FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Fachochschule Aachen, Campus Jülich

Fachbereich 09 - Medizintechnik und Technomathematik
Studiengang Angewandte Mathematik und Informatik

Seminararbeit

Evaluation von Form.io im Anwendungsfall einer Urkundenanforderung aus dem Geburtenregister

Autor:
Fabian Stelzen, 3272251

Betreuer:
Prof. Dr. Philipp Rohde
Pascal Pfeiffer

Aachen, 18. Dezember 2022

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die Seminararbeit mit dem Thema

Evaluation von Form.io im Anwendungsfall einer Urkundenanforderung aus dem Geburtenregister

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, alle Ausführungen, die anderen Schriften wörtlich oder sinngemäß entnommen wurden, kenntlich gemacht sind und die Arbeit in gleicher oder ähnlicher Fassung noch nicht Bestandteil einer Studien- oder Prüfungsleistung war.

Ich verpflichte mich, ein Exemplar der Seminararbeit fünf Jahre aufzubewahren und auf Verlangen dem Prüfungsamt des Fachbereiches Medizintechnik und Technomathematik auszuhändigen.

Name: Fabian Stelzer

Aachen, den 7.12.2022

Unterschrift der Studentin / des Studenten

Stelz

Zusammenfassung

Das Ziel dieser Seminararbeit war es, Form.io im Anwedungsfall einer Urkundenanforderung aus dem Geburtenregister zu evaluieren. Form.io ist dabei eine Form and Data Management Platform für progressive Web-Applikationen. Form.io verfügt über eine Vielzahl an sehr hilfreichen Werkzeugen zur Entwicklung eines Webformulars. Zuerst wurde eine Produktbeschreibung angefertigt, welche alle funktionalen und nicht-funktionalen Anforderungen, welche die Anwendung erfüllen können sollte, um für die Urkundenanforderung aus dem Geburtenregister eingesetzt werden zu können, sammelte. Hierbei wurde sich auf die für das Front-End relevanten Anforderungen konzentriert. Das Formular benötigt eine Liste an wichtigen typischen Formularkomponenten, die aber gut konfigurierbar sein mussten, sowie die Möglichkeit das Formular in mehrere frei navigierbare Stufen zu unterteilen. Es sollte sich dabei um ein dynamisches Formular handeln was verschiedenste Validierungsoptionen anbietet und bei fehlerhaften Eingaben dem Benutzer genau zeigt, was er ändern muss. In der Analyse wurde überprüft, wie eine Anwendung die mit Form.io entwickelt wurde, diese gestellten Anforderungen umsetzten kann. Aus der Analyse ging hervor, dass sich Form.io für jegliche Webanwendung mit Online-Formularen gut eignen würde, da es alle gestellten Anforderungen erfüllt und dabei auch noch Sicherheit, Benutzerfreundlichkeit und Wartbarkeit garantiert. Form.io sollte als sehr hilfreiches Werkzeug für die Entwicklung von Webformularen gesehen werden und wird in Zukunft noch viel Verwendung finden.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die Digitalisierung wird ein immer wichtigeres Thema in Deutschland. Standesämter sollen so viele Verwaltungsleistungen wie möglich in Zukunft digital anbieten. Darunter fällt auch die Urkundenanforderung aus dem Geburtenregister.
Eine Geburtsurkunde beinhaltet eine vom zuständigen Standesamt zugewiesene Nummer, sowie den vollen Namen, den Geburtstag und Geburtsort, die Konfession und Informationen über die Religionszugehörigkeit und die Berufe der Elternteile. Die Geburtsurkunde wird für den ersten Personalausweis, sowie für eine Versicherung und ein Bankkonto für das Kind benötigt. Selbst als Erwachsender wird die Geburtsurkunde für Elterngeld, Kindergeld und für eine Eheschließung benötigt.
Eine Geburtsurkunde kann am zuständigen Standesamt per Post, Fax oder Online beantragt werden. Für die Beantragung benötigt das Standesamt Informationen über den Antragsteller, die Beurkundete Person, sowie die gewünschte Anzahl und Art der Urkunden.
Um eine Urkundenanforderung online durchführen zu können müssen erst wichtige Informationen mithilfe eines Online-Formulars erfasst werden. Die Entwicklung eines Online-Formulars kann dabei sehr komplex werden und in der Wartbarkeit aufwendig sein, weshalb sich der Einsatz von Form.io anbietet.

1.2 Ziel der Arbeit

In dieser Seminararbeit werde ich evaluieren ob Form.io sich zur Umsetzung des Online-Formulars zur Urkundenanforderung aus dem Geburtenregister eignet. Aus dieser Evaluierung wird dann festgestellt, ob es sich für die regio iT lohnt, Form.io für weitere Projekte ähnlicher Art einzusetzen.

1.3 Gliederung

Zunächst wird eine Projektbeschreibung aufgestellt, um die zu erfüllenden Kriterien festzulegen. Danach wird eine Analyse von Form.io durchgeführt, aus der hervorgehen soll, wie viele der gestellten Anforderungen von Form.io erfüllt werden können. Zum Schluss gibt es ein Fazit, welches die Ergebnisse der Seminararbeit zusammenfasst und kritisch reflektiert, sowie einen Ausblick, welcher noch offene Fragen adressiert.

2 Projektbeschreibung

2.1 Motivation

Die Projektbeschreibung dient der Zusammenfassung aller Anforderungen an die Anwendung, welche die Urkundenanforderung aus dem Geburtenregister umsetzen soll. Anforderungen bilden das Grundgerüst zur Umsetzung eines Projektes. Die Sammlung dieser Anforderungen ermöglicht es, durch eine Analyse von Form.io zu prüfen, ob Form.io genug Abnahmekriterien erfüllt, um für die Entwicklung der Anwendung eingesetzt zu werden.
Es wird dabei unterschieden zwischen funktionalen- und nicht-funktionalen Anforderungen. Funktionale Anforderungen legen fest, was explizit von der Anwendung umgesetzt werden können soll. Nicht-funktionale Anforderungen beschreiben dabei, in welcher Qualität das System die gewünschten Leistungen erbringen kann und welche Randbedingungen eingehalten werden müssen.
In dieser Seminararbeit wird aufgrund der Komplexität einer solchen Anwendung Form.io eher auf Front-End Anforderungen getestet. Front-End bezeichnet dabei die Präsentationsebene einer Anwendung und ist in diesem Fall das Online-Formular, mit dem der Benutzer direkt interagiert. Meine Erfahrungen in der Entwicklung des Front-Ends einer solchen Anwendung, sowie die Betrachtung von Webanwendungen die bereits von Städten zur Urkundenanforderung aus dem Geburtenregister verwendet werden, bilden die Grundlage für die kommenden Anforderungen.

