Informatik
Hauptcampus

H O C H
S C H U L E
T R I E R

Implementierung einer Enterprise Search Engine für das Dietrich Online Projekt

Implementation of an enterprise search engine for the Dietrich Online project

Florian Reitz

Bachelor-Abschlussarbeit

Betreuer: Professor Doktor Christoph Schmitz

Trier, den 15.10.2019 15.3.2020

Vorwort

Ein Vorwort ist nicht unbedingt nötig. Falls Sie ein Vorwort schreiben, so ist dies der Platz, um z.B. die Firma vorzustellen, in der diese Arbeit entstanden ist, oder einigen Leuten zu danken, die in irgendeiner Form positiv zur Entstehung dieser Arbeit beigetragen haben. Auf keinen Fall sollten Sie im Vorwort die Aufgabenstellung näher erläutern oder vertieft auf technische Sachverhalte eingehen.

Kurzfassung

In der Kurzfassung soll in kurzer und prägnanter Weise der wesentliche Inhalt der Arbeit beschrieben werden. Dazu zählen vor allem eine kurze Aufgabenbeschreibung, der Lösungsansatz sowie die wesentlichen Ergebnisse der Arbeit. Ein häufiger Fehler für die Kurzfassung ist, dass lediglich die Aufgabenbeschreibung (d.h. das Problem) in Kurzform vorgelegt wird. Die Kurzfassung soll aber die gesamte Arbeit widerspiegeln. Deshalb sind vor allem die erzielten Ergebnisse darzustellen. Die Kurzfassung soll etwa eine halbe bis ganze DIN-A4-Seite umfassen.

Hinweis: Schreiben Sie die Kurzfassung am Ende der Arbeit, denn eventuell ist Ihnen beim Schreiben erst vollends klar geworden, was das Wesentliche der Arbeit ist bzw. welche Schwerpunkte Sie bei der Arbeit gesetzt haben. Andernfalls laufen Sie Gefahr, dass die Kurzfassung nicht zum Rest der Arbeit passt.

The same in english.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1

Einleitung und Problemstellung

Begonnen werden soll mit einer Einleitung zum Thema, also Hintergrund und Ziel erläutert werden.

Weiterhin wird das vorliegende Problem diskutiert: Was ist zu lösen, warum ist es wichtig, dass man dieses Problem löst und welche Lösungsansätze gibt es bereits. Der Bezug auf vorhandene oder eben bisher fehlende Lösungen begründet auch die Intention und Bedeutung dieser Arbeit. Dies können allgemeine Gesichtspunkte sein: Man liefert einen Beitrag für ein generelles Problem oder man hat eine spezielle Systemumgebung oder ein spezielles Produkt (zum Beispiel in einem Unternehmen), woraus sich dieses noch zu lösende Problem ergibt.

Im weiteren Verlauf wird die Problemstellung konkret dargestellt: Was ist spezifisch zu lösen? Welche Randbedingungen sind gegeben und was ist die Zielsetzung? Letztere soll das beschreiben, was man mit dieser Arbeit (mindestens) erreichen möchte.

2

Vergleich der Enterprise Search Engines

In ersten Schritt werden diverse Enterprise Search Engines evaluiert. Dafür wurde eine Anforderungsliste mit den Mitarbeitern erstellt. Die Systeme welche bei diesem Vergleich nach Features verglichen. Die Suchmaschinen, die am besten abschneiden werden anschließend aufgesetzt und genauer verglichen.

Diese Liste zeigt alle Basis-Funktionen, die für die Bibliothek hier K.O. Kriterien sind.

- Open Source oder Kostenlos
- Unterstützung von Facetten
- Ranking der Suchergebnisse
- Volltextsuche
- Support für PDF, SQL, XML
- Logging-Möglichkeit

Des Weiteren sind die folgenden Funktionen stark erwünscht, allerdings nicht ausschlaggebend zu Disqualifikation.

