



Duale Hochschule Baden-Württemberg
- Standort Stuttgart -
Fakultät für Technik

Studienarbeit “Entwicklung einer Running-App für Android”

im Studiengang “International Applied Computer Science”

Thema: Entwicklung einer Running-App für Android

Autoren: Roland Weinreich
roland.weinreich@gmail.com
MatNr. 5885985

Franz Flintzer
fflintzer@gmail.com
MatNr. 8677472

Version vom: 5. Juni 2015

Betreuer: Prof. Dr. Karl Friedrich Gebhardt

Zusammenfassung

Abstract

Inhaltsverzeichnis

Abbildungsverzeichnis	5
Tabellenverzeichnis	5
Listingverzeichnis	5
Abkürzungsverzeichnis	5
1 Einleitung	6
2 Anforderungsanalyse	6
2.1 Allgemein	6
2.2 Webserver	6
2.3 Android-App	7
2.4 Usability	7
2.5 Sicherheit	7
3 Architektur	8
3.1 Überblick	8
3.2 Datenbank	8
3.3 REST API	8
4 Analyse	9
4.1 Use Cases	9
4.2 Komponenten des Webserver	9
4.3 Komponenten der Android App	9
4.3.1 Allgemein	9
4.3.2 Kommunikation mit dem Webserver	9
4.3.3 Positionsbestimmung	9
4.3.4 Höhenbestimmung	10
4.3.5 Aufzeichnen des Laufes	11
4.3.6 Frontend der Android App	11
4.3.7 Darstellung des Laufes	11
5 Design	11
6 Implementierung	11
7 Testen	11
8 Anwendungsbeispiel	11
9 Ausblick	11
10 Fazit	11
Literaturverzeichnis	12
Anhang	13

Eidesstattliche Erklärung**13****Abbildungsverzeichnis**

1	Kommunikation zwischen App und Backend-Server	8
---	---	---

Tabellenverzeichnis**Listingverzeichnis****Abkürzungsverzeichnis**

CMS	Content Management System
CSS	Cascading Style Sheets
ERM	Entity Relationship Modell
GNU	GNU is not Unix
GPL	GNU General Public License
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IM	Instant Message
JS	JavaScript
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LGPL	GNU Lesser General Public License
OCR	Optical Character Recognition
RSS	Really Simple Syndication
SQL	Structured Query Language
TDD	Test-driven development
UGC	User Generated Content
WWW	World Wide Web
XMPP	Extensible Messaging and Presence Protocol

1 Einleitung

Als duale Studenten an unterschiedlichen Praxis- und Theoriestandorten gibt es längere Zeiträume in denen man Familie und Freunde nicht sehen kann. Das gemeinsame Joggen in der Freizeit fällt dann natürlich auch weg. Dazu kommen verschiedene Arbeitszeiten und teils sogar verschiedene Zeitzonen, die ein Treffen schon zeitlich nicht zulassen würden.

Besonders fehlen dann die Motivation überhaupt Sport zu machen sowie der gegenseitige Ansporn durchzuhalten oder das Tempo zu steigern.

Genau hier möchten wir mit unserer Android-App ansetzen, die es Läufern erlaubt, zeitlich und vom Ort unabhängig gegeneinander anzutreten.

2 Anforderungsanalyse

2.1 Allgemein

Die App soll es ermöglichen, einen Lauf mitzuschneiden, um ihn anschließend an eine Gruppe an Freunden zu veröffentlichen. Diesen ist es dann möglich, mit ihren Freunden zu laufen bzw. gegen diese anzutreten.

Um diese Funktionalität zu ermöglichen sind das Schreiben einer Android-App und das Aufsetzen eines Webserver nötig.

2.2 Webserver

Der Webserver stellt eine Schnittstelle bereit, die einem Klienten nach der Authentifizierung folgende Funktionalitäten bereitstellt:

- Nutzer abzufragen, anzulegen und zu löschen,
- Läufe abzufragen, anzulegen und zu löschen,
- Gruppen abzufragen, anzulegen und zu löschen,
- Nutzer zu Gruppen hinzuzufügen oder zu entfernen,
- Freunde einzuladen und
- Kacheln für Höhendaten anzufordern und herunterzuladen.

