ulm university universität
uulm

Fakultät für
Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Datenbanken und Informationssysteme

Konzeption und Realisierung einer mobilen Anwendung zur Verbesserung der Koloskopievorbereitung

Abschlussarbeit an der Universität Ulm

Vorgelegt von:
Bastian Wankmüller
bastian.wankmueller@uni-ulm.de

Gutachter:
Prof. Dr. Reichert

Betreuer:
Marc Schickler

2018

Fassung 12. Juli 2018

Satz: PDF-LaTeX 2ε

Kurzfassung

In Deutschland gilt ab einem Alter von 55 Jahren die Darmspiegelung als eine Vorsorgeuntersuchung und wird von den Krankenkassen übernommen. Sie ist die wichtigste und beste Vorsorgeuntersuchung für Darmkrebs, an welchem 2014 ca. 61.000 Menschen in Deutschland neu erkrankten. Bei einer Darmspiegelung wird ein Endoskop mit Kamera und weiteren Instrumenten in den Darm eingeführt um diesen zu begutachten.
Um eine gute Sicht in den Darm zu erlangen, muss dieser gereinigt werden. Dies ist mit einer aufwändigen Diät und zeitgerechter Einnahme des Abführmittels des Patienten verbunden. Nur bei einer guten Vorbereitung ist eine klare Sicht möglich. Hierfür wurde eine App entwickelt, welche den Patient bei der Vorbereitung unterstützt. Dafür werden Benachrichtigungen erzeugt mit Tipps zu der Diät und als Erinnerung für die Einnahme des Abführmittels. Auch kann über den Service Watson Assistant, einem Chatbot, fragen über die Ernährung gestellt werden und in der App alle relevanten Daten nachgeschaut werden.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Eine Darmspiegelung oder auch Koloskopie ist eine der wichtigsten Vorsorgeuntersuchungen, dabei ist sie die beste Methode zur Früherkennung von Darmkrankheiten. Durch eine Darmspiegelung kann der Arzt neben dem bekanntesten Grund, des Darmkrebs, auch Entzündungen, Durchblutungsstörungen sowie weitere Veränderungen der Darmschleimhaut erkennen. So lässt sich das Risiko durch die Entfernung von Polypen senken, aus welchem ein Krebsgeschwür hervorgehen kann. Eine Untersuchung findet sowohl zur Vorsorge als auch zur Untersuchung bei Darmbeschwerden wie z.B. blutigem Stuhl oder länger anhaltendem Durchfall/Verstopfung statt. So starben in 2013 insgesamt 12.085 Frauen und 13.608 Männer an den Folgen von Darmkrebs[19] und 2014 erkrankten ca. 28.000 Frauen und 33.100 Männer neu[5].
Hierfür wird in Deutschland eine Darmspiegelung als Vorsorgeuntersuchung ab einem Alter von 55 Jahren von den Krankenkassen übernommen[18], diese kann nach unauffälligem Befund nach zehn Jahren wiederholt werden.
Während der Darmspiegelung führt der Arzt ein Endoskop in den Darm ein, über welches er zum einen Luft bzw. Kohlendioxid in den Darm pumpt, um diesen zu weiten. Zum anderen befinden sich im Endoskop mehrere Instrumente wie ein Licht, eine Kamera mit Spülung und spezielle Instrumente um etwaige Polypen zu entfernen oder Proben zu entnehmen.
Im Normalfall dauert eine Darmspiegelung ca. 30 Minuten, falls der Darm gut gereinigt ist. Um dies zu bewerkstelligen ist eine aufwändige Vorbereitung auf Seiten des Patienten notwendig, wie das einhalten einer speziellen Diät[4].
Ist der Darm nicht ausreichend gesäubert, so befinden sich noch Rückstände von Essen bzw. trübe Flüssigkeiten im Darminneren vor. Diese können die Sicht des Endoskop verschlechtern oder sich auf der Linse festsetzen oder blockieren als Ablagerungen in der Darmschleimhaut die Sicht auf diese. Muss der Arzt nun die Ablagerungen während der Untersuchung entfernen und den Darm weitgehend rei-

nigen, so kann sich die Darmspiegelung erheblich verlängern. Bei stark vernachlässigter Vorbereitung muss die Darmspiegelung an einem neuen Termin wiederholt wird[17].
Ziel dieser Anwendung ist es dem Darmspiegelungspatienten zu helfen seine Vorbereitung optimal gestalten zu können. Hierfür zeigt die Anwendung alle relevanten Daten zur Vorbereitung und der einhergehenden Diät an und gibt dem Benutzer über Benachrichtigungen Tipps für den aktuellen Tag und dessen Ernährungsplan. Da trotz Diätplan noch Fragen zur Ernährung aufkommen können und Lebensmittel je nach Person eventuell unterschiedlich eingeordnet werden, wird der Service von IBM Watson Assistant genutzt, um einen Chatbot zu bauen. Dieser soll auf Fragen des Benutzers, ob dieser ein Lebensmittel in der aktuellen Phase seiner Vorbereitung zu sich nehmen darf beantworten. Erkennt der Chatbot eine Frage oder ein Lebensmittel nicht, so ist es dem Benutzer möglich eine E-Mail an den zuständigen Arzt mit der Frage zu senden.

1.2 Inhalt

In dieser Arbeit ging es darum,die Kenntnisse in Android weiter zu vertiefen und herauszufinden, in welchem Maß der Service Watson Assistant als Chatbot für Fragen fungiert. Hierfür wurde eine Android-App entwickelt, welche dem Benutzer bei der Vorbereitung zu einer Darmspiegelung hilft. Die App sollte die Aufgaben erfüllen, den Benutzer über Vorbereitungsschritte zu den Nahrungsmitteln und dem Medikament benachrichtigen, die relevanten Daten bereitstellen und eine Schnittstelle zu Watson bieten. Da eine Darmspiegelung in Deutschland ab 55 Jahren als Vorsorgeuntersuchung übernommen wird von den Krankenkassen, musste auch die Oberfläche einfach bedienbar und intuitiv sein.

1.3 Struktur

In dieser Arbeit werden zunächst die Anforderungen an das System definiert in Kapitel 2, diese sind notwendig um alle Features, die die Anwendung später erfüllen soll festzuhalten. Diese gliedern sich ich Funktionale und Nichtfunktionale Anforderungen.
Folgend werden die Grundlagen der Arbeit erläutert, dazu zählt eine kurze Einführung in die Vorbereitungsschritte für die Darmspiegelung in Kapitel 3 und eine

theoretische Zusammenfassung, wie Watson entstanden ist und abstrahiert funktioniert in Kapitel 4.
Daraufhin folgt in Kapitel 6 die Konzeption des Projektes in Form von Mockups und einer Beschreibung dieser, wie sich die App im späteren Verlauf verhalten soll. Vor dem endültigen Fazit und einem Ausblick auf Weiterentwicklung in Kapitel 8 folgt noch die Implementierung in Kapitel 7. In ihr werden die Funktionen der Anwendung erklärt, die zugrundeliegende Datenbank und das Datenmodell in Form von JSON. Neben der Anwendung selbst wird noch auf die Konfiguration von Watson eingegangen und zuletzt ein Anforderungsabgleich zu den vorher erwähnten Anforderungen gemacht.

2 Anforderungen

2.1 Funktionale Anforderungen

Funktionale Anforderungen spezifizieren das Endprodukt in den vorhanden Features. Für jedes Feature gibt es eine Funktionale Anforderungen, welche dieses abdeckt. Jede Anforderung besitzt neben einem Titel eine Beschreibung und die Abhängigkeiten zu anderen Anforderungen. Alle hier aufgeführten Anforderungen sind gleich priorisiert, deshalb wurde auf eine einzelne Priorisierung der Anforderungen verzichtet. Es müssen für ein gelingen der App daher alle Anforderungen umgesetzt werden.

2.1.1 Funktionale Anforderungen an die Anwendung

FA-1	Anmeldung
Beschreibung	Um die Anwendung für mehrere Personen nutzbar zu machen, muss ein Benutzer sich am System mit einem eigenen Account anmelden können.
Abhängigkeiten	

FA-2	Anzeige von verbleibender Zeit
Beschreibung	Eine der wichtigsten Informationen für die Vorbereitung, ist die verbleibende Zeit, da sich danach die Diät richtet. Dafür soll dem Benutzer direkt auf der Startseite angezeigt werden, wieviel Zeit noch bis zur Untersuchung verbleibend ist in der Form TT/SS.
Abhängigkeiten	FA-1, FA-7

FA-3	FAQ
Beschreibung	Der Benutzer soll nicht nur über die Benachrichtigungen informiert werden, es soll in einer Übersicht dargestellt werden, auf was in jeder Phase geachtet wird und wie die Medikamente eingenommen werden müssen.
Abhängigkeiten	FA-6

FA-4	Chat mit Watson
Beschreibung	Damit auf den Chatbot Watson zugegriffen werden kann, soll eine kleine Chatanwendung erstellt werden, in der mit Watson eine Unterhaltung geführt wird.
Abhängigkeiten	

FA-5	E-Mail an Doktor
Beschreibung	Da Watson nicht alle Fragen abdecken kann bzw. eventuell Fragen auch nicht versteht, soll es dem Benutzer möglich sein, wichtige Fragen direkt an den Arzt per E-Mail zu senden.
Abhängigkeiten	FA-8

FA-6	Registrierung: Auswahl Medikament
Beschreibung	Bei der Registrierung kann der Benutzer selbst auswählen, welches Medikament er als Abführmittel wählt.
Abhängigkeiten	FA-8

FA-7	Registrierung: Angabe von Untersuchungstermin
Beschreibung	Damit die Anwendung weiß, wann welche Benachrichtigungen erforderlich sind, muss der Benutzer den genauen Termin mit Uhrzeit angeben.
Abhängigkeiten	FA-8

FA-8	Registrierung: Anmeldeinformationen und Doktor E-Mail
Beschreibung	In der Registrierung werden die Informationen wie E-Mail und Passwort des Benutzers abgefragt für die Anmeldung. Zudem wird die E-Mail des Doktors abgefragt, um diesem später eine E-Mail schreiben zu können.
Abhängigkeiten	

Benachrichtigungen

FA-9	Ernährungstipps
Beschreibung	Der Benutzer erhält täglich eine bestimmte Anzahl an Benachrichtigungen, welche ihm helfen, eine optimale Vorbereitung zu vollziehen.
Abhängigkeiten	FA-7

FA-10	Weiterleitung auf FAQ-Seite
Beschreibung	Um alle wichtigen Informationen selbst nachschauen zu können, soll jede Benachrichtigung die Möglichkeit haben, auf die FAQ-Seite weiterzuleiten, sodass der Benutzer selbst alle Informationen überschauen kann.
Abhängigkeiten	

FA-11	Medikamentenerinnerung
Beschreibung	Neben der Ernährungsumstellung bekommt der Benutzer auch Benachrichtigungen, wann und wie er das Medikament einnehmen muss, um eine optimale Vorbereitung zu erlangen.
Abhängigkeiten	FA-6

FA-12	Frage von Watson an Arzt weiterleiten
Beschreibung	Kann eine Frage nicht von Watson beantwortet werden, so soll er mit einem Klick auf die dann erscheinende Benachrichtigung direkt zur Mailseite kommen um eine E-Mail an die hinterlegte Adresse des Arztes zu senden.
Abhängigkeiten	FA-8

2.1.2 Funktionale Anforderungen an Watson

FA-12	Antwortvariationen
Beschreibung	Bei gleichen Fragen soll es möglich sein, verschiedene Antworten zu liefern, um so eine natürlichere Unterhaltung zu führen.
Abhängigkeiten	

FA-13	Antwort passend zur Anfrage
Beschreibung	Watson soll dem Benutzer eine Antwort liefern, welche auf die Frage bzw. den Typ und die Eigenschaften der Anfrage eingeht. Somit wird eine natürlichere Unterhaltung ermöglicht.
Abhängigkeiten	

FA-14	Unterscheidung zwischen Phasen
Beschreibung	Im Dialog mit Watson wird die aktuelle Phase, jedes einzelnen Benutzers in der Wahl der Antwort berücksichtigt.
Abhängigkeiten	

FA-15	Unterscheidung von Tageszeiten
Beschreibung	Je nach Tageszeit soll automatisch die richtige Antwort gewählt werden, falls der Benutzer keine Mahlzeit angibt.
Abhängigkeiten	

FA-16	Allgemeine Fragen
Beschreibung	Neben den Fragen zu der Ernährung, soll Watson auch einige allgemeine Fragen zu der Untersuchung beantworten können.
Abhängigkeiten	

2.2 Nichtfunktionale Anforderungen

NFA-1	Usability
Beschreibung	Die Oberfläche soll intuitiv gestaltet sein, sodass nur für den Benutzer relevante Informationen sichtbar sind.

NFA-2	Effizienz
Beschreibung	Es sollen lange Ladezeiten verhindert werden.

NFA-3	Energiesparend
Beschreibung	Die App soll nicht dauerhaft Berechnungen ausführen, so soll der Akku des Smartphone geschont werden.

NFA-4	Zuverlässigkeit
Beschreibung	Die App soll bei einer richtigen Eingabe ein richtiges Ergebnis produzieren und falsche Eingaben abfangen.

3 Koloskopie

3.1 Vorbereitung

Für eine Darmspiegelung ist eine gut durchgeführte Vorbereitung entscheidend. Während den unterschiedlichen Vorbereitungsphasen wird der Benutzer dazu aufgefordert eine spezielle Diät einzuhalten, um seinen Darm gründlich zu reinigen. Bei der Darmspiegelung dürfen keine festen Bestandteile oder trübe Flüssigkeit vorhanden sein, welche die Sicht des Endoskops verschlechtern. Die Vorbereitung wird hier in vier Phasen unterteilt, welche im nächsten Abschnitt genauer erläutert werden.
Ohne ausreichende Vorbereitung und Reinigung des Darms muss dies der Arzt während der Untersuchung tun, wodurch diese sich deutlich verlängern kann. Nach einer Studie werden ca. 22% der Polypen übersehen, welche kleiner sind als 1cm[20], deshalb ist eine gute Vorbereitung notwendig für eine erfolgreiche Darmspiegelung.

3.2 Phasen

Die Vorbereitung kann in vier verschiedene Phasen unterteilt werden. Die Phasen bauen nacheinander auf sich auf, sodass nach jeder abgeschlossenen Phase neue Nahrungsmittel hinzukommen zu der Diät[9][11].

- 4 Tage oder mehr verbleibend:
  Die tatsächliche Vorbereitung beginnt 3-4 Tage vor der Untersuchung. Bis dahin darf der Patient sein gewohntes Essen zu sich nehmen. Es wird empfohlen schon ab 4-5 Tagen auf körnerhaltige Nahrung und Früchte zu verzichten. Diese können sich in der Schleimhaut festsetzen und die Kamera blockieren.

- 2 bis 4 Tage verbleibend:
  In diesem Abschnitt gilt ein striktes Verbot an Nahrung welche Kerne enthält wie zum Beispiel Müsli, Kiwis, Weintrauben und Vollkornprodukte. Kerne können bis zur Untersuchung im Darm verbleiben und die Kamera des Endoskops blockieren.
- 1 Tag verbleibend:
  Am Vortag ist ein leichtes Frühstück, wie ein Joghurt, erlaubt ohne Kerne oder Vollkornprodukte. Für das Mittagessen ist eine klare Brühe ohne Einlagen wie Nudel oder Kräuter möglich und keine feste Nahrung mehr erlaubt. Auf das Abendessen wird verzichtet und bis zur Untersuchung nichts mehr gegessen.
- Tag der Untersuchung:
  Am Tag der Untersuchung ist jegliche Nahrung bis zum Zeitpunkt der Untersuchung untersagt. Bis ein paar Stunden vor der Untersuchung muss auch das trinken gestoppt werden, das Trinken darf eine helle und klare Flüssigkeit sein. Nach der Untersuchung ist wieder normales Essen und Trinken erlaubt, allerdings wird empfohlen mit leichter Kost anzufangen.

4 IBM Watson

Das Ziel der IBM Forschung besteht unter anderem darin, Computersysteme zu entwickeln und verbessern. So setzte sich IBM als Herausforderung im Gebiet der natürlichen Fragenbeantwortung ein System zu entwickeln, dieses sollte zu einem Menschen in der amerikanischen Quizshow "Jeopardy" gleichstark sein. Durch wachsendes Interesse an Systemen, welche die natürliche Sprache verstehen und diese präzise analysieren können, wählte IBM diese Domäne. Das interessante daran für IBM war die Schwierigkeiten die diese Herausforderung mit sich brachte, da eine Kombination aus Informationsbeschaffung aus der Frage, die Interpretation und Auswertung dieser mithilfe künstlicher Intelligenz und Maschinenlernen zu bewältigen. Das Projekt lief unter dem Codenamen Watson, benannt nach dem Gründer von IBM: Thomas J. Watson.

4.1 Die Jeopardy-Herausforderung:

Die Anforderungen an Watson wurden mithilfe der Herausforderung von Jeopardy erstellt. In Jeopardy müssen die Teilnehmer aus einer großen Menge aus unterschiedlichen Fachbereichen Fragen in Form natürlicher Sprache schnell und sicher beantworten. Für falsche Antworten gibt es Strafen. Somit kommt es speziell auf Sicherheit, Präzision und Geschwindigkeit beim beantworten der Fragen an. Kein bisher existierender Algorithmus ist perfekt im Sprachverständnis um die Fragen zu verstehen, daher müssen alle Teile ein Maß an Sicherheit produzieren und weitergeben, um diese zu Kombinieren für die endgültige Antwort. In Jeopardy wird dieses Maß genutzt, um zu entscheiden ob eine Antwort gegeben werden wird.
Jede Frage bzw. Aufgabe besteht dabei aus der Kategorie und einem Tipp. Dabei sind einige Kategorien essentiell um den Tipp zu verstehen, manche sind hilfreich aber unnötig, und einige sind nutzlos, wenn nicht sogar irreführend für einen Computer [1].