- Support für PostgreSQL
- Backup Funktionen
- Auto-Korrektur und Auto-Vervollständigung
- Security Features
- PHP-Support
- bezahlter Support

Durch die begrenzten finanziellen Mittel und die lange Projektlaufzeit besteht die Notwendigkeit eine kostenfreie, im besten Fall sogar eine Open Source Suchmaschine zu finden. Auch äußerst wichtig ist der Support für Facetten, da Dietrich-Online als Suchmaschine den Nutzer einige Tools zum Verfeinern seiner Suchergebnisse zur Verfügung stellen will. Da wir hier mit großen Datenmengen arbeiten ist das Ranking auch von größer Bedeutung. Es können nicht alle Daten gleichzeitig dargestellt werden, von daher sollten die besten Treffer auch zuerst angezeigt werden. Dabei ist die Transparenz sehr wichtig. Wie das Ranking funktioniert muss erklärt und auf der Webseite veröffentlicht werden. Die Volltextsuche wird es möglich machen auch nach Schlüsselwörtern im Titel oder Beschreibungen zu suchen. Der Support für die verschiedenen Dateiformaten ergibt sich dadurch, dass

dieses Projekt stark gewachsen ist. Es gibt viele Prozessschritte, welche auf denselben Daten in verschiedenen Formen arbeiten. Darunter werden alle Einträge im XML Format bearbeitet, es gibt alle Scans als PDF und für die Webseite sind alle Daten nochmals in der Datenbank vorhanden. Und alle Daten sind wichtig, da die Mitarbeitern-Module auf anderen Daten arbeiten, als der Nutzer. Dabei ergibt sich allerdings auch das Problem, dass sich Daten in der Datenbank doppeln werden. Dies wird sicherlich später nochmals von Bedeutung. Als letztes ist es noch wichtig, dass zumindest ein Fehler-Logging geboten wird, damit schnell und effizient Probleme mit dem System erkannt und gelöst werden können. Ein erweitertes Monitoring ist ein Bonuspunkt.

Ein Support für PostgreSQL ist für dieses Projekt nicht so wichtig, allerdings könnte es sein, dass der Server später auch andere Datenbanken verwaltet. Dadurch wäre ein Support für diese Datenbankstruktur wünschenswert. Es gibt bei den Maschinen in der Bibliothek sowieso ein Backup-Maschinismus. Allerdings ist eine manuelle Backup-Lösung wünschenswert, um die Suchmaschine losgelöst zu sichern und gegebenenfalls auch einfach auf einen anderen Server umzuziehen zu können. Auto-Korrektur und Auto-Vervollständigung sind beide sehr interessant, um den Nutzer mehr Komfort-Funktionen bieten zu können, ohne selbst groß implementieren zu müssen. Die Sicherheitsfunktionen sind für die Suchmaschinen mit Web-Oberfläche interessant. Generell sollte der Server ja nur intern anzusprechen sein. Wenn es allerdings eine Web-Oberfläche gibt, kann es sein, dass diese per Reverse Proxy ansprechbar gemacht wird, um eine Administration aus dem Internet möglich zu machen. Daher wäre es gut, wenn der Server ein Login-System bietet. Einen PHP-Connector, welcher Objekte zum Umgang mit der Suchmaschine bietet, wäre auch wünschenswert. Allerdings bieten einige Suchmaschinen auch die Möglichkeit über JSON Anfragen an die Suchmaschine zu stellen. Es sollte zumindest eine der beiden Methoden verfügbar sein, damit die Suchmaschine einfach von PHP aus zu erreichen ist. Sollte es mal ein Problem geben, was nicht im Haus gelöst werden kann, wäre die Möglichkeit auf bezahlten Support von Vorteil.

2.1 Apache Lucene Core

Lucene Core ist eine Open Source Enterprise Search Engine von der Apache Foundation geschrieben in Java.

