Chatbot for Supporting Financial ServiceChannels

João Francisco Rodrigues
joaofrodrigues@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal
May 2019

Abstract

With the advances in Artificial Intelligence, machines have started to impersonate different human traits, like the artificial intelligence conversational entities, also called chatbots. Chatbots are computer programs that can carry out a conversation with the user, allowing a user to simply ask questions in the same manner that they would address another human. Chatbots are currently gaining a lot of popularity, especially in the business sector as they have the potential to automate customer service and reduce human efforts. They have grown and improved to capitalize on messenger platforms.

The technology at the core of the rise of the chatbot is natural language processing (NLP). Recent advances in machine learning have greatly improved the accuracy and effectiveness of natural language processing, making chatbots a viable option for many organizations.

In 2018, new legislation made banks provide a set of secure APIs access to their customers’ bank account information, making banking organizations be one that could benefit from chatbots. This will be presenting the benefits of including chatbots, along with the development of a chatbot for banking applications, which will be able to perform banking operations requested by the user. The user will be able to communicate with the bot, providing the user with a natural, intelligent and simple way to perform these operations, which you would usually do in a home banking application. This project was proposed by Link Consulting[1] in order to integrate it with other solutions from the company.

Keywords: Chatbot, Natural Language Processing, BankOnBox, Microsoft, SpeechRecognition

1. Introduction

1.1. Motivation

Chatbots represent a potential shift in how people interact with data and services online. The most intuitive definition for a Chatbot is, a software that can have a conversation with a human. They can perform a wide variety of functions based on user commands, often used online and in messaging apps, as intelligent virtual assistants.

Text communication nowadays takes a huge part in peoples’ lives. Almost all age groups for personal, family and social communication as well as for business purposes use text messaging, many companies are even using text messaging for communication between employees.

One of the most known and used messaging app is Facebook’s Messenger[2], and they saw great potential for chatbots, ever since they launched the Messenger Platform, which is nothing more than a toolbox for building bots[3], the company already has over 100,000 bots developed by its users, allowing companies to provide customer assistance and relationships through conversation, if users are using a communication channel they are familiar with, texting, to have a conversation with a bot, they are already used to the technology.

More and more companies are complementing their businesses with chatbots, customers always become more demanding, and with the accommodation of message applications, Facebook’s Survey shows “53% of people are more likely to shop with a business they can message”[4].

Regarding what nowadays people expect from businesses in terms of availability, “51% of people say a business needs to be available 24/7”[5], no one enjoys waiting endlessly on the phone, being put on hold, or having to push buttons to navigate from department to department, just to find the answer to a question that a bot can resolve in 30 seconds and at any time, without having to worry about working hours. Humans are restricted to the number of things we can do at the same time. A study suggests that humans can only concentrate on 34 things at the same time. For example, when we present phone numbers, we present them in groups of three and four, which helps us to remember the list[6]. If it goes beyond that you are bound to meet errors, chatbots, on the other hand, can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered instantly, in fact, re-
ports show that the vast majority of chatbot users reported productivity to be the main reason for using chatbots. These participants highlighted the ease, speed, and convenience of using chatbots. There are many anticipated benefits for Chatbots, which may include but not limited to:

- Helps you Save Money: Chatbots are a one-time investment, which helps businesses reduce staff required and possibly also bringing new customers.

Unlike human beings, chatbots work 24/7, they can be fast enough to provide this kind of predefined service with less error compared to a human. - Provides 100% satisfaction to customers: Humans react to others based on their mood and emotions, whereas chatbots are bound by some rules and obey them as long as they’re programmed to. They always treat a customer in the most polite and perfect way no matter who the person is.

- Personalized Experience: When you visit a website searching for a piece of information, you know its probably somewhere in there and you will end up having to scan around different web pages and menus to find what you are looking for. Instead with a chatbot, the interaction is resolved to the questions you make. Also, Chatbots can change to accommodate the users preferences, an easy example is the Chatbot could communicate in any language preferred by the user.

One of the business sectors that could benefit from chatbots are financial systems. On the 16th of November of 2015, the European Union passed the Revised Payment Services Directive (PSD2 EU Directive 2015/2366). In short, PSD2 obligates banks to provide third-party providers access to their customers accounts through open APIs, the API provides a secure avenue that allows bank account holders to access their banking data and services by granting access to third-party applications appropriately authenticated by the national regulators, following the consent of both the bank and customer.

