Web-Programmierung

WS 2015/16

- Dokumentation der Semesterarbeit -

Comeet

Thomas Krieger
Matrikel-Nr.: 257510

Marileen Stamer
Matrikel-Nr.: 258234

Torsten Garding
Matrikel-Nr.: 257523

Inhaltsverzeichnis

Inhaltsverzeichnis

1. Einführung

Als Prüfungsleistung entwickeln Sie eine kleine Anwendung (Semesterprojekt). Zu Ihrem Projekt erstellen Sie eine Dokumentation. Das Semesterprojekt bearbeiten Sie in Gruppen von 3 - 4 Personen. Es ist gut, wenn sich Teampartner mit ähnlichen Vorkenntnissen finden. Themen zur Auswahl:

1. Control Panel eines Meß- und Steuerungsgeräts
    - Ziel: Weboberfläche einer existierenden webbasierten Schnittstelle ansprechend gestalten
    - Einlesen und Darstellen von Messwerten
2. Webanwendung für selbstorganisierte Eventorganisation
    - Ziel: eine kleine Anwendung konzeptionieren und prototypisch umsetzen
    - Events anlegen, bearbeiten, Nutzer zuordnen o.ä.

Die beiden Themen werden auf der ersten Präsenz nochmal genauer vorgestellt und erläutert.

1.1 Projektbeschreibung

Als Semesteraufgabe im WS2015/16 soll im Modul Webprogrammierung eine Webanwendung zur selbstorganisierten Eventplanung entwickelt werden. System sowie Inhaltsgestaltung sind für die Studenten relativ frei wählbar. Alternativ konnte man ein Interface zur Steuerung einer Maschine entwickeln. Wir wählten die Eventplanung und nannten unser Portal „Comeet“ (für ‚Come and meet‘).

1.2 Funktionen von Comeet

Comeet stellt eine Anwendung dar, mit welcher registrierte Benutzer eigene Events/Veranstaltungen anlegen können. Dazu können neue Kontakte per E-Mail oder andere registrierte Benutzer direkt eingeladen werden. Die Benutzer können ebenfalls die Events ihrer Kontakte sehen, sowie ihre eigenen Events verwalten.

- Nutzer können sich registrieren
- Registrierte Nutzer können sich einloggen
- Eingeloggte Nutzer können:
    - Events anlegen
    - Events bearbeiten
    - Events löschen
    - Ihr Profil bearbeiten
    - Ihre Kontaktliste anzeigen und verwalten
    - An Events von Freunden teilnehmen
    - Sich aktiv an der Planung beteiligen (Mitbringliste)

1.3 Zusätzliche Ziele

- Per E-Mail eingeladene Kontakte können ihre Eventteilnahme bestätigen oder absagen

1.4 Verwendete Technologien

Im Folgenden führen wir alle verwendeten Technologien auf.

1.4.1 Grunt

Grunt ist ein "Task Runner" Tool, welches auf Node.js basiert. Wir verwenden dieses Tool während der Entwicklung, da es die Arbeit am Projekt sehr erleichtert. Wir verwenden die Module "connect" um einen lokalen Server zu starten, der die statischen Dateien ausliefert. Das "watch" Modul "beobachtet" während der Entwicklung das komplette Verzeichnis und erzeugt einen live reload im Browser. Wird nur am CSS etwas geändert, so wird auch nur die entsprechende CSS Datei neu geladen, ohne dass sich deswegen die ganze Seite neu lädt.

Des Weiteren werden die Module "less" und "compile handlebars" genutzt. Diese dienen zum kompilieren von less zu CSS und handlebars Templates zu HTML. Dadurch können wir im Baukasten-Prinzip entwickeln.

Verwendete Version: 0.1.13

1.4.2 Less / CSS

LESS ist ein Framework, welches die Fähigkeiten von CSS mit dynamischen Funktionen wie Variablen, Mixins, Operationen und Funktionen erweitert. LESS wird für die Entwicklung genutzt und von einem less-Compiler (den wir als Grunt Node Modul eingebunden haben) zu CSS kompiliert.

