ulm university universität ulm

Universität Ulm | 89069 Ulm | Germany

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

Konzeption und Realisierung einer Smartwatch-Schnittstelle für ein Mobile Crowd Sensing System am Beispiel von TrackYourTinnitus

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Sebastian Fuchs
sebastian.fuchs@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Dr. Rüdiger Pryss

2017

Fassung 7. September 2017

Satz: PDF-LaTeX 2$_\varepsilon$

Kurzfassung

Smartphones sind inzwischen ein fester Bestandteil der heutigen Gesellschaft geworden. Für alles erdenkliche gibt es inzwischen kleine Alltagshelfer, auch Apps genannt.
Die Bandbreite umfasst hier alles, von der Unterhaltung bis zur Gesundheit. Hierbei werden nicht nur Daten zu klinischen Studien erfasst. Die erhobenen Daten werden auch analysiert und ausgewertet. Speziell für Patienten, die unter Tinnitus leiden ist eine klassische Auswertung schwer, da diese oftmals nicht direkt in einer Episode stattfinden. Aus diesem Grund entwickelt die Universität Ulm ein Projekt welches es Patienten ermöglicht die Erkrankung direkt, mittels Smart Devices, aufzuzeichnen. Dies erlaubt den Patienten einen besseren Umgang mit ihrer Krankheit.

Da sich die Gruppe der Smart Devices immer weiterentwickelt und vergrößert, müssen Möglichkeiten erschlossen werden auch diese neuen Geräte miteinzubeziehen. Es sollen nun Varianten gefunden werden besagte Smartwatches in Mobile Crowd Sensing Systeme einzugliedern. Hierzu wurde untersucht, wie eine bestehende iPhone Applikation um eine iWatch Applikation erweitert werden kann. Dabei wurden verschiedene Möglichkeiten evaluiert.
Die hierbei gefundenen Ergebnisse wurden abschließend praktisch umgesetzt.

Danksagung

An dieser Stelle möchte ich mich bei allen beteiligten Personen bedanken, die durch ihre fachliche und persönliche Meinung zum Gelingen dieser Bachelorarbeit beigetragen haben.
Ein besonderer Dank gebührt hierbei Dr. Rüdiger Pryss, bei welchem ich jederzeit eine herausragende Betreuung genießen konnte.
Weiterhin möchte ich mich bei Prof. Dr. Manfred Reichert vom Institut für Datenbanken und Informationssysteme, für die Genehmigung der Arbeit bedanken.

Ein ganz besonderer Dank gebührt meiner Mutter, Ella Fuchs, die immer an mich geglaubt hat und mich immer unterstützte wo sie nur konnte.

Nicht weniger Dank gebührt meiner Partnerin, Johanna-Marie Loesewitz, die mich mit besonderer Hingabe unterstützte wo sie nur konnte.

Ein Dank gebührt auch dem Rest meiner Familie und auch meinen Freunden.
Alle genannten haben mich besonders in dieser Zeit immer tatkräftig unterstützt.

Ein Dank gebührt auch auch meinen Arbeitskollegen, welche mich in dieser turbulenten Zeit vertreten haben.

Ein ganz besonderer Dank gebührt Rick und Morty, welche mir gezeigt haben dass man notfalls in einer anderen Dimension nochmals eine Chance bekommen kann.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Problemstellung 2

1.2 Zielsetzung 3

1.3 Struktur der Arbeit 3

2 Grundlagen **5**

2.1 Tinnitus 5

2.1.1 Behandlungsmethoden 6

2.2 Smart Devices 6

2.3 Mobile Crowd Sensoring und Track Your Tinnitus 7

2.4 Apple Watch 8

2.4.1 Technologie 9

2.4.2 Vergleichbare Apps 10

3 Verwandte Arbeiten **11**

3.1 Mobiles Framework zur Unterstützung von Tinnitus Patienten 11

3.1.1 Aufbau des TYT Projekts 12

3.1.2 Aufbau der Website 13

3.1.3 Die Smartphone Apps des TYT 14

3.2 Using Wearables in the Context of Chronic Disorders 14

3.3 Prototyp einer iWatch Umsetzung 14

3.3.1 Analyse des Prototyps 15

3.3.2 Analyse der Studie 16

3.4 Fazit 16

4 Anforderungsanalyse **19**

4.1 Anwendungsfälle 19

4.2 Funktionale Anforderungen 21

4.3 Nicht-funktionale Anforderungen 22

5 Theorie **23**
5.1 Xcode . . . 23
5.1.1 Aufbau . . . 23
5.2 WatchKit . . . 24
5.3 WatchKit und iOS . . . 26
5.3.1 Unterschiede im Versionsverlauf der iWatch . . . 26
5.4 iPhone und die iWatch . . . 27
5.4.1 Verbindung mittels CoreData . . . 27
5.4.2 Verbindung mittels WatchConnectivity . . . 27

6 Praktische Umsetzung **29**
6.1 Wahl des Fragebogen . . . 29
6.2 Xcode und Track Your Tinnitus Projekt . . . 29
6.2.1 Podfiles . . . 30
6.3 Zusammenfügen der Projekte . . . 31
6.3.1 Einfügen eines WatchKit . . . 32
6.4 Verknüpfung der beiden Quelltexte . . . 33
6.5 Verbindung zwischen iPhone und iWatch . . . 34
6.5.1 Verbindung mit CoreData . . . 35
6.5.2 Verbindung mit WatchConnectivity . . . 35
6.5.3 Modellierung des Programmablaufs . . . 39

7 Führung durch die App **41**

8 Anforderungsabgleich **51**
8.1 Funktionale Anforderungen . . . 52
8.2 Nicht-funktionale Anforderungen . . . 53

9 Zusammenfassung **55**
9.1 Fazit . . . 55
9.2 Ausblick . . . 56

1

Einleitung

In der Bundesrepublik Deutschland leiden, Schätzungen zufolge, rund drei Millionen Menschen an Ohrgeräuschen, die allgemein hin auch als Tinnitus bezeichnet werden[1]. Als Tinnitus wird ein Symptom bezeichnet, bei welchem der betroffene Patient ein Ohrgeräusch wahrnimmt, welches jedoch keiner äußeren Quelle zugeordnet werden kann. Inzwischen berichtet jeder siebte in Deutschland lebende Mensch von einer Episode oder aber einer Erkrankung. Es ist deshalb umso wichtiger geworden, passende und ebenso funktionierende Behandlungsmethoden zu entwickeln. Denn gerade mit der inzwischen omnipräsenten Belastung der Gehörgänge, ist davon auszugehen, dass die Zahlen der Tinnituserkrankungen weiterhin steigen werden. Eine Behandlung oder gar eine Prävention ist für die Krankheit Tinnitus schwer zu bestimmen. Oft wird zwar eine zu laute Lärmquelle einer Tinnituserkrankung zugeordnet, konträr dazu kann der Tinnitus aber mittels des Einsatzes von Musik gebessert werden[2].
An oberster Stelle der Behandlungsmethoden gegen den Tinnitus steht oftmals das Ziel, den Patienten die Selbstbeobachtung und Wahrnehmung ihres Tinnitus zu lehren[3]. Selbstbeobachtung soll dem Patienten dabei helfen den Tinnitus oftmals als weniger störend wahrzunehmen[4] und dadurch die Alltagsfunktionalität wieder herzustellen.
Hierzu ist es nötig dem Patienten eine Möglichkeit zur Verfügung zu stellen, welche es ihm erlaubt den Tinnitus genau zu verfolgen. Meistens werden hierbei Fragebögen für die Erfassung der Daten verwendet[5].
Aussagen, da die Erfassung erst weit später als die zugehörige Episode erfolgt.

Die Aussagekraft eben solcher klassischen Fragebögen ist meist eher gering, da die Erhebung wie erwähnt retrospektiv erfolgt und somit die Details eher verschwindent sind.
Es würde sich daher die mobile Technologie anbieten, welche es erlauben würde solche Fragenbögen direkt auszufüllen[5]. Da außerdem die Zahl der Smartphonenutzer weiterhin rasant steigt[6], bietet es sich an in diesem Bereich *Mobile Crowd Sensing* Plattformen zu verwenden. Zwar werden Smartphones oftmals als gesundheitsschädliche Begleiter beschrieben[7], jedoch gibt es auch genügend Arbeiten welche aufzeigen, dass man die Geräte ebenso als *Well Phone* verwenden kann[8].
Doch neben Smartphones und Tablets gibt es noch weitere Innovationen im Markt der mobilen Geräte. Mit der Erweiterung besagter Klasse der mobilen Geräte um die *Smartwatches*, welche mit dem Smartphone interagieren, erfolgt eine nützlichere Erfassung der Daten des Nutzers.

1.1 Problemstellung

Bei der Behandlung des Tinnitus steht an oberster Stelle oftmals das Ziel dem Patienten eine Selbstbeobachtung und somit auch eine Wahrnehmung des Tinnitus zu lehren[3]. Die Selbstbeobachtung soll dem Patienten vor allem dazu dienen den Gedanken des Tinnitus als konsequente Störung loszulassen[4]. Da diese streng strukturiert mittels Fragebögen in begrenzten Zeiträumen des Tages stattfinden soll, könnte dies auch mittels einer Mobile Crowd Sensing Lösung realisiert werden. Hierbei existieren zwar bereits konkrete Umsetzungen für das Smartphone, jedoch speziell für Smartwatches nur wenige. Für die iWatch von Apple gibt es zwar einen Prototypen[9], jedoch ist dieser noch nicht praktisch umgesetzt worden.
Aus den zuvor getroffenen Schlüssen lässt sich nun folgende Problemstellung beschreiben. Zum einen gibt es immer mehr Menschen welche über eine Tinnituserkrankung klagen. Zum anderen gibt es die mobilen Geräte, welche neue Möglichkeiten im Bereich des Mobile Crowd Sensing schaffen.
Beides miteinander zu verbinden erscheint somit logisch.

1.2 Zielsetzung

Das Ziel dieser Arbeit soll es sein, Patienten mit einer Tinnituserkrankung eine Möglichkeit zu bieten, Episoden ihrer Erkrankung besser zu verfolgen.
Nachdem sogenannte *Watchables*, als Erweiterung eines Smartphones, eine immer höhere Nachfrage generieren, ist es von Interesse eine bereits existente Applikation so zu erweitern, dass die Nutzung auf einer Smartwatch möglich wird.
Der Vorteil einer Smartwatch liegt an der vergleichsweise einfacheren Bedienoberfläche und der leichteren Handhabbarkeit. Durch die schnellere Verfügbarkeit einer Smartwatch im Vergleich zu einem Smartphone soll es dem Patienten so möglich gemacht werden besseren Zugang zu den zur Verfügung gestellten Mitteln zu erhalten.

1.3 Struktur der Arbeit

Die Arbeit ist wie folgt strukturiert.
Zunächst soll in den Grundlagen das Krankheitsbild des Tinnitus genauer erläutert werden. Ebenso werden die technischen Grundlagen erläutert.
Anschließend werden verwandte Arbeiten vorgestellt und diskutiert. In der folgenden Anforderungsanalyse wird festgelegt, was notwendig ist um die dargestellten Ziele der Arbeit zu erreichen.
Darauf folgt eine Vorstellung der Umsetzungmöglichkeiten in theoretischer Form, ehe die praktische Umsetzung folgt.
Danach reiht sich eine genaue Vorstellung der Applikation ein, Schritt für Schritt. Mittels eines Anforderungsabgleichs, in welchem die Ziele ob ihres erreichens überprüft werden, wird die Arbeit dann beendet.

