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

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 gegenseite Ansporn durchzuhalten oder das Tempo zu steigern.

Genau hier möchten wir mit unserer Android-App ansetzen, die es Läufern erlaubt, zeitlich und vom Ort unabhänigig 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 Aufzetzen 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 Nutzername/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

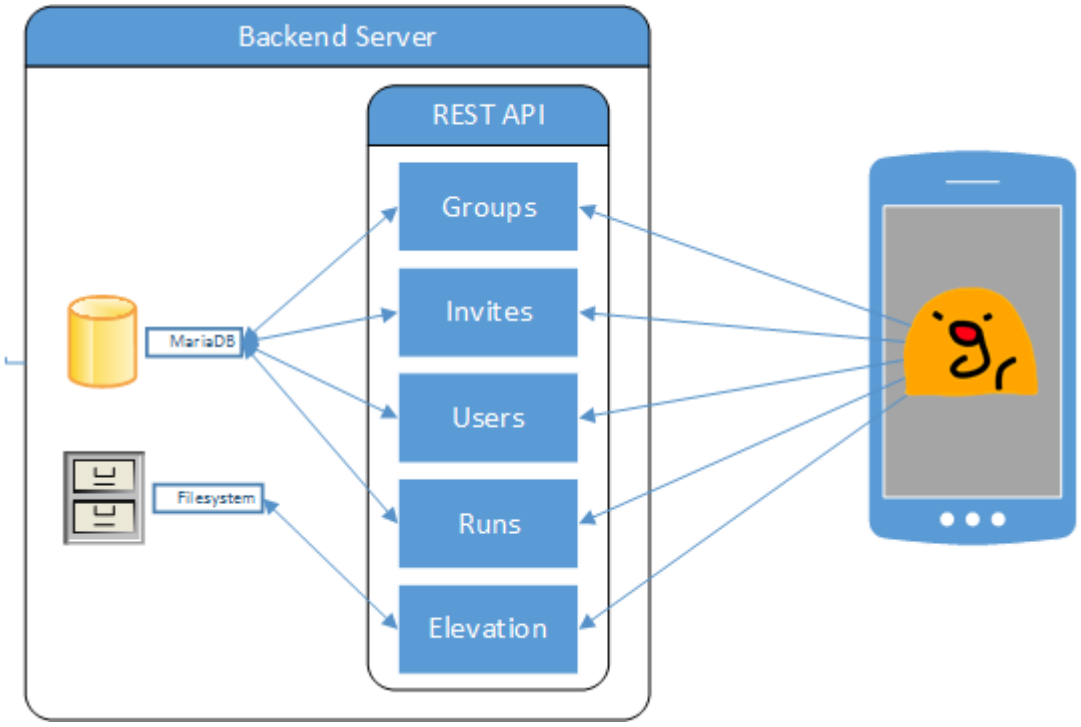

Abbildung 1: Kommunikation zwischen App und Backend-Server

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

3.2 Backend Server

3.2.1 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.

3.2.2 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”, einem stetig wachsenden Architekturmodell für Client-Server-Kommunikation bei Web Services.

REST Allgemein REST ist zustandslos, d.h. jede Nachricht enthält genügend Informationen um den Inhalt zu verstehen. Das ist besonders wichtig mit Augenmerk auf Skalierbarkeit. So ist die Lastenverteilung der REST-Anfragen auf mehrere Systeme sehr einfach, da keine sitzungspezifischen Daten o.Ä. zwischen den Servern synchronisiert werden müssen. Das gleiche gilt natürlich auch für High-Availability-Systeme.

REST schöpft den HTTP Funktionsumfang weitgehend aus, um u.A. erweitertes Caching und eine hohe Selbsterklärungsfähigkeit zu garantieren.

So erlaubt die Operation "POST" es, neue Ressourcen zu erstellen.

"GET" ermöglicht es, Resourcen anzufragen. "GET" ist nullipotent, d.h. es werden unter Garantie keine Änderungen an der Ressource stattfinden.

Mit "PUT" kann man eine Ressource ersetzen, mit "DELETE" löschen. Diese beiden Operationen bezeichnet man als "idempotent". Das bedeutet dass die Anfrage das gleiche Ergebnis produzieren wird, egal wie häufig diese wiederholt wird.

Jeder Ressource ist genau eine URL zugeordnet. Beispielsweise bekommt so der Client mit einer Anfrage mit der Operation "GET" auf "myserver.com/run" die Liste aller Läufe zurück, während die gleiche Anfrage auf "myserver.com/run/15" nur den Lauf mit der ID 15 zurückgibt.

Ein weitere Eigenschaft von REST ist es, dass die Kommunikationsform nicht standardisiert ist. So ist es möglich, anhand von Daten verpackt in JSON, HTML, XML oder auch in einem komplett eigenen Format zu kommunizieren.

