MyCampus Chatbot - A Conversational Agent that offers information to IST Students

Francisco Evangelista da Cruz Maria

Thesis to obtain the Master of Science Degree in

Telecommunications and Informatics Engineering

Supervisor: Prof. José Alberto Rodrigues Pereira Sardinha

Examination Committee

Chairperson: Prof. Ricardo Jorge Fernandes Chaves
Supervisor: Prof. José Alberto Rodrigues Pereira Sardinha
Member of the Committee: Prof. Alberto Abad Gareta

October 2019

Acknowledgments

First, I would like to thank my family for supporting me and helping me all of these years.

I would also like to thank my dissertation supervisor Prof. José Alberto Sardinha, who helped me and guided me through all of the project stages, without him this project would not be possible.

Lastly, I would like to thank my friends that helped through all of university and the project. They gave me every help that they could give and helped during the testing phase of the chatbot.

Thank you to all.

Abstract

In this thesis, we are going to discuss the creation of a personal assistant for the students of Instituto Superior Técnico, in the form of a chatbot. The objective of the bot is to facilitate the lifes of students and make existing information easily accessible. The bot will get all of its information from the FenixEdu Application Program Interface (API), this information includes degree information, student's information, courses information, payments, dates to evaluations and even information about rooms. The bot is also able to give the user notifications about course announcements. We also used and tested Google's Dialogflow as a tool to create chatbots.

Keywords

ChatBot; University personal assistant; DialogFlow; Firebase; FenixEdu API; IST;

Resumo

Nesta tese vamos falar sobre a criação de um assistente pessoal para os alunos do Instituto Superior Técnico, que vem na forma de um chatbot. O objetivo do bot é facilitar as vidas dos alunos fazendo informação existente mais fácil de aceder. O bot irá receber toda a sua informação do FenixEdu API, esta informação inclui é sobre cursos, o estudante, disciplinas, pagamentos, datas para avaliações e até informação sobre salas. O bot também é capaz de enviar notificações sobre anúncios de disciplinas para o utilizador. Também usámos e testámos o Dialogflow da Google como uma ferramenta para criar chatbots e usamos o Google's Firebase e o Cloud Scheduler.

Palavras Chave

ChatBot; Assistente Pessoal da Universidade; DialogFlow; Firebase; FenixEdu API; IST.

Contents

List of Figures

List of Tables

Acronyms

IST	Instituto Superior Técnico
A.L.I.C.E	Artificial Linguistic Internet Computer Entity
AIML	Artificial Intelligence Markup Language
XML	Extensible Markup Language
CMC	Computer Mediated Communication
LiSA	Link Student Assistance
SDK	Software Development Kit
HTML	Hypertext Markup Language
API	Application Program Interface
UI	User Interface
JSON	JavaScript Object Notation
URL	Uniform Resource Locator
AMP	Accelerated Mobile Pages
SUS	System Usability Scale
FAQ	Frequently Asked Questions
HTTP	Hypertext Transfer Protocol

1

Introduction

Contents

This thesis is going to discuss the creation and development of a voice and text based chatbot to help Instituto Superior Técnico (IST) students to get specific IST information. The application is an attempt of making the information already available in Fenix easily accessible.

In the next chapters we are going to explain the motivation behind the system, how we made it and tested it.

1.1 Motivation

To go about their normal day, students need access to some information given by the school. From the shuttle schedule, to when and where their next class is, the school cafeteria menu and even the news about Técnico and all that happens in it. To facilitate their lives this kind of information should be given in the easiest and most comfortable possible way.

At this moment this information about Técnico and students activities in Técnico is all available in the university website, the Fenix website and, some part of it, in the IST mobile application.

The Fenix website is where personal information about the user is and its one of the most used platforms by IST students. However, opening the browser, inserting the Fenix credentials and then search for the information is neither the quickest nor the most comfortable way of getting information, especially if it is simple information. An application that can be easily accessible and that has all of the user personal information is needed.

The IST website is where information about the school is located. This includes contacts to various school infrastructures, information about courses and more basic information.

In the IST mobile application, you can find information like the cafeteria menu, shuttle schedule and evaluation dates. The app can also sync your school calendar with your phone calendar. It is a quicker and easier way to get information, but it does not have as much data as the Fenix website, it lacks some information about the students.

So, as seen, there is a lot of information that the students need and it is scattered throughout multiple places. There is a need to link that information to make it easier to access. There is a need for something like a personal assistant, or a chatbot, that knows everything about students as well as IST. Something that would respond to user queries and know the user context to better help him. An app where all the information can be found, from the mail of the Academic Office to the link to the curriculum of a specific course.

A chatbot is a technology more and more often used to communicate with the user, because they guarantee an easy and natural way of giving valuable information. A chatbot is a computer program that through a written or talked conversation can comprehend and respond to the user depending on what he says and asks. This type of technology is being more frequently used to help customers, it is replacing the humans for being an easy way to answer costumers' questions and for that reason we have chosen this technology to deliver the information that the students need.

1.2 Contributions

The objective of this thesis was to use the chatbot technology to help students access their information easily and faster.

We created a chatbot application that can give the students access to different IST information, all specified in chapter 3. The applications can be accessed via Google Assistant and was created using Dialogflow and Firebase, both Google technologies. The user can talk to the chatbot, in English, by using text or speech. We tested the system and learned that the students like a chatbot to get there information and that the application is faster than the Fenix website.

In the end we made a system that is easy and intuitive to use. It provides information in the easiest and simplest way.

1.3 Thesis Structure

Our thesis is divided in 5 chapters, in chapter 2 we are going to present scientific papers that introduce chatbots, talk about similar projects and projects with similar architectures. Then, in chapter 3 we are going to describe how we built the system. In chapter 4, we are going to describe the tests we did and analyse the data. In the last chapter we conclude the thesis.

2

Related Work

Contents

In this chapter we are going to talk about research that is of value to our project. We will first talk about research that introduces chatbots and their interaction with humans. Then we are going to show some university chatbots. And we will end by showing projects that use Dialogflow as we do.

2.1 Introduction to chatbots and chatbot-human interaction

In this section we are going to present three papers that introduce us to chatbots, gave us a better understanding where chatbots can be used and help us to understand how humans communicate with a chatbot.

2.1.1 Chatbots: Are they Really Useful?

Bayan Atwell et al. [1] investigate applications to see where chatbots can be useful. Applications like information retrieval, business, education and e-commerce applications. They start the paper by talking a little bit about the history of Human Computer Interaction and in the other chapters they present practical chatbot applications that are found in daily life.

They start by introducing Artificial Linguistic Internet Computer Entity (A.L.I.C.E) [2]. Alice's knowledge comes from Artificial Intelligence Markup Language (AIML) files that store English conversation patterns. AIML, or Artificial Intelligence Mark-up Language, is a derivate of Extensible Markup Language (XML) and consists of data objects called AIML objects. They explain AIML in more detail and say that a disadvantage of AIML is that the categories have to be ”hand-coded” and so they made a program that makes it automatically.

They then talk about chatbots as an entertainment, the first chatbot of this kind was ELIZA [3] a chatbot that emulated a psychotherapist. It was really rudimental and only had specific answers to certain keywords. This system and the way it makes people talk about their problems inspired a lot of modern chatbots that want to fool users to think they are talking to a human.

Chatbots can also be information retrieval tools, in this paper they have some examples of chatbots that are used to retrieve information. One is called FAQchat [4] and in the paper was compared to Google. Another system that they talk about is the YPA [5], a system that allows users to retrieve information from British Telecom's Yellow Pages.

The final example of chatbot uses is in assisting E-Commerce, business and other domains. They start by talking about a system that helps clients find information in a shopping site. The results were very good and the users found it easy and that reduced the time they would take to find their desired item.

In the end they concluded that a chatbot must be a tool built to help people, facilitate their work and their interaction with computers.

This paper was helpful because it shows that a chatbot can be relevant and well received by humans in various areas and important areas to our project like information retrieval and helping students.

The paper was also helpful to ours as an introduction to chatbots and all that chatbots can do, it gave us a big insight on some of the earlier chatbot projects and chatbot history.

2.1.2 Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations

J. Hill et al. [6] gave us an insight on how communication changes when people communicate with a chatbot and when they do it with a human.

The paper starts by giving information on previous researches about human communication with some forms of Computer Mediated Communication (CMC) (computer mediated communication), but none of them done it comparing human communication with the new and popular way of CMC, chatbots.

To analyse this new form of communication, human-chatbot communication, the authors collected unaltered instant messages, conversations from volunteers of a university, to be the human-human conversational component of the test. For the human-chatbot conversations they used real conversations of Clevearbot, an internet chatbot, given by the developer. In total they investigated seven variables in the messages, words per conversation, messages per conversation, average number of words per message, the percentage of unique words per conversation, abbreviations, emoticons and profanity.

From previous works and researches, the authors were expecting that the average human would send fewer messages and write fewer words per message to bots than to humans, but this was not the case. The results were that indeed people sent fewer words per messages when talking to bots but sent more than twice as many messages that when talking to humans.

The reason the authors gave for this result is that people were modelling their communication to match that of the chatbot, to test this hypotheses they analysed the data and discovered that humans chatting with Cleverabor average a similar (small) number of words per message as Cleverbot and people chatting with humans average a similar (higher) number of words per message as their messaging partner. This and other data seem to confirm the previous hypotheses, people tend to model their communication to match that of their conversation partner.

Another expectation of the authors was that conversations with the bot would have more profanity. That was confirmed with almost a 30-fold average increase in profanity and sexually explicit conversations, when talking to the bot.

With this paper we have learned that when communicating people try to recreate the style of communication of their partner, so in our project we gave a good example of the type of communication that we want. Our chatbot talks in a simple and correct way. The paper also points that profanity and sexually explicit conversations is something that happens a lot in this type of communication, but we were not able to create specific responses to deal with this.

2.1.3 The relationship between motives for using a Chatbot and satisfaction with Chatbot characteristics in the Portuguese Millennial population: an exploratory study

T. D. Rieke et al. [7] tried to explore a part of chatbot research that is lacking, the analysis of chatbot characteristics and motives for using this technology. To analyse that they made two questionnaires, one about satisfaction with chatbot characteristics and another about motives for using chatbots. The survey was done to Portuguese millennials of the University of Porto, this population and their interests about chatbots are very interesting to this paper because the students of IST are in the same age and conditions.

First, in the paper, they start by giving an introduction to chatbots in general. They talk about different topics like chatbot characteristics, satisfaction about that characteristics, motives for using a chatbot and then about how they are going to do the survey and what methodology it is going to be used. But the important part for this project is the final part, the one that we are going to describe in more detail, where they analyse the results and reach some conclusions about what millennials in Portugal think and like about chatbots.

They start the results chapter by describing the participants of the questionnaire. The questionnaire was answered by 317 people but of them only 258 responses were found adequate. In these responses the number of female participants and male participants were practically the same and their ages were between 18 and 38, but the majority of the participants were under 26 years old. Most of them had a bachelor's degree or equivalent and a considering number or participants had completed their master's degree and doctorate.

The first question was about the familiarity with chatbots, as seen in other papers almost all of the students knew chatbots, but when asked if they used chatbots more than 30% said that they had rarely used chatbots or nothing at all. So chatbots are interesting for most of the participants but most of them do not use chatbots in a daily basis. When asked about a chatbot characteristic, that they prefer, the participants seem to be interested in a chatbot that is fast and accessible and not a chatbot with emotions and feelings.

In the end, with these results, they concluded that the Portuguese millennials have some experience with a chatbot, but rarely use one. Contrary to other research that classifies emotion in a chatbot as an important feature, Portuguese millennials prefer a bot with fewer emotions and more speed and accessibility, features like sadness or empathy may cause a decrease in satisfaction with the bot. Efficiency is the most important characteristic for this group. The last finding was that convenience, exploring and social are three big motives, for this group, to use a chatbot.

Even though the population of this questionnaire was of Porto University students we think that IST students may have similar beliefs and that we can learn a lot with this paper. We have learned that Portuguese millennials care about efficiency and one of the big reasons for them to use a chatbot is the chatbot being convenient, which is a goal of our project. Our bot should then be as efficient as possible. It should be a bot that gives an answer in a quick and correct way.

2.2 Similar projects

In this section we take a look at a Técnico application that already tries to help students and at three chatbot projects made to assist students in a University.

2.2.1 Técnico Application

One of the main objectives of our chatbot is to make Técnico students and faculty lifes better and easier, there is an application that already tries to help them. The Técnico application [1] is a mobile application that enables users to access some Técnico information easily.

The application shows news from IST and announcements from classes. It also shows the cafeteria menu for the day, the shuttle schedule, parking information, it enables students to take a ticket for the academic office, see courses information, evaluations, the student curriculum and the payments that the student needs to make. The application also allows students to sync their phones calendar with their class schedule.

The application does a really good job in showing some information, but it lacks in information about rooms and interacting with the students, by for example sending them notifications about important information.

With this application we have learned a little bit of what information is available right now to Técnico students and what our application has to do better to help the students and faculty.

2.2.2 Ambient Intelligence: The MyCampus Experience

N. Sadeh et al. [8] report about a five years project of the MyCampus group at Carnegie Mellon University. This group has been experimenting with Ambient Intelligence technologies aimed at enhancing everyday life. The project concentrates on developing an open Semantic Web infrastructure for context-aware services with an emphasis on privacy and usability. The project itself was implemented as an application that is aimed at enhancing everyday life of students and faculty, in the university campus, and it uses context-awareness to do that.

