ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

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

Evaluierung der Google Cardboard API anhand einer mobile medical App

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Lukas Stöferle
lukas.stoeferle@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Marc Schickler

Wintersemester 16/17

Fassung 29. März 2017

Satz: PDF-LaTeX 2$_\varepsilon$

Abstract

Die Omnipräsenz von mobilen Geräten ermöglicht neue und innovative Wege die Gesundheit der Bevölkerung zu fördern. Neben Anwendungen für Fitness und Ernährung finden auch immer mehr medizinische Anwendungen Verwendung auf dem Smartphone, die dem Benutzern bei der Bewältigung bestimmter Krankheiten helfen können. Tinnitus ist eine dieser Krankheiten für die es momentan kein Heilmittel gibt, die sich aber immer mehr und mehr verbreitet. Es existieren lediglich verschiedene Therapieansätze, die eine Verbesserung der Tinnitus-Belastung ermöglichen.

Ziel dieser Arbeit ist die Entwicklung einer mobilen Anwendung, die das Konzept der Tinnitus-Retraining-Therapie spielerisch umsetzt. Durch räumliches Hören soll der Benutzer die Schallquelle von bestimmten Geräuschen lokalisieren. Somit soll erreicht werden, dass der Benutzer den Tinnitus durch auditive Stimulation vorübergehend nicht mehr wahrnimmt. Eine begleitende Studie vergleicht die Umsetzung mit weiteren Anwendungen, die das selbe Konzept verfolgen. Die aus der Anwendung gewonnenen Erkenntnisse werden zum Schluss zusammengefasst und zusätzlich werden einige Verbesserungsmöglichkeiten genannt.

Danksagung

An dieser Stelle möchte ich mich zuerst bei *Herrn Prof. Dr. Manfred Reichert* für die Begutachtung dieser Bachelorarbeit herzlich bedanken. Zugleich gebührt ein besonderer Dank meinem Betreuer *Marc Schickler*, der mich jederzeit mit großer Hilfsbereitschaft unterstützt hat.

Außerdem möchte ich mich bei *Simon Stöferle* und *Sophia Stöferle* für das Korrekturlesen recht herzlich bedanken.

Zuletzt gilt allen *Studienteilnehmern/innen*, die sich für die Teilnahme an der begleitenden Studie bereit erklärt haben, großen Dank.

Inhaltsverzeichnis

1

Einleitung

Das menschliche Ohr ist ein vielseitiges Sinnesorgan, das sowohl die Verständigung unter Menschen aber auch die räumliche Orientierung durch auditive Wahrnehmung ermöglicht. Durch starke Lärmbelastung kann in vielen Fällen eine Erkrankung des Ohrs in Form von Tinnitus auftreten. Zwischen 5% und 15% der Gesamtbevölkerung sind von Tinnitus betroffen, wobei dieser bei rund 1% der Betroffenen eine exzessive Lautstärke vorweist, welche die Lebensqualität erheblich beeinträchtigt [1].

Mit Hilfe der mobilen Anwendung „Track your Tinnitus“ können Betroffene die individuellen Schwankungen ihrer Tinnituswahrnehmung erfassen. Mit einem Smartphone können die Schwankungen des Tinnitus systematisch gemessen werden. Somit kann herausgefunden werden, wie diese mit dem Tagesablauf und den Aktivitäten der Betroffenen zusammenhängen [2]. Die Anwendung wurde von der Tinnitus Research Initiative (TRI) in Kooperation mit dem Institut für Datenbanken und Informationssysteme der Universität Ulm entwickelt und soll in Zukunft durch weitere Funktionalitäten ergänzt und verbessert werden.

Benutzer sollen in Zukunft die Tinnitus-Belastung spielerisch verbessern können. Als Inspiration diente das Spiel „Audio Defence: Zombie Arena“, welches für die iOS Plattform erhältlich ist [3]. Hierbei verzichtet das Spiel komplett auf eine grafische Darstellung der sich nähernden Gegner. Lediglich die Geräusche aus dem Kopfhörer verraten dem Spieler aus welcher Himmelsrichtung sich die Zombies langsam annähern, die es zu bezwingen gilt. Bereits im vergangenen Jahr wurden Anwendungen für die Plattformen Android, iOS, WindowsPhone und Web (HTML5) entwickelt, die dieses Konzept für medizinische Zwecke verwenden [4, 5, 6, 7]. Nun soll eine weitere Android Anwendung entwickelt werden, die dieses Konzept mit aktueller Technik realisiert.

1.1 Ziel der Arbeit

Diese Arbeit beschäftigt sich mit der Entwicklung einer medizinischen Anwendung für Android Smartphones, welche die Tinnitus-Belastung der betroffenen Personen spielerisch verbessert. Mit Hilfe der *Google VR SDK* werden Geräusche in einem virtuellen Raum um den Spieler herum platziert, die über Kopfhörer wahrgenommen werden können [8]. Die Aufgabe des Spielers besteht darin die Richtung, aus der das Geräusch kommt, zu lokalisieren. Durch gezielte auditive Stimulation soll der Tinnitus für kurze Zeit vom Betroffenen nicht mehr wahrgenommen werden. Für eine intuitive Bedienung werden zur Positionsbestimmung die im Gerät verbauten Sensoren verwendet, die durch das Drehen und Neigen des Smartphones ein neuartiges Bedienkonzept ermöglichen [9]. Für Abwechslung sorgen drei Schwierigkeitsgrade, sowie eine Auswahl von Geräusche-Themen und weitere Einstellungsmöglichkeiten.

Alle erhobenen Daten werden im JSON-Format gespeichert, um die spätere Auswertung in einer begleitenden Studie zu erleichtern, in der die Anwendung mit anderen bereits existierenden Anwendungen verglichen wird.

1.2 Struktur der Arbeit

Die vorliegende Arbeit ist in insgesamt sieben Kapitel unterteilt. Zu Beginn werden in Kapitel 2 wichtige Grundlagen thematisiert, die im Laufe dieser Arbeit immer wieder aufgegriffen werden. Anschließend wird in Kapitel 3 die Planungsphase des Projekts genauer beschrieben, indem die Anforderungen, welchen die Anwendung entsprechen soll, aufgeführt und erläutert werden. Kapitel 4 befasst sich mit der Architektur und der Implementierung der Anwendung. Im Anschluss wird in Kapitel 5 die implementierte Anwendung auf die Einhaltung der zuvor gestellten Anforderungen untersucht und bewertet. Der Aufbau, Ablauf sowie die Ergebnisse der begleitenden Studie werden in Kapitel 6 genauer beschrieben. Abschließend werden in Kapitel 7 die wesentlichen Bestandteile dieser Arbeit zusammengefasst, gefolgt von einem Ausblick auf potentielle Verbesserungen und Erweiterungen der Anwendung.

2

Grundlagen

Im Folgenden Kapitel werden grundlegende Aspekte, die für ein besseres Verständnis dieser Arbeit beitragen, näher erläutert. Da die entstandene Anwendung sich auf die Zielgruppe der von Tinnitus betroffenen Menschen spezialisiert, wird daher zunächst die Gehörerkrankung selbst thematisiert. Daraufhin beschäftigt sich dieses Kapitel mit der virtuellen Realität, die sich seit einigen Jahren immer mehr und mehr zum Massenphänomen entwickelt. Zusätzlich werden die in der Anwendung verwendeten Komponenten zur Realisierung der virtuellen Realität genannt und beschrieben.

2.1 Tinnitus

Ein Tinnitus (lat. für „Ohrenklingen“) ist ein Symptom für eine Erkrankung des Gehörs und beschreibt die Wahrnehmung von Geräuschen, deren Herkunft nicht auf eine äußere physikalische Schallquelle zurückzuführen sind [10]. In Deutschland leiden etwa 4% der Bevölkerung unter Tinnitus, wobei dieser bei rund 15% zu weiteren Begleitsymptomen wie Schlafstörungen, Konzentrationsstörungen oder Überempfindlichkeit für laute Geräusche führt. In diesem Fall spricht man auch von einem dekompensierten Tinnitus. Die Geräusche werden von den Betroffenen Personen als sehr hohe Töne in Form von Piepsen, Pfeifen oder auch Zischen sowohl einseitig, als auch auf beiden Ohren wahrgenommen [11].

In der Medizin unterscheidet man zwischen dem *subjektiven* und *objektiven* Tinnitus. Der objektive Tinnitus ist strenggenommen kein Tinnitus, da die wahrgenommenen akustischen Signale durch den Organismus der betroffenen Personen erzeugt werden.

Dieser tritt aber nur bei weit weniger als 0,1% der Tinnitusbetroffenen auf und kann beispielsweise mit einem Stethoskop über dem Kopf oder Gehörgang vom Arzt gehört werden. Verursacht wird der objektive Tinnitus zum Beispiel durch einen Tumor oder durch das Zucken der Muskeln der Gehörknöchelchen. In den meisten Fällen kann diese Art von Tinnitus durch medikamentöse Behandlung oder eine Operation behandelt werden [12, 13].

Der subjektive Tinnitus ist die am meisten diagnostizierte Form von Tinnitus und tritt in der Regel mit einer Hörstörung auf. Die wahrgenommenen Geräusche sind nicht auf eine äußere Schallquelle zurückzuführen. Häufig werden diese durch eine Überaktivität von Nervenzellen ausgelöst und aufrechterhalten [13]. Bei akutem Tinnitus kann das Geräusch sowohl nur wenige Minuten lang aber auch über mehrere Wochen hinweg von den Betroffenen wahrgenommen werden, bevor es selbstständig verschwindet. Sobald das Geräusch nach drei Monaten immer noch wahrgenommen wird, ist vom chronischen Tinnitus die Rede. Die dauerhafte Heilung eines chronischen Tinnitus ist bisher nicht möglich, dieser kann jedoch durch verschiedene Therapien behandelt werden [10].

2.1.1 Verhaltenstherapeutische Ansätze

Durch das gezielte Aufarbeiten, Überprüfen und Bewerten von Sachverhalten soll diese Art von Therapie letztendlich zu einer positiven Veränderung des konkreten Verhaltens führen. Ein bekannter Ansatz ist die Kognitive Verhaltenstherapie, die sich mit den Einstellungen und Gedanken der Patienten im Hinblick auf die Krankheit beschäftigt [14].

2.1.2 Medikamentöse Therapieansätze

Durch medikamentöse Behandlung kann der Tinnitus reduziert werden, jedoch haben sich diese Therapien wegen erheblichen Nebenwirkungen nicht durchgesetzt [12].

2.1.3 Tailor-Made Notched Music Training

Eine Musiktherapie, bei der die Frequenz des Tinnitus bestimmt wird, die dann aus der Lieblingsmusik der Patienten herausgefiltert und verändert wird. Diese Musik muss vom Patienten über mehrere Stunden am Tag gehört werden. Erste Untersuchungen führten bei einigen Patienten zu einer Verbesserung der Tinnitus-Belastung [15].

2.1.4 Tinnitus-Retraining-Therapie

Mit Hilfe von Rauschgeneratoren oder anderen Umweltgeräuschen soll ein Zustand erreicht werden, in dem der Betroffene den Tinnitus nicht mehr bewusst wahrnimmt. Diese Art von Therapie relativiert den Begriff Heilung (Abwesenheit von Tinnitus) und führte sogar zum kompletten Verschwinden des bisher chronischen Tinnitus [16].

Die Anwendung, die in dieser Arbeit entsteht, beschäftigt sich mit dem Ansatz der Tinnitus-Retraining-Therapie in Kombination mit virtueller Realität.

2.2 Virtuelle Realität

Der Begriff *virtual reality (VR)* (engl. für „virtuelle Realität“) steht für eine computergenerierte virtuelle Umgebung, die gleichzeitig als Benutzeroberfläche fungiert. Der Benutzer kann eine Anwendung innerhalb der virtuellen Umgebung steuern und sich im Idealfall so wie in seiner bekannten Umgebung verhalten [17].

Häufig werden sogenannte Head-Mounted Displays als Ausgabegeräte verwendet, die vom Benutzer auf dem Kopf getragen werden und die durch eingebaute Sensoren das Erfassen der Kopfbewegungen (Head-Tracking) ermöglichen. Somit kann die virtuelle Umgebung je nach Position des Kopfes angepasst und verändert werden. Die virtuelle Umgebung muss für beide Augen separat berechnet werden, da diese durch die unterschiedlichen Blickwinkel verschiedene Bilder wahrnehmen, die im Gehirn zu einem Gesamtbild zusammengesetzt werden (Binocular Rendering). Dadurch wird eine deutlich

höhere Rechenleistung benötigt, die meist von einem externen Computer übernommen wird.

Ein ähnliches Prinzip wird auch bei der *Google Cardboard* VR-Brille verfolgt, die Ausgabe und Berechnung der virtuellen Umgebung vereint [18]. Die Brille an sich dient als Halterung für Smartphones, die aus Karton besteht und zwei Linsen sowie einen Magnetschalter vorweist (siehe Abbildung 2.1). Die ins Cardboard eingesetzten Smartphones übernehmen die Ausgabe und Berechnung der virtuellen Umgebung. Das Head-Tracking erfolgt über die Verwendung der im Gerät verbauten Sensoren. Über den Magnetfeldsensor erkennt das Gerät, ob der Benutzer den Magnetschalter betätigt. Sowohl die Anwendungen als auch die Smartphones müssen bestimmte Kriterien erfüllen, damit diese für Google Cardboard verwendet werden können. Neben den benötigten Sensoren sollten die Geräte eine hohe Rechenleistung vorweisen, damit eine flüssige Wiedergabe der VR-Inhalte möglich ist.

Abbildung 2.1: Google Cardboard [19]

Für die Entwicklung von VR-Anwendungen gibt es eine Vielzahl an verschiedenen Frameworks. Bekannte Vertreter in diesem Genre sind die zwei Grafik-Engines *Unity3D* sowie *Unreal Engine*, die für die Entwicklung von Spielen auf mehreren Plattformen verwendet werden [20, 21]. Die in dieser Arbeit entwickelte Anwendung verwendet jedoch die Programmierschnittstelle *Google VR*, welche im folgenden Abschnitt erläutert wird.

2.2.1 Google VR

Mit der Programmierschnittstelle *Google VR* können Entwickler auf ein modernes und umfangreiches Framework zurückgreifen, welches die von Google Cardboard benötigten Anforderungen abwickelt [8]. Die Schnittstelle wird neben den Plattformen Android und iOs auch für die Grafik-Engines Unity3D und Unreal Engine bereitgestellt. Mit Hilfe verschiedener UI-Komponenten können Entwickler einfache VR-Anwendungen, beispielsweise zur Betrachtung von 360 Grad Panoramabildern oder Videos, erstellen. In Kombination mit der Grafik-Pipeline *OpenGL ES* können auch virtuelle Umgebungen erstellt werden [22].

Durch die Verwendung der zusätzlich bereitgestellten 3D-Audio Komponente kann die Aufmerksamkeit des Benutzers gezielt durch auditive Stimulation kontrolliert werden. Diese Komponente ist essenzieller Bestandteil der in dieser Arbeit entstandenen Anwendung und wird nun im folgenden Abschnitt thematisiert.