2.2 Funktionale Anforderungen

2.2.1 Formularkomponenten

Ein Formular besteht aus den verschiedensten Komponenten, welche die Aufnahme von Informationen des Benutzers ermöglichen. Im folgenden werde alle Komponenten, die für ein gutes Formular zur Urkundenanforderung aus dem Geburtenregister benötigt werden aufgelistet und erläutert.

Texteingabe

Die Texteingabe ist der essenziellste Bestandteil eines Formulars. Ein Großteil der meisten Formulare besteht überwiegend aus simplen Texteingabefeldern. Dabei wird zwischen den üblichen Textfeldern und den Textbereichen unterschieden.
Textfelder werden für kurze Eingaben genutzt, wie zum Beispiel für den Vornamen einer Person. Im Gegenzug dazu sind Textbereiche für längere Eingaben über mehreren Zeilen

geeignet.
Für Textfelder und Textbereiche sollte es auch grundlegende Konfigurationsmöglichkeiten geben. Zum Beispiel sollte man ein Textfeld darauf einstellen können nur Zahlen zuzulassen. Einem Textbereich sollte eine maximale Anzahl Zeichen zugewiesen werden können.

Auswahlliste

Wenn der Benutzer aus einer begrenzen Anzahl an vordefinierten Antwortmöglichkeiten wählen soll, dann lohnt es sich, anstatt eines normalen Textfeldes, Auswahllisten einzusetzen. Auswahllisten bieten alle möglichen Antworten in Form einer normalerweise aufklappbaren Liste an, wovon dann nur noch die gewünschte Antwort selektiert werden soll. Es sollte die Option geben, eine Antwortmöglichkeit automatisch schon ausgewählt zu haben.

Datei-Upload

Bei manchen Formularen reichen einfache Texteingaben nicht aus, sondern man braucht sogar ganze Dokumente. Grund dafür könnte sein, dass die Nutzung des Online-Formulars ein besonderes Berechtigungsdokument benötigt. Üblicherweise öffnet sich beim Datei-Upload der Explorer, wonach dann der Benutzer zu den relevanten Dokumenten navigiert, um diese daraufhin hochzuladen. Dabei werden üblicherweise nur Ordner und Dateien mit dem vom Formular vorgegebenen Dateitypen beim Explorer angezeigt.

Radio-Button

Radio-Button ähneln der Auswahlliste. Sie sind kleine Buttons mit Testbeschreibungen, die bei Betätigung die Farbe ändern. Radio-Buttons lohnen sich eher bei der Auswahl von wenigen verschiedenen Optionen, da sie nicht so platzsparend sind wie die aufklappbaren Auswahllisten.

Checkbox

Die Checkbox, ist ein einfacher Kasten, welcher durch einen Klick mit der Maus sich ankreuzen lässt. Das Kreuz kann jederzeit durch einen weiteren Klick wieder entfernt werden. Normalerweise wird die Checkbox zur Bestätigung einer Aussage genutzt.

Button

Buttons werden nicht zur Erfassung von Eingabedaten genutzt, sondern für die Ausführung einer vordefinierten Aktion verwendet. Sie sind essenziell für jedes Formular, da sie für die Navigierbarkeit und Einreichung eines Formulars zuständig sind.

2.2.2 Mehrstufigkeit

Mit der Mehrstufigkeit ist die Unterteilung des Formulars in mehrere separate Schritte gemeint. Eine Unterteilung des Formulars bringt mehrere Vorteile mit sich.
Man kann durch eine Unterteilung eine große Menge an Information in kleineren wohlsortierten Schritten abfragen. Gerade bei einer hohen Anzahl an Eingabefeldern hilft diese Unterteilung dabei, dass das Formular keine einschüchternde Wirkung auf den Benutzern hat.
Besonders bei dem Formular für die Urkundenanforderung aus dem Geburtenregister bietet sich eine Unterteilung an. Es werden Informationen über die antragstellende Person, die beurkundete Person, die Anzahl und Art der Urkunden, sowie die Kenntnisnahme des Datenschutzhinweises benötigt. Das Formular kann also zum Beispiel in die folgenden Schritte unterteilt werden.

Antragstellende Person

In dem ersten Schritt sollen Informationen über die Person, die das Formular gerade ausfüllt, gesammelt werden. Es wird nach der Anrede, dem Titel, dem vollen Namen, dem Geburtsdatum, der vollen Anschrift, sowie den Kontaktdaten und der Berechtigung für die Urkundenanforderung gefragt.

Beurkundete Person

Für diesen Schritt wird der volle Name, mit Geburtsname, dem Geburtsdatum und dem Geburtsort abgefragt. Die beurkundete Person ist dabei die Person, dessen Geburtsurkunde der Antragsteller beantragen möchte. Antragsteller und die beurkundete Person können dabei auch die gleiche Person sein.
Dieser Schritt zeigt, dass sich eine Unterteilung in Schritte lohnt, wenn man bedenkt, dass insgesamt in dem Formular mehrmals identische personenbezogene Daten, wie zum Beispiel der volle Namen oder das Geburtsdatum, abgefragt werden müssen.

Anzahl und Art der Urkunden

Zuerst wird in diesem Schritt der Verwendungszweck für die Urkunde gewählt. Daraufhin kann man aus einer Sammlung an verschiedenen Urkundenarten, wie zum Beispiel dem Standardformat(DIN A4) oder Stammbuchformat(DIN A5), wählen, wie viele Urkunden man von welcher Art zugeschickt bekommen möchte.

Abschluss

Die Kenntnisnahme des Datenschutzhinweises bietet sich für den letzten Schritt an. Im Abschluss-Schritt kann der Benutzer seine meist kostenpflichtige Bestellung beantragen. Es werden nochmal viele der bisher eingegeben Daten zusammengefasst und der Preis für die Urkunden berechnet. Dieser hängt selbstverständlich von den Angaben im vorherigen Schritt ab.

2.2.3 Navigierbarkeit

Bereits ausgefüllte Schritte sollten auch im Nachhinein noch editierbar sein. Das heißt, dass es eine Möglichkeit geben muss von jedem Schritt des Formulars zurück zu einem vorherigen Schritt zu navigieren, um dort eventuell falsch eingetragene Daten zu korrigieren. Selbstverständlich sollte durch die Rückkehr zu einem vorherigen Schritt keine bereits eingegebenen Informationen verloren gehen.

