



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

Masterarbeit

Aleksander Soloninov

Matrikelnummer: 669037

Studiengang: Mobile Computing

Betreuer:

Prof. Dr. Bernd Ruhland

Masterarbeit eingereicht von **Aleksander Soloninov**

im Rahmen der **Masterprüfung** im Studiengang

Mobile Computing im Fachbereich Informatik

der **Hochschule Worms**

Betreuer Prüfer: **Prof. Dr. Bernd Ruhland**

Abgegeben am 7. Oktober 2019

Eidesstattliche 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 05.10.2019

Abstract

In dieser Masterarbeit wird eine native Anwendung für das Android Betriebssystem entwickelt, die als ein Tutor dienen soll. Dazu werden die folgenden Forschungsfragen gestellt: Inwiefern lässt sich eine mobile Tutor App als ein Tutor verwenden? Kann eine mobile Tutor App eine reale Tutorin oder einen realen Tutor ersetzen? Ist die Programmiersprache Java immer noch gut für die Android Entwicklung anzuwenden? In der Arbeit wird auch eine Analyse von den aktuellsten App-Arten, Entwicklungssprachen und der integrierten Entwicklungsumgebung vom Android Betriebssystem durchgeführt. Ferner wird eine Untersuchung von Tutorien an deutschen Universitäten veranstaltet. Dazu werden Sinn und Zweck von Tutorien, deren Arten und die wichtigsten Aufgaben von Tuto ren erklärt. Bei der Implementierung der App wird festgestellt, dass die Entwicklungssprache Java gut für die Entwicklung ist, aber eine Unterstützung von anderen Sprachen benötigt. Um die zwei weiteren Forschungsfragen zu beantworten, werden die Forschungsversuche mit der implementierten App an den ausgewählten Universitäten und Fachhochschule: Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms mit Hilfe eines Fragebogens realisiert. Es werden insgesamt 131 Studenten befragt. Die Antworten auf den Fragebögen zeigen, dass die App sich einfach bedienen und sich als einen Tutor einsetzen lässt. Darüber hinaus sagen die Ergebnisse aus, dass die größere Anzahl an Studenten sich durchaus vorstellen kann, dass die App einen Tutor auch ersetzen könne.

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Anfertigung dieser Masterarbeit unterstützt und motiviert haben.

Zunächst möchte ich mich bei meinem Betreuer Herrn Prof. Dr. Bern Ruhland, der meine Masterarbeit betreut und begutachtet hat, für die hilfreichen Anregungen und die konstruktive Kritik bei der Erstellung dieser Arbeit herzlich bedanken.

Ein besonderer Dank gilt allen Teilnehmern und Teilnehmerinnen meiner Befragung, Vielen Dank, dass ihr euch die Zeit für meine Befragung genommen habt. Nicht nur die sofortige Bereitschaft mich zu unterstützen, sondern auch eure Begeisterung für das Thema und eure Freundlichkeit schätze ich sehr. Ohne euch hätte diese Arbeit nicht entstehen können.

Tief verbunden und dankbar bin ich meiner Frau, Monika Korosik, für ihre unglaublich hilfreiche Unterstützung und ihr Verständnis bei der Anfertigung dieser Masterarbeit.

Ich danke vor allem meinen Eltern Alla Soloninova und Antonio Romano für jegliche Unterstützung in meinem Leben und während meiner gesamten Studienzeit. Danke, dass ihr mich in allen Lebenslagen unterstützt habt und mir immer mit Rat und Tat zur Seite standet.

Inhaltsverzeichnis

Abkürzungen.....	9
Abbildungsverzeichnis.....	10
Tabellenverzeichnis	12
Diagrammverzeichnis.....	13
Zeitplan	14
Gender Erklärung.....	15
1 Einleitung.....	16
1.1 Problemstellung.....	16
1.2 Motivation.....	17
1.3 Forschungskonzept.....	17
1.4 Zielsetzung und Erkenntnisinteresse	18
1.5 Forschungsstand.....	19
1.6 Gliederung der Arbeit	24
2 Grundlagen zur Entwicklung einer Android Applikation.....	26
2.1 Mobile Applikationen.....	26
2.1.1 Native App	27
2.1.2 Web App	28
2.1.3 Hybride App.....	28
2.2 Entwicklung der Sprachen für das Android Betriebssystem.....	29
2.2.1 Java.....	29
2.2.2 Kotlin.....	30
2.2.3 Scriptsprachen.....	30
2.3 Integrierte Entwicklungsumgebung.....	31

2.3.1 Android Studio.....	32
2.3.2 Basic for Android	34
2.3.3 Visual Studio.....	35
2.3.4 AIDE.....	36
2.4 Android Betriebssystem	37
2.4.1 Anwendungskomponenten.....	38
2.4.2 Pattern.....	40
2.5 Git Versionierung.....	42
2.6 Auswahl einer Technologie.....	42
3 Tutorien an den deutschen Universitäten.....	44
3.1 Sinn und Zweck von Tutorien an Universitäten	46
3.2 Arten von Tutorien.....	46
3.2.1 Orientierungstutorien.....	47
3.2.2 Fachtutorien und Übungsgruppen.....	47
3.3 Aufgaben der Tutoren und Tutorinnen.....	47
4 Konzeption und Entwurf.....	49
4.1 App Anforderungen	49
4.1.1 Funktionale Anforderungen	49
4.1.1 Optionale Anforderungen.....	50
4.1.1 Nicht-Funktionale Anforderungen.....	51
4.2 Mockup	51
4.3 Prototyp.....	55
5 Implementierung	59
5.1 Datenarchitektur.....	59
5.2 Benutzeroberflächen	64

5.2.1 Hauptbildschirm	64
5.2.2 Suche.....	68
5.2.3 Quizfragen	69
5.2.4 Interviewfragen.....	71
5.2.5 Einstellungen.....	72
5.2.6 Favoriten.....	73
5.3 Kapitelfazit	74
6 Anwendungsvergleich	75
6.1 Anforderungsabgleich.....	76
6.1.1 Abgleich der funktionalen Anforderungen	76
6.1.1 Abgleich der optionalen Anforderungen	77
6.1.2 Abgleich der nicht-funktionalen Anforderungen	77
6.2 Fragebogen und Forschungsversuch	78
6.3 Fragebogenauswertung	81
6.4 Kapitelfazit	92
7 Zusammenfassung und Ausblick	93
7.1 Zusammenfassung.....	93
7.2 Ausblick	94
8 Literaturverzeichnis	95
Anhang A. Quellcode.....	99
A1. Quellcode ContentParserAdapter.java	99
A2. Quellcode QuizParserAdapter.java	100

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: Udacity [5].....	20
Abbildung 1.2: Bildschirme der edX App.....	21
Abbildung 1.3: Bildschirme der Coursera App.....	22
Abbildung 1.4: Online Kurs RAUM 1 in der Socrative App	23
Abbildung 1.5: Gliederung der Arbeit.....	25
Abbildung 2.1: Native, Web und Hybride App [11]	26
Abbildung 2.2: Android Studio 3.4.1 mit geöffnetem Projekt Teach Me.....	32
Abbildung 2.3: Auswahl von Geräten im ADV Manager	34
Abbildung 2.4: Interface der IDE Basic for Android.....	35
Abbildung 2.5: Interface der IDE Visual Studio.....	36
Abbildung 2.6: AIDE Interface mit geöffnetem Projekt Teach Me.....	37
Abbildung 2.7: Marktanteil der mobilen Betriebssysteme weltweit [25]	38
Abbildung 2.8: Lebenszyklus einer Android Anwendung [26]	40
Abbildung 2.9: MVP Pattern [27]	41
Abbildung 3.1: Auswahl einer Übungsgruppe für die Einführung in die Theoretische Informatik in MÜSLI [29].....	44
Abbildung 3.2: : Informatik-Vorlesungsverzeichnis in JOGU-StINe [30]	45
Abbildung 3.3: Vorlesungsplan für das 1. Semester der Angewandten Informatik [31]..	46
Abbildung 3.4: Der Aufbau eines Tutoriums [37]	48
Abbildung 4.1: Mockup des Hauptbildschirms	52

Abbildung 4.2: Mockup der Kapitelansicht.....	53
Abbildung 4.3: Mockup des Quizbildschirms.....	54
Abbildung 4.4: Prototyp des Hauptbildschirms der App Teach Me	56
Abbildung 4.5: Prototyp der Kapitelansicht der App Teach Me	57
Abbildung 4.6: Prototyp des Quizfragenbildschirms der App Teach Me	58
Abbildung 5.1: Teil-Code aus CS_GermanContentFile.json	60
Abbildung 5.2: Teil-Code aus QA_GERMAN_C++.json.....	61
Abbildung 5.3: Funktion „parseJson“ aus ContentParserAdapter.java	62
Abbildung 5.4: Funktion „parseJson“ aus QuizParserAdapter.java.....	63
Abbildung 5.5: Hauptbildschirm der App Teach Me	65
Abbildung 5.6: Die Auswahl von Kapiteln der App Teach Me	66
Abbildung 5.7: Kapitelansicht der App Teach Me	67
Abbildung 5.8: Begriffssuche in der App am Beispiel des Suchwortes: shell	68
Abbildung 5.9: Quizfragenbildschirm.....	69
Abbildung 5.10: Quizfragenergebnisbildschirm.....	70
Abbildung 5.11: Interviewfragenbildschirm.....	71
Abbildung 5.12: Einstellungen.....	72
Abbildung 5.13: Favoriten.....	73
Abbildung 6.1 : Fragebogenmuster - Vorderseite.....	79
Abbildung 6.2 : Fragebogenmuster -Rückseite	80

Tabellenverzeichnis

Tabelle 1: Funktionale Anforderungen von F.1 bis F.4	49
Tabelle 2: Funktionale Anforderungen von F.5 bis F.11.....	50
Tabelle 3: Optionale Anforderungen.....	50
Tabelle 4: Nicht-funktionale Anforderungen.....	51
Tabelle 5: Abgleich der funktionalen Anforderungen	76
Tabelle 6: Abgleich der optionalen Anforderungen	77
Tabelle 7: Abgleich der nicht-funktionale Anforderungen von NF.1 bis NF.2	77
Tabelle 8: Abgleich der nicht-funktionale Anforderungen von NF.3 bis NF.5	78

Diagrammverzeichnis

Diagramm 1: Nutzung der Geräte beim Anwendungsvergleich	82
Diagramm 2: Erfahrung mit dem Umgang von Android Smartphones	82
Diagramm 3: Nutzung von ähnlichen Apps	83
Diagramm 4: Ergebnisse zu Frage 1 – Programmoberfläche	84
Diagramm 5: Ergebnisse zu Frage 2 – Inhalt	84
Diagramm 6: Ergebnisse zu Frage 3 – Programmsuche.....	85
Diagramm 7: Ergebnisse zu Frage 4 – Quiz	86
Diagramm 8: Ergebnisse zu Frage 5 – Fragen und Antworten	87
Diagramm 9: Ergebnisse zu Frage 6 – Einstellungen.....	87
Diagramm 10: Ergebnisse zu Frage 7 – Studiengang.....	88
Diagramm 11: Lässt sich die App einwandfrei benutzen?.....	89
Diagramm 12: Hat die App genug Lernstoff?	89
Diagramm 13: Kann die App einen Real-Tutor ersetzen?.....	90

Zeitplan

Dauer: 6 Monate (vom 07.04.2019 bis 07.10.2019)

Bis 08.04.2019: Literaturrecherche

Bis 15.04.2019: Konzeption und Technologie Auswahl

Bis 31.05.2019: Implementierung der App

Bis 24.06.2019: Forschungsversuch und App Test an der Universität Heidelberg durchführen und Feedback sammeln

Bis 30.07.2019: Ergebnisse des ersten Tests auswerten und Rohfassung Hauptteil

Bis 10.07.2019: App Erweiterung durchführen

Bis 01.08.2019: Forschungsversuch, Durchführung des App Tests an der Universität Mainz und Einholen von Feedback

Bis 07.08.2019: App Erweiterung durchführen

Bis 15.08.2019: Forschungsversuch, App Test an der Hochschule Worms durchführen und Feedback sammeln

Bis 25.09.2019: Auswertungsvergleich und Verfassung des Hauptteils

Bis 30.09.2019: Vervollständigung des Hauptteils und Schlussverfassung

Bis 01.10.2019: Überarbeitung und Korrektur

Bis 02.10.2019: Layout, Titelblatt und restliche Korrektur

Am 05.10.2019: Druck

Am 07.10.2019: Abgabe

Gender Erklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Masterarbeit die Sprachform des generischen Maskulinums angewendet. Es wird an dieser Stelle darauf hingewiesen, dass die ausschließliche Verwendung der männlichen Form geschlechtsunabhängig verstanden werden soll.

1 Einleitung

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 dazu immer in Bibliotheken, in Seminare und in Tutorien gehen.

1.1 Problemstellung

Die Verbesserung des Engagements der Studierenden ist eine Herausforderung, der sich viele Hochschuleinrichtungen stellen müssen [1]. Das Problem bei der heutigen Ausbildung ist, dass die meisten Pädagogen immer noch so unterrichten, wie sie es in der Vergangenheit getan haben. Die einseitige Kommunikation von Standardvorlesungen und PowerPoint-Präsentationen war früher nicht mehr effektiv, um das Lernen zu fördern. Im Laufe der Jahre wurden umfangreiche Strategien, Methoden und Lernwerkzeuge entwickelt, um dieses Problem zu lösen. In den letzten Jahren – bedingt durch die Zunahme der Zahl der Studenten, die über mobile Geräte auf das Internet zugreifen können – gab es ein wachsendes Interesse an der Nutzung mobiler Technologien zum Lernen, um die Beteiligung der Studenten zu verbessern [2].

In der Vorlesung wird nur ein Basisstoff vorgetragen und Studenten können der Lehrperson nicht immer Fragen stellen. Deswegen gibt es seit ca. 10 Jahren an vielen Universitäten zusätzliche „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, eine Tutorin oder einen Tutor zu besuchen. Dazu kommt noch in einigen Fällen die Sprachbarriere der Studierenden. Internet und die Google Suchmaschine bieten vielartige Möglichkeiten, um brauchbare und verwendbare Informationen 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 Zukunft ersetzen werden. Andererseits wird gesagt, dass die Technologie nur bestehende Probleme beim Lernen löst.

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. 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 eine reale Tutorin oder einen realen Tutor ersetzen?
- Ist die Programmiersprache Java immer noch gut für die Android Entwicklung anzuwenden?

Um den Fragen nachzugehen, wird die App Teach Me an der Hochschule Worms, der Universität Heidelberg und der Universität Mainz getestet. Hierbei handelt es sich um eine Live App, die getestet wird, wobei ein Fragebogen mit Feedback auszufüllen ist.

In der Arbeit werden die folgenden Methodik-Schritte durchgeführt:

1. Eine Analyse von Fragebögen mit Feedback wird durchgeführt, um zu untersuchen, wie die App beim Lernen unterstützen kann.
2. Wie hat die App mit ihren Benutzern kommuniziert und wie wurde das „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 Informatik-Studierende an Universitäten, die als ein Tutor dienen soll. In der App sollte man verschiedene kostenlose Informatik Kurse auf Deutsch, Englisch und anderen Sprachen in einer Text-, Bild- oder Videoform auswählen und die dazugehörigen Informationen sehen können. Mit einem Quiz und Interviewfragen in der App sollten Studierende ihr Lernwissen überprüfen. Die Konzeption, der Entwurf und die Implementierung der App werden in den Kapiteln 4 und 5 beschrieben.

Das zweite Ziel ist die Befragung der Informatik-Studenten an der Ruprecht-Karls-Universität Heidelberg, der Johannes-Gutenberg-Universität Mainz und der Hochschule Worms. 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.

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 6.3 sind die Diagramme dargestellt, die die Fragebogenergebnisse anzeigen.

Folgende Aufgaben werden zusätzlich an die Arbeit gestellt:

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

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 [3].

Udacity, eine private Online-Akademie, wurde von Thrun gegründet, in der nicht ganze Bildungseinrichtungen, sondern nur einzelne Wissenschaftler kooperieren. Er wählte eigene Kurse im Zusammenhang mit Informatik aus. Die Vorträge wurden in Form von kurzen Video-Vorträgen und kleinen Tests abgehalten. Es wurde die Kommunikation der Studenten auf dem Kurs Blog organisiert, wo die Studenten die Möglichkeit hatten, Fragen zu stellen. Aufbauend auf einem Likes-System standen die besten Fragen ganz oben auf der Liste. Die Studenten 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 [4]. Abbildung 1.1 bildet Udacity ab.

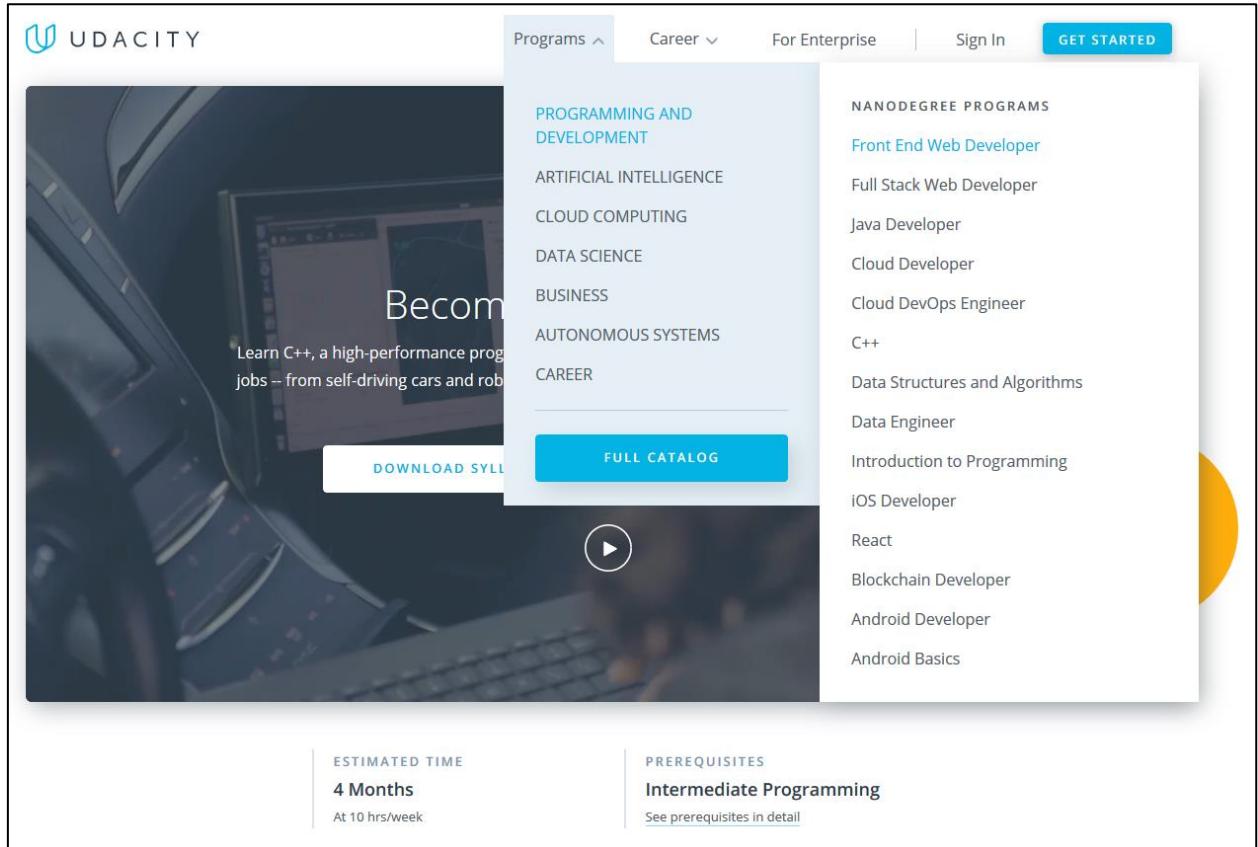


Abbildung 1.1: Udacity [5]

Im Mai 2012 gaben die Harvard University und das Massachusetts Institute of Technology die Gründung einer gemeinnützigen Partnerschaft bekannt, die bis heute 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 optimieren[6]. Abbildung 1.2 zeigt die App Oberflächen der edX App.

The screenshot shows the edX mobile application interface. On the left, there's a sidebar with navigation links: 'Discovery' (selected), 'Courses', 'Programs', and 'Degrees'. Below these are two course cards:

- HarvardX Data Science**: A 'PROFESSIONAL CERTIFICATE PROGRAM' featuring a digital circuit background.
- Wharton Strategic Management**: A 'PROFESSIONAL CERTIFICATE PROGRAM' featuring icons related to business and data analysis.

The main content area displays a course titled **Programming for Everybody (Getting Started with Python)**. At the top, it shows the title 'PYTHON GETTING STARTED' with a yellow 'M' logo. Below the title, the course description reads: "This course is a 'no prerequisite' introduction to Python Programming. You will learn about variables, conditional execution, repeated execution and how we use functions. The homework is done in a web browser so you can do all of the programming assignments on a phone or public computer." The University of Michigan logo is at the bottom of this section.

To the right, a vertical sidebar lists course sections with due dates and download counts:

- 1.1 Objekte: Aufgaben due Sep. 20, 2019 (1 download)
- 1.2 Klassen: Aufgaben due Sep. 20, 2019 (2 downloads)
- 1.3 Methoden und Parameter: Aufgaben due Sep. 20, 2019 (2 downloads)
- 1.4 Beziehungen: Aufgaben due Sep. 20, 2019 (1 download)
- 1.5 Zustände von Objekten: Aufgaben due Sep. 20, 2019 (1 download)

Below this sidebar, there are sections for 'Fragen und Feedback' and 'Kapitel 2: Algorithmen'.

Abbildung 1.2: Bildschirme der edX App

Eine Reihe anderer Universitäten, die dem Beispiel von Udacity und edX folgten, bieten auch Massive Open Online Courses an. So gründeten die Universitäten von Stanford, Princeton, Pennsylvania und Michigan eine neue Partnerschaft namens Coursera. Coursera präsentiert mehr als 2000 Kurse. Die Liste der Disziplinen ist breit gefächert: Ingenieursdisziplinen, Geisteswissenschaften, Kunst, Medizin, Biologie, Programmierung, Recht, 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 untertitelt [7]. Abbildung 1.3 stellt das Interface von Coursera dar.