2.3 Spatial Audio

Die *GvrAudioEngine* ist eine Audio-Rendering-Engine, die für mobile virtuelle Realität optimiert und Bestandteil der Google VR Schnittstelle ist. Ziel der Engine ist es dem Zuhörer ein realistisches räumliches Hörerlebnis (engl. „Spatial Audio“) zu bieten, indem man repliziert, wie Schallwellen mit der Umgebung und dem Kopf und den Ohren des Zuhörers in Wechselwirkung treten. Die in diesem Abschnitt diskutierte Thematik kann aus der englischen Dokumentation von Google VR entnommen werden [23].

Durch die räumliche Platzierung von Geräuschen erhält der Benutzer auditive Signale, die ihn dazu verleiten sollen in eine bestimmte Richtung zu blicken. Damit die Illusion der virtuellen Welt nicht durch fehlerhafte Wahrnehmung zerbricht, verwendet die Audio-Engine drei wesentliche Techniken zur Simulierung des räumlichen Hörerlebnisses.

Interaurale Zeitunterschiede Durch eine zeitversetzte Wiedergabe der am Ohr eintreffenden Schallwellen kann der Benutzer erkennen, aus welcher Richtung diese im

Verhältnis zur Position des Kopfes kommen. Die Zeitspanne hängt stark davon ab, wie weit die Schallquelle von der linken oder rechten Seite des Kopfes platziert ist.

Interaurale Volumenunterschiede Für Frequenzen über etwa 1,5 kHz werden vom Menschen Volumenunterschiede zwischen den Ohren wahrgenommen, die zur Bestimmung der Richtung der Schallwellen beitragen. Diese Technik wird von Menschen verwendet, wenn die Quelle höherer Frequenzen auf einer Seite des Kopfes liegt. Das Ohr auf der anderen Seite liegt dadurch im akustischen Schatten, der das Erkennen der Ankunftszeit unmöglich macht.

Spektralfilterung Töne aus verschiedenen Richtungen springen in unterschiedlicher Weise auf die Innenseite der Ohrmuschel. Dadurch werden die Klangfrequenzen in Abhängigkeit von der Richtung des Tons modifiziert. Menschen verwenden die Änderung der Frequenz um die Herkunft des Tones ausfindig zu machen.

2.3.1 Realisierung

Abbildung 2.2: Ambisonics-Technologie [23]

Für die Simulation der räumlichen Akustik wird die sogenannte Ambisonics-Technologie verwendet, die den Kopf des Benutzers mit einem virtuellen kugelförmigen Klangkörper umhüllt. Durch die Platzierung mehrerer virtueller Lautsprecher innerhalb dieses

Klangkörpers können Schallwellen aus beliebiger Richtung reproduziert werden (siehe Abbildung 2.2). Die Genauigkeit der synthetisierten Schallwellen hängt dabei von der Anzahl der virtuellen Lautsprecher ab. Durch Head-Tracking werden die virtuellen Lautsprecher so verschoben, dass die Schallwellen vom Benutzer immer von der gleichen Richtung wahrgenommen werden.

Die Reflexion der Schallwellen an verschiedenen Oberflächen wird ebenfalls von der Audio-Engine berücksichtigt, indem diese in drei Komponenten aufgeteilt werden. Die erste Komponente besteht aus der Schallwelle die auf direktem Weg von der Quelle am Ohr des Benutzers eintrifft (siehe Abbildung 2.3). Dabei verändert sich die Lautstärke des Tons in Abhängigkeit von der Distanz zwischen Benutzer und Schallquelle.

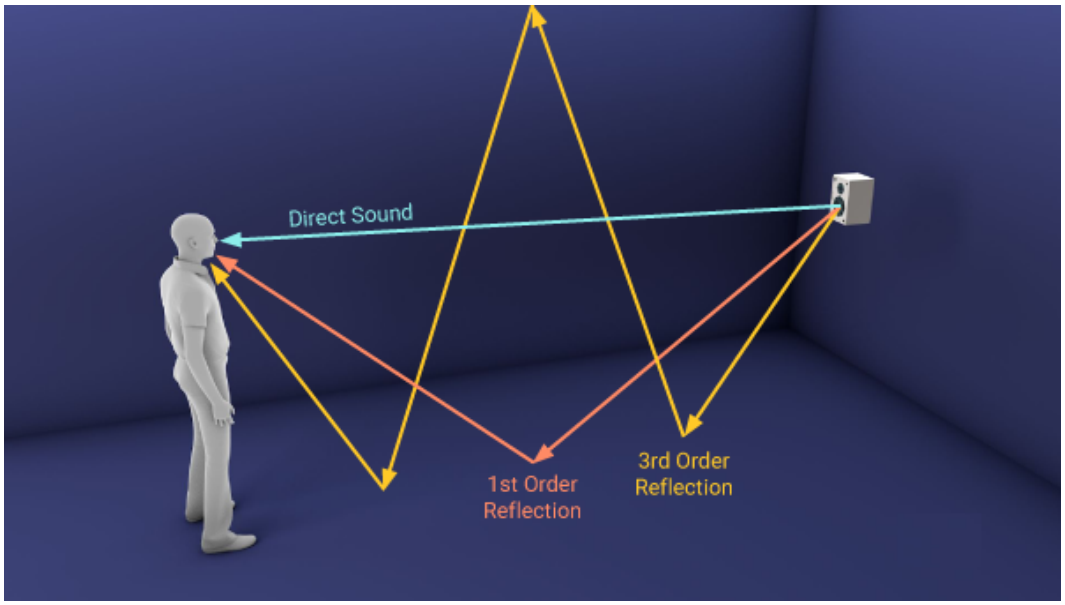

Abbildung 2.3: Reflexion von Schallwellen in der virtuellen Umgebung [23]

Reflektierte Schallwellen, die früh am Ohr des Benutzers eintreffen, bilden die zweite Komponente, die dem Benutzer einen Eindruck über die Größe und Form des Raumes vermitteln. Diese Reflexionen werden in Echtzeit berechnet, indem für jede Reflexion eine neue Schallquelle erstellt wird. Mit Hilfe einer leistungsstarken Hall-Engine können weitere Reflexionen in Echtzeit berechnet werden, die sich immer mehr und mehr verdichten, bis die einzelnen Schallwellen vom Benutzer nicht mehr unterschieden werden können. Diese Schallwellen bilden die dritte Komponente. Verändert sich die Größe

des Raumes oder die Oberflächenmaterialien der Wände, werden die Schallwellen in Echtzeit an die neuen Bedingungen angepasst.

Sobald sich ein virtuelles Objekt zwischen Schallquelle und Benutzer befindet, werden die Frequenzen von der Audio-Engine so verändert, dass hohe Frequenzen mehr als niedrige Frequenzen vom Objekt blockiert werden.

3

Anforderungen

Im folgenden Teil werden die Anforderungen, welchen die Anwendung entsprechen soll, aufgeführt und erläutert. Hierbei wird zwischen *funktionalen* (Was soll umgesetzt werden?) und *nicht-funktionalen* (Wie / unter welchen Rahmenbedingungen bzw. Qualitätskriterien?) Anforderungen unterschieden.

3.1 Funktionale Anforderungen

Die funktionalen Anforderungen an die Anwendung wurden während der Planungsphase so gewählt, dass die Anwendung für die von Tinnitus betroffenen Personen von Nutzen sein kann. Ein besonderer Wert wurde deshalb vor allem auf die Benutzerverwaltung sowie die auditive Stimulation in Form von Richtungshören gelegt, welche nun genauer erläutert werden.

3.1.1 Benutzerkonto

Damit ein Benutzer die Anwendung im vollen Umfang nutzen kann, soll er zunächst ein Benutzerkonto anlegen können, welches lokal auf dem Gerät gespeichert wird. Für die Registrierung werden persönliche Daten wie Vor-/Nachname, Geburtsdatum, Geschlecht, sowie E-Mail Adresse der Person benötigt. Um unbefugten Zugriff auf diese Daten zu verhindern, benötigt jeder Benutzer ein eigenes Passwort, welches nur in kryptischer Form vorliegen soll. Im Falle, dass ein Benutzer sein Passwort vergessen haben sollte, gibt es die Möglichkeit das Passwort zurück zu setzen. Außerdem soll das Passwort in der Anwendung geändert werden können. Jedes Benutzerkonto speichert

den Verlauf aller vergangenen Sitzungen, damit der Benutzer jederzeit auf vergangene Sitzungen zurückgreifen kann.

Die Anwendung soll außerdem die Verwendung mehrerer Benutzerkonten auf einem Gerät verwalten können. Die Benutzerkonten sollen in Form von Json-Dateien persistent auf dem Gerät gespeichert werden die nur vom jeweiligen Benutzer zugänglich sind. Die Datei soll vom Benutzer per Export-/Share-Funktion für andere Anwendungen bzw. Benutzer freigegeben werden können. Eine Import-Funktion soll im Falle von Datenverlust das Einspielen von Backups ermöglichen.

3.1.2 Überblick

Die Anwendung soll jedem Benutzer einen Überblick über seine Aktivitäten bieten. Hierbei soll ein Verlauf aller vergangenen Sitzungen in Form einer Liste für den Benutzer leicht zugänglich sein. Der Verlauf soll vom Benutzer manipulierbar sein, d.h. es können einzelne/mehrere Sitzungen gelöscht, geteilt, sowie exportiert und importiert werden. Zusätzlich soll die Liste nach Datum, Sitzungsdauer, Rundenanzahl, Trefferanzahl, Fehleranzahl und Schwierigkeitsgrad sowohl aufsteigend als auch absteigend sortiert werden können. Außerdem sollen Statistiken, die sich auf alle erfolgreich beendeten Sitzungen beziehen, dem Benutzer einen Überblick über seinen Fortschritt geben.

3.1.3 Anpassungsmöglichkeiten

Bevor ein Benutzer seine Fähigkeiten im Richtungshören unter Beweis stellen kann, soll eine Anleitung das weitere Vorgehen genauer erläutern. Hierbei sollen Einstellungen vorgenommen werden können, die Einfluss auf die nächste Sitzung haben (siehe Tabelle 3.1). Neben der Rundenanzahl können Toleranzbereich sowie Schwierigkeitsgrad vom Benutzer gewählt werden. Für Abwechslung soll eine Auswahl verschiedener Geräusche sorgen und außerdem soll die Lautstärke anpassbar sein. Für eine bessere Orientierung soll die Bildschirmwiedergabe eines virtuellen Raums, dessen Bodenfarbe ebenfalls vom Benutzer wählbar ist, sorgen.

Die Einstellungen sollen persistent gespeichert werden, um eine spätere Wiederverwendung zu ermöglichen. Der Benutzer soll außerdem die Möglichkeit haben, die gespeicherten Einstellungen erneut anzupassen.

Einstellung	**Beschreibung**
Rundenanzahl	Die Anzahl der Runden, die in einer Sitzung durchlaufen werden
Toleranz	Der Toleranzbereich (in Grad) in dem ein Treffer gültig ist
Schwierigkeitsgrad	Der Schwierigkeitsgrad der das Richtungshören erleichtern bzw. erschweren soll
Geräusche-Thema	Eine Kollektion aus zusammengehörigen Geräuschen
Lautstärke	Die Lautstärke der Geräusche
Virtueller Raum	Es wird ein virtueller Raum auf dem Bildschirm gezeichnet, damit der Benutzer sich besser orientieren kann
Bodenfarbe	Die Farbe des Bodens im virtuellen Raum

Tabelle 3.1: Einstellungsmöglichkeiten für das Richtungshören

3.1.4 Schwierigkeitsgrade

Das Richtungshören in Soda soll in drei Schwierigkeitsgraden erfolgen (siehe Tabelle 3.2). Damit wird je nach Schwierigkeitsgrad das Richtungshören für den Benutzer erleichtert bzw. erschwert. Der Benutzer kann somit anhand seiner Fähigkeiten einen passenden Schwierigkeitsgrad wählen, um das Richtungshören optimal zu trainieren. Der leichte Schwierigkeitsgrad soll Einsteiger motivieren sich mehr mit der Anwendung zu befassen. Dabei soll vom Benutzer nur ein Geräusch wahrgenommen werden, dessen Richtung der Benutzer finden soll. Für erfahrene Benutzer sorgt der Schwierigkeitsgrad „normal“ mit der zusätzlichen Wiedergabe eines Hintergrundgeräusches für Ablenkung.

Experten sollen sich am schweren Schwierigkeitsgrad mit der zusätzlichen Wiedergabe von zwei Hintergrundgeräuschen messen können.

Schwierigkeitsgrad	**Beschreibung**
Leicht	Es wird ein Ton abgespielt, dessen Richtung der Benutzer finden soll.
Normal	Es werden zwei Töne gleichzeitig abgespielt, wobei der Benutzer die Richtung eines Tones finden soll
Schwer	Es werden drei Töne gleichzeitig abgespielt, wobei der Benutzer die Richtung eines Tones finden soll

Tabelle 3.2: Schwierigkeitsgrade für das Richtungshören

3.1.5 Richtungshören

Die Hauptfunktionalität der Anwendung ist die auditive Stimulation in der gleichzeitig das Richtungshören getestet wird. Hierbei sollen die Einstellungen (siehe Abschnitt 3.1.3) verwendet werden, die der Benutzer im Vorhinein festgelegt hat. Der Benutzer soll mittels Kopfhörern je nach Schwierigkeitsgrad einen oder mehrere Geräusche wahrnehmen. Während der Wiedergabe der Geräusche soll die Anwendung in kurzen Abständen (z.B. alle 100ms) die Drehung des Benutzers in Grad speichern. Somit kann später überprüft werden ob der Benutzer Probleme beim Orten des Geräuschs hatte. Nachdem der Benutzer sich in die Richtung gedreht hat, aus der er das Geräusch wahrnimmt, soll er mit Hilfe des Cardboard-Buttons (physischer Knopf am Cardboard) die Richtung bestätigen können. Sollte kein Cardboard zur Verfügung stehen muss die Bestätigung mit einer Berührung auf dem Bildschirm erfolgen. Danach findet die Auswertung der Runde statt und der Benutzer soll ein Feedback über die Abweichung zur Schallquelle bekommen. Nachdem alle Runden erfolgreich absolviert wurden, wird die auditive Stimulation beendet und der Benutzer hat die Möglichkeit die Statistiken zur beendeten Sitzung einzusehen.

3.1.6 Erinnerungen

Um die Fähigkeiten im Richtungshören besser trainieren zu können, sollten Benutzer die Anwendung in regelmäßigen Abständen verwenden. Deshalb soll die Anwendung die Möglichkeit bieten einen Erinnerungs-Dienst einzurichten, der den Benutzer benachrichtigt, wenn eine weitere Sitzung fällig ist. Dabei kann der Benutzer den Tag und die Uhrzeit wählen, an dem er benachrichtigt werden möchte. Außerdem ist es möglich die Benachrichtigungen täglich und wöchentlich zu wiederholen, wobei die Anzahl der Tage/Wochen frei wählbar ist. Die Benachrichtigung erfolgt sowohl per E-Mail als auch lokal auf dem Gerät und beide Varianten können beliebig aktiviert bzw. deaktiviert werden.

3.2 Nicht-funktionale Anforderungen

Die Anwendung soll verschiedenen Qualitätskriterien entsprechen, die allgemein auch als nicht-funktionale Anforderungen bekannt sind. Dabei handelt es sich um Rahmenbedingungen, die die Anwendung einhalten soll um dem Benutzer ein problemloses und frustfreies Erlebnis bieten zu können.