[1] https://tecnico.ulisboa.pt/pt/viver/ servicos/aplicacoes-moveis/

They start the paper by talking about the architecture of the project. The system is composed of a number of agents or applications that the user can download. These applications are context-aware and each of them has to access some information about the user and is context to function correctly. This information is stored in a e-Wallet.

The e-Wallet is a repository of static knowledge about the user. The e-Wallet also contains knowledge about how to access more information about the user by invoking other resources, each represented as a Web Service. This is stored in the form of rules that the e-Wallet can, depending on some query, automatically identify and activate. The e-Wallet also stores users specific privacy rules, this ensures that information about the user is only disclose to those who have permission. They also inform that agents are not limited to only access e-Wallet information they use other Web Services to get information. The knowledge in the e-Wallet is divided in several categories:

Static knowledge. This knowledge is context-independent, it includes the users name, email and some preferences.

Dynamic knowledge. Context-sensitive knowledge, often involving some preferences like ”When in class I do not want to receive messages.”.

Service invocation rules. These rules help on how to get external information to the e-Wallet, both personal and public. An example of this mapping is a rule indicating that a query about the users' location can be answered by looking at the users GPS location in his phone.

Privacy preferences. In the privacy preferences we have the information about what the user is willing to disclose to others under different conditions. For example, the user can say that he lets his colleagues see his location only during the week between 8am and 5pm.

After that, the authors give more detail and examples on the implementation of the e-Wallet and what technologies they use. Then they talk about instantiating the MyCampus infrastructure and all of the context-aware services that they have done to help the students' lives.

The MyCampus infrastructure was conceived to enhance campus life at Carnegie Mellon University and over the years the team has worked to create and refine applications that can help students and faculty in their daily activities. Some of those applications are listed below:

Context-aware recommender services. Include recommender services for places where to eat, nearby movies and public transportation. The services can make recommendations based on the users' preferences, like his favourite type of food, his current location and also information like how much time is until the users next meeting, the weather, etc.

Context-aware message filtering services. Allow to specify when the user wants to see different types of messages based on the nature of the message. For example, they may only want to see urgent messages when on a meeting.

Context-aware reminder applications. Applications that remind users about tasks they have to perform in relation to their location and other contextual attributes.

Context-sensitive ”crime” alert applications. Warn users when they are entering an area more prone to incidents.

Collaboration applications. Enables people to selectively share PowerPoint presentations.

Community applications. These applications are extremely popular and can go from mundane calendar scheduling and people locators to more sophisticated virtual, context-aware poster applications.

In the end the authors concluded that in this type of context-aware applications the user privacy issues are very important and that capturing the preferences of the user is a major impediment to the broad acceptance of ambient intelligence technologies.

This paper is relevant for our project because it shows how context-aware apps work inside a University Campus and gives ideas of useful apps and features using context-awareness for our project, like a service that helps people navigate the campus or a service that helps people find an empty room depending on their location. It also tries to make these context-aware applications to help and improve the lifes of students and faculty in campus, a key part of our project.

2.2.3 Chatbot in a Campus Environment: Design of Link Student Assistance (LiSA), a Virtual Assistant to Help Students in Their University Life

M. Dibitonto et al. [9] did some research to implement LiSA a chatbot that helps students in their campus life. With this research they want to understand which information and services are better accessed by students and how the chatbot personality influences the user experience and interaction. To do that they do a survey to identify the users' needs and their inclination to use a chatbot.

A chatbot can do a number of services but in this paper they want to know what kind of services are most desired by students. If it should provide services like booking an exam or paying annual fees, if the messages should be delivered in pull and/or push mode. In pull mode the chatbot responds to questions by the user but in push mode it gives information to the user without any input. Another big question is what kind of personality should the chatbot have.

They then do a background introduction where they talk about what is a chatbot and why is it used, the conversational interface of a chatbot and the user experience. They have an example on this topic where they say that the use of the users first name during a conversation with the chatbot is perceived as positive. In the end of this section they conclude that a well-designed chatbot has to have a personality, has to establish a relationship with the user, understand the context and hold a conversation about general arguments. It should not try too much to imitate a human and should be able to overcome users' misspellings, other mistakes or rude comments.

In the next section they presented how to design a chatbot directed to university communities, based on analysis of other existing university chatbots. In an academic environment there are two chatbot categories: the first one is a chatbot that helps students to find the best university to their interests; the second one is a chatbot used in a specific university, it can do a specific task or be a virtual assistant to students. The chatbot they are going to develop is going to be a virtual assistance, this type of chatbot usually provides generic information like university facilities, admissions and courses. The type of information needed from the bot may vary during the academic year for example during the application process the chatbot can help with enrolment information. The information as mentioned before can be pulled our pushed. Pushed information can be relevant updates of deadlines, a change in the class schedule or important events coming soon. Pull information can be upcoming events, academic information or university facilities.

To know users' needs and preferences they design a questionnaire. The questionnaire was divided in three sections. The first section is about the participants personal information, the second investigate the information that students want to receive, how to receive it and the satisfaction about university services. The third and final section was to know participants familiarity with chatbots. To give this questionnaire they used LiSA a low-level dialog system chatbot built on the Facebook Messenger platform.

In this survey participated 100 students, most of them were Link Campus University students, some were students from different Italian Universities and others were high school students.

Table 2.1: Data about the tools students used to communicate with the Link Campus University

	Usage	Partially satisfied	Totally satisfied
Website	93%	28,6%	64,3%
Email	50&	15,4%	77%
Phone	53,3%	7,7%	92,3%
In person	75,9%	26,3%	68,4%

Table 2.1 shows results of the questionnaire, we can see that students use different tools to communicate with the University. The highest satisfaction rate comes from phone calls followed by in person requests which means that personal interaction is preferred to ask information. Another highlight from the questionnaire is that students do not use Facebook to get information, the main reason for that is that Facebook is considered an informal communication tool, not suitable for communication with the University.

In table 2.2 we have the results when students were asked how they usually look for several kinds of information. We see that enrolment is the first reason for students to ask for information and that

Table 2.2: Data about the tools students used to communicate with the Link Campus University

Types of Information					
	Enrollment	Scholarship	Opportunities	Events	Schedule
Website	21%	40%	36,7%	28,3%	21,7%
Email	18,5%	27,5%	22%	31,6%	27,8%
Phone	33%	12,5%	4,9%	26,6%	29,9%
In person	27,5%	20%	36,4%	13,5%	20,6%

they prefer to get it by phone or in person. Class schedule is another primary issue due to the possible changes, the preferred tools are email and phone, this is because it allows students to get this information without going to the university.

The other questions focus on the information Link Campus University offers to students. Some students said that they receive personalised information and a majority of them said that they find that kind of information useful, email seems to be the most used tool to provide this information.

All the students listed information that they would like to receive by the University via chatbot. That information is in Figure 2.1.

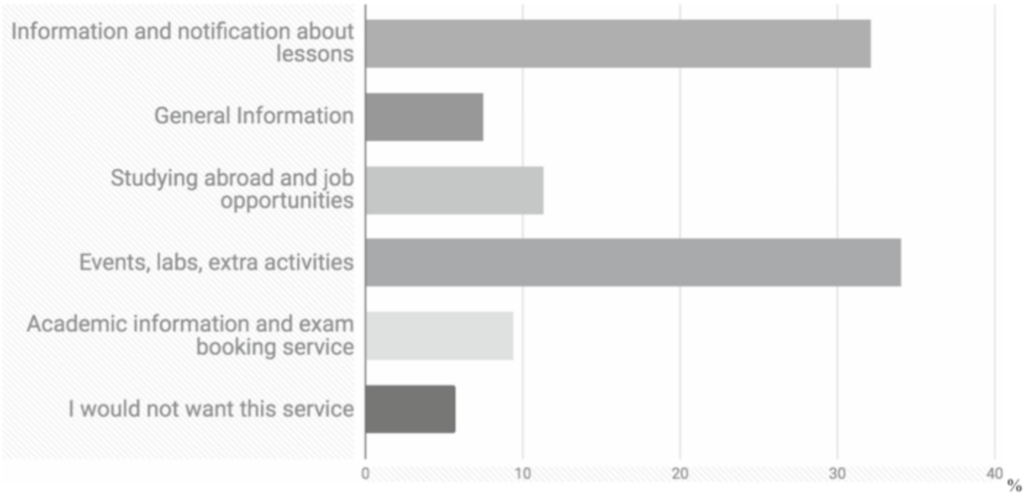


Figure 2.1: Data about what type of information students would receive by a university chatbot.

In the last section of the questionnaire the authors give information about the user's behaviour when talking to the questionnaire chatbot. It has been shown, in other papers, that people have a different behaviour during a conversation with a chatbot. In this study they found the same thing. They discovered that when added a sensible personality, hurt by aggressive behaviours, some students apologised to the bot and others tried to excuse their behaviour with the curiosity of testing the chatbot limits.

In the end they say that table 1 is going to be very important to define the information database of the chatbot, table 2 is interesting to design further services for LiSA. They also say that users would find useful to have the chatbot push information to them, especially class schedule changes. They would also like information about University's events and activities that could be personalised to student's interests. Exam booking and enrolment information it is also going to be taken into account.

This paper is relevant for our project because it gives a really good perspective of what students want of a University personal assistant. The questionnaire was done in a different University with a different campus and different information, but we can still learn with all the answers and conclusions that they take. It helps us to know what to use and not to use in our chatbot.

2.2.4 Evaluation, development & testing of an educational support chatbot

B. I. Systems et al. [10] determined if there is a use for chatbots in a university, specifically Cardiff Metropolitan University, were all of the questionnaires and data came from. In this paper the author shows all of the data collected in is questionnaire to students and staff and then shows how he implements the chatbot and all its functionalities.

The first survey done in this paper was to the students, in total 100 students took the survey in a week coming from a total of 27 different courses.

This survey had two main goals: know if students are satisfied with university communications and if they would adopt chatbot technology as a solution.

From the survey they found that: to communicate with the university students prefer email and then face to face communication. This shows that these students only communicate when they are in the university, most of them are only in there 3 or 2 times a week. This is a relative limited communication time for students who communicate face to face; the most frequent question that students need to ask is assignment queries and a lot of them are simple ones, like word count, due date and marking criteria; students say that they can find information that they need without difficulty in the university web site; in terms of communication platforms, the most used is WhatsApp but this kind of platform does not have a chatbot technology. But 94% of students have Facebook Messenger installed which has an open chatbot platform; 60% of students said that they would feel comfortable communicating with a chatbot to receive the same information they would get from a human; the final question was about chatbot personality and if a chatbot should have one. Students were split between a yes and a no/don't care, so the author concluded that students do not want to be fooled into thinking that they are talking to a human when they are not.

With this data the author concluded that the survey was favourable to the creation of a chatbot for the university. Students were comfortable with communicating with a chatbot, had a good platform to create a chatbot already installed on their phones and seemed responsive to the idea. They concluded that Facebook messenger is the best platform to have a chatbot. That the chatbot could help staff receive fewer emails a day and that the chatbot must have information from the university website to make it easier for staff that have difficulty using the site.

The system itself is all built using Oracles IBCS and Facebook Messenger to deploy the bot. The login is made in Moodle, the tool used by the university. After the initial greetings the boot asks the user login info and with that he establishes a session with Moodle through Moodle's Application Program Interface (API). After that the user can make the questions he wants, the bot responses are in two sections, a Q&A (questions and Answers) section made up by all FAQs (Frequently Asked Questions) found on the university site and the data that can be taken from Moodle via the API. When the user inputs a question that the boot cannot be sure in which intent it sits it gives intent options for the user to choose, guaranteeing that the user gets is correct answer. When the user asks broad questions like ”can I have assignment feedback” the bot also gives him a list of mores specific topics, in this case it gives a list of the user assignments. One interesting feature that the authors added is the possibility to talk to a human if the bot does not understand the question. In this case the bot asks the user if he wants to talk to a human and if he wants then the question will be placed to a human. Lastly a user can only be logged in 5 minutes before it as to re-authenticate. This feature was implemented because the access to Moodle gives access to a lot of sensitive information.

In the end they talk about the Facebook platform, that gives a lot of tools to get detailed analytics and add-ons to, for example, access web payments.

In this paper we see another example of a chatbot for a University, but unlike other solutions this chatbot also helps staff and takes them in to account. This solution gave us different ideas about student/staff preferences and features to have in a university chatbot, like different options if a question is vague. It also gave us an insight into the development of the bot, even though it was on a different platform that the one we are going to use, it was very helpful.

2.3 Architectural similar projects

In this section we show three projects that use Dialogflow to create their chatbots, the same tool that we used to create ours.

2.3.1 Conversational modelling for chatbots: current approaches and future directions

M. Mctear [11] examined the conversation models of some chatbot frameworks. We are going to pay more attention to DialogFlow, the framework we are going to use. The author also assesses if it models important conversational phenomena like follow-up questions, changes of topic and out-of-scope utterances.

First, they define two types of chatbots, a transactional chatbot that is used to accomplish a simple task and a conversational chatbot that engages in a chat for primarily social purposes. Our chatbot is of the first category.

The current frameworks extract intents and entities from the user's queries. An intent is something that the user wants to achieve and an entity is a parameter that is required to fulfil an intent.

