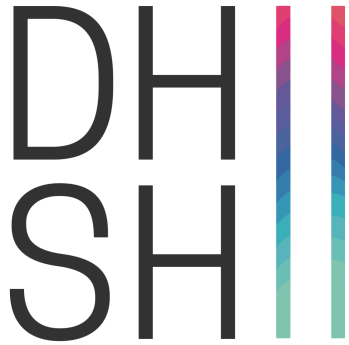


BACHELOR-THESIS

Fachrichtung Wirtschaftsinformatik

Chancen und Risiken einer Migration von Webanwendungen zu nativen Systemen

Studiengruppe:	119 WINF
Eingereicht von:	Fabian Reitz Hasseldieksdammer Weg 13 24114 Kiel +49 175 6392445 fabian.reitz@stud.dhsh.de
Erstgutachter DHSH:	Prof. Dr. Alexander Paar Hans-Detlev-Prien-Straße 10 24106 Kiel +49 431 3016255 alexander.paar@dhsh.de
Gutachter des Betriebes:	Marc Köster Mittelstraße 7 | Hinterhaus 24103 Kiel +49 431 53015400 koester@stadtwerk.org
Zweitgutachter DHSH:	Prof. Dr. Michael Sachtler Hans-Detlev-Prien-Straße 10 24106 Kiel +49 431 3016170
Abgabetermin:	16.05.2022

Inhaltsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
	deutsch: Programmierschnittstelle
CSS	Cascading Style Sheets
HTML	Hypertext Markup Language
W3C	World Wide Web Consortium
	deutsch: Internet Konsortium
Webapp	Web Application
	deutsch: Webanwendung

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Einleitung

Während es unmöglich ist, den Ursprung des Internets auf einen exakten Zeitpunkt festzulegen, lässt sich jedoch mit Gewissheit sagen, dass die Entwicklung des Internets einen bedeutsamen Wendepunkt in der Geschichte der Menschheit darstellt (Kleinrock 2010: 26). Floridi (2010) sieht in dem Internet als „Infosphere“ (Floridi 2010: 9) die vierte Revolution in einer Reihe von weltverändernden Wandlungen. Zu diesen Wandlungen gehören die Kopernikus-Revolution, die Darwin'sche Revolution und die Freud'sche Revolution. Diese Wenden veränderten das grundlegende Verständnis der Menschen sowohl über ihre Umwelt als auch über sich selbst (Floridi 2010: 8f.).

Das Internet befindet sich in einem stetigen Wandel. Der Beginn des modernen Internets wird im Kontext dieser Arbeit auf den Zeitpunkt datiert, als Sir Tim Berners-Lee die ersten Webkomponenten im Jahr 1990 entwickelte. Berners-Lee arbeitete zu diesem Zeitpunkt bei *CERN* in der Schweiz. Er entwickelte ein System zur Verwaltung von unternehmensinternen Information mittels des ersten Webbrowsers. Dieser war in der Lage, HTML-Dokumente von einem durch Berners-Lee entwickelten Webserver abzurufen und darzustellen (Berners-Lee 1990).
In der Geschichte des Internets finden sich einige Trends und Entwicklungen, welche als Meilensteine gesehen werden. Diese teilen das Internet historisch in Web 1.0, Web 2.0, Web 3.0 und Web 4.0 auf (Kollmann 2020: 133).
Den Beginn der Geschichte des Internets bildet das Web 1.0. Dieses basiert überwiegend auf den Ideen von Berners-Lee und wird durch die Etablierung des Internets in der Gesellschaft erweitert. Somit besteht das Web 1.0 lediglich aus statischen HTML-Seiten, welche den Nutzenden in einem Webbrowser angezeigt wird. Der dominierende Webbrowser der Neunzigerjahre ist der *Netscape Navigator* des Unternehmens *Netscape Communications* (O'Reilly 2005). Besonders für das Web 1.0 ist die binäre Rollenverteilung der Nutzenden, wie Kollmann beschreibt:

> „Zum einen gab es aktive Ersteller von Web-Inhalten, die, teils kommerziell, teils privat, Informationen einstellten und publizier-

> ten. Zum anderen gab es passive Konsumenten, die sich lediglich die bereitgestellten Inhalte ansehen konnten und auch gar keine andere Option hatten, als die Informationen zu empfangen und zu konsumieren“ (Kollmann 2020: 134).

