Johannes Kühnel, BSc

Chatbots for Brand Representation in Comparison with Traditional Websites

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Adj.-Prof. Dipl.-Ing. Dr.techn. Martin Ebner

Co-Supervisor
Dipl.-Ing. Markus Ebner

Institute for Interactive Systems and Data Science
Head: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

Graz, April 2019

This document is set in Palatino, compiled with pdfLaTeX2e and `Biber`.

The LaTeX template from Karl Voit is based on KOMA script and can be found online: https://github.com/novoid/LaTeX-KOMA-template

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date	Signature

Abstract

Chatbots have gained enormous popularity in recent years. IT giants such as Microsoft, Google and Facebook have taken an interest in automated conversations. Messaging apps like WhatsApp and Facebook Messenger are playing an increasingly important role in smartphone usage and communication in general - perfect conditions for chatbots.

This master thesis provides an overview of chatbots in general and in customer environments. Furthermore, platforms and services for the creation and development of bots as well as for the integration of Natural Language Understanding are examined.

As part of this work, a chatbot was developed for the Austrian IT company CodeFlügel. The chatbot, named Theodore, provides information about CodeFlügel via Facebook Messenger and webchat, similar to the existing company website. The design process and implementation of the chatbot as well as architectural considerations are explained throughout this document.

In a user study, participants perform typical tasks with the website and the chatbot. The usage of both platforms is evaluated in order to identify advantages and disadvantages as well as differences compared to the other technology and to draw conclusions. The study shows the mostly positive reactions of the users towards the chatbot, especially in terms of acceptance and faster completion times of specific tasks. Most users also perceive the chatbot as more entertaining than the website and could see themselves using more chatbots in the future.

Kurzfassung

Chatbots haben in den letzten Jahren enorm an Popularität gewonnen. IT-Größen wie Microsoft, Google und Facebook haben sich der automatisierten Konversationen angenommen. Nachrichten-Apps wie WhatsApp und Facebook Messenger nehmen eine immer wichtigere Rolle in der Smartphone-Nutzung und der Kommunikation im Allgemeinen ein - perfekte Voraussetzungen für Chatbots.

In dieser Masterarbeit wird ein Überblick über Chatbots im Allgemeinen und im Kundenumfeld gegeben. Weiters werden Plattformen und Services zur Erstellung und Entwicklung von Bots, sowie zur Integration von Natural Language Understanding, beleuchtet.

Im Rahmen der Arbeit wurde auch ein Chatbot für die österreichische IT-Firma CodeFlügel entwickelt. Der auf den Namen Theodore getaufte Chatbot vermittelt über Facebook Messenger und Webchat Informationen über CodeFlügel, ähnlich zur bereits bestehenden Firmen-Webseite. Der Design-Prozess und die Implementierung des Chatbots sowie Hintergründe zur Architektur werden im Laufe dieses Dokumentes erläutert.

In einer Nutzerstudie führen teilnehmende Personen typische Aufgaben mit der Webseite und dem Chatbot durch. Die Nutzung beider Plattformen wird evaluiert, um Vor- und Nachteile sowie Unterschiede gegenüber der jeweils anderen Technologie zu erkennen und Schlüsse zu ziehen. Die Studie zeigt die größtenteils positiven Reaktionen der Nutzer auf den Chatbot, insbesondere in Bezug auf dessen Akzeptanz und die schnellere Lösung bestimmter Aufgaben. Die meisten Benutzer empfinden den Chatbot als unterhaltsamer als die Website und könnten sich vorstellen, in Zukunft mehr Chatbots zu verwenden.

Contents

List of Figures

List of Tables

Listings

1. Introduction

Chatbots and conversational interfaces have become increasingly popular in recent years. At the developer conference *Build 2016*, Microsoft's CEO Satya Nadella boldly proclaimed that "Bots are the new apps" (USA Today, 2016). Several factors have nurtured this trend. On the one hand, messaging applications like WhatsApp and Facebook Messenger are attracting billions of users (see Table 2.7) and occupying a large portion of a user's screen time (Comscore, 2015). On the other hand, major technology companies like IBM, Microsoft, Google, Facebook and Amazon have taken an interest in providing development tools and natural language processing centered around chatbots, resulting in easier development and wider acceptance and adaption of conversational interfaces. (Klopfenstein et al., 2017)

In this master thesis, an overview and history of chatbots in general and in customer environments is given in chapter 2. This chapter also examines several services for Natural Language Understanding (NLU) and takes a closer look at popular messaging systems with bot support. Frameworks and platforms for developing chatbots and conversational interfaces are also discussed.

Chapter 3 presents Theodore, a chatbot that was developed for the Austrian IT company CodeFlügel. The chatbot provides information about CodeFlügel via Facebook Messenger and webchat, similar to the existing company website. The design process as well as architectural considerations are explained throughout this chapter. The implementation of the chatbot is described in detail in chapter 4.

In chapter 5, a user study is conducted where participants perform typical tasks with the website and the chatbot. The usage of both platforms is evaluated in order to identify advantages and disadvantages as well as differences compared to the other technology. Based on these evaluations, conclusions will further be drawn concerning, for example, differences in speed and user satisfaction. Findings to this study are shown in chapter 6.

Chapter 7 summarizes this thesis and chapter 8 provides an outlook of possible future work.

2. Chatbots

Chatbots (also referred to as chatterbots) are programs designed to understand natural language and to reply accordingly. They provide a service through a so-called conversational interface, as opposed to a normal program's Graphical User Interface (GUI). While chatbots often use some sort of Artificial Intelligence (AI) and sometimes even involve Machine Learning (ML), they can also use simple key words and pattern matching to "understand" what the user is saying. (Shawar and Atwell, 2007; Shevat, 2017)

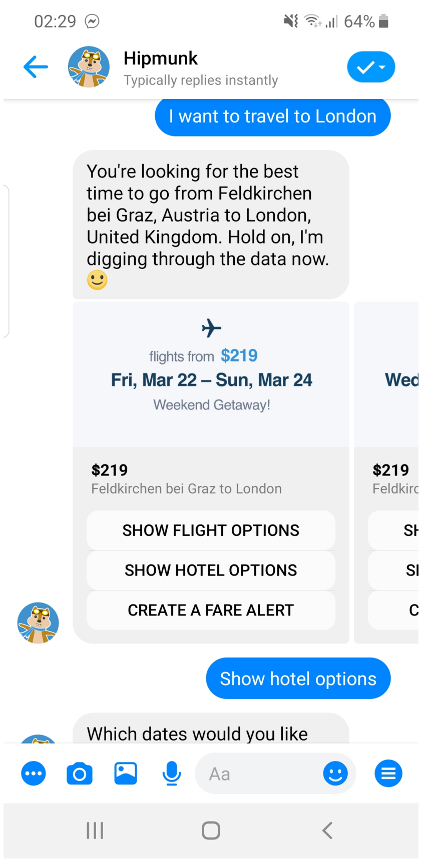
Chatbots are available for different platforms such as messaging applications (see section 2.5) or operating systems (for example, Apple Siri on iOS). Different use cases require different abilities and traits of a chatbot. For example, a chatbot could enable you to play a quiz and feature a personality specialized to entertain kids and figure 2.1 shows *Hipmunk*[1], a chatbot that helps you plan your travels on the chat app Facebook Messenger.

This chapter gives an overview of the history (section 2.1) of chatbots and the different types one can encounter (section 2.2). In section 2.3 several examples of chatbots in customer interaction are listed. Section 2.4, section 2.5 and section 2.6 shed light on different services and frameworks for building chatbots.

2.1. History

In 1966, Joseph Weizenbaum published a paper about his chatbot *ELIZA* (Weizenbaum, 1966). ELIZA is a computer program enabling communication

[1] `https://m.me/hipmunk`, visited on 2019-01-24


02:29

64%

Hipmunk

Typically replies instantly

I want to travel to London

You're looking for the best time to go from Feldkirchen bei Graz, Austria to London, United Kingdom. Hold on, I'm digging through the data now. 🙂

flights from $219

Fri, Mar 22 – Sun, Mar 24

Weekend Getaway!

$219

Feldkirchen bei Graz to London

SHOW FLIGHT OPTIONS

SHOW HOTEL OPTIONS

CREATE A FARE ALERT

Wed

$219

Feldkirc

SH

SI

C

Show hotel options

Which dates would you like

Aa

Figure 2.1.: Chatbot Hipmunk (`https://m.me/hipmunk`, visited on 2019-01-24) on Facebook Messenger provides travel information on request.

between a human and a computer using natural language. The chatbot was one of the first to attain public recognition and was even believed by some of its users to pass the *Turing Test*[2]. ELIZA used key words in the user's input to reply with corresponding answers. These answers were scripted. For example, requests containing "mother" would trigger a response like "Tell me more about your family" when using a psychotherapist ELIZA script.

Since then, several other chatbots like *ALICE* and *Jabberwacky* (Thompson, 2007) have been created. From 2011 onward, virtual assistants such as Apple's Siri[3], Google Now/Assistant[4] and Amazon's Alexa[5] helped to make chatbots publicly available and accepted. In 2013, the Chinese messenger WeChat started a bot platform (Chatbots Magazine, 2017) and in 2016 they had around 806 million active users[6], most of which were based in China. It was not until Facebook released their *Messenger Platform SDK* in 2016[7] though, that text-based chatbots gained new attention in the western world[8]. Back then, Messenger had around 1 billion active users, increasing to 1.3 billion[9] in 2017 with over 100,000 monthly active bots[10]. Research and advisory firm Gartner (2017), known for its *Gartner Hype Cycle*[11], even predicts that "more than 50% of companies will spend more per annum on bots and chatbot creation than traditional mobile app development" by 2021.

While popular app stores like Google Play[12] and Apple App Store[13] featured

[2]The Turing Test is a test of a machine or program's intelligence by determining if it is indistinguishable from a human counterpart in a textual conversion (Turing, 1950)

[3]Apple, 2011.

[4]Android, 2013.

[5]Android Central, 2015.

[6]Tencent, 2016.

[7]`https://newsroom.fb.com/news/2016/04/messenger-platform-at-f8/`, visited on 2019-01-24

[8]Bager, 2016.

[9]VentureBeat, 2017.

[10]Facebook, 2017.

[11]`https://www.gartner.com/en/research/methodologies/gartner-hype-cycle`, visited on 2019-01-24

[12]`https://play.google.com/store`, visited on 2019-03-24

[13]`https://www.apple.com/ios/app-store/`, visited on 2019-03-24

around two million mobile apps in Q3 2018[14], smartphone users spend about 80% of their smartphone app minutes in only three apps, with social media and messaging amounting to 35% of the overall time spent on apps, as shown by Comscore (2015). This makes it hard for new mobile applications to conquer the user's valuable time or even get installed. Chatbots do not need to be installed, however, they can be used from within existing applications right away. Another important advantage of chatbots compared to mobile apps is that they can drive sales while not being subject to revenue cuts by app store providers.

> "So that's a new way of how computing is accessed. Just like the browser in the past was born on the desktop PC, the web was born on the desktop, these conversational interfaces will be born on the devices you use today as an additional entry point, as an additional experience, but over time will fundamentally revolutionize how computing is experienced by everybody." (Satya Nadella, Microsoft CEO[15])

Machine Learning played a huge role in the resurgence of chatbots, and Artificial Intelligence (AI) has come a long way since ELIZA. The teams of *Deepmind*[16] and *OpenAI*[17] even created and trained bots to compete with the world elite in the popular e-Sports games StarCraft II[18] and Dota2[19]. Today, global players like Google, Microsoft, IBM and Facebook offer services for natural language processing and various other types of useful cognitive services for chatbots (see section 2.4).

According to Gartner's Van Baker (Gartner, 2018) "over 50% of medium to large enterprises will have deployed product chatbots" by 2020.

[14]Appfigures, 2018b; Appfigures, 2018a.

[15]Nadella, 2016b.

[16]https://deepmind.com, visited on 2019-01-24

[17]https://openai.com/, visited on 2019-01-24

[18]https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, visited on 2019-01-24

[19]https://blog.openai.com/the-international-2018-results/, visited on 2019-01-24

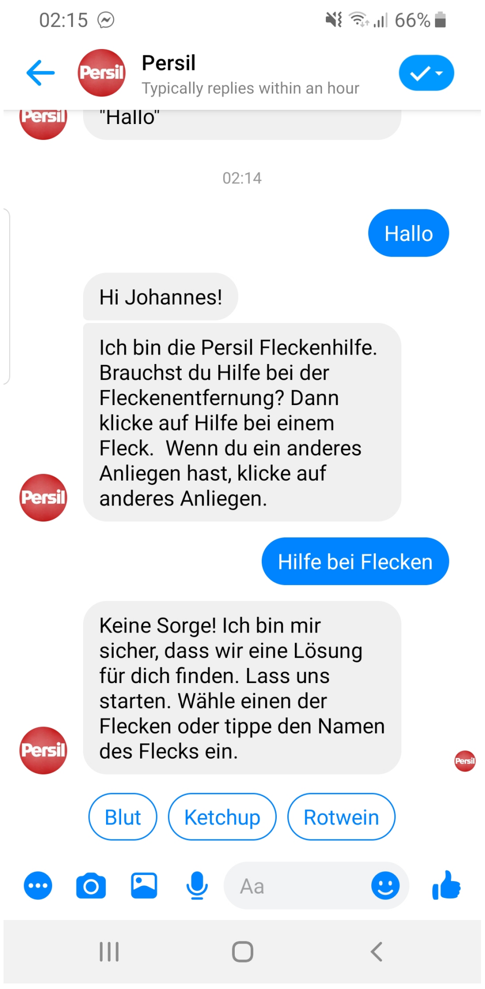
2.2. Types of Chatbots

Chatbots can vary in many aspects such as target audience, use case, knowledge and domain, type of communication (text or voice) and others. Chatbots for team messenger Slack[20] often work with many users inside a channel simultaneously (one-to-many) instead of with a single user (one-to-one) via direct message. Figure 2.2 shows two examples of one-to-one and one-to-many bots.

Virtual assistants such as Alexa and Siri need to cover many different knowledge and task domains, as they aid their users in playing music, managing shopping lists, turning on the lights and much more. Shevat (2017) names them "super-bots". These kinds of bots require an immense amount of work to provide all of those features. Such personal assistants mainly use voice to communicate with the user, which introduces text-to-speech and speech recognition to the architecture. Alexa can be extended by so called *Alexa Skills*, but they have to follow a specific set of rules for their activation. In comparison, chatbots such as Lunch Train (Figure 2.2) only have to fulfill a specific task and are easier to create and maintain while enabling a different behavior and usability. Figure 2.3 shows an overview of different bot types and traits (Shevat, 2017). Characterization and categorization of chatbots can be done in other ways as well. Lebeuf, Storey, and Zagalsky (2017) categorized bots in terms of their interaction model, intelligence and purpose, while Paikari and Hoek (2018) classified the three types *Information*, *Collaboration* and *Automation*.

Another important aspect of a bot is its target audience and use case. Some bots are built for customers of a specific service, like KLM Royal Dutch Airlines' *BB*[21], which enables users to book flights from within Facebook Messenger. Other chatbots help users in solving business tasks — for example, time tracking or project management. While consumer bots often have a funny personality to keep users engaged, business bots tend to be more task-oriented. (Shevat, 2017)

[20] `https://slack.com/`, visited on 2019-01-29
[21] `https://bb.klm.com/`, visited on 2019-01-25



02:15

Persil
Typically replies within an hour

"Hallo"

02:14

Hallo

Hi Johannes!

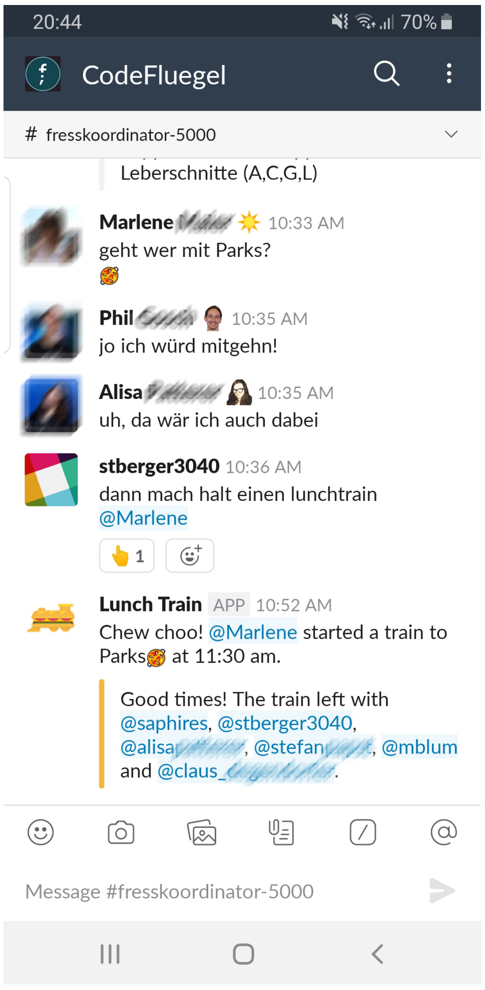
Ich bin die Persil Fleckenhilfe. Brauchst du Hilfe bei der Fleckenentfernung? Dann klicke auf Hilfe bei einem Fleck. Wenn du ein anderes Anliegen hast, klicke auf anderes Anliegen.

Hilfe bei Flecken

Keine Sorge! Ich bin mir sicher, dass wir eine Lösung für dich finden. Lass uns starten. Wähle einen der Flecken oder tippe den Namen des Flecks ein.

Blut Ketchup Rotwein

(a) Persil Germany chatbot on Facebook Messenger



20:44

CodeFluegel

fresskoordinator-5000

Leberschnitte (A,C,G,L)

Marlene [redacted] 10:33 AM
geht wer mit Parks?

Phil [redacted] 10:35 AM
jo ich würd mitgehn!

Alisa [redacted] 10:35 AM
uh, da wär ich auch dabei

stberger3040 10:36 AM
dann mach halt einen lunchtrain @Marlene

Lunch Train APP 10:52 AM
Chew choo! @Marlene started a train to Parks at 11:30 am.

Good times! The train left with @saphires, @stberger3040, @alisa[redacted], @stefan[redacted], @mblum and @claus_[redacted].

Message #fresskoordinator-5000

(b) Bot *Lunch Train* on Slack

Figure 2.2.: The Persil Germany chatbot (`https://m.me/PersilDeutschland`, visited on 2019-01-24) (a) helps users with spot removal on their clothing (one-to-one). *Lunch Train* (`https://slack.com/apps/A1BES823B-lunch-train`, visited on 2019-01-24) (b) helps coordinating lunch at the office (one-to-many).

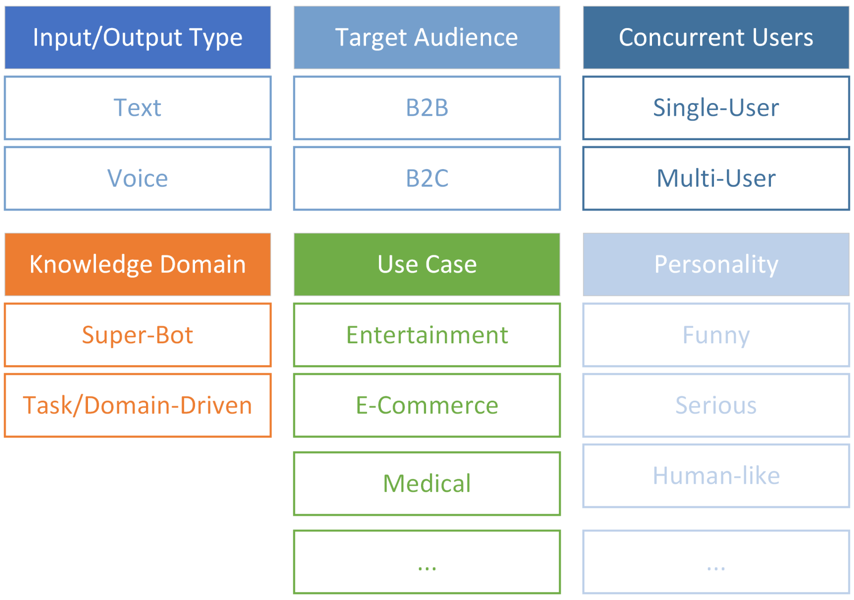


Figure 2.3.: Different types of chatbots based on Shevat (2017)

2.3. Chatbots for Customer Interaction and Related Work

As mentioned in the previous sections, chatbots provide an advantage compared to traditional mobile apps. Apps are prone to "app fatigue", which describes the tiredness of users to install new apps. Other driving factors of bots are the growth of messaging apps and social media, as well as the support of large companies like Facebook and Microsoft in the recent years (Chatbots Magazine and Khorozov, 2017; Klopfenstein et al., 2017). With these factors in mind, this chapter focuses on some case studies around the development and research of chatbots and conversational interfaces.

