Entwicklung einer Einkaufslisten-App

Markus Hülß
E-Mail: huma1200@stud.hs-coburg.de

Malcolm Kögler
E-Mail: koma1200@stud.hs-coburg.de

Tanja Mögn
E-Mail: mota1200@stud.hs-coburg.de

Daniel Müller
E-Mail: muda1200@stud.hs-coburg.de

***Abstract*—Mangels Verfügbarkeit von multiplattformfähigen Apps zur Verwaltung von Einkaufslisten entstand die Idee, eine solche App zu entwickeln. Um den Entwicklungsaufwand möglichst gering zu halten und nicht für jede Plattform separat eine App zu programmieren, entschieden wir uns für das Framework Phonegap. Dieses Paper beschäftigt sich mit der Konzeption und Entwicklung der App.**

I. Einleitung

Immer präsent und nicht mehr aus dem Alltag wegzudenken, unterstützt uns das Smartphone heute schon bei der Organisation und Kommunikation. Vor einigen Jahren hat man vor dem Einkauf noch schnell einen Einkaufszettel geschrieben. Aber wie schnell war der Zettel doch in der Tasche spurlos verschwunden, obwohl man sich so sicher war, ihn eingepackt zu haben. Heutzutage bietet sich das Smartphone als Möglichkeit an, einen Einkaufszettel zu speichern. Das bietet unter anderem die Möglichkeit, den Einkaufszettel zu jeder Zeit zu erweitern oder ihn mit anderen zu teilen. Ziel dieser Arbeit ist es, eine App zur Verwaltung von Einkaufszetteln zu entwickeln. Diese soll es ermöglichen, erstellte Einkaufszettel mit anderen zu teilen. Die jeweilige Plattform des App-Benutzers sollte dabei kein Hindernis darstellen. Die Projektgruppe wurde in zwei Teams aufgeteilt. Ein Team war für die Frontend-Entwicklung zuständig, das andere für das Backend. Der Programmcode wurde mittels des Versionierungssystems GIT verwaltet. Grafiken wurden mit Balsamiq und MS Visio 2013 erstellt. Die Kommunikation erfolgte teils persönlich, teils über E-Mails.

Am Anfang wurde eine Konkurrenzanalyse angefertigt, um einen Überblick über die bereits am Markt verfügbaren Apps zu ermöglichen. Als Nächstes wird die Erstellung von Prototypen anhand von Mockups beschrieben, welche die Grundlage für die Entwicklung bildeten. Darauf folgend werden die verwendeten Technologien in Frontend und Backend im Zusammehang mit dem Nutzen in der App vorgestellt. Zuletzt werden in einer Zusammenfassung die Projekterfolge und Misserfolge aufgezeigt.

II. Konkurrenzanalyse

Um sich einen Überblick über vorhandene Apps zu verschaffen, wurde eine Konkurrenzanalyse erstellt. Diese Konkurrenzanalyse beschränkt sich auf native Apps und lässt Cloud-basierte Lösungen außen vor. Zu diesem Zweck wurden die Top-Apps aus den App-Stores für die Plattformen Android, IOS und Windows Phone miteinander verglichen. Um die Apps miteinander vergleichen zu können, wurden drei Merkmale definiert. Diese Merkmale waren Plattformunabhängigkeit bzw. Multiplattformfähigkeit, die Möglichkeit, Einkaufslisten mit anderen Leuten zu teilen sowie eine zentrale Datenhaltung mit Sychronisierung der Daten über mehrere Geräte hinweg.

Am Markt gibt es mittlerweile mehrere Apps, die darauf spezialisiert sind, Einkaufslisten zu verwalten. Nur eine App bietet eine native Implementierung für die oben genannten Plattformen. Einige Apps bieten eine reduzierte Multiplattformfähigkeit in dem Sinne, dass ein Webservice zu Verfügung gestellt wird, über den auf die Daten zugegriffen werden kann. Die meisten Apps bieten ein Backbone zur zentralen Datenhaltung. Nur wenige Apps bieten die Möglichkeit, Einkaufslisten mit anderen zu teilen. Diese Fähigkeit ist teilweise darauf beschränk,t eine Einkaufsliste per E-Mail zu verschicken.

III. Mockups

Zu Beginn der Entwicklung wurde eine Möglichkeit gesucht, schnell einen Prototypen der App zu entwickeln. Zu diesem Zweck wurden Mockups für das Frontend erstellt. Diese wurden in Teamarbeit entworfen und somit hatte jeder im Team eine Vorstellung, wie die App später aussehen soll. In Abbildung 1 ist ein Beispiel für ein solches Mockup zu sehen. Die Abbildung 1 zeigt die Ansicht eines Einkaufszettels mit den zugehörigen Steuerelementen.

IV. Frontend

Ein wichtiges Ziel ist es, eine plattform- und auflösungsunabhängige Darstellung zu ermöglichen. Dabei soll das Frontend nicht für jede Plattform separat entwickelt werden. Dies soll sowohl den Zeitaufwand als auch potenzielle Fehlerquellen verringern. Der Browser dient als Grundlage für die Darstellung. Somit ist eine einfache Darstellung auf mobilen Geräten und Desktop-PCs möglich. Dieser Abschnitt befasst sich mit den beiden Komponenten ”Phonegap” und ”jQuery mobile”.