Dieser Passivität der Konsumierenden sind sich die Unternehmen der Zeit des Web 1.0 ebenfalls bewusst. So zeigen sich im Web 1.0 die ersten Ansätze von ausgefeilten E-Commerce-Strategien, welche darauf abzielen, Produkte und Dienstleistungen auf diesem neuen Markt zu vertreiben (Kollmann und Lomberg 2010: 1204).

Das auf das Web 1.0 folgende Web 2.0 entstand um 2005 und markierte somit die Zeit, als die dot-com-Blase geplatzt ist. Diese stetige Wende ist aus einer Konferenz zwischen den Unternehmen *O'Reilly* und *MediaLive International*. Die Idee einer nächsten Evolutionsstufe des Internets kam den Unternehmen bei einem Brainstorming. Dieses Brainstorming brachte letztendlich die *Web 2.0 Conference* hervor. Hierbei muss Erwähnung finden, dass Unternehmen das Buzzword *Web 2.0* als Marketing-Element missbrauchen. Das erschwert die Einordnung des Web 2.0 umso mehr, da viele dieser Unternehmen nichts mit den Definitionsansätzen der *Web 2.0 Conference* gemein haben. Die *Web 2.0 Conference* versucht sich an einer Definition über zentrale Aspekte dieser neuen Iterationsstufe des Internets. Dazu gehören die Ansätze *Web as a Platform* und *User-Generated Content*. Die Rolle der passiven Konsumierenden des Web 1.0 veränderte sich demnach zu den aktiven Teilnehmenden des Web 2.0. Erste soziale Medien ermöglichen eine Interaktion mit anderen Nutzenden, Bewertungen auf E-Commerce-Seiten verschaffen Käuferinnen und Käufern eine Stimme und digitale Enzyklopädien laden zum Teilen des eigenen Wissens ein. Zentrale Plattformen des Web 2.0 sind somit *Facebook*, *eBay* und *Wikipedia*. Anbieter einer Rich User Experience, vor allem *Google*, lösen die Riesen des Web 1.0, beispielsweise *Netscape*, ab (O'Reilly 2005). Dabei spielen die sieben Grundprinzipien des Web 2.0 eine zentrale Rolle: Globale Vernetzung, Kollektive Intelligenz, Datengetriebene Plattformen, Perpetual Beta, Leichtgewichtige Architekturen, Geräteunabhängigkeit und Reichhaltige Oberfläche (Kollmann und Häsel 2007, zitiert nach Kollmann 2020: 137).

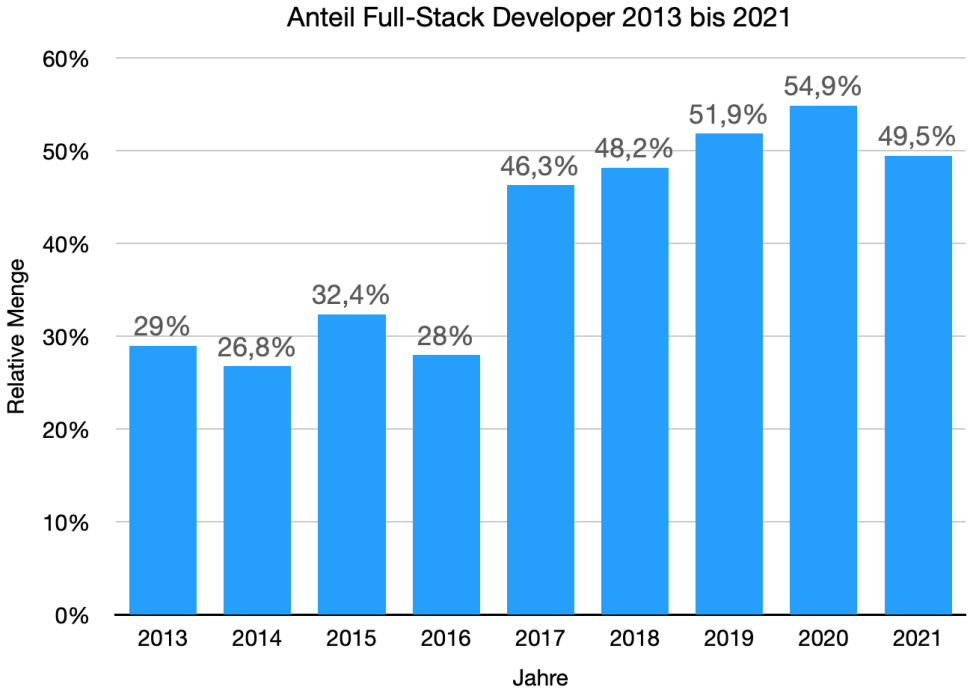
Die Fortführung, der Ausbau und die allzeitliche Zugänglichkeit des Web 2.0

