

Bachelorarbeit

Titel der Arbeit // Title of Thesis

Konzeption und Entwicklung eines Sprachassistenten für den Einsatz in der Pflege

Conception and development of a voice assistant for the care sector

Akademischer Abschlussgrad: Grad, Fachrichtung (Abkürzung) // Degree
Bachelor of Science (B.Sc.)

Autorenname, Geburtsort // Name, Place of Birth
Nico Hülscher, Ahaus

Studiengang // Course of Study
Medieninformatik

Fachbereich // Department
Informatik und Kommunikation

Erstprüferin/Erstprüfer // First Examiner
Prof. Dr. Jens Gerken

Zweitprüferin/Zweitprüfer // Second Examiner
Prof. Dr. Andreas Heinecke

Abgabedatum // Date of Submission
13.06.2018

Eidesstattliche Versicherung

Hülscher Nico

Name, Vorname // Name, First Name

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem Titel

Konzeption und Entwicklung eines Sprachassistenten für den Einsatz in der Pflege

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Gelsenkirchen, den 12. Januar 2019

Ort, Datum, Unterschrift // Place, Date, Signature

Abstract

Das Ziel dieser Bachelorarbeit ist die Entwicklung eines funktionierenden Prototyps für ein Unternehmen im Bereich der Pflege. Das Ergebnis soll hilfsbedürftigen Menschen die Interaktion mit der Plattform Pflegix erleichtern. Zudem wird in dieser Arbeit generell auf Sprachassistenten im Bereich der Pflege eingegangen. Dabei wird der Fokus auf die Möglichkeiten von Sprachassistenten in diesem Bereich gesetzt.

Für den praktischen Teil dieser Arbeit wird auf die Analyse der Zielgruppe und des Zielsystems, sowie die darauf folgende Konzeption des *Skills* und die Entwicklungsphase näher eingegangen. Der Skill, der in dieser Arbeit entwickelt wurde, zeigt dabei, dass Sprachassistenten zwar noch immer einige Schwächen haben, diese jedoch in Zukunft durch immer weitere Entwicklungen abgeschwächt oder sogar in Stärken umgewandelt werden. Vor allem aber ist diese Arbeit ein Anreiz für Entwickler aller Bereiche, bereits bestehender Plattformen ein Stück weit oder auch vollständig in einen Sprachkontext zu bringen. Der Skill der aus dieser Arbeit hervorgeht zeigt eindrucksvoll, wie weit fortgeschritten Sprachassistenten in der heutigen Zeit bereits sind, wie schnell die Entwicklung für solche Plattformen möglich ist und auch wie gut die Einbindung in bereits vorhandene Systeme abläuft.

In Bezug auf einen Sprachassistenten im Bereich der Pflege zeigt diese Arbeit, dass Sprachassistenten ein Weg sein könnten, um ein Stück weit für Entlastung der Pflegenden zu sorgen. Insbesondere spielt die natürlichere Interaktion mit der Technik (im Vergleich zu Smartphones, Tablets oder Computern) eine große Rolle, gerade bei Nutzung durch ältere Menschen mit wenig technischem Verständnis.

Inhaltsverzeichnis

1 Einleitung	1
1.1 Aufbau der Arbeit	1
1.2 Vorstellung des Unternehmens Pflegix GmbH	1
2 Theoretischer Hintergrund	3
2.1 Ambient Assisted Living	3
2.2 Sprachassistenten	4
3 Analyse	7
3.1 Zielgruppenanalyse	7
3.2 Zielsystem	8
3.2.1 Rahmenbedingungen	8
3.2.2 Sprachassistenten	8
3.2.3 Detailanalyse Amazon Alexa	10
3.2.3.1 Möglichkeiten der Entwicklung	10
3.2.3.2 Fähigkeiten der Alexa-Plattform	10
3.2.3.3 Hardware	10
3.2.4 Werkzeuge	11
3.2.4.1 Amazon Developer Console	11
3.2.4.2 Entwicklungsumgebung (IDE)	12
3.2.4.3 Hardware	12
3.2.4.4 Backend	13
4 Konzeption	14
4.1 Plattform Pflegix	14
4.2 Zielbeschreibung	14
4.3 Erstentwurf	15
4.3.1 Tests der Interaktion	15
4.3.2 Machbarkeitsanalyse	15
4.4 Neukonzeption	15
4.4.1 Anpassung der Rahmenbedingungen	16
4.4.2 Interaktionsmodell	16
4.4.2.1 Nachrichtenabfrage	16
4.4.2.2 Notfallsystem	18

5 Entwicklung	21
5.1 Authentifizierung	21
5.1.1 OAuth2 Protokoll	21
5.1.2 Authentifizierungsserver	23
5.2 Amazon Developer Console	24
5.2.1 Interaktionsmodell	24
5.2.2 Account-Linking	26
5.3 Intent-Handling	28
5.3.1 Vorbereitungen	28
5.3.2 Kategorisierung der Intents	29
5.3.3 Umsetzung der Intents	29
5.3.3.1 Nachrichten-Intent	29
5.3.3.2 Anfrage-Intent	32
5.3.3.3 Notfall-Intent	34
5.3.3.4 Andere Intents	34
5.4 Roll-Out	35
5.4.1 Voraussetzungen	35
5.4.2 Maßnahmen	36
6 Diskussion	37
6.1 Ungeplante Umsetzungen innerhalb der Arbeit	37
6.2 Grenzen dieser Arbeit	37
6.3 Ausblick	38
7 Persönliches Fazit	40

Abbildungsverzeichnis

1	Verteilung der Pflege in Deutschland (Wiesbaden, 2017)	4
2	Ergebnisse der Befragung von Hellwig et. al. (Hellwig et al., 2018)	5
3	Statistiken zu smarten Lautsprechern	9
4	Hard- und Software für die Entwicklung eines Alexa-Skills	11
5	Visual Studio Code und Card in der Alexa App	12
6	Grundlegende Funktionen des Skills	16
7	Interaktionsablauf Nachrichtenabfrage	17
8	Interaktionsablauf Notfallsystem	20
9	Implicit Code Grant (Amazon.com Inc., 2018b)	22
10	Auth Code Grant (Amazon.com Inc., 2018b)	22
11	Infrastruktur Pflegix	24
12	Intents des Pflegix Skills	26
13	Account Linking	28
14	Slots des Intents <i>request_messages</i>	30
15	Beispieldialog der Nachrichtenabfrage	32

1 Einleitung

Diese Bachelorarbeit beschäftigt sich mit der Konzeption und Entwicklung eines Sprachassistenten-Skills (im Folgenden nur noch „Skill“ genannt) für das Unternehmen Pflegix GmbH. Mit dem Skill soll älteren und kranken Menschen die Möglichkeit gegeben werden, die Plattform Pflegix auf eine natürliche Art und Weise zu nutzen und mit anderen Nutzern dieser Plattform zu interagieren.

1.1 Aufbau der Arbeit

Der Schwerpunkt der Arbeit ist die Skill-Entwicklung sowie die Dokumentation der einzelnen Entwicklungsphasen. Hierzu gehören die Analyse-, Konzeptions- und technische Entwicklungsphase.

Die Analysephase formuliert die Rahmenbedingungen, die durch das Unternehmen, beziehungsweise die Plattform Pflegix und die technischen Möglichkeiten der Amazon Alexa Plattform gegeben sind. Dabei wird nicht nur eine Nutzeranalyse durchgeführt, sondern auch eine Analyse der technischen Möglichkeiten der Plattform, um einen Skill zu entwickeln. Darüber hinaus wird analysiert, inwieweit Sprachassistenten schon heute im Pflegebereich eingesetzt und genutzt werden können.

Des Weiteren wird die Konzeptionsphase der Entwicklung dokumentiert. Das Ziel der Konzeptionsphase ist ein sinnvolles Setup sowie ein fehlertolerantes Abfragesystem für den Sprachassistenten zu entwickeln.

In der technischen Entwicklungsphase wird auf die zugrundeliegenden Technologien und die Programmierung des im Konzeptionsteil beschriebenen Interaktionsmodells eingegangen. Zuletzt wird die Skill-Entwicklung des Sprachassistenten kritisch reflektiert und ein Ausblick über zukünftige Möglichkeiten von Sprachassistenten in der Pflege gegeben.

1.2 Vorstellung des Unternehmens Pflegix GmbH

Das Unternehmen wurde im September 2016 im Entrepreneurship Zentrum Witten (EZW), einem Gründungszentrum von der Universität Witten/Herdecke gegründet. Die Pflegix GmbH entwickelt eine gleichnamige Internetplattform (Pflegix GmbH) und hat bis zum heutigen Zeitpunkt (Stand November 2018) ein Netzwerk von über 10000 Helfern/Helferinnen (im Folgenden nur *Helper*) und einigen Hundert Hilfebedürftigen (im Folgenden *Familien*) aufgebaut. Der Kern der Plattform ist die Vermittlung von Helfern und Familien.

Nachdem Pflegix im August 2018 bereits das Start-Up HelloCare, einen Mitbewerber aus Hamburg, übernommen hat, schloss es im März 2018 die erste Finanzierungsrounde in sechsstelliger

Höhe ab. Derzeit beschäftigt Pflegix neun Mitarbeiter. Andreas Helget und Tim Kahrmann sind die Geschäftsführer des Unternehmens. Herr Helget übernimmt dabei die Leitung in Entwicklungsfragen, während Herr Kahrmann vor allem in der öffentlichen Arbeit aktiv ist. Zudem ist bei Pflegix ein Mitarbeiter als Community-Ansprechpartner tätig und derzeit arbeiten dort fünf Werkstudenten sowohl in der Entwicklung als auch im Marketing-Bereich.

Die Plattform Pflegix steht in erster Linie als Web-Applikation zur Verfügung. Darauf aufbauend wurde in den letzten Monaten eine mobile Android- und iOS-App entwickelt, welche die Web-App ergänzen und dem Nutzer die Plattform einfacher zugänglich machen soll.

2 Theoretischer Hintergrund

Die Pflege in Deutschland ist mit etwa 2,9 Millionen Pflegebedürftigen und 356.000 Beschäftigten in ambulanten Pflegediensten (Wiesbaden, 2017) ein sehr zentrales Thema. Sowohl die Politik als auch große Unternehmen suchen deshalb nach Lösungen, die deutsche Bevölkerung auch im Alter noch unterstützen zu können.

Prognosen des statistischen Bundesamts besagen für das Jahr 2050 eine Bevölkerungsanzahl von 68,5 Millionen in Deutschland. Davon werden etwa 40 Prozent älter als 60 Jahre sein (Grzegorzek et al. (2017)).

Diese alarmierenden Zahlen motivieren viele Forschergruppen, aber auch Start-Ups und Unternehmen, in diese Richtung zu forschen. So werden bereits humanoide Roboter für die Pflege erforscht (Grzegorzek et al. (2017)). Da diese Roboter im menschlichen Umfeld - im Gegensatz zu industriellen Robotern - mit ständig wechselnden Bedingungen umgehen müssen, gibt es derzeit noch kaum erfolgreiche Einbindungen für solche humanoiden Roboter in den Alltag. Generell gibt es viele Ansätze, die Pflege effizienter, effektiver und besser zu machen. Von diesen Ansätzen werden im Folgenden *Ambient Assisted Living* und darauf basierend die Sprachassistenten näher ausgeführt.

2.1 Ambient Assisted Living

Ambient Assisted Living (AAL) „bezeichnet die Unterstützung meist älterer oder benachteiligter Menschen im täglichen Leben durch intelligente Technik.“ (Gabler Wirtschaftslexikon Online, 2018a) Durch den hohen Anteil an Pflegebedürftigen, die zu Hause versorgt werden (73 Prozent, siehe Abbildung 1) und die dadurch entstehende hohe Belastung von Angehörigen, bietet AAL eine gute Perspektive zur Entlastung Angehöriger und ambulanter Pflegedienste. Ziel dieser Entlastung ist es, den Pflegebedürftigen im Alltag unabhängiger zu machen.

Viele ältere oder benachteiligte Menschen benötigen schon bei kleinen Aufgaben Unterstützung. Einige dieser kleinen Aufgaben können aber durch Techniken übernommen werden, sodass die hilfebedürftigen Personen in der Lage sind, auch ohne Hilfe anderer Personen, diese Aufgaben zu bewältigen. Einfache Beispiele sind die Fernsteuerungen für Steckdosen oder Lampen. Dadurch haben Personen mit eingeschränkter Mobilität die Möglichkeit, von vielen Positionen aus ihr Heim zu steuern.

Zusätzlich zu einfachen Steuerungsmöglichkeiten wird aber auch an speziellen Anwendungen, wie Systemen zur Sturzerkennung, automatischen Abschaltung beispielsweise des Herds oder auch Einbruchserkennungssystemen geforscht.

Alle diese Forschungen haben den Zweck, dem Pflegebedürftigen bei der Bewältigung vie-



1 Einschl. teilstationärer Pflegeheime.

Abbildung 1: Verteilung der Pflege in Deutschland (Wiesbaden, 2017)

ler kleinerer Aufgaben zu helfen und unabhängiger zu machen, aber auch, um ihm wieder mehr Sicherheit zu geben (z.B. durch Einbruchserkennungssystemen oder auch Systeme zur Sturzerkennung). Da derlei Anwendungen nicht oder nur selten von den Krankenkassen übernommen werden, ist die Bereitschaft der Zielgruppe gering dafür zu zahlen. Daraus resultiert zwangsläufig ein Absatzproblem.

Somit ist es sinnvoll, zu versuchen unterstützende Technologien zu entwickeln, die mit derzeitigen Technologien vereinbar sind, sodass die Hemmschwelle zur Nutzung der Systeme im Vergleich zu neu entwickelten Systemen geringer ausfällt. Eine Option wären daher Sprachassistenten

2.2 Sprachassistenten

Eingeschränkte Personen benötigen oft schon bei kleinen Aufgaben Hilfe. Viele dieser Aufgaben können heute schon automatisiert werden, sodass keine Hilfe von professionellen oder informellen Pflegekräften benötigt werden würde. Eine Möglichkeit zur Steuerung dieser Systeme sind Sprachassistenten.

Das Gabler Wirtschaftslexikon definiert einen Sprachassistenten als „ein Dialogsystem, das Anfragen der Benutzer beantwortet und Aufgaben für sie erledigt, in privaten und wirtschaftlichen Zusammenhängen. Er ist auf dem Smartphone ebenso zu finden wie in Unterhaltungsgeräten und in Fahrzeugen“ (2018b).

Sprachassistenten bieten erweiterte Funktionalität in Form von Skills an. Diese werden ähnlich

einer Smartphone App von einer dritten Partei entwickelt und bieten zusätzlich zu den initial verfügbaren Funktionen eines Sprachassistenten weitere Möglichkeiten zur Interaktion. Damit bieten sie sich als universelles Werkzeug für die Heimsteuerung an.

Für die Steuerung des Eigenheims gibt es bereits zahlreiche Systeme von Drittherstellern, die es dem Nutzer ermöglichen, diese per Sprachassistent zu steuern. Das gilt zum Beispiel für Lichtsysteme wie Philips Hue Lampen oder das IKEA TRÅDFRI System. Die Möglichkeiten der Heimautomatisierung sind nahezu unbegrenzt und reichen von Lichtquellen- über Thermostat- beziehungsweise Heizungssteuerung bis hin zur Verwaltung von Alarmanlagen. Jedes Jahr kommen neue Systeme auf den Markt, die einzigartige Funktionen bieten. Zudem sind die Funktionen nicht auf die Steuerung des Eigenheims beschränkt. Auch software-spezifische Funktionen wie das Erstellen von Erinnerungen, Senden von Nachrichten oder auch die Beantwortung einfacher Fragen sind im Bereich des derzeit Möglichen.

