HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Studienarbeit

pillow

Mobile App Service für die Hotel Industrie

Hochschule für Technik Rapperswil
Frühjahressemester 2013

Autoren Moreno Feltscher
Peter Manser

Betreuer Prof. Dr. Markus Stolze
Marktrepräsentant Hotel Hof Weissbad

Aufgabenstellung

Aufgabenstellung Studienarbeit Abteilung I, FS 2013
Peter Manser, Moreno Feltscher

Mobile App Service für die Hotel Industrie

1. Betreuer

Betreuer dieser Arbeit ist
Prof. Dr. Markus Stolze, Institut für Software mstolze@hsr.ch

2. Ausgangslage

Grosse Hotels, die etwas auf sich halten haben heutzutage ihre eigene App. Für kleinere Hotels ist dies jedoch unerschwinglich. Ein einfacher Website der für Mobil-Geräte optimiert ist sind deutlich billiger in der Erstellung, aber auf Features wie Push Notifikationen oder ortsbezogene Dienste muss dann verzichtet werden. Zudem fehlt es Hotels oft an Personal welches mächtige Web-Content Management Systeme wie TYPO3 oder Wordpress bedienen kann.

3. Ziele der Arbeit

In dieser Studienarbeit soll eine prototypische Implementation eines Services entwickelt werden der spezifisch auf die Bedürfnisse von Hotels ausgerichtet ist und diesen erlaubt auf einfache Weise den mobilen Auftritt eines Hotels zu erstellen und zu warten.
Mit der prototypischen Implementation soll gezeigt werden, inwieweit sich Funktionen wie Push Notifikationen und ortsbezogene Dienste im Rahmen eines solchen Service realisieren lassen. Besonderes Augenmerk ist hierbei auf die einfache Bedienbarkeit der App durch Hotelgäste (z.B. Installation, erste Nutzung, Push, erneute Nutzung nach längerer Pause) und durch Hotelangestellte (Erstellung und Pflege von Inhalten) zu legen.
Konkret sind folgende System-Komponenten zu analysieren, zu prototypen und zu testen.

1. Cross Platform App (Einfache Bedienung durch verschiedenste Typen von Hotelgästen)

2. Content Management Platform und Service (einfache Bedienung durch Hotelangestellte)

4. Zur Durchführung

Mit dem HSR-Betreuer finden in der Regel wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen. Alle Besprechungen sind von den Studenten mit einer Traktandenliste vorzubereiten und die Ergebnisse in einem Protokoll zu dokumentieren, welches stets zugreifbar ist.
Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen (gemäss Projektplan) sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein Feedback. Eine definitive Beurteilung erfolgt aufgrund der am Abgabetermin abgelieferten Dokumentation. Die Evaluation erfolgt aufgrund des separat abgegebenen Kriterienkatalogs in Übereinstimmung mit den Kriterien zur SA Beurteilung. Es sollten hierbei auch die Hinweise aus dem abgegebenen Dokument „Tipps für die Strukturierung und Planung von Studien-, Diplom- und Bachelorarbeiten“ beachtet werden.

5. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollen den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 2 Exemplaren abzugeben.
Zudem ist eine kurze Projektresultatdokumentation im öffentlichen Wiki von Prof. Dr. Markus Stolzezu erstellen. Diese muss einen Link auf ein öffentlich zugängliches (z.B. YouTube) „Video Poster“ enthalten, welche das Resultat der Arbeit dokumentiert (das Video sollte auch im Original auf der CD/DVD enthalten sein).
Dieses Video sollte nicht mehr als 3 Minuten lang sein. Das Video enthält einen Intro-Screen mit HSR-Logo, Titel der Arbeit, Namen der Studenten und Namen des Betreuers/Dozenten und Industriepartner (Standbild PPT Folie im HSR Corporate Design, 5 sec). Im ersten Inhaltsteil des Videos (30 sec) wird erklärt warum das gelöste Problem relevant ist: Warum braucht es das System, warum ist die Arbeit ohne System mühsam oder unmöglich? Der zweite Inhaltsteil des Videos (max. 50 sec) gibt eine Systemübersicht und zeigt anhand eines Einsatzbeispiels was das Resultat dieser Arbeit wem bringt. Abschluss mit Zusammenfassung der „Nutzer-Benefits“. Im dritten Inhaltsteil (max. 30 sec) werden die „Errungenschaften“ der Arbeit prägnant dargestellt. Insbesondere werden gemeisterte technische und organisatorische Herausforderungen aufgelistet (z.B. Architektur auf PPT Folie im HSR Corporate Design). Im Abspann (5 sec) sollte der Intro-Screen wieder eingeblendet werden. Das Material für die Video-Erstellung wird von der HSR gestellt (Multimediastelle). Das Video ist mit Unterti-

teln auszustatten. Auf eine Tonspur sowie Musik ist zu verzichten. Beispiele von HSR-Videos zu verschiedenen Arbeiten finden Sie bei YouTube (z.B. Kinect Bodyscanner http://www.youtube.com/watch?v=Q1ngxAkiaRg)

6. Weitere Regeln und Termine

Im Weiteren gelten die allgemeinen Regeln zu Bachelor und Studienarbeiten „Abläufe und Regelungen Studien- und Bachelorarbeiten im Studiengang Informatik“ (https://www.hsr.ch/Ablaeufe-und-Regelungen-Studie.7479.0.html)

Der Terminplan ist hier ersichtlich (HSR Intranet) https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html

7. Beurteilung

Eine erfolgreiche SA zählt 8 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von ca. 25 bis 30 Stunden budgetiert. Entsprechend sollten ca. 240h Arbeit für die Bachelorarbeit aufgewendet werden. Dies entspricht ungefähr 17h pro Woche (auf 14 Wochen) und damit ca. 2 Tage Arbeit pro Woche pro Student.
Für die Beurteilung ist der HSR-Betreuer verantwortlich.
Die Bewertung der Arbeit erfolgt entsprechend der verteilten Kriterienliste.

Die Aufgabenstellung wurde am 18. Februar 2013 (Semesterstart) vorbesprochen, und am 6. März 2013 verschriftlicht. Die definitive Aufgabenstellung wurde am 11. März 2013 beschlossen. Letzte Typos am 29. Mai 2013 beisitigt.

Rapperswil, 30. März 2013

Prof. Dr. Markus Stolze
Institut für Software
Hochschule für Technik Rapperswil

Erklärung der Eigenständigkeit

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Moreno Feltscher

Peter Manser

Abstract

Kleine bis mittelgrosse Hotels mit beschränkten finanziellen Mitteln haben oftmals das Bedürfnis, ihre Kunden mittels Kommunikationshilfsmitteln an sich binden zu können. pillow als Cross Platform App erlaubt es deshalb solchen Hotels, auf einfache Weise ihre Kunden via Push Notifications erreichen zu können.
Als Repräsentant des Marktes stand das Hotel Hof Weissbad dem Projektteam zur Seite. pillow ermöglicht es Hotelbetreibern, mittels eines intuitiven CMS eigene Hotel Apps, bestehend aus verschiedenen Seiten welche das Hotel beschreiben, zusammenzustellen. Weiter können interessierte Besucher ihre Hotels favorisieren, um so automatisch mittels Einsatz von Push Notifications über Neuigkeiten informiert zu werden.
Das Backend (API und CMS) wurde mithilfe den Python Microframeworks Flask und Pewee umgesetzt, wobei für die persistente Datensicherung auf das Datenbanksystem PostgreSQL zurückgegriffen wird. Der durch den Benutzer verwaltbare Inhalt der einzelnen Apps wird hierbei auf dem Server gerendert und via API an den Client ausgeliefert. Weiter ist es für Hotelbetreiber möglich, im CMS Notifications an Interessenten zu versenden.
Die Mobilapplikation wurde auf der Basis von PhoneGap entwickelt. PhoneGap ermöglicht es dem Entwickler, Webapplikationen unter Einsatz von HTML5, JavaScript und CSS3 in einem Container nativ auf Mobilgeräten einzusetzen, ohne dabei auf native Features wie Push Notifications verzichten zu müssen. Als JavaScript-Framework wurde auf Backbone.js zurückgegriffen, welches dem Entwickler eine saubere Strukturierung des Codes ermöglicht und so die Maintainablity des Gesamtsystems gewährleistet. Bei der Entwicklung der Benutzeroberfläche wurde auf das Junior-Framework zurückgegriffen. Das UI sollte ansprechend wirken, da die App eine breite Interessensgruppe anzusprechen gedenkt, weshalb ein Fokus auf die Entwicklung einer modernen und für den Benutzer bekannten Oberfläche gelegt wurde.
Durch das Testingframework Jasmine wird mittels Behavior-Driven Tests die Lauffähigkeit und Korrektheit des JavaScript Codes sichergestellt.

Management Summary

Mobile Applikationen sind gegenwärtig im Trend und erleben einen wahrhaften Boom auf dem weltweiten Softwaremarkt. Die Nachfrage nach solchen Apps hat dazu geführt, dass sich heute viele Unternehmen mittels eines mobilen Auftritts repräsentieren wollen. Mit der steigenden Nachfrage sind jedoch auch die Kosten für die Entwicklung solcher Applikationen in die Höhe geschnellt, nicht zuletzt auch dadurch, dass auf verschiedenen Mobilplattformen entwickelt werden muss. Mussten anfangs lediglich wenige Plattformen, wie iOS oder Blackberry unterstützt werden, so ist die Nachfrage nach Android- oder Windows Phone Apps heute allgegenwärtig. Da die Apps grösstenteils nicht zwischen den verschiedenen Plattformen portiert werden können, treibt diese Entwicklung die Kosten rasant in die Höhe.

Um diesen plattformabhängigen Entwicklungsumgebungen Einhalt gebieten zu können, wurden so genannte Cross Platform Apps geschaffen. PhoneGap ist eine dieser Cross Platform Entwicklungsprodukte, welches es einem erlaubt, mittels Einsatz von Webtechnologien wie HTML5, JavaScript und CSS3 Applikationen zu entwickeln, welche dann auf verschiedene Mobilplattformen portiert werden können. Hierbei fallen die Nachteile reiner Webapplikationen wie zum Beispiel die fehlende Unterstützung von nativen Features grösstenteils weg. Marketingtechnisch stellt die Verfügbarkeit dieser Apps in den jeweiligen App Stores gegenüber den reinen browserorientierten Webapplikationen ebenfalls einen Vorteil dar.

pillow wurde durch den Einsatz der oben genannten Technologien als Cross Platform App entwickelt und kann somit auf verschiedenen Mobilplattformen zum Einsatz kommen. In einer ersten Version unterstützt die App die Plattform iOS von Apple. pillow wurde als App, welche dem Benutzer ein Hotelverzeichnis bietet, realisiert. Es soll somit für Hotelbetreiber möglich sein, auf eine einfache Art sein Hotel als App werbetechnisch verkaufen zu können. Hierzu wurde ein CMS, mithilfe wessen der Hotelier seine Inhalte verwalten kann, geschaffen. Das CMS kann mit allen gängigen Browsern bedient werden und erfordert somit keine Installation von Drittsoftware. Der CMS-Benutzer bestimmt nicht nur den Inhalt selbst, sondern kann auch dessen Darstellung beinflussen. Die einzelnen Hotelseiten werden dann vom Server zusammengestellt und mittels API für die Client-Software zur Verfügung gestellt. Das gesamte Backend (CMS und API) baut auf den Python Mikroframeworks Flask und Pewee auf. Für die persistente Datensicherung wird die Datenbanksoftware PostgreSQL eingesetzt. Damit eine klare Struktur bezüglich der Organisation des JavaScript Codes und somit die Wartbarkeit der App gewährleistet

werden kann, kommt das bewährte Framework Backbone.js zum Einsatz.
Bei der Gestaltung der App wurde darauf geachtet, dass ein modernes und konsistentes Erscheinungsbild zum Durchstöbern der Hotelinformationen einlädt.
Durch das Testingframework Jasmine wird mittels Behavior-Driven Tests die Lauffähigkeit und Korrektheit des JavaScript Codes sichergestellt.

Um es dem Benutzer besonders einfach zu machen, ein Hotel in pillow zu finden, wurde zusätzlich noch ein App-Redirecting umgesetzt. Dies ermöglicht es einem Hotel, deren Hotel-App mittels einer speziellen URL oder QR-Code direkt über den Browser des Mobilgeräts zu öffnen. Eine Anwendungsmöglichkeit dieses Redirects wäre zum Beispiel ein mit einem QR-Code versehenes Willkommensplakat am Eingang des Hotels, welches den Benutzer darauf hinweist, dass das besuchte Hotel ebenfalls bei pillow registriert ist.

Es wäre denkbar, dass mittels pillow ebenfalls die Webseite der jeweiligen Hotels abgelöst werden könnte. Da die Inhalte bereits auf dem Server vorliegen und deren Erscheinungsbild individuell angepasst werden kann, wäre ein solcher Schritt sehr naheliegend. Die technische Umsetzung dieser Idee würde keine allzu grossen Aufwendungen mit sich bringen, da nur der Rendering-Prozess entsprechend einer Ausgabe im Browser angepasst werden müsste. Diese Komponente war jedoch nicht Teil der Studienarbeit und soll deshalb als Zukunftsvision verstanden werden.

Inhaltsverzeichnis

Kapitel 1 Einleitung

Diese Studienarbeit befasst sich mit der Entwicklung einer Cross Platform App, welche mittels eines Service zentral verwaltet werden kann. Die App soll sich an Hotelbetreiber richten, welche auf eine einfache Weise ihre Kunden mithilfe von Push Notifications erreichen und somit binden wollen.

Motivation

Da wir uns in unserem Beruf stark mit der Webentwicklung befassen, wollten wir etwas in diesem Bereich zum Thema unserer Studienarbeit machen. Wir wurden im täglichen Leben schon mehrmals nach den Kosten für die Entwicklung einer Mobile App angefragt. Die daraus resultierende Antwort, welche den Kunden auf die immens hohen Kosten, welche vor allem durch die Entwicklung für verschiedene Plattformen verursacht wird, hinweist, wir jeweils mit Ernüchterung zur Kenntnis genommen. Diese Geschäftslücke wollen wir mit einer generischen App, welche einmal entwickelt, für verschiedene Hotellerie-Betriebe zum Einsatz kommen kann, erschliessen.

Mehrwert für den Benutzer

Gerade im Bereich der Hotellerie sehen wir einen grossen Mehrwert einer App, welche verschiedene Informationen zum Hotel selbst beinhaltet. Noch heute findet man auf dem Hotelzimmer Prospekte oder Infoschreiben in Papierform vor, welche dem Gast die Vorteile des Hotels anzupreisen versucht. Der Nachteil dieser Lösung ist in erster Linie die daraus resultierende örtliche Abhängigkeit, welche in diesem Falle das eigene Hotelzimmer darstellt, sowie der Ressourcenverschleiss in Form der auf Papier gedruckten Infoschreiben.
Ein weiterer interessanter Aspekt einer solchen Hotel-App aus Sicht des Kunden stellen angepriesene Sonderangebote dar. Hierbei punktet die Anwendung, da der Benutzer automatisch über solche Aktionen informiert wird und deshalb kein Angebot mehr verpasst.

Mehrwert für den Hotelbetreiber

Der wohl wichtigste Mehrwert, welcher aus dem Einsatz von pillow resultiert, ist die Kundenbindung. Diese wird dadurch erreicht, dass der Kunde per Push-Notification je-

weils auf dem Laufenden gehalten werden kann, unabhängig davon, ob er sich im oder ausserhalb des Hotels aufhält. Die kontextbasierte Differenzierung (Unterscheidung zwischen in- und out-house Informationen) verstärkt diesen Effekt umso mehr, da der Gast detailliert mit Informationen beliefert werden kann.
Der Ersatz von Informationsschreiben in Papierform ergibt für das Hotel selbst einen finanziellen Vorteil, da sowohl Ressourcen, wie auch Personalkosten, welche durch die Erarbeitung und Auslieferung der Dokumente anfallen, eingespart werden können.
Da die vom Hotelbetreiber der App zur Verfügung gestellten Informationen es pillow ebenfalls ermöglicht, eine Webseite für das Hotel zu deployen, könnte längerfristig gesehen auf eine losgelöste eigens entwickelte Webseite verzichtet werden, was wiederum einer Kosteinsparung gleichkommt.

Kapitel 2 Technische Herausforderungen

Die erarbeiteten technischen Herausforderungen wurden, wo als notwendig empfunden, in den Risiken unter Punkt D.7.1 „Risiken“entsprechend behandelt.

2.1. Architektur

Scalability

Die ganze Infrastruktur soll scalable sein, also auch falls viele Benutzer die App benutzen, soll die App schnell reagieren und Antwort geben.
Diese Herausforderung wird durch das Risiko „1 - Performance“ widerspiegelt.

Erweiterbarkeit

Die Architektur soll so gebaut werden, dass sie einfach erweiterbar ist.

2.2. Backend / CMS / API

Das Backend ist das zentrale Datenverwaltungstool welches dem Hotel dazu dient, seine Inhalte zu verwalten. Das Backend muss soweit generisch sein, dass es in Zukunft möglich ist, generische Seiten zu erstellen.

Generik

Die Seiten der Web App sollen soweit abstrahiert werden, dass es in Zukunft einfach möglich ist, neue Seiten zu erstellen. Dazu muss ein gutes Konzept und eine gute Datenstruktur ermittelt werden.
Diese Herausforderung wird durch das Risiko „8 - Generik CMS“ widerspiegelt.

API

Es muss eine gute API Struktur definiert werden, damit es einfach und effizient ist Abfragen zu erstellen. Es muss aus Performance gründen darauf geachtet werden, dass dem Benutzer nur die Daten geliefert werden, welche er wirklich benötigt.

2.3. Mobile App

Vorerst wird eine iOS App angezielt, welche eine generische App darstellt. Generisch heisst, dass diese App verschiedene Web Apps beinhalten kann. Ein Hotel erstellt also, wie im Teil „Backend / CMS“ beschrieben, seine eigenen Web App. Diese kann dann von dieser App heruntergeladen werden.

Performance

Der Einsatz von modernen Technologien in der Web App ist uns sehr wichtig, daher setzen wir auf HTML5 & CSS3. Dies besonders daher, da uns die Performance der App sehr wichtig ist.
Diese Herausforderung wird durch das Risiko „1 - Performance“ widerspiegelt.

App-Links

Um eine möglichst gute User Experience bieten zu können, soll es dem Benutzer sehr einfach gemacht werden, die gewünschte Hotel App zu finden und zu laden. Der Benutzer soll die Möglichkeit haben, einen QR Code zu scannen, welcher auf eine URL zeigt. Diese URL öffnet wenn möglich die Generische App und zeigt den die Web App des Hotels an. Falls die App noch nicht installiert ist, wird die URL im Safari geöffnet, wo die Web App angezeigt wird. Der Benutzer wird darauf hingewiesen, dass er durch Download und Installation der App eine verbesserte Nutzung erhält.

Abbildung 2.1.: App-Link Konzept

Es muss mittels Architekturprototyp analysiert werden, ob dies realisierbar ist. Falls ein gute Lösung gefunden wird, soll ein Blog-Post darüber verfasst werden. Diese Herausforderung wird durch das Risiko „4 - App-Links“ widerspiegelt.

Push-Notifications

Es soll für einen Hotelbetreiber möglich sein, den Benutzer mittels Versand von Push-Notifications informieren zu können. Hierbei wird zwischen Informationen, welche der Benutzer während des Hotelaufenthaltes, aber auch ausserhalb des Hotels erhält, unterschieden.
Diese Herausforderung wird in Risiko „5 - Push-Notification“ im Bezug auf die User Experience konkretisiert.

Caching
Eine Web App soll in unsere generischen App gespeichert werden können, damit sich dieses lokal auf dem Telefon befindet und diese überall verfügbar ist. Bei Aktualisierungen muss die App die neuen Inhalte herunterladen und diese wiederum abspeichern.

Background-Sync
Um die Usability noch weiter zu erhöhen, würde sich anbieten, eine WebApp via Background-Synchronisation zu aktualisieren. Hiermit ist gemeint, dass der Server bei einer Inhaltsaktualisierung eine Push-Notifikation an alle Benutzer sendet und dann dessen App, diese neuen Inhalte automatisch herunterlädt und einspielt, ohne das der Benutzer etwas davon merkt.
Da dieses Konzept momentan noch zu viele Unsicherheiten mit sich bringt, stellt wird diese Herausforderung nicht im Rahmen dieses Projektes behandelt.

Push Notifications
Das Hotel soll die Möglichkeit besitzen, den Gast via Push Nachrichten zu benachrichtigen. Dies könnte beispielsweise ein Spezialangebot („Heute Nachmittag Massagen 50%“). Die Schwierigkeit besteht hier besonders darin, dass man wissen muss, ob der Gast an der Information interessiert ist. Hier sollten verschiedene Faktoren in die Entscheidung miteinbezogen werden:

- Erlaubt der Gast Push Notifications?
- Check-in des Gasts (aktuell im Hotel?)
- Geo Location

Es könnte jedoch auch eine allgemeine Information sein, bei welcher nicht nur der Gast, welcher sich im Hotel befindet, interessiert ist. (Bspw. „Buchen Sie jetzt Ferien und erhalten Sie 20% Rabatt“)

Geo Location
Aufgrund der Geo Location soll dem Benutzer gleich angeboten werden, eine Web App herunterzuladen und diese lokal auf dem Smartphone abzuspeichern. Dieses Thema ist komplex, da einige kritische Punkte gibt, die zu beachten sind. Aus diesem Grund stellt das Thema Geo Location keinen Projektfokus dar.

- Genauigkeit
- Akkuverbrauch

App Store Submission
Das Ziel soll es sein, eine erste Version der App in den App Store zu bringen oder zumindest einen Submission Prozess mal durchspielen und somit ein Feedback von Apple zu erhalten, was mit der App noch nicht in Ordnung ist. Dies ist auch als Meilenstein 4 (siehe Meilensteine) eingeplant.

Kapitel 3 **Anforderungsspezifikation**

3.1. Actors & Stakeholders

3.1.1. Potenzieller Hotelgast

Der potenzielle Hotelgast möchte sich gerne über verfügbare Hotels informieren und installiert hierzu die entsprechende Mobile App, welche ihm Informationen über ein gewünschtes Hotel liefert. Weiter möchte der Gast über spezielle Angebote auf dem Laufenden gehalten werden. Dies wird ihm mithilfe des Einsatzes von Push-Notification ermöglicht.

3.1.2. Hotelgast vor Anreise

Der Hotelgast möchte sich im Vorfeld über seinen Aufenthalt informieren. Er kann über die App bereits Informationen über das Hotel beziehen.

3.1.3. Hotelgast vor Ort

Der Hotelgast möchte Informationen, welche seinem Hotelaufenthalt dienen, abrufen. Diese Informationen stehen ihm in der Mobile App zur Verfügung, sobald er sich im Hotel befindet und dies entsprechend bestätigt hat (Check-In).

3.1.4. Hotelier

Der Hotelier erfasst Informationen zu seinem Hotel. Hierbei unterscheidet er zwischen Daten, welche für den Gast entweder innerhalb oder ausserhalb des Hotels nützlich erscheinen.
Weiter zeigt er sich verantwortlich für die Art, wie die Informationen dargestellt werden sollen und aktiviert entsprechende Module.
Er benutzt zur Verwaltung das von pillow zur Verfügung gestellte CMS.

3.1.5. Administrator

Dem Admin stehen sämtliche funktionellen Möglichkeiten die der Server und die API haben zur Verfügung. Er zeigt sich ebenfalls Verantwortlich für die Verwaltung der Hotels.

3.2. Ist-Szenarien

Name	Rolle (Actor)
Edi Huber	Potenzieller Hotelgast
Kurt Müller	Hotelgast vor Anreise
Doris Keller	Hotelgast vor Ort
Hans Meier	Hotelier

Tabelle 3.1.: Anforderungsspezifikation: Beispiel-Personen Ist-Szenarien

3.2.1. Öffnungszeiten Erlebnisbad

Doris Keller gönnt sich am Abend nach dem Besuch einer Massage zusammen mit ihrer Kollegin noch ein Mineralwasser auf dem Stockwerk ihres Zimmers. Die beiden beschliessen, dem Erlebnisbad noch einen Besuch abzustatten, wissen jedoch nicht, ob dieses überhaupt noch geöffnet ist. Sie begeben sich deshalb zurück ins Zimmer, um auf dem aufliegenden Infoschreiben die entsprechenden Öffnungszeiten nachzuschlagen. Nach einiger Zeit ist die gewünschte Information gefunden. Leider ist das Bad bereits geschlossen und so entscheiden sich die beiden, den Besuch auf den morgigen Tag zu verschieben.

3.2.2. Spezialangebot Nordic Walking

Hans Meier möchte gerne sein Sonderangebot, welches 50 Minuten Nordic Walking zum Preis von CHF 80.00 anstelle von den üblichen CHF 95.00 anbietet, an den Mann bringen. Hierzu hängt er beim Eingang des Hotels ein entsprechendes Infoschreiben aus. Leider profitieren lediglich zwei ältere Paare von dem Angebot. Bei der Rückfrage bei anderen Hotelgästen stellt sich heraus, dass die Mehrheit das Schreiben schlicht übersehen hatten.

3.2.3. Spezialangebot Übernachtung

Edi Huber hat sich vor einiger Zeit über das Hotel Hof Weissbad auf deren Webseite informiert und war begeistert von der Vielzahl der Angebote, welche angepriesen werden. Leider ist ihm der Preis pro Einzelzimmer etwas zu hoch, weshalb er sich auch noch nach weiteren Hotels umschaut. Nur eine Woche später erfasst Herr Meier ein Spezialangebot, welches den Preis des Einzelzimmers von CHF 290.- auf CHF 200.- reduziert. Herr Huber erfährt nichts von diesem Angebot, da er die Webseite kein weiteres Mal konsultiert und entscheidet sich demzufolge für ein anderes Hotel, welches etwas günstiger ist.

3.2.4. Newsletter

Hans Meier verschickt vierteljährlich per E-Mail einen Newsletter an dessen Abonnenten mit den wichtigsten Geschehnissen rund um das Hotel Hof Weissbad. Eines Tages wird

er von einem älteren Stammgast darauf angesprochen, dass dieser seit Längerem seine E-Mails nur noch sporadisch überprüft und deshalb manch wichtige Information übersehen hat. Der Gast fragt sogleich nach, ob es vielleicht möglich wäre, dass die Informationen per SMS verschickt werden könnten.

3.2.5. Information über bevorstehenden Aufenthalt

Kurt Müller hat vor einiger Zeit telefonisch ein Zimmer im Hotel Hof Weissbad reserviert und freut sich riesig auf den bevorstehenden Aufenthalt. Drei Tage vor der Abreise erhält er deshalb automatisch eine SMS, welche ihn freundlich daran erinnert, dass seine Anreise kurz bevorsteht und man ihm eine gute Reise wünsche.

3.3. Soll-Szenarien

Folgende Personen werden in den Szenarien verwendet.

Name	**Rolle (Actor)**
Edi Huber	Potenzieller Hotelgast
Kurt Müller	Hotelgast vor Anreise
Doris Keller	Hotelgast vor Ort
Hans Meier	Hotelier

Tabelle 3.2.: Anforderungsspezifikation: Beispiel-Personen Soll-Szenarien

3.3.1. Öffnungszeiten Sauna

Doris Keller würde gerne in die Sauna gehen, weiss jedoch nicht, ob diese noch geöffnet hat. Sie nimmt ihr iPhone hervor und öffnet die App. Sie findet dort einen Link zum Gesundheitszentrum, wo sich die Öffnungszeiten der Sauna finden lassen. Die Sauna hat noch zwei Stunden geöffnet, daher kann sie sich gemütlich auf den Weg in die Sauna machen.

3.3.2. Wochenprogramm

Kurt Müller wird morgen ins Hotel anreisen, und interessiert sich, was ihn erwartet. Er weiss, dass das Hotel ein Wochenprogramm hat und öffnet daher die App auf seinem iPhone. Er sieht da den Link zum Wochenprogramm und kann sich so gut informieren, was diese Woche im Hof Weissbad ansteht.

3.3.3. Anreise

Das CRM merkt, dass Kurt Müller morgen anreisen wird. Das Hotel Hof Weissbad möchte Ihre Gäste jeweils herzlich empfangen und möchte daher dem Kurt Müller eine gute Reise wünschen und generiert hierfür eine Push Notifikation, welche an Kurt Müller gesendet wird. Kurt Müller erhält diese und freut sich auf den Aufenthalt im Hotel Hof Weissbad.