There are two main types of conversational interaction, one-shot queries and slot-filling dialogues. One-shot queries are user-initiative queries. They are initiated by the user and are simple questions/-commands with no context and a simple answer/action. Normally there are no follow up questions and no subsequent questions by the system; slot-filling dialogues are system-initiative, are initiated by the system and can have some context. For example, the system can ask if a person wants to travel and then if they say yes, asks were they want to travel. In this case the system maintains a context of the conversation. A more natural way of conducting a conversation is in a mixed-initiative way. After the system asks a question the users asks another question in the same context. But most of the systems do not allow this type of conversation. Follow up questions can be asked if the system does not understand the query or if does not have the intent needed.

After this explanation of conversation models the author compares some frameworks. We are only going to talk about DialogFlow. DialogFlow allows some of this behaviour, it allows follow-up intents to be added to any intent and provides some built-in follow-up intents. It sets up an output context for a parent intent so that the context can be maintained between follow-up intent and parent intent. Slot-filling is also guaranteed, DialogFlow uses a parameter table were the required parameters and their associated slots are specified.

This paper is a good introduction to conversational models and to chatbots in general, it gives us a basic understanding on conversational models and in the end it also has a topic on difficulties that developers have in chatbot creation. It also talks a bit about features in DialogFlow, the technology that we will use.

2.3.2 Developing an Intelligent Chat-bot Tool to assist high school students for learning general knowledge subjects

D. Dutta [12] describes a chatbot tool for high school students and how it was built. This tool will answer questions about general knowledge. At this moment to get an answer to their question's students have to google the information, read links and papers until they find the answer they want. With the chatbot the process will be simplified.

In this paper the authors first start by introducing the problem and analysing previous research on the benefits of students learning with chatbots. They conclude that this kind of intelligent tutoring system has a positive impact on students and that it has an effect on motivation which can facilitate learning.

Then they introduce chatbot technology, starting by talking about the components of a chatbot, the responder, classifier and graph-master. They also talk about the AIML language and text processing response and action, but this information is all standard in chatbot papers and we have already learned about it.

The interesting part to our project starts when they analyse and tests chatbot platforms to find the best one. The platforms analysed were Dialogflow (API.ai), Wit.ai, Luis.ai and Pandorabots. To test these platforms, they trained all of them with the same knowledge and then tested with the same input. From these tests they concluded that Pandorabots matched the least number of intents. In the case of false intent id matches, Wit.ai and Luis.ai scored a confidence score higher than Dialogflow and in this test a lower confidence score is preferred. They reached the conclusion that Dialogflow is the best tool based on the tests. Dialodflow also has a feature where the developer can add follow up intents.

After selecting the platform to build the chatbot, they explained how to build it. Dialogflow is a platform that has easy integration with a lot of platforms and devices. In this case the authors choose to make an android app and a web platform. For that they used node.js to build the server, BotUI JavaScript framework to create the interface and DialogFlow Hypertext Markup Language (HTML) and Android Software Development Kit (SDK) to learn and know how to contact the DialogFlow API. They also used a feature of DialogFlow that makes the developer able to easily add pre-made small talk intents to the chatbot, this enables the chatbot to answer questions like, how are you? making it a little more interesting for the learners.

With this paper we learned that a chatbot is a good tool to teach and get knowledge from, we also learned that the DialogFlow application is the best to make chatbots and learned more about developing chatbots in this app and how easy it is to do it.

2.3.3 TA-bot: An AI agent as a Teaching Assistant

TA-bot [13] is a chatbot developed to be a virtual teaching assistant with the objective to complement the human staff and enhance the learning experience. This bot is part of a family of bots where each bot has a different specialisation. TA-bot specialisation is to answer questions from the syllabus. In this paper the authors describe the architecture and design of the chatbot.

The architecture of this chatbot is composed of three tiers, the first one is the front-end used to interact with the bot, the second is the processing tier that handles the natural language processing and the final tier is the back-end that contains the knowledge of the bot. The front-end is in a Facebook Messenger application. The class that is going to use this bot uses a Facebook page as their main forum and this page is going to have the chatbot attached to it. The second tier is DialogFlow. The questions are going to go from Facebook to DialogFlow that is going to use its built-in natural language processor to process the data. The back-end uses Google App Engine and Google Big Tables. They use the Google infrastructure to simplify onto a single stack vector. Google DialogFlow also supports different integrations like Facebook Messenger, Telegram, Slack and more. In the figure below Figure 2.2, we can see the architecture of the chatbot.

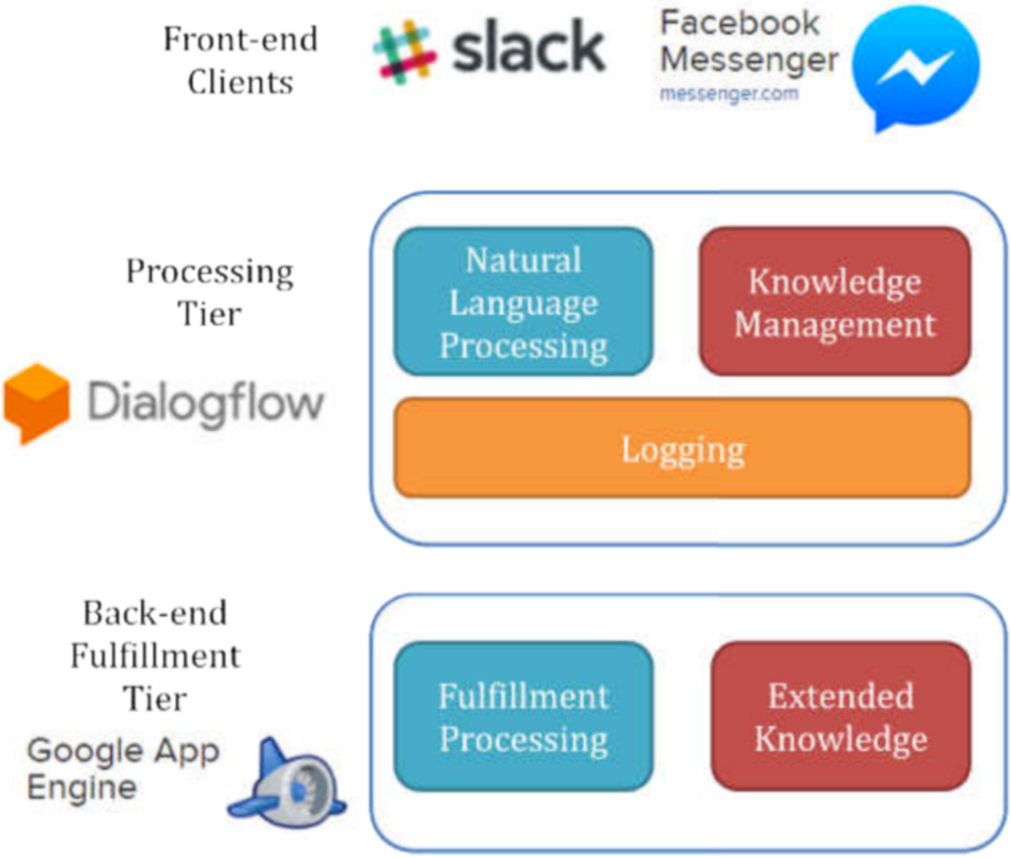


Figure 2.2: Architecture of TA-bot.

DialogFlow works with intents and entities. An intent is an intention of a user and in DialogFlow the developer can map the intent of a user to an action. In this case some of the intents of the chatbot are going to be, learning about deadlines for an assignment, inquiring about where the course meets and many more. An entity is anything that can be referenced like a textbook, the course, the classroom and so on.

There are two types of knowledge in the chatbot, “stable” knowledge like who teaches the class, where it is and “dynamic” knowledge, knowledge that changes every semester. They decided to make this part of the project in a separate back-end because it overcomes limitations in the processing tier, more complex queries can be made and the App Engine provides an easier interface to maintain and modify entities. In the App Engine, entities have a simple model, they can have some text-based information, some numeric information and/or date information.

In the end, after developing and testing, the authors concluded that a virtual teaching assistance can be created quickly and easily with out-of-the box technology.

This paper is of great use to ours because it gives as an insight on how the architecture of a chatbot should be. They use the same tools as we are going to use and gave us a better understanding on how to use them and what to use to create a chatbot using DialogFlow, like Google App Engine to do back-end of the chatbot.

3

Proposed Solution

Contents

In this chapter we are going to talk about the bot we created and how we did it. We will start by naming the features of the bot. Then we will explain the architecture and how we made it. We will end by describing the difficulties and problems that we had while developing it.

3.1 Features

The objective of this project was to make a personal assistant that can help and improve IST students university experience. The project was set to use the student's context to give them needed information. Although more context information was set to be used, due to time constrains and problems with the information, we had to restrict it to student's information given by Fenix. Like their schedule, courses, payments, curriculum and rooms information.

The user can communicate with the chatbot via text or speech. If the user communicates with text the chatbot will only respond with text, but if the user uses speech the chatbot will read the response. The chatbot responses are given in simple phrases, lists or links to websites.

With the information given by the Fenix API, we were able to make the following features to our personal assistant:

- Ability to go to the IST website in the assistant. It responds with a button, clinking on it redirects the user to the website.

- Gives a list with the title and description of the 10 most recent news in the IST website. If an item is pressed the user will be redirect to the news webpage.

- Gives the current academic term (ex: 1º Semestre 2019/2020)

- Gives information about a given IST service, for example the email of the management council. If the service or the information asked are not given by the user a list with the available options is given to the user. First a list with all the IST services available and then a list with the information available to that service, the information can vary from service to service.

- When asked about information for a degree a button with a link to that degree Fenix webpage is returned. The user can click the button to go to that webpage, where all of the information about that degree is.

- Gives the available parking spots in Alameda Campus and in the Saldanha parking area.

- Gives the location of the Alameda parking area and the Saldanha parking area.

- Gives the next hour that a shuttle will come in a shuttle stop. If a shuttle stop is not given by the student a list of the stops will appear, the user just needs to click the one he wants.

- Gives the schedule of the shuttles in a certain shuttle stop. The schedule is returned in the form of a list with hours.
- Gives the user's IST email.
- Gives a list of the courses that the user is enrolled. If he presses a course, it will take him to the course Fenix webpage.
- Gives the user's average in a finish course. If no course is given a list of the user's finished courses is given, if the user is enrolled in more than one degree a list of the degrees is returned first.
- Gives the user's average in a degree. If he has more than one degree a list of the degrees is given for him to choose.
- Gives the user's finished credits in a degree. If he has more than one degree a list of the degrees is given for him to choose.
- Gives the amount the user still has to pay in tuition fees.
- Gives the next payment that the user has, the date it expires, the amount, entity and reference.
- Gives the hour of the next class and the classroom number.
- Gives the date, the classroom and the course of the user's next evaluation.
- Gives the canteen menu for the day.
- Gives a list with the 10 most recent announcements of a course, if an item is clicked it redirects the user to the announcement Fenix webpage. If no course is given a list of the user's courses is returned. A suggestion is also given to receive notification on new announcements.
- Gives the option to receive notifications on new announcements, this option needs to be turned on by asking for it to the assistant.
- Gives a list with the 10 most recent summaries on a given course, if an item is clicked it redirects the user to the summary Fenix webpage. If no course is given a list of the user's courses is returned.
- Gives the group capacity for a given course, minimum and maximum capacity. If no course is given a list of the user's courses is given.
- Gives the enrolment period for group creation in a given course. If no course is given a list of the user's courses is given.

- Gives the user's group number, shift and members names in a course. If no course is given a list of the user's courses is given.

- Gives professors emails for a course. If no course is given a list of the user's courses is given.

- Gives the schedule for a given IST classroom. The schedule is presented in the form of a list with the class type, start hour and end hour (ex: Test starts: 10:30 ends: 12:30).

- Can give a list of all the features given above. Clicking on the name of one triggers it.

- Has a log in option when the user tries to access personal information that needs a Fenix permission. The log in only needs to be done once, first the user authorises the assistant to access is google information and then needs to log in into Fenix, using a link given by the assistant.

We came up with this list of features by trying to use all of the information available in the Fenix API. The API has a list of endpoints were the information is. As the endpoints return a JSON with the information, to create a feature, we just needed to think of a way to use that information. We tried to create features for all of the information available, but we think that there are endpoints that could be further explored. For example, the endpoint with information about spaces in IST as this information can be used to guide students through the campus or find rooms that don't have classes.

Our chatbot and the, already available, IST application have the same information in these categories: IST news; the cafeteria; shuttle; parking; information about courses, although in the IST application the user has to go to the course webpage to see group information, summaries and teachers emails while in the chatbot this is not necessary; evaluation information; curriculum; payments; and contact information. Features that our chatbot has that the IST app does not have are giving room information; giving the page of any IST degree; and giving notifications about recent announcements. The IST application can also give a ticket to go to a service while our chatbot cannot, as the ability to do this is not available in the Fenix API.

3.2 Arquitectural Overview

In this section we are going to talk about the technologies that we used to develop the chatbot. Most of the technologies will be services by Google, but we will also use the Fenix API to get information about the students. The figure below,Figure 3.1, shows how the project communicates between the different used tools.