2

Grundlagen

Nachfolgendes Kapitel stellt kurz die Tinnituserkrankung und mögliche Behandlungsmethoden vor. Mittels *Track Your Tinnitus* wird eine *Mobile Crowd Sensing* Methode vorgestellt. Außerdem werden verwandte Applikationen, welche bereits für die iWatch und das iPhone verfügbar sind, untersucht.
Schließlich werden noch die Applikation *Track Your Tinnitus* und die *iWatch* von Apple vorgestellt.

2.1 Tinnitus

Zunächst wird genauer auf das Krankheitsbild des Tinnitus eingegangen. Bei der Krankheit Tinnitus handelt es sich um eine Störung der Hörfunktion, welche kurzzeitig oder dauerhaft beim Patienten auftritt. Bei einem kurzzeitigen Auftreten von bis zu drei Monaten spricht man von einem *akuten Tinnitus*, während ein länger andauerndes Auftreten als *chronischer Tinnitus* bezeichnet wird. Man unterscheidet in der Diagnose zwischen einem *subjektiven* und einem *objektiven* Tinnitus. Während einem objektiven Tinnitus immer eine Grunderkrankung als Schallquelle zugeordnet wird, ist beim subjektiven Tinnitus, neben einer körperlichen Ursache, auch ein psychosomatischer Auslöser denkbar[10]. Bei einem objektiven Tinnitus können die Störungen, da von einer Erkrankung ausgehend, gemessen und auch untersucht werden.

Für beide Arten des Tinnitus gibt es verschiedene Ursachen. So können bei einem subjektiven Tinnitus unter anderem ein Hörsturz, Fremdkörper im Gehörgang oder auch Herz-Kreislauf-Krankheiten als Auslöser[11] fungieren. Bei einem objektiven Tinnitus

wiederrum können Gefäßmissbildungen oder sogar Tumore im Mittelohr einen Tinnitus verursachen. Bei einem sehr hohen Teil der Betroffenen liegt jedoch ein subjektiver Tinnitus vor[11]. Nach einer Untersuchung der DTL, der deutschen Tinnitus-Liga, sind in Deutschland rund 2,9 Millionen Erwachsene an spontan und lang andauernden Ohrgeräuschen erkrankt. Aus besagter Studie ergibt sich eine Zahl von ungefähr 270.000 Tinnitus-Neuerkrankungen pro Jahr[12]. Die Intensität des Tinnitus variiert hierbei über den Tag verteilt. So haben sowohl die Umgebungslautstärke, als auch der Stresslevel des Patienten starken Einfluss auf die jeweilige Wahrnehmung des Tinnitus[13]. Wie jede andere Krankheit sollte auch ein Tinnitus nicht unbehandelt bleiben.

2.1.1 Behandlungsmethoden

Das Ziel einer Tinnitusbehandlung besteht darin, den Patienten im Umgang mit dem Tinnitus zu schulen.
Diese Herangehensweise wird als *Tinnitus-Counseling* bezeichnet. Um dem Patienten hierbei den Umgang mit dem Tinnitus im Alltag zu ermöglichen, legt die Behandlung vor allem viel Wert auf das gezielte Aufarbeiten, Überprüfen und das Bewerten des Sachverhaltes.[14].
Da der Grad eines Tinnitus über den Tag verteilt stark variiert[13], wird oftmals viel Wert auf die Erfassung der Tagesform des Patienten gelegt. Für eine solche Aufzeichnung werden für gewöhnlich klassische Fragebögen verwendet[15]. In solchen klassischen Erfassungsmethoden werden jedoch weder Umweltfaktoren noch Stress des Patienten mit einbezogen[16]. Des weiteren sind besagte Fragebögen meist retrospektiv, da sie für gewöhnlich nicht direkt die aktuelle Stimmungslage des Patienten messen.

2.2 Smart Devices

Inzwischen ist es für viele Nutzer üblich das Smartphone, oder sonstige *Smart Devices*, für alltägliche Problemstellungen zu verwenden. Sei es das Verwenden des Smartphones als Begleiter für den alltäglichen Einkauf, oder aber als gesundheitlicher Helfer. Das Smartphone wird nicht nur als Begleiter für den Fitnesswandel verwendet, sondern

auch oft als sogenanntes Well Phone[8]. Es bietet sich daher an, auch die erwähnten Fragebögen zum Tinnitus zu digitalisieren. Dies erlaubt es dem Nutzer den Fragebogen auch unabhängig vom jeweiligen Ort auszufüllen. Eine leichtere Datenauswertung ist außerdem dadurch garantiert, dass die Daten des Fragebogen bereits digital vorliegen und nicht noch aufwendig manuell eingepflegt werden müssen. Hierbei sei das bereits existente Fragebogen-Framework der Universität Ulm, *QuestionSys*, erwähnt. Das Ziel ist es hierbei, den klassischen Studien entgegenzuwirken, welche mit Stift und Papier arbeiten und somit auch einen massiven Arbeitsaufwand für die Studienteilnehmer bedeuten[17]. Ein weiterer Vorteil von QuestionSys besteht darin, dass die Daten direkt digital vorliegen und ausgewertet werden können. Eine manuelle Eingabe der Daten entfällt somit.
QuestionSys besteht hierbei auf folgenden Komponenten. Zum Erstellen der Fragebögen wird der *Fragebogen Konfigurator* verwendet. Eine *Integration* von *Smart Devices* ist ebenso vorgesehen. Es wird außerdem eine *Datenschicht* zur Verfügung gestellt, welche einen sicheren Austausch von Daten garantiert.

Speziell für den Bereich des Tinnitus wird hierbei mittels der bestehenden *Track Your Tinnitus* Plattform eine Lösung zur Verfügung gestellt, auf die im nächsten Abschnitt genauer eingegangen wird.

2.3 Mobile Crowd Sensoring und Track Your Tinnitus

Track Your Tinnitus(TYT) ist ein Forschungsprojekt der Universität Ulm, welches auf der sogenannten *Mobile Crowd Sensing* Forschung basiert. Unter Mobile Crowd Sensing versteht man die Datenerfassung von größeren Menschenmengen mittels Smartphones und anderen mobilen Geräten. Diese Daten werden anschließend ebenfalls ausgewertet. Das Besondere an dieser Art der Datensammlung ist es, dass Daten durch die Kamera, das Mikrofon oder auch den GPS-Sensor erfasst und analysiert werden können. Man spricht hierbei von sogenannten stationären Daten[18]. Mobile Crowd Sensing stößt vor allem in der klinischen und psychologischen Behandlung auf großen Zuspruch[19]. So wird der Austausch von Daten und die Kommunikation zwischen Arzt und Patient

vereinfacht. Ebenso zeigen die Ergebnisse der Studie, dass Patienten in hohem Maße motiviert sind die Plattform zu nutzen, vor allem wenn sie selbst an der jeweiligen Krankheit leiden. Speziell bei jüngeren Patienten stößt eine solche Herangehensweise auf sehr viel Zuspruch, während ältere Patienten eine solche Form der Datenerfassung seltener nutzen[20].

Genauer wird in dieser Studie auf Track your Tinnitus eingegangen, welches ein Forschungsprojekt der Universität Ulm ist. Hierbei wird der Patient mittels speziell entwickelten Fragebögen zu seiner Erkrankung befragt. Das Ziel ist es hierbei festzustellen, wie die aktuelle Tinnitusempfindung des Patienten mit dem aktuellen Tagesablauf und seinen Aktivitäten zusammenhängt.

Die Resultate des Fragebogens werden an das *Backend* des Track your Tinnitus gesendet. Das Backend dient hierbei zur Sammlung der Daten über den Patienten. Ein Arzt kann nun auf das Backend zugreifen und hat somit Zugang zu den Daten des Patienten und kann diese auswerten.
Einen negativen Einfluss auf eine Tinnituserkrankung wird von Track Your Tinnitus nicht ausgeübt[21]. Da eine Tinnituserkrankung über den Tag verteilt in ihrer Intensität variiert[13], verfügt TYT über Notifikationen, sollte lange Zeit kein Fragebogen mehr ausgefüllt worden sein. Verfügbar ist Track your Tinnitus auf den Plattformen iOS und Android. Die Nutzung des Projekts wurde jedoch bereits erweitert. So wurde es möglich gemacht, mittels einer Android Smartwatch, der so genannten *Android Gear*, die Fragebögen auch mit der Smartwatch auszufüllen. Für die Apple Watch, kurz iWatch, wurde ein Prototyp[9] entworfen, welcher nun umgesetzt werden soll.
Zunächst erfolgt eine kurze Vorstellung der Apple Watch.

2.4 Apple Watch

Im Zuge der Smartwatch Entwicklung kündigte Apple im September 2014 die *iWatch* an und brachte das mobile Gerät im April des folgenden Jahres auf den Markt. Bei der iWatch handelt es sich um eine sogenannte *Smartwatch*. Unter diesem Begriff fasst man alle Armbanduhren zusammen, welche über die Funktionalität eines Computers

verfügen. Gekoppelt mit einem Smartphone bringt eine solche Smartwatch weitere Funktionalitäten mit sich, auch wenn nicht jede Smartwatch mit einem Smartphone gekoppelt sein muss. Ein Beispiel hierfür ist die *Gear S* von Samsung. Die iWatch wiederrum muss im Hintergrund mit einem iPhone gekoppelt sein. Wichtig ist hierbei, dass die iWatch mindestens mit einem iPhone 5 oder höher gekoppelt sein muss. Mit nachfolgenden iPhones und ab iOS 8.2 ist die iWatch absolut kompatibel.

Die iWatch selbst wiederrum verfügt über diverse Features. So kann sie mittels eingebauter Sensoren die Herzfrequenz des Nutzers messen, was sie vor allem in gesundheitlichen Bereichen sehr effektiv einsetzbar macht. Sie verfügt desweiteren über Lage- und Beschleunigungssensoren, welche vor allem zur Bewegungsmessung eingesetzt werden können, unter anderem in Fitnessapps.
Die Kopplung der iWatch mit einem iPhone erfolgt via Bluetooth[1]. Da die iWatch als Erweiterung des iPhones agiert, können Apps welche auf dem iPhone installiert sind wie gewohnt weiter genutzt werden, sofern die Kopplung bestehen bleibt.
So können, neben Mitteilungen welche an das Smartphone gesendet werden, auch Dienste, wie zum Beispiel das Wetter, angezeigt werden. Auch der Sprachassistent *Siri* kann genutzt werden. Zuletzt sind auch eher alltägliche Nutzungsmöglichkeiten wie die Handhabung einer iTunes-Playlist mittels der iWatch möglich.

2.4.1 Technologie

Die iWatch verfügt, wie schon das iPhone, über die nachfolgenden Schnittstellen. So sind Bluetooth 4.0 und Wi-Fi 802.11b/g verfügbar. Ein NFC[2] Chip ist ebenfalls in der iWatch verbaut, dieser wird jedoch vorerst nur nur von *Apple Pay*, dem Apple eigenen Bezahldienst, benutzt. Ebenfalls besitzt die iWatch einen Beschleunigungssensor und auch ein Gyroskop. Das Gyroskop ermittelt hierbei genaue Bewegungsdaten. Des Weiteren ist ein Herzschlagsensor an der Unterseite der Uhr verbaut. Zu guter Letzt verfügt die Apple Watch noch über ein Mikrofon, welches es erlaubt Spracheingaben vorzunehmen.

