ulm university universität
uulm

Universität Ulm | 89069 Ulm | Deutschland

Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Datenbanken und Informationssysteme

Design and Implementation of Just-In-Time Adaptive Interventions in an eHealth Platform

Masterarbeit an der Universität Ulm
Angestrebter Abschluss: Master of Science (MSc) in Computer Science

Vorgelegt von:
Johannes Ziegler
johannes-1.ziegler@uni-ulm.de
Matrikelnummer: 932195

Prüfer:
Prof. Dr. Manfred Reichert, Universität Ulm
Prof. Dr. Rüdiger Pryss, Julius-Maximilians-Universität Würzburg

Betreuer:
Robin Kraft, Universität Ulm

2021

Version 13. Dezember 2021

Zusammenfassung

Mit zunehmender Bedeutung von Internet und Smartphones entwickelten sich in der Psychotherapie neue, vielversprechende Behandlungsansätze, die Probleme der klassischen Face-2-Face Therapie lösen. Durch Internet- and Mobile-based Interventions (IMIs) können kosteneffektive, flächendeckende und anonyme therapeutische Behandlungen über das Internet bereitgestellt werden. Eine fortgeschrittene Art von IMIs sind Just-In-Time Adaptive Interventions (JITAIs). JITAIs nutzen Sensoren in Smartphones, um die Umgebung und das Verhalten der Patient:innen im Alltag zu analysieren. Aus diesen Informationen ermitteln sie den Kontext und stellen ihnen Interventionen zum richtigen Zeitpunkt und mit den richtigen Inhalten zur Verfügung. Folglich personalisieren JITAIs die Behandlung, unterstützen Patient:innen im Alltag und erkennen kritische Situationen.

Im Fokus dieser Masterarbeit steht die konzeptuelle und technische Umsetzung von JITAIs in E-Health Plattformen. Es wird eine mehrstufige Architektur vorgestellt, die verschiedene Datenströme modular erhebt und analysiert. Hierbei handelt es sich um Daten von Smartphone-Sensoren, Ecological Momentary Assessments und externen Geräten sowie um In-App Metadaten. Der Datenfluss kann einen Trigger auslösen, der Patient:innen eine Intervention vorschlägt. Es werden Visualisierungstechniken dieser Datenströme dargestellt sowie ein kontextsensitives Empfehlungssystem, das zusätzlich Feedback und Metadaten von Patient:innen berücksichtigt. Darüber hinaus zeigt die Arbeit auf, wie Interventionen und Inhalte abhängig von abgegebenen Antworten und persönlichen Informationen adaptiert werden können. Der praktische Teil erweitert die E-Health Plattform der Abteilung für Klinische Psychologie und Psychotherapie der Universität Ulm um die Adaptierung von Inhalten, sodass Patient:innen eine individuellere Behandlung erhalten.

Die vorliegende Arbeit schafft Grundlagen für die Einführung von JITAIs in E-Health Plattformen. Insbesondere kann die erarbeitete Architektur zukünftig als Ausgangspunkt dienen, um die Datenströme mithilfe von künstlicher Intelligenz zu analysieren.

Inhaltsverzeichnis

1 Einführung

Weltweit leiden rund 13 Prozent der Bevölkerung an einer psychischen Krankheit oder einer Substanzkonsumstörung. Zu diesem Ergebnis kam eine Studie der World Health Organisation (WHO) aus dem Jahr 2017 [1]. Der Anteil an Menschen die unter Depression oder einer Angststörung, leiden wurde dabei auf über 7 Prozent geschätzt. Dementsprechend hoch ist der Bedarf an Therapieplätzen. Zahlen belegen jedoch, dass die eigentlichen Behandlungsraten gering sind. Einer Studie aus dem Jahr 2007 zufolge erhielten nur zwischen 10 und 24 Prozent der Bedürftigen in Ländern mit geringem oder mittleren Einkommen therapeutische Hilfe. In mitteleuropäischen Ländern lag dieser Wert bei rund einem Drittel, wobei Deutschland und Frankreich mit jeweils 42 Prozent die Spitze bildeten [2].
Die niedrigen Behandlungsraten sind auf verschiedene Gründe zurückzuführen. Einer Studie von 2013 zufolge ist die persönliche Einstellung der Betroffenen gegenüber Psychotherapie der größte Faktor. Jedoch gaben auch 23 Prozent der Befragten an, dass unter anderem strukturelle Barrieren wie Finanzierung und Versorgungsengpässe sie von einer Behandlung abhielten [3].

1.1 Motivation

Mit Internet- and Mobile-based Interventions (IMIs) hat sich ein digitaler Ansatz für psychotherapeutische Behandlungen etabliert, dessen Wirksamkeit und Kosteneffektivität bereits mehrfach nachgewiesen wurde [4–7]. IMIs ermöglichen es, Angebote unabhängig von Zeit und Ort über Smartphones oder Computer in Anspruch zu nehmen und wirken dadurch dem Problem von Versorgungsengpässen und langen Wartezeiten entgegen. Die Einbindung von Smartphones in Behandlungen bie-

tet darüber hinaus noch weiteres Potential hinsichtlich Personalisierung. So lassen sich über Smartphones wichtige Faktoren wie Stimmung, Stress, Angst, Schlaf oder soziale und physische Aktivitäten von Patient:innen[1] im Alltag verfolgen und vorhersagen [8, 9]. Just-In-Time Adaptive Interventions (JITAIs) sind Interventionen, die durch Nutzung dieser Informationen im richtigen Moment und mit den richtigen Inhalten zur Verfügung gestellt werden. Sie können dadurch vor kritischen Situationen bewahren bzw. Hilfestellung geben. Die höhere Wirksamkeit von JITAI-Behandlungen macht sie im Vergleich zu Nicht-JITAI-Behandlungen attraktiv [10].

1.2 Problemstellung

Die Abteilung für Klinische Psychologie und Psychotherapie (KLIPS) der Universität Ulm möchte ihre E-Health Plattform eSano für Forschungszwecken um JITAIs erweitern. Hierfür fehlte es bisher an Ansätzen zur Umsetzung von JITAIs. Aus dem Begriff geht bereits hervor, dass Inhalte *just-in-time* und *adaptive* bereitgestellt werden, die genauen Anforderungen für dieses Feature sind jedoch unklar. Bisher gibt es auf der Plattform noch keine Möglichkeit, Daten zu erheben oder Analysen mit künstlicher Intelligenz durchzuführen. Auch die Adaptierung von Inhalten ist bisher nur bedingt möglich. Zwar besitzt eSano ein Feature, das Inhalte während der Bearbeitung von Lektionen auf Basis der Antworten von Patienten adaptiert. Dies ist jedoch nicht ausreichend, um eine personalisierte Behandlung zu bieten.
Zur Lösung dieses Problems sollen in der vorliegenden Masterarbeit die folgenden Fragen beantwortet werden:

1. Wie sieht der Funktionsumfang zur Umsetzung von JITAIs in E-Health Plattformen aus?
    a) Welche Funktionen sind notwendig, um Interventionen zu adaptieren?
    b) Welche Funktionen sind notwendig, um Interventionen *just-in-time* bereitzustellen?

[1] Zur besseren Lesbarkeit wird in der vorliegenden Arbeit auf die gleichzeitige Verwendung männlicher und weiblicher Sprachformen verzichtet. Im Folgenden wird das generische Maskulinum verwendet, womit beide Geschlechter gleichermaßen gemeint sind.

   c) Welche weiteren Funktionen sind notwendig, um Patienten eine personalisierte Behandlung zu bieten?

2. Wie können diese Funktionen technisch umgesetzt werden? Welche Techniken gibt es und was sind ihre Vor- und Nachteile?
3. Wie ist eine Architektur zur Umsetzung von JITAIs aufgebaut?
   a) Welche Datenströme gibt es und wie werden diese verarbeitet?
   b) Wie tauschen Client und Server Daten aus?

1.3 Struktur der Masterarbeit

Die Masterarbeit ist wie folgt gegliedert: Kapitel 2 schafft die Grundlagen, die für das Verständnis der weiteren Kapitel notwendig sind. Es erläutert wichtige Begriffe sowie den Kontext, in dem diese Arbeit entsteht. Kapitel 3 definiert JITAIs und leitet daraus Eigenschaften ab. Weiterhin werden verwandte Arbeiten vorgestellt, die sich mit JITAIs oder deren Teilaspekten beschäftigen. In Kapitel 4 werden funktionale und nicht-funktionale Anforderungen an JITAIs erarbeitet und in Kapitel 5 ein Konzept vorgestellt, das diese Anforderungen erfüllt. In Kapitel 6 wird ein Teil der Anforderungen praktisch umgesetzt. Kapitel 7 beendet die Thesis mit der Zusammenfassung der Ergebnisse und einem Ausblick über weitere Fragestellungen und Herausforderungen.

2 Grundlagen

Dieses Kapitel schafft die Grundlagen, die zum Verstehen der Masterarbeit notwendig sind. Es werden wichtige Begriffe erläutert sowie der Kontext, in dem die Masterarbeit entsteht.

2.1 Internet- and Mobil-based Interventionen

Eine Intervention ist eine psychotherapeutische Behandlung, die darauf abzielt, das Wohlbefinden eines Patienten zu steigern und die Art und Weise zu verbessern, wie ein Patient denkt und sich verhält [11]. IMIs werden von einem Patienten über einen Browser oder eine App abgerufen und hauptsächlich selbstständig bearbeitet [12, 13]. In Blended-Therapy-Ansätzen werden sie mit der klassischen Face-2-Face-Therapie kombiniert [12]. Das Anwendungsgebiet von IMIs reicht von Prävention bis hin zu Rehabilitation [12]. Die Inhalte von IMIs lehren dem Patienten meistens Wissen und Achtsamkeit über seine geistige Gesundheit und erzeugen dadurch einen positiven Effekt und Unterstützung [13]. Da IMIs webbasiert sind, beinhalten sie neben Texten oft Elemente wie Audios, Videos und Fragen. Der Zugriff auf IMIs erfolgt über einen passwortgeschützten, persönlichen Zugang, wobei dieser oft durch Features wie Chats, Erinnerungen, Foren und Spiele ergänzt ist [12]. IMIs sind in der Regel modular aufgebaut und bestehen aus einer Menge an Lektionen. Die Abarbeitung erfolgt konsekutiv (z. B. jede Woche ein Modul) oder im Sinne eines Baukastensystems, bei dem Patienten aus einer vordefinierten Menge an Lektionen auswählen [12].
Ein entscheidender Faktor über den Behandlungserfolg durch IMIs stellt die professionelle Unterstützung über einen E-Coach dar (Guidance). Die Rolle des E-Coach ist es, dem Patienten Rückmeldung über bearbeitete Interventionen zu geben und

ihn zu motivieren. Die Unterstützung durch einen E-Coach hat in der Forschung eine verminderte Abbruchrate sowie eine verstärkte Symptomreduktion ergeben [14]. Die Vorteile von IMIs gegenüber einer Face-2-Face-Therapie sind einfache Zugänglichkeit, bessere Skalierbarkeit, Anonymität, kürzere Wartezeit und Kosteneffektivität [15–17].

2.2 Kontext und Kontextsensitivität

Im Bereich Mobile Sensing und Ubiquitious Computing wird als Kontext jede Information bezeichnet, die zur Charakterisierung der Situation einer Entität verwendet werden kann. Dabei kann eine Entität eine Person, ein Ort oder ein physisches oder rechnendes Objekt sein [18]. Der Kontext umfasst die Umstände, die derzeitige Situation, die Bedingungen, die Umgebung oder die Umwelt einer Entität [19]. Unter Kontextsensitivität oder kontextsensitivem Rechnen wird die Berücksichtigung von Kontexten bei der Bereitstellung von Services für Nutzer verstanden [18].

2.3 Ecological Momentary Assessment

Die Behandlung eines Patienten in der klinischen Psychologie basiert in der Regel auf retrospektiven Selbstberichten von Patienten, die bei Forschungs- oder Klinikbesuchen erhoben werden [20]. Diese sind durch Erinnerungsverzerrungen eingeschränkt und eignen sich nicht gut zur Untersuchung der Veränderung des Verhaltens über einen längeren Zeitraum oder in einem bestimmten Kontext [20]. Unter einem Ecological Momentary Assessment (EMA) versteht man regelmäßige Stichproben des aktuellen Verhaltens und Erlebens eines Patienten in Echtzeit, in seiner natürlichen Umgebung [20]. Sie zielen darauf ab, Erinnerungsverzerrungen zu minimieren, die ökologische Validität zu maximieren und die Untersuchung von Mikroprozessen zu ermöglichen, die das Verhalten im realen Kontext beeinflussen [20]. Die Durchführung erfordert oft Hilfe von Technologien, die von schriftlichen Tagebüchern und Telefongesprächen bis zu elektronischen Tagebüchern und physiologischen Sensoren reichen [20].

2.4 Kontext

2.4.1 eSano

Die Masterarbeit entsteht im Rahmen des Projekts eSano[1], einer E-Health Plattform der KLIPS der Universität Ulm [21]. Es werden Wirksamkeit, Kosteneffektivität, Akzeptanz und Inanspruchnahme sowie Wirkungsfaktoren von Interventionen untersucht [22]. Im Fokus stehen begleitete und unbegleitete Interventionen, welche von Patienten selbstständig bearbeitet werden. Außerdem werden neue Therapieansätze entwickelt und erforscht, die künstliche Intelligenz nutzen.
eSano veröffentlicht in Studien Interventionen, die wiederum aus mehreren Lektionen bestehen [21]. Eine Lektion beinhaltet Hypermedia Elemente wie Texte, Bilder, Videos, Audios und Downloads sowie verschiedene Arten von Fragen. Die Architektur von eSano geht aus Abbildung 2.1 hervor. Eine Intervention wird im Content Management System (CMS) von geschulten Editoren erstellt und für eine Studie veröffentlicht. Nach der Veröffentlichung könne E-Coaches der Studie auf die Interventionen zugreifen und sie Patienten zuweisen. Patienten rufen die zugewiesenen Studien mittels einer (Web-)App ab und bearbeiten sie. Der E-Coach kann nach der Bearbeitung die Antworten der Patienten einsehen und ihnen Feedback geben. Haben Patienten alle Lektionen einer Intervention bearbeitet, so ist die Intervention abgeschlossen. Derzeit laufen verschiedene Studien bei eSano. Die Studie iCHIMPS beispielsweise steht für *Children of Mentally Ill Parents* und unterstützt Kinder und Jugendliche im Umgang mit psychisch oder suchtkranken Elternteilen. Die Studie ACTonDiabetes beschäftigt sich mit diabetesbezogenen Belastungen im Alltag und hilft Patienten dabei, ihr Wohlbefinden zu verbessern.

2.4.2 BetterCare

BetterCare ist eine Studie von eSano, die Anfang 2022 startet und Patienten in der Zeit nach ihrer Krebsdiagnose hilft. Die entwickelte Intervention heißt ebenfalls BetterCare und besitzt verschiedene Arten von Lektionen sowie ein Tagebuch (siehe Abbildung 2.2).

[1] www.esano.klips-ulm.de

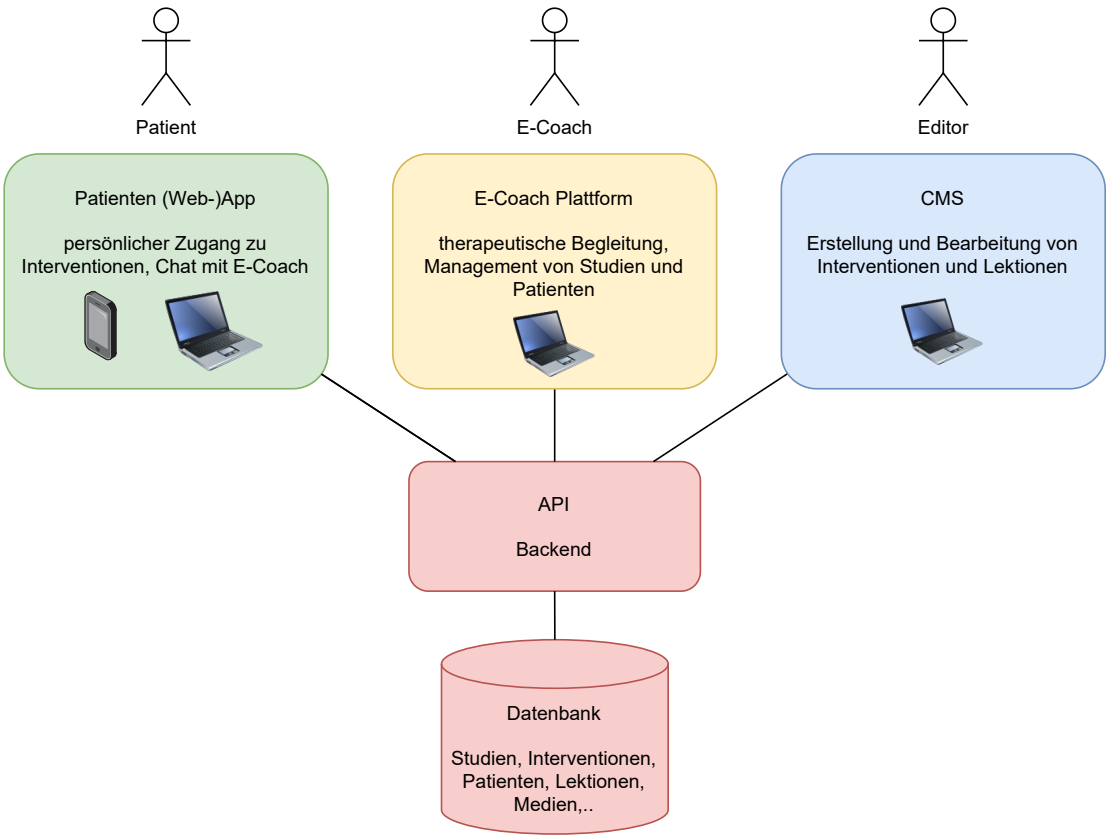


Abbildung 2.1: Architektur von eSano

- Grundkurs: Automatisch freigeschaltet für alle Teilnehmer. Informationen über Inhalte und Struktur. Wie funktioniert Therapie übers Internet?
- ACT: Drei nicht-krebsspezifische Basislektionen, die der Patient in beliebiger Reihenfolge bearbeitet.
- ACTonCancer: Drei bis vier krebsspezifische Lektionen, die ein Patient in beliebiger Reihenfolge bearbeitet.
- Befindlichkeitstagebuch: Tagebuch, das der Patient täglich bearbeitet. Beinhaltet Fragen zu Stimmung (5-stufig Likert), Schlaf (5-stufig Likert), sozialer Aktivität (Multiple-Choice) und körperlicher Aktivität (Angabe der Intensität als Zahl).
- Bonusmaterial: Zehn Lektionen, die in Abhängigkeit des Tagebuchs freigeschaltet werden.

Die Freischaltung des Bonusmaterials kann in eSano bisher nur manuell über den E-Coach erfolgen. Im praktischen Teil der Masterarbeit wird die Plattform erweitert, sodass die Bonuslektionen in Abhängigkeit der gegebenen Antworten aus den Tagebüchern automatisch freigeschaltet werden und die Intervention sich dadurch selbst an die Bedürfnisse des Patienten anpasst.

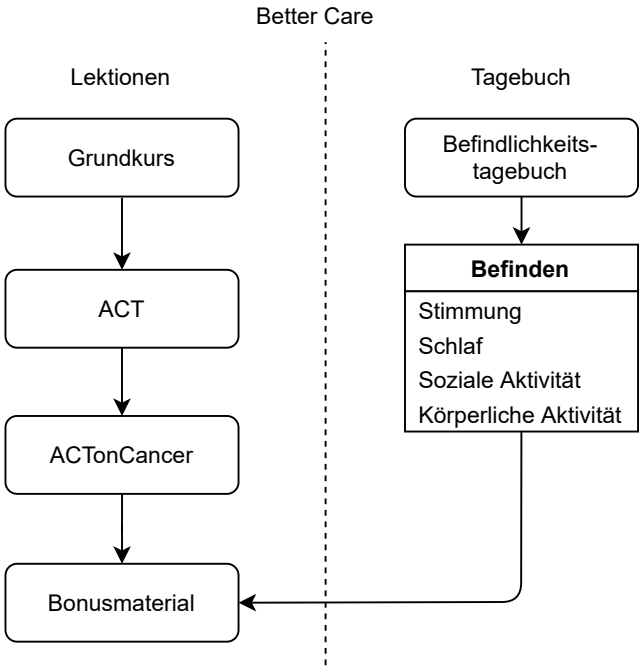


Abbildung 2.2: Interventionsaufbau von Better Care

3 Just-In-Time Adaptive Interventions

Dieses Kapitel führt den Begriff Just-In-Time Adaptive Interventions ein. Hierfür werden Definitionen aus der Literatur betrachtet und daraus Eigenschaften abgeleitet. Darüber hinaus werden verwandte Arbeiten vorgestellt, die sich mit JITAIs befassen.

3.1 Definition

Nahum-Shani et al. definieren Just-In-Time Adaptive Interventions als Interventionen, die so konzipiert sind, dass sie dynamisch auf die Bedürfnisse des Nutzers eingehen, indem sie die richtige Art oder Menge der Unterstützung zum richtigen Zeitpunkt bereitstellen [23]. Daten werden in Echtzeit unter anderem über Mobile Sensing erhoben und verwendet, um Rückschlüsse auf den emotionalen, sozialen, physischen und kontextuellen Zustands einer Person zu ziehen. Dadurch soll die Art, die Durchführung und der Zeitpunkt von Interventionen zu individualisiert werden [23]. JITAIs können so konzipiert sein, dass sie negative gesundheitliche Folgen verhindern, die Annahme gesunder Verhaltensweisen fördern und/oder einer Person helfen, gesunde Verhaltensweisen beizubehalten [23].
Eine andere Definition stammt von Naughton. Dieser versteht unter JITAIs unterstützende Interventionen, die über den Server, den Nutzer oder den Kontext getriggert werden. [24].

