



Semesterarbeit Informatik

Entwicklung einer interaktiven Mobile App mit Kommunikation via RESTful-Webservice zum Backend

Autor

Raffael Santschi
santsraf@students.zhaw.ch
Student im 7. Semester

Hauptbetreuung

Beat Seeliger
xsel@zhaw.ch

Abgabedatum

01.10.2014

Abstract

Der Turnverein Grafstal hat etwa 50 aktive Mitglieder und engagiert sich in der Leichtathletik. Der Verein verwaltet seit gut zwei Jahren seine Veranstaltungen und die dazugehörigen Anmeldungen auf einer Homepage. Im Zusammenhang mit den Feierlichkeiten des 125-jährigen Jubiläums wünschte er sich ein App.

Diese Semesterarbeit befasst sich mit der Erstellungen von dieser App, welche auf verschiedene Plattformen verfügbar sein sollte. Desweiteren war eine Fahrgemeinschaftsverwaltung für die Entlastung des Chats und Anmeldefunktionalität für die Veranstaltungen gewünscht.

Im Rahmen der Arbeit wurde die Stakeholdern evaluiert und mit diesen eine Anforderungsanalyse durchgeführt. Die Analyse brachte die wichtigsten Anforderungen an das App hervor, desweiteren wurden Mockups für die entstehenden Ansichten erstellt. Durch eine Nutzwertanalyse wurde die geeignete Architektur für das Apps ermittelt. Ausgehend von dieser Architektur wurde dann das App entwickelt

Bei der Ist-Analyse des Backends fiel auf, dass dieses keine klare Struktur aufwies, deshalb wurde ein Refactoring angedacht und durchgeführt. Im Anschluss wurde das Backend um ein RESTful API erweitert. Das resultierende App wurde mit dem Framework Phonegap und jQuery Mobile für Android und iOS erstellt. Die Daten für die verschiedenen Seiten werden per AJAX-Request über das RESTful API geladen. Desweiteren wurden die zwei Push-Nachrichten Dienste von Android und iOS in das App eingebunden. Eine Woche nach der Bereitstellung der App konnten bereits 50 Downloads verzeichnet werden. Die Mitglieder verwenden die Fahrgemeinschaftsverwaltung, der Chat wurde entlastet und die Anmeldungen an Veranstaltungen werden vermehrt über das App gemacht.

Vorwort

Apps sind aus der heutigen Zeit nicht mehr weg zu denken. Sie schaffen neue Möglichkeiten und erleichtern unser Leben. Sie sind jedoch auch mit neuen Pflichten verbunden, es wird von einem erwartet, dass man ständig online und erreichbar ist. Es wird von grösseren Dienstleistungsunternehmen in der heutigen Zeit erwartet, ein App zu besetzen, zumal inzwischen das Smartphone für unter 40jährige wichtiger ist als der Fernseher (siehe^[FH14]). An dem letzten Apple Event (siehe^[app14b]) wurde berichtet, dass bereits über 1.3 Millionen Apps im App Store sind. Mich faszinieren Apps schon eine ganze Weile, dies war mit einer der Gründe, warum ich an einem zweiwöchigen Austauschprojekt an der Grand Valley State University in Michigan teilnahm. Das Thema in diesem Austauschprogramm war Mobile App Entwicklung.

Das App für den Turnverein war für mich eine gute Gelegenheit für eine Semesterarbeit. Das Projekt hat mich von Anfang an gefesselt und war auch in der vorgegebenen Zeit umsetzbar. Während und nachdem Aufenthalt in Michigan hatten ich schon einmal mit einem Entwurf einer App angefangen, also wusste ich in etwa was auf mich zu kam.

Mein Dank gebührt meinen Kommilitonen, allen vorweg Roman Lickel, Max Schrimpf und Dominic Schlegel, für die Hilfe in technischen Fragen und Tipps zur Dokumentation. Ganz besonderes möchte ich mich bei Beat Seeliger für die gute Betreuung und seine konstruktive Kritik und bei meinen Korrekturlesern bedanken.

Erklärung betreffend das selbständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinar massnahmen der Hochschulordnung in Kraft.

Ort, Datum:

Unterschriften:

.....

.....

.....

.....

Das Original dieses Formulars ist bei der ZHAW-Version aller abgegebenen Projektarbeiten zu Beginn der Dokumentation nach dem Abstract bzw. dem Management Summary mit Original-Unterschriften und -Datum (keine Kopie) einzufügen.

Inhaltsverzeichnis

1	Projektübersicht	7
1.1	Ausgangslage	7
1.2	Ziele der Arbeit	7
1.3	Aufgabenstellung	7
1.4	Erwartete Resultate	8
1.5	Nicht-Ziele	8
1.6	Dokumentstruktur	8
2	Projektplanung	9
2.1	Meilensteine	9
2.2	Arbeitspakete	9
2.2.1	Planung	9
2.2.2	Requirement Engineering	10
2.2.3	Refactoring Backend	10
2.2.4	Entwicklung Mobile App	10
2.2.5	Tests	10
2.2.6	Dokumentation	10
2.3	Projektstrukturplan	11
2.4	Zeitplan	11
2.4.1	Geplante Abwesenheiten	11
2.4.2	Projektplan	12
2.4.3	Zeitschätzung auf Arbeitspaketebene	13
2.5	Risikoanalyse	13
2.5.1	Risikoermittlung	13
2.5.2	Risikobewertung	14
2.5.3	Risikomatrix	14
2.5.4	Massnahmen	15
3	Anforderungsdokument	17
3.1	Übersicht	17
3.1.1	System- und Kontextabgrenzung	17
3.1.2	Systemumgebung	18
3.1.3	Annahmen	18
3.1.4	Stakeholder	19
3.2	Anforderungen	20
3.2.1	Use Cases	20
3.2.2	Mockups	28
3.2.3	Anforderungen	33
4	Architektur	41
4.1	Übersicht	41
4.2	Backend	42
4.2.1	Locking	42
4.2.2	API	43
4.3	Lösungsvarianten	43
4.3.1	Native App	43
4.3.2	Xamarin	43
4.3.3	Phonegap	43

4.3.4	Nutzwertanalyse	44
4.4	Mobile App	46
5	Umsetzung des Prototypes	47
5.1	Entwicklungsumgebung	47
5.1.1	IDE - Integrated Development Environment	47
5.1.2	Versionierung	48
5.1.3	SDK - Software Development Kit	49
5.1.4	Webbrowser	49
5.1.5	Technische Geräte	49
5.1.6	Testen - Analysieren	50
5.2	Mobile App	52
5.2.1	Aufbau	52
5.2.2	Javascript Funktionen	57
5.2.3	Login	57
5.2.4	Provisioning Prozesse	57
5.3	Backend	58
5.3.1	Refactoring	58
5.3.2	Push-Nachrichten	60
6	Tests	61
6.1	Einführung	61
6.2	Testing	61
6.2.1	Testphasen	61
6.3	System- und Abnahmetest	62
6.3.1	Testprotokoll	62
6.4	Abnahme Protokoll	64
7	Schlussfolgerungen	65
7.1	Verwendung	65
7.2	Fazit	65
7.3	Ausblick	65
8	Verzeichnisse	67
	Literaturverzeichnis	67
	Abbildungsverzeichnis	71
	Tabellenverzeichnis	73
	Glossar	75
	Listingverzeichnis	77
A	Anhang	I
A.1	Projektmanagement	I

1 Projektübersicht

Die Übersicht dient dem Zweck einen generellen Überblick über das Dokument zu verschaffen. Sie beinhaltet die Ausgangslage, das Ziel dieser Arbeit, die Aufgabenstellung, die erwarteten Resultate und die Nicht-Ziele. Zusätzlich wird der Aufbau dieses Dokumentes erklärt.

1.1 Ausgangslage

Der Turnverein Grafstal wünscht sich auf sein 125-jähriges Jubiläum ein Mobile App. Der Verein hat bereits eine interaktive Webseite, welche in PHP geschrieben ist. Die Nutzung der Webseite wird immer grösser und bis jetzt werden ca. 800 Mitglieder damit verwaltet. In der letzten Zeit hat es sich eingebürgert, dass die Mitglieder sich per Whatsapp organisieren, wer ins Training fährt und wie viel Platz er noch im Auto hat. Dies und das Abrufen der wichtigsten Informationen sollen über das App möglich sein.

1.2 Ziele der Arbeit

Das Ziel ist eine Mobile App zu entwickeln, welche die wichtigsten Grundfunktionen (neueste Berichte anzeigen, Trainings-, Anlass-, Matchinformationen und die dazugehörigen An- und Abmeldefunktionen für die jeweiligen Events) abdeckt. Die wichtigste Funktion neben den Informationen ist die Fahrgemeinschaftsverwaltung. Durch diese Verwaltung soll der Whatsapp Chat entlastet und eine saubere Abwicklung gewährleistet werden. Der Verein erhofft sich vom App weiter eine höhere Anmeldungsquote, da das Anmelden durch wenige Klicks ermöglicht werden sollte. Zusätzlich kann bei der App die Push-Funktion genutzt werden, um Mitglieder auf Wichtiges aufmerksam zu machen. Für die Zukunft hat eine App weitere Vorteile, da auf GPS-Daten zugegriffen und offline Caching eingebaut werden kann, diese beiden Eigenschaften sind jedoch nicht Teil dieser Arbeit.

1.3 Aufgabenstellung

Folgende Punkte werden in der Semesterarbeit behandelt:

- Analysieren des Ist-Zustands
- Anforderungen an die Fahrgemeinschaftsverwaltung
- Entwicklung einer Mobile App mit Login
- Entwicklung einer Fahrgemeinschaftsverwaltung mit geeignetem Locking-Verfahren
- Abnahme der Mobile App durch den Vorstand

1.4 Erwartete Resultate

Folgende Punkte werden als Resultate der Semesterarbeit erwartet:

- Dokumentation des Ist-Zustandes
- Anforderungskatalog
- Beschreibung der Architektur und Locking-Verfahren bei der Fahrgemeinschaftsverwaltung
- Implementation der gewünschten Funktionen in einem Mobile App
- Live Schaltung des erweiterten Backends und Bereitstellung der App für den Download

1.5 Nicht-Ziele

Folgende Punkte wurden mit dem Auftraggeber als Nicht-Ziele definiert und sind somit nicht Teil dieses Projekts:

- Dieses Projekt erweitert die Funktionalität der Webseite nicht
- Es müssen keine Tests für die alten Funktionen geschrieben werden, ausser sie werden wegen diesem Projekt umgeschrieben
- Die Vermarktung des Apps ist Sache des Auftraggebers
- Lokales Speichern der Daten im App

1.6 Dokumentstruktur

Nachdem die Ausgangslage geklärt ist, kommen wir nun zum Aufbau dieses Dokuments. Dieses Dokument spiegelt die geleistete Arbeit wieder und ist in einzelne Kapitel unterteilt.

- Projektplanung: Schritte für die Erstellung des Projektplanes und Risikoanalyse
- Anforderungsdokument: System- und Kontextabgrenzung, Stakeholderliste, getroffene Annahmen und dann der Anforderungskatalog mit Use Cases, Mockups und Anforderungen
- Architektur: Übersicht über das ganze System, Architektur Beschreibung von Backend und App mit einer Nutzwertanalyse von den verschiedenen Lösungsvarianten
- Umsetzung: Beschreibung der Entwicklungsumgebung, Umsetzung von App und Backend
- Tests: Erläuterung der Test-Methoden und Abnahme Protokoll

Im Kapitel 8 sind alle Verzeichnisse zu finden, darunter auch ein Glossar, in welchem unbekannte Begriff erklärt werden. Falls es zu einem gewissen Begriff eine gängige Abkürzung gibt, wird diese beim ersten auftauchen des Wortes in Klammern geschrieben und danach verwendet, im Glossar finden sich dann beide Einträge.

2 Projektplanung

Dieses Kapitel handelt von der Projektplanung und den verschiedenen Arbeitspakete in diesem Projekt.

2.1 Meilensteine

Folgende Meilensteine wurden für dieses Projekt festgelegt:

Projektstart:	01.04.2014
Anforderungsdokument Entwurf	07.05.2014
Anforderungsdokument fertig	12.07.2014
Refactoring abgeschlossen	04.08.2014
App und Backend implementiert	30.08.2014
Test-Phase abgeschlossen	02.09.2014
App zur Freigabe eingesendet	02.09.2014
Dokumentation fertig	20.09.2014
Dokumentation korrigiert	27.09.2014

Tabelle 2.1: Meilensteine

2.2 Arbeitspakete

Das Projekt beinhaltet sechs Arbeitspakete:

- Planung
- Requirement Engineering
- Refactoring Backend
- Entwicklung Mobile App
- Testing
- Dokumentation

2.2.1 Planung

In der Planungsphase wird geschaut, was in dem Projekt erreicht werden muss und wie diese Tätigkeiten auf die vorhandene Zeit aufgeteilt wird. Es wird auch das erste Mal mit den Stakeholdern geredet und erste Abmachungen getroffen.

2.2.2 Requirement Engineering

Bei der Erstellung eines neuen Systems ist es immer wichtig, dass man alle Anforderungen kennt. Um die Anforderungen zu erfassen werden die Mitglieder und der Vorstand befragt, was sie sich von dem App wünschen. Die Mitglieder sind nur zum Teil Techniker, somit werden auch Anforderungen ein treffen, welche sehr schwierig um zu setzten sind und andere Anforderungen werden nicht aufgelistet, sondern einfach vorausgesetzt, sogenannte Basisfaktoren. Der Anforderungskatalog wird mit den Mitgliedern, welche sich eingebracht haben, angeschaut und am Schluss findet ein Review statt. (siehe dazu auch^[PR11])

2.2.3 Refactoring Backend

Als aller erste Entwicklungstätigkeit muss das Backend umgebaut werden, damit die Implementation für das Mobile App leichter fällt. Dazu wird der Ist-Stand aufgenommen und geschaut, welche Klassen umstrukturiert werden müssen, damit es danach leichter fällt. Danach wird entschieden, welches Pattern man beim Umbau der Klassen verfolgen möchte. Das Umbauen ist dann mit sehr viel Arbeit und Testen verbunden.

2.2.4 Entwicklung Mobile App

In diesem Arbeitspaket werden die Anforderungen umgesetzt. Das App wird entworfen und mit Inhalt gefüllt. Die ersten Tests werden durchgeführt und Unstimmigkeiten in den Anforderungen werden mit den Stakeholdern angeschaut.

2.2.5 Tests

Es gibt zwei Testsphasen, die eine ist dann, wenn das neue Backend online ist und die User, wie gewohnt, arbeiten können sollten. Die andere besteht darin, wenn die ersten Tests erfolgreich waren und das App für Android zum Download bereit gestellt wird. Nun können Mitglieder mit einem Android Device das App herunterladen und die ersten Geh-Versuche machen. Das Feedback wird bei beiden Phasen eingearbeitet und möglichst schnell aktualisiert.

2.2.6 Dokumentation

Die Dokumentation wird das ganze Projekt hindurch aktuell gehalten. Für das Erfassen dieses Dokuments wird \LaTeX und das \LaTeX -Template der ZHAW verwendet.

2.3 Projektstrukturplan

Anhand der Arbeitspaketen wurde ein Projektstrukturplan (siehe Abbildung 2.2) erstellt, welcher die Arbeitspakete in Teilaufgaben unterteilt. Diese Methode ist aus^[HMS09] bekannt.

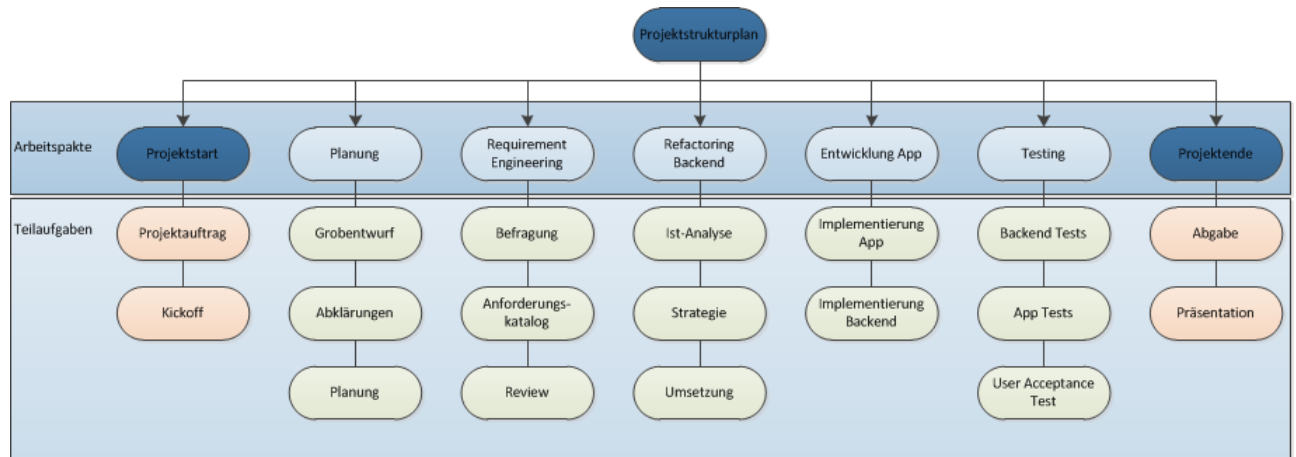


Abbildung 2.1: Projektstrukturplan

2.4 Zeitplan

Der Zeitplan gibt eine grobe Übersicht, wann an dem Projekt gearbeitet wird und wann die verschiedenen Tätigkeiten fertig werden. Die Angaben sind nur Richtwerte, da neben diesem Projekt noch einer Arbeit nachgegangen wird, verschiedenen Tätigkeiten im Verein ausgeführt werden und zusätzliche andere Arbeiten und Prüfungen Zeit benötigen.

