RHEINISCHE FACHHOCHSCHULE KÖLN
University of Applied Sciences

Fachbereich: Wirtschaft & Recht
Studiengang: Business Information Management (B.Sc.)

Dokumentation zum Laborprojekt

pick-it! - Eventmanagement

vorgelegt von: Maik Godinho (Mat.-Nr.: BWI2131010)
Denis Kündgen (Mat.-Nr.: BWI2131014)
Nina Ziegler (Mat.-Nr.: BWI2131008)

Dozent: Herr Patric Steffen

Sommersemester 2014

Inhaltsverzeichnis

1 Einleitung

Ziel dieser Projektarbeit ist die Entwicklung einer lauffähigen Webanwendung bestehend aus einem Webservice, einer Single-Page Anwendung für Desktop Browser sowie eines mobilen Webclient. Diese Webanwendung soll eine Planung von Veranstaltungen ermöglichen.

Zu einem Event können mehrere Geschenkwünsche und Mitbringsel in Form von z.B. Buffetbeiträgen erfasst werden. Andere Benutzer können sich für diese eintragen und somit reservieren. Mit dieser Webanwendung lassen sich also wunderbar Veranstaltungen, wie z.B. Grillpartys organisieren.

2 Vorgehensweise

Zur Entwicklung der Webanwendung waren unterschiedliche Aufgaben erforderlich. In gemeinsamen Workshops bzw. Skype-Konferenzen erfolgte die Verteilung und Abstimmung der einzelnen Aufgabenpakete.

- **Entwicklung eines Grobkonzepts**
  (gemeinsam)
- **Erstellung eines Datenbankmodells**
  (gemeinsam)
- **Erstellung eines PHP-Webservices**
  (Maik Godino)
- **Konfiguration der Adressumleitung mittels mod_rewrite**
  (Maik Godino)
- **Funktionalität für die Single-Page Anwendung in JavaScript**
  (Denis Kündgen)
- **CSS-Design sowie HTML-Struktur der Single-Page Anwendung**
  (Denis Kündgen)
- **Mobile-Client durch Nutzung des Sencha-Frameworks**
  (Nina Ziegler)
- **Erstellung der vorliegenden Dokumentation**
  (Nina Ziegler)

3 Anforderungsanalyse

Anhand der bereitgestellten Aufgabenstellung konnte bereits eine Vielzahl der Anforderung bestimmt werden.

Als **nichtfunktional** sind hier folgende Anforderungen aufzuführen:

- Performante Reaktion der Webanwendung, auch bei der Verarbeitung von mehr als 1000 Einträgen
- Unterstützung von mehreren, gleichzeitig interagierenden Anwendern
- Benutzerfreundliche Oberfläche und Bedienung
- Nachvollziehbare Verwendung von Konstrukten und verständliche Auscodierung
- Umsetzung als Single-Page Anwendung um PageRefreshs zu vermeiden
- Gegenseitiges Überschreiben von Änderungen verhindern

Als **funktional** sind hier folgende Anforderungen aufzuführen:

- Darstellung einer Übersicht über alle Elemente
- Anzeige von Details zu einem Element
- Bearbeiten/Anlegen/Löschen eines Elements
- Anzeige eines Bestätigungsfensters bei der permanenten Modifizierung von Daten (z.B. in der SQL Datenbank)

Während der Konzeptionierung der Anwendung konnten vier Objektentitäten identifiziert werden: Event, Entry, Contribution, User.

Ein Event (z.B. eine Geburtstagsfeier) stellt eine Sammlung von Geschenkwünschen dar, aus denen eingeladene Personen auswählen und für sich reservieren können. Es beinhaltet zudem allgemeine Informationen, wie beispielsweise ein Datum oder Veranstaltungsort. Jeder Anwender kann beliebig viele Events erstellen, einmal angelegt gehört es aber genau dieser Person.

Zu einem Event können beliebig viele Entrys (Einträge) angelegt werden. Neben dem Titel und einer erklärenden Beschreibung kann auch die maximal erforderliche Anzahl festgelegt werden. Dabei kann Host/Ersteller eines Events zwar sehen, ob ein Eintrag bereits reserviert wurde aber nicht, von wem er dieses erhalten wird.

Jeder Anwender kann einen Eintrag bzw. eine Teilmenge der gewünschten Anzahl für sich reservieren. Hierzu wird ein Objekt Contribution (Beitrag) erzeugt und der Name (wenn nicht eingeloggt) bzw. die UserID sowie die Menge gespeichert. Die Anzahl soll die gesamt gewünschte Anzahl des Entry-Objektes nicht überschreiten. Die Anzahl bereits reservierte Einträge soll ebenfalls vorab abgezogen werden.

Die Webanwendung soll sowohl im angemeldeten Zustand als auch öffentlich erreichbar sein. Einige Funktionen, insbesondere die Erstellung eines Events bzw. die Änderung eines Eintrags ist nur im angemeldeten Zustand möglich. Hierzu werden die Anwender in einem Objekt User mit Name, E-Mail Adresse (zur Passwortwiederherstellung) und Passwort als MD5 Hash gespeichert.

Fehler beim HTTP-Aufruf des Webservice werden abgefangen und mittels einer Messagebox ausgegeben. Bei der Eingabe von Inhalten wird das Datum auf syntaktische Korrektheit und die Eingaben auf Vollständigkeit (=Pflichtfelder) validiert. Sollten keine Inhalte vorhanden sein, wird dies über einen Hinweistext ausgegeben.