Hierbei werden folgende Anforderungen gestellt:

- Die Authentifizierung muss sicher sein, sodass das Passwort des Nutzers und seine persönlichen Daten sicher vor Fremdzugriff und Manipulation durch Unbefugte sind.

- Der Server muss innerhalb von 5s Sekunden reagieren.
- Mobile Datenbeschränkungen müssen bedacht werden, sowohl in Bezug auf Geschwindigkeit als auch Datenmengen.

2.3 Android-App

Die Android-App soll dem Nutzer folgenden Funktionalitäten bereitstellen:

- Anmeldung beim Webserver
- Registrierung beim Webserver
- Abmeldung vom Webserver
- Laufen ohne währenddessen gegen Freunde anzutreten
- Laufen um währenddessen die Position der Freunden mitverfolgen zu können
- Eintreten in Gruppen, Austreten aus Gruppen und Erstellen von Gruppen
- Einladen appfremder Nutzer in eine Gruppe
- Einladen anderer Nutzer der App in eine Gruppe
- Weitergabe des Administratorprivilegs innerhalb der Gruppe durch den Gruppenadministrator

2.4 Usability

Die App muss einfach zu bedienen sein. Wir richten uns allgemein an Sportler und Sportinteressierte. Außer grundlegenden Kenntnissen zur Nutzung von Android-Apps können wir keine Annahmen zu den technischen Fähigkeiten des Nutzers machen.

Teile der App sind für die Nutzung beim Joggen. Auch hierbei muss es dem Nutzer möglich sein relevante Informationen zu erkennen und sicher mit der App zu interagieren.

2.5 Sicherheit

Um Nutzer zu authentifizieren ist eine Nutzernamen/Passwort-Vergabe notwendig. Diese müssen ausreichend gesichert sein, um unauthorisierte Zugriffe auf Ressourcen zu verhindern.

Außerdem muss die Sicherheit des Gerätes zu jedem Zeitpunkt gewährleistet sein, d.h. die Einschleusung über die App von Schadcode darf nicht möglich sein.

3 Architektur

3.1 Überblick

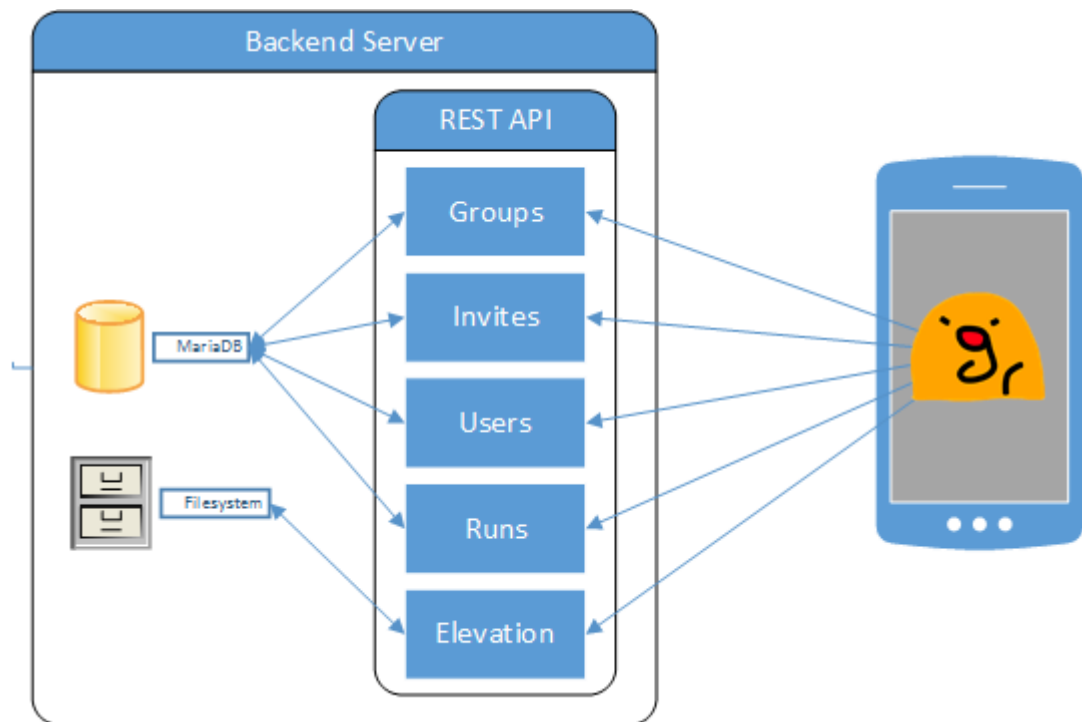


Abbildung 1: Kommunikation zwischen App und Backend-Server

Im folgenden erkläre ich die auf der Abbildung dargestellten Komponenten und warum diese für das Projekt gewählt wurden.