2.4.1 Geplante Abwesenheiten

Abwesenheit	Start - Ende
Seminararbeit Abgaben	01.06.2014 - 22.06.2014
Modulprüfungen und Vorträge Seminararbeit	23.06.2014 - 02.07.2014
Ferien	12.07.2014 - 03.08.2014

Tabelle 2.2: Geplante Abwesenheiten

2.4.2 Projektplan

Der Zeitplan basierend auf dem Projektstrukturplan (siehe 2.3) und den geplanten Abwesenheiten sieht wie folgt aus:

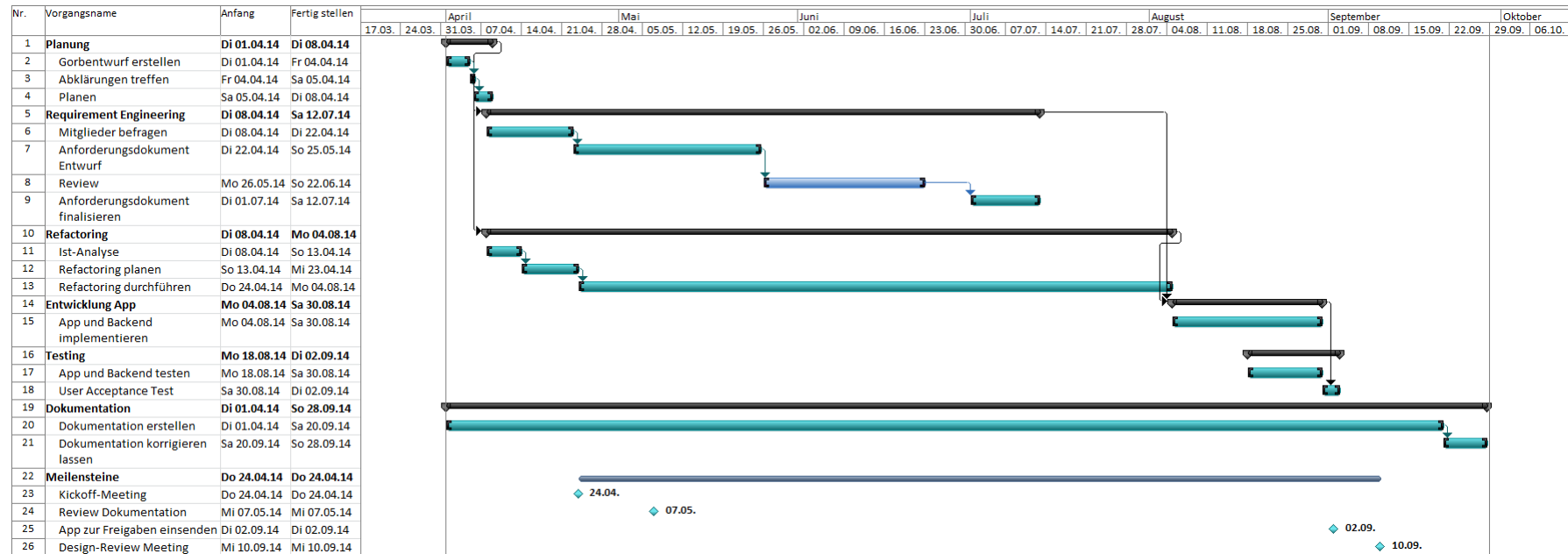


Abbildung 2.2: Projektplan

2.4.3 Zeitschätzung auf Arbeitspaketebene

Arbeitspaket	Schätzung (h)	Tatsächlich (h)
Planung	10	8
Requirement Engineering	12	15
Refactoring Backend	20	40
Entwicklung Mobile App	45	50
Tests	8	10
Dokumentation	60	70
Total	155	193

Tabelle 2.3: Zeitschätzung auf Arbeitspaketebene

2.5 Risikoanalyse

Jedes Projekt bringt Risiken mit sich, wenn diese nicht am Anfang analysiert und durch das ganze Projekt überprüft werden, kann es zu grossen Schwierigkeiten kommen. Die angewandten Methoden sind aus^[HHMS09] und aus dem dem Management Fach Projekt und Prozessmanagement bekannt.

2.5.1 Risikoermittlung

Die Risikoermittlung ist dafür da, dass man mögliche Risiken aufdeckt und die möglichen Folgen daraus festhält.

Risiko	Mögliche Folgen
Implementationsschwierigkeiten (Keine Erfahrung in der App Entwicklung)	<ul style="list-style-type: none"> ▪ Zeitplan nicht einhaltbar ▪ Einige Anforderungen müssen zurück gestellt werden ▪ Projektarbeit kann nicht durchgeführt werden
Ressourcen nicht verfügbar	<ul style="list-style-type: none"> ▪ Zeitplan nicht einhaltbar ▪ Einige Anforderungen müssen zurück gestellt werden ▪ Projektarbeit kann nicht durchgeführt werden
Mangelhaftes Endprodukt	<ul style="list-style-type: none"> ▪ Produkt muss überarbeitet werden, da keine Abnahme durch die Stakeholder erfolgt
Anforderungen nicht vollständig	<ul style="list-style-type: none"> ▪ Wichtige Funktionen stehen den Benutzern nicht zur Verfügung

Tabelle 2.4: Risikoermittlung

2.5.2 Risikobewertung

Das Schadensausmass und die Eintrittswahrscheinlichkeit der Risiken sind nach folgendem Schema bewertet worden:

Wert	Eintrittswahrscheinlichkeit	Schadensausmass
1	sehr unwahrscheinlich	vernachlässigbar
2	unwahrscheinlich	spürbar
3	wenig wahrscheinlich	verkraftbar
4	ziemlich wahrscheinlich	gefährlich
5	sehr wahrscheinlich	katastrophal

Tabelle 2.5: Risikobewertungsschema

Die Risiken aus der Risikobewertung (siehe 2.5.1) wurden anhand dieses Schemas bewertet.

$$\text{Risikofaktor} = \text{Eintrittswahrscheinlichkeit} * \text{Schadensausmass}$$

Risiko	Eintrittswahrscheinlichkeit	Schadensausmass	Risikofaktor
Implementationsschwierigkeiten	1	4	4
Ressourcen nicht verfügbar	2	5	10
Mangelhaftes Endprodukt	2	4	8
Anforderungen nicht vollständig	3	3	9

Tabelle 2.6: Risikobewertung

2.5.3 Risikomatrix

Anhand der Risikobeurteilung konnten die Risiken in eine Risikomatrix eingesetzt werden. Diese Matrix bietet einen guten Überblick über die Risiken und zeigt schnell, welche Risiken beachtet werden müssen.

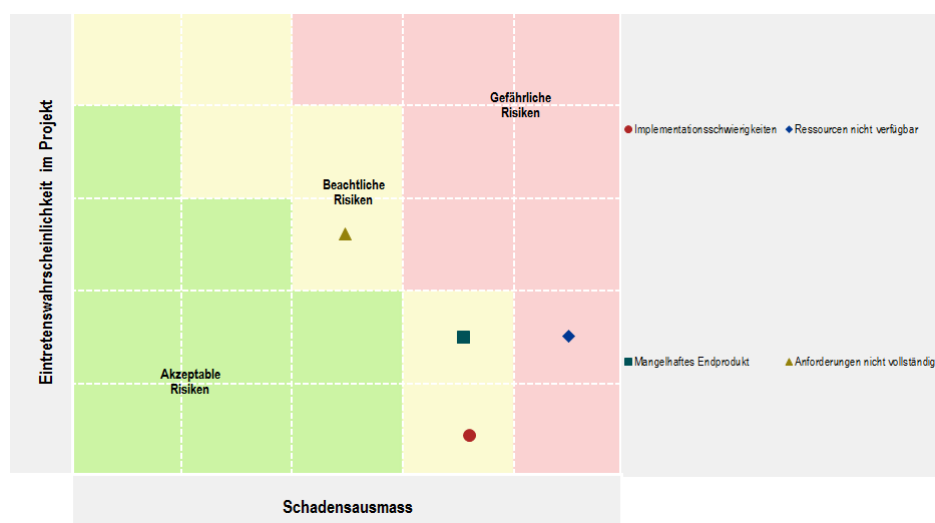


Abbildung 2.3: Risikomatrix

2.5.4 Massnahmen

Es wurden Massnahmen für die gefundenen Risiken gesucht und festgehalten. Die Massnahmen sind wiederum in vorbeugende Massnahmen und Eventualmassnahmen unterteilt.

Risiko	Massnahmen	
	Vorbeugende Massnahmen	Eventualmassnahmen
Implementationsschwierigkeiten	<ul style="list-style-type: none"> ▪ Im Zeitplan genügen Reserver einrechnen ▪ Kontaktpersonen zum Thema suchen 	<ul style="list-style-type: none"> ▪ Betreuer / Schulleitung informieren ▪ Verschiebungsgesuch stellen
Ressourcen nicht verfügbar	<ul style="list-style-type: none"> ▪ Fixe Zeiten einplanen ▪ Tätigkeiten priorisieren 	<ul style="list-style-type: none"> ▪ Verschiebungsgesuch mit guter Begründung stellen
Mangelhaftes Endprodukt	<ul style="list-style-type: none"> ▪ Anforderungskatalog sauber erstellen ▪ Testphase möglichst früh, damit noch genügen Zeit für die Einarbeitung des Feedbacks ist 	<ul style="list-style-type: none"> ▪ Lösung mit dem Kunden suchen ▪ Projekt verlängern
Anforderungen nicht vollständig	<ul style="list-style-type: none"> ▪ Review des Anforderungskataloges ▪ Testphase möglichst früh, damit noch genügen Zeit für die Einarbeitung des Feedbacks ist 	<ul style="list-style-type: none"> ▪ Anforderungen werden aufgenommen und in einen nächsten Release geplant

Tabelle 2.7: Risikoanalyse - Massnahmen

3 Anforderungsdokument

Das Anforderungsdokument legt die Basis für die Implementation. Es lohnt sich detaillierte Anforderungen zu erfassen, einerseits verhindert das Mögliche Missverständnis, andererseits ist es eine Absicherung, wenn der Kunde am Schluss etwas anderes behauptet.

3.1 Übersicht

In diesem Abschnitt wird die System- und Kontextabgrenzung dargelegt, die Systemumgebung beschrieben, die getroffenen Annahmen festgehalten und die verschiedenen Stakeholder mit ihren Erwartungen aufgelistet.

3.1.1 System- und Kontextabgrenzung

Der Systemkontext umfasst alle Aspekte, die für die Anforderungen des geplanten Systems relevant sind und nicht im Rahmen der Entwicklung dieses System gestaltet werden können.^[PR11]

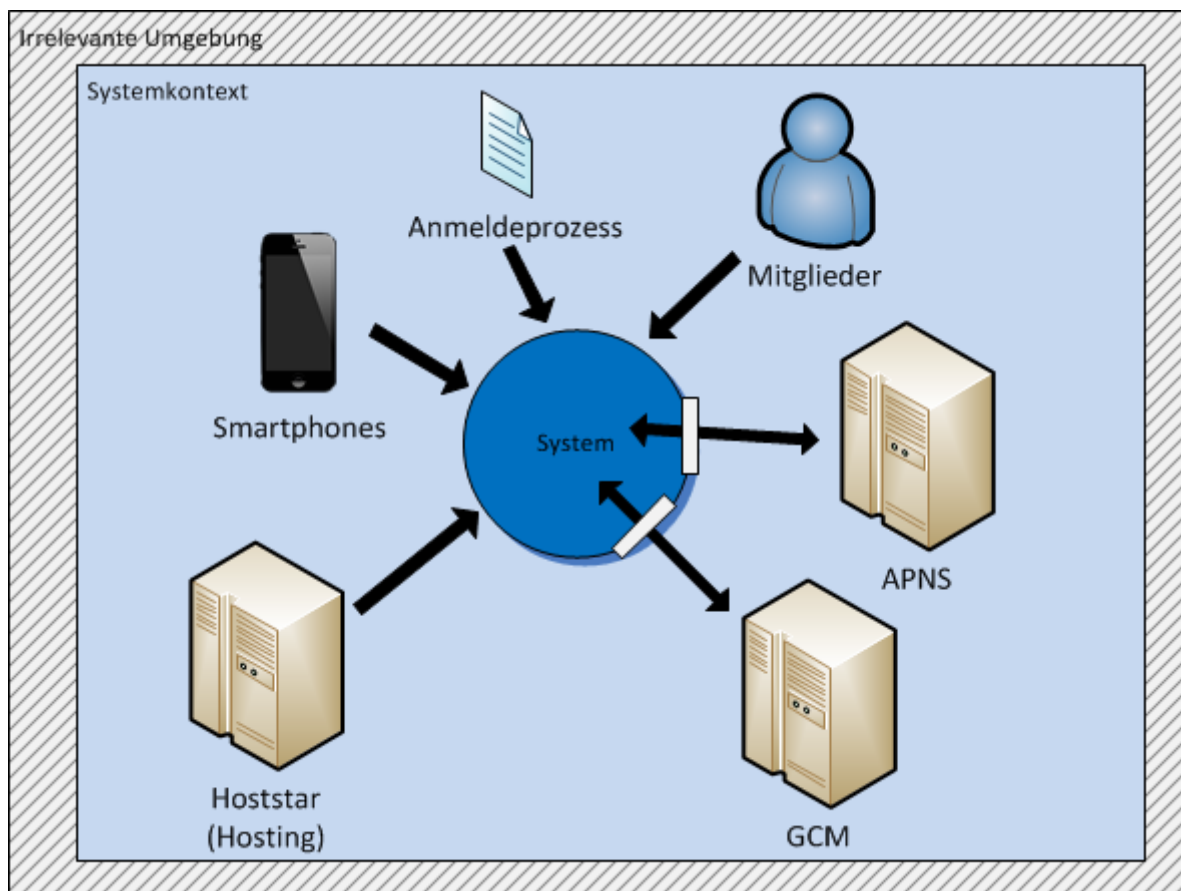


Abbildung 3.1: Systemkontext

Das System hat zwei relevanten Schnittstellen zur Aussenwelt, welche die Applikation beeinflussen, nämlich die beiden Push Notification Services Apple Push Notification Service (APNS) und Google Cloud Messaging for Android (GCM). Desweiteren wird das System von den verschiedenen Smartphone Typen, dem Anmeldeprozess, den Mitgliedern und nicht zu vergessen dem Hosting beeinflusst.

3.1.2 Systemumgebung

Die Systemumgebung definiert die Ausgangslage für das Projekt. Am Anfang des Projekts bestand das System aus einem Webserver, auf dem ein PHP Backend und interaktive Webseiten lief. Die Daten wurden in eine MySQL Datenbank abgespeichert. Die Seite wird rege benutzt, es sind etwa 35 Benutzer pro Tag online. Fast jedes Mitglied hat ein Smartphone, wobei die Aufteilung zwischen Android und iOS Geräten etwa ausgeglichen ist. In Abbildung 3.2 sieht man die Systemumgebung graphisch dargestellt.

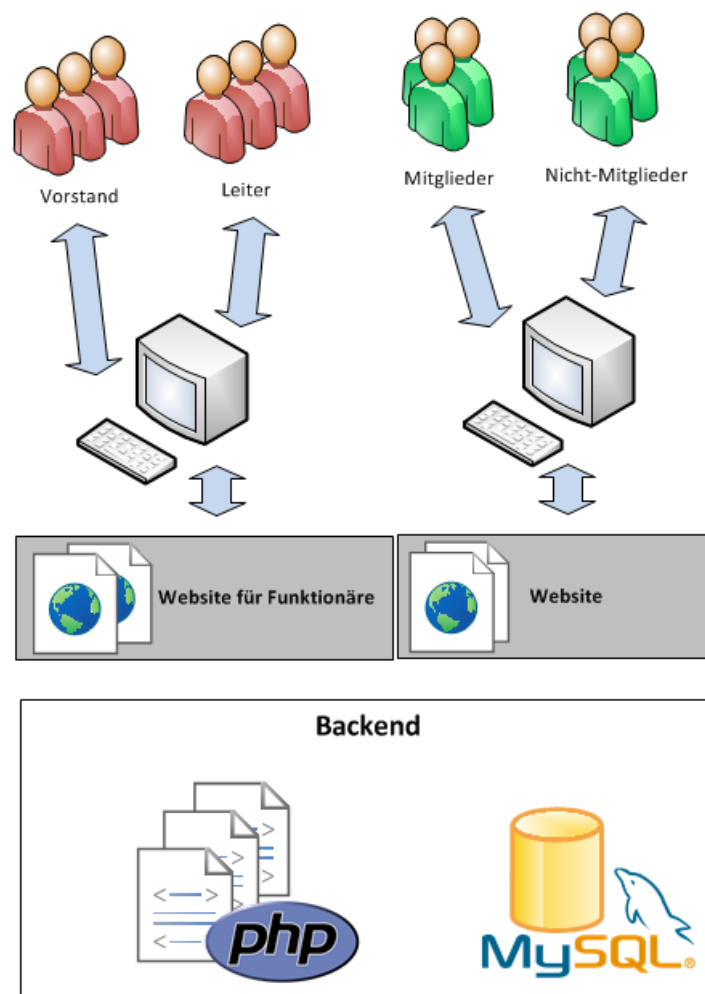


Abbildung 3.2: Systemumgebung

3.1.3 Annahmen

Es wurden keine Annahmen getroffen, durch die enge Zusammenarbeit mit dem Turnverein, konnten alle offenen Punkte in kürzester Zeit geklärt werden.

3.1.4 Stakeholder

Damit alle Anforderungen erfasst werden können, ist eine gute Stakeholder Analyse notwendig. In der Tabelle 3.1 wurden die Stakeholder für dieses Projekt zusammengetragen und ihre Erwartungen, ihre Einstellung und ihren Einfluss gegenüber diesem Projekt festgehalten.

Name	Erwartung	Einstellung	Einfluss
		-Positiv -Neutral -Negativ	-Hoch -Mittel -Niedrig
Vorstand	Der Vorstand ist der Kunde in diesem Projekt, er hat den Auftrag erteilt. Er erwartet vom App mehr Popularität vorallem bei den jüngeren Mitgliedern. Zusätzlich erhofft er sich etwas Werbung für den Verein.	Positiv	Hoch
Trainingsverantwortliche	Die Trainingsverantwortlichen sehen einen grossen Vorteil darin, dass die Anmeldung für die Trainings noch etwas vereinfacht werden soll. Sie finden es praktisch, dass sie die Trainingsinformationen schnell abrufen können.	Positiv	Hoch
aktive Mitglieder (U30)	Die jüngere Generation der aktiv Mitglieder warten gespannt auf das App. Es geht ihnen vorallem auch darum, einfach ein App von ihrem Verein zu haben.	Positiv	Hoch
aktive Mitglieder (Ü30)	Die ältere Generation der aktiv Mitglieder haben keine Erwartungen an das App, sie würden auch gut ohne auskommen.	Neutral	Mittel
Passiv- und Ehrenmitglieder	Passiv- und Ehrenmitglieder gehören auch eher zu dem älteren Semester und können mit Apps nicht viel anfangen. Sie sind dem App gegenüber jedoch nicht negativ eingestellt, zum Glück, da sie oft das Zünglein an der Waage sind.	Neutral	Mittel
Nicht-Mitglieder	Die Nicht-Mitglieder erhoffen sich schnelle und gute Informationen über Anlässe und die Vereine selbst.	Neutral	Niedrig

Tabelle 3.1: Stakeholder

3.2 Anforderungen

Bei dem Erfassen der Anforderungen wurde so vorgegangen, dass zuerst die Use-Cases definiert wurden. Diese wurden zusammen mit Vertretern des Turnvereins festgelegt. Die Beschreibung beinhaltet einen bereits klaren Ablauf für die einzelnen Use-Cases. Als nächstes wurden dann in mehreren Sitzungen die dazugehörigen Mockups gestaltet. Bevor man nun aber mit der Implementation anfangen konnte, mussten noch die Anforderungen abgeleitet werden. Die Anforderungen wurden anhand einer Mitglieder Befragung und einer kurzen Sitzung mit dem Vorstand evaluiert.

3.2.1 Use Cases

Das Use Case Diagramm (siehe Abbildung 3.3) zeigt zwei Akteure, Mitglieder und Nicht-Mitglieder und sechs Use Cases, welche in diesem Projekt umgesetzt wurden.

