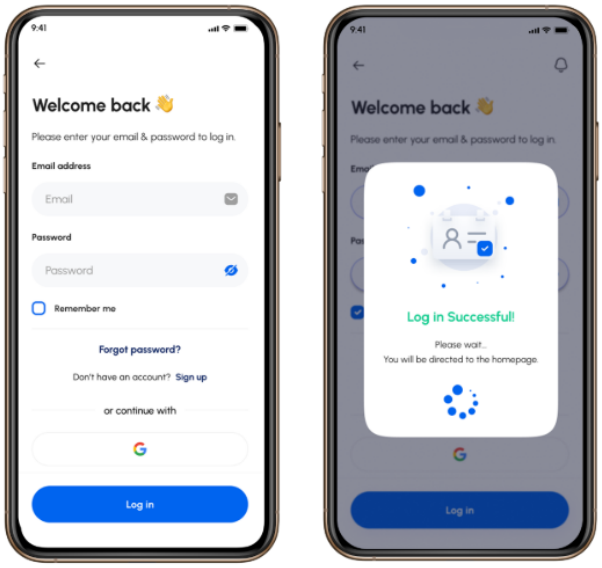


FIGURE 4.32 – Les étapes de connexion.

— **Démarrer le chat (Start Chat) :** Écran principal de messagerie. L'utilisateur peut commencer une nouvelle conversation avec le chatbot intelligent.

Comme illustré dans la figure 4.33.

FIGURE 4.33 – démarrage de chat.

Interaction par chat :

— **Saisir un message :** L'utilisateur écrit sa question ou sa demande dans le champ de discussion.

— **Génération de la réponse :** Le système traite la demande et prépare une réponse.
— **Réponse générée :** La réponse du chatbot s'affiche de manière structurée :
  — Une date liée à la procédure
  — Les étapes à suivre
  — Le dossier à fournir
  — Les contacts
  — Les services avec des liens (site web, Google Maps)

Comme illustré dans la figure 4.34.

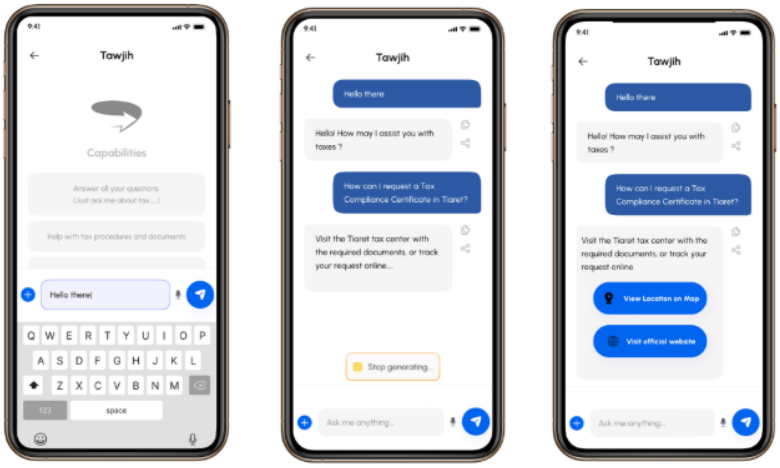


FIGURE 4.34 – Écrans d'interaction avec le chatbot.

Historique :

— **Écran Historique :** Accès à l'historique des conversations.
— **Type de recherche :** L'utilisateur peut rechercher une ancienne discussion à l'aide d'un mot-clé.
— **Résultats de recherche / Non trouvé :** Affichage des résultats ou d'un message si aucun résultat n'est trouvé.
— **Supprimer un historique spécifique :** Suppression d'une conversation précise.
— **Effacer tout l'historique :** Suppression de tout l'historique.
— **Vider :** État affiché si aucune discussion n'est enregistrée.

Comme illustré dans la figure 4.35.

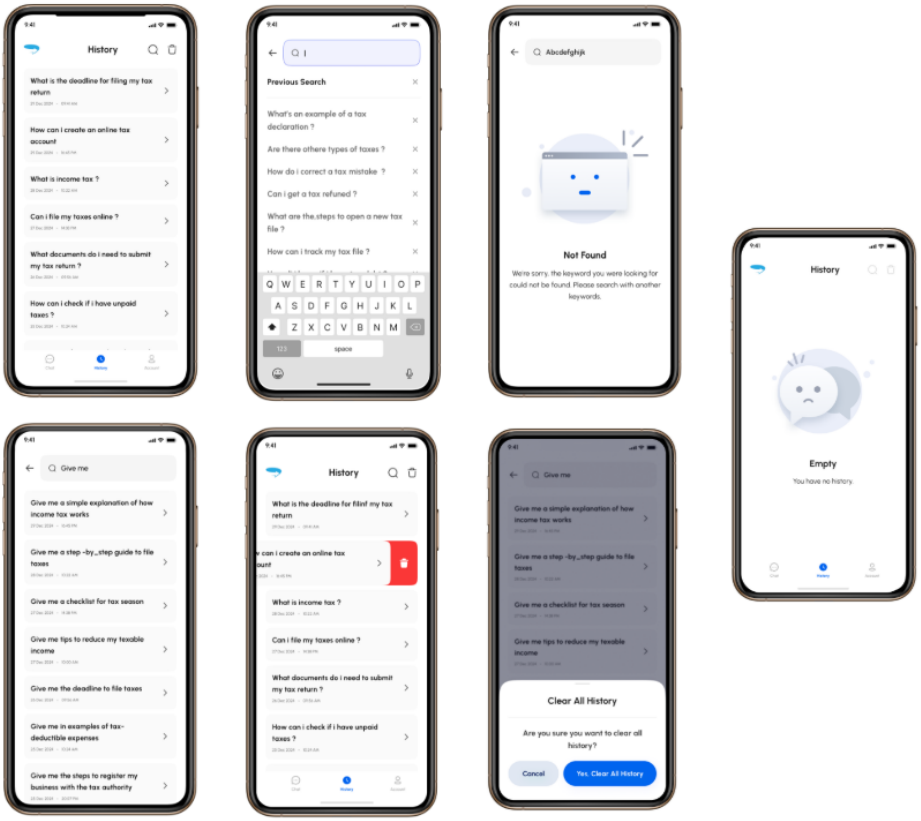


FIGURE 4.35 – l'historique des conversations dans l'application Tawjih.

— **Compte :** C'est l'écran où l'utilisateur accède aux paramètres et préférences de son compte. Il peut modifier les informations de profil et se déconnecter de l'application.

Comme illustré dans la figure 4.36.

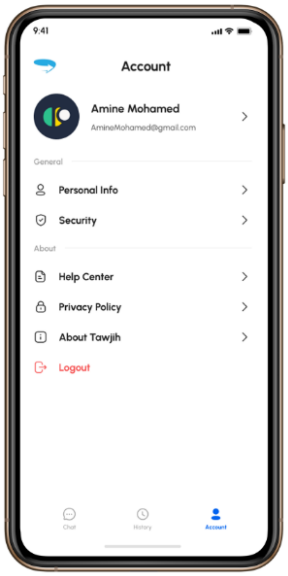


FIGURE 4.36 – Écran compte – Paramètres du compte utilisateur.

Paramètres

Accessible via le compte utilisateur. On y retrouve plusieurs sous-sections .

— **Informations personnelles :** permet la modification des informations personnelles de l'utilisateur (nom, prénom, etc.).
— **Sécurité :** offre la possibilité de mettre à jour le mot de passe ou d'effectuer une vérification de sécurité.
— **Centre d'aide :** accès aux ressources d'assistance pour résoudre les problèmes ou obtenir de l'aide.
— **Centre d'aide – FAQ Recherche :** permet de rechercher des réponses aux questions fréquentes (FAQ).
— **Politique de confidentialité :** consultation des règles et pratiques liées à la protection des données personnelles.
— **À propos de Tawjih :** affiche des informations générales sur l'application, ses objectifs et ses développeurs.
— **Modalité de déconnexion :** mécanisme de confirmation avant la déconnexion de l'utilisateur.

Comme illustré dans la figure 4.37.

