Eine Android App für alltägliche Studieninformationen am Beispiel der Heinrich-Heine-Universität Düsseldorf

Bachelorarbeit

von

Tobias Krauthoff

aus

Wesel

vorgelegt am

Lehrstuhl für Rechnernetze und Kommunikationssysteme
Prof. Dr. Martin Mauve
Heinrich-Heine-Universität Düsseldorf

August 2012

Betreuer:
Markus Koegel M. Sc.

Kurzbeschreibung

Smartphones gehören mittlerweile zu alltäglichen Wegbegleitern, die den Benutzern das Leben erleichtern, indem zu jeder Zeit alle gewünschten Informationen abrufbar sind. Leider sind diese Informationen oft auf verschiedenen Webseiten erreichbar und daher sind Komfortanwendungen sehr gefragt. Zum Zeitpunkt dieser Arbeit existierte noch keine Komfortanwendung für die Heinrich-Heine-Universität Düsseldorf, welche für Studenten Informationen des Studienalltags bereit hält.

Die in dieser Arbeit entwickelte Applikation soll für den Benutzer schnell und intuitiv zu bedienen sein. Sie sollte die wichtigsten Informationen, wie zum Beispiel den Lage,- Mensa- oder Stundenplan verwalten, sowie einen sicheren Informationsaustausch ermöglichen. Dazu zählte, dass Verbindungen mit dem Datenbankserver oder weiteren Systemen der Universität gesichert und authentifiziert stattfinden sollten. Es war nötig, das Framework und die Eigenschaften von Android genauestens zu kennen und zu nutzen, sowie den Server gegen Eingriffe von außen zu sichern. Zudem sollte die Applikation unabhängig von äußeren Einflüssen stabil laufen und sich ideal in das System des Smartphones integrieren.

Die Arbeiten wurden dadurch erschwert, dass keine Schnittstellen von der Universität zur Verfügung gestellt wurden. Weiterhin mussten zahlreiche Bugs behoben sowie fehlende Bibliotheken des Android-Systems umgangen werden.

Ziel war es, eine Applikation zu erstellen, die die wichtigsten, alltäglichen Informationen schnell und sicher für den Benutzer bereithält und auch offline verfügbar macht. Die Applikation wurde durch eine kontrollierte Fallstudie unterstützt und ausgebaut, sodass sie der Universität als Vorschlag für eine offizielle Applikation für Android-Endsysteme vorgelegt werden konnte.

Danksagung

Bedanken möchte ich mich vor allem bei meinem Betreuer Markus Koegel, der jederzeit ansprechbar war, sich viel Zeit für mich nahm und viele Anregungen lieferte. Außerdem auch bei Markus Schwägerin, welche in ihrer Tätigkeit als Diplom-Psychologin bei der Erstellung der Fallstudie Hilfestellung geleistet hat.

Außerdem möchte ich mich bei allen weiteren Mitarbeitern des Lehrstuhls Rechnernetze bedanken, welche immer bereit waren, Fragen zu beantworten oder Denkanstöße zu geben. Dies gilt auch für alle Kollegen und Freunde aus dem Rechnerlabor, die immer zur Verfügung standen. Besonderer Dank gilt dabei Raphael Bialon, welcher sich viel Zeit nahm, um mir den Wiedereinstieg in das Betriebssystem Ubuntu zu erleichtern.

Außerhalb der Universität möchte ich meiner Freundin danken, da sie mich immer motiviert hat und mir auch Kraft gab, die eine oder andere kürzere Nacht durchzustehen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einführung

Die Verbreitung und Nutzung von mobilen Endgeräten wie Smartphones hat in den letzten Jahren erheblich zugenommen [Tro12]. Neben der Bereitstellung von Telekommunikation und dem Zugang zum *World Wide Web* haben sich außerdem Komfortanwendungen etabliert, die den Benutzern Zugang zu Informationen erheblich erleichtern oder direkt mehrere Informationsquellen in einer Anwendung vereinen.

Für Studenten verschiedener Universitäten existieren Anwendungen, um alltäglich interessante Daten wie den jeweiligen Mensaplan oder die neuesten Ankündigen der Institute schnell zugreifbar zu machen. Für die Heinrich-Heine-Universität gibt es allerdings noch keine solche Anwendung.

Im Rahmen dieser Bachelorarbeit soll eine Smartphone-App für Android-Geräte entwickelt werden, welche Schnittstellen zu den wichtigsten Informationen beinhaltet. Gewünscht sind Anbindungen an den Stunden- und Mensaplan. Nützlich wäre auch ein Lageplan zur Orientierung auf dem Campus, der zudem eine einfache Navigation bietet. Da die Universität einen RSS-Feed anbietet, sollte auch dieser in der Applikation verfügbar sein. Zusätzlich ist ein Lesezeichen erwünscht, welches die wichtigsten Internetseiten der Universität beinhaltet. Als Zusatz soll ein Forum mit Nachrichtenfunktion eingebunden werden, das wichtige Informationen, die nicht auf der Universitätsseite stehen, schnell unter den Studenten verbreiten kann.

1.1 Aufbau dieser Arbeit

Zunächst wird in Kapitel 2 Android selbst und seine Architektur erklärt. Darauf folgen elementare Grundlagen einer Applikation, also die Bausteine der Programmierung. Zuletzt wird der Lebenszyklus einer Android-Applikation dargestellt. Kapitel 3 dient zur Übersicht der Applikation. Es werden die Funktionen und Anforderungen an die Applikation festgehalten. In Kapitel 4 werden die Schnittstellen der Applikation sowie benötigte Ressourcen zur Realisierung der Schnittstelle dargestellt. Kapitel 5 berichtet über Erfahrungen, Anregungen und Kritik, welche Probanden in einer kontrollierten Fallstudie festgehalten haben. Kapitel 6 bietet eine Zusammenfassung der Arbeit sowie eine Aussicht auf zukünftige Arbeiten.

1.2 Verwandte Arbeiten

Zum Zeitpunkt dieser Arbeit boten erst wenige Universitäten eine Applikation für alltägliche Informationen an, welche über den Applikationen-Markt von Google gefunden werden konnten. Im Folgenden werden die Applikationen von drei Universitäten und das mobile Angebot der Heinrich-Heine-Universität dargestellt.

mPortal Uni Koblenz-Landau Die Unversität Koblenz-Landau bietet eine Applikation [Goo12h] mit zahlreichen Schnittstellen, wie den Speiseplan, eine Suche im Bibliotheksverzeichnis, eine Übersicht über Busverbindungen und Einrichtungen sowie ein Mitarbeiterverzeichnis, als auch die aktuelle Auslastung der Computerräume an. Die Funktion des Lageplans und Uni-Mail sowie die Übersicht über den Hochschulsport sind nur Weiterleitungen, die den externen Browser öffnen. Allgemein beinhaltet die Applikation viele Informationen. Sie ist optisch jedoch nicht so hochwertig, wie die nachfolgende Applikation. Außerdem startet die Applikation im Vollbildmodus, sodass der Statusbalken am oberen Rand Aktualisierungen oder Mitteilungen des Systems verdeckt.

myTU Die Technische Universität Bergakademie Freiberg bietet eine optisch, ansprechende Applikation [Goo12i], die jedoch für jede Interaktion eine Internetverbindung benötigt. Die Applikation bietet ein Importtool für den Stundenplan, eine Stichwort- und Barcodesuche für Bücher aus der Bibliothek, mehrere Feeds sowie den Speiseplan. Leider stürzt die Applikation beim Drücken des Optionsbuttons von Android ab und nicht alle Fenster werden vollständig geladen, sodass manche Bildschirme ohne Inhalt sind. Außerdem sind nicht alle Fenster der Applikation in den Reitern integriert, sodass das Starten beziehungsweise Öffnen neuer Fenster den Reiter überdeckt.

Universität Hohenheim Die Universität Hohenheim bietet eine sehr ausgereifte Applikation [Goo12l] mit dem höchsten Informationsgehalt an, jedoch wird auch hier für jede Interaktion eine Internetverbindung benötigt. Die Applikation bietet ein Importtools für den Stundenplan, mehrere Feeds sowie den Speise- oder Lageplan. Der Lageplan wird mit Hilfe von Google-Maps realisiert. Außerdem besteht eine Weiterleitung zum eLearning-Portal sowie ein Portal für Kleinanzeigen und weiteren Schnittstellen, wie ein Adressbuch aller Uni-Mitarbeiter. Die Applikation ist sehr übersichtlich aufgebaut und auch optisch sehr ansprechend. Auch sie verdeckt bei neu geöffneten Fenstern den Reiter. Es fällt auf, dass keine Daten gespeichert werden und die Applikation von einer externen Software-Firma realisiert wurde.

HHU mobil Die Heinrich-Heine-Universität Düsseldorf bietet eine mobile Version ihrer Webseite [Hei12a], aber keine Applikation an. Diese Version hat nicht nur eine extra kurze URL, welche das Eingeben auf den kleinen Tastaturen von Smartphones erleichtert, sondern die Webseite passt sich automatisch der Bildschirmgröße des Endgerätes an. Die mobile Version bietet die gleichen Informationen und Optionen wie die normale Webseite. Daher ist die mobile Version nur eine kleinere und schlankere Darstellung der normalen Version. Sie bietet keine alltäglichen, wichtigen Informationen wie den Mensa- oder den Vorlesungsplan. Diese müssen weiterhin über die dafür vorgesehenen separaten Seiten abgerufen werden.

Kapitel 2

Über Android

Im Folgenden wird die Architektur von Android dargestellt und es werden die Bausteine einer Applikation erläutert. Zusätzlich wird der Lebenslauf einer Applikation dargestellt.

Android ist ein Betriebssystem für mobile Endgeräte wie Smartphones, welches Google als Open-Source-Software anbietet. Zum Zeitpunkt dieser Arbeit waren circa 80% aller Endgeräte mit Android 2 ausgestattet [Goo12e]. Daher werden für die in dieser Arbeit entwickelten Applikation nur Bibliotheken verwendet, welche auch für Android 2 zur Verfügung standen.

2.1 Die Architektur

Die Architektur von Android baut auf einem Linux-Kernel der Version 2.6 auf[1] . Diese Versionen wurden nach [Fel11, Seite 27] gewählt, da sie unter anderem folgende Eigenschaften bieten:

Gutes Sicherheits-Modell: Der Kernel übernimmt das Sicherheitsmanagement zwischen dem System und der Applikation.

[1] Ab Android 4 ist auch ein Kernel der Version 3 möglich [Goo12n].

Speichermanagement: Der Linux-Kernel handhabt die Adressierung und die Freigabe von Speicher für das System.

Prozessmanagement: Der Kernel managt die Prozessverwaltung sowie das Adressieren von Ressourcen für Prozesse.

Netzwerk-Stapel: Der Kernel regelt die Netzwerkkommunikation.

Außerdem bietet der Kernel zahlreiche Schnittstellen für Multimedia und Netzwerkkommunikation.

Die Laufzeitumgebung von Android basiert auf der Dalvik Virtual Machine. Der große Unterschied zur normalen Java Virtual Machine ist, dass die Java-VM auf einem Kellerautomaten basiert und die Dalvik-VM hingegen auf einer Registermaschine. Dadurch dass jede Applikation eine eigene VM verwendet, wird die Stabilität des Systems verbessert. Beim Absturz einer Applikation wird das System nicht beeinträchtigt, da nur die applikationseigene VM beendet wird. Die Anwendungen werden in Java geschrieben und später von einem Cross-Compiler für die Dalvik-VM übersetzt. Aus Performance-Gründen können Teile einer Applikation auf zahlreiche native Bibliotheken aus C oder C++ zugreifen. Android verfügt über ein angepasstes Java-Framework, welches nicht über rechenintensive oder grafiklastige Bibliotheken verfügt. Dazu zählen zum Beispiel Swing und Graphic2D, die die Darstellung einer PDF in Kapitel 4.2 erschwert haben.

Das Android-Framework setzt auf starke Modularität. Das heißt, dass alle Komponenten des Systems gleichberechtigt (bis auf die VM und das Kernsystem) sind; jede Anwendung tritt dabei dem System gegenüber als Benutzer auf. Dadurch müssen jeder Anwendung nötige Rechte zugeordnet werden. Dies trägt zur Sicherheit des Systems bei. Diese Rechte können bei der Installation einer Applikation eingesehen werden, sodass der Benutzer weiß, welche Rechte er der App ermöglicht. Die nötigen Rechte der hier entwickelten Applikation werden in Tabelle 4.1 dargestellt. Durch die starke Modularität kann jede Applikation ausgetauscht werden und der Benutzer kann das System für seinen Einsatz optimal konfigurieren und spezialisieren.

2.2 Grundlagen von Applikationen

Ein wichtiges Grundprinzip von Android ist das sogenannte *KISS*-Prinzip (Keep it simple, stupid). Das hat zur Folge, dass Code und Design strikt getrennt werden, da die Entwickler der Ansicht sind, dass kleine Änderungen im Design – sofern es im Quellcode festgelegt wurde – zu unerwünschten Nebeneffekten im Verhalten der Applikation führen können. Daher beinhaltet das Android Software Development Kit einen visuellen Designer für graphische Oberflächen, die mit xml-Dateien festgelegt werden. Außerdem bietet Android eine ausgefeilte Ordnerstruktur zum Speichern von Ressourcen, die das Programmieren erleichtern. Die Struktur kann in [Fel11, Seite 78f.] und auf [Goo12f] nachgelesen werden.

Das Android-Betriebssystem ist ein multi-user Linux-System, in dem jede Anwendung auch über eine eigene Benutzeridentifikation verfügt. Wie bei einem Betriebssystem üblich, verfügt jeder Benutzer – also jede Applikation – über diverse Rechte. Zum Beispiel kann die Telefonbuch-Applikation keine Kontakte anrufen, sondern verweist auf die Telefon-Applikation, welche Rufnummern anrufen kann und auch die dafür benötigten Rechte besitzt. Damit verfolgt Android das *principle of least privilege*, soll heißen, dass jede Applikation nur Zugriff auf die Dienste hat, welche sie auch wirklich benötigt.

Eine Applikation besteht aus vier verschiedenen Komponenten. Durch jede Komponente kann das System die Applikation starten, aber nicht alle Komponenten sind Eintrittspunkte, welche dem User zur Verfügung stehen. Jede dieser vier Komponenten bildet einen speziellen Block, der das Verhalten der Applikation beschreibt. Im Folgenden werden die vier Arten der Komponenten, welche auch auf [Goo12d] erwähnt werden, erläutert.

Activities Eine Activity wird durch einen einzelnen und fokussierten Bildschirm inklusive graphischer Oberfläche dargestellt. Diese Komponente ist die einzige, welche für den User sichtbar ist und Benutzereingaben entgegennehmen kann. Damit ist sie auch die am häufigsten benutzte Komponente. Beispielsweise kann ein Telefonbuch aus mehreren Activities bestehen, da eine Activity für die Übersicht und eine weitere für die Einträge

benötigt wird. Eine Applikation kann also mehrere Activities besitzen, von denen immer nur eine aktiv sein kann.

Services Ein Service läuft im Hintergrund, ohne direkt mit dem Benutzer zu interagieren. Beispielsweise werden E-Mails im Hintergrund abgerufen und dann der Activity übergeben, damit sich der Benutzer diese anschauen kann.

Content Providers Der Content Provider übernimmt die Verwaltung von geteilten Informationen, also von Datenstrukturen, die mehreren Applikationen zur Verfügung stehen. Zum Beispiel können Applikationen, wenn die Rechte erteilt wurden, auf den Inhalt des Telefonbuchs zugreifen, sofern dieses lokal in einer Datenbank vorliegt. Da Android eine Datenbank mit SQLite anlegt, wird ein Content Provider benötigt. SQLite verfügt über kein Benutzermanagement und daher darf nur das Hauptsystem auf die Datenbank zugreifen, da diese das Telefonbuch erstellt hat. Mehr dazu in Kapitel 4.1.1.

Broadcast Receivers Ein Broadcast Receiver dient dazu, auf systemweite Meldungen zu antworten. Eine Meldung wird zum Beispiel dann per Broadcast geschickt, sofern eine Applikation einen Anruf ausführen möchte. Dazu wird eine Broadcast-Nachricht benötigt, da der Benutzer die Telefon-Applikation ausgetauscht haben kann. Bei Erhalt des Broadcasts wird ein Service beauftragt, den gewünschten Dienst im Hintergrund zu starten. Die Visualisierung mit Hilfe einer Activity ist eine reine Komfortfunktion.

Weitere Bestandteile Es exisiteren noch weitere wichtige Bestandteile einer Applikation. Drei der vier Komponenten – Activity, Services und Broadcast Receivers – werden mit Hilfe eines sogenannten Intents[2]aufgerufen. Somit lässt sich nicht nur eine weitere Komponente aufrufen, sondern es lassen sich auch Parameter und Variablenwerte mit übergeben. Intents lassen sich explizit (an eine bestimmte Komponente adressieren) oder implizit (systemweit) schicken. Damit nicht jede Komponente alle impliziten Intents annimmt und diese genauer untersucht, lassen sich in der Manifest-Datei Filter für Intents

[2]Ein Intent ist eine interne Nachricht im Android-System, welche Nachrichten oder Aktionen beinhalten kann, die ausgeführt werden sollen.

festlegen. Diese filtern dann für eine bestimmte Komponente zum Beispiel alle Intents aus, bis auf diejenigen, welche eine Änderung des Netzwerkstatus beinhalten.

Jede Applikation verfügt über eine Manifest-Datei. Diese Datei enthält essentielle Informationen über die Applikation, welche benötigt werden, bevor die Applikation gestartet wird. Dazu zählt die Festlegung des minimalen API-Levels[3], die verwendete Hardware, eine Aufzählung aller Activities mit ihren jeweiligen Rechten und bei Bedarf den Intent-Filtern. Weitere Argumente können auf [Goo12c] nachgelesen werden.

2.3 Lebenszyklus einer Applikation

Damit Applikationen besser verstanden und auch effektiver erstellt werden können, ist es wichtig zu wissen, wie lange eine Activity lebendig ist beziehungsweise vom Benutzer Eingaben entgegennehmen kann und wann sensible Daten gespeichert werden müssen. Activities werden von der Applikation auf einem Stack verwaltet, bei dem die vorherige Activity immer unter der aktuellen Activity liegt. Keller des Stacks ist daher immer die Einstiegs-Activity aus der Manifest-Datei und *top of stack* ist die aktuelle, sichtbare Activity. Basierend auf der Stackarchitektur und wie in Abbildung 2.1 zu sehen ist, existieren drei wichtige Schleifen im Leben einer Applikation. Die *gesamte Lebenszeit* einer Activity befindet sich zwischen dem Aufruf von *onCreate()* und vor dem finalen Aufruf von *onDestroy()*. Dagegen befindet sich die sichtbare oder auch *aktive Lebenszeit* nur zwischen *onStart()* und *onStop()*. Während dieser Zeit ist garantiert, dass die Activity oben auf dem Stack liegt und für den Benutzer sichtbar ist. Außerdem gibt es noch die *vordergründige Laufzeit*, welche zwischen *onResume()* und *onPause()* ist. In dieser Zeit ist die Activity nicht nur oben auf dem Stack und sichtbar, sondern der Benutzer kann nun auch mit bereitgestellten Elementen der Activity interagieren. Wenn die Activity in der *vordergründigen Laufzeit* den Fokus verliert, wird sie pausiert. Gestoppt wird sie, sofern sie nicht mehr top of stack ist. Die Bedeutung der erwähnten Methoden kann offensichtlich aus der Namensgebung oder [Goo12a] entnommen werden.

