

Entwicklung einer mobilen Tutor Anwendung „Teach Me“ für das Android Betriebssystem– Konzeption, prototypische Umsetzung und Anwendungsvergleich.

Masterarbeit

Aleksandr Soloninov

Matrikelnummer: 679XXX

Studiengang: Mobile Computing

Betreuer:

Prof. Dr. Bernd Ruhland

Abstract

In dieser Masterarbeit wird eine native Anwendung für das Android Betriebssystem entwickelt, die als ein Tutor dienen soll. Dazu werden die folgende Forschungsfragen gestellt: Inwiefern lässt sich eine mobile Tutor App als ein Tutor verwenden? Kann eine mobile Tutor App einen realen Tutor oder Tutorin ersetzen? Ist Programmiersprache Java immer noch gut für Android Entwicklung anzuwenden? In der Arbeit wird auch eine Analyse von den aktuellsten App- Arten, Entwicklungssprachen, Integrierte Entwicklungsumgebung und vom Android Betriebssystem durchgeführt. Eine Untersuchung von Tutorien an den deutschen Universitäten wird veranstaltet. Dazu werden Sinn und Zweck von Tutorien, deren Arten und die wichtigsten Aufgaben von Tutoren erklärt. Bei Implementierung der App wird festgestellt, dass Entwicklungssprache Java gut für die Entwicklung ist, aber braucht eine Unterstützung von anderen Sprachen. Um die zwei weiteren Forschungsfragen zu beantworten, werden die Forschungsversuche mit der implementierten App an den ausgewählten Universitäten: Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms mit Hilfe eine Fragebogens realisiert. Es werden insgesamt 131 Studenten befragt. Die Antworten auf den Fragenbogen zeigen, dass die App sich einfach bedienen und sich als ein Tutor ersetzen lässt. Dazu sagen die Ergebnisse auch aus, dass die größere Anzahl an Studenten sich durchaus vorstellen kann, dass die App ein Tutor ersetze. Die Masterarbeit umfasst 88 Seiten, ausgestattet mit 27 Abbildungen, 8 Tabellen, 14 Diagramm und gemacht mit der Beteiligung von 31 Quellen.

Schlüsselwörter: Android, App, Entwicklung, Auswertung, Fragebogen, Tutor, Vergleich.

Danksagung

Einleitung

Erklärung.....	VIII
Abkürzungen.....	IX
Abbildungsverzeichnis.....	X
Tabellenverzeichnis	XII
Diagrammverzeichnis	XIII
Zeitplan.....	XIV
1. Einleitung	1
1.1 Problemstellung	1
1.2 Motivation	2
1.3 Forschungskonzept	2
1.4 Zielsetzung und Erkenntnisinteresse	3
1.5 Forschungsstand	4
1.6 Gliederung der Arbeit.....	8
2. Grundlagen zur Entwicklung einer Android Applications	10
2.1 Mobile Applikationen.....	10
2.1.1 Native App.....	11
2.1.2 Web App.....	12
2.1.3 Hybride App	13
2.2 Entwicklung Sprachen für Android Betriebssystem	14
2.2.1 Java	14
2.2.2 Kotlin	15
2.2.3 Scriptsprachen	15
2.3 Integrierte Entwicklungsumgebung	16
2.3.1 Android Studio.....	17
2.3.2 Basic for Android.....	19
2.3.3 Visual Studio	20
2.3.4 AIDE.....	21
2.4 Android Betriebssystem	23
2.4.1 Anwendungskomponenten.....	24
2.4.2 Pattern.....	25
2.5 Git Versionierung.....	27
2.6 Auswahl einer Technologie	28
3. Tutorien an den deutschen Universitäten	29
3.1 Sinn und Zweck von Tutoren an Universitäten.....	31
3.2 Arten von Tutorien.....	31
3.2.1 Orientierungstutorien	32
3.2.2 Fachtutorien und Übungsgruppen	32
3.3 Aufgaben der Tutoren und Tutorinnen	32

4 Konzeption und Entwurf.....	34
4.1 App Anforderungen.....	34
4.1.1 Funktionale Anforderungen	34
4.1.2 Optionale Anforderungen	35
4.1.3 Nicht-Funktionale Anforderungen.....	36
4.2 Mockups	37
4.3 Prototype.....	38
5. Implementierung	39
5.1 Datenarchitektur	39
5.2 Benutzeroberflächen.....	44
5.2.1 Hauptbildschirm.....	44
5.2.2 Suche.....	46
5.2.3 Quizfragen	47
5.2.4 Interviewfragen	48
5.2.5 Einstellungen	49
5.3 Kapitelfazit	50
6. Anwendungsvergleich	51
6.1 Anforderungen Abgleich	51
6.1.1 Abgleich der funktionalen Anforderungen.....	51
6.1.2 Abgleich der optimalen Anforderungen	53
6.1.3 Abgleich der nicht-funktionalen Anforderungen.....	54
6.2 Fragebogen.....	55
6.3 Forschungsversuch an der Ruprecht-Karls-Universität Heidelberg.....	55
6.4 Forschungsversuch an der Johannes-Gutenberg-Universität Mainz	56
6.5 Forschungsversuch an der Hochschule Worms.....	56
6.6 Fragebogenauswertung	57
6.7 Kapitelfazit	64
7. Zusammenfassung und Ausblick.....	65
7.1 Zusammenfassung.....	65
7.2 Ausblick.....	65
8. Literaturverzeichnis	66
Anhang A. Quellcode.....	67
A1. Quellcode ContentParserAdapter.java	68
A2. Quellcode QuizParserAdapter.java	69
Anhang B. Fragebogen.....	70
B1. Fragebogenmuster	70
B2. Fragebogen aus der Ruprecht-Karls-Universität Heidelberg.....	72
B2.1 Fragebogen von 03.06.2019. Anzahl: 18	72
B2.2 Fragebogen von 07.06.2019. Anzahl: 20	72
B2.3 Fragebogen von 18.06.2019. Anzahl: 4	72

B2.4 Fragebogen von 26.06.2019. Anzahl: 15	72
B3. Fragebogen aus der Johannes-Gutenberg-Universität Mainz.....	72
B3.1 Fragebogen von 03.07.2019. Anzahl: 14	72
B3.2 Fragebogen von 12.07.2019. Anzahl: 9	72
B3.3 Fragebogen von 18.07.2019. Anzahl: 11	72
B3.4 Fragebogen von 23.07.2019. Anzahl: 18	72
B4. Fragebogen aus der Hochschule Worms	72
B4.1 Fragebogen von 06.08.2019. Anzahl: 10	72
B4.2 Fragebogen von 14.08.2019. Anzahl: 12	72

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus anderen Schriften entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit ist noch nicht veröffentlicht oder anderweitig für Prüfungszwecke vorgelegt worden.

Worms, den

Abkürzungen

ADB:	Android Debug Bridge
AIDE:	Android IDE
APK:	Android Package
APP:	Applikation
AVD:	Android Virtual Device
B4A:	Basics for Android
CSS:	Cascading Style Sheets
HTML:	Hypertext Markup Language
IDE:	Integrierte Entwicklungsumgebung
MVP:	Model View Presenter
OS:	Betriebssystem
SDK:	Software Development Kit
VCS:	Version Control System

Abbildungsverzeichnis

Abbildung 1.1: 4 Bildschirme der edX App [4].	5
Abbildung 1.2 Coursera App [6].	6
Abbildung 1.3: Online Kurs RAUM 1 in der Socrative App.	7
Abbildung 1.4: Gliederung der Arbeit	9
Abbildung 2.1: Native- Web und Hybride App [9].....	10
Abbildung 2.2: Android Studio 3.4.1 mit Geöffneter Projekt Teach Me.....	17
Abbildung 2.3: Auswahl von Geräten in ADV Manager.	18
Abbildung 2.4: Interface der IDE Basic for Android.	20
Abbildung 2.5: Interface der IDE Visual Studio	21
Abbildung 2.6: AIDE Interface mit geöffnetem Projekt Teach Me.....	22
Abbildung 3.1: Auswahl einer Übungsgruppe für die Einführung in die Theoretische Informatik in MÜSLI. .	29
Abbildung 3.2: Informatik Vorlesungsverzeichnis für das Winter Semester 19/20 in JOGU-StIne.....	30
Abbildung 3.3: Vorlesungsplan für das 1. Semester der Angewandten Informatik.....	31
Abbildung 3.4: Der Aufbau eins Tutoriums [25].	33
Abbildung 4.1: Mockup der App Teach Me.	37
Abbildung 4.2: Prototype des Hauptbildschirm (rechts) und Quizfragen (links) der App Teach Me.	38
Abbildung 5.1: Teil Code aus CS_GermanContentFile.json	40
Abbildung 5.2: Teil Code aus QA_GERMAN_C++.json.....	41
Abbildung 5.3: Funktion „parseJson“ aus ContentParserAdapter.java.	42
Abbildung 5.4: Funktion „parseJson“ aus QuizParserAdapter.java.....	43

Abbildung 5.5: Hauptbildschirm (links) und geöffnete Modul: Grundlagen der Betriebssysteme (rechts).	44
Abbildung 5.6: Kapitel 4 – Kritischer Abschnitt aus des Moduls Grundlagen der Betriebssysteme.	45
Abbildung 5.7: Begriffssuche in der App am Beispiel Suchwort: shell	46
Abbildung 5.8: Quizfragenbildschirm(links) und Quizergebnisbildschirm(rechts).....	47
Abbildung 5.9: Interviewfragenbildschirm.....	48
Abbildung 5.10: Einstellungen	49
Abbildung 5.11: Favoriten.....	50

Tabellenverzeichnis

Tabelle 1: funktionale Anforderungen von F.1 bis F.4.	34
Tabelle 2: funktionale Anforderungen von F.5 bis F.11.	35
Tabelle 3: optionale Anforderungen.....	35
Tabelle 4: nicht-funktionale Anforderungen.	36
Tabelle 5: Abgleich den funktionalen Anforderungen von F.1 bis F.4.	51
Tabelle 6: Abgleich den funktionalen Anforderungen von F.5 bis F.11.	52
Tabelle 7: Abgleich den optionalen Anforderungen.....	53
Tabelle 8: Abgleich den nicht-funktionale Anforderungen.....	54

Diagrammverzeichnis

Diagramm 1: Nutzung der Geräte bei Anwendungsvergleich.	57
Diagramm 2: Erfahrung mit dem Umgang von Android Smartphones.	58
Diagramm 3: Nutzung von ähnlichen Apps.	58
Diagramm 4: Ergebnisse zu Frage 1 – Programmoberfläche.	59
Diagramm 5: Ergebnisse zu Frage 2 – Inhalt.	59
Diagramm 6: Ergebnisse zu Frage 3 – Programmsuche.	60
Diagramm 7: Ergebnisse zu Frage 4 – Quiz.	60
Diagramm 8: Ergebnisse zu Frage 5 – Fragen und Antworten.	61
Diagramm 9: Ergebnisse zu Frage 6 – Einstellungen.	61
Diagramm 10: Ergebnisse zu Frage 7: Studiengang.	62
Diagramm 11: lässt sich die App einwandfrei benutzen?	62
Diagramm 12: hat die App genug Lernstoff?	63
Diagramm 13: lässt sich die App als Real-Tutor ersetzen?	63

Zeitplan

Dauer: 6 Monate (vom 07.04.2019 bis 07.10.2019)

Bis 08.04.: Literaturrecherche.

Bis 15.04.: Konzeption und Technologie Auswahl.

Bis 31.05.: Implementierung der App.

Bis 24.06.: Forschungsversuch und App Test an der Universität Heidelberg durchzuführen und Feedback sammeln.

Bis 30.07.: Ergebnisse von erstem Test zählen und Rohfassung Hauptteil

Bis 10.07.: App Erweiterung durchzuführen.

Bis 01.08.: Forschungsversuch und App Test an der Universität Mainz durchzuführen und Feedback sammeln.

Bis 07.08.: App Erweiterung durchzuführen.

Bis 15.08.: Forschungsversuch und App Test an der Hochschule Worms durchzuführen und Feedback sammeln.

Bis 25.09.: Auswertungsvergleich und Hauptteil Verfassung.

Bis 30.09.: Vervollständigung Hauptteil und Schlussverfassung.

Bis 01.10.: Überarbeitung und Korrektur.

Bis 02.10.: Layout, Titelblatt und restliche Korrektur.

Bis 04.10.: Druck.

Bis 07.10.: Abgabe.

1. Einleitung

1.1 Problemstellung

Der Mensch will sich immer weiterentwickeln, dazu gehört auch sein Wissensstand.

Vor ca. 30 Jahren konnte man sich nicht vorstellen, dass man sich so viel Wissen aneignen kann, ohne dabei die Haustür zu verlassen. Man musste immer in Bibliotheken, Seminaren und Tutorien gehen.

Eines der Hauptprobleme im Hochschulunterricht ist die geringe Beteiligung der Studierenden, was zu einer geringen Lerneffizienz führt. Im Laufe der Jahre wurden umfangreiche Strategien, Methoden und Lernwerkzeuge entwickelt, um dieses Problem zu lösen. In den letzten Jahren mit der Zunahme der Zahl der Studenten, die über mobile Geräte auf das Internet zugreifen, gab es ein wachsendes Interesse an der Nutzung mobiler Technologien im Lernen, um die Beteiligung der Studenten zu verbessern.

In der Vorlesung wird nur ein Basisstoff vorgetragen und Studenten können nicht immer den Dozenten Fragen stellen, deswegen gibt es seit ca. 10 Jahren an vielen Universitäten extra „Tutorien-Kurse“ für Studenten. Meist wird dort nichts neues gelernt und die Tutorien können auch nicht immer alle Fragen beantworten oder viele Studenten sind schon berufstätig und haben deshalb gar keine Zeit einen Tutor zu besuchen. Dazu kommt noch in einigen Fällen die Sprachbarriere der Studierenden. Internet und Google Suchmaschine bieten vielartige Möglichkeiten, um brauchbare und verwendbare Information zu finden, nur benötigt man dafür viel Zeit. Mobile Endgeräte entwickeln sich und bieten weitere Möglichkeiten mit einer App Zeit und Sprachbarrieren zu reduzieren.

Mit mobilen Geräten können Studenten auf Bildungsressourcen zugreifen, sich mit anderen Benutzern verbinden und Inhalte innerhalb und außerhalb des Lernpublikums erstellen. Mobiles Lernen kann als eigenständige Lerntechnologie fungieren und auch in Verbindung mit anderen Informations- und Kommunikationstechnologien verwendet werden.

1.2 Motivation

Mit der Entwicklung der Technologie müssen traditionelle Lehrmethoden überarbeitet werden. Dieser Trend verursacht eine gemischte Reaktion. Einerseits gibt es Bedenken, dass neue Technologien Lehrer in der Regel ersetzen werden. Andererseits wird die Technologie nur bestehende Probleme beim Lernen lösen.

In den letzten Jahren hat sich gezeigt, dass moderne Geräte als Unterhaltungsgeräte bezeichnet werden. Ihnen stehen vermeintlich ernstere gewohnheitsmäßige Lehrmethoden gegenüber. Tatsächlich sind digitale Geräte längst zur alltäglichen Realität geworden. Darüber hinaus sind sie für die jüngere Generation bekannter und verständlicher als die Lehrmaterialien, die ältere Menschen gewohnt sind. Durch den Einsatz von Tablets und Smartphones sowie Lernspielen wird der Lernprozess sogar visueller.

1.3 Forschungskonzept

Im Rahmen der Masterarbeit werden die folgenden Fragen beantwortet:

- Inwiefern lässt sich eine mobile Tutor App als ein Tutor verwenden?
- Kann eine mobile Tutor App einen realen Tutor oder Tutorin ersetzen?
- Ist Programmiersprache Java immer noch gut für Android Entwicklung anzuwenden?

Um den Fragen nachzuzeichnen, wird die App Teach Me an der Hochschule Worms, Universität Heidelberg und Universität Mainz getestet. Hierbei handelt es sich um eine Live App zu testen und einen Fragebogen mit Feedback auszuführen.

In der Arbeit werden die folgenden Methodik-Schritte durchgeführt:

1. Eine Analyse von Fragenbögen und Feedback wird durchgeführt, um zu untersuchen, wie die App durch das Lernen helfen kann.
2. Wie wurde die App mit Benutzer kommuniziert und wie wurde auf die „Feedback“ von dem Benutzer aufgegriffen?

1.4 Zielsetzung und Erkenntnisinteresse

Um die Forschungsfragen zu beantworten, werden drei Ziele in dieser Masterarbeit gesetzt.

Das erste Ziel ist eine Konzeption und eine Implementierung einer nativen Android App Teach Me für die Informatik Studierenden an vielen Universitäten, die als ein Tutor dienen soll, zu entwickeln. In der App sollte man verschiedene Text oder Video Kurse auswählen und die dazugehörige Information sehen. Mit einem Quiz und Interviewfragen in der App sollte Studierende ihr Lernwissen überprüfen. Die Konzeption, der Entwurf und die Implementierung der App werden in den Kapiteln 4 und 5 beschrieben.

Zweites Ziel ist die Befragung der Informatik- Studenten an der Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms durchzuführen. Die Studierenden sollen die Fragebögen ausfüllen und ein Feedback über die App Teach Me hinterlassen. Die genaue Beschreibung dazu befindet sich im Kapitel 6.2 bis 6.5.

Das letzte Ziel dieser Masterarbeit besteht darin den Fragebogen auszuwerten und zu analysieren. Die Auswertung und die Analyse des Fragebogens sollen helfen die Forschungsfragen zu beantworten. Im Kapitel 6.6 sind die Diagramme dargestellt, die die Fragebogenergebnisse anzeigen.

Folgende Aufgaben werden auch noch an die Arbeit gestellt:

- Eine Analyse der bereits bestehenden App- Arten.
- Eine Analyse von bereits bestehenden Entwicklungssprachen für Android Betriebssysteme durchzuführen.
- Eine Analyse das Android Betriebssystem durchzuführen.
- Eine Analyse der Tutor Tätigkeit durchzuführen.
- Eine Konzeption, Entwurf und Implementierung einer nativen Android App Teach Me.
- Ein Anwendungsvergleich mit Fragebogen von App Teach Me und einem Real-Tutor durch Informatik Studierende an der Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms durchzuführen.
- Eine Auswertung der Fragebögen.

1.5 Forschungsstand

Durch die allgegenwärtige Verbreitung mobiler Geräte interagieren Menschen unterschiedlich mit Inhalten und der Welt. Durch die steigende Produktivität von Smartphones, Smartwatches und Tablets ermöglicht das mobile Lernen den Studenten den Zugriff auf Lernstoff von überall her, häufig von mehreren Geräten.

Mobiles lernen und Gamification sind zwei potenzielle pädagogische Instrumente, die sich in der Hochschulbildung ständig weiterentwickeln. Ihre Wirksamkeit als Lernmittel ist nicht vollständig verstanden, und Ihre Verwendung durch das Personal ist sporadisch und wird manchmal als schlecht im Vergleich zu herkömmlichen Methoden angesehen.

Im Herbst 2011 stand der Stanford-Professor Sebastian Thrun für eine lange Zeit im Mittelpunkt der Aufmerksamkeit der Presse, als sich fast 160.000 Studenten für seinen ersten offenen Online-Kurs über künstliche Intelligenz einschrieben. Der Unterricht fand parallel zu den Vollzeitstudenten von Stanford statt, die im Rahmen ihres Lehrplans denselben Kurs in Thruns Vorlesungen an der Universität studierten [1].

Udacity wurde von Thrun gegründet, um nicht ganze Bildungseinrichtungen, sondern nur einzelne Wissenschaftler zu kooperieren und wählte eigene Kurse im Zusammenhang mit Informatik. Die Vorträge werden in Form von kurzen Video-Vorträgen und kleinen Tests gemacht. Es wurde die Kommunikation der Studenten auf dem Kurs Blog organisiert, wo die Studenten die Möglichkeit hatten, Fragen zu stellen. Aufbauend auf dem Likes-System standen die besten Fragen ganz oben auf der Liste. Die Schüler konnten auch Fragen beantworten. Die nützlichsten und vollständigsten Antworten, basierend auf der Verwendung des fünf-Sterne-Systems, gingen nach oben.

So unterrichteten die Studenten die Studierenden. Nach den Bewertungen der Zuhörer gefiel ihnen die Form der Einreichung des Materials. Die Hand, die Formeln auf dem Bildschirm schrieb, erzeugte den Effekt der Anwesenheit des Lehrers, und die Kommentare und Kontrollaufgaben wurden mit einem guten Anteil an Humor konstruiert [2].

Im Mai 2012 gaben die Harvard University und das Massachusetts Institute of Technology die Gründung einer gemeinnützigen Partnerschaft bekannt, die erstklassige kostenlose Online-Kurse für beide Bildungseinrichtungen anbietet. Die Partnerschaft wurde edX genannt. Das Harvard-Management und das Massachusetts Institute of Technology nutzen die Online-Plattform nicht nur, um eine globale Gemeinschaft von Online-Zuhörern aufzubauen, sondern auch, um die Methoden des Online-Lernens zu verbessern. Die Plattform ermöglicht es auch Vollzeit-Studenten, ihr Wissen zu verbessern [3]. Abbildung 1.1 zeigt die App Oberflächen der edX App.

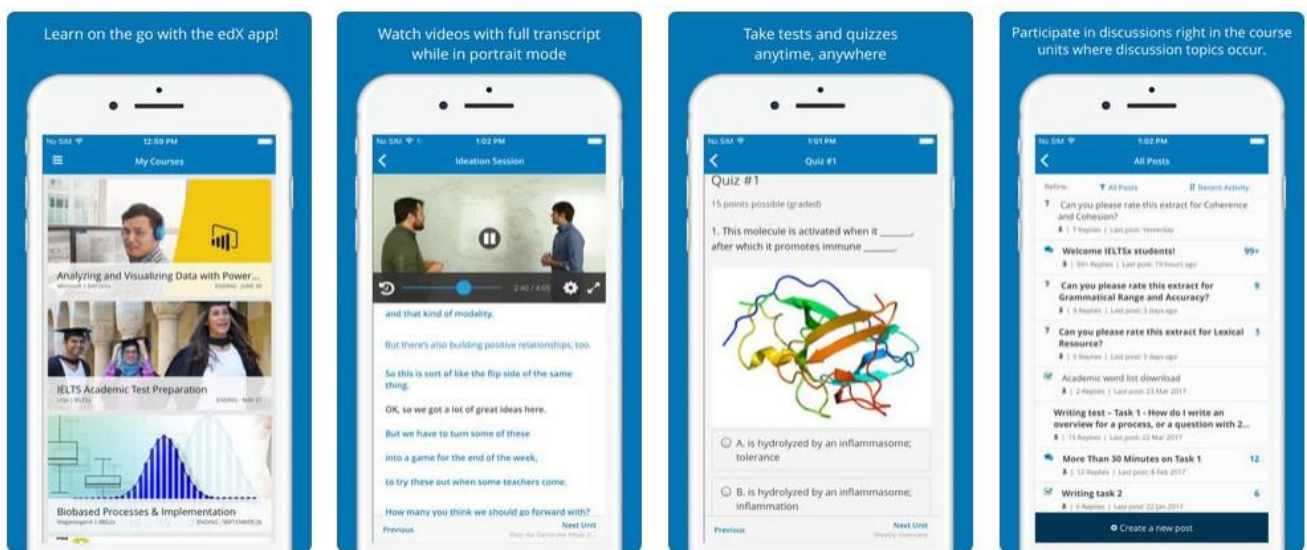


Abbildung 1.1: 4 Bildschirme der edX App [4].

Eine Reihe anderer Universitäten, die dem Beispiel von Udacity und edX folgten, bieten auch Massive Open Online Courses an. So gründeten Stanford, Princeton, Pennsylvania und Michigan Universitäten eine neue Partnerschaft namens Coursera.

Coursera präsentiert mehr als 2000 Kurse. Die Liste der Disziplinen ist breit genug: Ingenieurdisziplinen, Geisteswissenschaften, Kunst, Medizin, Biologie, Programmierung, Recht, Wirtschaft, Wirtschaft und andere. Die Kurse umfassen Video- Vorlesungen, Textaufzeichnungen, Hausaufgaben, Tests und Abschlussprüfungen. Die meisten Vorträge finden auf Englisch statt, aber viele von ihnen werden von Untertitelt [5]. Abbildung 1.2 stellt das Interface von Coursera dar.