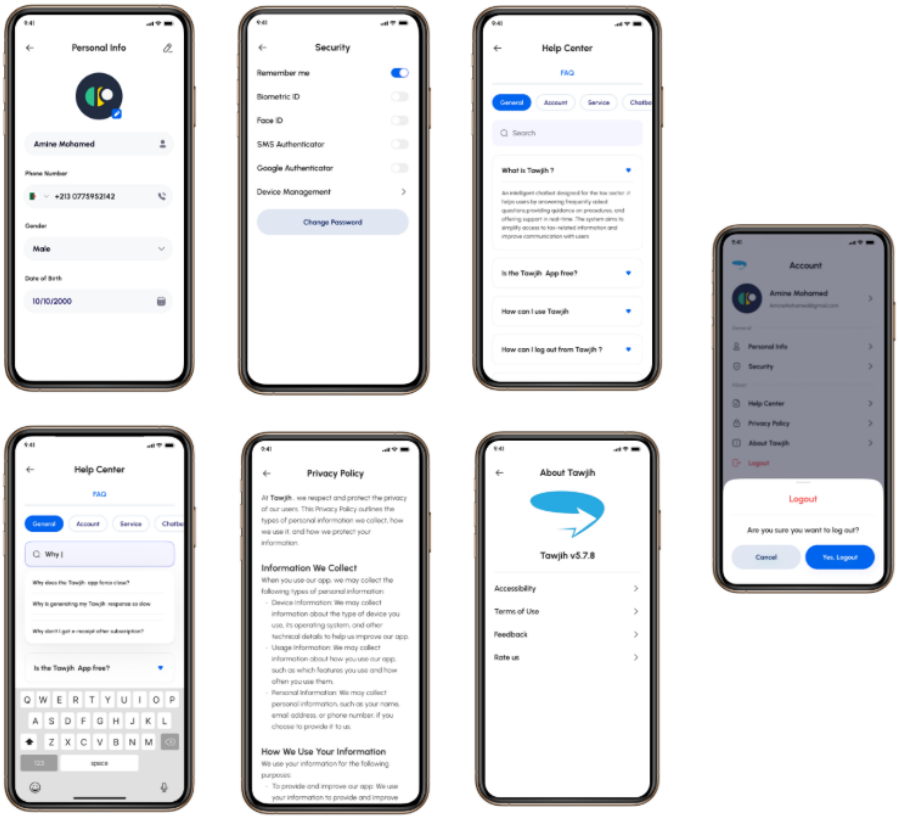


FIGURE 4.37 – Écrans des paramètres – Section paramètres de l'application.

4.8.4 Évaluation de l'expérience utilisateur (UX)

Nous avons évalué l'expérience utilisateur (UX) de notre interface à l'aide de cinq questions spécifiques, chacune notée sur une échelle de 1 (pas du tout d'accord) à 5 (tout à fait d'accord) :

— **Q1** : L'interface était facile à comprendre.
— **Q2** : Je me suis senti à l'aise lors de l'utilisation.
— **Q3** : Le design est visuellement agréable.
— **Q4** : J'ai pu effectuer les tâches prévues sans difficulté.
— **Q5** : L'expérience globale était satisfaisante.

La figure 4.38 illustré la moyenne des évaluations obtenues pour chaque question.

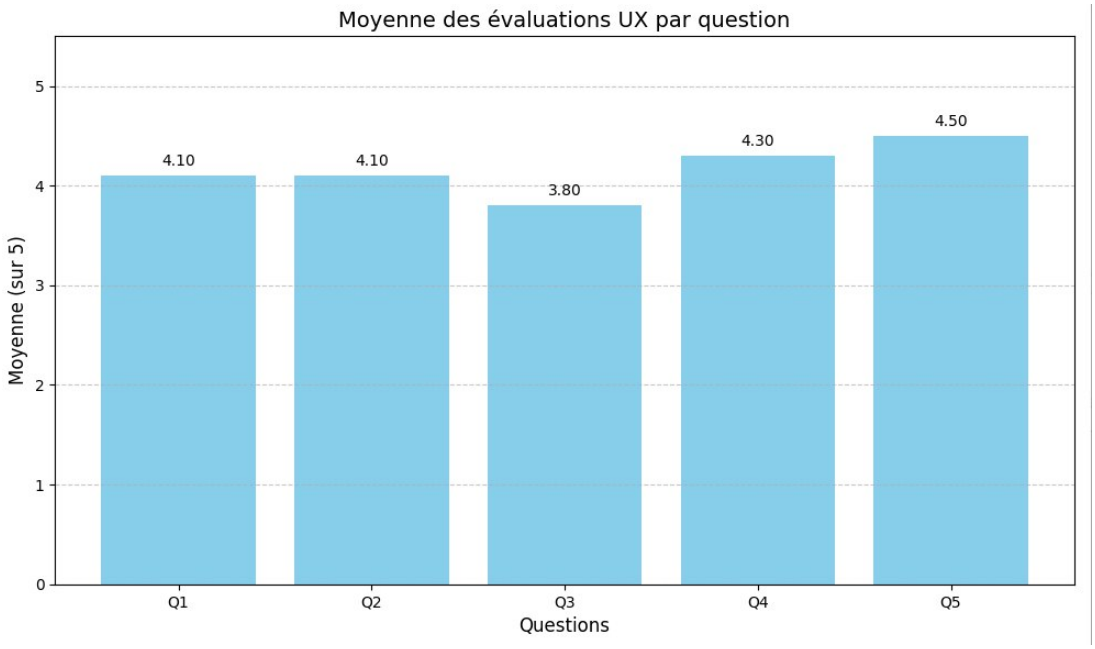


FIGURE 4.38 – Scores moyens des questions d'évaluation de l'expérience utilisateur (UX).

On observe que les résultats sont globalement positifs, avec une satisfaction élevée concernant l'expérience globale (**Q5 = 4,50**) et la facilité d'exécution des tâches (**Q4 = 4,30**). L'interface a également été jugée compréhensible (**Q1 = 4,10**) et confortable à utiliser (**Q2 = 4,10**), ce qui témoigne d'une bonne ergonomie générale.

Seule la question relative au design visuel (**Q3 = 3,80**) a obtenu une note légèrement inférieure, suggérant une marge d'amélioration possible au niveau de l'aspect esthétique de l'interface.

Ces résultats indiquent que l'interface est perçue comme efficace, intuitive et globalement satisfaisante. Un effort pourrait être envisagé pour améliorer l'aspect visuel afin d'atteindre un meilleur équilibre entre fonctionnalité et attractivité.

4.9 Diagramme de packages

Le diagramme ci-dessous présente l'architecture globale de notre système.

Le processus commence par la préparation des données (**Dataset**), suivie d'une étape d'augmentation (**DataAugmentation**) pour enrichir les exemples. Le modèle T5 est ensuite entraîné sur ces données, tandis que les modules **APIService** et **MobileApp** assurent la communication avec l'utilisateur final. Les résultats sont évalués via le module **Évaluation**. La figure 4.39 illustré cette architecture globale.

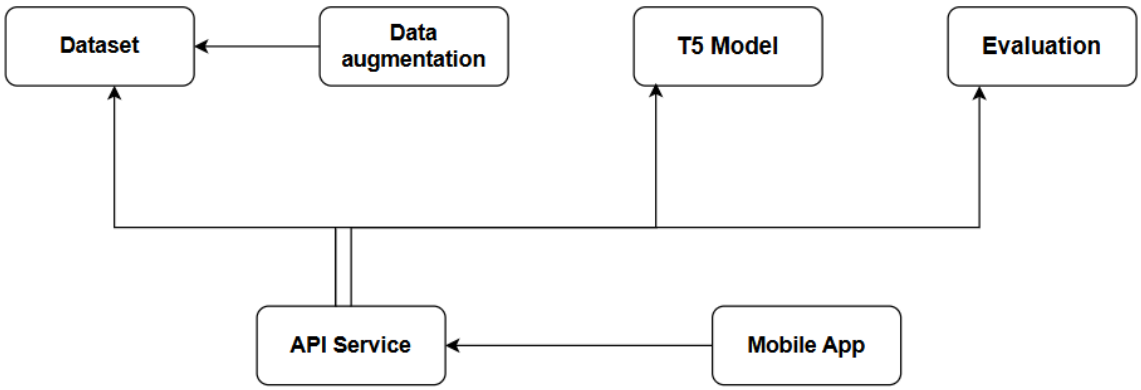


FIGURE 4.39 – Diagramme de package de l'architecture de notre projet.

4.10 Architecture de déploiement

Cette section présente les différents composants impliqués dans l'architecture de déploiement de notre système.

4.10.1 Composants de l'architecture de déploiement

Ce schéma illustre l'architecture d'un système de chatbot dans lequel l'utilisateur interagit via une application mobile ou web. L'interface utilisateur permet d'envoyer des requêtes qui sont ensuite traitées par un serveur backend développé avec Flask.

