ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

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

Entwicklung einer iOS Anwendung zur Unterstützung von Mitarbeiter der Qualitätssicherung vor Ort

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Markus Schiedel
markus.schiedel@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Johannes Schobel

2016

Fassung 27. Januar 2016

Satz: PDF-LaTeX 2ε

Kurzfassung

Mitarbeiter der Qualitätssicherung haben einen umfangreichen Aufgabenbereich, der sich damit beschäftigt, Kosten zu minimieren, die durch fehlerhafte Produkte entstehen. Sie sind dafür zuständig Fehler zu analysieren, um ihre Ursachen herauszufinden. Damit können Verfahren entwickelt werden, die das Auftreten des Fehlers verhindern. Eine detaillierte Analyse der Fehler erfordert ein grundlegendes Fachwissen aller Arbeitsschritte, die ein Produkt durchläuft. Dieses Wissen wird meist über entsprechende Fachliteratur bezogen. Solche Werke sind in der Regel sehr umfangreich und erfordern eine aufwändige Suche, um Informationen zu finden, die für einen speziellen Fehler relevant sind.

Aus diesem Grund ist das Ziel dieser Arbeit die Entwicklung einer mobilen Anwendung, welche die Mitarbeiter der Qualitätssicherung bei der Fehleranalyse unterstützt. Dafür werden zunächst einige Grundlagen vorgestellt, die sich mit Normen und der Funktionsweise eines REST Webservices beschäftigen. Außerdem wird die Struktur eines Pakets definiert, das die Fachliteratur der Qualitätssicherung enthält. Anschließend werden die Anforderungen dieser Anwendung analysiert, die den späteren Funktionsumfang beschreiben. Auf diesen Grundlagen wird die Implementierung der Anwendung vorgestellt und mit anderen Anwendungen verglichen, die ähnliche Ziele haben. Abschließend werden mögliche Erweiterungen für die Anwendung vorgestellt.

Inhaltsverzeichnis

1

Einleitung

Qualitätssicherung spielt in vielen Unternehmen eine wichtige Rolle, da sie sich damit beschäftigt, auftretende Mängel in Produkten und Prozessen zu minimieren. Jeder Fehler kann hohe Kosten verursachen, zum Beispiel durch Garantieansprüche der Kunden, Rückrufaktionen gefährlicher Produkte oder Verkaufseinbußen durch mangelhafte Qualität. Aus diesem Grund wird Qualitätssicherung im gesamten Entstehungsprozess eines Produktes eingesetzt. Die Mitarbeiter der Qualitätssicherung beschäftigen sich mit der Erkennung und Analyse von Fehlern, um anschließend Verfahren zu erarbeiten, die das Auftreten der Fehler minimiert. Durch laufende Kontrollen der Produkte wird überprüft, ob die Änderungen an den Verfahren Wirkung zeigen und sich die Qualität verbessert. In vielen Fällen sind Arbeitsprozesse und Qualitätsmerkmale der Produkte an geltende Normen gekoppelt [1]. Aus diesem Grund erfordert diese Aufgabe ein umfangreiches Wissen über die unternehmensinternen Abläufe, über die verschiedenen Arbeitsprozesse und über die einzuhaltenden Normen.

1.1 Problemstellung

Um erkannte Fehler oder Mängel bis ins Detail analysieren zu können, kann es für einen Mitarbeiter der Qualitätssicherung notwendig sein, sich zusätzliche Informationen in entsprechender Fachliteratur zu besorgen. Da die Produkte meist verschiedene Arbeitsschritte durchlaufen, die jeweils unterschiedliche Fachkenntnisse voraussetzen, müssen verschiedene Fachbücher nach möglichen Ursachen durchsucht werden. Der Mitarbeiter weiß dabei nicht mit welchem Buch er seine Suche beginnen soll und durchsucht im ungünstigsten Fall mehrere Bücher, die keine Informationen zu dem Problem beinhalten. Hinzu kommt, dass diese Werke in der Regel sehr umfangreich sind und unter Umständen nur wenige Seiten Informationen enthalten, die für die Bearbeitung des erkannten Problems relevant sind. Dies zieht den gesamten Prozess der Fehleranalyse in die Länge, wodurch zusätzliche Kosten für das Unternehmen entstehen können. Denn je länger die Ursache eines Problems nicht behoben wird, desto mehr Produkte werden mit dem gleichen Fehler produziert.

Daraus entsteht die Fragestellung, ob diese Fachliteratur, zusammen mit weiteren Informationen zur Fehleranalyse, auf einem mobilen Endgerät bereitgestellt werden kann. Um schnell auf diese Informationen zugreifen zu können, müsste es eine Suche geben, mit der gezielt nach bestimmten Themen gesucht werden kann. Weiterhin wäre es hilfreich, wenn Informationen aus verschiedenen Themenbereichen untereinander verlinkt werden könnten, um schnell einen Überblick über relevante Themen erhalten zu können.

1.2 Zielsetzung

Um diese Aufgaben zu lösen, wird in dieser Arbeit eine mobile Anwendung vorgestellt, deren Ziel es ist, für die Qualitätssicherung benötigte Literatur einfach und effizient zur Verfügung zu stellen. Dies bedeutet, dass keine Einlernphase zur Bedienung der Anwendung nötig ist und dass der Benutzer in wenigen Schritten die gewünschte Informationen angezeigt bekommt.

Um dies zu ermöglichen wird die Anwendung so implementiert, dass sie Informationen unabhängig von ihrem Inhalt anzeigen kann. Die einzige Voraussetzung dafür ist, dass diese Informationen zu Paketen zusammengefasst werden, die einer bestimmten Struktur folgen. Dadurch wird es möglich, Informationen beliebig untereinander zu verlinken, um ohne weiteres Suchen zusätzliche Informationen abrufen zu können. Gleichzeitig kann der Benutzer mit einer Stichwortsuche gezielt auf Informationen zugreifen. Mit einem Login wird sichergestellt, dass jeder Benutzer nur seine eigenen und für ihn relevanten Informationen angezeigt bekommt. Da am Einsatzort unter Umständen keine Internetverbindung vorhanden ist, müssen die Informationen auch offline bereitgestellt werden. Zudem kann der Datenbestand auf dem Gerät erweitert werden, indem neue Informationen individuell über einen REST Webservice heruntergeladen werden.

1.3 Struktur der Arbeit

In Kapitel 2 dieser Arbeit werden Grundlagen für ein einheitliches Verständnis der Arbeit geschaffen. Dazu gehört die Definition wichtiger Begriffe, Informationen über die Entstehung und den Nutzen von Normen, und die Funktionsweise eines REST Webservice. Außerdem wird die Struktur der Pakete definiert, die die Fachliteratur der Qualitätssicherung enthalten sollen. In Kapitel 3 werden die Anforderungen der mobilen Anwendung analysiert. Diese bestehen aus funktionalen und nichtfunktionalen Anforderungen. Zusätzlich werden Konzepte zur Bedienung der Anwendung erarbeitet. Das Kapitel 4 dokumentiert die Implementierung der mobilen Anwendung. Diese besteht aus der Dialogstruktur, welche die Navigation innerhalb der Anwendung beschreibt, der Dialoggestaltung, der Architektur und den wichtigsten Funktionen der Anwendung. In Kapitel 5 werden zwei bereits bestehende Anwendungen, die ähnliche Ziele verfolgen, analysiert und mit dieser Implementierung verglichen. Zuletzt werden die Ergebnisse dieser Arbeit in Kapitel 6 zusammengefasst und mit einem Ausblick Anregungen für zukünftige mögliche Erweiterungen geschaffen.

2

Grundlagen

In diesem Kapitel werden einige Grundlagen erarbeitet, die zu einem besseren Verständnis der Arbeit beitragen. Dafür werden zuerst wichtige Begriffe definiert, um ihnen eine eindeutige Bedeutung zuzuordnen. Anschließend wird der Entstehungsprozess und der Nutzen von Normen erläutert. Danach wird erklärt, wie der Webservice generell funktioniert, von dem die Anwendung Fachliteratur herunterladen kann und wie diese Literatur in ein für die Anwendung verständliches Format gebracht werden kann.

2.1 Fachwissen

In diesem Abschnitt wird die Bedeutung wichtiger Begriffe, die in dieser Arbeit verwendet werden, definiert und erklärt.

Repository stellt eine thematische Gruppierung verschiedener Inhalte dar.

Version ist eine spezielle Fokussierung der Inhalte nach einem bestimmten Kriterium.

Paket beschreibt den tatsächlichen Inhalt, also die Dokumente einer Version. Verschiedene Pakete der selben Version stellen Revisionen dieser dar und unterscheiden sich in der Regel nur geringfügig voneinander.

Inhalte eines Pakets sind Ordner und Dateien beliebiger Formate, wie Texte, Bilder, etc.

Anschaulich dargestellt kann man sich ein Repository als Ordner in einem Dateisystem vorstellen. Dieser Ordner hat einen Namen und wird in der Regel mit Dateien befüllt, die thematisch etwas mit dem Ordnernamen zu tun haben. Eine Version stellt dann einen weiteren Ordner in diesem Repository dar und dient dazu, die Dateien nach einem bestimmen Kriterium genauer zu sortieren oder zu filtern. Im Ordner der Version sind dann die tatsächlichen Inhalte zu finden, die das Paket bilden, also Texte, Bilder, Videos oder andere Dokumente.

Ein denkbares Beispiel wäre ein Repository mit dem Namen *Lackieren*. Darin wird alles zu diesem Thema gesammelt. Um die Inhalte noch genauer zu sortieren, werden drei Versionen mit den Namen *Nasslackieren*, *Trockenlackieren* und *Sonstiges* erstellt. In diesen Versionen werden dann die zugehörigen Dateien abgelegt. Abbildung 2.1 zeigt das Repository *Lackieren* mit diesen drei Versionen. In der Version *Nasslackieren* sind mögliche Inhalte zu sehen. Dies sind entweder Dateien, oder weitere Ordner, die den Inhalt weiter strukturieren.

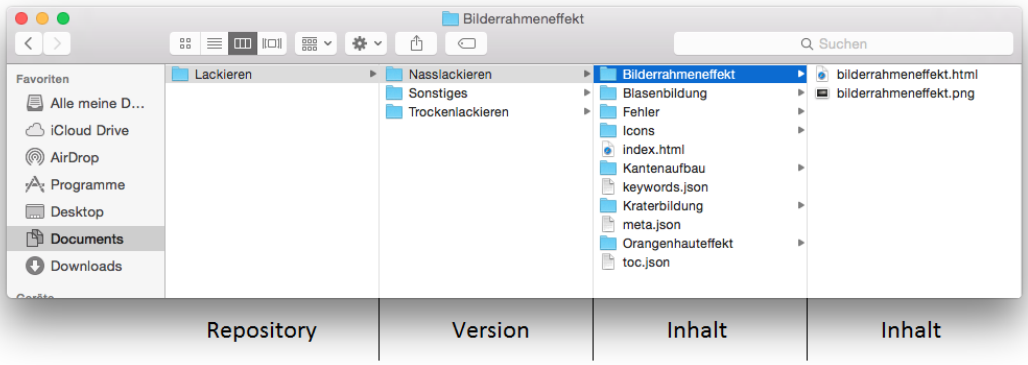

Abbildung 2.1: Möglicher Aufbau eines *Lackieren*-Repositories

Server ist eine online Anwendung, welche sich konform zur definierten API verhält. Der Server übernimmt die Authentifizierung der Benutzer und stellt Pakete zum Download bereit.

API ist die Programmierschnittstelle, über die Server und mobile Anwendung kommunizieren können.

Mobile Anwendung wird auf einem Smartphone oder Tablet ausgeführt.

System fasst den Server und die mobile Anwendung zusammen.

2.2 Normen

Zu den Aufgaben der Qualitätssicherung gehört es, Produkte und Prozesse zu analysieren, um mögliche Fehler zu erkennen und entsprechende Lösungsmaßnahmen zu entwickeln. In vielen Fällen dienen Normen dafür als Maßstab. In diesem Abschnitt wird erklärt, wie eine solche Norm entsteht und welche Vorteile sich daraus entwickeln.

2.2.1 Entstehung der Normen

Um sicherzustellen, dass die Normen den Interessen der Allgemeinheit genügt, wird bei der Entstehung oder Bearbeitung einer Norm ein bestimmter Prozess [3][4] durchlaufen. Eine Norm kann auf insgesamt drei Ebenen entstehen: Auf nationaler, europäischer oder internationaler Ebene. Während eine europäische Norm unverändert ins deutsche Normwerk übernommen werden muss, ist dies bei internationalen Normen nicht der Fall. Durch parallele Erarbeitungsverfahren kann eine Norm auf allen drei Ebenen eingeführt werden. Dabei vertritt das Deutsche Institut für Normung, kurz DIN, die Interessen der deutschen Experten. Das Europäische Komitee für Normung (CEN) verwaltet neben dem Europäischen Komitee für elektrotechnische Normung (CENELEC) das Bearbeiten und Erstellen europaweiter Normen. Die Internationale Organisation für Normung (ISO) und die Internationale elektrotechnische Kommission (IEC) verwalten die Erarbeitung internationaler Normen.

National Der erste Schritt für eine neue nationale Norm besteht aus einem formlosen Antrag, den jeder bei DIN einreichen kann. Es wird überprüft, ob Bedarf für dieses Thema besteht und ob die beim Normungsprozess beteiligten Kreise bereit sind das Projekt zu finanzieren. Ist dies der Fall, wird zunächst ein Entwurf erstellt und durch den Beuth Verlag [5] veröffentlicht. Anschließend kann von der Öffentlichkeit Stellung zum Entwurf genommen werden. Abschließend berät ein Ausschuss unter Berücksichtigung der Stellungnahmen über den finalen Inhalt der geplanten Norm und veröffentlicht die Norm gegebenenfalls.

Europäisch Soll eine neue europäische Norm erarbeitet werden, geschieht dies nur auf Antrag bestimmter Organisationen. Bevor der Normungsprozess beginnt, müssen die einfache Mehrheit und 71 Prozent der gewichteten Mehrheit der abstimmenden Organisationen dem Antrag zustimmen. Zudem müssen sich ausreichend viele Organisationen zur Mitarbeit verpflichten. Anschließend wird wie bei einer nationalen Norm überprüft, ob Bedarf für dieses Thema besteht und ob die beim Normungsprozess beteiligten Kreise bereit sind das Projekt zu finanzieren. Zusätzlich wird geprüft, ob bereits eine internationale Norm zu diesem Thema existiert, um diese gegebenenfalls als europäische Norm zu übernehmen.

