MSc program

"Human-Computer Interaction"

MSc Thesis

of Evangelos Poulios "Electrical Engineering"
R.N.: 1057071

Title:

Design, development and evaluation of a chatbot that supports the interaction of a platform user.

Supervisor:

Professor Kyriakos Sgarbas

Thesis Number: 1057071

Patras, 22/06/2021

University of Patras, Electrical and Computer Engineering Department [Evangelos D. Poulios]

Thesis structure

Acknowledgements

First and foremost, I would like to thank my thesis supervisor K. Sgarbas. for his help throughout the preparation of the current thesis.

Also, I would like to thank the tutors, as total, of the Human Computer Interaction Master for the knowledge that they imparted during the lectures and assignments.

Finally, a big thank you goes to my friends and my family who offered me great support during my studies.

Abstract

The current Thesis subject is: development of interactive dialog bot (chatbot) to support interaction between user and platform. To be more specific we chose to investigate the field of customer service, for the case of a hotel business. Customer service is an area where Artificial Intelligence (AI) can be utilized.

That area is being examined regarding the adoption of new innovative technologies and the final delivered service quality. Today, the customer service industry tends to make good use of new AI technologies to offer an improved service benchmark but also to develop new approaches in the same direction. AI seems to be able to offer remarkable efforts and gradually replace the physical presence of man.

The title of present Thesis was chosen as aim to approach a realistic field with the utilization of modern technologies and in particular AI. The title is specialized in the case of a hotel business, which is a characteristic example where software companies have already been employed. The final goal of the current work is to gain experience for handling technologies in fields where traditionally human resources have been employed and also to examine if these approaches can be the future of customer service.

Among the ecosystem of the present dialog software frameworks, we decided to choose the option of Microsoft Bot Framework. Using that framework, we designed an application that can answer questions based on a knowledge base but also can collect details through a dialogue, between the hotel business customer and the dialog software, in order to inform the customer about the rooms' availability.

The knowledge base may contain questions - answers from the official list of frequently asked questions (FAQ) that the hotel has posted on its website, from other information that the hotel communicates with various ways on its official website, and finally from an already offered knowledge base from the framework, by selecting the friendly chitchat mode. The other part of the application, which informs about rooms' availability, uses a language recognition service (LUIS) through a programming interface (API) to match the information entered by the customer with the pre-registered entities. Once the necessary information has been collected from the customer and verified through validation checks, it is forwarded to execute a query in a database designed to support the room reservation system.

The final dialog application can replace the hotel business's receptionist. Essentially, like a receptionist, it can answer questions about the hotel, as usually described in its official website, but also to execute queries in the room reservation system in order to inform the customer about rooms' availability. The dialog application can be offered both on a hotel website and on social media channels, supported by the hotel. Below, we will describe the Thesis structure and we will end up with conclusions.

In the First Chapter we introduce the categories of dialog software that exist today. We also mention basic terms that are necessary to understand the subject of conversational software.

We continue with Chapter Two where we list four widely accepted frameworks used in the field of dialog software development. Of these, we provide more details about the Microsoft Bot Framework, which is our choice to use in current Thesis.

In Chapter Three we analyze the utility of dialog software for the customer service industry. A typical case of this industry could be considered a hotel business. That was selected as a case for us to develop the dialog software application. In this chapter, we also define the goals but also the limits we have set for the desired functionality that our application will offer.

Chapter Four is a brief guide to the steps we followed in order to create services that works with AI and are essential part of the basic architecture that we developed for the conversational software.

Completing the Thesis in the Fifth Chapter we include the results for the design and development of the conversational software, while we also propose a heuristic evaluation methodology that can be applied to our application. We close the present work with Chapter Six where we cite a brief review.

Summing up, aiming at this Thesis to develop a dialog software to serve/ support the interaction of a platform user, we conclude that this approach is quite promising. Existing frameworks, as the one that we have selected, enable the development of valuable software for services that until recently were available only by humans. During the writing of the Thesis, we became acquainted with the architecture that the development of a dialog software follows.

The benefit of having an automated application for many business categories is undeniable. Such an approach, as AI is improving, will be able to offer increasingly better perspectives on conversational communication. Of course, as ourselves understand, a lot of effort is required to add additional features, in addition to the already classic features that are offered by dialog software.

In any case, the general goal is to expand the functionality offered by the dialog application in order to cover a wider range of services. Specifically, in our case the examination of human speech as additional means of communication is certainly a subject of interest. That prospect is already being considered in the dialog software industry. This opportunity would certainly be a worthwhile approach to continue current work. And of course, this could be extended to support more than one spoken language.

BLANK PAGE

Extended Abstract (in Greek)

Το θέμα της παρούσας Διπλωματικής εργασίας είναι: ανάπτυξη διαλογικού λογισμικού (chatbot) για την υποστήριξη της αλληλεπίδρασης χρήστη με μια πλατφόρμα. Ειδικά, επιλέξαμε να διερευνήσουμε τον τομέα της υποστήριξης πελατών, για την περίπτωση ξενοδοχειακής επιχείρησης. Η υποστήριξη υπηρεσιών εξυπηρέτησης πελατών είναι σίγουρα ένας τομέας όπου μπορεί να αξιοποιηθεί η Τεχνητή Νοημοσύνη.

Ως τομέας διερευνάται ως προς τον τρόπο εισαγωγής νέων τεχνολογιών και ως προς την παρεχόμενη ποιότητα υπηρεσιών. Στην βιομηχανία παροχής υπηρεσιών η τάση των εξελίξεων είναι η επαύξηση της παρεχόμενης ποιότητας υποστήριξης πελατών με την εισαγωγή τεχνολογιών Τεχνητής Νοημοσύνης, αλλά και η επέκταση νέων προσεγγίσεων. Η Τεχνητή Νοημοσύνη είναι ικανή να προσφέρει αξιόλογες προσπάθειες και σταδιακά να αντικαταστήσει την φυσική παρουσία του ανθρώπου.

Το θέμα της παρούσας Διπλωματικής επιλέχθηκε υπό το πρίσμα της προσέγγισης ενός ρεαλιστικού πεδίου με την αξιοποίηση σύγχρονων τεχνολογιών και ειδικότερα της Τεχνητής Νοημοσύνης. Εξειδικεύεται για την περίπτωση ξενοδοχειακής επιχείρησης, η οποία αποτελεί περίπτωση όπου έχει απασχολήσει ήδη εταιρείες σχεδιασμού διαλογικών λογισμικών. Το τελικό επιδιωκόμενο αποτέλεσμα είναι η απόκτηση εμπειρίας στην διαχείριση τεχνολογιών για αντικείμενα που παραδοσιακά απασχολούσαν ανθρώπινο προσωπικό αλλά και η διερεύνηση της αποτελεσματικότητας των προσεγγίσεων που φαντάζουν το μέλλον του τομέα της εξυπηρέτησης πελατών.

Για την ανάπτυξη του διαλογικού λογισμικού από το οικοσύστημα των υπαρχόντων βιβλιοθηκών (frameworks) επιλέχθηκε το Microsoft Bot Framework. Με την χρήση αυτής της βιβλιοθήκης σχεδιάσαμε μια τελική εφαρμογή που μπορεί να απαντάει ερωτήσεις βάσει μια βάσης γνώσης (knowledge base) αλλά και να συλλέγει λεπτομέρειες μέσω ενός διαλόγου, μεταξύ του πελάτη της ξενοδοχειακής επιχείρησης και του διαλογικού λογισμικού, ώστε να ενημερώνει τον πελάτη για τη διαθεσιμότητα σε δωμάτια.

Η βάση γνώσης δύναται να περιέχει ερωτήσεις και απαντήσεις από την επίσημη λίστα συχνών ερωτήσεων (FAQ) που διαθέτει το ξενοδοχείο αναρτημένη στην ιστοσελίδα του, από έτερες πληροφορίες που επικοινωνεί το ξενοδοχείο με διάφορα πεδία στην επίσημη ιστοσελίδα του, και τέλος από μια ήδη προσφερόμενη βάση γνώσης από την βιβλιοθήκη, με την επιλογή φιλικού χαρακτήρα (friendly chitchat). Το άλλο τμήμα της εφαρμογής, που ενημερώνει

για τη διαθεσιμότητα σε δωμάτια, χρησιμοποιεί μια υπηρεσία αναγνώρισης γλώσσας (LUIS) μέσω μια προγραμματιστικής διεπαφής (API) ώστε να αντιστοιχεί τις εισαγόμενες πληροφορίες από τον πελάτη με τις ήδη καταχωρημένες οντότητες. Αφότου συλλεχθούν από τον πελάτη οι απαραίτητες πληροφορίες και ελεγχθούν μέσω ελέγχων εγκυρότητας, αυτές προωθούνται για την εκτέλεση ερωτήματος σε Βάση Δεδομένων που έχει σχεδιαστεί για να υποστηρίζει το σύστημα παρακολούθησης κρατήσεων δωματίων.

Το τελικό διαλογικό λογισμικό μπορεί να αντικαταστήσει έναν υπάλληλο υποδοχής στην ξενοδοχειακή επιχείρηση. Ουσιαστικά, το λογισμικό μπορεί όπως ένας υπάλληλος, να απαντάει σε ερωτήσεις σχετικά με το ξενοδοχείο, όπως συνήθως περιγράφονται και στην επίσημη ιστοσελίδα του αλλά και να ανατρέχει σε Βάση Δεδομένων για τις κρατήσεις δωματίων ώστε να ενημερώνει τον πελάτη για την υπάρχουσα διαθεσιμότητα σε δωμάτια. Το διαλογικό λογισμικό μπορεί να προσφέρεται τόσο στην ιστοσελίδα του ξενοδοχείου όσο και σε κανάλια κοινωνικής δικτύωσης που υποστηρίζει το ξενοδοχείο. Παρακάτω, θα αναφερθούμε στη διάρθρωση των κεφαλαίων της παρούσας Διπλωματικής εργασίας για να καταλήξουμε στα συμπεράσματα που προέκυψαν.

Στο Πρώτο Κεφάλαιο κάνουμε εισαγωγή στις κατηγορίες των διαλογικών λογισμικών που υπάρχουν σήμερα. Ακόμη, αναφέρουμε βασικούς όρους που είναι απαραίτητοι για να κατανοήσουμε το θέμα του διαλογικού λογισμικού.

Συνεχίζουμε με το Δεύτερο Κεφάλαιο όπου παραθέτουμε τέσσερις ευρέως διαδεδομένες βιβλιοθήκες που χρησιμοποιούνται στο τομέα της ανάπτυξης διαλογικού λογισμικού. Εξ αυτών, παραθέτουμε περισσότερες λεπτομέρειες για το Microsoft Bot Framework, το οποίο είναι και η επιλογή μας να χρησιμοποιήσουμε στην συγκεκριμένη Διπλωματική εργασία.

Στο Τρίτο Κεφάλαιο αναπτύσσουμε τη χρησιμότητα του διαλογικού λογισμικού για τον κλάδο της εξυπηρέτησης/ υποστήριξης πελατών. Χαρακτηριστική περίπτωση αυτού του κλάδου θα μπορούσε να θεωρηθεί μια ξενοδοχειακή επιχείρηση. Αυτή η περίπτωση επιλέχθηκε από εμάς για να αναπτύξουμε την εφαρμογή διαλογικού λογισμικού. Στο κεφάλαιο αυτό, ακόμη, προσδιορίζουμε τους στόχους που έχουμε για την επιδιωκόμενη λειτουργικότητα που θα προσφέρει η εφαρμογή μας, αλλά και τα όρια της λειτουργικότητά της.

Το Τέταρτο Κεφάλαιο είναι ένας σύντομος οδηγός των βημάτων που ακολουθήσαμε για την δημιουργία υπηρεσιών, βασιζόμενες σε Τεχνητή Νοημοσύνη, που είναι απαραίτητες για την βασική δομή του διαλογικού λογισμικού που αναπτύξαμε.

Ολοκληρώνοντας την Διπλωματική εργασία στο Πέμπτο Κεφάλαιο περιλαμβάνουμε τα αποτελέσματα από τον σχεδιασμό και την ανάπτυξη του διαλογικού λογισμικού, ενώ ακόμη προτείνουμε μια μεθοδολογία ευρετικής αξιολόγησης της εφαρμογής μας. Κλείνουμε το κείμενο της παρούσας εργασίας με το Έκτο Κεφάλαιο όπου καταγράφουμε ένα σύντομο απολογισμό.

Συνοψίζοντας, έχοντας ως στόχο στη παρούσα Διπλωματική εργασία την ανάπτυξη ενός διαλογικού λογισμικού για την εξυπηρέτηση/ υποστήριξη της αλληλεπίδρασης ενός χρήστη μιας πλατφόρμας καταλήγουμε ότι αυτή η προσέγγιση είναι αρκετά ελπιδοφόρα. Οι υπάρχουσες βιβλιοθήκες, όπως και αυτή τη επιλογής μας, δίνουν τη δυνατότητα για ανάπτυξη λογισμικού αξιόλογου για δυνατότητες που μέχρι πρόσφατα διέθετε μόνο ο άνθρωπος. Στη διάρκεια συγγραφής της Διπλωματικής εργασίας αποκτήσαμε εξοικείωση με την αρχιτεκτονική που ακολουθεί η ανάπτυξη ενός διαλογικού λογισμικού.

Το όφελος ύπαρξης μια αυτοματοποιημένης εφαρμογής για πολλές κατηγορίες επιχειρήσεων είναι αδιαμφισβήτητο. Μια τέτοια προσέγγιση, καθώς η Τεχνητή Νοημοσύνη εξελίσσεται, θα μπορεί να προσφέρει αυξανόμενα καλύτερες προοπτικές διαλογικής επικοινωνίας. Βέβαια, όπως κατανοήσαμε και εμείς από την προσπάθεια μας, απαιτείται αρκετή προσπάθεια για την προσθήκη επιπλέον χαρακτηριστικών, πέραν των ήδη κλασσικών προσφερόμενων δυνατοτήτων από το διαλογικό λογισμικό.

Σε κάθε περίπτωση στόχος είναι η διεύρυνση της προσφερόμενης λειτουργικότητας από την διαλογική εφαρμογή ώστε να καλύπτεται ευρύτερο φάσμα αναγκών. Συγκεκριμένα, στην περίπτωση μας η εξέταση επέκτασης σε χρήση ανθρώπινης ομιλίας ως επιπρόσθετο μέσο επικοινωνίας σίγουρα συγκεντρώνει ενδιαφέρον. Είναι μια προοπτική που ήδη εξετάζεται και στη βιομηχανία διαλογικού λογισμικού. Σίγουρα αυτή η δυνατότητα θα αποτελούσε αξιόλογη προσπάθεια συνέχισης της παρούσας εργασίας. Και φυσικά αυτή η δυνατότητα θα μπορούσε να επεκταθεί για την υποστήριξη περισσότερων της μιας γλώσσας ομιλίας.

Acronyms

AI:	Artificial Intelligence
ANN:	Artificial Neural Networks
API:	Application Programming Interface
CI:	Conversational Interface
CLI:	Command Line Interface
FAQ:	Frequently Asked Questions
HCI:	Human Computer Interaction
JSON:	JavaScript Object Notation
LUIS:	Language Understanding Intelligence Service
NLU:	Natural Language Understanding
NLP:	Natural Language Processing
QnA:	Questions and Answers
SDK:	Software Development Kit
UI:	User Interface
UX:	User Experience

Contents

List of Figures

List of Tables

1 Introduction

1.1 Chatbot Definition

Nowadays, the majority of people have communicated at least once with a chatbot, in varying degrees. That software has an increasing role in our life. But how a chatbot is really defined? According to Lexico Dictionaries [1], a chatbot is

"A computer program designed to simulate conversation with human users, especially over the Internet."

A chatbot belongs to the wider category of Conversational Interfaces (CI). The CI are software interfaces that enhance human to communicate with a computer program, giving the sense of human agent. These are capable of handling text or aural human content, provided by a user, and performing meaningful actions. When a person is busy performing an activity, not having time to spend it with talks, or when he may need customer assistance these CI can be proven to be really helpful.

We can define two main types of CI: Chatbots and Voicebots. Chatbots are mainly based on Natural Language Understanding (NLU), in order to simulate a dialog between human and machine, and are often found in mobile applications and websites.

On the contrary, voicebots operate with the use of Speech Recognition Technology in order to interpret the human verbal information, and advantages by providing the feeling of communicating as a human [2].

Both, the two aforementioned categories make use of Rule Based Systems or AI. Rule Based Systems represent the pre-existed knowledge with rules. A definite number of rules is used to describe the system function logic with the use of assertions. That category of systems is the simplest form of AI as it is based on rules that are handling instances according to a set of insertions. Bots (both chatbots and voicebots) driven from AI, on the other hand, use AI

methods like Artificial Neural Networks (ANN). The ANNs convert each word into a vector, that procedure is called Natural Language Processing (NLP). In the case of Rule Based Systems we have specific recorded rules, while the ANN are structured, after training models, to configure appropriate parameters. The bot's response is given, as response to human input, after computation of matrix multiplications and nonlinear functions [3].

Another similar distinction for the types of bots that exists is between scripted and intelligent bots. As scripted bots are considered these that follow predetermined paths as they communicate with dialogs. The user in order to continue the dialog and counter the next step has to choose among specific options. Depending on the bot's scope these options can be any type like text, voice, touch input and others.

The second type of bots is intelligent bots. That type allows more flexible input from the users, because it is equipped with AI. Both that type can accept input in different types like text and voice. However, the main characteristic of these bots is that they can be improved through the conversations, that capability is responsible for the name of intelligent. Of course, that category has limitations. They can be used to specific knowledge domains and may encounter difficulties with memory management and ability to continue the conversation from where it left off. On the continue, instead of talking about bots (referring both to chatbots and voicebots) we will focus mainly in the case of chatbots, which is the subject of the current thesis.

1.2 Basics Terms for Chatbot Building

Here, we collect the basic terms of the chatbot building terminology. They will help us to communicate briefly and efficiently. For that reason, to the rest of the thesis we will refer to them when these terms can be applied.

Context

That term is used to include the information that is maintained from the conversation. Context is necessary for the chatbot in order to continue the conversation flowing. It is usually a JavaScript Object Notation (JSON) that is sent and received between chatbot application and the conversation service. In each turn between the conversation, the member's context is created and is followed by a serial identifier [4].

Intent

That keyword is used to include the intention of which the user may use the chatbot. An intent is the reason for which the user decided to take action and activate the chatbot. For example, in the case that a chatbot is used to support the customer service of clothe store, possible intents are: “I want to buy a jacket.”, “Are there any other stores in the city?”. In that case the chatbot should be able to support both these desires. It would be able to distinguish in which intent the user input message is referred each time, to intent productDesciption or locationDetails.

Entity

The important descriptions/ metadata that chatbots try to focus in user input are called entities. Entities can specify the subject of the conversation and give insights about the user's intents. For the above user’s intent of “I want to buy a jacket.” if the chatbot succeeded in understanding the user’s message and provided him customized products and not general like scarfs, that is considered a successful conversation. So, the two entities would be Product and Location, correspondingly for each intent example.

Utterance

Utterances for each intent are examples of how the user can describe his intent. For the two intents we can provide some utterances like:

- productDescription Intent

  I want to buy a red sweater.

  Show me trousers.

- locationDetails

  Where are the branches located?

  Show branches in the city [5].

1.3 NLP Approaches on Chatbots

We have already referred to AI chatbots as bots that instead of using Pattern Matching use Machine Learning approaches to extract the content from the user’s dialog and they are also capable of getting smarter after having a complete conversation. They get an initial training process and after that are

able to respond to every possible user input. For the implementation of these chatbots, often the solution is to use ANN. Actually, a Retrieval based model uses ANN to estimate a score for each one of the available responses and after that to select the most propriate from the set of responses. On the other hand, a more Generative model would be using deep learning techniques to form a new one response. Below, we are going to mention fundamental approaches in the field of chatbot building.

NLP - NLU

In AI, NLP is the domain that cares for the way computers can understand and interact with a natural language text message or voice input. NLP uses techniques derived from machine learning. To be more specific, a Natural Language Understanding (NLU) module is added in order to clarify from the raw user input a suitable response based on the user's intent.

The NLU is responsible to classify the intent and extract the entity from the user's input. The entities can be specified either from the user or the system while the intent classification can be performed from an algorithmic classifier or with a pre-trained model that was designed by humans with manual classification of the input messages. Similar process can be followed for entity extraction model. An already marked corpus can be used for training and every word can be matched with a label, in a manual process. Having already trained the models, the stage for classifying the user's input into correct intents and entities can be made automatically.

Artificial Neural Networks

ANN are used in both retrieval and generative chatbots. The user input is imported to the system, its vector representation is estimated, then the ANN is fed with it as a feature and the outcome is the chatbot's response.

Retrieval Models - Generative Models

Retrieval-based models do not create a new response but they decide to select one from the predefined set. The decision is made with a Rule-based system or with an ANN that works for the machine learning classifier. On the contrary, the Generative-based models do not depend upon a predefined set of responses but compose new responses. In those models, ANNs support supervised and unsupervised learning methods to optimize the quality of the response. Lastly, Generative-based bots are more appropriate for open-domain

conversations while, Retrieval-based bots with Rule-based models are ideal for closed domain conversations [6].

BLANK PAGE

2 The Microsoft Bot Framework Choice

2.1 The Ecosystem of Bot Framework Development

Nowadays, a lot of good options are available as bot frameworks. In recent years the field of developing bots has encountered big demand, driving a lot of companies to offer integrated bot solutions with modern tools.

Before we start analyzing our decision on Microsoft Bot Framework choice, we are going to compare the most representative and popular solutions of the Bot Development field. Figure 2.1 shows some of these solutions.

Figure 2.1: *Bot Framework Ecosystem.*

Rasa

Rasa framework is an open-source framework that consists of two compartments Rasa NLU and Rasa Core. The first one stands for the NLU concept while the Core is responsible to help with the implementation of full functional chatbots.

Advantages

- The bot can be hosted on a user own server, without requiring official hosting services.
- You can customize your models and handle your data the way you believe it helps the most for your features.
- Custom models and libraries are implemented for performing analysis and reports upon AI chatbots.
- Supports chatbots that can get smarter through interactive learning.
- Free and paid choices are available for framework's tools and platform.

Disadvantages

- The SDK is supported only for Node.js, PHP and Ruby languages.

Dialogflow

It is a Google product. The framework uses Google's Machine Learning algorithms.

Advantages

- It is offered in Google cloud services.
- Its libraries and guides are offered for several languages (like C#, Go, Java, Node.js, PHP, Python).
- There are prebuilt agents for cases like vehicle system bots, events search.
- Multilingual Functionality (provides support for more than 20 languages).
- Free and paid choices are available for framework's tools and platform.

Disadvantages

- The scalability and training process needs enough time to automate.
- Entities and intents are mapped in a complicated way.
- Machine Learning algorithms are predefined.

Microsoft Bot Framework

Advantages

- ❖ Multilingual Functionality.
- ❖ Technical support for all services.
- ❖ There are prebuilt entities.
- ❖ Compatible with other Microsoft Services (Azure services).
- ❖ Free and paid choices are available for framework's tools and platform.

Disadvantages

- ❖ Software Development Kit (SDK) is supported for C#, Python and Node.js.
- ❖ Chatbot Services usually require paid account subscription.

IBM Watson

IBM Watson is an AI project that offers general AI solutions. Beyond its use for chatbot building, it is capable of succeeding holistic solutions, in the industry field, when the goal is to achieve optimization and automation.

Advantages

- ❖ Offers a built-in language translator.
- ❖ It is connected with machine learning engine and AI that offers flexibility and integrated solutions.
- ❖ There are prebuilt entities.

Disadvantages

- ❖ Too many tools that can prove to be confusing [7], [8].

2.2 Introduction of Microsoft Bot Framework Software Development Kit

In the previous section, we referred to some characteristic frameworks of the framework development ecosystem. Now, we are going to supply more details about our decision's functionality.

The Microsoft Bot Framework was selected for the following reasons:

- It is supported by Microsoft's development team.
- It is characterized by wide approval. For that reason, there is a well-organized documentation.
- The SDK is modular and extensible. Moreover, there is implemented a variety of tools to support the whole bot's life cycle.
- Microsoft Azure offers complete software solutions like Cloud and Database Services, deployment solutions.

2.3 The Implementation Life Cycle of a Chatbot

Below we are going to describe the life cycle, Fig. 2.2, that follows the development process of a chatbot.

Figure 2.2: *The Microsoft Bot's Life Cycle [9].*

Plan

As any typical development product, here we also need to specify how our product is going to work, what capabilities we consider to be necessary and which techniques we will take advantage of.

Build

In the build process the CI is defined. We apply the Bot Framework Service, one of the most useful components of the Azure Bot Service and then we start message and event handling. More details will be provided in the next section of bot tools. There, briefly we will mention popular capabilities that can be selected to be added to our bot.

Test

There are provided some really helpful ways to examine the behavior of our bot. Any bugs or unexpected interactions can easily be controlled with these tools:

- Bot Framework Emulator

  The emulator is a stand-alone application that helps the debugging stage.

- Web Chat

  That convenience is suitable for people who cannot access directly the bot.

- Unit Tests

  The developer can control specific features by writing specific test scenarios in order to test that functionality.

Publish

The publish phase offers web availability either with Azure publish service or with its own web service.

Connect

The connection step gives user accessibility to our services. Applications (like Facebook messenger, Slack, Twilio etc.) that support sending and receipt messages through their channels connect our clients with these popular platforms.

Evaluate

The Microsoft developers team has understood how important are analytics and reports for commercial and sophisticated architectures. The Azure portal reports for traffic, delays, failed connections and other analytics insights [9].

2.3 Bot Tools

Microsoft Bot Framework consists of features and services which are useful for the development process of a bot. Some of these are well defined and encounter great popularity. For that reason, these features and services are recognized more like tools. Some of them we are going to mention below. The maturity of the bot tools was one of the reasons for selecting Microsoft Bot Framework for the development of the thesis chatbot.

LUIS Applications

The task of proving to your bot a natural conversation feel can be proven to be difficult. For that reason, Microsoft Bot Framework offers a cloud-based Application Programing Interface (API), accessed with endpoints, that identifies entities and intents in user messages, an example is shown in Fig. 2.3. That Language Understanding service (luis.ai) makes it easy for us to handle conversationally and contextually the user's dialogs. It offers Train, Test and Publish options.