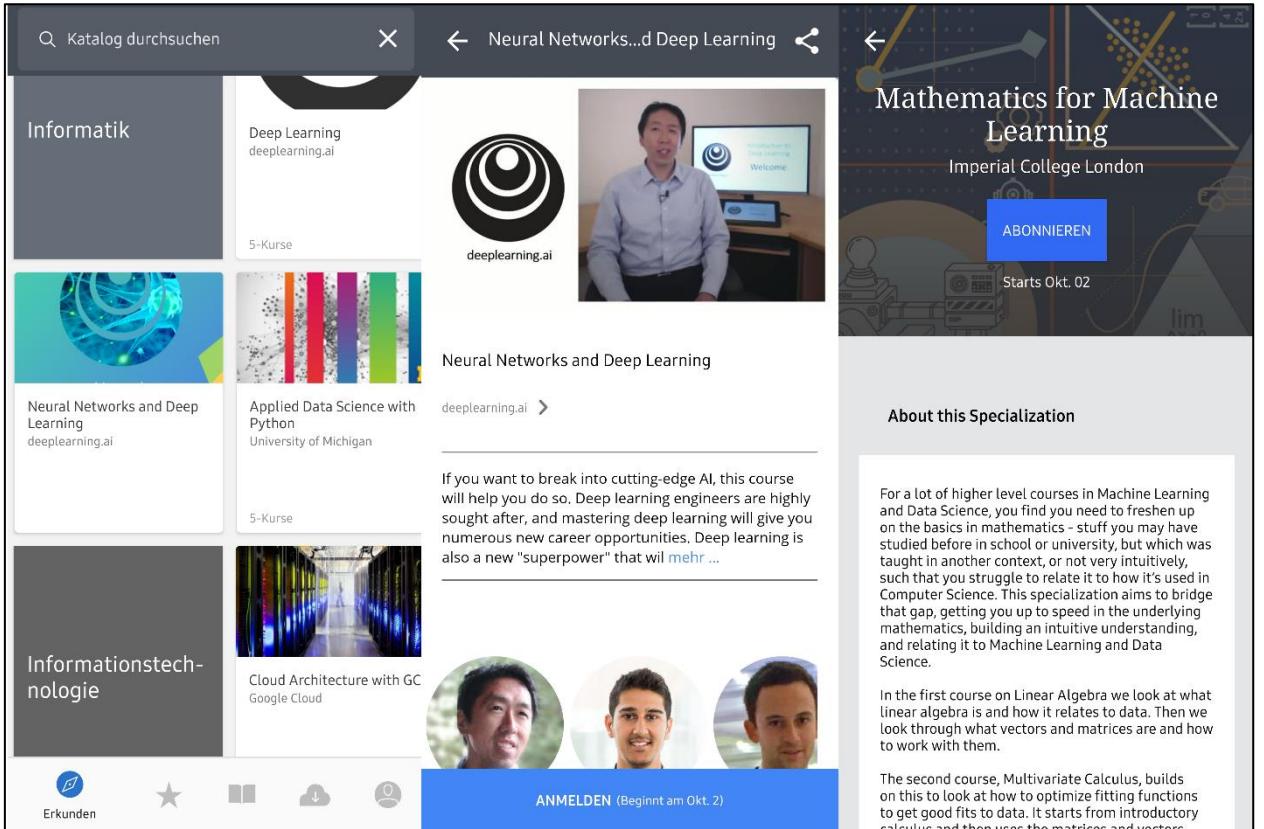


Abbildung 1.3: Bildschirme der Coursera App

In den letzten Jahren sind neue Werkzeuge im Kontext von Informations- und Kommunikationstechnologien entstanden, die die aktive Teilnahme, die Interaktion von Lehrenden 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 [8]. Das Socrative App Interface ist in Abbildung 1.4 dargestellt.

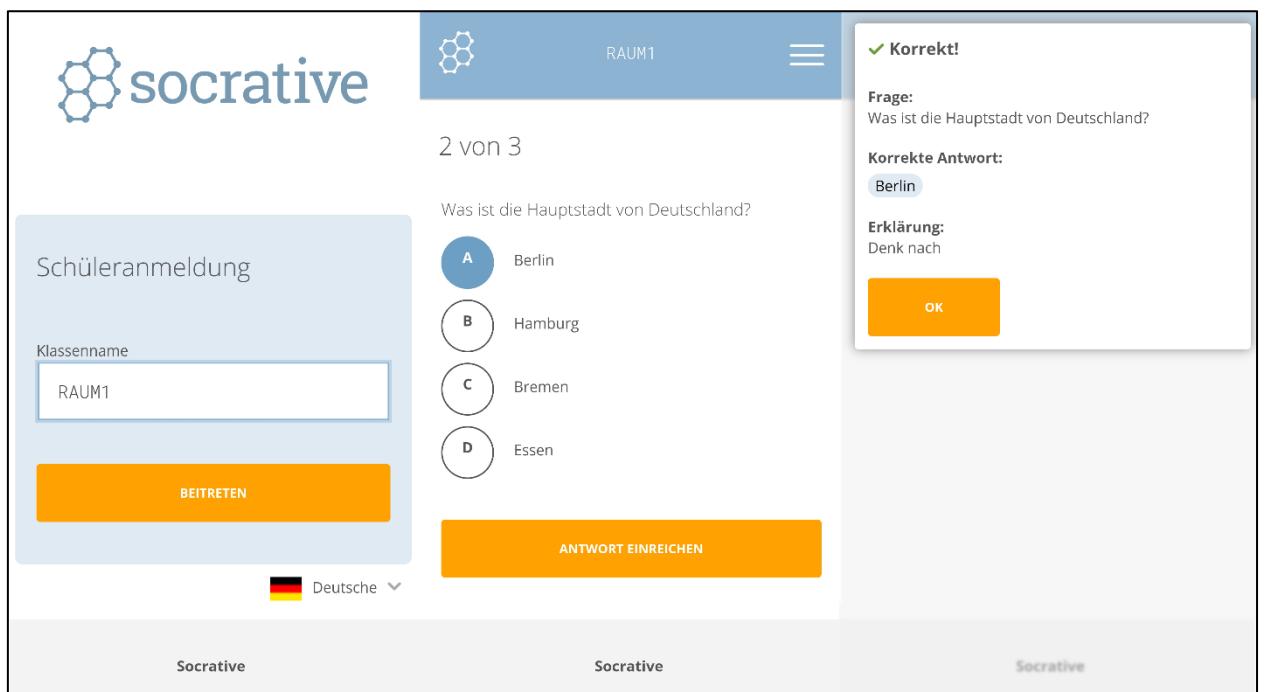


Abbildung 1.4: Online Kurs RAUM 1 in der Socrative App

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 Lehrenden 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 [9].

Ein Experiment an der Sheffield Hallam University wurde entwickelt, um die Teilnahme von Studenten in der Hochschulbildung für Ingenieurs-Studenten zu bewerten. Das Student Response-System basierte darauf, das sofortige Feedback eines Studenten mit kurzen Quizfragen von 10 bis 15 Minuten mit der Socrative Software zu erhalten. Die Struktur der Fragen war eine Mischung aus wahren oder falschen Antwortmöglichkeiten, Mehrfachauswahloptionen und kurzen Antwortfragen. Das Experiment wurde 2 Semester lang im einjährigen Ingenieurmodul 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 [10].

1.6 Gliederung der Arbeit

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

Kapitel 2 stellt die Grundlagen zur Entwicklung einer Android App dar. Zuerst werden die aktuellsten Arten von mobilen Applikationen beschrieben. Anschließend folgt die Beschreibung der Entwicklung der Sprachen für das Android Betriebssystem und ihrer dazu passenden, integrierten Entwicklungsumgebungen. Dazu werden das Android Betriebssystem und ihre Anwendungskomponenten und Patter erläutert. Am Schluss wird die Git Versionierung beschrieben und die Auswahl einer Technologie für die Entwicklung der App Teach Me durchgeführt.

Das Kapitel 3 beschäftigt sich mit der Bedeutung von Tutorien im Studienprozess. Es werden sowohl der Sinn und Zweck als auch die Arten von Tutorien beschrieben. Zum Schluss werden die Aufgaben der Tutoren und Tutorinnen dargestellt.

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

Daraufhin wird im Kapitel 5 die App Teach Me implementiert. Es wird die Inhaltsdatenstruktur 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 implementierten App abgeglichen. Anschließend folgt die Beschreibung des Fragebogens. Danach werden die Ergebnisse aus den Forschungsversuchen dargestellt und ausgewertet. Zum Schluss kommt das Kapitelfazit.

Abschließend folgen im Kapitel 7 eine Zusammenfassung und ein Ausblick.

Abbildung 1.5 stellt einen kurzen visuellen Überblick über das 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 der Sprachen für das Android Betriebssystem	Integrierte Entwicklungsumgebung
Android Betriebssystem	Git Versionierung	Auswahl einer Technologie
Kapitel 3 - Tutorien an den deutschen Universitäten		
Sinn und Zweck von Tutorien 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		
Anforderungsabgleich	Fragebogen	Fragenbogenauswertung
Kapitelfazit		
Kapitel 7 – Zusammenfassung und Ausblick		
Zusammenfassung	Ausblick	

Abbildung 1.5: Gliederung der Arbeit

2 Grundlagen zur Entwicklung einer Android Applikation

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 fallen die Vorstellung der App- Arten, der Entwicklungssprachen, der Entwicklungsumgebungen sowie des Android Betriebssystems. Zum Schluss des Kapitels erfolgt die technologische Auswahl für die Entwicklung der App Teach Me.

2.1 Mobile Applikationen

Der Entwickler kann aus drei Optionen für mobile Anwendungen wählen:

- Native App
- Web App
- Hybride App

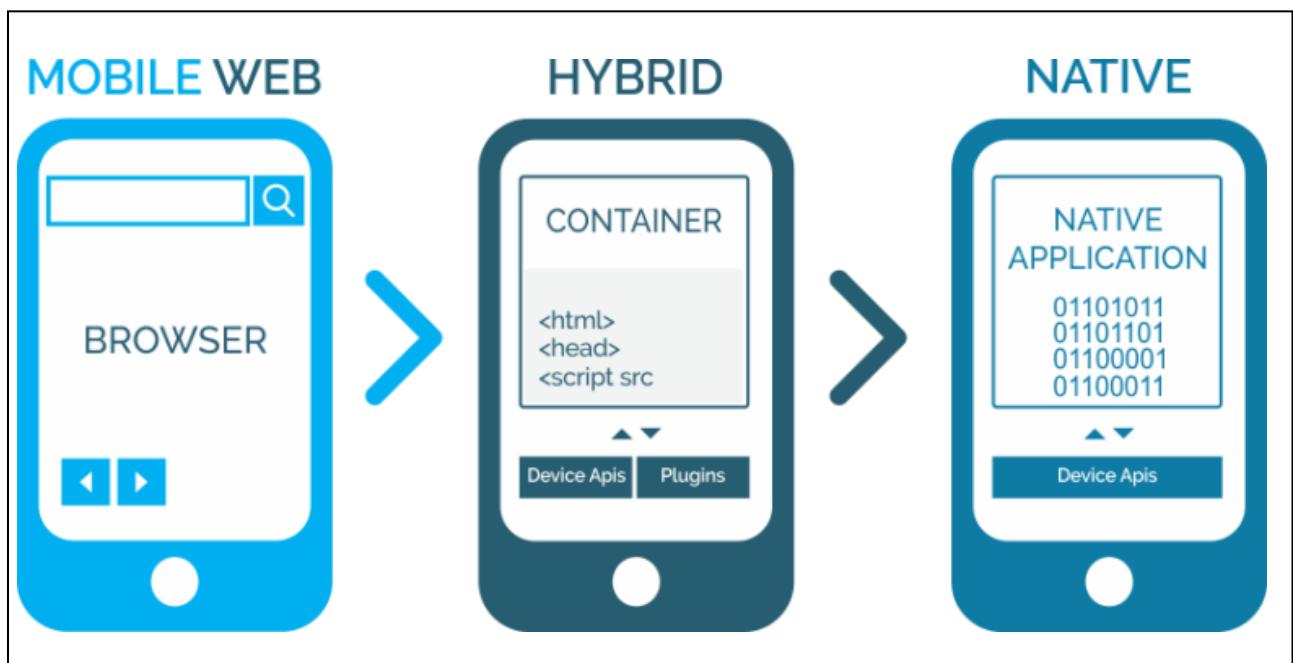


Abbildung 2.1: Native, Web und Hybride App [11]

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

- von der technischen Bewertung der Entwickler [12]
- von der Notwendigkeit auf Informationen auf dem Gerät zuzugreifen [12]
- von den Auswirkungen der Internetgeschwindigkeit auf die App [12]
- von einer oder mehreren Plattform-Apps [12]

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 einer nativen App ist 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 [13].

Einige andere wichtige Vorteile von nativen Anwendungen:

- Mit der Geolocation können Unternehmen ihre Treueprogramme oder Promotionen anpassen. Verbraucher können benachrichtigt werden, wenn sie in der Nähe von physischen Geschäften sind [12].
- Aktivitäten oder Inaktivität der Nutzer können leicht gesammelt und analysiert werden, wodurch die Wirksamkeit der gesamten Anwendung oder ihrer einzelnen Funktionen bewertet werden kann [12].
- Native Anwendungen neigen dazu, besser zu funktionieren. Web-Anwendungen werden manchmal erstellt, um native zu simulieren, aber sind auf Internetgeschwindigkeit und Designfähigkeiten beschränkt [12].

Mögliche Nachteile:

- Native Anwendungen sind oft teurer in der Entwicklung [12].
- Native Apps müssen von jedem App Store genehmigt werden [12].

2.1.2 Web App

Web Apps arbeiten über einen Webbrowser auf dem Gerät des Benutzers. Diese Apps sind im Wesentlichen individualisierte Websites, die so gestaltet sind, dass sie wie native Apps aussehen und verwendet werden, aber sich nicht wirklich auf dem Gerät des Benutzers befinden. Mit einer guten, qualitativ hochwertigen Entwicklung, die eine Größenanpassung und ein Scrollen beinhaltet, funktionieren Webanwendungen oft ähnlich wie native Anwendungen [13].

Vorteile von Webanwendungen:

- Webbasierte Anwendungen werden leichter unterstützt und können auf der Plattform mit jedem Betriebssystem funktionieren [12].
- Entwickler können Apps anbieten, ohne dass sie von App Stores genehmigt werden müssen [12].
- Schnellere Entwicklung von Schleifen mit CSS, HTML und JavaScript [12].

Und Nachteile:

- Webanwendungen haben keinen Zugriff auf das Gerät des Benutzers [12].
- Benutzer müssen Web App über das Internet verwenden, was die Sicherheitskontrolle erheblich reduziert [12].
- Die Suche nach einer App kann schwierig sein [12].

2.1.3 Hybride App

Hybridanwendungen stehen zwischen nativen und Webanwendungen. Hybride Apps werden so erstellt, dass die Apps 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 [13].

Vorteile von Hybrid-Anwendungen:

- Hybrid-Anwendungen bieten die beste Funktionalität und Personalisierung für den Benutzer [12].
- Entwickler sind nicht auf eine einzige Plattform beschränkt, sondern können eine hybride Anwendung erstellen, die mit mehreren Plattformen funktioniert [12].
- Hybride sind eine gute Option für Entwickler, die visuell gesättigte Anwendungen wie Spiele erstellen [12].

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 [12].
- Die Entwicklung erfordert zusätzliche Zeit und Mühe, damit eine solche Anwendung des Benutzers als nativ aussieht und sich anfühlt [12].
- App Stores können Hybridanwendungen ablehnen, die nicht reibungslos genug funktionieren [12].

2.2 Entwicklung der Sprachen für das Android Betriebssystem

Mit jeder Entwicklung einer Programmiersprache und mit jedem Framework sind eigene Komplexitäten und Nuancen wie auch Vor- und Nachteile verbunden. Im weiteren Verlauf dieses Kapitels werden die wichtigsten Sprachen für das Schreiben von Android-Anwendungen beschrieben und analysiert.

2.2.1 Java

Java ist offizielle Programmiersprache, die von der Android Studio-Entwicklungsumgebung unterstützt wird. Laut einer jährlichen Umfrage von Stack Overflow ist Java 2019 unter den fünf beliebtesten Programmiersprachen vertreten [14].

Java bezieht sich auf die meisten offiziellen Google-Dokumentationen, und es ist nicht schwer, zahlreiche bezahlte und kostenlose Bibliotheken und Handbücher zu finden.

Leider verhindert die Komplexität von Java, dass jeder damit programmiert. Als objektorientierte Programmiersprache hat sie eine Reihe von Funktionen in Form von Klassen, Konstruktoren, Ausnahmen, die zu App-Abstürzen 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 [15].

Bei der Entwicklung in Java unter Android werden nicht nur Java-Klassen verwendet, die Codes 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 wird 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. Für die Datenbank wird SQLite benutzt [15].

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. Kotlin ist 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, sie erst zu lernen, nachdem man sich mit Java-Wissen vertraut gemacht hat [15].

Kotlin ist mit Java kompatibel und verursacht keine Leistungseinbußen und größere Dateigrößen. Der Unterschied zu Java ist, dass es einen weniger dienstlichen, einen 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 [15].

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 [16].

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 Monetarisierungsmöglichkeiten, und es existiert ein anständiger Markt, in dem Entwickler nützliche Plugins finden können [17].

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 [18].

Programmierer können auf diese Weise Android-Anwendungen erstellen, indem sie Funktionen von Adobe Cordova verwenden. Es ist ein Open-Source-Framework, das auch iOS-Betriebssysteme Windows 10 Mobile, Blackberry, Firefox, und viele andere unterstützt. Damit Cordova nützlich ist, erfordert es vom Programmierer eine disziplinierte Arbeitsweise, um eine anständige Anwendung darin zu erstellen. Daher bevorzugen viele Programmierer das Ionic Framework-Projekt [18].

Es gibt auch eine andere Möglichkeit: die Verwendung der React Native Bibliothek. React Native kann auf Android und iOS bereitgestellt werden. Facebook, Instagram und andere große Unternehmen nutzen diese Bibliothek [19].

2.3 Integrierte Entwicklungsumgebung

Um eine Android-Anwendung zu entwickeln, muss man bestimmte Programmierungsumgebungen verwenden. 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 es, die Arbeit des Programmierers bequemer und produktiver zu machen.

IDE-Anforderungen:

- Sprachsyntax-Hervorhebung und Zeilennummerierung [20]
- Funktion zum Beenden des Codeschreibens und zum Anzeigen von Parametern [20]
- Debuggen der Anwendung [20]
- Versionsverwaltung [20]

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

2.3.1 Android Studio

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

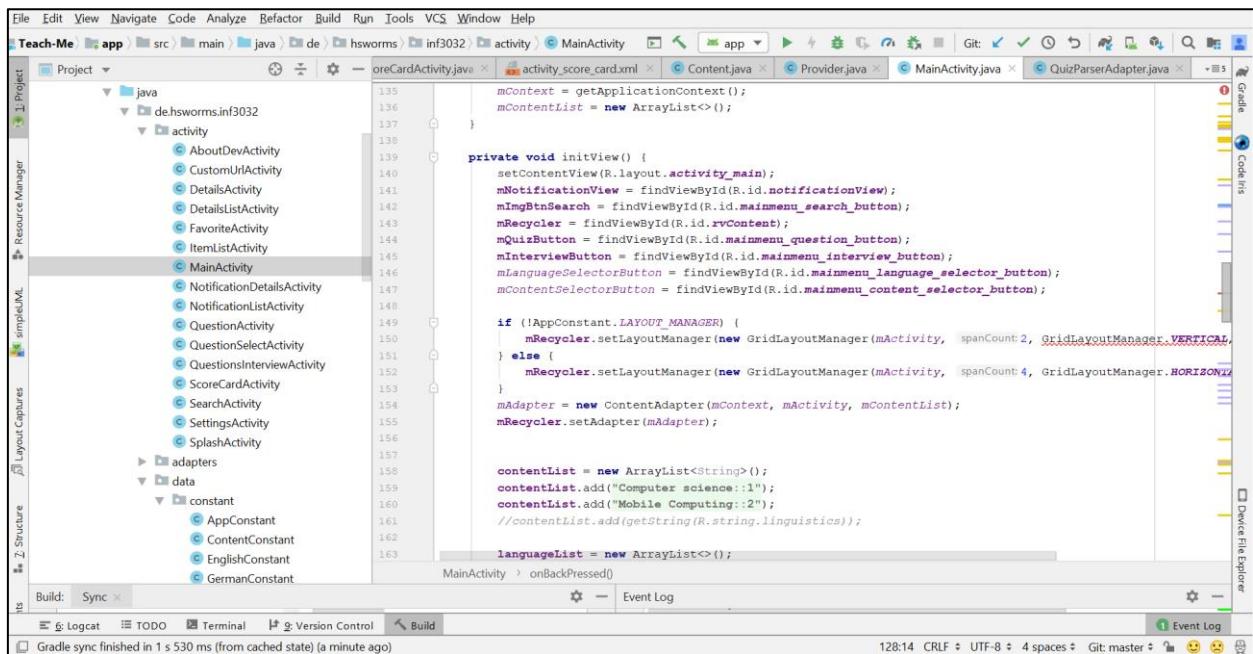


Abbildung 2.2: Android Studio 3.4.1 mit geöffnetem Projekt Teach Me

Android Studio fungiert als Editor für die vom Programmierer gewählte Programmiersprache. Es unterstützt Java, C++ sowie Kotlin. Android Studio fungiert auch als Compiler, der APK-Dateien und Dateisysteme erstellen kann, um das eigene 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 besteht dieses Paket aus dem Android SDK, aber Java JDK muss immer noch separat heruntergeladen und installiert werden [20].

Android Virtual Device Manager

Das AVD Manager-Tool kommt mit dem Android Studio zusammen im Paket. Die Abkürzung AVD steht für Android Virtual Device. Es ist ein Emulator, um Android-Anwendungen auf dem 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 dem 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 für die beliebtesten Gadgets bieten. Die Leistung des Werkzeugs wird ständig verbessert [20]. Die Abbildung 2.3 liefert Beispiele für Geräte, die auf AVD zu Verfügung stehen.

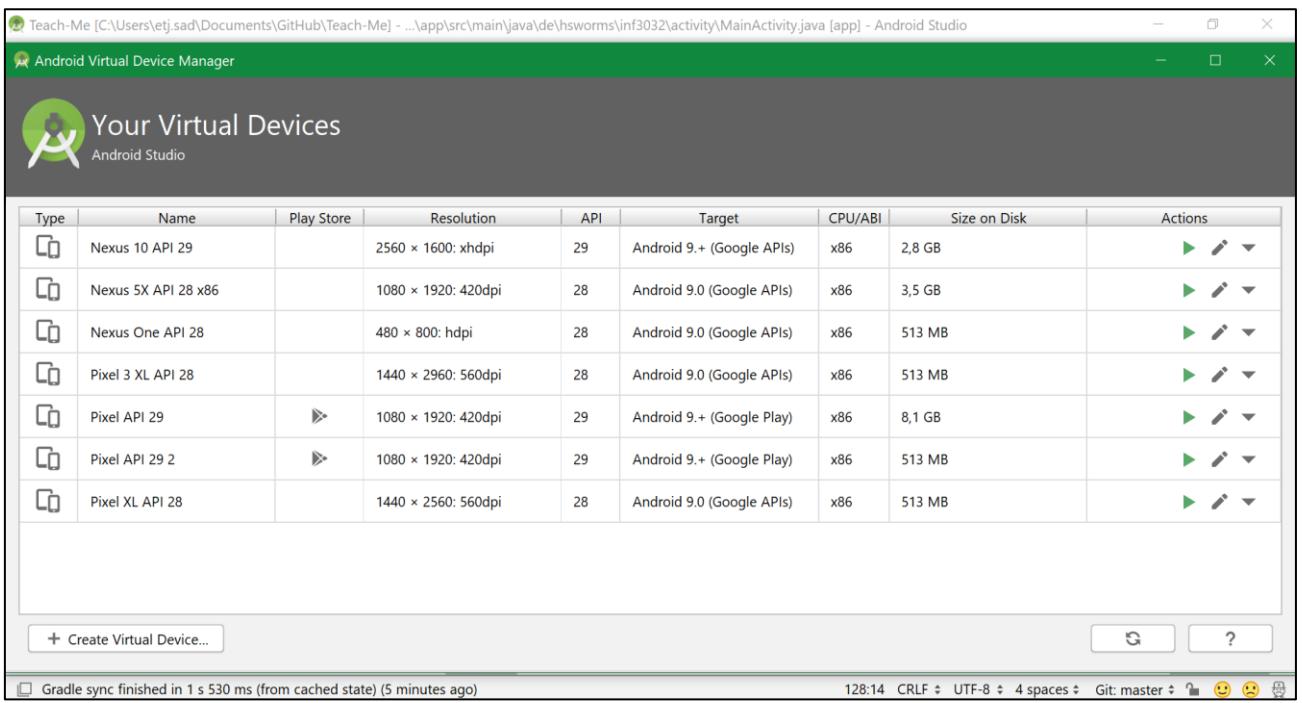


Abbildung 2.3: Auswahl von Geräten im ADV Manager

Android Device Monitor

Der Android Device Monitor ist ein weiteres integriertes Tool des Android Studios. Mit ADM 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. Zudem bekommen sie Netzwerkstatistiken und einen LogCat. Dieses Tool eignet sich hervorragend zum Testen der Anwendungsleistung [21].

