



CHECKSEM
SEMANTIC **INTELLIGENCE** RESEARCH

Projet Chatbot 2.0

AGENTS CONVERSATIONNELS

RAPPORT DE PROJET TUTORÉ | Dorian NAAJI – IQS4B2
IUT DIJON | DÉPARTEMENT INFORMATIQUE

Le projet Chatbot 2.0 est un approfondissement du travail que j'ai réalisé lors du projet tutoré du troisième semestre, dirigé par Monsieur Nicolle. Chatbot 2.0 est un projet qui se divise en trois grandes phases de travail. Premièrement, une phase d'**étude de technologies** puis la conception et le développement de l'**application** et finalement la conception et le développement de l'**interface utilisateur**. Mon choix de prolonger le projet entrepris au troisième semestre se justifie de l'intérêt que j'ai porté aux problématiques que posait le sujet, mais également du fait qu'aussi bien dans des domaines commerciaux que de la recherche, l'intelligence artificielle est un phénomène d'actualité qui ne cesse d'évoluer. Les agents conversationnels et assistants vocaux sont depuis 2016 incontournables pour de nombreux dispositifs et deviennent une alternative très souvent envisagée à certaines interfaces.

REMERCIEMENTS

Je tenais à remercier Monsieur Nicolle, superviseur du projet Chatbot durant le 3^e Semestre, sans qui ce projet Chatbot 2.0 n'aurait jamais vu le jour. Nos réunions régulières lors du 3^e Semestre étaient une source de motivation et son enseignement m'a permis d'enrichir ma vision du monde en tant qu'informaticien. Les deux projets « Chatbot » ont été pour moi un réel accomplissement personnel grâce à Monsieur Nicolle.

Je remercie également l'ensemble de l'équipe enseignante de l'IUT, notamment Monsieur Meunier qui par le biais de son enseignement m'a permis de maîtriser le développement en JavaScript, un langage de programmation. Également, toute ma gratitude s'accorde à Madame Menissier, qui tout au long des semestres, a toujours accompagné ses étudiants dans la rédaction de leurs documents dits « professionnels ».

TABLE DES MATIERES

| | |
|--|-----------|
| REMERCIEMENTS | 3 |
| TABLE DES ILLUSTRATIONS | 7 |
| CONVENTION TYPOGRAPHIQUE | 8 |
| I. THÈME DU PROJET | 9 |
| 1. PRÉAMBULE | 9 |
| 2. PRÉSENTATION ET THÈME DU PROJET | 9 |
| 2.1. AGENT CONVERSATIONNEL ET INTELLIGENCE ARTIFICIELLE | 9 |
| 2.2. BASE DE CONNAISSANCE | 10 |
| 2.3. EXISTANT : PROJET TUTEUR DU SEMESTRE 3 | 11 |
| 3. RÉFLEXIONS SUR LE TRAVAIL À EFFECTUER | 11 |
| 3.1. DÉFINITION DE LA MISSION DU PROJET | 11 |
| 3.2. ÉTUDE COMPARATIVE DE MOTEURS DE TRAITEMENT DU LANGAGE NATUREL | 12 |
| 3.3. ÉTUDE DE BASES DE CONNAISSANCES | 13 |
| II. TRAVAIL EFFECTUÉ | 14 |
| 1. MATÉRIEL, TECHNOLOGIES ET LOGICIELS UTILISÉS | 14 |
| 1.1. MATÉRIEL UTILISÉ | 14 |
| 1.2. TECHNOLOGIES UTILISÉES | 15 |
| 1.3. LOGICIELS UTILISÉS | 16 |
| 2. ORGANISATION | 17 |
| 2.1. MÉTHODE DE GESTION DE PROJET | 17 |
| 2.2. SUIVI DU PROJET | 17 |
| 3. REFACTORISATION ET AMÉLIORATION DU CODE SOURCE | 19 |
| 4. AMÉLIORATIONS DE L'INTERFACE UTILISATEUR | 19 |
| 5. AMÉLIORATION DU CHATBOT INITIAL : « WORDBOT » | 20 |
| 6. DÉVELOPPEMENT D'UN SECOND CHATBOT : « WEATHERBOT » | 20 |

| | | |
|-------------|--|-----------|
| 7. | DÉVELOPPEMENT D'UN TROISIÈME CHATBOT : « CURRENCYBOT » | 20 |
| 8. | CONCLUSION | 21 |
| 8.1. | LIVRABLE FONCTIONNEL | 21 |
| 8.2. | LEÇONS TIRÉES DE CE TRAVAIL | 21 |
| III. | DOCUMENTS ANNEXES | 22 |
| 1. | LIENS | 22 |
| 2. | ANNEXES | 23 |
| 3.1. | ANNEXE 1 – CHATBOT GOOGLE TRENDS | 23 |
| 3.2. | ANNEXE 2 – FONCTIONNEMENT GLOBAL D'UN CHATBOT | 24 |
| 3.3. | ANNEXE 3 – OFFRE DE STAGE AU CDS : NLU | 25 |
| 3.4. | ANNEXE 4 – ÉTUDE COMPARATIVE REALISEE LORS DU PROJET DE S3 DE DIFFERENTS MOTEURS DE NLU | 26 |
| 3.5. | ANNEXE 5 – CHAMP D'UTILISATION DU CHATBOT | 27 |
| 3.6. | ANNEXE 6 – INTERFACE DU CHATBOT S3 | 28 |
| 3.7. | ANNEXE 7 – LOGIQUE D'INTERACTION DU CHATBOT S3 | 29 |
| 3.8. | ANNEXE 8 – COMPETENCY QUESTIONS | 30 |
| 3.9. | ANNEXE 9 – DÉMONSTRATION DU CHATBOT | 31 |
| 3.10. | ANNEXE 10 – GRANDES PHASES ITÉRATIVES DU PROJET | 34 |
| 3.11. | ANNEXE 11 – ETUDE COMPARATIVES DE PLATEFORMES DE NLU | 35 |
| 3.12. | ANNEXE 12 – INTERFACE DE DIALOGFLOW | 36 |
| 3.13. | ANNEXE 13 – INTERFACE DE WIT.AI | 37 |
| 3.14. | ANNEXE 14 – INTERFACE DE LUIS.AI | 39 |
| 3.15. | ANNEXE 15 – SERVICES ET TARIFS OPENWEATHER MAP | 42 |
| 3.16. | ANNEXE 16 – SERVICES & TARIFS FIXER.IO | 43 |
| 3.17. | ANNEXE 17 – SPÉCIFICATIONS TECHNIQUES | 44 |
| 3.18. | ANNEXE 18 – EXTRAIT DU CODE SOURCE JAVASCRIPT DU CHATBOT S3 | 45 |
| 3.19. | ANNEXE 19 – DESIGN DU “WORDBOT” EN DÉBUT DE PROJET | 46 |
| 3.20. | ANNEXE 20 – LANDING PAGE EN DÉBUT DE PROJET | 47 |

| | | |
|-------|---|----|
| 3.21. | ANNEXE 21 – LANDING PAGE EN FIN DE PROJET | 48 |
| 3.22. | ANNEXE 22 – DESIGN DES 3 CHATBOTS..... | 49 |
| 3.23. | ANNEXE 23 – CLASSIFICATION INTENTIONS/ENTITÉS | 52 |
| 3.24. | ANNEXE 24 – REQUÊTE OPENWEATHER MAP..... | 53 |
| 3.25. | ANNEXE 25 – APPLICATION LUIS.AI WEATHERBOT | 54 |
| 3.26. | ANNEXE 26 – RÉPONSE DU WEATHERBOT | 55 |
| 3.27. | ANNEXE 27 – ENSEMBLE DES MONNAIES INDISPONIBLES | 56 |
| 3.28. | ANNEXE 28 – EXEMPLE DE CONVERSATION (CURRENCYBOT) | 57 |
| 4. | RÉSUMÉS ET MOTS CLÉS | 58 |

TABLE DES ILLUSTRATIONS

| | |
|---|----|
| Figure 1 : Représentation topologique de la glycine | 10 |
| Figure 2 : Création "d'entités doubles" sur Luis.ai | 12 |
| Figure 3 : Technologies de "back-end" envisagées | 13 |
| Figure 4 : Résumé des technologies utilisées | 15 |
| Figure 5 : Résumé des logiciels utilisés..... | 16 |
| Figure 6 : Planning prévisionnel du projet Chatbot 2.0..... | 17 |
| Figure 7 : Liste de tâches prévisionnelles | 18 |
| Figure 8 : Exemple de phrase aléatoire (1)..... | 19 |
| Figure 9 : Exemple de phrase aléatoire (2)..... | 19 |
| Figure 10 : Recherches du mot "Chatbot" dans tous les pays sur GOOGLE depuis avril 2013 - trends.google.ca..... | 23 |
| Figure 11 : Schéma détaillant de manière globale le fonctionnement d'un Chatbot - blog.octo.com | 24 |
| Figure 12 : Offre de stage relatif au développement d'un agent conversationnel au CDS..... | 25 |
| Figure 13 : Étude comparative synthétique réalisée lors du projet tutoré de s3..... | 26 |
| Figure 14 : Domaines d'utilisation du Chatbot S3 relié à DBpedia | 27 |
| Figure 15 : Interface du Chatbot pouvant être sélectionné en Anglais ou en Français | 28 |
| Figure 16 : Interactions du Chatbot S3 entre les différents acteurs technologiques : DBpedia(Cloud), JavaScript(côté client) et RASA(côté serveur) | 29 |
| Figure 17 : Competency Questions relatives au Chatbot. (Vert : Implémentation fonctionnelle / rouge : Implémentation abandonnée ou pas encore réalisée) | 30 |
| Figure 18 : Traductions Chatbot S3 | 31 |
| Figure 19 : Définition Chatbot S3..... | 32 |
| Figure 20 : Synonymes Chatbot S3 | 33 |
| Figure 21 : Cycle des tâches réalisées lors de l'implémentation d'un Chatbot | 34 |
| Figure 22 : Tableau comparatif de différentes caractéristiques de 3 plateformes de NLU en ligne..... | 35 |
| Figure 23 : Interface de création d'une application - Dialogflow | 36 |
| Figure 24 : Interface d'accueil - wit.ai..... | 37 |
| Figure 25 : Interface de création « d'entités » - wit.ai | 38 |
| Figure 26 : Statut de l'application - luis.ai | 39 |
| Figure 27 : Récapitulatif des applications disponibles - luis.ai | 40 |
| Figure 28 : Répartition des phrases d'utilisateur en fonction de leur "intent" - luis.ai..... | 40 |
| Figure 29 : Création "d'Intents" - luis.ai | 41 |
| Figure 30 : Grille de services OpenWeather Map..... | 42 |
| Figure 31 : Grille tarifaire et différents services proposés - fixer.io | 43 |
| Figure 32 : Tableau des spécifications techniques du Lenovo ideapad 100-15IBD - www3.lenovo.com | 44 |
| Figure 33 : Extrait du code JavaScript produit au s3 | 45 |
| Figure 34 : Interface du "Word-Bot" S4..... | 46 |
| Figure 35 : Landing Page réalisée en début de projet | 47 |
| Figure 36 : Landing page en fin de projet..... | 48 |
| Figure 37 : Design du "WordBot" - dorian-naaji.fr/chatbot-2-0..... | 49 |
| Figure 38 : Design du "Weather-Bot" - dorian-naaji.fr/chatbot-2-0 | 50 |
| Figure 39 : Design du "CurrencyBot" - DORIAN-NAAJI.FR/CHATBOT-2-0..... | 51 |
| Figure 40 : Donnée structurée retournée par luis.ai (2)..... | 52 |
| Figure 41 : donnée structurée retournée par luis.ai (1)..... | 52 |
| Figure 42 : Exemple de réponse d'une requête OpenWeather Map au format JSON | 53 |
| Figure 43 : "Entities" du WeatherBot | 54 |
| Figure 44 : "Intents" du WeatherBot..... | 54 |
| Figure 45 : Exemple de réponse du WeatherBot | 55 |
| Figure 46 : Monnaies de base indisponibles pour l'utilisateur | 56 |

CONVENTION TYPOGRAPHIQUE

Cette phrase est dans un format normal.

Cette phrase renvoie à une annexe.

Ce **mot**² est défini dans une note de bas de page

NB : Pour des raisons évidentes de facilité de lecture du présent document, les notes de bas de page se substitueront au lexique de fin.

I. THÈME DU PROJET

1. PRÉAMBULE

Le présent rapport fait part de mon travail réalisé lors du projet tutoré du 4^e semestre de DUT Informatique. Ce projet m'a permis de préciser mon projet personnel et professionnel, ce en poursuivant un travail déjà entrepris lors du projet tutoré du 3^e semestre. Il a également eu pour but de développer des **agents conversationnels**¹ (que l'on appellera Chatbots par la suite) en lien avec une base de **connaissance**² spécifique qui aura été choisie après une phase d'étude et différents comparatifs. Ce rapport se veut d'être compréhensible pour tous, néophytes en informatique y compris, c'est pourquoi une importance toute particulière sera donnée à l'explication du contexte et au thème du projet, avant d'entrer dans les aspects techniques de mes travaux.

2. PRÉSENTATION ET THÈME DU PROJET

2.1. AGENT CONVERSATIONNEL ET INTELLIGENCE ARTIFICIELLE

Le projet que j'entreprends lors du 4^e semestre concerne le domaine de **l'intelligence artificielle**³, par le biais du développement d'un Chatbot. Les Chatbots sont aujourd'hui au centre de l'attention en matière de nouvelles technologies informatiques. Le mot « Chatbot » est en effet devenu un « **buzzword**⁴ », compte tenu des *recherches faites sur Google (annexe 1)*. L'intérêt pour les Chatbots est en plein essor depuis avril 2016, date à laquelle Facebook ouvre une plateforme de développement de Chatbots sur **Messenger**⁵. Aujourd'hui très populaires, les Chatbots se déploient sur toutes les applications de messagerie instantanée et sur certains sites web notamment de **e-commerce**⁶, car très côtés auprès des utilisateurs. Les progrès en intelligence artificielle ces dernières années ont également rendu accessible au grand public la possibilité de développer des Chatbots et de les intégrer sur la plateforme de leur choix. Les Chatbots reposent tous sur un moteur de **traitement automatique du langage naturel**⁷.

Cet intérêt pour les Chatbots est également marqué du côté des grands acteurs du web : « Bots are the new apps. Pretty much everyone today who's building applications, whether they be desktop apps or mobile apps or websites, will build bots as the new interface⁸, » explique Satya Nadella, le directeur général de Microsoft. Mais Microsoft n'est pas la seule société dans cet état d'esprit, Facebook a déjà commencé à faire naître énormément de bots sur Messenger. On pourra également évoquer le Chatbot de Google, « Google Assistant » et « Alexa », le Chatbot d'Amazon. Sans oublier « Cortana », implanté au sein de tous les ordinateurs ayant pour **système d'exploitation**⁹ Windows.

¹ Un agent conversationnel (ou chatbot) est un programme informatique pouvant simuler une conversation par échanges textuels ou vocaux avec un ou plusieurs humain(s).

² Une base de connaissance est un ensemble de données spécifique à un domaine précis, sous une forme exploitable par un ordinateur.

³ L'intelligence artificielle est l'ensemble des techniques permettant de simuler une forme d'intelligence réelle.

⁴ Un buzzword est un terme utilisé pour désigner un produit ou une technologie d'actualité et à la mode.

⁵ Messenger est une application de messagerie instantanée reliée au géant du réseau social, Facebook.

⁶ Site web de vente en ligne.

⁷ Le traitement automatique naturel du langage (TALN ou « NLU » plus communément en anglais, « Natural Language Understanding ») consiste au traitement d'une phrase (vocale ou textuelle) d'un utilisateur visant à transformer celle-ci en donnée structurée et compréhensible par la machine.

⁸ Traduction : Les bots/Chatbots sont les nouvelles applications. Aujourd'hui, quiconque développant une application, qu'elle soit de bureau, mobile ou web, développera un bot/Chatbot en tant que nouvelle interface.

⁹ Un système d'exploitation est « la première application » d'un appareil informatique, installé après construction, qui permet de faire fonctionner et contrôler le dispositif informatique.

Un Chatbot est une application complexe composée de plusieurs éléments. Un *schéma détaillé du fonctionnement classique d'un Chatbot* est disponible en [annexe 2](#). De manière très classique, le fonctionnement d'un Chatbot est le suivant :

- L'utilisateur questionne le Chatbot, vocalement ou textuellement via une interface quelconque.
- Un moteur de traitement du langage naturel (NLU) reçoit le texte brut correspondant à la question de l'utilisateur, l'interprète et renvoie une donnée (la plupart du temps au format **JSON**¹⁰) structurée, sous la forme **intention/entités**¹¹. Pour mieux comprendre la classification intention/entités, se rendre à [l'annexe 23](#).
- Un **moteur conversationnel**¹² **côté serveur**¹³ ou **côté client**¹⁴ traite la donnée retournée par le moteur NLU et construit une réponse en fonction de celle-ci. Le moteur conversationnel est en lien avec une base de connaissance ou tout autre support pour obtenir des informations pour construire la réponse.
- La réponse est retournée à l'utilisateur

2.2. BASE DE CONNAISSANCE

Pour fonctionner de manière optimale, un Chatbot a besoin d'une technologie dite de **back-end**¹⁵. La plupart du temps, une base de connaissance est idéale pour faire office de back-end. Le Chatbot permet ainsi de faire la liaison entre l'utilisateur et la base de connaissance. Les bases de connaissances sont en effet difficiles à appréhender pour un utilisateur non-informaticien. Réaliser un Chatbot relié à une base de connaissance permet donc de rendre accessible l'expertise d'un domaine à tous. Prenons l'exemple d'une base de connaissance recensant toutes les protéines et acides aminés et leurs caractéristiques. Relier un Chatbot à une telle base de connaissance permettra de rendre accessible énormément d'informations à un utilisateur, sans réaliser le moindre effort en termes d'informatique, car il suffira de simplement questionner le Chatbot : « Quelle est la structure chimique de la protéine Glycine ? », et on pourra imaginer la réponse du Chatbot sous la forme d'une **formule topologique**¹⁶ de la molécule en question :

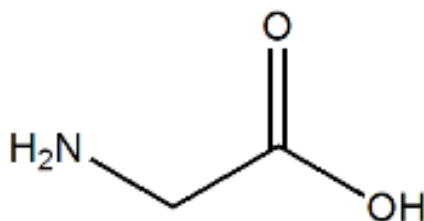


FIGURE 1 : REPRESENTATION
TOPOLOGIQUE DE LA GLYCINE

Ce genre de problématique (mettre à disposition à un utilisateur quelconque une base de connaissance) est un des intérêts actuels du développement d'agents conversationnels. On pourra par exemple citer *une offre*

¹⁰ Le JSON est un format de données textuelles.

