

Evaluation und Implementierung eines Marketingportals für Hochschulen

Bachelorarbeit

im Studiengang
Digital Media Marketing

vorgelegt von

Sebastian Schuler

am 28. Juni 2023
an der Hochschule Kaiserslautern

Erstprüfer/in:	Prof. Hendrik Speck
Zweitprüfer/in:	Prof. Dr. Daniel Stenger

Kurzfassung

Gegenstand der hier vorgestellten Arbeit ist die Entwicklung einer Online-Plattform, die alle Studiengänge, Hochschulen und Universitäten in Europa auf einer einzigen Plattform zusammenfasst. Diese Plattform ermöglicht es den Besuchern, nach verschiedenen Studiengängen und Hochschulen zu suchen und diese anhand einer Reihe von Kriterien zu bewerten. Die Hochschulen wiederum können die Plattform nutzen, um ihr Angebot strategisch zu bewerben und so ihren Bekanntheitsgrad zu erhöhen und mehr Studieninteressierte anzusprechen. Zur Erreichung der Ziele dieser Plattform zu erreichen, wurde besonderer Wert auf die Suchmaschinenoptimierung, die Integration dynamischer Inhalte und die automatische Aktualisierung der Inhalte gelegt. Um die gewünschten Ergebnisse zu erzielen, wurde Next.js für die Entwicklung der Website und Node.js für die Backend-Funktionalität verwendet. Zusätzlich wurde ein Algorithmus zur Bewertung und Klassifizierung der Social-Media-Profile aller Universitäten und Hochschulen entwickelt. Der Algorithmus verwendet einen umfassenden Kriterienkatalog, um die Social-Media-Aktivitäten der Universitäten gründlich zu analysieren und zu klassifizieren. Er basiert auf den Algorithmen, die in den Forschungsarbeiten "Market, Society, and Education – University Marketing" (Kloppenburger, 2012) und "Evaluation und Implementierung von Social-Media-Ratings von Hochschulen für ein Webportal" (Schuler, 2022) diskutiert wurden. Durch die Kombination von Website-Analyse, Social-Media-Rating und weiteren Ressourcen bietet die Plattform eine umfassende Informationsquelle sowohl für Studieninteressierte als auch für Hochschulen. Die erfolgreiche Entwicklung und Implementierung dieser Online-Plattform ist ein wichtiger Schritt zur Vereinheitlichung der Bildungslandschaft in Europa. Durch die Zentralisierung aller Studiengänge, Hochschulen und Universitäten und die Bereitstellung moderner Bewertungstools bietet die Plattform den Studieninteressierten die Möglichkeit, eine fundierte Wahl zu treffen. Zudem unterstützt sie die Hochschulen bei der effizienten Bewerbung ihrer Studiengänge.

Schlagwörter: Suchmaschinenoptimierung, Webentwicklung, Hochschulmarketing, Social-Media-Analyse

Abstract

The subject of the work presented here is the development of an online platform that unites all study programmes, colleges, and universities in Europe on a single platform. This platform allows visitors to effortlessly search for and evaluate different study programmes and universities based on a range of criteria. Universities, in turn, can use the platform to strategically promote their offerings, ultimately increasing their visibility and attracting a wider range of students. To facilitate the platform's goals, special emphasis was placed on search engine optimisation, dynamic content integration and automated content updates. The use of Next.js for website development and Node.js for backend functionality was key to achieving the desired results. In addition, an algorithm was developed to evaluate and rank the universities' social media profiles. The algorithm used a comprehensive set of criteria to provide a thorough analysis of the universities' social media efforts. Its foundation is based on the algorithms discussed in the research papers "Market, Society, and Education – University Marketing" (Kloppenburger, 2012) and "Evaluation and implementation of university social media ratings for a corporate website" (Schuler, 2022). By combining website analysis, social media evaluation, and other resources, the platform provides a comprehensive resource for prospective students and academic institutions alike. The successful development and implementation of this online platform represents a significant step towards streamlining the education landscape in Europe. By centralising study programmes, colleges, and universities, and providing advanced evaluation tools, the platform empowers students to make informed choices while assisting academic institutions to effectively promote their courses.

Keywords: Search engine optimisation, web development, university marketing, social media analysis

Inhaltsverzeichnis

Kurzfassung	2
Abstract	3
Inhaltsverzeichnis	4
Abbildungsverzeichnis	7
Tabellenverzeichnis	7
Abkürzungsverzeichnis	8
1 Überblick	9
1.1 Vorwort	9
1.2 Problemstellung und vorhandene Plattformen	10
1.3 Projektumfang und Ziele	10
1.3.1 Funktionale und nicht-funktionale Anforderungen	11
1.3.2 Zielgruppen	12
2 Literaturübersicht	13
2.1 Relevante Konzepte und Technologien	15
2.1.1 Node.js	15
2.1.2 React	16
2.1.3 Next	16
2.1.4 Google Lighthouse	17
2.1.5 Prisma	17
2.1.6 Mantine	18
2.1.7 PostgreSQL	18
2.1.8 TypeScript	18
2.1.9 Next-translate	19
2.1.10 Playwright	19
3 Systementwurf	20
3.1 Übersicht	20
3.2 Internationalisierung	20
3.3 Wahl der Technologie	20
3.3.1 Analyse von Bezahlssystemen	21
3.4 Systemarchitektur	21
3.5 URL-Struktur	23
3.6 Design	24
3.7 Datenbankmodell	26
3.7.1 Datenbanktabellen	27

4	Implementierung	30
4.1	Next.js – Das Erstellen der Webseite	30
4.1.1	Seitenweite Suchfunktion	30
4.1.2	Accounts – Registrieren, Einloggen und Authentifizieren	31
4.1.3	Anzeigen und Artikel	32
4.1.4	Messung der meistbesuchten Seiten.....	36
4.1.5	Automatisch eine Sitemap generieren	37
4.1.6	Paginierungen mit SEO	37
4.1.7	Online-Marketing mit Lighthouse.....	38
4.1.8	Objektkarten für eine bessere Suchmaschinenoptimierung.....	39
4.1.9	Von Nutzern hochgeladene Bilder speichern und darstellen.....	39
4.2	Hauptcrawler.....	39
4.2.1	Länder und Bundesländer initialisieren.....	40
4.2.2	Hochschuldaten verarbeiten.....	40
4.2.3	Liste aller Städte einfügen	41
4.2.4	Studiengangsrichtungen sammeln und übersetzen.....	41
4.2.5	Hochschulwebseiten nach Social-Media-Links scrapen	41
4.2.6	Social-Media APIs	42
4.2.7	Social-Media: Bewertungsstrategie und Analyse	44
4.2.8	Social-Media Länderdaten berechnen	48
4.2.9	Social-Media-Daten in der Datenbank speichern	48
4.3	Lighthouse-Skript & Online-Marketing-Analyse	48
4.3.1	Bewertungs-Prozess	48
4.3.2	Bewertungsstrategie	49
4.4	Screenshot-Skript	50
4.5	Server.....	51
5	Analyse und Tests.....	53
5.1	Performance, SEO, Barrierefreiheit, Best Practices.....	53
6	Diskussion	56
6.1	Herausforderungen	56
6.1.1	Datenerfassung und -integration:	56
6.1.2	Web Scraping und Datenqualität:	56
6.1.3	SEO-Maßnahmen	56
6.1.4	Backend für Hochschulen.....	57
6.1.5	Social-Media Datenerhebung und -analyse	57
6.1.6	Online-Marketing Datenerhebung und -analyse	57
6.2	Beschränkungen	57
6.3	Verbesserung der Social-Media-Analyse	58
7	Fazit	59

7.1	Erfolge	59
7.2	Ausblick	59
7.2.1	Fertigstellung der Basisplattform	59
7.2.2	Zukunft	60
Anhang A: Datenbankmodell		61
Anhang B: Datei- und Ordnerverzeichnis.....		62
Anhang C: Payment Gateway Analyse.....		63
Anhang D: Screenshots		64
Glossar		68
Literatur- und Quellenverzeichnis.....		69
Stichwortverzeichnis		72

Abbildungsverzeichnis

Abbildung 1: Systemarchitektur – Serverstruktur.....	22
Abbildung 2: Design – Color Scheme.....	24
Abbildung 3: Beispiel des Designs einer Hochschulkarte.....	25
Abbildung 4: Navigationsleiste der Webseite	25
Abbildung 5: Vereinfachtes Datenbankmodell.....	26
Abbildung 6: Benutzeroberfläche zum Erstellen von Werbeanzeigen	33
Abbildung 7: Beispiel des Anzeigen Layouts.....	34
Abbildung 8: Benutzeroberfläche, um Artikel zu erstellen	35
Abbildung 9: Objektkarte eines Landes	64
Abbildung 10: Objektkarte eines Bundeslandes	64
Abbildung 11: Objektkarte einer Stadt.....	64
Abbildung 12: Objektkarte einer Hochschule	65
Abbildung 13: Objektkarte einer Studiengangsrichtung	65
Abbildung 14: Objektkarte einer Online-Marketing-Analyse	65
Abbildung 15: Online-Marketing-Übersicht im Profil der FH-Dortmund.....	66
Abbildung 16: Detailansicht „Performance“ der Online-Marketing-Analyse.....	67
Abbildung 17: Angabe der genauen Problemstellen in der Online-Marketing-Analyse.....	67

Tabellenverzeichnis

Tabelle 1: URL-Struktur der Webseite.....	23
Tabelle 2: Ergebnisse der Lighthouse-Tests im Navigationsmodus	53
Tabelle 3: Ordner und Dateien auf dem Datenträger	62
Tabelle 4: Payment Gateway Analyse.....	63

Abkürzungsverzeichnis

DAAD	Deutscher Akademischer Austauschdienst
HRK	Hochschulrektorenkonferenz
PWA	Progressive-Web-App
FCP	First contentful paint
TBT	Total blocking time
LCP	Largest contentful paint
CLS	Cumulative Layout Shift
I18N	Internationalization
ORM	Object-relational mapping
CMS	Content management system
UUID	Universally unique identifier
WYSIWYG	What you see is what you get

1 Überblick

1.1 Vorwort

In den letzten zwei Jahrzehnten hat sich die Hochschullandschaft deutlich in Richtung digitaler Plattformen und Online-Marketingstrategien entwickelt (Motta and Barbosa 2018, 128). Europäische Hochschulen haben erkannt, dass sie sich an diesen digitalen Wandel anpassen müssen, um ihre Studienangebote effektiv zu bewerben und Studieninteressierte aus der ganzen Welt anzuziehen. Daher investieren die Hochschulen zunehmend in Online-Marketing, um ihre Sichtbarkeit zu erhöhen, mit potenziellen Studierenden in Kontakt zu treten und ihre Markenidentität zu stärken (Merten and Knoll 2019, 156).

Bei der Suche nach Hochschulen und deren Studienangeboten nutzen Studierende heute in erster Linie das Internet als Informationsquelle. Um wettbewerbsfähig zu bleiben, müssen die Hochschulen daher unbedingt eine starke Online-Präsenz aufbauen und effektive digitale Marketingstrategien umsetzen (Motta and Barbosa 2018, 134–35; Motta and Barbosa 2018, 143–44).

Wenn man vorhandene Online-Plattformen zur Studiensuche beobachtet, fallen einige Problematiken auf.

- Die Suchmaschinenoptimierung (SEO) ist zu einem entscheidenden Aspekt des Online-Marketings geworden, da sie die Sichtbarkeit und das Ranking von Websites in den Suchmaschinenergebnissen bestimmt. Bestehende Plattformen erfüllen diese Anforderungen nur unzureichend, durch den Einsatz von SEO-Techniken könnten die Nutzerzahlen verbessert werden.
- Die Social-Media-Präsenz der Hochschulen wird nicht berücksichtigt. Die Hochschulen betreiben ihre Social-Media-Profile mit unterschiedlichem Erfolg, die zur Etablierung als Marke beitragen können.
- Die Bemühungen der Hochschulen im Online-Marketing werden ebenfalls nicht weiter analysiert. Die Webseite einer Hochschule ist eine wichtige Informationsquelle für Studierende und Studieninteressierte.

Dieses Projekt adressiert die wachsende Notwendigkeit für Universitäten, innovative Ansätze im Online-Marketing zu verfolgen, ihre Websites zu optimieren und Social-Media-Plattformen strategisch zu nutzen. Durch die Analyse bestehender digitaler Marketingpraktiken im Hochschulbereich wollen wir den Hochschulen praktische Empfehlungen und Einblicke zur Verbesserung ihrer Online-Marketing-Bemühungen geben. Darüber hinaus soll Studieninteressierten ein umfassender Überblick über Studienangebote in ganz Europa gegeben werden. Neben allen wichtigen Basisinformationen zur Hochschule selbst und zum jeweiligen Studiengang, sind

insbesondere die Online-Marketing-Analyse sowie die Social-Media-Analyse von Bedeutung.

1.2 Problemstellung und vorhandene Plattformen

Es gibt zwar zahlreiche Online-Plattformen zum Vergleich von Hochschulen und Studiengängen, diese sind jedoch häufig nicht suchmaschinenoptimiert. Die Social-Media-Profile und Online-Marketing-Bemühungen der Hochschulen werden nicht analysiert und bewertet.

Der Hochschulkompass, die offizielle Lösung der HRK (Hochschulrektorenkonferenz), dient nur selbst als Suchmaschine und ist daher nicht suchmaschinenoptimiert. Es werden keine weiteren Analysen zu Social-Media oder Online-Marketing durchgeführt, der dargestellte Content beschränkt sich auf die Steckbriefe der Hochschulen zu ihren Studiengängen.

Bessere Ergebnisse liefert die Seite „studieren.de“, bei der SEO-Techniken eingesetzt wurden, um Hochschulen und Studiengänge als Suchergebnisse liefern zu können. Problematisch ist hier, dass die Inhalte von der Community bzw. der Hochschule selbst erstellt werden müssen. Es wird keine Art von generiertem Content angeboten, Social-Media-Analyse und Online-Marketing-Analyse sind nicht vorhanden.

1.3 Projektumfang und Ziele

Die Webseite wird als zentrale Plattform dienen, die Studieninteressierten und Hochschulangehörigen einen unkomplizierten Zugang zu detaillierten Informationen über europäische Hochschulen und deren Bildungsangebote bietet. Sie wird effektive Suchmaschinenoptimierungstechniken einsetzen, um eine hohe Sichtbarkeit und Platzierung in den Suchmaschinenergebnissen zu gewährleisten und somit ein breiteres Publikum zu erfassen. Um dies zu realisieren, wird eine umfangreiche Datensammlung aus verschiedenen Quellen wie offiziellen Universitätswebsites, akademischen Datenbanken, Regierungsportalen, Wikipedia und sozialen Medien benötigt. Die Daten sollen automatisch vom Server gesammelt werden, um sie periodisch aktualisieren zu können. Auf der Website sollen verschiedene Aspekte der Universitäten vorgestellt werden, darunter ihre Standorte, die angebotenen Kurse, die Zulassungsbedingungen, die Analysen der Social-Media- und Online-Marketing-Präsenzen. Eine Liste von Screenshots der Hochschulwebsites in chronologischer Reihenfolge soll den Nutzern einen Eindruck von der Entwicklung der einzelnen Websites vermitteln.

Hochschulen sollten die Möglichkeit haben sich zu registrieren. Dabei soll nach der Hochschuldomain gefiltert werden, um sicherzustellen, dass nur Mitarbeiter einen Hochschul-Account auf der Webseite erstellen können. Dieser soll die Möglichkeit

bieten, Anzeigen zu buchen, die über ein Tool auf der Seite definiert werden können. Als längerfristige Alternative zu einer Anzeige sollen auch Blogposts angeboten werden, die vollständig von Hochschulmitarbeitern erstellt werden können, um z.B. einen neuen Studiengang zu bewerben.

1.3.1 Funktionale und nicht-funktionale Anforderungen

Die primären Zielgruppen sind Studieninteressierte und Hochschulen, bzw. die Marketingabteilungen der Hochschulen.

1.3.1.1 Funktional

- Das System soll eine mehrfach redundante SEO-Struktur besitzen.
- Das System soll um alle weiteren Länder Europas erweiterbar sein.
- Das System soll jede Hochschule, Universität und jede sonstige höhere Bildungseinrichtung in Deutschland umfassen.
- Das System soll wichtige Grunddaten zu den Institutionen und deren Studiengängen darstellen.
- Das System soll automatisch alle verlinkten Social-Media-Profile einer Hochschule finden können.
- Das System soll Daten zu den Social-Media-Profilen auf Facebook, Instagram, Youtube und Twitter sammeln.
- Das System soll die gesammelten Social-Media-Daten analysieren und bewerten.
- Das System soll die Social-Media-Bewertungen zu jeder Hochschule darstellen.
- Das System soll durchschnittliche Social-Media-Bewertungen nach Land erstellen.
- Das System soll eine Suchfunktion haben.
- Das System soll es Studierenden und Hochschulmitarbeitern erlauben Accounts zu erstellen.
- Das System soll Hochschulmitarbeitern die Möglichkeit geben bezahlte Anzeigen zu buchen.
- Das System soll Hochschulmitarbeitern das Buchen und Gestalten von Blogposts erlauben, und diese anzeigen.
- Das System soll es Studierenden erlauben ihren Studiengang in Textform zu bewerten.
- Das System soll periodisch Screenshots von Hochschulwebsites machen und darstellen.
- Das System soll zusätzlichen Content zu Hochschulen aus Wikipedia laden.
- Das System soll eine Online-Marketing-Analyse mit jeder Hochschulwebsite durchführen.

- Das System soll Online-Marketing-Daten jeder Hochschule detailliert anzeigen.

1.3.1.2 Nicht-Funktional

- Das System soll möglichst stark automatisiert sein.
- Das System soll alle Seiten möglichst performant anbieten.
- Das System soll das n*soria Design fortführen.
- Das System soll eine Benutzerfreundliche Oberfläche haben.
- Das System soll auf hohe Nutzerzahlen skalierbar sein.
- Das System soll responsive sein und auf allen üblichen Geräten funktionieren.
- Das System soll Wartung und Updates unaufwändig und schnell machen.
- Das System soll eine definierte Liste von Sprachen unterstützen.
- Das System soll problemlos um weitere Sprachen erweiterbar sein.

1.3.2 Zielgruppen

Die Plattform richtet sich insbesondere an zwei Zielgruppen: Personen, die ein Hochschulstudium anstreben und Hochschulen, die für ihre Studiengänge werben möchten. Zunächst richtet sich die Plattform an Personen, die aktiv nach Informationen und Möglichkeiten für ein Hochschulstudium suchen. Unabhängig davon, ob es sich um Studieninteressierte aus Europa oder dem Rest der Welt handelt. Die Plattform bietet umfassende Ressourcen, Orientierungshilfen und eine intuitive Benutzeroberfläche, um ihnen dabei zu helfen, fundierte Entscheidungen über ihren Bildungsweg zu treffen. Zum anderen sind die Hochschulen selbst ein wichtiger Teil der Zielgruppe, da sie die Möglichkeit haben, Anzeigen zu schalten und ihr Angebot einem breiten Spektrum von Studieninteressierten vorzustellen. Durch die Nutzung der Plattform können Hochschulen ihre gewünschte Zielgruppe effektiv erreichen und für ihre einzigartigen Programme und Campus-Erlebnisse werben.

2 Literaturübersicht

Die nachfolgende Literatur war bei der Planung und Implementierung dieses Projektes hilfreich.

1. Das Praxisprojekt „**Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal**“ stellt eine Ausgangsbasis dar, in der Social-Media-Analysen erstellt und untersucht werden. Die Techniken, Formeln und Vorgehensweisen wurden in diesem Projekt grundlegend modifiziert und verbessert, mehr dazu ist im Abschnitt 4.2.7 (Social-Media Bewertungsstrategie und Analyse) zu finden.¹
2. Für die Social-Media-Analyse war die Arbeit "**Market, Society, and Education**" von Bedeutung. Darin analysierte Kloppenburg die Aktivitäten deutscher Hochschulen in sozialen Netzwerken analysiert. Für die Analyse wurden Daten erhoben und mit Hilfe von Formeln und Algorithmen ausgewertet. Insbesondere die Idee der Bewertungsformeln, Reichweite mit Interaktion zu verknüpfen wurde in diesem Projekt aufgegriffen.²
3. "**Social Media as a Marketing Tool for European and North American Universities and Colleges**" ist eine Studie, die die Nutzung sozialer Medien durch europäische und nordamerikanische Hochschulen untersucht. Analysiert wurden Daten zur Anzahl der Postings und zur Anzahl der Follower in den verschiedenen sozialen Medien. Die Methodik ist quantitativ bei einer Stichprobe von den Top 100 platzierten Hochschulen des „Academic Ranking of World Universities“ (ARWU) 2017. Es wurde festgestellt, dass europäische und nordamerikanische Universitäten und Hochschulen in Marketingaktivitäten in sozialen Medien investieren. Hinsichtlich der Anzahl der Social-Networking-Sites, Content-Sharing- und Microblogging-Plattformen konnten keine signifikanten Unterschiede zwischen den Mittelwerten der beiden unabhängigen Stichproben festgestellt werden. Die am häufigsten genutzten sozialen Medien sind ex aequo Facebook und Twitter, gefolgt von Youtube, Instagram und LinkedIn.³
4. Die niederländische Studie „**Higher Education Marketing - A Study on the Impact of Social Media on Study Selection and University Choice**“ untersucht die Rolle und Bedeutung sozialer Medien bei der Studien- und Hochschulwahl von Studieninteressierten im Vergleich zu traditionellen Hochschulmarketingkanälen. Die

¹ Sebastian Schuler, „Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal,“ (Bericht Praxisprojekt, 2022).