3.2.1 Zuverlässigkeit

Die Anwendung soll selbst dann noch funktionieren, wenn unvorhergesehene Eingaben oder Fehler in der Software auftreten. Im Falle eines Fehlers sollen alle Benutzerdaten wieder herstellbar sein, welche beim Neustart der Anwendung neu geladen werden. Werden Daten beschädigt, hat der Benutzer die Möglichkeit ein Backup des Benutzerkontos zu importieren.

3.2.2 Look & Feel

Um eine gute Usability zu erreichen soll die grafische Oberfläche der Anwendung sich am Designkonzept von *Material Design* orientieren. Somit soll gewährleistet werden,

dass Benutzer durch eine vertraute grafische Darstellung die Anwendung intuitiv bedienen können. Außerdem sollen Animationen das *Look & Feel* abrunden, sodass die Anwendung für die Benutzer ansprechender wirkt.

3.2.3 Leistung und Effizienz

Durch optimierte Ladevorgänge sollen kurze Ladezeiten erreicht werden, die wiederum eine schnelle Bedienung ermöglichen. Das selbe Kriterium soll auch für alle Speichervorgänge gelten.

3.2.4 Sicherheit

Passwörter sollen ausschließlich in kryptischer Form vorliegen, so dass zu keiner Zeit das Passwort eines Benutzers aus einer Datei ausgelesen werden kann. Somit sollen Benutzerdaten vor unbefugtem Zugriff geschützt werden.

3.2.5 Offline Anwendung

Die Anwendung soll ohne jegliche Verbindung zum Internet funktionieren und somit alle Daten auf dem Gerät speichern. Es soll lediglich für das Senden von E-Mails die Internet-Schnittstelle angesprochen werden.

4

Architektur & Implementierung

Im Folgenden werden die Zusammenhänge zwischen den eingesetzten Hard- und Softwarekomponenten der Anwendung genauer beschrieben. Hierbei werden auch Aspekte der dazugehörigen Implementierung genannt und erläutert. Außerdem werden Designentscheidungen, die während der Implementierung zu treffen wahren, herausgestellt und genauer erklärt. Anhand einer Gegenüberstellung von Mockups und Screenshots der fertigen Anwendung sollen die Designentscheidungen besser verdeutlicht werden. Da die Anwendung spezielle Soft- und Hardwarekomponenten benötigt, werden zum Schluss die Systemanforderungen geschildert.

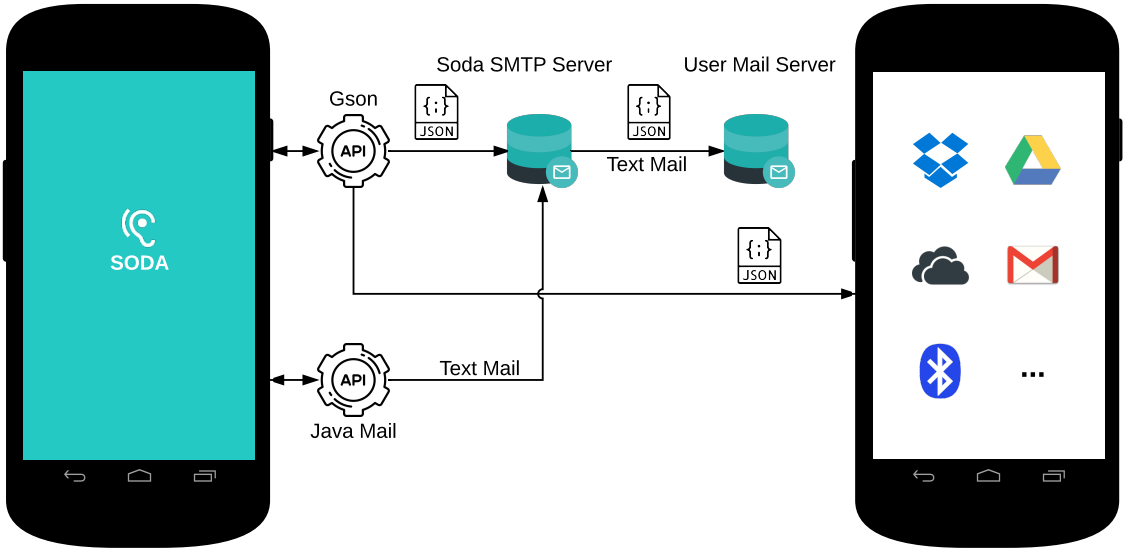

Abbildung 4.1: Software-Architektur der Anwendung

4.1 Konzept

Soda (Sound Origin Direction Application) ist eine mobile medical App, die sich an Personen richtet, deren Gehör durch Tinnitus beeinträchtigt wird. Die App soll es Benutzern ermöglichen ihr Gehör spielerisch zu trainieren, indem sie sich auf ein bestimmtes Geräusch konzentrieren. Je nach Schwierigkeitsgrad sollen neben dem zu ortenden Geräusch zusätzliche Hintergrundgeräusche wahrgenommen werden. Alle Geräusche werden in einem virtuellen Raum zufällig um den Benutzer herum platziert (siehe Abbildung 4.2).

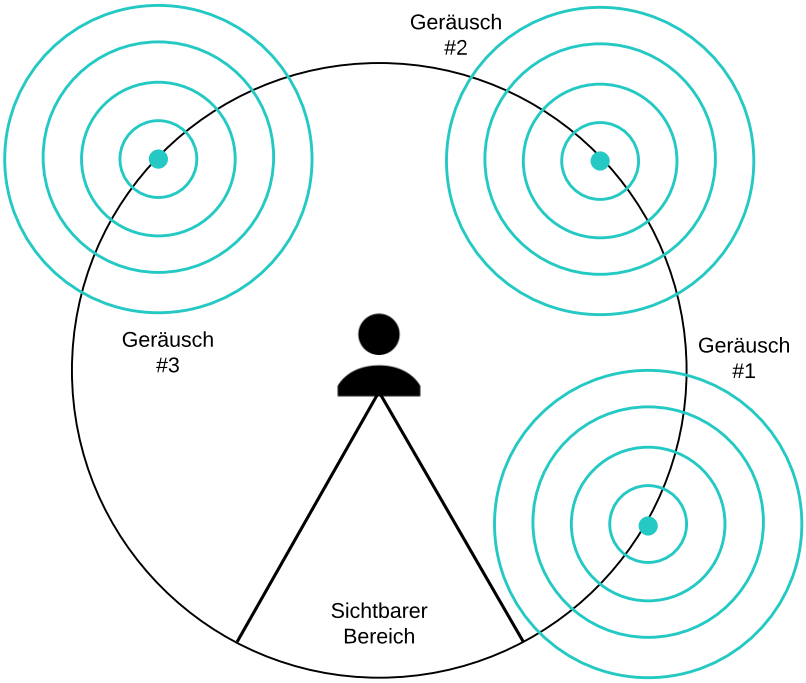

Abbildung 4.2: Konzept der Anwendung

Die Aufgabe des Benutzers besteht darin die Richtung, aus der das Geräusch kommt, durch eine Drehung mit dem Smartphone in der Hand zu lokalisieren. Um dies zu ermöglichen verändert sich das Geräusch immer in Relation zur aktuellen Blickrichtung des Benutzers. Nimmt der Benutzer das Geräusch zum Beispiel auf dem rechten Ohr wahr, muss er sich so weit nach rechts drehen bis das Geräusch auf beiden Ohren

wahrnehmbar ist und somit vor ihm liegt. Falls der Benutzer sich zu weit dreht, nimmt er das Geräusch nur noch auf dem linken Ohr wahr.

Das Richtungshören des Benutzers wird über mehrere Runden getestet, wobei die Rundenanzahl vom Benutzer selbst gewählt werden kann. Mit einer Berührung auf dem Bildschirm oder mit der Verwendung des Cardboard Magnetbutton wird die aktuelle Runde beendet und ausgewertet. Nach jeder Runde wird dem Benutzer die eigentliche Position des Geräuschs durch ein grafisches Objekt auf dem Bildschirm angezeigt. Durch auditives Feedback erfährt der Benutzer außerdem, ob er im zuvor gewählten Toleranzbereich (Abweichung in Grad zum Geräusch) gewesen ist.

Alle erhobenen Daten werden im JSON-Format gespeichert und sind zu jeder Zeit in der Anwendung zugänglich. Anhand dieser Daten können Statistiken erstellt werden, die dem Benutzer einen Überblick über seine Aktivitäten geben.

4.2 Plattform & Werkzeuge

Soda ist eine native Android Anwendung, die speziell für Smartphones entwickelt wurde. Die Anwendung wurde mit Hilfe von *Android Studio 2.2.x* in *Java 1.8* implementiert und getestet. Zur Versionsverwaltung wurde ein *GitHub* gehostetes private Repository verwendet. Für Tests wurde zuerst emulierte Hardware verwendet, später dann reale Hardware, da Sensoren benötigt worden sind.

4.3 Datenmodell & Datenhaltung

Abbildung 4.3 zeigt die Realisierung der Datenbasis der Anwendung. Alle Daten werden auf dem Gerät mit unterschiedlichen Methoden, welche von der Android Plattform bereitgestellt werden, persistent gespeichert. Die Klasse `User` beinhaltet alle Benutzerdaten, sowie einen Verlauf aller vergangenen Sitzungen und bildet somit die zentrale Instanz der Anwendung. Die Klassen `Reminder` und `VrSettings` sind bewusst nicht mit in die User-Klasse integriert, da User–Objekte als Json-Datei exportiert werden können. Somit

soll die Json-Datei nur wesentliche Daten enthalten, die bei einer späteren Auswertung von Nutzen sind. Aufgrund dieser Entscheidung werden Reminder- und VrSettings-Objekte als Json-String in den SharedPreferences, einem Key-Value Store, gespeichert. Die Anwendung ermöglicht es außerdem mehrere Benutzerkonten auf einem Gerät zu erstellen und verwalten. Somit kann die Anzahl der benötigten Geräte für eine potentielle Studie minimiert werden.

Für das Speichern und Laden von Dateien wurde die Klasse `FileUtil` implementiert, die alle Speicher- und Ladevorgänge asynchron abwickelt. Alle Dateien werden im Json-Format gespeichert, um den Ex- und Import von Benutzerkonten und Verläufen als Json-Datei zu vereinfachen. Hierfür wurde die von Google bereitgestellte *Gson-API* in die Anwendung integriert, die eine einfache Konvertierung von Java-Objekten in Json und zurück mit Hilfe der *Java Reflection API* ermöglicht [24].

Die Anwendung unterstützt zusätzlich zum Ex- und Import das Teilen von Benutzerkonten und Verläufen. Hierfür wird die Klasse `FileProvider` benötigt, da Benutzerkonten im *Internen Speicher* gespeichert werden und diese für andere Anwendungen nicht sichtbar bzw. zugänglich sind. Damit externe Anwendungen Zugriff auf diese Daten erhalten wird zuerst mit Hilfe des FileUtils eine temporäre Datei erstellt. Der FileProvider erstellt dann für diese Datei eine `content://Uri`, die dann per Intent an andere Anwendungen weitergeleitet werden kann und diese Zugriff auf die Datei erhalten. Somit können beispielsweise Benutzerkonten schnell und bequem als Json-Datei mit Hilfe von Cloud-Diensten gesichert werden.

Um Fehler und unvorhergesehene Abstürze aufgrund von Dateninkonsistenz vorzubeugen, werden in Soda alle Daten die durch den Benutzer eingegeben werden überprüft. Hierfür wurde die Klasse `InputManager` implementiert, die eine asynchrone Überprüfung der Daten ermöglicht. Sollten Daten inkonsistent sein, wird am dazugehörigen Eingabefeld die passende Fehlermeldung ausgegeben.

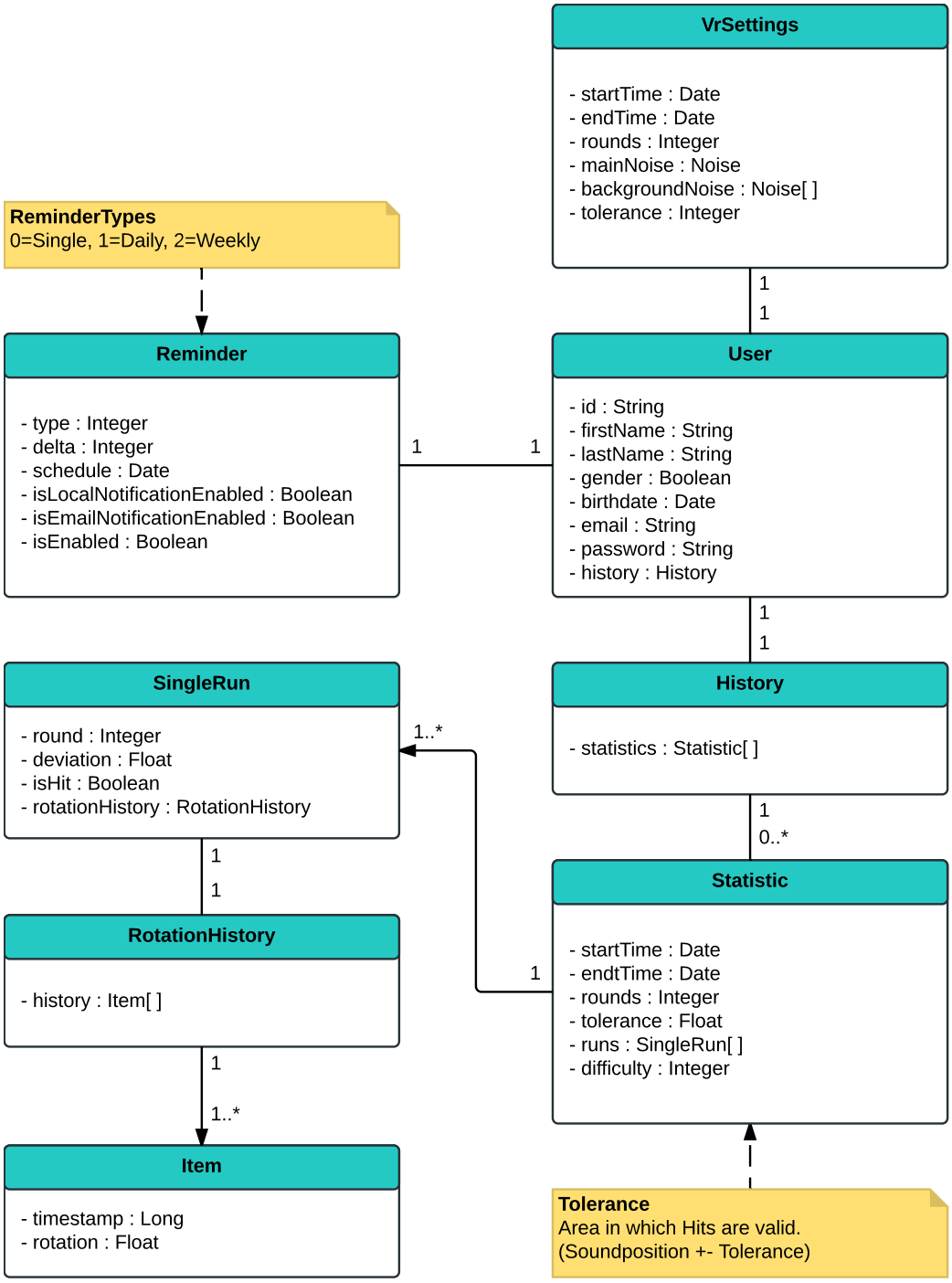

Abbildung 4.3: Datenmodell der Anwendung

4.4 Vorgehensweise

Die in der Planungsphase identifizierten Entities und Ressourcen wurden in Form von Java-Klassen modelliert, welche zur persistenten Datenhaltung dienen (siehe Abbildung 4.3). Dabei wurden einige Klassen während der Implementierungsphase hinzugefügt und angepasst.