3.3.4. Ankuft

Kurt Müller trifft in Hotel ein, wo er eine pillow Tafel findet mit einem QR Code. Um sich beim Hotel einzuchecken nimmt er sein iPhone hervor und scannt den QR Code ein. Die pillow-App öffnet sich und lädt automatisch die Hotel Hof Weissbad App und begrüsst Kurt Müller herzlich. Er ist jetzt im Hotel eingecheckt.

3.3.5. Frühstücks Buffet

Doris Keller sitzt gemütlich in ihrem Zimmer. Sie plant gerade ihren morgigen Tag und möchte daher wissen, wann es Frühstück gibt. Sie nimmt Ihr iPhone zu Hand und findet in den Allgemeinen Informationen, dass das Frühstücks Buffet von 8.00 - 10.30 geöffnet hat.

3.3.6. Bus nach Appenzell

Doris Keller möchte einen Ausflug ins Dorf Appenzell machen. Sie weiss, dass das Hotel einen Shuttle Bus anbietet, weiss jedoch nicht die genauen Abfahrtstermine. Sie nimmt ihr iPhone hervor und öffnet die den Abfahrtsplan, welcher in der App verlinkt ist. Sie hat herausgefunden, dass der Bus um 13:00 und 15:00 fährt.

3.3.7. Spezialangebot Massagen

Hans Meier will den Hotelgästen ein Spezialangebot für eine „Massage mit Appenzeller-Kräuter-Stempeln“ anbieten. Dazu geht er ins CMS und erfasst die Meldung: „Jetzt Massage mit Appenzeller-Kräuter-Stempeln für 80 CHF statt 105 CHF.“. Die Gäste welche sich aktuell im Hotel befinden erhalten die Meldung sofort auf ihrem iPhone. Doris Keller sitzt gerade in der Lobby, als sie die Meldung erhält. Sie geht zur Rezeption und macht sich einen Termin für diese Massage.

3.3.8. Vergünstigtes Ferienangebot

Hans Meier hat gemerkt, dass das Hotel für den April noch nicht ausgebucht ist. Um dies zu ändern, möchte er seine Gäste mit einer einfachen Mitteilung über ein Spezialangebot informieren. Er öffnet das CMS, klickt auf Notifikationen, und gibt den Text ein: „Ferien im April jetzt mit 20% Rabatt buchen.“. Kurt Müller entdeckt diese Meldung auf seinem iPhone und öffnet die App. Darauf findet er eine längere Mitteilung mit den genauen Angeboten. Durch Klick auf die Telefonnummer wird er gleich mit dem Hotel Hof Weissbad verbunden und kann so seine Ferien buchen.

3.3.9. Newsletter

Hans Meier hat einen Newsletter zusammen gestellt und möchte diesen den Gästen zur Verfügung stellen. Dazu geht er ins CMS und lädt das PDF hoch und verlinkt dieses in der App. Zu diesem Zeitpunkt möchte er den Gast jedoch nicht belästigen, da die

Neuigkeiten nicht wichtig sind. Kurt Müller entdeckt beim Nächsten öffnen der App, das der neue Newsletter vorhanden ist, er öffnet diesen und liest ihn durch.

3.3.10. Suche nach Hotel

Edi Huber sucht sich ein Hotel für die Herbstferien. Er stösst dabei auf die Plattform pillow wo er auch das Hotel Hof Weissbad findet. Dort findet er eine schöne Repräsentation des Hotels mit vielen Informationen und Fotos. Darauf hin entscheidet er sich beim Hotel Hof Weissbad anzurufen und einen 1 wöchigen Aufenthalt zu buchen.

3.4. User Stories

Aus den Szenarien wurden User Stories generiert. Diese wurden im Jira erfasst. Die User Stories wurden in zwei Komponenten unterteilt: App und CMS.

3.5. Nicht funktionale Anforderungen

3.5.1. Mengenanforderungen

Es sollen minimal 1‘000 Web Apps und 1’000 gleichzeitige Benutzer vom System verwaltet werden können. Zum jetzigen Zeitpunkt kann noch nicht genau ermittelt werden, wie viele Ressourcen (Server) hierfür nötig sind.
Die Mengenanforderungen werden mittels Belastbarkeitstest mithilfe der Software ApacheBench oder Apache JMeter überprüft.

Aktualisierung vom 30. Mai 2013
Die Belastbarkeitstest wurden mittels Apache JMeter durchgeführt. Die 1’000 Web Apps und 1’000 gleichzeitigen Benutzer konnten erfolgreich getestet werden. Detaillierte Resultate zu den Test und entsprechende Erkenntnisse sind im Anhang H Load Tests zu finden.

3.5.2. Qualitätsmerkmale

Funktionalität

Angemessenheit

Ein ressourcenschonender Umgang auf der Client-Seite ist sehr wichtig, weshalb die Übertragung von Daten vom Server an den Client möglichst die tief gehalten werden soll. Eine Übertragung neuer Daten soll nur dann vorgenommen werden, wenn neue Versionen des Contents auf der Serverseite vorliegen. Dies wird durch den Einsatz verschiedener Revisionen erreicht.

Aktualisierung vom 9. Mai 2013
Bei einem kurzen „Audit“ von uns, haben wir überprüft dass nur nötige Requests an

den Service gelangen. Der Client nimmt ein Caching vor, somit fragt er die Inhalte pro App nur einmal ab.

Richtigkeit

Bei einer gültigen Abfrage der Hotelinhalte sollen diese zu 100% den freigeschalteten Informationen entsprechen. Dabei muss jeweils versichert werden, dass die neuste Version des Inhalts zur Verfügung steht.
Weiter muss darauf geachtet werden, dass die über Geolocation in Erfahrung gebrachte Position des Benutzers korrekt einem Hotel zugeordnet wird.
Diese Anforderungen stellen eine Produktanforderung dar und wird somit nicht im Rahmen dieses Projekts behandelt.

Interoperabilität

Bei einer neueren Version von Client und Server soll gewährleistet sein, dass eine ältere Clientversion immer noch voll funktionsfähig ist (Rückwärtskompatibilität gewährleisten).
Da die Abgabe dieser Arbeit lediglich eine Produktversion umfasst, wird die Rückwärtskompatibilität nicht als Teil des Projekts behandelt.

Sicherheit

Durch die Verwendung der App sollen keine Sicherheitsdefizite entstehen. Der Server ist nur über definierte Schnittstellen zu erreichen (Service- und Clientschnittstellen). Datenschutzrelevante Daten werden über eine gesicherte Verbindung übertragen.
Bei der Umsetzung des Projekts wird kein Augenmerk auf die Sicherheit gelegt, weshalb auch keine Tests zum Beispiel mittels Security Audit vorgesehen sind.

Zuverlässigkeit

Bei einer funktionierenden Anbindung an das Internet soll in 90% der Fälle eine korrekte Kommunikation mit dem Server stattfinden.
Diese Anforderung wird im Rahmen der Belastbarkeitstests überprüft.

Aktualisierung vom 30. Mai 2013
Die Belastbarkeitstest wurden mittels Apache JMeter durchgeführt. Die Fehlerrate des Servers lag bei ~1%. Detaillierte Resultate zu den Test und entsprechende Erkenntnisse sind im Anhang H Load Tests zu finden.

Fehlertoleranz

Sofern keine Verbindung zum Server aufgebaut werden kann soll dies in der App entsprechend ersichtlich sein (benutzerfreundliche Statusmeldung).
Hierzu werden keine automatisierten UI-Tests durchgeführt, sondern lediglich manuelle Tests mit ausgeschalteter Datenverbindung direkt am Mobilgerät vorgenommen.

Abbildung 3.1.: Fehlertoleranz: App bei fehlender Internetverbindung

Benutzbarkeit

Verständlichkeit

Das GUI von App soll intuitiv zu bedienen sein. Dabei sollen bestehende moderne Konzepte adaptiert werden. Um die Verständlichkeit frühzeitig testen zu können werden während Sprint 2 - Entwurf und Sprint 3 - Prototyping erstellte Wireframes mittels Paper-Prototyping getestet.
Die Umsetzung des GUI des CMS steht nicht im Fokus dieses Projekts und wird deshalb nicht getestet.

Erlernbarkeit

Die App soll möglichst einfach zu installieren sein. Nach dem Herunterladen soll der Client mittels Scannen eines QR-Codes mit maximal 5 Klicks eingerichtet sein.

Bedienbarkeit

Es soll möglich sein die App nur mittels Touch Bedienung und Gestures zu bedienen (ohne Texteingabe). Der Aufwand um die App bedienen zu können, soll möglichst gering gehalten werden.
Es werden hierbei im Rahmen des Projekts keine automatisierten UI-Tests eingesetzt.

Attraktivität

Das visuelle Auftreten der Clientsoftware soll möglichst ansprechend und schlicht gehalten werden. Das einheitliche und durchdachte Design sollte beim Benutzer einen professionellen Eindruck hinterlassen.
Die Verständlich der Design-Konzepte werden mittel Paper-Protoyping-Tests eruiert.
Es werden hierbei im Rahmen des Projekts keine automatisierten UI-Tests eingesetzt.

Effizienz

Zeitverhalten

Das Zeitverhalten der App ist abhängig von der Stabilität und Geschwindigkeit der Verbindung. Dies ist je nach Standort unterschiedlich, daher ist momentan keine genaue Aussage darüber möglich. Die serverseitige Architektur soll so ausgelegt werden, dass die Ressourcen den Anforderungen entsprechend angepasst werden können. Tests mit dem App-Prototypen werden mehr Aufschluss über das Zeitverhalten geben.
Die maximale Antwortzeit des CMS bei einer gängigen Internetverbindung (5Mbps) soll 1 Sekunde betragen.
Diese Anforderung stellt eine Produktanforderung dar und wird somit nicht im Rahmen dieses Projekts behandelt.

Verbrauchsverhalten

Da der Webview unter iOS maximal 10MB für die Ausführung von JavaScript zur Verfügung stehen, muss der Clientteil so entwickelt werden, dass diese Limite nicht überschritten wird. Weiter müssen die entsprechenden Scripts in maximal 10 Sekunden abgehandelt werden. Resultierend kann gesagt werden, dass der Client-Teil so entwickelt werden muss, dass er sehr ressourcensparend operiert.
Die Auslastung der einzelnen Geräte wird bei der Entwicklung fortlaufend mittels entsprechender Debug-Tools überwacht. Es werden jedoch keine automatisierte Performance-Tests vorgenommen.

Wartbarkeit

Analysierbarkeit

Durch Fehlerlogging auf Serverseite, können Ursachen von Versagen schnell diagnostiziert werden.

Modifizierbarkeit

Änderungen am Serveralgorithmus und der Clientlogik müssen innerhalb von einem Manntag implementiert werden können. Andere Änderungen, die nicht das Kernsystem betreffen, sollten innerhalb von zwei Manntagen implementiert werden können.
Diese Anforderung stellt eine Produktanforderung dar und wird somit nicht im Rahmen dieses Projekts behandelt.

Stabilität
Änderungen an den Kernsystemen sollen zu 95% keine unerwarteten Fehler produzieren, andere Änderungen sollen zu 85% keine unerwarteten Fehler produzieren.
Diese Anforderung stellt eine Produktanforderung dar und wird somit nicht im Rahmen dieses Projekts behandelt.

Testbarkeit
Die serverseitige Implementation der Software kann einfach getestet werden, da nach MVC entwickelt wird. Die App sowie das CMS können als ganzes nicht ohne einen grossen Mehraufwand getestet werden (sprich UI-Tests). Einzelne Komponenten können jedoch ebenfalls in diesen Systemen mittels Unit Tests abgedeckt werden.

Übertragbarkeit

Anpassbarkeit
Die App soll unter iOS 6.1 oder höher lauffähig sein.
Das CMS soll in einem modernen Browser bedienbar sein.

Installierbarkeit
Die App soll über den Apple AppStore heruntergeladen und installiert werden können. Beim CMS ist keine besondere Installation notwendig (Internet-Browser reicht aus).

Koexistenz
Die Koexistenz der App ist durch das Betriebssystem iOS gewährleistet, welches entsprechende Schnittstellen zur Verfügung stellt. Für die serverseitige Implementation wird ein individueller Server aufgesetzt, daher wird dafür keine entsprechende Anforderung erfasst.

3.5.3. Randbedingungen / Abhängigkeiten

App-Plattform
Als Plattform, unter welcher unsere Mobile App zur Anwendung kommt, wurde iOS ab Version 6.1 gewählt. Rückwärtskompatibilität ist möglich, jedoch nicht garantiert, da zur Entwicklung einer App, welcher unter iOS 6 oder tiefer betrieben werden kann, die nötigen Ressourcen sowie Testgeräte fehlen.
Weiter wird Plattformunabhängigkeit angestrebt, das heisst, beim Design der App wird darauf geachtet, dass der Betrieb auch unter weiteren Plattformen wie Android oder Windows Phone möglich wäre. Dies wird durch den Einsatz Mobile-Framework PhoneGap im Zusammenspiel mit Webtechnologien ermöglicht.

Webtechnologien
Da bei der Entwicklung eine Plattformunabhängigkeit angestrebt wird, sollen folgende Webtechnologien zum Einsatz kommen:

HTML5
Da alle heutigen Mobile-Betriebssysteme HTML5-Unterstützung mit sich bringen, soll wo möglich auf dessen Features, wie den lokalen Speicher, zurückgegriffen werden.

CSS3
CSS3 erlaubt es uns, selbst aufwändige grafische Elemente oder Effekte, anstelle des Einsatzes von zum Beispiel JavaScript, performant im UI einzusetzen. Alle gängigen Mobile-Betriebssystem bringen bereits CSS3-Unterstützung mit sich, was dessen Einsatz legitimiert.
Zur besseren Übersichtlichkeit der einzelnen CSS3-Dateien können entsprechende Frameworks und Tools eingesetzt werden.

JavaScript
Zur Programmierung auf der Client-Seite wird auf JavaScript zurückgegriffen.

Client-Server Schnittstelle
Die Kommunikation zwischen dem Client (der App) und dem Server soll über eine serverseitige REST-API (state of the art, verhindert Overhead) erfolgen. Um unnötigen Overhead zu vermeiden sollen die Daten zudem im JSON-Format übermittelt werden.

3.6. Design Constraints

3.6.1. Cross Platform

Eine wichtige Anforderung an unser System ist es, eine Lösung zu entwickeln, welche einfach auf alle Plattformen wie iOS, Android, Blackberry, Windows Phone usw. portierbar ist. Diese Einschränkung wirkt sich auch sehr stark auf die Entscheidung für die Mobile-Abstrahierungsplattform (5.2.1) aus.

3.6.2. Push Notification

Eine wichtige Anforderung an unser System ist es, eine Lösung zu entwickeln, welche einfach auf alle Plattformen wie iOS, Android, Blackberry, Windows Phone usw. portierbar ist. Diese Einschränkung wirkt sich auch sehr stark auf die Entscheidung für die Mobile-Abstrahierungsplattform (5.2.1) aus.

Kapitel 4 **Domainanalyse**

4.1. Domainmodell

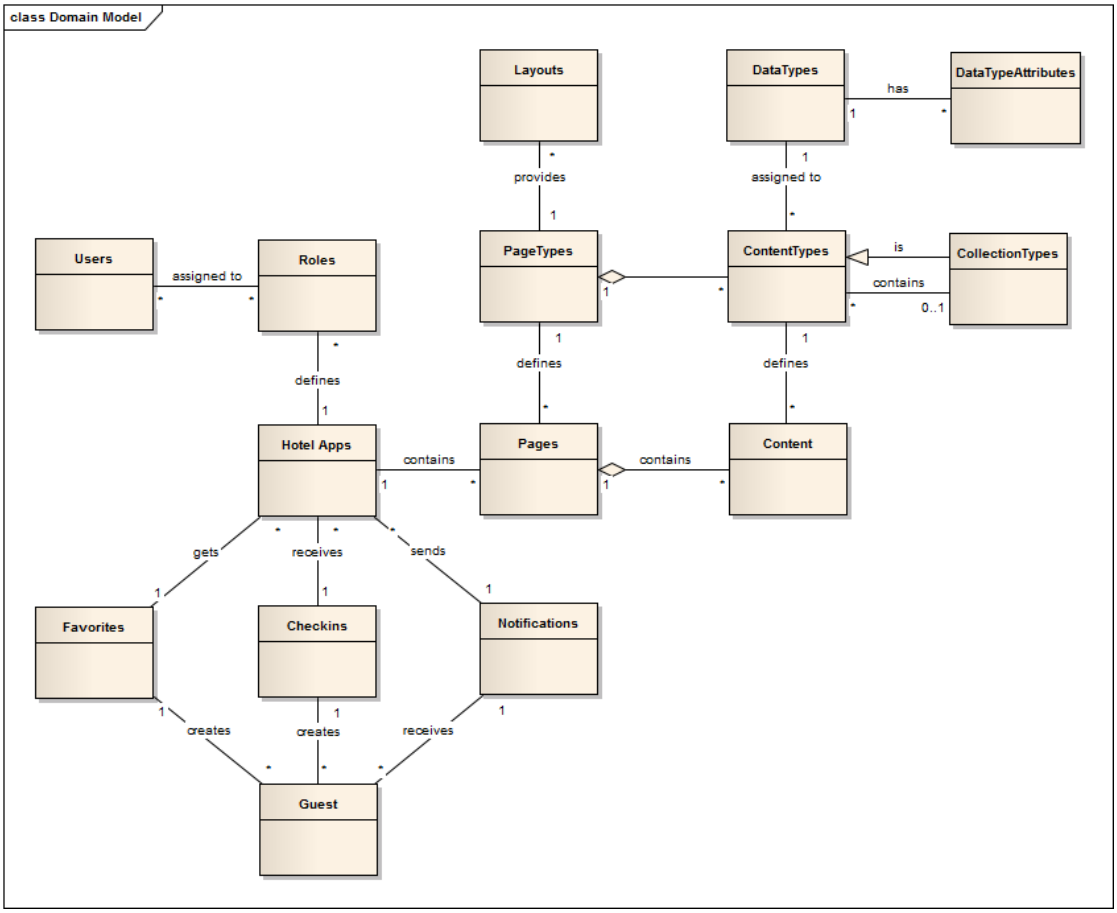

Abbildung 4.1.: Domainmodell

4.1.1. Hotel Apps

Die Hotel Apps stellen den zentralen Punkt des Systems dar.
Beispiel: App vom „Hof Weissbad“

4.1.2. Guest

Der Gast stellt der effektiven Benutzer des pillow Clients dar, welcher im Hotel zu Gast ist.
Beispiel: Gast „Edi Huber“

4.1.3. Favorites

Der Gast kann eine Hotel App zu seinen Favoriten hinzufügen und somit für ihn wichtig markieren.
Beispiel: Edi Huber favorisiert die App Hof Weissbad

4.1.4. Checkins

Der Gast kann sich in ein Hotel einchecken und sich somit für Updates dieses Hotel registrieren.
Beispiel: Edi Huber checkt in das Hof Weissbad (App) ein

4.1.5. Notifications

Das Hotel schickt Notifikationen an die Gäste um diese mit Angeboten und Neuigkeiten zu versorgen.
Beispiel: Das Hof Weissbad (App) sendet eine Notifikation „Massagenaktion“an Edi Huber.

4.1.6. Users

Der Hotelier ist der Benutzer der Hotel App und verwaltet deren Inhalte.
Beispiel: Hotelier „Hans Meier“

4.1.7. Permissions

Der Hotelier besitzt Berechtigungen auf eine oder mehrere Hotel Apps.
Beispiel: Hans Meier hat „Administratorenrechte“auf die App Hof Weissbad

4.1.8. PageTypes / Layouts

Der Seitentyp definiert den Typ und das Layout (Template) von Seiten
Beispiel: Seitentyp „Titel / Bild / Inhalt “mit Layout „Standard“

4.1.9. ContentTypes / DataTypes / DataTypeAttributes

Für Pages werden Inhaltstypen definiert, welche einem Datentype angehören. Dieser Datentyp kann verschiedene Attribute definieren.
Beispiel: Inhaltstyp „Teaserbild“ vom Typ „Bild“. Der Bild hat die Attribute „Quelle“ und „Titel“

4.1.10. Pages

Die Seite hat einen Seitentyp und ist einer App zugeordnet
Beispiel: Die App Hof Weissbad erstellt eine „Willkommensseite“ vom Typ Titel / Bild / Inhalt

4.1.11. Pages

Eine Seite hat Inhalte, welche gemäss Seitentyp und Inhaltstyp definiert sind.
Beispiel: Die Willkommensseite hat einen Titel „Willkommen“, ein Bild „Portrait Hof Weissbad“ und einen Einleitungstext über das Hotel.

5.1. Architektur Übersicht

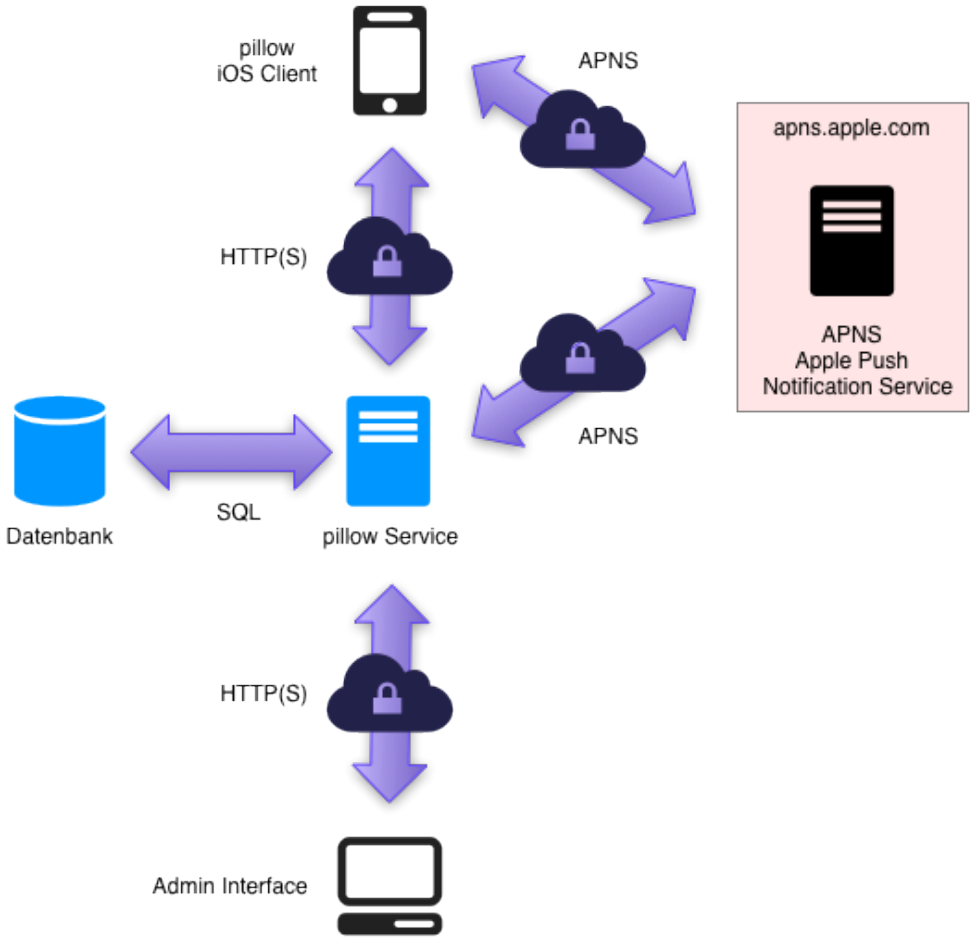

Abbildung 5.1.: Architektur Übersicht

5.1.1. Beschreibung

Zentraler Punkt des System ist der pillow Service. Er stellt die zentralle Schnittstelle zwischen den verschiedenen Teilsystemen dar. Der Service konsumiert die Daten direkt von der Datenbank, wo die Daten des Content Management System (CMS) persistent abgelegt werden. Die Daten werden, idealerweise verschlüsselt, via HTTP(S) vom pillow iOS Client oder vom Admin Interace (Web Client) abgegfragt. Ausserdem hat der Service die Aufgabe, Push Notifications via APNS zu versenden.

5.2. Detailkonzepte

5.2.1. Mobile-Abstrahierungsplattform

Bei der Evaluation der Mobile-Abstrahierungsplattform wurden sowohl aktuelle, populäre Abstraktions-Lösungen wie PhoneGap oder Sencha Touch, wie aber auch eine native Lösung in Betracht gezogen und dementsprechend in Form einer Nutzwertanalyse miteinander verglichen.
Hierbei wurden die für uns wichtigen Punkte wie Cross-Plattform-Kompatbilität, Performance oder auch unsere persönliche Erfahrung im Umgang mit dem Produkt, welche eine zeitsparende Entwicklung ermöglicht, besonders hoch gewichtet.

Nutzwertanalyse

		Bewertung Produkte [0-5]					
Kriterien	**Gewichtung [1-5]**	**Native**		**PhoneGap**		**Sencha Touch**	
Cross-Plattform	5	0	**0**	5	**25**	3	**15**
Push-Notification	4	5	**20**	4	**16**	0	**0**
Geolocation	3	5	**15**	4	**12**	5	**15**
Barcode-Scanner (Kamera)	4	5	**20**	3	**12**	0	**0**
Performance	5	5	**25**	3	**15**	4	**20**
Erfahrung	5	1	**5**	5	**25**	1	**5**
Community / Support	2	4	**8**	4	**8**	3	**6**
Open Source	2	1	**2**	5	**10**	2	**4**
Beständigkeit	2	4	**8**	4	**8**	4	**8**
Modularität	3	5	**15**	5	**15**	4	**12**
Dokumentation	3	3	**9**	3	**9**	4	**12**
Look and Feel	3	5	**15**	3	**9**	2	**6**
Gesamt			**142**		**164**		**103**

Abbildung 5.2.: Nutzwertanalyse Mobile-Abstrahierungsplattform

Sensitivitätsanalyse

Um die aus der Nutzwertanalyse gewonnenen Daten detailliert und unabhängig gewisser Kriterien verifizieren zu können, wurde eine Sensitivitätsanalyse durchgeführt.
Dem Kriterium „Erfahrung“ wurde deshalb bei der Erstellung der Nutzwertanalyse eine solch hohe Gewichtung zugesprochen, da die Erfahrung sich auf die Entwicklungszeit selbst auswirkt. Nichtsdestotrotz sind solche Werte natürlich sehr subjektiv und deshalb bei der Evaluation eines Produkts nicht wirklich zielführend. Wir haben deshalb im Rahmen der Sensitivitätsanalyse den genannten Faktor entfernt um unsere Entscheidung überprüfen zu können. Das Ergebnis dabei war, dass das Produkt PhoneGap nach wie vor am besten abschneidet, wenn auch nur noch mit zwei Punkten Vorsprung auf eine native App.

Entscheidung

Entsprechend oben dargestellter Analyse haben wir uns letztendlich für das Produkt PhoneGap entschieden. Der entscheidende Faktor bei dieser Entscheidung stellt sicherlich unsere Erfahrung bei der Entwicklung auf der Basis dieses Produkts, welche wir uns während unseres Challange Projekts im 5. Semester aneignen konnten, dar.
Weiter hat sich PhoneGap in einem ersten von uns im Zusammenhang mit der Abklärung des Risiken „App-Links“ angefertigten Prototyps als angemessene Lösung für unserer Aufgabenstellung erwiesen.

5.2.2. CMS / API

Bestehende CMS-Lösung

Auf dem Markt sind verschiedene CMS-Lösungen bereits erhältlich. Wir haben zwei dieser Lösungen genauer angeschaut und deren Adaptierbarkeit bezüglich der sehr spezifischen Problemstellung unseres Projekts untersucht.

WordPress

WordPress ist das am meisten verbreitete Blogging-System weltweit und wird vielerorts ebenfalls als CMS-Lösung eingesetzt. Sowohl Peter Manser, als auch Moreno Feltscher haben beruflich wie privat bereits verschiedene Web-Projekte auf der Basis von WordPress umgesetzt, was die Evaluierung entsprechend simpel gestaltete.
WordPress zeichnet sich unter anderem durch seine Erweiterbarkeit sowie ein ansprechendes Admin-Interface aus. Da dies grundlegende Anforderungen unseres Projektes sind, war eine Evaluation deshalb auch gerechtfertigt.
Leider mussten wir schnell feststellen, dass der vom Grundsystem gelieferte Funktionsumfang lediglich zu einem sehr kleinen Teil weiterverwendet werden kann und die allgemeine Architektur zum Teil sehr schlecht implementiert ist (allem voran schlechtes Datenbankdesign). Dieser Fakt hätte dazu geführt, dass wir einen Grossteil der Funktionalitäten selbst hätten implementieren müssen, was nicht wirklich den Grundgedanken einer Anbindung an ein bestehendes System darstellen kann.