machen das Internet zu der modernen Infosphere, die Floridi 2010 beschrieb. Durch die Möglichkeiten, die Entwicklerinnen und Entwicklern gegeben werden, ist es mit wenig Aufwand möglich, ganze Anwendungen über einen Webbrowser zugänglich zu machen. Insbesondere Anwendungen, welche ursprünglich für native Systeme entwickelt wurden, finden ihren Weg in die Cloud. Welche Vor- und Nachteile dieser Trend hat und ob eine Remigration zu nativen Systemen unter Verwendung moderner Technologien Sinn ergibt, wird in dieser Arbeit näher beleuchtet.

1.2 Problemstellung

Die Entwicklung von Anwendungen für das Web wird eine immer beliebtere Alternative zu nativen Anwendungen, wie beispielsweise installierbare Desktop-Anwendungen. Deutlich erkennbar wird der Trend bei einem Vergleich der *stackoverflow Developer Surveys* aus den Jahren 2015 bis 2021. Betrachtet man nun die Antworten *Full-Stack Developer* und trägt diese als Funktion der Jahre auf, lässt sich folgender Graph erkennen:

Abbildung 1: Anteil der Full-Stack Entwicklerinnen und Entwickler von 2013 bis 2021



Der Graph zeigt den Anteil von Full-Stack Entwicklerinnen und Entwicklern an der Gesamtmenge (siehe Anhang 1.1) der Antworten der jährlichen *stackoverflow Developer Survey* (*2015 Developer Survey* o.D.; *Developer Survey Results 2016* o.D.; *Developer Survey Results 2017* o.D.; *Developer Survey Results 2018* o.D.; *Developer Survey Results 2019* o.D.; *2020 Developer Survey* o.D.; *2021 Developer Survey* o.D.).

Der Anteil an Full-Stack Entwicklerinnen und Entwicklern nahm im Jahr 2017 stark zu und erreichte einen relativen Wert von 46,3 %. Der Anteil stieg weiter und erreichte im Jahr 2020 den Höchststand mit 54,9 %. Im Folgejahr viel der Wert gering. Mit einer zeitweise absoluten Mehrheit von Web-Developern unter den Teilnehmenden der Umfrage lässt sich eine Beliebtheit und Relevanz dieser Technologie schlussfolgern. Die steigende Anzahl zeigt auch, dass Web-Anwendungen über die Jahre an Relevanz gewonnen haben. Hier sei jedoch zu erwähnen, dass Full-Stack Developer nur einen Anteil der Teilnehmenden abbildet, die mit Webtechnologien arbeiten. Berufsbezeichnungen wie Front-End Developer oder Back-End Developer gehören ebenfalls zu den Webentwicklerinnen und -entwicklern, finden jedoch zur Minimierung der Komplexität hier keine Beachtung.