Verwendete Version: 2.5.3 (LESS) / CSS 3

1.4.3 Bootstrap Grid System

Da wir die Anwendung responsive machen wollen, haben wir das Gridsystem von Bootstrap eingesetzt um schneller starten zu können und nicht erst selbst erzeugen zu müssen. Das Gridsystem bringt Klassen für Spalteneinteilung und Breakpoints mit, wir nutzen ein fluid layout (volle Breite) und das Standard 12 Spalten Grid.

Verwendete Version: 3.3.6

1.4.4 Handlebars / HTML

Handlebars ist eine Template Engine, die recht leicht zu verwenden ist. Wir verwenden weniger die eigentliche Templating Funktionen (Expressions, Helper ...), sondern hauptsächlich nur die Basis- Funktionalität um für die einzelnen Bausteine der Anwendung einzelne Partials anzulegen, die man dann gegenseitig in anderen Templates inkludieren kann.

Verwendete Version: 4.0.5 / HTML 5

1.4.5 JavaScript Clientseitig

Wir versuchen, frontendseitig das meiste mit CSS zu lösen, es ist JavaScript notwendig für:

- Das Layer (Öffnen und Schließen)
- Formularvalidierung bei der Registrierung und beim Anlegen eines neuen Events
- Teilweise zum Setzen von CSS Klassen für CSS Animationen
- Ajax Post requests zum Server

Verwendete Version: ECMAScript 5.1

1.4.6 JavaScript Serverseitig (Express)

Da wir schon mit Node und dem Grunt Connect Modul arbeiten, haben wir beschlossen backendseitig doch kein PHP einzusetzen. Stattdessen nehmen wir einen Node.js Express Server, der die POST Requests abhandelt und mit der Datenbank kommuniziert, und gleichzeitig auch die statischen Inhalte ausliefert.

Diese Entscheidung hat evtl. einen Umbau zur Folge, sodass wir Grunt rauswerfen und auch die less und handlebars Dateien vom Express Server kompilieren lassen. Einen live reload müsste man auch auf diesem Wege hinbekommen.

Verwendete Version: ECMAScript 5.1

Hier ergaben sich während der Entwicklung Änderungen, siehe 3.1.

1.4.7 Pickaday Datepicker

Ein kleiner Datepicker, ohne Abhängigkeiten (wie jQuery o.ä.), der einfach einzubinden und zu stylen ist.

Quelle: https://github.com/dbushell/Pikaday

Wir nutzen ihn um ein Datum für ein Event zu selektieren und die nächsten zwei Kalendermonate in der Seitenleiste anzuzeigen

1.4.8 MySQL

MySQL ist eines der weltweit verbreitetsten relationalen Datenbankverwaltungssysteme. Es ist als Open-Source-Software sowie als kommerzielle Enterprise Version für verschiedene Betriebssysteme verfügbar und bildet die Grundlage für viele dynamische Webauftritte.

Verwendete Version: 5.0.96

1.4.9 PHP

PHP ist eine weit verbreitete Scriptsprache. Die verwendete Template-Engine ist in PHP geschrieben, ebenso wird die Template-Engine aus PHP heraus angesprochen. Zusätzlich kann man mit PHP sehr komfortabel Datenbanken ansprechen. Die Scriptsprache wird direkt auf dem Webserver ausgeführt, dieses vollkommen transparent für die Clients. Es ist später nicht mehr zu erkennen, ob der vorliegende Inhalt statisch oder dynamisch durch PHP erzeugt ist. PHP erfordert keine Installation auf Client-Rechnern.

Verwendete Version: 5.4.42

1.5 Zielumgebung

Als Zielsystem für das Event-Portal kommen jegliche Rechner mit den benötigten Serverdiensten in Frage. Benötigt wird zwingend ein Webserver mit PHP-Unterstützung. Ebenso muss ein MySQL5 Datenbanksystem vorhanden sein.