Shawar and Atwell (2015) showed a chatbot answering Frequently Asked Questions (FAQ) for the *School of Computing*[22] (University of Leeds). The bot used pre-processed data of the school's online FAQ on their website and pattern matching to map user input to questions and retrieve the correct answers. Users "found it a novel and interesting way to access the FAQ using natural language questions". Compared to a Google search, users were able to find a relevant answer more often and the majority of them preferred using the chatbot. Driving factors were the chatbots ability to provide direct answers to the question if there was only one matching answer and fewer links in general, resulting in less search time and better overview. (Shawar and Atwell, 2015)

Almost 50% of Internet users in the U.S. use social media to contact customer service. According to recent studies, more than two thirds of users expect an answer within an hour, but it usually takes over six times as long to get a response. Scaling can be a problem with high numbers of requests, which chatbots could solve. Xu et al. (2017) built a chatbot for customer service on Twitter, as a feasibility study on such software. They used deep learning algorithms and trained their bot with nearly a million Twitter conversations from more than 60 different companies and brands. One of their key findings was that more than 40% of customer requests are only emotional and not looking for any kind of information or help, possibly due to the nature of social media. Their customer service bot showed results close to a human

[22] https://engineering.leeds.ac.uk/computing, visited on 2019-03-14

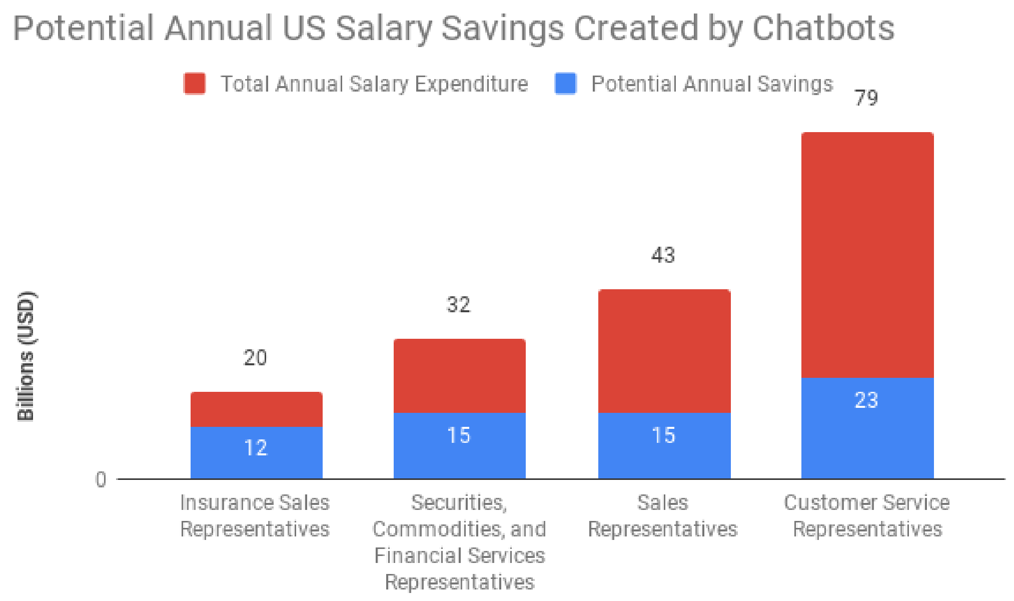


Figure 2.4.: Chart showing potential annual savings of US salaries in billion US dollars by using chatbots. Source: Business Insider — `https://www.businessinsider.de/80-of-businesses-want-chatbots-by-2020-2016-12`, visited on 2019-03-20

when handling such emotional requests, while informational requests seem to require more complex training. (Xu et al., 2017)

Another example for bots in customer interaction is *WAH Nails*[23]. Sharmadean Reid, the founder of the company, said that 30% of reservations were already handled by their chatbot only two months after its launch and plans to enhance it further using AI were already underway (Puscher, 2018). While this is only a single example, chatbots have the potential for significant savings in such applications, where human agents could be replaced or new services created. According to an infographic (Figure 2.4) by business magazine *Business Insider*[24], potential annual savings in customer service in the United States of America alone amount to $23 billion US dollars[25].

[23]`https://wah-london.com/`, visited on 2019-03-14
[24]`https://www.businessinsider.de`, visited on 2019-03-20
[25]`https://www.businessinsider.de/80-of-businesses-want-chatbots-by-2020-`

Beriault-Poirier, Tep, and Sénécal (2019) conducted an exploratory user study similar to the one shown in this thesis in chapter 5. The goal of the study was to learn more about the user experience provided by chatbots from different brands on Facebook Messenger, namely food and beverage company *Whole Foods*[26], clothing line *Tommy Hilfiger*[27] and travel search site *Skyscanner*[28]. Users were asked to complete two tasks for each brand (six in total) — one on their website and one with their Facebook Messenger chatbot. Chatbots and websites were then evaluated from 1 to 7 in terms of usefulness, ease of use, ease of learning and satisfaction. User experience scores and abandonment rates were favoring the companies' websites. Analysis of the participants' facial expressions, however, showed that chatbots generated more positive emotions. Participants liked the chatbots' ability to deliver quick and precise answers and also showed a willingness to use chatbots in the future. (Beriault-Poirier, Tep, and Sénécal, 2019)

Chatbots for customer interaction could also impose new problems. Heckmann and Kraus (2018) mention that depending on the specific use case, personal data must be handled carefully and usage of such data has to be disclosed. Commercial usage could also come with the requirement to inform users about those commercial intentions — for example, disclosing advertisements. Especially with the recent introduction of the General Data Protection Regulation (GDPR)[29], companies must take special care of protecting personal data in and outside of the European Union (EU). (Heckmann and Kraus, 2018)

While bots can help in building a brand and customer relation, a bot's personality is also able to do harm. As seen with Microsoft's Twitter-bot Tay (see section 2.5), machine learning and AI could lead to unforeseen behavior, which in turn could potentially damage a brand or company (Lebeuf, Storey, and Zagalsky, 2017). Brands and developers also have to keep in mind that a chatbot's presentation and persona can result in unwanted perception and

2016-12, visited on 2019-03-20

[26]https://www.wholefoodsmarket.com, visited on 2019-03-19

[27]http://www.tommy.com/, visited on 2019-03-19

[28]https://www.skyscanner.com/, visited on 2019-03-19

[29]https://ec.europa.eu/info/law/law-topic/data-protection_en, visited on 2019-03-17

emotions, especially in regards to the uncanny valley effect[30] (Araujo, 2018; Ciechanowski et al., 2019).

2.4. Natural Language Understanding

Natural Language Understanding (NLU) is the process of "understanding" natural language in computer science. In the case of a chatbot, a person passes commands, questions, or other input to the software via natural language. These inputs are then processed and the user's intent and additional parameters extracted. An intent describes the meaning or purpose of the input given by a user. A chatbot of a pizza delivery service could feature intents for ordering a pizza or asking for the order status. Additional parameters (entities) stand for terms tied to the intents. They could introduce a context, amount or other values. For the pizza, delivery the entities could be the type and amount of pizza as well as the delivery address. By recognizing what the user wants, the bot can use the appropriate workflow and reply accordingly.

NLU is often used in conjunction with machine learning — both are part of Artificial Intelligence (AI). In the case of chatbots, the AI is usually trained with big datasets of sample conversations. NLU services and frameworks help bot creators to focus on creating dialog workflows instead of having to deal with machine learning, neural networks and so on. (Shevat, 2017)

Amazon Lex

Amazon[31] *Lex*[32] launched in 2017 to allow developers to build chatbots and applications with the "same deep learning technologies that drive Amazon

[30]Described as "the feeling of eeriness and discomfort towards a given medium or technology that frequently appears in various kinds of human–machine interactions" by Ciechanowski et al. (2019).

[31]`https://www.amazon.com`, visited on 2019-02-02

[32]`https://aws.amazon.com/lex/`, visited on 2019-02-02

Name	Amazon Lex
URL	https://aws.amazon.com/lex/
Launch	2017
Supported Languages	English (US)
Speech to Text	yes
Versioning / Environments	yes / yes
Integrations	Webchat, Facebook Messenger, Slack, Kik and Twilio SMS
REST API	yes
SDKs	iOS, Android, Java, JS, Python, CLI, .NET, Ruby, PHP, Go, and C++
Pricing	limited free plan for the first year, standard rate is $0.00075 USD per text message

Table 2.1.: Amazon Lex fact sheet.

Alexa"[33]. It is part of *Amazon Web Services*[34] (AWS) and provides direct access to *Lambda* and other AWS services.

As for Alexa Skills, a sample input for intents is called an *utterance* and entities are named *slots*. Responses can be directly entered in the intent management. Versioning as well as environments are available, so a quick rollback to an earlier version is possible and different versions can be deployed to, for example, production and development systems. At this time, only English is supported by Amazon Lex and there are only a handful of integrations. A webchat is provided through an example project[35]. Table 2.1 shows Amazon Lex's features.

[33] https://aws.amazon.com/blogs/aws/amazon-lex-now-generally-available/, visited on 2019-02-02

[34] https://aws.amazon.com/, visited on 2019-02-02

[35] https://github.com/aws-samples/aws-lex-web-ui, visited on 2019-02-02

Name	Dialogflow
URL	https://dialogflow.com/
Launch	2014 (as API.AI)
Supported Languages	20, including English and German
Speech to Text	yes
Versioning / Environments	yes (beta) / yes (beta)
Integrations	Webchat, Facebook Messenger, Google Assistant, Microsoft Cortana, Slack, Kik, LINE, Skype, Spark, Telegram, Twilio, Twitter and Viber
REST API	yes
SDKs	Node.js, Python, Java, Go, Ruby, .NET and PHP
Pricing	standard free plan available, enterprise rate is $0.002 USD per text message

Table 2.2.: Dialogflow fact sheet.

Dialogflow

Dialogflow[36] was released in 2014 as API.AI. In 2016, it was acquired by Google[37] and renamed Dialogflow. Since then, it has become part of *Google Cloud Platform*[38] (GCP). In 2018, Dialogflow introduced versioning and environments (currently in a beta phase). Later, knowledge connectors (beta phase, English only) were added to the platform to simplify the building of FAQs and knowledge base answers, similar to Microsoft's *QnA Maker*[39].

Intents and entities are Dialogflow's core concepts. Intents can be annotated with input and output contexts to build dialog flows and have similar intents in different dialogs. Intents are also invokable by events (for example, if a user starts a conversation on a specific channel). Actions provide additional information for fulfillment webhooks and custom integrations using

[36] https://dialogflow.com/, visited on 2019-02-02

[37] https://www.google.com/, visited on 2019-02-02

[38] https://cloud.google.com/, visited on 2019-02-02

[39] https://azure.microsoft.com/en-us/services/cognitive-services/qna-maker/, visited on 2019-02-02

the REST API or SDKs. Dialogflow supports a wide array of integrations. Responses can be either text, platform specific rich messages or even custom payload. Table 2.2 shows Dialogflow's features and supported languages.

IBM Watson Assistant

Watson[40] is IBM[41]'s platform for AI services and applications. Previously known as *Watson Conversation*[42], *Watson Assistant*[43] is the platform's chatbot service. Watson Assistant provides both NLU capabilities in matching intents and extracting entities as well as building replies for detected intents. Replies in the dialog can feature text, images and quick replies (options). So-called Digressions enable users to ask unrelated questions during a conversation without interrupting the dialog flow. Versioning to manage the update of chatbots is only available in paid plans. Table 2.3 shows Watson Assistant's features and supported languages.

IBM's Watson cloud platform also offers additional services for natural language interaction. *Watson Natural Language Classifier* tries to apply specific classes to given sentences or phrases, similar to intent matching, while Watson *Natural Language Understanding* extracts entities, sentiment, emotion, keywords and so on. There are also services for text-to-speech and speech-to-text transformation.

Microsoft LUIS

Microsoft[44] *LUIS*[45] (Language Understanding) was made generally available in 2017 along with *Azure Bot Service*. LUIS is part of *Cognitive Services* on *Microsoft Azure*[46]. The service is for retrieving intents and entities only,

[40] https://www.ibm.com/watson/, visited on 2019-02-02
[41] https://www.ibm.com/, visited on 2019-02-02
[42] https://www.ibm.com/blogs/watson/2018/03/the-future-of-watson-conversation-watson-assistant/, visited on 2019-02-01
[43] https://www.ibm.com/cloud/watson-assistant/, visited on 2019-02-02
[44] https://www.microsoft.com/, visited on 2019-02-02
[45] https://www.luis.ai/, visited on 2019-02-02
[46] https://azure.microsoft.com/, visited on 2019-02-02

Name	IBM Watson Assistant
URL	`https://www.ibm.com/cloud/watson-assistant/`
Launch	2016 (as Watson Conversation)
Supported Languages	12, including English and German
Speech to Text	only with other services
Versioning / Environments	yes (paid) / no
Integrations	Facebook, Messenger, Slack, Wordpress (via plugin)
REST API	yes
SDKs	Node.js, Python, Swift, Java, Android, Unity, Ruby, Go and .NET
Pricing	limited free plan available, standard rate is $0.0025 USD per text message

Table 2.3.: IBM Watson fact sheet.

as Microsoft provides a sophisticated chatbot framework in *Bot Builder*[47] (see section 2.6) to handle conversations. Apart from training the machine learning algorithm, developers can also define *patterns* to improve their model. Table 2.4 shows the platform's features and supported languages.

Wit.ai

Wit.ai[48] was founded in 2013 and acquired by Facebook[49] in early 2015[50]. Table 2.5 shows Wit.ai's features and supported languages.

The intent matching works similar to Dialogflow's. Developers define certain intents as well as entities and provide examples. Requests are then analyzed and the system outputs the most likely matching intent and entities. Until February 2018, Wit.ai offered services called *Bot Engine* and *Stories*. Stories

[47] `https://github.com/Microsoft/BotBuilder`, visited on 2019-02-02
[48] `https://wit.ai`, visited on 2018-10-18
[49] `https://www.facebook.com/`, visited on 2019-02-02
[50] `https://medium.com/wit-ai/wit-ai-is-joining-facebook-deff3745fcf5`, visited on 2018-10-18

Name	Microsoft LUIS
URL	https://www.luis.ai/
Launch	2017
Supported Languages	over 12, including English and German
Speech to Text	only with other services
Versioning / Environments	yes / yes
Integrations	none (only in conjunction with Azure Bot Service)
REST API	yes
SDKs	Android, Java, Node.js, Python, Ruby, PHP, and .NET
Pricing	limited free plan available, standard rate is $1.5 USD per 1,000 transactions

Table 2.4.: Microsoft LUIS fact sheet.

Name	Wit.ai
URL	https://wit.ai
Launch	2013
Supported Languages	over 70, including English and German
Speech to Text	no
Versioning / Environments	no / no
Integrations	Facebook Messenger
REST API	yes
SDKs	Node.js, Python and Ruby
Pricing	free, including commercial use

Table 2.5.: Wit.ai fact sheet.

let users build whole conversations with example inputs and bot responses to train Bot Engine (basically their machine learning black box). This feature has been deprecated and since then, Wit.ai has been offering its intent matching only[51].

SAP Conversational AI

SAP[52] *Conversational API*[53], formerly known as *Recast.AI* (founded in 2015), is another NLU service specifically targeted at chatbots. Besides the usual concepts of intents and entities, it also features a sentiment analysis for some of its 50 supported languages. *Bot Builder* is the platform's management tool for building dialogs, defining triggers and dispatching actions (including webhooks and replies). Integrations for several major platforms, including Facebook Messenger, Slack and Twitter, as well as a webchat with rich messages are available through *Bot Connector*. Analytics and statistics for Bot Builder are also provided. Table 2.6 shows the features and supported languages.

Others

This section covers a selection of the most popular NLU services. There are, however, several other services and frameworks available — for example *RASA NLU*[54], an open-source NLU library written in Python.

2.5. Messaging Platforms With Bot Support

Chatbots provide a conversational interface, but they still need a platform and clients for the user to access them. For text-based bots, these could

[51]`https://medium.com/wit-ai/launching-built-in-nlp-for-messenger-and-sunsetting-bot-engine-beta-46e9038869a5`, visited on 2018-10-18
[52]`https://www.sap.com/`, visited on 2019-02-02
[53]`https://cai.tools.sap/`, visited on 2019-02-02
[54]`https://rasa.com`, visited on 2019-02-20

Name	SAP Conversational AI
URL	https://cai.tools.sap/
Launch	2015 (as Recast.AI)
Supported Languages	50, including English and German
Speech to Text	no
Versioning / Environments	yes / no
Integrations	Webchat, Facebook Messenger, Slack, Kik, LINE, Skype, Telegram, Twilio, Twitch, Twitter and others
REST API	yes
SDKs	Android, iOS, Node.js, PHP, Python and Ruby
Pricing	free for personal use, enterprise plan on request

Table 2.6.: SAP Conversational AI fact sheet.

be messaging platforms — for example, Facebook Messenger or Slack. They provide a messaging infrastructure and a GUI. The infrastructure is responsible for handling the transfer of messages and informing the chatbot and user about specific events such as reading receipts or people joining a chat room. Different messengers allow the usage of different types of messages. For example, some utilize quick replies (buttons with pre-defined responses), carousels and lists, sometimes refered to as *rich cards* or *message templates*. The GUI defines how these messages look and, therefore, how data is visualized. Voice-based chatbots often use their own devices such as Amazon's Echo devices for Alexa or iOS smartphones and tablets for Apple's Siri.

According to a survey by Mindbowser and ChatbotsJournal.com (2017), most businesses are interested in building chatbots for Facebook, their own website and Slack. The platforms usually differ in popularity (see Table 2.7) and audience. These differences include location, age, gender, devices and use cases. In this section, we focus on a few of the most prominent services in Europe and North America. There are several other popular social media

Messaging Platform	Monthly Active Users (in millions)
WhatsApp	1,500
Facebook Messenger	1,300
Weixin / Wechat	1,083
Twitter	326
Skype	300
Telegram	200
LINE	194

Table 2.7.: Monthly active users of popular messaging platforms with bot support according to Hootsuite and We Are Social (2019) and Nadella (2016a) and https://telegram.org/blog/200-million (visited on 2019-02-13).

and messaging platforms with bot support such as *Kik*[55], *Viber*[56] and *LINE*[57]. Listing and describing all of them would exceed the scope of this thesis. (Hootsuite and We Are Social, 2019, pp. 83-140)

WhatsApp

With 1,500 million monthly active users, the Facebook[58]-owned service *WhatsApp*[59] is the single most popular messaging app in the world (Hootsuite and We Are Social, 2019, pp. 81). Users must provide a phone number in order to use WhatsApp. Clients are available for Android, iOS and Windows Phone smartphones, internet browsers (web app) as well as Windows and macOS.

An *Application Programming Interface* (API) is only available for businesses via the *WhatsApp Business API*[60], which is in limited public preview at the time of writing (2019-02-20).

[55] https://www.kik.com/, visited on 2019-02-20
[56] https://www.viber.com/, visited on 2019-02-20
[57] https://line.me/, visited on 2019-02-20
[58] https://www.facebook.com/, visited on 2019-02-02
[59] https://www.whatsapp.com/, visited on 2019-02-20
[60] https://developers.facebook.com/docs/whatsapp, visited on 2019-02-20

Facebook Messenger

Facebook Messenger[61] is Facebook's messaging platform for mobile and desktop devices. It had around 1.3 billion users[62] and over 200,000 bots[63] in 2017, making it one of the biggest consumer chat platforms worldwide (Hootsuite and We Are Social, 2019, p. 81). In 2018, the platform surpassed 300,000 monthly active bots[64].

The platform offers documentation[65] of all supported message types, events and more. Messenger provides quick replies, media (image and video) messages and so-called templates for lists, carousels, buttons, receipts and airline-specific elements (for example boarding passes). Messenger also integrates webviews to show web content inside the messenger's interface. Another interface element are persistent menus, which provide a traditional user experience known from apps or websites.

In order to hook a bot to Facebook Messenger, a *Facebook App* has to be created on the platform. This app can then be linked to specific Facebook pages (for example of a business) and receives webhooks on messaging events to handle the conversation. Wit.ai (section 2.4) can be directly integrated from within the administration page of the app.

Clients exist for web browsers (web app), Android and iOS. Facebook also provides a *customer chat plugin*[66] (in beta status at the time of writing, 2019-02-20) for the integration of a chat window on a website.

[61] https://www.messenger.com/, visited on 2019-02-02

[62] VentureBeat, 2017.

[63] https://www.facebook.com/business/news/a-look-back-on-messenger-platform-in-2017/, visited on 2019-02-02

[64] VentureBeat, 2018.

[65] https://developers.facebook.com/docs/messenger-platform, visited on 2019-02-04

[66] https://developers.facebook.com/docs/messenger-platform/discovery/customer-chat-plugin, visited on 2019-02-20

Custom / Webchat

Custom-built platforms could be any application handling text or voice chat with a chatbot. While Facebook Messenger and others have incredibly large user bases, an account is necessary to interact with the chatbot. In addition, messages and data are routed over the platform's network. The platform sets limits and defines the possible message formats, while a custom implementation could establish a connection directly between the user and the chatbot, without the need for third-party services. It would also be able to feature custom message types and integrate into existing web and mobile applications. As shown by Mindbowser and ChatbotsJournal.com (2017), businesses are highly interested in using a chatbot on their own websites, which entail the previously mentioned advantages.

While custom integrations enable a high degree of customization, they do not provide the millions of potential users of existing ecosystems. Developers must also implement their own infrastructure and clients, if existing chat libraries do not suffice — for example due to their limited amounts of message types.

Slack

Slack[67] is a messenger for teams and workplaces with client apps for all major mobile and desktop platforms as well as web browsers. According to their own website, Slack has over 10 million daily active users[68] and has experienced immense growth since its launch in 2013. Slack hosts its own API[69] for building bots and apps as well as a public directory[70] with more than 1,500 entries.

Apart from one-on-one communication, Slack provides channels (chat rooms) for conversations with several users. Chatbots can utilize this and

[67]https://slack.com, visited on 2019-02-04

[68]https://slackhq.com/slack-has-10-million-daily-active-users, visited on 2019-02-04

[69]https://api.slack.com/, visited on 2019-02-04

[70]https://codefluegel.slack.com/apps, visited on 2019-02-04

provide services to a single user as well as to a whole group of users. Slack's UI elements include text, images, lists and buttons but also text input fields and select elements for interactive forms and dialogs.

Skype

Skype[71], a messaging application with support for text and video messages as well as voice calls, started in 2003 and was acquired by Microsoft in 2011. The platform's bot support was introduced five years later at the *Build 2016* conference. Skype had over 300 million monthly "connected" users at that time (Nadella, 2016a).

Bot development[72] is intertwined with *Microsoft Bot Framework* (section 2.6) and works in one-on-one as well as group conversations. A web chat named *Web Control*[73] with Skype branding is also provided.