Webanwendungen bieten zahlreiche Vorteile gegenüber konventionellen nativen Anwendungen, sowohl aus Sicht der Entwickelnden als auch aus Nutzenden-Sicht. Hier sei beispielsweise die universelle Erreichbarkeit von jedem Endgerät zu erwähnen. Jedoch müssen ebenfalls die Nachteile bedacht werden, sollte sich für die Entwicklung von Software als Webanwendung entschieden werden. Hier kann als Beispiel die Vielzahl von Webbrowsern mit uneinheitlichen Standards erwähnt werden. Entwicklerinnen und Entwickler müssen sich über die Implikationen der Wahl zwischen nativen Anwendungen und Webanwendungen im Klaren sein, um den Entwicklungsaufwand gering und die Nutzungserfahrung positiv zu gestalten. Welche Aspekte Front-End Developer bei dieser Wahl beachten müssen und welche Lösungen moderne Technologien für dieses Problem bieten, wird in dieser Arbeit aufgezeigt. Zentrale Betrachtung findet dabei der Gedanke einer Remigration von Webanwendungen zu nativen Anwendungen vor dem Hintergrund der großen Zahl an Webentwicklerinnen und -entwicklern.

1.3 Zielsetzung

Ziel dieser Arbeit ist die Dokumentation der Möglichkeiten, die sich Entwickelnden von Webanwendungen bieten, sollten sie eine Migration zu nativen Anwendungen anstreben. Abschließend soll der Versuch einer Vorhersage als begründete Vermutung getätigt werden, ob sich native Anwendungen im Anbetracht moderner Möglichkeiten im Vergleich zu Webanwendungen zukünftig durchsetzen werden.

1.4 Aufbau und Vorgehensweise

Um die Möglichkeiten der Entwicklung von Anwendungen bestmöglich zu vergleichen, werden zunächst die Browser mit den größten Marktanteilen verglichen. Die Betrachtung findet dabei sowohl aus Sicht der Nutzerinnen und Nutzer als auch aus der von Entwickelnden statt. Dabei wird ein besonderes Augenmerk auf die Unterschiede der Technologien gelegt. Darauf folgend werden die Betriebssysteme und ihre Eigenheiten verglichen, für welche native Software entwickelt werden soll. Auch hier werden die Marktführer der Betriebssysteme für Desktop und Smartphone beziehungsweise Tablet verglichen. Als letzter Bestandteil des Literature Reviews werden moderne Cross-Plattform-Technologien verglichen. Hierbei werden die Stärken und Schwä-

chen der Lösungen verglichen, wie auch die Programmiersprache vor dem Hintergrund eines Web-Developers.

Im Fokus des Praxisteils dieser Arbeit steht der Vergleich der Technologien aus praktischer Sicht. Hierzu werden mithilfe der betrachteten Technologien Anwendungen erstellt, welche vorab definierte Anforderungen erfüllen. Die Ausführungen werden daraufhin anhand der Entwicklungserfahrung verglichen. Der praktische Teil erfolgt dabei vor dem Hintergrund eines Web-Developers.

2 Möglichkeiten zur Entwicklung von Anwendungen

2.1 Webanwendungen

Die steigende Zahl der Full-Stack Entwickelnden der letzten Jahre zeigt den Bedarf nach Webanwendungen aus Sicht der Unternehmen (*2020 Developer Survey* o.D.). Aus diesem Trend lässt sich die Vermutung ableiten, dass die Relevanz von Webanwendungen für die Nutzenden und Kunden dieser Unternehmen ebenfalls zunimmt. Diese Vermutung wird dadurch bestätigt, dass die zwanzig am häufigsten besuchten Websites aus November 2021 Webanwendungen sind (Clement 2022). Beim Betrachten derartiger Statistiken stellt sich die Frage, ab wann eine Website eine Webanwendung ist. Zur Klärung folgen nun Definitionen der gängigen Begriffe für diese Arbeit:

> **Website**: Eine Website ist eine Sammlung von aufrufbaren Webseiten. Sie bildet einen Internetauftritt ab.
>
> **Webseite**: Eine Webseite ist ein einzeln aufrufbarer Bestandteil einer Website und bildet einen Zustand einer Ansicht im Webbrowser ab.