3.2 Eigenschaften von JITAIs

Aus der Definition von Nahum-Shani et al. werden für die Masterarbeit drei Eigenschaften abgeleitet, die eine E-Health Plattform mit JITAI-Unterstützung benötigt:

1. *Just-In-Time* Eigenschaft: Interventionen werden einem Patienten zum richtigen Zeitpunkt angeboten. Sie helfen Patienten zum Schutz oder zur Bewältigung von kritischen Situationen (Trigger) oder zur Etablierung neuer Verhaltensweisen.
2. *Adaptiveness* Eigenschaft: Interventionen und deren Inhalte passen sich an die Bedürfnisse und Situation des Patienten an. Inhalte werden außerdem personalisiert. In kritischen Situationen beinhalten Interventionen passende Gegenmaßnahmen.
3. *Adherence* Eigenschaft: Die Platform berücksichtigt Nutzerverhalten und Nutzerfeedback, um die *Just-In-Time* und *Adaptiveness* Eigenschaften pro Nutzer zu verbessern.

3.3 Verwandte Arbeiten

Im Folgenden werden Arbeiten vorgestellt, die sich mit JITAIs beschäftigen.

3.3.1 Forschung

JITAIs wurden bereits auf viele Aspekte untersucht. Insbesondere existieren zahlreiche Studien, die sich mit der Effektivität von JITAIs beschäftigen. Den Ergebnissen einer Metaanalyse mit 33 empirischen Studien zufolge, waren Behandlungen mit JITAIs wirksamer als Behandlungen ohne JITAIs, wobei moderate bis hohe Effekte beobachtet wurden [10]. Die Effekte wurden in den Bereichen mentale Gesundheit, Suchtbehandlung (Alkohol, Zigratten, Drogen), gesunde Lebensweise und Therapieadhärenz bei chronischen Krankheiten festgestellt. JITAIs eignen sich insbesondere für die Rückfallprevention beispielsweise bei Suchtbehandlungen oder Diäten.

Rückfälle treten nicht zufällig auf, sondern lassen sich unter Berücksichtigung interner und externer Risikofaktoren vorhersagen [25]. Mithilfe von Mobile Sensing werden diese Risikofaktoren in JITAIs beobachtet.
Ein Ansatz zur Entwicklung von JITAIs ist die Anwendung von Verhaltenstheorie, kombiniert mit experimentellen Ansätzen zur Optimierung der Intervention. Ein Beispiel für einen experimentellen Ansatz sind mikro-randomisierte Studien (MRT) [26]. Bei einem MRT werden das serverseitige Timing und die Version von JITAIs, Benachrichtigungen, Motivationsnachrichten, Vorschlägen, Tipps oder Erinnerungen randomisiert und die umittelbaren Effekte der Aktion gemessen. Die gewonnen Daten werden ausgewertet und zur Optimierung verwendet. So beschäftigen sich viele Studien mit der Entwicklung und Anwendung von MRTs [27–30].
Daneben gibt es Studien, die sich ausschließlich mit adaptiven oder *just-in-time* freigeschalteten Interventionen, also Teilaspekten von JITAIs, beschäftigen [31, 32]. Analog zu MRTs wurden beispielsweise zur Optimierung von adaptiven Interventionen die experimentellen Ansätze SMART (Sequential Multiple Assignment Randomized Trial), MOST (Multiphase Optimization Strategy) und Q-Learning entwickelt [32, 33].
Während die kurz- und mittelfristige Effektivität von JITAIs bereits untersucht wurde, gibt es noch keine Studien zu den Langzeiteffekten von JITAIs. Ebenfalls fehlen Untersuchungen zur Kosteneffektivität. Aus technischer Sicht ist das JITAI-Feature mit viel Mehraufwand verbunden, der zusätzliche Entwicklungs- und Wartungskosten verursacht. Diesen gilt es daher zu untersuchen.

3.3.2 Apps

Es existieren zahlreiche kommerzielle Apps im Google und Apple Store unter der Kategorie *Mental Health*. Den Ergebnissen einer Metaanalyse aus dem Jahr 2021 zufolge nutzen nur 5 von 28 untersuchten Apps aus den Stores Sensoren oder andere Geräteinformationen, um Symptome bei Nutzern zu beobachten. Die bekanntesten Apps die davon Gebrauch machen sind Youper[1] und MindDoc[2] (früher Moodpath) mit jeweils einer Million Downloads.

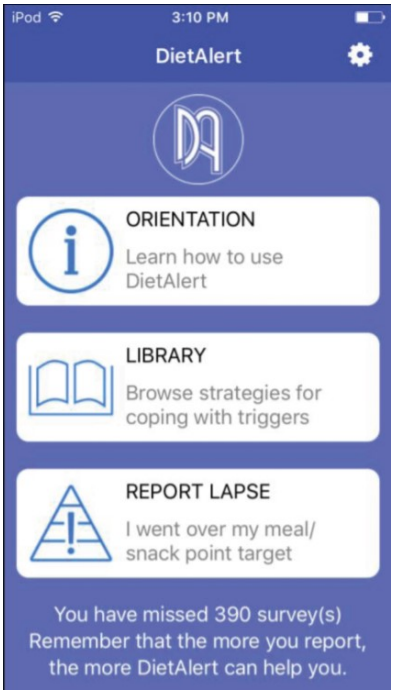
[1] www.youper.ai
[2] www.mymoodpath.com

Ebenfalls gibt es Apps, die gezielt für JITAI-Studien entwickelt worden sind. Beispiele hierfür sind OnTrack und MyQuit[3]. Sie wurden entwickelt um Menschen gezielt bei der Suchtbekämpfung und der Verhinderung von Rückfällen zu unterstützen [34, 35]. Sie sind daher ähnlich aufgebaut und besitzen beide Funktionen zur Selbstauskunft und zur Durchführung von EMAs (siehe Abbildung 3.1a und Abbildung 3.1d). OnTrack beispielsweise überwacht ca. 20 Risikofaktoren mithilfe von Ja-Nein-, Single-Choice-, Likert- und Slider-Fragen. Auf Basis der Daten wird ein Machine Learning Algorithmus ausgeführt, der das Rückfallrisiko in Echtzeit berechnet. Bei Bedarf wird der Nutzer alarmiert (siehe Abbildung 3.1c) und wählt aus einer Menge an Gegenmaßnahmen aus (siehe Abbildung 3.1b).

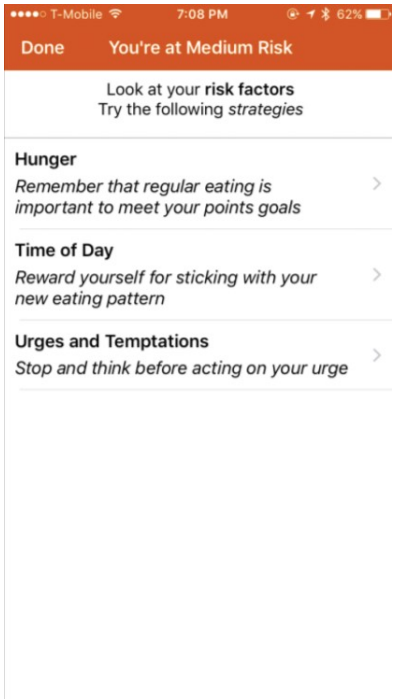
3.3.3 E-Health Plattformen

Es existieren viele E-Health Plattformen, die Dienste für Psychotherapeuten, Ärzte, Coaches oder Patienten anbieten. Im Gegensatz zu Apps sind die Plattformen weniger auf Selbsthilfe ausgelegt, sondern auf die Durchführung von IMIs mit einem E-Coach. Entsprechend gibt es auf den Plattformen Zugänge für Psychotherapeuten, um IMIs zu Verwalten und den Patienten zuzuteilen. Sie bieten Psychotherapeuten auch die Möglichkeit, IMIs selbst zu erstellen - unter anderem auch *adaptive* Inhalte, die für die Masterarbeit relevant sind. Beispiele für kommerzielle E-Health Plattformen sind Minddistrict[4], SilverCloud[5] oder Online-Therapy[6]. Außerdem existieren Plattformen wie Moodbuster[7] und Mindspot Clinic[8], die Dienste in einem staatlich geförderten Forschungskontext zur Verfügung stellen.
Minddistrict wird von der KLIPS bereits verwendet und daher hinsichtlich der *Adaptiveness* näher betrachtet. Minddistrict stellt zur Erzeugung von Inhalten einen WYSIWYG-Editor bereit, über den sich Texte, Fragen, Audios, Videos und Tabellen einfügen lassen (siehe Abbildung 3.2a) [36]. Über eine *valueOf* Funktion kann im Editor auf eine Frage referenziert werden, deren Antwort zur Laufzeit an der

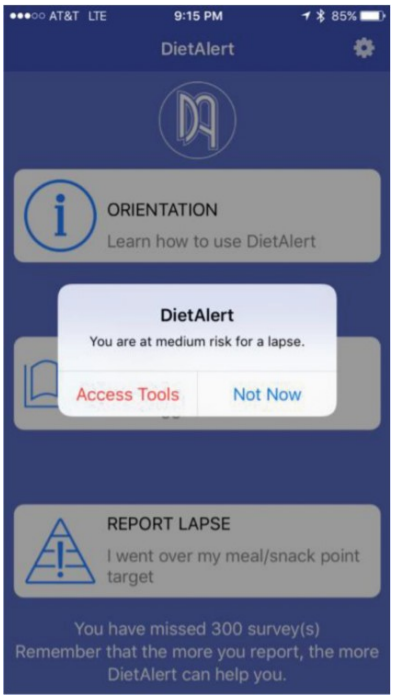
[3] www.myquit.ca
[4] www.minddistrict.com
[5] www.silvercloudhealth.com
[6] www.online-therapy.com
[7] www.moodbuster.science
[8] www.mindspot.org.au



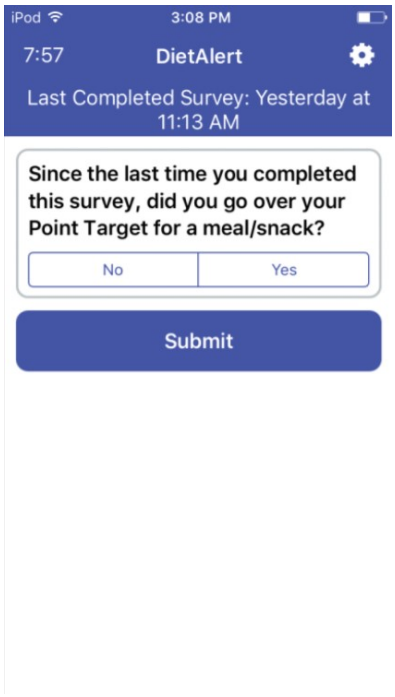
(a) Hauptmenü



(b) Auswahl von Gegenmaßnahmen



(c) Hinweis bei erhöhtem Rückfallrisiko



(d) EMA

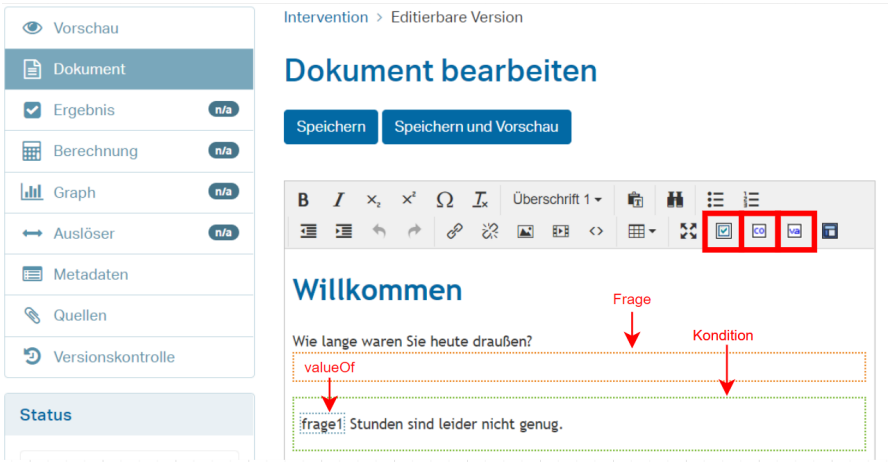
Abbildung 3.1: OnTrack App [34]

entsprechenden Stelle eingefügt wird. Inhalte lassen sich durch die Definition von Konditionen zur Laufzeit adaptieren. Dafür steht ein Konditionseditor zur Verfügung (siehe Abbildung 3.2b), in dem die Zielfrage, ein Vergleichsoperator, eine Konstante und ein Konstantentyp angegeben werden muss. Außerdem lassen sich über eine JSON-Eingabe auch komplexere Konditionen definieren, bei denen Konditionen über boolesche Operationen verknüpft werden. Über eine Vorschau lässt sich Testen, ob die Konditionen korrekt definiert wurden (siehe Abbildung 3.2c). Im Beispiel wird ein Patient gefragt, wie lange er heute an der frischen Luft war. Insofern er eine Stunde oder weniger an der frischen Luft war, erscheint die Meldung *X Stunden sind leider nicht genug*, wobei X über die *valueOf* Funktion den Wert der Antwort von *frage1* annimmt.

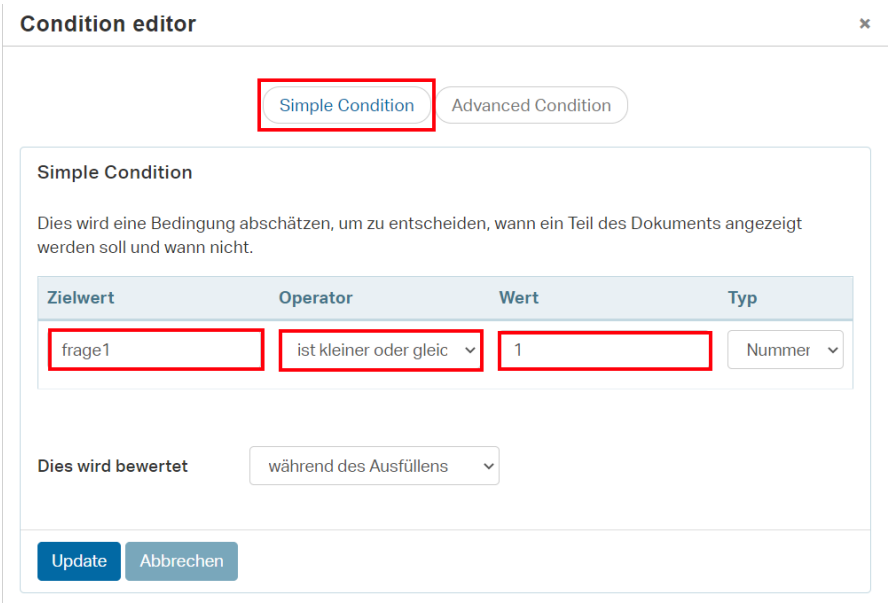
3.3.4 Open-Source Mobile-Sensing Frameworks

Es existieren einige Open-Source Mobile-Sensing Frameworks, die zur einfachen Integration von Mobile Sensing in Apps entwickelt wurden. Sie sind für die Masterarbeit interessant, da sie Einblicke in Architektur, Schnittstellen und Erweiterbarkeit geben. Beispiele hierfür sind AWARE[9] Radare-Base[10] und SensingKit[11]. Aufgrund der Durchführung von mehreren Studien mit AWARE, wird dieses Framework im Folgenden näher betrachtet [37, 38].
Mit AWARE lassen sich Sensordaten und Kontexte loggen, instrumentieren, teilen und wiederverwenden [39]. Das Framework besteht aus einem Server und aus einer App für Android und iOS. Im Hauptmenü der App werden alle Hardware- und Softwaresensoren angezeigt und über ein Untermenü konfiguriert (siehe Abbildung 3.3a). In einem weiteren Menü stehen analog Plugins zur Verfügung, die Kontexte aus den Sensordaten ermitteln. Im Stream-Menü werden die erhobenen Daten von Plugins angezeigt (siehe Abbildung 3.3b). Im Screenshot ist beispielsweise der Kontext `Noisey` für das Plugin `Ambient Noise` angezeigt und die derzeitige Lautstärke in Dezibel. Die App besitzt einen integrierten MQTT Client zum Datenaustausch über Peer-2-Peer und eine Websocket Schnittstelle, zum Strea-

[9] www.awareframework.com
[10] www.radar-base.org
[11] www.sensingkit.org



(a) Inhaltseditor



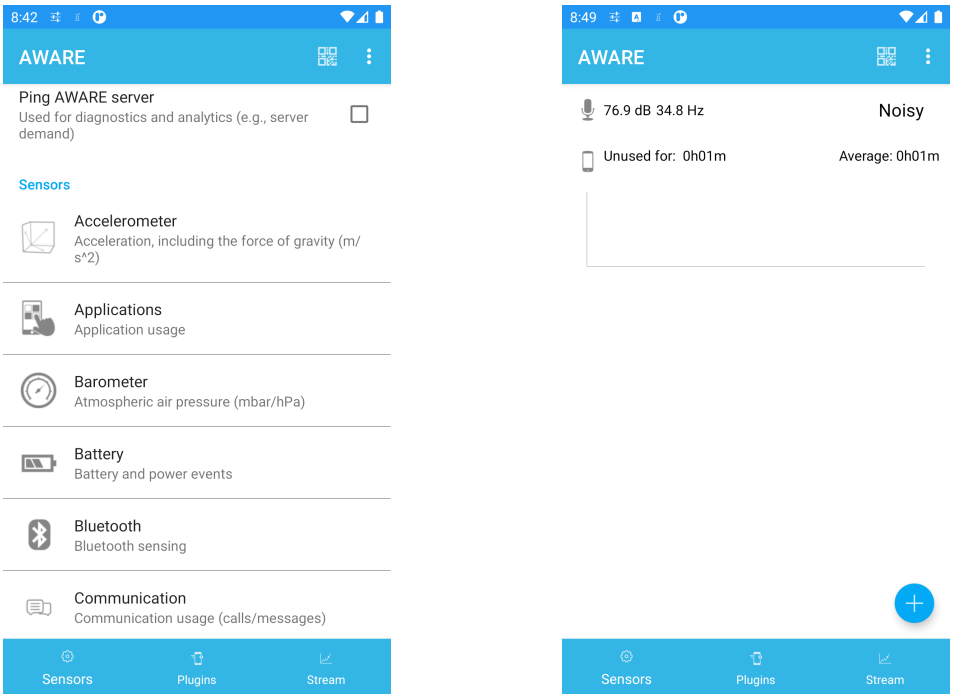
(b) Konditionseditor

(c) Vorschau

Abbildung 3.2: Minddistrict [36]

men der Daten in Echtzeit an den Server.

Serverseitig hat ein Researcher über ein Portal Zugriff auf laufende Studien, eingeschriebene Nutzer und deren Sensor- und Kontextdaten (siehe Abbildung 3.4). Er kann über die GUI Sensoren und Plugins auf Client-Geräten instrumentieren sowie Broadcast Nachrichten an alle Client-Geräte senden. Mit dem Plugin *ESM-Scheduler* werden Fragebögen in Echtzeit an Client-Geräte gesendet. Für die Fragebögen lassen sich verschiedene Fragetypen auswählen und zeitliche Begrenzungen setzen. Researcher können verschiedene Typen von Diagrammen über eine GUI erstellen (Column, Pie, Scatter, Line, Box, Histogramm) und diese zur Visualisierung der Daten durch Angabe eines Zeitraums wiederverwenden. Es wurde untersucht, ob sich das Framework eignet, um JITAIs in eSano zu realisieren. Dabei wurde festgestellt, dass die App und der Server seit mehreren Jahren nicht mehr entwickelt und gewartet werden [40, 41]. Tests mit der aktuellen Android API 30 ergaben, dass sich die App ohne Probleme bauen und installieren lässt, bei der Benutzung jedoch nicht alle Sensoren und Plugins korrekt funktionieren. Dies betrifft unter anderem die Funktion zum Senden und Empfangen von EMAs. Beim Aufsetzen des AWARE Servers gab es kleinere Probleme, die auf fehlende Dokumentation und veraltete Bibliotheken zurückzuführen sind. Es wurde herausgefunden, dass der Server mit dem PHP-Framework CodeIgniter programmiert wurde. Gemäß der CodeIgniter Dokumentation wurde eine Datenbank für den Server aufgesetzt und konfiguriert. Außerdem musste OAuth2.0 in *application/auth/config.php* konfiguriert sowie PHP wegen der veralteten Bibliothek `mcrypt` auf Version 7.0 downgegraded werden. Bei der Nutzung gab es kleinere Probleme mit der GUI.



