Hochschule für Angewandte Wissenschaften Hamburg
*Hamburg University of Applied Sciences*

# Bachelorarbeit

Denis Tomczyk

Web Scraping zur systematischen Extraktion von Nutzerkommentaren

*Fakultät Technik und Informatik*
*Department Informatik*

*Faculty of Engineering and Computer Science*
*Department of Computer Science*

**Denis Tomczyk**

# Web Scraping zur systematischen Extraktion von Nutzerkommentaren

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Wirtschaftsinformatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Zukunft
Zweitgutachter : Prof. Dr. Steffens

Abgegeben am 28.01.2020

**Denis Tomczyk**

**Thema der Arbeit**

Web Scraping zur systematischen Extraktion von Nutzerkommentaren

**Stichworte**

Web Scraping, Screen Scraping, Datenextraktion, Nutzerkommentare

**Kurzzusammenfassung**

In dieser Bachelorarbeit wird Web Scraping als Methode zur Datenextraktion betrachtet und eine prototypische Anwendung entwickelt. Es werden unterschiedliche Aspekte beim Web Scraping erläutert und Medienpräsenzen als potenzielle Datenquelle für Nutzerkommentare herangezogen.

**Denis Tomczyk**

**Title of the paper**

Web scraping for the systematic extraction of user comments

**Keywords**

Web scraping, screen scraping, data extraction, user comments

**Abstract**

This bachelor thesis describes web scraping as a method for data extraction and a prototypical application will be developed. Different aspects of web scraping will be explained and media presences are used as a potential data source for user comments.

# Inhaltsverzeichnis

# 1 Einleitung

## 1.1 Motivation

Seit der Geburtsstunde der ersten Website 1990 vermehrte sich die Anzahl der Websites und das damit verbundene Informationsvolumen im Internet gewaltig. [1] Im Laufe der Zeit entwickelte sich das sogenannte Social Web, welches eine neue Art der Vernetzung darstellt. Das Social Web gibt Nutzern die Möglichkeit Gespräche über Medieninhalte öffentlich sichtbar und im direkten Anschluss an diese Medieninhalte zu führen. [2] Dabei können sich diese in Form vom Nutzerkommentaren geführten Gespräche als nützlich erweisen. Hierbei gilt es eine effiziente Methode zu finden, um diese Daten systematisch zu extrahieren.

## 1.2 Aufgabenstellung

Ziel dieser Arbeit ist es sich der Thematik „Web Scraping“ zu nähern und eine Anwendung zu entwickeln, die auf mehreren Medienpräsenzen nutzbar ist. Unterschiedliche Websites und Ihre Strukturen erschweren es, dass übergreifend systematisch Nutzerkommentare extrahiert werden können. Im Rahmen dieser Arbeit wird eine prototypische Anwendung entwickelt um mit Web Scraping Nutzerkommentare zu extrahieren.

# 2 Grundlagen

Das folgende Kapitel dient der Herstellung eines grundlegenden Verständnisses über die Thematik des Web Scrapings. Ergänzend zu der Begriffsdefinition und der Erläuterung von eingesetzten Webtechnologien, wird auch auf ausgewählte Methoden beim Web Scraping eingegangen. Abschließend werden zudem unterschiedliche Anwendungsgebiete sowie auftretende Problematiken vorgestellt.

## 2.1 Definition

Die zentrale Aufgabe beim Web-Scraping ist es, gezielt relevante Informationen aus einer Vielzahl von Daten zu sammeln. [3] Hierbei werden im Kontext dieser Arbeit im späteren Verlauf unterschiedliche Methoden aufgezeigt. Bekannte häufige Synonyme sind unter anderem als “Screen Scraping” und “Web harvesting” bekannt. [4] Die italienische Gesellschaft für Wirtschaftsdemografie und -statistik beschreibt Web Scraping als Prozess, welcher es erlaubt Informationen aus Websites zu extrahieren. [5]

Darüber hinaus ist es wichtig die Begrifflichkeiten “Web Crawling” und “Web Scraping” zu differenzieren. Denn beide Verfahren verfolgen in der Arbeitsweise unterschiedliche Ziele. Die Gemeinsamkeit beruht darauf Daten zu sammeln. Web Crawler werden jedoch für das Analysieren und Indizieren von Webseiten benutzt. Sie erkennen auf Webseiten befindliche Hyperlinks, speichern diese Adressen in Listen und rufen letztlich diese Webseiten sequenziell zur erneuten Analyse auf. Somit „kriechen“ oder „schlängeln“ (engl. to crawl) sich jene von Webseite zu Webseite. Durch diese Vorgehensweise werden die besuchten Webseiten abhängig von der Aufgabe der Web Crawlers indiziert,

ausgewertet und gespeichert. [5] Web Crawler werden überwiegend von Suchmaschinen verwendet. [6]

Beim Web Scraping andererseits, liegt der Fokus auf der Extraktion der Daten. Die Funktionsweise von Web Scraping lässt sich in zwei Arbeitsschritte gliedern. Der erste Arbeitsschritt besteht darin, die Webseite abzurufen und im zweiten Arbeitsschritt die gewünschten Daten zu extrahieren. Die eigentliche Datenextraktion beim Web Scraping wird durch den sogenannten Wrapper durchgeführt. Wrapper sind Regeln oder Prozeduren um aus der Datenquelle die gesuchten Informationen zu extrahieren. [7] Es gibt eine oder mehrere konkret definierte Webseiten aus denen zuvor festlegte Informationen präzise extrahiert und strukturiert wiedergegeben werden. Je nach Anwendungsfall besteht die Möglichkeit, dass die gewonnenen Daten in Datenbanken oder in geeigneten Formaten für Tabellenkalkulationsprogramme gespeichert werden. [8]

## 2.2 Webtechnologien

Um nachzuvollziehen wie Web Scraping auf Webseiten funktioniert, ist es vorher essentiell die Webseitenstrukturen und eingesetzten Technologien zu verstehen. Hierbei wird auf den grundlegenden Aufbau, sowie der Einsatz relevanter Technologien von Websites eingegangen.

Eine Website definiert ein als Ganzes erkennbares Projekt, welches im Internet unter einem reservierten Domain-Namen, dem sogenannten “Uniform Resource Locator” (URL) aufrufbar ist. In der Regel besteht eine Website zusätzlich aus einem einheitlichen Seitenlayout, sowie einer festgelegten Navigationsmöglichkeit. Hierzu zählen beispielsweise Firmen- und Privatauftritte im Web, Online-Shops oder auch Suchmaschinen. [9]

**HTML**

Um Websites im Browser aufzurufen, werden HTML-Dokumente als Grundlage für das World Wide Web verwendet. HTML ist eine Auszeichnungssprache und steht als Abkürzung für “Hypertext Markup Language”. Mithilfe von HTML-Elementen wird dem HTML-Dokument die Struktur und die Semantik einzelner Komponenten festgelegt und ausgezeichnet. [10] Die Kennzeichnung charakteristischer Elemente wie beispielsweise Überschriften, Textabsätze oder Listen werden dadurch als solche durch Tag-Paare gekennzeichnet. [9] Webbrowser lesen diese Tag-Paare anschließend aus und geben dann den Inhalt entsprechend optisch erkennbar und strukturiert wieder. Alle HTML-Elemente eines Dokumentes setzen sich aus einer hierarchischen Baumstruktur zusammen. HTML ist der Grundbaustein beim Web Scraping von Websites.

**XML**

XML ist die Abkürzung für extensible Markup Language. Es ist ein textbasiertes Datenformat zur Darstellung hierarchisch strukturierter Daten. Die Daten werden im XML-Dokument von Textauszeichnungen umgeben. Diese Textauszeichnungen beschreiben die Daten in ihrer Art. Anders als bei HTML sind keine Tags in XML vorgegeben, sondern werden vom Entwickler selbst definiert. [9]

**CSS**

Die Funktion von CSS (engl. Cascading Stylesheet) ist, einer Website das richtige äußere Erscheinungsbild in Form von Design und Layout zu geben. Zuvor definierte HTML-Elemente können nun mit CSS in der dazugehörigen Darstellung modifiziert werden. Es kann somit festgelegt werden, wie die Inhalte im Webbrowser angezeigt werden sollen. [10] Das folgende Beispiel verdeutlicht den Einsatz von CSS in Bezug auf HTML Elemente:

```
<p class="moreinfo">For more information see the
<a href="http://www.example.com/report">final report</a>.</p>
```

Das p-Element dient dazu Texte in Absätze zu gliedern. Mit dem class-Attribut wird allen gekennzeichneten p-Elementen in dem Stylesheet eine einheitliche Darstellung bestimmt. Um alle „moreinfo“ p-Elemente in kursiver Schreibweise anzuzeigen genügt es dies in dem Stylesheet zu definieren:

```
p.moreinfo { font-style: italic }
```

Web Scraping Anwendungen bedienen sich dieser vordefinierten Struktur um gezielt auf die Inhalte zuzugreifen.

**DOM**

DOM (Document Object Model) ist eine Schnittstelle zwischen HTML und JavaScript, welche den Zugriff und Veränderung des HTML-Dokuments ermöglicht. Der Inhalt im HTML-Dokument kann verändert, hinzugefügt oder entfernt werden. Das Dokument wird in einer Baumstruktur mit einzelnen Knoten dargestellt.

**JavaScript**

JavaScript hat Zugriff auf das HTML-Dokument, um dynamisches Verhalten zu ermöglichen. HTML definiert die Struktur und den Inhalt einer Webseite. Das Design sowie das Layout werden durch CSS umgesetzt. Durch Kombination beider Komponenten, wird eine statische Webseite erzeugt. Für das Verhalten ist letztlich JavaScript zuständig. [11] Mit dem zusätzlichen Einsatz von JavaScript werden Webseiten dynamisch. [12] Denn es ermöglicht beispielsweise auf Benutzereingaben zu reagieren und in Folge dessen HTML Code auszulesen, zu verändern, zu entfernen oder neuen HTML Code hinzuzufügen.

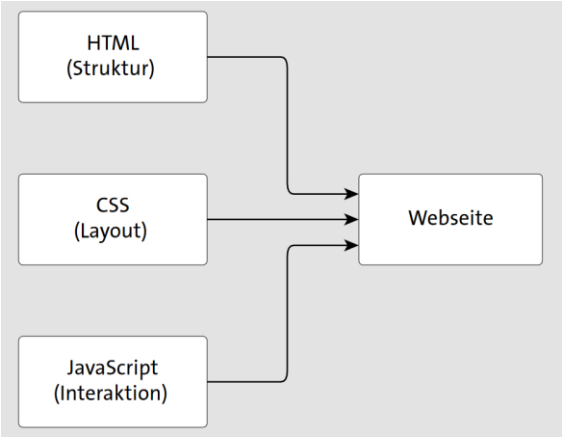


**Abbildung 1: Kombination aus HTML, CSS und JavaScript**
**Quelle: [11]**

Die Änderung erfolgt ausschließlich im HTML-Dokument welches sich im Arbeitsspeicher des Benutzers befindet. Das HTML-Dokument auf dem Webserver bleibt davon unberührt. Durch diese Manipulation am HTML-Dokument gibt JavaScript einer Webseite ein unvorhersehbares dynamisches Verhalten und dies kann sich auf das Web Scraping auswirken. [13]

**Ajax**

Ajax (engl. Asynchrone JavaScript and XML) beschreibt das Konzept der asynchronen Datenübertragung zwischen einem Webbrowser und Webserver mit Hilfe von JavaScript sowie auch XML. Somit werden mit Ajax ohne ein erneutes Laden der vollständigen Website, lediglich einzelne Bestandteile erneuert. Verzichtet eine Webseite auf den Einsatz der Ajax-Technologie, läuft der Prozessfluss einer klassischen Webanwendung oder Webseite synchron ab. Das heißt, dass die Skriptausführung angehalten wird, bis die angefragten Daten vom Webserver zurückgekommen sind. Doch das nun asynchrone Verhalten in Bezug mit Ajax zeigt auf, dass die Skriptausführung bei einer HTTP-Anfrage weitergeht, da diese Anfrage an den Webserver im Hintergrund ausgeführt wird und dem Besucher die Webseite nach wie vor zur Verfügung steht. [13] Dieses Verhalten hat den Vorteil, dass nur die tatsächlich benötigten Daten versendet werden und der entstehende Webtraffic reduziert wird. Jedoch gibt es mit Ajax auch Nachteile. Ein Nachteil wäre, dass die nachgeladenen Inhalte von Suchmaschinen nicht

erfasst werden können. Denn Suchmaschinen prüfen nach dem Aufrufen der Website nur den ursprünglichen HTML-Quelltext.