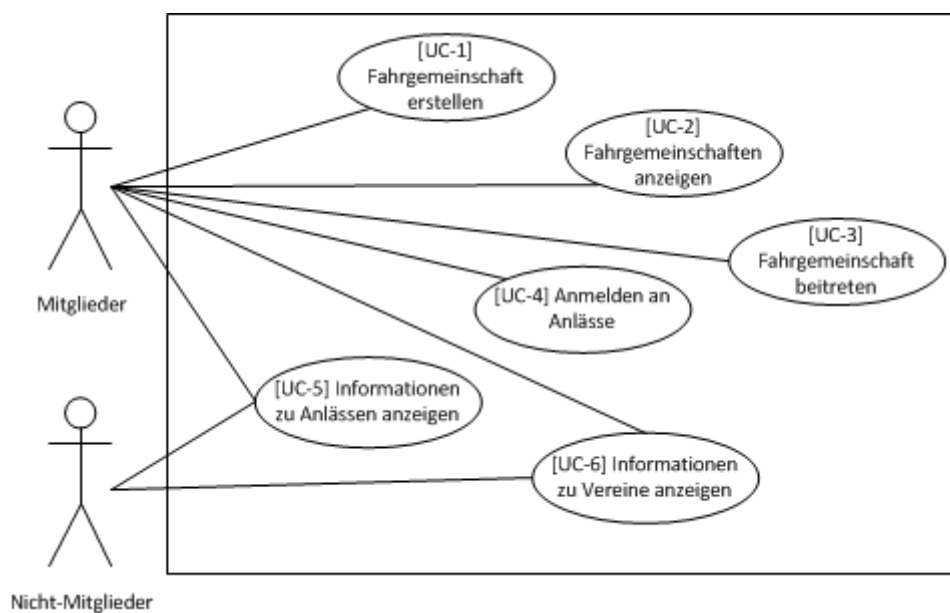


Abbildung 3.3: Use-Case Diagramm

Alle Use Cases wurden anhand der folgenden Vorlage (siehe Tabelle 3.2) spezifiziert. Diese Vorlage basiert auf Angaben von^[PR11]. Die Use Cases wurden möglichst genau beschrieben, was danach die Implementation vereinfachte.

Bezeichner	Eindeutiger Bezeichner
Name	Eindeutiger Name
Kritikalität	Kritikalität hinsichtlich des Schadensausmasses bei Fehlverhalten: hoch, mittel, leicht
Quelle	Stakeholder, Dokument, System
Verantwortlicher	Verantwortlicher Stakeholder
Beschreibung	Komprimierte Beschreibung
Auslösendes Ereignis	Angabe des Ereignisses, das den Use Case auslöst
Akteure	Auflistung der Akteure, die mit dem Use Case in Beziehung stehen
Vorbedingung	Eine Liste notwendiger Voraussetzungen, die erfüllt sein müssen, bevor die Ausführung des Use Case beginnen kann
Nachbedingung	Eine Liste von Zuständen, in denen sich das System unmittelbar nach der Ausführung des Hauptszenarios befindet.
Ergebnis	Beschreibung der Ausgaben, die während der Ausführung des Use Case erzeugt werden
Hauptszenario	Beschreibung des Hauptszenarios eines Use Case
Alternativszenarien	Beschreibung von Alternativszenarien des Use Case oder lediglich Angabe der auslösenden Ereignisse. Hier gelten oftmals andere Nachbedingungen.
Ausnahmeszenarien	Beschreibung von Ausnahmeszenarien des Use Case oder lediglich Angabe der auslösenden Ereignisse. Hier gelten oftmals andere Nachbedingungen.
Mockups	Querbezüge zu Mockups

Tabelle 3.2: Vorlage für Use Case Spezifikation

Bezeichner	UC-1
Name	Fahrgemeinschaft erstellen
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Marco Mathe
Beschreibung	Eine der Hauptaufgaben dieser App soll die Fahrgemeinschaftverwaltung sein. Jedes Mitglieder soll eine Fahrgemeinschaften erstellen können.
Auslösendes Ereignis	Mitglied möchte eine Fahrgemeinschaft erstellen
Akteure	TV-Mitglied
Vorbedingung	Das Mitglied ist angemeldet und es existiert eine Veranstaltung
Nachbedingung	Das Mitglied hat eine Fahrgemeinschaft erstellt
Ergebnis	Erstellung einer neuen Fahrgemeinschaft
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf eine der drei Veranstaltungstypen 4. Das System fragt eine Liste der Veranstaltungen beim Backend ab 5. Das System zeigt eine Liste der Veranstaltungen an 6. Das Mitglied sucht die Veranstaltung und klickt darauf 7. Das Mitglied klickt auf 'Fahrgemeinschaften' 8. Das Mitglied klickt auf 'Neue Fahrgemeinschaft' 9. Das System zeigt ein Formular mit Daten zur Fahrgemeinschaft an 10. Das Mitglied füllt die Felder aus und sendet das Formular ab 11. Das System schickt die Daten an das Backend 12. Das System zeigt 'Fahrgemeinschaft erfolgreich erstellt' an
Alternativszenarien	7a Das System weist das Mitglied auf fehlende Felder hin
Ausnahmeszenarien	7a Das System zeigt 'keine Internetverbindung vorhanden' an
Mockups	-

Tabelle 3.3: Use Case UC-1: Fahrgemeinschaft erstellen

Bezeichner	UC-2
Name	Fahrgemeinschaften anzeigen
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Marco Mathe
Beschreibung	Eine der Hauptaufgaben dieser App soll die Fahrgemeinschaftverwaltung sein. Die Mitglieder sollen sich eine Liste aller Fahrgemeinschaften anzeigen lassen können, um dann zu entscheiden, welcher sie sich anschliessen möchten.
Auslösendes Ereignis	Mitglied möchte Fahrgemeinschaften anschauen
Akteure	TV-Mitglied
Vorbedingung	Das Mitglied ist angemeldet und Fahrgemeinschaften sind vorhanden
Nachbedingung	Das Mitglied hat eine Liste von Fahrgemeinschaften erhalten
Ergebnis	Liste von Fahrgemeinschaften
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf eine der drei Veranstaltungstypen 4. Das System fragt eine Liste der Veranstaltungen beim Backend ab 5. Das System zeigt eine Liste der Veranstaltungen an 6. Das Mitglied sucht die Veranstaltung und klickt darauf 7. Das Mitglied klickt auf 'Fahrgemeinschaften' 8. Das System fragt eine Liste der Fahrgemeinschaften beim Backend ab 9. Das System zeigt eine Liste der Fahrgemeinschaften an
Alternativszenarien	5a Das System zeigt 'keine aktuellen Fahrgemeinschaften' an
Ausnahmeszenarien	4a Das System zeigt 'keine Internetverbindung vorhanden' an
Mockups	3.4(a), 3.11(a), 3.11(b), 3.12

Tabelle 3.4: Use Case UC-2: Fahrgemeinschaften anzeigen

Bezeichner	UC-3
Name	Fahrgemeinschaft beitreten
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Marco Mathe
Beschreibung	Eine der Hauptaufgaben dieser App soll die Fahrgemeinschaftverwaltung sein. Beim Beitreten einer Fahrgemeinschaft geht es darum, dass das Mitglied Interesse an der Fahrgemeinschaft hat und dieser gern beitreten möchte.
Auslösendes Ereignis	Mitglied möchte einer Fahrgemeinschaft beitreten
Akteure	TV-Mitglied
Vorbedingung	Das Mitglied ist angemeldet und Fahrgemeinschaften sind vorhanden
Nachbedingung	Das Mitglied ist einer Fahrgemeinschaft beigetreten
Ergebnis	Neuer Eintrag in der Gemeinschaft und Kapazitätenminderung
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf eine der drei Veranstaltungstypen 4. Das System fragt eine Liste der Veranstaltungen beim Backend ab 5. Das System zeigt eine Liste der Veranstaltungen an 6. Das Mitglied sucht die Veranstaltung und klickt darauf 7. Das Mitglied klickt auf 'Fahrgemeinschaften' 8. Das System zeigt eine Liste aller Fahrgemeinschaften an 9. Das Mitglied wählt eine aus und klickt auf 'Beitreten' 10. Das System zeigt 'Erfolgreich beigetreten' an
Alternativszenarien	<ol style="list-style-type: none"> 3a Das System zeigt 'Kein Platz mehr' an 4 Der User wählt eine andere Fahrgemeinschaft aus und versucht es erneut
Ausnahmeszenarien	<ol style="list-style-type: none"> 3a Das System zeigt 'keine Internetverbindung vorhanden' an
Mockups	3.4(a), 3.11(a), 3.11(b)

Tabelle 3.5: Use Case UC-3: Fahrgemeinschaft beitreten

Bezeichner	UC-4
Name	Anmelden an Anlässe
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Dominic Keller
Beschreibung	Die bekannte Anmelde-Funktion von der Homepage soll es auch beim App geben.
Auslösendes Ereignis	Mitglied möchte sich für einen Anlass anmelden
Akteure	TV-Mitglied
Vorbedingung	Das Mitglied ist angemeldet
Nachbedingung	Das Mitglied hat sich für einen Anlass angemeldet
Ergebnis	Erstellung einer neuen Anmeldung für den Anlass
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf 'Anlässe' 4. Das System fragt eine Liste der Anlässe beim Backend ab 5. Das System zeigt eine Liste der Anlässe an 6. Das Mitglied sucht den Anlass und klickt darauf 7. Das Mitglied klickt auf 'Anmelden' 8. Das System schickt die Anmeldung zum Backend 9. Das System zeigt 'Anmeldung erfolgreich' an
Alternativszenarien	<p>7a Das Mitglied sucht den Anlass und kann nicht auf 'Anmelden' klicken, da es bereits angemeldet ist</p>
Ausnahmeszenarien	<p>8a Das System zeigt 'keine Internetverbindung vorhanden' an</p>
Mockups	3.4(a), 3.7(a), 3.7(b), 3.8(a), 3.8(b), 3.9(a), 3.9(b)

Tabelle 3.6: Use Case UC-4: Anmelden an Anlässe

Bezeichner	UC-5
Name	Informationen zu Anlässen anzeigen
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Ivan Sebastiano
Beschreibung	Neben Funktionen für die Mitglieder soll die App auch Informationen über bevorstehende Anlässe für Mitglieder und Nicht-Mitglieder zur Verfügung stellen.
Auslösendes Ereignis	Mitglied möchte Informationen über einen bevorstehenden Anlass
Akteure	TV-Mitglied, Nicht Mitglieder
Vorbedingung	Anlässe sind vorhanden
Nachbedingung	Das Mitglied hat die gewünschten Informationen erhalten
Ergebnis	Informationen über den bevorstehenden Anlass
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf 'Anlässe' 4. Das System fragt eine Liste der Anlässe beim Backend ab 5. Das System zeigt eine Liste der Anlässe an 6. Das Mitglied sucht den Anlass und klickt auf ihn 7. Das System schickt die Anfrage an das Backend 8. Das System zeigt detaillierte Informationen über den Anlass an
Alternativszenarien	<ol style="list-style-type: none"> 8a Das System zeigt detaillierte Informationen über den Anlass und spezielle Informationen für Mitglieder an
Ausnahmeszenarien	<ol style="list-style-type: none"> 7a Das System zeigt 'keine Internetverbindung vorhanden' an
Mockups	3.4(a), 3.7(a), 3.7(b), 3.8(a), 3.8(b), 3.9(a), 3.9(b)

Tabelle 3.7: Use Case UC-5: Informationen zu Anlässen anzeigen

Bezeichner	UC-6
Name	Informationen zu Vereinen anzeigen
Kritikalität	hoch
Quelle	Stakeholder
Verantwortlicher	Ivan Sebastiano
Beschreibung	Mit Hilfe von der App sollen Informationen über die Vereine und Riegen, wie zum Beispiel Trainingszeiten und Trainingsinhalte, einfach und rasch zu finden sein.
Auslösendes Ereignis	Mitglied möchte Informationen über einen Verein erhalten
Akteure	TV-Mitglied, Nicht Mitglieder
Vorbedingung	-
Nachbedingung	Das Mitglied hat die gewünschten Informationen erhalten
Ergebnis	Informationen über den Verein
Hauptszenario	<ol style="list-style-type: none"> 1. Das Mitglied öffnet den Menü-Knopf 2. Das System zeigt das Menü an 3. Das Mitglied klickt auf 'Vereine' 4. Das Mitglied wählt nun den gewünschten Verein aus 5. Das System fragt die Informationen zum Verein beim Backend ab 6. Das System zeigt eine Liste der Informationen an
Alternativszenarien	-
Ausnahmeszenarien	<ol style="list-style-type: none"> 5a Das System zeigt 'keine Internetverbindung vorhanden' an
Mockups	3.4(a), 3.6(a)

Tabelle 3.8: Use Case UC-6: Informationen zu Vereinen anzeigen

3.2.2 Mockups

Um auf einer gemeinsamen Ebene mit dem Kunden zu diskutieren, wurden Mockups von den verschiedenen Seiten erstellt. Dies beugt einerseits Missverständnissen vor und ist für visuelle Typen sehr ansprechend. Die Mockups wurden in einer kleinen Gruppe erstellt und die Use Cases dienten als Vorlage. Die Zeichnungen waren zuerst grob auf Papier gezeichnet und wurden danach mit Hilfe von mybalsamiq (siehe^[zha14] bzw. ^[myb14]) digitalisiert. Bei den Mockups wurde mit möglichst ähnlichem und bereits bekanntem Aufbau und Elementen gearbeitet. Die Übersicht und das schnelle Zurechtfinden waren die Hauptkriterien, die Heuristiken von Nielsen von ^[MN90] waren dabei eine grosse Hilfe.

Navigation - Einstellungen

Die Navigation und die Einstellungen sind die Grundstruktur der App und deshalb wurden sie zuerst gemacht.

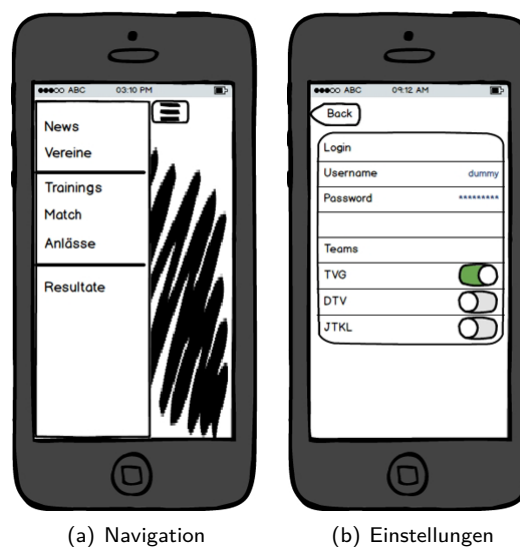


Abbildung 3.4: Navigation und Einstellungen

Neuigkeiten

Als Startseite sollte eine Seite gewählt werden, welche sich stetig verändert, somit fiel die Entscheidung auf die News Seite.



Abbildung 3.5: Neuigkeiten

Vereine

Die Gedanken zur Gestaltung der Informationen der Vereine sahen so aus:



Abbildung 3.6: Vereine

Veranstaltungen

Die verschiedenen Übersichten und Detailsichten der Veranstaltungen sollten ursprünglich so aussehen:

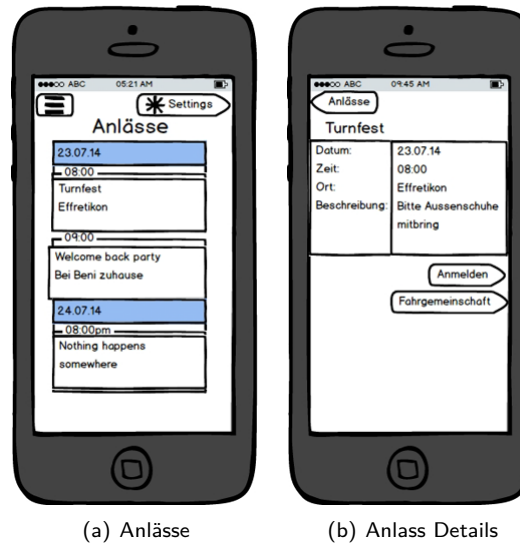


Abbildung 3.7: Anlässe

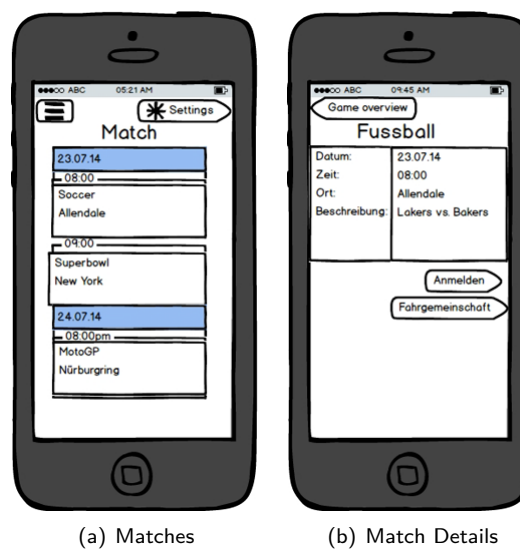


Abbildung 3.8: Matches

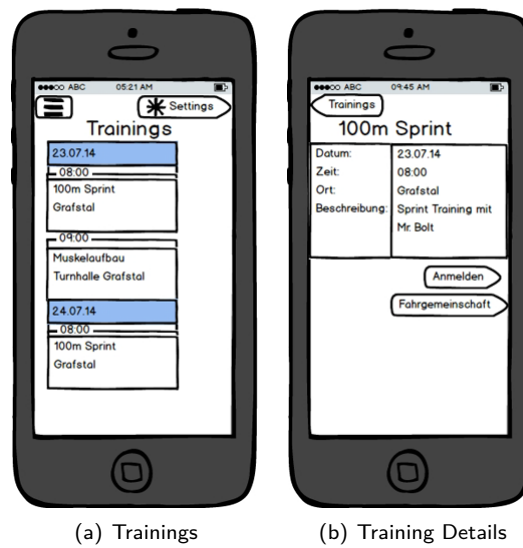


Abbildung 3.9: Trainings

Resultate

Das Mockup für die Resultat Darstellung sah eine veranstaltungsspezifische Darstellung vor, wurde dann im späteren Verlauf des Projektes auf eine personenspezifische Übersicht abgeändert. Diese Ansicht war kein Use Case der Stakeholder und wurde als möglicher Begeisterungsfaktor vom Entwickler hinzugefügt.

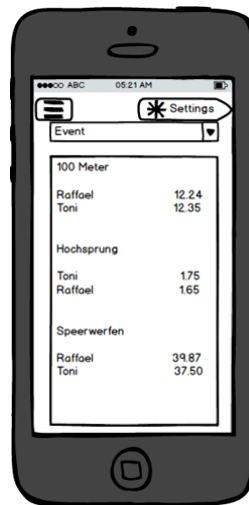


Abbildung 3.10: Resultate

Fahrgemeinschaften

Eine der wichtigsten Ansichten, welche nichts vergleichbares auf der Webseite hat, war jene von der Fahrgemeinschaft. Hier wurden schon die ersten Überlegungen getroffen, was für Attribute eine Fahrgemeinschaft benötigt.