¹¹ La structuration intention/entités (plus communément appelée « intent/entities structuration ») est un formalisme pour les données retournées par la plupart des moteurs de NLU.

¹² Un moteur conversationnel est un ensemble de tâches effectué dans un langage donné permettant de traiter des données structurées émanant d'un moteur de NLU.

¹³ Les traitements sont effectués sur un serveur (un ordinateur) distant.

¹⁴ Les traitements sont effectués côté client, par l'application (navigateur internet ou application bureau).

¹⁵ Arrière-plan. Une technologie de « back-end » est dans une application une technologie effectuant des traitements et des calculs sans que l'utilisateur ne s'en rende compte.

¹⁶ Une formule topologique est une représentation moléculaire.

de stage concernant le requêtage en langage naturel ([annexe 3](#)) du Centre de Données astronomiques de Strasbourg.

2.3. EXISTANT : PROJET TUTEUR DU SEMESTRE 3

Lors du projet tutoré du 3^e semestre, le but était de relier un moteur de NLU à **DBpedia**¹⁷. En tant que moteur de NLU, nous avons choisi **RASA** après une *étude comparative* ([annexe 4](#)), car il fonctionnait entièrement en local côté serveur et était **open source**¹⁸. Il permettait donc de faire fonctionner notre Chatbot indépendamment d'un service tiers. En reliant RASA à DBpedia, le but était de pouvoir demander au Chatbot *des définitions, des traductions et des synonymes de mots spécifiques* ([annexe 5](#)), dans une *interface* proposée à la fois en anglais et en français ([annexe 6](#)). Un schéma décrivant le fonctionnement de l'application est disponible en [annexe 7](#). À mi-chemin de la réalisation de ce projet, j'ai mis en place avec un autre membre du groupe une fiche de « **Competency questions** »¹⁹ ([annexe 8](#)) qui a permis de décrire l'état final du Chatbot. Une démonstration du Chatbot est disponible en [annexe 9](#).

Mon rôle lors de ce projet a été d'améliorer l'interface utilisateur, de gérer le fonctionnement côté serveur de RASA et de le relier à DBpedia via **JavaScript**²⁰. En d'autres termes, je m'occupais du développement en **HTML**²¹-**CSS**²²-JavaScript ainsi qu'à la *logique d'interaction du Chatbot* (citée précédemment en [annexe 7](#)). Du côté de DBpedia, il m'a fallu m'intéresser au langage de requête **SPARQL**²³ conjointement avec un autre membre du groupe responsable de la mise en forme des requêtes pour le Chatbot. J'ai eu l'occasion d'expérimenter l'utilisation de la bibliothèque **d3.js**²⁴, permettant de générer des affichages visuels (images) à partir de données brutes.

3. RÉFLEXIONS SUR LE TRAVAIL À EFFECTUER

3.1. DEFINITION DE LA MISSION DU PROJET

Le projet Chatbot 2.0 a consisté en l'étude de bases de connaissances / technologies de « back-end » pouvant être intégrées à un moteur de NLU. Dans l'absolu, le but était de trouver des bases de connaissances autres que DBpedia, et de vérifier leur pertinence ainsi que leur fonctionnement pour une éventuelle implémentation. Le projet s'est ainsi déroulé en *5 grandes phrases d'étude et de développement* ([annexe 10](#)) pour chaque base de connaissance exploitable trouvée. Le projet a donc été décomposé en cycles itératifs, nous y reviendrons par la suite ([2.2.](#)). De manière plus concrète, le projet Chatbot 2.0 a consisté au développement d'au minimums 2 nouveaux Chatbots en ligne, avec lesquels il est possible pour l'utilisateur de discuter et de poser des questions sur un thème donné, relatif à la base de connaissance utilisée. Lorsque, pendant le projet initial, nous avons développé un seul Chatbot durant le 3^e semestre, je me suis proposé d'en développer 2 à 3 durant le 4^e semestre de mon DUT. Également, j'ai imposé différentes règles en définissant la mission de mon projet :

¹⁷ DBpedia est une base de connaissance dont les données proviennent de Wikipédia.

¹⁸ Une application « open source » est une application dont le code source est visible par tous.

¹⁹ Un ensemble de questions qui permettent d'en savoir plus sur ce dont quoi est capable une entité (personne, objet, technologie...). Exemple : Les téléphones répondent tous « Oui » à la competency question « L'objet est-il capable de permettre la communication à distance ? ».

²⁰ Langage de programmation compilé par un navigateur internet.

²¹ HTML est un langage à balise utilisé pour produire des pages web interprétables par les navigateurs internet.

²² CSS est un langage permettant de gérer le positionnement et le style des pages HTML.

²³ SPARQL est un langage de requête permettant d'accéder aux données de DBpedia.

²⁴ Bibliothèque JavaScript permettant de générer différents visuels (graphiques, diagrammes, etc.)

- Un effort sera fait sur la présentation des réponses à l'utilisateur et à l'interface de manière à rendre l'interaction homme-machine très agréable et « user-friendly »
- Les différents agents conversationnels développés devront être uniquement fonctionnels en anglais pour une meilleure universalité.
- Aucun serveur dédié ne sera utilisé/loué durant ce projet pour des raisons financières évidentes, l'entière de l'application devra alors pouvoir fonctionner uniquement depuis un navigateur web grâce à l'accès à des technologies « **cloud**²⁵ ».

Finalement, le développement de 2 nouveaux Chatbots a été pensé lors de la phase d'études de différentes bases de connaissances : Un « WeatherBot », un Chatbot pouvant donner des informations sur la météo (Exemple de question d'utilisateur : « What's the weather like in Beaune ? ») et « GeoBot », un Chatbot en lien avec DBpedia et pouvant retourner des informations de densité et de population sur des territoires (villes, pays, continents). En milieu de projet et comme expliqué par la suite ([I – 3.3.2.](#)), mon choix a été de réaliser un « CurrencyBot » à la place du « GeoBot ». Le « CurrencyBot » est capable de convertir des sommes d'argent d'une monnaie à une autre. Le développement de ceux deux Chatbots, « WeatherBot » et « CurrencyBot » ainsi que différentes tâches annexes citées par la suite étaient les missions principales de mon projet.

3.2. ÉTUDE COMPARATIVE DE MOTEURS DE TRAITEMENT DU LANGAGE NATUREL

Dans un premier temps, pour qu'un agent conversationnel puisse fonctionner, il a besoin d'un moteur de traitement du langage naturel. Durant le projet au 3^e semestre, RASA a été choisi mais sa portabilité est très restreinte. Ce choix avait été fait afin de se familiariser un moteur NLU complexe et entièrement configurable. Pour le projet Chatbot 2.0, il était inenvisageable de réutiliser ce moteur, c'est pourquoi j'ai étudié différents moteurs de NLU disponibles en « **web-service**²⁶ » et ai réalisé une *étude comparative* ([annexe 11](#)) pour me guider dans mon choix, entre [luis.ai](#), [dialogflow](#) et [wit.ai](#)²⁷. Au terme de mon étude, mon choix s'est orienté vers [luis.ai](#), développé par Microsoft et utilisé par Cortana, l'agent conversationnel de Windows.

Mon choix s'était initialement orienté vers DialogFlow, qui proposait également les mêmes services que [luis.ai](#). L'*interface* ([annexe 12](#)) était très attractive et rendait agréable le développement des intent/entities mais le système de requêtage était difficile à mettre en place, car pas assez documenté. Quant à [wit.ai](#), j'ai tout de suite écarté ce moteur de NLU à cause de son *interface* ([annexe 13](#)) étrange et repoussante pour l'utilisateur, il était difficile de cerner le fonctionnement du service à cause de ce facteur, bien que celui-ci semble très performant.

[Luis.ai](#) est simple d'utilisation et permet la création de plusieurs applications, ce qui permet de rendre le moteur de NLU plus précis et de créer une application par agent conversationnel. La création d'une application est simple et rapide. L'*interface* ([annexe 14](#)) est agréable et permet de vite s'y retrouver dans la création de l'application et va à l'essentiel. Je l'ai donc choisie pour sa simplicité, tant pour la mise en place que pour le fonctionnement. Néanmoins, payant au-delà de 10 000 requêtes par mois, l'utilisation que j'allais faire de [luis.ai](#)



FIGURE 2 : CREATION "D'ENTITES DOUBLES" SUR LUIS.AI

²⁵ Accès à la puissance de stockage et de calcul de serveurs distants depuis Internet.

²⁶ Service entièrement utilisable en ligne.

²⁷ Des applications de NLU en ligne.

était convenable pour ne pas dépasser ce quota. Également, une raison de ma préférence pour *luis.ai* est le fait de pouvoir créer des entités contenant plusieurs mots de manière très simple. Par exemple, une entité « ville » va contenir des noms de villes. Avec *dialogflow*, il est compliqué de créer une ville sur deux mots, comme « Los Angeles ».

Dans l'absolu, tous ces services de nlu se valent et ont sensiblement les mêmes performances. Le reste est une question de goût et d'appréhension du service.

NB : Un détail technique de *luis.ai* est disponible dans la partie [II - 1.1.](#)

3.3. ÉTUDE DE BASES DE CONNAISSANCES

3.3.1. BASES DE CONNAISSANCES ENVISAGÉES

Un agent conversationnel a besoin d'une source de données pour pouvoir répondre de manière optimale à un utilisateur. C'est pourquoi les Chatbots sont souvent reliés à une base de connaissance en lien avec le domaine du Chatbot. Par exemple, Un Chatbot pouvant donner le prix de tous les articles présents sur Amazon devra être relié à la base de connaissance contenant l'ensemble des produits d'**Amazon**²⁸. Ainsi, le moteur conversationnel pourra requêter la base de connaissance sur un élément spécifique en fonction des données reçues par le moteur NLU. Lors de mon travail, je me suis orienté vers l'étude du fonctionnement de plusieurs bases de connaissances, dont 3 que j'ai retenues :

| DBPedia | OpenWeatherMap | Fixer.io |
|--|--|---|
| <ul style="list-style-type: none">• Utilisé comme base de connaissance du chatbot initial• Très complet | <ul style="list-style-type: none">• Recense des données météorologiques• Énormément de secteurs disponibles | <ul style="list-style-type: none">• Données sur les taux de changes• Monnaies nationales / supranationales du monde entier |

FIGURE 3 : TECHNOLOGIES DE "BACK-END" ENVISAGÉES

NB : l'ensemble des technologies citées ci-après sont détaillées plus en profondeur et sous un aspect plus technique dans la partie [II - 1.1.](#)

3.3.2. DBPEDIA

Pour vulgariser, DBpedia recense toutes les informations de **Wikipédia**²⁹ sous la forme d'une base de connaissances, afin de permettre à des applications informatiques d'aller y récupérer des informations.

²⁸ Site de vente en ligne.

²⁹ Encyclopédie en ligne.

Wikipédia est accessible et lisible par un être humain tandis que DBpedia est accessible par une machine. Mais DBpedia est une base de connaissance très difficile à exploiter pour un utilisateur quelconque, car compliquée à appréhender via un navigateur web et utilisant le langage SPARQL pour l'accès aux données. C'est pourquoi le projet du S3 a eu pour but de mettre à disposition la connaissance de DBpedia à n'importe quel utilisateur, mais ce projet aurait pu être largement prolongé. DBpedia groupe énormément d'informations sur énormément d'entités, il m'est donc venu à l'esprit lors de mes études de différentes bases de connaissances de réexploiter DBpedia une seconde fois de manière plus approfondie. Il s'avérera que lors du développement, j'ai pensé que réaliser une seconde fois un agent conversationnel en lien avec DBpedia serait redondant avec mon projet tutoré de S3 et même si l'agent conversationnel aurait pu être totalement différent de celui de S3, le fonctionnement interne aurait été sensiblement le même à 60%.

3.3.3. OPENWEATHER MAP

OpenWeather Map est un service en ligne offrant un service de requête permettant d'avoir accès aux données météorologiques d'un emplacement donné (latitude/longitude, nom de ville, identifiant de ville...). La compagnie propose *différents services* ([annexe 15](#)) dont certains sont accessibles gratuitement. J'ai donc pu utiliser le service « Current Weather » qui permettait d'avoir accès aux données météorologiques d'un lieu donné au moment où l'utilisateur effectue la demande.

3.3.4. FIXER.IO

Fixer.io est un service en ligne permettant d'obtenir des taux de changes de différentes monnaies du monde entier. Malheureusement, ce service et tous les services similaires proposent des *fonctionnalités* ([annexe 16](#): grille des différents services et abonnements) très restreintes lorsqu'utilisés gratuitement.

II. TRAVAIL EFFECTUÉ

1. MATÉRIEL, TECHNOLOGIES ET LOGICIELS UTILISÉS

1.1. MATERIEL UTILISÉ

L'ensemble du projet a été réalisé sur un ordinateur portable de marque Lenovo, référence « ideapad-100-15IBD ». Un descriptif technique de l'ordinateur est disponible en [annexe 17](#).

1.2. TECHNOLOGIES UTILISÉES

Luis.ai

- Moteur de traitement du langage naturel pour les agents conversationnels développés
- Choix argumenté (I - 3.2)

Javascript

- Langage de programmation exécuté côté client permettant de lier luis.ai à la base de connaissance de l'agent conversationnel concerné
- Indispensable pour une application web avancée exécutée côté client, choix par défaut

HTML 5

- Syntaxe et contenu de la page web du site développé
- Choix par défaut (norme pour les navigateurs web)

CSS 3

- Disposition et arrangement du contenu de la page web du site développé
- Choix par défaut (norme pour les navigateurs web)

DBPedia

- Base de connaissance riche
- A été utilisé seulement pour réintégrer le chatbot réalisé au S3 au site web développé durant le projet Chatbott 2.0

OpenWeather Map

- Base de connaissance météorologique la plus populaire
- Gratuite pour une utilisation de base
- Densité d'informations
- Beaucoup de lieux et villes disponibles

Fixer.io

- Similaire que les bases de connaissances monétaires concurrentes
- Choix pour sa simplicité d'utilisation.

FIGURE 4 : RÉSUMÉ DES TECHNOLOGIES UTILISÉES

1.3. LOGICIELS UTILISÉS



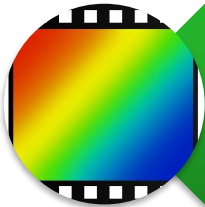
WebStorm est un environnement de développement web très performant et facilitant grandement le travail du développeur : auto-complétion, gestionnaire de fichiers intégré, coloration syntaxique, etc. C'est pour ces raisons que j'ai entièrement développé sous WebStorm.



FileZilla est un logiciel permettant de déposer des fichiers sur un serveur distant. Il m'a notamment été utile pour déployer mon projet en ligne chez mon hébergeur web.



Sublime Text 3 est un logiciel à mi-chemin entre l'environnement de développement et l'éditeur de texte. Disposant d'une coloration syntaxique et d'une interface chaleureuse, il m'a été utilisé pour réaliser des petites corrections sur des fichiers textes (JavaScript, HTML ou CSS)



PhotoFiltre est un logiciel de retouche d'image, il m'a été utile pour redimensionner certaines images utilisées par le site web.



Google Chrome est un navigateur Internet, il m'a servi à effectuer différentes recherches et prévisualiser les résultats de mon projet au cours de son avancement



Excel est un tableur, il m'a servi notamment lors de mes études comparatives à réaliser des tableaux ou lors de ma planification.



Word est un logiciel bureautique de traitement de texte. Il m'a été utile lors de la rédaction du présent rapport.

FIGURE 5 : RÉSUMÉ DES LOGICIELS UTILISÉS

2. ORGANISATION

2.1. METHODE DE GESTION DE PROJET

La gestion de projet en informatique est divisée en deux grandes familles.

Premièrement, la gestion de projet classique et prédictive : tout est prévu depuis le début. Un cahier des charges est rédigé, contenant l'ensemble des besoins auxquels devra répondre l'application. S'en suivent une analyse et un développement informatique, conformément au cahier des charges. Cela pose un cadre strict et rigide et un formalisme lourd. Un cahier des charges fige les fonctionnalités à développer. Cela demande dans un contexte professionnel de réaliser des avenants au contrat pour pouvoir modifier le cahier des charges, ce qui peut parfois entraîner d'interminables renégociations.

Deuxièmement, la gestion de projet via des méthodes dites agiles. Le formalisme y est moins formel et différent, les spécifications peuvent changer. Le projet n'est pas figé dès la rédaction du cahier des charges, et permet ainsi de réagir facilement face aux changements. En milieu professionnel, la gestion de projet agile permet une meilleure collaboration avec le client et impose des réunions courtes et fréquentes.