This means that now new providers, not necessarily banks, can consolidate your accounts information in one place and acquire insightful data for you. With the banks APIs, a single app can have access to banking information related to checking accounts (credit, debit, etc.), with contents like balance, movements, transfers, everything except information that is deemed sensitive and can directly identify the end user. This will lead to the appearance of multiple applications to ease access to our account information and make online payments without credit cards, i.e. direct access to our bank account via the Internet.

1.2. Objective
The main goal of this thesis is the development of a chatbot that would complement the current way to access the BankOnBox Home Banking solutions, presented in the next section

Instead of browsing a website, you will have a conversation with the Chatbot. The user should be able to communicate with the chatbot to request banking operations from all the financial institutions that use the BankOnBox solutions.

This bot is expected to support and only be used in Portuguese, and to offer the customers a different and more intuitive way to interact with this technology, the user will be able not only to communicate via text but also by voice.

1.3. BankOnBox
BankOnBox is a solution for Financial Institutions, developed by Link Consulting (Link), that was designed to bridge technologies, the needs of the Banks and the needs of their customers. BankOnBox provides Financial Institution’s solutions to implement Home Banking in their system, that have access to banking operations. With the upcoming changes in EU legislation, as described in the Introduction, the banks are being more forced into opening themselves up and allowing their customers to provide access to third-party applications to their bank accounts.

That’s basically the concept of Open Banking, banks must look at opening their client’s banking information to third parties through Application Programming Interfaces (APIs). In this project’s context, the chatbot should be able to freely access the banks’ API, to access banking information and also perform banking transactions.

Link is aware of this and is already developing a solution to implement Open Banking for financial institutions that use BOB’s technologies. The BOB API is a thesis project that was developed by Joo Silva Santos, which consists in implementing a set of APIs which are PSD2, which will let every user be able to access his banking information from applications other than the provided by his financial institution.

With this API it’ll possible to develop a chatbot, that will serve as a third-party application and is able to access users’ banking information, via this API. The API will provide access to the same main features that BOB Home Banking has, which will have to match the bot features:

- Access to a user’s bank account(s) balance;
- Access to a user’s bank account to initiate Transfers;
- Access to a user’s bank account to inquire about transfers information;

2. Background
A Chatbot is a computer program, powered by a set of rules and some artificial intelligence, that you can interact with via a chat interface, chatbots are
designed to imitate human conversations and carry out actions, based on voice commands and/or text inputs [14].

Over the past decade, quite a bit of advancement has been made in the area of chatbots, major Internet companies such as Google, Facebook, and Microsoft now see chatbots as the next popular technology and are at the moment highly used to provide a variety of different services.

There are basically two kinds of chatbots, first you have the bots that limit the user interaction, like the Q&A bots, Question, and Answer bots, but these bots don’t allow the user to ask questions, instead they are the ones that make the questions to the user and, at the same time, provide the user with a list of possible answers for the user to pick from, and for each option it gives it has a corresponding answer. Everything is pre-defined, the problem with this bot is they don’t even allow the user to really say anything, they just pick options.

Then there are bots that risk trying to parse anything the user says, those who are equipped with natural language processors, these bots can give the user the feeling that they’re having a conversation with a real person, rather than poking through tedious text options. These types of chatbots are designed to understand the user intent and keep a conversation with different persons, every person talks in a different way. For the context of this project, this is the type of bot that is to be implemented in this project.

2.1. Natural Language Processing

Natural Language Processing (NLP) is an integral part of any smart chatbot, perhaps the most important component of a chatbot.

The arrival of talkative assistants like Siri, Alexa and Cortana are all well and good, but there is no point if the device or service does not understand what the person is asking for, or talking about. Over the few years of their short lives, they have become better at understanding people thanks to NLP, a technology that learns words and phrases and their meaning to provide a better service.

In a spoken dialog system, natural language processing (NLP) converts the words in an utterance into a machine-readable meaning representation. [15]

Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language, it can also examine and use patterns in data to improve a program’s understanding, it’s basically the ability of a machine to analyze and understand human speech.

