Universidade do Minho
Escola de Engenharia
Departamento de Informática

Humberto João Alves Vaz

Chatbot for Digital Marketing and Customer Support

An Artificial Intelligence approach

July 2019

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Humberto João Alves Vaz

Chatbot for Digital Marketing and Customer Support

An Artificial Intelligence approach

Master dissertation
Master Degree in Computer Science

Dissertation supervised by
Rui Mendes
Ana Anjo

July 2019

ACKNOWLEDGEMENTS

During the writing of this dissertation I have received a great amount of support and assistance. I would like to thank my supervisor Dr. Rui Mendes, whose expertise and attention led me to a good path on the research topic.

I would like to acknowledge my colleagues at VILT for their wonderful spirit and collaboration. You supported me whenever I needed and were always ready to help me. I would like to single out some members at VILT: Mário Lameiras for the excellent opportunity that was conceded to me, Raphael Gonçalves for the patience and guidance, Lucas Oliveira and Gonçalo Rodrigues for their expert advices in the last few months.

Finally I would like to thank my parents Maria José and Agostinho for their wise counsels and for being present whenever I needed. I would like to thank my sister Bárbara and her boyfriend José for their kind support, my friend Carlos Afonso for providing the right amount of distraction, my two "homonymous" friends João Dias who spent great times in the past with me and all of my friends as well that heard my complains through rough times.

ABSTRACT

Human interaction with machines has never been so frequent as nowadays. In order to reduce the redundant workload of a human being that answers repeated and trivial questions regarding customer support on a digital marketing *website*, this work has the purpose of replacing this tedious job with an informatics tool, a dialogue tool.

A dialogue tool like a Chatbot that could handle customer support to a digital marketing *website*, provides the opportunity of placing human resources on ”non mechanical tasks”. Given that Chatbots exchange messages directly with customers, they could collect required protocol information in all the interactions. In spite of the possibility of needing human assistance, he will not need to ask these standard questions and will improve its efficiency.

By automating these required dialogues to answer questions about certain products, that would otherwise be responded by a human, the organizations will have the opportunity to place human resources in another sectors that are not so easily automated.

Keywords— Chatbot, Artificial Intelligence, Machine Learning

RESUMO

A interação humana com máquinas nunca foi tão frequente como nos dias de hoje. Com a intenção de reduzir a quantidade de trabalho de um ser humano que receberia ao responder a questões triviais e repetidas no que diz respeito a Suporte ao Cliente, este trabalho tem o propósito de substituir um trabalho entediante por uma ferramenta informática, uma ferramenta que possibilite o diálogo entre o cliente e o serviço de suporte.

Uma ferramenta como um Chatbot que poderia fornecer suporte ao cliente num *website* de marketing digital iria providenciar ás empresas a oportunidade de alocar trabalhadores para tarefas ”menos mecânicas”. Dado que os Chatbots trocam mensagens diretamente com os clientes, estes podem recolher informações que são sempre necessárias e protocolares em todas as interações. Assim sendo, mesmo que este diálogo requira possivelmente um ser humano, este irá prescindir de fazer estas perguntas padrão, melhorando assim a eficiência deste trabalho (Suporte ao Cliente).

Ao automatizar diálogos necessários para responder a questões acerca de produtos que, de outra forma seriam respondidas por um ser humano, as organizações estarão a poupar tempo e dinheiro que podem ser aplicados noutros sectores menos propícios a serem automatizados.

***Keywords*—** Chatbot, Artificial Intelligence, Machine Learning

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

”Hey my name is MacGyver, but you can call me Mac! How can I help you?”

Online retail transactions represent a huge part of the money earned by companies these days. It seems like a good idea to leverage the possible profit, offering easy and intuitive ways to increase the customer's happiness regarding the user experience while purchasing an enterprise Software-as-a-Service (SaaS) for their own companies.

VILT proposed this project with the intention of upgrading from an old-fashion FAQ into a Chatbot that could satisfy possible doubts from clients. The domain covered by this tool should be about the product(s) to be sold.

The aim is to create a tool that could collect new recurrent questions retrieved from users/clients day after day. This way we could achieve a list of frequent questions to be answered by a company worker with the perspective of updating the tool's domain with relevant information.

1.1 MOTIVATION

Humanity is living in years where information is easily accessible. With the help of the massification of information due to the internet and the technology's leap, people are increasingly buying only when well informed about the product. The availability and the clarity of the information gives confidence and satisfaction to the customer to proceed and buy a given product.

In the early days, people were afraid to buy on the internet. The way *e-commerce* works lacks in confidence comparing to traditional commerce. The product is not always the expected. There are a lot of ”Fishing schemes” and counterfeit products out there. There is a certain risk regarding sensible information security as well. These factors along with lack of organized information could be a barrier purchasing a product.

Nowadays, if a company wants to be the ”best selling company”, it will have to provide essential and complementary information about the products to be sold.

According to IBM, Chatbot solutions can reduce customer service costs by as much as 30% [Reddy].

In conclusion, if two of the company's premises are to maximize profit and customer satisfaction, and additionally it is willing to invest on Research and Development, this research should represent a good foundation.

1.2 OBJECTIVES AND GOALS

To transform this project into a successful work, it needs a clear list of intentions. Also, to achieve success it is needed an objective direction of what is expected to do. Therefore, it was defined the following goals for this project:

- Better understanding of the interaction between humans and machines;
- Exploration of Machine Learning state of the art solutions;
- Exploration of Natural Language Processing and Natural Language Understanding;
- Development of a tool that has the ability to establish a bond with a Human being and formerly answer possible doubts or desires;
- Exploration of relevant actions that could be automatized by an informatics tool;
- Replacement of a Human-being or an old-fashion FAQ page with the developed tool.

1.3 PLANNING

The following figures 1 and 2 illustrate the work plan undertaken in this project.

	Name	Start	Finish
1	**Dissertation**	**10/15/18, 8:00 AM**	**7/31/19, 5:00 PM**
2	Study of the involved technologies	10/15/18, 8:00 AM	11/16/18, 4:00 PM
3	Machine Learning study	10/15/18, 8:00 AM	11/14/18, 4:00 PM
4	Experiment diferent platforms/solutions	11/14/18, 4:00 PM	12/23/18, 12:00 AM
5	Build FIrst prototype	1/3/19, 8:00 AM	1/10/19, 4:00 PM
6	Improve/ extend first prototype	1/10/19, 4:00 PM	2/2/19, 12:00 AM
7	deploy chatbot in QA environment	1/10/19, 4:00 PM	2/2/19, 12:00 AM
8	Add first feature to the chatbot	1/31/19, 8:00 AM	4/1/19, 4:00 PM
9	Implement UI	2/5/19, 8:00 AM	6/5/19, 4:00 PM
10	Add second feature to the chatbot	3/30/19, 8:00 AM	4/19/19, 4:00 PM
11	Deploy Chatbot in production environment	6/5/19, 4:00 PM	7/6/19, 12:00 AM
12	Evaluation of the work done	6/5/19, 4:00 PM	6/16/19, 12:00 AM
13	Write the document	11/15/18, 8:00 AM	7/31/19, 5:00 PM

Figure 1.: Gantt's Diagram table

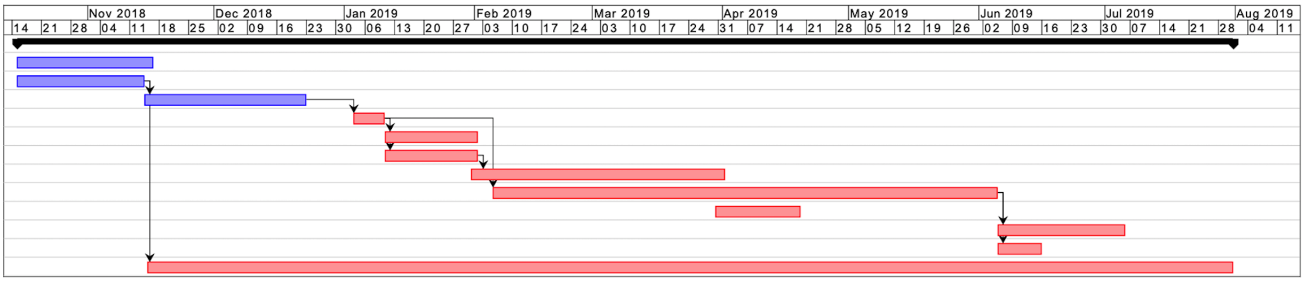

Figure 2.: Gantt's Diagram chart

2

STATE OF THE ART

This chapter will approach the conducted research according to the theme of the existing conversational interfaces and some relevant models used ”inside” these devices to accomplish astonishing voice-driven Chatbots also called ”Intelligent Virtual Assistants” or ”Virtual Personal Assistants”.

Initially research was based on Machine Learning. Some general concepts were analyzed on several research articles and books [Domingos (2012)] [Sordoni et al. (2015)] [Mitchell (1997)] .

In the following part, the research was about the existing *Web* Chatbots and Virtual Personal Assistants. It is possible to conclude that these assistants are life’s facilitators due to their useful capabilities and the ease of use. Those capabilities are possible using natural language directed by the human to the personal assistant. This research was made analyzing several related works and experiencing real-life situations Chatbots available on the *Web* and also testing those devices [Macaza (2017)][Bhagwat (2018)][Coheur (2015)][Vinyals and Le (2015)].

The latter phase consisted of analyzing and testing several tools on the market to build a proper Chatbot [Kurilchik (2017)] [Macaza (2017)] [Coheur (2015)] [Miguel and Ferreira (2017)] .

The following pages will present relevant methods and solutions related to the target goal.

2.1 NATURAL LANGUAGE PROCESSING

In this section the document will explain to the reader Natural Language Processing (NLP) and other related concepts.

- **Natural Language Processing** (or NLP) is an area of research that explores how computers process and manipulate natural language. Natural language can be divided in two categories, text and speech [De Feo and Hindriks (2016)]. Natural Language researchers are responsible for the design of language and acoustic models to be used in speech recognition systems and similar applications [Stylegar and Grimm (2003)].

  Many of the applications include a number of fields of study like speech recognition, artificial intelligence, expert systems and multilingual/cross language processing [De Feo and Hindriks (2016)].

- **Natural Language Understanding** is a subtopic of Natural Language Processing that studies the intention of sentences, it is also the study of the prediction of the elements of communication that lie outside the conceptual content of a sentence. Deals with problems concerning machine reading comprehension.

  NLU It is based on a set of "rules". It has a conceptual base that consists of a formal structure, can make predictions based on that conceptual structure, it is not limited to the understanding of isolated sentences and has formal rules for analyzing natural language utterances into the conceptual base [Schank (1972)].

2.2 MACHINE LEARNING

In this section the document will explain to the reader Machine Learning (ML) and other related concepts.

Machine Learning is the study of automatic performance improvement algorithms. The most fundamental idea behind Machine Learning is generalization [Domingos (2012)].

Since Machine Learning studies the generalization necessary for applying learning algorithms on data to build a model, it seems reasonable to ask ourselves "How does learning performance vary with the number of training examples presented?" or "Which learning algorithms are most appropriate for various types of learning tasks?"[Mitchell (1997)].

The following concepts will clarify the reader about some concepts that are inherent to Machine Learning.

- The **hypothesis space** is the set of all the evaluating functions that can represent the model [Mitchell (1997)]. For instance, if the problem we are dealing is a classification problem, the hypothesis space should be the set of all possible classifiers that could represent the model. However, not all of the functions can accurately represent the model, i.e. the true hypothesis/unknown function.
- A **feature** is a specification of the object in the hypothesis space. Each input variable, $\{X1, X2, ... Xn\} \in X$ represents a feature value of an object X. These values are usually aggregated in a vector whose size is determined by the number of properties of that object [Sebastiani (2002)]. For instance, if our data-set was composed by patients, gender, age or blood type could be examples of features from those patients.
- *"**Supervised learning** is the most common form of ML. Usually the desired result is a mapping from problem instances to a set of class values. A training set that contains examples of problem instances along with their desired class label are given to the system. The task now is to take the training set and use it to construct a generalized mapping that can label the instances correctly but, in addition, that can label other unseen examples correctly too. An example of such a problem can be found in direct marketing. Let us assume a company has much information about its customers, for example buying habits, living environment, age, income and so on. Based on previous experience on which customers respond to prospects the company sends out, a learning algorithm could use a relatively small set of customers to learn a mapping that classifies customers into responsive and non- responsive" [Otterlo (2010)].*

- **Classification** is the process of determining the class of an object using a classifier. The method consists of approximating a mapping function (f) from input values/features to discrete symbols (Y), [Otterlo (2010)]. A good example of classification usage is determining the genre of texts using features found on texts [Frost (2010)].

- *"A **classifier** is a system that inputs (typically) a vector of discrete and/or continuous feature values and outputs a single discrete value, the class" [Domingos (2012)].*

- **Conditional Random Field** or CRF is a framework for building probabilistic models to segment and label sequence data. CRF avoid a fundamental limitation of Maximum Entropy Markov models (MEMMs) and other discriminative Markov models based on directed graphical models, which can be biased towards states with few successor states [Lafferty et al. (2001)].

- **Support Vector Machines** or SVMs are classification algorithms that have been shown to be highly effective at traditional text categorization. The core idea behind the first procedure, i.e the training procedure is to find an hyperplane that is represented by a vector $\overrightarrow{W}$, that separates the document vectors in one class from those in other class by a *margin* as large as possible. It corresponds to a constrained optimization problem; letting $c_j \in \{1, -1\}$ be the correct class of document d_j, the solution can be written as:

$$\overrightarrow{W} = \sum_j a_j c_j \vec{d}_j, a_j \geq 0,$$

  where the a_j's are obtained by solving a dual optimization problem. Those $\vec{d}_j$ such that a_j is greater than zero are called support vectors, since they are the only document vectors contributing to $\overrightarrow{W}$. Classification of test instances is simply determining which side of $\overrightarrow{W}$'s hyperplane they belong [Frost (2010)].

- **Hidden Markov Models** or HMMs were built with the a generalization of mixture models [Ghahramani and Jordan (1997)]. HMMs are composed with a set of *States*, a set of *Observables*, a *transition probabilities* table and an *emission probabilities* table. The transition probabilities table has the probabilities of transitioning from a state to the same and to all the others states. The emission probabilities table has the conditional probabilities of being in a state s' given that we acknowledged an observation o', which represents P(s' |o'). Hidden Markov Models can be represented with a dynamic Bayesian network. It is to be noticed that to an external observer, the states are hidden, this is, only observations are visible to the observer [Parr (2003)]. It is possible to observe a Hidden Markov Model in figure 3.

- **Maximum Entropy Markov Model** or MEMM is a statistical model based on Hidden Markov Models (HMMs) that uses contextual features to predict/classify *Part-Of-Speech* (POS) tags. This technique is used in Natural Language Processing to accurately predict unseen text [Ratnaparkhi].

- **Artificial Neural Networks** (ANNs) are learning methods for discrete, real and vector valued functions using examples. There are several learning algorithms such as Back-propagation that use gradient descent to tune the network parameters to best fit a training set of input-output pairs.

These types of networks are among the most effective learning methods available today. ANNs have been mostly inspired by the observation of biological learning systems which are built of complex webs of interconnected neurons. Artificial Neural Networks could have more than one hidden layer of neutrons. When an ANN has more than one hidden layer between the input and the output layers are considered a type of Deep Neural Networks or DNNs [Graves et al. (2013)]. An example of a an ANN with two hidden layers can be found on figure 4.

In a coarse analogy, Artificial Neural Networks are constructed from a densely interconnected set of simple units, where each unit takes a number of real-valued inputs (that could be the output from previous units) and produces a single real-valued output that could become the input to many other units [Mitchell (1997)].

- **Recurrent Neural Networks** or RNNs are a variation of Artificial Neural Networks.

  In the case of RNNs used in language modeling, they are responsible for reading an input word x_i, and predict the next word x_{i+1}.

  Given an input sequence $x = (x_1,..,x_T)$, a Recurrent Neural Network computes the hidden vector sequence $h = (h_1,..,h_T)$ by iterating the following equations starting in $t=1$ until T:

$$h_t = H(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

$$y_t = W_{hy}h_t + b_y$$

  where the W terms stand for weight matrices, b terms are Bias Vectors and H is the hidden layer function [Graves et al. (2013)]. There are some variations of Recurrent Neural Networks. An example of a RNN can be seen in figure 5.

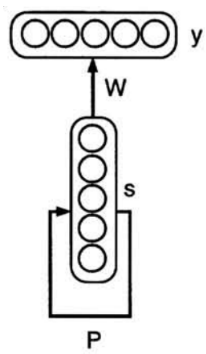

Figure 3.: Example of a Hidden Markov Model [Ghahramani and Jordan (1997)]

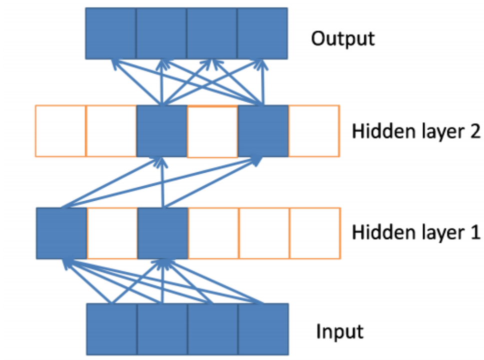

Figure 4.: Example of an Artificial Neural Network with two hidden layers [Glorot et al. (2011)]

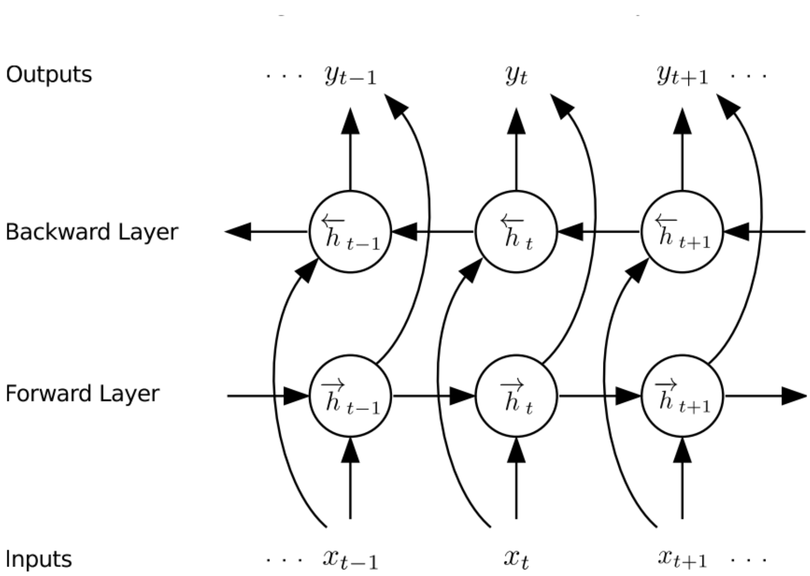

Figure 5.: Example of a Bi-directional Recurrent Neural Network [Graves et al. (2013)]

2.3 CHATBOTS

A Chatbot is a computer program that can interact with a human using natural language with textual or auditory inputs simulating a human being during the whole interaction.