Eine Voraussetzung für die Einführung einer Technologie ist, dass diese vom Nutzer akzeptiert wird (Renaud and van Biljon, 2008). Das Technology Acceptance Model (TAM) von Peek (Peek, 2017) definiert dafür vier wichtige Faktoren: Nützlichkeit, Benutzerfreundlichkeit, Einstellung zur Technologie und Erwartungseinhaltung.

Dazu befragten Andre Hellwig, Caroline Schneider, Dr. Sven Meister und Prof. Dr. Wolfgang Deiters in einer repräsentativen Studie mit 27 Teilnehmern (Hellwig et al., 2018), wie die Teilnehmer (Alter zwischen 50 und 95 Jahren) zu Sprachassistenten stehen.

Die Ergebnisse dieser Studie (siehe Abbildung 2) zeigen, dass bereichsübergreifend über 59 Prozent der

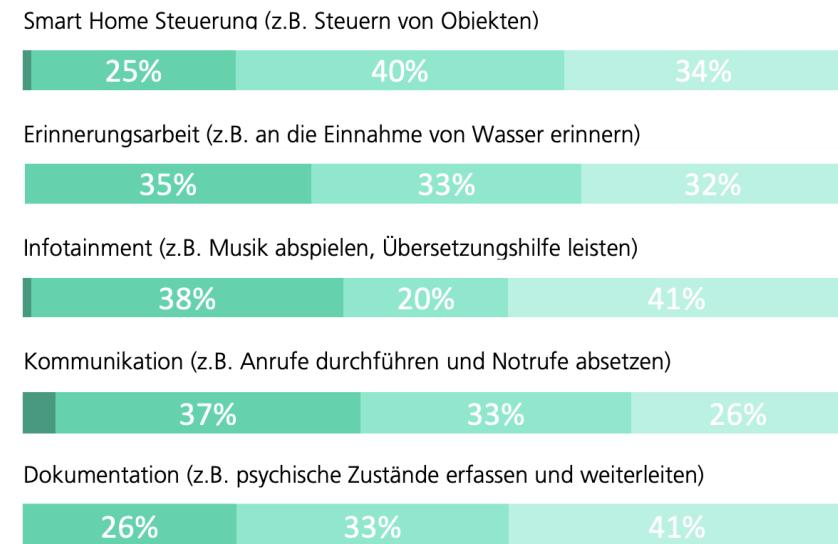


Abbildung 2: Ergebnisse der Befragung von Hellwig et. al. (Hellwig et al., 2018)

befragten Personen Sprachassistenten generell nutzen würden. Vor allem im Bereich der Kommunikation werden Sprachassistenten bereits genutzt. Im Allgemeinen ist die Studie ein guter Indikator dafür, wie gut die Systeme angenommen werden.

Ein oft angesprochener Aspekt im Zusammenhang mit Sprachassistenten, insbesondere solchen, die auf Zuruf die Sprache aufnehmen, ist der Datenschutz. Kritiker bemängeln, dass

der Nutzer nur begrenzt Zugriff auf die von den Sprachassistenten gesammelten Daten hat und nicht sicherstellen kann, dass wirklich nur die vom Nutzer speziell an den Sprachassistenten gerichteten Sätze erkannt und verarbeitet werden (Jackson and Orebaugh (2018)). Daher ist der undurchsichtige Datenschutz ein häufig genannter Kritikpunkt, wenn Personen auf virtuelle Assistenten angesprochen werden. Es ist jedoch anzumerken, dass jeder derzeit erhältliche Sprachassistent die Möglichkeit bietet, das Mikrofon abzuschalten. Durch diese Maßnahme ist jegliche Interaktion mit dem Sprachassistenten unterbunden und es werden keine Gespräche oder Gesprächsteile aufgezeichnet. Zu diesem Thema gibt beispielsweise Amazon folgendes Statement: „Echo-Geräte sind darauf ausgerichtet, nur das von Ihnen ausgewählte Aktivierungswort (Alexa, Amazon, Computer oder Echo) zu erkennen. Das Gerät erkennt das Aktivierungswort durch die Identifikation von akustischen Mustern, die das Aktivierungswort ausmachen. Ohne Erkennung des Aktivierungsworts durch das Gerät (oder die Aktivierung von Alexa durch eine Taste) werden Sprachaufzeichnungen nicht gespeichert oder in die Cloud geleitet.“(Amazon.com Inc., 2018b). Die Anbieter garantieren also dafür, dass Nutzer die Kontrolle über das Gerät haben und nicht umgekehrt. Studien zeigen, dass die Datenschutzrichtlinien der Unternehmen eingehalten werden und nur dann aufgezeichnet wird, wenn auch ein Sprachbefehl an die Geräte gerichtet wird (Perez et al. (2018)). Zusätzlich bieten die Unternehmen auch die Möglichkeit, aufgezeichnete Dialoge mit den Sprachassistenten einzusehen und bei Bedarf zu löschen. Da die Systeme allerdings darauf ausgerichtet sind, aus den Sprachaufnahmen des Nutzers zu lernen, um so das Dialogerlebnis der Spracherkennung zu verbessern, kann ein striktes Löschen jeder aufgenommenen Nachricht an den Assistenten zu einer suboptimalen Arbeitsweise des Gerätes führen.

Studien haben des Weiteren gezeigt, dass Sprachassistenten durch für den Menschen nicht-hörbare Töne angesprochen werden können (Song and Mittal (2017)). Das bietet beispielsweise für Macher von Werbungen oder Angreifer neue Möglichkeiten, Geräte zu manipulieren und anzugreifen.

Zusammenfassend bieten Assistenzsysteme ein enormes Potential, das Leben der Menschen zu vereinfachen. Da allerdings oft die Akzeptanz der Endnutzer fehlt, kommen viele der entwickelten Systeme nie in der breiten Bevölkerung an. Stattdessen setzen sich nur vereinzelt einige wenige Technologien durch. Eine dieser Technologien ist der virtuelle Assistent. Virtuelle Assistenten oder Sprachassistenten bieten dem Nutzer die Möglichkeit, auf natürliche Weise zu interagieren um an Informationen zu gelangen oder Nachrichten zu versenden. Daher bilden sie die Grundlage für diese Arbeit, die am Ende der Entwicklung einen Zugewinn für die Nutzer der Plattform Pflegix darstellen soll. Zunächst folgt nun eine Detailanalyse der Zielgruppen und des Zielsystems für dieses Projekt.

3 Analyse

Die Entwicklung einer Anwendung für einen Sprachassistenten bedarf - durch die starke Nutzernähe - einer sorgfältigen Analyse des Endnutzers. Zudem soll die Anwendung in diesem Fall eine sinnvolle Ergänzung zum bestehenden *Stack* bieten und sich dort eingliedern lassen können.

3.1 Zielgruppenanalyse

Für die Plattform Pflegix ergeben sich drei potentielle Nutzergruppen:

1. Helfer:

Der Helfer pflegt und betreut die Pflegebedürftigen. Die Betreuung wird meist vor Ort bei den Familien durchgeführt. Dazu kommen Aufgaben, die nicht lokal durchgeführt werden können, wie zum Beispiel das Tätigen des Wocheneinkaufs oder die Begleitung zu Arztbesuchen. Da ein Sprachassistent primär stationär genutzt wird (Amazon Echo), besteht für den Helfer während der Pflegetätigkeit nur ein geringes Nutzerpotential. Zwar können viele Sprachassistenten auch als App auf dem Smartphone genutzt werden, jedoch gibt es für diese Plattformen inzwischen die mobile App, welche im Vergleich zu einem solchen Skill einen breiteren Umfang bietet. Somit läge das meiste Potential für den Helfer darin, zu Hause einen Sprachassistenten zur Durchführung grundlegender Interaktionen zu nutzen (Nachrichten abrufen und senden).

2. Angehörige:

Der Angehörige/Die Angehörige (im Folgenden nur der Angehörige) übernimmt den technischen Interaktionsanteil auf der Plattform. Er verwaltet Bewerbungen von Helfern, schickt Anfragen, macht Termine aus und übernimmt die Kommunikation auf digitaler Ebene. Er wird in der Regel nicht vor Ort sein, wenn der Helfer der pflegebedürftigen Person Hilfe leistet. Somit besteht das Nutzungspotential auch für den Angehörigen darin, grundlegende Funktionen der Plattform zu nutzen. Denkbar wäre auch ein Kommunikationskanal für die pflegebedürftige Person und den Angehörigen.

3. Pflegebedürftige:

Der Pflegebedürftige wird in der Regel zu Hause betreut, also in einer Umgebung, für die Sprachassistenten in erster Linie entwickelt und in der sie zu einem Großteil auch eingesetzt werden (Statista, 2017b). Er hat möglicherweise eine technische Affinität, es muss jedoch damit gerechnet werden, dass technisch wenig bis keine Vorkenntnisse bei den Pflegebedürftigen vorhanden sind, beziehungsweise deren Angehörige nach der Hilfe für sie suchen. Sprachassistenten sind dabei für Senioren, beziehungsweise weniger technikaffine Menschen eine Technik mit einer geringeren Einstiegshürde als Smartphones, Handys, Tablets oder Computer, da

Sprachassistenten, wie der Name suggeriert, assistieren. Sie bieten eine vergleichsweise hohe Fehlertoleranz. Zudem muss sich der Nutzer nicht mehr merken, als das *Wake-Word*, also das Aktivierungswort („Alexa“, „Hey Siri“, Hey Cortana“, „Ok Google“) sowie den Namen des Skills (hier „Pflegix“). Sind diese Begriffe bekannt, ist es möglich mit dem Skill zu interagieren und auch bei Fehlern eine Hilfestellung zu bekommen, die daran erinnert, welche Funktionen der Skill dem Nutzer bietet.

Somit besteht bei dieser Nutzergruppe das höchste Potential, da die Pflegebedürftigen durch den Skill möglicherweise weniger auf ihre Angehörigen angewiesen sind und somit auch selbst die Kommunikation mit den Helfern übernehmen können. Zudem wäre es denkbar, dass über die Plattform Notfallkontakte eingespeichert werden, die bei Sturz oder Bewegungsunfähigkeit des Pflegebedürftigen kontaktiert werden können.

3.2 Zielsystem

Die Wahl des technischen Systems zur Entwicklung eines Sprachassistenten-Skills hängt, gerade wenn dieser als Ergänzung zu einem bestehenden Technologie-Stack dienen soll, stark von den bereits eingesetzten Systemen ab. Zusätzlich ist es notwendig, auch die Verbreitung der in Frage kommenden Sprachassistenten zu analysieren und deren Zugänglich- und Bedienbarkeit miteinander zu vergleichen. Daher werden zunächst Randbedingungen definiert, die die Anwendungsziele des Skills eingrenzen.

3.2.1 Rahmenbedingungen

Wie bereits in Kapitel 1 erläutert, dient der Skill dazu, Funktionen der Plattform Pflegix, allen voran die Nachrichtenübermittlung, für den Nutzer, per Sprache zugänglich zu machen. Daher muss der Skill von Beginn an so gestaltet werden, dass der Endanwender auch ohne besondere Vorkenntnisse in der Lage ist, alle Funktionen des Skills zu nutzen. Gerade bei weniger technikaffinen Personen kann eine zu geringe Fehlertoleranz des Systems zu einem hohen Grad an Ablehnung führen. Da Ablehnung den Willen zur Verwendung jeder Technologie deutlich mindert, muss der Skill so aufgebaut werden, dass der Nutzer nie das Gefühl eines Kontrollverlustes vermittelt bekommt und dass der Skill erwartungskonform handelt.

3.2.2 Sprachassistenten

Derzeit befinden sich vier größere Sprachassistenten auf dem Markt. Der bekannteste Sprachassistent Alexa von Amazon, der Google Assistant, Microsoft's Cortana und Siri von Apple. Generell verwendet jeder dieser Assistenten ein individuelles Aktivierungswort. Während man

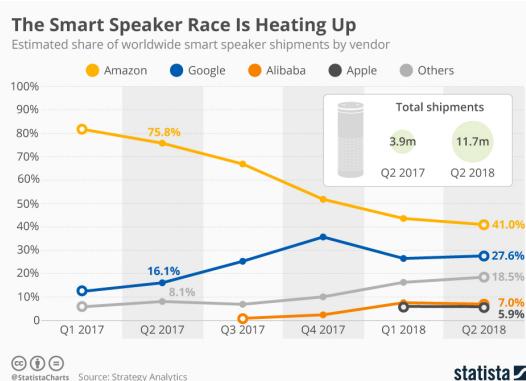
bei Alexa zwischen „Alexa“, „Echo“, „Amazon“ und „Computer“ wählen kann, ist es beim Google Assistant „Ok Google“, bei Microsoft’s Cortana „Hey Cortana“ und bei Siri von Apple „Hey Siri“.

Alle Systeme bauen auf drei Säulen auf: Der Hauptfokus liegt auf Beantwortung normaler Suchanfragen wie: „Wie wird das Wetter heute?“ oder „Wann wird die Uhr umgestellt?“. Ein weiterer Kern der Produkte ist die Steuerung von smarten Geräten, wie Glühbirnen, Heizthermostaten und Steckdosen. Die dritte Säule ist die Integration von anderen Herstellern – sowohl durch Skills, als auch durch direkte Hardware-/Protokollunterstützung.

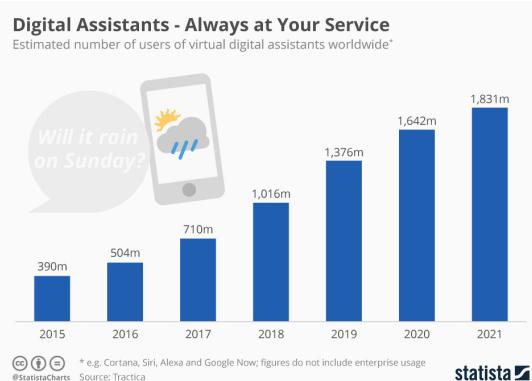
Durch die kostenlose Verfügbarkeit und einem einfachen Zugriff auf die Entwicklerfunktionen sieht man in den Skill-Stores einen sehr starken Anstieg der verfügbaren Skills (Statista, 2017a). Durch die starke globale Zunahme der Nutzung von smarten Lautsprechern mit integrierten Sprachassistenten (Statista, 2018a) sind die Chancen für Entwickler sehr positiv.

Der noch junge Markt für smarte Lautsprecher ist sehr volatil. War die Produktreihe der Echo-Lautsprecher von Amazon im zweiten Quartal 2017 noch mit 75,8 Prozent die meistverkaufte und -genutzte Plattform, so waren es im zweiten Quartal 2018 nur noch 41,0 Prozent (siehe Abbildung 3a). Umgekehrt nehmen die anderen smarten Lautsprecher an Marktvolumen zu sodass sich der Abstand untereinander zunehmend verringert.

Derzeit ist die Amazon-Echo Plattform am stärksten vertreten (41,0%), gefolgt von den smarten Lautsprechern von Google mit 27,6 Prozent. Dabei ist zu berücksichtigen, dass Apple mit seinem HomePod erst seit dem ersten Quartal 2018 verkauft wird und daher noch weniger Nutzer hat, wie einige andere Hersteller.



(a) Anteil an den globalen Verkäufen (Statista, 2018b)



(b) Voraussichtliche Nutzerzahlen global (Statista, 2016)

Abbildung 3: Statistiken zu smarten Lautsprechern

Generell ist das Wachstum im Bereich der smarten Lautsprecher enorm: Während im zweiten Quartal 2017 „nur“ 3,9 Millionen Geräte herstellerübergreifend verkauft wurden, so waren es

im zweiten Quartal 2018 bereits 11,7 Millionen Geräte. Zudem werden bis 2021 voraussichtlich 1,83 Milliarden Menschen Nutzer von Sprachassistenten sein (siehe Abbildung 3b). Dabei muss allerdings erwähnt werden, dass die Statistik der voraussichtlichen Nutzerzahlen nicht nur die stationären smarten Lautsprecher aufzählt, sondern auch mobile Sprachassistenten einbezieht (Google Assistant auf Android- und Siri auf iOS-Geräten). Dennoch kann aus der in Abbildung 3b dargestellten Steigerung ein enormes Wachstumspotential für Skills abgeleitet werden. Trotz aktuell noch geringer Akzeptanz und Nutzerzahl bedeutet dies für den Pflegix-Skill, dass die zunehmende Nutzung von Sprachassistenten von allen Nutzergruppen die Entwicklung und Einführung eines solchen Skills als sinnvoll und erfolgversprechend erscheinen lassen.