To understand how NLP works, we have to take a look at one of its main components, Natural Language Understanding (NLU), the most difficult part of NLP is understanding, or providing meaning to the natural language that the computer received, that’s NLU’s job. Natural Machine Understanding deals with machine reading comprehension, to understand the meaning of a text. NLU uses algorithms to reduce human speech into a structured ontology, although different NLU systems use different techniques, that with the right training the system should be able to understand the meaning of what you said.

The main drive behind NLP is to create chat and speech-enabled bots that can interact effectively with the public without supervision.

2.2. Speech Recognition

Speech is one of the most powerful forms of communication between humans, speech recognition software is used to convert spoken language into text by using speech recognition algorithms.

In natural speech there are hardly any pauses between successive words, also that in most spoken languages, the sounds representing successive letters blend into each other in a process termed co-articulation, so the conversion of the analog signal to discrete characters can be a very difficult process. For speech recognition, there are usually three common models that work together [16], the acoustic model takes the waveform of speech and chops it up into small fragments and figures out each sound that the person is speaking, then the pronunciation model that takes those sounds and strings them together to make words and finally the language model that takes the words and strings them together to make sentences.

Speech Recognition can be used by people with disabilities, for in-car systems, in customer service, voice may be the next major banking channel. The main problem with speech-enabled applications is having an interaction that feels natural to customers, where they can speak as they would to another human and not speak as if they’re talking to a robot. Companies are opening their speech recognition solutions, providing APIs or web services so that developers can bring speech recognition to their own apps and services.

2.3. Bot Framework

Bot framework is a set of predefined and pre-installed methods and classes created for bot developers. It provides developers a set of tools that help them write the code better and faster. In simple terms bot developers and programmers use development frameworks so they don’t have to create chatbots from scratch [17], saving learning and development time, with security measures implemented for you and documentation to ease functionality implementation.
These days, there are so many platforms offering a bot framework for building chatbots. You need somewhere in between 1 to 5 hours to understand a platform depending on your technical know-how. Once you have the proper understanding, you can build a basic chatbot in minutes.

2.4. BankOnBox Access
For the bot to be able to perform any banking operations, it has to be able to communicate with the BOB API, which defined a way to authenticate and/or authorized users. Authorization is about obtaining a consented access to the user’s information via the APIs. Authentication is about obtaining identification credentials, such as username and password from a user and validating those credentials.

2.4.1 OAuth2.0
OAuth2.0 is an open-standard authorization protocol or framework that describes how unrelated servers and services can safely allow authenticated access to their assets without actually sharing the initial, related, single login credential.

2.4.2 OpenID Connect
OIDC is a protocol used to enable clients to authenticate that the user is whom he says he is when requesting initial access and permissions to be consented for in regards to a service’s account. While OAuth is only concerned with providing authorization to resources that the user consented to, OIDC also ensures that the end user is authenticated with the server and application.

3. Solution
In this section our chatbot solution will be presented, first an introduction to the chatbot, the supported chatbot dialogs that the chatbot should have to understand user commands, the language understanding capabilities that are expected from the chatbot. After that the chatbot architecture will be presented, where the choices for each of the chatbot components, studied in chapter 2, will be explained.

3.1. BOB Chatbot Dialogs
This chatbot is designed to perform the banking operations available in BOB API, account balance, movements transfers, payments, that were described in section 1.3 and to access BOB API, in section 2.4 the user authentication. The bot must have a dialog prepared for each of these actions, meaning that depending on the user message, the chatbot must respond according to the request. Dialogs define what the chatbot can expect from the user, what it has to do, depending on the user message, and what should be answered.

To be able to perform this operations, the user must first be able to authenticate itself into the system, after that, natural language processing will be used to understand user inputs, to understand user intentions and decide which of the supported dialogs will be triggered.

In this section it will be explain each of the supported dialogs, as well as showing off what natural language processing capabilities are expected for this chatbot, what types of requests will the user be able to do and also how the chatbot will handle BOB API responses.

First we have the user authentication, the first interaction with the bot will always send the user to the authentication dialog, where the user should choose to login with BOB credentials, or it will also have the option to login with Facebook Login.