In 1950, there was a brilliant mathematician, Alan Turing, the father of the Artificial Intelligence, that asked the question *"Can machines think?"*, [Turing (1950)], proposing a way of testing the intelligence of a machine, the *Turing Test*.

That test proposed that a human would judge natural language conversations between a machine designed to generate human-like responses and another human. These three participants were physically separated from each other. The judge would have to guess who was the machine, knowing from the beginning that was "talking" with a human and a machine using only a computer keyboard to type messages [Turing (1950)].

In 1966, the first Chatbot that came to public was called ELIZA, a program developed by Joseph Weizenbaum that was able to establish a conversation with human beings pretending to be one as well. ELIZA's conversational model was based in the rephrasing of input sentences, when these matched the Knowledge Base. The intent of Weizenbaum, was to simulate a Rogerian psychotherapist [Weizenbaum (1966)].

Many years after, ELIZA is still one of the most widely known applications in AI and it is a base for numerous Chatbots exceeding the expectations given that many of the users believed that they were talking to a real human-being.

In 1972, a Neuropsychiatric Kenneth Colby formed at Yale School of Medicine created PARRY, a bot which impersonated a paranoid schizophrenic person. Colby considered ELIZA as a tool capable of handling several patients per hour and PARRY as a tool to model a mental disorder, paranoia [Colby (1999)].

Since 1991, many Chatbots with different goals started to rise. That happened because Hugh Loebner, an American inventor, back in 1990, created the first annual competition in artificial intelligence that awards prizes to the computer programs considered by the judges to be the most human-like, called "Loebner Prize". The contest was held each year and consisted in attempting to run the Turing Test on every contestant. The first winner of this challenge was Joseph Weintraub, the president of Thinking Software, Inc., in Woodside, New York. His computer program fooled 5 out of 10 judges into thinking it was an human [Epstein (2009)].

It was the year of 1995, and an interesting competing system in the Loebner prize came up. It was called "ALICE" and was invented by Dr. Richard Wallace, American author of AIML (Artificial Intelligence Markup Language) [Wallace (2009)].

ALICE was based on ELIZA, but differs from it by not playing a specific role, instead it tries to reflect a human in general. ALICE won a total of three times the competition of Loebner Prize (in 2000, 2001 and 2004). AIML it is an extension of XML still used these days that provides a simple mechanism for generating responses by pattern matching [Epstein and Roberts (2009)].

Nowadays, due to the technology evolution and the constant need of quick and valid information, Chatbots are becoming a large field in expansion. It is interesting to observe the diversity of services that they are applied to. The most common uses are related to *e-commerce* and Virtual Personal Assistants. Amazon's Echo and Alexa, Apple's Siri, and Microsoft's Cortana are examples of modern Chatbots available on the market [Weinberger (2017)].

Some of these bots take advantage of the advances in the Machine Learning field to provide relevant information based on *web* searches made by the bot. Some use *APIs* (Application Programming Interfaces) to automate actions (see figure 6). Other bots have a more spontaneous generative model to respond. The latter ones, use "Statistical Machine Translation"(SMT) with Recurrent Neural Networks (RNN's) to encode and decode inputs to respond to the user [Jurafsky (2018)].

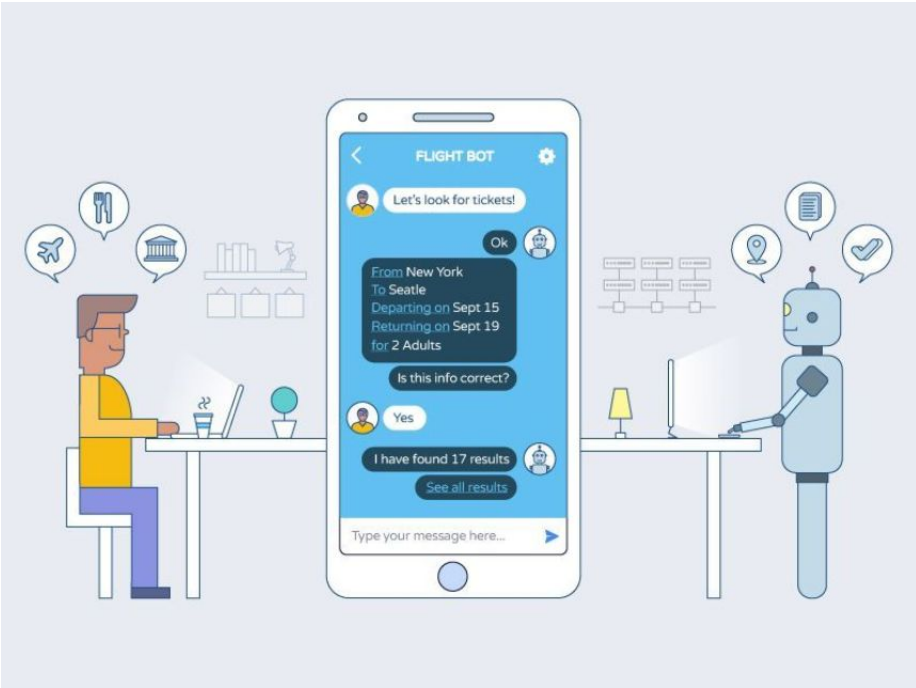

Figure 6.: Chatobts using modern API's [Ecommerce-Nation - visited at 23/07/2019]

2.3.1 *Web Chatbots*

Since our project will reside on a *website*, it seemed crucial to explore other *Web* Chatbots to have a broad vision of what are the tendencies and what features our tool should have.

The following part will present to the reader some of the relevant *Web* Chatbots found on the *Web*. We are going to do a comparison side-by-side in a table later presented.

Cleverbot

Cleverbot is a Chatbot that is available on the *Web*. It learns from the textual interaction with users. It has an API to develop a custom Chatbot or even using one of their pre-built ones. Cleverbot is not *Open-Source* and the services provided to the users related to building a Chatbot are paid [Cleverbot].

During a dialogue with a user seems to save the dialogue context. If a user asks him about something in one interaction like a person's name, and refers it in the following interaction as *"him"*, Cleverbot will respond accordingly with an answer referring to that person's name.

It has knowledge about well known facts. For instance, if the user asks him about "Donald Trump", he knows that he's a president and other factual topics.

It seems to be confused about object pronouns. For instance, when it should refer to a single person as *"him"*, it refers to single person as *"them"*. It is possible to see part of a conversation with Cleverbot on figure 7.

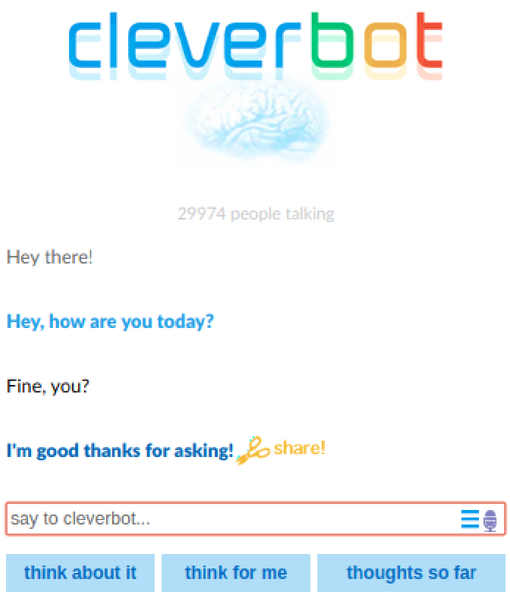

Figure 7.: Cleverbot [Cleverbot-website - visited at 05/11/2018]

Endurance

There is an Open Source project being developed by a company named *Endurance* that has great potential for scientists to better understand how Alzheimer's affects the patients' brains [Shewan (2017)].

The Endurance's Chatbot is being developed to be an universal robot-companion for older people.

The core idea is to provide to senior people and Alzheimer's patients an opportunity to communicate, stimulating their brains, by sharing their experiences, their memories or just making small talk on a daily basis [Endurance]. In figure 8 we can see an image of the Chatbot found on Endurance's landing page.

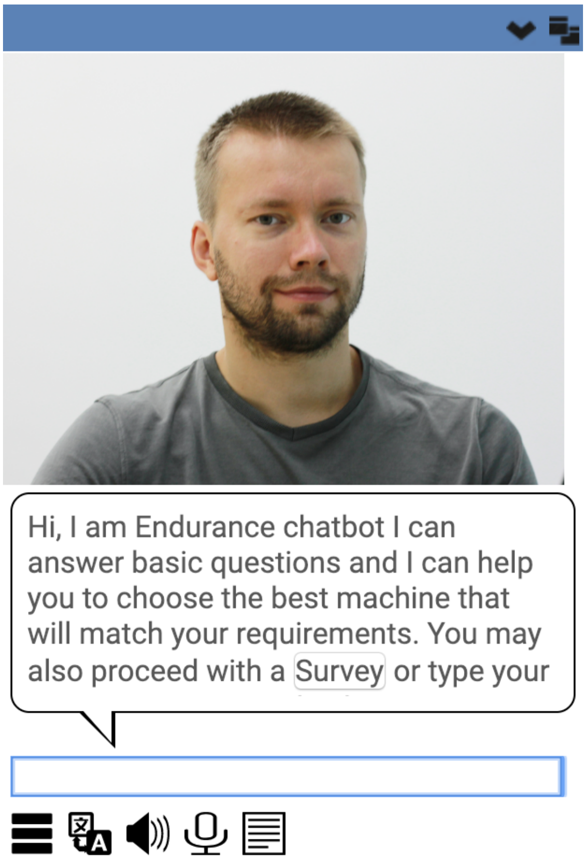

Figure 8.: Endurance's Landing page Chatbot [Endurance - visited at 05/11/2018]

Roof.ai

Roof.ai is a Chatbot developed for real-estate marketeers with the intention of generating potential leads interacting with customers via social media.

The bot uses Facebook to identify possible leads and then starts a friendly dialogue with a human-like tone to gather relevant data. After gathering the required information, it assigns the lead to a sales agent.

The bot is still under development, however Roof.ai seems to be incredibly accurate and it is likely to be interesting for real-estate marketeers [Shewan (2017)].

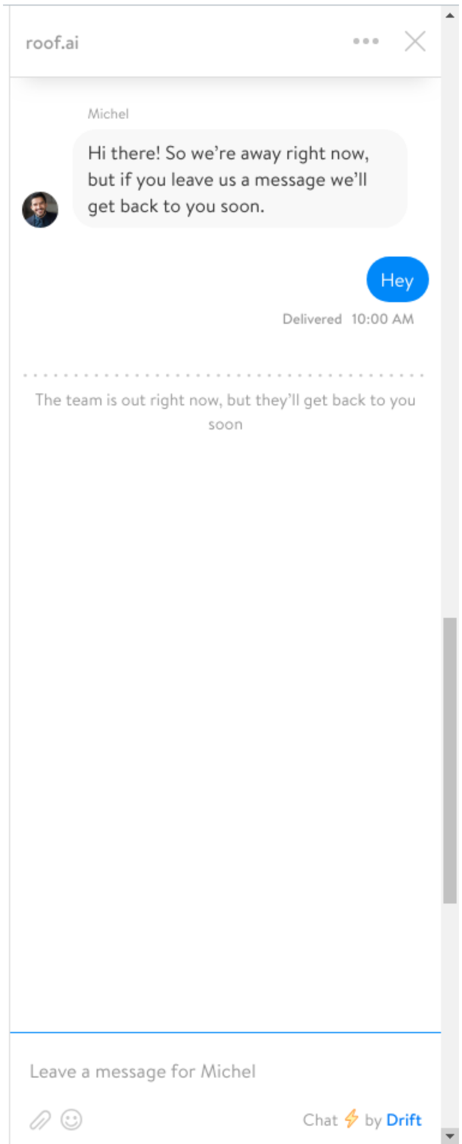

Figure 9.: Roof.ai Chatbot [Roof.ai visited at 08/11/2018]

It is possible to observe in figure 9, that this Chatbot was developed using *Drift* bots that will be presented later.

Summary

With the intention of providing a comparative analysis about Chatbots available on the *Web*, it was decided to present a table with some aspects that are visible when using the referred Chatbots.

Those aspects were chosen to perform a study regarding the User Experience in a holistic point of view. The aspects are the following: "Type of Conversation", "Relevant Feature", "Engagement" and "Overall Evaluation". The first two aspects are qualitative evaluations and the latter two are quantitative evaluations on a scale from zero to twenty.

The table 1 intends to summarize information regarding *Web* Chatbots approached in this document. It should be noticed that it is a subjective analysis.

Parameter	**Cleverbot**	**Endurance**	**Roof.ai**
Type of conversation	Fluid conversation. It has the ability to save conversation details. For example it saves who is the user.	It talks about general topics like the weather, nature, hobbies, movies, music and other.	It has a Marketing oriented conversation. Tries to gather the desired data to redirect it to a sales agent.
Relevant Feature	Autonomous learning when users interact with the bot	The Chatbot has an audible voice simulating a human.	Gathers required information in an intuitive way.
Engagement (0 - 20)	18	14	17
Overall Evaluation (0 - 20)	19	16	14

Table 1.: Comparison between different *Web* Chatbots

2.3.2 *Dialogue tools for Human Resources Services*

Since our goal is to add a Chatbot system to a Human Resource Service commercial *Website*, it seems indispensable to observe and compare dialogue systems or methodologies regarding the communication between clients and responsible company staff.

This following part will present some of the actual dialogue tools competitors. Among these tools there are forms to contact customer support, FAQs (Frequently Asked Questions) and non-autonomous Chatbots.

Non-autonomous Chatbots are characterized by a certain lack of domain that only a Customer Support employee can fill. These Chatbots use a *Decision Tree* dialogue system to gather data and frequently have to book a meeting with a customer support employee. These Chatbots have a clear advantage that can save hours of work to a company. They ask protocol questions that are strictly required. An example of these questions could be the purpose of the visit to the *website*, or they could ask questions with the purpose of gathering required protocol information like the name, professional email, or even their company's name.

Gusto - User Impression

The dialogue tool provided by the *Gusto* platform is basically a FAQ page to answer frequent questions. It is complemented with a form to respond to non-approached topics to provide a full domain of the knowledge.

It seems to present less engagement to the customer in comparison with a dynamic tool like a Chatbot. It is possible to observe the dialogue tool presented by *Gusto* in figure 10.

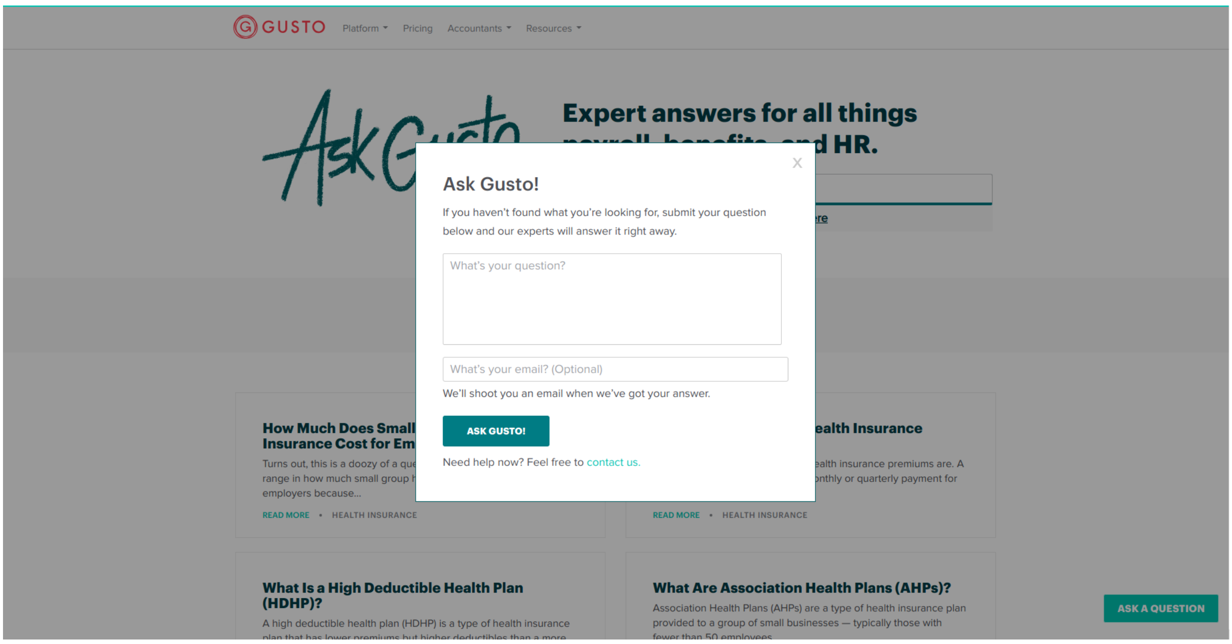

Figure 10.: Gusto [Gusto - visited at 06/12/2018]

Zenefits - User Impression

Zenefits seems to provide a more engaging dialogue tool for customer support. This tool could be called as a "non-autonomous" Chatbot, because it can only schedule a meeting. The bot asks the *email address* to contact the customer and afterwards presents a *decision tree* with the availability of the customer support worker. If the customer support worker is available, it starts the dialogue right away. In figure 11 it is possible to observe Zenefits' Chatbot on the right side.

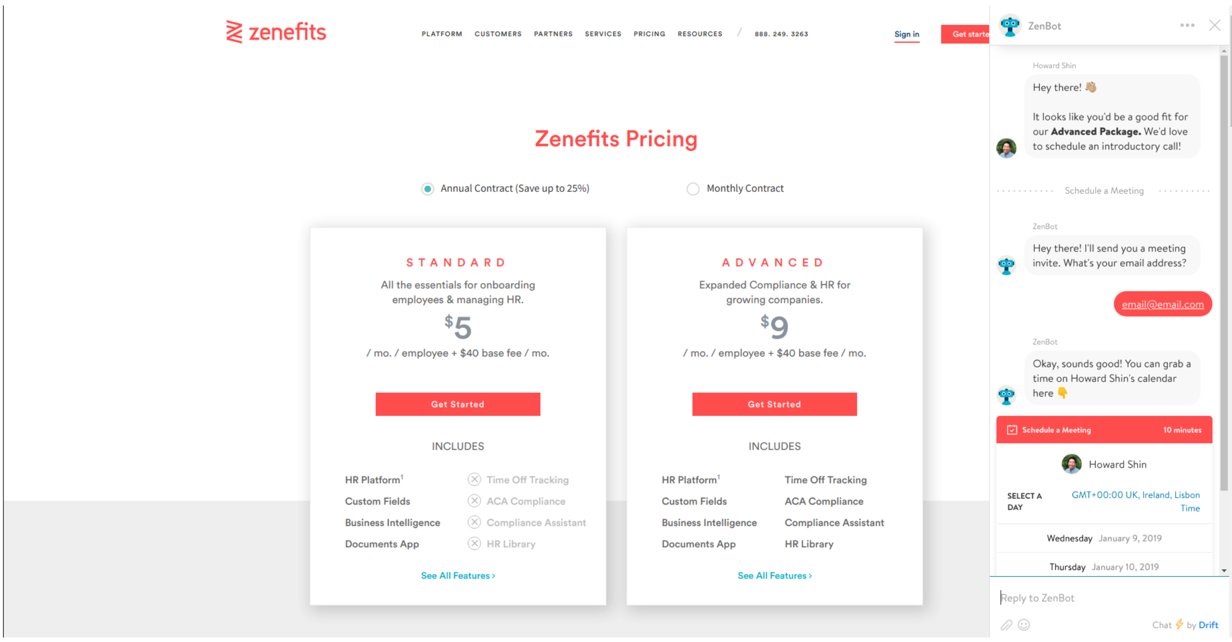

Figure 11.: Zenefits [Zenefits - visited at 23/12/2018]

Zoho - User Impression

Zoho dialogue tool seems to be more conservative. It provides a FAQ page complemented with a customer support team to clarify doubts uncovered by the FAQ section. The presented Chatbot is autonomous, i.e. doesn't have a human interaction. It seems an effective method to approach this task. However, it could have more engagement with the user. We can see the FAQ page and Zoho's Chatbot in figure 12.