Zwei Berechnungen werden im Rahmen dieser Projektarbeit implementiert. Dies ist zum einen die dynamische Anzeige einer Teilmenge aller Einträge durch den Suchparameter nach ID und zum anderen der Prüfung ob die Gesamtmenge bei der Eintragung einer Contribution bereits erreicht ist.

Auf Plausibilität wird bei der Eingabe des Datums geprüft, da dieses nicht in der Vergangenheit liegen darf. Ebenfalls bildet die beschriebene Errechnung der Anzahl eine Plausibilitätsprüfung dar.

4 Technische Umsetzung

Das gesamte Webprojekt gliedert sich in drei große Aufgabenbereiche, wovon jeder einem Projektteilnehmer zugeordnet wurde. Die Zuständigkeiten sind bereits unter Punkt 2 aufgeführt.

Der Webservice bildet das Backend für HTTP-Anfragen, Validierung und Datenbankabfragen.

Der AJAX-Client bildet die Oberfläche für Desktopbrowser, nutzt den Webservice für Datenanfragen und interagiert mit dem Anwender.

Der mobile Client auf Basis des Sencha Frameworks ist auf die Nutzung durch Smartphones und Tablets ausgelegt und bietet daher kompaktere Informationen und reduzierte Interaktionsmöglichkeiten.

4.1 Webservice

Über das Apache Modul mod_rewrite werden alle HTTP Anfragen auf die Klasse RequestHandler weitergeleitet. Dieser verarbeitet die übergebenen Parameter und und HTTP Request Methode und verteilt diese an Methoden der einzelnen Objekte zur Weiterverarbeitung. Die Validierung und Fehlerabfang geschieht im Hintergrund, sofern die Anfrage erfolgreich ausgeführt wurde, wird das Ergebnis als JSON-Objekt zurückgegeben.

Exemplarisch für alle anderen Anfragen, ist im folgenden Sequenzdiagramm der Aufruf aller Events dargestellt. Zunächst liest der RequestHandler das zu einem Objekt gehörende Model. Anschließend erfolgt der Verbindungsaufbau zur Datenbank mittels einer separaten Klasse MySql, womit diese auch in allen anderen Objekten genutzt weden kann.

Nach Abfrage der Eintragsgrenze (getDirtyLimitString()) vom Request Handler kann eine SELECT Anfrage an die Datenbank gestellt werden und anschließend in einer

Schleife pro Ergebnis die untergeordneten Einträge abgefragt werden. Das Resultat der Datenbank wird gegen das Model geprüft und bei Erfolg in ein JSON-Objekt umgewandelt. Dieses erhält die aufrufende Website als Response zurück.