[3] Kurz für *Android Developers Package Index*, also die verwendete Android Version.

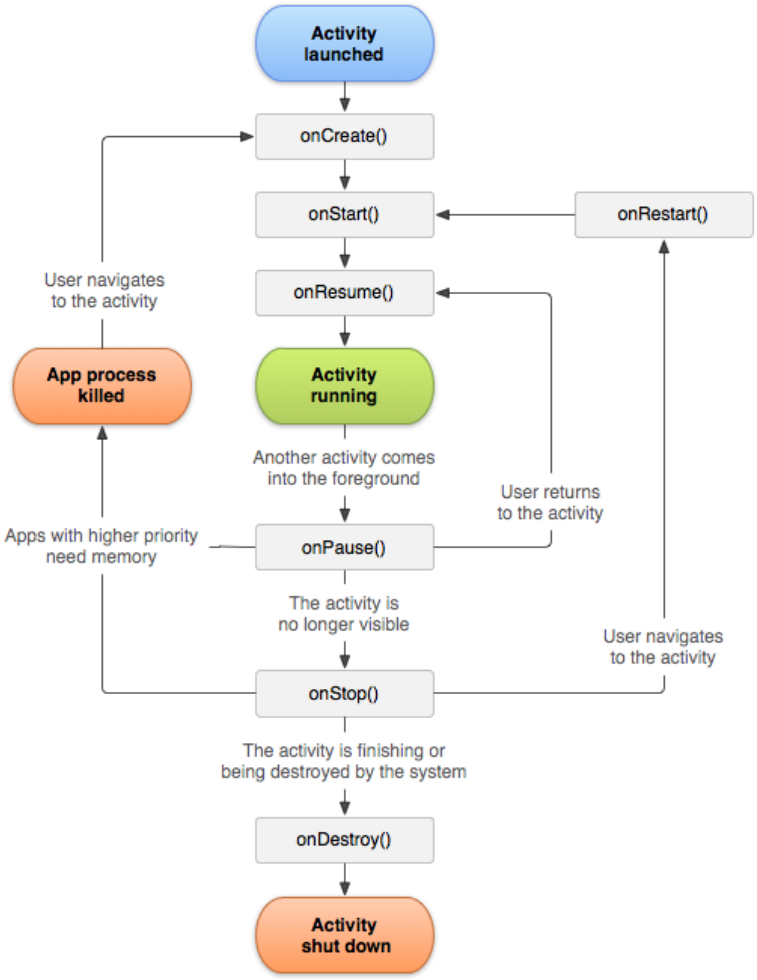


Abbildung 2.1: Die wichtigsten Stufen des Lebenszyklus einer Android-Anwendung nach [Goo12a]

Anzumerken ist noch, dass der Zustand einer Applikation durch das System oder dem Benutzer geändert werden kann. Beim Ersteren kann eine im Arbeitsspeicher liegende Instanz einer Applikation vom System beendet werden, sodass für aktive Applikationen mehr Arbeitsspeicher zur Verfügung steht. Beim Letzteren reicht schon eine Änderung der Lage des Handys, also ein Wechsel zwischen dem Portrait- und Landschafts-Modus. Dies hat zur Folge, dass die Instanz der Applikation zerstört und für den Landschaftsmodus neu aufgebaut wird. Grund dafür ist ein anderes Design, das vom Programmierer festgelegt werden kann. Die hier entwickelte Applikation verzichtet auf den Landschaftsmodus, da dadurch keine implementierte Activity übersichtlicher werden würden.

Kapitel 3

Zielsetzung der Arbeit

Die Service-Applikation für alltägliche Studieninformationen bietet sechs Dienste zum Bezug von universitäts-bezogenen Informationen der Heinrich-Heine-Universität Düsseldorf an. Im Folgenden werden die Anforderung an jeden dieser Dienste beschrieben.

Als erster Dienst der Applikation sollte eine Shoutbox implementiert werden, jedoch bietet diese für mehrere tausend Studenten keine optimalen Verwaltungsmechanismen. Deswegen wurde die Shoutbox erweitert und mehrere Topics wurden realisiert[1]. Der Aufbau sollte strukturiert und übersichtlich sein. Der Informationsaustausch mit der zentralen Datenbank sowie die Speicherung von Benutzernamen und Passwörtern mussten sicher sein. Durch die geringe Größe der Displays, sollten nur wichtige Bedienelemente angezeigt und für weitere Optionen die Options- und Kontextmenüs verwendet werden. Außerdem wäre es hilfreich, wenn unpassende oder beleidigende Einträge gemeldet werden können. Zusätzlich sollte ermöglicht werden, alle Einträge offline einsehen und verwalten zu können. Wünschenswert wäre eine Option, mit der Einträge über soziale Netzwerke und Textanwendungen geteilt werden können. Parallel zum Forum bietet sich ein Nachrichtendienst an, in dem die Benutzer schnell und einfach Nachrichten schreiben können. Letztendlich sollte die Möglichkeit geboten werden, dass alle Nachrichten öffentlich lesbar sind.

[1] Im Folgenden wird die Shoutbox als Forum bezeichnet.

Der zweite Dienst bietet Zugriff auf die Mensapläne. Dafür soll der Benutzer die gewünschte Mensa auswählen und danach den Mensaplan einsehen können. Auch diese sollen offline verfügbar sein.

Zu den Diensten der Universität Düsseldorf zählt auch die Verwaltung von Vorlesungsplänen, welche im Portal für „Lehre, Studium, Forschung“ (HIS-LSF) gespeichert sind. Diese sollen als kompakte Listenansicht gespeichert werden können und damit dauerhaft auf dem Smartphone verfügbar sein. Zusätzlich wäre eine Export-Funktion zum Smartphone-Kalender oder eine Anzeigefunktion der Vorlesungsräume auf dem Lageplan nützlich.

Gerade für Studienanfänger ist ein Lageplan von Vorteil. Dieser soll möglichst ressourcensparend, in verschiedenen Maßstäben und ohne gültige Internetverbindung verfügbar sein. Eine Funktion zum Markieren von Orten würde die Orientierung untersützen. Zusätzlich kann eine simple Navigation ermöglicht werden, die den kürzesten Weg, sowie Entfernung und benötigte Zeit anzeigt. Auch hier ist darauf zu achten, dass die Benutzung intuitiv und Android-konform ist. Richtlinien und Empfehlungen zur Entwicklung können unter [Goo12m] nachgelesen werden.

Zusätzlich wäre es nützlich, eine Schnittstelle zum offiziellen RSS-Feed der Uni Düsseldorf [Hei12b] zu implementieren. Die Nachrichten sollten offline verfügbar sein. Beim Herunterladen von Nachrichten sollte darauf geachtet werden, dass nur die neuen Nachrichten geladen werden. Hinzu kommt, dass der Benutzer die Möglichkeit hat, Nachrichten über soziale Netzwerke und Textanwendungen zu teilen.

Als letzter Dienst wurden zusätzlich wichtige Webseiten der Universität als Vorauswahl bereitgestellt. Diese sollen lediglich angezeigt werden, sodass der Benutzer auf der Seite navigieren kann. Dies soll wie in einem Browser möglich sein.

Da das Smartphone heutzutage zu einem tagtäglichen Begleiter geworden ist und die Applikation viele Dienste vereinigt sowie persönliche Daten und Einstellungen verwaltet, sollte die Applikation zusätzlich ein umfangreiches Optionsmenü mit allgemeinen und aktivitäts-spezifischen Einstellungen bieten.

Kapitel 4

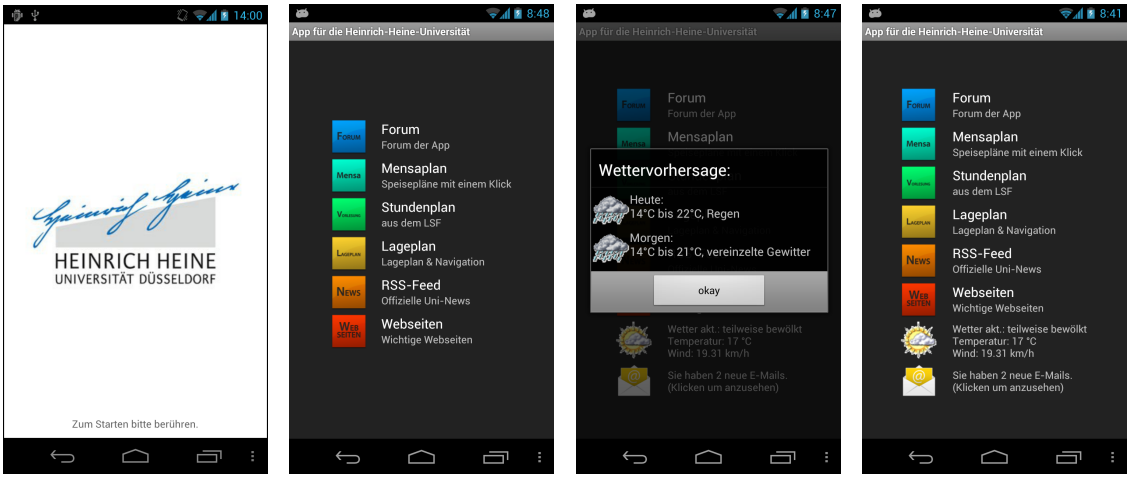
Die Applikation

Im Folgenden wird eine Übersicht über den Aufbau der „HHU App“ gegeben. Das Applikations-Logo und die Icons der Übersicht wurde mit Hilfe eines freien Icon Generators erstellt. Außerdem wurde die Titelleiste mit dem Applikationsnamen – außer in der Übersicht – aus Platzgründen ausgeblendet. Die Applikation wurde mit zahlreichen Geräten aus Tabelle 4.2 getestet. Als Einstiegspunkt wurde die in Abbildung 4.1a zu sehende Willkommens-Activity in der Android Manifest-Datei festgelegt.

Zusätzlich bekam die Willkommens-Activity ein Attribut in der Manifest-Datei zugewiesen, damit sie nicht auf dem lokalen Activity-Stack der Applikation liegt und somit nicht über den „zurück“-Button aufgerufen werden kann. Das Wechseln in den Landschaftsmodus wurde deaktiviert, da durch die Querlage des Endgeräts jede Activity unübersichtlicher wird. Zusätzlich wurden, wie in Kapitel 2.1 angesprochen, Rechte für die Applikation gesetzt. Folgende Rechte wurden der App eingeräumt:

Recht	Begründung
Abfrage Netzwerkstatus	Überprüfung, ob ein Download möglich ist
Internetzugriff	Daten vom Server laden / in die Webview laden
Lese- & Schreibrechte SD-Karte	Daten speichern
Zugriff feine Ortung	Ortung auf dem Lageplan via GPS
Zugriff grobe Ortung	Ortung auf dem Lageplan via WLAN / Funknetz
Lese- & Schreibrechte Kalendar	Eintragen von Vorlesungen aus dem Stundenplan

Tabelle 4.1: Übersicht über alle Rechte der Applikation



(a) Willkommens-Activity der Applikation

(b) Übersichts-Activity der Applikation

(c) Übersichts-Activity der Applikation mit Wettervorhersage

(d) E-Mail-Ansicht der Übersichts-Activity der Applikation

Abbildung 4.1: Willkommens- und Übersichts-Activities der Applikation

Alle setzbaren Rechte können auf [Goo12g] eingesehen werden. Zusätzlich sei darauf hingewiesen, dass die Bedienung und Bedienbarkeit den Richtlinien der Android-Guides auf [Goo12m] entsprechen.

Im Folgenden wird der Start der Applikation beschrieben. Sofern der Benutzer das Icon anklickt, öffnet sich die erste Ansicht beziehungsweise Activity der Applikation. Sobald diese Ansicht erzeugt wurde, schläft ein Thread drei Sekunden lang im Hintergrund. Danach wird via Intent die Übersichts-Activity aus Abbildung 4.1b aufgerufen. Zusätzlich kann der Benutzer durch Berühren der Oberfläche den Intent direkt senden.

Zur Verschönerung der Übersicht wurde eine Ansicht für das aktuelle Wetter sowie eine Vorhersage für den aktuellen Tag und den Folgetag aufgenommen. Eine Vorschau ist in Abbildung 4.1c zu sehen. In den Einstellungen kann der Wert gesetzt werden, ob die Wetteransicht sichtbar sein soll. Dieses Feature wurde integriert, da auch eine der Applikationen aus Kapitel 1.2 die Vorhersage beinhaltet und es dem Nutzer helfen kann, seine Anreisemöglichkeiten zu planen. Obwohl die Wetteranfrage weniger als fünf Kilobyte Datenverkehr verursacht, kann die Verzögerung beim Laden der Activity störend sein. Deswegen wurden alle Icons in die Ressourcen aufgenommen und die Wettereigen-

schaften werden mit einer 60-minütigen Lebenszeit gespeichert. Die Wetterdaten werden von [Yah12] bezogen.

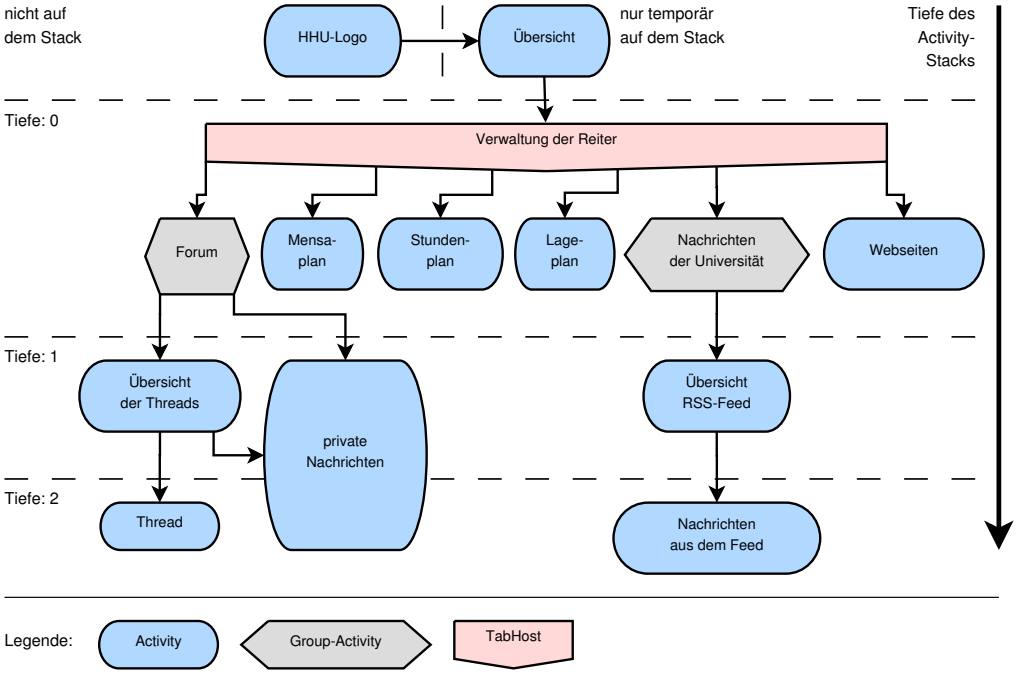


Abbildung 4.2: Activity-Hierarchie der Applikation

Damit der Benutzer eine bessere Übersicht seines E-Mail-Kontos der Universität hat, wurde eine Anzeige integriert, die die aktuelle Anzahl an neuen E-Mails im Universitäts-Postfach anzeigt (siehe 4.1d). Auch hier kann in den Einstellungen der Wert gesetzt werden, ob die Anzeige sichtbar sein soll. Die E-Mails werden mit Hilfe einer Fremdbibliothek von [Chi12] abgerufen, da die Bibliotheken von *Oracle* nicht kompatibel für Android sind. Beim Abrufen von E-Mails kann es zu Verzögerungen kommen, da die Fremdbibliothek ein Timeout von 30 Sekunden besitzt. Bei Berührung der Anzeige erhält der Benutzer eine Vorschau über alle Absender und Betreffe sowie die Option, die E-Mails direkt zu lesen. Dafür wird in der Applikation die Webseite vom Webmail-Programm der Universität geöffnet. Die Webseiten-Activity wird in 4.6 dargestellt.

Die nach der Übersicht folgende Ansicht beinhaltet eine Navigations-Symbolleiste am oberen Bildschirmrand. In dieser wurden sechs Reiter angelegt, welche dem Nutzer das Wechseln zwischen den Hauptfenstern ermöglicht. Standardmäßig repräsentiert ein Rei-

ter eine weitere Activity. Da aber sowohl das Forum, als auch der News-Feed der Universität, mehrere Activities beherbergen, wurden hier Gruppen von Activities von [Har12] verwendet. Dies wird in Kapitel 4.1 ausführlicher thematisiert. Wie in Abbildung 4.2 zu sehen ist, sind maximal drei Activities auf dem Stack. Dies ermöglicht nicht nur eine übersichtlichere Navigation, sondern schont auch Ressourcen, da weniger Activities verwaltet werden müssen. Die Activitiy mit den Einstellungen kann in jeder Activity (außer der Activity mit dem HHU-Logo) geöffnet werden und wurde aus Gründen der Übersichtlichkeit nicht mit in die Abbildung 4.2 aufgenommen.

In den nächsten Abschnitten werden die Reiter der Applikation dargestellt. Als Abschluss wird eine Auswertung bezüglich des Laufzeitverhaltens der Applikationen auf verschiedenen Endgeräten durchgeführt.

4.1 Activity: Forum

Das Forum soll den Austausch von Informationen unter den Studenten beschleunigen. Dazu zählt zum Beispiel die Information, wie sich die Sprechzeiten des Prüfungsamts geändert haben, da dafür kein Aushang existiert. Im Folgenden werden die verschiedenen Zustände des Forums, die Handhabung sowie die dahinterliegenden Mechanismen erklärt.

Wie am Anfang von Kapitel 4 erwähnt, befinden sich alle Activities des Forums in einer Group-Activity, da jeder Reiter nur eine Activity beinhalten kann. Im Folgenden werden die Group-Activity auch *Parent-Activity* und die eingebetteten Activities *Child-Activities* genannt. Es kann beobachtet werden, dass wenn der Benutzer zwischen einer Group-Activity und normalen Activity wechselt, die ausgewählte Child-Activity der Group-Activity nicht wiederhergestellt und die Anfangs-Activity der Parent-Activity geladen wird. Dies ist kein Fehlverhalten, sondern von Android gewünscht, da beim Wechseln der Reiter nur der Status der aktuellen Activity gesichert werden kann. Dies ist in diesem Fall die Parent-Activity. Der extra angelegte Activity-Stack zur Verwaltung der Child-Activity kann an der Stelle nicht gesichert werden. Da die Methoden, welche Attribute sichern können, keine Option zur Speicherung eines Activity-Stacks beinhalten. Auf ei-

ne Lösung mit statischen und globalen Variablen wurde verzichtet, da dies gegen die Grundlagen der objektorientierten Programmierung von Android verstößt.