Die Voraussetzungen sind bewusst so vage gehalten, damit das Endprodukt auf vielen Systemen eingesetzt werden kann. So ist es 'vollkommen' Betriebssystem und architekturunabhängig - vollkommen in dem Sinn, dass auf der Zielarchitektur natürlich ein Webserver mit PHP verfügbar sein muss.

Auf der Clientseite ist lediglich ein Internetbrowser nötig, der JavaScript beherrscht und aktiviert hat. Auch hier gibt es keine Einschränkung an Betriebssystem oder gar Hardwarearchitektur. Aus Zeitgründen ist das Design der Webseite nur an Chrome bzw. Iron angepasst. Andere Browser wie beispielsweise Microsoft Internet Explorer, Mozilla Firefox, Opera, Edge oder andere werden explizit nicht unterstützt – sie werden sicherlich in den meisten Fällen funktionieren, es kann aber zu Darstellungs- oder gar Funktionsfehlern kommen.

Das Layout der Seite basiert komplett auf CSS. Es werden weder browserspezifische 'Hacks' noch 'Weichen' eingesetzt

Als Scriptsprache auf dem Server kommt lediglich PHP zum Einsatz. PHP ist bei den meisten Hosting-Angeboten verfügbar, somit lässt sich das Endprodukt (hoffentlich) auf vielen dieser Angebote nutzen.

Als Scriptsprache für die Clients ist JavaScript in Verwendung. Die Auswahl hier ist nicht groß, es bleibt nur JavaScript übrig, sobald man mehrere Browser unterstützen möchte.

1.6 Entwicklungssysteme

Da unsere Gruppe verschiedene Hard- und Software einsetzte, mussten grundlegend erst einmal Wege gefunden werden, gemeinsame Arbeitsgrundlagen zu schaffen. Um weitestgehend gleiche Arbeitsumgebungen zu schaffen, wurde folgende Software eingesetzt:

- Node.JS für Bootstrap und Grunt
- Brackets und IntelliJ als Entwicklungsumgebung
- Notepad++ als Texteditor
- Chrome bzw. Iron als Web-Client zum Testen
- Git als Versionskontrolle

1.6.1 Entwicklungssystem von Thomas Krieger

Thomas setzte Windows 10 Professional als Betriebssystem ein. Zusätzlich zu dem oben genannten Softwarepacket nutzt er Eclipse als ergänzende Entwicklungsumgebung. Als Web-Client wurde SWIron verwendet.

1.6.2 Entwicklungssystem von Marileen Stamer

Marileen setzte MacOS 10.9.5 als Betriebssystem ein. Als Entwicklungsumgebung und Zugriff auf Git benutzte sie IntelliJ, als Web-Client kam Chrome zum Einsatz.

1.6.3 Entwicklungssystem von Torsten Garding

Torsten setzte anfangs Linux als Betriebssystem ein, stieg dann aber auf Windows 10 um. Zum Einsatz kamen Node.JS, Grunt, Git und Brackets, sowie SWIron als Web-Client.

2. Entwicklung

Zu Beginn haben wir uns zusammengesetzt und den Zweck und die Zielgruppe für unsere Anwendung bestimmt. Als Ergebnis kam dabei heraus, dass wir unsere Webseite zwar jung und aufgeschlossen darstellen wollen, als Zielgruppe aber grundsätzlich jeder angesprochen werden sollte. Daraus hat sich dann einerseits das Design ergeben, andererseits wurden aber Worte wie z.B. „Freunde“ durch neutraler klingende „Kontakte“ ersetzt. Während man privat sicher „Freunde“ einladen würde, würde man in einer Firma eher von Kollegen oder Mitarbeitern sprechen. Um eine Differenzierung zu vermeiden, entschieden wir uns an dieser Stelle für „Kontakte“. Dies bot die Möglichkeit, diese Webseite nicht nur privat, sondern auch geschäftlich nutzen zu können.