[1] Sinngemäß übersetzt aus [15]

Somit stellen die Tipps selbst eine Aufgabe dar, indem analysiert und herausgefunden werden muss, was gefragt ist und welche Teile zu der richtigen Antwort führen. Eine Frage besitzt allerdings nicht immer nur einen direkten Tipp, so müssen auch die Tipps auf mehrere Teiltipps untersucht werden. Dadurch ergeben sich nun zwei einzelne Tipps, wodurch nach zwei Antworten gesucht werden kann und die Schnittmenge beider Teile die Antwort zur ursprünglichen Frage enthält. Am Ende einer Runde Jeopardy gewinnt der Spieler mit dem meisten gesammelten Geld.
Da es neben den normalen Fragen noch drei doppelt gewertete Fragen gibt und am Ende eine finale Jeopardy-Frage steht, gibt das Geld alleine am Ende nicht wieder, wie gut der Spieler im gesamten Spiel abgeschlossen hat.
Um zu gewinnen müssen somit möglichst viele Fragen korrekt beantwortet werden, dafür werden die Präzision und die Prozent der beantworteten Fragen betrachtet. Die Präzision stellt hier den prozentualen Teil an Fragen dar, welche von Watson richtig beantwortet werden von den Fragen welche Watson an sich beantwortet. Die Prozent von beantworteten Fragen beschreibt den Anteil, welcher von Watson versucht wird zu beantworten, egal ob richtig oder falsch.
Ob eine Frage beantwortet wird oder nicht, hängt von dem Sicherheitsranking der Antwort ab. Je nach festgesetzter Schranke werden Fragen beantwortet, bei welchen eine Antwort eine höhere Sicherheit besitzt richtig zu sein als die Schranke. Dadurch kann das Verhalten durch den Kompromiss von Sicherheit und Risiko bestimmt werden, ob Watson riskiert die Frage falsch zu beantworten oder weniger Fragen beantwortet, diese aber mit einer höheren Sicherheit.

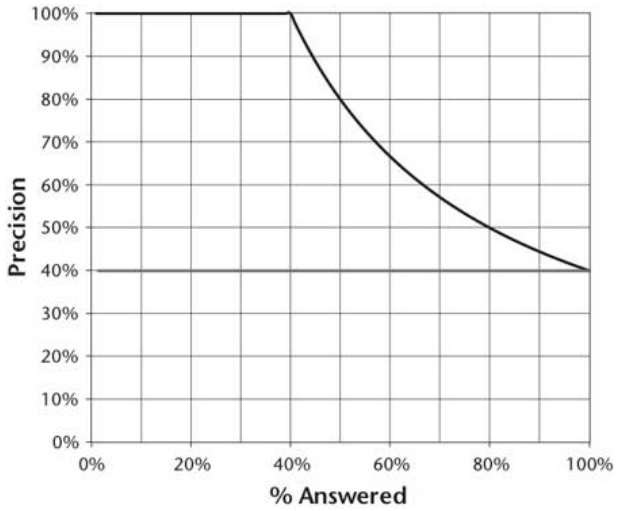

Abbildung 4.1: Präzision vs. Prozent beantworteter Fragen[15]

In Abb. 4.1 sind zwei theoretische Systeme zu sehen, beide haben eine Genauigkeit von 40%, beantworten also 40% der Fragen korrekt. Die obere Linie bildet ein System, welches ein Sicherheitsranking benutzt, welches für die 40% Genauigkeit perfekt ist, also bei 40% beantworteten Fragen diese immer korrekt sind, wohingegen das andere System nicht unterscheidet, ob die Frage richtig beantwortet wird oder nicht und somit eine konstante Präzision von 40% besitzt. Somit lässt sich mit gleicher Genauigkeit durch die Bewertung der Möglichen Antworten auf Richtigkeit eine höhere Präzision erzielen.

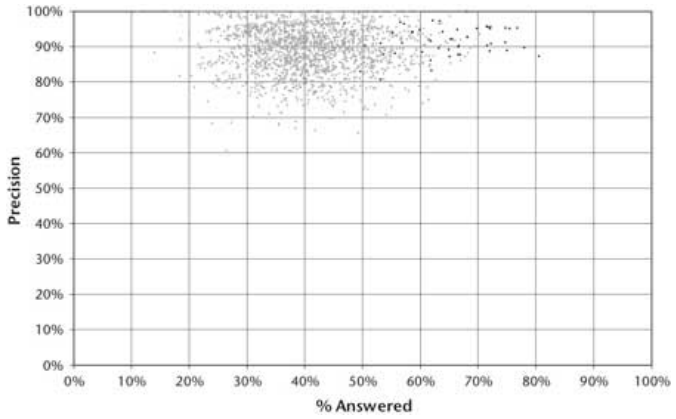

Abbildung 4.2: Menschliche Performance in Jeopardy[15]

In Abb. 4.2 ist die Verteilung von menschlichen Spielern zu sehen. Jeder Punkt stellt den Gewinner aus einer Runde Jeopardy dar, die Achsen sind gleich wie in Abb. 4.1, aus insgesamt 2000 analysierten Runden. Im Gegensatz zu den theoretischen Systemen besitzt ein Spieler lediglich einen Punkt und keine Kurve, da kein Sicherheitsranking dargestellt werden kann. Ein weiterer Unterschied ist hier, dass bisher der Aspekt des Buzzers für Watson außer acht gelassen wird, somit sind in dieser Abbildung nur die Fragen aufgeführt, welche der Spieler bekommen hat und schnell genug war. Dadurch ist auch die Zeit zu beachten, in welcher eine Antwort generiert und bewertet werden muss und wie hoch das Risiko gewählt wird eine falsche Antwort zu liefern.

4.2 DeepQA Architektur

Im folgenden wird die Architektur der DeepQA-Software auf einem hohen Abstraktionslevel gezeigt und Teile dessen beschrieben.

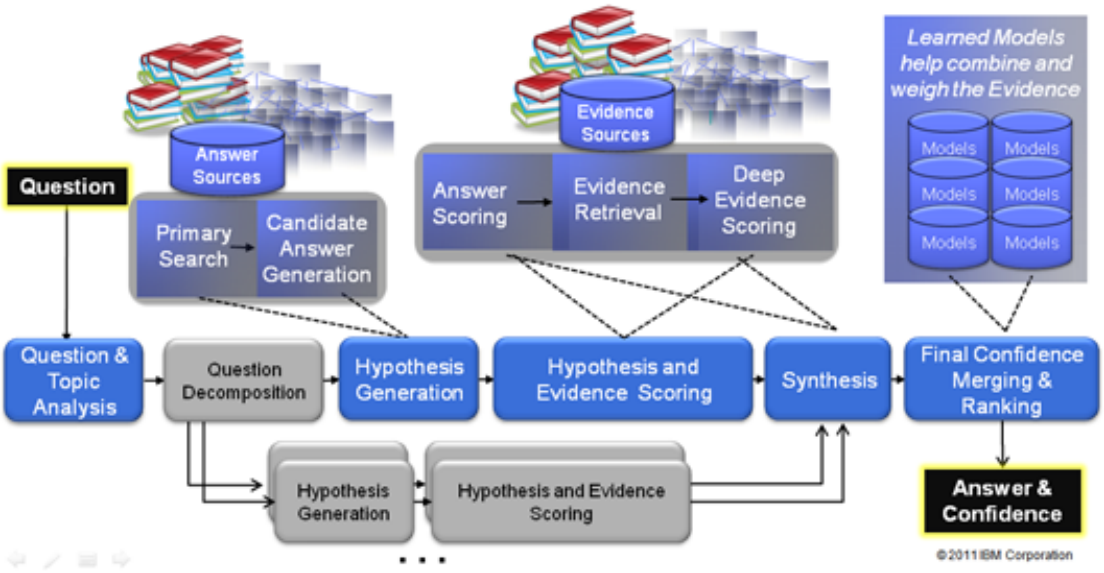

Abbildung 4.3: Architektur der DeepQA-Software auf hohem Abstraktionslevel[10]

Für die Architektur war vor allem wichtig, dass Anfragen parallel in Hinblick auf viele Interpretationsmöglichkeiten hin untersucht werden. Für den ersten Schritt der Auswertung ist also wichtig, die Frage und deren Kontext zu erfassen. Dazu werden zuerst viele Fragen aus der Domäne gewählt und durch diese die Fragetypen beschrieben. Um den Inhaltlichen Kontext der Frage zu erfassen, ist zudem wichtig Informationen zu der jeweiligen Domäne zu besitzen. Für die Jeopardy-Herausforderungen, welche aus vielen unterschiedlichen Domänen besteht, wurden als Quellen für Watson viele Enzyklopädien, Artikel etc. eingebunden. Nach dem einbinden von grundlegenden Daten und Fragen, wird der Wissensstand erweitert, dazu werden wichtige Dokumente aus der Domäne erkannt und weitere aus dem Internet heruntergeladen, diese werden dann textuell analysiert um wichtige Textpassagen zu filtern. Diese Textpassagen werden daraufhin bewertet ob diese informativ sind und zu den ursprünglichen Dokumenten passen. Übernommen werden dann nur diese Textpassagen, welche gut bewertet wurden in den Wissensstand.

Nun hat das System eine Wissensbasis, auf welcher es agiert. Als ersten Schritt wird die Frage analysiert, dafür versucht Watson die Frage zu verstehen und welche Antwort erwartet wird. Um nicht nur Fragen zu verstehen, welche aus nur einem Tipp bestehen, werden die Fragen auf Teilfragen bzw. Tipps untersucht.

So können wie zuvor erwähnt mehrere Tipps parallel verarbeitet werden und mehrere Antworten generiert werden.
Nachdem die Frage analysiert ist werden mithilfe der Analyse Hypothesen, also mögliche Antworten, generiert. Diese werden aus dem Wissen welches hinterlegt ist gesucht und Teile davon als Hypothesen zurückgegeben. Für jede Hypothese wird dann mithilfe der zugrundeliegenden Frage überprüft, ob diese eine mögliche Antwort darstellt und ein Sicherheitswert erstellt.
Auf die so erstellten Hypothesen wird ein Filter angewendet, welche die Hypothesen mithilfe des zuvor berechneten Ranking sortiert und die Hypothesen über dem Schwellenwert weitergibt zur weiteren Überprüfung.
Für jede der nun weiterhin zu überprüfenden Hypothesen werden weitere Analyseschritte durchgeführt, dafür wird für jede Hypothese durch auswerten von weiterem Wissen in Bezug auf diese ein genaueres Ranking erstellt. Dieser Teil stellt den größten Analyseschritt dar. Hierfür werden mehrere Arten der Bewertung benutzt, um anschließend für jeden Beweis zu einer Hypothese einen Rang zu erhalten, wie gut dieser die Hypothese beweist.
Nachdem alle Antworten gefiltert und bewertet wurden, gibt es noch einige Antworten, welche aus unterschiedlichen Quellen kommen und kleine Unterschiede haben, aber das gleiche Bedeuten. Die Äquivalenten Anworten werden dann von Watson erkannt und deren Bewertungen können kombiniert werden. Daraufhin werden die verbleibenden möglichen Antworten mit einem neuen Ranking versehen mithilfe der Kombinationen. Aufgrund dieses Rankings wird anschließend die Antwort gewählt und je nach Sicherheit bei Jeopardy der Buzzer betätigt.

5 Android

Android ist ein quelloffenes Betriebssystem für mobile Geräte wie Tablets und Smartphones. Entwickelt wurde Android von Google mit seiner Markteinführung in 2008. Seitdem wird Android von der Open Handset Alliance weiterentwickelt. Die Open Handset Alliance wurde 2007 von Google gegründet und ist eine Vereinigung von 84 Firmen im Bereich des mobilen Gerätemarktes und Technologie welche die Weiterentwicklung von Android betreiben [8].

5.1 Die Android Plattform

Die Androidkomponenten sind als Stapel entworfen, welche zu sehen sind in Abb. B.1. Die Grundlage bildet ein Linux Kernel, dieser basierte Anfangs auf der Version 2.6. Seit 2017 fordert Google bei neu hergestellten Geräten eine Version des Linux Kernels von 4.4 [2]. Der Linux Kernel übernimmt grundlegende Anforderungen wie Speicherverwaltung und Sicherheitsfeatures, zudem erlaubt es den Herstellern Treiber für einen weit verbreiteten Kernel zu implementieren.
Darauf aufbauend kommt die HAL-Schnittstelle, welche den darüber liegenden Schichten Standardschnittstellen bietet um auf die Hardwarekomponenten zuzugreifen.
Seit der Android-Version 5.0 werden Prozesse in ihrer eigenen Instanz einer Android-Runtime(ART) gestartet. Eine ART ist speziell dafür ausgelegt einen Prozess in einer virtuellen Maschine mit wenig Speicherverbrauch zu betreiben.
Neben Java, welches für die meisten Anwendungen benutzt wird, benötigen Komponenten wie HAL und ART teilweise native Bibliotheken in C und C++. Diese Bibliotheken können teils durch Java-APIs von Anwendungen benutzt werden.
Die Java-API bietet dem Entwickler den Funktionsumfang des Androidsystems. Diese Java-API bietet alle Funktionen zum Erstellen einer Android-App. Hierbei wird nicht unterschieden zwischen Systemanwendungen und eigener Anwendung, da der Entwickler den vollen Umfang der API nutzen kann.

5.2 Activity Lifecycle

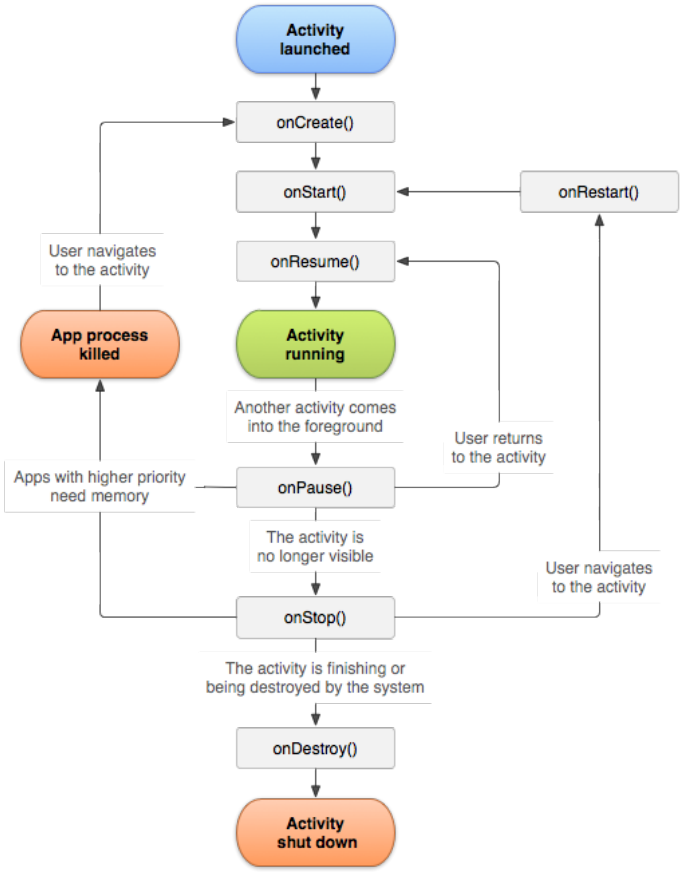

Abbildung 5.1: Darstellung des Lebenszyklus einer Activity[1]

Den Grundbaustein einer Anwendung bieten Activitys. Eine Activity besteht aus einer eigenen Activity-Klasse sowie einer eigenen Layout-Datei. Eine Activity durchläuft den in Abb. 5.1 dargestellten Zyklus von dem Zeitpunkt der Erstellung bis zur Löschung dieser aus dem Arbeitsspeicher des Smartphones.
Hervorzuheben ist hier die onCreate() und onStart() Methoden, in diesen findet die Initialisierung der Activity statt. Es werden alle benötigten Daten für die Activity aus z.B. der Datenbank gelesen und die Elemente der View gebunden.
Die Layout-Datei, welche jede Activity besitzt, ist als eine XML-Datei hinterlegt, in ihr wird die grafische Oberfläche und deren Elemente definiert und über IDs referenziert. Mithilfe der IDs ist es anschließend im Java-Code der Activity einfach die Elemente im Code zu verknüpfen und zu verändern.