WeChat (Weixin)

WeChat[74] — or *Weixin* in China — is one of the largest messaging platforms on the market (see Table 2.7). It is mostly used in China (Hootsuite and We Are Social, 2019, p. 83) and provides its own ecosystem for apps and bots. WeChat plays a central role in Chinese everyday life, as businesses tend to use service accounts and chatbots for customer interaction. Payments can be handled directly in the app with *WeChat Pay* as well. (Chatbots Magazine, 2017)

APIs and SDKs for developing applications are first and foremost targeted at Chinese businesses, with most of them being only available for specific regions or languages[75]. Clients are available for iOS, Android, Windows and macOS. A web app for browsers is also provided.

[71] `https://www.skype.com`, visited on 2019-02-19
[72] `https://dev.skype.com/bots`, visited on 2019-02-19
[73] `https://dev.skype.com/webcontrol`, visited on 2019-02-20
[74] `https://www.wechat.com`, visited on 2019-02-19
[75] `https://open.wechat.com`, visited on 2019-02-20

Telegram

Telegram[76] launched in 2013 and was created by the founders of Russian social media platform *VKontakte*. By 2018, the messaging platform had already acquired 200 million monthly active users. Telegram offers clients for Android, iOS, Windows Phone, macOS and Windows, as well as web browsers.

APIs[77] to support bots on the platform have been added in 2015[78]. The bot APIs provide a variety of features, including payments, games, custom keyboards, quick replies (*inline keyboards*) and commands known from chat systems like *IRC* (Internet Relay Chat) and Slack (section 2.5) — typically invoked by a slash (/), for example */help*. Registering a bot is done in a unique way, namely by issuing a */newbot*-command on the Telegram bot *BotFather*[79].

Twitter

Twitter[80] is a social media and networking platform featuring short user posts known as *tweets*. Beside those tweets, the platform also offers direct messaging and is used by around 326 million users monthly. APIs[81] to cover both tweets and direct messages are publicly available.

Bots on Twitter gained attention in 2016 with Microsoft's infamous *Tay*, a chatbot targeted at 18- to 20-year-old Americans in an attempt to bring artificial intelligence to the platform. Tay was designed to interact with other people on the social media platform. In less than two days, Tay was trained with political and racist tweets by other users to formulate its own inflammatory postings and was taken offline shortly after. (The Guardian, 2016)

[76] `https://telegram.org`, visited on 2019-02-20
[77] `https://core.telegram.org/bots`, visited on 2019-02-20
[78] `https://telegram.org/blog/bot-revolution`, visited on 2019-02-20
[79] `https://telegram.me/botfather`, visited on 2019-02-20
[80] `https://www.twitter.com`, visited on 2019-02-20
[81] `https://developer.twitter.com/`, visited on 2019-02-20

Email and SMS

While messaging apps like Facebook Messenger are relatively new, traditional communication platforms like *email* and *SMS* (Short Message Service) are still relevant and sometimes even preferred over messaging apps (Ubisend, 2016). Bot integration can be achieved in different ways. *Microsoft Bot Framework* (section 2.6) enables developers to connect a chatbot to an *Office 365* email account. However, arbitrary email accounts can also be used, if they can be accessed — for example with a protocol like Internet Message Access Protocol (IMAP). Handling SMS requires a device or gateway capable of sending or receiving such messages, such as *Twilio*[82].

Others

In contrast to the other mentioned platforms, *Amazon Alexa*[83] is not a messaging platform but a voice assistant featured on devices such as Amazon's *Fire* tablets[84] and the speaker *Echo*[85]. Although it does not provide the ability to host other chatbots, it is a complex chatbot on its own and enhancing its feature set is enabled by *Alexa Skills*.

2.6. Bot Frameworks

Bot frameworks help developers in creating chatbots by providing methods and concepts to manage dialogs. This includes context and session handling, which work as ways to identify the user and add memory capabilities to the chatbot. Passing messages to NLU services is also part of most frameworks

[82]`https://www.twilio.com`, visited on 2019-02-20

[83]`https://developer.amazon.com/alexa`, visited on 2019-02-04

[84]`https://press.aboutamazon.com/news-releases/news-release-details/introducing-all-new-amazon-fire-7-and-fire-hd-8-amazon-alexa`, visited on 2019-02-04

[85]`https://press.aboutamazon.com/news-releases/news-release-details/amazon-introduces-alexa-and-echo-echo-dot-and-echo-plus-canada`, visited on 2019-02-04

through *middleware*. Frameworks also try to abstract connections to bot platforms, so developers do not have to deal with the specifics of each platform in most cases. This is especially helpful, if several platforms with different message formats need to be supported. Another advantage of bot development frameworks compared to service platforms is the usage of professional version control systems like Git[86].

This section covers some of the more popular bot creation frameworks. There are others, but examining all of them would be beyond the scope of this thesis.

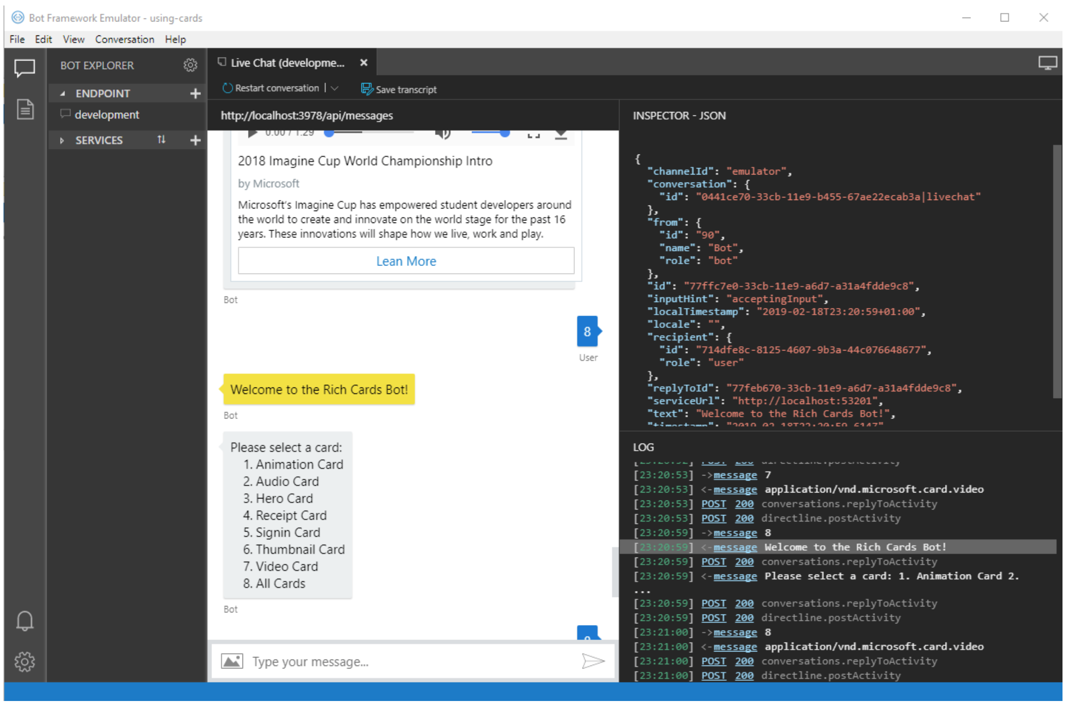
Microsoft Bot Framework

Microsoft Bot Framework is a collective term for Microsoft's *Bot Builder*[87] and *Azure Bot Service*[88]. Although Bot Builder is the actual framework, both are mentioned here, as the SDK is strongly connected to the Azure services.

Bot Builder comes with methods to manage dialogs and user and conversation state. Microsoft also distributes a chatbot emulator (Figure 2.5) for local testing of conversations[89]. Support for Natural Language Understanding is available for Microsoft LUIS by default. Other services have to be added with *middleware*.

The Bot Builder SDK supports C# and JavaScript. Preview builds for Java and Python are also available. The documentation[90] is extensive, and samples for many different features and use cases are provided[91]. Integrations to major messaging platforms Facebook Messenger, Slack, Kik, LINE, Skype, Telegram, Twilio and others are available through Azure Bot Service. Microsoft maintains a feature-rich and highly customizable webchat[92] based on

[86] `https://git-scm.com/`, visited on 2019-03-08
[87] `https://github.com/Microsoft/BotBuilder`, visited on 2019-03-05
[88] `https://azure.microsoft.com/en-us/services/bot-service/`, visited on 2019-03-06
[89] `https://github.com/Microsoft/BotFramework-Emulator`, visited on 2019-03-08
[90] `https://docs.microsoft.com/en-us/azure/bot-service/`, visited on 2019-03-06
[91] `https://github.com/Microsoft/BotBuilder-Samples/`, visited on 2019-03-08
[92] `https://github.com/microsoft/botframework-webchat`, visited on 2019-03-06



2018 Imagine Cup World Championship Intro

by Microsoft

Microsoft's Imagine Cup has empowered student developers around the world to create and innovate on the world stage for the past 16 years. These innovations will shape how we live, work and play.

Welcome to the Rich Cards Bot!

Please select a card:
1. Animation Card
2. Audio Card
3. Hero Card
4. Receipt Card
5. Signin Card
6. Thumbnail Card
7. Video Card
8. All Cards

INSPECTOR - JSON

```
{
  "channelId": "emulator",
  "conversation": {
    "id": "0441ce70-33cb-11e9-b455-67ae22ecab3a|livechat"
  },
  "from": {
    "id": "90",
    "name": "Bot",
    "role": "bot"
  },
  "id": "77ffc7e0-33cb-11e9-a6d7-a31a4fdde9c8",
  "inputHint": "acceptingInput",
  "localTimestamp": "2019-02-18T23:20:59+01:00",
  "locale": "",
  "recipient": {
    "id": "714dfe8c-8125-4607-9b3a-44c076648677",
    "role": "user"
  },
  "replyToId": "77feb670-33cb-11e9-a6d7-a31a4fdde9c8",
  "serviceUrl": "http://localhost:53201",
  "text": "Welcome to the Rich Cards Bot!",
```

LOG

```
[23:20:53] ->message 7
[23:20:53] <-message application/vnd.microsoft.card.video
[23:20:53] POST 200 conversations.replyToActivity
[23:20:53] POST 200 directline.postActivity
[23:20:59] ->message 8
[23:20:59] <-message Welcome to the Rich Cards Bot!
[23:20:59] POST 200 conversations.replyToActivity
[23:20:59] <-message Please select a card: 1. Animation Card 2.
...
[23:20:59] POST 200 conversations.replyToActivity
[23:20:59] POST 200 directline.postActivity
[23:21:00] ->message 8
[23:21:00] <-message application/vnd.microsoft.card.video
[23:21:00] POST 200 conversations.replyToActivity
[23:21:00] POST 200 directline.postActivity
```

Figure 2.5.: Microsoft's Bot Framework comes with its own emulator for testing and debugging.

React[93]. Using Bot Builder is free, hosting on Azure and registering channels to connect to messaging platforms is subject to Azure's pricing[94].

Botkit

Botkit[95] is a popular open source bot framework based on *Node.js*[96]. In November 2018, Microsoft announced the acquisition of *XOXCO*[97], the company behind Botkit.

[93] https://reactjs.org/, visited on 2019-03-06
[94] https://azure.microsoft.com/en-us/pricing/details/bot-service/, visited on 2019-03-06
[95] http://botkit.ai, visited on 2019-03-05
[96] https://nodejs.org/, visited on 2019-03-05
[97] http://www.xoxco.com/, visited on 2019-03-05

The framework uses a modular approach with so-called *skills* and *middleware*. Skills are JavaScript modules adding features to a chatbot. For example, when a sports bot "hears" something like "Austria vs. Germany", it's *match-result*-skill could call an API endpoint to fetch the result of a match and send it to the user. Natural Language Understanding has to be implemented through the use of middleware or with community add-ons. Support for Microsoft LUIS (section 2.4) is integrated into *Botkit CMS*[98]. Botkit CMS is a tool to design and manage dialogs using a visual approach similar to the service platforms described in section 2.7.

A disadvantage of Botkit is its limitation to only support one messaging platform per bot. An integration with Microsoft Bot Framework has been added to overcome this issue[99], but adds reliance on Microsoft's services and increases server hops resulting in longer sending times of messages between client and server. Focusing on one platform per bot allows for a deeper integration, however, as platform-specific features can be used more efficiently.

Because of Botkit's architecture as a Node.js library, bots run as Node.js-apps and have to be built in the same language using JavaScript. Botkit supports integration with Slack, Cisco Webex and Jabber, Microsoft Teams, Facebook Messenger, Twilio SMS and IPM, Google Hangouts as well as Microsoft Bot Framework. A simple (embeddable) webchat with quick replies is included in the library via project templates. These boilerplates are available for most of the platforms[100].

BotMan

BotMan[101] stands out as one of the few popular PHP bot frameworks. BotMan also comes as a project template for the popular PHP framework Laravel[102], named *BotMan Studio*.

[98] `https://github.com/howdyai/botkit-cms`, visited on 2019-03-06
[99] `https://botkit.ai/docs/readme-botframework.html`, visited on 2019-03-05
[100] `https://github.com/search?q=org%3Ahowdyai+starter`, visited on 2019-03-05
[101] `https://botman.io/`, visited on 2019-03-06
[102] `https://laravel.com/`, visited on 2019-03-06

NLU with Dialogflow (section 2.4) works out of the box. Other NLU services can be implemented via middleware. Data can be stored in a built-in storage system. The documentation[103] of BotMan is comprehensive and BotMan Studio even includes a testing framework based on PHPUnit[104].

Integrations (*drivers*) for several major platforms such as Facebook Messenger, Slack, Telegram, Twilio, WeChat and others, including an embeddable webchat based on Preact[105], are provided out of the box. As with Botkit (section 2.6), integration with Microsoft Bot Framework (section 2.6) is also possible.

Botpress

Botpress[106] is a bot platform and framework labelled as a "lightweight, fast and flexible on-premise bot building platform"[107] and ships with a custom NLU engine, an administration dashboard / visual editor (similar to the sites mentioned in section 2.7), analytics and a chat emulator. Botpress describes its approach of handling content as a Content Management System (CMS).

Botpress has recently been rewritten in TypeScript and currently only supports webchat. Previous support for Facebook Messenger, Telegram, Microsoft Bot Framework and Twitter still needs to be migrated from version 10 to 11[108]. External NLU providers such as Dialogflow (section 2.4) can be enabled through *hooks* and *middleware*, but they do not integrate into the graphical interface. An SDK to extend the platform via *Botpress Modules* is available.

[103] `https://botman.io/2.0/welcome`, visited on 2019-03-09
[104] `https://phpunit.de/`, visited on 2019-03-09
[105] `https://preactjs.com`, visited on 2019-03-09
[106] `https://botpress.io`, visited on 2019-03-06
[107] `https://botpress.io/docs/introduction/`, visited on 2019-03-09
[108] `https://help.botpress.io/t/what-happened-to-channels-in-11-5-1/1109/2`, visited on 2019-03-10

Others

Botmaster.ai[109] is a Node.js framework capable of running a single bot with multiple messaging channels. Developers can use Facebook Messenger's message format and the framework translates messages to each other platform. The framework is easy to use but development seems to have slowed down in early 2018[110].

Rasa Core[111], a bot framework, and *Rasa NLU*, a NLU library, are both part of the *Rasa Stack*, a collection of open-source tools for building chatbots. The framework is written in Python and does not rely on third party services for NLU, which is an advantage for sensitive data. Connectors for Facebook Messenger, Slack, Microsoft Bot Framework, Rocket.Chat, Mattermost, Telegram and others are provided. A webchat channel using Socket.IO[112] or regular HTTP requests via a REST API is also included in the framework.

Hubot[113] is a Node.js framework made by GitHub[114]. Similar to other frameworks, the bot template can be extended with scripts. Scripts can be written in either JavaScript or CoffeeScript. Adapters for shell interaction and Campfire[115] are included, while others, for example Slack and IRC, are provided by the community[116].

bBot[117] is a bot framework sponsored by Rocket.Chat[118], an alternative to Slack (section 2.5). The framework is inspired by Hubot and based on Node.js. It is currently in alpha status and only supports integration with

[109] `http://botmasterai.com`, visited on 2019-03-07
[110] `https://github.com/botmasterai/botmaster/commits/master`, visited on 2019-03-09
[111] `https://rasa.com`, visited on 2019-03-07
[112] `https://socket.io/`, visited on 2019-03-2019
[113] `https://hubot.github.com/`, visited on 2019-03-07
[114] `https://github.com/`, visited on 2019-03-09
[115] `https://basecamp.com/retired/campfire`, visited on 2019-03-10
[116] `https://www.npmjs.com/search?q=hubot%20adapter&ranking=popularity`, visited on 2019-03-10
[117] `http://bbot.chat/`, visited on 2019-03-07
[118] `https://rocket.chat/`, visited on 2019-03-10

Rocket.Chat. Support for Slack, Facebook Messenger, Telegram and others is planned for future releases.

2.7. Bot Service Platforms

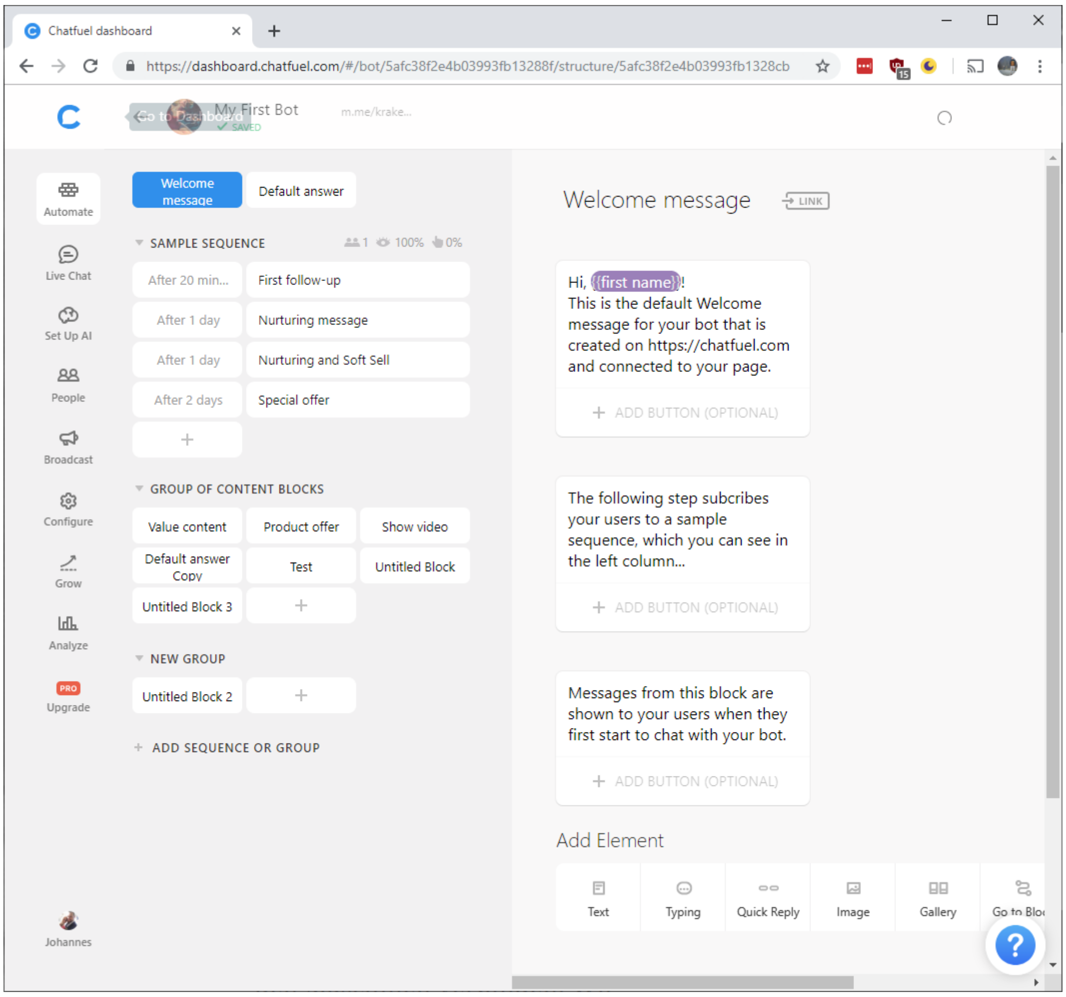
Bot frameworks tend to cater to an audience of developers. Bot service platforms, on the other hand, use a more visual approach with graphical user interfaces for dialog and flow editing to enable non-technical users to build chatbots. The easier handling often comes with less features and control over the bot, compared to frameworks and custom implementations. It is also more difficult to get the most out of every messaging platform, as the integrations are managed by those service platforms.

This section covers some of the more popular bot service platforms. There are others, but examining all of them would be beyond the scope of this thesis.

Chatfuel[119] is solely targeting chatbot creation for Facebook Messenger. It provides a clear user interface (see Figure 2.6) and many building blocks, including calls to other REST APIs. Chatfuel supports many languages and incorporates most of Facebook Messenger's features such as persistent menus, broadcasting and handover (to a live agent). Bot replies are triggered using Chatfuel's "AI rules", a simple NLU approach where input messages are mapped to an output (reply). Simple analytics about users and user interaction are also available. A free plan with limited features is available, the *Pro* plan starts at $15 USD per month[120].

Snatchbot[121] provides a webchat and integrates some of the most popular messaging channels with Facebook Messenger, Slack, E-Mail, Skype, Twilio and LINE. NLU/NLP with intent and entity matching is available and several languages are supported. Snatchbot is free to use, a *Pro* plan with extended support and removal of branding is available for $30 USD per month.

[119]`https://chatfuel.com/`, visited on 2019-03-10
[120]`https://chatfuel.com/pricing.html`, visited on 2019-03-10
[121]`https://snatchbot.me/`, visited on 2019-03-10



Hi, {{first name}}!
This is the default Welcome message for your bot that is created on https://chatfuel.com and connected to your page.

The following step subcribes your users to a sample sequence, which you can see in the left column...

Messages from this block are shown to your users when they first start to chat with your bot.

Figure 2.6.: Chatfuel provides a simple GUI to build dialogs.