3.2.3 Detailanalyse Amazon Alexa

Vor allem aufgrund des Marktvolumens, wurde die Entscheidung getroffen, einen Skill für die Alexa-Plattform zu entwickeln. Diese hat die derzeit größte Nutzerzahl und stellt damit vermutlich die meisten Geräte für die Nutzer der Plattform Pflegix.

3.2.3.1 Möglichkeiten der Entwicklung

Skills für Amazon Alexa können auf unterschiedliche Arten entwickelt werden. Möglich ist die Nutzung von Amazon's eigenem Dienst „AWS Lambda“, einer *serverless* Plattform. Sie bietet die Möglichkeit, in verschiedenen Programmiersprachen Code zu schreiben und diesen event-basiert ausführen zu lassen. Alternativ kann auf die von Amazon zur Verfügung gestellten Alexa-Skills-Kit-SDKs zurückgegriffen werden. Diese sind für verschiedene Technologien verfügbar (Java, Node.js, Python) und beinhalten Methoden und Funktionen, um mit den Server-Anfragen des Skills umzugehen.

Durch den Stack, der bereits bei Pflegix vorhanden ist (Node.js-Server), entschied man sich für das Alexa-Skills-Kit für die Node.js Plattform.

3.2.3.2 Fähigkeiten der Alexa-Plattform

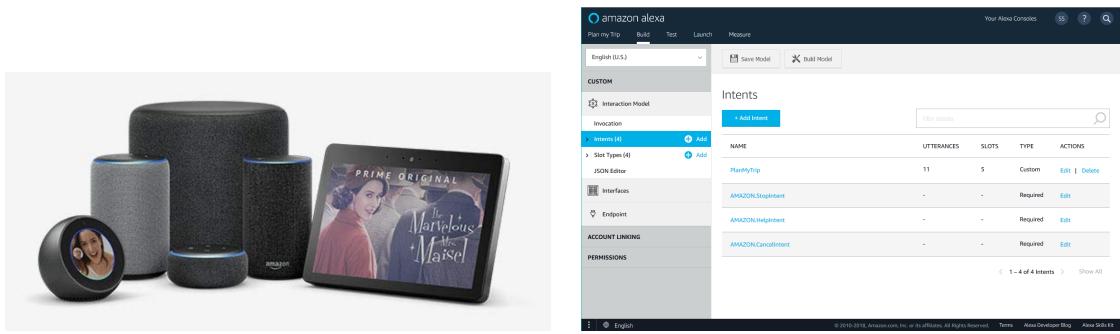
Alexa bietet viele komfortable Möglichkeiten, um bereits vorhandene Plattformen sinnvoll zum Einsatz zu bringen. Es werden beispielsweise die Möglichkeiten geboten, Benachrichtigungen an Alexa-Hardware zu senden oder auch auf die Einkaufsliste des Nutzers zuzugreifen. Welche Möglichkeiten davon genutzt werden sollen, wird in Kapitel 4 näher ausgeführt.

3.2.3.3 Hardware

Amazon bietet für seinen digitalen Assistenten verschiedene Hardwareausführungen (siehe Abbildung 4a) an. Neben einem kleinen, nur für einfache Interaktion gedachten smarten

Lautsprecher Echo Dot, gibt es größere Ausführungen mit besserem Lautsprecher (Echo und Echo Plus). Des Weiteren wird auch Hardware mit Display angeboten (Echo Spot und Echo Show).

Außerdem kann, wie bereits in Kapitel 3.1 erwähnt, die Alexa App auf dem Smartphone (Android, iOS) für die Interaktion mit Alexa genutzt werden. Dabei ist anzumerken, dass der Fokus der Entwicklung auf dem lokal positionierten Gerät liegt, da, wie später im Kapitel 4 näher erläutert, dies der Kernbereich für die Nutzung des Skills sein wird.



(a) Alexa ProduktFamilie (Amazon.com Inc., 2018b)

(b) Ansicht der Developer Console (Amazon.com Inc., 2018a)

Abbildung 4: Hard- und Software für die Entwicklung eines Alexa-Skills

3.2.4 Werkzeuge

Für die Entwicklung eines Alexa Skills sind verschiedene Werkzeuge im Einsatz. Zusätzlich zu einer Entwicklungsumgebung, erfordert die Entwicklung einen Amazon Developer Account, der den Skill bei Amazon erstellt und verwaltet. In der Developer Console von Amazon (siehe Abbildung 4b) werden für den Skill alle einfachen Einstellungen getroffen. Für Tests sollte zudem eine Alexa, eine Alexa App oder auch ein Alexa-Emulator zur Verfügung stehen. Zuletzt wird ein *Backend* benötigt, dass die Fragen verarbeitet und die Rückmeldung an das Alexa Gerät sendet.

3.2.4.1 Amazon Developer Console

Die Amazon Developer Console (Abbildung 4b) bietet die grundlegenden Funktionen an, die zum späteren Zeitpunkt in der Entwicklung für eine Verzahnung von Sprachinterface und Backend sorgen. In erster Linie dient die Plattform dazu, mögliche Sprachbefehle zu definieren, die später im Backend verarbeitet und beantwortet werden. Zusätzlich dazu werden grundlegende Konfigurationen eingestellt. Ein Beispiel dafür ist die URL des API-Endpunktes oder der Authentifizierungs-Endpunkt. Der API-Endpunkt verarbeitet die Anfragen des Nutzers,

während der Authentifizierungs-Endpunkt die Verknüpfung des Amazon-Accounts mit dem Pflegix-Konto bearbeitet. Außerdem bietet Amazon hier die Möglichkeit, verschiedene Interfaces für den Skill freizuschalten. So kann dort beispielsweise eingestellt werden, dass der Skill einen Bildschirm unterstützt oder Zugriff auf die Listen (zum Beispiel die Einkaufsliste) des Nutzers erfordert.

3.2.4.2 Entwicklungsumgebung (IDE)

Dadurch, dass die Entwicklung des Skills mit Node.js (Node.js Foundation, 2018b) realisiert wird, grenzt sich der Bereich an sinnvoll einsetzbaren IDEs ein. Sinnvoll für die Entwicklung ist eine IDE, die das Debugging eines Node.js beziehungsweise in diesem Fall eines Express.js Servers (Node.js Foundation, 2018a) ermöglicht und für einen optimalen Workflow Autovervollständigung bietet. Die Wahl fiel dabei letztendlich auf Visual Studio Code (siehe Abbildung 5a) von Microsoft. Diese IDE zeichnet sich dadurch aus, dass es eine große Menge an Plug-Ins (Erweiterungen) gibt, die nicht nativ enthaltene Funktionen problemlos nachrüsten, das ganze Programm open-source ist und kostenlos genutzt werden kann.

3.2.4.3 Hardware

Für reale Testbedingungen ist es sinnvoll, ein entsprechendes Endgerät vor Ort zu haben. Für einfache Tests hat sich die App als sinnvoll erwiesen. Diese bietet sowohl die Fähigkeit, einfache Sprachbefehle zu verarbeiten als auch die „Cards“ genannten visuellen Rückmeldungen (siehe Abbildung 5b) anzulegen. Für den Test unter realen Bedingungen mit Testpersonen sollte allerdings nur auf einen Amazon Echo zurückgegriffen werden, da der Fokus des Skills auf der Sprache und nicht dem Bildschirm liegen soll.

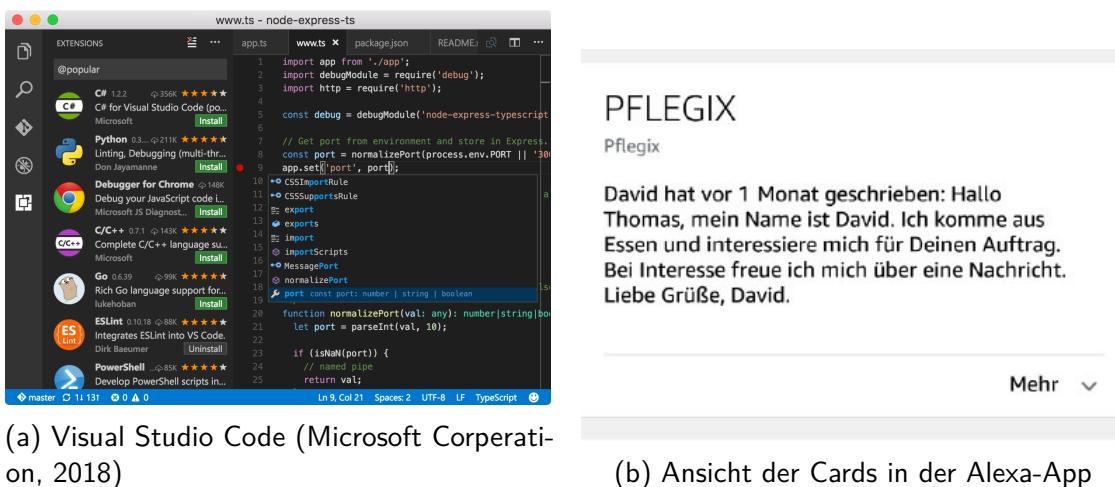


Abbildung 5: Visual Studio Code und Card in der Alexa App

3.2.4.4 Backend

Wie bereits im IDE-Unterkapitel erwähnt, kommt ein Express.js Server zum Einsatz. Express.js ist ein Framework, dass auf Node.js aufbaut und für Webapplikationen grundlegende Funktionen eines Webservers bereitstellt. Für den hier entwickelten Skill wurde Node.js in der Version 10.0.0 und Express.js in der Version 4.10 verwendet. Dieser Server dient sowohl zur Authentifizierungs- als auch zur Anfrageverarbeitung mit Zugriff auf die relevanten Datenbankinformationen. Da bei Pflegix für das Backend der Website und App derzeit ein Express.js Server genutzt wird, werden bei fortgeschrittener Entwicklung die Alexa-Endpunkte in das Hauptbackend integriert. Für die reibungslose Entwicklung ist es sinnvoll vorerst einen leichteren Server aufzusetzen, da so potentielle, mit anderen Paketen zusammenhängende, Fehler vermieden werden können. Durch den Dienst Ngrok (Shreve, 2018) wird der auf dem Entwicklungsgerät ausgeführte Server vom Internet aus, durch dynamische Adresszuweisung, erreichbar gemacht. Dies ermöglicht erst die lokale Entwicklung eines Alexa-Skills.

Zusätzlich zum Express.js Server ist eine lokale Instanz der Pflegix-Datenbank notwendig. Pflegix nutzt dafür eine MongoDB-Datenbank der Version 4.0.5-build.1 (MongoDB Inc, 2018). Zugriffe vom Server auf diese Datenbank werden durch Mongoose.js (Mongoose, 2018) ermöglicht, welches Serveranfragen direkt in Objekte überführen kann.

4 Konzeption

Anhand der in der Analyse beschriebenen Rahmenbedingungen wird in diesem Kapitel ein erster Konzeptionsentwurf realisiert. Anhand dieses ersten Entwurfs soll das Projekt auf Machbarkeit untersucht werden, damit es dann im Anschluss konzeptioniert und ausgearbeitet wird.

4.1 Plattform Pflegix

Damit ein Konzept für den Skill ausgearbeitet werden kann, muss zunächst einmal die Plattform Pflegix genauer untersucht werden. Auf der Plattform Pflegix gibt es zwei Nutzergruppen: Helfer und Familien. Familien werden Nutzer genannt, die entweder für sich selbst oder für Angehörige nach Hilfe suchen. Helfer sind dementsprechend die Personen, die ihre Hilfe für Familien anbieten. Die Bereiche, in denen Hilfe angeboten oder gesucht wird, reichen von Einkaufs- über Technikhilfe bis hin zur Grundpflege. Pflegix möchte dabei keine Konkurrenz zu stationärer Betreuung sein, sondern eher die oft überforderten ambulanten Pflegedienste entlasten.

Über die Plattform Pflegix haben die Nutzer die Möglichkeit, die Inserate von Hilfesuchenden zu durchstöbern (Helfer) beziehungsweise die Helfer für die eigenen Aufträge im „Marktplatz“ zu finden (Familien). Haben die Nutzer einen passenden Eintrag gefunden, können sie einen ersten Kontakt aufbauen. Darauf folgt dann ein erstes Kennenlerntreffen oder auch direkt ein Auftrag. Dieser Auftrag wird von den Familien in einen Rahmen gesetzt, beispielsweise drei Einsätze pro Woche mit je 4 Stunden im Bereich der Grundpflege und Haushaltshilfe. Ist ein Auftrag aktiv, erfassen Helfer ihre geleisteten Stunden über die App oder das Webinterface. Daraus wird am Ende des Monats eine automatische Rechnung erstellt, die der Familie zugesandt wird. Als Provision erhält Pflegix dabei einen kleinen Prozentsatz der gezahlten Beiträge.

4.2 Zielbeschreibung

Aus den Ergebnissen der Analyse sowie den durch die Plattform Pflegix gegebenen Faktoren, lassen sich die Zielbeschreibungen für das Projekt definieren.

Der Skill soll in erster Linie den eingeschränkten und hilfesuchenden Nutzern der Plattform den Zugang erleichtern. Dabei soll es möglich sein, über den Skill Nachrichten an bekannte Helfer zu senden sowie Anfragen an diese zu verschicken. Des Weiteren soll es für die pflegebedürftigen Nutzer möglich sein, Notfallkontakte im System zu hinterlegen, um diese in einem solchen Fall per Sprache benachrichtigen zu können. Durch spezielle Hilfe- und Unterstützungsfunktionen soll der Zugang zu dieser Technologie vereinfacht werden. Wünschenswert wäre zudem, wenn der Skill eine Kommunikation in Echtzeit zwischen den Parteien ermöglichen könnte.

4.3 Erstentwurf

Als Erstentwurf soll eine vereinfachte Nachrichtenabfrage und -versendung möglich sein. Hierzu wurde in der AWS Lambda Console in Node.js ein erstes Interaktionsmodell realisiert. Dieses soll eine Grundstruktur für die Nachrichtenabfrage bieten und einen ersten Eindruck geben, wie zuverlässig der Alexa-Spracherkennungsdienst die gesprochenen Nachrichten erkennt. Um das zu realisieren, wurden einige wenige Skill-Funktionen (*Intents*) erstellt. Die Intent bestehen aus vordefinierter Sätzen, welche diesen Intent aufrufen sollen. Als Skill-Funktionen wurden sowohl die Nachrichtenabfrage („Habe ich neue Nachrichten“) als auch das Nachrichten senden („Schreibe Testnachricht“) für den Erstentwurf realisiert.

Sofern mehrere Nachrichten von unterschiedlichen Personen empfangen wurden, soll der Nutzer die Wahl haben, die Nachricht einer bestimmten Person abzufragen. Hierzu soll Alexa nachfragen, welche Nachricht nun vorgelesen werden soll.

4.3.1 Tests der Interaktion

Erste Tests des Erstentwurfs zeigten, dass Alexa das gesprochene Wort gut in Schrift umwandeln konnte und die Nachrichten auf diese Weise zuverlässig erkannt wurden. Auch funktionierte die Abfrage der Demo-Nachrichten zuverlässig.

Nicht ganz so zuverlässig funktionierte die Abfrage bei mehreren Absendern. Alexa erkannte oft genutzte Namen recht zuverlässig, jedoch wurden oft andere Schreibweisen von den Namen erkannt („Stefan“ statt „Stephan“), sodass ein einfaches Abgleichen der Namen nicht möglich war.