Étant donné la courte durée du 4^e semestre, mon choix s'est porté vers les méthodes agiles, dont une qui m'a particulièrement séduit : la méthode « Scrum ». Cette méthode repose sur un modèle itératif et incrémental : on définit un temps de cycle (entre 7 à 30 jours) durant lequel on va développer un livrable fonctionnel du projet final. Au terme de chaque cycle, le livrable fonctionnel est incorporé au précédent livrable fonctionnel développé ; on progresse « morceau par morceau ». Lors du lancement du projet, on réalise un backlog produit, un document qui regroupe la liste des fonctions à développer dans le produit fini, qui est similaire à la « Competency Question », qui est l'état final du backlog produit.

2.2. SUIVI DU PROJET

Conformément à la méthode Scrum j'ai utilisé des formalismes légers afin de suivre l'avancement de mon projet : planning prévisionnel théorique, « **todo list** ³⁰ prévisionnelle ».

| Jours Mois | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | | |
|--------------------------------------|---|---|---|---|---|-----------------|---|---|---|----|----|----|----|----|----|--|----|----|----|-------------------|----|----|----|----|----|----|----|-------------------------|----|----|----|--|--|--|
| Février | Etudes d'API de NLU, de bases de connaissance exploitables et de faisabilité (les technologies sont-elles exploitables ?) | | | | | | | | | | | | | | | Conception Objet & Conception de l'interface | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | Competency Question | | | | | | | | | | | | | | | | | | |
| Mars | Rédaction du rapport | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | Refactoring | | | | | Dév. WeatherBot | | | | | | | | | | Dév. CurrencyBot | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Dév. Luis.ai | | | | | | | | | | | | | | | | | | | | Amélio. NLU | | | | | | | | | | | | | | |
| Développement front-end et interface | | | | | | | | | | | | | | | | | | | | Mise en place SPA | | | | | | | | Améliorations front-end | | | | | | |

FIGURE 6 : PLANNING PREVISIONNEL DU PROJET CHATBOT 2.0

³⁰ Ensemble de tâches à réaliser



FIGURE 7 : LISTE DE TÂCHES PRÉVISIONNELLES

3. REFACTORISATION ET AMÉLIORATION DU CODE SOURCE

La première tâche du projet a été la **refactorisation**³¹ du code dans son entièreté (HTML, CSS et JavaScript compris). En effet, lors du projet tutoré du 3^e semestre, je n'avais encore jamais expérimenté le développement en JavaScript. C'était pour moi une totale nouveauté. En tant que débutant, je n'étais pas conscient qu'il était possible de développer sous JavaScript en **orienté objet**³² j'ai alors développé l'application dans son entièreté dans une seule **classe**³³, c'est-à-dire dans un seul fichier JavaScript, ce qui est très mauvais. Le *fichier en question (annexe 18)* était alors long de 917 lignes. Il y a donc eu une conception objet, mais toute la logique de raisonnement du Chatbot se faisait dans le même fichier, ce qui n'avait pas de sens. Ces 917 lignes de codes ont été refactorisées en 10 fichiers soit 10 classes d'objets. S'ajoute à cela les deux autres Chatbots ayant été développés, ce qui fait que l'application dans sa totalité est divisée en 23 fichiers, soit 23 classes : cela rend le code beaucoup plus facile à maintenir, à débbugger et le rend beaucoup plus lisible. Les fichiers en question sont visionnables sur mon profil GitHub :

- [Lien vers l'ancien fichier JavaScript \(917 lignes\)](#)
- [Lien vers le dossier contenant les nouvelles classes](#)

Quant aux améliorations du code, il a été question d'utiliser la bibliothèque jQuery, permettant de manipuler des éléments HTML d'une page web et de manipuler des requêtes entre le client et le serveur.

4. AMÉLIORATIONS DE L'INTERFACE UTILISATEUR

Tout au long du projet, l'interface a subi des améliorations. Au début du projet, le site web était *uniquement composé du Chatbot du S3 amélioré (annexe 19)*. Puis, étant donné qu'il était par la suite du projet planifié que deux autres Chatbots allaient être développés, une « **landing-page**³⁴ » (*annexe 20*) a été réalisée afin de donner le choix à l'utilisateur de choisir le Chatbot avec lequel il souhaitait converser. Mais étant donné la lourdeur des chargements entre les Chatbots, cela n'était pas du tout agréable pour l'utilisateur. La mise en place d'un « **Single Page Application**³⁵ » a alors été élaborée, entièrement fonctionnelle grâce au HTML et CSS ; des transitions CSS sont utilisées pour naviguer entre les Chatbots et ceux-ci sont accessibles depuis la même page. Il suffit ensuite de cliquer sur un bouton en haut à gauche de la page pour retourner sur la nouvelle *landing page (annexe 21)*. Des animations ont été ajoutées sur les liens hypertextes de manière à rendre l'interface plus agréable pour l'utilisateur ([accès au site](#)). De plus, un soin particulier au design a été réalisé pour être en accord avec le thème des Chatbots. Le *design final des 3 Chatbots, « WordBot », « WeatherBot » et « CurrencyBot » est disponible en annexe 22*. Également, les réponses du Chatbot ont été pensées afin d'être le plus attractives possibles pour l'utilisateur. Notamment, l'ajout d'emojis et de remarques a été ajouté pour créer une certaine empathie avec le Chatbot.

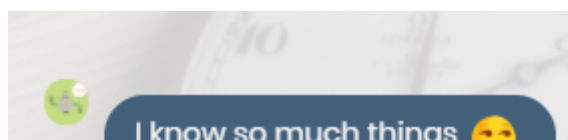


FIGURE 9 : EXEMPLE DE PHRASE ALEATOIRE (2)

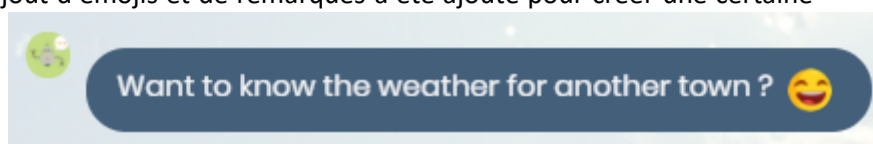


FIGURE 8 : EXEMPLE DE PHRASE ALEATOIRE (1)

³¹ La refactorisation de code permet d'améliorer grandement sa lisibilité sans changer son fonctionnement.

³² La programmation orientée objet est un paradigme de programmation qui vise à représenter une entité, du monde physique ou de l'application par une classe.

³³ Dans un programme informatique, une classe est un ensemble de fonctionnalités relatives à un objet du programme développé.

³⁴ Page d'accueil

³⁵ Application web où tout peut se réaliser depuis une seule page : une seule page est chargée et il n'y a pas de chargements lors de l'accès à d'autres catégories du site

5. AMÉLIORATION DU CHATBOT INITIAL : « WORDBOT »

Les améliorations sur le WordBot portaient essentiellement sur la manière dont le moteur NLU lui.ai appréhendait les questions. Parfois, les questions étaient mal interprétées et classifiées dans les mauvaises « intents », c'est pourquoi il a fallu entraîner le Chatbot en ajoutant des exemples de phrases types qu'un utilisateur pourrait envoyer au Chatbot. L'application lui.ai permet cela, et entraîne ainsi la logique du Chatbot.

6. DÉVELOPPEMENT D'UN SECOND CHATBOT : « WEATHERBOT »

Le développement du WeatherBot a consisté en l'intégration de la base de connaissance OpenWeather map au moteur conversationnel programmé en JavaScript. Dans un premier temps, il a fallu appréhender les *données retournées lors d'une requête à la base de connaissance* ([annexe 24](#)). Ensuite, il a fallu créer l'application lui.ai en fonction des différentes requêtes possibles d'un utilisateur ([annexe 25](#)). Puis, il a fallu coder l'ensemble des classes JavaScript nécessaires à l'interaction avec la base de connaissance et l'application lui.ai (pour rappel, le code est disponible [ici](#)). À son stade final, le Chatbot était capable de restituer la météo actuelle d'une ville donnée ([annexe 26](#)). Malheureusement, tout le code nécessaire avait été réalisé pour répondre à un utilisateur sur des prévisions météo, mais le « 16 day forecast » (prévision météo sur 16 jours) d'OpenWeather Map, formule la plus intéressante pour les prévisions météorologiques est payante. Il restait alors l'option du « 5 day forecast » mais était trop contraignante pour réaliser des prévisions météorologiques dans le temps imparti ([annexe 15](#)). La prévision météo du « WeatherBot » reste néanmoins la principale amélioration future. La forme choisie pour restituer les réponses à l'utilisateur a été le tableau stylisé avec « hover³⁶ ». Avec plus de temps, il aurait été également intéressant de créer des courbes de températures ou autres graphiques à l'aide de la bibliothèque d3.js.

7. DÉVELOPPEMENT D'UN TROISIÈME CHATBOT : « CURRENCYBOT »

Initialement, le « CurrencyBot » devait être le « GeoBot », un Chatbot capable de restituer des données démographiques et de superficie sur des territoires donnés, en lien avec DBpedia. Mais comme expliqué précédemment ([I-3.3.2](#)), cela aurait été redondant avec le projet de S3 et il était préférable de s'approprier une autre base de connaissances. J'ai alors choisi de réaliser un Chatbot capable de convertir des sommes d'argent d'une monnaie à une autre. Pour ce Chatbot, les différences entre toutes les phrases des utilisateurs étaient tellement grandes et le nombre d'entités si grand qu'utiliser une application de NLU aurait été très complexe et aurait mal fonctionné. J'ai donc développé une interface qui permet à l'utilisateur d'entrer des valeurs et de choisir les conversions qu'il souhaite réaliser, afin de restreindre les saisies de l'utilisateur qui auraient été trompeuses pour le Chatbot. De plus, cela facilite la saisie pour l'utilisateur. Pour des raisons financières, seulement la conversion de sommes d'argent d'Euro à toutes les autres monnaies est possible. En effet, il est impossible d'obtenir les taux de changes de la base de connaissance depuis une autre monnaie sans *souscrire à un abonnement* ([annexe 16](#)). Néanmoins, l'application est prête à recevoir les taux de changes en cas d'un abonnement à une formule payante de la base de connaissance ([annexe 27](#)), toutes les monnaies ont été ajoutées et ont été grisées pour signaler à l'utilisateur qu'il est impossible de changer de monnaie de base de conversion. Des *exemples de conversation avec le CurrencyBot* sont disponibles en [annexe 28](#).

³⁶ Propriété CSS permettant de changer le comportement d'un élément lors d'un pointage de souris.

8. CONCLUSION

8.1. LIVRABLE FONCTIONNEL

L'ensemble de mon travail est testable sur mon site web à l'adresse dorian-naaji.fr/chatbot-2-0/.

L'ensemble du code est éventuellement visionnable sur un [dépôt GitHub](#) créé à cet effet.

8.2. LEÇONS TIREES DE CE TRAVAIL

Travailler sur des agents conversationnels une seconde fois m'a permis de perfectionner mes compétences dans ce domaine et sera un atout de taille pour ma vie professionnelle future, étant donné l'intérêt porté par les entreprises aux Chatbots de nos jours. Ce projet est perfectible et je l'améliorerai durant mon temps libre afin de pouvoir présenter ce travail, de la manière la plus aboutie possible, lors notamment d'entretiens. Ce fut une expérience riche et qui vaudra la peine d'être intégrée à mon CV. Pouvoir prolonger le projet de S3 qui m'a passionné est une réelle chance.

III. DOCUMENTS ANNEXES

1. LIENS

<http://www.openweathermap.com/api>, site de la base de connaissance OpenWeatherMap

<http://learn.jquery.com/>, site contenant des informations et de la documentation sur jQuery

<https://stackoverflow.com/>, forum de discussion permettant d'échanger sur des problèmes relatifs à la programmation et à l'informatique.

2. ANNEXES

3.1. ANNEXE 1 – CHATBOT GOOGLE TRENDS



FIGURE 10 : RECHERCHES DU MOT "CHATBOT" DANS TOUS LES PAYS SUR GOOGLE DEPUIS AVRIL 2013 - TRENDS.GOOGLE.CA

3.2. ANNEXE 2 – FONCTIONNEMENT GLOBAL D'UN CHATBOT

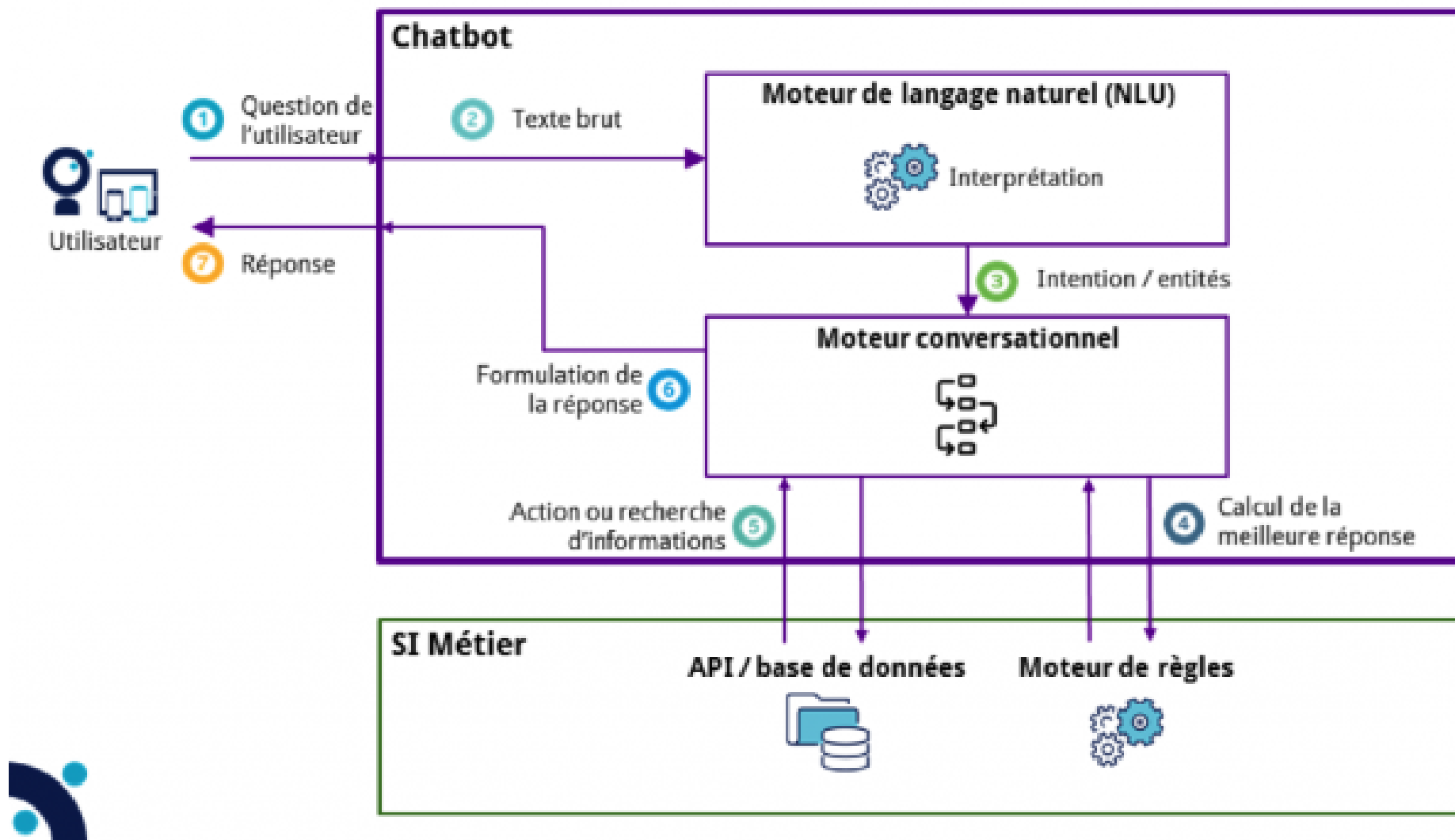


FIGURE 11 : SCHEMA DETAILLANT DE MANIERE GLOBALE LE FONCTIONNEMENT D'UN CHATBOT - BLOG.OCTO.COM



Centre de Données astronomiques de Strasbourg (CDS)

Observatoire astronomique de Strasbourg

11 rue de l'Université 67000 Strasbourg

Stages 2018

Sujet 1

A noter : nous proposerons *plusieurs sujets* au fur et à mesure de nos réflexions et vous pouvez évidemment postuler à plusieurs d'entre eux, suivant vos centres d'intérêt.

Pour l'ensemble des sujets proposés : gratification mensuelle nette de 568,76 euros, plus possibilité de prise en charge partielle des frais de transport (abonnements tram, train, etc.).

Contact: andre.schaaff@astro.unistra.fr (en précisant le(s) sujet(s) qui vous intéresse(nt))

Contexte: Ces dernières années de nombreux étudiants de DUT ont réalisé leur stage au CDS (<http://cdsweb.unistra.fr/>) en abordant des sujets variés (développements mobiles, visualisations en OpenGL/WebGL, HTML5/CSS3/JavaScript/JQuery, Cloud, réalité virtuelle, bases de données distribuées, etc.) Le CDS est une équipe d'une trentaine de personnes (astronomes, informaticiens, documentalistes) au sein de l'Observatoire de Strasbourg (env. 80 personnes). Le CDS héberge plusieurs services de données astronomiques de renommée mondiale. Il est par ailleurs un acteur majeur de l'IVOA (International Virtual Observatory Alliance, <http://www.ivoa.net>, également appelé Observatoire Virtuel ou OV) qui définit des standards et développe des prototypes pour l'interopérabilité des données et des services astronomiques.

Sujet de stage 1 : " Interrogation de services de données astronomiques en Langage Naturel"

Les services du CDS sont actuellement essentiellement interrogeables au travers de "formulaires" classiques (exemple : pour rechercher des catalogues astronomiques, on donnera par exemple un nom d'objet astronomique ou une position dans le ciel, un rayon, ..., des paramètres pour formater le résultat (par exemple les 50, 100, ... premiers ou illimité, le format de la sortie (HTML, CSV, XML, etc.)).