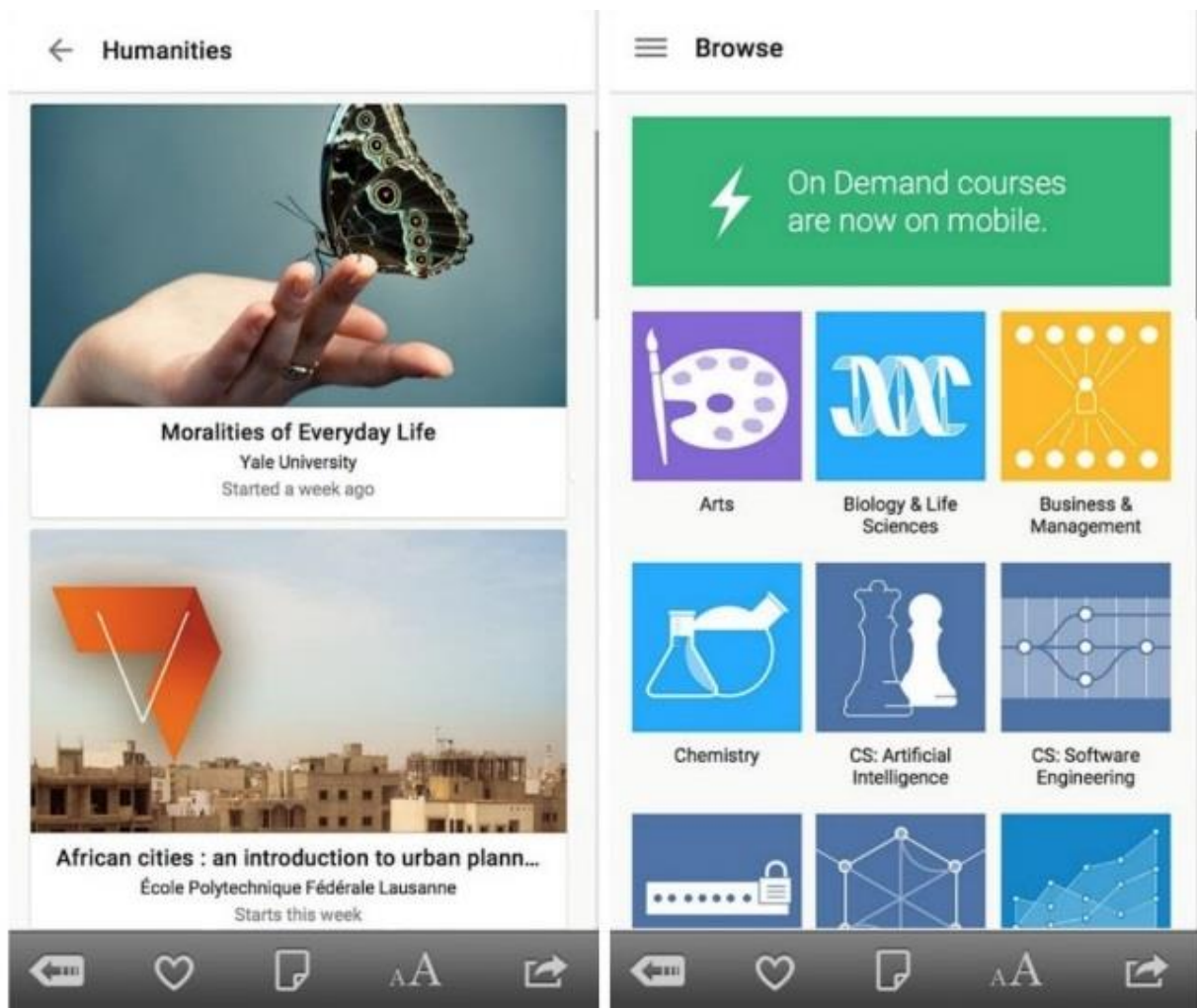


Abbildung 1.2 Coursera App [6].

In den letzten Jahren sind neue Werkzeuge im Kontext von Informations- und Kommunikationstechnologien entstanden, die die aktive Teilnahme, die Interaktion von Lehrer und Studierenden fördern. Eine der beliebtesten Techniken sind Anwendungen, die auf Smartphones, Tablets oder Computern mit iOS -, Android- oder Windows-Betriebssystemen installiert werden können. Eines der besten bewerteten Tools ist die kostenlose Socrative App, die für Pädagogen und Studenten über das Internet oder durch herunterladen auf einem elektronischen mobilen Endgerät leicht zugänglich ist. Die App Interface ist in Abbildung 1.3 dargestellt.

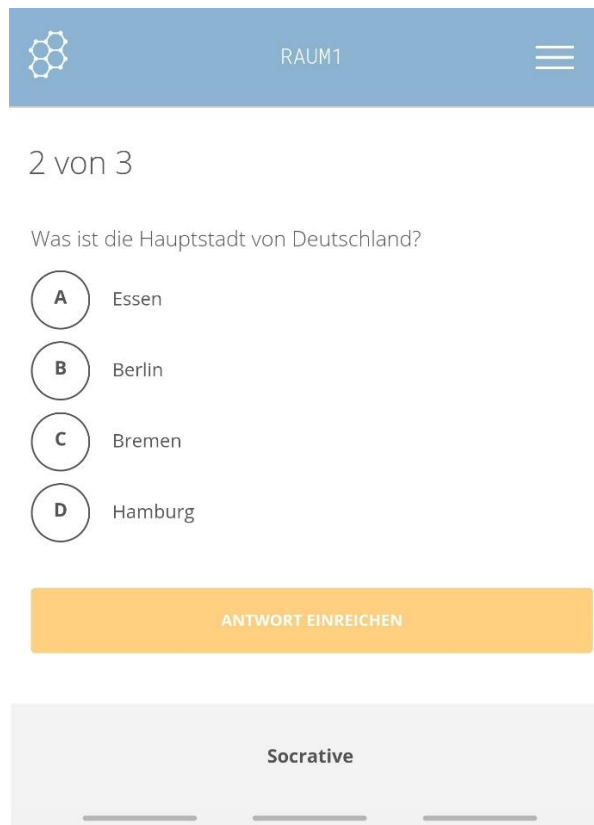


Abbildung 1.3: Online Kurs RAUM 1 in der Socrative App.

An der Sunway University Malaysia wurde eine Studie zur Entwicklung und Implementierung des mobilen Lernens mit einem Socrative Online Response System durchgeführt, um das Engagement der Studierenden zu erhöhen. Insgesamt nahmen 45 Informatikstudenten an dieser experimentellen Studie Teil. Aktivitäten wie Umfragen, Übungen, Quiz und Spiele wurden verwendet, um Diskussionen und Möglichkeiten der Kommunikation zwischen dem Lehrer und den Studierenden zu stimulieren. Sowohl qualitative als auch quantitative Daten wie Bewertungen von Studenten und akademische Ergebnisse wurden analysiert. Die Ergebnisse zeigten, dass die Studenten mit dem mobilen Lernen zufrieden waren. Die Kurse haben das Ausbildungsniveau erfolgreich erhöht und die akademischen Leistungen der Schüler verbessert [7].

Ein Experiment an der Sheffield Hallam University wurde entwickelt, um die Teilnahme von Studenten in der Hochschulbildung für Studenten-Ingenieure zu bewerten. Das Student Response-System basierte darauf, das sofortige Feedback eines Studenten mit kurzen Quizfragen von 10 bis 15 Minuten mit Socrative Software zu erhalten. Die Struktur der Fragen war eine Mischung aus wahren oder falschen, Mehrfachauswahl und kurzen Antwortfragen. Das Experiment wurde durch 2 Semester des einjährigen Ingenieurmoduls durchgeführt. Die Ergebnisse des Experiments wurden auf der

Grundlage der schulischen Leistungen der Studierenden und durch einen Studentenfragebogen analysiert. Die Ergebnisse zeigten, dass 53% der Studenten Ihre Leistung verbesserten, während 23% gleichgeblieben sind. Qualitative Daten zeigten, dass die Studenten eine Verbesserung ihrer Lernerfahrung verspürten [8]

1.6 Gliederung der Arbeit

Das 1. Kapitel beschäftigt sich mit der Problemstellung, Motivation, dem Forschungskonzept, der Zielsetzung und Erkenntnisinteresse, dem Forschungsstand und der Gliederung der Arbeit.

Kapitel 2 stellt die Grundlagen zur Entwicklung einer Android App dar. Zu Erst werden die Aktuellste Arten von Mobilen Applications beschrieben. Anschließend folgt die Beschreibung Entwicklung Sprachen für Android Betriebssystem und dazu passenden Integrierte Entwicklungsumgebung. Dazu wird das Android Betriebssystem und ihre Anwendungskomponenten und Patter erläutert. Am Schluss wird Git Versionierung beschrieben und einer Auswahl einer Technologie für Entwicklung des App Teach Me durchgeführt.

Das Kapitel 3 erzählt über die Bedeutung von Tutorien im Studienprozess. Es werden der Sinn und Zweck sowie auch Arten von Tutorien beschrieben. Zum Schluss werden die Aufgaben der Tutoren und Tutorinnen dargestellt.

Kapitel 4 beschäftigt sich mit Konzept und Entwurf. Zuerst werden die App Anforderungen definiert. Im Folgenden werden die Mockups und App Prototypen nach definierten Anforderungen erschaffen.

Daraufhin wird im Kapitel 5 die App Teach Me implementiert. Es wird die Inhaltdatenstruktur und Quizdatenstruktur entwickelt. Danach kommt die Beschreibung der wichtigsten Oberflächen der App. Zum Schluss kommt das Kapitelfazit.

Das Kapitel 6 beschäftigt sich mit dem Anwendungsvergleich, zuerst werden die Anforderungen der implementieren App abgeglichen. Anschließend folgt die Beschreibung des Fragebogens. Danach werden die Ergebnisse aus den Forschungsversuchen dargestellt und ausgewertet. Zum Schluss kommt das Kapitelfazit.

Abschließend kommt im Kapitel 7 eine Zusammenfassung und ein Ausblick.

Abbildung 1.4 stellt einen kurzen visuellen Überblick über dem Kapitel der Arbeit vor.

Kapitel 1 - Einleitung		
Problemstellung	Motivation	Forschungskonzept
Zielsetzung und Erkenntnisinteresse	Forschungsstand	Gliederung der Arbeit

Kapitel 2 - Grundlagen zur Entwicklung einer Android Applications		
Mobile Applikationen	Entwicklung Sprachen für Android Betriebssystem	Integrierte Entwicklungsumgebung
Android Betriebssystem	Git Versionierung	Auswahl einer Technologie

Kapitel 3 - Tutorien an den deutschen Universitäten		
Sinn und Zweck von Tutoren an Universitäten	Arten von Tutorien	Aufgaben der Tutoren und Tutorinnen

Kapitel 4 - Konzeption und Entwurf		
App Anforderungen	Mockups	Prototyp

Kapitel 5 - Implementierung		
Datenarchitektur	Benutzeroberflächen	Kapitelfazit

Kapitel 6 - Anwendungsvergleich		
Anforderungen Abgleich	Fragenbogen	Forschungsversuch an Ruprecht-Karls-Universität Heidelberg
Forschungsversuch an Johannes-Gutenberg-Universität Mainz	Forschungsversuch an der Hochschule Worms	Fragenbogenauswertung
Kapitelfazit		

Kapite 7 – Zusammenfassung und Ausblick	
Zusammenfassung	Fazit

Abbildung 1.4: Gliederung der Arbeit

2. Grundlagen zur Entwicklung einer Android Applications

Android bietet Entwicklern viele Möglichkeiten: es ist eine universelle, offene Plattform, die von Millionen von Nutzern auf der ganzen Welt verwendet wird. Es gibt viele Tools für Android-Entwickler.

Im Kapitel 2 werden zunächst die Grundlagen vorgestellt, die für den weiteren Verlauf der Arbeit benötigt werden. Darunter fällt die Definition eines App- Arten, die Definition einer Entwicklungssprachen, Entwicklungsumgebungen sowie die Definition die das Android Betriebssystem. Zum Schluss des Kapitels wird die Technologie Auswahl für Entwicklung der App Teach Me durchgeführt.

2.1 Mobile Applikationen

Der Entwickler kann aus drei Optionen für mobile Anwendungen wählen:

- Native App
- WebApp
- Hybride App

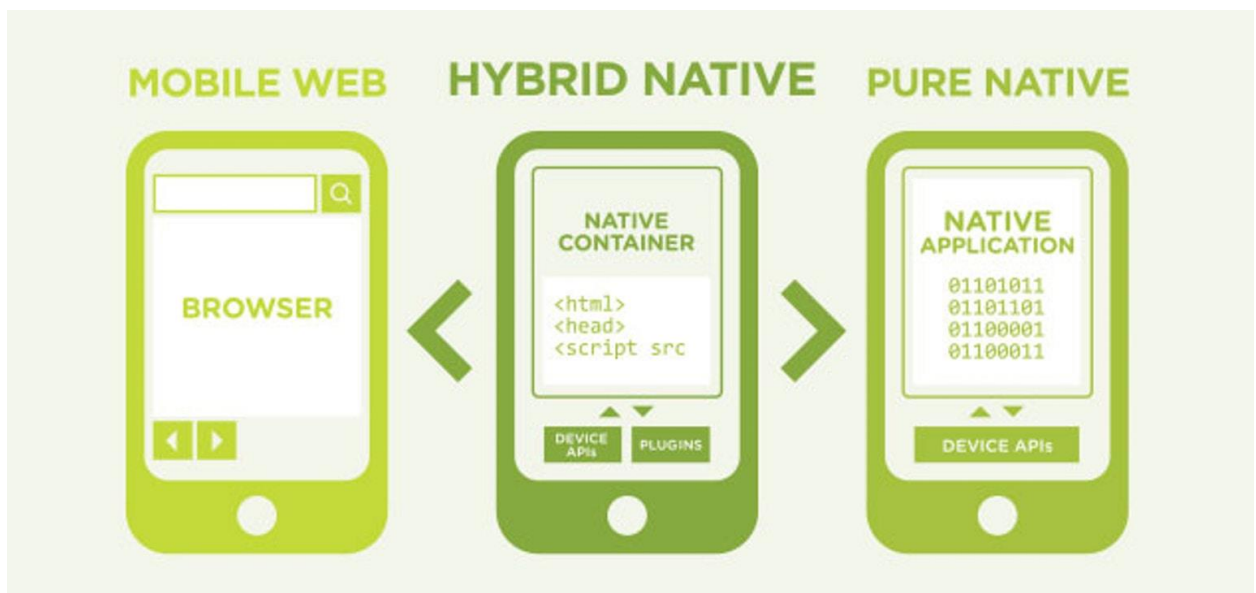


Abbildung 2.1: Native- Web und Hybride App [9].

Die Auswahl eines geeigneten mobilen App-Modells ist ein sehr wichtiger Schritt in der Entwicklung, die von mehreren Faktoren beeinflusst wird, wie Beispielsweise:

- die technische Bewertung der Entwickler
- die Notwendigkeit auf Informationen auf dem Gerät zuzugreifen;
- die Auswirkungen der Internetgeschwindigkeit auf die App;
- eine oder viele Plattform-App

Als Nächstes wird im Kapitel jede Art von Anwendung analysiert, die Vor- und Nachteile untersucht. Es wird bestimmt, was die beste Wahl in der einen oder anderen Situation ist, wenn eine Mobile App erstellt wird.

2.1.1 Native App

Unter Native App wird eine Mobile Anwendung gemeint, die für eine bestimmte Plattform erstellt und direkt auf dem Gerät des Benutzers installiert wird. Solche Apps lädt der Benutzer über den App Store einer Plattform wie dem Google Play Store für Android und dem Apple App Store für iOS herunter.

Mit nativen Anwendungen können Unternehmen die Anwendung nach individuellen Anforderungen herstellen, so dass der Benutzer sie zusätzlich zu der Website oder einem anderen Kanal, den er bereits verwendet hat, bequem nutzen kann. Diese Integrität ist ein wesentlicher Vorteil von nativen Anwendungen [10].

Einige andere wichtige Vorteile von nativen Anwendungen:

- Mit der Geolocation können Unternehmen Ihre Treueprogramme oder Promotionen anpassen. Verbraucher können benachrichtigt werden, wenn er in der Nähe von physischen Geschäften ist.
- Diese Aktivitäten oder Inaktivität des Benutzers können leicht gesammelt und analysiert werden, wodurch die Wirksamkeit der gesamten Anwendung oder Ihrer einzelnen Funktionen bewertet werden kann.
- Native Anwendungen neigen dazu besser zu funktionieren. Web-Anwendungen werden manchmal erstellt, um native zu simulieren, aber sind auf Internetgeschwindigkeit und Designfähigkeiten beschränkt.

Mögliche Nachteile:

- Native Anwendungen sind oft teurer in der Entwicklung, insbesondere für Unternehmen, die Anwendungen auf plattformübergreifenden Betriebssystemen benötigen.
- Native Apps müssen von jedem App Store genehmigt werden, und der Prozess, auf den Benutzer aufmerksam gemacht werden, kann schwierig sein.

2.1.2 Web App

Web App Apps arbeiten über einen Webbrowser auf dem Gerät des Benutzers. Diese Apps sind in den wesentlichen individualisierten Websites, die so gestaltet sind, dass sie wie native Apps aussehen und verwendet werden, aber befinden sich nicht wirklich auf dem Gerät des Benutzers. Mit einer guten, qualitativ hochwertigen Entwicklung, die Größenanpassung und scrollen beinhaltet, funktionieren Webanwendungen oft ähnlich wie native Anwendungen [11].

Vorteile von Webanwendungen:

- Webbasierte Anwendungen werden leichter unterstützt und können auf der Plattform mit jedem Betriebssystem funktionieren.
- Entwickler können Apps anbieten, ohne dass App von App Stores genehmigt werden müssen.
- Schnellere Entwicklung von Schleifen mit CSS, HTML und JavaScript.

Und Nachteile:

- Webanwendungen haben keinen Zugriff auf das Gerät des Benutzers.
- Benutzer müssen Web App über das Internet verwenden, was die Sicherheitskontrolle erheblich reduziert.
- Die Suche nach einer App kann schwierig sein.

2.1.3 Hybride App

Hybridanwendungen sind etwas zwischen nativen und Webanwendungen. Hybride App werden so erstellt, dass die App, wie native Anwendungen aussehen und verwendet werden. Hybride Apps werden auch auf dem Smartphone des Benutzers installiert und können in App Stores gefunden werden. Der Unterschied besteht darin, dass Hybride Apps unbedingt innerhalb einer nativen Anwendung gehostet und erstellt werden müssen, um über WebView zu arbeiten [12].

Vorteile von Hybrid-Anwendungen:

- Hybrid-Anwendungen bieten die beste Funktionalität und Personalisierung für den Benutzer.
- Entwickler sind nicht auf eine einzige Plattform beschränkt, sondern können eine hybride Anwendung erstellen, die mit mehreren Plattformen funktioniert.
- Hybride sind eine gute Option für Entwickler, die visuell gesättigte Anwendungen wie Spiele erstellen.

In jedem Fall gibt es einige Nachteile, die bei der Auswahl einer Hybrid-App in Betracht gezogen werden sollten:

- Zu komplexe Anwendungen sind am besten nativ.
- Die Entwicklung erfordert zusätzliche Zeit und Mühe, damit eine solche Anwendung des Benutzers als nativ aussieht und sich anfühlt.
- App Stores können Hybridanwendungen ablehnen, die nicht reibungslos genug funktionieren.

2.2 Entwicklung Sprachen für Android Betriebssystem

Mit jeder Entwicklung Sprache und jedem Framework sind ihre eigenen Komplexitäten und Nuancen, Vor- und Nachteile verbunden. Weiter in diesem Kapitel werden die wichtigsten Sprachen für das Schreiben von Android-Anwendungen beschrieben und analysieren.

2.2.1 Java

Die offizielle Programmiersprache, die von der Android Studio-Entwicklungsumgebung unterstützt wird. Laut einer jährlichen Umfrage der Stack Overflow ist Java 2018 in die fünf beliebtesten Programmiersprachen eingestiegen.

Java bezieht sich auf die meisten offiziellen Google-Dokumentation, und es ist nicht schwer, bezahlte und kostenlose Bibliotheken und Handbücher zu finden-es gibt viele von Ihnen.

Leider verhindert die Komplexität von Java, dass jeder daran programmiert. Als objektorientierte Programmiersprache hat es eine Reihe von Funktionen in Form von Klassen Konstruktoren, Ausnahmen, die zu fallenden Anwendungen, während der Arbeit und anderen Momenten führen, die bei der Entwicklung immer berücksichtigt werden müssen. Der Code in Java ist jedoch leicht zu lesen und zu strukturieren, insbesondere wenn die akzeptierten Standards für die Gestaltung eingehalten werden [13].

Bei der Entwicklung in Java unter Android werden nicht nur Java-Klassen verwendet, die Code enthalten, sondern auch XML-Manifest Dateien, die dem System grundlegende Informationen über das Programm liefern, und Gradle, Maven oder Ant Build-Systeme, die Befehle in Groovy, POM und XML-Sprachen schreiben. Standardmäßig werden in Projekten Gradle verwendet, und in den Anfangsphasen der Entwicklung in Java müssen die in Groovy geschriebenen Dateien praktisch nicht bearbeitet werden. Für das Layout des UI-Teils wird normalerweise auch die XML-Sprache verwendet [13].

2.2.2 Kotlin

Die Sprache Kotlin wurde offiziell im Mai 2017 auf Google I/O eingeführt und von Google als die zweite offizielle Programmiersprache unter Android nach Java positioniert, nur ein wenig einfacher zu verstehen. Java-Kenntnisse werden hier benötigt, um die Arbeitsprinzipien von Kotlin, die Allgemeine Struktur der Sprache und ihre Besonderheiten zu verstehen. Viele Entwickler betrachten Kotlin als Wrapper über Java und empfehlen, es erst zu lernen, nachdem es sich mit Java-Wissen vertraut gemacht haben [14].

Kotlin ist mit Java kompatibel und verursacht keine Leistungseinbußen und größere Dateigrößen. Der Unterschied zu Java ist, dass es weniger dienstlichen, sogenannten Boilerplate Code benötigt, so dass es stromlinienförmiger und leichter zu lesen ist. Seine Schöpfer haben es geschafft, Nullpointerexception zu vermeiden und die Kompilierung wird wegen der kleinen Dinge wie dem vergessenen „;-“-Zeichen nicht mehr unterbrochen [14].

2.2.3 Scriptsprachen

Lua

Lua ist eine alte Skriptsprache, die ursprünglich als Erweiterung für Programme in komplexeren Sprachen erstellt wurde. In dieser Sprache gibt es einige Merkmale, die Lua von einer Reihe von ähnlichen unterscheiden. Zum Beispiel der Beginn von Arrays mit 1 statt 0, oder das Fehlen von nativen Klassen [15].

Daher kann Lua für bestimmte Aufgaben als primäre Programmiersprache verwendet werden. Das beste Beispiel dafür ist das Corona SDK. Mit Corona können Programmierer leistungsstarke, funktionsreiche Anwendungen erstellen, die auf Windows, Mac, Android, iOS und sogar Apple TV und Android TV bereitgestellt werden können. Corona bietet auch Monetarisierung Möglichkeiten, und es ist ein anständiger Markt, in dem Entwickler nützliche Plugins finden können [15].

HTML 5, CSS und JavaScript

Diese drei Sprachen, die einst für die Entwicklung von Front-End-Anwendungen in einer Webumgebung entwickelt wurden, haben sich seitdem zu etwas größerem entwickelt. HTML 5, CSS und JavaScript-Tools reichen nun aus, um eine Vielzahl von Anwendungen für mobile Geräte und für klassische PCs zu erstellen. Im Wesentlichen erstellt der Programmierer eine Webanwendung, die die ganze Macht und Magie von Offline-Plattformen nutzen kann.

Programmierer können auf diese Weise Android-Anwendungen erstellen, indem die Funktionen von Adobe Cordova verwenden. Es ist ein Open-Source-Framework, das auch iOS-Betriebssysteme unterstützt, Windows 10 Mobile, Blackberry, Firefox, und viele andere. Was auch immer Cordova nützlich ist, es erfordert eine ernsthafte Arbeit, um eine anständige Anwendung darin zu erstellen. Daher bevorzugen viele Programmierer das Ionic Framework-Projekt.

Es gibt auch eine andere Möglichkeit: die Verwendung der React Native Bibliothek. Es kann auf Android, iOS bereitgestellt werden. Facebook, Instagram und andere große Unternehmen nutzen diese Bibliothek [15].