4.3.2 Machbarkeitsanalyse

Der erste Test des Demo-Setups zeigte, dass eine Nachrichtenabfrage zuverlässig und einfach zu handhaben ist. Auch das Senden der Nachrichten funktioniert zuverlässig, jedoch mit kleineren Abstrichen (Alexa formatiert alles in kleingeschriebenen Text und hat Probleme mit der Zeichensetzung). Schwieriger wird das richtige Erfassen der Namen bei der Auswahl mehrerer Nachrichten. In diesem Fall muss ein Back-Up eingebaut werden, sodass auch diese Funktion zuverlässig gesteuert werden kann. Generell ist die Umsetzung der gewünschten Funktionen für einen Skill sehr gut machbar.

4.4 Neukonzeption

Durch die gesammelten Erfahrungen des Erstentwurfs wird das Konzept im Folgenden überarbeitet und weiter ausgebaut. Dazu werden die Rahmenbedingungen angepasst und das

Interaktionsmodell für den Skill erstellt.

4.4.1 Anpassung der Rahmenbedingungen

Um einen Skill zu entwickeln ist es notwendig, Rahmenbedingungen zu definieren. Die Rahmenbedingungen dienen dem Zweck, den Skill möglichst einfach und verständlich für den Nutzer zu machen. Gleichzeitig müssen technische Beschränkungen in der Konzeption beachtet werden. Der Skill soll primär den Hilfesuchenden zur Verfügung stehen. Daher ist hier die Rahmenbedingung klar gegeben: Ältere oder beeinträchtigte Personen mit teils eingeschränkter Mobilität. Durch die Plattform Pflegix in ihrem derzeitigen Stand sind zudem weitere Limitierungen oder Rahmenbedingungen gegeben: Es gibt die Möglichkeit Nachrichten abzurufen und zu senden sowie Helfer (von der Hilfesuchenden-Seite gesehen) anzufragen. Da der Anwender des Skills die hilfesuchende Person sein soll, fallen weitere Plattform-Funktionen wie die Zeiterfassung weg. Der Notruf wird im Rahmen dieser Arbeit ebenfalls nicht behandelt, ist stellt somit den Umfang des Skills in übergeordneten Kategorien dar (siehe Abbildung 6), auf dem das Interaktionsmodell aufbaut.

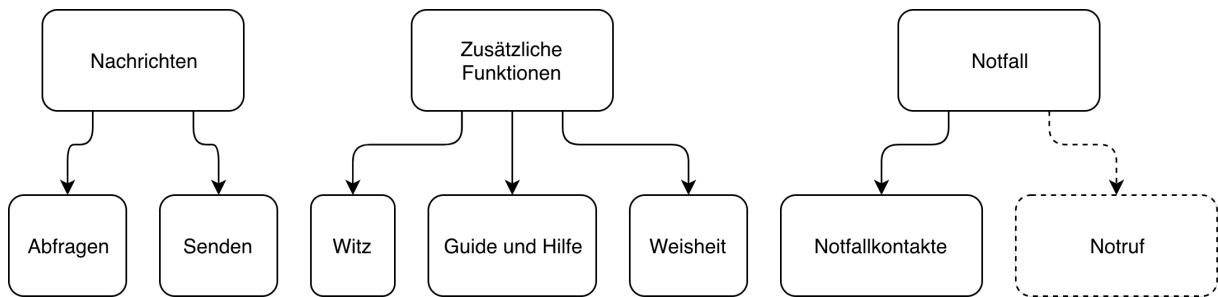


Abbildung 6: Grundlegende Funktionen des Skills

4.4.2 Interaktionsmodell

Da das Interaktionsmodell in vollständiger Größe zu umfangreich ist, um dieses hier komplett aufzuführen, wird im Folgenden nur auf Teilbereiche der vollständigen Interaktion eingegangen. Diese sind die Nachrichtenabfrage, das Notfallsystem und in groben Zügen auch der Guide. Wie in den Abbildungen 7 und 8 zu sehen ist, soll der Skill für den Nutzer sehr einfach zu nutzen sein. Der Nutzer soll dabei möglichst einfache Fragen von Alexa gestellt bekommen, auf die meist mit einem Wort geantwortet werden kann.

4.4.2.1 Nachrichtenabfrage

Um die Nachrichtenabfrage zu starten, wird ein Intent als Startpunkt benötigt. Dieser kann

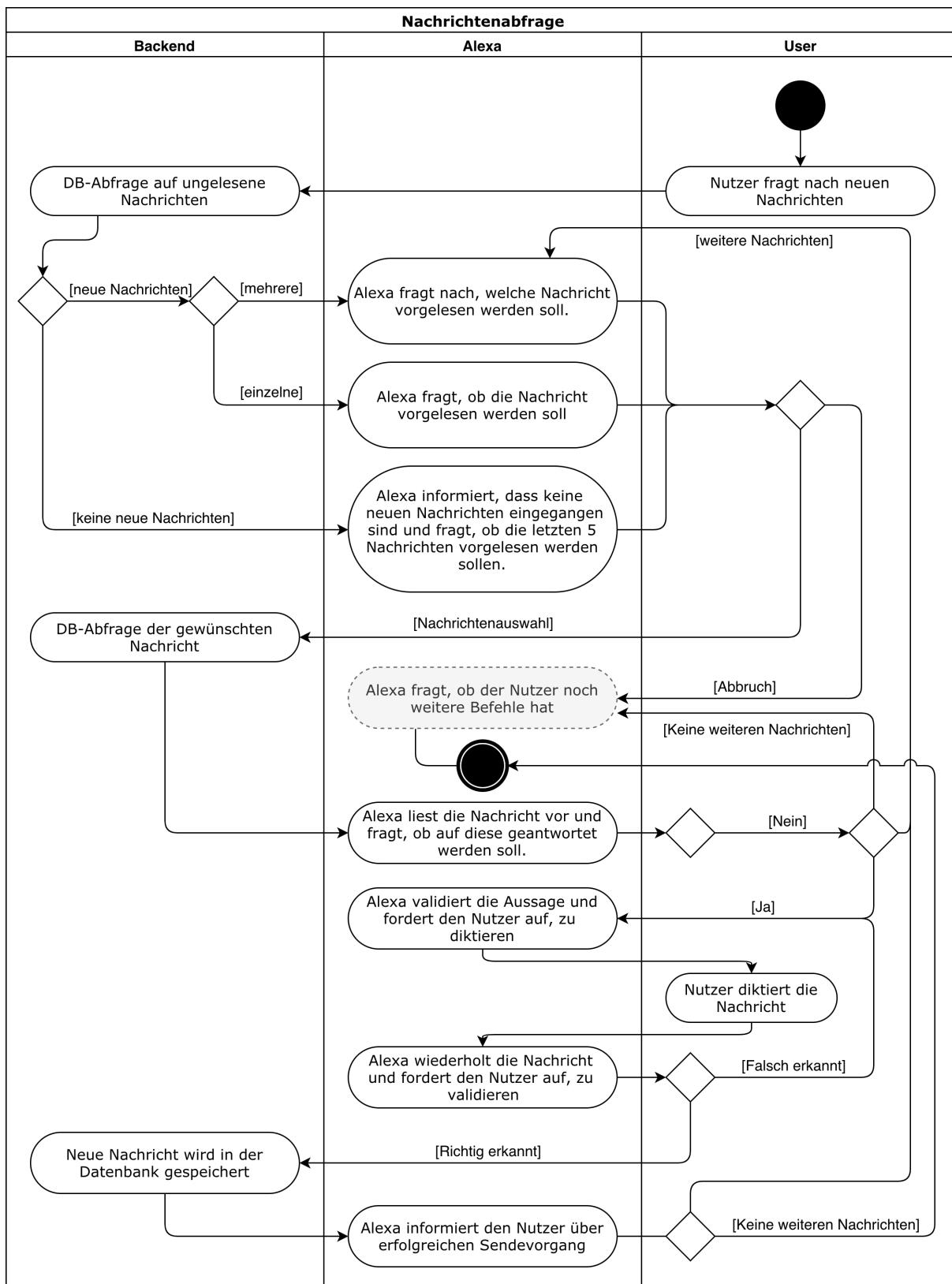


Abbildung 7: Interaktionsablauf Nachrichtenabfrage

verschiedene Formen haben. So ist es möglich, für ein und dieselbe Funktion der Nachrichtenabfrage verschiedene Sätze oder Trigger anzugeben (Beispielsweise: „Habe ich neue Nachrichten?“, „Nachrichten“, „Hat mit jemand geschrieben?“).

Nachdem der Nutzer auf diese Weise dem Backend mitgeteilt hat, dass eine Nachrichtenabfrage gewünscht ist, fragt dieses die Datenbank auf neue Nachrichten ab. Sollten neue Nachrichten gefunden werden, durchläuft Alexa so lange die „Lesen - Antworten“- Schleife, bis keine weiteren Nachrichten im Puffer existieren.

Generell wird eine Nachricht auf folgende Weise durchlaufen: Zuerst nennt Alexa den Autor und fragt den Nutzer, ob die Nachricht vorgelesen werden soll (analog zu dem derzeitigen Ansatz in Apps, in denen für Chats vorab angezeigt wird, ob neue Nachrichten vorhanden sind). Sollte der Nutzer nun Alexa den Befehl geben, die Nachricht vorzulesen, so wird Alexa das tun und im Anschluss den Nutzer fragen, ob er nun direkt darauf antworten möchte. Wenn der Nutzer nun antworten möchte, so sagt Alexa ihm, dass er nun die Nachricht diktieren kann. Sobald der Nutzer die Nachricht diktiert hat, bestätigt Alexa die Eingabe durch eine Wiederholung des Gesagten. Gibt der Nutzer zur Sendung der erkannten Nachricht seine seine Zustimmung, wird dem Backend mitgeteilt die Nachricht in der Datenbank abzuspeichern. Sollte die Nachricht nicht korrekt erfasst worden sein, bietet Alexa immer die Möglichkeit, wieder einen Schritt zurückzugehen und das Diktat zu wiederholen.

Sollte der Nutzer nicht auf die ihm vorgelesene Nachricht antworten wollen, so wird entweder die folgende Nachricht vorgelesen oder, sofern keine Nachrichten mehr im Puffer sind, nachgefragt, ob der Nutzer noch weitere Befehle an den Skill stellen möchte. Ist der Nutzer an der Stelle angelangt, an welcher weder weitere Nachrichten im Puffer sind noch eine Nachrichtenabfrage im Gange ist, so endet die Interaktion mit der Frage Alexas, ob es noch weitere Dinge zu tun gibt, bei der sie helfen kann.

4.4.2.2 Notfallsystem

Analog zur Nachrichtenabfrage funktioniert auch das Notfallsystem (siehe Abbildung 8). Es wird zuerst dieser Teilbaum des Skills durch einen einführenden Satz () zum Beispiel: „Notfall“ oder „SOS“) aktiviert. Daraufhin durchläuft der Nutzer wenige Dialoge, die einerseits eine Notfallnachricht aufnehmen und andererseits fragen, ob auch ein Notdienst benachrichtigt werden soll. Da die Funktion der Benachrichtigung eines Notdienstes erst für die Zukunft geplant ist, wird sie nicht vollständig in dieser Arbeit entwickelt.

Da eine solche Notfallnachricht äußert dringlich ist und der Nutzer sich vielleicht in Lebensgefahr befinden, sollte die Interaktion mit Alexa möglichst kurz sein.

Im dargestellten Modell muss der Nutzer im ersten Schritt das Notfallsystem starten, im zweiten Schritt wählt er den Empfänger und im dritten Schritt nimmt er eine Notfallbeschreibung auf.

Nachdem die Nachricht dann im vierten Schritt validiert wurde, geht die Notfallnachricht an die angegebenen Empfänger. Möchte der Nutzer auch noch einen Notdienst informieren, so folgen zwei beziehungsweise drei (bei geänderter Notfallbeschreibung) weitere Schritte.

Kern des Notfallsystems sind die Notfallkontakte. Diese werden im Backend hinterlegt und können in einem Notfall informiert werden. Anhand des mit Alexa übermittelten Notfalls können die Notfallkontakte (Angehörige und vertrauensvolle und bekannte Helfer) sofort die Lage und Dringlichkeit einschätzen. In Kombination mit dem Echo Connect (Amazon.com Inc., 2018c), das über Alexa die Möglichkeit bietet, Anrufe im Festnetz zu führen, wäre es dann für die entsprechenden Notfallkontakte oder auch Notdienste möglich, den Nutzer, der diesen Notfall gemeldet hat, anzurufen. Dieser wiederum kann den Anruf dann über Alexa führen.

Die zweite Möglichkeit ist, sollte es keine Notfallkontakte geben oder sollte es ein sehr kritischer Notfall sein, einen Notdienst zu informieren. Dafür wäre eine Kooperation mit einem solchen Notdienst nötig. Dieser bekommt dann ähnlich wie die Notfallkontakte eine Benachrichtigung über einen Notfall und hat dann die Möglichkeit, den Nutzer direkt per Telefon zu kontaktieren. Den Anruf kann der Pflegebedürftige dann via Alexa führen.

Nachdem die relevanten Fragen beantwortet und die Notfallmeldungen versendet wurden, terminiert der Skill.

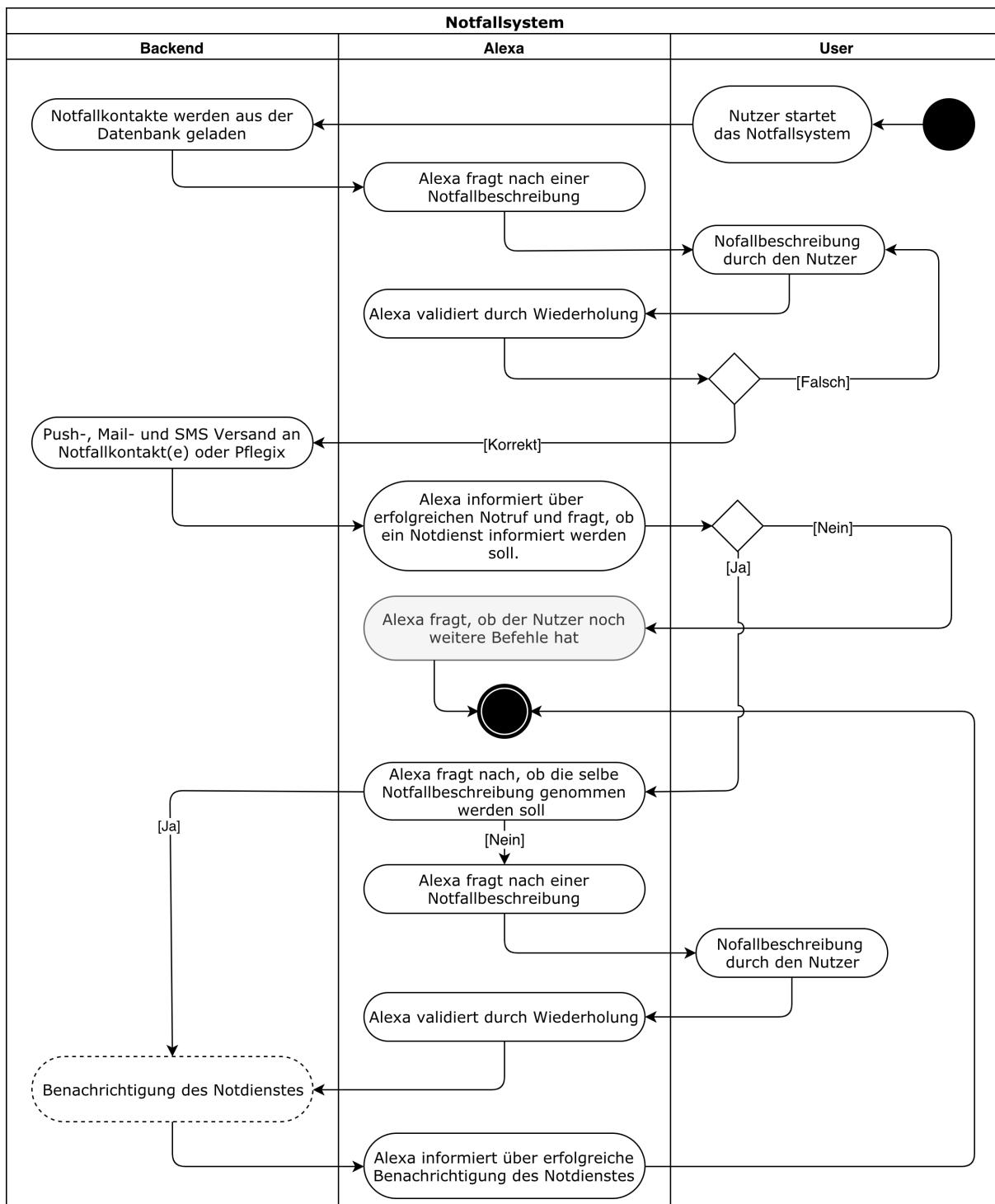


Abbildung 8: Interaktionsablauf Notfallsystem

5 Entwicklung

Im Folgenden wird die Programmierung des Alexa Skills behandelt.