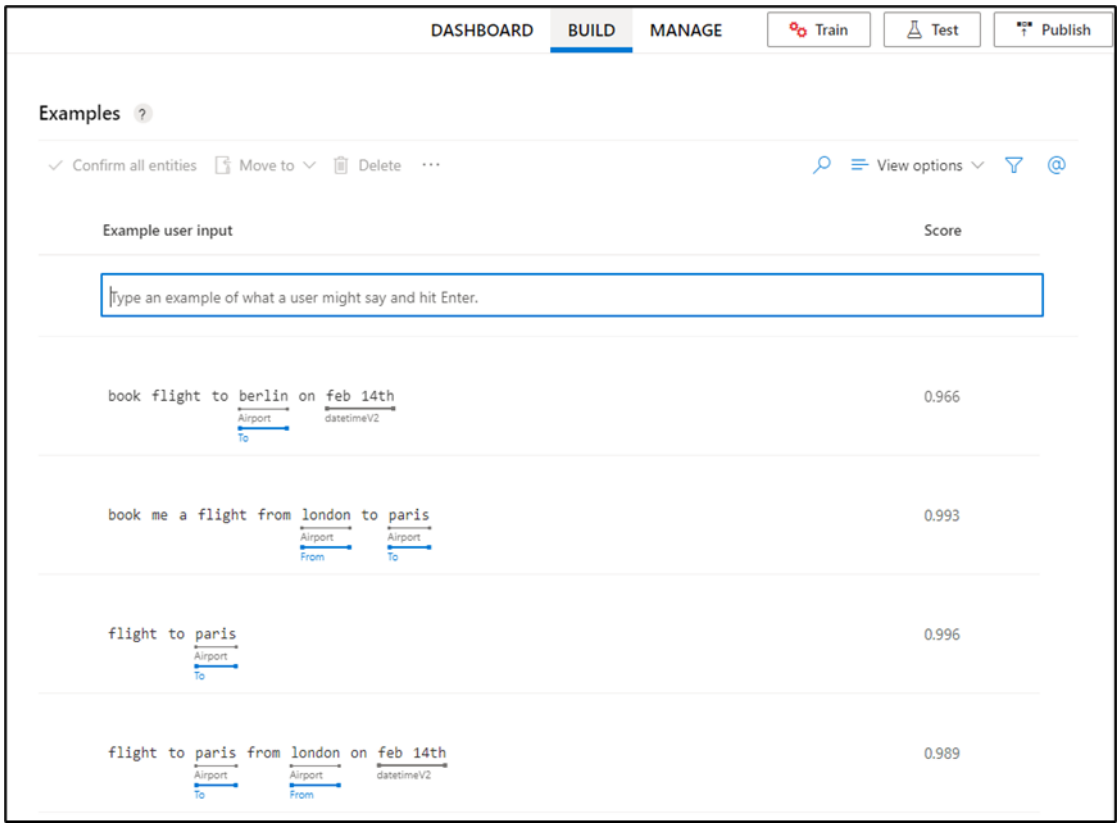


Figure 2.3: *LUIS entities section.*

The developer either with Command Line or from the LUIS platform can specify intents, entities and train the service for new utterances. These details are presented in a JSON format file. He can observe the LUIS intent score and how entities define the corresponding intent.

QnA Knowledge Base

When we have static information or for the case that after getting a request or question, we desire to respond the same answer every time, the QnA Maker is the appropriate tool. Actually, QnA Maker is a service that provides NLP through cloud API (qnamaker.ai).

The developer organizes pairs of questions and responds in order to turn his content into a knowledge base consisting of question and answer sets. Among the advantages of that tool is that you can use Frequently Asked Questions (FAQ) pages, support websites, excel files, product manuals.

About the way that question and answer are related we have:

- For the question we target, a number of alternative sentences should be also provided.

- ❖ During the search, metadata labels can help with the filtering of answer's selection.
- ❖ Follow-up prompts can be connected in order to make available a smoother flow [10].

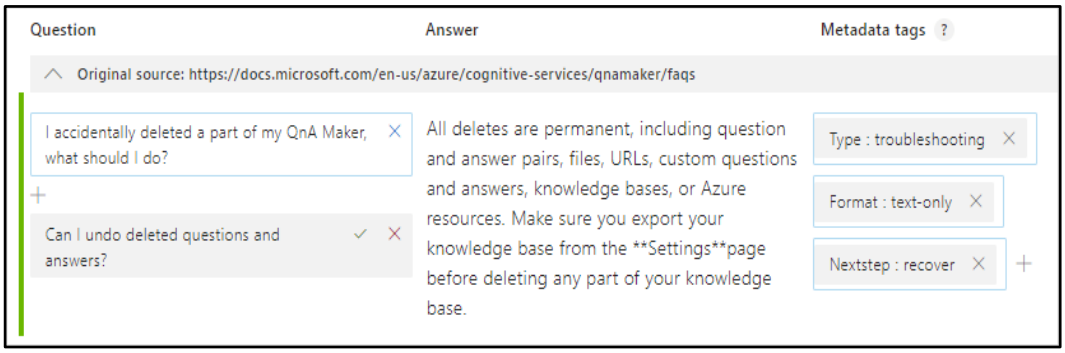


Figure 2.4: *Recorded QnA to Knowledge Base.*

Skills bots

In order to avoid complicated architectures a bot can be divided to skills and consumer. A skill is bot that can execute the role for which it is responsible. A skill consumer is a bot that can energize one or more skills. The primary bot that interacts with the user is called root bot and that is responsible to face user input demands.

For better file arranging the developer creates a JSON file that describes the tasks which the skill bot can perform, received and sent parameters, the skill endpoints that we can use. That file is called skill manifest and turns out to be a useful guide for a developer who does not have access to source code but wants to use that tool.

3 The Chatbot Role in the Context of Customer Support

3.1 Chatbots and the Transformation of Customer Support

The future of AI is really hopeful. Chatbots and virtual agents follow that path. The Businesses field has adopted the AI potential and is growing parallelly with it. Some of the areas, with automating impact, that we can observe that is, customer service, sales departments and marketing strategy. Regarding the customer service, chatbots can improve customer engagement and lead to great customer expectation. Below, we briefly explain for which specific aspects chatbots can help businesses to offer topmost customer support.

Provide Direct Customer Support

One common complaint of customers is the huge waiting times in order to be served. In thhe current way of living every customer, like us, wants to avoid long queues and get an instant answer, the time he wants it. With chatbots a business can reduce customer frustration of an impatient customer and offer great customer satisfaction.

The instant customer service, enhanced from an implemented chatbot, can be available any time, even not working hours of the business customer support department. By developing 24 hours a day/ 7 days a week customer support a business brand name can reinforce satisfaction and loyalty. Even the resolution of a trivial issue regarding common FAQs can engage customer and improve his satisfaction.

Chatbots Reduce Costs

Instead of hiring people and paying salaries chatbots can lead to dramatic reduction of operation costs. By deploying a chatbot the business can deal with a great volume of queries.

A chatbot compared to a human agent offers advantages regarding training costs and occupation of infrastructures. There is a wide variety of automated or customized chatbot solutions that are offered at reasonable prices and have proved to be valuable assistant to the goal of improving the customer experience.

Collect on the Spot Customer Feedback

Another key subject that transformed customer support is the way AI agents record customer feedback. Feedback is a valuable resource that offers insights of the business weaknesses. The modern approach of implementing chatbot solutions records in real time the customer experience without to require forms or filled up questionnaires.

Simply, by questioning a human kind query at the end of chat dialog a chatbot can collect the valuable feedback. Automated and well-trained agents can perceive when is the appropriate time to collect the feedback and which are the questions that respectively apply to each customer's case [11].

3.2 Role Description

The main aspect of the current thesis is the design and development of a chatbot that helps and interacts with a platform user. In order we complete that aspect, we need firstly to clarify the role and the scope of the desired chatbot.

As we mentioned in the above unit, chatbots have met great acceptance in the domain of businesses. All the time, new approaches and services are being offered. The implementation of chatbots in the domain of customer support has encountered remarkable interest. The tendency is gradually chatbots to offer such support that human intervention will not be necessary.

Of course, there are cases where chatbots have not accomplished that dynamic to outperform human capabilities. However, generally, chatbots intend to replace human inference, without reducing user satisfaction. In that concept, we have been attracted to examine a chatbot development in the case of hotel business. The case of hotel business is a characteristic example where

customers want to be informed about issues related with the hotel. For that case a chatbot approach seems to be ideal as it outmatches a human agent in the aspect of handling a bigger number of questions, simultaneously, any time during the day with a predictable and well-organized manner.

Inside that context, we define our chatbot role description as the achievement of a human-like dialog between chatbot and customer, for the case of hotel business. Our chatbot will be a closed domain chatbot, compared with a human employee, however we aim for it to be able to outcompete him in specific aspects. Below, we are going to declare specifically the desired role of our chatbot.

3.3 Functionality Goals

We have already mentioned that the chatbot we are going to design and develop is going to support the needs of a hotel business. Going back to chapter 1, where we defined different chatbot categories, now, for our case we can declare our CI as a chatbot which has AI capabilities, that is based in ANN, and are more capable of a Rule Based System [12].

After the declaration of the general role that our chatbot is responsible to complete we are going to define concrete goals of the intended functionality we target to achieve.

As it is a chatbot:

a. **Communication entirely in Natural Human Language, for a specific domain, for English language.**

The chatbot we intend to design will understand Natural Human Language. There are other types of inputs that CI can understand like, voice, images – computer vision, but we do not aim to include them. NLP is enough to offer an easy, at the same time, efficient way of communication. The speaking chatbot language will be English. English is a common global language. According to Eurostat Statistics [13] 96% of students of secondary education attended English as a foreign language. While, based on Statista [14] in 2021 English dominates globally as the most spoken language (either natively or as a second language).

b. **Availability for customer support, 24 hours a day, 7 days a week.**

It was previously mentioned that one of the aspects that transformed customer support, with the advent of chatbots, is the real time customer service.

A chatbot is needed to be available all the time even when the hotel business has no employee in the reception.

c. **Ability to deal with different customer requests, without to create long time waiting queues.**

The chatbot service is going to run remotely in a cloud service and customers will be able to question in real time different aspects without need to wait in a queue.

d. **Capability to predict the prompted answer, for a specific domain.**

In order to gain customer satisfaction, we should be able to predict any reaction of our chatbot to customer input. The way that chatbot has been built should be concrete in order we can define which issues it can handle. In our case, we aim to deal with issues closely related with the operation of a hotel business. We are not going to support decision making of chatbot in Generative models that produce «unpredictable» responses to customers.

As we aim to be capable of AI functionality:

a. **Achievement of human-like interlocutory behavior.**

Our goal is the chatbot to be able to have a human-like behavior. An amicable character can offer familiarity in the dialog with the customer.

b. **Ability to understand Natural Human Language without requiring creation of a rule for each case.**

The very early chatbots were required to be implemented by a rule-based model. Today, modern techniques offer the capability to use NLU approaches to cover the need of defining the inputs that are responsible for triggering the corresponding rule.

c. **Intent identification, with use of NLU techniques.**

Targeting to build a chatbot that has predictable behavior and is well defined, we will classify every customer input to specific intents. That is important in order we can classify the user utterances and in the continue extract the valuable entities. Of course, here we tap into NLU strengths.

d. **Reasoning for choosing a concrete decision.**

We want chatbot to be able to return the same answer based on a reasoning. By using AI we can classify utterances based upon a similarity ratio. And by adjusting that ratio we can achieve better fitting of our model.

e. **Ability to conversate with human-like dialogs.**

In some cases, we do not want to support just a simple FAQ list. But we need to guide our customer to a dialog in order to extract the information we want and give him the valuable response.

3.3 Limitations

Having declared in the previous unit the functionality that we target our chatbot to be able to achieve, we are going now to define the bounds within the functionality, we desire, will be placed. For each individual functionality goal we are going to describe the extent at which every goal is going to be completed.

As it is a chatbot:

a. **Communication entirely in Natural Human Language, for a specific domain, for English language.**

The chatbot will provide customer support regarding issues related to the hotel business. It could handle questions regarding the FAQ list of the hotel, and provide availability information. A customer will be able to get an answer related to information like: what services does the hotel offer; how can he get there or room availability information for the date period he desires to visit the hotel. Dialogs and details unrelated with the operation of the hotel will not be supported.

b. **Availability for customer support, 24 hours a day, 7 days a week.**

The proposed chatbot solution will be available all the time. The Azure platform offers cloud services that are available 24 hours a day, 7 days a week.

c. **Ability to deal with different customer requests, without to create a long time waiting queue.**

The current implementation offers simultaneous handling of different customers' requests. The way we have organized the management state requires the completion of customer dialog to be able a new customer to use the chatbot effectively, when it used in a website that does not support login with social media platforms. For cases where the customer can be identified with his personal social media account, we support the simultaneous handling of different dialogs from different customers.

d. **Capability to predict the prompted answer, for a specific domain.**

For the specific field of a hotel business a knowledge base has been created. In that knowledge base we have stated concrete pairs of questions and answers.

As we aim to be capable of AI functionality:

a. **Achievement of human-like interlocutory behavior.**

We desire to offer a familiarity at the dialogs that the chatbot will perform but that it is not the main goal. Some simple sense that the dialogs have vivacity is going to be offered, but we do not intend to pursue the customer that he talks with a real human. Complex conversations and answers based upon Generative Models is out of the scope we intend to implement.

b. **Ability to understand Natural Human Language without requiring creation of a rule for each case.**

The input of the customer does not require to be exactly the same with the user input that we used to train our model. The cognitive services that we use are able to classify a new input to an intent which has similar utterances examples.

c. **Intent identification, with use of NLU techniques.**

NLU technology is used to specify which intent is the most likely to be the correct one. We developed a small number of four intents, that are enough to fulfil our chatbot needs.

d. **Reasoning for choosing a concrete decision.**

The decision of selecting one intent compared to others is based on a measure of similarity between the new utterance and the pre-existed utterance examples. Having completed the decision reasoning stage the scheme proceeds to others stages that we have specified.

e. **Ability to conversate with human-like dialogs.**

As previously has been declared, the chatbot will stand for customer support upon hotel FAQ and room availability. It aims to be used on a social channel, like Facebook, or the official hotel website. Its role is specific and the dialogs that will be able to conversate are related with the issue of room availability. The chatbot will engage the customer in a dialog from which it will extract the necessary details in order to reply for the existence or not of available rooms.

4 Chatbot Building Process

4.1 Overview of the Next Steps

In sections 2.3 and 2.4 we talked about the implementation life cycle of chatbot and the bot tools respectively. The Bot Framework SDK as the most Bot Frameworks owns a documentation that helps to get familiar with the Framework's architecture. In the official Bot Framework page [9] there is a complete guide of how to perform the basics. The documentation is well organized and is supported by a Microsoft team which continuously keeps it updated, owning several repositories on GitHub [15] which include sample code of the documentation chapters.

Even though, the helpful documentation when someone tries to build something more complicated, he needs to investigate things on his own. The sample code is enough to get a good perception and start your own ideas trials.

The current chapter has been created with only aim to help a new one on the Microsoft Bot Framework to get a complete idea of how different things can be combined to build a helpful chatbot application. We go beyond the trivial cases and try to build an automated bot that can support the customer department of a hotel business.

Based on section 3.3, where we defined the goals of the functionality we target to implement, we guided to plan, build and test stages of our bot life cycle. Having followed these stages forward and backward, several times, we ended up implementing the abilities of the Bot Framework that we describe below.

We declared that our chatbot will be able to handle a FAQ list but also to provide information about room availability. To get the required details about the query related with room availability our bot will call a dialog process. With the LUIS applications (section 4.2) we will understand the intents of different utterances while with SQL language (section 4.3) we manage the book details of the hotel business.

4.2 LUIS

A bot, since it is not human, needs a way to understand natural language. That need can be satisfied with the use of a cloud-based API. LUIS (luis.ai) is a service below the Microsoft Cognitive Services that offers bot ability to recognize intents and entities on every utterance.

4.2.1 LUIS Application

To be able to create LUIS applications we need to create an Azure account on Azure portal [16]. After that we create a new resource group. The process is easy as we see in Fig. 4.1 and 4.2. For more details about the management of resource group visit the website [17].

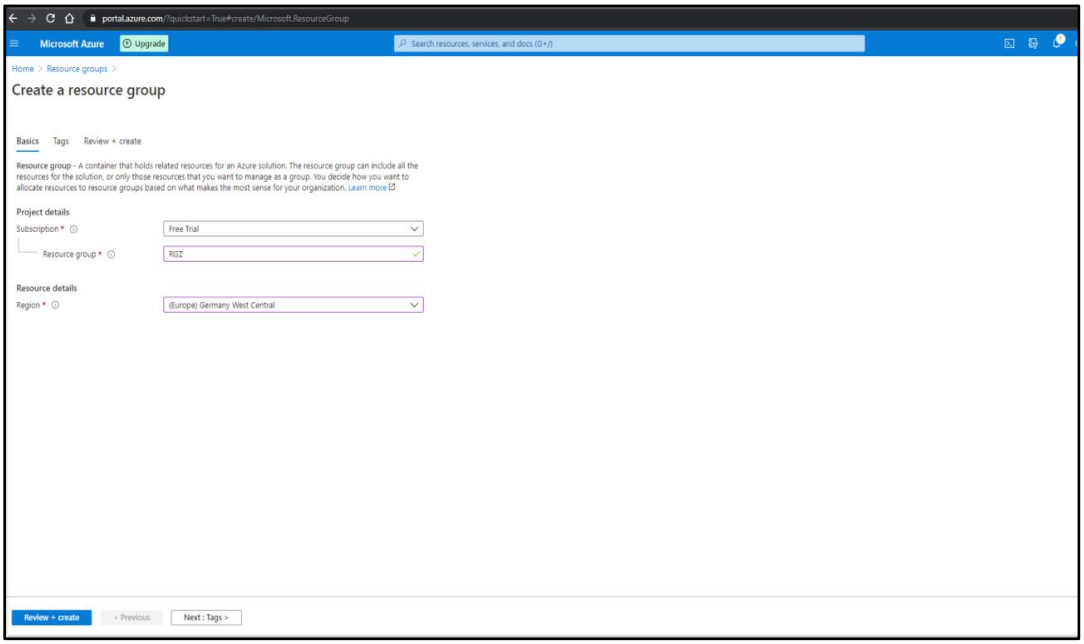


Figure 4.1: *Azure portal, fields to fill to create a new resource group, part 1.*

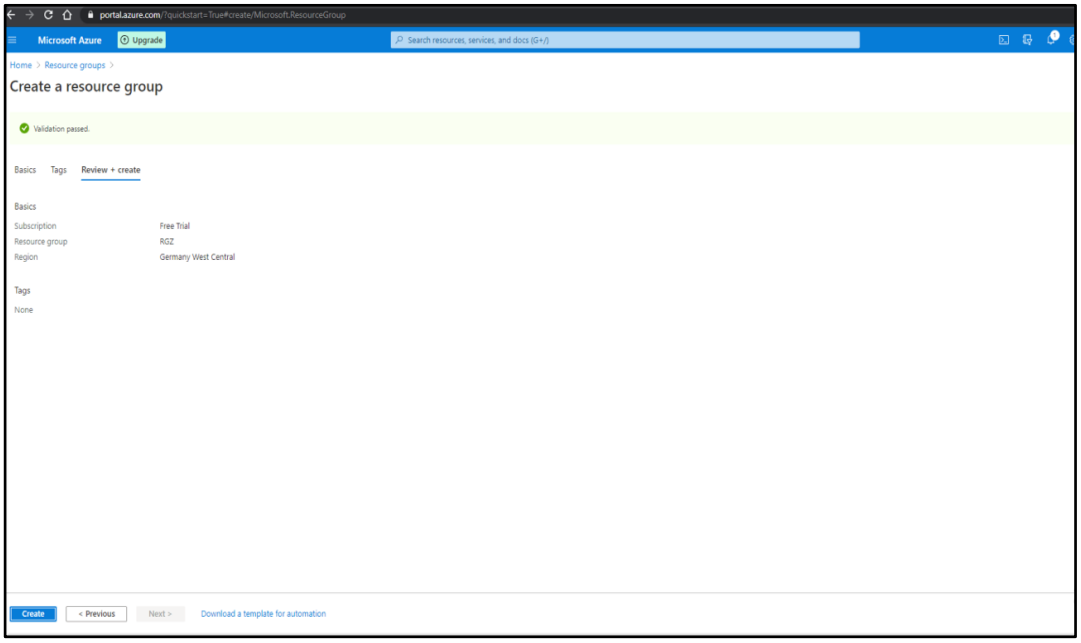


Figure 4.2: *Azure portal, fields to fill to create a new resource group, part 2.*

Having created a new resource group, we get to luis.ai to create our new applications. As we see in Fig. 4.3 the first time, we get to luis.ai we need to create a new resource below which we are going to build the new LUIS applications. We may need to wait a couple of minutes after the creation of resource group to get that available as an option in the creation of new authoring resource.

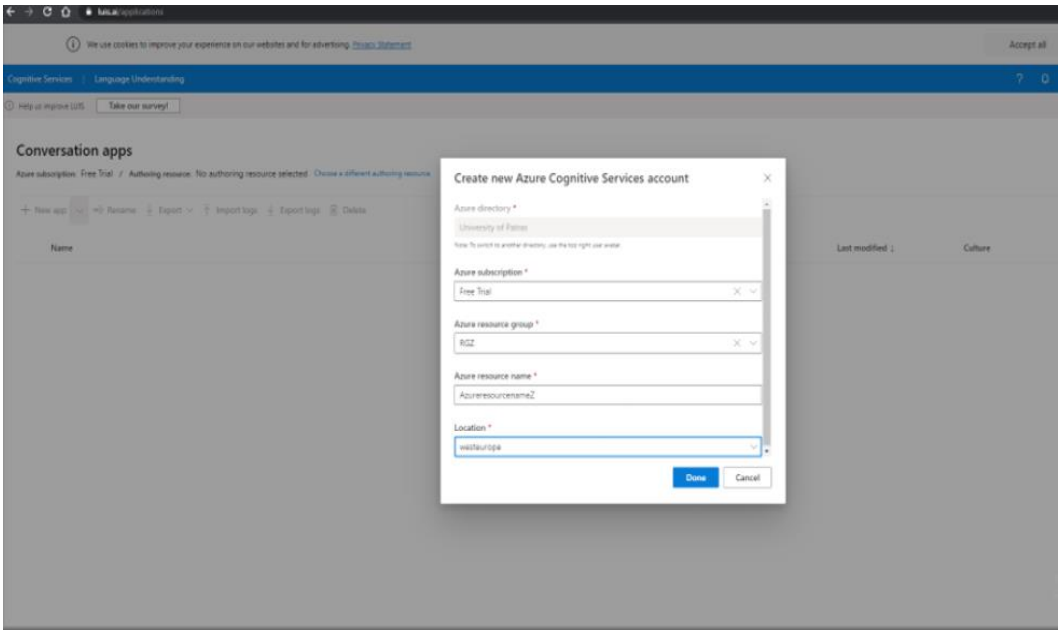


Figure 4.3: *Azure portal, fields to fill to create a new Azure resource name.*

Now, we are going to proceed with the creation of two LUIS applications. There are two ways to accomplish it. The first is from the interface of the luis.ai. We select “New app” and we fill the fields with the desired values. To continue we can define the intents and entities from the menu, see Fig. 4.4.

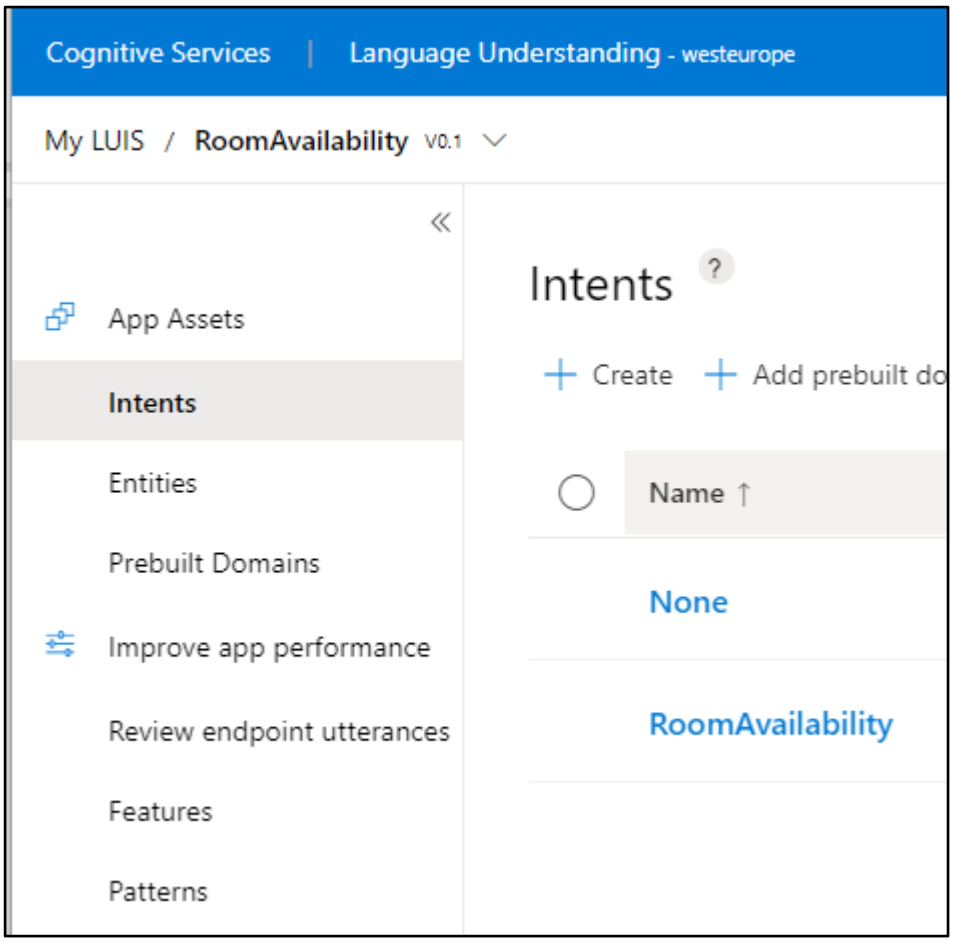


Figure 4.4: *LUIS platform interface with the supported options.*

There is also an alternative way to create the intents and the entities immediately, from the LUIS interface, with the import of a JSON type file which includes the same information. At the GitHub repository, we were referred earlier, on the samples folder and more specifically at the cognitiveModels file of the corebot sample we can find a JSON example for the case of flight booking LUIS bot app.

The second way is to create a LUIS application by using the LUIS command line interface (CLI). With the installation of the Bot Framework CLI we can have access to the same choices as with the luis.ai interface.

There is a complete tutorial for that here [18].

For our case, of the hotel business chatbot we created two LUIS applications. One, for the cases that utterances include book details and one for

cases where there are not provided any book details and the bot is responsible to extract each of these, one by one. So, we execute the process two times, one for each application. We need to record the application ID because we will need them later.

4.2.2 Knowledge base

Wanting to include the ability our chatbot to answer queries about the hotel business FAQ list we decided to make use of a helpful skill that a chatbot can implement. For that reason, we designed a knowledge base. Below we mention the steps we have followed.

Firstly, it is needed to create a new QnA service. We can perform it from the Azure portal. We fill the fields with values as in the below Fig. 4.5 and 4.6.