Pandorabots[122] uses an XML-based language called *Artificial Intelligence Markup Language* (AIML)[123] to build bots. Users can edit the AIML-files directly or use a *What You See Is What You Get* (WYSIWYG)-like editor. Mitsuku[124], a four-time winner of the Loebner Prize Turing Test, is built in AIML and runs on Pandorabots. The platform's free plan is limited to 1,000 interactions a month. Additional messages cost $0.0025 USD each.

Dialogflow is a NLU platform, but provides content management and dialog building as well. See section 2.4 for more details about the platform and its integrations.

[122] https://home.pandorabots.com/, visited on 2019-03-10
[123] http://www.aiml.foundation/, visited on 2019-03-10
[124] http://www.square-bear.co.uk/mitsuku/home.htm, visited on 2019-03-10

3. Theodore, a Chatbot

3.1. Introduction

As part of this master's thesis, a chatbot called *Theodore* was created. The chatbot's main use is to represent a company, namely the Austrian-based software developer *CodeFlügel GmbH*. CodeFlügel is located in Graz, Austria and specializes in developing augmented reality mobile applications. Virtual reality, traditional apps, web apps and custom projects are also part of their offered services. CodeFlügel has about twenty employees consisting of software developers, sales staff and other people from the operative business such as a human resources manager, marketing personnel and the chief executive officer.

Like most companies, CodeFlügel uses a website[1] to represent themselves on the Internet and inform potential interested parties about their services and other kinds of information. Content and features of the website include, but are not limited to, information about what CodeFlügel offers, a blog and a list of open positions. CodeFlügel also engages in social media platforms, one of which is Facebook[2]. Facebook offers the possibility to directly contact a company via their own Facebook Messenger[3]. Messenger is a chat and messaging platform providing APIs to build third-party tools and — more importantly — chatbots. See section 2.5 for more information.

Theodore was designed to reproduce most of the website's features and to work with Facebook Messenger as well as a custom webchat, which could

[1] `https://www.codefluegel.com`, visited on 2018-10-28
[2] `https://www.facebook.com`, visited on 2019-01-20
[3] `https://developers.facebook.com/docs/messenger-platform`, visited on 2019-01-20

be embedded on a website. Existing APIs and content should be (re-)used as much as possible.

3.2. Target Audience

In order to design a company chatbot, it is imperative to recognize and understand the target audience of CodeFlügel. The website users can be divided into three (3) groups:

- Type A – Clients
- Type B – Potential Employees
- Type C – Blog Readers and Everyone Else

In addition to defining the target audience and the different user groups, a representative persona was created for each group. Personas help to put oneself in the shoes of the users and to make defining scenarios for the various user groups easier.

Type A – Clients

A client or potential client is someone interested in purchasing a product or service from the company — typically a mobile application or a website. Some of them might have a very specific request or idea, while others are interested in consultation and guidance. They most likely want to know what the company is offering, their level of expertise in specific technologies and how to contact them. The technological expertise of the customers varies greatly and ranges from beginners to experts. For clients, the chatbot would not only offer information, but also function as a demonstration of the company's previous work on chatbots.

Persona

Name: Julia David
Age: 36

Occupation: Chief Digital Officer for a large company
Description: Julia is in charge of taking the right steps towards the digital age in a large company producing industrial machinery. She is task-oriented and focus-driven. Julia is proud of her achievements and her career, as she had to overcome many obstacles as a woman in such a male-dominated field.

Type B – Potential Employees

Potential employees are usually students with little job experience, but a technological background or — less likely — people interested in marketing, sales or human resources. They could use the website or chatbot for general information about the company or to inform themselves about job openings or projects the company has worked on.

Persona

Name: Peter Müller
Age: 24
Occupation: Student
Description: Peter is a quiet but tech-savvy student from Graz, Austria. He is close to finishing his Master in Computer Science and is looking for his entry job into the IT world. Peter likes to tinker around with smart devices and has experience in Android development. He is rather introverted and shy when it comes to face-to-face communication but opens up when using modern media channels.

Type C – Blog Readers and Everyone Else

This group of users is probably not looking for anything specific about the company, but rather for one of the varying blog posts or general information about the technology used at the company. They most likely found the website / chatbot via a search engine such as Google[4] or a social media channel such as Facebook. The chatbot provides little benefit for such users, besides interacting with the user and possibly generating interest and publicity.

Persona

Name: Alisa Blum
Age: 28
Occupation: Marketing Assistant
Description: Alisa lives in Berlin, as she loves getting to know new people and the near endless possibilities for social activities of such a large city. She is an open-minded person and always looking for new creative ways to engage with customers. Although she is not so technically well-versed, she is very interested in marketing topics and the possibilities offered by apps and augmented reality.

3.3. Dialog-Design

With such a target audience in mind, the chatbot's requirements and a feature set were defined in consultation with CodeFlügel. The chatbot should feature almost all of the website's content. Since the interaction is different from browsing a website, some additional design decisions have to be made. For example, a help function and a fallback dialog help to enhance the user experience.

Greeting

Theodore must be able to greet users in order to help them understand that they are chatting with the company chatbot of CodeFlügel. Thus, the chatbot first introduces himself and then shows some examples about what it can help the user with.

[4] `https://www.google.com`, visited on 2019-01-20

About CodeFlügel

The chatbot should be able to provide a short explanation about what CodeFlügel does and which services they offer. Sample utterances are:

- "Was macht CodeFlügel?" ("What does CodeFlügel do?")
- "Über CodeFlügel" ("About CodeFlügel")
- "Wer seid ihr?" ("Who are you?")

About Products and Service

Products and services should be explained by Theodore. Furthermore, the chatbot should provide information about completed projects. Sample utterances are:

- "Was ist Augmented Reality?" ("What is Augmented Reality?")
- "Was ist AR?" ("What is AR?")
- "Was ist Virtual Reality?" ("What is Virtual Reality?")

Contact Information

The most important contact information should be displayed by the chatbot. This includes the office address, phone number and email address. A link to Google Maps[5] is provided to help users in locating the office. Sample utterances are:

- "Wo findet man euch?" ("Where can I find you?")
- "Kontakt" ("Contact")
- "Wie kann ich euch erreichen?" ("How can I contact you?")

[5] `https://maps.google.com`, visited on 2019-01-20

Open Positions

Theodore should inform users about open positions and provide a way to contact CodeFlügel. In case of no vacancies, an information about speculative applications and where to apply is shown. Sample utterances are:

- "Habt ihr offene Stellen?" ("Do you have vacancies?")
- "Sucht ihr Entwickler?" ("Are you looking for developers?")
- "Jobs" ("Jobs")

Newsletter Subscription

CodeFlügel uses Mailchimp[6] as a service to handle their newsletter. The chatbot should provide a way to subscribe to the newsletter by accessing the Mailchimp API[7]. Sample utterances are:

- "Habt ihr einen Newsletter?" ("Do you have a newsletter?")
- "Ich möchte mich für den Newsletter anmelden" ("I want to register for your newsletter")
- "Newsletter" ("Newsletter")

Blog

Blog posts are published regularly by CodeFlügel. The website uses Wordpress[8] as its content management system. Theodore should output the most recent blog posts by accessing the website's REST API[9]. Sample utterances are:

- "Blogged ihr?" ("Do you blog?")
- "Habt ihr einen Blog?" ("Do you have a blog?")
- "Blog" ("Blog")

[6]https://mailchimp.com, visited on 2019-01-20
[7]https://developer.mailchimp.com, visited on 2018-07-03
[8]https://wordpress.org, visited on 2019-01-20
[9]https://developer.wordpress.org/rest-api, visited on 2018-04-16

Social Media Links

CodeFlügel's social media handles should be shown and linked to their respective platforms, if a user asks for this information. This includes Facebook, Instagram[10] and Twitter[11]. Sample utterances are:

- "Seid ihr auf Facebook?" ("Are you on Facebook?")
- "Habt ihr Instagram?" ("Are you on Instagram?")
- "Social Media" ("Social Media")

Help

Another important aspect in designing a chatbot is to provide help, if a user gets stuck or does not know what to do. Consequently, if a user asks for help, a short description about the chatbot's abilities should be shown. Sample utterances are:

- "Hilfe" ("Help")

Fallback / Error Handling

The chatbot is made for a specific domain which defines the capabilities. If a user input is outside of this domain or the NLU engine is unable to detect the user's intent, some kind of fallback should be triggered. For example, Theodore is not designed to announce the weather. Thus, if asked whether it is going to rain today, the chatbot should tell the user that he could not understand what the user wants or that it does not know an answer to the specific request. If this happens several times, the chatbot should present a way to get in contact with a real person.

Theodore should feature informal language and a friendly attitude. The chatbot should make clear that it is a program and not pretend to be a real person, which should help to avoid the uncanny valley effect mentioned

[10] `https://www.instagram.com`, visited on 2019-01-20
[11] `https://twitter.com`, visited on 2019-01-20

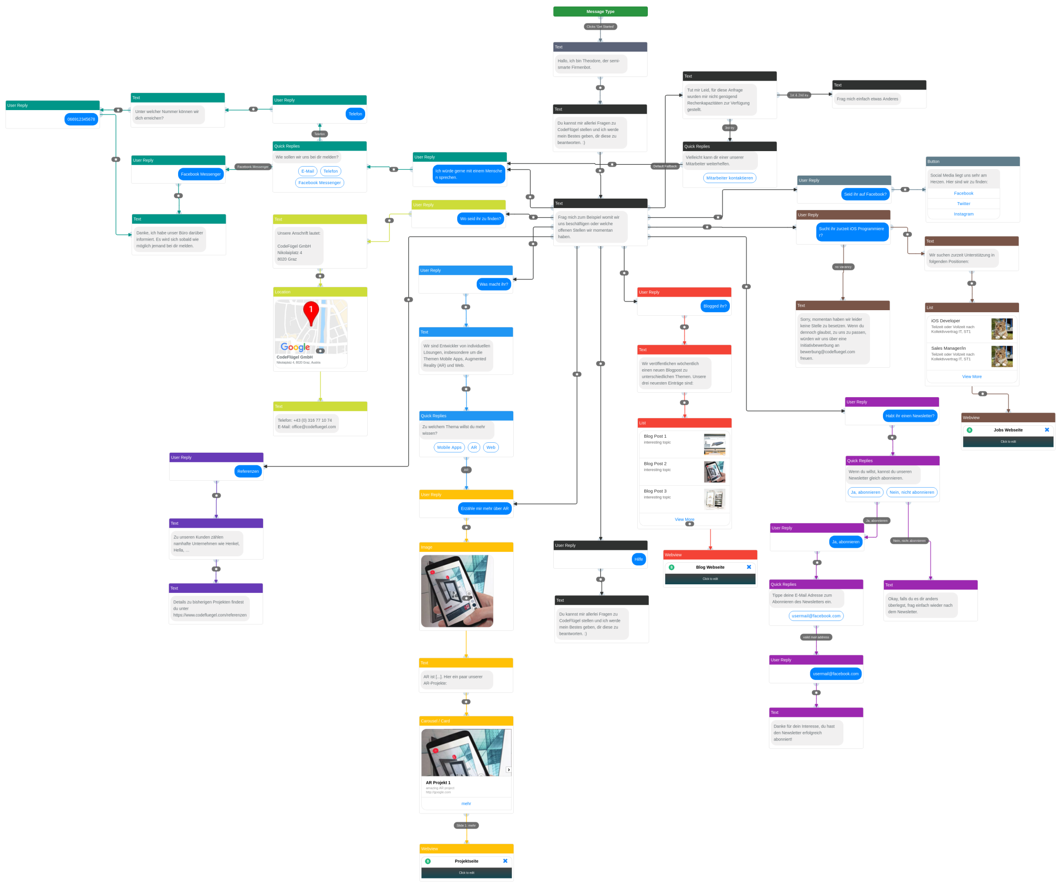


Figure 3.1.: Dialog tree showing all possible dialogs.

in section 2.3. Figure 3.1 shows a tree consisting of all possible dialogs Theodore is able to handle.

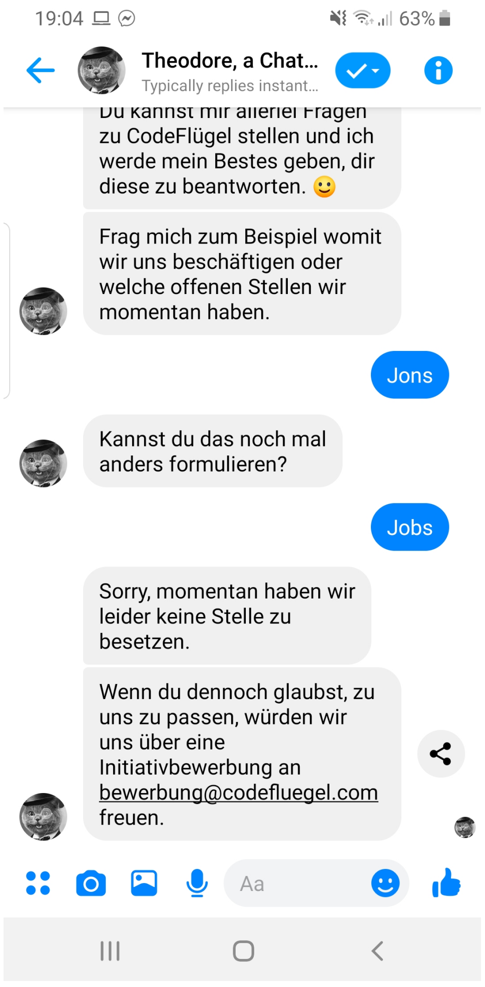
3.4. Platforms

Since CodeFlügel maintains a Facebook page as well as a website, the chatbot should be usable on both platforms. Facebook Messenger is used for the first, while a custom webchat is used for the second. Messenger already provides an existing messaging platform and chat experience. Only the backend needs to be developed for this platform. A webchat, however, also needs its own User Interface (UI) and ways to communicate with the backend.

The same backend is used for both platforms. Facebook provides an API to communicate with Messenger as well as webhooks to handle incoming messages and actions. The webchat uses websockets to offer a fast and responsive chat experience. In order to streamline the development process and to increase the maintainability of the chatbot, Facebook Messenger's message format is used for both platforms and its UI components are recreated for the webchat. These components include simple text (Figure 3.2), Messenger's Generic (Figure 3.3), List (Figure 3.4), Button and Media Templates. Typing indicators (Figure 3.5) to simulate the behavior of typing messages and quick reply buttons (Figure 3.6) were implemented as well. This also makes the different platforms more consistent and increases usability as users are presented with a familiar chat experience.

3.5. Architecture

Instead of using a bot development framework (see section 2.6) to start the development, a custom implementation was chosen. While such frameworks provide functions and templates for quickly achieving first results, they tend to only support subsets of the features offered by some bot platforms. Theodore should also run on both Facebook Messenger and a webchat on the company's website. Botkit (section 2.6) seemed like a good fit on first glance,


19:04

Theodore, a Chat...
Typically replies instant...

Du kannst mir allerlei Fragen zu CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. 🙂

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Jons

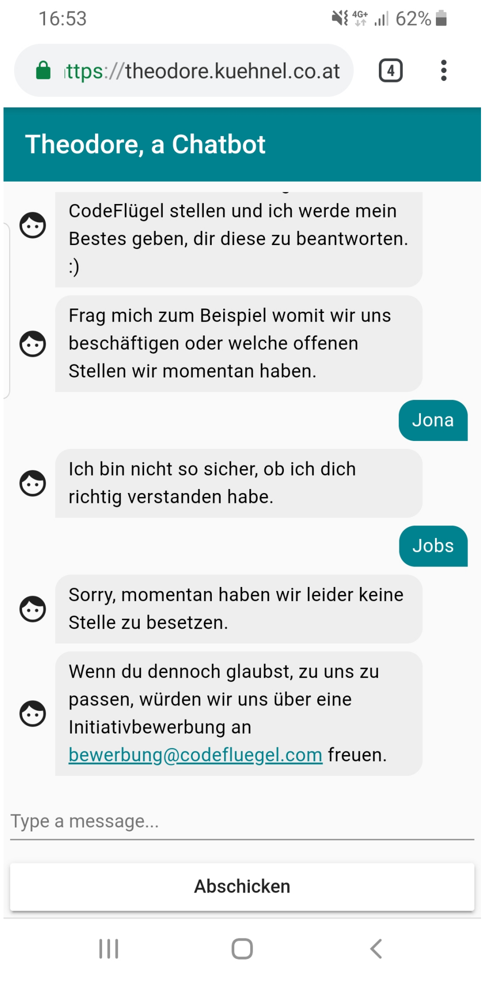
Kannst du das noch mal anders formulieren?

Jobs

Sorry, momentan haben wir leider keine Stelle zu besetzen.

Wenn du dennoch glaubst, zu uns zu passen, würden wir uns über eine Initiativbewerbung an bewerbung@codefluegel.com freuen.

(a) Facebook Messenger


16:53

ttps://theodore.kuehnel.co.at

Theodore, a Chatbot

CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. :)

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Jona

Ich bin nicht so sicher, ob ich dich richtig verstanden habe.

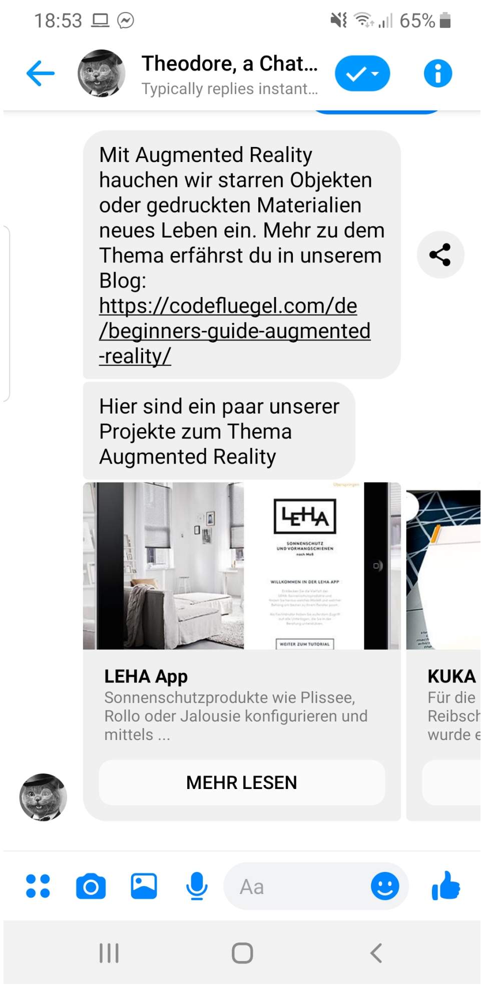
Jobs

Sorry, momentan haben wir leider keine Stelle zu besetzen.

Wenn du dennoch glaubst, zu uns zu passen, würden wir uns über eine Initiativbewerbung an bewerbung@codefluegel.com freuen.

(b) Custom Webchat

Figure 3.2.: Simple text messages in Facebook Messenger (a) and the custom webchat (b).


18:53 65%

Theodore, a Chat...
Typically replies instant...

Mit Augmented Reality hauchen wir starren Objekten oder gedruckten Materialien neues Leben ein. Mehr zu dem Thema erfährst du in unserem Blog: https://codefluegel.com/de/beginners-guide-augmented-reality/

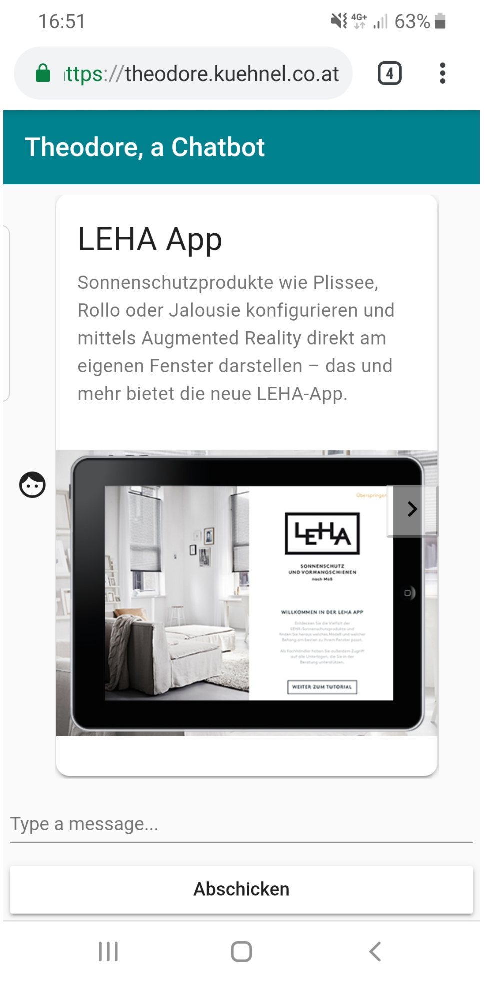
Hier sind ein paar unserer Projekte zum Thema Augmented Reality

LEHA App
Sonnenschutzprodukte wie Plissee, Rollo oder Jalousie konfigurieren und mittels ...

MEHR LESEN

KUKA
Für die Reibsch wurde e

(a) Facebook Messenger


16:51 63%

ttps://theodore.kuehnel.co.at

Theodore, a Chatbot

LEHA App

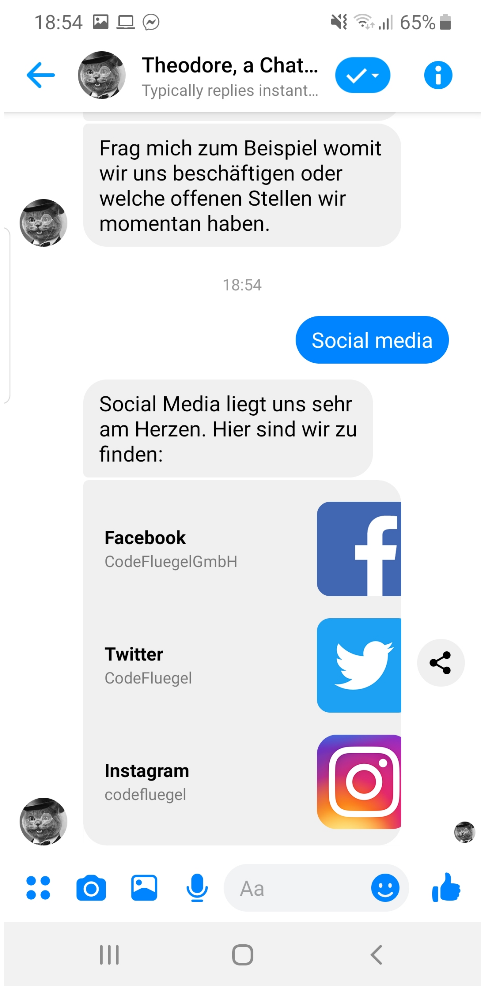
Sonnenschutzprodukte wie Plissee, Rollo oder Jalousie konfigurieren und mittels Augmented Reality direkt am eigenen Fenster darstellen – das und mehr bietet die neue LEHA-App.