Locomotive

Locomotive bietet es einem an, benutzerdefinierte Seiten- und Inhaltstypen zu erstellen. Diese können danach dazu verwendet werden, Inhalte auf einer Webseite auszugeben oder mittels REST-API entsprechenden Diensten zur Verfügung zu stellen. Da unser Produkt es vorsieht, dass der Benutzer eigene Seiten- und Inhaltstypen definieren kann, erschien uns Locomotive als die perfekte CMS-Lösung und wurde deshalb während der Prototyping-Phase auch eingehend getestet.
Leider wurde schnell klar, dass die Admin-Oberfläche ziemlich chaotisch daherkommt. Selbst für Personen mit einem gewissen technischen Background war es nicht ganz einfach, neue Inhaltstypen zu definieren. Da bei unserer Arbeit ein grosser Fokus auf die Usability gelegt werden soll, wurde uns schnell klar, dass genanntes Produkt sich nicht für den Einsatz als CMS eignet.

Entwicklung eigener CMS-Lösung

Da bestehende CMS-Systeme sich während der Prototyping-Phase als nicht geeignet für unsere Arbeit erwiesen, wurde die Entwicklung einer eigenen Lösung in Betracht gezogen.

Sprache

Es gibt verschiedene Sprachen welche gute Frameworks anbieten, welche uns bei der Entwicklung unseres generischen CMS (inkl. API) unterstützen. Die Frage ist also eher, welche Sprache wird gewählt - nicht welches Framework.

Ruby und Python bieten beide sehr gute Frameworks wie Rails und Django. Auch sehr gute Microframeworks sind in beiden Sprachen vorhanden: Sinatra und Flask. Beides sind gute Sprachen mit guten Konzepten, welche sich auch oft überschneiden.

Schlussendlich fällt die Entscheidung auf Python aus Erfahrungsgründen, da wir fundierteres Wissen in Python besitzen.

Framework vs Microframework

Die Entscheidung, was für eine Art Framework gewählt wird, ist eine Grundsatz Entscheidung. In Python stehen für beide Arten sehr gute Frameworks zur Verfügung.

Django (Framework)

Django bringt eine funktionsumfängliche Basis: Integrierter OR-Mapper, Administrationsoberfläche, Template Language und eine Modulsystem, welches einfach macht weitere Module einzubinden. Im allgemeinen schreibt Django viel vor, d.h. es ist eine klare Struktur gegeben. Für unser Projekt ist viel Flexibilität gefordert, so dass wir wahrscheinlich viele Komponenten von Django gar nicht so verwenden können und sich daher nicht wirklich unterstützend, sondern störend auswirkend können. Mehr Informationen zu Django.

Flask (Microframework)

Flask ist ein Microframework, welches auf Werkzeug und Jinja 2 basiert. Es bietet eine gute Grundstruktur, welche eine flexiblen Aufbau einer Applikation erlaubt. Flask bringt die wichtigsten Grundfunktionalitäten mit sich: Development Server, Request

Dispatching, Templating Language und integrierter Support für Unit Testing. Es ist ebenfalls sehr gut dokumentiert. Peter Manser kennt die Community gut und hat schon einige Sachen (privat / beruflich) mit diesem Framework gemacht. Mehr Informationen zu Flask.

Entscheidung

Aufgrund der Erfahrung mit Flask und die Flexibilität welche es bietet, entscheiden wir uns in einem ersten Schritt für Flask. Wir glauben damit die beste Lösung zu haben, da es uns erlaubt den Fokus auf die wichtigen Punkte in unserem Projekt zu legen.

5.2.3. Mobile App

Bei der Auswahl des HTML5 Mobile App Frameworks haben wir verschiedene auf dem Markt verfügbare Produkte untersucht und miteinander verglichen.

Sencha Touch

Sencha Touch wurde bereits als Abstrahierungsplattform unter Punkt 5.2.1 „Mobile-Abstrahierungsplattform“ evaluiert. Sencha Touch erlaubt es einem über die Abstrahierungs-Funktionalität hinaus ebenfalls, nur einen Teil des Gesamtprodukts zu Erstellung von mobile-fähigen HTML5-Applikationen zu benutzen. Hierbei kommt das JavaScript-Framework Sencha ExtJS zum Einsatz.
Trotz dem sehr guten Echo aus der Community und von Berufskollegen, welche uns Sencha Touch empfohlen haben, haben weder P. Manser, noch M. Feltscher haben bisher Applikationen auf der Basis von Sencha ExtJS entwickelt, was vom Projektteam als negativer Punkt aufgefasst wurde. Mehr Informationen zu Sencha Touch.

jQuery Mobile

jQuery Mobile ist ein touchoptimiertes JavaScript Framework für Smartphones & Tablets, welches auf der Basis von jQuery entwickelt wird. jQuery Mobile unterstützt hierbei die gängigsten Mobile Plattformen und hat eine breite Nutzerbasis.
Das Projektteam hat jQuery Mobile bereits in verschiedenen Schulprojekten eingesetzt und kennt das Framework deshalb. Leider wusste das Framework hierbei teilweise nicht wirklich zu überzeugen und zeichnete sich teilweise als sehr resourcenintensiv aus. Einen weiteren Schwachpunkt des Frameworks sieht das Projektteam in der Art, wie jQuery Mobile das DOM zur Darstellung manipuliert, was projektspezifische Anpassungen erschwert. Mehr Informationen zu jQuery Mobile.

Junior

Junior ist ein sehr junges Mobile Framework, welches verschiedene sich bewährte Frameworks (Modernizr, Zepto, Zepto flickable, Lodash, Backbone) zusammenfasst. Zur Darstellung der Inhalte wird auf das Prototyping-Framework Ratchet zurückgegriffen.
Ein Nachtteil hierbei stellt die Abhängigkeit von iOS-basierten Geräten dar, was im Zusammenhang mit der in der Studienarbeit geforderten Spezifikationen jedoch nicht im

Konflikt steht.
Mehr Informationen zu Junior.

Entscheidung

Da P. Manser schon verschiedene Projekte auf der Basis von Backbone umgesetzt hat und dessen Aufbau kennt, haben wir uns entschieden, Junior einzusetzen. Dieser Entscheid ist darauf zurückzuführen, dass wir ein Framework, welches sehr anpassungsfähig ist, einsetzen wollen. Erste Tests anhand eines Architekturprototypen haben uns dann auch in unserem Vorhaben bestärkt, da jegliche geforderten Design-Wünsche auf einfache Weise umgesetzt werden konnten.

5.2.4. Datenbank

Unsere Software muss, um die gewünschte Funktionalität gewährleisten zu können, in der Lage sein, verschiedene Informationen, wie Benutzer-, Standort- und Servicedaten verarbeiten und bereitstellen zu können. Um diese persistente Datenhaltung sicherzustellen ist der Einsatz eines geeigneten Datenbanksystems beinahe unabdingbar. Bekanntlich sind verschiedene dieser Systeme auf dem Markt erhältlich. Wir haben deshalb verschiedene Datenbanken im Bezug auf unser Projekt analysiert und eine entsprechende Evaluation vorgenommen. Es wurde hierbei ein Hauptaugenmerk auf nicht kommerzielle Produkte, welche eine relationale Datenhaltung ermöglichen, gelegt.

SQLite

SQLite ist eine Programmbibliothek, die ein relationales Datenbanksystem enthält. SQLite unterstützt einen Großteil der im SQL-92-Standard festgelegten SQL-Sprachbefehle. Unter anderem implementiert SQLite Transaktionen, Unterabfragen (subselects), Sichten (views), Trigger und benutzerdefinierte Funktionen. Das System ist vor allem für den Embedded-Einsatz entworfen, daher fehlen Funktionen wie die Möglichkeit, Objektberechtigungen zu verwalten (GRANT, REVOKE). Auch ein in der Konsole und in Shell-Skripten verwendbares, einfaches Frontend ist vorhanden. Die gesamte Datenbank befindet sich in einer einzigen Datei, eine Client/Server-Architektur ist nicht vorhanden.

Vorteile

- Keine Installation notwendig
- Einfach aufgebaut

Nachteile

- Geringer Funktionsumfang
- Keine Client/Server-Architektur
- Keine Möglichkeit, Datenbank zu skalieren

- Keine Berechtigungshierarchie implementierbar; kein Mehrbenutzerbetrieb möglich
- Für umfangreiche Datenhaltung ungeeignet, da sich Datenbank in einer Datei befindet
- Schlechte Performance, da Datenbank als Datei abgelegt wird

MySQL

MySQL ist die weltweit am meisten eingesetzte OpenSource-Datenbanklösung und wird von Oracle vertrieben. MySQL kann auf über 20 Plattformen eingesetzt werden, u.a. Linux, Windows, Mac OS, Solaris, HP-UX und IBM AIX.

Vorteile

- Stabilität
- Skalierbarkeit
- Enormer Funktionsumfang
- Ermöglicht komplexe Abfragen mit Unterabfragen (Subselects), auch geschachtelt

Nachteile

- Administration nur direkt auf Datenbank möglich, nicht über Shell
- Vertrieb durch kommerzielle Firma

PostgreSQL

Als objektrelationales Datenbanksystem implementiert PostgreSQL die Speicherung nicht atomarer Daten, Vererbung und Objektidentitäten und erlaubt Benutzern, das System um selbstdefinierte Datentypen, Operatoren und Funktionen zu erweitern. Die Unterstützung der referentiellen Integrität und ein fortschrittliches Transaktionsmanagement gehören ebenfalls zu den Leistungsmerkmalen von PostgreSQL, wie die Definition von Triggern und Regeln, mit denen Zugriffe auf Datenbankobjekte gesteuert werden können.

Vorteile

- Wird in Modulen Dbs1, Dbs2 und InfSys (durch Projektteam besucht) an der HSR unterrichtet
- Freie Nutzung: Reine OpenSource-Lösung, wird durch OpenSource Community gepflegt
- Umfassendes Transaktionskonzept, das Multiversion Concurrency Control (MVCC) unterstützt

- Ermöglicht komplexe Abfragen mit Unterabfragen (Subselects), auch geschachtelt
- Referenzielle Integrität (u. a. Constraints, Fremdschlüssel)
- Mengenoperationen
- Maximale Datenbankgröße nur durch zur Verfügung stehenden Speicher begrenzt
- Views, die mit Hilfe von Regeln (Rules und Triggers) auch schreibfähig sein können (Updatable Views)
- Trigger und gespeicherte Prozeduren (stored procedures) sind in verschiedenen Sprachen möglich:PL/pgSQL, PL/c, PL/Tcl, PL/Python, PL/Perl, PL/Java, PL/-PHP, PL/Ruby, PL/R, PL/sh, PL/Scheme, PL/Parrot, PL/V8 (derzeit noch experimentell)
- Schnittstellen zu vielen Programmiersprachen, u. a. C, Delphi, C++, Java/JDBC, Tcl, PHP, Perl, Python, Ruby sowie zu ODBC und .NET
- Export und Import sowohl von Daten als auch von Datenbankstrukturen (Schemata)
- Erweiterbarkeit durch Funktionen, selbstdefinierbare Datentypen und Operatoren

Nachteile

- Performanceeinbussen (im Vergleich zu MySQL) bei grossen Datenbanken

5.3. Externes Design App

5.3.1. Wireframes

Beim Entwurf des Externen Designs wurde zuerst die Grundstruktur des App-Designs anhand von Wireframes konzipiert. Eine erste Version der verschiedenen Screens sah folgendermassen aus:

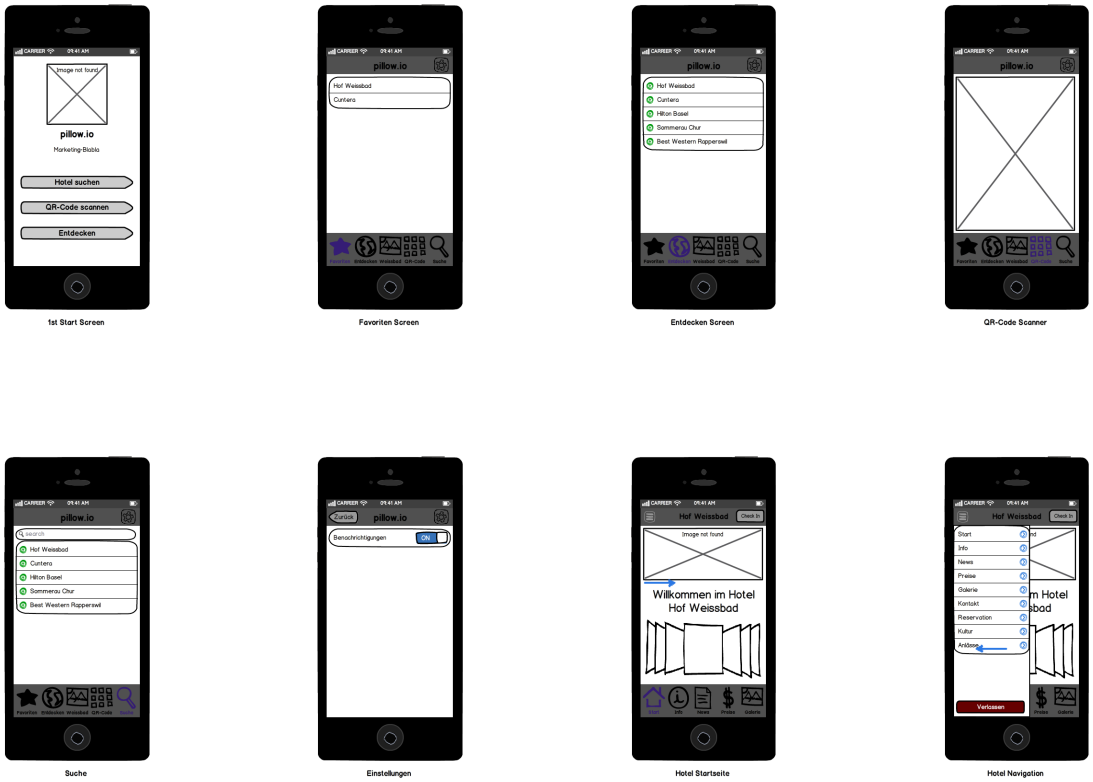

Abbildung 5.3.: Wireframes App - Version 1

Aufgrund von Feedbacks durch Prof. Dr. Markus Stolzeund auch durch Testbenutzer, welche die Mockups getestet hatten (siehe Paper-Prototyping), wurden einige Veränderungen den den vorliegenden Wireframes vorgenommen:

- Der „1st Start Screen“ , welcher beim ersten Start der App erscheinen sollte, wurde entfernt. Dies aus dem Grund, da es den Benutzer nur verwirren würde, da der Screen nur ein einziges Mal angezeigt würde. Die neue Lösung sieht vor, dass der Benutzer mittels Tooltips durch die App geführt wird und als „1st Start Screen“ die Auflistung der Favoriten erscheint.
- Die Menübar am unteren Rand wurde entfernt, da es sich dabei um ein iOS-

Designelement handelt und somit dem Grundgedanken einer Cross Platform App widerspricht.

- Der Button zum Öffnen der Navigation innerhalb der Hotel App wurde mit „Menü“ beschriftet, da bei den Paper-Prototyping Tests aufgefallen war, dass die technisch nicht sehr bewanderten Personen Mühe hatten, die Funktion des Buttons zu erkennen.
- Die Such- und Scan-Funktion wurde direkt in den „Entdecken“Screen integriert, da es sich um dieselbe Funktionalität handelt.

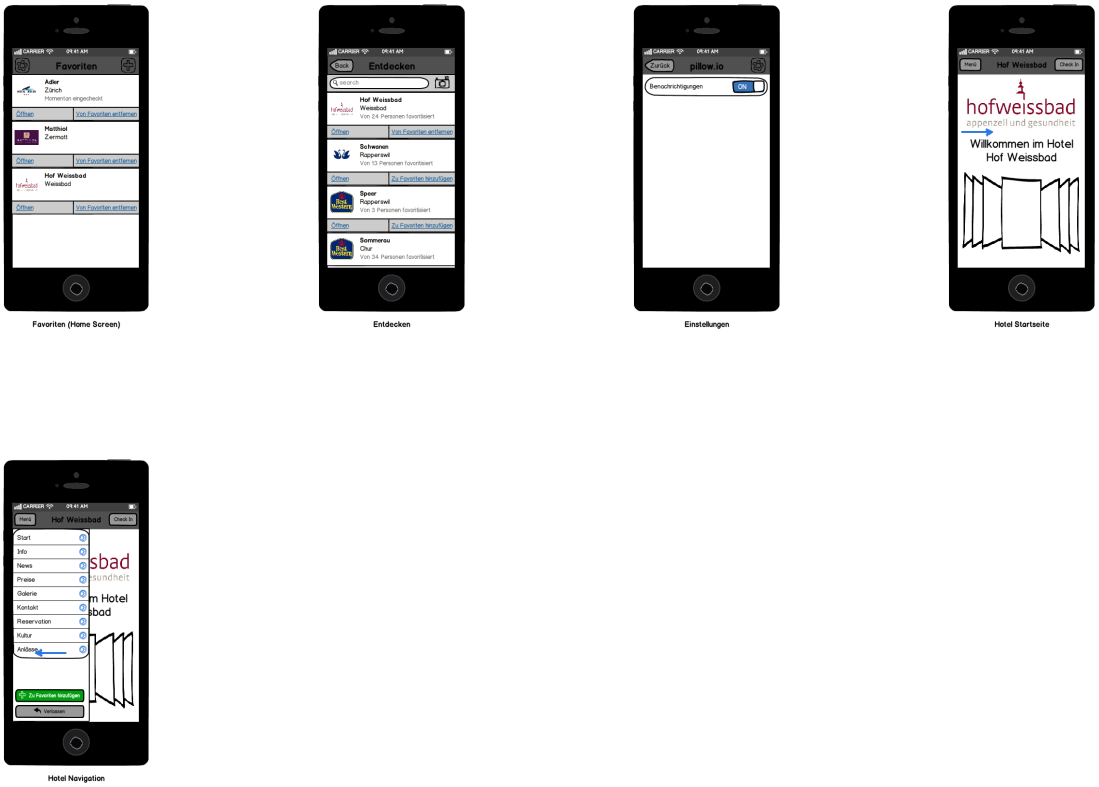

Abbildung 5.4.: Wireframes App - Version 2

Wie aus der obigen Illustration zu entnehmen ist, wurde die Anzahl der Screens signifikant verkleinert. Das Projektteam ist überzeugt, dass dies dem Benutzer zugute kommt, da er sich schneller zurecht findet und keine Redundanz im Bezug auf die Inhalte vorliegt.

5.3.2. Paper-Prototyping

Um die Adaptierbarkeit der erstellten Wireframes zu testen, wurde das Testverfahren *Paper Prototyping* angewendet. Bei der Auswahl der Testpersonen wurde besonders dar-

auf geachtet, dass diese der breiten Interessensgruppe der App Rechnung tragen.

Aufgabenstellung

1. Öffnen Sie die App
2. Suchen Sie unter den verfügbaren Hotels das Hotel Hof Weissbad und informieren Sie sich über deren Zimmerpreise
3. Kehren Sie zur Übersicht der verfügbaren Apps zurück
4. Schalten Sie in den Einstellungen die Benachrichtigungen aus
5. Fügen Sie das Hotel Hof Weissbad zu Ihren Favoriten hinzu
6. Öffnen Sie die Hotel-App des Hotel Hof Weissbads und checken Sie sich ein
7. Entfernen Sie das Hotel Hof Weissbad wieder von Ihren Favoriten

Tests

Ursula Feltscher (55 Jahre alt, Hausfrau & Legastenietherapeutin Teilzeit, iPhone-Benutzerin, kaum technischer Background)
U. Feltscher fand sich erfreulicherweise trotz eher spärlicher Erfahrung im Umfeld der mobilen Applikationen sehr schnell in der App zurecht. Sie konnte die gestellten Aufgaben problemlos bewältigen, obschon einige Unklarheiten, welche noch detaillierter erläutert werden, vorhanden waren.

Positive Punkte

- Unterteilung der Kontexte „App“ und „Hotel-App“ wurde verstanden
- Tooltips beim ersten Start der App gemäss Benutzer sehr hilfreich

Negative Punkte

- Menü in Hotel-App nicht gefunden, da Symbol dafür unbekannt war
- Einstellung erst nach einigen Versuchen gefunden, da auch hier Symbol nicht wirklich bekannt
- Option „Check In“ in Hotel-App unklar

Markus Feltscher (55 Jahre alt, Direktor Gebäudeversicherung, iPhone-Benutzer, tägliche Arbeit am Computer & Smartphone im geschäftlichen Umfeld)
M. Feltscher tat sich bei der Bedienung der App oftmals schwer und hatte einige Mühe das Konzept mit der Unterteilung der Kontexte „App“ und „Hotel-App“ zu verstehen. Nichtsdestotrotz konnten alle Aufgaben mit ein paar kleinen Hilfestellungen bewältigt werden.

Positive Punkte

- Tooltips beim ersten Start der App gemäss Benutzer sehr hilfreich
- Hinzufügen von Hotels zu Favoriten als intuitiv bezeichnet
- Benachrichtigungs-Einstellungen schnell gefunden

Negative Punkte

- Menü in Hotel-App nicht gefunden, da Symbol dafür unbekannt war
- Brauchte lange, die Aufgaben bewältigen zu können, da die Kontext-Unterteilung unklar war und deshalb zwischen den Screens hin und her gewechselt wurde
- Laut Benutzer zu viele Informationen auf den einzelnen Screens vorhanden

Christoph Bühler (25 Jahre alt, Informatik-Student HSR, iPhone-Benutzer, überdurchschnittlicher technischer Background)
C. Bühler als Power User konnte wie erwartet alle gestellten Aufgaben problemlos bewältigen, hat jedoch noch einige hilfreiche Feedbacks abgeben können.

Positive Punkte

- Intuitive Bedienung der App, hat sich schnell zurecht gefunden

Negative Punkte

- Klick auf Tooltips, welche beim ersten Starten der App angezeigt werden, öffnen nicht die entsprechenden Screens
- Hotels in der „Entdecken-Übersicht“ sollten klickbar sein, nicht nur über Text-Link das entsprechende Hotel darstellen
- Menü-Button in Hotel-App unklar - handelt es sich um Menü des Hotels oder Menü der App

Anregungen

- Plus-Button bei Start-Screen eventuell falsch, da Hotel-Übersicht damit dargestellt wird ⇒ eventuell durch Lupen-Button ersetzen

- „Zu Favoriten hinzufügen“- und „Öffnen“-Funktion in „Entdecken“-Screen visuell (Favorisieren als Stern) statt mit Text darstellen *Rightarrow* Screen wäre weniger überladen
- Favoriten eventuell als Stream von Neuigkeiten darstellen, da sonst verwirrend mit zusätzlichen „Entdecken“-Screen

Schlussfolgerung

Die Tests haben gezeigt, dass unsere Überlegungen beim Erstellen der Wireframes grundsätzlich beim Benutzer Anklang finden. Einige Unklarheiten, wie zum Beispiel das Aufrufen des Hotel-Menüs sowie die überladenen Screens, wurden durch Überarbeitung der Wireframes, welche unter 5.3.1 „Wireframes“ abgebildet sind, bewerkstelligt.

5.3.3. Produkt

Bei der Entwicklung des externen Designs der App wurden die anhand der Wireframes gewonnenen Erkenntnisse berücksichtigt. Bei der Farbwahl haben wir uns am folgenden pillow-Logo orientiert:

Abbildung 5.5.: Logo pillow

Nachfolgend einige Screenshots der fertiggstellten App:

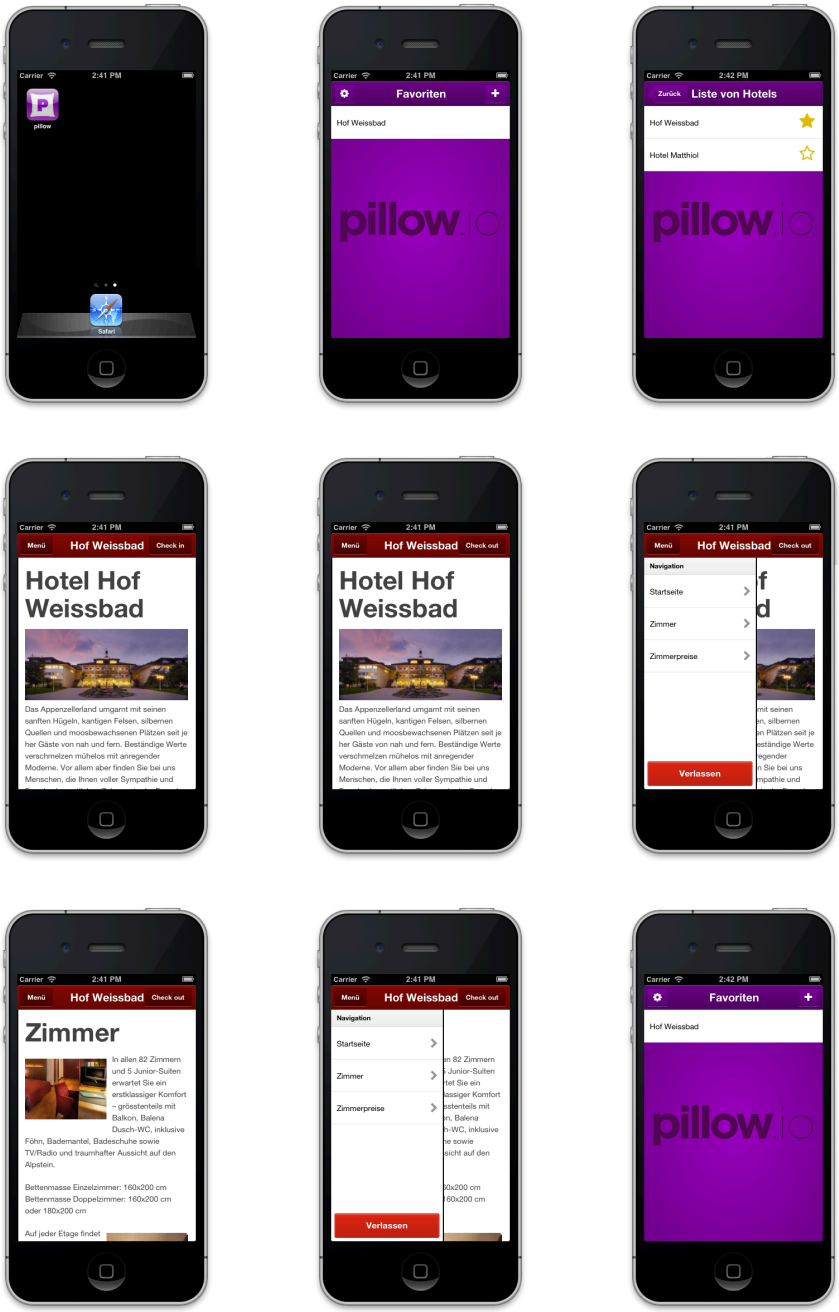

Abbildung 5.6.: App: Screenshots Teil 1

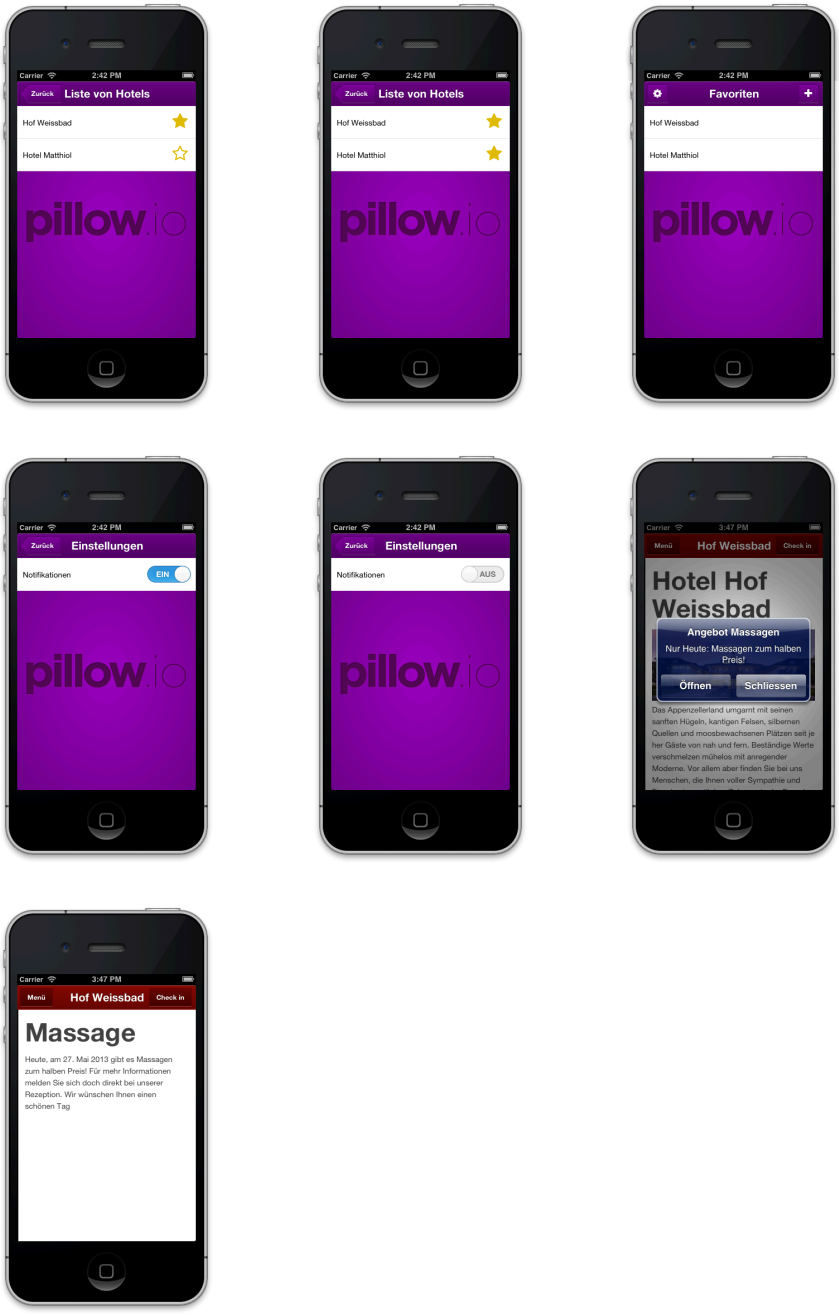

Abbildung 5.7.: App: Screenshots Teil 2

5.3.4. Barrierefreiheit

Die Hauptscreens des externen Designs wurden ebenfalls mittels Color Oracle auf Accessability getestet. Hierbei wurden folgende drei Sehschwächen analysiert:

Farbschwäche Grün

Betrifft ca. 5% der Männer
Ergebnis:

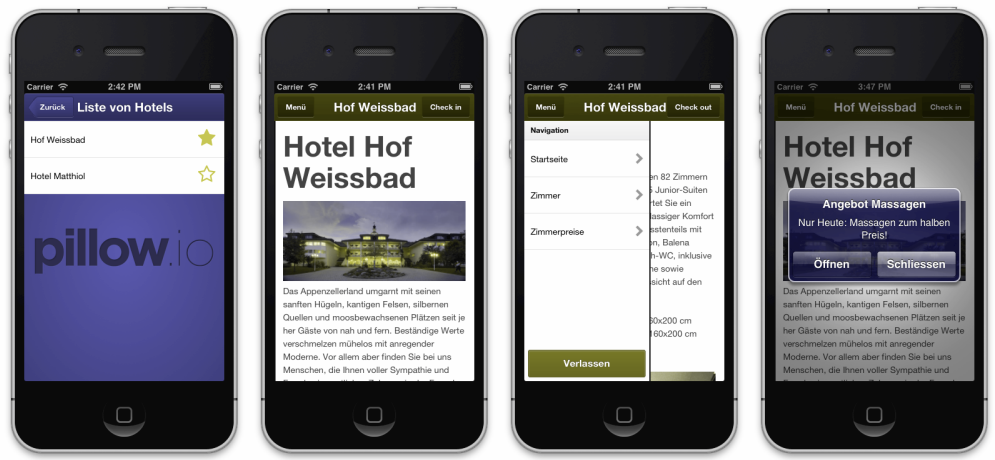

Abbildung 5.8.: App Barrierefreiheit: Farbschwäche Grün

Farbschwäche Rot

Betrifft ca. 2.5% der Männer
Ergebnis:

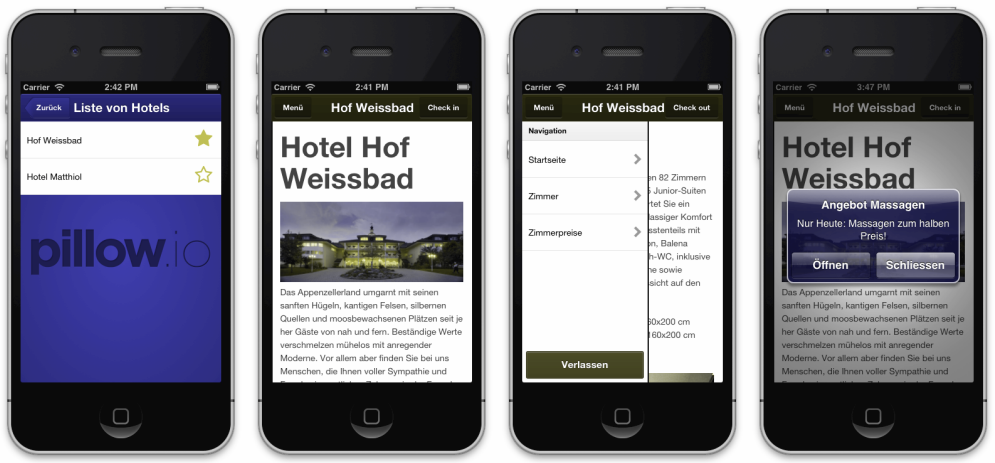

Abbildung 5.9.: App Barrierefreiheit: Farbschwäche Rot

Farbschwäche Blau

Betrifft ca. 0.5% der Männer

Ergebnis:

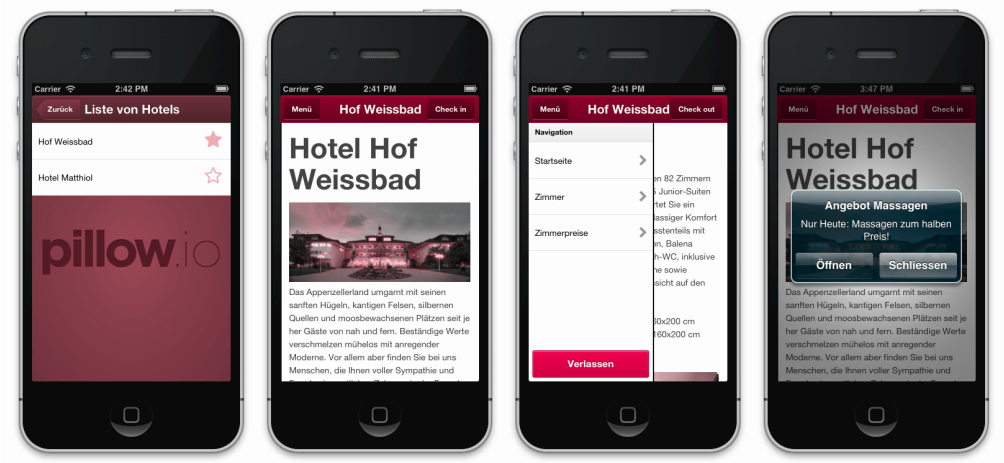

Abbildung 5.10.: App Barrierefreiheit: Farbschwäche Blau

Fazit: Wie zu erkennen ist, stellen diese Sehschwächen keinerlei Probleme bei der Handhabung der App dar.

5.4. Internes Design App

5.4.1. Komponentenmodell

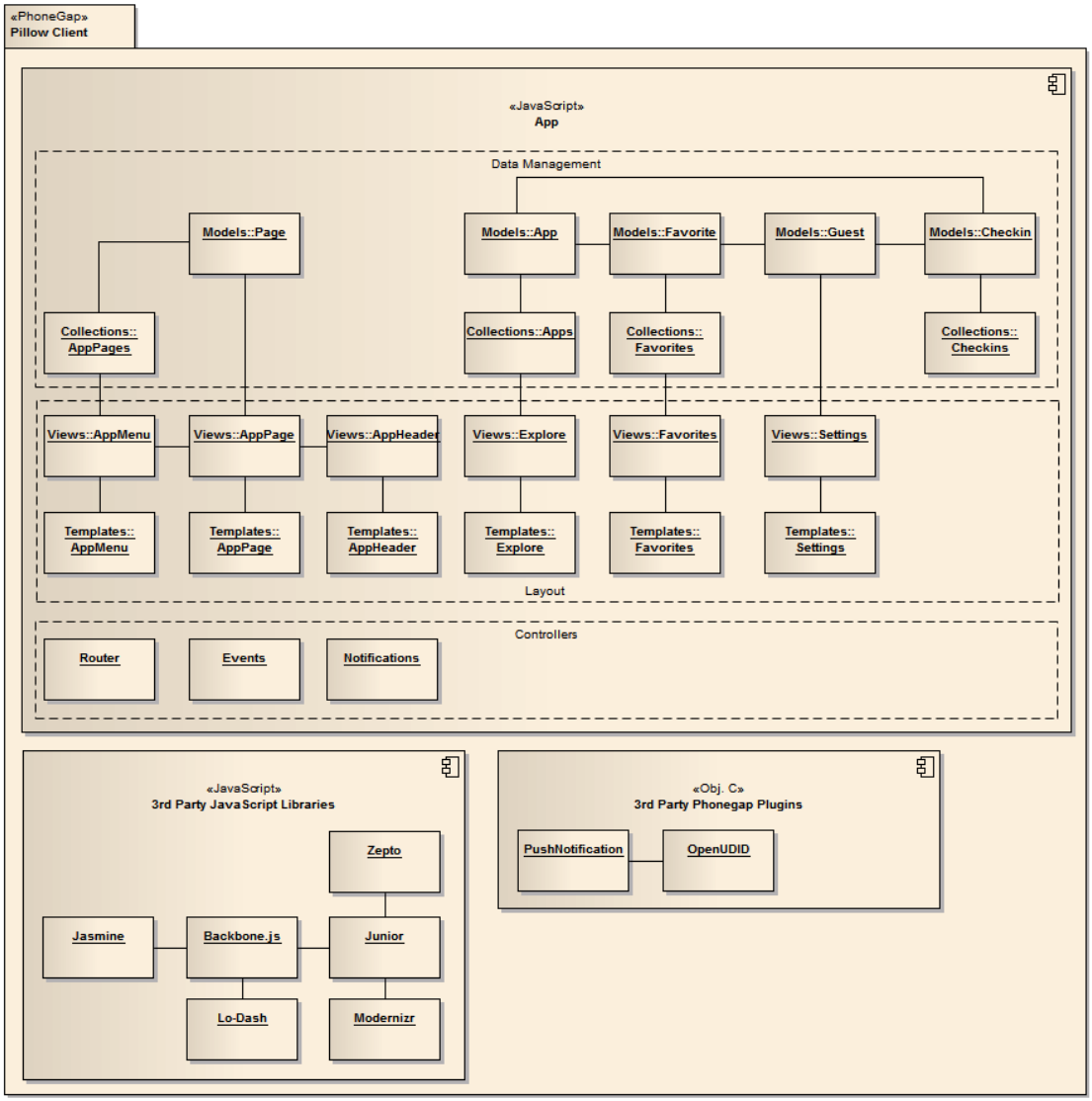

Abbildung 5.11.: App: Komponentenmodell

Der Pillow Client umfasst verschiedene Komponenten, auf welche wir hier kurz eingehen.

App

Die App stellt die Kernkomponente des pillow Clients dar. Sie wurde in Anlehnung an das

MVC-Pattern mithilfe der 3rd Party Library Backbone.js strukturiert. Es ist deshalb nur eine Anlehnung an das MVC-Pattern, als dass Backbone gewisse Controller-Funktionen in den Views implementiert. Weitere Informationen zu diesem Ansatz findet man auf der Webseite von Backbone.js.
Hier noch eine Auflistung der Subkomponente:

- **Data Management**
  Dieser Teil des pillow Clients umfasst die Models und Collections. Collections fassen dabei mehrere Models zusammen. Der Aufbau wurde so gewählt, dass die Backend-Resourcen möglichst einheitlich abgebildet werden. Die Kommunikation zum Backend mittels API geschieht auf dieser Ebene und nutzt dazu die durch Backbone.js zur Verfügung gestellten Funktionen.

- **Layout**
  Diese Subkomponente beinhaltet alle Views, welche sich für die Darstellung der Daten der jeweiligen Models/Collections verantwortlich zeigen. Weiter implementieren die Layout-Komponenten das Handling der Benutzerinteraktionen, wie zum Beispiel das Drücken eines Buttons.

- **Controllers**
  - **Router**: Navigationsschnittstelle; Erstellt entsprechend der aufzurufenden Seite die jeweilige View-Instanzen
  - **Events**: Registriert benötigte JavaScript- und PhoneGap-Events
  - **Notifications**: Hub für Notifikationen / Messages, welche dem Benutzer angezeigt werden können

3rd Party JavaScript Libraries

Der pillow Client setzt auf verschiedenen JavaScript Frameworks auf:

- **Backbone.js**
  Kern-Library, welche die Struktur der Architektur vorgibt und Funktionalitäten zur Anzeige von Inhalten in JavaScript Apps bereitstellt.

- **Zepto**
  jQuery-Pendant, jedoch auf Mobile-Applikationen ausgerichtet. Erlaubt einem die Manipulation von DOM-Elementen und stellt CSS3-Animationen als JavaScript-Funktionen zur Verfügung.

- **Modernizr**
  Library, welche die HTML5- und CSS3-Unterstützung des Browsers überprüft und entsprechende Weichen bereitstellt.

- **Junior**
  Vereint alle oben genannten Libraries und stellt mithilfe des CSS-Frameworks Ratchet ein Framework zur Entwicklung von Mobile-Applikationen zur Verfügung.

- **Lo-Dash**
  Low-level JavaScript Utility Library, welche verschiedene Methoden zur Unterstützung zur Verfügung stellt. Wird überdies im pillow Client auch als Templating-Engine benutzt.

- **Jasmine**
  JavaScript Testing-Framework, dessen Einsatz unter D.8.4 „Testing“ genauer umschrieben wird.

3rd Party PhoneGap Plugins

Um die Funktionalität von PhoneGap um die Einsatzmöglichkeit von Push Notifications zu erweitern, werden folgende Plugins eingesetzt:

- **PushNotification**
  Push Notifications Plugin für iOS und Android, welches eine JavaScript Schnittstelle zum Abrufen dieser Art von Nachrichten zur Verfügung stellt.

- **OpenUDID**
  OpenUDID erlaubt es dem Plugin„PushNotifications “, eine App-spezifische ID zu generieren. Dies ist notwendig, da Apple keine Apps, welche auf die Device ID zugreifen, in ihrem App Store zulässt.

5.4.2. Klassendiagramm

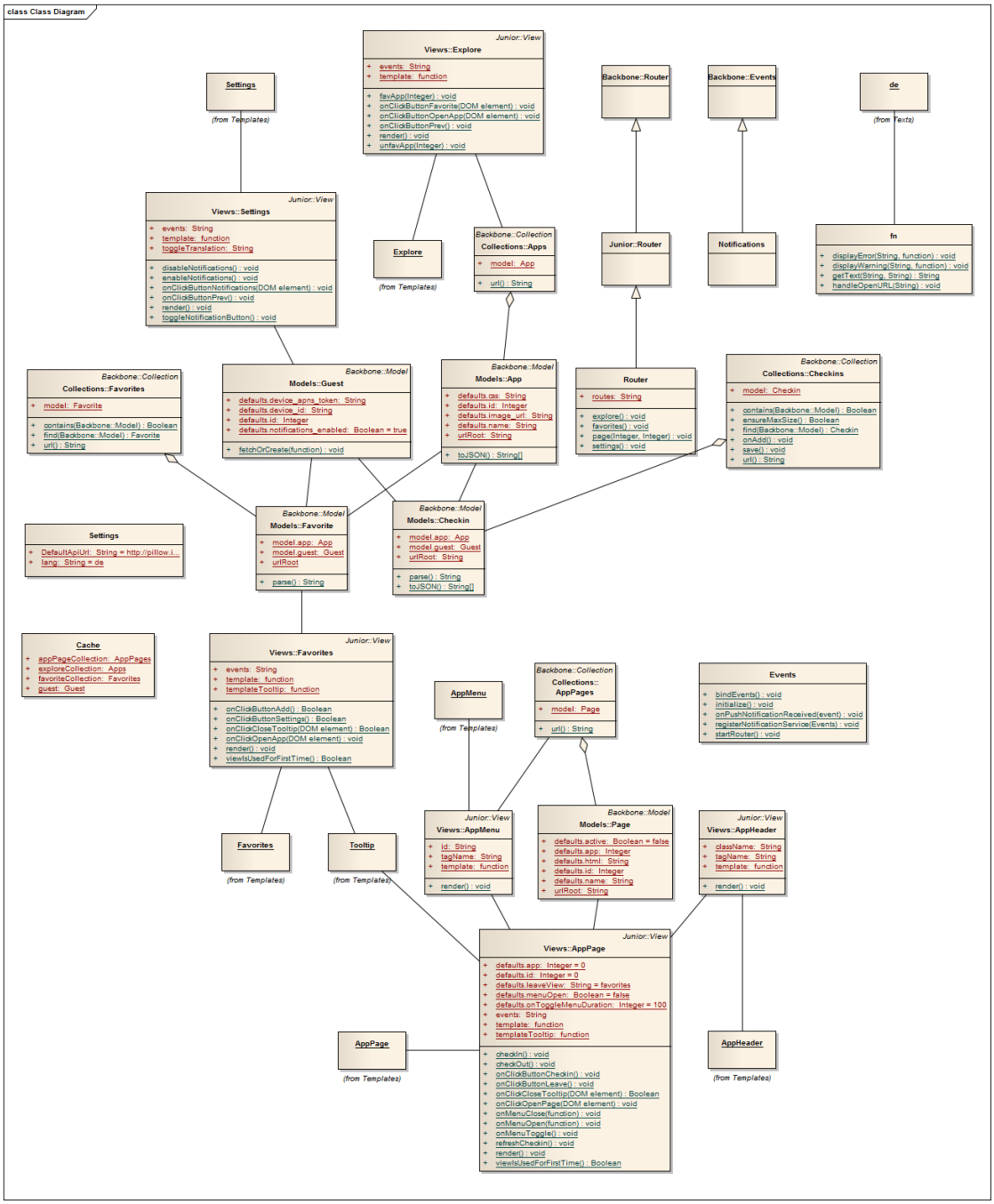

Abbildung 5.12.: App: Klassendiagramm

Der Aufbau der pillow App ist zum grössten Teil durch die Vorgaben von Backbone.js geprägt. Nebst diesen Standard-Vorgaben wurden auch noch einige „Best Practice“ Ansätze aus dem Umfeld der Backbone.js-Community umgesetzt. So wurde beispielsweise die View „AppPage“ mit ihren Sub Views „AppMenu“ und „AppHeader“ nach dem Pattern Sub views der Backbone Patterns [Cru] implementiert.

Eine explizite Beschreibung aller Klassen ist zudem der Code Dokumentation zu entnehmen.

5.4.3. Paketdiagramm

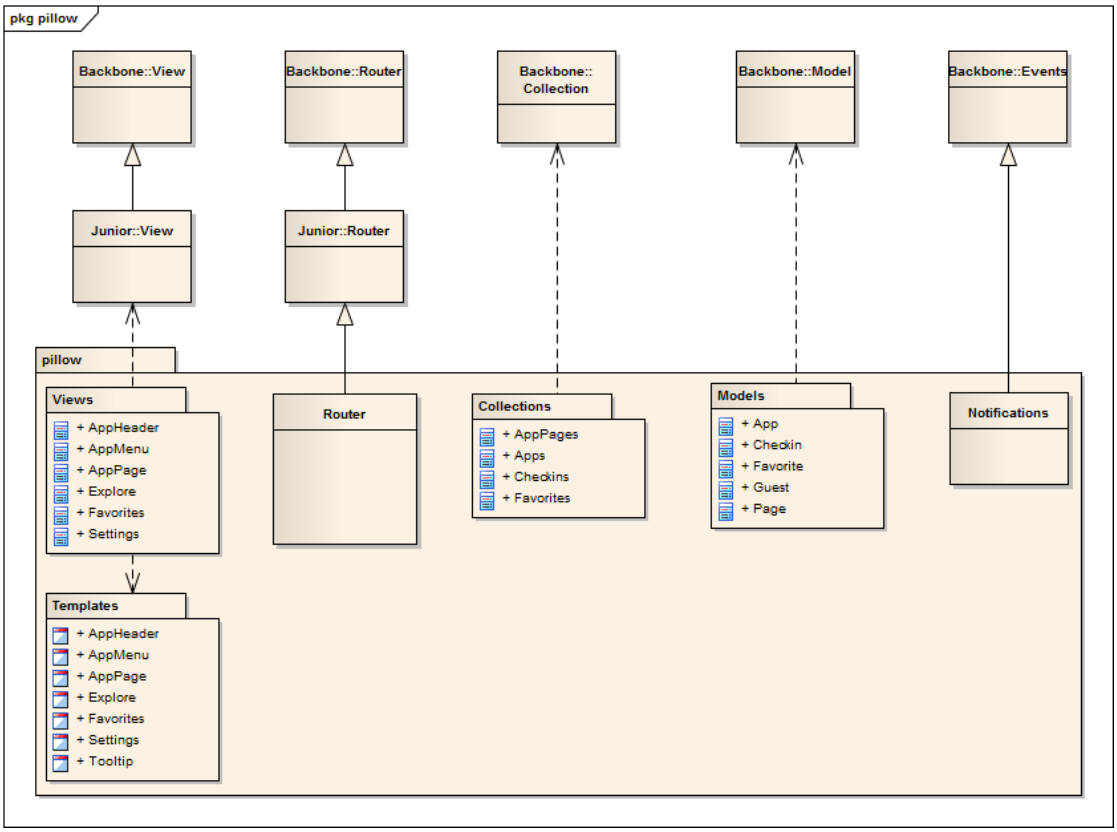

Abbildung 5.13.: App: Paketdiagramm

Anhand des obenstehenden Paketdiagramms können die wichtigsten Abhängigkeiten innerhalb des pillow Clients wohl am besten illustriert werden. Wie man sieht, wurden die jeweiligen gleichartigen Klassen in Paketen (beziehungsweise JavaScript Namespaces) zusammengefasst, um eine klare Struktur sicherstellen zu können.
Weiter ist ersichtlich, dass Junior die Funktionalitäten von Backbone.js dahingehend erweitert, als dass es die Page Transitions sowie die Touch-Events überschreibt und so den „Look and Feel“ einer nativen App zu vermitteln.

Bei der Strukturierung der einzelnen Namespaces wurde auf die Namespace Konvention der Backbone Patterns [Cru] zurückgegriffen, wobei diese um den Namespace „Templates “ erweitert wurde.

Die JavaSript-Files wurden dementsprechend gemäss der File Naming Konvention der Backbone Patterns [Cru] organisiert und wiederum um das Verzeichnis „templates “ erweitert:

```
├── app.js
├── backbone.pillow.js  # Backbone Patching
├── cache.js  # Globales Caching Objekt
├── collections
│   ├── apppages.js
│   ├── apps.js
│   ├── checkins.js
│   └── favorites.js
├── events.js
├── functions.js
├── lang
│   └── de.js
├── models
│   ├── app.js
│   ├── checkin.js
│   ├── favorite.js
│   ├── guest.js
│   └── page.js
├── notifications.js  # Globales Notifikationen-Handling
├── router.js
├── settings.js
├── setup.js
├── templates
│   ├── appheader.js
│   ├── appmenu.js
│   ├── apppage.js
│   ├── explore.js
│   ├── favorites.js
│   ├── settings.js
│   └── tooltip.js
└── views
    ├── appheader.js
    ├── appmenu.js
    ├── apppage.js
    ├── explore.js
    ├── favorites.js
```

```
└── settings.js
```

5.5. Internes Design Service

5.5.1. Komponentenmodell

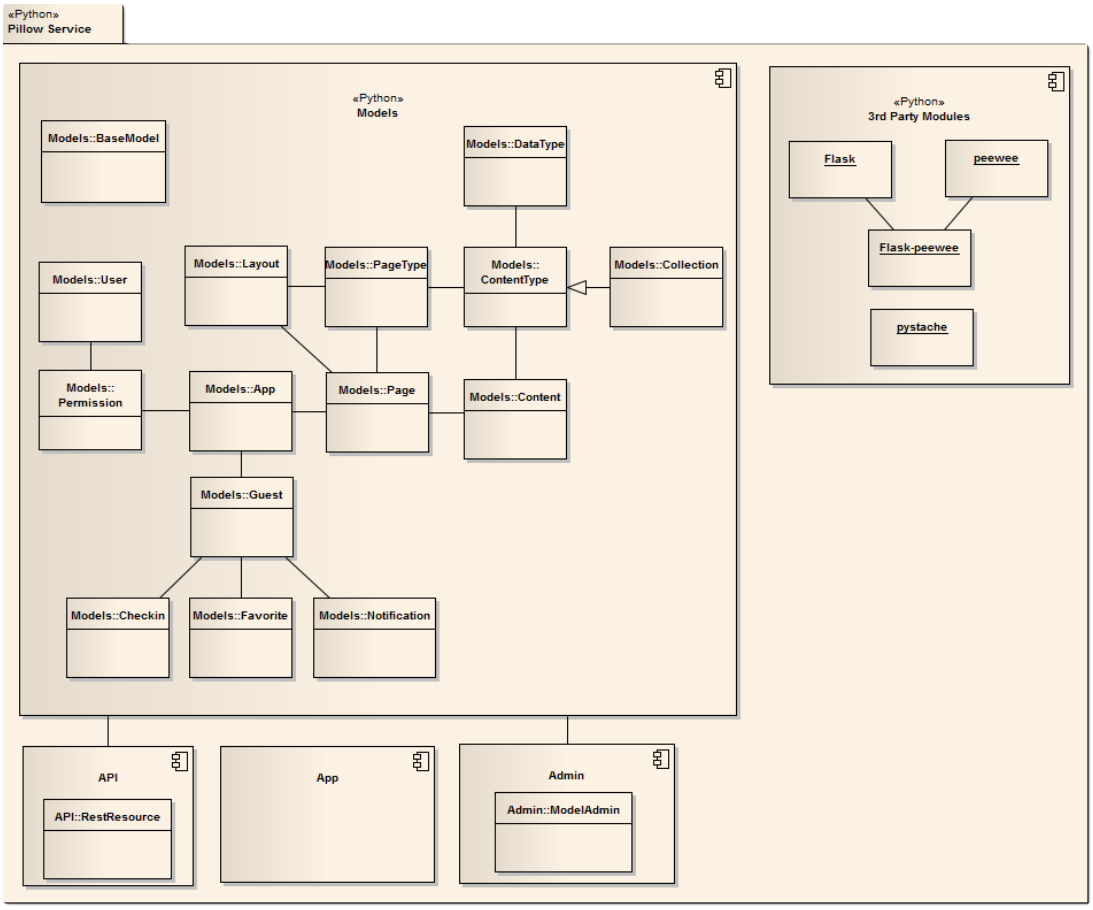

Abbildung 5.14.: Service: Komponentenmodell

Der pillow Service umfasst verschiedene Komponenten, auf welche wir hier kurz eingehen.

Models

Die Models bilden die Grundlage für den Service. Sie bilden die Datencontainer und bilden die Abbildung der Datenbank.

API
In der API werden (Rest)Resourcen basierend auf Models definiert, welche definieren, was via API abgerufen bzw. modifziert werden kann.

Admin
Für das ganze Admin GUI ist die Admin Komponente verantwortlich. Diese definiert AdminModels basierend auf Models und verwaltet alle Zugriffe via Web Interface inklusive Prüfung derer Authentifikation.

App
Die App stellt die zentrale Anlaufstelle für den Service da. Sie nimmt alle Requests, welche an den Service gesendet werden entgegen und leitet dieses an die richtige stellen weiter. So wird beispielsweise ein Request, welche an die Route „/api/*“ gerichtet werden, an die API Komponente weitergeleitet.