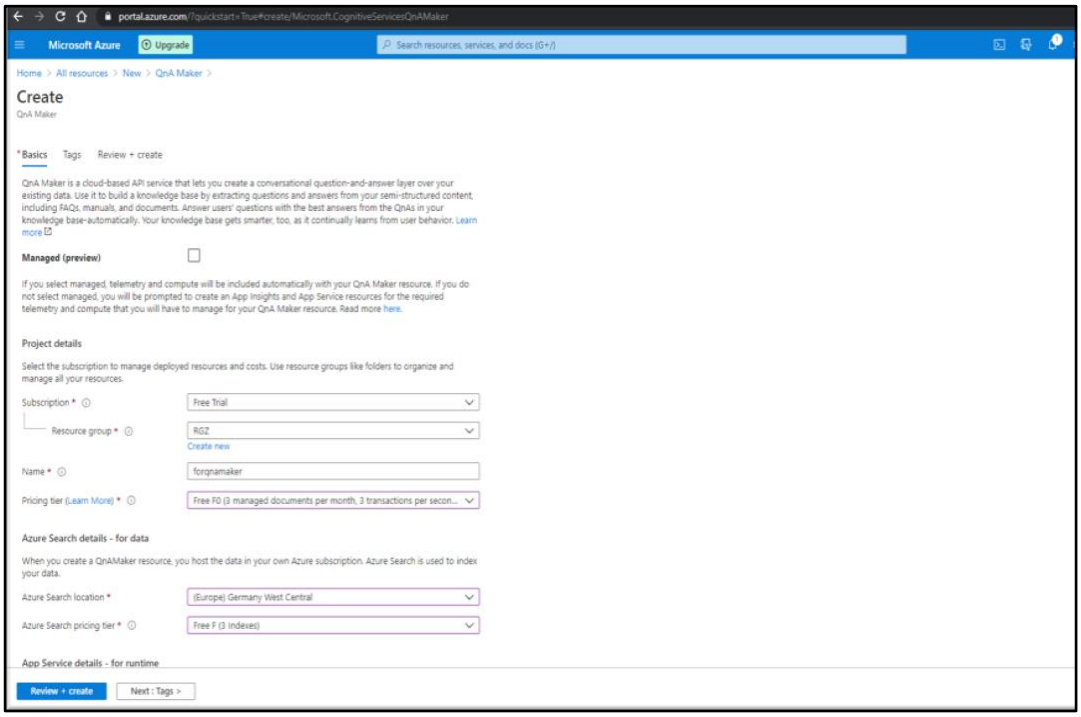


Figure 4.5: *Filling the fields of a new QnA Maker service, part 1.*

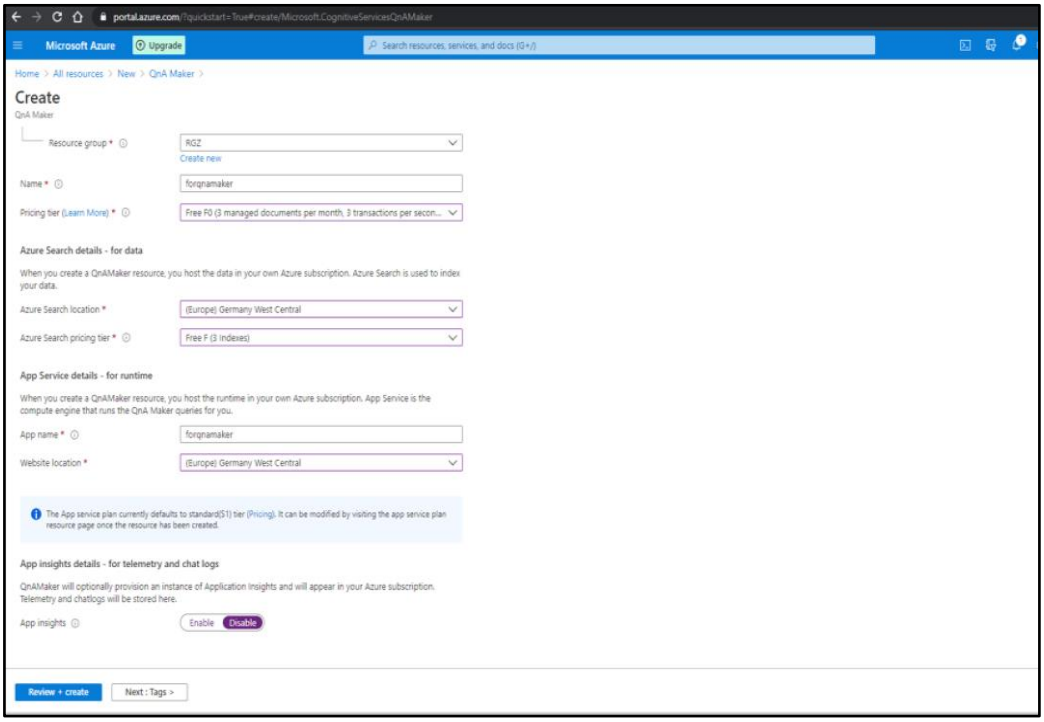


Figure 4.6: *Filling the fields of a new QnA Maker service, part 2.*

The input values we filled above are summarized in the below Fig. 4.7.

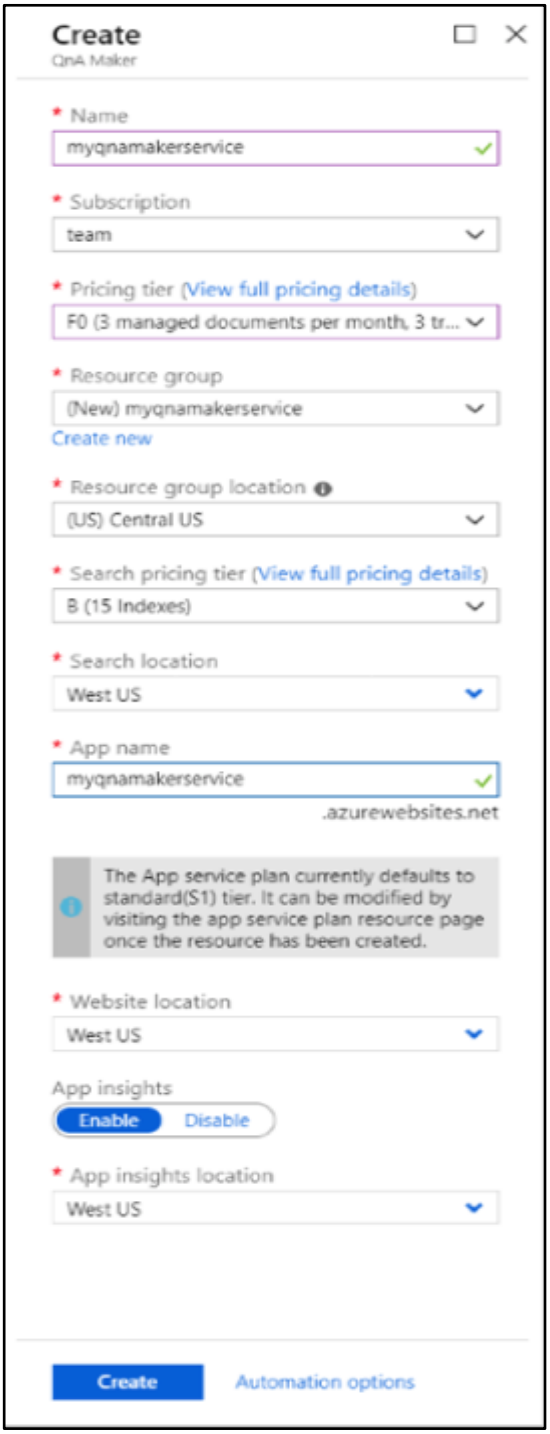


Figure 4.7: *Review of the selected options for the QnA Maker service.*

The QnA Maker service may take a couple of minutes to get available. To continue, we browse the qnamaker.ai and select “Create a knowledge base”. We fill the fields like below Fig 4.8, 4.9 and 4.10.

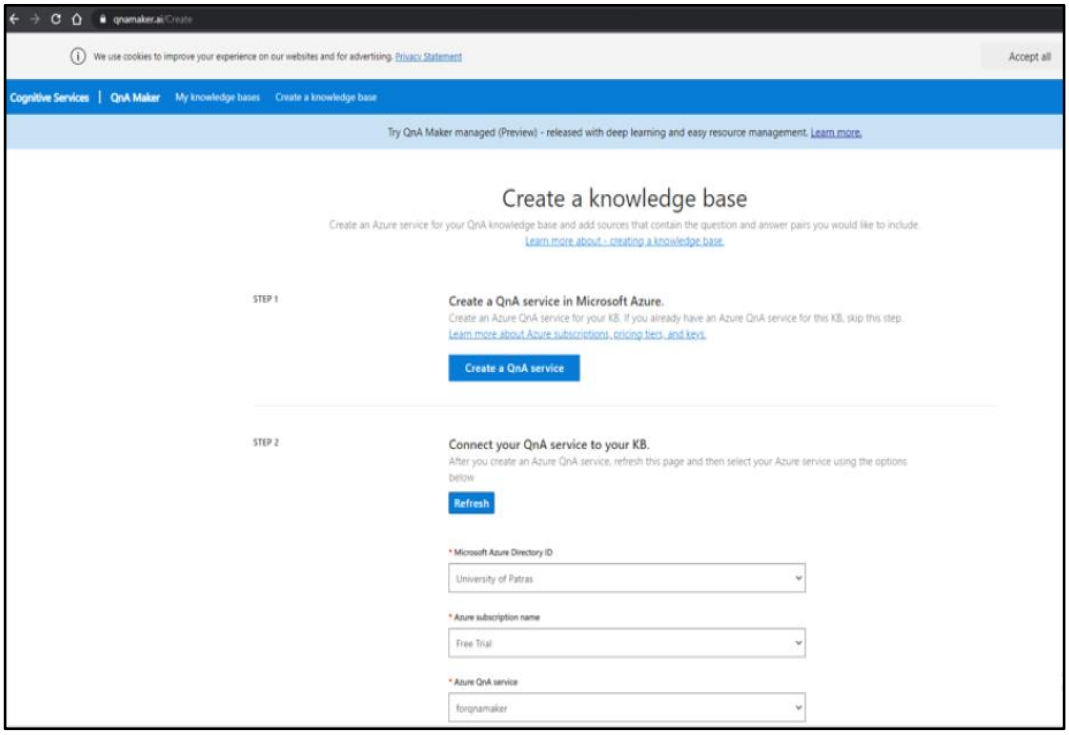


Figure 4.8: *The required options for the creation of a new knowledge base, part 1.*

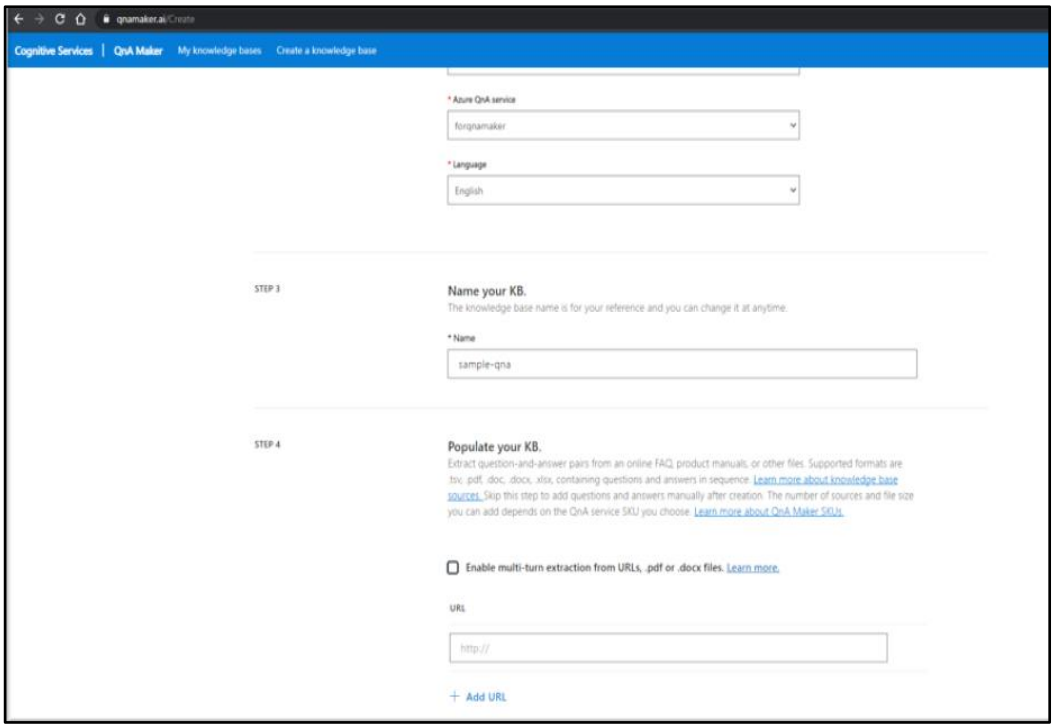


Figure 4.9: *The required options for the creation of a new knowledge base, part 2.*

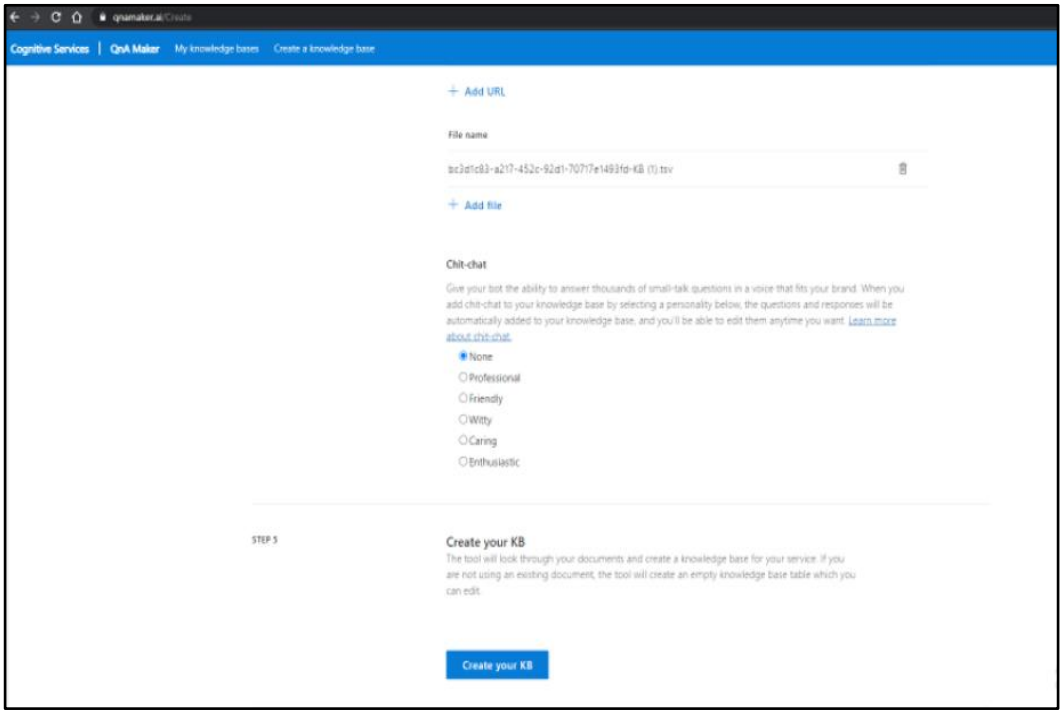


Figure 4.10: *The required options for the creation of a new knowledge base, part 3.*

In above Fig. 4.10, on step 4 we see that there are three ways to create pairs of questions and answers.

- The first way is to enable extraction from urls, pdf, xlsx, tcv and others. That option is useful. The knowledge base may be required to be changed, for that reason a file is useful to keep there the pairs of QnAs and update as required.

- The second method is to skip step 4 and get to qnamaker.ai afterwards and add one by one the question and answer pairs. In Fig. 4.11 we see how we can insert questions and answers as pairs. We can group differently formatted questions and pair it with exactly one answer. For the answer part we can insert options like text, picture hyperlinks and others.

- The third way is to select a combination of the above two ways.

There is also available, on step 4, the option to set a personality upon the knowledge base we will add. We can select among Professional, Friendly, none and others. That selection adds a set of predefined answers and questions based on the personality we have selected. The profit of that is to be able to answer common questions of general interest that a customer usually may ask.

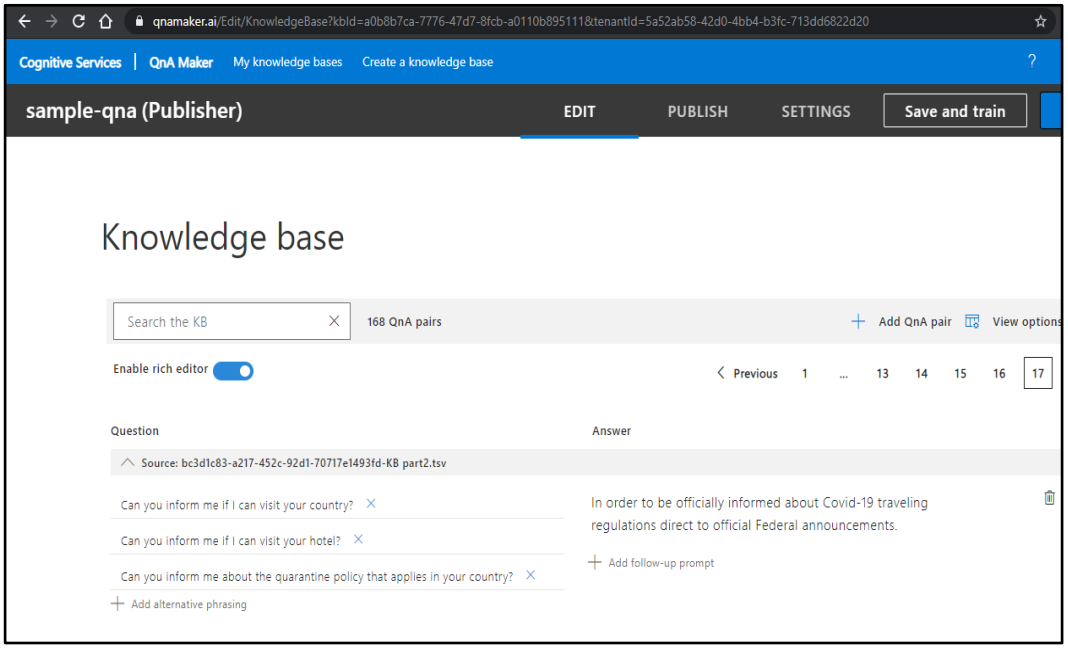


Figure 4.11: *The knowledge base interface for recording pairs of QnA.*

That was the process for the setup of the QnA Maker service in Azure. We need to record the Cognitive Services Key 1 provided for the QnA Maker service. This will be used as <azure-qna-service-key1> when adding the QnA Maker application to our dispatch application.

More analytic instructions someone can find here [19].

Until now we have shown how to create LUIS applications and knowledge base. In the next section we connect them and explain how to create a LUIS application that has embedded in one application the above services.

4.2.3 Dispatch File

In the current section we are going to create one joined-up file that will include the functionality of the above LUIS applications and knowledge base. That file will integrate all these together. That is the final LUIS application which our chatbot is going to use for its operation. Its role is to decide whether sub LUIS applications or knowledge base best matches the user input.

Until now we did not execute any training and testing steps. The tutorials we were referred above mention training and testing steps. Of course, that is a good practice that can inform us about vulnerabilities. However, until now we did not show something like that for simplicity purposes. We encourage anyone who wants to develop the chatbot we describe for the hotel business to follow step by step the complete instructions from the official tutorials.

Earlier, we presented the capabilities of the Microsoft Bot Framework. There, we mentioned that someone can use one of his favorite programming languages. At the current step we need to base upon a copy of the NLP with dispatch sample, located at GitHub repository [15]. For the current thesis we selected JavaScirpt as language of our choice.

For our next steps the working directory will be cognitiveModels folder. There we should have the JSON files that correspond to LUIS applications that were created at unit 4.1.1.

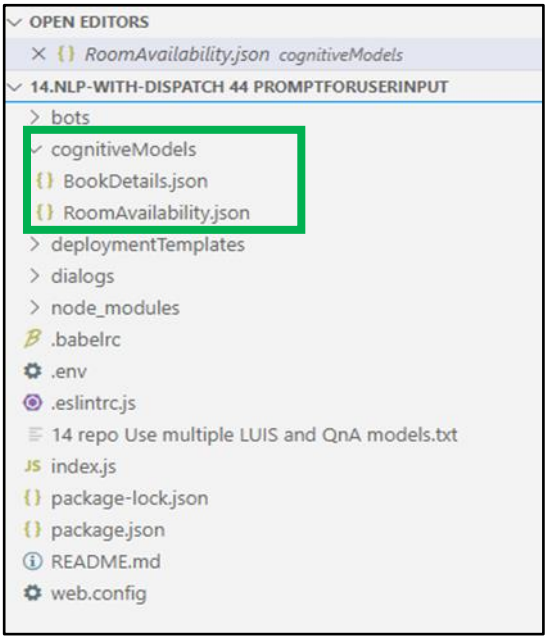


Figure 4.12: *The containers of project folder.*

On the command line we execute the following commands:

- npm i -g npm
- npm i -g botdispatch
- dispatch init -n dispatchfile --luisAuthoringKey "6f7981b17e7a406b9e88a78715386f93" --luisAuthoringRegion westeurope
- dispatch add -t luis -i "d1219549-a117-42cf-b383-01922f986371" -n "BookDetails" -v 0.1 -k "6f7981b17e7a406b9e88a78715386f93" --intentName BookDetails
- dispatch add -t luis -i "b2e0bdcf-922d-437c-8e98-2019f03876fb" -n "RoomAvailability" -v 0.1 -k "6f7981b17e7a406b9e88a78715386f93" --intentName RoomAvailability

- dispatch add -t qna -i "a0b8b7ca-7776-47d7-8fcb-a0110b895111" -n "sample-qna" -k "8b93602be98547d6a8b8f052580d7d30" --intentName q_sample-qna
- dispatch create

We replace the corresponding parameters with the correct values that apply in each case. After that we insert the service connection values in .env file.

```
MicrosoftAppId=""
MicrosoftAppPassword=""

QnAKnowledgebaseId="<knowledge-base-id>"
QnAEndpointKey="<qna-maker-resource-key>"
QnAEndpointHostName="<your-hostname>"

LuisAppId=<app-id-for-dispatch-app>
LuisAPIKey=<your-luis-endpoint-key>
LuisAPIHostName=<your-dispatch-app-region>
```

Figure 4.13: *The parameters that .env file can define.*

Now, our dispatch application has been created. For more details of how programmatically we can call it refer here [20].

4.3 Storing and Retrieving Data

In the current section we proceed with the setting of an SQL server and database. The aim of building the hotel business chatbot is to replace the need of a human at the reception section. Our chatbot needs to have access to booking details. That is necessary because we should support the functionality, a customer to be able to start a dialog with the chatbot to be informed about room availability.

We selected SQL as database language. On Azure platform someone can host his database. We decided to follow that way to have all of our data saved in the same platform with our application and avoid multiple accounts and settings. Additionally, Azure platform offers plenty of useful tools like performance overview, metrics and alerts, support and troubleshooting and others. The SQL services are well designed and quarantine smooth operation.

4.3.1 SQL Server and SQL Database

Now we proceed with the creation of a SQL server and a SQL database. We get in Azure portal and select “Create SQL database”. In Fig. 4.14 we see the fields that are required to be filled. Parallelly with the creation of the SQL database we select “Create new”, below the server selection field. We fill up the server parameters, select “OK” and “Review + create”.

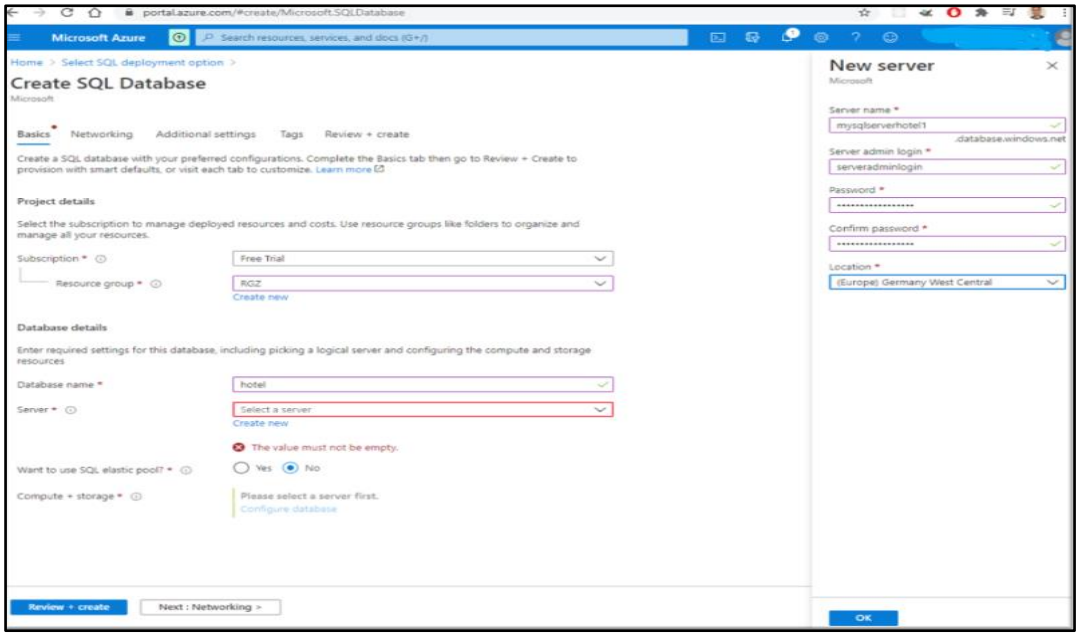


Figure 4.14: *Filling the fields to create SQL Database and SQL Server.*

In Fig 4.15 we see some of the options that are offered from Azure for our server. We can browse between them. However, at least we should configure the firewalls option. We need to add a rule with the IPs of the client that the firewall should consider valid to access the server. That is important because in other ways our application will not be able to process the stored booking details of the hotel business.

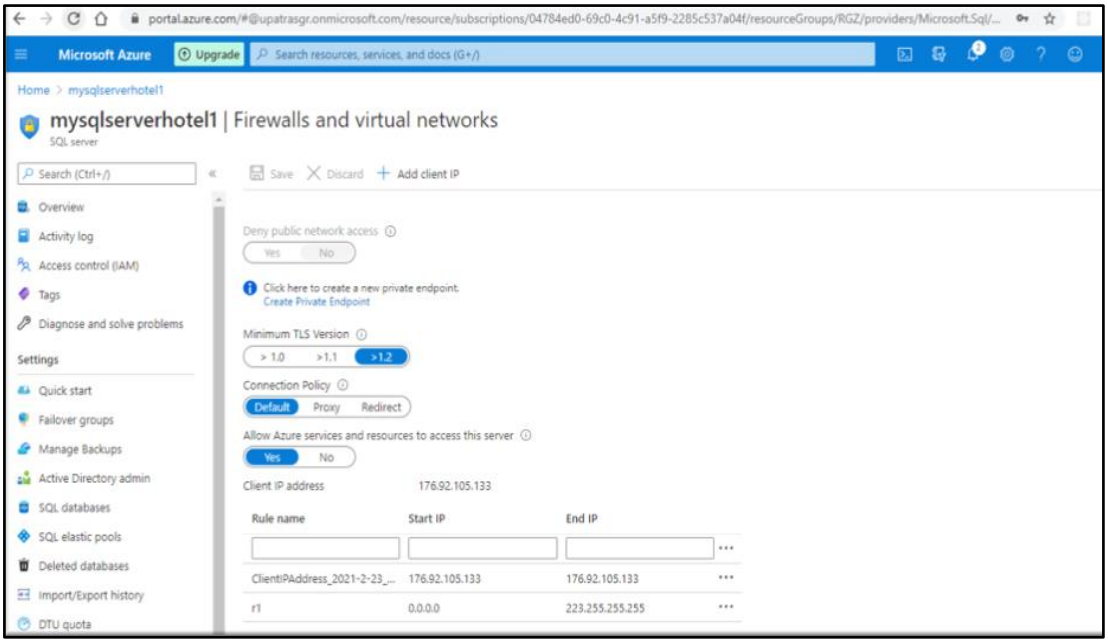


Figure 4.15: *Edit access for IPs to the service.*

4.4 Incorporation of Skills

The chatbot we designed to be able to satisfy the declared goals-limitations should combine different skills. We went beyond a core bot and tried to incorporate basics skills to develop a robust chatbot solution. The following list includes the main skills we adjusted as it was required to support the desired chatbot behavior.