5.1 Authentifizierung

Da der Pflegix Alexa-Skill nutzerspezifische Inhalte (Nachrichten, Personen) anbieten soll, ist eine Authentifizierung gegen den Pflegix-Server notwendig. Amazon bietet für die Verknüpfung des Amazon-Accounts mit dem Account des eigenen Servers zwei Möglichkeiten an, die beide auf dem OAuth2 Protokoll beruhen, welches nun näher erläutert wird.

5.1.1 OAuth2 Protokoll

Das OAuth Protokoll wurde in der ersten Version 2006 entwickelt. Im Jahr 2009 wurde es durch die Version zwei abgelöst, da die erste Version Sicherheitslücken im Design hatte. Im Folgenden wird von OAuth2 gesprochen. Es ist ein offener Standard, der auf vielen Websites und von vielen Unternehmen genutzt wird, um eine Anmeldung und Authentifizierung auf den Servern zu gewährleisten. Das OAuth-Protokoll bietet Websites und Anwendungen die Möglichkeit, sich untereinander zu verknüpfen. Ein klassisches Beispiel dafür sind die Anmeldemöglichkeiten von Google, Facebook und anderen großen Unternehmen („Sign in with...“), die auf vielen unabhängigen Seiten zu finden sind.

Die für die Entwicklung eines Skills zur Verfügung stehenden Möglichkeiten zur Verknüpfung der Accounts sind im speziellen die beiden Grant-Methoden „Implicit Code Grant“ und „Auth Code Grant“ des OAuth Protokolls.

Beim Implicit Code Grant (siehe Abbildung 9) wird der Nutzer auf eine Authentifizierungseite des Servers (hier des Pflegix Authentifizierungsservers) weitergeleitet. Dort gibt der Nutzer seine Login-Daten ein, die der Server verifiziert. Wenn die Authentifizierung korrekt war, gibt der Authentifizierungsserver einen Access-Token an den User Agent (hier die Alexa App) zurück. Mithilfe dieses Tokens kann sich der Alexa Service nun auf den Servern (in diesem Beispiel den Resource Server von Pflegix) authentifizieren, um so nutzerspezifische Daten anzufragen.

Beim Auth Code Grant (siehe Abbildung 10) bekommt der User Agent statt eines Access Tokens einen Autorisierungscode, welchen der Alexa Service immer wieder gegen einen neuen Access Token eintauschen kann.

Da im Regelfall die OAuth-Server so konfiguriert sind, dass ein Access Token nicht lange gültig ist (Größenordnung mehrere Tage), präferiert Amazon die Auth Code Methode, da der Autorisierungscode deutlich länger Gültigkeit besitzt und so eine erneute Autorisierung durch den Nutzer nicht nötig ist.

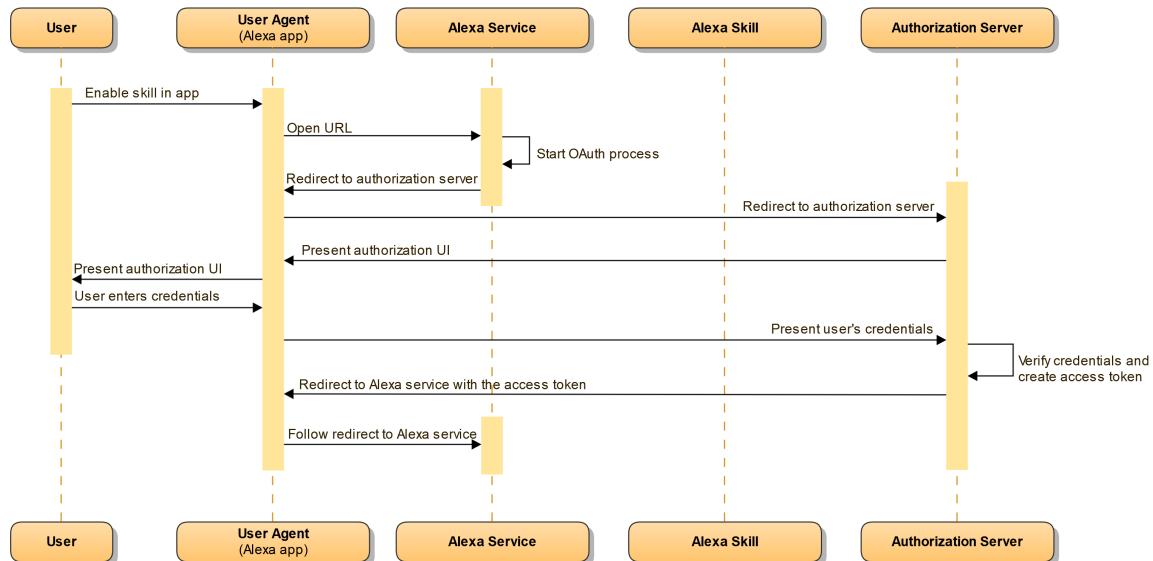


Abbildung 9: Implicit Code Grant (Amazon.com Inc., 2018b)

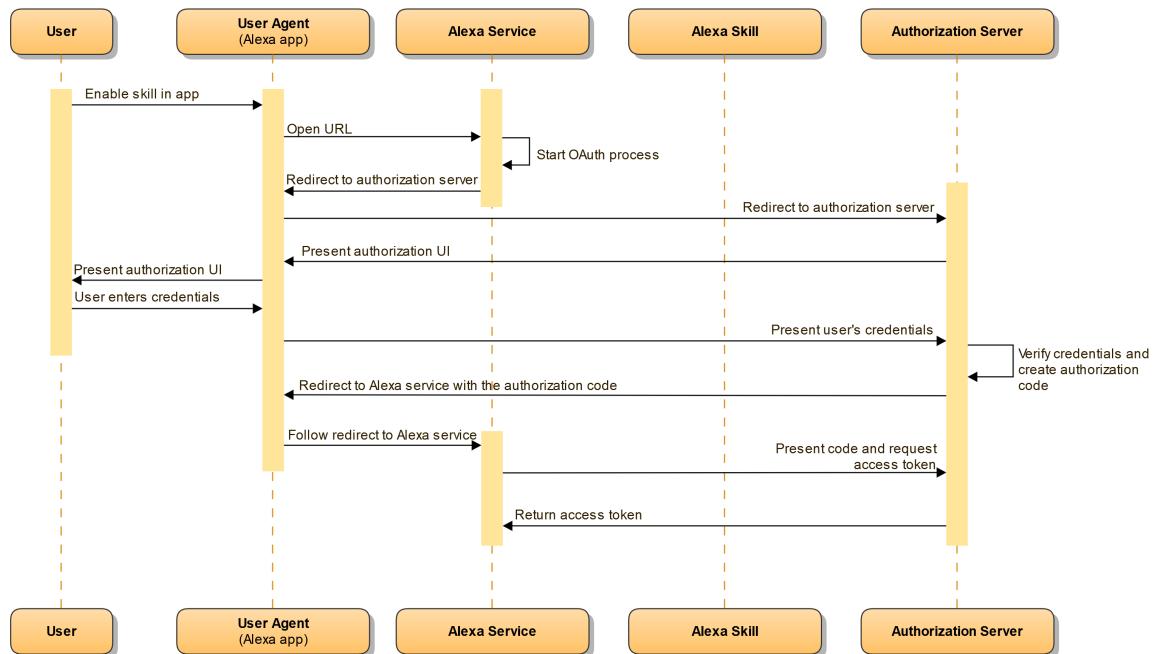


Abbildung 10: Auth Code Grant (Amazon.com Inc., 2018b)

5.1.2 Authentifizierungsserver

Um das Account-Linking zwischen Pflegix und Alexa zu ermöglichen, muss zuerst gewährleistet sein, dass Pflegix eine der genannten Grant-Methoden, optimalerweise die Auth Code Grant Methode, unterstützt. Dafür war es notwendig, einen Authentifizierungsserver, der auf dem OAuth2 Protokoll basiert, zu erstellen, denn zu diesem Zeitpunkt wurde der Nutzer mittels JSON-Webtokens authentifiziert.

Basierend auf dem Authentifizierungsserver von Hassanabad (2018) sollte der Server mit der vorhandenen Datenbank verknüpft werden. Zuerst wurden dafür alle irrelevanten Code-Fragmente aus dem Repository von Frank Hassanabad entfernt und eine Verknüpfung mit der Datenbank (MongoDB) geschaffen. Um nun sowohl die Access-Tokens als auch die Refresh-Tokens in der Datenbank speichern zu können, musste zuerst ein Datenbank-Modell für beide Tokens erstellt werden, welches die relevanten Daten der Tokens beinhalten soll (tokenID, userID, role, expirationDate, clientID, scope). Daraufhin war es dann möglich, die durch den Server erstellten Tokens in der Datenbank abzuspeichern, sodass das vorhandene Backend auf die Tokens zugreifen kann, um so Webanfragen autorisieren zu können.

Hier ist zu sehen, wie eine AccessToken mithilfe von Mongoose in der Datenbank abgespeichert wird:

```
// Create a new AccessToken from AccessToken-MongoDB-Model
// with given parameters
AccessToken.create({
  tokenID,
  expirationDate,
  userID,
  role,
  clientID,
  scope
}).then(res => {
  console.log('Accessstoken: ', res);
})
```

Um nun sicherzustellen, dass Personen ohne korrektes Passwort einen validen Access-Token erhalten, ist es zuerst nötig, in der Datenbank anzufragen, ob es einen Nutzer gibt. Daraufhin wird das zur Authentifizierung übergebene Passwort *gehasht* (beziehungsweise durch eine Funktion auf eine festgesetzte Größe gebracht) und mit dem nutzerspezifischen *Salts* (einer Zeichenkette zur Erhöhung der Entropie) verschlüsselt. Das Resultat dieses Vorgangs wird mit der DB abgeglichen. Erst wenn dieser Vorgang erfolgreich ist wird ein Token erzeugt,

zurückgesendet und abspeichert.

Da für die Entwicklung des Alexa Skills der Auth Code Grant notwendig ist, müssen für die einzelnen Clients, also übergeordneten Gruppen (beispielsweise Alexa, App oder normale Website), jeweils eine ClientID und eine ClientSecret erstellt werden, mit denen sich diese Clients gegenüber dem Authentifizierungsserver zu erkennen geben. Das soll eine zusätzliche Sicherheitsebene darstellen, da nur dann Login-Anfragen verarbeitet werden, wenn auch der Client identifiziert wurde. Zusätzlich bietet dies die Möglichkeit, das Verbinden von anderen Organisationen oder Websites zuzulassen aber auch einzuschränken.

In folgender Grafik ist dargestellt, wie der Stack für die Website vor und nach der Einbindung des Authentifizierungsservers aufgebaut war.

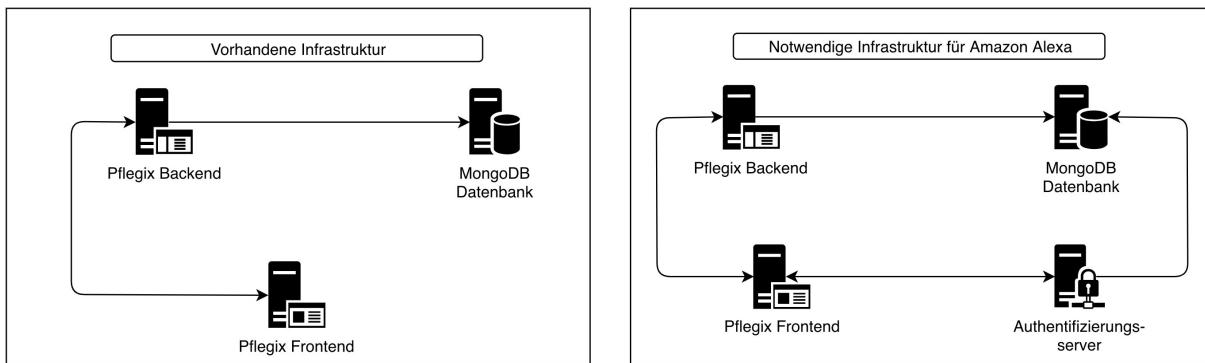


Abbildung 11: Infrastruktur Pflegix

Nachdem die Authentifizierungsmöglichkeit für den Alexa-Skill entwickelt war, wurde mit der eigentlichen Skill-Programmierung fortgefahrene.

5.2 Amazon Developer Console

In der Amazon Developer Console wird die Grundlage für die Interaktion des Nutzers mit dem Alexa-Skill geschaffen, indem durch sie die ersten Eintrittspunkte für den Nutzer erstellt werden.

5.2.1 Interaktionsmodell

Um das in Kapitel 4 beschriebene Interaktionsmodell nun auf den Skill zu übertragen, werden zunächst sogenannte Intents erstellt, die als Sprachbefehle dienen. Ein einzelner dieser Sprachbefehle kann dabei verschiedene Ausführungen haben.

Da eine Funktion eines Skills von verschiedenen Nutzern oft anders gestartet wird, ist es nötig möglichst viele verschiedene Wörter oder Sätze für einen Sprachbefehl zu definieren. Nur so

kann gewährleistet werden, dass ein Nutzer auch ohne große Vorkenntnisse und ohne Handbuch auskommt.

Um das Dialog-Management von Alexa nutzen zu können, sind zusätzlich zu den Intents noch Slots notwendig. Diese Slots sind als Platzhalter zu verstehen, die vom Nutzer im Dialog erfragt werden sollen. Folgendes Beispiel erklärt dabei den Ablauf eines Dialogs mit den entsprechenden Verknüpfungspunkten der Alexa Developer Console:

Ein Nutzer sagt „Alexa, starte Pflegix“ und startet damit den Skill. Daraufhin sagt der Nutzer „Schreibe eine Nachricht“ und startet damit den „new_message“-Intent (siehe Abbildung 12). Alexa erkennt nun, dass für das erfolgreiche Abschließen dieses Intents erfragt werden muss, wer der Empfänger ist und wie die Nachricht lautet. Daher arbeitet Alexa diese Fragen ab („An wen soll die Nachricht gehen?“, „Diktiere mir Deine Nachricht“). Sind alle fehlenden (und benötigten) Slots mit Inhalt gefüllt und bei Bedarf auch bestätigt worden, informiert Alexa das Backend über das Abschließen des Dialogs, sodass nun die Nachricht gesendet werden kann. Als Grundlage des Skills wurden analog zu den in Abbildung 6 dargestellten Methoden verschiedene Intents erzeugt, welche in Abbildung 12 zu sehen sind. Zudem ist in dieser Abbildung zu sehen, dass ein Intent aus vielen Sprachbefehlen besteht (Beispiel: request_messages mit sieben Befehlen). Um nun mehr Fälle abdecken zu können, was Nutzer potentiell sagen könnten um einen Intent anzusprechen, sind vor allem Realtests notwendig, durch die weitere Sprachbefehle zu einem Intent hinzukommen.