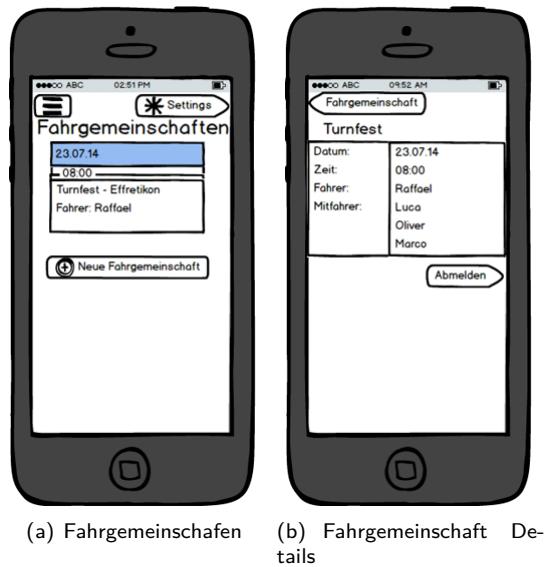


Abbildung 3.11: Fahrgemeinschaften



Abbildung 3.12: Fahrgemeinschaft erstellen

3.2.3 Anforderungen

Alle Anforderungen wurden anhand der folgenden Vorlage (siehe Tabelle 3.9) erfasst. Diese Vorlage basiert auf Angaben von^[PR11] und wurde mit eigenen Attributen erweitert.

ID	Eindeutiger Identikator
Priorität	Must, Should, Nice to have
Anforderungstyp	Funktional Anforderung, Qualitätsanforderung, Randbedingung
Name	Eindeutiger, charakterisierender Name
Use Case	Referenz zum zugehörigen Use Case
Beschreibung	Beschreibung der Anforderung
Begründung	Bedeutung der Anforderung für das geplante System
Akzeptanz Kriterium	Messbare Abnahmekriterien
Abhängigkeiten	Referenz zu anderen Anforderungen
Antragssteller	Stakeholder Gruppe, die um diese Anforderung bittet
Risiken	Weisst auf allfällige Risiken betreffend der Anforderung hin

Tabelle 3.9: Vorlage für Anforderungen

Die Beschreibung der Anforderungen wurden zusätzlich mit einer Satzschablone (siehe Abbildung 3.13) aus^[PR11] erstellt. Dies hat den Vorteil, dass die Anforderungen normiert und exakt da herkommen. Die

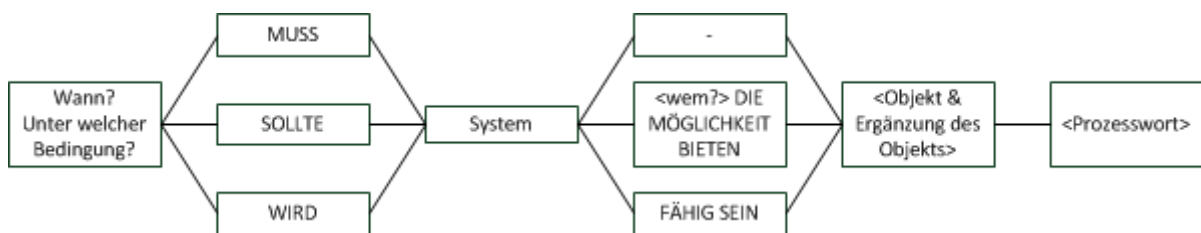


Abbildung 3.13: Satzschablone

Abstufungen muss, sollte und wird werden verwendet um die Wichtig der Anforderungen auszudrücken.

Funktional Anforderungen

ID	RE-F1
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Login Möglichkeit bei der Mobile App
Use Case	-
Beschreibung	Falls ein Mitglied vom Turnverein auf der Homepage registriert ist, muss das System dem besagten Mitglied die Möglichkeit bieten sich über ein Formular am Backend anzumelden.
Begründung	Einige Informationen dürfen nur für Mitglieder zugänglich gemacht werden. Zusätzlich vereinfacht es das Anmelden, Erstellen und Abrufen von benutzerspezifischen Daten
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Bei bekanntem Username und Passwort muss es dem Mitglied möglich sein, sich beim Backend anzumelden und dann erweiterte Möglichkeiten und Informationen zu erhalten. 2. Falls ein unbekannter Username oder das falsche Passwort eingegeben wurden kommt eine entsprechende Fehlermeldung.
Abhängigkeiten	-
Antragssteller	Dominic Keller
Risiken	Falls diese Anforderung nicht erfolgreich implementiert werden kann, gibt es Schwierigkeiten bei sehr vielen anderen Anforderungen

Tabelle 3.10: Anforderung RF-F1

ID	RE-F2
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Fahrgemeinschaft erstellen
Use Case	Use Case UC-1: Fahrgemeinschaft erstellen
Beschreibung	Falls das Mitglied angemeldet ist, muss das System dem besagten Mitglied die Möglichkeit bieten über ein Formular eine neue Fahrgemeinschaft zu eröffnen.
Begründung	Seit einiger Zeit wird vermehrt der Vereins-Chat mit endlosen Diskussionen für Fahrgemeinschaftsorganisationen gebraucht, was einige Mitglieder stört und eigentlich auch nicht der Sinn und Zweck dieses Chats ist. Jedes Mitglied sollte über das App die Möglichkeit haben seine Fahrdienste zur Verfügung zu stellen.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Das angemeldete Mitglied kann eine Fahrgemeinschaft eröffnen 2. Bei ungültigen oder nicht ausreichenden Informationen kommt eine entsprechende Fehlermeldung
Abhängigkeiten	Anforderung RF-F1
Antragssteller	Marco Mathe
Risiken	-

Tabelle 3.11: Anforderung RF-F2

ID	RE-F3
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Übersicht über Fahrgemeinschaft
Use Case	Use Case UC-2: Fahrgemeinschaften anzeigen
Beschreibung	Falls das Mitglied angemeldet ist und eine Fahrgemeinschaft im System besteht, muss das System dem besagten Mitglied die Möglichkeit bieten eine Liste aller aktuellen Fahrgemeinschaften anzeigen zu lassen.
Begründung	Das Mitglied benötigt eine Übersicht der aktuellen Fahrgemeinschaften, damit es sich für eine entscheiden kann.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Das angemeldete Mitglied kann die Fahrgemeinschaften anzeigen lassen 2. Das angemeldete Mitglied sieht einen Hinweis, falls es keine Fahrgemeinschaften gibt
Abhängigkeiten	Anforderung RF-F1, Anforderung RF-F2
Antragssteller	Marco Mathe
Risiken	Falls nur die Anforderung RF-F2 implementiert wird und diese nicht, ist das Fahrgemeinschaft-System nicht funktionsfähig

Tabelle 3.12: Anforderung RF-F3

ID	RE-F4
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Fahrgemeinschaft beitreten
Use Case	Use Case UC-3: Fahrgemeinschaft beitreten
Beschreibung	Falls das Mitglied angemeldet ist, eine Fahrgemeinschaft im System besteht, das besagte Mitglied noch nicht beigetreten ist und die Fahrgemeinschaft noch Kapazität aufweist, muss das System dem besagten Mitglied die Möglichkeit bieten sich über einen Knopf bei einer Fahrgemeinschaft einzutragen.
Begründung	Damit die Fahrgemeinschaften auch funktionieren, muss es den Mitgliedern möglich sein, sich selbständig bei den Fahrgemeinschaften einzutragen.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Das angemeldete Mitglied kann einer Fahrgemeinschaft beitreten 2. Das angemeldete Mitglied kann einer Fahrgemeinschaft nicht erneut beitreten 3. Das angemeldete Mitglied erhält eine Fehlermeldung falls die Fahrgemeinschaft keine Kapazität mehr aufweist
Abhängigkeiten	Anforderung RF-F1, Anforderung RF-F2, Anforderung RF-F3
Antragssteller	Marco Mathe
Risiken	Falls nur die Anforderung RF-F2 und Anforderung RF-F3 implementiert werden und diese nicht, ist das Fahrgemeinschaft-System nicht funktionsfähig

Tabelle 3.13: Anforderung RF-F4

ID	RE-F5
Priorität	Should
Anforderungstyp	Funktional Anforderung
Name	Anmelden an Vereinsanlässe
Use Case	Use Case UC-4: Anmelden an Anlässe
Beschreibung	Falls das Mitglied angemeldet ist, ein Anlass im System besteht und das besagte Mitglied noch nicht angemeldet ist, sollte das System dem besagten Mitglied die Möglichkeit bieten sich über einen Knopf für den Anlass anzumelden.
Begründung	Die Anmeldequote der Mitglieder ist nicht so hoch wie erwünscht, durch das Anbieten der Anmelde-Möglichkeit über das App soll das Anmelden vereinfacht und die Anmeldequote erhöht werden.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Das angemeldete Mitglied kann sich für einen Anlass anmelden 2. Das angemeldete Mitglied kann sich für einen Anlass nicht erneut anmelden
Abhängigkeiten	Anforderung RF-F1, Anforderung RF-F2, Anforderung RF-F3
Antragssteller	Dominic Keller
Risiken	-

Tabelle 3.14: Anforderung RF-F5

ID	RE-F6
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Informationen über Vereinsanlässe
Use Case	Use Case UC-5: Informationen zu Anlässen anzeigen
Beschreibung	Falls ein ein Anlass im System ist, muss das System dem Benutzer der App die Möglichkeit Informationen über Vereinsanlässe zu erhalten.
Begründung	Über die App kann man die Informationen schnell und einfach abrufen, somit sollte den Anlässe mehr Beachtung geschenkt werden.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Der Benutzer erhält Informationen über den ausgewählten Anlass 2. Der Benutzer sieht einen Hinweis, falls keine Anlässe vorhanden sind
Abhängigkeiten	-
Antragssteller	Ivan Sebastiano
Risiken	-

Tabelle 3.15: Anforderung RF-F6

ID	RE-F7
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Erweiterte Informationen über Vereinsanlässe
Use Case	Use Case UC-5: Informationen zu Anlässen anzeigen
Beschreibung	Falls das Mitglied angemeldet ist und ein Anlass im System besteht, sollte das System dem besagten Mitglied die Möglichkeit bieten zusätzliche Informationen über den Anlass zu erhalten.
Begründung	Einige Informationen (z. B. Anmelde-Listen) sind aus Datenschutz oder anderen Gründen nicht für alle Nutzer der App bestimmt und sollten nur angemeldeten Mitgliedern angezeigt werden.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Das angemeldete Mitglied sieht die erweiterten Informationen zum Anlass 2. Nicht angemeldete Mitglieder sehen die erweiterten Informationen zum Anlass nicht
Abhängigkeiten	Anforderung RF-F1, Anforderung RF-F6
Antragssteller	Ivan Sebastiano
Risiken	-

Tabelle 3.16: Anforderung RF-F7

ID	RE-F8
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Informationen über Vereine und Riegen
Use Case	Use Case UC-6: Informationen zu Vereinen anzeigen
Beschreibung	Das System muss dem Benutzer der App die Möglichkeit bieten Informationen über die verschiedenen Vereine und Riegen zu erhalten.
Begründung	Die App ist nicht nur für die Mitglieder des Vereins sondern auch für Aussenstehende und sollte auch zu Werbezwecken dienen. Jeder kann somit aktuelle Trainingszeiten und Inhalte der Trainings anschauen und erhält Kontaktinformationen.
Akzeptanz Kriterium	<ol style="list-style-type: none"> 1. Der Benutzer erhält Informationen über die verschiedenen Vereine und Riegen
Abhängigkeiten	-
Antragssteller	Ivan Sebastiano
Risiken	-

Tabelle 3.17: Anforderung RF-F8

ID	RE-F9
Priorität	Must
Anforderungstyp	Funktional Anforderung
Name	Aktuellen Berichte
Use Case	Use Case UC-6: Informationen zu Vereinen anzeigen
Beschreibung	Das System muss dem Benutzer der App die Möglichkeit bieten aktuellen Berichte von den einzelnen Anlässen oder Vereine lesen zu können.
Begründung	Die App ist nicht nur für die Mitglieder des Vereins sondern auch für Aussenstehende und sollte auch zu Werbezwecken dienen. In den Berichten wird über aktuelle Anlässe und Geschehnisse geschrieben, ganz nach dem Motto 'Tue gutes und sprich darüber'.
Akzeptanz Kriterium	1. Der Benutzer kann die aktuellen Berichte lesen
Abhängigkeiten	-
Antragssteller	Dominic Keller
Risiken	-

Tabelle 3.18: Anforderung RF-F9

ID	RE-F10
Priorität	Nice to have
Anforderungstyp	Funktional Anforderung
Name	Push-Nachrichten versenden
Use Case	-
Beschreibung	Falls das Mitglied angemeldet ist, Push-Nachrichten zu gelassen werden und eine solche vom Backend versendet wurde, sollte das System dem besagten Mitglied eine Push Notification zu stellen.
Begründung	Trainingsverantwortliche und Vorstandsmitglieder können die Mitglieder mit Push-Nachrichten schneller über kurzfristige Änderungen informieren.
Akzeptanz Kriterium	1. Das angemeldete Mitglied erhält die Push-Nachrichten
Abhängigkeiten	Anforderung RF-F1
Antragssteller	Oliver Zimmermann
Risiken	-

Tabelle 3.19: Anforderung RF-F10

Qualitätsanforderung

ID	RE-NF1
Priorität	Must
Anforderungstyp	Qualitätsanforderung
Name	Unterstützung von Android
Use Case	-
Beschreibung	Das System muss fähig sein auf Android Geräten zu laufen.
Begründung	Die Statistik der aktuellen Homepage zeigt, dass über 50% der Zugriffe über Mobiltelefone und Tablet von Android Plattformen stammen.
Akzeptanz Kriterium	1. Ein Nutzer mit einem Android Mobiltelefon kann die App öffnen und verwenden
Abhängigkeiten	-
Antragssteller	Dominic Keller
Risiken	-

Tabelle 3.20: Qualitätsanforderung RF-NF1

ID	RE-NF2
Priorität	Must
Anforderungstyp	Qualitätsanforderung
Name	Unterstützung von iOS
Use Case	-
Beschreibung	Das System muss fähig sein auf iOS Geräten zu laufen.
Begründung	Die Statistik der aktuellen Homepage zeigt, dass gut 43% der Zugriffe über Mobiltelefone und Tablet von iOS Plattformen stammen.
Akzeptanz Kriterium	1. Ein Nutzer mit einem iPhone kann die App öffnen und verwenden
Abhängigkeiten	-
Antragssteller	Marco Mathe
Risiken	-

Tabelle 3.21: Qualitätsanforderung RF-NF2

4 Architektur

In diesem Kapitel wird auf die ganze Struktur und die Architektur des Backends und des Apps eingegangen. Die Architektur legt die Grundlage für die Umsetzung und sollte mit Bedacht festgelegt werden.

4.1 Übersicht

Die Systemumgebung (siehe Abbildung 4.1) besteht aus einem Webserver, auf welchem das Backend, geschrieben in PHP, die Webseite und neu auch die API-Schnittstelle für das App läuft. Die Web-Schnittstelle für Funktionäre wird in diesem Bild separat aufgeführt, da sie viel mehr Funktionen bietet, als für normale Mitglieder und Nicht-Mitglieder, sie ist jedoch über die gleiche Webseite ansprechbar. Die RESTful API, mit welcher das App kommuniziert, beherrscht momentan nur die in den Anforderungen benötigten Funktionen.

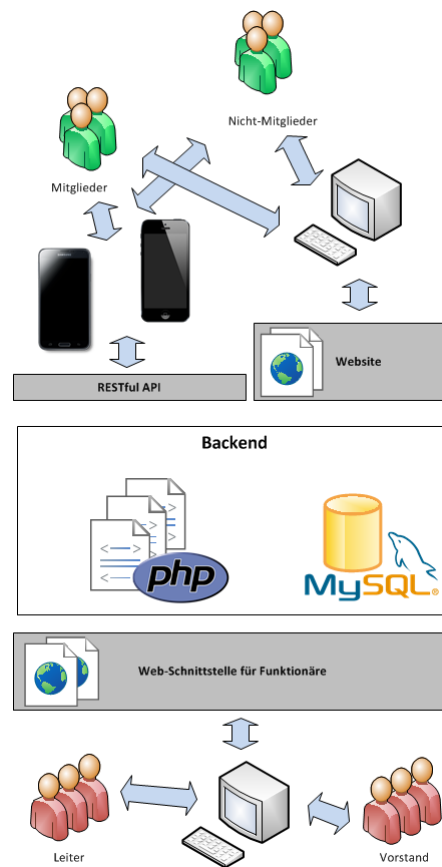


Abbildung 4.1: System Übersicht

4.2 Backend

Im PHP Backend wird seit diesem Projekt Doctrine als Objektrelationaler Mapper (ORM) verwendet. Ein ORM hat den Vorteil, dass man mit einer Konfiguration Tabellen mit Objekten verknüpfen kann und somit viel Arbeit im Bezug auf die Datenbankkommunikation wegfällt. Ein Nachteil bei einem ORM kann sein, dass man Funktionen, welche Datenbank spezifisch sind, nicht verwenden kann oder diese nur mit grossem Aufwand. Doctrine bietet auch Locking-Mechanismen, welche in diesem Projekt benötigt werden. Es gibt die Möglichkeit die Entity-Konfiguration in ein File auszulagern oder mit Annotationen direkt im Objekt zu erfassen, zweiteres ist in diesem Projekt der Fall.

4.2.1 Locking

Das Problem bei einer Fahrgemeinschaftsverwaltung ist, dass man eine beschränkte Anzahl an Plätzen hat. Wenn man jetzt davon ausgeht, dass es noch drei freie Plätze hat und vier Mitglieder gleichzeitig auf den Anmelde-Knopf klicken, wäre die Fahrgemeinschaft plötzlich überbucht. Das Szenario ist etwas unwahrscheinlich, jedoch werden die Fahrgemeinschaften immer sehr Zeitnah an der Veranstaltung erstellt und somit ist die Zeitspanne, in der sich Mitglieder anmelden, nicht sehr gross. Eine Überbuchte Fahrgemeinschaft führt zu Ärger, welcher durch ein gezieltes Locking einfach vermieden werden kann.

Es gibt zwei verschiedene Varianten von Locking, das optimistische Locking und das pessimistische Locking. Das optimistische Locking funktioniert so, dass eine Zeitstempel oder eine Versionsnummer (Fussnote: besser, da der Zeitstempel bei sehr schnellen und vielen Transaktionen und je nach Auflösungsgenauigkeit fehler verursachen könnte) abgespeichert wird, wenn man das Objekt lädt. Wenn man nun etwas verändern möchte, wird diese Versionsnummer mitgegeben und geprüft, ob sich die Version seit diesem Zeitpunkt bereits verändert hat. Das pessimistische Locking wird mittels einem Lock, sei dass einem zeilenbasierten oder tabellenbasierten Lock, und einer Transaktion umgeben.