(a) Aktivierung von Sensoren

(b) Verfolgen von Datenströmen

Abbildung 3.3: AWARE App [40]

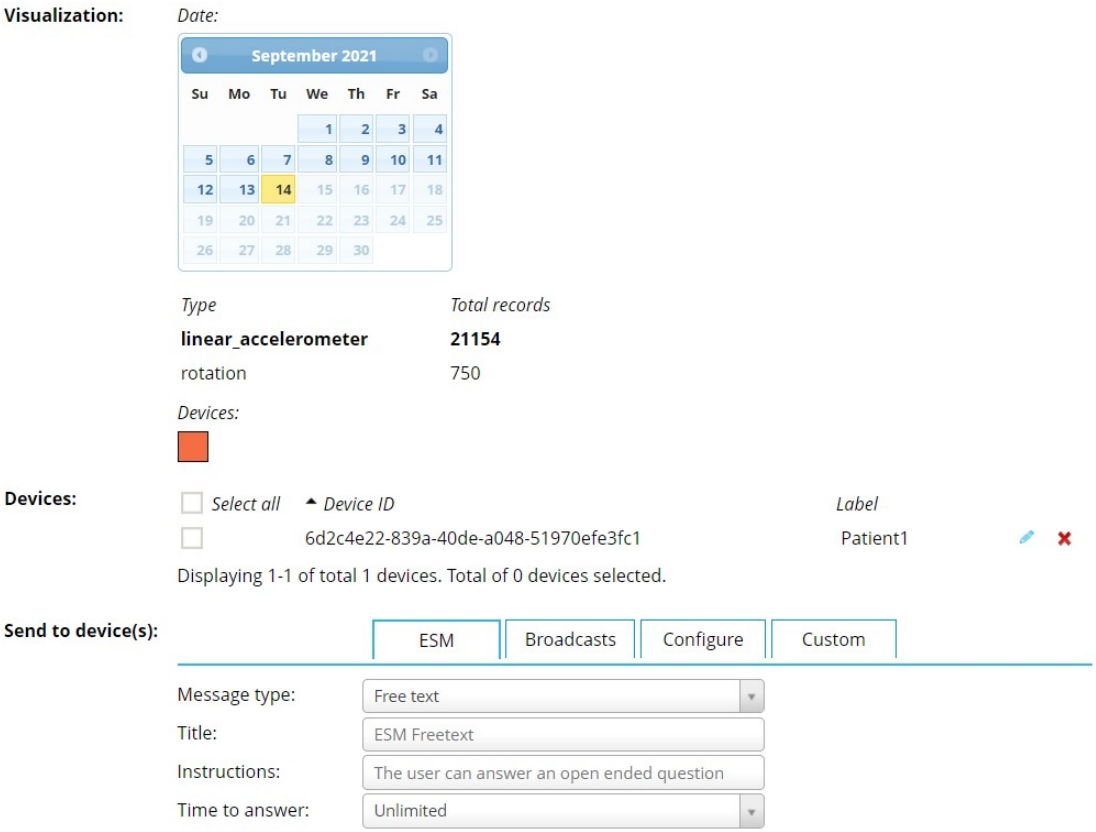


Abbildung 3.4: AWARE Server [41]

4 Anforderungen

Dieses Kapitel stellt die Anforderungen an JITAIs zusammen. Sie sind gekennzeichnet mit einer ID der Form `XX-Y`. `XX` untergliedert die Anforderung in Kategorien. `Y` identifiziert eine Anforderung innerhalb einer Kategorie. Jede Anforderung besitzt eine Priorität und ist der *Just-In-Time* (JIT), *Adaptiveness* (ADA) oder *Adhärenz* (ADH) Eigenschaft zugeordnet, wobei die erstgenannte Eigenschaft der primären Zuordnung entspricht.

4.1 Funktionale Anforderungen

Die folgenden Anforderungen sind funktional und damit maßgeblich für die Umsetzung von JITAIs.

Mobile Sensing

MS-01 *Erhebung von Daten*
Es können Daten auf Smartphones von Patienten erhoben werden. Er kann Sensoren über eine GUI aktivieren, deaktivieren und konfigurieren.
JIT
++

MS-02 *Berechnung von Kontexten*
Es können Kontexte auf Basis von gesammelten Daten und anderen Kontexten berechnet werden.
JIT
++

MS-03 *Senden von Interventionen*
Interventionen können automatisiert an einen Patienten gesendet werden, wenn ein bestimmter Kontext eintritt.
JIT
++

MS-04 *Synchronisierung*
Gesammelte Daten und Kontexte auf Smartphones und externen Geräten werden in regelmäßigen Abständen oder bei Bedarf mit dem Server synchronisiert.
JIT
++

MS-05 *Steuerung durch den E-Coach*
Der E-Coach kann Sensoren über das Internet aktivieren, deaktivieren und konfigurieren.
JIT
+

MS-06 *Registrierung externer Geräte*
Es können externe Geräte wie Smartwatches in das System integriert werden.
JIT, ADA
0

EMA

EMA-01 *Durchführung von EMAs*
Der E-Coach kann EMAs mit Patienten durchführen. Sie können in regelmäßigen Abständen oder bei Bedarf durchgeführt werden.
JIT, ADA
++

EMA-02 *EMA-Antworten*
Die Antworten von EMAs stehen dem E-Coach zur Einsicht zur Verfügung. Sie können außerdem zur Berechnung von Kontexten genutzt werden.
JIT, ADA
++

EMA-03 *Durchführung von EMAs durch den Patient*
Der Patient kann bei Bedarf freiwillig EMAs durchführen.
JIT, ADA
+

Adaptierung

AC-01 *Adaptierung von Interventionen*
Lektionen können in Abhängigkeit abgegebener Antworten freigeschaltet werden.
ADA
++

AC-02 *Adaptierung von Lektionen*
Inhalte von Lektionen passen sich auf Basis abgegebener Antworten an. Inhalte können erscheinen/verschwinden sowie *n*-mal wiederholt werden.
ADA
++

AC-03 *Endnutzer-Programmierung*
Die Adaptierungen können von Editoren über eine GUI in die Interventionen und Lektionen programmiert werden.
ADA
++

AC-04 *Personalisierung*
Der Editor kann Stellen in Lektionen angeben, an welchen persönliche Informationen eines Patienten eingetragen werden. Persönliche Informationen werden zur Personalisierung der Anwendungen verwendet.
ADA, ADH
++

Empfehlungssystem

ES-01 *Patient kann Feedback geben*
Der Patient kann Feedback auf Interventionen geben.
ADH
++

ES-02 *Empfehlung von Interventionen*
Auf Basis von Feedback, persönlichen Präferenzen und dem aktuellen Kontext, können dem Patienten Interventionen empfohlen werden.
ADH, JIT, ADA
++

Visualisierung

VI-01 *E-Coach verfügt über Dashboard*
Der E-Coach verfügt über ein Dashboard, in dem Sensing-, Kontext-, EMA- und Metadaten zu einem Patienten angezeigt werden.
JIT, ADA, ADH
++

VI-02 *Patient verfügt über Dashboard*
Der Patient verfügt über ein Dashboard, auf dem er gesammelte Sensing-, Kontext-, EMA- und Metadaten einsehen kann. Er kann die Datenströme in Echtzeit verfolgen.
JIT, ADH
+

Metadaten

ME-01 *Erhebung von Metadaten*
Plattform erhebt Metadaten zu einem Nutzer (beispielsweise bei der Bearbeitung von Interventionen) und stellt diese dem E-Coach oder zur Berechnung von Kontexten bereit.
ADH, JIT, ADA
+

4.2 Nicht-funktionale Anforderungen

Im Folgenden werden die nicht-funktionalen Anforderungen aufgelistet, die es zu beachten gilt.

NF-01 *Gesundheit des Patienten*
In Medizinprodukten hat die Gesundheit der Patienten höchste Priorität. Kritische Daten, die negative Auswirkungen auf den Patienten haben könnten, sind nur für den E-Coach einsehbar. Ebenfalls sollte das System Unterstützung offline bereitstellen können.
++

NF-02 *Privatsphäre des Patienten*
Die Plattform nutzt Mechanismen zum Schutz der Privatsphähre des Patienten. Nur Patienten und deren E-Coaches haben Zugriff auf gesammelte Daten, wobei es *NF-01* zu beachten gilt.
++

NF-03 *Sicherheit des Systems*
Da die Plattform höchst sensible Daten speichert, muss die technische Sicherheit des Systems gewährleistet sein, um Datenleaks oder ähnliches zu verhindern.
++

NF-04 *Erweiterbarkeit*
Das System sollte einfach erweiterbar sein.
++

5 Entwurf

Dieses Kapitel stellt den Entwurf zur Umsetzung von JITAIs entlang der *Just-In-Time*, *Adaptiveness* und *Adherence* Eigenschaften vor. Die funktionalen Anforderungen werden entsprechend ihrer Thematik und primären Zuordnung im zugehörigen Unterkapitel behandelt.

5.1 Just-In-Time

Dieses Unterkapitel stellt einen Entwurf zur Umsetzung der *Just-In-Time* Eigenschaft vor.

5.1.1 Architektur

Damit heterogene und hohe Datenmengen einheitlich verarbeitet werden, ist die Frage nach einer Architektur bei der Umsetzung der *Just-in-Time* Eigenschaft maßgeblich. Als Ausgangspunkt wird die Architektur von Peter Hösch aus seiner Abschlussarbeit mit dem Titel *Entwurf and Implementierung eines Frameworks für Mobile Context-Awareness im Bereich eHealth* am Institut für Datenbanken und Systeme verwendet [42]. Die Architektur besteht aus Sensor- und Auswertungsmodulen, wobei Sensormodule Daten über die System API des Smartphones erheben (siehe Abbildung 5.1). Sie lesen beispielsweise Daten von Hardwaresensoren wie dem Beschleunigungssensor. Es stehen jedoch auch Softwaresensoren zur Verfügung, die z. B. Tastaturanschläge oder die Anzahl an ein- und ausgehenden Nachrichten und Anrufen lesen. iOS stellt bereits einige APIs auf höherem Abstraktionsniveau bereit. Unter anderem existiert eine API, die Tremores erkennt und diese in leicht,

mild, moderat und stark kategorisiert. Ein Überblick über die verfügbaren Sensoren in Android und iOS findet sich in Tabelle A.1 und Tabelle A.2.
Auswertungmodule lesen Daten von Sensormodulen oder anderen Auswertungsmodulen, um den Kontext eines Patienten auf höherem Abstraktionsniveau zu berechnen. Ein Beispiel hierfür ist ein Modul, das die Art der Fortbewegung kennt und dazu GPS- und Beschleunigungsdaten über Sensormodule ausliest. Das Modul stellt den berechneten Kontext wiederum für andere Module zur Verfügung.
Der Datenaustausch ist in der Architektur von Peter Hösch nicht weiter spezifiziert. Bei der Implementierung wird zum Lesen von Daten Polling verwendet [42, 43]. Ohne Beschränkung der Allgemeinheit wird für die Architektur der vorliegenden Masterarbeit asynchrone und eventgetriebene Kommunikation mit der System API und zwischen Modulen angenommen, die Events, Listener und Callbacks nutzt. Dies hat den Vorteil, dass nach der Registrierung eines Listener die Kommunikation ausschließlich unidirektional erfolgt und bei hochfrequentem Lesen kein Overhead durch vielfache Funktionsaufrufe entsteht. In Android erfolgt der Zugriff auf Sensoren in der Regel über das *Android Sensor Framework*, das Teil des *Android SDK* ist. Sensordaten können dort in Form von Events beispielsweise über die Reportmodi `CONTINOUS` und `ON_CHANGE` empfangen werden [44]. Für iOS empfiehlt sich die Nutzung des *Core Motion* Frameworks [45]. Diese Architektur wird im nächsten Schritt um Triggermodule erweitert. Ein Triggermodul überwacht Auswertungs- und Sensormodule und führt beim Eintritt eines Zustands eine Aktion aus. Das wichtigste Triggermodul im Kontext von JITAIs stellt der Freischalttrigger dar, der neue Interventionen für den Patienten freischaltet. Die Intervention wird bei Aktivierung des Triggers vorgeladen. Löst der Trigger aus, so schickt die App dem Patienten eine lokale Push Benachrichtigung, die ihn auf die freigeschaltete Intervention weiterleitet. Durch das Vorladen ist die App unabhängig von einer funktionierenden Internetverbindung zum Zeitpunkt, an dem der Trigger auslöst.
Diese leicht erweiterte Architektur schafft bereits die Grundlage dafür, Interventionen *just-in-time* bereitzustellen. Bisher wurden nur Sensormodule angenommen, die Daten aus Hardware- und Softwaresensoren des Smartphones über die System API lesen. Zur Erfüllung der Anforderungen MS-06, EMA-02 und ME-01 werden drei konkrete Sensormodule in die Architektur eingeführt, die jeweils eine bestimmte Art von Daten bündeln und für Analysemodule zur Verfügung stellen.

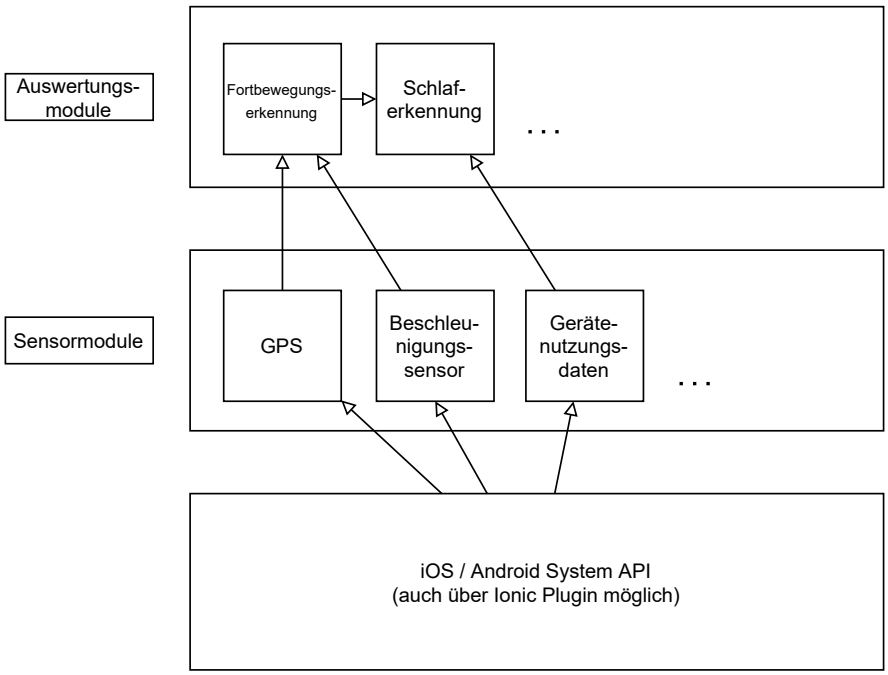


Abbildung 5.1: Ausgangsarchitektur aus der Abschlussarbeit von Peter Hösch (basierend auf [42])

Das erste Modul ist das *EMA-Modul*, das die Antwortbögen aller durchgeführten EMAs eines Patienten bündelt. Das zweite Modul ist das *In-App Metadaten* Modul, das alle Metadaten, die innerhalb der App erhobenen werden, bündelt. Das dritte Modul ist das *Externe Geräte* Modul, das die Daten von externen Gerät bündelt und den Code enthält, der zum Verbindungsaufbau und Lesen notwendig ist. Die resultierende Architektur ist in Abbildung 5.2 ersichtlich. Sie erfüllt die Anforderung, Interventionen *just-in-time* bereitzustellen auf Basis von Hardwaresensoren, Softwaresensoren, EMAs, Metadaten und Daten von externen Geräten.
Damit ein E-Coach die Daten einsehen kann, müssen sie serverseitig zur Verfügung stehen. Ein lokaler Datenbankservice empfängt hierfür die Updates aller Module und speichert diese gemäß einer Datenschnittstelle in einer lokalen Datenbank. Die Datenbank synchronisiert sich alle n Minuten mit der Datenbank des Backends.
Daten von Sensormodule durchlaufen vor Speicherung in der lokalen Datenbank einen *Filter- und Anonymisierungs-Service*. Das Filtern stellt sicher, dass nur Daten, die einen Mehrwert bringen, an den Server übertragen werden und um die Datenbank nicht mit unnötigen Daten zu füllen. Die Anonymisierung erfolgt expli-

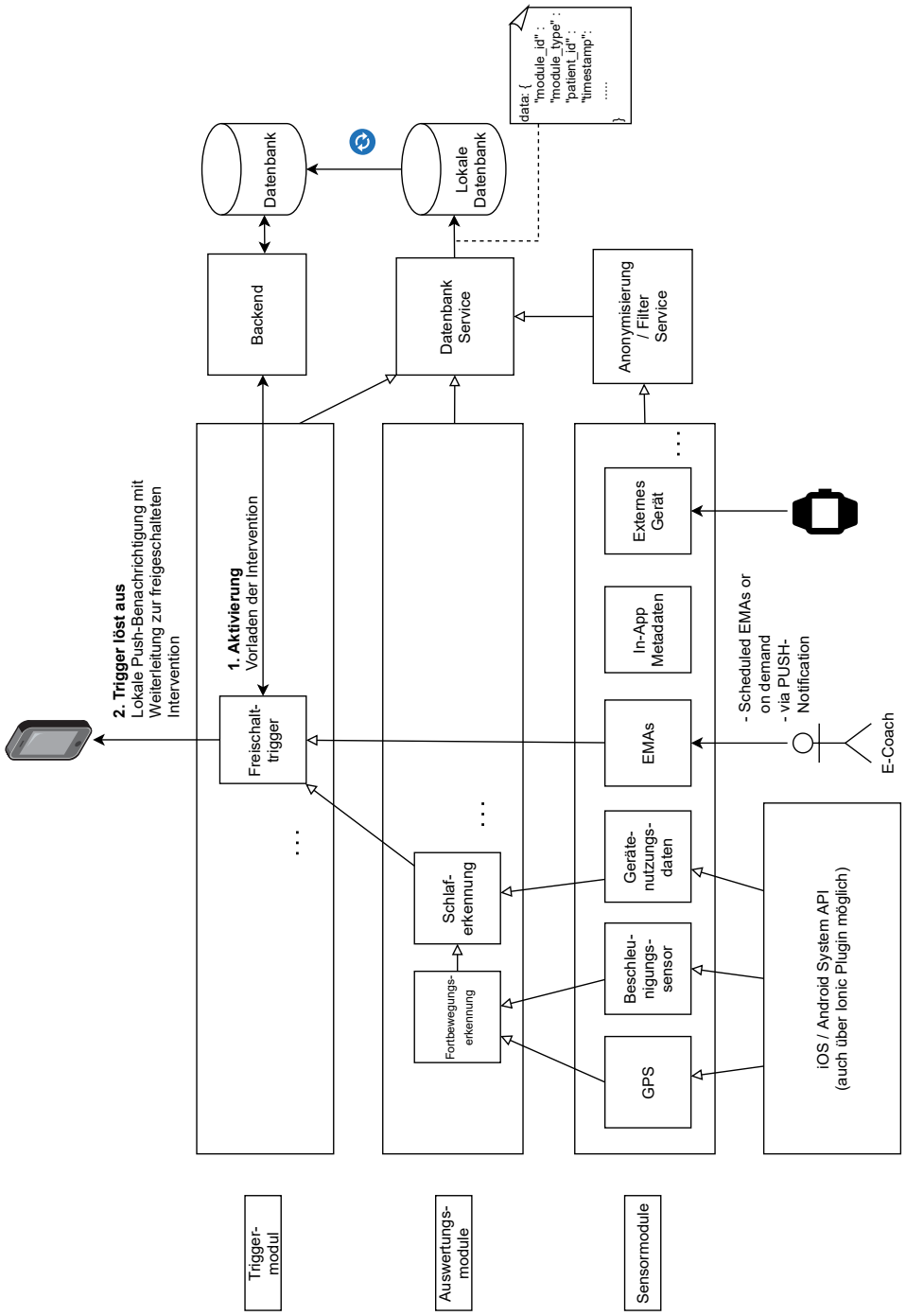


Abbildung 5.2: Architektur zur Umsetzung der *Just-In-Time* Eigenschaft

zit für Module, deren Daten die Privatsphäre eines Patienten gefährden, jedoch serverseitig einen Mehrwert bringen. So kann für GPS-Daten beispielsweise eine Transformationen implementiert werden, die den absoluten Standort eines Patienten verschleiert, bevor die Daten an den Server gesendet werden.

5.1.2 Schnittstellen

Für die Architektur werden Modul-, Daten- und Steuerschnittstellen definiert. Diese werden als UML-Diagramm dargstellt und die Attribute und Funktionen jeweils näher erläutert.

Modulschnittstelle

Die Modulschnittstelle stellt die einheitliche Kommunikation und Integration von Modulen in der App sicher (siehe Abbildung 5.3). Sie umfasst die folgenden Attribute und Funktionen:

- `active`: Interner Zustand des Moduls, der beschreibt, ob Modul aktiviert oder deaktiviert ist. Die Aktivierung erfolgt über `activate(.)` und `deactivate()`
- `config`: Aktuelle Konfiguration des Moduls. Überschreiben einer Konfiguration erfolgt über activate(.).
- `data`: Attribut mit aktuellem Datensatz des Moduls.
- `data_emitter`: Emitter, der ein Event ausstößt, wenn Daten nach `data` geschrieben werden. Andere Module empfangen Updates durch die Registrierung von Listener auf diesem Attribut.
- `dependencies`: Beschreibt Abhängigkeiten zu anderen Modulen über Array an `module_id`. Wird nur implementiert von Analyse- und Triggermodulen. Beispiel: Damit ein Analysemodul *Fortbewegungserkennung* funktioniert, müssen die Sensormodule *GPS* und *Beschleunigungssensor* aktiviert sein.
- `module_id`: Identifiziert Modul systemweit.
- `module_typ`: Jedes Modul ist ein Sensor-, Analyse- oder Triggermodul.