Schon bereits bei der Suchmaschine Google wird der Besucher mit Ajax konfrontiert. Sobald der erste Tastenanschlag in die Suchmaske eingetippt wird, schlägt die Suchmaschine bereits erste passende Begriffe vor. Mit jedem weiteren Tastenanschlag wird eine erneute Anfrage an den Webserver gestellt und der Inhalt der vorgeschlagenen Suchanfrage aktualisiert. [14]

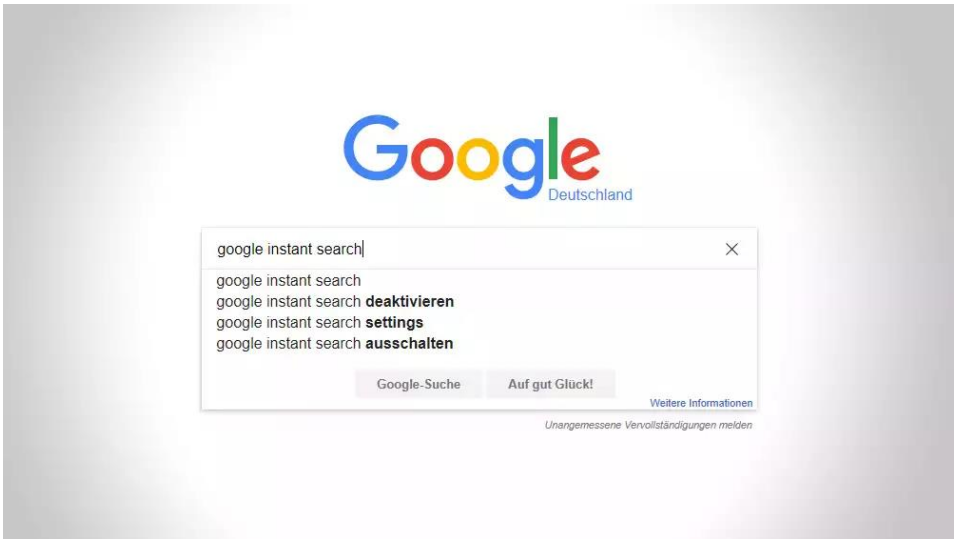


**Abbildung 2: Google Instant Suchanfrage**

## 2.3 Methoden

Zeitgleich mit der Geburtsstunde des World Wide Webs gab es auch bereits schon erste in der Praxis angewendete Scraping Methoden. Im Laufe der Zeit wurden kontinuierlich weitere Methoden entwickelt und hinzugefügt. Die Auswahl einer effektiven Methode hängt von den gegebenen Möglichkeiten, Anwendungszweck, und dessen Ziel ab. Dies richtet sich nach der Art, wie die entsprechenden Inhalte auf der Website aufgerufen und geladen werden.

**Manuelles Scraping**

Beim manuellen Scraping übernimmt der Mensch die Rolle der Web Scraping Anwendung. Hierbei ruft der Benutzer die Website im Webbrowser auf und durchsucht diese selbstständig nach den gesuchten Informationen und Daten. Diese werden letztlich aus der Webseite kopiert und beispielswiese in ein Tabellenkalkulationsprogramm gespeichert. Diese triviale Methode eignet sich bevorzugt bei sehr geringen Datenmengen, sowie in Situationen in denen der Programmieraufwand in keiner Relation zu den benötigen Daten steht.

**Protokollbasiertes Web Scraping**

Websites lassen sich in statische und dynamische Websites einordnen. Zur ersten Klasse gehören die „statischen“ Websites, bei denen sich Darstellung und Inhalte nach dem HTTP-Abruf im Browser nicht verändern. [15] Hierfür wird das Protokoll-Basierte Web Scraping eingesetzt. Sobald die Abfrage über die URL an den Server gestellt wurde, antwortet dieser mit dem HTML Dokument und wird dem Benutzer im Webbrowser als Website angezeigt. Der Prozess ist nun abgeschlossen und Veränderungen an den Inhalten nicht möglich. Folgt der Benutzer einer Verlinkung auf der Website, beispielsweise in Form eines Menüpunkts, wird eine neue Anfrage an den Server gestellt und ein neuer Prozess beginnt. Die Website wird erneut vollständig aufgerufen.

Diese Art von Websites sind für Web Scraping Anwendungen relativ einfach zu bewältigen, da der komplette HTML Code strukturiert und vollständig zur Verfügung steht. Bei der Programmierung einer Protokoll-Basierten Web Scraping Anwendung wird auf die grafische Darstellung der Website in Form eines Browsers verzichtet und durch einen HTTP-Client ersetzt. Der HTTP-Client sendet die entsprechenden Serveranfragen und nimmt die Antworten entgegen. Es bestehen keine Interaktionsmöglichkeiten mit der Website.

**Browsersimuliertes Web Scraping**

Browsersimuliertes Web Scraping kommt zum Einsatz, wenn in Websites dynamischer Inhalt vorhanden ist. Hierbei können sich Darstellung und Inhalte nachträglich verändern, ohne dass der Besucher einen neuen HTTP-Abruf starten muss. Dynamische Webseiten beinhalten für gewöhnlich den Einsatz von JavaScript oder Ajax. Protokollbasierte Web Scraper können das enthaltene JavaScript nicht interpretieren, was zur Folge hat, dass die entsprechenden Elemente nicht geladen werden und somit kein Inhalt angezeigt wird. Durch die Simulation eines realen Webbrowsers mit allen herkömmlichen Funktionalitäten wird dieses Problem beim browsersimulierten Web Scraping gelöst. Zusätzlich können durch den vorhanden grafischen Webbrowser Interaktionen mit den Inhalten der Website durchgeführt werden und somit Aktivitäten eines Benutzers simuliert. Der Datenzugriff beim Protokollbasierten, wie auch beim Browsersimulierten Web Scraping kann über unterschiedliche Herangehensweisen erfolgen:

- Zugriff über die HTML Elemente
- Zugriff über reguläre Ausdrücke
- Zugriff über das Document-Object-Model
- Zugriff über XPath

**APIs**

Es muss jedoch nicht immer auf die eben genannten Methoden zurückgegriffen werden. Dies ist der Fall, sofern durch den Website-Betreiber Zugriff auf eine vorhandene API (engl. application programming interface), also auf die Schnittstelle der Website, gewährt wird. Einhergehend mit der Zugriffsberechtigung auf die API ist, dass mögliche Problematiken (z.B. Urheberrecht, Abwehrmechanismen) bereits von Beginn an vermieden werden können.

## 2.4 Anwendungsgebiete

Web Scraping wird in vielen unterschiedlichen Gebieten im World Wide Web angewendet. Klassische Voraussetzungen für den Einsatz von Web Scraping können sein:

- Es werden größere Mengen an Daten von einer oder mehreren Webseiten benötigt. [8]
- Es werden kontinuierlich neue Daten von einer oder mehreren Webseiten veröffentlicht, wobei diese Webseiten keine Schnittstelle für den automatischen Abruf bereitstellen. [8]

Sofern eine dieser Voraussetzungen als Problem identifiziert wird, lohnt es sich häufig auf Web Scraping zurückzugreifen. Im Folgenden eine Auflistung einiger Anwendungsgebiete.

**Forschungszwecke**

Soziale Medien wie Facebook und Twitter bieten Wissenschaftlern in Bezug auf Kommunikation, Verhalten und weiteren Forschungsvorhaben einen höchst nützlichen Informationspool. Nicht selten stellen die entsprechenden Sozialen Medien leider nur eingeschränkte API Schnittschnellen für derartige Zwecke zur Verfügung. Daher ist Web

Scraping ein attraktives Mittel für die Informationsgewinnung bei derartigen Forschungszwecken. [16]

**Preisvergleichsportale**

Preisvergleichsportale, Internetbörsen für Flüge und weitere touristische Angebote nutzen Web Scraping, um einen Überblick sämtlicher Angebote von Online-Shops, Airlines und Veranstaltern zu erhalten. Die gewonnen Informationen werden für das eigene Geschäftsmodell genutzt und den Besuchern auf ihrer Plattform provisionsgesteuert angeboten. [17] [18]

**Content-Aggregatoren**

Content-Aggregatoren sind auf das Sammeln von frisch veröffentlichen Daten und Informationen spezialisiert. Die gesammelten Daten werden dem User kompakt und strukturiert präsentiert. Hierzu gehören beispielsweise News-Aggregatoren. Diese extrahieren aktualisierte Nachrichten aus unterschiedlichen Medienpräsenzen und stellen diese ihren Nutzern gebündelt zur Verfügung. Job-Aggregatoren hingegen nutzen Jobbörsen und Unternehmens Websites um eine Übersicht von aktuellen Stellenangeboten auf der eigenen Plattform bereit zu stellen. [18]

**Web-Analyse-Tools**

Web-Analyse-Tools ermitteln mit Hilfe von Suchmaschinen die Platzierung von Websites auf unterschiedliche Suchbegriffe und bereiten diese Daten für deren Kunden auf. Dadurch können Webseitenbetreiber und Unternehmen mit den erhobenen Daten eine fundierte Suchmaschinenoptimierung durchführen. [19] [20]

## 2.5 Problematiken

Die automatische Extraktion von Informationen und Daten im Internet findet nicht bei allen Beteiligten Zuspruch. Durch unterschiedliche Ansichtsweisen und dem breiten Spektrum an Technologien, entstehen auch rasch Problematiken. Diese Konflikte finden zusätzlich zur technischen Ebene auch in der Rechtsprechung statt.

**Rechtliche Aspekte**

Die Ansicht der Gerichte zum Thema Web Scraping hat sich in dem letzten Jahrzehnt drastisch geändert. Klagen gegen das automatische Extrahieren von Informationen und Daten waren noch bis ins Jahr 2009 erfolgreich. [17] Der allgemeine Konsens der Oberlandesgerichte (OLG) sieht dies mittlerweile anders. Unter bestimmten Gesichtspunkten werden Klagen gegen Web Scraping zunehmend abgelehnt. Im Fall einer Klage gegen eine Preisvergleichsplattform, welche die Online-Angebote der Airlines auf der eigenen Plattform bereitgestellt hat, wurde vom OLG Hamburg und dem OLG Frankfurt entschieden, dass die automatische Extraktion der Informationen zulässig sei. Die Gerichte bezogen sich in Ihrem Urteil auf drei grundlegende Aspekte:

- Die Websites der Airlines wurden durch das Web Scraping nicht überlastet [17]
- Die Websites sind rechtlich und technisch frei zugänglich [17]
- Es werden keine wesentlichen Bestandteile der Datenbank kopiert [17]

Die rechtliche Komplexität beim Web Scraping dehnt sich jedoch noch auf weitere Gesichtspunkte wie z.B. das Urheberrecht, Vertragsrecht oder virtuelle Hausrecht aus und kann dennoch nicht pauschalisiert als rechtlich unbedenklich betrachtet werden. [18]

### Abwehrmechanismen

Daten aller Art sind in der heutigen Zeit für die Plattformen ein wertvolles Gut. Aus diesem Grund gibt es auch unterschiedliche technische Ansätze sich vor unerwünschter Datenextraktion zu schützen.

### Captcha Abfragen

Bei Captcha Abfragen (engl. Completely Automated Public Turing test to tell Computers and Humans Apart) wird der Besucher dazu aufgefordert simple Rätsel oder Aufgaben in Form von einer Texteingabe, einer Rechenaufgabe oder der Erkennung bestimmter Muster zu lösen, bevor eine entsprechende Handlung auf der Website durchgeführt werden kann. [21] Ziel ist es durch diesen Test einen Menschen von einer Maschine unterscheiden zu können (sog. Turing Test). Sofern die Aufgabe korrekt gelöst wurde, erhält der Anfragende Zugriff auf die Inhalte. Es existieren unterschiedliche Arten von Captcha-Tests die auch ständig weiterentwickelt werden. [22] Captcha Abfragen können beispielsweise beim Aufrufen bestimmter Webseiten, bei Login-Formularen oder bei Suchanfragen eingebunden werden.

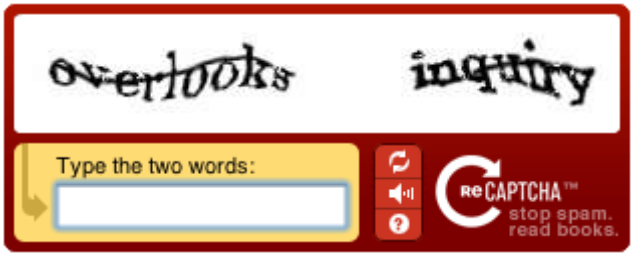


**Abbildung 3: Captcha Abfrage in Text-Form**
**Quelle: [23]**

Mit Hilfe von künstlicher Intelligenz, Pixelerkennung in Bilddateien und Algorithmen zur Zeichenerkennung können bereits viele Captcha-Abfragen mittlerweile von Computern gelöst werden. [23]

**Geschützte Bereiche**

Die Inhalte von geschützten Bereichen auf Websites sind nicht öffentlich einsehbar. Zugriff auf diesen Teilbereich wird nur einer bestimmten Personengruppe durch Eingabe eines Passwortes oder einer vorherigen Registrierung gewährt. Eine Extraktion der Daten kann aus diesem Grund nur eingeschränkt vollzogen werden, sofern eben eine dieser Zugriffberechtigungen vorliegen.