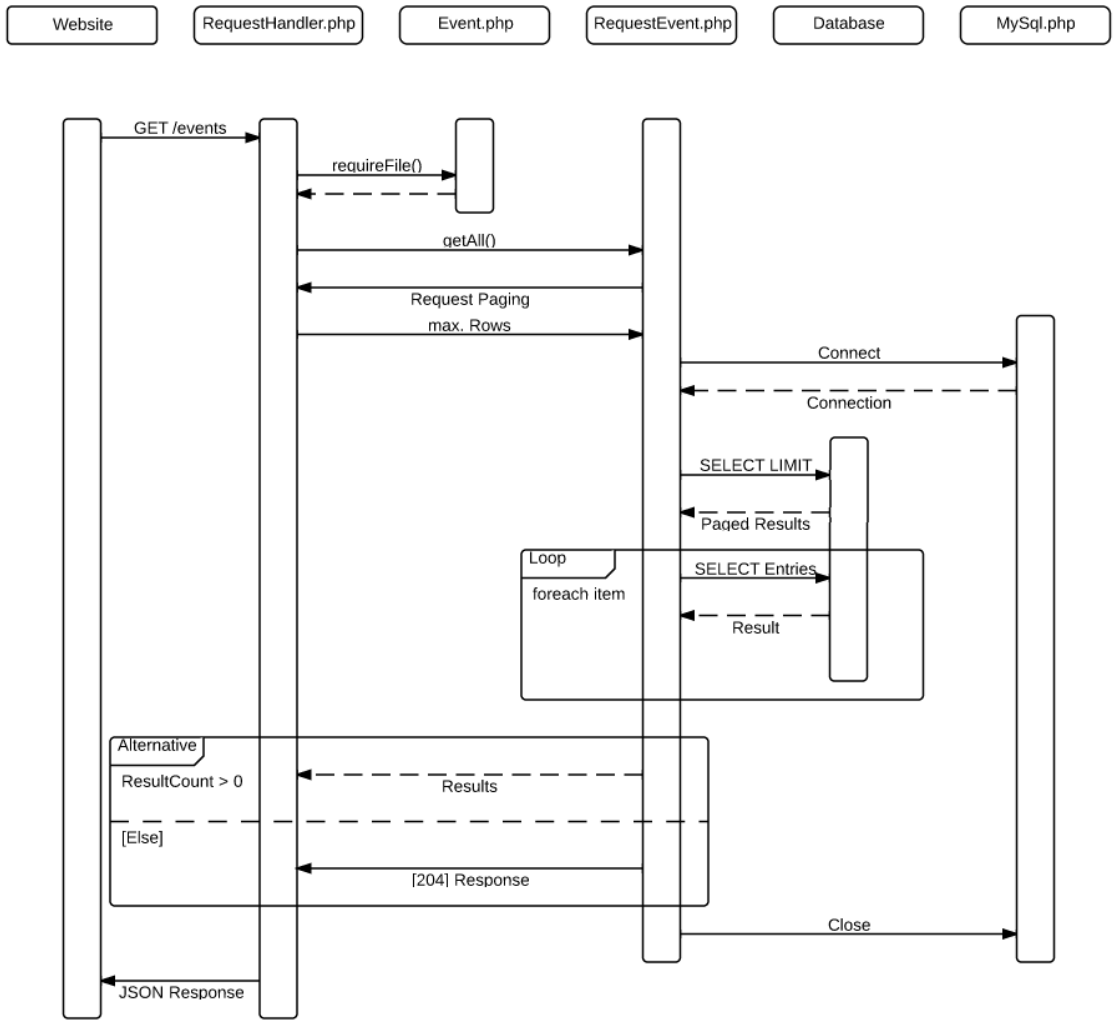

Abbildung 1 - Sequenzdiagramm einer GET Anfrage

4.2 AJAX-Client

Der AJAX-Client ist mit der JavaScript Bibliothek jQuery und den Plugins jQuery UI (für Dialoge) und blockUI umgesetzt. Aufgrund der Anforderung einer Single-Page Anwendung sind alle notwendigen HTML-Container in der index.html bereits definiert und werden nach Benutzerinteraktionen dynamisch ein- und ausgeblendet. Für jedes Widget ist eine Klasse erstellt worden um die Funktionen zu modularisieren.

Eine Suche nach Beschreibung, Titel oder Ort filtert die Einträge nach dem angegebenen Schlagwort. Aus der Menüleiste heraus kann direkt ein neues Event erstellt werden. Hierbei wird das Formular für die Anzeige von Events wiederverwendet, jedoch ohne Inhalte vorzubefüllen.

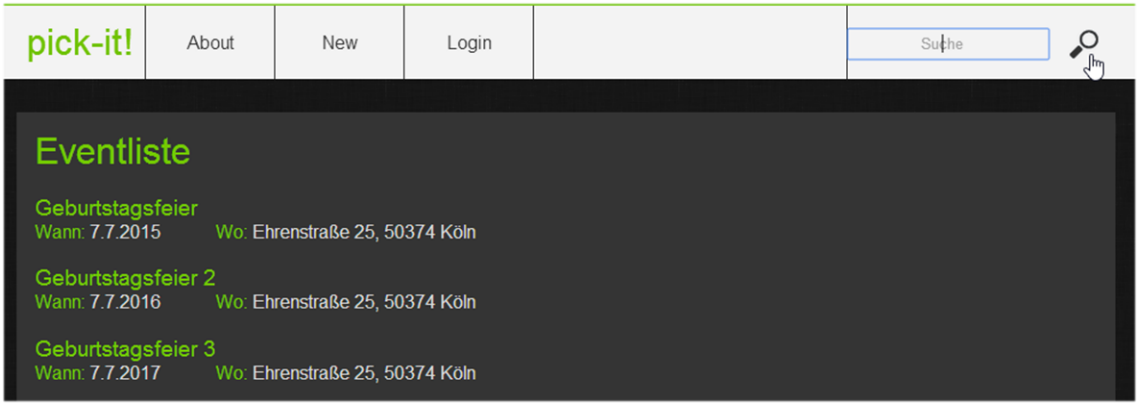

Abbildung 2 - Anzeige aller Einträge

Beim Klick auf werden die Details geladen und der aktuelle Inhalt dynamisch mit dem neuen Inhalt ausgetauscht. Dieses Formular wird direkt im Bearbeitungsmodus geöffnet. Die Validierung der Formulare erfolgt über das Widget event.eventvalidation.js und wird sofort beim Verlassen des Fokus ausgeführt.

Neben den Informationen zu einem Event werden auch die zugeordneten Einträge angezeigt. Hierbei ist die Anzahl der Eingabefelder dynamisch plus 1, das heißt sobald ein neuer Eintrag hinzugefügt wird, erweitert sich die Liste um eine neue Zeile.