2.3 Integrierte Entwicklungsumgebung

Um eine Android-Anwendung zu entwickeln, müssen bestimmte Programmierumgebungen verwendet werden. Von Google gibt es eine offizielle Entwicklungsumgebung namens Android Studio. Neben der offiziellen IDE gibt es mehrere Analoga.

Die Integrierte Entwicklungsumgebung ist eine integrierte Software-Entwicklungsumgebung. Diese Software ermöglicht die Arbeit des Programmierers bequemer und produktiver zu machen.

IDE-Anforderungen:

- Sprachsyntax-Hervorhebung und Zeilennummerierung.
- Funktion zum Beenden des Codeschreibens und anzeigen von Parametern.
- Debuggen der Anwendung.
- Die Möglichkeit der Integration mit dem System der Kontrolle der Versionen des Codes.

IDE ist der Ort, an dem der Entwickler, die meiste Zeit verbringt, so dass die Wahl richtig gemacht werden muss.

2.3.1 Android Studio

Android Studio ist die offizielle IDE für Android von Google erstellt. Deshalb ist Android Studio die Nummer eins für Entwickler, die Apps erstellen möchten, nach Googles Materialdesign und Zugriff auf erweiterte Plattformfunktionen. Die Abbildung 2.2 zeigt das Umgebungsinterface des Android Studio 3.4.1

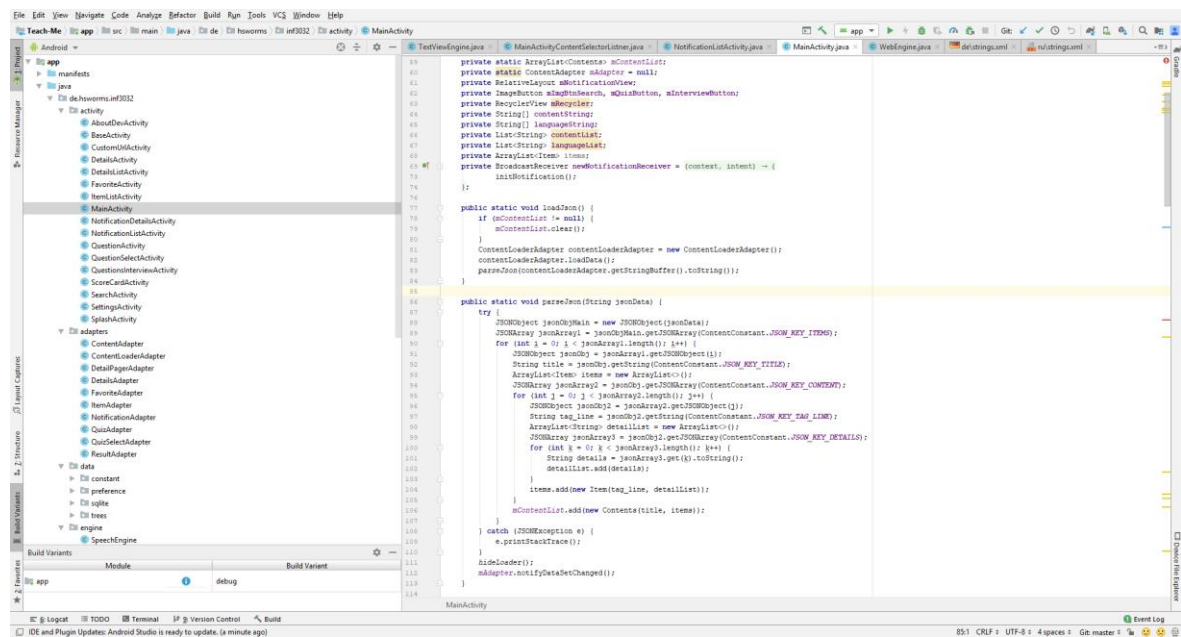
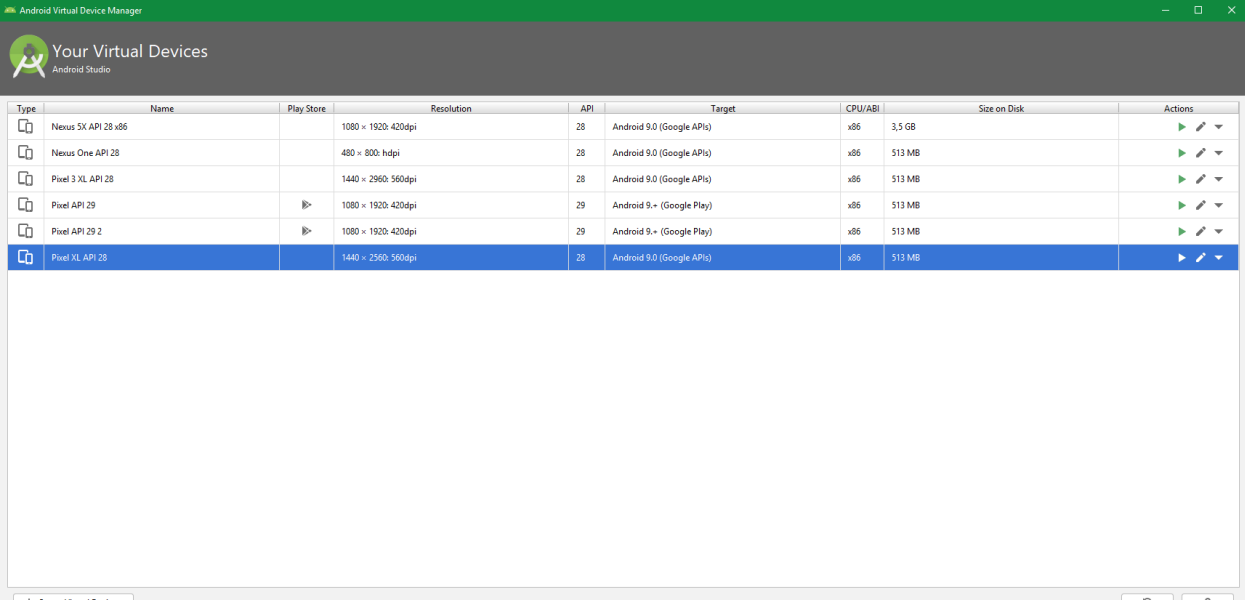


Abbildung 2.2: Android Studio 3.4.1 mit Geöffneter Projekt Teach Me.

Android Studio fungiert als Editor für die von Ihnen gewählte Programmiersprache. Es unterstützt Java, C++ sowie Kotlin. Android Studio fungiert auch als Compiler, der APK-Dateien und Dateisysteme erstellen kann, um Ihr Projekt zu organisieren. Darüber hinaus enthält es einen XML-Editor und einen erweiterten Layout-Editor. Android Studio bietet eine ganze Reihe von zusätzlichen Tools. Glücklicherweise können die meisten jetzt ein einzelnes Paket herunterladen. Im Wesentlichen kommt dieses Paket mit dem Android SDK, aber Java JDK muss immer noch separat heruntergeladen und installiert werden [16].

Android Virtual Device Manager

Das AVD Manager-Tool kommt mit Android Studio. Die Abkürzung AVD steht für Android Virtual Device. Es ist ein Emulator, um Android-Anwendungen auf Computer auszuführen. Es ist ein sehr nützliches Tool, mit dem Programmierer Ihre Anwendungen testen können, ohne sie auf physischen Geräten installieren zu müssen. Noch wichtiger ist, dass mit AVD Manager viele Emulatoren mit unterschiedlichen Bildschirmgrößen, Spezifikationen und Android-Versionen erstellt werden können. Entwickler können sehen, wie Ihre Kreation auf jedem Gerät aussehen wird, und damit Unterstützung unter den beliebtesten Gadgets bieten. Die Leistung des Werkzeugs wird ständig verbessert [17]. Die Abbildung 2.3 liefert ein Beispielweise Geräten, die auf AVD zu Verfügung stehen.



The screenshot shows the 'Android Virtual Device Manager' window. At the top, it says 'Your Virtual Devices' and 'Android Studio'. Below this is a table with columns: Type, Name, Play Store, Resolution, API, Target, CPU/ABI, Size on Disk, and Actions. The table lists several virtual devices, with 'Pixel XL API 28' selected (highlighted in blue).

Type	Name	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Nexus 5X API 28 x86		1080 x 1920: 420dpi	28	Android 9.0 (Google APIs)	x86	3.5 GB	
	Nexus One API 28		480 x 800: hdpi	28	Android 9.0 (Google APIs)	x86	513 MB	
	Pixel 3 XL API 28		1440 x 2960: 560dpi	28	Android 9.0 (Google APIs)	x86	513 MB	
	Pixel API 29		1080 x 1920: 420dpi	29	Android 9.0 (Google Play)	x86	513 MB	
	Pixel API 29.2		1080 x 1920: 420dpi	29	Android 9.0 (Google Play)	x86	513 MB	
	Pixel XL API 28		1440 x 2960: 560dpi	28	Android 9.0 (Google APIs)	x86	513 MB	

At the bottom left, there is a button '+ Create Virtual Device...'. At the bottom right, there are two buttons: a refresh icon and a help icon.

Abbildung 2.3: Auswahl von Geräten in ADV Manager.

Android Device Monitor

Android Device Monitor ist ein weiteres integriertes Tool des Android Studio. Mit ADM dem können Entwickler ihr physisches oder virtuelles Gerät überwachen, während es läuft. Über ADM können Entwickler Informationen darüber erhalten, wie viele Prozesse im Stream ausgeführt werden, Netzwerkstatistiken und LogCat. Dieses Tool eignet sich hervorragend zum Testen der Anwendungsleistung [17].

Android Debug Bridge

Android Debug Bridge ist ein Befehlszeilentool, mit dem Programmierer Dateien auf und von Gerät kopieren, Apps installieren und deinstallieren und auf allen Android-basierten Geräten Daten sichern und wiederherstellen können [17].

2.3.2 Basic for Android

Basic for Android ist ein wenig bekanntes Tool für die Entwicklung von Android-Anwendungen von Anywhere Software, spezialisiert auf das Konzept der schnellen Anwendungsentwicklung. B4A ist eine IDE und Interpreter, mit dem Entwickler die Anwendungen mit der Programmiersprache Basic erstellen können. Basic ist eine prozedurale Programmiersprache, die fast wie normales Englisch gelesen wird. B4A ist eine IDE mit viele nützliche erweiterte Funktionen wie drahtloses Debuggen über Bluetooth, einen visuellen Editor und mehreren weiteren Modulen. Abbildung 2.4 zeigt die Umgebungsinterface des B4A

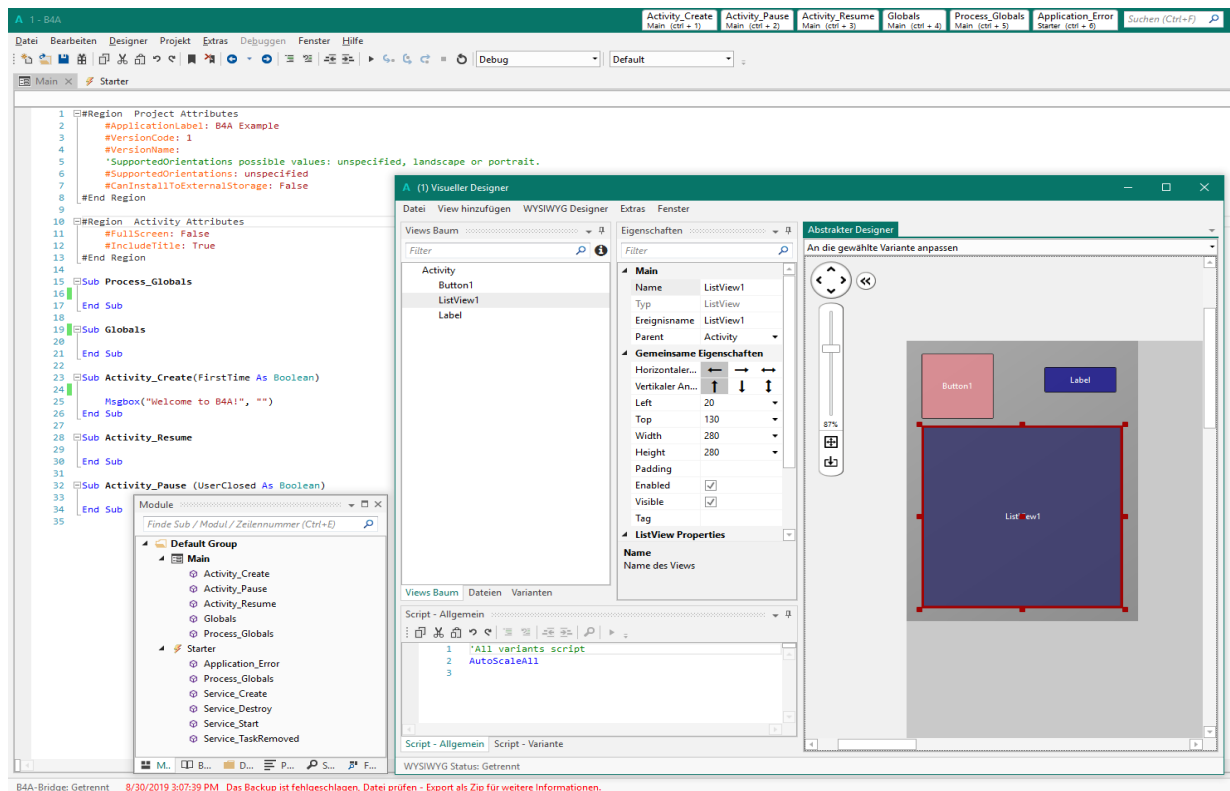


Abbildung 2.4: Interface der IDE Basic for Android.

Mit B4A können Entwickler fast alles tun, was die auch mit Android Studio machen, aber viel schneller und mit weniger Vorlagen. Jede Leistungsminderung im Vergleich zu anderen IDEs ist minimal. Entwickler sollten jedoch immer noch die offizielle Methode zum Erstellen von Anwendungen erlernen, insbesondere wenn bestimmte Java-Bibliotheken verwenden möchten, die für Basic for Android entwickelt werden können [18].

2.3.3 Visual Studio

Visual Studio ist Microsofts IDE, die eine Reihe von Entwicklungssprachen unterstützt, einschließlich C#. VB.net, JavaScript und mehr. Mit Visual Studio enthaltenen Xamarin-Framework können Entwickler plattformübergreifende Anwendungen mit C# erstellen und dann auf mehreren Geräten testen, die mit der Cloud verbunden sind. Es ist eine gute und kostenlose Wahl, die App sowohl für Android als auch für iOS zu veröffentlichen und der Code zweimal zu schreiben. Es ist auch eine gute Wahl für diejenigen, die bereits mit C# und/oder Visual Studio vertraut sind. Der Nachteil ist, dass Xamarin bei der Verwendung von Java-Bibliotheken unbequem ist und wie bei jeder anderen Android Studio-Alternative die Unterstützung von Google und die erweiterten integrierten Funktionen verloren gehen [18].

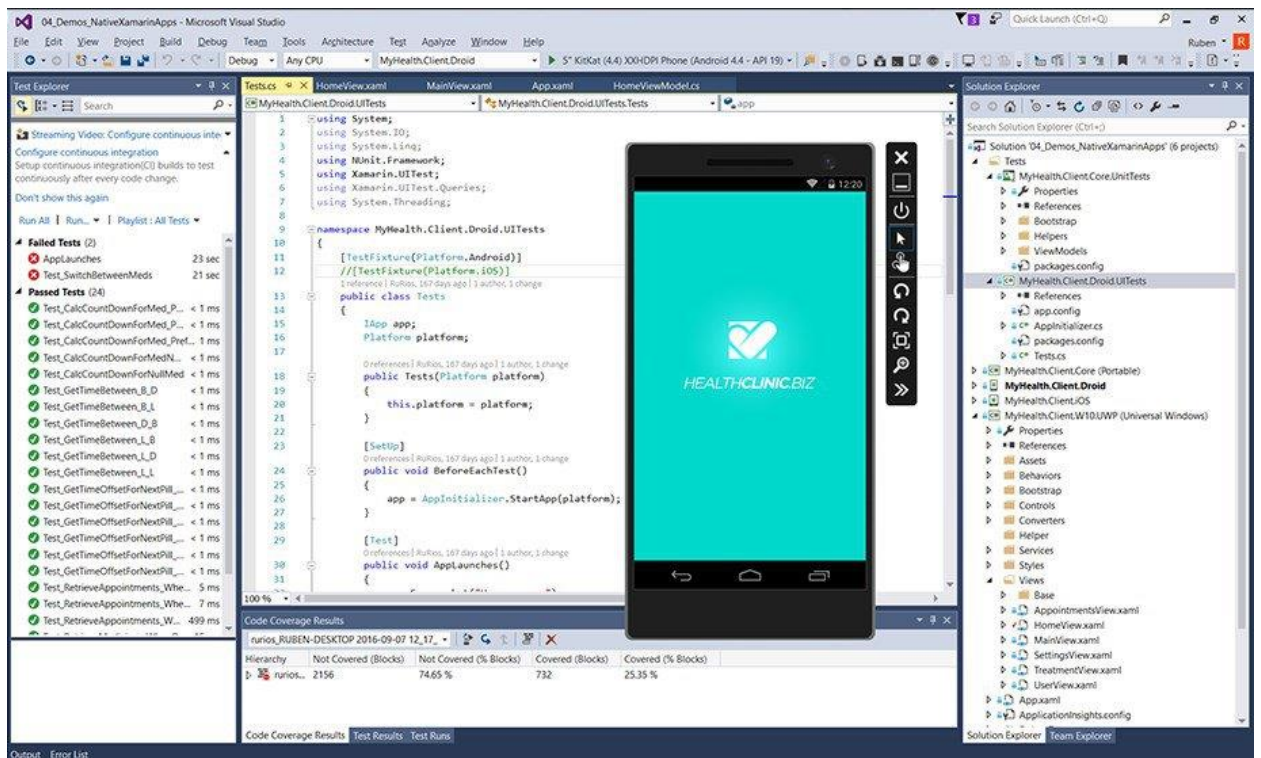


Abbildung 2.5: Interface der IDE Visual Studio

2.3.4 AIDE

AIDE steht für Android IDE und ist insofern einzigartig, als es auf Android selbst funktioniert. Das bedeutet, dass die Apps auf Smartphone oder Tablet werden erstellt und dann auf demselben Gerät getestet werden können. IDE funktioniert sehr gut mit Samsung DeX. AIDE fehlen einigen Funktionen aus Android Studio und es hat keinen wirklichen Vorteil gegenüber funktionelleren IDEs für die Entwicklung unter Android. IDE ist mehr für die Studierenden, die gerade Ihre Informatik Studium erst begonnen haben. Studenten können ihr eigenes mobiles oder einfaches Java / C++ Projekte zu entwickeln. AIDE bietet die Möglichkeit ein vorgeschriebenes Programmierlehrbuch zu lesen und gleichzeitig den Code von dort in Echtzeit zu überprüfen [18]. Die Programmoberfläche ist in Abbildung 2.6 dargestellt.

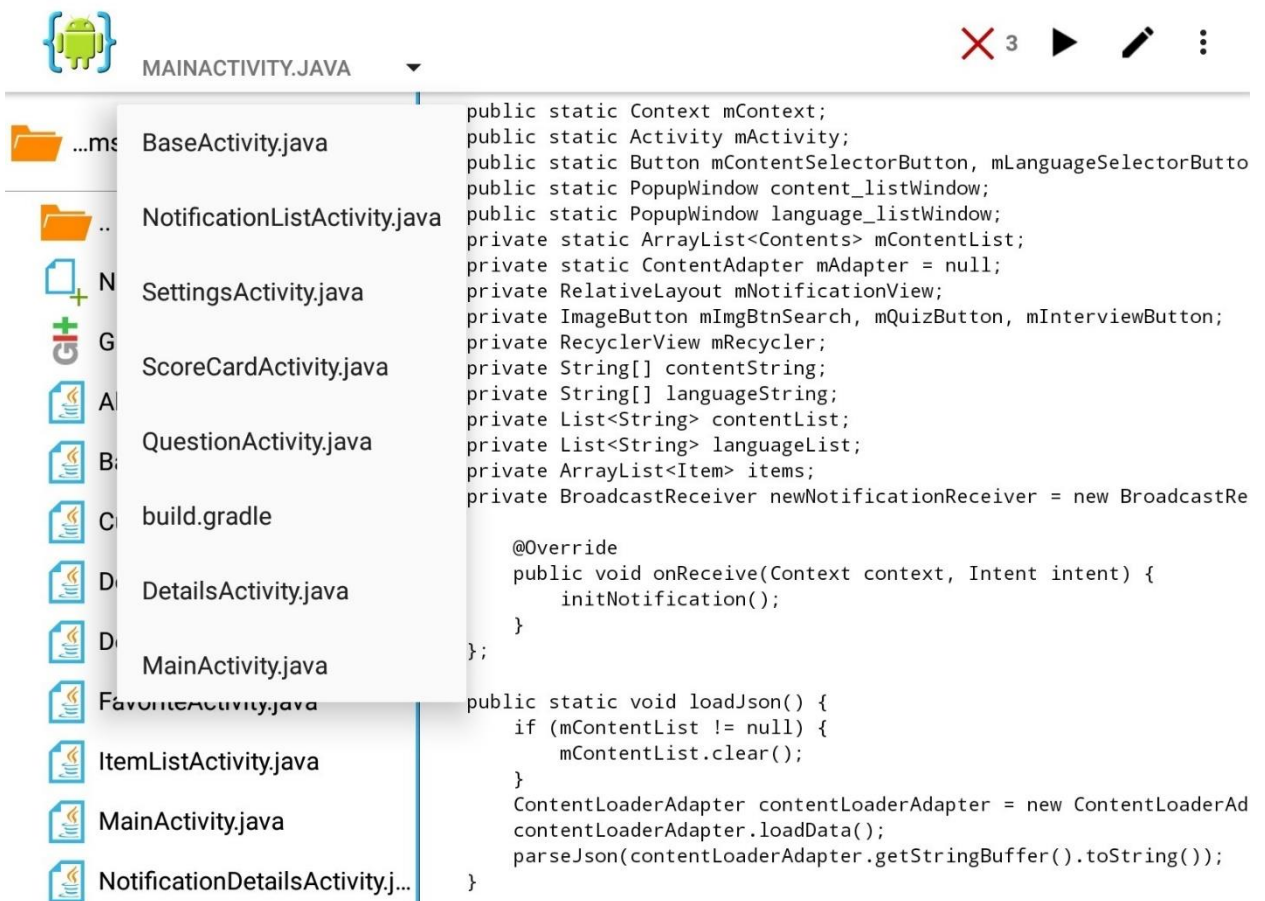


Abbildung 2.6: AIDE Interface mit geöffnetem Projekt Teach Me.

2.4 Android Betriebssystem

Android ist Betriebssystem für Smartphones, Internet-Tablets, E-Books, digitale Player, Armbanduhren, Spielekonsolen, Netbooks, Smartbooks, Google-Brillen, Fernseher und andere Geräte. Basierend auf dem Linux-Kernel und Googles eigener Java Virtual Maschine-Implementierung. Ursprünglich entwickelt von Android, Inc., die dann von Google gekauft wurde. Anschließend hat Google die Gründung der Open Handset Alliance initiiert, die sich jetzt mit der Unterstützung und Weiterentwicklung der Plattform beschäftigt. Mit Android könnten Java-Anwendungen erstellt, die das Gerät über die von Google entwickelten Bibliotheken steuern. Mit dem Android Native Development Kit können Bibliotheken und Anwendungskomponenten portieren, die in C und anderen Sprachen geschrieben sind [19].