5.3 Fragments

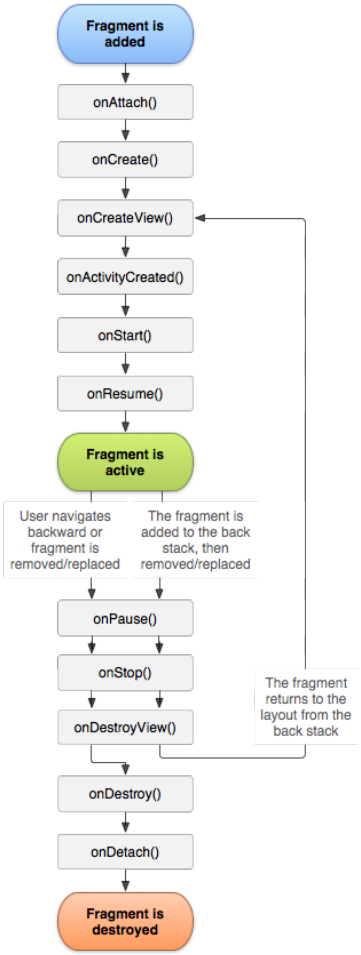

Abbildung 5.2: Darstellung des Lebenszyklus eines Fragmentes

Fragmente können ein Teilbereich der grafischen Oberfläche sein innerhalb einer Activity. Sie können allerdings nur im Kontext einer eigenen Activity existieren, besitzen aber ihren eigenen Lebenszyklus mit eigenen Events und Oberfläche. Mithilfe mehrerer Fragmente können so vielschichtige Oberflächen gestaltet werden, wobei jedes einzelne Fragment ausgetauscht und unabhängig von anderen verändert werden kann.

6 Konzept & Entwurf

Die App richtet sich größtenteils an Menschen über 50 Jahren. Eine Darmspiegelung wird in Deutschland ab 55 Jahren als Vorsorgeuntersuchung empfohlen. Aufgrund dessen wurde das Design Schlicht gehalten und von der Startseite sind alle Funktionen sofort zu erreichen.
Im folgenden Kapitel wird das Konzept und ein Entwurf des Aufbaus und Funktionen mithilfe von Mockups aufgezeigt. Das Datenformat, für welches JSON gewählt wurden, wird hier nicht aufgeführt, da dies mit den wachsenden Anforderungen an dieses während der Implementierung iterativ entstanden ist.

6.1 Datenbank

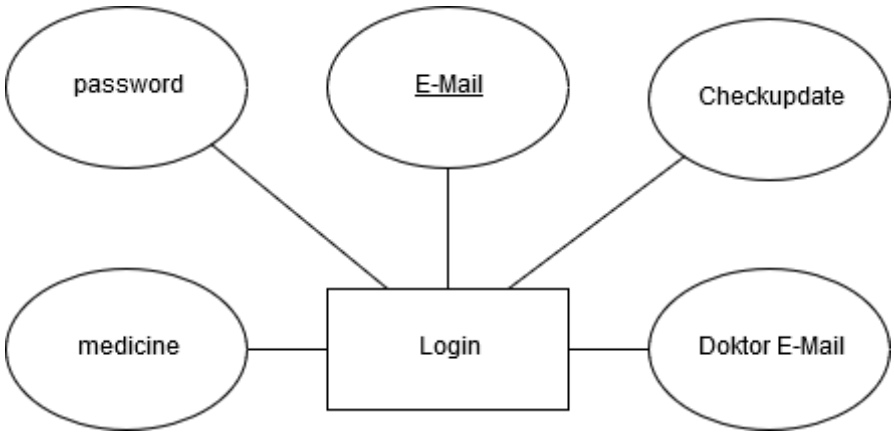

Abbildung 6.1: Konzept der Datenbank

In der Datenbank werden die wichtigsten Daten zu den Benutzern gespeichert. Die E-Mail dient als Schlüssel eines Datensatzes. Es werden alle Daten gespeichert, die der Benutzer bei der Registrierung angibt. Für den Untersuchungstermin wird ein formatierter Text gespeichert mit dem Datum und der Uhrzeit.

6.2 Mockups

Die Mockups wurden mit Hilfe von Ninjamock entworfen. Ninjamock ist ein Browser-basiertes Tool um Mockups mithilfe einer grafischen Oberfläche und einem Drag & Drop System zu erstellen. Als Features bietet Ninjamock dem Benutzer die Möglichkeit, die Elemente auf der GUI anzuordnen per Abstand zum Rand des Fensters, diese Anordnungsoption ist auch wiederzufinden im ConstraintLayout von Android. Jedes Element kann mit einem Link zu einem anderen Mockup versehen werden, dadurch kann eine Simulation einer App erstellt werden.

Login:

Abbildung 6.2: Konzept der Loginseite

Die Loginseite ist die Startseite der App, welche bei jedem Neustart der App geöffnet wird. Zu sehen ist ein kleiner Text mit einer Beschreibung der Grundidee. Für den Login stehen ein Feld für die eigene E-Mail und ein Feld für das Passwort bereit, welche beim Einloggen auf Richtigkeit überprüft werden müssen. Neben der Option sich am System anzumelden besteht auch die Möglichkeit sich einen neuen Account zu erstellen, dies führt zu der Registrierung.

Registrierung:

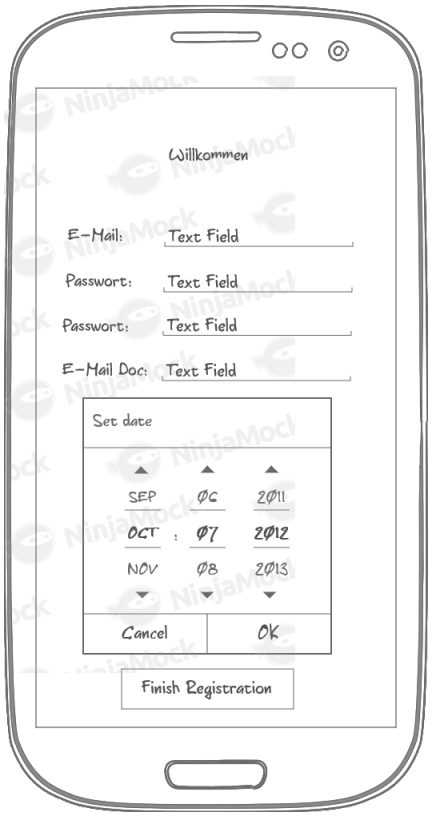

Abbildung 6.3: Konzept der Registrierungsseite

Für eine Registrierung am System, muss der Benutzer für die spätere Anmeldung seine eigene E-Mail und ein Passwort angeben. Um sicherzustellen, dass das Passwort richtig eingegeben wurde, ist unter dem ersten Passwortfeld eine zweites zur Wiederholung. Das letzte Textfeld ist für die E-Mail des zuständigen Doktors, um so später einen Kontakt herzustellen, falls Fragen zur Vorbereitung nicht durch die App selbst oder Watson beantwortet werden können. Als vorletztes Element auf der Seite befindet sich ein DatePicker, um das Datum der Untersuchung festzulegen. Von diesem Termin aus werden dann in der App alle weiteren Prozesse und Benachrichtigungen beeinflusst. Zuletzt gibt es noch einen Knopf um die Registrierung abzuschließen. Vor dem Abspeichern der eingegebenen Daten sollen diese alle nochmals überprüft werden, ob es sich um valide E-Mailadressen handelt, das Passwort den Anforderungen entspricht und das Datum nicht in der Vergangenheit liegt. Ist die Registrierung erfolgreich, so wird der Benutzer auf die Loginseite weitergeleitet.

Startseite:

Abbildung 6.4: Konzept der Startseite

Die Startseite ist in drei Teile unterteilt. Der Kopfbereich des Layouts bilden zwei Informationseinheiten welche horizontal unterteilt sind. Die obere Einheit beinhaltet die Tage, welche bis zum Untersuchungstermin noch verbleibend sind, inklusive der Stunden. Die untere Einheit ist eine TextView welche dem Benutzer einen zu dem aktuellen Tag spezifischen Tipp zu seiner Vorbereitung bereitstellt. In der unteren Einheit befinden sich vier große Knöpfe, diese nehmen den Rest der Bildschirmoberfläche ein. Über diese sind direkt alle weiteren Activitys verfügbar.

Settings:

Abbildung 6.5: Konzept der Hauptseite für Einstellungen

Es werden dem Benutzer zwei Kategorien vorgegeben, um das Verhalten der App an seine eigenen Bedürfnisse anzupassen. Er kann die Benachrichtigungen anpassen, ob diese erscheinen und die Zeit, in welchen die Benachrichtigungen für die Ernährung erscheinen.

Abbildung 6.6: Konzept der Einstellungen für Benachrichtigungen

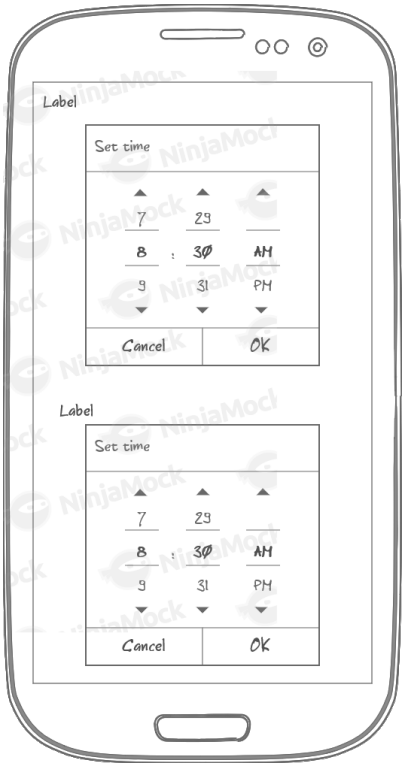

Abbildung 6.7: Konzept der Einstellungen für Zeitslots der Ben.

Unter den Einstellungen zu den Zeitslots finden sich zwei Einstellungsmöglichkeiten zu jeweils der Start und Endzeit zwischen welchen die Benachrichtigungen an den Benutzer gesendet werden sollen. Wird das Textfeld für die Start oder Endzeit geklickt, öffnet sich jeweils ein “TimePickerDialog” von Android, worüber die Stunde und die Minuten angegeben werden können. Unter den Einstellungen zu den Benachrichtigungen kann der Benutzer entscheiden, ob die Benachrichtigungen angezeigt werden, und ob diese Stumm sind beziehungsweise den aktuellen Einstellungen des Systems entsprechen. Benachrichtigungen werden zwischen zwei Arten unterschieden, Benachrichtigungen zu Ernährungstipps und Fehlerreport von Watson.

Watson:

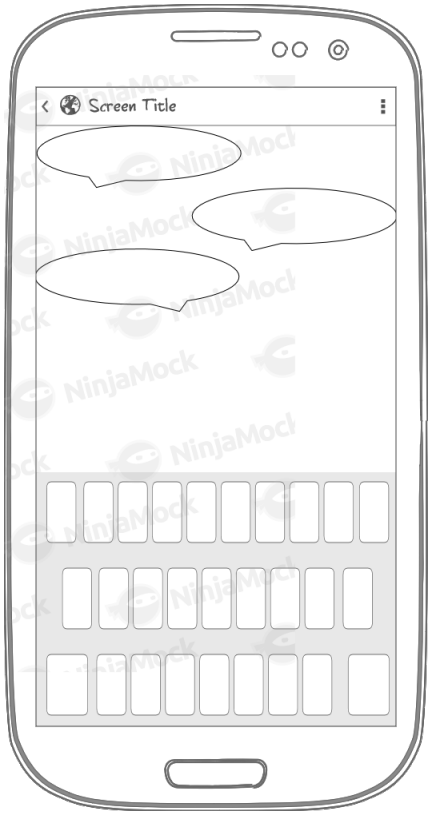

Abbildung 6.8: Konzept der Chatseite für Watson

Die Watson-Activity stellt dem Benutzer einen rudimentären Chat zur Verfügung. Wird die Activity geöffnet, so wird von der App automatisch eine Initialisierungsnachricht gesendet und Watson begrüßt den Benutzer entsprechend. Darauf beantwortet Watson Fragen über die aktuelle Diät und auch kleinere Aspekte über die Untersuchung allgemein. Fragen zur Medizin soll Watson bewusst nicht beantworten, da dies unter anderem abhängig von einer anderen Medikamentation des Benutzers sein kann.

Mail:

Abbildung 6.9: Konzept der Mailseite

Dem Benutzer wird eine einfache Benutzeroberfläche dargestellt mit zwei Texteingabefeldern. Der Benutzer muss lediglich einen Betreff angeben und den Inhalt verfassen. Nachdem die E-Mail von der App versendet werden soll, öffnet sich das Standard E-Mailprogramm des Smartphones um den Text an den angegebenen Arzt zu versenden.

FAQ:

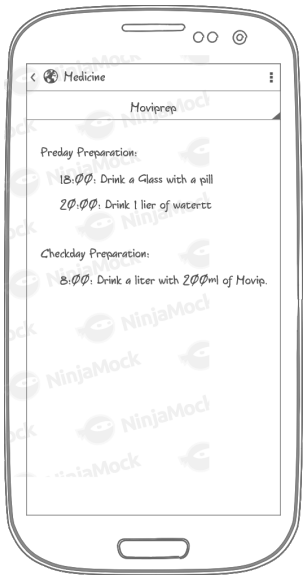

Abbildung 6.10: Konzept der FAQ-Seite: Medikament

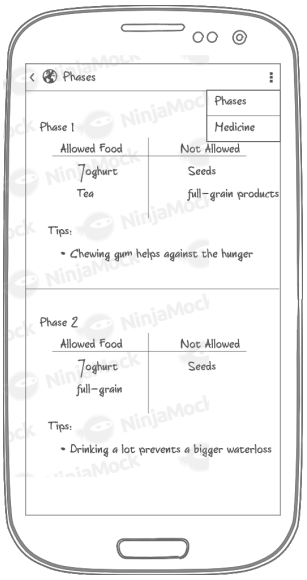

Abbildung 6.11: Konzept der FAQ-Seite: Phasen

In der FAQ-Activity sind alle von der App bereitgestellten Daten zur Vorbereitung enthalten und werden kompakt nach Phase oder gegebenenfalls nach Medizin präsentiert. Über ein Menü in der Taskbar der Activity ist es möglich zwischen den FAQ-Fragments der Medizin und der Phasen umzuschalten.

- Phase:
  Innerhalb des Phasen-Fragmentes werden alle Informationen zu einer Phase dargestellt. Für jede Phase gibt es eine Tabelle, welche unter " Allowed Food" (dt. erlaubtes Essen) Vorschläge für Essen gibt, welche aktuell der Vorbereitung helfen. Unter "Not Allowed" (dt. nicht erlaubt) ist die Diät der jeweiligen Phase aufgelistet mit Lebensmitteln, welche kontraproduktiv für die Vorbereitung sind und nicht verzehrt werden dürfen. Unter dieser Tabelle ist eine Liste mit Tipps zu finden. Diese Tipps helfen dem Benutzer seine Vorbereitung optimal zu gestalten und eventuelle Nebenwirkungen der Vorbereitung zu minimieren.
- Medizin:
  Im oberen Bereich des Medizin-Fragments befindet sich ein Spinner, um zwischen den verfügbaren Medikamenten zu wechseln. Aufgeteilt ist die Activity horizintal in zwei Teile für den Vortag und Untersuchungstag.

7 Implementierung

7.1 App

7.1.1 Datenbank

Die Datenbank der Anwendung wurde mit SQLite erstellt. Auf die Wahl eines Frameworks oder API für eine Datenbank wurde verzichtet, da die Datenbank mit drei Tabellen nicht sehr groß ist und diese einfache Datenwerte speichert und keine Objekte durch Tabellen dargestellt und gespeichert werden müssen.
Die Datenbank besteht, wie zuvor erwähnt aus drei Tabellen. Darin werden die Anmeldedaten der Benutzer lokal gespeichert sowie die JSON-Datei und die Benachrichtigungen mit ihrer Zeit als Zwischenspeicher.

Die Tabelle für Medizin ist hier in keiner Abbildung aufgeführt, da diese aus nur einem Eintrag mit einer Spalte besteht, welche das JSON-Format beinhaltet.

Tabelle für Benutzerdaten:

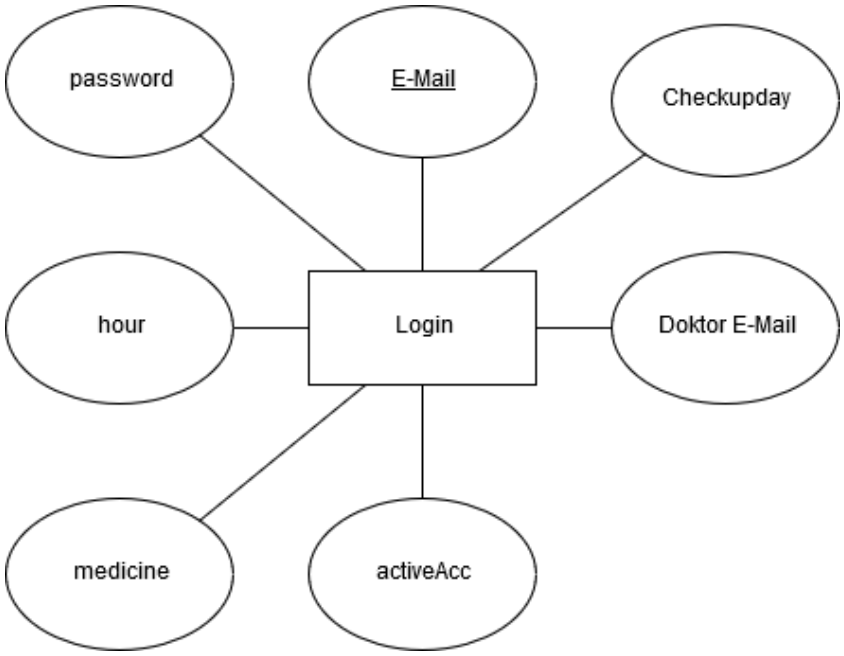

Abbildung 7.1: ER-Diagramm der Logintabelle

In dieser Tabelle werden alle Daten zu einem Benutzer gespeichert, welche er bei der Registrierung angibt. Die E-Mail des Benutzer selbst ist hierbei der Schlüsseleintrag einer Zeile, über welche diese eindeutig referenziert werden kann. Des weiteren wird über die Benutzer-E-Mail auch die E-Mail des zuständigen Doktors zugeordnet, sodass diese öfters in dem Datensatz vorhanden sein darf.
Für die Anmeldung wird das Passwort gespeichert, da es sich hier um eine rein lokale Anwendung handelt im Klartext.
Für den Untersuchungstermin gibt es zwei Spalten, zum einen das Datum der Untersuchung und die Uhrzeit, zu welcher die Darmspiegelung planmäßig durchgeführt werden soll. Das Datum wird als Text gespeichert in der Formatierung "TT/MM/JJJJ", wodurch die Anzahl der Spalten, im Gegensatz zu einzelnen Spalten für Tag und Monat, gespart werden. Für die Vorbereitung wird hier auch das Medikament dem Benutzer zugeordnet. Die Spalte Aktiver Account dient hier, damit mehrere Benutzer die Anwendung gleichzeitig benutzen können. Es wird jeweils der aktuelle Benutzer als aktiv gesetzt und geändert, falls sich ein anderer Benutzer anmeldet.