Android Debug Bridge

Android Debug Bridge ist ein Befehlszeilentool, mit dem Programmierer Dateien auf und vom Gerät kopieren, Apps installieren und deinstallieren und auf allen Android basierten Geräten Daten sichern und wiederherstellen können [20].

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 ein 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 vielen nützlichen, erweiterten Funktionen wie drahtlosem Debuggen über Bluetooth, einem visuellen Editor und mehreren weiteren Modulen. Abbildung 2.4 zeigt das Umgebungsinterface des B4A [22].

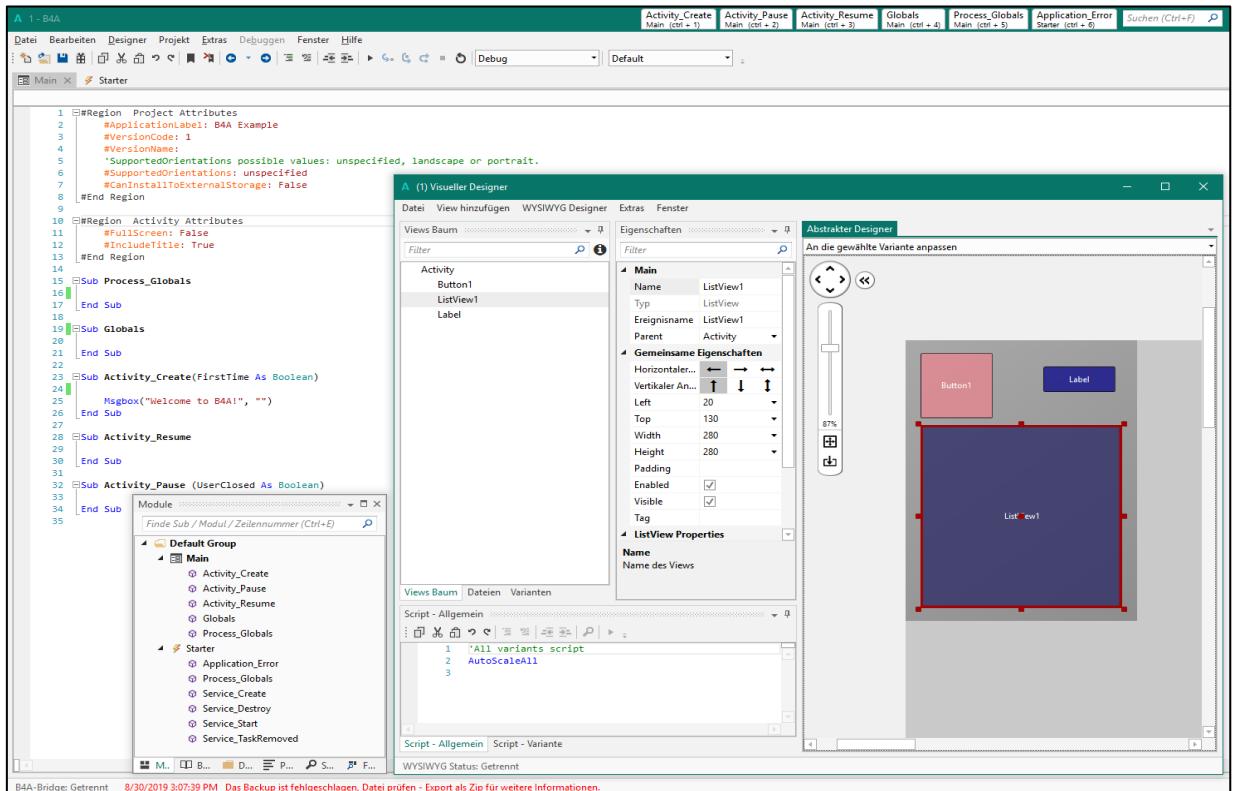


Abbildung 2.4: Interface der IDE Basic for Android

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 sie bestimmte Java-Bibliotheken verwenden möchten, die für Basic for Android entwickelt werden können [22].

2.3.3 Visual Studio

Visual Studio ist Microsofts IDE, die eine Reihe von Entwicklungssprachen unterstützt, einschließlich C#. VB.NET, JavaScript und weitere. Mit dem im 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 sinnvolle und kostenlose Wahl, die App sowohl für Android als auch für iOS zu veröffentlichen und den 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 [23].

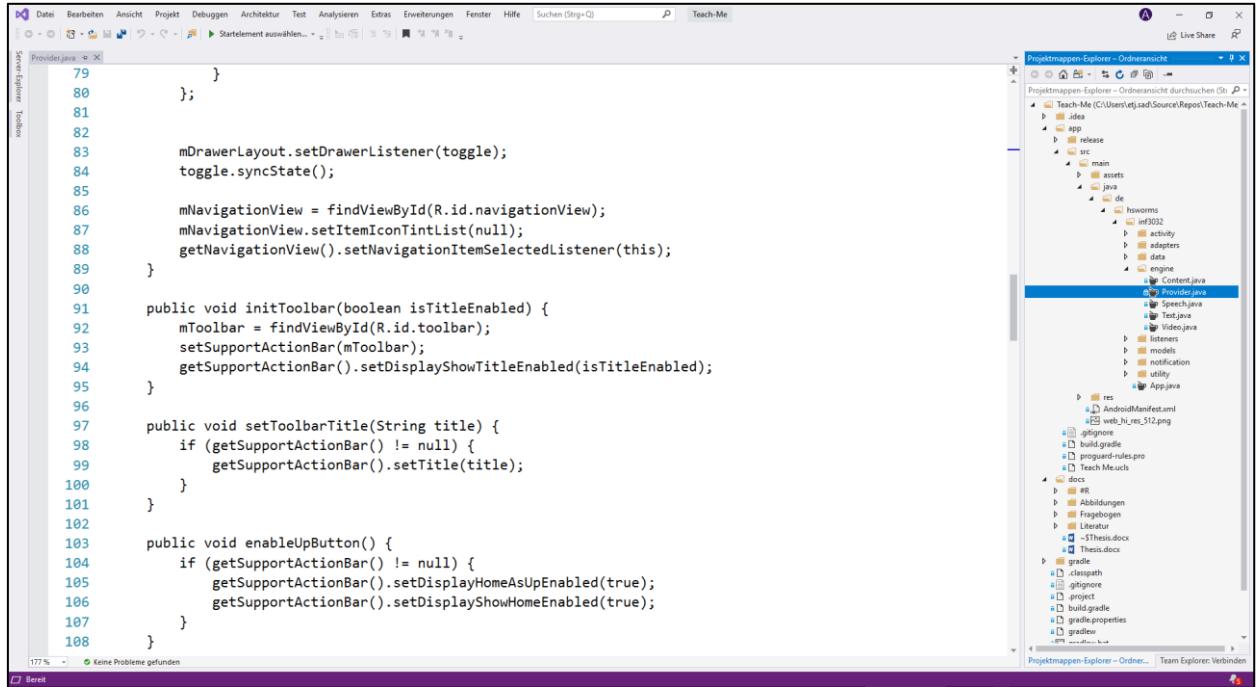


Abbildung 2.5: Interface der IDE Visual Studio

2.3.4 AIDE

AIDE steht für Android IDE und ist insofern einzigartig, weil es auf Android selbst funktioniert. Das bedeutet, dass die Apps auf dem Smartphone oder Tablet erstellt und dann auf demselben Gerät getestet werden können. IDE funktioniert sehr gut mit Samsung DeX. AIDE fehlen einige 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 ihr Informatik Studium erst begonnen haben. Studierende können ihr eigenes, mobiles oder einfaches Java / C++ Projekt entwickeln. AIDE bietet die Möglichkeit, ein vorgeschriebenes Programmierlehrbuch zu lesen und gleichzeitig den Code von dort in Echtzeit zu überprüfen [24]. Die Programmoberfläche ist in Abbildung 2.6 dargestellt.

The screenshot shows the AIDE (Android IDE) interface. At the top, there's a toolbar with icons for file operations like close, save, and more. Below the toolbar, the title bar says "MAINACTIVITY.JAVA". The left side features a file tree with various Java files and resources. The main right pane displays the source code for "MainActivity.java". The code includes imports for Context, Activity, Button, PopupWindow, ArrayList, ContentAdapter, RelativeLayout, ImageButton, RecyclerView, String[], List<String>, and ArrayList<Item>. It also includes a Broadcast Receiver named "newNotificationReceiver". The code is annotated with JavaDoc-style comments and several overridden methods, notably "onReceive".

```
public static Context mContext;
public static Activity mActivity;
public static Button mContentSelectorButton, mLanguageSelectorButton;
public static PopupWindow content_listWindow;
public static PopupWindow language_listWindow;
private static ArrayList<Contents> mContentList;
private static ContentAdapter mAdapter = null;
private RelativeLayout mNotificationView;
private ImageButton mImgBtnSearch, mQuizButton, mInterviewButton;
private RecyclerView mRecycler;
private String[] contentString;
private String[] languageString;
private List<String> contentList;
private List<String> languageList;
private ArrayList<Item> items;
private BroadcastReceiver newNotificationReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        initNotification();
    }
};

public static void loadJson() {
    if (mContentList != null) {
        mContentList.clear();
    }
    ContentLoaderAdapter contentLoaderAdapter = new ContentLoaderAdapter();
    contentLoaderAdapter.loadData();
    parseJson(contentLoaderAdapter.getStringBuffer().toString());
}
```

Abbildung 2.6: AIDE Interface mit geöffnetem Projekt Teach Me

2.4 Android Betriebssystem

Android ist das 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 können Java-Anwendungen erstellt werden, die das Gerät über die von Google entwickelten Bibliotheken steuern. Mit dem Android Native Development Kit können Bibliotheken und Anwendungskomponenten portiert werden, die in C und anderen Sprachen geschrieben sind [20].

Anfang 2019 gelang es Forschungen aus verschiedenen Quellen herauszufinden, dass der durchschnittliche Anteil von Android-Smartphones auf dem Smartphone-Markt im Juli 2019 76,08% betrug. Insgesamt wurden im Jahr 2019 mehr als 344 Millionen Geräte

verkauft [25]. Die Abbildung 2.7 zeigt den Marktanteil der mobilen Betriebssysteme weltweit.

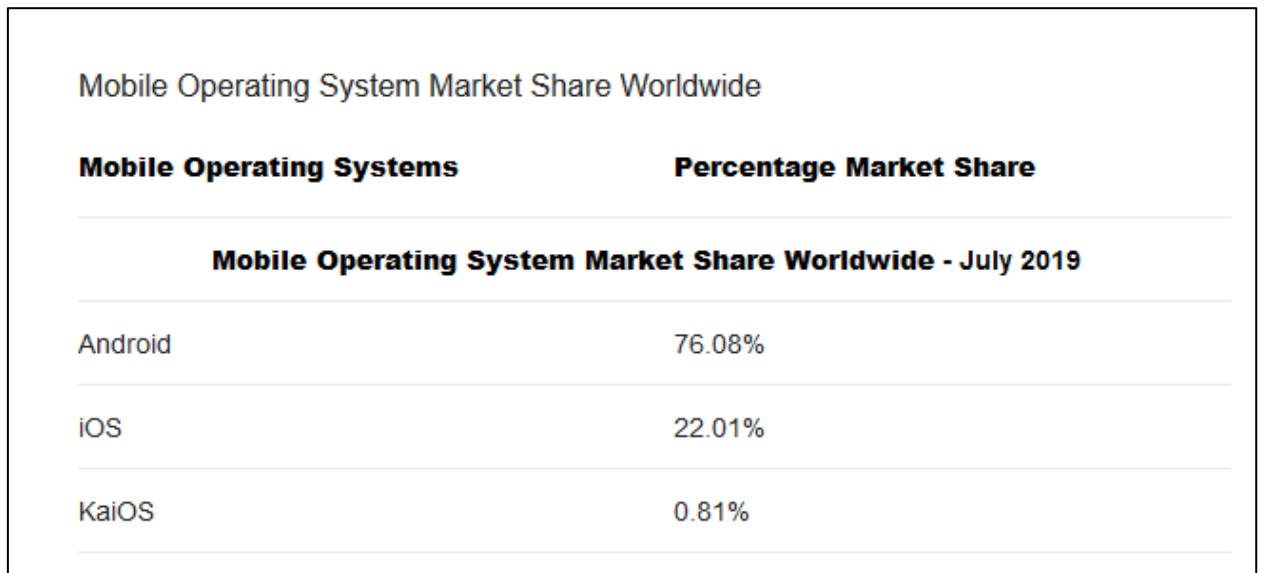


Abbildung 2.7: Marktanteil der mobilen Betriebssysteme weltweit [25]

2.4.1 Anwendungskomponenten

Android-Apps bestehen aus den folgenden Teilen:

- **Activity** ist ein Ansichtsschema für Android-Apps, zum Beispiel für den Bildschirm, den der Benutzer sieht. Die Android-App kann mehrere Aktivierungen haben und kann während der Ausführung der App zwischen ihnen wechseln [20].
- **Views** aktiviert die Benutzeroberfläche, die von Klassen-Widget erstellt wird [20].
- **Services** führt Hintergrundaufgaben aus, ohne eine Benutzeroberfläche bereitzustellen. Entwickler können den Benutzer über das Android-Benachrichtigungssystem benachrichtigen [20].
- Der **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 [20].
- **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 einer 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 [20].

- Der **Broadcast Receiver** akzeptiert Systemnachrichten und implizite Intents und 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 [20].

Der Lebenszyklus einer Anwendung in Android wird vom System streng überwacht und hängt von den Bedürfnissen des Benutzers und den verfügbaren Ressourcen ab. Die Entscheidung, die Anwendung zu starten, trifft das System. Das System unterliegt bestimmten angegebenen und logischen Regeln, mit denen das System bestimmen kann, ob eine Anwendung heruntergeladen, gehalten oder beendet werden kann. Wenn der Benutzer derzeit mit einem bestimmten Fenster arbeitet, bietet das System die entsprechende Anwendung. Umgekehrt wird die Anwendung beendet, sobald das Fenster unsichtbar ist und das System entscheidet, dass die Anwendung beendet werden muss, um zusätzliche Ressourcen zu mobilisieren. In Android sind die Ressourcen begrenzter, da Android die Funktionsweise von Apps strenger überwacht [15].

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.

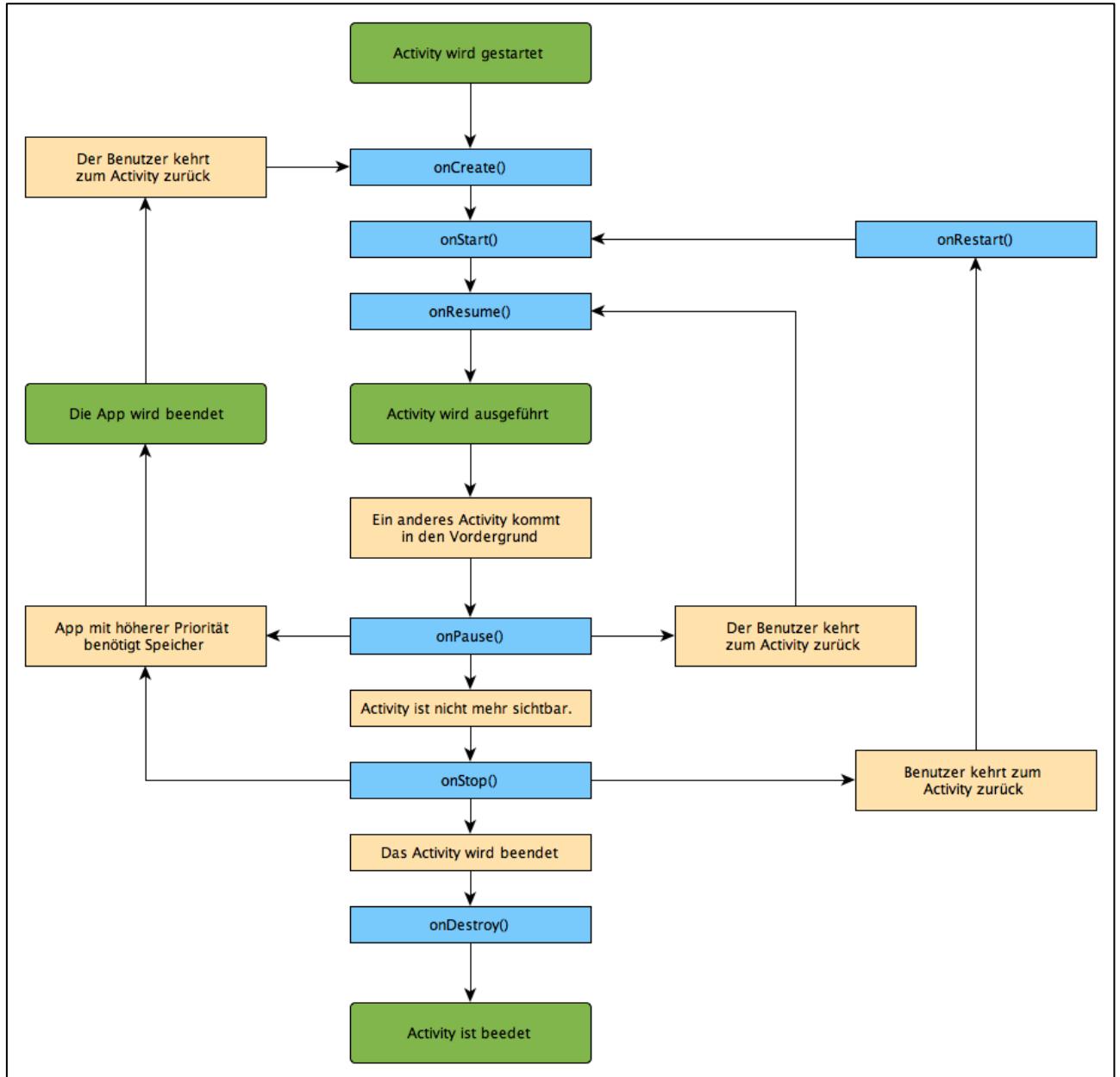


Abbildung 2.8: Lebenszyklus einer Android Anwendung [26]

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.

Model View Presenter

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

- In Model werden die Anwendungsdaten gespeichert. Das Model ist nicht nur für die Speicherung von Anwendungsdaten verantwortlich, sondern besteht auch aus Komponenten, die für das Erstellen, Bereitstellen und Abrufen von Daten verantwortlich sind. Alle diese Funktionen laufen im Hintergrund [27].
- Der Presenter vermittelt zwischen der Verarbeitung von Daten, die von Model und dem Aufruf von Methoden und von View abgerufen werden, und implementiert die Verbindung der UI-Komponenten mit den Daten. Presenter-Methoden werden von Activity/Fragment- Lebenszyklusmethoden aufgerufen und sind oft symmetrisch [27].
- 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 der Presenter von ihm verlangt. View kann jede Aktivität oder jedes Fragment in der App unter Android Betriebssystem sein [27].

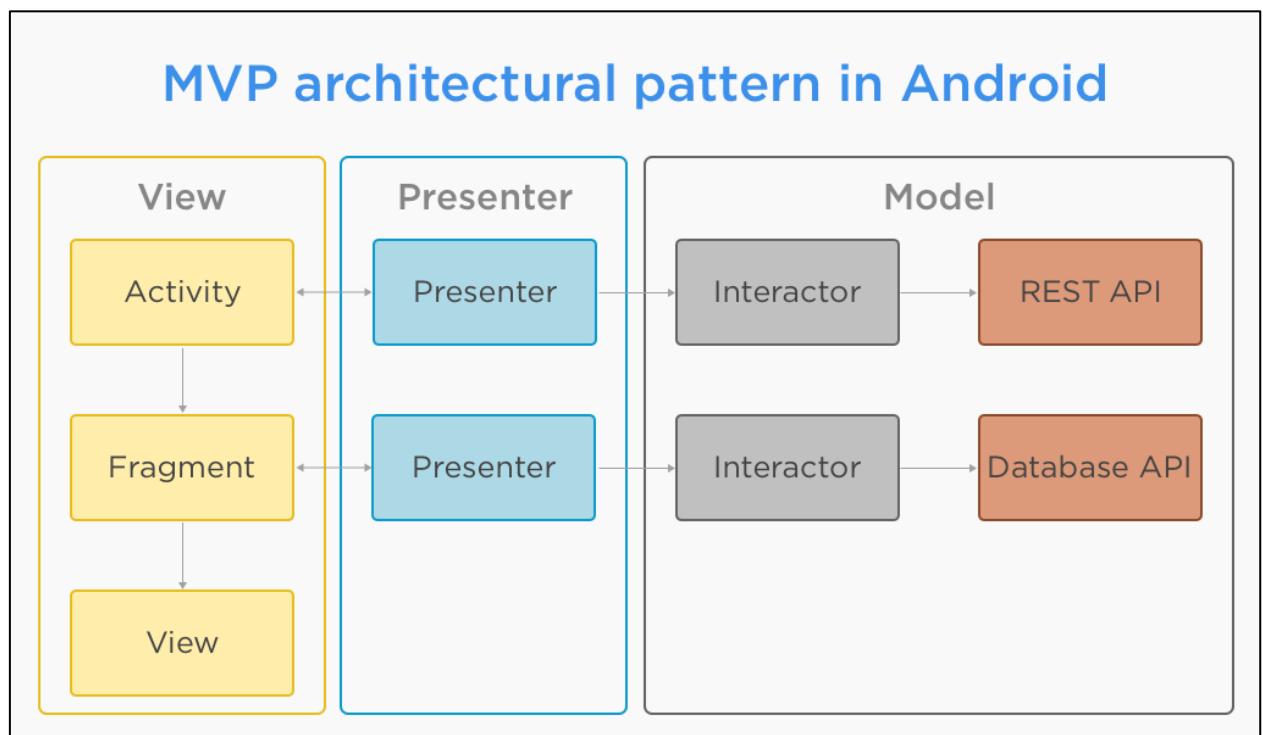


Abbildung 2.9: MVP Pattern [27]

Der Model View Presenter sollte Schnittstellen für mehr Flexibilität bei der Codeänderung darstellen. MVP hat eine Reihe von Vorteilen gegenüber dem Standard App Entwicklungsschema, 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. Mit MVP ist es auch möglich, einen ziemlich einfachen und verständlichen Code zu erreichen [27].

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.

Die Diplomarbeit nutzt den 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 für Projekte mit dem Git-Versionskontrollsystem als auch ein soziales Netzwerk für Entwickler. Benutzer können eine unbegrenzte Anzahl von Repositoryn erstellen, von denen jedes ein Wiki, ein Issues-tracking-System und eine Möglichkeit zur Durchführung von Code Review bietet [28].

2.6 Auswahl einer Technologie

Bei der Entwicklung der Anwendung stellte sich die Frage nach der Wahl der Sprache, der IDE und der 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, zu ändern, zu testen, zu debuggen und zu verstehen.

Die Anwendungsarchitektur muss folgende Hauptkriterien erfüllen:

- Die Flexibilität des Systems – die Veränderung eines Fragments des Systems darf seine anderen Fragmente nicht beeinflussen und die Folgen der architektonischen Fehler sollten in einem angemessenen Umfang begrenzt sein.

- Die Erweiterbarkeit des Systems – die Möglichkeit, dem System neue Entitäten und Funktionen hinzuzufügen, ohne seine Hauptstruktur zu stören. Die Architektur sollte es ermöglichen, zusätzliche Funktionen bei Bedarf leicht zu erweitern.
- Die Testbarkeit – ein Code sollte einfach zu testen sein, weniger Fehler enthalten und zuverlässig funktionieren. Aber Tests verbessern nicht nur die Qualität des Codes. Viele Entwickler kommen zu dem Schluss, dass die Anforderung einer guten Testbarkeit ein wesentlicher Faktor dafür ist, um zu einem guten Design zu führen.