[1] http://www.bluetooth.com
[2] http://nearfieldcommunication.org

2.4.2 Vergleichbare Apps

Wie bereits erwähnt verfügen Smartwatches mit einem Android Betriebssystem bereits über eine Track Your Tinnitus Applikation. Für die iWatch wiederrum scheint es keinerlei Applikationen zu geben, welche bei einer Tinnitusbehandlung unterstützend sind (Stand: Juli 2017). So gibt es zwar mit *Tinnitracks* eine App für das iPhone welche mittels Musik eine Tinnituserkrankung therapiert[2], besagte App verfügt aber über keine iWatch Erweiterung. Ähnlich zur beschriebenen Tinnitracks Applikation arbeitet *Free Tinnitus HQ*. Diese App dient dazu mittels eigenen Klangmixen die Symptome des Tinnitus zu überlagern und somit für eine Entspannung des Patienten zu sorgen[22].
Doch auch Free Tinnitus HQ verfügt über keine Erweiterung für die iWatch. Im Umkehrschluss würde dies bedeuten dass Track Your Tinnitus die erste Applikation wäre, welche in diesem Themengebiet über eine iWatch App verfügen würde. Es wäre vor allem die erste Applikation, welche sich nicht auf eine Behandlung mittels Musik fokussiert, sondern auf das Aufzeichnen von den Patientendaten.
Was ein Alleinstellungsmerkmal bedeuten würde.

3

Verwandte Arbeiten

In diesem Kapitel werden verwandte Arbeiten betrachtet, welche sich mit ähnlichen Themen befassen.
Hierzu werden drei Arbeiten konkreter beleuchtet. So untersucht die erste Arbeit die Umsetzung eines mobilen Frameworks zur Unterstützung von tinnitusgeschädigten Patienten. Die zweite Arbeit behandelt die Verwendung von Smartwatches im Zusammenhang mit klinischen Studien.
Zu Letzt wird noch eine Arbeit betrachtet, welche den Prototyp eines solchen Fragebogens für die Apple Watch konstruiert.

3.1 Mobiles Framework zur Unterstützung von Tinnitus Patienten

Im Rahmen der Diplomarbeit von Jochen Herrmann wurde eine Erfassung von klinischen Parametern hinsichtlich an Tinnitus erkrankten Patienten entwickelt. Hierbei setzte der Autor vor allem auf die Smartphones zur Visualisierung und Überwachung der Schwankungen der Tinnitus Wahrnehmung. Im Rahmen dieser Arbeit wurde eine Website entwickelt und ebenso eine App für Android und für iOS. Für die konkrete Umsetzung wurde das Projekt Track Your Tinnitus verwendet[23].

Nachfolgend wird der Aufbau des Track Your Tinnitus Projekts erklärt.

3.1.1 Aufbau des TYT Projekts

Das Track Your Tinnitus Projekt besteht aus zwei großen Komponenten. Dem *Server* und den *Apps* für das Smartphone. Nachfolgend, in Abbildung 3.1, ist der genaue Aufbau dargestellt.

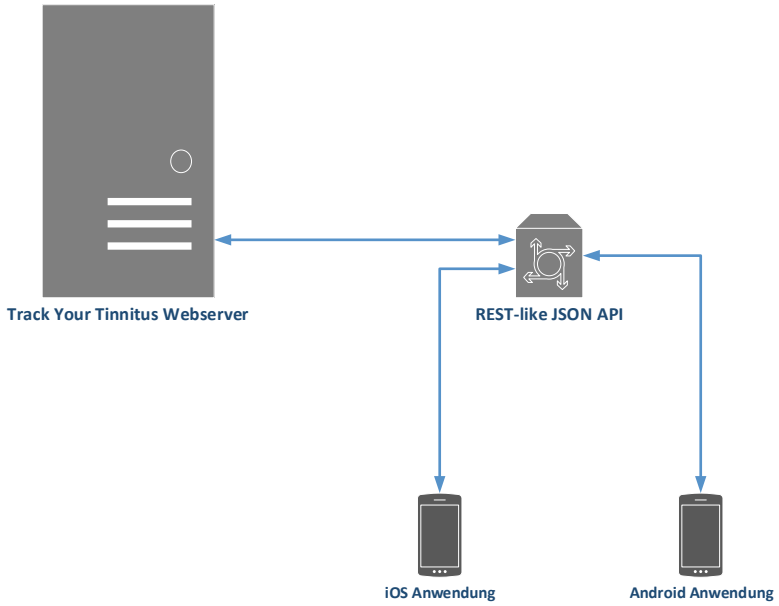

Abbildung 3.1: Aufbau des Track Your Tinnitus Projektes

Der *Server*, welcher in Kontakt mit den Smartphones steht, ist für die Netzdienste zuständig. Hierbei verwendet der Server des TYT Projekts das Betriebssystem *Linux*. Der Webbrowser erhält seine Dokumente hierbei mittels eines Webservers, welcher mit *Apache* arbeitet.
Für eine sichere Verwahrung der Daten wird *MySQL* verwendet, welches durch *PHP* als Skriptsprache unterstützt wird. Mittels des *Laravel Frameworks* wird die Webanwendung realisiert. Die Smartphone Apps wiederrum sind jeweils nativ. Das bedeutet, dass die Apple iPhone App mithilfe von *Objective C* realisiert wurde. Die Android Applikation wurde mittels der Programmiersprache *Java* realisiert.

Auf Grundlage einer *REST-ähnlichen JSON-API* wird die Kommunikation zwischen den Anwendungen auf dem Smartphone und der Webanwendung realisiert.
Als *REST* (REpresentational State Transfer Architektur) wird hierbei eine Architektur bezeichnet, welche beschreibt in welcher Art die Web Standards so eingesetzt werden, dass die dem Web gerecht werden[24]. Die notwendige Schnittstelle, welche die Kommunikation zwischen den Smartphones und der Webanwendung ermöglicht, ist durch die *JSON API* gegeben. Zur Verwaltung der externen Bibliotheken verwendet Track Your Tinnitus, zumindest für die iOS App, sogenannte *Cocoa Pods*[25]. Hierbei werden die benötigten Bibliotheken in einer einzelnen Datei gehalten, der sogenannten *PodFile*. Mit dem Verwenden von CocoaPods erschafft man somit eine zentrale Anlaufstelle und muss nicht jede verwendete Bibliothek einzeln verwalten[26].

Im Anschluss werden nun weiter die Website und die Smartphone Applikation des Track Your Tinnitus Projekts erklärt.

3.1.2 Aufbau der Website

Möchte man die Smartphone Applikationen des Track Your Tinnitus nutzen, sei es für Android oder iPhone, so ist zunächst eine Registration auf der Website notwendig[1]. Hierbei werden die Anmeldedaten gespeichert. Direkt anschließend hat der Nutzer die Möglichkeit die Fragebögen zu bearbeiten. Möchte der Nutzer jedoch die Schwankungen seiner Erkrankung verfolgen, so ist es notwendig, dass die Smartphone App installiert wird, da nur hier die Möglichkeit der Verfolgung besteht. Nach dem Login wird der Nutzer direkt auf eine Übersicht zu seinen bearbeiteten Fragebögen weitergeleitet. Hat der Nutzer einen Fragebogen vollständig bearbeitet, so ist dieser grün eingefärbt. Ist der Fragebogen nicht vollständig, erhält er eine blaue Signalfarbe. Ein leerer Fortschrittsbalken ist nur dann sichtbar, wenn der Fragebogen noch nicht gestartet wurde.
Dadurch soll es dem Nutzer ermöglicht werden, einen schnelleren und besseren Überblick über seine bereits absolvierten Fragebögen zu bekommen[23]. Ebenso erhält der Nutzer auf seiner persönlichen Seite eine Übersicht über seine erfassten Daten aus dem Smartphone.

[1] http://www.trackyourtinnitus.com

3.1.3 Die Smartphone Apps des TYT

Um eine Überwachung über die Tinnituserkrankung zu erhalten, ist es zwingend notwendig die Smartphone Apps zu verwenden, da eine Überwachung nur auf diesen realisiert wurde. Verfügbar ist die Track Your Tinnitus App für das Apple iPhone ab iOS6 und iOS7, als auch für die Android Geräte ab Version 2.3.
Zum Zeitpunkt des Entstehens der Arbeit war für das Windows Phone von Microsoft keine App verfügbar[1]. Nach dem *Login* kann der Nutzer damit beginnen, die Fragebögen auszufüllen. Sollte der Nutzer lange keine Fragenbögen mehr ausgefüllt haben, so erhält er einen Notifikation auf den Sperrbildschirm seines Smartphones.

3.2 Using Wearables in the Context of Chronic Disorders

Viele Apps wie Track Your Tinnitus ermöglichen es Patienten ihre gesundheitliche Situation besser zu verstehen oder selbige besser durchzustehen. Diese Anwendungen sind oftmals jedoch nur für die jeweiligen Smartphones verfügbar und nicht für andere Geräte.
Da, wie bereits in Kapitel 2 erwähnt, in den letzten Jahren immer mehr *Wearables* in Form von Smartwatches erscheinen, wurde im Rahmen einer Studie der Gebrauch von sogennanten Wearables bei chronischen Erkrankungen untersucht[19]. Hierzu wurde überprüft ob es möglich ist Fragebögen, wie den Track Your Tinnitus Fragebogen, auf sogenannten *Wearables* umzusetzen. Unter die Gruppe der Wearables fallen auch die in Kapitel 2 vorgestellten Smartwatches. Im Rahmen dieser Studie wurde festgestellt, dass die Verwendung von Smartwatches im Mobile Crowd Sensing sinnvoll ist.

3.3 Prototyp einer iWatch Umsetzung

Track Your Tinnitus existiert schon in einer Umsetzung für das iPhone und für alle Android Geräte. Für die Android Geräte ist bereits eine Smartwatch-Erweiterung verfügbar.

[1] http://www.trackyourtinnitus.com

Da eine iWatch Version noch nicht verfügbar war, entstand ein *Prototyp*[9] für die iWatch. Mit besagtem Prototyp wurde in einer Studie untersucht, ob die Apple Watch im Mobile Crowd Sensing Bereich einsetzbar sei.
Nachfolgend wird zunächst der Prototyp vorgestellt und anschließend sollen die wichtigsten Resultate der Studie analysiert werden.

3.3.1 Analyse des Prototyps

Die prototypische Realisierung bestand aus vier Fragebögen. Jedoch waren jeweils zwei Fragebögen gleichartig, nur die *Designs* waren unterschiedlich.
Es wurden folglich zwei Design-Arten eingesetzt.

Design 1

Das erste Design hatte dabei *Single Choice* Antworten, *Binärantworten*, eine *Spracheingabe* und auch *Sliderantworten* zur Auswahl.
Nachfolgend soll erklärt werden, worin sich diese Antwortmöglichkeiten unterscheiden.
Bei einer Single Choice Antwort kann der Nutzer aus vielen möglichen Antworten genau eine auswählen.
Eine Binärantwort kann mittels Ja/Nein Tasten beantwortet werden.
Die Spracheingabe wiederrum ist, wie der Name schon sagt, eine Eingabe mittels des Mikrofons der iWatch.
Zuletzt gab es noch die Möglichkeit mittels eines Schiebers den Grad der aktuellen Tinnitusschwere einzustellen. Eine solche Art der Eingabe wird als Sliderantwort bezeichnet.