Falls eine neue Norm erarbeitet werden soll, wird zunächst ein Norm-Entwurf erstellt und allen nationalen Normungsorganisationen zur Verfügung gestellt, die innerhalb von drei Monaten eine nationale Stellungnahme abgeben müssen. In Deutschland wird dazu eine deutsche Fassung veröffentlicht, bei der alle Interessenten in den ersten zwei Monaten die Möglichkeit haben eine eigene Stellungnahme zu verfassen. Im Anschluss berät sich der nationale Ausschuss über den Inhalt der nationalen Stellungnahme. Diese beinhaltet entweder einen überarbeiteten Entwurf oder eine Zustimmung zur Veröffentlichung des Entwurfs.

Falls nach den drei Monaten die Annahmekriterien für den Norm-Entwurf erfüllt werden, wird diese veröffentlicht. Ansonsten wird ein Schluss-Entwurf erstellt, über den innerhalb von zwei Monaten abgestimmt wird. Um diese Abstimmung erfolgreich abzuschließen muss eine einfache Mehrheit und mindestens 71 Prozent der gewichteten Stimmen der CEN- und CENELEC-Mitglieder für den Schluss-Entwurf stimmen.

International Analog zu einer europäischen Norm wird eine internationale Norm nur auf Antrag bestimmter Organisationen erarbeitet. Außerdem muss die einfache Mehrheit der auf dem Fachgebiet beteiligten nationalen Normungsorganisationen dem Antrag zustimmen und sich eine ausreichende Anzahl an Mitgliedern zur Mitarbeit verpflichten. Ist dies der Fall, erarbeitet eine Arbeitsgruppe innerhalb von zwei Monaten einen Komitee-Entwurf, der anschließend von den in der Arbeitsgruppe beteiligten nationalen Normungsorganisationen kommentiert wird. Falls genug Organisationen dem Komitee-Entwurf zustimmen, wird ein Norm-Entwurf erstellt und an alle ISO- bzw. IEC-Mitglieder verteilt, der analog zur europäischen Norm innerhalb von drei Monaten kommentiert werden kann.

Falls nach den drei Monaten dem Norm-Entwurf zugestimmt wird, wird er als internationale Norm veröffentlicht. Ansonsten wird ein Schluss-Entwurf erstellt, über den innerhalb von zwei Monaten abgestimmt wird. Damit der Schluss-Entwurf als internationale Norm veröffentlicht werden kann, müssen mindestens zwei Drittel aller beteiligten Mitglieder bei der Abstimmung zustimmen und es darf nicht mehr als ein Viertel negative Stimmen geben. Abbildung 2.2 zeigt die Entstehung einer internationalen Norm, die den gesamten Prozess durchläuft.

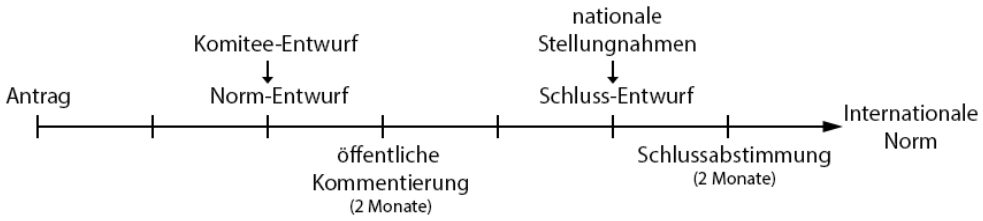

Abbildung 2.2: Entstehung einer internationalen Norm

2.2.2 Nutzen der Normen

Normen finden sich in fast allen Bereichen des Alltags wieder und helfen dabei, Sicherheits- und Qualitätsmerkmale festzulegen. Im folgenden wird erklärt, wie die verschiedenen Partien von den Normen profitieren können.

Verbraucher Der Verbraucherschutz selbst spielt in den Normen eine große Rolle [6], denn es werden unter anderem wichtige Fragen bezüglich der Sicherheit von Produkten geklärt. Des weiteren befassen sie sich neben der Verträglichkeit von Produkten und Prozessen mit Gesundheit und Umwelt auch mit dem Verbraucher selbst, da auch Themen wie Barrierefreiheit oder Elektromobilität behandelt werden. Zudem wird darauf geachtet, dass Produkte eine gewisse Gebrauchstauglichkeit besitzen und dass angewandte Verfahren rechtmäßig sind.

Wirtschaft Ein Unternehmen kann auf zwei verschiedene Weisen von Normen profitieren [4]. Eine Möglichkeit besteht darin sich aktiv am Normungsprozess zu beteiligen. Dafür muss ein Unternehmen zuerst Geld investieren, da Arbeitseinsatz und Reisen der eigenen Experten Kosten verursacht. Im Gegenzug erhält das Unternehmen allerdings die Möglichkeit eigene Interessen in den Prozess einzubringen und erhält Informationen zum aktuellen Stand der Technik, da Wissen mit allen beteiligten Kreisen ausgetauscht wird. Gleichzeitig bietet dies die Chance, Konkurrenten zu beobachten und verschafft einen Wettbewerbsvorteil gegenüber Unternehmen, die sich nicht am Normungsprozess beteiligen. Die zweite Möglichkeit von Normen zu profitieren besteht für ein Unternehmen darin sie selbst anzuwenden. Durch den Einsatz von genormten Arbeitsprozessen entstehen Effizienzsteigerungen und Kosteneinsparungen im gesamten Unternehmen. Außerdem werden dadurch Auftragsverhandlungen und Markteinführungen vereinfacht, da auf bekannte Verfahren und Qualitätsmerkmale zurückgegriffen werden kann. Dies gibt den Verbrauchern ein Vertrauen zu den Produkten, da sie wissen, dass gewisse Mindestanforderungen erfüllt werden müssen.

Staat Normen entlasten den Staat bei der Gesetzgebung [7], da sich Rechtsvorschriften nur auf Rahmenbedingungen beschränken können. Für die Umsetzung dieser Bedingungen kann dann auf die jeweilige Norm verwiesen werden. Normen beschreiben immer den aktuellen Stand der Technik, da der Normungsprozess vorschreibt, dass sie spätestens alle fünf Jahre überprüft und angepasst werden müssen. Dadurch wird die Aufgabe, die Rahmenbedingungen technisch umzusetzen, an Experten übertragen.

2.3 REST Webservices

Der Server, mit dem die mobile Anwendung kommuniziert, ist ein REST Webservice. Der Begriff REST steht für Representational State Transfer [8]. In diesem Abschnitt werden die wesentlichen Eigenschaften eines solchen Webservices erläutert.

Client-Server Die erste Eigenschaft beschreibt, dass alle Eigenschaften der Client-Server-Architektur erfüllt werden müssen. Das Grundprinzip dieser Architektur besagt, dass alle Aufgaben des User-Interfaces getrennt von den Daten zu behandeln sind. Der Server kümmert sich dabei ausschließlich um die Datenhaltung.

Zustandslosigkeit Die Kommunikation zwischen Client und Server muss zu jedem Zeitpunkt zustandslos sein. Das bedeutet, dass der Client bei jeder Anfrage an den Server alle notwendigen Informationen mitsenden muss, die der Server benötigt, um die Anfrage zu verstehen.

Caching Der Client kann entsprechend markierte Daten cachen und sie zu einem späteren Zeitpunkt wiederverwenden, um die Performance des Systems zu erhöhen.

Einheitliche Schnittstelle Alle Ressourcen, sprich Daten des Servers sollen über eine einheitliche Schnittstelle zugreifbar sein. Um eine solche Schnittstelle zu erhalten müssen 1., 2. und 3. der folgenden Eigenschaften erfüllt werden:

1. Adressierbare Ressourcen: Jede Ressource auf dem Server muss adressierbar sein. Im Web wird dafür eine URL verwendet.
2. Verschiedene Repräsentationen der Ressourcen: Jede Ressource kann prinzipiell an die Anforderungen des Clients angepasst werden, indem sie in eine entsprechendes Format umgewandelt wird. Typische Dateiformate sind JSON oder XML.
3. Selbstbeschreibende Nachrichten: Jede Nachricht muss alle Informationen enthalten um vom Server, bzw. Client als Ganzes verstanden werden zu können.
4. Hypermedia As The Engine Of Application State (HATEOAS): Zusätzlich zu den Ressourcen werden vom Server Links zu zugehörigen Ressourcen mitgeschickt. Dies bietet eine Grundlage für die zustandslose Kommunikation zwischen Client und Server, da damit die Navigation des Servers definiert wird. Der Client muss

nur eine URL kennen, die Einstiegs-URL des Servers. Alle weiteren URLs werden dynamisch erzeugt und je nach Anfrage an den Client gesendet. Das Erfüllen dieser Eigenschaft ist optional, da es in der Regel sehr viel Aufwand bedeutet.

Mehrschichtiges System Die Aufgaben des Servers können auf einzelne Komponenten aufgeteilt werden, die jeweils nur eine bestimmte Aufgabe erledigen. Typische Aufgaben sind Load-Balancing, Datentransformationen, Caching, Authentifizierung und Autorisierung.

Code-On-Demand Die Funktionalitäten eines Clients können erweitert werden, indem zusätzlicher Code in Form von Applets oder Skripten vom Server heruntergeladen wird. Allerdings ist das Erfüllen dieser Eigenschaft optional, da das Herunterladen und Ausführen von Code nicht in allen Szenarien möglich ist. Zum Beispiel könnten Java Applets von einer Firewall oder den Browsereinstellungen blockiert werden.

2.4 Definition der Inhalte eines Pakets

Damit die Fachliteratur der Qualitätssicherung von der Anwendung verarbeitet und angezeigt werden kann, muss sie in Pakete zusammengefasst werden und dabei bestimmte Kriterien erfüllen, die im Folgenden definiert werden. Einige Dateien müssen dafür an vorgegebenen Pfaden liegen. Diese werden als relative Pfade angegeben und haben als Referenz die "Inhalt"-Ebene aus Abbildung 2.1. Alle Pfade, die innerhalb des Pakets verwendet werden, folgen derselben Regel.

Die Navigation innerhalb eines Pakets wird zusammen mit den Inhalten, die angezeigt werden können, in einer *table of contents*-Datei definiert. Die Datei hat den Namen *toc.json* und wird am Pfad *./toc.json* erwartet. Dabei ist der Aufbau aus Anhang A.1 vorgesehen. Das oberste Objekt heißt *navigation* und beinhaltet eine Liste an rekursiven Objekten, von denen jedes einen Inhalt im Paket repräsentiert. Jeder Inhalt besitzt die Elemente *id*, *title*, *uri*, *decription*, *icon* und *children*. Bei der URI kann eine Datei referenziert werden, die beim öffnen des Inhalts angezeigt werden soll. Falls ein Pfad für ein Icon angegeben ist, wird es zusammen mit dem Titel und der Beschreibung in

der Navigation angezeigt. Für die Liste *children* werden die gleichen rekursiven Objekte verwendet, um damit einen Navigationsbaum aufbauen zu können.

Um die Suche in den Repositories so effizient wie möglich zu gestalten, wird keine Volltextsuche verwendet, sondern für jede Version werden Stichwörter in einer Datei hinterlegt, die den Inhalt in wenigen Worten zusammenfassen. Diese Datei muss den Namen *keywords.json* haben und wird am Pfad *./keywords.json* erwartet. Die Datei muss den Vorgaben aus Anhang A.2 entsprechen. Das oberste Objekt heißt *keywords* und beinhaltet eine Liste an Objekten. Jedes dieser Objekte besitzt eine Liste an Stichwörtern und eine URI, die eine zu den Stichwörtern passende Datei referenziert.

Zu jeder Version gibt es zusätzliche Informationen, die nichts mit dem Inhalt zu tun haben. Diese werden in der *meta.json* am Pfad *./meta.json* festgehalten. Dabei werden alle Informationen, die in Anhang A.3 zu sehen sind, gespeichert.

Für jeden Inhalt in einem Paket wird vorgesehen, dass er in einem HTML-Dokument aufbereitet wird. Dafür werden die Skripte und Stylesheets von jQuery [9] und Bootstrap [10] von der Anwendung bereitgestellt und müssen nicht in einem Paket mitgeliefert werden. Möchte man zusätzliche Anpassungen vornehmen, können eigene Skripte in der Datei *script.js* am Pfad *./js/script.js* und eigene Stylesheets in der Datei *style.css* am Pfad *./css/style.css* mitgeliefert werden. Diese Dateien sind optional und müssen nur bei Bedarf dem Paket beigelegt werden.

Alle anderen Dateien, insbesondere die Inhalte, können beliebig innerhalb des Pakets strukturiert und benannt werden, da die entsprechenden Informationen in der *toc.json* hinterlegt sind. Abbildung 2.1 zeigt ein Paket, das diesen Vorgaben entspricht.

Um Inhalte in einem HTML-Dokument zu verlinken gibt es zwei Möglichkeiten. Die erste Möglichkeit erlaubt es ausschließlich Inhalte aus dem selben Paket zu verlinken. Dafür wird an der gewünschten Stelle des HTML-Dokuments ein *a*-Tag mit dem Attribut *href* eingesetzt. Der Wert dieses Attributs muss der relative Pfad zum Inhalt sein, der verlinkt werden soll. Ein Link, der die *index.html* aus Abbildung 2.1 verlinkt, würde so aussehen:
Link
Mit der zweiten Möglichkeit können Inhalte aus einem anderen Paket verlinkt werden. Dafür wird ebenfalls das *a*-Tag verwendet. Allerdings haben diese Links zusätzlich zum

Attribut *href* noch die Attribute *data-repository*, *data-version* und *data-uri*. Der Wert des Attributs *href* besteht bei diesem Link nur aus einem Platzhalter, den die Anwendung gegebenenfalls ersetzt. Alle anderen Attribute beziehen sich auf das Paket, in dem ein Inhalt verlinkt werden soll. Die Attribute *data-repository* und *data-version* haben den Wert der entsprechenden ID, die auch in der *meta.json* verwendet wird. Das Attribut *data-uri* muss als Wert den relativen Pfad des Inhalts haben, der verlinkt werden soll. Ein entsprechender Link könnte so aussehen:
Link

Mit Hilfe dieser Grundlagen werden im nächsten Kapitel die Anforderungen an eine Anwendung analysiert, die Mitarbeiter der Qualitätssicherung bei der Fehleranalyse unterstützen kann.

3

Anforderungsanalyse

In diesem Kapitel werden alle Anforderungen der mobilen Anwendung erläutert. Zuerst werden dazu sowohl die funktionalen als auch die nichtfunktionalen Anforderungen analysiert. Anschließend werden die Anforderungen im Hinblick auf die beteiligten Rollen und den dazugehörigen Abläufen genauer betrachtet, um sie abschließend in einem konzeptuellen Design des User-Interface zu veranschaulichen.

3.1 Funktionale Anforderungen

In diesem Abschnitt werden alle funktionalen Anforderungen der mobilen Anwendung aufgelistet und erläutert. Diese Anforderungen beschreiben, was die Anwendung leisten soll, also die Aufgaben, die der Benutzer mit Hilfe der Anwendung bewältigen kann.