- Send and receive text message
- Save user and conversation data
- Create your own prompts to gather user input
- Implement sequential conversation flow
- Create a bot using adaptive, component, waterfall, and custom dialogs
- Handle user interruptions in adaptive dialogs

Each one from the above skills works uneventfully. Though, when a combination of all these skills is tried, the final functionality turns to require an attentive handling. In the next chapter, where we will talk, among others, about results and difficulties, we will analyze them by citating specific cases.

4.5 Microsoft Bot Framework Emulator

The steps with the instructions of how to create the dispatch core bot are following trials to add new code parts and configure the desired functionality. Really helpful in that process is the Bot Framework Emulator.

Bot Framework Emulator is a desktop application that serves the need of bot developers to test and debug bots they developed. Instead of deploying bot in Azure cloud we can send and receive messages locally. Beyond avoiding the testing in the cloud we also acquire the ability to inspect the exchange information in the chat UI together with logs from the JSON requests and answers between bot and user.

The following Fig. 4.16 and 4.17 depict how the emulator looks like and how the debug information is presented. More details can be found here [21].

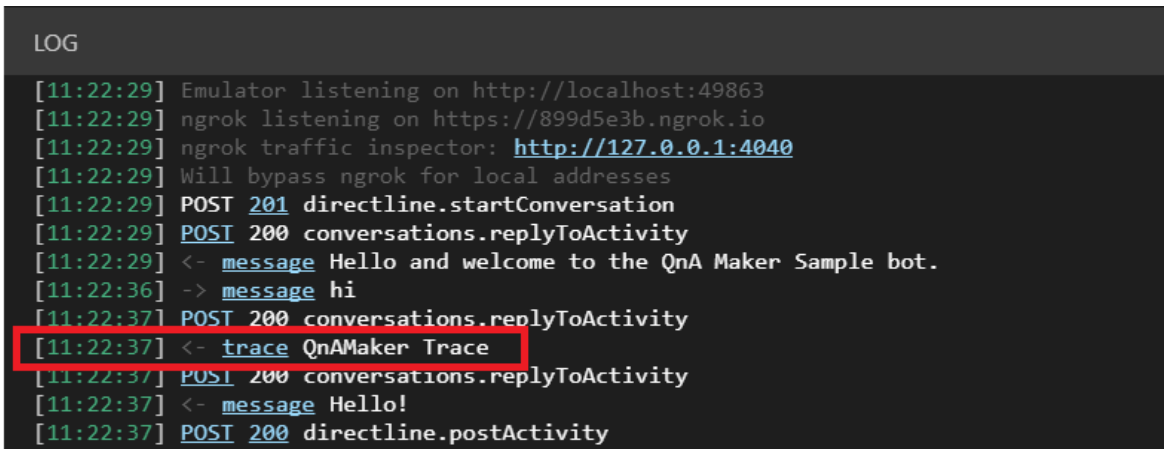


Figure 4.16: *Microsoft Bot Framework Emulator environment, example application.*

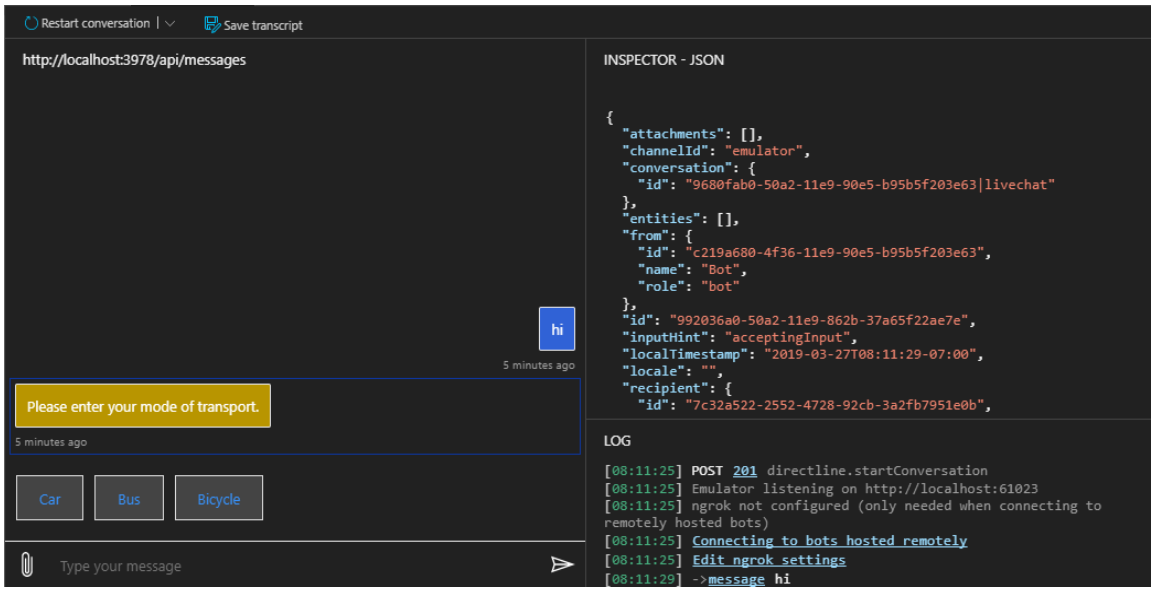


Figure 4.17: *Microsoft Bot Framework Emulator environment, log information from example application.*

4.6 Bot Deployment

When we have completed the steps of creating a chatbot it is time to test it. The Emulator is a great application for that case. We move from building the chatbot to testing it and back again. That process had been repeated until we finally embed all the different functionalities, we have planned for our chatbot. The time we have finished with that we are ready to deploy our chatbot. However, the deployment is not performed just once. It may be repeated in case we find differences in the behavior of local and deployed chatbot.

We start by opening the command prompt to the file we have saved our project. We run the below commands to set our subscription of choice:

- az login
- az account set --subscription "<azure-subscription>"

After that, we create a new Azure application registration with the following command:

- az ad app create --display-name "displayName" --password "AtLeastSixteenCharacters_0" --available-to-other-tenants

We keep notes for the password and the appId that is returned with JSON. We will use them later.

Now, it is our turn to create Bot Application Service, by deploying ARM template (with existing Resource Group) and using New App Service Plan.

We run the below command with values (appId, appSecret) we recorded earlier.

- az deployment group create --resource-group "<name-of-resource-group>" --template-file "<path-to-template-with-preexisting-rg.json>" --parameters appId="<app-id-from-previous-step>" appSecret="<password-from-previous-step>" botId="<id or bot-app-service-name>" newWebAppName="<bot-app-service-name>" newAppServicePlanName="<name-of-app-service-plan>" appServicePlanLocation="<region-location-name>" --name "<bot-app-service-name>"

Having arrive here we are ready to prepare our local project filles by running:

- az bot prepare-deploy --code-dir "." --lang Javascript

A new file is generated (web.config). That file is necessary to be placed in the root folder of our project and will work for Azure App Services to get a successful result.

For our case, when we are developing a JavaScript chatbot we change folder so to get where we have app.js or index.js files saved. From that file, and having included all the dependencies files in our project file we create a single zip file inside the current project folder. For the last command we are going to use Kudu engine which is a way to deploy components to Azure website. Kudu deployment considers by default that it deploys from zip fills and does not need to follow additional build steps. So, we target to the path and zip file name to run the last command:

- az webapp deployment source config-zip --resource-group "<resource-group-name>" --name "<name-of-web-app>" --src "<project-zip-path>"

That was the chatbot deployment procedure. We are ready to observe our chatbot behavior on WebChat, by incorporating it on a website or by connecting it to a channel.

BLANK PAGE

5 Results and Evaluation

5.1 Overview of the Developed Chatbot

In the previous chapter we described how we can create a chatbot that is designed to offer customer support for the case of a hotel business. Now, we will present the actual functionality that we have developed, results of what is actually capable our chatbot to perform and cases that it cannot handle so effectively. In order to develop a realistic chatbot, we used information from a real 5 stars hotel [22], that belongs to a well-recognized luxury hotel group.

Our chatbot is responsible for answering customer's input messages. So, when a customer triggers, with his message, our chatbot, its role is to get that input and handle it according to our designed goals. We have created a Bot Application Service that is responsible to serve our chatbot in Azure cloud. That service communicates with the LUIS dispatch application we have earlier created. The LUIS dispatch application service can be accessed through the values we filled in the .env file, inside our project folder.

As we saw in section 4.2.3 (Dispatch File) the Dispatch Application is a LUIS application that includes three resources, a knowledge base and three LUIS Applications (BookDetails, RoomAvailability, None). The None intent is automatically created in every LUIS application. Our chatbot accesses that dispatch application by a cloud API and that is the first layer where our chatbot architecture decides which of the three resources to use. A schema of our chatbot developed architecture is depicted in Fig 5.1. There we can see the next layer that is defined from a more detailed description.

In the second layer, we see that our Knowledge Base is divided into three subparts. The first one includes QnAs predefined from the friendly Chitchat personality that we have selected when we were creating the knowledge base. The second part includes QnAs from the official FAQ list the hotel hosts in its official website. The third part, that is included in our knowledge base, consists

of information that is included in the hotel site and we considered to be possibly questioned.

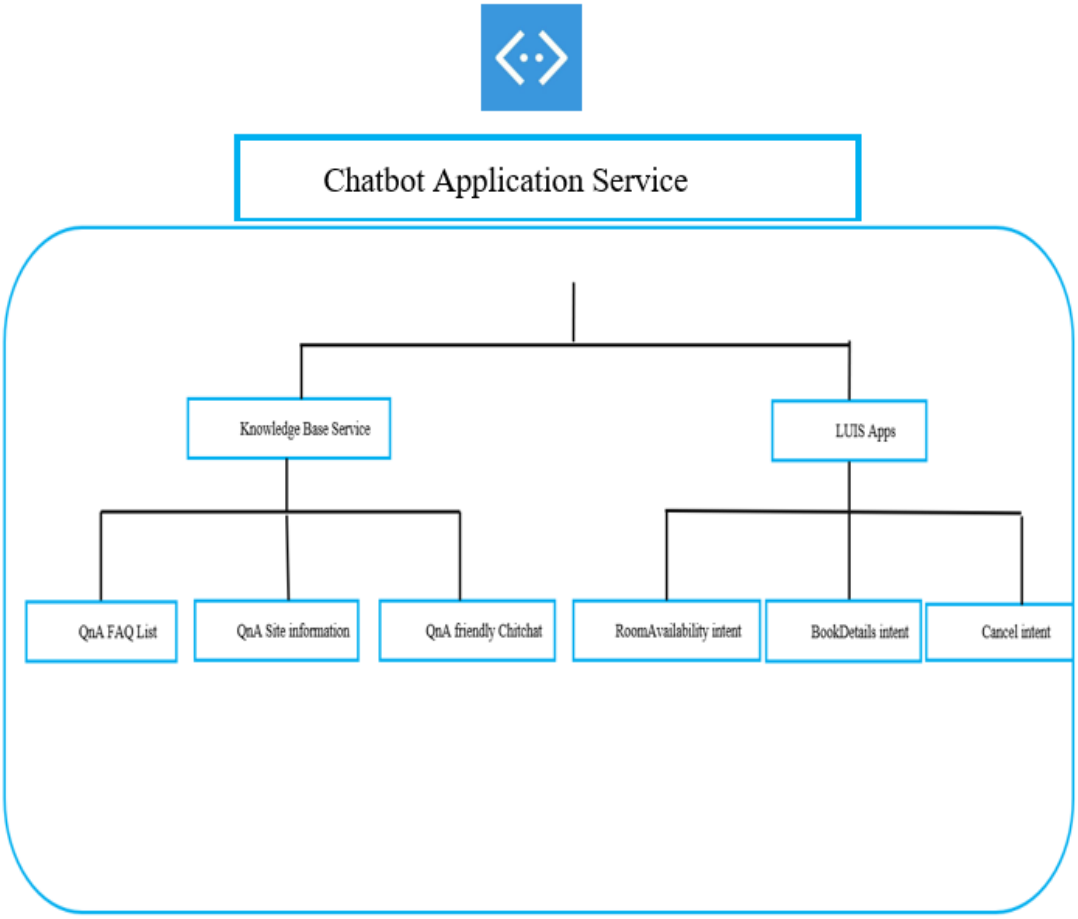


Figure 5.1: *Schema of the final two-layer chatbot architecture.*

As regards the LUIS application, that application is consisted of the three LUIS applications, they have created in section 4.2.1 and later was used to feed the dispatch file. One application stands for the RoomAvailability intent. To that intent are classified utterances like:

- Do you have available rooms?
- Room availability
- Are there any rooms available?
- Which days do you have available rooms?

That intent is used when the customer's input does not include specific details about his desire to book a room. For that reason, when the chatbot classifies the input with RoomAvailability intent it starts a dialog in order to

extract the details that are required to be able to answer about room availability at the hotel business.

In that dialog, chatbot is responsible to ask the following 4 questions:

1. What date do you desire to check in?
2. What date do you desire to check out?
3. How many rooms do you want?
4. What room category do you desire?

After getting the response upon each question the chatbot performs the following checks for each corresponding question.

1. Checks that the input is a valid date, later from the current date.
2. Checks that the input is a valid date, later from the current date and later from the recorded check in date.
3. Checks that the input is integer number, text or numeric type.
4. Checks that the input is a valid room category according to a predefined list with official room category names.

In case that the validity result is negative the chatbot prompts the customer to answer again with a valid input. If the validation is successful then the chatbot returns a message with the valid customer input. After that message, the chatbot asks from the customer to confirm his answer in order to record it and continue with the next question. If the chatbot has collected all the necessary details then, it is ready to inform the customer about the room availability that applies.

The second application stands for BookDetails intent. To that intent are classified utterances like:

- I want three junior suites from 5 May to 9 May.
- Do you have available any junior suite unter den linden?
- I want one room from 05/07/21.
- Can I book two executive double rooms?

When that intent is selected, entities extraction is supported. The entities that chatbot can perceive is:

1. Check in date
2. Check out date
3. Quantity of rooms that customer asks for
4. Room category

For each one of the above entities, the chatbot performs the same validity checks like in the case of RoomAvailability intent. As we see in the above utterances the customer is free to skip re-entering some of the entities that the current LUIS application supports. The chatbot gets and validates the given

input entities and runs the same dialog we described earlier in order to match the remaining entities.

In case that the validity result is negative the chatbot prompts the customer to answer again with a valid input. If the validation is successful then the chatbot returns a message with the valid customer input. After that message, the chatbot asks from the customer to confirm his answer in order to record it and continue with the next question. If the chatbot has collected all the necessary details then, it is ready to inform the customer about the room availability that applies.

Next, we are going to present some representative dialogs of the functionality that the chatbot is capable of offering.

5.2 Representative Dialogs

We are going to cite some of the common cases we expect to meet after the publication of our chatbot to a web site for the hotel business. The same result can be produced with the interaction with Microsoft Bot Emulator.

On Annex A we show the conversation from a feasible dialog between a customer and the hotel business chatbot.

In the start we have some questions about how the customer can get to the hotel, how much time is the journey from the airport to the hotel, how much does a taxi ride costs. After that, the customer is informed about the means of transport that he can use to arrive at the hotel. Also, he asks if there are available rooms. That input phrase according to our LUIS application matches with RoomAvailability intent, so it starts the predefined dialog that we have defined.

In order for the chatbot to be able to answer, it starts a dialog where it collects the necessary information. Having the chatbot collected, the details that are required it offers to the customer an answer regarding room availability. On Annex A is also available the timeline about the current conversation. There we can observe whether the LUIS application or the QnA application is used.

On Annex B we observe that another customer wants to be informed about room availability. Previously, he had accessed photos from the rooms and he probably made his decision. He directly questions the chatbot if there are available three executive double rooms, with check in 06/08 and check out 06/09. The chatbot matches that input with BookDetails intent. That intent classifies all the details, that the customer provided, to the entities that we have declared. The chatbot asks from the customer to confirm each detail individually. That time, the chatbot does not ask the customer to provide input

information but asks to confirm the information that was extracted from the LUIS application.

That decision was selected because the customer may have provided wrong information. Such cases we observe commonly in the details about check in and check out dates. If someone provides 06/08 and 06/09 respectively for check in and checkout dates, he may refer to the set, 06 August - 06 September or 08 June - 09 June. That is happening because there are supported different types of date format around the world [23]. In the LUIS application that we have developed, the customer is free to provide date input in different ways. He can specify the date in the following ways:

- in five days
- in a month
- the first Monday of September
- the day after tomorrow
- 1 September
- 09/01
- 1 September 2021

From the answer that the chatbot returns to the customer we are informed that there is not any executive double room available. The customer if still wants all the rooms to be the same category he can select one of the other available options. Or he can decide to combine different options from the available rooms. Of course, the customer is always free to ask for a different time period if there is availability for the room category, he is interested in.

With the completion of the room availability dialog the customer provides another utterance for the BookDetails intent. For the new utterance (new turn) the chatbot clears its memory (user data) and it starts new dialog to verify each input information.

That is apparent from the query that is executed and the response that is returned from the database.

Whenever the customer decides to quit the dialog that option is supported. With an input that the customer targets to finish the dialog (None intent) the chatbot obeys and ends the dialog. After that the customer can ask anything he may want.

On Annex C we see another conversation between chatbot and customer. We can observe the validation check that chatbot performs. When the customer provides as input “double room” the chatbot informs him that it did not understand preference of specific room category. In Annex D we cite other cases of validation checks that chatbot is able to perform. The validation checks were analyzed in section 5.1. For some cases where the customer did not provide the necessary information the chatbot is making some assumptions. For

example, if the customer neglects to provide the year detail in check in/out date, the chatbot assumes that is referred to the current year.

5.3 Difficulties upon the Scenarios and Limitations

Now, we will talk about issues that were presented during the chatbot development and have affected the final application. The aspect of interacting with a human and more specific with a customer is a complicated issue. The customer during his conversation can talk about different things and with different ways. The input that a customer can provide may have a metaphoric meaning or may be unrelated with the scope of the chatbot. As we understand that state sets limitations for our chatbot application. Even modern AI chatbots in some circumstances can present an inability to handle effectively a conversation.

The chatbot we developed in the current thesis is conditioned. It can handle effectively and succeed in its scope but there are scenarios that it cannot react successfully. The limitations that we analyzed in section 3.3 specify the chatbot's functionality.

We declared earlier that the chatbot will communicate in English language. Other languages can be supported from Microsoft Bot Framework. However, some of its services may be limited and of course require a different development approach [24].

The developed chatbot can run locally but also in a cloud service. We presented the steps we can follow to build our chatbot upon an Azure cloud service that can make it available 24 hours a day, 7 days a week.

Regarding generally the chatbot's scope, the chatbot application is specified to be used for a hotel business. For that reason, it cannot support a general dialog and conversate regarding abstract issues. The developed chatbot can respond to customers' questions provided that its developer has predicted that possible question. For example, the more QnA sets are inserted in the knowledge base the merrier the chatbot would be able to answer effectively a question.

In our case in order to provide a satisfied interaction we incorporated AI capabilities. These capabilities are restricted to the dispatch LUIS application that we have implemented. Totally, we used one knowledge based, consisting of three parts, 3 LUIS applications for the RoomAvailability, BookDetails and None intent. These are based upon 9971, 5, 16 and 6 training utterances, respectively.

Of course, during the chatbot building process we observed cases where it did not respond the way we desired. We proceeded to adjust its architecture. Such a common case is the wrong classification of the QnA set. We can fix that issue from the qnamaker.ai portal, Fig 5.2. Either by editing the example question, either by providing more similar examples or by correcting the result from the test section. After that intervention we can publish our knowledge base and start using the new version.

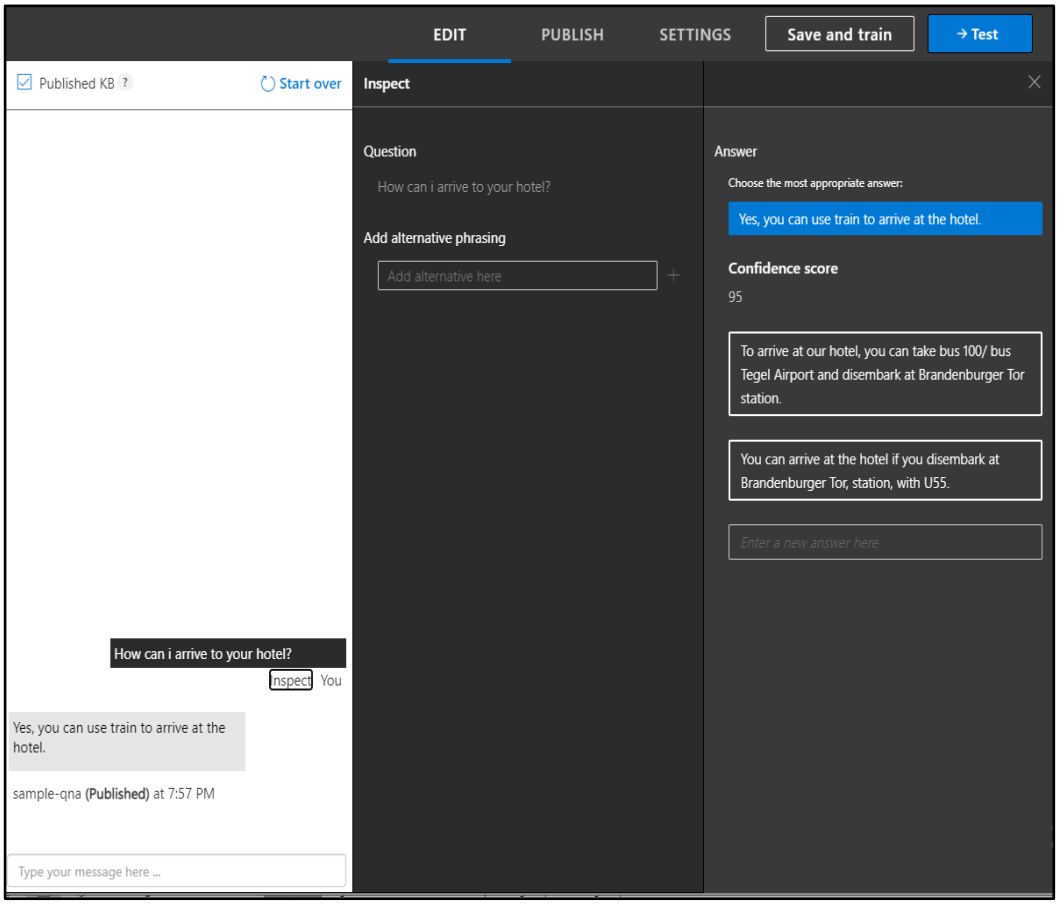


Figure 5.2: *QnA Maker service, test section.*

An analog solution is supported for the case where we have wrong classification of an utterance. We can also edit the example utterances, provide more example utterances or we can correct the wrong application's behavior. In Fig. 5.3 we see an example of how we can inspect for a new utterance and adjust the intent that is appropriate to handle that utterance.

There is also the option to adjust the chatbot application from the source code. In some circumstances, we can invest only to source code to achieve the desired functionality. Such a characteristic case is the managing state. In our chatbot we need to remember certain things through the conversation. That can

be achieved by a meaningful handling of storage layer, state management and state property accessors. These three layers are responsible to support more sophisticated conversations. The degree of skillfulness that a chatbot can offer is closely related with the way we utilize these layers. However, the more functionality we desire the more complicated it tends to be the chatbot. A considerable number of bot's life cycle turns is required in order we end with a version that can support the most of our goals.

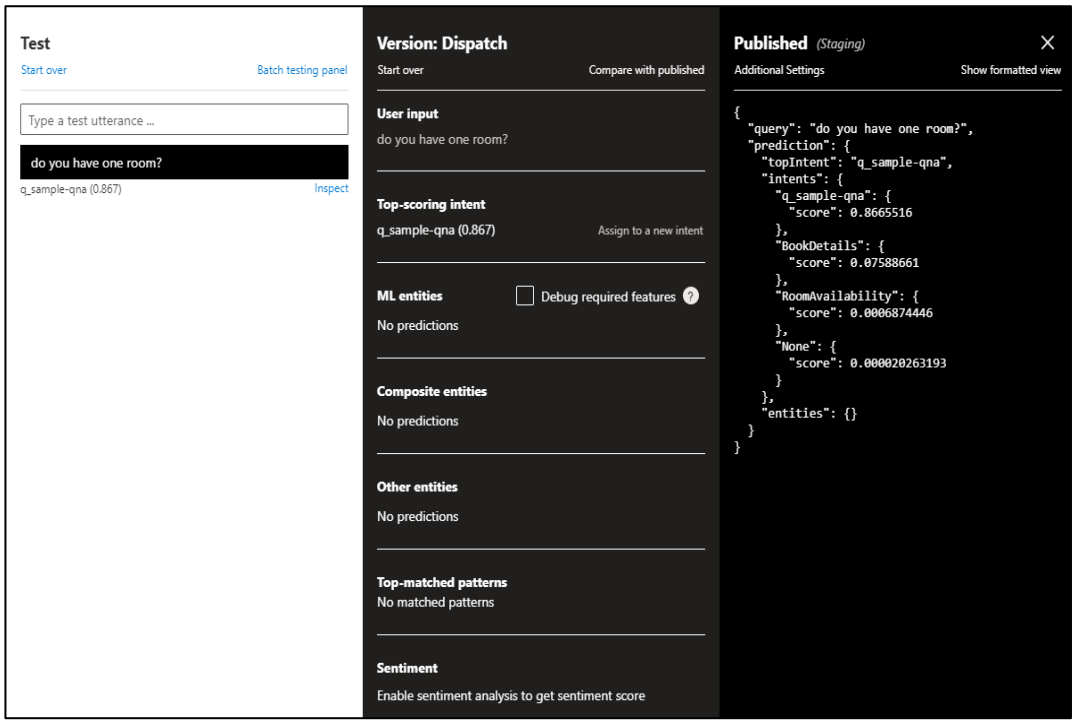


Figure 5.3: *LUIS application, test section.*

All the above are capable of providing us a consideration of the different things that we have to think about during our chatbot development. In any case, the more we test our bot the merrier it gets closely to our target. The final chatbot will be a product of the adjustments and the assumptions we have made during its life cycle.