Design 2

Das zweite Design beinhaltet alle Antwortmöglichkeiten aus dem ersten Design, mit einer Erweiterung. So wurde noch eine weitere Eingabemöglichkeit hinzugefügt, die sogenannte *Pickerantwort*.
Mittels der Pickerantwort ist es dem Nutzer möglich, einen Wert aus einer abgebildeteten

Sequenz auszuwählen. So kann der Nutzer beispielsweise den Grad des Tinnitus in Prozent einstellen und diesen speichern.
Abschließend sollen nun die Ergebnisse der Studie analysiert und diskutiert werden.

3.3.2 Analyse der Studie

Zunächst wird das Studiendesign betrachtet, danach sollen die Ergebnisse der Studie erläutert werden.

Studiendesign

Beim Design der Studie wurde davon ausgegangen, dass die Probanden noch nie in Kontakt mit einer Smartwatch gekommen sind. Dementsprechend wurde zunächst der Aufbau der Uhr anhand einer einfachen Einführung erklärt. Anschließend konnten die Probanden anfangen die Fragebögen zu bearbeiten, beginnend mit dem ersten Fragebogen. Sobald der Proband mit dem Ausfüllen der Fragebögen fertig war, hatte er am Apple iPad weitere Vergleichsfragebögen zu bearbeiten. Diese Fragebögen erhoben *personenrelevante Merkmale*, wie Alter und Geschlecht.

Ergebnisse

Mittels der Studie wurde belegt, dass ein mobiles Smartwatch-Fragebogensystem viel Zuspruch bei der befragten Zielgruppe erhielt.
In den Ergebnissen der Studie war jedoch deutlich erkennbar, dass die Probanden Fragen bezüglich ihrer Gesundheit nicht in der Öffentlichkeit per Spracheingabe beantworten wollten.

3.4 Fazit

Anhand der ersten Arbeit wurde aufgezeigt, wie mittels eines entwickelten Frameworks entwaige Schwankungen der Tinnituswahrnehmung überwacht werden können. Die

dazu eigens realisierte Website des Track Your Tinnitus Projekts ermöglicht es dabei den Nutzern die Fragebögen direkt zu bearbeiten. Mittels der Smartphone Apps können dabei die Tinnitusschwankungen leichter überwacht werden, da auf die Fragebögen mittels einer Notifikation hingewiesen wird.

In der zweiten Arbeit sollte untersucht werden ob es möglich wäre, die bekannten Fragebögen des Track Your Tinnitus Projekts für Wearables, unter die auch die iWatch fällt, umzusetzen. Hierbei wurde anhand einer Studie festgestellt, dass Smartwatches für den Einsatz im medizinischen Bereich durchaus in Frage kommen können, einzig die Umsetzung ist noch nicht vorhanden.

Mittels der dritten Arbeit wurde eine solche Umsetzung als Prototyp realisiert. Hierzu wurden zwei Designs an einer Gruppe von Probanden getestet. Im Rahmen dieses Tests wurden einige Hypothesen geprüft, welche für die endgültige Umsetzung von hoher Relevanz sein werden.

4

Anforderungsanalyse

In diesem Kapitel werden Anforderungen an das Projekt definiert, welche den Systemumfang beschreiben. Beginnend mit Anwendungsfällen, schließt das Kapitel mit der Unterteilung der Anforderungen in *funktionale* und *nicht-funktionale* Anforderungen.

4.1 Anwendungsfälle

Anforderungen werden in zwei Kategorien eingeteilt. Einerseits in *funktionale* und andererseits in *nicht-funktionale* Anforderungen. Der Unterschied besteht darin, dass funktionale Anforderungen beschreiben was das System *leisten* soll. Nicht-funktionale Anforderungen wiederrum geben eine Übersicht darüber wie das System *arbeiten* soll. Nach IEEE-Standard[1] gelten nachfolgende Qualitätsmerkmale.

Eindeutig
Jede Anforderung besitzt nur eine einzige Interpretationsmöglichkeit
Vollständig
Jede Anforderung ist formal und inhaltlich vollständig formuliert
Konsistent
Jede Anforderung ist mit anderen und mit sich selbst konsistent
Korrekt
Jede Anforderung ist korrekt
Verifizierbar
Jede Anforderung ist verifizierbar

[1] https://www.ieee.org/index.html

Realisierbar
Jede Anforderung ist umsetzbar
Klassifizierbar
Jede Anforderung ist bezüglich juristischer Verbindlichkeiten klassifizierbar
Bewertbar
Jede Anforderung ist hinsichtlich Priorität, Wichtigkeit und auch Kritikalität bewertbar

Werden diese Kriterien auf die Realisierung des Fragebogensystems der iWatch übertragen, so erwartet der Nutzer eine Anzahl an Fragen welche er beantworten kann. Hierbei unterscheiden sich verschiedene Eingabemöglichkeiten in ihrer Darstellungsart, die nachfolgend aufgelistet werden.

Single Choice
Der Nutzer erhält die Möglichkeit aus mehreren Antwortmöglichkeiten eine auszuwählen
Binärantwort
Aus zwei Antwortmöglichkeiten erhält der Nutzer die Möglichkeit eine auszuwählen
Slider
Der Nutzer kann seine Antwort anhand eines Wertes innerhalb einer Skala abgeben
Picker
Der Nutzer gibt seine Antwort mittels eines Wertes innerhalb einer Sequenz ab

Anschließend werden nun die dargestellen Antwortmöglichkeiten für das Projekt unterteilt, in funktionale und nicht-funktionale Anforderungen.

4.2 Funktionale Anforderungen

Bezug nehmend auf die Funktionalitäten des Systems, entstehen folgende funktionale Anforderungen.

Nr.	Bezeichnung	Beschreibung
1	Anzeigen des Fragebogens	Beim Start der App soll der Nutzer den Fragebogen angezeigt bekommen
2	Beantworten von Fragebögen	Mittels der oben genannten Antwortmöglichkeiten soll es dem Nutzer möglich sein alle Fragen zu beantworten.
3	Erweiterbarkeit	Bestimmten Nutzern soll es möglich gemacht werden neue Fragen oder neue Fragebögen hinzuzufügen
4	Konsistenz der Daten	Der Nutzer muss nicht den selben Fragebogen mehrmals ausfüllen. Daten der iWatch werden gespeichert.
5	Änderung der gegebenen Antworten	Dem Nutzer soll es möglich sein zur vorherigen Frage zurückzukehren um die Antwort zu ändern, sollte diese falsch eingegeben worden sein.

Tabelle 4.1: Übersicht der funktionalen Anforderungen

4.3 Nicht-funktionale Anforderungen

Bei den nicht-funktionalen Anforderungen wird Bezug darauf genommen, wie das Fragebogensystem die erwünschten funktionalen Anforderungen erfüllt.

Nr.	Bezeichnung	Beschreibung
1	Technische Anforderungen	Das System wird mittels Xcode auf MacOS 10.12.5 entwickelt und soll unter watchOS 2 und iOS 7 oder neuer lauffähig sein
2	Verfügbarkeit	Die App soll jederzeit zur Verfügung stehen
3	Stabilität und Robustheit	Die App soll gegenüber jeden Eingaben robust funktionieren. Nach einem Neustart des Systems soll die Anwendung fortgeführt werden können
4	Nutzerfreundlichkeit	Der Nutzer soll eine möglichst einfache und selbsterklärende Bedienung vorfinden
5	Watch-Kit Erweiterung im existenten iOS Projekt	Das bereits existente Projekt für iOS soll so erweitert werden dass die Verwendung einer iWatch möglich ist
6	Erweiterung der existenten Applikationen	Beide existenten Applikationen, der WatchOS Prototyp und die iOS App, sollen miteinander verbunden werden

Tabelle 4.2: Übersicht der nicht-funktionalen Anforderungen

Durch die nun beschriebenen Anforderungen kann mit der Realisierung im nächsten Schritt begonnen werden.

5 Theorie

Im nachfolgenden Kapitel wird genauer auf die Entwicklungsumgebung für Apple Applikationen eingegangen, ebenso wie auf die Umsetzung auf den einzelnen Geräten. Am Ende wird noch diskutiert, welche Möglichkeiten zur Verfügung stehen eine iPhone und eine iWatch miteinander zu verbinden. Zunächst wird jedoch die Entwicklungsumgebung von Apple, *Xcode*, genauer vorgestellt.

5.1 Xcode

Xcode[1], die Entwicklungsumgebung von Apple, erlaubt es Programme für macOS, iOS, tvOS und ebenso watchOS zu entwickeln. Eine Entwicklung von Apps für die oben genannten Plattformen ist nur mittels Xcode möglich.
Anders als jedoch zum Beispiel C-IDEs, kann Xcode nicht auf anderen Plattformen verwendet werden, sondern nur auf macOS. Die aktuelle Version von Xcode (Stand: Juli 2017) ist 9 beta 4. Neu hinzugefügt wurde mit der Version 9 die Unterstützung für *SWIFT 4*. Swift ist neben *Objective-C* eine der Programmiersprachen welche Xcode unterstützt. Eine Verwendung von *C* und *C++* ist jedoch ebenso möglich. Hauptsächlich werden jedoch die beiden erstgenannten Programmiersprachen verwendet.

5.1.1 Aufbau

Die wichtigsten Bestandteile der Oberfläche von Xcode sollen nachfolgend erläutert werden. So ist der *Quellcodeeditor* hauptsächlich dafür verantwortlich, den Code der

[1] https://developer.apple.com/xcode/

verwendeten Programmiersprache zu betrachten und zu bearbeiten. Der *Interfacebuilder* wiederrum dient dazu, für die entwickelte Anwendung mittels *Drag und Drop* eine graphische Oberfläche zu erstellen. Besagte Oberfläche wird als *Storyboard* bezeichnet und funktioniert ähnlich zu anderen Entwicklungsumgebungen, wie zum Beispiel Visual Studio[2] von Microsoft.

Entwicklung mit Xcode

Mittels Xcode können verschiedene Arten der Anwendung entwickelt werden. So können für das iOS Betriebssystem von Apple, welches auf den Smartphones und Tablets verwendet wird, eine *Single-View Application* oder eine *Page-Based Application* entwickelt werden. Die Single-View Application enthält eine leere *View* und einen dazugehörigen *View Controller*. Die Page-Based Applikationen basieren wiederrum auf *Datenquellen* und können nicht in voller Gänze mittels eines *Storyboards* implementiert werden[27]. Möchte man Applikationen für die iWatch entwickeln, so benötigt man mit *WatchKit* ein spezielles, sogenanntes, *Target*. Auf dieses soll nun nachfolgend eingegangen werden.

5.2 WatchKit

Das *WatchKit Framework* beinhaltet alle Klassen, welche für die WatchKit Erweiterung nötig sind, um das Interface und somit den Programmablauf eines watchOS Programms beeinflussen zu können.
Hierzu besitzt die watchOS App mindestens einen *Interface Controller*. Wie eingangs bereits dargelegt, ist es notwendig die iWatch mit einem iPhone gekoppelt zu haben, um die volle Funktionen der iWatch nutzen zu können. Das bedeutet, dass das Projekt nicht nur auf der Uhr ausgeführt wird, sondern auch eine Applikation für das iPhone im Hintergrund entwickelt werden muss. Der Grund hierfür ist die Zweigeteiltheit der App. Ein Teil der App, das Storyboard und andere benötigte Ressourcen, werden auf der App ausgeführt. Der restliche Teil wird auf dem iPhone ausgeführt. Somit handelt es sich also bei einer iWatch App um eine Erweiterung einer iPhone Applikation[28]

[2] https://www.visualstudio.com/de/

Das bedeutet für die Umsetzung folgendes:
Zum einen besteht das gesamte Xcode Projekt aus einem iPhone Projekt. Dieses beinhaltet ein *Storyboard* und auch den oder die *ViewController* für die App des iPhone. Zusätzlich zu dem iPhone Projekt kommt noch das WatchKit-Projekt hinzu.