Benachrichtigungstabelle:

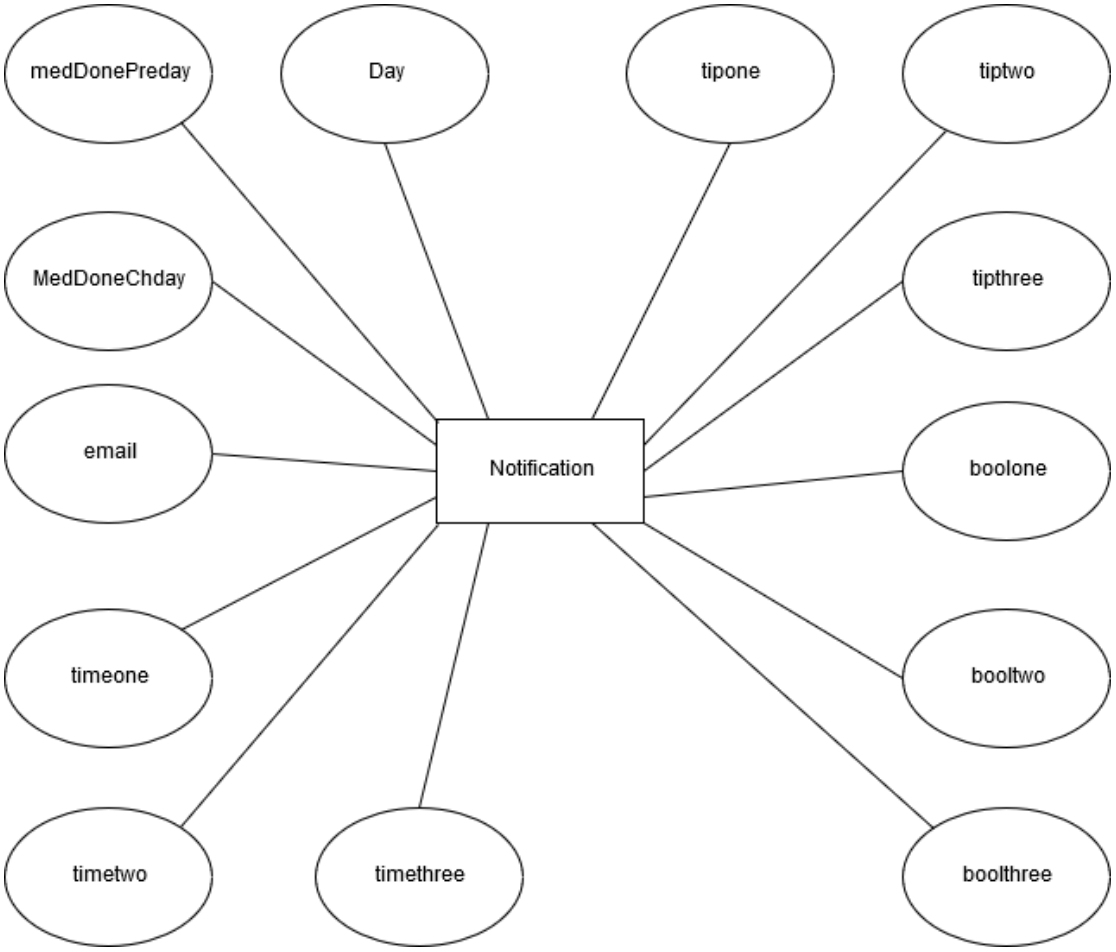

Abbildung 7.2: ER-Diagramm der Benachrichtigungstabelle

Für die Speicherung zu den Einzelheiten der Benachrichtigung dient diese Tabelle, in ihr werden die Daten über den Zeitraum des Tages gespeichert, sodass diese nicht durch einen Absturz verloren gehen. Da diese Tabelle nur eine Zeile enthält, entfällt die Wahl eines Schlüssel.
Da die Benachrichtigungen jeden Tag zufällig generiert werden, wird der aktuelle Tag des Monats gespeichert. Für die Benachrichtigungen werden die jeweils generierten Zeiten als Millisekunden gespeichert. Des weiteren gibt es drei boolean-Felder, in welchen gespeichert wird, ob die Benachrichtigung dem Benutzer schon angezeigt wurden, um sicherstellen zu können, dass keine Benachrichtigung doppelt erzeugt wird.

Für jede Zeit wird auch der Tipp in der Datenbank vermerkt, sodass die Erzeugung der Objekte aus dem JSON-Format nur bei der Generierung der Zeiten erfolgt.
Da die Benachrichtigungen für die Medizin am Vortag je nach Tageszeit festgeschrieben bzw. am Tag der Untersuchung sich nach der Untersuchungszeit richten, werden zu diesen keine Zeiten gespeichert. Hierfür gibt es zwei Spalten mit Text, "medPreDone" und "medChDone" . Diese beiden Felder beinhalten die Zeiten von den bisher erschienenen Benachrichtigungen, da es bisher keine zwei gleichen Zeiten gibt für ein Medikament, werden somit alle angezeigt. Falls es zwei gleiche Zeiten geben sollte, so werden diese beiden als zwei Benachrichtigungen gleichzeitig erzeugt. Um mehrere Benutzer auch in den Benachrichtigungen zu unterstützen, wird noch die E-Mail eines Benutzers gespeichert, für welchen die Zeiten generiert wurden.

7.1.2 Activitys

Registrierung:

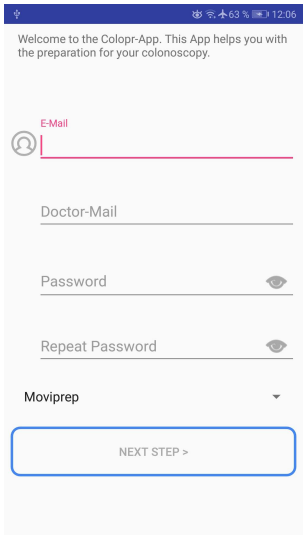

Abbildung 7.3: Abfrage der Benutzerdaten

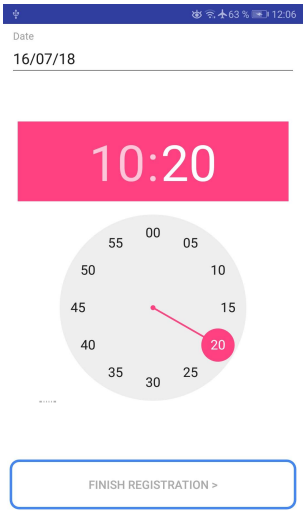

Abbildung 7.4: Abfrage zum Termin

Hat der Benutzer noch kein Konto bzw. keinen Account in der App erstellt, kann er er sich mithilfe von zwei Seiten registrieren. Auf der ersten Seite der Registrierung finden sich im Kopfbereich ein kleiner Einleitungstext. Darauf folgen vier Texteingabefelder für die E-Mail des Benutzer, die E-Mail des zuständigen Doktors und das Passwort mit Passwortwiederholung. Zuletzt befindet sich ein Spinner zur Auswahl des gewählten Abführmedikaments und ein Knopf für die zweite Seite der Registrierung. Auf dieser befindet sich ein Textfeld, über welches sich bei einem Klick ein DatePickerDialog öffnet um das Datum der Untersuchung anzugeben. Folgend kommt ein TimePickerDialog um auch die genaue Uhrzeit der Untersuchung anzugeben. Mit einem Klick auf Registrieren wird die Registrierung abgeschlossen und die eingegebenen Daten überprüft. Bei erfolgreicher Registrierung wird die Login-Seite geladen und die Daten in der lokalen Datenbank gespeichert. Als Voraussetzung muss eine valide E-Mail Adresse angegeben werden und das Passwort muss aus mindestens sechs Zeichen bestehen.

Login:

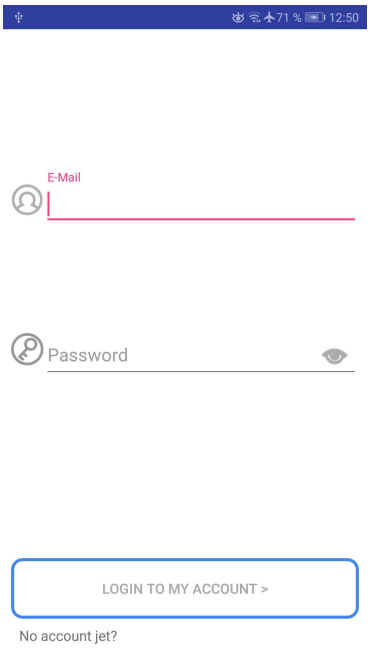

Abbildung 7.5: Loginseite der App

Die Login-Seite besteht aus einem Textfeld für die E-Mail des Benutzers und dessen Passwort. Bei erfolgreicher Anmeldung gelangt der Benutzer auf die Startseite, bei einer falschen Eingabe erscheint eine Meldung mit dem Text "Login failed". Unter dem Login-Knopf befindet sich ein in kleiner Text als Link zu der Registrierung für neue Benutzer.

Startseite:

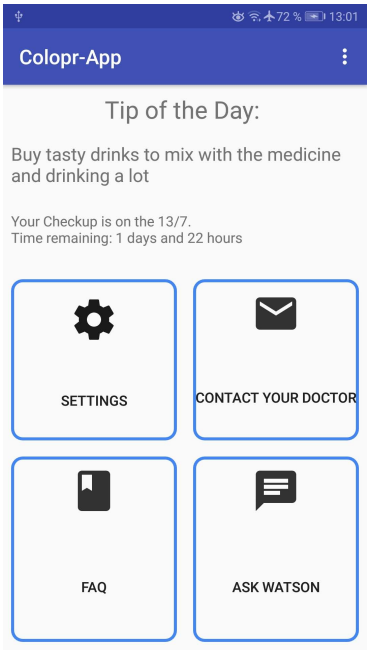

Abbildung 7.6: Startseite der App

Nach der erfolgreichen Anmeldung gelangt der Benutzer auf die Startseite, welche einen ersten Überblick ermöglicht. Im oberen Bereich der Seite findet sich der Tipp des Tages, dieser gibt dem Benutzer je nach Vorbereitungsphase einen Tipp welcher für die Zeit einer Phase konsistent bleibt.
Darunter befinden sich zwei Textzeilen für eine direkte Erinnerung an den Termin. In der oberen Zeile wird textuell dargestellt an welchem Datum die Untersuchung stattfindet. Die zweite Zeile zeigt einen Timer an, welcher die noch verbleibende Zeit bis zu der Untersuchung in Tagen und Stunden anzeigt. Dadurch weiß der Benutzer direkt nach dem Einloggen in die App wieviel Zeit noch verbleibend ist. Der nachfolgende Bereich, welcher den Hauptteil der Seite darstellt, bietet einen direkten Zugriff auf alle wichtigen weiteren Activitys der App. Diese sind wie in dem Entwurf in 6.2 und deren Anordnung und Verlinkung wurde daraus übernommen. Die vier Knöpfe beinhalten jeweils ein Icon, welche den Nutzen der aufzurufenden Seite beschreiben und eine textuelle Beschreibung.

Einstellungen:

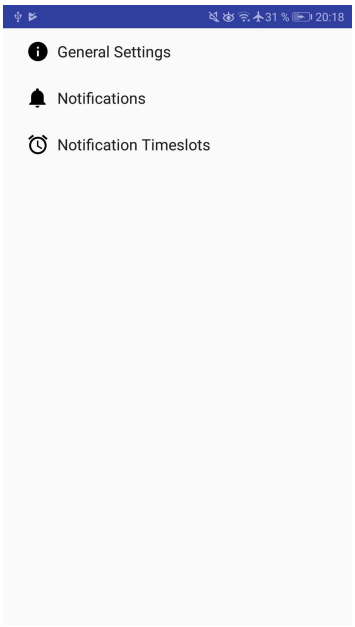

Abbildung 7.7: Hauptseite der Einstellungen

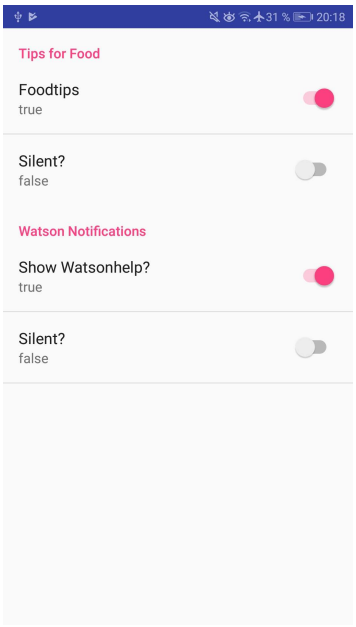

Abbildung 7.8: Einstellungen zu Benachrichtigungen

Realisiert wurden die Einstellungsoptionen mithilfe der SharedPreference-API von Android, mithilfe dieser können schnell Optionen realisiert werden dank einfacher Anpassung von XML-Code. Die Einstellungen der App wurden in drei Kategorien eingeteilt zur individuellen Anpassung an den jeweiligen Benutzer.
In den allgemeinen Einstellungen gibt es aktuell nur eine Einstellungsmöglichkeit, ob die E-Mails in der App selbst geschrieben werden und dann mit Inhalt an die Standard E-Mailapplication des Handys weitergeleitet wird oder beim Klick auf der Startseite direkt eine E-Mailapp aufgerufen wird mit lediglich der E-Mail des Doktors.
In den Einstellungen zu den Benachrichtigungen gibt es zwei Unterkategorien, Ernährungstipps und Watsonsupport. Für jede dieser Kategorien kann jeweils separat festgelegt werden, ob diese Benachrichtigungen erscheinen und ob diese Stumm sind. Dadurch ist dem Benutzer die Möglichkeit gegeben keine Benachrichtigungen zu empfangen welche nicht erwünscht sind. Medikamentenerinnerungen werden immer gesendet, da diese den Hauptbestandteil der Vorbereitung darstellen.

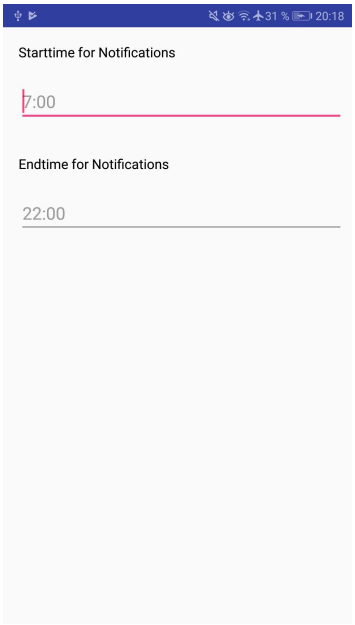

Abbildung 7.9: Übersicht zu Start und Endzeit

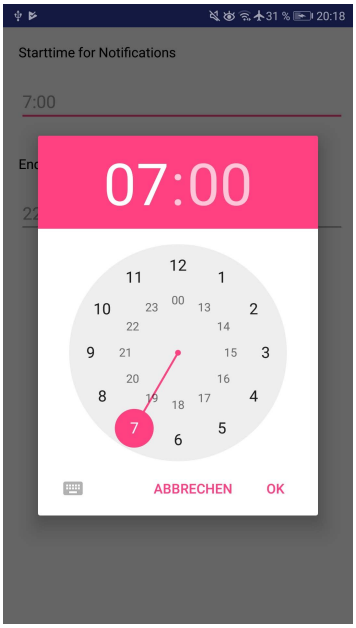

Abbildung 7.10: Aufgeklappter TimePickerDialog

Im dritten Unterpunkt befinden sich die Optionen zu welcher Zeit Benachrichtigungen zur Ernährung erscheinen. Die Benachrichtigungen zu den Medikamenten sind darin nicht enthalten, da diese von dem Untersuchungstag und der Untersuchungszeit direkt abhängen. Für die Start- und Endzeit für die Benachrichtigungen gibt es zwei Textfelder, mit einem Klick auf eines der beiden Textfelder öffnet sich ein TimePickerDialog von Android, über welchen der Benutzer die Zeit in Stunden und Minuten genau angeben kann.