² Michael Kloppenburg, „Market, Society, and Education - University Marketing,“ (Dissertation, Informatik und Mikrosystemtechnik, Hochschule Kaiserslautern, 2012).

³ Joana Motta und Maria Barbosa, „Social Media as a Marketing Tool for European and North American Universities and Colleges,“ *Journal of Intercultural Management* 10, Nr. 3 (2018), <https://doi.org/10.2478/joim-2018-0020>.

Methodik war quantitativ, die Stichprobe wurde aus einem Access-Panel mit über 120.000 Mitgliedern nach dem Wahrscheinlichkeitsverfahren der geschichteten Stichprobe ausgewählt. Die Zielgröße der Stichprobe wurde auf N=400 festgelegt, und die Größe der Zielschichten war proportional zur Verteilung der Schülerpopulation in den zwölf Provinzen. Nutzer wurden in drei Kategorien geteilt: Einfache Nutzer, soziale Nutzer und informelle Nutzer. Für jede Kategorie wurden unterschiedliche Merkmale identifiziert.⁴

5. Der Artikel **"How Canadian universities use social media to brand themselves"** untersucht die Social-Media-Marketingstrategien, die von kanadischen Universitäten als Instrument für das institutionelle Branding, die Rekrutierung und die Einbindung von einheimischen und internationalen Studierenden eingesetzt werden. Die Stichprobe umfasst die Gesamtpopulation der kanadischen Hochschulen (N = 106). Über einen Zeitraum von sechs Monaten wurden qualitative Daten von Facebook und Twitter erhoben. Zusätzlich wurden Daten zu den Immatrikulationszahlen erhoben, um einen Zusammenhang zwischen der Nutzung sozialer Medien und den Schwankungen der Einschreibungszahlen herzustellen. Die Ergebnisse zeigen, dass die Twitter-Plattform im Allgemeinen weitaus beliebter ist, um Konversationen zu führen, Facebook jedoch die bevorzugte Website für von Universitäten initiierte Postings bleibt, wobei sich die meisten dieser Postings auf Nachrichten und Veranstaltungen auf dem Campus/von Studierenden beziehen.⁵
6. Die Studie **"The university brand and social media: using data analytics to assess brand authenticity"** untersuchte die Authentizität gängiger Markenversprechen an drei kanadischen Universitäten anhand von Twitter- und Facebook-Posts zwischen Februar und April 2016 mit Hilfe eines cloudbasierten Text- und Netzwerkanalysetools. Ein Teilergebnis der Studie ist, dass sich die Social-Media-Strategien noch weitgehend auf Push-Benachrichtigungen beschränken, was darauf hindeutet, dass die Universitäten Gelegenheiten verpassen, ihre Marke zu stärken und negativen Botschaften entgegenzuwirken.⁶
7. Das Kapitel **"Internationalisierung, internationales Hochschulmarketing und Marktforschung: Wie Marketing Intelligence im Hochschulsektor zur Organisationsentwicklung und Positionierung von Hochschulen beiträgt"**

⁴ Efthymios Constantinides und Marc C. Z. Stagno, „Higher Education Marketing: A Study on the Impact of Social Media on Study Selection and University Choice,“ *International Journal of Technology and Educational Marketing* 2, Nr. 1 (2012), <https://doi.org/10.4018/ijtem.2012010104>.

⁵ Charles H. Bélanger, Suchita Bali, und Bernard Longden, „How Canadian universities use social media to brand themselves,“ *Tertiary Education and Management* 20, Nr. 1 (2014), <https://doi.org/10.1080/13583883.2013.852237> (2014).

⁶ James Pringle und Samantha Fritz, „The university brand and social media: using data analytics to assess brand authenticity,“ *Journal of Marketing for Higher Education* 29, Nr. 1 (2019), <https://doi.org/10.1080/08841241.2018.1486345>.

untersucht, warum Internationalisierung gerade für Hochschulen wichtig ist. Es diskutiert das internationale Hochschulmarketing, das für dieses Projekt von besonderer Bedeutung ist.⁷

8. Die Autoren von "**Marketing your university on social media: a content analysis of Facebook post types and formats**" untersuchten mittels einer Inhaltsanalyse die Themen und Formate von 5 932 Facebook-Posts führender US-amerikanischer Colleges und Universitäten. Die Ergebnisse zeigen, dass bestimmte inhaltliche Themen, wie z. B. Leichtathletik, das Engagement signifikant erhöhen, während andere eher zu einem geringeren Engagement führen. Darüber hinaus ist das Format, wie z. B. die Einbeziehung von nutzergenerierten Inhalten, ein weiterer Faktor, der zum Engagement beiträgt.⁸

2.1 Relevante Konzepte und Technologien

In diesem Abschnitt werden die Konzepte und Technologien, die in diesem Projekt verwendet wurden, näher erläutert.

2.1.1 Node.js

Node.js ist eine Laufzeitumgebung, die es Entwicklern ermöglicht, JavaScript-Code auf der Serverseite außerhalb der Grenzen eines Webbrowsers auszuführen. Es ermöglicht es Entwicklern, skalierbare, ereignisgesteuerte und effiziente Netzwerkanwendungen zu erstellen. Es bietet Zugriff auf verschiedene Module und Bibliotheken und eignet sich daher für die Entwicklung von Webservern, Kommandozeilentools, Echtzeitanwendungen und vielem mehr. Die leichtgewichtige und modulare Architektur sowie ein umfangreiches Ökosystem von Paketen über npm (Node Package Manager) machen es zu einer beliebten Wahl für die Entwicklung von serverseitigen Anwendungen und Microservices. Node.js ist für eine hohe Geschwindigkeit, Skalierbarkeit und Vielseitigkeit bekannt und ermöglicht es Entwicklern, eine breite Palette von JavaScript-Anwendungen sowohl auf der Client- als auch auf der Serverseite zu erstellen.⁹

⁷ Wolfgang Merten und Thorsten Knoll, *Handbuch Wissenschaftsmarketing: Konzepte, Instrumente, Praxisbeispiele* / Wolfgang Merten, Thorsten Knoll, 1. Aufl. 2019 (Wiesbaden: Springer Gabler, 2019). <https://doi.org/10.1007/978-3-658-25353-0>, 155.

⁸ Adam Peruta und Alison B. Shields, „Marketing your university on social media: a content analysis of Facebook post types and formats,” *Journal of Marketing for Higher Education*, 2018, <https://www.semanticscholar.org/paper/Marketing-your-university-on-social-media%3A-a-of-and-Peruta-Shields/a41bdbb031e51988bdb8f717f4e045ac7cff616d>.

⁹ OpenJS Foundation, *Node.js* (OpenJS Foundation), zuletzt geprüft am 04.06.2023, <https://nodejs.org>.

2.1.2 React

React ist eine JavaScript-Bibliothek zur Erstellung von Benutzeroberflächen, die einen deklarativen und komponentenbasierten Ansatz für die Webentwicklung bietet. Es wird von Meta entwickelt und verfügt über ein umfangreiches Ökosystem an unterstützenden Tools, Bibliotheken und von der Community bereitgestellten Paketen, die eine Vielzahl von Optionen für das Erstellen skalierbarer und funktionsreicher Webanwendungen bieten.

Mit React können Entwickler komplexe Benutzeroberflächen erstellen, indem sie kleinere, in sich geschlossene Komponenten zusammenstellen, die ihre eigene Logik und ihren eigenen Zustand gekapselt halten. Komponenten können in verschiedenen Teilen einer Anwendung mehrfach genutzt werden, was die Wiederverwendbarkeit und Wartbarkeit des Codes verbessert. React folgt einem einseitigen Datenfluss, bei dem Änderungen über den *State* verwaltet und als *Props* an untergeordnete Komponenten weitergegeben werden.

React unterstützt von sich aus kein Server-seitiges Rendering (SSR). Es kann jedoch in Verbindung mit anderen Bibliotheken und Frameworks wie Next.js (2.1.3) verwendet werden, um serverseitiges Rendering zu ermöglichen.¹⁰

2.1.3 Next

Next.js ist ein weit verbreitetes React-Framework, das serverseitiges Rendering (SSR) und statische Seitengenerierung (SSG) für die Entwicklung moderner Webanwendungen ermöglicht. Es bietet einen robusten Satz von Funktionen und Konventionen, die die Entwicklung skalierbarer React-Anwendungen erleichtern.

Mit Next.js können Entwickler dynamische und interaktive Benutzeroberflächen erstellen und dabei von den Vorteilen des serverseitigen Renderings profitieren. Das bedeutet, dass die erste Seite auf dem Server gerendert wird, was die Leistung verbessert und eine bessere Suchmaschinenoptimierung (SEO) ermöglicht. Darüber hinaus unterstützt Next.js das clientseitige Rendering für nachfolgende Seitenaktualisierungen und sorgt so für eine reibungslose Benutzererfahrung. Ein weiteres Hauptmerkmal ist die Unterstützung der statischen Seitengenerierung. Dies ermöglicht es Entwicklern, statische HTML-Seiten zum Zeitpunkt der Kompilierung zu rendern, die direkt über ein Content Delivery Network (CDN) bereitgestellt werden können. Dieser Ansatz ist ideal für inhaltsorientierte Websites oder Anwendungen mit Seiten, die keine Echtzeitdaten benötigen. Next.js fügt sich nahtlos in das React-Ökosystem ein und ermöglicht es Entwicklern, gängige Tools und Bibliotheken zu nutzen. Es bietet auch API Routing,

¹⁰ Meta Open Source, *React* (Meta), zuletzt geprüft am 02.06.2023, <https://react.dev/>.

wodurch die Erstellung von "serverless" Funktionen ermöglicht wird, die für die Handhabung von API-Anfragen oder serverseitiger Logik verwendet werden können.

Insgesamt hilft Next.js dabei hochleistungsfähige, SEO-freundliche und skalierbare Webanwendungen zu erstellen, indem es die Möglichkeiten des serverseitigen Renderings und der statischen Website-Generierung nutzt und gleichzeitig die Flexibilität und Benutzerfreundlichkeit von React beibehält.¹¹

2.1.4 Google Lighthouse

Google Lighthouse ist ein automatisiertes Open-Source-Tool, das die Leistung, Zugänglichkeit, SEO und Best Practices von Webseiten bewertet. Lighthouse führt eine Reihe von Audits für eine bestimmte Webseite durch. Zu diesen Audits gehören Metriken wie die Ladezeit der Seite, die erste inhaltsreiche Darstellung, die Zeit bis zur ersten Interaktion und mehr. Lighthouse prüft auch die Barrierefreiheit der Seite, z. B. den Farbkontrast, die Tastaturnavigation und die korrekte semantische Markierung. Es untersucht die Suchmaschinenoptimierung (SEO) der Seite durch Analyse von Meta-Tags, Überschriften und anderen relevanten Faktoren. Außerdem werden bewährte Verfahren wie die Verwendung von HTTPS, Caching und mobilfreundliches Design überprüft.¹²

2.1.5 Prisma

Prisma ist ein Open-Source-Datenbankwerkzeug und ORM, das den Datenbankzugriff und die Datenbankverwaltung für Entwickler vereinfacht. Es stellt eine Schnittstelle für die Interaktion mit Datenbanken bereit und bietet eine Reihe von Tools und Bibliotheken zur Vereinfachung von Datenbankfunktionen. Mit Prisma können Entwickler das Datenmodell und das Schema für ihre Anwendung mithilfe einer deklarativen Sprache definieren. Prisma generiert dann auf der Grundlage dieses Schemas einen Datenbank-Client-Code, der es den Entwicklern ermöglicht, CRUD-Operationen (Create, Read, Update, Delete) in der Datenbank mithilfe einer API durchzuführen. Dadurch entfällt die Notwendigkeit, Datenbankabfragen auf niedriger Ebene manuell zu schreiben, was die Komplexität und das Fehlerpotenzial reduziert. Prisma unterstützt verschiedene Datenbanken, darunter auch PostgreSQL.¹³

¹¹ Vercel, *Next.js* (Vercel), zuletzt geprüft am 02.06.2023, <https://nextjs.org/>.

¹² Google LLC, *Lighthouse* (Google LLC), zuletzt geprüft am 30.05.2023, <https://developer.chrome.com/docs/lighthouse/overview/>.

¹³ Prisma Data, Inc., *Prisma* (Prisma Data, Inc.), zuletzt geprüft am 04.06.2023, <https://www.prisma.io/>.

2.1.6 Mantine

Mantine ist eine React-Komponentenbibliothek, die eine Sammlung von anpassbaren UI-Komponenten bietet, um die Entwicklung moderner Webanwendungen zu vereinfachen. Mantine beinhaltet eine große Auswahl an vorgefertigten Komponenten, darunter Buttons, Formulare, Modals, Navigationselemente, Datenanzeigen und vieles mehr. Diese Komponenten wurden nach Best Practices erstellt, um Barrierefreiheit, Reaktionsfähigkeit und Browser-Kompatibilität zu gewährleisten. Die Bibliothek bietet eine saubere und intuitive API, um die Komponenten zu konfigurieren und an spezifische Bedürfnisse anzupassen. Mantine bietet auch ein Theming-System, das eine unkomplizierte Anpassung von Stilen und visuellen Aspekten ermöglicht und die Konsistenz mit dem Gesamtdesign einer Anwendung sicherstellt. Darüber hinaus bietet es integrierte Unterstützung für Dark Mode, Responsive Design und RTL-Textrichtung, um ein angenehmes Benutzererlebnis auf verschiedenen Geräten und bei unterschiedlichen Anforderungen zu gewährleisten.¹⁴

2.1.7 PostgreSQL

PostgreSQL ist ein leistungsstarkes relationales Datenbankmanagementsystem. Es ist bekannt für seine Robustheit, Skalierbarkeit und den umfangreichen Funktionsumfang, was es zu einer gängigen Wahl für verschiedene Anwendungen macht, von kleinen Projekten bis hin zu großen Unternehmenssystemen. Es unterstützt verschiedene Datentypen, darunter numerische, Text-, Datums- und Zeitangaben, JSON und weitere. Es bietet erweiterte Indizierungsfunktionen, die eine effiziente Suche und Abfrage von Daten ermöglichen. Insgesamt ist PostgreSQL ein vielseitiges und zuverlässiges relationales Datenbankmanagementsystem, das eine solide Grundlage für die Datenspeicherung und -verwaltung in einem breiten Spektrum von Anwendungen bietet, von simplen Webanwendungen bis hin zu komplexen Unternehmenssystemen.¹⁵

2.1.8 TypeScript

TypeScript ist eine Programmiersprache, die JavaScript um statische Typisierung und andere Funktionen erweitert (auch als Superset bezeichnet). Sie ermöglicht es Entwicklern, Typen für Variablen, Funktionsparameter und Rückgabewerte zu definieren, wodurch Fehler während der Entwicklung erkannt und die Codequalität verbessert wird. TypeScript unterstützt Schnittstellen, Klassen, Module und Generika und macht den Code strukturierter und skalierbarer. Es bietet eine verbesserte Tooling- und IDE-Unterstützung, die eine bessere Code-Vervollständigung und Refactoring

¹⁴ Vitaly Rtishchev, *Mantine* (mantinedev), zuletzt geprüft am 04.06.2023, <https://mantine.dev/>.

¹⁵ The PostgreSQL Global Development Group, *PostgreSQL* (The PostgreSQL Global Development Group), zuletzt geprüft am 04.06.2023, <https://www.postgresql.org/>.

ermöglicht. TypeScript fördert modulare und objektorientierte Programmierpraktiken und verbessert die Skalierbarkeit und Wartbarkeit von Codebasen. Die Typ-Annotationen dienen als Dokumentation und erleichtern die Zusammenarbeit und die Erstellung genauer Dokumentationen.¹⁶

2.1.9 Next-translate

Die Node.js-Bibliothek "next-translate" erweitert die Internationalisierungsfähigkeit (i18n) von Next.js-Anwendungen und bietet Unterstützung für mehrsprachige Übersetzungen. Sie vereinfacht das Hinzufügen von Übersetzungen zu einem Next.js-Projekt durch Funktionen wie automatische Spracherkennung, dynamische Übersetzung von Inhalten und nahtlose Integration mit Next.js-Routing bietet. Es ermöglicht Entwicklern, die Sprachlokalisierung effizient zu verwalten und den Benutzern eine lokalisierte Erfahrung zu bieten. Übersetzungen werden zunächst nach Sprache und dann nach Seite im Ordner „*locales*“ gruppiert.¹⁷

2.1.10 Playwright

Playwright ist ein Web-Automatisierungstool, das Entwicklern eine Reihe leistungsstarker APIs zur Automatisierung von Browser-Interaktionen bietet. Es ermöglicht die Automatisierung von Aufgaben wie Website-Tests, Web-Scraping und die Erstellung browserbasierter Workflows. Playwright unterstützt mehrere Webbrowser, darunter Chromium, Firefox und WebKit, und bietet browserübergreifende Kompatibilität. Entwicklern können Automatisierungsskripte in verschiedenen Programmiersprachen wie JavaScript, Python und C# schreiben. Playwright wurde entwickelt, um den Prozess der Automatisierung von Browseraktionen zu vereinfachen und so Web-Automatisierungsaufgaben effizienter und zuverlässiger zu machen.¹⁸

¹⁶ Microsoft, *TypeScript* (Microsoft), zuletzt geprüft am 04.06.2023, <https://www.typescriptlang.org/>.

¹⁷ Aral Roca Gomez, *next-translate*, zuletzt geprüft am 07.06.2023, <https://github.com/aralroca/next-translate>.

¹⁸ Microsoft, *Playwright* (Microsoft), zuletzt geprüft am 15.06.2023, <https://playwright.dev/>.

3 Systementwurf

3.1 Übersicht

Die Webseite soll Daten zu Hochschulen und deren Studiengängen aus einer Datenbank darstellen. Diese Daten sollen aus verschiedenen Quellen mit Hilfe von Skripten gesammelt werden. Die Skripte müssen automatisch aufgerufen werden können, um den Datensatz periodisch zu aktualisieren.

3.2 Internationalisierung

Um alle relevanten Sprachen abzudecken, werden die wichtigsten in Europa vertretenen Sprachen berücksichtigt. Dazu gehören Deutsch, Englisch, Französisch, Spanisch, Italienisch, Portugiesisch, Russisch und Türkisch. Zur Zielgruppe gehören Studieninteressierte aus der ganzen Welt, also auch internationale Studierende aus Ländern außerhalb Europas. In Deutschland kommen die meisten internationalen Studierenden aus China und Indien¹⁹, daher müssen auch Mandarin-Chinesisch und Hindi auch unterstützt werden. Um die Top 10 der meistgesprochenen Sprachen der Welt²⁰ zu vervollständigen, werden auch Arabisch und Urdu angeboten.

Die Internationalisierung wurde mit der Bibliothek „next-translate“ realisiert, mehr dazu in Abschnitt 2.1.9.

3.3 Wahl der Technologie

Dieser Abschnitt beschreibt die Technologien und warum sie für dieses Projekt gewählt wurden.

- - Gründe für die Wahl von React waren unter anderem die vorhandenen Kenntnisse der Projektdurchführenden. Aber auch die möglichen Erweiterungen, Dokumentationen und Hilfestellungen, die React als das am häufigsten verwendete Frontend-Framework mit sich bringt.
- Da es sich um eine suchmaschinenoptimierte Webseite handelt, benötigen wir ein Framework wie **Next.js**, um React-Code statisch anzubieten. Eine mögliche Alternative wäre Remix. Zum Zeitpunkt des Projektstarts hatte Next jedoch einen größeren und ausgereifteren Funktionsumfang.

¹⁹ Erudera, „Germany International Student Statistics,” zuletzt geprüft am 06.06.2023, <https://erudera.com/statistics/germany/germany-international-student-statistics/>.

²⁰ Ethnologue, „What are the top 200 most spoken languages?,” zuletzt geprüft am 07.06.2023, <https://www.ethnologue.com/insights/ethnologue200/>.

- **Prisma** reduziert den Entwicklungsaufwand erheblich, was insbesondere in diesem zeitkritischen Projekt relevant war. Zu beachten ist, dass performancekritische Anfragen, beispielsweise das Messen der Popularität bestimmter Unterseiten, als rohe SQL-Anfragen geschrieben werden sollten.
- **PostgreSQL** ist kostenlos, open-source und ein stabiles und ausgereiftes Datenbanksystem, wodurch es eine sichere Wahl für dieses Projekt war.
- Der Hauptgrund für **Node.js** war die volle Kompatibilität mit TypeScript. Dadurch können die gleichen Typen in Crawler und Webseite genutzt werden, um Entwicklungszeit zu sparen und Fehler zu vermeiden.

Der gewählte Technologie-Stack ist sehr anpassbar und dynamisch im Vergleich zu einem bestehenden CMS. Er bringt viel Arbeit mit sich, erfüllt aber alle spezifischen Anforderungen, die das Projekt und dessen Ziele mit sich bringen.