**IP-Blockierung**

Der Webserver protokolliert jeden Besucher auf der Website anhand seiner IP-Adresse. Sperrungen der IP-Adresse können den Besucher den Zugang zur Website unterbinden. Potenzielle Web Scraping Anwendungen, die von einem Webserver aus agieren, werden somit ausgeschlossen. Es können sogar auch ganze IP-Adressbereiche gesperrt werden. Der vermeidliche Vorteil der Anfragensperre beherbergt auch einen Nachteil, denn IP-Blockierungen lassen sich mit wenig Aufwand durch eine Verschleierung der IP-Adresse z.B. durch einen Proxy-Server umgehen.

**Veränderung der Webseitenstruktur**

Eine Veränderung der Webseitenstruktur kann abhängig von der Art der Veränderung eine Scraping Anwendung funktionsunfähig machen. Gesuche Elemente, CSS-Klassenbezeichnungen oder Verzeichnisse können nicht mehr aufgefunden werden und führt dazu, dass die Web Scraping Anwendung ebenfalls verändert bzw. an die neue Webseitenstruktur angepasst werden muss.

**Dynamische Webseiten**

Dynamische Webseiten repräsentieren die Problematik, dass die Inhalte einer Webseite dynamisch verändert oder generiert werden. Zum Zeitpunkt des initialen Abrufens der Webseite existieren bestimmte Inhalte noch nicht, sondern werden erst mit Hilfe von JavaScript im Browser erstellt. Aufgrund der Tatsache, dass das Protokoll-Basierte-Scraping das JavaScript ignoriert und nicht verarbeiten kann, bleibt die Webseite ohne

Inhalte. [15] Aus diesem Grund stößt hier das Protokoll-Basierte-Scraping an seine Grenzen.

**Ungewöhnliche Formate**

Häufig wollen sich Webseitenbetreiber durch den Einsatz ungewöhnlicher Dateiformate auf Ihren Webseiten vor einer Datenextraktion schützen. [24] In diesem Fall werden Texte, Tabellen und weitere Inhalte beispielsweise in Grafiken als Bilddatei eingebunden. Den durchschnittlichen Besucher tangiert diese Methode kaum, da dieser weiterhin seine gewünschten Inhalte angezeigt bekommt. Scraping Anwendungen wiederum können die in der Bilddatei befindlichen Inhalte nicht interpretieren.

# 3 Analyse

In diesem Kapitel werden Medienpräsenzen im Internet näher betrachtet und daraus resultierend erste bereits erkennbare Anforderungen für eine Web Scraping Anwendung festgehalten.

## 3.1 Medienpräsenzen als Datenquelle

Heutzutage bieten einige Medienpräsenzen den Lesern die Möglichkeit über den Inhalt ihrer journalistischen Beiträge auf der eigenen Website zu diskutieren, Meinungen auszutauschen oder sich mit dem Thema auseinander zu setzen. In der Kommunikationswissenschaft wird dies auch als Anschlusskommunikation bezeichnet. [25] Die Begriffe „journalistischer Beitrag“, „Beitrag“ und „Artikel“ werden in diesem Zusammenhang synonym verwendet. Medienpräsenzen sind Plattformen, dessen Inhalte als journalistische Beiträge auf ihrer eigenen Website veröffentlichen. Diskussionsbereiche sind virtuelle Plätze im direkten Anschluss dieser journalistischen Beiträge, in denen die Leser in Form von Nutzerkommentaren kommunizieren können.

Ziel der Diskussionsbereiche ist es einen offenen kommunikativen Austausch zu einem spezifischen Thema unter den Nutzern zu führen. Diese Diskussionen werden im Internetzeitalter als Weiterentwicklung des klassischen Leserbriefs eingeordnet. [2] Nutzerkommentare stellen für das Web Scraping einen besonderen Reiz dar. Bislang war es mühsam zwischenmenschliche Kommunikation über massenmediale Inhalte systematisch zu erfassen. Die meisten Gespräche über Medien fanden im sozialen Umfeld statt. [26] Mit der Entwicklung des Social Web und der Einführung von Diskussionsbereichen auf Medienpräsenzen ergab sich vermehrt ein öffentlich

erkennbarer Dialog der Leser. [2] Dieser Dialog wird im unmittelbaren Anschluss der journalistischen Beiträge in den Diskussionsbereichen der Medienpräsenzen in Form von Nutzerkommentaren ausgetragen. Das Interaktionspotenzial beschränkt sich jedoch nicht auf den jeweiligen journalistischen Beitrag, sondern auch auf die Beiträge anderer Nutzer. [2] Dadurch ist es möglich, dass Nutzer statt nur gegen den Journalisten oder das Thema des Beitrags, auch miteinander einen Diskurs abhalten. Jedoch ist gleichzeitig zu beachten, dass jegliche Kommunikation unter redaktioneller Kontrolle und Moderation über tolerierte Kommentare stattfindet.

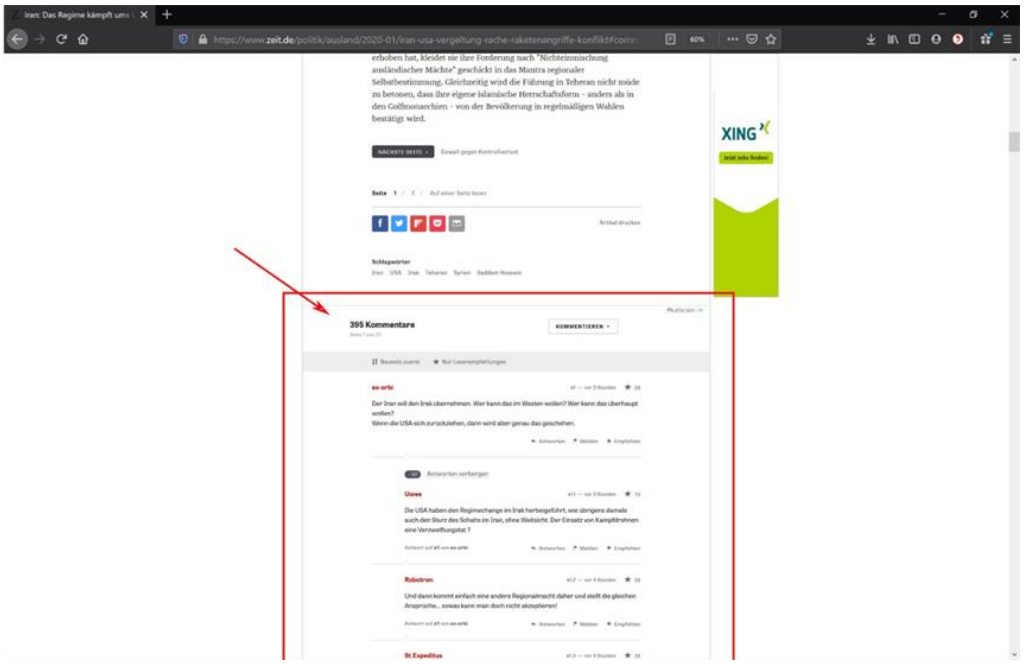


**Abbildung 4: Diskussionsbereich**
**Quelle: Zeit.de**

In Rahmen dieser Arbeit werden die Medienpräsenzen „Spiegel Online“ (SPON) und „Die Zeit“ (ZEIT) hierfür verwendet. Um aktiv an einer Diskussion teilzunehmen und Nutzerkommentare veröffentlichen zu dürfen, bedarf es einer vorherigen Registrierung. Der Zugriff auf die Diskussionsbereiche ist uneingeschränkt und ohne vorherige Registrierung möglich. Hierbei eignen sich besonders Medienpräsenzen für das Web Scraping als interessante Datenquelle.

## 3.2 Bestandteile von Diskussionsbereichen

Jede Website ist in ihrer Struktur des HTML Dokuments und Auswahl der verwendeten Webtechnologie zu unterscheiden. Aus diesem Grund ist es in der Regel notwendig für jede Website ein eigenes Scraping-Szenario zu entwerfen. Um eine Web Scraping Anwendung für möglichst viele Medienpräsenzen zu implementieren, ohne dass der Benutzer wiederholt ein Scraping-Szenario entwerfen muss, wird eine einheitliche Struktur benötigt. Bei der Betrachtung der Bestandteile von den Diskussionsbereichen bei SPON und ZEIT kann eine Struktur erkannt und abstrahiert werden. Ein Diskussionsbereich ist eine Liste von erstellten Nutzerkommentaren. Ein Nutzerkommentar kann als Container mit Attributen interpretiert werden. In diesem Container befinden sich die einzelnen atomaren Bestandteile in Form eines Nutzernamens und dem eigentlichen geschriebenen Kommentar in Textform.

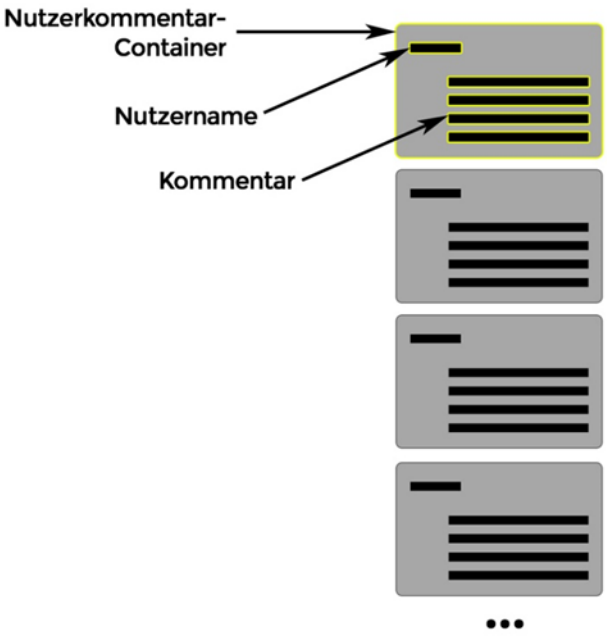


Abbildung 5: Struktur von Nutzerkommentaren
(eigene Darstellung)

Mit jedem neuen generierten Nutzerkommentar wird ein weiterer Container an die Liste im Diskussionsbereich gehängt. Diese Struktur kann somit für eine systematische

Extraktion der Nutzerkommentare für die zu erstellende Anwendung genutzt werden. Der Vorteil durch ein festes Scraping-Szenario ist, dass der Nutzer keine eigenen Szenarien analysieren und entwerfen muss. Jedoch beschränkt sich die Funktionalität auf den in der Anwendung festgelegen Ablauf.

Aufgrund der Vielzahl unterschiedlich eingesetzter Webtechnologien, erweist es sich als Hürde mit einem festgelegten Ablauf beim Web Scraping an die benötigten Daten heranzukommen. Im Januar 2020 wurden grundlegende Erneuerungen der Internetpräsenz von Spiegel.de durchgeführt. Nun werden die Nutzerkommentare nicht mehr im gleichen Schritt mit der Website geladen, sondern werden erst in einem separaten Inlineframe mit Ajax nach einem expliziten Aufruf nachgeladen. Ähnlich verhält es sich bei Zeit.de. Dort wird wiederrum nur ein Teil der Nutzerkommentare mit der Website geladen und einzelne Gesprächsflüsse und Reaktionen auf Kommentare müssen durch einen aktiven Aufruf mit einem Klick nachgeladen werden.

## 3.3 Anwendungsfälle

In diesem Anschnitt werden mögliche Anwendungsfälle des Benutzers aufgezeigt. Anwendungsfälle beschreiben eine Handlung zwischen dem Benutzer und einem System, um das beschriebene Ziel zu erreichen. Zu beachten ist, dass die beschriebenen Abläufe möglichst simpel gehalten werden.