2.2.4 Dynamisches Formular

Formulare können sich in Format oder Intention sehr ähneln. Um verschiedene Versionen eines Formulars zu erstellen lohnt es sich, das Formular dynamisch umzusetzen. Das heißt, dass Änderungen im Formular in Echtzeit auftreten können. Welche Änderung zu welchem Zeitpunkt umgesetzt werden soll ist von Fall zu Fall unterschiedlich. Im Folgenden gehe ich auf zwei dynamische Änderungen ein, wo das Formular zur Urkundenanforderung aus dem Geburtenregister von dem Aufbau eines dynamischen Formulars profitiert.

Ein- und Ausblendung von Feldern

Es kann sein, dass die Felder des Formulars sich je nach Art des Benutzers ändern sollen. Ein Beispiel dafür könnte sein, dass das Formular für natürliche Personen und juristische Personen zur Verfügung stehen soll. Als natürliche Person gilt jede Privatperson, wobei ein Unternehmen, das eine eigene Rechts- und Geschäftsfähigkeit besitzt als juristische Person gilt. Zu juristischen Personen gehören also Aktiengesellschaften, GmbHs, Genossenschaften, Vereine und Stiftungen. Der einzige Unterschied, der bei der Urkundenanforderung aus dem Geburtenregister für die Personenarten entstehen könnte, ist bei den Informationen zum Antragsteller. Mithilfe eines dynamischen Formulars kann man bestimme Formularkomponenten so ein- und ausblenden, dass je nach der Personenart ein anderes Antragstellerformular entsteht. Im Vergleich zu der natürlichen Personen, würde eine juristische Person anstatt der Felder für die Anrede, des Titels und des vollständigen Namen, Felder für den Firmennamen und den Ansprechpartner bekommen. Die Kontaktinformationen könnten um ein E-Mail-Zusatzfeld erweitert werden. Alle anderen Schritte des Formulars müssten nicht angepasst werden, weshalb sich die Nutzung von einem dynamischen Formular für den Anwendungsfall mehr lohnt, als separate Formulare für die zwei Personenarten zu entwickeln.

Automatische Felderbefüllung

Das automatische Befüllen von Feldern eines Formulars kann einem Benutzer viel Zeit ersparen. Es gibt viele Situationen in dem eine automatische Füllung der Felder gerechtfertigt wäre.
Im ersten Schritt des Formulars werden Informationen über den Antragsteller selbst gesammelt. Dazu gehört das Feld für die Berechtigung zur Urkundenanforderung. Dieses Eingabefeld existiert, da eine Person nicht einfach die Geburtsurkunde einer beliebigen anderen Person ohne Berechtigung beantragen kann. Ein Beispiel dafür wäre die Berechtigung als Antragsteller seine eigene Geburtsurkunde zu beantragen. In diesem Fall wäre der Antragsteller auch

die beurkundete Person. Es wäre unsinnig für den Antragsteller seinen vollständigen Namen und Geburtsdatum doppelt eingeben zu müssen. In einem dynamischen Formular könnte man dafür sorgen, dass diese Felder bei der Auswahl der Berechtigung als beurkundete Person automatisch befüllt werden.
Ein weiterer Einsatzfall für die automatische Felderbefüllung könnte durch die Implementierung der eID entstehen. Die eID ist ein Online-Ausweis, welcher für digitale Dienstleistungen genutzt werden kann. Auf der eID-Karte befindet sich ein Logo für die Online-Ausweisfunktion, eine Dokumentennummer, der vollständige Name, der Tag der Geburt, der letzte Tag der Gültigkeit, eine Zugangsnummer, der Ort der Geburt und die ausstellende Behörde. Wenn das Standesamt mit eID-Karten arbeitet, und man die dort enthaltenen Informationen bei einer Urkundenanforderung aus dem Geburtenregister beachtet, so kann man den vollständigen Namen, sowie den Geburtstag und Geburtsort schon im ersten Schritt des Formulars befüllen. Es würde sogar Sinn machen die Informationen, die an die eID gebunden sind nicht im Formular editierbar zu machen.
Zuletzt ist noch zu Erwähnen, dass im Abschluss-Schritt des Formulars eine Zusammenfassung der bisher erfassten Informationen stehen sollte. Wenn eine Person jedoch noch eine Eingabe von einem vorherigen Schritt editieren möchte, so muss sich selbstverständlich auch die Zusammenfassung dynamisch auf die geänderten Informationen anpassen.

2.2.5 Validierung

Die Validierung der Eingabefelder ist einer der wichtigsten Aspekte eines guten Formulars. Durch die Validierung wird sichergestellt, dass die eingetragen Informationen vordefinierten Regeln entsprechen. Dies soll garantieren, dass die Informationen für den Anwendungszweck brauchbar sind. Im folgenden werde ich auf verschiedene Fälle eingehen, die durch eine gute Validierung behandelt werden sollten.

Pflichtfelder

Nicht jedes Feld eines Formulars ist von essenzieller Bedeutung. Es wird zwischen Pflichtfeldern und optionalen Feldern unterschieden. Pflichtfelder sind dabei, meist durch ein Asterisk gekennzeichnete Felder, die unbedingt ausgefüllt werden müssen.
Für eine Urkundenanforderung aus dem Geburtenregister sind Informationen wie der vollständige Name des Antragstellers essenziell und somit als Pflichtfeld eingestuft. Ob der Benutzer aber einen Doktortitel hat ist hingegen eine nebensächliche Information. Das Formular kann also auch ohne die Befüllung der optionalen Felder eingereicht werden.
Bei dynamischen Formularen ist zu beachten, dass durch bestimmte Aktionen des Nutzers Pflichtfelder entfallen oder hinzugefügt werden können. Bei dem Beispiel mit der Unterteilung der Personenarten würde der Firmenname der juristischen Person also als Pflichtfeld hinzukommen. Die Felder für den Vornamen und Nachnamen würden allerdings wegfallen, obwohl es ursprüngliche Pflichtfelder waren. Die Liste an Pflichtfeldern die überprüft werden muss, ist also variabel.