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 im Android Studio kombiniert, die alle bei der Entwicklung der Anwendung Teach Me verwendet wurden.

3 Tutorien an den deutschen Universitäten

Viele deutsche Universitäten und Fachhochschulen kombinieren die Tutorien mit Übungsgruppen und für viele Studenten ist Anwesenheit bei den Übungsgruppen Pflicht. Um eine Teilnahmebestätigung für die Tutorien oder die Übungsgruppe zu erhalten, müssen Studierenden 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 zugeteilt.

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

The screenshot shows the MÜSLI interface for the 'Mathematisches Übungsgruppen- und Scheinlisten-Interface'. At the top, there is a yellow header bar with the MÜSLI logo and the title. Below it is a navigation bar with links: Übersicht, Vorlesungen, Angaben ergänzen, E-Mail ändern, Passwort ändern, API, Abmelden, Kontakt, and Neues. The main content area displays information for the course 'Einführung in die Theoretische Informatik'. It includes a link to the internet page: https://www.math.uni-heidelberg.de/logic/ss19/theoinf_ss19.html. A section titled 'Übungsgruppen' lists available tutorials:

Zeit	Raum	Auslastung	Tutor	Kommentar
Mo 14:00	SR Statistik 02.104	<div style="width: 22%; background-color: #000; height: 10px;"></div> 22/25	Patrick Arras	Beitreten
Mo 16:00	Hörsaal Mathematikon	<div style="width: 25%; background-color: #000; height: 10px;"></div> 25/25	Markus Schäfers	Beitreten
Mo 16:00	SR Statistik 02.104	<div style="width: 21%; background-color: #000; height: 10px;"></div> 21/25	Tim Karl	Beitreten
Do 14:00	SR 9	<div style="width: 12%; background-color: #000; height: 10px;"></div> 12/25	Tim Karl	Beitreten
Do 14:00	SR Statistik 02.104	<div style="width: 23%; background-color: #000; height: 10px;"></div> 23/25	Patrick Arras	Beitreten
Do 16:00	SR 1	<div style="width: 25%; background-color: #000; height: 10px;"></div> 25/25	Markus Schäfers	Beitreten
So 23:59	Mare ingenii	<div style="width: 6%; background-color: #000; height: 10px;"></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 [29]

Der Fachbereich der Physik, der Mathematik und der Informatik der Johannes-Gutenberg-Universität Mainz bietet auch für Informatik-Studierende eine Website des Instituts für Informatik an. Auf der Website können die Studenten die Vorlesungen und Übungsgruppen auswählen. Die Anmeldung für die Übungsgruppen erfolgt durch JOGU-StInNe. JOGU-StInNe ist ein offizielles Studentenportal der Johannes-Gutenberg-Universität. Dort können sich Studenten nicht nur für die Übungsgruppen zu den Vorlesungen anmelden, sondern auch ihre Leistungen oder Studienbescheinigungen bekommen. Die Abbildung 3.2 stellt das Vorlesungsverzeichnis für das Wintersemester 19/20 von JOGU-StInNe dar.

Vorlesungsverzeichnis	
Übersicht > Fachbereich 08 - Physik, Mathematik und Informatik > Informatik > Vorlesungen	
Veranstaltungen / Module	Veranstaltungsort Raum
Veranstaltung / Modul Dozenten / Modulverantwortliche Zeitraum	
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 in JOGU-StInNe [30]

Die Hochschule Worms hat auch ein 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 [31]

3.1 Sinn und Zweck von Tutorien an Universitäten

Ein Tutorium ist eine Veranstaltung, die von einer Lehrperson im Rahmen eines Kurses oder einer Unterrichtsveranstaltung angeboten wird. Studenten können den Unterrichtsstoff der Lehrveranstaltung noch einmal mit einem Tutor wiederholen und vertiefen. Tutorien sind im Regelfall freiwillig, können aber auch von den Lehrpersonen 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 in einem höheren Fachsemester. Ein Tutor hilft der Lehrperson als studentische Hilfskraft und unterstützt mit seinem Wissen andere Studenten insbesondere niedriger Fachsemester [32].

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 beinhalten. Die letztere Variante ist in Großbritannien weit verbreitet. Man nennt sie das Personal Tutoring System, dabei wird jedem Studierenden ein Hochschullehrer, der ihm als Ansprechperson dienen soll, zugewiesen [33]. Im Gegensatz dazu wird in Deutschland wegen der hohen Studierendenzahlen eher auf Gruppenbetreuung gesetzt. Dazu werden in Deutschland fast ausschließlich studentische Tutoren eingesetzt. Tutorien unterteilen sich in zwei Arten: Orientierungstutorien und Fachtutorien / Übungsgruppen.

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 in die Hochschule, ins Studienfach und in den Hochschulort. Diese Einführungen können sich an Studienfänger im Allgemeinen oder aber auch nur an bestimmte Zielgruppen richten. Viele Anforderungen sind bei einem Start ins Studium auszubilden. Zum einen muss der Studierende nun lernen, für seine akademischen Leistungen selbst verantwortlich zu sein[34].

Zum anderen muss er auch lernen, sich außerhalb der Universität zurechtzufinden, 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.

3.2.2 Fachtutorien und Übungsgruppen

Unterrichtsveranstaltungen werden meistens im ersten Studienjahr von Fachtutorien oder Übungsgruppen begleitet. Ein Fachtutorium oder eine Übungsgruppe ermöglicht es den Studenten, erforderliche Grundfähigkeiten für das Studium zu erwerben, weiter auszubauen und auch zu trainieren. Viele Lernangebote erfordern eine Lernselbstständigkeit des Studierenden, ohne dass man Lernziele auszuformulieren braucht. Wie ein Tutorium abläuft, wird von der jeweiligen Lehrperson selbst bestimmt. Eine enge Zusammenarbeit zwischen Lehrperson und Tutor ist dabei unablässig, damit das Tutorium so gut wie möglich an die Lernveranstaltung angepasst ist [34].

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 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. Studenten sollten nicht mehr mit

ihren Fragen und Problemen daheim allein gelassen werden, sondern in den Übungsgruppen die Chance haben, von ihrem Tutor betreut zu werden. Tutoren müssen neben ihrer eigenen Lehrtätigkeit auch bei den Vorbereitungen von Veranstaltungen helfen, die Lehrperson unterstützen und ausländische Studenten betreuen. [35].

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 [36].
- **Lernbegleitung:** Tutoren führen sowohl in Inhalte als auch in Arbeitsformen ein. Tutoren stellen, wo nötig, Arbeitsmittel zur Verfügung und sorgen für die Lösung von Raumproblemen [36].
- **Moderation:** Die Moderation kann sowohl bei der Erarbeitung oder Diskussion von Inhalten, bei Absprachen, als auch bei zwischenmenschlichen Störungen und Konflikten sinnvoll eingesetzt werden [36].
- **Information:** Alle notwendigen Informationen müssen immer durch den Tutor zur Verfügung gestellt werden. Das können Informationen des Professors, Informationen zu den Klausurterminen oder zur Benotung sein.
- **Rückmeldung geben:** Sowohl Anerkennung als auch nachvollziehbare Kritik gehören zu den Aufgaben eines Tutors [36].

Die Abbildung 3.4 zeigt, wie der Aufbau eines Tutoriums aussieht.



Abbildung 3.4: Der Aufbau eines Tutoriums [37]

4 Konzeption und Entwurf

Als erstes werden im 4. Kapitel die funktionalen, 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. Die 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? In den Tabellen 1 und 2 wird beschrieben, welche funktionalen 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 die Möglichkeit haben, verschiedene Lernmodule zur Auswahl zu haben.
F.2	Suche	Der Nutzer soll die Möglichkeit haben, den Inhalt durch ein Begriffswort zu suchen.
F.3	Quizfragen	Der Nutzer soll die Möglichkeit haben, Quizfragen zu beantworten.
F.4	Interviewfragen	Der Nutzer soll die Möglichkeit haben, die Interviewfragen zu beantworten.

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

Nº	Aufgabe	Beschreibung
F.5	Video	Der Nutzer soll die Möglichkeit haben, den Videolernstoff anzuschauen.
F.6	Textgröße	Der Nutzer soll die Möglichkeit haben, die Textgröße des Inhalts zu ändern.
F.7	Sprachauswahl	Der Nutzer soll die Möglichkeit haben, den Inhalt in einer anderen Sprache zu lernen.
F.8	Favoriten	Der Nutzer soll die Möglichkeit haben, den Inhalt zu seinen Favoriten hinzuzufügen.
F.9	Textvorlesen	Der Nutzer soll die Möglichkeit haben, sich den Inhalt vorlesen zu lassen.
F.10	Inhaltkopieren	Der Nutzer soll die Möglichkeit haben, den Inhalt zu kopieren.
F.11	Einstellungen	Der Nutzer soll die Benachrichtigungen an- und ausschalten können.

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

4.1.1 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 die Möglichkeit haben, den Lerninhalt und Quizergebnisse zu teilen.
0.2	Breitbild	Der Nutzer soll die Möglichkeit haben, den Inhalt im Breitbild angezeigt zu bekommen.
0.3	Ton	Die Quizfragen sollen mit Ton begleitet werden.
0.4	Quizergebnisse	Der Nutzer soll die Möglichkeit haben, die Quizergebnisse anzusehen.
0.5	Benachrichtigung	Der Nutzer soll die Möglichkeit haben, Benachrichtigungen vom Entwickler zu bekommen.

Tabelle 3: Optionale Anforderungen

4.1.1 Nicht-Funktionale Anforderungen

Nichtfunktionale Anforderungen bilden eine Klasse von Anforderungen, die aufgrund Ihrer allgemeinen Bedeutung häufig projektübergreifend eingesetzt werden. Dabei werden die Qualitätsattribute der Funktionen, die Anforderungen an die Anwendung als Ganzes, die Anforderungen an die Implementierung der Systemerstellung und die Anforderungen an Tests, an die Einführung, den Support und den Betrieb untersucht.

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 sollten einheitlich und einfach aufgebaut sein.
NF.2	Usability-Ziele einhalten	Die App sollte einfach und verständlich aufgebaut sein.
NF.3	Verfügbarkeit	Die App sollte in jeder Auflösung darstellbar sein. Beispielweise sollte die App auch für Tablets angeboten werden.
NF.4	Selbsterklärbarkeit	Die App sollte selbsterklärend aufgebaut sein. Nutzer sollten 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 Mockup

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[13]. In Abbildung 4.1, 4.2, und 4.3 sind erste Entwürfe der App Teach Me abgebildet.

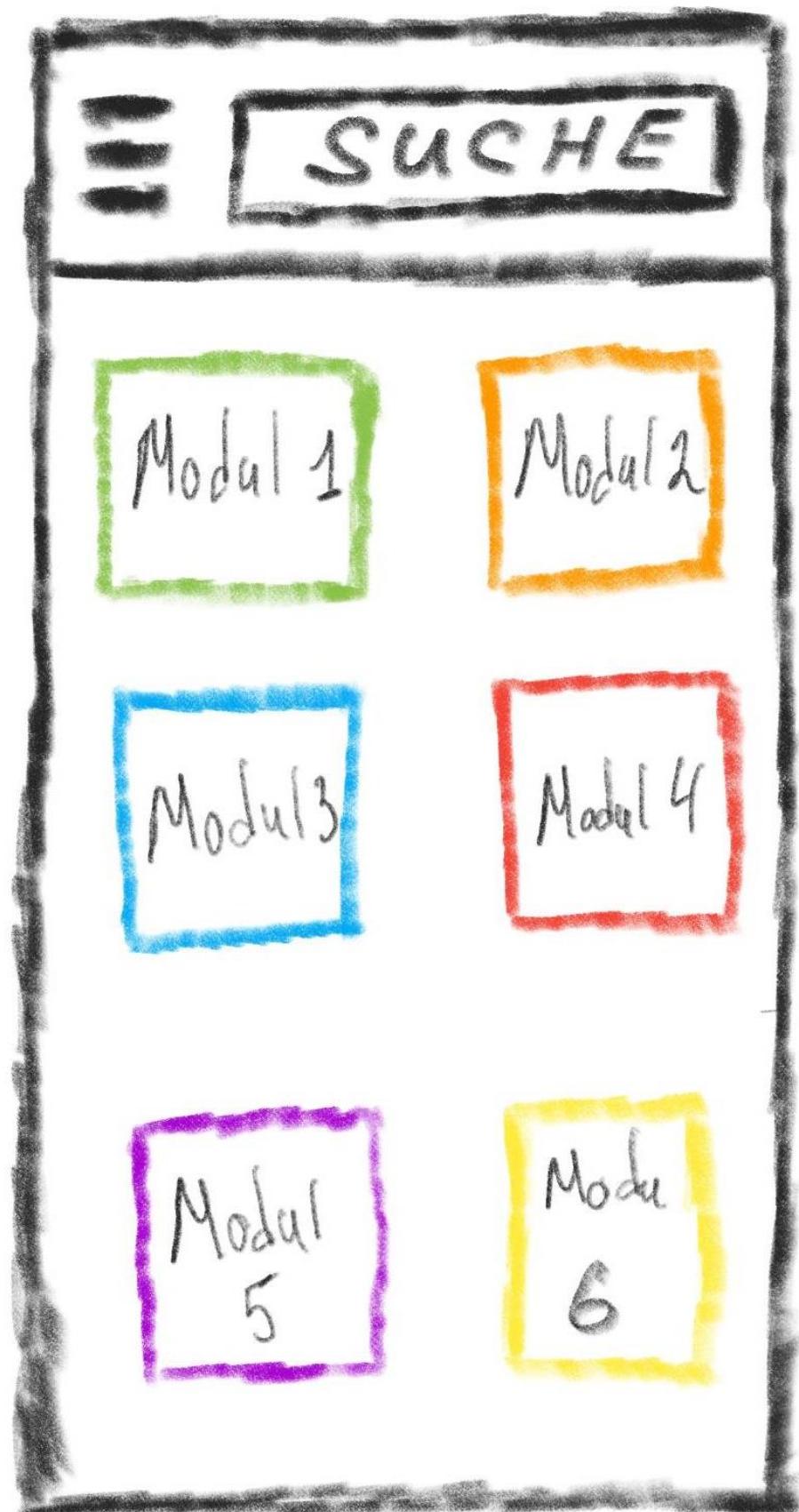


Abbildung 4.1: Mockup des Hauptbildschirms

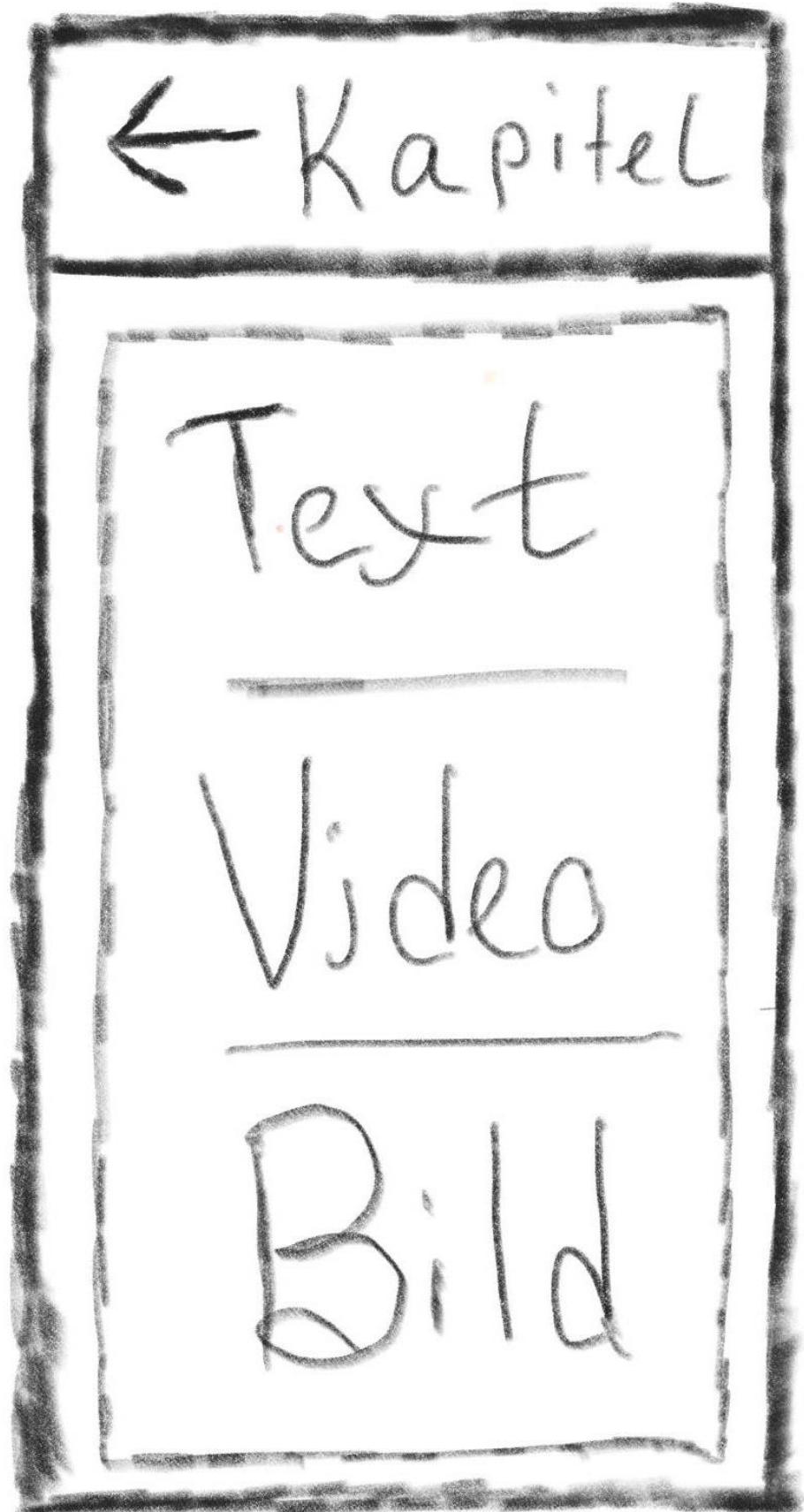


Abbildung 4.2: Mockup der Kapitelansicht

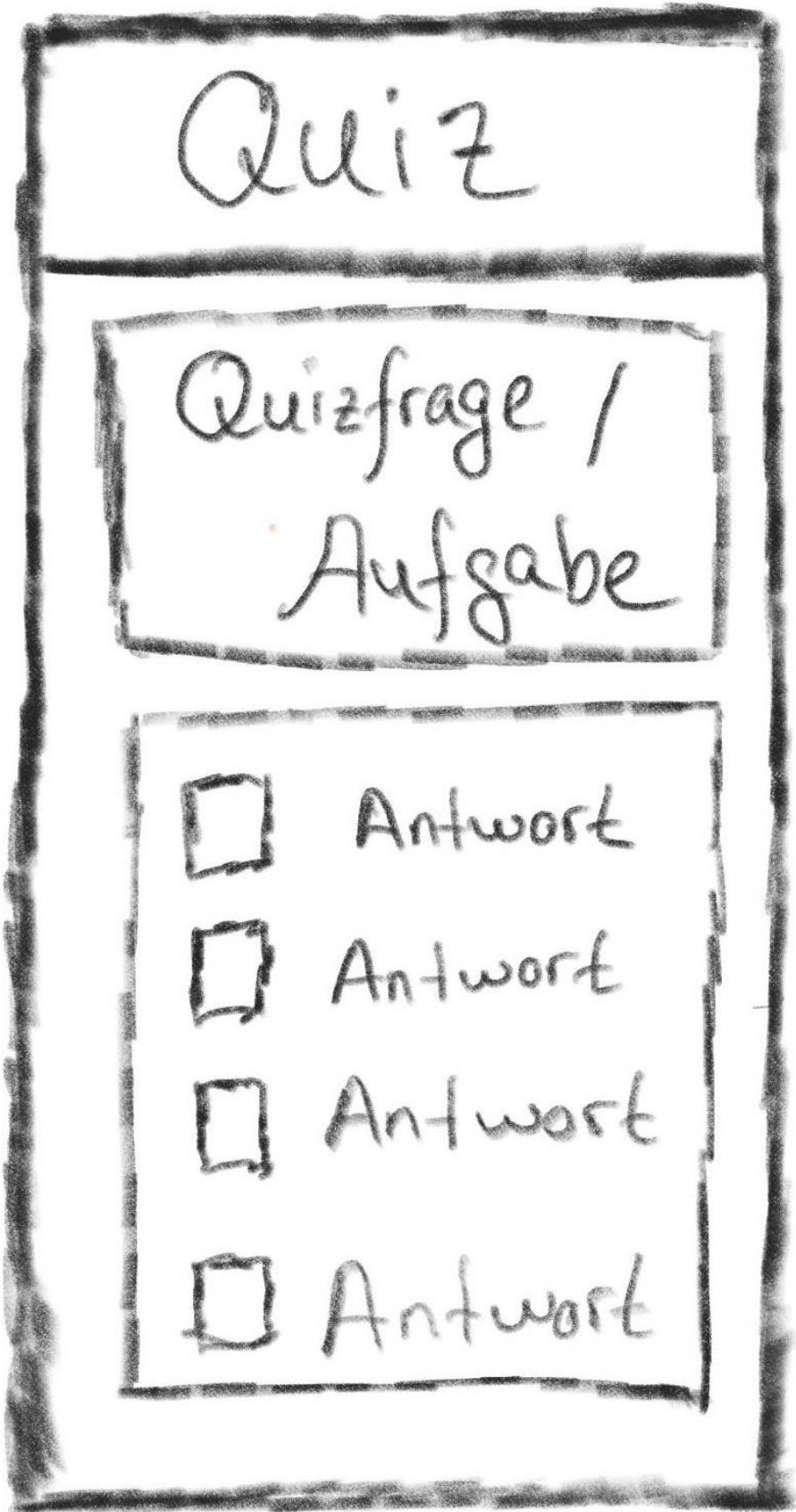


Abbildung 4.3: Mockup des Quizbildschirms

4.3 Prototyp

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 IDE 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 [13].

In den Abbildungen 4.4, 4.5 und 4.6 sind die Prototypen des Hauptbildschirms, der Kapitelansicht und der Quizfragen der App Teach Me zu sehen.