Folgendes Beispiel soll den Nutzen der Slots zur effizienten Dialogführung anhand des „request_messages“-Intents erläutern:

Der genannte Intent besteht zum einen Teil aus den *Utterances*, also den verschiedenen Sprachbefehlen, um diesen Intent anzusprechen. Zusätzlich dazu befinden sich die Slots *read*, *reply*, *message* und *selected* innerhalb des Intents. Die Aufgabe, die durch das Ansprechen dieser Funktion des Skills ausgeführt werden soll, ist die Nachrichtenabfrage. Sind nun mehrere neue Nachrichten für den Nutzer empfangen worden, so kann der Nutzer durch einen Dialog mit Alexa den booleschen Slot (entweder wahr oder falsch) *read* füllen, indem er bestätigt oder auch verneint, dass die Nachrichten vorgelesen werden sollen. Um nun eine von mehreren Nachrichten speziell abzurufen, wird der numerische Slot *selected* in einem kurzen Dialog gefüllt. Wurde die Nachricht nun erfolgreich vorgelesen, wird der Nutzer gefragt, ob er nun auf diese Nachricht antworten möchte, woraufhin dieser den booleschen Slot *reply* mit Inhalt füllt. Wird der Nutzer daraufhin von Alexa aufgefordert, eine Nachricht zu diktieren, wird der Freitext-Slot *message* ausgefüllt und der Dialog wäre damit abgeschlossen.

The screenshot shows the Alexa developer console interface. The top navigation bar includes links for 'Your Skills', 'Pflegix', 'Build' (which is selected), 'Test', 'Distribution', 'Certification', and 'Analytics'. On the far right, there are search, filter, and feedback forum icons. The main content area is titled 'CUSTOM' and 'Interaction Model'. On the left, a sidebar lists 'Invocation' and 'Intents (23)' with a '+ Add' button. The 'Intents (23)' section is expanded, showing categories like 'request_messages', 'new_message', 'new_request', 'emergency', and 'message', each with several sub-intents. To the right, a table titled 'Intents' displays a list of intents with columns for 'NAME', 'UTTERANCES', 'SLOTS', 'TYPE', and 'ACTIONS'. The table includes rows for 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', 'AMAZON.StopIntent', and various custom intents such as 'request_messages', 'new_message', 'new_request', 'emergency', 'help', 'wisdom', 'guide', and 'joke'. Each row has an 'Edit' and 'Delete' link under the 'ACTIONS' column.

Abbildung 12: Intents des Pflegix Skills

Analog zu diesem Beispiel wird der Nutzer mithilfe einfacher Dialoge durch komplexere Mechanismen des Skills geführt. Dadurch ist einerseits gewährleistet, dass keine hohen Anforderungen an das Gedächtnis des Nutzers gestellt werden und andererseits wird durch gezieltes Nachfragen und Einholen von Validierung durch den Nutzer die Fehlerhäufigkeit drastisch reduziert. Zusätzlich zu den Slots im oben aufgeführten Beispiel, die nicht notwendig für das Abschließen des Intents sind, gibt es die Möglichkeit, Slots als notwendig anzulegen. Das bietet dem Entwickler die Möglichkeit dem Skill über das Backend nicht nur mitzuteilen, welchen speziellen Slot Alexa nun erfragen soll sondern auch durch alle notwendigen Slots zu iterieren und diese zu erfragen. Das senkt den Codeumfang des Backends erheblich.

5.2.2 Account-Linking

Damit der Skill mit den Pflegix Servern kommunizieren und Anfragen für den Nutzer an diesen senden kann, muss dem Skill mitgeteilt werden, welcher Endpunkt angesprochen werden soll. Da für eine schnelle Entwicklung mit schnellen Testphasen ein lokales Entwicklungsumfeld gewählt wurde, wurde das Tool Ngrok (Shreve, 2018) genutzt, um die lokale Maschine über das

Internet erreichbar zu machen. Die entsprechende URL wurde dann in der Amazon Developer Console hinterlegt.

Zusätzlich zum einfachen Endpunkt, der die Anfragen verarbeitet, ist noch ein Account-Linking (siehe Abbildung 13) notwendig. Dieses bewirkt, dass sobald der Amazon- und der Pflegix-Account einmal verknüpft wurden, jede Alexa-Instanz des Nutzers (Smartphone, Echo) Zugriff auf die Pflegix-Daten hat.

Dazu wird eine Authorisierungs URI genutzt, die den Nutzer beim Initialisieren des Skills auf eine Login-Seite des Authentifizierungsservers leitet. Sobald die Anmeldung dann vollzogen und Alexa durch den Nutzer Zugriff auf die Pflegix-Server gewährt wurde, wird an eine der Weiterleitungs-URLs der in diesem Vorgang produzierte Autorisierungscode weitergegeben. Mithilfe dieses Codes ist Alexa dann in der Lage den Access Token abzurufen (über die angegebene Access Token URI) und, sollte ein Access Token nicht mehr gültig sein, auch jederzeit einen neuen abzurufen. Der Vorgang des Erneuerns des Access Tokens geschieht durch den Alexa Service automatisch und benötigt keine weitere Konfiguration. Dadurch, dass Alexa mithilfe des Access Tokens die Möglichkeit hat, den Server Endpunkt für die Anfragen anzusprechen und sich gegen diesen als Nutzer authentifizieren kann, wird im Folgenden die Programmierung des Intent-Handlings behandelt.

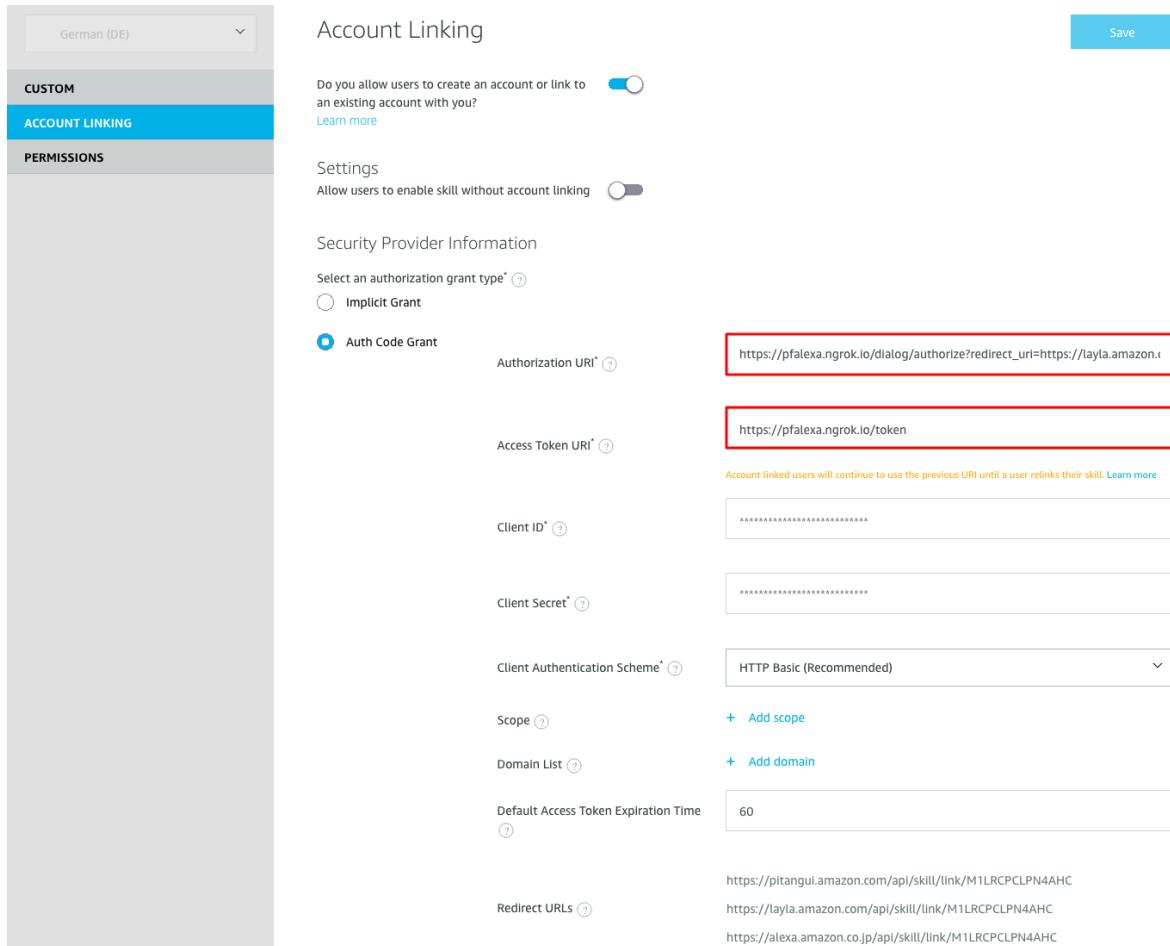


Abbildung 13: Account Linking

5.3 Intent-Handling

Das Intent-Handling macht den Hauptteil dieser Arbeit aus. Erst durch das Zusammenfügen des Intent-Handlings mit den im vorherigen Unterkapitel behandelten Aktionen in der Amazon Developer Console entsteht ein nutzbarer Skill.

5.3.1 Vorbereitungen

Die Möglichkeiten für die Alexa-Plattform ein Backend bereitzustellen sind vielfältig. Zusätzlich zu der von Amazon angebotenen AWS Lambda-Infrastruktur, bietet Amazon verschiedene SDKs an, mit denen Entwickler arbeiten können. Diese sind alle in unterschiedlichen Programmiersprachen entwickelt und bieten daher eine gute Möglichkeit, Alexa in vorhandene Strukturen einzufügen.

Da die Infrastruktur von Pflegix in Node.js entstanden ist, bot sich für diese Arbeit das

Node.js „Amazon Skills Kit“ an. Simultan zu dem bereits in Kapitel 5.1.2 beschriebenen Authentifizierungsserver sollte der Alexa-Server entstehen.

5.3.2 Kategorisierung der Intents

Da sich recht früh in der Entwicklung herausstellte, dass die einzelnen Intents zu umfangreich werden würden und durch die Menge zu schreibender Intents eine mangelnde Übersicht drohte, wurden die einzelnen Intents in eigene Module ausgelagert. Dadurch sollte einerseits gewährleistet sein, dass der Code übersichtlich und lesbar bleibt und andererseits, dass die Wartbarkeit des Codes deutlich vereinfacht wird.

Zusätzlich zu diesem Schritt wurden noch globale Variablen angelegt. Diese sollen für die Dauer eines Gesprächs diverse Informationen bereithalten (Nutzername, wiederkehrende Sprachausgaben, abgerufene Nachrichten und andere häufig genutzte Variablen). Das ist nötig, um einen personalisierten Skill anbieten zu können, ohne bei jeder Anfrage des Nutzers mehrere Datenbankabfragen durchführen zu müssen, um den Namen, die Id oder auch die Nachrichten neu abzufragen. Die globalen Variablen dienen also der effizienteren Abfrage von Nutzerinformationen.

5.3.3 Umsetzung der Intents

Im Folgenden wird nun für alle Intents kurz geschildert, wie diese umgesetzt wurden. Speziell wird auf die Intents der Nachrichtenabfrage, der Anfrage eines Helpers und der Notfallfunktion eingegangen.

5.3.3.1 Nachrichten-Intent

Zuerst wurde die Nachrichtenabfrage implementiert (Intent: „request_messages“). Diese sollten den Zweck erfüllen, dass ein Nutzer neue Nachrichten oder - sofern es keine neuen Nachrichten gab - die letzten fünf Nachrichten abrufen kann.

Die einzelnen Intents müssen entsprechend der Dialogführung einer gewissen Reihenfolge unterstehen (siehe Reihenfolge Abbildung 14). In dem Fall der Nachrichtenabfrage heißt dies, dass der Nutzer - sofern es neue Nachrichten gab - eine Möglichkeit der Auswahl bekommt, welche dieser Nachrichten er abrufen möchte („selected“). Gibt es lediglich alte Nachrichten, wird der Nutzer gefragt, ob die letzten fünf Nachrichten vorgelesen werden sollen („read“). Sollte der Nutzer eine Nachricht ausgewählt haben, wird diese vorgelesen und der Nutzer wird gefragt, ob er auf die Nachricht antworten möchte („reply“). Bestätigt er diese Abfrage, bittet Alexa den Nutzer, eine Nachricht zu diktieren („message“).

Intent Slots (4) [?](#)

ORDER ?	NAME ?	SLOT TYPE ?	ACTIONS
1	selected	AMAZON.NUMBER	Edit Dialog Delete
2	read	BOOLEAN	Edit Dialog Delete
3	reply	BOOLEAN	Edit Dialog Delete
4	message	AMAZON.SearchQuery	Edit Dialog Delete

Abbildung 14: Slots des Intents *request_messages*

Damit Alexa Informationen darüber bekommt, ob und wie viele neue Nachrichten für den Nutzer vorhanden sind, wird zuerst eine Datenbankabfrage für den entsprechenden Nutzer ausgeführt, die das Attribut *lastRead* einer Konversation prüft. Daraufhin werden die zu der entsprechenden Konversation gehörenden neuen Nachrichten abgefragt. Im Folgenden sind beide Datenbankabfragen aufgeführt. Die Verknüpfung für die oben aufgeführte Logik geschieht an anderer Stelle im Code.

```
// Query to get all conversations of a user.
exports.find = userId =>
  Promise.resolve(Conversation.find({ participants: userId })
    .sort('-createdAt')
    .populate({
      path: 'participants',
      model: User,
      select: 'firstname lastname role slug'
    }).exec()
  )
```

```
// Query to get all unread messages for a specified conversation.
exports.findNew = (conversation, lastRead) =>
  Promise.resolve(Message.find({
    conversation: conversation._id,
    messagetype: "text",
    updatedAt: { $gte: lastRead }
  })
```

```
.sort("-createdAt")
.populate({
    path: "author",
    select: "firstname lastname role slug"
})
.exec()
);
```

Wenn nun beispielsweise keine neue Nachricht gefunden wurde, fragt Alexa den Nutzer ob dieser die letzten fünf Nachrichten hören möchte. Daraufhin muss das Backend zuerst den entsprechenden Slot *read* überprüfen um abzufragen, ob der Nutzer dieser Frage zustimmte. Dann wird der aktuelle *message_intent* innerhalb des Codes geändert, um die folgenden Dialogabfragen des Nutzers richtig zuordnen zu können.

Der *message_intent* dient dabei als Positionsvariable innerhalb des *request_messages*-Intents. Diese Variable wird vor allem dafür genutzt, die vier Slots mehrfach nutzen zu können und um den Verlauf der Abfragen zu bestimmen.

Als Beispiel würde bei neuen Nachrichten die Variable auf 'ask_to_read' gesetzt, während sie bei keinen neuen Nachrichten und der daraus folgenden Frage, ob die letzten fünf alten Nachrichten vorgelesen werden sollen, auf 'ask_to_read_old' gesetzt wird. Dies ist vor allem deswegen nötig, damit die Slots für beide Fälle genutzt werden können sowie um beim Durchlaufen von Nachrichten den korrekten Array mit Nachrichten zu wählen.

Daraufhin wird die angefragte Konversation mit den neuen beziehungsweise der letzten Nachricht aus der Datenbank abgerufen und in ein lesbaren Format für die Ausgabe mit der Alexa-Plattform gebracht.

Zuletzt wird der Nutzer noch gefragt, ob er auf diese Nachricht antworten will. Um Alexa nun mitzuteilen, dass der *reply*-Slot abgefragt wird, ruft man

```
this.emit(':elicitSlot', 'reply', speechOutput, repromptOutput)
```

auf. Diese gibt Alexa den Befehl, mithilfe des „elicitSlot-Directives“ den Slot *reply* zu erfragen. Hierfür gibt man einen String mit (*speechOutput*), den Alexa daraufhin dem Nutzer sagt und einen weiteren String (*repromptOutput*) für den Fall, dass der Nutzer entweder zu spät, gar nicht oder falsch auf die Frage antwortete.