Anschließend haben wir uns über die Funktionen und den Umfang geeinigt. Da der Zeitrahmen recht begrenzt war, versuchten wir uns auf das notwendige zu beschränken. Auf Dinge wie einen integrierten Chat oder Nachrichtendienst wurde daher verzichtet, aber es steht die Idee Raum, die Webseite im Anschluss nach dem Semester weiter zu entwickeln. Im weiteren Verlauf entstanden der Name und das Logo. „Comeet“ ist ein kleines Wortspiel und setzt sich aus den Wörtern „Come and meet“ zusammen, was gleichzeitig als Slogan für unsere Eventplanungs-Webseite verwendet wird. Das Logo mit dem Stern in der Mitte und dem umkreisenden Pfeil symbolisiert dabei einmal die Gemeinschaft, ohne die ein Event ja gar nicht möglich ist. Andererseits stellt es auch einen Kometen dar, der einen Stern umkreist. Die ersten Entwürfe wurden auf Papier angefertigt. Diese wurden später auf den Computer übertragen und dabei verfeinert. Als nächstes besprachen wir, mit welchen Technologien wir arbeiten wollen, und entschieden uns für die unter 1.4 genannten. Das führte dazu, dass wir diese auf jedem genutzten Computer lauffähig machen mussten, was sich bei drei unterschiedlichen Systemen anfangs als nicht ganz so einfach herausstellte. Nachdem wir alle über eine annähernd gleiche Softwarekonfiguration verfügten begannen wir, die ersten Bausteine für die einzelnen Seiten der Plattform zu erstellen. Wir erstellten zu den .hbs Dateien die entsprechenden .less Dateien unter Zuhilfenahme des Grid Systems von Bootstrap. Im Groben orientierten wir uns an den erstellten Entwürfen, stellten aber schnell fest, dass sie auf der fertigen Seite nicht wie geplant wirkten und änderten sie ab. Anschließend bauten wir die einzelnen .hbs Bausteine zu den Seiten unserer Website zusammen und erstellten die Navigation. Hinterher beschäftigten wir uns mit der Funktionalität der Formulare, Buttons, etc. - allerdings erstmal nur so weit, dass wir die Funktionalität optisch darstellen konnten. Bis zu diesem Zeitpunkt hatten wir auf eine nötige

Datenbank verzichtet und uns auf die Elemente beschränkt, welche wir für die Präsentation für nötig hielten. Die letzten zwei Wochen (je Mo und Do) vor der zweiten Präsenz haben wir mit eingehenden Tests der Seite und letzten Anpassungen für die Präsentation verbracht, sodass wir einen lauffähigen Dummy präsentieren konnten. Mittels Grunt konnten wir während der Präsentation praktisch live einen Dummy insoweit vorstellen, dass man durch Klicks die später zu erwartenden Reaktionen simulieren konnte.

Unmittelbar nach der Präsentation begannen wir mit der Implementierung von JavaScript und PHP, welche die simulierten Funktionen durch echte ersetzen sollten. Es wurde eine MySQL Datenbank aufgesetzt, die notwendigen Tabellen angelegt und mittels erster PHP Testcodes probiert, diese Tabellen auszulesen – in einem weiteren Schritt wurden http-PostRequests mit JavaScript erzeugt, um die Interaktion zwischen JavaScript und PHP auszuprobieren. Anschließend wurden Stück für Stück die simulierten Funktionen ersetzt.