Damit die Benutzerdaten des angemeldeten Benutzers zu jeder Zeit abrufbar sind, wurde das *Singleton-Pattern* verwendet, welches sich in vergangenen Projekten durchaus bewährt hat (siehe Quelltext A.3). Die Verwendung eines Singeltons erhöht jedoch auch die Fehleranfälligkeit der Anwendung, da die Instanz des Singletons durch den *Activity Lifecycle* von Android gelöscht werden kann. Um diese Fehler abzufangen wurde die Klasse `UserUtil` implementiert, mit deren Hilfe bei jedem `onResume()` Aufruf die Instanz des Singeltons überprüft wird (siehe Quelltext A.4).

Die Anwendung bedient sich einer Reihe von `Activities`, die alle eine bestimmte Aufgabe erfüllen und ihre eigene grafische Oberfläche (UI) besitzen (siehe Abbildung 4.4). Im Folgenden werden diese Aufgaben genauer beschrieben und wichtige Designentscheidungen, die sich während der Implementierung herausgestellt haben, erläutert.

4.4.1 SplashScreen

Die `SplashActivity` wird beim Start der Anwendung aufgerufen und verwendet das *Launch Screen Pattern* [25]. Sie überprüft ob sich bereits ein Benutzer angemeldet hat, indem im Singelton nach dem hinterlegten User-Objekt gesucht wird. Falls dieses nicht existiert, wird zusätzlich in den SharedPreferences gesucht. Während dieses Vorgangs wird dem Benutzer eine Platzhalter Oberfläche angezeigt. Sobald die Suche einen Treffer erzielt, werden Nutzerdaten geladen und der Benutzer wird zur OverviewActivity weitergeleitet. Schlägt die Suche fehl, wird der Benutzer zur LoginActivity weitergeleitet, in der er sich erneut anmelden kann.

Wird die SplashActivity mit bestimmten Parametern aufgerufen, ist es möglich das User-Objekt des angemeldeten Benutzers sowohl aus dem Singelton als auch aus den

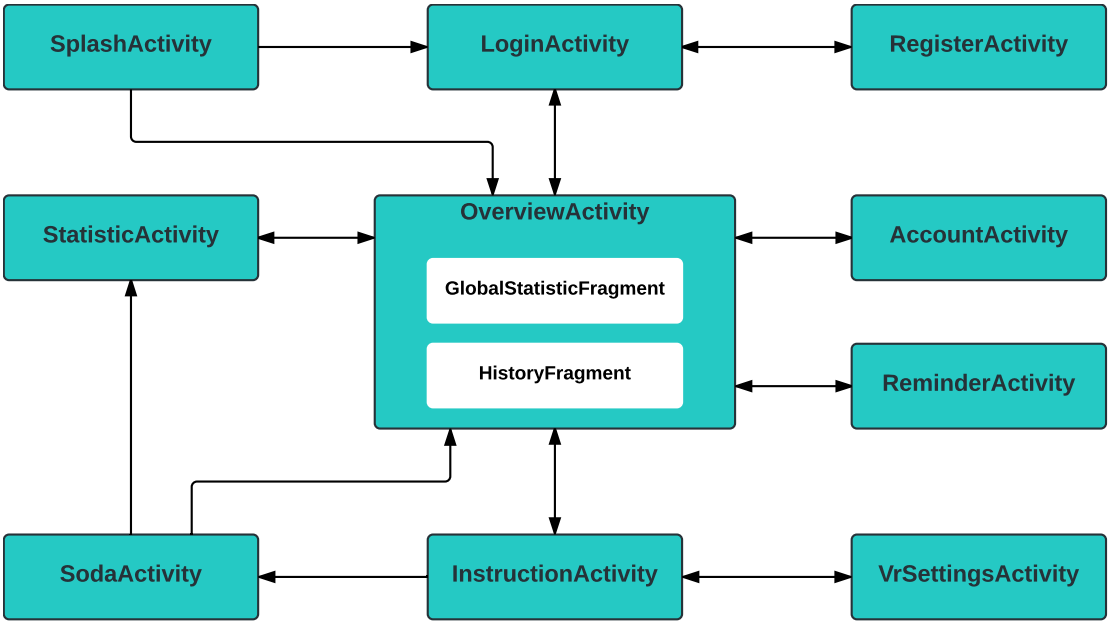

Abbildung 4.4: Übersicht über alle erstellten Activities

SharedPreferences zu löschen. Diese Option wurde bewusst implementiert, damit das UserUtil im Falle eines Fehlers den aktuellen Benutzer abmelden und ihn zurück zur LoginActivity weiterleiten kann.

4.4.2 Login

In der `LoginActivity` wird der Benutzer dazu aufgefordert seine E-Mail Adresse und sein Passwort einzugeben. Nachdem er mit einem Klick auf den „Anmelden“-Button seine Eingabe bestätigt, werden die eingegebenen Daten überprüft. Zuvor werden alle registrierten Benutzer geladen, damit alle benötigten Daten sofort vorhanden sind. Die Anmeldedaten des Benutzers werden dann mit denen der registrierten Benutzer verglichen. Stimmen die Anmeldedaten mit den Daten eines Benutzers überein, wird dieser angemeldet. Besitzt ein Benutzer noch kein Benutzerkonto, hat er die Möglichkeit zur RegisterActivity zu navigieren, in der er ein neues Benutzerkonto anlegen kann.

Des Weiteren bietet die *LoginActivity* die Möglichkeit bereits vorhandene Benutzerkonten zu importieren (siehe Abbildung 4.5). Mit Hilfe einer externen Dateiexplorer-Anwendung,

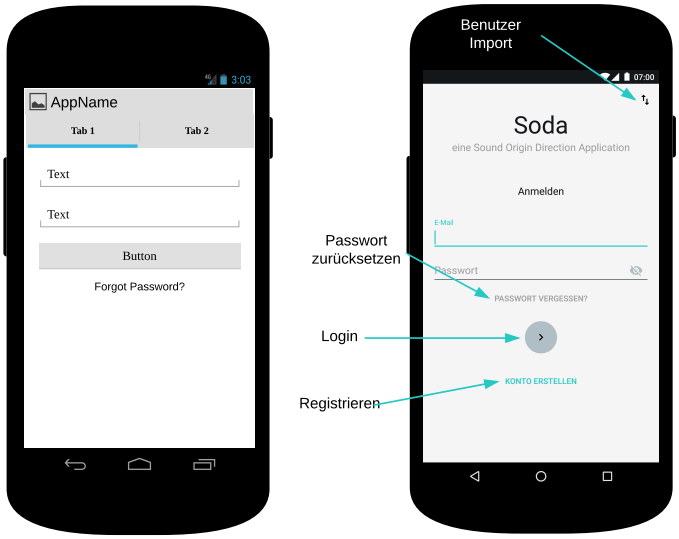

Abbildung 4.5: Mockup vs. Screenshot: LoginActivity

die ebenfalls auf dem Gerät installiert ist, können Json-Dateien ausgewählt werden die beim Import-Vorgang geprüft werden. Durch die Verwendung von Gson können dann die Java-Objekte anhand der Json-Dateien generiert werden. Zur Vermeidung von doppelten bzw. identischen Benutzerkonten werden die Benutzer-ID und die E-Mail Adresse des importierten Benutzers überprüft. Sofern diese nicht von anderen Benutzerkonten verwendet werden, kann ein Benutzerkonto erfolgreich importiert werden.

Besitzt ein Benutzer bereits ein Konto, dessen dazugehöriges Passwort er vergessen hat, ist es möglich das Passwort zurückzusetzen. Zuerst wird der Benutzer dazu aufgefordert die E-Mail Adresse anzugeben, die an das Benutzerkonto verknüpft ist. Dadurch soll verhindert werden, dass Benutzer die Passwörter anderer Benutzer versehentlich ändern können. Ist die E-Mail Adresse gültig, soll der Benutzer den Vorgang bestätigen. Nach der Bestätigung wird das vergessene Passwort durch ein zufällig generiertes 8-stelliges Password ersetzt. Das neue Passwort wird dann per E-Mail mit Hilfe der *Android Mail API* an die Adresse des Benutzers gesendet. Ab diesem Zeitpunkt kann sich der Benutzer mit dem neuen Passwort anmelden [26]. Hierfür wurde ein GMX-Benutzerkonto erstellt, mit dessen SMTP-Zugang die E-Mails versendet werden können.

4.4.3 Registrierung

Die Registrierung sollte eigentlich ein zusätzliches Feature der LoginActivity werden (siehe Abbildung 4.5). Wegen Problemen mit den *Android Runtime Permissions* wurde jedoch die Registrierung in eine separate Activity, die `RegisterActivity`, verlagert. Dadurch wurden alle Probleme beseitigt und die LoginActivity wirkt zusätzlich nicht mehr überladen.

Da Benutzerkonten auch als Json-Datei exportiert werden können, muss das Passwort der Benutzer durch eine kryptografische Funktion unkenntlich gemacht werden. Hierfür wurde die `Bcrypt` Bibliothek für Java verwendet [27]. Neben der Hashfunktion bietet diese Bibliothek auch eine Vergleichsfunktion die in der LoginActivity zum Einsatz kommt.

Bevor Benutzer in der RegisterActivity neue Benutzerkonten erstellen können, müssen sie zuerst alle erforderlichen Daten eingeben. Mit einem Klick auf den „Registrieren“-Button werden zuerst alle eingegebenen Daten mit Hilfe des InputManagers überprüft. Sind die eingegebenen Daten gültig, wird automatisch ein neues Benutzerkonto angelegt. Der Benutzer wird nach diesem Vorgang angemeldet.

4.4.4 Überblick über alle Aktivitäten

Die `OverviewActivity` ist die zentrale Activity der Anwendung, da der Benutzer hier die Möglichkeit hat in alle weiteren Activities zu gelangen (siehe Abbildung 4.6). Außerdem kann sich der Benutzer jederzeit Abmelden indem er die „Zurück“-Taste oder das Logout-Icon in der Toolbar verwendet. Für die OverviewActivity wurde bewusst ein Tab-Layout mit zwei Fragmenten gewählt, damit der Benutzer all seine Daten schnell und einfach abrufen kann. Die beiden Fragmente besitzen jeweils eine spezifische Toolbar mit unterschiedlichen Aktionen.

Das `GlobalStatisticFragment`, welches im ersten Tab zu finden ist, dient als Übersicht über alle vergangenen Sitzungen. Dem Benutzer werden Statistiken, die sich auf die vergangenen Sitzungen beziehen, angezeigt. Damit die Statistiken nicht nur in reiner Textform dargestellt werden, wurde eine externe Diagramm-Bibliothek namens `MPAndroidChart` verwendet [28]. Das Fragment wirkt durch die Darstellung

eines Kuchendiagramm, welches das Treffer/Fehler Verhältnis anzeigt, optisch deutlich ansprechender.

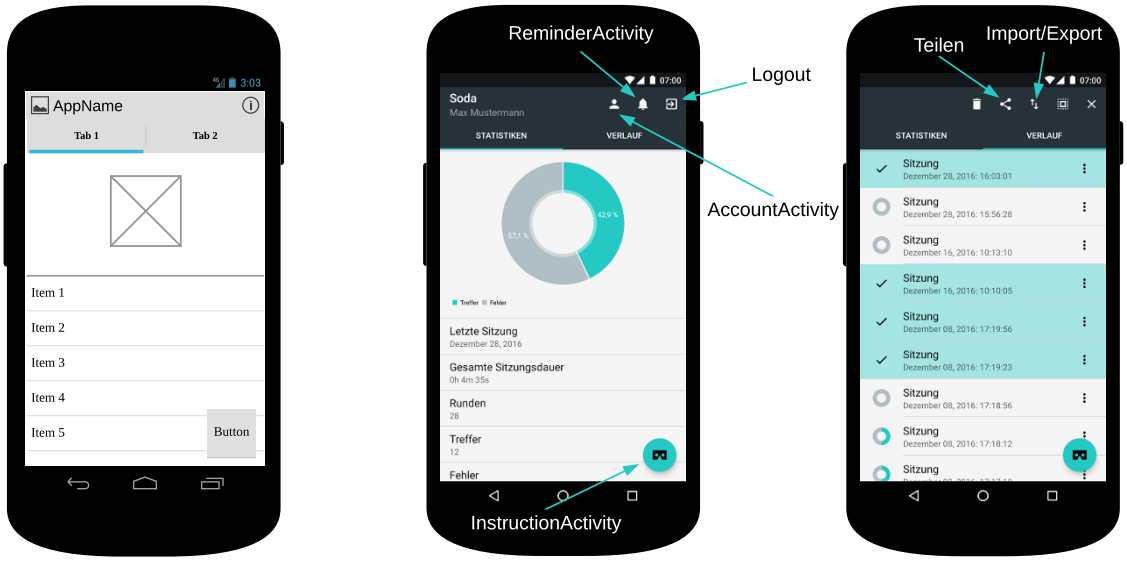

Abbildung 4.6: Mockup vs. Screenshot: OverviewActivity

Der zweite Tab beherbergt das `HistoryFragment`, welches alle vergangenen Sitzungen in einer Liste anzeigt. Hierfür wurde die `RecyclerView` verwendet, da diese schnell und einfach zu implementieren ist und sie nebenbei effizienter als normale ListViews arbeitet. Der Benutzer kann diese Liste nach mehreren Kriterien sortieren. Da oft auch nur ein oder zwei Sitzungen interessant für einen Export sind, bietet die Liste auch die Selektion eines oder mehrerer Objekte. Durch eine Animation und einer anderen Hintergrundfarbe bekommt der Benutzer ein visuelles Feedback über die momentan ausgewählten Objekte (siehe Abbildung 4.6). Der Benutzer kann die ausgewählten Objekte teilen, exportieren und auch löschen. Diese Features werden während der Selektion in der Toolbar für den Benutzer zugänglich.

Da beide Fragmente sich auf das History-Objekt des Benutzers beziehen und diese Daten im HistoryFragment vom Benutzer manipulierbar sind, müssen beide Fragmente sich gegenseitig benachrichtigen sobald Änderungen vorgenommen wurden. Zuerst wurde versucht dieses Problem durch die erneute Instanziierung des nicht angezeigten Fragments zu beheben. Leider führte dieser Lösungsansatz zu immer mehr Problemen

und musste daher verworfen werden. Da beide Fragmente sich auf die History-Instanz des User-Objekts im Singelton beziehen, war schnell ein alternativer Lösungsweg über das *Listener-Konzept* gefunden. Beide Fragmente implementieren einen Listener, der im Singelton registriert wird und die Fragmente benachrichtigt, sobald sich Daten im Singelton ändern. Somit wissen beide Fragmente wann Daten geändert werden und können diese neu laden und verarbeiten.