FA1 Die Anwendung soll einen Login besitzen, der nur autorisierten Benutzern Zugang zur Anwendung gewährt. Dafür soll eine E-Mail Adresse, bzw. ein Benutzername, ein Passwort und eine URL eingegeben werden. Um erfolgreich eingeloggt zu werden, müssen die Daten vom Server an der angegebenen URL bestätigt werden.

FA2 Für den Fall, dass keine Internetverbindung vorhanden ist, soll es möglich sein, dass sich der Benutzer mit seinen Daten auch offline einloggen kann. Voraussetzung dafür ist, dass sich der jeweilige Benutzer mindestens ein Mal erfolgreich am Server eingeloggt hat. Von den Daten, die dafür lokal gespeichert werden, muss mindestens das Passwort angemessen verschlüsselt werden.

FA3 Die Anwendung soll zwischen verschiedenen Benutzern unterscheiden können. Das heißt, es soll für jeden Benutzer jederzeit möglich sein sich online und offline einzuloggen. Dies bedeutet, dass die Anwendung für jeden Benutzer unterschiedliche Dateien verwalten soll. Damit soll jeder Benutzer nur seine eigenen Repositories und Versionen sehen können.

FA4 Die Anwendung sollte Einstellungen zur Verfügung stellen, die jeder Benutzer individuell für sich anpassen kann. Es sollte möglich sein sich warnen zu lassen, bevor Daten über eine mobile Datenverbindung ausgetauscht werden. Außerdem soll der Benutzer einstellen können, ob die Anwendung automatisch nach dem Login die lokalen Pakete auf Updates überprüfen soll.

FA5 Der Benutzer soll die Möglichkeit haben seine eigenen Daten zu löschen. Dies sind zum Einen nicht mehr benötigte Pakete und zum Anderen die Benutzerdaten, die für den Login-Prozess ohne Internetverbindung gespeichert werden.

FA6 Die Anwendung soll über eine definierte API mit dem Server kommunizieren können und alle Funktionen der API auf dem Gerät abbilden.

FA7 Die Anwendung soll die Liste aller online verfügbaren Repositories abrufen und anzeigen können. Zusätzlich soll der Benutzer die Möglichkeit haben einen Suchbegriff einzugeben, der an den Server übermittelt wird, welcher die Liste der Repositories anhand des Suchbegriffs eingrenzt. Außerdem müssen für alle Repositories die zugehörigen Versionen geladen und angezeigt werden können.

FA8 Für eine online Version soll es dem Benutzer möglich sein diese zu kaufen, abonnieren, herunterladen oder prüfen, ob es seit dem letzten Download ein neues Paket gibt. Außerdem soll der Benutzer das Abonnement einer Version wieder beenden können, wobei er auswählen kann, ob nur das aktuell verwendete Gerät, oder alle Geräte gekündigt werden sollen. Wird ein Paket heruntergeladen, soll dieses automatisch dem entsprechenden Benutzer zugewiesen werden. Falls es mehrmals heruntergeladen wird, soll das alte Paket überschrieben werden.

FA9 Der Benutzer soll seine auf dem Server gespeicherten Informationen einsehen können. Dies sind unter anderem allgemeine Profilinformationen, wie zum Beispiel Vor- und Nachname. Dazu gehören auch Informationen darüber, welche Repositories, bzw. Versionen der Benutzer gekauft, bzw. abonniert hat.

FA10 Alle Pakete, die auf dem Gerät gespeichert sind, sollen in einer Liste dargestellt werden. Durch die Auswahl eines Pakets soll dieses geöffnet werden und die Anwendung soll die Navigation des Pakets anzeigen. Wird ein Dokument ausgewählt, soll der Inhalt geladen und angezeigt werden. Zusätzlich soll es möglich sein Inhalte aus verschiedenen Paketen zu verlinken, um Querbezüge herstellen zu können.

FA11 Der Benutzer soll mit Hilfe eines Suchfeldes eine Stichwortsuche anstoßen können, die ihm alle passenden Treffer anzeigt. Dabei soll es möglich sein sowohl in allen verfügbaren Repositories als auch in einer ausgewählten Version zu suchen. Wird anschließend ein Treffer ausgewählt, soll das darin referenzierte Dokument analog zu FA10 geöffnet und angezeigt werden.

FA12 Die Anwendung soll eine Möglichkeit bieten sich auszuloggen, so dass keine Daten ohne einen erneuten Login einsehbar sind.

FA13 Die Anwendung sollte auf Push-Benachrichtigungen vom Server reagieren können und den Inhalt dem Benutzer präsentieren, selbst wenn die Anwendung nicht geöffnet ist.

3.2 Nichtfunktionale Anforderungen

In diesem Abschnitt werden alle nichtfunktionalen Anforderungen der mobilen Anwendung aufgelistet und erläutert. Diese Anforderungen beschreiben zusätzliche Kriterien, welche die Anwendung erfüllen soll, um eine möglichst hohe Qualität zu erreichen.

N-FA1 Die Benutzbarkeit, bzw. Usability der Anwendung soll bei der Gestaltung eine große Rolle spielen. Dabei soll es sowohl für unerfahrene Benutzer, als auch für Experten möglich sein, die Anwendung ohne Training intuitiv bedienen zu können. Das heißt, alle Funktionen müssen als solche klar erkennbar sein. Dazu gehört ebenfalls, dass die Anwendung dem Benutzer jederzeit Informationen über den aktuellen Status anzeigt. Dadurch soll klar erkennbar sein, ob eine Eingabe bereits verarbeitet wurde, oder ob noch Wartezeiten zu erwarten sind. Außerdem soll sich die Anwendung an die von iOS vorgegebenen UI-Styleguides halten, dass sich die Benutzer an bekannten Bedienkonzepten orientieren können.

N-FA2 Die Anwendung muss so robust wie möglich programmiert werden, um Abstürze zu vermeiden. Als potentielle Fehlerquellen sind der Benutzer und der Server anzusehen. Dem Benutzer soll es nicht möglich sein die Anwendung durch eine unvorhergesehene Bedienweise zum Absturz zu bringen. Zusätzlich ist es wichtig, dass er keine Daten aus Versehen lösschen kann. Falls die Anwendung unerwartete Antworten vom Server bekommt, muss sie damit umgehen können. Dies ist zum Beispiel dann der Fall, wenn sich der Server nicht konform zur API verhält.

N-FA3 Die Anwendung soll zuverlässig sein und die definierten Funktionalitäten umsetzen. Das heißt es ist darauf zu achten, dass alle Eingaben auch die erwarteten Ausgaben produzieren.

N-FA4 Außerdem soll sie effizient gestaltet werden. Das bedeutet einerseits, dass alle Funktionen schnell und einfach zu finden sind. Andererseits sollen ineffiziente Algorithmen oder viele Serveranfragen möglichst vermieden werden, um eventuelle Ladezeiten so kurz wie möglich zu halten.

3.3 Anwendungskontext

In diesem Abschnitt werden die funktionalen Anforderungen nochmals mit einem anderen Blickwinkel betrachtet. Zuerst werden die Anwendungsfälle mit den zugehörigen Akteuren aufgelistet. Anschließend werden die Anwendungsfälle in kleine Teilschritte zerlegt, um ihren gesamten Ablauf zu verdeutlichen.

3.3.1 Anwendungsfälle

Bei der Benutzung der Anwendung wird zwischen zwei Akteuren unterschieden: Einem Besucher und einem Benutzer.

Wie in FA1 beschrieben und Abbildung 3.1 dargestellt, hat ein Besucher nur die Möglichkeit sich am System anzumelden. Alle anderen Funktionen stehen ihm nicht zur Verfügung. Dabei kann nicht ausgewählt werden, ob der Besucher sich online oder offline anmelden möchte, dies wird von der Anwendung entschieden. Sobald sich ein Besucher erfolgreich am System angemeldet hat, gilt er als Benutzer und kann die Anwendung in ihrem gesamten Umfang benutzen. Ein Benutzer kann seine eigenen, lokalen Repositories verwalten und sich die online verfügbaren Repositories anzeigen lassen. Zu jedem online Repository können die zugehörigen Versionen angezeigt werden, um anschließend die definierten Interaktionen einer Version durchzuführen, die etwas später beschrieben werden. Zudem kann der Benutzer seine bereits gekauften, oder seine

abonnierten Versionen einsehen, indem er zunächst sein Profil öffnet. Abschließend kann der Benutzer seine individuellen Einstellungen zur Benutzung der Anwendung ändern, oder sich von der Anwendung abmelden. Dies entspricht den Anforderungen FA4, FA7, FA9 und FA12.

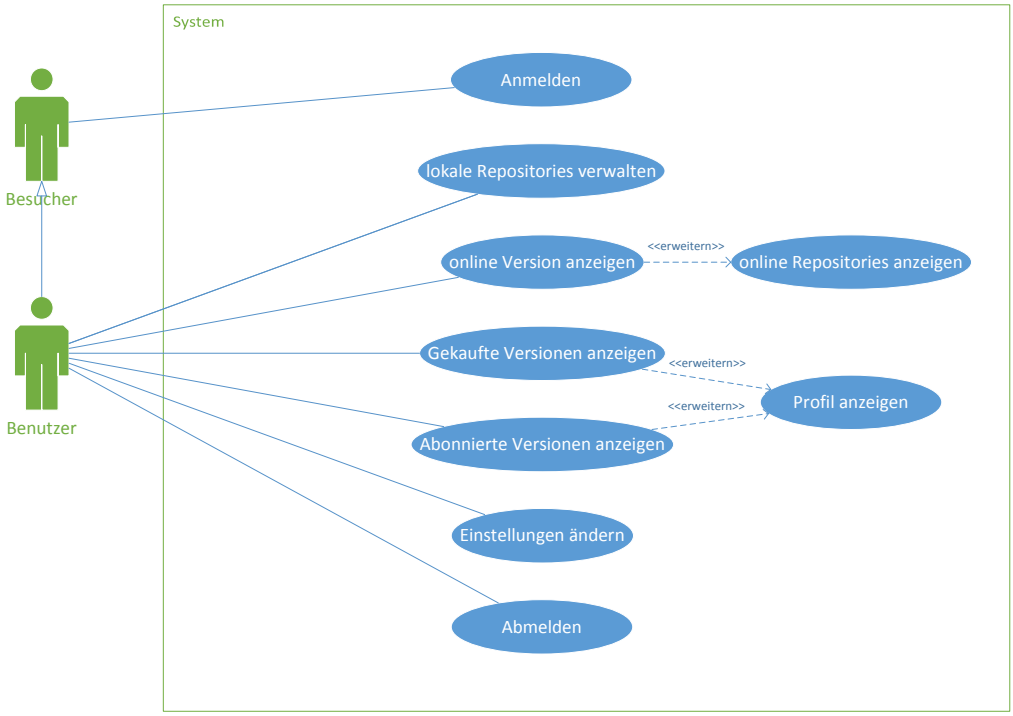

Abbildung 3.1: Anwendungsfälle des Systems

Konform zu FA10 und FA11 kann der Benutzer durch seine lokal gespeicherten Repositories navigieren und sich Dokumente daraus anzeigen lassen. Außerdem kann der Benutzer durch eine Eingabe in ein Suchfeld nach bestimmten Dokumenten suchen und sie im nächsten Schritt anzeigen lassen. Ebenfalls ist es ihm möglich seine Pakete zu löschen. Dies wird in Abbildung 3.2 veranschaulicht.

Abbildung 3.3 zeigt alle in FA8 erläuterten Interaktionen einer Version, die der Benutzer durchführen kann. Er kann eine Version kaufen, eine bereits gekaufte Version herunterladen, eine Version abonnieren, ein Abonnement beenden und bei einer gekauften Version prüfen, ob sich das Paket geändert hat, also ein Update zur Verfügung steht.

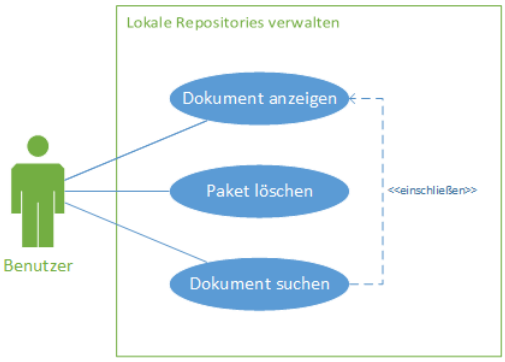

Abbildung 3.2: Anwendungsfälle der lokalen Repositories

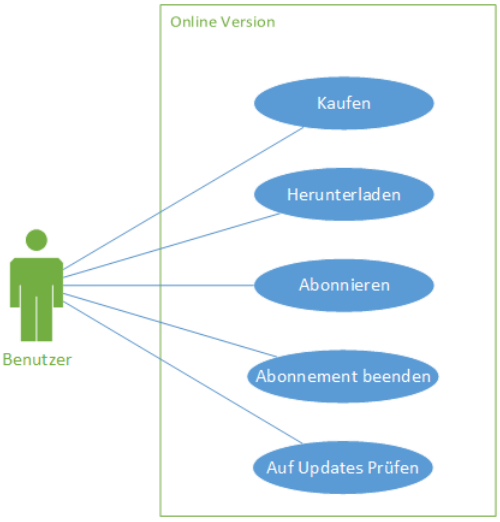

Abbildung 3.3: Anwendungsfälle einer online Version

3.3.2 Flussdiagramme

Im Folgenden werden alle Anwendungsfälle in ihre einzelnen Interaktionen aufgeteilt und deren Zusammenhänge mit Hilfe von Flussdiagrammen veranschaulicht.

Login

Abbildung 3.4 zeigt den gesamten Ablauf des Logins. Dafür muss der Benutzer zunächst alle Logindaten eingeben. Dazu gehören eine E-Mail Adresse, bzw. ein Benutzername, das zugehörige Passwort und die URL des Servers, an dem sich die Anwendung einloggen soll. Durch einen Tap auf den *Login*-Button wird der Prozess gestartet und es wird versucht die Daten an der Server zu senden.

Wenn eine Verbindung mit dem Internet vorhanden ist, werden die Daten an den Server übermittelt, der anschließend die Informationen überprüft. In seiner Antwort teilt er der Anwendung mit, ob der Login erfolgreich war. Ist dies der Fall werden die eingegebenen Daten lokal gespeichert, damit sich Benutzer künftig auch ohne Internetverbindung einloggen kann. Lehnt der Server den Login ab kann der Benutzer seine Eingaben ändern und es erneut versuchen.