As seen in the figure the architecture of the project is composed of the Google Assistant app, Dialogflow, Firebase and some external APIs:

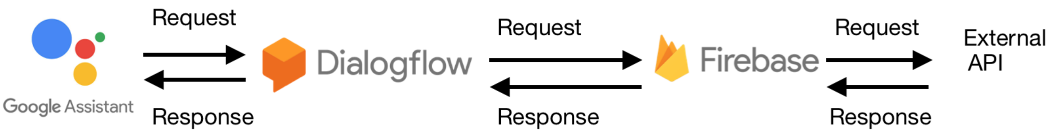


Figure 3.1: Tecnico Assistant architecture.

The Google Assistant app: Is our interface and is where the user inputs requests. It then communicates with Dialogflow by sending what the user wrote, so that Dialogflow can interpret what the user said.

Dialogflow: Will get the answers to the user. Dialogflow job is to interpret the user query and infer the right way to treat the query, when it does, it sends information about which code to run and the information that the code needs to Firebase. It receives the response that has to give to the user by Firebase.

Firebase: Where the code is located. The code in Firebase generates an answer to give the user with the information from the user query and information that is requested to different APIs. Firebase also has the database. All of the code in Firebase is written in Node.js.

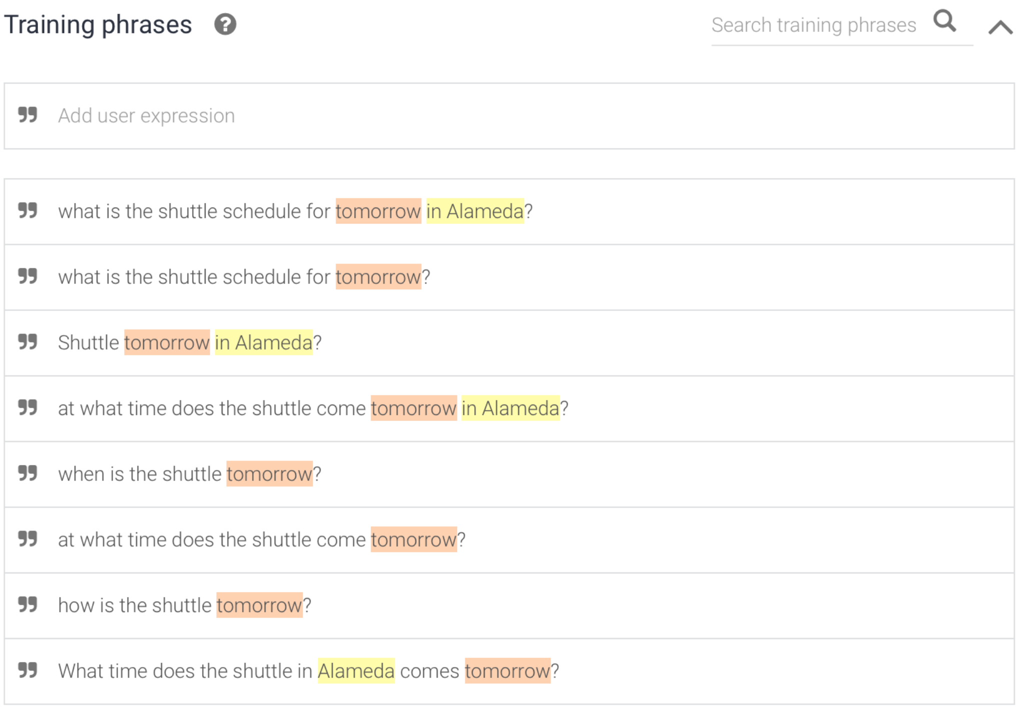
The external APIs: FenixEdu API and Google Cloud Scheduler. These APIs have endpoints to be contacted in order to get information, the information is returned in a JavaScript Object Notation (JSON) object.

All of these technologies and how we used them in our project are going to be explained in more detail in this chapter.

3.2.1 Dialogflow

Dialogflow is a Google platform that enables users to build voice and text-based conversational interfaces. The platform has all the logic implemented to handle the human-computer interaction. A Dialogflow application first starts with an agent that comes with default fallback and default welcome intents, the default fallback intent responds to the user when the chatbot does not know what to say and the default welcome intent welcomes the user to the chatbot. The agent helps in processing user input into structured data that can be used to return an appropriate response. The developer only has to fill it with intents and make the logic to create the responses. It was because of this lack of needing to worry with human-computer interaction logic, and the ease to create intents that we chose this platform.

We first started our app by creating the agent. We then filled it with intents, we made a total of 40 intents, each of them resolves a different query by the user or responds to an app event. For example, the getNextPayment intent is triggered when someone asks for their next payment, while the setupPush is triggered when a user wants to receive notifications. An intent defines how to map user input to a corresponding response. This is done with the help of the training phrases, a section of the intent where examples of user queries are put, as we see in Figure 3.2. With these examples the agent can see which intent training phrases is the closest match to the user input. We have around 10 training phrases for each of our 40 intents.



what is the shuttle schedule for tomorrow in Alameda?

what is the shuttle schedule for tomorrow?

Shuttle tomorrow in Alameda?

at what time does the shuttle come tomorrow in Alameda?

when is the shuttle tomorrow?

at what time does the shuttle come tomorrow?

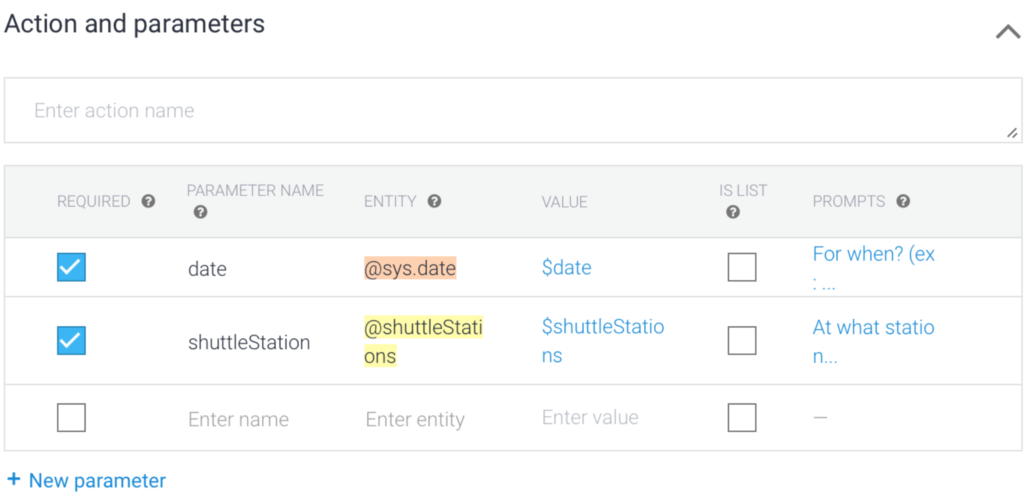
how is the shuttle tomorrow?

What time does the shuttle in Alameda comes tomorrow?

Figure 3.2: Dialogflow training phrases.

When handling queries by the user we may need some information from the query, like the shuttle station or the schedule of the shuttle. This type of information is called an entity. Entities are also specified in the training phrases. An entity can be system specific like @sys.date or can be created by us like @shuttleStation. For example, in the sentence 'when is the shuttle in alameda tomorrow?', alameda would be a value of the entity shuttleStation while tomorrow would be a sys.date entity. We can see in Figure 3.3, entities in an intent, the date entities are orange while the shuttleStations ones are yellow. If we notice in Figure 3.2, the entities in the training phrases are selected with the same color scheme.

We used system entities like date and any. The entity any matches everything, it does not have a specific value. We also created 6 entities. As seen in Figure 3.4 an entity created by us is a list of different reference values, on the left, and the synonyms that match the value, on the right. When an entity is matched in a sentence, the reference value will be passed to the code that resolves the intent.



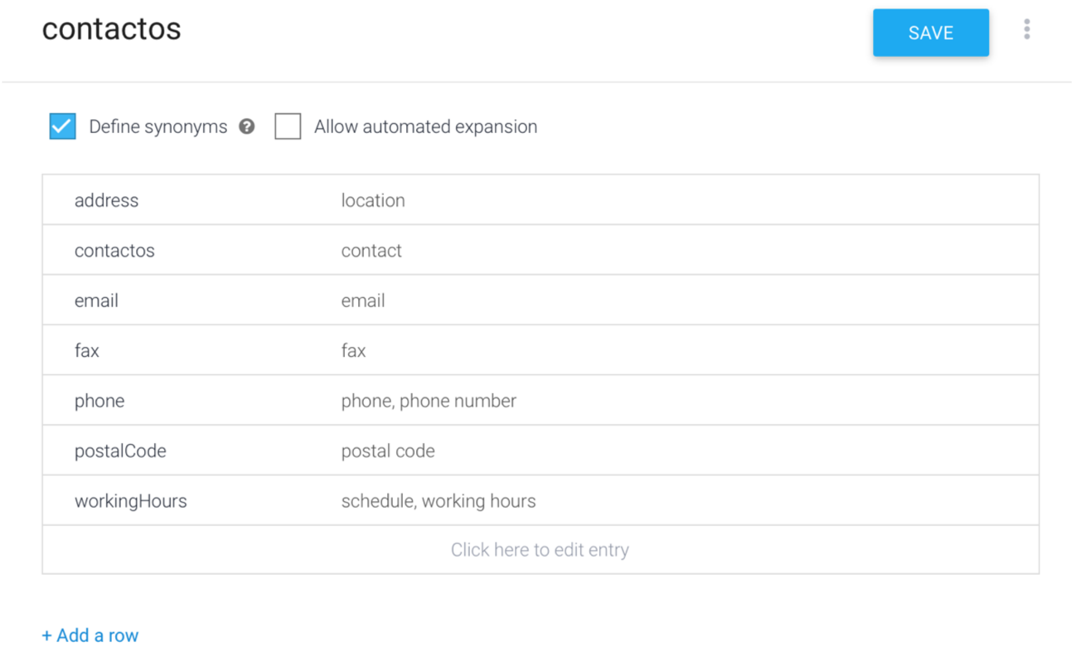
Action and parameters

Enter action name

REQUIRED	PARAMETER NAME	ENTITY	VALUE	IS LIST	PROMPTS
☑	date	@sys.date	$date	□	For when? (ex : ...
☑	shuttleStation	@shuttleStations	$shuttleStations	□	At what station...
□	Enter name	Enter entity	Enter value	□	—

+ New parameter

Figure 3.3: Dialogflow action and parameters



contactos

☑ Define synonyms □ Allow automated expansion

address	location
contactos	contact
email	email
fax	fax
phone	phone, phone number
postalCode	postal code
workingHours	schedule, working hours
Click here to edit entry	

+ Add a row

Figure 3.4: Contactos entity

Our six entities are: contactos, that has the information available in a IST service, like fax, phone, postalCode etc; degree that has all of the IST degrees; estacionamentos, that has the IST parking locations; IST, that has all of the different ways that a student can refer to IST, for example Técnico, IST, Instituto Superior Técnico, this was used to facilitate the writing of the training phrases; istPlaces has all the IST services that a student can access information of; and shutlleStations has all the shuttle stations. When creating each training phrase in our intents we had to identify the entities present in the phrase.

In an intent there is also a field called event, in this field an event name can be put and the intent will trigger when that event is called. For example, when our code sends the event to set up notifications, the intent that has that event is triggers. Events can be of the system or created by the programmer. In our intents list we have intents that only exist to respond to system events, these intents have no training phrases, only events attached to them. They exist for our code to run when that event is triggered. System events are User Interface (UI) specific events, Log in events, push notification events and the exit event. Some intents also have events created by us so we can call them. For example, when a user clicks an option in the available information list an event is triggered to call the intent the user wants.

The last part of Dialogflow that we used was Fulfillments. Fulfillments connects the intent to the code that resolves it. With this the code is hosted elsewhere, and the developer only needs to specify, in Dialogflow, an endpoint to communicate with the code. When an intent is called Dialogflow will send an Hypertext Transfer Protocol (HTTP) POST to the endpoint with information about the intent, like the name, the entities present in the users question and the question.

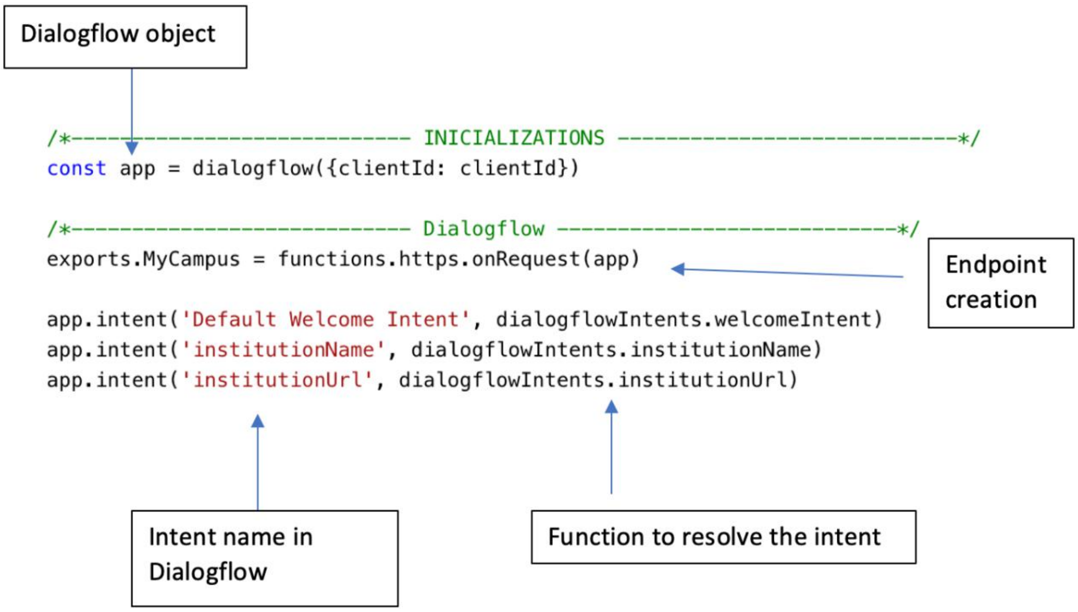
3.2.2 Firebase

Firebase is a Google platform that offers several cloud services like cloud storage, authentication services, website hosting and more. This is where all the code and data are, we chose this platform because it has an easy integration with Dialogflow.