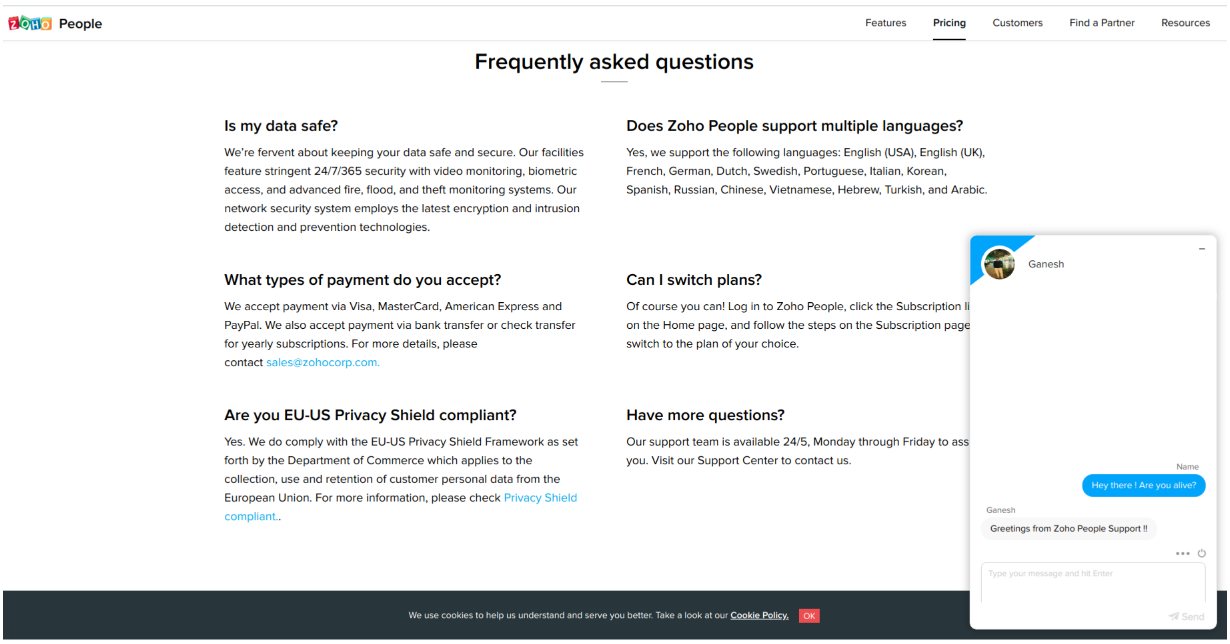

Figure 12.: Zoho [zoho - visited at 23/12/2018]

Summary

The idea behind this section is to provide an analysis on the communication tools available on similar Human Resources Services software.

These dialogue tools provided by the Human Resources Services *Websites* were evaluated by certain criteria to provide a comprehensive analysis when comparing them side-by-side. These criteria were the following: "Attributes", "Relevant Feature", "Engagement" and "Overall Evaluation". The first two parameters are qualitative evaluations and the latter two are quantitative evaluations on a scale from zero to twenty.

The table 2 purpose is to summarize information regarding dialogue systems present on commercial *Websites* of Human Resources tools approached in this document. It should be noticed that it is a subjective analysis.

Parameter	**Gusto's dialogue tool**	**Zenefits' dialogue tool**	**Zoho's dialogue tool**
Attributes	It has a FAQ page. Doesn't provide a Chatbot.	The Chatbot works as a Tier 1 agent and provides an interface for a human agent.	The Chatbot available knows relevant topics regarding the product to be sold. It complements the FAQ page.
Relevant Feature	None	Schedule a meeting	None
Engagement (0 - 20)	10	16	15
Overall Evaluation (0 - 20)	10	17	16

Table 2.: Comparison between different dialogue/communication systems available on Human Resources Services

2.3.3 *Virtual Personal Assistants*

It was decided to approach Virtual Personal Assistants (VPA's) for two reasons. The first reason is because it has similar features to an Electronic Commerce Chatbot, since it establishes a dialog between a human and a machine using textual or voice methodologies. The second reason is due to a Chatbot Framework (*Dialogflow*, former *API.ai*) that was based on a Virtual Personal Assistant.

In the last decade, the Chatbot world had a major impact in our daily routine. For instance, the newspapers reading were replaced by a simple voice request directed to a Virtual Personal Assistant (VPA) that will tell the news with a imitation of a human voice. The restaurant reservation phone call has been changed to another voice request redirected to the restaurant's calendar or to a phone call between a human and a VPA pretending to be a human. These examples are one of the endless type of appointments that today is being done with cutting edge technology.

For a proper analysis an investigation was carried out by trying several Virtual Personal Assistants (VPAs) in order to collect and summarize their differences.

The table that will be presented later aims to compare side-by-side some relevant features concerning several **voice Chatbots** integrated in physical devices. The following part intends to demonstrate these VPA's referred below in a comparative way. The details chosen were based on specifications like the user-agent voice change according to the type of the dialogue between the Chatbot and the user, or the availability on different devices and other aspects.

Amazon Echo

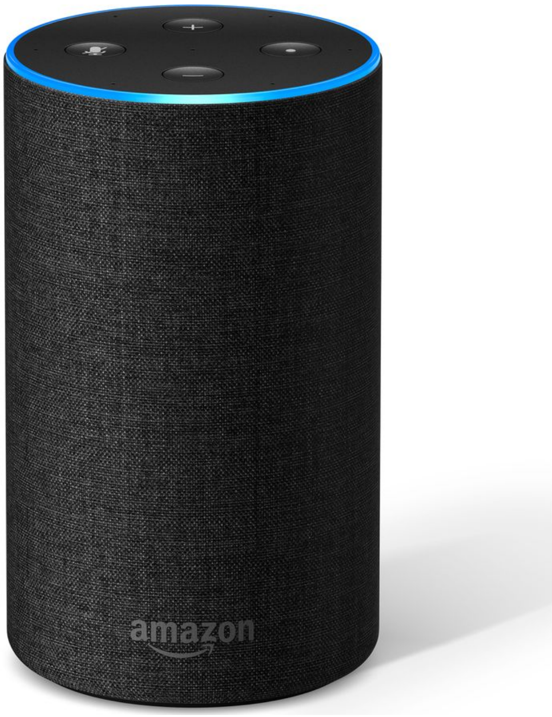

Figure 13.: Example of an Amazon Echo device [Amazon (b) - visited at 26/07/2019]

Amazon Echo is a brand of smart speakers developed by Amazon. There is a voice-controlled intelligent personal assistant named "Alexa" which connects to "Echo devices". Users can call her "Alexa" but can change this wake word to "Amazon", "Echo" or "Computer" [Amazon (a)].

The device's main features are the following: voice interaction, making shopping lists, music playback, making to-do lists, setting up alarms, streaming pod-casts, and playing audiobooks, in addition to providing weather, traffic and other real-time information, it can also control several smart devices, acting as a home automation hub [Amazon (a)].

In the default mode, Amazon Echo continuously listens to all speech, monitoring for the wake word to be spoken, which is primarily set up as "Alexa". The device's microphones can be disabled by pressing a mute button to turn off the audio processing circuit [Bloomberg]. We can observe an Amazon Echo device on figure 13.

Google Home/Assistant

Figure 14.: Example a Google Home device [Google visited at 26/07/2019]

Google Home devices are speakers that enable users to speak voice commands and interact with the services through the Google's intelligent personal assistant named Google Assistant [Techradar (2019)].

There is available an enormous number of native and third-party apps. The main features are the following: play music, receive updated news, control playback of video and photos using only the voice to dictate orders [Techradar (2019)].

These devices also have integrated a support for home automation that provides voice activated smart home appliances [Techradar (2019)].

One of the latest updates, brought by Google in May 2017, enables some functionalities like: setting reminders on *Google Calendar*, making appointments, hands-free phone calling and other interesting features [Palmer (2017)]. We can see an example of a Google Home device in figure 14.

Siri

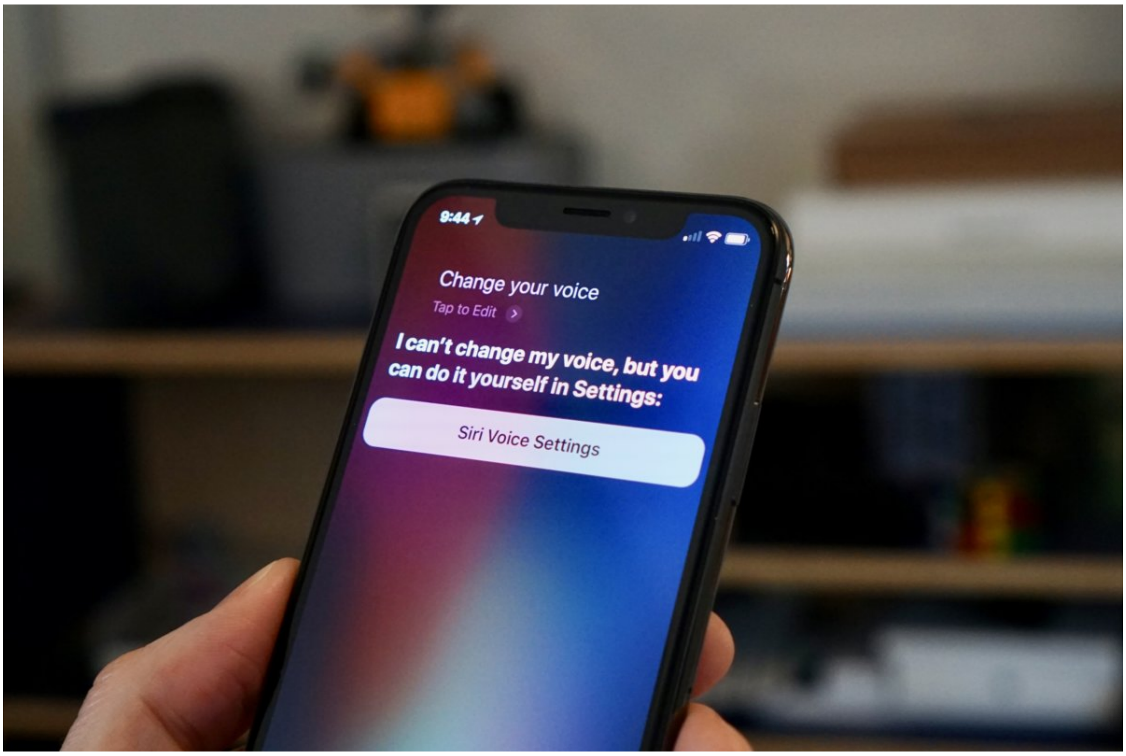

Figure 15.: Siri [Jason Cross - visited at 26/07/2019]

Siri is a Virtual Personal Assitant (VPA) that is part of Apple's Operating Systems such as macOS, iOS and other operating systems built-in Apple's IoT's (Internet of Things) devices. The VPA is capable of making recommendations, answering questions and performing actions by using several Webservices *APIs* when the user issues a voice query using natural language [Bosker (2013)].

Siri is a project originally developed by SRI International Artificial Intelligence Center. It uses a speech recognition engine provided by Nuance Communications and an advanced Machine Learning model to function properly [Bosker (2013)].

The VPA supports a variety of user commands including phone call actions, scheduling events and reminders, making reservations, handling device setting, searching the internet and several interesting features [Purewal and Cipriani (2017)]. It is possible to observe an iPhone device whose user is currently using Siri in figure 15.

Cortana

Figure 16.: Cortana [Matt Mcgee visited at 26/07/2019]

Cortana is a Virtual Personal Assistant (VPA) created by Microsoft for its Operating Systems, i.e., Windows 10, Windows 10 Mobile, Windows Phone 8.1, etc [Foley (2014)].

It is capable of setting reminders, recognizing natural voice and answering questions using the Bing Search engine *API*. [Sams (2015)]

This VPA is currently available with the following languages: Portuguese, English, German, French, Spanish, Italian, Chinese and Japanese [Microsoft].

Cortana appeared for the first time at Micrsoft BUILD Developer Conference that took place in San Francisco in April 2014 [Lau (2014)]. We can observe a Windows phone whose user is currently interacting with Cortana in figure 16.

Summary

In order to compare new trends on dialogue tools we decided to do a research on different platforms known as Virtual Personal Assistants and evaluate them side-by-side.

To accomplish the desired evaluation we decided to choose relevant parameters. Those were the following: "Waiting time", "Type of conversation", "User agent voice change", "Available on different devices", "Making a reservation".

The table 3 intends to summarize information regarding Virtual Personal Assistants approached in this document. It should be noticed that it is a subjective analysis.

Parameter	**Amazon Echo**	**Google Home**	**Siri**	**Cortana**
Manufacturer	Amazon	Google	Apple	Microsoft
Waiting time	5 secs	2 secs	5 secs	3 secs
Type of conversation	It is quite limited. Sometimes there is no proper voice recognition. Alexa (Amazon Echo's agent) specifically answers to the users interactions.	There is more dialogue between the user and the Google Assistant (the Google Home agent). The agent interacts even if the dialogue is rhythmically paused.	The agent specifically answers to the users interactions. It only answers to the voice of the user. There is no voice change according to the theme of the conversation.	It is quite limited too. Sometimes there is no proper voice recognition. The agent specifically answers to the users interactions.
User agent voice change	No	Yes	No	No
Available on different devices	No	No	Yes	Yes
Making a reservation	Yes	Yes	Yes	Yes

Table 3.: Comparison between different Virtual Personal Assistants

2.4 Artificial Intelligence Markup Language

This section should provide an analysis of the AIML language as an option to develop the desired Chatbot. AIML or Artificial Intelligence Markup Language is a Markup Language based on XML designed between 1995 and 2002 by Richard Wallace, initially to extend ELIZA into *ALICE*, Artificial Linguistic Internet Computer Entity.

This technology allows a developer to create a natural language software agent using an abstraction of Regular Expressions called AIML Elements. The most important AIML Elements are the following:
- **Categories:** A category consists of at least two elements, a pattern and template.
- **Patterns:** A pattern is a String to match one or more user inputs.
- **Templates:** A template is intended to be the response to a Pattern. It can reference variables.

The following example was elaborated to give a little immersion in the AIML language as well as expressing the interaction between a human and a Chatbot designed with this markup language .

```
<category>
  <pattern>WHAT IS YOUR NAME</pattern>
  <template><![CDATA[My name is <bot name="name"/>.]]></template>
</category>
```

These rules represent the Chatbot Knowledge Base. In this research it was possible to experiment some Chatbot AIML interpreters written in languages like *Java* or *PHP*. It is possible to conclude that most of them are similar and do not stand out very much from each other.

2.5 Chatbot Developing Platforms/AI Chatbot Frameworks

With the increased need of developing Chatbots for commercial and informative use, platforms that facilitate the creation of a Chatbot from scratch have risen. These platforms were created by big name companies like Google or Facebook with the perspective of automatically deploying the "generated Chatbot" to several platforms like the chat of some company's Facebook page for such tasks as providing information of facilitating the acquisition of goods or services.

The Chatbot's domain is defined using *Entities* and *Intents*. An *Entity* is a powerful tool used for extracting parameter values from natural language inputs. And an *Intent* represents what the user is trying to achieve.

According to most of these tools' communities, it is good practice to provide several examples of the same *Entity* in every *Intent*.

Chatbot Developing Platforms presented ahead are responsible for the resolution of two problems.

The first problem is a classification problem and can be solved with different approaches. This classification is related to the identification of the user's *Intent* when prompting the system.

The model used depends on the amount of data and the number of intents that the domain requires. In these tools the user can not choose the model, instead it is likely that the model is chosen automatically by the tool. The most commonly used model for solving this problem is *Support Vector Machines (SVM)* [Mctear (2018)].

The second problem is colloquially named as Slot Filling. It is a task that is responsible for parsing the strings that mainly correspond to the predicate part of a sentence and put them in the right slot.

For example if the user prompts: *"How much costs your platform?"*, the agent has to map the following words: "costs" and "platform" to an *Intent* and *Entity* respectively. A suited model for this task could be *Conditional Random Fields (CRF)* [Mctear (2018)].

The "Slot Filling" problem is also being solved using Recurrent Neural Networks. [Yang and Liu (2015)].

Knowing that *"In ML terminology, the classification problem is an activity of supervised learning, since the learning process is "supervised" by the knowledge of the categories and of the training instances that belong to them"*, [Sebastiani (2002)] and given the previously mentioned facts it seems reasonable to presume that it is an ensemble of a *Rule-based* and a *Supervised Learning* classification problem.

To conclude, the assumption above is not a guarantee but a suspicion of the methodologies used in the Chatbot development tools.

2.5.1 *Dialogflow*

Dialogflow is the platform developed by Google to achieve an intelligent Chatbot using a user friendly UI. Formerly known as **API.ai** originally developed by *Speaktoit* acquired by *Google* in 2016.

This tool was based on the Virtual Personal Assistant named *"Assistant"* developed by *Speaktoit* that used *Natural Language Understanding* (NLU) to interact with its users.

Since *Dialogflow* is not *Open-Source* we can not know for sure what type of Machine Learning is happening within it. But there is some tweaks that can improve the performance of the agent:

1 - Match Mode - The user can choose between **ML only** and **Rule-based and ML**;
2 - ML Classification Threshold - Defines the threshold value for the confidence score. If the returned value is less than the threshold value, then a fallback intent will be triggered or, if there are no fallback intents defined, no intent will be triggered.
3 - Training Set - Adding or removing phrases to the agent;
4 - Validation Set - It is recommended by the *Dialogflow* community to repeat phrases or words to induce the learner to tune the model hyper-parameters;
Dialogflow defines a large group of concepts that makes its gears turn smoothly. It uses entities, intents, actions, contexts along with machine learning for training the intended (classification) model. It has built-in knowledge about some topics off the shelf like casual talk and weather for instance. So there is no need for the user to train the agent for these topics.