Ce serveur agit comme un intermédiaire entre l'utilisateur, le modèle d'intelligence artificielle et la base de données. Le modèle d'IA reçoit les données d'entrée, les analyse, et génère une réponse intelligente. Le serveur peut également accéder à la base de données pour stocker ou récupérer des informations comme les questions, réponses ou données utilisateurs. L'ensemble du système peut être déployé sur une infrastructure cloud ou un serveur local. Comme illustré dans la figure 4.40.

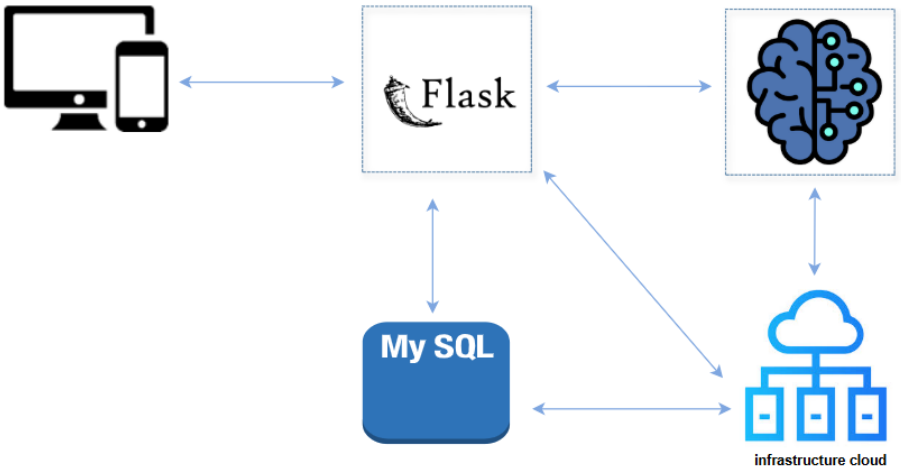


FIGURE 4.40 – Architecture de Déploiement.

L'architecture de déploiement de notre assistant intelligent comprend plusieurs composants essentiels :

— **Interface utilisateur (mobile/web)** : permet à l'utilisateur d'interagir avec le système. La figure 4.41 présente l'écran de discussion de l'interface utilisateur.

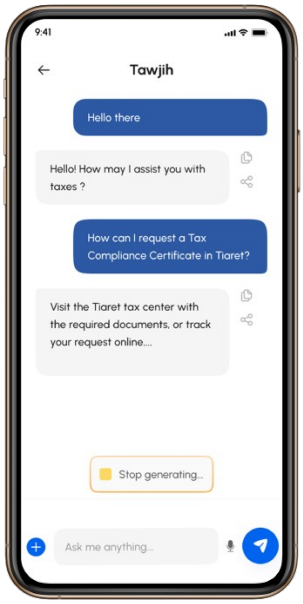


FIGURE 4.41 – Écran de discussion de l'interface utilisateur.

— **Base de données** : stocke les informations nécessaires comme les questions/réponses et les données des utilisateurs.

La figure 4.42 présente la configuration de la base de données locale pour le chatbot.

```
def db_config():
    return { "host": "localhost", "user": "root", "password": "", "database": "
        ↪ chatbot.db" }
```

FIGURE 4.42 – Configuration de la base de données locale pour le chatbot.

Ce code définit une fonction appelée `db-config` qui retourne un dictionnaire avec les informations de connexion à une base de données.

Plus précisément, il utilise l'hôte local (localhost), l'utilisateur root, un mot de passe, et la base de données s'appelle `chatbot.db`.

La figure 4.43 présente un extrait de code Python correspondant au nettoyage de la saisie utilisateur.

```
def clean_user_input(text):
    text = re.sub(r"[^\w\s]", "", text)
    return re.sub(r"\s+", " ", text).strip()
```

FIGURE 4.43 – Un extrait de code Python correspondant au nettoyage de la saisie utilisateur.

Ce code définit une fonction `clean-user-input` qui nettoie la saisie de l'utilisateur en supprimant les caractères spéciaux et les espaces superflus.

La figure 4.44 présente la connexion à la base de données.

```
conn = mysql.connector.connect(**db_config())
```

FIGURE 4.44 – Connexion à la base de données.

Ce code établit une connexion à la base de données `MySQL` en appelant la fonction ‘`db-config()`‘, qui retourne un dictionnaire contenant les informations nécessaires (hôte, utilisateur, mot de passe, nom de la base de données).

La figure 4.45 présente un extrait de code Python correspondant à l'extraction de l'intention utilisateur.

```
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(corpus)
similarities = cosine_similarity(tfidf_matrix[-1], tfidf_matrix[:-1]).flatten()
```

FIGURE 4.45 – Un extrait de code Python correspondant à l'extraction de l'intention utilisateur.

Ce code utilise la méthode `TF-IDF` pour transformer un corpus de textes en vecteurs numériques représentant l'importance des mots dans chaque document. Ensuite, il calcule la similarité cosinus entre le dernier vecteur (correspondant à la requête utilisateur) et tous

les autres vecteurs du corpus. Cette mesure permet d'identifier l'intention de l'utilisateur en trouvant les documents les plus proches de sa requête.

La figure 4.46 présente un extrait de code Python correspondant à la récupération de la réponse .

```
cursor.execute("SELECT response, id FROM response WHERE id_intent = %s", (
    ↪ intent_id,))
```

FIGURE 4.46 – Un extrait de code Python correspondant à la récupération de la réponse .

Ce code exécute une requête SQL qui sélectionne la réponse (response) et son identifiant (id) dans la table response, en filtrant selon l'identifiant de l'intention. La variable `intent id` correspond à l'intention détectée précédemment, ce qui permet d'obtenir la réponse appropriée à la requête de l'utilisateur.

La figure 4.47 présente un extrait de code Python correspondant à la récupération des services associées à une réponse spécifique.

```
cursor.execute("SELECT nom_service, link FROM services WHERE id_response = %s",
    ↪ (response_id,))
```

FIGURE 4.47 – Un extrait de code Python correspondant à la récupération des services associées à une réponse spécifique.

Cette requête récupère `le nom` et `le lien` des services liés à une réponse précise, identifiée par `response-id`, afin de fournir des informations complémentaires à l'utilisateur.

La figure 4.48 présente un extrait de code Python correspondant à la récupération des étapes associées à une réponse spécifique.

```
cursor.execute("SELECT step FROM steps WHERE id_response = %s", (response_id,))
```

FIGURE 4.48 – Un extrait de code Python correspondant à la récupération des étapes associées à une réponse spécifique.

Cette requête extrait les différentes étapes liées à une réponse donnée, permettant de guider l'utilisateur dans le processus correspondant.

La figure 4.49 présente un extrait de code Python correspondant à la récupération des documents à fournir liés à une intention et une réponse spécifiques.

```
cursor.execute("SELECT nom, url FROM dossierafournir WHERE id_intent = %s AND
    ↪ id_response = %s", (intent_id, response_id))
```

FIGURE 4.49 – Un extrait de code Python correspondant à la récupération des documents à fournir liés à une intention et une réponse spécifiques.

Cette requête récupère les noms et liens des documents nécessaires en fonction de l'intention détectée et de la réponse donnée, afin d'informer l'utilisateur des dossiers à fournir.

La figure 4.50 présente un extrait de code Python correspondant à la récupération des contacts et de leurs détails en fonction de l'intention détectée.

```
contacts = get_contacts(intent_id, cursor)
details = get_contact_details(contact_id, cursor)
```

FIGURE 4.50 – Un extrait de code Python correspondant à la récupération des contacts et de leurs détails en fonction de l'intention détectée.

Ces appels de fonctions permettent d'obtenir la liste des contacts liés à une intention ainsi que les informations détaillées associées à un contact spécifique.

— **Serveur backend (API)** : gère les requêtes des utilisateurs et communique avec le modèle d'IA.

La figure 4.51 illustre un extrait d'une API Flask.

```
app = Flask(__name__)

@app.route("/chat", methods=["POST"])
def chat():
    user_input = request.json.get("message", "").lower()
    cleaned_input = clean_user_input(user_input)
    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)
    intent_row = get_intent_from_keywords(cleaned_input, cursor)
    response_row = get_response(intent_row["id_intent"], cursor)
    rag_result = rag_answer(cleaned_input)
```

FIGURE 4.51 – Extrait d'une API Flask.