Reguläre Ausdrücke

Ein regulärer Ausdruck, kurz REGEX, ist eine Zeichenkette die ein Muster beschreibt. Mithilfe von regulären Ausdrücken können Texte auf ein Muster getestet werden. Eingaben mit klar beschriebenen Mustern können durch Nutzung eines regulären Ausdrucks validiert werden. Ein Muster gibt vor wie viele Zeichen, welcher Art, wann vorkommen dürfen. Zum Beispiel haben E-Mails und Telefonnummern sehr klare Muster. Es folgt ein REGEX für die meisten E-Mails.

```
"^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"
```

Gültigkeitsbereiche

Der Gültigkeitsbereich ist meistens vor allem bei der Eingabe von Zahlenwerten relevant. Im Anwendungsfall einer Urkundenanforderung aus dem Geburtenregister ist die Angabe eines Gültigkeitsbereich für das Geburtsdatum der beurkundeten Person besonders wichtig. Das Geburtsdatum darf nämlich nicht mehr als 110 Jahre zurückliegen, da die Geburtsurkunde sonst bereits Teil der öffentlichen Stadtsarchive geworden wäre. Der Gültigkeitsbereich des Geburtsdatums einer beurkundeten Person beträgt also 110 Jahre in die Vergangenheit bis heute.
Bei der Anzahl und Art der Urkunden macht es wenig Sinn negative Zahlen zu erlauben. Ein Gültigkeitsbereich von zwischen 0 bis ungefähr 10 für die Urkundenanzahl mit der Regelung, dass mindestens eine Urkundenart einmal ausgewählt werden muss, würde am meisten Sinn machen.

Datei-Upload

Es kann vorkommen, dass für das Formular zusätzliche Dokumente vom Benutzer gebraucht werden. Zum Beispiel wenn der Antragsteller weder die beurkundete Person selbst, noch ein naher Verwandter der beurkundeten Person ist. Eine Person die trotzdem eine Geburtsurkunde anfordern möchte, braucht dafür also eine besondere Berechtigung, die er mit einem passenden Dokument per Datei-Upload belegen muss.
Zum Beispiel könnten die Eltern eines Kindes dem Antragsteller die Einverständnis dazu gegeben haben. Eine schriftliche Vollmacht, den Personalausweis eines der Elternteile, sowie der Personalausweis der antragstellenden Person, müssten daraufhin hochgeladen werden. Doch ist es wichtig zu validieren ob die Datei des Antragstellers auch bestimmte Regeln erfüllt.
Eine hochgeladene Datei ist nur dann valide wenn die Anzahl der Dateien, sowie die deren Dateigröße in einem vordefinierten Bereich liegen. Auch welche Dateitypen hochgeladen werden sollen kann man begrenzen. Für diesen Anwendungsfall sind hauptsächlich die Dateitypen PDF, JPEG, PNG relevant.

Schrittweise Validierung

In dem mehrstufigen Formular soll der Benutzer vorwärts und rückwärts zwischen den Schritten navigieren können. Wenn der Benutzer sich dazu entscheidet, dass sie etwas in einem

vorherigen Schritt editieren möchte, dann sollte man so lange keinen Schritt nach vorne machen dürfen, bis der Schritt komplett valide ist. Das heißt, dass jedes Feld, dass sich in dem momentanen Schritt des Formulars befindet validiert ist, bevor man wieder Schritte vorwärts tätigen kann. Diese Vorwärtsbegrenzung muss existieren, da die hinteren Schritte auf den Informationen der vorherigen Schritte aufbauen.

2.2.6 Fehler

Wenn der Benutzer eine fehlerhafte Eingabe tätigt, muss dies auf irgendeine Art dem Antragsteller mitgeteilt werden. Der Fehler muss also visuell für den Benutzer dargestellt werden.

Der erste Schritt um den Nutzer bei der Fehlerbehebung zu helfen, ist die typische rote Markierung des fehlerhaften Formularfelds. Hierdurch weiß der Benutzer wenigstens schon mal welches Feld er am Formular überarbeiten muss.
Die Identifizierung des Validierungsfehlers sollte jedoch nicht die Aufgabe des Benutzers sein. Man sollte in einer kleinen Fehlermeldung anzeigen, was geändert werden muss. Ein Feld kann aber auch mehrere Fehler gleichzeitig haben.
Das Geburtsdatum-Feld der beurkundeten Person ist ein Pflichtfeld, mit einem Gültigkeitsbereich der eingehalten werden muss, sowie einem REGEX, welcher das Datumsformat festlegt. In Deutschland wäre das offensichtlich das übliche ”TT.MM.JJJJ”Format.
Es gibt mehrere Wege, diese Fehler für den Benutzer anzuzeigen. Es könnten untereinander alle auftretenden Verletzungen der Validierung in Fehlermeldungen vorkommen oder es wird immer die am meisten relevante Fehlermeldung zuerst angezeigt. Wenn die dann behoben ist, wird der darunterliegende Fehler, falls dieser noch nicht gelöst ist, angezeigt. Im Falle des Datums kann man zuerst dem Nutzer mitteilen, dass es ein Pflichtfeld ist und somit nicht leer sein darf. Diese Fehlermeldung reicht allein aus, da das Muster und der Gültigkeitsbereich sowieso nicht vorhanden sein können bei einem leer-gelassenen Feld. Das würde dann gefolgt werden von der Abfrage des Musters, gefolgt von der Überprüfung des Gültigkeitsbereiches.
Zu beachten ist, dass natürlich bei dem Aufruf des Formulars nicht direkt alle Pflichtfelder rot markiert sein sollten. Es bietet sich an bei dem Versuch durch einen Button in den nächsten Schritt zu wechseln, die Validierung durchzuführen und dann alle Felder mit fehlerhaften Eingaben zu markieren und mit den richtigen Fehlermeldungen auszustatten. Ab diesem Punkt sollte dann die Fehlermeldung in Echtzeit sich auf die Eingabe des Nutzers anpassen.

2.3 Nicht-funktionale Anforderungen

2.3.1 Sicherheit

Sicherheit ist ein sehr wichtiger Teil der nicht-funktionalen Anforderungen. Erst vor einigen Monaten wurde in den Medien über eine große Cyberattacke auf die Industrie- und Handelskammer, kurz IHK, berichtet. Es wurde sämtliche IT-Systeme der IHK angegriffen, was darin resultierte, dass die Systeme sicherheitshalber heruntergefahren werden mussten. Kontakt zur IHK war nur noch telefonisch oder in Person möglich. Nur aufgrund der gut funktionierenden Sicherheitssysteme konnten der Abgriff von Daten verhindert werden. Fälle wie dieser zeigen,

wie wichtig das Thema Sicherheit für alle Bereiche der IT ist.
Im Folgenden werden drei wichtige Aspekte der Sicherheit aufgezählt und erläutert.