This platform enables the user to create "Agents" that are individual customizable Chatbots. Entities and Intents can be added to the Agent in order to improve its Knowledge Base and conversation domain.
"*Dialogflow* allows follow-up intents to be added to any parent intent and provides several built-in follow-up intents that address the most common use cases: fallback (for queries that the Chatbot cannot answer), yes/no, later, cancel, and custom." [Mctear (2018)].
Dialogflow has an ability to extend its knowledge using *fulfillment* [Ho (2018)]. With this ability it extends queries using a *Backend* that communicates via *Http* requests with the *Dialogflow* platform.

A huge advantage of *Dialogflow* is its *Pre-built agents*. *Pre-built agents* are a knowledge collection and structured data that are always ready to be added to an agent. The Pre-built agents include common verbs and context types. There are some pre-built Chatbots for different purposes like buying movie tickets, currency conversion, food delivery and many others.
Another useful feature in *Dialogflow* is *Small Talk*. It has several customizable modules regarding the type of conversation that the user is having with the bot. "About the agent", "Courtesy" and "Emotions" are one of the few customizable *Small Talk* modules that can improve the dialogue between the user and the agent. It is possible to observe a glance of the *Dialogflow's* console in figure 17.

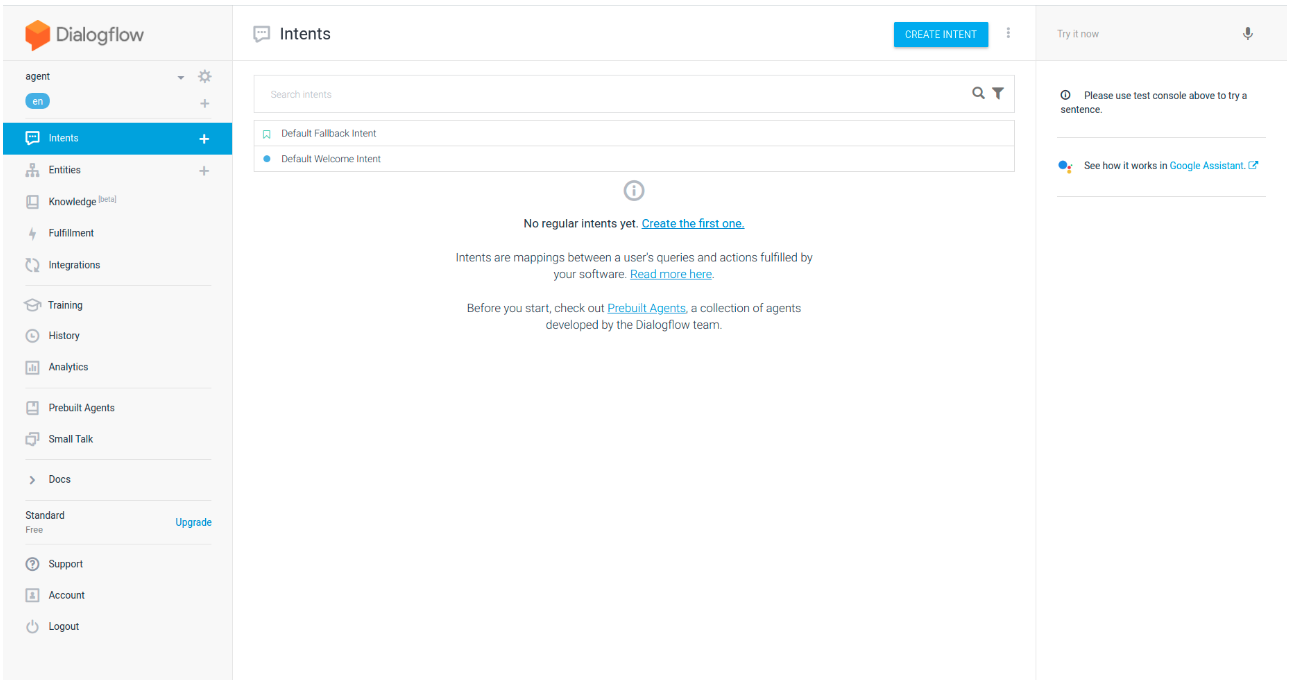

Figure 17.: Dialogflow console [Dialogflow - visited at 26/07/2019]

2.5.2 *Wit.ai*

Wit.ai is an *Open-Source* service acquired by Facebook in 2015 that supplies a combination of speech recognition and machine learning to developers. Wit.ai has a feature that allows users to create their voice commands. Those commands are recognized and converted to JSON (*JavaScript Object Notation*) [Wit.ai Team].

Just like *Dialogflow*, Wit.ai has *Intents* and *Entities*. *Intents* represent an action to be executed. *Entities* represent a specific object, or a specific thing that the Chatbot has to know to execute an *Intent* [Wit].

The application has a feature called *roles* used to learn to differentiate entities in different contexts. It has special entities to recognize URLs, emails, duration, temperature and other special words.

Wit.ai, just like *Dialogflow*, has a console to create and control the Chatbot. A perspective of the console can be observed in figure 18.

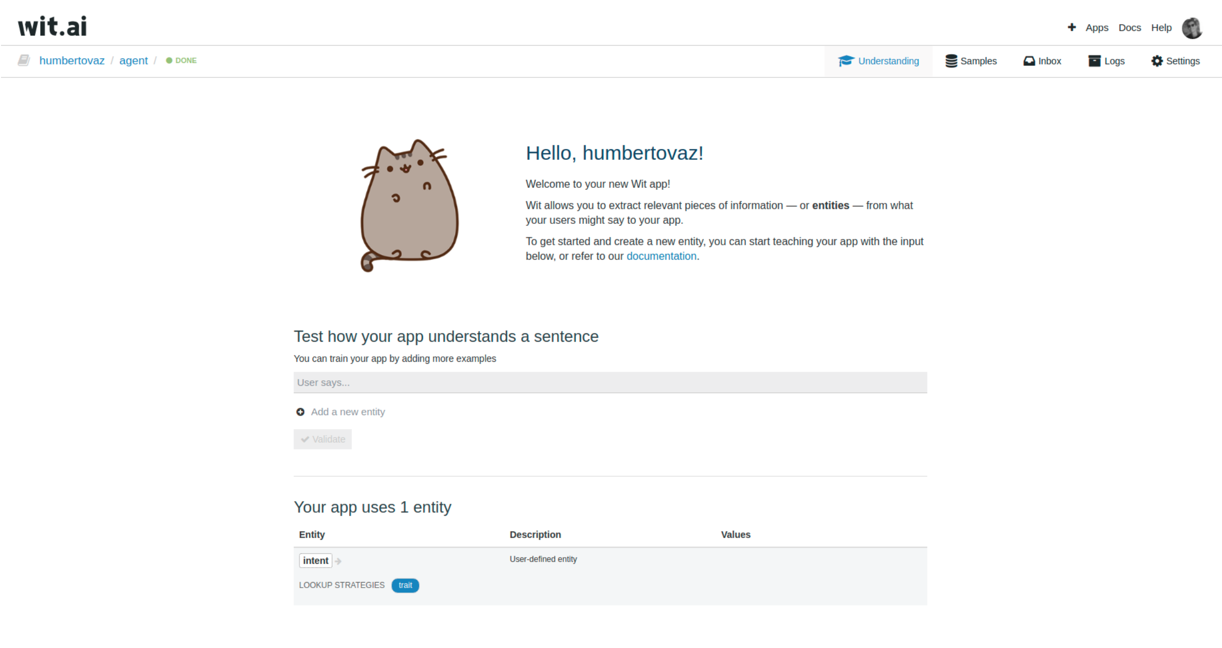

Figure 18.: Wit.ai console [Wit.ai - visited at 24/07/2019]

2.5.3 *IBM Watson*

IBM's Watson is a tool capable of providing to the user a favourable Chatbot development environment. It is capable of answering questions in natural language. This tool was developed in IBM's DeepQA project [Ferrucci et al. (2013)]. Figure 20 should illustrate the pipeline of procedures from the question asked by the user to the answer provided by the Chatbot. This tool provides a console to control the Chatbot agent. It is possible to observe IBM Watson's console in figure 19.

When created, IBM stated that, *"more than 100 different techniques are used to analyze natural language, identify sources, find and generate hypotheses, find and score evidence, and merge and rank hypotheses"* [IBM].

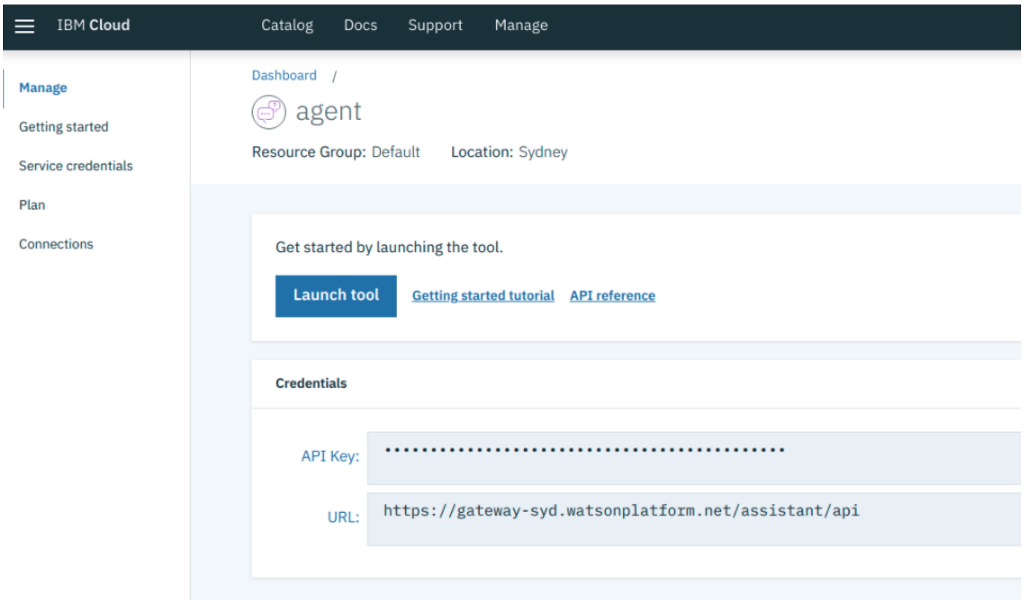

Figure 19.: IBM Watson console [IBM Cloud - visited at 26/07/2019]

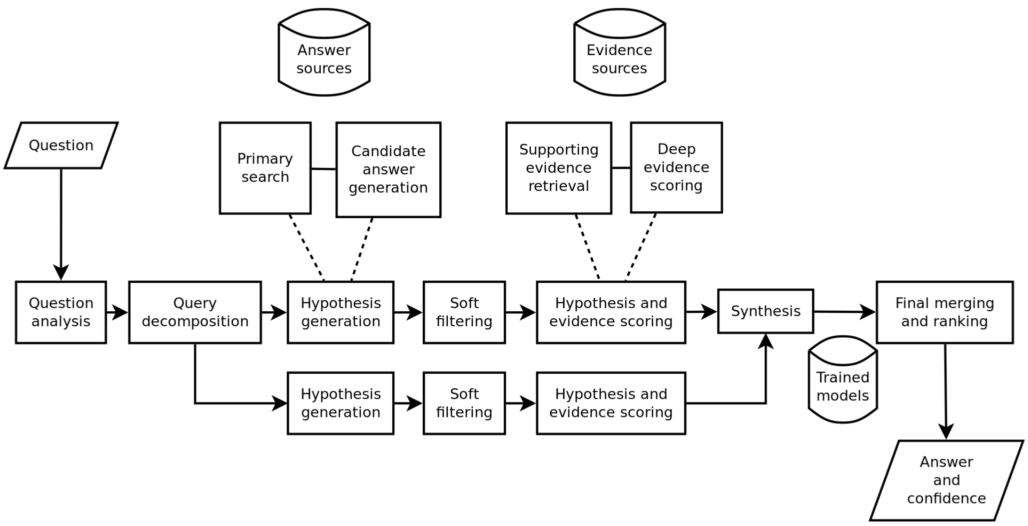

Figure 20.: IBM's DeepQA [Zhang (2011)]

2.5.4 *Leadbot*

Leadbot is a product developed by a company called *Drift* that provides tools for developing Chatbots for companies' *websites*. The final product will consist of a custom Chatbot built for the company's purposes. It provides a dialogue based on a *decision-tree*. When the dialogue falls off the bot's Knowledge Base's domain, it books meetings with Company's workers. This feature is known as *ticket creation*. There are pre-built Chatbots available off the shelf and there is an option for building a custom Chatbot too. Those pre-built Chatbots are designed to cover a variety of use cases and they are named *Playbooks* [Drift]. We can see part of a conversation with LeadBot in figure 21.

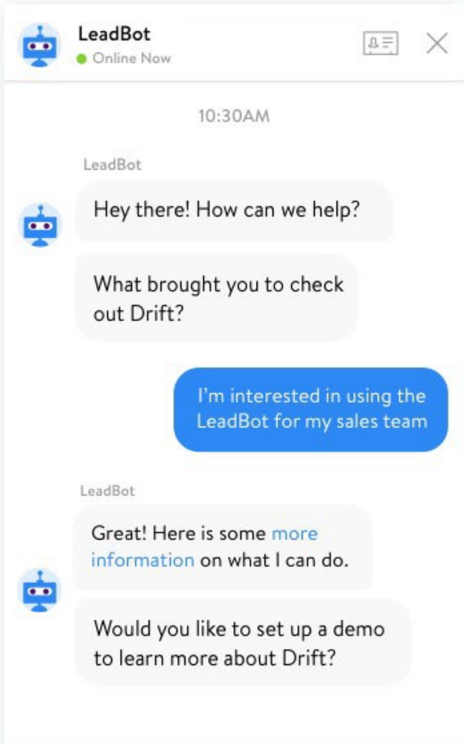

Figure 21.: Leadbot [Drift - visited at 26/07/2019]

2.5.5 *Summary*

The content of the table 4 summarizes the tools available on the market to develop a Chatbot.

Parameter	Wit.ai	IBM Watson	Dialogflow (API.ai)	Pure AIML	Driftbot
Manufacturer	Facebook	IBM	Google	None	Drift
Type	Chatbot Framework	Chatbot Framework	Chatbot Framework	Rule-based	Rule-based
Open Source	Yes	No	No	Yes	Yes
Pros	Free	Automated predictive analysis; Multilingual (10 languages).	Pre-built base agents; SDK supports several programming languages; Multilingual (15 languages).	The Chatbot's behaviour is much more predictable in comparison with ML alternatives.	The Chatbot's behaviour is much more predictable in comparison with ML alternatives.
Cons	The learning curve seems to be steeper.	Expensive comparing to Dialogflow.	Unsatisfactory documentation.	Can't answer questions that are not preconceived in the Chatbot's domain.	Can't answer questions that are not preconceived in the Chatbot's domain; It will book a meeting frequently with a Company worker; Need Customer Support.
Usage limit	No strict limit	10.000 messages/month	180 messages/minute	No	No
Paid	No	Freemium	Freemium	No	Yes
Learning Curve	8	7	6	9	5

Table 4.: Comparison between the different Chatbot developing tools

2.6 GLOSSARY

The following glossary present on table 5 has an intent of clarifying the reader for some general concepts approached in this document.

Concept	Definition
Artificial Intelligence (AI)	Intelligence demonstrated by machines, in contrast with the natural intelligence displayed by humans and other animals.
Machine Learning	Field of Artificial Intelligence that focuses on the ability of machines to receive a dataset and learn for themselves instead of being programmed.
Class	Final Prediction/label.
Classification	Process of predicting a class.
Classifier	Entity that executes classification.
Feature	Attribute or quality.
Support Vector Machines	Classification algorithm.
Artificial Neural Network	Computing system based in the biological neural network.
Recurrent Neural Network	Variation of Artificial Neural Network computing system.
Dialogflow	Google's Chatbot Framework.
Wit.ai	Facebook's Chatbot Framework.
IBM Watson	IBM's Chatbot Framework.
API	Application Programming Interface.
REST	Representational State Transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services.
Liquibase	Open-Source database-independent library for tracking, managing and applying database schema changes.
Hibernate	Object-relational mapping tool for the Java programming language.
JPA	Java Persistence API.
Spring Boot	Application Framework and inversion of control container for the Java platform.
Kotlin	Programming language.
Java	Programming language.
NGINX	Reverse Proxy.
PostgreSQL	Object-Relational Database system.
HTTP	HyperText Transfer Protocol. This protocol is used all over the web.

Table 5.: Glossary

3

THE PROBLEM AND ITS CHALLENGES

This chapter is intended to expose to the reader the decisions and the problems found developing the solution.

The solution found was based on some aspects, such as time available, learning curve of the chosen technologies, and others.

After this research, it was decided that the Chatbot model used should be a Rule-based model with Machine Learning since our main requirement is to satisfy doubts related to the product to be sold on the commercial *Website*. In order to make this happen, a proper Chatbot framework was chosen to be able to retrieve an interaction by typing directly on the console.

3.1 SYSTEM ARCHITECTURE

The intended architecture for this system is composed by three distinctive parts. The *client-side* app, the *server-side* app, and the "*Chatbot-side* app".

In order to achieve a stable environment it was chosen to use containers in all the pieces of this project. All the components illustrated in figure 22 are **Docker containers**.

The system uses three apps that have different purposes. The *client-side* app is supposed to supply the user interface for the end user. Also provides an entry to a secure channel with the function of exchanging messages between the user and the Chatbot itself using a server app. The *server-side* app or just server app is supposed to receive messages, deliver them to the Chatbot, receive its response and finally hand over the message to the user. Finally, the Chatbot app, will receive messages, process them and respond accordingly to its Knowledge Base. In the figure 22 we have a graphical representation of the system architecture.