For the Balance dialog the user has two different options, the regular request, where the user just asks for the balance, but the user will also be able to ask for a specific bank account, asking by the account’s name.

In the movements dialog the user has the regular request, with this the movements from the account defined as a main account will be shown. The user can also make a request for the desired account, same as before, and a specific transfer date, making the bot only show movements results that match the user request.

Finally, the payment dialog the user will be able to make a regular request for a payment, where a card will be presented with the fields required to make a payment. This card fields are filled by the user, or can be filled by the chatbot if the user provides them along the request. The request can include an IBAN number, of the recipient, and the amount to be sent. If the user provides the IBAN and/or the amount, the chatbot will fill the card fields automatically for the user. User’s will also be able to save a contact list, linking a name to an IBAN number, the user will be able to make the payments request by using the contact’s name in the request, instead of writing the IBAN number every time.

3.2. Solution Architecture
In this section, we present the solution architecture, Figure 1 to the objective previously mentioned in section 1.2. This section will include the solution for each objective that was presented and explain each of the decisions that were made in order to reach the final architecture.

The first phase was to decide how was the chatbot going to be built, afterward, how the Chatbot will be able to process Natural Language Process-
ing (NLP), which should work for Portuguese, then the communication to the BOB API, which will communicate with BankOnBox services to perform banking operations. And finally, enable the user to speak with the bot, that has to be able to understand Portuguese, along with the necessary features that were implemented to improve the final solution.

![Figure 1: BOB Chatbot Architecture](image)

### 3.3. Microsoft Bot Framework

For this project, it was chosen to develop the chatbot using the Microsoft Bot Framework. Microsoft introduced its own bot framework in early 2016. Microsoft bot framework SDK, similarly to other frameworks, provides the resources a developer needs to build an intelligent conversational chatbot that can provide a natural interaction. The main components of the framework are presented next.

#### 3.3.1 Bot Connector

Bot Connector allows a bot to exchange messages with any of the channels that are available in Microsoft Bot Framework, by using REST API and JSON over secure protocol HTTPS. This also enables the option to communicate with any clients personal chat application by using the Direct Line API. The Direct Line Channel allows the developer to connect to their bot from anywhere.

#### 3.3.2 Bot Builder

The Bot Builder is an SDK for .NET Framework developers for developing bots using Visual Studio and Windows. The SDK supports C# and Node.js. The kit consists of Bot Application, Bot Controller, and Bot Dialog templates. Bot Application templates already contain a simple project with all of the components for a bot already integrated with cognitive services. These already include methods to accept messages and a dialog builder to generate an appropriate response.

### 3.4. NLP - Microsoft Language Understanding (LUIS)

LUIS stands for Language Understanding Intelligent Service, an entirely cloud-based machine learning-based service to build natural language into apps, in this case, a chatbot. LUIS is built on prior work in Microsoft Research on interactive learning [18].

LUIS brings in artificial intelligence (AI) to applications so that computers and humans can speak with each other seamlessly, it allows for developers to create smart applications, without having to think or worry about designing the various Natural Language Processing (NLP) and Machine Learning (ML) techniques, developers can build custom natural language understanding (NLU) models interactively.

LUIS can be defined in three key concepts, utterances, which are text input from the user that your app needs to understand, intents, that represents actions the user wants to perform, and entities, representing detailed information that is relevant in the utterance, extraneous information about the intent. Entities can be extracted with machine-learning, which allows LUIS to continue learning about how the entity appears in the utterance, they can be extracted without machine-learning, matching either exact text or a regular expression, or even in patterns can be extracted with a mixed implementation, in summary, LUIS offers different entities for different needs.

For LUIS to be able to interpret users intent and context it must be trained, the developer has to create the type of instances the system should be expecting and can do something with, and then it should provide it with a set of examples strings that exemplify that intent. The more examples the systems have for each query the more accurate the system gets.

### 3.5. Speech Recognition - Bing Speech API

The chatbot should also be able to recognize user speech, and the speech recognition had to be able to understand the Portuguese-Portugal language and Microsoft’s Bing Speech API is one of them.

Microsoft’s Cognitive Services Bing Speech API, which converts spoken audio to text. The API can be directed to turn on and recognize audio coming from the microphone in real-time, recognize audio coming from a different real-time audio source, or even recognize audio from within a file.