3.2 Datenbank

Als Datenbankverwaltungssystem haben wir uns entschieden “MariaDB” zu nutzen. MariaDB ist das Pendant zu “MySQL”, jedoch ist es schneller, sicherer und hat eine aktivere Community. Beide Teammitglieder könne bereits auf längere Erfahrung in MySQL zurückgreifen. Die Arbeit mit MariaDB war aufgrund des weitgehend identischen Befehlssatzes sehr unkompliziert.

3.3 REST API

Um der App eine geeignete Schnittstelle zu Daten über das Internet bereitzustellen haben wir uns für eine REST API entschieden. REST steht für “Representational State Transfer”, einer stetig wachsenden

4 Analyse

4.1 Use Cases

4.2 Analyse von Entitäten

4.3 Komponenten des Webservers

4.4 Komponenten der Android App

Die folgenden Abschnitte befasst sich mit der Analyse der Komponenten der Ghostrunner App. Notwendig sind für die Aufzeichnung des Laufes beispielsweise eine Möglichkeit der Positions- und Höhenbestimmung. Es muss außerdem eine Möglichkeit der Authentifizierung und Kommunikation mit dem Webserver geben. Die Android Plattform wird auf mögliche Lösungen untersucht, welche hier beschrieben werden.

4.4.1 Kommunikation mit dem Webserver

Für die Kommunikation mit dem Webserver muss eine Internetverbindung zur Verfügung stehen. Diese ist bei allen Geräten, auf denen unsere App laufen soll Standard. Programmatisch wird also nur ein HTTP-Client benötigt, der von der Android-Plattform bereitgestellt wird. Er ermöglicht, dass HTTP-Anfragen wie GET und POST gestellt werden und Informationen in HTTP-Header und Body übergeben werden können. Beim Warten auf Antworten vom Webserver kommt es automatisch zu nicht vorhersehbaren Verzögerungen. Android erlaubt es aus diesem Grund nicht, diese Netzwerkanfragen im Main Thread der App zu starten, weil sonst während der Wartezeiten die komplette Benutzeroberfläche einfriert. Bei den Standard HTTP-Clients von Android kann die Netzwerkanfrage durch die Benutzung der Klasse AsyncTask in einen separaten Thread ausgelagert werden. [Pro]

Vereinfacht werden kann dieser Schritt durch die Benutzung des Android Asynchronous HTTP Client. Die quelloffene Bibliothek bietet einfache funktionen für die HTTP-Anfragen GET, POST, PUT und DELETE, die für die Kommunikation mit RESTful-APIs notwendig sind und empfängt Antworten vom Server automatisch asynchron, wodurch der UI-Thread nicht blockiert wird. [Smi]

4.4.2 Anmeldung und Authentifizierung

Ein weiterer wichtiger Aspekt ist die Authentifizierung beim Webserver. Nach einer Anmeldung mit Nutzernamen und Passwort wird vom Server ein Sicherheitstoken erstellt, das bis zu seinem ebenfalls vom Server definierten Ablaufdatum von Client benutzt werden kann, um sich anzumelden. So kann bei einem eventuellen Angriff auf die Netzwerkkommunikation nur noch ein zeitlich, nicht mehr die vollständigen Es macht Sinn, diese Authentifizierung an einer einzigen Stelle im Client zu behandeln.

4.4.3 Positionsbestimmung

Für eine genaue Laufanalyse muss zu jedem Zeitpunkt die Position des Läufers bekannt sein, denn daraus lassen sich Geschwindigkeit und gelaufene Strecke berechnen. Die Positionsbestimmung erfolgt über das eingebaute GPS-Modul des Smartphones, dessen Genauigkeit in Städten durch WLAN-basierte Ortung und GSM-Ortung durch das Mobilfunknetz verbessert werden kann. Die Module zur Ortung können über das Package `android.location` der Android API angesprochen werden. Eine Positionsbestimmung ist also mit Boardmitteln eines Android Smartphones ohne Umwege möglich. Eine neuere und verbesserte Möglichkeit bietet die Google Location Services API. Sie ist Teil der Google Play Services, also auf allen Android Geräten verfügbar, die diesen und andere Google Dienste installiert haben, was einem Großteil der Geräte entspricht. Die verschiedenen Ortungsmöglichkeiten werden hier optimiert und zusammengeführt. Programmierer können auf einfache Weise Positionsupdates anfordern und diese über verschiedene Parameter bezüglich Genauigkeit, Updatehäufigkeit und Akkuverbrauch seinen Bedürfnissen anpassen. [?]