Ce code crée une API web qui reçoit la question d'un utilisateur, la nettoie, détecte l'intention en interrogeant une base de données, récupère une réponse correspondante, puis utilise un modèle RAG (Retrieval-Augmented Generation) pour générer une réponse plus précise et contextualisée.

— **Modèle d'IA** : génère des réponses intelligentes basées sur les données d'entrée.

4.10.2 Étape du déploiement du modèle d'IA

L'étape du déploiement du modèle d'IA comprend quatre étapes principales : Chargement du modèle, sauvegarde locale du modèle, chargement des éléments sauvegardés, invocation du modèle.

4.10.2.1 Chargement du modèle

La figure 4.52 présente le chargement du modèle.

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
import torch

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
tokenizer = AutoTokenizer.from_pretrained("google/flan-t5-base")
model = AutoModelForSeq2SeqLM.from_pretrained("google/flan-t5-base").to(device)
```

FIGURE 4.52 – Chargement du modèle.

Ce script charge un modèle de traduction ou de génération de texte basé sur Flan-T5 (un modèle de type "Seq2Seq" de Google), en utilisant la bibliothèque Hugging Face Transformers, avec PyTorch comme backend.

4.10.2.2 Sauvegarde locale du modèle

La figure 4.53 présente le sauvegarde locale du modèle.

```
# Sauvegarde du mod le et du tokenizer
model.save_pretrained("saved_model")
tokenizer.save_pretrained("saved_model")

# Sauvegarde de l'index Faiss
import faiss
faiss.write_index(index, "faiss_index.idx")

# Sauvegarde du DataFrame contextuel
df.to_pickle("df.pkl")
```

FIGURE 4.53 – Sauvegarde locale du modèle.

Afin d'assurer la pérennité et la réutilisation du système, les principaux composants sont sauvegardés de manière persistante. Il s'agit du modèle entraîné, de son tokenizer associé, de l'index vectoriel FAISS pour la recherche efficace, ainsi que des données contextuelles structurées. Cette démarche permet de restaurer ultérieurement l'ensemble de l'environnement fonctionnel sans nécessiter de réentraînement ou de reconstruction.

4.10.2.3 Chargement des éléments sauvegardés

La figure 4.54 présente le chargement des éléments sauvegardés.

```
# Chargement du mod le et du tokenizer
tokenizer = AutoTokenizer.from_pretrained("saved_model")
model = AutoModelForSeq2SeqLM.from_pretrained("saved_model").to(device)

# Chargement de l'index Faiss et du DataFrame
index = faiss.read_index("faiss_index.idx")
df = pd.read_pickle("df.pkl")
```

FIGURE 4.54 – Chargement des éléments sauvegardés.

Ce code illustre le chargement des différents composants sauvegardés du système. Il permet de restaurer le modèle pré-entraîné et son tokenizer depuis le dossier(saved model), tout en les plaçant sur l'appareil approprié (CPU ou GPU). Par ailleurs, il recharge également l'index FAISS, utilisé pour la recherche vectorielle rapide, ainsi que le DataFrame pandas contenant les données contextuelles.

4.10.2.4 Invocation du modèle

La figure 4.55 présente l'invocation du modèle.