E-Mail:

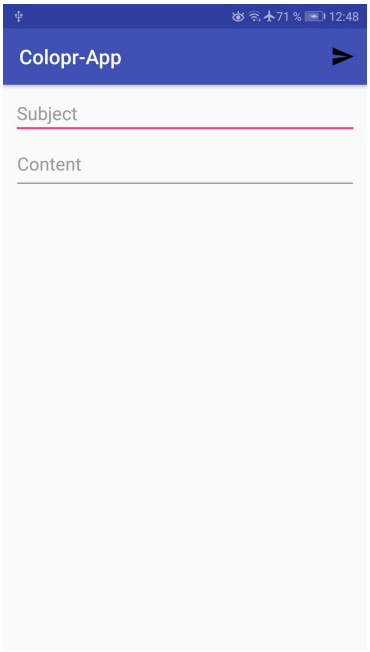

Abbildung 7.11: Mailseite

Die Seite bietet eine einfach gehaltene Oberfläche, auf ihr sind zwei Textfelder für den Betreff der zu verschickenden E-Mail und dem Inhalt. Soll die E-Mail gesendet werden, wird eine Uri erstellt mit den Daten zur Doktor-E-Mail, dem Betreff und dem Inhalt. Über diese Uri mit dem Schema "mailto" wird die Standard E-Mailapp des Smartphones geöffnet oder ggfs. eine Auswahl an möglichen Apps vorgeschlagen. Damit öffnet die gewählte App ein E-Mail Fenster zum versenden, wobei alle Felder eingetragen sind und die E-Mail direkt versendet werden kann.

FAQ:

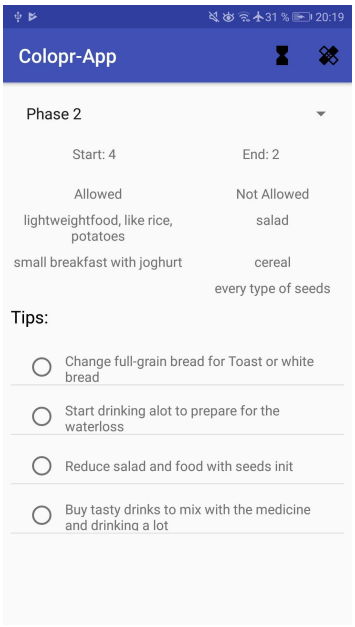

Abbildung 7.12: FAQ zu den Vorbereitungsphasen

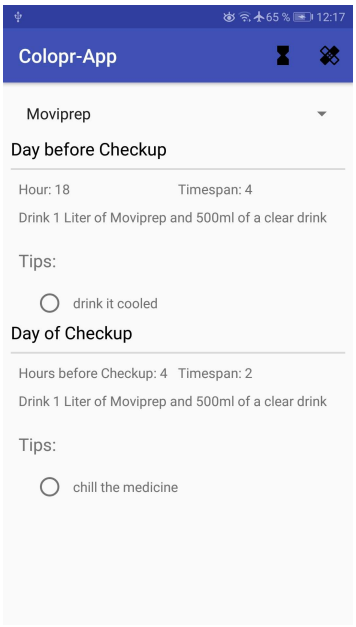

Abbildung 7.13: FAQ zu den Medikamenten

Die FAQ-Seite besteht aus zwei einzelnen Fragmenten, welche in das Layout der Activity geladen werden. Beim starten wird standardmäßig das Fragment für Phasen geladen. Die zwei Fragmente können über die Symbole in der Toolbar der Activity getauscht werden, es kann zwischen Phasen- und Medikamenten-FAQ gewählt werden. Alle Daten, welche angezeigt werden, sind in dem JSON-Format, siehe 7.2 , vorhanden und werden dem Benutzer übersichtlich und sortiert dargestellt.

- Phasen-Fragment:
  Das Phasen-Fragment zeigt dem Benutzer alle in der App vorhandenen Informationen zu den jeweiligen Vorbereitungsphasen. Beim Start wird aufgrund der verbleibenden Tage die aktuelle Phase geladen. Als oberstes Element ist ein Spinner zu sehen mit allen Phasen.
  Zu jeder Phase werden die Start- und Endtage angegeben, in welchen sie startet und endet. Folgend ist eine Tabelle, in der Spalte zu " Allowed" befinden sich Vorschläge bzw. Beispiele für Nahrungsmittel, welche der Vorbereitung helfen. Komplementär dazu sind in der zweiten Spalte mit "Not Allowed" Nahrungsmittel aufgeführt, welche die Vorbereitung negativ beeinträchtigen und somit die Diät des Benutzers darstellen. Unter der Tabelle ist noch eine Liste mit allen Tipps zu der angezeigten Phase um eine Möglichkeit zu bieten einen besseren Überblick zu bekommen.

- Medikamenten-Fragment:
  Wie in dem Phasen-Fragment befindet sich im oberen Bereich ein Spinner, hier sind nun alle verfügbaren Medikamente zum Abführen enthalten. Das Fragment ist in zwei Teile aufgeteilt, der erste Teil beschreibt, falls vorhanden, die Medikamenteneinnahme am Vortag der Untersuchung und der zweite Teil die Einnahme am Tag der Untersuchung. Die Struktur für Vortag und Untersuchungstag ist identisch. Für jede vorhandene Zeit wird ein Eintrag erstellt, dieser besteht aus zwei Zeitangaben. Die erste Angabe bezieht sich auf die Stunde, beim Vortag auf die Uhrzeit und beim Untersuchungstag auf die Anzahl der Stunden vor dem Termin. Die zweite Angabe gibt dem Benutzer die Zeitspanne an, in welcher der Vorbereitungsschritt ausgeführt werden muss. Zu jedem Vorbereitungsschritt gibt es eine Liste mit Tipps zur Einnahme des Medikaments.

Watson:

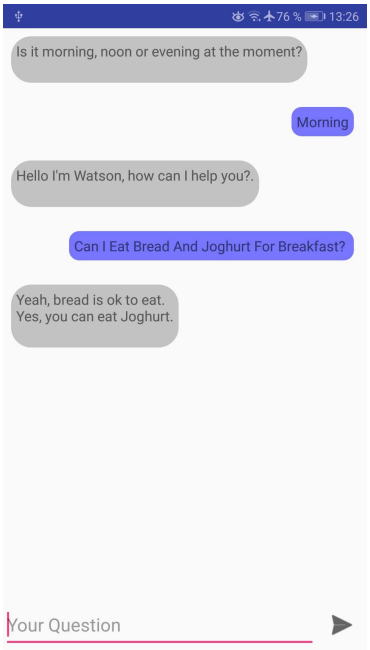

Abbildung 7.14: Beispielchat mit Watson

Auf der Chatseite für den Chatbot Watson befindet sich ein rudimentärer Chat. Wird die Seite geöffnet, so sendet die App automatisch eine Initialisierungsnachricht mit den Parametern des Tageszeitraums und den verbleibenden Tagen. Ausgehend von diesen Informationen werden die passenden Antworten gewählt. Unterstützt werden hauptsächlich Fragen zur Ernährung bzw. Lebensmitteln.

7.1.3 Benachrichtigungen

Die App bietet drei unterschiedliche Arten von Benachrichtigungen. Sie informiert den Benutzer je nach Tageszeit mit einem Tipp für die Ernährung und über die Vorbereitungsschritte für das Abführmedikament.
Zuerst wurden die Benachrichtigungen mithilfe eines Service implementiert. Dieser sollte im Hintergrund laufen, und im Falle einer Benachrichtigung einen IntentService nutzen um eine Benachrichtigung zu erzeugen. Da allerdings ab dem API Level 26 von Android Services im Hintergrund anders als zuvor behandelt und diese einschränkt, sodass der Service beendet wurde, sobald die App nicht mehr im Vordergrund ist. Deshalb wurden die Benachrichtigungen dann mithilfe einer Tabelle in der Datenbank als Zwischenspeicher über einen AlarmManager realisiert. Beim starten der App bzw. des Telefons wird eine Instanz des Alarms erstellt und jeweils nach einem Zeitintervall von einer Minute erneut aufgerufen, um zu überprüfen, ob eine neue Benachrichtigung erzeugt werden muss.

Ernährungtipps:

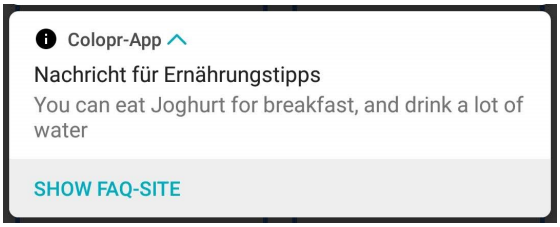

Abbildung 7.15: Benachrichtigung für Essenstipps

Für die Ernährung werden pro Tag drei unterschiedliche Tipps als Benachrichtigungen bereitgestellt, welche wie zuvor erwähnt zu der aktuellen Tageszeit passend sind. Als Standard ist in den Einstellungen ein Zeitrahmen von 7-22 Uhr eingestellt, in diesem werden die Benachrichtigungen zeitlich zufällig erzeugt. Der Zeitrahmen ist dabei in drei gleich große Zeiträume unterteilt, in welchen die Benachrichtigung erscheint. Nachdem die Benachrichtigung einmal erzeugt wurde, wird ein Eintrag in der Datenbank erstellt, sodass es keine Wiederholungen gibt. Die Nachrichten sind wie in 7.2.1 beschrieben der Tageszeit angepasst. In Anhang A ist zu sehen, wie die Generierung der Zeiten in der App funktioniert.

Medikamentenerinnerung:

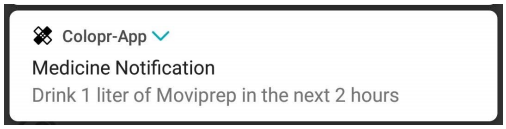

Abbildung 7.16: Benachrichtigung für Medikamente

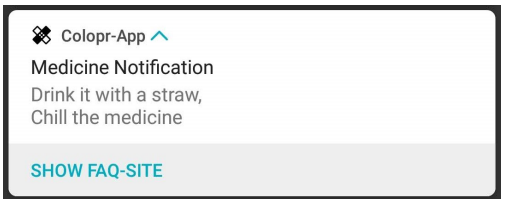

Abbildung 7.17: Ausgeklappte Benachrichtigung

Zu einer guten Vorbereitung gehört großteils die richtige Einnahme des Abführmittels. Die Benachrichtigungen sind abhängig von der Uhrzeit bzw. des Termins am Vor-/Untersuchungstag. Die Benachrichtigung enthält im Standardzustand einen Text, welcher textuell beschreibt wie das Abführmittel eingenommen werden soll und in welchem Zeitrahmen dies geschehen muss. Sie ist aufklappbar und enthält darin alle vorhandenen Tipps.

Unbeantwortete Frage an Watson:

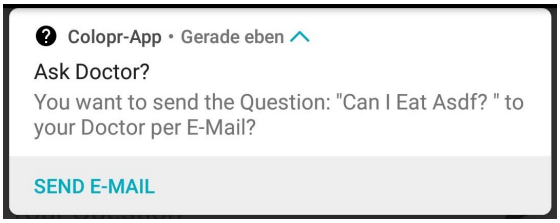

Abbildung 7.18: Benachrichtigung für eine nicht erkannte Frage

In dem Fall, dass Watson dem Benutzer keine Antwort geben kann, erscheint eine Benachrichtigung. Ist gerade ein Chat mit Watson aktiv, und es wird eine Anfrage gestellt, welche eine Fehlernachricht zurück gibt, so wird eine Benachrichtigung generiert, welche die Frage des Benutzers enthält. Diese bietet ihm dann auch die Möglichkeit, die Mailseite direkt mit der Frage aufzurufen.

7.2 JSON-Format

Als Datenquelle wurde ein JSON-Format gewählt. In diesem JSON-Objekt werden alle Daten, welche relevant für die Vorbereitung sind, gespeichert. Das Format wurde gewählt, da sich mit JSON einfach komplexere Sachverhalte und Abhängigkeiten darstellen lassen. Für die App wurde darauf geachtet, dass alle Daten in diesem JSON-Objekt vorhanden sind, um somit in einem späteren Server-Client Szenario diese Daten über Updates anpassbar zu machen und ggfs. personalisierte Vorbereitungspläne zu erstellen und einzufügen.
Das Format ist in zwei Teile unterteilbar, es gibt ein JSON-Objekt für alle Phasen der Vorbereitung und jeweils für jedes Medikament ein JSON-Objekt als direkte Kindelemente.

7.2.1 Phasen

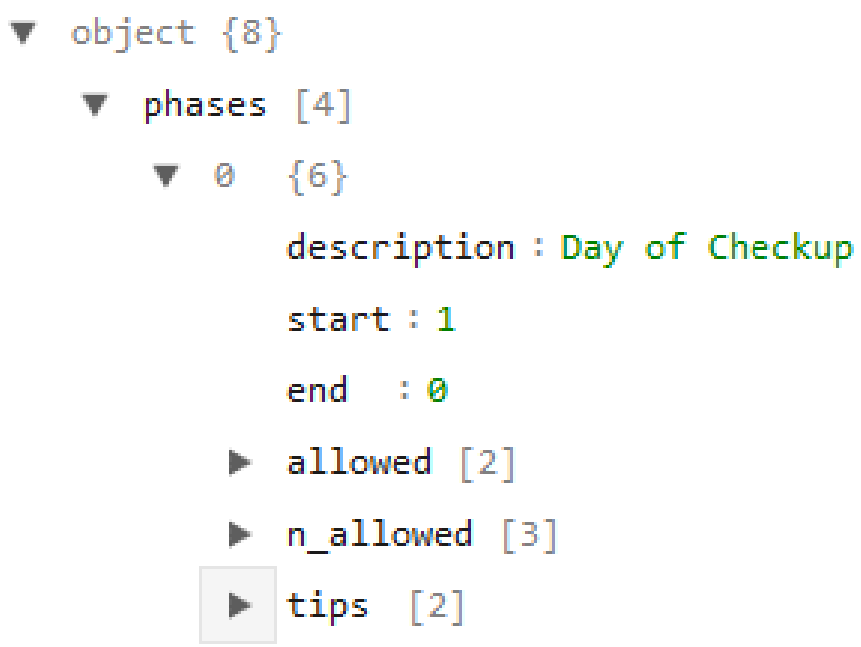

Abbildung 7.19: Aufbau des JSON-Formates für eine Phase[1]

Ein Phasenobjekt beschreibt die Aspekte der Vorbereitung des jeweiligen Zeitabschnittes. Jedes Objekt besitzt dabei sechs Kindelemente.
Es beinhaltet eine Beschreibung als Text, daraufhin kommen der Start- und Endtag, welche die verbleibenden Tage der Phase abbilden bis hin zum Untersuchungstag. Der Starttag wird excludiert und der Endtag wird inkludiert, wodurch sich feste Grenzen ergeben und ggfs. auch Überschneidungen möglich sind zwischen Phasen.
Die verbleibenden drei Kindelemente sind Arrays mit Text, zwei dieser Arrays sind für die optimalen beziehungsweise nicht erlaubten Lebensmittel. Das letzte Array beinhaltet Tipps für die Vorbereitung, diese sind für jede Phase spezifisch und geben dem Benutzer die Option seine Vorbereitung optimal zu vollziehen. In dem Array befinden sich immer genau vier Tipps, dabei sind drei Tipps für die Ernährungstipps und einer für den Tipp des Tages auf der Startseite.

[1] https://jsoneditoronline.org/

7.2.2 Medikament

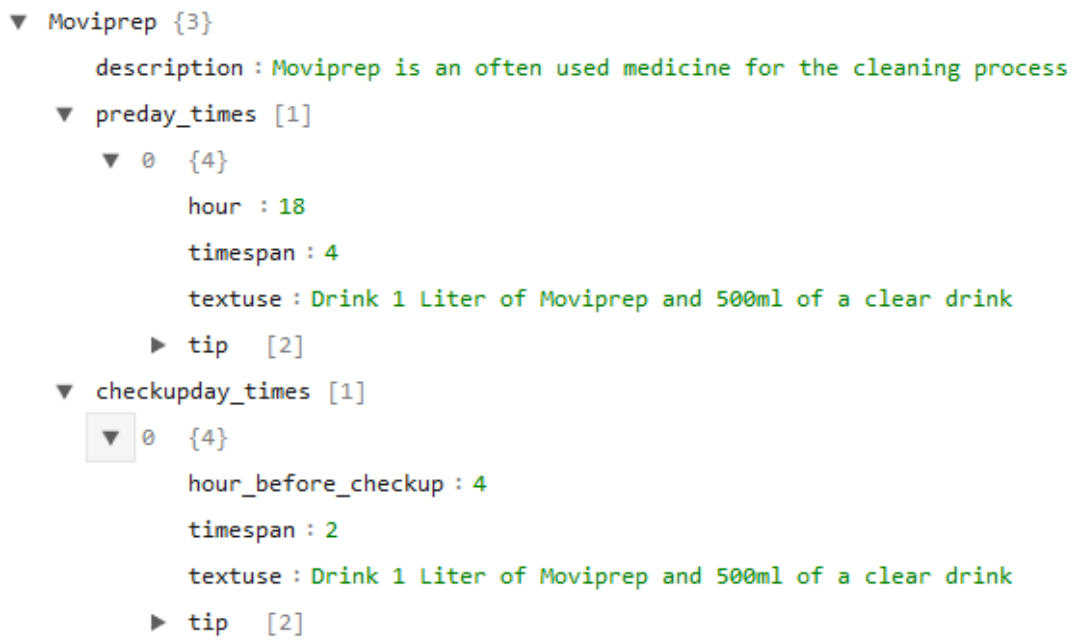

Abbildung 7.20: Aufbau des JSON-Formates für die Medikamente[2]

Ein Medikamentenobjekt besitzt drei Kindelemente, welche ein Beschreibungstext und zwei Arrays für die Vorbereitung sind. Die beiden Arrays beinhalten die Einnahmeinformationen zu dem Medikament für den Vortag bzw. den Untersuchungstag. Die Kindelemente der beiden Arrays folgen demselben Aufbau in ihrer Struktur, der einzige Unterschied besteht darin, dass für den Vortag die Stunde des Tages angegeben wird und beim Untersuchungstag dies abhängig von der Untersuchungszeit ist. Jedes Array besteht aus einem oder mehreren Vorbereitungsschritten.
Ein Vorbereitungsschritt beinhaltet die Zeit der Ausführung sowie die Zeitspanne in welcher diese vollzogen werden muss. Zu jedem Schritt existiert ein Text, welcher die Benutzung des Medikamentes beschreibt und ein weiteres Array welches Tipps zur Einnahme in Textform enthält.