Das Lucene Projekt wurde im Jahre 1997 vom Entwickler Doug Cutting gestartet. 2001 ist es dann der Apache Foundation als Teil des Jakarta-Projekts beigetreten und wurde 2005 ein eigenes Hauptprojekt der Foundation. [1]

Lucene Core erfüllt alle der Grundanforderungen. Für das Monitoring gibt es eine Klasse, die es auch ermöglicht, dass langsame Query's geloggt werden. Zudem bietet es Support für PostgreSQL und Auto-Korrektur/Auto-Vervollständigung. Da es keine Web-Oberfläche besitzt, gibt es auch keine weiteren Sicherheitsfunktionen. Einen PHP-Connector gibt es leider auch nicht, man müsste daher mit PHP direkte Systemaufrufe an Java machen. Bezahlten Support gibt es hier nicht, da dieses Projekt zur Apache Foundation gehört. [2]

2.2 Terrier

Terrier ist eine Open Source Enterprise Search Engine geschrieben in Java. Entwickelt und gepflegt wird diese von der University of Glasgow. Sie existiert bereit seit 10 Jahren und besitzt, laut Webseite, eine breite Nutzerbasis. Terrier erfüllt leider nicht alle Grundanforderungen, da es keine direkte Möglichkeit gibt SQL zu indexieren. Es gibt allerdings eine Möglichkeit das SQL in JSON zu konvertieren und dieses dann in die Suchmaschine einzupflegen. Auch scheint es keinen Support für Facetten gegeben. [3]

2.3 Sphinx

Sphinx ist eine Suchmaschine entwickelt von Andrew Aksyonoff. Das Akronym steht für „SQL Phrase Index“.[4] Bis zur Version 2 wurde sie aktiv Open Source entwickelt. Ab Version 3 wurde die Entwicklung Closed Source. Auf der Github-Seite steht: „The sources for 3.0 will also be posted here when we decide to make those publicly available.“ [5], also gibt es kein genaues Datum ob und wann die Version 3 Open Source geht. Version 3.1.1 wurde im Oktober 2018 veröffentlicht und seitdem lässt sich auch nichts mehr über das Projekt finden. Von daher ist davon auszugehen, dass das Projekt nicht mehr weitergeführt wird.

Zu den Features ist festzuhalten, dass es keinen nativen PDF-Support in der Open-Source Version gibt, ab der Version 3 wurde jedoch ein Dokumenten-Speicher eingebaut. Allerdings werden die anderen Anforderungen alle erfüllt. Es existiert, laut Webseite, sogar ein bezahlter Support, allerdings ist fraglich, ob man mit der Firma noch in Kontakt treten kann. [6]

2.4 Apache Solr

Apache Solr ist eine, auf Lucene Core 2.1 basierende, viel eingesetzte Search Engine von der Apache Foundation. Sie erweitert Lucene Core, um ein grafische Benutzeroberfläche und einige weitere Funktionen. Die Entwicklung dafür begann 2004 als ein internes Projekt von CNET um eine bessere Suche für die eigene Webseite zu bieten. Später im Jahre 2006 hat CNET dann den Source Code an die Apache Foundation weitergegeben. Dadurch wurde es zu einem eigenen Projekt bei der Apache Foundation. Im Jahre 2009 wurde Solr dann in das Apache Lucene Projekt eingefügt. Dort wird es auch aktuell noch weiterentwickelt. [7]

Solr wird unter anderem von DuckDuckGo und Best Buy eingesetzt. Durch die Unterstützung von der Apache Foundation längerfristige Weiterentwicklung abzusehen.

Da Solr zur Apache Foundation gehört, ist es Open Source. Es bietet viele Funktionen von Haus, womit es alle Grundanforderungen erfüllt und besitzt darüber hinaus auch Support für fast alle Bonus-Features. Einzig und allein gibt es keinen bezahlten Support, dafür allerdings eine große Community, welche man durch einen Mailing Listen oder IRC erreichen kann. [8]

2.5 ElasticSearch

Eine weitere große Enterprise Search Engine ist ElasticSearch. Auch dieses Projekt arbeitet auf der Basis von Lucene. Zu den bekanntesten Kunden zählen Ebay und Adobe. Gestartet wurde das Projekt in den jungen 2000ern von Shay Banon, um eine Verwaltung für die Rezepte seiner Frau zu schaffen. Im Juni 2012 haben sich dann Logstash, ein Logging Dienst, Kibana, ein UI für ElasticSearch, und ElasticSearch zusammengetan. So enstand der ELK-Stack. Die enstandene Firma nennt sich: ElasticSearch Incorporated. Seitdem wurden der Produktkatalog stetig erweitert und die Produkte weiterentwickelt. Viele der weiteren Produkte sind allerdings nicht mehr Open-Source oder kostenlos. Der ELK-Stack ist allerdings weiterhin, und es wurde versprochen, dass es so bleibt, kostenlos und ElasticSearch zudem auch als Open-Source Variante zu haben. Eine genauere Aussage, welche Features nur in der kostenlosen und nicht in der Open-Source Variante zu finden sind, finden sich in der Tabelle 2.1.