Das WatchKit-Projekt selbst besteht aus zwei Ordnern, dem *WatchKit App* Ordner und dem *WatchKit Extension* Ordner. Der WatchKit App Ordner beinhaltet das *Storyboard* der watchOS Anwendung, während der WatchKit Extension Ordner für den *ViewController* der watchOS Applikation zuständig ist. Der WatchKit Extension Ordner kann natürlich, sollte es notwendig sein, mehr als nur einen Controller beinhalten.
Anzeigen auf der Watch, die innerhalb einer Applikation dargestellt werden, werden als *Views* bezeichnet. Innerhalb der *Views* gibt es deutliche Unterscheidungen. Für die gewöhnliche Interaktion mit dem Benutzer sollte man reguläre Views benutzen. Für diesen Verwendungszweck können in den Interface Controllern sogenannte *Scenes* verwendet werden. Hierbei ist es dem Nutzer einzig möglich Daten einzugeben und abzuändern.
Soll eine Benachrichtigung erfolgen, zum Beispiel über eingehende Anrufe oder Nachrichten, so kann dies mittels einer *Notification Scene* gelöst werden. Mittels einer Notification Scene kann der Nutzer direkt auf die eingehende Nachricht reagieren. So ist es dem Nutzer möglich beispielsweise mittels vorgefertigen Antworten auf Textnachrichten zu antworten. Es muss jedoch sichergestellt sein, dass die sogenannte *Parent-App* auf dem iPhone installiert ist und über eine iWatch Erweiterung verfügt.

Bis hin zur WatchOS2 gab es noch die Möglichkeit eines *Glance*. Ein Glance erlaubt dem Nutzer, im Gegensatz zu den bisherigen Views, keine Dateneingabe. Der Glance dient nur dazu, den Nutzer über Sachverhalte in Kenntnis zu setzen. So kann diese Variante beispielsweise dazu genutzt werden, den Nutzer über seinen Blutdruck aufzuklären. Mit der WatchOS3 wurden Glances jedoch komplett gestrichen und durch das *Dock* ersetzt[29].

5.3 WatchKit und iOS

Wie bereits beschrieben, ist es notwendig ein iPhone mit einer Apple Watch gekoppelt zu haben, damit die Apple Watch in ihrer Vollständigkeit funktionieren kann.
Da es im Rahmen dieser Arbeit notwendig ist, ein bereits existierendes *WatchKit Projekt* in ein ebenso bereits existentes *iOS Projekt* zu integrieren, sollen nun verschiedene Möglichkeiten dargelegt werden die Arbeit umzusetzen. Zunächst soll untersucht werden ob es in den verschiedenen Versionsverläufen der iWatch Änderungen gegeben hat.
Darauf aufbauend soll eine theoretische Umsetzung der Arbeit diskutiert werden. Hierzu wird überprüft, welche Art des Fragebogens[9] verwendet wird. Ebenso werden verschiedene Möglichkeiten diskutiert, wie ein Datenaustausch zwischen dem iPhone und der iWatch möglich gemacht werden kann.

Zuerst wird jedoch untersucht, ob ein drastischer Unterschied zwischen den verschiedenen Versionsverläufen der iWatch vorhanden ist.

5.3.1 Unterschiede im Versionsverlauf der iWatch

Die Apple Watch, und somit auch ihr Betriebssystem *watchOS*, liegen inzwischen in der dritten Version vor, die neue Version watchOS4 ist, Stand Juli 2017, für den Herbst des gleichen Jahres angekündigt. Mit watchOS1 war es nur möglich, die Watch App auf der Apple Watch auszuführen. Dies bedeutete im Umkehrschluss, dass die Apple Watch nur als erweiterte Anzeige diente. Alle Rechenoperationen waren auf das iPhone ausgelagert[30].
Dies hatte eine äußerst schlechte Performance der Anwendungen zur Folge und die völlige Abhängigkeit vom iPhone. Erst mit dem Erscheinen von *watchOS2* war es möglich, auch Code auf die Watch auszulagern. Apps werden seither auf der Watch selbst ausgeführt[31].
Das lag vor allem daran, dass Apple das sogenannte `WatchConnectivity`[32] Framework einführte. Auf das Framework wird im praktischen Teil der Arbeit nochmals genauer eingegangen. Mit der watchOS3 wurde dann das Framework nochmals um einige Operationen erweitert. [29].

5.4 iPhone und die iWatch

Zur Kopplung gibt es mehrere Möglichkeiten, wie nachfolgend erläutert.

5.4.1 Verbindung mittels CoreData

Apple stellt mittels `CoreData`[33] ein Framework zur Persistierung von Objekt-Graphen zur Verfügung. `CoreData` nutzt *SQLite* als Grundlage und bietet eine gute Abstraktion der Datenbank, daher ist es relativ schnell und leicht zu integrieren. `CoreData` verfügt über einen graphischen Editor, welcher es erlaubt, Datenschemata zu erstellen und diese in Klassen zu transformieren. Besagte in `CoreData` gespeicherte Daten lassen sich dann mittels einfachen *Datenbankabfragen* ermitteln.
Es muss also keine eigene Logik für die Suchen und Sortierungen implementiert werden. Sollten die Daten geändert werden, so bietet `CoreData` dafür eigens verfügbare Schnittstellen. Besagte Schnittstellen informieren dann über Änderungen an den Datensätzen. Da `CoreData` bereits seit mehreren Jahren verfügbar ist, lässt sich daher eine stabile Nutzung garantieren.
Des Weiteren ist `CoreData` außerdem auch für die AppleWatch verfügbar.

Eine weitere Möglichkeit ist mittels `WatchConnectivity` gegeben.

5.4.2 Verbindung mittels WatchConnectivity

`WatchConnectivity`[32] erlaubt eine Zwei-Wege Kommunikation zwischen einer iOS App und der dazugehörigen watchOS Applikation.
Dieses *Framework* soll verwendet werden um Daten zwischen den beiden Applikationen auszutauschen. Es ist hierbei möglich kleinere Teile von Daten und ebenso ganze Dateien auszutauschen. Die meisten Transfers geschehen dabei im Hintergrund. Ein Datenaustausch in Echtzeit ist jedoch ebenso möglich. Es ist wichtig die Verbindung zum richtigen Zeitpunkt des *Application-Lifecycles*, also der Laufzeit der Applikation, aufzubauen.

Ist dies nicht der Fall, so könnte es zu dem Verlust der Daten kommen, da eine Weiterverarbeitung eben dieser Daten nicht geregelt ist.

Nachdem der theoretische Hintergrund beleuchtet wurde, folgt nun die praktische Umsetzung.

6

Praktische Umsetzung

Die in den vorherigen Kapiteln aufgeführten analytischen Ergebnisse sollen nun umgesetzt werden. Hierbei werden auch die beiden existierenden Arbeiten, Track Your Tinnitus[23] und der bereits erstelle Prototyp für die iWatch[9], berücksichtigt. Nachfolgend werden die Schritte für die Umsetzung genauer beschrieben. Hierbei wird auf das Zusammenführen der beiden existenten Projekte eingegangen.

Begonnen wird hierbei mit der Wahl des endgültig genutzten Fragebogens.

6.1 Wahl des Fragebogen

Für den bisherigen Prototypen[9] wurden vier verschiedene Fragebögen verwendet, welche von den Probanden evaluiert werden sollten. Hierbei wurde gezeigt, dass *63 Prozent* der Nutzer nicht dazu bereit sind, Fragen bezüglich der Gesundheit an öffentlichen Plätzen mittels *Spracheingabe* zu beantworten. Die Probanden sind jedoch generell dazu bereit, Fragen bezüglich ihrer Gesundheit mittels eines Fragebogens zu beantworten. Aufgrund dieser Resultate wurde für die konkrete Umsetzung des Prototypen der in den Umfragen verwendete *Fragebogen 2* umgesetzt. Eine vollständige Betrachtung des Fragebogens findet sich in Kapitel 7.

6.2 Xcode und Track Your Tinnitus Projekt

Für die Arbeit an den beiden Projekten, Track your Tinnitus und dem Prototyp der watchOS App, wurde Xcode in der Version 8.8.3 verwendet. Da Xcode, wie in Kapitel 2

erwähnt, nur auf einem Apple Betriebssystem funktioniert, wurde ein MacBook mit iOS 10.12.25 Sierra verwendet. Track Your Tinnitus wurde in der aktuellen Version bearbeitet. Hierbei kam es initial zu einigen Problemen, welche nachfolgend genauer erläutert werden. Eine Lösung wird, sofern verfügbar, ebenso direkt dargelegt.

6.2.1 Podfiles

Wie in Kapitel 3, Verwandte Arbeiten, erläutert wurde, arbeitet TYT mittels *CocoaPods*. CocoaPods sind ein Werkzeug, mit welchen externe Bibliotheken in genutzte Xcode Projekte eingebunden werden können. Es ist hierdurch möglich, eine Vielzahl an *Open-Source-Bibliotheken* zu den eigenen Projekten hinzuzufügen. Die Projektabhängigkeiten werden mittels einer sogenannten *Podfile-Datei* im Projekt beschrieben[34]. Eine Podfile muss, bevor das Projekt via der *.xcworkspace Datei* geöffnet wird, mithilfe der Konsole installiert werden. Hierbei muss im Verzeichnis des Projekts das Kommando `pod install` ausgeführt werden.

Mögliche Fehler

Bei der oben genannten Konstellation des Betriebssystems, der Xcode Version, als auch der Projektversion, kam es jedoch zu nachfolgender Fehlermeldung.

```
diff: /../Podfile.lock: No such file or directory
diff: /Manifest.lock: No such file or directory
error: The sandbox is not in sync with the Podfile.lock. Run 'pod install' or update your CocoaPods
  installation.
```

Abbildung 6.1: PodFile Fehlermeldung

Um diesen Fehler zu korrigieren war es notwendig, die bisher existierende *Podfile* umzuschreiben und somit an den gängigen Standard anzupassen.

```
source 'https://github.com/CocoaPods/Specs.git'
platform :ios, :deployment_target => "6.0"
target 'TrackYourTinnitus' do
pod 'AFNetworking', '~> 1.2'
pod 'SIAlertView'
pod 'TimeScroller', '~> 2.0.0'
end
```

Listing 6.1: Implementierung der Podfile

Wichtig war es hierbei, wie in *Listing 6.1* gesehen, sowohl eine *source*, als auch das *target* anzugeben.
Nach dem die Podfile ersetzt wurde, kann der Befehl `pod install` nochmals ausgeführt werden und das Projekt sollte sich ohne weitere Fehler starten lassen.

6.3 Zusammenfügen der Projekte

Das Zusammenfügen der Projekte war die Kernaufgabe dieser Arbeit. Daher soll nun genauer untersucht, wie die Apple Watch und das iPhone zusammengeführt werden können, so dass das Projekt komplett funktioniert. Zunächst wird erklärt, wie eine WatchKit Extension in ein bereits existentes Projekt eingefügt werden kann.