Nous prototypons actuellement un outil permettant à un utilisateur de réaliser une requête en langage naturel (exemples : « **What is the effective temperature of Sirius ?** », « **What is the redshift of galaxies members of the Virgo cluster ?** »). Cela peut concerner un seul service ou plusieurs.

Nous avons des outils permettant de reconnaître par exemple des noms d'objets astronomiques dans un texte. Nous avons également exploré les outils proposés par Stanford (autour du « Natural Language Processing »). Nous avons aussi défini une première liste d'exemples de requêtes.

Nous avons déjà investigué le champ de la réalité virtuelle et ce type d'interaction pourrait être également très utile dans un domaine où l'interface utilisateur est extrêmement réduite. On peut alors envisager d'y associer une reconnaissance vocale.

Il y a encore beaucoup de choses à faire et à améliorer avant une mise en ligne. Notamment dans le cas des requêtes difficiles à interpréter, car soit ambiguës soit mal formulées (pour diverses raisons, mauvaise connaissance du domaine, première utilisation). Il n'est pas possible de laisser un utilisateur se débrouiller seul sans lui apporter une aide (en plus de la documentation en ligne habituelle) durant la saisie afin de réduire les risques d'échec et lui permettre également de comprendre pourquoi sa requête n'aboutit pas au résultat espéré. Cette aide peut intervenir et se concrétiser à divers niveaux, durant la saisie (complétion / propositions par exemple), après la saisie par reconstruction (correction de fautes d'orthographe, mots manquants, unité des valeurs numériques, etc.) de la requête.

Connaissances souhaitées : Goût pour les aspects R&D, rigueur. Un langage de programmation (à priori parmi Java, Python et/ou JavaScript, ...) suivant les technologies utilisées.

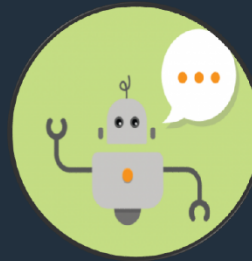
FIGURE 12 : OFFRE DE STAGE RELATIF AU DEVELOPPEMENT D'UN AGENT CONVERSATIONNEL AU CDS

3.4. ANNEXE 4 – ÉTUDE COMPARATIVE REALISEE LORS DU PROJET DE S3 DE DIFFERENTS MOTEURS DE NLU

| Nom de l'API | License | Language de l'API | Langue doc/code | Documentation | Facilité de déploiement | Fonctionnalités nombreuses | Domaine/clients | "Canaux" |
|---|--|--|-----------------|----------------------------|-------------------------|----------------------------|---|--|
| Rasa.ai | Open source | Python | Anglais | Très bonne | Difficile | Oui | Assurance Banque Santé Télécommunications Voyage | localement |
| ChatScript | MIT license | C,C++ | Anglais | Bonne | Oui | Moyen | ChatScript est la base de l'entreprise linguistique naturelle pour une variété de startups technologiques | - |
| MS Bot Framework | Open source et disponible pour tous sur github | C#, Javascript | Anglais | Très bonne | Oui | Moyen | Permet de fabriquer et déployer de bots de hautes qualités | À partir de sites web ou d'applications pour écrire des sms, de Skype, de Facebook Messenger et d'autres services populaires |
| Botkit | Open source | Javascript | Français | Correct | Oui | Non | Botkit permet la création d'application et de ChatBot ainsi que de leur intégration sur les principales plates-formes de messagerie | Slack, Cisco Spark, Microsoft Teams, Facebook Messenger, Twilio SMS, Twilio IPM, Skyoe, Group.me, Telegram, Kik, email |
| IBM Watson Conversation Service | Une version propose la gratuité de l'API, et il y a des offres standart et premium qui elles sont payantes | Node SDK Java SDK Python SDK iOS SDK Unity SDK | Anglais | Très bonne | Oui | Moyen | Elle est utile dans les domaines de la sécurité social, dans la finance et dans le commenrce. | Fonctionne à partir d'image, de texte et de paroles articulées |
| Pandorabots | License payante après un essai de 10 jours | SDKs: Java Node.js Python Ruby PHP Go | Français | Correct | Oui | Moyen | Convient pour les cas basiques, la publicité, enseignement en ligne, universités Pour des discussions textuelles ou vocales | Apparait dans les discussions, sur les applications natives, sur internet, dans les jeu, les réseaux sociaux |

FIGURE 13 : ÉTUDE COMPARATIVE SYNTHETIQUE REALISEE LORS DU PROJET TUTEORE DE S3

3.5. ANNEXE 5 – CHAMP D'UTILISATION DU CHATBOT



Projet Chatbot

Agent conversationnel

Hey chatbot ! Tu pourrais me donner toutes les traductions d'automobile que tu connais, s'il te plaît ?!

| TRADUCTION | LANGUE |
|---------------------------|-------------|
| Automobile | Français |
| Gépjármű | Hongrois |
| Kendaraan bermotor | Indonésien |
| Kraftfahrzeug | Allemand |
| Motor vehicle | Anglais |
| Motorlu taşıt | Turc |
| Motorové vozidlo | Tchèque |
| Motorvoertuig | Néerlandais |
| Pojazd mechaniczny | Polonais |
| Моторно превозно средство | Bulgare |
| モータービークル | Japonais |

- » Demandez lui une définition
- » Demandez lui une traduction
- » Demandez lui des synonymes

Il vous répondra

Le chatbot est le premier pas vers une IA autonome



Dorian NAAJI - Alexis GUYOT - Lélian RUFFIN - Bastien GRENIER
Raphael CORDIEROT - Valentin NOËL - Emmanuel DEREPAS



FIGURE 14 : DOMAINES D'UTILISATION DU CHATBOT S3 RELIE A DBPEDIA

3.6. ANNEXE 6 – INTERFACE DU CHATBOT S3

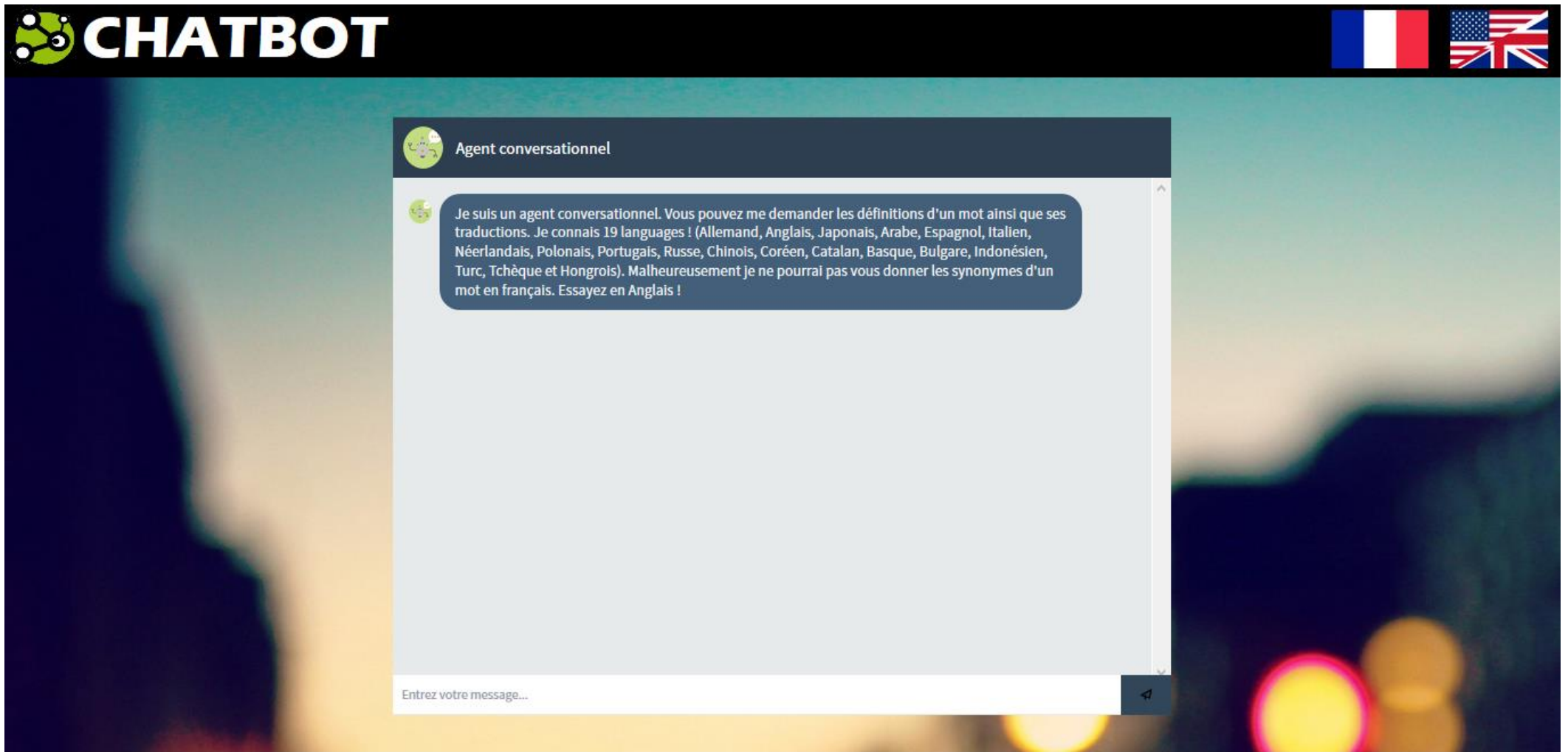


FIGURE 15 : INTERFACE DU CHATBOT POUVANT ETRE SELECTIONNE EN ANGLAIS OU EN FRANÇAIS

3.7. ANNEXE 7 – LOGIQUE D'INTERACTION DU CHATBOT S3

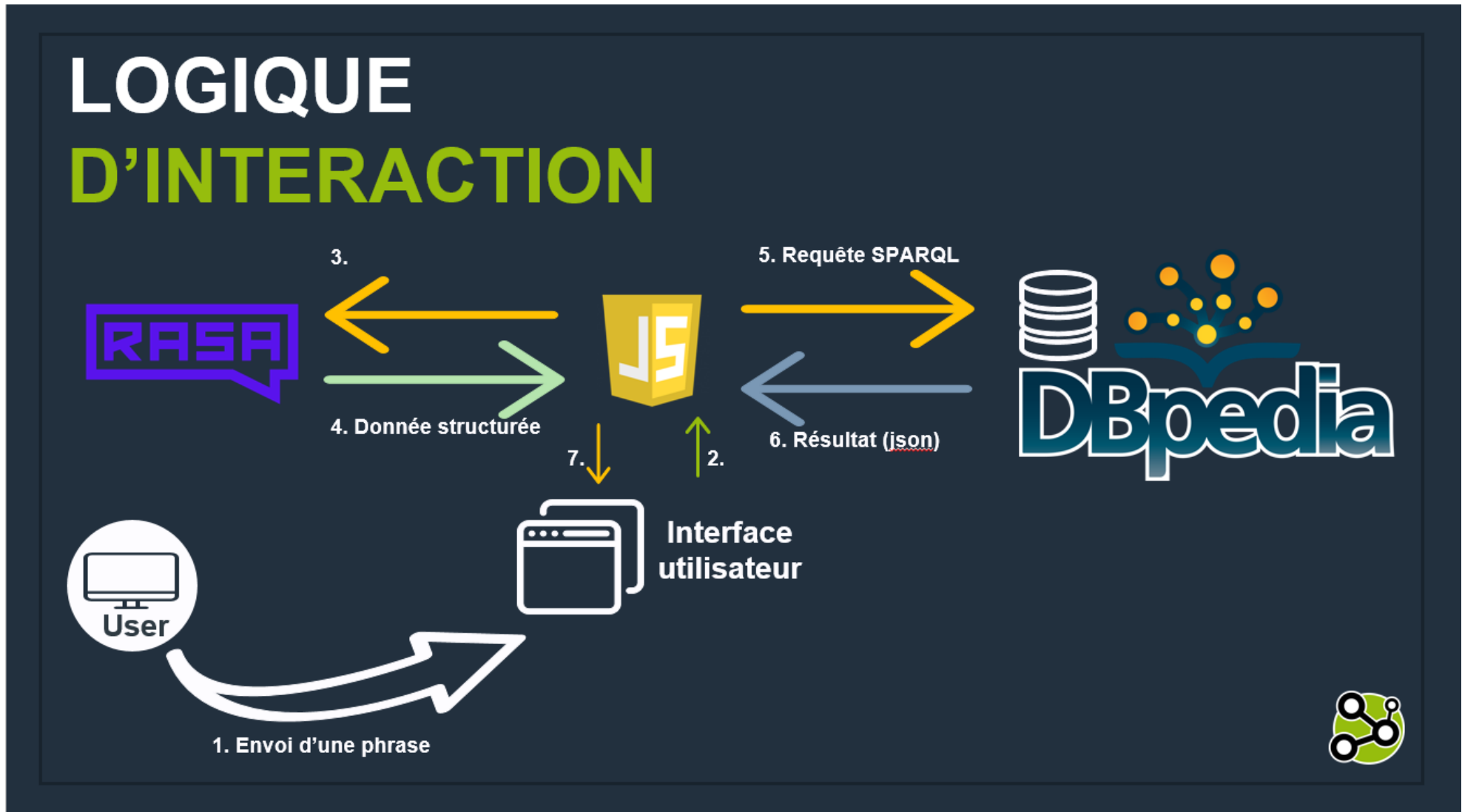


FIGURE 16 : INTERACTIONS DU CHATBOT S3 ENTRE LES DIFFERENTS ACTEURS TECHNOLOGIQUES : DBPEDIA(CLOUD), JAVASCRIPT(COTE CLIENT) ET RASA(COTE SERVEUR)

3.8. ANNEXE 8 – COMPETENCY QUESTIONS

| | |
|--|--|
| Le chatbot sait-il saluer l'utilisateur ? | |
| Le chatbot sait-il donner la définition d'un mot sans mise en forme ? | |
| Le chatbot sait-il remonter les traductions d'un mot sans mise en forme ? | |
| Le chatbot sait-il remonter les synonymes d'un mot sans mise en forme ? | |
| Le chatbot sait-il donner les mots apparentés du mot sans mise en forme ? | |
| Le chatbot sait-il répondre aux tâches précédentes avec une mise en forme quelconque ? | |
| Le chatbot est-il capable de mettre en forme les traductions d'un mot sous forme d'un tableau à 2 colonnes : - Colonne 1 : Mot traduit - Colonne 2 : Langue + éventuellement un drapeau | |
| Le chatbot est-il capable de mettre en forme les synonymes et mots apparentés d'un mot sous forme d'une carte mentale ? | |
| Le chatbot est-il capable de restituer toutes les informations simultanément concernant un mot : définition, traduction(s), synonymes, mots apparentés | |
| Le chatbot est-il capable de restituer les informations concernant un mot en comprenant le type d'information que veut l'utilisateur ? (Définition si l'utilisateur demande une définition, synonyme(s) si l'utilisateur demande des synonymes, etc.) | |
| Le chatbot sait-il répondre à une question de quantité sans mise en forme particulière ? (type : Combien de fois la terre tourne-t-elle sur elle même pour une durée de 1 an) | |
| Le chatbot sait-il répondre à une question de quantité avec mise en forme particulière de type camembert ou histogramme | |

FIGURE 17 : COMPETENCY QUESTIONS RELATIVES AU CHATBOT. (VERT : IMPLEMENTATION FONCTIONNELLE / ROUGE : IMPLEMENTATION ABANDONNEE OU PAS ENCORE REALISEE)