ElasticSearch erfüllt alle der Grundanforderungen, auch in der Open-Source Variante. Auch viele der optionalen Features kann man in der Open Source Variante genießen. Einzig die Sicherheitsfunktionen, wie rollen-basierte Authentifizierung sind der kostenlosen Variante vorbehalten. Eine Möglichkeit auf bezahlten Support besteht auch. [9]

2.6 Fess

Fess ist eine Enterprise Search Engine basierend auf ElasticSearch entwickelt von dem japanischen Unternehmen CodeLibs. Die Suchmaschine ist komplett Open-Source und wird unter der Apache-Lizenz entwickelt.

Die Suchmaschine erfüllt alle Grundanforderungen. Darüber hinaus bietet es Support für PostgreSQL, Backups (sogar über die Web-Oberfläche) und Auto-Korrektur und Vervollständigung. Es gibt keinen direkten PHP Support, allerdings können anfragen über JSON geschickt werden. Ein bezahlter Support ist auch über die Firma N2SM Incorporated. [10] möglich. Bei dieser Arbeiten anscheinend auch einige der Entwickler von Fess. Sicherheitsfunktionen werden über rollen-basierte Authentifizierung mitgeliefert. [11]

2.7 Algolia

Algolia ist eine cloud-basierte Search Engine, welche unter anderem von Twitch und Lacoste verwendet wird. Die Suchmaschine wird hierbei als SAAS (Software as a Service) angeboten. Hierbei lädt man die Daten auf Algolia Server und dann daraufhin über eine API-Schnittstelle die Suchen auf den Daten in der Cloud ausführen.

Sie erfüllt alle Grundanforderungen, wobei allerdings in der kostenlosen Variante grade einmal 10 Tausend Einträge und 50 Tausend Operationen im Monat erlaubt sind. Auch die optionalen Anforderungen werden so weit alle erfüllt. Der

bezahlte Support wird ab der Starter Edition für 30 Dollar im Monat mitgeliefert. [12]

2.8 Manticore Search

Manticore Search Engine ist eine Open-Source Lösung basierend auf Sphinx 2.3. Nachdem Sphinx Closed Source gegangen ist, wurde auf der letzten offenen Version die erste Version von Manticore Search entwickelt. Zu den großen Kunden zählen unter anderem Craigslist und Boardreader.

Manticore erfüllt fast alle Grundanforderungen, allerdings ist kein nativer PDF-Support gegeben. Es muss daher auf eine Konvertierung der Daten auf XML gesetzt werden. Es findet sich außerdem eine Unterstützung von PostgreSQL, sowie Auto-Korrektur und Vervollständigung. Auch werden Log-Dateien produziert. Zuletzt gibt es noch eine Option auf bezahlten Support. Die Supportkosten sind dabei direkt auf der Webseite angegeben und belaufen sich auf 3000 Dollar im Jahr für den Standard Support. [13]

2.9 Xapian

Xapian ist eine Open-Source Enterprise Suchmaschine, welche von Zeit-Online, der Universitätsbibliothek Köln und der Debian Webseite genutzt wird. Die Suchmaschine basiert auf Open Muscat, einer Suchmaschine, welche an der Cambridge Universität in den 1980ern von Dr. Martin Porter entwickelt wurde. In 2001, als Open Muscat Closed Source ging, haben sich einige Entwickler die letzte offene Version genommen und diese weiterentwickelt.

