Praktikumsbericht

Joshua Coelho Mestre

4. September 2019

Zusammenfassung

Der Bericht zum absolvierten Pflichtpraktikum bei Scholz & Volkmer Wiesbaden.

Inhaltsverzeichnis

1 Einleitung

Scholz & Volkmer ist eine Kreativagentur zur digitalen Markenführung mit ungefähr 130 Mitarbeitern und Firmensitz in Wiesbaden und Berlin.

Die Agentur arbeitet im Kundenauftrag und erstellt Websites, Mobile Apps und weiter digitale Produkte zur Verbesserung der digitalen Präsenz.

In meiner Zeit bei Schovo arbeitete ich vorwiegend als Frontend Entwickler für den Kunden Adidas, der neben seiner Corporate Seite auch einen Blog bei Schovo entwickeln lässt.

Während meines Praktikums war ich für die Umsetzung eines umfangreichen A/B-Tests der auf dem oben erwähnten Blog namens "Gameplan A"durchgeführt werden sollte.

2 Kalenderwoche 23

Das Praktikum begann mit einem großen Meeting bei dem ich die Gelegenheit bekam mich der ganzen Firma vorzustellen.

Darauf folgte der Erhalt meines Arbeitsrechners in der IT-Abteilung sowie das treffen mit meiner Praktikumsbetreuerin.

Meine Praktikumsbetreuerin zeigte mir meinen Arbeitsplatz und gab mir eine Einführung in die firmeneigene Softwarestruktur und Abläufe. Darunter zum Beispiel den zum Dateien-Austausch genutzten Fileserver, das häufig genutzte Ticketsystem Jira und das für den Informationsaustausch genutzte Confluence.

Ich machte mich danach mit den eben genannten Systemen vertraut und richtete meinen Arbeitsrechner ein.

Am folgenden Tag durfte ich mich mit dem Code der firmeneigenen Website vertraut machen. Dazu klonte ich das, das sich auf dem Gitlab-Server befindliche Projekt, und lies es mittels Docker-Container auf meinem Arbeitsrechner laufen.

Das Setup des Projekts erfolgte mittels im Repository enthaltener Scripte.

Da geplant war mich als Backend-Developer einzusetzen und es sich bei der S-V.de Webseit um ein Django Projekt handelt, absolvierte ich den Rest der Woche ein detailliertes Django Tutorial um meine im Studium erhaltenen Django-Kenntnisse aufzufrischen und zu verbessern.

Das Ziel des Tutorials war es, abgesehen von der Verbesserung meiner Django Kenntnisse, eine Website zu erstellen, die es den Usern erlaubt Links zu posten und diese einer Bestimmten Category zuzuordnen.

Zusätzlich dazu sollte es den Usern ermöglicht werden direkt auf der Seite das Internet nach neuen Seiten anhand von Suchbegriffen zu durchforsten und bereits vorhandene Links zu liken".

Das Tutorial befasste sich mit allen relevanten Gesichtspunkten der Webentwicklung und vermittelte zusätzlich zu dem Basiswissen der Django-Programmierung auch Inhalte wie

die Nutzung von CSS-Frameworks wie Bootstrap und die Nutzung externer APIs um die Funktionalität der Webseite um eine Websuche zu erweitern.

3 Kalenderwoche 24

Zu Beginn der zweiten Woche beschäftigte ich mich mit einem auf der s-v.de auftretenden Problem.

Im falle einer Unerreichbarkeit der CDN-Server auf denen die Medien der Seite Abgelegt sind sollten diese aus dem Cach des Django Caches geladen werden, was jedoch nicht zuverlässig geschah.

Ich beschäftigte mich zunächst mit der lokalen Reproduktion des Fehler um ihn dann schrittweise zu lokalisieren. Auf Grund der Projektlage wurde ich jedoch von dem s-v.de-Projekt abgezogen und auf die Adidas Corporate Seite angesetzt.

Der Kunde wünschte sich ein Update für das sich auf der Seite befindliche Medien-Archiv.

Das für die Filterung der Medienformate benutzte Dropdown-Menü, welches es dem Nutzer ermöglicht aus Audio, Video und Bilder zu wählen, sollte durch drei nutzerfreundlichere Checkboxen ersetzt werden.

Nach dem Klonen und lokalen einrichten des Projekts bestand meine Arbeit zunächst daraus mich mit dem Projekt und dem verwendeten, firmeneigenen Javascript Framework, vertraut zu machen.

Nach meiner Orientierung begann ich mit der Umsetzung des Updates, indem ich zunächst das unerwünschte Dropdown-Menü aus dem betroffenen HTML-Template entfernte und das Markup für die geforderten Checkboxen inklusive des, von der Konzepterin geforderten, Stylings hinzufügte.

Der nächste Schritt beinhaltete Implementierung der Filter-Funktionalität der Checkboxen, wobei sich das Hauptproblem darin ergab, dass statt wie im Falle des Dropdown nun nicht mehr nur eine Kategorie Medien ausgewählt werden kann sondern mehrere oder sogar gar keine Checkbox ausgewählt sein kann.

4 Kalenderwoche 25

Um das Problem der Mehrfachfilterung der Medien durch die Checkboxen zu lösen habe ich mich intensiv mit dem bereits vorhandenen Filter-Modul beschäftigt.

Die Anforderung war das bereits existierende Modul mit möglichst geringer Kopplung und möglichst minimal inversiv dahingehend zu verändern, dass es mit der Eingabe über die Checkboxen die gewünschten Medien einblendet.

Da das Filter-Modul mehrfach auf der Seite benutzt wird und bis dahin ausschließlich über die Dropdown-Menüs gesteuert wurde stand ich vor der Herausforderung die Funktionalität des Moduls um die Checkbox zu erweitern jedoch das verarbeiten der Eingaben durch die Dropdowns nicht zu behindern.

Um der Anforderung gerecht zu werden habe ich innerhalb des Moduls eine Funktion implementiert, die im Falle einer Eingabe durch die Checkboxen aufgerufen wird und diese soweit verarbeitet, dass sie danach von wie gewohnt von dem Filter weiter genutzt werden kann.

Durch die selbst implementierte Funktion musste ich anschließend nur noch geringfügige Anpassungen an dem Modul vornehmen, da das nun nicht mehr nur noch ein Filter aktiv sein kann sondern mehrere.

Die Änderung beschränkte sich auf die Erweiterung des Moduls um die Möglichkeit durch ein Filter-Array zu iterieren statt nur ein einziges Filterwort zu akzeptieren.

Nachdem ich den Task beendet hatte ließ ich meinen Code von einem Senior Developer reviewen, der mich auf einige Sachen hinwies, die ich nicht bedacht hatte.

Unter anderem hatte ich Funktionen benutzt, die von Internet Explorer 11 nicht unterstützt werden. Dies ist vor allem problematisch, da viele Rechnerau Seiten des Kunden diesen Browser noch benutzen.

Ich behob die durch die Code-Review entdeckten Fehler und nutzte die Gelegenheit um noch weitere kleine Verbesserungen vorzunehmen, die ich durch mein verbessertes Verständnis für Javascript und das benutzte Framework entdeckt habe.

5 Kalenderwoche 26

Da ich mich nun in der Projekt Adidas Corporate Seite, und insbesondere in das Medien-Archiv, eingearbeitet hatte übernahm ich weitere Tickets diesbezüglich.

Ich kümmerte mich in dieser Woche um einen bekannten Bug, der zur Auswirkung hatte, dass wenn man sich das Medien-Archiv im apple-eigenen Browser, Safari, anschauen wollte, keine Bilder zu sehen waren.

Ich begann damit den Code zu debuggen und mir parallel dazu sowohl die für die Bilder benutzten Templates, als auch das für das laden der Bilder verantwortliche Javascript-Modul zu Gemüte zu führen.

Mir fiel beim Debuggen auf, dass wenn ich das loadedEvent für die Bilder händisch abfeuerte, alle Bilder korrekt angezeigt wurden.

Dieses Verhalten wies darauf hin, dass die Bilder bereits alle geladen waren, jedoch auf Grund des fehlenden Events "hidden"blieben. Dies entsprach dem Code den ich im Bilder-Modul vorfand, wo mit einem Eventlistener auf das Laden der Bilder gewartet wurde um dann alle gleichzeitig anzeigen zu können.

Das Laden der Bilder übernimmt ein Lazyloader, welcher das src-Attribut der Images erst setzt, wenn sie benötigt werden, sodass unnötige clientseitige Operationen und Request gespart werden können.

Die Vermutung lag nahe, dass das Event gefeuert wird noch bevor das Bilder-Modul den Eventlistener anmelden kann.

Durch weiteres Debugging fiel schlussendlich auf, dass bereits im Template das src-Attribut gesetzt wurde anstatt die Quelle in das data-src-Attribut zu schreiben, was zum laden der Bilder führte noch bevor das Javascript initialisiert war wodurch nicht auf das Event reagiert werden konnte, da es nicht gefeuert wurde, da durch das bereits gesetzte Attribut der Lazyloader übergangen wurde.

Eine Korrektur des Fehlers im Template eliminierte den Bug und ermöglichte eine Korrekte anzeige in allen gängigen Browsern.

6 Kalenderwoche 27

In der Kalenderwoche 27 begann meine Arbeit an dem Projekt Gameplan A. Dabei handelt es sich um den Blog von Adidas der sich an sogenannte Business-Athleten richten soll.

Ich wurde mit der Aufgabe Betraut drei Umfangreiche A/B-Tests auf der Seite zu implementieren, die dazu dienen sollen den Blog, der auch zur Akquise von jungen aufstrebenden Mitarbeitern für Adidas, für Nutzer attraktiver zu gestallten.

Zu beginn der Woche erhielt ich ein Briefing zu den geplanten A/B-Tests durch das Projektmanagement und das Konzept.

Der erste umzusetzende Test beschäftigte sich mit der Tonalität des Blogs. Die These die es zu überprüfen galt ist, dass eine andere Formulierung der Headlines zu mehr Pageviews pro Session führt.

Die Anforderung lautete einen A/B-Test mit Hilfe von Google Optimize zu entwickeln, der diese These entweder bestätigt oder verwirft.

Google Optimize ist ein Tool welches erlaubt A/B-Tests durchzuführen und dabei auch andere Google Tools wie den Tagmanager und/oder Google Analytics einzubinden. Dadurch eigente sich Optimize besonders gut für die Verwendung auf Gameplan A, da beide genannten Tools verwendet werden.

Optimize ermöglicht durch einen eigenen Editor zusätzliches CSS und/oder Javascript für eine Variante der Seite zu injizieren.

Da es darum ging auf der gesamten Seite alle Headlines der Blog-Posts zu ändern und dies über Javascript sehr rechenintensiv gewesen wäre entschieden wir uns für eine Implementierung über das CSS der Seite.

Wir nutzten die Möglichkeiten von Optimize und setzten bei der Variante der Seite eine CSS-Klasse im Body-Tag namens "gpa-test-ab-headline-X"wodurch wir mit zusätzlichem CSS die wir dann bereits mit der Seite an den Client liefern auf die Variante reagieren können und alle Headlines des Blogs austauschen können.

Am Ende der Woche besuchte ich eine SEO Schulung welche von einem SEO Experten

gehalten wurde, der seitens der Agentur engagiert wurde um den Kunden eine bessere Performance bei Suchmaschinen bieten zu können.

Die Schulung erstreckte sich über einen gesamten Tag, wobei sich der Vormittag einem Überblick über das Thema SEO, und der konzeptionellen Aspekte des Themas widmete.

Der Nachmittag befasste sich dann mit allen technischen Aspekten von SEO, was für mich und meine Abteilungs-Kollegen eine besonders hohe Relevanz hatte und sehr gut aufbereitet wurde.

7 Kalenderwoche 28

In dieser Woche befasste ich mich zunächst mit der Backend-Seitigen Anpassungen, die für den A/B-Test nötig waren.

Bei dem Backend handelt es sich bei Gameplan A um eine auf Kundenbedürnisse angepasste Version von Wordpress, die mit einem komplett eigenentwickeltem Theme operiert.

Um ein Austauschen der Titel zu realisieren müssen die Editoren des Blogs zunächst die Möglichkeit haben den einzelnen Blog-Posts einen alternativen Titel zu geben und diesen für den jeweiligen Post zu speichern um den alternativen Titel später an den Client zu liefern.

Ich habe dafür das Blog-Post Formular, mit dem Posts erstellt und bearbeitet werden, im Backend um das Feld Älternative Headlinesërweitert und die dort hinterlegten Titel dann in den Metadaten des Post gespeichert.

Dies erforderte ein eigenes php-Modul zur Verarbeitung und Speicherung der Daten und die Erstellung eines neues Twig-Templates zur Erstellung der zusätzlichen Feldes.

Die Twig-Templating-Sprache wird im Gameplan A Projekt benutzt um eine klare Trennung des php-Codes und des Markups zu erhalten.

Neben meiner Tätigkeit für Gameplan A wurde ich von einem Projektmanager, der sich primär um den Kunden Fraport kümmert um eine Technische Konsultation gebeten. Dabei ging es darum, dass der Kunde in Besitz neuer Geo-Json Daten des Flughafen Frankfurt

gelangt ist und nun die Frage aufkam was genau diese Daten repräsentieren und welche Möglichkeiten zur Visualisierung dieser Daten es gibt.

Da ich bereits in einem Projekt im Fach Mobile Computingïn Kontakt mit Geo-Json-Daten gekommen bin konnte ich Fragen bezüglich der Daten zügig und zur Zufriedenheit des Kunden in einem Meeting klären.

Für den A/B-Test habe ich mich nach den Ergänzungen im Backend Angefangen die gespeicherten Titel überall da im Template mitzugeben, wo der Titel eines Blogs auftaucht. Ich habe es in ein Attribut namens "data-test-ab-headline-X"des entsprechenden HTML-Tags geschrieben um mittels der CSS-Funktion attr() darauf zugreifen zu können.

Um den Titel schlussendlich mittels CSS auszutauschen habe ich in einem zusätzlichen .scss-File die Schriftgröße des Originalen Titels auf Null gesetzt und dann mittels einer Pseudo-Klasse den Content der Headline mit dem Wert des oben genannten Attributes gesetzt.

Das oben beschriebene Austauschen findet selbstverständlich nur Anwendung wenn durch Optimize die in Kapitel 6 beschriebene Klasse gesetzt wird. Passiert dies nicht hat der Nutzer das Original erhalten und sieht dementsprechend auch die Originalen Headlines.

8 Kalenderwoche 29

In Fortsetzung zu meiner Arbeit am A/B-Test bezüglich dem Austausch der Blog-Post-Titel setzte ich mich in dieser Woche zunächst an die Behandlung der Sonderfälle der Headlines, die ebenfalls durch das eigens für den Test hinzugefügte CSS abgedeckt werden müssen.

Ein Beispiel dafür ist die sich ändernden Schriftgrößen, die sich nach der Breite des Viewports richten.

Durch die Nutzung der CSS-Pseudo-Klasse wird die ursprüngliche Font-Size überschrieben und muss explizit für die Pseudo-Klasse erneut angegeben werden.

Ein Weiterer Sonderfall ist ein kleiner Teaser für den nächsten Blog-Post der Auftaucht wenn man den Aktuellen schon fast durchgelesen hat.

Der Titel des angeteaserten Artikels wird mittels Javascript in den Teaser geschrieben. Um diesen Titel auszutauschen kam ich nicht drum herum ein kleines Javascript-Modul zu schreiben, welches erkennt ob es sich bei der präsentierten Seite um das Original oder die Variante handelt und dem entsprechend den benötigten Titel in den Teaser schreibt.

Ich habe alle, extra für den Test angelegten, .js- oder .scss-Dateien deutlich mit dem Präfix "gpa-ab-test-ïm Namen gekennzeichnet sodass sie zu einem späteren Zeitpunkt, nach Ablauf des Tests, einfach gefunden und identifiziert werden können um sie zu entfernen, da sie im Normalfall nicht benötigt werden.

In der Mitte der Woche wurde ich für einen dringenden Auftrag für den neu gewonnen Kunden Aldi vom A/B-Test abgezogen und gebeten für einen Usability-Test einen Klickdummy zu entwickeln.

Die herkömmlichen Methoden, wie zum Beispiel die Verwendung der bekannten inVision Software gelang auf Grund der gehäuften Nutzung von Slidern im Klickdummy, an ihre Grenzen.

Zusammen mit einem Senior Developer setzte ich ein möglichst unkompliziertes Webpack Projekt auf, welche eine schnelle Entwicklung der Dummys ermöglichte und arbeitete Hand in Hand mit den Konzeptern und Gestaltern daran sowohl den Status Quo als auch die verbesserte Version des selben umzusetzen.

Es handelte sich dabei um die Angebotsübersicht innerhalb der Aldi App, welche durch reduzierten Einsatz von Slidern für den Nutzer einfacher zu bedienen werden soll.

Da der Test bei einem externen Dienstleister laufen sollte mussten wir beide Versionen des Klickdummys anschließend noch auf einem öffentlich zugänglichen Server mittels FTP deployen.

Zum Abschluss der Woche beschäftige ich mich noch mit dem sorgfälltigen Testen meiner A/B-Test implementierung.

Ich überprüfte sowohl die Browser-Kompatibilität als auch die einwandfreie Funktion bei verschiedenen Bildschirmauflösungen.

9 Kalenderwoche 30

Zum Abschluss der Woche kehrte ich auf Kundenwunsch zurück zur Adidas Corporate Seite um neu aufgekommene Anforderungen umzusetzen.

Der Kunde wünschte, dass die von mir implementierten Checkboxen initial alle gecheckt sein sollten. Um dies zu gewährleisten war lediglich eine kleine Änderung an dem HTML-Template der Checkboxen von nöten.

Um gezielter einen bestimmten Zustand des Medien-Archiv teilen zu können wünschteder Kunde sich zusätzlich die Implementierung von sogenannten Deeplinks.

Diese Deeplinks ermöglichen das teilen einer Seite wobei der Empfänger durch, in der Url codierte, Informationen genau den Zustand einer Seite ansehen kann, den der Absender ihm zukommen lassen möchte.

Diese zusätzlichen Informationen in der Url sind nötig, da eine Seite nicht bei jeder Interaktion neu geladen wird sondern durch Javascript verändert wird. Teilt man nun den Link ohne zusätzliche Informationen würde der Empfänger die Seite in ihrem initialen Zustand erhalte, was unter Umständen nicht gewünscht ist.

Nach der Verarbeitung der in der Url mitgegebenen Zustands-Informationen durch das in Javascript geschriebene Deeplink-Modul, wird die vom Absender durchgeführte Interaktion beim Empfänger der Links virtuell wiederholt.

Um dies für die Filterung der Inhalte des Medien-Archivs zu ermöglichen musste ich die für das Deeplink-Modul benötigten Informationen bei Interaktion des Users in die Url integrieren.

Abgesehen von der Arbeit an den Deeplinks beschäftige ich mich in dieser Woche mit der Umsetzung des zweiten A/B-Tests, der sich mit der Verminderung der Share-Möglichkeiten befasst.

Die Hypothese bei diesem Test verifiziert werden sollte behauptet, dass eine Verringerung der Share-Möglichkeiten auf vier Möglichkeiten, einen Anstieg der Gesamtanzahl der Shares zur Folge hat.

Konkret lautete die Aufgabe demnach die Share-Möglichkeiten bei der Variante der Seite, von ursprünglich neun, auf vier zu minimieren.

Die Verbleibenden Share-Möglichkeiten sollten Facebook, Linkedin, Twitter und Mail sein.

Wie auch bei den Headline-Test, benutzte ich Google Optimize um eine CSS-Klasse zu setzen um eigens für den Test geschriebene CSS-Regeln zur Anwendung zu bringen.

Ich versteckte die unerwünschten Share-Buttons indem ich deren Display-Attribut auf nnoneßetzte und behob im Zuge meiner Arbeit an den Share-Buttons einen Bug, der bei der Eliminierung des Google-Plus Buttons entstanden war und bis dato nicht aufgefallen war.

Durch den Fehlenden Google-Plus Button war eine Lücke in der Sharebar entstanden, die genau der breite des fehlenden Buttons.

10 Kalenderwoche 31

Nach der vorläufigen Beendigung meiner Arbeit an dem Share-Test begann ich mit der Vorbereitung des dritten und umfangreichsten Tests.

Die Hypothese die in diesem Fall geprüft werden sollte besagt, dass eine Änderung der Bilder von Hochglanz-Aufnahmen zu Bildern mit einem Selbst-Gemacht-Look mehr Pageviews pro Session generieren.

Ich habe mich für die Umsetzung zunächst wie im Falle des Headline-Tests darum gekümmert, dass es den Editoren möglich ist ein Alternatives Foto hochzuladen.

Die geschieht wieder über das für die Editoren gewohnte Wordpress-Post-Formular, in das ich ein zusätzliches Feld implementiert habe welches es erlaubt ein gänzlich neues Foto hochzuladen, oder ein anderes bereits hochgeladenes Bild auszuwählen.

Um die Quelle des alternativen Bildes für den späteren Austausch in der Variante der Seite für das Javascript zugänglich zu machen ergänzte ich in den Betroffenen Templates die img-Tags um das Attribut "data-v1-src"worin ich anschließend die Quelle des alternativen Bildes hinterlegte.

Des ursprüngliche Umsetzungsplan war die alle betroffenen Bilder, mit der Alternativen

Quelle im Tag, gleichzeitig durch ein, durch Google Optimize in der Variante der Seite injiziertes Javascript, auszutauschen.

Da dies, dadurch das es sich bei GameplanA um ein sehr bild-lastigen Blog handelt, jedoch eine sehr rechenintensiver Task ist, war ich gezwungen den Plan zu ändern.

Durch einen signifikanten Performance unterschied wäre die Integrität des Tests bedroht gewesen, da eine Änderung der Messdaten nicht nur auf die Änderung der Bild-Sprache zurück zu führen wäre.

Zudem bestand die Gefahr, das die originalen Bilder geladen würden noch bevor das Script das src-Attribut hätte manipulieren können, womit wieder rum kein Verlass auf die Messdaten sein würde.

Die originalen Bilder könnten dem Benutzer kurze Zeit angezeigt werden bevor sie ausgetauscht würden, was den User irritieren würde, was zum Beispiel ein Anstieg der Bounce-Rate führen könnte.

11 Kalenderwoche 32

Zur Lösung der in Kapitel 10 beschriebenen Probleme musste ich zunächst ein weiteres Gameplan-A-Ticket bearbeiten.

Zur allgemeinen Performance Verbesserung der Website sollte Blogweit ein Lazy Loader für Bilder implementiert werden, der es ermöglicht Bilder erst dann zu laden wenn sie im Viewport erscheinen beziehungsweise kurz davor.

Dies setzt man in der Regel um indem man das src-Attribut des Img-Tags leer lässt sodass initial keine Bilder vom Browser geladen werden.

Später werden die src-Attribute bei erscheinen der Bilder im Viewport gesetzt und somit das Laden der Bilder durch den Browser angestoßen.

Zur Feststellung ob ein Bild im Viewport der Browser erscheint habe ich die sogenante InterserctionObserver API benutzt, die von den meisten modernen Browser angeboten wird. Für Internet Explorer gibt es zusätzliche Polyfills die eine Benutzuung der Api ermöglichen.

Ich habe den IntersectionObserver in einen selbstgeschriebenen Service eingebaut, der als Singelton von allen Javascript-Modulen benutzt werden kann.

Die Module übergeben sich selbst dem Service, welcher wiederum die Img-Elemente aus dem betreffenden Markup heraus sucht und dem Intersection Observer übergibt.

Tauchen die Elemente nun im Viewport auf so wird der Intersection Observer getriggert und führt eine ihm mitgegebene im Lazy Loader Service definierte Funktion aus, die das Attribut "data-srcäus dem betreffenden ELement ausliest und es in das src-Attribut schriebt.

Durch die Implementierung des Lazy Loader Service braucht es nun initial keine gesetzten src-Attribute mehr was eines der in Kapitel 10 beschriebenen Probleme löste.

Zudem löste das Just In Time laden der Bilder auch das oben erwähnte Performance Problem.

Ich setzte nun mit Google Optimize lediglich eine Globale Variable, auf die ich in der Load-Funktion des Lazy Loader Services reagieren konnte mit einer einfachen if-Abfrage.

Ist die Variable gesetzt benutzt die Load-Funktion das in 10 beschriebene "data-v1-src" Attribut, wenn nicht dann das "data-src".

12 Kalenderwoche 33

13 Kalenderwoche 35

14 Fazit