5.4 Evaluation Process

One of the most important parts in the process of creating a new product or service is the evaluation schema. We do not design products for ourselves, but we create them in order to serve our customer's needs. Before we deliver our final product, it is common to evaluate it in order to avoid usability issues. Depending on the product's context, evaluation methods can be applied

multiple times. For the current work we will explain the evaluation schema that it can be followed to conceive usability problems. There are different approaches regarding the evaluation process. The variants consist upon the type of evaluators (experts – non experts), the time of the evaluation test (early stages, final product), the number of repetitions (once, multiple times). Even when we have finished with the evaluation process it remains the stage of analyzing the feedback and implementing propriate corrections.

Beyond we are going to describe the evaluation schema that we suggest to be followed for the current work. The evaluation process and the adoption of improvements that will handle issues and findings is a time-consuming stage that needs to be performed with attention. We covered the sections of chatbot design and development and now we will briefly deal with the evaluation process. We are not going to follow the whole evaluation schema however; we encourage any person who targets to implement a professional application and targets to offer it in the software industry to consider our suggestions.

5.4.1 Proposed Evaluation Schema

Proposed Method for Evaluation

The framework that we support to be followed is the Heuristic Evaluation of J. Nielsen [25]. It is an approach that matches the needs of software interfaces. It meets great acceptability and it has been analyzed from a lot of others researchers' work. The advantages of that approach have been mentioned from the own creator of that method but also from others researchers [26].

The key parts of that method are the following:

- Evaluators Selection
- Definition of Evaluators Tasks
- How Evaluators Reports
- Data Analysis

Evaluators Selection

The person or team that is responsible to care about the whole project of the chatbot needs to take care of the overall Evaluation Schema that will be followed. Regarding the evaluators they should be people that are representative participants for the execution of the usability testing. The

candidates are needed to be potential visitors of the hotel. Accordingly with the characteristics and the proportion of the actual visitors from the past it can be specified who candidate evaluators are appropriate to take part in the survey. The goal is only prospective customers to evaluate the chatbot application. Evaluators that have experience with the booking process and are familiar with web technologies. It is supposed that we intend to satisfy that kind of customers and not the expert ones in User Interaction (UI)/ User Experience (UX). These experts may be involved as a final refinement evaluation schema.

After analyzing the characteristics (age, education, experience with specific message platforms etc.) and having selected the propriate one evaluators that satisfies them we proceed with the definition of the number of people that should be involved. In Human Computer Interaction (HCI) there has not been determined the exact number of participants that is required to perform a successful usability testing. According to J. Nielsen and T. Landauer [27] for small project 7 persons seem to be optimal while for a medium to large project 15 persons are considered to be optimal. However, the majority of researchers agree with the statement that with a participants' number greater than 10 persons up to 80% of usability problems will be effectively tracked.

In each case the target is to find the majority of problems. The evaluation process is dynamic. According to our targets and the evaluation results we can perform adjustments and repeat some steps. Also, the number of evaluators is closely related with the available budget for the usability testing. For the reason that the usability testing requires from the participants time, it is common to reward the participants with money compensation. That is a parameter that should be considered in the evaluators' recruitment process.

Definition of Evaluators Tasks

The team that is responsible to organize the evaluation process should clarify the tasks that evaluators are needed to complete. The evaluators are considered to be personas that are engaged in concrete scenarios. These scenarios have been chosen as cases that would examine effectively our application efficiency and would offer great perception of usability and functionality issues. The scenarios and the documentation of the engagement process is really important. A well-defined process will reduce misconceptions and will offer focus on crucial key points that we try to detect.

How Evaluators Reports

Having provided all necessary directions to our participants we proceed to the way that our participants will be engaged. Each participant, individually and independently, is going to fill a questionnaire based on J. Nielsen work [25], Table 5.1. Based on that questionnaire an individual report is being composed and includes all the findings that argue for the violation of a heuristic.

For each violation that has been reported the evaluator assigns a severity score number, Table 5.3, to each finding and at the same time he proposes a solution for the issue.

Having completed the filling of the report, the evaluators are invited to discuss together the findings and provide a final report that gathers and sorts the insights of that evaluation process. The final report usually consists of a table that sums the findings along with the corresponding severity, a table like the Table 5.2 may be used, graphical representations of the same data, and a section with proposed solutions for each finding. Additionally, screenshots or video are commonly included for supporting the findings.

Data Analysis

Having completed the stage of collecting the evaluators data we proceed with the analysis of that information. At that stage we try to make good use of the criticism about our application and follow adjustments that will improve potential weaknesses. However, at that final step we should decide which findings can help us to reconsider our chatbot implementation and which are not closely related with our goals and the requirements that we have defined in the early steps of the bot's life cycle.

5.5 Questionnaire Template

In that section we are going to define the questionnaire that we propose to be used for the evaluation process. As we mentioned earlier, we follow the Heuristic Evaluation Framework that J. Nielsen [25], first analyzed and today that framework still is actively supported by Nielsen Norman Group [28]. We proceed to an adoption of the questionnaire to better fit our special application's needs.

Table 5.1: *Adapted questionnaire of Jakob Nielsen's 10 general principles for interaction design evaluation.*

Heuristic No.		Description
1	**Visibility of system status**	The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time. Present feedback to the user as quickly as possible (ideally, immediately). Build trust through open and continuous communication.
2	**Match between system and the real world**	The design should speak the users' language. Use words, phrases, and concepts familiar to the user, rather than internal jargon. Follow real-world conventions, making information appear in a natural and logical order.
3	**User control and freedom**	Show a clear way to exit the current interaction, like a Cancel button.
4	**Consistency and standards**	Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions. Follow established industry conventions (external consistency). Follow up messages to reduce ambiguity.
5	**Error prevention**	Good error messages are important, but the best designs carefully prevent problems from occurring in the first place. Either eliminate error-prone conditions, or check for them and present users with a confirmation option before they commit to the action. Provide guidance and constraints.
7	**Flexibility and efficiency of use**	Shortcuts — hidden from novice users — may speed up the interaction for the expert user such that the design can cater to both inexperienced and experienced users. Allow users to tailor frequent actions. Allow for customization, so users can make selections about how they want the product to work. Provide accelerators like keyboard shortcuts and touch gestures.
9	**Help users recognize, diagnose, and recover from errors**	Error messages should be expressed in plain language (no error codes), precisely indicate the problem, and constructively suggest a solution. Offer users a solution, like a shortcut that can solve the error immediately. Ability to understand misspelling words.

Table 5.2: *Table for illustration of evaluation results.*

Heuristic No.	Number of Problems	Problem Severity Score					Average Severity
		1	2	3	4	5	
1							
2							
3							
4							
5							
7							
9							
Total							

The following 0 to 4 rating scale can be used to rate the severity of usability problems:

Table 5.3: *Definition of severity score scale.*

Severity score	Meaning
1	I don't agree that this is a usability problem at all
2	Cosmetic problem only: need not be fixed unless extra time is available on project
3	Minor usability problem: fixing this should be given low priority
4	Major usability problem: important to fix, so should be given high priority
5	Usability catastrophe: imperative to fix this before product can be released

Each participant to the evaluation of our application, firstly, needs to complete the questionnaire and secondly together with the others participants is called to provide a final report that sums up the observations and findings of each participant individually. The raw data are these which will give us a better understanding of our application's weaknesses. It is a useful, designer and evaluator to have separate roles in order a clear and impartial prospect to be achieved. That stage of evaluation will feed back the bot's life cycle and that procedure can be repeated as many times as we believe that is necessary to reach the desired level of functionality. However, in each case we should not neglect that we are designing an application that aims to satisfy the

requirements and the goals that we have set, and not an application that handles every situation that a potential user may imagine. In the next final section, we sum up the current thesis work and end up with some recommendations for future improvements.

6 Conclusion

6.1 Summary

The current thesis was an assignment about using chatbot in the interaction of a platform user. We tried to develop a chatbot that would be able to answer user questions instead of requiring the user to browse a site and fill fields. We noticed that nowadays AI has achieved great success in the fields of NLU. There are approaches and frameworks that are evaluated as capable to support realistic conversations.

In our case, the choice we ended up to use was Microsoft Bot Framework. We went beyond the core bot that it is commonly used and we designed/ developed CI that is capable of helping a customer that asks details about the hotel or hotel's room availability. We implemented a dispatch LUIS application that consisted of a knowledge base and a LUIS application, responsible to handle RoomAvailability and BookDetails intents. In order for our chatbot to have access at the room management system we connected it with a SQL database. When a customer asks for room availability the chatbot can execute SQL queries and can get a real time overview of the availability state.

During the thesis we learned about CI terminology and NLU context. With the use of Microsoft Bot Framework, we understand the architecture that a bot usually follows and in which layers it uses each supportive service. Additionally, we tried to put ourselves in customers' seats in order to understand what they may want to ask the chatbot for but also by which way. In that step it was necessary to specify which circumstances (definition of limitations) the chatbot will be able to handle.

Our thesis is an attempt to support the customer service in the case of a hotel business. We set goals and limitations because the development of a chatbot that can behave like a human was out of our scope. For that reason, we were testing/ evaluating our chatbot in every step of its life cycle and we have to proceed to crucial adjustments when we were considering it to be necessary.

6.2 Future Work

As AI extends its potentiality and its application fields further steps can be made in the current thesis. Our work was the beginning and below we notice supplement issues that can be further investigated.

Multi-language Support

Since, we developed a chatbot that is going to be used for a hotel business the aim to support languages from different countries seems in order. That extension is going to expand the number of customers that will be able to conversate with the chatbot.

More Flexibility

Flexibility is a common target for each software product. That of course applies also for the case of chatbot. For example, the customer can ask for room availability and the chatbot can reply that there is no availability in the specific time period for its selected room category. It would be useful in that case the chatbot to go beyond providing alternatives of different room categories, the same time period, and to provide the user information of when the room category he desires is again available.

More flexibility is going to improve chatbot functionality and offer it a more elegant behavior. Of course, that would be feasible after engaging the managing state of Microsoft Bot Framework. New services would be necessary to be developed leading in turn to new storage requirements.

Evaluation

Every CI needs to be evaluated. Some times when the product is targeted to be used from customers a more sophisticated framework is required. The more heterogeneous the final users are the merrier complicated is for the developer to predict their expectations. For that reason, an evaluation framework could provide useful insights for the implemented interactivity.

Extension to Support Voice Conversations

Chatbots have a longer history compared to voicebots. However, nowadays there is an increasing interest in the attempt to engage voice for interaction with AI software. That is a really demanding task but some already implemented attempts have proved that is feasible and really promising for increasing the modalities that AI can support. That issue was intentionally left last as it includes high potentiality and it seems to be the future of sophisticated CI.

BLANK PAGE

References

[1] (Chatbot | Definition of chatbot in English by Lexico Dictionaries, 2019)

[2] T. Gersil and I. Hilal, “The Role of Conversational Interfaces in the Future of Digital and Technology”, Degree Project, KTH Royal Institute of Technology, School of EECS, Sweden 2020

[3] J. Winkler, “Conception and Realization of a Chatbot-System to support Psychological and Medical Procedures”, ULM University, Master Thesis, 2019

[4] https://www.ibm.com/cloud/blog/importance-context-chatbot-conversation (accessed 22/02/2021)

[5] R. Khan and A. Das, “Build Better Chatbots, A Complete Guide to Getting Started with Chatbots”, Springer Science+Business Media New York, 2018, ISBN-13 (electronic): 978-1-4842-3111-1

[6] E. Adamopoulou and L. Moussiades. “Chatbots: History, technology, and applications”, Elsevier Ltd, 9 November 2020

[7] https://www.spaceo.ca/top-ai-chatbot-frameworks/ (accessed 22/02/2021)

[8] https://discover.bot/bot-talk/guide-to-bot-buiding-frameworks/ (accessed 23/02/2021)

[9] https://docs.microsoft.com/en-us/azure/bot-service/bot-service-overview-introduction?view=azure-bot-service-4.0 (accessed 19/02/21)

[10] https://docs.microsoft.com/en-us/azure/cognitive-services/qnamaker/overview/overview (accessed 05/01/2021)

[11] https://www.revechat.com/blog/customer-service-chatbots/ accessed 02/03/2021)

[12] https://azure.microsoft.com/pt-br/blog/microsoft-conversational-ai-tools-enable-developers-to-build-connect-and-manage-intelligent-bots/ (accessed 3/3/2021)

[13] https://ec.europa.eu/eurostat/statistics-explained/index.php/Foreign_language_learning_statistics (accessed 27/02/2021)

[14] https://www.statista.com/statistics/266808/the-most-spoken-languages-worldwide/ (accessed 27/02/2021)

[15] https://github.com/microsoft/BotBuilder-Samples (accessed 01/03/2021)

[16] https://portal.azure.com/ (accessed 04/03/2021)

[17] https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/manage-resource-groups-portal (accessed 04/03/2021)

[18] https://docs.microsoft.com/en-us/azure/bot-service/bot-builder-howto-bf-cli-deploy-luis?view=azure-bot-service-4.0 (accessed 07/03/2021)

[19] https://docs.microsoft.com/en-us/azure/bot-service/bot-builder-howto-qna?view=azure-bot-service-4.0&tabs=js (accessed 07/03/2021)

[20] https://docs.microsoft.com/en-us/azure/bot-service/bot-builder-tutorial-dispatch?view=azure-bot-service-4.0&tabs=js (accessed 07/03/2021)

[21] https://docs.microsoft.com/en-us/azure/bot-service/bot-service-debug-emulator?view=azure-bot-service-4.0&tabs=csharp (accessed 07/03/2021)

[22] https://www.kempinski.com/en/hotels/about-us (accessed 12/03/2021)

[23] https://en.wikipedia.org/wiki/Date_format_by_country (accessed 13/3/2021)

[24] https://docs.microsoft.com/en-us/composer/how-to-use-multiple-language (accessed 14/03/2021)

[25] Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. Proc. ACM CHI'94 Conf. (Boston, MA, April 24-28), 152-158.

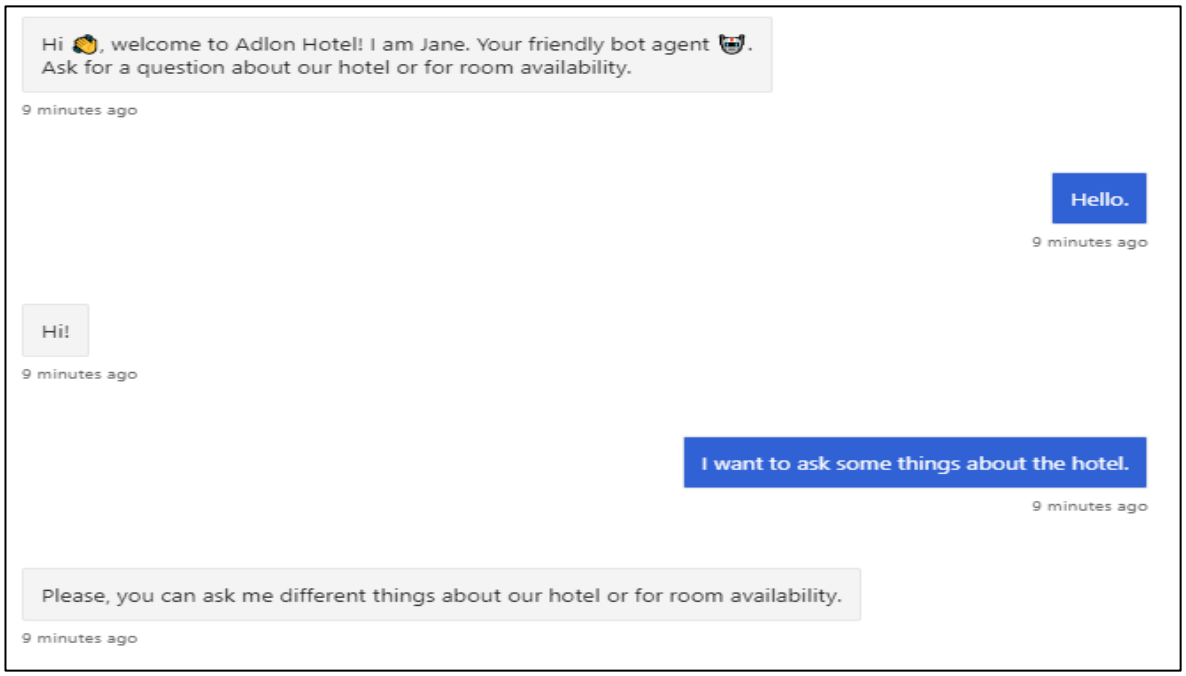
[26] https://www.toptal.com/designers/usability/usability-analysis-how-to-run-a-heuristic-evaluation (accessed 21/04/2021)

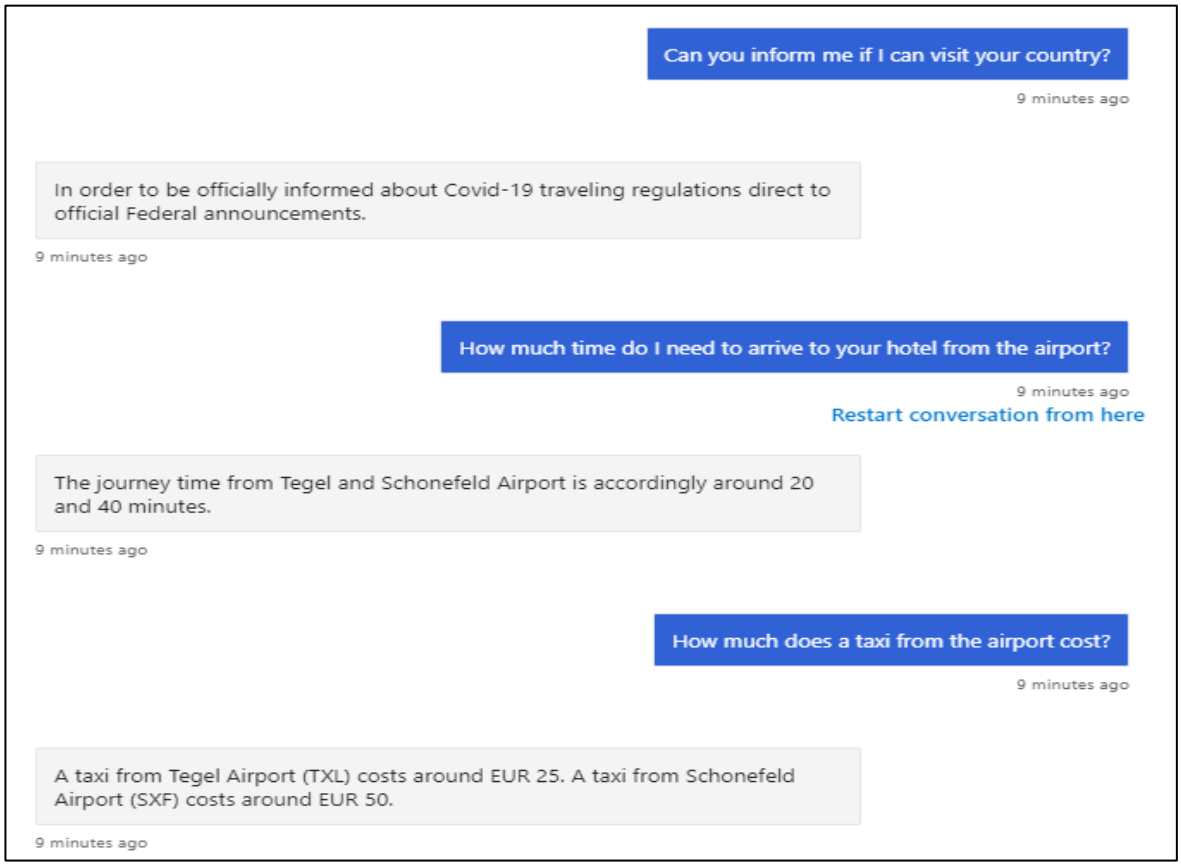
[27] J. Nielsen and T. K. Landauer, “A mathematical model of the finding of usability problems”, Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, May 1993

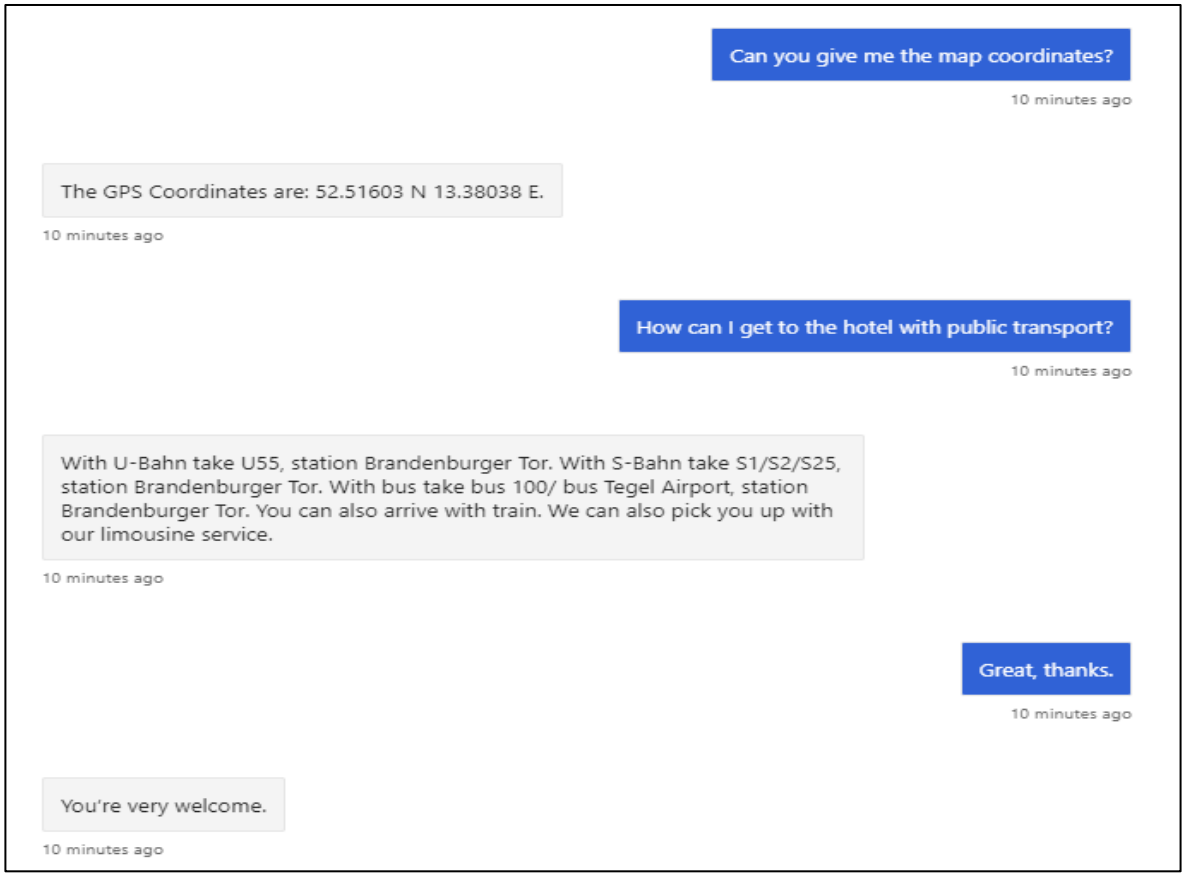
[28] https://www.nngroup.com/ (accessed 22/04/2021)