3.9. ANNEXE 9 – DEMONSTRATION DU CHATBOT

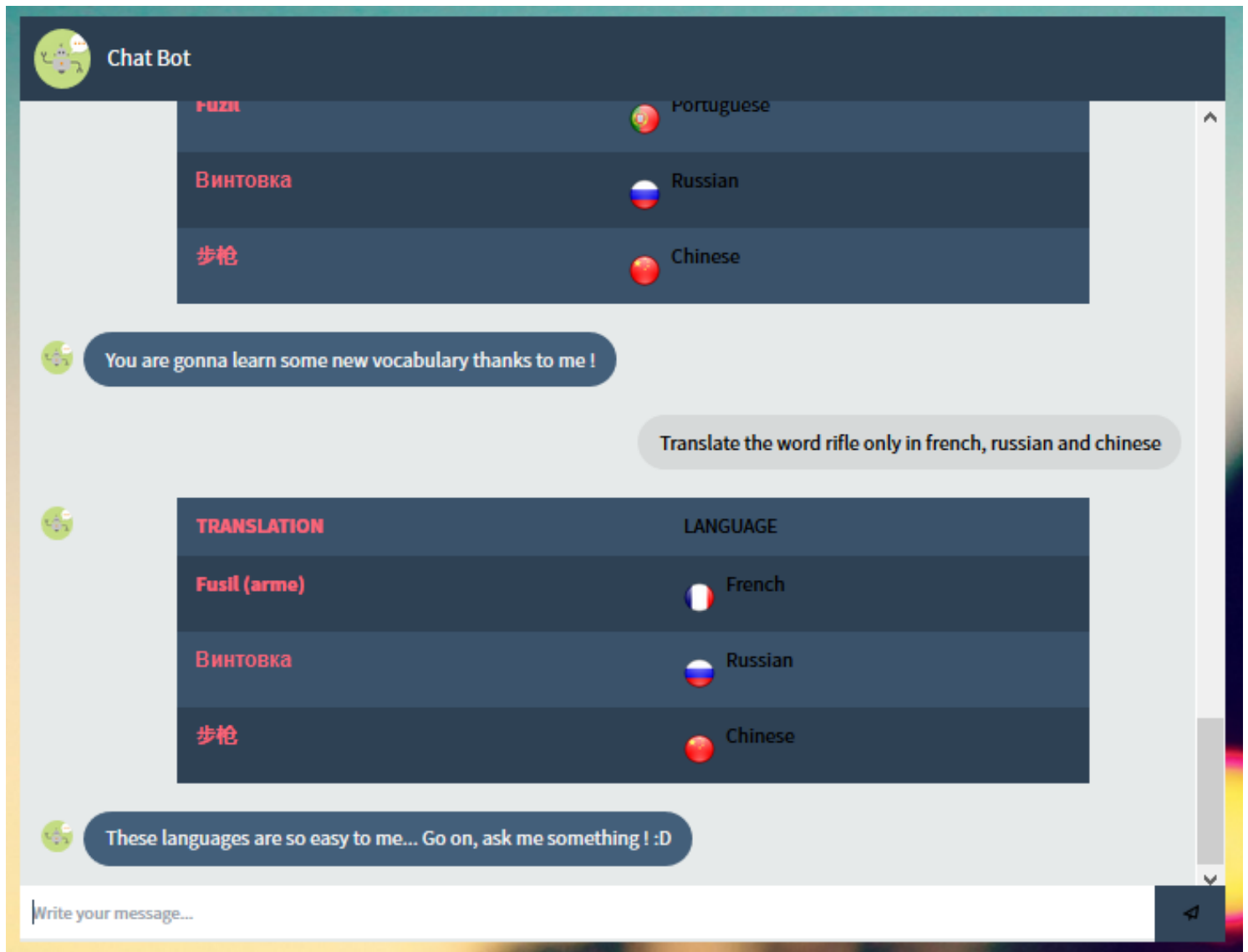


FIGURE 18 : TRADUCTIONS CHATBOT S3

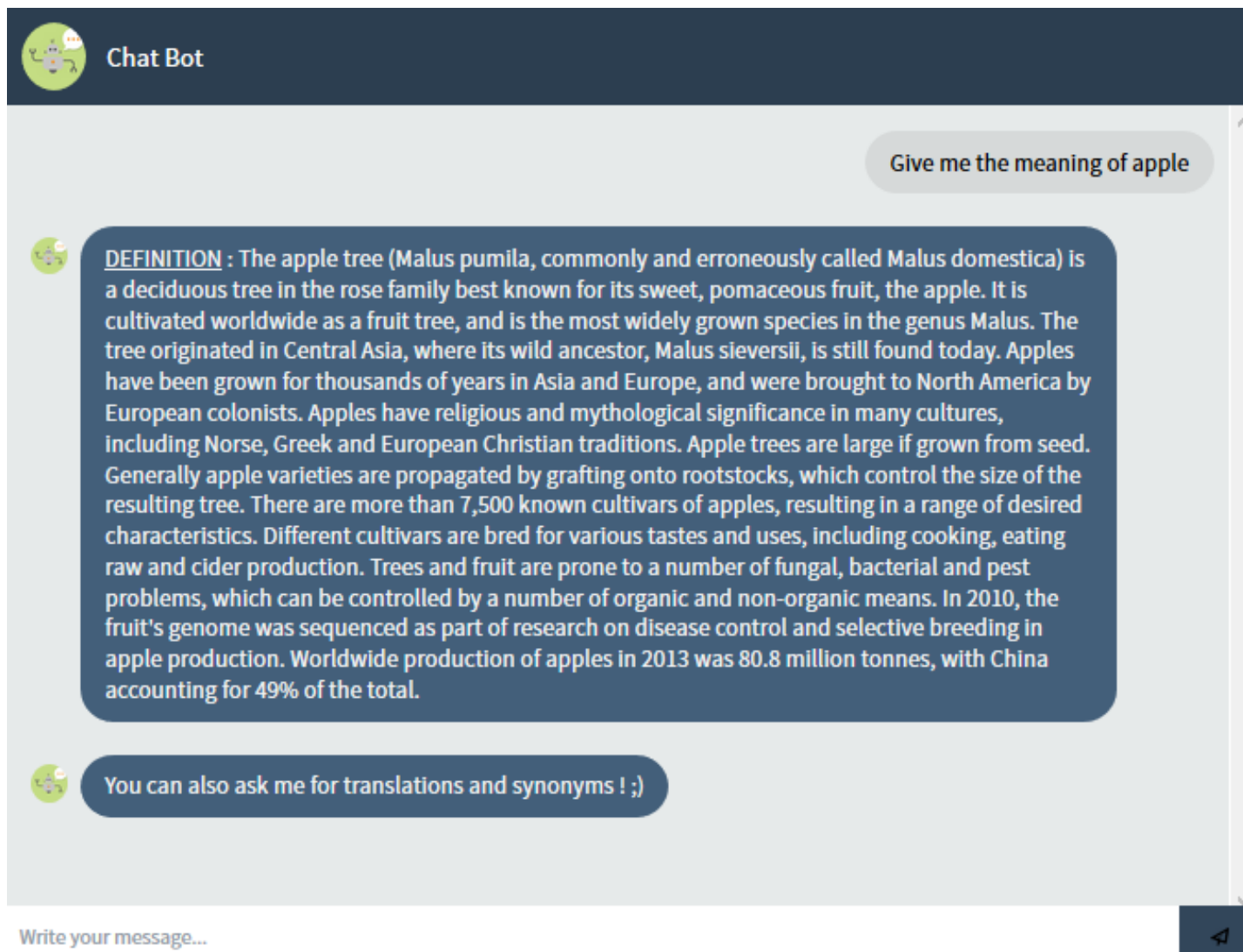


FIGURE 19 : DEFINITION CHATBOT S3

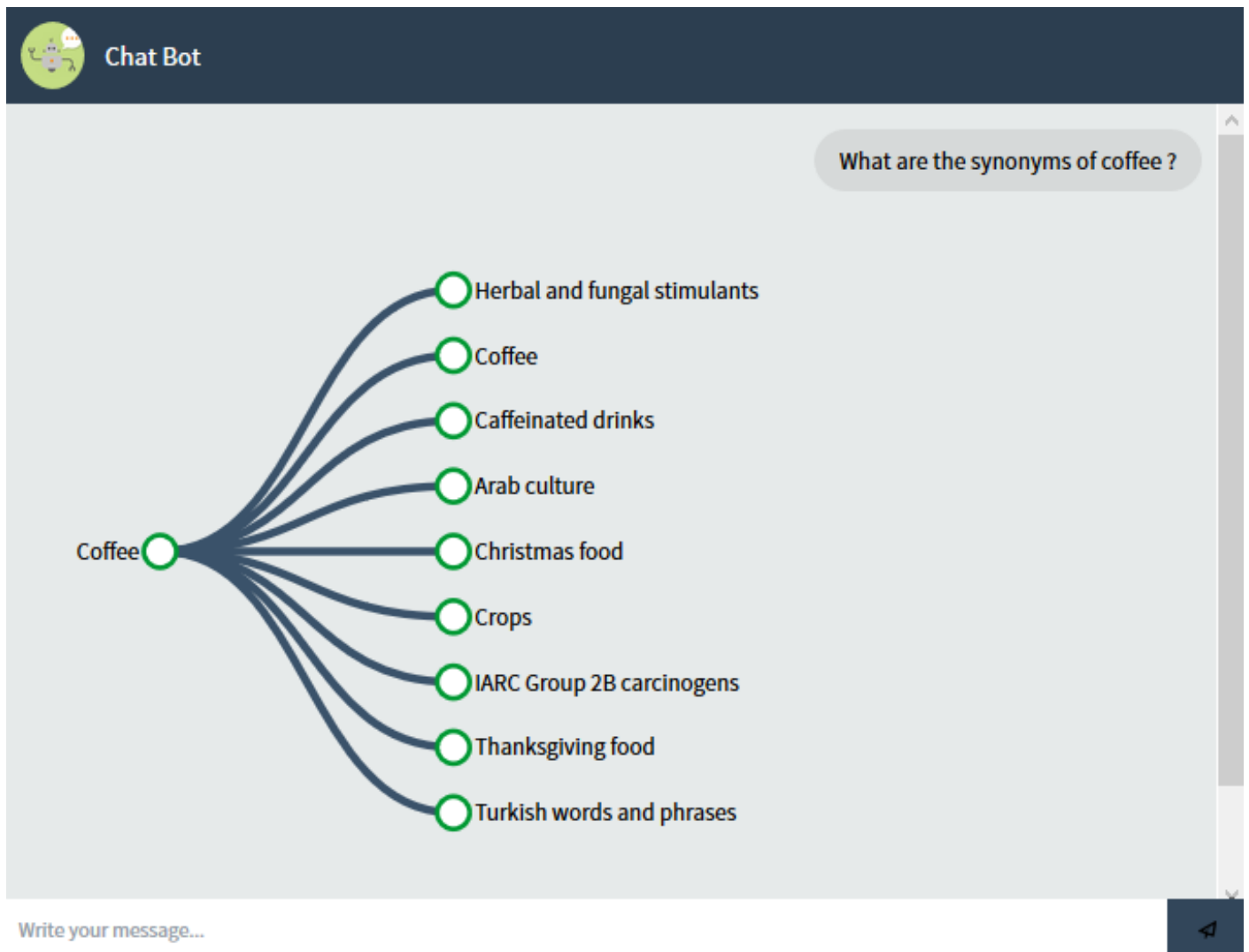


FIGURE 20 : SYNONYMES CHATBOT S3

3.10. ANNEXE 10 – GRANDES PHASES ITERATIVES DU PROJET



FIGURE 21 : CYCLE DES TACHES REALISEES LORS DE L'IMPLEMENTATION D'UN CHATBOT

3.11. ANNEXE 11 – ETUDE COMPARATIVE DE PLATEFORMES DE NLU

| Nom de l'application de NLU | Compagnie offrant le service | Fonctionnalités | Licence | Langues supportées | Domaines de NLU | Prise en main |
|--|------------------------------|--|--|--|---|---|
| Microsoft Language Understanding Intelligent Service (LUIS.AI) | Microsoft | 1. Classification intents/entities 2. Une application LUIS = un domaine spécifique d'agent conversationnelle (meilleure lisibilité) 3. Active learning 4. Classes, modèles pré-existants depuis Cortana 5. Requêtage HTTP et format de retour JSON | LUIS API-FREE : 10 000 transactions gratuites par mois LUI API-BASIC : 1,50\$ les 1 000 transactions, 50 transactions par seconde | English, French, Italian, German, Spanish, Brazilian Portuguese, Japanese, Korean and Chinese. | Designé pour permettre au développeur de construire des applications qui peuvent comprendre le langage humain et leurs requêtes | Simple |
| Wit.ai | Facebook | Intents/entities, mise en contexte, action suivante, Natural Language Processing | Gratuit | Albanian, Arabic, Azerbaijani, Bengali, Bosnian, Bulgarian, Burmese, Catalan, Chinese, Croatian, Czech, Danish, Dutch, English , Estonian, Finnish, French, Georgian, German, Greek, Hebrew, Hindi, Hungarian, Icelandic, Indonesian, Italian, Japanese, Korean, Latin, Lithuanian, Macedonian, Malay, Norwegian, Persian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak, Slovenian, Spanish, Swahili, Swedish, Tagalog, Tamil, Thai, Turkish, Ukrainian and Vietnamese. | Bots, applis mobile, domotique, objets connectés, robots. | Interface nuisant à la compréhension, prise en main relativement complexe |
| Dialogflow | Google | 1. Dialogflow retourne l'intention la plus probable en fonction d'une requête donnée. 2. Classification intent/entities Format de sortie JSON uniquement | Gratuit | Portugais, Chinois simplifié & traditionnel, Anglais , Hollandais, Français, Allemand, Italien, Japonais, Coréen, Russe, Ukrainien, Espagnol | Bots, applications, services ou dispositifs électroniques (smartphones, téléphones...) | Très simple |

FIGURE 22 : TABLEAU COMPARATIF DE DIFFERENTES CARACTERISTIQUES DE 3 PLATEFORMES DE NLU EN LIGNE

3.12. ANNEXE 12 – INTERFACE DE DIALOGFLOW

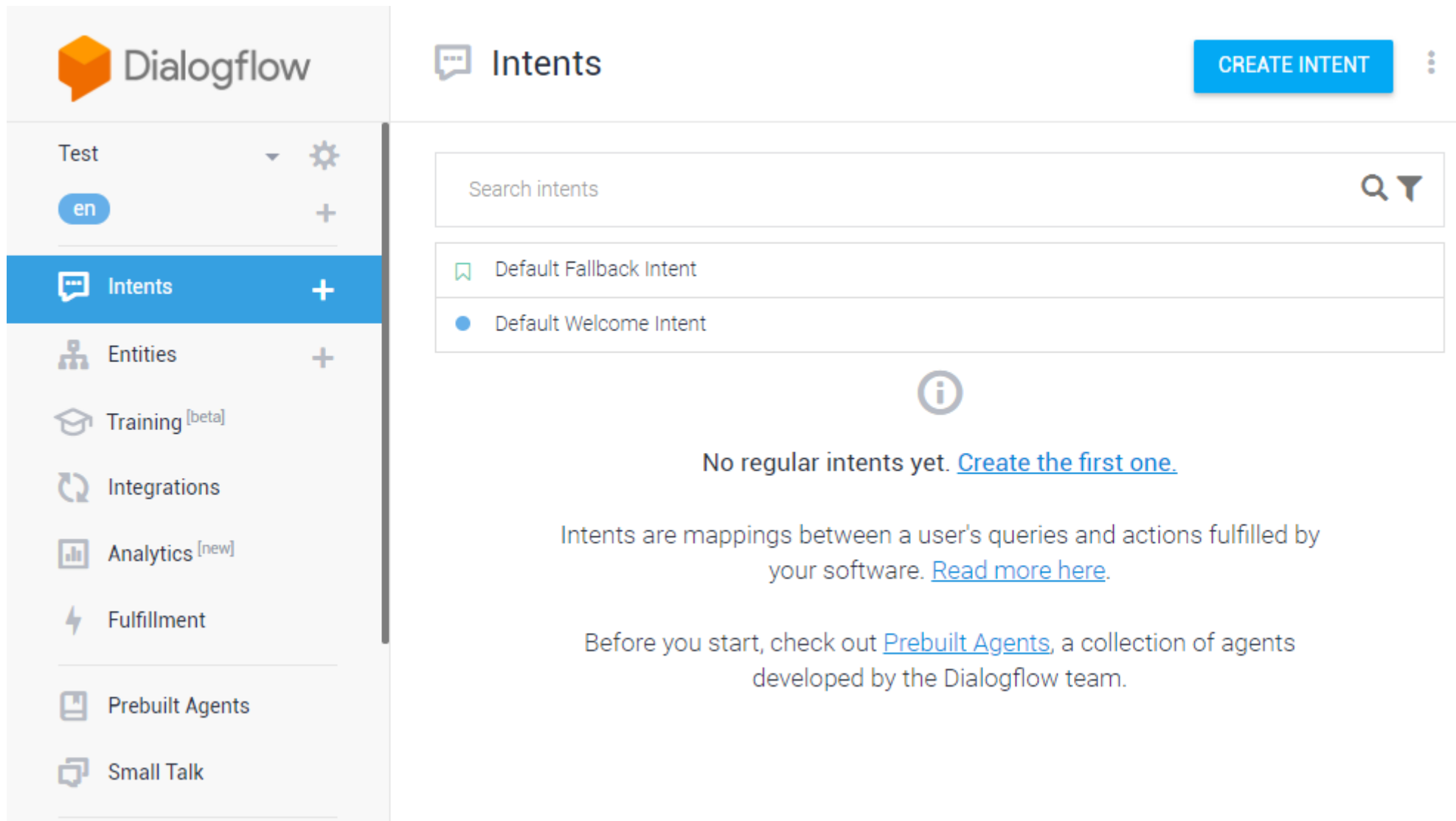


FIGURE 23 : INTERFACE DE CREATION D'UNE APPLICATION - DIALOGFLOW

3.13. ANNEXE 13 – INTERFACE DE WIT.AI

 DorianNaaji / Test / ●

Test how your app understands a sentence

You can train your app by adding more examples

User says...

⊕ Add a new entity

✓ Validate

Your app uses 2 entities

| Entity | Description | Values |
|---|--|---------|
| <div><div>intent →</div><div>LOOKUP STRATEGIES trait</div></div> | User-defined entity | get_Lol |
| <div><div>wit/temperature →</div><div>ROLES <u>get_Temp</u></div></div> | Temperature in degrees Celsius or Fahrenheit | |

FIGURE 24 : INTERFACE D'ACCUEIL - WIT.AI

intent

LOOKUP STRATEGIES ⓘ

trait

free-text & keywords

free-text

keywords

User-defined entity

Insights ⓘ

Validate more expressions to get insights for this entity 😊

Trait Values

Trait value ⓘ

get_Lol

➕ Add a new trait

Expressions

Filter by:

All values ▼

“ Search through your expressions.