[2] https://jsoneditoronline.org/

7.3 Watson

Watson bietet über eine Weboberfläche eine Vielzahl an Services an, welche auf dem System Watson basieren. Die Einarbeitung über diese gelang sehr schnell und einfach, auch die gute und verständliche Dokumentation von IBM selbst. Die Erstellung von Intents und Entities ist übersichtlich gestaltet, sodass die Konzeption des Dialogmoduls die hauptsächliche Arbeit darstellt.
Zu Beginn der Bachelorarbeit gab es noch kein Fuzzy Matching, so mussten die Entitäten genau in der Anfrage wiedergegeben werden, da sonst keine korrekte Erkennung stattfindet, siehe 7.3.2.

7.3.1 Intents

Ein Intent beschreibt eine Absicht, welche der Benutzer an Watson stellt bzw. welches Ziel oder welcher Zweck vom Benutzer gewünscht wird. Da Watson in meinem Anwendungsfall dafür gedacht ist Fragen zu beantworten, bestehen die Intents hauptsächlich aus Fragetypen.

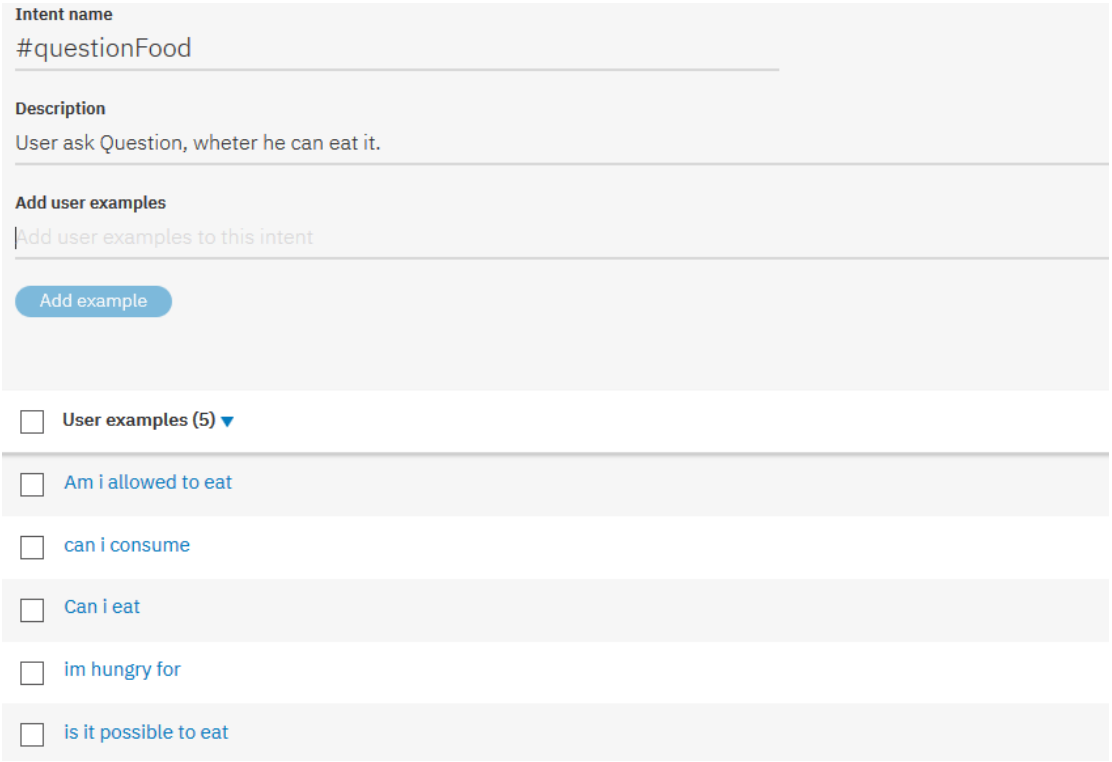

Abbildung 7.21: Beispiel eines Intents

Ein Intent besteht aus einer ID, diese beginnt immer mit einem #. Über die ID wird später der Intent im Dialog referenziert und erkannt. Zu jedem gibt es noch eine Beschreibung, welche lediglich für den Benutzer der Weboberfläche relevant ist. Der wichtigste Teil sind die Beispiele, für einen Intent können mehrere angegeben werden. Die Beispiele müssen relevant und repräsentativ sein, umso besser diese die Benutzereingaben repräsentieren, wird der richtige Intent erkannt. Watson bietet die Möglichkeit einer Mehrfacherkennung, so können auch zwei Absichten in einer Anfrage erkannt und verarbeitet werden.

7.3.2 Entities

Entitäten stellen ein Objekt bzw. einen primitiven Datentyp dar, welcher in Kombination mit einem Intent zu dem Verständnis der Absicht des Benutzers vervollständigt.

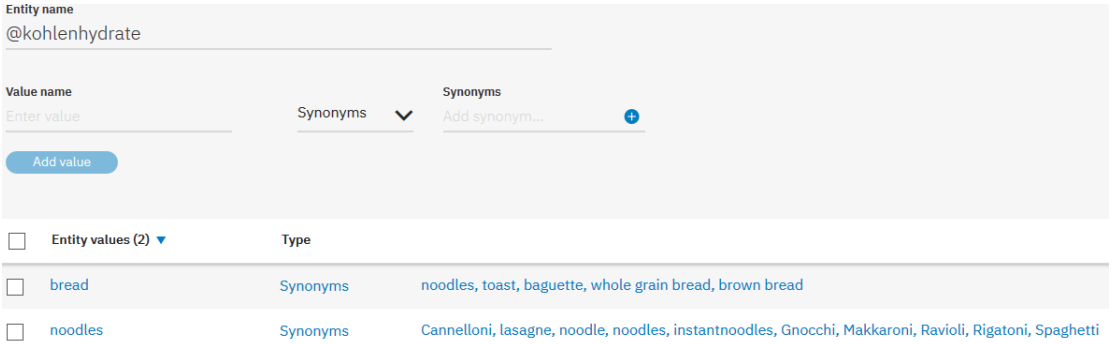

Abbildung 7.22: Beispiel einer Entität

Eine Entität besitzt auch eine ID, angefangen mit einem @, welche hierbei auch als Beschreibung dient. Für eine Entität können Werte angegeben werden, welche Oberbegriffe sind und für jeden Wert können weitere Synonyme angegeben werden. In Abb. 7.22 ist zu sehen, dass hier Nudel der Oberbegriff ist und weitere Arten von Nudeln als Synonyme angegeben sind. Neben Synonymen gibt es auch die Möglichkeit für Werte bestimmte Muster anzugeben welche auf Übereinstimmungen durch reguläre Ausdrücke geprüft werden.
Leider lassen sich keine Entitätengruppen einfach definieren, sodass eine Entität durch zwei oder mehr andere Entitäten definiert wird um grob zu filtern. So muss um eine Filterung zu realisieren alle Beispiele der Unterentitäten auch in der Entitätengruppe vorhanden sein.
Fuzzy Matching:
Standardmäßig ist dieses Feature deaktiviert, durch dieses kann die Erkennung von Entitäten in Benutzereingaben verbessert werden. Dadurch ist es nicht mehr nötig den genauen Wortlaut einer Entität und deren Beispiele anzugeben, sondern es wird auch auf Ähnlichkeiten in der Syntax geachtet. Dafür wendet Watson drei Teilkomponenten an:

- Normalformenreduktion:
  Hierbei wird der Wortstamm des Wortes überprüft, wo wird bei angegebenen plural oder einer Variation des Wortes der Wortstamm gesucht und auch auf diesen überprüft.

- Rechtschreibfehler:
  Ist in der Benutzereingabe ein Rechtschreibfehler bzw. eine leichte syntaktische Abweichung zu einer Entität, so wird diese trotzdem richtig zugeordnet.
- Teilweise Übereinstimmung:
  Hierbei wird nach einer teilweisen Übereinstimmung mit einer Entität gesucht. Im Gegensatz zu einer exakten Übereinstimmung werden nun der Entität eine niedrigere Präzision zugewiesen.

7.3.3 Dialog

Das Dialogmodul ist der Hauptbestandteil von Watson Assistant. Im Dialogmodul werden mithilfe der zuvor erstellten Intents und Entitäten ein Dialogbaum erstellt. Die Dialogstruktur wird hierbei von Watson von dem jeweils aktuellen Knoten bzw. vom Ursprung von oben nach unten durchgeführt.

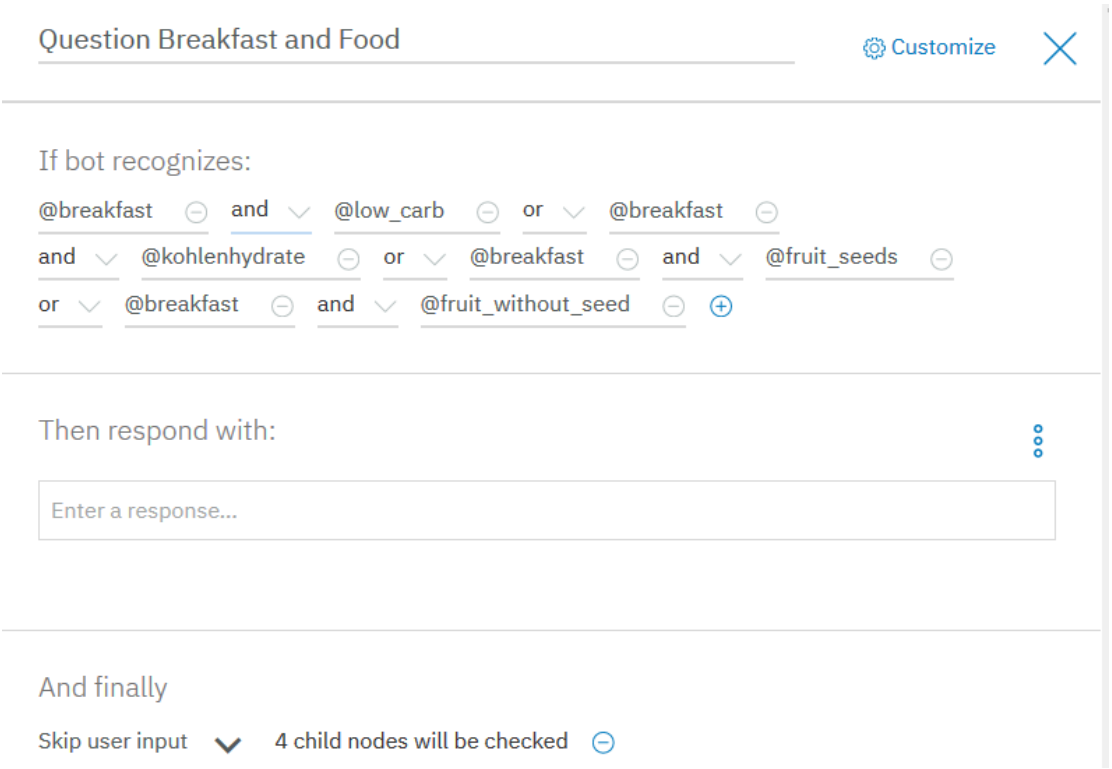

Abbildung 7.23: Beispiel eines Knoten

Jedem Knoten kann ein Name bzw. eine Beschreibung gegeben werden, welche im Dialogmodul in der Übersicht gezeigt wird. Ob ein Knoten ausgewertet wird, kann durch die zuvor erstellten Intents und Entitäten festgelegt werden, falls mehrere erkannt werden sollen, so lassen sich diese mit einer logischen Verknüpfung verbinden, siehe Abb. 7.23. Wie hier zu sehen ist, ist keine Klammerung möglich, sodass nur Paare von Verknüpfungen möglich sind. Standardmäßig hat ein Knoten eine oder mehrere Antworten, welche auf die gleiche Bedingung antworten die der Knoten selbst festlegt, und dadurch eine Varianz in den Antworten möglich ist.

Zuletzt muss noch angegeben werden, was passiert, nachdem der Knoten bzw. dessen Antwort ausgewertet wurde. Als Standard wird auf eine weitere Eingabe des Benutzers gewartet und diese dann von den Kindknoten des aktuellen Knoten ausgewertet. Sind keine Kindknoten vorhanden, wird die Eingabe vom Ursprung aus ausgewertet. Als weitere Möglichkeiten gibt es, eine Antwort zu überspringen und mit der aktuellen Eingabe weiter zu verfahren in den Kindknoten oder nach der Abhandlung des Knotens zu einem anderen Knoten zu springen, welcher dann auch ausgewertet wird.
Über Customize lässt sich ein Knoten mit weiteren Features verfeinern, diese werden im weiteren genauer Vorgestellt, den Kontext Editor kann man über die Standardansicht im Abschnitt der Antworten aktivieren.

weitere Features Dialog:

- Kontext:
  In dem Kontext einer Unterhaltung können Entitäten bzw. Werte in Variablen gespeichert werden. Diese können dann zur Auswertung in Knoten oder in der Antwort verwendet werden bzw. bei mehreren Antworten entscheiden, welche gewählt wird. Der Kontext wird für jede Unterhaltung neu Initialisiert und speichert die Werte während der Laufzeit der aktuellen Unterhaltung.

- Slots:
  Durch das Feature von Slots lassen sich mehrere Entitäten einfacher abfragen und angegeben werden vom Benutzer, ein Beispiel ist unter B.2 zu finden. Dadurch kann der Benutzer mit einer einzigen Eingabe mehrere Entitäten angeben, welche daraufhin im Kontext des Dialoges gespeichert werden können. Wird ein Knoten mit Slots ausgeführt, so wird überprüft, ob alle Entitäten angegeben wurden und diese ggfs. dann im Kontext gespeichert werden. Slots können optional oder erforderlich sein, wird eine Frage angegeben, so erscheint diese falls die Entität nicht erfüllt wurde und der Slot ist erforderlich. Wird keine Frage für die Entität angegeben, so ist diese automatisch optional und wird falls mitgeliefert gespeichert. Ist eine erforderliche Entität nicht erfüllt, so wird dem Benutzer die vordefinierte Frage nach dieser gestellt, solange bis alle Entitäten erfüllt sind. Ist keine einzige Entität gegeben, sondern lediglich die Bedingung des Knotens erfüllt, so kann eine Frage nach allen Entitäten geliefert werden. Wenn alle Slots angegeben wurden führt der Knoten normal fort mit der Antwort bzw. der Einstellung für das Beenden des Knotens.

- Multiple Answers:
  Mithilfe mehrerer Antworten ist es möglich, je nach Kontext bzw. Entitäten welche extra gegeben wurden, die Antwort aufgrund dessen zu variieren. Damit stellt jede Antwort eine Art kleinen Knoten dar. Auch hier gibt es den Kontext Editor, sodass die Eingabe im Kontext gespeichert werden kann. Je nachdem welche Antwortmöglichkeit gewählt wird, wird eine Antwort gegeben und anschließend wird Standardmäßig der Knoten beendet. Es gibt allerdings die Möglichkeit für jede Antwort zu einem anderen Knoten zu springen.

- Kontext Editor:
  In jedem Knoten lässt sich der Kontext Editor aktivieren, dadurch lassen sich Entitäten, welche für den Knoten angegeben sind speichern. Dieses Feature ist ähnlich zu Slots, der Unterschied hierbei ist, dass die Entitäten nicht erforderlich oder optional angebbar sind und der Bentuzer nicht aufgefordert werden kann, den fehlenden Wert anzugeben.

Watson übernimmt die Analyse des eingegebenen Textes auf die hinterlegten Intents und Entitäten. Ist die Eingabe ausgewertet so wird die Dialogstruktur von oben nach unten durchlaufen und bei einem passenden Wert der Knoten ausgewertet. Dadurch ist es anfänglich schwer mehrere Benutzereingaben auf einmal zu verarbeiten. Um mehrere Entitäten in einem Knoten zu verarbeiten, müsste dieser mit allen eventuell anderen möglichen Entitäten als Multiple Answers konfiguriert werden. Dies ist eine größere Einschränkung, da eine Frage im normalen Sprachgebrauch auch mehrere Lebensmittel mit einschließt und daraus nicht zwei oder mehr Fragen werden. So wurde lange Zeit die Möglichkeit direkt mehrere Lebensmittel auf einmal einzugeben auf ein einziges Eingeschränkt. Dieses Problem wurde mithilfe der Endbedingung eines Knotens gelöst, dadurch lässt sich nach dem Auswerten des ersten Knotens zu dem nächsten Knoten weiterspringen und diesen auch auswerten. Daraus folgt nun auch, dass der Benutzer für eine Eingabe von X unterschiedlichen Lebensmittel auch X Antworten erhält, für jeden Typ eine. Abgearbeitet werden die Knoten in der Reihenfolge, wie schlecht sie für die Vorbereitung sind. So fängt die Auswertung an mit Früchten, welche Samen enthalten und endet mit leichter Kost wie Joghurt oder Toast, welche kaum Rückstände hinterlassen und gut verdaulich sind.
Der Dialog beginnt, indem Watson den Nutzer begrüßt, dafür wird beim starten der Activity am Ende der Initialisierung eine Initialisierungsnachricht gesendet, diese enthält die zwei benutzten Kontextvariablen für die verbleibenden Tage und die Unterteilung der Tageszeit in Morgen, Mittag und Abend.