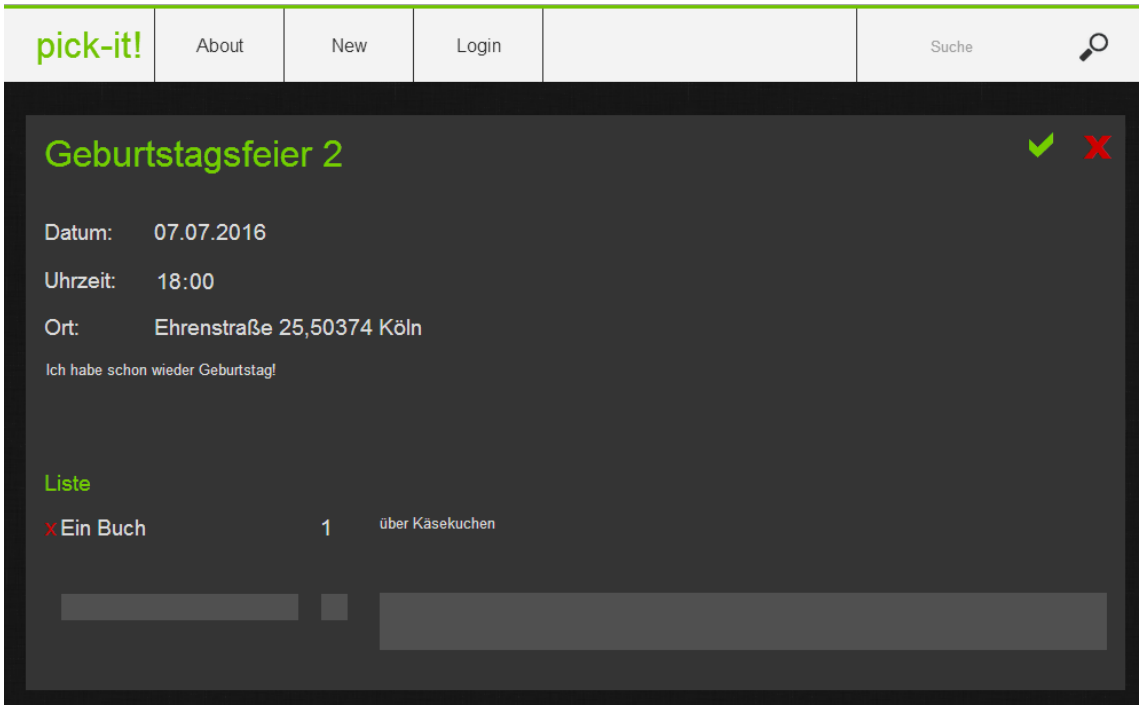

Abbildung 3 - Bearbeitungsformular eines Events

Beim Löschen oder Speichern wird dies erst nach Bestätigung in die Datenbank geschrieben. Hierfür sind entsprechende Dialoge als Widgets implementiert.

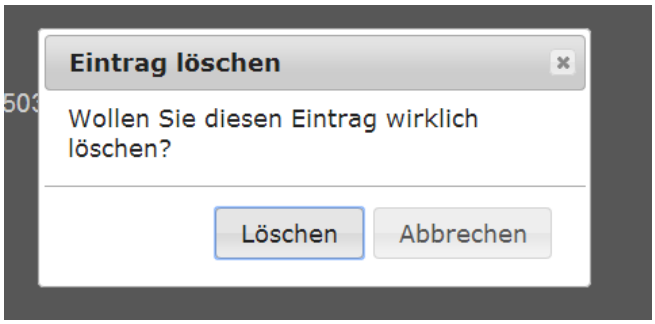

Abbildung 4 - Bestätigungsdialog

Durch das jQuery Plugin blockUI wird die Oberfläche für Eingaben gesperrt, sobald ein Request an den Webservice geschickt wird um Inkonsistenzen bei Netzwerklatenzen zu vermeiden.

4.3 Mobile-Client

Für jede Klasse wurde je ein Store, Model, View und Controller erstellt. Darüber hinaus gibt es einen allgemeinen Controller für übergreifende Events wie Aktionen in der Navigationbar und Fehlerbehandlung. Mehrere Controller kommen daher zum Einsatz, da zum einen für jedes Objekt gleichelautende Methoden unterschiedlich Aktionen ausführen und zum anderen durch die Aufteilung der Methoden die Webanwendung insgesamt strukturiert.

In der View EventListView.js können unterschiedliche Gesten Aktionen ausführen. Hier musste jedoch von der Verwendung des Events itemtap gewichen werden, da noch weitere Tap-Events geplant waren und nahezu alle auch ein itemtap auslösen würden . Mit allen bekannten Werkzeuge und Methoden konnte dieses nicht unterbunden werden, weswegen sich für die Events itemsingletap, itemdoubletap und itemswipe entschieden wurde.

Bei einem itemsingletap-Event auf ein Event wird die Methode navigateTo() des Controllers EventListDetailsCtrl aufgerufen und nur die benötigten Inhalte dynamisch in den Store EventListDetailsStore nachgeladen. Anschließend wird die EventListDetailsView auf die MainView gepusht.

Abbildung 5 - Verhalten bei itemsingletap

Bei einem itemdoubletap-Event wird das EventListForm.js instanziiert und mit den vollständigen Werten aus dem Store befüllt angezeigt. Für das Verhalten des Speichernbuttons, ist das onSave() Event des Controllers EventListCtrl zuständig. Dies führt jedoch nur zur Aktualisierung im Store aber nicht zur persistenten Speicherung in der Datenbank.