Wenn die Applikation vom Benutzer ausgewählt wird, wird zuerst die Parent-Activity gestartet. Diese beinhaltet keinerlei *Widgets*[1]. Daher verfügt die Group-Activity über drei Activities mit jeweils folgendem Inhalt:

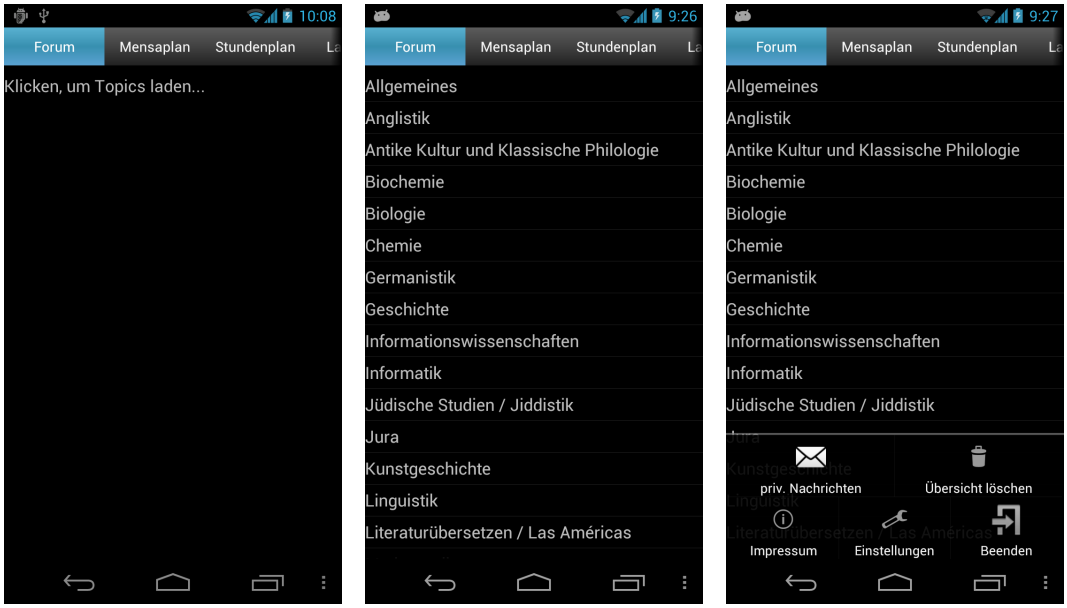
- Eine Übersicht über alle verfügbaren Threads, wie in Kapitel 4.1.1 dargestellt.
- Den Inhalt eines jeden Threads, erläutert in Kapitel 4.1.2.
- Den Nachrichtendienst, wie in Kapitel 4.1.3 dargestellt.

Standardmäßig wird die Übersicht des Forums von der Parent-Activity aufgerufen. Somit besitzt die Activity als Widget wieder eine Activity. Dies macht kontext-basierte Operationen – wie das Aufrufen des Kontext- und Optionsmenü – komplizierter. Da die Fremdklasse der Group-Activity nur die Verwaltung des Activity-Stacks bereitstellte, wurden Mechanismen ergänzt, um beim Aufruf von kontext-basierten Methoden die Anfrage von der Parent-Activity an die aktuelle Child-Activity weiterzugeben.

4.1.1 Die Übersicht

Beim erstmaligen Aufruf des Forums sieht der Benutzer die Oberfläche aus Abbildung 4.3a. Es kann der Schriftzug oder der Hintergrund berührt werden, damit eine Übersicht vom Server geladen wird. Die Übersicht besteht aus einer einfachen Datenbanktabelle. In Abbildung 4.4 ist diese Tabelle als Datenbankdiagramm dargestellt. Jedes Topic repräsentiert die Überschrift eines Thread des Forums. Da Android *SQLite* verwendet, wurde als Primärschlüssel eine Integer-Variable verwendet, welche automatisch inkrementiert wird. Dadurch fällt die manuelle Verwaltung des Primärschlüssels weg und es kann einfacher mit der Datenbank interagiert werden. Zu beachten ist, dass die Datenbank des Servers mit *MySQL* verwaltet wird.

[1] Ein Widget (kurz für Window-Gadget) ist ein Element in der Activity, welches Aktionen vom Benutzer entgegennehmen kann.



(a) Übersicht des Forums beim erstmaligen Starten der Applikation

(b) vollständig geladene Übersicht des Forums

(c) Options-Menü eines Post

Abbildung 4.3: Verschiedene Zustände der Übersicht der Forums-Funktion

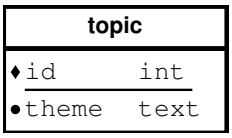


Abbildung 4.4: Topic-Tabelle der Forums-Datenbank

Sofern der Benutzer seine Übersicht aktualisiert, wird eine gesicherte Verbindung zum Server aufgebaut. Der Server wurde so eingerichtet, dass nur sichere Verbindungen akzeptiert werden, damit auch sensible Daten (Benutzername und Passwort) sicher übertragen werden. Das Zertifikat wurde mithilfe von *OpenSSL* [The12] ausgestellt. Damit die Applikation nur das für den Server ausgestellte Zertifikat akzeptiert, wurde mithilfe von *BouncyCastle*[2] [Tau12] eine Schlüsselsammlung für die App erstellt und eingebunden. Sicherheitsmechanismen des Datenbankservers werden in Kapitel 4.1.4 erläutert. Nach dem Aufbau der Verbindung werden mittels HTTP-Request die Daten ausgetauscht. Die Requests werden auf dem Server von PHP-Skripten entgegengenommen und

[2]BouncyCastle ist eine leichtgewichtige Kryptografie-Bibliothek, die im Android-Framework eingebunden ist.

alle Antworten JSON-kodiert zurückgeschickt. Die Antwort enthält die gesamte Liste von Topics, welche auf dem Smartphone in eine Datenbank-Tabelle eingetragen werden. Wenn der Benutzer über eine vollständige Übersicht der Threads verfügt (wie in Abbildung 4.3b zu sehen ist) kann er einen Thread auswählen, um die dazugehörige Activity zu laden.

Um in Android mit SQLite-Datenbanken zu arbeiten, stehen separate Bibliotheken zur Verfügung. Die Datenbank ist für den Benutzer eines Endgeräts über einen Dateimanager nicht erreichbar, da sie in dem Systemverzeichnis `/data/` liegt. Alle Datenbanken werden unter `/data/<packagename>/databases/<dbname>` gespeichert. Auf die Systemverzeichnisse kann nur zugegriffen werden, sofern der Benutzer sich als Root bei seinem Endgerät anmeldet. Dies ist aktuell nur mit einem modifiziertem Kernel möglich. Das Aufspielen der Kernel ist für unerfahrene Benutzer nicht empfehlenswert.

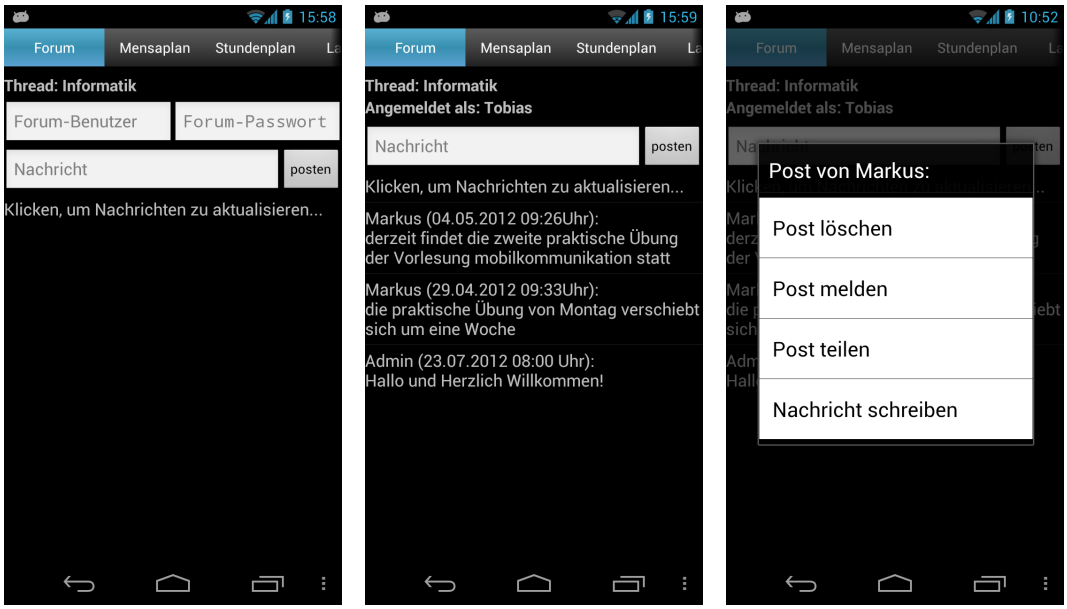
Android verwendet SQLite, da es schnelle Zugriffe ermöglicht. Diese werden ermöglicht, indem eine geöffnete Datenbank immer im Arbeitsspeicher der Anwendung liegt. SQLite bietet im Gegensatz zu MySQL kein Benutzermanagement. Dies ist für Applikationen unter Android irrelevant, da jede Datenbank in einem geschützten Pfad liegt und daher nur für die Applikation selbst sichtbar ist. Eine mehrfache Nutzung der Datenbank ist nur mit dem in Kapitel 2.2 erwähnten Content Provider möglich. Deswegen werden keine verschiedenen Benutzer für die Datenbank benötigt.

Zusätzlich wurde ein Kontextmenü eingebunden, welches in Abbildung 4.3c zu sehen ist. Über die erste Schaltfläche kann die Activity zur Nachrichten-Verwaltung geöffnet werden, die in Kapitel 4.1.3 beschrieben wird. Außerdem kann der Benutzer seine aktuelle Topic-Tabelle entfernen. Dies ist hilfreich, wenn er die Daten der Applikation auf Werkszustand zurücksetzen möchte. Dafür steht aber auch in den Einstellungen eine Option bereit, welche in Kapitel 4.7 erwähnt wird. Die Schaltfläche mit dem Informationssymbol zeigt das Impressum an. Diese Schaltfläche lässt sich in jedem Optionsmenü finden. Genauso die „beenden“-Schaltfläche, welche einen Dialog öffnet, in dem der Benutzer noch einmal bestätigen muss, dass die App beendet werden soll. Aus Sicht des Android-Systems ist ein manuelles Beenden einer Applikation nicht vorgesehen. Durch diese Option wird in den Lebenszyklus aus Abbildung 2.1 eingegriffen. Außerdem übernimmt, wie in Kapitel 2.1 beschrieben, er Kernel die Speicherverwaltung. Da jedoch

ein Großteil der Benutzer von Computern gewohnt ist, Programme manuell zu beenden, wurde dies in jedem Optionsmenü mit eingebunden.

4.1.2 Die Threads

Beim erstmaligen Öffnen eines Threads sind keine Posts vorhanden und die Ansicht gleicht der Abbildung 4.5a. Sofern der Benutzer in den Einstellungen Benutzername und Passwort hinterlegt hat, werden die Textfelder ausgeblendet und es wird ein Schriftzug mit dem hinterlegten Benutzernamen angezeigt. Sofern diese nicht hinterlegt wurden, werden die Anmeldedaten aus Sicherheitsgründen beim Wechseln der Activity verworfen. Diese Sicherheitsmaßnahme wurde getroffen, da sich nicht nur der Eigentümer des Gerätes anmelden kann, sondern auch Freunde und Bekannte. Damit deren Anmeldedaten nicht an den Eigentümer gelangen, werden sie beim Wechseln der Activity gelöscht.



(a) Thread-Übersicht beim erstmaligen Starten der Applikation

(b) vollständig geladener Thread

(c) Optionsmenü

Abbildung 4.5: Verschiedene Zustände in einem Thread des Forums

Alle Posts eines Threads können auch ohne gültigen Login geladen werden. Dazu werden die aktuell höchste ID aller Posts sowie die aktuelle Topic-ID als Content an einen HTTP-Post angehangen. Dadurch können alle neuen Posts auf dem Server abgefragt und als Antwort des Servers gesendet werden. Die Antwort des Servers wird in die Tabelle des Endgeräts eingefügt. Die Beziehungen der Tabellen lassen sich aus Abbildung 4.6 entnehmen. Demnach besitzt jeder Eintrag in der post-Tabelle eine ID als Primärschlüssel und alle Posts des gleichen Threads besitzen die gleiche Topic-ID. Somit werden bei einer Anfrage alle Posts zurückgeschickt, welche die selbe Topic-ID haben und deren ID größer ist, als die der geschickten Post-ID. Sofern der Nutzer in den Einstellungen angegeben hat, dass er alle Threads gleichzeitig aktualisieren möchte, so wird nur die höchste Post-ID als Content geschickt. Zusätzlich wird eine Liste aller gelöschten Posts sowie aller privaten Nachrichten geladen, dies wird im späteren Verlauf erklärt. Nach jeder Aktualisierung des Threads besitzt der Nutzer eine Spiegelung des aktuell ausgewählten oder aller Threads; ein Beispiel kann in Abbildung 4.5b gesehen werden.

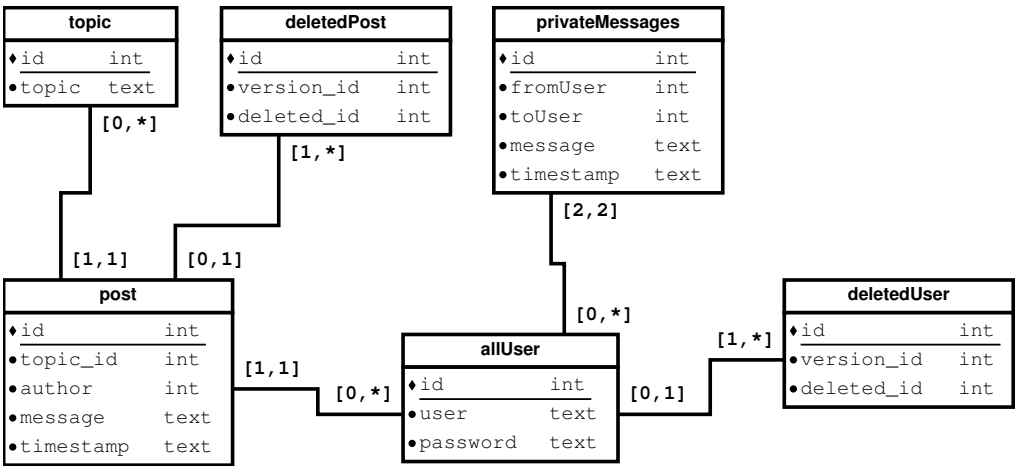


Abbildung 4.6: Datenbankdiagramm des Forums

Im Folgenden wird das Vorgehen beim Posten einer Nachricht dargestellt. Um eine Nachricht zu posten, wird ein gültiges Paar aus Benutzernamen und Passwort erwartet. Ist die Kombination aus Benutzername und Passwort fehlerhaft, so erhält der Nutzer nach dem Übertragungsversuch eine detaillierte Fehlermeldung, welche später genauer erwähnt wird. Für den weiteren Verlauf wird davon ausgegangen, dass der Benutzername sowie das Passwort korrekt sind und eine Verbindung zum Internet besteht. Besteht keine

Verbindung, so wird die Nachricht auf der internen SD-Karte[3] gespeichert und bei der nächsten Aktualisierung oder wie in Kapitel 4.1.4 geschildert, hochgeladen.

Die gesamte Kommunikation mit dem Server wird asynchron abgearbeitet. Ansonten würde die Applikation keine weiteren Benutzereingaben verarbeiten und Android würde dem Benutzer eine Schließung der Applikation anbieten. Während der Übertragungen wird der Benutzer per Pop-up über den aktuellen Fortschritt informiert. Der erste HTTP-Post-Request alle neuen Posts ab.

Der zweite HTTP-Post dient zur Identifizierung der vom Systemadministrator gelöschten Posts. Die Spalten in der *deletedPost*-Tabelle bestehen aus drei Integern. Einer ist Primärschlüssel, einer dient als Versions-ID und ist Fremdschlüssel auf den Primärschlüssel der post-Tabelle und der dritte Integer repräsentiert den gelöschten Post. Dadurch, dass der zweite Integer als Versionsnummer behandelt wird, ist damit sicher gestellt, dass bei einer Aktualisierung durch den Benutzer, alle auf dem Server gelöschten Posts auch auf dem Endgerät gelöscht werden. Der Content des HTTP-Post ist daher nur die höchste Post-ID der lokalen Datenbank. Die Antwort enthält die ID's aller gelöschten Posts, welche auch auf dem Endgerät gelöscht werden.

Da der Benutzer Nachrichten verfassen kann, ohne eine Verbindung zum Internet nachzuweisen, werden als dritte Aktion alle auf der SD-Karte gesicherten Nachrichten an das Forum geschickt und bei Erfolg in die Tabelle des Endgeräts eingetragen. Die gespeicherten Nachrichten wurden in einer Textdatei mit den Informationen über die ID des Topics, den Benutzernamen, das Passwort, die Nachricht und den Zeitstempel bei der Erstellung abgelegt. Daher werden die Posts nicht nach Zeit sondern nach der eindeutigen ID sortiert. Der Zeitstempel wird bewusst auf dem Gerät erstellt, da sonst verspätet eingetragene Posts nicht erkannt werden können und der Thread schnell unübersichtlich wirkt. Die lokal gespeicherten Nachrichten werden erst nach dem Aktualisieren der Tabelle aus der ersten Aktion geschickt, da sonst die ID der Posts auf dem Smartphone nicht mehr synchron mit der auf der Datenbank sind. Nach Ausführung der zweiten Aktion ist die lokale Tabelle mit der auf dem Server synchron und es kann die neue Nachricht hochgeladen werden. Alle hier geschilderten Bezüge der Tabellen werden in Abbildung 4.6 dargestellt.

[3] Im Folgenden ist mit dem Begriff der SD-Karte immer die interne SD-Karte gemeint.

Die letzte Aktion der asynchronen Klasse betrifft das Herunterladen von privaten Nachrichten. Dies geschieht nur, sofern der Benutzer den Wert in den Einstellungen gesetzt hat. Ansonsten muss der Benutzer die Nachrichten in der dafür vorgesehenen Activity laden. Wurden Nachrichten automatisch geladen, erscheint ein Pop-up, das dem Benutzer die Anzahl an neuen Nachrichten mitteilt und dem Benutzer die Option anbietet, die Nachrichten-Activity zu öffnen. Nach dem vierten ausgeführten HTTP-Post werden die heruntergeladenen Nachrichten in die Listenansicht der Activity eingetragen oder es wird eine detaillierte Fehlermeldung angezeigt, damit der Benutzer erkennt, welche Aktionen fehlgeschlagen sind.

Im Optionsmenü befinden sich sechs Schaltflächen. Die Erste blendet die Text-Widgets am oberen Bildschirmrand ein oder aus, sodass die Nachrichten-Liste mehr Platz hat. Die zweite Schaltfläche löscht alle Einträge aus der gesamten Thread-Tabelle. Diese Einstellung ist nützlich, sofern der Benutzer manuell Einträge gelöscht hat, die er bei der nächsten Aktualisierung wieder sehen möchte. Die nächste Schaltfläche öffnet die Activity zur Verwaltung der persönlichen Nachrichten. Die letzten Drei sind die Schaltflächen für das Impressum, die Einstellungen und zum Beenden der Applikation.

Das Kontextmenü erscheint beim Anklicken eines Posts. Hier kann zwischen löschen, melden, teilen und dem Schreiben einer Nachricht ausgewählt werden. Das Löschen eines Posts erfolgt in der lokalen Tabelle und daher ist das Löschen der Tabelle nützlich. Mit der „melden“-Funktion öffnet sich die E-Mail-Applikation des Endgeräts mit einem vorgefertigten Text. Die Empfänger-E-Mail ist die Adresse der Entwickler und der Text beinhaltet alle nötigen Angaben zur Identifizierung des Posts. Auf diesem Weg kann zum Beispiel Spam oder eine Beleidigungen gemeldet werden. Mithilfe der „teilen“-Funktion kann der gewählte Post in sozialen Netzwerken oder textverarbeitenden Applikationen geteilt werden. Die letzte Option öffnet die Nachrichten-Activity, um den Autor des Posts eine Nachricht zu schreiben.