Sie erfüllt alle der Grundanforderungen, wenn auch Logging nur im Grundsinne erfüllt wird, da nur Fehlermeldungen ausgegeben werden. Des Weiteren bietet die Suchmaschine Support für PostgreSQL. Auch eine Replikations-Funktion wird mitgeliefert. Sie bietet auch Auto-Korrektur und Auto-Vervollständigung. Ein Login-System mit Sicherheitsfunktionen gibt es durch das Fehlende Frontend Administration nicht. Es gibt allerdings die Möglichkeit mit Omega eine CGI-Suche zu nutzen. Diese Suche bietet allerdings keine Administration, sondern nur eine grafische Oberfläche für Suchanfragen.

Auch gibt es eine Möglichkeit für bezahlten Support. Auf der Webseite werden zwei Firmen angegeben, welche bezahlten Support bieten. Allerdings funktioniert der Link aktuell nur für eine der beiden Firmen aktuell. Zudem ist ein PHP-Connector für die Suchmaschine vorhanden, was die Einbindung ist das Projekt vereinfacht. [14]

2.10 Datafari

Datafari ist eine Open-Source Enterprise Suchlösung vom französischen Entwickler France Labs. Das Entwicklerstudio wurde 2011 gegründet und hat sich es sich zum

Ziel gemacht, die beste Open-Source Enterpise Suchlösung zu erstellen. [15] Als Fundament dafür wurde hierbei Solr verwendet. Dies wurde dann mit dem ELK-Stack für die Analyse gemischt. Zu den Kunden zählt unter anderem das Linux Magazin, welches diese Suchmaschine in einer ihrer Ausgaben vorstellt. [16]

Die Suchmaschine erfüllt alle Grundanforderungen. Darüber hinaus bietet sie auch Support für PostgreSQL, Auto-Korrektur und Vervollständigung, sowie den bezahlten Support. Eine Backup-Funktion gehört zu den Premium-Funktionen, genauso wie erweiterte Sicherheitsfunktionen. Allerdings ist zumindest die Rollenbasierte-Authentifizierung auch in der Open-Source Variante zu haben. Einen direkt PHP-Connector gibt es nicht, allerdings wird eine HTTP-API zu verfügung gestellt, welche es ermöglicht per POST-Request Anfragen zu stellen. [17]

2.11 Tabellarischer Vergleich

Alle Suchmaschinen die zumindest die Grundanforderungen erfüllen, werden hier in der Tabelle 2.1 nun nochmals aufgeführt für einen leichteren Vergleich.

2.12 Vorauswahl

Nach einem ersten Feature-Vergleich haben nur 7 Suchmaschinen die Grundanforderungen erfüllt. Davon werden nun 4 Stück in den genaueren Vergleich genommen, bei dem die Systeme nun aufsetze und teste. Ich gehe nun die Suchmaschinen der Reihe nach durch und gebe zur jeder eine Begründung warum oder warum sie es nicht in den genauen Vergleich geschafft hat.

2.12.1 Lucene Core

Lucene Core scheidet dadurch aus das es zum einen keine direkte Schnittstelle liefert, die gut mit PHP zur erreichen ist. Die einzige Möglichkeit wären direkte System-Calls, wodurch es schwerer ist, die Systeme voneinander zu separieren, zum Beispiel auf verschiedenen Server zu legen. Zum anderen gibt es für Lucene Core ja eine Erweiterung, namentlich Solr, welches alle diese Probleme löst. [2]

2.12.2 Sphinx

Sphinx wäre eine interessante Alternative gewesen, allerdings durch den Kommunikationsverlust und die gestoppten Updates (Es gab schon seit über einem Jahr kein Update mehr), ist dieses Projekt wohl als tot anzusehen. [6]

2.12.3 Solr

Wie schon bei Lucene Core kurz angesprochen, liefert Solr viele der Funktionen, die in diesem Umfeld benötigt werden, direkt mit. Dazu besitzt es eine Web-Oberfläche