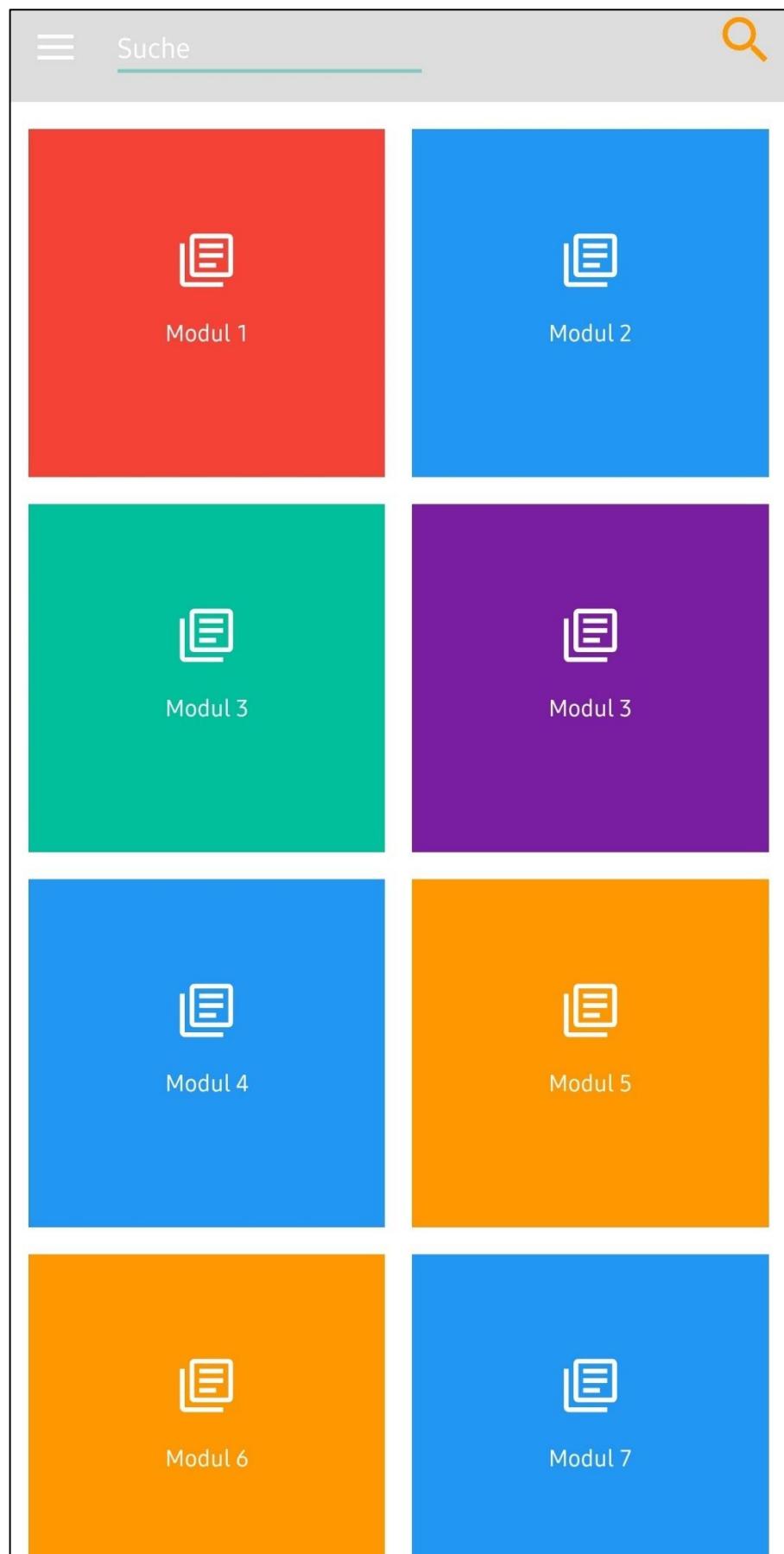


Abbildung 4.4: Prototyp des Hauptbildschirms der App Teach Me



Abbildung 4.5: Prototyp der Kapitelansicht der App Teach Me

Quizaufgabe. Quizaufgaben werden mit HTML Sprache geschrieben.

Beispielsweise:

Was ist die Ausgabe des folgenden Programms?

```
#include<iostream>
```

```
using namespace std;
main() {
    int i = 13, j = 60;
    i^=j;
    j^=i;
    i^=j;
    cout<<j<<" "<<j;
}
```

73 73

60 13

13 60

60 60

NÄCHSTE FRAGE

Abbildung 4.6: Prototyp des Quizfragenbildschirms der App Teach Me

5 Implementierung

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

5.1 Datenarchitektur

Alle Inhaltsdaten und Quizfragen der App Teach Me werden in JSON Objekt 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 [36]. Inhaltsdaten und Quizdaten werden mit Hilfe der Android-Komponente WebView dargestellt. WebView ist eine Komponente von Android zur Darstellung von JSON, HTML oder Video Dateien.

Inhaltsdaten

App Inhalte werden in JSON Objekt gespeichert. JSON Objekt besteht aus vielen Arrays. Die Arrays unterteilen sich in 3 Variablen:

- title: Beschreibung der Titel des Moduls.
- qType: Beschreibung des Inhaltstyps. Diese Variable wurde nur im Debug Modus verwendet.
- content: Hier wird der Inhalt des Moduls in HTML Sprache geschrieben. Content unterteilt sich auch in 3 weitere Variablen:
 - tag_line: Beschreibung der Titel des Kapitels.
 - qType: Beschreibung des Inhaltstyps. Diese Variable wurde nur im Debug Modus verwendet.
 - Details: Inhalt des Kapitels.

In Abbildung 5.1 ist ein Teil-Code aus der deutschen Version der JSON-Inhaltsdaten der App Teach Me zu sehen.

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

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

Herk&ouml;mliche Softwareentwicklung bestand oftmals
16: darin, zur L&ouml;sung 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:     ],
28:   }
29: }


```

Abbildung 5.1: Teil-Code aus CS_GermanContentFile.json

Quizdaten

App Quizfragen werden in JSON Objekt gespeichert, der nur aus einem Array *questionnaires* besteht. JSON Array *questionnaires* besteht aus vielen Arrays. Arrays unterscheiden sich auch aus 3 Variablen:

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

Abbildung 5.2 zeigt einen Teil-Code aus den deutschen Quizfragen des Moduls C++.

```
1: {
2:   "questionnaires": [
3:     {
4:       "question": " Ein Trigraph beginnt mit",
5:       "answers": [
6:         "#",
7:         "##",
8:         "?",
9:         "??"
10:      ],
11:      "correct_answer": 2
12:    },
13:    {
14:      "question":
15:        "Was ist die Ausgabe des folgenden Programms?
16:        <p>#include<iostream><br />\nusing namespace std;<br />\nclass abc
17:        {</p>\n\n<p>&nbsp;&nbsp; public:<br />\n&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
18:        int i;</p>\n\n<p>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; i = i;<br
19:        />\n&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; }<br />\n};</p>\n<p>main() {<br
20:        />\n&nbsp;&nbsp; abc m(5);<br />\n&nbsp;&nbsp;<br />\n&nbsp;&nbsp;
21:        cout<i;<5;<br />\n</p>\n",
22:      "answers": [
23:        "5",
24:        "Garbage",
25:        "Error at the statement i=i;",
26:        "Compile error: 'i' declared twice."
27:      ],
28:      "correct_answer": 1
29:    },
30:  },
```

Abbildung 5.2: Teil-Code aus QA_GERMAN_C++.json

Inhaltparser

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

Quizparser

Für Quizparser ist die Klasse QuizParserAdapter zuständig. Die Abbildung 5.4 liefert einen Teilabschnittcode aus der oben genannten Klasse. Die Codezeilen 1-5 erzeugen ein Array. Ab den Codezeile 6 bis 30 wird eine For-Schleife durch die Arraylänge durchgeführt. In der Schleife wird mit Hilfe der Schlüsselwörter wie `JSON_KEY_QUESTIONNAIRY`, `JSON_KEY_QUESTION`, `JSON_KEY_CORRECT_ANS` und `JSON_KEY_ANSWERS` der Quizinhalt aus den Quizdaten abgelesen. Danach wird der Quizinhalt zum Quizmodel eingefügt und

auf einem Quizfragenbildschirm angezeigt. Im Anhang A.2 befindet sich der gesamte Quellcode der QuizParserAdapters.

```
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 ist das erste, dass der Nutzer nach dem Begrüßungsbildschirm sieht. Auf dem Hauptbildschirm kann der Nutzer verschiedene Lernmodule auswählen oder sich zu Quizfragen, Interviewfragen oder einfach zur Suche navigieren. Der Nutzer kann auch den Studiengang und die Sprache wechseln. Bei jedem neuen Abruf des Hauptbildschirms wechselt das Farbschema der Module wahllos. Die Abbildung 5.5 zeigt das Interface des Hauptbildschirms.

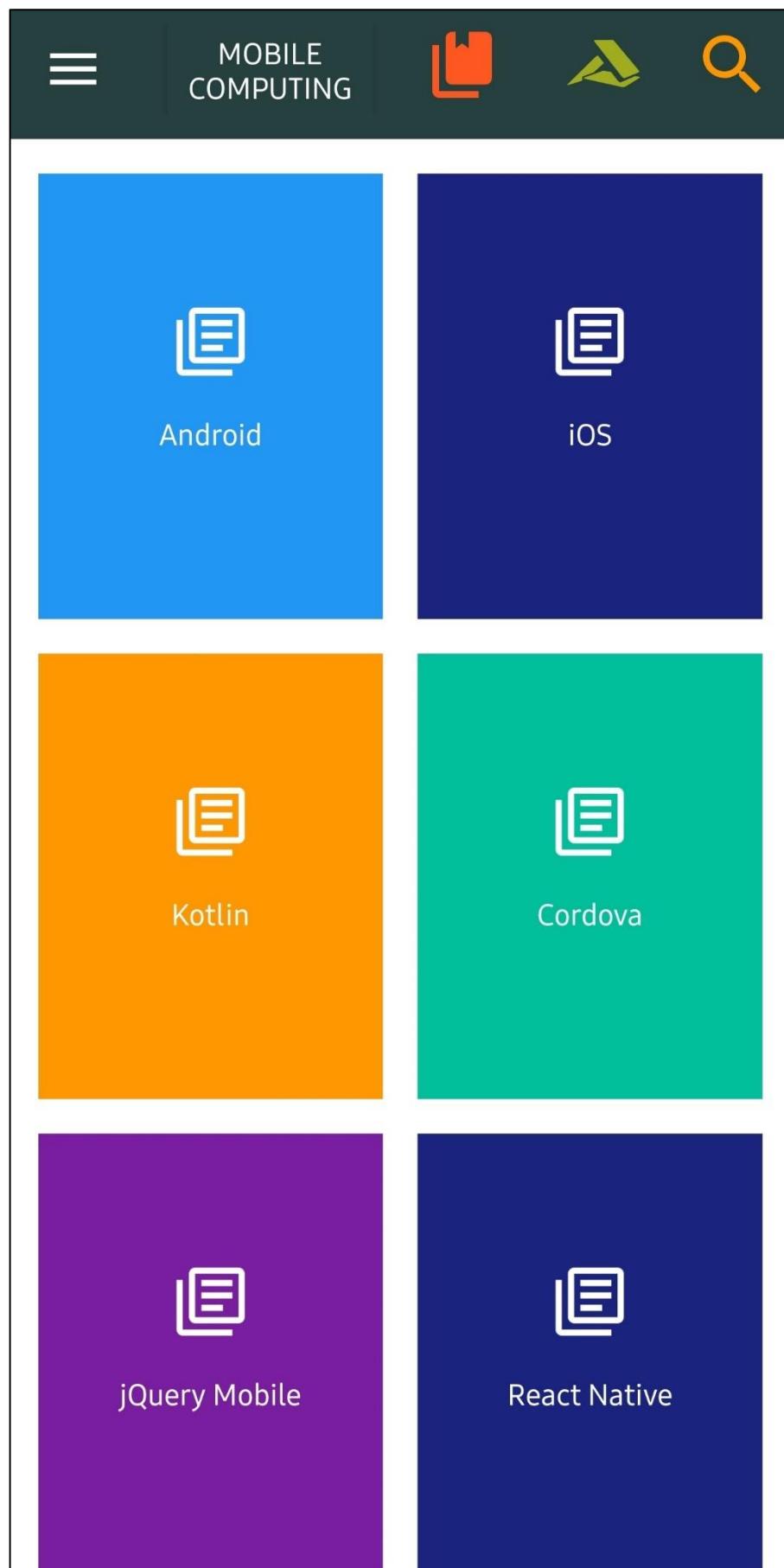


Abbildung 5.5: Hauptbildschirm der App Teach Me

Die Abbildung 5.6 zeigt eine Auswahl von Kapiteln des geöffneten Moduls: Grundlagen der Betriebssysteme der App Teach Me. Wie bereits auf dem Hauptbildschirm wechselt hier das Farbschema zufällig.

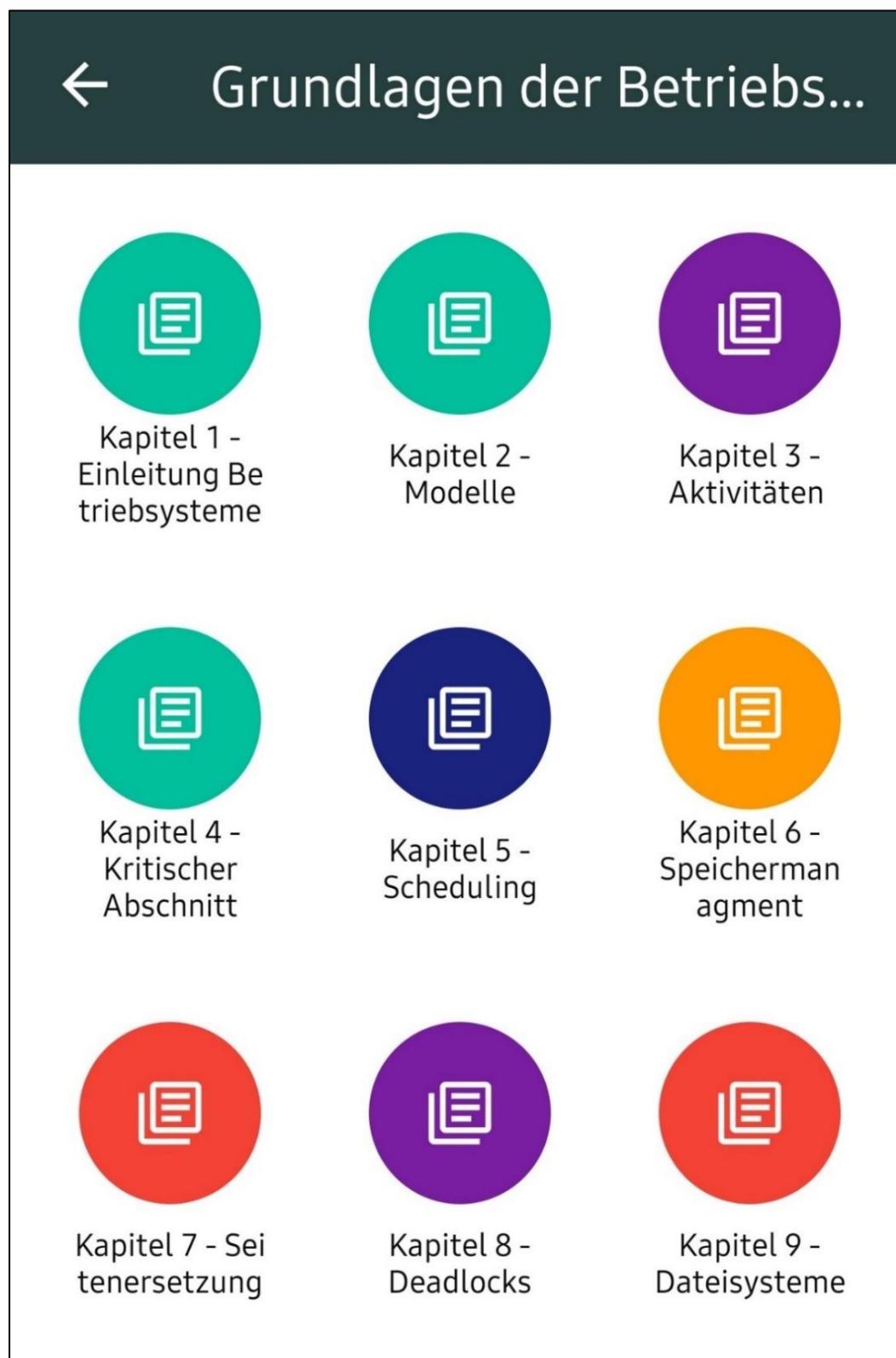


Abbildung 5.6: Die Auswahl von Kapiteln der App Teach Me

Durch die Navigation in den Modulen kann der Nutzer Kapitel auswählen. Im Kapitelbereich kann der Nutzer ein Kapitel favorisieren, Inhalte des Kapitels kopieren oder teilen und sich das Kapitel vorlesen lassen. Der Nutzer kann sich auch zurück zu dem gewählten Modul navigieren. Abbildung 5.7 liefert eine Darstellung der Kapitelansicht.

The screenshot shows a mobile application interface for a chapter titled "Kapitel 5 - Scheduling". At the top, there is a dark header bar with navigation icons: a back arrow, three dots, a heart, a square, a double arrow, the page number "1 / 1", and a vertical ellipsis. The main content area has a white background. The chapter title "Kapitel 5 - Scheduling" is displayed in large, bold, black font. Below the title is a text block in black font, which reads: "Scheduling wird notwendig, wenn die Anzahl der gleichzeitig laufenden Prozesse die Anzahl der physikalischen Prozessoren übersteigt. Die Verwaltung kann dezentral (Prozesse selbst) oder zentral erfolgen. Es gibt zwei grundlegende Modelle für zentrale Scheduling-Verwaltung:". Underneath this text is a bulleted list in black font: • Deterministisches Modell - alle Informationen bekannt (Anzahl Prozesse/Betriebsmittel/Zeitdauer) • Probabilistisches Modell - offen (Anzahl Prozesse nicht bekannt) oder geschlossen (Anzahl Prozesse bekannt, aber Bedienzeiten nicht). Below this list is another section title "Strategien zur Prozessorzuteilung" in bold black font. Underneath this title is another bulleted list: • Durchsatz (Anzahl bearbeiteter Prozesse pro Zeiteinheit maximieren)

Abbildung 5.7: Kapitelansicht der App Teach Me

5.2.2 Suche

Durch die Navigation des Hauptbildschirms kann der Nutzer die Suchfunktion der App nutzen. Die Suche erfolgt nach Eingabe eines Begriffswortes und liefert alle Ergebnisse aus den Modulen beziehungsweise Kapiteln. Abbildung 5.8 liefert eine Darstellung der Suchergebnisse mit dem Begriffswort shell.



Abbildung 5.8: Begriffssuche in der App am Beispiel des Suchwortes: shell

5.2.3 Quizfragen

Die 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 die Möglichkeit, eine Frage zu überspringen. Richtige Antworten werden mit grüner Farbe und einem dazugehörigen Ton unterstrichen, während die falschen Antworten mit Rot markiert und ebenfalls mit einem passenden Ton begleitet werden. Nach Beendigung der Quizfragen ist es möglich, sich die Ergebnisse anzuschauen. Dort kann der Nutzer die Statistik seines Quizzes ansehen, indem er Antworten auf seine Fragen erhält. Der Nutzer kann die Ergebnisse teilen oder das Quiz noch einmal wiederholen. Die Abbildung 5.9 zeigt das Interface des Quizbildschirms.

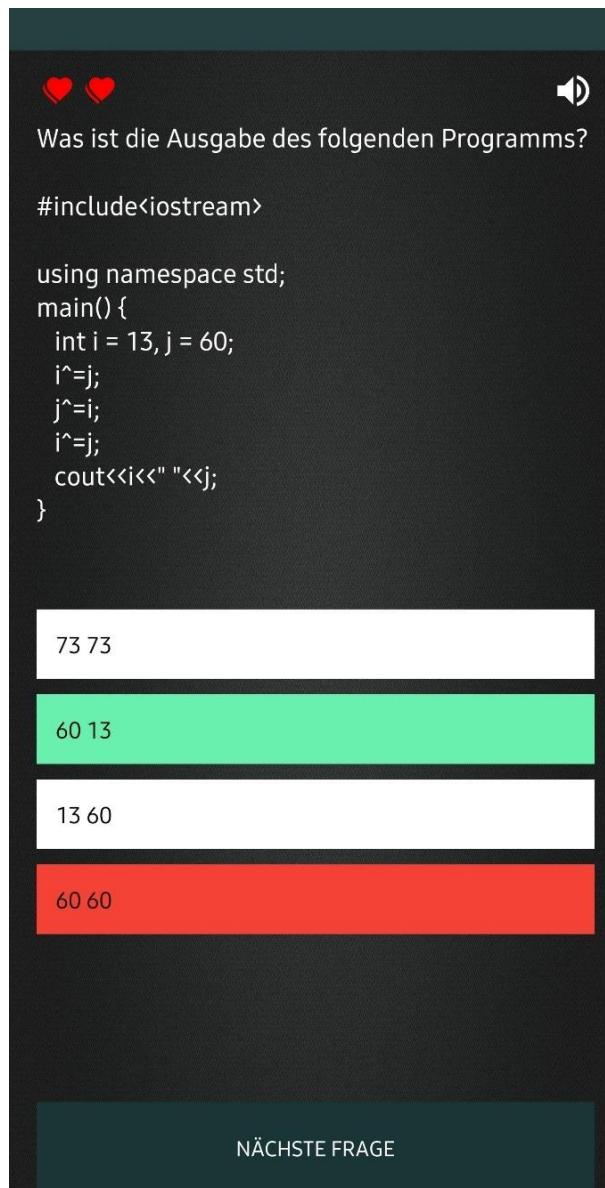


Abbildung 5.9: Quizfragenbildschirm

Die Abbildung 5.10 zeigt das Interface des Quizfragenergebnisbildschirms.

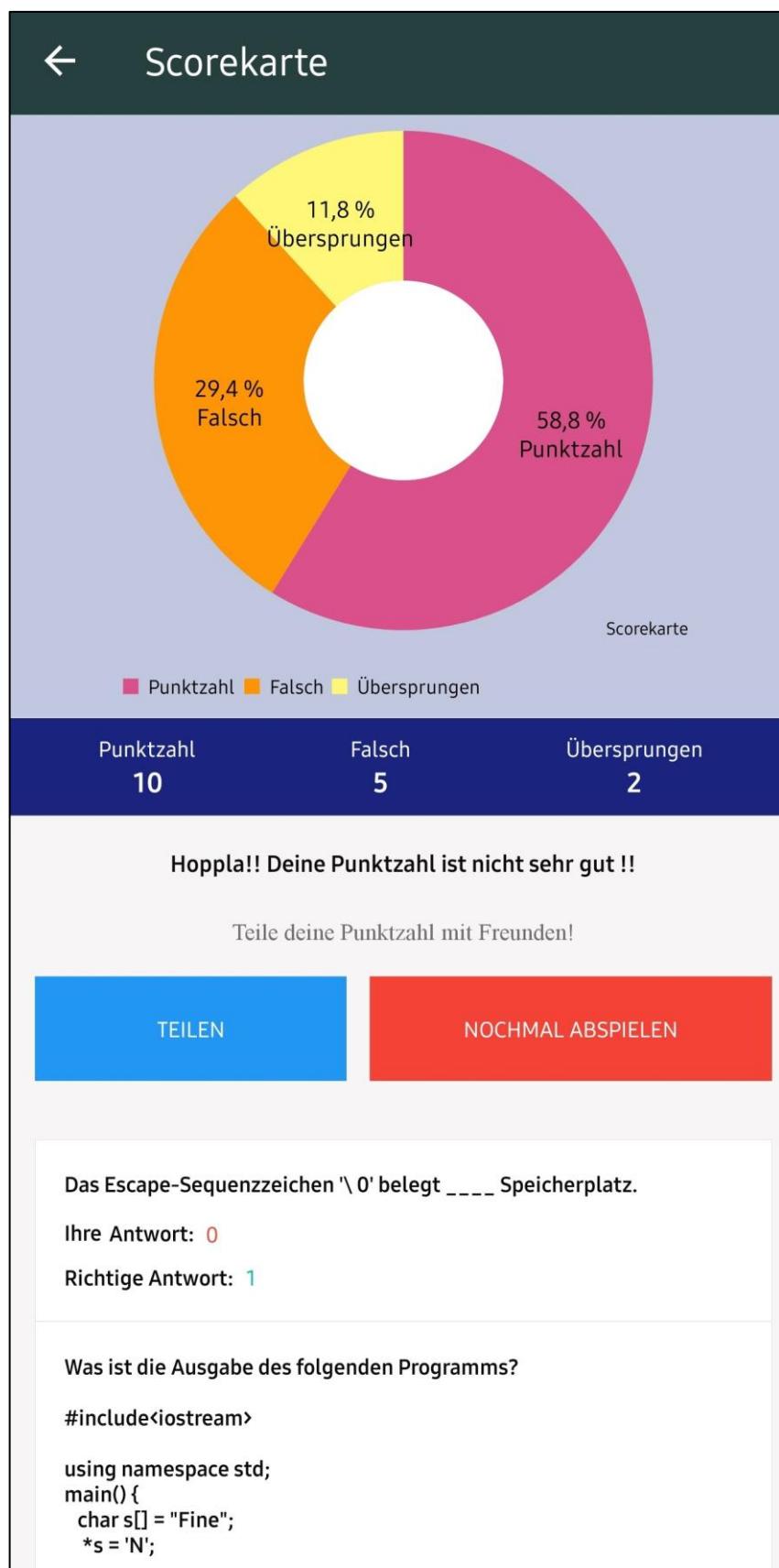


Abbildung 5.10: Quizfragenergebnisbildschirm

5.2.4 Interviewfragen

In der App gibt es parallel zu den Quizfragen auch Interviewfragen. Der Nutzer kann sich zu den Interviewfragen einfach vom Hauptbildschirm aus navigieren, dort kann sich der Nutzer das Thema auswählen und die dazugehörigen Fragen aussuchen, um eine Antwort zu bekommen. Die Abbildung 5.11 stellt das Interface der Interviewfragen dar.