In Firebase we first used Cloud Functions, a service that allows us to host and execute code in the Google Cloud. With Functions and the Actions on Google Client Library is easy to treat the Dialogflow post requests, this requests come with a JSON object. The Actions on Google Client Library abstracts the JSON and how to treat the information of the request, it enables us to treat the requests easily.

In Figure 3.5 we show how we create an endpoint to receive Dialogflow post requests and how we add functions to resolve each intents request. In the first line we start by creating a dialogflow object, named app, this object comes from the actions-on-google library and knows how to treat post requests coming from Dialogflow. After that, in the second line, we create an endpoint using a Firebase function, we pass the app object as an argument because this is the object that is going to receive the requests and treat them. The only thing that is missing now is showing to the app object what functions to use to treat each intent, for that we use the intent function of the object. As we can see in the final lines of the figure, when we call the intent function we pass it a string and a function, the string is the name of the intent in Dialogflow and the function is the one that is going to resolve that intents requests. So in line, 'app.intent('Default Welcome Intent', dialogflowIntents.welcomeIntent)' we are saying to the object that when a request from the Default Welcome Intent comes the function dialogflowIntents.welcomeIntent should be called. All of the functions that treat the intents request, like dialogflowIntents.welcomeIntent, only have one argument, conv, this argument is passed to them by the app object and has all of the information available in the HTTP POST JSON, sent by Dialogflow. The argument also has the logic to respond to Dialogflow, by calling the 'ask' function we can send a string to respond to the user. For example by calling 'conv.ask('Hello, how can I help you?')', we say to the user 'Hello, how can i help you?'.

This was the most complicated part of the project, we had to create logic to respond to each intent. Each function gets different information from the Fenix API and treats it in a different way to get a response. To help the development of our intents we created two libraries. The first one, fenixAPI.js, has all of the logic needed to talk to the Fenix API. We created a function to talk to each endpoint available in the API, by calling one of these functions we get the information available in that endpoint. The second library, localResponses.js, has the logic to respond to a user already implemented. These libraries helped us to only worry about how to treat the information of an intent when we were doing a new intent function.

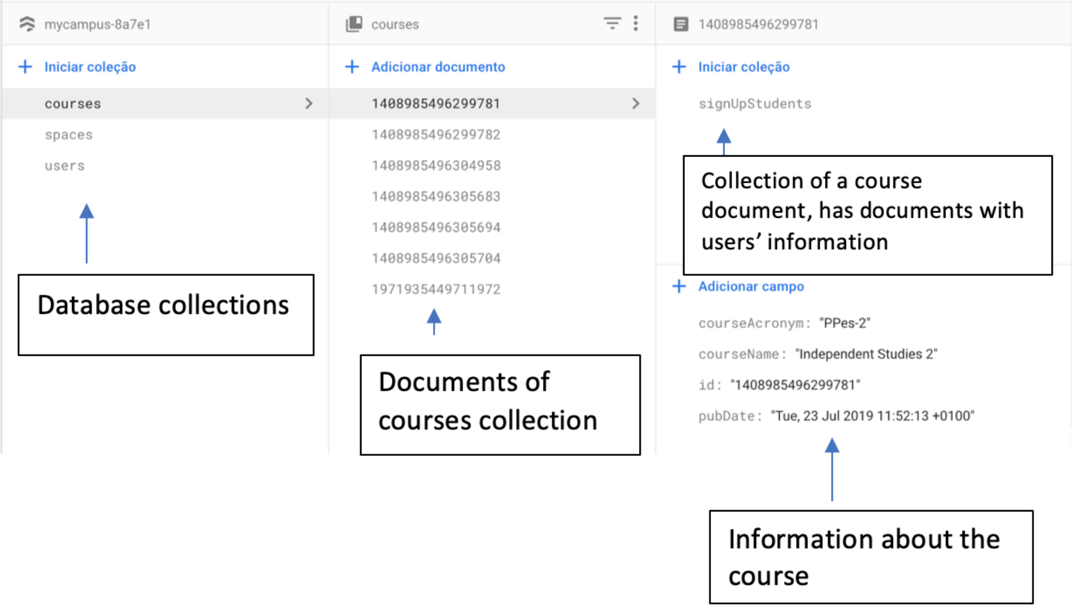


```
/*---------------------------- INICIALIZATIONS ----------------------------*/
const app = dialogflow({clientId: clientId})

/*---------------------------- Dialogflow ----------------------------*/
exports.MyCampus = functions.https.onRequest(app)

app.intent('Default Welcome Intent', dialogflowIntents.welcomeIntent)
app.intent('institutionName', dialogflowIntents.institutionName)
app.intent('institutionUrl', dialogflowIntents.institutionUrl)
```

Figure 3.5: Code from the project

Cloud Functions also allows monitoring of our functions and allows us to easily deploy the code with simple terminal commands in our own computer, after installing the Cloud Functions SDK.

The other service of Firebase that we used was Cloud Firestore a NoSQL database in the cloud. In this NoSQL database the information is stored in JSON format and the database is composed of collections. These collections are made up of documents that have fields and can also have collections inside of them. For example, one of the collections we have in our database is a courses' collection. This collection is used to store information needed to send notifications to users. It has documents for each course that students enrolled to receive notifications. Each of these documents has a name, four fields and a collection, we can see this in Figure 3.6. The fields are: the course acronym, for example PPes-2; the course full name; the course id; and the date of the last announcement in the course. The collection, inside the document, has documents representing the students that want to receive notifications about this course announcements. These documents have ids from google assistant that identify the user's device. We can see this collection in the figure, its name is signUpStudents. So, our courses collection is filled with documents and each of these documents represents a course, has some fields that store needed information, about that course, and a collection of documents that has documents representing users.



courses
spaces
users

1408985496299781
1408985496299782
1408985496304958
1408985496305683
1408985496305694
1408985496305704
1971935449711972

1408985496299781
signUpStudents

courseAcronym: "PPes-2"
courseName: "Independent Studies 2"
id: "1408985496299781"
pubDate: "Tue, 23 Jul 2019 11:52:13 +0100"

Database collections

Documents of courses collection

Collection of a course document, has documents with users' information

Information about the course

Figure 3.6: Firebase Database

We have two more collections in our database. A students collection that has documents representing students. These documents have for their name the student's email and are composed of three fields: one with the access token needed to access the students private information in Fenix; another with the refresh token to get a new access token; and the final one with the IST id for the student.

The final collection that we have is made up of documents representing spaces in IST, this collection is a copy of the information available in the Fenix API. We copied the information to the database because it was quicker to access than the API.

3.2.3 Fenix API

Fenix is where all of the student's information is located and we use the FenixEdu API to get all of the information in the chatbot. The API itself is a list of endpoints, to get the information in an endpoint a get request must be made, in response we obtain a JSON object with the demanded information. For example, to get information about contacts we make a HTTP REQUEST to `https://fenix.tecnico.ulisboa.pt/api/fenix/v1/contacts`.

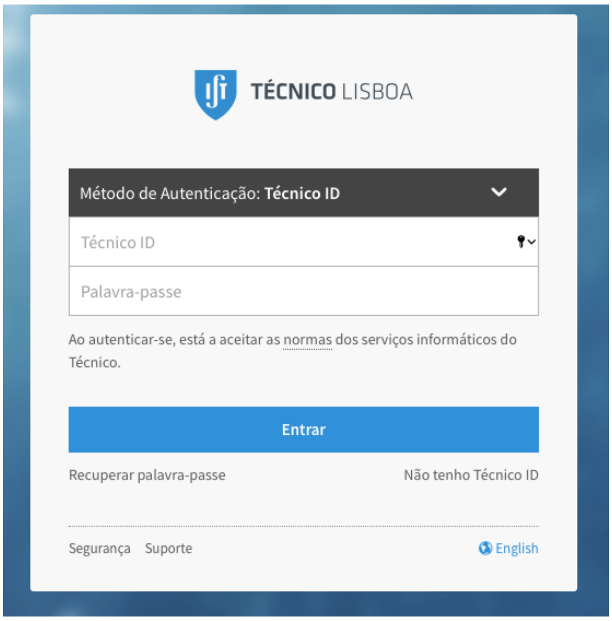
The endpoints that we use are the canteen menu information, that gives the canteen menu for the week. The IST contacts that gives us contacts like email, phone and address for various IST establishments. The information about parking in IST, the shuttle schedule and an endpoint that gives the academic terms that can be used in other endpoints.

There are three endpoints for degree information. The first one gives a list of all degrees in IST and some information about them like the id, the Uniform Resource Locator (URL) for the degree page and more. The second one can be accessed by using a URL like the one above but with the degree id and only gives information about that degree, for example we can access it with `https://fenix.tecnico.ulisboa.pt/api/fenix/v1/degrees/` + id + '?academicTerm=' + academicTerm. The third and final one, we don't use, but it can also be accessed with the degree id, gives all of the courses in the degree, the ids for them, as well as the name, credits and academic terms.

To get information about courses there are five endpoints, all of them are access with the course id. The first endpoint gives information like the URL page of the course, the teachers names, the evaluation method and more. The second one gives us the evaluations information. The third gives us the students groups for the labs. The fourth gives us the course schedule and the last one gives us a list of students in the course. We used them all except the last two.

To get information about a specific person there are seven endpoints. The first one gives specific information about the person. The second gives all of the dates for all of the student's classes. The third gives all of the dates for evaluations, this last two give the information in JSON or in ICal. The final ones give information about all the courses that the person is in, the curriculum, the payments that the person has done and has to do, written evaluations and an endpoint to enroll on those evaluations. We don't use this last one but use the rest of them.

To access a person's information an access token is needed, this token is given when a student logs in in a specific URL. For example, for our chatbot to get the access token of the user he has to log in in `https://fenix.tecnico.ulisboa.pt/oauth/userdialog?client_id=1414440104755269&redirect_uri=https://us-central1-mycampus-8a7e1.cloudfunctions.net/FenixApiEndpoint`, this URL was created so that when the user finishes login in to Fenix the access token is sent in an HTTP POST to an endpoint specified by us, this endpoint has the logic to store the access token in the users document, in the database. In Figure 3.7 we can see what appears to the user when he goes to the log in link.



Método de Autenticação: **Técnico ID**

Ao autenticar-se, está a aceitar as normas dos serviços informáticos do Técnico.

Figure 3.7: Fenix Log In

The final information that the Fenix API has is about spaces. Spaces are organized in a tree where the top space (Alameda, Taguspark) has buildings and the buildings have floors with classrooms, each of them is identified by an id. There is an endpoint that identifies the top spaces and another that gives the information about a specific space, when given the id. With this we can go through the tree and find schedules for rooms. There is also an endpoint that gives us a blueprint of the space, but this endpoint is not used in our chatbot.

3.2.4 Google Cloud Scheduler

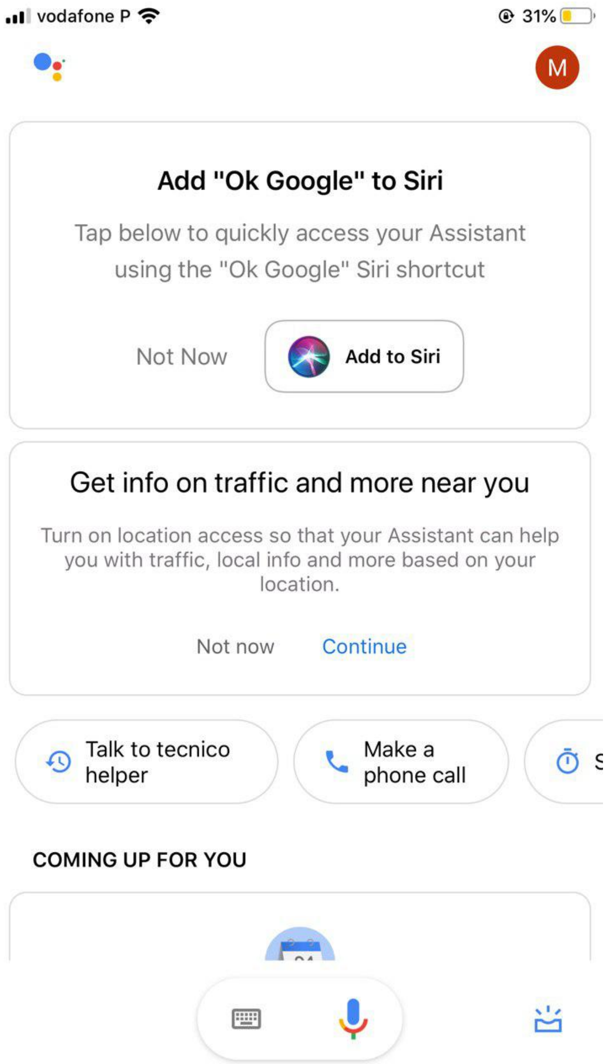
The Google Cloud Scheduler is used in our assistant to run a task at a specific time. We use this platform to once a day send a get request to a specific endpoint in our Firebase Functions. This endpoint is used to send notifications to all of the users that ask the assistant to receive them. We will talk more in detail about how we do this in this chapter.