Die Implementation in diesem Projekt sieht für den User wie ein optimistisches Locking aus, alle vier Mitglieder, aus dem Beispiel oben, sehen den Anmelde-Knopf und einer erhält am Schluss die Fehlermeldung, dass es keinen Platz mehr hat. Im Backend ist es jedoch mit einem pessimistischen Locking auf Zeilenebene gelöst. Die Transaktion wird eröffnet, die Zeile wird gelockt, der Request wird überprüft, der Eintrag wird gespeichert und danach wird die Zeile mit dem Beenden der Transaktion wieder freigegeben. Um diesen Vorgang etwas besser zu veranschaulichen wurde ein Sequenzdiagramm (siehe Abbildung 4.2) erstellt, dies wurde anhand der Vorlage in ^[GKRS13].

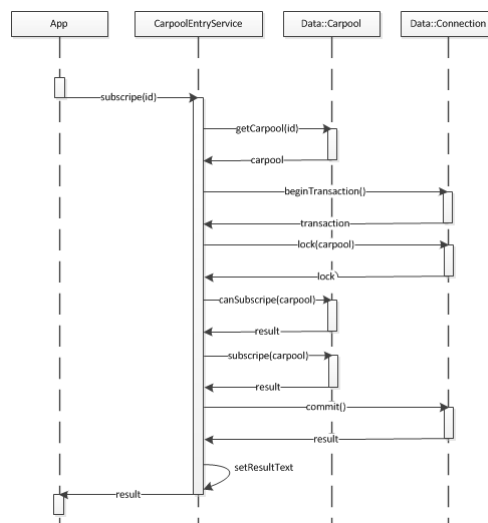


Abbildung 4.2: Sequenzdiagramm - Locking Verfahren

4.2.2 API

Das App enthält so gut wie keine Daten, sondern holt diese immer von dem Backend. Dazu wurde ein RESTful API erstellt, welches JavaScript Object Notation-Objekte (JSON-Objekte) zurück liefert. Das API funktioniert nach dem De-facto-Standard (siehe^[wik14e]). Bei einem Listen-Aufruf ohne Einträge liefert es eine leere Liste zurück, wenn ein spezifisches Objekt abgerufen wird, liefert es null zurück. Bei Aktionen wird ein JSON-Objekt mit den Attributen 'success' und 'error_message' zurück gegeben.

Das API hat eine Versionsnummer in der URL, damit kann sichergestellt werden, dass ältere Versionen von der App immer noch lauffähig sind, wenn etwas an der Schnittstelle geändert wird. Dieser Punkt ist sehr wichtig, da man den Usern nicht vorschreiben kann, wann und ob sie ihr App updaten.

4.3 Lösungsvarianten

Es gibt verschiedene wege ein App zu entwickeln und jede hat seine Vor- und Nachteile. In diesem Abschnitt werden drei verschiedenen Methoden miteinander verglichen und die beste für diesen Anwendungszweck ausgewählt.

4.3.1 Native App

Früher gab es nur diese Variante der App Entwicklung, man musste für jede Plattform die bereitgestellte Integrated Development Environment (IDE) verwenden und die dazugehörige Sprache lernen. Bei Apple ist das XCode und Objective-C, bei Android früher Eclipse Android Development Tools (ADT) und heute Android Studio basierend auf IntelliJ beide Male mit der Programmiersprache Java. Die Vorteile sind voller Funktionsumfang und gute Unterstützung bei der Entwicklung durch die IDE's. Das führen von zwei oder mehreren separaten Code-Sourcen ist jedoch sehr aufwendig und mühsam.

4.3.2 Xamarin

Xamarin (siehe^[xam14]) ist ein sehr mächtiges Framework, welches es ermöglicht den Code der App in C# zu schreiben und dies dann in Native Code umzuwandeln. Xamarin liefert seine eigene IDE kann aber auch in Visual Studio integriert werden Grosse Firmen wie 3M, AT&T und HP verwenden Xamarin für ihre Apps. Der grosse Vorteil ist, dass man fast alle Funktionen einer Native App verwenden kann und dazu nur eine Sprache und eine IDE beherrschen muss.

4.3.3 Phonegap

Phonegap (siehe^[pho14]) basiert auf Cordova (siehe^[cor14]), ein Framework von Apache, mit welchem es möglich ist eine Webseite geschrieben in HTML und Javascript in eine Native App umzuwandeln. Phonegap bietet Javascript Plugins um die Native Funktionen wie zum Beispiel Geolocation, Kompass oder Push-Nachrichten zu benutzen. Das Framework erstellt für die gewünschte Plattform ein App mit einer Webview und fügt die nötigen Klassen für die Plugins hinzu. Ein positiver Nebeneffekt ist bei dieser Methode, dass man die Webseite auch online stellen kann und alles ausser die Plugins benutzt werden kann. Phonegap bietet im Gegensatz zu den anderen Varianten keinen Möglichkeit grafischen Benutzeroberflächen zu modellieren oder ähnliches, es kümmert sich lediglich um die Konvertierung in die verschiedenen Plattformen und stellt die Schnittstellen zu den Native Funktionen bereit. Phonegap wird oft mit einem für mobile Geräte optimiertem Framework wie zum Beispiel jQuery Mobile (siehe^[jqu14]) kombiniert.

4.3.4 Nutzwertanalyse

Bewertungskriterien

In der Nutzwertanalyse werden folgende Punkte betrachtet und nach dem angegebenen Schema bewertet und dann gewichtet. Die Kriterien sind grösstenteils aus den Softwarequalitätsmerkmalen nach ^[iso] abgeleitet.

Aufwand

- **Beschreibung:** Wie gross ist der geschätzte Aufwand mit dieser Methode?
- **Bewertung:** 1: sehr hoch, 10: sehr niedrig
- **Gewichtung:** 5 (Die Zeit für dieses Projekt ist beschränkt und die Entscheidung könnte zu einem Risiko werden)

Benutzbarkeit

- **Beschreibung:** Wie schnell hat man diese Variante erlernt?
- **Bewertung:** 1: sehr langsam, 10: sehr schnell
- **Gewichtung:** 4 (Umso mehr Zeit für das Erlernen investiert wird, umso weniger Zeit hat man für die Implementierung)

Übertragbarkeit

- **Beschreibung:** Wie flexibel ist diese Variante? Kann sie auf verschiedenen Plattformen laufen?
- **Bewertung:** 1: sehr spezifisch, nicht portabel, 10: sehr flexibel
- **Gewichtung:** 4 (Die Anforderungen geben vor, dass das App mindestens auf Android und iOS laufen muss)

Funktionalität

- **Beschreibung:** Wie gross ist der Funktionalitätsumfang?
- **Bewertung:** 1: sehr eingeschränkt, 10: sehr weit reichend
- **Gewichtung:** 3 (In der ersten Version der App wird noch nicht viel Funktionalität gebraucht)

Kosten

- **Beschreibung:** Wie viel kostet diese Lösung?
- **Bewertung:** 1: sehr teuer, 10: gratis
- **Gewichtung:** 1 (Der Turnverein kommt für die Kosten auf und ist auch bereit etwas dafür zu bezahlen)

Bewertung

Anhand der zuvor definierten Kriterien wurde eine Bewertung vorgenommen. Diese Bewertung ist nicht generell gültig, sie bezieht sich auf die Erfahrung des Entwicklers dieser Arbeit und das Projekt selbst.


Kriterium	Native App 	Xamarin 	Phonegap 
Aufwand	2 (10)	4 (20)	7 (35)
Begründung	Schon beim Support von 2 Plattformen enorm gross	C# lernen, in Xamarin einarbeiten	jQuery Mobile erlernen, Phonegap kennen lernen
Benutzbarkeit	5 (20)	4 (16)	7 (28)
Begründung	Die IDEs sind sehr umgänglich, jedoch nicht sehr viel Erfahrung in Objectiv-C	Keine Erfahrung in C# und fast keine Erfahrung mit Visual Studio/Xamarin	Viel Erfahrung in HTML/Javascript, jedoch keine Erfahrung mit jQuery Mobile
Übertragbarkeit	2 (8)	7 (28)	8 (32)
Begründung	Jede Plattform seinen eigenen Code	Einschränkung bei der Wiederverwendung von UI Code	Ausser einigen Zeilen exakt der gleiche Code
Funktionalität	10 (30)	9 (27)	7 (21)
Begründung	Alles was die Produkte bietet, kann verwendet werden	Nahezu alles kann verwendet und gemacht werden	Plugins bieten nicht den kompletten Funktionsumfang
Kosten	9 (9)	3 (3)	9 (9)
Begründung	Developer Accounts kosten, ansonsten gratis	Developer Accounts kosten, jedoch Kosten pro Plattform	Developer Accounts kosten, ansonsten gratis
Total (gewichtet)	28 (77)	27 (94)	38 (125)

Tabelle 4.1: Nutzwertanalyse - App Variante

Fazit

Das Resultat der Nutzwertanalyse ist eindeutig, die Entwicklung mittels Phonegap ist für dieses Projekt mit Abstand die beste Lösung. Die Methode hat noch weitere Vorteile, durch dass das der Code nur in eine App verpackt wird und er nicht in eine andere Sprache übersetzt wird, gibt es viel weniger Overhead. Zudem ist es möglich den Source auch in der App ohne Problem anzupassen, dass heisst, falls der Convert nicht mehr funktioniert, weil er zum Beispiel nicht mehr unterhalten wird, kann man trotzdem noch weiterentwickeln und muss nicht zu erst ein Reverse Engineering machen.

4.4 Mobile App

Die Nuzwertanalyse 4.3.4 hat gezeigt, dass die beste Methode für dieses Projekt das Phonegap Framework ist. Dies wurde im Projekt auch so umgesetzt und mit jQuery Mobile kombiniert. jQuery Mobile hat die Grundidee alle Seiten in dem selben HTML-File zu speichern, bietet jedoch auch die Möglichkeit dies in verschiedene Files aufzuteilen. In diesem Projekt wurde die zweite Variante gewählt, weil damit zwar ein wenig Code-Duplikation generiert wurde, aber jede Seite einzeln getestet werden konnte, nur die nötigen Elemente geladen werden müssen und vorallem damit die HTML-Files übersichtlich bleiben. Die einzelnen Seiten sind sehr einfach aufgebaut, sie bestehen aus einem Header, in welchem das Menü, bzw. in Unterseiten die Back-Taste, und die Einstellungen aufgerufen werden können. Der Inhalt wird in einer ListView angezeigt. Die Daten werden über einen jeweiligen Asynchronous JavaScript and XML-Request (Ajax-Request) vom Backen abgeholt und dann in die ListView abgefüllt.

Von Phonegap wurden die Plugins PushNotification, für das Empfangen von Push-Nachrichten, und Device verwendet. Das PushNotification Plugin muss für Android und iOS spezifisch initialisiert werden und um herauszufinden, auf welcher Plattform das App läuft, wurde das Device Plugin verwendet.

5 Umsetzung des Prototypes

In diesem Kapitel wird kurz auf die Entwicklungsumgebung, welche für dieses Projekt verwendet wurde, eingegangen und danach wird erklärt, wie die Umsetzung des Mobile App Prototypes und des Backends durchgeführt wurde.

5.1 Entwicklungsumgebung

Ein Softwareprojekt benötigt immer eine gewisse Umgebung, welche die erforderlichen Funktionen erfüllt. Die Entwicklung eines Apps mit Hilfe von Phonegap kommt mit sehr wenig aus und die Entwicklungsumgebung ist sehr schnell aufgebaut.

5.1.1 IDE - Integrated Development Environment

Eine IDE, um welche man sicher nicht herum kommt, ist XCode, sie wird benötigt um das App zu paketieren und direkt zu der Analyse von Apple hochzuladen. Zusätzlich wurde noch Aptana Studio, welches für Webentwicklung ausgelegt ist, verwendet. Hier konnte man aber auf beliebige Alternativen umsteigen und notfalls auch mit einem gewöhnlichen Texteditor arbeiten. Die Vorteile einer IDE sind Syntax-Highlighting, -Checking und Autocompletion.

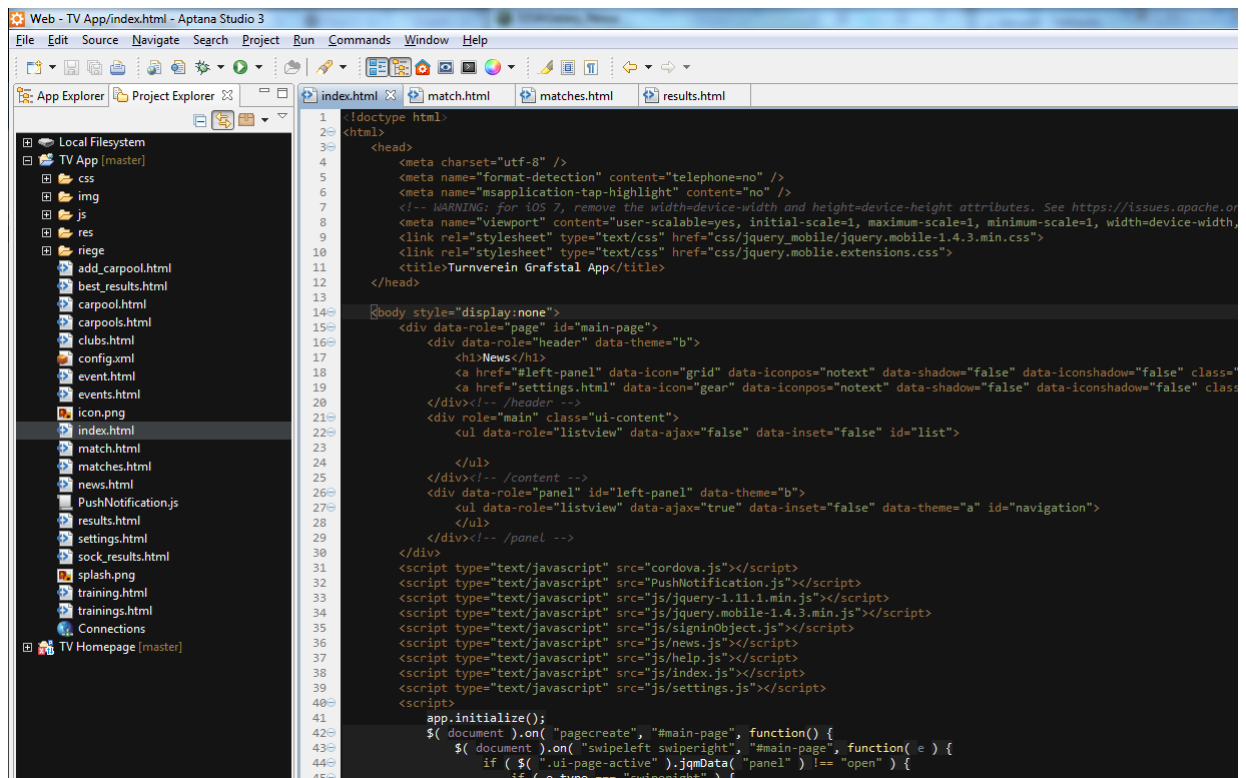


Abbildung 5.1: Aptana Studio

5.1.2 Versionierung

Versionierung ist in der Softwareentwicklung ein sehr wichtiges Thema, früher war das ein manueller Task, heute gibt es verschiedenste Tools, welche einem dabei unterstützen. In diesem Projekt wurde git (ref) verwendet, was eines der verbreitetsten Versionierungstools ist. Das Remote Repository wurde auf Github (ref) erstellt. Es wurde geschaut, dass der Code jeden Abend auf das Repository geladen wurde, damit man auch gleich ein Backup hat.

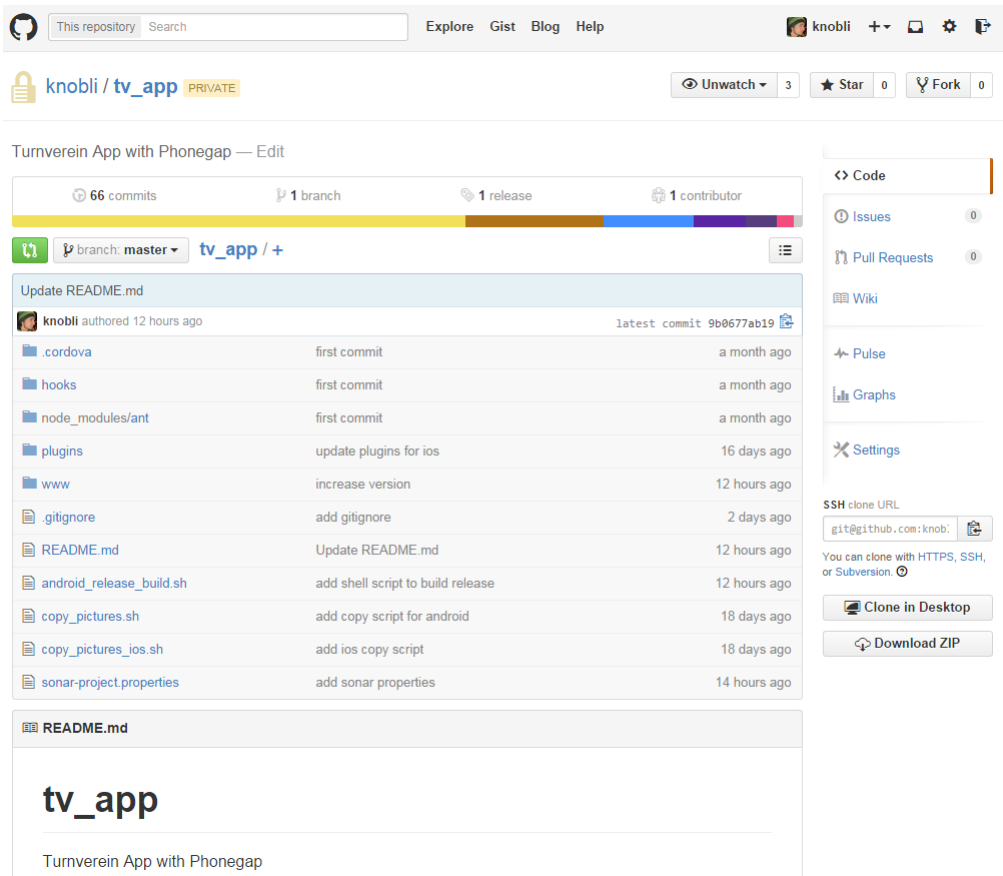


Abbildung 5.2: Github Repository

5.1.3 SDK - Software Development Kit

Für die Entwicklung der App auf den beiden Plattformen Android und iOS wurden die dazugehörigen SDKs gebraucht, um die Applikation in einem Emulator laufen zu lassen. Vorallem bei Android, mit seinen diversen Gerätevariationen, macht das Sinn, weil man beim Emulator jedes beliebige Device emulieren kann.

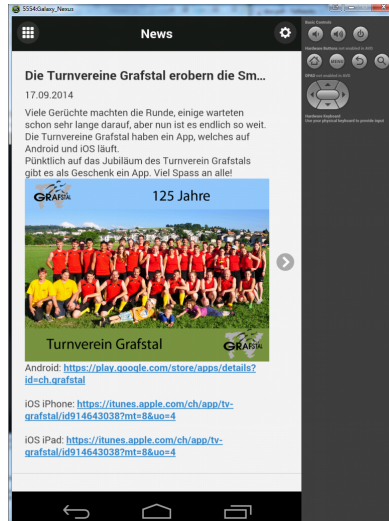


Abbildung 5.3: Android Emulator

5.1.4 Webbrowser

Die App wurde nicht nur auf Emulatoren getestet, sondern auch in Browsern. Dazu wurde auf Windows mit dem Google Chrome und auf Mac OSX mit Safari gearbeitet.