3.3.1 Analyse von Bezahlssystemen

Ein Bezahlssystem-Anbieter ist für die Abwicklung von Kreditkarten- oder Direktzahlungen im E-Commerce verantwortlich. Er erleichtert die Zahlungstransaktion durch die Übertragung von Informationen zwischen einem Zahlungsportal (im Falle dieses Projektes, dieses Webportal) und der gewählten Zahlungsoption (z. B. PayPal oder eine Bank). Zu den Anforderungen gehört, dass er keine monatlichen Kosten, eine möglichst geringe Transaktionsgebühr und eine unkomplizierte Einbindung in die bestehende Applikation der Webseite hat. Verschiedene Anbieter wurden analysiert, um die optimale Wahl für das Projekt zu finden. Das Ergebnis der Analyse ist im Anhang C: Payment Gateway Analyse zu finden. Stripe und Square sind die einfachsten und gleichzeitig günstigsten Lösungen für die gegebenen Parameter.

3.4 Systemarchitektur

Next.js bildet das Frontend der Webseite und bietet serverseitiges Rendering und andere Funktionen, die die Performance, das Benutzererlebnis und die Suchmaschinenoptimierung der Anwendung verbessern. Next bietet fertige Lösungen für einige Teilziele dieses Projekts. In Verbindung mit einer i18n-Bibliothek kann die Internationalisierung in beliebig vielen Sprachen realisiert werden. Alle für die Suchmaschinenoptimierung relevanten Eigenschaften und Anforderungen (z. B. Metadaten) werden standardmäßig unterstützt.

Die Next-App verwendet die Bibliothek „Prisma“, um die Verbindung und den Austausch mit der Datenbank zu verwalten. Prisma bietet eine verbesserte Entwicklungserfahrung mit Typsicherheit und automatischer Vervollständigung. Dies spart Entwicklungszeit und reduziert das Fehlerrisiko.

Die Anwendungsdaten werden in der PostgreSQL-Datenbank gespeichert. Diese ist leistungsstark und bietet die Möglichkeit, strukturierte Daten zu speichern. Zusätzlich unterstützt sie komplexe Abfragen und Transaktionen.

Der Server führt in regelmäßigen Abständen mehrere Skripte aus. Diese sind für das Sammeln der erforderlichen Daten aus verschiedenen Quellen, wie APIs oder externen Systemen, verantwortlich. Die gesammelten Daten werden dann verwendet, um die PostgreSQL-Datenbank mit aktuellen Informationen zu aktualisieren. Diese Skripte werden so geplant, dass sie in bestimmten Intervallen ausgeführt werden, um sicherzustellen, dass die Daten in der Datenbank auf dem neuesten Stand sind.

Die Skripte sind in drei Teilprojekte geteilt:

- Der **Hauptcrawler** agiert als primäre Datensammler und Verarbeiter. Er bereitet gegebene Daten auf, sammelt Daten von sozialen Medien und analysiert gesammelte Daten. Dabei sind die Funktionen einzeln aufrufbar und logisch voneinander getrennt.
- Das **Screenshot-Skript** erzeugt Bildschirmfotos der Hochschulwebseiten und erstellt jeweils einen Eintrag in der Datenbank.
- Das **Lighthouse-Skript** nutzt das Node-Modul Lighthouse, um Webseiten zu testen und analysieren. Ergebnisse werden in Dateien gespeichert mit jeweils einer Referenz in der Datenbank.

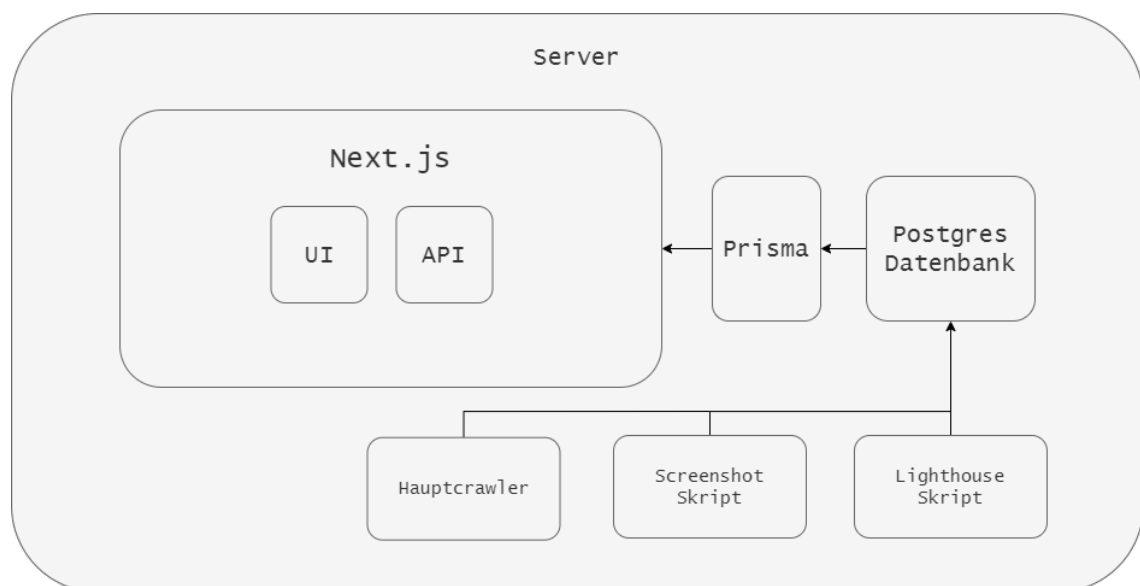


Abbildung 1: Systemarchitektur – Serverstruktur

Zusammengefasst nutzt diese App-Architektur Next und React für das Frontend, PostgreSQL für die Datenspeicherung und serverseitige Node.js-Skripte für die Datensammlung und -aktualisierung (Abbildung 1). Diese Kombination ermöglicht die

Erstellung einer dynamischen Anwendung, die vollautomatisch möglichst aktuelle Daten bereitstellt.

3.5 URL-Struktur

Die drei primären URL-Pfade */institution*, */location* und */category* bilden eine mehrfach redundante URL-Struktur (siehe Tabelle 1). Diese hat die Zweck den PageRank der Seite zu verbessern.

Tabelle 1: URL-Struktur der Webseite

URL-Pfad	Kommentar
... / institution / [Land] / [Hochschule] / ...	Jedes Hochschulprofil hat folgende Unterseiten: Profil, Social-Media, Online-Marketing, Studiengänge, Screenshots, Reviews, Artikel
... / location / [Land] / [Bundesland] / [Stadt]	Auf jeder Seite einer bestimmten Stadt werden Hochschulen verlinkt die einen Standort in der Stadt haben.
... / category / [Studiengangsrichtung]	Auf jeder Seite einer bestimmten Studiengangsrichtung werden passende Studiengänge verlinkt.
... / news / [Artikel]	Eine Liste der Pressemitteilungen und gesponserten Artikel aller Hochschulen.
... / analysis / social-media	Analysen die sich auf die Social-Media-Statistiken der Hochschulen beziehen. Das Social-Media-Ranking ist eine der Unterseiten.
... / analysis / online-marketing	Analysen die sich auf die Online-Marketing-Analysen beziehen.
... / login ... / register	Hier können Nutzer einen Account registrieren oder sich mit einem vorhandenen einloggen.
... / account ... / account / create-ad ... / account / manage-ads ... / account / history ... / account / support ... / account / settings	Ansicht die man nur als eingeloggtter Nutzer sehen kann. Hochschulen können hier Anzeigen und Artikel buchen und verwalten. Zusätzlich kann ein Buchungsverlauf eingesehen oder der Support kontaktiert werden.

... / search / ...	Seitenweite Suchfunktion die den Nutzer die Datenbank nach Hochschulen, Studiengängen oder Orten durchsuchen lässt.
... / about / imprint ... / about / privacy	Impressum und Datenschutzerklärung

Die URL-Struktur aus Tabelle 1 gibt alle möglichen Pfade an, die ein Nutzer gehen kann. Bei jedem Pfad wird von der Basis-URL ausgegangen, im Entwicklungszeitraum war dieser <https://analyse.hs-kl.de>.

3.6 Design

Das Design entspricht dem Styleguide des Forschungsprojektes n*soria an der Hochschule Kaiserslautern. Das in Abbildung 2 dargestellte Farbschema wurde als Grundlage genommen, um ein für den speziellen Anwendungsfall geeignetes Farbschema zu erstellen.

	Accents	Backgrounds	Body Text
Primary Colors	 #FF9100 RGB(255, 145, 0)	 #FFFFFF RGB(255, 255, 255)	 #000000, 87% Opacity RGBA(0, 0, 0, 0.87)
Secondary Colors	 #FF6D00 RGB(255, 109, 0)	 #D9D9D9 RGB(217, 217, 217)	 #F7F7F7 RGB(127, 127, 127)
Tertiary Colors	 #FFE9CC RGB(255, 233, 234)		

Abbildung 2: Design – Color Scheme

Die Schriftart „Roboto“ von Google wurde ebenfalls aus dem n*soria Styleguide in dieses Projekt übernommen. Die Schriftgrößen werden in REM angegeben, wodurch alle Schriftgrößen von der Standardgröße des jeweiligen Gerätes abhängig sind. Auch Abstände und Breakpoints werden ebenfalls in REM angegeben. Es gibt fünf Breakpoints, um die Webseite unabhängig vom Gerät des Nutzers darstellen zu können.

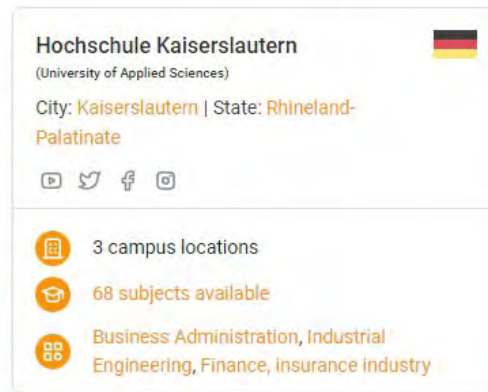


Abbildung 3: Beispiel des Designs einer Hochschulkarte

Ein Konzept, das immer wieder aufgegriffen wurde, sind die Karten zu den einzelnen Elementen. So hat jede Hochschule in der Liste aller Hochschulen eine eigene Karte. Auf dieser werden Details zum jeweiligen Element dargestellt und verlinkt. Im Beispiel der Hochschulkarte (Abbildung 3) sind die Stadt, das Bundesland, die sozialen Medien, die Liste aller Studiengänge und die drei beliebtesten Studiengänge dargestellt und verlinkt. Weitere Objekte, die dieses Konzept nutzen, sind Studiengänge, Städte, Bundesländer, Länder und Studiengangsrichtungen. Die Listen dieser Karten sind durchsuchbar, filterbar und durch eine Paginierung in Seiten unterteilt. Diese erfüllt alle SEO-spezifischen Anforderungen, mehr dazu im Abschnitt (4.1.6 Paginierung).



Abbildung 4: Navigationsleiste der Webseite

Die Navigationsleiste bietet primär Verlinkungen zu den drei wichtigen URL-Pfaden (Siehe Abschnitt 3.5 URL-Struktur). Zusätzlich werden die Kategorien „Analyse“ und „News“ verlinkt. „Analyse“ ist dabei eine Sammlung aller Social-Media- und Online-Marketing-Analysen und „News“ ein Archiv aller Pressemeldungen und Artikel von Hochschulen. Die aktuell ausgewählte Seite ist dabei immer farblich markiert. Auch Funktionalität zum Sprachwechsel oder zur Suche auf der Seite ist hier gegeben.

3.7 Datenbankmodell

Die Datenbankstruktur besteht aus mehreren miteinander verbundenen Tabellen zur Organisation und Speicherung von Informationen und Beziehungen zwischen Daten. In Abbildung 5 ist ein vereinfachtes Datenbankmodell zu sehen, das sich auf die Namen aller Tabellen und deren Beziehungen zueinander beschränkt.

Eine detaillierte Version des Datenbankmodells befindet sich im Anhang A: Datenbankmodell.

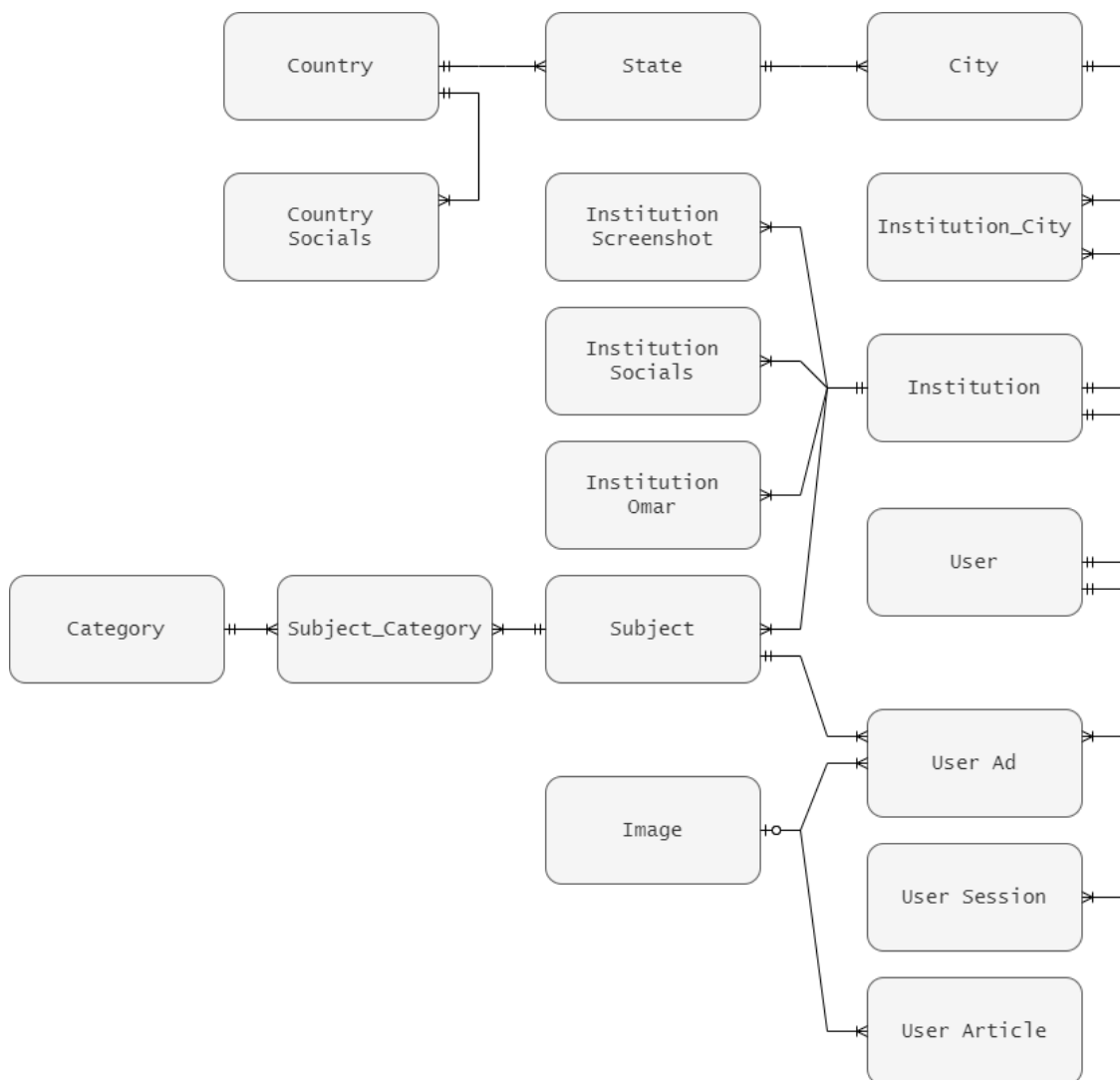


Abbildung 5: Vereinfachtes Datenbankmodell

3.7.1 Datenbanktabellen

3.7.1.1 COUNTRY

Die Tabelle „COUNTRY“ enthält alle europäischen Länder. Der Primärschlüssel ist eine dreistellige ID. Die Spalte „URL“ enthält einen eindeutigen String, der das jeweilige Land in der URL identifiziert. Dieses Prinzip wird in vielen anderen Tabellen verwendet. Da Ländernamen übersetzt werden können, enthält die Spalte „TRANSLATIONS“ alle verfügbaren Übersetzungen. Die Funktionsweise der Spalte „POPULAR“ wird im Abschnitt 4.1.4 (Messung der meistbesuchten Seiten) näher erläutert.

3.7.1.2 STATE

Die Tabelle „STATE“ enthält die Bundesländer aller in der Datenbank vertretenen Länder. Das Feld „ID“ ist der Primärschlüssel und „COUNTRY_ID“ der Fremdschlüssel zur Tabelle „Country“. Der Name eines Bundeslandes wird in den Feldern „NAME_EN“ und „NAME_NATIVE“ nur in Englisch und in der Landessprache des jeweiligen Landes gespeichert.

3.7.1.3 CITY

Die Tabelle „CITY“ enthält alle relevanten Städte in Europa. Über den Fremdschlüssel „STATE_ID“ wird die Stadt einem bestimmten Bundesland zugeordnet. Die Städtenamen werden grundsätzlich nicht übersetzt und liegen somit nur in der offiziellen Form des Landes vor. Zusätzlich sind im Feld „POSTCODES“ alle Postleitzahlen der Stadt als Liste gespeichert, um dem Benutzer die Suche zu erleichtern.

3.7.1.4 INSTITUTION

Die Tabelle „Institution“ enthält Universitäten und Hochschulen aus ganz Europa. Der Primärschlüssel ist eine zufällig generierte UUID. Die Spalten „WEBSITE“, „NAME“ und „URL“ sind alle eindeutige Zeichenketten. Es gibt Fremdschlüssel in den Feldern „MAIN_LOCATION“ und „OMAR_ID“. Sie geben die Stadt an, in der sich der Hauptstandort befindet, sowie den entsprechenden Eintrag in der Online-Marketing-Tabelle an. Die Tabelle „INSTITUTION_CITY“ verknüpft Städte mit Hochschulen, um sicherzustellen, dass eine Hochschule mehrere Standorte haben kann.

3.7.1.5 SUBJECT

Die Tabelle „SUBJECT“ enthält alle erfassten Studiengänge. Sie werden über den Fremdschlüssel „INSTITUTION_ID“ einer Hochschule und über „CITY_ID“ einem bestimmten Standort zugeordnet. Verschiedene relevante Daten wie die Semesterzahl, den Abschluss, den Link zur Webseite oder Zulassungsbedingungen werden ebenfalls in dieser Tabelle gespeichert. Der Primärschlüssel ist ebenfalls eine generierte UUID.

3.7.1.6 CATEGORY

Die Tabelle „CATEGORY“ enthält alle verfügbaren Studiengangsrichtungen. Die Felder „URL“ und „NAME_NATIVE“ müssen dabei eindeutig sein. Der Name wird in der Originalsprache und der Übersetzung ins Englische gespeichert. Dabei gibt „NATIVE_LANG“ an um welche Sprache es sich dabei handelt. Die Tabelle „SUBJECT_CATEGORY“ verknüpft Studiengänge mit Studiengangsrichtungen in einer n:m-Beziehung.

3.7.1.7 INSTITUTION_OMAR

Die Tabelle „INSTITUTION_OMAR“ verfolgt, für welche Hochschulen eine Online-Marketing-Analyse durchgeführt wurde, und unter welchem Dateinamen diese gespeichert ist. Zusätzlich soll sie zu einem späteren Zeitpunkt um das Datum erweitert werden, um mehrere Analysen zu speichern, die in bestimmten Abständen durchgeführt wurden.

3.7.1.8 INSTITUTION_SCREENSHOT

Die Tabelle „INSTITUTION_SCREENSHOT“ enthält alle Screenshots der Hochschulwebsites. Mit Hilfe des Fremdschlüssels „INSTITUTION_ID“ werden sie der jeweiligen Hochschule zugeordnet. Das Feld „FILENAME“ ist eindeutig und gibt den Dateinamen an, unter dem das Bild auf dem Server gespeichert ist. Einige Metadaten wie der Zeitpunkt und die URL sind ebenfalls vorhanden.

3.7.1.9 INSTITUTION_SOCIALS

Die Tabelle „INSTITUTION_SOCIALS“ enthält den Social-Media-Datensatz zu jeder Hochschule, zugeordnet durch den Fremdschlüssel „INSTITUTION_ID“. Die Links zu den Social-Media-Profilen werden als Textobjekte gespeichert, und die einzelnen Bewertungen jedes Profils einer Hochschule als Zahl. Alle weiteren Details, Analysen und Bewertungen sind in Form eines JSON Objektes aufgeteilt nach Social-Media-Plattform. Weitere Informationen zu den gespeicherten JSON-Daten im Abschnitt 4.2.7 Social-Media Bewertungsstrategie und Analyse.

3.7.1.10 COUNTRY_SOCIALS

Die Tabelle „COUNTRY_SOCIALS“ enthält in jeder Zeile die Social-Media-Statistiken eines bestimmten Landes. Die durchschnittlichen Gesamtbewertungen aller Profile im jeweiligen Land und die Anzahl der Profile werden als Zahlen gespeichert. Alle weiteren Details sind wie in der vorherigen Tabelle „INSTITUTION_SOCIALS“ in Form eines JSON Objektes vorhanden, mehr dazu im Abschnitt 4.2.8 Social-Media Länderdaten berechnen.