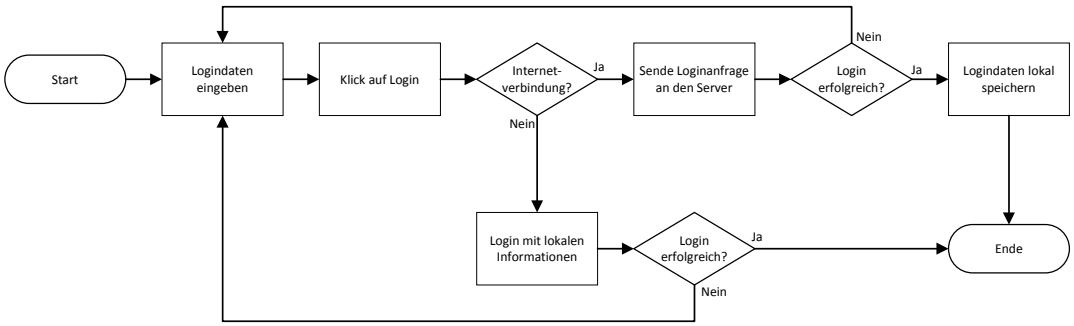

Abbildung 3.4: Anwendungsfall: Login

Sollte keine Internetverbindung bestehen, werden automatisch alle lokal gespeicherten Benutzerinformationen mit den eingegeben Daten verglichen. Falls die eingegebenen Informationen mit den lokalen Daten übereinstimmen, hat sich der Benutzer erfolgreich eingeloggt. Ist dies nicht der Fall beginnt der Ablauf von vorne und der Benutzer kann den Login mit veränderten Daten erneut versuchen.

Für alle folgenden Anwendungsszenarien wird vorausgesetzt, dass sich der Benutzer bereits erfolgreich eingeloggt hat.

Dokument anzeigen

Um sich ein Dokument anzeigen zu lassen kann der Benutzer entweder selbst in den lokalen Repositories navigieren und dort das gewünschte Dokument auswählen oder er kann danach suchen lassen und es in den Suchergebnissen öffnen. Bei der Suche hat der Benutzer die Möglichkeit in allen Repositories suchen zu lassen oder zuvor eine Version zu öffnen, um nur in dieser Version zu suchen. Abbildung 3.5 zeigt die möglichen Vorgehensweisen, wie der Benutzer sich ein Dokument anzeigen lassen kann.

Navigiert der Benutzer selbst zum gewünschten Dokument, öffnet er zuerst die Anzeige der lokalen Repositories. Dort werden ihm alle seine Repositories mit den zugehörigen Versionen angezeigt, die auf dem Gerät gespeichert sind. Wählt der Benutzer eine Version aus, wird diese geöffnet und er kann durch den Inhalt navigieren. Sobald das gewünschte Dokument gefunden wurde, wählt der Benutzer dieses aus und es wird in einer eigenen Ansicht dargestellt.

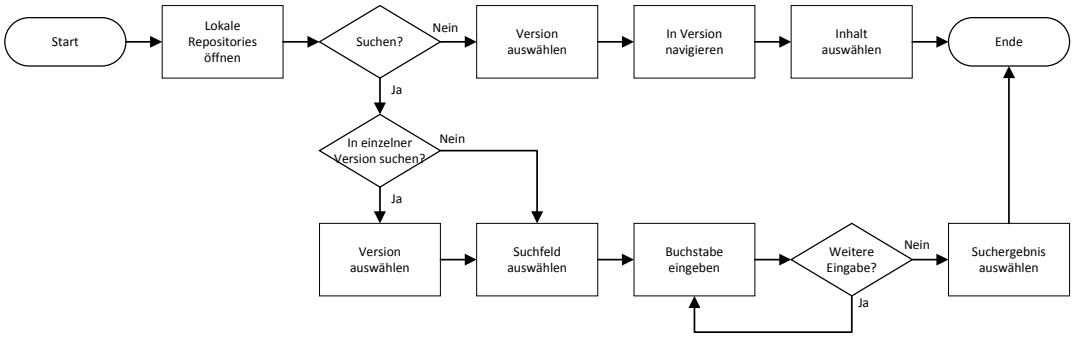

Abbildung 3.5: Anwendungsfall: Dokument anzeigen

Um eine Suche über alle Repositories zu starten, muss der Benutzer zuerst die Anzeige der lokalen Repositories öffnen. Dort befindet ein Suchfeld, das ausgewählt werden kann, um den Suchmodus zu aktivieren. Bei jeder einzelnen Eingabe werden die Suchergebnisse automatisch aktualisiert. Wenn das gesuchte Dokument in der Liste der Ergebnisse auftaucht, kann der Benutzer dieses Ergebnis auswählen, um direkt das referenzierte Dokument zu öffnen.

Die Suche in einer einzelnen Version läuft fast analog zur Suche über alle Repositories ab. Der Benutzer öffnet zunächst die Anzeige der lokalen Repositories. Dort wählt er dann, im Gegensatz zur vorigen Suche, die Version aus, in der er suchen möchte. Innerhalb der Navigation in der Version befindet sich ebenfalls ein Suchfeld, das ausgewählt werden kann, um den Suchmodus zu aktivieren. Bei einer Eingabe wird dann nur in der ausgewählten Version gesucht.

Online Funktionen

Alle Abläufe in diesem Paragraph benötigen einerseits eine Internetverbindung und andererseits muss sich der Benutzer zuvor am Server eingeloggt haben. Hat er sich offline eingeloggt, stehen diese Funktionen nicht zur Verfügung.

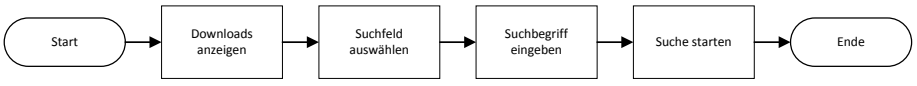

Abbildung 3.6: Anwendungsfall: Filtern der online Repositories

Wie es in Abbildung 3.6 dargestellt ist, kann der Benutzer die angezeigten Repositories im Downloadbereich filtern, indem er diesen zunächst öffnet und dort das Suchfeld auswählt. Dann muss er seinen gesamten Suchbegriff eingeben und auf *Suchen* tippen. Die Anwendung sendet daraufhin eine Anfrage mit dem Suchbegriff an den Server, der eine gefilterte List zurück sendet, die direkt angezeigt wird.

Alle in Abbildung 3.3 dargestellten Anwendungsfälle folgen dem in Abbildung 3.7 dargestellten Ablauf. Er ist beispielhaft für den Kauf einer online Version angepasst.

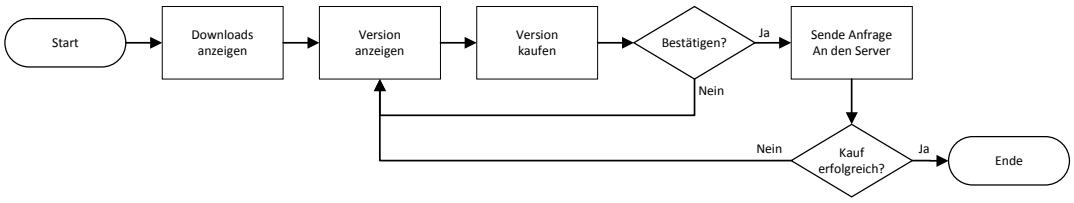

Abbildung 3.7: Anwendungsfall: online Version kaufen

Um eine Version zu kaufen muss der Benutzer den Downloadbereich öffnen, in dem alle online verfügbaren Repositories und die zugehörigen Versionen angezeigt werden. Dort kann der Benutzer die Version auswählen, die er kaufen möchte. Dadurch wird eine Detailansicht der Version geöffnet, in der die wichtigsten Informationen über diese Version dargestellt sind. Hier hat der Benutzer nun die Möglichkeit die Version zu kaufen. Wenn er auf den *Kaufen*-Button tippt, wird ein Popup-Fenster angezeigt, in dem der Kauft nochmals bestätigt werden muss. Wird dies vom Benutzer bestätigt, sendet die Anwendung eine Anfrage an den Server, dass der Benutzer diese Version kaufen möchte. Hat der Benutzer zu wenig Guthaben, sendet der Server einer Fehlermeldung, die als Popup angezeigt wird. Bestätigt der Server den Kauf, wird dies dem Benutzer angezeigt und er hat anschließend die Möglichkeit die Version herunterzuladen.

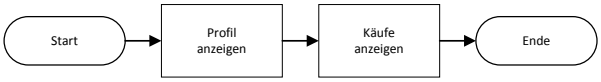

Abbildung 3.8: Anwendungsfall: gekaufte Versionen anzeigen

Alternativ zum Kauf kann der Benutzer in der Detailansicht der Version diese auch herunterladen, abonnieren oder ein Abonnement beenden. In jedem dieser Fälle sendet

die Anwendung zunächst eine Anfrage an den Server, die dann mit einer spezifischen Nachricht beantwortet wird.

Um sich die bereits gekauften Versionen anzeigen zu lassen, muss der Benutzer sein Profil öffnen. Dort hat er die Möglichkeit sich entweder seine gekauften, oder seine abonnierten Versionen anzeigen zu lassen. Dieser Ablauf ist in Abbildung 3.8 zu sehen.

Logout

Abbildung 3.9 zeigt, wie sich der Benutzer ausloggen kann. Dazu muss der Benutzer sein Profil öffnen und dort auf den *Logout*-Button tippen. Es wird ein Popup-Fenster angezeigt und der Benutzer muss sein Vorhaben bestätigen. Bricht er den Logout ab, wird nur das Popup-Fenster geschlossen. Bestätigt der Benutzer, wird er direkt ausgeloggt und der Login-Bildschirm wird angezeigt.

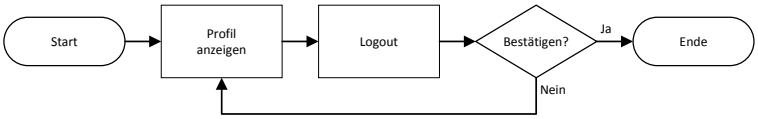

Abbildung 3.9: Anwendungsfall: Logout

Einstellungen ändern

Wenn der Benutzer die Einstellungen der Anwendung verändern möchte, muss er dem Ablauf in Abbildung 3.10 folgen. Zunächst muss er die Ansicht für die Einstellungen öffnen. Dort kann er die gewünschten Einstellungen vornehmen. Wurde alles angepasst ist der Ablauf fertig. Eine Bestätigung der Änderungen ist nicht nötig, da dies von der Anwendung automatisch durchgeführt wird.

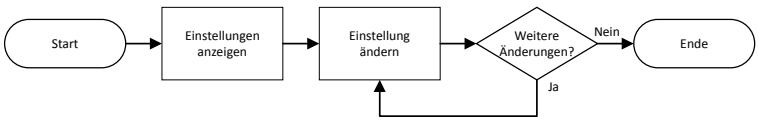

Abbildung 3.10: Anwendungsfall: Einstellungen ändern

3.3.3 Dialoggestaltung

Die erste Ansicht, die der Benutzer zu sehen bekommt, wenn er die Anwendung öffnet, ist der *Login*-Bildschirm, der in Abbildung 3.11 zu sehen ist. Dort kann er seine Benutzerdaten eingeben, also den Benutzername zusammen mit dem Passwort, und die URL, die zum Server führen sollte. Tippt der Benutzer auf den *Login*-Button, versucht sich die Anwendung, wie in Abschnitt 3.3.2 beschrieben, einzuloggen. Sobald der Login erfolgreich ist, wird das Hauptmenü angezeigt, das in Abbildung 3.12 skizziert ist. Von dort hat der Benutzer Zugang zu seinen lokal gespeicherten Repositories, zum Downloadbereich, zu den Einstellungen und zu seinem Profil.

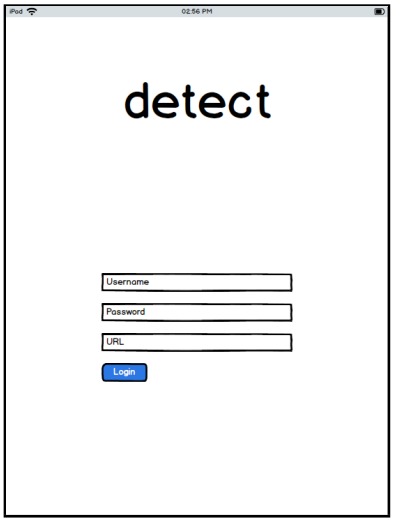

Abbildung 3.11: Mockup: Login

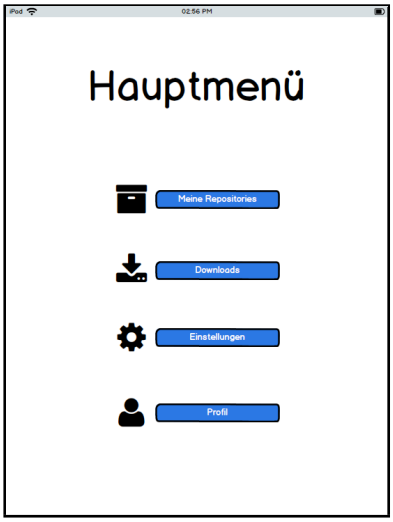

Abbildung 3.12: Mockup: Hauptmenü

Tippt der Benutzer im Hauptmenü auf den *Meine Repositories*-Button gelangt er zur Ansicht, die in Abbildung 3.13 zu sehen ist. Es werden alle lokal gespeicherten Repositories des Benutzers zusammen mit einer Beschreibung dargestellt. Tippt der Benutzer auf den *Zurück*-Button, wird ihm wieder das Hauptmenü angezeigt. Tippt er in das Suchfeld wird die Suche, wie in Abschnitt 3.3.2 erläutert, gestartet. Wählt der Benutzer ein Repository aus, werden ihm die zugehörigen Versionen, zusammen mit einem Titelbild und einer kurzen Beschreibung, nach Abbildung 3.14, angezeigt. Der Button oben links führt den Benutzer zurück zur Listenansicht der Repositories.