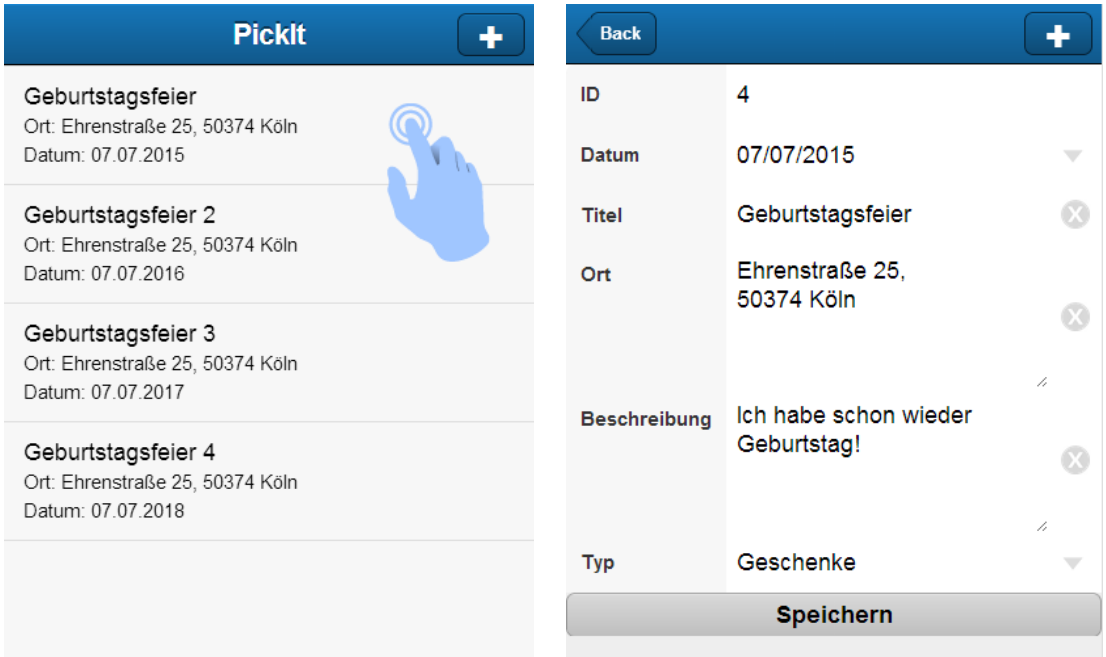

Abbildung 6 - Verhalten bei itemdoubletap

Bei einem itemswipe-Event wird ein Bestätigungsdialog geöffnet. Je nach Option verhält sich die Anwendung unterschließlich. Ein Klick auf ‘No’ schließt und belässt die Einträge unverändert. Ein Klick auf ‘Yes’ sendet einen DELETE-Request und aktualisiert bei erfolgreichem Abschluss die Auflistung aller Events.

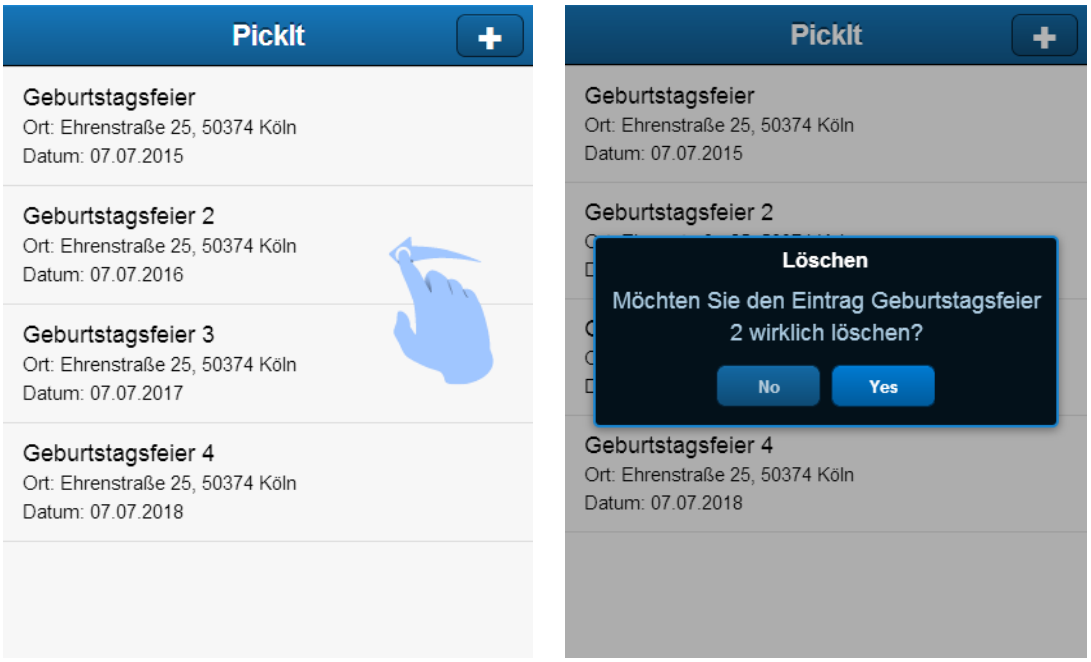

Abbildung 7 - Verhalten bei itemswipe

4.4 Datenmodell

Für jede Klasse wurde eine Tabelle in der SQL Datenbank erstellt. stamp bezeichnet hierbei die aktuelle Versionsnummer. Die Spaltennamen, Datentypen und Beziehungen können dem Diagramm entnommen warden.