3.7.1.11 USER

Die Tabelle „USER“ enthält registrierte Nutzer. Dazu gehören Marketing-Accounts von Hochschulen und Review-Accounts von Studierenden. Der Primärschlüssel ist eine generierte ID. Jeder Account ist zusätzlich durch den Fremdschlüssel „INSTITUTION_ID“ mit einer Hochschule verbunden. Das Passwort ist mittels einer Hashfunktion verschlüsselt.

3.7.1.12 USER_AD

Die Tabelle „USER_AD“ enthält alle Anzeigen, die von Hochschulen gebucht wurden. Sie werden mit einer generierten UUID als Primärschlüssel identifiziert. Mehrere Fremdschlüssel existieren um den zugehörigen Nutzer, den zugehörigen Studiengang (optional) und das zugehörige Bild (optional) zu verbinden. Alle weiteren Daten, um die Anzeige darzustellen, sind auch in der Tabelle enthalten.

3.7.1.13 USER_ARTICLE

Die Tabelle „USER_ARTICLE“ enthält alle Artikel und Pressemitteilungen, die von Hochschulen erstellt wurden. Der Primärschlüssel ist eine generierte UUID und Fremdschlüssel existieren für den zugehörigen Benutzer und das Thumbnail des Artikels. Der eigentliche Inhalt wird als formatierter JSON-Text gespeichert, siehe Abschnitt 4.1.3.4 (Artikel Erstellen).

3.7.1.14 USER_IMAGE

Die Tabelle „USER_IMAGE“ enthält alle von Benutzern hochgeladenen Bilder. Der Primärschlüssel ist eine generierte UUID, die gleichzeitig als Dateiname des jeweiligen Bildes dient. Dies hat den Vorteil, dass Bilder, die nicht in Anzeigen oder Artikeln verwendet werden, leicht gefunden und gelöscht werden können.

3.7.1.15 USER_SESSION

Die Tabelle „USER_SESSION“ listet alle vorhandenen Login-Tokens der Benutzer und deren Ablaufdatum auf. Damit ein Benutzer über einen bestimmten Zeitraum und über mehrere Browser-Sitzungen hinweg eingeloggt bleiben kann, wird das Token als Cookie lokal gespeichert und kann dann mit dem Token in der Tabelle verglichen werden. Der Primärschlüssel ist das Token und der Fremdschlüssel das Feld „USER_ID“.

4 Implementierung

Die Implementierung des Systems ist in mehrere Abschnitte gegliedert, von denen jedes ein eigenes Projekt ist, mit dem Ziel die Webseite wie erwartet darzustellen.

4.1 Next.js – Das Erstellen der Webseite

Wie in Abschnitt 3.3 (Wahl der Technologie) begründet und in Abschnitt 2.1.3 (Next) näher erläutert, wird Next.js zur Erstellung der Webseite verwendet. Die verwendete Version von Next.js ist 13.3, sollte aber auf dem neuesten Stand gehalten werden, um wichtige Sicherheits- und Performance-Updates zu erhalten. Next bietet mit der neuen Version 13 auch eine neue, verbesserte Dateistruktur für Projekte, den „App Router“. Da sich dieser zum Zeitpunkt der Projekterstellung noch in der Testphase befindet, verwendet dieses Projekt die klassische Methode, den „Pages Folder“. Der Wechsel zwischen diesen beiden Methoden ist ein zeitaufwändiger Prozess, bietet aber viele Vorteile, darunter eine deutlich verbesserte Performance. Daher sollte die Entwicklung von Next in der Zukunft beobachtet werden, um den richtigen Zeitpunkt für den Wechsel zu finden.

4.1.1 Seitenweite Suchfunktion

Die Suchfunktion sollte es dem Nutzer ermöglichen, nach Ländern, Bundesländern, Städten, Hochschulen und Studiengängen zu suchen. Die Suche sollte angemessen schnell sein und Filter zur Verbesserung der Suchergebnisse anbieten.

4.1.1.1 Next API-Route

Zu diesem Zweck wurde eine API-Route erstellt die im API-Ordner unter *search/global.ts* zu finden ist. Durch die Implementierung als API-Route kann die Suchfunktion auf verschiedene Arten aufgerufen werden. Wichtige Parameter sind dabei der Suchtext, die Sprache und die gewählten Filter. Über die *fetch*-Funktion wird die API-Route an zwei wichtigen Stellen verwendet. Die erste ist die Funktion, die durch einen Klick auf den „Suchen“-Button gestartet wird, sie befindet sich in der Komponente der Seite „Search“. Die Funktion *getServerSideProps*, wird ausgeführt, wenn Next die Seite auf das Rendern vorbereitet. Hier wird geprüft, ob die für eine Suche benötigten URL-Parameter vorhanden sind. Ist das der Fall, wird das Suchergebnis sofort nach dem Aufruf der URL angezeigt. Dies hat den Vorteil, dass Suchergebnisse von Benutzern verlinkt werden können.

4.1.1.2 Datenbank durchsuchen

Die eigentliche Suche erfolgt in der Funktion *prismaGlobalSearch*. Unter dem Typ *SearchResult* werden die Details der einzelnen Suchergebnisse zusammengefasst. Die

Suche in der Datenbank erfolgt als „rohe Prisma-Anfrage“ mit der Funktion *prisma.\$queryRaw*. Da Prisma die ILIKE-Funktionalität von PostgreSQL noch nicht unterstützt ist es nötig die SQL-Abfrage von Hand auszuschreiben. Dies geschieht für jede durchsuchbare Tabelle einzeln.

4.1.1.3 Diskussion

Die gewählte Suchstrategie liefert akzeptable bis gute Ergebnisse bei geringem Arbeitsaufwand. Eine mögliche Verbesserung, die besonders bei den aktuellen Trends in der Technologiebranche relevant ist, wäre ein LLM (Large Language Model). Die wahrscheinlich einfachste Umsetzung der Suchfunktion mit einem LLM wäre GPT-4. Die Aufgabe des Models würde darin bestehen, die Suchanfrage des Nutzers nach Schlüsselwörtern zu durchsuchen und die Entscheidung zu treffen, welche Datenbanktabellen danach durchsucht werden. Das Resultat wäre eine Suchfunktion, die bei Anfragen in natürlicher Sprache gute Ergebnisse liefern kann.

4.1.2 Accounts – Registrieren, Einloggen und Authentifizieren

In der aktuellen Version können sich Mitarbeiter von Hochschulen mit der E-Mail-Adresse der Hochschule registrieren. Die gleiche Funktionalität ist auch für Studierende geplant, um Rezensionen schreiben zu können. Beim Anlegen eines Accounts muss zunächst die Hochschule aus einer Liste ausgewählt werden. Für eine Hochschule kann es beliebig viele Studierendenaccounts geben, aber nur einen Marketing-Account. Mitarbeiter von Hochschulen können im Marketing-Account Anzeigen und Artikel buchen (siehe Abschnitt 4.1.3).

4.1.2.1 Registrierung

Auf der Seite */register* können sich neue Benutzer registrieren. Zuerst muss eine Hochschule aus der Liste ausgewählt werden. Im nächsten Schritt wird eine E-Mail-Adresse benötigt, diese muss der Domain der gewählten Hochschule entsprechen. Das Versenden einer Bestätigungsmail wird zu einem späteren Zeitpunkt hinzugefügt. Nachdem das Registrierungsformular ausgefüllt wurde, wird es an die Next.js API der Webseite gesendet. Die API-Schnittstelle */api/account/register* ist dafür zuständig. Sie überprüft die eingegebenen Daten und vergleicht sie mit der Datenbank, die E-Mail-Adresse muss einzigartig sein und die Hochschule darf keine weiteren Marketing-Accounts besitzen. Um die Passwörter sicher zu speichern, wird die kryptologische Hashfunktion *bcrypt* verwendet. Diese erzeugt eine verschlüsselte Zeichenkette für das Passwort, die sicher in der Datenbank gespeichert werden kann. Nach der Freischaltung durch die Bestätigungsemail ist der Account sofort nutzbar, der Nutzer kann sich sofort einloggen.

4.1.2.2 Anmeldung

Auf der Seite */login* können sich Benutzer mit ihren Account-Daten anmelden. Die Verifizierung der Daten erfolgt wie zuvor die Registrierung über die Next.js API-Schnittstelle, in diesem Fall */api/account/login*. Die kryptologische Hashfunktion *bcrypt* bietet eine Funktion, um das eingegebene Passwort mit dem in der Datenbank gespeicherten Hash-Wert zu vergleichen. Sollte die E-Mail oder das Passwort nicht mit einem Account übereinstimmen, wird eine Fehlermeldung angezeigt. Stimmen die Daten überein, wird der Benutzer zum Dashboard der Anwendung weitergeleitet.

4.1.2.3 Sessions

Damit Benutzer angemeldet bleiben und weiter mit der Webseite interagieren können, werden Sessions benötigt. Jede Session hat standardmäßig eine Lebensdauer von drei Tagen. Bei jeder Anmeldung wird ein neuer Eintrag in einer Datenbanktabelle (siehe 3.7.1.15 *USER_SESSION*) erzeugt. Dabei wird ein einzigartiger Token mit Hilfe des *Crypto*-Moduls von Next.js generiert. Dieser Token wird mit der entsprechenden Nutzer-ID verbunden und bekommt eine Lebenszeit zugewiesen. Gleichzeitig wird der Token auch an den Client gesendet und dort als Cookie im Browser des Nutzers abgespeichert. Bei jedem Besuch einer Seite, die einen Login voraussetzt, wird vom Server geprüft, ob der Token gültig ist. Dies geschieht in der Funktion *getServerSideProps* der jeweiligen Seite. Ist kein gültiger Token vorhanden, wird der Nutzer auf die Login-Seite weitergeleitet. Zusätzlich benötigt der Server ein simples Skript, das die User-Session-Tabelle durchläuft und alle Einträge mit abgelaufener Lebensdauer löscht. Es kann automatisch in regelmäßigen Abständen vom Server ausgeführt werden.

4.1.3 Anzeigen und Artikel

Ein Marketing-Account einer Hochschule ist notwendig, um Anzeigen und Artikel auf der Plattform zu buchen. Der Preis hängt von verschiedenen Faktoren ab, die der Kunde selbst ändern kann. Anzeigen sind gesponserte Werbeanzeigen, die an verschiedenen Stellen auf der Plattform erscheinen können und als solche klar gekennzeichnet sind. Sie können entweder für eine Hochschule selbst, für einen bestimmten Studiengang einer Hochschule oder für einen Artikel einer Hochschule werben. Artikel sind definiert als Pressemitteilungen oder gesponserte Inhalte. Kunden können längere und formatierte Texte erstellen, um z. B. Veranstaltungen, Veränderungen auf dem Campus oder neue Studiengänge zu bewerben.

4.1.3.1 Anzeigen Erstellen

Anzeigen können im „App-Bereich“ der Plattform (zugänglich nach Login) unter dem Menüpunkt „Neue Anzeige“ gebucht werden. Zuerst muss der Typ festgelegt werden, Anzeige oder Artikel (4.1.3.4 Artikel Erstellen). Eine Anzeige verlinkt auf eine

bestimmte Seite der Plattform, entweder auf eine Hochschule, einen Studiengang oder einen Artikel. Der Titel entspricht standardmäßig dem Namen der Hochschule bzw. des Studiengangs, kann aber vom Kunden angepasst werden. Die Größe bestimmt, ob ein Bild vorhanden ist oder nicht (mehr dazu in 4.1.3.2 Anzeigenlayout). Eine kurze Beschreibung und ein Bild (bei den Größen „Groß“ und „Medium“) sind ebenfalls nötige Angaben. Der Zeitraum, in dem die Anzeige aktiv sein soll, hat großen Einfluss auf den Preis der Buchung. Er wird in Tagen angegeben, angefangen bei einem Tag nach dem aktuellen Tag. Eine Vorschau zeigt, wie die Anzeige für den Benutzer aussehen wird (siehe Abbildung 6: Benutzeroberfläche zum Erstellen von Werbeanzeigen).

Link Details

Verknüpftes Objekt *

☐ Institution ☒ Studiengang

Mehr erfahren

Studiengang *

Mehr erfahren

Titel *

Mehr erfahren

Größe *

☐ Groß ☒ Medium ☐ Klein

Mehr erfahren

Beschreibung *

Mehr erfahren


Bild *

Mehr erfahren

Zeitraum auswählen *

Mehr erfahren

Vorschau



Applied Life Sciences
[Institutionsname]

Viel Biologie und Chemie!

Übersicht

4 Tage ausgewählt

2,00 € geschätzte Gesamtkosten

Abschicken

Abbildung 6: Benutzeroberfläche zum Erstellen von Werbeanzeigen

Diese Funktionalität wurde mit Mantine-Komponenten in einem Formular implementiert. Die Daten werden an die Next.js API gesendet und überprüft. Bei fehlerhaften Angaben wird eine Fehlermeldung an den Nutzer gesendet, ansonsten wird der Nutzer zum Zahlungsanbieter (siehe 3.3.1 Analyse) weitergeleitet. Nach erfolgreichem Abschließen

der Bezahlung wird die Anzeige in der Datenbank gespeichert und ab dem Startdatum auf der Plattform angezeigt.

4.1.3.2 Anzeigenlayout

Um die drei verschiedenen Größen der Anzeigen anzuordnen, wird ein neuer Wert erstellt, der auf der Größe einer Anzeige basiert: die Kosten. Anzeigen werden in Reihen angeordnet, jede Reihe hat maximal 8 Einheiten. Große Anzeigen kosten vier Einheiten, mittlere zwei Einheiten und kleine eine Einheit. Zusätzlich werden Anzeigen nach Größe absteigend sortiert (von links nach rechts, von oben nach unten). Im Beispiel in Abbildung 7 sind eine große, eine mittlere und zwei kleine Anzeigen vorhanden.

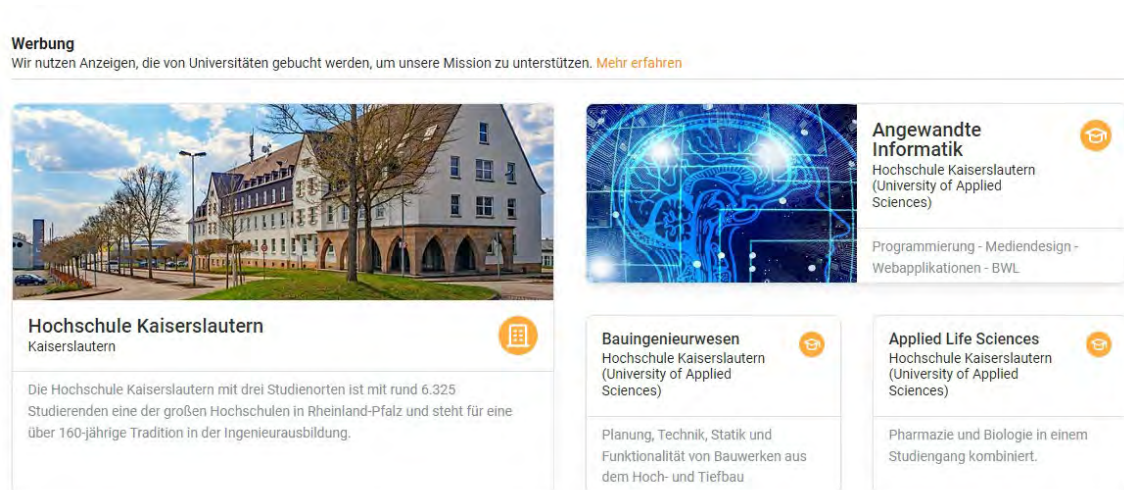


Abbildung 7: Beispiel des Anzeigen Layouts

4.1.3.3 Anzeigen Darstellen

Die von Hochschulen erstellten Anzeigen werden in einem getrennten Bereich auf vielen Seiten der Plattform dargestellt. Zu beachten ist, dass die Anzahl der dargestellten Anzeigen begrenzt ist. Sie werden nach Größe priorisiert, Anzeigen über dem Limit werden auf der jeweiligen Seite nicht dargestellt.

4.1.3.4 Artikel Erstellen

Die Artikel können auf der gleichen Seite wie auch die Anzeigen erstellt werden, indem der entsprechende Button oberhalb der Oberfläche „Anzeige Erstellen“ angeklickt wird. Der Titel und der Auszug des Artikels können frei gewählt werden. Zusätzlich wird ein Bild als Thumbnail benötigt, wie gut das Format des hochgeladenen Bildes passt, kann der Kunde in der Vorschau erkennen. Die Erstellung der eigentlichen Inhalte ist trivial, die Anwendung bietet einen voll funktionsfähigen WYSIWYG-Editor. Alle gängigen

Textformatierungen werden unterstützt, weitere können nachträglich hinzugefügt werden.

Der Texteditor basiert auf einer in Mantine angepassten Version von Tiptap²¹. Der Inhalt wird als JSON exportiert. Der exportierte JSON-Inhalt basiert auf dem strikten ProseMirror Schema, das definiert, wie der Inhalt strukturiert ist. Knotenpunkte werden mit Attributen versehen und ineinander verschachtelt. Es wird ein neuer Eintrag in der Datenbanktabelle „USER_ARTICLE“ (siehe 3.7.1.13) angelegt, der Content wird dabei als JSON abgespeichert.

The screenshot shows a web form for creating an article. At the top, there's a 'Type' dropdown with options 'Link' and 'Press release / Sponsored article'. Below this are four main sections, each with a 'Learn more' link: 'Title' (with a text input containing 'Offener Campus'), 'Excerpt' (with a text input containing 'Die Hochschule Kaiserslautern hat am 10.03.2023 einen Tag des offenen Campus...'), 'Image' (with a file input showing '640px-Hochschule_KL_Campus_ZW_Gebäude_H.jpg'), and 'Content' (a rich text editor with a toolbar and a character count of '45/1000 characters'). The 'Content' section also has a list of tags: 'Adresse', 'Ablauf', 'Programm', and 'Weitere Informationen...'. At the bottom, there's a 'Summary' section showing '€10.00 estimated total cost.' and a prominent orange 'Submit booking' button.

Abbildung 8: Benutzeroberfläche, um Artikel zu erstellen

4.1.3.5 Artikel Darstellen

Nach der Veröffentlichung kann der Artikel auf der entsprechenden Hochschuleite und in der Liste aller Artikel und Pressemitteilungen auf der Seite „/news“ gefunden werden. Artikel sind als Karten gestaltet, mit einem Vorschaubild, Textausschnitt, Erstellungsdatum, Sprache und dem Namen der Hochschule. Um den Inhalt des Artikels darzustellen, muss er vorher verarbeitet werden, da er noch in Form des JSON-Schemas

²¹ Überdosis, *Tiptap* (Überdosis), zuletzt geprüft am 14.06.2023, <https://tiptap.dev/>.

existiert. Dafür ist die eigens erstellte Klasse „TipTapParser“ (src: lib/parser/tiptapParser.tsx) zuständig. Die Knotenpunkte werden rekursiv in Mantine Komponenten umgewandelt und bekommen die jeweiligen Attribute zugewiesen. Diese können dann von Next dargestellt werden.

4.1.3.6 Geplante Erweiterungen

Durch die zeitliche Beschränkung konnten zwei Funktionen nicht mehr rechtzeitig implementiert werden, die Mehrsprachigkeit und die genaue Platzierung von Anzeigen. Die Grundlagen für beide sind bereits vorhanden.

Im Fall der Mehrsprachigkeit werden Texte mit dem jeweiligen Sprachcode in der Datenbank gespeichert, dieser wird dann auch bei dem Auslesen beachtet. Es fehlt aber noch die Benutzeroberfläche um Titel, Beschreibung oder Text in mehreren Sprachen zu verfassen.

Auch die genaue Seitenauswahl für Anzeigen ist teilweise implementiert, in der Spalte „*placement*“ der Datenbanktabelle „USER_AD“ wird festgelegt auf welchen Seiten die Werbeanzeige dargestellt werden soll. Dabei können Bedingungen festgelegt werden, wie z. B. „nur auf den Seiten der Städte in Rheinland-Pfalz anzeigen“. Es fehlt jedoch die passende Benutzeroberfläche.

4.1.4 Messung der meistbesuchten Seiten

Auf der Startseite sind mehrere Abschnitte, in denen die bei den Besuchern beliebtesten Seiten hervorgehoben werden. Dazu gehören die meistbesuchten Seiten von Studiengängen, Hochschulen, Studiengangsrichtungen und Städten. Es wird periodisch geprüft und berechnet werden, wie beliebt diese Seiten sind.

Die gewählte Lösung ist zeitsparend, hat aber den Nachteil, dass sie nicht gut bei hohen Nutzerzahlen skaliert. Eine spezielle Datenbanktabelle wurde erstellt, um jeden einzelnen Aufruf einer bestimmten Seite zu zählen. Jeder Eintrag in der Tabelle entspricht einem Klick. Der Typ der geöffneten Seite und deren ID werden in zwei weiteren Spalten gespeichert. Die Tabelle wird täglich vom Server ausgelesen, dabei wird der Popularitätswert jeder Seite mit dem neuen Wert berechnet. Durch die unterschiedliche Gewichtung der beiden Werte kann ein ungefährer Trend ermittelt werden.