	LC	**SH**	**AS**	**ES**	**FE**	**AG**	**XP**	**DF**
Open Source oder Kostenlos	x	x	x	x	x	x	x	x
Unterstützung von Facetten	x	x	x	x	x	x	x	x
Ranking der Suchergebnisse	x	x	x	x	x	x	x	x
Volltextsuche	x	x	x	x	x	x	x	x
Support für PDF, SQL, XML	x	x*	x	x	x	x	x	x
Monitoring / Logging	x	x	x	x	x	x	x?	x
Support für PostgreSQL	x	x	x	x	x	x	x	x
Backup	-	-	x	x	x	x+	-	-
Auto-Korrektur und Vervollständigung	x	x	x	x	x	x	x	x
Security Features	-	-	x-	x*	x	x	-	x
PHP Support	-	x	x	x	-	x	x	-
bezahlter Support	-	x	-	x	x	x	x	x
unter aktiver Entwicklung**	x	-	x	x	x	x	x	x
offizielles Docker Image	-	-	x	x	x	-	-	x
Synonym Support	x	x	x	x	x	x	x	x
Web-Interface	-	-	x	x	x	x	-	x
Plugin Support	-	x	x	x	x	-	-	-
JSON oder RESTful API	-	x*	x	x	x	-	x–	x
SQL-Like Query Support	-	x	-	x	-	-	-	-

Tabelle 2.1. Feature-Vergleich der verschiedenen Enterprise Suchmaschinen

* = Feature nur in der kostenlosen Variante verfügbar.
** = Update innerhalb des letzten halben Jahres
– = Nur mit Omega CGI installiert
+ = Anbieter kümmert sich um das Feature
- = Funktion nur per Plugin Implementiert

Die Tabelle vergleicht einige Features der ausgewählten Search Engines. Dabei wurden die Namen aus Platzgründen wie folgt abgekürzt:

- LC = Lucene Core 2.1
- SH = Sphinx 2.3
- AS = Apache Solr 2.4
- ES = ElasticSearch 2.5
- FE = Fess 2.6
- AG = Algolia 2.7
- XP = Xapian 2.9
- DF = Datafari 2.10

zur Administration. Durch die aktive Entwicklung unter der Apache-Lizenz und die große Community ist auch eine Langzeit-Entwicklung sehr wahrscheinlich. Daher ist Solr die erste der vier Kandidaten für das genauere Testen. [8]

2.12.4 ElasticSearch

Auch ElasticSearch basiert auf Lucene, ist aber im Gegensatz nicht komplett Open-Source und bietet auch eine kommerzielle Version an, was allerdings auch bedeutet, dass es bezahlten Support gibt. Die Community und der Kundenkreis ist auch groß, was eine Weiterentwicklung sehr wahrscheinlich macht. Auch diese Suchmaschine bietet eine Web-Oberfläche mit besonderem Augenmerk auf die Visualisierung der Daten, was für spätere Administratoren einen einfacheren Einstieg in die Administration liefern könnte. Daher wird auch ElasticSearch den genaueren Vergleich mit eingebunden. [9]

2.12.5 Fess

Fess ist eine Suchmaschine, welches auf ElasticSearch basiert, was ja seinerseits auf Lucene basiert. Von den Funktionen her bietet Fess, dank der Basis, viele Möglichkeiten. Es gibt auch kommerziellen Support, allerdings nur von einer japanischen Firma. Dadurch kann es schwere werden mit dem Support in Kontakt zu treten, was mich dazu veranlasst Datafari 2.10 dieser Suchmaschine vorzuziehen. Von der Idee her machen die beiden Firmen ja etwas ziemlich Ähnliches. [11]

2.12.6 Algolia

Als einzige Cloud-only Lösung im Vergleich, bietet Algolia eine interessante Alternative. Leider sind im kostenlosen Bereich nicht genügend Einträge speicherbar. Auch sind 50.000 Operationen zu wenig für die das Dietrich-Online Projekt. Von daher fällt diese Suchmaschine durch diese Limitationen raus. [12]

2.12.7 Xapian

Xapian ist als einzige Suchmaschine ohne Web-Administration im engeren Vergleich. Durch die Nutzung der Suchmaschine für die Bibliothek Köln gibt es einen Kunden der Software, welcher einen ähnlichen Anwendungsfall besitzt. [18] Dadurch und die Erfüllung vieler weiterer Kriterien kommt diese Suchmaschine auch in die engere Auswahl. [14]