Anfang 2019 gelang es Forschungen aus verschiedenen Quellen herauszufinden, dass der durchschnittliche Anteil von Android-Smartphones auf dem Smartphone-Markt im Juli 2019 beträgt 76,08%. Insgesamt wurden im Jahr 2019 mehr als 344 Millionen Geräte verkauft [20]. Abbildung 2.7 zeigt den Marktanteil der mobilen Betriebssysteme weltweit

Mobile Operating System Market Share Worldwide

Mobile Operating Systems	Percentage Market Share
Mobile Operating System Market Share Worldwide - July 2019	
Android	76.08%
iOS	22.01%
KaiOS	0.81%

Abbildung 2.7: Marktanteil der mobilen Betriebssysteme weltweit. [20]

2.4.1 Anwendungskomponenten

Android-Apps bestehen aus den folgenden teilen:

- **Activity** ist ein Ansichtsschema für Android-Apps. Zum Beispiel der Bildschirm, den der Benutzer sieht. Die Android-App kann mehrere Aktivierungen haben und kann während der Ausführung der App zwischen ihnen wechseln.
- **Views** Benutzeroberfläche aktiviert, die von Klassen-Widget erstellt wird.
- **Services** führt Hintergrundaufgaben aus, ohne eine Benutzeroberfläche bereitzustellen. Entwickler können den Benutzer über das Android-Benachrichtigungssystem Benachrichtigen.
- **Content Provider** stellt Daten für Anwendungen bereit. Die Anwendung kann Daten mit anderen Anwendungen teilen. Android enthält eine SQLite-Datenbank, die ein Content-Provider sein kann.
- **Intents** sind asynchrone Nachrichten, die es einer Anwendung ermöglichen, Funktionen von anderen Diensten abzufragen oder zu aktivieren. Die App kann direkte Intents zu einem Dienst oder Activity machen, oder von Android nach registrierten Intent Diensten und Apps fragen. Zum Beispiel kann die Anwendung über Intent einen Kontakt aus der Kontaktanwendung des Geräts anfordern. Die Anwendung registriert sich selbst im Internet über Intent Filter.
- **Broadcast Receiver** akzeptiert Systemnachrichten und implizite Intents, kann verwendet werden, um auf eine Änderung des Systemstatus zu reagieren. Die Anwendung kann sich als Empfänger bestimmter Ereignisse registrieren und kann gestartet werden, wenn ein solches Ereignis eintritt.

Der Lebenszyklus einer Anwendung in Android wird vom System streng überwacht und hängt von den Bedürfnissen des Benutzers, den verfügbaren Ressourcen ab. Die Entscheidung, die Anwendung zu starten, trifft das System. Das System unterliegt bestimmten angegebenen und logischen Regeln, mit denen System bestimmen kann, ob eine Anwendung heruntergeladen, angehalten oder beendet werden kann. Wenn der Benutzer derzeit mit einem bestimmten Fenster arbeitet, hat das System der entsprechenden Anwendung. Umgekehrt, wenn das Fenster unsichtbar ist und das System entscheidet, dass die Anwendung beendet werden muss, um zusätzliche Ressourcen zu mobilisieren, wird die Anwendung beendet. In Android sind die Ressourcen begrenzter, so dass Android die Funktionsweise von Apps strenger überwacht [17].

Grundlegende Methoden des Anwendungslebenszyklus:

- `protected void onCreate()`
- `protected void onStart()`
- `protected void onRestart()`
- `protected void onResume()`
- `protected void onPause()`
- `protected void onStop()`
- `protected void onDestroy();`

Die Abbildung 2.8 zeigt wie der Anwendungslebenszyklus läuft.

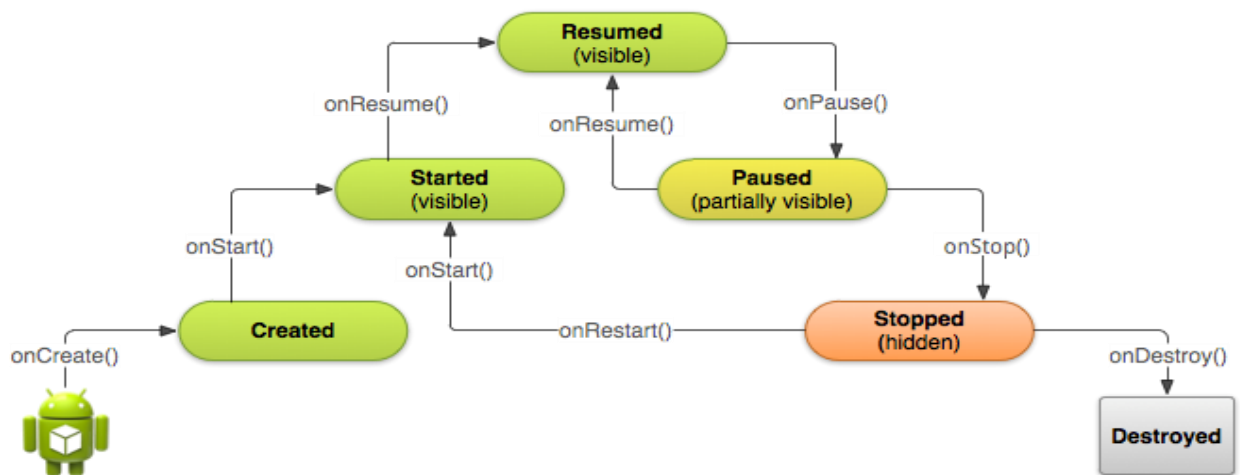


Abbildung 2.8: Lebenszyklus einer Android Anwendung [21].

2.4.2 Pattern

Bei der Entwicklung komplexer Anwendungen können Probleme auftreten, die wahrscheinlich vorher aufgetreten sind und bereits eine große Anzahl von Lösungen haben. Solche Lösungen werden als Muster bezeichnet. In der Regel wird über Designmuster und Architekturmuster gesprochen. MVP vereinfachen die Entwicklung von Anwendungen [22].

Model View Presenter

Model View Presenter ist ein Entwicklungsmuster für das Android-Betriebssystem, das vorschlägt, die Anwendung in die folgenden drei Teile aufzuteilen:

- Model ist ein Wrapper für die empfangenen Daten. Dabei gibt es einen besonderen Unterschied, wo die Daten nicht sein sollten, die Daten der Netzwerkanforderungen oder die Daten der Interaktion des Benutzers mit der Benutzeroberfläche. Ein guter Ort, um Caches zu implementieren. Es ist eine gute Praxis, für jede Antwort des Servers ein einzigartiges Modell zu erstellen, um Schnittpunkte und nachfolgende Probleme bei API-änderungen zu reduzieren.
- Presenter verbindet zwischen der Verarbeitung von Daten, die von Model und dem Aufruf von Methoden und von View abgerufen werden, und implementiert der Verbindung der UI-Komponenten mit den Daten. Presenter-Methoden werden von Activity/Fragment- Lebenszyklusmethoden aufgerufen und sind oft symmetrisch.
- View zeigt die empfangenen Daten aus Presenter an. In der richtigen Implementierung hat das View Objekt keine Ahnung von den empfangenen Daten von außen, es sollte nur anzeigen was Presenter von ihm verlangt. View kann jede Aktivität oder Fragment in der App unter Android OS sein.

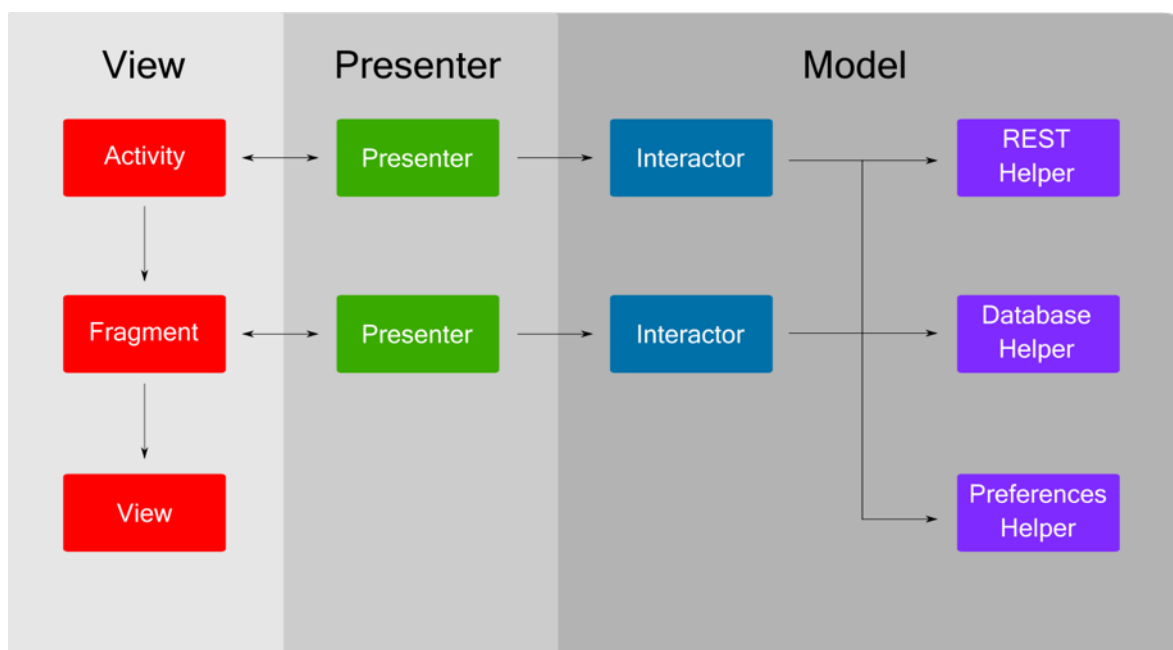


Abbildung 2.9: MVP Pattern. [22]

Model View Presenter sollte Schnittstellen für mehr Flexibilität bei der Codeänderung darstellen.

MVP hat eine Reihe von Vorteilen gegenüber dem Standard App Entwicklung Schema, eines davon ist eine gute Testabdeckung, die sowohl für die Implementierung der App als auch für die Flexibilität wichtig ist. Die Anwendung weiß nicht, was passiert, bevor das Element auf dem Bildschirm reflektiert wird, die gesamte Logik ist von der Ansicht geschlossen. Zwei Punkte sind für die Schaffung einer guten Architektur berücksichtigt. Da MVP die Logik und die Anzeige von Elementen teilen, und wenn die Anwendung einen Client – Server-Struktur teilt und Anforderungen an den Server. Mit MVP ist es auch möglich, einen ziemlich einfachen und verständlichen Code zu erreichen, dies ist der Dritte Punkt einer guten Anwendungsarchitektur [17].

2.5 Git Versionierung

Git ist eine Art Version Control System. Version Control System ist ein Programm, um mit ständig wechselnden Informationen zu arbeiten. VCS kann viele Versionen derselben Datei speichern und zu einem früheren Zustand zurückkehren.

Diplomarbeit nutzt Hosting-Service von Github.com und das Projekt Teach Me ist unter: <https://github.com/AuroraSyN/Teach-Me> verfügbar

Github.com es ist ein Web-Service von Projekten mit dem Git-Versionskontrollsystem sowie als soziales Netzwerk für Entwickler. Benutzer können eine unbegrenzte Anzahl von Repositorien erstellen, von denen jedes ein Wiki, ein Issues-tracking-System, eine Möglichkeit zur Durchführung von Code Review und vieles mehr bietet.

2.6 Auswahl einer Technologie

Bei der Entwicklung der Anwendung stellte sich die Frage nach der Wahl der Sprache, DIE und Architektur. Es ist notwendig, die Optimalen für dieses Projekt auszuwählen. Eine gute Architektur macht den Prozess der Entwicklung und Wartung des Programms einfacher und effizienter. Ein Programm mit guter Architektur ist einfacher zu erweitern, ändern, testen, debuggen und zu verstehen.

Die Anwendungsarchitektur muss mehrere Hauptkriterien erfüllen, nämlich:

- Die Flexibilität des Systems – die Veränderung eines Fragments des Systems darf seine anderen Fragmente nicht beeinflussen und die Folgen der architektonischen Fehler sollen in dem angemessenen Umfang begrenzt sein.
- Erweiterbarkeit des Systems – die Möglichkeit, dem System neue Entitäten und Funktionen hinzuzufügen, ohne seine Hauptstruktur zu stören. Die Architektur sollte es ermöglichen, zusätzliche Funktionen beim Bedarf leicht zu erweitern.
- Testbarkeit – Code, der einfacher zu testen ist, weniger Fehler enthält und zuverlässiger funktioniert. Aber Tests verbessern nicht nur die Qualität des Codes. Viele Entwickler kommen zu dem Schluss, dass die Anforderung einer guten Testbarkeit auch eine Führungskraft ist, die automatisch zu einem guten Design führt.

Die Anwendung muss mehrere Hauptkriterien erfüllen, die zur Verbesserung der Architektur beitragen und den Code näher an den Begriff saubere Architektur bringen. Alle Kriterien für eine gute Anwendung werden im MVP Pattern, in der Programmiersprache Java und in der Android Studio kombiniert, die bei der Entwicklung der Anwendung Teach Me verwendet wurden.

3. Tutorien an den deutschen Universitäten

Viele deutsche Universitäten kombinieren die Tutorien mit Übungsgruppen und für viele Studenten ist Anwesenheit bei den Übungsgruppen Pflicht. Um eine Teilnahmebestätigung an den Tutorien oder Übungsgruppe zu erhalten, müssen Studierende sich über ein Informationssystem anmelden. Universitäten in Deutschland haben verschiedene Informationssysteme für die Studierenden. In den Informationssystemen bekommen die Studierenden ihr Vorlesungsverzeichnis, ihre Übungsgruppenauswahl oder -einteilung und die Tutorien.

Fakultät für Mathematik und Informatik der Ruprecht-Karls-Universität Heidelberg bietet Mathematisches Übungsgruppen- und Scheinlisten-Interface. Dort befinden sich alle Vorlesungen, Übungsgruppen und Tutorien gelistet. Studenten können in MÜSLI ihr Modul auswählen und zu der dazugehörigen Übungsgruppe sich anmelden. Die Abbildung 3.1 zeigt das Interface vom MÜSLI an.

MÜSLI

Mathematisches Übungsgruppen- und Scheinlisten-Interface

[Übersicht](#) [Vorlesungen](#) [Angaben ergänzen](#) [E-Mail ändern](#) [Passwort ändern](#) [API](#) [Abmelden](#) [Kontakt](#) [Neues](#)

Einführung in die Theoretische Informatik

Internetseite zu dieser Vorlesung: https://www.math.uni-heidelberg.de/logic/ss19/theoinf_ss19.html

Übungsgruppen

Zu dieser Vorlesung werden die folgenden Übungsgruppen angeboten:

Zeit	Raum	Auslastung	Tutor	Kommentar
Mo 14:00	SR Statistik 02.104	<div><div></div></div> 22/25	Patrick Arras	Beitreten
Mo 16:00	Hörsaal Mathematik	<div><div></div></div> 25/25	Markus Schäfers	
Mo 16:00	SR Statistik 02.104	<div><div></div></div> 21/25	Tim Karl	Beitreten
Do 14:00	SR 9	<div><div></div></div> 12/25	Tim Karl	Beitreten
Do 14:00	SR Statistik 02.104	<div><div></div></div> 23/25	Patrick Arras	Beitreten
Do 16:00	SR 1	<div><div></div></div> 25/25	Markus Schäfers	
So 23:59	Mare ingenii	<div><div></div></div> 6/100		Für alle, die Ihre Zulassung von letztem Jahr haben Beitreten

Abbildung 3.1: Auswahl einer Übungsgruppe für die Einführung in die Theoretische Informatik in MÜSLI.

Fachbereich 08 – Physik, Mathematik und Informatik der Johannes-Gutenberg-Universität Mainz bietet auch für Informatik Studierende eine Website des Instituts für Informatik. Auf der Website können die Studenten die Vorlesungen und Übungsgruppen auswählen. Die Anmeldung für die Übungsgruppen erfolgt durch JOGU-StIne. JOGU-StIne ist eine offizielles Studentenportal der Johannes-Gutenberg-Universität. Dort können Studenten nicht nur für die Übungsvorlesungen anmelden, sondern auch ihre Leistungen oder Studienbescheinigungen bekommen. Die Abbildung 3.2 liefert das Vorlesungsverzeichnis für Winter Semester 19/20 von JOGU-StIne an.

Vorlesungsverzeichnis

Übersicht > Fachbereich 08 - Physik, Mathematik und Informatik > Informatik > Vorlesungen

Veranstaltungen / Module	
Veranstaltung / Modul Dozenten / Modulverantwortliche Zeitraum	Veranstaltungsart Raum
08.079.010 Einführung in die Programmierung Dr. rer. nat. Stefan Endler Di, 15. Okt. 2019 [14:00] - Di, 4. Feb. 2020 [16:00]	Vorlesung/Übung
08.079.015 Einführung in die Softwareentwicklung Univ.-Prof. Dr. Andreas Hildebrandt Mo, 14. Okt. 2019 [14:00] - Mo, 3. Feb. 2020 [16:00]	Vorlesung/Übung
08.079.020 Software-Engineering / Software-Technik Univ.-Prof. Dr. Stefan Kramer Fr, 18. Okt. 2019 [10:00] - Fr, 7. Feb. 2020 [14:00]	Vorlesung/Übung
08.079.050 Formale Sprachen und Berechenbarkeit Markus Blumenstock; Univ.-Prof. Dr. Friederike Schmid Mi, 16. Okt. 2019 [10:00] - Mi, 5. Feb. 2020 [12:00]	Vorlesung/Übung
08.079.055 Komplexitätstheorie Univ.-Prof. Dr. Ernst Althaus Di, 15. Okt. 2019 [10:00] - Di, 4. Feb. 2020 [12:00]	Vorlesung/Übung
08.079.060 Datenstrukturen und effiziente Algorithmen Frank Fischer Mo, 14. Okt. 2019 [10:00] - Do, 6. Feb. 2020 [12:00]	Vorlesung/Übung
08.079.080 Technische Informatik Univ.-Prof. Dr. André Brinkmann Mo, 14. Okt. 2019 [10:15] - Mo, 3. Feb. 2020 [12:00]	Vorlesung/Übung

Abbildung 3.2: Informatik Vorlesungsverzeichnis für das Winter Semester 19/20 in JOGU-StIne.

Hochschule Worms hat auch eine Informationssystem LSF. LSF steht für Lehre, Studium und Forschung. Dort können die Studierenden Vorlesungen und Übungsgruppen auswählen. Die Abbildung 3.3 stellt das LSF dar.

Studentisches Leben		Veranstaltungen	Organisationseinheiten	Prüfungsmanagement
---------------------	--	-----------------	------------------------	--------------------

Sie sind hier: [Home](#) → [Studiengangpläne](#)

> Studiengang - Lehrplan

Liste: → kurz →

Angewandte Informatik, Abschluss 05, PrüfungsOrdnung 2012 (05938), Semester von: 1, Semester bis: 1

Einzel- oder Blockveransta

Vst.-Nr.	Veranstaltung	Vst.-Art	Semester	FB / Einrichtung
(Keine Nummer)	Diskrete Mathematik	Vorlesung	WiSe 2019/20	Informatik (FB)
(Keine Nummer)	Einführung in die Informatik	Vorlesung/Übung	WiSe 2019/20	Angewandte Informatik
(Keine Nummer)	Hardware-Konzepte (Grundlagen/Grundkomponenten)	Vorlesung/Übung	WiSe 2019/20	Informatik (FB)
(Keine Nummer)	Prozedurale Programmierung (Programmieren 1)	Vorlesung/Übung	WiSe 2019/20	Informatik (FB)
(Keine Nummer)	Selbst- u. Methodenkompetenz (Soft Skills 1)	Vorlesung	WiSe 2019/20	Informatik (FB)

Abbildung 3.3: Vorlesungsplan für das 1. Semester der Angewandten Informatik.

3.1 Sinn und Zweck von Tutoren an Universitäten

Ein Tutorium ist eine Veranstaltung, die von einem Dozenten im Rahmen seines Kurses oder einer Unterrichtsveranstaltung angeboten wird. Studenten können den Unterrichtsstoff der Lehrveranstaltung noch einmal mit einem Tutor wiederholen und oder vertiefen. Tutorien sind im Regelfall freiwillig, können aber auch vom Dozenten vorgeschrieben werden. Tutorien sind auch dazu da, gemeinsam die Hausaufgaben zu bearbeiten, der Tutor unterstützt und steht bei Fragen zur Verfügung.

In den meisten Fällen ist der Tutor selbst noch ein Student, aber in einem höheren Fachsemester. Ein Tutor hilft dem Dozenten als studentische Hilfskraft und unterstützt andere Studenten, die vor allem Erstsemester oder in einem niedrigen Fachsemester sind, mit seinem Wissen.

3.2 Arten von Tutorien

Es ist nicht selten, dass ein Tutor mehrere Tutorien leitet. Ein Tutorium kann sowohl eine Gruppenbetreuung als auch eine persönliche Betreuung sein. Das letztere ist in Großbritannien weit verbreitet, man nennt es das Personal Tutoring System, dabei wird jedem Studierenden ein Hochschullehrer, der ihm als Ansprechperson dienen soll, zugewiesen. Im Gegensatz dazu wird in Deutschland eher auf Gruppenbetreuung gesetzt wegen der hohen Studierendenzahlen. Dazu werden in Deutschland fast ausschließlich studentische Tutoren eingesetzt. Tutorien unterscheiden sich in zwei Arten: Orientierungstutorien und Fachtutorien / Übungsgruppen [23].

3.2.1 Orientierungstutorien

Am Anfang des ersten Semesters gibt es meistens Orientierungstutorien, die sich im Regelfall über drei Tage erstrecken, diese geben den Studienanfängern eine Einführung an der Hochschule, im Studienfach und am Hochschulort. Diese Einführungen können sich an Studienfänger im Allgemeinen oder aber auch nur an bestimmte Zielgruppen richten. Viele Anforderungen folgen bei einem Start ins Studium. Zum einen muss der Studierende nun lernen, für seine akademischen Leistungen selbst verantwortlich zu sein.

Zum anderen muss er auch lernen sich außerhalb der Universität zu Recht zu finden, da für viele der Eintritt in eine Hochschule mit einem Wohnortswechsel verknüpft ist. Darum gibt es bei Orientierungstutorien auch Informationen über die Stadt und über das Leben als Student [23].

3.2.2 Fachtutorien und Übungsgruppen

Unterrichtsveranstaltungen werden meistens im ersten Studienjahr von Fachtutorien oder Übungsgruppen begleitet. Ein Fachtutorium und Übungsgruppe ermöglicht den Studenten erforderliche Grundfähigkeiten für das Studium zu erwerben, weiter auszubauen und auch zu trainieren. Viele Lernangebote erfordern eine Lernselbstständigkeit von den Studierenden, ohne dass man ihnen beim Entwickeln zu helfen braucht oder Lernziele auszuformulieren. Wie ein Tutorium abläuft, wird vom jeweiligen Dozenten selbst bestimmt. Eine enge Zusammenarbeit zwischen Dozentem und Tutor ist dabei unablässig, damit das Tutorium so gut wie möglich an die Lernveranstaltung anzupassen ist [23].

3.3 Aufgaben der Tutoren und Tutorinnen

Fachlich sind Tutoren meist sehr gut, doch dies bringt den Studierenden wenig, wenn die didaktischen Kompetenzen sehr zu wünschen übriglassen und in der Kürze der Zeit eine gute Erklärung fehlt. Schnell wird die Übungsgruppe als überflüssig erachtet, da die Lösungen eh nur eilig vorgestellt werden können. Es werden kaum Fragen gestellt, weil sich niemand traut. Der Mehrwert der Tutorien hängt stark von den didaktischen Fähigkeiten der Tutoren ab.

Dabei bieten Tutorien viele Möglichkeiten, den Vorlesungsstoff noch einmal in einer kleineren Gruppe zu besprechen, das Wissen durch Präsenzaufgaben zu verfestigen und gemeinsame Diskussionen zu führen. Studis sollten nicht mehr mit ihren Fragen und Problemen daheim alleine gelassen werden, sondern in den Übungsgruppen die Chance haben, von ihrem Tutor betreut zu werden. Tutoren müssen auch bei den Vorbereitungen von Veranstaltungen helfen, zu dem auch den Dozenten unterstützen und ausländische Studenten betreuen, neben ihrer eigenen Lehrtätigkeit [24].

Folgende Aufgaben liegen im Tätigkeitsbereich eines Tutors:

- **Gruppenleitung:** Der Tutor sorgt für eine Struktur, einen zeitlichen Ablauf, die Verteilung von Aufgaben, so dass die Ziele bis zum Ende des Semesters erreicht werden können.
- **Lernbegleitung:** Tutoren führen sowohl in Inhalte als auch in Arbeitsformen ein. Tutoren stellen, wo nötig Arbeitsmittel zur Verfügung und sorgen für die Lösung von Raum-problemen.
- **Moderation:** Moderation kann sowohl bei der Erarbeitung oder Diskussion von Inhalten, bei Absprachen, als auch bei zwischenmenschlichen Störungen und Konflikten sinnvoll eingesetzt werden.
- **Information:** Alle notwendigen Informationen müssen immer durch das Tutor zur Verfügung gestellt werden. Das können Informationen des Professors oder Leitung des Semesters oder Informationen
- **Rückmeldung geben:** Sowohl Anerkennung als auch nachvollziehbare Kritik gehören zu den Aufgaben ein Tutor.

Die Abbildung 3.4 zeigt wie der Aufbau eines Tutoriums abläuft.