5.1.5 Technische Geräte

Push-Nachrichten können auf Emulatoren, wie auch in Webbrowsern nicht getestet werden, dazu benötigt man ein physisches Gerät. Die App wurde auf einem iPhone 4S, iPhone 5S, iPad und Samsung Ace getestet. Für die Entwicklung wurde ein Windows PC verwendet und damit XCode benutzt werden konnte, wurde zusätzlich noch auf einem iMac gearbeitet.

5.1.6 Testen - Analysieren

Neben den Tests an Geräten, Emulatoren wurden auch noch automatisierte Tests und Analysen durchgeführt. Für die automatisierten Tests im Backend wurde PHPUnit (siehe ^[php14]) verwendet und für die statische Code Analyse wurde Sonar (siehe ^[son14]) mit dem PHP und Web Plugin (siehe Abbildung 5.4 und 5.5) aufgesetzt und verwendet. Damit das API direkt getestet werden konnte, wurde das Chrome App Advanced REST client (siehe Abbildung 5.6) verwendet.

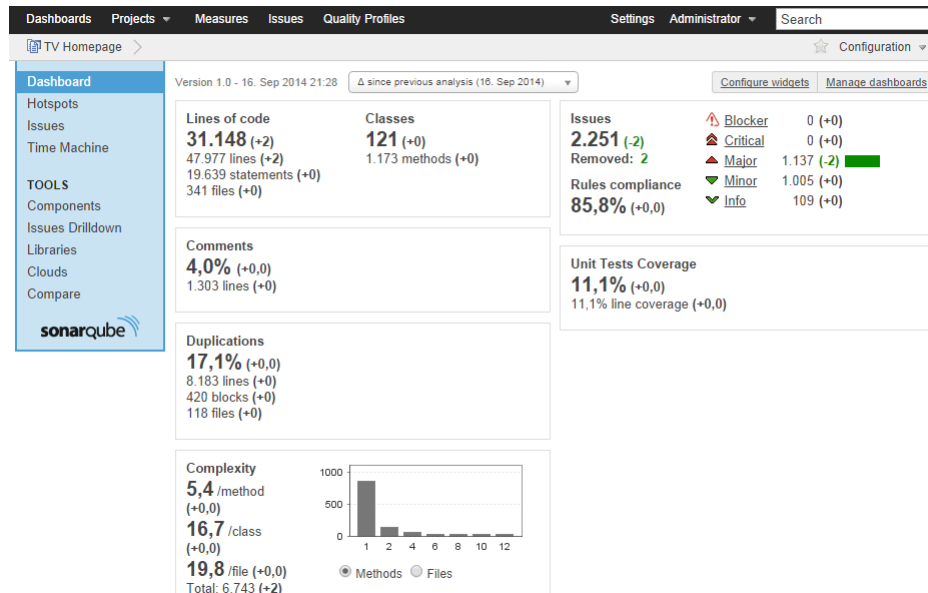


Abbildung 5.4: Sonar - statische Code Analyse Backend

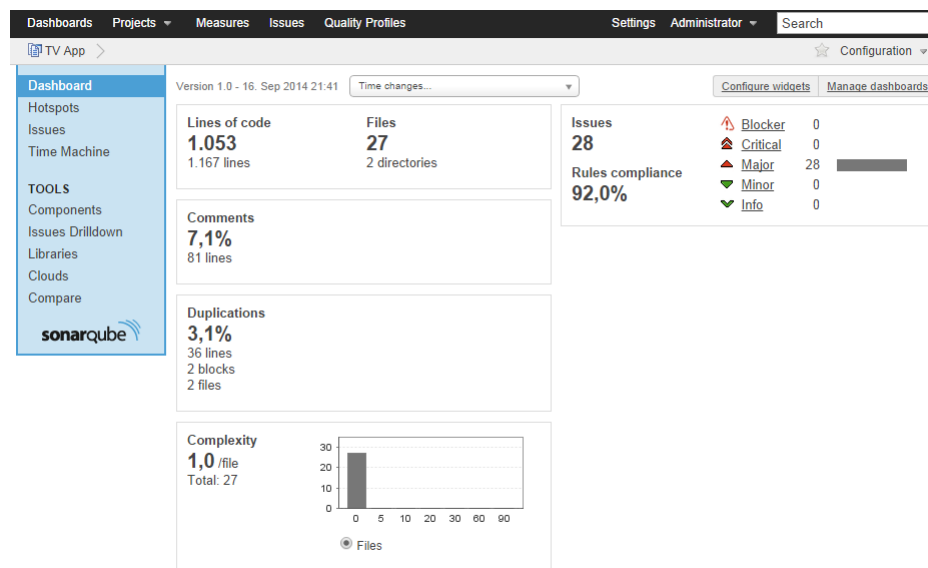


Abbildung 5.5: Sonar - statische Code Analyse App

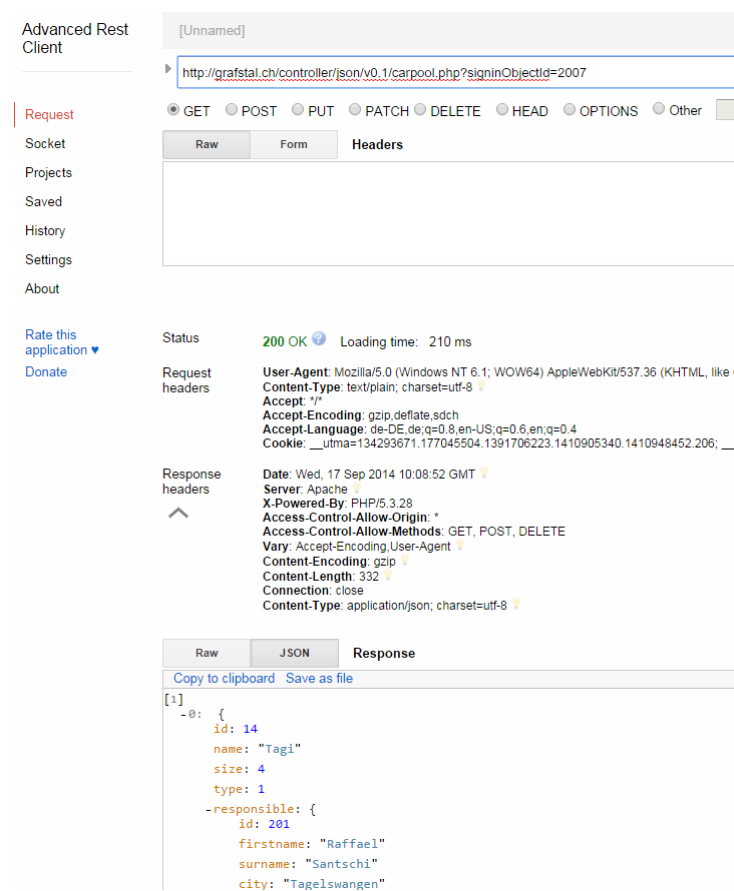


Abbildung 5.6: Advanced Rest Client

5.2 Mobile App

In diesem Unterkapitel werden neben einem detaillierten Aufbau der App, die unterschiedlichen Provisioning Prozesse und das Refacotring des Backends beschrieben. Desweiteren wird eine Übersicht der Push-Nachricht Implementation vermittelt.

5.2.1 Aufbau

Das Menü (siehe Abbildung 5.7) der App wurde sehr schlicht gehalten und sehr leicht erweiterbar gemacht. Da das Menü in verschiedenen Seiten gebraucht wird, werden die Elemente zentral verwaltet. Die Einträge News, Vereine und Veranstaltungen sind für alle Benutzer sichtbar, wenn auch mit anderem Inhalt und Design. Der Link zu den Resultaten ist nur für angemeldete Benutzer verfügbar.

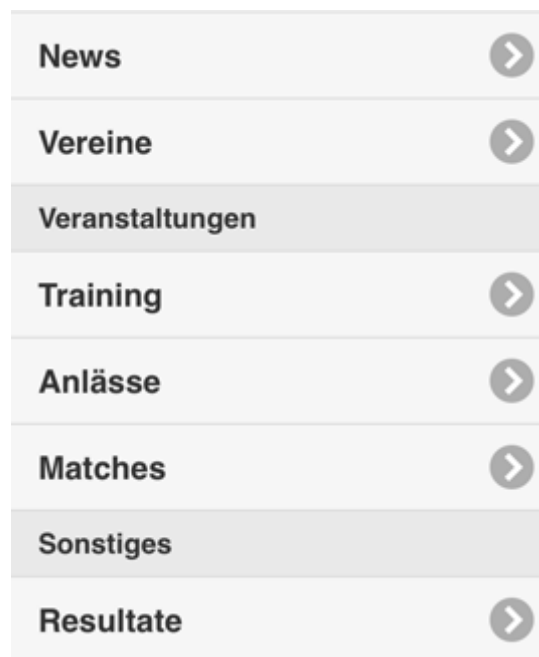


Abbildung 5.7: Navigation im App

News

Die News Seite (siehe Abbildung 5.8) ist zu gleich auch die Startseite, wenn das App aufgemacht wird. Auf ihr werden die neuesten drei Berichte angezeigt. Das System ist wie auf der Homepage, die Elemente enthalten nur einen Einleitungstext und erst wenn man auf das Element klickt kann man den ganzen Bericht lesen. Dies hat den einfachen Grund, dass die Berichte zum Teil sehr lange sind und dann die anderen Berichte verloren gehen. Bei angemeldeten Mitgliedern wird zusätzlich noch die nächste Veranstaltung angezeigt, damit man sich sofort an- bzw. abmelden kann und wichtige Informationen nachschlagen kann.

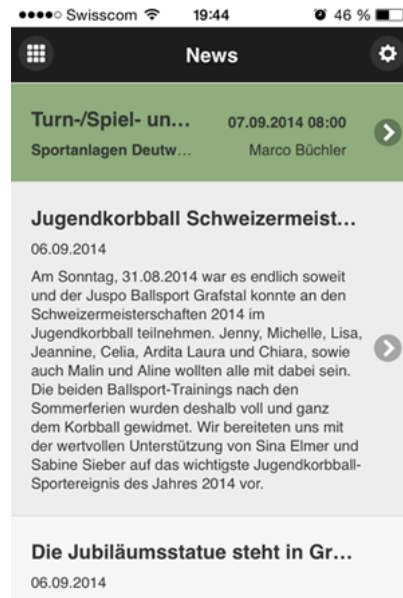


Abbildung 5.8: News Seite

Vereine

Die Vereine Seite (siehe Abbildung 5.9) listet alle Vereine und Riegen der Turner Familie Grafstal auf. Wen man auf eine Riege klickt, erhält man zusätzliche Informationen, wie zum Beispiel die Trainingszeiten. Diese Seite enthält vorallem Informationen für die Öffentlichkeit oder ist ein Nachschlagewerk für Mitglieder.

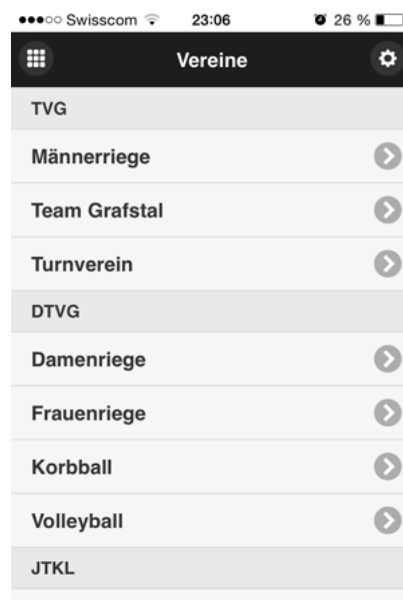


Abbildung 5.9: Vereinsseite

Veranstaltungen

Die drei Veranstaltungsseiten sind alle gleich aufgebaut (siehe Abbildung 5.10(b)), sie zeigen die aktuellen Elemente mit Name, Datum, Zeit, Ort und Verantwortlichen. Falls der Benutzer angemeldet ist, wird durch Farbe angezeigt, ob er sich angemeldet (grün), abgemeldet (rot) oder nichts von beidem (grau) hat.

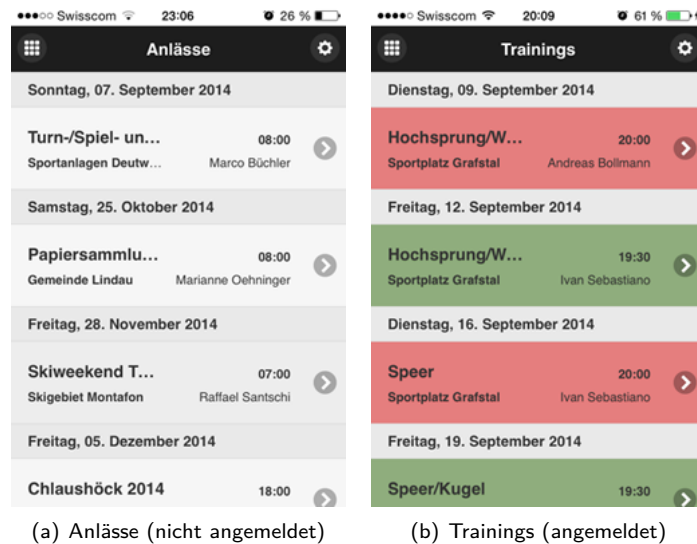


Abbildung 5.10: Veranstaltungen

Wenn auf eine Veranstaltung geklickt wird, geht eine Detailansicht (siehe Abbildung 5.11) auf. In dieser Ansicht erhält man weitere Information, kann sich an- und abmelden und kommt auch zu den Fahrgemeinschaften. Die Information, wer sich angemeldet hat, sehen nur angemeldete Benutzer.



Abbildung 5.11: Anlass Details

Fahrgemeinschaften

Bei jeder Veranstaltung können Fahrgemeinschaften erstellt werden, welche dann in der Übersicht (siehe Abbildung 5.12(a)) angezeigt werden. Man sieht den Namen, den Fahrer und den Wohnort vom Fahrer, zusätzlich sieht man noch, wie viele Plätze es noch frei hat oder dass man bereits angemeldet ist. In der Detailansicht (siehe Abbildung 5.12(b)) kann man sich anmelden und erhält eine Liste der Mitfahrenden.

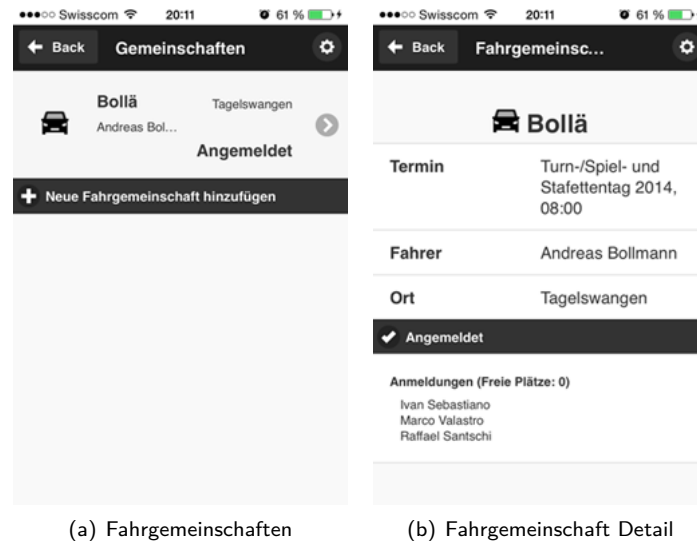


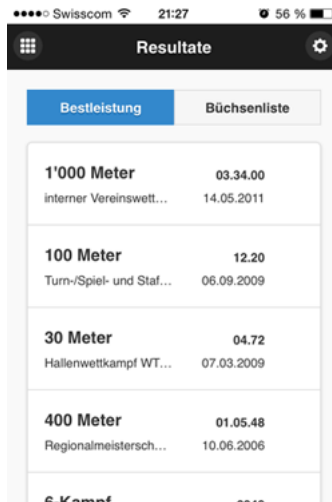
Abbildung 5.12: Veranstaltungen

Im Formular (siehe Abbildung 5.13) für die Erstellung einer neuen Fahrgemeinschaft muss man einen Namen angeben und kann den Typ der Gemeinschaft wählen, nur wenn es eine 'Car'-Fahrgemeinschaft ist, muss man die freien Plätze angeben.

Abbildung 5.13: Fahrgemeinschaft erstellen

Resultate

Unter Resultate (siehe Abbildung 5.14) finden angemeldete Benutzer ihre Bestleistungen und die letzten Resultate.

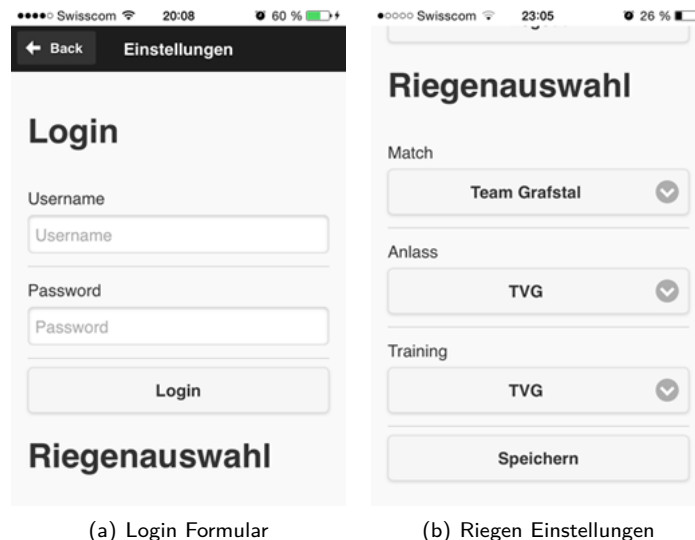


Bestleistung		Büchsenliste	
1'000 Meter	03.34.00		
interner Vereinswett...	14.05.2011		
100 Meter	12.20		
Turn-/Spiel- und Staf...	06.09.2009		
30 Meter	04.72		
Hallenwettkampf WT...	07.03.2009		
400 Meter	01.05.48		
Regionalmeistersch...	10.06.2006		
6. Kampf			

Abbildung 5.14: Resultate Seite

Einstellungen

In den Einstellungen kann sich der Benutzer nicht nur einloggen (siehe Abbildung 5.15(a)), sondern auch wählen, für welche Riege er die Matches, Anlässe und Trainings sehen möchte (siehe Abbildung 5.15(b)). Diese Filter werden dann auf die Veranstaltungsübersichtseiten angewendet.



(a) Login Formular

(b) Riegen Einstellungen

Abbildung 5.15: Einstellungen

5.2.2 Javascript Funktionen

Die Javascript Funktionen wurden möglichst generisch gewählt, damit man sie an verschiedenen Orten wiederverwenden kann. Mit diesem Ansatz ist man bei den verschiedenen Veranstaltungsseiten mehr wenig Code ausgekommen und es war einfach die nächste bevorstehende Veranstaltung auf der Startseite gleich anzuzeigen, wie in den anderen Seiten. Die Funktionen wurden in unterschiedliche Files geschrieben, welche nach ihrem Zweck benannt wurden.