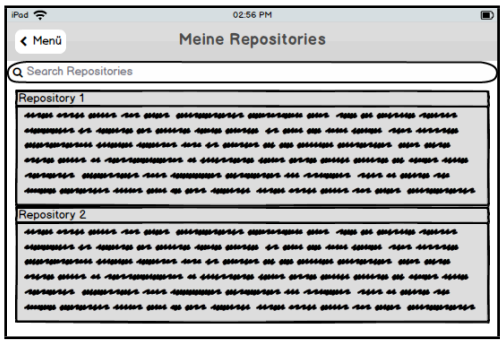

Abbildung 3.13: Mockup: Meine Repositories

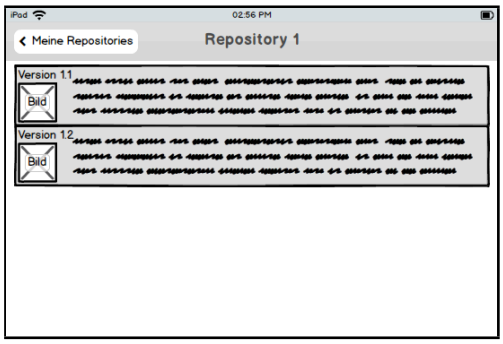

Abbildung 3.14: Mockup: Versionen eines Repositories

Wählt der Benutzer eine Version aus, werden deren Inhalte geöffnet und angezeigt. In der Detailansicht einer Version ist der Bildschirm in zwei Teile aufgeteilt, wie es in Abbildung 3.15 zu sehen ist. Auf der linken Seite befindet sich die Navigation des Pakets, während auf der rechten Seite das aktuell geöffnete Dokument angezeigt wird. In der oberen Leiste gibt es einen *Zurück*-Button, der zur Listenansicht der Versionen führt. Darunter befindet sich ein Suchfeld, mit dem die geöffnete Version durchsucht werden kann. Unter dem Suchfeld befindet sich die Navigation des Pakets, in der die Inhalte des Pakets aufgelistet sind. Wenn der Benutzer einen Inhalt auswählt, wird je nach dem, ob es sich um eine Datei oder einen Ordner handelt, entweder das Dokument geöffnet und auf der rechten Seite angezeigt, oder es wird der Ordner geöffnet und die darin enthaltenen Inhalte auf der linken Seite aufgelistet.
Wenn der Benutzer das Suchfeld auswählt, wird die virtuelle Tastatur angezeigt und die Suche gestartet. Bei jedem eingegebenen Buchstabe werden die Suchergebnisse aktualisiert und in einer Liste angezeigt. Wenn der Benutzer ein Ergebnis auswählt, wird das darin verlinkte Dokument geöffnet und auf der rechten Seite angezeigt.

Tippt der Benutzer im Hauptmenü auf den *Downloads*-Button wird die Ansicht mit den online Inhalten geöffnet. Dafür sendet die Anwendung eine Anfrage an den Server, um Informationen über alle online verfügbaren Repositories und den zugehörigen Versionen zu bekommen. Auch hier befindet sich oben links der *Zurück*-Button, welcher den Benutzer ins Hauptmenü führt. Auf der rechten Seite befindet sich ein *Aktualisieren*-Button, welcher beim Betätigen eine Anfrage an den Server sendet, um erneut die online

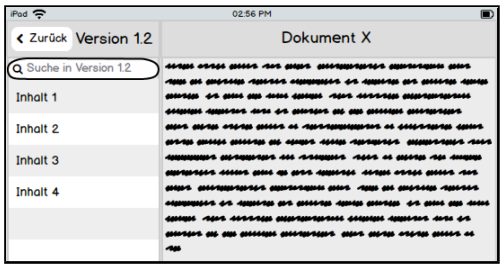

Abbildung 3.15: Mockup: Version

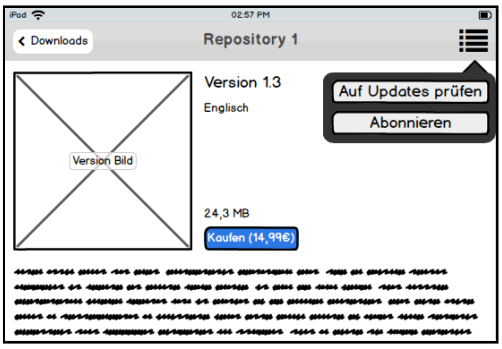

Abbildung 3.16: Mockup: Online Version

verfügbaren Inhalte zu laden. Darunter ist die Suchleiste, die die Suche nach Abbildung 3.6 durchführt. Darunter werden die online Repositories und Versionen, ähnlich zu Abbildung 3.14 aufgelistet.

Wenn der Benutzer eine Version auswählt, wird eine Detailansicht nach Abbildung 3.16 geöffnet. Darin werden alle Informationen der Version dargestellt und der Benutzer hat die Möglichkeit diese Version zu kaufen, herunterzuladen, abonnieren oder auf Updates zu prüfen. Falls er die Version bereits abonniert hat, kann er das Abonnement hier auch wieder beenden.

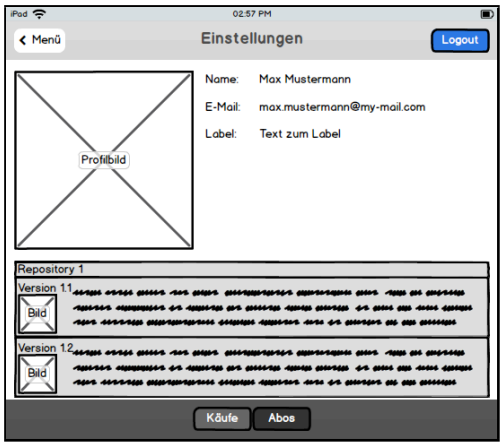

Abbildung 3.17: Mockup: Profil

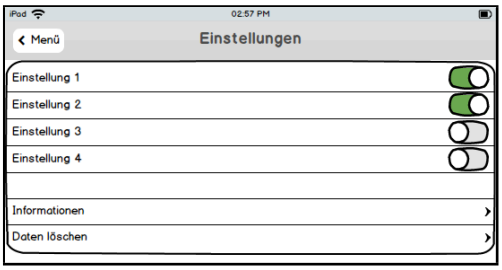

Abbildung 3.18: Mockup: Einstellungen

Tippt der Benutzer im Hauptmenü auf den *Profil*-Button, öffnet sich die entsprechende Ansicht. Oben links befindet sich der *Zurück*-Button, der ins Hauptmenü führt. Auf der

rechten Seite ist der *Logout*-Button, mit dem sich der Benutzer von der Anwendung abmelden kann. Wenn er auf diesen Button tippt, wird ein Popup-Fenster angezeigt und der Benutzer muss sein Vorhaben bestätigen. Bestätigt der Benutzer den Logout, wird ihm der Login-Bildschirm angezeigt. Unter diesen beiden Buttons werden alle Benutzerinformationen in einer Liste angezeigt, wie es in Abbildung 3.17 abgebildet ist. Darunter werden entweder alle gekauften Versionen des Benutzers dargestellt, oder die Versionen, die er abonniert hat. Welches von Beiden angezeigt wird, kann in der unteren Leiste ausgewählt werden.
Wenn der Benutzer im Hauptmenü auf den *Einstellungen*-Button tippt, wird ihm die Ansicht in Abbildung 3.18 mit den Einstellungen der Anwendung angezeigt. Dort kann er die von ihm gewünschten Änderungen durchführen, welche nach jeder Änderung automatisch gespeichert werden. Hier kann der Benutzer auch seine Daten löschen, die für den offline Login auf dem Gerät gespeichert werden. Wählt er die entsprechende Einstellung aus, wird ihm zuvor ein Popup-Fenster angezeigt, in dem er den Löschvorgang bestätigen muss. Bestätigt der Benutzer den Vorgang, wird das Popup-Fenster geschlossen und die Daten werden gelöscht. Bricht er den Vorgang ab, wird das Fenster ohne weitere Aktionen geschlossen. Über den *Zurück*-Button kann der Benutzer wieder zum Hauptmenü navigieren.

Nachdem in diesem Kapitel alle Anforderungen der mobilen Anwendung analysiert und Konzepte zur Funktionsweise und zum Design vorgestellt wurden, wird im nächsten Kapitel die Umsetzung der Anwendung dokumentiert.

4

Implementierung

In diesem Kapitel wird die Umsetzung der Anwendung auf der iOS Plattform dokumentiert. Zuerst werden die Navigation innerhalb der Anwendung sowie die Gestaltung der einzelnen Ansichten erläutert und mit den ursprünglichen Ideen aus Abschnitt 3.3.3 abgeglichen. Anschließend wird die Architektur mit Hilfe eines Klassendiagramms veranschaulicht. Zuletzt werden die wichtigsten Funktionen, die zur Laufzeit verwendet werden, aufgelistet und im Detail beschrieben.

4.1 Dialogstruktur

Dieser Abschnitt befasst damit, wie jeder einzelne Dialog erreicht werden kann, also mit der Navigation innerhalb der Anwendung.

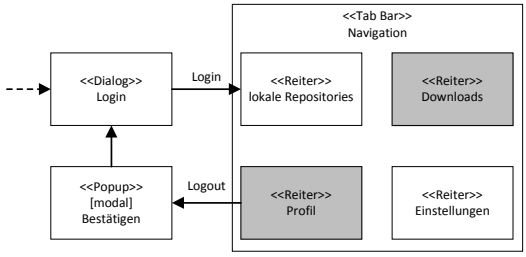

Abbildung 4.1: Dialogstruktur: Anwendung

Sobald die Anwendung gestartet wird, bekommt der Benutzer den *Login*-Dialog zu sehen. Nach erfolgreichem Login wird dem Benutzer direkt der Dialog angezeigt, in dem die lokalen Repositories aufgelistet werden. Das Hauptmenü aus Abbildung 3.12 wird dabei übersprungen, um alle Dialoge in so wenigen Schritten wie möglich erreichen zu können. Das Hauptmenü wurde durch eine Tab Bar ersetzt, die es ermöglicht jederzeit zwischen den Dialogen *lokale Repositories*, *Downloads*, *Profil* und *Einstellungen* wechseln zu können. Befindet sich der Benutzer im *Profil*-Dialog, kann er sich von der Anwendung abmelden. Nach der Bestätigung eines Popups kommt der Benutzer zurück zum Login-Dialog. Diese Navigationsmöglichkeiten werden in Abbildung 4.1 veranschaulicht, wobei die grau hinterlegten Dialoge in einem weiteren Diagramm jeweils noch genauer spezifiziert werden.

Abbildung 4.2: Dialogstruktur: Downloads

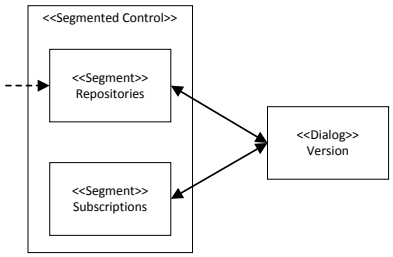

Abbildung 4.3: Dialogstruktur: Profil

Abbildung 4.2 zeigt die Struktur des *Download*-Dialogs. Sobald dieser geöffnet wird, werden alle online Repositories mit den zugehörigen Versionen angezeigt. Dabei wurden die Abbildungen 3.13 und 3.14 aus dem zuvor bereits genannten Grund zusammengefasst. Wenn der Benutzer eine Version auswählt, wird ihm eine Detailansicht mit allen Informationen dieser Version angezeigt. Über einen Button wird ein Popup angezeigt, in

dem einige Funktionen der Version ausgeführt werden können. Dies sind unter Anderem das Abonnieren der Version oder das Abonnement der Version zu beenden. Wird eine Funktion ausgeführt oder das Popup geschlossen, kommt der Benutzer zurück zum Version-Dialog.

Abbildung 4.3 zeigt die Struktur des Profil-Dialogs. Zunächst werden dort alle Repositories und Versionen angezeigt, die der Benutzer gekauft hat. Über ein Segmented Control kann jederzeit zwischen gekauften Repositories und Versionen, und den abonnierten Repositories und Versionen gewechselt werden. Wählt der Benutzer eine Version aus, wird ihm dieselbe Detailansicht der Version, wie im Downloadbereich, angezeigt.

4.2 Dialoggestaltung

In diesem Abschnitt wird die tatsächliche Gestaltung der einzelnen Dialoge vorgestellt. Dabei werden zum einen die Interaktionsmöglichkeiten erläutert, und zum anderen direkte Vergleiche mit den Mockups aus Abschnitt 3.3.3 angestellt. Die Screenshots wurden jeweils auf die wesentlichen Inhalte reduziert.

Abbildung 4.4: Dialoggestaltung: Login

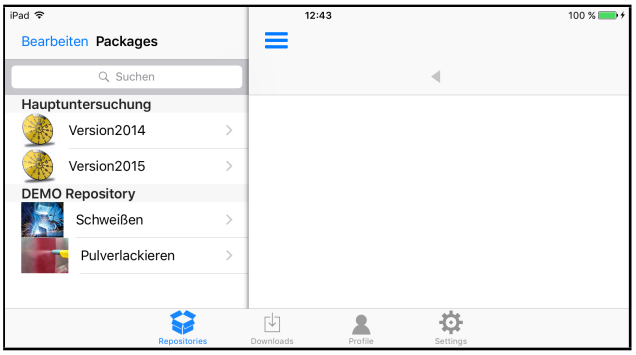

Abbildung 4.5: Dialoggestaltung: Lokale Repositories

Der Login-Dialog in Abbildung 4.4 wird dem Benutzer direkt nach dem Start der Anwendung angezeigt. Analog zu Abbildung 3.11 gibt es drei Textfelder zur Eingabe der Logindaten und den *Login*-Button.

Tippt der Benutzer auf den *Login*-Button und wird erfolgreich eingeloggt, wird ihm direkt der Dialog mit den lokalen Repositories angezeigt. Im Gegensatz zu Abbildung 3.12 gibt es kein Hauptmenü. Die einzelnen Bereiche können über einen Tap auf das entsprechende Symbol in der Tab Bar am unteren Bereich der Dialoge erreicht werden. In Abbildung 4.5 ist zu sehen, dass der Dialog mit den lokalen Repositories in zwei Teile aufgeteilt ist. Auf der linken Seite befindet sich die Liste mit allen lokalen Repositories und den zugehörigen Versionen. Gleichzeitig befindet sich dort das Suchfeld, mit dem der Benutzer Inhalte in allen Repositories suchen kann. Wählt der Benutzer eine Version aus, wird diese Liste mit den Inhalten der Version ersetzt und der Benutzer kann durch das Paket navigieren. Außerdem wird im Zuge dessen die Suche auf die ausgewählte Version beschränkt. Sobald der Benutzer einen Inhalt öffnet, entweder über die direkte Navigation, oder über die Suche, wird dieser auf der rechten Seite des Dialogs angezeigt. Mit dem Button links oben kann die Ansicht auf der linken Seite ein- oder ausgeblendet werden, siehe Abbildung 4.6.