**Nutzerkommentare extrahieren**

| Name des Anwendungsfalls | Nutzerkommentare extrahieren |
|---|---|
| Nummer | U1 |
| Kurzbeschreibung | Der Benutzer hat die Möglichkeit Einstellungen in der Anwendung zu tätigen und den Extraktionsvorgang zu starten. |
| Vorbedingungen | Die Anwendung wurde gestartet. |
| Nachbedingungen | Die gewählten Einstellungen wurden von der Anwendung für die Datenextraktion übernommen. |
| Typischer Ablauf | 1. Der Benutzer wählt eine Datenquelle aus.<br>2. Der Benutzer trägt die URL der Website ein.<br>3. Der Benutzer wählt als Exportmethode das Datenformat „CSV“ aus.<br>4. Der Benutzer startet die Extraktion der Nutzerkommentare. |
| Alternativer Ablauf | Der Benutzer kann eine andere Datenquelle für die Extraktion bestimmen.<br>Der Benutzer kann als weitere Exportmethode eine Datenbankverbindung anlegen. |

**Datenquellen verwalten**

| | |
|---|---|
| Name des Anwendungsfalls | Datenquellen verwalten |
| Nummer | U2 |
| Kurzbeschreibung | Der Benutzer hat die Möglichkeit neue Datenquellen hinzuzufügen, bestehende Datenquellen zu verändern oder zu entfernen. |
| Vorbedingungen | Die Anwendung wurde gestartet.<br>Um eine Datenquelle zu verändern oder zu entfernen muss diese vorher angelegt worden sein. |
| Nachbedingungen | Neue angelegte Datenquellen werden in der Anwendung erfolgreich hinzugefügt.<br>Zustände von bearbeiteten Datenquellen werden übernommen.<br>Entfernte Datenquellen werden nicht mehr angezeigt. |
| Typischer Ablauf | 1. Der Benutzer wählt „Hinzufügen“ aus um eine neue Datenquelle zu hinterlegen.<br>2. Der Benutzer legt die Datenquelle mit den analysierten Websitedaten an.<br>3. Der Benutzer kann die hinterlegte Datenquelle für eine Datenextraktion verwenden. |
| Alternativer Ablauf | Der Benutzer kann bereits hinterlegte Datenquellen bearbeiten und verändern.<br>Der Benutzer kann bereits hinterlegte Datenquellen entfernen. |

**Exportmethode „Datenbank“ verwenden**

| | |
|---|---|
| Name des Anwendungsfalls | Exportmethode „Datenbank“ verwenden |
| Nummer | U3 |
| Kurzbeschreibung | Der Benutzer hat die Möglichkeit eine Datenbankverbindung hinzuzufügen, um die Nutzerkommentare an die Datenbank zu exportieren. |
| Vorbedingungen | Die Anwendung wurde gestartet.<br>Zugriffsdaten für die Datenbank müssen hinterlegt werden. |
| Nachbedingungen | Die Zugriffsdaten werden dauerhaft in der Anwendung gespeichert. |
| Typischer Ablauf | 1. Der Benutzer wählt Funktionalität zur Bearbeitung der Datenbankverbindung aus.<br>2. Der Benutzer hinterlegt Zugriffsdaten der Datenbank.<br>3. Der Benutzer speichert die Daten und verlässt die Funktionalität.<br>4. Der Benutzer kann nun als Exportmethode „Datenbank“ verwenden. |
| Alternativer Ablauf | Die Zugriffsdaten der Datenbank sind bereits hinterlegt. Der Benutzer braucht nur als Exportmethode „Datenbank“ auswählen. |

**User-Agent Profile verwalten**

Beim Aufrufen von Websites senden Webbrowser in der Anfrage an den Server unterschiedliche Metadaten. Ein relevanter Bestandteil davon nennt sich „User-Agent“. Dort werden Name und Version des verwendeten Webbrowsers mitgeteilt. Um nicht als Datenschürfer erkannt zu werden, bedienen sich Web Scraping Anwendung häufig der Methode in ihrer Anfrage ein User-Agent Profil eines gewöhnlichen Webbrowsers mitzuteilen. [27]

| Name des Anwendungsfalls | User-Agent Profile verwalten |
|---|---|
| Nummer | U4 |
| Kurzbeschreibung | Der Benutzer hat die Möglichkeit ein bevorzugtes User-Agent Profil für das Web Scraping zu nutzen oder ein bereits hinterlegtes Profil zu bearbeiten. |
| Vorbedingungen | Die Anwendung wurde gestartet. |
| Nachbedingungen | Ein ausgewähltes User-Agent Profil wird für die Datenextraktion verwendet.<br>Veränderte User-Agent Profildaten werden dauerhaft in der Anwendung gespeichert. |
| Typischer Ablauf | 1. Der Benutzer wählt User-Agent Profil aus.<br>2. Der Benutzer wählt Funktionalität zum Bearbeiten des User-Agent Profils aus.<br>3. Der Benutzer verändert die User-Agent Profildaten.<br>4. Der Benutzer speichert die Veränderungen. |
| Alternativer Ablauf | Der Benutzer wählt ein User-Agent Profil seiner Wahl für die Datenextraktion aus. |

## 3.4 Funktionale Anforderungen

Funktionale Anforderungen sind Aussagen bezüglich der Funktionalitäten die das System leisten sollte. Es ist wichtig die aufgestellten Anforderungen möglichst konkret zu formulieren, um diese im Anschluss erfolgreich messen zu können. [27] Die Formulierungen der funktionalen Anforderungen reichen von allgemein beschriebenen bis hin zu sehr spezifischen Anforderungen. [28]

**Datenquellen verwalten**

Die Anwendung soll dem Benutzer ermöglichen, aus weiteren Datenquellen (Medienpräsenzen) Nutzerkommentare extrahieren zu können. Der Benutzer soll:

- Datenquellen hinzufügen können
- Datenquellen verändern können
- Datenquellen entfernen können

**Exportmedium festlegen**

Die Anwendung soll dem Benutzer unterschiedliche Arten des Datenexports anbieten können. Der Benutzer soll:

- Nutzerkommentare an eine Datenbank exportieren können
- Nutzerkommentare als Comma-separated-value (CSV) Datei exportieren können

**User-Agent HTTP-Header verwalten**

Die Anwendung soll unterschiedliche User-Agent Profile zur Verschleierung der tatsächlichen Identität anbieten können. Der Benutzer soll:

- User-Agent Profil auswählen können
- User-Agent-Profil verändern können

**Persistente Datenhaltung**

Die Anwendung soll die Möglichkeit bieten, die getätigte Einstellungen zu speichern, welche bei einem Neustart der Anwendung automatisch geladen werden. Zu den Einstellungen gehören hinterlegte, veränderte oder entfernte Datenquellen, eventuelle Zugriffsdaten für eine Datenbank, sowie das Laden der User-Agent Profile. Der Benutzer soll:

- Vorgenommenen Einstellungen beim erneuten Aufrufen der Anwendung wiederfinden

**Aktivität anzeigen**

Die Anwendung soll dem Benutzer durch eine Statusmeldung die aktuelle Aktivität mitteilen. Der Benutzer soll:

- Kenntnis über die aktuell durchgeführte Handlung im Web Scraping Vorgang haben

## 3.5 Nichtfunktionale Anforderungen

Bei nichtfunktionalen Anforderungen wird festgelegt, wie die Eigenschaften des Gesamtsystems umgesetzt werden sollen. Sie ergeben sich aus den Bedürfnissen des Benutzers. Häufig sind nichtfunktionale Anforderungen relevanter als die funktionalen Anforderungen. Denn sofern eine nichtfunktionale Anforderung unerfüllt bleibt, kann dies das gesamte System unbrauchbar werden lassen. [28]

**Zuverlässigkeit**

Grundvoraussetzung für die Verwendung einer Web Scraping Anwendung ist die Zuverlässigkeit der Datenextraktion. Zur Erfassung der korrekten Daten und Informationen ist es notwendig, dass eine systematische Extraktion stets sichergestellt ist.

**Erweiterbarkeit**

Internetseiten sind in einem ständigen Wandel und Entwicklung der Webtechnologien ausgesetzt. Deshalb sollte es möglich sein die Web Scraping Anwendung auf die Strukturen weiterer Medienpräsenzen zu erweitern.

**Benutzbarkeit**

Um die Anwendung für den Benutzer möglichst komfortabel zu erstellen, muss die Bedienung intuitiv erfolgen. Dies schließt ein, dass die Bedienelemente klar strukturiert sind und der Benutzer sich innerhalb der Anwendung schnell zurechtfindet.

# 4 Entwurf

Unter Berücksichtigung der gewonnen Erkenntnisse an die spätere Anwendung wird in diesem Kapitel ein erster Design Entwurf konzipiert.

## 4.1 Hauptansicht

Die Startansicht der Anwendung ist der Einstiegspunkt für den Benutzer. Um die Bedienung möglichst benutzerfreundlich zu gestalten, wird der Aufbau der Anwendung auf die relevanten Bedienelemente reduziert. Die Interaktionsreihenfolge des Benutzers wird folgendermaßen modelliert:


Abbildung 6: Interaktionsreihenfolge Benutzer (eigene Darstellung)

Die Struktur der Bedienelemente in vertikaler Ausrichtung beginnt mit dem Logo der Anwendung. Anschließend befinden in der konzipierten Interaktionsreihenfolge die Datenquellen, individuelle (Datenquellen-) Einstellungen, User-Agent Profil und das Textfeld für den Beitragslink. Mit den unterschiedlichen Schaltflächen „Anlegen“ und

„Bearbeiten“ öffnen sich jeweils neue Anwendungsfenster um den gewünschten Vorgang (anlegen oder bearbeiten) abzuschließen. Abschließend muss der Benutzer die präferierte Exportmethode auswählen und startet mit dem Start-Button den Extraktionsvorgang.


**Abbildung 7: Designentwurf der Benutzeroberfläche**

## 4.2 Datenpersistenz

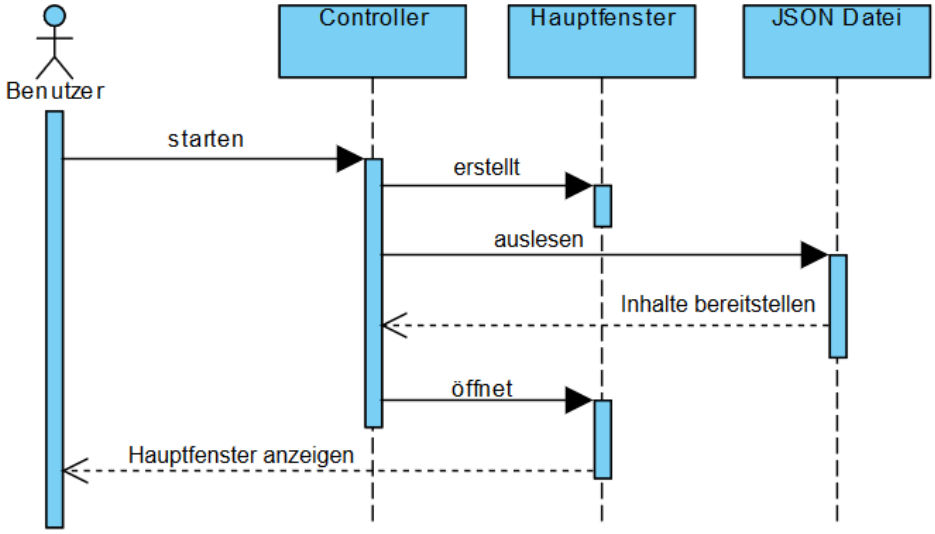


Abbildung 8: Sequenzdiagramm der Datenpersistenz

Das in Abbildung 9 modellierte Sequenzdiagramm soll die Interaktion des Benutzers mit der Benutzeroberfläche in Bezug auf die Datenpersistenz beim Aufrufen der Anwendung aufzeigen. Alle getätigten Anwendungseinstellungen sollen bei jedem erneuten Starten der Anwendung wiederhergestellt werden. Dies soll mit der Verwendung einer JSON-Datei, in dem sämtliche Zustandsveränderungen in der Benutzeroberfläche geschehen, realisiert werden. Die JSON-Datei wird mit dem Starten der Anwendung vom Controller nach dem Erstellen des Hauptfensters ausgelesen und die entsprechenden Einstellungen anhand der Werte wiederhergestellt. Anschließend öffnet sich das Hauptfenster mit den zuletzt getätigten Einstellungen. Die unterschiedlichen Datenquellen sollen innerhalb der JSON-Datei in einer Liste gespeichert werden können. Die individuellen Einstellungen hingegen benötigen nur einen Zustand in Form eines Boolean, String oder Integer.