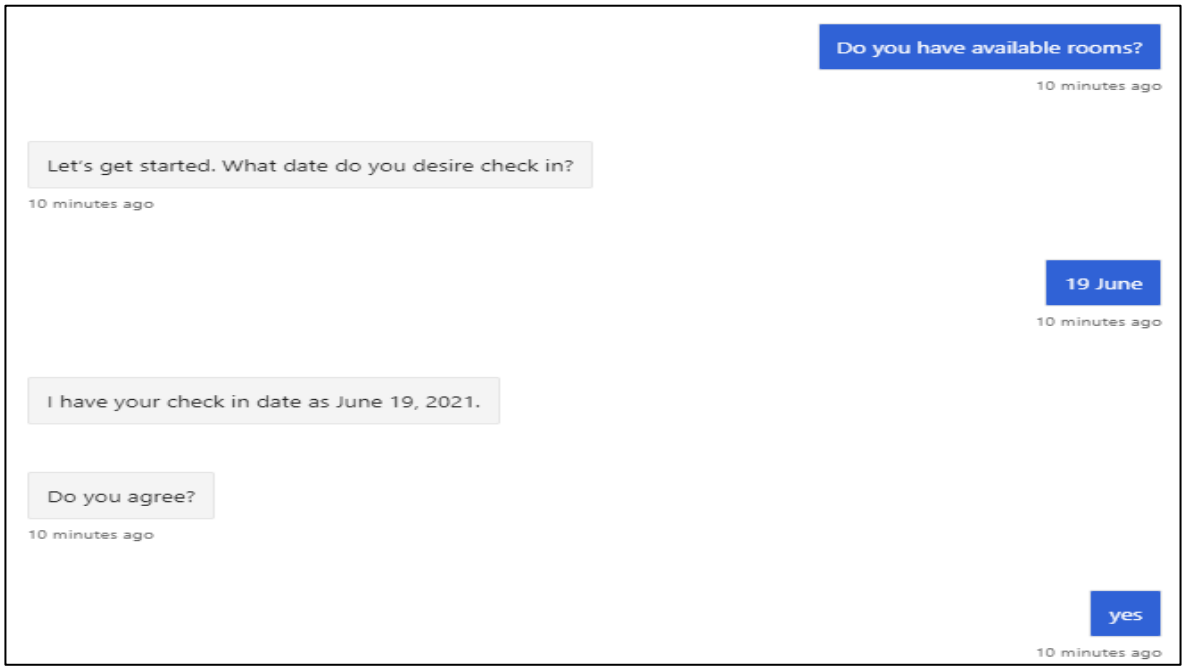
Annexes

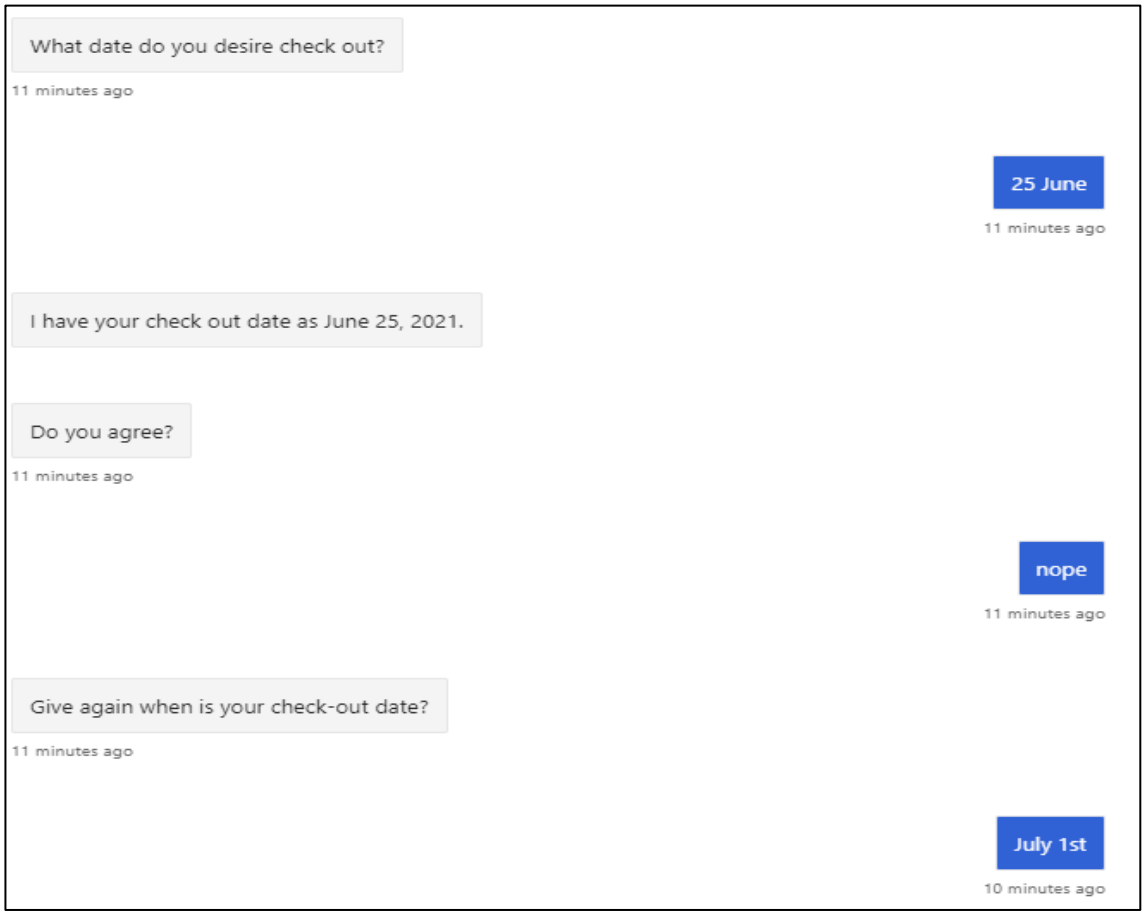
Annex A



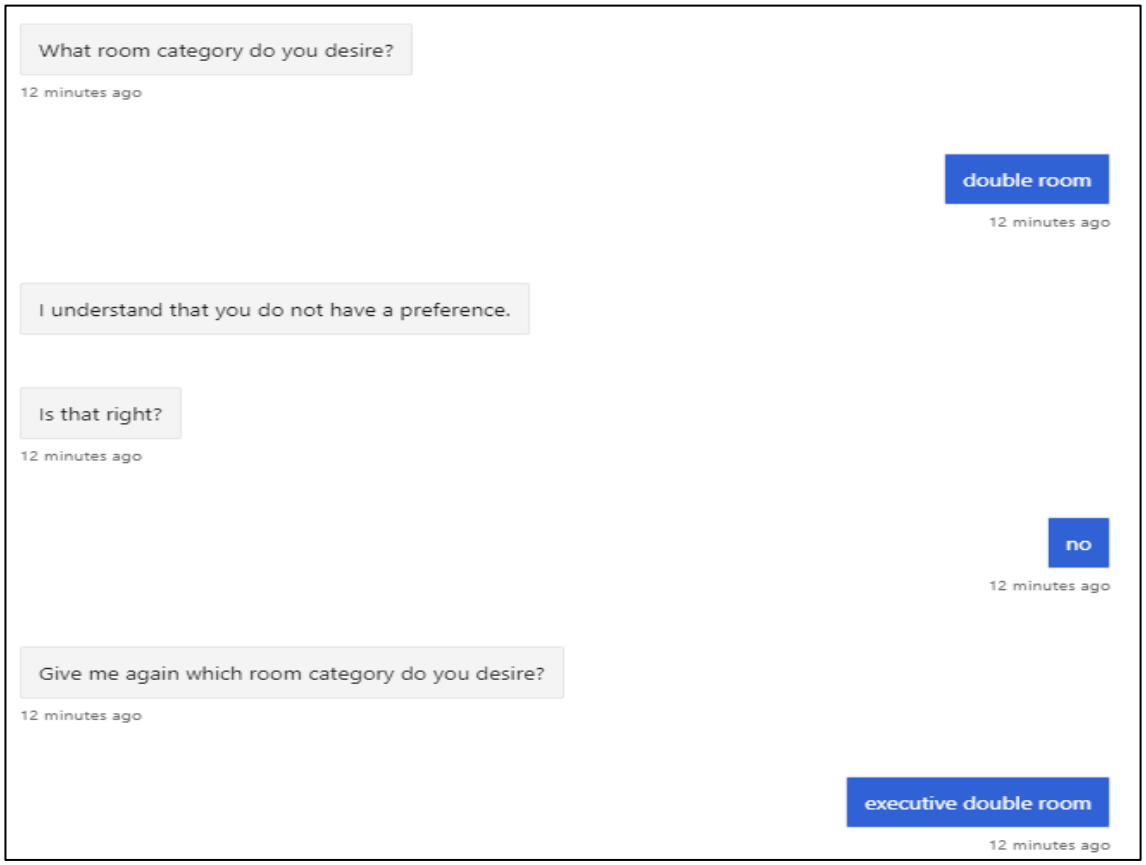


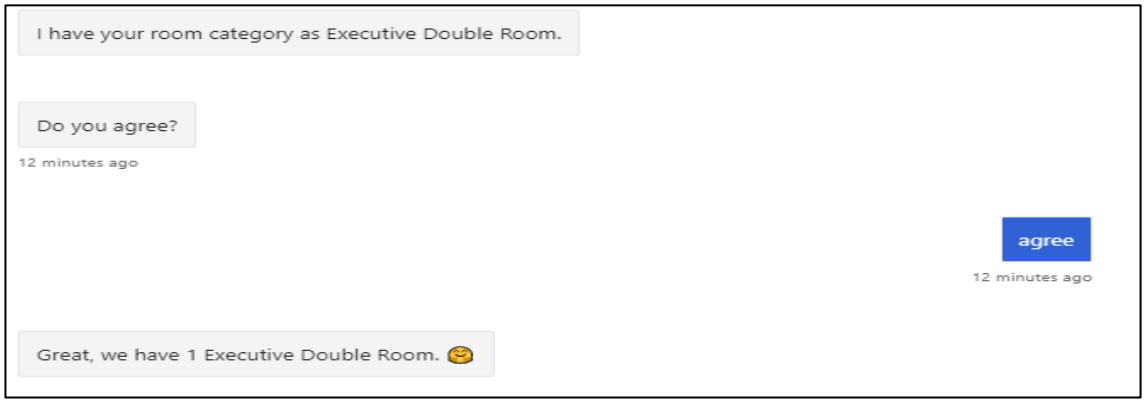

Can you give me the map coordinates?
10 minutes ago
The GPS Coordinates are: 52.51603 N 13.38038 E.
10 minutes ago
How can I get to the hotel with public transport?
10 minutes ago
With U-Bahn take U55, station Brandenburger Tor. With S-Bahn take S1/S2/S25, station Brandenburger Tor. With bus take bus 100/ bus Tegel Airport, station Brandenburger Tor. You can also arrive with train. We can also pick you up with our limousine service.
10 minutes ago
Great, thanks.
10 minutes ago
You're very welcome.
10 minutes ago

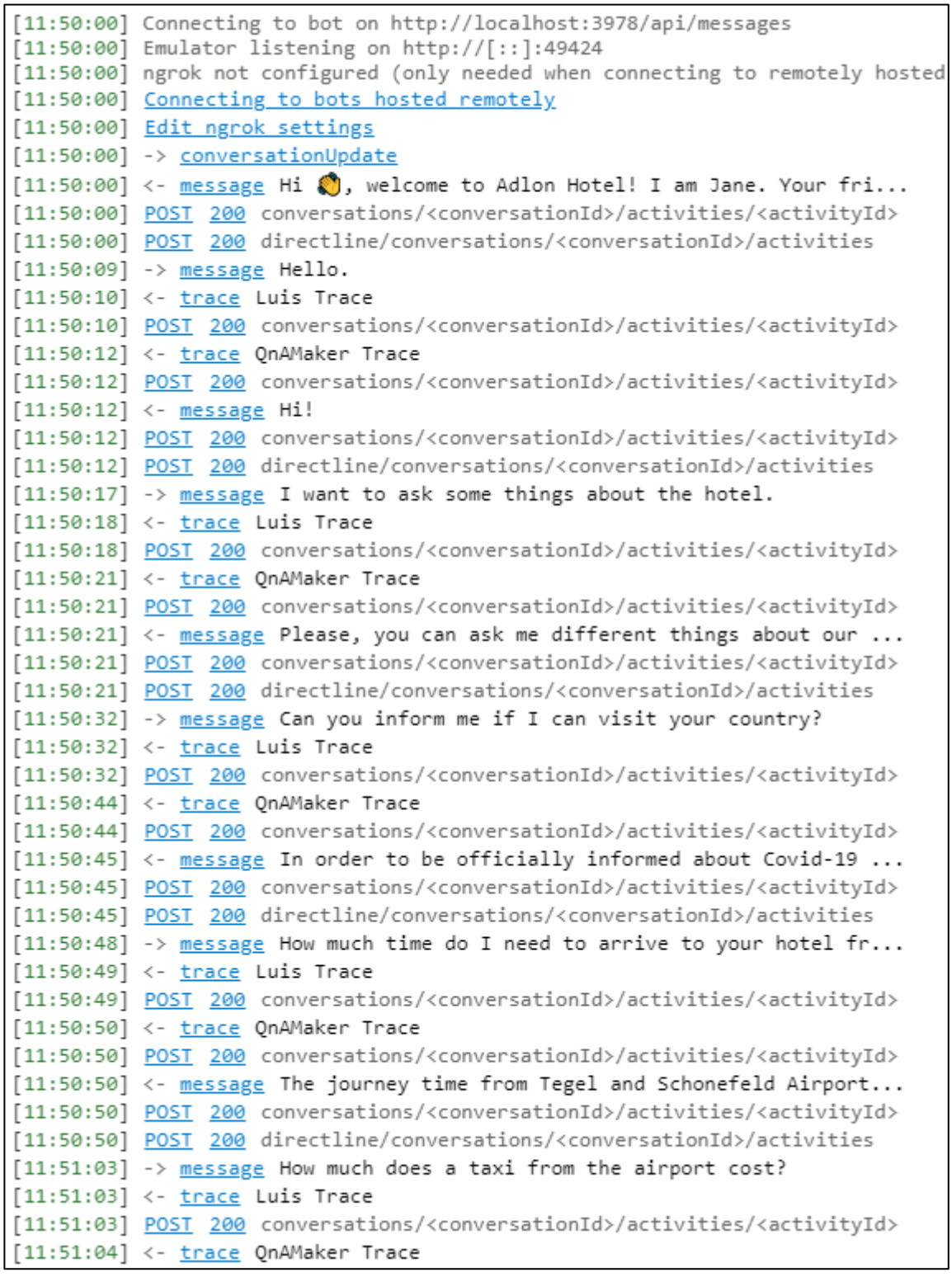

Do you have available rooms?
10 minutes ago
Let's get started. What date do you desire check in?
10 minutes ago
19 June
10 minutes ago
I have your check in date as June 19, 2021.
Do you agree?
10 minutes ago
yes
10 minutes ago


What date do you desire check out?
11 minutes ago
25 June
11 minutes ago
I have your check out date as June 25, 2021.
Do you agree?
11 minutes ago
nope
11 minutes ago
Give again when is your check-out date?
11 minutes ago
July 1st
10 minutes ago


I have your check out date as July 1, 2021.
Do you agree?
12 minutes ago
of course
12 minutes ago
How many rooms do you want?
12 minutes ago
1
11 minutes ago
I have your number of rooms: 1.
Do you agree?
11 minutes ago
yes
11 minutes ago


What room category do you desire?
12 minutes ago
double room
12 minutes ago
I understand that you do not have a preference.
Is that right?
12 minutes ago
no
12 minutes ago
Give me again which room category do you desire?
12 minutes ago
executive double room
12 minutes ago


I have your room category as Executive Double Room.
Do you agree?
12 minutes ago
agree
12 minutes ago
Great, we have 1 Executive Double Room.

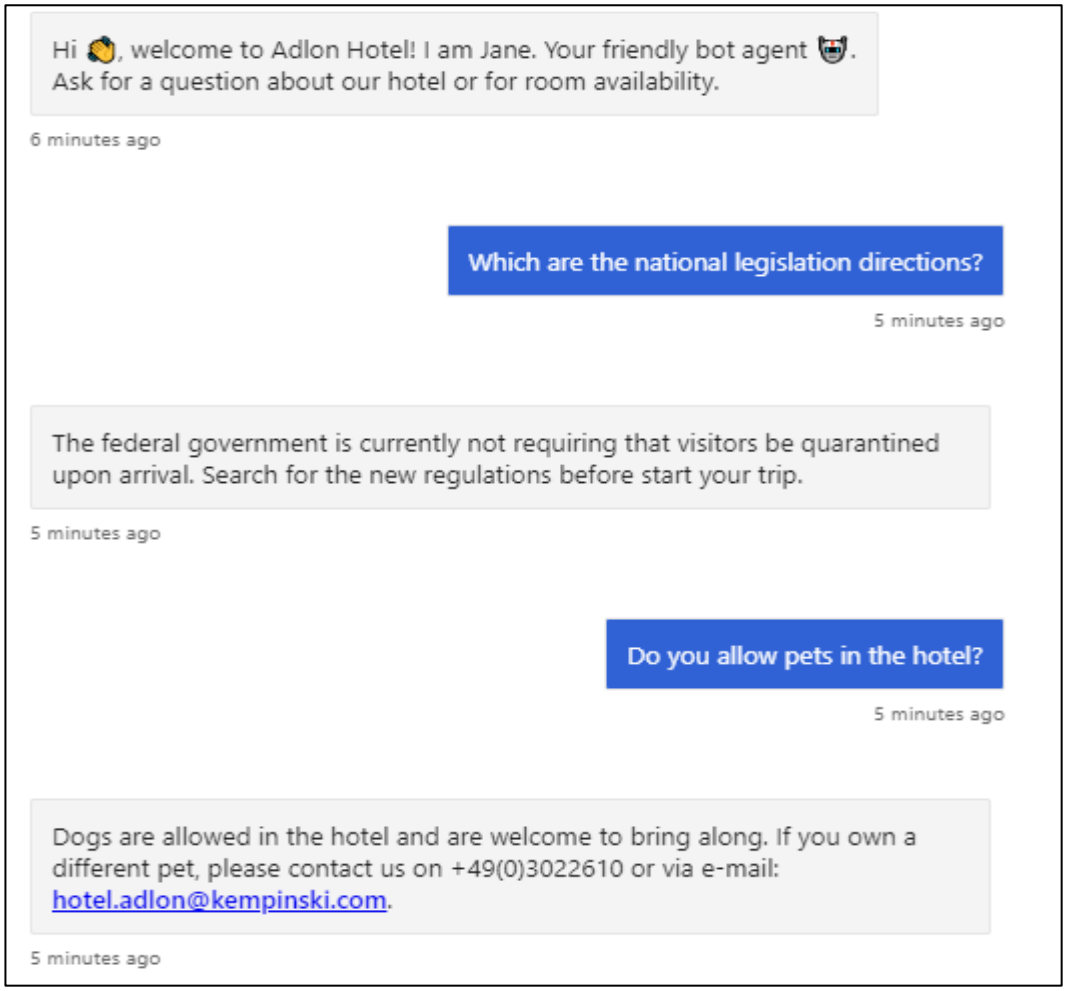

[11:50:00] Connecting to bot on http://localhost:3978/api/messages
[11:50:00] Emulator listening on http://[::]:49424
[11:50:00] ngrok not configured (only needed when connecting to remotely hosted
[11:50:00] Connecting to bots hosted remotely
[11:50:00] Edit ngrok settings
[11:50:00] -> conversationUpdate
[11:50:00] <- message Hi 👋, welcome to Adlon Hotel! I am Jane. Your fri...
[11:50:00] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:00] POST 200 directline/conversations/<conversationId>/activities
[11:50:09] -> message Hello.
[11:50:10] <- trace Luis Trace
[11:50:10] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:12] <- trace QnAMaker Trace
[11:50:12] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:12] <- message Hi!
[11:50:12] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:12] POST 200 directline/conversations/<conversationId>/activities
[11:50:17] -> message I want to ask some things about the hotel.
[11:50:18] <- trace Luis Trace
[11:50:18] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:21] <- trace QnAMaker Trace
[11:50:21] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:21] <- message Please, you can ask me different things about our ...
[11:50:21] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:21] POST 200 directline/conversations/<conversationId>/activities
[11:50:32] -> message Can you inform me if I can visit your country?
[11:50:32] <- trace Luis Trace
[11:50:32] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:44] <- trace QnAMaker Trace
[11:50:44] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:45] <- message In order to be officially informed about Covid-19 ...
[11:50:45] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:45] POST 200 directline/conversations/<conversationId>/activities
[11:50:48] -> message How much time do I need to arrive to your hotel fr...
[11:50:49] <- trace Luis Trace
[11:50:49] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:50] <- trace QnAMaker Trace
[11:50:50] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:50] <- message The journey time from Tegel and Schonefeld Airport...
[11:50:50] POST 200 conversations/<conversationId>/activities/<activityId>
[11:50:50] POST 200 directline/conversations/<conversationId>/activities
[11:51:03] -> message How much does a taxi from the airport cost?
[11:51:03] <- trace Luis Trace
[11:51:03] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:04] <- trace QnAMaker Trace

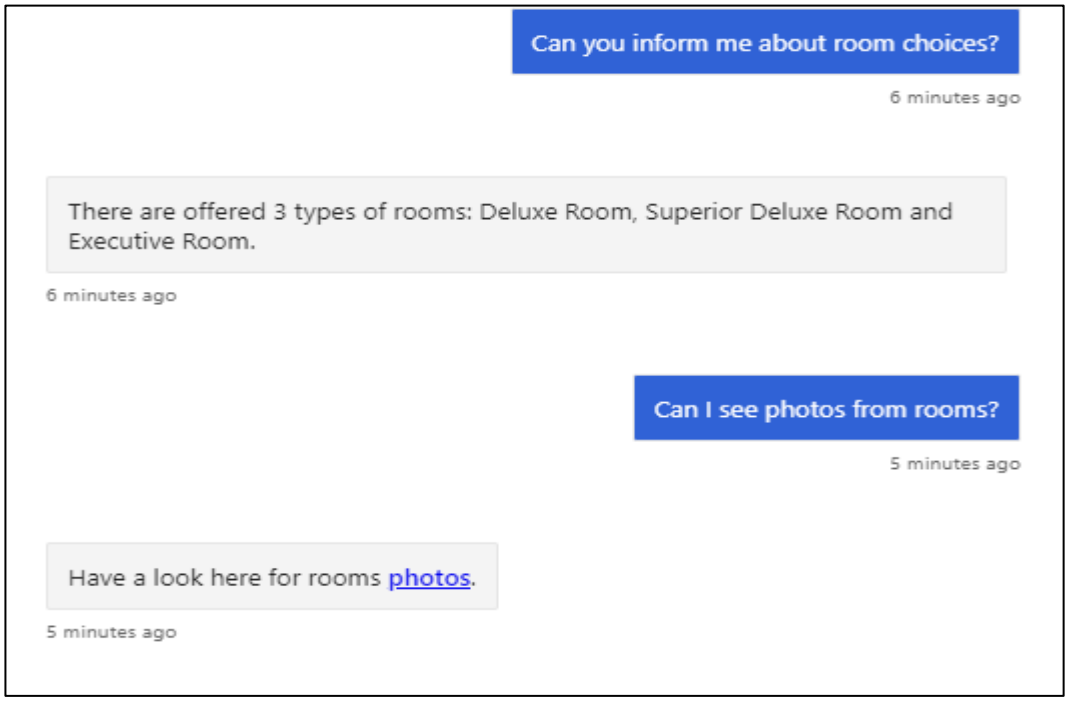
```
[11:51:04] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:04] <- message A taxi from Tegel Airport (TXL) costs around EUR 2...
[11:51:04] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:04] POST 200 directline/conversations/<conversationId>/activities
[11:51:11] -> message Can you give me the map coordinates?
[11:51:12] <- trace Luis Trace
[11:51:12] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:13] <- trace QnAMaker Trace
[11:51:13] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:13] <- message The GPS Coordinates are: 52.51603 N 13.38038 E.
[11:51:13] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:13] POST 200 directline/conversations/<conversationId>/activities
[11:51:21] -> message How can I get to the hotel with public transport?
[11:51:21] <- trace Luis Trace
[11:51:21] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:22] <- trace QnAMaker Trace
[11:51:22] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:22] <- message With U-Bahn take U55, station Brandenburger Tor. W...
[11:51:22] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:22] POST 200 directline/conversations/<conversationId>/activities
[11:51:30] -> message Great, thanks.
[11:51:30] <- trace Luis Trace
[11:51:30] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:31] <- trace QnAMaker Trace
[11:51:31] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:31] <- message You're very welcome.
[11:51:31] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:31] POST 200 directline/conversations/<conversationId>/activities
[11:51:39] -> message Do you have available rooms?
[11:51:39] <- trace Luis Trace
[11:51:39] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:39] <- message Let's get started. What date do you desire check i...
[11:51:39] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:39] POST 200 directline/conversations/<conversationId>/activities
[11:51:48] -> message 19 June
[11:51:49] <- message I have your check in date as June 19, 2021.
[11:51:49] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:49] <- message Do you agree?
[11:51:49] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:49] POST 200 directline/conversations/<conversationId>/activities
[11:51:54] -> message yes
[11:51:54] <- message What date do you desire check out?
[11:51:54] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:54] POST 200 directline/conversations/<conversationId>/activities
```

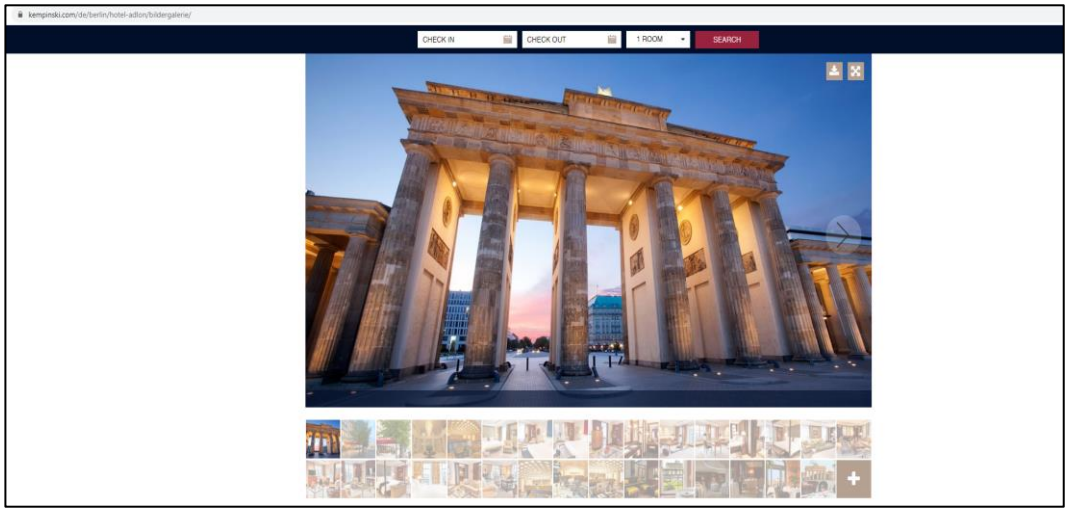
```
[11:51:54] -> message yes
[11:51:54] <- message What date do you desire check out?
[11:51:54] POST 200 conversations/<conversationId>/activities/<activityId>
[11:51:54] POST 200 directline/conversations/<conversationId>/activities
[11:52:05] -> message 25 June
[11:52:05] <- message I have your check out date as June 25, 2021.
[11:52:05] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:05] <- message Do you agree?
[11:52:05] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:05] POST 200 directline/conversations/<conversationId>/activities
[11:52:15] -> message nope
[11:52:15] <- message Give again when is your check-out date?
[11:52:15] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:15] POST 200 directline/conversations/<conversationId>/activities
[11:52:35] -> message July 1st
[11:52:35] <- message I have your check out date as July 1, 2021.
[11:52:35] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:35] <- message Do you agree?
[11:52:35] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:35] POST 200 directline/conversations/<conversationId>/activities
[11:52:52] -> message of course
[11:52:52] <- message How many rooms do you want?
[11:52:52] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:52] POST 200 directline/conversations/<conversationId>/activities
[11:52:56] -> message 1
[11:52:56] <- message I have your number of rooms: 1.
[11:52:56] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:56] <- message Do you agree?
[11:52:56] POST 200 conversations/<conversationId>/activities/<activityId>
[11:52:56] POST 200 directline/conversations/<conversationId>/activities
[11:53:02] -> message yes
[11:53:02] <- message What room category do you desire?
[11:53:02] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:02] POST 200 directline/conversations/<conversationId>/activities
[11:53:14] -> message double room
[11:53:14] <- message I understand that you do not have a preference.
[11:53:14] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:14] <- message Is that right?
[11:53:14] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:14] POST 200 directline/conversations/<conversationId>/activities
[11:53:21] -> message no
[11:53:21] <- message Give me again which room category do you desire?
```