4.1.4.1 High-Performance Alternative

Es gibt verschiedene Möglichkeiten, um eine bessere Performance bei hohen Nutzerzahlen zu ermöglichen. Da dies aber erst bei schätzungsweise mehr als 10 000 gleichzeitigen Nutzern ein Problem wird, kann es zunächst ignoriert werden. Zu den möglichen Alternativen gehören Redis und Kafka. Redis ist ein Datenbanksystem, das die Daten im Arbeitsspeicher des Servers halten kann, um so hohe Lese- und

Schreibgeschwindigkeiten zu erzielen. Kafka (auch: „Apache Kafka“) dient dem Verarbeiten und Speichern von Datenströmen. Es stellt eine Schnittstelle zu Drittsystemen bereit und ist durch eine hohe Skalierbarkeit für Big-Data-Anwendungen geeignet. Die genannten Lösungen sprengen den Rahmen des aktuellen Projektes und sollten nur in der Zukunft bedacht werden, sollten Performanceprobleme auftreten.

4.1.5 Automatisch eine Sitemap generieren

Sitemaps sind ein wichtiges Hilfsmittel für Suchmaschinen, um neue URLs auf der Webseite zu finden. Um eine möglichst gute SEO-Performance zu ermöglichen, hat sich dieses Projekt maßgebend an der offiziellen Dokumentation²² von Google orientiert. Da die Plattform ein absolutes Minimum von 50 000 Unterseiten hat, ist eine Sitemap zwingend notwendig.

Implementiert ist das automatische Generieren der Sitemap mit Hilfe der Bibliothek *next-sitemap*²³. Durch Auslesen der Datenbank können alle vorhandenen Seiten dynamisch in die Sitemap eingetragen werden. Dabei bietet *next-sitemap* weitere Funktionen, die für SEO von Vorteil sind. Dazu gehört das automatische Splitten der Sitemap in mehrere Dateien, die von der Hauptdatei referenziert werden. Die Angabe des Zeitpunkts der letzten Änderung ist für einige Seiten nützlich, z. B. für die Social-Media-Analyse. Dadurch können Suchmaschinen erkennen, wann eine bestimmte Seite im Index aktualisiert werden sollte.

4.1.6 Paginierungen mit SEO

Paginierung bedeutet, dass Seiten, die zu viele Elemente in einer Liste enthalten, auf mehrere Seiten verteilt und durch Buttons und Links miteinander verbunden werden. Dies wird auf mehreren Seiten der Plattform angewendet, z. B. bei der Auflistung von Hochschulen, Studiengängen und Städten in einem bestimmten Bundesland.

4.1.6.1 Bedeutsame Eigenschaften für die SEO-Performance

In der offiziellen Dokumentation²⁴ gibt Google Empfehlungen zur SEO-Performance:

- Die aktuelle Seite wird mit einem Query-Parameter angegeben. Dafür bietet Next.js die „*useRouter()*“ Funktion. Weitere Query-Parameter existieren für die Suchanfrage und die Reihenfolge der Objekte.

²² Google LLC, „Sitemaps,“ zuletzt geprüft am 14.06.2023, <https://developers.google.com/search/docs/crawling-indexing/sitemaps/overview>.

²³ Vishnu Sankar, *next-sitemap* (Vishnu Sankar), zuletzt geprüft am 14.06.2023, <https://www.npmjs.com/package/next-sitemap>.

²⁴ Google LLC, „Pagination, incremental page loading, and their impact on Google Search,“ zuletzt geprüft am 23.06.2023, <https://developers.google.com/search/docs/specialty/ecommerce/pagination-and-incremental-page-loading>.

- Die Seiten müssen sequenziell (vorwärts und rückwärts) verlinkt werden, mit Links zum Anfang und Ende der Liste. Zusätzlich können Seiten dazwischen verlinkt werden. Die Komponente in „*components/Pagination/CustomPagination.tsx*“ erstellt eine generische Paginierung, die genau den SEO-Anforderungen entspricht. Die Basis wird durch die Bibliothek „Mantine“ gebildet und angepasst.
- Jede Seite in der Liste muss eine eigene kanonische URL haben. Dafür ist das *Head*-Tag von Next.js zuständig, in dem die kanonische URL sowie *next* und *prev* angegeben werden.
- Jede Seite muss mit ihrer kanonischen URL in die Sitemap eingetragen werden. Es dürfen keine anderen Query-Parameter in der URL vorkommen.

4.1.6.2 Zusätzliche Funktionen: Suche und Sortierung

Da einige der Listen mehr als 100 Einträge enthalten, benötigen die Benutzer Hilfsmittel, um die Suche zu vereinfachen. Zu diesem Zweck wurde eine Such- und eine Sortierfunktion hinzugefügt. In der aktuellen Version kann nur nach dem Namen des Eintrags gesucht werden, mit Ausnahme der Städte, die auch über die Postleitzahl gefunden werden können. Die Objekte können auf- und absteigend nach Name oder nach Popularität sortiert werden. Die Implementierung der Popularität wird in 4.1.4 (Messung der meistbesuchten Seiten) näher erläutert.

4.1.7 Online-Marketing mit Lighthouse

Die gesammelten Online-Marketing-Daten, d. h. die Analysen der Hochschulwebseiten, werden auf der Plattform detailliert dargestellt. In Abschnitt 4.3 (Lighthouse-Skript & Online-Marketing-Analyse) wird erklärt, wie die Daten erhoben und gespeichert werden. Jedes Hochschulprofil verfügt über einen Reiter „Online-Marketing“, der einen Überblick über die fünf verschiedenen Kategorien gibt (S. 66, Abbildung 15: Online-Marketing-Übersicht im Profil der FH-Dortmund). Hinter jeder Kategorie befindet sich eine Seite mit einer Detailansicht der Audits, die für die Bewertung der Kategorie durchgeführt wurden (S. 67, Abbildung 16: Detailansicht „Performance“ der Online-Marketing-Analyse). Da die Lighthouse-Daten nur als JSON-Datei in einer von Google vorgegebenen Struktur vorliegen, müssen sie zunächst in eine bessere Struktur gebracht werden. Dazu dienen die Dateien im Ordner „*lighthouse*“ im Projektpfad „*lib/lighthouse/...*“. Die Funktion „*getMinifiedLhrCategory()*“ wandelt eine bestimmte Kategorie in ein reduziertes TypeScript-Objekt um, das nur die notwendigen Daten enthält. Dadurch wird die Größe der von Next.js gesendeten HTML-Datei klein gehalten.

Die Konvertierung der JSON-Daten ist eine komplexe Programmierarbeit, die sich aber nicht vermeiden lässt. Derzeit ist das Konvertierungskonzept fest codiert und setzt voraus, dass Lighthouse die Struktur nicht verändert. Eine mögliche Verbesserung wäre, die

Funktion dynamischer zu gestalten, um unabhängiger zu werden und Fehler zu vermeiden.

4.1.8 Objektkarten für eine bessere Suchmaschinenoptimierung

Innerhalb einer Linkliste, wie z. B. der Liste der Studiengänge, werden mehr Informationen als nur ein Link und der Name angegeben. Dies hat Vorteile für den Benutzer und für die Suchmaschinenoptimierung. Es werden so viele relevante Details wie möglich angezeigt, die - wenn möglich - mit der entsprechenden Seite verlinkt sind. Die Implementierung aller Objektkarten kann im Ordner `./components/Card/...` eingesehen werden. Es gibt 13 verschiedene Karten:

- Studiengang
- Studiengangsrichtung
- Hochschule
- Stadt
- Bundesland
- Land
- Online-Marketing
- Social-Media
- Social-Media-Kategorie
- Artikel
- Werbeanzeige (klein, mittel, groß)

Im Anhang D: Screenshots sind Beispielbilder der Objektkarten enthalten.

4.1.9 Von Nutzern hochgeladene Bilder speichern und darstellen

Bei der Erstellung von Artikeln und Anzeigen können Benutzer eigene Bilder hochladen (4.1.3 Anzeigen und Artikel). Diese werden im Dateisystem des Servers gespeichert und erhalten einen Eintrag in der Datenbank (3.7.1.14 USER_IMAGE) mit Verweis auf die entsprechende Bilddatei. Um die Bilder anzeigen zu können wurde eine Next.js API im Pfad `./pages/api/image/[imageId].ts` erstellt. Diese erwartet in den Abfrageparametern eine ID und einen Dateityp. Sie durchsucht die Datenbank nach einem passenden Bild und gibt es bei einem Treffer als Antwort zurück. Um Bilder aus der API anzuzeigen, kann nun der API-Link als `src` eines Bildes gesetzt werden.

4.2 Hauptcrawler

Der Hauptcrawler ist im Wesentlichen eine Sammlung verschiedener Skripte und Funktionen, deren Hauptaufgabe darin besteht, die benötigten Daten zu sammeln und in der Datenbank zu speichern. Es handelt sich um eine „Command Line Application“ (dt.

Kommandozeilen-Anwendung) in Node.js, deren verschiedene Funktionen mit Parametern aufrufbar sind. Dadurch können sie entweder manuell durch einen Admin oder automatisch durch den Server aufgerufen werden. Dafür muss das Skript selbst mit entsprechenden Parametern gestartet werden, z. B. „*npm run dev full-setup*“ würde die Datenbank schnell in einem Befehl initialisieren, bietet dem Entwickler allerdings keine Möglichkeit den Prozess zu kontrollieren oder dabei einzugreifen.

4.2.1 Länder und Bundesländer initialisieren

Die entsprechende Funktion wird mit den Parametern „*db init-all*“ aufgerufen. Die Länder und deren Bundesländer sind fest codiert. Der Grund dafür ist der geringe Aufwand im Vergleich zu einer dynamischen Lösung. Auch wenn diese Funktion in der Live-Anwendung nur einmal ausgeführt werden muss, ist es während der Entwicklung trotzdem von großem Vorteil, mit wenigen Befehlen eine neue Datenbank für die Plattform vorbereiten zu können. Die hartkodierten Daten werden mit Hilfe von Prisma in die Datenbank eingefügt, sind aber noch auf Deutschland beschränkt.

4.2.2 Hochschuldaten verarbeiten

Mit dem Parameter „*parse-hrk-data*“ werden mehrere Funktionen ausgeführt, die eine gegebene *.xlsx*-Datei verarbeiten und das Ergebnis in zwei Dateien speichern, die relevanten Städte und die Hochschuldaten.

Die Ausgangsdatei wurde dem Forschungsprojekt von der HRK zur Verfügung gestellt. Sie enthält alle Hochschulen in Deutschland, deren Studiengänge und einige Basisinformationen wie Web-Adresse und Standort. Sie wird mit Hilfe der Bibliothek „*Sheet.js*“²⁵ verarbeitet, die Funktion „*parseHrkFile*“ extrahiert die Daten aus der Datei und verknüpft sie miteinander. Problematisch ist, dass die Funktion stark von der Quelldatei abhängig ist. Daher muss sie bei jeder Änderung angepasst werden.

Ein weiteres Problem war die Art, in der die Datei einer Hochschule mehrere Standorte zuordnet: für jeden Standort existiert eine neue Zeile. Die in der Funktion „*mergeInstitutionDuplicates*“ enthaltene Logik sucht nach Duplikaten und führt diese zusammen.

Im nächsten Schritt werden die gesammelten Daten in die zwei Dateien gespeichert. Die erste ist eine einfache Sammlung aller Hochschulen, Studiengänge und ihrer Attribute. Die zweite ist komplexer und soll die Namen aller Städte enthalten, in denen es mindestens eine Hochschule gibt. Dazu werden zunächst alle Namen der Städte gesammelt und dann mit der Liste aller Städte aus 4.2.3 verglichen. Der Vergleich muss

²⁵ SheetJS LLC, *SheetJS* (SheetJS LLC), zuletzt geprüft am 30.05.2023, <https://sheetjs.com/>.

unterschiedliche Schreibweisen der Namen umgehen können, in der Zukunft könnte der Algorithmus möglicherweise mit der Levenshtein-Distanz verbessert werden.

Die Parameter „*db push-institutions*“ und „*db push-subjects*“ speichern die gesammelten Hochschulen und Studiengänge in der Datenbank ab.

4.2.3 Liste aller Städte einfügen

Die Datenbanktabelle für Städte enthält nur Städte, die mit Hochschulen verknüpft sind, beschränkt sich aber noch auf Deutschland. Realisiert wird dies, indem die passende Tabelle²⁶ von „opendatasoft“ als JSON exportiert wird. Mit den Parametern „*db push-cities*“ werden die in 4.2.2 gesammelten Städtenamen mit der Liste aller Städte verglichen. Alle Städte mit Verbindungen zu Hochschulen können jetzt in die Datenbank eingefügt werden (Tabelle: 3.7.1.3 CITY). Dabei wird eine einzigartige URL durch eine Funktion („*getUrlFromName*“) generiert, die ungültige Zeichen ersetzt.

Durch Aufruf des Skriptes mit den Parametern „*db push-institution-locations*“ wird die Verknüpfungstabelle für Hochschulstandorte, die Hochschulen mit Städten verbindet, gefüllt.

4.2.4 Studiengangsrichtungen sammeln und übersetzen

Die Studiengangsrichtungen wurden indirekt schon in 4.2.2 gesammelt, müssen aber noch verknüpft in die Datenbank eingefügt werden. Ausgeführt wird dies durch den Parameter „*db push-subject-types*“. Wie in 3.7.1.6 CATEGORY beschrieben, haben Studiengänge nur den nativen Namen und eine englische Übersetzung, die API vom DeepL Translator generiert wird.

Durch den Aufruf mit den Parameter „*db push-subject-type-unions*“ wird die notwendige Verknüpfungstabelle befüllt, da es sich um eine n:m Beziehung handelt.

4.2.5 Hochschulwebseiten nach Social-Media-Links scrapen

Um Social-Media-Analysen durchführen zu können, werden zuerst alle Profile der Hochschulen auf den sozialen Medien benötigt. Da die Webseiten der Hochschulen bereits aus 4.2.2 bekannt sind, bietet es sich an diese nach Links zu sozialen Medien zu durchsuchen.

Die aktuelle Implementierung nutzt dafür die Node.js Bibliothek „crawler“. Es wird eine Liste aller Hochschulwebseiten übergeben, die dann abgearbeitet wird. Dabei wird die gesamte Startseite nach Links durchsucht, die mit verschiedenen Regex-Schemas verglichen werden, um Links zu Facebook, Twitter, Youtube und Instagram Profilen zu

²⁶ „Postleitzahlen - Germany,“ Opendatasoft, zuletzt geprüft am 30.05.2023, <https://public.opendatasoft.com/explore/dataset/georef-germany-postleitzahl/table/>.

finden. An der Suche wurden zahlreiche kleine Verbesserungen vorgenommen, um falsch-positive Funde zu verringern. Das größten Probleme sind Hochschulen, die keine eigene Webseite haben, bspw. Polizei-Fachhochschulen, die auf der Webseite des jeweiligen Bundeslandes vertreten sind. Diese mussten manuell gefunden und hartkodiert ausgefiltert werden. Die Ergebnisse werden in der Datei „*socialMediaData.json*“ mit der URL der Webseite als ID gespeichert.

4.2.5.1 Mögliche Verbesserung

Eine große Schwäche der gewählten Lösung ist JavaScript. In Tests waren bei 8% der Hochschulen die Social-Media-Links mit JavaScript eingebunden, und damit unsichtbar für den Crawler. Getestet wurde dies durch eine manuelle Prüfung der Seiten, zu denen keine Links zu sozialen Medien gefunden wurden.

Die mögliche Alternative ist Playwright, ein Automatisierungstool für Webbrowser, das auch in 4.4 zum Erstellen von Screenshots verwendet wurde. Es kontrolliert einen Browser und kann somit ohne Probleme JavaScript ausführen. Der Nachteil ist die komplexere Implementierung und schlechtere Performance. Außerdem wird vorausgesetzt, dass das System, auf dem das Tool ausgeführt wird, eine grafische Oberfläche unterstützt. Dies kann insb. bei Serverumgebungen ein Problem werden.

4.2.6 Social-Media APIs

Zu jeder Hochschule werden möglichst viele Daten der Social-Media-Profile gesammelt. Da die Authentifizierung durch Meta nicht rechtzeitig akzeptiert wurde, mussten die APIs von Instagram und Facebook übersprungen werden. Es wird davon ausgegangen, dass sie nachträglich hinzugefügt werden, weshalb entsprechende Referenzen im Code und in der Datenbank vorhanden sind. Alle API spezifischen Funktionen befinden sich in „*src/api/...*“ in den Ordnern der jeweiligen sozialen Medien.

4.2.6.1 Youtube

Die Kommunikation mit der Youtube API ist mithilfe der offiziellen Node.js Bibliothek „*googleapis*“ realisiert. Die Datei „*src/api/Youtube/YoutubeApiHandler.ts*“ enthält eine komplexe Klasse, um die Verbindung aufzubauen und die gewünschten Daten auszulesen. Nach jedem Schritt wird der Fortschritt lokal in einer Datei gespeichert, um der eventuellen Limitierung der API vorzubeugen. Dadurch kann das Skript an mehreren Tagen nacheinander ausgeführt werden, arbeitet bis die API-Limits erreicht sind, und kann am nächsten Tag an der gleichen Stelle weitermachen.

Gegeben ist eine Liste von URLs der Youtube-Kanäle aller Hochschulen. Die Hilfsfunktion „*getChannelTitleFromURL*“ nutzt ein Regex-Muster, um entweder die Channel-ID oder den Nutzernamen zu finden. Die URL eines Youtube-Kanals kann auf viele verschiedene Arten geschrieben werden, die alle durch das Regex-Pattern abgedeckt

werden, um zuverlässig die nötigen Informationen zu finden. Das Ergebnis ist eine Liste von IDs und Nutzernamen der Youtube-Kanäle.

Die IDs können übernommen werden, die Nutzernamen reichen jedoch nicht aus, um Anfragen zu machen. Darum wird mit der Funktion „*getChannelID*“ nach Kanälen mit entsprechenden Namen gesucht. In den meisten Fällen liefert die API das gewünschte Ergebnis, kann in Einzelfällen jedoch den falschen Kanal zurückgeben. Youtube bietet keine andere Lösung, weshalb diese Einzelfälle von Hand korrigiert werden müssen. Nach diesem Schritt besteht die Liste nur noch aus Kanal-IDs.

Die Liste der IDs wird iteriert, um die Metadaten der Kanäle zu sammeln. Es wird versucht, lokal verfügbare Daten zu verwenden, wenn diese nicht älter als 30 Tage sind. Die Funktion „*getChannelStatistics*“ ruft die Youtube-API mit einer Liste von maximal 50 Kanal-IDs auf. Deshalb muss die ganze Liste in mehrere Teillisten aufgeteilt werden, um sie alle einzeln an die API zu senden. Als Ergebnis wird eine Objektliste aller Youtube-Kanäle zurückgegeben, die wichtige Metainformationen wie die Anzahl der Abonnenten enthält.

In dieser Liste fehlen jedoch noch die Daten über die hochgeladenen Videos der Kanäle. Dieser Schritt ist durch die limitierte API in zwei Teile geteilt, das Sammeln der Video-IDs und das Sammeln der eigentlichen Video-Daten. Das zweite Zeichen bei Channel-IDs auf Youtube ist immer „C“, wird dieses zu „U“ geändert, ist das Ergebnis die ID der „Uploads“ Playlist des Kanals. Die Funktion „*getVideoIds*“ nutzt diese ID, um eine feste Anzahl von Video-IDs abzufragen, die sich in der Playlist befinden. Jeder Hochschul-Kanal hat damit auch eine Liste der IDs hochgeladener Videos.

Um den Prozess abzuschließen, müssen zu jeder dieser Video-IDs die entsprechenden Video-Daten abgefragt werden. Zuständig dafür ist die Funktion „*getVideoStatistics*“. Sie teilt jede Liste von Video-IDs in Unterlisten mit je 50 IDs. Diese werden dann an die API gesendet, um Details zu jedem Video zu sammeln.

Das Ergebnis ist eine Datei im lokalen System, die Metadaten zu allen Youtube-Kanälen und einer beliebigen Anzahl ihrer Videos enthält.

4.2.6.2 Twitter

Die Anbindung an die Twitter API erfolgt über die inoffizielle Node.js Bibliothek „*twitter-api-v2*“²⁷. Die Klasse in der Datei „*src/api/Twitter/TwitterApiHandler.ts*“ sendet Anfragen an die API und liest die gewünschten Daten aus. Wie auch bei der Youtube-API-Funktion werden die Daten nach jedem Zwischenschritt lokal gespeichert. Durch das

²⁷ Paul-Louis Hery, *node-twitter-api-v2*, zuletzt geprüft am 20.06.2023, <https://www.npmjs.com/package/twitter-api-v2>.

Zwischenspeichern kann das Skript über mehrere Tage ausgeführt werden, als Lösung für die bestehenden API-Limits.

Im ersten Schritt werden die Benutzernamen der Hochschulen aus der Liste der Twitter-Links gefiltert und gespeichert. Da die Anzahl der Profile, die gleichzeitig abgefragt werden können, auf 100 beschränkt ist, muss die gesamte Liste in Blöcke zu je 100 aufgeteilt werden. Die Funktion „*getUserProfiles*“ sendet die Benutzernamen an die API und erhält als Antwort die Metadaten zu jedem Profil.

Die Liste der Metadaten wird im nächsten Schritt um die Tweet-Daten ergänzt. Dazu lädt die Funktion „*getUserTweets*“ die gewünschte Menge an Tweets aus der Timeline des Profils. Diese werden mit den Metadaten in einem Objekt pro Profil kombiniert und lokal gespeichert.