4.4.5 Statistiken

In der `StatisticActivity` werden, ähnlich wie im GlobalStatisticFragment, Statistiken zu der vom Benutzer ausgewählten Sitzung angezeigt. Durch die Verwendung von MPAndroidChart wirkt die Benutzeroberfläche mit Hilfe von Diagrammen optisch ansprechender für den Benutzer. Hier wird neben dem Kuchendiagramm, welches das Treffer/Fehler Verhältnis abbildet, auch ein Liniendiagramm verwendet. Es wird jeweils nur ein Diagramm angezeigt und der Benutzer kann zwischen beiden Diagrammen wechseln. Die Werte aus der RotationHistory werden mit Hilfe des Liniendiagramms anhand einer Zeitachse grafisch abgebildet. Somit hat der Benutzer einen Überblick über seine Bewegungen während der Sitzung.

Hierbei ist zu beachten, dass bei Drehung über 360 Grad das Diagramm bei dem nächsten Wert einen Sprung zur 0 macht. Dreht sich ein Benutzer während eines Sitzung immer im Bereich zwischen 0 und 360 Grad bildet das Liniendiagramm sehr viele Sprünge zwischen diesen Werten ab. Leider wurde für dieses Problem keine passende Lösung gefunden.

4.4.6 Benutzerprofil

Der Benutzer kann in der `AccountActivity` jederzeit seine persönlichen Daten und sein Passwort ändern. Sollte er zum Beispiel das Passwort in der LoginActivity zurücksetzen, kann er in der AccountActivity das zufällig generierte Passwort durch ein anderes Passwort ersetzen. Damit die grafische Oberfläche nicht allzu leer wirkt, wird das Geschlecht des Benutzers sowohl im Eingabefeld als auch im oberen Bereich als

Icon angezeigt (siehe Abbildung 4.7). Nachdem der Benutzer seine persönlichen Daten geändert hat, kann er mit einem Klick auf den „Speichern“-Button die Änderungen speichern. Zuvor werden wieder alle Eingaben durch den InputManager überprüft.

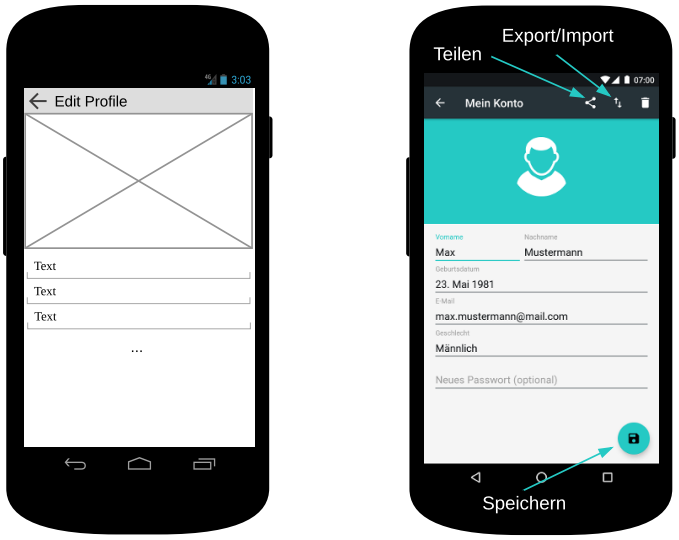

Abbildung 4.7: Mockup vs. Screenshot: AccountActivity

Wird das Benutzerkonto nicht mehr benötigt, besteht die Möglichkeit dieses zu löschen. Damit der Benutzer das Konto nicht aus Versehen löscht, wurde eine optionale Backup-Funktion implementiert, die standardmäßig aktiviert ist. Vor dem Löschen erscheint ein Dialogfeld, in dem der Vorgang bestätigt werden muss und die Backup-Funktion aktiviert bzw. deaktiviert werden kann. Ist das Backup aktiviert, wird vor dem Löschen das Benutzerkonto als Json-Datei exportiert. Der Benutzer wird nach einem erfolgreichen Löschen der Benutzerdatei automatisch abgemeldet und zur LoginActivity geleitet.

Da die Lebensdauer von Smartphones je nach Benutzer sehr gering ausfällt, sollten Benutzerdaten nicht nur auf dem Gerät gespeichert werden. Durch die Implementierung einer Teilen- bzw. Export-Funktion, die das Konto eines Benutzers als Json-Datei speichert, sollen Backups ermöglicht werden. Die Dateien können dann per USB-Verbindung auf den Speicher eines anderen Gerätes übertragen werden. Außerdem können Dateien durch die optionale Teilen-Funktion über Cloud-Dienste oder andere Programme gesichert werden.

4.4.7 Erinnerungen

Erinnerungen sind für jeden Benutzer standardmäßig deaktiviert. In der `ReminderActivity` können diese jedoch aktiviert aber auch wieder deaktiviert werden. Entscheidet der Benutzer sich für eine tägliche bzw. wöchentliche Erinnerung wird der Tag für die nächste Erinnerung automatisch berechnet. Die Erinnerungen erfolgen entweder per lokaler Benachrichtigung auf dem Gerät oder per E-Mail, die beide aktiviert bzw. deaktiviert werden können. Sobald Tag und Uhrzeit in der Zukunft liegen, wird automatisch ein Countdown in der Toolbar angezeigt, damit der Benutzer weiß dass der Erinnerungs-Dienst aktiv ist.

Damit die Anwendung bis zu diesem Zeitpunkt nicht den kompletten Zeitraum im Hintergrund aktiv ist, wird die von Android bereitgestellte Klasse `AlarmManager` verwendet [29]. Sobald der zuvor ausgewählte Zeitpunkt eintrifft, wird vom AlarmManager ein Broadcast gesendet, der vom `ReminderReceiver` abgefangen wird (siehe Quelltext A.5). Dieser sendet mit Hilfe der Java Mail API eine E-Mail an die Adresse des Benutzers, sofern die E-Mail Erinnerung aktiviert ist. Ist die lokale Erinnerung aktiviert, wird eine Notification erstellt, die in der Statusleiste des Geräts angezeigt wird. Sofern tägliche oder wöchentliche Erinnerungen aktiviert sind, wird der Tag für die nächste Erinnerung berechnet und der AlarmManager wird erneut gestartet.

Der AlarmManager machte zu Beginn Probleme, sobald das Gerät neu gestartet wurde. Nach weiterer Recherche stellte sich heraus, dass dieser nur aktiv ist, solange das Gerät im Betrieb ist. Um dies zu umgehen, wurde ein weiterer Receiver, der `BootCompleteReceiver`, implementiert. Sobald das Gerät nach einem Neustart einsatzbereit ist, wird ein Broadcast gesendet der vom BootCompleteReceiver abgefangen wird. Dieser lädt dann mit Hilfe des FileUtils alle registrierten Benutzer und startet den AlarmManager für alle Benutzer, die Erinnerungen aktiviert haben.

4.4.8 Anleitung

Die `InstructionActivity` wird vor jeder Sitzung aufgerufen und bereitet den Benutzer in drei Schritten auf die Sitzung vor. Diese Activity wurde als kleines Tutorial

implementiert, um neuen Benutzern die Handhabung der Anwendung zu erleichtern. Zu Beginn wird automatisch überprüft, ob der Benutzer bereits eine Sitzung abgehalten hat. Das erfolgt über das VrSettings-Objekt, welches im Singelton gespeichert ist und Einstellungen zur Sitzung enthält. Sind diese Einstellungen vorhanden können sie vom Benutzer wiederverwendet und verändert werden (siehe Abbildung 4.8).

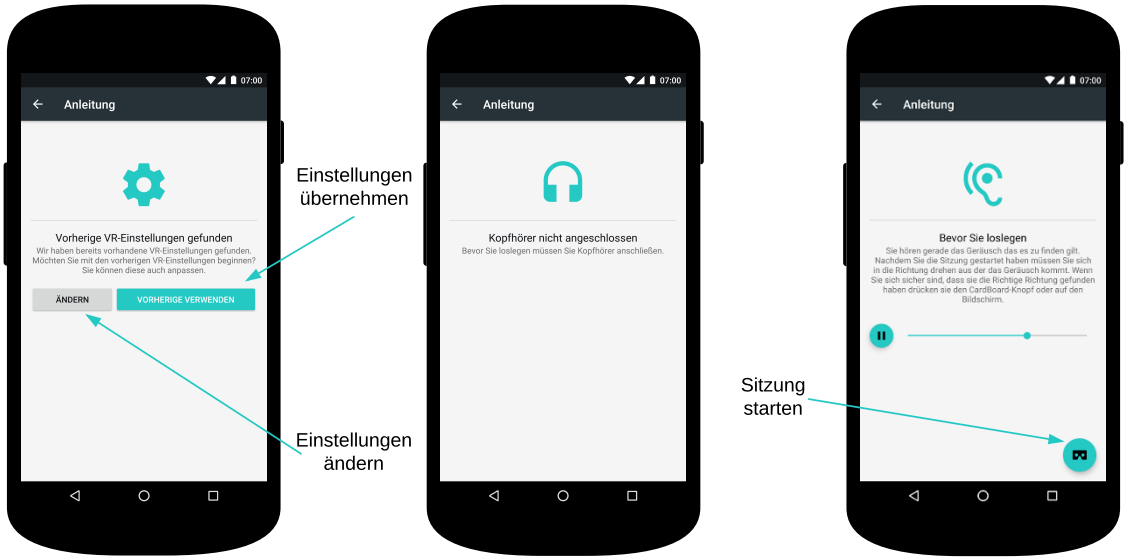

Abbildung 4.8: Screenshot: InstructionActivity

Im zweiten Schritt überprüft die Anwendung ob bereits Kopfhörer angeschlossen wurden. Ist dies nicht der Fall, so wird der Benutzer dazu aufgefordert Kopfhörer anzuschließen. Damit die Anwendung erkennt, ob Kopfhörer angeschlossen sind, wurde der `HeadsetPlugReceiver` implementiert. Dieser fängt den durch das Anschließen gesendete Broadcast ab und benachrichtigt die InstructionActivity mit Hilfe eines Listeners über den Status der Kopfhörer. Dieser Schritt wird automatisch übersprungen, sofern bereits Kopfhörer angeschlossen wurden.

Im letzten Schritt hört der Benutzer das Geräusch, dessen Richtung er später finden soll. Während das Geräusch abgespielt wird, kann der Benutzer die Medien-Lautstärke anpassen. Zusätzlich wird ihm textuell der Ablauf der Sitzung erläutert. Mit einem Klick auf den „Start“-Button, der nur im dritten Teil angezeigt wird, kann die Sitzung gestartet werden.

4.4.9 Einstellungsmöglichkeiten

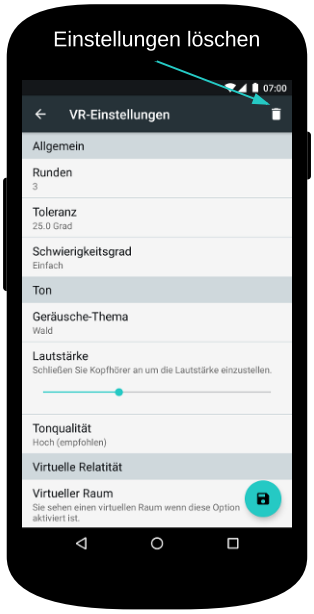

Abbildung 4.9: Screenshot: VrSettingsActivity

Die durch die InstructionActivity aufgerufene `VrSettingsActivity` bietet dem Benutzer die Möglichkeit Einstellungen für die nächste Sitzung anzupassen. Somit kann jede Sitzung vom Benutzer neu konfiguriert werden (siehe Abbildung 4.9). Nachdem alle Einstellungen angepasst wurden, kann der Benutzer mit einem Klick auf den „Speichern“-Button die Änderungen übernehmen. Sind alle Felder ausgefüllt, wird die Activity beendet und der Benutzer wird zurück zur InstructionActivity geleitet.

Damit die InstructionActivity erkennt, dass Einstellungen verändert wurden, wird die VrSettingsActivity über die Methode `startActivityForResult(Intent intent)` aufgerufen. Das setzt voraus, dass die VrSettingsActivity einen Rückgabeparameter beim Beenden der Activity sendet. Dieser wird über einen Callback in der InstructionActivity weiter verarbeitet.

4.4.10 Richtungshören

Sobald ein Benutzer eine neue Sitzung startet, wird er zur `SodaActivity` weitergeleitet, die alle Funktionalitäten für das Richtungshören enthält. Durch die Verwendung der `Google VR SDK` konnten alle benötigten Komponenten und Funktionen realisiert werden [8].

Die für die grafische Oberfläche verwendete `GvrView` dient zur Wiedergabe der von `Open GL ES` gerenderten Vr-Inhalte [30]. Die GvrView kommuniziert automatisch mit den Sensoren des Gerätes und ermöglicht dadurch das Tracking der aktuellen Position im virtuellen Raum. Somit wird ohne zusätzlichen Aufwand der Blickwinkel im virtuellen Raum verändert, sobald das Gerät bewegt wird. Zusätzlich unterstützt die GvrView die Stereo-Wiedergabe von Vr-Content und ist somit mit Google Cardboard kompatibel. Interaktionen mit dem am Cardboard angebrachten Magnetbutton können über die Callback-Methode `onCardboardTrigger()` abgefangen werden. Diese wird bei einem Klick auf den Bildschirm ebenfalls aufgerufen und ermöglicht somit auch die Verwendung der Anwendung ohne Google Cardboard. Die GvrView der SodaActivity ermöglicht lediglich die Wiedergabe eines virtuellen Raums, der in den Einstellungen vom Benutzer verändert aber auch deaktiviert werden kann (siehe Abbildung 4.10).

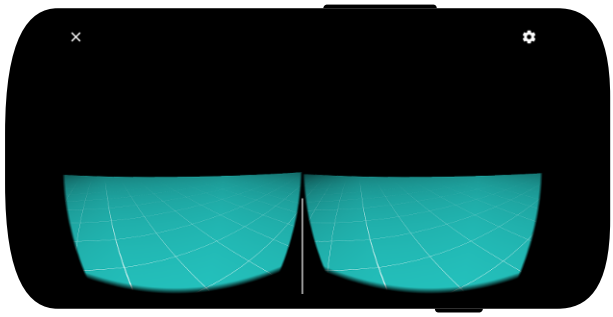

Abbildung 4.10: Screenshot: SodaActivity

Während der Implementierung wurde versucht anstelle des virtuellen Raums ein 360 Grad Panoramabild zu verwenden. Leider war die Implementierung eines Panoramabilds in der GvrView im ersten Ansatz wegen mangelnder Open GL Kenntnissen nicht möglich. Der zweite Lösungsweg führte über die Verwendung der `VrPanoramaView`, die es sehr einfach ermöglicht Panoramabilder als Vr-Content zu verwenden [31]. Dadurch dass

die VrPanoramaView nicht auf Open GL zurückgreift, fehlten jedoch die benötigten Elemente zur Realisierung der räumlichen Akustik. Dadurch wurden beide Lösungsansätze verworfen, was die Wiederverwendung des virtuellen Raums zur Folge hatte.

Das wichtigste Feature der SodaActivity ist die Wiedergabe von Geräuschen, die zufällig im Raum platziert werden. Die aus der Google VR SDK verwendete `GvrAudioEngine` ist eine Audio Rendering Engine, die eine realistische Wiedergabe von Tönen im Raum ermöglicht [23]. Beim Start der Activity werden alle benötigten Audiodateien von der GvrAudioEngine im Hintergrund geladen, welche eine ID zugewiesen bekommen. Hierbei muss beachtet werden, dass die Audiodateien im Mono-Format gespeichert werden, da sonst der Ursprung der Geräusche nicht wahrnehmbar ist. Über die zugewiesene ID werden die Audiodateien beim Start jeder Runde abgespielt und gestoppt, sobald die Runde durch das Tätigen des Magnetbuttons oder mit einem Klick auf den Bildschirm beendet wird. Die Position der Geräusche kann über xyz-Koordinaten beliebig verändert werden.