6.3.1 Einfügen eines WatchKit

Das Hinzufügen einer WatchKit Erweiterung mittels Xcode in ein bereits existentes Projekt erfolgt mithilfe eines Klicks auf *File-New-Target*. Dort kann dann durch ein Klick auf WatchOS eine *WatchKitApplication* ausgewählt werden, wie im nachfolgenden Bild dargestellt. Im nächsten Feld kann nun der Name des Projekts eingegeben werden.

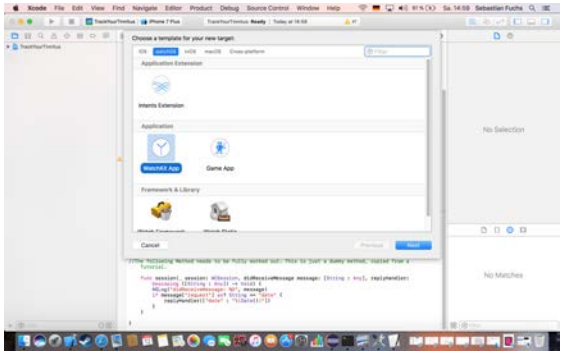

Abbildung 6.2: Hinzufügen eines WatchKit Targets in Xcode

Letztlich ist noch zu entscheiden mit welcher Programmiersprache gearbeitet werden soll.

Die Programmiersprachen der beiden Teile müssen hierbei nicht übereinstimmen. Es kann, wie es in dieser Arbeit der Fall ist, ein Teil in *Objective-C* geschrieben sein und ein anderer in *Switch*. Am Ende sollte Xcode über nachfolgend abgebildete Ordnerstruktur verfügen.

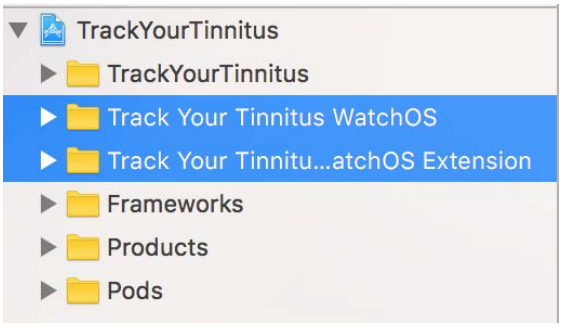

Abbildung 6.3: Ordnerstruktur in Xcode nach Hinzufügen des WatchKit

Die blau markierten Ordner stellen hierbei die neu hinzugefügten dar. Es ist zwar möglich, dass beide Projektteile in verschiedenen Programmiersprachen verfasst sind, jedoch zieht dies einige Konsequenzen mit sich, welche in den nachfolgenden Abschnitten genauer erläutert werden.

6.4 Verknüpfung der beiden Quelltexte

Ein wichtiger Punkt ist der Austausch von Daten der App auf dem iPhone und der Erweiterung auf der iWatch. Hierzu müssen die beiden Projekte, also auch deren Quelltexte, miteinander verknüpft werden. Wie bereits in Kapitel 3 erwähnt, ist Track Your Tinnitus in der Programmiersprache *Objective-C* verfasst worden. Dies stellt für Xcode kein Problem dar, da es- wie in Kapitel 2 erwähnt- eine der unterstützten Programmiersprachen ist. Der bisher existente Prototyp des iWatch Projektes wurde in *Switch* erstellt, der eigens entwickelten Programmiersprache von Apple. Es ist nun also notwendig, eine Möglichkeit zu finden diese beiden Projekte zu vereinen. Dies wird durch sogenannte *Bridging-Header* ermöglicht. Apple bezeichnet diese Funktion als *mix and match*, um so

Apps- welche in unterschiedlichen Programmiersprachen erstellt wurden- miteinander kompatibel zu machen[35].

In diesem Fall wird ein Swift Projekt in ein bereits existierendes Objective-C Projekt eingefügt. Sobald der Swift-Code in das Objective-C Projekt eingefügt wurde, kann weiterhin regulärer Objective-C Code verwendet werden, wenn mit Swift-Klassen gearbeitet werden möchte.
Eine Swift-Klasse muss jedoch eine *Unterklasse* einer Objective-C Klasse sein, um in Objective-C verfügbar sein zu können[35]. Durch einen Import können die Inhalte des Swift-Codes in dem Objective-C Projekt verwendet werden[35]. Möglich gemacht wird das durch einen einfachen Import, wie in *Listing 6.2* dargestellt wird.

```
1 #import "ProductModuleName-Swift.h"
```

Listing 6.2: Import im Bridging Header

Wobei `ProductModuleName` ersetzt wird durch den Namen der jeweiligen Klasse, welche importiert werden soll.

Um Objective-C Code in einem Switch Projekt wiederrum verwenden zu können, benötigt man einen sogenannten *Umbrella Header*. In besagtem Umbrella-Header muss jeder Objective-C Header importiert sein, welcher mit Swift verwendet werden soll. Für Swift ist somit jeder importierte Header öffentlich sichtbar. Von da an sind alle Inhalte der Objective-C Dateien für das Swift Projekt verfügbar, ohne dass die betroffenen Objective-C Dateien nochmals seperat importiert werden müssen.

6.5 Verbindung zwischen iPhone und iWatch

Im vorherigen Kapitel wurden bereits zwei Möglichkeiten dargestellt, wie eine Verbindung zwischen einem iPhone und einer iWatch realisiert werden könnte. Es wird nun anhand der Praxis evaluiert, welches Vorgehen sich hierbei als die bessere Möglichkeit darstellt. Nachfolgend werden beide Alternativen, `WatchConnectivity` und `CoreData`, untersucht.

6.5.1 Verbindung mit CoreData

Wie schon in Kapitel 5 dargelegt, existiert mit `CoreData`[33] ein Framework welches mittels SQLite eine gute Abstraktion einer Datenbank bietet. Zwar kann `CoreData` auch auf der iWatch verwendet werden, jedoch stellt sich hierbei recht bald ein Problem ein. Beide Datensätze, sowohl jener auf dem iPhone als auch der auf der iWatch, müssen konsistent gehalten werden. Ein solcher Abgleich der Datensätze ist jedoch aufwendig zu implementieren.

Für dieses Projekt soll die iWatch jedoch eher als Client dienen, während das iPhone als zentrale Datenquelle fungiert und so Zugriff auf den Track Your Tinnitus Server hat. Um eine solche Client-Server Verbindung herzustellen, bietet sich das `WatchConnectivity` Framework an, welches in Kapitel 5 kurz vorgestellt wurde und nun komplett erläutert werden soll.

6.5.2 Verbindung mit WatchConnectivity

Mittels des `WatchConnectivity` Frameworks[32] ist es möglich eine Kommunikation zwischen dem iPhone und der iWatch aufzubauen. Wichtig beim Verwenden des Frameworks ist es die Verbindung zum richtigen Zeitpunkt im *Application-Lifecycle* aufzubauen. Wird die Verbindung zu früh aufgebaut, so gehen die Daten verloren da noch kein *Handler* der Daten verfügbar ist. Wird die Verbindung zu spät aufgebaut, so kommt es ebenfalls zu Datenverlusten. Ein genauer Zeitpunkt zum Erstellen der Verbindung ist je nach Verwendungszweck und Projektaufbau daher unterschiedlich. Nachfolgend ist ein solcher Verbindungsaufbau anhand des `InterfaceControllers` dargelegt.
Hierbei handelt es sich um den `InterfaceControllers` auf Seiten der iWatch. Daher wurde Swift als Programmiersprache verwendet.
Aus Gründen der Übersichtlichkeit wurde darauf verzichtet, den ganzen Quelltext abzubilden. Folglich werden nur die elementar wichtigen Bestandteile dargestellt und erklärt. Das folgende *Listing* stellt hierbei dar, wie die Klasse `WatchConnectivity` eingebunden werden muss, um Zugriff auf die entsprechenden Methoden zu erhalten.

```
1 import WatchKit
2 import Foundation
3 import WatchConnectivity
4
5
6 class InterfaceController: WKInterfaceController,
    WCSessionDelegate {
```

Listing 6.3: Implementierung der Klasse WatchConnectivity

Ist die Klasse nun wie dargelegt eingebunden, kann eine Verknüpfung mit der Uhr und dem iPhone erfolgen. Dies geschieht mittels der nachfolgenden Befehlsabfolge, welche in *Listing 6.4* dargestellt wird.

Betrachtet wird hierbei die Funktion `willActivate()` um einen möglichst zeitnahen Verbindungsaufbau zu garantieren. Damit soll, wie bereits in Kapitel 5 dargestellt, ein Datenverlust umgangen werden.

```
override func willActivate() {

        super.willActivate()

         if (WCSession.isSupported()) {
              session = WCSession.default()
              session.delegate = self
              session.activate()
         }
     }
```

Listing 6.4: Herstellen der Verbindung iWatch-seitig

Bei der Funktion `willActivate()` handelt es sich um eine bereits implementierte Funktion in der `InterfaceController` Klasse. Da diese Funktion aufgerufen wird sobald die App auf der Watch startet, ist es sinnvoll innerhalb dieser Funktion die `WCSession` zu starten.
Als `WCSession` wird hierbei das Objekt bezeichnet, welches es möglich macht, dass Nachrichten zwischen der iWatch und dem iPhone ausgetauscht werden können[32]. Dies geschieht mittels einer `IF-Abfrage`, da überprüft werden soll ob es sich bei dem zu koppelnden Gerät überhaupt um ein unterstütztes Gerät handelt. Eine solche Abfrage erweist sich als sinnvoll, sollte die Verbindung zur Uhr nicht mehr bestehen, oder aber von einem Gerät erfolgen welches sich nicht als unterstützt herausstellt, beispielsweise ein *iPad*.
Würde auf diese `IF-Abfrage` verzichtet werden, so bestünde also die Gefahr eines Verlustes von Daten.

Nachfolgend, in Listing 6.5 wird nun dargelegt, wie eine solche Verbindung auf Seiten des iPhone erfolgen kann. Da es sich bei dieser konkreten Umsetzung um ein iOS Projekt in Objective-C handelt wird nochmals eine seperate Erläuterung erfolgen.

```
#import <WatchKit/WatchKit.h>
#import <WatchConnectivity/WatchConnectivity.h>

(...)

- (void)applicationDidFinishLaunching {

    if ([WCSession isSupported]) {
        WCSession *session = [WCSession defaultSession];
        session = [WCSession defaultSession];
        session.delegate = self;
        [session activateSession];

    }
   }
}
```

Listing 6.5: Herstellen der Verbindung iOS-seitig

Dieser, abgesehen von der Syntax, relativ synchrone Verbindungsaufbau garantiert auch eine relative Konsistenz der Daten. Der Grund liegt auch hierbei am Zeitpunkt des Aufbaus der Verbindung, zum Start der Applikation. Zu beachten ist dass beide Klassen, `WatchKit` und `WatchConnectivity`, eingebunden werden müssen. Ist dies bei beiden Partnern, iOS und WatchKit, erfolgt, können nun die Methoden der Klasse verwendet werden, welche nachfolgend vorgestellt werden.

Methoden der Klasse WatchKit

Mittels der `sendMessage` Methode ist es möglich, Nachrichten und Datenpakete abzusenden. Eben diese Datenpakete lassen sich durch eine ID identifizieren. Mithilfe eines `reply Handlers` ist nun möglich eine direkte Antwort auf ein solches Paket abzusenden. Hiermit kann eine Art des direkten Datentransfers ermöglicht werden[36]. Eine weitere Form der Datenübertragung wird mit der Methode `transferUserInfo` zur Verfügung gestellt. Hierbei werden die Daten nicht direkt an die Uhr geschickt, sondern erst zum optimalen Zeitpunkt. Als Gradmesser einer optimalen Verbindung gelten sowohl die Verbindungsstärke zur Uhr, als auch eine mögliche WLAN Verbindung.
So sind die Daten beim Start der Applikation für die Uhr direkt verfügbar. Anzumerken ist hierbei auch, dass die Daten weiterübertragen werden, selbst wenn die App nur noch im Hintergrund aktiviert ist, oder aber komplett terminiert wurde.