- carpool.js: an- und abmelden bei Fahrgemeinschaften, anzeigen von Fahrgemeinschaften
- help.js: Datum formatieren, Listen-Elemente erstellen, Navigationselemente
- index.js: initialisieren der Push-Benachrichtigungen
- news.js: Berichte laden
- result.js: Resultate anzeigen
- settings.js: Dropdowns initialisieren, an- und abmelden, Riegefilter setzen
- signinObject.js: Veranstaltungen laden, Veranstaltungsdetails anzeigen, an- und abmelden bei Veranstaltungen

5.2.3 Login

Da das Backend nur über eine HTTP-Verbindung verfügte und die Anmeldedaten nicht im localStorage gespeichert werden sollten, wurde darauf verzichtet die Anmeldedaten bei jeder Anfrage an das Backend mitzusenden. Bei der Anmeldung wird die Mitglieder-ID gespeichert und dann bei einer Anfrage mitgeteilt. In einem späteren Projekt sollte diese Lösung durch ein Session-ID abgelöst werden.

5.2.4 Provisioning Prozesse

Die Entwicklung des Apps ist das eine, die Bereitstellung zum Download das andere. Phonegap hilft beim Provisioning vor allem bei Android, hier erstellt es, richtig konfiguriert, ein signiertes APK-Paket, welches in der Google Play Developer Console hochgeladen werden kann. Zuerst muss man sich jedoch einen Developer Account erstellen, danach kann man die Beschreibung und die Screenshots für das App hochladen. Das ganze dauert etwa 10 Minuten und 1 Stunde nach dem Absenden hat man sein App in Google Play. Es empfiehlt sich die App noch mit einer Absender-ID zu verknüpfen, damit man auch Statistiken über die Downloadzahlen sieht. (siehe dazu auch ^[and14a])

Bei Apple sieht das etwas anders aus, Phonegap kann kein Paket erstellen, es erstellt lediglich das Projekt. Diese Projekt muss man dann in XCode öffnen und von dort aus das Paket erstellen. Doch zuerst muss man einen Developer Account haben, diesen besitzt man wahrscheinlich bereits, da man sonst nicht auf realen Geräten testen kann, zusätzlich muss man ein Distribution Zertifikat erstellen und in XCode hinzufügen. Danach muss man in iTunes Connect neben den gleichen Informationen wie bei Android, noch einige Angaben zu den Inhalten des Apps machen. Desweiteren muss man Review Informationen bereitstellen, das heißt einen Test Benutzer und Informationen zum Entwicklung angeben. Erst wenn das App erfasst wurde, kann man in XCode ein Archiv erstellen und dieses direkt zu iTunes Connect hochladen. Nun werden schon die ersten Tests durchgeführt, ob zum Beispiel die Versionsnummern übereinstimmen und ob für jedes unterstützte Gerät Screenshots erfasst wurden. Falls dies nicht der Fall ist, wird das App abgelehnt und man muss die Dinge beheben. An diesem Punkt beginnt das warten, die durchschnittliche Review-Zeit beträgt 8 Tage. (siehe dazu auch ^[app14a] und ^[app14c])

5.3 Backend

5.3.1 Refactoring

In diesem Unterkapitel wird kurz der vorgefundene Ist-Zustand beschrieben und danach die Schritte, welche unternommen wurden, um das Backend wartbarer und flexibler zu machen.

Ist-Zustand

Am Anfang dieses Projekts war in jeder Klasse HTML- mit PHP-Code verschmischt und SQL-Abfragen waren überall verstreut. Dies machte es schwierig den Code zu warten. Zusätzlich wurde jede save-, update-, load- und delete-Anweisung von Objekten, wenn es dann solche gab, selber implementiert, was sehr mühesam und unflexibel war. Zudem erweiterten ähnliche Objekte zwar eine Super-Klasse, teilten sich aber keine Tabelle und hatten somit auch verschiedene ID-Sequenzen.

Schnell wurde klar, dass man nicht mit diesem Stand ein API aufbauen sollte, welches dann von dem App angesprochen wird. Um den Code besser zu strukturieren und übersichtlicher zu machen, wurde Doctrine eingesetzt. Doctrine ist, wie schon im Kapitel 4.2 erklärt, ein ORM, welches richtig eingesetzt viel Arbeit abnehmen kann. Damit man Doctrine jedoch sinnvoll verwenden konnte, benötigte es zuerst einige Zeit für die Analyse und die Umstellungen.

Refactoring

Zuerst wurde geschaut, was zusammengefasst werden konnte. Alle Entities, bei welchen man sich anmelden kann, wurden ganz getreu dem 'Don't repeat yourself' Prinzip als SigninObject zusammengefasst, da sie sehr viele gleiche Attribute und Funktionen haben. Im gleichen Zug wurde die Datenbank normalisiert, da zum Teil gleiche Ortsnamen in verschiedenen Tabellen vorhanden waren. Da die einzelnen Typen jedoch auch spezifische Attribute haben, wurde eine Joined-Inheritance (siehe ^[inh14b] und ^[inh14a]) angewendet, welche von vielen ORMs unterstützt wird. Die Joined-Inheritance basiert auf einer Grundtabelle, welche alle gemeinsamen Attribute beinhaltet und einer spezifischen Tabelle, welche die anderen Attribute beinhaltet. Die Tabellen werden über eine DiscriminatorColumn miteinander verknüpft. In diesem Konstrukt wurde auch ein Strategy-Pattern (siehe ^[GHJV09]) verwendet, um verschiedene Informationen im Kalender¹ anzuzeigen.

Dieser Umbau hat den grossen Vorteil, dass für jeden Veranstaltungstyp, sei das ein Training, Match, Sitzung, Anlass oder Helfereinsatz, die gleiche Anmeldefunktion verwendet werden kann. Der Umbau machte sich auch bei der Entwicklung der Fahrgemeinschaftsverwaltung bezahlt, als diese nur ein mal implementiert werden musste.

Das Datenbankdiagramm (siehe Abbildung 5.16) zeigt einen kleinen Ausschnitt aus dem Datenbank Schema von früher. Jeder Veranstaltungstyp hatte seine eigene Ortstabelle und seine eigene Verknüpfungstabelle für die Anmeldungen. Wenn man sich nun vorstellt, dass dieses Konstrukt für die fünf verschiedenen Typen vorhanden war, kann man sich denken, wie viel Tabellen schon bereits nur für die Anmeldungen vorhanden waren. Nach dem Refactoring sah das Datenbankdiagramm (siehe Abbildung 5.17) etwas einfacher aus, die Orte waren nun in einer Tabelle, die Anmeldungen auch und alle gemeinsamen Attribute wurden in der TerminObjekte-Tabelle gespeichert.

¹Der Kalender (nicht in diesem Projekt entwickelt) kann über ein ics-File auf Geräten eingebunden werden.

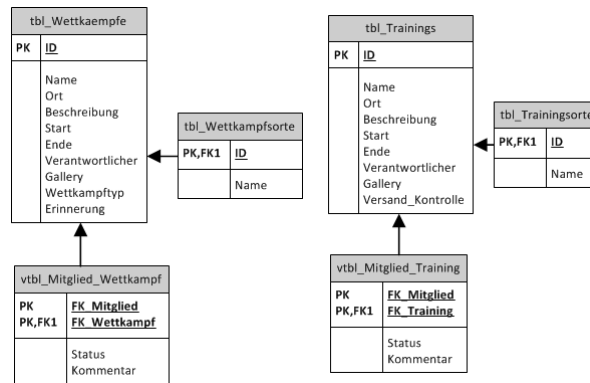


Abbildung 5.16: Datenbankdiagramm alt

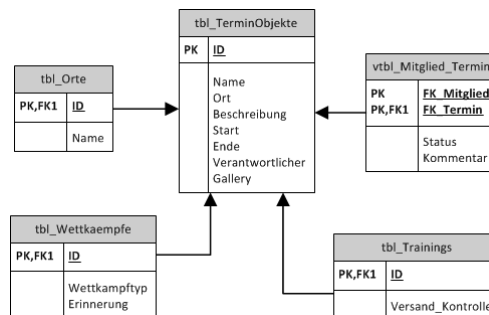


Abbildung 5.17: Datenbankdiagramm neu

Durch das Refactoring werden auch weniger SQL-Statments direkt in den Webseiten abgesetzt, es wird vermehrt² mit Objekte gearbeitet. Ein Repository liefert die gewünschten Objekte zur Webseite, die Abfrage Logik ist somit zentralisiert. Das Repository arbeitet mit einer Object Query Language (OQL), bei welcher man nicht über die Spalten der Tabelle Werte abfragt, sondern über Attribute des Objekts. Eine Änderung des Spaltennamens hat somit keine grössere Auswirkung als eine kleine Änderung in der Entity.

Die Struktur sieht nach der Implementierung von Doctrine, wie folgt aus:

- Entity: Attribute, Getter und Setter
- Entity-Manager: persistiert, löscht und ladet Entities oder gibt das Repository der Entity zurück
- Repository: spezifische Abfrage-Logik
- Services³: Business-Logik

²Nur an Orten, wo die Objekte innerhalb dieses Projektes angefasst wurden

³Services werden nicht von Doctrine zur Verfügung gestellt, wurden aber wegen ... implementiert (ref)

Facts and Figures

Das Refactoring war zwar sehr Zeit aufwendig, hat jedoch bei einigen Implementationen die Komplexität stark reduziert, Zeitersparnisse eingebracht und wird auch in Zukunft sehr viel erleichtern.

- Testabdeckung: Das Testen stellte sich vorher als extrem schwierig heraus und ist nun mit den Kapselungen viel einfacher. Es konnte während des Projekts doch immerhin eine Testabdeckung von 11.1% erreicht werden. Dieses Resultat wird natürlich noch sehr verfälscht durch Seiten und Funktionen, welche nicht Teil dieses Projekts waren.
- Duplikate: Der Prozentsatz der duplizierten Linien konnte um fast 6% reduziert werden, was etwa 4'000 Lines of code bedeutet.
- Komplexität: Die Komplexität pro Klasse konnte von 40 auf 17 gesenkt werden.
- Rule compliance: Die Rule compliance konnte um knapp 7% gesteigert werden.
- Issues: Über 3000 Issues, darunter alle kritischen und Blocker Issues, konnten behoben werden.

5.3.2 Push-Nachrichten

Das Versenden von Push-Nachrichten funktioniert bei iOS und Android ähnlich, jedoch sind die Vorbedingungen sehr unterschiedlich. Bei Android wird ein API-Key gebraucht, diesen kann man in der Google Developers Console erstellen. Dazu muss man ein Projekt erstellen und das 'Google Cloud Messaging for Android' aktivieren. Nun kann man einen neuen API-Key erstellen, welcher dann im Backend benötigt wird. Die Projektnummer ist zugleich die Sender ID und wird bei der Registrierung des produktiven Apps über Phonegap benötigt. (siehe dazu ^[and14b] und ^[dev14])

Bei iOS ist das etwas anders, als erstes braucht man ein Zertifikat, da nur ein Server mit diesem Zertifikat Push-Nachrichten an den APNS senden darf. Es gibt ein Zertifikat für den Entwicklungsserver und eines für den produktiven Server, über den Entwicklungsserver können nur verknüpfte⁴ Geräte im Entwicklungsmodus erreicht werden. Zusätzlich muss für das Testen der Push-Nachrichten noch ein iOS App Development Provisioning Profile erstellt werden, welches dann auch mit dem Developer Account und den Test Geräten verknüpft wird. Bei der Erstellung der Zertifikate ist es wichtig, dass die App ID mit dem Projekt übereinstimmt, ansonsten kommen die Push-Nachrichten nicht an. Die Registrierung des produktiven Apps über Phonegap ist ganz einfach und nicht Projekt spezifisch. (siehe dazu ^[ios14])

Nach der Registrierung bei dem jeweiligen Servicen schickt das App den Geräteschlüssel mit der Mitglieder ID ans Backend, welches den Schlüssel mit der Verknüpfung zum Mitglied persistiert. Nun hat man den Schlüssel, welchen man für das gezielt Versenden von Push-Nachrichten benötigt.

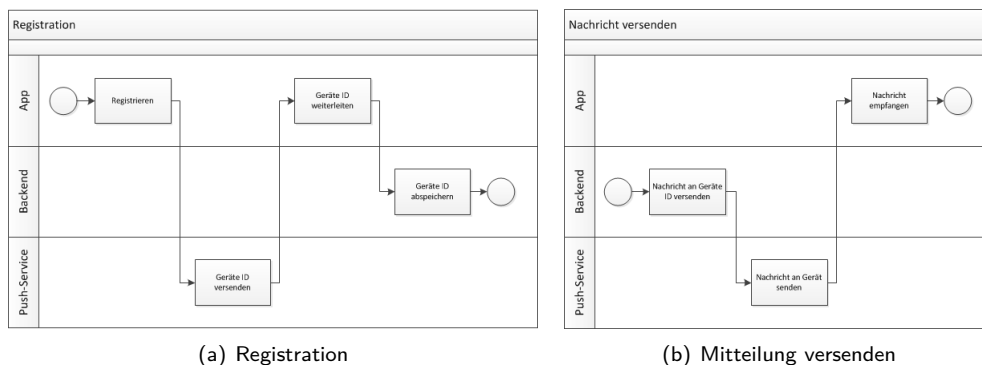


Abbildung 5.18: Ablauf von Push-Nachrichten

⁴um das App auf einem Gerät zu testen, muss man es zuerst mit dem Developer Account verknüpfen

6 Tests

In diesem Kapitel wird auf die verschiedenen Tests und Varianten, welche in diesem Projekt verwendet wurden, eingegangen.

6.1 Einführung

In diesem Projekt hat man die sieben Grundsätze aus^[SL12] als Leitlinie beim Testen genommen:

1. **Testen zeigt die Anwesenheit von Fehlern:** Testabdeckung mit Sonar ermittelt und Anforderungen überprüft
2. **Vollständiges Testen ist nicht möglich:** Es wurde solange getestet bis die Fehlerfindungsrate einen bestimmte Punkt unterschritten hatte
3. **Mit dem Testen frühzeitig beginnen:** Die Tests für die neuen Funktionen wurden zur gleiche Zeit, wie der Code, geschrieben
4. **Häufung von Fehlern:** Wenn Fehler auftraten, dann wurde diese Komponente nach der Behebung nochmals intensiv getestet
5. **Zunehmende Testresistenz (Pesticide paradox):** Als Änderungen am Code gemacht wurden, wurden die Tests erweitert bzw. angepasst
6. **Testen ist abhängig vom Umfeld:** Dieses System ist nicht so sicherheitskritische, wie eine Bank, jedoch werden die Dienste rege benutzt, somit sollte die Testintensität in einem normalen Rahmen liegen.
7. **Trugschluss: Keine Fehler bedeutet eine brauchbares System:** Die Stakeholder wurden schon bei der Mock-up Erstellung mit einbezogen und bekamen auch während der Entwicklung immer wieder einen Blick in den aktuellen Stand.

6.2 Testing

Komponententests und Integrationstests wurden im Backend mit PHPUnit durchgeführt. Bei den Integrationstests wurde eine In-Memory Datenbank verwendet, welche bei jedem Testlauf neugeladen wird, somit ist man komplett unabhängig von äusseren Einflüssen. Die Testabdeckung wurde mit Sonar überprüft, hier wurde vorallem ein Augenmerk auf die neuen Funktionen und in diesem Projekt verwendete Funktionen gelegt. Neben den verschiedenen Entitäten wurden die Repositories und die Services getestet. (siehe^[SL12])

6.2.1 Testphasen

Es wurde festgelegt, dass das Projekt nicht mit der Implementation fertig ist, sondern mit der definitiven Auslieferung. Damit der Kunde genug Erfahrungen sammeln kann, wurden Testphasen definiert und erst nach Beendigung dieser Testphasen, konnte das Produkt abgenommen werden. Die Testphasen wurden getrennt, damit Fehler klar abgegrenzt werden konnten und für die gemeldeten Fehler auch mehr Zeit zur Verfügung stand.

Backend

Die Testphase des Backends begann nachdem lokalen Testing des Refactorings und der produktiv Schaltung des Backends. Die Mitglieder wurden informiert, dass sie theoretisch keine grossen Änderungen feststellen sollten und sie sollten sich doch bitte bei negativen Abweichungen zur vorherigen Version sofort melden. Es wurden nur wenige Spezialfälle gemeldet, welche schnell behoben werden konnten.

App

Da das Freigeben der iOS App mehrere Tage benötigt, wurde die erste Testphase auf Android gemacht, damit man schneller reagieren konnte. Das App wurde hochgeladen und nur einigen Personen davon erzählt, nach kurzer Zeit gab es bereits über 10 Downloads. Das Feedback war gut und nur einige kleine Fehler wurden gefunden. Alle Fehler und ein paar kleine Verbesserungsvorschläge wurden sofort umgesetzt, getestet und dann eine neue Version hochgeladen. Grössere Anforderungen wurden erfasst und auf die kommenden Versionen verschoben. Als keine negativen Rückmeldungen mehr kamen und auch beim Nachfragen nichts mehr zum Vorschein kam, wurde die Testphase mit dem Einliefern des iOS Apps beendet.

6.3 System- und Abnahmetest

Nach erfolgreichem Testing und abschluss der Testphasen, wurde für den Abnahmetest ein Testprotokoll erstellt, welches vorgängig im Systemtest schon durchgegangen wurde. Das Testprotokoll wurde danach dem Kunden übergeben und von diesem selber nochmals abgearbeitet.

6.3.1 Testprotokoll

Das Testprotokoll basiert auf den Use Cases (siehe 3.2.1) und den Anforderungen (siehe 3.2.3). Akzeptanz Kriterien mit einen UND- oder ODER-Verknüpfung wurden in aufgesplittet, um sicher zu gehen, dass beide Bedingungen erfüllt sind.