Abbildung 4: Der Aufbau eines Tutoriums [25].

4 Konzeption und Entwurf

Als erstes werden im 4. Kapitel von funktionale, optionalen und nicht-funktionalen Anforderungen beschrieben, die in die App implementiert werden sollen. Danach werden das Mockup und der Prototyp dargestellt.

4.1 App Anforderungen

Im Kapitel 4.1 werden funktionale-, optionale und nicht-funktionale Anforderungen beschrieben. Diese Anforderungen müssen bestimmte Qualitätsanforderungen erfüllen. Anforderungen müssen eindeutig, korrekt, vollständig und verifizierbar sein. Folgende Tabellen zeigen diese Anforderungen, welche an die mobile Anwendung Teach Me gestellt werden.

4.1.1 Funktionale Anforderungen

Funktionale Anforderungen bilden die wichtigsten Funktionen der mobilen Anwendung. Um diese zu ermitteln, muss die Frage gestellt werden: Was soll das System können? Auf der Tabelle 1 und 2 wird beschrieben, welche funktionale Anforderungen die App Teach Me erfüllen soll. Es werden somit Funktionen vorgestellt, welche dem Nutzer angeboten werden sollen.

No	Aufgabe	Beschreibung
F.1	Lernmodul	Der Nutzer soll eine Möglichkeit haben um sich die verschiedene Lernmodule zu Auswahl haben.
F.2	Suche	Der Nutzer soll eine Möglichkeit haben Inhalt nach einem Begriffswort suchen zu können.
F.3	Quizfragen	Der Nutzer soll eine Möglichkeit haben, um die Quizfragen zu beantworten zu können.
F.4	Interviewfragen	Der Nutzer soll eine Möglichkeit haben die Interviewfragen beantworten zu können.

Tabelle 1: funktionale Anforderungen von F.1 bis F.4.

Nº	Aufgabe	Beschreibung
F.5	Video	Der Nutzer soll eine Möglichkeit haben den Videolernstoff anzuschauen.
F.6	Textgröße	Der Nutzer soll eine Möglichkeit haben die Textgröße von Inhalt zu wechseln.
F.7	Sprachauswahl	Der Nutzer soll eine Möglichkeit haben den Inhalt auf einer anderen Sprache zu lernen.
F.8	Favoriten	Der Nutzer soll eine Möglichkeit haben den Inhalt zu Favoriten hinzufügen.
F.9	Textvorlesen	Der Nutzer soll eine Möglichkeit haben den Inhalt sich vorlesen zu lassen.
F.10	Inhaltskopieren	Der Nutzer soll eine Möglichkeit haben den Inhalt zu kopieren.
F.11	Einstellungen	Der Nutzer soll in die Benachrichtigungen an- und auszuschalten.

Tabelle 2: funktionale Anforderungen von F.5 bis F.11.

4.1.2 Optionale Anforderungen

Die Tabelle 3 zeigt die optionalen Anforderungen der App Teach Me. Optionale Anforderungen bilden den Zusatz zu den wichtigsten Funktionen der mobilen Anwendung.

Nº	Aufgabe	Beschreibung
0.1	Teilen	Der Nutzer soll eine Möglichkeit haben den Lerninhalt und Quizergebnisse zu teilen.
0.2	Breitbild	Der Nutzer soll eine Möglichkeit haben den Inhalt mit Breitbild anzuzeigen.
0.3	Ton	Die Quizfragen sollen dem Ton begleiten.
0.4	Quizergebnisse	Der Nutzer soll eine Möglichkeit haben die Quizergebnisse anzusehen.
0.5	Benachrichtigung	Der Nutzer soll eine Möglichkeit haben die Benachrichtigungen von Entwickler zu bekommen.

Tabelle 3: optionale Anforderungen.

4.1.3 Nicht-Funktionale Anforderungen

Nichtfunktionale Anforderungen bilden eine Klasse von Anforderungen, die aufgrund Ihrer Allgemeinen Bedeutung Häufig projektübergreifend eingesetzt werden. Es untersucht die Qualitätsattribute der Funktionen, Anforderungen an die Anwendung als Ganzes, Anforderungen an die Implementierung der Systemerstellung und Anforderungen an Test, Einführung, Support und Betrieb.

Die Tabelle 4 zeigt die nicht-funktionalen Anforderungen der App Teach Me

Nº	Aufgabe	Beschreibung
NF.1	Einheitliche Darstellungsformen der Quizfragen	Die Darstellungsmöglichkeiten der Fragen sollen einheitlich und einfach aufgebaut werden.
NF.2	Usability Ziele einhalten	Die App sollte einfach und verständlich aufgebaut werden.
NF.3	Verfügbarkeit	Die App sollte in jeder Auflösung darstellbar sein. Beispielsweise sollte die App auch für Tablets angeboten werden.
NF.4	Selbsterklärbarkeit	Die App sollte selbsterklärend aufgebaut sein. Nutzer sollte keine Probleme beim Verständnis der Funktionen oder Icons haben.
NF.5	Aktuelle Version der Betriebssysteme	Die Oberfläche sollte sich dem aktuellen Design anpassen.

Tabelle 4: nicht-funktionale Anforderungen.

4.2 Mockups

Mockups sind sehr wichtig in der Entwicklung von Apps. Mockups werden relativ früh im Systementwicklungsprozess erstellt. Dadurch kann sehr schnell geklärt werden, ob man auf dem richtigen Weg ist. Papier Mockups haben einige Vorteile gegenüber elektronischen Prototypen. Mockups sind schneller erstellbar und dadurch billiger. Außerdem sind Mockups änderungsfreundlicher, da sie einfacher umzugestalten sind. In Abbildung 4.1 sind ein paar erste Entwürfe der App Teach Me anzusehen.

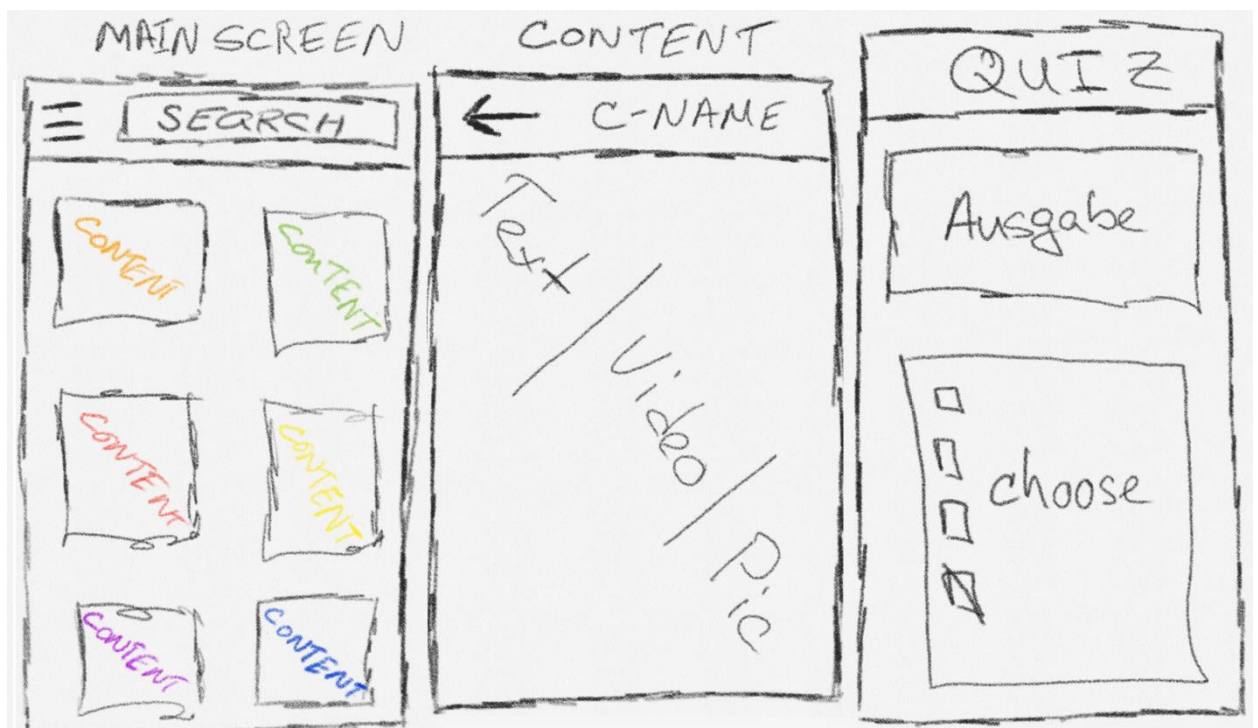


Abbildung 4.1: Mockup der App Teach Me.

4.3 Prototype

Prototypen basieren auf den Mockups. Da Prototypen lauffähig sind, kann damit schon ansatzweise die Funktionalität eines Systems dargestellt werden. Ein weiterer Vorteil gegenüber Mockups betrifft das Aussehen, denn dieses ist schon voll entwickelt. Dadurch bekommt man ein Gefühl für die spätere Anwendung und die Developer haben eine Vorlage zum Implementieren. In Adobe XD kann ein Prototyp bei der Entwicklung in DIE Android Studio importiert werden. Adobe XD steht für Power, Präzision und Qualität. Designer können mit XD interaktive Prototypen iterieren und zur Prüfung auf gängigen Geräten und Plattformen wie Windows, MacOS, iOS und Android freigeben.

In die Abbildung 4.2 sind Prototype des Hauptbildschirms und Quizfragen der App Teach Me zu sehen.

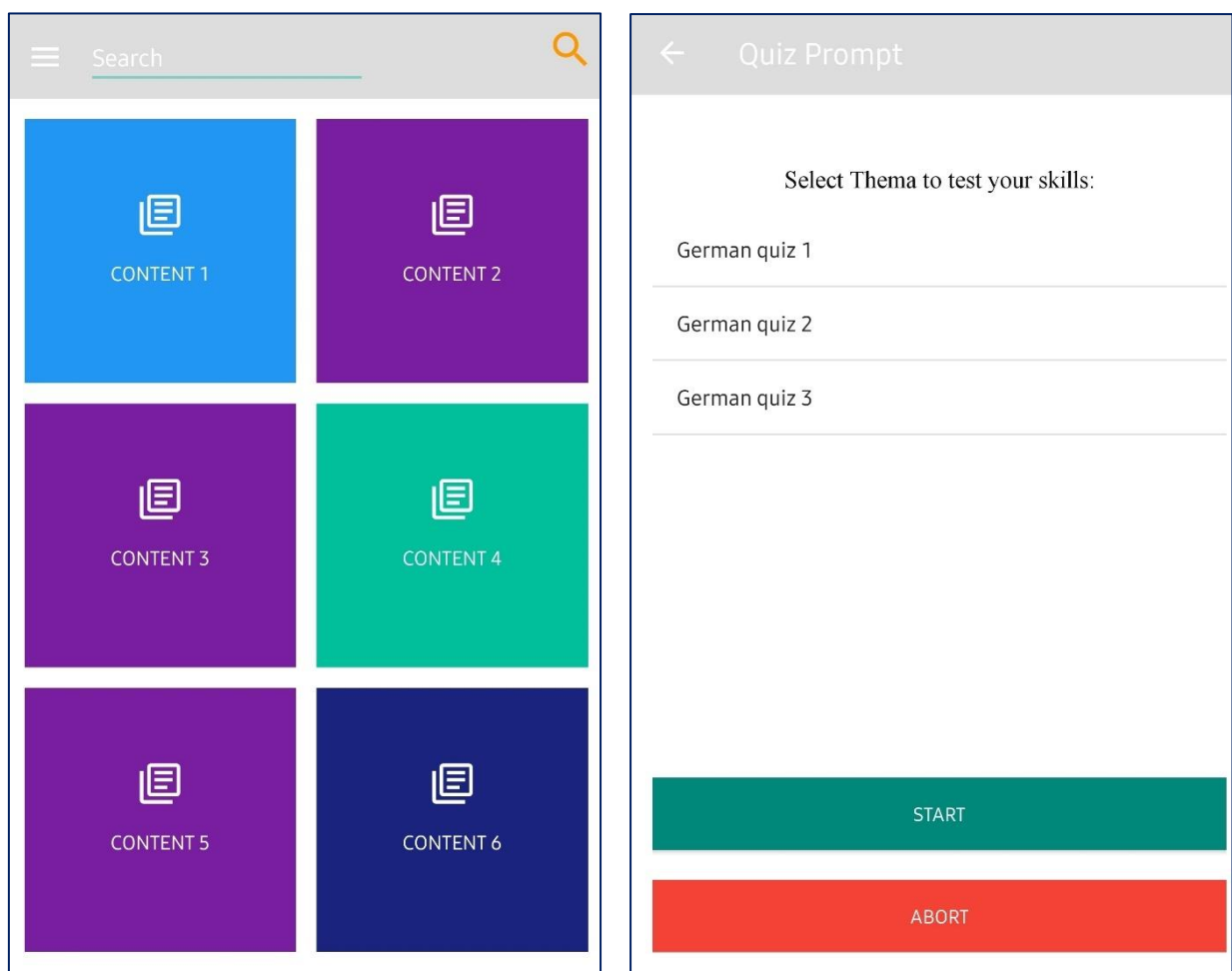


Abbildung 4.2: Prototype des Hauptbildschirm (rechts) und Quizfragen (links) der App Teach Me.

5. Implementierung

Die App Teach Me wurde unter den funktionalen, optionalen nicht-funktionalen Anforderungen, von der Mockup, Prototype und in DIE Android Studio 3.4.1 und AIDE mit Java, JSON, HTML Sprachen zu entwickelt. In der Kapitel 5 wird beschrieben, wie der Inhalt und den Datenformat der App aufgebaut und wie der Inhalt mit den anderen App Komponenten in Verbindung steht. Zum Schluss werden die wichtigsten Benutzeroberflächen dargestellt.

5.1 Datenarchitektur

Alle Inhaltdaten und Quizfragen der App Teach Me werden in JSON gespeichert. JSON ist eine Abkürzung für JavaScript Objekt Notation — ein Datenübertragungsformat. JSON stammt aus JavaScript, aber es ist für die Verwendung in vielen anderen Sprachen verfügbar, einschließlich Python, Ruby, PHP und Java. Inhaltsdaten und Quizdaten werden mit Hilfe des Android Komponente WebView dargestellt. Das WebView ist eine Komponente von Android zur Darstellung von JSON, HTML oder Video Dateien [26].

5.1.1 Inhaltdaten

App Inhalten werden in JSON Objekt gespeichert. JSON Objekt besteht aus vielen Arrays. Arrays unterscheiden sich aus 3 Variablen:

- title: Beschreibung der Titel des Moduls.
- qType: Beschreibung der Inhaltstyp. Diese Variable wurde nur in Debug Modus verwendet.
- content: hier wird den Inhalt des Moduls in HTML Sprache geschrieben. Content unterscheidet sich auch in 3 weiteren Variablen:
 - tag_line: Beschreibung der Titel des Kapitels
 - qType: Beschreibung der Inhaltstyp. Diese Variable wurde nur in Debug Modus verwendet.
 - Details: Inhalt des Kapitels.

In die Abbildung 5.1 ist ein Teil Code aus der deutschen Version von JSON Inhaltdaten der App Teach Me zu sehen.

Codezeile 1-3 ist die JSON Objekt mit einem Array *items*. Ab Zeile 4 bis 6 ist das Modul Beschreibung. Die Zeilen von 7 bis 20 beschreiben den Inhalt des Kapitels. Ab die Zeile 21 beginnt neu Beschreibung eins anderen Moduls.

```

1: {
2:   "items": [
3:     {
4:       "title": "Objektorientiertes Programmieren in Java",
5:       "qType": "BIG_de_1",
6:       "content": [
7:         {
8:           "tag_line": "Objekte",
9:           "qType": "SMALL_de_1",
10:          "details": [
11:            "<h2>Objekte</h2>\n\n<p>Java ist eine objektorientierte Sprache. Das
12:              verlangt vom Entwickler neben dem Erlernen neuer Sprachelemente auch
13:              eine neue &laquo;objektorientierte&raquo; Denkweise. In Form eines
14:              Tutorials soll hier mit dieser Denkweise vertraut gemacht
15:              werden.</p>\n\n<p>Herkömmliche Softwareentwicklung bestand oftmals
16:              darin, zur Lösung eines vorgegebenen Problems Algorithmen zu
17:              entwerfen und diese in Prozeduren zu .....</p>\n"
18:          ]
19:        },
20:        {
21:          "tag_line": "Klassen",
22:          "qType": "SMALL_de_2",
23:          "details": [
24:            "<h2>Klassen</h2> .....</p>\n"
25:          ]
26:        },
27:      ]

```

Abbildung 5.1: Teil Code aus CS_GermanContentFile.json

5.1.2 Quizdaten

App Quizfragen werden in JSON Objekt, der nur aus einem Array *questionnaires* besteht.

JSON Array *questionnaires* besteht aus vielen Arrays. Arrays unterscheiden sich auch aus 3 Variablen:

- question: Beschreibung der Titel der Quizfrage
- answers: Beschreibung der möglichen Antworten
- correct_answer: Korrekte Antwort

Abbildung 5.2 zeigt ein Teil Code aus der Deutsche Quizfragen den Modul C++ an.

[illegible]

Abbildung 5.2: Teil Code aus QA_GERMAN_C++.json

5.1.3 Inholdparser

Für Inholdparser ist die Klasse ContentParserAdapter zuständig. Die Abbildung 5.3 zeigt ein Teilabschnittcode aus der oben genannten Klasse. Codezeilen 1-7 erzeugen Array aus jsonData. Ab die Codezeile 7 bis 34 wird eine for-schleife durch die Arraylänge durchgeführt. In der Schleife wird das Inhalt mit Hilfe die Schlüssel Wörter wie *JSON_KEY_TITLE*, *JSON_KEY_CONTENT*, *JSON_KEY_TAG_LINE* und *JSON_KEY_DETAILS* aus der Inholddaten abgelesen und in ein Array List *mContentList* hinzugefügt. Die Array List *mContentList* wird weiter als Content Provider in der App benutzt. In Anhang A.1 befindet sich der gesamte Quellcode der ContentParserAdapter.

```
1: public static void parseJson(String jsonData) {
2:     try {
3:         JSONObject jsonObjMain = new JSONObject(jsonData);
4:         JSONArray jsonArray1 = jsonObjMain.
5:             getJSONArray(ContentConstant.JSON_KEY_ITEMS);
6:
7:         for (int i = 0; i < jsonArray1.length(); i++) {
8:             JSONObject jsonObj = jsonArray1.getJSONObject(i);
9:
10:            String title = jsonObj.getString
11:                (ContentConstant.JSON_KEY_TITLE);
12:
13:            ArrayList<Item> items = new ArrayList<>();
14:
15:            JSONArray jsonArray2 = jsonObj.
16:                getJSONArray(ContentConstant.JSON_KEY_CONTENT);
17:
18:            for (int j = 0; j < jsonArray2.length(); j++) {
19:                JSONObject jsonObj2 = jsonArray2.getJSONObject(j);
20:                String tag_line = jsonObj2.
21:                    getString(ContentConstant.JSON_KEY_TAG_LINE);
22:
23:                ArrayList<String> detailList = new ArrayList<>();
24:
25:                JSONArray jsonArray3 = jsonObj2.
26:                    getJSONArray(ContentConstant.JSON_KEY_DETAILS);
27:
28:                for (int k = 0; k < jsonArray3.length(); k++) {
29:                    String details = jsonArray3.get(k).toString();
30:                    detailList.add(details);
31:                }
32:                items.add(new Item(tag_line, detailList));
33:            }
34:            mContentList.add(new Contents(title, items));
35:        }
36:    } catch (JSONException e) {
37:        e.printStackTrace();
38:    }
39:    hideLoader();
40:    mAdapter.notifyDataSetChanged();
41: }
```

Abbildung 5.3: Funktion „parseJson“ aus ContentParserAdapter.java.

5.1.4 Quizparser

Für Quizparser ist die Klasse QuizParserAdapter zuständig. Die Abbildung 5.4 liefert ein Teilabschnittcode aus der oben genannten Klasse. Codezeilen 1-5 erzeugen ein Array. Ab die Codezeile 6 bis 30 wird eine for-schleife durch die Arraylänge durchgeführt. In der Schleife wird mit Hilfe die Schlüsselwörter wie *JSON_KEY_QUESTIONNAIRY*, *JSON_KEY_QUESTION* *JSON_KEY_CORRECT_ANS* und *JSON_KEY_ANSWERS* Quizinhalt aus der Quizdaten abgelesen. Danach wird der Quizinhalt zu das Quizmodel eingefügt und auf die Quizfragenbildschirm angezeigt. In Anhang A.2 befindet sich der gesamte Quellcode der QuizParserAdapter.

```
1: public void parseJson() {
2:     try {
3:         JSONObject jsonObjMain = new JSONObject(jsonData);
4:         JSONArray jsonArray = jsonObjMain.getJSONArray
5:             (ContentConstant.JSON_KEY_QUESTIONNAIRY);
6:         for (int i = 0; i < jsonArray.length(); i++) {
7:             JSONObject jsonObj = jsonArray.getJSONObject(i);
8:
9:             String question = jsonObj.getString(
10:                 ContentConstant.JSON_KEY_QUESTION);
11:
12:             int correctAnswer = Integer.parseInt
13:                 (jsonObj.getString
14:                     (ContentConstant.JSON_KEY_CORRECT_ANS));
15:
16:             JSONArray jsonArray2 = jsonObj.getJSONArray
17:                 (ContentConstant.JSON_KEY_ANSWERS);
18:
19:             ArrayList<String> contents = new ArrayList<>();
20:             ArrayList<String> backgroundColors = new ArrayList<>();
21:
22:             for (int j = 0; j < jsonArray2.length(); j++) {
23:                 String item_title = jsonArray2.get(j).toString();
24:                 contents.add(item_title);
25:                 backgroundColors.add(AppConstant.COLOR_WHITE);
26:             }
27:             QuestionActivity.mItemList.add(new QuizModel
28:                 (question, contents, correctAnswer, backgroundColors));
29:             Collections.shuffle(QuestionActivity.mItemList);
30:         }
31:         hideLoader();
32:         updateQuestionsAndAnswers();
33:
34:     } catch (JSONException e) {
35:         e.printStackTrace();
36:     }
37: }
```

Abbildung 5.4: Funktion „parseJson“ aus QuizParserAdapter.java.

5.2 Benutzeroberflächen

Im Kapitel 5.2 werden die wichtigsten Benutzeroberflächen der App Teach Me beschrieben.

5.2.1 Hauptbildschirm

Der Hauptbildschirm, das ist das erste das der Netzer nach dem Begrüßungsbildschirm sieht. Auf dem Hauptbildschirm kann der Nutzer verschiedene Lernmodule auswählen oder sich zu Quizfragen, Interviewfragen oder einfach zur Suche sich navigieren. Der Nutzer kann auch den Studiengang und die Sprache wechseln. Die Abbildung 5.5 zeigt das Interface des Hauptbildschirms und ein geöffnetes Modul: Grundlagen der Betriebssysteme.