Umsetzung

Es folgt eine Beschreibung wie die Methoden günstig eingesetzt werden können.
Möchte ein direkter Datentransfer ermöglicht und somit die Antworten des Fragebogen direkt gespeichert werden, so würde sich die Methode `sendMessage`[36] anbieten.
Wird jedoch ein versenden der Daten in einem großen Paket präferiert, so würde sich mit `transferUserInfo`[36] eine gute Alternative finden lassen, welche vor allem einen sicheren Datenempfang versprechen würde.
Bei der konkreten Umsetzung kann anhand der oben beschriebenen Merkmale jedoch eine Orientierung erfolgen.

6.5.3 Modellierung des Programmablaufs

Eine Anforderung an das Projekt war es, dass die App durch bestimmte Personen erweiterbar sein soll. Dies bedeutet konkret, dass Fragebögen oder auch einzelne Fragen hinzugefügt werden können. Da Apple sowohl für iOS, als auch für watchOS, die Anwendung von *Model View Controller* als Entwicklungsgrundlage empfiehlt[37], kann

dies wie folgt umgesetzt werden.
Jede View wird in Xcode in dem sogenannten `Interface.storyboard` hinterlegt. Besagtes Storyboard ist im WatchOS Ordner auffindbar. Hier wird auch festgelegt in

Abbildung 6.4: Ein Screenshot des Storyboards

welcher Reihenfolge die jeweiligen Views angezeigt werden sollen. Sollen nun weitere Fragen oder gar Fragebögen hinzufügt werden, so geschieht dies mittels dem Ergänzen einer View und dem Abändern des bisherigen Ablaufs. Die Funktionalität erhalten die jeweiligen Views durch ihre zugehörigen Controller. In WatchOS existiert mit der Klasse `WKInterfaceController` der Supertyp für jeden `ViewController`[38]. So gilt es auch darauf zu achten, dass bei einer Erweiterung des Projekts um weitere Views auch die Controller angepasst werden. Ein Beispiel ist hierbei die Methode `awakewithContext`, wie in *Listing 6.6* dargestellt.

```
1 func awake(withContext context: Any?)
```

Listing 6.6: Die Funktion awake(withContext)

Diese Methode erlaubt es den Controller mit spezifischen Daten erscheinen zu lassen, die in einer vorherhigen View eingegeben wurden.
Dies gilt es beim Ablauf des Programms zu beachten.

7

Führung durch die App

In diesem Kapitel soll die Applikation für die iWatch kurz vorgestellt werden. Hierzu wird, mittels Screenshots der Applikation, eine kurze Vorstellung der jeweiligen Anzeigen gegeben.
Unabhängig von der Wahl der Eingabe bleibt die Reihenfolge der ablaufenden Anzeigen absolut identisch.

Soll die App gestartet werden, so geschieht dies auf der iWatch mittels des dazugehörigen Icons, hier in Abbildung 7.1 zentral dargestellt.

Abbildung 7.1: Homebildschirm der iWatch mit der TYT App zentral

Durch das Öffnen der App, welches wie auf dem Smartphone durch ein Drücken des Icons geschieht, erhält der Nutzer nachfolgende Ansicht 7.2.

Abbildung 7.2: Erster Bildschirm der Applikation

Hierbei wird der Nutzer befragt ob aktuell eine Tinnitus-Wahrnehmung stattgefunden hat.

Wurde die bisherige Lösung mittels einer *Binäreingabe* realisiert, erfolgt nun die Realisierung mittels eines *Sliders*. Anhand dieses Sliders kann der Nutzer einstellen, in welchem Maße der eben wahrgenommene Tinnitus belastend war.

Abbildung 7.3: Abfrage bezüglich des Belastungsgrades des Tinnitus

Die folgenden drei Abbildungen wurden alle mithilfe einer *Single-Choice* Auswahl realisiert. Hierbei kann der Nutzer mittels graphisch visualisierten Antwortmöglichkeiten festlegen, wie die aktuelle Stimmungslage ist, siehe Abbildung 7.4. In Abbildung 7.5 wird der Grad der Aufregung abgefragt. Abschließend wird in Abbildung 7.6 der aktuelle Stresslevel erhoben.

Abbildung 7.4: Abfrage bezüglich der aktuellen Stimmungslage

Abbildung 7.5: Abfrage bezüglich des aktuellen Grads der Aufregung

Abbildung 7.6: Abfrage bezüglich des aktuellen Empfindens des Stressgrades

Anschließend wird der Nutzer weitergeleitet auf die Frage wie hoch der aktuelle Grad seiner Konzentration war. Diese Abfrage wird mittels eines *Pickers* realisiert. Durch diese Alternative wird es möglich, einen genauen Grad des Konzentrationsniveaus abzufragen.

Abbildung 7.7: Anzeige zur Eingabe des Konzentrationsniveaus

Die letzte Dateneingabe ist nun wieder *binärer Natur*. Hierbei wird der Nutzer gefragt ob er sich ausgeschlafen fühle.

Abbildung 7.8: Abfrage des Nutzers ob er ausgeschlafen ist

Die letzte Anzeige der Applikation ist eine Bestätigungsanzeige, dass der Fragebogen nun beendet ist. Der Nutzer ist angehalten auf den Button *Speichern* zu klicken um seine eingegebenen Daten zu sichern.

Abbildung 7.9: Anzeige für das Ende des Fragebogens

8

Anforderungsabgleich

In nachfolgendem Kapitel werden die in Kapitel 4 aufgestellten Anforderungen untersucht und auf ihre Umsetzung überprüft.
Zunächst werden hierzu die *funktionalen* Anforderungen betrachtet. Anschließend wird der Fokus auf die *nicht-funktionalen* Anforderungen gelegt.

Dabei wird ein spezieller Fokus bei den funktionalen Anforderungen auf die Umsetzung gelegt, bei den nicht-funktionalen Anforderungen hingegen wird überprüft ob diese im Rahmen dieser Arbeit umgesetzt wurden.

8.1 Funktionale Anforderungen

Es werden erneut die funktionalen Anforderungen aus Kapitel 4 aufgeführt. Die aufgeführten Funktionen werden mit der Realisierung verglichen.

Nr.	Bezeichnung	Umsetzung
1	Anzeigen des Fragebogens	Erfüllt (siehe Kapitel 6)
2	Beantworten von Fragebögen	Erfüllt (siehe Kapitel 6)
3	Erweiterbarkeit	Erfüllt (siehe Kapitel 5)
4	Konsistenz der Daten	Nicht Erfüllt
5	Änderung der gegebenen Antworten	Erfüllt (siehe Kapitel 6)

Tabelle 8.1: Überprüfung der funktionalen Anforderungen

8.2 Nicht-funktionale Anforderungen

Abschließend werden die nicht-funktionalen Anforderungen betrachtet und abgeglichen.

Nr.	Bezeichnung	Umsetzung
1	Technische Anforderungen	Erfüllt. Das System ist unter WatchOS 2 oder neuer lauffähig. Durch eine Optimierung der Podfile (Kapitel 5) ist auch eine neuere iOS Version möglich.
2	Verfügbarkeit	Erfüllt. Selbst im nicht-gekoppelten Zustand mit einem iPhone ist die App lauffähig. Für das Speichern der Daten ist eine Kopplung jedoch unerlässlich.
3	Stabilität und Robustheit	Erfüllt. Da die App, wie bereits der Prototyp, eine niedrige Komplexität aufweist können mögliche Fehlerquellen gering gehalten werden.
4	Nutzerfreundlichkeit	Erfüllt. Durch das einfache Design wird eine einfache und selbsterklärende Bedienung ermöglicht.
5	Watch-Kit Erweiterung im existenten iOS Projekt	Erfüllt. Siehe hierzu Kapitel 5.
6	Erweiterung der existenten Applikationen	Diese Bedingung wurde teilweise erfüllt. Eine Verbindung der beiden Apps wurde realisiert (siehe Kapitel 5). Ein Datenaustausch, wie bei dem funktionalen Anforderungsabgleich bereits erwähnt, findet jedoch nicht statt.

Tabelle 8.2: Überprüfung der nicht-funktionalen Anforderungen

9

Zusammenfassung

In diesem Kapitel werden die wichtigsten Aspekte der Arbeit aufgegriffen und die Besonderheiten hervorgehoben. Mit Kapitel 9.2 erfolgt ein Ausblick auf mögliche Erweiterungen der Anwendung.

Zu Beginn der Arbeit wurde die Erkrankung Tinnitus vorgestellt. Meistens werden damit Störgeräusche im Ohr verbunden. Man unterscheidet hierbei zwischen einem subjektiven und einem objektiven Tinnitus.
Es gibt verschiedene Herangehensweisen eine Tinnituserkrankung zu behandeln. Eine Möglichkeit stellt hierbei das Track Your Tinnitus Projekt dar. Hierbei wird mittels einer eigens entwickelten Anwendung die Tinnituswahrnehmung aufgezeichnet.
Realisiert wird das mithilfe von speziell entwickelten Fragebögen. Ein solches Fragebogensystem wurde bereits als Prototyp für die iWatch entwickelt.
Um diesen nun zu realisieren mussten zuvor Anforderungen aufgestellt werden, welche die Anwendung erbringen soll. Hierbei fand eine deutliche Trennung zwischen funktionalen und nicht-funktionalen Anforderungen statt. Besagte Anforderungen wurden in Kapitel 4 dargestellt und anschließend in Kapitel 8 auf ihre Umsetzung geprüft.

9.1 Fazit

Das Ziel dieser Arbeit war es das bisher als Prototypen umgesetzte Fragebogensystem für die iWatch komplett funktional umzusetzen. Hierbei wurde untersucht, wie eine funktionierende iOS Anwendung um eine iWatch Funktion erweitert werden kann.
Es wurde hierfür zunächst überprüft, welcher Fragebogen in der endgültigen Realisierung verwendet werden könnte.

Anhand der Studienergebnisse der prototypischen Realisierung wurde die Wahl des Fragebogens konkretisiert. Die Wahl ist hierbei auf die zweite Variante gefallen. Anschließend wurden Möglichkeiten untersucht um die beiden bisherigen Arbeiten, die Umsetzung der TrackYourTinnitus iOS App und den iWatch Prototypen, miteinander zu verbinden.

Hierbei musste zuerst die iOS App abgeändert werden, da es hierbei zu Fehlern mit den CocoaPods gekommen war. Anschließend wurde der iOS App eine WatchKit Erweiterung hinzugefügt. Um eine Verbindung zwischen den beiden Teilen zu ermöglichen wurden zwei Varianten analysiert. CoreData und WatchConnectivity. Da WatchConnectivity über deutlich mehr Methoden verfügt und die iWatch eher als Erweiterung zum zugehörigen iPhone dienen soll, wurde WatchConnectivity verwendet.

Sollte man den existenten Fragebogen ändern oder das ganze System erweitern wollen, so ist dies aufgrund des Model View Controller Aufbaus ebenso möglich.