```
response = rag_answer("Quels sont les documents requis pour obtenir une carte d'
    ↪ identit  ?")
```

FIGURE 4.55 – Invocation du modèle.

cette commande illustre l'appel au modèle pour obtenir une réponse à une question utilisateur. Elle représente l'utilisation effective du système après chargement.

— **Infrastructure cloud ou Serveur Local** : héberge l'ensemble du système pour un accès sécurisé et évolutif.

4.11 Principales limites rencontrées

La table 4.4 présente une synthèse des principales limites identifiées au cours du développement du projet, en précisant les causes et la situation actuelle pour chaque cas.

Limites identifiées	**Explications**	**Situation actuelle**
Absence d'intégration (front-end/back-end)	Retard dans l'entraînement du modèle dû à des ressources limitées	Le chatbot fonctionne via API, et le front-end peut être testé indépendamment.

TABLE 4.4 – Synthèse des principales limites du projet.

4.12 Conclusion

Dans ce chapitre, nous avons détaillé toutes les étapes de l'implémentation de notre assistant intelligent. Nous avons commencé par présenter les technologies et outils utilisés, avant de décrire le processus de construction du dataset, d'entraînement du modèle T5 et d'évaluation de ses performances à l'aide de différentes métriques. Ensuite, nous avons abordé la conception de l'interface utilisateur en mettant en avant l'expérience utilisateur, du flux d'utilisateur jusqu'aux captures de l'application mobile. Enfin, nous avons décrit l'architecture de déploiement du modèle, en expliquant les différents composants et les étapes nécessaires pour rendre le modèle opérationnel dans un environnement réel.

Ce travail d'implémentation constitue une base solide pour l'intégration de solutions intelligentes dans les services administratifs. De plus, nous avons identifié et présenté les principales limites rencontrées au cours du développement, qu'il s'agisse de contraintes techniques, de ressources ou d'intégration. Ces limites nous permettent d'envisager des pistes concrètes d'amélioration dans les travaux futurs.

Conclusion et perspectives

II. Conclusion et Perspectives

II.1- Conclusion

Aujourd'hui, offrir un soutien administratif efficace aux citoyens nécessite des systèmes intelligents capables de comprendre les requêtes complexes et de fournir des réponses précises et contextuelles. Si de nombreuses solutions de chatbot visent à réduire la charge de travail et à améliorer la réactivité des services publics, la plupart peinent à gérer des demandes nuancées et des contextes variés.

Pour relever ces défis, nous avons développé un chatbot intelligent, performant, conçu pour accompagner les citoyens dans leurs démarches administratives. Notre système adopte une architecture hybride combinant une base de connaissances structurée avec un modèle de langage pré-entraîné, en l'occurrence Flan-T5, intégré selon l'approche RAG (Retrieval-Augmented Generation). Cette combinaison permet d'assurer des réponses pertinentes grâce à une recherche sémantique optimisée, suivie d'une génération de texte contextualisée.

Ce projet nous a également permis de mettre en pratique nos connaissances académiques en intelligence artificielle et en traitement automatique du langage naturel, tout en renforçant nos compétences en développement logiciel, en gestion de bases de données et en conception d'interfaces utilisateur adaptées aux flux de travail administratifs réels.

II.2- Perspectives

À la suite des résultats encourageants obtenus dans ce projet, les perspectives futures visent à valoriser et à étendre cette solution à d'autres domaines d'intérêt public. Le modèle développé pourra servir de base à la conception de systèmes similaires dans des secteurs clés tels que la santé ou l'éducation, participant ainsi à l'amélioration de l'accès aux services publics et à l'optimisation des interactions entre les usagers et les administrations.

Ce projet ouvre également la voie à une intégration plus large de ces technologies dans les stratégies nationales de transformation numérique, ce qui pourrait considérablement améliorer l'interaction entre l'administration et les citoyens tout en réduisant la charge sur les ressources humaines.

Une extension multilingue du système constitue également une piste prometteuse afin de garantir l'accessibilité des services à un public plus large, indépendamment de la langue utilisée.

Enfin, ce travail constitue un socle solide pour des recherches ultérieures dans le domaine de l'intelligence artificielle appliquée au service public, favorisant le développement d'applications innovantes basées sur l'analyse linguistique, l'apprentissage automatique et l'interaction naturelle entre l'homme et la machine.

[redacted] Annexe

البطاقة التقنية للمشروع

Moumene Manar Lardjane Imane	الاسم واللقب Votre prénom et nom
Tawjih	الاسم التجاري للمشروع Intitulé de votre projet
Société à Responsabilité Limitée	الصفة القانونية للمشروع Votre statut juridique Your legal status
0669111168 0794861332	رقم الهاتف Votre numéro de telephone
manarmanar20212002@gmail.com lardjaneimane26@gmail.com	البريد الالكتروني Votre adresse e-mail Your email address
Tiaret-Tiaret	مقر مزاولة النشاط (الولاية- البلدية) Votre ville ou commune d'activité

البطاقة التقنية للمشروع

مشروعنا يتمثل في تطوير شات بوت ذكي يهدف إلى تحسين الخدمات الإدارية المقدمة للمواطنين من خلال الذكاء الاصطناعي ومعالجة اللغة الطبيعية باللغة العربية. يعتمد النظام على قاعدة بيانات منظمة للأسئلة والأجوبة، ويُكملها نموذج لغوي ذكي للإجابة عن الاستفسارات غير المعروفة	مسبقًا. يوفر المشروع واجهة استخدام تفاعلية للمواطن، ونظام إدارة خلفي لتحديث المعطيات. كما يتم تقييم أداء الشات بوت باستخدام مقاييس مثل الدقة، مؤشر الجودة اللغوية، ومسافة التعديل. يساهم المشروع في رقمنة الإدارة، تسهيل الوصول إلى المعلومات، وتقليل الضغط على الموظفين.

تحديد المشكل الذي يواجهه الزبون

ما هي المشكلة التي تريد حلها؟	يعاني المواطنون من صعوبة في الوصول إلى معلومات إدارية دقيقة ومحدثة بسبب ضعف التفاعل في المنصات الرقمية وتأخر الردود من المصالح التقليدية، مما يؤدي إلى تضييع الوقت والتنقل غير الضروري.
ما هي البيانات المتوفرة لديك التي تدل على وجود المشكلة المحددة؟	خلال زيارتنا الميدانية لمصالح الضرائب، لاحظنا ضغطاً كبيراً على الإدارة في التعامل مع استفسارات المواطنين، حيث أكد مهندس الإعلام الآلي تكرار نفس الأسئلة من الموظفين والمواطنين، مما يعيق سير العمل ويؤثر على الفعالية، كما أن هاتفه كان يرن باستمرار. الوضع نفسه سُجّل على مستوى البلدية، حيث تتكرر الاستفسارات يومياً حول الوثائق والإجراءات، مما يؤدي إلى إهدار وقت الموظفين وخلق فوضى. هذه الملاحظات كشفت غياب آليات ذكية لتوفير المعلومات، ما دفعنا لاقتراح حل رقمي تفاعلي لمعالجة هذا الإشكال.
ما هي المشاريع الأخرى التي استهدفت نفس المشكلة والتي جرى تنفيذها؟	service-public.dz. موقع "بوابة الخدمات العمومية" الجزائرية **الوصف**: منصة حكومية تقدم معلومات حول الوثائق والخدمات الإدارية. نقاط القوة: محتوى رسمي، متوفر على الإنترنت. **القيود**: غير تفاعلي، لا يجيب على الأسئلة الفردية، التصميم قديم، غير مدعوم بشات بوت أو ذكاء اصطناعي.
ماهي أهداف مشروعك و/أو نتائجه المتوقعة؟	**أهداف المشروع** توفير شات بوت ذكي يجيب فورًا على الأسئلة الإدارية. تقليل الضغط على الموظفين وتقليل التنقلات غير الضرورية للمواطنين. تسهيل الوصول إلى المعلومات الإدارية بشكل رقمي وسهل. **النتائج المتوقعة** رضا أكبر للمواطنين عن الخدمات. تحسين أداء الموظفين الإداريين. تقليل عدد الاستفسارات المتكررة. إمكانية تطوير المشروع ليشمل إدارات أخرى.

القيمة المقترحة وفق المعايير التالية

القيمة المبتكرة أو الجديدة	1. حل ذكي لمشكلة واقعية 2. استخدام تقنيات حديثة 3. تحسين جودة الخدمة الإدارية 4. لا توجد حلول مماثلة حالياً 5. قابل للتطوير والتوسيع
القيمة بالتخصيص	تخصيص الإجابات حسب نوع المستخدم تكييف المحتوى حسب الإدارة أو القطاع دعم تعدد اللغات واللهجات المحلية تخصيص آليات البحث والرد
القيمة بالسعر	تكلفة منخفضة مقارنةً بحلول تقليدية توفير الوقت والجهد مرونة في الاستخدام والتطوير قيمة اقتصادية مضافة

القيمة بالتصميم	واجهة مستخدم بسيطة وسهلة الاستخدام جربة تفاعلية سلسة دعم متعدد الأجهزة جمالية وعصرية تخصيص واجهة المستخدم