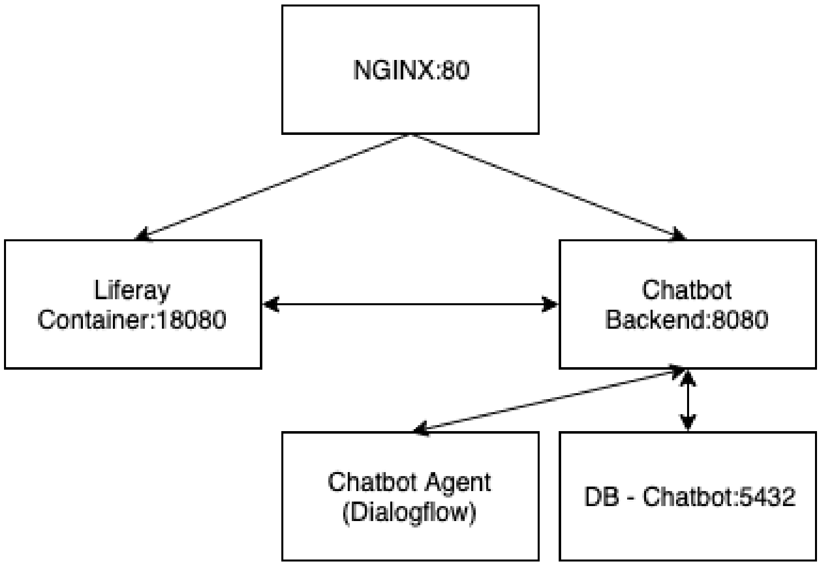

Figure 22.: System Architecture

The technology stack used in this project had to fit seamlessly in the *Website* architecture. So we had to adapt the development process to our requirements. The user interface, that is the *"Chatbot Frontend"* app would have to "live" in a *Liferay Server* system that builds *Http* requests using *Javascript* on top of a *Apache Tomcat*, however not using it for communications with the *Chatbot Backend* app since it wasn't compatible with our version of *Hibernate* (later presented).

Docker Container

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, system tools, system libraries and settings [Docker]. It is possible to observe a scheme of a Docker container in figure 23.

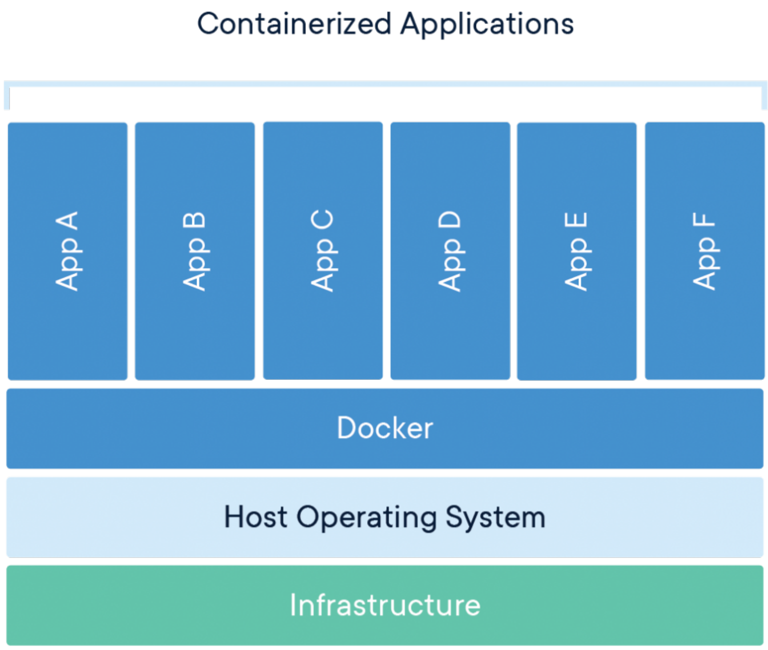

Figure 23.: Scheme of a Docker container [Docker]

Liferay

Liferay is an *Open-Source* CMS (*Content Management System*) written in *Java* whose main purpose is to supply a tool to update or modify the content of a *Website* that non-experts could use without any programming language skills. Since it is *Java* based we had to integrate the *HTML* and *CSS* code into *JSP* (*Java Server Page*) that could execute the rendering of our page on *server-side* and deploy it to the *Liferay Server* as *.war* (Web application ARchive) file.

Chatbot Backend

We could develop a system that could communicate directly between Chatbot portlet and the Chatbot agent, but it wouldn't be safe to expose sensible information like *JWT tokens* in the Chatbot portlet. In terms of the *server-side* application, we had to think about performance and security related questions. Since the used version (**v2beta1**) of the *Dialogflow* agent had several *API's* like *Node.Js*, *Python*, and *Java* regarding the preference of the programming language by the developer, it would make sense to use and develop a *Backend* in a JVM language like *Java* or *Kotlin*. *Kotlin* was chosen because it carries almost all the advantages from *Java* and several more related to the ease, quickness of development and safety related operations.

The *server-side* framework chosen was the *Spring Boot 2 Framework* with an embebed *Apache Tomcat* to help the *Kotlin* app to create a *REST API* and consequently receive messages from users, deliver them to the Chatbot agent, receive the response from Chatbot agent, and finally, hand the message to the user (Chatbot portlet).

Chatbot's Database

In order to store data related to Chatbot's user feedback we had to think in a Database to make data persistent. *PostgreSQL* was chosen since it is an *Open-Source relational database* with a wide community. This Database is going to be used to gather the user experience by classifying each message and to save the scheduled meetings to clarify possible clients about the platform that is being sold on the commercial *Website*.

Chatbot Agent

The core of this bot resides in a Chatbot Framework maintained by *Google*, named *Dialogflow*. This tool's main purpose is to detect the user intent present in a message and respond according to its Knowledge Base. This tool was deeply analyzed and reported in section *"Chatbot Developing Tools/AI Chatbot Frameworks"* of the previous chapter.

3.2 DEVELOPMENT

This section is supposed to introduce to the reader the strategy used in the development, the decisions made in order to implement the desired solution supplying the desired outcome.

These topics will be approached in the following sections. It is to be noticed that every decision regarding the strategy of the implementation was made taking in account some factors like time needed to implement a given solution by always seeing the big picture of the desired end product.

3.2.1 *Decisions*

In this journey we came across some possible solutions. The decisions had to be taken focusing on the overall final result, that was a functional Chatbot that could serve the initial purposes. It is to be noticed that the research provided different paths with different levels of difficulty. For instance, we could chose other machine learning models and train/tweak them from scratch. But we would have to consume much more resources. Those resources would include time and advanced computational hardware like a set of expensive high-end Graphics Processing Units (GPUs).

Instead, the aim was set on delivering a functional product in short time. To do that, we chose the proper tools to help the process of developing the Chatbot.

One of the requirements of the Chatbot was to work on different environments. To ensure that, some actions had to be taken in the *Dialogflow* console and on the *Backend* app.

Three different *Environments* were created on *Dialogflow*, called **dev**, **qa**, and **prod**. These environment names correspond to the following environments respectively:

- **Development** - local environment to run the program.
- **Quality Assurance** - remote environment for running the program and allow to test the changed code via either automated checks or non-automated techniques.
- **Production** - remote environment that users directly interact with.

Were created three different configurations in the *Backend* with the proper location of the resources and desired properties.

These environments would run on different machines and development environments. They would also contain different Knowledge Bases as well.

To conduct the user in the desired way and to prevent him or her from "wandering" the bot's Knowledge Base, when the user prompts some message that is off the bot's Knowledge Base, it was decided to use a decision tree structure to make the user choose the topic that he or she wants to be clarified. This works as a *fallback intent* and will be approached in the section *System Summary*.

Regarding the quality of the service, it was decided to provide a "switch" to enable and disable the Chatbot. To accomplish that, first it was decided to use the *Spring Boot* tools to read an environment variable. Depending on the state of that variable (*true* or *false*), the Chatbot would appear or not appear to the user. Given that the *Liferay* environment was making it difficult to accomplish the intended behaviour, it was decided to approach the problem in another way.

In order to accomplish the desired mechanism/switch, it was decided to make a simple *ajax* request from the *client-side* that basically queries the *server-side* app about the state of that variable. If the *server-side* responds with *"true"*, the Chatbot is going to appear to the user. That is accomplished by removing the *CSS* class *hidden*. If the *server-side* responds with "false", the *HTML* object that is rendering the Chatbot will be removed from *DOM* to ensure that the user wouldn't undesirably exploit the Chatbot.

To ensure that the dialogue is quite similar to a dialogue with a human being, it was decided to use an animation of a typing message with a time of response corresponding to the bot's time of thinking/processing the message, and answering accordingly.

It was decided to implement a *Feedback System* to improve the Knowledge Base of the Chatbot. To validate each answer given by the Chatbot, we decided to *"mark"* each answer as relevant or non-relevant to our *Feedback System*. So, if an answer is related with the platform it will have a "like" button attached that sends information to the database.

Since the user could have more specific doubts that falls off the Chatbot domain, we decided to implement a *"Book a Demo"* feature that builds the bridge between our Chatbot (Tier 1) and a human worker (Tier 2). This way our Chatbot could inform the platform's Support Team and confirm the user intent of scheduling a meeting with the support team by sending emails to both of them.

Both features will be later presented.

3.2.2 *Implementation*

Regarding the system's backbone implementation, it was decided to use an iterative methodology, that was based on CI/CD (*Continuous Integration/Continuous Deployment*). That methodology consisted in a cycle that increasingly develops features of the Chatbot (Continuous Integration), and deploys them to the chosen developing environment (Continuous Deployment). This methodology was chosen because it eases the software development and system's deployment. This mechanism is capable of building, testing and deploying the app for each *merge request* to the master branch on the repository, softening the implementation and deployment job.

Concerning the Chatbot agent, we have used two features: *Intent* and *Knowledge Base*. Intents are more interactive and are capable of defining complex dialogues with more than one context (enlightened later), also could provide different responses for the same *Intent* if we want to express a random response from a list. This feature can be observed in figures 24 and 25.

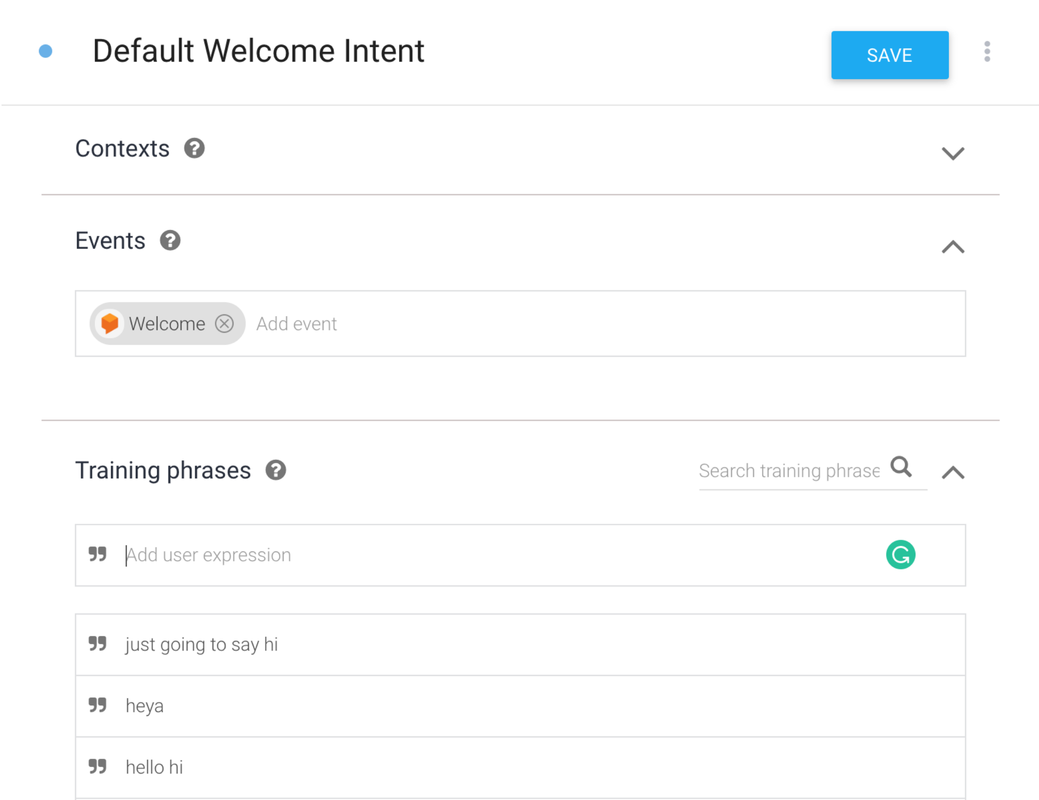

Figure 24.: Intent feature with context, training phrases [Dialogflow - visited at 26/07/2019]

We used the feature *Knowledge Base* to keep the questions and answers of a FAQ. This feature was able to create a model which could classify the user intent and respond with one straight answer. Both features (*Intents* and *Knowledge Base)* were used in the desired solution. It is possible to observe the Knowledge Base section of the console in figure 26.

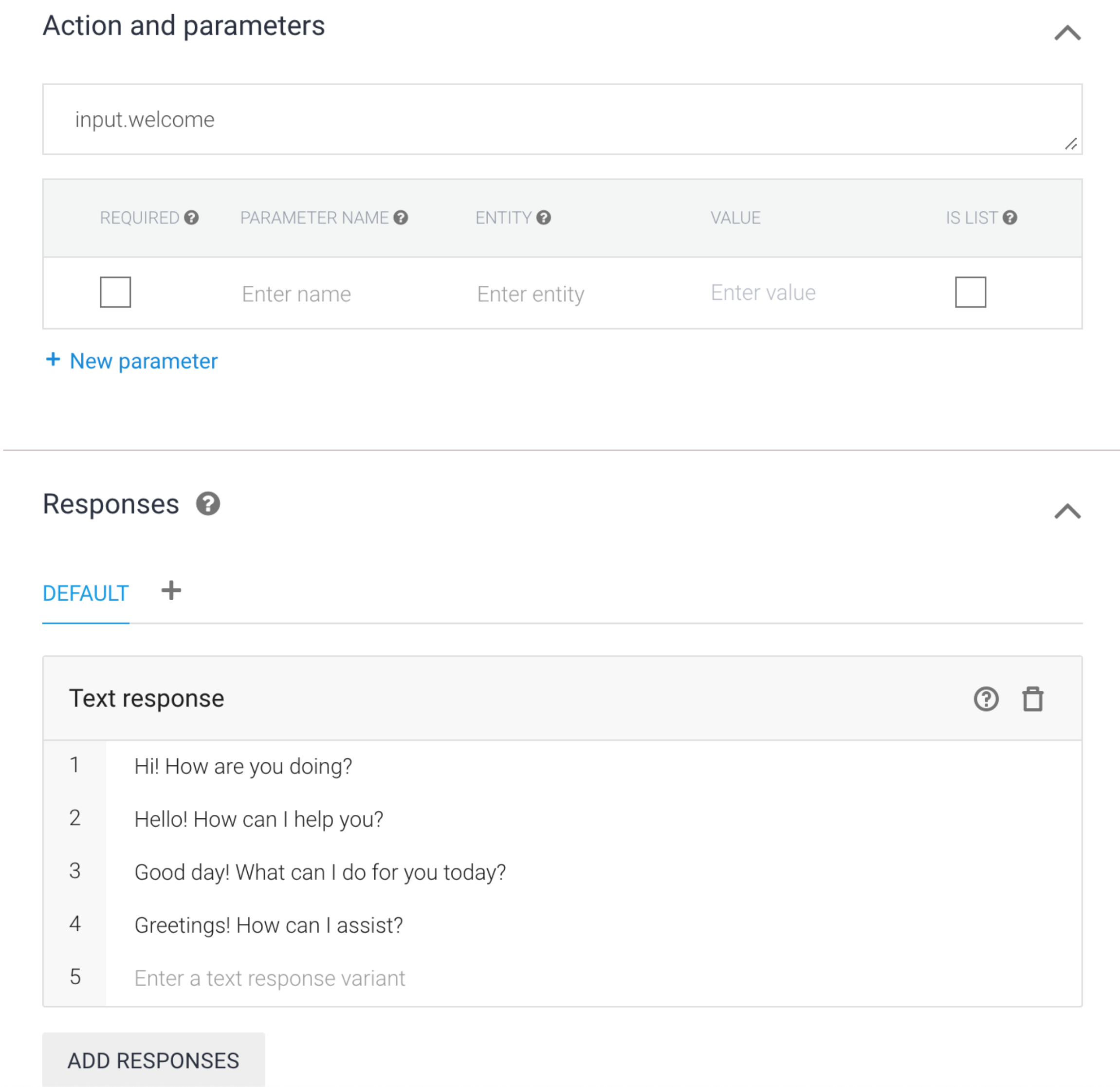

Figure 25.: Intent feature with Actions, Parameters and Responses [Dialogflow - visited at 26/07/2019]

Regarding the *Context* feature, it is responsible for providing the user with a concise dialogue with several interactions of questions and answers forcing the Chatbot agent to ”jump” between *intents* approaching a human conversation with several queries to get one response. This feature was explored although abandoned since introduced more problems than benefits concerning the determinism of the answers.

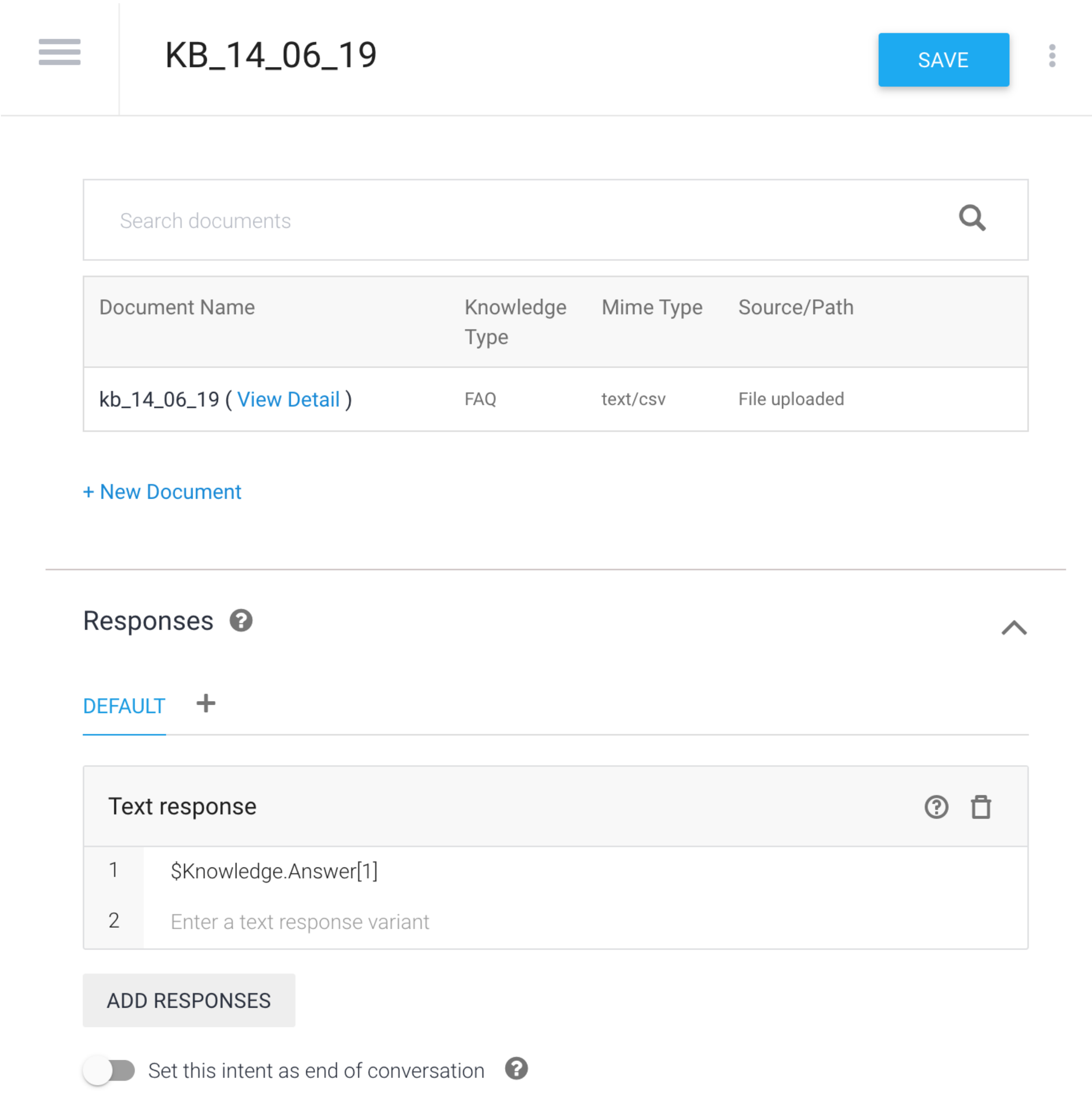

Figure 26.: Knowledge Base feature [Dialogflow - visited at 26/07/2019]

Since we could have *intents* overlapping *Knowledge Base* answers, this Chatbot tool provide us a mechanism to control the preference when the system has close matches between *intents* and the *Knowledge Bases*. Basically it displays an interval varying from **-1** to **1** in a decimal interval. If the user chooses a negative number it should make the agent prefer the result from the *Intent*, otherwise it makes the agent prefer the result from the *Knowledge Bases* as it can be seen in figure 27.