Datenschutz

Datenschutz steht für das Recht auf informationelle Selbstbestimmung, also dem Recht selbst über die Preisgabe und Verwendung seiner eigenen personenbezogener Daten zu bestimmen, sowie dem Schutz vor missbräuchlichen Verarbeitung der personenbezogenen Daten. Die Anwendung sollte den Datenaustausch von sensiblen Daten so gut wie möglich sichern.

Sicherheitslücken

Eine Sicherheitslücke beschreibt einen Fehler in einer Soft- oder Hardware, durch die ein Angreifer in ein System eindringen kann. Sie entstehen durch unzureichende Sicherheitsmaßnahmen, sowie durch Programmierfehler. Sicherheitslücken sollten schnellstmöglich aufgefunden werden um dann sofort behoben werden zu können.

Authentifizierung

Authentifizierung ist die Prüfung eines Identitätsnachweises auf seine Authentizität. Wenn man sich auf einer Website mit einem Konto anmelden möchte braucht man meistens einen Benutzernamen mit einem Passwort. Authentisierung würde dabei die Eingabe der Anmeldedaten beschreiben, während mit Authentifizierung die Überprüfung dieser Anmeldedaten gemeint wäre. Die Anwendung sollte sichere Mittel zur Authentifizierung bereitstellen. Idealerweise sollte es noch 2-Faktor-Authentifizierung geben. Selbst wenn eine Person an die Anmeldedaten einer anderen Person kommen würde, so ist durch einen zweiten Authentifizierungsschritt, wie zum Beispiel durch einen Bestätigungscode der an das Handy gesendet wird und bei der Anmeldung zusätzlich eingegeben werden muss, das Konto immer noch sicher.

Autorisierung

Welche Rechte eine Person hat, die sich erfolgreich authentifizieren konnte, ist durch die Autorisierung geregelt. Sie verteilt Privilegien anhand der Rolle, welche die authentifizierte Person besitzt. Zwei authentifizierte Nutzer können komplett verschiedene Rechte haben, da es sich zum Beispiel bei einer Person um einen Administratoren handeln könnte. Die Anwendung sollte Rollen den Benutzern und Rechte den Rollen zuweisen können.

2.3.2 Benutzerfreundlichkeit

Benutzerfreundlichkeit bedeutet im Anwendungsfall einer Urkundenanforderung aus dem Geburtenregister, wie effizient, effektiv die Nutzung und Entwicklung des Formulars ist. Für die Entwicklung ist dabei noch die Dokumentation von Relevanz. Effektivität beschreibt dabei die Genauigkeit mit der ein Ziel erreicht werden konnte. Mit Effizienz ist der Aufwand im Verhältnis mit der Effizienz gemeint.

2.3.3 Wartbarkeit

Wartbarkeit beschreibt die Einfachheit, Modifizierungen an der Anwendung vorzunehmen, um Fehler zu beheben, oder andere Anpassungen an der Anwendung vorzunehmen, Regelungen zu den benötigten Informationen für eine Urkundenanforderung aus dem Geburtenregister könnten sich leicht ändern. In diesem Fall muss eine schnelle und nicht aufwendige Änderung des Formulars möglich sein.

3 Analyse von Form.io

3.1 Motivation

In der Projektbeschreibung wurden viele Anforderungen an eine Webanwendung gesammelt, die mithilfe eines Formulars, es dem Benutzer ermöglichen sollen, eine Urkunde aus dem Geburtenregister des zuständigen Standesamtes anzufordern. Die Analyse soll feststellen, ob eine Anwendung, die mit der Hilfe von Form.io entwickelt wurde, diese Anforderungen erfüllen kann.

3.2 Form.io

Form.io ist eine Form and Data-Management-Platform für progressive Webanwendungen.
Progressive Web Anwendungen, kurz PWA, sollen sich wie native Applikationen anfühlen. Native Applikationen beschreiben dabei normale Handy-Apps, welche normalerweise in Programmiersprachen wie Java oder Kotlin geschrieben werden. PWAs werden hingegen mit typischen Webanwendungsprogrammiersprachen wie JavaScript in Kombination mit HTML und CSS umgesetzt. Dies ermöglicht auch die Nutzung von typischen Front-End-Webapplikationsframeworks wie Angular und React.
Eine PWA kann im Endeffekt über eine Website im Browser und über eine Handy-App nutzbar gemacht werden. Durch Form.io wird eine Plattform zur schnellen Entwicklung komplexer Formulare im Front-End zur Verfügung gestellt, während automatisch eine angepasste API-Plattform im Back-End generiert wird. Diese API-Plattform wird automatisch mit dem Formular verbunden und passt sich dynamisch an das Formular an.
Die schnelle Entwicklungszeit wird durch den simplen ”Drag & Drop” Workflow des Form-Builders ermöglicht. Diese dynamischen Formulare lassen sich direkt und simpel in eine progressive Web-Applikation einbinden. Um die generelle Funktionsweise von Form.io verständlicher zu machen, werde ich im Folgenden den Prozess zur Formularerstellung und Einbettung erläutern, sowie kurz auf die Formiojs-Bibliothek eingehen.

3.2.1 Formularerstellung

Als aller erstes muss ein Projekt auf der Form.io Webseite erstellt werden. Ein Projekt kann aus mehreren Formularen bestehen. Für alle Formulare des Projekts sind dann generelle Einstellungen, Ressourcen und Zugriffsrechte konfigurierbar.
In der Formular-Sektion kann schnell für das Projekt ein neues Formular angelegt werden. Zur Erstellung benötigt man einen Titel, einen Namen und einen API-Pfad.

Titel Der Titel ist der von den Benutzern lesbare Name des Formulars.

Name Der Name ist der Maschinenname des Formulars, Dieser ist relevant für API-Anfragen und der Importierung des Formulars in andere Projekte.

API-Pfad Der API-Pfad wird zur Einbettung in die progressive Webanwendung benötigt.

Nachdem das Formular angelegt wurde kann man mit dem ”Drag & Drop”-Form-Builder verschiedenste Felder zu einem Formular zusammensetzen. Jedes Feld ist dabei mit vielen Konfigurationsmöglichkeiten ausgestattet. Die Konfiguration ist dabei in folgende Abschnitte unterteilt.

Konfigurationsabschnitte

Display In diesem Abschnitt befinden sich überwiegend visuelle Konfigurationsmöglichkeiten, wie zum Beispiel das Label, die Beschreibung und den Platzhaltertext für ein Feld.

Data Die Konfiguration von den Eingabedaten ist durch diesen Abschnitt geregelt. Darunter fällt zum Beispiel die Definition eines Standardwertes oder Eingabeformates, sowie ob die Daten beschützt, also nicht einfach über die API abrufbar sein sollen.