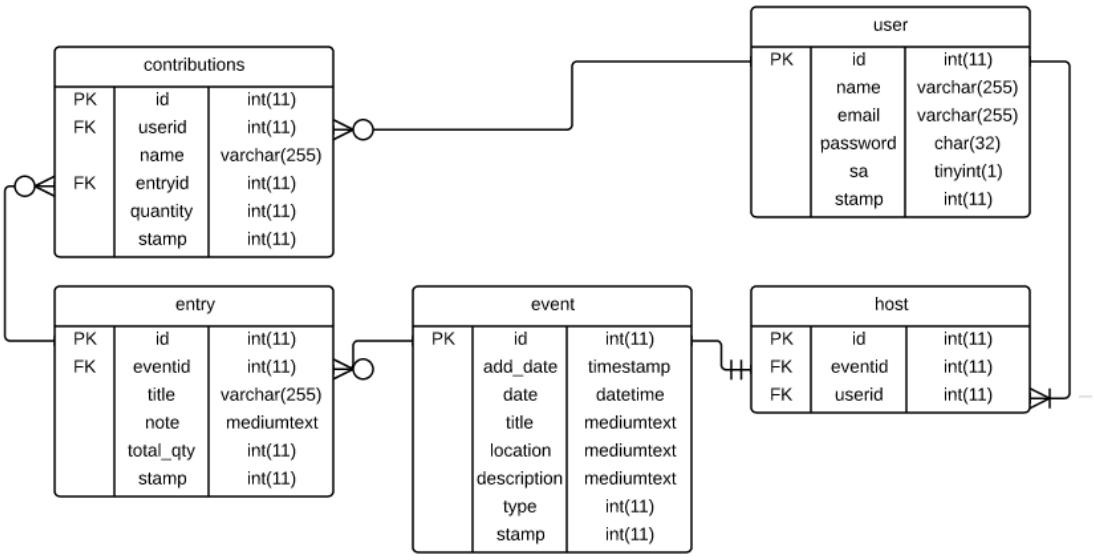

Abbildung 8 – Datenbankmodell

5 Eingereichte Ergebnisse

Alle im Rahmen dieser Prüfungsarbeit selbst erstellten Dateien befinden sich innerhalb der rechts dargestellten Ordnerstruktur. Hierbei ist, abweichend von der ursprünglichen Struktur, der AJAX-Client innerhalb des Ordners /srcphp zu finden um den Aufruf mittels mod_rewrite in einem späteren Schritt auf dieses Verzeichnis umzulenken.

Einzelne eingereichte Dateien/Bibliotheken sind nicht selbst erstellt. Diese müssen jedoch zwingend zur Lauffähigkeit dieser Webanwendung erforderlich und sind daher trotzdem aufgeführt und in der Auflistung entsprechend gekennzeichnet.

Bei der Angabe der folgenden Dateien wird der Pfad ausgehend vom Wurzelverzeichnis des Projektes (C:\xampp\htdocs\RFH-SS2014-WebMobileDevelopment-Projektarbeit\) angeben. Der Document_Root muss aber zwingend bei htdocs enden.

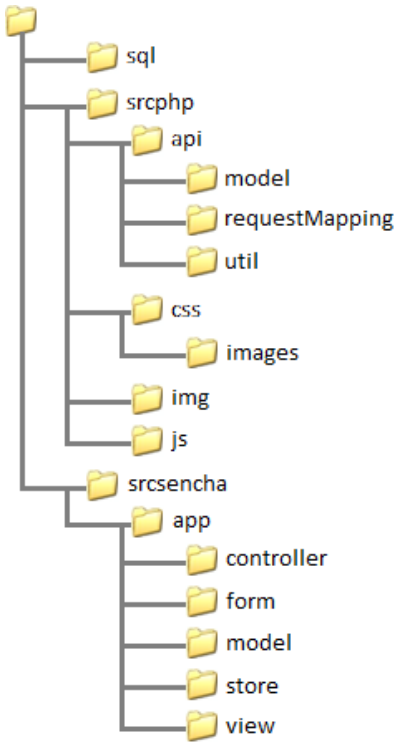

Abbildung 9 - Ordnerstruktur des Webprojekts

5.1 Installationsdateien

\sql\install.sql	Aktuellste Installationsanweisung (v003) für die SQL-Datenbank
\sql\v000to001.sql	SQL Änderungen von v000 auf v001
\sql\v001to002.sql	SQL Änderungen von v001 auf v002
\sql\v002to003.sql	SQL Änderungen von v002 auf v003
\srcphp\.htaccess	URL Weiterleitungen mittels das Apache-Modul mod_rewrite

5.2 PHP WebSerivce

\srcphp\api\MySql.php	Verbindungsaufbau zur MySQL Datenbank
\srcphp\api\RequestHandler.php	Verarbeitung aller eingehenden Webservice Anfragen und Weiterleitung an die angefragte Entität
\srcphp\api\Settings.php	Einstellungen (Zugangsdaten) für die SQL Verbindung und die Angabe der Maximalen Einträge pro Seite für das Paging.

\srcphp\api\model\Event.php	Repräsentiert ein Objekt “Event”
\srcphp\api\model\EventContribution.php	Repräsentiert ein Objekt “Contribution”
\srcphp\api\model\EventEntry.php	Repräsentiert ein Objekt “Entry”
\srcphp\api\model\IBaseModel.php	Repräsentiert ein Basis Model Interface
\srcphp\api\model\User.php	Repräsentiert ein Objekt “User”
\srcphp\api\requestMapping\IBaseRequest.php	Repräsentiert ein Basis Request Interface
\srcphp\api\requestMapping\RequestEvent.php	Definiert die Abfragen (get/update/insert/delete) für ein Event
\srcphp\api\requestMapping\RequestEventContribution.php	Definiert die Abfragen (get/update/insert/delete) für einen Beitrag (Contribution)
\srcphp\api\requestMapping\RequestEventEntry.php	Definiert die Abfragen (get/update/insert/delete) für einen Eintrag
\srcphp\api\requestMapping\RequestUser.php	Definiert die Abfragen (get/update/insert/delete) für einen User
\srcphp\api\util\IOUtil.php	Derzeit nicht verwendet.

5.3 Design für AjaxClient

\srcphp\index.html	Beinhaltet Struktur und Templates für Container, die bei der Bedienung der Webanwendung dynamisch via jQuery erzeugt/modifiziert werden
\srcphp\css\desktop.css	Alternative Definition der CSS-Eigenschaften
\srcphp\css\desktop2.css	Definition der CSS-Eigenschaften für die gesamte Single-Page Anwendung
\srcphp\css\jquery-ui-1.10.4.custom.css	Definition der CSS-Eigenschaften für jQuery Oberflächenelemente (z.B. Widgets, Popup-Fenster, etc.) (nicht selbst erstellt)
\srcphp\css\images\animated-overlay.gif	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_flat_0_aaaaaa_40x100.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_flat_75_ffffff_40x100.png	nicht mehr in Verwendung