3rd Party Python Modules
Der pillow Service setzt auf verschiedenen Python Modulen auf:

- **Flask**
  Flask ist ein Microframework für Python.
- **peewee**
  peewee ist ein schmales, expressives Object-Relational Mapping Framework mit Support für PostgreSQL, MySQL und SQLite geschrieben in Python.
- **flask-peewee**
  flask-peewee ist eine Modul zur Integration von Flask und peewee.
- **pystache**
  Pystache ist eine Python Implementation von Mustache. Mustache ist eine logikfreie Template Sprache. Pystache dient als Grundlage für das Rendering.

5.5.2. Klassendiagramm

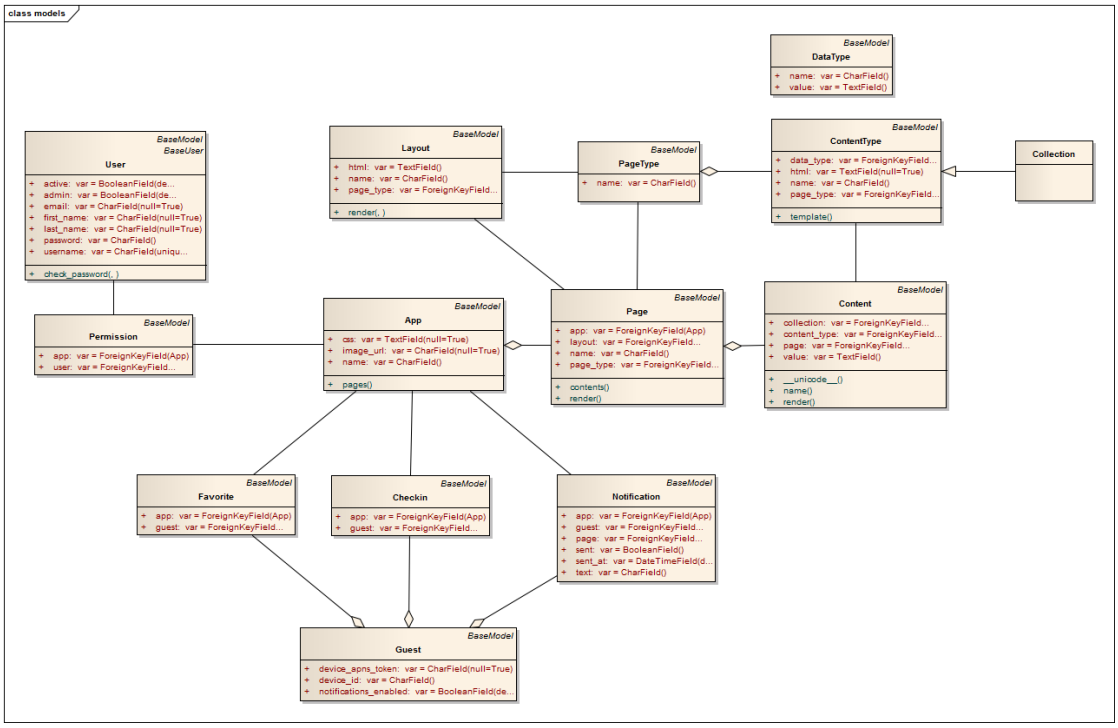

Abbildung 5.15.: Service: Klassendiagramm

Das Klassendiagramm leitet sich stark vom Domainmodell und dem Entity Relationship Modell ab. Alle Klassen erben von dem BaseModell, welche die somit die Basis aller anderen Klassen bildet.

5.6. Design Datenbank

5.6.1. Entity Relationship Modell

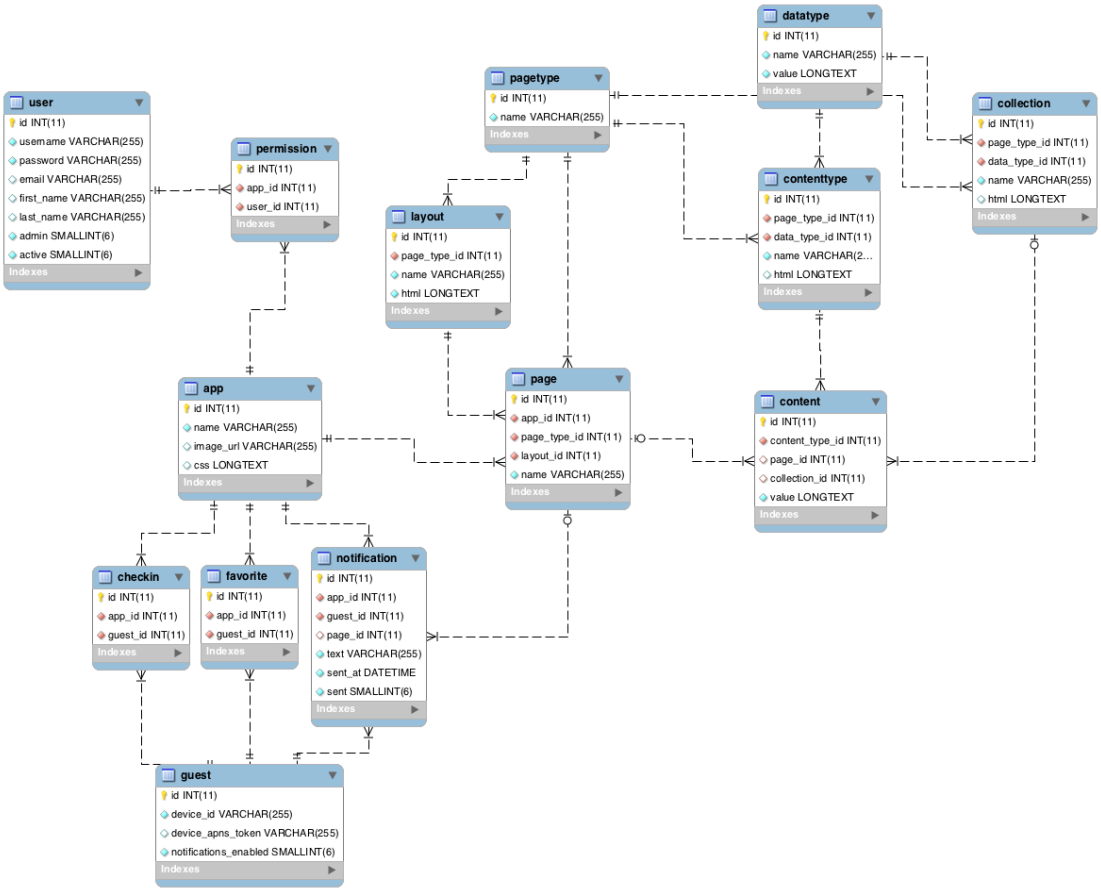

Abbildung 5.16.: Architektur Übersicht

Gast

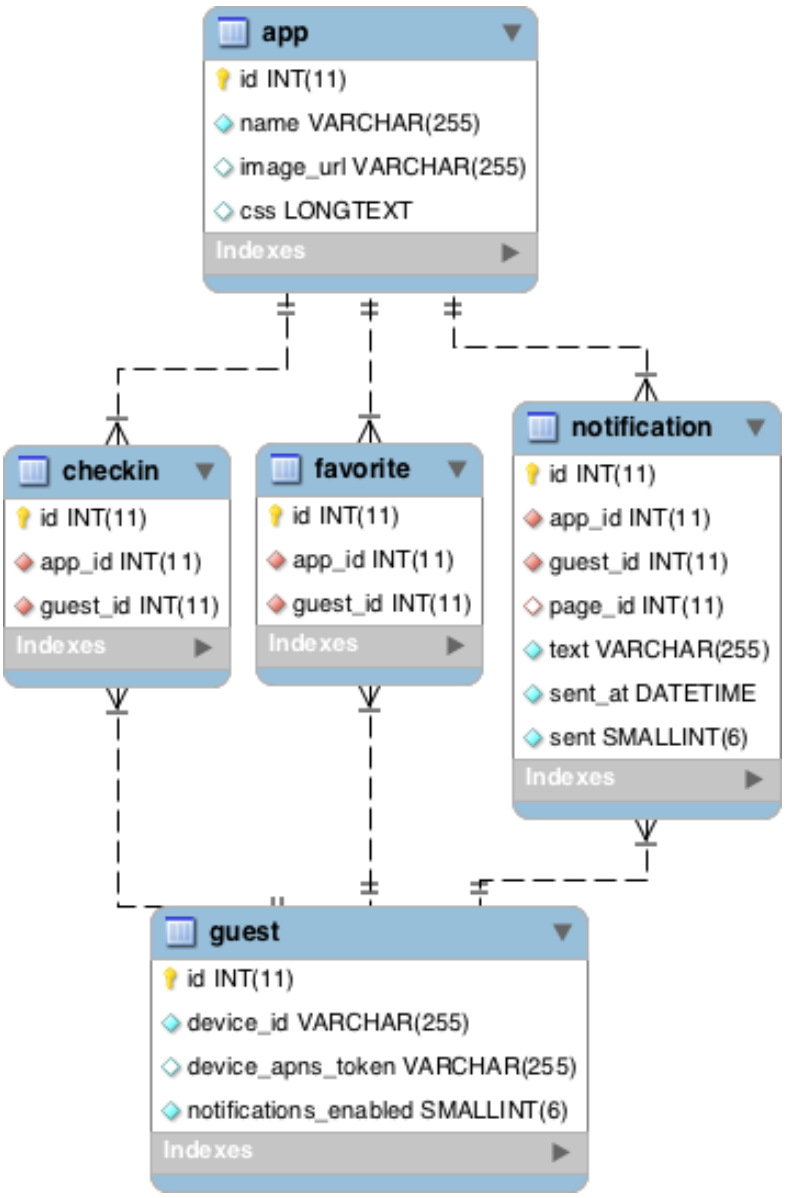

Abbildung 5.17.: ERM: Gast mit Checkins, Favoriten und Notifications

Zwischen dem Gast und der App gibt es 3 wichtige Verknüpfungen: Checkins, Favoriten und Notifikationen. Ein Gast bzw. eine App kann beliebig viele davon haben/erhalten.

Benutzer

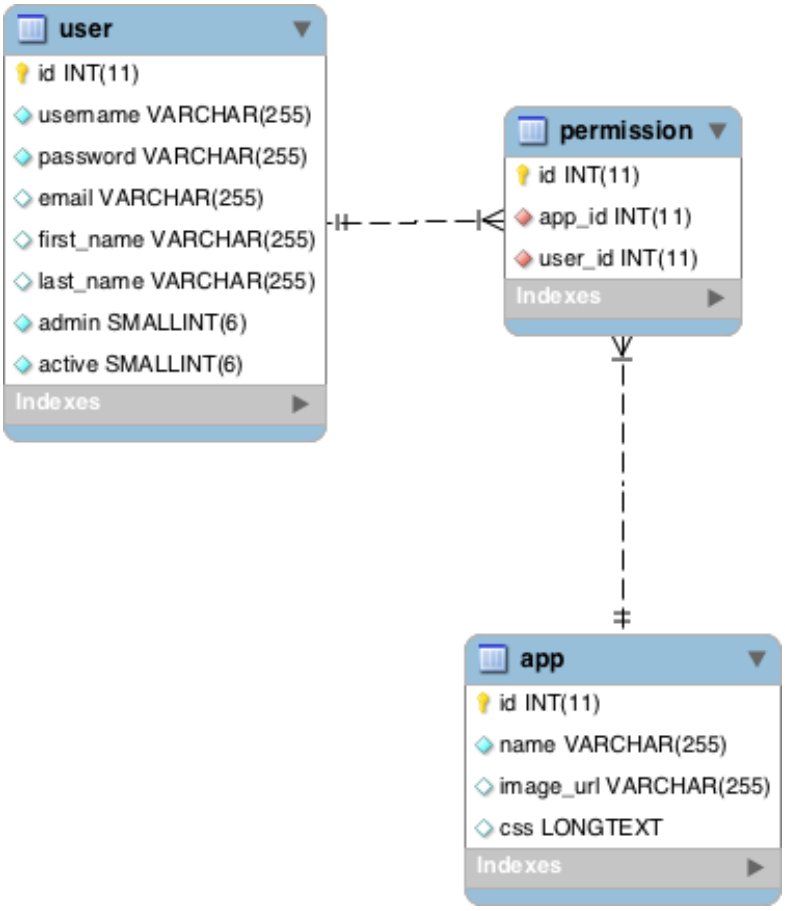

Abbildung 5.18.: ERM: Gast mit Berechtigungen in Apps

Der Benutzer hat Berechtigungen an einer oder mehreren Apps.

Layout App

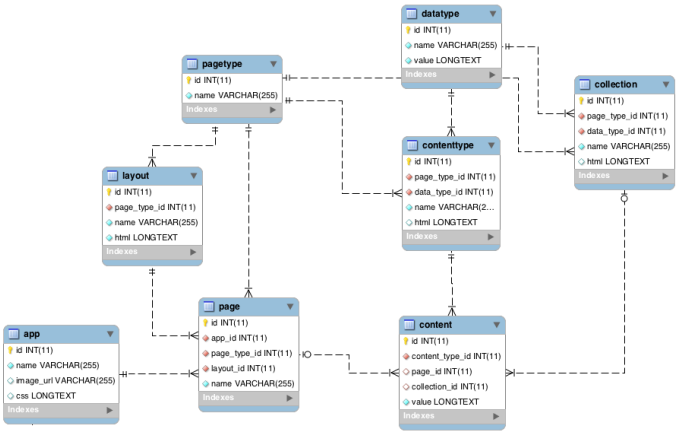

Abbildung 5.19.: ERM: Layout der App

Dieser Bereich dient der Generierung der App inkl. Template Rendering. Eine App besitzt beliebig viele Seiten (Page) mit Inhalten (Content). Diese Seiten und Inhalte sind entsprechend als Typen abstrahiert um die Generalität zu gewährleisten und Redundanz zu vermeiden.

5.7. Sequenzdiagramme

5.7.1. Startup Prozess

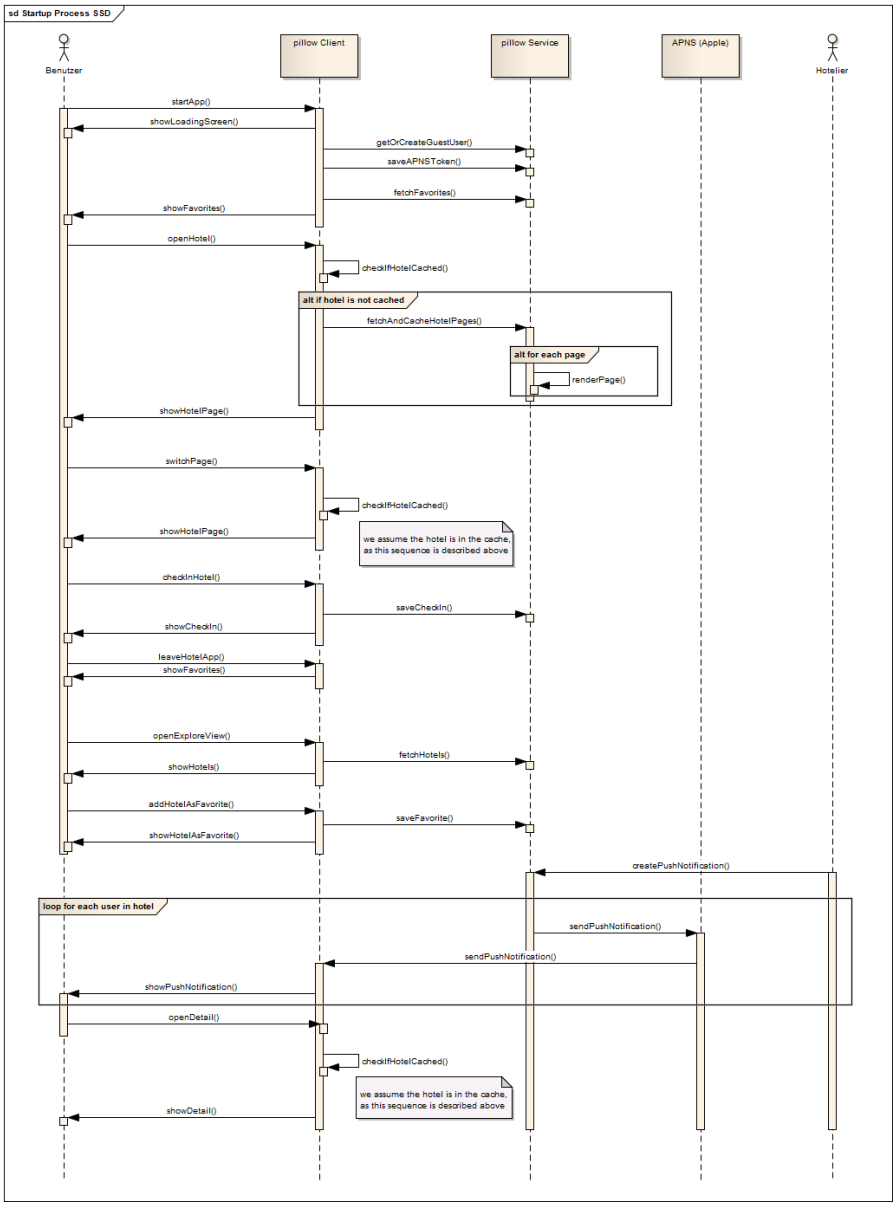

Abbildung 5.20.: Sequenzdiagramm: Startup Prozess

Im Sequenzdiagramm wird gezeigt, wie sich das System beim Start des pillow Clients verhält.

5.7.2. Page Rendering

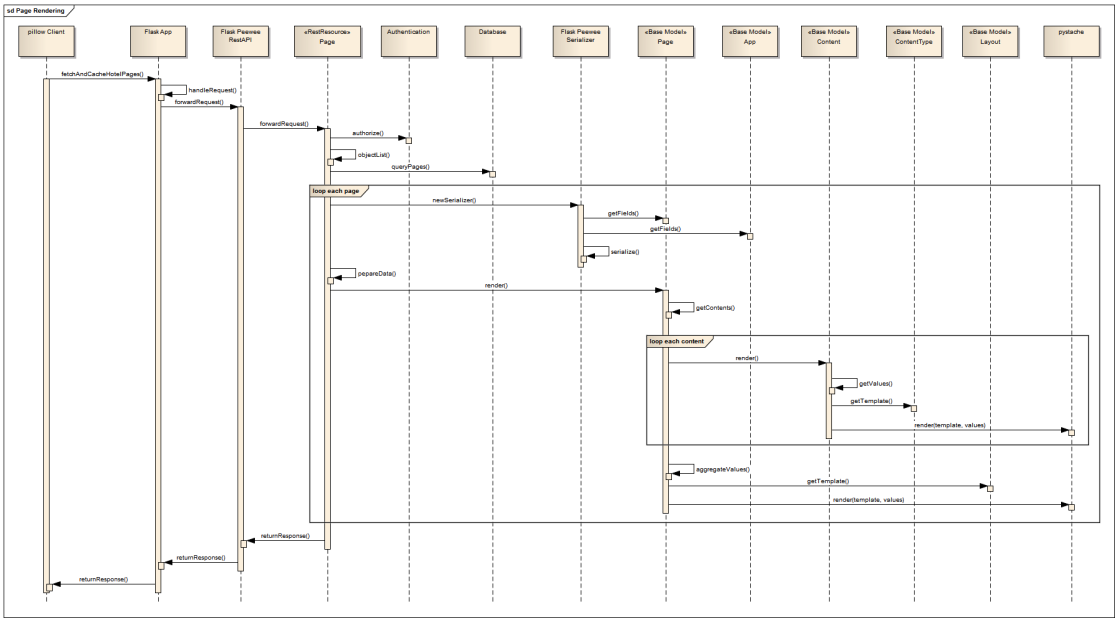

Abbildung 5.21.: Sequenzdiagramm: Page Rendering

Im Sequenzdiagramm wird gezeigt, wie sich das System bei der Abfrage einer Hotel Seite verhält und eine solche Seite im System gerendered wird.

5.8. API

Die API ist REST (RESTful) aufgebaut. Es wird sich an die gängigen HTTP Standards [Irv+] gehalten. Folgende Resourcen sind via API verfügbar:

5.8.1. App

/app

Listet alle Apps auf.

Methoden

- GET

Filter

- Keine

Beispiel


```
{
  "meta": {
    "model": "app",
    "next": "",
    "page": 1,
    "previous": ""
  },
  "objects": [
    {
      "image_url": "http://lgsh.ch/pillow/images/weissbad/
         logo_hofweissbad.jpg",
      "id": 1,
      "css": "",
      "name": "Hof Weissbad"
    },
    {
      "image_url": "http://matthiol.ch/wp-content/themes/
         starkers/images/logo.png",
      "id": 3,
      "css": "",
      "name": "Hotel Matthiol"
    }
```

```
  ]
}
```

/app/1
Liefert die App mit der ID 1 zurück.

Methoden

- GET

Filter

- Keine

Beispiel

```
{
  "image_url": "http://lgsh.ch/pillow/images/weissbad/
     logo_hofweissbad.jpg",
  "id": 1,
  "css": "",
  "name": "Hof Weissbad"
}
```

5.8.2. Page

/page
Listet alle Pages auf.

Methoden

- GET

Filter

- app / Beispiel: app=1

Beispiel


```
{
  "meta": {
    "model": "page",
    "next": "",
    "page": 1,
    "previous": ""
  },
  "objects": [
    {
      "layout": 1,
      "name": "Startseite",
      "app": {
        "image_url": "http://lgsh.ch/pillow/images/weissbad/
           logo_hofweissbad.jpg",
        "id": 1,
        "css": "",
        "name": "Hof Weissbad"
      },
      "html": "<h1>Hotel Hof Weissbad</h1><img src=\"http://
         lgsh.ch/pillow/images/weissbad/front_image.jpg\" /><p>
         Das Appenzellerland umgarnt ...</p>",
      "page_type": 1,
      "id": 1
    },
    {
      "layout": 3,
      "name": "Zimmer",
```

```
      "app": {
        "image_url": "http://lgsh.ch/pillow/images/weissbad/
           logo_hofweissbad.jpg",
        "id": 1,
        "css": "",
        "name": "Hof Weissbad"
      },
      "html": "<h1>Zimmer</h1><p><img src='http://lgsh.ch/
         pillow/images/weissbad/doppelzimmer1.jpg'>In allen 82
         Zimmern und 5 Junior-Suiten ...</p>",
      "page_type": 3,
      "id": 2
    }
  ]
}
```


/page/1
Liefert die Page mit der ID 1 zurück.

Methoden

- GET

Filter

- Keine

Beispiel


```
{
  "layout": 1,
  "name": "Startseite",
  "app": {
    "image_url": "http://lgsh.ch/pillow/images/weissbad/
       logo_hofweissbad.jpg",
    "id": 1,
    "css": "",
    "name": "Hof Weissbad"
  },
  "html": "<h1>Hotel Hof Weissbad</h1><img src=\"http://lgsh.ch
     /pillow/images/weissbad/front_image.jpg\" /><p>Das
     Appenzellerland umgarnt ...</p>",
```

```
  "page_type": 1,
  "id": 1
}
```

5.8.3. Guest

/guest
Liefert den Guest mit der enstprechenden Device ID zurück.

Methoden

- GET

Filter (notwendig)

- device_id / Beispiel: device_id=123

Beispiel


```
{
  "meta": {
    "model": "guest",
    "next": "",
    "page": 1,
    "previous": ""
  },
  "objects": [
    {
      "notifications_enabled": true,
      "device_apns_token": null,
      "id": 13,
      "device_id": "123"
    }
  ]
}
```


/guest
Erstellt einen Gast

Methoden

- POST

Filter

- Keine

Beispiel (Request)

```
{
  "notifications_enabled": true,
  "device_apns_token": null,
  "device_id": "123"
}
```

Beispiel (Response)

```
{
  "notifications_enabled": true,
  "device_apns_token": null,
  "id": 13,
  "device_id": "123"
}
```

5.8.4. Favorit

/favorite
Erstellt einen Favorit für eine App / Guest

Methoden

- POST

Filter

- Keine

Beispiel (Request)

```
{
  "guest_id": 1,
  "app_id: 1
}
```

Beispiel (Response)

```
{
  "app": {
    "image_url": "http://lgsh.ch/pillow/images/weissbad/
       logo_hofweissbad.jpg",
    "id": 1,
    "css": "",
    "name": "Hof Weissbad"
  },
  "id": 37,
  "guest": {
    "notifications_enabled": true,
    "device_apns_token": "0af3850bf057e4fc7ce8fc1658f54a5e8f7fc
       2db833f587eb74be36e81e0bafa",
    "id": 1,
    "device_id": "4263c90670bedc91c72094cb53109aac"
  }
}
```

/favorite/1

Löscht einen Favorit für eine App / Guest

Methoden

- DELETE

Filter

- Keine

Beispiel (Request)

```
{}
```

Beispiel (Response)

```
{
  "deleted": 1
}
```

5.8.5. Checkin

/checkin
Erstellt einen Checkin für eine App / Guest

Methoden

- POST

Filter

- Keine

Beispiel (Request)

```
{
  "guest_id": 1,
  "app_id: 1
}
```

Beispiel (Response)

```
{
  "app": {
    "image_url": "http://lgsh.ch/pillow/images/weissbad/
       logo_hofweissbad.jpg",
    "id": 1,
    "css": "",
    "name": "Hof Weissbad"
  },
  "id": 42,
  "guest": {
    "notifications_enabled": true,
    "device_apns_token": "0af3850bf057e4fc7ce8fc1658f54a5e8f7fc
       2db833f587eb74be36e81e0bafa",
    "id": 1,
    "device_id": "4263c90670bedc91c72094cb53109aac"
  }
}
```

/checkin/1

Löscht einen Checkin für eine App / Guest

Methoden

- DELETE

Filter

- Keine

Beispiel (Request)

```
{}
```

Beispiel (Response)

```
{
  "deleted": 1
}
```

Anhang A **Abbildungen, Tabellen & Quellcodes**

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

Anhang B Literatur

[All] Subbu Allamaraju. *RESTful Web Services Cookbook*. O'Reilly Media / Yahoo Press. URL: http://shop.oreilly.com/product/9780596801694.do (besucht am 30. 05. 2013).

[Cru] Rico Sta Cruz. *Backbone.js Patterns*. URL: http://ricostacruz.com/backbone-patterns/ (besucht am 22. 05. 2013).

[FF] Christopher Ferris und Joel Farrell. *What Are Web Services?* URL: http://delivery.acm.org/10.1145/780000/777335/p31-ferris.pdf (besucht am 30. 05. 2013).

[FT] Roy T. Fielding und Richard N. Taylor. *Principled Design of the Modern Web Architecture*. URL: http://www.ics.uci.edu/~taylor/documents/2002-REST-TOIT.pdf (besucht am 30. 05. 2013).

[Irv+] R. Fielding UC Irvine u. a. *Hypertext Transfer Protocol – HTTP/1.1*. URL: http://www.w3.org/Protocols/rfc2616/rfc2616.html (besucht am 30. 05. 2013).

[Pic] Roman Pichler. *Scrum - Agiles Projektmanagement erfolgreich einsetzen*. dpunkt.verlag. URL: http://www.dpunkt.de/buecher/2682/scrum-.html (besucht am 30. 05. 2013).

[Sim] Kai Simons. *Story Points verständlich erklärt*. URL: http://ksimons.de/2011/06/story-points-verstandlich-erklart/ (besucht am 06. 03. 2013).