This project will use the JavaScript client libraries since they support the most intuitive way
for speech recognition, which is real-time, continuous stream audio recognition, this enables users to transcribe audio into text in real time, and supports to receive the intermediate results of the words that have been recognized so far, instead of having to record the audio sample, send it to the API, and wait for the API to return the recognized text.

3.6. Bing Spell Check API
One of the natural language understanding challenges is that the user can make spelling mistakes while writing, to ease LUIS’ correct prediction, the Bing Spell Check API was used, to automatically correct user writing errors. The Bing Spell Check API is part of Microsoft’s Cognitive Services, which allows you to perform contextually grammatical and spell checking in an utterance. Instead of relying on dictionary-based rulesets, which means comparing each word with a known list of correctly spelled words

3.7. Facebook Login Graph API
To provide the user with a different possibility to authenticate the user, and get access to BOB’s operations, other than the BOB’s authentication method, the application will allow the user to authenticate itself with it’s Facebook credentials, taking advantage of Facebook’s Login and Graph API.

Facebook Login is a fast and convenient way for people to create an account, or in this projects case, link an existing account with the users Facebook account, and from then on, to log into the application.

Facebook Graph API [19] is the primary way to get data into and out of the Facebook platform. It’s an HTTP-based API can be used to make you application query data and utilize user’s data.

3.8. Azure Cosmos DB
For the bots know whom it is talking to and be able to perform operations for the corresponding user, the bot will have a storage system, Cosmos DB, to register users, and link them to the multiple accounts they might have.

Azuers Cosmoos Database, Cosmos DB [45], is Microsofts proprietary globally-distributed, multimodel database service. A database is an organized collection of information that can be easily ac-cessed, managed and updated, using database services like Cosmos Db, a system is able to interact with end-users, other applications, and the database itself to capture, store and analyze data

4. Implementation
This Chapter will focus on explaining, regarding the Solution that was defined in the previous Chapter [9], the architecture implementation, how was the bot implemented and integrated with the other components, the decisions made during the implementa-

4.1. LUIS, Natural Language Processing
LUIS is our key component of the chatbot, it offered a fast and effective way of adding language understanding to the application, in order to understand and control the user requests. For the purpose of this project I had to make sure that the bot had at least one intent for each of the BOB API features, the intents initially needed were for requesting account Balance, ”Saldo” intent, for initiating Transfers, ”Transferencias” intent, and for requesting transfers informations, ”Consultas” Intent.

To make Luis able to correctly identify these intents, and in order to train it, LUIS needs to be fed with utterances examples that should represent a possible user request for the desired intent, Microsoft provides a graphical web interface for creating and training the application.

For each utterance LUIS associates a corresponding score, this score represents LUIS degree of certainty, that the utterance is associated with that given intent. This score is greatly increased the more utterances you have, ideally, you want your intents to be distinguishable from one another with a high degree of certainty. Also, to drastically improve the quality of LUIS predictions, you can specify entities. Entities help in identifying the parameters which are required to take a specific action. While intents are required, entities are optional.

LUIS success is based on training, the more you utterances you provide it with, the better results you will have, the more options you are covering, the better it will react, and it should be always learning more, the more you and the user use it. The techniques used to create, detect and train LUIS to be able to correctly identify the needed intents can be read in the full thesis report.

4.2. OpenID
As previously mentioned, in section 2.4.2, the main Authentication and Authorization mechanisms for the BOB API, is done through OpenID Connect, from section 2.4.2.

Since the project is being developed in C#, using .NET Framework, I decided to take advantage of the IdentityModel.OidcClient2 library [20]. IdentityModel.OidcClient2 is a C#/NetStandard OpenID Connect Client Library for native Applications, an implementation of the OIDC/OAuth2 for native apps specification for C#. This library allows the developer to create an OpenID client, that given the correct options, this client is responsible for generating the necessary security measure, that along with the users chat conversational reference, are saved in a database document, section 4.4.3 and for preparing the endpoint URL, for the
user to be able to connect with BOB’s Login.