4.1.3 Der Nachrichtendienst

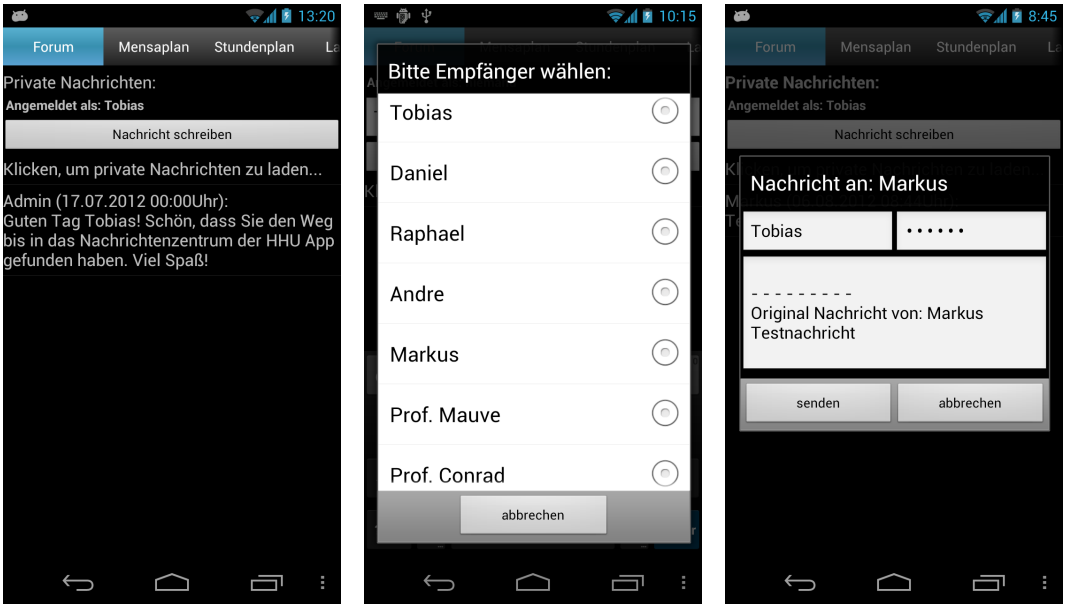
Wie in Kapitel 4.1 dargestellt, kann die Nachrichten-Activity über zwei Wege aufgerufen werden. Der erste Weg führt über das Optionsmenü in der Übersichts- oder Thread-

Activity. Der zweite Eintrittspunkt benutzt das Kontextmenü eines Posts sowie die Möglichkeit über das Pop-up, das bei der Aktualisierung eines Threads angezeigt wird. Wird der zweite Weg gewählt und wurden die Parameter in den Einstellungen gesetzt, so kann die Activity wie in Abbildung 4.7a aussehen. Wurden keine Parameter hinterlegt, werden zwei Textfelder für Benutzername und Passwort angezeigt.

Möchte der Benutzer eine Nachricht schreiben, so wird eine gültige Internetverbindung und eine gültige Kombination aus Benutzername und Passwort verlangt. Als erste Aktion – nach dem Berühren des Button „Nachricht schreiben“ – wird die Benutzer-Tabelle mit den bekannten Mechanismen aktualisiert. Die Tabellen und Verwaltung der ID's erfolgt analog zu den in Kapitel 4.1.2 erläuterten Methoden. Nach dem Laden der Benutzerliste kann der Benutzer aus der Liste den Empfänger wählen (siehe Abbildung 4.7b). Daraufhin erscheint ein Pop-up, in dem der Empfänger eingetragen wurde. Zusätzlich ist ein großes Textfeld zum Schreiben der Nachricht vorhanden. Außerdem existieren drei Buttons, mit denen der Benutzer die Aktion abbrechen, die Nachricht senden oder zurückgehen und einen anderen Empfänger wählen kann. Wird eine Nachricht abgeschickt, so wird im PHP-Skript zuerst der Benutzer authentifiziert und erst dann werden die Werte in die privateMessages-Tabelle aus Abbildungen 4.6 eingetragen. Der Benutzer wird auch hier benachrichtigt, ob die Nachricht erfolgreich übermittelt werden konnte.

Im Gegensatz zur post-Tabelle werden die Nachrichten nach dem Download vom Server aus der Tabelle des Server sgelöscht und existieren dadurch nur auf dem Endgerät. Des Weiteren können Nachrichten von mehreren Nutzern auf einem Endgerät gespeichert werden. Die Nachrichten werden vom Server gelöscht, da die heutige Kommunikation stark auf Messaging-Diensten wie der klassischen SMS oder WhatsApp [Wha12] aufbaut. Dadurch beinhalten Nachrichten im Forum keine sensiblen Daten, welche archiviert werden müssen.

Im Optionsmenü der Nachrichten-Activity befinden sich zu den drei bekannten Schaltflächen noch zwei zusätzliche zwei Schaltflächen, mit denen sich jeweils die Tabelle mit allen Benutzern oder die Tabelle mit allen Nachrichten löschen lassen. Beim Anklicken einer Nachricht erscheint ein Menü mit den vier Werten „löschen“, „antworten“, „weiterleiten“ und „melden“. Die erste Option bedarf keinerlei Erklärung. Die zweite Option öffnet das bekannte Pop-up aus Abbildung 4.7c, wobei als Empfänger der Nachrichten-



(a) Darstellung einer Nachricht

(b) Darstellung aller User, denen eine Nachricht geschrieben werden kann

(c) Pop-up zum Schreiben einer Nachricht

Abbildung 4.7: Verschiedene Zustände der Nachrichtenfunktion

autor gewählt wird und die ursprüngliche Nachricht im Textfeld angefügt ist. Möchte der Benutzer die Nachricht weiterleiten, so wird zuerst wieder – wie oben beschrieben – die Benutzer-Tabelle aktualisiert. Danach kann ein Adressat gewählt werden. Auch hier ist die Nachricht wieder als Anhang beigefügt. Die letzte Option ist analog zu der „melden“-Funktion aus Kapitel 4.1.2.

4.1.4 Strukturen im Backend

An dieser Stelle sei noch einmal darauf hingewiesen, dass jede Kommunikation mit dem Server mittels SSL und dem selbst erstellten Zertifikat stattfindet. Die Kommunikation mit der Datenbank wird mithilfe von PHP-Skripten erledigt (siehe Kapitel 4.1.1). Wie in Abbildung 4.6 gesehen werden kann, befindet sich die Datenbank in der dritten Normalform. Die erste Normalform wird eingehalten, da jedes Attribut einen atomaren Wertebereich besitzt, also keine geschachtelten oder mengenwertige Attribute enthalten sind. Die zweite Normalform wird eingehalten, da zusätzlich jedes Attribut, das kein Pri-

märschlüssel ist, abhängig von allen Primärschlüsseln ist. Zusätzlich wird auch die dritte Normalform eingehalten, da kein Nichtschlüsselattribut transitiv von einem Schlüsselkandidaten abhängig ist. Einzige Ausnahme bildet die allUser-Tabelle, in der die Spalte *password* von der Spalte *user* abhängig ist, wobei diese wiederum von dem Primärschlüssel *id* abhängig ist. Allerdings wird darauf geachtet, dass die Spalte *user* eindeutig ist und somit werden *id* und *user* als Primärschlüssel verwendet. Dieses Schema findet man bei mehreren Foren, wie zum Beispiel bei `phpBB 3.0.10` [php12].

Damit die Sicherheit gewährleistet wird, wurden Passwörter nicht als Klartext in der Tabelle abgelegt, sondern sogenannte *salted Hashs*[4]. Zur Berechnung des Hash wird MD5 verwendet. Das PHP-Skript, welches den Benutzer authentifiziert, berechnet direkt den salted MD5-Hash des Passworts. Würde kein Salt verwendet werden, so wäre die Datenbank Wörterbuchangriffen[5] ausgeliefert. Durch die Verwendung eines Salts mit n Zeichen, existieren 2^n neue Einträge zu dem gehörigen Passwort im Wörterbuch. Dadurch sinkt die Erfolgswahrscheinlichkeit enorm und der Zeitaufwand steigt stark an. Außerdem sind alle Sonder- und Steuerzeichen in Passwörtern verboten und werden gefiltert, damit eine SQL-Injection nicht möglich ist. Zusätzlich beendet sich das Skript direkt, sofern der SQL-Query nach dem Paar aus Benutzername und Passwort-Hash den Wert null liefert.

Neben dem salted Hash und der Verhinderung gegen Injections, wurde für die Anfragen der PHP-Skripte ein Benutzer für den Datenbankzugriff angelegt, welcher nur Select- und Insert-Statements ausführen darf.

Da der Benutzer die Möglichkeit hat, eine Nachricht in einen Thread zu schreiben, ohne dass er online ist, wird im Folgenden der dafür zugrundeliegende Service erklärt. Dazu wird die Nachricht auf die SD-Karte geschrieben und entweder manuell beim Senden eines Posts, beim Aktualisieren der Liste oder beim Aktivieren des WLANs hochgeladen. Dazu dient ein Broadcast Receiver, welcher nur die Intents empfängt, die eine Änderungen des Netzwerkstatus beinhalten. Der Service, der die Posts hochlädt, wird vom Broadcast Receiver nur dann aufgerufen, wenn der Intent die Werte einer gültigen

[4] Eine beliebige Zeichenfolge wird *Salt* genannt, sofern diese an einen Klartext vor der Bildung des Hashs angefügt wird. Das Ergebnis der Hashfunktion wird *salted Hash* genannt.

[5] Als Wörterbuchangriff wird die Methode bezeichnet, ein Passwort mithilfe einer Passwörterliste zu entschlüsseln.

Verbindung beinhaltet und Nachrichten auf der SD-Karte existieren. Der Benutzer erhält mittels Toast[6] eine Benachrichtigung über Erfolg oder Verlust der Nachrichten. Ein Verlust der Nachrichten kann entstehen, da keine Rückmeldung über die Korrektheit von Benutzernamen und Passwort gegeben wird. Eine wiederholte Eingabe des Passworts ist nicht vorgesehen, da die Nachricht eventuell nicht vom Benutzer geschrieben wurde oder ein Pop-up den Benutzer während der Verwendung einer weiteren Applikation unterbrechen könnte.

4.2 Activity: Speisepläne der Mensen

Da das Angebot der Mensa reichhaltig ist und täglich wechselt, wurde eine Schnittstelle zu den Speiseplänen des Studentenwerks Düsseldorf integriert. Das heißt, dass die gewünschten Speisepläne der aktuellen Woche auf dem Endgerät gespeichert und zur Verfügung gestellt wird. Im Folgenden wird das Herunterladen und das Verwalten der Speisepläne erläutert werden.

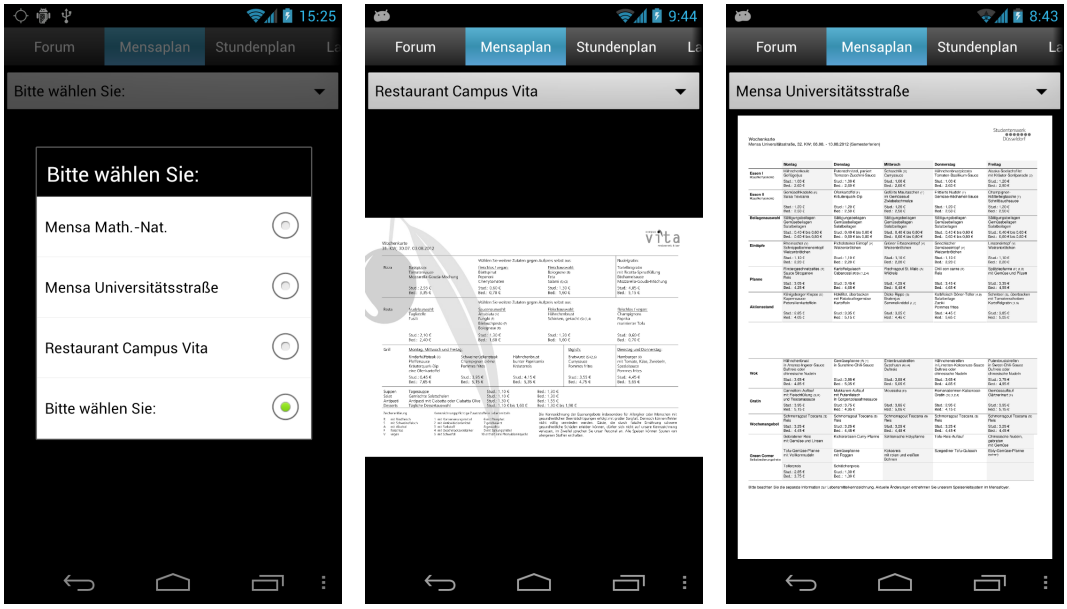
4.2.1 Download der Speisepläne

Der Speiseplan der gewünschten Mensa kann über das Spinner-Widget[7] am oberen Bildschirmrand – wie in Abbildung 4.8a zu sehen – ausgewählt werden. Allerdings ist das vordefinierte Widget unzureichend, da es immer über ein vordefiniertes Element verfügen muss. Daher wäre immer das erste Element der Liste als Vorauswahl definiert. Das Probleme wurde gelöst, indem ein neutrales Element definiert wurde, welches bei der Auswahl des ersten Elements aus der Liste entfernt wird.

Wird eine Mensa ausgewählt, so wird der Name der dazugehörigen Datei generiert. Der Name entspricht dem Namen der Datei auf dem Server des Studentenwerks Düsseldorf und besteht aus dem Datum vom aktuellen Wochenanfang und Monat sowie

[6] Ein Toast ist ein kleines Textfeld, welches nur für einen kurzen Zeitraum auf der aktuellen Activity eingeblendet wird.

[7] Ein Spinner ist eine Kombination aus einem Button mit Bild an der rechten Seite. Bei einem Klick erscheint ein Pop-up mit eingebetteter Liste.



(a) Spinner-Widget beim ersten Aufruf

(b) Auswahl des Speiseplans vom Restaurant Campus Vita

(c) Zweiseitige Auswahl des Speiseplans von der Mensa Universitätsstraße

Abbildung 4.8: Verschiedene Zustände der Mensa-Activity

vom Wochenende und Monat. Beispieweise wird die Kalenderwoche vom 06.08 bis zum 10.08.2012 durch den Bezeichner `06%2008%20-10%2008%2012` repräsentiert, wobei `%20` in URLs ein Leerzeichen repräsentiert. Dabei wird beachtet, dass an einem Wochenende der Ausdruck für die kommende Woche generiert wird und Sonderfälle wie ein Monats- oder Jahresumbruch beachtet werden. Da Android keine Bibliothek zum Verwalten, Konvertieren oder Anzeigen von PDFs bereitstellt, wird der generierte Bezeichner per HTTP-Post an den Datenbankserver geschickt. Auf diesem stehen nicht nur PHP-Skripte zum Ansprechen der Datenbank zur Verfügung, sondern auch alle weiteren Dateien, welche von der App bezogen werden können. Die Konvertierung wird vom Server erledigt, da die Datei dort gespeichert und für alle Benutzer zur Verfügung gestellt werden kann. Zusätzlich trägt dies zur Stabilität der Applikation bei, da ein geänderter Pfad des Studentenwerks nur im PHP-Skript auf dem Server geändert werden muss.

Trifft ein HTTP-Post auf dem Server ein, so wird zuerst kontrolliert, ob der im Content beschriebene Plan schon existiert. Ist dies der Fall, wird der Wert *true* als Antwort geschickt. Danach kann der Plan mit einem HTTP-Get-Request heruntergeladen werden.

Die URL für den Speiseplan setzt sich aus Server-URL, Bezeichner der Mensa sowie dem Bezeichner für das Datums zusammen. Ist der gewünschte Plan noch nicht gespeichert, so setzt der Server die benötigte URL zusammen, lädt den Speiseplan der Mensa herunter und konvertiert diesen mithilfe von Imagemagick [Ima12] in das PNG-Format. Damit der Speiseplan auf dem Smartphone beziehungsweise für den Download nicht zu viel Speicher verbraucht, wird dieser unter anderem nur in Graustufen konvertiert. Zusätzlich werden, wie in Abbildung 4.8c mehrseitige PDFs untereinander gehangen, sodass ein großes Bild entsteht. Sofern der Server die PDF geladen und konvertiert hat, wird das Arbeitsverzeichnis auf alte Dateien mit derselben Mensa-Bezeichnung durchsucht. Existiert eine alte Datei wird diese gelöscht. Nach der Konvertierung und dem Löschen der alten Datei sendet der Server eine HTTP-Response mit dem Wert *true* zurück.

Da das PHP-Skript den übermittelten Ausdruck als gültigen Bezeichner hinnimmt, entsteht eine Schwachstelle. Dadurch könnte ein bösartiger Benutzer mithilfe eines verfälschten Datums den Server beauftragen, den aktuellen Plan zu verwerfen und einen nicht aktuellen Plan zu laden. Dies wird aber behoben, indem als weitere Attribute der aktuelle Wochenanfang als Datum mit übertragen und auf dem Server abgeglichen wird. Nur wenn beide Werte übereinstimmen, wird das Skript ausgeführt. Andernfalls schickt der Server eine Antwort mit dem Wert *false* zurück und der Benutzer wird auf das falsche Datum hingewiesen.

Durch die Konvertierung des Speiseplans kann seine Größe um ein Vielfaches steigen. Dies liegt daran, dass PDF-Dateien Vektorgrafiken sind und die Endgeräte standardmäßig nur Rastergrafiken darstellen können. Erfahrungsgemäß sind die Pläne der Mensa der mathematisch-naturwissenschaftlichen Fakultät, sowie des Restaurants Campus Vita unter 800 Kilobyte groß und der Plan für die Hauptmensa benötigt weniger als 1500 Kilobyte, da die PDF zweiseitig ist. Die Speicherung als Bilddatei bietet zusätzlich den Vorteil, dass die Dateien direkt in der Fotogalerie des Endgeräts erscheinen und somit die Applikation nicht gestartet werden muss.

4.2.2 Verwalten und Anzeigen der Speisepläne

Verwaltet werden die Speisepläne anhand des generierten Ausdrucks für das Datum. So kann geprüft werden, ob eine Datei sowohl vorhanden und als auch gültig ist. Da Android kein Widget zur Verfügung stellt, welches das Vergrößern und Verkleinern von Bildern ermöglicht, wurden Klassen von [Ort12] eingebunden und um weitere Mechanismen ergänzt. Ein Beispiel der Darstellung von Speiseplänen wird in Abbildung 4.8b und 4.8c dargestellt.