# 5 Realisierung

Dieses Kapitel beschreibt die prototypisch implementierte Anwendung namens „CommentScraper“. Mit dieser Anwendung können Nutzerkommentare von Medienpräsenzen systematisch extrahiert werden. Zur Veranschaulichung der Funktionsweise von CommentScraper wurden die Internetauftritte von Spiegel.de und Zeit.de herangezogen. Zudem hat der Benutzer grundsätzlich die Möglichkeit weitere Medienpräsenzen als Datenquellen zu hinterlegen. Aufgrund der Tatsache, dass es sich beim CommentScraper um eine prototypische Anwendung handelt, wurde das Thema Fehlerbehandlung kaum berücksichtigt. Fehler können somit zum abrupten Beenden der Anwendung führen. Primär ist die Aufgabe eines Prototyps die Entdeckung und mögliche Lösung von auftretenden Problemen. [27]

## 5.1 Verwendete Werkzeuge

Um eine Web Scraping Anwendung auf die in Kapitel 3 beschriebenen Spezifikationen zu entwerfen, gilt es zunächst sich für eine Programmiersprache zu entscheiden. Darauf aufbauend werden dann entsprechend der Programmiersprache die unterschiedlichen Frameworks und Bibliotheken näher betrachtet werden. Im Rahmen dieser Arbeit und der Realisierung von CommentScraper wurde „Python“ ausgewählt. Python belegt als weltweit beliebteste Programmiersprache laut dem PYPL (PopularitY of Programming Language Index) den ersten Platz. [28]


**Abbildung 9: Beliebteste Programmiersprache der Welt**
**Quelle: [28]**

In Python gibt es viele nützliche Frameworks und Bibliotheken, welche hilfreich für die Konstruktion einer Web Scraping Anwendung sind. Folgende Voraussetzungen müssen gegeben sein:

- Bedienung der Benutzerfläche muss intuitiv erfolgen
- Funktionalität mit modernen Websites muss gegeben sein
- Möglichkeit die Websites beim Extrahieren in Echtzeit zu sehen
- Auswahl geeigneter Exportformate

Bei der Auswahl der richtigen Werkzeuge müssen die geforderten Voraussetzungen und die in Kapitel 3 analysierten Anforderungen erfüllt werden. Folgende Werkzeuge wurden aus diesem Grund ausgewählt:

„PyQt“ ist ein GUI-Toolkit an das beliebte plattformübergreifende Qt-Anwendungsframework, das zur Erstellung plattformübergreifender grafischer Anwendungen verwendet wird. Das duale Lizenzsystem umfasst auch eine kostenfreie Open-Source-Lizensierung. PyQt steht auch für andere Programmiersprachen wie unter anderem Ruby, C#, Java, Perl und noch weitere zur Verfügung. [29]

„Selenium“ dient ursprünglich als ein Framework zum automatisierten Testen von Websites und Webanwendungen. Es wird ebenfalls als freie Software unter der Apache-2.0-Lizenz veröffentlicht und ist mit verschiedenen Programmiersprachen kompatibel. Selenium ist ein gutes Werkzeug für das Web Scraping von modernen Websites mit dynamischen Inhalten und der Verwendung von JavaScript. Es können verschiedenste Interaktionen mit der Website mit Selenium durchgeführt werden. Klicks, Ausfüllen und Absenden von Formularen, Scrollen sind nur einige dieser Möglichkeiten. [30]

„BeautifulSoup“ ist ebenfalls eine unter der Open-Source-Lizenz stehende Python Bibliothek. Mit Hilfe von BeautifulSoup lassen sich die erhaltenen HTML Dokumente des Website Servers parsen. Besonders durch die Vielzahl unterschiedlicher Möglichkeiten Elemente zu finden und untersuchen eignet sich BeautifulSoup für den Einsatz beim Web Scraping. [31]

„Pickle“ ist ein Standartmodul von Python mit dessen Hilfe jedes Python-Objekt in eine Datei gespeichert werden kann. Dadurch werden die Objekte persistent abgelegt und zum benötigten Zeitpunkt eingelesen und aufgerufen. [32]

„MySql-Connector“, „JSON“ und „CSV“ sind bereits fest implementierte Bestandteile von Python. Zusätzliches installieren von Modulen in Python ist somit nicht notwendig. Der MySql-Connector ermöglicht es eine Verbindung zu einer MySql Datenbank herzustellen und beispielsweise die extrahierten Daten in die Datenbank zu schreiben. Mit den Modulen JSON und CSV können Dateien im JSON- oder CSV-Format gelesen und auch geschrieben werden.

## 5.2 Architektur

Bei der Auswahl der Architektur für den CommentScraper wurde auf ein Architekturmuster zurückgegriffen. Architekturmuster sind abstrakte Beschreibungen einer empfohlenen Vorgehensweise, die bereits erfolgreich in Systemen und Umgebungen getestet wurden. [28] Die Implementierung von CommentScraper wurde nach dem Architekturmuster Model-View-Controller (MVC) durchgeführt. Dieses Architekturmuster ermöglicht eine Trennung der Logik von der Benutzeroberfläche und wird in drei Komponenten eingeteilt. Das Model ist für die Logik zuständig und verwaltet die Zustände von Objekten. In der Regel erfolgt dort die Kommunikation mit einer Datenbank. Die View dient der Darstellung von Daten im Model. Im Idealfall kennen sich die View und das Model nicht. Der Controller fungiert als Vermittler zwischen der View und dem Model und nimmt Eingaben des Benutzers entgegen, um die Daten im Model zu manipulieren.

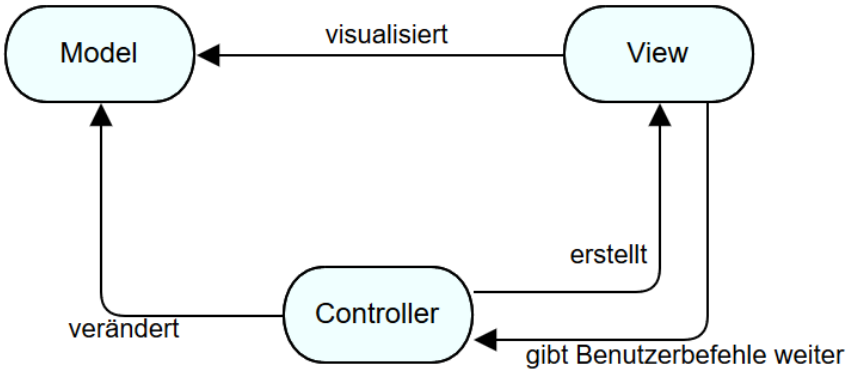


Abbildung 10: Model-View-Controller Architekturmuster

Die View ist die grafische Benutzeroberfläche und besteht aus dem Hauptfenster und diversen Nebenfenstern. Der Controller wird beim Starten der Anwendung initialisiert und erstellt die View, durch die das Hauptfenster geöffnet wird. Getätigte Eingaben in der View nimmt der Controller entgegen, leitet sie an das Model weiter und startet die Klasse Scraper. Diese Klasse beginnt einen neuen Thread damit die grafische

Benutzeroberfläche während der Datenextraktion nicht blockiert wird. Jegliche Einstellungen, Datenquellen und Zustände werden in die JSON-Datei geschrieben und zur Darstellung für die View bereitgestellt. Lediglich das Passwort für eine hinterlegte MySQL Datenbank wird mit dem Pickle-Modul separat in einem Byte Steam gespeichert, um zu verhindern, dass das Passwort in Klarschrift zu finden ist.

## 5.3 Benutzeroberfläche

Die Benutzeroberfläche vom CommentScraper soll durch klare Strukturen der einzelnen Komponenten intuitiv zu bedienen sein. Die Umsetzung erfolgt mit Hilfe von PyQt5. Zusätzlich zu dem Hauptfenster bestehen weitere Dialogfenster, die durch die Auswahl der Schaltflächen „Anlegen“ oder „Bearbeiten“ aufgerufen werden.


Abbildung 11: CommentScraper Benutzeroberfläche

**Hauptfenster**

Das Hauptfenster besitzt eine Mindestgröße und kann beliebig bis zur Vollbildansicht maximiert werden. Die einzelnen Komponenten passen sich der Fenstergröße dynamisch an. Der Aufbau der einzelnen Komponenten erstreckt sich im Hauptfenster vertikal. Unterhalb der Menüzeile befindet sich zentriert das CommentScraper Logo.

**Titelleiste**

In der Titelleiste befindet sich das Logo als Icon und der Name der Anwendung „CommentScraper“.

**Menüzeile**

In der Menüzeile befinden sich die Menüpunkte „File“ und „Export“. Mit der Auswahl „File“ gibt es die Auswahl einer neuen Datenquelle zu hinterlegen, eine bereits bestehende Datenquelle zu bearbeiten oder. das Hauptfenster mit „beenden“ zu schließen. Im „Export“ Menüpunkt befindet sich die Option „MySQL Verbindung“. Wird dies ausgewählt öffnet sich das Fenster zum Speichern einer MySQL Datenbankverbindung.

**Datenquellen**

In der Komponente „Datenquelle“ befindet sich ein Dropdown Menü und Buttons zur Verwaltung der Datenquellen. Mit einem Dropdown Menü kann eine bestimmte Datenquelle für den nächsten Scraping Vorgang ausgewählt werden.

- Button „Neue Datenquelle“ öffnet das Fenster zum Anlegen einer neuen Datenquelle
- Button „Datenquelle bearbeiten“ öffnet das Fenster zum Bearbeiten einer bereits bestehenden Datenquelle
- Button „Datenquelle entfernen“ öffnet ein Fenster um zu bestätigen, dass die aktuelle ausgewählte Datenquelle entfernt werden soll

**Webbrowser-Einstellungen**

In der Komponente Webbrowser-Einstellungen befinden sich Checkboxen und ein Schieberegler um das Verhalten des Webbrowsers zu verändern. Beim Aktivieren der Checkbox „Webbrowser anzeigen“ wird beim Scraping Vorgang das Webbrowser Fenster geöffnet. Beim Aktivieren der Checkbox „Cookie Banner“ wird der Webbrowser beim Erscheinen eines Cookie-Hinweises, den Hinweis akzeptieren und den Cookie-Banner schließen. Die Hinterlegung der korrekten CSS-Klasse in der Datenquelle ist vorausgesetzt. Beim Aktivieren der Checkbox „Versteckte Kommentare“ werden Nutzerkommentare, welche erst durch eine Interaktion mit einem Klick auf eine bestimmte Sektion geladen werden, aufgerufen. Dies ermöglicht, dass auch diese mit JavaScript nachgeladenen Kommentare erfolgreich extrahiert werden können. Der Schieberegler „Aktionspause“ definiert eine Pause in der gewählten Anzahl in Sekunden zwischen unterschiedlichen Interaktionen des Webbrowsers.

**User-Agent-Profil**

Die Komponente „User-Agent-Profil“ besteht aus einem Dropdown Menü und einem Button. Mit dem Dropdown Menü kann der aktuell zu verwendende User-Agent Header ausgewählt werden. Mit dem Button „Bearbeiten“ kann das aktuell ausgewählte User-Agent-Profil aktualisiert werden.

**URL – Textfeld**

Im Textfeld „URL“ wird für die Datenextraktion die URL zu dem journalistischen Beitrag eingegeben.

**Exportmethode**

In der Komponente „Exportmethode“ hat der Benutzer die Auswahl zwischen zwei Exportmöglichkeiten. Der Export kann als CSV-Datei oder an eine MySQL Datenbank erfolgen.

**Statuszeile**

In der Statuszeile unterhalb der Exportmethoden wird die aktuell durchgeführte Handlung und die erfassten Nutzerkommentare angezeigt. Eine Handlung wird solange angezeigt, bis eine nächste relevante Handlung durchgeführt wird. Es existieren folgende Statusmeldungen:

| Statusanzeige | Beschreibung |
|---|---|
| „-„ | Die Anwendung ist im Standby und führt aktuell keine Handlung aus. |
| „Scraping“ | Die Anwendung erfasst die lokalisierten Nutzerkommentare. |
| „Abgeschlossen“ | Der Scraping-Prozess ist abgeschlossen. |
| „Website aufrufen“ | Eine Anfrage an den Server zum Aufbau der Website wurde gestellt. |

**START – Schaltfläche**

Der Button „START“ führt das Scraping-Szenario mit den zu dem Zeitpunkt ausgewählten Einstellungen aus. Alle Komponenten des Hauptfensters werden deaktiviert, so dass keine weiteren Veränderungen möglich sind. Die Beschriftung wechselt zu „STOP“ um das Scraping-Szenario zu stoppen. Nach Abschluss der Scraping Szenarios oder dem manuellen Stoppen des Vorgangs, werden die grafischen Komponenten wieder aktiviert.

## 5.4 Medienpräsenz untersuchen

Der CommentScraper kann um mehrere Datenquellen erweitert werden. Hierfür werden die Klassen von den entsprechenden Elementen benötigt. Jeder Browser bietet die Möglichkeit, Websites zu untersuchen. Mit einem Rechtsklick auf das benötigte Element (Nutzerkommentar-Container, Nutzername, Kommentar) öffnet sich das Kontextmenü des Browsers. Der Menüeintrag des Kontextmenüs von Webbrowsern wie Google Chrome, Mozilla Firefox oder dem Internet Explorer nennt sich „Element untersuchen“.