- `module_description`: Beschreibung des Moduls zur Integration in die GUI.
- `config_description`: Beschreibung der Parameter und deren Konfigurationsmöglichkeiten zur Integration in die GUI.
- `data_description`: Beschreibung des Datenformats des Moduls. Kann als Template für Datenbank des Moduls verwendet werden.
- `version`: Aktuelle Version des Moduls.
- `activate(config: Object = default): boolean`: Funktion zur Aktivierung des Moduls. Rückgabewert beschreibt, ob Aktivierung erfolgreich war. Optional mit Angabe einer Konfiguration.
- `deactivate(): boolean`: Funktion zur Deaktivierung des Moduls. Rückgabewert beschreibt, ob Deaktivierung erfolgreich war.
- `is_available()`: Überprüft, ob das Modul auf diesem Gerät verwendet werden kann (siehe bspw. API Levels in Tabelle A.1 und Tabelle A.2).
- `get_data(from, to): ModuleData[]`: Gibt Daten innerhalb eines Zeitraums zurück.
- `visualize(from, to): Diagramm`: Gibt ein Diagramm für den angegebenen Zeitraum zurück. Bei der Umsetzung über WebViews wird der Funktion ein `targetHTML` Element als Parameter übergeben.

Wie in Unterabschnitt 5.1.1 erläutert, erfolgt der Datenaustausch innerhalb der Architektur über Events. Ist dies bei einer System API nicht möglich, so kann ein Sensormodul zum Lesen der Daten Polling verwenden und die Daten trotzdem über `data` für registrierte Listener bereitstellen. Dadurch wird das Lesen per Polling nicht an andere Module weitergegeben. Analog kann ein Sensormodul implementiert werden, das Updates kontinuierlich über die System API empfängt, selbst jedoch nur bei Veränderung der Daten ein Event ausstößt. Die Unterstützung mehrerer Modi ist ebenfalls möglich.

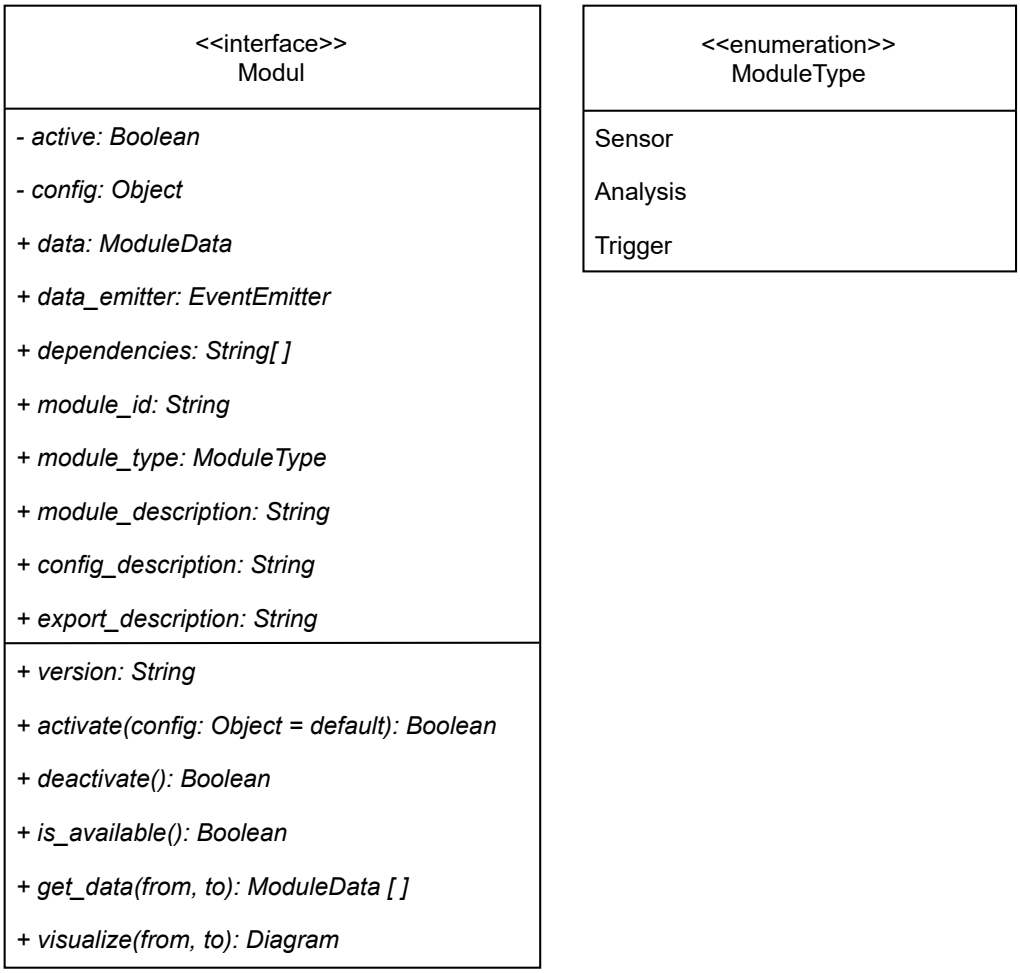


Abbildung 5.3: Modulschnittstelle

Datenschnittstelle

Die Datenschnittstelle stellt die einheitliche Speicherung und Verarbeitung heterogener Daten sicher (siehe Abbildung 5.4). Sie umfasst die folgenden Attribute:

- `module_id`: Identifiziert, von welchem Modul der Datensatz stammt.
- `module_type`: Identifiziert, von welchem Modultyp der Datensatz stammt.
- `parent_id`: Identifiziert, von welchem Patienten der Datensatz stammt.
- `timestamp`: Zeitpunkt, zu welchem der Datensatz erzeugt wurde.

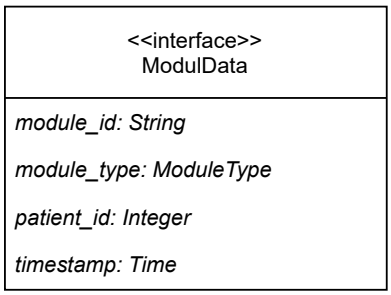


Abbildung 5.4: Datenschnittstelle

Steuerschnittstelle

Die Steuerschnittstelle stellt die einheitliche Ansteuerung der Module über Steuernachrichten sicher (siehe Abbildung 5.5). Sie ermöglicht die Implementierung eines Handlers der Steuernachrichten aus der E-Coach Plattform verarbeitet. Hierfür bietet sich die Nutzung von Silent Push Benachrichtigungen an. Die Steuerschnittstelle umfasst die folgenden Attribute:

- `module_id`: Identifiziert, welches Modul angesteuert wird.
- `action`: Identifiziert, welche Funktion auf dem angesteuerten Modul ausgeführt wird (siehe activate(.) und deactiavte() in `modulschnittstelle`).
- `configuration`: Daten zur Konfigurierung des angesteuerten Moduls.

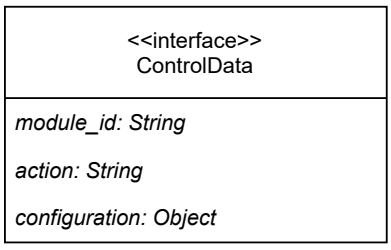


Abbildung 5.5: Steuerschnittstelle

5.1.3 EMAs und das EMA-Modul

Zur standardisierten Verarbeitung von Antwortbögen werden EMAs als eigene Entität gespeichert, die eine Menge an Fragen beinhalten (siehe Listing 5.1). Antwort-

bögen speichern die `ema_id` als Fremdschlüssel und referenzieren damit auf das zugehörige EMA.
Das EMA-Modul empfängt Steuernachrichten des E-Coaches, die Anweisungen zur Durchführung oder Planung von EMAs enthalten. Entsprechend der Anweisungen nutzt das Modul Timer oder lokale Push Benachrichtigungen, um die Anweisungen umzusetzen. Nach der Bearbeitung eines EMAs wird der Antwortbogen in das `data` Attribut des Moduls gespeichert, wodurch registrierte Listener das Update erhalten. Durch die Standardisierung der EMAs über eigene Entitäten stellen Listener über `ema_id` fest, ob das Update für sie relevant ist oder nicht.

```
// EMA
{
  "ema_id": 1
  "questions"{
     {"question_id": 1, "type": "likert-5", "label": "Wie
         fuehlst du dich heute?"},
     {"question_id": 2, "type": "yes/no", "label": "
        Hattest du heute sozialen Kontakt?"},
     {"question_id": 3, "type": "multiple choice", "label
        ": "Wie fuehlst du dich heute?"}
  }
}
// Antwortbogen zum Speichern in 'data' Attribut
{
  "module_id": "EMA",
  "ema_id": 1,
  "patient_id" : 1
  "answersheet_id": 1
  "answers": [
      {"question_id": 1, "answer": 5},
      {"question_id": 2, "answer": "Yes" },
      {"question_id": 3, "answer": [1, 3, 4]}
  ]
  "timestamp": 1629196656
}
```

Listing 5.1: Representation von EMAs und Antwortbögen

Angelehnt an das Cordova-Plugin LocalNotifications[1] werden EMAs per Definition von einmaligen oder wiederkehrenden Zeitpunkten sowie durch Geofences geplant (siehe Listing 5.2).

```
1  "module_id": "EMA",
2  "action": "configure"
3  "configuration":[
4    {
5      "ema_id": 1
6      "every": [
7        {"week": 2, "day": 5, "hour":18, "minute":30}
8      ],
9      "location": [
10       {"center": [x, y], "radius": 15, "notifyOnEntry": "
             true"}
11     ]
12   }
13 ]
```

Listing 5.2: Definition von Zeitpunkten und Geofences

5.1.4 Beispiel

Wie die Freischaltung einer Intervention *just-in-time* abläuft, wird anhand eines Beispiels veranschaulicht. Listing 5.3 zeigt die Implementierung von `activate(.)` (siehe Abschnitt 5.1.2) für einen Freischalttrigger, der einem Patient eine Intervention anbietet, wenn er nicht einschlafen kann. Bei Aktivierung des Triggers wird hierfür ein Listener für das EMA-Modul registriert (siehe Z. 11). Empfängt das Modul ein Answersheet aus dem EMA-Modul, wird überprüft, ob dieses zum EMA mit der ID 1 gehört (siehe Z. 12). Ist dies der Fall, so wird über die Antworten der Fragen

[1] www.github.com/katzer/cordova-plugin-local-notifications

mit ID 6 und 7 ausgelesen, ob der Patient schlecht schläft und Hilfestellung haben möchte (siehe 13-16). Dies wird in den beiden Variablen `reportedBadSleep` und `wantsHelp` gespeichert. Ebenfalls wird ein Listener für ein Modul zur Schlaferkennung registriert, das intern GPS-, Bewegungs- und Handynutzungsdaten verwendet. Empfängt der Trigger nachts ein Event, dass der Patient nicht schläft, so wird geprüft, ob er im letzten EMA schlechten Schlaf angegeben und Hilfestellung angefordert hatte (siehe Z. 21). Ist das der Fall, so wird ihm eine Intervention über eine lokale Benachrichtigung geschickt.

```
1  // Constants
2  EMA_ID = 1
3  EMA_QUESTION_ID_SLEEP = 6
4  EMA_QUESTION_ID_HELP = 7
5  // State
6  reportedBadSleep = False
7  wantsHelp = False
8
9  activate(configuration: Object): boolean {
10  // Register for changes in EMA module
11  getModule("EMA").data_emitter.onChange((answersheet) => {
12    if (answersheet.ema_id === EMA_ID) {
13        reportedBadSleep = answersheet.answers
14                           [EMA_QUESTION_ID_SLEEP].answer
15        wantsHelp = answersheet.answers
16                           [EMA_QUESTION_ID_HELP].answer
17     }
18  });
19  // Register for changes in Sleepdetection module
20  getModule("SLEEPDETECTION").data_emitter.on...((sleeps) => {
21    if (isNight() && !sleeps && reportedBadSleep && wantsHelp){
22        sendIntervention()
23    }
24  )}
25  return True;
26 }
```

Listing 5.3: Funktionsweise eines Freischalttriggers

5.1.5 Probleme bei der Umsetzung auf PWAs

Für eSano wird untersucht, ob sich das Konzept zur Realisierung der *Just-In-Time* Eigenschaft auf Web-Apps und PWAs umsetzen lässt. Hierbei gibt es derzeit folgende Probleme:

1. Die Notifications API zum Triggern von lokalen Benachrichtigungen wird von Browsern in iOS-Geräten nicht implementiert [46]. Dies ist relevant zur Durchführung von EMAs, wenn das Gerät offline ist.

2. In Browsern für Android- und iOS-Geräten existiert derzeit keine API, mit der lokale Benachrichtigungen ausgelöst werden können, wenn die Web-App oder PWA geschlossen ist (vgl. native Android Alarm Manager API[2]). Analog können keine Push Benachrichtigungen empfangen werden, wenn die Web-App oder PWA geschlossen ist.

3. Browsern für Android- und iOS-Geräten haben keine verlässliche API, mit der lokale Benachrichtigungen empfangen werden können, wenn die Web-App oder PWA sich im Hintergrund befindet. Hinsichtlich der Service Worker API kann Code zwar im Hintergrund ausgeführt werden. Nach Spezifikation können Service Worker jedoch jederzeit vom Browser terminiert werden [47]. Die Terminierung geschah bei Tests in Chrome und Firefox ungefähr nach einer Stunde. Service Worker werden aus dem terminierten Zustand wieder reaktiviert, wenn der Browser eine Benachrichtigung über die Push API empfängt. Für die Umsetzung von EMAs mit lokalen Benachrichtigungen macht sie das jedoch ungeeignet.

4. Für Browser in iOS-Geräten gibt es keine APIs, um auf Sensoren zuzugreifen [48]. Apple verweigert dies aus Datenschutzgründen [49].

5. Firebase[3] und One Signal[4], zwei der größten Anbieter von Services Push Benachrichtigungen, bieten keine ausreichenden Funktionen zur Planung von Push Benachrichtigungen an. Die serverseitige Planung von Push Benachrichtigungen müsste demnach selbst entwickelt werden.

[2] developer.android.com/reference/android/app/AlarmManager
[3] firebase.google.com
[4] www.onesignal.com

Derzeit läuft ein Vorhaben zur Erweiterung der Notifications API um Notification Triggers in Google Chrome [50]. Etabliert sich die API, so werden lokale Benachrichtigungen auf Android über die native AlarmManager API geplant und trotz geschlossener Web-App oder PWA ausgelöst.

5.2 Adaptiveness

Dieses Unterkapitel stellt einen Entwurf zur Umsetzung der *Adaptiveness* Eigenschaft vor. Teile davon basieren auf der Masterarbeit von Lena Stenzel mit dem Titel *"Design and Implementation of a Module to Enable Conditional Content in Internet- and Mobile-based Interventions"* und sind daher bereits in eSano implementiert.

5.2.1 Konditionen, Wiederholungen und Blöcke

Zur Definition von Konditionen wird die Datenstruktur aus Listing 5.4 verwendet. Diese erlaubt es komplexe Bedingungen in Form eines Evaluationsbaumes zu definieren [51].
Der `ifBlock` enthält den Wurzelknoten. Ein Knoten ist entweder ein innerer Knoten oder ein Blatt. Der Typ wird gespeichert in `nodeType`. Innere Knoten entsprechen booleschen Operatoren und besitzen eine Menge an Kindknoten als Argumente. Diese können wiederum einem inneren Knoten entsprechen. Durch die Rekursion entsteht der Evaluationsbaum. Ein Blatt zeigt immer auf die Frage einer Lektion durch `questionID` und definiert mit `operation` und `value` eine evaluierbare Kondition durch die Nutung von Vergleichsoperatoren. `leafType` speichert den Fragetyp, auf den das Blatt zeigt. Entsprechend des Typs kann `value` geparst werden.
Der Knoten aus Zeile 14 entspricht dem unären NOT-Operator und besitzt daher nur einen Kindknoten.

```
1 {
2     "condition": {
3         "ifBlock": {
4             "nodeType": "and",
5             "children": [
```

```
                    {
                      "nodeType": "leaf",
                      "leafType": "slider",
                      "questionID": 4,
                      "value": 5,
                      "operation": "less"
                    },
                    {
                      "nodeType": "not",
                      "children": [
                        {
                          "nodeType": "leaf",
                          "leafType": "date",
                          "questionID": 1,
                          "value": "2021-09-07",
                          "operation": "equals"
                        }
                      ]
                    }
                ]
            },
            "thenBlock": {
                "repeat": {
                    "question": false,
                    "numberOfTimes": 3
                }
            }
        },
}
```

Listing 5.4: Datenstruktur für Konditionen

In `thenBlock` ist definiert, was passiert, wenn die Kondition zu `True` evaluiert. Ist `questionFalse`, so definiert `numberOfTimes` die statische Anzahl, mit der innere Elemente des Blocks wiederholt werden. Ist `questionTrue`, so wird unter Angabe von `questionID` eine dynamische Wiederholung definiert. Während der Patient die

Lektion bearbeitet, wird die Antwort der referenzierten Frage zur Laufzeit benutzt, um den Block n-mal zu wiederholen. Das Attribut zeigt immer auf eine Frage mit einer numerischen Antwort.
Damit ist die Frage geklärt, wie Konditionen und Wiederholungen definiert werden. Zur Adaptierung von Inhalten innerhalb von Lektionen muss jedoch noch gekennzeichnet werden, welche Inhalte unter Einhaltung einer Konditionen versteckt oder angezeigt werden. Hierfür werden Blöcke als neues Konzept eingeführt. Diese sind an Konditionen gekoppelt und besitzen eine Menge an inneren Elementen, die abhängig von der Kondition angezeigt, versteckt oder wiederholt werden. Zur Kennzeichnung von Blöcken innerhalb Lektionen werden die zwei neuen Elemente `BlockOpens` und `BlockCloses` eingeführt.

5.2.2 Endnutzer-Programmierung

Die Erstellung und Anpassung von Lektionsinhalten erfolgt durch Editoren und E-Coaches. Zur Definition von Blöcken wird der Inhaltseditor wie in Abbildung 5.6 erweitert.

1) Es können nun BlockOpens Elemente eingefügt werden.

1.1) Auswahl, ob ein Block konditional, aufklappbar oder beides ist.

1.2) Auswahl, ob Kondition einfach oder komplex ist. Bei einer einfachen Kondition wird ein Blatt definiert. Für eine komplexe Kondition öffnet sich ein Konditionseditor (siehe Abbildung 5.7).

1.3) Auswahl, ob der Block innere Elemente wiederholt und ob die Wiederholung statisch oder dynamisch abläuft. Bei der dynamischen Wiederholung erscheint ein Dropdown Menü, das alle Fragen der Lektion beinhaltet.

2) Einfügen innerer Elemente in den Block.

Zusätzlich wird E-Coaches und Editoren ein Konditionseditor wie in Abbildung 5.7 zur Verfügung gestellt, mit dem sie Konditionen für Blöcke mittels Drag-and-Drop programmieren können. Der Editor erzeugt die Datenstruktur aus Unterabschnitt 5.2.1 und funktioniert wie folgt:

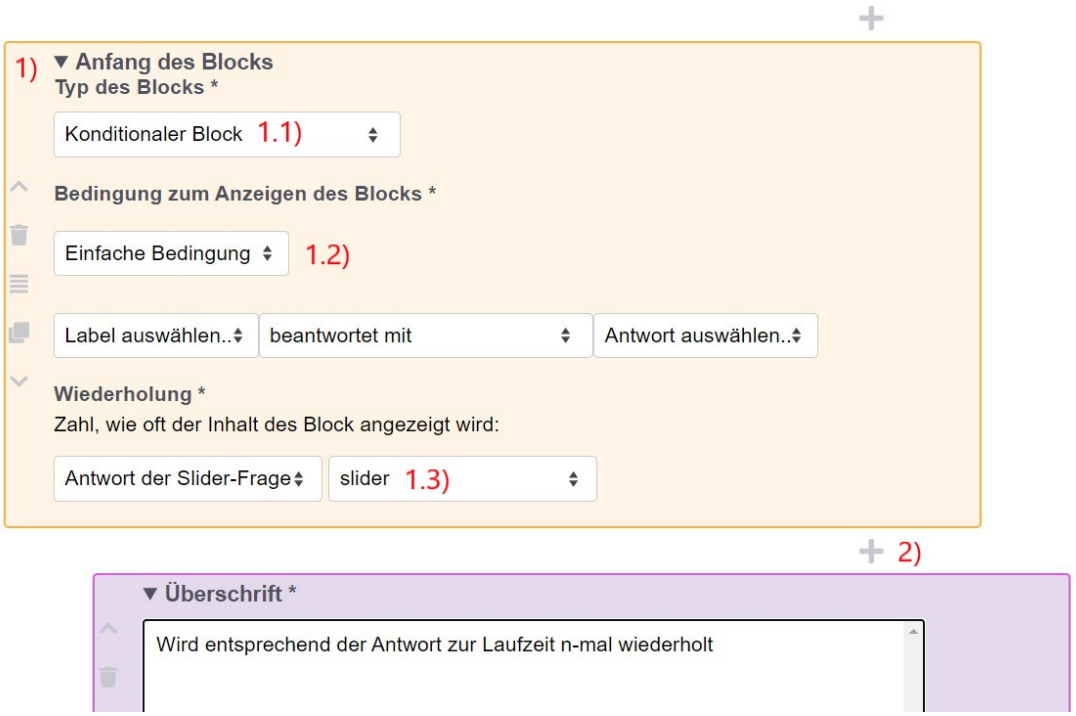


Abbildung 5.6: Definition von Blöcken durch Endnutzer-Programmierung

1) Scrollbares Menü zur Auswahl verschiedener Knoten.