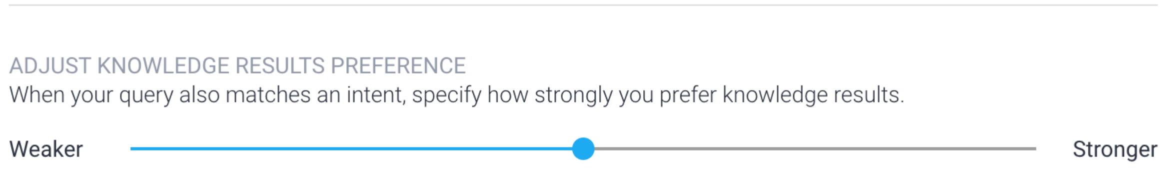

Figure 27.: Adjust Knowledge Base results [Dialogflow - visited at 26/07/2019]

3.2.3 *User experience*

The Chatbot interface will be presented below. It was accomplished thinking about the user experience. This Chatbot is present in every page to escort the user along the *website*. It should be like a tourist guide for the *Web* page in order to clarify the user regarding the questions they may have. Also it should be polite to the users when asking about unknown topics. When the user prompts greetings messages or make small talk and the Chatbot agent should respond properly. In figure 28, 29 and 30 we can see three different views of Mac, the Chatbot either closed, opened or having a conversation with a user respectively.

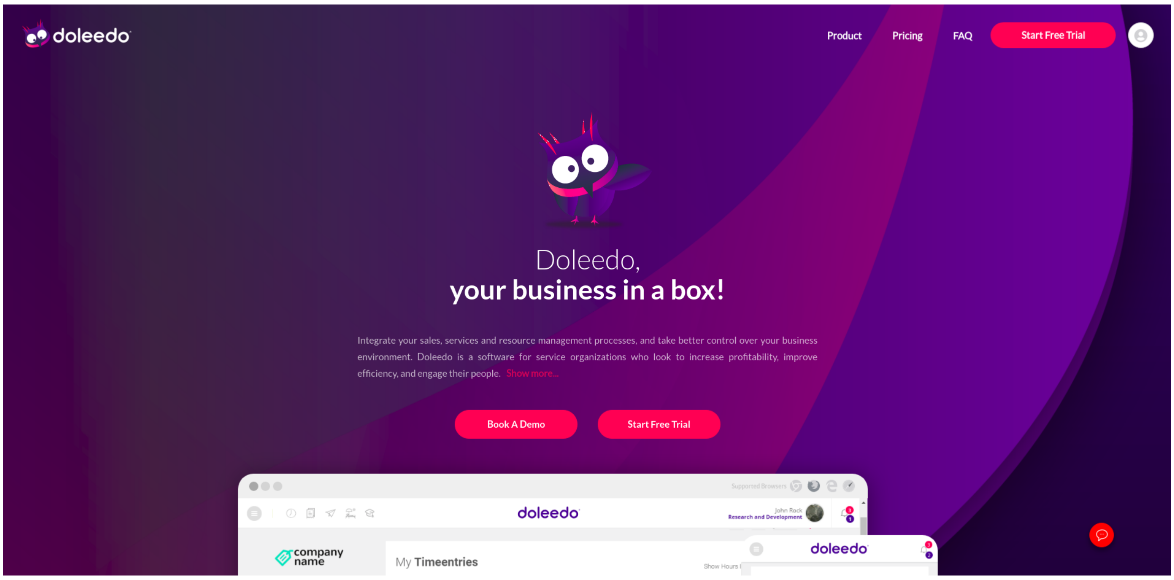

Figure 28.: The *website* with Chatbot closed at the bottom right corner [Dialogflow - visited at 26/07/2019]

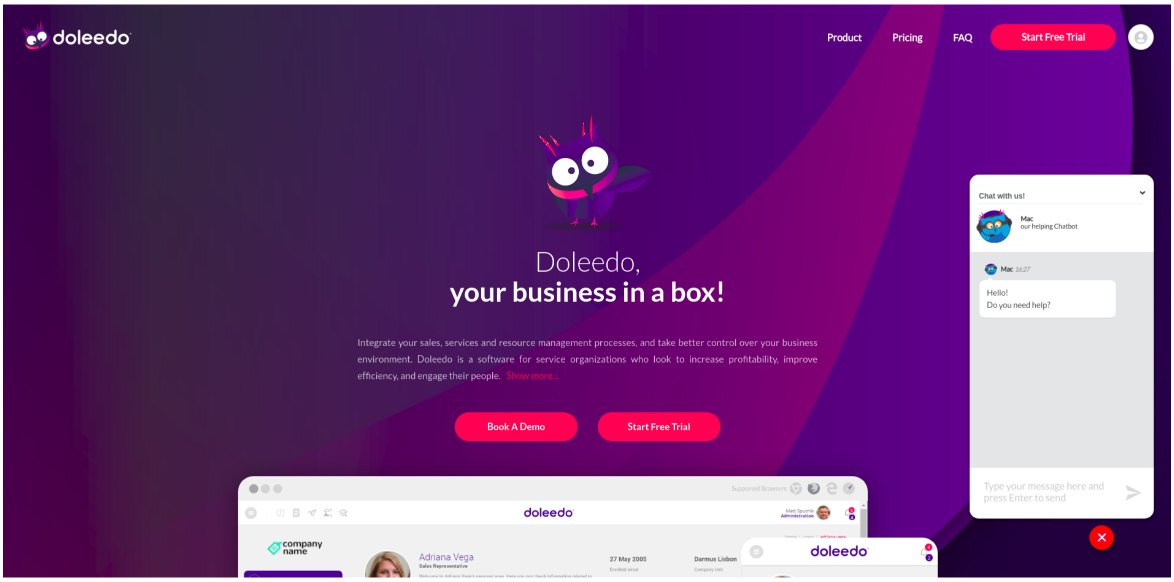

Figure 29.: The *website* with Chatbot opened at the bottom right corner

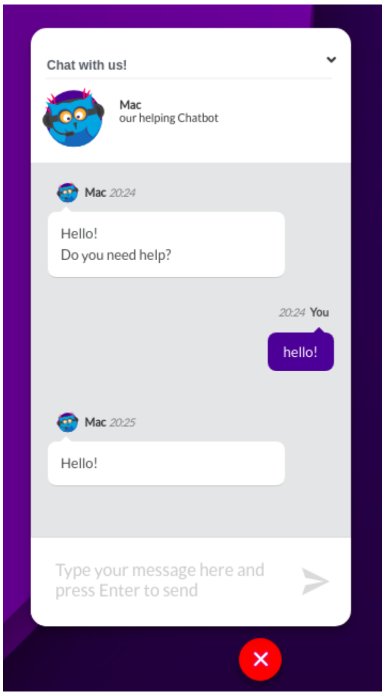

Figure 30.: Mac the helping Chatbot

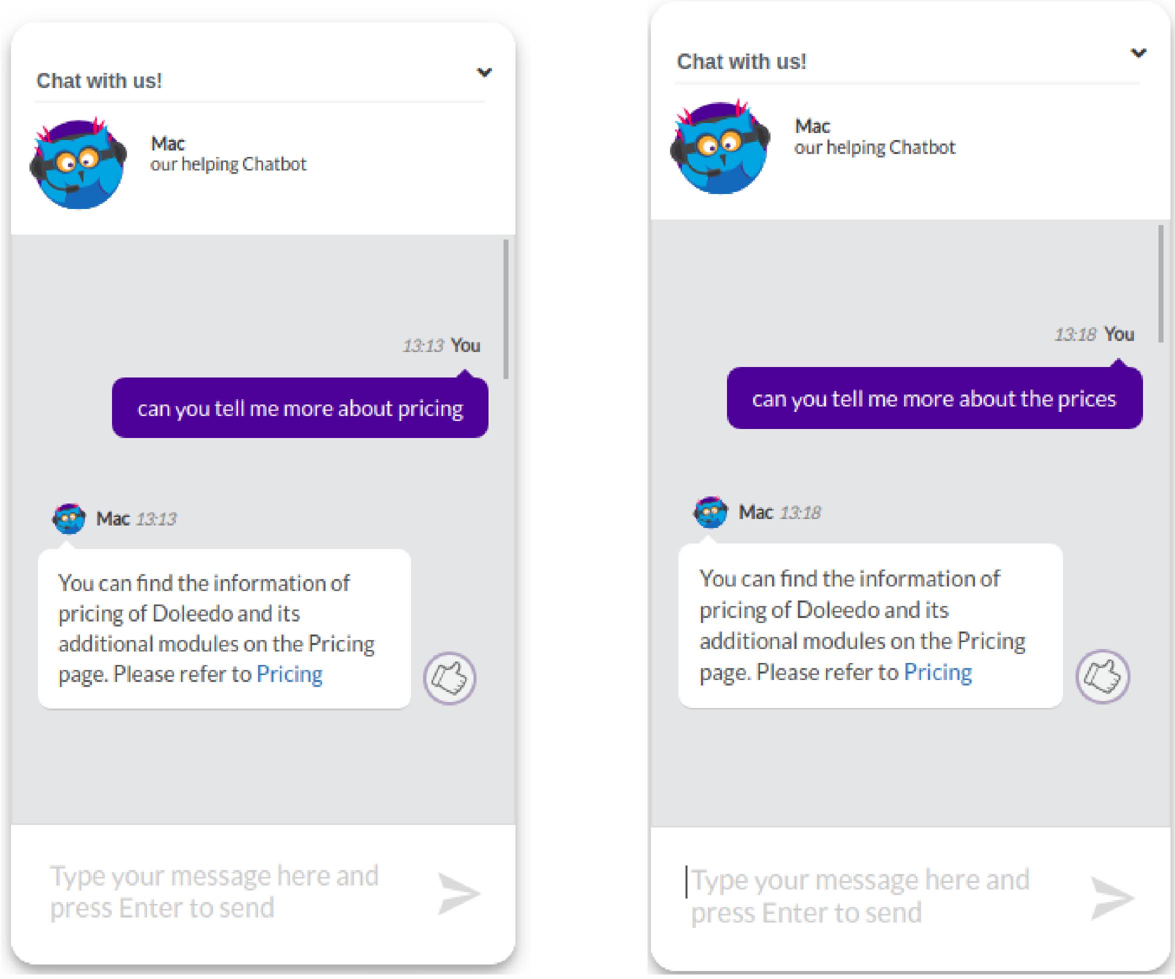

Figure 31.: Prompting the Chatbot with same intention but different phrases in Mobile environment

When asking about the product "Doleedo" or even about the terms and features that comes along with it, "Mac" (the Chatbot), presents an answer with a clarified message about the topic queried. Also, in order to ease the navigation on the *website*, "Mac" presents messages with "redirectable" *hyperlinks* for some of the *website* pages. Those messages with *hyperlinks* can be seen in figure 31.

It is to be noticed that the user could type the same question in other words or form, that would represent no issue regarding the classification of the message. That behaviour can be observed in figure 31.

3.2.4 *Feedback System*

In order to validate and provide a ”Data and metrics” analysis system we implemented a *Feedback System* to obtain the evaluation of the clients for each answer. Since this bot has a small talk module, we decided only to add a ”like” button on the answers retrieved by the Chatbot that would make sense to obtain a feedback. In our specific case the answers that make sense to get a rank from the users would be the answers whose theme was about the platform to be sold on the *website*.

With this system it would be possible to modify the Knowledge Base to make it more accurate. For instance if the clients do not ”rank” certain questions we could consider to reformulate its form or meaning, or even remove them from the Knowledge Base.

Another case could be if we spot a trend regarding a certain theme of conversation. In this particular case we should extend the Knowledge Base and provide more information or even more topics about that trend.

This system consists of a mechanism to validate each answer given by the Chatbot, that consists in a like button (that stands for approval) that accompanies each answer that has relevance for the user. In figure 32 we can see a dialogue where appears a message accompanied with a ”like” button.

As it can be seen in figure 33 the user can classify answers as useful by clicking the ”like” button. In order to distinguish if the user already sent its feedback regarding each answer, the button initially presents a faded gray color. When the user hovers over the button with its mouse, the button turns itself green and turns over itself in 35 degrees to the left.

Finally when the user clicks the like button, it fades itself to a dark blue to mark this answer as already classified as useful, sends an *Http* POST request to our *REST API* to store the data regarding the user feedback and disables the pointer to prevent spamming the system with repeated feedbacks. It is possible to observe a view of this event concerning the user interface in figure 34.

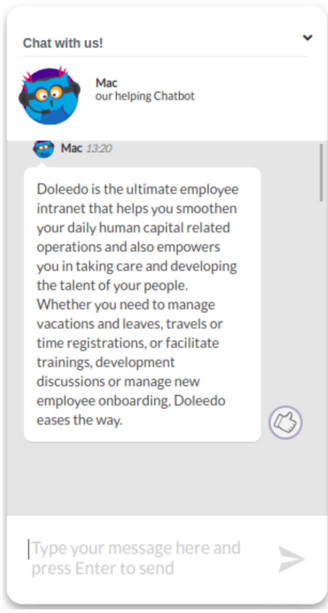

Figure 32.: Before a click on like button

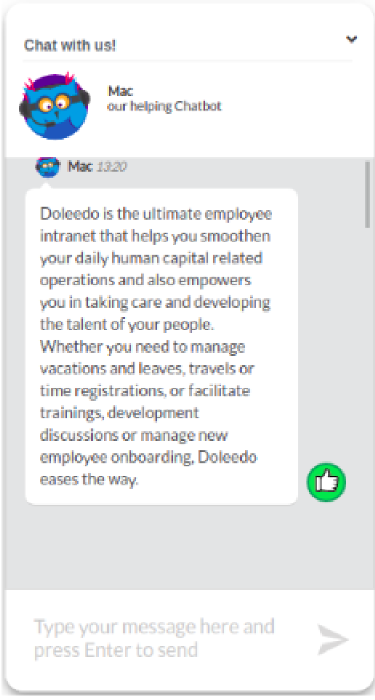

Figure 33.: Hover on like button

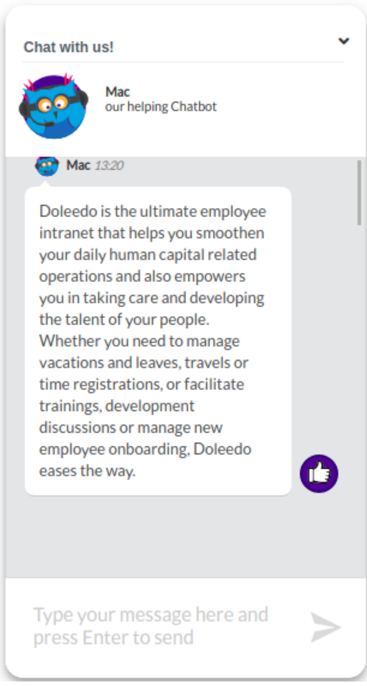

Figure 34.: After a click on like button

3.2.5 *Book a Demo*

With the idea of delivering a Chatbot that could satisfy all the questions that the user may have we have thought about a feature that could answer questions uncovered by the Chatbot's Knowledge Base.

This action is available for the user in two cases.

The first case is when the user expresses his/her desire of booking a demo in the Chatbot. The bot is going to trigger a Button confirming that the user wants to "Book a Demo".

The second case is when the user asks for content that is off the Chatbot's domain of conversation, we thought that could be a way of redirecting the user to "Book a Demo".

For both cases we would need a mechanism that could gather required data. Since this platform was designed for companies, the required fields are: **name** of the client that is responsible for the acquisition of this platform, **email** of the client, the **name of the company** that he is representing, and a **range of days** that the client is willing to appear in a chat conversation with a responsible for this platform. This feature can be observed in the figure 35. Later in the *System Summary* section we are going to present how this feature can be triggered.

When the user fills the form and presses the submit button, the system will store the appointment in the database using a POST request to the *REST API*, and send an email to the client (figure 36) confirming the appointment and an email to the platform responsible (figure 37) using a Simple Mail Transfer Protocol (SMTP) *Java* library.