القيمة بالأداء العالي	سرعة استجابة فورية اعتمادية واستقرار تحمل الضغط العالي تكامل سلس مع الأنظمة الأخرى كفاءة في استهلاك الموارد
القيمة بالخدمة الشاملة	دعم تفاعلي مستمر تحديث مستمر للمعلومات تحليل بيانات المستخدمين توفير قنوات تواصل متعددة تخصيص حسب حاجات الإدارة
قيم أخرى	القيمة الاجتماعية القيمة التقنية القيمة التعليمية القيمة البيئية القيمة التنظيمية

شرائح العملاء أو الزبائن Customer Segments

Géographique الجغرافية	Démographique (B2C)	Démographique (B2B)	Psychographique العوامل النفسية و الشخصية	Comportemental السلوكيات
Continent القارة أفريقيا	Age العمر +18	Secteur القطاع Services sector	Classe sociale الطبقة الاجتماعية الطبقة المتوسطة الطبقة الدنيا	Usage استخدام
Pays الدولة الجزائر	Sexe الجنس ذكر انثى	Nombre d'employés عدد العمال في القطاع	Niveau de vie المستوى المعيشي Standard Of living ضعيف متوسط مرتفع	Loyauté الوفاء
Région الجهة الغرب	Revenus annuel متوسط الدخل	Maturité de l'entreprise نضج المؤسسة	Valeurs القيم إمكانية الوصول، الأمان، سهولة الاستخدام، المتانة الاقتصادية	Intérêt اهتمام
Département الولاية تيارت	Etat matrimonial الحالة الاجتماعية	Situation financière الحالة المالية للمؤسسة	Personnalité الشخصية الموثوقية. المسؤولية الاجتماعية. سهولة الوصول.	Passion الهواية و شغف
llVille الدائرة او البلدية	Niveau d'étude المستوى الدراسي	Détention/ actionnariat	Convictions المعتقدات	Sensibilité حساسيات

Tiaret	All	الملكية/المساهمة		
Quartier الحي /	Profession المهنة الكل	Valorisation/ capitalisation boursière التقييم / القيمة السوقية	Présence digitale et sur les réseaux sociaux استعمال التكنولوجيا في التواصل إمكانية التواصل مع العملاء على شبكات التواصل الاجتماعي	Habitude de Consummation عادة الاستهلاك
Climat المناخ /	Culture الثقافة /	Business model نموذج الأعمال	Centres d'intérêts مراكز الاهتمام المجتمع الاقتصاد المالي الذكاء الاصطناعي	Mode de Paiement طرق الدفع
	Religion الدين الاسلام	Secteur servi القطاع الذي يخدمه		Connaissance المعرفة
	Langue اللغة العربية الإنجليزية العربيزي (arabizi)	Technologie utilisée التكنولوجيا المستعملة Flutter Python		Nature de la Demande طبيعة الطلب

		Mysql		
Fréquence d'achat عدد مرات الطلب على السلعة		Format du produit ou packaging شكل المنتج أو التعبئة والتغليف Mobile app (تطبيق هاتف محمول)		

قنوات التوزيع Channels

المبيعات المباشرة	بيع رخص استخدام الشات بوت مباشرة للدوائر والمؤسسات الحكومية. تقديم خدمات الدعم الفني والتدريب للمؤسسات التي تعتمد النظام. توفير نسخ مخصصة أو مدعمة بمزايا إضافية تُباع مباشرة حسب حاجة الإدارة. التعاقد المباشر مع البلديات، المكاتب الإدارية، ووزارات لتبني الشات بوت.
تجار الجملة	شركات تكنولوجيا المعلومات أو الاستشارات موزعو البرمجيات شركات تطوير الحلول الرقمية
الموزعون	شركات تكنولوجيا متخصصة شركات الاتصالات شركات التدريب منصات البيع الإلكتروني
توزيع التجزئة	تقديم الشات بوت كتطبيق ويب أو تطبيق هاتف ذكي متاح للتنزيل والاستخدام المباشر من قبل المواطنين والموظفين. البيع عبر منصات رقمية متخصصة في حلول الإدارة الحكومية والخدمات الإلكترونية. توفير نسخ فردية أو اشتراكات شهرية للمستخدمين أو الإدارات الصغيرة التي ترغب في تحسين خدماتها دون الحاجة لتعاقدات كبيرة. الترويج عبر الحملات الإعلامية ووسائل التواصل الاجتماعي لجذب المستخدمين النهائيين.

العلاقة مع العملاء Customer Relationship

كيف تدير علاقاتك مع العملاء؟	. توفير دعم فني متواصل التواصل المستمر استقبال الملاحظات والاقتراحات تخصيص الخدمة تقديم تدريب وورش عمل بناء علاقات طويلة الأمد

ماهية أهم البرامج التي ستعتمد عليها في ادارة العلاقة مع الزبون Microsoft Dynamics Monday CRM	Hub Spot CRM: هو البرنامج الأساسي لإدارة علاقات العملاء في مشروعنا. يتميز بسهولة الاستخدام، إمكانية تتبع جميع تفاعلات المواطنين مع الشات بوت، وتنظيم بياناتهم بشكل مرتب. كما يدعم أتمتة المهام، إرسال التذكيرات، وتحليل الأداء عبر تقارير دقيقة تساعدنا في تحسين الخدمة باستمرار. نستخدمه أيضًا لربطه مع الشات بوت عبر واجهة برمجة التطبيقات، مما يسمح بتسجيل بيانات المحادثات ومتابعة استفسارات المستخدمين بشكل آلي وفعّال.

Zoho CRM الخ.....	

الشركاء الأساسيون Key Partners

الشركاء	معلومات حول الشركاء	طبيعة الشراكة
المشرفون الأكاديميون	المشرفون الأكاديميون هم أساتذة جامعيون أو باحثون مختصون يتولّون متابعة المشروع من الجانب العلمي. يتمتعون بخبرة أكاديمية تساعد على توجيه الفريق وضمان أن المشروع يسير وفق أسس منهجية صحيحة، خاصة إذا كان ضمن إطار بحثي أو أكاديمي.	العلاقة مع المشرفين الأكاديميين شراكة علمية واستشارية، حيث يقدمون الدعم المعرفي والتوجيه المنهجي للمشروع، ويساهمون في مراجعة وتحسين المخرجات لضمان جودتها ومطابقتها للمعايير الأكاديمية. هذه العلاقة ليست علاقة تنافس، بل هي ترتيب متبادل المنفعة يقوم على الاحترام والتعاون المستمر، حيث نعتبر المشرفين شركاء لنا في العملية الأكاديمية، وعلاقتنا معهم تشبه علاقة المشتري والمورد التي ترتكز على تبادل المعرفة والخبرة لضمان نجاح المشروع.
إدارات الضرائب	إدارات الضرائب هي جهات حكومية مسؤولة عن تطبيق القوانين الضريبية وجمع الإيرادات الوطنية. تتميز هذه الإدارات بخبرتها الواسعة في تنظيم السياسات الضريبية وتوفير الإرشادات للمكلفين لضمان الامتثال الضريبي وتحقيق العدالة المالية.	تقوم الشراكة مع إدارات الضرائب على التعاون والتنسيق لتبادل المعلومات والخبرات، مع التأكيد أن الشريك ليس منافسًا بل هو شريك في عملية ترتيب متبادلة المنفعة. علاقتنا مع إدارات الضرائب هي علاقة مشتري ومورد، حيث نستلهم خبراتهم ونستفيد من توجيهاتهم لضمان توافق المخرجات مع اللوائح والقوانين المعمول بها، مع الحفاظ على التعاون المستمر والشفافية بين الطرفين.
مزودو نماذج الذكاء الاصطناعي	مزودو نموذج الذكاء الاصطناعي هم شركات أو مؤسسات تزوّدنا بالنماذج والخوارزميات المتقدمة التي تعتمد عليها حلولنا الذكية. لديهم خبرة تقنية عالية في تطوير نماذج تعلم الآلة والتعلم العميق، ويساهمون في توفير البنية التحتية اللازمة لتشغيل هذه النماذج بكفاءة وفعالية.	ترتكز الشراكة مع مزودي نماذج الذكاء الاصطناعي على تعاون تقني وتجاري قائم على ترتيب متبادل المنفعة، حيث لا تُعد هذه العلاقة تنافسية بل هي شراكة استراتيجية تشبه علاقة المشتري والمورد. نستفيد من خبراتهم وتقنياتهم لتطوير مشروعنا، مع الحفاظ على علاقة شفافة ومستدامة تحقق مصالح الطرفين وتدعم نجاح الحلول المقدمة.

هيكل التكاليف structure Costs

تكاليف التعريف بالمنتج أو المؤسسة Frais d'établissement	بما أن الفريق مكوّن من مختصين في الإعلام الآلي، سيتم الاعتماد على وسائل رقمية منخفضة التكلفة للترويج، مثل: إنشاء موقع إلكتروني بسيط.. حملات مجانية عبر وسائل التواصل الاجتماعي.. عروض تجريبية موجهة للإدارات. المشاركة في فعاليات ومعارض محلية. كل هذه الأنشطة ستُنفذ داخل الفريق، مما يقلل التكاليف بشكل كبير دون التأثير على جودة التعريف بالمشروع. من100, 000دج
تكاليف الحصول على العدادات (الماء- الكهرباء .....) Frais d'ouverture de compteurs (eaux-gaz-….)	دج 15,000~ عداد الكهرباء دج 12,000~ عداد الماء دج 25,000~ عداد الغاز دج 52,000~ المجموع
تكاليف (التكوين- برامج الاعلام الالي المختصة) Logiciels, formations	اقتناء حاسوب بمواصفات عالية يدعم بيئات الذكاء الاصطناعي/ بطاقة رسومية قوية لتسريع عملية التدريب والتجارب 400,000دج
Dépôt marque, brevet, modèle تكاليف براءة الاختراع و الحماية الصناعية و التجارية	رسوم تسجيل براءة الاختراع، حماية الاسم التجاري والشعار، والاستشارات القانونية لضمان حقوق الملكية الفكرية. كما تشمل أيضًا حماية الكود البرمجي والمحتوى الرقمي من النسخ أو الاستغلال غير القانوني. 20,000دج
Droits d'entrée تكاليف الحصول على تكنولوجيا او ترخيص استعمالها	/
Achat fonds de commerce ou parts شراء الأصول التجارية أو الأسهم	/
Droit au bail الحق في الإيجار	/
Caution ou dépôt de garantie وديعة أو وديعة تأمين	/
Frais de dossier رسوم إيداع الملفات	من 20.000دج
Frais de notaire ou d'avocat تكاليف الموثق-المحامي-.....	من 50.000دج
Enseigne et éléments de communication تكاليف التعريف بالعلامة و تكاليف قنوات الاتصال	من 50.000دج
Achat immobilier شراء العقارات	/
Travaux et aménagements الأعمال والتحسينات الاماكن	من 200.000دج

Matériel الآلات- المركبات- الاجهزة	من 900.000 دج سطح مكتبين للمسؤولين. - خادم للاستضافة -.
Matériel de bureau تجهيزات المكتب	من 300.000 دج طاولات وكراسي وطابعات,
Stock de matières et produits تكاليف التخزين	Hosting: Cloud / Algeria Telecom استضافة: سحابية / اتصالات الجزائر من 500.000 دج
Trésorerie de départ التدفق النقدي(الصندوق) الذي تحتاجه في بداية المشروع.	من 1.000.000دج

المجموع = 3.592.000دج

نفقاتك أو التكاليف الثابتة الخاصة بمشروعك

Assurances التأمينات	من 50.000 دج
Téléphone, internet الهاتف و الانترنت	باقة إنترنت 30 ميجابايت 30.000 دينار/شهريًا
Autres abonnements اشتراكات أخرى	/
Carburant, transports الوقود و تكاليف النقل	/