Das Ergebnis ist eine Datei auf dem lokalen System. Sie enthält Metadaten zu allen Twitter-Profilen und eine beliebige Anzahl ihrer Tweets.

4.2.6.3 Verbesserungen zum Vorgänger

Der grobe Ablauf dieser Prozesse war bereits durch die Vorgängerarbeit „Evaluation und Implementierung von Social-Media-Ratings von Hochschulen für ein Webportal“²⁸ vorgegeben, wurde aber grundlegend überarbeitet. In diesem Projekt wurden Fehler behoben, die Genauigkeit erhöht und die Sammlung von Videos und Tweets neu implementiert. Dadurch wird besser mit den Rate-Limits der APIs umgegangen und es werden nur die korrekten Kanäle und Profile gefunden. Außerdem wurden die Übersichtlichkeit und Code-Struktur verbessert, um das Skript wartbarer zu machen.

4.2.7 Social-Media: Bewertungsstrategie und Analyse

Ziel ist es, die Präsenz der Hochschulen in den sozialen Medien zu analysieren und zu bewerten. Die für dieses Projekt ausgewählten Social-Media-Plattformen sind Twitter und Youtube mit der Option, zu einem späteren Zeitpunkt Facebook und Instagram hinzuzufügen. TikTok wäre ebenfalls relevant, die entsprechende API ist zum Zeitpunkt des Projekts in Europa nicht verfügbar. Eine Excel-Tabelle mit Beispielen und Details zum neuen Bewertungsalgorithmus befinden sich auf dem beigelegten Datenträger unter dem Namen „social-media-score-algorithm“ (siehe Anhang B: Datei- und Ordnerverzeichnis).

4.2.7.1 Vorgänger

Bewertungsstrategien wurden bereits in den Projekten „Evaluation und Implementierung von Social-Media-Ratings von Hochschulen für ein Webportal“ - Schuler (2022) und

²⁸ Schuler, „Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal.“

„Market, Society, and Education – University Marketing“ - Kloppenburg (2012) erstellt. Sie wurden als Ausgangspunkt für dieses Projekt genutzt, aber weiter angepasst und verbessert. Der Bewertungsalgorithmus selbst wurde angepasst, um die Social-Media-Performance der Hochschulen besser abbilden zu können. Dazu wurden zusätzliche Metriken erhoben und analysiert, die in den früheren Projekten nicht zur Verfügung standen. Darunter insbesondere die Impressionen der Tweets, die in der aktuellen API offen ausgelesen werden können. Zuvor mussten diese anhand der Anzahl der Follower des Autors geschätzt werden.

4.2.7.2 Bewertungsalgorithmus - Einführung

Die Bewertung basiert unabhängig von der Social-Media-Plattform auf zwei Werten, dem Profilwert und dem Interaktionswert. Die Berechnung hängt von der jeweiligen Plattform ab. Die Algorithmen sind für alle sozialen Medien gleich aufgebaut, unterscheiden sich jedoch in den Metriken, die für die Berechnung verwendet werden. Die relevanten Dateien können im Pfad „src/rating/...“ des Teilprojekts „uni-index-crawler“ gefunden werden. Um alle Hochschulen zu analysieren und zu bewerten, muss das Skript mit dem Parameter „social-ratings-to-file“ aufgerufen werden.

4.2.7.3 Bewertungsalgorithmus für Youtube-Kanäle

Der Interaktionswert wird aus den durchschnittlichen Likes, Favoriten und Kommentaren pro Video berechnet. Er gibt an, wie oft die Nutzer im Durchschnitt mit den einzelnen Videos interagieren. Die Durchschnittswerte der Videos werden nur für die geladene Stichprobe berechnet, nicht für alle Videos im Kanal. Die Stichprobe hatte eine Größe von 150 Videos, kann aber beliebig angepasst werden.

$$Interaction = averageLikes + averageFavourites + averageComments$$

Der Profilwert ist die Summe aus Abonnenten, Videos und durchschnittlichen Videoaufrufen. Die durchschnittlichen Videoaufrufe werden aus der Gesamtzahl der Videos und Aufrufe berechnet.

$$Profile = Subscribers + Videos + \left(\frac{Viewcount}{Videos}\right)$$

Das erste unskalierte Rating ist das Produkt aus Interaktionswert, Profilwert und einem Multiplikator. Durch die Multiplikation von Interaktions- und Profilwert, können gute Bewertungen für hohe Zahlen von Abonnenten nur erreicht werden, wenn die entsprechende Interaktion vorhanden ist. Zusätzlich wird noch der Profilwert addiert, um ihm eine stärkere Gewichtung zu garantieren. Der Multiplikator besteht aus einem oder mehreren Werten, die das Rating um einen gewissen Prozentsatz beeinflussen können. In der Formel für Youtube, besteht er aus einem Wert, dem Meta-Angaben-Check. Er wird gesetzt, wenn mindestens 90% der Videos im Kanal fünf Tags und eine Beschreibung

von angemessener Länge (mindestens die Hälfte der erlaubten Zeichen, d. h. 400 Zeichen) haben. Ist diese Bedingung erfüllt, wird der Multiplikator auf 1,05 gesetzt, und erhöht somit das Rating um 5 %.

$$Rating = (Interaction * Profile * Multiplier) + Profile$$

Das maximale Rating kann erst ermittelt werden, wenn die Ratings für alle Youtube Kanäle der Hochschulen berechnet wurden. Es entspricht dem höchsten Wert in der Liste.

$$MaxRating = \max(Rating_1, Rating_2, \dots, Rating_n)$$

Der Normalisierungswert löst ein wesentliches Problem, das ansonsten bei den Bewertungen auftritt. Da die Differenz zwischen dem besten und dem zweitbesten Kanal so groß ist, würden die Bewertungen relativ zum besten Kanal alle null werden. Durch eine logarithmische Skalierung wird dieser Effekt abgeschwächt.

$$Norm = \log(MaxRating)$$

Der Normalisierungswert ist die Grundlage für die Berechnung der Endbewertung von null bis zehn für jeden Kanal. Dafür wird der Logarithmus des Ratings durch den Normalisierungswert geteilt, und mit einer Skalierung potenziert. Der Skalierungsfaktor kann frei eingestellt werden. Höhere Werte verstärken den Kompressionseffekt, die Unterschiede zwischen den Bewertungen werden geringer. Niedrigere Werte schwächen den Kompressionseffekt und führen zu größeren Unterschieden zwischen den Bewertungen. Der Skalierungsfaktor 1,25 lieferte im Projektverlauf die besten Ergebnisse. Da das Ergebnis eine Zahl zwischen null und eins ist, wird es mit zehn multipliziert, um eine Bewertung zwischen null und zehn zu erhalten.

$$Bewertung = \left(\frac{\log(Rating)}{Norm} \right)^{Skalierung} * 10$$

Die Bewertungen werden auf zwei Stellen gerundet und in einer lokalen Datei gespeichert.

4.2.7.4 Bewertungsalgorithmus für Twitter-Profile

Aufgrund der Ähnlichkeit der Struktur mit dem Bewertungsalgorithmus von Youtube wird in diesem Abschnitt nur auf die Unterschiede eingegangen.

Der Interaktionswert bezieht sich auf die Tweets. Er errechnet sich aus der Summe der durchschnittlichen Likes, Retweets, Antworten und Zitationen. Aufgrund der starken Limitierungen der Twitter API, muss eine Stichprobe der letzten 150 Tweets pro Profil verwendet werden. Die durchschnittlichen Werte werden aus der Stichprobe ermittelt.

$$Interaction = averageLikes + averageRetweets + averageReplies + averageQuotes$$

Der Profilwert ist die Summe aus:

- Nutzern die der Hochschule folgen
- Nutzer denen die Hochschule folgt
- Die Anzahl der Tweets der Hochschule
- Die Anzahl der Listen in denen die Hochschule erscheint
- Die durchschnittlichen Impressionen eines Tweets

Da Twitter keine Daten zu den Gesamtimpressionen der Tweets gibt, werden die durchschnittlichen Impressionen durch die Stichprobe (n) berechnet.

$$Profile = Followers + Following + Listed + Tweets + \left(\frac{\sum Tweet Impressions}{n} \right)$$

Die Berechnung des Ratings und aller folgenden Werte entspricht dem Bewertungsalgorithmus für Youtube (4.2.7.3). Eine Ausnahme bildet der Multiplikator, der sich für Twitter aus mehreren Werten zusammensetzt. Der Erste ist an die Bedingung geknüpft, dass die Website der Hochschule direkt in deren Profil verlinkt ist; Er macht 5 % aus. Weitere 5 % werden vergeben, wenn das Profil der Hochschule auf Twitter verifiziert ist. Mehrere Multiplikatoren werden multiplikativ in der Berechnung des Ratings angewandt.

4.2.7.5 Weitere Werte berechnen und Prozess abschließen

Mehrere zusätzliche Metriken werden berechnet und den Metadaten des jeweiligen Social-Media-Objekts hinzugefügt.

Eine davon ist die durchschnittliche Anzahl von Videos pro Monat auf Youtube und die durchschnittliche Anzahl von Tweets pro Monat auf Twitter. Beide werden auf die gleiche Weise berechnet. Zuerst wird ein Zeitraum bestimmt, in dem die Tweets / Videos der Stichprobe veröffentlicht wurden, vom ältesten bis zum jüngsten. Die Anzahl der Posts in dem Zeitraum wird durch die Anzahl der Monate geteilt und ergibt die durchschnittlichen Posts pro Monat.

Für die Youtube-Kanäle werden die fünf am häufigsten verwendeten Tags ermittelt. Dazu werden alle Tags aller Videos in der Stichprobe aufgelistet und gezählt. Die fünf am häufigsten verwendeten Tags werden gespeichert.

Die fünf besten Youtube-Videos eines Kanals werden nach der Anzahl der Likes, Kommentare, Favoriten und Aufrufe bewertet. Bei den fünf besten Tweets werden die Metriken Likes, Retweets, Antworten, Zitationen und Impressionen berücksichtigt. Sie werden auch dem Ergebnisobjekt hinzugefügt.

Alle Ergebnisse werden nach Hochschule und Social-Media-Plattform getrennt in einer lokalen Datei („socialRatings.json“) gespeichert.

4.2.8 Social-Media Länderdaten berechnen

Jedes Land mit Hochschuldaten auf der Plattform verfügt über eine eigene Länderstatistik. Diese kann mit anderen Ländern verglichen werden, um zu sehen, wie unterschiedlich Hochschulen in den verschiedenen europäischen Ländern in sozialen Medien agieren.

Dafür werden von allen Metriken Durchschnittswerte erstellt. Diese beziehen sich auf die Anzahl der Profile auf der jeweiligen Social-Media-Plattform und nicht auf die Gesamtzahl der Hochschulen im Land. Sie werden nach Land geordnet in einer eigenen Datenbanktabelle gespeichert (3.7.1.10 COUNTRY_SOCIALS).

4.2.9 Social-Media-Daten in der Datenbank speichern

Durch den Start des Skripts mit den Parametern „*db push-social.media*“, werden die in der lokalen Datei gesammelten Social-Media-Daten an die Datenbank gesendet. Dabei werden alle in der Tabelle vorhandenen Daten gelöscht bzw. überschrieben. Anschließend kann mit den Parametern „*db push-country-social-media*“ die Statistik pro Land erstellt und in der Datenbank gespeichert werden.

4.3 Lighthouse-Skript & Online-Marketing-Analyse

Um die Effektivität der Online-Marketing-Bemühungen zu bewerten, wird das Lighthouse Tool (siehe 2.1.4) von Google verwendet. Die Ergebnisse sollen automatisch für jede Hochschulwebseite berechnet und lokal gespeichert werden. Das Programm selbst basiert auf Node.js und nutzt Googles Lighthouse Toolkit (siehe 2.1.4), um die Webseiten zu analysieren. Das Teilprojekt „uni-index-lighthouse“ enthält die Implementierung dieses Programms.

4.3.1 Bewertungs-Prozess

Wie beim Screenshot-Skript wird Prisma verwendet, um alle Hochschulen und ihre Webseiten bereitzustellen. Es werden nur die ID, die URL der Webseite und die interne URL benötigt. Da die interne URL einzigartig ist, kann sie als lokaler Dateiname für die Ergebnisse genutzt werden. Bei der Iteration der Liste wird zunächst geprüft, ob die Datei bereits vorhanden ist. Wenn nicht, wird die Bewertungsfunktion gestartet. Zunächst erstellt die Bibliothek „chrome-launcher“, eine Browserinstanz von Chrome. Danach kann die Lighthouse-Funktion gestartet werden, wichtig sind die dabei angegebenen Optionen. Der Output muss im JSON-Format erfolgen, der Port muss dem des Browsers entsprechen und mehrere Audits können übersprungen werden, da sie nicht relevant sind. Das Ergebnis wird in einer lokalen Datei gespeichert und die Analyse der nächsten Hochschulwebseite startet.

4.3.2 Bewertungsstrategie

Für jede der fünf Kategorien gibt es eine unterschiedliche Anzahl Audits. Ein einzelner Audit prüft einen bestimmten Wert oder eine bestimmte Eigenschaft der Webseite. Sie werden unterschiedlich gewichtet und können weitere Details zu ihrem Ergebnis liefern. Dazu gehören konkrete Problemstellen und Verbesserungsvorschläge, wie z. B. ein langer FCP (First contentful paint) oder schwer zu lesender Text durch schlechten Kontrast. Jede Kategorie hat eine Gesamtbewertung von null bis 100, die sich aus allen Audits und deren Gewichtungen berechnet, die ihr angehören. Jeder Audit verändert die Bewertung, kann aber auch weitere Details angeben. Audits können konkrete Fehlerstellen nennen, Verbesserungsmöglichkeiten aufzählen, Fehler erklären und mehr.

4.3.2.1 Performance

In der Kategorie Performance gibt es fünf Audits, die als Metriken dienen und integral für die Performancewertung einer Webseite sind.

- Der FCP (First contentful paint) misst den Zeitpunkt, ab welchem der primäre Inhalt der Webseite sichtbar wird. Er entspricht der Zeitspanne vom ersten Laden der URL bis zum Rendern des Contents, der ohne scrollen sichtbar ist.
- Der Geschwindigkeitsindex (Eng.: "Speed-Index") misst, wie schnell der Inhalt beim Laden der Seite visuell dargestellt wird. Dazu wird ein Video des Ladeprozesses aufgenommen und der visuelle Verlauf zwischen den Frames analysiert.
- Die TBT (Total blocking time) misst die Gesamtzeit, in der eine Webseite nicht auf Benutzereingaben wie Klicks, Bildschirmberührungen oder Tastatureingaben reagieren kann.
- Der LCP (Largest contentful paint) misst, wann das größte Inhaltselement im Viewport auf dem Bildschirm dargestellt wird. Dies entspricht in etwa dem Zeitpunkt, an dem der Hauptinhalt der Seite für die Benutzer sichtbar ist.
- Der CLS (Cumulative Layout Shift) ist eine nutzerzentrierte Kennzahl zur Messung der visuellen Stabilität, die dabei hilft, zu quantifizieren, wie oft Nutzer unerwartete Layoutverschiebungen erleben - ein niedriger CLS-Wert trägt dazu bei, dass die Seite ansprechend ist.

Es gibt zahlreiche weitere Audits mit schwächerer Gewichtung in der Kategorie.

4.3.2.2 Barrierefreiheit

In der Kategorie Barrierefreiheit liegt der Schwerpunkt auf der technischen Ausführung von HTML-Elementen. Geprüft wird unter anderem der Farbkontrast von Elementen, der es Nutzern erschweren kann, Inhalte zu lesen. Die Tastaturbedienbarkeit stellt sicher, dass interaktive Elemente, wie z. B. Buttons und Links, auch über die Tastatur aufgerufen und

aktiviert werden können. Alt-Texte von Bildern geben einen Kontext für Nutzern, die auf Bildschirmlesegeräte angewiesen sind.

4.3.2.3 Best Practices

In „Best Practices“ wird die ordnungsgemäße Verwendung von HTML, CSS und JavaScript sowie die Implementierung von HTTPS und anderen bewährten Verfahren geprüft. Audits identifizieren ob APIs und Bibliotheken veraltet sind, ob JavaScript und CSS optimiert sind und vieles mehr.

4.3.2.4 SEO

Die Suchmaschinenoptimierung wird in der Kategorie SEO geprüft. Dazu gehört die korrekte Anwendung von Meta-Tags, Seitentiteln und anderen Elementen, die die Sichtbarkeit einer Webseite in Suchmaschinenergebnissen beeinflussen. Dabei werden auch Empfehlungen zur Verbesserung gegeben, bspw. die optimale Länge von Meta-Beschreibungen.

4.3.2.5 Progressive-Web-App

Die PWA-Kategorie bewertet, ob eine Webseite die von Google gestellten Kriterien erfüllt. Diese zielen darauf ab, den Nutzern ein app-ähnliches Erlebnis zu bieten. Es werden Faktoren wie die Implementierung von Service-Workern, Offline-Unterstützung und Installierbarkeit bewertet. Diese Kategorie ist optional und wirkt sich nur als Bonus auf die Gesamtbewertung aus.

4.4 Screenshot-Skript

Ziel ist es, periodisch einen Screenshot jeder Hochschulwebseite zu erstellen, um diese im Tab „Screenshots“ des jeweiligen Hochschulprofils auf der Plattform darzustellen. Verantwortlich dafür ist das Teilprojekt „uni-index-screenshots“. Es basiert auf einem Node.js Programm, das Playwright (siehe 2.1.10) nutzt, um die gewünschten Webseiten zu rendern.

4.4.1.1 Screenshot-Prozess

Zuerst wird mit Prisma eine Liste aller Hochschulwebseiten aus der Datenbank geladen. Diese werden durch ihre IDs weiterhin eindeutig identifiziert. Wenn alle URLs gesammelt wurden, wird mit Playwright ein Browser geöffnet, in diesem Fall ist Chromium die beste Wahl (siehe Abschnitt 4.4.1.2). Ist der Browser bereit, wird die Liste von Hochschulen iteriert, dabei wird geprüft ob in den letzten 30 Tagen bereits ein Screenshot erstellt wurde. Dadurch wird sichergestellt, dass Skript beliebig oft gestartet werden kann, aber für jede Hochschule nur einen Screenshot im Monat anfertigt. Um einen möglichst guten Screenshot zu erstellen, müssen mehrere Aktionen durchgeführt

werden. Der Browser muss langsam bis zum Ende und dann wieder bis zum Anfang der Seite scrollen. Dadurch wird sichergestellt dass alle Elemente geladen wurden. Danach kann der erste Screenshot der gesamten Seite erstellt werden. Um später eine Liste aller Screenshots darstellen zu können werden auch Thumbnails benötigt. Die der vollen Seite sind zu groß, weshalb ein weiterer Screenshot vom oberen Bereich erstellt wird. Davor werden die Scrollbalken mit *overflow = hidden* versteckt.

Die beiden Screenshots bekommen eine ID und werden lokal gespeichert. Gleichzeitig wird die ID zusammen mit einigen Metadaten in der Datenbank abgespeichert. Dadurch kann eine Übersicht von Screenshots erstellt werden, ohne dass das Dateisystem durchsucht werden muss.

4.4.1.2 Lösung für Cookie-Popups

Ein großes Problem waren die Cookie-Popups, die auf den meisten Seiten angezeigt wurden. Sie blockieren einen Großteil der Screenshots und verdecken in vielen Fällen den Rest der Website.

Die angewandte Lösung besteht aus einer Browsererweiterung und verschiedenen Einstellungen in Chromium. Dazu muss Playwright den Browser mit *launchPersistentContext*, also einem beständigen Kontext, öffnen. Dabei muss ein lokaler Ordner für die Benutzereinstellungen sowie die Browsererweiterung selbst zur Verfügung gestellt werden. Durch das Setzen der Startparameter *--disable-extensions-except* und *--load-extension* auf den Dateipfad der Erweiterung, soll sichergestellt werden, dass nur diese aktiv ist. Die Erweiterung selbst ist die Chromium Version von „I don't care about Cookies“²⁹.

4.5 Server

Die erste Live-Version wurde auf einem Ubuntu-Server mit der Version 22.04 LTS getestet. Zu den Abhängigkeiten gehören insbesondere die LTS-Version von Node.js und eine PostgreSQL-Datenbank der Version 15. In der ersten Testversion dieses Projekts war der Server terminalbasiert, ohne grafische Oberfläche. Da Playwright jedoch einen Browser öffnen und Webseiten rendern muss, wird in der finalen Version eine grafische Oberfläche benötigt.

4.5.1.1 Next.js Anwendung im Hintergrund

Um die Anwendung im Hintergrund auszuführen, wird das Tool PM2³⁰ benötigt. Es startet und verwaltet die Next.js Anwendung im Hintergrund.

²⁹ Daniel Kladnik, *I don't care about cookies* (Daniel Kladnik @ kiboke studio), zuletzt geprüft am 16.06.2023, <https://www.i-dont-care-about-cookies.eu/>.

³⁰ Keymetrics, *PM2* (Keymetrics), zuletzt geprüft am 26.06.2023, <https://pm2.keymetrics.io/>.

4.5.1.2 Webserver

Um die von PM2 gestartete Anwendung nach außen sichtbar zu machen, wird ein Webserver benötigt. Eine mögliche Option war NGINX³¹, ein open-source Webserver, Reverse Proxy und E-Mail-Proxy.