Type a message...

Abschicken

(b) Custom Webchat

Figure 3.3.: Generic Templates in Facebook Messenger (a) and the custom webchat (b).


18:54 65%

Theodore, a Chat...
Typically replies instant...

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

18:54

Social media

Social Media liegt uns sehr am Herzen. Hier sind wir zu finden:

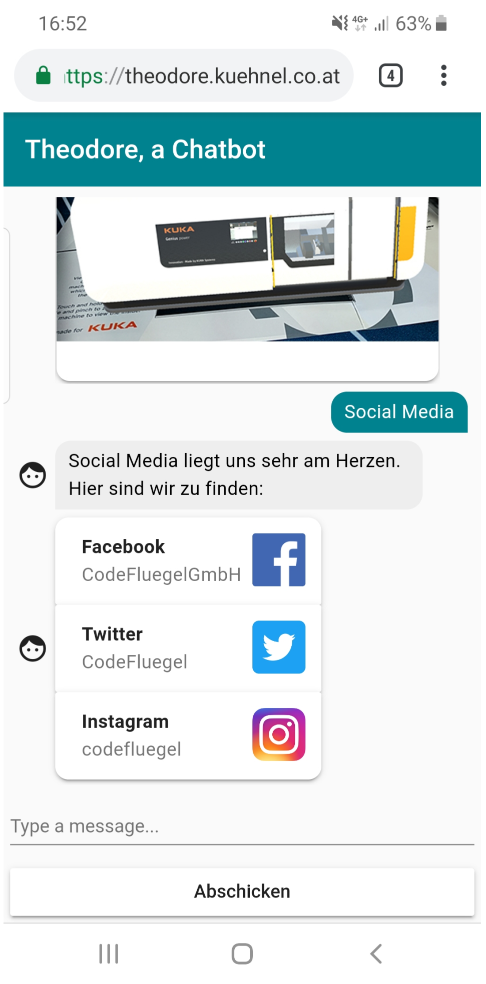
Facebook
CodeFluegelGmbH

Twitter
CodeFluegel

Instagram
codefluegel

Aa

(a) Facebook Messenger


16:52 63%

ttps://theodore.kuehnel.co.at

Theodore, a Chatbot

Social Media

Social Media liegt uns sehr am Herzen. Hier sind wir zu finden:

Facebook
CodeFluegelGmbH

Twitter
CodeFluegel

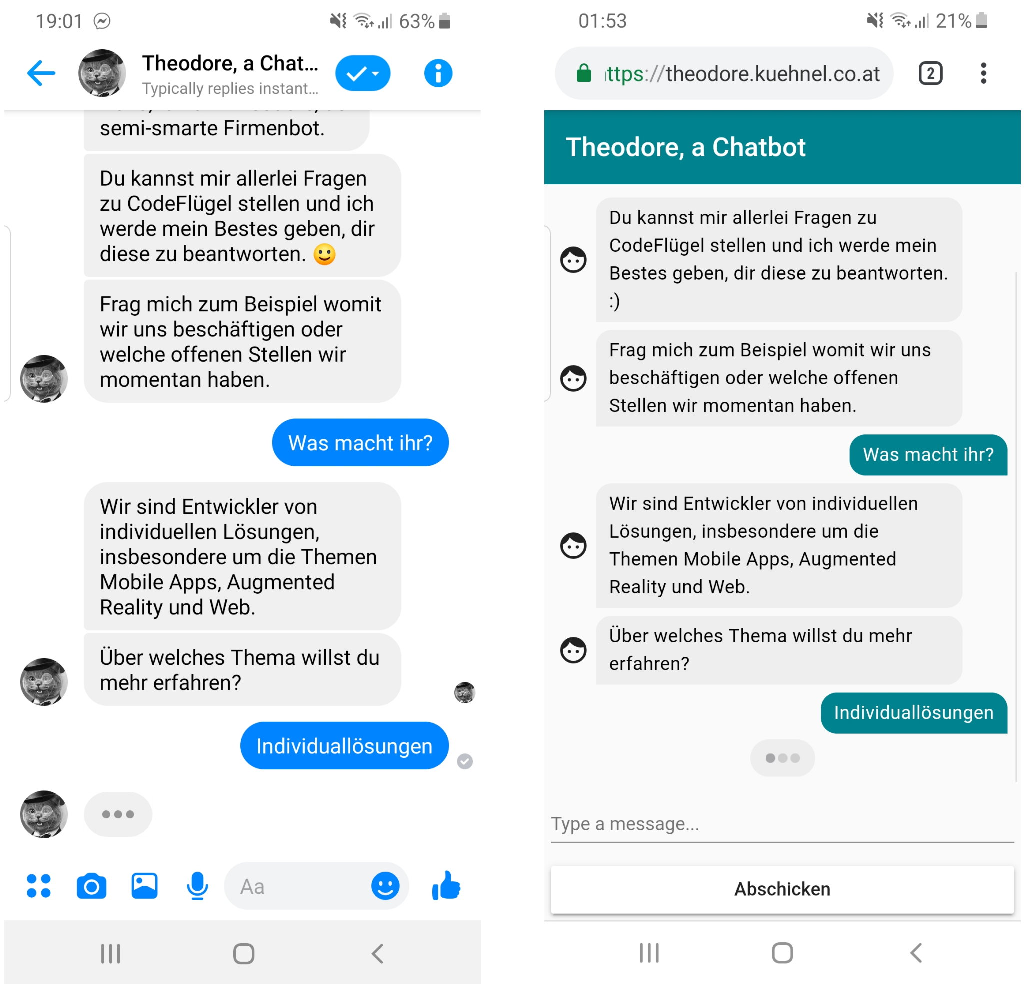
Instagram
codefluegel

Type a message...

Abschicken

(b) Custom Webchat

Figure 3.4.: Lists in Facebook Messenger (a) and the custom webchat (b).



19:01 63%

Theodore, a Chat...
Typically replies instant...

semi-smarte Firmenbot.

Du kannst mir allerlei Fragen zu CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. 🙂

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Was macht ihr?

Wir sind Entwickler von individuellen Lösungen, insbesondere um die Themen Mobile Apps, Augmented Reality und Web.

Über welches Thema willst du mehr erfahren?

Individuallösungen

Aa

01:53 21%

ttps://theodore.kuehnel.co.at

Theodore, a Chatbot

Du kannst mir allerlei Fragen zu CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. :)

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Was macht ihr?

Wir sind Entwickler von individuellen Lösungen, insbesondere um die Themen Mobile Apps, Augmented Reality und Web.

Über welches Thema willst du mehr erfahren?

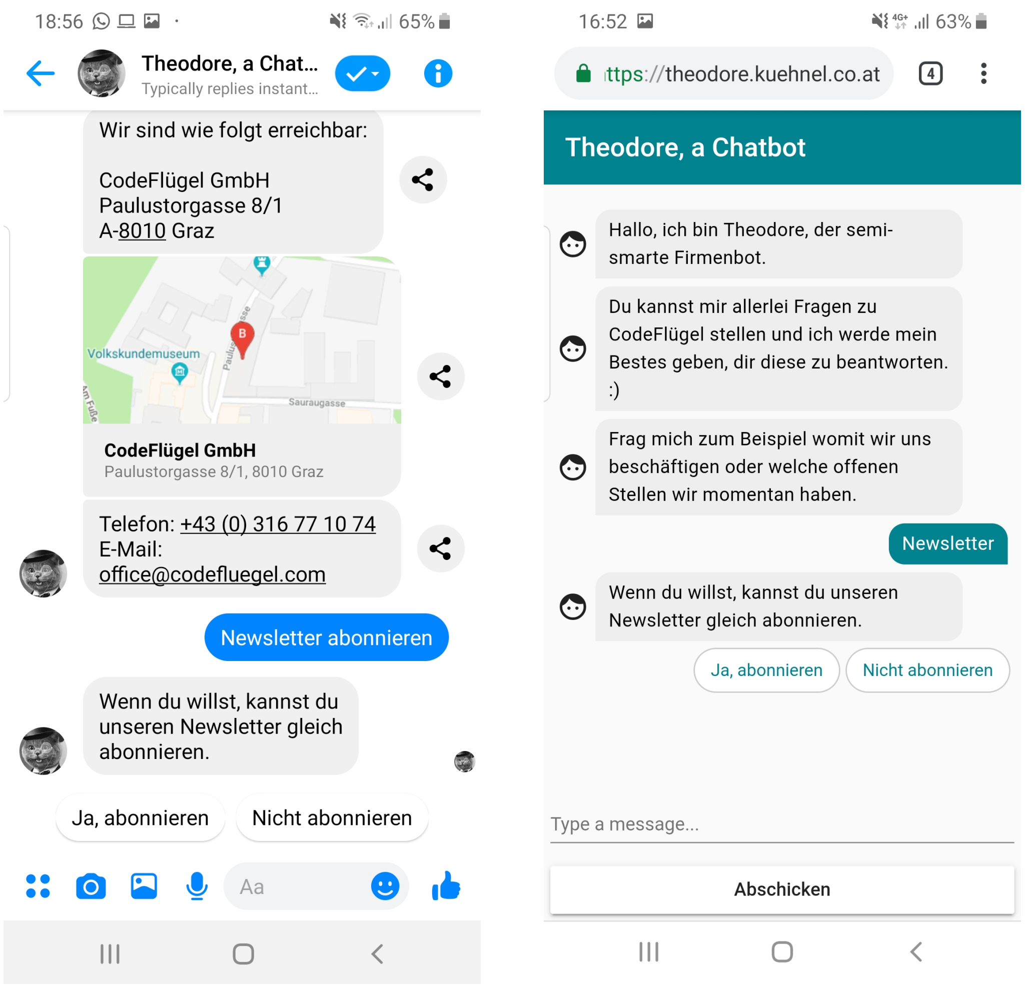
Individuallösungen

Type a message...

Abschicken

(a) Facebook Messenger (b) Custom Webchat

Figure 3.5.: Typing indicators in Facebook Messenger (a) and the custom webchat (b).



Wir sind wie folgt erreichbar:

CodeFlügel GmbH
Paulustorgasse 8/1
A-8010 Graz

CodeFlügel GmbH
Paulustorgasse 8/1, 8010 Graz

Telefon: +43 (0) 316 77 10 74
E-Mail:
office@codefluegel.com

Newsletter abonnieren

Wenn du willst, kannst du unseren Newsletter gleich abonnieren.

Ja, abonnieren | Nicht abonnieren

(a) Facebook Messenger

Hallo, ich bin Theodore, der semi-smarte Firmenbot.

Du kannst mir allerlei Fragen zu CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. :)

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Newsletter

Wenn du willst, kannst du unseren Newsletter gleich abonnieren.

Ja, abonnieren | Nicht abonnieren

(b) Custom Webchat

Figure 3.6.: Quick reply buttons in Facebook Messenger (a) and the custom webchat (b).

but it requires running two separate instances to support both platforms. Microsoft Bot Framework (section 2.6) introduces another reliance on an external service — which in turn adds to the processing time of messages and raises potential privacy concerns in light of the EU's General Data Protection Regulation (GDPR)[12].

Initially, Theodore was planned to work with several different NLU/NLP services such as Google's Dialogflow (previously known as API.AI), Microsoft's LUIS, IBM's Watson and Facebook's Wit.ai (see section 2.4 for more information). During development, this approach proved rather difficult, as these services use different concepts (handling of parameters, intents and so on) and need a high amount of abstraction and trade-offs to be interchangeable. While plugins for other services could still be created and used, the chatbot now uses Dialogflow (version 2). Dialogflow was chosen because it shows promising development with features such as versioning and environments. It also offers a concept which is easy to use and understand and, therefore, enables fast prototyping. Another important aspect working in Dialogflow's favor is their free Standard Edition, which comes with a sufficient quota of text requests (see section 2.4).

Dialogflow's Messenger and webchat integrations, however, provide only little room for customization and lack support for advanced features such as message receipts (Facebook Messenger) and UI templates (webchat). Therefore, the chatbot uses a dedicated backend to handle incoming messages. On the user side, a client application (webchat or Facebook Messenger) is used. In the case of the webchat, communication happens directly between the server and the client. When using Facebook Messenger, client requests are first sent to Facebook's server before reaching Theodore's backend and vice versa.

Figure 3.7 shows the chronological structure of the communication between a user and the chatbot on the webchat. First, the client sends the user's input to the backend. The backend then sends the request to a NLU service, which extracts an intent and possible parameters to get to know what the user wants. If necessary, additional APIs are visited. For example, the latest blog posts are retrieved from the website using the Wordpress REST API.

[12]`https://ec.europa.eu/info/law/law-topic/data-protection_en`, visited on 2019-03-17

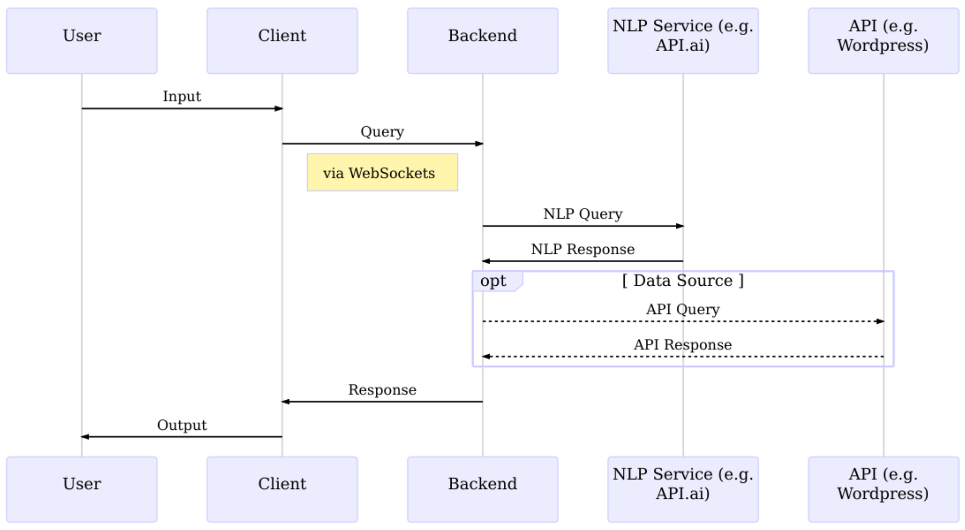


Figure 3.7.: Communication structure between a user and the chatbot on a webchat.

Subsequently, a proper response is created. After all this processing, the response is sent back to the client and presented to the user.

The chatbot has to access different APIs in order to provide all required features. Wordpress's REST API is used to retrieve blog posts as well as open positions from the website. Mailchimp's REST API is used to subscribe users to the newsletter, if they want to. Other content and responses such as product and service descriptions are stored in files or in Dialogflow's intent handling.

4. Implementation

Theodore is written from scratch using Node.js (version 8)[1], Socket.IO[2] and Dialogflow[3]. The backend handles connections from and to both Facebook Messenger and a custom webchat, written in HTML and TypeScript with the popular frontend framework Angular[4]. Communication is done with normal HTTP requests and WebSockets. This chapter shows some details about the implementation.

4.1. Structure

The chatbot was built with modularity in mind, in order to easily adapt to different needs and provide an extendable base for other projects. *Adapters* for NLU capabilities and analytics — like *Chatbase*[5] — are placed into `adapters/`, while *Bots* (implementations for additional platforms) reside in `bots/`. Currently, Theodore features scripts for Facebook Messenger, a webchat via Socket.IO and a complete adapter for Dialogflow. API connections and other features are called *Components* and can be added to `components/`. All frontend code related to the webchat lies in `src/`.

To support the aforementioned modularity, a *config*-package was used. Depending on the environment variable `NODE_ENV`, a corresponding configuration from `config/` is loaded (`config/default.json` being the default

[1] `https://nodejs.org/`, visited on 2019-03-31
[2] `https://socket.io/`, visited on 2019-03-31
[3] `https://dialogflow.com/`, visited on 2019-03-31
[4] `https://angular.io/`, visited on 2019-04-04
[5] `https://chatbase.com/`, visited on 2019-03-31

setup). The configuration file (sample shown in Listing 4.1) allows for quick and easy changes to adapters, components and API settings.

```json
{
  "nlp": {
    "dialogflow": {
      "project_id": "theodore",
      "environment": "staging"
    }
  },
  "web": {
    "enabled": true,
    "name": "WebChat",
    "url_path": "/"
  },
  "messenger": {
    "enabled": true,
    "name": "Facebook Messenger",
    "access_token": "ETHORZJGJWRGOJ",
    "verify_token": "some_token"
  },
  "app": {
    "name": "Theodore",
    "description": "der CodeFlügel Chatbot"
  },
  "server": {
    "port": 60000,
    "socketPort": 65282,
    "socketKey": "/path/to/key/privkey.pem",
    "socketCert": "/path/to/key/fullchain.pem"
  },
  "company": {
    "name": "CodeFlügel GmbH",
    "contact": "theodore@codefluegel.com",
    "web": "https://codefluegel.com/"
  },
  "blog": {
    "web": "https://codefluegel.com/blog/",
    "api": "https://codefluegel.com/wp-json/wp/v2/posts"
  }
}
```

Listing 4.1: Sample configuration for the backend application.

4.2. Natural Language Understanding

As mentioned in section 3.5, the initial goal was to make the implementation agnostic, meaning any NLU service (section 2.4) could be used. While still theoretically possible with the current implementation, only a module for Dialogflow has been completed, as it is sufficient for the case study in chapter 5. Although alternative adapters to extract intents and parameters are provided for Wit.ai, LUIS and Watson, the program needs some slight modifications for them to work properly, as Dialogflow's *Actions* are used, which are not supported by all NLU services. During development, Dialogflow released a version 2 of its API, so the adapter has been migrated from version 1 to version 2 to support its newly introduced features, *versioning* and *environments*. These features allowed for easier publishing to testing and production environments.

Dialogflow uses Google's *Protocol Buffers*[6] to encapsulate custom payloads, so special handling of this data had to be integrated. There were several solutions to this problem:

1. Using a Protocol Buffers module like *protobuf.js*[7]
2. Using Dialogflow's example code[8]
3. Using a custom solution

The latter was chosen, as it does not add another dependency and is shorter than the solution provided by Dialogflow's example. Listing 4.2 shows the custom approach.

```
processCustomPayloadMessage(object) {
  let outputMessage = Array.isArray(object) ? [] : {};
  Object.entries(object).forEach(([key, value]) => {
    if (value.kind == 'structValue') {
```

[6] https://developers.google.com/protocol-buffers/docs/reference/google.protobuf#google.protobuf.Struct, visited on 2019-04-01

[7] https://github.com/dcodeIO/protobuf.js, visited on 2019-04-01

[8] https://github.com/googleapis/nodejs-dialogflow/blob/f4017c534bae7f2b088e5e7d2da3a60cbf80642f/samples/structjson.js, visited on 2019-04-01

```
      outputMessage[key] = this
        .processCustomPayloadMessage(
          value.structValue.fields);
    } else if (value.kind == 'listValue') {
      outputMessage[key] = this
        .processCustomPayloadMessage(
          value.listValue.values);
    } else if (value.kind == 'stringValue') {
      outputMessage[key] = value.stringValue;
    } else if (value.kind == 'boolValue') {
      outputMessage[key] = value.boolValue;
    } else {
      outputMessage[key] = value;
    }
  });
  return outputMessage;
}
```

Listing 4.2: Method to deserialize custom payload from Protocol Buffers to JavaScript objects.

4.3. Content Management

The chatbot delivers content from various sources and, therefore, has to access different APIs and implement processing of such data. Job openings and blog posts are directly fetched from CodeFlügel's Wordpress website via Wordpress's REST API. Fetching happens by using simple HTTP GET requests to `https://codefluegel.com/wp-json/wp/v2/posts` for blog posts and `https://codefluegel.com/wp-json/wp/v2/jobs` for open positions. The latter content type is manually added to Wordpress, as it is not supported out of the box by the Content Management System (CMS). The content is received as JavaScript Object Notation (JSON) data and only needs marginal editing before being delivered to the user. Static content such as social media handles and contact information (mail, phone, location et cetera) is saved to Dialogflow.