4.4.4 Höhenbestimmung

Zusätzlich zu Positions- und Zeitmessung wird für unsere Anwendung noch eine weitere Variable - die Steigung - benötigt. Wenn Läufer verschiedene Strecken laufen, muss diese berücksichtigt werden, um einen fairen Vergleich zu schaffen. Dazu soll später Läufern, die eine größere Steigung überwinden mussten als andere ein Zeitbonus zugesprochen werden.

Für die Bestimmung der Höhe gibt es verschiedene Möglichkeiten, die im Folgenden kurz erläutert werden.

Höhenbestimmung per GPS Grundsätzlich ist per GPS eine dreidimensionale Positionsbestimmung möglich, also neben der horizontalen Positionierung auch eine vertikale Höhenmessung. Diese kann jedoch aufgrund der für horizontale Positionsbestimmung optimierten Positionierung der GPS-Satelliten starken Schwankungen unterliegen. Insbesondere in Städten kommt hierzu das Problem der Reflektion des GPS Signals an hohen Gebäuden, durch die die Positionsgenauigkeit weiter beeinträchtigt wird. Die Beeinträchtigung ist horizontal weniger signifikant als vertikal, und wird durch die Nutzung der im vorherigen Abschnitt genannten zusätzlichen Positionierungsmethoden noch weiter optimiert. Vertikal ist eine solche Optimierung nicht möglich. Noch größer ist die Fehlersignifikanz bei dem Versuch, die Höhe über Normalnull zu berechnen, die von dem vom GPS-System genutzten rein mathematischen Ellipsoid WGS-84 abweicht. Durch die Umrechnung vergrößern sich entsprechende Fehler. [?] Letzteres ist allerdings für unsere Arbeit aber nicht relevant, da wir keine genaue Höhe, sondern nur die relativen Unterschiede für die Steigungsberechnung benötigen. Trotzdem kön-

nen die möglichen starken Schwankungen der gemessenen Höhe für unsere Anwendung zum Problem werden.

Höhenbestimmung per Web-API

Höhenbestimmung per SRTM-Rohdaten

Kombination verschiedener Methoden Insbesondere durch Kombination von GPS und Nutzung der SRTM-Daten sollten sich sehr genaue Ergebnisse erzielen lassen. Die Implementierung des Höhenbestimmungsmoduls sollte also flexibel genug sein, um das zuzulassen. Erreicht wird diese Flexibilität durch die Implementierung eines Interfaces **ElevationService**, das die Methode **getElevation(double latitude, double longitude)** besitzt. Durch Nutzung dieses Interfaces können verschiedene Services einfach ausgetauscht oder mit wenig zusätzlichem Programmieraufwand kombiniert werden. Hier muss nur noch ein Algorithmus entwickelt werden, der bestimmt, auf welche Weise verschiedene Höhendaten kombiniert werden. Für die Studienarbeit haben wir uns aber für die explizite Implementierung der SRTM-Methode entschieden, Veränderungen in späteren Versionen sind durch das Interface jederzeit ohne großen Aufwand möglich.

4.4.5 Aufzeichnen des Laufes

4.4.6 Frontend der Android App

4.4.7 Darstellung des Laufes

5 Design

6 Implementierung

7 Testen

8 Anwendungsbeispiel

9 Ausblick

10 Fazit

Literaturverzeichnis

- [Pro] PROJECT, Android Open S.: *Connecting to a Network*. <http://developer.android.com/training/basics/network-ops/connecting.html>
- [Smi] SMITH, James: *Android Asynchronous Http Client*. <http://loopj.com/android-async-http>

Anhang

Eidesstattliche Erklärung

Eidesstattliche Erklärung zur <-Arbeit>

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Unterschrift :

Ort, Datum :