Durch eine NGINX-Konfigurationsdatei wird ein Serverblock eingerichtet, der auf Port 80 auf Anfragen wartet. Alle Anfragen die im Stammverzeichnis („/“) eingehen, werden an den Backend-Server von Next.js weitergeleitet. Darüber hinaus werden mit Let's Encrypt³² Zertifikate erstellt und bereitgestellt, um eine sichere SSL-Version der Seite anbieten zu können.

4.5.1.3 CronJobs

CronJobs werden benötigt, um die verschiedenen Skripte automatisch ausführen zu lassen. Cron ermöglicht es die Ausführung von Aufgaben oder Skripten in bestimmten Intervallen oder zu bestimmten Zeiten zu planen und zu automatisieren.

³¹ F5, *NGINX* (F5), zuletzt geprüft am 26.06.2023, <https://www.nginx.com/>.

³² Internet Security Research Group, *Let's Encrypt* (Internet Security Research Group), zuletzt geprüft am 26.06.2023, <https://letsencrypt.org/>.

5 Analyse und Tests

5.1 Performance, SEO, Barrierefreiheit, Best Practices

Die Leistungstests wurden mit Google Lighthouse auf der Live-Version der Plattform durchgeführt. Die Browsererweiterung von Lighthouse bietet drei verschiedene Modi: Navigation, Zeitspanne, Momentaufnahme. Der Navigationsmodus erfolgte auf mehreren ausgewählten Seiten und Unterseiten, wobei alle Metriken berücksichtigt wurden. Der Zeitspannenmodus misst Layoutverschiebungen, die Performance von JavaScript und Interaktionen über einen bestimmten Zeitraum. Er liefert keine Gesamtperformance-Ergebnisse und kann keine momentbasierten Metriken (z. B. LCP) analysieren. Die Momentaufnahme wurde stichprobenartig an komplexen Formularen wie z. B. der Anzeigenerstellung durchgeführt. Sie soll Probleme mit der Barrierefreiheit des Formulars aufzeigen.

5.1.1.1 Navigationsmodus

Im Navigationsmodus wird für jede der vier Kategorien eine Bewertung von 0 bis 100 vergeben. Bewertungen unter 90 sind verbesserungswürdig. Alle Bewertungen beziehen sich auf die Desktop-Version der Plattform, da diese die höchste Priorität hatte.

Tabelle 2: Ergebnisse der Lighthouse-Tests im Navigationsmodus

Seite	Performance /100	Barrierefreiheit /100	Best Practices /100	SEO /100
/ (Index)	99	90	100	91
/ locations	93	96	100	100
/ categories	100	91	100	92
/ institutions	93	96	100	100
/ institution / germany / freie-universitaet-berlin / social-media	100	98	100	100
/ institution / germany / freie-universitaet-berlin / subjects	90	96	100	100
/ location / germany	93	96	100	100

Keine der getesteten Seiten hatte größere Probleme, alle Bewertungen lagen zwischen 90 und 100 Punkten. Dennoch können an vielen Stellen kleinere Verbesserungen vorgenommen werden. Insbesondere die Anzahl der Objekte im DOM ist mit durchschnittlich über 1400 Elementen zu hoch.

5.1.1.2 Zeitspannenmodus

Während des Tests wurden folgende Seiten in Reihenfolge angesteuert:

- Index
- / categories
- / institutions
- / analysis
- / news /
- / locations
- / location / germany
- / location / germany / rheinland-pfalz
- / location / germany / rheinland-pfalz / kaiserslautern
- / location / germany / rheinland-pfalz / kaiserslautern / hochschule-kaiserslautern-university-of-applied-sciences
- / location / germany / rheinland-pfalz / kaiserslautern / hochschule-kaiserslautern-university-of-applied-sciences / subjects
- / location / germany / rheinland-pfalz / kaiserslautern / hochschule-kaiserslautern-university-of-applied-sciences / social-media
- / location / germany / rheinland-pfalz / kaiserslautern / hochschule-kaiserslautern-university-of-applied-sciences / social-media / twitter

Die Ergebnisse werden als Anzahl der bestandenen Audits angegeben. In der Kategorie Performance wurden 21 von 24 Audits bestanden. Das Hauptproblem war die Anzahl der DOM-Elemente, neben statischen Bildern, die noch nicht optimiert waren. In der Kategorie Best Practices wurden 8 von 8 Audits bestanden.

5.1.1.3 Momentaufnahme

Die Momentaufnahme zeigt die Anzahl der erfolgreichen Audits in den vier Kategorien. Die vierte Kategorie, SEO, ist für die getestete Unterseite nicht relevant, da das Formular nur nach Anmeldung verfügbar ist. Daher kann und sollte es nicht von Suchmaschinen gefunden werden. Die getestete URL war „*/account/create-ad*“ und wurde in allen verschiedenen Zuständen getestet.

In der Kategorie Performance wurden drei von vier Audits bestanden. Ein Problem betraf das Platzhalterbild in der Vorschau der Anzeige, da es größer als die dargestellte Größe ist, was zu längeren Ladezeiten als nötig führt. Im Bereich Barrierefreiheit wurden 19 von

21 Audits bestanden. Bei den gefundenen Fehlern handelte es sich um falsche Attribute für ARIA-Felder. In der letzten relevanten Kategorie „Best Practices“ wurden alle vier Audits erfolgreich bestanden.

6 Diskussion

6.1 Herausforderungen

Die Entwicklung der Web-Plattform bringt eine Reihe von Herausforderungen mit sich, die im Laufe des Projekts bewältigt werden mussten. Diese Herausforderungen betreffen verschiedene Aspekte, einschließlich der Datenerfassung, der Datenverarbeitung, der Entwicklung von Algorithmen und der Skalierbarkeit des Systems. In diesem Abschnitt werden einige der wichtigsten wissenschaftlichen Herausforderungen diskutiert, die im Laufe des Projekts aufgetreten sind.

6.1.1 Datenerfassung und -integration:

Eine große Herausforderung bestand darin, umfassende und aktuelle Informationen über die Universitäten, Studiengängen oder auch Städten zu erhalten. Die Sammlung von Daten aus verschiedenen Quellen erforderte sorgfältige Extraktions- und Integrationsprozesse. Darüber hinaus war es schwierig die Genauigkeit und Konsistenz der gesammelten Daten zu gewährleisten und gleichzeitig mit Inkonsistenzen zwischen den verschiedenen Datenquellen umzugehen.

6.1.2 Web Scraping und Datenqualität:

Web Scraping, eine Technik zur Extraktion von Daten aus Websites, stellte eine eigene Reihe von Herausforderungen dar. Die Websites der Universitäten weisen häufig unterschiedliche Strukturen und Formate auf, was die Entwicklung eigener Scraping-Skripte erforderlich machte. Der Umgang mit dynamischen Websites, die clientseitiges Rendering oder Anti-Scraping-Maßnahmen verwenden, brachte zusätzliche Komplexität mit sich. Die Sicherstellung der Qualität und Zuverlässigkeit der gescrapten Daten, einschließlich der Behandlung fehlender oder unvollständiger Informationen, stellte eine weitere wissenschaftliche Herausforderung dar.

6.1.3 SEO-Maßnahmen

Die Optimierung der Website für die Sichtbarkeit und das Ranking in Suchmaschinen ist von entscheidender Bedeutung. Die Umsetzung von SEO-Techniken wie Meta-Tag-Optimierung, Optimierung der URL-Struktur und Content-Optimierung stellt eine Herausforderung dar. Die Balance zwischen User Experience und technischen SEO-Anforderungen erfordert eine sorgfältige Analyse und Umsetzung der geplanten Features.

6.1.4 Backend für Hochschulen

Eine weitere Kernfunktionalität war ein Backend-System, das es den Hochschulen ermöglicht, ihre Anzeigen und Artikel zu erstellen und zu verwalten. Bei der Entwicklung muss auf Benutzerfreundlichkeit, Datenspeicherung und effizientes Content Management geachtet werden. Die Gewährleistung von Datenintegrität, Sicherheit und Performance bei gleichzeitiger Möglichkeit für die Kunden, Anzeigen und Artikel zu erstellen und zu veröffentlichen, erhöht die Komplexität des Backend-Entwicklungsprozesses.

6.1.5 Social-Media Datenerhebung und -analyse

Die Entwicklung der notwendigen Infrastruktur für die Interaktion mit Social-Media-APIs wie Twitter und Youtube erwies sich als komplexes Problem. Der Aspekt der Automatisierung erfordert ein Skript, das mit allen Eventualitäten, Datenstrukturen und anderen Problemen umgehen kann. Sobald die Daten gesammelt sind, werden Algorithmen benötigt, um die Hochschulprofile in den sozialen Medien zu analysieren und zu bewerten.

6.1.6 Online-Marketing Datenerhebung und -analyse

Die Verwendung von Google Lighthouse für die Analyse von Hochschulwebsites, ihre Bewertung anhand der Lighthouse-Kennzahlen und die Darstellung der Ergebnisse, stellt eine Reihe von Herausforderungen dar. Die Integration des Skripts in die Google Lighthouse API und die Einrichtung einer nahtlosen Interaktion mit dem Tool kann eine Herausforderung darstellen. Die Automatisierung des Analyseprozesses für eine große Anzahl von Universitätswebsites stellt eine Herausforderung in Bezug auf Skalierbarkeit und Effizienz dar. Das Skript muss in der Lage sein, ein hohes Volumen an Anfragen zur Analyse von Websites zu bewältigen.

6.2 Beschränkungen

Die genaue SEO-Performance des Projekts ist noch unklar, da in den Live-Tests noch kein Analysetool integriert war (Google Analytics war aus Datenschutzgründen nicht möglich). Die Plattform selbst ist an einigen Stellen noch unfertig, z.B. ist der Payment-Provider noch nicht integriert, studentische Rezensionen sind noch nicht möglich, es ist noch keine „Job-API“ integriert und diverse Backend-Funktionen fehlen. Dazu gehören die automatische Prüfung hochgeladener Bilder auf anstößige Inhalte und der Versand von Bestätigungsmails bei der Registrierung.

6.3 Verbesserung der Social-Media-Analyse

In diesem Abschnitt werden die Verbesserungen und Unterschiede zu den beiden Forschungsarbeiten "Market, Society, and Education – University Marketing" (Kloppenburg, 2012) und "Evaluation und Implementierung von Social-Media-Ratings von Hochschulen für ein Webportal" (Schuler, 2022) im Detail definiert. Die Ausführungen beziehen sich auf den Teil der Social-Media-Bewertung, da dieser in beiden Arbeiten im Fokus stand.

In der Forschungsarbeit "Evaluation und Implementierung von Social-Media-Ratings von Hochschulen für ein Webportal" wurde der Versuch unternommen, bestimmten Metriken Gewichtungspunkte zuzuordnen. Die Punkte wurden relativ zueinander vergeben: Ein Youtube-Video war beispielsweise 100 Punkte wert, ein Tweet 15 Punkte. Gründe dafür waren unter anderem die Menge des Inhalts in einem Video und die Aufmerksamkeit, die es erfordert. Dieses Konzept wurde zugunsten von dem neuen Algorithmus verworfen. Die Länderstatistiken wurden übernommen und erweitert, während die Crawler als Grundlage genommen und erheblich verbessert wurden.

Die Forschungsarbeit "Market, Society, and Education – University Marketing" hat eigene mathematische Formeln zur Bewertung von Social-Media-Profilen entwickelt. Diese dienten als Vorbild für die Entwicklung der Bewertungsalgorithmen in diesem Projekt. Einige Metriken, wie z.B. Tweet Impressionen, sind erst seit kurzem verfügbar und wurden in die Algorithmen integriert.

Es wurden auch einige neue Metriken hinzugefügt, wie z. B. die durchschnittliche Anzahl von Videos pro Monat oder die durchschnittliche Anzahl von Tweets pro Monat. Außerdem werden Listen mit den am häufigsten verwendeten Video-Tags oder Wörtern in Tweets erstellt.

7 Fazit

7.1 Erfolge

Am Ende des Projekts stand eine funktionsfähige Online-Plattform mit allen notwendigen Backend-Funktionen zur Verfügung, um vollständig automatisiert werden zu können. Die Basis der Plattform, die Darstellung und Vernetzung von Hochschul- und Studiengangsdaten, ist vollständig fertiggestellt. Verschiedene Kernfunktionen wie Internationalisierung, die Suchfunktion mit Filtern oder auch Paginierung sind implementiert. Die Registrierung ist noch auf Hochschulen beschränkt, die Funktionalität zur Erstellung von Anzeigen und Artikeln ist gegeben und es fehlt nur noch ein Zahlungsanbieter. Die erstellten Anzeigen und Artikel werden an bestimmten Stellen und zu bestimmten Zeiten auf der Plattform angezeigt. Die Performance- und SEO-Bewertung der Plattform durch Google Lighthouse ergibt ausschließlich hohe Bewertungen von über 90%. Dies hat sich auch in den Live-Tests bestätigt, die Benutzeroberfläche reagiert schnell, die Ladezeiten sind minimal. Die Anbindung der Social-Media-APIs im Backend wurde im Vergleich zum Vorgängerprojekt³³ verbessert. Das Skript liefert deutlich verbesserte Ergebnisse. Fehler wurden minimiert und es werden mehr Daten pro Social-Media-Profil gesammelt. Insbesondere der Algorithmus zur Bewertung der Social-Media-Profile liefert deutlich bessere Ergebnisse (siehe 6.3 Verbesserung der Social-Media-Analyse) als die Vorgänger „Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal“ - Schuler (2022) und „Market, Society, and Education – University Marketing“ - Kloppenburg (2012).

7.2 Ausblick

7.2.1 Fertigstellung der Basisplattform

Vor dem Start der Plattform müssen zwei wichtige Funktionen implementiert werden: das Zahlungssystem und der Mailserver für den Versand der Bestätigungsmails. Auch die Einrichtung des Servers ist kein triviales Projekt, neben der Next.js-Anwendung selbst und der Datenbank müssen alle Skripte mit CronJobs verwaltet werden. Für die Anbindung an die Meta APIs für Facebook und Instagram werden lediglich die entsprechenden Zugangsdaten benötigt. Der Gesamtanalyse der Social-Media- und Online-Marketing-Daten fehlt es an weiteren Grafiken, Inhalten und Diagrammen. Auch

³³ Schuler, „Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal.“

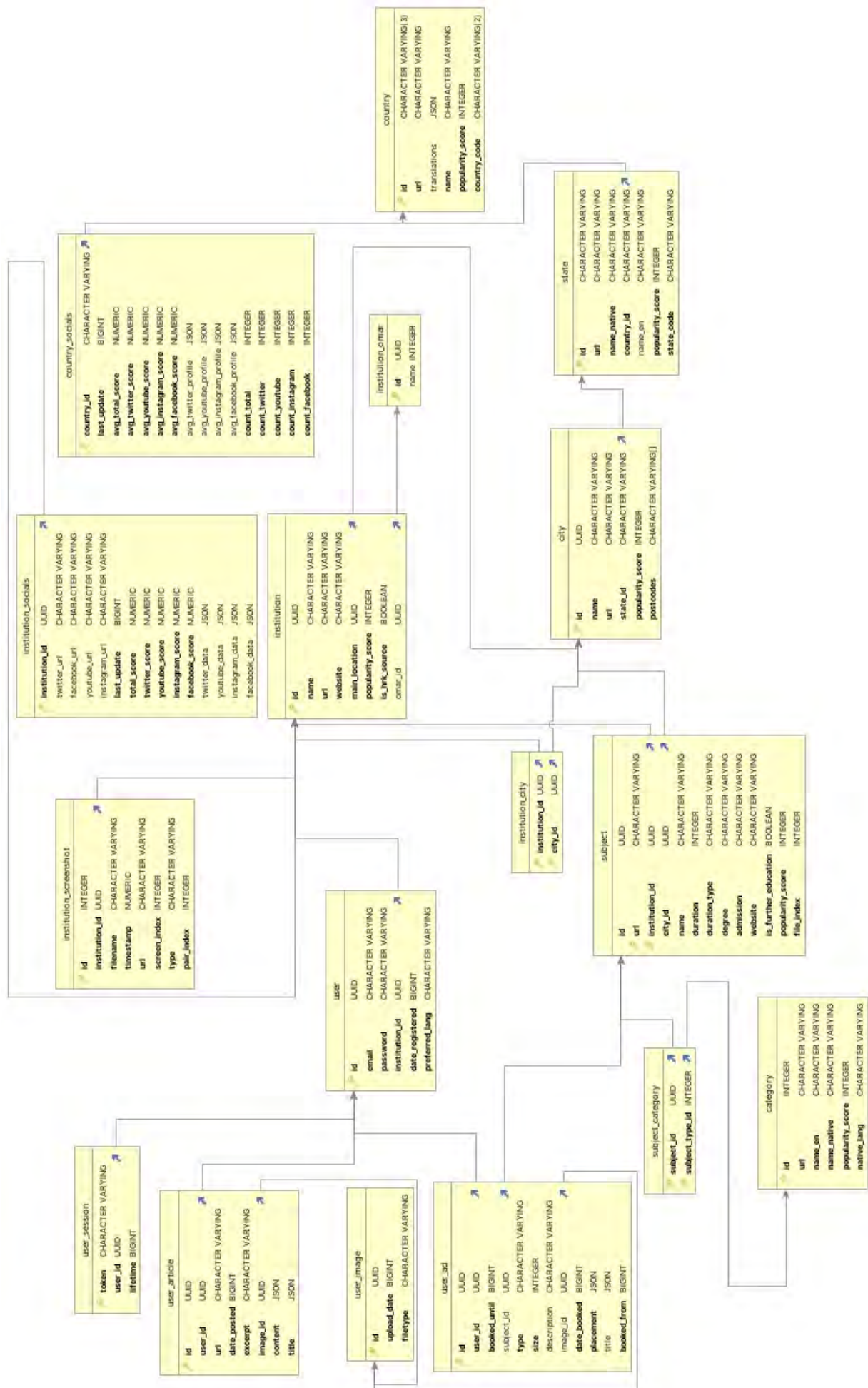
das volle Potential der Wikipedia API ist noch nicht ausgeschöpft, es könnten noch mehr Texte und Bilder zu jeder Hochschule gesammelt und dargestellt werden.

7.2.2 Zukunft

Während der Entwicklung des Projekts wurden zahlreiche Ideen gesammelt, die die Plattform auf unterschiedliche Weise bereichern würden, für deren Umsetzung jedoch die Zeit fehlte. Die einflussreichsten in einer Liste:

- Stipendien in Europa auflisten und mit geeigneten Hochschulen und Studiengängen verknüpfen.
- Eine oder mehrere APIs verwenden, um Studiengänge mit potenziellen Arbeitsplätzen zu verknüpfen. Daraus könnten auch verschiedene Werte abgeleitet werden, wie z.B. das Durchschnittsgehalt.
- Google-Anzeigen integrieren.
- Ermöglichung der Suche in natürlicher Sprache über einen LLM.
- Durchführung einer Sentiment-Analyse für verschiedene Social-Media-Posts von Universitäten. Ziel ist es, den emotionalen Ton oder die Stimmung zu ermitteln. Dies beinhaltet die Analyse und Kategorisierung der Sprache, die in Posts und Kommentaren verwendet wird, um festzustellen, ob die vermittelte Stimmung positiv, negativ oder neutral ist.
- Durchführung einer Content-Analyse für Social-Media-Posts von Hochschulen. Sie zielt darauf ab, Muster, Themen und Trends innerhalb der Inhalte zu identifizieren, um Einblicke in Themen, Stimmungen, Nutzerverhalten und andere relevante Aspekte des Social-Media-Diskurses zu erhalten. Dabei kann ebenfalls ein LLM helfen, bspw. durch die Aufforderung, einen Post zu analysieren und bestimmten vordefinierten Kategorien zuzuordnen.
- Die von den Hochschulen gebuchten Anzeigen können an bestimmten, vordefinierten Stellen in einer Liste angezeigt werden. Dadurch wird die Sichtbarkeit der Anzeigen für die Nutzer erhöht.

Anhang A: Datenbankmodell



Anhang B: Datei- und Ordnerverzeichnis

Nachfolgend sind die Ordner und Dateien aufgelistet, die sich auf dem Datenträger befinden.

Tabelle 3: Ordner und Dateien auf dem Datenträger

Name / Pfad	Zweck	Format
Source Files / uni-index-mantine	Implementierung der eigentlichen Web-Plattform als Next.js Applikation (siehe 4.1 Next.js – Das Erstellen der Webseite)	Projektordner
Source Files / uni-index-crawler	Implementierung des Hauptcrawlers (siehe 4.2 Hauptcrawler)	Projektordner
Source Files / uni-index-lighthouse	Implementierung des Lighthouse-Skripts (siehe 4.3 Lighthouse-Skript & Online-Marketing-Analyse)	Projektordner
Source Files / uni-index-screenshots	Implementierung des Screenshot-Skripts (siehe 4.4 Screenshot-Skript)	Projektordner
Tabellen / social-media-score-algorithm	Algorithmus für das Bewerten der sozialen Medien in Tabellenform, detaillierte Erklärung mit Beispielen	Excel-Tabelle
Tabellen / data-collection-review	Analyse der potenziellen Datenquellen und APIs	Excel-Tabelle
Literatur	Ausgewählte Literatur die in diesem Projekt verwendet wurde	Ordner
Datenbank	SQL-Skripte aller Datenbanktabellen (siehe 3.7.1 Datenbanktabellen)	Ordner

Anhang C: Payment Gateway Analyse

Die folgende Tabelle zeigt einige der wichtigsten Eigenschaften, die sich bei der Analyse aller Anbieter ergeben haben.