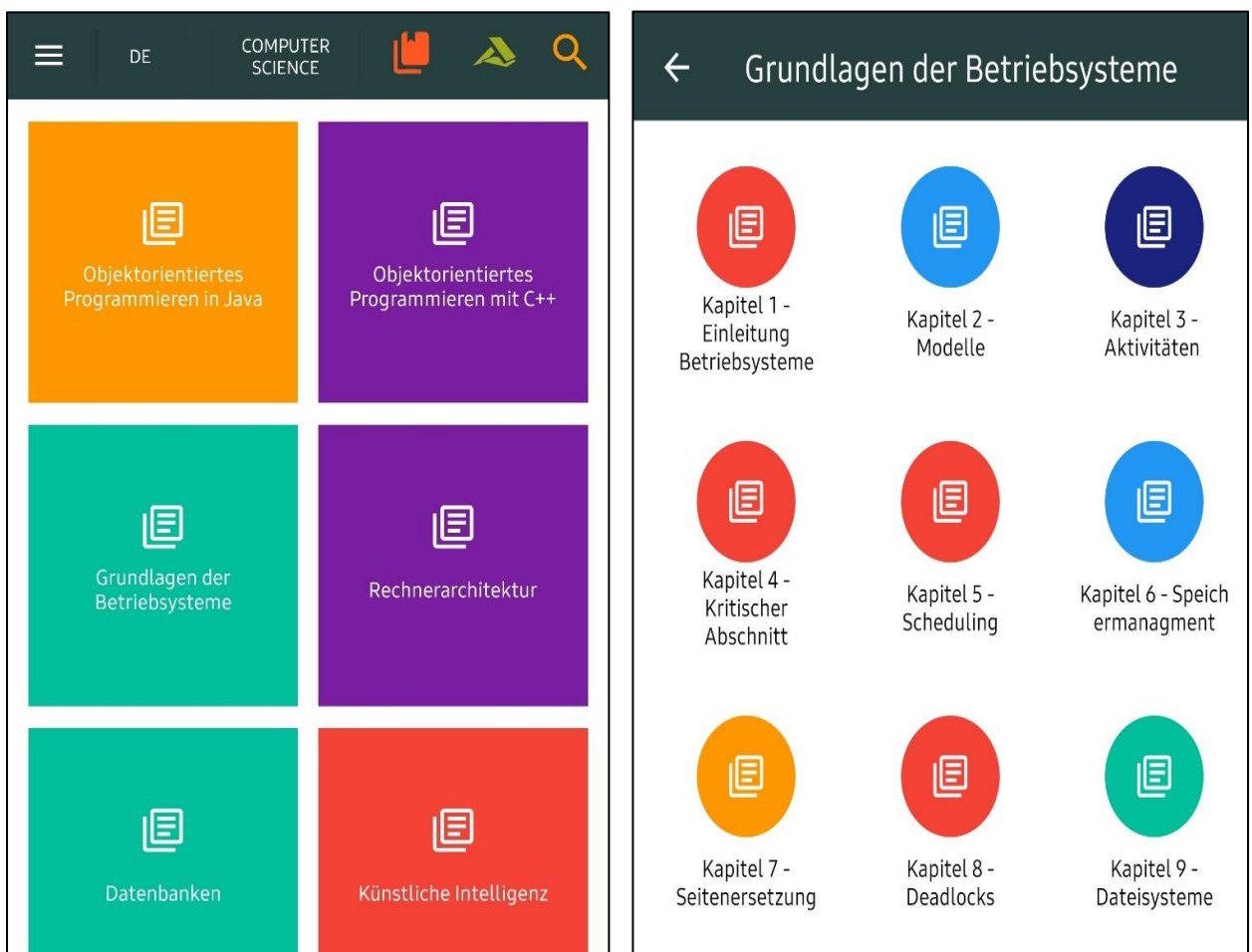


Abbildung 5.5: Hauptbildschirm (links) und geöffnete Modul: Grundlagen der Betriebssysteme (rechts).

Durch die Navigation in den Modulen kann der Nutzer Kapitel auswählen. Im Kapitelbereich kann der Nutzer Kapitel favorisieren, Inhalt des Kapitels kopieren oder teilen und sich das Kapitel vorlesen lassen. Auch kann der Nutzer sich zurück zu dem gewählten Modul navigieren. Die Abbildung 5.6 zeigt das Interface Kapitel 4 – Kritischer Abschnitt aus des Moduls Grundlagen der Betriebssysteme.

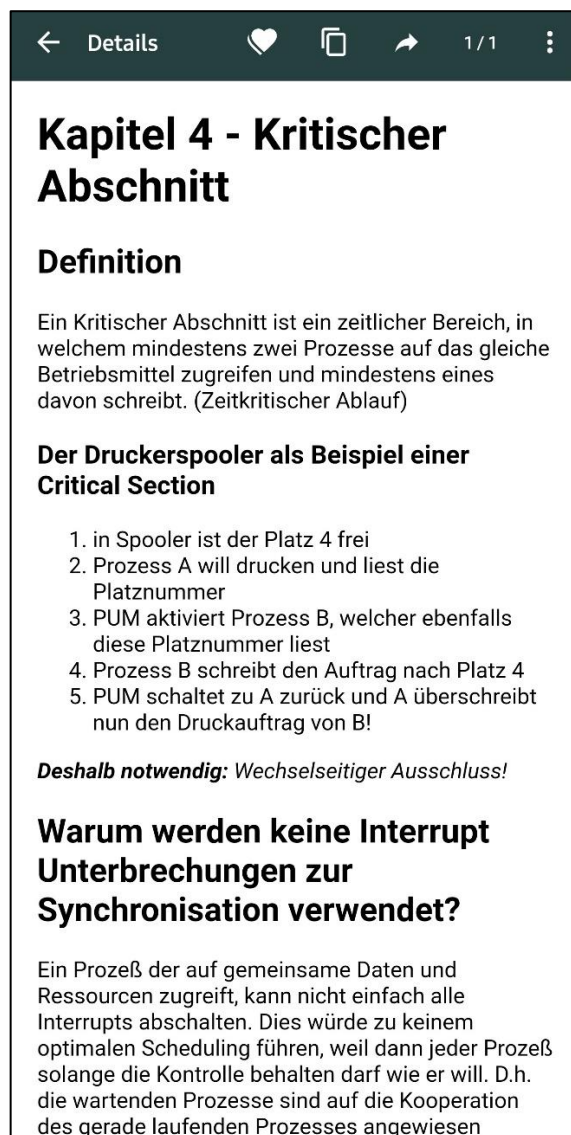


Abbildung 5.6: Kapitel 4 – Kritischer Abschnitt aus des Moduls Grundlagen der Betriebssysteme.

5.2.2 Suche

Durch die Navigation des Hauptbildschirms kann der Nutzer die Suchfunktion der App nutzen. Die Suche erfolgt nach einem Begriffswort und liefert alle Ergebnisse aus den Modulen beziehungsweise Kapiteln. Abbildung 5. 7 liefert eine Darstellung der Suchergebnisse nach einem Begriffswort shell.

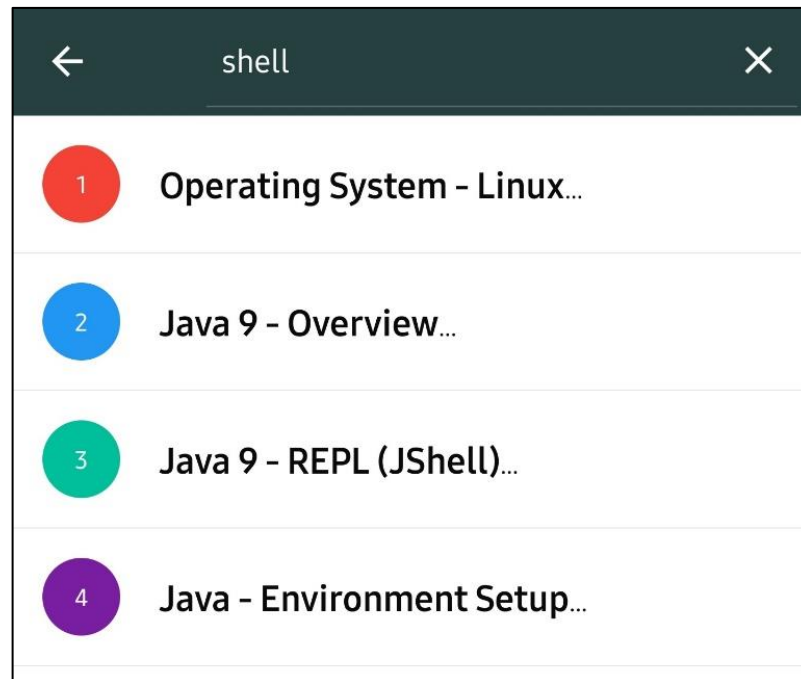


Abbildung 5.7: Begriffssuche in der App am Beispiel Suchwort: shell

5.2.3 Quizfragen

Navigation durch die Quizfragen erfolgt durch den Hauptbildschirm. Die Quizfragen sind in einer Art von Multiple Choice dargestellt. Der Nutzer hat 5 Versuche eine Frage zu beantworten oder einfach die Frage zu überspringen. Richtige Antworten werden mit grüner Farbe und dazugehörigen Ton unterstrichen, während die falschen Antworten werden mit rot markiert und einem dazugehörigen Ton begleitet. Nach Beendigung der Quizfragen ist es möglich die Ergebnisse sich anzuschauen. Dort kann der Nutzer die Statistik seines Quiz ansehen, indem er Antworten auf seine Fragen erhält. Der Nutzer kann die Ergebnisse teilen oder das Quiz noch einmal wiederholen. Die Abbildung 5.8 zeigt das Interface des Quizbildschirms und der Quizergebnisse.

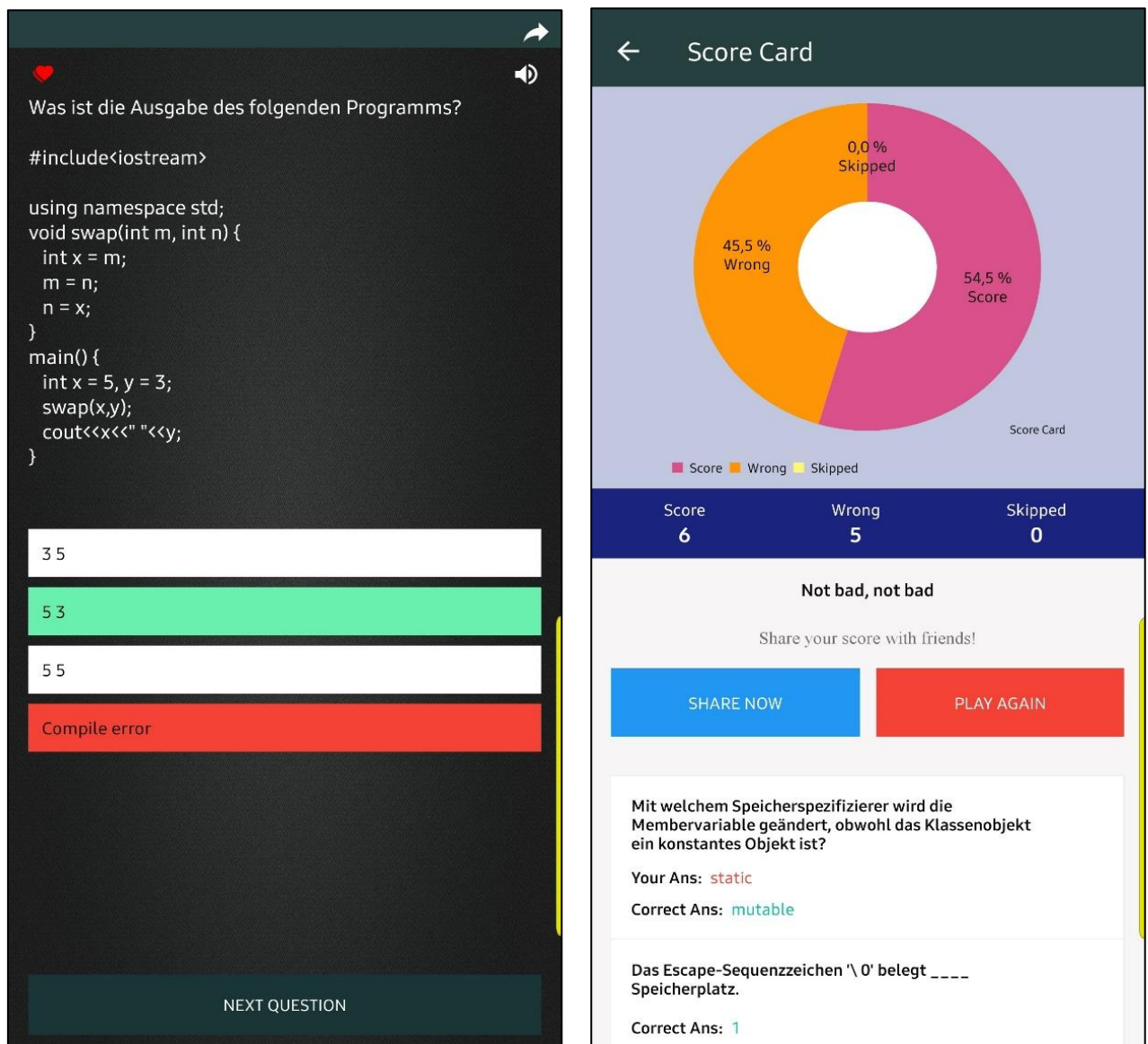


Abbildung 5.8: Quizfragenbildschirm(links) und Quizergebnisbildschirm(rechts).

5.2.4 Interviewfragen

In der App gibt es parallel zu den Quizfragen auch Interviewfragen. Der Nutzer kann zu den Interviewfragen einfach aus dem Hauptbildschirm sich navigieren, Dort kann der Nutzer sich das Thema auswählen und die dazugehörigen Fragen aussuchen, um eine Antwort zu bekommen. Die Abbildung 5.9 stellt das Interface der Interviewfragen dar.

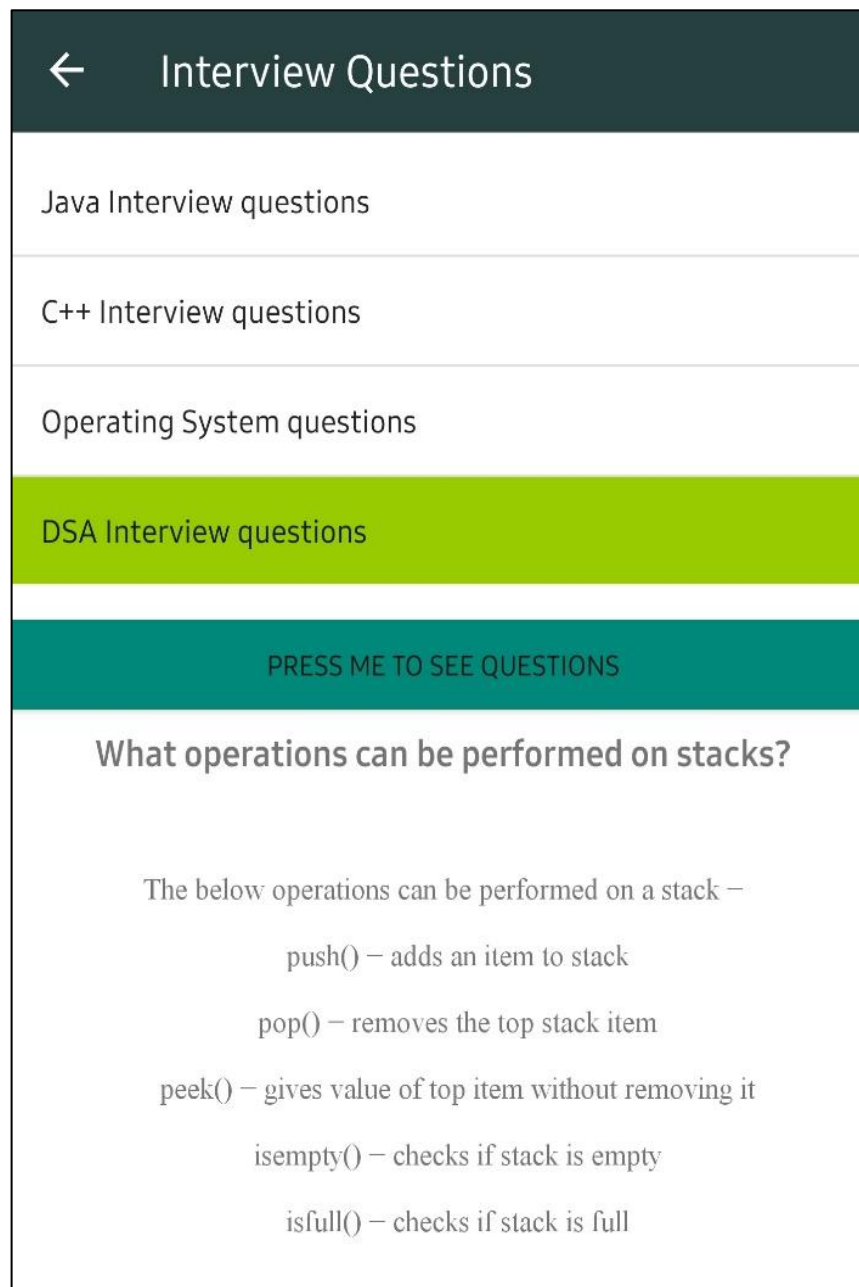


Abbildung 5.9: Interviewfragenbildschirm

5.2.5 Einstellungen

In den Einstellungen kann der Nutzer die Textgröße und Sprache des Inhalts ändern. Es ist auch möglich die Benachrichtigungen ein und auszuschalten. In den Einstellungen kann der Nutzer auch die Vollbildoption nutzen. Die Abbildung 5.10 stellt den Bildschirm der Einstellungen dar.

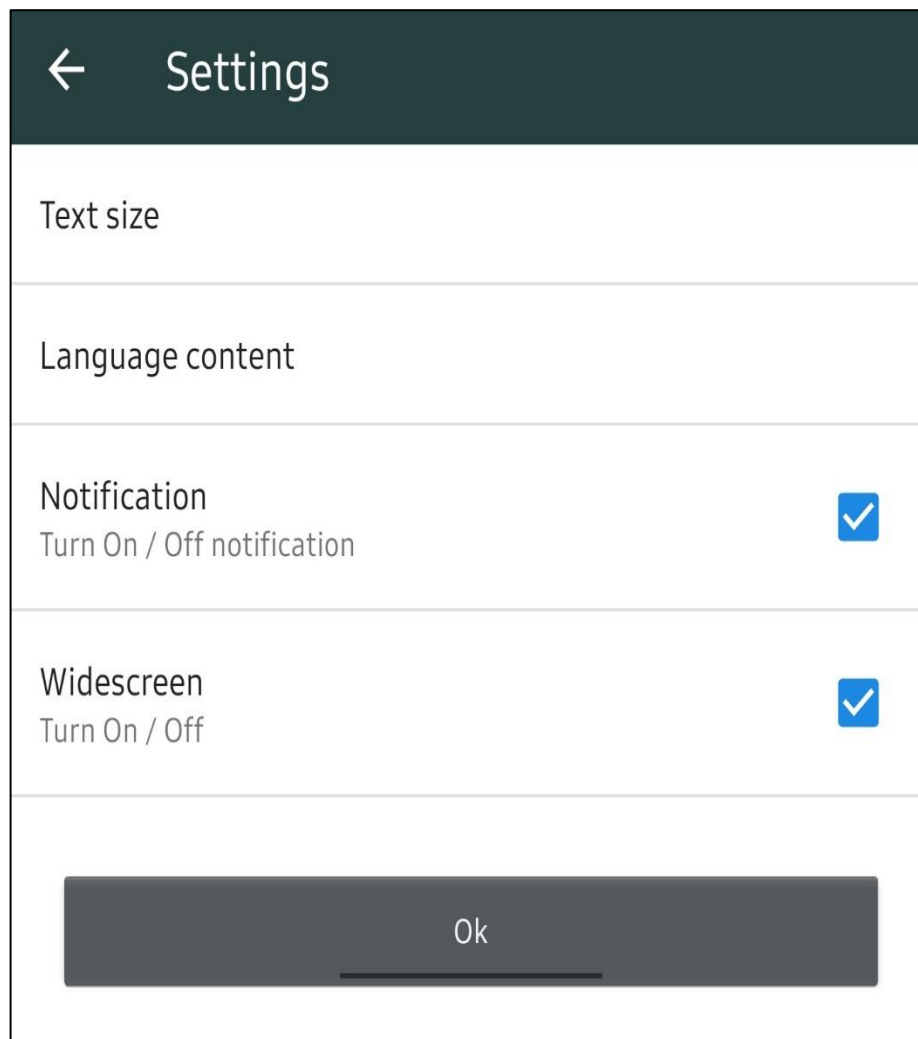


Abbildung 5.10: Einstellungen

5.2.6 Favoriten

Die Kapitel, die der Nutzer sich anschaut, kann er auch favorisieren.

In den Favoriten kann der Nutzer seine Kapitel ansehen. Von dort kann der Nutzer die Kapitel auch wieder löschen. Die Abbildung 5.11 liefert das Interface der Favoriten.

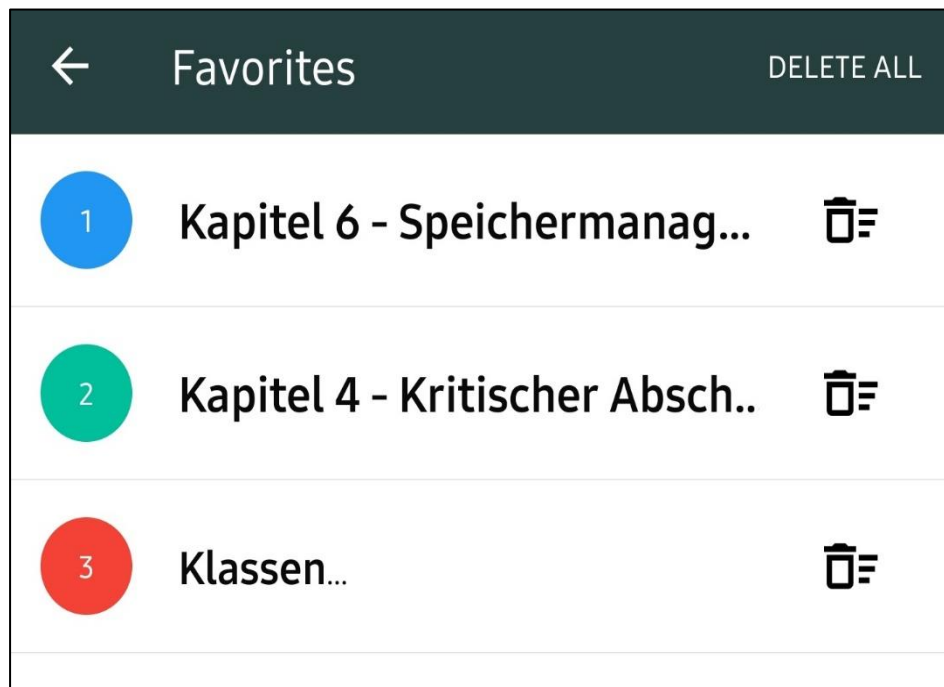


Abbildung 5.11: Favoriten

5.3 Kapitelfazit

Am Anfang des Kapitels wird die Datenstruktur definiert und beschrieben. Im folgenden kommt die Beschreibung vom Inthaltparser und Quizparser. Zum Schluss des Kapitels werden die wichtigsten Oberflächen der App dargestellt.

Bei Implementierung der App könnte die Forschungsfrage: Ist Programmiersprache Java immer noch gut für Android Entwicklung anzuwenden? Beantwortet werden. Java bietet immer noch viele Möglichkeiten, die App zu entwickeln. Aber nur mit Java ist es nicht möglich Android Anwendungen zu entwickeln. Android Oberflächen werden in XML Sprache geschrieben und für das Datenübertragungsformat wird JSON oder SQLite benutzt.

6. Anwendungsvergleich

Zuerst werden im Kapitel 6. Die Anforderungen, die im Kapitel 4. 1 gestellt wurden, abgeglichen. Nach dem Anforderungsvergleich wird die Struktur des Fragebogens beschrieben, die von den Studierenden der Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms beantwortet wurden. Im Kapitel 6.6 werden die Fragebögen mit Hilfe von Diagrammen ausgewertet. Zum Schluss kommt das Kapitelfazit, wo die Forschungsergebnisse stehen.

Um den Anwendungsvergleich durchzuführen, werden folgende Android-Geräte benutzt:

- Samsung Galaxy Note 8
- Samsung Galaxy S7
- Lenovo Tab 10
- Huawei P

6.1 Anforderungen Abgleich

In diesem Kapitel werden die in Kapitel 4.1 genannten funktionalen, optionale und nicht-funktionalen Anforderungen abgeglichen.

6.1.1 Abgleich der funktionalen Anforderungen

Der App Teach Me erfüllt alle funktionalen Anforderungen, die in Kapitel 4.1 gestellt wurden. Die Tabellen 5 und 6 zeigen der Abgleich der funktionalen Anforderungen.

Nº	Aufgabe	Erfüllt	Beschreibung
F.1	Lernmodul	Ja	Der Nutzer kann den Modulen auswählen. Siehe Kapitel 5.2.1.
F.2	Suche	Ja	Der Nutzer kann nach ein Begriffswort ins Modulen suchen. Siehe Kapitel 5.2.2.
F.3	Quizfragen	Ja	Der Nutzer kann die Quizfragen teilnehmen nähmen. Siehe Kapitel 5.2.3
F.4	Interviewfragen	Ja	Der Nutzer kann sich mit den Interviewfragen abfragen. Siehe Kapitel 5.2.4

Tabelle 5: Abgleich den funktionalen Anforderungen von F.1 bis F.4.