Abbildung 5.11: 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 5.12 stellt den Bildschirm der Einstellungen dar.

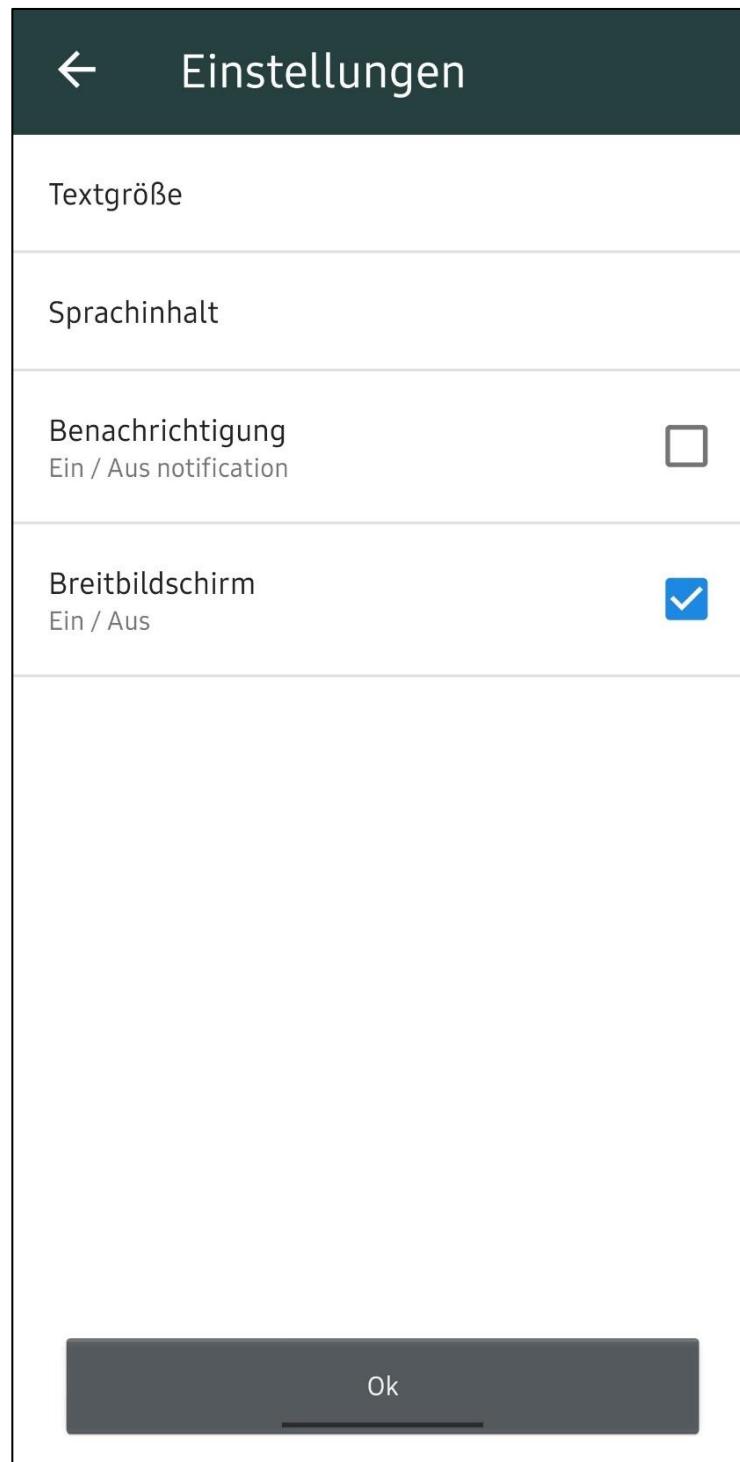


Abbildung 5.12: Einstellungen

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 5.13 liefert das Interface der Favoriten.

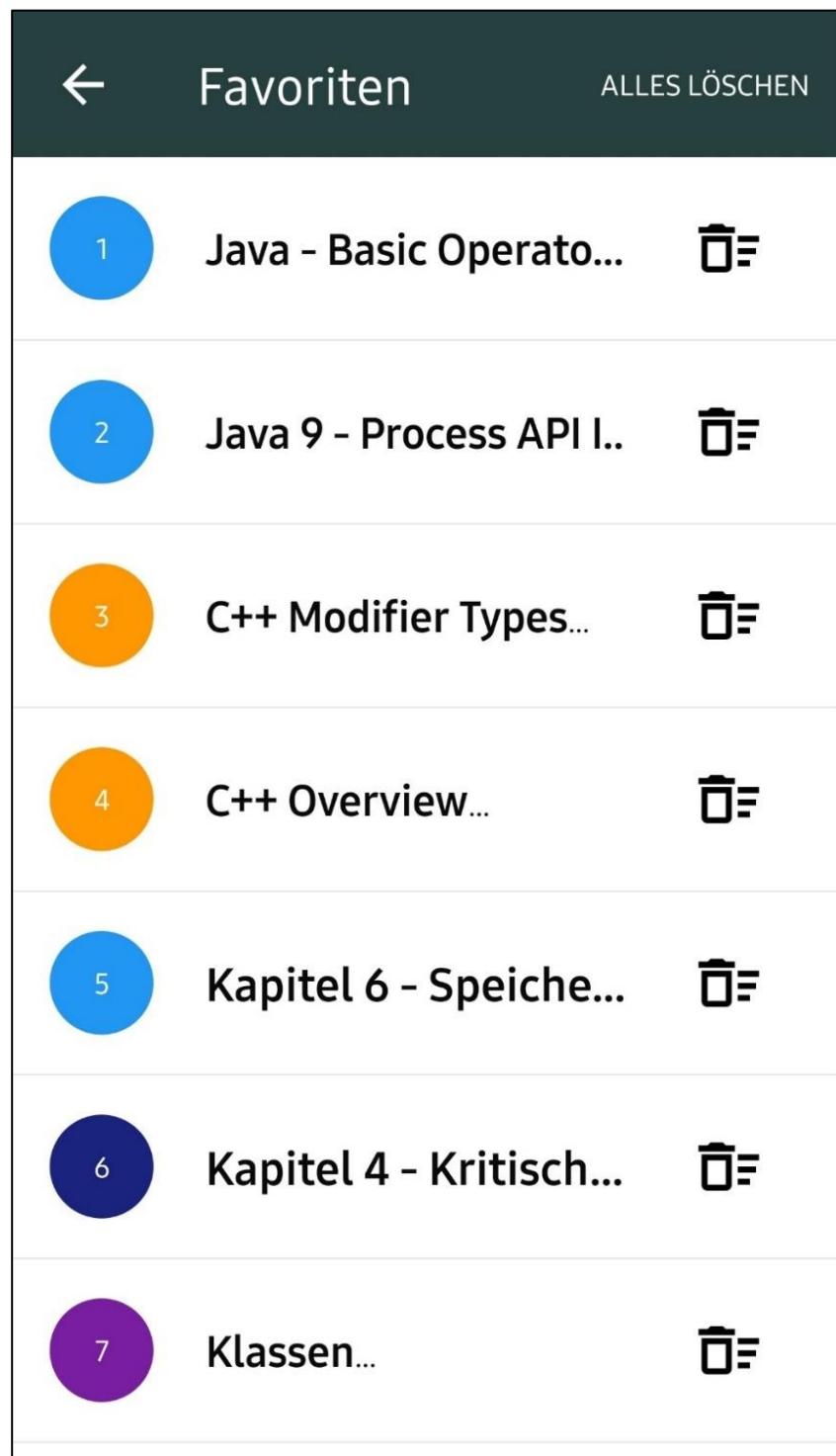


Abbildung 5.13: Favoriten

5.3 Kapitelfazit

Am Anfang des Kapitels wurde die Datenstruktur definiert und beschrieben. Darauf folgte die Beschreibung des Inhaltsparsers und Quizparsers. Zum Schluss des Kapitels wurden die wichtigsten Oberflächen der App dargestellt.

Bei der Implementierung der App konnte die Forschungsfrage, ob die Programmiersprache Java immer noch gut für die Android-Entwicklung anzuwenden ist, 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 der XML Sprache geschrieben und für das Datenübertragungsformat wird JSON Object 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, der Johannes-Gutenberg-Universität Mainz und der Hochschule Worms beantwortet wurden. Im Kapitel 6.3 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 Yoga Tab 10 Plus
- Huawei P smart 2017

Das Samsung Galaxy Note 8 ist ein Tablet mit einem Bildschirm von 6.3 Zoll und einer Displayauflösung mit 2960 x 1440 Pixeln. Samsung Galaxy Note 8 funktioniert unter dem Android 9 Pie Betriebssystem. Das Samsung Galaxy S7 ist ein Smartphone, dessen Bildschirm von 5,1 Zoll mit einer Displayauflösung von 2560 x 1440 Pixeln und unter Android 8 Oreo in Betrieb ist. Lenovo Yoga Tab 10 Plus ist ein Tablet mit einem Bildschirm von 10,1 Zoll und einer Displayauflösung von 2560 x 1600 Pixeln. Das Tablet läuft mit Android 6.0 Marshmallow. Huawei P smart ist ein Smartphone mit einem Bildschirm von 5.65 Zoll und einer Displayauflösung von 1920 x 1080 Pixeln. Huawei P smart funktioniert unter Android 9 Pie Betriebssystem.

6.1 Anforderungsabgleich

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

6.1.1 Abgleich der funktionalen Anforderungen

Die App Teach Me erfüllt alle funktionalen Anforderungen, die in Kapitel 4.1 gestellt wurden. Die Tabelle 5 zeigt den Abgleich der funktionalen Anforderungen.

Nº	Aufgabe	Erfüllt	Beschreibung
F.1	Lernmodul	Ja	Der Nutzer kann ein Modul auswählen. Siehe Kapitel 5.2.1.
F.2	Suche	Ja	Der Nutzer kann nach einem Begriffswort in den Modulen suchen. Siehe Kapitel 5.2.2.
F.3	Quizfragen	Ja	Der Nutzer kann an den Quizfragen teilnehmen. Siehe Kapitel 5.2.3
F.4	Interviewfragen	Ja	Der Nutzer kann sich mit den Interviewfragen abfragen. Siehe Kapitel 5.2.4
F.5	Video	Ja	Durch Android WebView kann der Nutzer Video Module / Inhalte ansehen. Siehe Kapitel 5.1
F.6	Textgröße	Ja	Der Nutzer kann die Textgröße in den Einstellungen der App verändern. Siehe Kapitel 5.2.5
F.7	Sprachauswahl	Ja	Der Nutzer kann die Sprache in den Einstellungen oder in der Schnellleiste im Hauptbildschirm ändern. Siehe Kapitel 5.2.5
F.8	Favoriten	Ja	Der Nutzer kann Module favorisieren. Siehe Kapitel 5.2.6
F.9	Textvorlesen	Ja	Der Nutzer kann sich das Kapitel vorlesen lassen. Siehe Kapitel 5.2.1
F.10	Inhaltkopieren	Ja	Der Nutzer kann Kapitelinhalte in den Buffer kopieren. Siehe Kapitel 5.2.1
F.11	Einstellungen	Ja	Der Nutzer kann Benachrichtigungen an- und ausschalten. Siehe Kapitel 5.2.5.

Tabelle 5: Abgleich der funktionalen Anforderungen

6.1.1 Abgleich der optionalen Anforderungen

Teach Me erfüllt alle optionalen Anforderungen, die in Kapitel 4.1 gestellt wurden, wie der Tabelle 6 zu entnehmen ist.

Nº	Aufgabe	Erfüllt	Beschreibung
0.1	Teilen	Ja	Der Nutzer kann Quizfragen und Kapitel teilen. Siehe Kapitel 5.2.1 und Kapitel 5.2.3.
0.2	Breitbild	Ja	Der Nutzer kann das Breitbild in den Einstellungen einschalten. Siehe Kapitel 5.2.5.
0.3	Ton	Ja	Die Quizfragen werden von einem Ton begleitet. Siehe Kapitel 5.2.3.
0.4	Quizergebnisse	Ja	Der Nutzer kann richtige und falsche Antworten ansehen. Siehe Kapitel 5.2.3.
0.5	Benachrichtigung	Ja	Der Nutzer kann Benachrichtigungen vom Entwickler bekommen.

Tabelle 6: Abgleich der optionalen Anforderungen

6.1.2 Abgleich der nicht-funktionalen Anforderungen

In der App werden alle nicht-funktionalen Anforderungen, die in Kapitel 4.1 gestellt wurden, erfüllt, wie der Tabelle 7 und 8 zu entnehmen ist.

Nº	Aufgabe	Erfüllt	Beschreibung
NF.1	Einheitliche Darstellungsformen der Quizfragen	Ja	Die Darstellung der Quizfragen ist in einem einfachen Antwort-Wahl-Verfahren dargestellt.
NF.2	Usability Ziele einhalten	Ja	Durch die Hierarchie ist die App einfach und verständlich aufgebaut.

Tabelle 7: Abgleich der nicht-funktionalen Anforderungen von NF.1 bis NF.2

Nº	Aufgabe	Erfüllt	Beschreibung
NF.3	Verfügbarkeit	Ja	Diese App kann auch auf einem Tablet geöffnet werden. Es würden sich alle Elemente skalieren.
NF.4	Selbsterklärbarkeit	Ja	Es wurden Standard-Icons des Material Designs 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 der nicht-funktionale Anforderungen von NF.3 bis NF.5

6.2 Fragebogen und Forschungsversuch

Fragebogen

Die Studierenden werden zuerst gebeten, ihre Erfahrung mit dem Umgang von Android Smartphones anzugeben - auf 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.

Die Abbildungen 6.1 und 6.2 zeigen das Fragenbogenmuster.

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:

Folgekurs

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

8

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

0000000000

- 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?

1 1 : 1

J. Neurosci., July 1, 2009 • 29(27):8773–8784 • 8783

r leicht

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

sehr leicht

Abbildung 6.1 : Fragebogenmuster - Vorderseite

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?



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?



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?



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?



Lässt sich die App „Teach Me“ einwanderfrei 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

Abbildung 6.2 : Fragebogenmuster -Rückseite

Forschungsversuch

Der Forschungsversuch an der Ruprecht-Karls-Universität Heidelberg hat am 3., 7., 18. und 26. Juni 2019 stattgefunden. Insgesamt wurden 57 Informatik-Studenten befragt. Die Befragung fand am Campus im Neuenheimer Feld in den Mathematik-Gebäuden INF 227, INF 306 und in der Zentralmensa vom Neuenheimer Feld statt.

Der Forschungsversuch an der Johannes-Gutenberg-Universität Mainz fand am 3., 12., 18. Und 23. Juli statt. Es haben 52 Informatik Studenten am Campus an der Umfrage teilgenommen.

Der Forschungsversuch an der Hochschule Worms hat am 6. und 14. August stattgefunden. Es wurden insgesamt 22 Studenten am Campus aus Informatik und Mobile Computing befragt.

6.3 Fragebogenauswertung

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

Die Studierenden haben vier mobile Android Geräte zur Auswahl gehabt. Am häufigsten wurde beim Anwendungsvergleich das Samsung Galaxy Note 8 gewählt. Das Diagramm 1 veranschaulicht die Statistik für die Nutzung der Geräte beim Anwendungsvergleich.

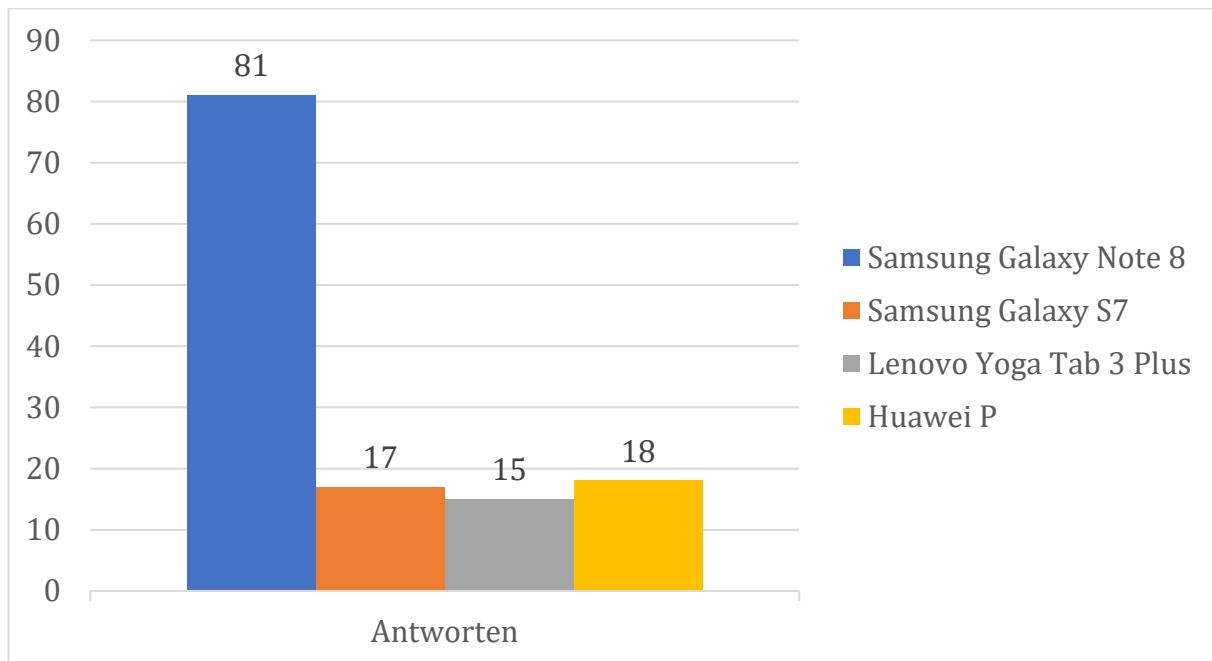


Diagramm 1: Nutzung der Geräte beim Anwendungsvergleich

Als Zweites wurden die Studenten gebeten, ihre Erfahrung mit dem Umgang von Android Smartphones anzugeben. Die Studierenden sollten die Antworten auf einer Skala von sehr gut bis ungenügend ankreuzen. Auf dem Diagramm 2 ist zu sehen, dass 43 Studenten einen sehr guten Umgang mit Android Geräten haben. Weitere 56 haben mit gut und 26 mit befriedigend geantwortet. Nur 6 Studenten haben beim Umgang mit Android Smartphones ausreichende Kenntnisse angegeben.

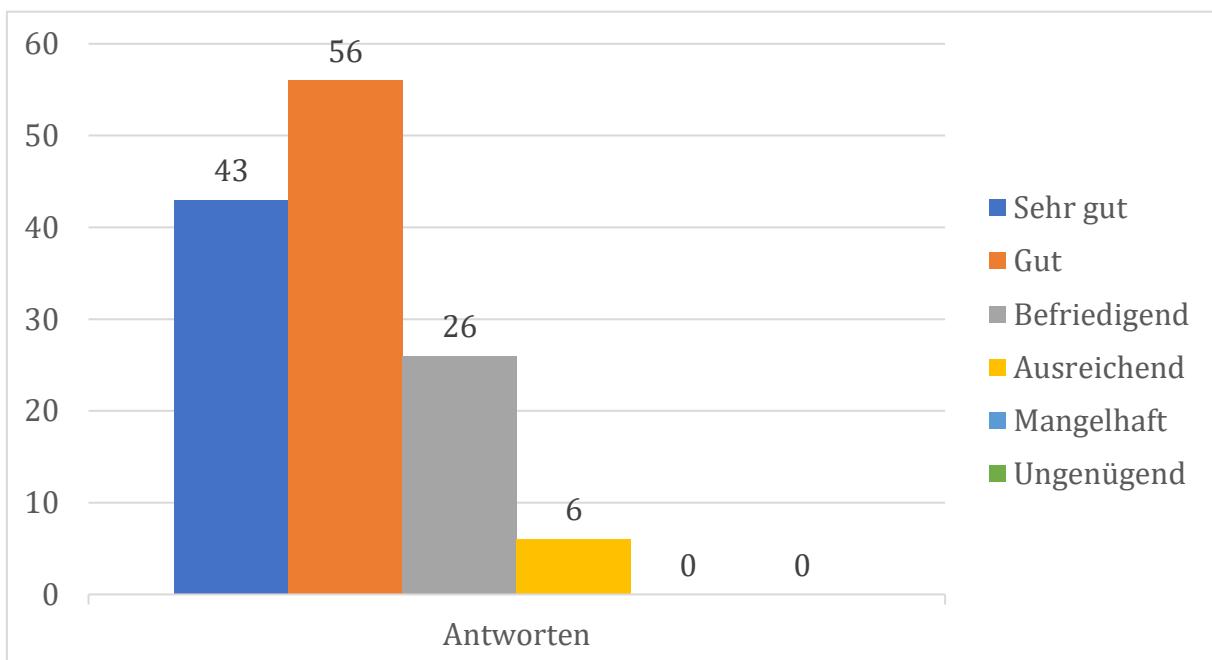


Diagramm 2: Erfahrung mit dem Umgang von Android Smartphones

Danach mussten die Befragten antworten, ob sie ähnliche Apps auf ihrem Smartphone bereits nutzen. 108 der Befragten haben diese Frage verneint, während 23 der Befragten dies bejahten. Das Diagramm 3 veranschaulicht die Statistik zur Nutzung von ähnlichen Apps.