Validation Im Validationsabschnitt kann eingestellt werden, unter welchen Bedingungen das Feld als valide angenommen wird. Die Markierung als Pflichtfeld oder die Definition einer erlaubten maximalen Eingabelänge können zum Beispiel in dieser Sektion konfiguriert werden.

API Im API-Abschnitt werden für die API wichtige Einstellungen vorgenommen. Darunter fällt beispielsweise der Name des Feldes im API-Endpunkt.

Condition Der Condition-Abschnitt ermöglicht es, dem Feld einen Wahrheitswert zuzuweisen. Ob das Feld nun wahr oder falsch ist, hängt von einer selbst definierten Bedingung ab.

Logic Im Logic-Abschnitt werden selbst definierte Aktionen zu selbst definierten Triggern gebaut. Ein Trigger wartet auf die Erfüllung einer Bedingung und führt dann die dementsprechende konfigurierte Aktion aus.

Layout Im Layout-Abschnitt kann eine Map von HTML-Attributen für das jeweilige Feld festgelegt werden. Attribute, die durch Form.io automatisch generiert werden haben jedoch Priorität.

Aktionen

Nachdem man seine Felder eingefügt und konfiguriert hat, müssen noch Aktionen definiert werden. Aktionen beschreiben, was bei der Einreichung eines Formulars genau passieren soll. Zum Beispiel könnte eine automatisch generierte E-Mail an den Benutzer versendet werden.

Launch

Zuletzt kann man über die Launch-Sektion das Formular starten. Dies startet die Form-View, die das Formular aus Benutzersicht anzeigt. Das Formular kann jetzt auf seine Funktionalität geprüft werden. Es wird außerdem ein Link bereitgestellt, mit dem auch andere dieses Formular ausfüllen und einreichen können.

3.2.2 Formulareinbettung

Im "Drag & Drop" Form-Builder gibt es den Embed Abschnitt, welcher eine Anleitung anzeigt, wie das Formular in die Webanwendung eingebettet werden kann. Bei der Projekterstellung hat man angegeben, auf welches Framework das Projekt spezifiziert werden soll. Je nach Framework ändert sich selbstverständlich wie das Formular eingebettet werden muss. Für das Angular Framework sieht die Einbettung zum Beispiel wie folgt aus. Es wird mit dem Node-Package-Manager, kurz npm, das Form.io Module installiert und in dem Root-Module importiert. Bei einer beliebigen Angular-Komponente wird zuletzt dann einfach im HTML-Template mit den Formio-Tag und dem API-Pfad als Attribut das Formular eingebettet. Mit dieser einen Zeile wird das gesamte Formular mit allen Konfigurationen und Funktionen implementiert.

3.2.3 Formiojs Bibliothek

Die Formiojs-Bibliothek kann auch mit dem Node-Package-Manager installiert werden. Die Bibliothek verfügt über einen Form-Renderer und Builder und kann als JavaScript SDK genutzt werden.

Form Renderer

Der Form-Renderer ermöglicht die Instantiierung des Formulars. Dies geschieht durch die createForm-Funktion der Bibliothek. Der Funktion muss als Übergabeparameter der API-Pfad eines bereits existierenden Formulars oder ein passendes JSON-Schema übergeben werden.

Form Builder

Der Form-Builder ist eine Erweiterung des Form-Renderers. Es kann sich ein ”Drag & Drop”-Form-Builder, wie der auf der Form.io Website, eingefügt werden. Dieser ist eigentlich ein glorifizierter JSON-Schema-Builder. Der Vorteil dieser direkten Implementierung in eine Webanwendung ist, dass der FormBuilder gut konfiguriert werden kann. Man kann eigene Felder mit diversen Standardkonfigurationen selbst definieren, die dann in dem ”Drag & Drop”-Form-Builder Interface automatisch eingefügt werden.

3.2.4 JavaScript SDK

Die JavaScript SDK der Formiojs-Bibliothek erlaubt es, direkt in der Web-Anwendung mit der Form.io API zu interagieren. Dabei kann man verschiedenste API-Endpunkte ansteuern. Zum Beispiel kann man, mithilfe der Formular-ID den Endpunkt eines bestimmten Formulars wählen und das Formular überarbeiten, speichern oder löschen lassen. Alternativ nimmt man zum Beispiel den Action-API-Endpunkt um ganz eigene, komplexe Aktionen zu schreiben, die bei der Einreichung des Formulars ausgeführt werden könnten.

3.3 Auswertung

3.3.1 Funktionale Anforderungen

Formularkomponenten

Der ”Drag & Drop”-Form-Builder verfügt über alle geforderten Formularkomponenten und bietet meistens sogar noch bessere Konfigurationen und Lösungen für die in der Projektbeschreibung stehenden Einsatzfälle an. Er verfügt über Textfelder und Textbereiche, welche mit Hilfe der Input-Masken-Konfiguration genau darauf spezifiziert werden können, wie viele Zeichen welcher Art erlaubt sein sollen. Zusätzlich gibt es auch noch Textfelder, die bereits auf typische Anforderungen spezialisiert sind. Es gibt spezialisierte Felder für E-Mails, URLs, Telefonnummern, Adressen, Daten, Zeiten und Währungen. Auswahllisten mit Standardwerten, Radio-Buttons und Checkboxen sind auch im Form-Builder enthalten und können schnell implementiert werden. Selbst der Datei-Upload landet einfach per ”Drag & Drop” schnell im Formular. Es können Dateitypen, sowie erlaubte Dateigrößen schnell und einfach konfiguriert werden. Zuletzt sind auch Buttons mit selbst definierbaren Aktionen vorhanden. Grundlegend gibt es von Anfang an die Aktion, dass bei der Einreichung des Formulars, die Einreichung bei Form.io in einer Datenbank gespeichert wird.

Mehrstufigkeit

Laut der Produktbeschreibung lohnt sich es, das Formular in vier separate Schritte zu zerlegen. Dabei ist immer nur ein Schritt gleichzeitig bearbeitbar. Um ein mehrseitiges Formular

umzusetzen zu können, muss man sich mit der sogenannten Wizard-Klasse von Formiojs beschäftigen. Dieser nimmt die Panel-Komponenten eines Form.io Formulars und interpretiert diese als separate Seiten.