Nº	Aufgabe	Erfüllt	Beschreibung
F.5	Video	Ja	Durch die Android WebView kann Nutzer Video Module / Inhalte ansehen. Siehe Kapitel 5.1
F.6	Textgröße	Ja	Der Nutzer kann die Textgröße durch Einstellungen in der App zu wechseln. Siehe Kapitel 5.2.5
F.7	Sprachauswahl	Ja	Der Nutzer kann die Sprache durch Einstellungen oder durch Schnellleiste in Hauptbildschirm zu wechseln. Siehe Kapitel 5.2.5
F.8	Favoriten	Ja	Der Nutzer kann Modulen favorisieren. Siehe Kapitel 5.2.6 Favoriten
F.9	Textvorlesen	Ja	Der Nutzer kann das Kapitel sich vorlesen lassen. Siehe Kapitel 5.2.1
F.10	Inhaltskopieren	Ja	Der Nutzer kann Kapitels Inhalt in Buffer kopieren. Siehe Kapitel 5.2.1
F.11	Einstellungen	Ja	Der Nutzer kann Benachteiligungen an- und auszuschalten.

Tabelle 6: Abgleich den funktionalen Anforderungen von F.5 bis F.11.

6.1.2 Abgleich der optimalen Anforderungen

Teach Me erfüllt alle optionalen Anforderungen, die in Kapitel 4.1 gestellt wurden. Die Tabelle 7 der Abgleich der optionalen Anforderungen.

Nº	Aufgabe	Erfüllt	Beschreibung
0.1	Teilen	Ja	Der Nutzer kann Quizfragen und Kapiteln teilen. Siehe Kapitel 5.2.1 und Kapitel 5.2.3.
0.2	Breitbild	Ja	Der Nutzer kann Breitbild in den Einstellungen einzuschalten. Siehe Kapitel 5.2.5.
0.3	Ton	Ja	Die Quizfragen werden mit dem Ton begleiten. Siehe Kapitel 5.2.3.
0.4	Quizergebnisse	Ja	Der Nutzer kann richtige und falsche Antworten anzusehen. Siehe Kapitel 5.2.3.
0.5	Benachrichtigung	Ja	Der Nutzer kann Benachrichtigungen von Entwickler bekommen.

Tabelle 7: Abgleich den optionalen Anforderungen.

6.1.3 Abgleich der nicht-funktionalen Anforderungen

In der App werden alle nicht-funktionalen Anforderungen, die in Kapitel 4.1 gestellt wurden, erfüllt. Die Tabelle 8 der Abgleich der optionalen Anforderungen.

Nº	Aufgabe	Erfüllt	Beschreibung
NF.1	Einheitliche Darstellungsformen der Quizfragen	Ja	Die Darstellung der Quizfragen ist in einfache Antwort-Wahl-Verfahren dargestellt.
NF.2	Usability Ziele einhalten	Ja	Durch die Hierarchie ist die App einfach und verständlich aufgebaut.
NF.3	Verfügbarkeit	Ja	Diese App könnte auch in einem Tablet geöffnet werden. Es würden sich allen Elementen skalieren.
NF.4	Selbsterklärbarkeit	Ja	Es wurden Standard-Icons des Material Design verwendet, um die Selbsterklärbarkeit zu steigern.
NF.5	Aktuelle Version der Betriebssysteme	Ja	Die App unterstützt die aktuellen Versionen der Android Betriebssysteme (Android 5-10).

Tabelle 8: Abgleich den nicht-funktionale Anforderungen.

6.2 Fragebogen

Die Studierenden werden zuerst gebeten ihre Erfahrung mit dem Umgang von Android Smartphones anzugeben mit einer Skala von Ungenügend bis Sehr gut. Danach werden die Studierenden gefragt, ob sie schon ähnliche Apps benutzen. Als nächstes kommen 7 Aufgaben, die sich mit der Programmoberfläche, dem Inhalt, der Programmsuche, dem Quiz, den Fragen und Antworten, den Einstellungen und dem Studiengang in der App Teach Me beschäftigen. Die Befragten sollten auch den Schwierigkeitsgrad der Aufgaben auf einer Skala von 1 bis 10, wobei 10 sehr schwierig und 1 sehr leicht darstellt, bewerten. Nach den Aufgaben sollten die Studierenden mit Ja oder Nein drei weitere Fragen beantworten. Bei den Fragen ging es darum, ob die App einwandfrei nutzbar sei, ob sie genug Lernstoff beinhaltet und zu allerletzt, ob sie einen echten Tutor ersetzen könne. Am Ende des Fragebogens konnten die Befragten noch ein Feedback hinterlassen.

Im Anhang B1 befindet sich das Muster des Fragebogens.

6.3 Forschungsversuch an der Ruprecht-Karls-Universität Heidelberg

Der Forschungsversuch an der Ruprecht-Karls-Universität Heidelberg hat im Juni 2019, an den Tagen 3., 7., 18., 26. stattgefunden. Insgesamt wurden 57 Informatik Studenten befragt. Die Befragung fand am Campus Im Neuenheimer Feld im Mathematikon Gebäude, INF 227, INF 306 und die Zentralmensa vom Neuenheimer Feld statt. Viele der Befragten hinterließen kein Feedback. Nur wenige sind gut und deutlich formuliert. Ein folgendes Feedback hinterließ der 20-jährige Angewandte Informatik Student:

Die App lässt sich einwandfrei benutzen, aber es fehlt ein wenig Lernstoff [26].

Ein anderes Feedback von der 22-jährigen Studentin aus Dual Angewandten Informatik:

Die App ist einfach gebaut und gut strukturiert. Zudem besitzt sie auch ein gutes Quizmodell, das für die Klausuren relevant ist. Aber um einen Tutor vollständig ersetzen zu können, muss die App noch mehr bieten [27].

Nach diesem Forschungsversuch wurden weitere Lernmodule hinzugefügt.

6.4 Forschungsversuch an der Johannes-Gutenberg-Universität Mainz

Der Forschungsversuch an der Johannes-Gutenberg-Universität Mainz fand im Juli, den 3., 12., 18. und 23 statt. Es haben 52 Informatik Studenten am Campus an der Umfrage teilgenommen. Genau wie an der Universität Heidelberg hinterließen nicht viele Studenten ein Feedback. Meistens waren sie kurz und undeutlich geschrieben, nur zwei waren deutlich ausformuliert. Ein Student des 4. Semesters des Informatik Studiengangs hat folgendes geschrieben:

Die App hat verschiedene Studiengänge und genug Lernstoff für die ersten Semester. Aber die App muss noch für die anderen Plattformen offen zugänglich sein. Und es muss noch spezieller Lernstoff hinzugefügt werden. [28].

Auch ein Student des 1. Semesters des Informatik Studiengangs hinterlässt folgende Feedback:

Einfaches Design, gutes Quiz und die App lässt sich gut bedienen [29].

6.5 Forschungsversuch an der Hochschule Worms

Der Forschungsversuch an der Hochschule Worms, hat am 6. Und 14. August stattgefunden. Es wurden insgesamt 22 Studenten aus Informatik und Mobile Computing befragt. Es wurde nur ein Feedback hinterlassen:

Die App bietet viele Module, die im Studium auch relevant sind und mit dem Quiz kann man sich selbst prüfen, ansonsten ist die App einfach gebaut und lässt sich gut bedienen.[30].

6.6 Fragebogenauswertung

Beim Anwendungsvergleich wurden zur Evaluation der App Befragungen mit 131 Studenten durchgeführt. Dabei wurde die Funktionsweise der App demonstriert und den Studenten Zeit gegeben die App auszuprobieren. Danach haben die Befragten die Fragebögen ausgefüllt und wurden nach ihrem allgemeinen Feedback befragt.

Die Studierenden haben zur Auswahl 4 mobile Android Geräte gehabt. Am meisten beim Anwendungsvergleich wurde das Samsung Galaxy Note 8 gewählt. Das Diagramm 1 bietet eine Statistik für Nutzung der Geräte bei Anwendungsvergleich an.

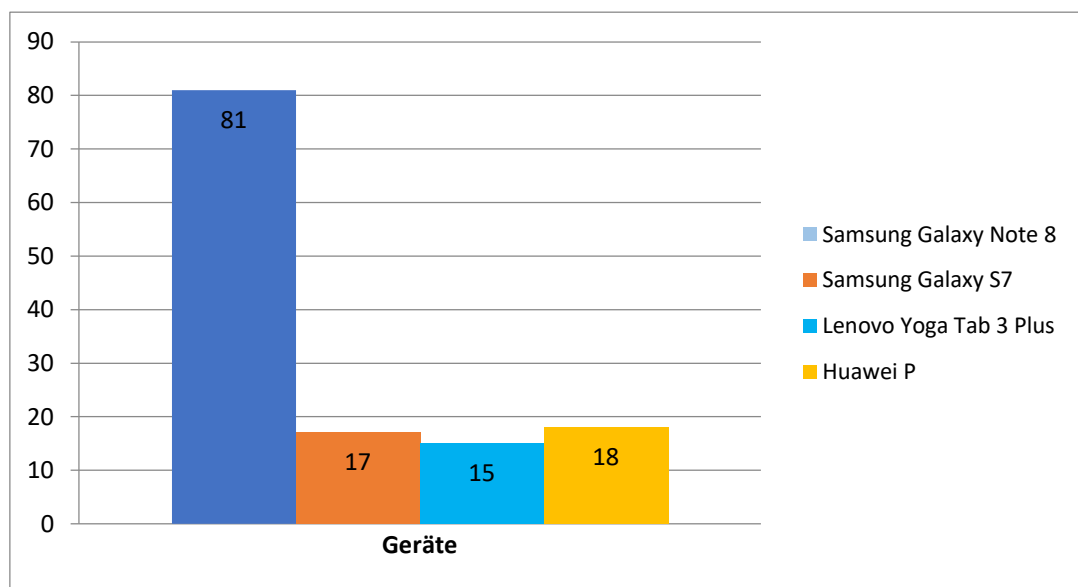


Diagramm 1: Nutzung der Geräte bei Anwendungsvergleich.

Als Zweites wurden die Studenten gebeten ihre Erfahrung mit dem Umgang von Android Smartphone anzugeben. Auf dem Diagramm 2 ist zu sehen, dass dreiundvierzig Studenten einen sehr guten Umgang mit Android Geräten haben. Davon haben 56 gut ausgefüllt und 26 befriedigt angegeben. Nur 6 Studenten haben beim Umgang mit Android Smartphones ausreichend angegeben.

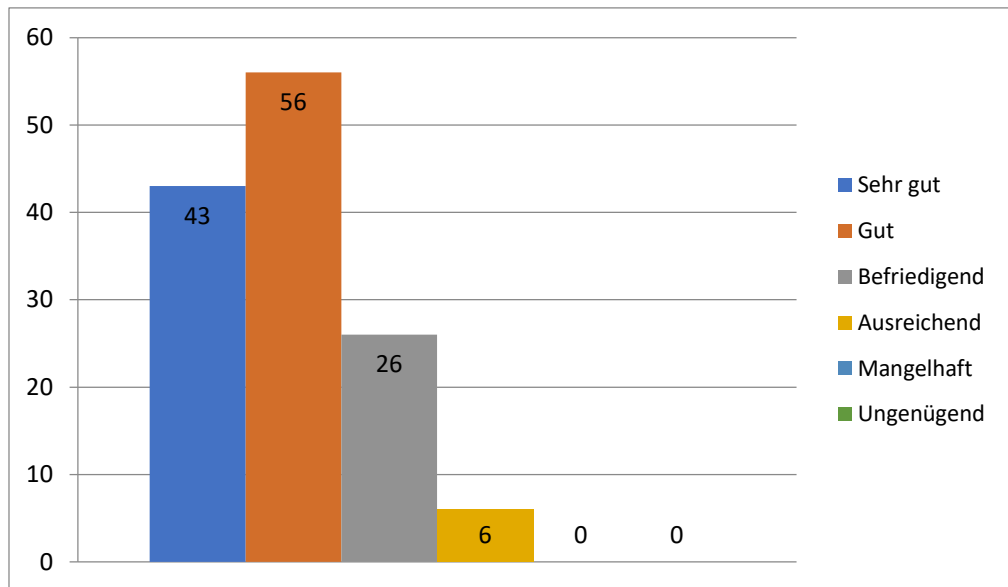


Diagramm 2: Erfahrung mit dem Umgang von Android Smartphones.

Danach mussten die Befragten antworten, ob sie ähnliche Apps auf ihrem Smartphone bereits nutzen. Hundertacht der Befragten haben diese Frage verneint, während dreiundzwanzig der Befragten bejahten. Das Diagramm 3 zeigt die Statistik von der Nutzung von ähnlichen Apps.

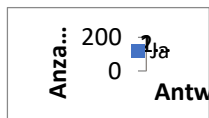


Diagramm 3: Nutzung von ähnlichen Apps.

Als erste Frage zu der App Teach Me sollten die Studenten sich mit der Programm Oberfläche vertraut machen. Das Diagramm 4 zeigt den Schwierigkeitsgrad der Aufgabe. Dreiundsiebzig Studenten kreuzten an, dass die Aufgabe sehr leicht sei. Fünfundzwanzig Befragte sagten, dass sie kleine Probleme mit der Aufgabe gehabt haben. Zweiunddreißig Studenten haben den Schwierigkeitsgrad im Bereich von 3 bis 6 angekreuzt. Nur eine Person fand die Aufgabe sehr schwierig.

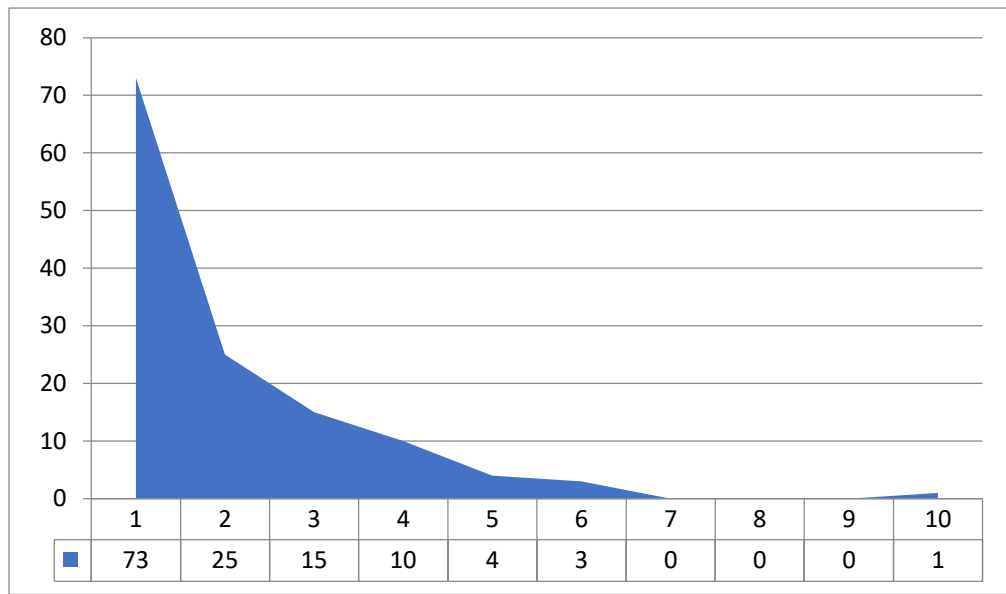


Diagramm 4: Ergebnisse zu Frage 1 – Programmoberfläche.

Als zweite Frage sollten die Befragten sich mit dem App Inhalt vertraut machen. Sechsendsechzig Studenten kreuzten an, dass die Aufgabe sehr leicht war. Vierundsechzig wählten den Schwierigkeitsgrad im Bereich 2 bis 5. Ein Student hat die Aufgabe mit Schwierigkeitsgrad 8 bewältigt. Das Diagramm 5 liefert die Ergebnisse zu Frage 2.

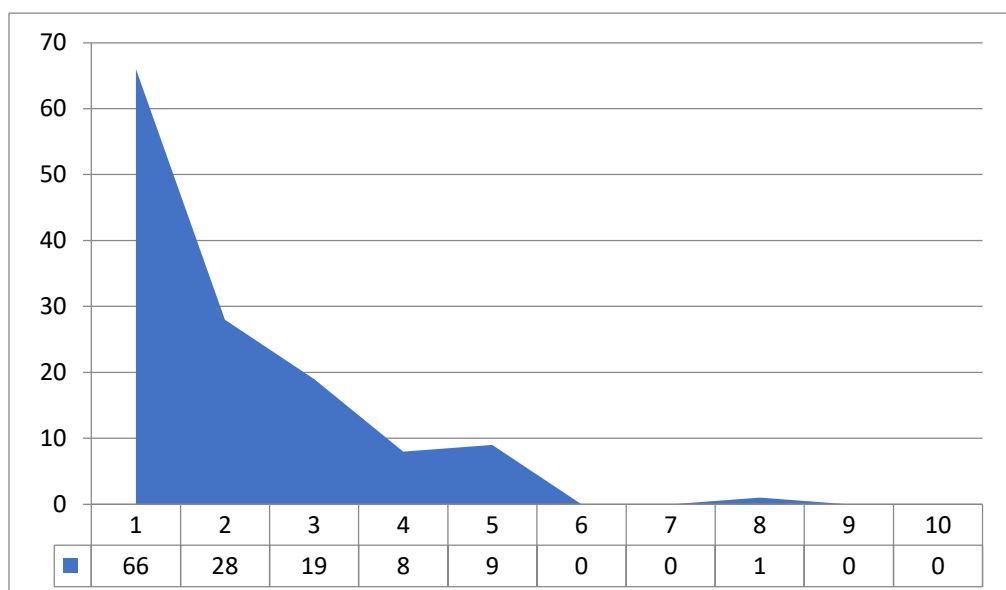


Diagramm 5: Ergebnisse zu Frage 2 – Inhalt.

Bei der dritten Frage sollten die Studenten sich mit der Programmsuche beschäftigen. Wie auf Diagramm 6 zu sehen ist, achtundfünfzig Befragten finden diese Aufgabe sehr leicht. Dreiundsiebzig Studenten hatten kleine Schwierigkeiten bei der Aufgabe gehabt und wählten die Werte im Bereich von 2 bis 5.

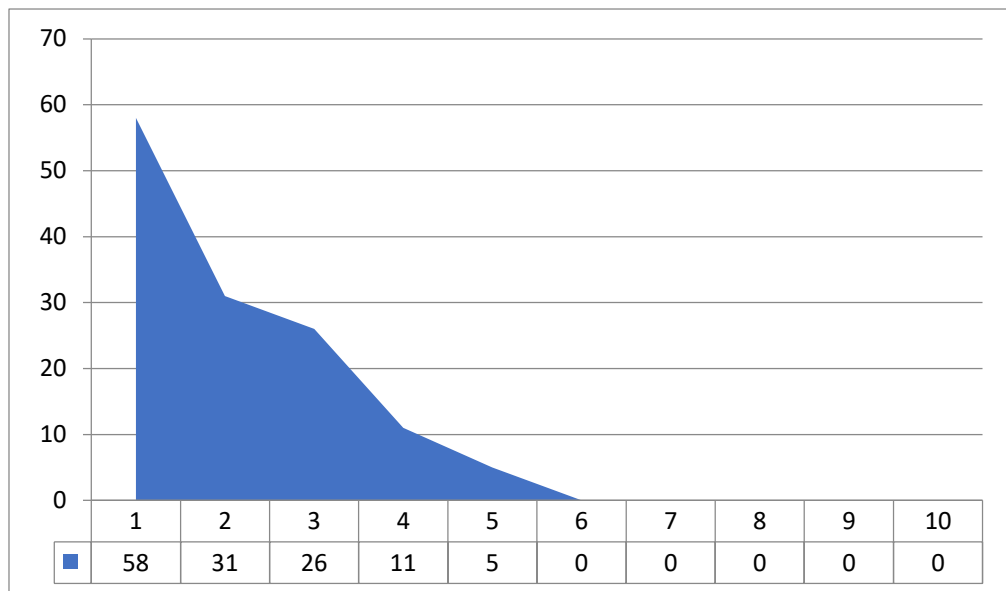


Diagramm 6: Ergebnisse zu Frage 3 – Programmsuche.

Sich vertraut zu machen mit der Quizoberfläche war die 4. Aufgabe. Nur dreiunddreißig Studenten fanden die Aufgabe leicht. Aber die meisten haben die Aufgabe mit Werten im Bereich von 2 bis 5 angekreuzt. Nur 3 Studenten finden die Aufgabe ein wenig schwierig. Das Diagramm 7 liefert die Ergebnisse dafür.

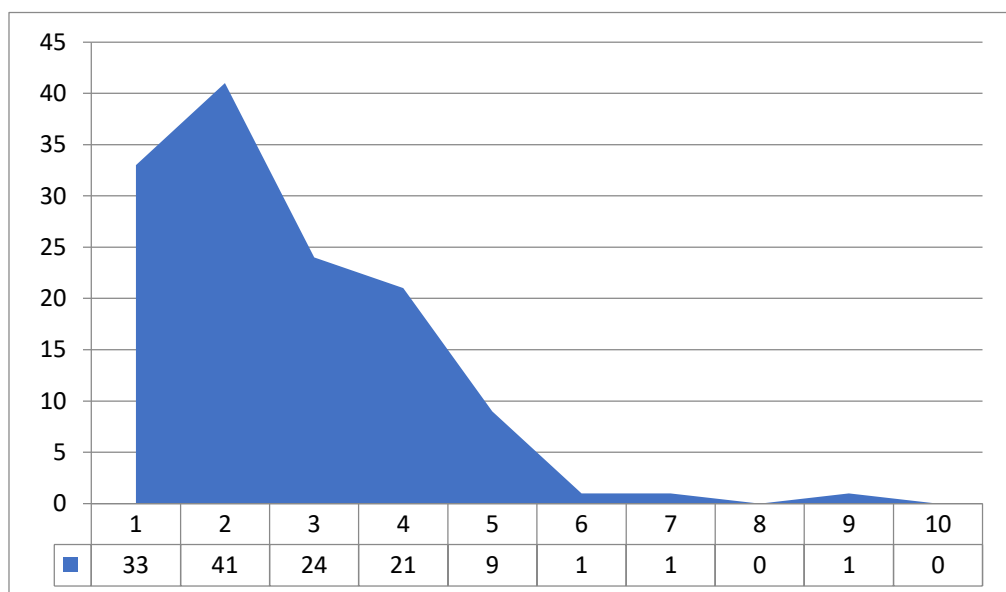


Diagramm 7: Ergebnisse zu Frage 4 – Quiz.

In Folge sollten sich die Studenten mit der Oberfläche der Fragen vertraut machen. Für neununddreißig Personen war die Aufgabe sehr leicht. 25 der Befragten gaben an, dass sie minimale Probleme mit der Aufgabe hatten. Der Mittelbereich der Werte haben 63 Personen angegeben. Für eine Person war die Aufgabe fast unmöglich. Und die restlichen 3 Befragten haben mit Mühe geschafft die Aufgabe zu bewältigen. Das Diagramm 8 zeigt den Gesamteindruck zu der Frage 5 – Fragen und Antworten.

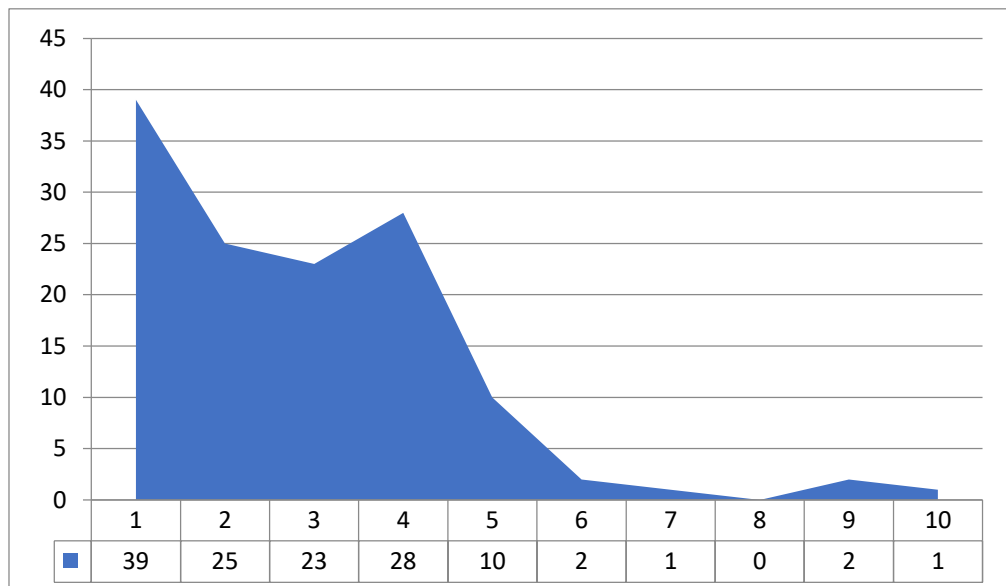


Diagramm 8: Ergebnisse zu Frage 5 – Fragen und Antworten.