Ein Speichern der eingegeben Daten ist ebenso möglich, konnte im Rahmen dieser Arbeit aber nicht realisiert werden. Abschließend wurde die Applikation nochmal in einem Walkthrough vorgestellt.

Zusammenfassend lässt sich also sagen dass eine Realisierung eines Fragebogensystems für die iWatch als Erweiterung des iPhones realisierbar ist. Da ein solches Fragebogensystem auf der iWatch einzigartig ist, kann es als Motivation angesehen werden ein solches MCS System auch in Verbindung mit anderen Arten von Fragebögen umzusetzen.

9.2 Ausblick

Die Verbindung zwischen der iPhone App und der iWatch Applikation wurde in dieser Arbeit umgesetzt. Da diese Art der Verbindung streng dem Model View Controller Konzept folgt, ist eine Erweiterung dieser Arbeit zu jedem Zeitpunkt möglich. Es können hierbei sowohl neue Fragen und sogar Fragebögen hinzugefügt werden. Wenn nötig sogar neue Funktionen.

Da die Tinnituserkrankung das Leben der Patienten in den unterschiedlichsten Phasen des Alltags stark beeinflusst, ist es wichtig den Verlauf der Erkrankung zu überwachen. Da die iWatch über Sensoren für die Herzfrequenz verfügt, wäre es möglich eine Abfrage eben dieser in ein Fragebogensystem einzubauen.
Mit dem Erscheinen der vierten Instanz der iWatch im Herbst 2017 könnten entwaige Neuerungen ebenso bedacht werden.

Außerdem ist eine Verwendung außerhalb der körperlichen Gesundheitsbranche denkbar. So kann durch das Verwenden von Wearables eine deutlich größere Anzahl an Daten erhoben und auch analysiert werden. Dies macht ein solches System vor allem bei psychologischen Erhebungen sehr interessant.
Denn die direkte Reaktion sorgt für eine unverfälschte Wiedergabe der Daten.

Literaturverzeichnis

[1] Frank, Wilhelm und Konta, B.u.S.G.: Therapie des unspezifischen Tinnitus ohne organische Ursache. Deutsches Institut für Medizinische Dokumentation und Information (2006)

[2] tinnitracks.de: Internetauftritt der Plattform Tinnitracks. „http://www.tinnitracks.com/de“ (13.08.2017) [Online; Aufgerufen am 13-August-2017].

[3] : Internetauftritt des Tinnituszentrum der Charite. „https://tinnituszentrum.charite.de/patienten/tagesklinik/“ (13.08.2017) [Online; Aufgerufen am 13-August-2017].

[4] Marek, A.: Psychosomatik in der HNO-Heilkunde. Thieme-Verlag (2009)

[5] Pryss, R., Probst, T., Schlee, W., Schobel, J., Langguth, B., Neff, P., Spiliopoulou, M., Reichert, M.: Mobile Crowdsensing for the Juxtaposition of Realtime Assessments and Retrospective Reporting for Neuropsychiatric Symptoms. In: 30th IEEE International Symposium on Computer-Based Medical Systems (CBMS 2017), IEEE Computer Society Press (2017)

[6] Heise.de: Umfrage: 78 Prozent der Deutschen nutzen Smartphone. https://www.heise.de/newsticker/meldung/Umfrage-78-Prozent-der-Deutschen-nutzen-Smartphones-3632629.html (22.2.2017) [Online; Aufgerufen am 13-August-2017].

[7] Quast, C.: Smartphone als Krankmacher (2016) [Online; Aufgerufen am 13-August-2017].

[8] Moser, L.E., Melliar-Smith, P.M.: Personal Health Monitoring Using a Smartphone. In: 2015 IEEE International Conference on Mobile Services. (2015) 344–351

[9] Elsässer, V.: Konzeption und Realisierung für ein mobiles Smartwatch-Fragebogensystem zur Erfassung klinischer Parameter Tinnitus-geschädigter Patienten. Bachelorarbeit (2016)

[10] Magazin, T.: Psychische Ursachen für Tinnitus. (https://www.tinnitus-mag.de/ursachen-von-tinnitus/tinnitus-und-psyche/) [Online; Aufgerufen am 13-August-2017].

[11] Tinnitracks: Ursachen von tinnitus. (http://www.tinnitracks.com/de/tinnitus/ursachen) [Online; Aufgerufen am 13-August-2017].

[12] Deutsche Tinnitus Liga: Prävalenz des Tinnitus. (https://www.tinni.net/epidemiologie.htm) [Online; Aufgerufen am 13-August-2017].

[13] Probst, T., Pryss, R., Langguth, B., Rauschecker, J., Schobel, J., Reichert, M., Spiliopoulou, M., Schlee, W., Zimmermann, J.: Does tinnitus depend on time-of-day? An ecological momentary assessment study with the TrackYourTinnitus application. Frontiers in Aging Neuroscience **9** (2017) 253–253

[14] Tinnitracks: Behandlung von Tinnitus. http://www.tinnitracks.com/de/tinnitus/behandlung (2013) [Online; Aufgerufen am 13-August-2017].

[15] Tibez: Tinnitus Fragebogen. (https://www.tibez.de/tinnitus-fragebogen.html) [Online; Aufgerufen am 13-August-2017].

[16] Pryss, R., Schlee, W., Langguth, B., Reichert, M.: Mobile Crowdsensing Services for Tinnitus Assessment and Patient Feedback. In: 6th IEEE International Conference on AI & Mobile Services (IEEE AIMS 2017), IEEE Computer Society Press (2017)

[17] Questionsys: Website des QuestionSys Projekt. (https://www.uni-ulm.de/in/iui-dbis/forschung/laufende-projekte/questionsys/) [Online; Aufgerufen am 13-August-2017].

[18] SeNDA: SeNDA- Security of Networks and Distributed Applications. (http://senda.uab.es/node/15) [Online; Aufgerufen am 13-August-2017].

[19] Schickler, M., Pryss, R., Reichert, M., Heinzelmann, M., Schobel, J., Langguth, B., Probst, T., Schlee, W.: Using Wearables in the Context of Chronic Disorders - Results of a Pre-Study. In: 29th IEEE Int'l Symposium on Computer-Based Medical Systems. (2016) 68–69

[20] Probst, T., Pryss, R., Langguth, B., Spiliopoulou, M., Landgrebe, M., Vesala, M., Harrison, S., Schobel, J., Reichert, M., Stach, M., Schlee, W.: Outpatient Tinnitus Clinic, Self-Help Web Platform, or Mobile Application to Recruit Tinnitus Study Samples? Frontiers in Aging Neuroscience **9** (2017) 113–113

[21] Schlee, W., Pryss, R., Probst, T., Schobel, J., Bachmeier, A., Reichert, M., Langguth, B.: Measuring the Moment-to-Moment Variability of Tinnitus: The TrackYourTinnitus Smart Phone App. Frontiers in Aging Neuroscience **8** (2016) 294–294

[22] Free Tinnitus HQ: iTunes Seite von Free Tinnitus HQ. (`https://itunes.apple.com/de/app/free-tinnitus-hq-gratis-tinnitus-hq-naturklänge-übertönen/id891630056?mt=8`) [Online; Aufgerufen am 13-August-2017].

[23] Herrmann, J.: Konzeption und technische Realisierung eines mobilen Frameworks zur Unterstützung tinnitusgeschädigter Patienten. PhD Thesis (2014)

[24] Bayer, T., Sohn, D.M.: Eine Einführung: REST Web Services. (`http://t3n.de/magazin/rest-web-services-einfuhrung-219976/`) [Online; Aufgerufen am 13-August-2017].

[25] Cocoa Pods Dev-Team: What is Cocoa Pods? (`https://cocoapods.org`) [Online; Aufgerufen am 13-August-2017].

[26] StackOverflow: Advantage of using Cocoa-Pods. (`https://stackoverflow.com/questions/38804692/what-is-the-advantage-of-using-cocoapods`) [Online; Aufgerufen am 13-August-2017].

[27] Ole Begemann: Differences Between Xcode Project Templates for iOS Apps. (`https://oleb.net/blog/2013/05/xcode-project-templates-difference/`) [Online; Aufgerufen am 13-August-2017].

[28] Christian Bleske: (iOS-Apps programmieren mit Swift) (dpunkt.verlag (2016)).

[29] Apple: Switch between apps using the Dock on your Apple Watch. (https://support.apple.com/en-us/HT206992) [Online; Aufgerufen am 13-August-2017].

[30] StackOverflow: Watch Kit app with Watch OS 1 and 2. (https://stackoverflow.com/questions/33208199/watch-kit-app-with-watch-os-1-and-2) [Online; Aufgerufen am 13-August-2017].

[31] Ralf Ebert: watchOS 3 Tutorial: Kommunikation zwischen iPhone und Apple Watch. (https://www.ralfebert.de/tutorials/watchos-watchkit-connectivity/) [Online; Aufgerufen am 13-August-2017].

[32] Apple.com: WatchConnectivity. (https://developer.apple.com/documentation/watchconnectivity) [Online; Aufgerufen am 13-August-2017].

[33] Apple.com: What Is Core Data? (https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/CoreData/index.html) [Online; Aufgerufen am 13-August-2017].

[34] Ralf Ebert: 13. Bibliotheken über CocoaPods einbinden. (https://www.ralfebert.de/ios/cocoapods/) [Online; Aufgerufen am 13-August-2017].

[35] Apple.com: Swift and Objective-C in the Same Project. (https://developer.apple.com/library/content/documentation/Swift/Conceptual/BuildingCocoaApps/MixandMatch.html) [Online; Aufgerufen am 13-August-2017].

[36] stackoverflow.com: WatchConnectivity - using sendMessage. (https://stackoverflow.com/questions/34051505/watchconnectivity-using-sendmessage) [Online; Aufgerufen am 13-August-2017].

[37] Apple Developer: Model-View-Controller. (https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html) [Online; Aufgerufen am 13-August-2017].

[38] Apple Developer: WKInterfaceController. (https://developer.apple.com/documentation/watchkit/wkinterfacecontroller) [Online; Aufgerufen am 13-August-2017].

Abbildungsverzeichnis

3.1 Aufbau des Track Your Tinnitus Projektes 12

6.1 PodFile Fehlermeldung 30
6.2 Hinzufügen eines WatchKit Targets in Xcode 32
6.3 Ordnerstruktur in Xcode nach Hinzufügen des WatchKit 33
6.4 Ein Screenshot des Storyboards 40

7.1 Homebildschirm der iWatch mit der TYT App zentral 42
7.2 Erster Bildschirm der Applikation 43
7.3 Abfrage bezüglich des Belastungsgrades des Tinnitus 44
7.4 Abfrage bezüglich der aktuellen Stimmungslage 45
7.5 Abfrage bezüglich des aktuellen Grads der Aufregung 46
7.6 Abfrage bezüglich des aktuellen Empfindens des Stressgrades 47
7.7 Anzeige zur Eingabe des Konzentrationsniveaus 48
7.8 Abfrage des Nutzers ob er ausgeschlafen ist 49
7.9 Anzeige für das Ende des Fragebogens 50

Tabellenverzeichnis

4.1 Übersicht der funktionalen Anforderungen 21
4.2 Übersicht der nicht-funktionalen Anforderungen 22

8.1 Überprüfung der funktionalen Anforderungen 52
8.2 Überprüfung der nicht-funktionalen Anforderungen 53

Name: Sebastian Fuchs Matrikelnummer: 741896

Erklärung

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

Ulm, den 15.08.2017 .....

Sebastian Fuchs