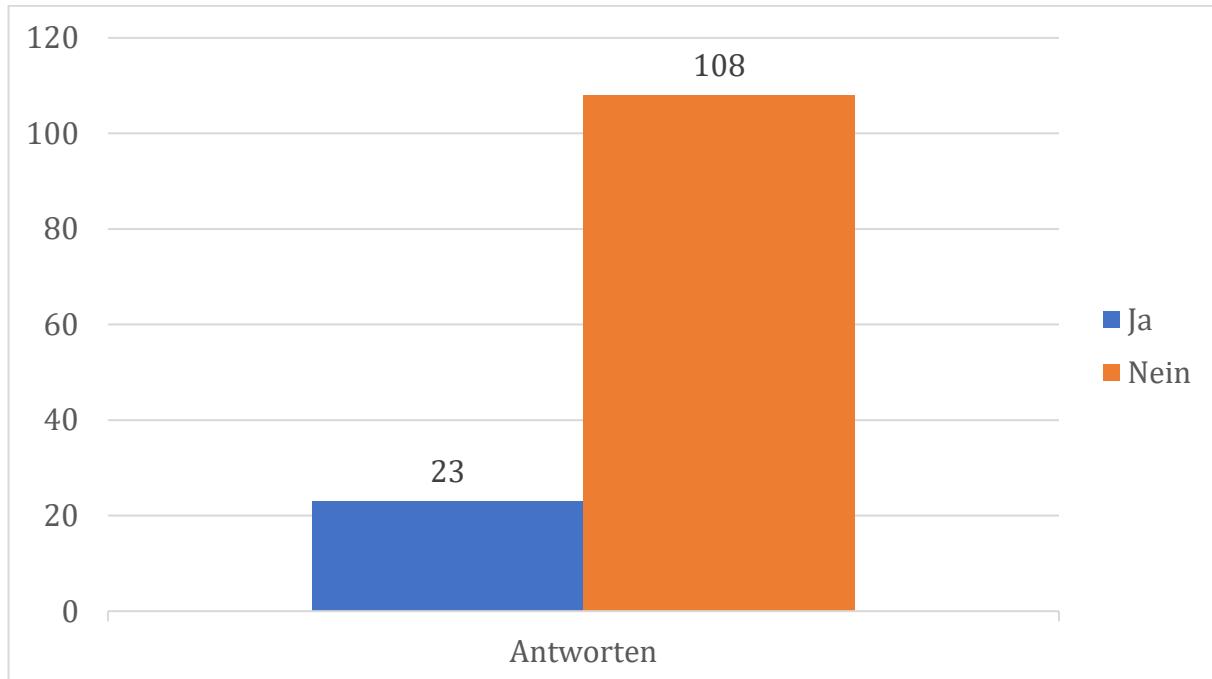


Diagramm 3: Nutzung von ähnlichen Apps

Als erste Aufgabe zur App Teach Me sollten sich die Studenten mit der Programmoberfläche vertraut machen. Das Diagramm 4 zeigt den Schwierigkeitsgrad der Aufgabe. 73 Studenten kreuzten an, dass die Aufgabe sehr leicht sei. 25 Befragte sagten, dass sie kleine Probleme mit der Aufgabe gehabt haben. 32 Studenten haben den Schwierigkeitsgrad im Bereich von 3 bis 6 angekreuzt. Nur eine Person fand die Aufgabe sehr schwierig.

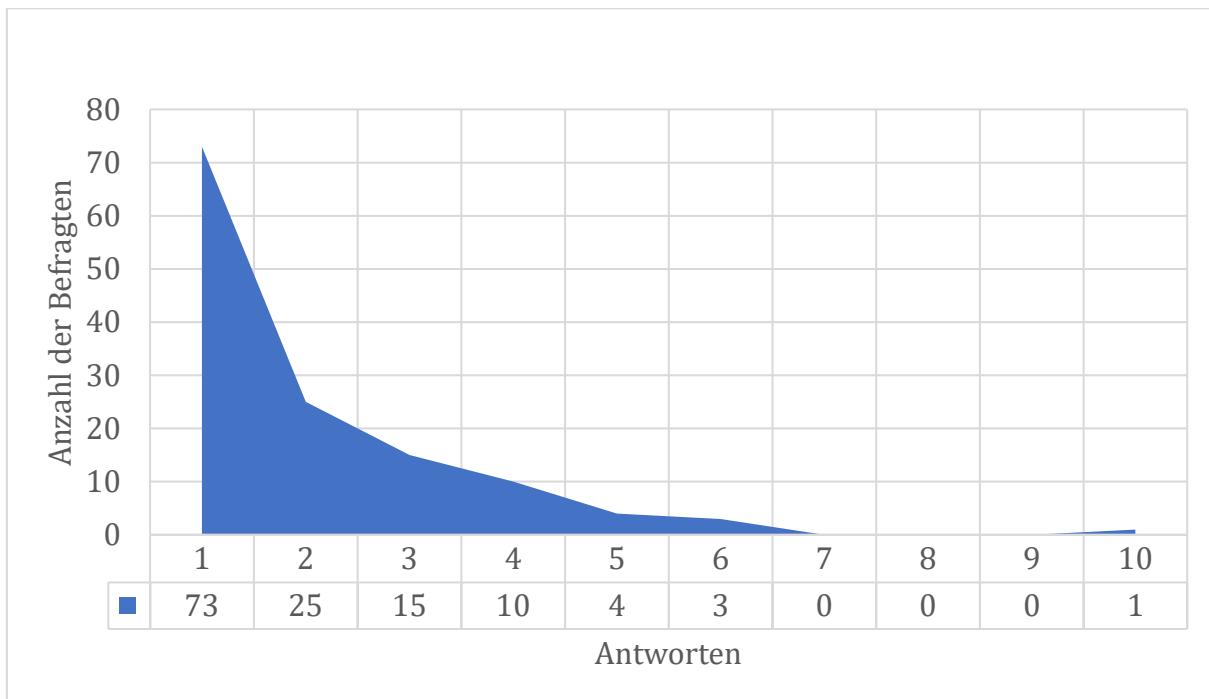


Diagramm 4: Ergebnisse zu Frage 1 – Programmoberfläche

Als zweite Aufgabe sollten die Befragten sich mit dem App Inhalt vertraut machen. 66 Studenten kreuzten an, dass die Aufgabe sehr leicht war. 64 empfanden den Schwierigkeitsgrad im Bereich zwischen 2 und 5. Ein Student hat die Aufgabe mit Schwierigkeitsgrad 8 bewältigt. Das Diagramm 5 liefert die Ergebnisse zu Frage 2.

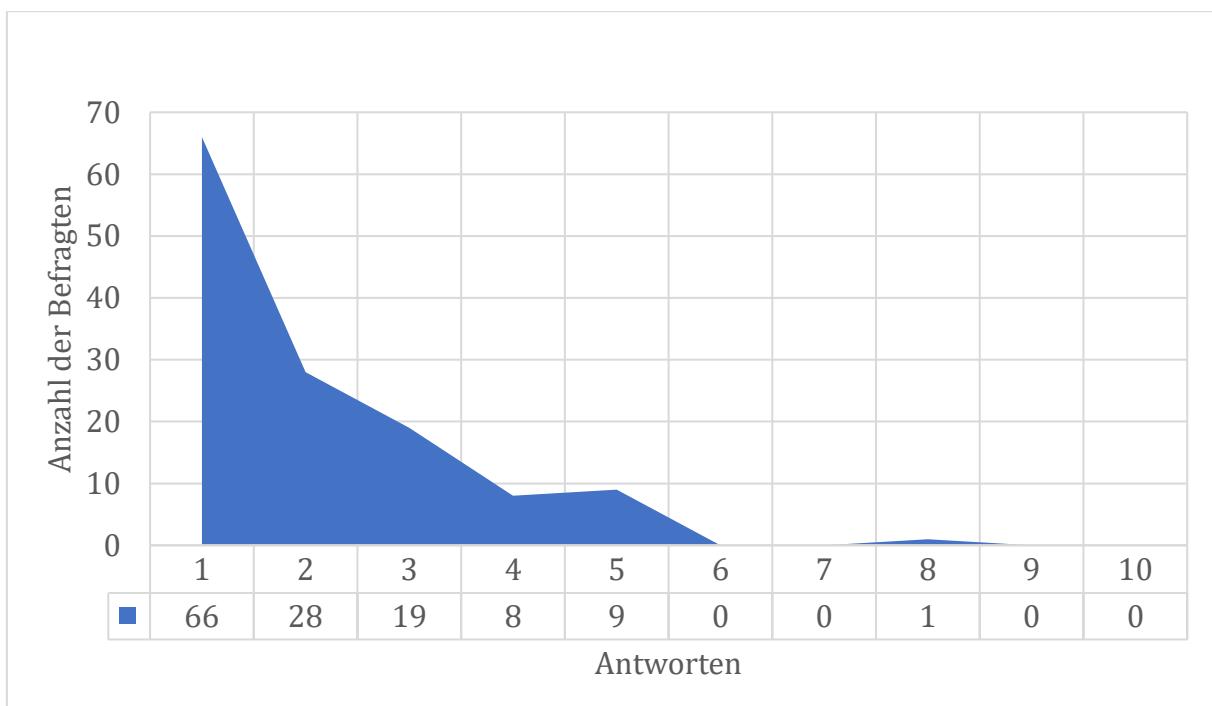


Diagramm 5: Ergebnisse zu Frage 2 – Inhalt

Bei der dritten Aufgabe sollten sich die Studenten mit der Programmsuche beschäftigen. Wie auf Diagramm 6 zu sehen ist, finden 58 der Befragten diese Aufgabe sehr leicht. 73 Studenten hatten kleine Schwierigkeiten bei der Aufgabe gehabt und wählten Werte im Bereich von 2 bis 5.

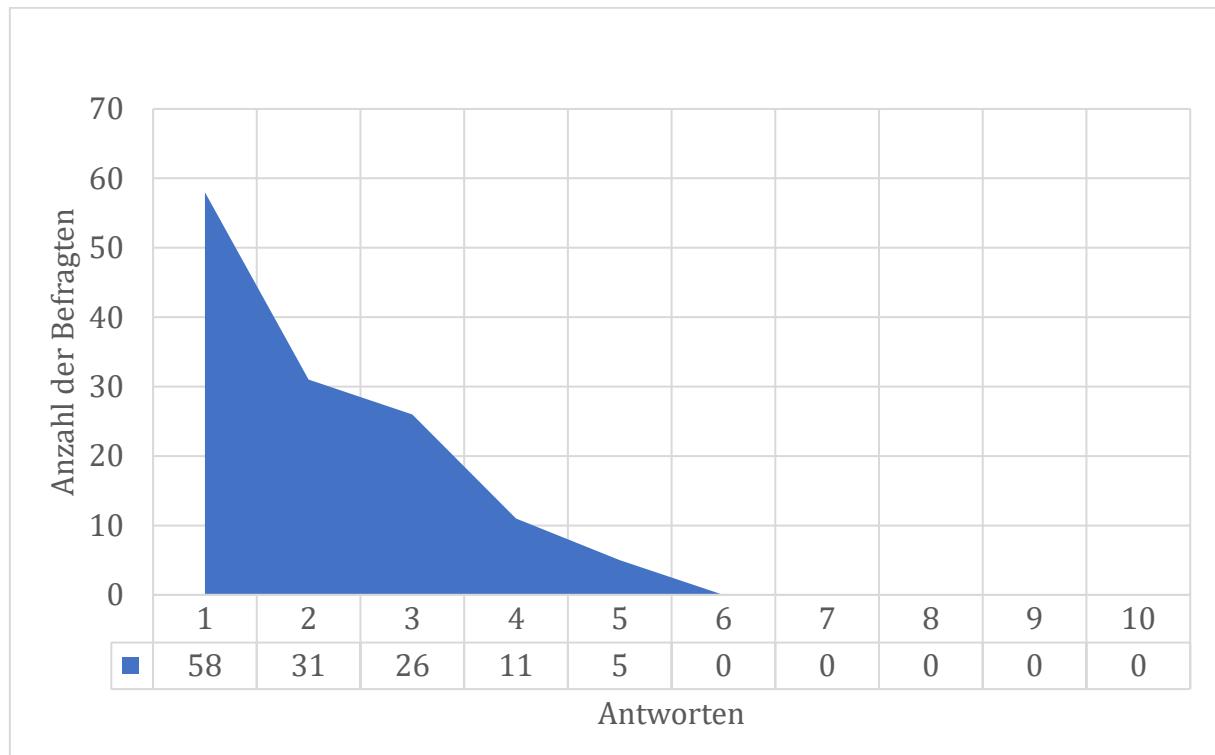


Diagramm 6: Ergebnisse zu Frage 3 – Programmsuche

Sich vertraut zu machen mit der Quizoberfläche war die vierte Aufgabe. Nur 33 Studenten fanden die Aufgabe leicht. Aber die meisten haben die Aufgabe mit Werten im Bereich von 2 bis 5 bewertet. Nur 3 Studenten finden die Aufgabe ein wenig schwierig. Das Diagramm 7 veranschaulicht die Ergebnisse dafür.

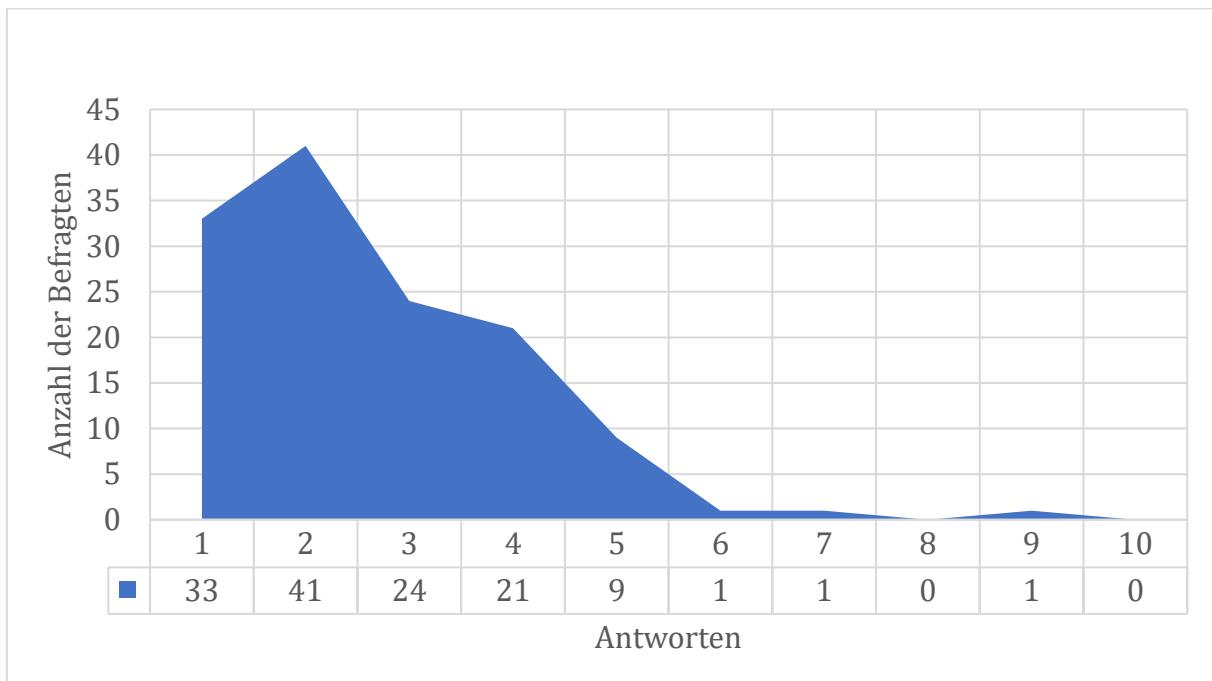


Diagramm 7: Ergebnisse zu Frage 4 – Quiz

Daraufhin sollten sich die Studenten mit einer Aufgabe zur Oberfläche der App beschäftigen. Für 39 Personen war die Aufgabe sehr leicht. 25 der Befragten gaben an, dass sie minimale Probleme mit der Aufgabe hatten. Den Mittelbereich der Werte haben 63 Personen angegeben. Für eine Person war die Aufgabe fast unmöglich. Und die restlichen 3 Befragten haben sie mit Mühe bewältigt. Das Diagramm 8 zeigt den Gesamteindruck zu Frage 5 – Fragen und Antworten.

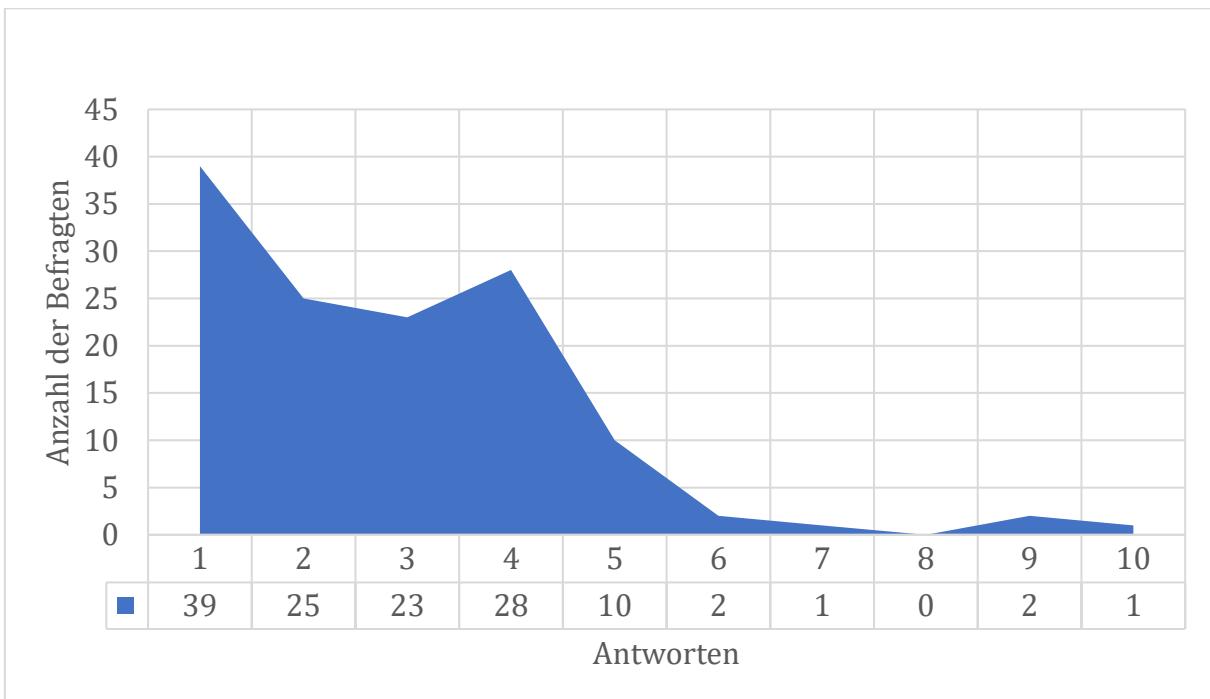


Diagramm 8: Ergebnisse zu Frage 5 – Fragen und Antworten

Schließlich 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. Das Diagramm 9 stellt die Ergebnisse zu Frage 6 - Einstellungen.

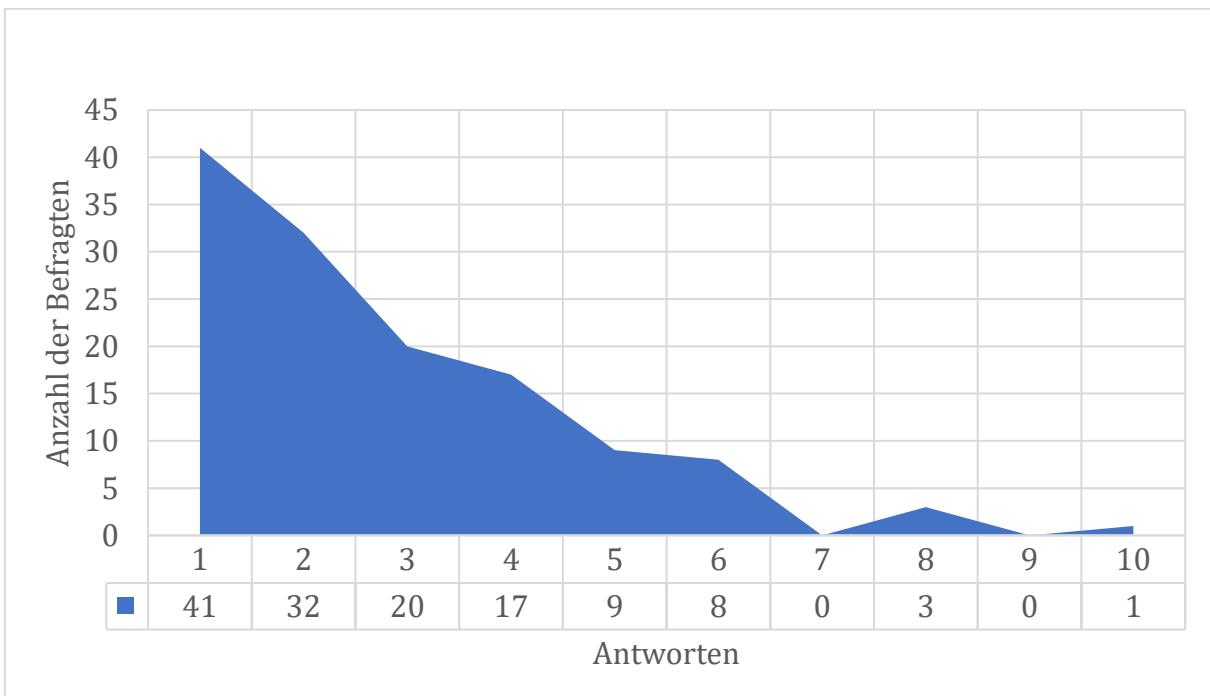


Diagramm 9: Ergebnisse zu Frage 6 – Einstellungen

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

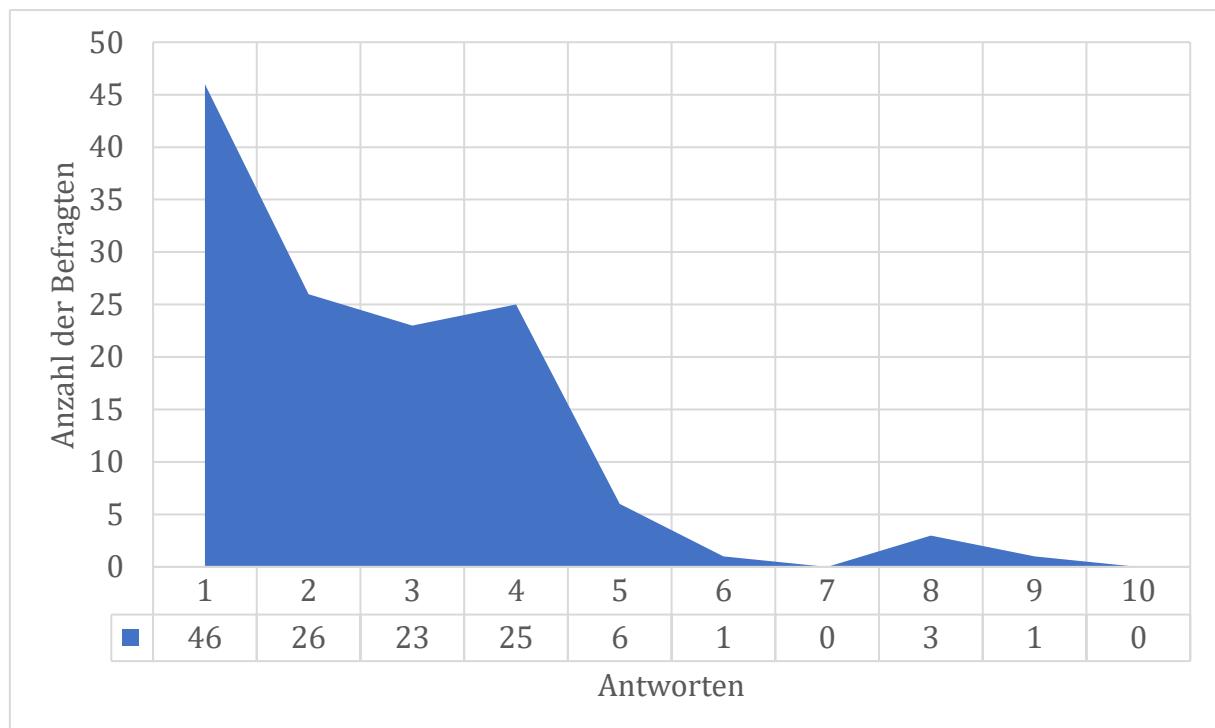


Diagramm 10: Ergebnisse zu Frage 7 – Studiengang

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

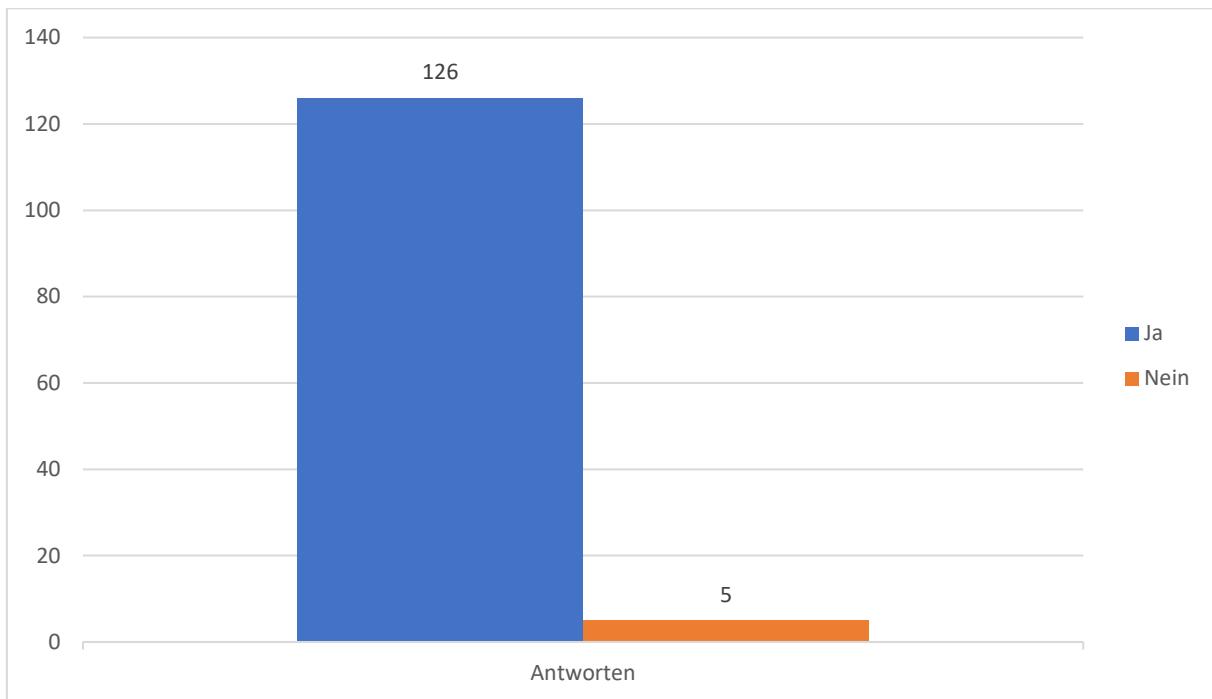


Diagramm 11: Lässt sich die App einwandfrei benutzen?