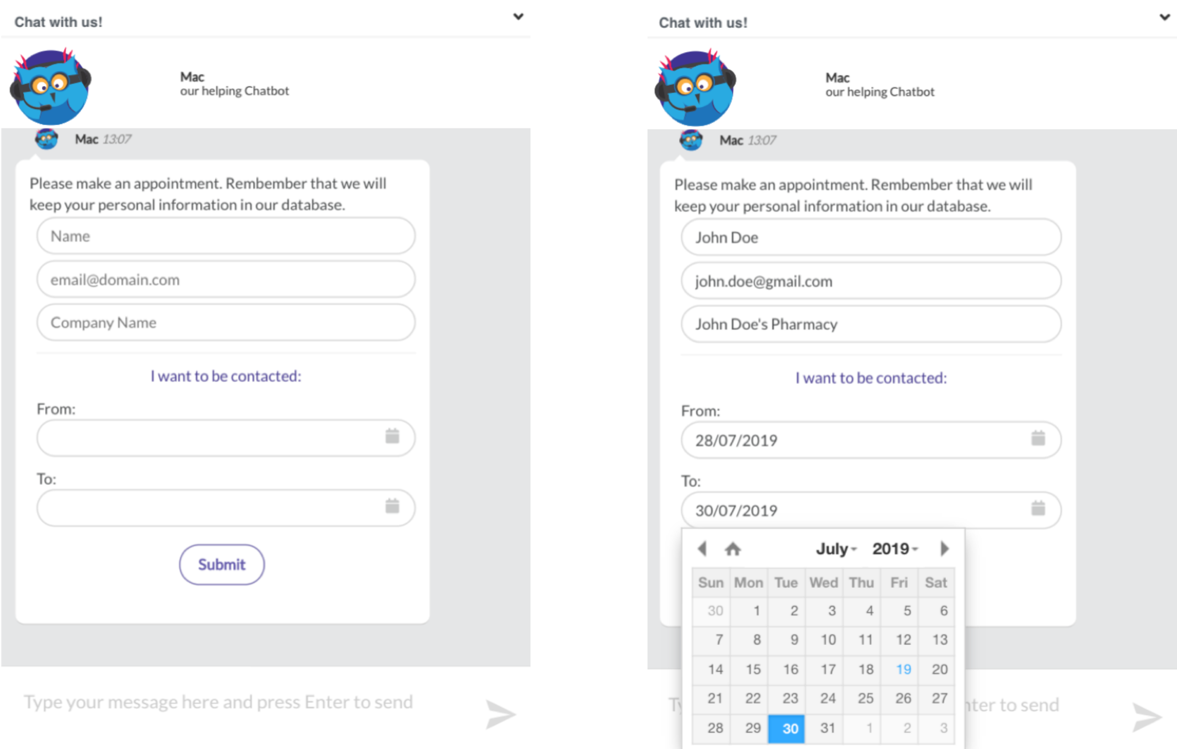

Figure 35.: Make an Appointment in Mobile environment

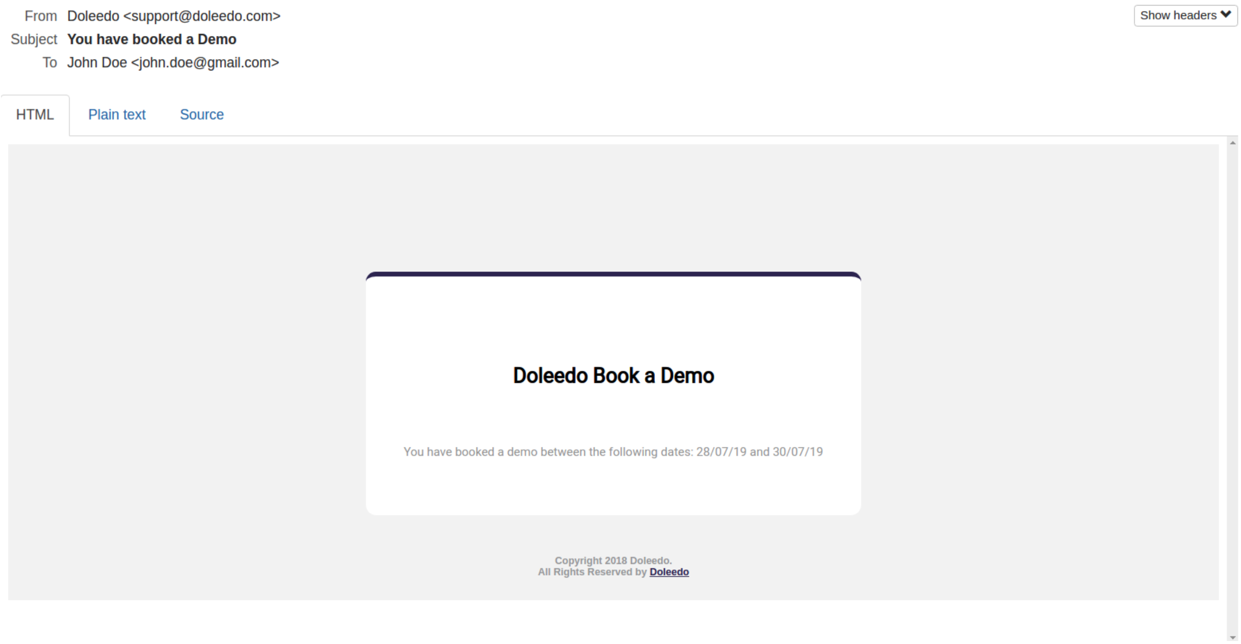

Figure 36.: Email to user that requested a demo

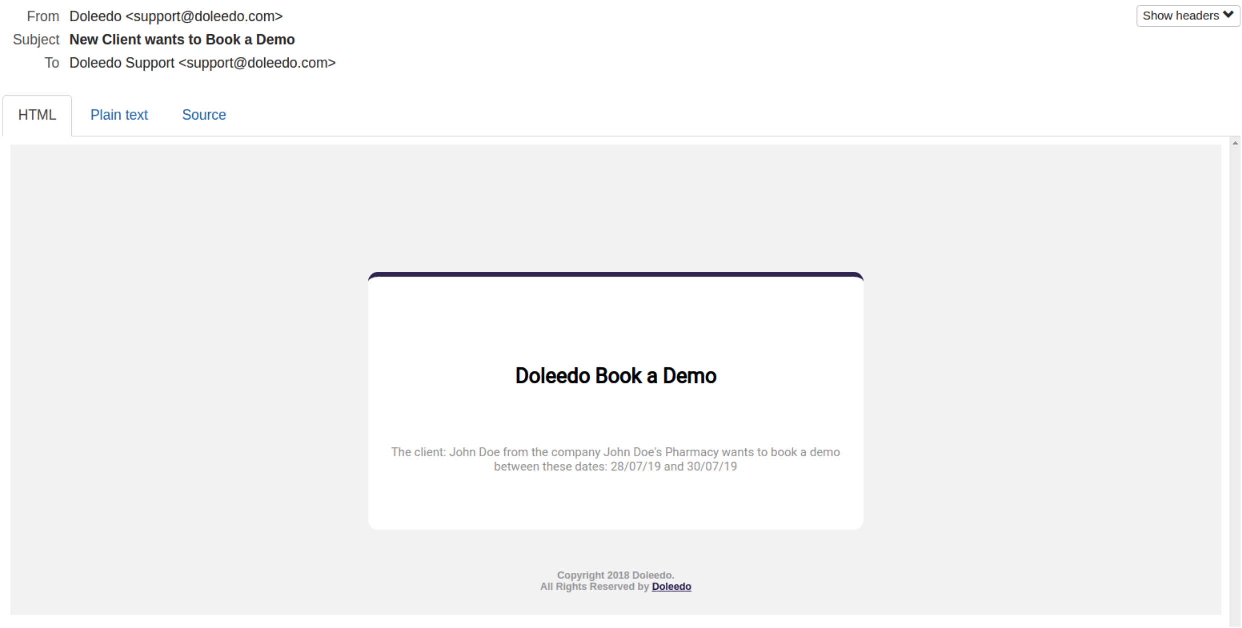

Figure 37.: Email to the support team

3.3 DATA PERSISTENCE

Concerning data persistence, we have managed to create three *Entities* that will correspond to three *Tables* on the Database named *AnswerRating*, *Answer*, and *Appointment*.

Regarding the Database, we used *Liquibase* to define the *Entities* conversion to **tables** on the database. Since we are using the *Hibernate* implementation of JPA (*Java Persistence API*) for data persistence, the *Entity* attributes should be written in camel case. Although, we should define those tables (former entities) in *Liquibase* with underscores instead of camel case. For example, when defining the entity *answerRating*, we should refer to it as *answer_rating* table in *Liquibase* in order to be able to use this data persistence system.

The first entity, that is *AnswerRating*, will have an *id*, the *answer_id*, a *session_id* that corresponds to a user ID, *sessionId*, retrieved by *Apache Tomcat*, and a *timestamp*.

Regarding the Database, the primary key on the table *answer_rating* will be a composite primary key of *answer_id* (int4) (corresponding to a foreign key on the *Answer* table) and *session_id* (varchar(255)). The idea is to prevent the same user from classifying the same answer more than once.

The second entity, *Answer*, has an *id* (Long), and a *sentence* (String) as properties annotated with *@Column JPA* annotation.

Concerning the database, this second entity will correspond to a table called *answer* and will have an *id* (int) row as primary key and a *sentence* (*varchar(2048)*). The purpose of this table (*answer*) is to map each *sentence* to an unique *id*.

The third and last entity is named *Appointment*. It is an entity with several properties: an *id* (Long), a *name* (String), an *email* (String), a *company* (String), a *dateFrom* (String), a *dateTo* (String). These last two properties are related to the appointment date range chosen by the user and are annotated with the *JPA* annotation *@Column*.

In terms of database, this last entity (*Appointment*) will be represented as the table *appointment* with the following columns: *id* (int4), *name* (varchar(255)), *email* (varchar(255)), *company* (varchar(255)), *date_from* (varchar(255)), *date_to* (varchar(255)).

The database scheme mentioned above is shown in figure 38.

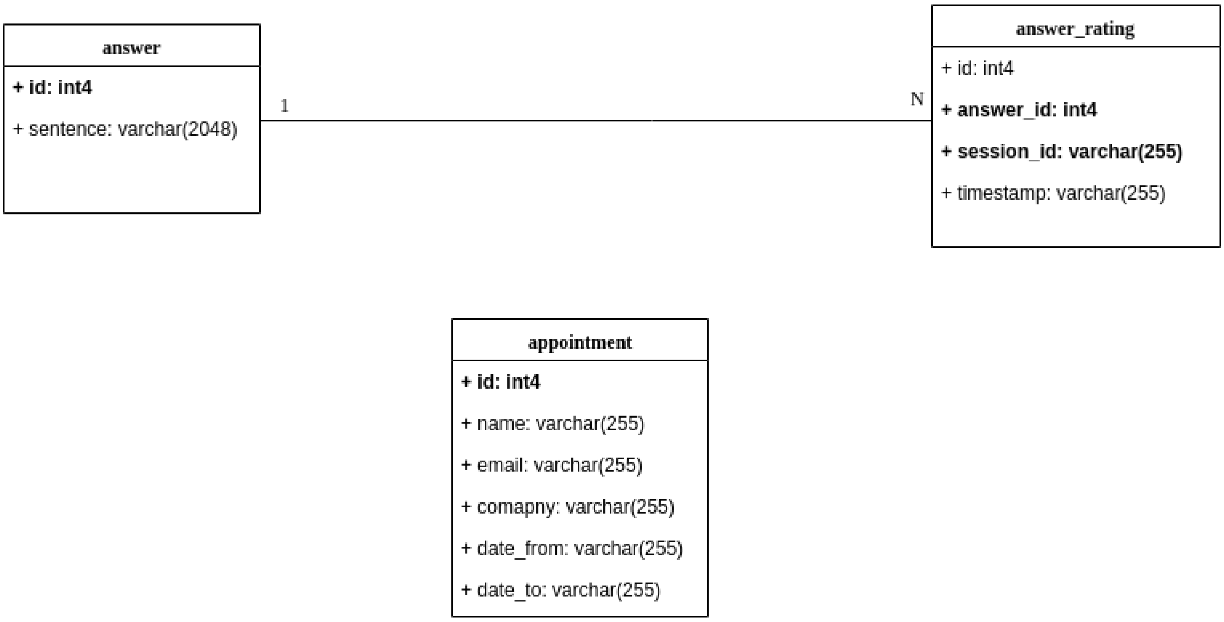

Figure 38.: Database Entity Relation (ER) Diagram

3.4 System Summary

One of the goals of this project is to impact the number of sales trying to convince the possible online clients by clarifying doubts about the product and even to book a meeting with a responsible. For that, we need to develop and deploy a Chatbot with some features. Those features need to be aligned with the product branding. When we talk about this subject we need to know the product in a holistic way. In terms of aspect and user interface we need to display a fun user interface with bright colours.

In order to transform this Chatbot into a tool that is capable of doing more than talking, we have decided to combine with a mechanism that gathers the usefulness of each answer *(Feedback System)* and added a system to schedule meetings (*Book a Demo*), by ”building the bridge” between the Chatbot, working as ”Tier 1”, and the support team, working as ”Tier 2”. It is possible to observe in picture 39 the two ways of triggering the ”*Book a Demo*” feature and also a button that redirects the user to the product demo environment.

Also we need to presume that the user is not willing to do much so the Chatbot needs to understand words with typos and unfinished sentences. And if the user ”falls” off the Chatbot's Knowledge Base, it needs to ”redirect” it to its domain. As so, we need to present a ”fallback intent” with known Chatbot topics. We can observe in figure 39 the ”fallback intent” when the user asks uncovered topics by the Chatbot. Figure 39 also shows the bright colours, the funny looking of the User Interface (UI) and also the welcoming looking of ”Mac's” avatar. All these factors should provide a desirable experience for the end users.

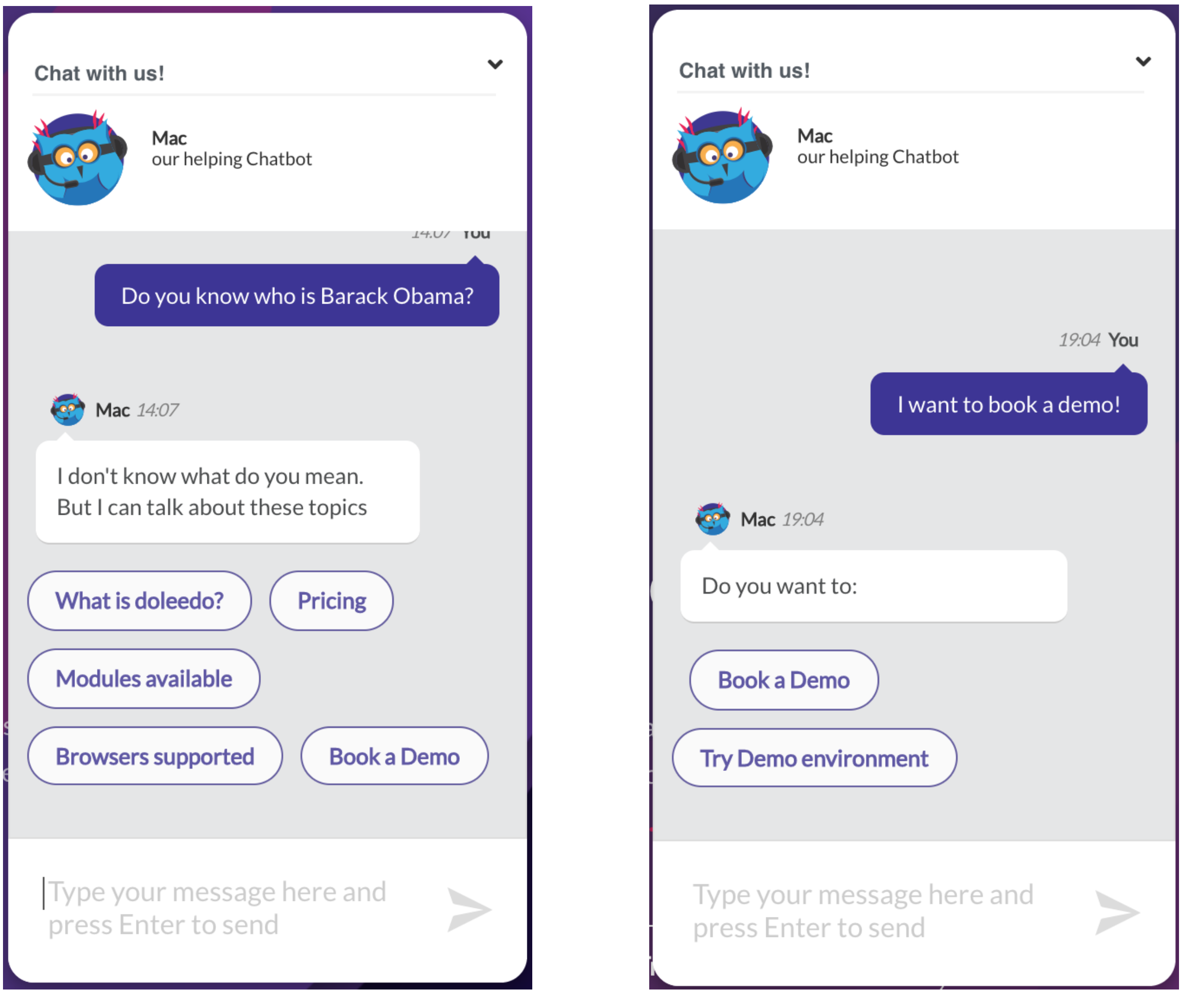

Figure 39.: Fallback intent and prompting the book a demo intent

3.5 EXPERIMENT SETUP

Given that the *production* Webpage where the Chatbot "lives" is not in the last version that is present on the *Quality Assurance* (QA) environment, we made a study with a group of ten people that agreed to impersonate a company's collaborator doing a study about buying the platform in cause in the QA environment.

The criteria used to evaluate were based on the following parameters: usefulness, ease of use, quality of the dialogues and most interesting feature.

For each participant, was given a questionnaire to fill in the end of their experiment on the *website* while they were "escorted" by "Mac", the Chatbot.

In the beginning we informed each participant that this platform was an intranet for small or medium companies and they are supposed to be in a investigation journey to acquire a new product for their company. Also, each participant was informed that was available a Chatbot to clarify all the doubts regarding the platform.

3.6 RESULTS

The following sections are going to approach the result analysis in order to understand the trend regarding the Chatbot evaluation by the group of people that agreed to test and evaluate the experience with "Mac" the Chatbot.

With the idea of drawing some conclusions it was decided to extract some metrics from the evaluations. To do so, we collected some evaluations about certain parameters of the Chatbot. Part of the parameters are supposed to be quantitative varying from **zero** to **twenty**: "Usefulness","Ease of Use", "Quality of the Dialogues", and one is qualitative/subjective: "Most Interesting feature".

The results of the experiment are available below in tables 6 and 7.

3.7 DISCUSSION

As it can be observed in tables 6 and 7, the parameter with the highest rating was the **Ease of Use** with a classification of **16.8** and the *"Most Interesting Feature"* pointed by the users were *"Recognizes user intent no matter the form of the question"* and *"Book a Demo"* both tied with **4** votes.

Despite the fact that the sample is small (*i.e.* 10 users), this short study demonstrates that "Mac" the Chatbot can replace a "Tier 1" human worker regarding main customer doubts and establishing an interface for a "Tier 2" worker with the *"Book a Demo"* feature.

3.8 STUDY SUMMARY

The Chatbot was tested by a group of ten persons willing to impersonate a company worker interested in buying a new product for their company. They executed this test knowing in advance that there was a chat and they were talking with a computer program, a Chatbot, that answers them with the help of a Knowledge Base. So the "Turing test" was not applied since the group already knew that they were interacting with a Chatbot and their main task consisted in evaluating it. Also every-

Participant	Usefulness (0-20)	Ease of Use (0-20)	Quality of the Dialogues (0-20)	Most Interesting feature
1	14	18	15	Recognizes user intent no matter the form of the question.
2	16	17	15	Recognizes user intent no matter the form of the question.
3	13	17	14	Book a Demo.
4	16	19	17	Messages with redirectable *hyperlinks*
5	18	17	15	Book a Demo.
6	12	16	14	Recognizes user intent no matter the form of the question.
7	19	15	16	Book a Demo.
8	17	18	16	Book a Demo.
9	18	17	16	Recognizes user intent no matter the form of the question.
10	14	14	13	Small Talk.