3.2.5 Google Assistant

Google Assistant is a personal assistant made by Google, it can research on google, talk to the user, see the weather and much more. The assistant is available in Android phones, iPhones, computers and many more Android and Google devices. With the help of Actions on Google and Dialogflow, Google lets developers create apps for the assistant.

We used the Google Assistant as our user interface because it had an easy integration with Dialogflow and our intents, it saved us time developing an app to be our UI and it is available in a number of devices that our users use, the IST students.

The integration of our intents into the Google Assistant was made almost automatic, with a click of a button our intents were working in the Assistant. Google also has a simulator so we can test on any device, in our computer. This was available in the Actions Console. We can see the Google Assistant application in Figure 3.8



vodafone P

31%

M

Add "Ok Google" to Siri

Tap below to quickly access your Assistant using the "Ok Google" Siri shortcut

Not Now

Add to Siri

Get info on traffic and more near you

Turn on location access so that your Assistant can help you with traffic, local info and more based on your location.

Not now

Continue

Talk to tecnico helper

Make a phone call

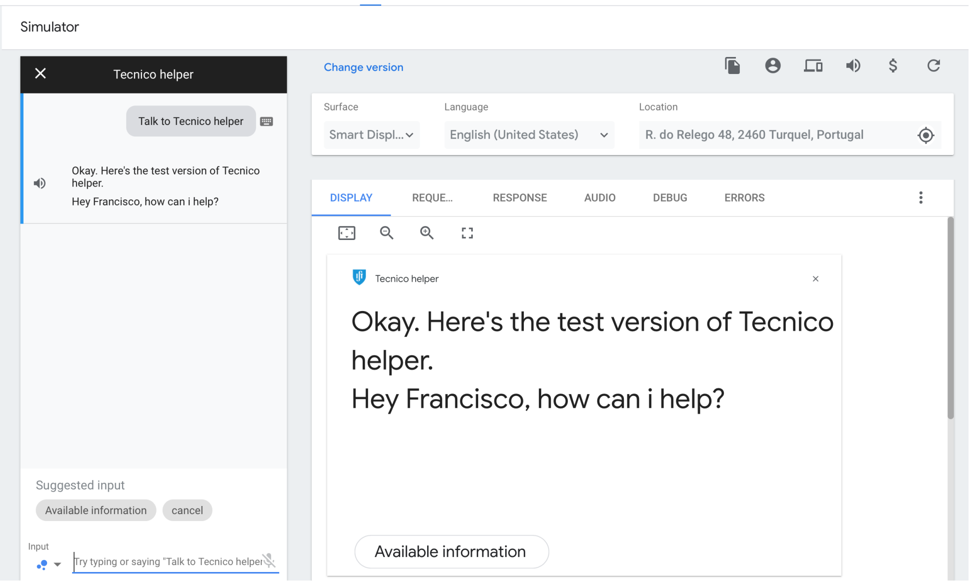
COMING UP FOR YOU

Figure 3.8: Google Assistant IPhone Application

3.2.6 Actions on Google

Actions on Google is a platform that enables developers to make apps in the Google Assistant. An action is support for a user request, it is represented by an intent and his fulfilment [1]. When we integrate our Dialogflow project with Google Assistant our intents became an action in the Actions console. They appear in the list of actions of our app.

The Actions console enables us to test our app with a simulator that can simulate a phone, speaker or smart display. We can also change the language of the assistant or the location. We can see the request and response packets and most important we can deploy our app. The deployment can be done to be an alpha test, a beta test or a full release of the app. To deploy the app the developer needs to fill a list of information about the app, like the logo, the privacy policy, category of the app and many more. We can see the simulator in Figure 3.9



Talk to Tecnico helper

Okay. Here's the test version of Tecnico helper.
Hey Francisco, how can i help?

Surface: Smart Displ...

Language: English (United States)

Location: R. do Relego 48, 2460 Turquel, Portugal

Try typing or saying "Talk to Tecnico helper"

Figure 3.9: Actions Simulator

As mentioned before we also use the Actions on Google Client Library [2], this library enables us to use different features of the assistant like certain UI elements and interact with Dialogflow. In our project we code and use this library in node.js.

3.2.7 Features and Actions on Google Client Library

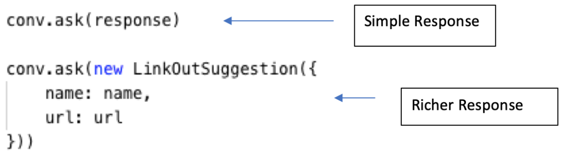
As we mention before we used the Actions on Google Client Library too easily communicate with Dialogflow and to use features of the Google Assistant. With this library all of the functions that treat the intents request have, as an argument, the conv object. This object has information about the user query

[1] https://developers.google.com/actions/overview

[2] https://actions-on-google.github.io/ actions-on-google-nodejs/2.12.0/index.html

and a function that responds to the user, sends the response to Dialogflow. This function is the ask function.

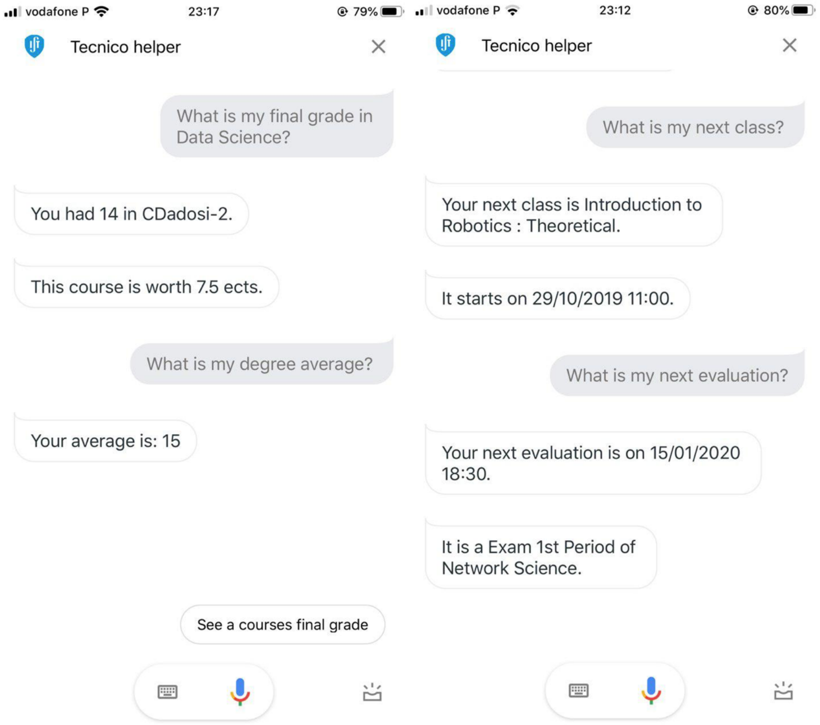
The ask function can return a simple response to the user, like a string, or it can return an object that represents a richer type of response, we can see this in Figure 3.10. In our chatbot we used several richer types of responses, a BrowseCarousel, Suggestions, LinkOutSuggestions and Lists [3]. In this subsection we are going to talk about these types of response, what features use them and how they look on our chatbot. All of this responses support text to speech, the chatbot can give them in the form of voice and text at the same time.



```
conv.ask(response)

conv.ask(new LinkOutSuggestion({
    name: name,
    url: url
}))
```

Figure 3.10: Examples of the ask function

First, we have the simple responses, this is when the chatbot responds with only a string. In almost all of the features the response comes in this way. We can see some examples in Figure 3.11.



What is my final grade in Data Science?

You had 14 in CDadosi-2.

This course is worth 7.5 ects.

What is my degree average?

Your average is: 15

See a courses final grade

What is my next class?

Your next class is Introduction to Robotics : Theoretical.

It starts on 29/10/2019 11:00.

What is my next evaluation?

Your next evaluation is on 15/01/2020 18:30.

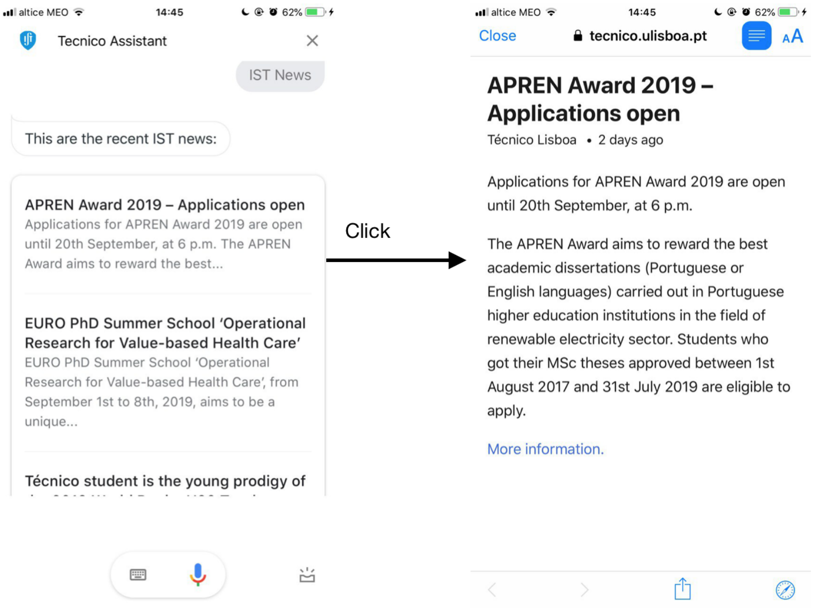
It is a Exam 1st Period of Network Science.

Figure 3.11: Examples of simple responses

The first richer response that we used was the BrowseCarousel. The BrowseCarousel is a rich

[3] https://developers.google.com/actions/reference/rest/Shared.Types/AppResponse#carouselbrowse

response that allows users to scroll verticality between a list of items and select one. When an item is selected the content of the item will be open in a web page or in an Accelerated Mobile Pages (AMP) browser. This type of rich response is designed to show web content, it has a minimum of 2 items and a maximum of 10 items. We used this to show the IST news, the user's courses, announcements and summaries. All of the items in the BrowseCarousel have a webpage attached to it. We can see an example of this type of response in Figure 3.12. The first photo shows a BrowseCarousel with the IST news and the second shows the webpage that appears when we click in the first item.



IST News

This are the recent IST news:

APREN Award 2019 – Applications open
Applications for APREN Award 2019 are open until 20th September, at 6 p.m. The APREN Award aims to reward the best...

EURO PhD Summer School 'Operational Research for Value-based Health Care'
EURO PhD Summer School 'Operational Research for Value-based Health Care', from September 1st to 8th, 2019, aims to be a unique...

Técnico student is the young prodigy of

APREN Award 2019 – Applications open

Técnico Lisboa • 2 days ago

Applications for APREN Award 2019 are open until 20th September, at 6 p.m.

The APREN Award aims to reward the best academic dissertations (Portuguese or English languages) carried out in Portuguese higher education institutions in the field of renewable electricity sector. Students who got their MSc theses approved between 1st August 2017 and 31st July 2019 are eligible to apply.

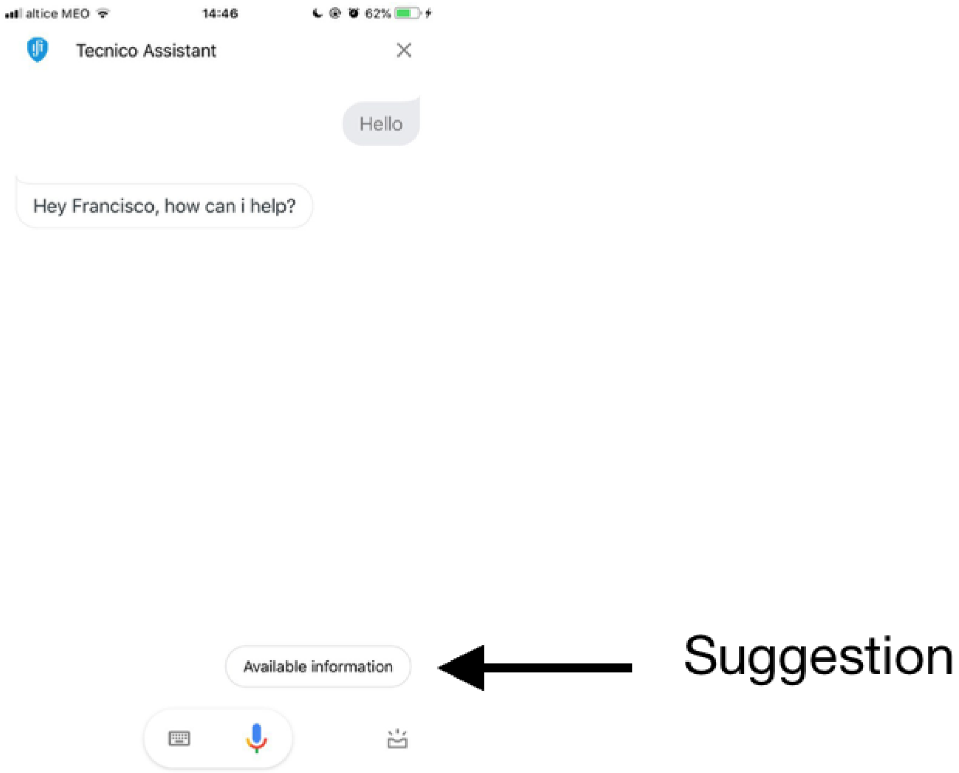
More information.