Eine Webanwendung ist eine besondere Form einer Website. Eine Website ist im ursprünglichen Ansatz eine Sammlung von HTML-Dokumenten. Diese werden von einem Server über das HTTP-Protokoll an einen Client, meist ein Webbrowser, übertragen. So besteht hier nur eine Richtung des Datenaustauschs: von dem Server zum Client. Eine Webanwendung setzt einen Server voraus, welcher Businesslogik besitzt. Dazu werden Schnittstellen, sogenannte APIs, von dem Client angesprochen. Die APIs eines Webservers ermöglichen eine bidirektionale Kommunikation. So kann eine nutzende Person Daten von dem Server empfangen und an diesen übermitteln. Der Server

verarbeitet die Daten und bietet dem Nutzenden eine Webseite mit dynamischem Inhalt an.
Praktische Beispiele von Webanwendungen sind jederzeit im Internet einsehbar, so auch die eingangs erwähnten meistbesuchten Websites im November 2021. Diese umfassen *google.com* auf dem ersten Platz mit 45,41 Milliarden monatlichen Aufrufen, gefolgt von *youtube.com* und *facebook.com* (Clement 2022).

Ein Frontend einer Webanwendung bedient sich den drei grundsätzlichen Programmier- und Markup-Sprachen des Webs: *HTML*, *CSS* und *JavaScript*. Es ist möglich, jedoch ineffizient, eine Webanwendung nur mit diesen Technologien zu entwicklen. Moderne Frameworks und Libraries vereinfachen Probleme und Herausforderungen und sorgen so für eine angenehmere Entwicklungserfahrung und eine höhere Entwicklungsgeschwindigkeit. Zu den bekanntesten Technologien gehören unter anderem *React.js*, *Angular* und *Vue.js* (Clement 2022). Webframeworks und -libraries vereinfachen den Aufwand der Entwicklung durch die zentrale Nutzung von *JavaScript* und *TypeScript*, wobei *CSS* und *HTML* durch technologiespezifische Aspekte erweitert werden.
Ziel der Nutzung von Webframeworks und -libraries ist eine einfache Kommunikation mit einem Backend oder anderen APIs, die Abdeckung einer Vielzahl von Browsern und letztendlich das Rendern von HTML, CSS und clientseitigem JavaScript im Browser der nutzenden Person. Welche Vor- und Nachteile Webanwendungen verglichen mit nativen Technologien mit sich bringen, wird im Folgenden erläutert.

2.1.1 Vor- und Nachteile der Entwicklung von Webanwendungen

Jobe (2013: 27) begründet den rapiden Fortschritt von Webanwendungen mit der Etablierung der Technologie *HTML5*. *HTML5* und die verwandten Technologien *CSS3* und *JavaScript* sorgen dafür, dass Webapps mit nativen Anwendungen in den Punkten Funktionalität, Design, Interaktion und Einsatz von Multimedia konkurrieren können. Als besondere Vorteile erwähnt Jobe (2013: 28) die Update-Frequenz und die Entwicklung als Ganzes. Der Autor sieht in der Monetarisierung, verglichen mit nativen Anwendungen, Nachteile, da für das Web keine einheitlichen Monetarisierungsstrategien existieren. Da

sich dieser Aspekt seit der Veröffentlichung der Arbeit jedoch weiterentwickelt hat, wird dieser Punkt ebenfalls betrachtet.
Nachteile sieht Jobe (2013: 28) in den Punkten Schaffen und Konsumieren von Inhalten, User Experience, Performanz und Funktionalität. Hierbei ist auffällig, dass die Anzahl der Nachteile überwiegt. Wie kritisch diese Nachteile jedoch sind, und ob die Vorteile dennoch überwiegen, wird im Folgenden unter modernen Gesichtspunkten erläutert.