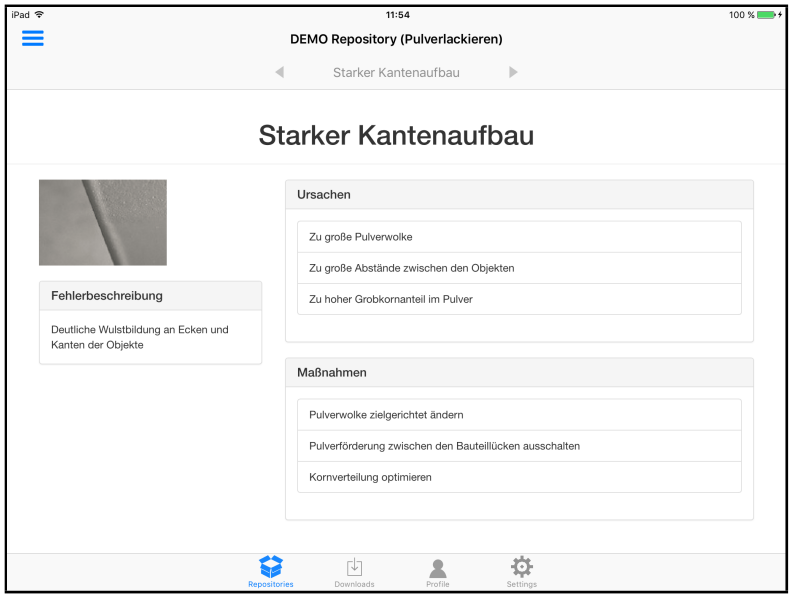

Abbildung 4.6: Dialoggestaltung: Geöffneter Inhalt

Durch diesen Aufbau werden die Mockups in Abbildungen 3.13, 3.14 und 3.15 in einem einzelnen Dialog zusammengefasst, um die Anwendung in diesem Bereich einfacher und übersichtlicher zu gestalten.

Tippt der Benutzer auf das *Download*-Symbol in der Tab Bar am unteren Rand, wird ihm der Dialog in Abbildung 4.7 angezeigt. Am oberen Rand befindet sich die Suchleiste, mit der die online Repositories durchsucht werden können, und ein Button, mit dem die Liste der online Repositories und Versionen aktualisiert werden kann. Darunter werden zunächst nur alle Repositories in einer Liste angezeigt. Sobald der Benutzer eines davon auswählt, expandiert sich die Liste und alle Versionen, die zum ausgewählten Repository gehören, werden sichtbar. Sobald der Benutzer eine Version auswählt, gelangt er zur Detailansicht der Version in Abbildung 4.8. Dort werden alle zugehörigen Informationen angezeigt und der Benutzer hat die Möglichkeit diese Version zu kaufen, bzw. herunterzuladen. Rechts oben befindet sich ein Button, der die weiteren Interaktionsmöglichkeiten mit der Version in einem Popup anzeigt. Damit wurde das Konzept aus Mockup 3.15 komplett umgesetzt.

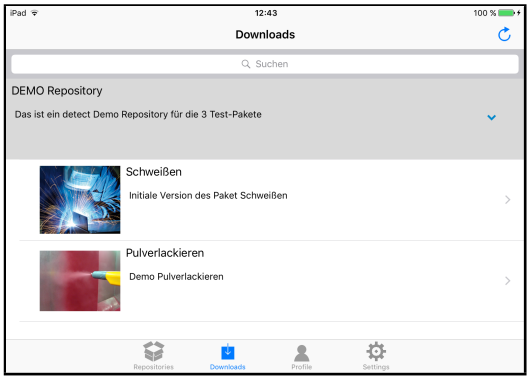

Abbildung 4.7: Dialoggestaltung: Downloads

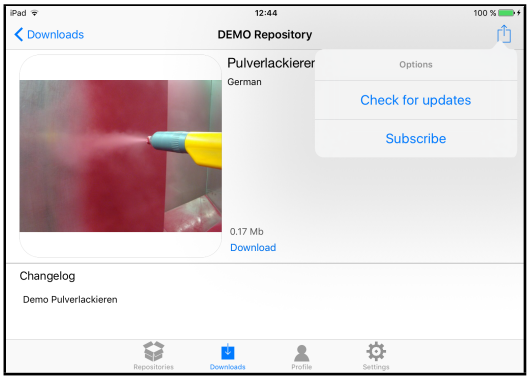

Abbildung 4.8: Dialoggestaltung: Version

Tippt der Benutzer auf das *Profile*-Symbol in der Tab Bar, wird ihm der Dialog in Abbildung 4.9 angezeigt. Rechts oben befindet sich der *Logout*-Button, mit dem sich der Benutzer von der Anwendung abmelden kann und zurück zum Login-Dialog kommt. Darunter befindet sich das Segmented Control, mit dem der Benutzer die angezeigten Daten im unteren Bereich des Dialogs auswählen kann. Standardmäßig werden dort

die gekauften Repositories und Versionen des Benutzers angezeigt. Wird die andere Option ausgewählt, werden die abonnierten Repositories und Versionen des Benutzers angezeigt. Unter dem Segmented Control sind das Profilbild und Informationen des Benutzers zu sehen. Im unteren Bereich werden, analog zu Dialog 4.7, zunächst nur die Repositories aufgelistet. Wählt der Benutzer ein Repository aus, wird die Anzeige um die zugehörigen Versionen erweitert. Wenn der Benutzer anschließend eine Version auswählt, wird ihm die Detailansicht der Version aus Abbildung 4.8 angezeigt.

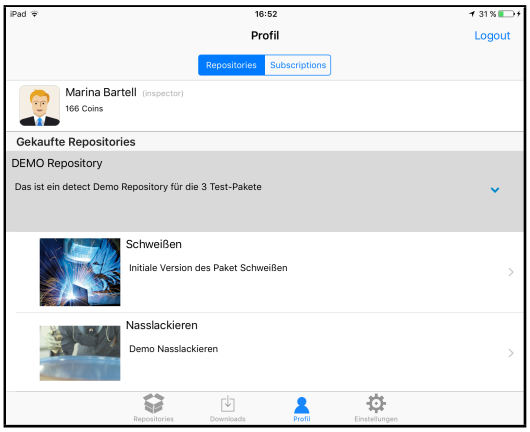

Abbildung 4.9: Dialoggestaltung: Profil

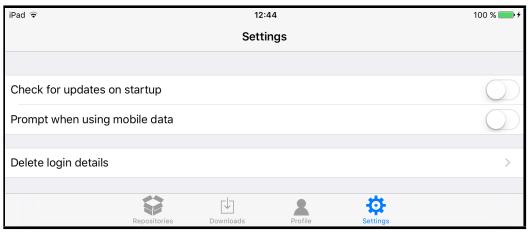

Abbildung 4.10: Dialoggestaltung: Einstellungen

Der Dialog mit den Einstellungen der Anwendung ist über das *Einstellungen*-Symbol in der Tab Bar erreichbar. Wie es in Abbildung 4.10 zu sehen ist, werden dort alle Einstellungsmöglichkeiten, analog zum Mockup 3.18, untereinander aufgelistet.

4.3 Architektur

In diesem Abschnitt wird die Architektur der Anwendung mit Hilfe von Klassendiagrammen erläutert. Es werden jeweils die Bedeutung der einzelnen Klassen und die Relationen zwischen den Klassen betrachtet. Der Aufbau der Anwendung lässt sich dabei in zwei Teile aufteilen: Zum einen das Datenmodell, mit dem alle Informationen zur Laufzeit gehalten werden, und zum anderen das User-Interface-Modell, mit dessen Hilfe die Dialoge gestalten werden.

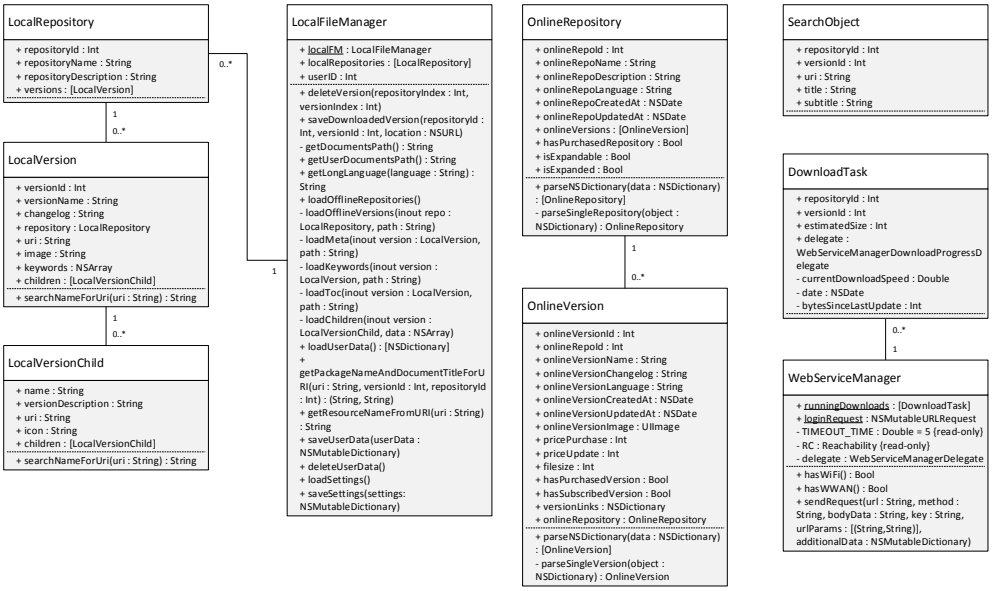

Abbildung 4.11: Klassendiagramm: Datenmodell

Abbildung 4.11 zeigt das Datenmodell. Die wichtigste Klasse hierbei ist der *LocalFileManager*, da er alle Lese- und Schreiboperationen auf der Festplatte des Gerätes übernimmt. Innerhalb der Klasse wird beim Starten der Anwendung eine statische Instanz erzeugt, auf die zur Laufzeit von der gesamten Anwendung aus zugegriffen werden kann. Sobald sich ein User eingeloggt hat, wird die Variable *userID* gesetzt, um die lokalen Daten zuordnen zu können. Gleichzeitig werden die lokalen Repositories geladen und in die Liste *localRepositories* eingetragen. Dabei repräsentiert eine Instanz der Klasse *LocalRepository* ein Repository. Analog verhält es sich bei der Klasse *LocalVersion*, wobei die rekursive Klasse *LocalVersionChild* dazu verwendet wird die hierarchische Navigation innerhalb der Version abzubilden. Sie werden ausschließlich dafür benutzt lokale Daten zu verwalten und kommen nur im Bereich *Repositories* aus Abbildung 4.5 zum Einsatz. An dieser Stelle wird ebenfalls die Klasse *SearchObject* verwendet, um die notwendigen Informationen während der Suche zusammenfassen und anzeigen zu können.

Im Gegensatz zu den *Local*-Klassen werden die Klassen *OnlineRepository* und *OnlineVersion* ausschließlich dafür verwendet die entsprechenden Daten vom Server zu halten. Beide Klassen besitzen jeweils eine Funktion, die ein JSON-Objekt vom Server

in die eigene Klasse umwandelt. Sie kommen im Downloads- und Profilbereich aus den Abbildungen 4.7, 4.8 und 4.9 zum Einsatz. Um diese Daten vom Server zu erhalten werden HTTP-Nachrichten mit der Klasse *WebServiceManager* versendet und empfangen. Die Funktion *sendRequest* erzeugt dabei eine HTTP-Anfrage und sendet diese an den Server. Die Funktion beinhaltet eine Callback-Methode, die aufgerufen wird, sobald eine Antwort von Server zurückkommt. Dabei wird die Antwort überprüft und an die Variable *delegate* weitergereicht. In dieser Variable wird eine Klasse referenziert, die die Antwort des Servers verarbeiten kann. Soll ein Paket heruntergeladen werden, wird dafür die Klasse *DownloadTask* verwendet. Sie informiert ihre *delegate* Variable in kurzen Abständen über den Zustand des Downloads. Dabei werden beispielsweise Informationen zum Fortschritt des Downloads und die aktuelle Downloadgeschwindigkeit weitergereicht. Sobald der Download abgeschlossen ist wird die Funktion *saveDownloadedVersion* des *LocalFileManagers* aufgerufen um das Paket zu verarbeiten.

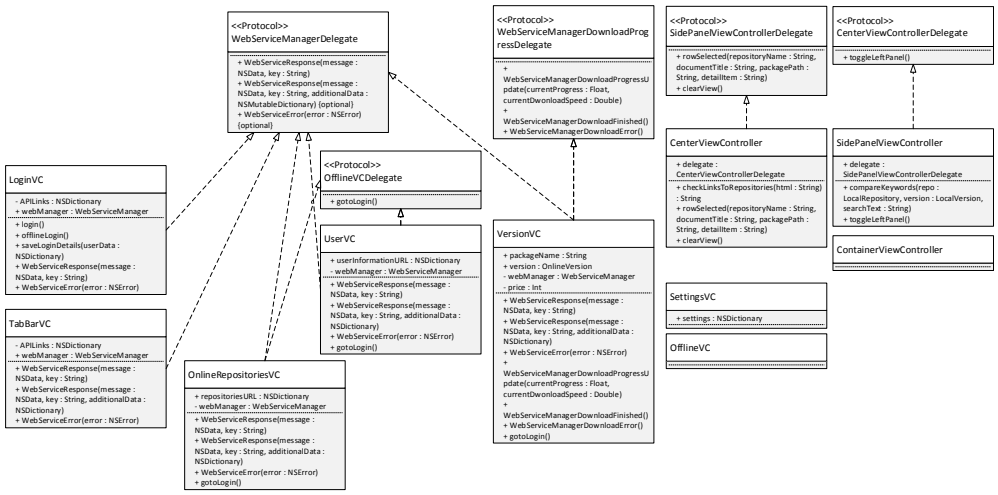

Abbildung 4.12: Klassendiagramm: User-Interface-Modell

Die Navigation in der Anwendung wird zusammen mit dem User Interface von den Klassen aus Abbildung 4.12 umgesetzt. Für eine übersichtlichere Darstellung in einem Diagramm wurden alle Variablen und Funktionen, die ausschließlich für das User Interface benötigt werden, weggelassen. Jeder Dialog, der eine Verbindung mit dem Server benötigt, um Informationen abzufragen, implementiert das *WebServiceManager-Delegate* Protokoll. Dies ermöglicht es der *WebServiceManager*-Klasse die Antworten

des Servers an den entsprechenden Dialog zu übergeben. Die Klasse *VersionVC*, die dazu verwendet wird eine online Version anzuzeigen, implementiert zusätzlich noch das *WebServiceManagerDownloadProgressDelegate* Protokoll, mit dessen Hilfe spezielle Informationen eines Downloads übergeben und angezeigt werden können.
Für den Dialog 4.5 der lokalen Repositories werden insgesamt drei Klassen verwendet. Der *SidePanelViewController* wird zur Anzeige der Navigation in den Repositories und Versionen, und zur Suche in den Repositories und Versionen verwendet. Mit Hilfe des *SidePanelViewControllerDelegate* Protokolls kann dem *CenterViewController* mitgeteilt werden, dass ein neuer Inhalt angezeigt werden soll. Der *CenterViewController* zeigt die Inhalte an und kann mit Hilfe des *CenterViewControllerDelegate* Protokolls den *SidePanelViewController* anzeigen oder ausblenden lassen. Die Klasse *ContainerViewController* sorgt dafür, dass der *SidePanelViewController* zusammen mit dem *CenterViewControllerDelegate* auf dem selben Dialog angezeigt wird.