1.1) Auswahl eines Blattknotens.

1.2) Auswahl eines inneren Knotens, wobei mehrere boolesche Operationen zur Verfügung stehen. Die grauen Flächen repräsentieren die Anzahl der Kindknoten, die für die Operation eingefügt werden müssen.

2) Arbeitsfläche, auf die Knoten mittels Drag-and-Drop gezogen werden. Im Beispiel wurde ein UND-Knoten als Wurzelknoten definiert.

2.1) Einfügen des ersten Kindknotens für die UND-Operation. Im Beispiel wurde ein Blatt eingefügt, das über das Dropdown-Menü in 2.1.1) auf eine Slider-Frage referenziert. In 2.1.2) erfolgt die Auswahl der Vergleichsoperation `lessequal` und in 2.1.3) die Konstante, mit der die Antwort der Slider-Frage verglichen wird.

2.2) Einfügen eines zweiten Kindknotens. Im Beispiel wurde noch nichts eingefügt.

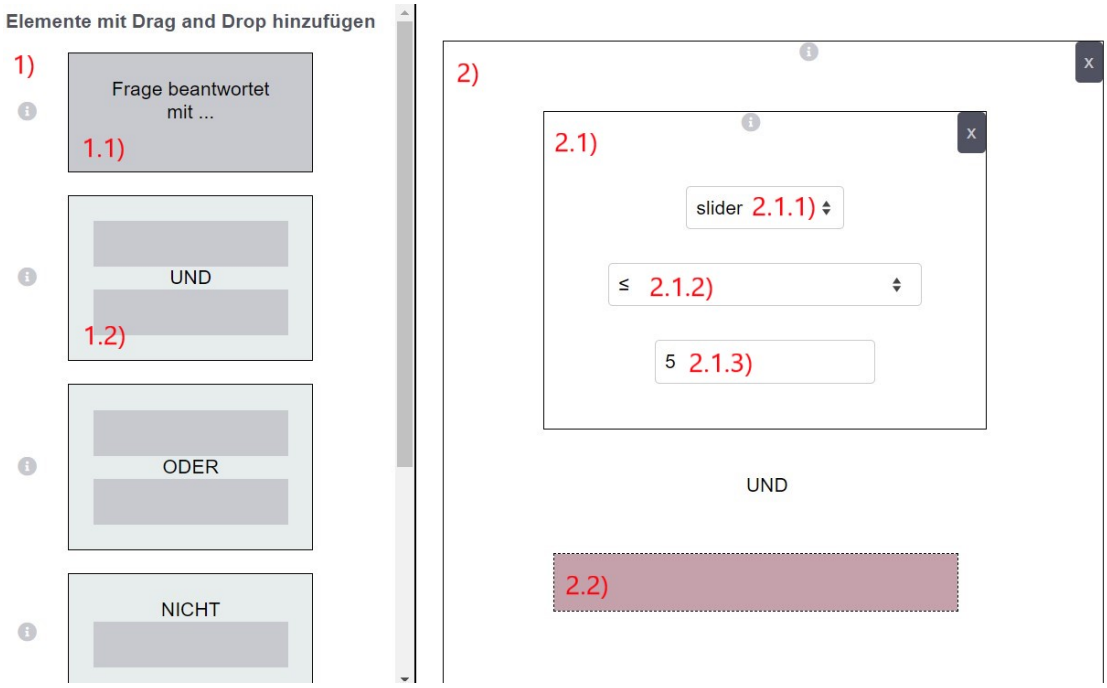


Abbildung 5.7: Definition von Konditionen durch Endnutzer-Programmierung

5.2.3 Evaluation zur Laufzeit

Die App evaluiert die Konditionen aller Fragen, während der Patient die Lektion erarbeitet. Dies geschiet durch Nutzung von Events, mit denen ein Benachrichtigungssystem wie in Abbildung 5.8 erzeugt wird.
Jede Frage in der Lektion besitzt zur Laufzeit ein Attribut `answer`, in das die aktuelle Antwort der Frage gespeichert wird. Jedes BlockOpens Element besitzt zur Laufzeit ein Attribut `hide`. Alle Elemente, die sich innerhalb eines Blocks befinden, registrieren einen Listener für das Attribut `hide` des direkten Vaterblocks. Alle BlockOpens Elemente, für die eine Kondition definiert wurde, registrieren einen Listener bei allen Fragen, die in ihrem Konditionsbaum referenziert sind. Beantwortet ein Nutzer eine Frage, so speichert das Element die Antwort in `answer`. Dadurch werden alle registrierten BlockOpens Elemente mit der ausgewählten Antwort benachrichtigt. Sie fügen die Antwort über die `question_id` in ihren Konditionsbaum ein und evaluieren diesen. Evaluiert die Kondition als `True`, so speichert das BlockOpens Element `False` in sein `hide` Attribut. Die registrierten inneren Elemente empfangen die Änderung und erscheinen bzw. verstecken sich entsprechend der Nachricht. Mit dem

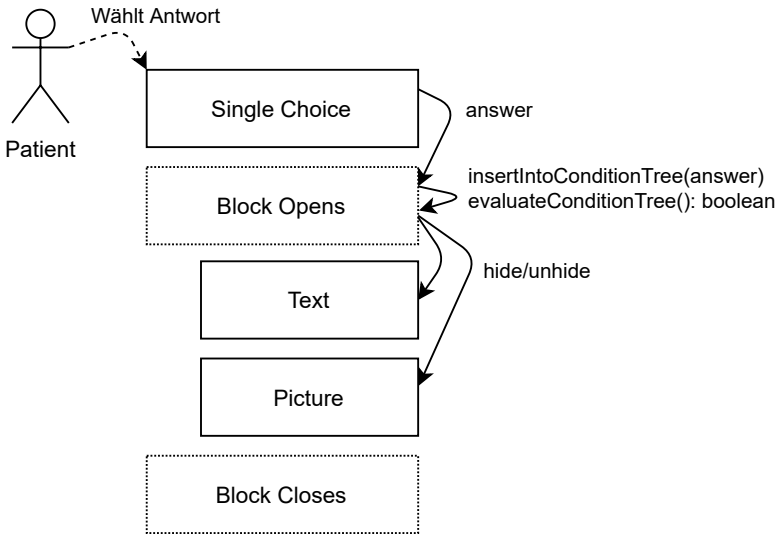


Abbildung 5.8: Inhalte zur Laufzeit ein- und ausblenden

selben Prinzip werden auch dynamische Wiederholungen umgesetzt. Empfängt ein BlockOpens Element die Antwort einer numerischen Frage, so erzeugt es Kopien seiner inneren Elemente und fügt sie zwischen sich und dem zugehörigen BlockCloses Element ein (siehe Abbildung 5.9). Ist der Block zusätzlich noch ein konditionaler Block, so müssen für die neuen Elemente ebenfalls die Listener zu dem BlockOpens Element registriert werden.

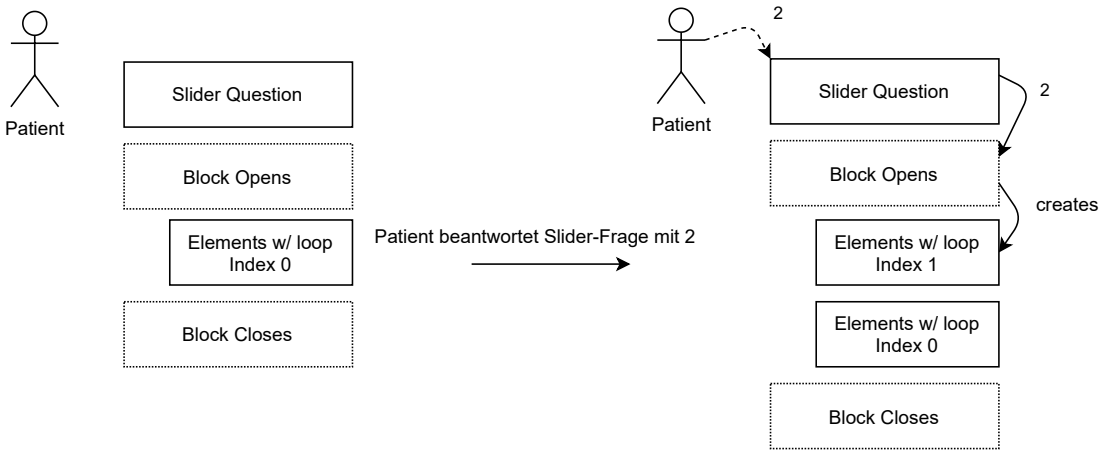


Abbildung 5.9: Inhalte zur Laufzeit wiederholen

Durch das Erzeugen neuer Elemente durch Kopien können Elemente nicht mehr eindeutig über ihre ID identifiziert werden. Das ist problematisch beispielsweise

beim Abgeben von Antworten, da im Nachhinein die Zuordnung von Antworten zu Fragen nicht mehr eindeutig ist. Dieses Problem wird mit Hilfe von Loop-Indizes gelöst. Alle Elemente, die einen dynamischen Wiederholungsblock als Vater haben, bekommen das Attribut `Loop-Index` zugewiesen. Werden neue Elemente eingefügt, so bekommen diese `Loop-Index + 1` als ihren `Loop-Index` zugewiesen. Zur eindeutigen Zuordnung von Antworten werden die Loop-Indizes aller Fragen zusammen mit der ID in den Antwortbogen gespeichert.

5.2.4 Substitutionen

Zur Personalisierung der App und der Inhalte wird pro Patient ein umfangreicher Datensatz angelegt, in den E-Coaches und Patienten Werte eintragen. Auf den Datensatz wird in der App zugegriffen, um Dialoge anzupassen.
Um die Interventionen zu personalisieren, definiert der Editor Substitutionen innerhalb der Intervention (siehe Listing 5.5). Eine statische Substitution referenziert auf ein Attribut im Datensatz des Patienten, wobei für den Fall dass das Attribut noch nicht gesetzt wurde ein Default-Wert hinterlegt ist. Die Auswertung statischer Substitutionsregeln erfolgt bei Beginn der Intervention.

```
1 {
2     "questions": [
3         {
4             "element_id": 1,
5             "element_type": "text",
6             "text": "Hallo {{PATIENT.NAME, schoen dich
                  wieder zu sehen | schoen dich kennen zu
                  lernen}}"
7         }
8     ]
9 }
```

Listing 5.5: Statische Substitutionen

Dynamische Substitutionsregeln referenzieren auf die Antwort einer Frage (siehe Listing 5.6). Die Antwort der referenzierten Frage wird während der Bearbeitung

der Intervention zur Laufzeit an die Position der Substitution eingefügt.

```
{
    "questions": [
        {
            "element_id": 1,
            "element_type": "question/single-choice",
            "label": "Wie geht es dir gerade?",
            "choices": ["Gut", "Schlecht"]
        },
        {
            "element_id": 2,
            "element_type": "question/freetext",
            "label": "Warum denkst du geht es dir gerade
               {{1}}?"
        }
    ]
}
```

Listing 5.6: Dynamische Substitutionen

5.3 Adherence

Dieses Unterkapitel stellt einen Entwurf zur Umsetzung der *Adherence* Eigenschaft vor.

5.3.1 Empfehlungssystem

Hierfür wird ein Algorithmus für ein Empfehlungssystem vorgestellt und in die Architektur aus Unterabschnitt 5.1.1 integriert.

Algorithmus

Der folgende Algorithmus beobachtet depressive Symptome bei einem Patienten und schlägt Gegenmaßnahmen vor. Er basiert auf dem Paper von Fabian Wahle

mit dem Titel *Mobile Sensing and Support for People With Depression: A Pilot Trial in the Wild* [52] und berücksichtigt den Kontext des Patienten sowie Feedback und persönliche Präferenzen.
Interventionen werden anhand von Charakteristiken durch Domänenexperten in die Kategorien *Achtsamkeit*, *Entspannung*, *physische Aktivität* und *soziale Aktivität* unterteilt. Verfügbare Kontexte werden bezüglich dieser Kategorien gewichtet. Die Gewichtung repräsentiert, wie relevant ein Kontext für eine Kategorie ist. Die Punktezahl einer Kategorie wird als *BasketScore* bezeichnet und wie folgt berechnet:

$$BasketScore_m = w_1 * scaleToRange(x_{1_{max}}, x_{1_{min}}, x_1) + ...+ \\ w_n * scaleToRange(x_{n_{max}}, x_{n_{min}}, x_n)$$

w_n: Gewichtung des Kontexts n bezüglich der Kategorie m
x_n: Wert des Kontexts über die letzten 24 Stunden
$scaleToRange()$: Berechnet Anteil von x_n innerhalb $[x_{n_{max}}, x_{n_{max}}]$

In der ersten Phase stammt $x_{n_{min}}$ und $x_{n_{max}}$ aus evidenzbasierten Annahmen über depressives Verhalten von Menschen. Liegen genügend Daten vor, so wird in einer zweiten Phase zur Verbesserung der Empfehlungsqualität $x_{n_{min}}$ und $x_{n_{max}}$ als $\mu \pm (2 * \sigma)$ definiert. Unter Berücksichtigung des Durchschnittswerts eines Kontexts und der 2-Sigma-Regel ist die Empfehlung dadurch besser auf die eigentliche Verhaltensweisen des Patienten zugeschnitten.
Hat eine Kategorie a eine höhere Punktzahl als eine Kategorie b, so ist eine Intervention aus a zu diesem Zeitpunkt hilfreicher als eine Intervention aus b. Innerhalb einer Kategorie wird die für jede Intervention eine Punktezahl *InterventionScore* wie folgt berechnet:

$$InterventionScore = 0.75 * (pastRatings/5) \\ - 0.25 * cancelationRate \\ + 0.5 * (if\ random \leq 0.05)$$

$pastRatings$: Durchschnittliche Bewertung der Intervention in der Vergangenheit
$cancelationRate$: Anteil, wie oft der Patient die Intervention in der Vergangenheit abgebrochen hat

$random$: Gleichverteilte Zufallsvariable in $[0, 1]$

Die statischen Gewichtungen werden unter folgenden Annahmen gesetzt: Vergangene Bewertungen repräsentieren die Präferenzen des Patienten und sollen den höchsten Einfluss auf die Punktezahl haben. Abgebrochene Interventionen wirken sich negativ auf die Punktezahl aus. Weil ein Abbruch nicht gezwungenermaßen heißt, dass der Patient die Intervention nicht mochte, wird die Abbruchrate geringer gewichtet, als vergangene Bewertungen. Der Zufallswert verhindert, dass Interventionen, die in der Vergangenheit im Durchschnitt schlecht bewertet wurden, über einen zu langen Zeitraum nicht empfohlen werden.
Interventionen sind für einen Zeitraum gesperrt, nachdem sie bearbeitet wurden. Die Anzahl an Stunden berechnet sich in Abhängigkeit der durchschnittlichen Bewertung aus der Vergangenheit wie folgt:

$$BlockTime = 36 * (6 - pastRating)$$

Integration

Das Empfehlungssystem wird als Triggermodul in die Architektur aus Unterabschnitt 5.1.1 eingefügt (Empfehlungstrigger). Der Empfehlungstrigger empfängt Kontexte von Analysemodulen und berechnet bei jedem Update die relevanteste Kategorie. Ist der *BasketScore* einer Kategorie höher als ein festgelegter Schwellwert, bietet er dem Patienten *n* Interventionen mit dem höchsten *InterventionScore* aus dieser Kategorie an. Der Schwellwert wird von Domänenexperten bestimmt und gibt einem E-Coach die Möglichkeit, den Grad der Hilfestellung anzupassen. Der Nutzer bearbeitet die Intervention und bewertet diese. Daraufhin werden die Werte *cancelRate*, *pastRatings*, *blockTime*, *random* und *InterventionScore* der bearbeiteten Intervention aktualisiert.

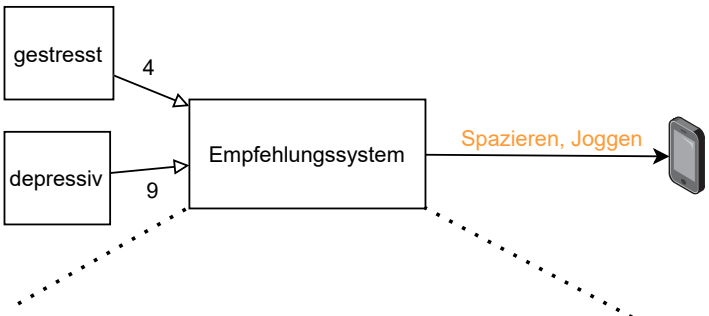
Beispiel

Der Algorithmus und die Integration werden im Folgenden anhand eines Beispiels veranschaulicht (siehe Abbildung 5.10). Die Gewichtungswerte, der Schwellwert und $x_{n_{min}}$ bzw. $x_{n_{max}}$ sind frei erfunden und wurden nicht von Domänenexperten

bestimmt.
Es existieren zwei Analysemodule, die berechnen, wie gestresst und depressiv ein Patient ist. Die Analysemodule liefern jeweils einen Wert im Intervall [0, 10]. Für die Kategorie *physische Aktivität* stehen die Interventionen *Spazieren* und *Joggen* zur Auswahl, für die Kategorie *Entspannung* die Intervention *Meditation*. *Spazieren* hat der Patient bereits mit 5 und 4 bewertet. *Joggen* und *Meditation* hat er jeweils mit 5 Sternen bewertet, wobei er *Joggen* einmal abgebrochen hat.
Der Empfehlungstrigger empfängt nun die Werte *gestresst = 4* und *depressiv = 9*. Das Empfehlungssystem berechnet die `BasketScores` aller Kategorien, wobei $BasketScore_{physischeAktivität} = 0.4 * 0.2 + 0.9 * 0.8 = 0.8$ den Schwellwert von 0.8 erfüllt. Der Empfehlungstrigger sucht in der Interventionstabelle nach den zwei Interventionen in der Kategorie *physische Aktivität* mit dem höchsten *InterventionScore*. Er wählt die Interventionen *Joggen* und *Spazieren* und schickt sie dem Nutzer über eine lokale Benachrichtung aufs Smartphone.
Für $scaleToRange$ ist bisher $x_{n_{min}} = 0$ und $x_{n_{max}} = 10$ verwendet worden. Ergibt sich nach Anpassung des Kontexts *depressiv* über die 2-Sigma-Regel beispielsweise $scaleToRange(3, 8, .)$ so bedeutet dies, dass der Patient in der Vergangenheit zu wenig Hilfe erhalten hat.
Es existieren interessante Varianten zur Integration des Empfehlungssystems. Fabian Wahle verwendete beispielsweise die *BasketScores*, um den Patienten selbst über ein klickbares Symbol die Kategorie auswählen zu lassen. Je höher der *BasketScore*, desto größer erscheint das Symbol in der GUI. Der Patient erhält dadurch mehr Entscheidungsfreiheit. Zusätzlich könnten für die Empfehlung Nutzerprofile berechnet und berücksichtigt werden.

5.3.2 Visualisierung der Daten

Gemäß der Datenschnittstelle sind alle Datensätze mit Timestamps gekennzeichnet und eignen sich zum Abtragen auf einer zeitlichen Achse. Bei näherem betrachten von EMA-, Sensor-, Kontext- und Metadaten fallen verschiedene Zielmengen auf, in welche die Daten abbilden. Die Daten werden anhand dieser Charakteristik kategorisiert und mit Hilfe verschiedener Diagramme abgebildet.



	gestresst	depressiv
Wert über letzten 24h	4	9
scaleToRange(0,10, .)	0.4	0.9

Gewichtungen			**BasketScore**	**BasketScore >= 0.8**	**BasketScore >= 0.8 & maxIntervention(2)**
soziale Aktivität	0.0	1.0	0.21	False	-
physische Aktivität	0.2	0.8	0.8	True	Spazieren, Joggen
Entspannung	0.9	0.1	0.45	False	-
Achtsamkeit	0.3	0.1	0.21	False	-

Intervention	Kategorie	cancelRate	pastRating	blockTime	random	**InterventionScore**
Spazieren	physische Aktivität	0.0	4.5	0	0	0.675
Joggen	physische Aktivität	0.5	5	0	0	0.625
Meditation	Entspannung	0.0	5	0	0	0.75

Abbildung 5.10: Integration des Empfehlungssystems in Architektur

Intervalle als Zielmenge

Eine Kategorie von Daten bildet Zeitpunkte auf Punkte oder Teil-Intervalle von Intervallen ab. Die Intervalle können diskret, stetig, offen, halboffen oder geschlossen sein. Beispiele hierfür sind Bewegungsdaten von Sensoren (stetig, geschlossen), Likert- und Slider-Fragen in EMAs (diskret, geschlossen) oder die Anzahl der Logins an einem Tag bei Metadaten (diskret, halboffen). Abbildung 5.11 veranschaulicht beispielhaft die Visualisierung von Datensätzen, die Zeitpunkte auf Punkte in

einem diskreten, geschlossenen Intervall abbilden, mit Hilfe eines Bar-Diagramms. Das Beispiel stellt die Antworten des Patienten auf Fragen nach dessen Befindlichkeit dar. Dabei standen ihm Werte auf dem Intervall `[0,5]` zur Auswahl. Für die Abbildung auf offene Intervalle eignen sich das Minimum und das Maximum des Datensatzes als obere und untere Grenze der y-Achse. Analog zu den positiven Werten kann das Bar-Diagramm auch negative Werte darstellen.
Das Diagramm eignet sich zudem zur Visualisierung von Abbildungen im multidi-

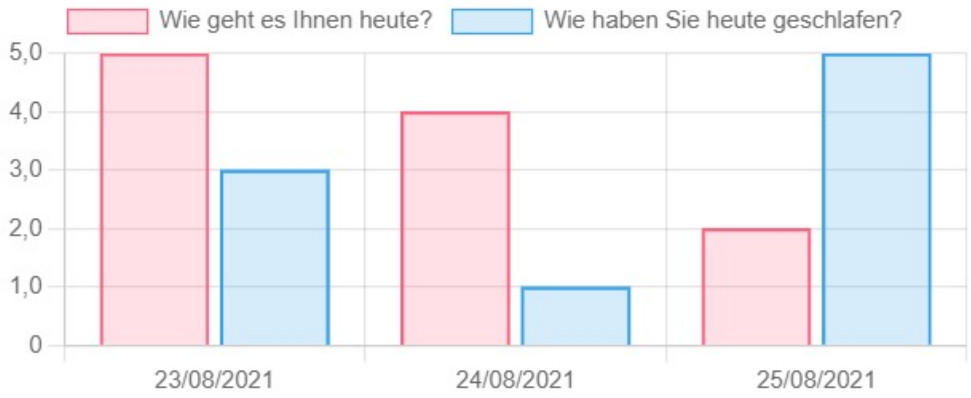


Abbildung 5.11: Visualisierung von Intervallen

mensionalen Raum, was beispielsweise bei Bewegungssensoren eine Rolle spielt. Diese bilden einen Zeitpunkt als Tripel `(x, y, z)` im dreidimensionalen Raum ab.