2.1.1.1 Update-Frequenz

Jobe (2013: 28) gibt die Frequenz, mit welcher Updates für eine Webanwendung geliefert werden nicht direkt als Vorteil an, sondern vergleicht lediglich den formalen Update-Vorgang von nativen Programmen über beispielsweise App-Stores mit dem informalen Update-Möglichkeiten von Webanwendungen. Bei nativen Anwendungen muss in diesem Kontext Erwähnung finden, dass der Nutzerin oder dem Nutzer in der Regel die Entscheidung überlassen wird, ob die installierte Software aktualisiert werden soll. Diese Wahl impliziert jedoch die Gefahr, Sicherheitsupdates zu verpassen. Aus Sicht eines Entwicklenden sollte darauf geachtet werden, dass der Nutzende auf wichtige Sicherheitsupdates aufmerksam gemacht wird. Sollte sich für eine native Softwarelösung basierend auf App-Stores entschieden werden, besteht die Möglichkeit, einen Changelog zu führen und so die Nutzenden vor einer Aktualisierung über Änderungen zu informieren.
Webanwendungen hingegen können jederzeit aktualisiert werden, ohne dass Nutzende informiert oder um Genehmigung gebeten werden müssen. Vorteile dieses Ansatzes sind beispielsweise die User Experience einer stets aktuellen Software, es werden keine Daten heruntergeladen und installiert, was abhängig von der Bandbreite und der Speichergeschwindigkeit längere Zeit in Anspruch nehmen kann, und zuletzt profitieren Nutzenden schnell von Sicherheitsupdates. Bei äußerst kritischen Problemen der Software, kann der Zugang zu den Webservern temporär gesperrt werden während Wartungsarbeiten durchgeführt werden. Auf diese Weise kann sichergegangen werden, dass in dem Moment keine Person Zugriff auf die Software hat.

2.1.1.2 Entwicklung

Wie bereits eingangs erwähnt, bilden die Standards *HTML5*, *CSS3* und *JavaScript* die Grundlage der modernen Webentwicklung. Sollte sich demnach für die Entwicklung der Anwendung als Webapp entschieden werden, sollten die Entwickelnden diese Technologien beherrschen. Für besondere Anwendungsfälle pflegt das *World Wide Web Consortium*, kurz W3C, eine Liste von verfügbaren Standards und Technologien auf dem Weg zur Standardisierung (*STANDARDS* o.D.). Im Folgenden wird eine Auswahl von anwendungsbezogenen Webtechnologien gezeigt, welche die Erfahrungen für Nutzende und Entwickelnde verbessern können. Diese sind nach ihren Status durch die W3C aufgeteilt:

Tabelle 1: Ausgewählte Standards und zugehörige Status der W3C

W3C Recommendations	**Proposed Recommendations**	**Candidate Recommendations**	**Working Draft**
MathML	Payment Request API	WebXR Device API	Picture-in-Picture
SVG	Geolocation API	Media Capture and Streams	Web GPU
Server-Sent Events		Accelerometer	Geolocation Sensor
HTML5 Web Messaging		Gyroscope	

Die aufgelisteten Technologien haben sehr spezielle Anwendungsgebiete. *MathML* ist beispielsweise eine Markup-Sprache zum Erstellen mathematischer Formeln, etwa wie mit LaTeX. Im Kontext dieser Arbeit sind jedoch native APIs, wie *Accelerometer* oder *Gyroscope*, von besonderer Wichtigkeit, da sie die Fähigkeiten und Einsatzgebiete von Webanwendungen mit denen von nativen Anwendungen verschwimmen lassen (*STANDARDS* o.D.).

Wie die Standards und die als Standard antizipierten Technologien des W3C zeigen, werden Merkmale nativer Software ihren Weg in den Webbrowser

finden. Die Abstraktion dieser über HTML, CSS und JavaScript vereinfacht dabei die plattform- und browserunabhängige Entwicklung der gewünschten Software, vorausgesetzt die Browser unterstützen den Standard.

Literaturverzeichnis

2015 Developer Survey (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2015#work-occupation [abgerufen am 12. 04. 2022].

2020 Developer Survey (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2020#developer-roles [abgerufen am 12. 04. 2022].

2021 Developer Survey (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2021#section-developer-roles-developer-type [abgerufen am 12. 04. 2022].

Berners-Lee, T. (1990): *Information Management: A Proposal*, CERN, [online] https://www.w3.org/History/1989/proposal.html [abgerufen am 08. 04. 2022].