Anhang C Glossar

Android

Android ist ein Open Source Betriebssystem für mobile Geräte, das von der Open Handset Alliance (Hauptmitglied: Google) entwickelt wird. 26, 77

Apache JMeter

Performance testing für statische und dynamische Resourcen. - Apache JMeter Webseite 22, 23, 77

ApacheBench

Apache HTTP server benchmarking. - Apache Webseite 22, 77

APNS

Apple Push Notification Service (APNS) ist ein Service von Apple, welcher es erlaubt via Push Technologie Nachrichten an iOS Devices zu senden. 32, 77

Backbone.js

JavaScript MVC-Framework, welches durch seine Einfachheit überzeugt. - Backbone.js Webseite 49, 52, 53, 77, 130

CMS

Content Management System 32, 77

Content Management System

Content Management System (CMS) bezeichnet ein System, zur gemeinsamen Erstellung, Verwaltung und Organisation von Inhalten. 77

DOM

Document Object Model (DOM) ist eine Spezifikation einer Schnittstelle für den Zugriff auf HTML- oder XML-Dokumente. 35, 77

git

Git iste ein Versionsverwaltungs-System. - git Webseite 77, 92

GitHub

GitHub ist ein webbasierter Hosting-Dienst für Software-Entwicklungsprojekte. Er verwendet namensgebenderweise das Versionsverwaltungs-System Git. - Github Webseite 77, 85, 92

gunicorn

Gunicorn „Green Unicorn“ist ein Python WSGI HTTP Server für UNIX. - Gunicorn Webseite 77, 121

HTTP

Das Hypertext Transfer Protocol (HTTP) ist ein Protokoll zur Übertragung von Daten über ein Netzwerk. Es wird hauptsächlich eingesetzt, um Webseiten aus dem World Wide Web (WWW) in einen Webbrowser zu laden. - Wikipedia Artikel - [Irv+] 64, 77

Jasmine

Behavior-Driven Test Framework um JavaScript Code zu testen. - Jasmine Webseite 50, 77, 92

Jenkins

Jenkins ist ein erweiterbares, webbasiertes System zur kontinuierlichen Integration. - Jenkins Webseite 77, 94

Jira

Jira ist eine von der Firma Atlassian entwickelte webbasierte Anwendung zur Fehlerverwaltung, Problembehandlung und operativem Projektmanagement. - JIRA Webseite 77, 85, 91

jQuery

jQuery ist eine freie, umfangreiche JavaScript-Bibliothek, welche komfortable Funktionen zur DOM-Manipulation und -Navigation zur Verfügung stellt. - jQuery Webseite 49, 77

JSON

JSON (JavaScript Object Notation) ist ein kompaktes Datenformat in für Mensch und Maschine einfach lesbarer Textform zum Zweck des Datenaustauschs zwischen Anwendungen. 27, 77

Locomotive

Locomotive ist ein freies CMS, welches auf dem Framework Ruby und Rails aufsetzt. 34, 77

Mocking

Mock-Objekte implementieren die Schnittstellen, über die das zu testende Objekt auf seine Umgebung zugreift. Sie stellen sicher, dass die erwarteten Methodenaufrufe vollständig, mit den korrekten Parametern und in der erwarteten Reihenfolge durchgeführt werden. Das Mock-Objekt liefert keine Echtdaten zurück, sondern vorher zum Testfall passend festgelegte Werte. Das Mock-Objekt kann somit dazu verwendet werden ein bestimmtes Verhalten nachzustellen. 77, 93

nginx

Nginx ist ein gratis, open-source, high-performance HTTP server und reverse proxy. - Nginx Webseite 77, 122

PhoneGap

PhoneGap ist ein frei verfügbares Framework, um Apps zu entwickeln, die auf mehreren Mobilplatformen lauffähig sind. 26, 32, 33, 50, 77

pillow

pillow ist der Produktname der vorliegenden Semesterarbeit. 77

QR Code

QR Code (Quick Response Code) ist ein zweidimensionaler Code. 16, 21, 77

REST

REST (Representational State Transfer) ist ein Programmierparadigma für Webanwendungen. [FF] [All] [FT] 27, 34, 77, 93, 122

RESTful

REST 64, 77

Scrum

Scrum (englisch für Gedränge) ist ein Vorgehensmodell der Softwaretechnik. Der Ansatz von Scrum ist empirisch, inkrementell und iterativ. Er beruht auf der Ansicht, dass die meisten modernen Entwicklungsprojekte zu komplex sind, um durchgängig planbar zu sein. - Wikipedia Artikel 77

Sencha ExtJS

Ext JS ist ein clientseitiges JavaScript- bzw. Ajax-Framework für interaktive Webanwendungen, das für Open-Source-Projekte unter der GPL, für andere Projekte unter kommerziellen Lizenzen erhältlich ist. In erster Linie bietet Ext JS eine umfangreiche Sammlung von Steuerelementen. - Sencha ExtJS Webseite 35, 77

Sencha Touch

Sencha Touch ist ein HTML5-Framework zur Entwicklung mobiler Anwendungen. - Sencha Touch Webseite 32, 77

Story Points

Story Points repräsentieren Komplexität einer User Story. 77, 84, 130

Windows Phone

Windows Phone ist ein Betriebssystem für mobile Geräte, das von Microsoft entwickelt wird. 26, 77

WordPress

WordPress ist ein freies CMS, welches auf der Skriptsprache PHP basiert. 33, 77

Anhang D **Projektplanung**

D.1. Projektorganisation

D.1.1. Mitglieder

Name	E-Mail	Aufgaben und Verantwortungen
Moreno Feltscher	mfeltsch@hsr.ch	Projektleitung, Entwicklung, Dokumentation, Grafische Elemente
Peter Manser	pmanser@hsr.ch	Entwicklung, Dokumentation, Server, Kundenkontakt

Tabelle D.1.: Projektplanung: Projektmitglieder

Externe Schnittstellen

Name	E-Mail	Aufgaben und Verantwortungen
Prof. Dr. Markus Stolze	mstolze@hsr.ch	Projektbetreuung
Mischa Trecco	mtrecco@hsr.ch	Code Reviews

Tabelle D.2.: Projektplanung: Externe Schnittstellen

D.2. Projektmodell SCRUM

Das Projekt wird nach einem leicht angepassten **Scrum-Vorgehensmodell** durchgeführt. Wir haben das angestrebte Modell dahingehend angepasst, dass doch noch gewisse Meilensteine definiert werden können. Diese terminbedingten Informationen sieht ein reines Vorgehen nach Scrum nicht vor, wird aber von unserer Arbeit gemäss vordefinierter Termine gefordert.
Erschwerend hinzu kommt, dass die kleine Projektteamgrösse von zwei Personen ein Scrum-Vorgehen mit den festgelegten Projekt-Rollen wie Scrum Master oder Product Owner praktisch verunmöglicht. Wir haben uns deshalb darauf geeinigt, die beiden Rollen selbst zu übernehmen. Faktisch heisst dies, dass wir die User Stories selbst erfassen werden und diese nach bestem Wissen aufgrund der vom Projektplan geforderten Produkte selbst priorisieren. Hier werden auch die Interessen des Kunden berücksichtigt.

Um unsere Scrum Erfahrungen und Praktiken zu verbessern haben wir folgendes Buch beigezogen: [Pic].

D.2.1. Sprints

Das Projekt wurde in Sprints aufgeteilt, wobei am Ende einzelner Sprints jeweils sofern möglich ein Meilenstein gesetzt werden soll. Die Sprints sind unter D.5 „Plan“ genauer beschrieben.

Sprint Planning

Am Anfang jedes Sprints wurde jeweils ein Sprint Planning durchgeführt.

Anlässlich dieser Plannings wurden die User Stories für den kommenden Sprint definiert und anhand von Story Points gewichtet. Die Vergabe der Anzahl Story Points wurde anhand der Empfehlungen des Artikels „Story Points verständlich erklärt“ [Sim] vorgenommen.

Weiter wurde der abgeschlossene vorhergehende Sprint jeweils anhand einer Retrospektive besprochen und Verbesserungen aufgenommen.

Abbildung D.1.: Retrospektive von Sprint 2

D.3. Projekt Management

Das Projekt und deren User Stories (gemäss Scrum) werden mittels Jira verwaltet. Dies erlaubt uns ideale Tools (wie bspw. Greenhopper mit dem Agile Board) zu verwenden.
URL: http://jira.smartive.ch
Alle Projektbeteiligten haben einen Login erhalten und haben somit vollen Zugriff.

D.4. Versionsverwaltung

Als Versionverwaltung wird der Service von GitHub genutzt. GitHub hat sich bereits bei einigen Projekten bewährt und ist auch für dieses Projekt perfekt geeignet.
URL: https://github.com/petermanser/HSR-Semesterarbeit

D.5. Plan

Das Projekt wurde in sechs Sprints unterteilt. Die Sprints wurden jeweils, wie unten beschrieben, einem Thema unterstellt.

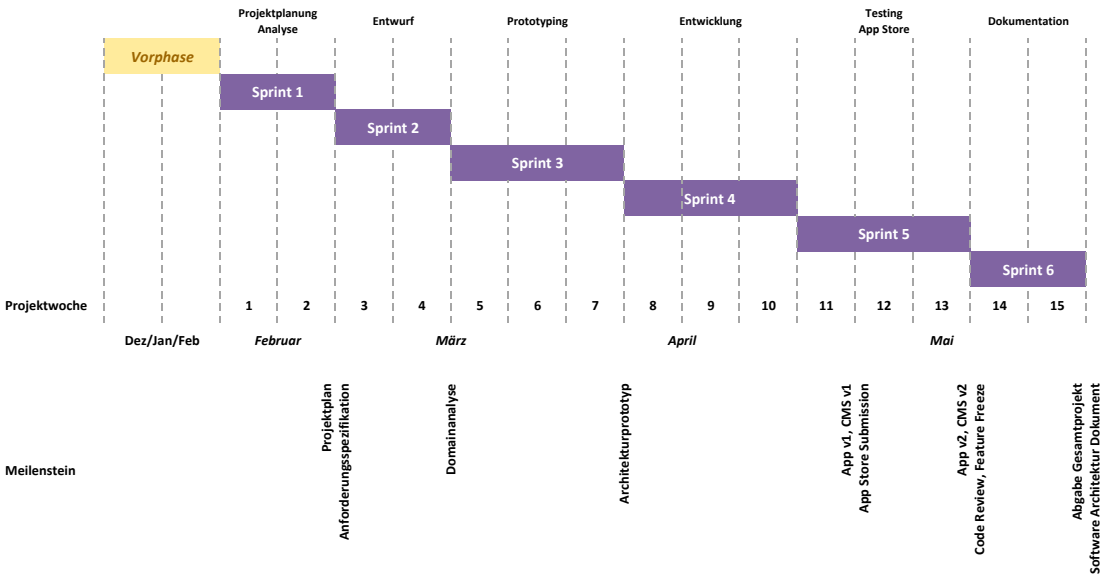

Abbildung D.2.: Projektplan mit sechs Sprints

D.5.1. Vorphase

In der Vorphase wurden verschiedenste Abklärungen angestellt. Es wurde ein Betreuer für die Studienarbeit gefunden, Prof. Dr. Markus Stolze. Zudem hat sich das Hotel Hof

Weissbad als Industriepartner zur Verfügung gestellt, wo wir am 13. / 14. Februar bereits einen Aufenthalt verbringen durften.

D.5.2. Sprint 1 - Projektplanung & Analyse

Ziel des ersten Sprints war die Planung der einzelnen Sprints sowie die Analyse der Anforderungen. Anfänglich waren 7 Sprints zu je 2 Wochen geplant, ein Replanning während der Projektphase war jedoch notwendig, da die Anfertigung des Architekturprototyps 3 anstelle von 2 Wochen in Anspruch nahm. Entsprechend wurden auch Sprints 4 und 5 um je eine Woche verlängert.

Die angefertigten Produkte „Projektplan“ und „Anforderungsspezifikation“ entsprechen dem Meilstein 1 (siehe Meilensteine).

D.5.3. Sprint 2 - Entwurf

In der Phase des zweiten Sprints wurde anhand der Anforderungsspezifikationen eine Domainanalyse angelegt. Desweiteren wurde das externe Design der App mittels Wireframes in einem ersten Entwurf konzipiert und anhand von Paperprototyping-Tests überprüft.

Das angefertigte Dokument „Domainanalyse“ entspricht dem Meilstein 2 (siehe Meilensteine).

D.5.4. Sprint 3 - Prototyping

Der Inhalt dieses Sprints bestand in der Entwicklung eines Architekturprototypen, welcher möglichst alle in Sprint 1 - Projektplanung & Analyse und Sprint 2 - Entwurf zusammengetragenen Risiken beseitigen sollte. Weiter wurden die in Sprint 2 - Entwurf angefertigten Mockups aufgrund des Prototyping Testfeedbacks angepasst.

Das angefertigte Produkt „Architekturprototyp“ entspricht dem Meilstein 3 (siehe Meilensteine).

D.5.5. Sprint 4 - Entwicklung

Gemäss den gewonnen Erkenntnisse in der Entwicklung des Architekturprototyps wurde während dieses Sprints sowohl die App, als auch das CMS sowie die API entwickelt. Ziel war es, Ende des Sprints eine erste Version dieser Produkte fertigzustellen.

Die Entwicklung und das zugrundeliegende Testing der Komponenten dauerte jedoch eine Woche länger, sodass sich der zugrundeliegende Meilstein 4 (siehe Meilensteine) sich entsprechend um eine Woche nach hinten verschob.

D.5.6. Sprint 5 - Testing & App Store

Ziel des fünften Sprints war eine erste Submission der App in den App Store. Dieses Ziel sollte möglichst früh, sprich innerhalb der ersten Woche erreicht werden, damit bei einem Reject seitens Apple noch genug Zeit vorhanden wäre, entsprechende Anpassungen an

der App vorzunehmen. Der Prozess der App Store Submission ist unter D.7.3 „App Store Zulassung“ genauer beschrieben.

Anhand der Rückmeldung von Apple wurde dann die App weiterentwickelt und eine zweite Version in den App Store submitted. Dies entspricht dem Meilenstein 5 (siehe Meilensteine).

Diese Submission stellte sogleich auch den Feature Freeze dar.

D.5.7. Sprint 6 - Dokumentation

Während den letzten beiden Wochen war der Fokus auf die Dokumentation der erstellten Produkte gerichtet. So mussten vielerlei mit Stift und Papier angefertigten Skizzen digitalisiert werden. Weiter wurde ein Abstract, ein Management Summary, ein Projektposter sowie ein Video hergestellt, welcher die Funktionalitäten von pillow aufzeigen soll.

Das angefertigte „Software Architektur Dokument“ sowie die Fertigstellung der Arbeit entspricht dem Meilenstein 6 (siehe Meilensteine).

D.6. Meilensteine

#	Name	Woche
1	Projektplan & Anforderungsspezifikation	2
2	Domainanalyse	4
3	Architekturprototyp	7
4	App v1 & CMS v1 & App Store Submission	11
5	App v2 & CMS v2 & Code Review & Feature Freeze	13
6	Software Architektur Dokument & Abgabe Gesamtprojekt	15

Tabelle D.3.: Projektplanung: Meilensteine

D.7. Risiko Management

D.7.1. Risiken

#	**Name**	**Beschreibung**	**Massnahme**	**Gewichtung (1-10)**
1	Performance	Die App ist zu langsam, sie ruckelt und stockt bei der Benutzung.	Moderne Technologien einsetzen, sauberes/effektives HTML/JS geschrieben. Ständiges Testing.	8

2	Usability CMS	Das CMS ist schlecht bedienbar und stellt sich als nicht intuitiv dar.	Usability Konzepte befolgen, Prototyping/UX Tests	4
3	Usability App	Die App ist schlecht bedienbar, dem Gast ist nicht klar was er damit anstellen kann.	Usability Konzepte befolgen, Prototyping/UX Tests	6
4	App-Links	Die App soll App-Links direkt in der App öffnen. Es ist noch nicht klar inwiefern dies möglich ist.	Andere Apps analysieren (YouTube). Falls diese Möglichkeit nicht besteht, verschlechtert sich die Usability und es muss eine gute Alternativlösung gefunden werden.	5
5	Push Notifications	Der Gast darf sich von den Push Notifications nicht gestört fühlen.	Es muss ein kluges Konzept entwickelt werden um die Benutzer zu benachrichtigen.	6
6	PhoneGap	Native Features benutzen	Das Team ist erfahren mit PhoneGap, jedoch ist noch nicht alles bekannt.	4
7	Geo Location	Akkuverbrauch & Triangulation.	Informationen sammeln, Prototyp bauen.	6
8	Generik CMS	Das CMS wurde nicht oder zu generisch gebaut.	Gute Analyse mittels Daten und Szenarien mit Kunde. Eine gute Abstraktion und Datenmodell entwerfen.	6
9	Skalierung CMS	Das CMS soll auch aus Usability technischer Sicht skalieren, d.h. es soll übersichtlich und einfach bedienbar bleiben.	Usability Tests, Paper Prototyping. Evtl. A/B Testing.	7
10	App Store Zulassung	Es ist unklar, ob die App in den App Store zugelassen wird.	App Store Richtlinien studieren	7

Tabelle D.5.: Projektplanung - Risiko Management: Risikoliste

D.7.2. Umgang mit Risiken

Ziel ist es, alle technischen Risiken so schnell wie möglich zu eliminieren. Spätestens mit dem Meilenstein „Architekturprototyp“ sollten alle technischen Risiken abgeschlossen sein. Jeweils an den Sprint Plannings werden bestehende Risiken im Team neu überprüft und bewertet. Dabei werden beseitigte Risiken dokumentiert und neue Risiken evaluiert und erfasst. Allenfalls eingetretene Risiken werden im Team besprochen und es wird ein Lösungsansatz zur möglichst schnellen Beseitigung erarbeitet.

Risiken nicht im Fokus dieser Arbeit Folgende Risiken sind leglich Produktrisiken, und werden nicht im Rahmen dieser Arbeit behandelt:

- Usability CMS
- Geo Location
- Generik CMS
- Skalierung CMS

D.7.3. Beseitigte Risiken

Performance App Die Performance der App fällt geräteabhängig sehr unterschiedlich aus. Auf dem iPhone 5 läuft die App sauber und flüssig, auf dem iPhone 4 jedoch kommt es teilweise zu einem kurzen Ruckeln. Diese Erkenntnis stellt einen bekannten Tradeoff beim Einsatz von Mobile Apps dar.

Performance, Skalierung und Generik CMS Da der Fokus der Studienarbeit nicht mehr auf der Umsetzung des CMS liegt, werden den im Zusammenhang mit dem CMS stehenden Risiken keine Priorität mehr zugesprochen.

Usability App Der Aufbau der App wurde vorgängig anhand von Wireframes mittels Paper Prototyping (siehe 5.3.2 „Paper-Prototyping“) getestet.

App-Links Das Konzept der App-Links funktioniert und wurde durch einen Prototyp gezeigt. Mehr Details dazu im Ticket HSRSA-102.

Push Notifications Das Konzept der Push Notifikations funktioniert und wurde durch einen Prototyp gezeigt. Es wurde einerseits der Serverteil mittels Ruby implementiert und andererseits auf dem Client diese Notifikationen empfangen und verarbeitet.

PhoneGap Durch die „Push Notifications“ wurde eine native Komponente von PhoneGap verwendet. Dies war zwar problematisch, da die Dokumentation teilweise verstreut und veraltet ist, jedoch konnte das Plugin erfolgreich eingesetzt werden.

App Store Zulassung Das Risiko der App Store Zulassung konnte grösstenteils behoben werden. Es wurden einige Checklisten und Tipps gesammelt und im Ticket HSRSA-91 dokumentiert.

Testprotokoll vom 8. Mai 2013 Wir haben die App ausführlich (gemäss Checkliste) getestet. Folgende Punkte wurden getestet:

#	**Beschreibung**	**Ergebnis**
1	Delete the app from your device, turn off WiFi, off cellular data, now install and test app. Does it work properly (as much as it can without Internet)? Does it at least tell the user that a network connection is required (if it is) or does it crash?	Okay
2	If you use CLLocationManager: Delete the app, fresh install and run, but do not allow app to have Location Data. Does the app behave well or does it crash? Does it at least tell the user that it can't run without location data (if that is a requirement)? Does it work on an iPod Touch that does all geo location using WiFi only?	Nicht verwendet
3	Run the app in the simulator and for each view controller do the following steps: (a) From the iPhone Simulator menu select „Hardware" –> „Simulate Memory Warning" , (b) now navigate around your app to other view controllers and see if everything is working, (c) repeat test for another view controller.	Okay
4	If you support older firmware (ie: iOS 3.1.3), install your app on a device running 3.1.3 and test it there (if you don't have one, use the 3.2 simulator).	Okay
5	Start your app while on a phone call or when Personal Hotspot is active. Are all the screen layouts correct (the status bar is 40px high instead of 20)? Did the bottom 20px of the view get pushed off the screen or did it resize correctly?	Kein Footer Menü. Alles okay
6	Accept a phone call while in your app, does it resign active and resume properly? Do sounds from your app stop playing while in the phone call?	Okay
7	Start your app while playing music, does the music continue to play? Do your sounds mix properly or fade the music appropriately?	Okay
6	Test performance on a slower devices with limited RAM such as: iPhone 3G (128MB RAM, 412Mhz CPU) or iPod Touch (1st or 2nd gen).	Konnten wir nicht testen

7	Run the Clang static analyzer and fix (or at least understand) every warning.	Gemacht. Hier vertrauen wir auf PhoneGap
8	Make sure NSZombiesEnabled is NO in the environment variables (caution: not sure if this is still a problem)	Hier vertrauen wir auf PhoneGap

Tabelle D.7.: Projektplanung - Risiko Management: Checkliste App Submission

Aktualisierung vom 8. Mai 2013 Die App wurde von Apple abgelehnt. Das native Notification Plugin für PhoneGap hat auf die Device UUID zugegriffen, was gemäss Apple Richtlinien nicht (mehr) erlaubt ist. Wir haben diesen Fehler sofort korrigiert und die App nochmals submitted.

Aktualisierung vom 13. Mai 2013 Die App wurde von Apple abgelehnt, als Begründung wurden angegeben, dass die „Check In“ Funkiton bei Hotels nicht erklärt wird und somit die App nicht angenommen werden kann. Wir werden dies durch Tooltips korrigieren und somit nochmals submitten.

Aktualisierung vom 23. Mai 2013 Wir haben die Fehler behoben und Tooltips eingebaut um den Benutzer, besonders bei der ersten Benutzung, zu führen. Die App hat den Review Prozess bestanden und befindet sich im App Store.

D.8. Qualitätsmassnahmen

D.8.1. Dokumentation

Die Dokumentation dieser Studienarbeit erfolgt in LaTeX. Dieses Dokumentformat ermöglicht es uns, Dokumentationsänderungen jeweils auf der eingesetzten Versionisierungsplattform einzuchecken, was uns wiederum ein gemeinsames Bearbeiten der Dokumente ermöglicht.
Da es elementar ist, dass sich jedes Teammitglied auf die Aktualität der Dokumente verlassen kann, sollen Änderungen regelmässig ein-, bzw. ausgecheckt werden.
Dokument-Reviews sollen regelmässig, mindestens jedoch am Ende der einzelnen Sprints durchgeführt werden.

D.8.2. Projektmanagement

Die Verwaltung des Projekt geschieht im Projektverwaltungstool Jira, welches unter Punkt D.3 „Projekt Management“genauer beschrieben wird. Dies ermöglicht es allen Beteiligten, jeweils die aktuellsten Informationen über den Projektverlauf einsehen zu können, was sich wiederum positiv auf die gesamte Arbeit auswirkt, da automatisch eine Qualitätssicherung mittels Vier-Augen-Prinzip stattfindet.

D.8.3. Usability

Eine ansprechende Usability im Bezug auf die App sowie das CMS stellt bei dieser Arbeit ein Key-Feature dar, da sich das Produkt längerfristig verkaufen sollte und deshalb ein besonderes Augenmerk auf eine einfache Handhabung genannter Produkte gelegt werden muss.
Um diese gute Usability sicherstellen zu können, müssen möglichst früh entsprechende Tests durchgeführt werden. Geplant ist die Erarbeitung erster Wireframes entsprechender **Paper-Prototyping**-Tests während Sprint 2 - Entwurf und Sprint 3 - Prototyping. Weiter sind nach Beendigung der Entwicklung während der Testing-Phase in Sprint 5 - Testing & App Store weitere Usability-Tests eingeplant worden.

D.8.4. Entwicklung

Unser Projekt wird mithilfe der Versionskontrollsoftware git auf der Versionisierungsplattform GitHub abgelegt. Da Git eine dezentrale Versionierung verfolgt, ist ein Backup des Codes gewährleistet, da jedes der Projektmitglieder eine lokale Kopie des Codes besitzt. Es werden nur Quellcodedateien die lauffähig und getestet sind eingecheckt („stable trunk“). Somit wird allen Projektmitgliedern den Zugriff auf aktuellen und funktionsfähigen Code gewährt. Fälschliche Manipulationen des Quellcodes können durch ältere Versionen rückgängig gemacht werden.

Testing

Wo möglich wird die Qualität des Client Codes durch **Behavior Tests** sichergestellt. Hierfür wird Jasmine eingesetzt. Jasmine erlaubt es einem, in einer gut lesbaren Form einzelne Funktionalitäten des Codes zu testen.

```
1  describe("Included matchers:", function() {
2
3    it("The 'toBe' matcher compares with ===", function() {
4      var a = 12;
5      var b = a;
6
7      expect(a).toBe(b);
8      expect(a).not.toBe(null);
9    });
10
11   describe("The 'toEqual' matcher", function() {
12
13     it("works for simple literals and variables", function() {
14       var a = 12;
15       expect(a).toEqual(12);
16     });
17
18     it("should work for objects", function() {
19       var foo = {
20         a: 12,
21         b: 34
22       };
```

```
23        var bar = {
24          a: 12,
25          b: 34
26        };
27        expect(foo).toEqual(bar);
28      });
29    });
30  });
```

Quellcode D.1: Beispiel eines einfachen Jasmine-Tests

Ein weiterer Vorteil von Jasmine stellt der Support von Mocking dar. Gerade im Bezug auf den pillow Client ist dies sehr hilfreich, da so die REST API durch einen Mock abgekapselt werden kann.

```
1  describe("Manually ticking the Jasmine Mock Clock", function() {
2    var timerCallback;
3
4    beforeEach(function() {
5      timerCallback = jasmine.createSpy('timerCallback');
6      jasmine.Clock.useMock();
7    });
8
9    it("causes a timeout to be called synchronously", function() {
10     setTimeout(function() {
11       timerCallback();
12     }, 100);
13     expect(timerCallback).not.toHaveBeenCalled();
14     jasmine.Clock.tick(101);
15     expect(timerCallback).toHaveBeenCalled();
16   });
17
18   it("causes an interval to be called synchronously", function() {
19     setInterval(function() {
20       timerCallback();
21     }, 100);
22
23     expect(timerCallback).not.toHaveBeenCalled();
24
25     jasmine.Clock.tick(101);
26     expect(timerCallback.callCount).toEqual(1);
27
28     jasmine.Clock.tick(50);
29     expect(timerCallback.callCount).toEqual(1);
30
31     jasmine.Clock.tick(50);
32     expect(timerCallback.callCount).toEqual(2);
33   });
34 });
```

Quellcode D.2: Beispiel eines Jasmine-Tests inklusive Mocking

Automatisierte Tests
Bei jedem Commit wird Jenkins, welcher auf einem virtuellen Schulserver läuft, benachrichtigt. Jenkins holt sich dann die neusten Code-Sourcen und Tests auf dem Repository ab und führt diese aus. Sollte ein Test fehlschlagen, werden die Projektmitglieder per E-Mail darüber benachrichtigt. So kann die Codequalität jederzeit sichergestellt werden.

Packaging
Um die richtige Reihenfolge der Code Includes und optimierte Performance zu garantieren, wird der JavaScript Code beim Deployment (App Store Submission) compressed und minified. Dies wird durch den JavaScript parser/compressor/beautifier UglifyJS gewerkstelligt.

Code Reviews
Code-Reviews werden fortlaufend durchgeführt. Infolge knapper Terminplanung soll jedoch darauf geachtet werden, dass kritische Softwarekomponenten bei Reviews priorisiert werden. Reviews sollen jeweils durch die für den Code verantwortliche Person vorbereitet werden, um einen speditiven Ablauf gewährleisten zu können.

Code Reviews durch externe Person
Ebenfalls ist ein Code Review durch eine externe Person geplant, welche uns durch unseren Betreuer zugeorndet wird. Somit kann eine gute Code Qualität gewährleistet werden.

Zuweisung
Aktualisierung vom 8. April 2013: Für den Code Review wurde uns Mischa Trecco zugeordnet. Wir haben ihn diesbezüglich kontaktiert und einen Termin vereinbar.

Code Review Besprechung
Am 15. Mai 2013 wurde der Code Review gemacht. Das Dokument ist im Anhang F Code Review von Mischa Trecco zu finden.
Es wurde mit Mischa Trecco vereinbar, dass die Dokumentation des Codes mithilfe von JSDoc, ein Pendant zu JavaDoc, vorgenommen wird. Die entsprechende Code Dokumentation ist unter http://pillow.io/static/jsdoc/pillow.html zu finden.

Code Style Guidelines
Für die Sprachen, welche keine Guidelines aufgelistet sind, gilt es die Best Practices aus der Community zu befolgen.

Python
Bei Python Code halten wir uns an die Definition von Guido van Rossum und Barry Warsaw in PEP-8

JavaScript

Für Javascript hat Airbnb eine ausführliche und gute Style Guide ausgearbeitet, an welcher wir uns orientieren.