Frais de déplacement et hébergement تكاليف التنقل و المبيت	/
Eau, électricité, gaz فواتير الماء – الكهرباء- الغاز	من 30.000دج
Mutuelle التعاضدية الاجتماعية	/
Fournitures diverses لوازم متنوعة	من 35.000 دج
Entretien matériel et vêtements صيانة المعدات والملابس	/
Nettoyage des locaux تنظيف المباني	من 20.000 دج
Budget publicité et communication ميزانية الإعلان والاتصالات	من 60.000 دج

المجموع =225.000 da

مصادر الإيرادات Revenue Stream

Apport personnel ou familial المساهمة الشخصية أو العائلية	1.500.000دج
Apports en nature (en valeur) التبرعات العينية	/
Prêt n°1 (nom de la banque) قرض رقم 1 اسم البنك	/
Prêt n°2 (nom de la banque) قرض رقم 2 اسم البنك	/

Prêt n°3 (nom de la banque) قرض رقم3 اسم البنك	/
Subvention n°1 (libellé) منحة 1	/
Subvention n°2 (libellé) منحة 2	/
Autre financement (libellé) تمويل آخر	/

المجموع = 1.500.000دج

رقم الأعمال

بيع المنتج في السنة الأولى Votre chiffre d'affaires de la première année

بيع المنتج في السنة الأولى	متوسط أيام العمل في الشهر
1Mois الشهر 20.000	20
2Mois الشهر 22.000	20
3Mois الشهر 24.000	20
4Mois الشهر 23.000	20
5Mois الشهر 26.000	20
6Mois الشهر 27.000	20
7Mois الشهر 28.000	20
8Mois الشهر 29.000	20
9Mois الشهر 30.000	20
10Mois الشهر 31.000	20
11Mois الشهر 32.000	20
12Mois الشهر 33.000	20

المجموع =325.000

النسبة المئوية للزيادة في حجم الأعمال بين كل شهر لسنة الأولى؟ 6.12%

بيع المنتج في السنة الثانية Votre chiffre d'affaires de la deuxième année

بيع المنتج في السنة الثانية	متوسط أيام العمل في الشهر
1Mois الشهر 21.224	20
2Mois الشهر 23.346	20
3Mois الشهر 25.469	20

4Mois 24.408 الشهر	20
5Mois 27.591 الشهر	20
6Mois 28.551 الشهر	20
7Mois 29.612 الشهر	20
8Mois 30.673 الشهر	20
9Mois 31.836 الشهر	20
10Mois 32.897 الشهر	20
11Mois 33.958 الشهر	20
12Mois 35.019 الشهر	20

المجموع =344.000

النسبة المئوية للزيادة في حجم الأعمال بين كل شهر لسنة الثانية؟4.47 %

Votre chiffre d'affaires de la troisième année بيع المنتج في السنة الثالثة

بيع المنتج في السنة الثالثة	متوسط أيام العمل في الشهر
1Mois الشهر22.167	20
2Mois الشهر24.386	20
3Mois الشهر26.607	20
4Mois الشهر25.495	20
5Mois الشهر28.821	20
6Mois الشهر29.830	20
7Mois الشهر30.934	20
8Mois الشهر32.038	20
9Mois الشهر33.238	20
10Mois الشهر34.341	20
11Mois الشهر35.445	20
12Mois الشهر36.549	20

المجموع = 359.851

النسبة المئوية للزيادة في حجم الأعمال بين كل شهر لسنة الثالثة ؟4.47%

تطور حجم رقم الأعمال في السنة

- النسبة المئوية للزيادة في حجم الأعمال بين السنة 1 والسنة 2؟ النسبة المئوية للزيادة
- في حجم الأعمال بين السنة 2 والسنة 3 ؟

حاجتك لرأس المال العامل

متوسط مدة الاعتمادات الممنوحة للعملاء بالأيام Durée moyenne des crédits accordés aux clients en jours	30 يوم
متوسط مدة ديون الموردين بالأيام Durée moyenne des dettes fournisseurs en jours	365 يوم

رواتب الموظفين و مسؤولين الشركة

رواتب الموظفين Salaires employés	50.000 دج / شهريا
صافي أجور المسؤولين Rémunération nette dirigeant	60.000 دج/ شهريا

Business Model Canvas

Designed for:

Designed by:

Date:

Version:

Key Partners

Who are our Key Partners? Who a
key suppliers? Which Key Resour
we acquiring from partners? Whic
Activities do partners perform?

MOTIVATIONS FOR PARTNERS
Optimization and economy, Reduc
risk and uncertainty, Acquisition o
particular resources and activities

Key Activities

What Key Activities do our Value
Propositions require? Our Distribu
Channels? Customer Relationship
Revenue streams?

CATEGORIES:
Production, Problem Solving,
Platform/Network

Key Resources

What Key Resources do our Value
Propositions require? Our Distribu
Channels? Customer Relationship
Revenue Streams?

TYPES OF RESOURCES: Physic
Intellectual (brand patents, copyrig
data), Human, Financial

Value Propositions

What value do we deliver to the c
Which one of our customer's prob
we helping to solve? What bundle
products and services are we offe
each Customer Segment? Which
needs are we satisfying?

CHARACTERISTICS: Newness,
Performance, Customization, "Ge
Job Done", Design, Brand/Status,
Cost Reduction, Risk Reduction,
Accessibility, Convenience/Usabil

Customer Relationships

What type of relationship does ea
Customer Segments expect us to
and maintain with them? Which o
we established? How are they inte
with the rest of our business mode
costly are they?

Channels

Through which Channels do our C
Segments want to be reached? H
we reaching them now? How are
Channels integrated? Which ones
best? Which ones are most cost-e
How are we integrating them with
routines?

Customer Segments

For whom are we creating value?
our most important customers? Is
customer base a Mass Market, Ni
Market, Segmented, Diversified, M
sided Platform

Cost Structure

What are the most important costs inherent in our business model? Which Key Resour
most expensive? Which Key Activities are most expensive?

I

BUSINESS MORE: Cost Driven (leanest cost structure, low price value proposition, m
automation, extensive outsourcing), Value Driven (focused on value creation, premium
proposition).

SAMPL

CHARACTERISTICS: Fixed Costs (salaries, rents, utilities), Variable costs, Economie
scale, Economies of scope

Revenue Streams

For what value are our customers really willing to pay? For what do they currently pay?
they currently paying? How would they prefer to pay? How much does each Revenue S
contribute to overall revenues?

TYPES:

sale, Usage fee, Subscription Fees, Lending/Renting/Leasing, Licensing, Brokerage fee
Advertising

FIXED PRICING: List Price, Product feature dependent, Customer segment dependent
dependent

DYNAMIC PRICING: Negotiation (bargaining), Yield Management, Real-time-Market

	Designed for:	*Designed by:*	*Date:*	*Version :*
Business Model Canvas		Moumene Manar Lardjane Imane	-29/05/2025	

Key Partners	Key Activities	Value Propositions	Customer Relationships	Customer Segments
المشرفون الأكاديميون إدارات الضرائب - مزودو نماذج الذكاء الاصطناعي - منصات التدريب - موردو الأجهزة "حاسوب، بطاقة - T100 : الموارد بيانات واقعية، أدوات تطوير، نماذج" أنشطة الشركاء ا دعم فني، تجريب،تدريب	صيانة الشات بوت وتحديثه تدريب النموذج ببيانات - الإدارات UX/UI تصميم - النشر على الويب والموبايل - التسويق الرقمي - دعم فني وإدارة المنصة -	ردود سريعة ودقيقة تقليل الضغط على الإدارات – تحسين تجربة المواطن – - الوصول في أي وقت تخصيص، راحة، أداء عال – تصميم جذاب، تكلفة منخفضة –	تواصل فوري وسلس استقبال شكاوى واقتراحات – إشعارات بالتحديثات – علاقة مبنية على الأتمتة – والدعم الذكي تكلفة منخفضة وثقة عالية –	المواطنون من مختلف الأعمار إدارات الدولة ومؤسساتها - سوق متخصص ومتعدد الجوانب - :نخلق القيمة لـ المواطنين "معلومة سريعة" الموظفين "تقليل الضغط "
	Key Resources		**Channels**	
	مطورون , فريق ,بشري : مشرفون دعم بشري :مالية تمويل ميدئي، اشتراكات وأدوات مادية : اجهزة ,استضافة سحابية , نماذج ذكاء اصطناعي قاعدة بيانات , ,فكرية : التصميم ملكية الاسم		تطبيق موبايل بسيط موقع إلكتروني – شاشات إلكترونية بالإدارات – :تكامل البيانات، وصول سريع الفعالية تطبيق رقمي وموقع منخفضي :التكاليف التكلفة	

Cost Structure

- اشتراكات "LLM"
- شراء أجهزة وخوادم
- تكوين متقدم ودورات CRM
- تطوير الواجهات والخوارزميات
- صيانة وتحديث التطبيق
- تكاليف ثابتة ومتغيرة
التركيز على الجودة والقيمة

Revenue Streams

- اشتراكات شهرية/سنوية للمؤسسات
- رسوم حسب الطلب للمستخدمين
- ترخيص الاستخدام للهيئات الحكومية
- بيع النسخة الكاملة (Asset Sale)
- إعلانات تعليمية وإدارية
سعر ثابت، حسب عدد المستخدمين أو تفاوضي التسعير:
اعتماد على الاشتراكات والتراخيص الدخل:

Bibliographie

[1] E. Adamopoulou and L. Moussiades. Chatbots : History, technology, and applications. *Machine Learning with Applications*, 2 :100006, 2020.

[2] E. Adamopoulou and L. Moussiades. An overview of chatbot technology. In *Artificial Intelligence Applications and Innovations*, volume 584 of *Lecture Notes in Computer Science*, pages 373–383. Springer, May 2020.

[3] M. M. Adaobi, M. Nyangu, and H. S. Fook. The role of artificial intelligence in transforming customer experience in the service industry in nigeria. *International Journal of Academic Research in Business & Social Sciences*, 14(11) :2531, Nov. 2024.