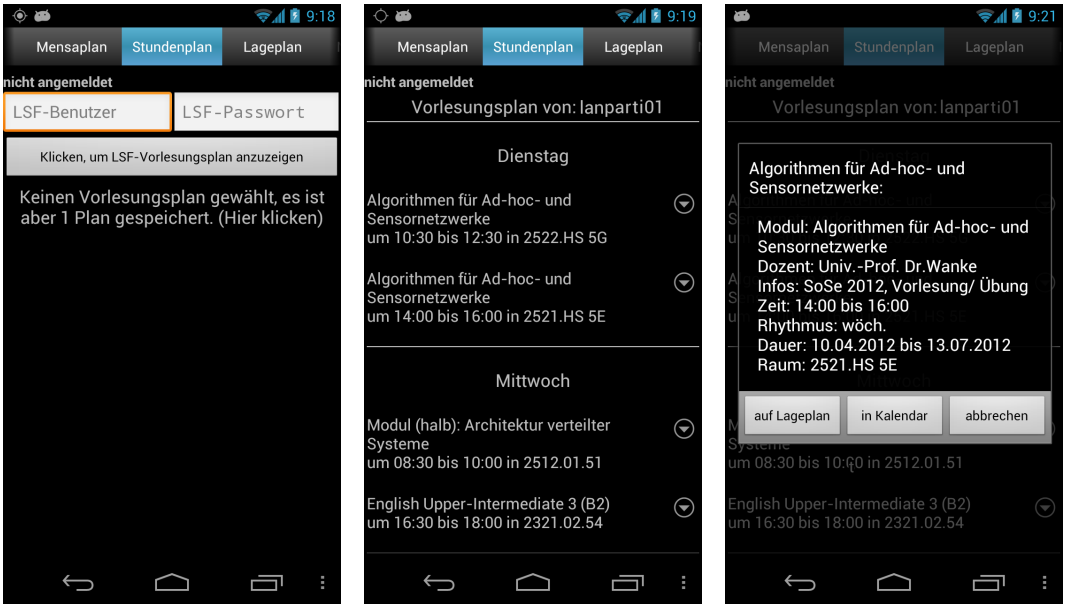
Damit die Bilddatei geladen werden kann, muss das Bild vorher als eine Bitmap geladen werden. Für jede Bitmap alloziert Android Speicher, jedoch verwaltet Android diesen unzureichend. Grund dafür ist, dass der Garbage Collector der Dalvik-VM sehr träge ist und den nicht mehr allozierten Speicher zu spät freigibt. Dadurch kann es beim häufigen Nachladen von Bildern passieren, dass das System nicht mehr genug Speicher zur Verfügung hat. Der maximal verfügbare Speicher wird durch den Kernel begrenzt. Der Dokumentation der Entwickler zufolge wurde der Garbage Collector ab Android 3.0 überarbeitet. Die Dokumentation ist auf [Goo12b] zu finden. Der Fehler wurde umgangen, indem beim Laden einer neuen Bitmap die alte Instanz der Bitmap sowie alle Referenzen auf die Instanz entfernt werden und ein Thread mindestens 50 Millisekunden schläft. Danach wird der Garbage Collector manuell aufgerufen, um den Speicher freizugeben. Die Pause von 50 Millisekunden hat sich durch mehrmaliges Testen von verschiedenen Zeiten als besonders effizient erwiesen, da nach dieser Zeit der Erfolg des Garbage Collectors besonders hoch und eine Verzögerung von 50 Millisekunden für den Benutzer erträglich ist. Leider ist zu beobachten, dass dieses Prinzip den Fehler nicht vollständig behebt und dieser bei aggressivem Nachladen provoziert werden kann. Da es von den Entwicklern nicht vorgesehen ist, zoombare Bilder einzubinden, bieten sie an dieser Stelle keine Lösung an. Außerdem wird in der Dokumentation nicht erwähnt, wie die Galerie-Applikation von Android Bilder verwalten und zoomen kann.

Sofern der Benutzer einen weiteren Reiter auswählt, wird der von der Bitmap allozierte Speicher mit den obigen Mechanismen freigegeben. Dadurch ist beim wiederholten Öffnen des Mensaplan-Reiters der aktuelle Plan nicht mehr sichtbar, dafür jedoch ein Button neben dem Spinner-Widget, mit dem sich der Mensaplan direkt anzeigen lässt.

Der Speicher wird an der Stelle freigegeben, da es sonst zu Komplikationen mit dem Speicherverbrauch in Verbindung mit dem Lageplan aus Kapitel 4.4 kommen kann.

4.3 Activity: Verwaltung der Vorlesungspläne aus dem HIS-LSF

Für jeden Studenten ist es wichtig, einen Überblick über seine Vorlesungen zu haben. Deswegen wurde eine Schnittstelle zum Stundenplan des HIS-LSF in die Applikation integriert. Im Folgenden wird dargestellt, wie der Stundenplan bezogen und die Informationen analysiert werden, wie auf Änderungen von Weiterleitungen im LSF reagiert wird, welche Schwachstellen dabei entstehen können und welche Interaktionen mit einem gespeicherten Stundenplan möglich sind.



(a) Activity beim erstmaligen Starten der Applikation

(b) Vorlesungsplan in der Übersicht

(c) Zusätzliche Informationen zu einer Veranstalung

Abbildung 4.9: Verschiedene Zustände der Vorlesungsplan-Activity

4.3.1 Download des Vorlesungsplans aus dem HIS-LSF

Beim Anklicken des Reiters „Stundenplan“ sieht der Benutzer die bekannte minimalistische und übersichtliche Oberfläche. Der Benutzer wird mithilfe eines Textfelds direkt informiert, wie viele Stundenpläne schon auf dem Endgerät gespeichert sind. Die Auswahl der gespeicherten Stundenpläne kann über das Optionsmenü oder durch das Anklicken des Textfeldes geöffnet werden. Außerdem befinden sich zwei editierbare Textfelder auf der Activity, in denen Benutzername und Passwort des HIS-LSF eingegeben werden können. Sofern der Benutzer seine Anmeldedaten in den Einstellungen hinterlegt hat, sind die beiden Textfelder nicht sichtbar und der Benutzer wird über einen Schriftzug am oberen Rand informiert, welche Daten hinterlegt wurden. Passwörter werden wie gewohnt in Sternchen angezeigt. Die dargestellten Widgets können in Abbildung 4.9a gesehen werden.

Damit der Stundenplan als Listenansicht aus dem Vorlesungsverzeichnis bezogen werden kann, werden mehrere Links benötigt. Diese beinhalten die Startseite des HIS-LSF, um ein Cookie zu beziehen (sofern kein Cookie auf der SD-Karte vorhanden ist), die Weiterleitung nach dem Login, die Adresse des Vorlesungsplans in langer Listenansicht und den Link zum Logout. Wenn Weiterleitungen geändert werden, können die Links von der Applikation aktualisiert und in einer Textdatei auf der SD-Karte gespeichert werden. Diese werden dann anstelle der vordefinierten Links beim Starten der Activity geladen.

Bei fehlgeschlagener Anmeldung wird der Benutzer über ein Pop-up darauf hingewiesen. Das Pop-up bietet die drei Optionen, den Benutzernamen und Passwort wiederholt einzugeben (sofern der Benutzer sich verschrieben hat), den Download des Stundenplans abzubrechen oder die Weiterleitungen neu auszulesen. Wählt der Benutzer die letzte Option, so wird die Aktion im Hintergrund ausgeführt. Die Links können gefunden werden, indem die selben Seiten aufgerufen werden, als wenn der Benutzer über einen Browser seinen Stundenplan als Listenansicht aufrufen möchte. Die Antworten vom Server werden mithilfe von bekannten Strukturen analysiert und es wird versucht, die Weiterleitungen zu finden und zu speichern. Dabei können Fehler entstehen, sofern die Umgebungen – also die HTML-Tags und Attribut der Tags – verändert werden. Natürlich kann der Benutzer die Weiterleitungen nicht nur über das Pop-up, sondern auch über das Options-

menü auslesen und auch entfernen lassen. Da das HIS-LSF nur Anfragen auf Port 443 entgegennimmt, sind alle Verbindung mit SSL gesichert. Leider können Zertifikatfehler bei Verbindungen zum LSF nicht abgefangen werden, da die verwendeten Bibliotheken von Android diese Fehler nicht behandeln. Dies hat zur Folge, dass die Applikation nicht resistent gegen serverseitige Fehler des HIS-LSF-System ist.

Möchte der Benutzer seinen Vorlesungsplan speichern, kann er dies durch Berühren des Buttons aus Abbildung 4.9a veranlassen. Sollte schon eine Datei mit seinem Benutzernamen existieren, wird der Benutzer darauf hingewiesen. Anschließend muss bestätigt werden, dass der Plan erneut geladen werden soll. Alle Anfragen zwischen der Applikation und dem Server werden mit HTTP-Requests ausgeführt. Während der Stundenplan asynchron zur Applikation geladen wird, wird ein Fortschrittsbalken eingeblendet und aktualisiert. Es werden sechs Schritte benötigt, damit nach dem Klick auf den Button der Stundenplan als Listenansicht dem Benutzer angezeigt werden kann. Bei jedem Schritt wird die Internetverbindung überprüft, damit bei Abbruch der Verbindung die Aktion direkt beendet und der Benutzer darüber informiert wird.

Um den Stundenplan herunterzuladen, muss als erstes ein Cookie bezogen werden, damit alle Aktionen vom Server identifiziert werden können. Das Cookie wird entweder mithilfe eines HTTP-Get-Requests von der Startseite des Vorlesungsverzeichnisses bezogen oder von der SD-Karte geladen, sofern ein Cookie gespeichert wurde. Danach wird in der Login-URL Benutzername, Passwort und der Wert vom Cookie eingesetzt, damit die Applikation sich mithilfe eines HTTP-Post-Requests anmelden kann. Als Content werden zusätzlich Benutzername und Passwort angehangen. Dies wird vom Server benötigt und konnte mittels dem Plugin *Live HTTP Headers* [Moz12] erkannt werden. Als HTTP-Header aller Anfragen wurden der Header von Firefox 11.0 benutzt, welcher ebenfalls mithilfe von Live HTTP Headers ausgelesen werden konnte. Nach Erhalt der Antwort des Servers wird die HTTP-Response-Nachricht untersucht. Dieser kann Klartext beinhalten, sofern Benutzername und Passwort nicht korrekt angegeben wurden. Ist dies der Fall, erscheint das oben genannte Pop-up. Wenn der Benutzer erfolgreich eingeloggt wurde, wird die Seite, welchen den Stundenplan als Listenansicht beinhaltet, angefordert und vorübergehend als String gespeichert. Diese wird später analysiert. Zum Schluss wird der Benutzer aus dem HIS-LSF abgemeldet. Dies geschieht sowohl im regulären Ablauf, als auch wenn Fehler auftreten. Ab diesem Zeitpunkt beinhaltet eine

Variable die HTML-Seite mit den nötigen Informationen, welche danach einem Parser übergeben wird.

4.3.2 Parsen und weitere Aktionen mit dem Vorlesungsplan

Bevor der Parser die Eingabe verarbeiten kann, wird die Quelldatei von allen Leerzeilen, führenden und folgenden Leerzeichen bereinigt, der Text von ISO 8859-1 in UTF-8 konvertiert und Zeilenumbrüche zwischen Datums- sowie Zeitangaben entfernt. Ein Problem für das Anzeigen der Vorlesungen war, dass die Vorlesungen noch nicht nach Tagen und Uhrzeiten sortiert wurden und die HTML-Struktur sehr geschachtelt ist. Daher wird die Eingabedatei analysiert und jede Vorlesung in einem Objekt gespeichert. Sofern das Dateiende erreicht wird, werden alle Objekte sortiert. Die verwendeten Mechanismen zur Filterung von relevanten Informationen können im Quelltext eingesehen werden. Danach können die Vorlesungen auf die SD-Karte geschrieben werden. Als Dateiname wird unter anderem der Benutzername des HIS-LSF verwendet, damit die Datei später eindeutig identifiziert werden kann. Dadurch kann die Datei auch außerhalb der Applikation vom Benutzer manuell gelesen und verwendet werden.

Wird eine Datei angezeigt, so wird beim Einlesen jede Vorlesung als separates Listenelement angezeigt. Zusätzlich werden die Wochentage zur Unterteilung der Liste eingeblendet. Ein beispielhafter Stundenplan ist in Abbildung 4.9b zu sehen. Wie gesehen werden kann, beinhaltet die Liste nur die drei wichtigsten Informationen: Vorlesungsname, Zeit und Ort. Bei Berührung des Listenelements wird der Dialog aus Abbildung 4.9c geöffnet. Dieser bietet zwei wichtige Funktionen. Erstens kann auf der Lageplan-Activity der Vorlesungsort eingetragen werden. Dies ist sehr nützlich, da der Benutzer dann den Ort nicht manuell eintragen muss. Zweitens kann die Vorlesung in den Kalender des Android-Systems übertragen werden. Sollten in der Vorlesung Angaben fehlen, wird der Benutzer darauf aufmerksam gemacht, dass diese manuell nachgetragen werden müssen. Die Vorlesung wird nur für den angegeben Zeitraum und Turnus eingetragen. Die Spezifikationen zum Übertragen der Attribute können in [Des09] nachgelesen werden.

Neben den drei bekannten Schaltflächen im Optionsmenü, wurden zwei Schaltflächen zum Auslesen und Löschen der Weiterleitungen aus Kapitel 4.3.1 hinzugefügt. Eine wei-

tere Schaltfläche blendet ein Pop-up mit allen gespeicherten Plänen ein. Dieser erscheint nur, sofern Stundenpläne bereits gespeichert wurden. Die dritte neue Schaltfläche löscht den aktuell gewählten Vorlesungsplan vom System. Eine weitere Schaltfläche blendet die Textfelder sowie den Button am oberen Bildschirmrand ein oder aus und die letzte Fläche löscht das gespeicherte Cookie, damit die nächste Anfrage unabhängig von vorher ausgeführten Anfragen gestartet wird.

4.4 Activity: Lageplan mit Navigation

Beim Druck auf den Reiter mit der Beschriftung „Lageplan“ öffnet sich die Activity, welche den Lageplan mit Navigationsmodus verwaltet. Beim Starten der Activity wird kontrolliert, ob Paramater zum Zeichnen übergeben wurden, da die Activity nicht nur über den Reiter, sondern auch über den Vorlesungsplan gestartet werden kann. In beiden Fällen befindet sich die Karte, das Eingabefenster zum Suchen von Orten sowie ein Button zum Eintragen der Orte und zum Starten der Navigation auf der Activity. In Abbildung 4.10a ist als Beispiel Subway markiert. Die Karte ist ein Screenshot von OpenStreetMap [OSM12] und steht unter der *Creative Commons Attribution Share Alike 2.0*-Lizenz [CCB12]. Die Wahl fiel auf eine Bilddatei, da das Nachladen von Kartenmaterialien in verschiedenen Stufen zu rechen- und speicherintensiv ist. Außerdem bietet eine Bilddatei mehr Kontrolle. Die hier entwickelte Applikation soll alle Dienste offline zur Verfügung stellen können und dabei nicht auf das Angebot von fremden Applikationen zurückgreifen. Das Bild wird mit den Klassen von [Ort12] dargestellt, welche schon in Kapitel 4.2 erwähnt wurde.

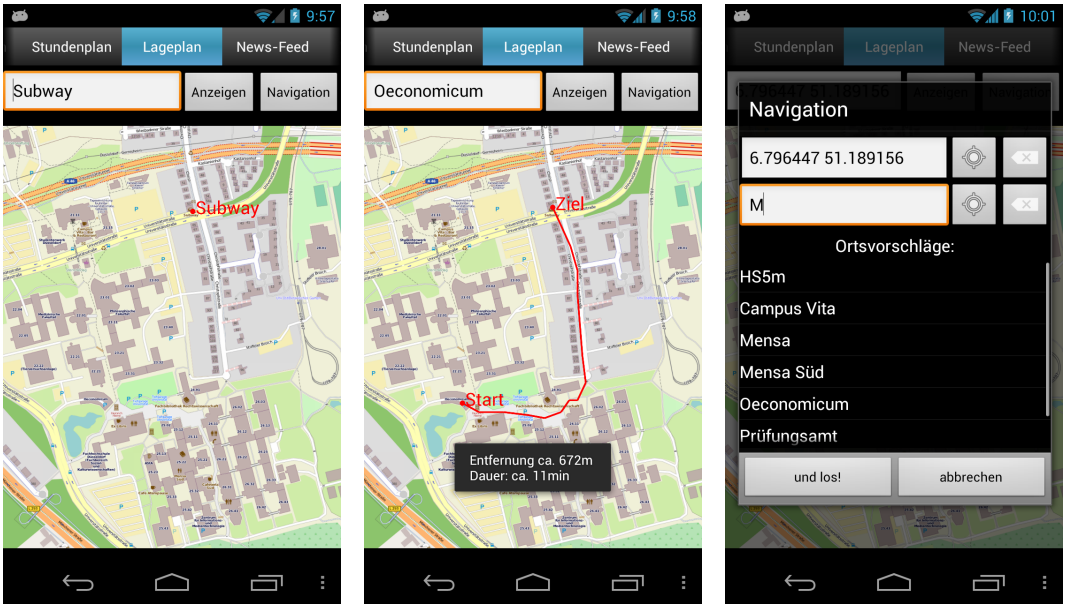
Beim Laden einer Bitmap unterscheidet Android zwischen veränderbaren (*mutable*) und nicht veränderbaren (*immutable*) Bitmaps. Eine veränderbare Bitmap wird benötigt, weil auf dem Lageplan Orte eingezeichnet werden sollen und Android die Markierungen in die Bitmap schreibt und nicht nur auf die Oberfläche. Wenn das Bild geladen und dekodiert wird, ist dieses nicht veränderbar und in der Bibliothek für Android 2.2 existiert der Wert zum Laden einer veränderbaren Bitmap noch nicht. Diese Option steht wieder erst Android 3.0 zur Verfügung, daher entstehen beim Laden des Lageplans konvertierungsbedingte Verzögerungen, die stark schwanken können. Diese entfallen, sofern der

oben genannte Wert gesetzt werden würde (diesbezüglich werden in Kapitel 4.8 Tests beschrieben). Für die Konvertierung einer nicht veränderbaren in eine veränderbaren Bitmap existieren zwei Wege. Der Erste würde eine veränderbare Kopie der Bitmap erstellen. Dies hat aber den enormen Nachteil, dass dieselbe Speichergröße ein zweites Mal alloziert werden müsste, die Applikation nicht mehr ressourcenschonend wäre und Android bei zu hohem Speicherverbrauch die Applikation beenden würde. Der zweite Weg schreibt die Bitmap in eine temporäre Datei auf der SD-Karte und erstellt die Bitmap danach manuell neu. Hier wird keine weitere Bitmap-Instanz geladen. Durch die Schreib- und Lesezugriffe entstehen jedoch Verzögerungen beim Laden der Bitmap.

Die GPS-Positionen für das Einzeichnen der Orte auf dem Lageplan sowie für die Routenberechnung im Navigationsmodus bezieht die Activity direkt beim Starten aus zwei xml-Quelldateien, welche gelesen werden. Hier greift das Prinzip der starken Modularität von Android, da die Dateien über das Kontextmenü vom Server neu bezogen werden können und somit neue Bezeichner oder Routen ohne ein Update der Applikation schnell verteilt werden können. Das Herunterladen der beiden Dateien erfolgt über die schon erwähnte gesicherte Verbindung zum Server der Applikation. Dem Benutzer ist es frei gestellt, die neu bezogenen Dateien zu löschen und mit den Standarddateien zu arbeiten. Die GPS-Positionen der Gebäude und Orte wurden manuell ausgelesen, da mehrere Messungen mit Testgeräten – welche ebenfalls in Kapitel 4.8 verwendet wurden – zu große Abweichungen zu den Daten von OpenStreetMap ergaben.

4.4.1 Lageplan

Wenn der Benutzer einen Ort auf der Karte angezeigt haben möchte, so trägt er den gewünschten Namen des Ortes in das Textfeld ein. Beim Klick auf den Button „Anzeigen“ wird der Ort auf der Karte angezeigt. Die Eingabe ignoriert Groß- und Kleinschreibung und akzeptiert mehrere Bezeichner für denselben Ort. Zusätzlich erscheint unter dem Textfeld eine Liste mit möglichen Orten. So lassen sich Hörsäle mit oder ohne das Präfix „HS“ eingeben und die Universitätsverwaltung kann über die Gebäudenummer „16.11“ oder mit dem Namen „Verwaltung“ gefunden werden. Weitere Bezeichner können leicht nachgetragen werden, da die xml-Datei zwei Arten von Abbildungen beinhaltet. Die Erste ist die Abbildung von einem Namen auf eine eindeutige ID und die



(a) Lageplan mit hervorgehobener Position

(b) Anzeigen einer Route inklusiv Informationen

(c) Navigationsmenü des Lageplans

Abbildung 4.10: Verschiedene Zustände des Lageplans mit Navigationsfunktion

Zweite ist die Abbildung der ID auf die GPS-Koordinaten. Sollte der Benutzer allerdings einen ungültigen Bezeichner verwendet haben, so wird er darauf hingewiesen und auf die Übersicht über alle Orte im Kontextmenü verwiesen. Über dieses Menü lässt sich eine Übersicht aller Orte anzeigen, aus denen der Benutzer auch seinen gesuchten Ort auswählen und anzeigen lassen kann.