\srcphp\css\images\ui-bg_glass_55_fbf9ee_1x400.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_glass_65_ffffff_1x400.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_glass_75_dadada_1x400.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_glass_75_e6e6e6_1x400.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_glass_95_fef1ec_1x400.png	nicht mehr in Verwendung
\srcphp\css\images\ui-bg_highlight-soft_75_cccccc_1x100.png	nicht mehr in Verwendung
\srcphp\css\images\ui-icons_222222_256x240.png	nicht mehr in Verwendung
\srcphp\css\images\ui-icons_2e83ff_256x240.png	nicht mehr in Verwendung
\srcphp\css\images\ui-icons_454545_256x240.png	nicht mehr in Verwendung
\srcphp\css\images\ui-icons_888888_256x240.png	nicht mehr in Verwendung
\srcphp\css\images\ui-icons_cd0a0a_256x240.png	nicht mehr in Verwendung
\srcphp\img\background.png	Hintergrundgrafik
\srcphp\img\gradient.png	Grüne Farbverlauf zur Hervorhebung des gehoverten Menüpunktes
\srcphp\img\logo.gif	nicht mehr in Verwendung
\srcphp\img\new-icon.png	nicht mehr in Verwendung
\srcphp\img\search.png	Lupensymbol für das Suchformular

5.4 Ajax Client

\srcphp\js\application.js	Definition von Events und Fehlerbehandlung, die allgemein auftreten können
\srcphp\js\event.canceldialog.js	Dialogfenster (Widget), das beim Abbruch einer

	Eingabe angezeigt wird
\srcphp\js\event.deletedialog.js	Dialogfenster (Widget), das beim Löschen eines Eintrags angezeigt wird
\srcphp\js\event.errordialog.js	Dialogfenster (Widget), das im Fehlerfall angezeigt wird
\srcphp\js\event.eventcreate.js	Widget zur Neuanlage bzw. Änderung eines Events
\srcphp\js\event.eventlist.js	Widget zur Auflistung aller Events bzw. aller Suchtreffer
\srcphp\js\event.eventvalidation.js	Methodemsammlung, welche den Inhalt vor dem Speichern auf fehlerhafte Eingaben überprüft.
\srcphp\js\event.menubar.js	Widget für die Navigationspunkte
\srcphp\js\event.menusearch.js	Widget für das Suchfenster
\srcphp\js\jquery-1.10.2.js	JavaScript Bibliothek u.a. zur vereinfachten, dynamischen DOM-Manipulation *(nicht selbst erstellt)*
\srcphp\js\jquery-ui-1.10.4.custom.js	JavaScript Bibliothek, beinhaltet vordefinierte UI-Elemente zur Interaktion (Dialog, Effekte, Widgets) *(nicht selbst erstellt)*
\srcphp\js\jquery.blockUI.js	jQuery Plugin um die Oberfläche für Eingaben zu sperren. *(nicht selbst erstellt)*

5.5 Mobile Client

\srcsencha\app.js	Zur Registrierung aller verwendeten Klassen, steuert außerdem die Ladeanimation und und weist auf Updates hin. *(nicht selbst erstellt)*
\srcsencha\app\controller\EventListCtrl.js	Controller für Interaktionen bezüglich eines oder mehrerer Events
\srcsencha\app\controller\EventListDetailCtrl.js	Controller für Interaktionen bezüglich eines oder mehrerer Einträge

\srcsencha\app\controller\EventSubItemsCtrl.js	Controller für Interaktionen bezüglich eines oder mehrerer Beiträge
\srcsencha\app\controller\MainCtrl.js	Controller für allgemeine Interaktionen
\srcsencha\app\model\EventDetailModel.js	Definiert die Datenstruktur eines Eintrags
\srcsencha\app\model\EventModel.js	Definiert die Datenstruktur eines Events als auch den Proxy für die Datensynchronisation via Webservice
\srcsencha\app\model\EventSubItemModel.js	Definiert die Datenstruktur eines Beitrags
\srcsencha\app\store\EventDetailStore.js	Verwaltet alle Instanzen der Model-Klasse EventDetailModel.js
\srcsencha\app\store\EventsStore.js	Verwaltet alle Instanzen der Model-Klasse EventModel.js
\srcsencha\app\store\EventSubItemStore.js	Verwaltet alle Instanzen der Model-Klasse EventSubItemModel.js
\srcsencha\app\view\EventListDetailView.js	Definiert, dass der Inhalt des EventDetailStore als Liste geladen wird. Die Formatierung ist vorgegeben.
\srcsencha\app\view\EventListForm.js	Definiert ein Formular zur Anzeige und Bearbeitung aller Details zu einem Event.
\srcsencha\app\view\EventListView.js	Definiert, dass der Inhalt des EventStore als Liste geladen wird. Die Formatierung ist vorgegeben.
\srcsencha\app\view\EventSubItemsView.js	Definiert, dass der Inhalt des EventSubItemStore als Liste geladen wird. Die Formatierung ist vorgegeben.
\srcsencha\app\view\Main.js	Definiert die Navigationsleiste bzw. den Container für den Inhalt. Initial wird die EventListView geladen.