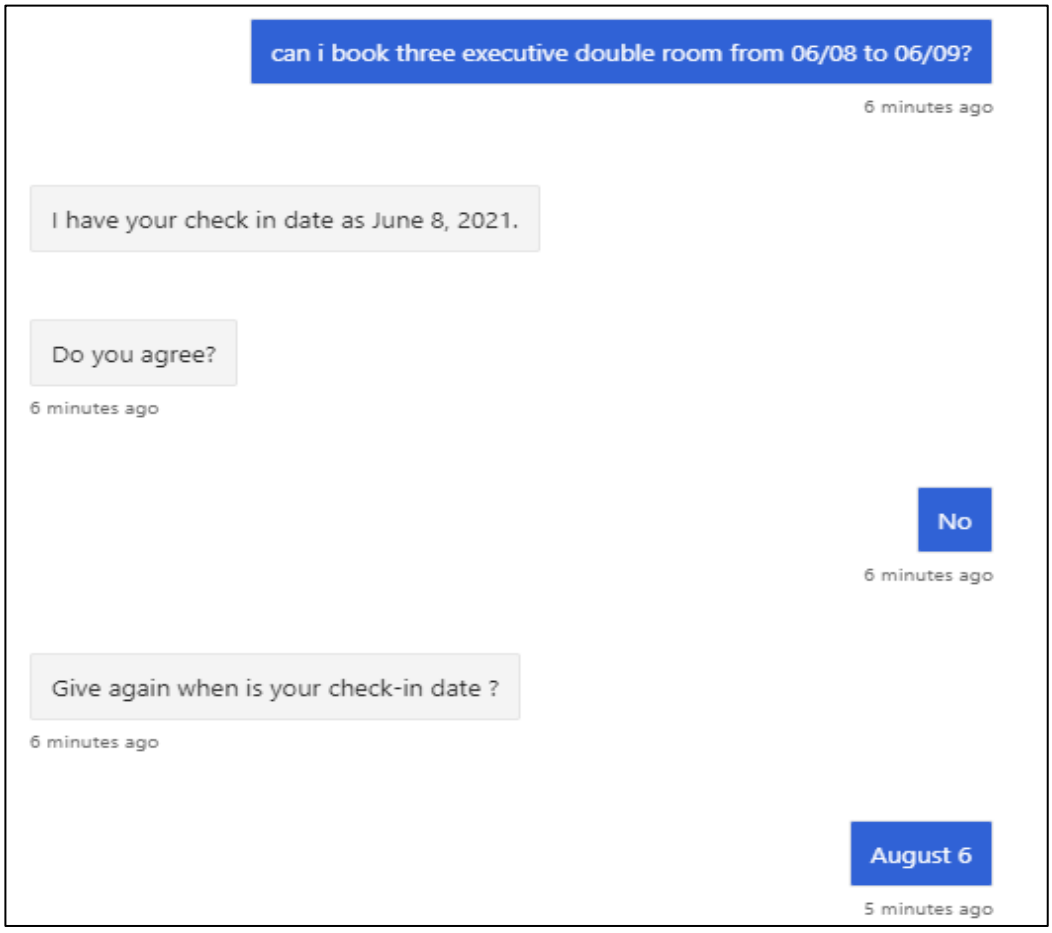
```
[11:53:21] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:21] POST 200 directline/conversations/<conversationId>/activities
[11:53:39] -> message executive double room
[11:53:39] <- message I have your room category as Executive Double Room...
[11:53:39] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:39] <- message Do you agree?
[11:53:39] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:39] POST 200 directline/conversations/<conversationId>/activities
[11:53:48] -> message agree
[11:53:48] <- message Great, we have 1 Executive Double Room.
[11:53:48] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:48] <- message Completed, ask something else.
[11:53:48] POST 200 conversations/<conversationId>/activities/<activityId>
[11:53:48] POST 200 directline/conversations/<conversationId>/activities
```

Annex B

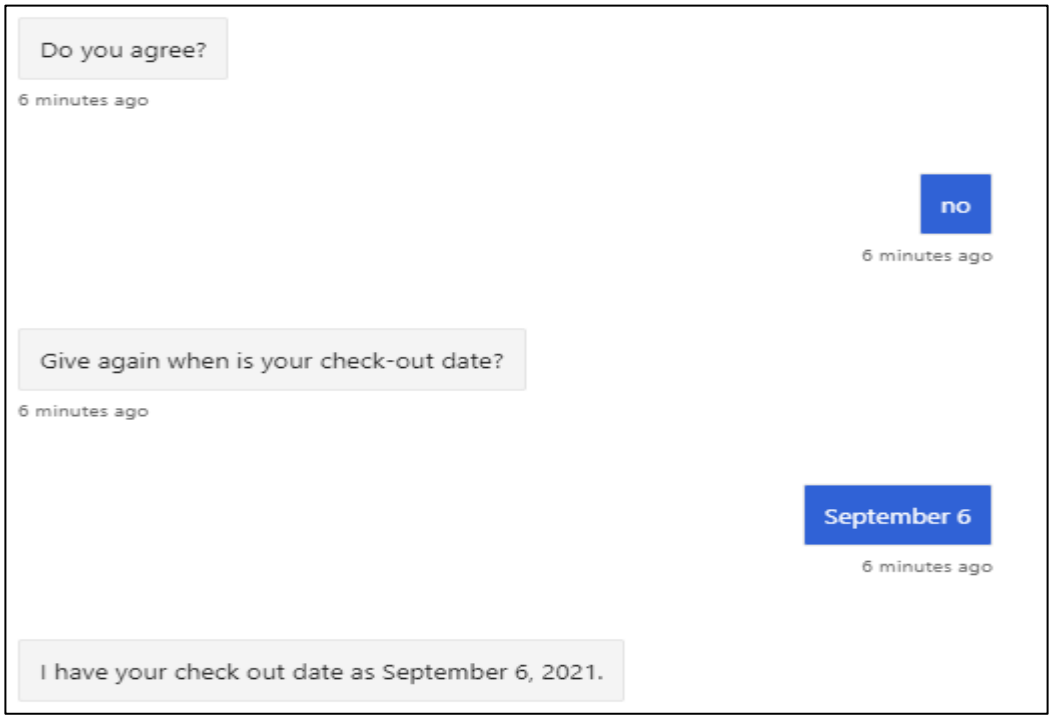


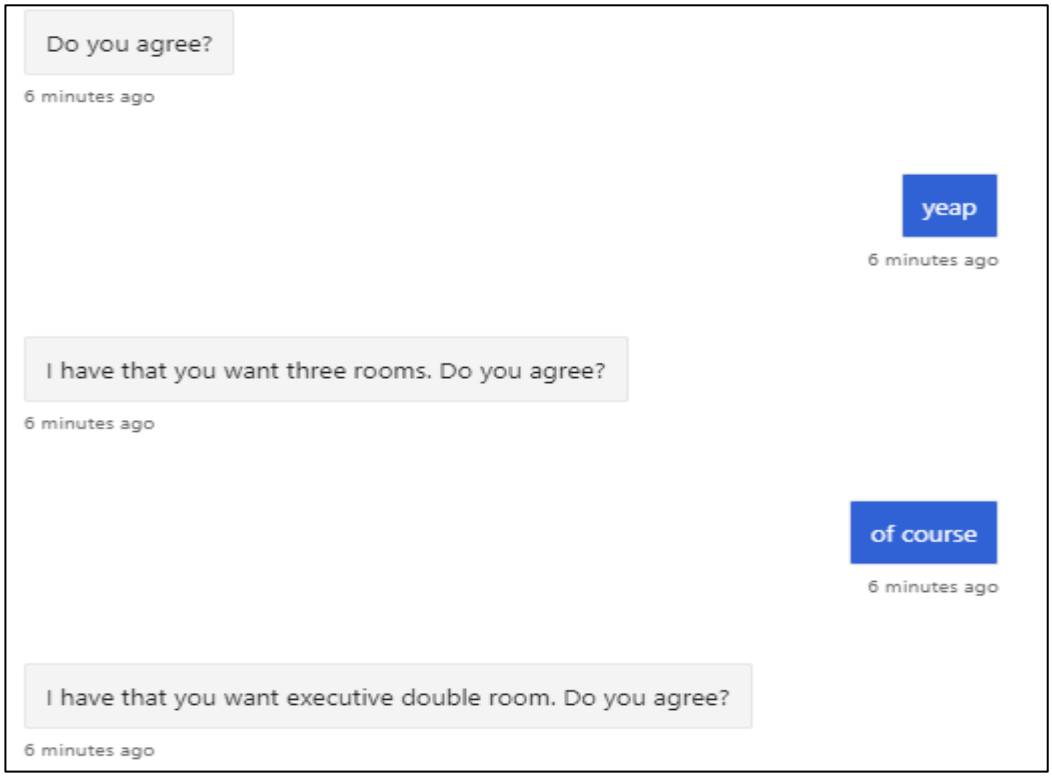

Can you inform me about room choices?
6 minutes ago
There are offered 3 types of rooms: Deluxe Room, Superior Deluxe Room and Executive Room.
6 minutes ago
Can I see photos from rooms?
5 minutes ago
Have a look here for rooms photos.
5 minutes ago

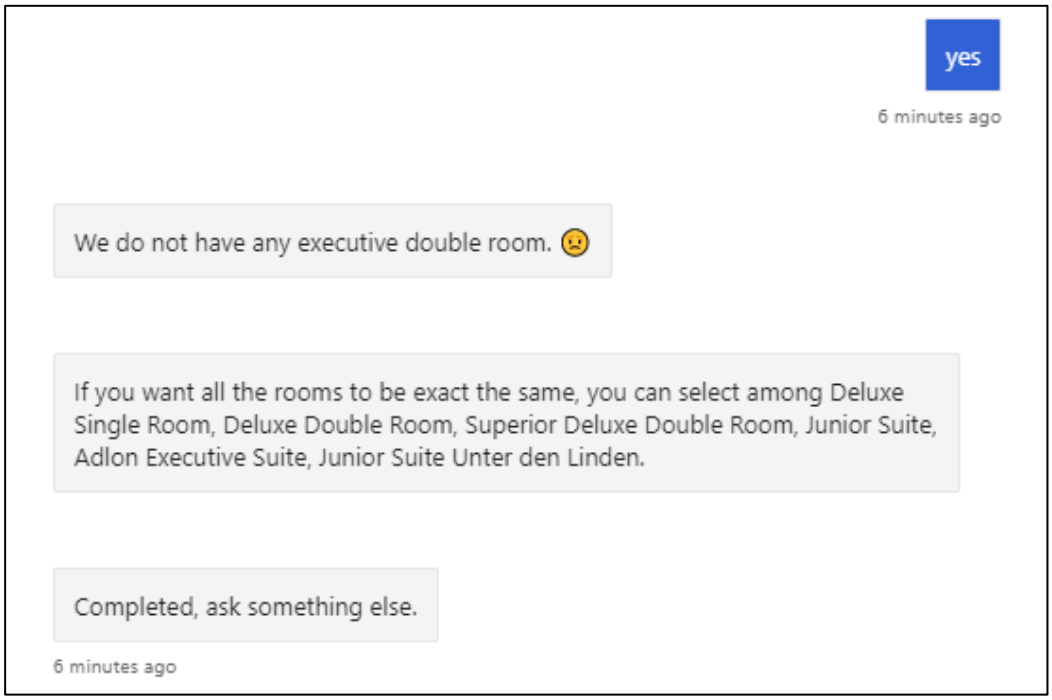

CHECK IN
CHECK OUT
1 ROOM
SEARCH


can i book three executive double room from 06/08 to 06/09?
6 minutes ago
I have your check in date as June 8, 2021.
Do you agree?
6 minutes ago
No
6 minutes ago
Give again when is your check-in date ?
6 minutes ago
August 6
5 minutes ago


I have your check in date as August 6, 2021.
Do you agree?
6 minutes ago
yes
6 minutes ago
I have your check out date as June 9, 2021.


Do you agree?
6 minutes ago
no
6 minutes ago
Give again when is your check-out date?
6 minutes ago
September 6
6 minutes ago
I have your check out date as September 6, 2021.

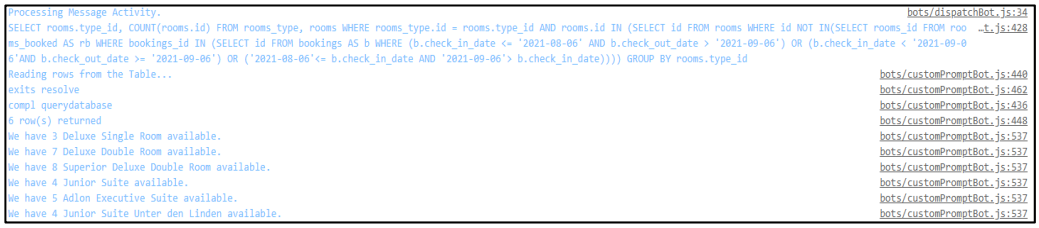

Do you agree?
6 minutes ago
yeap
6 minutes ago
I have that you want three rooms. Do you agree?
6 minutes ago
of course
6 minutes ago
I have that you want executive double room. Do you agree?
6 minutes ago

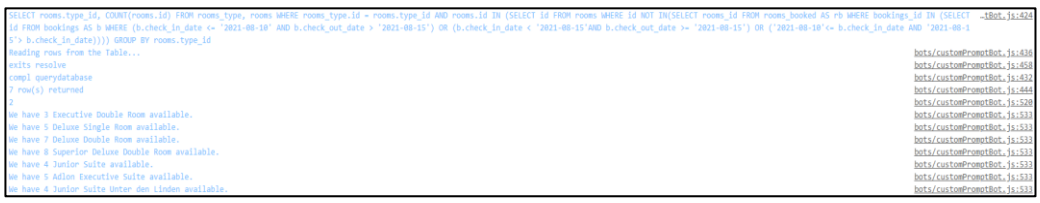

yes
6 minutes ago
We do not have any executive double room.
If you want all the rooms to be exact the same, you can select among Deluxe Single Room, Deluxe Double Room, Superior Deluxe Double Room, Junior Suite, Adlon Executive Suite, Junior Suite Unter den Linden.
Completed, ask something else.
6 minutes ago


Can i book two Deluxe Single rooms check in 10 August and check out 15 August ?
7 minutes ago
I have your check in date as August 10, 2021.
Do you agree?
7 minutes ago
yes
6 minutes ago
I have your check out date as August 15, 2021.
Do you agree?
6 minutes ago
yes
6 minutes ago

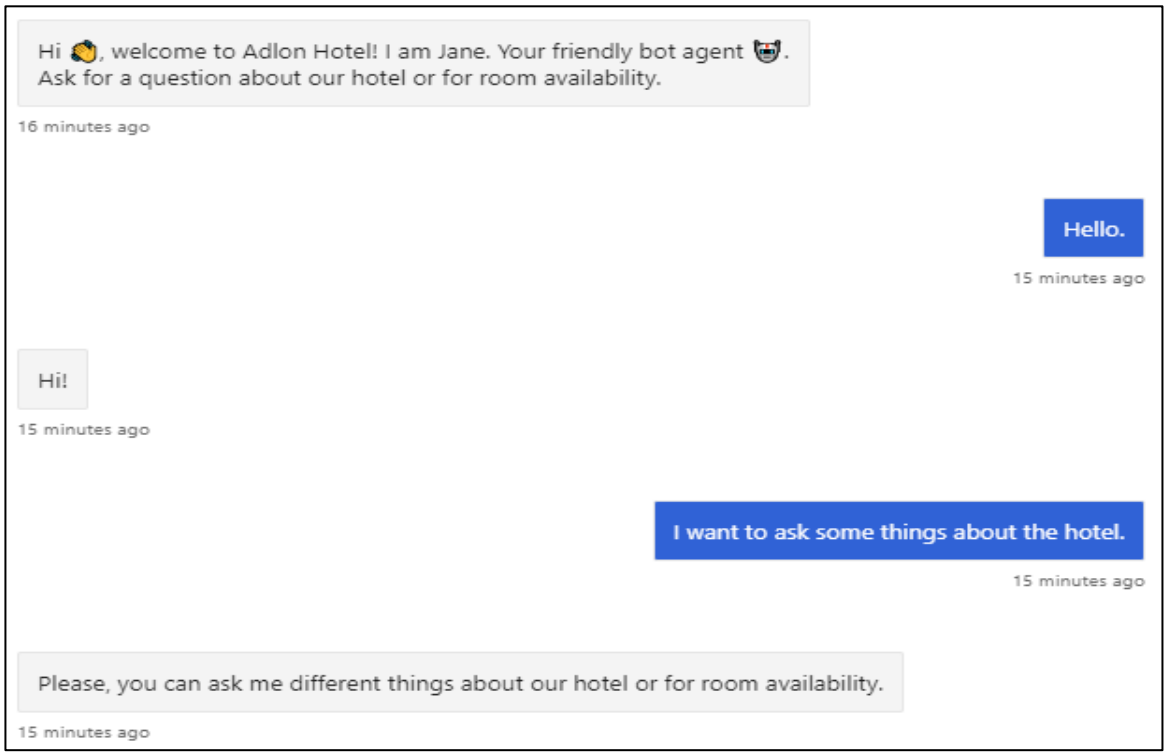

I have that you want two rooms. Do you agree?
8 minutes ago
yes
8 minutes ago
I have that you want deluxe single rooms. Do you agree?
8 minutes ago
exit
7 minutes ago
Cancelling ...
7 minutes ago

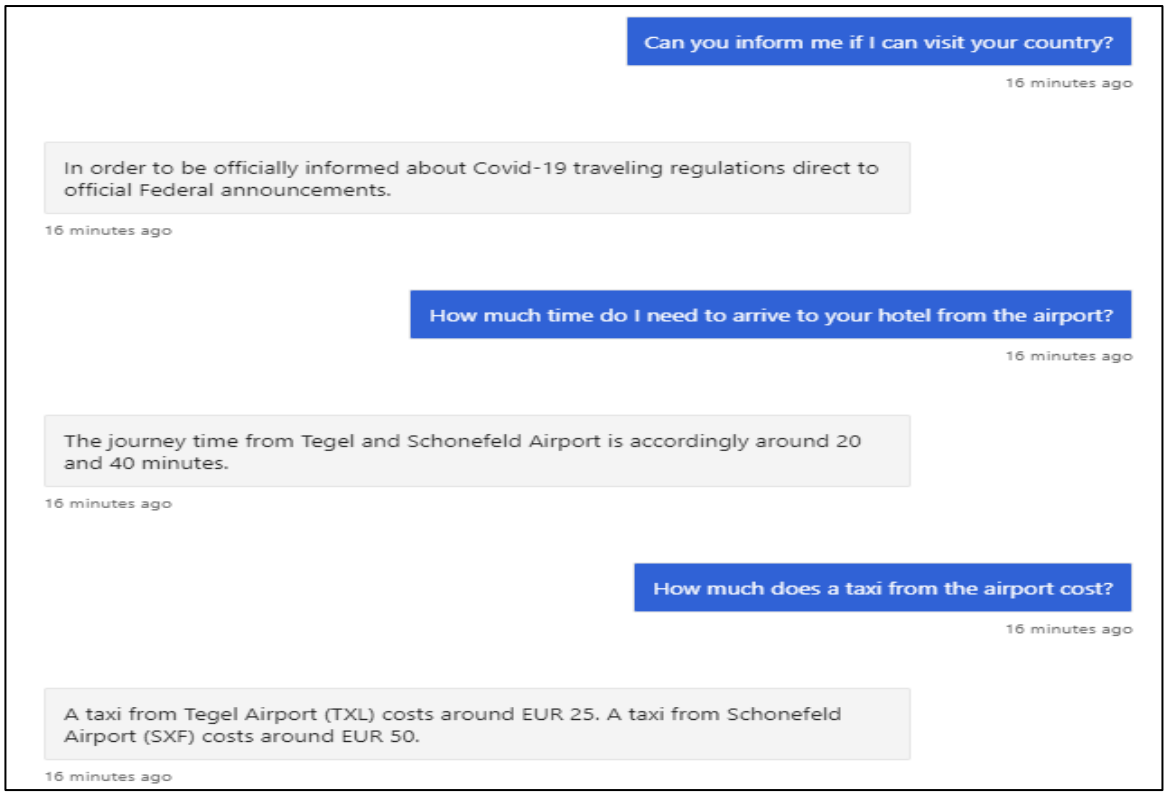

do you have email?
Just now
You can Email us on: hotel.adlon@kempinski.com.
Just now

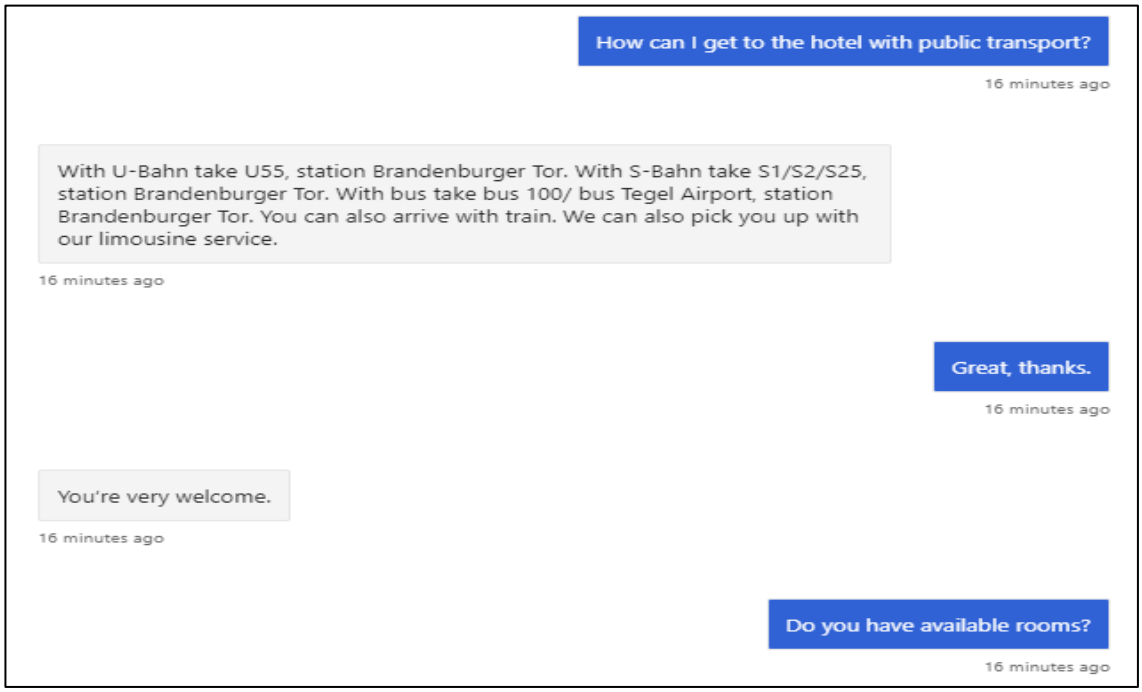

Processing Message Activity. bots/dispatchBot.js:34
SELECT rooms.type_id, COUNT(rooms.id) FROM rooms_type, rooms WHERE rooms_type.id = rooms.type_id AND rooms.id IN (SELECT id FROM rooms WHERE id NOT IN(SELECT rooms_id FROM roo …t.js:428
ms_booked AS rb WHERE bookings_id IN (SELECT id FROM bookings AS b WHERE (b.check_in_date <= '2021-08-06' AND b.check_out_date > '2021-09-06') OR (b.check_in_date < '2021-09-0
6'AND b.check_out_date >= '2021-09-06') OR ('2021-08-06'<= b.check_in_date AND '2021-09-06'> b.check_in_date)))) GROUP BY rooms.type_id
Reading rows from the Table... bots/customPromptBot.js:440
exits resolve bots/customPromptBot.js:462
compl querydatabase bots/customPromptBot.js:436
6 row(s) returned bots/customPromptBot.js:448
We have 3 Deluxe Single Room available. bots/customPromptBot.js:537
We have 7 Deluxe Double Room available. bots/customPromptBot.js:537
We have 8 Superior Deluxe Double Room available. bots/customPromptBot.js:537
We have 4 Junior Suite available. bots/customPromptBot.js:537
We have 5 Adlon Executive Suite available. bots/customPromptBot.js:537
We have 4 Junior Suite Unter den Linden available. bots/customPromptBot.js:537

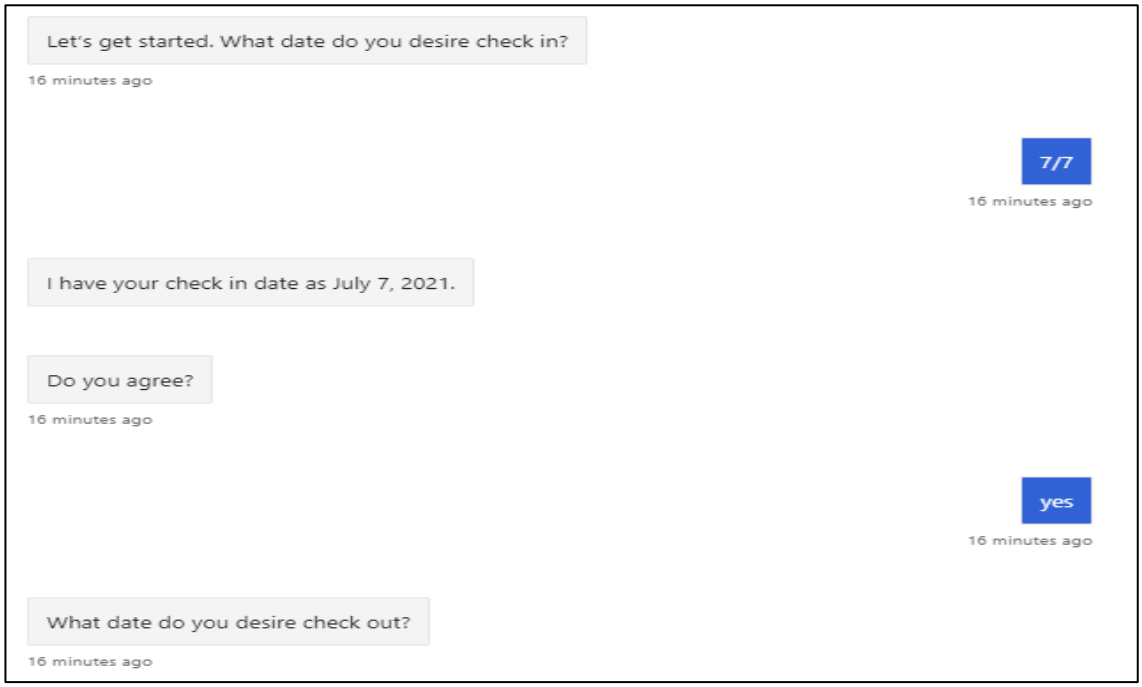

SELECT rooms.type_id, COUNT(rooms.id) FROM rooms_type, rooms WHERE rooms_type.id = rooms.type_id AND rooms.id IN (SELECT id FROM rooms WHERE id NOT IN(SELECT rooms_id FROM rooms_booked AS rb WHERE bookings_id IN (SELECT …tBot.js:424
id FROM bookings AS b WHERE (b.check_in_date <= '2021-08-10' AND b.check_out_date > '2021-08-15') OR (b.check_in_date < '2021-08-15'AND b.check_out_date >= '2021-08-15') OR ('2021-08-10'<= b.check_in_date AND '2021-08-1
5'> b.check_in_date)))) GROUP BY rooms.type_id
Reading rows from the Table... bots/customPromptBot.js:436
exits resolve bots/customPromptBot.js:458
compl querydatabase bots/customPromptBot.js:432
7 row(s) returned bots/customPromptBot.js:444
2 bots/customPromptBot.js:520
We have 3 Executive Double Room available. bots/customPromptBot.js:533
We have 5 Deluxe Single Room available. bots/customPromptBot.js:533
We have 7 Deluxe Double Room available. bots/customPromptBot.js:533
We have 8 Superior Deluxe Double Room available. bots/customPromptBot.js:533
We have 4 Junior Suite available. bots/customPromptBot.js:533
We have 5 Adlon Executive Suite available. bots/customPromptBot.js:533
We have 4 Junior Suite Unter den Linden available. bots/customPromptBot.js:533