To deal with the server login response, an HTTP handler was defined, that is responsible for handling the authentication result, the bot will retrieve the security measures that were saved in the database and decide whether the request is valid or not, if success the bot will make another request to the server with the data received from the previous response, that will send a result response which gives us the user Claims, the users identity information, from which we retrieve and use the UserId, UserName, for user identification, but also the access Token and refresh token, that are necessary to make calls to the BOB API. The user’s information are stored on the respective databases, in 4.4.2 and 4.4.1.

Figure 2 represents the Authorization Code Flow of that was implemented.

4.3. Facebook Login
In order to enable the users to log in not only through BOB, it was requested that the user should be able to also use another type of account, in this case, but it was also decided the chatbot allow the user to login through their Facebook account. A browser-based log in flow using entirely server-side code, using browser redirects, so I manually built a login flow [21], which is represented in Figure 3. The implementation starts by, creating the redirect URL to our Facebook application Login endpoint, which will open the typical Facebook Login screen, if the user successfully logs in, Facebook will send the response. To handle this response an ASP.NET Web API route was created, this response comes with a parameter, "code", this "code" is an encrypted string unique to each log in request that has to be exchanged for the access token, by making an HTTP GET request to Facebook’s Graph API, with the necessary parameters. Having retrieved this token the application can now make a request for the user’s Facebook information, which it only needs the user’s facebook id, in order to store it in the database, section 4.4.2, in 4.4.1.4.

Figure 3: OpenID Connect Connection Flow

4.4. Cosmos DB
One of the early decisions was to use a Database, at first all the bot needed was to store user data, but the project needs were discovered and changed during its development, and in the end, there were three different database collections that were created.

4.4.1 Bot Collection
This collection is automatically created by the bot, there’s a new document for each conversation started, then when a new user enters the conversation, the user document is created, that keeps the data for the user during the conversation lifespan and along with that the private conversation document, that saves data for the user within the current conversation, this project only uses the user data document, where it saved the data received from OpenId log in, from section 4.4.2, so that the bot can have quick access to any of the user informations within the lifespan of the conversation context, and used them accordingly.

4.4.2 Account Collection
In order to store the user’s BOB account with the corresponding user data, and being able to retrieve this data the next time the user logged in, a new database collection was created the Account Collection. This collection is responsible for storing the user’s account information, received from OpenId log in, from section 4.4.2, and the document identifier is defined with the user’s BOB identifier, defining
a document for each user account, that can be retrieved by the user’s BOB identifier. This is also the document where the "facebookId" is stored and links it to the user’s BOB account, section 4.3.

4.4.3 State Collection

As previously explained, in OpenId section 4.2 there was a need to create a state Document Collection, to store the request security parameters and users conversation reference. This document is only supposed to be used once, so these are the only documents that are dispensable and can be deleted after the first use.

4.5. User Interface

To make the bot accessible to users, the bot should be hosted on a web hosting service, in this case, the bot was already hosted in Azure [22], and it was chosen to host a web page, as our bot user interface, so that anyone could get easy access to it. This web page needed a chat window, the message delivery platform, to integrate this, the web page implements the Bot-Framework Web Chat [23], as our Web Chat platform, that provides the ability for users to interact with your bot directly in a web page. In the end, our user interface, the chatbot, is hosted in https://lasttestlinkbot.azurewebsites.net/ and Figure 4 shows an example of what this interface looks like.

4.6. Bing Speech API, Speech Recognition

To enable speech in the Web Chat [24], that was implemented in the User Interface, 4.5, the JavaScript code that invokes the Web Chat control, was customized to include speech, this is done by enabling it via "speechOptions", which tell the bot which speech recognition and/or synthesizer to use, in this case, as referred in section 2.2, the Bing Speech API, that is defined as the speech recognition, and you have to provide it with the Speech API access key, and finally the "locale", which defines the language to be recognized, in this case, "pt-PT".

4.7. BOB Bot

In the final section of the implementation, it will be explained how the bot was created, and implemented all the previous components, to achieve the desired architecture from section 3.

To create a bot follow the steps provided by bot service v3 documentation [25], the steps are going to the Azure portal [26], create a bot, providing the requested information, and this will automatically create a bot for you, that is already ready to use. While creating a bot you are asked if you want to use a template to create your bot, choosing the Language understanding template the bot will automatically be communicating with a LUIS application.

