Praktikumsbericht

01.10.18 - 31.03.19
Independent Studies bei Prof. Diehl
Universität Trier, Lehrstuhl für Softwaretechnik

Benedikt Lüken-Winkels
Studiengang: M.Sc. Informatik (Kernfach)
Matrikelnummer: 1138844
Email: s4beluek@uni-trier.de

ARHS Spikeseed
Rue Nicolas Bové 2B
1253 Luxembourg

16. Mai 2019

Inhaltsverzeichnis

1 Vorstellung des Unternehmens

Die ARHS Group ist ein Zusammenschluss verschiedener Entitäten mit insgesamt rund 1040 Mitarbeitern und Niederlassungen in Griechenland, Belgien und Luxembourg, die sich auf verschiedene Bereiche der IT spezialisieren. Die Entität, der ich zugewiesen wurde heißt ARHS Spikeseed.

1.1 ARHS Spikeseed

ARHS Spikeseed befasst sich mit Projekten rund um Webanwendungen und Business Logik für den öffentlichen Sektor:

- Beratung unter Anderem beim Umstieg auf Cloud-Dienste oder genereller Business-Planung
- Migration von Systemen (auch Planung und Vorbereitung von Migrationen)
- Entwicklung cloudbasierter Systeme oder Weiterentwicklung bestehender Systeme
- Beratung bei DevOps für bestehende Systeme

Grundsätzlich bietet ARHS Spikeseed keine eigenen Produkte an. Die Ausnahme stellt das von ARHS Spikeseed entwickelte Produkt Fleetback dar. Ich wurde für die Dauer des Praktikums dem Entwicklerteam von Fleetback zugewiesen.

2 Projekt des Praktikums

Abbildung 1: Fleetback-Logo[1]

Fleetback wurde 2014 zunächst exklusiv in Zusammenarbeit mit einem Autohaus entwickelt und ist das einzige Produkt, was von Spikeseed angeboten wird. Es handelt sich um ein CRM-System (Customer-Relationship-Management) mit mehreren Modulen, dass sich rund um den Autohandel, beziehungsweise den Arbeitsbetrieb in einem Autohaus dreht. Dafür wird eine Webanwendung und eine mobile Anwendung für Android und iOS verwendet, die sich wiederum in die zwei Bereiche - Backoffice und Frontoffice - unterteilen. Die Backoffice ist der Teil der Anwendung, mit dem

[1] https://www.fleetback.com/w/app/images/logo/fleetback.svg, Zugriff 11.04.19

Fleetback-Nutzer threadartige Dateien über Kunden, bzw deren Anfragen erstellen. Diese Threads (als Vorgang bezeichnet) enthalten je nach Modul verschiedene Informationen und Möglichkeiten Kontakt zum Kunden aufzunehmen, was über Frontoffice-Dateien geschieht. In einer Frontoffice-Datei werden die entprechenden Informationen und Anfragen aus dem Vorgang als HTML-Dokument präsentiert, welche unter einem statischen Link verfügbar gemacht und dem Kunden zugesandt wird. Dieser hat wiederum, je nach Modul, verschiedene Antwortmöglichkeiten, welche dann von Fleetback verarbeitet und an das Autohaus weitergeleitet werden. Fleetback hat verschiedene Module:

- Das Verkaufsmodul von Fleetback stellt ein Dealer-Management-System dar, mit dem Kunden und Neukunden eines Autohändlers verwaltet werden können und über das die Kommunikation über Angebote und Ähnliches laufen kann. Ziel der Anwendung ist es, gut designte Angebote erstellen zu können und eine moderne, einfache Oberfläche dafür zu bieten. Außerdem wird Managern des Autohauses eine Erhebung von Verkaufsstatistiken ermöglicht.
- Das Kundendienstmodul ist für Reparaturen und die Benutzung in Werkstätten vorgesehen. Hier geht es darum, dem Kunden möglichst viel Transparenz über anstehende und erledigte Arbeiten am Auto zu gewährleisten, indem bei der Darstellung für die Kunden (Frontoffice) Videos und Fotos hinzugefügt werden.
- Das Auditmodul übernimmt die Erfassung und das Management von Audits über die Arbeitsabläufe im Autohaus und ist für den internen Gebrauch innerhalb des Autohauses gedacht.

2.1 Programmaufbau

Die Realisierung von Fleetback gestaltet sich folgendermaßen.

- Java Spring Boot für das Backend
- AngularJS für das Frontend (aktuell findet eine Umstellung auf React statt)
- MongoDB für die Daten-Persistenz
- Amazon AWS für den Deploy der Anwendung
- Continous Integration Pipeline auf Gitlab

2.2 Entwicklungsprozess

Bei ARHS Spikeseed wird agile Softwareentwicklung angewandt, die sich durch verschiedene Merkmale auszeichnet:

- Das Arbeitsjahr wird durch zwei Roadmap-Meetings zweigeteilt. Bei einem Roadmap-Meeting präsentiert jeweils der Leiter eines jeden Teams des Projekts die bisherigen Fortschritte seit dem letzten Roadmap-Meeting. Dann werden zusammen mit der Führungsebene zukünftige Ziele und Kritik besprochen. Beispielsweise zeigt die Entwicklungsabteilung neue Features, die Marketingabteilung eine neue Kampagne und das Verkaufsteam die letzten Quartalszahlen.

- Die Zeit zwischen den Roadmap-Meetings ist in zweiwöchige Sprints unterteilt, die jeweils mit einer einstündigen Präsentation (Demo) enden, bei der jeder Entwickler seinen Fortschritt der vergangenen zwei Wochen präsentiert. Der neue Sprint beginnt dann mit dem auf die Demo folgenden Sprint-Planning, wobei, falls die individuellen Aufgaben erledigt wurden oder nur kleine Anmerkungen während der Demo gemacht wurden, neue Aufgaben durch den Teamleiter zugewiesen werden.

- Täglich, zur selben Uhrzeit, findet ein maximal 20-minütiges Treffen der am Projekt beteiligten Entwickler statt, wo jeder kurz zusammenfasst, woran er am vergangenen Tag und an diesem Morgen gearbeitet hat und ob Probleme aufgetreten sind und womit er plant, den Rest des Tages zu verbringen. Gibt es Probleme, werden diese nach dem Stand Up Meeting besprochen, da sie nicht immer für alle relevant sind.

3 Monatsberichte

3.1 Oktober '18

Das Fleetback-Team teilt sich in die Bereiche Development, Marketing, Sales, Support und Design auf. Der Bereich Development besteht aus Entwicklern für die verschiedenen Teile der Anwendung (App für iOS, App für Android, App für AppleTV, Backoffice Web-App). Zunächst erhielt ich eine detailierte Top-Down-Vorstellung des Produktes und wofür Fleetback verwendet werden kann. Daraufhin wurde ich der Weiterentwicklung der Backoffice zugeteilt und erhielt zu Beginn als Aufgabe verschiedene Bugfixes zu erstellen, wodurch ich die Struktur und den Aufbau des Projekts erforschen konnte. Die meisten Bugs beschränkten sich auf Darstellungsfehler der Web-App, also den durch AngularJs erstellten Teil. Da ich im Vorfeld des Praktikums bereits bei einigen privaten Projekten mit Angular gearbeitet hatte, war lediglich die Navigation, beziehungsweise das Zurechfinden im Projekt, dass seit 2014 stetig entwickelt wurde und somit durchaus komplex ist, schwierig. So konnte ich nach der ersten Woche bereits meine ersten Änderungen bei der Demo präsentieren. Im darauffolgenden Sprint-Planning

wurde mir die Aufgabe zugeteilt, ein Skript zur automatischen Erstellung der neuen Bedienungsanleitung von Fleetback zu erstellen.

3.1.1 User-Guide-Skript

Da Fleetback von Autohäusern und Werkstätten in verschiedenen Ländern und somit auch in verschiedenen Sprachen verwendet wird, bedeutet ein Update der Benutzeranleitung einige Schritte, die mit jedem neuen veröffentlichten Feature wiederholt werden müssen. Bis dahin wurde die Anleitung in Form von PDFs bereitgestellt, was zu einer leichter zu aktualisierenderen HTML-Seite geändert werden sollte. Meine Aufgabe war es, ein Skript zu schreiben, dass es Mitarbeitern ohne Programmiererfahrung ermöglicht aus einer Exceltabelle, bestehend aus Übersetzungen, also dem Inhalt der Anleitung, die Benutzeranleitung in den verschiedenen vorhandenen Sprachen zu generieren. Hierfür sollte ich NodeJs verwenden und meine Änderungen einem Projekt hinzufügen, dass verschiedene für Fleetback verwendete Tools beinhaltet. Für das Projekt musste ich mit der Marketing und der Design Abteilung des Teams zusammenarbeiten, da Marketing für den Inhalt und später auch die Erstellung der Benutzeranleitung verantwortlich ist und Design für das Template, in das der Inhalt eingefügt wird. Problematisch hierbei war besonders das automatische Einfügen der Bildpfade, da die Dateinamen im Excel-Dokument, welches aus der Wordvorlage der vorherien Anleitung erstellt wurde, Sonderzeichen, wie ein Accent Aigu, Leerzeichen oder keine Dateiendungen hatten. Dies lies sich jedoch mit einfachen Kommandozeilenbefehlen beheben.

3.1.2 Dealer-Managerment-Inteface

Mein nächstes Projekt war die Erstellung eines Tools für das Supportteam von Fleetback, welches sich um Anfragen und Probleme von Kunden kümmert. Beim Dealer-Management-Interface sollten alle relevanten Informationen über die Fleetback-nutzenden Autohäuser, zusammengetragen werden und darüber hinaus, Platz für weitere manuell einzutragende Informationen, wie Kontaktpersonen oder Kommentare sein. Zu realisieren war dies mit dem gleichen Technology Stack, wie bei Fleetback, allerdings sollte das Frontend durch ReactJs ersetzt werden. Da sich das Projekt außerhalb von Fleetback befinden und unabhängig davon laufen sollte, wurde mir ein quasi leeres Spring Boot Template vorbereitet, an dem ich mich versuchen sollte. Das Backend des Dealer-Management-Interface war an sich sehr schlicht, da hier lediglich Authentifizierung und Datenpersistenz vorhanden sein mussten, was überwiegend von Spring Boot und Spring Security automatisiert wird. Da ich allerdings zuvor noch nicht mit ReactJs gearbeitet hatte, musste ich mich zunächst einen Tag einarbeiten, wobei mir einer der Senior Developer des Teams, welcher bereits Erfahrung damit hatte,

half. Zusammen mit dem Supportteam überarbeiteten wir das Mockup und verfeinerten die benötigten Funktionen. ReactJs erwies sich dabei als sehr flexibel und leicht anpassbar an Änderungen, bei Kritiken oder beim Hinzufügen neuer Features wie Filtern bei der Suche oder Ähnlichem. Sobald ich ein lauffähiges Programm hatte wurde eine Continous Itegration Pipeline in Gitlab eingerichtet, welche den Build übernimmt und einen automatischen Deploy des Development Branches in der Testumgebung des Amazon Web Services vornimmt, sodass das Tool testbar ist.

3.2 November '18

Den größten Teil des Novembers verbrachte ich mit der Verfeinerung und Anpassung des Dealer-Management-Interfaces und der Vorbereitung des Deploys. Zudem erhielten wir die Übersetzungen der Benutzeranleitung auf verschiedenen Sprachen, welche in die Benutzeranleitungen durch das von mir geschriebene Skript übersetzt wurden. Außerdem widmete ich mich diversen Bugfixes. Ein großes Problem des Projektes zeigte sich hier: Dadurch, dass kontinuierlich neue Features geplant und eingeplegt werden bleibt kaum Zeit für Refactoring des Codes. Fleetback war zunächst nicht für den Umfang ausgelegt, den es nun hat. Viele der Komponenten, sowohl im Backend, als auch im Frontend sind sehr spezifisch für einen Fall geschrieben, sodass sie nicht wiederverwendbar sind. Ein Beispiel hierfür bietet ein Bug, den zu lösen mich viel Zeit gekostet hat (und scheinbar vorher auch schon andere). Innerhalb einer Datei, die mit Fleetback an einen Kunden des Autohauses geschickt wird, können Preise für die Reparaturen festgelegt werden. Diese können ohne oder mit Mehrwertsteuer, mit Rabatt, mit Rabatt auf die Mehrwertsteuer oder prozentualem Rabatt eingegeben werden. All diese verschiedenen Möglichkeiten sorgen für eine sehr komplizierte Struktur im Code, da der gleiche Wert durch fünf verschiedene Eingabefelder geändert werden kann, die sich wiederum untereinander aktualisieren. Innerhalb dieser Aktualisierungen kam es zu Rundungsfehlern bei bestimmten Eingaben. Wäre von Anfang an geplant gewesen, dass die Eingabe mit Mehrwertsteuer und den restlichen Möglichkeiten hinzugefügt werden, hätte man das gesamte Modul anders aufbauen können. Dieses Phänomen scheint zieht sich durch einen Großteil des Projektes zu ziehen. Die Komponenten werden nicht generisch und wiederverwendbar genug geschrieben.

3.3 Dezember '18

Nach einigen Wochen des Bugfixen, wurde mir die Überarbeitung, einer Funktion zugeteilt. Diese Funktion besteht aus einer Konfigurationsseite und einer Integration in den Dateien (Preisangebote, Reperaturvorschläge), die von den Fleetback-Nutzern wiederum an ihre Kunden geschickt werden. Ziel war es, den Leitern des Autohauses die Erstellung verschiedener Nachrichten

in den vorhandenen Sprachen, die dann den Angestellten zur Verfügung stehen zu ermöglichen. Ich erhielt ein Mockup vom Desginteam und erörterte mit meinem Teamleiter, wie sich die Funktion in verschiedenen Randfällen verhalten sollte, da es bei der Auswahl der Übersetzung durch die Angestellten Sonderfälle gab. Der Bearbeitung dieser Anfrage folgte eine Vorstellung vor Mitarbeitern des Sales-Teams, die mit der Umsetzung äußerst zufrieden waren.

3.4 Januar '19

Da einer der Kollegen die Abteilung wechselte und nicht mehr an Fleetback arbeitete übernahm ich sein Projekt, einen Chatbot zu erstellen, der Kunden eine automatisierte Kontaktaufnahme im Schadensfall ermöglicht.

3.4.1 Chatbot (Fleetbot)

Der Chatbot sollte in verschiedenen Sprachen (französisch, englisch und deutsch) verfügbar und als Webchat und via Facebook Messenger erreichbar sein. Das Microsoft Bot Framework deckt diese Anforderungen besser ab, als beispielsweise das IBM Watson Framework, daher hat sich mein Vorgänger dafür entschieden. Bevor der Chatbot bei einem Kunden, der bei der Entwicklung des Chatbots mitgewirkt hat, live geschaltet wurde, galt es noch diverse Feinheiten anzupassen und den Gesprächsverlauf zu überarbeiten. Hierfür waren Äderungen im (Javascript-)Code der Chatbot Implementierung und der API-Calls im Fleetback-Backend nötig.

3.5 Februar '19

Ab dem ersten Sprint im Februar war die Prämisse, dass in den kommenden Monaten zunächst keine neuen Features für Fleetback implementiert werden. Stattdessen sollten bestehende Funktionen und Module des Produktes verfeinert und übrig gebliebene Tickets, die meistens kleinere Bugfixes waren, gelöst werden, da ein Großteil der im letzten Roadmapmeeting vorgestellten Features implementiert waren. Somit bestand die Arbeit darin kleinere Anfragen zu implementieren. Mir wurden mehrere Anfragen die das Hinzufügen von Konfigurationsmöglichkeiten für verschiedene Features umfassten zugewiesen.

3.6 März '19

3.6.1 User Guide Editing Engine

Eine der Anforderungen, die an das von mir zu Anfang geschriebene User Guide Skript gestellt waren, war, dass die HTML-Seiten aus einer einzelnen Excel-Tabelle erstellt werden kann. Dies bedeutete allerdings, dass mehrere Leute Ihre Änderungen an diesem Dokument synchronisieren mussten,

weshalb wir auf Google-Sheet umgestiegen sind. Das Bearbeiten der Benutzeranleitung via Spreadsheet stellte sich allerdings auf Dauer als sehr umständlich und unflexibel heraus und die Verwendung der reinen Skripte erforderte die Installation von Node und die Verwendung der Kommandozeile, was sich als nichttauglich für die Marketingabteilung zeigte. Daraufhin stand der Vorschlag im Raum dafür ein Lightweight CMS zu verwenden um den Nutzern ein WYSIWYG zu ermöglichen. Meine Aufgabe bestand darin, verschiedene Lösungen zu vergleichen und zu präsentieren. Unter Anderem schlug ich eine selbstgeschriebene Lösung, da sich der benötigte Featureumfang sehr gering hielt und sich die Möglichkeit einer weiteren praktischen Anwendung meiner gelernten Fähigkeiten in React bot, wofür der Teamleiter sich dann auch entschied. Da sich die Nutzerzahl des Tools sehr gering hielt, war es nicht zwingend nötig eine eigene Webanwendung zu hosten, also wählte ich ElectronJs als Desktop-Wrapper für die Webanwendung. Electron ist sehr leicht konfigurierbar und kompatibel mit MacOS und Windows, weshalb es sich optimal für den Anwendungszweck eignete. In Zusammenarbeit mit den zukünfitgen Nutzern stellte ich Funktionen zusammen und verfeinerten die Anwendung.

4 Fazit und Rückblick auf das Praktikum

Das Praktikum hat mir einen tiefen und klaren Einblick in die Aufgaben und das Arbeitsleben eines Programmierers in der freien Wirtschaft ermöglicht. Unglücklicherweise trat ich dem Fleetback-Team zu einem Zeitpunkt bei, in dem keine großen neuen Features implementiert wurden, was zur Folge hatte, dass ich immer wieder an kleineren Projekten arbeitete. Dies hatte zwar den Vorteil, dass ich viele verschiedene Technologien anschneiden, allerdings keinen Bereich wirklich vertiefen konnte. Parallel zum Praktikum widmete ich mich daher einem privaten Projekt, wo ich die Programmiertechniken und Ideen, die ich lernte weiterführend anwendete. Hierbei fiel die viele Programmierpraxis, die ich durch das Praktikum hatte schwer ins Gewicht und die Umsetzung meiner Vorstellungen für das Projekt war ein Leichtes.

Am Ende des Praktikums stellte sich mir die Frage, ob ich bei der Firma bleibe und das Beenden meines Masters in die fernere Zukunft verschiebe. Die Entscheidung, das Angebot abzulehnen fiel mir leicht. Die Aufgaben, die ich erhielt und die Arbeit als solche bereiteten mir durchaus Freude, da die Anfragen meistens abwechslungsreich und fordernd waren. Allerdings fiel es mir sehr schwer zu akzeptieren, dass all die Arbeit und all die Zeit, die auf das Entwickeln verwendet wurde lediglich einem sehr kleinen Kreis von Nutzern dient und ausschließlich wirtschaftliche Interessen verfolgt.

Das Praktikum hat mir dabei geholfen, zu erkennen, dass ich mir selbst nicht vorstellen kann in der freien Wirtschaft zu arbeiten, sondern meine Zeit als Programmierer mit der Anreicherung von Wissen durch Forschung verbringen möchte. Eine Möglichkeit sehe ich zum Beispiel im Mittelweg zwischen Forschung und Wirtschaft bei der Arbeit in Forschungsinstituten.