Als sechstes sollten die Befragten sich mit den Einstellungen vertraut machen. Für viele der Befragten(41) war die Aufgabe sehr einfach, 69 Personen hatten kleine Schwierigkeiten. Den Mittelbereich haben 17 Personen angekreuzt und die letzten 4 Personen hatten große Schwierigkeiten gehabt. Das Diagramm 9 bietet die Ergebnisse zu Frage 6 - Einstellungen.

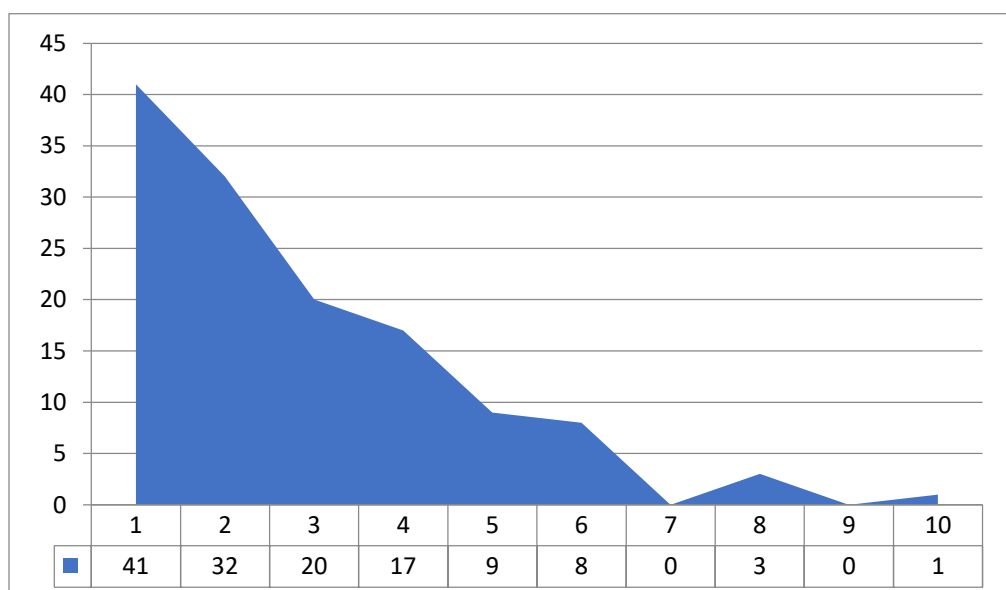


Diagramm 9: Ergebnisse zu Frage 6 – Einstellungen.

Die letzte Frage an die App sollten sich die Befragten mit dem Studiengang vertraut machen. Wie das Diagramm 10 zeigt, 46 Personen kreuzten an, dass die Aufgabe ohne Probleme zu lösen war. Die Werte im Bereich von 2 bis 4 wurden von 74 Studenten befüllt, den Mittelbereich haben nur 7 Leute gewählt. Die restlichen 4 Personen hatten Probleme mit lösen der Aufgabe gehabt.

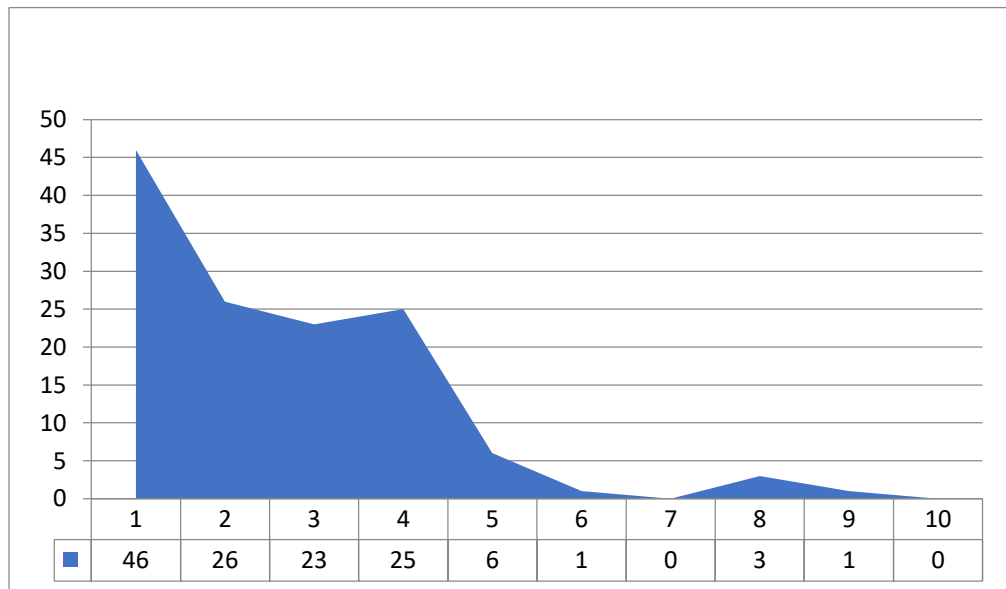


Diagramm 10: Ergebnisse zu Frage 7: Studiengang.

Nachdem die Studenten sich mit der App vertraut gemacht haben, mussten sie die Frage beantworten, ob die App Teach Me einwandfrei benutzen lässt. Bei 126 war die Antwort Ja, und bei 5 war die Antwort Nein. Während bei der Wiedergabe des Videomoduls ist die App abgestürzt. Das Diagramm 11 liefert das Ergebnis, ob die App einwandfrei benutzen lässt.

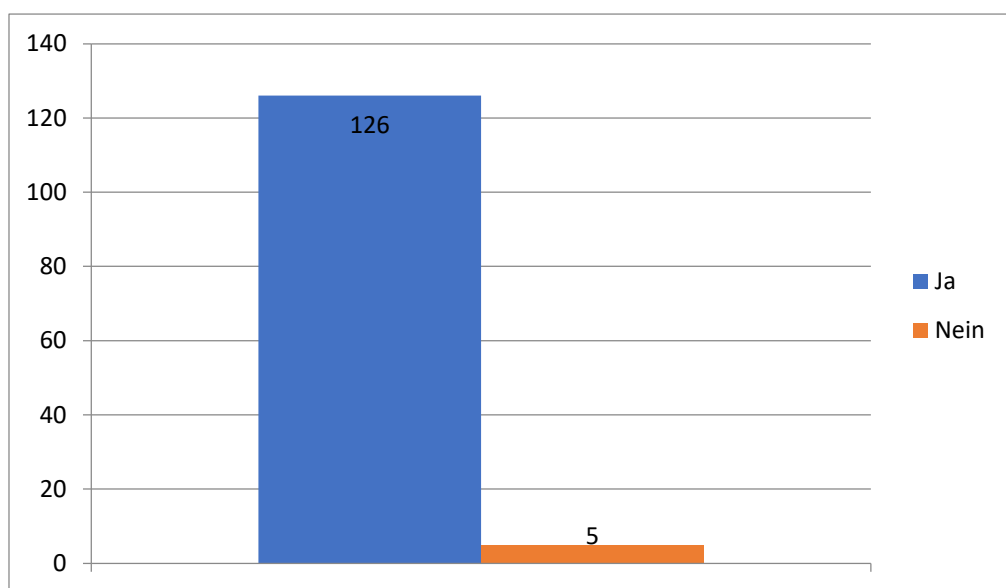


Diagramm 11: lässt sich die App einwandfrei benutzen?

Vorletzte Frage des Fragebogens war, ob die App genug Lernstoff besitzt. Diese Frage wurde von 106 Studenten bejaht, während 25 diese verneinten. Problem sei, dass es nicht genug Lernstoff vor allem für die höheren Semester gäbe. Das Diagramm 12 zeigt die Ergebnisse dafür.

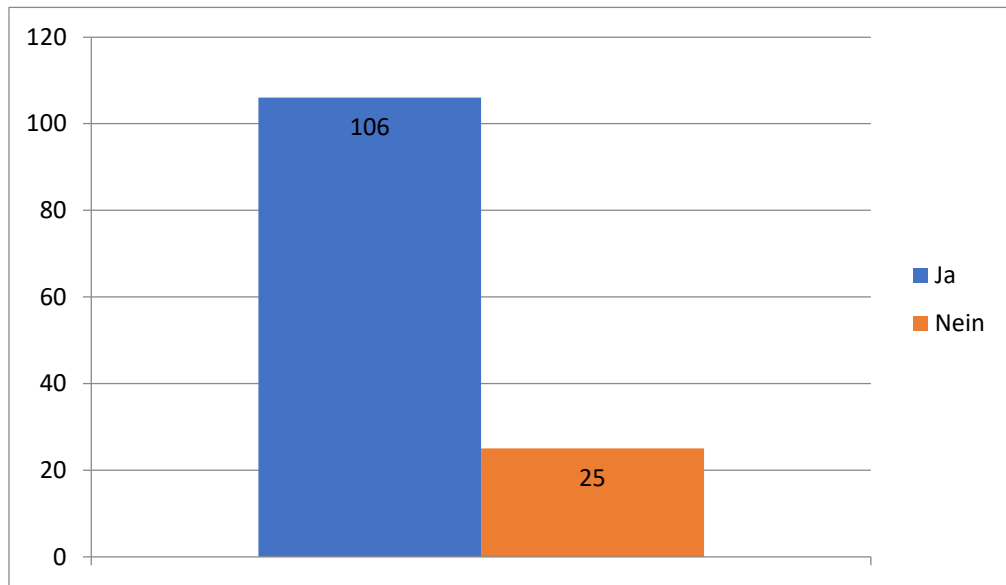


Diagramm 12: hat die App genug Lernstoff?

Bei der letzten Frage des Fragebogens mussten die Studenten entscheiden, ob die App Teach Me tatsächlich einen echten Tutor ersetzen könne. 106 Studenten könnten sich vorstellen, dass die App einen echten Tutor ersetzen könne. 25 beantworteten mit einem klaren Nein, da sie meinten, dass die App noch nicht genug Lernstoff besäße und ein weiteres Problem sahen sie darin, dass die App bis jetzt nur für Android zur Verfügung stehe. Es sollte für die andere Plattformen wie iOS und auch für Web entwickelt werden. Das Diagramm 13 bietet die Statistik, ob die App Teach Me einen echten Tutor ersetzen könne.

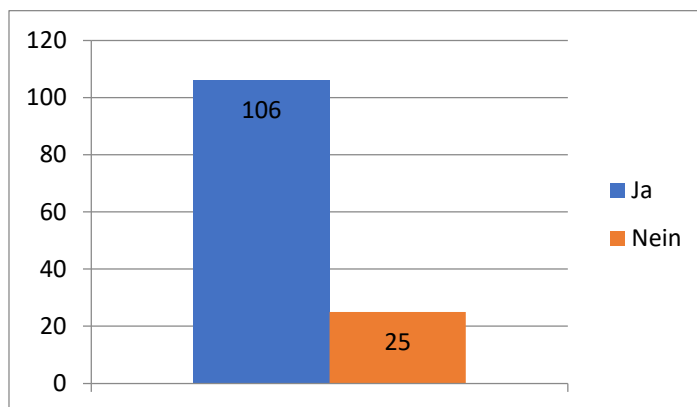


Diagramm 13: lässt sich die App als Real-Tutor ersetzen?

6.7 Kapitelfazit

Als erstes wurde im Kapitel 6 eine App Anforderung, die im Kapitel 4.1 definiert wird, abgeglichen. Die App Teach Me erfüllt alle gestellten Anforderungen. Danach wird in diesem Kapitel die Struktur des Fragebogens beschrieben. Im Folgenden kommt die Beschreibung der Forschungsversuche an der Universität Heidelberg, Universität Mainz und der Hochschule Worms. Es werden insgesamt 131 Studenten befragt. Zum Ende des Kapitels wurde die Fragebogenauswertung durchgeführt. Die Antworten auf den Fragebogen zeigen, dass die App sich einfach bedienen und sich als ein Tutor ersetzen lässt. Viele der Befragten können sich durchaus vorstellen, dass die App Teach Me einen echten Tutor ersetze, aber bei der App fehlen noch einige Funktionen, wie Sprachassistent und Lernstoff, vor allem für die höhere Semester.

7. Zusammenfassung und Ausblick

7.1 Zusammenfassung

#TODO

In dieser Masterarbeit wird eine native Anwendung für das Android Betriebssystem entwickelt, die als ein Tutor dienen soll. Dazu werden die folgende Forschungsfragen gestellt: Inwiefern lässt sich eine mobile Tutor App als ein Tutor verwenden? Kann eine mobile Tutor App einen realen Tutor oder Tutorin ersetzen? Ist Programmiersprache Java immer noch gut für Android Entwicklung anzuwenden? In der Arbeit wird auch eine Analyse von den aktuellsten App- Arten, Entwicklungssprachen, Integrierte Entwicklungsumgebung und vom Android Betriebssystem durchgeführt. Eine Untersuchung von Tutorien an den deutschen Universitäten wird veranstaltet. Dazu werden Sinn und Zweck von Tutorien, deren Arten und die wichtigsten Aufgaben von Tutoren erklärt. Bei Implementierung der App wird festgestellt, dass Entwicklungssprache Java gut für die Entwicklung ist, aber braucht eine Unterstützung von anderen Sprachen. Um die zwei weiteren Forschungsfragen zu beantworten, werden die Forschungsversuche mit der implementierten App an den ausgewählten Universitäten: Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms mit Hilfe eine Fragebogens realisiert. Es werden insgesamt 131 Studenten befragt. Die Antworten auf den Fragenbogen zeigen, dass die App sich einfach bedienen und sich als ein Tutor ersetzen lässt. Dazu sagen die Ergebnisse auch aus, dass die größere Anzahl an Studenten sich durchaus vorstellen kann, dass die App ein Tutor ersetze. Die Masterarbeit umfasst 88 Seiten, ausgestattet mit 27 Abbildungen, 8 Tabellen, 14 Diagramm und gemacht mit der Beteiligung von 31 Quellen.

7.2 Ausblick

#TODO

8. Literaturverzeichnis

1. [Online] On Story and Execution: Sebastian Thrun, Udacity, and The Future
<https://medium.com/udacity/on-story-and-execution-sebastian-thrun-udacity-and-the-future-77ce6e415208>
2. [Online] Udacity - About Us <https://www.udacity.com/us>
3. [Online] MIT and Harvard announce edX
<https://news.harvard.edu/gazette/story/2012/05/mit-and-harvard-announce-edx/>
4. [Online] #PICTURE_1
5. [Online] Coursera <https://blog.coursera.org/about/>
6. [Online] #PICTURE_2
7. [Online] USING SOCRATIVE TO ENHANCE IN-CLASS STUDENT ENGAGEMENT AND COLLABORATION <http://dx.doi.org/10.5121/ijite.2015.4302>
8. [Online] Improving Student Engagement in Higher Education through Mobile -Based Interactive Teaching Model Using Socrative
<http://eprints.sunway.edu.my/695/1/Lim%20Woan%20Ning%20improving%20student%20engagement.pdf>
9. [Online] #PICTURE_4
- 10.
- 11.
- 12.
- 13.
- 14.
- 15.
- 16.

17.

18.

19.

20

21.

22.

23.

24.

25.

26. Anhang B2.4 die 8. Fragebogen vom 26.06.2019 aus der Ruprecht-Karls-Universität Heidelberg

27. Anhang B2.1 die 13. Fragebogen von 03.06.2019 aus der Ruprecht-Karls-Universität Heidelberg

28. Anhang B3.4 die 12 Fragebogen von 23.07.2019 aus der Johannes-Gutenberg-Universität Mainz

29. Anhang B3.2 die 12 Fragebogen von 12.07.2019 aus der Johannes-Gutenberg-Universität Mainz

30. Anhang B4.2 die 12 Fragebogen von 14.07 aus der Hochschule Worms.

31.

Anhang A. Quellcode

A1. Quellcode ContentParserAdapter.java

```
package de.hsworms.inf3032.adapters;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;

import de.hsworms.inf3032.data.constant.ContentConstant;
import de.hsworms.inf3032.models.content.Contents;
import de.hsworms.inf3032.models.content.Item;

import static de.hsworms.inf3032.activity.MainActivity.mAdapter;
import static de.hsworms.inf3032.activity.MainActivity.mContentList;
import static de.hsworms.inf3032.engine.Provider.hideLoader;

public class ContentParserAdapter {

    public static String jsonData;

    public ContentParserAdapter (String _jsonData){
        jsonData = _jsonData;
    }

    public static void parseJson() {
        try {
            JSONObject jsonObjMain = new JSONObject(jsonData);
            JSONArray jsonArray1 = jsonObjMain.
                getJSONArray(ContentConstant.JSON_KEY_ITEMS);

            for (int i = 0; i < jsonArray1.length(); i++) {
                JSONObject jsonObj = jsonArray1.getJSONObject(i);

                String title = jsonObj.getString
                    (ContentConstant.JSON_KEY_TITLE);

                ArrayList<Item> items = new ArrayList<>();

                JSONArray jsonArray2 = jsonObj.
                    getJSONArray(ContentConstant.JSON_KEY_CONTENT);

                for (int j = 0; j < jsonArray2.length(); j++) {
                    JSONObject jsonObj2 = jsonArray2.getJSONObject(j);
                    String tag_line = jsonObj2.
                        getString(ContentConstant.JSON_KEY_TAG_LINE);

                    ArrayList<String> detailList = new ArrayList<>();

                    JSONArray jsonArray3 = jsonObj2.
                        getJSONArray(ContentConstant.JSON_KEY_DETAILS);

                    for (int k = 0; k < jsonArray3.length(); k++) {
                        String details = jsonArray3.get(k).toString();
                        detailList.add(details);
                    }
                    items.add(new Item(tag_line, detailList));
                }
                mContentList.add(new Contents(title, items));
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        hideLoader();
        mAdapter.notifyDataSetChanged();
    }
}
```

A2. Quellcode QuizParserAdapter.java

```
package de.hsworms.inf3032.adapters;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

import de.hsworms.inf3032.activity.QuestionActivity;
import de.hsworms.inf3032.data.constant.AppConstant;
import de.hsworms.inf3032.data.constant.ContentConstant;
import de.hsworms.inf3032.models.quiz.QuizModel;

import static de.hsworms.inf3032.activity.QuestionActivity.updateQuestionsAndAnswers;
import static de.hsworms.inf3032.engine.Provider.hideLoader;

public class QuizParserAdapter {

    String jsonData;

    public QuizParserAdapter(String _jsonData){
        jsonData = _jsonData;
    }

    public void parseJson() {
        try {
            JSONObject jsonObjMain = new JSONObject(jsonData);
            JSONArray jsonArray = jsonObjMain.getJSONArray(
                ContentConstant.JSON_KEY_QUESTIONNAIRY);
            for (int i = 0; i < jsonArray.length(); i++) {
                JSONObject jsonObj = jsonArray.getJSONObject(i);

                String question = jsonObj.getString(
                    ContentConstant.JSON_KEY_QUESTION);

                int correctAnswer = Integer.parseInt(
                    jsonObj.getString(
                        ContentConstant.JSON_KEY_CORRECT_ANS));

                JSONArray jsonArray2 = jsonObj.getJSONArray(
                    ContentConstant.JSON_KEY_ANSWERS);

                ArrayList<String> contents = new ArrayList<>();
                ArrayList<String> backgroundColors = new ArrayList<>();

                for (int j = 0; j < jsonArray2.length(); j++) {
                    String item_title = jsonArray2.get(j).toString();
                    contents.add(item_title);
                    backgroundColors.add(AppConstant.COLOR_WHITE);
                }
                QuestionActivity.mItemList.add(new QuizModel(
                    question, contents, correctAnswer,
                    backgroundColors));
                Collections.shuffle(QuestionActivity.mItemList);
            }
            hideLoader();
            updateQuestionsAndAnswers();
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```

Anhang B. Fragebogen

B1. Fragebogenmuster

Fragebogen für die Informatik Studenten der _____
um eine **Evolution** am _____ über die App „Teach Me“ durchzuführen.

Teach Me – ist ein Android App, die im Rahmen einer Masterarbeit für Studierende an vielen Universitäten gebaut wurde. Die App soll als ein Tutor dienen, besser gesagt einen Tutor ersetzen. Mit Hilfe dieser App sollen Studierende ihr Wissen verbessern.

Testgerät:

Studiengang:

Alter:

Erfahrung mit dem Umgang von Android Smartphones?

Ungenügend	Mangelhaft	Ausreichend	Befriedigend	Gut	Sehr gut

Haben Sie ähnliche Apps schon benutzt?

JA NEIN falls ja, welche _____
☐ ☐

1. Bitte machen Sie sich mit der „Programmoberfläche“ vertraut.

- Öffnen Sie ein Thema Ihrer Wahl
- Wählen Sie ein Modul aus
- Wählen Sie ein anderes Modul aus
- Navigieren Sie sich wieder zum Hauptbildschirm

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

2. Bitte machen Sie sich mit dem „Inhalt“ vertraut.

- Öffnen Sie ein Thema Ihrer Wahl
- Wählen Sie ein Modul aus
- Lassen Sie sich das ausgewählte Modul vorlesen
- Kopieren Sie den Inhalt des Moduls
- Verschicken Sie den Inhalt des Moduls weiter
- Navigieren Sie sich wieder zum Hauptbildschirm

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

3. Bitte machen Sie sich mit der „Programmsuche“ vertraut.

- Öffnen Sie eine Suche
- Schreiben Sie in der Suchleiste zum Beispiel: „Interfaces“
- Wählen Sie aus den Ergebnissen das Modul: Interfaces
- Navigieren Sie sich zum ersten Suchergebnis
- Favorisieren Sie dieses Modul
- Navigieren Sie sich wieder zum Hauptbildschirm und öffnen Sie Zusatzleiste
- Wählen Sie Favoriten und löschen Sie danach das favorisierte Modul
- Navigieren Sie sich wieder zum Hauptbildschirm

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

4. Bitte machen Sie sich mit dem „Quiz“ vertraut.

- Öffnen Sie ein Quiz
- Wählen Sie ein Thema aus:
- Prüfen Sie Ihr Wissen durch das Quiz.
- Navigieren Sie sich wieder zum Hauptbildschirm

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

5. Bitte machen Sie sich mit dem „Fragen und Antworten“ vertraut.

- Öffnen Sie „Fragen und Antworten“
- Wählen Sie ein Thema aus
- Wählen Sie eine Frage aus
- Navigieren Sie sich wieder zum Hauptbildschirm

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

6. Bitte machen Sie sich mit dem „Einstellungen“ vertraut.

- Öffnen Sie die Einstellungen
- Wechseln Sie die Sprache auf Englisch
- (Optional) Wechseln Sie die Textgröße aus.
- Navigieren Sie sich wieder zum Hauptbildschirm
- Führen Sie die Frage 1 bis 4 erneut durch

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

7. Bitte machen Sie sich mit dem „Studiengang“ vertraut.

- Wählen Sie anderen Studiengang
- Führen Sie die Frage 1 bis 5 erneut durch

Schwierigkeitsgrad der Aufgabe?

sehr leicht

sehr schwierig

○ ○ ○ ○ ○ ○ ○ ○ ○ ○

Lässt sich die App „Teach Me“ einwandernfrei benutzen?

JA NEIN Probleme _____

○ ○

Hat die App „Teach Me“ genug Lernstoff?

JA NEIN Probleme _____

○ ○

Lässt sich die App „Teach Me“ als Real-Tutor ersetzen?

JA NEIN Probleme _____

○ ○

Feedback

B2. Fragebogen aus der Ruprecht-Karls-Universität Heidelberg

B2.1 Fragebogen von 03.06.2019. Anzahl: 18

B2.2 Fragebogen von 07.06.2019. Anzahl: 20

B2.3 Fragebogen von 18.06.2019. Anzahl: 4

B2.4 Fragebogen von 26.06.2019. Anzahl: 15

B3. Fragebogen aus der Johannes-Gutenberg-Universität Mainz

B3.1 Fragebogen von 03.07.2019. Anzahl: 14

B3.2 Fragebogen von 12.07.2019. Anzahl: 9

B3.3 Fragebogen von 18.07.2019. Anzahl: 11

B3.4 Fragebogen von 23.07.2019. Anzahl: 18

B4. Fragebogen aus der Hochschule Worms

B4.1 Fragebogen von 06.08.2019. Anzahl: 10

B4.2 Fragebogen von 14.08.2019. Anzahl: 12