Die eingezeichneten Orte passen sich der Zoomstufe an. Dies wurde ermöglicht, indem Methoden ergänzt wurden, sodass nach einer Zoom-Geste die Textgröße neu berechnet und die Orte neu eingezeichnet werden. Dafür wird die Bitmap erneuert und es werden alle vom Benutzer angezeigten Orte eingetragen. Auch hier entsteht wieder eine Verzögerung, die durch die oben genannten Faktoren der nicht veränderbaren Bitmap zustande kommt. Ein Tutorial der Entwickler schlägt vor, dass ein Viertel des Applikations-Cache aller Bitmaps der Applikation zugeteilt werden, damit die Zugriffszeiten verringert werden. Allerdings wird hier davon ausgegangen, dass der Entwickler nur mit Thumbnails arbeitet und nicht mit hochauflösendem Bildmaterial, das gezoomt werden kann. Die vom Benutzer eingetragenen Orte sind solange in einer Liste gespeichert, bis der Benutzer diese über das Kontextmenü löscht oder den Navigationsmodus betritt. Die Liste

wird beim Pausieren der Activity gesichert, sodass diese nicht verloren gehen und beim Wiederaufruf der Activity erneut eingezeichnet werden. Dies betrifft auch Routen.

Zusätzlich kann der Benutzer seine aktuelle GPS-Position einzeichnen lassen. Diese Option steht ebenfalls im Kontextmenü bereit, allerdings muss hierfür das GPS-Modul angeschaltet werden. Ist dieses ausgeschaltet, so wird der Benutzer über ein Pop-up gefragt, ob er in die Einstellungen geführt werden möchte, da die Applikation das GPS-Modul aus Sicherheitsgründen nicht selber einschalten kann. Ist das Modul eingeschaltet, lauscht eine Hilfsklasse nach der aktuellen Position. Damit die Akkuleistung nicht beeinträchtigt wird, lauscht die Instanz der Hilfsklasse nur solange, wie eine beliebige Activity der Applikation im Vordergrund ist. So kann der Benutzer während der Ortung noch weitere Dienste der Applikation in Anspruch nehmen und sobald die Activity in den Speicher gelegt wird, beziehungsweise der Benutzer zum Homescreen zurückkehrt, wird die Instanz beendet. Veränderungen der GPS-Position werden nur registriert, sofern zwischen dem aktuellen Signal und dem letzten Signal ein Meter oder 500 Millisekunden liegen. Der Benutzer erhält zusätzlich eine Benachrichtigung, sofern erstmalig eine Position gefunden werden konnte. Bei jeder weiteren Veränderung der Position bleibt die Benachrichtigung aus. Befindet sich die Position auch auf dem Gelände der Universität, so ist diese gültig und der Benutzer kann die Position einzeichnen und teilen. Beim Teilen öffnet sich – wie in Kapitel 4.1.2 – eine Übersicht über alle auf dem Android-System installierten, textbasierten Applikationen. Somit kann sich der Benutzer nicht nur auf dem Campus orientieren, sondern auch Freunden mitteilen, wo er sich befindet. Damit diese die Position auf dem Campus finden, akzeptiert die Texteingabe am oberen Bildschirmrand GPS-Koordinaten.

4.4.2 Navigation

Die Navigation kann direkt über den Button auf der Activity gestartet werden. Sofern Orte auf der Karte eingezeichnet sind, wird der Benutzer gefragt, ob diese gelöscht werden sollen. Im Navigationsmodus wird maximal eine Route angezeigt, damit die Übersicht nicht zu viele Routen anzeigt und damit die Übersichtlichkeit verloren gehen würde. Bestätigt der Benutzer das Löschen der aktuellen Orte, so wird ein Pop-up mit zwei Textfeldern und jeweils einem Button an der Seite eingeblendet. Dabei wird der einge-

gebene Ort aus der vorherigen Ansicht als Start übernommen. Sofern der Benutzer Start- oder Zieladresse angibt, werden Ortsvorschläge gemacht, aus denen ausgewählt werden kann (wie in Abbildung 4.10b zu sehen). Es wurde darauf geachtet, dass Orte mit mehreren Bezeichnern nur mit der passenden Bezeichnung vorhanden sind. Zum Beispiel kann die Bibliothek über die Bezeichner „Bib“, „Bibliothek“, „Universitätsbibliothek„, „Unibib “ und „ULBD “ gefunden werden, jedoch wird bei der Eingabe eines „u“ zuerst nur Universitätsbibliothek angezeigt, wobei „ul“ das Wort ULBD als Vorschlag angibt. Zusätzlich kann der Benutzer einen Button mit dem GPS-Ortungssymbol drücken und seine aktuelle GPS-Position in das Textfeld einfügen lassen, sofern das GPS-Modul angeschaltet und die Position gültig ist. Werden GPS-Positionen per Hand eingetragen, so müssen diese mit einem Komma voneinander getrennt werden.

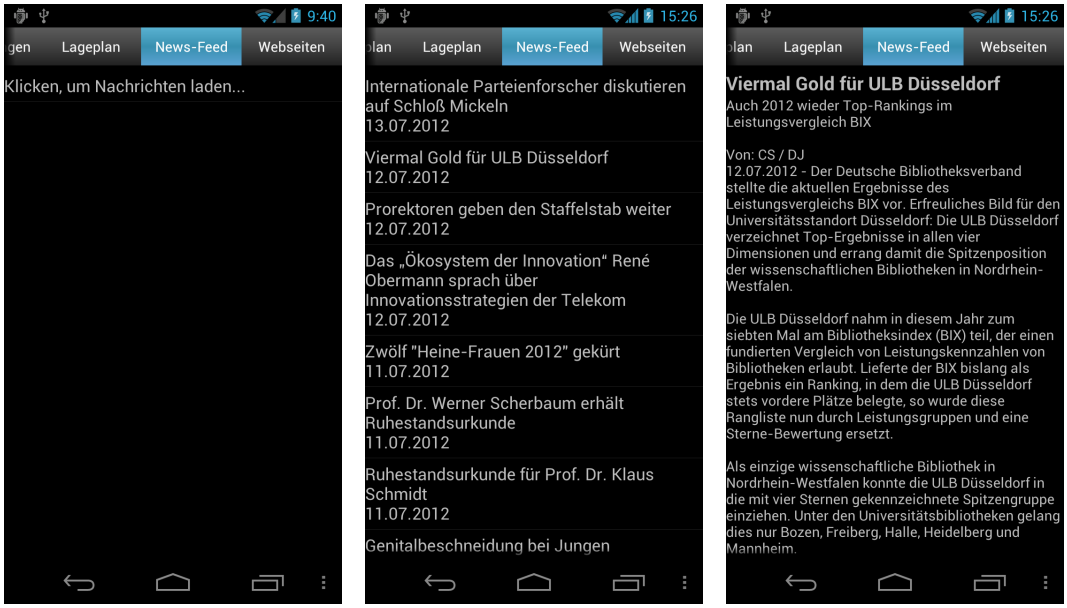
Wenn der Benutzer die Orte bestätigt, wird die Route im Hintergrund berechnet. Als erstes werden dazu die Knotenpunkte des gespeicherten, gewichteten Graphen bestimmt, welche am nähsten am Start- und Zielpunkt liegen. Der Graph ist in der zweiten xml-Datei gespeichert und wird ebenfalls beim Start der Activity eingelesen. Als Gewicht der Kanten wurde der geometrische Abstand der GPS-Koordinaten gewählt. Die Knoten wurden ebenfalls aus den oben genannten Gründen per Hand eingelesen. Die nahe gelegenen Knoten lassen sich ebenfalls über den geometrischen Abstand bestimmen. Die kürzeste Route zwischen den beiden Knoten wird mithilfe des Dijkstra-Algorithmus berechnet. Dazu wurden Klassen von [Vog12] eingebunden. Als Rückgabewert der Methode wird der kürzeste Pfad als eine Liste mit Knoten angegeben. Diese werden ebenfalls gespeichert und eingezeichnet. Zusätzlich werden noch Verbindungslinien mit eingezeichnet, welche beim Zoomen ebenfalls skaliert werden. Dadurch, dass die Knotenpunkte gespeichert werden, ist auch die Route nach einem Wechsel der Activities weiterhin sichtbar.

Zusätzlich zur Anzeige der Route wird die Entfernung vom Start bis zum Ziel sowie die benötigte Zeit eingeblendet. Die Schrittgeschwindigkeit kann in den Einstellungen des Lageplans vorgenommen werden, mehr dazu in Kapitel 4.7. Die Entfernung vom Start bis zum Ziel ist gleich der Entfernung aller Teilstrecken von einem Knoten des Pfades bis zum Nächsten. Zur Umrechnung von GPS-Koordinaten in Meter wurde die Orthodrome basierend auf dem WGS84-Elipsoid[8] berechnet [Ven12].

[8] World Geodetic System 1984

4.5 Activity: News-Feed

Die Group-Activity des „News“-Reiters ist eine Schnittstelle zum offiziellen RSS-Feed der Universität Düsseldorf [Hei12b]. Zur Darstellung existieren zwei Activities. Die erste Activity übernimmt die Darstellung und Verwaltung der Übersicht aller Nachrichten und die zweite Activity zeigt die vom Benutzer ausgewählte Nachricht an. Die Mechanismen zur Verwaltung von mehreren Activities wurde in Kapitel 4.1 dargestellt.



(a) Activity beim ersten Aufruf

(b) gefüllte Liste mit Titeln und Publikations-Datum

(c) Die zum Titel dazugehörige Nachricht

Abbildung 4.11: Verschiedene Zustände der News-Feed-Activity

4.5.1 RSS-Feed der Universität Düsseldorf

Sofern keine Nachrichten auf dem Endgerät gespeichert sind, findet der Nutzer eine leere Ansicht wie in Abbildung 4.11a vor. Durch Klicken auf den Schriftzug wird die Quelldatei des RSS-Feeds der Universität mithilfe eines HTTP-Post-Requests angefordert. Die HTTP-Response-Nachricht enthält die xml-Datei des News-Feeds, welche zum Parsen

auf der SD-Karte gespeichert wird. Nach der Speicherung werden pro Nachricht der Titel und das Datum der Nachricht sowie der Link zur Nachricht gefiltert und in Variablen gehalten. Hat der Benutzer in den Einstellungen den Wert gesetzt, dass jede Nachricht vorabgeladen werden soll, so werden direkt die Inhalte der Nachrichten angefordert und gespeichert. Als Dateiname wird der Nachrichtentitel gewählt, welcher von Leerzeichen bereinigt wurde. Somit lassen sich die Dateien auch später noch eindeutig identifizieren. Durch die Sicherung der Daten in Variablen können die Titel und Zeitstempel der Nachrichten in einer Liste angezeigt werden. Lädt der Benutzer erneut die Übersicht beziehungsweise möchte diese aktualisieren, so werden die aktuellen Nachrichten nicht verworfen, sondern es findet ein Abgleich der Informationen statt. Alle Nachrichten, die nicht mehr aktuell sind, werden verworfen und neue Nachrichten werden mit aufgenommen.

4.5.2 Darstellung der Nachricht aus dem RSS-Feed

Die Listenansicht reagiert auf einen kurzen und einen langen Klick. Bei einem langen Klick wird die „teilen“-Optionen für die Nachricht in einem Pop-up zur Auswahl angeboten. Bei Auswahl der Option werden dem Benutzer wieder alle Applikationen angeboten, welche Textnachrichten teilen können. Wird ein Titel aus der Liste in Abbildung 4.11b durch einen kurzen Klick ausgewählt, so ist die Nachricht entweder auf der SD-Karte gespeichert oder wird mit den bekannten Mechanismen geladen. Der Link zur Nachricht befindet sich in einer Variablen. Zu beachten ist, dass die Quelldatei, die vom Server als Antwort geschickt wird, ein HTML-Dokument ist, das erst analyisiert werden muss. Die Nachricht befindet sich auf der Hauptseite der Universität und daher darf nur der Text der Nachricht ausgelesen werden. Bilder sowie die dazugehörigen Untertitel der Nachricht werden gefiltert. Sofern die Nachricht vorhanden ist, werden die Nachricht und der Titel als Anhang eines Intents an die zweite Activity gesendet, welche die Leseansicht des News-Feeds ist.

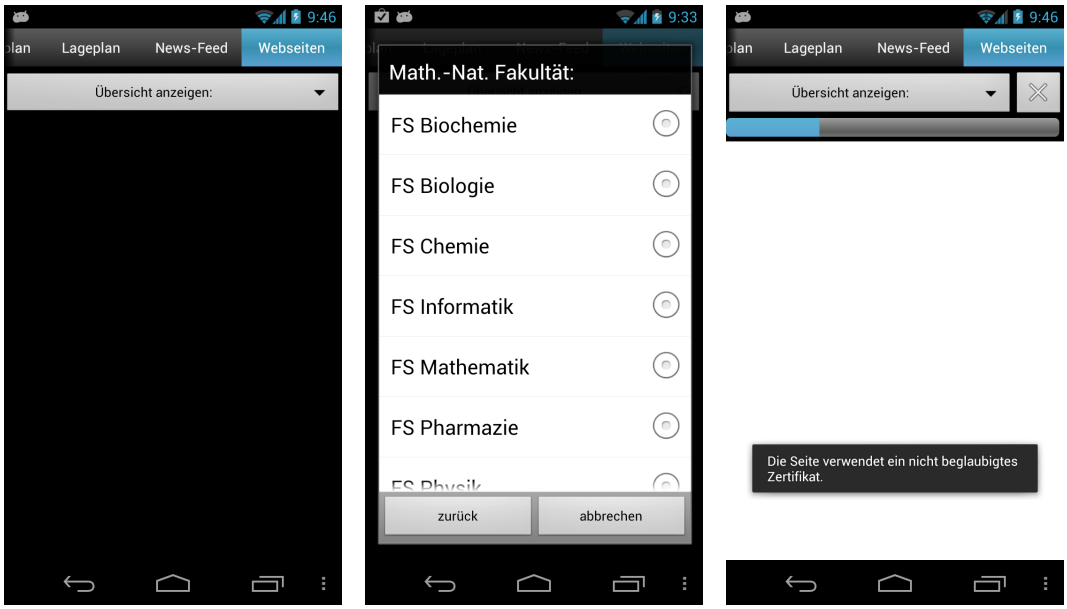
Das Leseangebot befindet sich –wie in Abbildungen 4.11c zu sehen – in einem Textfeld, welches in einer scrollbaren Ansicht eingebettet ist. Bei Berührung mit dem Displays erscheint für 1,5 Sekunden das Zoom-Werkzeug, mit dem sich die Größe des Textes an-

passen lässt. Alternativ kann auch eine vordefinierte Größe in den Einstellungen vorgenommen werden.

Das Optionsmenü der beiden Activities beinhaltet die drei Standard-Schaltfächen zum Anzeigen der Einstellungen beziehungsweise das Impressum sowie die Schaltfäche zum Beenden der Applikation.

4.6 Activity: Schnellzugriff auf wichtige Webseiten

Damit der Benutzer in dem System-Browser keine Lesezeichen für Seiten der Universität verwalten muss, wurde eine Schnellauswahl der Universitäts-Webseiten in die App eingebunden. Beim Start der Activity kann die Übersicht aus Abbildung 4.12a gesehen werden. Das vorgestellte Spinner-Widget aus Kapitel 4.2 konnte nicht verwendet werden, da ein zweistufiges Auswahlmenü benötigt wird.Beim Start der Activity wird eine



(a) Activity beim ersten Aufruf

(b) Auswahl der vorgegebenen Seiten

(c) Toast beim Laden einer Seite mit unsicherem Zertifikat

Abbildung 4.12: Verschiedene Zustände der Webseiten-Activity

Webview[9] und ein Optionsmenü initialisiert. Das Optionsmenü bietet neben den drei bekannten Schaltflächen drei neue Schaltflächen an. Eine Schaltfläche löscht den Cache und Verlauf der Webview und die anderen beiden laden das Webseiten-Verzeichnis vom Server und löschen dieses vom Endgerät. Alle Webseiten sind (wie die Koordinaten und der Graph des Lageplans aus Kapitel 4.4) in einer xml-Datei gespeichert. Daher können neue Webseiten oder Updates der Applikation eingepflegt werden und es greift wieder das von Android gewünschte Prinzip der starken Modularität. Es wird beim Start der Activity zuerst versucht, eine neue, lokal gespeicherte xml-Datei einzulesen und erst danach wird auf die xml-Datei aus den Ressourcen der Applikation zugegriffen. Zusätzlich existieren in der xml-Datei Einträge von Seitennamen, die JavaScript benötigen. JavaScript wird in der Webview nur für die authorisierten Seiten aus der xml-Datei erlaubt und ist standardmäßig deaktiviert, da es nicht ressourcenschonend ist.

Beim Lesen der xml-Datei, wird ein zweistufiges Menü erstellt und beim Klick auf den Button in Abbildung 4.12a in einem Pop-up angezeigt. Die erste Stufe enthält den Eintrag „Allgemeine Seiten“ sowie einen Eintrag für jede Fakultät. Das danach folgende Menü enthält alle kontextspezifischen Seiten, passend zum ausgewählten ersten Element. Bei erneuter Auswahl wird eine Seite geladen. Das Auswahlmenü der mathematisch-naturwissenschaftlichen Fakultät kann in Abbildung 4.12b gesehen werden.

Beim Initialisieren der Webview musste darauf geachtet werden, dass die wichtigsten Einstellungen genutzt werden. Dazu gehören die Unterstützung zum Zoomen und das Anzeigen vom Scroll-Balken an der Seite. Zusätzlich werden die in den Einstellungen gesetzten Werte geladen, mehr zu den Einstellungen in Kapitel 4.7. Außerdem wurde das Verhalten der Webview überschrieben, sodass ein Fortschrittsbalken den Ladestatus der Webseite anzeigt. Ein Button nebem dem Balken kann bei Bedarf das Laden abbrechen oder ein Neuladen veranlassen. Zusätzlich wurden SSL-Fehlermeldungen abgefangen, da zum Beispiel die Fachschaft Informatik ein Zertifikat benutzt, das in Android nicht beglaubigt ist. Dem Benutzer wird eine Meldung über das nicht sichere Zertifikat eingeblendet und das Laden wird fortgesetzt. Eine Bestätigung durch den Benutzer ist nicht notwendig, da alle Seiten zu der Universität gehören und als sicher gelten.

In dieser Activity hat der „zurück“-Button des Android-Systems zwei Bedeutungen. Als

[9] Eine Webview ist ein integriertes Layout, welches Webseiten anzeigen kann.

erstes dient der Button zur Navigation im Browser und lädt die vorherige Seite. Erst wenn keine Seite mehr geladen werden kann, öffnet sich ein Pop-up mit der Frage, ob der Benutzer die Applikation wirklich beenden möchte.