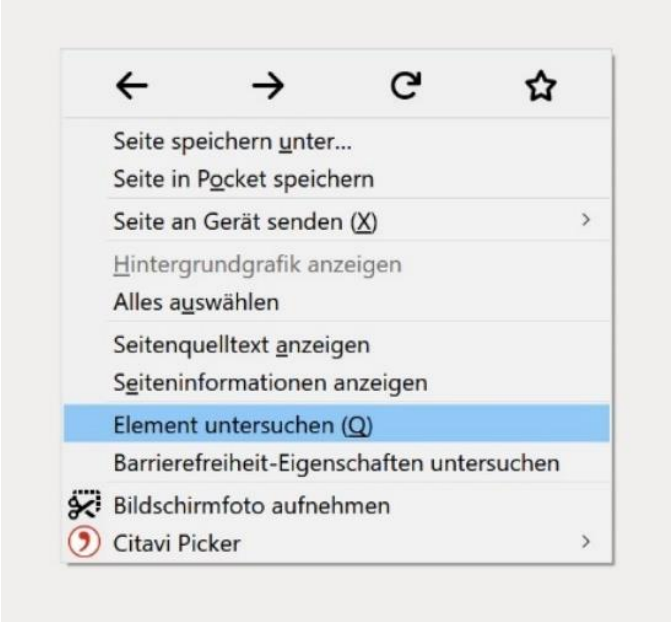


Abbildung 12: Element untersuchen im Webbrowser

Bei Ausführung dieser Funktion öffnet sich die Entwickleransicht im Webbrowser. Dort kann der Aufbau jeder Website genauer analysiert werden. Außerdem wird der komplette Quellcode in der Entwickleransicht strukturiert angezeigt. Abhängig von der Platzierung des Rechtsklicks, wird das Element in den Vordergrund gestellt, das zum Untersuchen ausgewählt wurde.

Der Benutzer erhält dort eine Vielzahl von Informationen über die aktuelle Website beispielsweise in Form der HTML Struktur, Einblick in das Cascading Stylesheet und den Netzwerkverkehr zwischen Website und Server. Für den CommentScraper werden grundsätzlich die Element-Klassen von dem iterierbaren Nutzerkommentar-Container und den darin befindlichen Nutzernamen sowie Kommentar benötigt.

In den folgenden Abbildungen wird in der Entwickleransicht ein Artikel auf der Spiegel.de Internetseite untersucht. Die Auswahl zeigt den aktuell angewählten Nutzerkommentar-Container mit der Element-Klasse.

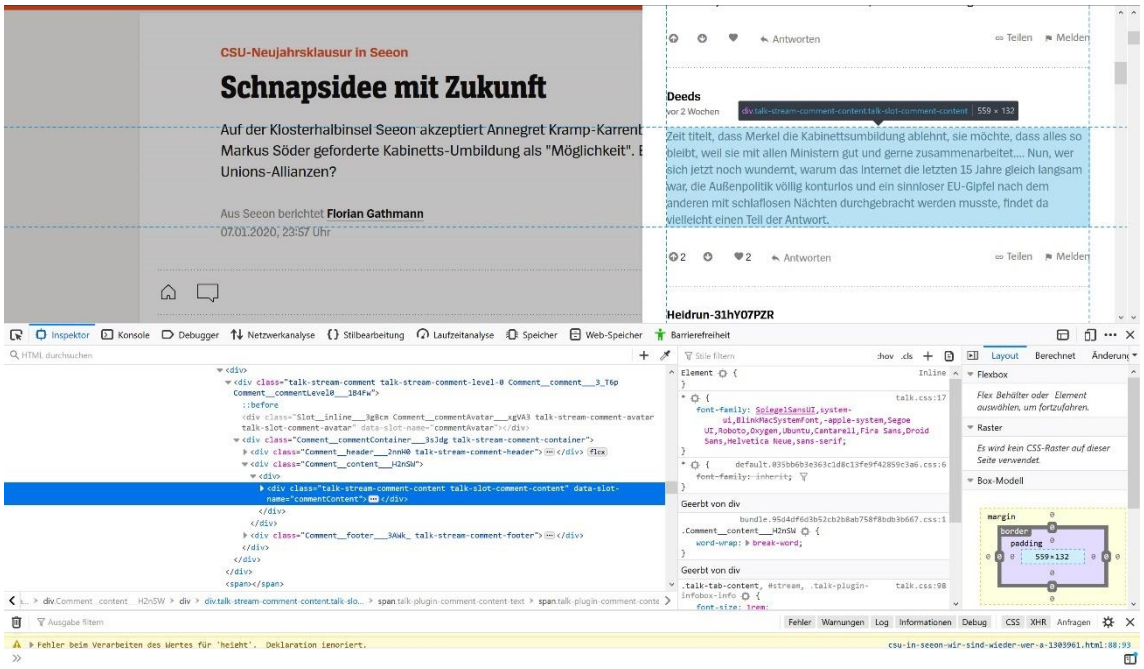


**Abbildung 13: Entwickleransicht Website**

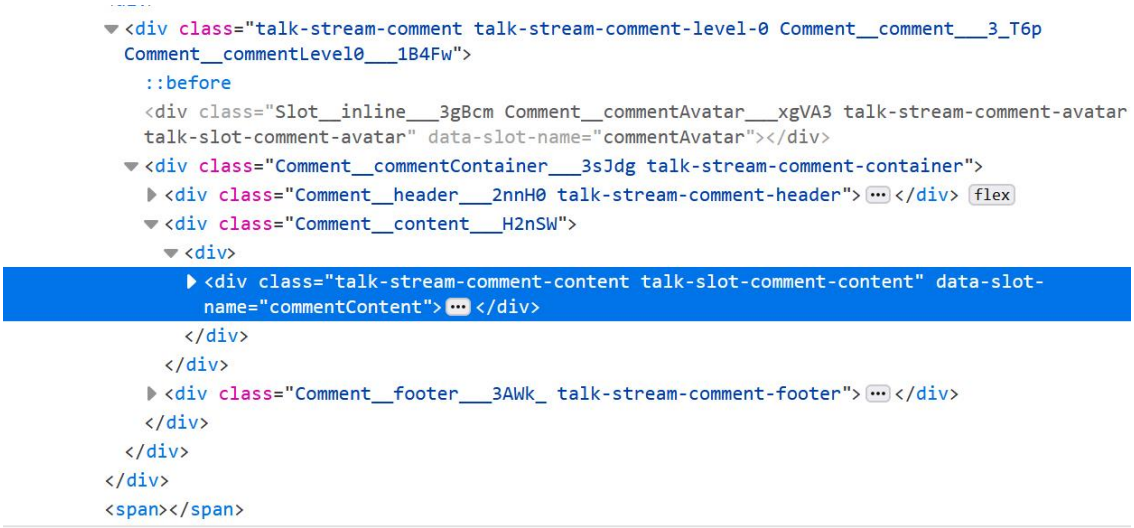


Abbildung 14: Detailansicht des HTML Dokumentes

Aus der Medienpräsenz Spiegel.de werden nach der Analyse folgende Werte festgehalten:

| **Bestandteil** | **Klasse** |
|---|---|
| Nutzerkommentar-Container | .talk-stream-comment |
| Nutzername | .talk-slot-comment-author-name |
| Kommentar | .talk-slot-comment-content |

## 5.5 Bedienung der Anwendung

In diesem Abschnitt werden die unterschiedlichen möglichen Anwendungsfälle des Benutzers mit der Anwendung CommentScraper dargestellt.

### 5.5.1 Datenquellen verwalten

Um eine neue Datenquelle anzulegen muss im Hauptfenster die Schaltfläche „Anlegen“ ausgewählt werden. Es erscheint ein Dialogfenster „Datenquelle anlegen“. Hier können nun vom Benutzer die entsprechenden Textfelder ausgefüllt werden.

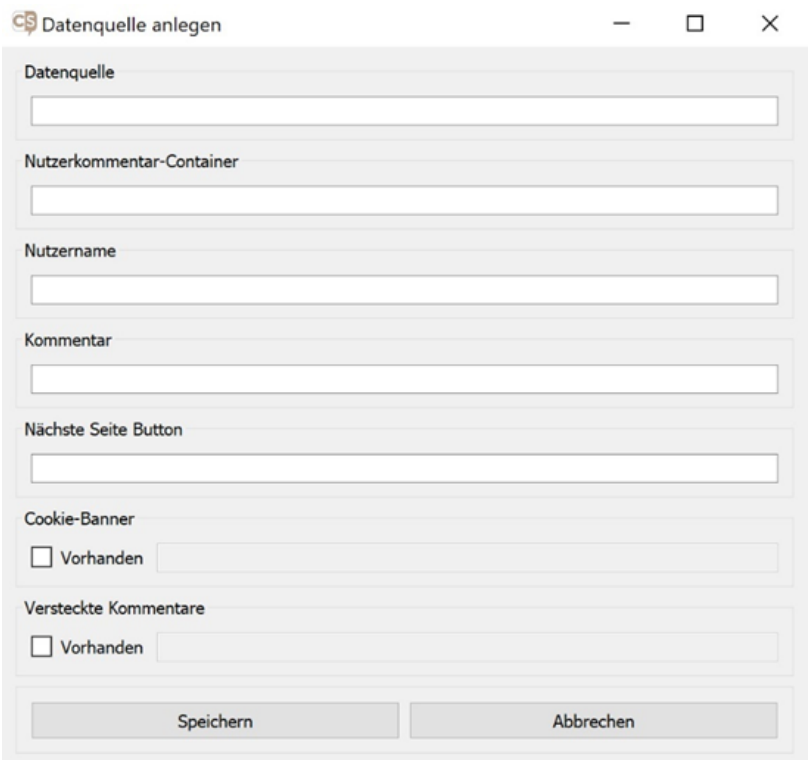


Abbildung 15: Datenquelle anlegen im CommentScraper

Folgende Inhalte werden durch den Benutzer in die Textfelder eingetragen:

| Bezeichnung | Eingabe des Benutzers |
|---|---|
| Datenquelle | Name der Datenquelle. Kann frei gewählt werden. |
| Nutzerkommentar-Container | Klassenbezeichnung des Containers welcher die Bestandteile Nutzername und Kommentar enthält. |
| Nutzername | Klassenbezeichnung welche dem Nutzernamen zugeordnet wurde. |
| Kommentar | Klassenbezeichnung welche dem Kommentar zugeordnet wurde. |
| Nächste Seite Button | Klassenbezeichnung der Schaltfläche welche zur Folgeseite leitet. |
| Cookie-Banner | Klassenbezeichnung der Schaltfläche welche die Cookies im Cookie-Banner akzeptiert. |
| Versteckte Kommentare | Klassenbezeichnung des Bereiches in dem der Benutzer klickt um die Aktion zu starten, dass versteckten Kommentare geladen werden. |

Nach dem Ausfüllen der Textfelder werden die Inhalte mit „Speichern“ dauerhaft in der Anwendung gespeichert. Die Datenquelle ist nun in dem Auswahl-Menü vorzufinden und kann für einen Scraping Vorgang genutzt werden.

### 5.5.2 Konfiguration

Neben der Datenquelle können weitere Einstellungen für die Funktionalität der Anwendung essenziell sein. Diese sind unter der Beschriftung „Webbrowser-Einstellungen“ zu finden. Zur Auswahl stehen drei Checkboxen um die dazugehörige Funktion zu aktivieren.

**Browser unsichtbar**

Standardgemäß ist der interne Webbrowser der Anwendung für den Benutzer nicht wahrzunehmen. Dies kann aber durch das Herausnehmen der Aktivierung unter „Browser unsichtbar“ umgestellt werden. Anschließend wird beim Starten des Scraping Vorganges der Webbrowser angezeigt.

**Cookie-Banner**

Mit „Cookie-Banner“ wird die Anweisung gegeben, eine Interaktion mittels eines Klicks durchzuführen. Dies ist notwendig sofern ein Button für die nächste Seite angewählt werden soll. Denn durch das vorhandene Cookie-Banner kann der Webbrowser, Elemente unter diesem hervorstehenden Banner nicht erreichen.

**Versteckte Kommentare**

Beim Aufrufen der Website kann es der Fall sein, dass nicht alle Inhalte in den Nutzerkommentaren angezeigt werden. Dies kann durch die Programmierung und der verwendeten Technologie der Website so vorgesehen sein. Beispielsweise bei dem Einsatz von Ajax werden die fehlenden Inhalte erst durch eine aktive Handlung (Klick auf das Element) des Benutzers nachgeladen. Dies wird mit der Aktivierung dieser Funktion simuliert und alle fehlenden Inhalte abgerufen.