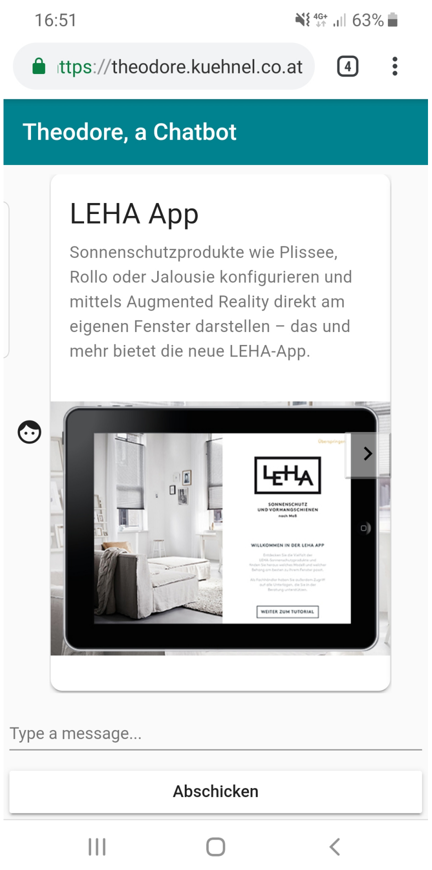
Only product and service data is put into a JavaScript file as a component (`components/products.json`), since it provides easier managing for larger data blocks than Dialogflow, as shown in Listing 4.3. When the NLU service recognizes an intent for showing information about products and services, it checks if data for the specific parameter is available and then shows a so-called "carousel" UI element[9] containing said information. For example, — referring to Listing 4.3 — if the user asks the chatbot "Was ist Augmented Reality?" (English translation: "What is Augmented Reality?"), the NLU engine detects the *products*-intent with "Augmented Reality" as its parameter. Dialogflow has been configured to even recognize synonyms such as "AR". The chatbot will then reply with the *description*-field as a text message and the *projects*-array as a carousel. The data source for this content can be easily changed, if necessary, as it is currently implemented as a component.

```
const data = {
  'Augmented Reality': {
    image_id: '',
    description: 'Mit Augmented Reality hauchen wir starren
        Objekten oder gedruckten Materialien neues Leben ein.
        Mehr zu dem Thema erfährst du in unserem Blog: https
        ://codefluegel.com/de/beginners-guide-augmented-
        reality/',
    projects: [
      {
        title: 'LEHA App',
        image: 'https://codefluegel.com/wp-content/uploads
            /2017/03/refleha2.jpg',
        description: 'Sonnenschutzprodukte wie Plissee, Rollo
             oder Jalousie konfigurieren und mittels Augmented
             Reality direkt am eigenen Fenster darstellen -
            das und mehr bietet die neue LEHA-App.',
        url: 'https://codefluegel.com/reference/leha-app/'
      },
      {
        title: 'KUKA App',
        image: 'https://codefluegel.com/wp-content/uploads
            /2017/04/refkuka.jpg',
        description: 'Für die neue Innovation von KUKA, die
            Reibschweißmaschine KUKA Genius, wurde eine eigene
```

[9]A slidable container - in our case consisting of one or more Generic Templates (see Figure 3.3 and Figure 4.1).

```
                Augmented Reality App erstellt, die auf kreative
               Weise deren top USPs aufzeigt.',
            url: 'https://codefluegel.com/reference/kuka-app/'
          }
        ]
      }
    };
```

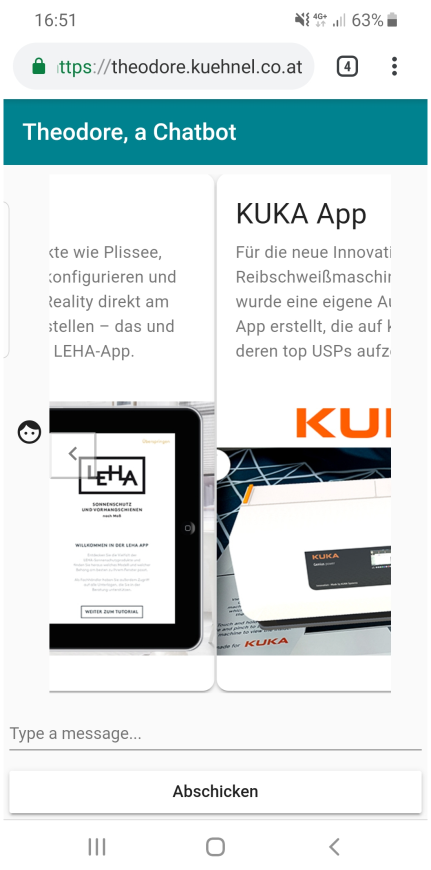
Listing 4.3: Content structure for products and services.



LEHA App

Sonnenschutzprodukte wie Plissee, Rollo oder Jalousie konfigurieren und mittels Augmented Reality direkt am eigenen Fenster darstellen – das und mehr bietet die neue LEHA-App.

(a) Generic Template #1



KUKA App

Für die neue Innovati Reibschweißmaschi wurde eine eigene A App erstellt, die auf k deren top USPs aufz

(b) Generic Template #2

Figure 4.1.: Generic Template in a carousel container (a) and sliding to the next template (b) with a swipe gesture.

4.4. Message Format

To unify the development for all platforms and make messages usable by all of them, a message format for the communication between the backend and clients was specified. Since supporting Facebook Messenger was already a requirement, its well-documented message format[10] was used. Theodore's implementation provides methods to build messages according to this structure. Sample messages in JSON format are shown in Listing 4.4 and Listing 4.5.

```
"message":{
  "text":"hello, world!"
}
```

Listing 4.4: A simple text JSON message in Facebook Messenger's format.

```
{
  "message":{
    "attachment":{
      "type":"template",
      "payload":{
        "template_type":"generic",
        "elements":[
           {
            "title":"<TITLE_TEXT>",
            "image_url":"<IMAGE_URL_TO_DISPLAY>",
            "subtitle":"<SUBTITLE_TEXT>",
            "default_action": {
              "type": "web_url",
              "url": "<DEFAULT_URL_TO_OPEN>",
              "messenger_extensions": <TRUE | FALSE>,
              "webview_height_ratio": "<COMPACT | TALL | FULL>"
            },
            "buttons":[<BUTTON_OBJECT>, ...]
          },
          ...
```

[10] https://developers.facebook.com/docs/messenger-platform/send-messages, visited on 2019-04-01

```
            ]
          }
        }
      }
    }
```

Listing 4.5: A generic template JSON message in Facebook Messenger's format.

4.5. Facebook Messenger

To add support for Facebook Messenger, a Facebook app had to be created[11]. This app links the Facebook page of a business — in this case CodeFlügel — to a backend application. The URL of the backend is provided to the Facebook app and an *access token* is then generated to authenticate future requests to Facebook. On the backend side, methods to handle necessary webhooks had to be implemented. A *verification token* is used for the initial setup to tie the backend application to the Facebook app and to allow it to receive webhooks whenever a message is sent to the page or specific events occur. The verification is done after providing Facebook with the backend URL via a HTTP GET request to `/fb/webhook`. Message-related webhooks are handled by POST requests to the same URL.

After a webhook is received, the application checks whether it is a message or another relevant event — for example, a user is joining the conversation. Events such as message receipts or reading indicators are ignored. Afterwards, the message is processed as described in section 3.5 and Figure 3.7.

The Facebook API for sending messages and retrieving additional information (for example, user details) is located at `https://graph.facebook.com/v2.11/`. The method for sending message data as well as typing indicators is shown in Listing 4.6, with `FB_PAGE_ACCESS_TOKEN` being the previously mentioned access token.

[11] `https://developers.facebook.com/docs/messenger-platform/getting-started/app-setup`, visited on 2019-04-11

```
callSendAPI(messageData) {
  return new Promise((resolve, reject) => {
    request({
      uri: 'https://graph.facebook.com/v2.11/me/messages',
      qs: { access_token: FB_PAGE_ACCESS_TOKEN },
      method: 'POST',
      json: messageData
    })
      .then(resolve)
      .catch(reject);
  });
}
```

Listing 4.6: Method for sending message objects to Facebook.

4.6. Webchat

As mentioned at the beginning of this chapter, the webchat is built in Angular (initially with version 5, later upgraded to version 6), using HTML, Sassy CSS (SCSS)[12] and TypeScript. The webchat also uses Socket.IO[13] for transmitting messages over WebSockets, a technology to enable (live) two-way communication between browser and server[14]. Unlike with Facebook Messenger, the webchat does not rely on a third-party server infrastructure for sending messages. If the browser does not support this feature or the communication fails due to a server misconfiguration, HTTP requests are used as a fallback.

Server configuration proved to be challenging, as the Node.js application was handling web hooks — via HTTPS (Hypertext Transfer Protocol Secure) — as well as WebSockets (*ws*) and serving the Angular (client) app. This was especially true in combination with WebSocket Secure (*wss*), as the Node.js application is usually not exposed publicly, but served through a reverse proxy. A reverse proxy server is "a type of proxy server that typically sits

[12] `https://sass-lang.com/`, visited on 2019-04-06
[13] `https://socket.io/`, visited on 2019-03-2019
[14] `https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API`, visited on 2019-04-04

behind the firewall in a private network and directs client requests to the appropriate backend server"[15]. This adds an abstraction layer and security, but also requires additional steps for the configuration of our backend and web app.

In the end, a separate HTTPS server for the socket connections was created in the Node.js app, to handle secure WebSockets. The socket port 65282 was then directly exposed to enable access from the web chat (Angular web app). The rest of the backend was sitting behind the reverse proxy, which was also in charge of handling HTTPS connections with TLS (Transport Layer Security). Necessary certificates were issued with *Let's Encrypt*[16].

[15] https://www.nginx.com/resources/glossary/reverse-proxy-server/, visited on 2019-04-10
[16] https://letsencrypt.org/, visited on 2019-04-10

5. Case Study - Website Usage vs. Bot Usage

5.1. Introduction

In order to see how the chatbot performs and whether users accept it as an alternative to a website, a user study was conducted. Tests took place from November 6th to December 22nd 2018 with individual participants. Users were chosen based on a specific user classification. The goal was to only introduce participants which are relevant to the company's website[1] (and chatbot respectively) — for example, potential employees. Those users were asked to perform several common tasks on the company website and with a corresponding chatbot. Timings of those tasks and user feedback were recorded to get quantitative and qualitative results.

The test company used was CodeFlügel GmbH, an Austrian software developing company specialized in mobile apps and augmented reality. For more information about the company see chapter 3.

5.2. Setup

5.2.1. User Selection

Participants were chosen to reflect the user base of the company's website. The relevant user groups and personas have been explained in section 3.2:

[1] `https://www.codefluegel.com`, visited on 2018-10-28

- Clients
- Potential Employees
- Blog Readers and Everyone Else

A total of twenty (20) users were selected based on the previously outlined classification. This approach makes it possible to better adapt the selection of test subjects to the actual users of the website and to create more appropriate tasks. At least five (5) participants per user type were among them. The age ranged from 20 to 33, while the average user was 27.7 years old. 60% of the users had a bachelor or master's degree while another 25% were still working on their bachelor (with 20% working on their master's degree). 70% had a technical background (mostly IT), either through work or education.

5.2.2. Preparation

At the beginning of the test session, users were greeted and given a summary about the project as well as about the user study and the upcoming steps. Then they were asked to fill out a consent form and a questionnaire about their background, knowledge in different technologies such as chat applications, augmented reality and programming as well as their expectations of a company's website and chatbot (see Appendix A on page 85). The goal was to collect enough background information about the users to be able to find possible connections between the use of the chatbot or the website and the different backgrounds.

5.2.3. Setup and Tasks

The test setup consisted of a workspace with a computer, two monitors, a keyboard and a mouse (Figure 5.1). The participants were placed in front of the computer while the facilitator was sitting to their left, a little further back, to observe the test person's actions and measure the time of each task. The right monitor showed instructions for the tasks, while the left one was used to perform the tasks. The actions on the left monitor were captured with the screen capturing software OBS Studio[2].

[2] `https://obsproject.com`, visited on 2019-01-04

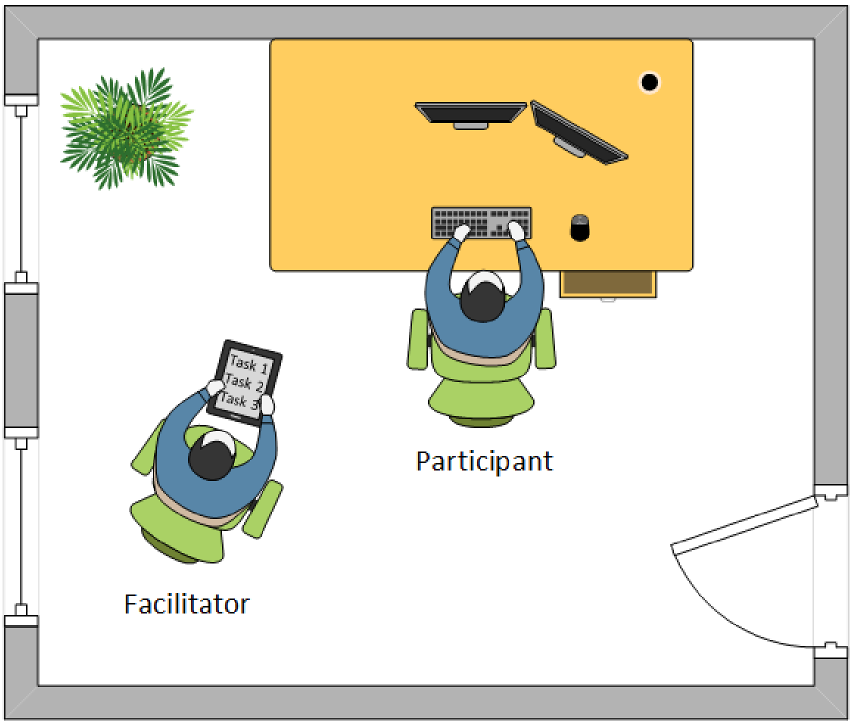
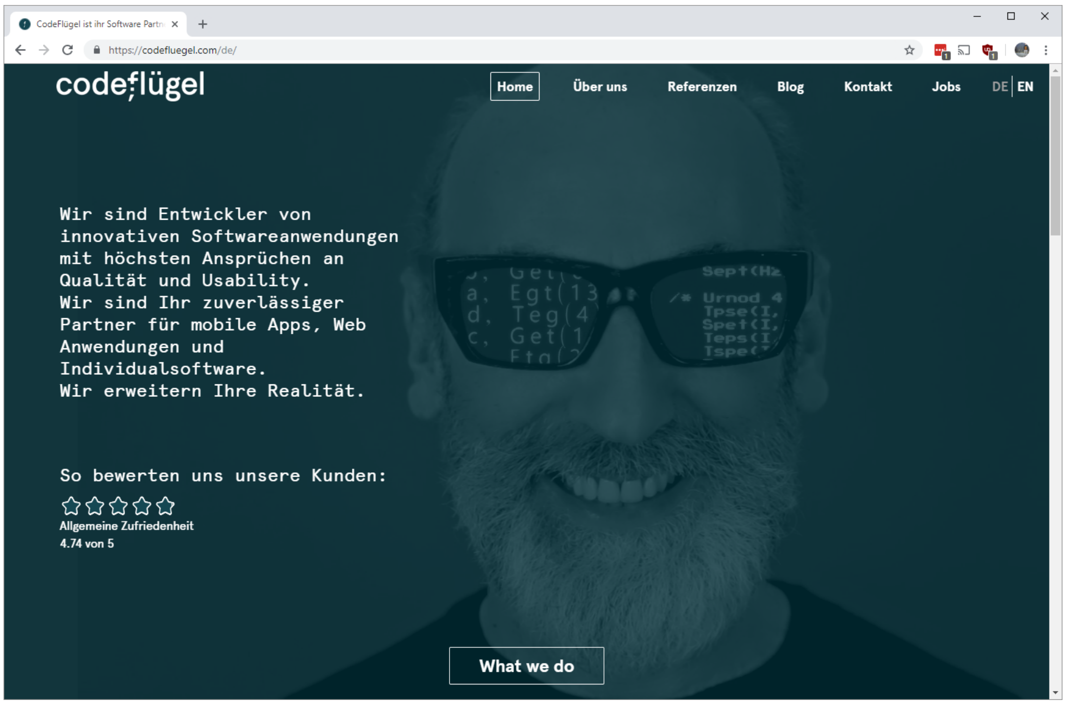


Figure 5.1.: The user test setup inspired by Andrews (2018, p.118)

#	Description and Goal	
1	Task:	Find out what CodeFlügel does or which services they provide.
	Goal:	"Developer of Mobile Apps, Web and Augmented/Virtual Reality."
2	Task:	Find out where CodeFlügel's office is and what phone number you can call.
	Goal:	User finds address and phone number.
3	Task:	Find out which companies CodeFlügel has already implemented projects for.
	Goal:	User finds reference page[a]/list.
4	Task:	Find and open the latest blog entry.
	Goal:	User opens the latest blog entry.
5	Task:	Sign up for the newsletter with the e-mail address <firstname>.<surname>@codefluegel.com.
	Goal:	User signs up for the newsletter.
6	Task:	Find out if and which jobs are currently available.
	Goal:	User finds job page[b]/list.
7	Task:	Find a way to try Augmented Reality (AR) for yourself.
	Goal:	User finds demo app.
8	Task:	Find at least one Augmented Reality (AR) project created by CodeFlügel.
	Goal:	User names at least one (1) Augmented Reality (AR) project.
9	Task:	Find out what Augmented Reality (AR) actually is.
	Goal:	User opens AR explanation.[c]

Table 5.1.: This table shows the tasks users have to complete with both the website and the chatbot.

[a]https://codefluegel.com/referenzen/, visited on 2018-10-28
[b]https://codefluegel.com/jobs/, visited on 2018-10-28
[c]https://codefluegel.com/beginners-guide-augmented-reality/, visited on 2018-10-28

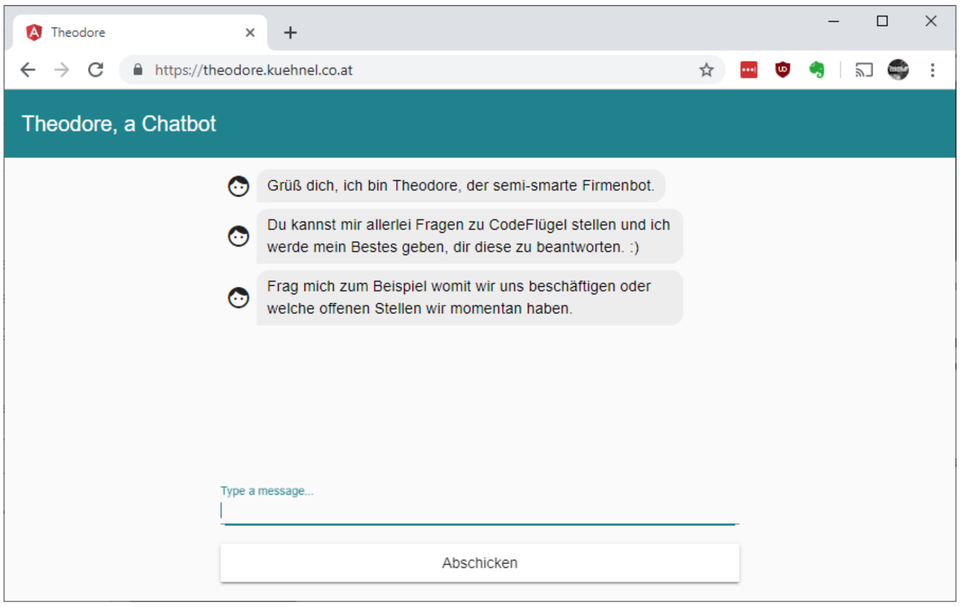


Wir sind Entwickler von innovativen Softwareanwendungen mit höchsten Ansprüchen an Qualität und Usability.
Wir sind Ihr zuverlässiger Partner für mobile Apps, Web Anwendungen und Individualsoftware.
Wir erweitern Ihre Realität.

So bewerten uns unsere Kunden:
☆☆☆☆☆
Allgemeine Zufriedenheit
4.74 von 5

Figure 5.2.: Home screen of the company website, visited on 2018-11-04

A list of nine (9) typical tasks was defined (Table 5.1), taking into account the various user groups and the target audience of the company's website. The list included things such as looking for the latest blog post, searching for job openings and finding at least one completed augmented reality project of the company. First, the tasks had to be performed on the company's website. In order to increase the quality of feedback and observations, users were asked to verbalize their thoughts, similar to so called *thinking-aloud-tests* (Andrews, 2018, p.135). Each task instruction had to be read out loud by the participants. After that, a timer was started by the facilitator. The user had to clearly state the solution to the given task and, if it was the right one, the timer was stopped. When all nine tasks were completed, the same tasks had to be performed using the chatbot instead of the website.



Theodore, a Chatbot

Grüß dich, ich bin Theodore, der semi-smarte Firmenbot.

Du kannst mir allerlei Fragen zu CodeFlügel stellen und ich werde mein Bestes geben, dir diese zu beantworten. :)

Frag mich zum Beispiel womit wir uns beschäftigen oder welche offenen Stellen wir momentan haben.

Figure 5.3.: Webchat with the company chatbot

5.2.4. Feedback and Interviews

After the completion of the tasks, another questionnaire had to be filled out (see Appendix B on page 85). The questionnaire was there to gather the participant's feedback, recognize potential flaws in the chatbot design and measure the acceptability of the company chatbot. An interview was then conducted to clear up any potential ambiguities and obtain better-quality feedback.

6. Findings

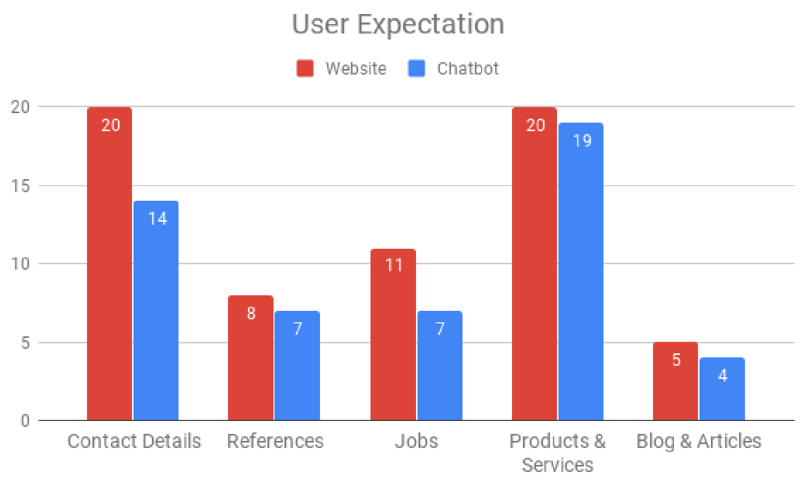


Figure 6.1.: What users expect of a company website and chatbot

Figure 6.1 shows user expectation of the company's chatbot and website respectively. Expectations were pretty close. Information about *Products & Services* and *Contact Details* were expected by all users from the website, while 70% (Contact Details) to 85% (Products & Services) expected the same from the chatbot. *References*, *Jobs* and *Blog & Articles* only amounted to 20% – 55% percent of the users' expectations.