Navigation

Um zwischen den einzelnen Seiten navigieren zu können, benötigt man den Form-Controller. Wenn der Form-Renderer mit der createForm-Funktion instantiiert wird, erhält man ein Promise-Objekt. Mithilfe der then-Funktion kann man eine Sektion mit Code erstellen, der ausgeführt wird, sobald das Formular fertig gerendert ist. Diese Sektion nennt man Form-Controller. Dort kann man für die Wizard-Events wie gotoNextPage-Event den auszuführenden Code schreiben. Mithilfe der nextPage- oder prePage-Funktion kann man genau definieren, wann man wie die Seiten des Wizards navigieren könne soll.

Dynamisches Formular

Für ein dynamisches Formular sollen verschiedene Felder nach spezifizierten Bedingungen ein- und ausgeblendet werden können. Dies ist möglich durch den Logic-Abschnitt der Konfiguration. Dort kann man sich eine sogenannte Logik bauen. Diese besteht aus einen Logik-Namen und einen Trigger. Der Trigger ist die spezifische Situation, unter der eine Aktion ausgeführt werden soll. Eine Aktion ist auch Teil einer Logik. Sie hat einen Aktionsnamen und einen Typen. Die restlichen Eigenschaften einer Aktion hängen von dem ausgewählten Typen ab. Um ein Feld ein- und auszublenden kann meine eine Aktion vom Typ Property nehmen. Wie man dem Namen schon entnehmen kann, lässt sich damit einer der Eigenschaften der Komponente ändern. Dort kann man einfach der Hidden-Komponente den gewünschten Wahrheitswert geben.

Die automatische Befüllung von Feldern unter spezifischen Bedingungen lässt sich ebenfalls mit der Logik-Konfiguration umsetzen.

Validierung

Von der Projektbeschreibung sind verschiedenste Validierungsoptionen vorgeschrieben. Diese Konfigurationen findet man selbstverständlich im Validation-Abschnitt. Enthalten sind dabei selbstverständlich die Pflichtfeld-Validierung, welche sogar automatisch das Asterisk hinzufügt und die REGEX-Musterabgleichung. Gültigkeitsbereiche müssen über die Konfiguration "Custom Validation" selbst in JavaScript geschrieben werden. Um eine Validierung bei jedem Schrittwechsel durchzuführen, muss man diese Funktionalität erstmal in den bereits selbtsgeschriebenen Events ergänzen.

Fehler

Die roten Feldmarkierung und die spezifischen dynamischen Fehlermeldung sind alle schon automatisch in Form.io implementiert.

3.3.2 Nicht-funktionale Anforderungen

Sicherheit

Form.io verfügt über einige Sicherheitsoptionen, welche sich mit den in den Anforderungen genannten Sicherheitsaspekte beschäftigen.

Datenschutz Im Paragraph zum Datenschutz wurde eine möglichst sichere Behandlung von sensiblen Daten verlangt. In der Konfiguration eines jeden Feldes gibt es im Data-Abschnitt der Konfiguration die Option Field-Level-Encryption einzusetzen. Diese Konfiguration kann also für alle Felder des Formulars, in denen sensible Informationen gespeichert werden sollen, aktiviert werden. Es sollte bedacht werden, dass Passwörter nicht verschlüsselt werden sollten, da Two-Way-Encryption eingesetzt wird.

Sicherheitslücken Mithilfe von Audit-Logging ist es möglich Sicherheitslücken festzustellen. Audit-Logs enthalten normalerweise Informationen darüber, wer was zu welcher Zeit getan hat. In den Audit Logs von Form.io sind folgende Informationen aufzufinden. Das Datum, das Event, einen Universally-Unique-Identifier, kurz uuid, die Projekt-ID, die Session-ID, die Nutzer-ID und zuletzt noch zusätzliche Informationen. Die Audit-Logs helfen auch bei dem Wiederherstellungsprozess, der Vermeidung einer Wiederholung des Problems und zur Einschätzung der Anzahl von Betroffenen im Falle eines richtigen Datenlecks.

Authentifizierung Form.io bietet SAML, OAuth, LDAP, E-Mail-Authentication und Two-Factor-Authentication an. SAML steht dabei für Security Assertion Markup Language und ist ein XML-Framework zum Austausch von Authentifizierungs- und Autorisierungsinformationen. OAuth bezeichnet zwei Protokolle, welche zusammen eine sichere standardisierte API-Autorisierung für Web-Anwendungen ermöglichen. LDAP steht für Lightweigt Directory Access Protocol und ermöglicht es sich direkt bei einen LDAP-Service wie OpenLDAP zu authentifizieren. E-Mail-Authentication ermöglicht, neben der Authentifizierung selbst, noch die automatische Anmeldung. Alle diese Optionen kombiniert mit der Möglichkeit Two-Factor-Authentication zu implementieren reichen, um einen sicheren Authentifizierungsprozess zu gewährleisten.

Autorisierung Rollen und Rechte von Benutzern bilden das Grundgerüst eines guten Autorisierungssystem. Standardgemäß gibt es die Rollen Administrator, Anonymous, Authenticated und Everyone. Es können jedoch jederzeit neue Rollen hinzugefügt werden. Rechte werden mit Permission-Scopes und Permissiontypes umgesetzt. Es gibt die Submission-Scope, die den Zugriff auf die Einreichungen innerhalb eines Formulars regelt. Über dem Submission-Scope steht die Form-Scope, welche den Zugriff auf das gesamte Formular definiert. Der größte Permission-Scope ist der Project-Scope, welcher den Zugriff auf alle Formulare eines Projektes ermöglicht. Für jedes dieser Permission-Scopes gibt es eine lange Reihe an Permission-Types, die den Rollen zugewiesen werden können. Die hohe Anzahl an verschiedenen Rechten, mit den verschiedensten Geltungsbereichen ermöglichen eine sehr flexible und spezialisierte Autorisierung.

Benutzerfreundlichkeit

In diesem Teil der Analyse wird die Benutzerfreundlichkeit anhand der Effizienz und Effektivität der Entwicklung und Nutzung des Formulars gemessen. Außerdem wird die Dokumentation von Form.io bewertet.

Effektivität Das Ziel des Benutzers ist es eine Urkunde aus dem Geburtenregister zu beantragen. Durch die korrekte Ausfüllung des Formulars wird dieses Ziel selbstverständlich erfüllt. Für den Entwickler war das Ziel eine Anwendung mit Form.io zu schreiben, die diese Urkundeanforderung ermöglicht. Da alle funktionalen Anforderungen erfüllt worden sind und Form.io eine Vielzahl an Funktionen zur Formularentwicklung und dem Datenmanagement, sowie Konfigurationsoptionen für Felder und Sicherheit bietet, schneidet Form.io in Sachen Effektivität gut ab.