ID	Test	Herkunft	erfüllt / nicht erfüllt
1a	Bei bekanntem Username und Password muss es dem Mitglied möglich sein, sich beim Backend anzumelden	Anforderung RF-F1	erfüllt
1b	und dann erweiterte Möglichkeiten und Informationen zu erhalten.	Anforderung RF-F1	erfüllt
2a	Falls ein unbekannter Username eingegeben wurden kommt eine entsprechende Fehlermeldung.	Anforderung RF-F1	erfüllt
2b	Falls ein falsche Passwort eingegeben wurden kommt eine entsprechende Fehlermeldung.	Anforderung RF-F1	erfüllt
3	Das angemeldete Mitglied kann eine Fahrgemeinschaft eröffnen.	Anforderung RF-F2	erfüllt
4a	Bei ungültigen Informationen kommt eine entsprechende Fehlermeldung	Anforderung RF-F2	nicht erfüllt
4b	Bei nicht ausreichenden Informationen kommt eine entsprechende Fehlermeldung	Anforderung RF-F2	erfüllt
5	Das angemeldete Mitglied kann die Fahrgemeinschaften anzeigen lassen	Anforderung RF-F3	nicht erfüllt
6	Das angemeldete Mitglied sieht einen Hinweis, falls es keine Fahrgemeinschaften gibt	Anforderung RF-F3	erfüllt

7	Das angemeldete Mitglied kann einer Fahrgemeinschaft beitreten	Anforderung RF-F4	erfüllt
8	Das angemeldete Mitglied kann einer Fahrgemeinschaft nicht erneut beitreten	Anforderung RF-F4	erfüllt
9	Das angemeldete Mitglied erhält eine Fehlermeldung falls die Fahrgemeinschaft keine Kapazität mehr aufweist	Anforderung RF-F4	plz test
10	Das angemeldete Mitglied kann sich für einen Anlass anmelden	Anforderung RF-F5	erfüllt
11	Das angemeldete Mitglied kann sich für einen Anlass nicht erneut anmelden	Anforderung RF-F5	plz test
12	Der Benutzer erhält Informationen über den ausgewählten Anlass	Anforderung RF-F6	erfüllt
13	Der Benutzer sieht einen Hinweis, falls keine Anlässe vorhanden sind	Anforderung RF-F6	erfüllt
14	Das angemeldete Mitglied sieht die erweiterten Informationen zum Anlass	Anforderung RF-F7	erfüllt
15	Nicht angemeldete Mitglieder sehen die erweiterten Informationen zum Anlass nicht	Anforderung RF-F7	erfüllt
16a	Der Benutzer erhält Informationen über die verschiedenen Vereine	Anforderung RF-F8	erfüllt
16b	Der Benutzer erhält Informationen über die verschiedenen Riegen	Anforderung RF-F8	erfüllt
17	Der Benutzer kann die aktuellen Berichte lesen	Anforderung RF-F9	erfüllt
18	Das angemeldete Mitglied erhält die Push-Nachrichten	Anforderung RF-F10	erfüllt
19a	Ein Nutzer mit einem Android Mobiltelefon kann die App öffnen	Qualitätsanforderung RF-NF1	erfüllt
19b	Ein Nutzer mit einem Android Mobiltelefon kann die App verwenden	Qualitätsanforderung RF-NF1	erfüllt
20a	Ein Nutzer mit einem iPhone kann die App öffnen	Qualitätsanforderung RF-NF2	erfüllt
20b	Ein Nutzer mit einem iPhone kann die App verwenden	Qualitätsanforderung RF-NF2	erfüllt

6.4 Abnahme Protokoll

Nach dem erfolgreichen Abnahmetest wurde vom Kunde das Abnahme Protokoll unterzeichnet.

Abnahmeprotokoll für Turnverein App

Raffael Santschi
Rietstrasse 5
8317 Tagelswangen

Marco Mathe
Präsident Turnverein Grafstal
Illnauerstrasse 13
8307 Effretikon

Hiermit bestätige ich im Namen des Turnvereins, dass die gewünschten Anforderungen zu unserer Zufriedenheit umgesetzt wurden. Wir haben folgende Funktionen angeschaut und getestet:

- Login Möglichkeit bei der Mobile App
- Fahrgemeinschaft erstellen
- Übersicht über Fahrgemeinschaft
- Fahrgemeinschaft beitreten
- Anmelden an Vereinsanlässe
- Informationen über Vereinsanlässe
- Erweiterte Informationen über Vereinsanlässe
- Informationen über Vereine und Riegen
- Aktuellen Berichte
- Push-Nachrichten versenden
- Unterstützung von Android
- Unterstützung von iOS

Folgende Bemerkungen haben wir noch:

Marco Mathe
Präsident
Turnverein Grafstal

Abbildung 6.1: Abnahmeprotokoll

7 Schlussfolgerungen

In der Schlussfölderung wird kurz auf die Verwendung des Produktes, das Fazit des Erfassers und den Ausblick eingegangen.

7.1 Verwendung

Das App fand schnell grossen Anklang, die Neuigkeit über das App verbreitete sich automatisch. Die Turnvereine machten die Öffentlichkeit zusätzlich über Facebook und ihre Homepage auf das App aufmerksam. Eine Woche nach der Bekanntmachung hatten bereits 50 Personen das App heruntergeladen. Die Fahrgemeinschaftsverwaltung wird benutzt und die Anmeldungen an Veranstaltungen werden vermehrt über das App gemacht.

7.2 Fazit

Die Entwicklung einer App hat sehr viel Spass gemacht und die Zeit verging wie im Flug. Ich fand es toll das angeignete Wissen über die Anforderungsanalyse und Architektur einsetzen zu können. Die klar definierten Anforderungen haben mir bei der Entwicklung geholfen und liessen nicht viel Spielraum für Missverständnisse. Die enge Zusammenarbeit mit den Mitgliedern war mir bei der Implementation ebenfalls eine grosse Hilfe.

Phoneyap ist ein gutes Framework, welches einem viel Arbeit abnimmt und in diesem Projekt die richtige Entscheidung war. Ein App mit Phoneyap zu entwickeln, lohnt sich bereits ab zwei Plattformen, da der Wartungsaufwand sonst enorm gross wird. Ich fand es auch toll, dass es auf HTML und Javascript basiert, was mir den Einstieg erleichtert. jQuery Mobile ist zwar mächtig, aber nur so lange man die Standardelemente und Funktionen verwendet, sonst kann es sehr schnell in viel Arbeit ausarten. Die Wahl jQuery zu verwenden, war eine gute und effiziente Lösung für diesen Prototyp, ich werde mir jedoch nochmals überlegen, ob ich vielleicht nicht auf ein flexibleres Framework wechseln möchte.

Das Refactoring war in diesem Projekt dringend nötig, hat jedoch auch viel mehr Zeit gekostet als geplant. Ich wünschte mir für spätere Projekte, dass ich bereits Tests vorfinde, was den Umbau einiges einfacher gemacht hätte. Es war spannend eine grössere Datenbankmigration auf produktiven Daten zu planen und dann auch durchzuführen. Mit der neuen Testumgebung und dem ORM machte das Entwickeln richtig Spass.

7.3 Ausblick

Ich werde in einem weiteren Schritt wahrscheinlich das App mit Angular JS erweitern. Es sind bereits viele neue Anforderungen von den Mitgliedern eingetroffen, welche in den kommenden Monaten umgesetzt werden. Das positiven Feedback der Mitglieder haben mich sehr gefreut und auch motiviert. Die App ist für den Turnverein, zu mindest für die jüngere Generation, ein voller Erfolg und so soll es auch sein. Zum Schluss kann ich nur noch sagen, es ist ein tolles Gefühl seine eigene App im App Store zu sehen!

8 Verzeichnisse

Literaturverzeichnis

- [and14a] *Apps hochladen - Android Developer-Hilfe*. <https://support.google.com/googleplay/android-developer/answer/113469?hl=de>. Version: September 2014
- [and14b] *Getting Started | Android Developers*. <http://developer.android.com/google/gcm/gs.html>. Version: September 2014
- [app14a] *App Distribution Guide: Submitting Your App*. <https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/SubmittingYourApp/SubmittingYourApp.html>. Version: September 2014
- [app14b] *Apple - Live - September 2014 Special Event*. <http://www.apple.com/live/2014-sept-event/>. Version: September 2014
- [app14c] *iOS: Veröffentlichung im App Store*. <http://www.ralfebert.de/ios/app-store-distribution/>. Version: September 2014
- [cor14] *Apache Cordova*. <http://cordova.apache.org/>. Version: Mai 2014
- [dev14] *Tutorial: Implement Push Notifications in your PhoneGap Application : Devgirl's Weblog*. <http://devgirl.org/2013/07/17/tutorial-implement-push-notifications-in-your-phonegap-application/>. Version: September 2014
- [FH14] FUCHS, Thomas ; HAMANN, Andreas: *Digitalisierungsbericht 2014: Rundfunk und Internet - These, Antithese, Synthese?* VISTAS Verlag, 2014. – ISBN 9783891585900
- [GHJV09] GAMMA, Erich ; HELM, Richard ; JOHNSON, Ralph ; VLISSIDES, John: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series, 2009
- [GKRS13] GHARBI, Mahboub ; KOSCHEL, Arne ; RAUSCH, Andreas ; STARKE, Gernot ; PREISENDANZ, Christa (Hrsg.): *Basiswissen für Softwarearchitekten*. dpunkt.verlag GmbH, 2013
- [HHMS09] HINDEL, Bernd ; HÖRMANN, Klaus ; MÜLLER, Markus ; SCHMIED, Jürgen ; PREISENDANZ, Christa (Hrsg.): *Basiswissen - Software-Projektmanagement*. 3., überarbeitete und erweiterte Auflage. dpunkt.verlag GmbH, 2009
- [inh14a] *7. Inheritance Mapping — Doctrine 2 ORM 2 documentation*. <http://doctrine-orm.readthedocs.org/en/latest/reference/inheritance-mapping.html>. Version: Mai 2014
- [inh14b] *Java Persistence/Inheritance - Wikibooks, open books for an open world*. http://en.wikibooks.org/wiki/Java_Persistence/Inheritance. Version: Mai 2014
- [ios14] *Apple Push Notification Services in iOS 6 Tutorial: Part 1/2 - Ray Wenderlich*. <http://www.raywenderlich.com/32960>. Version: September 2014
- [iso] *ISO/IEC 9126-1:2001 Software engineering - Product quality*
- [jqu14] *jQuery Mobile*. <http://jquerymobile.com/>. Version: Mai 2014
- [MN90] MOLICH, Rolf ; NIELSEN, Jakob ; DENNING, Peter J. (Hrsg.): *Improving a human-computer dialogue*. ACM New York, 1990
- [myb14] *myBalsamiq*. <http://balsamiq.com/products/mockups/mybalsamiq/>. Version: September 2014

- [pho14] *Phonegap*. <http://phonegap.com/>. Version: Mai 2014
- [php14] *PHPUnit – The PHP Testing Framework*. <https://phpunit.de/>. Version: Juni 2014
- [PR11] POHL, Klaus ; RUPP, Chris ; PREISENDANZ, Christa (Hrsg.) ; SCHÖNFELDT, René (Hrsg.) ; LÖTSCH, Nina (Hrsg.): *Basiswissen - Requirements Engineering*. 3., korrigierte Auflage. dpunkt.verlag GmbH, 2011
- [SL12] SPILLNER, Andreas ; LINZ, Tilo ; PREISENDANZ, Christa (Hrsg.): *Basiswissen Softwaretest*. 5., überarbeitete und aktualisierte Auflage. dpunkt.verlag GmbH, 2012
- [son14] *SonarSource - Continuous Inspection of Code Quality*. <http://www.sonarsource.com/>. Version: Mai 2014
- [wik14a] *Ajax (Programmierung)*. [http://de.wikipedia.org/wiki/Ajax_\(Programmierung\)](http://de.wikipedia.org/wiki/Ajax_(Programmierung)). Version: September 2014
- [wik14b] *JavaScript Object Notation*. http://de.wikipedia.org/wiki/JavaScript_Object_Notation. Version: September 2014
- [wik14c] *Mock-up*. <http://de.wikipedia.org/wiki/Mock-up>. Version: May 2014
- [wik14d] *Objektrelationale Abbildung*. http://de.wikipedia.org/wiki/Objektrelationale_Abbildung. Version: September 2014
- [wik14e] *Representational State Transfer*. http://de.wikipedia.org/wiki/Representational_State_Transfer. Version: September 2014
- [xam14] *Mobile Application Development to Build Apps in C# - Xamarin*. <http://xamarin.com/platform>. Version: Mai 2014
- [zha14] *ZHAW - myBalsamiq*. <https://zhaw.mybalsamiq.com/login>. Version: May 2014

Abbildungsverzeichnis

2.1	Projektstrukturplan	11
2.2	Projektplan	12
2.3	Risikomatrix	14
3.1	Systemkontext	17
3.2	Systemumgebung	18
3.3	Use-Case Diagramm	20
3.4	Navigation und Einstellungen	28
3.5	Neuigkeiten	29
3.6	Vereine	29
3.7	Anlässe	30
3.8	Matches	30
3.9	Trainings	31
3.10	Resultate	31
3.11	Fahrgemeinschaften	32
3.12	Fahrgemeinschaft erstellen	32
3.13	Satzschablone	33
4.1	System Übersicht	41
4.2	Sequenzdiagramm - Locking Verfahren	42
5.1	Aptana Studio	47
5.2	Github Repository	48
5.3	Android Emulator	49
5.4	Sonar - statische Code Analyse Backend	50
5.5	Sonar - statische Code Analyse App	50
5.6	Advanced Rest Client	51
5.7	Navigation im App	52
5.8	News Seite	53
5.9	Vereinsseite	53
5.10	Veranstaltungen	54
5.11	Anlass Details	54
5.12	Veranstaltungen	55
5.13	Fahrgemeinschaft erstellen	55
5.14	Resultate Seite	56
5.15	Einstellungen	56
5.16	Datenbankdiagramm alt	59
5.17	Datenbankdiagramm neu	59
5.18	Ablauf von Push-Nachrichten	60
6.1	Abnahmeprotokoll	64

Tabellenverzeichnis

2.1	Meilensteine	9
2.2	Geplante Abwesenheiten	11
2.3	Zeitschätzung auf Arbeitspaketebene	13
2.4	Risikoermittlung	13
2.5	Risikobewertungsschema	14
2.6	Risikobewertung	14
2.7	Risikoanalyse - Massnahmen	15
3.1	Stakeholder	19
3.2	Vorlage für Use Case Spezifikation	21
3.3	Use Case UC-1: Fahrgemeinschaft erstellen	22
3.4	Use Case UC-2: Fahrgemeinschaften anzeigen	23
3.5	Use Case UC-3: Fahrgemeinschaft beitreten	24
3.6	Use Case UC-4: Anmelden an Anlässe	25
3.7	Use Case UC-5: Informationen zu Anlässen anzeigen	26
3.8	Use Case UC-6: Informationen zu Vereinen anzeigen	27
3.9	Vorlage für Anforderungen	33
3.10	Anforderung RF-F1	34
3.11	Anforderung RF-F2	35
3.12	Anforderung RF-F3	35
3.13	Anforderung RF-F4	36
3.14	Anforderung RF-F5	37
3.15	Anforderung RF-F6	37
3.16	Anforderung RF-F7	38
3.17	Anforderung RF-F8	38
3.18	Anforderung RF-F9	39
3.19	Anforderung RF-F10	39
3.20	Qualitätsanforderung RF-NF1	40
3.21	Qualitätsanforderung RF-NF2	40
4.1	Nutzwertanalyse - App Variante	45

Glossar

In diesem Abschnitt werden Abkürzungen und Begriffe kurz erklärt.

Abk	Abkürzung
Ajax	(siehe Asynchronous JavaScript and XML)
Asynchronous JavaScript and XML	Ajax ... bezeichnet ein Konzept der asynchronen Datenübertragung zwischen einem Browser und dem Server. Dieses ermöglicht es, HTTP-Anfragen durchzuführen, während eine HTML-Seite angezeigt wird, und die Seite zu verändern, ohne sie komplett neu zu laden. ^[wik14a]
Basisfaktor	Basisfaktoren (unterbewusste Anforderungen) muss das System in jedem Fall vollständig erfüllen, sonst stellt sich beim Stakeholder massive Unzufriedenheit ein. ^[PR11]
Begeisterungsfaktor	Begeisterungsfaktoren (unbewusste Anforderungen) sind Merkmale eines Systems, deren Wert ein Stakeholder erst erkennt, wenn er sie selbst ausprobieren kann oder sie vom Requirements Engineer vorgeschlagen werden. ^[PR11]
Büchsenliste	Die Büchsenliste wurde vom Turnverein eingeführt, um die Mitgliedern anzuspornen. Ziel ist es bei jedem Wettkampf mindestens gleich gut, wie im letzten Wettkampf zu sein. In der Liste werden die letzten Resultate aufgeführt, damit man weiss, was das minimale Ziel ist. Falls ein Mitglied ein schlechteres Resultat erzielt, muss er einen vordefinierten Betrag in die Büchse (daher der Name) zahlen.
Entity	Entity (auch Entität) ist ein eindeutig zu bestimmendes Daten-Objekt
Integrated Development Environment	Eine integrierte Entwicklerumgebung (englisch Integrated Development Environment) beinhalten einen Texteditor, Compiler (falls dieser benötigt wird), Debugger, Formatierungsfunktionen und in dem Kontext der Appentwicklung die Möglichkeit der Erstellung von grafischen Benutzeroberflächen.
IDE	(siehe Integrated Development Environment)
JavaScript Object Notation	Die JavaScript Object Notation, kurz JSON ..., ist ein kompaktes Datenformat in einer einfach lesbaren Textform zum Zweck des Datenaustauschs zwischen Anwendungen. ^[wik14b]
JSON	(siehe JavaScript Object Notation)
localStorage	localStorage ist ein begrenzter Speicherbereich im Document Object Model (DOM), welcher auch nach dem Schliessen des Browsers ausgelesen werden kann
Mockup	Der aus dem Englischen stammende Begriff Mock-up oder Mockup (auch Maquette) bezeichnet im Deutschen beispielsweise eine Attrappe. Er wird heute aber meist für ein maßstäblich gefertigtes Modell bzw. eine Nachbildung zu Präsentation zwecken benutzt, demgegenüber ist der Prototyp ein funktionsfähiges Modell. ^[wik14c]
Objektrelationale Abbildung	Objektrelationale Abbildung (englisch object-relational mapping, ORM) ist eine Technik der Softwareentwicklung, mit der ein in einer objektorientierten Programmiersprache geschriebenes Anwendungsprogramm seine Objekte in einer relationalen Datenbank ablegen kann. Dem Programm erscheint die Datenbank dann als objektorientierte Datenbank, was die Programmierung erleichtert. ^[wik14d]
Object-relational mapping	(siehe Objektrelationale Abbildung)

Object Query Language	Object Query Language (OQL) ist stark an SQL angelehnt, wobei man nicht mit den Spalten der Tabelle, sondern mit den Attributen des Objekts Abfragen erstellt.
OQL	(siehe Object Query Language)
ORM	(siehe Object-relational mapping)
PHP	PHP (rekursives Akronym und Backronym für „PHP: Hypertext Preprocessor“, ursprünglich „Personal Home Page Tools“) ist eine Skriptsprache mit einer an C und Perl angelehnten Syntax, die hauptsächlich zur Erstellung dynamischer Webseiten oder Webanwendungen verwendet wird. [?]]
Projektstrukturplan	Ein Synonym für Work Breakdown Structure (WBS): Eine in der Regel an den Liefergegenständen orientierte Anordnung von Projektelementen, die den Gesamtinhalt und -umfang des Projekts strukturiert und definiert. [HHMS09]
Stakeholder	Stakeholder sind für den Requirement Engineer wichtige Quellen zur Identifikation möglicher Anforderungen des Systems. [PR11]
Session-Token	Eine Session-ID ist eine eindeutige ID, welche bei der Anmeldung generiert wird, sie kann dann zur Authentifizierung verwendet werden. Eine Session-ID ist nur eine gewisse Zeit gültig
Apple Push Notification Service (APNS) und Google Cloud Messaging for Android (GCM)	

Listings

A Anhang

A.1 Projektmanagement