Antwortet der Nutzer nun auf diesen Befehl, wird eine neue Anfrage an das Backend geschickt, die den ausgefüllten Slot enthält, woraufhin das Backend entsprechend der in Abbildung 7 gezeigten Abläufe den Dialog fortführt.

Ein typischer Dialogablauf für das Anfragen der Nachrichten ist in Abbildung 15 zu sehen.

Während der Entwicklung wurde schnell absehbar, dass einzelne Intents nicht durch einen einzigen Satz des Nutzers abgedeckt werden konnten, ohne dass oft Fehler bei der Erkennung auftraten. Daher wurde als grundlegendes Konzept der Benutzerführung das Aufsplitten der einzelnen Intents gewählt, um durch Bestätigungen und einfache Fragen die Möglichkeiten für das Auftreten von Fehlern zu reduzieren. Diese Art der Benutzerführung wurde in jedem Intent weitestgehend berücksichtigt, sodass viele komplexere Befehle, wie das Erstellen einer Nachricht an eine bestimmte Person, in mehrere Unterbefehle gegliedert wurde.

Diese Art der Benutzerführung kommt zugleich dem Nutzer zugute, da dieser sich einerseits nicht merken muss, welche Informationen für einen Befehl benötigt werden und andererseits für die wichtigsten dieser Subinformationen eine bestätigende Abfrage durch Alexa enthält. Dies ermöglicht, nur einen Teil des Befehls zu korrigieren, anstatt den kompletten, zusammengesetzten Befehl neu diktieren zu müssen.

Ein etwas anderer Ansatz für die Dialoggestaltung wurde bei der Umsetzung des Intents für die Anfrage an Helfer (*new_request*) verfolgt.

5.3.3.2 Anfrage-Intent

Zuerst einmal muss erwähnt werden, dass die Funktion der konkreten Terminanfragen an Helfer derzeit noch nicht innerhalb des Pflegix-Systems integriert wurde, allerdings als geplantes Feature für dieses Jahr ansteht. Daher wurde dieser Intent als eine Art spezieller Nachrichtenversand gebaut. Die Terminanfragen, die durch den Intent *new_request* erstellt werden, werden daher derzeit in eine lesbare Nachricht umgewandelt, die dann an den entsprechenden Helfer gesandt wird. Dabei wurde diese Funktion bewusst nicht in den *request_messages*- oder *new_message*-Intent mit integriert, obwohl die derzeitige Funktion recht ähnlich zu einer einfachen Nachricht ist. Alleine schon durch die zukünftige Umsetzung einer solchen Funktion ist ein entsprechender

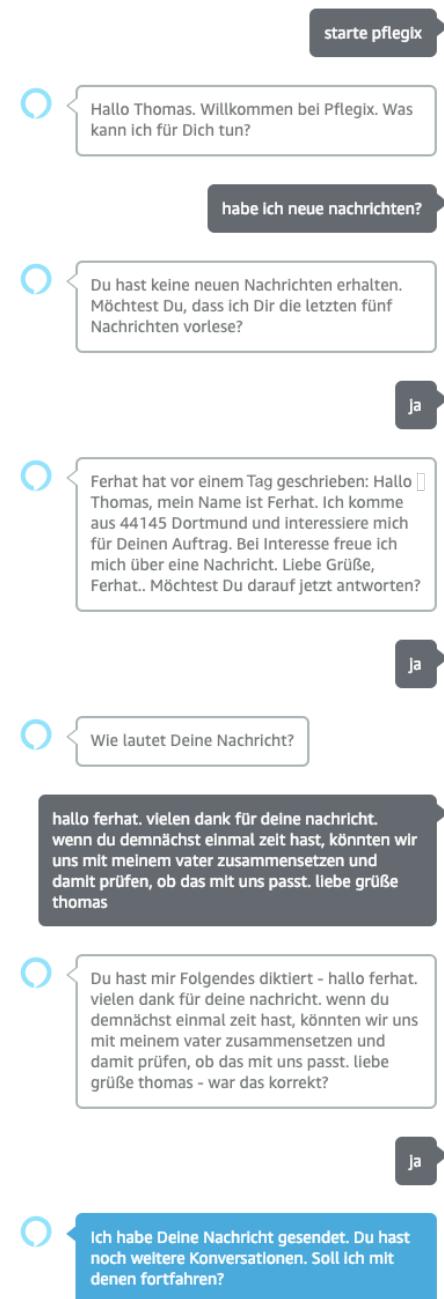


Abbildung 15: Beispieldialog der Nachrichtenabfrage

Intent sinnvoll, da die Abfragen für eine Anfrage im einfachen Nachrichtenversand störend wirken.

Anders als beim bereits vorgestellten *request_messages*-Intent, werden beim Anfrage-Intent alle Slots benötigt, um den Intent erfolgreich abzuschließen. Im Code wird das vor allem dadurch sichtbar, dass statt der vielen vorher benötigten if-Bedingungen und Abfragen der Slots nur ein *:delegate*-Directive genutzt wird, bis alle Slots valide gefüllt sind:

```
| this.emit(':delegate')
```

Die Befehle, die für diesen Intent genutzt werden können, sind dabei in der Lage einen, mehrere oder gar keinen der Slots direkt zu füllen. Da die einzelnen Slots verschiedenen Funktionen dienen, haben sie auch weitestgehend verschiedene Typen. Dadurch ist es möglich, in einem Befehl mehrere Slots zuverlässig zuzuordnen. Beispielsweise können die Slots *time* vom Typ „AMAZON.Date“ und *user* vom Typ „AMAZON.DE_FIRST_NAME“ durch ihre unterschiedlichen Typdefinitionen korrekt zugeordnet werden.

Ein Slot des *new_request*-Intents ist jedoch nicht für die korrekte Verarbeitung des Intents notwendig. Der *selected*-Slot wird erst dann benötigt, wenn der Nutzer eine Person genannt hat, deren Name nicht in der Liste der verfügbaren Kontakte gefunden wurde. Wurde ein Name korrekt angesagt, erkannt und wurde dieser in den Kontakten gefunden, wird dieser Slot nicht abgefragt.

Dennoch ist es in diesem Intent möglich das *:delegate*-Directive zu nutzen, um alle anderen Slots zu erfragen. Der Unterschied zwischen den benötigten und nicht benötigten Slots findet sich in der Amazon Developer Console. Dort kann man genau diesen Punkt für jeden Slot einstellen. Das genannte Directive wird dabei nur die Slots erfragen, die auch als benötigt gekennzeichnet wurden.

Konkret wurden für den *new_request*-Intent fünf Slots verwendet. Der Slot *user* ist das Freitext-Feld für den Vornamen des gewünschten Partners. Der Slot *date* erfasst das Datum, während der Slot *time* die entsprechende Uhrzeit enthält. Der Slot *tasks* vom Typ „AMAZON.SearchQuery“ dient als Freitextfeld für die Ansage von Aufgaben, die für den genannten Zeitpunkt anfallen. Die genannten Slots sind allesamt essentiell für die Erfüllung des Intents. Der letzte, bereits erwähnte, *selected*-Slot ist vom Typ „AMAZON.Number“ und dient der einfachen Auswahl bei Nennung einer Liste.

Für diesen Intent wurde die Entscheidung getroffen, dass einzelne Slots keine Bestätigungsabfrage erhalten sollen. Bei den vier genannten notwendigen Slots würde jeweils eine Frage für die Bestätigung den kompletten Dialog in die Länge ziehen. Da in vielen Durchläufen gerade die Zeit und das Datum zuverlässig erkannt wurden und das Freitextfeld für die Aufgaben bereits am Ende des Dialogs der Intents steht, wurden für diese keine Abfragen eingerichtet. Für den

Vornamen des Partners wurde keine zusätzliche Abfrage eingerichtet. Bei unterschiedlichen Schreibweisen eines Vornamens kann teilweise kein korrekter Eintrag in den Kontakten gefunden werden - in diesen Fällen wird der erste Buchstabe des Vornamens genommen und mit den Kontakten abgeglichen. Diese Kontakte werden als Auswahl an den Nutzer zurückgegeben. Die Auswahl, die der Nutzer daraufhin trifft, wird in dem *selected-Slot* gespeichert und vom Backend weiter verarbeitet.

Nachdem alle Slots gefüllt wurden, wird eine Bestätigung durch den Nutzer erfragt, indem alle Slots noch einmal wiederholt werden, sodass ein Nutzer noch die Möglichkeit der Korrektur hat.

5.3.3.3 Notfall-Intent

Zuletzt wird nun auf die Notfallfunktion detaillierter eingegangen. Wie auch die Funktion der Versendung von Anfragen ist auch die Notfallfunktion noch kein Bestandteil der Pflegix-Plattform. Durch angestrebte Partnerschaften mit lokalen Pflegeunternehmen und die dadurch zur Verfügung stehende Möglichkeit des ambulanten Notdienstes, wurde jedoch diese Funktion als Prototyp eingebaut.

Der Zweck dieses Intents besteht darin, dass die Pflegebedürftigen über Alexa ihre Notfallkontakte und bei entsprechenden Partnerschaften auch Notdienste kontaktieren können. Notfallkontakte können dabei Familienmitglieder oder auch spezielle Helfer sein. Für diese Arbeit wurde daher zuerst ein Model für einen Notfallkontakt erstellt. Dieser soll dabei, sofern der Notfallkontakt auf Pflegix registriert ist, eine Verknüpfung zu diesem Helfer enthalten. Falls der Notfallkontakt außerhalb von Pflegix besteht wird dieser mit Vorname, Nachname, E-Mail Adresse und Telefonnummer angegeben werden.

Prototypisch wurde für die auf Pflegix registrierten Helfer die Notfallfunktion eingebaut, sodass diese per Push-Benachrichtigungen über den Notfall informiert werden.

Zunächst mussten für den Testaccount Notfallkontakte angelegt werden. Diese wurden aus der Datenbank genommen, jedoch mit geänderter Adresse für die Push-Notifications. Durch den Dienst *OneSignal* wurden die Push-Meldungen erstellt und an die angegebenen Adressen gesendet. OneSignal ist ein Dienst, über den verschiedene Endgeräte mit Push- oder E-Mail-Meldungen angesprochen werden können.

Auch die Abfrage, ob ein Notdienst informiert werden soll, wurde im Rahmen dieser Arbeit umgesetzt, jedoch ohne eine entsprechende Wirkung. Somit steht diese Funktion als erweiterbarer *Stub* für die weitere Entwicklung bereit.

5.3.3.4 Andere Intents

Zusätzlich zu den bereits vorgestellten Intents wurden noch weitere Intents erstellt. Dazu

gehören der *new_message*-Intent, der es dem Nutzer ermöglicht, direkt eine Nachricht an einen bestimmten Nutzer zu versenden. Der *wisdom*-Intent soll, genauso wie die beiden Intents *joke* und *joke_of_the_day* eine kleine Spielerei für den Nutzer sein, die keinen weiteren Nutzen hat, außer Unterhaltung zu beiten.

Außerdem wurde ein Guide als Intent umgesetzt, der den Nutzer durch den Skill führen kann, wenn der Nutzer dies wünscht. Dabei lernt der Nutzer jede Funktion einmal kennen, indem er sie nutzt.

Auch wenn die oben genannten Intents den Großteil der Arbeit ausmachen, ist es nötig, die von Amazon vordefinierten Intents umzusetzen. Vor allem der *AMAZON.StopIntent* und der *AMAZON.HelpIntent* sowie der *AMAZON.CancelIntent* sind hier anzuführen, da diese eine zentrale Rolle für die Navigation innerhalb des Skills spielen.

5.4 Roll-Out

Nachdem nun vorgestellt wurde, wie die Intents umgesetzt wurden und der Skill veröffentlicht werden kann, ist der Roll-Out nun Bestandteil dieses Unterkapitels.

5.4.1 Voraussetzungen

Um einen Skill für die Alexa-Plattform veröffentlichen zu können, müssen bestimmte Voraussetzungen erfüllt werden.

Amazon verlangt für die ordnungsgemäße Veröffentlichung in erster Linie weitere Informationen zu dem Skill, um diesen genauer einordnen zu können. Dazu gehören zusätzlich zum Skill-Namen auch eine Beschreibung des Skills und einige Beispielsätze, die den Nutzen des Skills widerspiegeln. Außerdem müssen einige Stichwörter angegeben werden, über die der Skill zu finden ist.

Da es sich um eine Software handelt die veröffentlicht wird, ist auch bei einem Alexa-Skill das Angeben einer URL zu den Datenschutz-Richtlinien sowie den Nutzungsbedingungen Pflicht. Neben allgemeinen Angaben, wie beispielsweise der Nutzung von In-Skill-Käufen oder an welche Nutzergruppe sich der Skill richtet, erfragt Amazon auch, ob der Skill innerhalb der Länder, in denen Amazon agiert, exportiert werden darf oder ob der Skill Werbung enthält.

Zuletzt gibt man an, in welchen Ländern der Skill verfügbar sein soll und ob der Skill öffentlich oder firmenintern genutzt werden soll.

Sind alle diese Angaben gemacht, muss der Skill zuerst einige automatisierte Tests durchlaufen, die erst einmal die Eingaben für die Veröffentlichung validieren und danach einige Funktionstests durchlaufen, die in erster Linie sicherstellen sollen, dass der Skill nur auf Anfragen von Amazon reagiert.

5.4.2 Maßnahmen

Um den Skill im Marktplatz von Amazon anbieten zu können, mussten zunächst alle Angaben gemacht werden. Nachdem diese auch validiert wurden, zeigte der funktionale Test, dass der Server zum damaligen Zeitpunkt noch keine Validierung der Alexa-Anfragen, die an das Backend gehen, eingebaut hatte. Dieses Problem konnte durch eine Middleware (Shah et al., 2017) gelöst werden, die die einzelnen Anfragen vor der Verarbeitung validiert und nur im erfolgreichen Fall die Anfrage verarbeitet. Dadurch wird sichergestellt, dass nur von Amazon signierte Anfragen der Alexa-Plattform verarbeitet werden und unsignierte oder falsch signierte Anfragen nicht bearbeitet werden.

6 Diskussion

Dieses Kapitel beschäftigt sich mit den Limitierungen des praktischen Teils dieser Arbeit. Dabei wird vor allem auf die Schwierigkeiten dieser Arbeit eingegangen und herausgestellt, aus welchen Gründen manche vorgestellten Features umgesetzt wurden und andere wiederum nicht.

6.1 Ungeplante Umsetzungen innerhalb der Arbeit

Einige Voraussetzungen für die Entwicklung eines eigenen Alexa Skills waren zu Beginn dieser Arbeit nicht erfüllt. Der Entwicklungsschritt, der abseits der Intent-Entwicklung am meisten Zeit in Anspruch genommen hat, war die Erstellung eines Authentifizierungsservers nach dem OAuth2-Standard, der mit den vorhandenen Systemen zusammenarbeiten kann. Dadurch, dass hierfür ein komplett neuer Server mit Datenbankanbindung an die vorhandene Datenbank erstellt werden musste, noch bevor eine sinnvolle Weiterarbeit am eigentlichen Skill möglich war, reduzierte sich die Zeit zur Entwicklung des Skills. Leider konnten durch diesen unvorhergesehenen Schritt einige sinnvolle oder auch wünschenswerte Features nicht eingebaut werden, wie im nächsten Teil erläutert.

Es gibt zwar für das Alexa Skills Kit von Amazon eine Dokumentation der Funktionen, die genaue Anwendungen einzelner Funktionen und Methoden wird jedoch erst durch die Beispielprojekte von Amazon ersichtlich. Auch diese Tatsache hat die Entwicklung weiter verzögert, da beispielsweise die Information zu den Directives, mit denen die einzelnen Slots gefüllt und abgefragt werden konnten, erst später gefunden wurde und bis dato ohne dieses hilfreiche Feature gearbeitet werden musste.