Tabelle 4: Payment Gateway Analyse

Name	Monatliche Gebühr	Transaktionsgebühr	Einbindung / Development
Stripe	Keine	1,5% + 0,25€	Entwicklerfreundlich, zahlreiche Anpassungen und Integrationen, beste React-Unterstützung als Standard
Square	Keine	1,4% + 0,25€	Gut dokumentiert, Unterstützung von React
Braintree	Keine	1,9% + 0,30€	Keine React-Unterstützung, stattdessen wird eine RESTful API verwendet
Helcim	Keine	Ab 2,40% + 0,30€	Javascript-Bibliothek und RESTful API
Paddle	Keine	5% + 0,50€	Javascript-Bibliothek und RESTful API
Stax	Ab 99 USD	Keine	RESTful API
Merchant One	Vorhanden, undeutliche Angaben	Keine Angaben	API

Anhang D: Screenshots

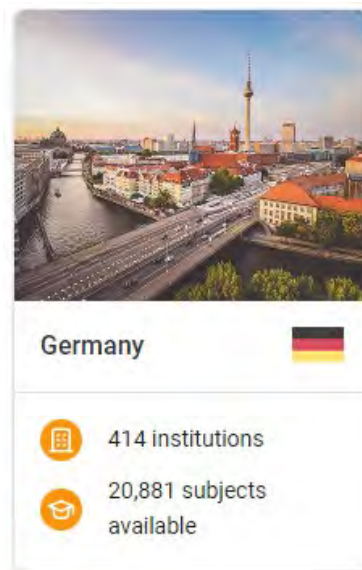


Abbildung 9: Objektkarte eines Landes

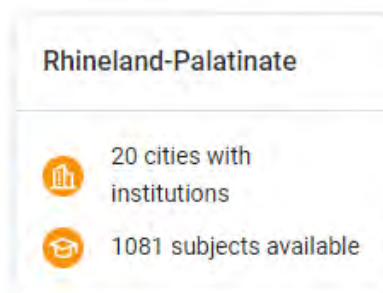


Abbildung 10: Objektkarte eines Bundeslandes

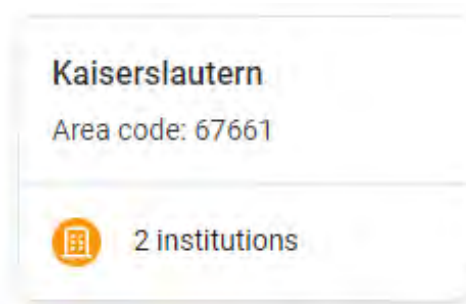


Abbildung 11: Objektkarte einer Stadt

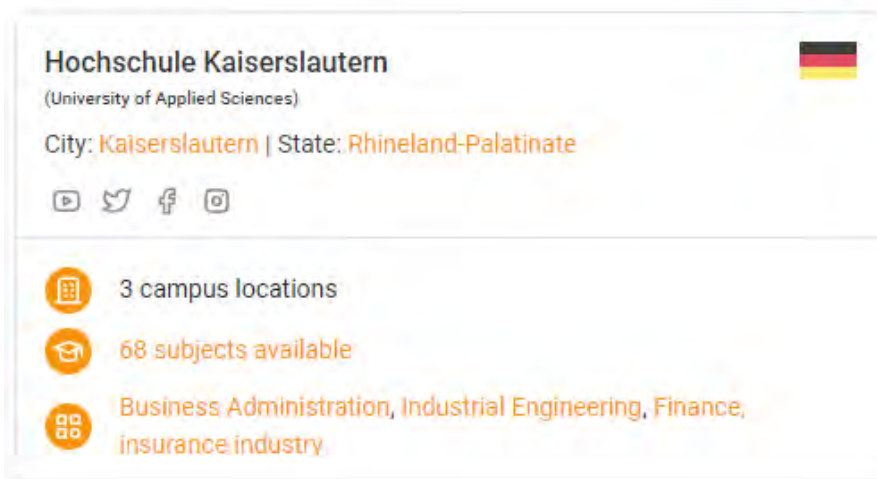


Abbildung 12: Objektkarte einer Hochschule

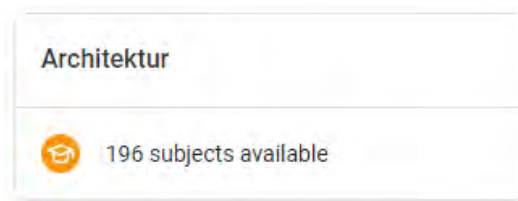


Abbildung 13: Objektkarte einer Studiengangsrichtung

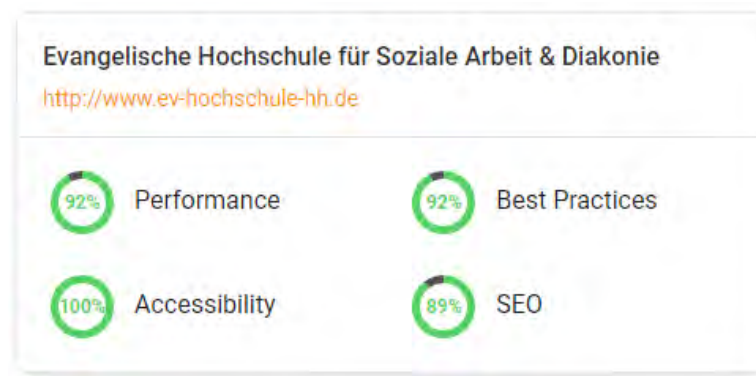


Abbildung 14: Objektkarte einer Online-Marketing-Analyse

Online Marketing Analysis

We analysed the website of Fachhochschule Dortmund based on Google Lighthouse.

Last update: 13/02/2023

Website analysed: <http://www.fh-dortmund.de>

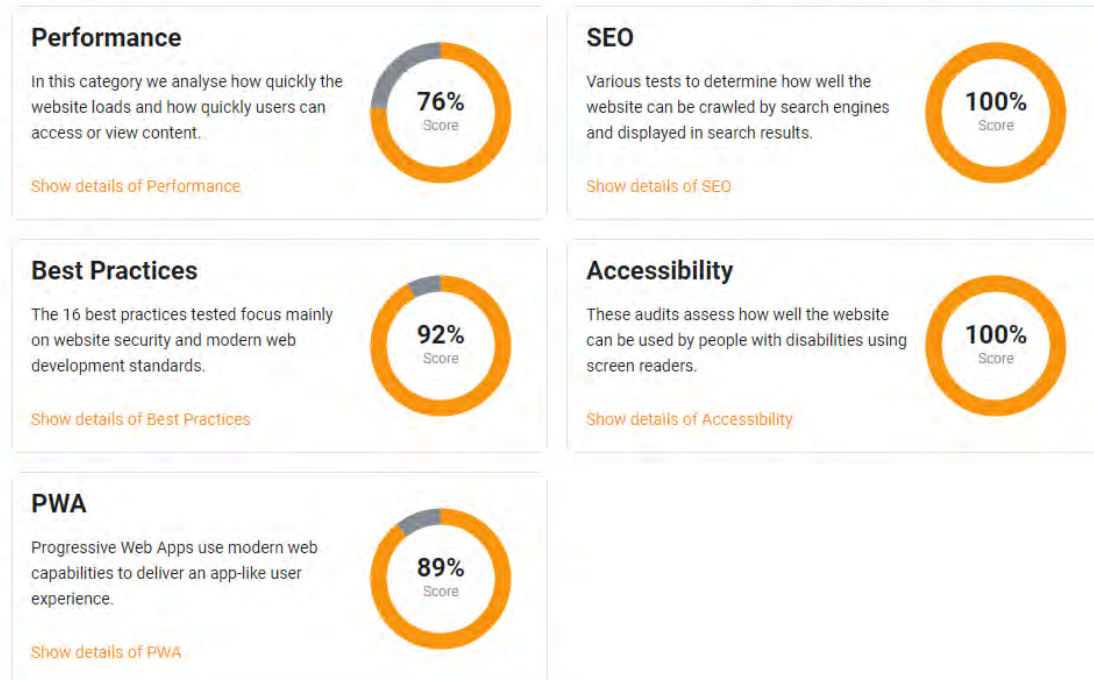


Abbildung 15: Online-Marketing-Übersicht im Profil der FH-Dortmund

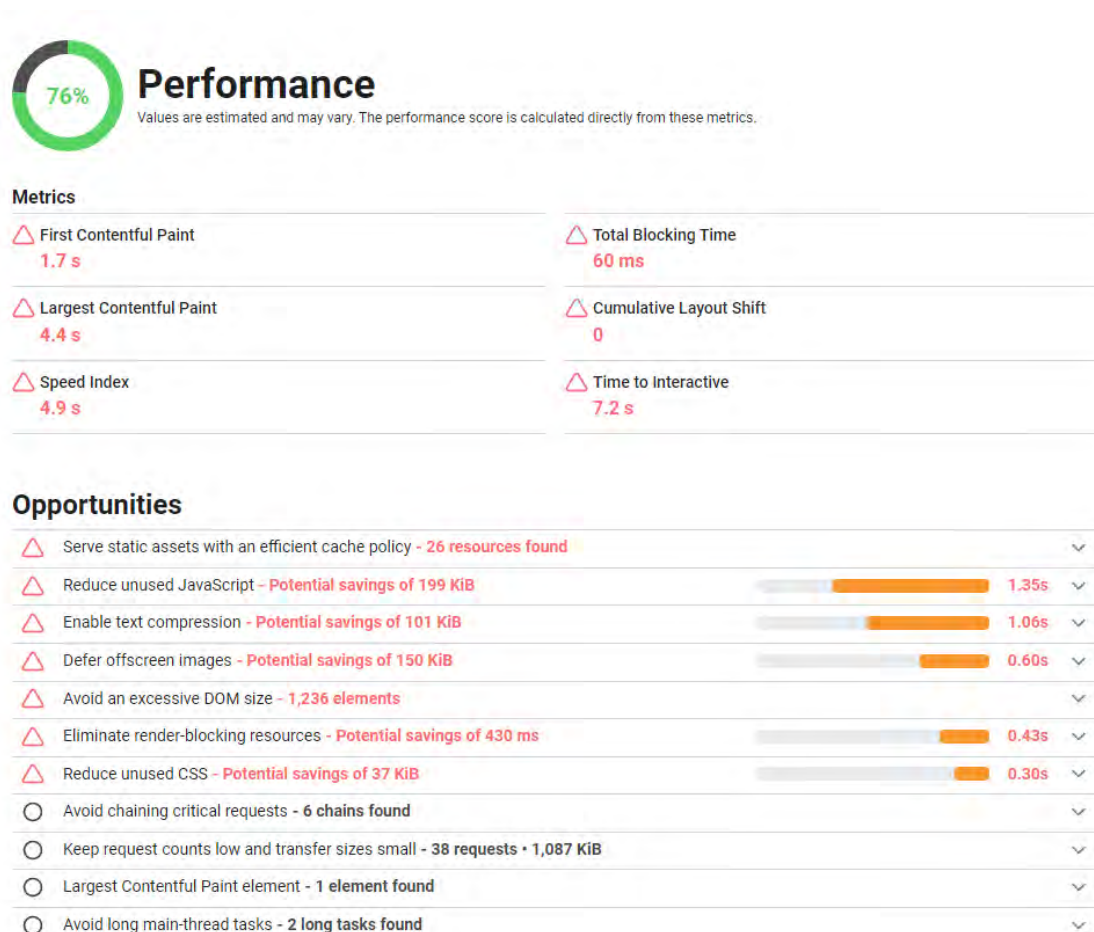


Abbildung 16: Detailansicht „Performance“ der Online-Marketing-Analyse

Avoid chaining critical requests - 6 chains found

Keep request counts low and transfer sizes small - 38 requests • 1,087 KiB

Largest Contentful Paint element - 1 element found

Avoid long main-thread tasks - 2 long tasks found

Lists the longest tasks on the main thread, useful for identifying worst contributors to input delay. [Learn more](#)

URL	Start Time	Duration
https://analytics.fh-dortmund.de/piwik.js	7470.61 ms	164.00 ms
https://www.fh-dortmund.de/	1318.34 ms	56.00 ms

Abbildung 17: Angabe der genauen Problemstellen in der Online-Marketing-Analyse

Glossar

JSON (auch: JSON-Objekte): JSON ist ein Datenformat für den Datenaustausch zwischen mehreren Anwendungen. Es ist unabhängig von Programmiersprachen und bringt Struktur in einen Datensatz. Mögliche Werte sind Nullwerte, Zahlen, Zeichenketten, Boolesche Werte, Arrays und Objekte.³⁴

Scrapen / scraper: Das (Web-) Scrapen ist ein Verfahren, um Daten aus Websites zu extrahieren. Der Webscraper ist dabei das Werkzeug, meistens in Form von Software oder einer Programmbibliothek. Dabei ist der Prozess der Datengewinnung in der Regel automatisiert. Es werden bestimmte Daten gesammelt und in eine lokale Datenbank oder in eine lokale Datei gespeichert, um sie später weiter verarbeiten zu können.

Kryptologische Hashfunktion: Eine kryptologische Hashfunktion ist ein mathematischer Algorithmus, der aus einer Eingabe (oder Nachricht) eine Zeichenkette mit fester Größe erzeugt, die als Hash-Wert oder Hash-Code bezeichnet wird. Sie ist so konzipiert, dass es rechnerisch nicht möglich ist, die ursprüngliche Eingabe aus dem Hash-Wert abzuleiten.

Session: Eine Session (dt. Sitzung) bezeichnet die bestehende Verbindung zwischen einem Client und einem Server. Der Login ist dabei der Anfang und Logout das Ende der Session. Für längere Sessions werden Cookies genutzt, da diese auf dem Client gespeichert werden. Die eindeutige Session-ID ist dabei der einzige Wert, der gespeichert werden muss und besteht aus einer Zeichenkette.

Large Language Model (LLM): Ein LLM ist ein leistungsfähiges Modell der künstlichen Intelligenz, das in der Lage ist, menschliche Sprache zu verstehen und zu erzeugen. LLMs werden mit großen Mengen von Textdaten trainiert und sind in der Lage, schlüssige und kontextuell relevante Antworten auf eine Eingabeaufforderung zu erzeugen. Sie können für verschiedene Aufgaben der natürlichen Sprachverarbeitung eingesetzt werden, einschließlich Chatbots, Sprachübersetzung, Textgenerierung, Beantwortung von Fragen und vieles mehr.

³⁴ International Organization for Standardization, *Information technology — The JSON data interchange syntax: ISO/IEC 21778:2017* (2017), zuletzt geprüft am 14.06.2023, <https://www.iso.org/standard/71616.html>.

Literatur- und Quellenverzeichnis

- Adam Peruta, und Alison B. Shields. „Marketing your university on social media: a content analysis of Facebook post types and formats.” *Journal of Marketing for Higher Education*, 2018. <https://www.semanticscholar.org/paper/Marketing-your-university-on-social-media%3A-a-of-and-Peruta-Shields/a41bdbb031e51988bdb8f717f4e045ac7cff616d>.
- Aral Roca Gomez. *next-translate*. Zuletzt geprüft am 07.06.2023. <https://github.com/aralroca/next-translate>.
- Bélanger, Charles H., Suchita Bali, und Bernard Longden. „How Canadian universities use social media to brand themselves.” *Tertiary Education and Management* 20, Nr. 1 (2014): 14–29. <https://doi.org/10.1080/13583883.2013.852237>.
- Constantinides, Efthymios, und Marc C. Zinck Stagno. „Higher Education Marketing: A Study on the Impact of Social Media on Study Selection and University Choice.” *International Journal of Technology and Educational Marketing* 2, Nr. 1 (2012): 41–58. <https://doi.org/10.4018/ijtem.2012010104>.
- Daniel Kladnik. *I don't care about cookies*. Daniel Kladnik @ kiboke studio. Zuletzt geprüft am 16.06.2023. <https://www.i-dont-care-about-cookies.eu/>.
- Erudera. „Germany International Student Statistics.” Zuletzt geprüft am 06.06.2023. <https://erudera.com/statistics/germany/germany-international-student-statistics/>.
- Ethnologue. „What are the top 200 most spoken languages?.” Zuletzt geprüft am 07.06.2023. <https://www.ethnologue.com/insights/ethnologue200/>.
- F5. *NGINX*. F5. Zuletzt geprüft am 26.06.2023. <https://www.nginx.com/>.
- Google LLC. *Lighthouse*. Google LLC. Zuletzt geprüft am 30.05.2023. <https://developer.chrome.com/docs/lighthouse/overview/>.
- Google LLC. „Pagination, incremental page loading, and their impact on Google Search.” Zuletzt geprüft am 23.06.2023. <https://developers.google.com/search/docs/specialty/ecommerce/pagination-and-incremental-page-loading>.
- Google LLC. „Sitemaps.” Zuletzt geprüft am 14.06.2023. <https://developers.google.com/search/docs/crawling-indexing/sitemaps/overview>.
- International Organization for Standardization. *Information technology — The JSON data interchange syntax: ISO/IEC 21778:2017.*, 2017. Zuletzt geprüft am 14.06.2023. <https://www.iso.org/standard/71616.html>.
- Internet Security Research Group. *Let's Encrypt*. Internet Security Research Group. Zuletzt geprüft am 26.06.2023. <https://letsencrypt.org/>.

- Keymetrics. *PM2*. Keymetrics. Zuletzt geprüft am 26.06.2023.
<https://pm2.keymetrics.io/>.
- Kloppenburg, Michael. „Market, Society, and Education - University Marketing.”
Dissertation, Informatik und Mikrosystemtechnik, Hochschule Kaiserslautern, 2012.
- Merten, Wolfgang, und Thorsten Knoll. *Handbuch Wissenschaftsmarketing: Konzepte, Instrumente, Praxisbeispiele* / Wolfgang Merten, Thorsten Knoll. 1. Aufl. 2019.
Wiesbaden: Springer Gabler, 2019. <https://doi.org/10.1007/978-3-658-25353-0>.
- Meta Open Source. *React*. Meta. Zuletzt geprüft am 02.06.2023. <https://react.dev/>.
- Microsoft. *Playwright*. Microsoft. Zuletzt geprüft am 15.06.2023.
<https://playwright.dev/>.
- Microsoft. *TypeScript*. Microsoft. Zuletzt geprüft am 04.06.2023.
<https://www.typescriptlang.org/>.
- Motta, Joana, und Maria Barbosa. „Social Media as a Marketing Tool for European and North American Universities and Colleges.” *Journal of Intercultural Management* 10, Nr. 3 (2018): 125–54. <https://doi.org/10.2478/joim-2018-0020>.
- Opendatasoft. „Postleitzahlen - Germany.” Zuletzt geprüft am 30.05.2023.
<https://public.opendatasoft.com/explore/dataset/georef-germany-postleitzahl/table/>.
- OpenJS Foundation. *Node.js*. OpenJS Foundation. Zuletzt geprüft am 04.06.2023.
<https://nodejs.org>.
- Paul-Louis Hery. *node-twitter-api-v2*. Zuletzt geprüft am 20.06.2023.
<https://www.npmjs.com/package/twitter-api-v2>.
- Pringle, James, und Samantha Fritz. „The university brand and social media: using data analytics to assess brand authenticity.” *Journal of Marketing for Higher Education* 29, Nr. 1 (2019): 19–44. <https://doi.org/10.1080/08841241.2018.1486345>.
- Prisma Data, Inc. *Prisma*. Prisma Data, Inc. Zuletzt geprüft am 04.06.2023.
<https://www.prisma.io/>.
- Schuler, Sebastian. „Evaluation und Implementierung von Social Media Ratings von Hochschulen für ein Webportal.” Bericht Praxisprojekt, 2022.
- SheetJS LLC. *SheetJS*. SheetJS LLC. Zuletzt geprüft am 30.05.2023.
<https://sheetjs.com/>.
- The PostgreSQL Global Development Group. *PostgreSQL*. The PostgreSQL Global Development Group. Zuletzt geprüft am 04.06.2023. <https://www.postgresql.org/>.
- Überdosis. *Tiptap*. Überdosis. Zuletzt geprüft am 14.06.2023. <https://tiptap.dev/>.
- Vercel. *Next.js*. Vercel. Zuletzt geprüft am 02.06.2023. <https://nextjs.org/>.

Vishnu Sankar. *next-sitemap*. Vishnu Sankar. Zuletzt geprüft am 14.06.2023.

<https://www.npmjs.com/package/next-sitemap>.

Vitaly Rtishchev. *Mantine*. mantinedev. Zuletzt geprüft am 04.06.2023.

<https://mantine.dev/>.

Stichwortverzeichnis

CronJobs	53	Node.js	16
Google Lighthouse	18	Playwright	20
Internationalisierung	21	PostgreSQL	19
JSON	69	Prisma	18, 22
Kryptologische Hashfunktion	32, 33, 69	React	17
Large Language Model (LLM)	61, 69	Scrapen / Scraper	69
Mantine	19	Session	33, 69
Next	17	TypeScript	19
Next-translate	20		