Während die Geräusche zu hören sind wird im Hintergrund alle 100ms die Drehung des Benutzers bzw. des Geräts in Grad gespeichert. Mit Hilfe eines Threads, der alle 100ms die SodaActivity über einen Listener informiert, wird die Drehung zusammen mit einem Zeitstempel in das RotationHistory-Objekt der aktuellen Runde gespeichert. Somit muss keine separate Log-Datei erstellt werden und alle Daten einer Sitzung sind in einem Statistic-Object vereint.

Mit Hilfe der in der SodaActivity verwendeten `TextToSpeech API` bekommt der Benutzer auditives Feedback über die Geschehnisse und Interaktionen [32]. Die API ermöglicht es geschriebenen Text, der sowohl in Englisch als auch in Deutsch hinterlegt ist, gesprochen auszugeben. Zusätzlich soll der Benutzer beim Beenden jeder Runde visuelles Feedback über die Position des gesuchten Geräuschs bekommen. Landet der Benutzer einen Treffer, wird ihm ein grüner Würfel an der Position des Geräusches angezeigt. Im anderen Fall ist der Würfel rot gefärbt und symbolisiert daher einen Fehler. Sobald alle Runden erfolgreich beendet wurden, wird der Benutzer zur StatisticActivity geleitet, in der er Statistiken zur Sitzung einsehen kann.

4.5 Systemanforderungen

Wie jede Anwendung benötigt Soda auch spezielle Hard- und Softwarevoraussetzungen (siehe Tabelle 4.1). Dadurch, dass die Anwendung eine native Android Anwendung ist, kann diese auch nur auf Android Smartphones verwendet werden. Geräte unter der Android Version 4.4 (KitKat) werden nicht unterstützt, da die Google VR API Geräte ab Version 4.4 oder höher benötigt. Zusätzlich werden Gyroskop, Beschleunigungssensor und ein Magnetometer vorausgesetzt. Für eine flüssige Darstellung wird ein leistungsstarkes Smartphone empfohlen, da die Wiedergabe von VR-Content mit konstanten 60fps (Bilder pro Sekunde) erfolgen sollte. Für schärferes VR-Erlebnis wird eine Displayauflösung von Full-HD Display oder besser empfohlen.

Hardware	
Prozessor	moderne CPU (4 Kerne) / GPU
Arbeitsspeicher	min. 2GB
Speicher Anwendung	15,3MB
Speicher Daten	100MB oder mehr
Sensoren	Gyroskop, Beschleunigungssensor, Magnetometer
Display	min. 1920x1080px (empfohlen)
Schnittstellen	Kopfhöreranschluss (Klinke) oder Bluetooth
Externe Geräte	Kopfhörer mit Klinkenstecker oder Bluetooth
Software	
Betriebssystem	Android Version 4.4+
Grafik	Unterstützung von Open GL ES

Tabelle 4.1: Systemanforderungen der Anwendung

4.6 Installation

Dadurch dass Soda ein Prototyp ist, kann dieser auch nicht im *Google Play Store* gefunden werden. Um trotzdem eine schnelle und einfache Installation der Anwendung

zu garantieren wurde am Ende der Implementierung eine Applikation-Datei erstellt. Somit kann auch die Installation vom Endbenutzer selbst vorgenommen werden.

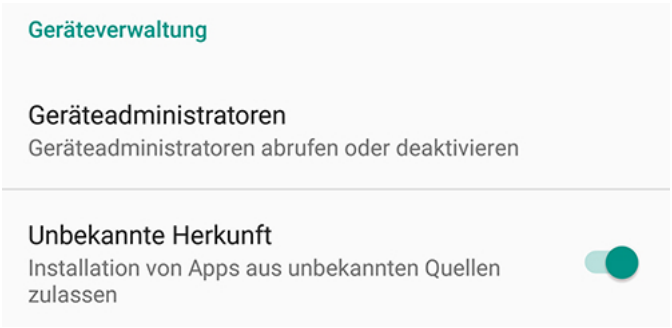

Abbildung 4.11: Erlauben der Installation externer Anwendungen

Bevor die Anwendung auf einem Android Gerät installiert werden kann, muss überprüft werden ob das Gerät den Systemanforderungen entspricht. Andernfalls ist die Funktionalität eventuell eingeschränkt. Des Weiteren muss die Option „Unbekannte Herkunft“ in den Einstellungen aktiviert sein. Diese Option ist auf jedem Gerät standardmäßig deaktiviert und kann unter *Einstellungen* → *Sicherheit* aktiviert werden. Sie ermöglicht es Anwendungen anhand einer Applikation-Datei auf dem Gerät zu installieren, die per USB-Verbindung auf das Gerät kopiert werden können. Mit einer Datei-Explorer Anwendung kann diese Datei dann über den Paket-Manager von Android geöffnet und installiert werden.

5

Anforderungsabgleich

Im folgenden Kapitel wird die Umsetzung der an die Anwendung gestellten Anforderungen bewertet. Für die Bewertung wurde eine Skala von „sehr gut“ (1) bis „ungenügend“ (6) verwendet. Die Wertungen ergeben sich aus der Art und der Qualität der Umsetzung der jeweiligen Anforderungen. Zusätzlich wird erneut auf die Anforderungen eingegangen um die dazugehörige Bewertung zu verdeutlichen.

5.1 Funktionale Anforderungen

Im folgenden Abschnitt wird die Qualität der Umsetzung der funktionalen Anforderungen bewertet (siehe Tabelle 5.1). Die in Abschnitt 3.1 beschriebenen funktionalen Anforderungen wurden alle in der Anwendung umgesetzt und erhielten eine Bewertung zwischen „sehr gut“ (1) und „gut“ (2).

Funktionale Anforderungen	1	2	3	4	5	6
Benutzerkonto	✓					
Überblick	✓					
Anpassungsmöglichkeiten		✓				
Schwierigkeitsgrade	✓					
Richtungshören		✓				
Erinnerungen	✓					

Tabelle 5.1: Anforderungsableich: funktionale Anforderungen

5.1.1 Benutzerkonto

Über die RegisterActivity ist es möglich, neue Benutzerkonten zu erstellen. Hierbei müssen vorerst die in den Anforderungen gestellten persönlichen Daten eingegeben werden. Das Passwort der Benutzer wird mit Hilfe von Bcrypt gehasht und somit unzugänglich für Andere gemacht. Außerdem können Passwörter jederzeit zurückgesetzt oder geändert werden. Die vergangenen Sitzungen eines Benutzers sind in Form eines Verlaufs in das Benutzerkonto integriert und somit jederzeit erreichbar.

Damit nicht jeder Benutzer ein separates Gerät benötigt, ermöglicht die Anwendung das Erstellen und die Verwendung mehrerer Benutzerkonten. Die hinterlegten Benutzerdaten sind in einem nicht zugänglichen Verzeichnis gespeichert und somit nicht zugänglich für den Benutzer. Lediglich mit der integrierten Import- und Exportfunktion können Benutzer ihr Benutzerkonto als Json-Datei sichern und wiederherstellen. In der LoginActivity ist es möglich eine Json-Datei, welche ein Benutzerkonto enthält, zu importieren. Die AccountActivity ermöglicht wiederum das Benutzerkonto als Json-Datei zu exportieren.

Bewertung: *sehr gut*

5.1.2 Überblick

Durch die in der OverviewActivity verwendeten Fragmente, werden Statistiken und Verlauf in einer Activity realisiert. Durch einen einfachen Tab-Wechsel kann der Benutzer entweder Statistiken über alle vergangenen Sitzungen oder den Sitzungsverlauf betrachten. Im GlobalStatisticsFragment werden dem Benutzer folgende Informationen angezeigt:

- Treffer/Fehler Verhältnis in Form eines PieCharts
- Datum der letzten Sitzung
- Sitzungsdauer über alle vergangenen Sitzungen
- Summe der beendeten Runden aller Sitzungen
- Summe der Treffer aller Sitzungen

- Summe der Fehler aller Sitzungen
- Treffer/Fehler Verhältnis als Fließkommazahl

Das HistoryFragment realisiert den Verlauf, der vom Benutzer manipulierbar ist. Durch Selektierung einer oder mehrerer Sitzungen kann der Benutzer die ausgewählten Sitzungen löschen, teilen und exportieren. Außerdem besteht die Möglichkeit einen Verlauf in Form einer Json-Datei zu importieren. Über die Sortierfunktion lässt sich der Verlauf nach den in den Anforderungen gestellten Kriterien sortieren.

Bewertung: *sehr gut*

5.1.3 Anpassungsmöglichkeiten

Die Einstellungen zum Richtungshören werden in den SharedPreferences für jeden Benutzer persistent gespeichert und können daher wiederverwendet werden. Der Benutzer kann folgende Einstellungen vornehmen:

- Rundenanzahl
- Toleranz
- Schwierigkeitsgrad
- Geräusche-Thema
- Lautstärke
- Audioqualität
- Virtueller Raum
- Bodenfarbe für den virtuelle Raum

Bevor eine neue Sitzung gestartet wird, wird die InstructionActivity aufgerufen, in der zunächst die oben genannten Einstellungen angepasst werden können. Nachdem die Einstellungen bestätigt wurden überprüft die Anwendung ob bereits Kopfhörer angeschlossen sind. Nachdem der Benutzer Kopfhörer angeschlossen hat wird das Geräusch

für das Richtungshören abgespielt, dessen Lautstärke ebenfalls anpassbar ist. Gleichzeitig wird dem Benutzer das Richtungshören in textueller Form erklärt.

Bewertung: *gut*

5.1.4 Schwierigkeitsgrade

Die Anwendung stellt für das Richtungshören drei Schwierigkeitsgrade zur Wahl. Die verschiedenen Schwierigkeitsgrade funktionieren exakt so, wie sie in den Anforderungen festgelegt wurden. Somit wird je nach Schwierigkeitsgrad das Richtungshören für den Benutzer erleichtert bzw. erschwert.

Bewertung: *sehr gut*

5.1.5 Richtungshören

Während der Initialiserung der SodaActivity werden die zuvor festgelegten Einstellungen angewandt und alle benötigten Audiodateien geladen. Während des Ladevorgangs sind keine Benutzeraktionen möglich. Nach der Initialisierung wird dem Benutzer per auditivem Feedback (über die TextToSpeech API) mitgeteilt, dass er mit einem Klick die Sitzung starten kann. Der Klick erfolgt entweder mit einem Tipp auf den Bildschirm oder mit dem Cardboard-Button, sofern ein Cardboard vorhanden ist. Nun hört der Benutzer je nach gewähltem Schwierigkeitsgrad ein bzw. mehrere Geräusche, von denen er die Richtung des in der InstructionActivity abgespielten Geräuschs finden muss. Parallel wird alle 100ms die Drehung des Benutzers in Grad gespeichert. Nachdem der Benutzer die Richtung mit einem Klick bestätigt, wird die Runde beendet und der Benutzer bekommt per auditivem Feedback mitgeteilt ob es sich um einen Treffer oder einen Fehler handelt. Außerdem wird er wieder aufgefordert mit einem Klick die nächste Runde zu starten. Nachdem alle Runden beendet wurden, wird die Sitzung zum Verlauf des Benutzers hinzugefügt und gespeichert. Der Benutzer wird dann zur StatisticActivity weitergeleitet, in der er Statistiken zur Sitzung betrachten kann.

Bewertung: *gut*

5.1.6 Erinnerungen

Erinnerungen können mit Hilfe der ReminderActivity aktiviert und konfiguriert werden. Es könne folgende Einstellungen vom Benutzer getätigt werden:

- Tag/Datum und Uhrzeit
- Typ (Einmalig, Täglich, Wöchentlich)
- Art der Benachrichtigung (lokal, E-Mail)

Sofern Tag und Uhrzeit in der Zukunft liegen, wird automatisch mit Hilfe des AlarmManager ein Alarm auf Tag und Uhrzeit gesetzt. In der ReminderActivity wird zusätzlich ein Countdown angezeigt. Sobald der Tag und die Uhrzeit eintreffen werden die gewählten Erinnerungen angezeigt. Ist der Typ auf täglich oder wöchentlich gesetzt wird außerdem automatisch der Tag um den gewählten Zeitraum in die Zukunft verschoben und erneut ein Alarm gesetzt.

Bewertung: *sehr gut*

5.2 Nicht-funktionale Anforderungen

Der folgende Abschnitt befasst sich mit der Bewertung der Umsetzung der nicht-funktionalen Anforderungen (siehe Tabelle 5.2). Alle nicht-funktionalen Anforderungen wurden im Bereich zwischen „sehr gut“ (1) und „gut“ (2) eingestuft.

Nicht-funktionale Anforderungen	1	2	3	4	5	6
Zuverlässigkeit		✓				
Look & Feel	✓					
Leistung und Effizienz		✓				
Sicherheit		✓				
Offline Anwendung	✓					

Tabelle 5.2: Anforderungsableich: nicht-funktionale Anforderungen

5.2.1 Zuverlässigkeit

Die Anwendung ist für ihre kurze Entwicklungsdauer sehr stabil, was bedeutet dass in der Regel keine Abstürze zu erwarten sind. Sollten trotzdem unerwartete Fehler im Zusammenhang mit dem Singelton auftreten stürzt die App trotzdem nicht ab, da diese durch das UserUtil abgefangen werden. Trotz umfangreichem Fehlerhandling ist es nicht unwahrscheinlich, dass bestimmte Fehler auftreten, die die Anwendung zum Abstürzen bringen können.

Sollten im schlimmsten Fall Fehler auftreten, die die Anwendung beschädigen, kann diese erneut installiert werden. Danach sind zwar alle Benutzerdaten nicht mehr vorhanden, diese können aber durch den Json-Import wiederhergestellt werden, sofern Backups erstellt wurden.

Bewertung: *gut*

5.2.2 Look & Feel

Alle UI-Komponenten entsprechen größtenteils den Material Design Richtlinien [33]. Da dieses Designkonzept erst ab Android 5.0 eingeführt wurde, mussten die Support-Libraries von Google verwendet werden um auch Material Design auf Geräte mit Android 4.4 zu bringen [34]. Die verwendeten Icons sind hauptsächlich Material-Design nativ und wurden im Zusammenhang mit den entsprechenden UI-Komponenten verwendet. An bestimmten Stellen wurden Animationen in die UI mit eingebaut, die das Look & Feel abrunden und die Anwendung für den Benutzer ansprechender wirkt.

Bewertung: *sehr gut*

5.2.3 Leistung und Effizienz

Durch asynchrone Speicher- und Ladevorgänge wird der Main-Thread der Anwendung nicht blockiert. Dadurch können während dessen andere Aufgaben erledigt werden, wie zum Beispiel Initialisierungen oder Ähnliches. Trotzdem ist immer noch genügend

Potenzial für Leistungsoptimierung vorhanden, da trotz Nebenläufigkeit die Anwendung an bestimmten Stellen kurz ruckelt. Alles in allem ist die Anwendung effizient und ressourcensparend.