Diskrete Mengen als Zielmenge

Eine weitere Kategorie von Daten bildet Zeitpunkte auf diskrete Mengen ab. Beispiele hierfür sind Single-Choice oder Multiple-Choice-Fragen in EMA-Daten mit ein- bzw. mehrlementigen Teilmenge der Menge `{Option A, Option B, ...}`. Ein weiteres Beispiel sind Kontexte, die den Zustand eines Patienten berechnen, z. B. ob ein Patient schläft oder nicht. Dabei wird ein Zeitpunkt auf einelementige Teilmengen der Menge `{Patient schläft, Patient schläft nicht}` abgebildet.
Abbildung 5.12 veranschaulicht beispielhaft Datensätze, die einen Zeitpunkt auf eine Menge abbilden, mit Hilfe eines Punkte-Diagramms. Auf der y-Achse wird die Zielmenge abgetragen, die x-Achse gibt die Zeitpunkte an. Die gesetzten Punkte

bilden die Teilmengen.

Abbildung 5.12: Visualisierung von Teilmengen diskreter Mengen

Texte als Zielmenge

Eine weitere Kategorie von Daten bildet Zeitpunkte auf Text ab. Abbildung 5.13 veranschaulicht, wie diese Daten visualisiert werden können. Das Beispiel zeigt Daten einer Textfrage in einem EMA, bei welchem ein Patient täglich seine Gedanken notieren soll. Anwendungsbeispiele bei Sensor- oder Kontextdaten sind das Loggen von Tastaturanschlägen oder die Spracherkennung über das Mikrofon.

Abbildung 5.13: Visualisierung der Abbildung von Zeitpunkten auf Texte

Chart.js

Die Diagramme aus Unterabschnitt 5.3.2 wurden mithilfe von Chart.js[5] erstellt. Der Code der Diagramme kann aus Listing A.1 entommen werden. Die Bibliothek basiert auf Javascript und lässt sich damit in Web-Apps, PWAs und Native-Apps (WebView) integrieren. Sie bietet verschiedene Diagramm-Typen, verschiedene Möglichkeiten zur Beschriftung und Skalierung der Achsen und ist über Callback APIs äußerst anpassbar. Die Bibliothek hat standardmäßig Nutzerinteraktionen implementiert, durch die mittels Anklicken der Legende, Datensätze ein- oder ausgeblendet werden. Diagramme, Achsen und Beschriftungen skalieren automatisch und passen sich großen bzw. kleinen Zeiträumen sowie den Datensätzen und der Bildschirmgröße entsprechend an. Alternativ können Daten aggregiert oder Diagramme scrollbar gemacht werden. Die Bibliothek eignet sich damit zur Visualisierung der Daten für E-Coaches und Patienten.

GPS-Daten als Zielmenge

GPS-Daten werden in der Regel über den Längen- und Breitengrad angegeben und verfügen über einen Zeitstempel. Interessant ist der hauptsächliche Standort eines Patienten über einen Zeitraum und die Relation der Daten zu Points-Of-Interest (POIs). Ein POI ist beispielsweise das Zuhause, die Arbeit oder das Fitnessstudio. Ein Patient kann zu Beginn selbst POIs angegeben. Weitere lassen sich durch Visualisierung der GPS-Daten erschließen. Hierfür eignet sich die Nutzung von Heatmaps (siehe Abbildung 5.14). Eine Heatmap verstärkt die Bereiche auf der Karte, für die eine hohe Dichte an Datensätzen existiert. Über die euklidische Distanz, Geofences oder Cluster-Techniken lässt sich abschätzen, wie viele Punkte des Datensatzes sich auf einen POI beziehen. Daraus wird schließlich der Zeitanteil des POI in Relation zum gesamten Datensatz abgeschätzt. Zum genaueren Verfolgen des Standorts wird ein Zeit-Slider benutzt, der den Datensatz anhand des Timestamps durchläuft und den zugehörigen Punkt auf der Karte einzeichnet.

[5] www.chartjs.org

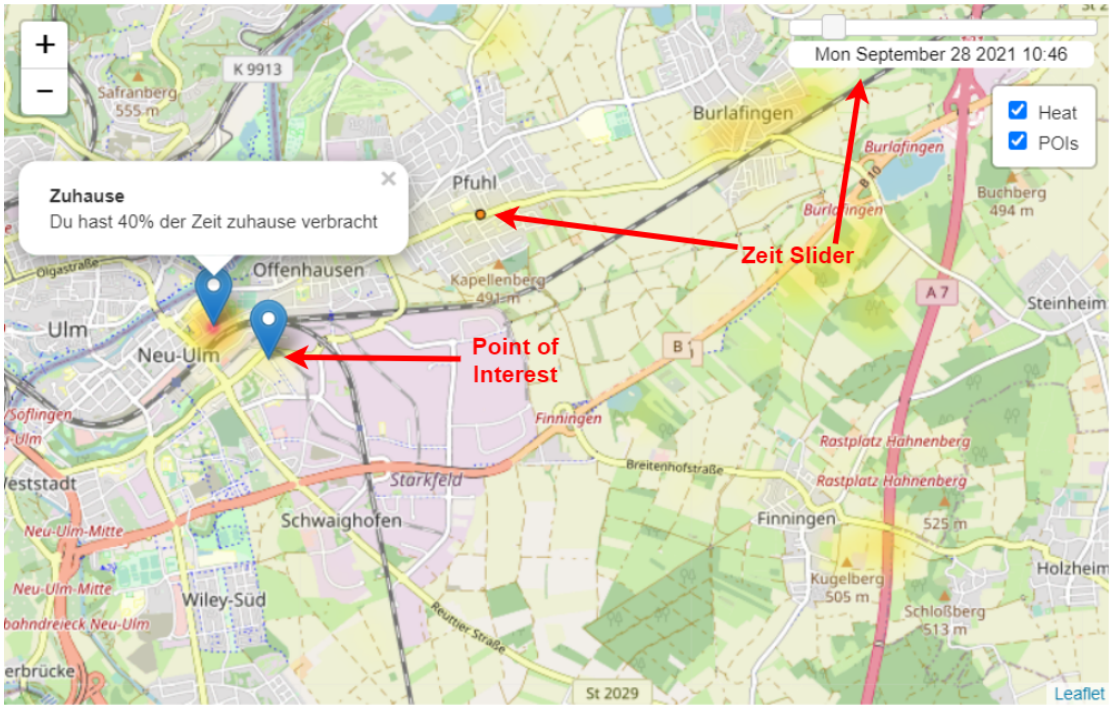


Abbildung 5.14: Heatmap mit Point-Of-Interests, Time-Slider und zeitlicher Abschätzung

Abbildung 5.14 wurde mithilfe von Openstreatmap[6], Leaflet.js[7], Heatmap.js[8] und LeafletSlider[9] erstellt. Leaflet.js erlaubt es, Datensätze über eine Karte zu schichten und mittels Checkboxen zur Laufzeit ein- und auszublenden. Dies wurde für die Heat-Visualisierung und für POIs benutzt. Der zugehörige Code befindet sich in Listing A.2. Außerdem lässt sich mit Leaflet.js auf Karten eine Strecken anstatt von Punkten einzeichnen oder ein Play-Button zum automatischen Verfolgen des Standorts integrieren. Die genutzten Bibliotheken basieren auf Javascript und lassen sich folglich in Web-Apps, PWAs und Native-Apps (WebView) integrieren.

[6] www.openstreetmap.org
[7] www.leafletjs.com
[8] www.patrick-wied.at/static/heatmapjs
[9] www.github.com/Falke-Design/LeafletSlider

6 Implementierung

Für den praktischen Teil der Masterarbeit steht die *Adaptiveness* Eigenschaft des Konzepts im Fokus. eSano erfüllt bereits die Anforderungen AC-02, AC-03, AC-04 und wird im Rahmen dieser Arbeit um AC-01 erweitert. Die Implementierung für das CMS, die E-Coach Plattform und das Backend wird nach Vorstellung des Konzepts an die dafür zuständigen Programmierer delegiert. Die Umsetzung in der App erfolgt eigenständig. Sie wird in den folgenden Kapiteln anhand der *Better Care* Studie demonstriert (siehe auch Unterabschnitt 2.4.2). Der gezeigte Code des Kapitels stammt aus der App. Zur besseren Lesbarkeit wurde der Code an manchen Stellen gekürzt (`...`) und die Pseudofunktion `in` bzw. `not in` verwendet.

6.1 Neuer Freischalttyp

eSano besitzt verschiedene Typen zum Freischalten einer Lektion. Es wird ein neuer Typ `conditional` eingeführt, der beschreibt, dass eine Lektion automatisiert freigeschaltet wird, wenn eine Kondition erfüllt ist. Der Dialog zur Konfiguration von Interventionen im CMS und der E-Coach Plattform wird angepasst, sodass im Dropdown-Menü der neue Typ *automatisierte Freischaltung* erscheint (siehe Abbildung 6.1). Wird dieser ausgewählt, so erscheint ein weiteres Dropdown-Menü, über das die Lektion oder das Tagebuch gewählt wird, auf das sich die Kondition bezieht. Ebenfalls erscheint der Button *Bedingung festlegen* zur Weiterleitung zum Konditionseditor, welcher im CMS bereits vorhanden ist. Die Kondition einer Lektion des Freischalttyps *conditional* wird gemäß der Datenstruktur aus Unterabschnitt 5.2.1 an die Lektion geknüpft und mittels `questionnaire_id` auf die Lektion referenziert, auf die sich die Kondition bezieht. In Listing 6.1 wird definiert, dass die konditionale Lektion *Fatigue - Damit leben* von *Better Care* freigeschaltet wird, insofern

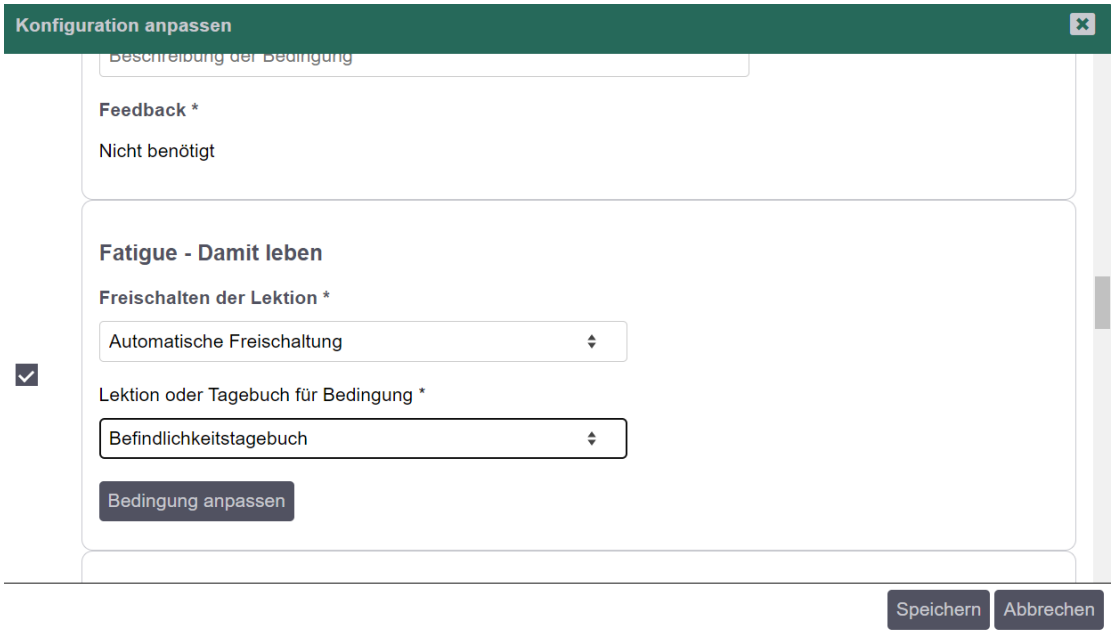


Abbildung 6.1: Angepasste Interventionskonfiguration im CMS

eine Multiple-Choice-Frage des Tagebuchs mit dem Wert *Erschöpfung* beantwortet wurde.

```
...
"default_configuration": {
  "questionnaire_configuration": [
...
    {
      "id": 3375, // Fatigue - Damit leben
      "unlock_type": "conditional",
      "condition": {
        "questionnaire_id": 3373, // Tagebuch
        "ifBlock": {
          "nodeType": "leaf",
          "leafType": "question",
          "questionLabel": 30, // Multiple-Choice
          "value": "erschoepfung"
        }
      }
    },
```

```
18  ...
```

Listing 6.1: Datenstruktur für neuen Lektionstyp

6.2 Freischaltung von Zusatzinhalten

Bearbeitet der Patient das Tagebuch und gibt seine Antworten ab, so werden anschließend die Konditionen aller Lektionen des Typs `conditional` überprüft. Voraussetzung ist, dass die Lektion noch nicht freigeschaltet ist und die Kondition sich auf das Tagebuch bezieht. (siehe Listing 6.2 Z. 2-4).

```
1  for (let lection of lections){
2     if (lection.unlockType === 'conditional' &&
3        this.lectionID === lection.condition.lectionID &&
4        lection not in unlocked_questionnaires){
5        // insert answers
6        this.elements
7          .filter(element =>
8            element instanceof QuestionModel &&
9            element.answer !== null)
10         .map(question => question.answer)
11         .map(answer => lection.condition.insertAnswer(answer))
12       // evaluate the condition
13       if (condition.evaluate()){
14           // offer lection to user
15           lection not in offerToUser ?
16             offerToUser.push(lection)
17       }
18    }
19 }
```

Listing 6.2: Evaluation konditionaler Lektionen

Alle Lektionen, deren Kondition als `True` evaluieren, erscheinen dem Patienten als scrollbare Liste in einem Popup (siehe Abbildung 6.2). Durch An- und Abwählen schaltet der Patient neue Lektionen frei, wobei angewählte Lektionen grün umrandet sind. Wählt der Patient keine der vorgeschlagenen Lektionen aus, so wird er

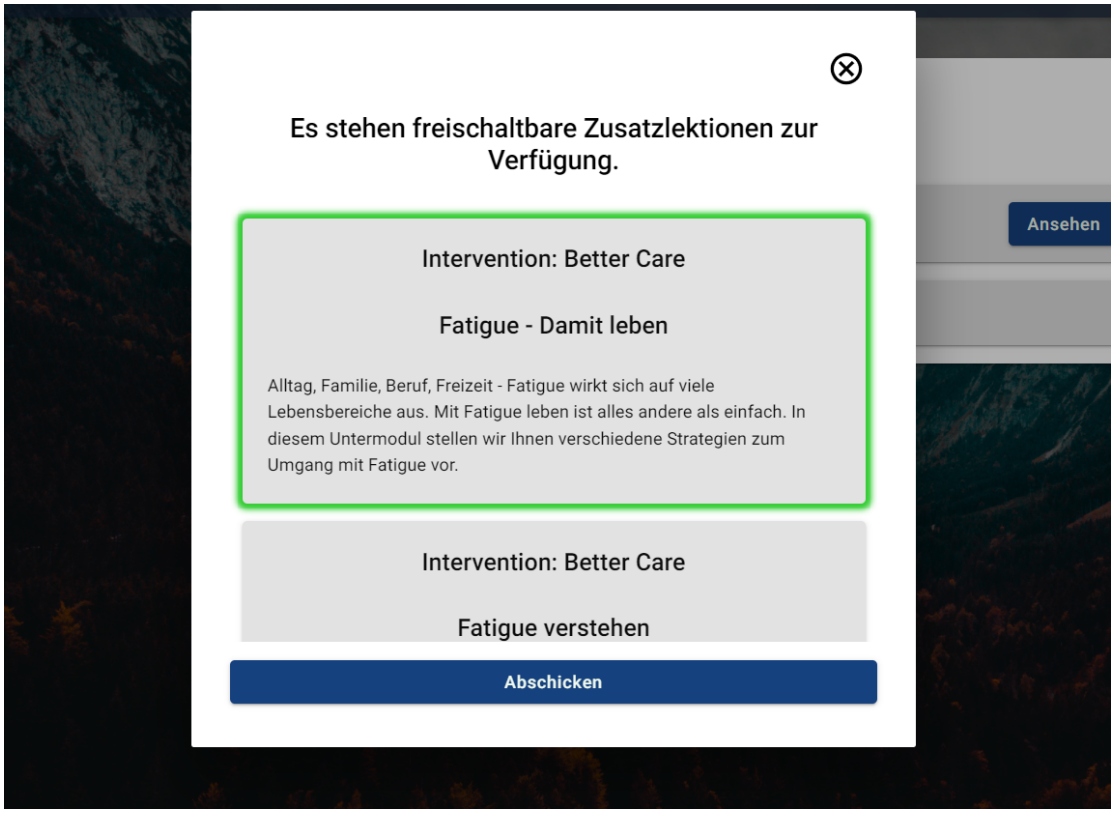


Abbildung 6.2: Popup in Patientenapp zur Freischaltung von Zusatzinhalten

vor Verlassen des Dialogs gewarnt. Die Freischaltung der angewählten Lektionen erfolgt über eine Unlock API (siehe Listing 6.3). Der Server prüft vor Freischaltung, ob die Intervention sich in laufendem Zustand befindet und die referenzierten Lektionen bzw. Tagebücher Teil der Intervention sind.

```
1 {
2   "questionnaire_id" : 3073 // Tagebuch
3   "unlock_questionnaires": [3375], // Fatigue - Damit
        leben
4 }
```

Listing 6.3: POST-Request zum Freischalten konditionaler Lektionen

Insofern die Freischaltung erfolgreich war, wird der Patient benachrichtigt. Besucht er nach einer erfolgreichen Freischaltung die Interventionsübersicht, werden ihm die zusätzlichen Lektionen unter dem Abschnitt *Zusatzlektionen* angezeigt (siehe

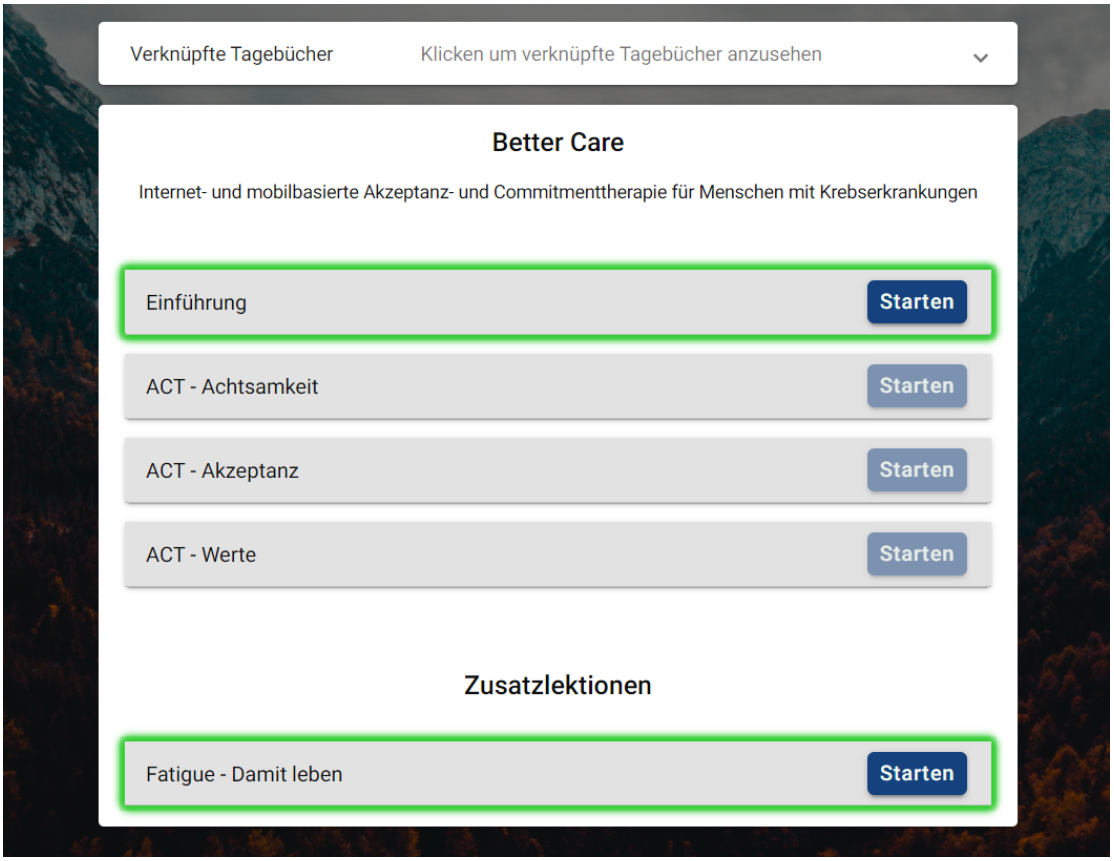


Abbildung 6.3: Angepasste Interventionsübersicht in der Patientenapp

Abbildung 6.3).