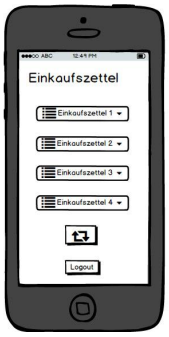

Abbildung 1: Einzelansicht

A. Phonegap

Die auf dem Markt vorhandenen Applikationen entwickeln für Android, iPhone und Windows Phone jeweils separate native Apps. Das erfordert tiefgehendes Know-how für die einzelnen Plattformen und weitergehende separate Behandlung von Fehler- und Change-Requests. Aus diesem Grund wird bei der Entwicklung der App auf Phonegap gesetzt. Phonegap ist ein Framework zum Erstellen von nativen Anwendungen für die gängigsten mobilen Plattformen durch die Verwendung von HTML5. Das Konzept von Phonegap ist, auf einer Seite das Frontend einer Anwendung über eine angezeigte Webseite im Browser zu realiseren. Auf der anderen Seite stellt es zusätzlich Javasript API's bereit, die einem Zugriff auf bestimmte Hardware wie die Kamera oder bestimmte Sensoren ermöglichen. Auch kann auf Events wie beispielsweise eine schwache Funkverbindung reagiert werden. Somit sind Funktionalitäten nativer Apps gegeben, ohne sich intensiv mit der Architektur jeglicher Zielplattformen auseinanderzusetzen. Zusätzlich baut Phonegap am Ende aus diesen Komponenten eine native App, welche sich in dem dementsprechenden Store anbieten lässt.

B. jQuery Mobile

Ein zusätzliches Problem neben den unterschiedlichen Plattformen, ist die Vielzahl an unterschiedlichen Browsern und Displayauflösungen der Endgeräte. Die Anwendung soll ein einheitliches Design, unabhängig von Browser und Auflösung besitzen. Ein Benutzer soll sich nicht bei einem Wechseln zwischen einem normalen und einem mobilen Browser bei der Bedienung umgewöhnen müssen. Um dieser Anforderung nachzukommen, wird auf das jQuery Mobile Framework zurückgegriffen. Es ermöglicht eine einfache Umsetzung von Webseiten mit einem responsive Design und bietet eine große Auswahl unterstützter Browser (sowohl mobile als auch non-mobile). Somit wird mit einer einzigen Version des Frontends eine große Menge an Endgeräten unterstützt und dies trägt zu einer schnellen Entwicklung der Anwendung bei. Zusätzlich ist es mit dem Multi-Page-Layout-Konzept möglich, mit einer einzigen HTML-Seite alle Ansichten einer App darzustellen. Dadurch muss nicht bei jedem Wechsel einer Ansicht eine neue Seite heruntergeladen werden, womit sich die Performance verbessert und Traffic gespart wird. (Bisherige Quelle bisher nur die jQuery Mobile-Website)

V. Backend

Ein weiteres wichtiges Ziel ist es, eine endgeräteunabhängige Datenspeicherung zu gewährleisten. Dafür ist ein Backend, das von überall über das Internet zu erreichen ist, vonnöten. Für die Umsetzung des Projekts entschieden wir uns für einen einfachen Webserver und die Nutzung von PHP und MySQL zur Datenhaltung und für JSON nach dem ReST-Paradigma zur Kommunikation zwischen Anwendung und Backend. Dieser Abschnitt befasst sich mit den Komponenten PHP, MySQL und JSON, welche die Umsetzung der Anforderungen ermöglichen.

A. PHP

PHP wird im Backend verwendet, um die Schnittstelle zwischen Anwendung und Datenbank auf dem Webserver zu stellen. PHP unterstützt unzählige Datenbanktypen und ein breites Spektrum an Funktionalitäten zur Konvertierung in das JSON-Format.

B. MySQL

MySQL ist ein weit verbreitetes relationales Datenbanksystem und kam vor allem wegen seiner Plattformunabhängigkeit und der kostenfreien Nutzung unter der OpenSource-Lizenz zur Umsetzung der Datenbank in Frage. Bei der Auswahl spielte auch die Möglichkeit der Verwendung von verschiedenen Speicherssubsystemen eine Rolle. Durch die Verwendung von InnoDB stehen Rollback und Transaktionssicherungsmöglichkeiten zur Verfügung, dies ist ein wichtiger Gesichtspunkt im Bezug auf Datensicherheit.

C. JSON

Aufgrund der Verwendung des auf Javascript basierenden Frameworks JQuery-Mobile zur Erstellung einer Webapp lag die Verwendung von JSON als Format zum Datenaustausch zwischen Anwendung und Backend nahe. Dadurch ist es möglich, die Daten vom Backend ohne vorherige Konvertierung in der Anwendung zu verwenden. Mithilfe von PHP lässt sich so die Schnittstelle nach dem ReST-Programmierparadigma gestalten,welche die geforderten Daten im JSON zurückliefert.