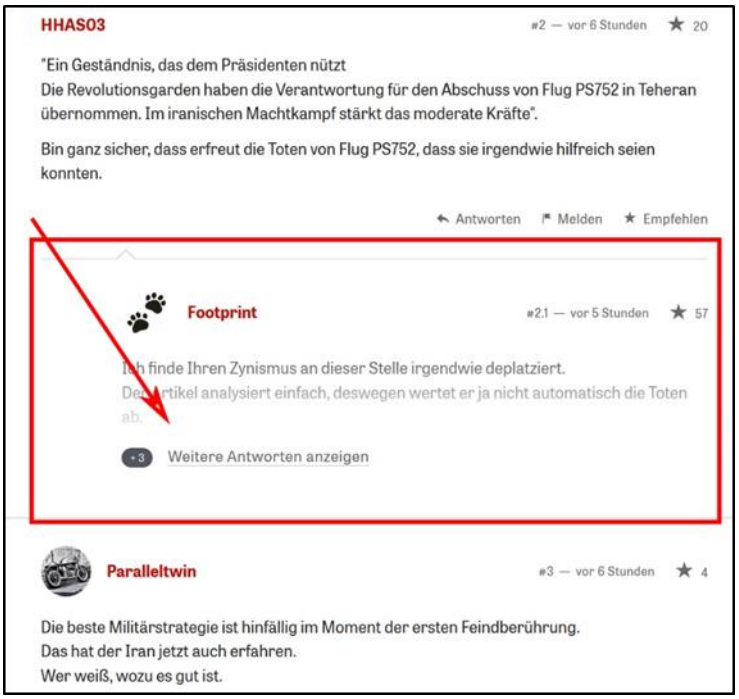


Abbildung 16: Versteckte Nutzerkommentaransicht auf der Website

**Aktionspause**

Beim Aufrufen dieser Inhalte ist es teilweise erforderlich, dass der Programmablauf wartet, bevor die nächste Handlung im Scraping Prozess durchgeführt wird. Aus diesem Grund gibt es den Schieberegler mit dem eine Aktionspause festgelegt werden kann. Dieser definiert an diesen empfindlichen Stellen, dass die angegebene Dauer an Sekunden gewartet wird, bis es im Programmablauf weiter geht. Der Schieberegler kann zwischen Null und Zehn Sekunden festgelegt werden.

**User-Agent Profil**

Um nicht als Web Scraping Anwendung oder eine automatisierte Software erkannt zu werden, können in den Metadaten des internen Webbrowsers ein User-Agent Profil hinterlegt werden. Dieses simuliert der besuchten Website, dass der Aufruf der Internetseite durch einen bestimmten Webbrowser erfolgt.

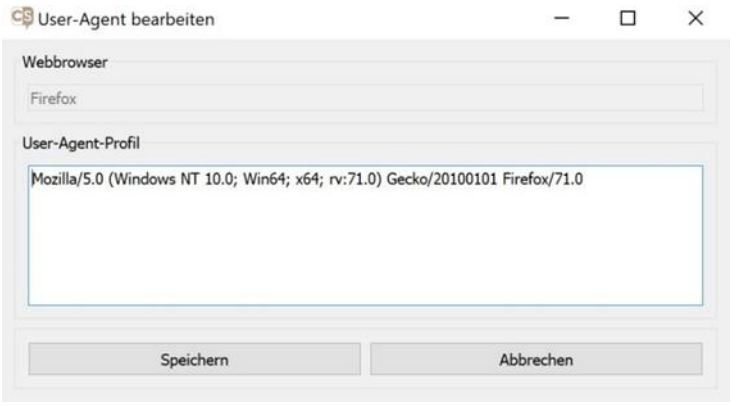


Abbildung 17: User-Agent Profil bearbeiten im CommentScraper

### 5.5.3 Export der Nutzerkommentare

Im letzten Schritt hat der Benutzer die Möglichkeit eine Exportmethode auszuwählen. Es kann zwischen CSV und Datenbank gewählt werden. Es ist nur eine Exportmethode zur selben Zeit auswählbar.


Abbildung 18: Auswahl der Exportmethode im CommentScraper

Bei der Exportmethode CSV werden die extrahierten Nutzerkommentare als CSV-Datei gespeichert. Der Dateiname wird dabei automatisch in Form eines Zeitstempels von der Anwendung festlegt. Jedes Nutzerkommentar wird in eine eigene Zeile geschrieben. Das Dateiformat CSV beschreibt eine Textdatei zur Speicherung einfach strukturierter Daten. Tabellenkalkulationsprogramme und Datenbanken sind fähig CSV-Dateiformate einzulesen, exportieren und diese Daten weiterzuverarbeiten.

Als weitere Exportmethode steht „Datenbank“ zur Verfügung. Um Datenbank nutzen zu können ist es vorher notwendig, in der Menüzeile „Export“ und „MySQL Verbindung“ die Zugriffsdaten für eine Datenbank zu hinterlegen. Dies kann im Dialogfenster

„Datenbankverbindung bearbeiten“ durchgeführt werden. Dort sind alle Felder bis auf das Passwort in Klarschrift zu sehen. Die hinterlegten Zugriffsdaten werden, wie auch die Datenquellen, dauerhaft gespeichert und sind bei einer erneuten Ausführung der Anwendung vorhanden.

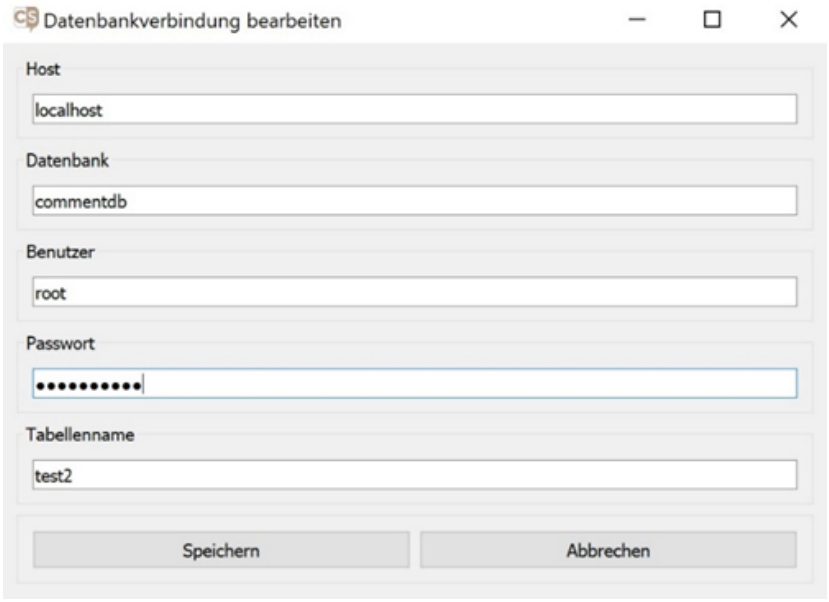


**Abbildung 19: Datenbankverbindung bearbeiten im CommentScraper**

## 5.6 Systematische Abfolge

Das Scraping-Szenario unterliegt einer systematischen Abfolge. Die einzelnen Schritte lassen sich wie folgt einordnen:

| Nr. | Werkzeug | Vorgehensweise |
|---|---|---|
| 1. | Selenium | Zu Beginn wird die eingegebene URL aufgerufen und die Website vollständig geladen. |
| 2. | Selenium | Sofern der Benutzer angegeben hat, dass die Website einen Cookie-Banner enthält, wird dieser mit einer Klick-Interaktion geschlossen. |
| 3. | Selenium | Sofern der Benutzer angegeben hat, dass die Website verdeckte Kommentare enthält, welche nur durch eine Interaktion mit der Website mittels JavaScript/Ajax im Quellcode erscheinen, werden diese nun iterativ abgerufen und folglich auch dem Quellcode hinzugefügt. |
| 4. | BeautifulSoup | Der gesamte Quellcode der Website wird geparst. |
| 5. | BeautifulSoup | Anschließend werden alle im Quellcode befindlichen Nutzerkommentare extrahiert. |
| 6. | Selenium | Sind keine Nutzerkommentare mehr vorhanden, wird geprüft ob eine Folgeseite existiert. Sollte eine weitere Seite vorhanden sein, wird diese aufgerufen und erneut das Szenario ab Schritt zwei durchgeführt. |
| 7. | JSON<br>Mysql.connector | Abschließend werden die Nutzerkommentare in das ausgewählte Dateiformat exportiert. Das Scraping-Szenario ist nun abgeschlossen. |

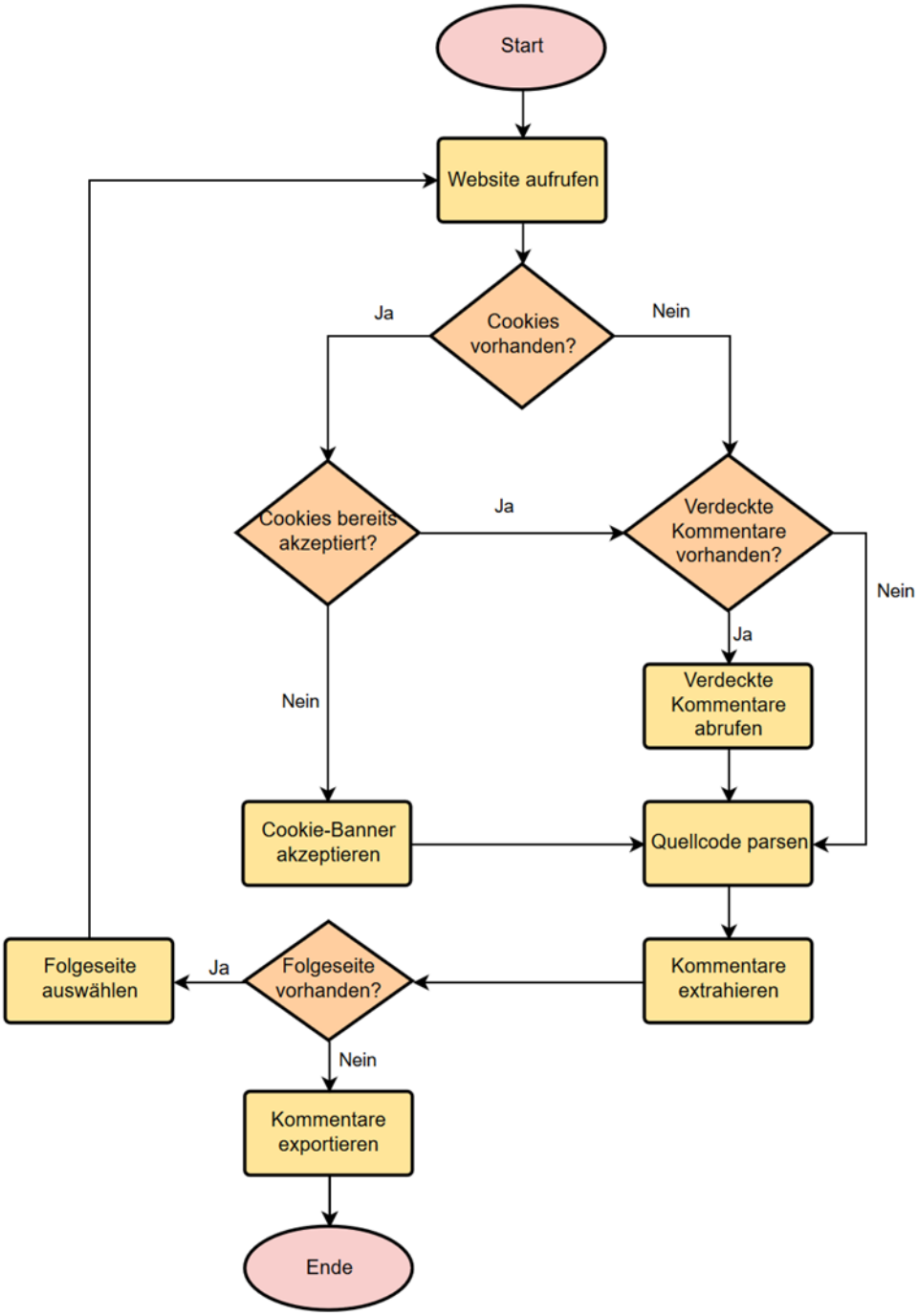


Abbildung 20: Systematische Abfolge vom Scrapingszenario

# 6 Evaluierung

In diesem Kapitel wird die Anwendung CommentScraper evaluiert. Zu Beginn werden diverse Kriterien für eine fachgerechte Bewertung festgelegt. Daraufhin werden die Medienpräsenzen für die Evaluierung ausgewählt und im Anschluss die Ergebnisse der Evaluierung vorgestellt. Es soll herausgefunden werden, ob und in welchem Maße sich der CommentScraper für die Extraktion von Nutzerkommentaren eignet. Es werden vier Medienpräsenzen mit Diskussionsforen ausgewählt, um die systematische Extraktion von Nutzerkommentaren zu überprüfen. Die Herangehensweise beim Testen lautet wie folgt:

1. Untersuchung der Medienpräsenz
2. Medienpräsenz als Datenquelle hinzufügen
3. Anpassung der Browser-Einstellungen
4. Extraktion der Nutzerkommentare
5. Auswertung der Daten

Zu Beginn werden die Medienpräsenzen mit der in Kapitel 5.4 beschriebenen Methode untersucht und die benötigten Klassen aus den Elementen erfasst. Als nächstes werden die Medienpräsenzen als eine neue Datenquelle in der Anwendung erfasst. Sofern nötig, werden die zur Verfügung stehenden Anpassungen in den Browser-Einstellungen genutzt. Im nächsten Schritt kann die Extraktion der Nutzerkommentare beginnen. Abschließend werden die Daten auf Vollständigkeit überprüft und ausgewertet.