Figure 3.12: News BrowseCarrousel from Tecnico Assistant and AMP browser from when an item is cliked

The Suggestions response, as the name indicates, is a suggestion to the user. Suggestions are used to suggest intents, to steer the conversation or give related information. When a suggestion is used a button will appear in the base of the assistant, if the button is pressed the suggested intent is triggered, we can see this in Figure 3.13. We used Suggestions in: the Welcome intent to suggest the list of available information; when we give the available parking spots, to suggest asking about the location of the spots; when the user asks the degree average, we suggest seeing a course average; and when the user sees his announcements, we suggest allowing announcements notifications.

A LinkOutSuggestion is just like a Suggestion, a button that appears in the base of the assistant, but when pressed sends the user to a website. We use this to get the user to the Fenix authorisation website, to give the user the IST website and to give the user a Fenix degree webpage.

A List is just like the names suggests, a list of items. Differently from the BrowseCarousel when a user clicks an item a webpage does not appear but rather something programmed by the developer will happen. A list can also have a larger number of items, going from the minimum of 2 items to a maximum



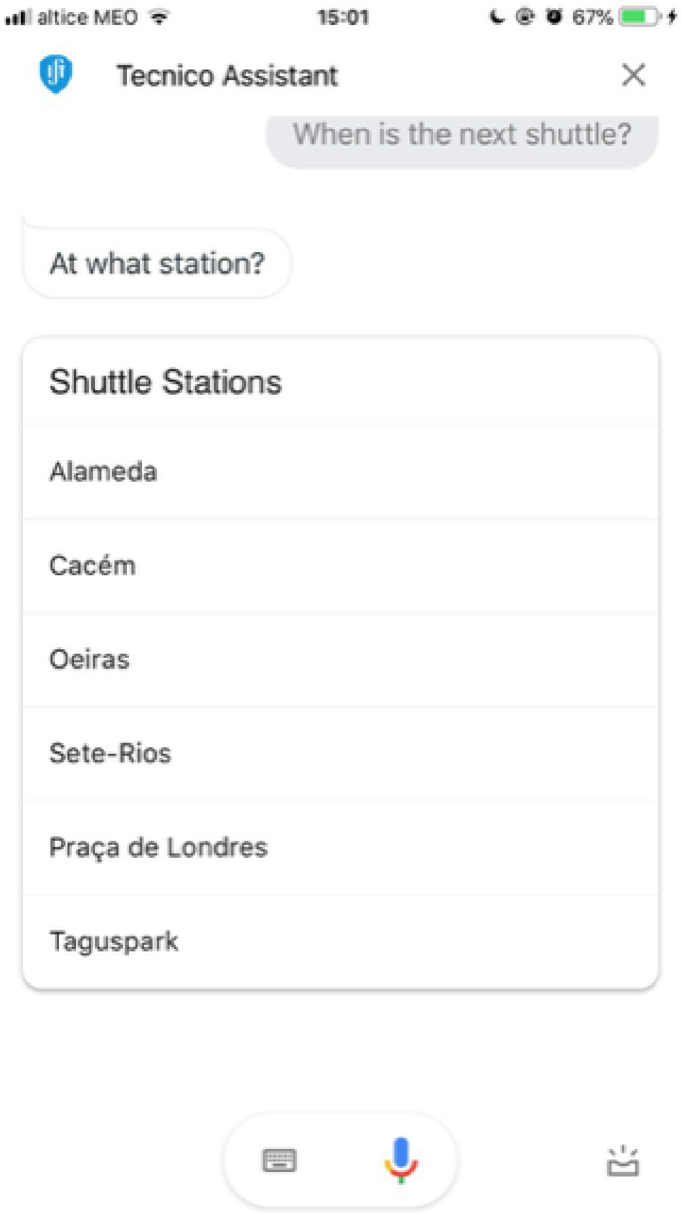
Hello

Hey Francisco, how can i help?

Figure 3.13: The Suggestions response in the opening intent

of 30 items. We used this to show the shuttle schedule, where each item is a time the shuttle will be in that station, as we see in Figure 3.14. But we used it primarily to know what the user wants when he does not say. For example, if the user says, ’when is the next shuttle?’ the assistant does not know of which shuttle stop the user wants this information. So, the assistant will show the user a list with all the shuttle stops for him to choose, when the user clicks the one we wants, the next shuttle hour, in that stop, will be responded. We use this instead of follow-up intents because we want to give the user the options he can choose from, this way he does not make mistakes typing it. Another use of the List is the available information list that is suggested in the welcome intent, this list has all of the features in the chatbot, if one is pressed the intent for that feature will be called.

The log in of a user is made using a LogIn object from the Actions on Google Client Library, this object triggers intents that are going to ask the user if it’s ok for the application to access user information, like the user’s email. If the user says yes and the log in is successful the application will have access to this information and will create an entry for that user in the database. The id for this entry is the user’s email and it stores the user IST id, the access and refresh token for the Fenix API. These tokens are used to access personal Fenix information about the user. After the user says yes for the Google Assistant log



When is the next shuttle?

At what station?

Shuttle Stations

Alameda

Cacém

Oeiras

Sete-Rios

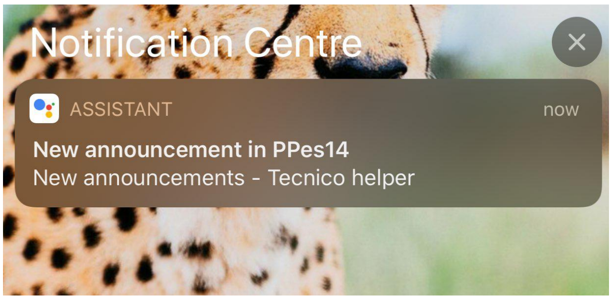
Praça de Londres

Taguspark

Figure 3.14: List of shuttles available

in, our app sends him a LinkOutSuggestion that sends the user to the Fenix log in, once the log in is made we receive the two tokens and the log in is finished. The log in only needs to be done once and will trigger when needed.

To send notification to users our app uses an id that is specific for each user. When a user says he wants to receive notifications he is registered in the database in each of his courses. Each course has a database entry with a students list, and information about the last announcement. Once per day all of the courses in the database are cheeked for new announcements and the students are sent a notification. The hour and the times that the courses are checked can be changed in Google Cloud Scheduler. We can see an example of a notification in Figure 3.15.



Notification Centre

ASSISTANT now

New announcement in PPes14
New announcements - Tecnico helper

Figure 3.15: List of shuttles available

3.3 Difficulties and Problems

While developing this application one of the biggest problems was understanding how Dialogflow and the Actions on Google Client Library work. There are many ways of doing the same thing and more Google Libraries that do the same. Although Google has a lot of documentation it sometimes is too much and very confusing, having different ways of doing the same operation and deprecated information. In the end we were able to understand it and to use it like we described in this chapter. The other major problem for developing our project was the Fenix API, it was very difficult to know what kind of information the API would return and the only away to know was by sending requests. We were going to make the assistant capable of helping staff and professors but without a staff or professor account to test and know the information that the API would return we were not able. Another feature that we were not capable of execution was the assistant telling students empty classrooms, although the API has information about all IST classrooms and their schedules it does not have a flag to distinguish a room were students can study from , for example, a bathroom or professors office. There is also a lot of incoherent information. For example, we thought that we could use room capacity to distinguish between a room and a WC, but in some buildings the bathroom does not have a capacity and in others it has.

4

Work Evaluation

Contents

To evaluate our system we decided to first test its performance, comparing to other IST technologies, its usability, using the System Usability Scale (SUS), and get the user's opinion on it.

We made the test to twenty users, all IST students. To test the performance we gave the testers some tasks to complete, from there we collected metrics like the time to complete a task and number of errors the user made. In the end the user answered a questionnaire where we used the SUS and other questions to test the usability of the system and the user satisfaction.

We asked our twenty users about their gender, age and the number of years they are enrolled in IST. We can see the results in table 4.1.

Table 4.1: Gender and Age of the Testers

Gender		**Age**			**Enrollment Period**					
Male	**Female**	**18-21**	**21-27**	**>27**	**1**	**2**	**3**	**4**	**5**	**>5**
15	5	0	20	0	5	2	0	3	6	4

All of the students that made the test were between 21 and 27 years of age, they were all doing their masters but their knowledge and experience of IST varied, as some students only had been there for one year and others over 5.

4.1 Tasks

We started the test by explaining to the users how the app works, we also gave them 5 minutes to explore the chatbot. All the users did the tasks in an iPhone 8 Plus. After that we gave him eight tasks to complete. First on the chatbot and then on another IST technology. We can make 25 different tasks with the features of our chatbot. But to make the test less lengthy we chose the eight that we thought were more important and represented the chatbot:

T1 - Find the next shuttle in Alameda.

T2 - Find the user next payment.

T3 - Find the announcements of a course.

T4 - Find the next evaluation of the user.

T5 - Find the schedule of a classroom.

T6 - Find the final grade of a user's course.

T7 - Find the user's next class.

T8 - Find the user's IST email.

All tasks were given a start point and an end point, so that we can register the time they took. The tasks were first done on the chatbot, with the end point being when the chatbot gave the information asked. The other IST technologies were the IST website, for the first task, and the IST Fenix website for the others, the endpoint was when the user reached the information asked. To evaluate the chatbot in these tasks we recorded two metrics:

Time to complete the task - the time that the user took to complete the task.

Number of errors - the times the task had to be started over.

With data from the times the testers took to complete the tasks we calculated their average and standard deviation. This information and the errors count is in table 4.2.

Table 4.2: Tasks times and errors results

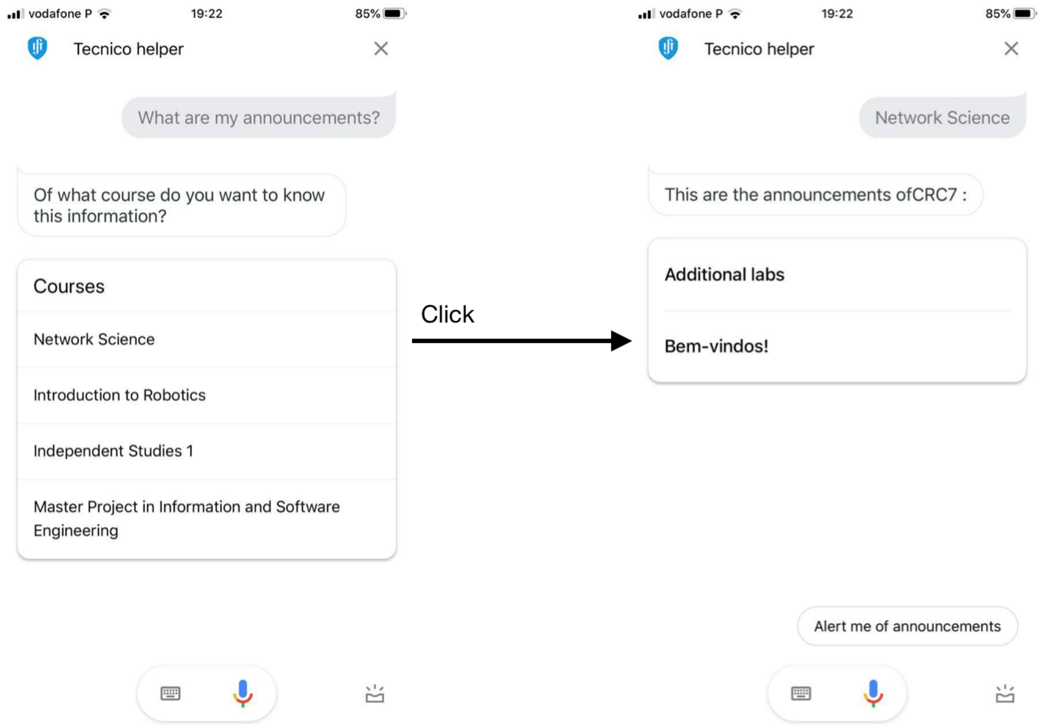
	Average Time (s)		**Errors**	
Tasks	**Chatbot**	**Alternatives**	**Chatbot**	**Alternatives**
T1	11,69 ± 3,51	23,69 ± 7,82	0	10
T2	8,32 ± 2,03	13,82 ± 5,78	1	3
T3	16,18 ± 5,96	7,94 ± 2,76	3	0
T4	9,62 ± 3,89	10,17 ± 2,39	2	0
T5	11,57 ± 3,79	22,98 ± 13,50	3	8
T6	13,19 ± 3,47	16,44 ± 6,96	2	5
T7	8,05 ± 2,90	8,66 ± 3,11	1	0
T8	8,27 ± 2,56	11,10 ± 5,18	1	4

As seen in the table, the chatbot has better times than the alternatives, except in task three, we can conclude that the chatbot is quicker than the Alternatives.