VI. Architektur

Damit die einzelnen Komponenten ordnungsgemäß so zusammenarbeiten, benötigt es noch einer passenden Architektur. Aufgrund der verwendeten Technologien wird eine Drei-Schichten-Architektur verwendet. Somit lässt sich eine Trennung von Präsentation und Anwendungslogik realisieren.

Die Abbildung 2 zeigt den genauen Aufbau der Architektur. Ein Client sendet seine Aktion per HTTP-Request an den

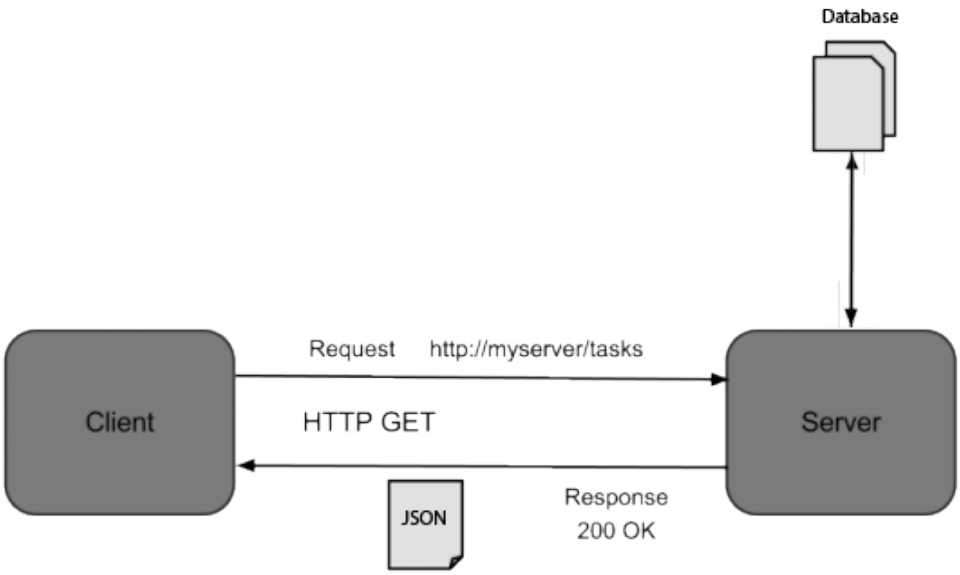

Abbildung 2: 3-Wege-Architektur für die App

Webserver. Dieser holt geforderte Daten aus der Datenbank oder ändert diese. Die von der Datenbank erhaltenen Daten werden in das JSON-Format konvertiert. Danach werden sie als Inhalt des HTTP-Response an den Client gesendet. Der genauere Ablauf ist im Abbildung 3 ersichtlich.

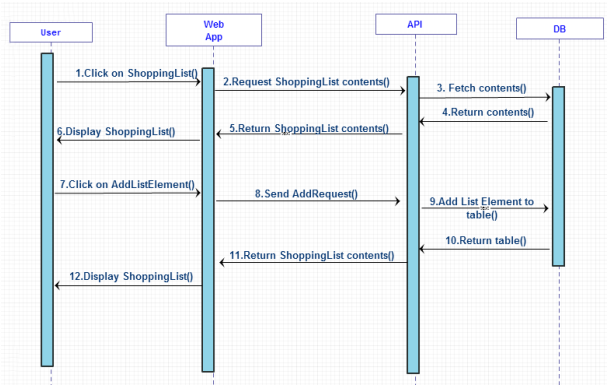

Abbildung 3: Sequenzdiagramm

VII. Fazit

Die Verwendung des Phonegap Frameworks brachte den Vorteil, dass ohne große Rücksicht auf die Plattform entwickelt werden konnte. Der Einsatz von JQuery mobile ermöglicht eine einheitliche Darstellung der mobilen Web-App auf allen Geräten. Die Einbindung von anderen Frameworks war mit Komplikationen verbunden. Da viele Javascript-Frameworks den DOM-Tree manipulieren, gestaltet sich ein gleichzeitiger Einsatz mehrerer Javascript-Frameworks als schwierig.

VIII. Ausblick

Die App besitzt bereits alle notwendigen Funktionen, um eine Einkaufsliste zu erstellen, zu editieren und zu teilen. In der weiteren Entwicklung zeigen sich die Vorteile, die von Phonegap geboten werden. So kann beispielsweise ein Offline-Modus entwickelt werden, der es ermöglichen soll, die App auch ohne permanente Internetverbindung zu nutzten. Des Weiteren ist an eine intelligentere Unterstützung der App im Alltag gedacht. Die App soll beispielsweise den User benachrichtigen, falls ein anderer User in einer gemeinsamen Einkaufsliste einen Eintrag geändert oder hinzugefügt hat. Ein weitere Verbesserung wäre auch die permanente Lokalisierung des Users und eine Reaktion der Einkaufslisten-App, wenn sich der User in der Nähe oder in einem Einkaufsladen befindet.