5.2.4 Sicherheit

Passwörter werden ausschließlich mit der Hashfunktion der Bcrypt Bibliothek gehasht und gespeichert. Das kryptische Passwort wird somit unkenntlich gemacht und das richtige Passwort kann nicht reproduziert werden. Außerdem werden die Benutzerkonten, welche als Json-Datei persistent gespeichert werden, im privaten App-Speicher gespeichert, der nur von der Anwendung selber zugänglich ist. Somit können die Daten nicht unbefugt kopiert oder verändert werden.

Bewertung: *gut*

5.2.5 Offline Anwendung

Die Anwendung benötigt keinerlei Internetanbindung während der Laufzeit. Lediglich die Verwendung der Java Mail API benötigt eine Internetverbindung zum Versenden der E-Mails. Der Datenverbrauch hält sich daher sehr in Grenzen, sofern überhaupt die Mail API verwendet wird. Anwendungen, die beim Teilen von Benutzerprofilen oder Verläufen aufgerufenen werden, könnten ebenfalls eine Internetverbindung benötigen.

Bewertung: *sehr gut*

6

Studie

Das folgende Kapitel beschreibt den Aufbau, die Durchführung sowie die Ergebnisse und Erkenntnisse der begleitenden Studie. Ziel dieser Studie ist es, einen Vergleich zwischen der Anwendung und den bereits existierenden Anwendungen zu erlangen.

6.1 Aufbau

Der für die Studie verwendete Fragebogen wurde aufgrund des geringen Umfangs als Paper-Pencil Variante umgesetzt [35]. Dieser baut auf dem Fragebogen der vorherigen Studien auf und muss von jedem Studienteilnehmer ausgefüllt werden [4, 5, 6, 7]. Durch die Unterzeichnung einer Einverständniserklärung bestätigt jeder Teilnehmer die anonyme Nutzung seiner erhobenen Daten (siehe Anhang B). Neben soziodemographischen Daten, wie Alter und Geschlecht werden die Teilnehmer auch nach vergangenen oder vorhandenen Gehörerkrankungen gefragt. Ebenfalls wird die Erfahrung mit mobilen Geräten und das vom Teilnehmer verwendete Smartphone erfasst. Eine durch dieser Studie hinzugekommene Frage befasst sich mit dem Typ der Kopfhörer, die später zusammen mit der Anwendung verwendet werden. Zum Schluss wird der Teilnehmer nach der Erfahrung mit Videospielen gefragt.

Alle von der Anwendung erhobenen Daten sind im Benutzerkonto der jeweiligen Teilnehmer gespeichert. Die automatisch generierte Benutzer-ID wird zur Identifikation der Teilnehmer verwendet. Durch den Export des Benutzerkontos als Json-Datei können alle benötigten Daten später von einem Studienbetreuer ausgelesen werden.

6.2 Ablauf

Jeder Studienteilnehmer füllt zuerst den ausgehändigten Fragebogen aus. Danach erstellt er über die Anwendung ein neues Benutzerkonto. Nachdem sich der Benutzer angemeldet hat, erfolgen insgesamt sechs Sitzungen in zufälliger Reihenfolge, die von einem Studienbetreuer vorgegeben wird. Dies ist erforderlich, da sich die Benutzer erst an das neuartige Bedienkonzept gewöhnen müssen. Die zufällige Reihenfolge verteilt somit den Störfaktor „Eingewöhnungszeit“ auf alle sechs Sitzungen. Folgende Einstellungen werden in den jeweiligen Sitzung vorgenommen:

Sitzung	Schwierigkeitsgrad	Geräusche-Thema
1.	Leicht	Wald
2.	Normal	Wald
3.	Schwer	Wald
4.	Leicht	Büro
5.	Normal	Büro
6.	Schwer	Büro

Jede Sitzung erfolgt in drei Runden mit einer Toleranz von 25 Grad. Die Option „virtueller Raum“ ist ebenfalls in jeder Sitzung mit der Raumfarbe „Soda“ aktiviert.

Nachdem alle sechs Sitzungen erfolgreich beendet wurden, wird das Benutzerkonto als Json-Datei exportiert. Diese beinhaltet insgesamt 18 Datensätze mit den jeweiligen Abweichungen und 6 Datensätze mit der benötigten Zeit. Die Daten werden dann von einem Studienbetreuer ausgelesen und in eine Tabelle eingetragen.

6.3 Ergebnisse & Vergleich

Anhand dieser Studie soll ein Vergleich zwischen der Anwendung und den ähnlichen Anwendungen für die anderen Plattformen ermöglicht werden. Für die Auswertung wurden alle relevanten Daten aus den Benutzerkonten der 20 Studienteilnehmer, die zwischen 18 und 53 Jahren alt sind, herausgefiltert und tabellarisch festgehalten. Unter

den Probanden befanden sich 15 Männer sowie 5 Frauen, darunter 6 Personen mit einer Tinnituserkrankung. Die verwendeten Kopfhörertypen beschränkten sich auf In-Ear und offene Bügelkopfhörer.

Der folgende Abschnitt befasst sich mit den Ergebnissen der Studie, die mit den Ergebnissen der vorherigen Studien verglichen werden.

6.3.1 Sitzungsdauer

Tabelle 6.1 zeigt die durchschnittliche Dauer der jeweiligen Sitzungen, die unter den vorgegebenen Einstellungen abgehalten wurden. In allen Studien erfolgte das Richtungshören pro Sitzung jeweils in drei Runden. Da in dieser Studie nicht die Dauer der einzelnen Runden sondern nur die benötigte Zeit pro Sitzung erfasst wurde, ist somit kein direkter Vergleich möglich.

Dauer (in Sekunden)	**Leicht**	**Normal**	**Schwer**
Wald	111,95	134,90	152,00
Büro	130,90	131,90	148,45

Tabelle 6.1: Durchschnittliche Dauer der jeweiligen sechs Sitzungen

Die Probanden der vorherigen Studien benötigten pro Runde zwischen 9,74 und 15,97 Sekunden [7]. Somit lässt sich trotzdem die Vermutung aufstellen, dass die durchschnittliche Sitzungsdauer in dieser Studie länger ist als in den vorherigen. Mit einer gemittelten Gesamtdauer von 809,7 Sekunden wird diese Vermutung bekräftigt.

Aus Tabelle 6.1 lässt sich außerdem entnehmen, dass die Dauer der Sitzungen im Schnitt bei steigendem Schwierigkeitsgrad länger wird. Mit Hilfe einer Varianzanalyse könnten diese Vermutungen bestätigt beziehungsweise widerlegt werden. Diese Studie beschäftigt sich jedoch lediglich mit dem Vergleich der verschiedenen Plattformen um den Rahmen dieser Arbeit nicht zu sprengen.

6.3.2 Winkelabweichung

In Tabelle 6.2 werden die gemittelten Abweichungen zur tatsächlichen Schallquelle gezeigt. Vergleicht man die Werte mit den Werten aus Tabelle 6.1 erkennt man, dass die Probanden im Schnitt länger für Sitzungen mit dem Geräusche-Thema „Büro“ benötigten, jedoch bessere Ergebnisse erzielten.

Winkel (in Grad)	**Leicht**	**Normal**	**Schwer**
Wald	30,00	34,70	37,29
Büro	23,75	25,81	31,16

Tabelle 6.2: Durchschnittliche Abweichung zur tatsächlichen Schallquelle

Die durchschnittliche Abweichungen der in den vorherigen Studien verwendeten Geräusche liegt zwischen 13,3 und 55,5 Grad [7]. Mit einer gemittelten Abweichung von 33,99 Grad (Wald) und 26,91 Grad (Büro) liegen die Ergebnisse dieser Studie im Mittelfeld.

6.4 Diskussion

Im folgenden Abschnitt werden die aus der begleitenden Studie gewonnen Erkenntnisse erläutert und genauer beschrieben. Außerdem werden einige Schwächen, die bei der Durchführung der Studie aufgefallen sind, beschrieben.

6.4.1 Erkenntnisse

Manche Probanden hatten während dem Richtungshören Probleme mit der Ortung der Schallquelle. Bei einigen war die zu findende Geräuschquelle teilweise nur auf dem rechten Ohr wahrzunehmen und veränderte sich bei einer Drehung nur minimal oder gar nicht. Bei anderen wiederum traten keinerlei Probleme auf, was die Trefferzahlen wiederspiegeln (siehe Abbildung 6.1). Der Grund für das Problem konnte jedoch nicht

ausfindig gemacht werden, jedoch könnte eine Aktualisierung der Google VR API Abhilfe schaffen, da inzwischen eine überarbeitete Version verfügbar ist.

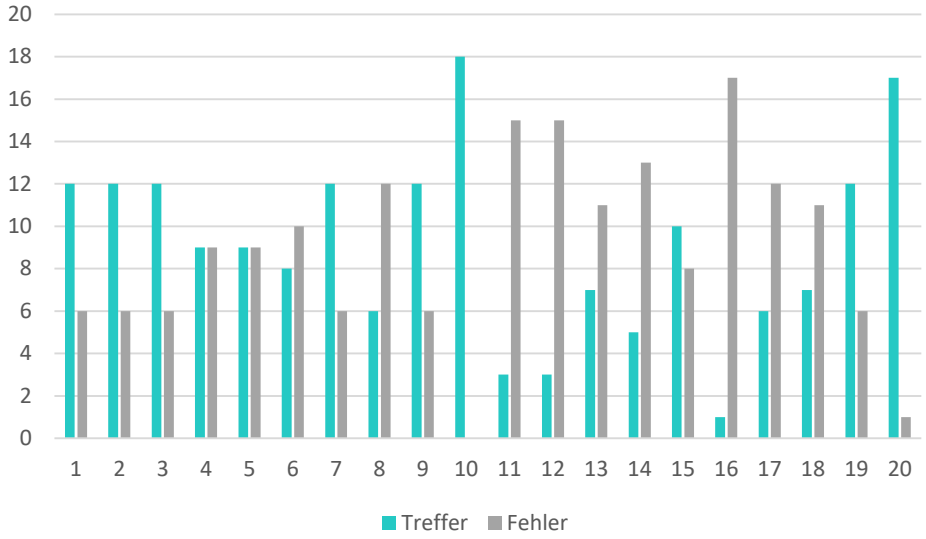

Abbildung 6.1: Anzahl der Treffer und Fehler der einzelnen Studienteilnehmer

Trotz einiger Problemen waren alle Studienteilnehmer sehr auf das Richtungshören konzentriert. Ein paar der von Tinnitus betroffenen Probanden berichteten sogar, dass sie den Tinnitus während der auditiven Stimulation nicht mehr wahrgenommen haben. Andere konnten jedoch keine Verbesserung während den Sitzungen wahrnehmen.

Die Anwendung wurde stichprobenartig auf den Smartphones einiger Probanden installiert. Dabei stellte sich heraus, dass viele Geräte nicht kompatibel waren, da die benötigten Sensoren fehlten. Somit kann die Anwendung nicht auf jedem beliebigen Android Smartphone verwendet werden.

Alles in allem schnitt die Anwendung beim Vergleich nur mittelmäßig ab. Dadurch, dass sich Implementierung sowie Studiendesign teilweise unterscheiden, können sich auch die erhobenen Daten unterscheiden. Neben den eben genannten Problemen sorgten möglicherweise auch weitere Faktoren dazu, dass manche Probanden sehr lange benötigten und teilweise sehr große Abweichungen erzielten. Die Probleme werden im folgenden Abschnitt beschrieben.

6.4.2 Schwachstellen der Studie

Dadurch, dass die Studie ohne Google Cardboard durchgeführt wurde, nahmen die Probanden sowohl den virtuellen Raum als auch die reale Umgebung wahr. Einige berichteten, dass sie teilweise die Schallquelle anhand verschiedener Merkmale im realen Raum orteten. Dies führte dazu, dass sie sich nicht genug auf die Geräusche konzentrierten und dadurch häufig Fehler durch große Abweichungen erzielten.

Außerdem waren viele Studienteilnehmer am Anfang mit dem neuartigen Bedienkonzept überfordert. Zusätzlich wussten einige Probanden nicht welches Geräusch sie orten sollen. Dadurch fallen die ersten Sitzungen der meisten Teilnehmer schlechter aus als die restlichen. Eine präzisere Einführung kann in diesem Fall Abhilfe für dieses Problem schaffen. Durch die zufällige Reihenfolge konnte dieses Problem minimiert, aber nicht eliminiert werden.

Einige Probanden neigten dazu sich schnell und ruckartig zu drehen. Dadurch wird die Positionserkennung unpräzise und eine flüssige Wiedergabe der virtuellen Umgebung auf dem Bildschirm ist nicht mehr möglich. Zusätzlich kann der Studienteilnehmer den Wendepunkt verpassen, bei dem das Geräusch die Kopfhörerseite wechselt. Somit wird das Richtungshören vom Proband selbst deutlich erschwert und erzielt dadurch schlechtere Ergebnisse. Dieses Problem kann ebenfalls durch eine präzisere Einführung umgangen werden.

Des Weiteren wurde die Studie an mehreren Orten durchgeführt, an denen teilweise das Richtungshören durch Umgebungsgeräusche erschwert wurde. Viele Probanden bevorzugten die offenen Bügelkopfhörer, die das Ohr nicht komplett abdecken. Somit nimmt der Hörer neben den Geräuschen aus dem Kopfhörer weitere Geräusche aus der Umgebung wahr. Probanden die lieber die In-Ear-Kopfhörer bevorzugten nahmen diese Störgeräusche nicht wahr.

7

Zusammenfassung

Im Rahmen dieser Bachelorarbeit entstand eine mobile Anwendung für Android Smartphones, die sich auf die Zielgruppe der von Tinnitus betroffenen Menschen spezialisiert. Ziel dieser Anwendung ist es die Tinnitus-Belastung der betroffen Personen spielerisch zu verbessern. Durch gezielte auditive Stimulation soll der Tinnitus für kurze Zeit vom Benutzer nicht mehr wahrgenommen werden. Dies erfolgt in Form eines kleinen Spiels, in dem verschiedene Geräusche um den Benutzer platziert werden, die über Kopfhörer wahrgenommen werden können. Die Aufgabe des Spielers besteht darin, die Richtung einer bestimmten Schallquelle zu lokalisieren, indem er sich in eine beliebige Richtung dreht. Dieses neuartige Bedienkonzept wurde verwendet um die Anwendung auch für Google Cardboard kompatibel zu machen.

Die Anwendung speichert während einer Sitzung verschiedene Daten in einer Json-Datei, die ein späteres Auswerten, beispielsweise in Form einer Studie, ermöglichen. Anhand der erhobenen Daten können zum Beispiel Schlussfolgerungen über die räumliche Hörfähigkeit des Benutzers gezogen werden.

Eine begleitende Studie machte sich genau diese Daten zunutze und ermöglichte einen Vergleich mit anderen Anwendungen, in denen dieses Konzept ebenfalls umgesetzt wurde. Durch die Studie konnten zusätzlich hilfreiche Erkenntnisse gewonnen werden, die zu einer potentiellen Verbesserung der Anwendung führen können.

Im folgenden Abschnitt werden Verbesserungsmöglichkeiten, die eine sinnvolle Erweiterung für die Anwendung darstellen, genauer beschrieben.

7.1 Ausblick

Die aus der Anwendung gewonnen Erkenntnisse bilden den Grundstein für eine mobile Anwendung, die täglich das Leben vieler von Tinnitus betroffener Menschen erleichtern könnte. Mit Hilfe vieler sinnvoller Erweiterungen könnte die Anwendung den von Tinnitus betroffenen Personen besser helfen.