Every time that the bot receives a POST message, the bot retrieves the activity from the POST body, and checks for the activity type. For this bot we are only interested in the 'message' type, which represents that there’s been a communication between the user and the bot, but we can still separate it between two 'message' types, if there is a parameter value in the message, it means that the message is the response data from a card attachment or just a normal Text message. In the card case, I decided to use Adaptive Cards [27]. The Adaptive Card format is a simple yet powerful declarative layout format that provides a lot of flexibility, allowing for visually rich cards and adds a more rich interaction for the user, even to these cards are simple JSON objects. And then there’s the normal message type, the text message, which the bot will send to LUIS, to predict the message intent, this will forward this message to your LUIS application endpoint, an HTTP endpoint utilizing a REST API. The communication is done using JSON packages, LUIS will perform the predictions it was trained to, section 4.1, and return the prediction answer. LUIS response returns the text that the user sent, along with its prediction, where the 'topScoringIntent' will define the highest scored intent prediction, a function will be run depending on the defined 'topScoringIntent', each function represents one of the created intents in LUIS.

Figure 4: Chabot Interface Usage Example
These intents also represent the BankOnBox operations, from section 4.3, to implement these operations, one must make the necessary calls to the BOB API.

The API Calls are made via a REST API, with JSON, where each request has to be made to a specific API Endpoint URL with the according path that corresponds to the operation one needs to perform, as well as some requested parameters, depending on the request, for example the balance and movements request. For the balance request, the user has to provide the unique identifier of the account number for the operation, that is sent along with the request URL, and, on the other hand, payment request is an HTTP POST request, where you need to send the payment information in a specific JSON format, along with the HTTP request, this JSON format is determined in the API. In return, all these requests return a JSON file, with the requested information.

In order to be able to perform these actions, the user needs to prove that it is authorized to perform such actions, which is done through the use of the BankOnBox OpenId Access Token, explained in section 4.2, this token should be sent in the HTTP request authorization header, without it the user is not allowed to perform any actions in the BOB API.

The full explanation of how the user authenticates into the system and the dialogs created for each operation, balance, movements, and payments, are explained in the full thesis report.

5. Conclusions

This project presents a solution to developing a chatbot for financial institutions, that take advantage of BankOnBox solutions, the chatbot is able to perform the banking actions on behalf of the user.

It was also studied what chatbots are, as well as the needed components to develop one, which lead to, developing the chatbot by taking advantage of one of the available Chabot platforms, to aid the chatbot creation process as well and integrating all the necessary NLP and speech recognition capabilities. Different solutions were studied and considered, the state of the art, and finally, Microsoft Bot Framework, is one of the current top platforms in the market, was the platform elected.

Taking advantage of the easy integration of cognitive services, through the use of this platform, adding natural language processing (NLP), using LUIS linguistic capabilities, as well as integrating speech recognition to the chatbot, using Bing Speech API. Both these tools were also chosen for their compatibility with the required language, Portuguese.

The NLP tool used, LUIS, allowed the chatbot to understand user commands, making use of intents and entities, in order to understand users requests and to provide the bot with the context banking operation. The chatbot communicates with BankOnBox’s API, BOB API, in order to perform these banking operations on behalf of the user. The applications also implemented Bing Speech API services, which enable the bot to recognize speech, providing the user with a different way to communicate with the Chabot.

The objective of developing a chatbot application was successfully achieved, the bot was made accessible to the public, a web page was created solely to interact with the chatbot, but the chatbot could also be embedded in a banking institution web page. It can understand users’ requests for banking operations, like proving the user’s accounts balance, movements transactions and payments, and could be trained to perform much more in the future. This version of the bot produced the basic functionality, but it can be extended based on users’ needs and feedbacks.

Acknowledgements

I would like to thank Professor Jos Alves Marques for giving me the chance to work on this project, for the opportunity and patience. My advisor, Rui Rasteiro for all the technical help and always being available to help me. Toms Teixeira for their help and inputs provided. I also have a special thanks for my parents for loving me and giving me everything I have, and to my close friends for all their strength and support, their company and for their companionship in the last couple years, I will always be there for all of you. And finally my girlfriend, for picking me up when I am down, and for pushing me forward went I didn’t have the strength, I thank you and I love you.

References