Annex C

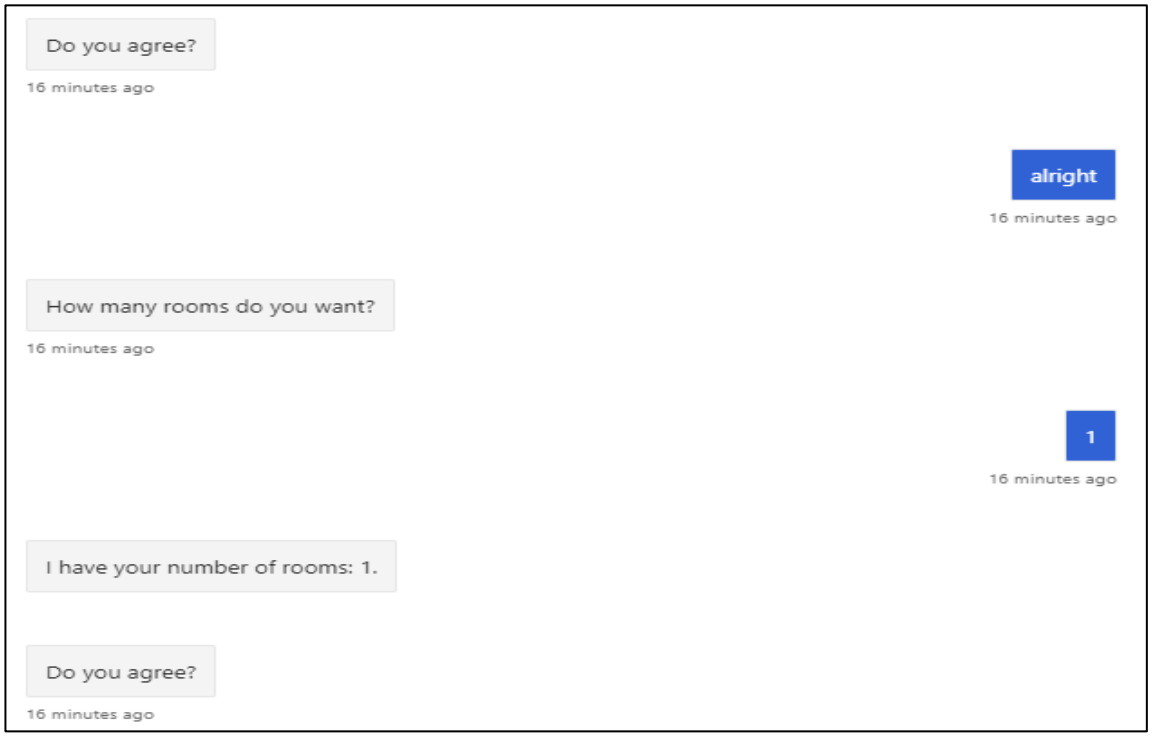

Hi, welcome to Adlon Hotel! I am Jane. Your friendly bot agent.
Ask for a question about our hotel or for room availability.
16 minutes ago
Hello.
15 minutes ago
Hi!
15 minutes ago
I want to ask some things about the hotel.
15 minutes ago
Please, you can ask me different things about our hotel or for room availability.
15 minutes ago

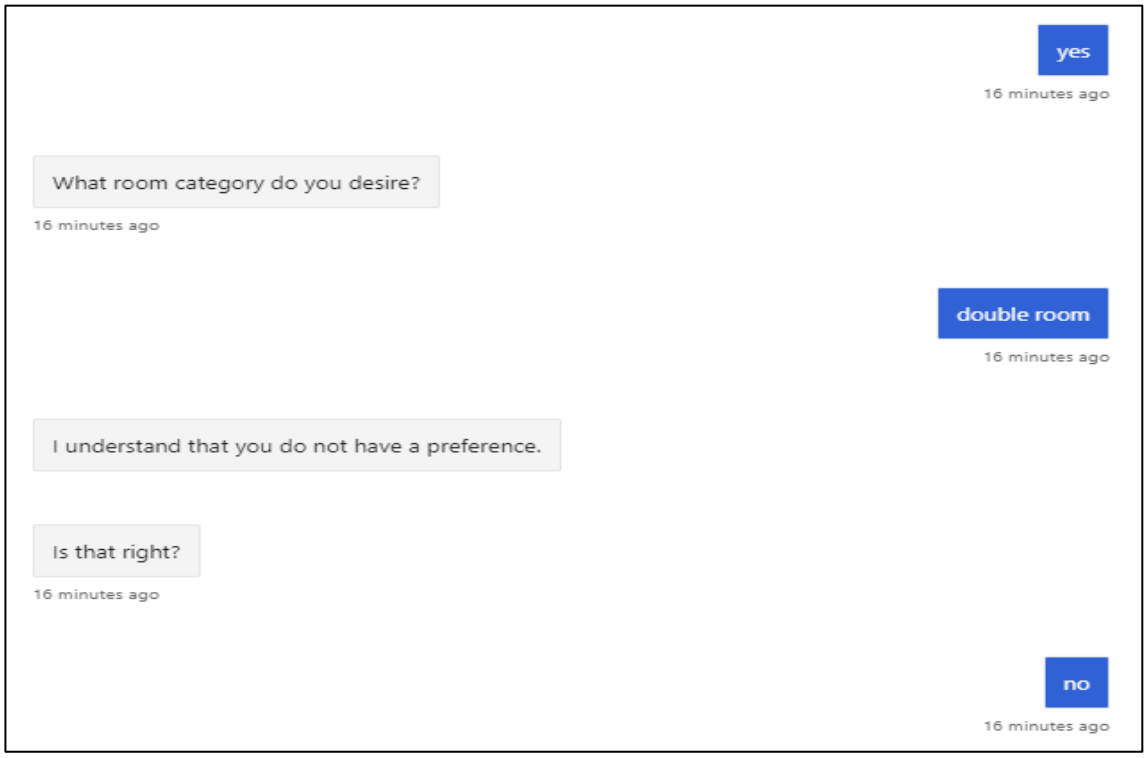

Can you inform me if I can visit your country?
16 minutes ago
In order to be officially informed about Covid-19 traveling regulations direct to official Federal announcements.
16 minutes ago
How much time do I need to arrive to your hotel from the airport?
16 minutes ago
The journey time from Tegel and Schonefeld Airport is accordingly around 20 and 40 minutes.
16 minutes ago
How much does a taxi from the airport cost?
16 minutes ago
A taxi from Tegel Airport (TXL) costs around EUR 25. A taxi from Schonefeld Airport (SXF) costs around EUR 50.
16 minutes ago

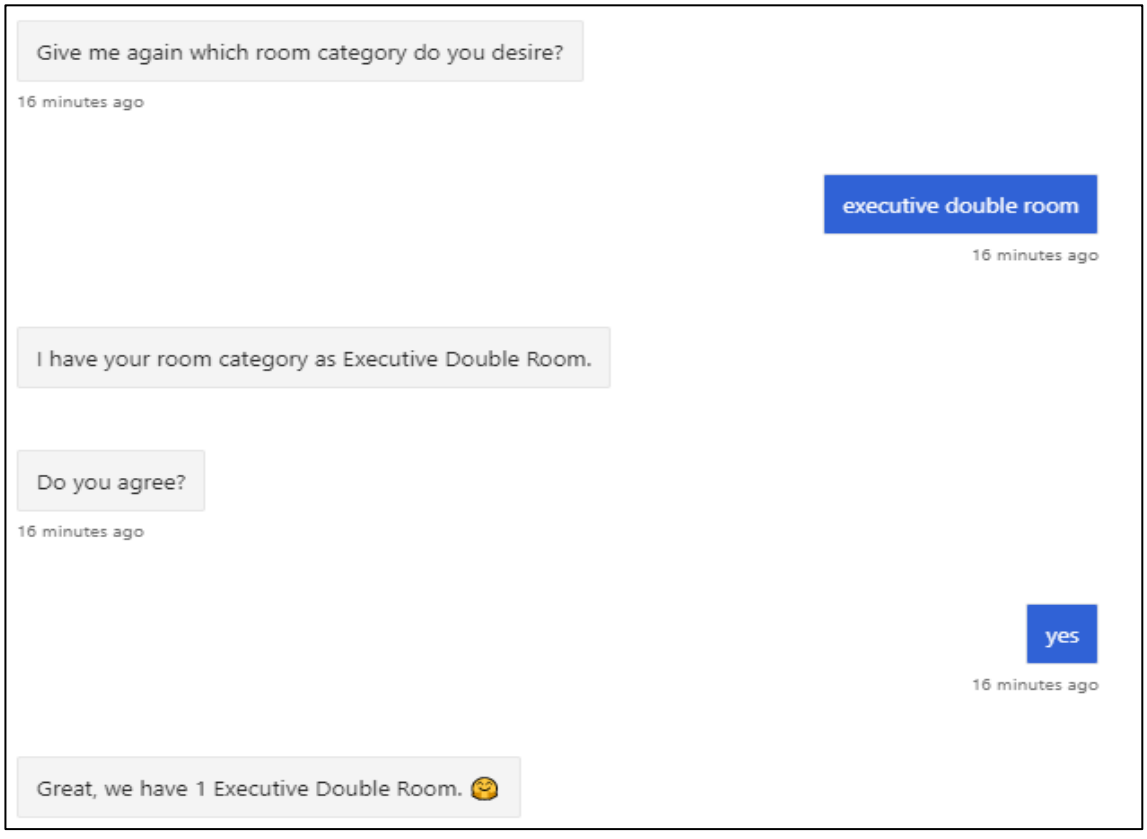

How can I get to the hotel with public transport?
16 minutes ago
With U-Bahn take U55, station Brandenburger Tor. With S-Bahn take S1/S2/S25, station Brandenburger Tor. With bus take bus 100/ bus Tegel Airport, station Brandenburger Tor. You can also arrive with train. We can also pick you up with our limousine service.
16 minutes ago
Great, thanks.
16 minutes ago
You're very welcome.
16 minutes ago
Do you have available rooms?
16 minutes ago


Let's get started. What date do you desire check in?
16 minutes ago
7/7
16 minutes ago
I have your check in date as July 7, 2021.
Do you agree?
16 minutes ago
yes
16 minutes ago
What date do you desire check out?
16 minutes ago

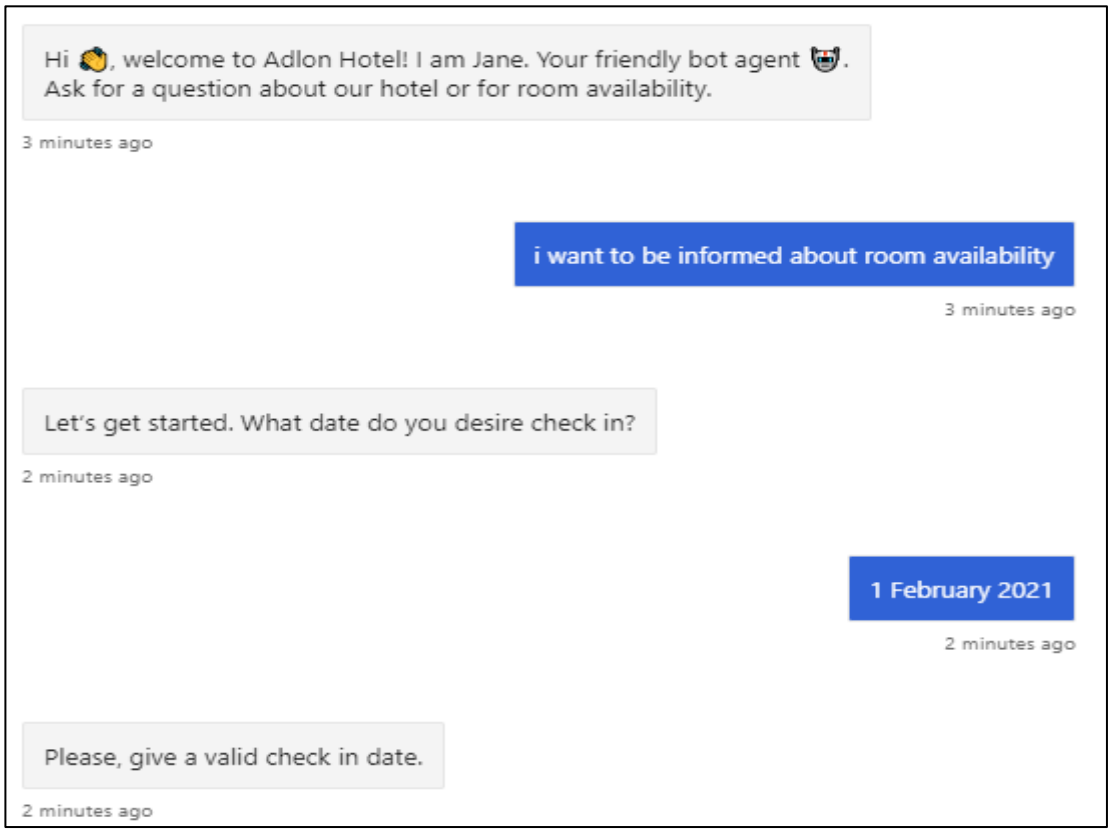

12/7
16 minutes ago
I have your check out date as December 7, 2021.
Do you agree?
16 minutes ago
no
16 minutes ago
Give again when is your check-out date?
16 minutes ago
12 July
16 minutes ago
I have your check out date as July 12, 2021.

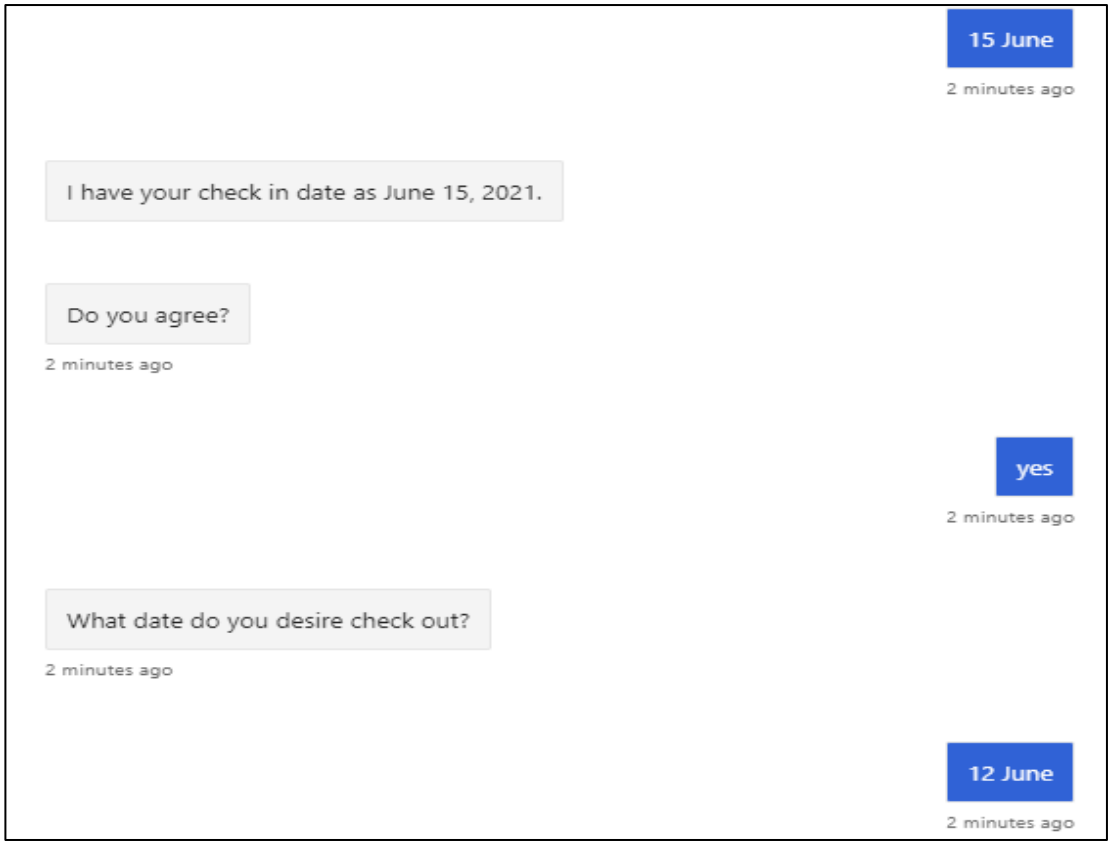

Do you agree?
16 minutes ago
alright
16 minutes ago
How many rooms do you want?
16 minutes ago
1
16 minutes ago
I have your number of rooms: 1.
Do you agree?
16 minutes ago

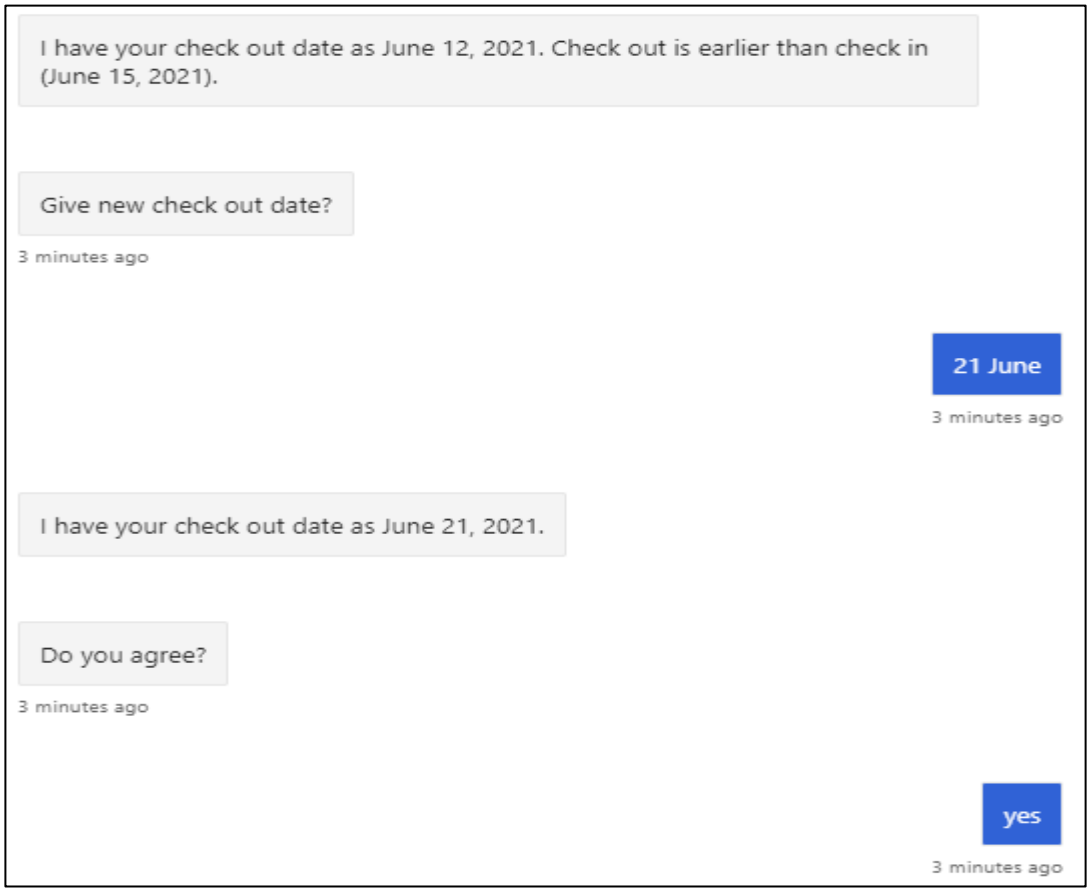

yes
16 minutes ago
What room category do you desire?
16 minutes ago
double room
16 minutes ago
I understand that you do not have a preference.
Is that right?
16 minutes ago
no
16 minutes ago

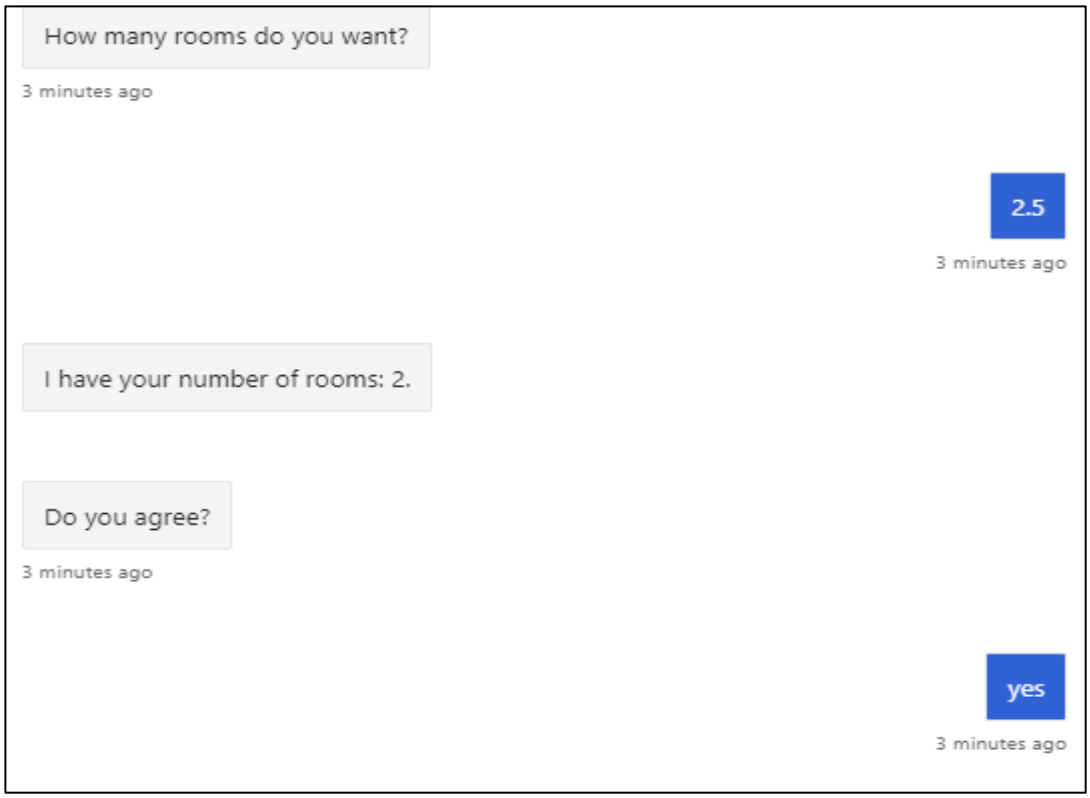

Give me again which room category do you desire?
16 minutes ago
executive double room
16 minutes ago
I have your room category as Executive Double Room.
Do you agree?
16 minutes ago
yes
16 minutes ago
Great, we have 1 Executive Double Room.

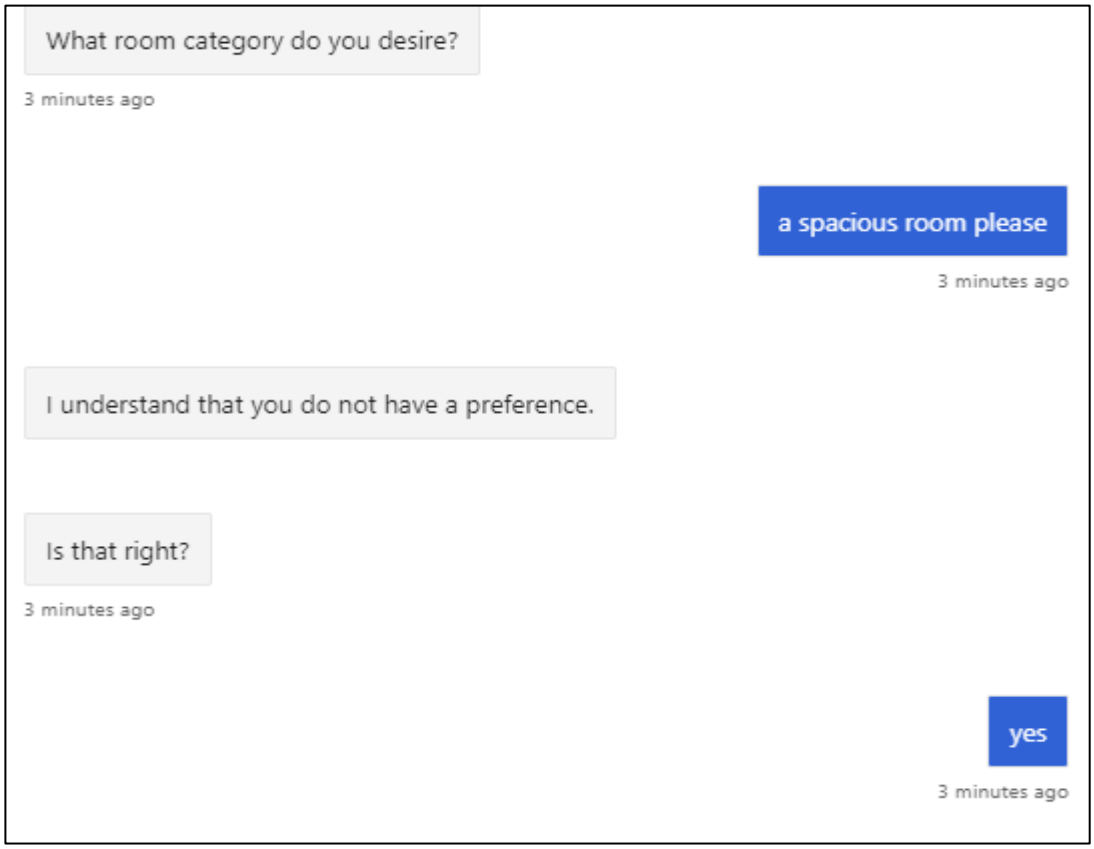
Annex D

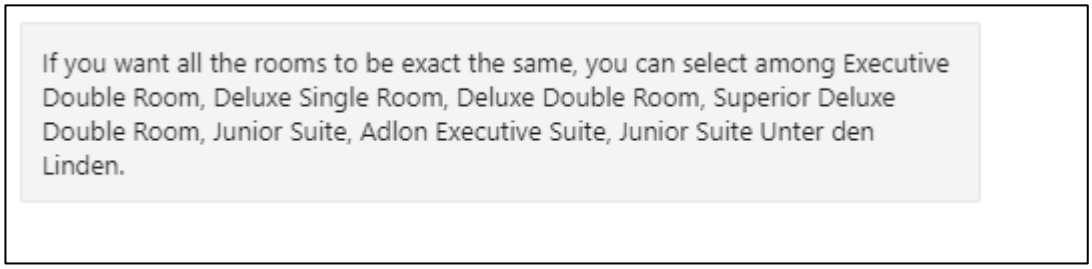

Hi, welcome to Adlon Hotel! I am Jane. Your friendly bot agent.
Ask for a question about our hotel or for room availability.
3 minutes ago
i want to be informed about room availability
3 minutes ago
Let's get started. What date do you desire check in?
2 minutes ago
1 February 2021
2 minutes ago
Please, give a valid check in date.
2 minutes ago


15 June
2 minutes ago
I have your check in date as June 15, 2021.
Do you agree?
2 minutes ago
yes
2 minutes ago
What date do you desire check out?
2 minutes ago
12 June
2 minutes ago


I have your check out date as June 12, 2021. Check out is earlier than check in (June 15, 2021).
Give new check out date?
3 minutes ago
21 June
3 minutes ago
I have your check out date as June 21, 2021.
Do you agree?
3 minutes ago
yes
3 minutes ago


How many rooms do you want?
3 minutes ago
2.5
3 minutes ago
I have your number of rooms: 2.
Do you agree?
3 minutes ago
yes
3 minutes ago


What room category do you desire?
3 minutes ago
a spacious room please
3 minutes ago
I understand that you do not have a preference.
Is that right?
3 minutes ago
yes
3 minutes ago


If you want all the rooms to be exact the same, you can select among Executive Double Room, Deluxe Single Room, Deluxe Double Room, Superior Deluxe Double Room, Junior Suite, Adlon Executive Suite, Junior Suite Unter den Linden.