## 6.1 Bewertungskriterien

Die Bewertungskriterien sollen unterschiedliche Betrachtungswinkel umfassen.
Hierzu gehört zum einem die Überprüfung ob die Funktionalen- und Nichtfunktionalen Anforderungen erfüllt wurden. Zum anderen die Überprüfung der Anwendung in Hinsicht auf die Kompatibilität mit der Datenquelle sowie der Vollständigkeit der extrahierten Nutzerkommentare.

Das Kriterium „Funktionale- und nichtfunktionale Anforderungen“ kann folgende Werte annehmen:

- „Vollständig erfüllt“ falls die Anforderung erfüllt wurde
- „Nicht erfüllt“ falls die Anforderung nicht erfüllt wurde

Das Kriterium „Kompatibilität mit der Datenquelle“ kann folgende Werte annehmen:

- „Kompatibel“ falls mit der Datenquelle eine unterbrechungsfreie Extraktion der Nutzerkommentare möglich war
- „Inkompatibel“ falls mit der Datenquelle die Extraktion der Nutzerkommentare unterbrochen wurde

Das Kriterium „Vollständigkeit der extrahierten Nutzerkommentare“ kann folgende Werte annehmen:

- „Vollständig“ falls alle Nutzerkommentare vollständig extrahiert wurden
- „Unvollständig“ falls nur ein Teil die Nutzerkommentare extrahiert wurden
- „Nicht vorhanden“ falls keine Nutzerkommentare extrahiert wurden

## 6.2 Ergebnisse

Im Folgenden wurden die Ergebnisse der funktionalen und nichtfunktionalen Anforderungen tabellarisch dargestellt:

| **Funktionale Anforderungen** | **Beschreibung** |
|---|---|
| Datenquellen verwalten | Vollständig erfüllt |
| User-Agent HTTP-Header verwalten | Vollständig erfüllt |
| Persistente Datenhaltung | Vollständig erfüllt |
| Aktivität anzeigen | Vollständig erfüllt |

| **Nichtfunktionale Anforderung** | **Beschreibung** |
|---|---|
| Zuverlässigkeit | Vollständig erfüllt |
| Erweiterbarkeit | Vollständig erfüllt |
| Benutzbarkeit | Vollständig erfüllt |

Die Medienpräsenzen Spiegel.de und Zeit.de wurden als Datenquellen verwendet. Bei jeder Datenquelle wurden drei zufällig ausgewählte journalistischen Beiträge zwischen 50 bis 400 Nutzerkommentaren ausgesucht.

| Datenquelle | Kompatibilität mit der Datenquelle | Vollständigkeit der extrahierten Nutzerkommentare |
|---|---|---|
| Spiegel.de | Kompatibel | Vollständig |
| Zeit.de | Kompatibel | Vollständig |

**Spiegel.de**

| Beitragslink | Extrahierte Nutzerkommentare |
|---|---|
| https://www.spiegel.de/politik/deutschland/csu-in-seeon-wir-sind-wieder-wer-a-1303961.html | 67/67 |
| https://www.spiegel.de/wirtschaft/unternehmen/tesla-an-der-boerse-wertvoller-als-volkswagen-a-f1fcdd99-f382-4743-addd-dbc1a8216cb1 | 119/119 |
| https://www.spiegel.de/auto/e-fuels-die-riskante-wette-auf-synthetische-kraftstoffe-a-82ffad81-b84e-4ef7-a330-634cdfd1a2f5 | 212/212 |

**Zeit.de**

| Beitragslink | Extrahierte Nutzerkommentare |
|---|---|
| https://www.zeit.de/arbeit/2020-01/deutscher-gewerkschaftsbund-dgb-arbeit-befristung-probezeit-studie | 165/165 |
| https://www.zeit.de/wissen/gesundheit/2019-11/masern-impfpflicht-deutschland-gesundheit-bundestag | 317/317 |
| https://www.zeit.de/2019/36/niedrigzinsen-notenbanken-ezb-finanzkrise-kreditvergabe-inflation | 281/281 |

Der CommentScraper hat bei den ausgewählten Beiträgen sämtliche Nutzerkommentare erfolgreich erfasst und extrahiert. In jedem Extraktionsvorgang wurden abwechselnd beide Exportmethoden (CSV und Datenbank) ausgewählt und anschließend auf Vollständigkeit überprüft.

# 7 Schluss

## 7.1 Zusammenfassung

Im Verlauf dieser Arbeit wurde das Web Scraping betrachtet und eine prototypische Anwendung für die systematische Extraktion von Nutzerkommentaren mit dem Fokus auf Medienpräsenzen entwickelt. Beginnend mit den Grundlagen wurden ebenso auftretende Problematiken beim Web Scraping aufgezeigt. Im Anschluss wurden die unterschiedlichen Medienpräsenzen als potenzielle Datenquelle der Nutzerkommentare analysiert und erste Anwendungsfälle sowie Anforderungen an die zukünftige Anwendung gestellt. Nach der Analyse und dem Entwurf erfolgte die Realisierung der Anwendung. Diese wurde als Desktop-Anwendung plattformübergreifend in Python entwickelt. Abschließend wurde die Anwendung unter festgelegten Bewertungskriterien evaluiert und die Ergebnisse vorgestellt. Positiv angewandt bietet Web Scraping die Möglichkeit, mit Inhalten anderer Websites einen hohen Mehrwert für viele Zwecke.

## 7.2 Ausblick

Mit den extrahierten Nutzerkommentaren des CommentScraper, könnten zu ausgesuchten Themengebieten beispielsweise Verhaltens- oder Sprachanalysen durchgeführt werden. Die gewonnen Informationen eignen sich darüber hinaus als Grundlage zum Erstellen von Profilen oder Stereotypen. Die Anwendung könnte zudem um weitere zukünftig benötigte Funktionen erweitert werden.

# Literaturverzeichnis


[1] Statista, „Anzahl der Webseiten weltweit in den Jahren 1992 bis 2015," Internet Live Stats; Netcraft, März 2016. [Online]. Available: https://de.statista.com/statistik/daten/studie/290274/umfrage/anzahl-der-webseiten-weltweit/. [Zugriff am 10 01 2020].

[2] M. Ziegele, Nutzerkommentare als Anschlusskommunikation, Springer Fachmedien Wiesbaden GmbH, 2015, p. 15.

[3] S. Munzert, C. Rubba, P. Meißner und D. Nyhuis, „Automated data collection with R," in *A practical guide to web scraping and text mining*, Chichester, Wiley, 2015.

[4] R. Mitchell, Web Scraping with Python: Collecting Data from the Modern Web, O'Reilly Media, Inc., 2015.

[5] G. Barcaroli und S. D. Scannapieco Monica, „On the use of internet as a data source for official statistics: A strategy for identifying enterprises on the web," Rivista Italiana di Economia Demografia e Statistica, 2016.

[6] X. Yuan, M. H. MacGregor und J. Harms, „An Efficient Scheme to Remove Crawler Traffic from the Internet," IEEE, Miami, FL, USA, USA, 2002.

[7] R. Baumgartner, „Methoden und Werkzeuge zur Webdatenextraktion," in *Semantic Web — Geschichte und Ausblick einer Vision*, Springer Berlin Heidelberg, 2006, pp. 420-435.

[8] J. Seipel, „Webscraping als Methode der Informationsbeschaffung," Verein für Medieninformationen und Mediendokumentation, 2016.

[9] S. Münz, Professionelle Websites, Addison-Wesley Verlag, 2006.

[10] World Wide Web Consortium, „https://www.w3.org," [Online]. Available: https://www.w3.org/standards/webdesign/htmlcss. [Zugriff am 21 November 2019].

[11] P. Ackermann, JavaScript - Das umfassende Handbuch, Rheinwerk Computing, 2017.

[12] World Wide Web Consortium, „https://www.w3.org," [Online]. Available: https://www.w3.org/standards/webdesign/script. [Zugriff am 21 November 2019].

[13] J. Wolf, HTML5 und CSS - Das umfassende Handbuch, Rheinwerk Computing, 2016.

[14] Computerwoche, 2010. [Online]. Available: https://www.computerwoche.de/a/google-startet-instant-suche. [Zugriff am 24 November 2019].

[15] A. Blätte, J. Behnke, K.-U. Schnapp und C. Wagemann, Computational Social Science: Die Analyse von Big Data (Schriftenreihe Der Sektion Methoden Der Politikwissenschaft), Nomos, 2018.

[16] S. J. Dr. Golla und M. Dr. v. Schönfeld, „Kratzen und Schürfen im Datenmilieu. Web Scraping in sozialen Netzwerken zu wissenschaftlichen Forschungszwecken," DFV Mediengruppe, Berlin, 2019.

[17] C. Solmecke, „https://www.wbs-law.de,“ 15 Mai 2013. [Online]. Available: https://www.wbs-law.de/urheberrecht/ist-screen-scraping-legal-15081/. [Zugriff am 01 Dezember 2019].

[18] C. Dr. Ulbricht, „http://www.rechtzweinull.de,“ 08 Juni 2009. [Online]. Available: http://www.rechtzweinull.de/archives/100-screen-scraping-wann-ist-das-auslesen-und-die-veroeffentlichung-fremder-daten-zulaessig.html. [Zugriff am 2019 Dezember 01].

[19] M. Nowak-Trytko und F. Reese, „https://t3n.de,“ 23 November 2012. [Online]. Available: https://t3n.de/magazin/marktuberblick-web-analyse-controlling-tools-wissen-232033/. [Zugriff am 02 Dezember 2019].

[20] J. Ihlenfeld, „https://www.golem.de,“ 12 Dezember 2012. [Online]. Available: https://www.golem.de/news/scraping-google-geht-gegen-seo-tools-vor-1212-96307.html. [Zugriff am 02 Dezember 2019].

[21] World Wide Web Consortium, „https://www.w3.org,“ [Online]. Available: https://www.w3.org/TR/WCAG21/#dfn-captcha. [Zugriff am 02 Dezember 2019].

[22] V. P. Singh und P. Preet, „Survey of Different Types of CAPTCHA,“ International Journal of Computer Science and Information Technologies, 2014.

[23] [Online]. Available: http://www.captcha.net/. [Zugriff am 05 01 2020].

[24] Spiegel, „https://www.spiegel.de/,“ 17 April 2014. [Online]. Available: https://www.spiegel.de/netzwelt/web/google-knackt-den-captcha-code-a-964955.html. [Zugriff am 03 Dezember 2019].

[25] S. Jansen, „https://www.e-recht24.de,“ 22 März 2011 . [Online]. Available: https://www.e-recht24.de/news/datenschutz/6604-impressum-als-grafikdatei.html. [Zugriff am 03 Dezember 2019].

[26] M. Weber, Der soziale Rezipient, Springer VS, 2015.

[27] V. Gehrau und L. Goertz, Gespräche über Medien unter veränderten medialen Bedingungen, Springer, 2010.

[28] Mozilla Corporation, „http://developer.mozilla.org/,“ [Online]. Available: https://developer.mozilla.org/de/docs/Web/HTTP/Headers/User-Agent. [Zugriff am 28 Dezember 2019].

[29] S. Kleuker, Grundkurs Software-Engineering mit UML, Springer Vieweg, 2013.

[30] I. Sommerville, Software Engineering, Pearson Studium, 2018.

[31] F. Suhr, 07 Januar 2019. [Online]. Available: https://de.statista.com/infografik/16544/anteile-der-populaersten-programmiersprachen-weltweit/. [Zugriff am 20 Dezember 2019].

[32] Riverbank Computing Limited, „https://riverbankcomputing.com,“ [Online]. Available: https://riverbankcomputing.com/software/pyqt/intro. [Zugriff am 22 Dezember 2019].

[33] ThoughtWorks, „https://selenium.dev/,“ [Online]. Available: https://selenium.dev/. [Zugriff am 22 Dezember 2019].

[34] L. Richardson, „https://www.crummy.com,“ [Online]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Zugriff am 22 Dezember 2019].

[35] Python Software Foundation, „https://www.python.org,“ [Online]. Available: https://docs.python.org/3/library/pickle.html. [Zugriff am 22 Dezember 2019].

# Abbildungsverzeichnis

# Versicherung über Selbstständigkeit

*Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.*

*Hamburg, den* _____ _____