Table 6.: Study Results

Average Usefulness	Average Ease of Use	Average Quality of dialogue	Most Interesting feature
15.7	16.8	15.1	Recognizes user intent no matter the form of the question (4); Book a demo (4).

Table 7.: Extracted Metrics from the study

one in the tested group agreed on being honest regarding their opinion and that they were playing the role of an interested customer and defending legitimately their alleged company's interest.

With this short study we could better understand which Chatbot features were considered the most interesting. Those were the following: *"Recognizes user intent no matter the form of the question"* and *"Book a Demo"* both tied with four votes. These two trends reinforce the idea of the Chatbot being a contributing factor to leverage the platform sales: by connecting with and comprehending people expressing in different forms and when filling up a form that was triggered by a user intent to *"Book a Demo"*. Both facts could provide an interface that would reduce the resistance when talking to a computer program and should supply a better connection regarding the human-computer interaction.

4

CONCLUSION

The following sections will approach the conclusions in the end of this journey. There are three objectives concerning this chapter. The first objective is to state a summary of the project and dissertation that could allow a straightforward comprehension of the work done. The second objective would be an analysis on the completion of the objectives and goals. And the last objective would be regarding the field of research to develop the ideas explored here.

4.1 CONCLUSIONS

This project had several objectives regarding the implementation of an interface that enabled the contact between humans and a machine. The way people interact with computers and other information systems change with many factors, and this project does not intend to transmit the idea that people would prefer to talk to a Chatbot instead of a person. However, it could prove that an interaction with a Chatbot could improve the business in a digital marketing *website*.

This *website* needed a Chatbot agent capable of classifying human intentions when interacting with it. As such, an agent was created in the *Dialogflow* platform.

It also needed a *client-side* and *server-side* application to be integrated in the *website*. These were developed in separate containers since the architecture of the project was previously decided.

The project led to an approval study that consisted of users rating some aspects of the Chatbot as well as stating what was in their personal opinion the best feature present in the product and even though the sample was short we can conclude that generally, obtaining a Chatbbot should be a serious improvement to the business regarding online selling and customer relation.

4.2 Prospect for future work

A project with a scope turned into the digital marketing has a limited nature. Nevertheless, it is rather important to notice that it is always possible to extend this project in order to take advantage of the involved technology. Also, it is important to note that the validation done by the small sample was done by simple students that are not fully aware of company requirements and only evaluated the Chatbot regarding their user experience. As such it is possible that company workers could evaluate the Chatbot in other parameters and the global evaluation could be different. Given that this tool requires interaction with a public, this project would benefit with an increase of its functionality if the interaction with the audience were massive and also used the *Feedback System*. Otherwise, it could mislead the continuous improvement of the agent. Even so, this dissertation is concluded with the assurance that a relevant step was given in the direction of a better human computer communication approximating the developed tool to a Virtual Personal Agent.

BIBLIOGRAPHY

Amazon. Amazon.com Help: Set Up Your Echo, a. URL https://www.amazon.com/gp/help/customer/display.html?nodeId=201601770.

Amazon. Amazon Echo (2nd generation) — Alexa Speaker, b. URL https://www.amazon.com/all-new-amazon-echo-speaker-with-wifi-alexa-dark-charcoal/dp/B06XCM9LJ4.

Vyas Ajay Bhagwat. Deep Learning for Chatbots. *Master's Projects*, 2018. URL https://scholarworks.sjsu.edu/etd_projects/630/.

Bloomberg. Amazon Echo Is a Listening, Talking, Music-Playing Speaker for Your Home - Bloomberg. URL https://www.bloomberg.com/news/articles/2014-11-06/amazon-echo-is-a-listening-talking-music-playing-speaker-for-your-home.

Bianca Bosker. SIRI RISING: The Inside Story Of Siri's Origins – And Why She Could Overshadow The iPhone — HuffPost, 2013. URL https://www.huffingtonpost.com/2013/01/22/siri-do-engine-apple-iphone_n_2499165.html?ec_carp=3498241145807706155&guccounter=1.

Cleverbot. Cleverbot API – The official Cleverbot API. URL https://www.cleverbot.com/api/.

Cleverbot-website. Cleverbot.com - a clever bot - speak to an AI with some Actual Intelligence? URL https://www.cleverbot.com/.

Luísa Coheur. Chatbots: On Demand Creation of Conversational Agents Luís Filipe Amaral Pinhanç os dos Santos Engenharia Infor atica e de Computadores Examination Committee. (October), 2015. URL https://fenix.tecnico.ulisboa.pt/downloadFile/563345090414247/dissertacao.pdf.

Kenneth Mark Colby. *Machine Conversations*. 1999. ISBN 978-1-4419-5092-5. doi: 10.1007/978-1-4757-5687-6. URL http://link.springer.com/10.1007/978-1-4757-5687-6.

Giuseppe De Feo and Jean Hindriks. The Strathprints institutional repository (https://strathprints.strath.ac.uk) is a digital archive of University of Strathclyde research outputs. It has been developed to disseminate open access research outputs, expose data about those outputs, and enabl. *Journal of Economic Behaviour and Organization*, 106:213–226, 2016. ISSN 1996-0786. doi: 10.5897/AJEST2016.2108.

Dialogflow. Dialogflow. URL https://console.dialogflow.com/api-client/#/agent/a817fe8f-44c5-4cf1-84eb-54f40adacd1b/editIntent/9019754f-fef7-400f-a7a6-b9cd05d19892/.

Docker. What is a Container? — Docker. URL https://www.docker.com/resources/what-container.

Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78, 2012. ISSN 00010782. doi: 10.1145/2347736.2347755. URL http://dl.acm.org/citation.cfm?doid=2347736.2347755.

Drift. Introducing LeadBot 2.0 – Drift. URL https://www.drift.com/leadbot/.

Ecommerce-Nation. Chatbots: Advantages and Disadvantages of these Tools. URL https://www.ecommerce-nation.com/chatbots-advantages-and-disadvantages-of-these-tools/.

Endurance. ChatBots for Senior People and Patients with Alzheimer's Disease - EnduranceRobots. URL http://endurancerobots.com/azbnmaterial/chatbots-for-senior-people-and-patients-with-alzheimer-s-disease/.

Robert Epstein. The quest for the thinking computer. *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, 13(2):3–12, 2009. ISSN 0738-4602. doi: 10.1007/978-1-4020-6710-5{_}1.

Robert Epstein and Gary Roberts. *Parsing the Turing Test*. 2009. ISBN 9781402096242. doi: 10.1007/978-1-4020-6710-5. URL http://www.springerlink.com/index/10.1007/978-1-4020-6710-5.

David Ferrucci, Anthony Levas, Sugato Bagchi, David Gondek, and Erik T. Mueller. Watson: Beyond jeopardy! *Artificial Intelligence*, 199-200:93–105, 2013. ISSN 00043702. doi: 10.1016/j.artint.2012.06.009. URL http://dx.doi.org/10.1016/j.artint.2012.06.009.

Mary Jo Foley. Microsoft's 'Cortana' alternative to Siri makes a video debut — ZDNet, 2014. URL https://www.zdnet.com/article/microsofts-cortana-alternative-to-siri-makes-a-video-debut/.

Bruce W Frost. Book Reviews. *British journal of hospital medicine (London, England : 2005)*, 71(11): 656–Unknown, 2010. ISSN 1750-8460. doi: 10.3115/1118693.1118704. URL http://www.ncbi.nlm.nih.gov/pubmed/21063265.

Zoubin Ghahramani and Michael I Jordan. Factorial Hidden Markov Models. Technical report, 1997. URL http://papers.nips.cc/paper/1144-factorial-hidden-markov-models.pdf.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep Sparse Rectifier Neural Networks. Technical report, 2011. URL http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf.

Google. Google Store oficial para dispositivos e acessórios produzidos pela Google. URL https://store.google.com/?srp=/product/google_home.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS Alex Graves, Abdel-rahman Mohamed and Geoffrey Hinton Department of Computer Science, University of Toronto. *IEEE International Conference*, (3): 6645–6649, 2013. ISSN 09310509. doi: 10.1093/ndt/gfr624.

Gusto. Gusto. URL https://gusto.com/.

Francis Ho. TA-bot: An AI agent as a Teaching Assistant. (May):5, 2018. doi: 10.13140/RG.2.2.34344.06408.

IBM. Watson for President 2016. URL http://watson2016.com/.

IBM Cloud. IBM Cloud. URL https://cloud.ibm.com/login.

Jason Cross. Hey, Siri: 142 useful voice commands for Siri — Computerworld. URL https://www.computerworld.com/article/3261408/hey-siri-142-useful-voice-commands-for-siri.html.

Dan Jurafsky. Conversational Agents AKA Dialog Agents. 2018.

Eleonora Kurilchik. Chatbots as a Digital Marketing Communication Tool. 2017. URL https://www.theseus.fi/bitstream/handle/10024/131171/Thesis_Kurilchik.pdf?sequence=1&isAllowed=y%0Ahttps://www.theseus.fi/bitstream/handle/10024/131171/Thesis_Kurilchik.pdf?sequence=1.

J. Lafferty, A. Mccallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data Part of the Numerical Analysis and Scientific Computing Commons Recommended Citation "Conditional Random Fields: Probabilistic Models for Segmenting and Labelin. *Proc. of ICML*, 2001(June):282–289, 2001. ISSN 1750-2799. doi: 10.1038/nprot.2006.61. URL http://repository.upenn.edu/cis_papersPublisherURL:http://portal.acm.org/citation.cfm?id=655813PublisherURL:http://portal.acm.org/citation.cfm?id=655813ThisconferencepaperisavailableatScholarlyCommons:http://repository.upenn.edu/cis_papers/159.

Chris Lau. Why Cortana Assistant Can Help Microsoft in the Smartphone Market - TheStreet, 2014. URL https://www.thestreet.com/story/12534433/1/why-cortana-assistant-can-help-microsoft-in-the-smartphone-market.html.

Brígida Solange Macaza. *M © 2017*. PhD thesis, 2017.

Matt Mcgee. Microsoft: No Plans For Ads In Cortana Yet; Debating Bringing To Androids & iOS - Search Engine Land. URL https://searchengineland.com/msft-plans-monetize-cortana-yet-want-pervasive-193988.

Michael Mctear. Conversational Modelling for Chatbots : Current Approaches and Future Directions. *Conference on Electronic Speech Signal Processing (ESSV 2018)*, 2018. URL http://essv2018.de/wp-content/uploads/2018/03/2_Keynote_MichaelMcTear_ESSV2018.pdf.

Microsoft. Cortana's regions and languages. URL https://support.microsoft.com/en-us/help/4026948/cortanas-regions-and-languages.

Tiago Miguel and Moreira Ferreira. Adaptive Automotive Chatbot. 2017.

Tom M Mitchell. *(Mcgraw-Hill International Edit) Thomas Mitchell-Machine learning-McGraw Hill Higher Education (1997)*. 1997. ISBN 0070428077. doi: 10.1016/j.cub.2007.11.035.

Martijn Van Otterlo. *The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for The Markov Decision Process Framework in First-order Domains*. 2010. ISBN 978-90-365-2677-7.

Jordan Palmer. Google Home will now let you schedule calendar appointments, reminders coming soon, 2017. URL https://www.androidpolice.com/2017/05/17/google-home-will-now-let-schedule-calendar-appointments-reminders-coming-soon/.

Ron Parr. CPS260/BGT204.1 Algorithms in Computational Biology. Technical report, 2003. URL https://www2.cs.duke.edu/courses/fall03/cps260/notes/lecture14.pdf.

Sarah Purewal and Jason Cipriani. The complete list of Siri commands - CNET, 2017. URL https://www.cnet.com/how-to/the-complete-list-of-siri-commands/.

Adwait Ratnaparkhi. A Maximum Entropy Model for Part-Of-Speech Tagging. Technical report. URL https://www.aclweb.org/anthology/W96-0213.

Trips Reddy. How chatbots can help reduce customer service costs by 30% - Watson. URL https://www.ibm.com/blogs/watson/2017/10/how-chatbots-reduce-customer-service-costs-by-30-percent/.

Roof.ai Roof.ai. Roof - AI Assistant for Real Estate. URL https://roof.ai/.

Bryan Sams. Windows 10: Cortana now syncs reminders - Neowin, 2015. URL https://www.neowin.net/news/windows-10-cortana-now-syncs-reminders.

Roger C. Schank. Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology*, 3(4):552–631, 1972. ISSN 00100285. doi: 10.1016/0010-0285(72)90022-9.

Fabrizio Sebastiani. P1-Sebastiani. 34(1):1–47, 2002. ISSN 03600300. doi: 10.1145/505282.505283.

Dan Shewan. 10 of the Most Innovative Chatbots on the Web — WordStream, 2017. URL https://www.wordstream.com/blog/ws/2017/10/04/chatbots.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. 2015. ISSN 1550235X. doi: 10.1103/PhysRevB.92.155314. URL http://arxiv.org/abs/1506.06714.

Frans Arne Stylegar and Oliver Grimm. Ein sp??kaiser- und v??kerwanderungszeitlicher kanal in spangereid, s??norwegen. *Archaologisches Korrespondenzblatt*, 33(3):445–455, 2003. ISSN 0342734X. doi: 10.3115/1075812.1075844.

Techradar. Google Home review — TechRadar, 2019. URL https://www.techradar.com/reviews/google-home.

A. M. Turing. Computing Machinery Inteligence. *Mind*, XX(77):150–153, 1950. ISSN 0026-4423. doi: 10.1093/mind/XX.77.150. URL https://academic.oup.com/mind/article-lookup/doi/10.1093/mind/XX.77.150.

Oriol Vinyals and Quoc Le. A Neural Conversational Model. 37, 2015. ISSN 0099-2399. doi: 10.1007/978-3-319-19291-8{_}22. URL http://arxiv.org/abs/1506.05869.

Richard S. Wallace. The anatomy of A.L.I.C.E. *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, pages 181–210, 2009. ISSN 00035769. doi: 10.1007/978-1-4020-6710-5{_}13.

Matt Weinberger. Amazon Echo vs. Google Home vs. Microsoft Cortana vs. Apple Siri - Business Insider, 2017. URL https://www.businessinsider.com/amazon-echo-google-home-microsoft-cortana-apple-siri-2017-1.

Joseph Weizenbaum. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966. ISSN 00010782. doi: 10.1145/365153.365168. URL http://portal.acm.org/citation.cfm?doid=365153.365168.

Wit. Wit — Wit guide (legacy). URL https://wit.ai/docs/complete-guide.

Wit.ai. Wit.ai - agent. URL https://wit.ai/humbertovaz/agent/entities.

Wit.ai Team. Introducing Wit Speech API – Wit.ai – Medium. URL https://medium.com/wit-ai/introducing-wit-speech-api-6c3a514cf245.

Xiaohao Yang and Jia Liu. Using Word Confusion Networks for Slot Filling in Spoken Language Understanding. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 23(3):1353–1357, 2015. ISSN 2307-387X. doi: 10.1002/hrm.

Zenefits. #1 HR Software — Human Capital Management (HCM) — Zenefits. URL https://www.zenefits.com/.

Du Zhang. Inconsistency-Induced Heuristics for Problem Solving. In *SEKE 2011 - Proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering*, pages 137–142, 2011.

zoho. Zoho - Cloud Software Suite and SaaS Applications for Businesses. URL https://www.zoho.com/.

.1 APPENDINX

.1.1 *Experiment with Recursive Neural Networks*

To achieve the desired result was decided to do an experiment that consisted of building a Chatbot from scratch. In order to accomplish a Chatbot that could understand users using regular or misspelled words, after a deep research was decided to use a specific RNN algorithm commonly used in NLP named NMT (Neural Machine Translation) that tries to predict the likelihood of a sequence of words.

This model had to be trained to recognize natural language, so one way to achieve that was to capture english dialogues from movies subtitles. It was downloaded a 20GB set of subtitles from OpenSubtitles *website* with the *XML* format and continued to its processing.

That text processing consisted in text parsing that intended to capture questions and answers that composed a dialogue between two different persons. This parsing could be done in different ways, but it was chosen the programming language *AWK* due to personal familiarity and its speed processing files.

In order to parse questions, it was used this script:

```
    #!/usr/bin/nawk -f
BEGIN {
    FS="[><]"
    f = 0;
}
/[a-zA-Z ,;0-9]+[\.!]+/ {
    if(f==1){
        answer[question]=$1;
        f=0;
    }
}
/[a-zA-Z ,;0-9]+\?/    {question = $1; f=1;}
END{
    for(q in answer){
        print q;
    }
}
```

To parse answers was used this script:

```
#!/usr/bin/nawk -f
BEGIN {
    FS="[><]"
    f = 0;
}
```

```
/[a-zA-Z ,;0-9]+[\.!]+/ {
    if(f==1){
        answer[question]=$1;
        f=0;
    }
}
/[a-zA-Z ,;0-9]+\?/{
    question = $1;
    f=1;
}
END{
    for(q in answer){
        print answer[q];
    }
}
```

This methodology used two "stages", the training and the testing stage. Due to that, the model must be "fed" with four files that corresponded to the questions and answers files for the two stages.

The division in train and test data was 80% for train directory and 20% for the test directory.

Given that every answer corresponds to only one question, the software used had to have two files for each "stage". And it was required that these two files matched. That is, the question at line number one in questions file should correspond to the question for the answer at line number one present in the answers file and vice-versa. This pattern should be preserved until end of files.

This text processing took place in a directory full of XML files. In order to achieve the desired text processment in four different files, these were the runned commands:

```
awk -f parse_xml_subtitles_answers.awk ~/chatbot_dataset/testData/*xml  >> test.to
awk -f parse_xml_subtitles_questions.awk ~/chatbot_dataset/testData/*xml  >> test.from
awk -f parse_xml_subtitles_answers.awk ~/chatbot_dataset/trainData/*xml  >> train.to
awk -f parse_xml_subtitles_questions.awk ~/chatbot_dataset/trainData/*xml  >> train.from
```

Neural Machine Translation or NMT is a technique that uses a large Artificial Neural Network to predict a sequence of words. It is used as a technique in the modern natural language translators.

This setup consisted on a Python script found on Github that is the implementation of Chatbot using Google's NMT system. This project was abandoned due to the fact that it would need a handful of resources and time consuming either training and developing the system. It would be rather difficult or unenforceable in the time that this project was meant to be.

A

SUPPORT MATERIAL

Auxiliary results which are not main-stream; or

Details of results whose length would compromise readability of main text; or

Specifications and Code Listings: should this be the case; or

Tooling: Should this be the case.

NB: place here information about funding, FCT project, etc in which the work is framed. Leave empty otherwise.