Bei der vorletzten Frage des Fragebogens ging es darum, ob die App genug Lernstoff besäße. 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.

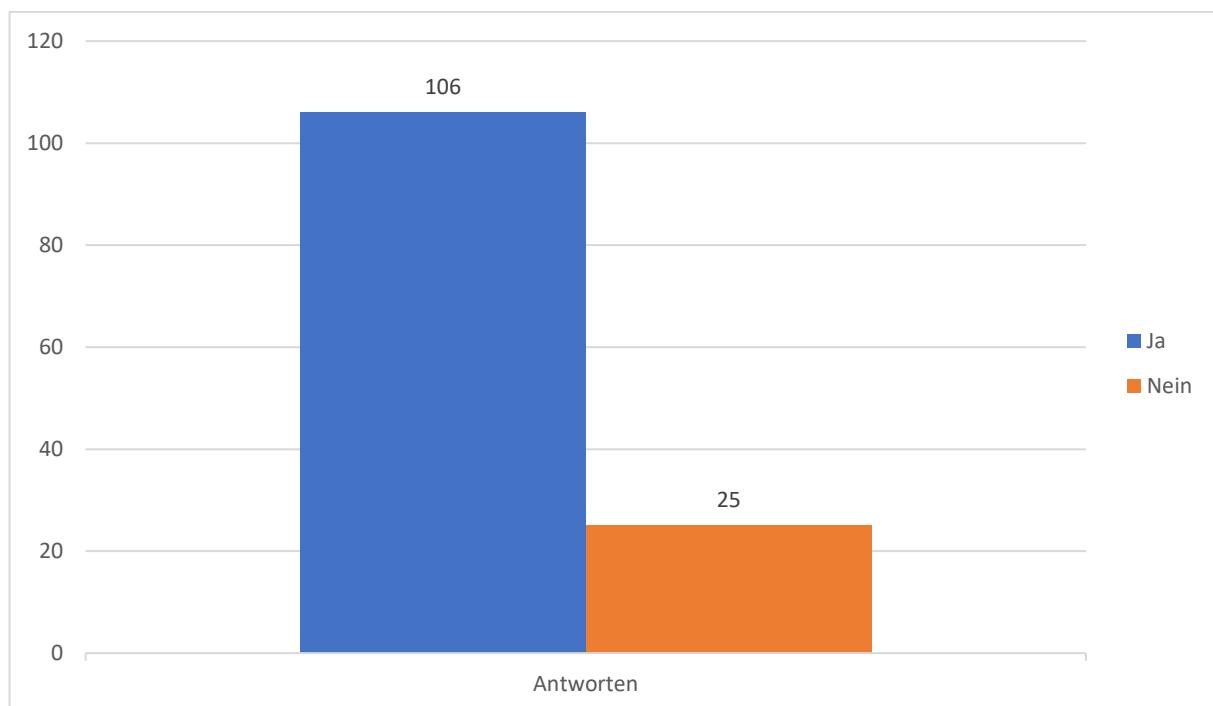


Diagramm 12: Hat die App genug Lernstoff?

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 antworteten mit einem klaren Nein, da sie meinten, dass die App noch nicht genug Lernstoff besäße. Ein weiteres Problem sahen sie darin, dass die App bis jetzt nur für Android zur Verfügung stünde. Sie sollte für die andere Plattformen wie iOS und auch für Web entwickelt werden. Das Diagramm 13 bietet die Statistik zu dieser thematisierten Frage.

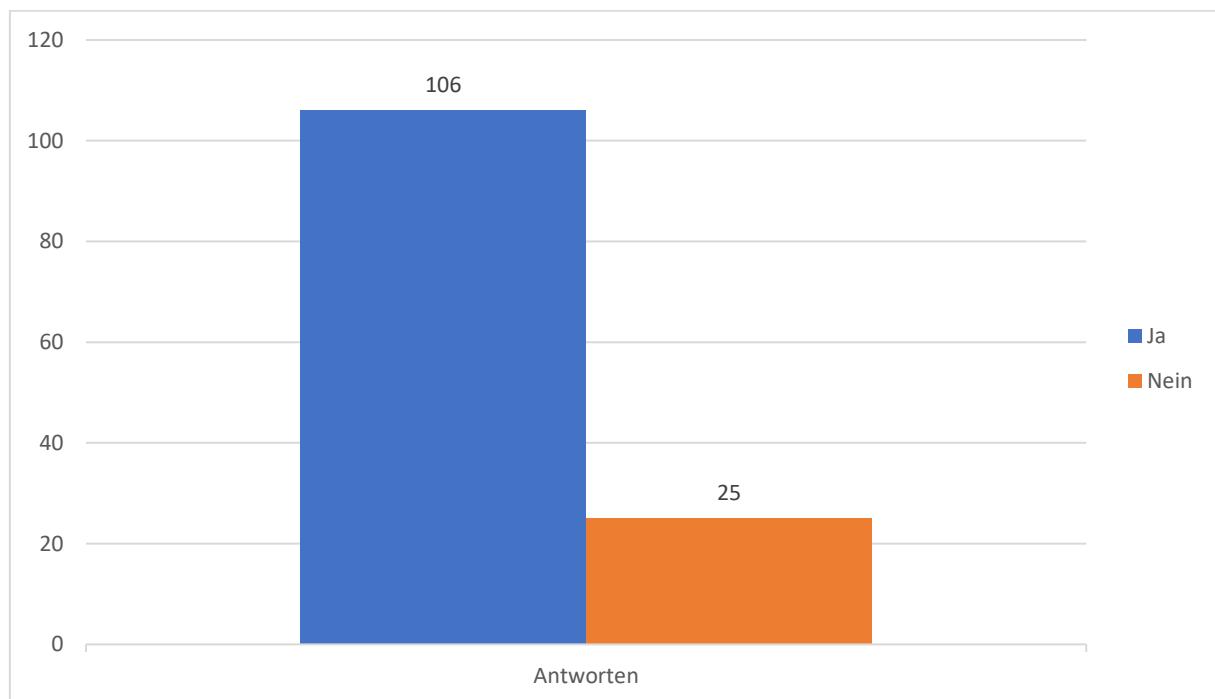


Diagramm 13: Kann die App einen Real-Tutor ersetzen?

Viele der Befragten an der Ruprecht-Karls-Universität Heidelberg hinterließen kein Feedback. Nur wenige sind gut und deutlich formuliert. Ein folgendes Feedback hinterließ ein 20-jähriger Student der 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 [39].

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

Die App lässt sich einwandfrei benutzen, aber es fehlt ein wenig Lernstoff [40].

Ein weiteres Feedback hinterließ ein 21-jähriger Informatik-Student:

Das App ist bei der Video-Wiedergabe abgestürzt. App besitzt ein leichtes Design, aber bietet nicht so viele Module / Lernstoff [41].

Genau wie an der Universität Heidelberg hinterließen nicht viele Studenten an der Johannes-Gutenberg-Universität Mainz ein Feedback. Meistens war es 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. [42].

Auch ein Student des 1. Semesters des Informatik Studiengangs hinterließ folgendes Feedback:

Einfaches Design, gutes Quiz und die App lässt sich gut bedienen [43].

Der Befragten an der Hochschule Worms hinterließen nur ein Feedback:

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[44].

6.4 Kapitelfazit

Als erstes wurden im Kapitel 6 die App Anforderungen, die im Kapitel 4.1 definiert wurden, abgeglichen. Die App Teach Me erfüllt alle gestellten Anforderungen. Danach wurde in diesem Kapitel die Struktur des Fragebogens beschrieben. Es folgte die Beschreibung der Forschungsversuche an der Universität Heidelberg, der Universität Mainz und der Hochschule Worms. Es wurden insgesamt 131 Studenten befragt. Zum Ende des Kapitels wurde die Fragebogenauswertung durchgeführt. Die Antworten auf den Fragenbögen zeigten, dass die App sich einfach bedienen und viele der Befragten konnten sich zudem durchaus vorstellen, dass die App Teach Me einen echten Tutor ersetzen könne. Doch fehlten bei der App noch einige Funktionen wie beispielsweise ein Sprachassistent und Lernstoff vor allem für die höheren Semester.

7 Zusammenfassung und Ausblick

Im 7. Kapitel wird eine Zusammenfassung der Masterarbeit verfasst. Anschließend kommt noch ein Ausblick.

7.1 Zusammenfassung

Ziel dieser Masterarbeit war die Entwicklung einer nativen App für das Android Betriebssystem, die als ein Tutor dienen sollte. Diesbezüglich wurden drei Forschungsfragen gestellt: Inwiefern lässt sich eine mobile Tutor App als ein Tutor verwenden? Kann eine mobile Tutor App eine reale Tutorin oder einen realen Tutor ersetzen? Ist die Programmiersprache Java immer noch gut für die Android-Entwicklung anzuwenden?

Im 2. Kapitel wurde eine Analyse von den aktuellsten App-Arten, Entwicklungssprachen, integrierten Entwicklungsumgebungen und vom Android Betriebssystem durchgeführt.

Eine Untersuchung von Tutorien an den deutschen Universitäten wurde in Kapitel 3 veranstaltet. Dazu wurden Sinn und Zweck von Tutorien, deren Arten und die wichtigsten Aufgaben von Tutoren erklärt.

Das 4. Kapitel hat sich mit der Konzeption und dem Entwurf beschäftigt. Dort wurden die funktionalen, optionalen und nicht-funktionalen App-Anforderungen für die App Teach Me gesetzt. Anschließend wurden das Mockup und der Prototyp dargestellt.

Im 5. Kapitel wurde die App Teach Me implementiert. Zuerst wurde die Datenstruktur der App beschrieben, die aus JSON Objekt besteht. Inhaltsdaten und Quizdaten wurden mit Hilfe der Android-Komponente WebView in der App Teach Me dargestellt. Ferner wurden der dazugehörige Inhaltsparser und Quizparser im selben Kapitel beschrieben und mit dem Quellcode belegt. Nachfolgend wurden die wichtigsten Benutzeroberflächen wie die Haupt-, Such-, Quizfragen-, Interview-, Einstellungs- und Favoriten-Bildschirme beschrieben. Bei der Implementierung der App und durch Analyse der App-Arten und Entwicklungssprache für das Android Betriebssystem wurde festgestellt, dass die Programmiersprache Java sich gut für die Entwicklung eignet, aber eine Unterstützung von anderen Sprachen benötigt. Die Oberflächen wurden in Android mit XML beschrieben und für die Datenstruktur wird vor allem JSON Objekt oder SQLite benutzt.

Zum Schluss wurde in Kapitel 6 ein Anwendungsvergleich durchgeführt. Als erstes wurden die Anforderungen abgeglichen. Weiter im Kapitel wurde der Fragebogen beschrieben, mit diesem und mit der implementierten App fand eine Befragung der Informatik-Studierenden an den ausgewählten Universitäten und Fachhochschule statt: an der Ruprecht-Karls-Universität Heidelberg, der Johannes-Gutenberg-Universität Mainz und der Hochschule Worms. Es wurden insgesamt 131 Informatik-Studenten befragt. Die Antworten auf den Fragenbögen zeigten, dass viele Studenten eine gute Erfahrung mit der Nutzung von Android Geräten haben. Von 131 befragten Studenten nutzen nur 23 eine ähnliche App für ihr Studium. Antworten auf den Fragebögen zeigten auch, dass die App Teach Me sich einfach und leicht bedienen lässt. Darüber hinaus sagten die Ergebnisse auch aus, dass die größere Anzahl an Studenten sich durchaus vorstellen könne, dass die App einen echten Tutor ersetzen könnte. Doch fehlten bei der App noch einige Funktionen wie beispielsweise ein Sprachassistent und Lernstoff vor allem für die höheren Semester.

Aus all den oben genannten Punkten kann man schließen, dass die Programmiersprache Java für die Entwicklung von mobilen Anwendungen für Android Betriebssystem-Geräte gut geeignet ist. Eine gut gestaltete App und genügend Lernmaterial in der App können einen echten Tutor ersetzen. Darüber hinaus muss die App noch für weitere Betriebssysteme wie iOS oder Windows und auch für Web entwickelt werden.

7.2 Ausblick

Mit der Entwicklung des Internets und der mobilen Technologie kann davon ausgegangen werden, dass mobile Anwendungen einen Tutor ersetzen werden. Anwendungen müssen für alle Arten von Plattformen entwickelt werden. Die App Teach Me und die Umfrage zeigen, dass viele Studenten sich vorstellen können, dass mobile Apps einen echten Tutor ersetzen werden. Dabei stellen sich folgende weiterführende Fragen: Woran liegt es, dass so viele Studenten sich vorstellen können, dass die App einen realen Tutor ersetzen kann? Liegt es womöglich an der Qualität der Tutoren oder der didaktischen Aufbereitung des Lernstoffs für die Studenten? Oder am Mangel an Flexibilität in Bezug auf die Unterrichtszeiten? Außerdem kann man sich Fragen, ob eine App überhaupt Lernstoff für höhere Semester bereitstellen kann, da es in höheren Semestern weniger auf die Fülle des Lernstoffs ankommt, was in den Umfragen bemängelt wurde, sondern auf das Verständnis komplexerer Zusammenhänge, die von einer App eventuell gar nicht abgefragt werden können.

8 Literaturverzeichnis

- [1] ONLINE: Improving Student Engagement. Abgerufen am 5 Oktober 2019, von <https://cie.asu.edu/ojs/index.php/cieatasu/article/view/745/162>
- [2] ONLINE: Pearson. Student Mobile Device Survey 2015. Abgerufen am 5 Oktober 2019, von <https://www.pearsoned.com/wp-content/uploads/2015-Pearson-Student-Mobile-Device-Survey-College.pdf>
- [3] ONLINE: On Story and Execution: Sebastian Thrun, Udacity, and The Future. Abgerufen am 5. Oktober 2019, von <https://medium.com/udacity/on-story-and-execution-sebastian-thrun-udacity-and-the-future-77ce6e415208>
- [4] ONLINE: Udacity - About Us. Abgerufen am 5. Oktober 2019, von <https://www.udacity.com/us>
- [5] ONLINE: Udacity. Abgerufen am 5 Oktober 2019, von <https://www.udacity.com/>
- [6] ONLINE: MIT and Harvard announce edX. Abgerufen am 5. Oktober 2019, von <https://news.harvard.edu/gazette/story/2012/05/mit-and-harvard-announce-edx/>
- [7] ONLINE: Coursera. Abgerufen am 5. Oktober 2019, von <https://blog.coursera.org/about/>
- [8] ONLINE: Socrative Reviews & Product Details. Abgerufen am 5. Oktober 2019, von <https://www.g2.com/products/socrative/reviews>
- [9]. ONLINE: Improving Student Engagement in Higher Education through Mobile-Based Interactive Teaching Model Using Socrative. Abgerufen am 5. Oktober 2019, von <http://eprints.sunway.edu.my/695/1/Lim%20Woan%20Ning%20improving%20student%20engagement.pdf>
- [10] ONLINE: Using Socrative to Enhance In-Class Student Engagement and Collaboration. Abgerufen 5. Oktober 2019, von <http://dx.doi.org/10.5121/ijite.2015.4302>

- [11] ONLINE: NATIVE APPS, MOBILE SITES AND HYBRID APPS. Abgerufen am 5. Oktober 2019, von <http://www.welkinfort.com/2016/06/18/native-apps-mobile-website-hybrid-apps/>
- [12] SCHILLING, KAROLINA: Apps Machen. Der Kompaktkurs für Designer: Von der Idee bis zum klickbaten Prototyp. München: Carl Hanser Verlag, 2016.
- [13] SEMLER, JAN: App-Design. Alles zu Gestaltung, Usability und User Experience. Bonn: Rheinwerk Verlag, 2016.
- [14] ONLINE: Developer Survey Results 2019. Abgerufen am 5. Oktober 2019, von <https://insights.stackoverflow.com/survey/2019>
- [15] KÜNNETH, THOMAS: Android 8. Das Praxisbuch für Java-Entwickler. 5. Auflage Bonn: Rheinwerk Verlag, 2018.
- [16] LERUSALIMSCHY, ROBERTO: Programmieren mit LUA. München: Open Source Press, 2006.
- [17] FERNANDEZ, MICHELLE M.: Corona SDK Mobile Game Development: Beginner's Guide - Second Edition. Packt Publishing, 2015.
- [18] FRANKE, FLORIAN: Apps mit HTML5, CSS3 und JavaScript: für Android, iPhone und iPad. Bonn, Rheinwerk Verlag, 2015.
- [19] ONLINE: React Native. Learn once, write anywhere. Abgerufen am 5. Oktober 2019, von <https://facebook.github.io/react-native/>
- [20] MEIER, RETRO: Professionelle Android App-Entwicklung. Weinheim, WILEY-VCH Verlag, 2019.
- [21] ONLINE: Device Monitor. Abgerufen am 5. Oktober 2019, von <http://www.androiddocs.com/tools/help/monitor.html>
- [22] SEAGRAVE WYKEN: B4A. Rapid Android App Development using BASIC. CreateSpace Independent Publishing Platform, 2015.

[23] GEIRHOS, MATTHIAS: Professionell entwickeln mit C# 6 und Visual Studio 2015. Bonn, Rheinwerk Verlag, 2016.

[24] ONLINE: Develop Apps on Your Android Device with AIDE. Abgerufen am 5. Oktober 2019, von <https://www.codeproject.com/Articles/1103585/Develop-Apps-on-Your-Android-Device-with-AIDE>

[25] ONLINE: Mobile Operating System Market Share Worldwide. Abgerufen am 5. Oktober 2019, von <https://gs.statcounter.com/os-market-share/mobile/worldwide>

[26] ONLINE: Die Klassen Activity und Intent. Abgerufen am 5. Oktober 2019, von <https://individuapp.com/softwareentwicklung/android-kurs/die-klassen-activity-und-intent/>

[27] ONLINE: Architectural Guidelines to follow for MVP pattern in Android. Abgerufen am 5. Oktober 2019, von <https://android.jlelse.eu/architectural-guidelines-to-follow-for-mvp-pattern-in-android-2374848a0157>

[28] ONLINE: Github. Built for developers. Abgerufen am 5. Oktober, von <https://github.com/>

[29] ONLINE: MÜSLI. Mathematisches Übungsgruppen- und Scheinlisten-Interface. Abgerufen am 5. Oktober 2019, von <https://muesli.mathi.uni-heidelberg.de/>

[30] ONLINE: Vorlesungsverzeichnis. Abgerufen am 5. Oktober 2019, von https://jogustine.uni-mainz.de/scripts/mgrqispi.dll?APPNAME=CampusNet&PRGNAME=ACTION&ARGUMENTS=-AfmWuXXhVKuH8uKmNfPhFPo-w9NaAbJrYm~q0~CXybASzcA2EfJTN0fwXexZq51Qbp-54ry0yFKsxDEEJ8X~aSDmdFCiPj1z8K8p5SfdvIDHoUCs4QBI0mr6VEnHZDyG gegFheLVpsIR1hlW1bXfNzfLh9Ifnuc0yEF1eulDpvIWt7qVP4d0y09YJ0fg5olLrqtVpF8E1oiRlEAs_

[31] ONLINE: Studiengang – Lehrplan. Abgerufen am 5. Oktober 2019, von https://lsf.hsworms.de/qisserver/rds?state=wplan&r_zuordabstgv.semvonint=1&r_zuordabstgv.sembrisint=1&missing=allTerms&k_parallel.parallelid=&k_abstgv.abstgvnr=105&r_zuordabstgv.phaseid=&week=40_2019&act=stg&pool=stg&show=liste&P.vx=kurz&P.subc=plan

[32] ONLINE: Tutorium. Abgerufen am 5. Oktober 2019, von
<https://www.studieren.at/uni-abc/tutorium/>

[33] KNAUF, HELEN: Tutorenhandbuch. Einführung in die Tutorenarbeit. Bielefeld,
UniversitätsVerlagWebler, 2012.

[34] ANTOSCH-BARDOHN, JANA; BEEGE BARBARA; PRIMUS NATHALIE: Tutorien
erfolgreich gestalten. Ein Handbuch für die Praxis. Paderborn, Ferdinand Schöningh, 2016.

[35] HILLEBRECHT, STEFFEN : Tutorien und Seminare vorbereiten und moderieren: Eine
kleine Trickkiste für Tutoren und wissenschaftliche Mitarbeiter. Wiesbaden: Springer
Fachmedien Wiesbaden, 2016.

[36] Stahl, Eberhard: Dynamik in Gruppen. Handbuch der Gruppenleitung. Weinheim:
BeltzPVU, 2007.

[37] ONLINE: Angebote für Tutorinnen und Tutoren. Abgerufen am 5. Oktober 2019, von
<https://www.uni-konstanz.de/lehren/qualifizierungsmoeglichkeiten-fuer-lehrende/qualitut/fuer-tutorinnen-und-tutoren/>

[38] ONLINE: What is JSON? Abgerufen am 5. Oktober 2019, von
<https://developers.squarespace.com/what-is-json>

[39] Fragebogen № 8 vom 26.06.2019

[40] Fragebogen № 13 vom 03.06.2019

[41] Fragebogen № 18 vom 07.06.2019

[42] Fragebogen № 12 vom 03.07.2019

[43] Fragebogen № 12 vom 23.06.2019

[44] Fragebogen № 12 vom 14.08.2019

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();
        }
    }
}
```