D.9. Rückblick

D.9.1. Stundenaufwand pro Mitglied

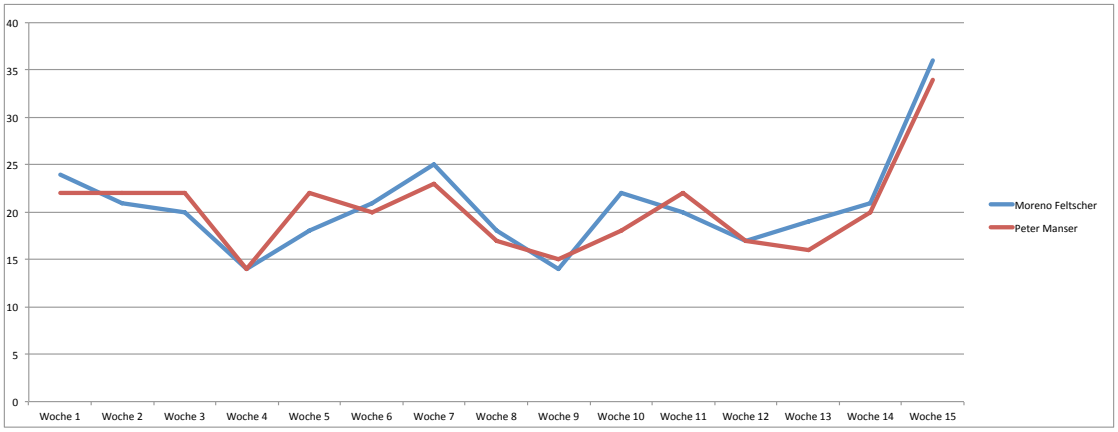

Abbildung D.3.: Projekt: Studenaufwand pro Mitglied

Woche	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**Total**
M. Feltscher	24	21	20	14	18	21	25	18	14	22	20	17	19	21	36	310
P. Manser	22	22	22	14	22	20	23	17	15	18	22	17	16	20	34	304

Tabelle D.9.: Projekt: Studenaufwand pro Mitglied

D.9.2. Stundenaufwand pro Komponente

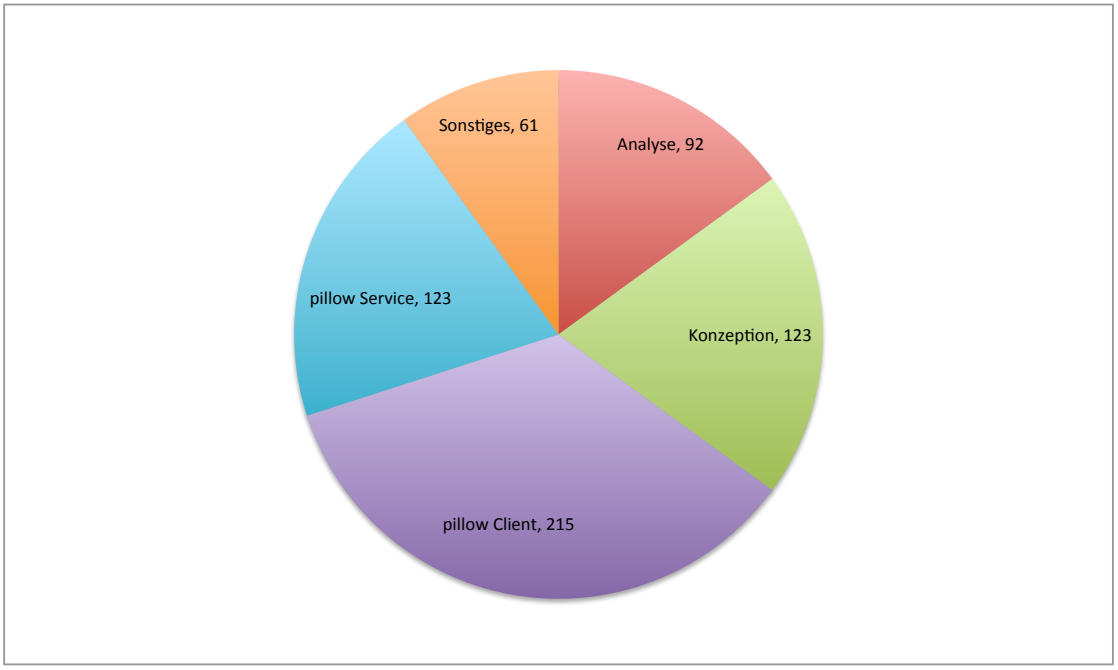

Abbildung D.4.: Projekt: Studenaufwand pro Komponente

Komponente	**Aufwand [h]**
Analyse	92
Konzeption	123
pillow Client	215
pillow Service	123
Sonstiges	61
Total	**614**

Tabelle D.11.: Projekt: Stundenaufwand pro Komponente

E.1. Kick-Off Meeting - Woche 1

Montag, 18. Februar 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

Treffen mit Industriepartner

PM erzählt zu Beginn der Sitzung kurz, dass wir vom 13. auf den 14. Februar im Hotel Hof Weissbad zu Gast waren, um das Ambiente rund um das Hotel gleich selber erleben zu können. Wir durften zwei erlebnisreiche und im Bezug auf unsere Studienarbeit interessante Tage erleben.
Der Aufenthalt gab auch Aufschluss über den eigentlichen Mehrwert einer solchen App für den Gast. So wurde dem Projektteam schnell klar, dass ein Fokus ebenfalls auf „In-House“-Informationen (sprich, Informationen welche dem Benutzer während seines Aufenthalts im Hotel zur Verfügung stehen) gelegt werden muss.

Erläuterung Entscheid generische App

Auf erneute Nachfrage durch MS nach dem Grund für den Lösungsansatz mittels generischer App, bei welcher dasselbe native App-Gerüst für alle Hotels verwendet wird, erläutern PM und MF nochmals kurz die wichtigsten Punkte welche zu diesen Entscheid geführt haben:

- Native Features können verwendet werden:
  - Geolocation für Automatisches Check-In des Gastes in Hotel
  - Push-Notifications um den Benutzer über aktuelle Sonderaktionen oder Ähnlichem informieren zu können

- App-Content ist sofort verfügbar, da der Verifikationsprozess einer nativen App wegfällt

Erläuterung Entscheid Entwicklung eigenes CMS

Prof. Dr. Markus Stolzewirft die berechtige Frage auf, warum kein bestehendes CMS zum Einsatz kommt und anstelle dessen ein eigenes CMS entwickelt wird. Das Projektteam unterstreicht diesen Beschluss mit folgenden Punkten:

- Anforderung an CMS sind zu spezifisch (spezielle Content-Types)
- Gängige CMS sind zu „mächtig“, sprich, sie bieten zu viel Funktionalitäten im Bezug auf die vom Projekt vorgegebene Problemstellung, sie bieten somit keinen Mehrwert
- Zusätzliche Abhängigkeit von Drittprodukt, Änderung an Datenstrukturen müssten jeweils vorzeitig entsprechend implementiert werden

Testing / Usability

Da eine gute Usability von App, wie auch CMS, natürlich gerade im Bezug auf das entsprechende Kundensegment ausschlaggebend ist, wurde das diesem Punkt auch entsprechende Beachtung während der Sitzung geschenkt. Es wurde festgehalten, dass entsprechende Usability-Tests Teil des Projekts sein sollen. Die entsprechenden Verfahren und ihre Anwendung für das Projekt werden vom Projektteam noch festgelegt.

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Technische Herausforderungen definieren
- Entscheidung bezüglich Projektmodell (RUP/SCRUM) fällen
- Projektplan anlegen (1. Version)
- Risikoliste anlegen (1. Version)

Abschluss / Gültigkeit

Protokoll anlässlich „Meeting - Woche 2“ am 25.2.2013 abgenommen.

E.2. Meeting - Woche 2

Montag, 25. Februar 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	E-Mail	Kürzel
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Kickoff-Meetings wurde abgenommen
- Die technische Herausforderungen wurden besprochen und detailliert diskutiert:
  - Neue mögliche Herausforderung: Synchronisation der Inhalte im Hintergrund - Prozess wird über Push-Notification direkt vom Server auf dem Client ausgelöst (Machbarkeit prüfen und eventuell umsetzen)
  - Herausforderung App-Links (Inhalte werden über Hyperlinks direkt in App geladen): Diese Herausforderung priorisiert behandeln, da ein Mehrwert für weitere App-Entwicklungen gewonnen werden kann (eventuell Blog-Post oder ähnliches darüber verfassen, damit zukünftige Apps von gewonnen Erkenntnissen profitieren könnten)
  - Eine Veröffentlichung der App im Apple AppStore wird angestrebt. Der Publish-Prozess soll demnach frühstmöglich (Richtwert Ende Woche 10) erfolgen
- Die zu diesem Zeitpunkt bestehende Risikoliste wurde erläutert und weitere Punkte definiert:
  - Neuer Punkt *CMS*: CMS soll skalieren und gute Usability muss gewährleistet werden
  - Aufgrund von neuer technischen Herausforderung *Publishing AppStore* muss abgeklärt werden, ob die im Projekt angestrebte Art einer generischen App überhaupt in Store zugelassen ist
- Da gute Usability ein Grundkriterium dieses Projekts darstellt, sollen möglichst früh entsprechende Wireframes angelegt und UX-getestet werden (konkreter Zeitpunkt: Sprint 2 - 3)

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Bewertungsmatrix studieren und Fragen notieren

- Analyse gemäss Sprint-Planning durchführen
- Szenarien erarbeiten und Herrn F. Bach zur Validierung zukommen lassen
- Aufgabenstellung ausarbeiten (MS)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 4. März 2013 um 17.00 Uhr im Büro 6.113 statt.

E.3. Meeting - Woche 3

Montag, 4. März 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	E-Mail	Kürzel
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 2 wurde abgenommen
- Die Bewertungsmatrix, welche durch MS zur Verfügung gestellt wurde, wurde besprochen und einzelne Punkte konkretisiert:
  - Abnahme der Meilensteine werden jeweils an den Wochenmeetings vorgenommen und entsprechend protokolliert
  - Vision Dokument nicht Teil der Arbeit, da bereits vorgängig mit Marktrepräsentator Hotel Hof Weissbad definiert
  - Benutzerbeobachtung nicht Teil des Projekts, da bereits vor Beginn des Projekts bei Besuch Hotel Hof Weissbad vorgenommen
  - „ Zeitschätzung“ pro User Story wird in Story Point anstelle von Anzahl Stunden vorgenommen
  - Konkurrenzanalyse nicht Teil des Projekts, gesammelte Links von MS inklusive Ergänzungen von MF und PM sollen jedoch in Anhang des Projektdokuments aufgeführt werden
  - GUI-Tests für Farbenblinde: Tests mittels bestehender Tools reicht aus
  - Entwicklung App: Anstelle von reinem JavaScript eventuell typisierte Sprache wie TypeScript oder CoffeeScript verwenden (muss evaluiert werden)
- MF und PM erläutern abgeschlossenen Sprint 1 inklusive Sprint Review
- Der für den Code-Review zuständige Betreuer wird infolge von Personalmutationen noch zugewiesen (vorerst erhält MS Zugriff auf das Projekt-Repository)
- Fokus der Arbeit wird von MF und PM erläutert und durch MS abgenommen:
  - Usability
  - AppLink
  - AppStore Submission
  - (Caching)
- Definitiver Fokus wird mit Architekturprototyp evaluiert

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Angepasste Bewertungsmatrix an MS liefern (MF + PM)
- Ausgearbeitete Szenarien zu Validation an MS liefern, danach an Hotel Hof Weissbad zu Priorisierung (MF + PM)
- Bearbeitung User Stories gemäss Sprint Planning (MF + PM)
- MS Zugang zu GitHub Repository einrichten (PM)
- Aufgabenstellung ausarbeiten (MS)
- Betreuer für Code Reviews melden (MS)

Nächste Sitzung

Die nächste Sitzung findet voraussichtlich am Montag, 11. März 2013, um 17.00 Uhr im Büro 6.113 statt.

E.4. Meeting - Woche 4

Montag, 11. März 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 3 wurde abgenommen
- Die durch MF und PM angepasste Bewertungsmatrix wurde durch MS abgesegnet
- Aufgabenstellung wurde besprochen und von allen Anwesenden inhaltlich gutgeheissen; einzige Änderung: Vorgehen Erstellung Video wird zu einem späteren Zeitpunkt konkretisiert
- Die Szenarien, welche für das Hotel Hof Weissbad ausgearbeitet wurden, wurden besprochen und abgenommen
- Eine erste Version des Domain Models wurde besprochen und Änderungsvorschläge durch MS aufgenommen
- Sprint-Replanning
  - Auf Geheiss von MS wird das Projektteam den laufenden Sprint etwas anpassen und erste Arbeiten am Architekturprototypen vornehmen, um technische Risiken effektiv angehen zu können
  - Dokumentation des Entwurfs wird teilweise auf Sprint 3 ausgelagert, da gewisse Erkenntnisse erst nach Arbeiten am Architekturprototypen vorliegen

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Ausgearbeitete Szenarien an Herrn Bach (Hotel Hof Weissbad) zu Priorisierung liefern (MF + PM)
- Betreuer für Code Reviews melden (MS)
- Domain Model gemäss Änderungsvorschlägen MS anpassen (MF + PM)
- Review Dokumente Sprint 1 (Projektplanung, Technische Herausforderungen, Anforderungsspezifikation) vornehmen (MS)
- Bearbeitung User Stories gemäss angepasstem Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 18. März 2013, um 17.00 Uhr im Büro 6.113 statt.

E.5. Meeting - Woche 5

Montag, 18. März 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	E-Mail	Kürzel
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 4 wurde abgenommen
- Rückmeldung F. Bach (Hotel Hof Weissbad) bezüglich eingereichter Szenarien: Verifikation, dass bei Herrn Bach dem Feature *Push-Notifications* hohe Priorität zugesprochen wird ⇒ wird priorisiert behandelt
- Dokumentationen über das Design der Lösung werden zeitlich nach hinten verschoben (Sprint 7), da Designentscheide anhand jetzigem Wissensstand noch gar nicht getroffen werden können
- Besprechung der von MF und PM erstellten Mockups für App mit MS:
  - Aufteilung auf zwei unterschiedliche Menüs (Grund-App / Hotel-App) wirkt verwirrend
  - Menüs am unteren Rand einer App sind ein Design-Konzept von iOS, da App jedoch plattformübergreifend eingesetzt werden soll muss eventuell auf ein neutrales Konzept zurückgegriffen werden
  - Unterscheidung zwischen Menüpunkten „Entdecken“ und „Suchen“ im Hauptmenü der App macht zu jetzigen Zeitpunkt wenig Sinn
  - Einheitliche Startseite für Hotels mit etwaiger Hauptnavigation soll angedacht werden
  - Zusätzlich zu Navigation in Hotel-App soll ein Wechsel zwischen einzelnen Seiten mittels Swipe-Gestures evaluiert werden
- Von MS eingebrachte oben genannte Feedbacks bezüglich App-Mockups werden von MF und PM detailliert besprochen und geprüft, um anschliessend mittels Paper-Prototyping getestet werden zu können
- Der Einsatz bestehender CMS-Frameworks macht gemäss MS durchaus Sinn, sofern Funktionsumfang Anforderungen deckt, damit kann Fokus auf andere Teile wie *Push-Notifications* gelegt werden ⇒ Framwork wird mittels Prototyping evaluiert

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Rückfrage an F. Bach (Hotel Hof Weissbad) bezüglich tiefster Priorität *QR-Code* (MF + PM)
- Betreuer für Code Reviews melden (MS)
- Review Dokumente Sprint 1 (Projektplanung, Technische Herausforderungen, Anforderungsspezifikation) vornehmen (MS)
- Überarbeiten erstellte Mockups / Paper-Prototyping (MF + PM)
- Bearbeitung User Stories (hauptsächlich Prototyping App/CMS) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 25. März 2013, um 17.00 Uhr im Büro 6.113 statt.

E.6. Meeting - Woche 6

Montag, 25. März 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 5 wurde abgenommen
- Review Projektplan, Technische Herausforderungen und Anforderungsspezifikation durchgeführt und Änderungswünsche aufgenommen:
  - Zusammenhang zwischen technischen Herausforderungen und Risiken entsprechend kennzeichnen
  - Nichtfunktionale Anforderungen müssen überarbeitet werden ⇒ Konkretere Zahlen nennen, Unterscheidung zwischen Produkt- und Projektanforderung genau spezifizieren
  - Sensitivitätsanalyse fehlt
- Entscheid, dass Fokus nicht mehr auf CMS (inklusive Usabilty / GUI) zu liegen kommt, da neue Herausforderung *Push-Notifications* aufgenommen wurde

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Betreuer für Code Reviews melden (MS)
- Dokumente Sprint 1 (Projektplanung, Technische Herausforderungen, Anforderungsspezifikation) gemäss Review überarbeiten (MF + PM)
- Überarbeiten erstellte Mockups / Paper-Prototyping (MF + PM)
- Bearbeitung User Stories (hauptsächlich Prototyping App/CMS) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 8. April 2013, um 17.00 Uhr im Büro 6.113 statt.

E.7. Meeting - Woche 8

Montag, 8. April 2013 - 17.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 6 wurde abgenommen
- Als Code Review Betreuer wurde Mischa Trecco durch MS festgelegt, erste Kontaktaufnahme steht noch aus
- Sprint-Anpassung: Das Projektteam musste den Sprint 3 um eine Woche verlängern, da sich die Entwicklung des Architekturprototypen als zeitintensiver als vorerst angenommen herausstellte. Als Konsequenz daraus, wird der Sprint 4 gestrichen und der Sprint 5 auf neu ebenfalls 3 Wochen festgelegt
- PM stellt Architekturprototypen, welcher wichtige Risiken beseitigt (namentlich App-Links, Push Notifications, PhoneGap) vor
- MF stellt überarbeitete Wireframes vor und erklärt vorgenommene Paper Prototyping Tests mit drei Personen (2x Personen über 50 Jahre alt, sowie 1x Informatikstudent)
  - Neue Wireframes finden sowohl bei Benutzer als auch bei MS Anklang, da vieles vereinfacht wurde
  - MS merkt an, dass bei Paper Prototyp gestellte Aufgabenstellung zu detailliert beschrieben ist und lediglich Wireframes, nicht jedoch Grundidee des Produkts abdeckt
  - QR-Code Scanning in Zusammenhang mit Hotel-Plakat muss noch getestet werden
  - Um Zeitaufwand, welcher solche Tests mit sich bringen, möglichst gering zu halten, hält MS jedoch fest, dass nur noch neue Elemente genauer getestet werden sollen, bestehende Tests reichen so aus

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Code Review Betreuer bezüglich Code Review Daten kontaktieren (MF + PM)

- Bearbeitung User Stories (Entwicklung App/CMS) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 22. April 2013, um 17.00 Uhr im Büro 6.113 statt.

E.8. Meeting - Woche 11

Montag, 29. April 2013 - 16.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 8 wurde abgenommen
- Das Projektteam hat in Woche 9 mit Code Review Betreuer Mischa Trecco ein erstes Meeting durchgeführt, M. Trecco muss Projektteam noch GitHub-Account mitteilen, damit Repository für ihn freigegeben werden kann
- Projektteam stellt MS aktuellen Stand des Projekts (App, Backend und Datenmodell) vor und gibt bekannt, dass die App per Anfangs nächster Woche in App Store submitted werden kann
- Bezüglich App Store Submission bringt MS ein, dass eventuell auch S. Hunkeler oder K. Gaunt noch einige Inputs geben können

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Code Review Betreuer bezüglich GitHub Account-Daten kontaktieren (MF + PM)
- Code Review Betreuer bezüglich Datum für Feedback erstes Review anfragen (MF + PM)
- Hotel-Daten auffrischen (Input F. Bach bezüglich Responsive Webseite beachten) (MF + PM)
- App Store Submission, vorgängig zusammengestellte Checkliste abarbeiten und oben genannte IFS-Mitarbeiter kontaktieren (MF + PM)
- Bearbeitung User Stories (Testing Entwicklung App CMS) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Montag, 13. Mai 2013, um 14.00 Uhr im Büro 6.113 statt.

E.9. Meeting - Woche 13

Montag, 13. Mai 2013 - 14.00 Uhr

Sitzungsteilnehmer

Teilnehmer	E-Mail	Kürzel
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 11 wurde abgenommen
- Ein erster Code Review wurde durch M. Trecco vorgenommen - das entsprechende Feedback wurde durch das Projektteam besprochen und einzelne Punkte bereits verbessert
- Code Review Meeting war für Woche 12 geplant, konnte jedoch nicht durchgeführt werden, da M. Trecco Termin versäumte
- Das Projektteam informiert MS über erfolgte App Store Submission, welche in Woche 14 erfolgte. Hierbei wurden die IFS-Mitarbeiter S. Hunkeler und K. Gaunt vorgängig noch im Bezug auf deren Erfahrungen in diesem Bereich durch das Projektteam kontaktiert.
- MS weist darauf hin, dass nun unbedingt an der Architekturdokumentation weitergearbeitet werden soll, damit diese durch ihn in einem Review gesichtet werden kann

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Code Review Meeting mit M. Trecco festlegen und durchführen (MF + PM)
- Architektur dokumentieren (unter anderem Komponentensicht und Sequenzdiagramm der App) (MF + PM)
- Bearbeitung User Stories (Testing Entwicklung App CMS) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Mittwoch, 23. Mai 2013, um 13.00 Uhr im Büro 6.113 statt.

E.10. Meeting - Woche 14

Donnerstag, 23. Mai 2013 - 16.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das Protokoll des Meetings Woche 13 wurde abgenommen
- Die Besprechung des Code Reviews wurde mit M. Trecco in Woche 13 durchgeführt und die einzelnen Punkte besprochen. Es wurde zudem mit M. Trecco vereinbart, dass der Code mithilfe von JSDoc (JavaScript-Pendant zu JavaDoc) dokumentiert wird.
- MS wurde informiert, dass die App nun im App Store verfügbar ist
- Die angefertigten Architekturdokumente (Übersicht, Architekturdiagramm, Komponentenansicht, ERM und Systemsequenzdiagramm Startup Process) wurden besprochen. MS brachte einige Änderungsvorschläge zu den einzelnen Dokumenten ab, welche durch das Projektteam aufgenommen wurden und überarbeitet werden.
- MS weist darauf hin, dass das Rendering der Inhalte ebenfalls in einem Systemsequenzdiagramm abgebildet werden sollte, da es sich um eine Kernkomponente der Softwarelösung handelt
- Das Projektteam stellt MS das Storyboard für das Video vor. MS weist darauf hin, dass bei der Erstellung des Videos auf Musik verzichtet werden sollte, Tonspuren im Sinne von gesprochenen Erläuterungen jedoch durchaus Sinn machen würden.
- PM fragt nach, wie das Poster auszusehen habe. MS erklärt, dass dies in etwa den im Video verwendeten Aufbau haben soll („vom Problem zur Lösung“-Ansatz)

Weiteres Vorgehen

Folgende Arbeiten sind bis zur nächsten Sitzung zu erledigen:

- Überarbeitung der Architekturdokumente gemäss Feedback MS (MF + PM)
- Anfertigen des Videos (MF + PM)
- Anfertigen des Posters (MF + PM)
- Bearbeitung User Stories (Dokumentation) gemäss Sprint Planning (MF + PM)

Nächste Sitzung

Die nächste Sitzung findet am Mittwoch, 29. Mai 2013, um 14.00 Uhr im Raum 1.212 statt.

E.11. Meeting - Woche 15

Donnerstag, 29. Mai 2013 - 14.00 Uhr

Sitzungsteilnehmer

Teilnehmer	**E-Mail**	**Kürzel**
Prof. Dr. Markus Stolze	mstolze@hsr.ch	MS
Peter Manser	pmanser@hsr.ch	PM
Moreno Feltscher	mfeltsch@hsr.ch	MF

Beschlüsse (Diskussion)

- Das eingereichte Video wurde besprochen
  Feedback MS:
  - Einleitender Satz soll eingepflegt werden, welcher Produkt und dessen Vorteile beschreibt
  - Produkt soll noch mehr von der marketingtechnischen Seite beleuchtet werden, indem die Vorteile für die Hotels hervorgehoben werden
  - Demo soll Prozess des QR-Code Scans beinhalten
  - Gewisse Slides sollen mit Titel versehen werden, damit sich auch nicht mit dem Projekt vertraute Personen ein Bild davon machen können
- Das eingereichte Poster wurde besprochen
  Feedback MS: Poster soll in Anlehnung an die Änderungsvorschläge zum Video überarbeitet werden: Es fehlt ein Leitsatz, welcher pillow und dessen Vorteile beschreibt
- Das eingereichte Abstract wurde besprochen
  Feedback MS: Ein einleitender Satz soll hinzugefügt werden, welcher dem Leser gleich den Kontext, in welchem sich das vorgestellte Produkt bewegt, aufzeigt
- Der eingereichte Blog Post wurde besprochen
  Feedback MS: Das Konzept der App Links soll noch etwas umfangreicher beschrieben werden
- MS erläutert, dass die Dokumentation der Arbeit lediglich in elektronischer Form vorzuliegen hat, eine gedruckte Variante ist nicht von Nöten
- MS erklärt, dass die beiden Extra-Arbeiten „Video“ und „Blog-Post“ auch erst nach Abschluss des Projekts vom 31. Mai abgegeben werden können

Weiteres Vorgehen

Folgende Arbeiten sind bis zum Abschluss des Projekts zu erledigen:

- Überarbeiten des Posters gemäss Feedback (MF + PM)
- Überarbeiten des Abstracts gemäss Feedback (MF + PM)
- Bearbeitung User Stories (Dokumentation) gemäss Sprint Planning (MF + PM)
- Abschluss des Projekts (MF + PM)

Folgende Arbeiten dürfen nach Abschluss des Projekts erledigt werden:

- Überarbeiten des Videos gemäss Feedback (MF + PM)
- Überarbeiten des Blog Posts gemäss Feedback (MF + PM)

Anhang F Code Review von Mischa Trecco

Mischa Trecco (mtrecco@hsr.ch) vom 6. Mai 2013, bezieht sich auf Revision 678aefbe.

F.1. src > iOS > www

1. Optimierung der Performance durch Reduzierung der Anzahl Web-Request:
   a) Bei einer offline-App weniger relevant, aber grundsätzlich besser:
      i. Mehrere CSS-Dateien in eine kombinieren und minimieren.
      ii. Mehrere JS-Dateien in eine kombinieren und minimieren.
   b) Am besten so dass in der Entwicklung alle Scripts einzeln und nicht minimiert eingebunden werden und beim release die kombinierten/optimierten.

2. var self = this; Finde ich super. (klarer als var that = this. Ich persönlich würde dann aber nur noch self in der Funktion verwenden und nicht mehr mit this arbeiten.)

```
1  var self = this;
2  if (!pillow.Cache.favoritesCollection) {
3      pillow.Cache.favoritesCollection = new pillow.Collections.Favorites
           ({guest: pillow.Cache.guest});
4      pillow.Cache.favoritesCollection.fetch({success: function(data) {
5          self.openFavoritesView();
6      }});
7  } else {
8      self.openFavoritesView();
9  }
10 this.openFavoritesView = function() {
11     self.renderView(new pillow.Views.Favorites({collection: pillow.Cache
           .favoritesCollection}));
12 };
```

3. App.js Line 27, 28 pillow.events.initialize();
   a) Kann sein das pillow.events nicht definiert ist.

4. App.js Line 41f
   a) Kann sein das pillow.Cache nicht definiert ist.

5. App.js Line 31 Jr.Router.extend und weitere (e.g. views.js)
   a) Jr als Argument der Funktion übergeben
   b) Grundsätzlich: Alle benötigten „Referenzen“ der Funktion übergeben

6. App.js Line 1-20: Diesen Code würde ich in ein separates js-file verschieben, gehört nicht zum Rest in App.js

```
1  // Override backbone's parse function due to different API
       specifications
2  Backbone.Collection.prototype.parse = function (response) {
3      if (_.isObject(response.objects) ) {
4          return response.objects;
5      }else{
6          return response;
7      }
8  };
9  Backbone.Model.prototype.parse = function (response) {
10     if (_.isObject(response.objects) ) {
11         return response.objects[0];
12     }else{
13         return response;
14     }
15 };
16 Backbone.Model.prototype.url = function() {
17     var base = _.result(this, 'urlRoot') || _.result(this.collection, '
           url') || urlError();
18     if (this.isNew()) return base;
19     return base + (base.charAt(base.length - 1) === '/' ? '' : '/') +
20 };
```

7. Views.js: sicher dass Jr.Navigator, Jr.Views, pillow.Templates existiert?

8. Views.js:
   a) Ich nehme an folgendes Konzept ist durch das eingesetzte Framework vorgegeben:

```
1  events: {
2      'click #favorites-btn-add': 'onTapAddButton',
3       'click .list > li > a': 'openItem'
4  },
```

   b) Es ist insoweit gefährlich als dass dies euer ViewModel stark an die View koppelt. Es ist sehr einfach und gut möglich, dass das Template-Design angepasst wird und dann ein Event-Selektor wie z.B. „click .list > li > a“nicht mehr gültig ist.