Text

FIGURE 25 : INTERFACE DE CREATION « D'ENTITIES » - WIT.AI

3.14. ANNEXE 14 – INTERFACE DE LUIS.AI

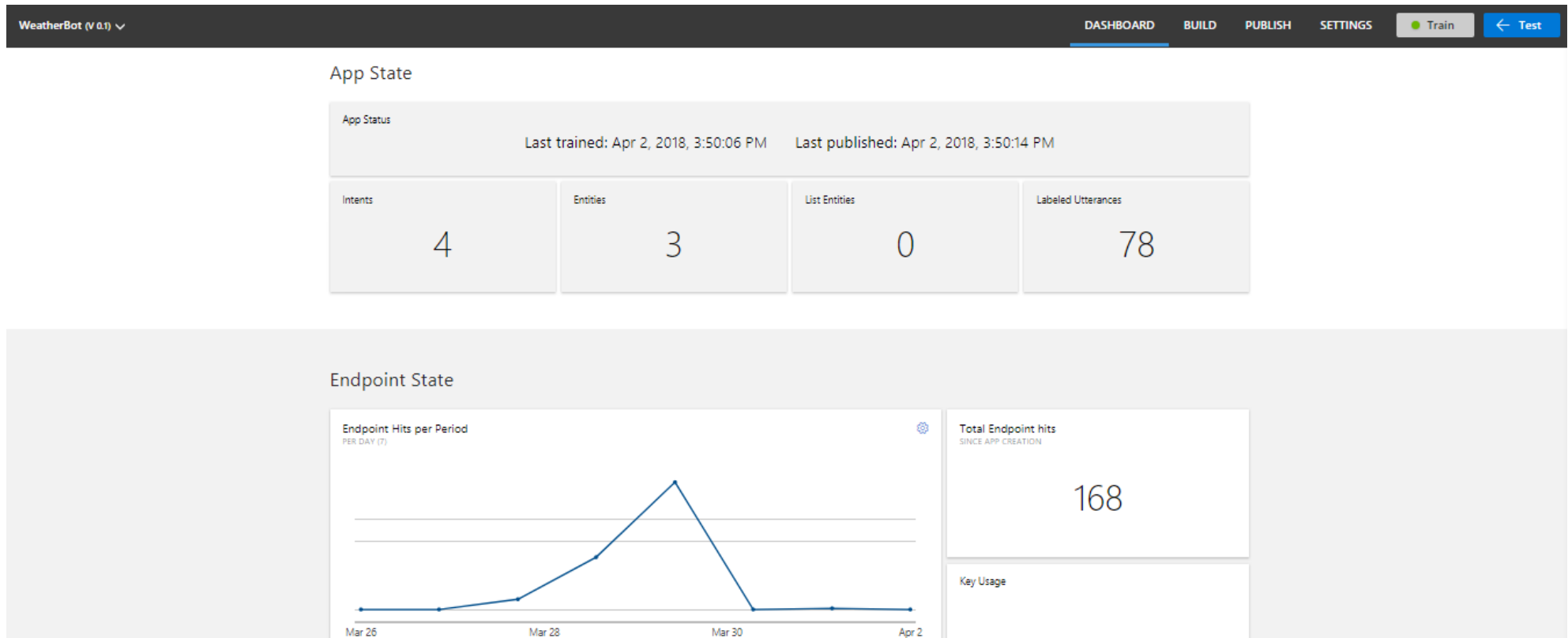


FIGURE 26 : STATUT DE L'APPLICATION - LUIS.AI

Detailed Model View

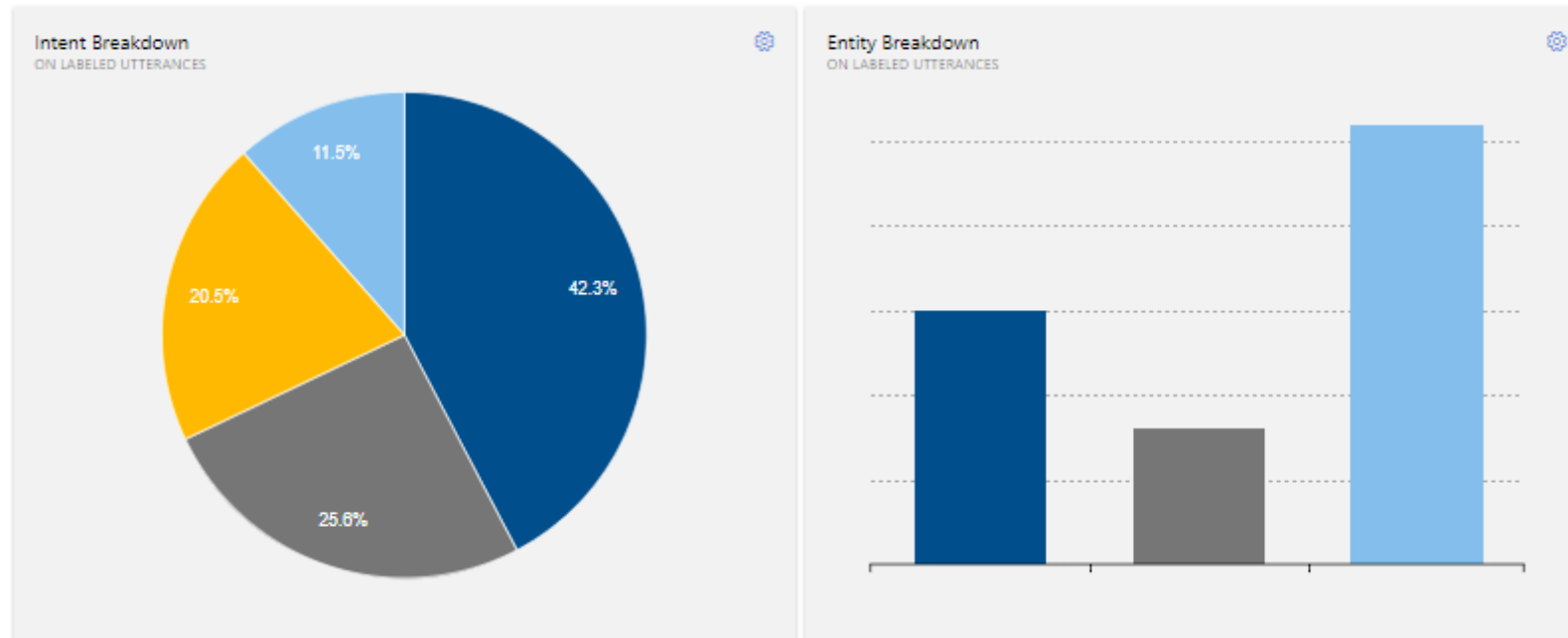


FIGURE 28 : REPARTITION DES PHRASES D'UTILISATEUR EN FONCTION DE LEUR "INTENT" - LUIS.AI

My Apps ?

[Create new app](#)
[Import new app](#)

| Name | Culture | Created date | Endpoint hits | |
|------------------------------------|---------|--------------|---------------|-----|
| ChatBot (V 0.1) | en-us | 12/16/2017 | 473 | ... |
| WeatherBot (V 0.1) | en-us | 3/27/2018 | 168 | ... |

FIGURE 27 : RECAPITULATIF DES APPLICATIONS DISPONIBLES - LUIS.AI

Language UnderstandingMy appsDocsPricingSupportAbout

WeatherBot (V 0.1) ▾DASHBOARDBUILDPUBLISHSETTINGSTrainTest

App AssetsIntentsEntitiesImprove app performanceReview endpoint utterancesPhrase listsPREVIEWPrebuilt Domains

Intents ?

Create new intentAdd prebuilt domain intentSearch intents

| Name | Utterances | |
|-------------------------|------------|-----|
| current_weather_no_town | 16 | ... |
| greet_user | 9 | ... |
| None | 20 | |
| weather_town | 33 | ... |

FIGURE 29 : CRÉATION “D'INTENTS” - LUIS.AI

3.15. ANNEXE 15 – SERVICES ET TARIFS OPENWEATHER MAP


| <div>  <div> Weather Maps API Price Partners Stations Widgets <div> °C °F </div> News About </div> </div> | | | | | |
|---|---------------------------------------|---------------------------|---------------------------|---------------------------|---------------------------|
| | Free | Startup | Developer | Professional | Enterprise |
| Price Price is fixed, no other hidden costs. | Free | 40 USD / month | 180 USD / month | 470 USD / month | 2,000 USD / month |
| Subscribe | Get API key and Start | Subscribe | Subscribe | Subscribe | Subscribe |
| Calls per minute (no more than) | 60 | 600 | 3,000 | 30,000 | 200,000 |
| Current weather API | ✓ | ✓ | ✓ | ✓ | ✓ |
| 5 days/3 hour forecast API | ✓ | ✓ | ✓ | ✓ | ✓ |
| 16 days/daily forecast API | - | ✓ | ✓ | ✓ | ✓ |
| Weather maps API | ✓ | ✓ | ✓ | ✓ | ✓ |
| Bulk download | - | - | - | ✓ | ✓ |
| UV index (beta) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Air pollution (beta) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Weather alerts (beta) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Service | | | | | |
| Availability | 95.0% | 95.0% | 99.5% | 99.5% | 99.9% |
| SLA | - | - | - | - | ✓ |
| Weather API data update | < 2 hours | < 2 hours | < 1 hour | < 10 min | < 10 min |
| Weather maps data update | < 3 hours | < 3 hours | < 3 hours | < 3 hours | < 3 hours |
| API lifetime support | Current version | Current version | Current version | All versions | All versions |
| SSL | ✓ | ✓ | ✓ | ✓ | ✓ |
| License for maps, APIs, and other products | CC BY-SA 4.0 | CC BY-SA 4.0 | CC BY-SA 4.0 | CC BY-SA 4.0 | CC BY-SA 4.0 or custom |
| License for data and database | ODbL | ODbL | ODbL | ODbL | ODbL or custom |
| Tech support | Helpdesk | Helpdesk | Helpdesk | Direct | Direct 24x7 |

FIGURE 30 : GRILLE DE SERVICES OPENWEATHER MAP

3.16. ANNEXE 16 – SERVICES & TARIFS FIXER.IO






| MOST POPULAR | | | | |
|---|---|---|---|--|
|  FREE \$0 No hidden costs SHOW OPTIONS |  BASIC \$10 Price / month SHOW OPTIONS |  PROFESSIONAL \$40 Price / month SHOW OPTIONS |  PROFESSIONAL PLUS \$80 Price / month SHOW OPTIONS |  ENTERPRISE Volume Pricing CONTACT US |
| </> 1.000 API Calls / mo Hourly Updates Limited Support Historical Data | </> 10.000 API Calls / mo Hourly Updates Premium Support Historical Data SSL Encryption All Base Currencies Conversion Endpoint | </> 100.000 API Calls / mo 10-minute Updates Premium Support Historical Data SSL Encryption All Base Currencies Conversion Endpoint Time-Series Endpoint | </> 500.000 API Calls / mo 60-second Updates Premium Support Historical Data SSL Encryption All Base Currencies Conversion Endpoint Time-Series Endpoint Fluctuation Endpoint | </> Volume API Calls / mo 60-second Updates Dedicated Support Historical Data SSL Encryption All Base Currencies Conversion Endpoint Time-Series Endpoint Fluctuation Endpoint Custom Solutions CONTACT US |

FIGURE 31 : GRILLE TARIFAIRE ET DIFFERENTS SERVICES PROPOSES - FIXER.IO

3.17. ANNEXE 17 – SPECIFICATIONS TECHNIQUES

| | |
|------------------------|---|
| Processor | Up to 5th Generation Intel® Core™ i3-5005U Processor (2.0GHz 3MB) |
| Operating System | Windows 10 Home |
| Graphics | Up to Intel® HD Graphics 5500 |
| Memory | Up to 4GB DDR3L 1600 MHz |
| Storage | Up to 1TB 5400 RPM |
| Webcam | 0.3M single microphone |
| Audio | 1 x 1.5W speaker |
| Battery | 4 Cell 32 Watt Hour Li-Cylindrical |
| Display | 15.6" HD Glossy (1366 x 768) with integrated camera |
| Dimensions (W x D x H) | 14.88" x 10.43" x 0.89" |
| Weight | 5.1 lbs |
| Bluetooth® | WiFi 802.11 b/g/n, Bluetooth® 4.0 |
| Connectors | 1 x USB 3.0, 1 x USB 2.0, HDMI-out, 4-in-1 card reader, Audio combo jack, RJ-45 |

FIGURE 32 : TABLEAU DES SPECIFICATIONS TECHNIQUES DU LENOVO IDEAPAD 100-15IBD - WWW3.LENOVO.COM

3.18. ANNEXE 18 – EXTRAIT DU CODE SOURCE JAVASCRIPT DU CHATBOT S3

```
880     .transition()
881     .duration(duration)
882     .attr("d", function(d)
883     {
884         var o =
885         {
886             x: source.x, y: source.y
887         };
888         return diagonal(
889         {
890             source: o, target: o
891         });
892     })
893     .remove();
894
895     // keep position in memory
896     nodes.forEach(function(d)
897     {
898         d.x0 = d.x;
899         d.y0 = d.y;
900     });
901 }
902
903 // display children on click
904 function click(d)
905 {
906     if (d.children)
907     {
908         d._children = d.children;
909         d.children = null;
910     }
911     else
912     {
913         d.children = d._children;
914         d._children = null;
915     }
916     update(d);
917 }
```