2.1 Verzeichnisstruktur

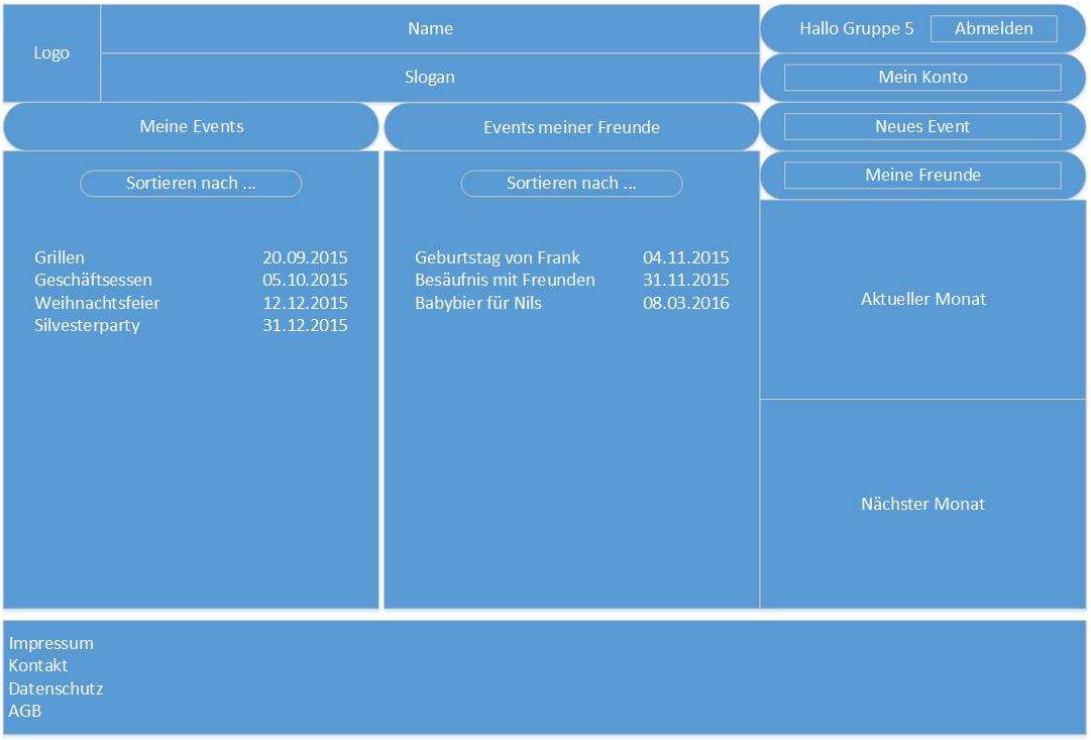
Zuerst der Aufbau unserer Entwicklungsumgebungen

- **Comeet (Projektverzeichnis)**
  - .**tmp** (Output Verzeichnis für deploybare Webanwendung)
  - **dev** (Arbeitsverzeichnis)
    - **assets** (Bilder, favicon usw.)
    - **js**
    - **partials** (handlebars Bausteine)
    - **php**
    - **styles** (less Dateien)
  - Gruntfile.js
  - Package.json

Nun der Aufbau auf dem Live-System

2.2 Design

Im folgenden Bild ist das Wireframe, welches ganz zu Beginn als Grundidee für den Aufbau des Portals gefertigt wurde. Seit Erstellung sind wir von dem Layout nicht abgewichen und haben auf dieser Grundlage die komplette Seitenstruktur aufgebaut.



2.3 Datenbank

Benutzerdaten sowie der Inhalt des Blogs werden in einer Datenbank namens HTO01FLQZCHT_2 gespeichert. In der Datenbankdesignphase wurden fünf Tabellen angelegt. So existieren je eine Tabelle für

- Benutzerdaten (Users)
- Kontakte (Contacts)
- Events (Events)
- Mitbringsel usw. (Items)
- Teilnehmer (Attendees)

Die Tabellen werden nun ein wenig näher untersucht.

2.3.1 Attendees

```
--

-- Tabellenstruktur für Tabelle `Attendees`

--

CREATE TABLE IF NOT EXISTS `Attendees` (

  `Event_ID` int(11) NOT NULL,

  `User_ID` int(11) NOT NULL,

  `status` varchar(50) default NULL,

  KEY `Event_ID` (`Event_ID`),

  KEY `User_ID` (`User_ID`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

In dieser Tabelle werden die Teilnehmer für die jeweiligen Events hinterlegt.

2.3.2 Contacts

```
--

-- Tabellenstruktur für Tabelle `Contacts`

--

CREATE TABLE IF NOT EXISTS `Contacts` (

  `User_ID` int(11) NOT NULL,

  `Contact_ID` int(11) NOT NULL,

  KEY `User_ID` (`User_ID`),

  KEY `Contact_ID` (`Contact_ID`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Hier werden Kontakte miteinander verknüpft – es beinhaltet als erstes den User und als zweites den zugewiesenen Kontakt.

2.3.3 Events

```
--

-- Tabellenstruktur für Tabelle `Events`

--

CREATE TABLE IF NOT EXISTS `Events` (

  `Event_ID` int(11) NOT NULL auto_increment,

  `Title` varchar(255) NOT NULL,

  `Description` varchar(5000) NOT NULL,

  `Street` varchar(255) NOT NULL,

  `Nr` varchar(30) NOT NULL,

  `Postcode` varchar(30) NOT NULL,

  `City` varchar(255) NOT NULL,

  `CalendarDate` varchar(100) NOT NULL,

  `User_ID` int(11) NOT NULL,

  `MapLink` varchar(500) default NULL,

  `TimeInfo` varchar(100) default NULL,

  `mm` int(11) NOT NULL default '-1',

  `yyyy` int(11) NOT NULL default '0',

  `dd` int(11) NOT NULL default '0',

  PRIMARY KEY  (`Event_ID`),

  KEY `User_ID` (`User_ID`)

) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=83 ;
```

Dies ist die Event Tabelle. In ihr werden alle relevanten Daten abgelegt, die für das Event wichtig sind, wie „wann“, „wo“ usw. Ebenso wird hier auch die UserID mitgespeichert um festzulegen, wem das Event gehört bzw. es angelegt hat.

2.3.4 Items

```
--

-- Tabellenstruktur für Tabelle `Items`

--

CREATE TABLE IF NOT EXISTS `Items` (

  `Item_ID` int(11) NOT NULL auto_increment,

  `Event_ID` int(11) NOT NULL,

  `User_ID` int(11) NOT NULL,

  `Name` varchar(255) NOT NULL,

  PRIMARY KEY  (`Item_ID`),

  KEY `Event_ID` (`Event_ID`),

  KEY `User_ID` (`User_ID`)

) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=510 ;
```

Die Items Tabelle beinhaltet Dinge, um die sich ein bestimmter Teilnehmer kümmert. Jedem Item wird eine UserID und eine EventID zugewiesen. Ist ein Item noch keinem User zugewiesen, so bekommt die UserID eine „0“.

2.3.5 Users

```
--

-- Tabellenstruktur für Tabelle `Users`

--

CREATE TABLE IF NOT EXISTS `Users` (

  `User_ID` int(11) NOT NULL auto_increment,

  `Firstname` varchar(255) NOT NULL,

  `Lastname` varchar(255) NOT NULL,

  `Email` varchar(255) NOT NULL,

  `Birthdate` varchar(50) NOT NULL,

  `Picture` varchar(255) default NULL,

  `Password` varchar(255) NOT NULL,

  `Username` varchar(255) NOT NULL,

  `Status` varchar(50) default NULL,

  UNIQUE KEY `User_ID` (`User_ID`)

) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=81 ;
```

Die User-Tabelle beinhaltet alle User-relevanten Daten, die bei der Registrierung angelegt und über das User Profil geändert werden können.

2.4 Funktion

Als zentrale Steuerdatei wird die index.html im Basisverzeichnis des Webservers verwendet. Mit HTML und CSS wird die Webseite generell konstruiert, mit JavaScript werden die Interaktionen zwischen User und Webseite durchgeführt. JavaScript kommuniziert anschließend mit PHP-Seiten, welche serverseitig die Verbindung zur MYSQL Datenbank herstellen und notwendige Datensätze aufbereiten bzw. bereitstellen. Die Ergebnisse werden anschließend mit JavaScript und HTML visualisiert.

2.5 Background-Engine

Wir haben versucht die Anwendung so einfach wie möglich zu gestalten was bedeutet, dass wir Aspekte wie Sicherheit und Angriffsschutz völlig außer Acht gelassen haben. Da wir ausschließlich mit HTML, JavaScript und PHP arbeiten, sind keine speziellen Anforderungen am Background vonnöten. Es werden PostRequests im JavaScript erzeugt, welche dann mittels PHP bearbeitet und mit einem HTTP Response beantwortet werden.

3. Implementierung

Wir haben zunächst das HTML Grundgerüst entwickelt, auf Basis des Gridsystems von Bootstrap. Dabei haben wir die einzelnen Seitenbausteine (Komponenten) in Handlebars Partials angelegt. Diese Komponenten sind im Markup gekennzeichnet durch data-component="name-der-Komponente".

Das Styling haben wir per LESS nach demselben Prinzip erzeugt, indem es pro Komponente eine .less Datei gibt, die genauso heißt wie die im HTML bezeichnete Komponente. So konnte jeder gut an einzelnen Teilen arbeiten und die Wiederauffindbarkeit von Styles zu Markup ist optimal gegeben.

Dann haben wir nach und nach die JavaScript Funktionalitäten für das Frontend implementiert, sowie die Kommunikation mit dem Backend.

Auf der Startseite zeigen wir auf der Desktop Version einen Slider an, der über das Produkt informiert (rein mit CSS gebaut). In der Mobilen Variante fällt der Slider weg.

Im Ausgeloggten Zustand wird der Header zum Einloggen oder registrieren angezeigt.

Im eingeloggten Zustand wird der Header mit dem Benutzerprofil angezeigt.

Im eingeloggten Zustand (nach dem Registrieren) gelangt man über die Seitenleiste zu den Unterseiten „neues Event“ und der Eventübersicht „meine Events“. Es wird außerdem die Freundesliste angezeigt, diese hat eine Aufklapp-Funktion.

Und zusätzlich die nächsten zwei Kalendermonate (über den Pikaday Datepicker Kalender) mit den Markierungen an welchen Tagen Events stattfinden, an denen man bereits teilnimmt.

Auf der Seite „neues Event“ befindet sich ein Formular über das man die nötigen Eingaben zu einem Event machen muss. Das Formular wird erst per JavaScript validiert und dann abgeschickt an eine PHP Datei.

Als besondere Funktionen, sind hier eine Google Maps Karte zur Anzeige der Adresse eingebaut, sowie eine Listenfunktion über Textfelder Dinge zu benennen, die zu dem Event von Teilnehmern mitgebracht werden können.

Events die bereits gespeichert sind, kann man über „meine Events“ einsehen. Auf der linken Seite stehen die eigenen Events (bzw. in der mobilen Variante oben), diese kann man anklicken um die Detailinfos zu sehen und um das Event zu bearbeiten oder zu löschen.

Auf der rechten Seite stehen die Events von Kontakten aus der Kontaktliste, klickt man diese an, so erhält man die Detailinfos und die Möglichkeit teilzunehmen und dabei etwas von der Liste des Erstellers mitzubringen.

Auf der Detailansicht wird auch die Google Maps Karte verwendet.

Kontaktliste:

Um Events von Kontakten zu sehen muss man die Kontaktliste bearbeiten (rechts über die Seitenleiste). Auf der Bearbeiten Seite kann man über eine Suche die User aus der Datenbank finden und zu seiner eigenen Kontaktliste hinzufügen und auch wieder entfernen.

Mein Konto:

Über die „Mein Konto“ Funktion oben rechts kann man seine eigenen Profilinformationen bearbeiten und so Benutzername, Email Adresse und Passwort ändern. Es ist auch vorgesehen, das Profilbild zu ändern, das ist aus zeitlichen Gründen aktuell nicht implementiert.

Solange der User online ist, wird eine PHP Session Variable bereitgehalten über die die User spezifischen Daten aus der DB abgefragt werden. Ist ein User länger als 10 Minuten inaktiv wird er über einen JavaScript Timeout abgemeldet. Ein User kann sich aber auch aktiv ausloggen über den abmelden Link oben rechts.

3.1 Abweichungen vom Entwurf

Zu 1.4.6:

Statt der Backend-Logik mit JavaScript haben wir uns dann doch für PHP entschieden, da Thomas hier schon Vorkenntnisse mitbrachte und der Server auf dem die Anwendung läuft PHP unterstützt, jedoch kein Node.js.

Wir setzten PHP ein, indem wir Anfragen vom JavaScript an PHP Dateien senden, welche dann Daten aus der DB holen und sie zurückliefern.

3.2 Programmlogik

3.3 Einschränkungen

Im Moment kann man das Profilbild noch nicht ändern. Es war zwar keine Forderung seitens der Aufgabenstellung, wir hätten es aber gerne mit integriert. Voraussetzung wäre ein FTP Upload gewesen mit Rechtevergabe bzw. Beschränkung für ein entsprechendes Verzeichnis. Diese Implementierung hätte uns zusätzlich mehr Zeit gekostet die am Ende einfach nicht mehr zur Verfügung stand.

Eine weitere Einschränkung hatten wir seitens des Providers für den Webspace. T-Online stellt mit dem Webspace zwar auch eine MySQL Datenbank zur Verfügung, aber beschränkt die Engine ausschließlich auf die Nutzung von MyISAM bzw. verbietet die Umstellung auf InnoDB. Damit waren Funktionalitäten wie Foreign-Keys für die Verknüpfung der Tabellen miteinander nicht möglich, was eine andere Art der Datenabfragen bedurfte.

Ebenso hatten wir angedacht, dass man Kontakte zu seinen eigenen Events einladen kann, jedoch reicht die aktuelle Funktionalität nur insoweit, dass man selbst die Events seiner Kontakte sehen und an diesen teilnehmen kann.

3.4 Quellcode

Der Quellcode befindet sich mit in dem Archiv, deshalb wird er hier nicht noch mal zusätzlich aufgeführt.

3.5 Installation

- Voraussetzung: Node.js und Grunt müssen installiert sein
- Datenbank muss vorhanden sein (siehe Exportdatei) -> in der config.php kann man die Einstellungen zur DB machen
- Es muss ein PHP Server laufen
- Im Verzeichnis wo das Gruntfile liegt, einfach "npm install" ausführen
- dann "grunt" ausführen (hbs und less werden kompiliert) (Es wird ein lokaler Server gestartet, das ist nur für Entwicklung)
- Hat man den grunt einmal gestartet und die Dateien kompiliert, dann kann man "grunt copy" ausführen
- Dadurch liegen nun ALLE benötigten Dateien im .tmp Verzeichnis, so wie sie auf den PHP Server geladen und dort ausgeführt werden können

4. Schlusswort

Die Arbeit an diesem Projekt hat uns dreien sehr viel Spaß gemacht. Es gab viele kopfzerbrechende Momente, aber für alle aufkommenden Probleme haben wir früher oder später eine Lösung gefunden.

Von Anfang an waren wir mit voller Begeisterung dabei und haben schnell entdeckt, wie viel Potential so ein Portal bieten kann – wir haben das Portal, so wie es im aktuellen Zustand zum Zeitpunkt der Abgabe mit diesem Dokument ist, auf einem Webspace von T-Online hochgeladen, sodass es öffentlich zur Verfügung steht. Man kann sie unter der Domain http://www.comeet.eu erreichen.

Wir haben noch viele Ideen und Funktionen, die man in dieser Webseite einbauen könnte, jedoch reichte dafür die vorgegebene Zeit nicht aus. Sollte es unsere Zeit zulassen, so haben wir die Motivation, das Portal mit der Zeit um weitere Funktionen zu ergänzen.

- Folgt - ;-)