5.6 Dokumentation

\doc\Dokumentation.docx	Die vorliegende Dokumentation
doc\API*	Die technische Dokumentation des PHP-WebServices

\doc\API\class-Event.html

\doc\API\class-EventContribution.html

\doc\API\class-EventEntry.html

\doc\API\class-IBaseModel.html

\doc\API\class-IBaseRequest.html

\doc\API\class-IOUtil.html

\doc\API\class-MySQL.html

\doc\API\class-RequestEvent.html

\doc\API\class-RequestEventContribution.html

\doc\API\class-RequestEventEntry.html

\doc\API\class-RequestHandler.html

\doc\API\class-RequestUser.html

\doc\API\class-Settings.html

\doc\API\class-User.html

\doc\API\index.html

\doc\API\package-api.html

\doc\API\package-None.html

\doc\API\resources

\doc\API\source-class-Event.html

\doc\API\source-class-EventContribution.html

\doc\API\source-class-EventEntry.html

\doc\API\source-class-IBaseModel.html

\doc\API\source-class-IBaseRequest.html

\doc\API\source-class-IOUtil.html

\doc\API\source-class-MySQL.html

\doc\API\source-class-RequestEvent.html

\doc\API\source-class-RequestEventContribution.html

\doc\API\source-class-RequestEventEntry.html

\doc\API\source-class-RequestHandler.html

\doc\API\source-class-RequestUser.html

\doc\API\source-class-Settings.html

\doc\API\source-class-User.html

\doc\API\tree.html

\doc\API\resources\collapsed.png

\doc\API\resources\combined.js

\doc\API\resources\footer.png

\doc\API\resources\inherit.png

\doc\API\resources\resize.png

\doc\API\resources\sort.png

\doc\API\resources\style.css

\doc\API\resources\tree-cleaner.png

\doc\API\resources\tree-hasnext.png

\doc\API\resources\tree-last.png

\doc\API\resources\tree-vertical.png

6 Installationsanleitung

Schritt 1: Erstellen eines Projektverzeichnis „RFH-SS2014-WebMobileDevelopment-Projektarbeit" im Dokumentroot des Apache-Webservers

Schritt 2: Entpacken des Projektarchivs in das erstellte Projektverzeichnis

Schritt 3: Das korrekte Verzeichnis der Datei .htaccess prüfen. Diese befindet sich im /srcphp Verzeichnis

Schritt 4: Erstellung der SQL Datenbank „pick-it" über das beigefügte SQL Script unter /sql/install.sql

Aufruf des WebService über:
http://localhost/RFH-SS2014-WebMobileDevelopment-Projektarbeit/srcphp/api/events

Aufruf der AJAX-Anwendung über:
http://localhost/RFH-SS2014-WebMobileDevelopment-Projektarbeit/srcphp

Aufruf des Mobile-Client über:
http://localhost/RFH-SS2014-WebMobileDevelopment-Projektarbeit/srcsencha

7 Fazit

Ziel war die Entwicklung einer thematisch individuellen Webanwendung, hier stellte sich jedoch heraus, dass der Umfang des geplanten Projektes in der zur Verfügung stehenden Zeit nicht umgesetzt werden konnte. Daher sind folgende Funktionen zum Abgabezeitpunkt abweichend bzw. nicht implementiert.

Beim Aufbau der API haben wir uns, dazu entschieden eine eigenständige SQL Klasse zu erstellen. Dies hat den Vorteil, dass alle Klassen auf diese Schnittstelle zugreifen können und bei Änderungen diese nur zentral an einer Stelle durchgeführt werden müssen.

Zudem wird die Anzahl der über den Webservice zurückgegebenen Elemente auf 10 limitiert um bei einer großen Anzahl von Datensätzen die übertragene Datenmenge gering zu halten. Folgende Pages können über den Parameter /page<n> abgerufen werden.Die Anzahl weiterer Pages wird im HTTP-Header X-MaxPages mitgegeben. Das Paging wird in der Sencha Anwendung jedoch nicht berücksichtigt, da der Zugriff auf den zusätzlichen HTTP-Header nicht Bestandteil der Vorlesung war. Dort werden nur die ersten 10 Elemente zurückgegeben.

Insbesondere die Datums- und Uhrzeitverarbeitung führte zu langen Fehlersuchen. Unterschiedliche Browsereinstellungen und Sprachregionen erschwerten diese und zwangen zu Datumskonvertierungen. Darüber hinaus bietet das Sencha Framework kein bekanntes Formularelement um die Uhrzeit anzuzeigen.

Das Bearbeiten und Neuanlegen eines Eintrags im mobilen Client wird derzeit nur in den Store, jedoch nicht in die Datenbank geschrieben und ist nach einem Neuladen der Seite verloren. Zudem erfolgt die Definition des Proxys nun im Model anstelle des Stores, um in späteren Entwicklungen aus einem FormPanel Daten speichern zu können.

Die gewünschte dynamische Berechnung auf Basis von Daten auf Anwendungsseite ist aktuell nicht implementiert.

Ebenfalls abweichend von der Aufgabenstellung umfasst das WebProjekt eine dreistufige Hierarchie mit Events > Einträgen > Contributions. Dies führte dazu, dass eine höhere Anzahl voneinander abhängigen Abfragen, Klassen und Controllern/EventHandlern umgesetzt wurden und die Contributions nur im Mobile Client angezeigt werden.

Durch diverse unterschiedliche CSS Implementierungen der gängigen Browser ist das AJAX-Frontend nur im Internet Explorer reibungslos aufrufbar.

Die ursprünglich geplante Userverwaltung wurde aus Zeitgründen nach hinten gestellt, daher werden auch alle Unterelemente grundsätzlich angezeigt. Insbesondere die Verarbeitung von Sessions bzw. Cookiezugriff führte zu diese Entscheidung. Eine simple Zugriffssteuerung für authentifizierte Benutzer kann jedoch bei Bedarf über .htusers/.htpasswd mit Apache Bordmitteln realisiert werden.

Darstellungsverzeichnis