6.3 Evaluation der Konditionen

Die Evaluation der Konditionen erfolgt in der ersten Version in der App, da vorhandener Code aus AC-02 wiederverwendet werden kann, bis das Backend die Evaluation umsetzt. Die Evaluation läuft wie folgt ab: Der boolsche Evaluationsbaum aus Unterabschnitt 5.2.1 wird mithilfe der Klassen `Node` und `Leaf` aufgebaut, wobei Leaf eine Unterklasse von Node ist (siehe Listing 6.4). Anschließend werden alle Antworten des Patienten anhand `position_id` in die Blätter eingefügt. Blätter ohne Antwort speichern als Default `null`.

```
1 export class Node {
```

```
2       children: Array<Node>;
3       operation: string;                  // bool operator
4   }
5
6   export class Leaf extends Node {
7       type: QuestionTypes;
8       answer: any;
9       positionID: number;
10      value: any;
11      operation: string;                  // compare operator
12      secondValue: number;
13  }
```

Listing 6.4: Knoten und Blätter als Klassen

Nach der Initialisierung wird der Evaluationsbaum durch Aufruf von `evaluate(): boolean | null` auf dem Wurzelknoten mittels Tiefensuche rekursiv durchlaufen (siehe Listing 6.5). Ist ein Knoten ein Blatt, so wird die Methode `evaluateLeaf(): boolean | null` aufgerufen. Erfüllt das Blatt die Kondition, wird `True` zurückgegeben. Erfüllt es sie nicht, so wird `False` zurückgegeben. Fehlen Antworten, um die Kondition zu evaluieren, so wird `null` zurückgegeben. Ein Knoten kann also drei Wahrheitswerte annehmen.
Das Ergebnis eines rekursiven Aufrufs liegt in den Zeilen 11 und 14 vor. Anhand von `operation` wird eine Fallunterscheidung vorgenommen, die entsprechende boolsche Funktion auf die beiden Ergebnisse angewandt und das Ergebnis zurückgegeben. Insofern ein Operator der boolschen Funktion `null` ist, gibt der Knoten ebenfalls `null` zurück (siehe Z. 21,27,33). Wichtig ist, an dieser Stelle `null` und nicht `False` zurückzugeben, da sonst der Wahrheitswert des Knotens verfälscht wird.

```
1
2   public evaluate(): boolean | null {
3       if (this instanceof Leaf){
4           // base case of recursion
5           // evaluate comparison defined in Leaf
6           let answer = this.evaluateLeaf();
7           return answer;
8       } else {
9           // inner node of recursion
```

```
10          // recursive call to children
11          let answer = this.children[0].evaluate();
12          let answer2;
13          if (this.operation !== 'not'){
14              answer2 = this.children[1].evaluate();
15          }
16          // apply boolean operation to results of children
17          // and return result
18          switch (this.operation){
19              case 'and':
20                  if (answer === null || answer2 === null){
21                      return null;
22                  }
23                  return answer && answer2;
24
25              case 'or':
26                  if (answer === null || answer2 === null){
27                      return null;
28                  }
29                  return answer || answer2;
30
31              case 'not':
32                  if (answer === null){
33                      return null;
34                  }
35                  return !answer;
36              default:
37          }
38      }
39  }
```

Listing 6.5: Rekursive Auswertung des Evaluationsbaums

Auswertung der Blätter

In eSano können Inhalte von Lektionen in Abhängigkeit von Antworten dynamisch zur Laufzeit wiederholt werden. Um Fragen trotz Wiederholungen über `position_id` eindeutig zu identifizieren, wurde `loop_index` eingeführt, sodass n-mal wiederholte

Fragen zwar die selbe `position_id`, jedoch einen unterschiedlichen `loop_index` besitzen (siehe Unterabschnitt 5.2.1 und Anforderung AC-02). Dies muss im Evaluationsbaum ebenfalls berücksichtigt werden. Ein Blatt im Evaluationsbaum speichert daher nicht nur genau eine Antwort ab, sondern besitzt ein Array `answers`. Die Antwort einer Frage mit `loop_index` n wird an die n-te Stelle von `answers` gespeichert.
Beim Aufruf von `evaluateLeaf()` wird anfangs geprüft, ob eine Antwort im Array gespeichert wurde. Falls nein, so wird `null` zurückgegeben (siehe Listing 6.6 Z. 3-5). Ist ein Antwort vorhanden und die Operation `answered`, so wird `true` zurückgegeben (siehe Z. 6-10). Ist die Operation nicht `answered`, wird das Blatt in Abhängigkeit des Fragetyps evaluiert. Hierfür wird eine Fallunterscheidung anhand `type` vorgenommen (siehe Z. 13). Je nach Fragetyp ist die Kondition eines Blattes in den Attributen `value operation` und `secondValue` definiert und wird mit den abgegebenen Antworten verglichen.

```
1  public evaluateLeaf(): boolean | null {
2      // check if there exist answers
3      if (this.answers.length === 0){
4          return null;
5      }
6      if (this.operation === 'answered'){
7          // includes TEXT_STRING and TEXT_AREA
8          // as they always have operation 'answered'
9          return true;
10     }
11     let result: boolean;
12     // check what type of question this leaf is and evaluate
13     switch (this.type) {
14         case QuestionTypes.SINGLE_CHOICE:
15             result = this.evalSingleChoiceLeaf();
16             break;
17         case QuestionTypes.MULTIPLE_CHOICE:
18             result = this.evalMultipleChoiceLeaf();
19             break;
20         case QuestionTypes.SLIDER:
21             result = this.evalSliderLeaf();
22             break;
23         case QuestionTypes.YES_NO:
```

```
24                result = this.evalYesNoLeaf();
25                break;
26            case QuestionTypes.TEXT_DATE:
27                result = this.evalDateLeaf();
28                break;
29            case QuestionTypes.TEXT_STRING ||
30                 QuestionTypes.TEXT_AREA:
31                result = false;
32                break;
33            default:
34        }
35      return result;
36  }
```

Listing 6.6: Auswertung eines Blattes

Für die Evaluation in der ersten Version gilt, dass sich eine Kondition immer auf eine Frage bezieht, unabhängig von `loop_index`. Es gilt also ein implizites `OR` zwischen Antworten der gleichen Frage. Daher wird bei der Evaluation eines Blattes über `answers` iteriert und bei der ersten wahren Kondition `true` zurückgegeben (siehe Listing 6.7 Z. 2-4). Für zukünftige Versionen kann an dieser Stelle ein zusätzlicher Flag für Knoten eingeführt werden. Dieser erlaubt es einem Editor zu definieren, dass die Kondition eines Knotens sich ausschließlich auf Fragen innerhalb eines Wiederholungsblocks bezieht, also auf Fragen mit demselben `loop_index`, und nicht über Wiederholungsblöcke hinweg.
Bei der Auswertung von Ja-Nein-Fragen enthält `value` den Wahrheitswert `true` oder `false` wird mit der Antwort des Patienten in `answer` verglichen (siehe Listing 6.7).

```
1  private evalYesNoLeaf(): boolean{
2      for (let answer of this.answers){
3          if (answer !== null && answer === this.value){
4              return true;
5          }
6      }
7      return false;
8  }
```

Listing 6.7: Auswertung einer Ja-Nein-Frage

Für Single-Choice-Fragen wird getestet, ob `answer` gleich `value` entspricht (siehe Listing 6.8).

```
1 private evalSingleChoiceLeaf(): boolean{
2     for (let answer of this.answers){
3         if (answer !== null && answer === this.value){
4             return true;
5         }
6     }
7     return false;
8 }
```

Listing 6.8: Auswertung einer Single-Choice-Frage

Für Multiple-Choice-Fragen ist eine Antwort ein Array, das eine Teilmenge der zur Verfügung stehenden Antwortmöglichkeiten enthält. Daher wird mittels `includes()` verglichen, ob `value` in `answer` enthalten ist (siehe Listing 6.9). Die Evaluation von Single- und Multiple-Choice-Fragen findet auf Basis von Labels statt, wobei ein Label eine Antwortmöglichkeit identifiziert. Dadurch werden Konditionen unabhängig von der Frontendsprache des CMS oder der App definiert bzw. ausgewertet.

```
1  // the choices of the user are inside an array
2  private evalMultipleChoiceLeaf(): boolean {
3      for (let answerArray of this.answers){
4          // check if value is subset of users choice
5          if (answer !== null && this.value in answerArray)){
6              return true;
7          }
8      }
9      return false;
10 }
```

Listing 6.9: Auswertung einer Multiple-Choice-Frage

Slider-Fragen sind Fragen, bei denen ein Patient einen Wert innerhalb eines gegebenen Intervalls auswählt. Bei der Definition von Konditionen für Slider-Fragen bestimmt ein Editor, ob die Antwort eines Patienten größer, kleiner, kleinergleich, größer, größergleich, gleich oder ungleich einer Konstante innerhalb des Intervalls ist. Bei der Evaluation wird `operation` ausgelesen und die Antwort des Patienten

mit der Konstante in `value` verglichen (siehe Listing 6.10). Ebenfalls gibt es die Operation `between`, die überprüft, ob die Antwort sich zwischen zwei Konstanten des Intervalls befindet. Die zweite Konstante wird bei der Operation aus `secondValue` ausgelesen und getestet ob `value <= answer <= secondValue` gilt.

```
1  private evalSliderLeaf(): boolean {
2  for (let answer of this.answers){
3   if (this.operation === "equal" && answer === this.value ||
4       this.operation === "notEqual" && answer !== this.value ||
5       this.operation === "less" && answer < this.value ||
6       this.operation === "lessEqual" && answer <= this.value ||
7       this.operation === "greater" && answer > this.value ||
8       this.operation === "greaterEqual" && answer >= this... ||
9       this.operation === "between" && answer >= this.value &&
10      answer <= this.secondValue)){
11        return true;
12      }
13   }}
14    return false;
15 }
```

Listing 6.10: Auswertung einer Slider-Frage

Analog läuft die Definition und Evaluation von Datumsfragen ab. Hier wird in einer Kondition ebenfalls ein Vergleichsoperator ausgewählt und geprüft, ob sich das ausgewählte Datum des Patienten beispielsweise zeitlich davor oder innerhalb einer definierten Zeitspanne befindet. Ähnlich zu Antwortmöglichkeiten in Single- und Multiple-Choice-Fragen, muss die Logik für die Vergleiche bei Datumsoperationen immer auf derselben Zeitzone operieren. `answer` und `value` enthalten ein Datum daher immer im ISO-Format, also der UTC-Zone (siehe Listing 6.11 Z. 2).

```
1  // all Dates saved as
2  // new Date(Date.UTC(chosenDate.getFullYear(),
       chosenDate.getMonth(), chosenDate.getDate(),
       chosenDate.getHours(), chosenDate.getMinutes())).
       toISOString()
3  private evalDateLeaf(): boolean {
4    let value = new Date(this.value);
5    for (let answer of this.answers){
6      let answer = new Date(answer);
```

```
7       if (this.operation === "equals" && answer === value ||
8           this.operation === "notEquals" && answer !== value ||
9           this.operation === "less" && answer < value ||
10          this.operation === "lessEqual" && answer <= value ||
11          this.operation === "greater" && answer > value ||
12          this.operation === "greaterEqual"&& answer >= value ||
13          this.operation === "between" && answer >= value &&
14          answer <= date){
15          return true;
16      }
17    }
18    return false;
19  }
```

Listing 6.11: Auswertung einer Datumsfrage

Für Textfragen kann in einer Kondition nur definiert werden, ob eine Frage beantwortet wurde oder nicht. Entsprechend enthält eine Kondition für eine Textfrage immer die Operation `answered` und wird bereits vorher abgefangen, insofern eine Antwort existiert (siehe Listing 6.6 Z. 6). Falls nicht, so wird in der Fallunterscheidung `false` zurückgegeben.

7 Fazit

Dieses Kapitel fasst die Ergebnisse zusammen und gibt einen Ausblick auf zukünftige Herausforderungen von JITAIs in eSano und im Allgemeinen.

7.1 Zusammenfassung

In dieser Masterarbeit wurde ein Konzept zur Umsetzung von JITAIs in E-Health Plattformen erarbeitet. Nach der Betrachtung verwandter Arbeiten von JITAIs wurden funktionale und nicht-funktionale Anforderungen formuliert. Die funktionalen Anforderungen lassen sich in die Kategorien *Mobile Sensing*, *EMAs*, *Adaptierung*, *Empfehlungssystem*, *Visualisierung* und *Metadaten* unterteilen und sind jeweils einer der drei Eigenschaften von JITAIs, *Just-In-Time*, *Adaptivness* und *Adherence*, zuzuordnen.
Für die Umsetzung der *Just-In-Time* Eigenschaft wurde eine mehrstufige und modulare Architektur entworfen. Sensormodule erheben Daten auf Smartphones von Patienten oder bündeln EMA- und Metadaten sowie Daten von gekoppelten externen Geräten. Analysemodule nutzen Sensormodule, um Kontexte auf höherem Abstraktionsniveau zu berechnen. Trigger überwachen Analysemodule und schalten bei Überschreitung eines Schwellwerts Interventionen frei. Ebenfalls wurden Modul-, Daten- und Steuerschnittstellen definiert. Die Architektur sieht außerdem einen Anonymisierungs- und Filter-Service vor zur besseren Kontrolle über die Datenströme.
Für die Umsetzung der *Adaptivness* Eigenschaft wurde eine Datenstruktur vorgestellt, um komplexe Konditionen für Fragen aus Lektionen definieren zu können. Es wurde ein Benachrichtungssystem entworfen, das Konditionen während der Bearbeitung von Lektionen evaluiert und Inhalte in Abhängigkeiten der Antworten ei-

nes Patienten dynamisch erscheinen, verschwinden oder wiederholen lässt. Zudem wurden eine GUI zur Definition der Konditionen mittels Endnutzer-Programmierung sowie dynamische und statische Substitutionen zur Personalisierung von Texten in Lektionen entworfen.
Für die Umsetzung der *Adherence* Eigenschaft wurde ein Empfehlungssystem vorgestellt, das Interventionen kategorisiert und eine Punktezahl für Kategorien und Interventionen berechnet. Der Algorithmus berücksichtigt Bewertungen, persönliche Präferenzen und den Kontext eines Patienten. Weiterhin wurde dargestellt, wie gesammelte Daten aus Mobile Sensing visualisiert werden können. Die Daten lassen sich anhand ihrer Zielmenge in die Kategorien *Intervalle*, *diskrete Mengen*, *GPS* und *Text* unterteilen und abbilden. Die Visualisierungen basieren auf Javascript Bibliotheken und sind dadurch in Web-Apps, PWAs und Native-Apps integrierbar.
Im praktischen Teil der Masterarbeit wurde die Plattform eSano von der KLIPS der Universität Ulm hinsichtlich der *Adaptiveness* erweitert, um Lektionen ab sofort in Abhängigkeit von Antworten des Patienten freizuschalten. Nach der Bearbeitung eines Tagebuchs erscheint einem Patienten gegebenenfalls ein Dialog, in dem er Zusatzlektionen durch An- und Abwählen freischaltet. Das Feature wird in der Studie *Better Care* verwendet, um Stimmung, Schlaf, soziale und körperliche Aktivitäten von Patienten mittels Tagebüchern abzufragen und darauf zu reagieren.

7.2 Ausblick

Das Konzept aus Kapitel 5 veranschaulicht, dass es möglich ist, Interventionen skalierbar anzubieten und trotzdem auf die individuellen Bedürfnisse eines einzelnen Patienten einzugehen. Die Möglichkeit, künstliche Intelligenz durch Module in Smartphones von Patienten zu integrieren, ist außerdem vielversprechend, um depressive Episoden oder Panikattacken zu erkennen. Die erhobenen Daten sind für E-Coaches eine wertvolle Informationsquelle, die ihnen einen datenbasierten Einblick in den Alltag eines Patienten gibt. Es gilt nun, das Konzept umzusetzen und Aspekte wie Wirksamkeit, Adhärenz und Kosteneffektivität zu evaluieren.
In Bezug auf eSano ist das neue Feature in Kombination mit Stimmungstagebüchern sehr nützlich, da zum Zeitpunkt der Bearbeitung auf spezifische Probleme

eines Patienten reagiert werden kann. Für eine Verkürzung der Reaktionszeit ist es wichtig, die Plattform bezüglich *Just-In-Time* weiterzuentwickeln. Dies ist derzeit nur bedingt möglich, da die Patientenapp als PWA vorliegt (siehe Unterabschnitt 5.1.5). Voraussetzung hierfür wäre die Einführung einer native mobile App.
Zusammenfassend zeigt die Masterarbeit, dass es Möglichkeiten gibt, die Behandlung von weit verbreiteten Krankheiten wie Depression und Angststörung mit technischen Lösungen zu unterstützen. Dies kann als ein wichtiger Schritt gewertet werden, um die eingangs erwähnten niedrigen Behandlungsraten nachhaltig zu erhöhen. Dadurch würde ein enormer Beitrag zur mentalen Gesundheit in der Bevölkerung geleistet werden.

A Appendix

Android Sensor	ab API Level	Default Mode	Rückgabewert
Acceleration	3	CONTINOUS	$(x, y, z) in m/s^2$
Ambient Temperature	14	ON_CHANGE	Umgebungs-temperatur in Grad Celsius
Gravity	9	CONTINOUS	$(x, y, z) in m/s^2$
Gyroscope	9	CONTINOUS	Winkel-geschwindigkeit um x-, y- und z-Achse
Heart Beat	9	SPECIAL_ TRIGGER	Event pro Herzschlag
Heart Rate	9	ON_CHANGE	Herzrate in bpm
Light	3	ON_CHANGE	Umgebungslicht in Lux
Linear Acceleration	9	CONTINOUS	$(x, y, z) in m/s^2$
Magnetic Field	3	CONTINOUS	Magnetfeld der Umgebung in micro-Tesla
Orientation	3	CONTINOUS	Azimuth-, Roll- und Nick-Winkel
Pressure	9	CONTINOUS	Atmospheric pressure in millibar
Proximity	3	ON_CHANGE	Entfernung in cm
Relative Humidity	20	ON_CHANGE	Rel. Luftfeuchtigkeit in %
Rotation Vector	9	CONTINOUS	Orientierung in Vektor (x, y, z)
Significant Motion	18	ONE_SHOT	Genau 1 Event wenn Lokationswechsel mit-tels Auto, Fahrrad,.. statt findet
Stationary Detect	24	SPECIAL_ TRIGGER	Event nach >5 Sek. stillstand
Step Counter	19	ON_CHANGE	#Schritte seit Reboot
Step Detector	19	SPECIAL_ TRIGGER	Event pro Schritt

Tabelle A.1: Native Android Sensoren [44, 53]