4.7 Activity: Einstellungsmenü

Das Einstellungsmenü wird durch eine *PreferenceActivity* dargestellt. Sie besitzt alle grundlegenden Eigenschaften, die auch die Activity aus Kapitel 2.2 besitzt. Im Gegensatz zur normalen Activity, besteht diese immer aus einer Listenansicht und Android übernimmt die Speicherung aller gesetzten Werte, sofern die Activity pausiert. Allerdings wird weder in Literatur wie [Fel11] oder dem Manual zur Activity [Goo12j] erwähnt, wo die Einstellungen gespeichert werden. Deswegen wird an dieser Stelle davon ausgegangen, dass sie ebenfalls für den Benutzer nicht zugänglich in den system-eigenen Dateien liegen. Diese Vermutung konnte anhand eines gerooteten Endgeräts bestätigt werden, allerdings ist der Pfad abhängig von der verwendeten Android-Version.

Das Einstellungsmenü der Applikation bietet für jeden Reiter optionale Möglichkeiten inklusive eines allgemeinen Reiters. In diesem kann der Benutzer neben der in der Applikation verwendeten Schriftgröße alle Dateien löschen sowie alle Einstellungen auf die vordefinierten Werte setzen. Möchte der Benutzer alle Dateien löschen, werden ihm in einem Pop-up die Anzahl an Ordnern, Dateien sowie Tabellen angezeigt und der belegte Speicher angezeigt. Bei Bedarf können alle Dateipfade angezeigt werden oder die Dateien direkt gelöscht werden. Außerdem kann ein Fehlerbericht an die Entwickler der App geschickt werden. Bei jedem schwerwiegenden Fehler wird ein Eintrag in eine lokale Log-Datei auf der SD-Karte geschrieben und diese kann per E-Mail über die Schaltfläche verschickt werden. Android selbst bietet ebenfalls bei Absturz einer Applikation die Möglichkeit, einen Fehlerbericht zu senden. Zusätzlich können die Wert für die Wettervorhersage sowie E-Mail-Anzeige gesetzt werden, da diese standardmäßig nicht gesetzt sind.

Die Einstellungskategorie des Forums bietet zwei Felder zum Markieren sowie zwei Textfelder. Das erste Markierungsfeld beinhaltet den Wert, ob bei einer Aktualisierung

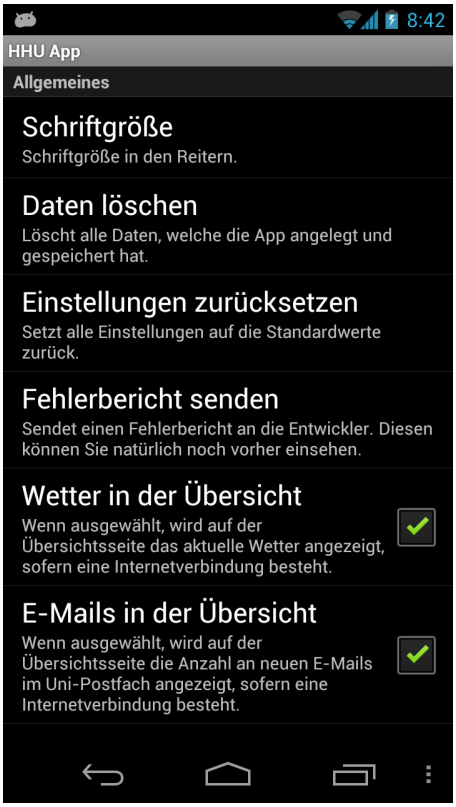


Abbildung 4.13: Ein Ausschnitt aus den Einstellungsmöglichkeiten

eines Threads nur die Nachrichten des Threads oder die Nachrichten des gesamten Forums aktualisiert werden sollen. Daher wird dem Content des HTTP-Post-Request nur die höchste Post-ID und nicht die aktuelle Topic-ID angehangen. Das PHP-Skript versucht bei beiden Möglichkeiten eine Topic-ID auszulesen, jedoch ist sie in einem Fall null und daher kann die korrekte MySQL-Anfrage an die Datenbank des Server gestellt werden. Das zweite Feld repräsentiert den Wert, ob bei Aktualisierung eines Threads die persönlichen Nachrichten mit geladen werden sollen. Ist der Wert gesetzt, so werden beim Aktualisieren des Topics oder Posten eine Nachricht zum Schluss weitere Requests gesendet. Sofern die Antwort Nachrichten enthält, wird dem Benutzer über ein Pop-up die Anzahl der Nachrichten angezeigt und die Möglichkeit geboten, diese direkt zu lesen. Die Textfelder stehen für den Benutzernamen sowie für das Passwort des Forums. Zu beachten ist, dass das Passwort nirgends als Klartext erscheint, sondern immer mittels Sternchen dargestellt wird.

Das Einstellungsmenü für die Mensa bietet lediglich eine Vorauswahl für die Mensa. Das

heißt, dass der Benutzer angeben kann, ob beim Öffnen des Reiters direkt ein Speiseplan angezeigt werden soll oder ob dies erst im Reiter selbst ausgewählt werden muss.

Die Kategorie für den Vorlesungsplan beherbergt ebenfalls zwei Textfelder, welche die gleiche Aufgabe wie die des Forums übernehmen. Allerdings werden hier die Daten für das HIS-LSF eingetragen.

Für den Lageplan kann eine Schrittgeschwindigkeit angeben werden. Diese wird bei der Navigation benötigt, da dort abgeschätzt wird, wie lange der Benutzer für die angegebene Route benötigen wird.

Die Einstellungen des News-Feeds beinhalten nur ein Markierungsfeld. Bei Markierung des Feldes wird bei Aktualisierung des Index jede Nachricht vorab gerufen. Dies verbraucht jedoch mehr Traffic und Zeit, da jede Nachricht geladen, analysiert und gespeichert werden muss.

Die letzte Einstellung betrifft den Browser. In dem Abschnitt kann entschieden werden, ob eine Webseite auf die Breite des Smartphones angepasst werden soll. Das heißt, dass Android versucht, Elemente umzuordnen, um die Seite auf die Breite des Endgeräts anzupassen. Dies ist jedoch nicht mit jeder Seite möglich. Außerdem lässt sich der Wert für das automatische Laden von Bildern auf Webseiten setzen, der standardmäßig immer wahr ist.

4.8 Laufzeiten auf verschiedenen Endgeräten

Da die Activity des Lageplans mit den unter Android 2.2 zur Verfügung stehenden Bibliotheken Verzögerungen beim Laden aufweist, werden im Folgenden die Ladezeiten auf verschiedenen Endgeräten verglichen. Zur Orientierung der Leistung wurden Benchmarks mit Hilfe der Applikation *Quadrant Advanced Edition* [Goo12k] durchgeführt. Zu beachten ist, dass die in Tabelle 4.2 angegebenen Werte nicht direkt verglichen werden dürfen, sondern nur als Orientierung dienen. Es wurden Benchmark-Tests aus den Bereichen für die CPU, den Speicher, Lese- und Schreibzugriffe auf die SD-Karte sowie das Rendern von 2D-Graphiken durchgeführt. Alle Tests wurden mehrmals unter gleichen

Bedingungen gestartet. Die ersten vier Smarthpones stammen aus der Galaxy-Reihe des Herstellers Samsung, das letzte Testgerät ist ein Tablet. Außerdem ist noch zu beachten, dass das Samsung Galaxy S I9000 keine originale Firmware besaß, sondern eine modifizierte Rom vom Team RemICS in der Version 1.1 [Tea12].

Gerät	Kernel	OS	CPU	Taktung	Ram	Benchm.
Nexus I9250	3.0.31	4.1	2	1200 mHz	1024 MB	1600
S I9000	3.0.8	4.0	1	1000 mHz	512 MB	1660
S3 I9300	3.0.15	4.0	4	1400 mHz	1024 MB	7060
Ace S5840	2.6.35.7	2.3	1	800 mHz	278 MB	1090
Medion Life P4310	2.6.35.7	2.3	1	800 mHz	512 MB	1030
Archos 101 Tablet	2.6.29	2.2	1	1000 mHz	256 MB	1260

Tabelle 4.2: Hardware und Benchmarks verschiedener Testgeräte

Aus Tabelle 4.2 kann entnommen werden, dass zwei Smartphones aus dem Einsteiger-, ein Smartphone aus dem Highend-Bereich, zwei Geräte aus der Mittelklasse, wobei eines stark modifiziert war, sowie ein Tablet vorlagen. Lauffähig war die Applikation auf allen Geräten, da nur Bibliotheken aus Android 2.2 verwendet wurden. Der Test bestand aus einer Alltags-Simulation im Gebrauch der App, das heißt, dass jeder Reiter geöffnet und alle rechenintensiven Dienste in Anspruch genommen wurden. Es hat sich gezeigt, dass alle Testgeräte eine Verzögerung beim Öffnen der Lageplan-Activity aufweisen. Lediglich das Markieren von Orten steht in Abhängigkeit zu den Benchmark-Ergebnissen, denn beim Samsung Galaxy S3 war kaum eine Verzögerung merkbar. Allerdings konnte der Lageplan auf dem Samsung Galaxy S und dem Archos 101 Tablet am weichsten gezoomt werden. Zusätzlich wurde eine Heapanalyse der Endgeräte mithilfe des *DDMS*-Tools[10] durchgeführt. Nach dem Öffnen des Lageplans zeigte sich, dass der Applikations-Heap den maximalen Speicherbereich allozierte, welcher vom System zugewiesen wurde. Der tatsächlich allozierte Speicher schwankte jedoch stark. Außerdem wurde eine instabile Version der Applikation getestet, welche die erwähnte *inMutable*-Option aus Kapitel 4.4 ausnutze und somit für Android 4 optimiert war. Es zeigte sich, dass die Verzögerung minimiert werden konnte und Interaktionen weicher erschienen.

[10] Kur für Dalvik Debug Monitor Server.

Kapitel 5

Benutzer-Erfahrungen

Der größte Anreiz zur Durchführung einer kontrollierten Fallstudie war es, Intuitivität und Bedienung der Applikation überprüfen zu können. Zusätzlich konnten Meinungen und Kritiken von außenstehenden Personen eingeholt werden. Im Folgenden werden der Aufbau sowie die Ergebnisse der Fallstudie beschrieben.

5.1 Aufbau der Fallstudie

Die Fallstudie ist in zwölf Abschnitte gegliedert, die in Abbildung A.2 und A.3 eingesehen werden können. Der erste Abschnitt beinhaltet grobe persönliche Angaben, damit das Ergebnis besser gewertet werden kann. Zum Beispiel ist es interessant zu erfahren, wie schnell sich ein Proband mit keinerlei Erfahrung in Android mit der Applikation vertraut machen kann, als ein Proband, welcher sein Android-Smartphone täglich nutzt. Der zweite und dritte Abschnitt gibt den Probanden maximal fünf Minuten Zeit, um sich mit der App und dem Gerät vertraut zu machen sowie seinen ersten Eindruck zu schildern. Die nächsten sieben Abschnitte sind Aufgaben zu den Funktionen der App, eine für jeden Reiter. Der siebte Abschnitt beinhaltet eine Aufgabe zum Einstellungsmenü. Die letzten beiden Abschnitte bieten die Möglichkeit die Nutzbarkeit und Intuitivität der Funktionen zu bewerten sowie Verbesserungsvorschläge und Kritik zu unterbreiten. Das parallel geführte Protokoll kann in Abbildung A.4 und A.5 ebenfalls im Anhang eingese-

hen werden. Das Protokoll sollte den Probanden im Verhalten beim Erledigen der sieben Aufgaben beurteilen sowie die benötigte Zeit ermitteln, da sich mithilfe dieser Angaben Rückschlüsse auf die Intuitivität der Applikation ziehen lassen konnten

5.2 Auswertung der Fallstudie

Das Ergebnis der Fallstudie ist eindeutig, allerdings nicht repräsentativ, da die kontrollierte Studie mit 19 Probanden geführt wurde, wovon 17 Informatik im Hauptfach studieren. Daher waren die Testpersonen technisch versiert und entsprachen nicht dem Durchschnitt aller Anwender der Applikation. Weitere Auswertungen der persönlichen Daten können Kapitel A.1 entnommen werden. Als Regel zur Deutung der Abbildungen 5.1 und 5.2 gilt, dass jede Auswertung eine vertikale Linie besitzt. Die Balken links der Linie repräsentieren positive Eigenschaften, Balken auf der rechten Seite, negative.

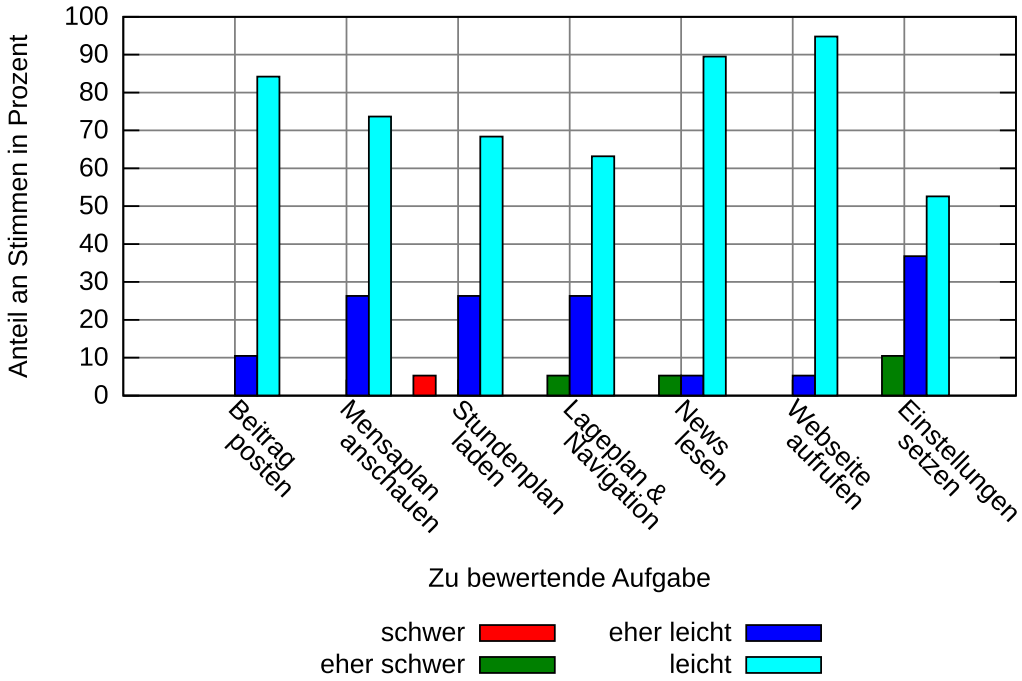


Abbildung 5.1: Bewertung der Schwierigkeit beim Erledigen alltäglicher Aufgaben

Wie in Abbildung 5.1 zu sehen ist, war das Setzen von Werten im Einstellungsmenü für die Probanden die schwierigste Aufgabe. Diese Schwierigkeit wurde durch das Testge-

rät verursacht, da der Button für das Optionensmenü leicht übersehen werden konnte. Der Grafik kann ebenfalls entnommen werden, dass ein Proband das Herunterladen des Stundenplans als „schwer“ empfand. Dies lag an unklaren Bezeichnungen der Widgets im Stundenplan-Reiter sowie an einer fehlenden Meldung, sofern keine Vorlesungen im Stundenplan eingetragen wurden. Beide Eigenschaften wurden verbessert. Insgesamt ist aber zu sehen, dass fast alle Probanden die Aufgaben schnell lösen konnten und dies als „leicht“ oder „eher leicht“ eingestuft haben. Auch wenn es Probanden gab, die Aufgaben teilweise als „eher schwer“ eingestuft haben, so brauchte kein Proband außergewöhnlich lange zum Lösen einer Aufgabe. Daraus folgt, dass die Informationen auch im Alltag schnell bereit stehen und abgerufen werden können. Allerdings konnte bei der Bearbeitung der Aufgaben festgestellt werden, dass die Probanden im zweiten Abschnitt (Einarbeitungsphase) neugieriger waren, als während der Bearbeitungszeit. Dort war es Ihnen wichtiger, die Aufgaben zu erledigen, als den kompletten Funktionsumfang der Applikation kennen zu lernen. Daher wäre es interessant, einen Feldstudie zu starten um weitere Meinungen zu erhalten.

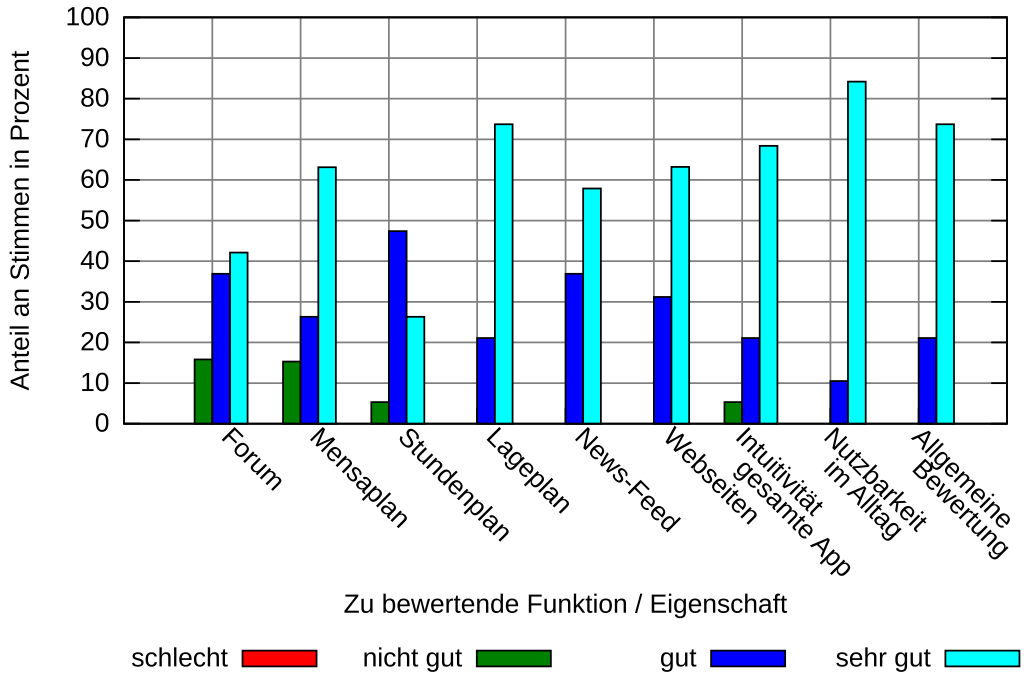


Abbildung 5.2: Bewertung der einzelnen Funktionen der Applikation durch Probanden

In der Bewertung der Funktionen (Abbildung 5.2) wird deutlich, dass das Forum und die Anbindung an den Stundenplan nicht zufriedenstellend waren. Die Probanden äußerten

dazu, dass sie bei dem Begriff Forum eine komplexere Struktur mit mehr Menüs – wie im genannten phpBB-Board – erwartet hätten. Der Stundenplan war nicht zufriedenstellend, da der Reiter während der Tests noch den Bezeichner „LSF“ trug und daher das gesamte Vorlesungsverzeichnis zur Einsichtnahme erwartet wurde. Eine weitere hilfreiche Anmerkungen war, dass das Herunterladen aller Mensapläne der kommenden Wochen wünschenswert sei. Anmerkungen, welche unklare oder zu knappe Bezeichnungen von Widgets bemängelten, wurden nach der Studie Folge geleistet.