Im Dialogmodul sind drei Arten von Fragen hinterlegt, diese sind zuständig für Essensfragen, allgemeine Fragen zur Untersuchung und Fragen über die Dauer. Hauptsächlich wurde sich in dieser Arbeit auf die Auswertung der Fragen zum Essen konzentriert, da dies der grundlegende Sinn des Chatbots für den Benutzer ist. Eine vollständige Frage des Benutzers wird mit den Informationen zum aktuellen Essen, also Frühstück, Mittagessen oder Abendessen, und einem oder mehreren Nahrungsmitteln erwartet. Der Dialogteil für Essensfragen ist in drei Teile gegliedert, je nachdem wieviel Informationen der Benutzer in seiner Anfrage angibt.
Zuerst werden vollständige Fragen ausgewertet, dafür wird am Anfang auf das Essen und die Nahrungsmittel überprüft. Sind beide in der Anfrage gegeben, so wird daraufhin mit der Kontextvariable der verbleibenden Tage geprüft, siehe Abb. B.3, und anschließend je nach Tag das Nahrungsmittel ausgewertet, siehe Abb. B.4, und eine Antwort gegeben. Sind nun mehrere Nahrungsmittel angegeben, so werden diese nach und nach abgearbeitet wie zuvor beschrieben und ggf. mehrere Antworten geliefert.
Als nächsten Teil wird überprüft, ob der Benutzer lediglich das Essen angegeben hat. In diesem Fall wird der Benutzer gefragt, was er als Nahrungsmittel essen möchte. Nach einer Eingabe springt der Knoten dann in den jeweiligen Knoten im ersten Teil, da nun alle nötigen Informationen vorhanden sind und wertet diese wie im ersten Teil beschrieben aus.
Der dritte Teil ist für eine Eingabe von nur Nahrungsmitteln, in diesem Teil ist wieder keine weitere Eingabe des Benutzers nötig, hier wird je nach der Kontextvariable für die Tageszeit entschieden, um welches Essen es sich handelt. Wie im zweiten Teil wird nun mit allen Informationen wieder in den ersten Teil gesprungen, in welchem alle Antworten für die Nahrungsmittel in der jeweiligen Phase hinterlegt sind.

7.4 Anforderungsabgleich

Im folgenden Kapitel werden die vor der Implementierung erstellten Anforderungen betrachtet und entschieden, ob diese umgesetzt wurden.

7.4.1 Funktionale Anforderungen an die Anwendung

FA-1	Anmeldung ☑
Beschreibung	Um die Anwendung für mehrere Personen nutzbar zu machen, muss ein Benutzer sich am System mit einem eigenen Account anmelden können.

FA-2	Anzeige von verbleibender Zeit ☑
Beschreibung	Eine der wichtigsten Informationen für die Vorbereitung, ist die verbleibende Zeit, da sich danach die Diät richtet. Dafür soll dem Benutzer direkt auf der Startseite angezeigt werden, wieviel Zeit noch bis zur Untersuchung verbleibend ist in der Form TT/SS.

FA-3	FAQ ☑
Beschreibung	Der Benutzer soll nicht nur über die Benachrichtigungen informiert werden, es soll in einer Übersicht dargestellt werden, auf was in jeder Phase geachtet wird und wie die Medikamente eingenommen werden soll.

FA-4	Chat mit Watson ☑
Beschreibung	Damit auf den Chatbot Watson zugegriffen werden kann, soll eine kleine Chatanwendung erstellt werden, in der mit Watson eine Unterhaltung geführt wird.

FA-5	E-Mail an Doktor ☑
Beschreibung	Da Watson nicht alle Fragen abdecken kann bzw. eventuell Fragen auch nicht versteht, soll es dem Benutzer möglich sein, wichtige Fragen direkt an den Arzt per E-Mail zu senden.

FA-6	Registrierung: Auswahl Medikament ☑
Beschreibung	Bei der Registrierung kann der Benutzer selbst auswählen, welche Medikament er als Abführmittel wählt.

FA-7	Registrierung: Angabe von Untersuchungstermin ☑
Beschreibung	Damit die Anwendung weiß, wann welche Benachrichtigungen erforderlich sind, muss der Benutzer den genauen Termin mit Uhrzeit angeben.

FA-8	Registrierung: Anmeldeinformationen und Doktor E-Mail ☑
Beschreibung	In der Registrierung werden die Informationen wie E-Mail und Passwort des Benutzers abgefragt für die Anmeldung. Zudem wird die E-Mail des Doktors abgefragt, um diesem später eine E-Mail schreiben zu können.

Benachrichtigungen

FA-9	Ernährungstipps ☑
Beschreibung	Der Benutzer erhält täglich eine bestimmte Anzahl an Benachrichtigungen, welche ihm helfen, eine optimale Vorbereitung zu vollziehen.
Bemerkung	Der Benutzer erhält genau drei Benachrichtigungen am Tag, diese sind für die drei Mahlzeiten am Tag vorgesehen, siehe 7.2.1.

FA-10	Weiterleitung auf FAQ-Seite ☑
Beschreibung	Um alle wichtigen Informationen selbst nachschauen zu können, soll jede Benachrichtigung die Möglichkeit haben, auf die FAQ-Seite weiterzuleiten, sodass der Benutzer selbst alle Informationen überschauen kann.

FA-11	Medikamentenerinnerung ☑
Beschreibung	Neben der Ernährungsumstellung bekommt der Benutzer auch Benachrichtigungen, wann und wie er das Medikament einnehmen muss, um eine optimale Vorbereitung zu erlangen.

FA-12	Frage von Watson an Arzt weiterleiten ☑
Beschreibung	Kann eine Frage nicht von Watson beantwortet werden, so soll er mit einem Klick auf die dann erscheinende Benachrichtigung direkt zur MailActivity kommen um eine E-Mail an die hinterlegte Adresse des Arztes zu senden.

7.4.2 Funktionale Anforderungen an Watson

FA-12	Antwortvariationen ☑
Beschreibung	Bei gleichen Fragen soll es möglich sein, verschiedene Antworten zu liefern, um so eine natürlichere Unterhaltung zu führen.
Bemerkung	Es sind hauptsächlich Variationen von Antworten vorhanden, diese müssen aber immer das Nahrungsmittel enthalten, da mehrere Antworten zurück gegeben werden können.

FA-13	Antwort passend zur Anfrage ☑
Beschreibung	Watson soll dem Benutzer eine Antwort liefern, welche auf die Frage bzw. den Typ und die Eigenschaften der Anfrage eingeht. Somit wird eine natürlichere Unterhaltung ermöglicht.
Bemerkung	Wie in FA-12 beschrieben ist diese Anforderung nun für jede Antwort vorgeschrieben.

FA-14	Unterscheidung zwischen Phasen ☑
Beschreibung	Im Dialog mit Watson wird die aktuelle Phase, jedes einzelnen Benutzers in der Wahl der Antwort berücksichtigt.

FA-15	Unterscheidung von Tageszeiten ☑
Beschreibung	Je nach Tageszeit soll automatisch die richtige Antwort gewählt werden, falls der Benutzer keine Mahlzeit angibt.

FA-16	Allgemeine Fragen ☑
Beschreibung	Neben den Fragen zu der Ernährung, soll Watson auch einige allgemeine Fragen zu der Untersuchung beantworten können.
Bemerkung	Es werden ein paar Fragen rund im die Untersuchung beantwortet, allerdings in keinem großen Umfang an Fragen.

8 Fazit und Ausblick

8.1 Fazit

Nach einiger Einarbeitungszeit in den Service von IBM Watson wurden die Möglichkeiten und Begrenzungen schnell ersichtlich. Durch die leicht verständliche und intuitiv zu bedienende Weboberfläche, lassen sich schnell Fortschritte beim Bauen eines Chatbots sehen. Die Dokumentation auf der IBM Homepage selbst ist umfangreich und gibt einen guten Überblick, zur Servicenutzung und den Bedienungsmöglichkeiten. Die Intents und Entitäten waren in kurzer Zeit erstellt.
Allerdings ist hierbei ernüchternd, dass keine Entitäten in anderen Entitäten als Definition vorhanden sein können, um somit Gruppen zu bilden und um diese einfacher zu Filtern. Nach dem Erstellen der Grundlagen in Form von Intents und Entitäten wurde der Hauptbestandteil im Dialogmodul gebildet. Hier zeigt der Service nun seine Stärken, wie ein mögliches Zwischenspeichern von jeglichen Werten und einer beliebigen Kombination von Intents und Entitäten, wodurch sich komplexe Anfragen genau voneinander trennen lassen. Mithilfe der Auswertung von Knoten, mehreren Antworten und weiter verfahren je nach Wert werden viele Möglichkeiten für eine Dialogstruktur geboten. Durch diese Vielzahl an Auswertungen mehrerer Entitäten in einer Anfrage zu übernehmen wird der ganze Dialog auch schnell unübersichtlich.
Beim Testen des Dialoges kam dann auch Ernüchterung auf. Werden Entitäten nicht genau genannt, so werden diese nicht erkannt ohne Fuzzy Matching. Auch werden unbekannte Wörter keinen Entitäten zugeordnet, welche diesen in ihrer Bedeutung ähneln oder sogar Synonyme sind. So sind eine Vielzahl an Beispielen gefragt, welche möglichst alle Benutzereingaben widerspiegeln. Dies war besonders im Hinblick auf die Entwicklungsgeschichte von Watson überraschend, da der Service so starre Vorgaben hat und lediglich nach diesen vorgeht. Die Erwartungen an Watson konnten somit nicht komplett erfüllt werden, zumindest nicht für den Gebrauch im Rahmen dieser Arbeit.

Abschließend ist zu sagen, dass sich der Service auch hauptsächlich an Unternehmen richtet, welche die Interaktionen mit Benutzern über das Internet oder am Telefon für ihren Kundenservice automatisieren möchten. Für diesen Einsatzzweck ist der Service von Watson durch die Vorgaben von Benutzerantworten und durch kontinuierliches Anpassen mithilfe der geführten Gesprächen sicherlich ein gutes Endprodukt. Für den Einsatzzweck als ein Chatbot, welcher mit dem Benutzer eine Unterhaltung führt und Antworten selbst in natürlicher Sprache wiedergeben sollte allerdings zu komplex, dies zu realisieren.

8.2 Ausblick

Für eine weitere Verbesserung von Watson, gäbe es die Möglichkeit, eine Studie mit vielen Benutzern durchzuführen. Dabei muss darauf geachtet werden, welche Teile der Anfragen Watson richtig erkennt und welche Fragen allgemein von den Probanden gestellt werden. Somit könnte man den aktuellen Service mithilfe echter Testeingaben verbessern und mehrere Arten von Fragen beantworten, welche bisher nicht beantwortet werden können.
Eine Verbesserungsmöglichkeit für die Anwendung wäre ein Server-Client Modell. Damit könnte die App auf mehreren Geräten von der gleichen Person genutzt werden, ohne die App bei jedem Gerät einzeln zu konfigurieren.
Nicht jede Person kann die Vorbereitung gleichermaßen vollziehen, z.B. bei Diabetespatienten müssen diese ggfs. etwas am Tag der Untersuchung zu sich nehmen. Auch Dauermedikationen von Patienten, welche diese nicht absetzten können wird aktuell nicht berücksichtigt, diese könnten Nebenwirkungen mit dem Abführmittel haben und müssen in Absprache mit dem Arzt abgesetzt oder verringert werden. Hierbei wäre mithilfe einer Server-Client Architektur die Möglichkeit gegeben, dass der Zuständige Arzt über eine eigene App die Vorbereitungspläne seiner Patienten anpassen kann.

A Quelltexte

Quelltext zur Berechnung der Zeiten von Benachrichtigungen.

```
1
2  /**@param con
3  *
4  * The method gets the information from the database,
5  * calculates the random times for the Notifications
6  * and saves them in the database
7  */
8  public void setup(Context con) {
9  System.out.println("IN SETUP METHODE IN ALARM");
10 int day = cal.get(Calendar.DAY_OF_MONTH);
11 Random random = new Random();
12
13 //Get Settings from Sharedpreferences
14 SharedPreferences sp =
15   con.getSharedPreferences(namespace,con.MODE_PRIVATE);
16 int startHour = sp.getInt("start_time_hour",7);
17 int startMinute = sp.getInt("start_time_minute", 00);
18 int endHour = sp.getInt("end_time_hour",22);
19 int endMinute = sp.getInt("end_time_minute",00);
20 //Setup the Calendar for the Milliseconds
21 Calendar tmp_start = Calendar.getInstance();
22 Calendar tmp_end = Calendar.getInstance();
23 tmp_start.set(cal.get(Calendar.YEAR),
24   cal.get(Calendar.MONTH),cal.get(Calendar.DAY_OF_MONTH),
25   startHour,startMinute);
26 long startMillis = tmp_start.getTimeInMillis();
27 tmp_end.set(cal.get(Calendar.YEAR),
28 cal.get(Calendar.MONTH),cal.get(Calendar.DAY_OF_MONTH),
29 endHour,endMinute);
```

```
30 long endMillis = tmp_end.getTimeInMillis();
31 long timespan = endMillis - startMillis;
32 //Get equal timespaces, so that 1 Notification lays
33 //between 1/3 of the timespace +- random 1/6 timespace
34 int plus_millis = (int) timespan/6;
35 int minus_millis = -plus_millis;
36 Calendar tmp_cal = Calendar.getInstance();
37 //Initialize boolean
38 boolean bool_one = false;
39 boolean bool_two = false;
40 boolean bool_three = false;
41
42 //Generate random times
43 tmp_cal.setTimeInMillis(startMillis+plus_millis);
44 long millis_one = tmp_cal.getTimeInMillis() +
45   random.nextInt((plus_millis - minus_millis) + 1)
46   + minus_millis;
47 long millis_mid_one = tmp_cal.getTimeInMillis();
48
49 tmp_cal.setTimeInMillis(startMillis+3*plus_millis);
50 long millis_two = tmp_cal.getTimeInMillis() +
51   random.nextInt((plus_millis - minus_millis) + 1)
52   + minus_millis;
53 long millis_mid_two = tmp_cal.getTimeInMillis();
54
55 tmp_cal.setTimeInMillis(startMillis+5*plus_millis);
56 long millis_three = tmp_cal.getTimeInMillis() +
57   random.nextInt((plus_millis - minus_millis) + 1)
58   + minus_millis;
59 long millis_mid_three = tmp_cal.getTimeInMillis();
60
61 //Debugging
62 printTime(millis_one);
63 printTime(millis_two);
64 printTime(millis_three);
65
66 //Check if time for Notification is already over
67 if(cal.getTimeInMillis() > millis_mid_one+plus_millis){
```

```
68 bool_one = true;
69 }
70 if(cal.getTimeInMillis() > millis_mid_two+plus_millis){
71 bool_two = true;
72 }
73 if(cal.getTimeInMillis() > millis_mid_three+plus_millis){
74 bool_three = true;
75 }
76
77 String tipone = "";
78 String tiptwo = "";
79 String tipthree = "";
80 //Get the Tips from the JSON-File
81 ArrayList<Phase> phases = json_model.getPhases();
82 for(Phase p : phases){
83 if(days < p.getStart() && days >= p.getEnd()){
84 String[] tmp = p.getTip();
85 tipone = tmp[0];
86 tiptwo = tmp[1];
87 tipthree = tmp[2];
88 }
89 }
90 //Safe the generated values in the DB
91 sd.setNotications(millis_one,millis_two,millis_three,
92   bool_one,bool_two,bool_three,tipone,tiptwo,
93   tipthree,day);
94 }
```