80% of participants were faster using the chatbot (Figure 6.2). The average difference in time needed for all tasks was 2 minutes and 54 seconds in favor of the chatbot. The average completion time of all tasks was 3m 38s with the chatbot and 6m 33s with the website, while the averages for one task were 24 seconds (chatbot) and 43 seconds (website) (Figure 6.3).

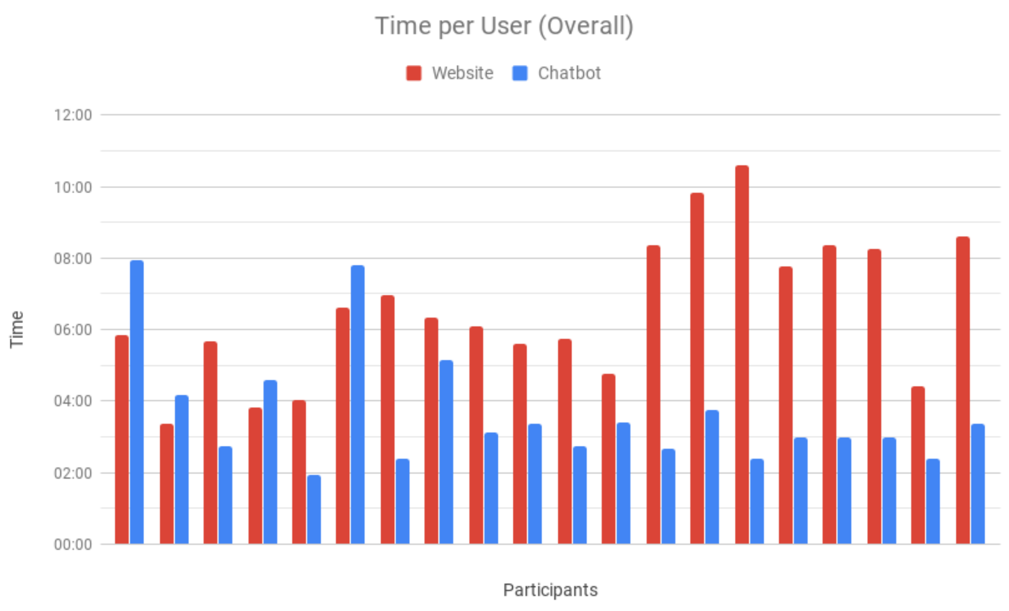


Figure 6.2.: Total time needed to complete the tasks (per User)

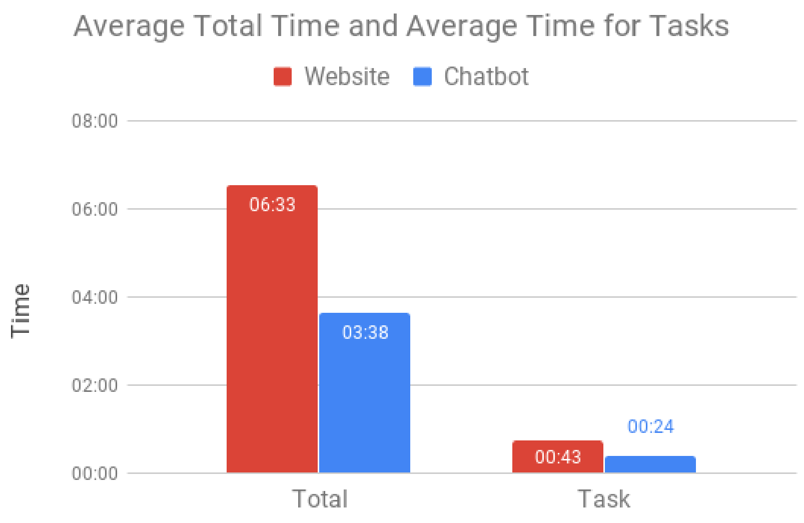


Figure 6.3.: Average total time and average time for tasks

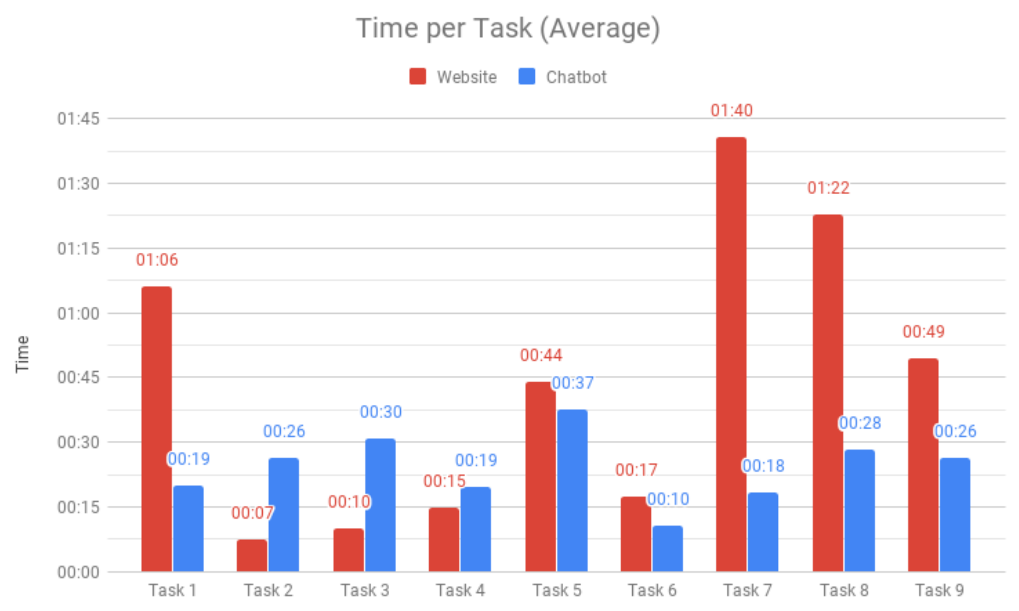


Figure 6.4.: Average time per task

Six out of the nine tasks were completed faster with the chatbot, although two of those six were only completed about six seconds faster (Figure 6.4). Tasks involving the website were faster or equally fast when a corresponding and self-explanatory (e.g. Blog) menu entry existed, while tasks seeking specific information not immediately accessible by the main menu (tasks 1, 7, 8 and 9) where significantly faster achieved with the chatbot. This is similar to what Beriault-Poirier, Tep, and Sénécal (2019) concluded in their research.

In terms of perceived completion time, 80% of test users thought they were faster using the chatbot. 75% correctly assessed the chatbot as being faster, while one of the participants thought completion time was faster, but the opposite was true. Another one thought their website tasks were completed faster, but in reality, the chatbot run was the quicker one. 15% of the test users believed to have been equally quick with both platform tasks while actually being faster with the website.

Although the chatbot matched every participant's expectations, seven out

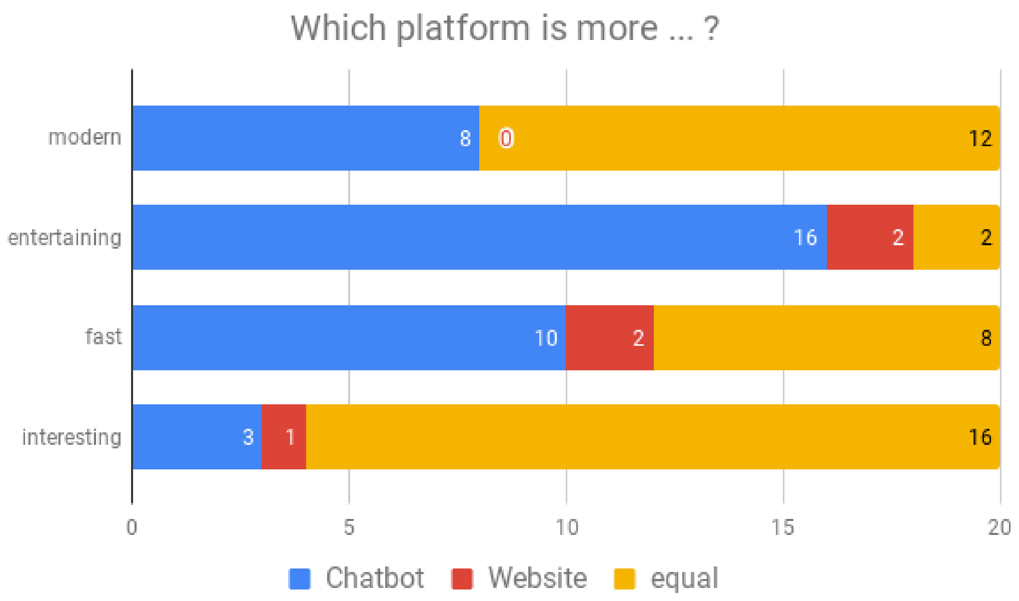


Figure 6.5.: What users say about the company's chatbot and website

of the 20 users said the input-to-intent matching could be improved, as the chatbot did not understand every message and reformulating the request was necessary. Figure 6.5 shows the participants' feedback regarding the platforms. 80% of the users found the chatbot more *entertaining* than the website — vice versa, only 10% were more entertained using the website. 40% deemed the chatbot more *modern*, while none said anything similar about the website. The chatbot was also described as *fast* by 50%, while only 10% used this adjective for the website.

60% of the participants said they preferred a chatbot with informal communication. The other 40% did not care about whether the chatbot approached the conversation formally or informally (Figure 6.6). This pretty much confirms what Survata and LivePerson (2017, p. 15) published in their study.

Hill, Ford, and Farreras (2015) showed that humans tend to use shorter messages when conversing with chatbots. Our study revealed that 50% of users relied on a combination of whole sentences and simple keywords to

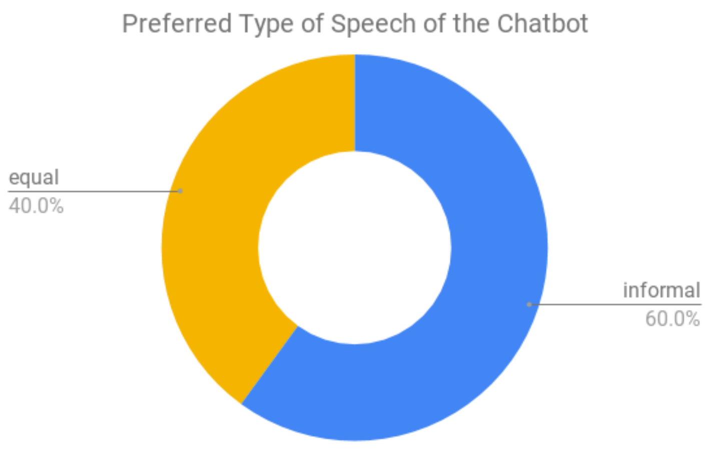


Figure 6.6.: Preferred type of speech of the chatbot

communicate with the chatbot. 30% used only sentences, while 20% based their messages solely on keywords (Figure 6.7).

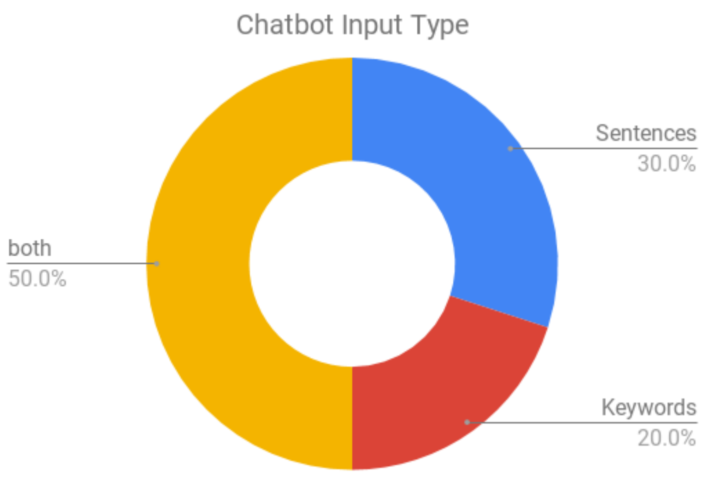


Figure 6.7.: The chart shows how the users communicated with the chatbot

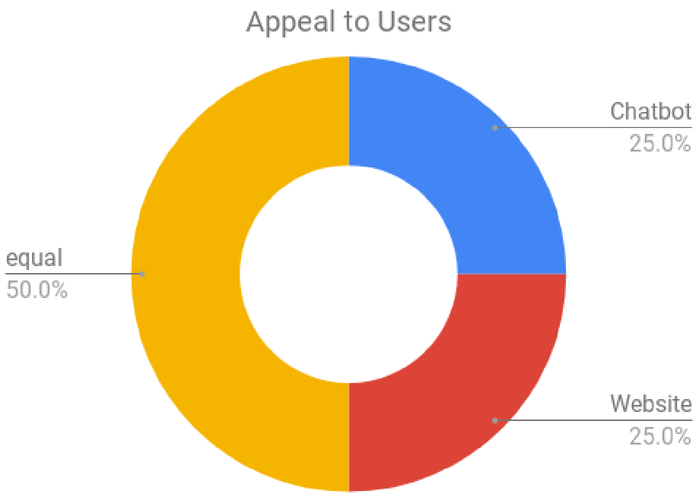


Figure 6.8.: Which platform was more appealing to the users

Half the users thought the website and chatbot were equally appealing to them (Figure 6.8). The website was favored by 25% of the participants, partially due to being used to websites or due to being able to explore the content more freely. Beriault-Poirier, Tep, and Sénécal (2019) suspected similar reasons for favoring websites over chatbots in their study. The other 25% found the chatbot more appealing. Reasons to favor the chatbot were instantaneous answers to specific questions without the hassle of clicking through menus and searching through pages, as well as its interactive nature, which resulted in more fun for some of the participants.

Overall, the company chatbot was received positively and 85% of participants could see themselves using more chatbots in the future (Figure 6.9).

Despite extensive background checks of the participants (see Appendix A), no correlation between the time needed to complete the tasks and the level of IT knowledge was observed.

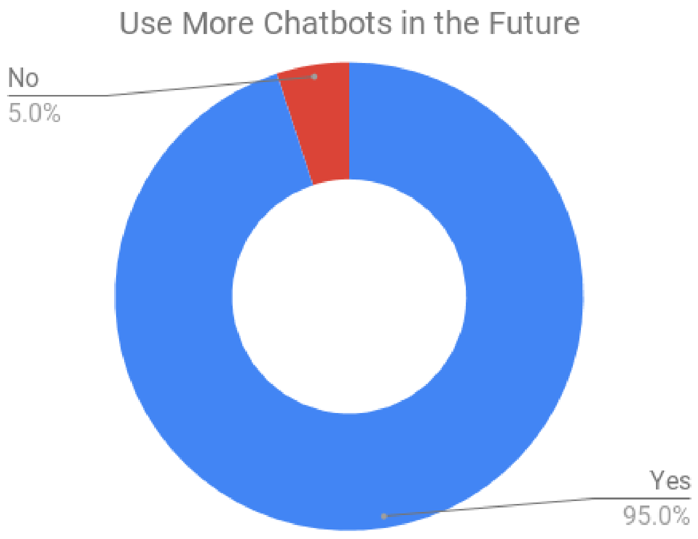


Figure 6.9.: 95% would use more chatbots

7. Conclusion

In chapter 2, chatbots and their history were introduced. Section 2.2 shed some light on chatbots for customer interaction and related academic work. Section 2.4, section 2.5, section 2.6 and section 2.7 listed frameworks, services and platforms for building and running chatbots.

Chapter 3 outlined the design of *Theodore*, a chatbot built for the Austrian company *CodeFlügel*. Target audience, dialog design, platforms and architecture were discussed. The chatbot's implementation in Node.js (backend) and Angular (web chat) was explained in detail in chapter 4.

In chapter 5, a case study about the aforementioned chatbot was conducted. Typical use cases of a company chatbot were worked out in order to compare such a chatbot with a traditional company website and measure its performance. Section 5.2 presented the setup of the study as well as the testing procedure.

The results of this case study were presented in chapter 6. The user study showed that chatbots are definitely accepted by users. The users provided positive feedback about the chatbot, only some had minor issues with the matching of their queries and the size of the chat window. However, this could be improved easily by slightly changing the web chat design and providing more training data to the chatbot and the respective NLU engine. The study also revealed that it is important to train for whole sentences as well as key words, as both are commonly used by users. This was also discovered by Hill, Ford, and Farreras (2015). The chatbot was faster than the website and the users perceived it as more entertaining. Thus, it could potentially help companies in reaching new or younger audiences and in improving specific parts of their websites, such as the FAQs.

8. Future Perspectives

Building on this study, further testing of the impact of UI improvements can be done. Natural Language Processing and matching of user inputs to intents also bear potential to increase the acceptance and enhance the user experience. In general, more chatbots could be tested and more users from different backgrounds and age groups could provide more valuable insight.

Another aspect to investigate further is the effect of different chatbot personas on the user experience and the influence of brand perception, as slightly touched upon by this thesis in section 2.3.

One could also evaluate the possibilities of distributing and marketing bots. Do chatbot "marketplaces" provide a similar exposure to mobile app stores such as Google Play[1] and Apple App Store[2]?

Conversion rates and other Key Performance Indicators (KPIs) are things that could be explored in more detail as well. These analytics could show, whether chatbots are economically feasible and here to stay.

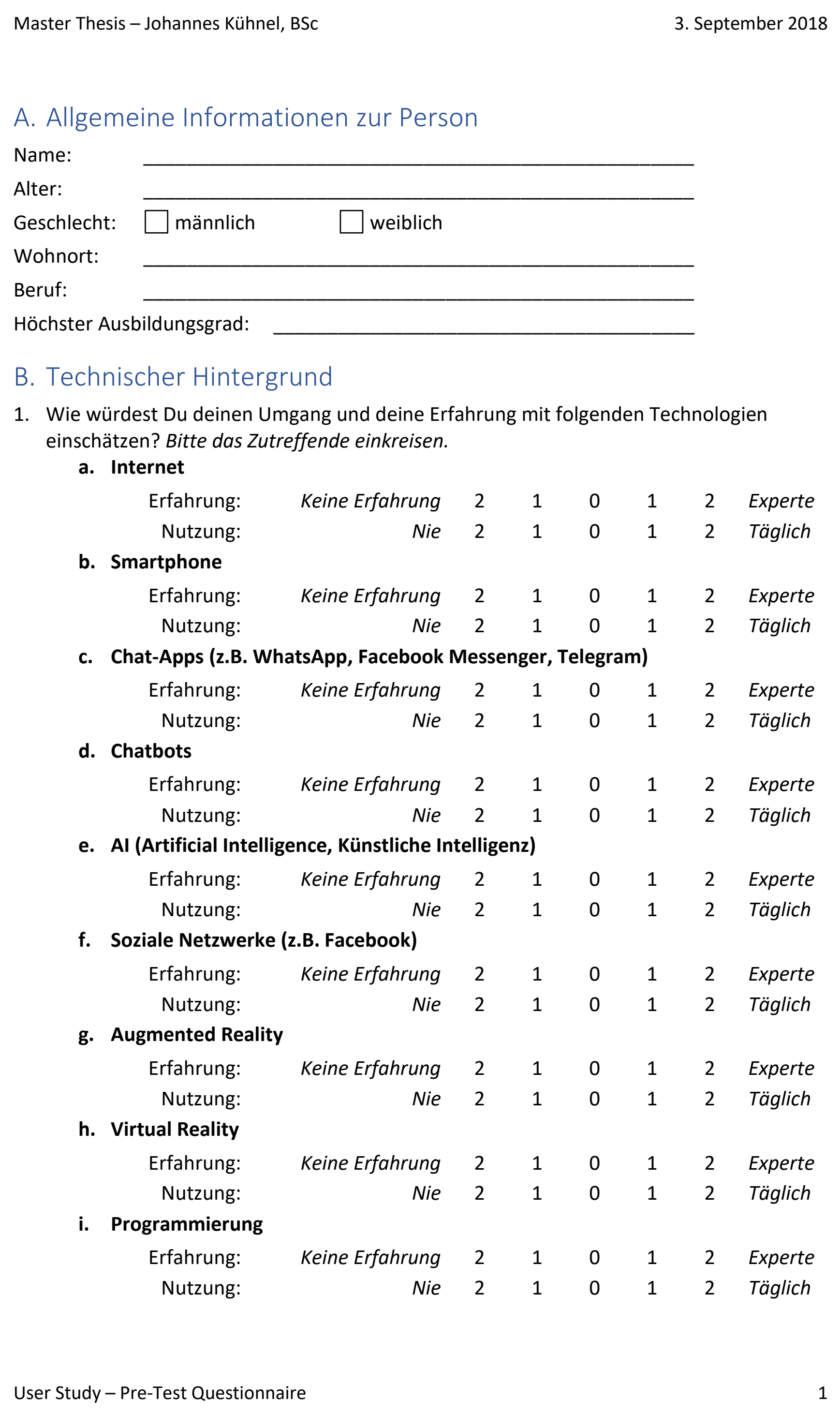
[1] `https://play.google.com/store`, visited on 2019-03-24
[2] `https://www.apple.com/ios/app-store/`, visited on 2019-03-24

Appendix

Appendix A.