Line 1, Column 1

FIGURE 33 : EXTRAIT DU CODE JAVASCRIPT PRODUIT AU S3

3.19. ANNEXE 19 – DESIGN DU “WORDBOT” EN DEBUT DE PROJET

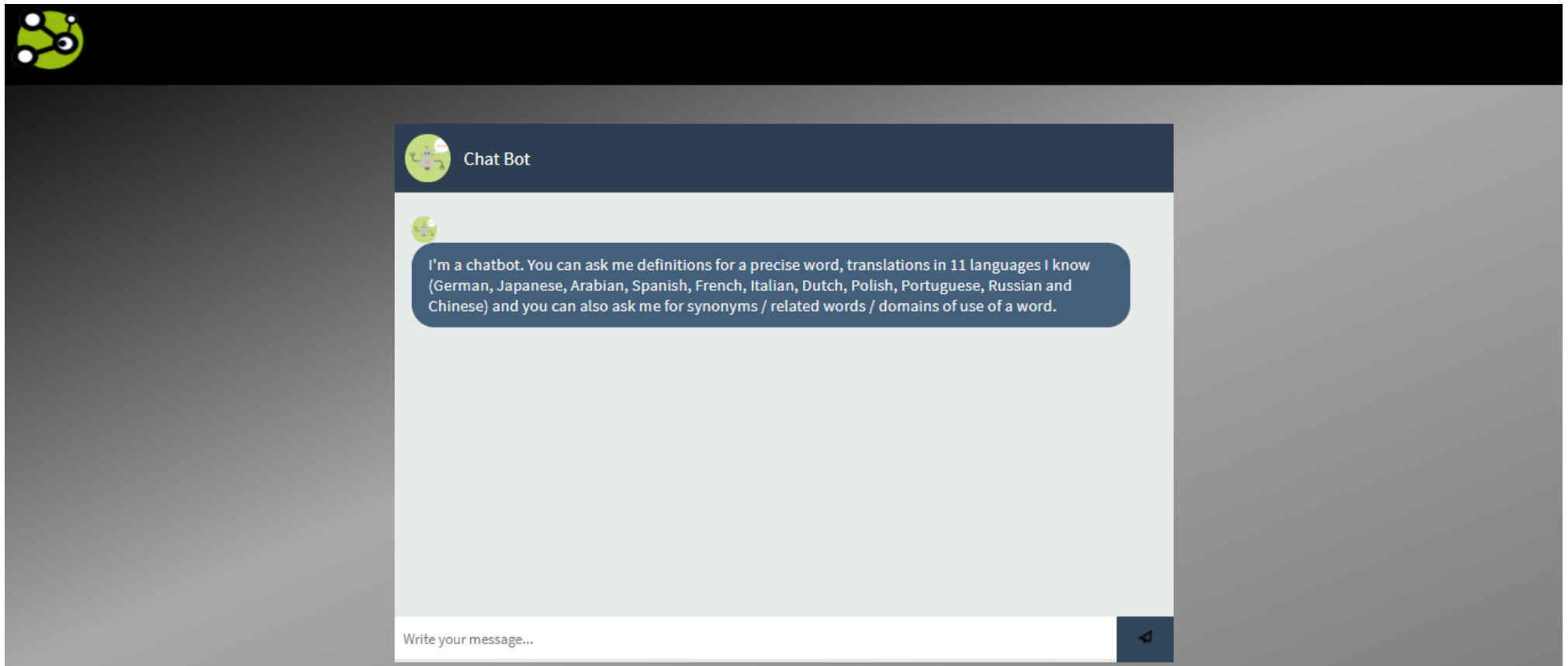


FIGURE 34 : INTERFACE DU "WORD-BOT" S4

3.20. ANNEXE 20 – LANDING PAGE EN DEBUT DE PROJET

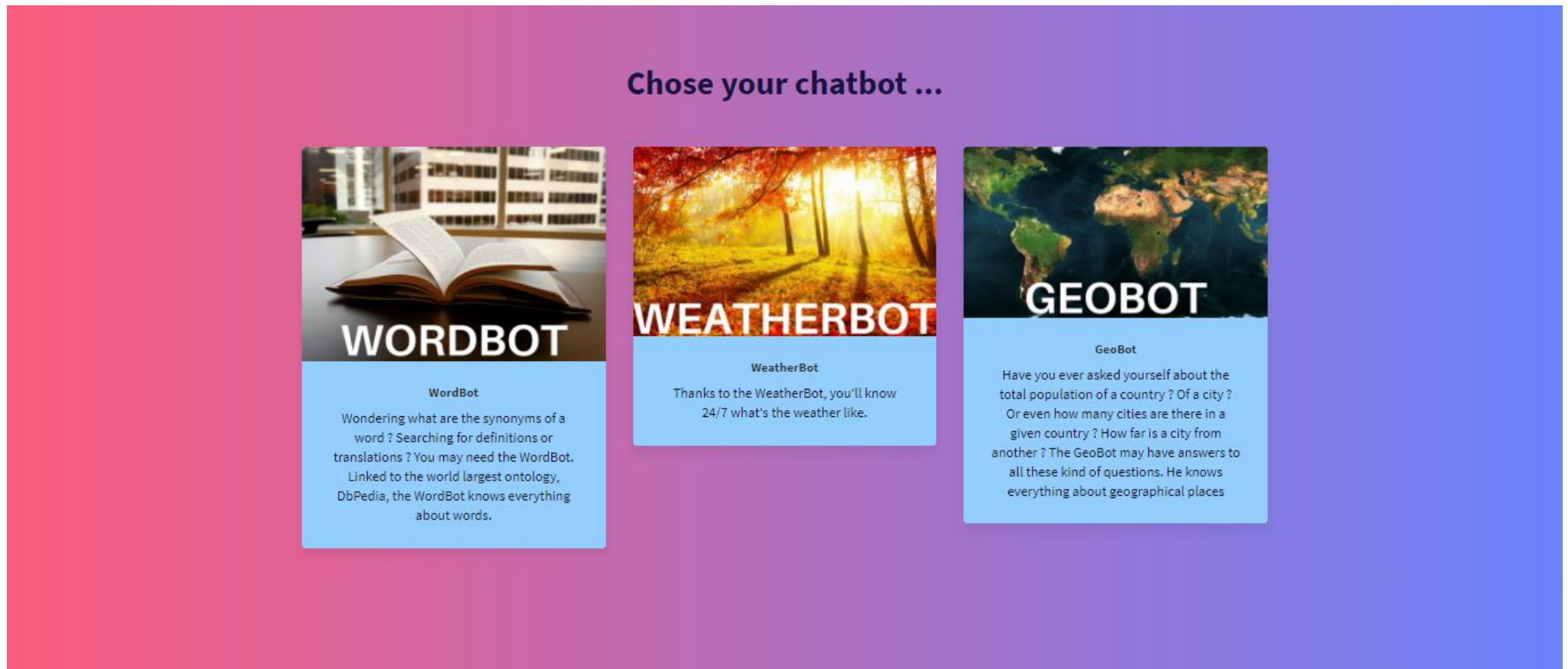


FIGURE 35 : LANDING PAGE REALISEE EN DEBUT DE PROJET

3.21. ANNEXE 21 – LANDING PAGE EN FIN DE PROJET

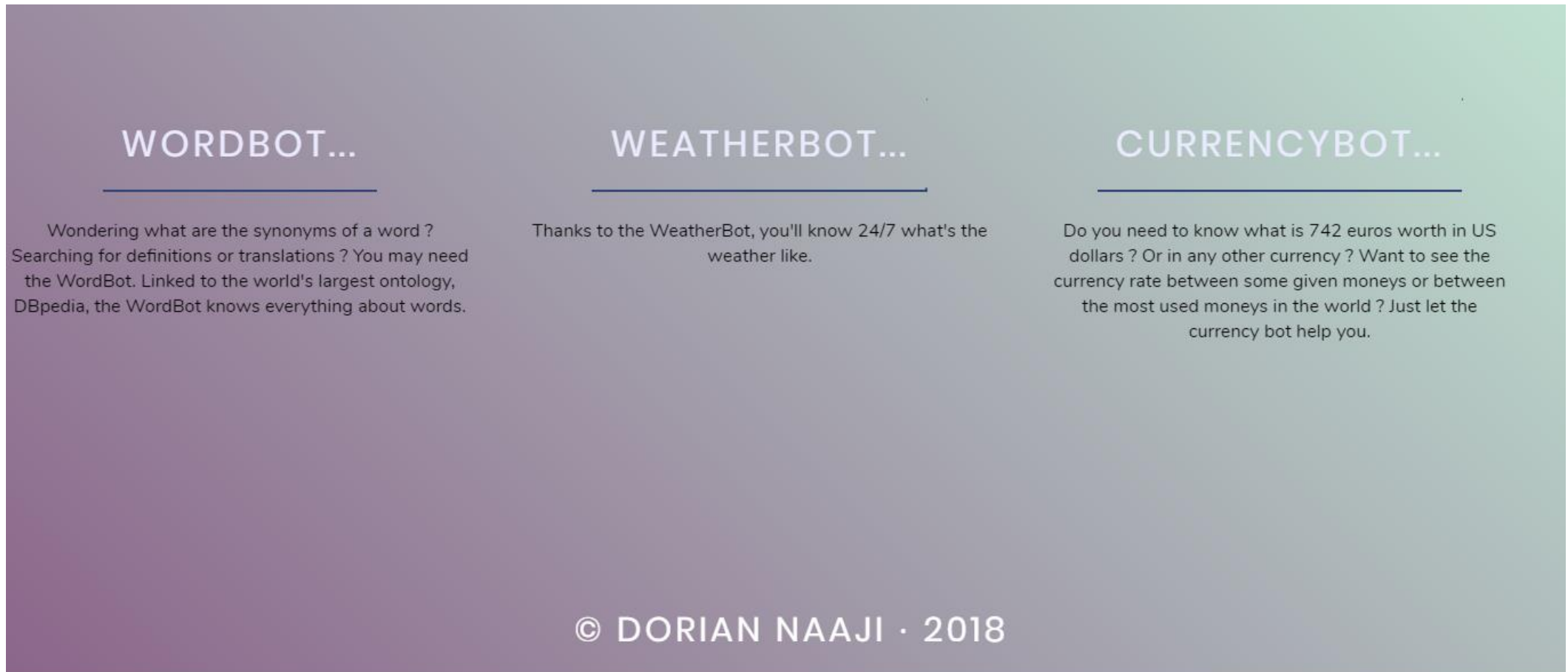


FIGURE 36 : LANDING PAGE EN FIN DE PROJET

3.22. ANNEXE 22 – DESIGN DES 3 CHATBOTS

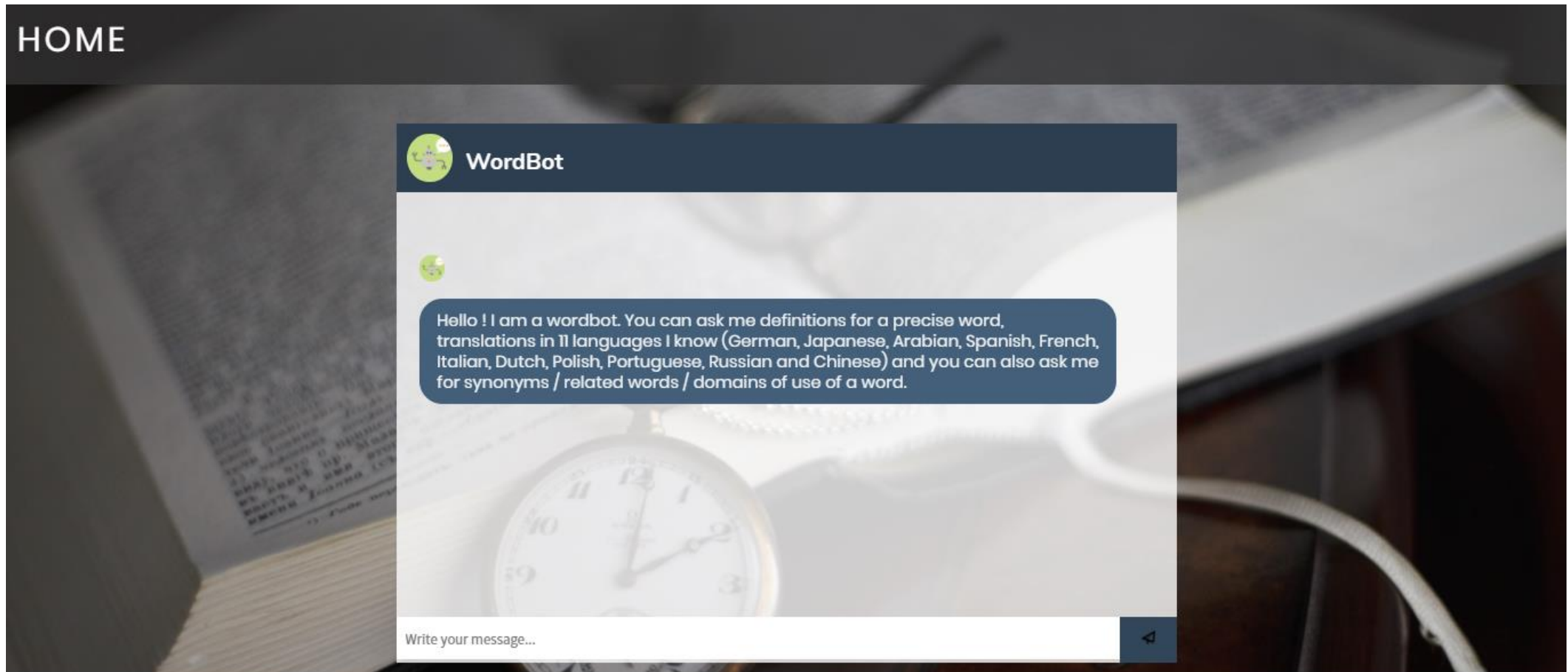


FIGURE 37 : DESIGN DU "WORDBOT" - DORIAN-NAAJI.FR/CHATBOT-2-0

HOME

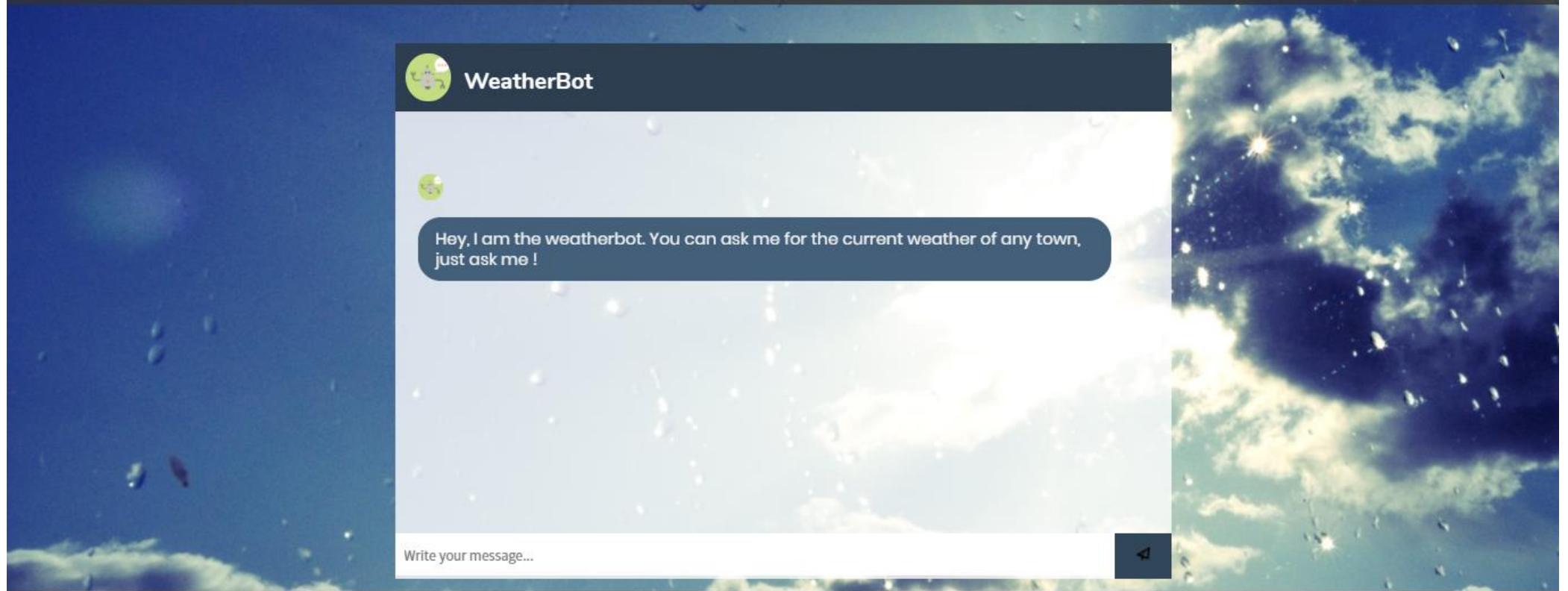


FIGURE 38 : DESIGN DU "WEATHER-BOT" - DORIAN-NAAJI.FR/CHATBOT-2-0

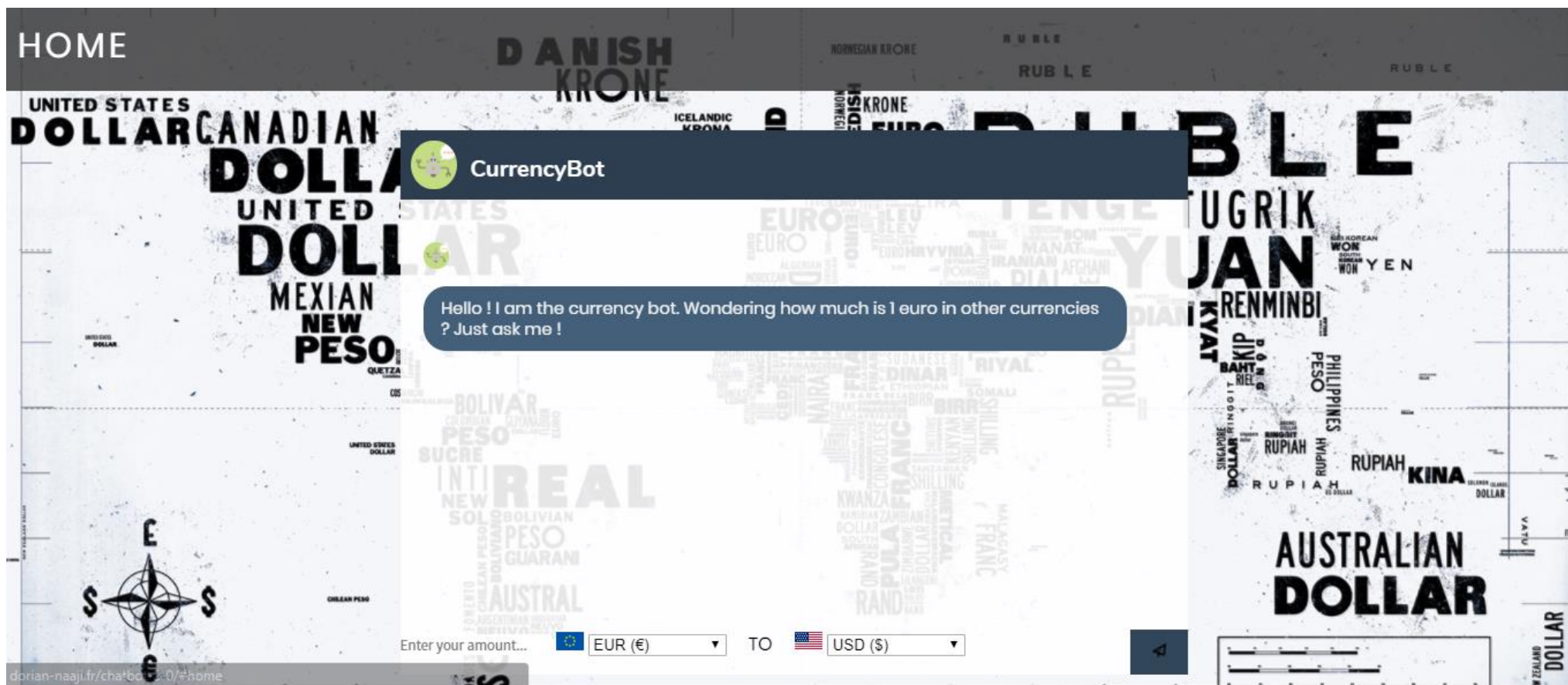


FIGURE 39 : DESIGN DU "CURRENCYBOT" - DORIAN-NAAJI.FR/CHATBOT-2-0

3.23. ANNEXE 23 – CLASSIFICATION INTENTIONS/ENTITÉS

La classification intent/entities est utilisée par la plupart des applications de NLU. Cela permet de transformer une phrase simple, comme « What is an apple » (qu'est-ce qu'une pomme) en donnée structurée compréhensible par la machine. Ici, luis.ai interprète la phrase et nous dit que « l'intent » (l'intention) la plus probable de l'utilisateur est « search_word_definition » (recherche de définition) et la seule « entity » (entité) trouvée est le mot « apple » (pomme). (voir Figure 41)

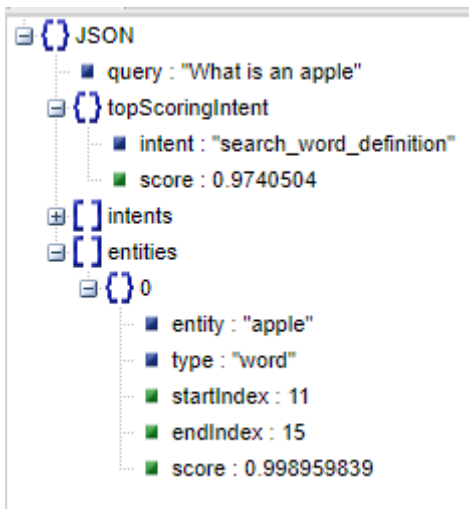


FIGURE 41 : DONNÉE STRUCTURÉE RETOURNÉE PAR LUIS.AI (1)