Spring Framework und Tomcat Für die Realisierung der Webservices haben wir uns entschieden das Spring Framework zu nutzen. Ersteres wurde gewählt da es bei einem Teammitglied relevant für die Bachelorarbeit wird und sich so in das Framework eingearbeitet werden konnte.

Das quelloffene Framework erlaubt es u.A. skalierbare, performante, reichhaltige Web-Services aufzusetzen. Es ist möglich spezifische Sicherheitsmaßnahmen umzusetzen, JSON, XML o.Ä. automatisch zu serialisieren und deserialisieren, Anfragen zu validieren usw., d.h. es ist mehr als ausreichend für den Rahmen dieser Studienarbeit.

Die Programmierung im Spring-Framework erfolgt mit Java (mit zusätzlichen Annotationen) und XML. Es vereinfacht das Programmierprinzip Dependency Injection durch wahlweise das automatische Einlesen von hinterlegten XML-Dateien oder Annotationen im Java-Code. Auch das Programmierparadigma "Aspektorientierte Programmierung" wird dem Programmierer nahegelegt. Spring besteht aus vielen kleineren Frameworks, von denen einige Verwendung im Projekt gefunden haben:

- Sping Core - Das "Kern"-Framework.
- Spring Web - Um Web-Applikationen zu erstellen.

- Spring Web MVC - Fügt ein DispatcherServlet hinzu dass Anfragen an Handler/Controller weiterleitet und damit das MVC-Programmiermuster stark vereinfacht.[1].
- Spring Security - Die Sicherheitskomponente für das Spring Framework.

Als Application Server wurde Tomcat gewählt, da beide Teammitglieder diesen vorher bereits mehrfach aufsetzen durften und eventuell spätere Sicherheitsmaßnahmen wie das Einbinden eines Zertifikats um den HTTP-Verkehr über SSL zu verschlüsseln in wenigen Minuten möglich ist.

JSON Als Format für den Austausch von Daten zwischen Server und Client haben wir uns für JSON entschieden.

JSON weist im Vergleich zum stärksten Konkurrenten XML eine hohe Lesbarkeit auf und lässt sich sowohl auf dem Server als auch auf der App sehr einfach von und zu Java-Objekten serialisieren und deserialisieren.

Auf dem Server übernimmt diese Aufgabe Jackson, auf der App Gson.

Maven Das manuelle Einbinden von JARs in Java-Projekte stellt eine Herausforderung da. Hierbei müssen diese von vertrauenswürdigen Quellen manuell heruntergeladen werden, in den richtigen Ordner verschoben und importiert werden.

Um den Aufwand zu minimieren nutzen wir Maven, welche sich selbst um das Herunterladen und Einbinden der Java-Archive kümmert. Dabei ist auch das Aktualisieren der Pakete durch das Ändern einer einzigen Zeile in der Maven-Konfigurationsdatei möglich.

3.2.3 App

Die App ist für Android 5.0 ab API 21 geschrieben.

Für das Betriebssystem Android haben wir uns entschieden da beide Teammitglieder ein Android-Smartphone besitzen. Da es sich bei dieser Studienarbeit um kein Projekt mit wirtschaftlicher Motivation handelt und die API einige Verbesserungen gegenüber älteren Versionen mitbringt hielt uns von der Entscheidung auch nicht die relativ geringe Verbreitung von 12,4% der APIs 21 und 22 ab[2] (Stand 1. Juni 2015).

[1] Introduction to Spring Web MVC framework, vgl. [JHD+] [S.437]
[2] Dashboards, vgl. [Unb]

4 Analyse

5 Design

6 Implementierung

7 Testen

8 Anwendungsbeispiel

9 Ausblick

10 Fazit

Literaturverzeichnis

[JHD+] Johnson, Rod ; Hoeller, Juergen ; Donald, Keith ; Sampaleanu, Colin ; Harrop, Rob ; Risberg, Thomas ; Arendsen, Alef ; Davison, Darren ; Kopylenko, Dmitriy ; Pollack, Mark ; Templier, Thierry ; Vervaet, Erwin ; Tung, Portia ; Hale, Ben ; Colyer, Adrian ; Lewis, John ; Leau, Costin ; Fisher, Mark ; Brannen, Sam ; Laddad, Ramnivas ; Poutsma, Arjen ; Beams, Chris ; Abedrabbo, Tareq ; Clement, Andy ; Syer, Dave ; Oliver Gierke, Rossen S. ; Webb, Phillip ; Winch, Rob ; Clozel, Brian ; Nicoll, Stephane ; Deleuze, Sebastien: *Spring Framework Reference Documentation*. `http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html`

[Unb] Unbekannt: *Dashboards*. `http://http://developer.android.com/about/dashboards/index.html`

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.

$Unterschrift$: $Ort, Datum$: