Eine Entwicklung im Rahmen des Praktikum Datenbanken und Verteilte Systeme

Jan-Hendrik Briese	**6523408**
Maximilian Manfred Knapperzbusch	**6535090**
Anne Speerschneider	**6524692**
Julian Tiemann	**6542232**

INHALT

EMERGENCY
DIE IDEE

In der heutigen Zeit ist alles und jeder Vernetzt. Trotzdem dauert es mit unter zu lange, bis Krankenwagen oder Feuerwehr bei einem Unfall eintreffen oder aber es ist kein Handwerker greifbar.

Hier wollen wir mit eMergency die Lücke füllen. Von Unfall über Notfall und Feuer bis hin zu Handwerksleistungen. Mit eMergency kann auf einfache Weise nach Hilfe gesucht werden.

DER TITEL, DAS LOGO

Wie sind wir auf die Idee für Titel und Logo gekommen. Ganz einfach: Unsere erste Intension war eine Hilfeleistungs-App bei Notfällen (engl. Emergency) wie z.B. einem Herzinfarkt. Daher die EKG Linie im Logo. Das e in eMergency steht für die Verbindung zur Moderne, dem eBusiness.

DER WORKFLOW

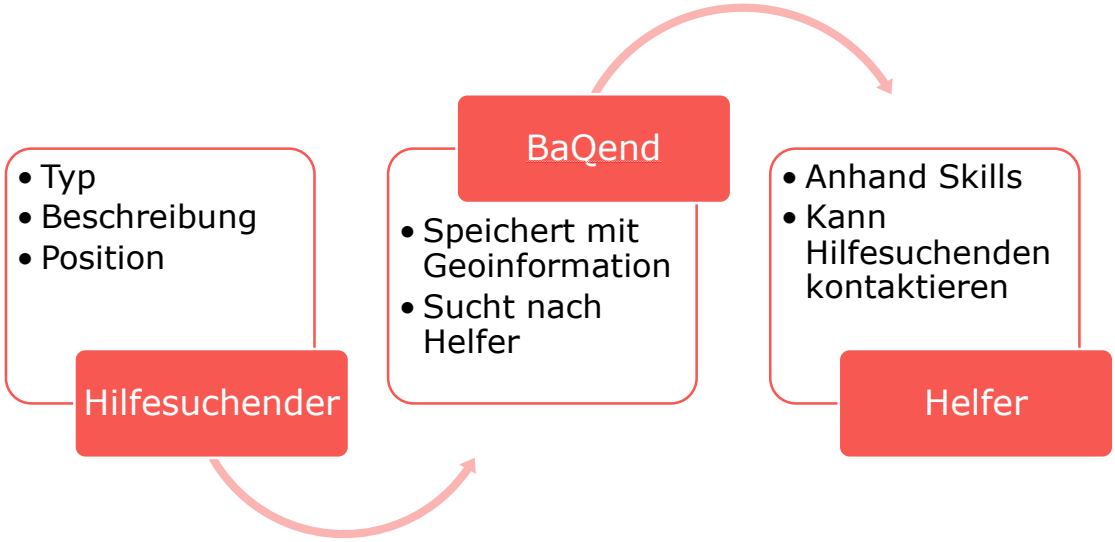

Abbildung 1: Der erste Workflow Entwurf

DIE TECHNOLOGIE

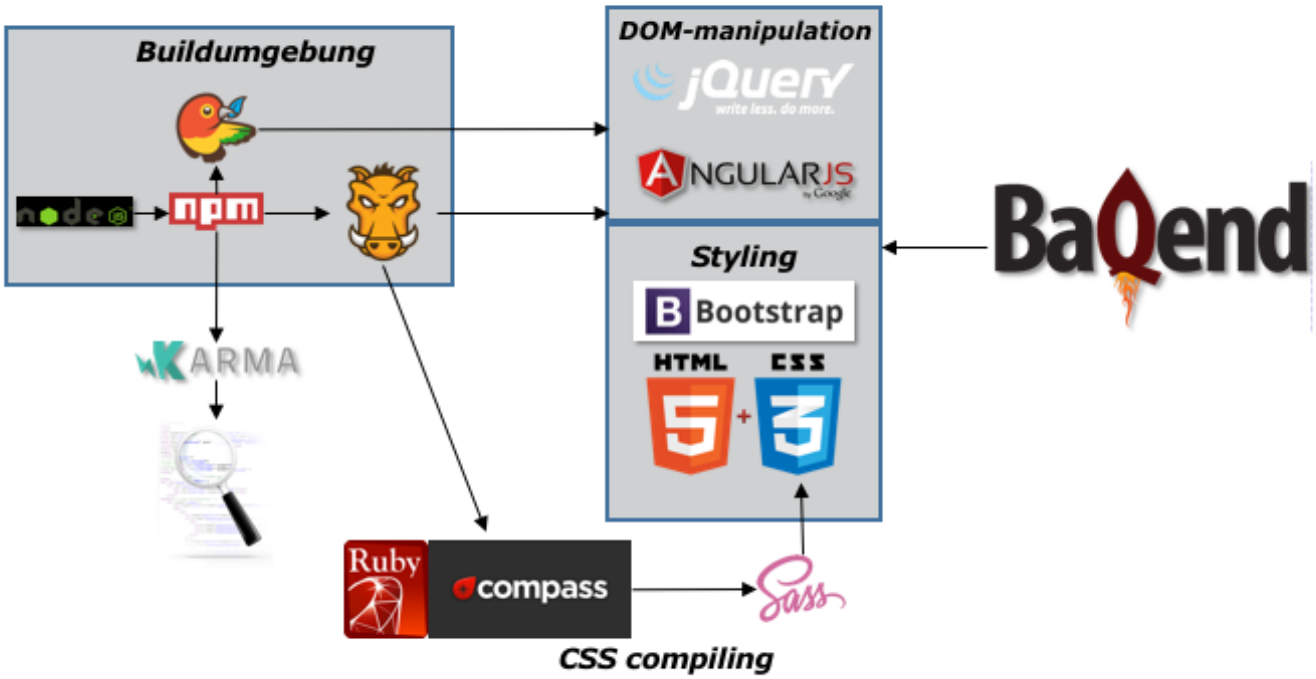

Abbildung 2: Eingesetzte Technologien

Für das Styling der Seite haben wir uns dem CSS-Framework Bootstrap bedient. Dieses stellt Klassen bereit, die im HTML die Möglichkeit eines erweiterbaren, responsiven Grundlayouts bieten. Dies ist gerade für unsere Anwendung wichtig, da diese besonders auf Mobilgeräten einwandfrei dargestellt werden soll.

Daneben haben wir CSS3 zum Styling von DOM-Elementen und SASS im Einsatz. SASS bietet neben einer übersichtlicheren hierarchischen Darstellung von CSS auch die Möglichkeit, Variablen zu verwenden. Zum Beispiel kann eine Standardfarbe und Schriftgröße definiert und in einer Variablen vorgehalten werden, so dass diese nicht an jedem Punkt, wo diese Werte verwendet werden sollen, neu geschrieben werden müssen, sondern lediglich die Variable aufgerufen wird.

Für die DOM-Manipulation benutzen wir Javascript sowie JQuery und AngularJS. JQuery ist eine Bibliothek für Javascript und stellt grunsätzliche Manipulationsfunktionen wie z.B. Animationen bereit. AngularJS als MVC-Framework für Javascript hilft dabei, HTML zu

dynamisieren und bietet bekannte MVC-Vorteile, wie die Aufteilung in View, Controller und Model, wobei das Model durch Services realisiert ist.

Unsere Buildumgebung beruht im Grunde auf Node.js. Dies ist ein Build-Tool inklusive einem Paketmanager npm, welches u.a. folgendes bereitstellt:

- *JavaScript Tests*
- *JS-Minifier (Leerraum entfernen und Datei verkleinern)*
- *uglifier (macht Dateien unleserlich für Externe Diebe)*
- *css minifier*
- *bower*
- *grunt*

Hierüber lässt sich ein lokaler Server starten, um die Seite lokal auf dem eigenen Rechner entwickeln zu können. Über Node.js können Dateien überwacht werden z.B. können SASS Dateien automatisiert in CSS Dateien kompiliert und im Hintergrund nach Javascript-Fehlern gesucht werden.

Grunt ist ein Tasker, der verschiedene definierte und zusammengestellte Aufgaben (Tasks) zu einer Aufgabe zusammenfasst und diese durchgeht. Zum Beispiel der Befehl “grunt serve” führt die Überprüfung der JS- und CSS Dateien auf Fehler aus, kompiliert SASS in temporäre CSS Dateien, überprüft ob alle Abhängigkeiten (bower oder npm Pakete) installiert und auf dem neusten Stand sind und stellt einen lokalen Server bereit, auf den zugegriffen werden kann. Der Befehl “grunt build” kompilliert SASS, verkleinert Bilder, komprimiert und merged CSS- und JS Dateien und kopiert alles in einen separaten Ordner, der anschließend auf jedem Server ausgeführt werden kann.

DIE UMSETZUNG

In unserem Projekt orientierten wir uns an modernen Planungs- und Koordinations-Werkzeugen wie Scrumboards, GitHub, Slack und anderen.

Abbildung 3: Werkzeuge zur Koordination

DIE PLANUNG

Zur Organisation der einzelnen Aufgabenpakete haben wir uns für eine agile Entwicklung entschieden. Die einzelnen Anforderungen wurden als Userstories formuliert und in Teilpakete aufgeteilt. Diese Teilpakete konnten von jedem Teammitglied aus dem Produktbacklog gezogen und bearbeitet werden. Eine Übersicht über die gerade aktiv bearbeiteten, erledigten und noch offenen Items lieferte uns Trello.

Trello bietet die Möglichkeit ein digitales Scrumboard für das gesamte Projektteam zentral zu verwalten.

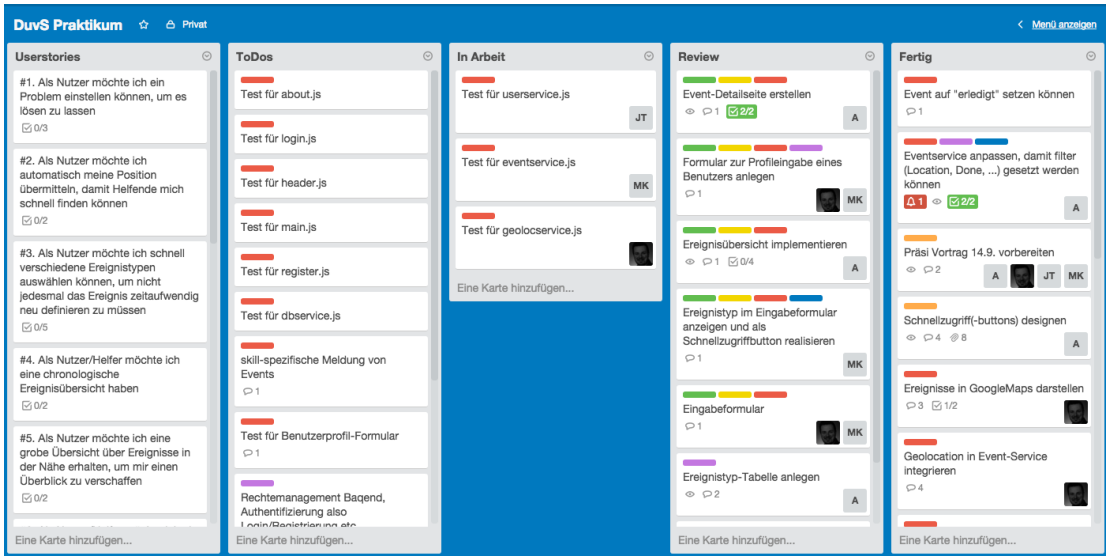

Abbildung 4: Scrumboard auf Trello.com

DAS LAYOUT

Zu Beginn der Entwicklung stand die Klärung des Seitenlayouts. Da unsere Anwendung besonders für die mobile Verwendung ausgelegt sein würde, haben wir darauf geachtet, ein Layout zu entwerfen, welches sowohl auf Mobilgeräten als auch auf Desktop-Systemen funktioniert.

Im Fokus stand der große Inhaltsbereich, welcher alle aktuellen Ereignisse auf einer Google Map sowie in tabellarischer Form übersichtlich darstellt. Darüber wird konsistent auf allen Seiten eine Navigation angezeigt, über die der Nutzer zum einen immer wieder auf die Hauptseite zurückwechseln (Escape-Pattern: Klick auf Anwendungslogo) und zum anderen zu seinem Benutzerprofil gelangen kann.

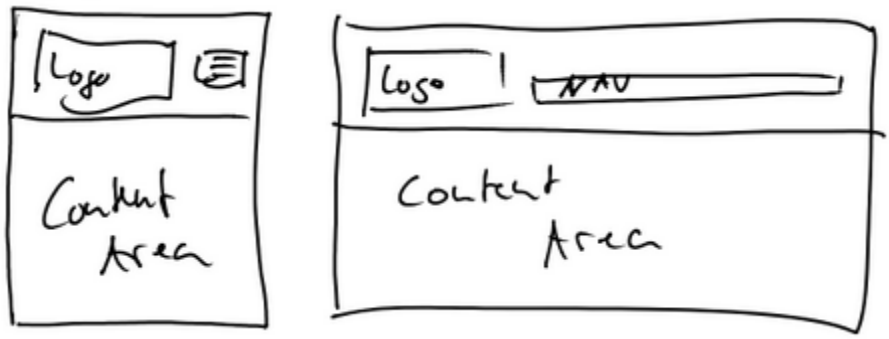

Abbildung 5: Erster Layout Entwurf

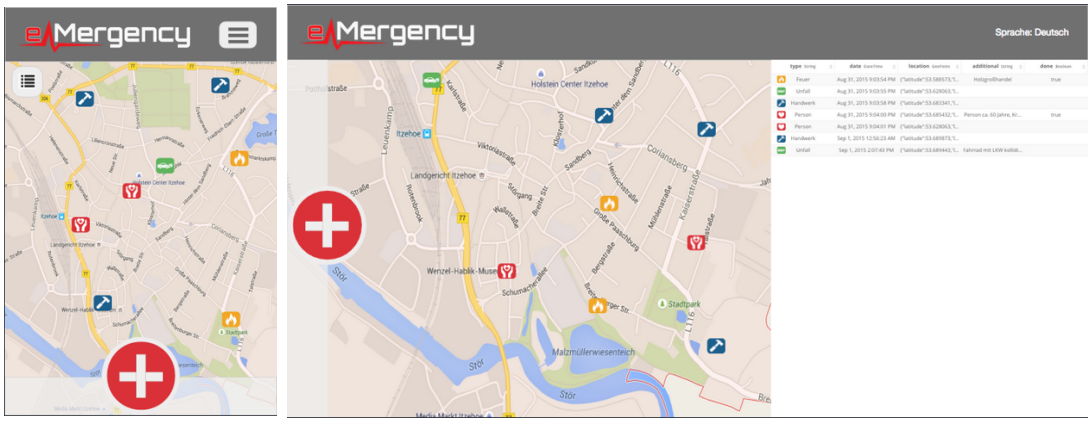

Abbildung 6: Entwurf für Mobilgeräte (links) und Desktop (rechts)

Darüber hinaus sollte es für Hilfesuchende einfach und schnell möglich sein, ein neues Ereignis zu erstellen. Dieser Anforderung sind wir nachgekommen, in dem das Formular zum Erstellen eines neuen Ereignisses direkt auf der Hauptseite über ein großes Plus-Zeichen aufgerufen werden kann. Auf diesem kann er verschiedene Fähigkeiten angeben z.B. Rettungsschwimmer, Erste-Hilfe-Kurs absolviert etc., welche bei der Benachrichtigung über ein Ereignis in seiner Nähe berücksichtigt werden. Somit erhält ein Helfer immer auch nur solche Hilfegesuche, auf die er qualifiziert reagieren kann. Ansonsten findet keine Vorfilterung der Ereignismeldungen statt.

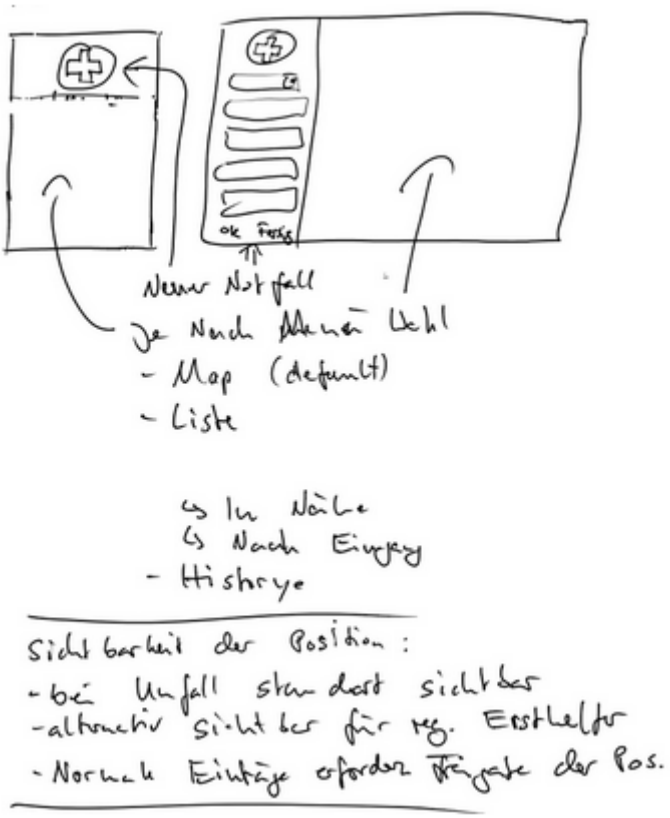

Abbildung 7: Entwurf des Ereignishandlings

Abbildung 8: Entwurf für die Ereignisgenerierung, Mobil (links) und Desktop (rechts)

Neben der Möglichkeit, ein detailliertes Ereignis zu erstellen, gibt es die Option der Schnellbuttons, welche ein Ereignis mit einem Klick einstellen. Diese Buttons sind klar im oberen Teil des Ereignis-Erstellformulars dargestellt und erklären dem Benutzer durch ihre jeweilige Farb- und Iconwahl auf einen Blick, welche Art von Ereignis durch welchen Button eingestellt wird. Zu Beginn haben wir uns auf vier wesentliche Ereignistypen beschränkt: Personenschaden, Feuer, Unfall/Panne, Handwerk. Diese könnten bei Bedarf jedoch beliebig erweitert bzw. differenziert werden.

Außerdem haben wir Wert darauf gelegt, dass auch noch nicht registrierte Nutzer einen Hilferuf absetzen können. Das Plus-Zeichen wird auf der Seite für Gäste immer dargestellt und führt zum Ereignis-Erstellformular.

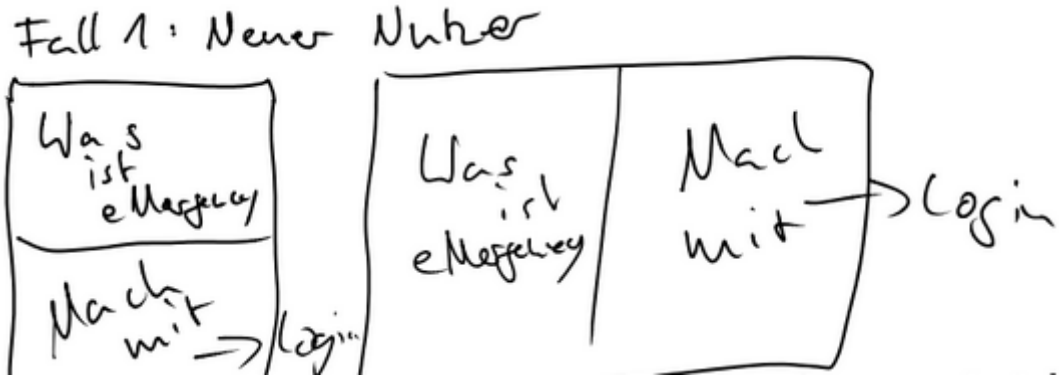

Abbildung 9: Entwurf für nicht registrierte Nutzer

Abbildung 10: Entwurf Startseite, Mobil (links) und Dekstop (rechts)

Zur Vermittlung des Bedienkonzeptes wurde ein Mockup mit Atomic entworfen. Dieses Mockup-Tool bietet im Vergleich zu anderen Mockup-Tools eine realistische Oberflächengestaltung und -darstellung sowie die Möglichkeit der realistischen Aktionsgestaltung für Übergänge zwischen den einzelnen Seiten.

DER AUFBAU

Kern ist eine index.html, die den Seitenaufbau vorgibt, also Header, Footer, benutzte Scripte etc. Der Content-Bereich wird über AngularJS und den Angular-eigenen Router gesteuert. Je nach Route (also URL) wird nach dem MVC-Muster ein View eingebunden, der von dem passenden Controller mit Daten versorgt wird. Pro Seite gibt es einen Controller. Spezielle Bereiche im HTML können auch einen eigenen Controller haben, wie zB der Header, der vor allem für das User-Profil zuständig ist und auf allen Seiten gebraucht wird.

Die Daten beziehen die Controller über Services, die die Schnittstelle zwischen den Controllern (also dem Frontend) und der Datenbank (also dem Backend) sind. Sie sorgen dafür, dass die Controller mit dem Backend kommunizieren können. Richtig aufbereitet werden die Daten schon im Backend (in dem Fall Baqend). Die Ajax-Anfragen an den Server laufen alle asynchron ab und basieren auf Javascript promises, die von den Services auch wieder als promise an die Controller zurückgegeben werden. So wird sichergestellt, dass den Variablen auch erst dann der erwartete Wert zugewiesen wird, wenn die Abfrage fertig ist.

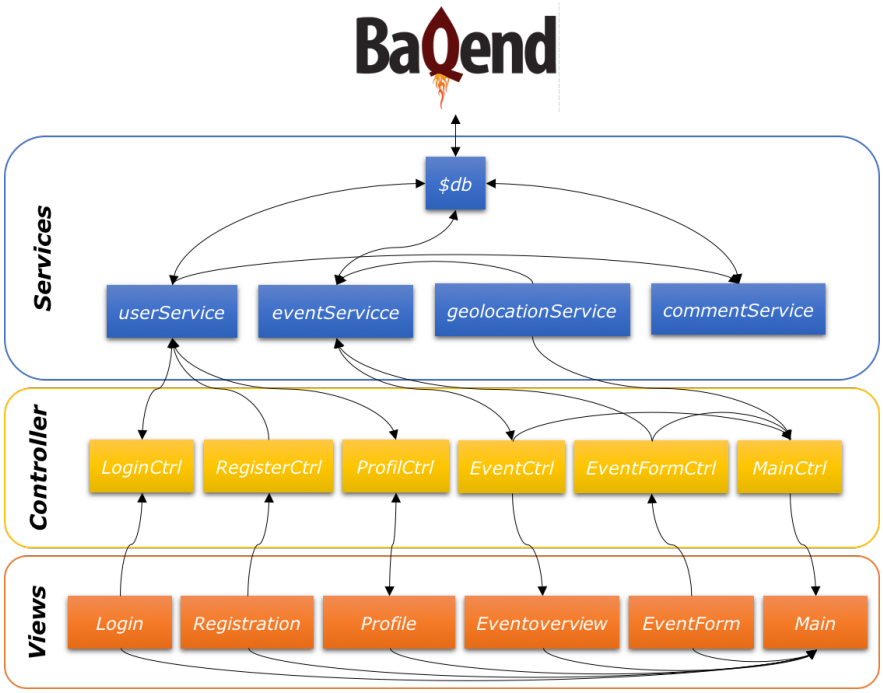

Abbildung 11: Komponenten der Anwendung

DER CONTENT

Wie schon Eingangs erwähnt, setzen wir Bootstrap als Webframework für das Frontend ein. So konnte man nach einer kurzen Einarbeitungsphase recht schnell die einzelnen Formulare aufbauen und gestalten. In einigen Fällen haben wir uns die Elemente angepasst, so dass sie unseren Wünschen entsprachen. Im Fall der „QuickEvent-Buttons" zum Beispiel haben wir uns die Symbole in Photoshop erstellt und über den Button gelegt.

Was sich als etwas schwieriger entpuppte, war das Anordnen der Buttons und Eingabefelder. Bootstrap ist in der Lage die Seite in ein Gridlayout zu unterteilen. Bei den „QuickEvent-Buttons ging dies auch leicht von statten, da alle Buttons gleich groß sind und man so die 12 Spalten des Layouts gut einteilen konnte. Die Probleme traten dann bei der Anordnung der „Eventbeschreibung" auf, in dieser sollen Nutzer in der Lage sein, genaue Angaben zu einem Notfall zu machen. Da aber in diesem Fall nicht alle Elemente die gleiche Größe haben, war die Einteilung nicht so leicht, da manche Eingabefelder durch das Layout verzerrt wurden, oder die Anordnung der Elemente nicht passte, so waren der „Abbruch" und „Sende" – Button zu weit auseinander.

UNSER FAZIT

Kommen wir zu aller erst mal zu unserem größten Problem. Den Tests. Wie sich herausstellte, ist das testen auf promisses extrem kompliziert, weshalb wir die Tests erst einmal mit einer niedrigen Priorität hintenangestellt haben.

Im großen und Ganzen hatten wir sonst keine großen Probleme. Die Organisation und Kommunikation mit Hilfe von Tools wie Trello und Slack funktionierte einwandfrei. Jeder im Team wusste zu jeder Zeit wo wir stehen.

Das Projekt war von Begin an lauffähig, anfangs nur als simple HTML-Seite, und wurde Stück für Stück mit Funktionalitäten gefüllt. Diese Art des Vorgehens erwies sich als sehr praktisch da unterschiedliche Teammitglieder an unterschiedlichen Funktionalitäten arbeiten konnten.

Zu guter Letzt möchten wir uns noch beim Baqend-Team bedanken. Probleme wurden umgehend geklärt und Bugfixes schnell eingestellt. Der Support war vorbildlich.