4.4 Funktionen

Im Folgenden sind die wichtigsten Funktionen dokumentiert, die zur Laufzeit verwendet werden. Der Name der Funktionen ist in Pseudocode angegeben, zusammen mit einer Beschreibung was die Funktion macht und welche Parameter benötigt werden.

F1 Login: Dieser Funktion werden beim Aufruf drei Parameter übergeben: Eine E-Mail, ein Passwort und eine URL. Die Anwendung versucht sich an der eingegebenen URL mit den Benutzerdaten einzuloggen. Falls eine Internetverbindung besteht, wird ein HTTP-Request an die URL gesendet. Falls die Antwort konform zur Definition der API ist, werden die E-Mail Adresse und das Passwort an die in der Antwort enthaltenen URL für den Login gesendet. Wird der Login vom Server bestätigt, werden zusätzlich noch die Profilinformationen des Benutzers abgefragt. Falls keine Internetverbindung besteht werden die lokal gespeicherten Profile durchsucht. Sobald ein Eintrag passend zur E-Mail Adresse gefunden wird, werden die Passwörter verglichen und loggen den Benutzer gegebenenfalls offline ein. Wenn die online oder offline Anmeldung erfolgreich ist, wird die *userId* des Benutzers

gespeichert, um eine Zuordnung zu den lokalen Paketen zu ermöglichen. Bei einer erfolgreichen online Anmeldung werden zusätzlich noch die E-Mail Adresse und das mit dem bcrypt-Algorithmus gehashte Passwort gespeichert, um die offline Anmeldung zu ermöglichen. Falls der Benutzer in den Einstellungen die Option aktiviert hat, dass die Anwendung beim Start alle lokalen Pakete auf Updates überprüfen soll, wird dies direkt nach dem Login durchgeführt. Mit dieser Funktion werden die Anforderungen FA1 und FA2 und ein Teil von Anforderung FA4 umgesetzt.

F2 Lade Einstellungen: Direkt nach einem erfolgreichen Login werden die Einstellungen des jeweiligen Benutzers mit Hilfe der *userId*, die beim Aufruf übergeben werden muss, geladen und auf die Anwendung angewandt. Wenn sich der Benutzer zum ersten Mal einloggt, wird in seinem Verzeichnis eine Datei mit den Standardeinstellungen erstellt. Dadurch wird ein weiterer Teil von Anforderung FA4 umgesetzt.

F3 Lade offline Repositories: Nach den Einstellungen werden alle Repositories des Benutzers geladen. Dafür wird zunächst der Ordner mit der entsprechenden *userId* durchsucht. Für jedes gefundene Repository werden die zugehörigen Versionen durchsucht. Dabei werden die toc.json, die keywords.json und die meta.json geöffnet, überprüft und verarbeitet. Werden alle Vorgaben aus Abschnitt 2.4 erfüllt, werden für die Repositories und Versionen entsprechende Objekte erzeugt. Diese Funktion erfüllt einen Teil von Anforderung FA10, wobei das Anzeigen der Daten im User Interface implementiert ist. Außerdem wird zusammen mit den Funktionen F1 und F2 Anforderung FA3 umgesetzt.

F4 Suche und ersetze Links in Inhalt: Diese Funktion wird jedes mal Aufgerufen, wenn ein Inhalt angezeigt werden soll. Dafür wird der Funktion ein HTML Dokument übergeben, welches auf Platzhalter durchsucht wird, die Inhalte aus anderen Paketen referenzieren. Für jeden Platzhalter wird jeweils überprüft ob die referenzierten Inhalte lokal verfügbar sind, also bereits heruntergeladen wurden. Ist dies der Fall, wird an der entsprechenden Stelle ein Link eingefügt. Damit wird der restliche Teil von Anforderung FA10 umgesetzt.

F5 Vergleiche Schlüsselworte: Nach jedem Buchstaben, den der Benutzer in das Suchfeld eingibt, wird diese Funktion mit dem Suchbegriff als Parameter aufgerufen. Dabei wird je nach Kontext in allen Repositories oder nur in einer Version gesucht. Für jeden Treffer wird ein *SearchObject* erzeugt und einer Liste hinzugefügt, die dem Benutzer angezeigt wird. Diese Funktion implementiert Anforderung FA11.

F6 sende HTTP-Anfrage: Alle Anfragen an den Server, außer dem Download eines Pakets, werden über diese Funktion gesendet. Dabei wird eine HTTP-Anfrage erzeugt, die je nach übergebenen Parametern mit allen für den Server notwendigen Informationen angereichert wird. Sobald eine Antwort zurück kommt, wird diese zum Bearbeiten weitergereicht. Je nach Dialog, in dem sich der Benutzer befindet, wird eine andere Anfrage gesendet. Insgesamt werden mit Hilfe dieser Funktion die Anforderungen FA6, FA7, FA9, und ein Teil von Anforderung FA8 erfüllt.

F7 Lade Paket herunter: Diese Funktion wird aufgerufen, sobald bei einer online Version der *Download*-Button geklickt wird. Dabei werden eine *repositoryId* und eine *versionId* übergeben. Falls der Benutzer in den Einstellungen die Option aktiviert hat, dass die Anwendung eine Warnung anzeigt, wenn ein Paket über das mobile Datennetzwerk heruntergeladen werden soll, wird bei einer entsprechender Internetverbindung ein Popup angezeigt. Anschließend wird das mit den IDs referenzierte Paket heruntergeladen und im temporären Ordner der Anwendung zwischengespeichert. Danach wird das in einem Archiv verpackte Paket entpackt und in die Ordnerstruktur der Anwendung integriert. Zuletzt wird die Liste der lokalen Repositories aktualisiert, sodass die Inhalte des Pakets direkt angesehen werden können. Mit dieser Funktion werden die restlichen Teile der Anforderungen FA4 und FA8 implementiert.

F8 Lösche Paket: Sobald der Benutzer ein Paket löscht, wird diese Funktion mit einer *repositoryId* und einer *versionId* als Parameter aufgerufen. Dabei wird das entsprechende Paket aus der Ordnerstruktur entfernt. Falls es sich um das letzte Paket in einem Repository handelt, wird das Repository ebenfalls gelöscht. Diese Funktion setzt Anforderung FA5 um, wobei das Löschen der Benutzerdaten über die Einstellungen der Anwendung möglich ist.

F9 Logout: Diese Funktion meldet den Benutzer von der Anwendung ab und zeigt den *Login*-Dialog an. Ohne einen erneuten Login sind keine Daten mehr einsehbar, womit Anforderung FA12 umgesetzt wird.

Diese Implementierung erfüllt fast alle Anforderungen, die zu Beginn der Arbeit gestellt wurden. Im nächsten Kapitel werden bereits bestehende Anwendungen mit dieser Implementierung verglichen.

5

Verwandte Arbeiten

Da in dieser Arbeit eine Anwendung für iOS erstellt wird, werden zum Vergleich Anwendungen aus dem iTunes Store herangezogen, die ähnliche Absichten verfolgen. In diesem Kapitel werden zwei dieser Anwendungen vorgestellt und genauer betrachtet, die dem Funktionsumfang der hier erstellten Anwendung am ähnlichsten scheinen. Dafür wird zunächst der grobe Funktionsumfang beschrieben und anschließend mit der Implementierung der eigenen Anwendung verglichen.

5.1 NormenBibliothek

In diesem Abschnitt wird die NormenBibliothek [11] Anwendung des VDE Verlages analysiert. Das Ziel der NormenBibliothek ist es allen Kunden des VDE Verlages ihre Abonnements und gekauften Fachbücher komfortabel auf einem mobilen Endgerät zu

Verfügung zu stellen. Da für den Zugang zur Anwendung ein gültiges Abonnement der Normenbibliothek benötigt wird, wurden die Screenshots aus dem iTunes Store [12] übernommen. Um dennoch Vergleiche zur Funktionalität anstellen zu können, wird die Beschreibung der Anwendung von der Herstellerseite [11] herangezogen.

Sobald der Benutzer die Anwendung startet bekommt er einen Login-Dialog zu sehen, sodass nur autorisierten Benutzern Zugang zur Anwendung gewährt wird. Loggt sich der Benutzer erfolgreich ein, wird ein Grunddatenbestand auf das Gerät heruntergeladen, um die Inhalte auch offline zur Verfügung zu stellen, und es wird der *Home*-Dialog auf der linken Seite in Abbildung 5.1 angezeigt. Von dort ist der Grunddatenbestand, zusammen mit einigen anderen Informationen, abrufbar. Außerdem bietet die Anwendung die Möglichkeit bereits erworbene Fachbücher herunterzuladen und anzusehen. Der mittlere und rechte Dialog in Abbildung 5.1 zeigt jeweils eine Liste an Normen, bzw. Fachbüchern, die über den entsprechenden Listeneintrag im *Home*-Dialog erreichbar sind. Zusätzlich verspricht die Anwendung, dass bei der Herausgabe von neuen Normen der Datenbestand über Updates aktualisiert werden kann.

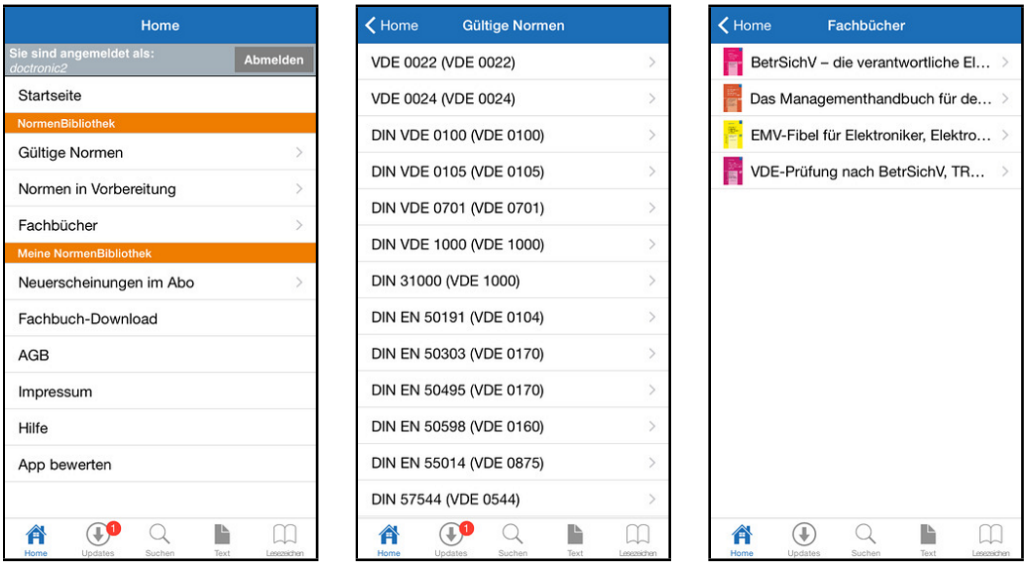

Abbildung 5.1: NormenBibliothek: Home und Inhalte

Über die TabBar am unteren Bereich der Anwendung kann ein *Suchen*-Dialog geöffnet werden, über den alle Inhalte schnell zu finden sind. Wie Abbildung 5.2 zeigt, stehen

dafür mehrere Möglichkeiten zur Verfügung. Neben einer Volltextsuche in allen Inhalten kann außerdem im Titel oder einem Stichwortverzeichnis gesucht werden. Alternativ kann der Benutzer nach einer bestimmten Norm bzw. Ausgabe suchen. Mit Hilfe von Lesezeichen können Inhalte markiert werden, um gegebenenfalls schneller wiedergefunden zu werden. An jedes Lesezeichen kann der Benutzer eigene Notizen anheften, um zusätzliche Anmerkungen zu speichern. Abbildung 5.3 zeigt den Dialog der Lesezeichen, zusammen mit einem geöffneten Inhalt.

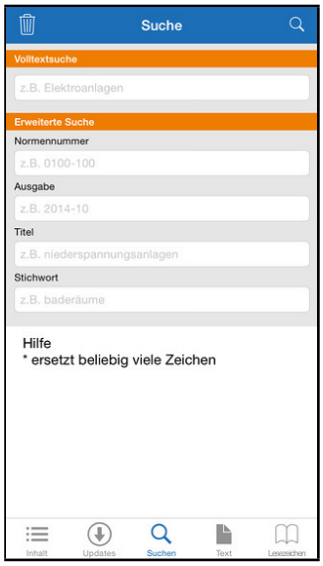

Abbildung 5.2: NormenBibliothek: Suche

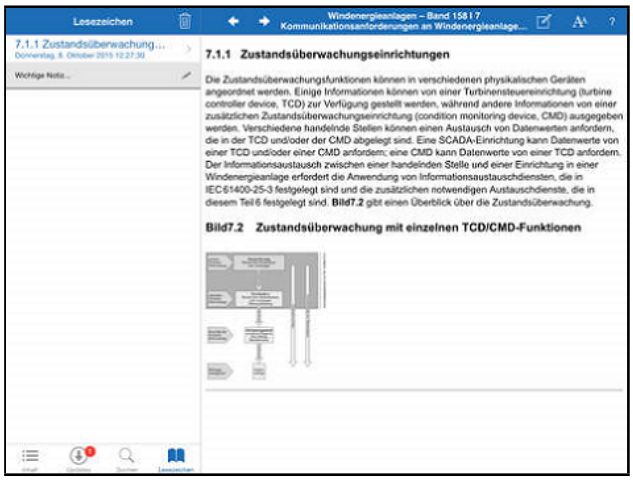

Abbildung 5.3: NormenBibliothek: Lesezeichen

Nach Angaben des Herstellers werden bei den Inhalten alle Verweise der DIN-VDE-Normen intern und extern verlinkt. Was dies genau bedeutet geht aus der Beschreibung nicht hervor, doch vermutlich können dadurch relevante Informationen durch das Öffnen eines Links direkt geöffnet und angesehen werden. Als letzte Funktion wird die Möglichkeit genannt eine Frage an Experten zu stellen. Falls also Unklarheiten bezüglich der Inhalte bestehen, können diese durch eine Kommunikation mit einem bereitgestellten Service beseitigt werden.

5.2 Flooright

Dieser Abschnitt befasst sich mit dem Funktionsumfang der Flooright [13] Anwendung. Flooright [14] wurde konzipiert um Normen, Informationen und Tipps für alle Belange der Bodenbelagsindustrie bereitzustellen.