B Abbildungen

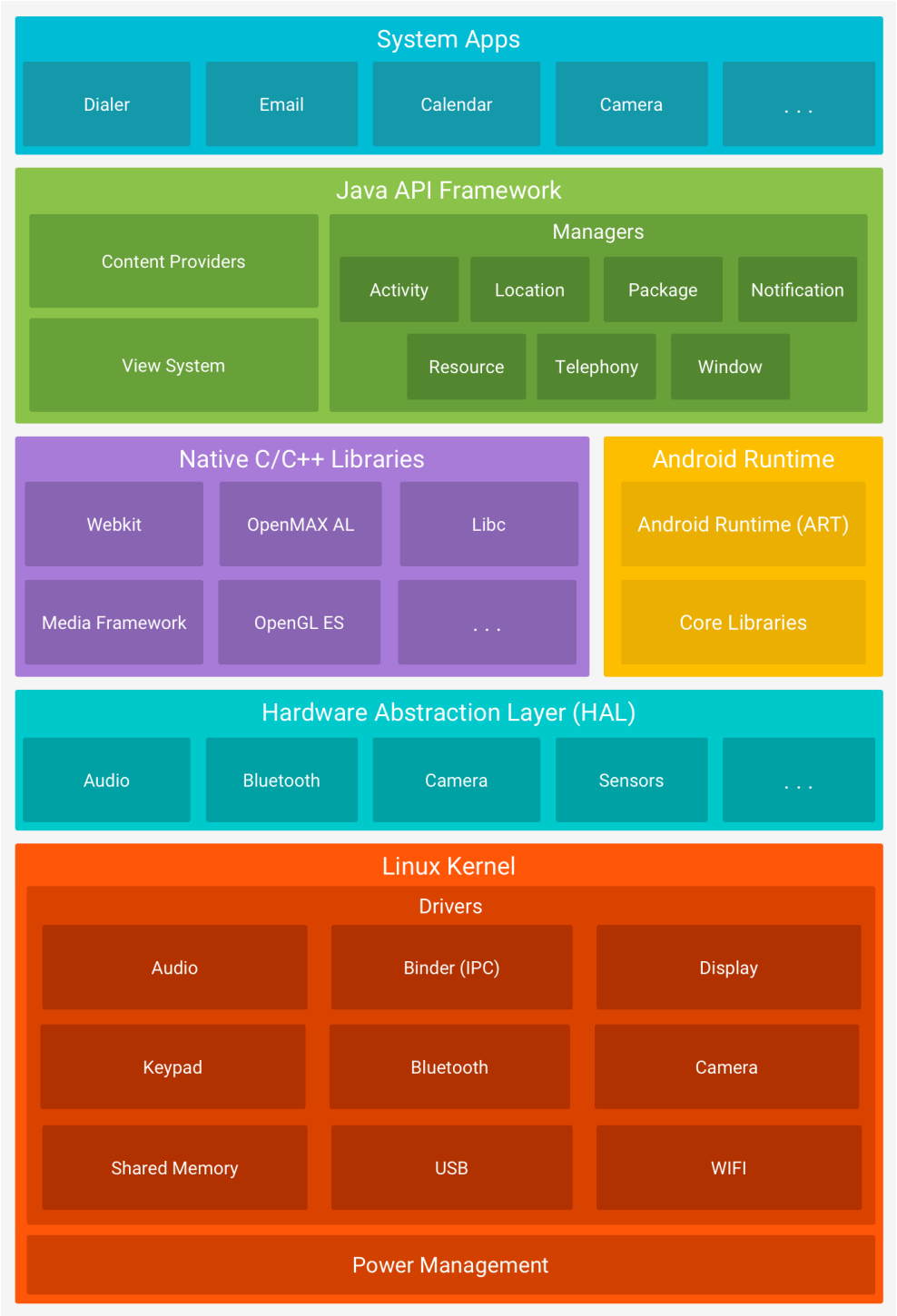

Abbildung B.1: Übersicht über die Architektur von Android

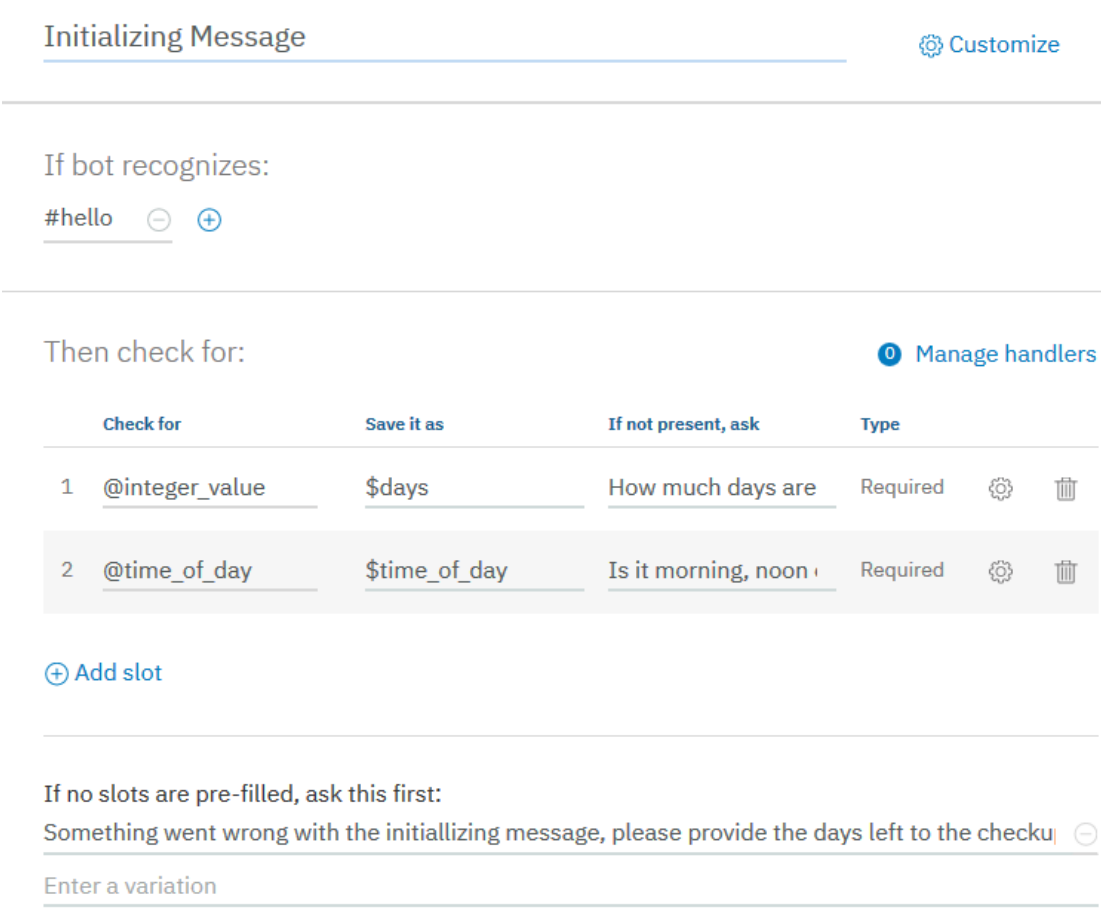

Abbildung B.2: Slotfeature eines Knotens

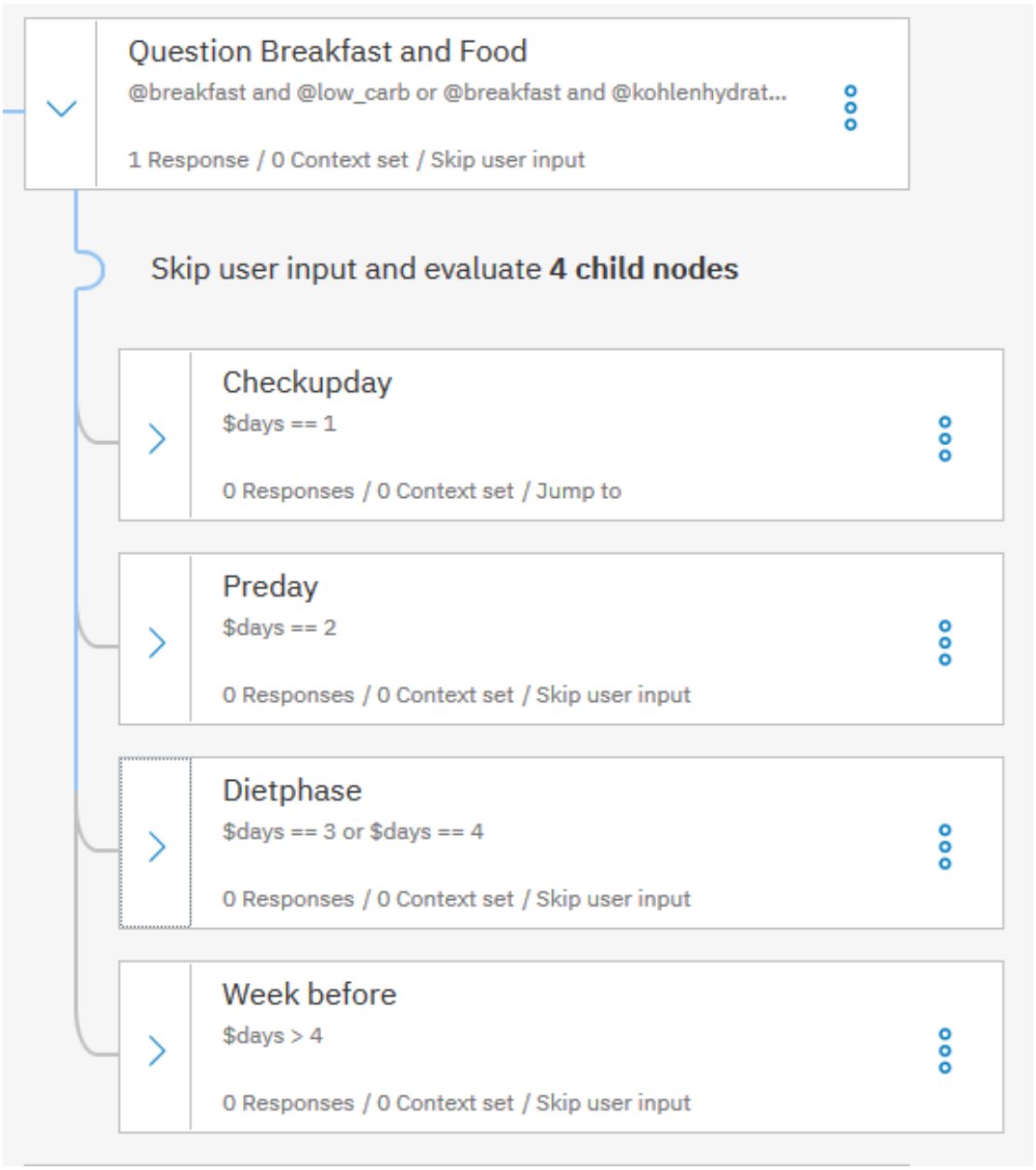

Abbildung B.3: Auswahl der Phase im Dialog

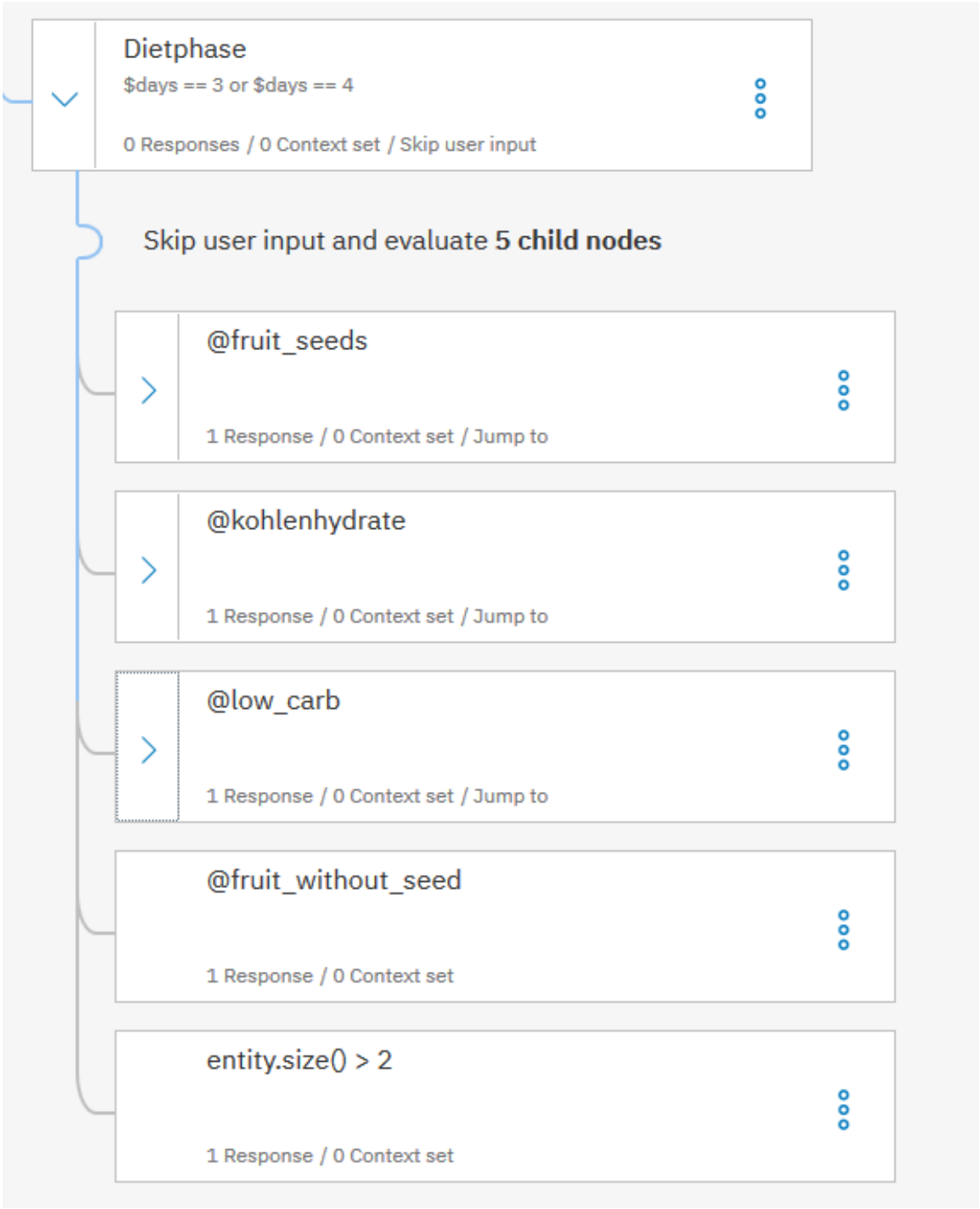

Abbildung B.4: Abarbeitung des Nahrungsmittels

Literaturverzeichnis

[1] *Android Developers*. http://developer.android.com, . – Stand: 07.07.2018

[2] *Android Source*. http://source.android.com, . – Stand: 10.06.2018

[3] *API Documentation Watson*. https://www.ibm.com/watson/developercloud/conversation, . – Stand: 26.06.2018

[4] *Darmspiegelung: Gründe, Vorteile, Risiken*. https://www.apotheken-umschau.de/diagnose/darmspiegelung, . – Stand: 10.07.2018

[5] *Deutsches Krebsforschungszentrum*. https://www.krebsinformationsdienst.de/tumorarten/darmkrebs, . – Stand: 11.07.2018

[6] *Dokumentation Watson*. https://console.bluemix.net/docs/services/conversation, . – Stand: 07.07.2018

[7] *Material Design Google*. http:/metrial.io, . – Stand: 11.07.2018

[8] *Open Handset Alliance*. https://www.openhandsetalliance.com/, . – Stand: 05.06.2018

[9] *Picoprep*. http://www.picoprep.de/, . – Stand: 10.07.2018

[10] *Research Team IBM Watson*. https://researcher.watson.ibm.com/researcher/view_group.php?id=2099, . – Stand: 07.07.2018

[11] *Vergleich der Abführmittel*. http://darmlifestyle.de/ced-und-co/darmspiegelung-abfuehrmittel/, . – Stand: 03.04.2018

[12] FERRUCCI, D. ; LEVAS, A. ; BAGCHI, S. ; GONDEK, D. ; MUELLER, E. T.: Watson: beyond jeopardy! In: *Artificial Intelligence* 199 (2013), S. 93–105

[13] HIGH, R. : The era of cognitive systems: An inside look at IBM Watson and how it works. In: *IBM Corporation, Redbooks* (2012)

[14] LALLY, A. ; FODOR, P. : Natural language processing with prolog in the ibm watson system. In: *The Association for Logic Programming (ALP) Newsletter* (2011)

[15] MAGAZINE, A. : The ai behind watson—the technical article. In: *AI Magazine* (2010)

[16] MARKOFF, J. : Computer wins on 'jeopardy!': trivial, it's not. In: *New York Times* 16 (2011)

[17] PARRA-BLANCO, A. ; NICOLÁS-PÉREZ, D. ; GIMENO-GARCÍA, A. ; GROSSO, B. ; JIMÉNEZ, A. ; ORTEGA, J. ; QUINTERO, E. : The timing of bowel preparation before colonoscopy determines the quality of cleansing, and is a significant factor contributing to the detection of flat lesions: a randomized study. In: *World journal of gastroenterology: WJG* 12 (2006), Nr. 38, S. 6161

[18] SIEVERDING, M. : Männer und Inanspruchnahme von Krebsfrüherkennungsuntersuchungen. (2011)

[19] STARKER, A. ; BUTTMANN-SCHWEIGER, N. ; KRAYWINKEL, K. ; KUHNERT, R. : Inanspruchnahme der Darmspiegelung in Deutschland. Version: 2017. http://dx.doi.org/10.17886/RKI-GBE-2017-115. In: *Journal of Health Monitoring*. Robert Koch-Institut, Epidemiologie und Gesundheitsberichterstattung, 2017. – DOI 10.17886/RKI–GBE–2017–115

[20] VAN RIJN, J. C. ; REITSMA, J. B. ; STOKER, J. ; BOSSUYT, P. M. ; VAN DEVENTER, S. J. ; DEKKER, E. : Polyp miss rate determined by tandem colonoscopy: a systematic review. In: *The American journal of gastroenterology* 101 (2006), Nr. 2, S. 343

Name: Bastian Wankmüller Matrikelnummer: 879014

Erklärung

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Bastian Wankmüller