Clement, J. (2022): *Most popular websites worldwide as of November 2021, by total visits*, statista, [online] https://www.statista.com/statistics/1201880/most-visited-websites-worldwide/ [abgerufen am 22. 04. 2022].

Developer Survey Results 2016 (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2016#developer-profile-developer-occupations [abgerufen am 12. 04. 2022].

Developer Survey Results 2017 (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2017#developer-profile-_-developer-type [abgerufen am 12. 04. 2022].

Developer Survey Results 2018 (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2018#developer-profile-_-developer-type [abgerufen am 12. 04. 2022].

Developer Survey Results 2019 (o.D.): stackoverflow, [online] https://insights.stackoverflow.com/survey/2019#developer-profile-_-developer-type [abgerufen am 12. 04. 2022].

Floridi, L. (2010): *Information - A Very Short Introduction*, New York: Oxford University Press Inc.

Jobe, W. (2013): Native Apps vs. Mobile Web Apps, in: Bd. 7, Nr. 4, S. 6, [abgerufen am 28. 04. 2022].

Kleinrock, L. (2010): An Early History of the Internet, in: *History of Communications*, Bd. 48, Nr. 8, S. 26–36, [online] https://ieeexplore.ieee.org/document/5534584 [abgerufen am 08. 04. 2022].

Kollmann, T. (2020): *Handbuch Digitale Wirtschaft*, Wiesbaden: Springer Gabler.

Kollmann, T. und Lomberg, C. (2010): Web 1.0, Web 2.0 and Web 3.0: The Development of E-Business, in: *Encyclopedia of E-Business Development and Management in the Global Economy*, Bd. 1, S. 1203–1210, [online] https://www.igi-global.com/book/encyclopedia-business-development-management-global/37278 [abgerufen am 09. 04. 2022].

O'Reilly, T. (2005): *What Is Web 2.0*, O'Reilly, [online] https://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html [abgerufen am 09. 04. 2022].

STANDARDS (o.D.): W3C, [online] https://www.w3.org/standards/ [abgerufen am 02. 04. 2022].

Eidesstattliche Erklärung

Ich erkläre an Eides Statt, dass ich meine Hausarbeit „Digitalisierung eines analogen Prozesses unter Anwendung moderner UI- und UX-Designmethoden - am Beispiel einer webbasierten Anwendung zur Erstellung von Rechnungen“ selbstständig und ohne fremde Hilfe angefertigt habe und dass ich alle von anderen Autoren wörtlich übernommenen Stellen wie auch die sich an Gedankengänge anderer Autoren eng anlehnenden Ausführungen meiner Arbeit besonders gekennzeichnet und die Quelle nach den mir von der Dualen Hochschule Schleswig-Holstein angegebenen Richtlinien zitiert habe.

Kiel, den 13.01.2022

_____

Fabian Reitz

Anhang

Anhang 1: Tabellen

Anhang 1.1: stackoverflow Developer Surveys 2015 bis 2021

Tabelle 2: Anzahl der Antworten der stackoverflow Developer Surveys 2015 bis 2021

Jahr	Absolute Anzahl aller Antworten	Relativer Anteil Full-Stack Developer
2013	8.218 Antworten	29 %
2014	7.346 Antworten	26,8 %
2015	22.148 Antworten	32,4 %
2016	49.525 Antworten	28 %
2017	36.125 Antworten	46,3 %
2018	92.098 Antworten	48,2 %
2019	81.335 Antworten	51,9 %
2020	49.370 Antworten	54,9 %
2021	66.484 Antworten	49,5 %

Die Tabelle zeigt die Anzahl aller Antworten pro Jahr der *stackoverflow Developer Survey*, sowie den relativen Anteil von Full-Stack Developern an diesen Antworten (*2015 Developer Survey* o.D.; *Developer Survey Results 2016* o.D.; *Developer Survey Results 2017* o.D.; *Developer Survey Results 2018* o.D.; *Developer Survey Results 2019* o.D.; *2020 Developer Survey* o.D.; *2021 Developer Survey* o.D.).