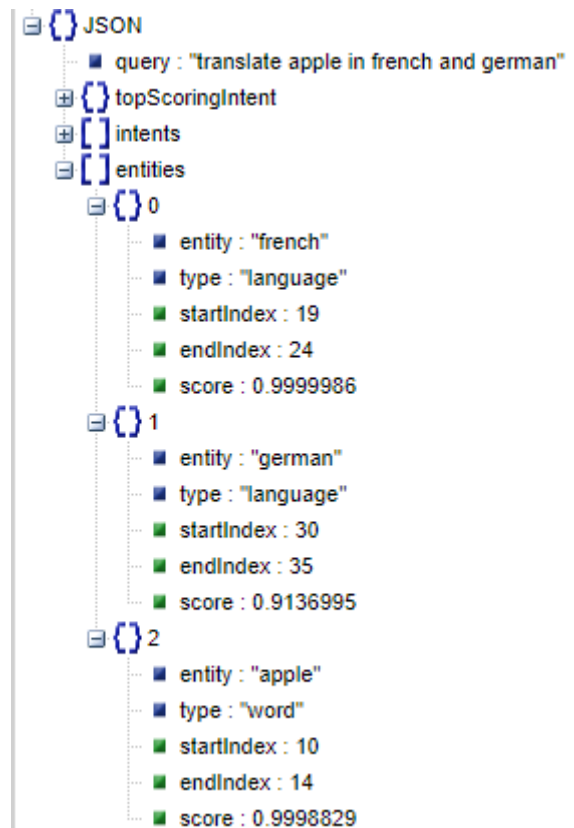


FIGURE 40 : DONNÉE STRUCTURÉE RETOURNÉE PAR LUIS.AI (2)

3.24. ANNEXE 24 – REQUETE OPENWEATHER MAP

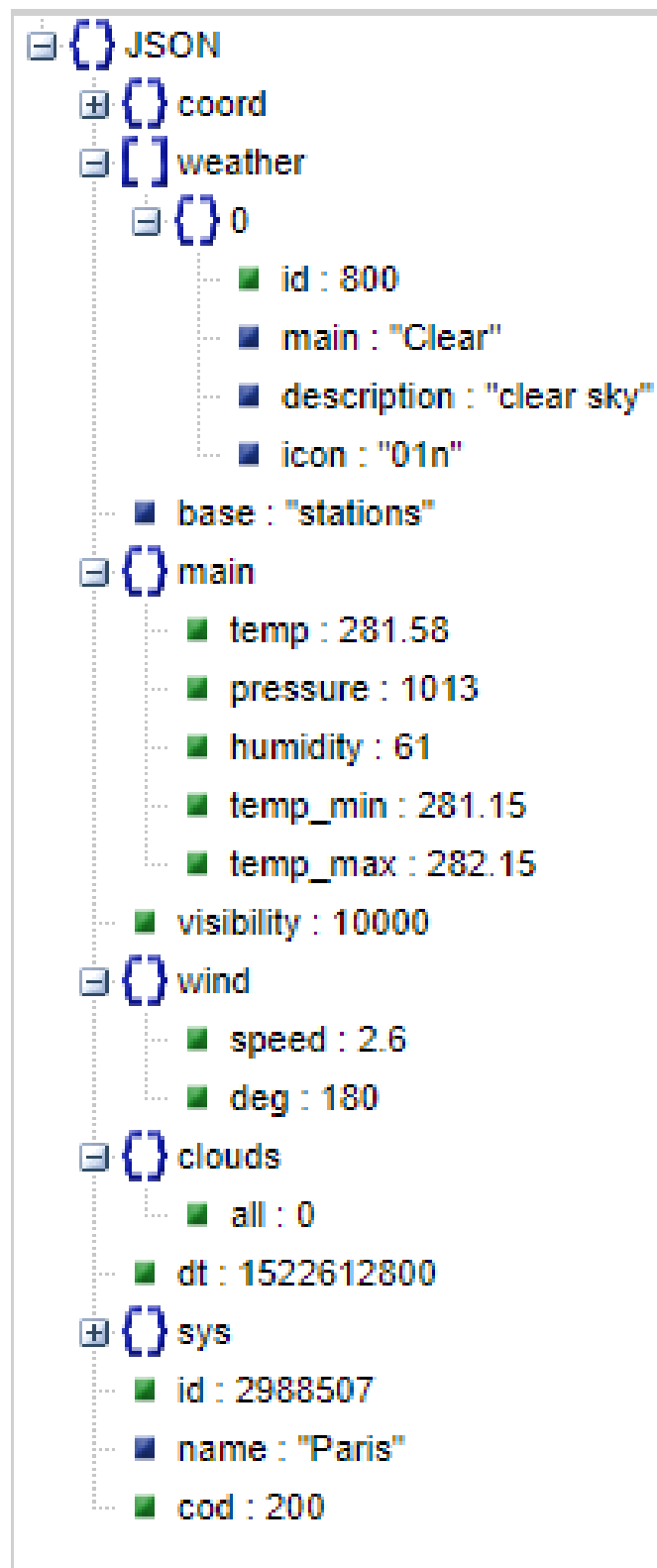


FIGURE 42 : EXEMPLE DE REPONSE D'UNE REQUETE
OPENWEATHER MAP AU FORMAT JSON

3.25. ANNEXE 25 – APPLICATION LUIS.AI WEATHERBOT

Intents ?

| <div>Create new intent</div> <div>Add prebuilt domain intent</div> | |
|--|------------|
| Name | Utterances |
| current_weather_no_town | 16 |
| greet_user | 9 |
| None | 20 |
| weather_town | 33 |

FIGURE 44 : "INTENTS" DU WEATHERBOT

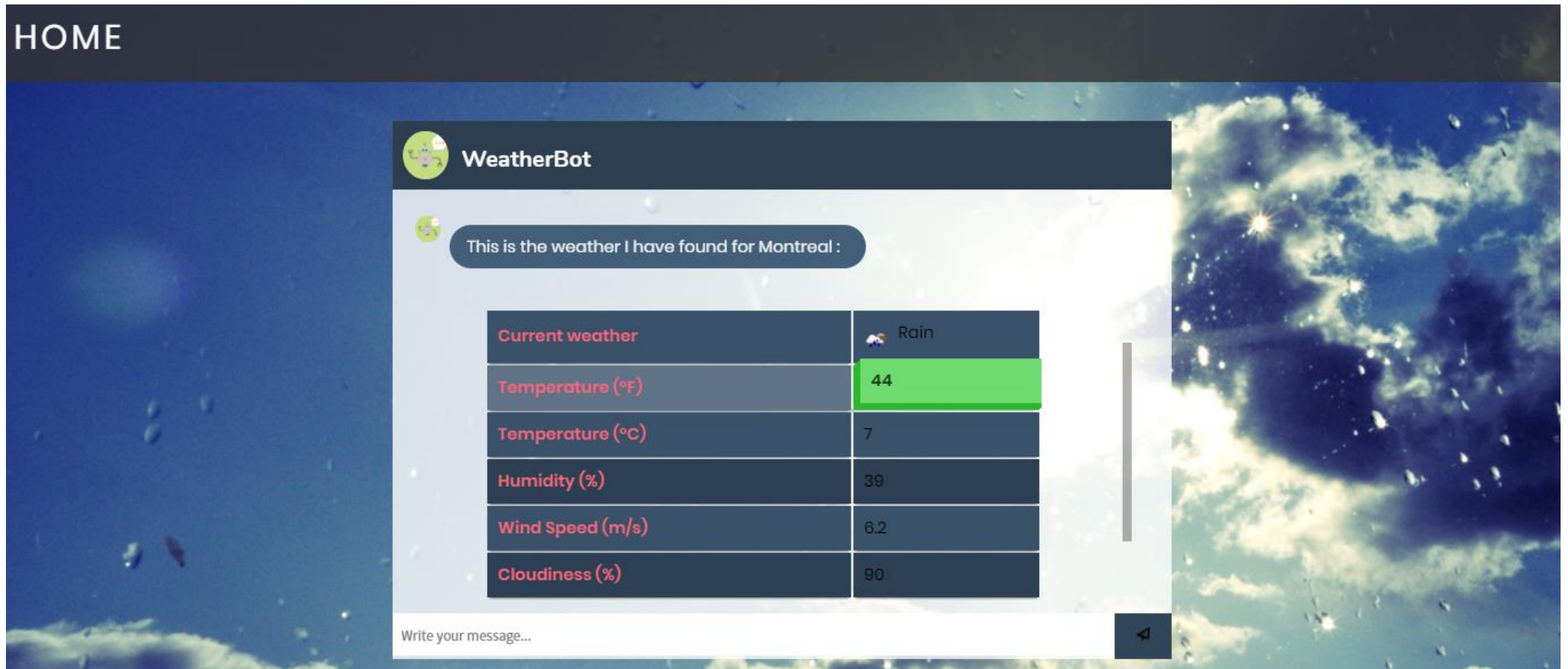
Entities ?

| <div>Create new entity</div> <div>Manage prebuilt entities</div> <div>Add prebuilt domain entity</div> <div>S</div> | | |
|---|-----------|--------------------|
| Name | Type | Labeled Utterances |
| city | Simple | 26 |
| city_composite | Composite | 8 |
| datetimeV2 | Prebuilt | N/A |
| weather_adjective | Simple | 15 |


FIGURE 43 : "ENTITIES" DU WEATHERBOT

3.26. ANNEXE 26 – REPONSE DU WEATHERBOT

HOME



The screenshot shows a chat window titled 'WeatherBot' with a green robot icon. A message from the bot says 'This is the weather I have found for Montreal :'. Below this is a table with weather details. The background of the chat window is a blue sky with white clouds. At the bottom, there is a text input field labeled 'Write your message...' and a dark blue button with a white paper plane icon.

| | |
|------------------|--|
| Current weather |  Rain |
| Temperature (°F) | 44 |
| Temperature (°C) | 7 |
| Humidity (%) | 39 |
| Wind Speed (m/s) | 6.2 |
| Cloudiness (%) | 90 |

Write your message...

FIGURE 45 : EXEMPLE DE REPONSE DU WEATHERBOT

3.27. ANNEXE 27 – ENSEMBLE DES MONNAIES INDISPONIBLES

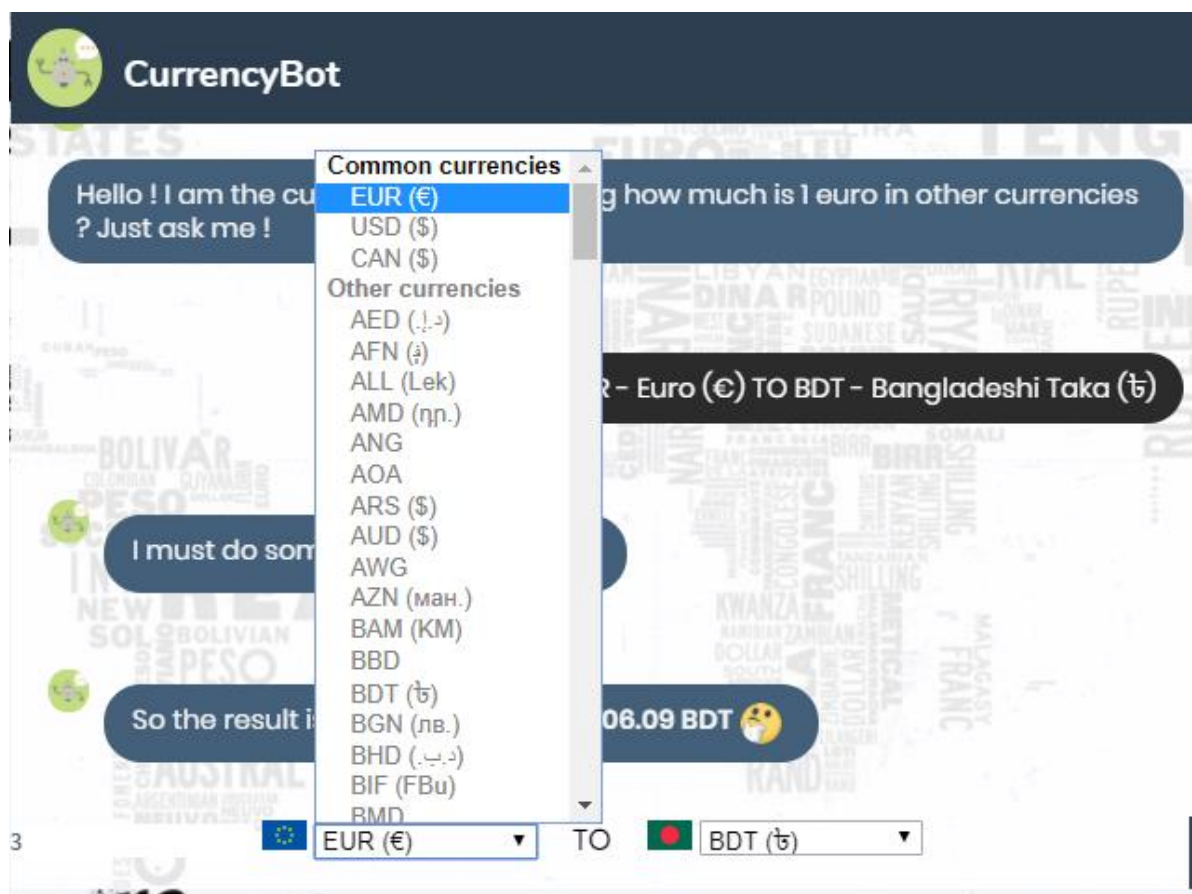


FIGURE 46 : MONNAIES DE BASE INDISPONIBLES POUR L'UTILISATEUR

3.28. ANNEXE 28 – EXEMPLE DE CONVERSATION (CURRENCYBOT)

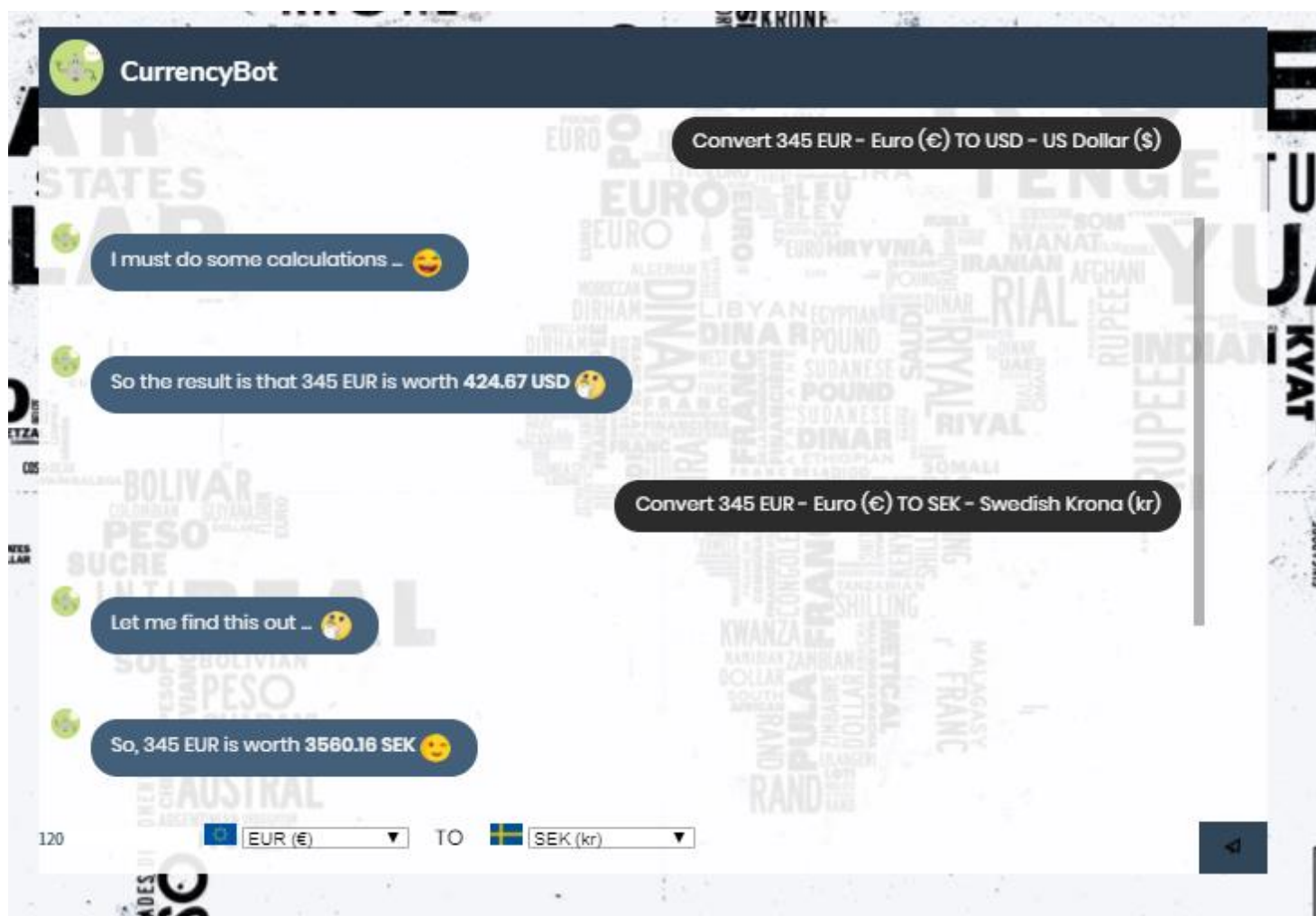


FIGURE 47 : EXEMPLE DE CONVERSATION AVEC LE "CURRENCYBOT"

4. RÉSUMÉS ET MOTS CLÉS

Dans le cadre d'un projet tutoré lors de mon 4^e Semestre à l'IUT de Dijon, j'ai eu l'opportunité d'approfondir le thème vu en projet tutoré au 3^e Semestre : le développement d'agents conversationnels. J'ai ainsi réalisé un agent conversationnel permettant de répondre à un utilisateur sur la météo dans une ville donnée et un second permettant de convertir des sommes d'argent dans toutes les monnaies nationales et supranationales recensées dans le monde.

Mots clés : Agent conversationnel, JavaScript, HTML, CSS, Traitement Automatique du Langage Naturel

During my last semester at the Institute of technology of Dijon, I had the opportunity to see more in detail what I've been doing during my third semester project: developing Chatbots. I've developed a weather bot that could give the current weather of a city and a currency bot that could convert EUROS in a given money thanks to given rates exchanges.

Keywords: Chatbot, JavaScript, HTML, CSS, Natural Language Understanding