Apple Sensor	**ab API Level** (iOS, macOS, Mac Catalyst, watchOS)	**Beschreibung**	**Rückgabewert**
Accelerometer	19	Beschleunigung des Geräts für alle drei Achsen	(x,y,z) in m/s
Attitude	4.0+, 10.15+, 13.0+, 2.0+	Ausrichtung des Geräts	Rotationsmatrix, Quaternion, Euler-Winkel (Roll, Nick, Gier)
Altitude + Barometer	8.0+, -, 13.0+, 2.0+	Höhenlage, Barometer	Meter und Kilopascal
Ambient Light	14.0+, -, 14.0+, -	Misst Umgebungslicht	Lux, Farbton
Device Motion	4.0+, 10.15+, 13.0+, 2.0+	Kapselt Informationen mehrerer Sensoren in einem Objekt (Ausrichtung, Rotationsrate, Beschleunigung) eines Geräts	Objekt
Device Usage Report	14.0+, -, 14.0+, -	Nutzungsfrequenz und -dauer des Geräts, bestimmer Apps, Webseiten und Benachrichtigung	Objekt
Dyskinetic Symptom	12.0+, -, 13.0+, 5.0+	Misst ob Dyskinesie statt gefunden hat in einminütigen Intervallen	Gibt Angaben gemäß der Gleichung in Prozent zurück: percentUnlikely + percentLikely = 1.0.
Fall Detection	-, -, -, 7.2	Event bei Fall	Event
Gyroscope	4.0+, 10.15+, 13.0+, 2.0+	Einzelne Messung der Rotation des Geräts	Winkel-geschwindigkeit um x-, y- und z-Achse
Headphone Motion	14.0+, -, 14.0+, 7.0+	Misst Bewegung der Kopfhörer	k.A.
Keyboard Metrics beta	15.0+, - 15.0+, -	Wort- und Emojizähler, Stimmungserkennung (absolutistisch, verärgert, ängstlich, durcheinander, Beschäftigung mit Tot, depressiv, gesundheitlich besorgt, wenig Energie, positiv, traurig)	Integer, Enum
Magnetometer	5.0+, 10.15+, 13.0+, 2.0+	Magnetfeld der Umgebung	(x,y,z)
Message Usage Report	14.0+, -, 14.0+, -	Zählt ein-, ausgehende Nachrichten und #Kontakte in einem Zeitintervall	Integer, Zeitintervall
Motion Activity	7.0+, -, 13.0+, 2.0+	Misst Fortbewegung (stationär, laufen, renne, Auto, Fahrrad)	Event mit boolean + Startdatum
Movement Disorder Manager	-, -, -, 5.0+	Aggregiert Tremor und Diskyntik Messungen	Objekt
On Wrist State	14.0+, -, 14.0+, -	Misst ob Smartwatch getragen wird und wenn ja, ob am linken oder rechten Arm	Enum, Boolean
Pedometer	8.0+, 10.15+, 13.0+, 2.0+	Schrittzähler, Gelaufene Meter, Stockwerke, Pace	NSNumber
Phone Usage Report	14.0+, 14.0+	Zählt ein-, ausgehende Anrufe und #Kontakte über Zeitintervall	Integer, Zeitintervall
Rotation Rate	14.0+, 14.0+	Rotationsmessungen	Event pro Schritt
Speech Recognition Metadata	14.5+, 11.3+, 14.5+, -	Misst Live Sprache (Länge und Pausen von Gesprächen, Stimmlage, Stimmung, Timing)	Zeitintervalle, k.A. zur Stimmlage und Stimmung
Siri Speech Metrics	14.5+, 11.3+, 14.5+, -	Misst Sprache bei Siri Befehlen (Länge und Pausen von Gesprächen, Stimmlage, Stimmung, Timing)	Zeitintervalle, k.A. zur Stimmlage und Stimmung
Telephony Speech Metrics	14.5+, 11.3+, 14.5+, -	Misst Sprache bei Telefonaten (Länge und Pausen von Gesprächen, Stimmlage, Stimmung, Timing)	Zeitintervalle, k.A. zur Stimmlage und Stimmung
Tremor	12.0+, -, 13.0+, 5.0+	Misst ob ein Tremor begonnen hat und wie stark in einminütigen Intervallen	(nicht, leicht, mild, moderat, stark) in Prozent
Visits	14.0+, -, 14.0+, -	Aggregiert Informationen über regelmäßig besuchte Lokationen (Ankunft, Abfahrt, Entfernung von Zuhause, Typ(Gym, Zuhause, Schule, Arbeit)), Fortschrittsmessung in täglicher Routine	Objekt

Tabelle A.2: Apple Sensoren aus den Frameworks Core Motion und SensorKit [45, 54]

```
// import moment.js, jquery.js, chart.js<style>
// define canvas, canvas2 and canvas3

// for likert, number and range based questions
var config = {
  type: 'bar',
  data: {
    datasets: [
      {
        label: 'Wie geht es Ihnen heute?',
        data: [
          {
            x: '23/08/2021',
            y: 5,
          },
          {
            x: '24/08/2021',
            y: 4,
          },
          {
            x: '25/08/2021',
            y: 2,
          },
        ],
        fill: false,
        backgroundColor: 'rgba(255, 99, 132, 0.2)',
        borderColor: 'rgb(255, 99, 132)',
        borderWidth: 1,
      },
      {
        label: 'Wie haben Sie heute geschlafen?',
        data: [
          {
            x: '23/08/2021',
            y: 3,
          },
          {
            x: '24/08/2021',
            y: 1,
```

```
            },
            {
              x: '25/08/2021',
              y: 5,
            },
          ],
          fill: false,
          backgroundColor: 'rgba(54, 162, 235, 0.2)',
          borderColor: 'rgb(54, 162, 235)',
          borderWidth: 1,
        },
      ],
    },
    options: {
      responsive: true,
      plugins: {
        title: {
          display: true,
          text: 'EMA Data of patient XY',
        },
      },
      scales: {
        xAxes: [
          {
            type: 'time',
            time: {
              format: 'DD/MM/YYYY',
              tooltipFormat: 'll',
            },
            scaleLabel: {
              display: true,
              labelString: 'Date',
            },
          },
        ],
        yAxes: [
          {
            ticks: {
              stepSize: 1,
            },
```

```
80            scaleLabel: {
81              display: true,
82              labelString: 'value',
83            },
84          },
85        ],
86      },
87    },
88  };
89
90  // for set based questions like single choice and multiple
      choice
91  var config2 = {
92    type: 'line',
93    data: {
94      datasets: [
95        {
96          label: 'Welche Aktivitäten haben Sie heute gemacht?',
97          data: [
98            { x: '23/08/2021', y: 'Arbeit' },
99            { x: '23/08/2021', y: 'Sport' },
100           { x: '24/08/2021', y: 'Arbeit' },
101           { x: '25/08/2021', y: 'Musik' },
102         ],
103         borderWidth: 0,
104         pointRadius: 5,
105         backgroundColor: 'rgb(255, 205, 86)',
106       },
107     ],
108     yLabels: ['Musik', 'Sport', 'Arbeit'],
109     xLabels: ['23/08/2021', '24/08/2021', '25/08/2021'],
110   },
111   options: {
112     scales: {
113       x: {},
114       y: {type: 'category'},
115     },
116   },
117 };
118
```

```
119   // for text based questions
120 var config3 = {
121   type: 'line',
122   data: {
123     labels: ['23/08/2021', '24/08/2021'],
124     datasets: [
125       {
126         label: 'Was ging Ihnen heute durch den Kopf?',
127         data: [
128           {
129             x: '23/08/2021',
130             y: 0,
131             answer: 'Heute habe ...',
132           },
133           {
134             x: '24/08/2021',
135             y: 0,
136             answer: 'Heute habe ich ...',
137           },
138           {
139             x: '25/08/2021',
140             y: 0,
141             answer: 'Heute habe ich ...',
142           },
143         ],
144         borderWidth: 0,
145         pointRadius: 5,
146         backgroundColor: 'rgb(75, 192, 192)',
147       },
148     ],
149   },
150   options: {
151     plugins: {
152       tooltip: {
153         caretPadding: 20,
154         caretSize: 10,
155         callbacks: {
156           // Tooltip customize
157           label: function (context) {
158             return context.dataset.data[0].answer;
```

```
159             },
160           },
161         },
162       },
163       layout: {
164         padding: {
165           bottom: 150,
166         }
167       },
168       responsive: true,
169       gridLines: {
170         display: false,
171       },
172       scales: {
173         x: {
174           grid: {
175             display: false,
176           },
177         },
178         y: {
179           display: false,
180           beginAtZero: true,
181           ticks: {
182             display: false,
183           },
184           grid: {
185             display: false,
186           },
187         },
188       },
189     },
190   };
191
192   window.onload = function () {
193     var ctx = document.getElementById('canvas').getContext('2d')
          ;
194     window.myLine = new Chart(ctx, config);
195     // analog for canvas2 and canvas3
196     ...
197   };
```

Listing A.1: Code zur Visualisierung der Datenströme mittels Chart.js 3.5.1

```
// import jquery, leafletjs, heatmap.js, SliderControl.js

// points of interest
var home = {lat: 48.397539, lng: 10.009817}
var work = {lat: 48.423281, lng: 9.954603}
var gym = {lat: 48.395003, lng: 10.015921}

var addressPoints = [
[home.lat, home.lng, "500"],
[work.lat, work.lng, "500"],
[gym.lat, gym.lng, "100"],
// noise
...
]

// preprocessing for heatmap and poi layer
var min = 1;
var max = 500;
var applyOffset = 0
var data = addressPoints.map(function (p) {
  var offset = {lat: -0.5, lng: 0}
  applyOffset += 1
  return {
    lat: applyOffset > 3 ? p[0] + offset.lat : p[0],
    lng:  applyOffset > 3 ? p[1] + offset.lng : p[1],
    count: parseInt(p[2]) || 0 };
  });
applyOffset = 0
var timeOffset = 0
var dataGeoJson = {
"type": "FeatureCollection",
"features" : addressPoints.map(function (p) {
    var offset = {lat: -0.5, lng: 0}
    applyOffset += 1
    timeOffset += 1
    return {
              "type": "Feature",
```

```
                "properties": {
                  "time" : 1632819729 + timeOffset
                },
                "geometry": {
                  "type": "Point",
                  "coordinates": [
                    applyOffset > 3 ? p[1] + offset.lng : p[1],
                    applyOffset > 3 ? p[0] + offset.lat : p[0]
                  ]
                }
    }
})};

var map = new L.Map('map').setView([home.lat, home.lng], 12);
L.tileLayer(
  'http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
  {maxZoom: 19}
).addTo(map);

var heatmapLayer = new HeatmapOverlay({
  "radius": 25,
  "maxOpacity": .5,
  "scaleRadius": false,
  "useLocalExtrema": true,
  "latField": 'lat',
  "lngField": 'lng',
  "valueField": 'count',
  "blur": 0.95,
  "gradient": {
    '.15': 'yellow',
    '.8': 'orange',
    '.95': 'red'
  }
});

heatmapLayer.setData({max: max, data: data});

var homeMarker = L.marker([home.lat, home.lng])
  .bindPopup("<b>Zuhause</b><br>Du hast 40% der Zeit zuhause
     verbracht")
```

```
77    .openPopup();
78  var workMarker = L.marker([work.lat, work.lng])
79    .bindPopup("<b>Work</b><br>Du hast 40% der Zeit auf der
          Arbeit verbracht")
80    .openPopup();
81  var gymMarker = L.marker([gym.lat, gym.lng])
82    .bindPopup("<b>Gym</b><br>Du hast 5% der Zeit im Gym
          verbracht")
83    .openPopup();
84
85  var POILayer = L.layerGroup([homeMarker, workMarker, gymMarker
      ])
86
87
88  // Layer of points you can trace with a time slider
89  var pointLayer = L.geoJson(dataGeoJson, {
90    pointToLayer: function (feature, latlng) {
91      return L.circleMarker(latlng, {
92          radius: 3,
93          fillColor: "#ff7800",
94          color: "#000",
95          weight: 1,
96          opacity: 1,
97          fillOpacity: 0.8
98      });
99    }
100 });
101
102 // time slider
103 var sliderControl = L.control.sliderControl({
104   position: "topright",
105   layer: pointLayer,
106   timeAttribute: "time",
107   isEpoch: true,
108   sameDate: true,
109   follow: 1,
110   alwaysShowDate: true
111 });
112
113 map.addControl(sliderControl);
```

```
114 sliderControl.startSlider();
115
116 // add overlays
117 var overlays = {"Heat": heatmapLayer, "POIs": POILayer};
118 L.control.layers("", overlays, {collapsed: false}).addTo(map)
```

Listing A.2: Code zur Visualisierung von Geodaten mittels Openstreetmap, Leaflet.js, Heatmap.js und LeafletSlider.js

Literatur

[1] Hannah Ritchie Saloni Dattani und Max Roser. „Mental Health“. In: *Our World in Data* (2021). URL: https://ourworldindata.org/mental-health.

[2] Philip S Wang u. a. „Use of mental health services for anxiety, mood, and substance disorders in 17 countries in the WHO world mental health surveys“. In: *The Lancet* 370.9590 (2007), S. 841–850.

[3] Laura Helena Andrade u. a. „Barriers to mental health treatment: results from the WHO World Mental Health surveys“. In: *Psychological medicine* 44.6 (2014), S. 1303–1317.

[4] Gavin Andrews u. a. „Computer therapy for the anxiety and depressive disorders is effective, acceptable and practical health care: a meta-analysis“. In: *PloS one* 5.10 (2010), e13196.

[5] Gerhard Andersson und Nickolai Titov. „Advantages and limitations of Internet-based interventions for common mental disorders“. In: *World Psychiatry* 13.1 (2014), S. 4–11.

[6] Koenigbauer Josephine u. a. „Internet-and mobile-based depression interventions for people with diagnosed depression: a systematic review and meta-analysis“. In: *Journal of affective disorders* 223 (2017), S. 28–40.

[7] E Hedman u. a. „Cost-effectiveness and long-term effectiveness of internet-based cognitive behaviour therapy for severe health anxiety“. In: *Psychological medicine* 43.2 (2013), S. 363–374.

[8] Ensar Arif Sağbaş, Serdar Korukoglu und Serkan Balli. „Stress detection via keyboard typing behaviors by using smartphone sensors and machine learning techniques“. In: *Journal of medical systems* 44.4 (2020), S. 1–12.

[9] Bhanusree Yalamanchili u. a. „Real-time Acoustic based Depression Detection using Machine Learning Techniques“. In: *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. IEEE. 2020, S. 1–6.

[10] Liyuan Wang und Lynn Carol Miller. „Just-in-the-moment adaptive interventions (JITAI): a meta-analytical review“. In: *Health communication* 35.12 (2020), S. 1531–1544.

[11] Pia von Blanckenburg und Nico Leppin. „Psychological interventions in palliative care“. In: *Current opinion in psychiatry* 31.5 (2018), S. 389–395.

[12] Harald Baumeister, Jiaxi Lin und David Daniel Ebert. „Internet-und mobilebasierte Ansätze: Psychosoziale Diagnostik und Behandlung in der medizinischen Rehabilitation (Leitthema)“. In: *Bundesgesundheitsblatt, Gesundheitsforschung, Gesundheitsschutz* 60.4 (2017), S. 436–444.

[13] Azy Barak, Britt Klein und Judith G Proudfoot. „Defining internet-supported therapeutic interventions“. In: *Annals of behavioral medicine* 38.1 (2009), S. 4–17.

[14] Harald Baumeister u. a. „The impact of guidance on Internet-based mental health interventions—A systematic review“. In: *Internet interventions* 1.4 (2014), S. 205–215.

[15] Pim Cuijpers u. a. „Recruiting participants for interventions to prevent the onset of depressive disorders: Possibile ways to increase participation rates“. In: *BMC Health Services Research* 10.1 (2010), S. 1–6.

[16] Tara Donker u. a. „Economic evaluations of Internet interventions for mental health: a systematic review“. In: *Psychological medicine* 45.16 (2015), S. 3357–3376.

[17] Nickolai Titov u. a. „Clinical and cost-effectiveness of therapist-guided internet-delivered cognitive behavior therapy for older adults with symptoms of depression: a randomized controlled trial“. In: *Behavior therapy* 46.2 (2015), S. 193–205.

[18] Gregory D Abowd u. a. „Towards a better understanding of context and context-awareness“. In: *International symposium on handheld and ubiquitous computing*. Springer. 1999, S. 304–307.

[19] Upkar Varshney. „Context-awareness in Healthcare“. In: *Pervasive Healthcare Computing*. Springer, 2009, S. 231–257.

[20] Saul Shiffman, Arthur A Stone und Michael R Hufford. „Ecological momentary assessment“. In: *Annu. Rev. Clin. Psychol.* 4 (2008), S. 1–32.

[21] Robin Kraft u. a. „eSano – An eHealth Platform for Internet- and Mobile-based Interventions“. In: *2021 43rd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*. 2021, S. 1997–2002.

[22] eSano E-Health Team. *Über uns*. URL: https://esano.klips-ulm.de/de/uber-uns/ (besucht am 23. Juli 2021).

[23] Inbal Nahum-Shani u. a. „Just-in-time adaptive interventions (JITAIs) in mobile health: key components and design principles for ongoing health behavior support“. In: *Annals of Behavioral Medicine* 52.6 (2018), S. 446–462.

[24] Felix Naughton. „Delivering “Just-In-Time” smoking cessation support via mobile phones: current knowledge and future directions“. In: *Nicotine & Tobacco Research* 19.3 (2017), S. 379–383.

[25] G Alan Marlatt und Dennis M Donovan. *Relapse prevention: Maintenance strategies in the treatment of addictive behaviors*. Guilford press, 2005.

[26] Predrag Klasnja u. a. „Microrandomized trials: An experimental design for developing just-in-time adaptive interventions.“ In: *Health Psychology* 34.S (2015), S. 1220.

[27] Mashfiqui Rabbi u. a. „Toward increasing engagement in substance use data collection: development of the Substance Abuse Research Assistant app and protocol for a microrandomized trial using adolescents and emerging adults“. In: *JMIR research protocols* 7.7 (2018), e166.

[28] Predrag Klasnja u. a. „Efficacy of contextually tailored suggestions for physical activity: a micro-randomized optimization trial of HeartSteps“. In: *Annals of Behavioral Medicine* 53.6 (2019), S. 573–582.

[29] Lauren Bell u. a. „Notifications to improve engagement with an alcohol reduction app: protocol for a micro-randomized trial“. In: *JMIR research protocols* 9.8 (2020), e18690.

[30] Samuel L Battalio u. a. „Sense2Stop: A micro-randomized trial using wearable sensors to optimize a just-in-time-adaptive stress management intervention for smoking relapse prevention“. In: *Contemporary Clinical Trials* 109 (2021), S. 106534.

[31] Joshua M Smyth und Kristin E Heron. „Is providing mobile interventions"just-in-time"helpful? An experimental proof of concept study of just-in-time intervention for stress management“. In: *2016 IEEE Wireless Health (WH)*. IEEE. 2016, S. 1–7.

[32] Linda M Collins, Susan A Murphy und Victor Strecher. „The multiphase optimization strategy (MOST) and the sequential multiple assignment randomized trial (SMART): new methods for more potent eHealth interventions“. In: *American journal of preventive medicine* 32.5 (2007), S112–S118.

[33] Inbal Nahum-Shani u. a. „Q-learning: a data analysis method for constructing adaptive interventions.“ In: *Psychological methods* 17.4 (2012), S. 478.

[34] Evan M Forman u. a. „OnTrack: development and feasibility of a smartphone app designed to predict and prevent dietary lapses“. In: *Translational behavioral medicine* 9.2 (2019), S. 236–245.

[35] Christian Jules Cerrada u. a. „Development of a just-in-time adaptive intervention for smoking cessation among Korean American emerging adults“. In: *International journal of behavioral medicine* 24.5 (2017), S. 665–672.

[36] Minddistrict. *The CMS: keep control of your ehealth content*. URL: https://www.minddistrict.com/ehealth-platform/cms (besucht am 21. Juli 2021).

[37] Sangwon Bae u. a. „Mobile phone sensors and supervised machine learning to identify alcohol use events in young adults: Implications for just-in-time adaptive interventions“. In: *Addictive behaviors* 83 (2018), S. 42–47.

[38] John Rooksby, Alistair Morrison und Dave Murray-Rust. „Student perspectives on digital phenotyping: The acceptability of using smartphone data to assess mental health“. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 2019, S. 1–14.

[39] Denzil Ferreira. *AWARE - Open-source Context Instrumentation Framework For Everyone*. URL: https://awareframework.com/ (besucht am 6. Juli 2021).

[40] Denzil Ferreira. *AWARE Client*. URL: https://github.com/denzilferreira/aware-client (besucht am 6. Juli 2021).

[41] Denzil Ferreira. *AWARE Server Dashboard*. URL: https://github.com/denzilferreira/aware-server (besucht am 6. Juli 2021).

[42] Peter Hösch. „Entwurf and Implementierung eines Frameworks fuer Mobile Context-Awareness im Bereich eHealth“. 2020. URL: http://dbis.eprints.uni-ulm.de/1983/.

[43] Peter Hoesch. *Context Awareness Framework*. URL: https://gitlab.klips-ulm.de/mobilesensing/context-awareness-framework (besucht am 11. Aug. 2021).

[44] Google LLC. *Android Developer Documentation - Reporting modes*. URL: https://source.android.com/devices/sensors/report-modes (besucht am 31. Aug. 2021).

[45] Apple Inc. *Apple Developer Documentation - The object for starting and managing motion services*. URL: https://developer.apple.com/documentation/coremotion/cmmotionmanager (besucht am 31. Aug. 2021).

[46] Mozilla Foundation. *Mozilla Developer Network - Using the Notifications API*. URL: https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API/Using_the_Notifications_API#browser_compatibility (besucht am 25. Aug. 2021).

[47] World Wide Web Consortium (W3C). *Service Workers Nightly*. URL: https://w3c.github.io/ServiceWorker/#service-worker-lifetime (besucht am 27. Aug. 2021).

[48] Mozilla Foundation. URL: https://developer.mozilla.org/en-US/docs/Web/API/Sensor_APIs#browser_compatibility (besucht am 20. Okt. 2021).

[49] Heise Medien GmbH & Co. KG. URL: https://www.heise.de/news/Apps-im-Browser-Apple-stellt-sich-gegen-16-Web-Schnittstellen-4799813.html (besucht am 20. Okt. 2021).

[50] Thomas Steiner, Peter Beverloo und Richard Knoll. URL: https://web.dev/notification-triggers/ (besucht am 27. Aug. 2021).

[51] Lena Stenzel. „Design and Implementation of a Module to Enable Conditional Content in Internet and Mobile-based Interventions“. 2019. URL: http://dbis.eprints.uni-ulm.de/1866/.

[52] Fabian Wahle u. a. „Mobile sensing and support for people with depression: a pilot trial in the wild“. In: *JMIR mHealth and uHealth* 4.3 (2016), e5960.

[53] Google LLC. *Android Developer Documentation - Sensors*. URL: https://developer.android.com/guide/topics/sensors (besucht am 31. Aug. 2021).

[54] Apple Inc. *Apple Developer Documentation - The sensors an app can read.* URL: https://developer.apple.com/documentation/sensorkit/srsensor#3681604 (besucht am 31. Aug. 2021).

Name: Johannes Ziegler Matrikelnummer: 932195

Erklärung

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

Ulm, den .....

Johannes Ziegler