Sobald der Benutzer die Anwendung startet, wird der linke Dialog in Abbildung 5.4 angezeigt. Beim ersten Start wird automatisch eine Liste der verfügbaren Inhalte abgerufen. Sobald der Benutzer einen Eintrag auswählt, wird der verlinkte Inhalt heruntergeladen und wie im mittleren Dialog in Abbildung 5.4 angezeigt, wobei jeder Inhalt aus einem PDF-Dokument besteht. Falls in einem Dokument auf andere Informationen verlinkt wird, werden diese über den Webbrowser abgerufen und angezeigt. Eine Verlinkung zwischen den Inhalten selbst ist damit nicht möglich. Alle verfügbaren Inhalte werden in einer TabBar am unteren Bereich des Bildschirms kategorisiert. Tippt der Benutzer auf den *Mehr*-Button in der TabBar, werden ihm weitere Kategorien angezeigt. Jede Kategorie beinhaltet eine eigene Liste an verfügbaren Dokumenten, die abgerufen wird, sobald die Kategorie geöffnet wird. Alle Listen und die Dokumente werden nach ihrem Download auf dem Gerät gespeichert und sind anschließend offline verfügbar. Über den *Suchen*-Button in der TabBar gelangt der Benutzer zum rechten Dialog in Abbildung 5.4, wobei dort ein Fehlverhalten der Anwendung zu sehen ist. Im oberen Bereich kann ein Stichwort in das Suchfeld des Dialogs eingegeben werden, nach dem gesucht werden soll. Darunter werden alle Suchergebnisse aufgelistet, die nach dem Auswählen heruntergeladen und analog zum mittleren Dialog in Abbildung 5.4 angezeigt werden.

Während der Bedienung der Anwendung fallen zwei Fehler auf, die mit der Suche zusammenhängen. Jedes Suchergebnis besteht aus einem Titel und einem Untertitel. Falls im Titel ein Umlaut enthalten ist, wird dieser mit der HTML-Codierung für ein Unicode Zeichen dargestellt, z.B *ä* anstatt *ä*. Die Umlaute im Untertitel werden dahingegen korrekt dargestellt. Scrollt der Benutzer bei der Suche nach unten, werden weitere Suchergebnisse Blockweise nachgeladen und angezeigt. Sobald das letzte Suchergebnis erreicht wird, werden die Untertitel aller Suchergebnisse durch den Text *"Suche war erfolglos."* ersetzt. Beide Fehler sind im rechten Dialog der Abbildung 5.4 zu sehen.

Abbildung 5.4: Flooright: Inhalte und Suche

5.3 Vergleich der Anwendungen

In diesem Abschnitt werden beide Anwendungen mit der hier erstellten Anwendung verglichen. Tabelle 5.1 zeigt einen Überblick, wie sich die Anwendungen unterscheiden. Falls eine Markierung in Klammern steht, wurden Annahmen über die entsprechende Funktion gemacht.

Alle Inhalte der Flooright Anwendung sind frei verfügbar, weshalb sie auf einen Login verzichten kann. Im Gegensatz dazu ist der Zugang sowohl zur NormenBibliothek, als auch zur hier erstellten Anwendung nur Benutzern mit gültigen Zugangsdaten möglich. Ein offline Login scheint bei der NormenBibliothek nicht möglich zu sein, da bei einem Loginversuch ohne Internetverbindung direkt eine entsprechende Fehlermeldung angezeigt wird. Allerdings gibt es auf dem *Login*-Dialog eine Option "angemeldet bleiben", die dem Benutzer nach einer erfolgreichen Anmeldung am Server beim nächsten Start der Anwendung auch ohne Internetverbindung Zugang gewährt. Alle drei Anwendungen stellen einen Download der Inhalte von einem Server bereit. Sobald ein Inhalt heruntergeladen wurde, kann dieser ohne Internetverbindung geöffnet und angezeigt werden. Außerdem bieten alle drei Anwendungen die Möglichkeit alle verfügbaren Inhalte zu durchsuchen.

	NormenBibliothek	Flooright	eigene Anwendung
Login erforderlich	✓	✗	✓
offline Login	(✗)	✗	✓
Mehrere Benutzer an einem Gerät	✓	✓	✓
Inhalte offline verfügbar	✓	✓	✓
Inhalte online beziehbar	✓	✓	✓
Suche	✓	✓	✓
Verlinkungen zwischen Inhalten	(✓)	✗	✓
Profilbereich	(✗)	✗	✓
Einstellungen	✗	✗	✓
Lesezeichen	✓	✗	✗
Hilfe über einen Service	✓	✗	✗

Tabelle 5.1: Vergleich der Anwendungen

Flooright ist die einzige Anwendung, bei der die Inhalte nicht untereinander verlinkt werden können, da alle Links über den Webbrowser des Betriebssystems geöffnet werden. Im Vergleich zu den anderen Anwendungen, besitzt nur die hier erstellte Anwendung einen Profilbereich, der Informationen zum aktuellen Benutzer anzeigt. Außerdem gibt es in keiner der beiden anderen Anwendungen Einstellungen, mit denen das Verhalten der Anwendung individuell für jeden Benutzer angepasst werden kann. Dahingegen ist die NormenBibliothek die einzige Anwendung, die es einerseits ermöglicht Inhalte mit einem Lesezeichen zu versehen und andererseits Hilfe über einen Service anbietet. Beide Funktionalitäten wären eine nützliche Erweiterung für die hier entworfene Anwendung.

Der Vergleich mit diesen Anwendungen hat interessante mögliche Erweiterungen der eigenen Anwendung hervorgebracht. Im nächsten Kapitel werden alle Ergebnisse dieser Ausarbeitung zusammengefasst und Ideen für zukünftige Erweiterungen vorgestellt.

6

Fazit

Abschließend ist festzustellen, dass in dieser Arbeit alle ursprünglichen Ziele erreicht wurden, wobei anzumerken ist, dass Anforderung FA13 nicht umgesetzt wurde. Diese besagt, dass die Anwendung auf Push-Benachrichtigungen reagieren können soll. Damit ein Server eine Nachricht an den Apple Push Notification Service senden kann, welcher anschließend die Anwendung benachrichtigt, wird ein Zertifikat [15] benötigt, das im Umfang dieser Arbeit nicht zur Verfügung stand.

Durch die Definition der Inhalte eines Pakets wurde eine Grundlage geschaffen, die es ermöglicht, beliebige Informationen, unabhängig von ihrem Inhalt, von der Anwendung anzeigen zu lassen. Außerdem können Inhalte damit auch paketübergreifend verlinkt werden, wodurch thematische Querbezüge einfacher hergestellt werden können. Damit können gesuchte Informationen effizient mit Hilfe der Anwendung nachgelesen werden. Die eingebaute Suche vereinfacht das Finden von Informationen zusätzlich.

Die Loginfunktion erlaubt eine Unterscheidung zwischen den Benutzern, so dass mehrere Benutzer das selbe Gerät benutzen können. Die Umsetzung des offline Logins brachte an dieser Stelle einige unerwartete Probleme mit sich. Damit sich ein Benutzer ohne Internetverbindung einloggen kann, muss sein Passwort auf dem Gerät gespeichert werden. Um dem Benutzer einen gewissen Datenschutz zu bieten muss das Passwort an dieser Stelle verschlüsselt werden. Dafür war es geplant den bcrypt-Algorithmus einzusetzen. Allerdings existiert kein Framework, das diesen Algorithmus für iOS implementiert. Aus diesem Grund wurde im Zuge dieser Arbeit die entsprechende Java-Implementierung übersetzt, um den offline Login, der in der Anwendung eine wichtige Rolle spielt, vollständig zu implementieren. Ein weiteres Problem entstand, als ein Update für die Programmiersprache veröffentlicht wurde. Dieses Update veränderte die Syntax an einigen Stellen, was dazu führte, dass das gesamte Projekt angepasst werden musste.

Mit Hilfe des Logins und durch eine entsprechende Ordnerstruktur im Dateisystem wird sichergestellt, dass jeder Benutzer ausschließlich Zugriff auf seine eigenen Informationen erhält. Die Anbindung an einen Webservice erlaubt es jedem Benutzer individuell zusätzliche Informationen herunterzuladen.

Insgesamt löst die hier entwickelte Anwendung das Problem, dass ein Mitarbeiter der Qualitätssicherung zur Analyse eines Fehlers mehrere Fachbücher benötigt. Auf einem mobilen Endgerät können diese Informationen immer mitgeführt werden, um den Mitarbeiter vor Ort zu unterstützen. Außerdem löst die Anwendung das Problem, dass die Suche nach speziellen Informationen in Fachbüchern einige Zeit in Anspruch nimmt. Über eine Stichwortsuche können relevante Informationen direkt abgerufen werden. Als letztes wird das Problem gelöst, dass die Fachbücher keinen direkten Bezug zueinander haben. Ein erkannter Fehler eines Produktes kann Ursachen aus verschiedenen Fachbereichen haben. Die jeweiligen Fachbücher decken dabei nur ihren eigenen Themenbereich ab. In der Anwendung können Inhalte beliebig verlinkt werden. Dadurch ist es möglich Querbezüge zwischen verschiedenen Fachgebieten herzustellen.

Abschließend werden einige Ideen, die unter anderem während der Bearbeitung dieser Arbeit aufgekommen sind, für zukünftige Erweiterungen der Anwendung vorgestellt.

6.1 Ausblick

Eine erste mögliche Erweiterung wäre die Implementierung der Push-Benachrichtigungen aus Anforderung FA13, die den Benutzer zum Beispiel über Updates für Pakete informieren können.

Wie in Abschnitt 5.1 festgestellt wären Favoriten eine nützliche Erweiterung für die Anwendung. Mit Hilfe dieser Funktion könnte der Benutzer wichtige Inhalte markieren und eventuell mit eigenen Kommentaren versehen, um über einen eigenen Dialog direkt auf diese Inhalte zugreifen zu können.

Eine weitere nützliche Funktion wäre die Möglichkeit Ishikawa-Diagramme, die auch Ursache-Wirkungs-Diagramm [16] genannt werden, innerhalb der Anwendung erstellen zu können. Dieses Diagramm hilft bei der Analyse eines Problems um die möglichen Ursachen herauszufinden. Dabei werden die einzelnen Ursachen so lange in kleinere Teile zerlegt, bis die Fehlerquelle gefunden wurde.

Zusätzlich wäre eine Möglichkeit zur direkten Kommunikation mit einem Servicecenter denkbar. Dabei sollte es möglich sein Text-, Foto-, Audio- und Videodateien in beide Richtungen übertragen zu können, um ein Problem im Zweifelsfall eindeutig identifizieren zu können. So könnte ein Mitarbeiter der Qualitätssicherung, der ein Problem erkennt aber keine detaillierten Fachkenntnisse im Problembereich besitzt, direkt Unterstützung einer Servicekraft anfordern um das Problem bestenfalls selbst lösen zu können.

Zusammenfassend lässt sich sagen, dass mit dieser Anwendung große Mengen verschiedenster Informationen schnell zur Hand sind, um auftauchende Unklarheiten bei der Qualitätssicherung effizient beseitigen zu können. Zusammen mit den vorgestellten Erweiterungen könnte die Produktivität unter der Verwendung dieser Anwendung sogar noch weiter gesteigert werden.

Literaturverzeichnis

[1] Koubek, A.: Praxisbuch ISO 9001:2015: Die neuen Anforderungen verstehen und umsetzen. Carl Hanser Verlag GmbH & Company KG (2015)

[2] Webber, L., Wallace, M., Hesse-Hujber, M.: Qualitätssicherung für Dummies. Für Dummies. Wiley (2012)

[3] Hartlieb, B., Kiehl, P., Müller, N.: Normung und Standardisierung: Grundlagen. Beuth Studium. Beuth Verlag GmbH (2009)

[4] DIN Deutsches Institut für Normung e. V.: Über Normen und Standards. (`http://www.din.de/de/ueber-normen-und-standards`) Accessed: 21.01.2016.

[5] Beuth Verlag GmbH: Beuth Verlag. (`http://www.beuth.de/de/`) Accessed: 21.01.2016.

[6] Klein, M., Kiehl, P.: Einführung in die DIN-Normen. Teubner (2001)

[7] e.V., D.: Auswertung der Experteninterviews: Arbeits-, Verbraucher-, Umweltschutz, Wirtschaftsverbände, Öffentliches Interesse. Gesamtwirtschaftlicher Nutzen der Normung. Beuth Verlag GmbH (2001)

[8] Fielding, R.T.: Architectural styles and the design of network-based software architectures. PhD thesis, University of California, Irvine (2000)

[9] The jQuery Foundation: jQuery. (`https://api.jquery.com/`) Accessed: 18.12.2015.

[10] Otto, M., Thornton, J.: Bootstrap. (`http://getbootstrap.com/`) Accessed: 18.12.2015.

[11] VDE Verlag GmbH: NormenBibliothek App. (https://www.normenbibliothek.de/app.html) Accessed: 05.01.2016.

[12] VDE Verlag GmbH: NormenBibliothek App. (https://itunes.apple.com/de/app/normenbibliothek/id443404605?mt=8&ls=1) Accessed: 05.01.2016.

[13] Märki, N.: Flooright. (https://itunes.apple.com/de/app/flooright/id450492639?mt=8) Accessed: 17.01.2016.

[14] Flooright AG: Flooright. (http://www.flooright.ch/) Accessed: 17.01.2016.

[15] Apple Inc.: iOS Developer Library - App Distribution Guide. (https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/AddingCapabilities/AddingCapabilities.html) Accessed: 12.01.2016.

[16] Syska, A.: Produktionsmanagement: Das A - Z wichtiger Methoden und Konzepte für die Produktion von heute. Gabler Verlag (2007)

A

Quelltexte

In diesem Anhang sind einige wichtige Quelltexte aufgeführt.

```
{ "navigation":[
    {
      "id":1,
      "title":"Titel des Inhalts",
      "uri":"./files/index.html",
      "description":"Beschreibung des Inhalts",
      "icon":{
        "android":"./files/icons/android/info.png",
        "ios":"./files/icons/ios/info.png",
        "windows":"./files/icons/win/info.png",
        "www":"./files/icons/web/info.png"
      },
      "children":[]
    }
  ]}
```

Listing A.1: Aufbau der toc.json

```
{
  "keywords":[
    {
      "keyword":["Schluesselbegriffe","zum","Vergleichen"],
      "uri":"./files/data/document.html"
    }
  ]
}
```

Listing A.2: Aufbau der keywords.json

```
{
  "id":1,
  "repository_id":13,
  "version_id":52,
  "name":"Repository Name",
  "description":"Beschreibung des Repositories",
  "versionname":"Version Name",
  "changelog":"Schreibfehler ausgebessert",
  "image":{
    "path":"./files/images/preview.jpg",
    "mimetype":"image/jpeg"
  },
  "released_at":"2016-01-21",
  "is_released":"1",
  "price":{
          "purchase":"20",
          "update":"5"
  },
  "created_at":{
        "date":"2016-01-21 12:15:00.000000",
        "timezone_type":3,
        "timezone":"UTC"
  },
  "updated_at":{
    "date":"2016-01-21 12:15:00.000000",
    "timezone_type":3,
    "timezone":"UTC"
  }
}
```

Listing A.3: Aufbau der meta.json

Abbildungsverzeichnis

Tabellenverzeichnis

Name: Markus Schiedel Matrikelnummer: 789805

Erklärung

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

Ulm, den .....

Markus Schiedel