9. Collections.js: Ich würde den Code nicht selber versuchen zu optimieren (hier mit Klammern weglassen) sondern auf Leserlichkeit setzen und dann mit einem Tool minimieren/optimieren.

```
if (searchModel instanceof pillow.Models.App) context = 'app';
else if (searchModel instanceof pillow.Models.Guest) context = 'guest';
else return false;
```

10. Collections.js, Events.js, Functions.js .. Grundsätzlich: Fehler-/Meldungs-Texte auslagern und als Konstante verwenden. (Magic String Smell) So sind sie alle an einem Ort verwaltbar und einfach lokalisierbar.
    a) Dies gilt ebenfalls für URLs (z.B. in models.js Line 91):

```
urlRoot: pillow.Settings.DefaultApiUrl + 'favorite/',
```

F.2. src > cms

1. Ich persönlich würde vermeiden Passwörter/Secrets plain im code zu haben. Würde sie mind. base64 kodieren um shoulder-surfing zu erschweren. Darüber kann man aber lange diskutieren ;)

```
DATABASE = {
    'engine': 'peewee.MySQLDatabase', 'name': 'pillow',
    'user': 'root',
    'passwd': 'XXXX'
}
SECRET_KEY = 'XXXX'
```

2. create_db.py Eventuell realistischere Inhalte?

```
Content.create(page=p1a, content_type=ct2, value='{"value": "Willkommen
    im Hof Weissbad, bla bla bla bla bla..."}')
```

F.3. Gesamter Code

1. Ich bin kein Fan von quantitativer Kommentierererei aber gar keine Kommentare smellt ziemlich. Es wäre hilfreich bei Codeblöcken bei denen nicht auf einen Blick klar ist was sie machen, einen kurzen Kommentar zu schreiben.
    a) Korrektur: Habe einen Kommentar gefunden > app.js, der ist gut :)
    b) Vielleicht würde in eurem Fall auch schon eine kurze Beschreibung der JS-„Klasse“ genügen um den Code schneller/besser zu verstehen.

F.4. Fazit

Folgende Punkte würde ich besonders ins Auge fassen: 10, 13 (entspricht „Gesamter Code“), 6, 5 Guter Code. Weiter so!

Anhang G Linksammlung

G.1. Konkurrenz

- http://www.restoapp.com/
- http://www.appyourself.net/de/samples
- https://www.mobilescreen.ch/preise.php
- http://www.csdoon.org/2012/10/app-creator-for-iphone-android-mac-and.html
- http://ibuildapp.com/
- http://www.appsbar.com/
- http://appsmakerstore.com/
- http://seattleclouds.com/
- http://holycowapps.com/blog/app-builder-for-iphone-and-android-why-holy-cow...
- http://www.appnotch.com/index.aspx
- http://www.apps-builder.com/de/home
- http://easyapps.ch/index.php/kontakt-oben
- http://www.fluxdvd.com/index.php/solutions-left/content2app
- http://www.quickappscms.org/
- http://dailytekk.com/2012/01/31/useful-startups-appafolio-the-best-iphone-app...
- http://www.appmachine.com/
- http://www.app-machine.com/de/Start-1.html?l=1
- https://www.kitapps.com/
- http://www.andromo.com/
- http://www.shoutem.com/
- http://www.fastcompany.com/1825568/make-iphone-android-app-without...

G.2. Hotel Apps

- http://www.hotelalexzermatt.com/de/hotel/service/news/news-detail/article/...
- http://www.ascot.ch/de/news/my-ascot-app
- http://www.continental.ch/deu/default.shtml
- http://appvz2.iphone-blog.ch/hotel-restaurant-rossli-stansstad/
- http://www.bernerhof-gstaad.ch/inguide-app.html#.UPKQmInjlm0
- https://itunes.apple.com/ch/app/hotel-swiss-star/id562555637?mt=8
- https://itunes.apple.com/ch/app/bad-bubendorf-hotel/id446172292?mt=8

G.3. Hotel Apps Overviews

- http://blog.lesroches.edu/alumni/susana/mobile-apps-hotels/
- http://www.ehospitalitytimes.com/?p=49104
- http://www.villacarona.ch/en/iphone-apps.html
- http://media.claridge.ch/d/claridge/media/htr.pdf
- http://www.ehospitalitytimes.com/?p=55840

Anhang H Load Tests

H.1. Beschreibung

Die Loadtest wurden an der HSR mit 2 Rechnern durchgeführt. Ziel war es, die Belastbarkeit der Server zu analysieren und die Server Einstellungen diesbezüglich zu optimieren.

Um dies vernünftig zu testen und den Live Betrieb nicht zu stören haben wir eine 1:1 Abbildung des Live Systems auf der Domäne test.pillow.io aufgeschalten. Zudem haben wir ein Testdatenscript „create_load_db.py“ geschrieben um entsprechende Testdaten generieren zu können.

H.2. Resultat

Bei den Tests haben wir herausgefunden, das unser Service 1'000 Apps mit 1'000 konkurrenten Benutzern gut verwalten kann. Es gibt lediglich eine Fehlerrate von ~1%. Der gunicorn Server wurde in der Konfiguration diesbezüglich auf 16 Worker Threads optimiert.

H.2.1. Screenshot Apache JMeter

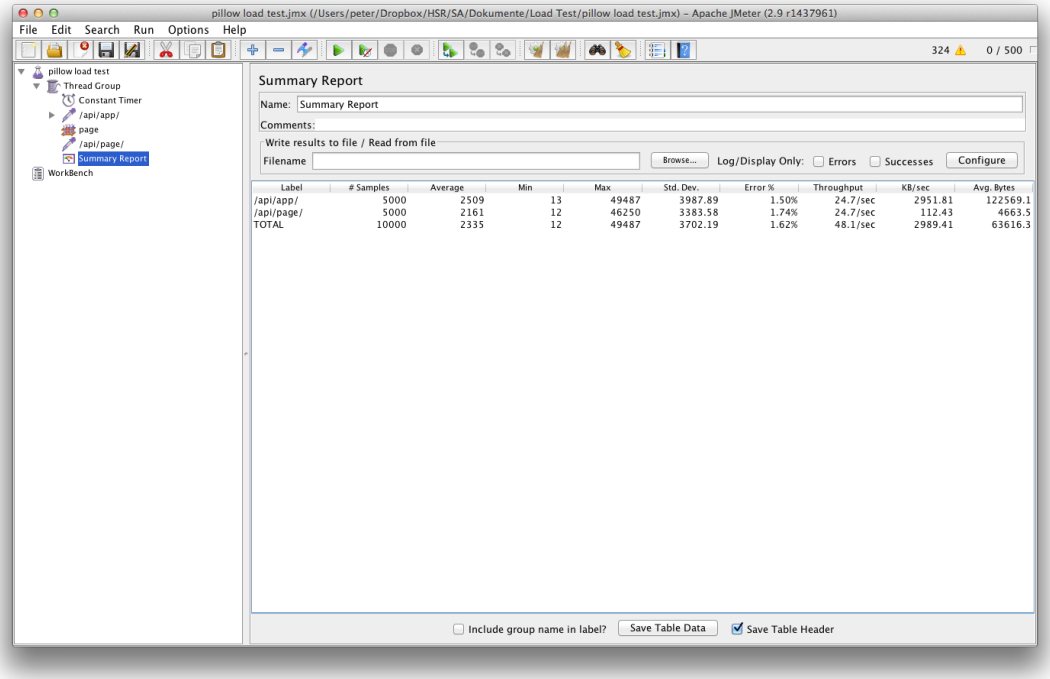

Abbildung H.1.: Screenshot Apache JMeter

H.2.2. Momentaufnahme Server

```
$ top
top - 09:09:47 up 175 days,  4:33,  1 user,  load average: 3.79, 1.60, 0.66
Tasks:  77 total,   6 running,  71 sleeping,   0 stopped,   0 zombie
Cpu(s): 26.5%us, 10.9%sy,  0.0%ni, 60.1%id,  1.3%wa,  0.0%hi,  1.2%si,  0.0%
    st
Mem:   1048576k total,  1048576k used,        0k free,        0k buffers
Swap:  2097152k total,        0k used,  2097152k free,   717672k cached
```

H.2.3. Optimierung

Wenn wir nun mehr Benutzer handlen wollen, haben wir Möglichkeit, mehr Service Instanzen aufzustellen. Da unsere APIs dem REST Programmierparadigma folgen, stellt dies keinerlei Problem dar. Um dann die Requests zu handeln, müsste ein Load Balancer (bspw. nginx) davor geschalten werden, um die Requests auf die Services zu verteilen.

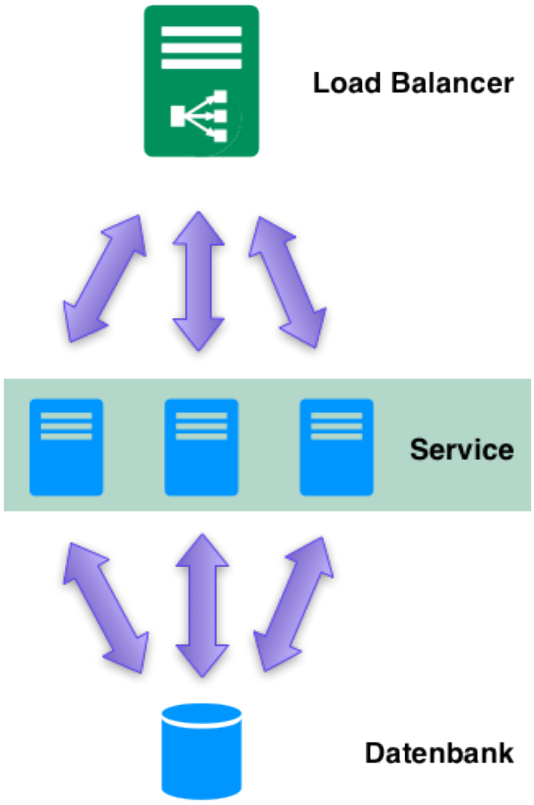

Abbildung H.2.: Skizze Load Balancing

H.3. Konfiguration

Unsere Apache JMeter Load Test File sieht folgendermassen aus:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <jmeterTestPlan version="1.2" properties="2.4" jmeter="2.9 r1437961">
3    <hashTree>
4      <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="pillow
           load test" enabled="true">
5        <stringProp name="TestPlan.comments"></stringProp>
6        <boolProp name="TestPlan.functional_mode">false</boolProp>
7        <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
8        <elementProp name="TestPlan.user_defined_variables" elementType="
             Arguments" guiclass="ArgumentsPanel" testclass="Arguments" testname
             ="User Defined Variables" enabled="true">
9          <collectionProp name="Arguments.arguments"/>
10       </elementProp>
11       <stringProp name="TestPlan.user_define_classpath"></stringProp>
12     </TestPlan>
13     <hashTree>
```

```
14      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname
            ="Thread Group" enabled="true">
15        <stringProp name="ThreadGroup.on_sample_error">continue</stringProp>
16        <elementProp name="ThreadGroup.main_controller" elementType="
              LoopController" guiclass="LoopControlPanel" testclass="
              LoopController" testname="Loop Controller" enabled="true">
17          <boolProp name="LoopController.continue_forever">false</boolProp>
18          <stringProp name="LoopController.loops">10</stringProp>
19        </elementProp>
20        <stringProp name="ThreadGroup.num_threads">500</stringProp>
21        <stringProp name="ThreadGroup.ramp_time">30</stringProp>
22        <longProp name="ThreadGroup.start_time">1333478267000</longProp>
23        <longProp name="ThreadGroup.end_time">1333478267000</longProp>
24        <boolProp name="ThreadGroup.scheduler">false</boolProp>
25        <stringProp name="ThreadGroup.duration"></stringProp>
26        <stringProp name="ThreadGroup.delay"></stringProp>
27      </ThreadGroup>
28      <hashTree>
29        <ConstantTimer guiclass="ConstantTimerGui" testclass="ConstantTimer"
              testname="Constant Timer" enabled="true">
30          <stringProp name="ConstantTimer.delay">5000</stringProp>
31        </ConstantTimer>
32        <hashTree/>
33        <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="
              HTTPSamplerProxy" testname="/api/app/" enabled="true">
34          <elementProp name="HTTPsampler.Arguments" elementType="Arguments"
                guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="
                true">
35            <collectionProp name="Arguments.arguments"/>
36          </elementProp>
37          <stringProp name="HTTPSampler.domain">test.pillow.io</stringProp>
38          <stringProp name="HTTPSampler.port">80</stringProp>
39          <stringProp name="HTTPSampler.connect_timeout"></stringProp>
40          <stringProp name="HTTPSampler.response_timeout"></stringProp>
41          <stringProp name="HTTPSampler.protocol">http</stringProp>
42          <stringProp name="HTTPSampler.contentEncoding"></stringProp>
43          <stringProp name="HTTPSampler.path">/api/app</stringProp>
44          <stringProp name="HTTPSampler.method">GET</stringProp>
45          <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
46          <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
47          <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
48          <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
49          <stringProp name="HTTPSampler.implementation">Java</stringProp>
50          <boolProp name="HTTPSampler.monitor">false</boolProp>
51          <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
52        </HTTPSamplerProxy>
53        <hashTree>
54          <HeaderManager guiclass="HeaderPanel" testclass="HeaderManager"
                testname="HTTP Header Manager" enabled="true">
55            <collectionProp name="HeaderManager.headers">
56              <elementProp name="Accept-Language" elementType="Header">
57                <stringProp name="Header.name">Accept-Language</stringProp>
```

```
58                <stringProp name="Header.value">de-ch,de;q=0.8,en-us;q=0.5,en
                      ;q=0.3</stringProp>
59              </elementProp>
60              <elementProp name="Accept" elementType="Header">
61                <stringProp name="Header.name">Accept</stringProp>
62                <stringProp name="Header.value">text/html,application/xhtml+
                      xml,application/xml;q=0.9,*/*;q=0.8</stringProp>
63              </elementProp>
64              <elementProp name="User-Agent" elementType="Header">
65                <stringProp name="Header.name">User-Agent</stringProp>
66                <stringProp name="Header.value">Mozilla/5.0 (Macintosh; Intel
                       Mac OS X 10.7; rv:8.0.1) Gecko/20100101 Firefox/8.0.1</
                      stringProp>
67              </elementProp>
68              <elementProp name="Accept-Encoding" elementType="Header">
69                <stringProp name="Header.name">Accept-Encoding</stringProp>
70                <stringProp name="Header.value">gzip, deflate</stringProp>
71              </elementProp>
72              <elementProp name="Accept-Charset" elementType="Header">
73                <stringProp name="Header.name">Accept-Charset</stringProp>
74                <stringProp name="Header.value">ISO-8859-1,utf-8;q=0.7,*;q
                      =0.7</stringProp>
75              </elementProp>
76            </collectionProp>
77          </HeaderManager>
78          <hashTree/>
79        </hashTree>
80        <RandomVariableConfig guiclass="TestBeanGUI" testclass="
              RandomVariableConfig" testname="page" enabled="true">
81          <stringProp name="maximumValue">1000</stringProp>
82          <stringProp name="minimumValue">1</stringProp>
83          <stringProp name="outputFormat"></stringProp>
84          <boolProp name="perThread">true</boolProp>
85          <stringProp name="randomSeed"></stringProp>
86          <stringProp name="variableName">page</stringProp>
87        </RandomVariableConfig>
88        <hashTree/>
89        <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="
              HTTPSamplerProxy" testname="/api/page/" enabled="true">
90          <elementProp name="HTTPsampler.Arguments" elementType="Arguments"
                guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="
                true">
91            <collectionProp name="Arguments.arguments">
92              <elementProp name="app" elementType="HTTPArgument">
93                <boolProp name="HTTPArgument.always_encode">false</boolProp>
94                <stringProp name="Argument.value">${page} </stringProp>
95                <stringProp name="Argument.metadata">=</stringProp>
96                <boolProp name="HTTPArgument.use_equals">true</boolProp>
97                <stringProp name="Argument.name">app</stringProp>
98              </elementProp>
99            </collectionProp>
100         </elementProp>
101         <stringProp name="HTTPSampler.domain">test.pillow.io</stringProp>
```

```
102           <stringProp name="HTTPSampler.port">80</stringProp>
103           <stringProp name="HTTPSampler.connect_timeout"></stringProp>
104           <stringProp name="HTTPSampler.response_timeout"></stringProp>
105           <stringProp name="HTTPSampler.protocol">http</stringProp>
106           <stringProp name="HTTPSampler.contentEncoding"></stringProp>
107           <stringProp name="HTTPSampler.path">/api/page</stringProp>
108           <stringProp name="HTTPSampler.method">GET</stringProp>
109           <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
110           <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
111           <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
112           <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
113           <stringProp name="HTTPSampler.implementation">Java</stringProp>
114           <boolProp name="HTTPSampler.monitor">false</boolProp>
115           <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
116         </HTTPSamplerProxy>
117         <hashTree/>
118         <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
                testname="Summary Report" enabled="true">
119           <boolProp name="ResultCollector.error_logging">false</boolProp>
120           <objProp>
121             <name>saveConfig</name>
122             <value class="SampleSaveConfiguration">
123               <time>true</time>
124               <latency>true</latency>
125               <timestamp>true</timestamp>
126               <success>true</success>
127               <label>true</label>
128               <code>true</code>
129               <message>true</message>
130               <threadName>true</threadName>
131               <dataType>true</dataType>
132               <encoding>false</encoding>
133               <assertions>true</assertions>
134               <subresults>true</subresults>
135               <responseData>false</responseData>
136               <samplerData>false</samplerData>
137               <xml>true</xml>
138               <fieldNames>false</fieldNames>
139               <responseHeaders>false</responseHeaders>
140               <requestHeaders>false</requestHeaders>
141               <responseDataOnError>false</responseDataOnError>
142               <saveAssertionResultsFailureMessage>false</
                      saveAssertionResultsFailureMessage>
143               <assertionsResultsToSave>0</assertionsResultsToSave>
144               <bytes>true</bytes>
145             </value>
146           </objProp>
147           <stringProp name="filename"></stringProp>
148         </ResultCollector>
149         <hashTree/>
150       </hashTree>
151     </hashTree>
152   </hashTree>
```

```
153 </jmeterTestPlan>
```

Quellcode H.1: pillow.jmx

Anhang I Lessons Learned

I.1. PhoneGap und Plugins

PhoneGap ist ein lebendes Produkt, das sieht man auch dadurch, dass alleine seit Beginn unserer Arbeit 3 neue Versionen released wurden. Nun ist es so, dass wenn PhoneGap updated auch die Plugins eventuellerweise Anpassungen machen müssen, was nicht gewährleistet ist, da die meisten Plugins von irgendwelchen (privaten) Entwicklern gepflegt werden. (Beispiel: das von uns einegesetzte PushNotification Plugin). Es kann also durch aus sein, das Plugins nicht mehr gepflegt werden bzw. veraltet sind. Das gilt auch bei der Auswahl des Plugins, man muss unbedingt auf die Kompatibilität der Plugins mit der eingesetzten PhoneGap Version achten.

Bei der Auswahl des Plugins haben wir folgende Tipps (sortiert nach Wichtigkeit):

- Die PhoneGap Version muss supported sein
- Das Plugin sollte auf GitHub (oder ähnlich) published sein
- Das Plugin sollte beliebt sein (s. Anzahl „Stars“)
- Es ist eine Dokumentation des Plugins vorhanden
- Der letzte Commit sollte nicht länger als ein Monat zurück liegen
- Es sollte nicht viele offene Issues und/oder Pull Requests geben

Diese Punkte sollten bei der Auswahl eines soliden PhoneGap Plugins helfen.

I.2. Debugging PhoneGap

Wenn man eine PhoneGap App schreibt wird man bald man an den Punkt kommen, an welchem etwas nicht funktioniert. Die stellt sich als besonders problematisch raus, da man weder eine Console mit JavaScript Fehlern bzw. Warnungen. Nachdem man sich dann eine Weile mit „alert()“rumgequält hat und den Fehler noch immer nicht gefunden hat, will man eine bessere Lösung.

Nach einer Suche nach guten Tools sind wir auf ein geniales Tool namens „weinre“ gestossen. Dieses erlaubt es einen ganzen Web Inspector via Remote anzuhängen. Somit

hat man via Remote Zugriff auf den ganzen DOM inklusive JavaScript Console, was das Debuggen enorm vereinfacht.

I.3. Paper Prototyping

Beim Paper Prototyping ist darauf zu achten, dass man dem Benutzer so wenig Informationen wie möglich gibt und nur mit wenigen Worten beschreibt, was das Ziel des Tests ist. Zudem ist darauf zu achten, dass man nicht die Begriffe verwendet, welche auch schon auf dem Screen vorkommen - das machts dann zu einfach für den Benutzer.

I.3.1. Schlechte Beispiele

- Öffne die pillow App und füge einen Favoriten über das „+ Menü“ hinzu.
- Öffne das Hotel Hof Weissbad, gehe ins Menü und öffne die Seite Zimmer, um die Preise zu finden.

I.3.2. Bessere Beispiele

- Markier ein Hotel für dich als wichtig.
- Zeige mir die Zimmerpreise des Hotel Hof Weissbad.

Anhang J Persönlicher Bericht Moreno Feltscher

Der Umstand, dass wir ein selbst gewähltes Thema in unserer Studienarbeit einbringen durften war für mich sowohl herausfordernd als auch motivierend.
Dahingehend herausfordernd, als dass man sich seine Aufgabenstellung bis zu einem gewissen Grad selbst zusammenzustellen hatte und deshalb im Bereich der Planung einige Stolpersteine zu überwinden hatte. So wäre das Projekt an sich anfangs beinahe gescheitert, da einer der betreuenden Dozenten in letzter Sekunde abgesprungen ist. Glücklicherweise konnten wir daraufhin mit Prof. Dr. Markus Stolzeeinen äusserst motivierten Projektbetreuer in Empfang nehmen.
Motivierend im Sinne des Umstands, dass es sich bei dem Thema um eine Arbeit im Bereich der Internettechnologien, welchen ich als sehr spannend erachte und auch zu meinem Beruf machte, handelte.

Meine Motivation für das Projekt war über die gesamte Projektdauer hoch, was nicht zuletzt dem Umstand zuzuschreiben war, dass ich mir sehr viele neue Konzepte und Praktiken anzueignen hatte und so der Lernfortschritt zu keinem Zeitpunkt auf der Strecke blieb. So trug ich zum Beispiel meinen Teil zur Entwicklung des pillow Clients auf dem Framework Backbone.js bei und musste mich anfangs erst einmal in die mir zu diesem Zeitpunkt noch unbekannte Materie einarbeiten. Zugegebenermassen ist die Lernkurve bei Backbone.js eher steil und somit hatte ich zu Beginn meine Probleme diesbezüglich. Nichtsdestotrotz hat sich die Designentscheidung, auf dieses Framework zu setzen, bewährt und so konnte ich schon bald die ersten Erfolge verzeichnen.

Weiter konnte ich von Peters Erfahrung im Bereich der Webentwicklung über die gesamte Projektdauer hinaus profitieren. So konnte ich zum Beispiel meine Kenntnisse im Bereich der API-Entwicklung anhand des eingesetzten Python-Framework „Flask“ vertiefen und mich durch Peters Expertise diesbezüglich weiterentwickeln.

Die Entscheidung, auf das Projektmodell SCRUM zu setzen war für mich ebenfalls sehr lehrreich, da ich das Modell bisher noch nie in der Praxis anwenden konnte. Glücklicherweise konnte ich auch hier auf Peters Erfahrung zählen und so waren mir Story Points oder „Sprint Plannings“bald kein Fremdwort mehr.

An dieser Stelle möchte ich meinem Projektmitglied ganz herzlich für seine Hilfestellungen und manchmal auch Geduld danken.

Weiter möchte ich mich natürlich auch bei unserem Industriepartner „Hotel Hof Weissbad“, bei welchem wir zwei super Tage anlässlich des Projekt Kickoffs geniessen durften, bedanken.

Abschliessend gilt unserem Projektbetreuer, Prof. Dr. Markus Stolze, natürlich auch ein grosses Dankeschön. Dank ihm durften wir die Arbeit erst realisieren, was sicherlich nicht selbstverständlich ist. Weiter fühlten wir uns jederzeit gut unterstützt durch seine Inputs und Feedbacks, welche allesamt sehr produktiv waren.

J.1. Lessons Learned

- SCRUM eignet sich zur Durchführung eines Studienprojekts, wenn auch zum Teil sehr aufwändig
- LaTeX zur Dokumentation ist sehr aufwändig zu lernen, da Lernkurve sehr steil, bietet jedoch vielerlei Vorteile gegenüber proprietären Office-Produkten
- Beim nächsten Projekt etwas mehr Zeit für Dokumentation einrechnen, um gegen Ende nicht in Zeitnot zu geraten

Anhang K Persönlicher Bericht Peter Manser

Anfangs Herbstsemester 2012 wurden wir mit der Thematik der Studienarbeit konfrontiert. Schnell wurde mir klar, dass ich nicht ein normales Thema aus einer Liste auswählen will, sondern die Initiative ergreiffen möchte und ein eigenes Thema eingeben will.

Die Voraussetzungen dazu waren eine überzeugende Idee, einen motivierten Dozenten und idealerweise noch einen Industriepartner zu finden.

Die Aufwand vor der Studienarbeit war nicht gerade gering, da wir uns bereits viele Gedanken über das Projekt (Thema) machen mussten und zudem noch unser Projekt bei verschiedenen Lehrern pitchen mussten. Glücklicherweise hat sich nach einem „Missverständnis“mit eindem anderen Dozenten, Prof. Dr. Markus Stolzebereit erklärt unsere Studienarbeit zu betreuen. Ein Industriepartner war schnell gefunden - ein renomiertes Hotel, das Hotel Hof Weissbad, aus meinem Heimatort.
Nach einigen Meetings und Skizzen war dann auch schon Semesterstart und somit Start der Studienarbeit. Mit vollem Elan sind wir ins Projekt gestartet und hatten bald einmal unsere ersten Zeilen Code geschrieben. Erwähnenswert für mich ist die Unterstützung durch unseren Betreuer, der sich immer die Mühe gemacht hat nicht nur zu beraten, sondern mitzudenken und Ideen miteinzubringen.
Das Ziel von uns war es stets das Projekt in der Bachelorarbeit weiter zu behandeln, das Produkt Business tauglich zu machen und nach dem Studium idealerweise daraus ein Startup zu machen. Dieser Wunsch wurde dann jedoch von einer neuen Reglementation der HSR zu Nichte gemacht. Somit wurde mir klar, dass die Arbeit nach der Studienarbeit nicht weitergeführt wird - was für mich ein rechter Dämpfer war.

Die Zusammenarbeit mit Moreno war sehr angenehm, wir haben uns gegenseitig stets motiviert eine gute Lösung zu bauen und mehr zu leisten, als „nötig“ist. Zudem konnten wir einander viele Sachen beibringen, da wir viel durch Pair Programming und Code Reviews viel Knowledge sharing betrieben haben.

Leider haben wir in der mittleren Phase des Projekts das Dokumentieren etwas vernachlässigt und zu sehr zur Seite geschoben. Das haben wir am Schluss stark gespührt. Das Testing kam ebenfalls viel zu kurz, was man erst am Schluss merkt - also dann, wenn es zu spät ist. Für die Bachelorarbeit setze ich mir daher das Ziel, mit Tests und Dokumentation früh zu beginnen und diese fortlaufend zu pflegen.

Alles in allem war es für mich eine sehr spannende Arbeit mit einem erfolgreichen Resultat. Die Arbeit durfte ich mit einem motivierten und unterstützenden Betreuer und einem perfekten Projektpartner absolvieren - so machen Projekte wirklich Freude!

K.1. Lessons Learned

- LaTeX ist eine gute Alternative zu Microsoft Word - braucht jedoch viel Geduld und Übung
- Apple's Submission Prozess ist ziemlich aufwändig und Bedarf einiger Erfahrung und Einarbeitung
- Scrum, leicht adoptiert, eignet sich auch hervorragend für Schulprojekte