6.2 Grenzen dieser Arbeit

An einigen Stellen dieser Arbeit wurde bereits deutlich, dass geplante Teile nicht oder nur teilweise umgesetzt wurden. Ein Beispiel dafür ist das Senden von Anfragen.

Richtige Anfragen, beziehungsweise eine Planung der Helfertätigkeiten, sollen in naher Zukunft in der Plattform Pflegix umgesetzt werden. Da aber eine spätere Umsetzung in das System geplant ist, wurde dieser Intent vorerst als Stub entwickelt. Dieser hat nun zwar alle nötigen Informationen um eine Anfrage zu versenden, da allerdings diese Funktion im System noch nicht integriert ist, wird vorerst lediglich eine Nachricht mit den Daten versandt.

Wie bereits erwähnt, wurden auch beim Notfall-Intent bereits für die Zukunft eingeplante Features als Templates eingebaut. Es wird zwar eine Abfrage nach einem Notdienst durchgeführt, jedoch hat dies noch keine Auswirkungen - es wird also noch kein Notdienst kontaktiert. Für die Zukunft ist die Funktion des Sendens von Notfällen jedoch ein interessantes Thema, bietet

Alexa doch im Gegensatz zum herkömmlichen Telefon die Möglichkeit, Kontakte per Sprache zu kontaktieren.

Ungeachtet der Tatsache, dass im Rahmen dieser Arbeit eine Notfallfunktion entwickelt wurde, sollte jedoch in Frage gestellt werden, ob und wie sinnvoll eine solche Funktion in der Praxis ist. Auf der einen Seite bietet sie, beispielsweise im Falle eines Sturzes die Möglichkeit, Kontakte über den Notfall zu informieren, indem einfache Sprache genutzt werden kann. Damit wäre diese Funktion gerade in Fällen, in denen die Mobilität der pflegebedürftigen Person eingeschränkt ist, eine gute und vor allem realisierbare Lösung. Andererseits ist die Handhabung wenig praktikabel. Um einen Notfall melden zu können muss der Nutzer zuerst den Skill starten, bevor er dann den entsprechenden Intent aufrufen kann. Dieser wiederum benötigt weitere Informationen, die der Nutzer angeben muss.

Diese Umsetzung eines Notfallsystems ist recht kompliziert und daher auch nicht für ambulante Notfälle geeignet. Gerade wenn weitere Informationen abgefragt werden müssen, könnten der Standort und die Lautstärke eines Alexa-Geräts das erfolgreiche Erstellen eines solchen Notfalls verhindern. Auch der Aufruf ist umständlich. Statt „Alexa, Notfall“ oder „Alexa, ich brauche Hilfe“ ist mindestens folgender Satz notwendig: „Alexa, sage Pflegix starte Notfall“. Gerade in Schockmomenten oder auch generell bei der ersten Nutzung dieses Intents dürfte es dem Nutzer schwer fallen, den entsprechenden Satz für den Aufruf des Intents aus dem Gedächtnis abzurufen.

Daher ist zwar die Idee hinter dem Notfall-Intent gut, jedoch wird der Intent vermutlich nicht in lebensbedrohlichen Notfällen aufgerufen. Stattdessen richtet sich diese Funktion mehr an Situationen, in denen eher eine gesellschaftliche Hilfe benötigt wird.

Zusätzlich zu den limitierenden Faktoren der Plattform Pflegix gab es zudem weitere limitierende Faktoren, die zuerst geplante Eigenschaften des Skills einschränkten.

Da der Skill unter anderem die Funktion bietet, Nachrichten zu senden und abzurufen, wäre es sinnvoll, auch Notifications für den Skill bereitzustellen. Da sich die Integration von diesen allerdings zum Zeitpunkt der Entwicklung noch in einer geschlossenen Beta-Phase befand, gab es leider keine Möglichkeit, Notifications in den Skill einzubauen.

6.3 Ausblick

In erster Linie wurde bereits in Kapitel 6.2 erwähnt, welche Features im Rahmen dieser Arbeit noch nicht entwickelt werden konnten, jedoch in Zukunft in den Alexa-Skill eingebaut werden sollen. Zusätzlich dazu werden nun einige weitere Ausblicke gegeben, in welche Richtung die Entwicklung des Skills fortgeführt werden kann.

Für die Zukunft wird die Plattform Pflegix ein Buchungssystem einführen. Über dieses sollen

Helper zu freien Terminen gebucht werden können. Wünschenswert wäre dann, dass die entsprechenden Termine in dem mit Amazon verknüpften Konto erscheinen. So hat der Nutzer dann auch über Alexa die Möglichkeit, auf Termine der Pflegix-Plattform zuzugreifen.

Des Weiteren besteht die Möglichkeit, auch die Zeiterfassung über Alexa laufen zu lassen. In diesem Fall sagt der Helper dem Skill, wann die Zeiterfassung gestartet oder beendet werden soll. Da die Arbeitszeit jedoch nach derzeitigem Stand nur vom Helper erfasst werden kann, der Skill jedoch die Pflegebedürftigen als Zielgruppe hat, wurde diese Idee für die erste Version des Skills verworfen.

Eine mögliche Weiterentwicklung wäre die Einbindung des Displays in den Skill, um auch die Kontaktsuche zu ermöglichen.

Neben den Pflegix-spezifischen Möglichkeiten sollen in Zukunft einige, derzeit noch in der Entwicklungsphase befindliche Features der Alexa-Plattform, genutzt werden.

Gerade für Geräte mit einem Display arbeitet Amazon derzeit an einer Verbesserung der Darstellung. Die sogenannte *Amazon Presentation Language* soll Entwicklern die Möglichkeit geben, auf einfache Weise UI-Elemente für den Skill anzugeben und zu erstellen.

Auch für eine natürlichere Betonung bietet Amazon eine Möglichkeit. Mithilfe der *Speech Synthesis Markup Language* sowie den *speechcons* können kleinere Wortphrasen oder auch Sätze in einer natürlicheren Betonung ausgegeben werden. Dies hat vor allem eine natürlichere Sprachausgabe und damit einen menschlicheren Dialog zur Folge.

Eine andere sinnvolle Erweiterung des Skills ist das Erstellen von Erinnerungsmeldungen. Diese könnten durch Alexa ausgegeben werden, wenn beispielsweise ein Termin mit einem Helper bevorsteht oder der Helper eine Anfrage angenommen hat. Somit stünde dem Nutzer eine ganze Informationsebene live zur Verfügung.

7 Persönliches Fazit

Am Ende der Entwicklung dieses ersten Prototyps eines Skills für die Plattform Pflegix kann nun ein Fazit gezogen werden.

Zuerst einmal zeigt dieser Skill, dass durch Sprachassistenten und den natürlichen Umgang mit der Sprache die Eingangshürde in neue Technologien deutlich niedriger ist als zuvor. Anders als bei Smartphones, Tablets und Computern benötigen Sprachassistenten - abgesehen von der erstmaligen Einrichtung - nahezu keine technische Affinität. Stattdessen ist es für die Nutzer nur erforderlich, eine Hand voll Befehle zu kennen, die die Nutzung von Sprachassistenten zulassen.

Insbesondere zeigt diese Arbeit aber, dass viele Funktionen einer recht umfangreichen Webplattform auch durch die Sprache gesteuert werden können. Dabei möchte ich vor allem die Funktion des Nachrichtenversands hervorheben.

Durch die ausgezeichnete Sprache-zu-Text-Funktion der Alexa-Plattform ist es möglich, einfache Nachrichten über Alexa zu versenden. Diese werden zwar noch nicht mit Groß- und Kleinschreibung erkannt, aber dennoch bietet diese Funktionalität einen enormen Wertzuwinn für den Skill. Ob in Zukunft eine Erkennung der Groß- und Kleinschreibung durch Alexa direkt möglich sein wird, konnte nicht in Erfahrung gebracht werden. Um dieses Problem jedoch in begrenztem Maße zu umgehen, wurde die API von languagetool.org (2019) zur Korrektur von Groß und Kleinschreibung genutzt.

Zusätzlich zu dem Nachrichtenversand ist die Notfallfunktion des Skills anzuführen, um daran abgeleitet auch einen Schluss ziehen zu können, welche Auswirkungen diese Arbeit auf den gesamten Pflegebereich hat.

Die im Rahmen dieser Arbeit entwickelte Funktion für Notfallmeldungen zeigt sehr deutlich, welche Vorteile ein Sprachassistent haben kann. Insbesondere durch die Kombination mit bereits verfügbarer Hardware (Echo Connect) wird die Alexa-Plattform zu einer sinnvollen Investition für den Pflege-Sektor. Alleine durch die Möglichkeit, mittels eines Sprachzurufs einen Notfall zu melden, wird ein Gefühl der Sicherheit bei den Betroffenen auftreten aber andererseits nutzt diese Möglichkeit des Hilferufs in vielen Fällen in denen sonst keine Hilfe mehr gerufen werden könnte (Sturz mit Bewegungsunfähigkeit). Dadurch bietet gerade dieser Bereich innerhalb der Pflege ein enormes Potential.

Leider sind derzeit nur wenige spezielle Skills im Bereich der Pflege zu finden, die einen wirklichen Zugewinn für das Leben von Pflegebedürftigen darstellen. Daher ist zu hoffen, dass in naher Zukunft mehr in diesen Bereich investiert wird. Gerade in Hinblick auf sowohl den Fachkräftemangel als auch den demographischen Wandel würde dies einerseits dem Staat zugute kommen, indem Pflege gezielter bei den Bedürftigen ankommt, als auch den Pflegebedürftigen,

die durch Sprachassistenten und insbesondere auch solche Skills, wie dem hier entwickelten, einen enormen Zuwachs an Eigenständigkeit im Alltag erleben.

Daher stellt diese Arbeit nur einen Teil des derzeit schon Möglichen dar, zeigt aber auch, welches enorme Potential für den Bereich der Pflege die Sprachassistenten darstellen.

Persönlich bin ich mit dem Ergebnis des hier entwickelten Skills ausgesprochen zufrieden. Trotz anfänglicher Schwierigkeiten steht am Ende dieser Arbeit ein bereits brauchbarer und vor allem aber auch schnell erweiterbarer Skill, der den Zugang zur Plattform Pflegix vereinfacht. Zusätzlich dazu, habe ich durch die Entwicklung diese Skills für einen Sprachassistenten einen ganz anderen Blick auf Nutzerinteraktionen werfen können. Meiner Meinung nach bietet die Nutzung von Sprachassistenten einen enormen Zugewinn für das alltägliche Leben, da einfache Aufgaben per Sprache erledigt werden können.

Dennoch darf auch in dem Fazit der Kritikpunkt des Datenschutzes nicht fehlen. Denn auch wenn die Hersteller hinter den Sprachassistenten - durch beispielsweise das Löschen von Sprachaufnahmen - ein Gefühl der Sicherheit der eigenen Daten beim Nutzer suggerieren, sollte eben dieser Nutzer darauf achten, welche Informationen er diesen Herstellern zur Verfügung stellt.

Literatur

Languagetool, 2019. URL <https://languagetool.org/de/>.

Amazon.com Inc. Amazon Developer Console Build Page, 2018a. URL <https://m.media-amazon.com/images/G/01/DeveloperBlogs/AlexaBlogs/AlexaSkillsKit/Build{ }Page.{ }CB502299172{ }.png?t=true>.

Amazon.com Inc. Amazon, 2018b. URL <https://www.amazon.de/>.

Amazon.com Inc. Echo Connect, 2018c. URL <https://www.amazon.de/Echo-Connect-erfordert-Alexa-f{ä}iges-Telefonanschluss/dp/B071D5NW6R>.

Gabler Wirtschaftslexikon Online. Definition Ambient Assisted Living, 2018a. URL <https://wirtschaftslexikon.gabler.de/definition/ambient-assisted-living-53583/version-276661>.

Gabler Wirtschaftslexikon Online. Definition Virtueller Assistent, 2018b. URL <https://wirtschaftslexikon.gabler.de/definition/virtueller-assistent-99509>.

Marcin Grzegorzek, Claudia Muhl, and Ulrich Reiser. Mit Robotern gegen den Pflegenotstand Care-O-bot 4 View project. Technical report, 2017. URL <https://www.researchgate.net/publication/264040699>.

Frank Hassanabad. Oauth2orizeRecipes, 2018. URL <https://github.com/FrankHassanabad/Oauth2orizeRecipes>.

Andre Hellwig, Caroline Schneider, Sven Meister, and Wolfgang Deiters. Sprachassistenten in der Pflege - Potentiale und Voraussetzungen zur Unterstützung von Senioren. In Raimund Dachselt and Gerhard Weber, editors, *Mensch und Computer 2018 - Tagungsband*, Bonn, 2018. Gesellschaft für Informatik e.V. doi: 10.18420/muc2018-mci-0341. URL <https://dl.gi.de/handle/20.500.12116/16665>.

Catherine Jackson and Angela Orebaugh. A study of security and privacy issues associated with the Amazon Echo. Technical Report 1, 2018. URL <http://www.inderscience.com/storage/f184310116597122.pdf>.

Microsoft Corporation. Visual Studio Code, 2018. URL <https://code.visualstudio.com/>.

MongoDB Inc. MongoDB, 2018. URL <https://www.mongodb.com/>.

Mongoose. Mongoose ODM, 2018. URL <https://mongoosejs.com/>.

- Node.js Foundation. Express - Node.js web application framework, 2018a. URL <https://expressjs.com/>.
- Node.js Foundation. Node.js, 2018b. URL <https://nodejs.org/en/>.
- Sebastiaan Peek. *Understanding technology acceptance by older adults who are aging in place: A dynamic perspective*. Iskamp Printing, Enschede, 1 edition, 2017. URL <https://www.researchgate.net/publication/320508305>.
- Alfredo J Perez, Sheralli Zeadally, and Jonathan Cochran. A review and an empirical analysis of privacy policy and notices for consumer Internet of things. *Security and Privacy*, 1(3): e15, 2018. doi: 10.1002/spy2.15. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.15>.
- Pflegix GmbH. Pflegix, 2018. URL <https://www.pflegix.de>.
- Karen Renaud and Judy van Biljon. Predicting Technology Acceptance and Adoption by the Elderly: A Qualitative Study. In *Proceedings of the 2008 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries: Riding the Wave of Technology*, pages 210–219. ACM, 2008. doi: 10.1145/1456659.1456684. URL <http://doi.acm.org/10.1145/1456659.1456684>.
- Tejas Shah, Mike Reinstein, and Daniel Doubrovkine. Alexa Verifier Middleware, 2017. URL <https://github.com/alexa-js/alexa-verifier-middleware>.
- Alan Shreve. ngrok - secure introspectable tunnels to localhost, 2018. URL <https://ngrok.com/>.
- Liwei Song and Prateek Mittal. Inaudible voice commands. *CoRR*, abs/1708.07238, 2017. URL <http://arxiv.org/abs/1708.07238>.
- Statista. Digital Assistants - Always at Your Service, 2016. URL <https://www.statista.com/chart/5621/users-of-virtual-digital-assistants/>.
- Statista. Amazon's Alexa Is a Fast Learner, 2017a. URL <https://www.statista.com/chart/8304/alexa-skills/>.
- Statista. Where People Use Voice Assistants, 2017b. URL <https://www.statista.com/chart/7841/where-people-use-voice-assistants/>.
- Statista. Global smart speaker unit shipment, 2018a. URL <https://www.statista.com/statistics/792598/worldwide-smart-speaker-unit-shipment/>.

Statista. The Smart Speaker Race Is Heating Up, 2018b. URL <https://www.statista.com/chart/13931/smart-speaker-shipments/>.

Statistisches Bundesamt Wiesbaden. Pflegestatistik 2015 -Pflege im Rahmen der Pflegeversicherung - Deutschlandergebnisse. Technical report, 2017. URL www.destatis.de.