Erweitert man die Anwendung mit der Möglichkeit bestimmte Frequenzbereiche aus den abgespielten Geräuschen herauszufiltern, könnte die Tinnitus-Retraining-Therapie mit dem Tailor-Made Notched Musik Training kombiniert werden. Dadurch könnte möglicherweise die Tinnitus-Belastung der betroffenen Personen weiter verbessert werden. Leider kann dieses Konzept beim subjektiven Tinnitus noch nicht verfolgt werden, da die Töne nicht auf eine physikalische Schallquelle zurückzuführen sind. Beim objektiven Tinnitus hingegen wäre diese Erweiterung optimal, da die Frequenzbereiche gemessen werden können.

Außerdem könnte ein weiterer Ansatz mit Hilfe eines Frequenzgenerators realisiert werden, der es ermöglicht Töne in bestimmten Frequenzbereichen wiederzugeben. Dadurch können die Frequenzen so gewählt werden, dass sie den Tinnitus überdecken und dieser vom Benutzer nicht mehr wahrgenommen werden kann. In Kombination mit dem Richtungshören können dadurch möglicherweise bessere Ergebnisse erzielt werden als mit zufällig gewählten Geräuschen. Die Google VR API unterstützt die Erzeugung von Frequenzen nicht, jedoch kann beispielsweise eine temporäre Audio-Datei erstellt werden, die genau diese Frequenz wiedergibt. Diese Datei kann dann mit der GvrAudioEngine geladen werden, die die Wiedergabe der Frequenzen ermöglicht.

Damit die Anwendung auch einen Anreiz für die tatsächliche Verwendung von Google Cardboard bietet, sollte die Anwendung dem Benutzer während dem Richtungshören eine zum Geräusche-Thema passende virtuelle Welt anzeigen. Die virtuelle Umgebung könnte beispielsweise mit Hilfe eines Panoramabilds realisiert werden, welches dann auf dem Bildschirm ausgegeben wird. Ein weiterer Vorteil von Google Cardboard ist, dass der Benutzer die reale Welt optisch nicht mehr wahrnimmt und sich somit voll und ganz auf die virtuelle Welt und deren dazugehörigen Geräusche konzentrieren kann.

Durch die Bereitstellung eines „Creator Tools“ kann zusätzlich die Langzeitmotivation der Benutzer gefordert werden. Das Tool soll die selbstständige Gestaltung der virtuellen Umgebung mitsamt dem passenden Geräusche-Thema für den Endbenutzer ermöglichen. Die erstellten Welten können dann mit anderen Benutzern geteilt werden, die diese Welt selbst verwenden und bewerten können. Dadurch kann schnell eine Gemeinschaft entstehen, die immer wieder neue Welten zum Vorschein bringt. Die Bereitstellung solcher Funktionalitäten birgt natürlich immer die Gefahr, dass ungewollte Inhalte entstehen, die nicht dem Zweck dieser Anwendung entsprechen. Alternativ könnte ein Download-Portal erstellt werden, in dem vom Entwickler erstellte Welten zum Download bereitgestellt werden.

Zusätzlich könnte ein Cloud-Speicher bereitgestellt werden, der alle Benutzerdaten sowie die erstellten Welten zentral speichert. Somit haben Benutzer von überall einfachen und schnellen Zugriff auf ihre Daten und haben die Möglichkeit im Falle eines Datenverlustes verlorene Daten wiederherzustellen. Dabei muss beachtet werden, dass es sich um medizinische Datensätze handelt und dadurch eine verschlüsselte Verbindung zum Server benötigt wird. Sensible Daten sollten außerdem verschlüsselt werden, damit diese nicht von unbefugten Personen eingesehen werden können.

Durch die Unterstützung weiterer Plattformen wie iOS und WindowsPhone können die von Tinnitus betroffenen Personen auch unterwegs die Anwendung zur Verbesserung der Tinnitus-Belastung verwenden.

Literaturverzeichnis

[1] Kreuzer, P.M., Vielsmeier, V., Langguth, B.: Chronic tinnitus: an interdisciplinary challenge. Dtsch Arztebl Int **110** (2013) 278–84

[2] Initiaive, T.R.: Track your Tinnitus. https://www.trackyourtinnitus.org (2017) Eingesehen am 15.03.2017.

[3] Breithut, J.: In dieser App kommen Zombies durch den Kopfhörer. http://www.spiegel.de/netzwelt/games/audio-defence-zombie-arena-fuer-ios-a-1000419.html (2017) Eingesehen am 15.03.2017.

[4] Blome, J.D.: Implementation and evaluation of a mobile android application for auditory stimulation of chronic tinnitus patients. (2015)

[5] Henkel, F.: Implementation and evaluation of a mobile web application for auditory stimulation of chronic tinnitus patients. (2015)

[6] Mayer, S.: Implementation and evaluation of a mobile ios application for auditory stimulation of chronic tinnitus patients. (2015)

[7] Weidhaas, M.: Implementation and evaluation of a mobile windows-application for auditory stimulation of chronic tinnitus patients. (2015)

[8] Google: Google VR SDK for Android. https://developers.google.com/vr/android/ (2017) Eingesehen am 06.03.2017.

[9] Schobel, J., Schickler, M., Pryss, R., Nienhaus, H., Reichert, M.: Using vital sensors in mobile healthcare business applications: Challenges, examples, lessons learned. In: 9th Int'l Conference on Web Information Systems and Technologies (WEBIST 2013), Special Session on Business Apps. (2013) 509–518

[10] GmbH, T.Z.S.: Alles über Tinnitus. http://www.tinnitus-zentrum.ch/index.php?option=com_content&view=article&id=3&Itemid=9 (2017) Eingesehen am 17.03.2017.

[11] Tinnitracks: Tinnitus-Symptome. http://www.tinnitracks.com/de/tinnitus/symptome (2017) Eingesehen am 17.03.2017.

[12] Goebel, G., Büttner, U.: Grundlagen zu tinnitus: Diagnostik und therapie. psychoneuro **30** (2004) 322–329

[13] Tinnitracks: Objektiver Tinnitus. http://www.tinnitracks.com/de/tinnitus/objektiver-tinnitus (2017) Eingesehen am 17.03.2017.

[14] Tinnitracks: Behandlung von Tinnitus. http://www.tinnitracks.com/de/tinnitus/behandlung (2017) Eingesehen am 17.03.2017.

[15] e.V., D.T.L.: Tailor-made notched music training (TMNMT). http://www.tinnitus-liga.de/pages/presse/pressemitteilungen/archiv/tinnitracks.php (2017) Eingesehen am 17.03.2017.

[16] e.V., D.T.L.: Tinnitus-Retraining-Therapie. http://www.tinnitus-liga.de/pages/tinnitus-sonstige-hoerbeeintraechtigungen/tinnitus/tinnitus-retraining-therapie.php (2017) Eingesehen am 17.03.2017.

[17] In: Virtuelle Realität. Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 5–43

[18] Google: Google Cardboard. https://vr.google.com/cardboard/ (2017) Eingesehen am 15.03.2017.

[19] Staff, P.S.: Google Cardboard Is Virtual Reality On A Budget. http://www.popsci.com/google-cardboard (2017) Eingesehen am 18.03.2017.

[20] Technologies, U.: Unity3D. https://unity3d.com/de (2017) Eingesehen am 19.03.2017.

[21] Games, E.: Unreal Engine. https://www.unrealengine.com/ (2017) Eingesehen am 19.03.2017.

[22] Group, K.: OpenGL ES. https://www.khronos.org/opengles/ (2017) Eingesehen am 19.03.2017.

[23] Google: Google VR SDK - GvrAudioEngine. https://developers.google.com/vr/android/reference/com/google/vr/sdk/audio/GvrAudioEngine (2017) Eingesehen am 06.03.2017.

[24] Google: Gson API - Konvertierung von Java Objekten in JSON. https://github.com/google/gson (2017) Eingesehen am 02.03.2017.

[25] Google: Launch Screen Pattern - Material Design Guidelines. https://material.io/guidelines/patterns/launch-screens.html (2017) Eingesehen am 03.03.2017.

[26] Oracle: Java Mail API. http://www.oracle.com/technetwork/java/javamail/index.html (2017) Eingesehen am 03.03.2017.

[27] Miller, D.: Bcrypt. http://www.mindrot.org/projects/jBCrypt/ (2017) Eingesehen am 05.03.2017.

[28] Jahoda, P.: MPAndroidChart. https://github.com/PhilJay/MPAndroidChartl (2017) Eingesehen am 03.03.2017.

[29] Google: AlarmManager. https://developer.android.com/reference/android/app/AlarmManager.html (2017) Eingesehen am 05.03.2017.

[30] Google: Google VR SDK - GvrView. https://developers.google.com/vr/android/reference/com/google/vr/sdk/base/GvrView (2017) Eingesehen am 06.03.2017.

[31] Google: Google VR SDK - VrPanoramaView. https://developers.google.com/vr/android/reference/com/google/vr/sdk/widgets/pano/VrPanoramaView (2017) Eingesehen am 06.03.2017.

[32] Google: TextToSpeech Engine. https://developer.android.com/reference/android/speech/tts/TextToSpeech.html (2017) Eingesehen am 06.03.2017.

[33] Google: Material Design Guidelines. https://material.io/guidelines/ (2017) Eingesehen am 06.03.2017.

[34] Google: Android Support Library. https://developer.android.com/topic/libraries/support-library/index.html (2017) Eingesehen am 06.03.2017.

[35] Schobel, J., Schickler, M., Pryss, R., Maier, F., Reichert, M.: Towards process-driven mobile data collection applications: Requirements, challenges, lessons learned. In: 10th Int'l Conference on Web Information Systems and Technologies (WEBIST 2014), Special Session on Business Apps. (2014) 371–382

A

Quelltexte

```
{
    "id" : String,
    "firstName" : String,
    "lastName" : String,
    "gender" : Boolean,
    "birthdate" : Date,
    "email" : String,
    "password" : String,
    "history" : {
        "statistics" : [ Statistic ]
    }
}
```

Listing A.1: Benutzerdaten im Json-Format

```
{
    "startTime" : Date,
    "endTime" : Date,
    "rounds" : Integer,
    "tolerance" : Float,
    "difficulty" : Integer,
    "runs" : [{
        "round" : Integer,
        "deviation" : Float,
        "isHit" : Boolean,
        "rotationHistory" : {
            "history" : [{
                "item" : {
                    "timestamp" : Long,
                    "rotation" : Float
```

```
                }
            }]
        }
    }]
}
```

Listing A.2: Statistiken im Json-Format

```
public class Soda extends Application {

    private static Soda instance = new Soda();

    private User loggedInUser;
    private VrSettings vrSettings;
    private Reminder reminder;
    private List<OnSodaChangeListener> mSodaListeners;
    private List<OnReminderChangeListener> mReminderListeners;

    /**
     * Private Constructor.
     */
    private Soda() {
        // Required empty constructor
    }

    /**
     * Returns the Instance of this Class.
     */
    public static synchronized Soda getInstance() {
        return instance;
    }

    ...
}
```

Listing A.3: Ausschnitt aus der Singelton Klasse der Anwendung

```
@Override
protected void onResume() {
    super.onResume();
```

```
4       // Check Singleton Instance
5       UserUtil.logInCurrentUser(mContext, new UserUtil.OnLoginListener() {
6               @Override
7               public void onLoginComplete(User user, int status) {
8                   if (status == UserUtil.STATUS_SUCCESS) {
9                       // No Error, do something
10                  } else {
11                      // Error, logout current User and return to Login Screen
12                  }
13              }
14          });
15  }
```

Listing A.4: Überprüfung des Singeltons mit Hilfe der `UserUtil` Klasse

```
1   public class ReminderReceiver extends WakefulBroadcastReceiver {
2
3       public static final String ACTION_REMIND = "action_remind";
4
5       @override
6       public void onReceive(Context context, Intent intent) {
7           if(intent != null && intent.getAction.equals(ACTION_REMIND); {
8               // Do something
9           }
10      }
11  }
```

Listing A.5: Abfangen eines Broadcast in der `ReminderReceiver` Klasse

B

Fragebogen der begleitenden Studie

ulm university universität
uulm

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

Fragebogen

zur Evaluierung von auditivier Stimulation mit Hilfe einer
Sound Origin Direction Applikation
für die Android Plattform

Jeder Studienteilnehmer füllt zuerst diesen Fragebogen aus. Danach erstellt er über die Anwendung ein neues Benutzerkonto, mit dem dann 6 unterschiedliche Sitzungen in randomisierter Reihenfolge erfolgen. Die genaue Reihenfolge erhält der Teilnehmer über einen Studienbetreuer.

In den 6 Sitzungen werden jeweils folgende Einstellungen übernommen:

1. **Schwierigkeitsgrad**: leicht, **Geräusche-Thema**: Wald
2. **Schwierigkeitsgrad**: normal, **Geräusche-Thema**: Wald
3. **Schwierigkeitsgrad**: schwer, **Geräusche-Thema**: Wald
4. **Schwierigkeitsgrad**: leicht, **Geräusche-Thema**: Büro
5. **Schwierigkeitsgrad**: normal, **Geräusche-Thema**: Büro
6. **Schwierigkeitsgrad**: schwer **Geräusche-Thema**: Büro

Jede Sitzung erfolgt in **3 Runden** mit einer **Toleranz von 25 Grad**. Die Option „virtueller Raum" ist ebenfalls in jeder Sitzung mit der Raumfarbe „Soda" aktiviert.

Nachdem alle Sitzungen erfolgreich beendet wurden, erfolgt ein **Export** des Benutzerkontos, indem der Teilnehmer über „mein Konto" das Konto als Json-Datei auf dem Gerät speichert.

1. Basisdaten

ID: ……..
(wird von einem Studienbetreuer nachträglich eingetragen)

Alter: ……..

Geschlecht: männlich ☐ weiblich ☐

2. Haben Sie Tinnitus oder andere Erkrankungen bezüglich ihres Gehörs?

.....
.....
.....

3. Wie viel Erfahrung haben Sie mit der Verwendung mobiler Geräte?

wenig				viel
☐	☐	☐	☐	☐
1	2	3	4	5

4. Welches Smartphone verwenden Sie üblicherweise ?

.....

5. Welcher Kopfhörertyp wird bei den Sitzungen verwendet?

Ohrhörer: Earbuds ☐ In-Ear ☐

Bügelkopfhörer: offen ☐ geschlossen ☐

6. Wie viel Erfahrung haben Sie mit Videospielen?

wenig				viel
☐	☐	☐	☐	☐
1	2	3	4	5

Ich wurde vom Versuchsleiter über den Ablauf der Studie aufgeklärt und bin damit einverstanden, dass im Rahmen dieser Studie erhobene personengebundene Daten in anonymisierter Form für wissenschaftliche Zwecke verwendet werden.
Mir ist bekannt, dass ich meine Einwilligung jederzeit ohne Angabe von Gründen und ohne nachteilige Folgen für mich zurückziehen und einer Weiterverarbeitung meiner Daten jederzeit widersprechen und ihre Löschung bzw. Vernichtung verlangen kann.

.....
Datum, Ort, Unterschrift

Abbildungsverzeichnis

Tabellenverzeichnis

Name: Lukas Stöferle Matrikelnummer: 829033

Erklärung

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

Ulm, den .....

Lukas Stöferle