Regarding the alternatives times, we can see that the quickest tasks, task three, four and seven are tasks that the student is more used to do, they are seeing an announcement, the students next evaluation and his next class. These tasks are automatic for the students and are easily done. While tasks like seeing the next shuttle date, task one, and seeing a rooms schedule, task five, are slower because they do not use it as often. Task one, was the only one done in the IST website and the one that was more hidden and harder to get to, some students said that they don't even use the IST website to find this information but Google, to get to the right place of the website, as is quicker and easier. We can also see that task one and five are the ones with higher standard deviation, this is because some students knew where this information was and others not. Interestingly we observed, during the tests, that students from Taguspark were quicker doing task one and slower doing task five, as they use the shuttle frequently and the Taguspark campus is small. While students from Alameda were quicker using task five, because the campus is bigger, and slower doing task one because they do not use the shuttle as frequently.

Regarding the chatbot times, the one that was slower that when preformed in the alternative was task three. This can be because if a student does not specify the course that he wants to know the announcement, options appear for him to choose the course, as seen in fig. 4.1

We can actually see that the tasks that take more time in the chatbot, task one, three and six are all tasks when if the student do not give enough information an option will appear for them to choose. This



What are my announcements?

Of what course do you want to know this information?

Courses

Network Science

Introduction to Robotics

Independent Studies 1

Master Project in Information and Software Engineering

Network Science

This are the announcements ofCRC7 :

Additional labs

Bem-vindos!

Alert me of announcements

Figure 4.1: Options in tecnico helper.

of course causes the completion of the task to be slower.

About the errors, almost all of the chatbot tasks had at least one error, this was because sometimes the chatbot did not understood what the user wanted from his question. As we said before, the chatbot is trained with example phrases, and if the user's question was too different from the example phrases, the chatbot said that he did not understood the question. This type of error only appeared in the beginning of the tests because as soon as they appeared we trained the chatbot. Another type of error was when students used speech to talk to the bot and it would understand them badly. Also, when a user did not write a classroom name correctly, for example a1 instead of A1, and sometimes, as we use the parameter any to get the classroom name and this parameter can be anything, it would recognise Room A1 as the room name instead of just A1. In task six, if the user did not specify the course, he wanted to know the final grade, the chatbot, sometimes thought the user was asking about the degree final grade. Because the way they are both asked is similar. For example, the user would ask 'what is my course average?' and the chatbot would give the final average of the user's degree. This happened because in the training phrases of the intent, that gives the degree average, is the phrase 'what is my average?'. So, the chatbot thought that the phrase 'what is my course average?' was closer to the training phrases of the intent that gives the degree average than to the intent that gives the course average. This can be fixed by adding 'what is my course average?' to the training phrases of course average.

The errors in the alternatives were all because the students had difficulty reaching the information and had to take several paths to reach it. As we can see, the number of errors in the alternatives are greater than in the chatbot, and the errors are in tasks that the students don't do very often or where the information is hard to find.

With the data from the times and the errors we can conclude that our chatbot preforms better in tasks that the user uses less often, as it gives an easy way to find the information, while in the alternatives the information can be hidden and hard to find. We can also conclude that the chatbot becomes better as testing is done, and the training phrases are updated, the use of 'any' as a parameter can create some errors that can be bad for the user experience and that the bot can sometimes miss match some questions, if there are similar alternatives.

4.2 Questionnaire

After completing the tasks the students were asked to respond to a questionnaire, appendix A. The questionnaire was divided in three parts. In the first part we collected information about the tester, in the second we made questions about the SUS, that we are going to explain bellow, and in the final part we asked the tester about the chatbot.

We already talked about the first part of the questionnaire in the beginning of this chapter, there were twenty users being all of them from IST and doing their master degree.

The next section of our questionnaire was aimed at testing the usability of the system, for that we used the SUS [14]. The SUS is a 10 questions questionnaire with five responses each, that goes from strongly agree to strongly disagree. This system has references in over 1300 articles and publications and gives a reliable result in small sample sizes, as ours. To interpreter these results the system has a way of transforming the results into a grade, going from F to A [15].

After doing the calculations our system ended up with a B grade, this means that our system does not have usability problems but still has room to improve. We can also take from the first question of the SUS, I think that I would like to use this system frequently, that the students would like to use the system as they all responded with a 4 or 5, being 5 the closest to strongly agree.

The final part of our questionnaire was done to assess the chatbot and what the students liked about it. We started by asking them if they prefer to use the chatbot or the alternatives, all testers answered that they preferred the bot. This tells us that, for the students, the chatbot is a better and more comfortable way of getting information than the alternatives.

The second question gave a list of all of the chatbot features and asked them to choose their favorites. The three favorites, with 80% of the testers selecting them were: give the time of the next shuttle; give the students next class; give the students next evaluation. These three features are the ones that impact the students' academic life the most, his next class, shuttle and exam are information that he needs to know daily. This tells us that the information the students like the most are the ones that they need every day and are important to their academic life, while the IST news or get some IST service email were less popular features, as they don't affect their academic life very much.

The third question was optional and asked the tester if he thought that was something missing in the system. One tester said that a Frequently Asked Questions (FAQ) section was interesting, another asked for access to more information and a last one said that it would be interesting if the chatbot was, in the already available, IST mobile application.

Although the user did not use an app without the chatbot, the last question asked the testers if the chatbot functionality was something important or if they would prefer an application with the same information but without the chatbot. Only 2 students said that they would prefer something without the chatbot and 18 said they like the chatbot.

With the questionnaire we can conclude that the students liked the chatbot and they would use it. We learned that information that affects the student daily life is more important and the students actually care about the chatbot technology and not only the information available. We have also got some ideas to make the system better.

5

Conclusion

A student needs a lot of information for is normal academic life, in this project we tried to create an application that gives needed information to students in a more comfortable and quick way. We also tested how students engage with the chatbot technology, if it is a good way of getting information in an academic setting and how the new technologies to create chatbots work, especially Dialogflow.

We used Dialogflow and other Google technologies to develop the chatbot. We had more difficulties with Dialogflow at the start while we were still learning how it worked, it can be a tricky technology to understand when we need to navigate through countless documentations and there are many ways of doing the same thing. But after learning how it works, doing intents and new functionalities on the chatbot can be easy. Integrating with Fenix was also something difficult, especially trying to integrate the Fenix log in in the Google Assistant and the Google log in system.

Dialogflow also understood the testers really well, matching the questions of the users to the right intents, even if they were slightly different from the training phrases. It only had problems when two intents were similar and when the any argument was used, something that should be avoided. In the end Dialogflow was a good an easy way of developing a chatbot, with it we only needed to do the connection between it and the information.

After concluding the development of the system we tested its usability, performance and user opinion. The tests were very positive, our system is great in terms of usability and is performance is better that the alternatives that exist. By analysing the results we saw that even with the users favourite features, of the system, being the ones closest to the alternatives in performance, the users still preferred our system. This tells us that we achieved our goal of having a more comfortable way of giving information to students, and that they prefer a chatbot to give them information they use daily and not only information that they don't know where it is.

For future work, the chatbot can have more information and could be put in the IST mobile application, so that the users do not have to download another app to use it. Space information could be further explored and a Q&A section could be done.

In the end we were able to develop our system and deliver a product that the students of IST liked. We could also conclude that we reached our goal of developing a system that gives information to students in a more quick and comfortable way.

Bibliography

[1] Bayan Atwell; Eric Abu Shawar, “Chatbots: are they really useful?” *LDV-Forum: Zeitschrift für Computerlinguistik und Sprachtechnologie*, vol. 22, no. 1, pp. 29–49, 2007.

[2] Artificial Intelligence Foundation, “The A. L. I. C. E,” 2007. [Online]. Available: http://www.alicebot.org

[3] J. Weizenbaum, “ELIZA&Mdash;a Computer Program for the Study of Natural Language Communication Between Man and Machine,” *Commun. ACM*, vol. 9, no. 1, pp. 36–45, 1 1966. [Online]. Available: http://doi.acm.org/10.1145/365153.365168

[4] T. Conference, P. P. Wydawnictwo, F. Uniwersytetu, A. Mickiewicza, B. A. Shawar, E. Atwell, and A. Roberts, “FAQChat as an information retrieval system,” 2005.

[5] “1999 Third International Conference on Knowledge-Based Intelligent Information Engineering Systems. Proceedings (Cat. No.99TH8410),” in *1999 Third International Conference on Knowledge-Based Intelligent Information Engineering Systems. Proceedings (Cat. No.99TH8410)*, 8 1999, pp. 0_1–.

[6] J. Hill, W. Randolph Ford, and I. G. Farreras, “Real conversations with artificial intelligence: A comparison between human-human online conversations and human-chatbot conversations,” *Computers in Human Behavior*, vol. 49, pp. 245–250, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.chb.2015.02.026

[7] T. D. Rieke and H. Martins, “The relationship between motives for using a Chatbot and satisfaction with Chatbot characteristics in the Portuguese Millennial population: an exploratory study.” 2018.

[8] N. Sadeh, F. Gandon, and O. Byung Kwon, “Ambient Intelligence: The MyCampus Experience,” no. July, 2005.

[9] M. Dibitonto, K. Leszczynska, F. Tazzi, and C. M. Medaglia, “Chatbot in a Campus Environment: Design of LiSA, a Virtual Assistant to Help Students in Their University Life,” 2018, pp. 103–116.

[10] B. I. Systems and S. W. Count, “Evaluation , development and testing of an educational support chatbot,” vol. 378, 2018.

[11] M. Mctear, “Conversational Modelling for Chatbots : Current Approaches and Future Directions,” *Conference on Electronic Speech Signal Processing (ESSV 2018)*, 2018. [Online]. Available: http://essv2018.de/wp-content/uploads/2018/03/2_Keynote_MichaelMcTear_ESSV2018.pdf

[12] D. Dutta, “Developing an Intelligent Chat-bot Tool to assist high school students for learning general knowledge subjects,” p. 13, 2017. [Online]. Available: https://smartech.gatech.edu/handle/1853/59088

[13] F. Ho, “TA-bot: An AI agent as a Teaching Assistant,” no. May, p. 5, 2018.

[14] A. S. f. P. Affairs, “System Usability Scale (SUS),” 9 2013.

[15] “MeasuringU: Measuring Usability with the System Usability Scale (SUS).” [Online]. Available: https://measuringu.com/sus/

A

Questionnaire of Project

Tecnico Helper

*Obrigatório

Caracterização do Utilizador

1. **Sexo** *

   Marcar apenas uma oval.

   ○ Masculino
   ○ Feminino

2. **Idade** *

   Marcar apenas uma oval.

   ○ 18-21
   ○ 21-27
   ○ > 27

3. **Numero de matrículas no IST** *

   Marcar apenas uma oval.

   ○ 1
   ○ 2
   ○ 3
   ○ 4
   ○ 5
   ○ > 5

System Usability Scale

4. **Acho que gostaria de usar este sistema frequentemente** *

   Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

5. **Achei o sistema desnecessariamente complexo** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

6. **Achei o sistema facil de usar** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

7. **Acho que iria precisar da ajuda de um técnico para usar o sistema** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

8. **Achei que as varias funções do sistema estavam bem integradas** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

9. **Achei que havia muita inconsistência no sistema** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

10. **Acho que a maioria das pessoas iria aprender a usar o sistema facilmente** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

11. **Achei o sistema muito imcómodo para usar** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

12. **Senti-me confiante a usar o sistema** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

13. **Preciso de aprender muitas coisas antes de usar o sistema** *

Marcar apenas uma oval.

	1	2	3	4	5	
Discordo Fortemente	○	○	○	○	○	Concordo Fortemente

Avaliação do sistema

14. **Comparando o Técnico Helper com as alternativas existentes, qual foi o seu preferido?** *

Marcar apenas uma oval.

○ Tecnico Helper

○ Alternativas existentes

15. **Se respondeste Alternativas existentes, porque preferes esta opção?**

16. **Quais foram as feature que gostaste mais no sistema?** *

Marque todas que se aplicam.

- □ Dar as noticias do IST
- □ Dar informação como email, numero de telefone de serviços do IST
- □ Dar o link para o site de um curso, pedido pelo utilizador
- □ Lugares de estacionamento disponíveis na Alameda
- □ Localização do parque de estacionamento da Alameda
- □ Dar a hora do próximo shuttle numa determinada paragem
- □ Dar o horário de uma estação do shuttle
- □ Dar o email do ist do utilizador
- □ Lista das disciplinas do utilizador com link para o site das disciplinas
- □ Dar a média do utilizador num curso
- □ Dar a média do utilizador numa disciplina
- □ Dar os créditos feitos por um utilizador num curso
- □ Dar a qualidade de dinheiro que o utilizador ainda tem que pagar das propinas
- □ Dar o próximo pagamento do utilizador com a quantidade, entidade e a referencia
- □ Dar próxima aula do utilizador
- □ Dar próxima avaliação do utilizador
- □ Dar menu da cantina
- □ Dar anúncios de uma disciplina
- □ Possibilidade de receber notificação sempre que uma disciplina tem um novo anúncio
- □ Dar sumários de uma disciplina
- □ Email de um professor

17. **Que feature achas que falta ao sistema?**

____

____

____

____

____

18. **Para ti a característica de chatbot é essencial para o sistema ou preferias um sistema com as mesmas features mas sem chatbot?** *

Marcar apenas uma oval.

- ○ Sistema com chatbot
- ○ Sistema sem chatbot