[4] B. A. Alazzam, M. Alkhatib, and K. Shaalan. Artificial intelligence chatbots : A survey of classical versus deep machine learning techniques. *Information Sciences Letters*, 12(4) :1217–1233, Apr. 2023.

[5] H. S. Arslan, M. Fishel, and G. Anbarjafari. Doubly attentive transformer machine translation. *arXiv preprint*, arXiv :1807.11605, July 2018.

[6] P. B, J. M. Nandhini, and T. Gnanasekaran. An analysis of the applications of natural language processing in various sectors. In V. D. A. Kumar, S. Malathi, V. E. Balas, M. Favorskaya, and T. Perumal, editors, *Advances in Parallel Computing*. IOS Press, 2021.

[7] L. Benaddi, C. Ouaddi, A. Jakimi, and B. Ouchao. A systematic review of chatbots : Classification, development, and their impact on tourism. *IEEE Access*, 12 :78799–78810, 2024.

[8] M. Chen, M. Gascó-Hernandez, and M. Esteve. The adoption and implementation of artificial intelligence chatbots in public organizations : Evidence from u.s. state governments. *American Review of Public Administration*, 1 :1–16, 2023.

[9] J. H. R. Chiru. The evolution of natural language processing, October 2023. Medium.

[10] P. Choudhary and S. Chauhan. An intelligent chatbot design and implementation model using long short-term memory with recurrent neural networks and attention mechanism. *Decision Analytics Journal*, 9 :100359, Dec. 2023.

[11] T. Czerny. A brief intro to large language models (llms), January 2024. Medium.

[12] S. Dash. What is retrieval-augmented generation (rag) ?, September 2023. Analytics Vidhya.

[13] P. Dhyani. Natural language processing : Top key challenges & applications, January 2022. Jellyfish Technologies.

[14] V. Ditote, V. Bidve, D. K. Singh, S. S. Pathan, et al. Are ai and chat bots services effects the psychology of users in banking services and financial sector. ResearchGate, Aug. 2023. Available at : https://www.researchgate.net/.

[15] Eastgate Software. Top 8 applications of natural language processing (nlp), October 2024. Medium.

[16] Indiana Steel Fabricating. Understanding fine-tuning of large language models (llms), 2023.

[17] P. Foy. Understanding transformers & the architecture of llms, 2024. MLQ.ai Blog.

[18] A. Følstad and M. B. Skjuve. Chatbots for customer service : User experience and motivation. In *Proceedings of the Conference*, Aug. 2019.

[19] K. Gao, S. He, Z. He, J. Lin, Q. Pei, J. Shao, and W. Zhang. Examining user-friendly and open-sourced large gpt models : A survey on language, multimodal, and scientific gpt models. *arXiv preprint*, arXiv :2308.14149, August 2023.

[20] Y. Gao et al. Retrieval-augmented generation for large language models : A survey. *arXiv preprint*, arXiv :2312.10997, December 2023.

[21] GeeksforGeeks. Libraries in python, 2023.

[22] M. U. Hadi et al. A survey on large language models : Applications, challenges, limitations, and practical usage. *Authorea Preprints*, 3, 2023. Preprint.

[23] H. Hassan et al. Review and classification of ai-enabled covid-19 ct imaging models based on computer vision tasks. *Computers in Biology and Medicine*, 141 :105123, 2022.

[24] X. Huang and A. CIS. *Chatbot : Design, Architecture, and Applications.* University of Pennsylvania : School of Engineering and Applied Science, Pennsylvania, 2021.

[25] S. Jana, R. Biswas, K. Pal, and S. Biswas. The evolution and impact of large language model systems : A comprehensive analysis. ResearchGate preprint, March 2024. Preprint.

[26] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3) :535–547, 2019.

[27] B. Joshi. Levels in natural language processing (nlp), February 2022. Medium.

[28] S. S. Khandare. *Mastering Large Language Models : Advanced Techniques, Applications, Cutting-Edge Methods, and Top LLMs (English Edition).* BPB Publications, 2024.

[29] Labs, Kodexo. The role of generative ai in natural language processing (nlp), December 2023. Medium.

[30] J. R. López and M. J. González. Artificial intelligence in public administration : A case study of citizen interaction in local governments. *Public Administration Review*, 80(4) :678–695, 2020.

[31] S. Mashatian et al. Building trustworthy generative artificial intelligence for diabetes care and limb preservation : A medical knowledge extraction case. *J. Diabetes Sci. Technol.*, page 19322968241253568, May 2024.

[32] Wes McKinney. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, pages 51–56, 2010.

[33] L. Monigatti. Advanced retrieval-augmented generation : From theory to llamaindex implementation, February 2024. TDS Archive.

[34] K. K. Nirala, N. K. Singh, and V. S. Purani. A survey on providing customer and public administration based services using ai : chatbot. *Multimedia Tools and Applications*, 81(16) :22215–22246, 2022.

[35] M. Pant, T. K. Sharma, O. P. Verma, R. Singla, and A. Sikander, editors. *Soft Computing : Theories and Applications : Proceedings of SoCTA 2018*, volume 1053 of *Advances in Intelligent Systems and Computing*, Singapore, 2020. Springer Singapore.

[36] V. B. Parthasarathy, A. Zafar, A. Khan, and A. Shahid. The ultimate guide to fine-tuning llms from basics to breakthroughs : An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint*, arXiv :2408.13296, August 2024.

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch : An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[38] R. Patil and V. Gudivada. A review of current trends, techniques, and challenges in large language models (llms). *Applied Sciences*, 14(5) :2074, 2024.

[39] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthias Perrot, and Édouard Duchesnay. Scikit-learn : Machine learning in python. *Journal of Machine Learning Research*, 12 :2825–2830, 2011.

[40] Pallets Projects. Flask documentation, 2024.

[41] Eleonora Pura. *Designing a Chatbot*. Ph.d. thesis, University of Zurich, 2021.

[42] Python Software Foundation. json — JSON encoder and decoder. Python 3.12.3 Documentation, 2024.

[43] M. A. K. Raiaan et al. A review on large language models : Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 12 :26839–26874, 2024.

[44] M. Ramponi. The full story of large language models and rlhf, May 2023. News, Tutorials, AI Research.

[45] Real Python. Working with json data in python, 2024.

[46] Nils Reimers and Iryna Gurevych. Sentence-bert : Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv :1908.10084*, 2019.

[47] Yasser Sabbour. deep-translator : A flexible free and unlimited python tool to translate between different languages, 2024.

[48] S. Sarbabidya and T. Saha. Role of chatbot in customer service : A study from the perspectives of the banking industry of bangladesh. *International Review of Business Research Papers*, 16(1) :231–248, Mar. 2020.

[49] E. S. Shahul. Data augmentation in nlp : Best practices from a kaggle master, 2023.

[50] A. Srivastav. Building your own large language model (llm) : A step-by-step guide, October 2024. Medium.

[51] K. Taghandiki and M. Mohammadi. Large language models training, challenges, applications, and development. *TechRxiv*, June 2024.

[52] Tithi. Essential text pre-processing techniques for nlp !, September 2021. Analytics Vidhya.

[53] Femi Uzoka, Linda Smith, and Wei Chen. Ai-powered chatbots in customer service : Balancing efficiency and interaction quality. *arXiv preprint arXiv :2310.05421*, 2024.

[54] S. Wahbi, K. Khaddouj, and N. Lahlimi. Study of the relationship between chatbot technology and customer experience and satisfaction. *International Journal of Accounting, Finance, Auditing, Management and Economics (IJAFAME)*, 4(6-1) :758–771, 2023.

[55] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clément Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Alexander M. Rush. Transformers : State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations*, pages 38–45, 2020.

[56] J. J. Y. Zhang, A. Følstad, and C. A. Bjørkli. Organizational factors affecting successful implementation of chatbots for customer service. *Journal of Internet Commerce*, 2021.