Effizienz Der Beantragunsprozess einer Urkunde aus dem Geburtenregister wurde für den Nutzer an vielen Stellen optimiert. Das Auto-Fill-Feature von dynamischen Formularen beschleunigt die Ausfüllung des Online-Formulars. Durch die Validation wird soweit es geht dafür gesorgt, dass Fehleingaben vermieden werden. Diese würden den Antrag ungültig machen, was den Beantragunsprozess stark in die Länge ziehen und somit ineffizient machen würde. Auch für den Entwickler ist die Umsetzung einer Webanwendung mit Form.io unglaublich effizient. Für die meisten Formulare reicht der Standard "Drag & Drop"Form-Builder der Form.io Website. Selbst bei sehr komplexen Formularen ist Form.io sehr effizient, da keine Zeit mit grundlegenden Formularaufbau und Eigenschaften verschwendet werden muss. Daraus folgt, dass Form.io zur Entwicklung von Webanwendung mit Formularen sehr gut geeignet ist.

Dokumentation Die Dokumentation Form.io ist sehr gut. Es wird die Funktionsweise und Idee hinter Form.io ausführlich und mit Grafiken erklärt. Es gibt für Benutzer und Entwickler getrennte Guides mit allen wichtigen Informationen zur Entwicklung oder Nutzung von Form.io. Es gibt für Formiojs und die Form.io API nochmal separate detaillierte Dokumentationen. Außerdem sind viele Step by Step Guides in der Dokumentation zu finden. Die Dokumentation lässt eigentlich nicht viel zu wünschen übrig.

Wartbarkeit

Das Formular zur Urkundenanforderung aus dem Geburtenregister folgt bestimmten Auflagen, die bestimmen, was für Informationen das Standesamt braucht, um eine Geburtsurkunde verschicken zu können. Diese Auflagen können sich jederzeit ändern, was natürlich Änderungen in dem Formular mit sich bringt. Änderungen an dem Formular sind aber mit Form.io sehr schnell umgesetzt. Man muss nur einmal auf der Form.io Seite die benötigten Formularänderungen vornehmen und alle Anwendungen, die das Form.io Formular eingebettet haben bekommen die aktuelle Versionen des Formulars. Auch in Sachen Wartbarkeit schließt Form.io gut ab.

Ergebnis

Aus der Analyse geht hervor, dass mit Form.io alle funktionalen und nicht-funktionalen Anforderungen umsetzbar sind. Es gibt im Form-Builder genug Komponenten um so gut wie jegliches Formular umzusetzen. Außerdem können dem Form-Builder selbst definierte Felder hinzugefügt werden. Mehrstufigkeit und Navigierbarkeit sind mit Hilfe der Wizard-Klasse umsetzbar. Durch die vielen Konfigurationsmöglichkeiten sind die Eigenschaften eins dynamischen Formulars gegeben. Validierung und dessen Fehlermeldungen bei fehlerhaften Eingaben wird ebenfalls durch die Konfigurationsmöglichkeiten der einzelnen Felder umgesetzt. In Sachen Sicherheit bietet Form.io eine Vielzahl an Optionen und auch in der Benutzerfreundlichkeit und Wartbarkeit schließt es gut ab. Daraus folgt das Form.io für die Entwicklung von komplexen Formularen in progressiven Webanwendung absolut gut geeignet ist.

4 Fazit und Ausblick

Das Ziel dieser Seminararbeit war es, Form.io als Grundlage für die Entwicklung einer Anwendung zur Urkundenanforderung aus dem Geburtenregister zu evaluieren. Die Projektbeschreibung diente als Grundlage für die Analyse von Form.io. Die Ergebnisse dieser Analyse zeigen auf, das sich Form.io in der Umsetzung der funktionalen und nicht-funktionalen Anforderungen ziemlich gut geschlagen hat. Form.io sollte auf jeden Fall als eine Option in Betracht gezogen werden, wenn neue Anwendungen mit Webformularen entwickelt werden sollen. Vor allem bei Projekten mit simpleren Anforderungen an das Formular zeigte Form.io extreme Effizienz. Für eine anknüpfende Evaluation sollten auf jeden Fall die nötigen Back-End Anforderungen in der Projektbeschreibung beachtet werden. Die gesammelten Informationen aus dem Formular werden selbstverständlich nicht einfach an ein Standesamt weitergeleitet. Sie müssen erst im Backend zu einer korrekten Form für das Standesamt verarbeitet werden. Form.io und ähnliche Form und Data Management Platforms werden sehr wahrscheinlich in Zukunft an Relevanz gewinnen, da durch die immer wichtig werdende Digitalisierung, mehr und mehr Verwaltungsleistungen digitalisiert werden sollen.

A Literaturverzeichnis

Braun Micheal(12. Januar 2016): Nicht-funktionale Anforderungen, [online] https://www.pst.ifi.lmu.de/Lehre
15-16/jur-pm/braun-ausarbeitung.pdf

Henrich, Andreas (2012): Web Engineering, Aspekte der systematischen Entwicklung von
Web-Anwendungen, [online] https://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai¿*ehrstuehle*/*medie*
$Sem\_WS2011 - 21\_Web - Engineering.pdfpage = 84$

Eine Geburtsurkunde beantragen, [online] https://www.stadte-gemeinden.de/geburtsurkunde.html

(13. Oktober 2022): Gebursturkunde,[online] https://de.wikipedia.org/wiki/Geburtsurkunde

Gute Formulare bauen, [online] https://nozilla.de/tipp/gute-formulare-bauen/

Die eID-Karte für Bürgerinnen und Bürger der EU und des EWR, [online] https://www.personalausweisportal
und-buerger/eID-karte-der-EU-und-des-EWR/eid-karte-der-eu-und-des-ewr-node.html

A Form and Data Management Platform for Progressive Web Applications, [online] https://www.form.io/

Regulärer Ausdruck, [online] https://de.wikipedia.org/wiki/Regulärer_Ausdruck

(27. November 2020): Peronsenstandsgesetz, [online] https://wiki.genealogy.net/Personenstandsgeset

(11. März 2022):Internet-Sicherheit von Formularen optimieren, [online] https://www.hosteurope.de/blog/inte
sicherheit-von-formularen-optimieren/

Welcome to Form.io, [online] https://help.form.io/ JavaScript Powered Forms by <form.io>,
[online] https://formio.github.io/formio.js/docs/

Lemcke Markus: Was ist Usability / Benutzerfreundlichkeit?, [online] https://www.marlem-
software.de/marlemblog/2014/12/10/was-ist-usability-benutzerfreundlichkeit/