Wie erwähnt, waren die Probanden während der Einarbeitungsphase neugieriger als in den Aufgaben. Allerdings lässt sich sehen, dass die Probanden bei den ersten vier Aufgaben, welche die ersten vier Reiter betreffen, zielstrebiger waren, als bei den letzten Aufgaben. Das lag daran, dass der News-Feed, die gespeicherte Webseiten-Liste und das Einstellungsmenü nicht den gleichen Mehrwert bieten, wie die anderen Reiter. Die Orientierungslosigkeit bei der Stundenplan- und Einstellungs-Aufgabe kan von unklaren Bezeichnungen der Widgets sowie dem schon erwähnten, versteckten Button.

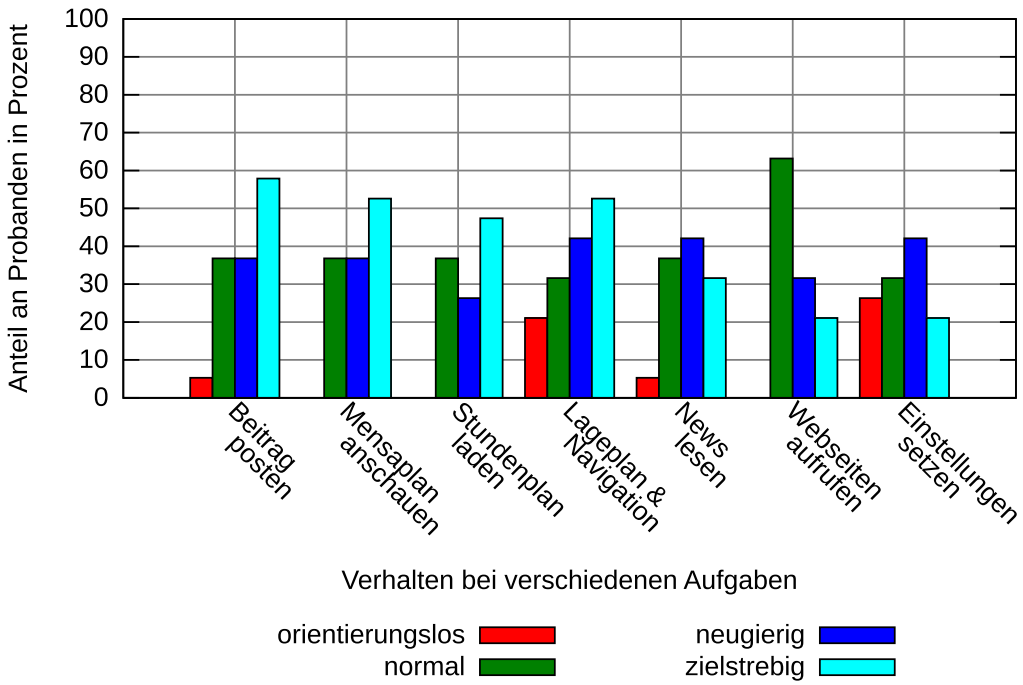


Abbildung 5.3: Bewertung des Verhalten der Probanden während der Fallstudie

Kapitel 6

Fazit und Ausblick

6.1 Fazit

Aktuell sind mobile Applikationen, die den Benutzern den Zugang zu Informationen erheblich erleichtern oder mehrere Informationsquellen in einer Anwendung vereinen, sehr gefragt. In dieser Arbeit wurde eine App am Beispiel der Heinrich-Heine-Universität Düsseldorf entwickelt, welche die wichtigsten, alltäglichen Informationen für Studenten bereithält. Im Gegensatz zu den Applikationen aus Kapitel 1.2 wurde diese Applikation ohne von der Universität zur Verfügung gestellte Schnittstellen und Mittel realisiert. Lediglich wurde für die technische Umsetzung sowie zur Speicherung von Daten ein Server vom Lehrstuhl Rechnernetze zur Verfügung gestellt. Außerdem wurde darauf geachtet, dass die Applikation den Richtlinien der Android-Programmierung entspricht und alle Inhalte Offline zur Verfügung stehen. Zusätzlich beinhaltet die Applikation nützliche Dienste und zusätzliche Funktionen wie zum Beispiel, dass Vorlesungen in den Kalender eingetragen werden können oder eine offline-Navigation zur Verfügung steht. Daher ist die in der Arbeit entwickelte Applikation eine Hilfe des Alltags an der Heinrich-Heine-Universität und sollte an die Studenten weitergegeben werden.

Die entwickelte Applikation bietet alle wichtigen Informationen der Universität, die ein Student am Tag oder in der Woche benötigt. Sie befindet sich schon in einem sehr stabilen Zustand. Daher lässt sich die App im Alltag der Universität nutzen und es könnte

eine Feldstudie gestartet werden, in der Studenten die Applikation im Alltag testen können, damit eventuell letzte unerwartete, auftretende Komplikationen behoben werden können.

6.2 Ausblick

Damit die Applikation den Studenten zur Verfügung gestellt werden kann, sollte das Design ansprechender und das Forum aus Kapitel 4.1 umfassender werden. Ideal wäre es, zunächst ein Forum mit Webinterface aufzubauen, wie das in Kapitel 4.1.4 erwähnte phpBB-Board und dann Schnittstellen für die Applikation anzubieten.

Für weitere Smartphones lässt sich ebenfalls die Applikation umsetzen, da lediglich die Widgets und Activities angepasst werden müssen. Die wichtigen Kernstrukturen und Ideen der Methoden zum Bezug der Informationen bleiben erhalten. Außerdem arbeitet der Server unabhängig von dem Betriebssystem Android. Zusätzlich sollte die Applikation für Android 4 überarbeitet werden, da in Android 4 verbesserte Bibliotheken und vereinfachte Strukturen angeboten werden. Insbesondere die Bibliotheken für Bitmaps und die Verwaltung von Group-Activities wurden verbessert.

Es ist ebenfalls denkbar, weitere Dienste mit aufzunehmen. Nützlich wäre eine Ausbau des E-Mail-Verzeichnises der Universität oder eine Maske für die Universitäts- und Landesbibliothek Düsseldorf.

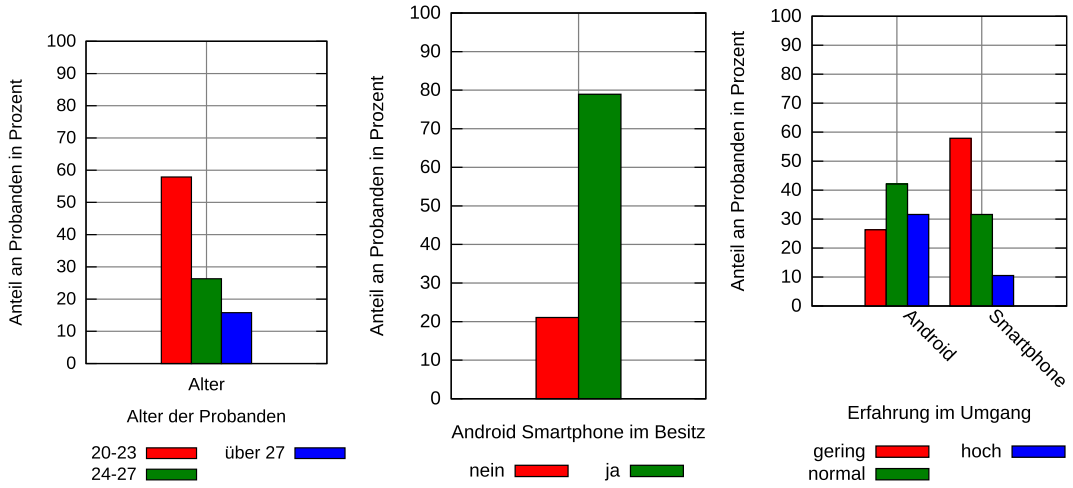
Da die veröffentlichte Fallstudie nur in einem begrenzten Rahmen mit Probanden desselben Studienfachs durchgeführt werden konnte, ist eine Wiederholung der Fallstudie in einem größeren Rahmen durchaus empfehlenswert. Die Applikation ist für Studenten gedacht, damit deren Alltag einfacher und unkomplizierter werden kann. Daher sollten diese auch Einfluss auf die Entwicklung der Applikation haben.

Anhang A

Anhang

A.1 Persönliche Angaben der Probanden

Die Probanden konnten in der Fallstudie persönliche Angaben zu ihrem Studium sowie zu den Erfahrungen im Umgang mit Smartphones und Android machen. Die Werte in Abbildung A.1 basieren auf den Daten von 19 Probanden.



(a) Alter der Probanden

(b) Anteil an Probanden, die ein Android-Smarthphone besitzen

(c) Erfahrung der Probanden mit Smartphones und Android

Abbildung A.1: Auswertung der persönlichen Angaben der Probanden

A.2 Fallstudie

Die Fallstudie zur Bachelorarbeit sollte Schwächen in der Bedienung der Applikation offenlegen sowie letzte Fehler finden. Die Probanden wurden mithilfe einer geschlossenen Diskussionsrunde der HHU-Informatiker auf der Plattform Facebook, mit E-Mails und mit Flyern geworben. Die Ergebnisse der Fallstudie können in Kapitel 5 eingesehen werden und in Abbildung A.2 und A.3 ist das Formblatt zu sehen, welches ebenfalls in Kapitel 5 erklärt wurde.

A.3 Protokoll der Fallstudie

Damit die Fallstudie besser ausgewertet und das Verhalten der Probanden besser bewertet werden konnte, wurde parallel zur Fallstudie ein Protokoll für den Studienleiter erstellt. Dieses ist in Abbildung A.4 und A.5 abgebildet sowie in Kapitel 5 erklärt.

Fallstudie zur Bachelorarbeit

Eine Android-App für alltägliche Studieninformationen am
Beispiel der Heinrich-Heine-Universität Düsseldorf

von Tobias Krauthoff (tobias.krauthoff@.uni-duesseldorf.de)

Proband Nr.: _____

Angaben zur Person:

Alter:	☐ unter 20	☐ 20–23	☐ 24–27	☐ über 27
Ich studiere Informatik:	☐ ja	☐ nein		
Ich besitze ein Smartphone mit Android:	☐ ja	☐ nein		
Wie viel Erfahrungen habe ich im Allgemeinen mit Smartphones:	☐ viel	☐ normal	☐ gering	
Wie viel Erfahrungen habe ich mit Android–Smartphones:	☐ viel	☐ normal	☐ gering	

Aufgabe 1
Sie haben fünf Minuten Zeit, um sich mit der Applikation vertraut zu machen.

Aufgabe 2
Beschreibung Sie bitte kurz Ihren ersten Eindruck:

Aufgabe 3
Schreiben Sie einen Eintrag in das Forum der mathem. naturwiss. Fakultät. Benutzen Sie dafür den Benutzernamen: `dummy` und das Passwort: `0000`.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 4
Schauen Sie nach, was es am Dienstag in dem Restaurant campus vita zu Essen gibt.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 5
Laden Sie Ihren Stundenplan als Liste herunter.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

1

Abbildung A.2: Fallstudie Seite 1

Aufgabe 6
Suchen Sie die Universitätsbibliothek per eingebauter Suchfunktion auf dem Lageplan und lassen Sie sich die Route vom Hörsaal 5D bis zum Campus Vita anzeigen.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 7
Lesen Sie sich eine beliebige Nachricht im Reiter News-Feed durch.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 8
Öffnen Sie die Webseite der Fachschaft Informatik.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 9
Öffnen Sie die Einstellungen und setzen Sie alle Parameter der Kategorie „Forum “.

Bitte bewerten Sie subjektiv die Schwierigkeit dieser Aufgabe:

leicht	eher leicht	eher schwer	schwer
☐	☐	☐	☐

Aufgabe 10
Bitte bewerten Sie die Applikation bezüglich folgender Kriterien:

	sehr gut	gut	nicht gut	schlecht
Community	☐	☐	☐	☐
Mensaplan	☐	☐	☐	☐
Anbindung LSF	☐	☐	☐	☐
Lageplan	☐	☐	☐	☐
News–Feed	☐	☐	☐	☐
Webseiten	☐	☐	☐	☐
Intuitivität der gesamten App	☐	☐	☐	☐
Nutzbarkeit im Alltag	☐	☐	☐	☐
Allgemeine Bewertung	☐	☐	☐	☐

Aufgabe 11
Was könnte verbessert werden, bzw. welche Funktion sollte mit aufgenommen werden?

2

Abbildung A.3: Fallstudie Seite 2

Protokoll der Fallstudie zur Bachelorarbeit

Eine Android-App für alltägliche Studieninformationen am
Beispiel der Heinrich-Heine-Universität Düsseldorf

von Tobias Krauthoff (tobias.krauthoff@.uni-duesseldorf.de)

Proband Nr.: _____

Aufgabe 1
Der Proband hat 5 min Zeit, um sich mit der Applikation vertraut zu machen. Gegenfalls kurze Hilfestellung geben.

Aufgabe 2
Der Proband beschreibt kurz seinen ersten Eindruck.

Aufgabe 3
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Aufgabe 4
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Aufgabe 5
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Abbildung A.4: Protokoll zur Fallstudie Seite 1

Aufgabe 6
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

_____

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Aufgabe 7
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

_____

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Aufgabe 8
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

_____

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Aufgabe 9
Aufgabe gelöst nach _____ min und _____ sec.
Bemerkungen:

_____

Verhalten des Probanden war:

zielstrebig	neugierig	normal	orientierungslos
☐	☐	☐	☐

Abbildung A.5: Protokoll zur Fallstudie Seite 2

Literaturverzeichnis

[CCB12] *Creative Commons Legal Code*. http://creativecommons.org/licenses/by-sa/2.0/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Chi12] CHILKAT SOFTWARE INC.: *Chilkat for Android Library Download and Install Instructions*. http://www.chilkatsoft.com/chilkatAndroid.asp. Version: 2012, Abruf: 04. August 2012.

[Des09] DESRUISSEAUX, B.: *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*. RFC 5545 (Proposed Standard). http://www.ietf.org/rfc/rfc5545.txt. Version: September 2009 (Request for Comments). Updated by RFC 5546

[Fel11] FELKER, Donn: *Android Application Development for DUMMIES*. Wiley Publishing, Inc., 2011. ISBN 978–0–470–77018–4

[Goo12a] GOOGLE INC.: *Activity | Android Developers*. http://developer.android.com/reference/android/app/Activity.html#ActivityLifecycle. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12b] GOOGLE INC.: *Android 3.0 Platform | Android Developers*. http://developer.android.com/about/versions/android-3.0.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12c] GOOGLE INC.: *The AndroidManifest.xml File | Android Developers*. http://developer.android.com/guide/topics/manifest/manifest-intro.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12d] GOOGLE INC.: *Application Fundamentals | Android Developers*. http://developer.android.com/guide/components/fundamentals.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12e] GOOGLE INC.: *Dashboards | Android Developers*. http://developer.android.com/about/dashboards/index.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12f] GOOGLE INC.: *Managing Projects | Android Developers*. http://developer.android.com/tools/projects/index.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12g] GOOGLE INC.: *Manifest.permission | Android Developers*. http://developer.android.com/reference/android/Manifest.permission.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12h] GOOGLE INC.: *mPortal Uni Koblenz-Landau - Android Apps auf Google Play*. https://play.google.com/store/apps/details?id=de.uni_koblenz_landau.studpo. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12i] GOOGLE INC.: *myTU - Android Apps auf Google Play*. https://play.google.com/store/apps/details?id=de.tufreiberg.mytu. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12j] GOOGLE INC.: *Preference Activity | Android Developers*. http://developer.android.com/reference/android/preference/PreferenceActivity.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12k] GOOGLE INC.: *Qudrant Advanced - Android Apps auf Google Play*. https://play.google.com/store/apps/details?idcom.aurorasoftworks.quadrant.ui.advanced. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12l] GOOGLE INC.: *Universität Hohenheim - Android Apps auf Google Play*. https://play.google.com/store/apps/details?id=com.excelsisnet.android.unihohenheim. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12m] GOOGLE INC.: *User Interface Guidelines | Android Developers*. http://developer.android.com/guide/practices/ui_guidelines/index.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Goo12n] GOOGLE INC.: *What is Android? | Android Developers*. http://developer.android.com/about/index.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Har12] HARLOW, Eric: *TICE: Experience - Multiple Android Activities in a TabActivity*. http://ericharlow.blogspot.de/2010/09/experience-multiple-android-activities.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Hei12a] HEINRICH-HEINE-UNIVERSITÄT DÜSSELDORF: *Universität Düsseldorf: HHU mobil*. http://m.hhu.de/. Version: Juli 2012, Abruf: 24. Juli 2012.

[Hei12b] HEINRICH-HEINE-UNIVERSITÄT DÜSSELDORF: *www.uni-duesseldorf.de/home:News*. http://www.uni-duesseldorf.de/home/rss.xml. Version: Juli 2012, Abruf: 25. Juli 2012.

[Ima12] IMAGEMAGICK STUDIO LLC: *ImageMagick: Convert, Edit, Or Compose Bitmap Images*. http://www.imagemagick.org/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Moz12] MOZDEV COMMUNITY ORGANIZATION: *mozdev.org - livehttpheaders: index*. http://livehttpheaders.mozdev.org/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Ort12] ORTIZ, Mike: *MikeOrtiz/TouchImageView*. https://github.com/MikeOrtiz/TouchImageView. Version: Juli 2012, Abruf: 25. Juli 2012.

[OSM12] *OpenStreetMap*. http://www.openstreetmap.org/. Version: Juli 2012, Abruf: 25. Juli 2012.

[php12] PHPBB DEUTSCHLAND E. V.: *Willkommen auf phpBB.de • phpBB.de*. https://www.phpbb.de/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Tau12] TAU CETI CO-OPERATIVE LTD.: *bouncycastle.org*. http://bouncycastle.org/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Tea12] TEAM REMICS: *Remics-UX | Samsung Galaxy S I9000*. http://remics.bennz.eu/download-Samsung_Galaxy_S_I9000. Version: Juli 2012, Abruf: 25. Juli 2012.

[The12] THEOPEN SSL PROJECT: *OpenSSL: The Open Source toolkit for SSL/TLS*. http://www.openssl.org/. Version: Juli 2012, Abruf: 25. Juli 2012.

[Tro12] TROST, Silke: *Nielsen: Deutschland | Nielsen präsentiert erste Ergebnisse aus aktuellem Smartphone Insights Report für Deutschland*. http://www.nielsen.com/de/de/insights/presseseite/2011/nielsen-praesentiert-erste-ergebnisse-aus-aktuellem-smartphone-i.html. Version: Juli 2012, Abruf: 24. Juli 2012.

[Ven12] VENESS, Chriss: *Calculate distance and bearing between two Latitude/Longitude points using Haversine formula in JavaScript*. http://www.movable-type.co.uk/scripts/latlong.html. Version: Juli 2012, Abruf: 25. Juli 2012.

[Vog12] VOGEL, Lars: *Dijkstra's Shortest Path Algorithm in Java*. http://www.vogella.com/articles/JavaAlgorithmsDijkstra/article.html. Version: Juli 2012, Abruf: 25. Juli 2012.

[Wha12] WHATSAPP INC.: *WhatsApp Messenger*. http://www.whatsapp.com/. Version: Juli 2012, Abruf: 26. Juli 2012.

[Yah12] YAHOO! INC.: *Yahoo! Weather - Yahoo! Developer Network*. http://developer.yahoo.com/weather/. Version: Juli 2012, Abruf: 24. Juli 2012.

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Düsseldorf, 06.August 2012

Tobias Krauthoff

Diese CD enthält:

- eine *pdf*-Version der vorliegenden Bachelorarbeit
- die LaTeX- und Grafik-Quelldateien der vorliegenden Bachelorarbeit samt aller verwendeten Skripte
- die Applikation, geschrieben in Java
- die Websites der verwendeten Internetquellen
- die digitalisierten Fragebögen der kontrollierten Fallstudie