Questionnaire (Pre-Test)

Appendix A. Questionnaire (Pre-Test)



Master Thesis – Johannes Kühnel, BSc 3. September 2018

A. Allgemeine Informationen zur Person

Name: ____

Alter: ____

Geschlecht: ☐ männlich ☐ weiblich

Wohnort: ____

Beruf: ____

Höchster Ausbildungsgrad: ________________________________

B. Technischer Hintergrund

1. Wie würdest Du deinen Umgang und deine Erfahrung mit folgenden Technologien einschätzen? *Bitte das Zutreffende einkreisen.*

a. Internet

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

b. Smartphone

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

c. Chat-Apps (z.B. WhatsApp, Facebook Messenger, Telegram)

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

d. Chatbots

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

e. AI (Artificial Intelligence, Künstliche Intelligenz)

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

f. Soziale Netzwerke (z.B. Facebook)

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

g. Augmented Reality

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

h. Virtual Reality

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

i. Programmierung

Erfahrung:	*Keine Erfahrung*	2	1	0	1	2	*Experte*
Nutzung:	*Nie*	2	1	0	1	2	*Täglich*

User Study – Pre-Test Questionnaire 1

2. Mit welchem Betriebssystem arbeitest Du für gewöhnlich **am Computer**?

   ☐ Windows 10 ☐ macOS ☐ Anderes: ______________________

3. Mit welchem Betriebssystem arbeitest Du für gewöhnlich **am Smartphone oder Tablet**?

   ☐ Android ☐ iOS ☐ Anderes: ______________________

4. Welches Gerät nutzt Du am häufigsten?

   ☐ Computer ☐ Smartphone ☐ Tablet

C. Erwartungshaltung

1. Was erwartest Du dir von einer Firmenwebseite?

   ☐ Kontaktdaten ☐ Referenzen ☐ Stellenausschreibungen
   ☐ Informationen zu Produkten & Serviceleistungen ☐ Blog / Artikel zu Kernthemen
   ☐ Anderes:

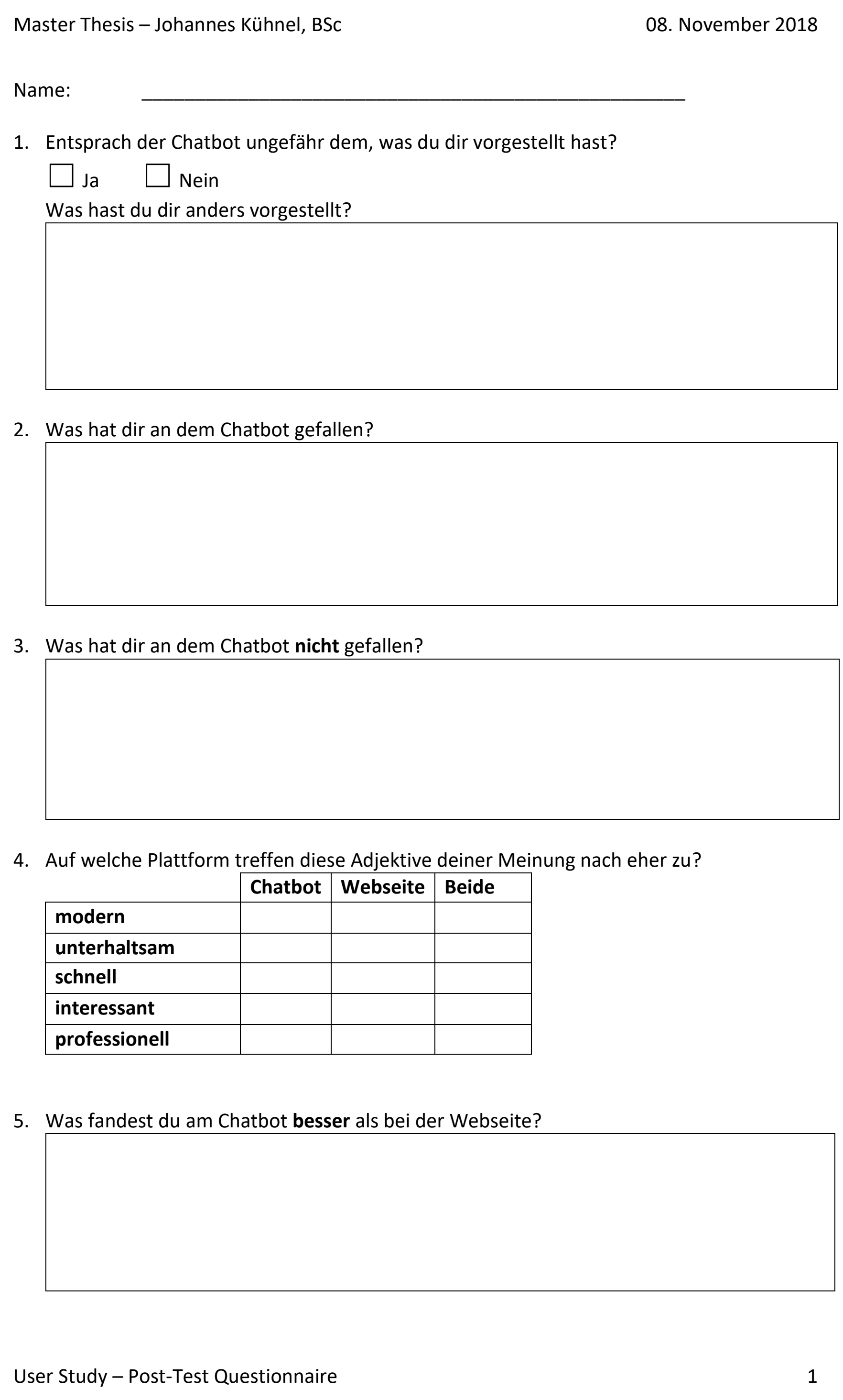
2. Was erwartest Du dir von einem Firmen-Chatbot?

   ☐ Kontaktdaten ☐ Referenzen ☐ Stellenausschreibungen
   ☐ Informationen zu Produkten & Serviceleistungen ☐ Blog / Artikel zu Kernthemen
   ☐ Anderes:

Appendix B.

Questionnaire (Post-Test)

Appendix B. Questionnaire (Post-Test)



Master Thesis – Johannes Kühnel, BSc 08. November 2018

Name: ____

1. Entsprach der Chatbot ungefähr dem, was du dir vorgestellt hast?

   ☐ Ja ☐ Nein

   Was hast du dir anders vorgestellt?

2. Was hat dir an dem Chatbot gefallen?

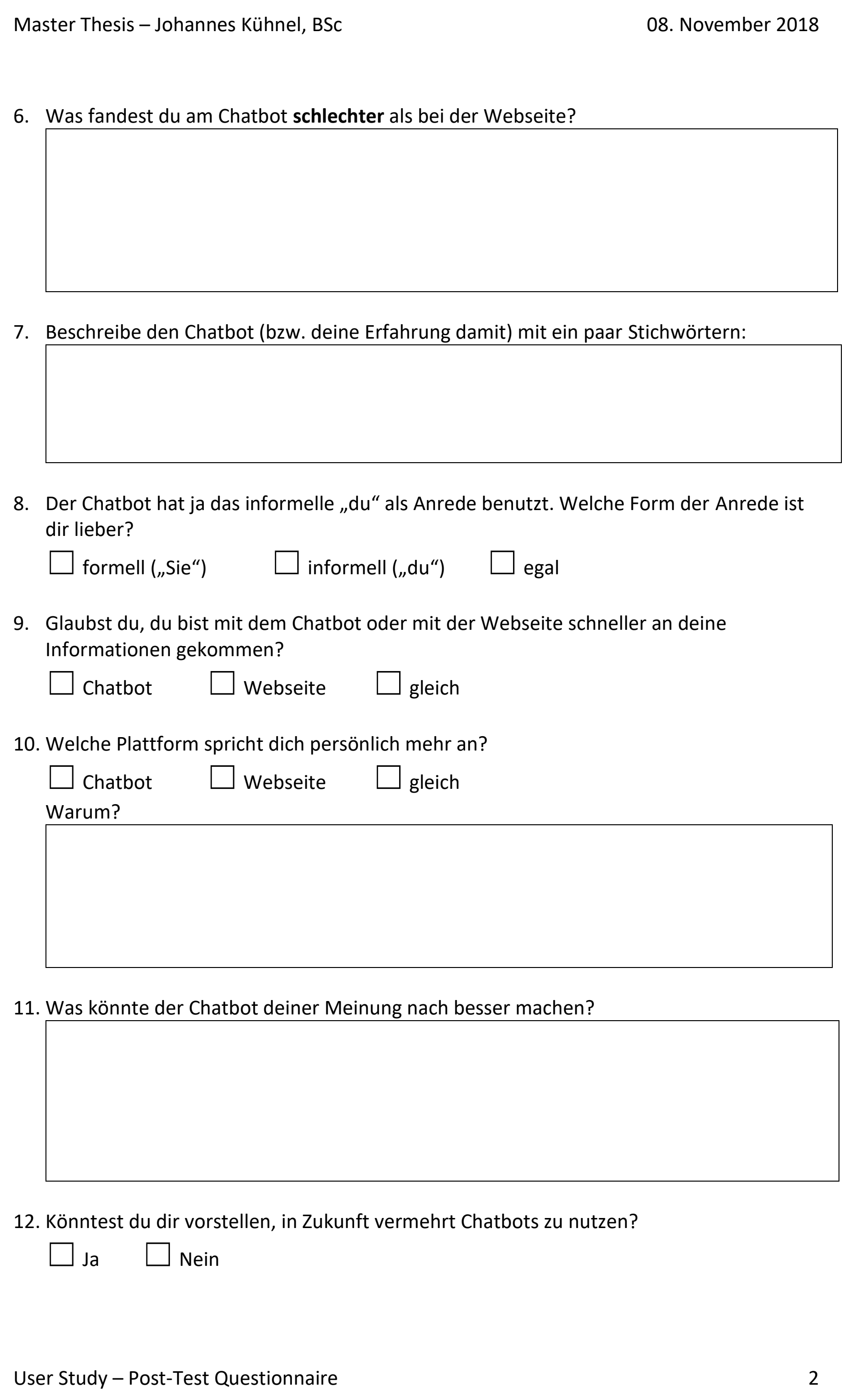
3. Was hat dir an dem Chatbot **nicht** gefallen?

4. Auf welche Plattform treffen diese Adjektive deiner Meinung nach eher zu?

	Chatbot	**Webseite**	**Beide**
modern			
unterhaltsam			
schnell			
interessant			
professionell			

5. Was fandest du am Chatbot **besser** als bei der Webseite?

User Study – Post-Test Questionnaire 1



Master Thesis – Johannes Kühnel, BSc 08. November 2018

6. Was fandest du am Chatbot **schlechter** als bei der Webseite?

7. Beschreibe den Chatbot (bzw. deine Erfahrung damit) mit ein paar Stichwörtern:

8. Der Chatbot hat ja das informelle „du“ als Anrede benutzt. Welche Form der Anrede ist dir lieber?

   ☐ formell („Sie“) ☐ informell („du“) ☐ egal

9. Glaubst du, du bist mit dem Chatbot oder mit der Webseite schneller an deine Informationen gekommen?

   ☐ Chatbot ☐ Webseite ☐ gleich

10. Welche Plattform spricht dich persönlich mehr an?

    ☐ Chatbot ☐ Webseite ☐ gleich

    Warum?

11. Was könnte der Chatbot deiner Meinung nach besser machen?

12. Könntest du dir vorstellen, in Zukunft vermehrt Chatbots zu nutzen?

    ☐ Ja ☐ Nein

User Study – Post-Test Questionnaire 2

Bibliography

Andrews, Keith (June 12, 2018). *HCI Course Notes*. Graz University of Technology, p. 118 (cit. on pp. 63, 65).

Android (Oct. 2013). *KitKat 4.4*. URL: https://www.android.com/versions/kit-kat-4-4/ (visited on 01/24/2019) (cit. on p. 5).

Android Central (June 23, 2015). *Amazon Echo is now available for everyone to buy for £179.99, shipments start on July 14*. URL: https://www.androidcentral.com/amazon-echo-now-available-everyone-buy-17999-shipments-start-july-14 (visited on 01/24/2019) (cit. on p. 5).

Appfigures (Oct. 2018a). *Number of available apps at Google Play from 2nd quarter 2015 to 3rd quarter 2018*. URL: https://www.statista.com/statistics/289418/number-of-available-apps-in-the-google-play-store-quarter/ (visited on 01/24/2019) (cit. on p. 6).

Appfigures (Oct. 2018b). *Number of available apps in the Apple App Store from 2nd quarter 2015 to 3rd quarter 2018*. URL: https://www.statista.com/statistics/779768/number-of-available-apps-in-the-apple-app-store-quarter/ (visited on 01/24/2019) (cit. on p. 6).

Apple (Oct. 4, 2011). *Apple Launches iPhone 4S, iOS 5 & iCloud*. URL: https://www.apple.com/newsroom/2011/10/04Apple-Launches-iPhone-4S-iOS-5-iCloud/ (visited on 01/24/2019) (cit. on p. 5).

Araujo, Theo (Aug. 2018). "Living up to the chatbot hype: The influence of anthropomorphic design cues and communicative agency framing on conversational agent and company perceptions." In: *Computers in Human Behavior*. Vol. 85, pp. 183–189. DOI: 10.1016/j.chb.2018.03.051 (cit. on p. 13).

Bager, Jo (2016). "Gesprächige Automaten. Chatbots: Viele praktische Quasselprogramme am Horizont." In: *c't* 24/2016, pp. 112–114 (cit. on p. 5).

Beriault-Poirier, Amélie, Sandrine Prom Tep, and Sylvain Sénécal (2019). "Putting Chatbots to the Test: Does the User Experience Score Higher with Chatbots Than Websites?" In: *Human Systems Engineering and Design*. Springer International Publishing, pp. 204–212. ISBN: 978-3-030-02053-8. DOI: 10.1007/978-3-030-02053-8_32 (cit. on pp. 12, 69, 73).

Chatbots Magazine (Mar. 12, 2017). *China, WeChat, and the Origins of Chatbots*. URL: https://chatbotsmagazine.com/china-wechat-and-the-origins-of-chatbots-89c481f15a44 (visited on 01/24/2019) (cit. on pp. 5, 24).

Chatbots Magazine and Anatoly Khorozov (Mar. 1, 2017). *Trends Driving the Chatbot Growth*. URL: https://chatbotsmagazine.com/trends-driving-the-chatbot-growth-77b78145bac (visited on 03/14/2019) (cit. on p. 10).

Ciechanowski, Leon et al. (Mar. 2019). "In the shades of the uncanny valley: An experimental study of human–chatbot interaction." In: *Future Generation Computer Systems*. Vol. 92, pp. 539–548. DOI: 10.1016/j.future.2018.01.055 (cit. on p. 13).

Comscore (Sept. 22, 2015). *The 2015 U.S. Mobile App Report*. Tech. rep. Comscore. URL: https://www.comscore.com/Insights/Presentations-and-Whitepapers/2015/The-2015-US-Mobile-App-Report (visited on 01/24/2019) (cit. on pp. 1, 6).

Facebook (Apr. 18, 2017). *Messenger Platform 2.0 Debuts at F8*. Ed. by Mikhail Larionov. Facebook. URL: https://developers.facebook.com/blog/post/2017/04/18/messenger-platform-2.0/ (visited on 01/24/2019) (cit. on p. 5).

Gartner (Oct. 3, 2017). *Gartner Top Strategic Predictions for 2018 and Beyond*. Gartner, Inc. URL: https://www.gartner.com/smarterwithgartner/gartner-top-strategic-predictions-for-2018-and-beyond/ (visited on 01/22/2019) (cit. on p. 5).

Gartner (Mar. 28, 2018). *Chatbots Will Appeal to Modern Workers*. Gartner, Inc. URL: https://www.gartner.com/smarterwithgartner/chatbots-will-appeal-to-modern-workers/ (visited on 01/22/2019) (cit. on p. 6).

Heckmann, Jörn and Michael Kraus (2018). "You, Robot. Rechtliche Risiken beim Einsatz von Chatbots." In: *iX* 6/2018, pp. 70–72 (cit. on p. 12).

Hill, Jennifer, W. Randolph Ford, and Ingrid G. Farreras (2015). "Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations." In: *Computers in Human Behavior*. Vol. 49, pp. 245–250. DOI: 10.1016/j.chb.2015.02.026 (cit. on pp. 70, 75).

Hootsuite and We Are Social (2019). *Digital 2019. Global Digital Overview*. Tech. rep. Mindbowser. URL: https://datareportal.com/reports/digital-2019-global-digital-overview (visited on 02/04/2019) (cit. on pp. 21, 22, 24).

Klopfenstein, Lorenz Cuno et al. (June 2017). "The Rise of Bots: A Survey of Conversational Interfaces, Patterns, and Paradigms." In: *Proceedings of the 2017 Conference on Designing Interactive Systems*. DIS '17, pp. 555–565. ISBN: 978-1-4503-4922-2. DOI: 10.1145/3064663.3064672 (cit. on pp. 1, 10).

Lebeuf, Carlene, Margaret-Anne Storey, and Alexey Zagalsky (Dec. 25, 2017). "Software Bots." In: 35 (1), pp. 18–23. DOI: 10.1109/MS.2017.4541027 (cit. on pp. 7, 12).

Mindbowser and ChatbotsJournal.com (2017). *Chatbot Survey 2017*. Tech. rep. Mindbowser. URL: http://mindbowser.com/chatbot-market-survey-2017/ (visited on 02/04/2019) (cit. on pp. 20, 23).

Nadella, Satya (Mar. 25, 2016a). *Build 2016. Keynote Presentation*. Microsoft. URL: https://channel9.msdn.com/Events/Build/2016/KEY01#time=1h41m11s (visited on 02/13/2019) (cit. on pp. 21, 24).

Nadella, Satya (July 11, 2016b). *Satya Nadella: Worldwide Partner Conference 2016*. Microsoft. URL: https://news.microsoft.com/speeches/satya-nadella-worldwide-partner-conference-2016/ (visited on 01/28/2019) (cit. on p. 6).

Paikari, Elahe and André van der Hoek (May 27, 2018). "A Framework for Understanding Chatbots and their Future." In: *CHASE '18 Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering*. 2018 ACM/IEEE 11th International Workshop on Cooperative and Human Aspects of Software Engineering, pp. 1–6. ISBN: 978-1-4503-5725-8. DOI: 10.1145/3195836.3195859 (cit. on p. 7).

Puscher, Frank (2018). "Holpriger Start. Chatbots für Unternehmen." In: *iX* 6/2018, pp. 58–62 (cit. on p. 11).

Shawar, Bayan Abu and Eric Atwell (2007). "Chatbots: Are They Really Useful?" In: *LDV-Forum* 22.1, pp. 29–49. URL: https://jlcl.org/content/2-allissues/20-Heft1-2007/Bayan_Abu-Shawar_and_Eric_Atwell.pdf (visited on 01/24/2019) (cit. on p. 3).

Shawar, Bayan Abu and Eric Atwell (Sept. 2015). "A chatbot as a Question Answering Tool." In: *2015 International Conference on Advances in Software, Control and Mechanical Engineering*. 2015 International Conference on Advances in Software, Control and Mechanical Engineering, pp. 1–6. ISBN: 978-93-84422-37-0. DOI: 10.17758/UR.U0915120 (cit. on p. 10).

Shevat, Amir (May 17, 2017). *Designing Bots*. O'Reilly Media, Inc. ISBN: 978-1-4919-7482-7 (cit. on pp. 3, 7, 9, 13).

Survata and LivePerson (Apr. 2017). *How Consumers View Bots In Customer Care*. Tech. rep. LivePerson. URL: https://www.liveperson.com/resources/reports/bots-in-customer-care/ (visited on 02/20/2019) (cit. on p. 70).

Tencent (Aug. 17, 2016). *Tencent Announces 2016 Second Quarter And Interim Results*. Tencent Holdings Limited. URL: https://www.tencent.com/en-us/articles/1500541477476593.pdf (visited on 01/24/2019) (cit. on p. 5).

The Guardian (Mar. 24, 2016). *Tay, Microsoft's AI chatbot, gets a crash course in racism from Twitter*. Ed. by Elle Hunt. The Guardian. URL: https://www.theguardian.com/technology/2016/mar/24/tay-microsofts-ai-chatbot-gets-a-crash-course-in-racism-from-twitter (visited on 02/22/2019) (cit. on p. 25).

Thompson, Clive (2007). "I Chat, Therefore I Am..." In: *Discover* (2007): *Brain*. URL: http://discovermagazine.com/2007/brain (visited on 01/24/2019) (cit. on p. 5).

Turing, A. M. (Oct. 1950). "I. — Computing Machinery And Intelligence." In: *Mind* LIX (236), pp. 433–460. DOI: 10.1093/mind/LIX.236.433. URL: https://doi.org/10.1093/mind/LIX.236.433 (cit. on p. 5).

Ubisend (2016). *2016 Mobile Messaging Report*. Tech. rep. Ubisend. URL: https://insights.ubisend.com/2016-mobile-messaging-report (visited on 02/20/2019) (cit. on p. 26).

USA Today (Mar. 30, 2016). *Microsoft CEO Nadella: "Bots are the new apps"*. Ed. by Theresa Chong. USA Today. URL: https://eu.usatoday.com/story/tech/news/2016/03/30/microsof-ceo-nadella-bots-new-apps/82431672/ (visited on 01/28/2019) (cit. on p. 1).

VentureBeat (Sept. 14, 2017). *Facebook Messenger passes 300,000 bots*. Ed. by Khari Johnson. VentureBeat. URL: https://venturebeat.com/2017/09/14/facebook-messenger-passes-1-3-billion-monthly-active-users/ (visited on 01/24/2019) (cit. on pp. 5, 22).

VentureBeat (May 1, 2018). *Facebook Messenger passes 1.3 billion monthly active users*. Ed. by Khari Johnson. VentureBeat. URL: https://venturebeat.com/2018/05/01/facebook-messenger-passes-300000-bots/ (visited on 02/20/2019) (cit. on p. 22).

Weizenbaum, Joseph (Jan. 1966). "ELIZA — a Computer Program for the Study of Natural Language Communication Between Man and Machine." In: *Communications of the ACM* 9.1, pp. 36–45. ISSN: 0001-0782. DOI: 10.1145/365153.365168. URL: http://doi.acm.org/10.1145/365153.365168 (cit. on p. 3).

Xu, Anbang et al. (May 2017). "A New Chatbot for Customer Service on Social Media." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017 CHI Conference on Human Factors in Computing Systems, pp. 3506–3510. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025496 (cit. on pp. 10, 11).