2.12.8 Datafari

Datafari ist der letzte Kandidat, der es in die engere Auswahl schafft, wie oben schon erwähnt gewinnt diese Suchmaschine gegen Fess, durch die Entwicklung in Frankreich und der daher besser zu erreichende Support. Darüber hinaus ist es interessant zu sehen, ob das Entwicklerstudio schafft Solr sinnvoll zu erweitern und die Datenaufbereitung mit ElasticSearch so zu liefern, dass sich die Suchmaschine trotzdem noch wie aus einem Guss anfühlt. [17]

2.12.9 Abschluss

Daraus ergibt sich nun, dass die folgenden Suchmaschinen es in die genauere Auswahl geschafft haben: die beiden Platzhirsche Apache Solr 2.4 und ElasticSearch 2.5, sowie Datafari 2.10 als Open-Source Erweiterung von Solr und Xapian 2.9 als Lucene-freie Alternative. Ich hätte auch gerne eine Suchmaschine, welche auf Sphinx basiert dabei gehabt, allerdings, war die einzige noch aktive Alternative zu Sphinx direkt Manticore Search. Diese hat aber bisher noch keinen PDF-Import, welcher leider zwingend erforderlich ist.

3

Genauer Vergleich

In diesem Kapitel werden die vorher ausgewählten Suchmaschinen genauer verglichen. Dafür werden alle vier Suchmaschinen aufgesetzt und getestet. Hier wird Wert auf alle Aspekte des Prozesses gesetzt. Da ich dieses Projekt nicht nach meiner Bachelor-Arbeit wohl nicht weiter verfolgen kann, ist es auch wichtig zu schauen, wie leicht ein neuer Administrator sich in das System einlernen kann, beziehungsweise wie leicht das System zu verstehen und administrieren ist. Deshalb wird auch die Dokumentation verglichen und geschaut, wie groß die Community der einzelnen Suchmaschinen ist. Die genaueren Kriterien werden nun im Folgenden mit Erklärungen aufgeführt. Da es nicht für jede der Suchmaschinen ein offizielles Docker-Image gibt, werden der Fairness halber alle Systeme manuell aufgesetzt.

Das Test-System hat folgende Spezifikationen:

- CPU: 4 Kerne
- RAM: 16 Gigabyte
- Festplattenspeicher: 20 GB
- Betriebssystem: Ubuntu 18.04.03 LTS

Auf das System wird daraufhin die MySQL Datenbank von Dietrich-Online Projekt aufgespielt. Dies für diesen Vergleich die einzige Datenquelle sein.

3.1 Aufbau der Tests

3.1.1 Installation

Im ersten Schritt wird die Installation bewertet, dabei wird geschaut, wie einfach es ist die Software zu installieren. Hierbei ist es wichtig zu schauen, wie simpel die Installation ist. Existiert zum Beispiel ein Installations-Wizard? Wie viel muss manuell in den Dateien geändert werden? Müssen viel externe Programme nachinstalliert werden?

3.1.2 Oberfläche

Als Nächstes folgt der Ersteindruck der Software und Oberfläche. Dabei wird geschaut, wie übersichtlich die Oberfläche ist, falls eine gegeben ist, und wie

verständlich das System für Einsteiger ist. Dafür wird im ersten Schritt möglichst auf die Dokumentation verzichtet, um einen Ersteindruck zu liefern, wie gut die Oberfläche für sich selbst spricht. Dies dient dazu um, zu schauen wie der neue Administrator bestimmte Aufgaben ohne Vorkenntnisse erfüllen kann. Besondere Punkte dabei sind zum Beispiel: Wie viel kann man über die Oberfläche konfigurieren? Lassen sich Updates direkt über die Oberfläche einspielen? Ist die Seite responsive? Wie funktioniert die Nutzerverwaltung?

3.1.3 Indexierung

Hier geht es darum festzustellen, wie einfach eine Indexierung der einzelnen Dateien möglich ist. Darunter fällt zum Beispiel: Kann man die Daten über die grafische Benutzeroberfläche indexieren lassen? Kann man das System darauf anweisen Änderungen direkt neu zu indexieren?

3.1.4 Dokumentation

Im dritten Schritt wird die Dokumentation analysiert. Hierbei wird das Augenmerk auf die Übersichtlichkeit und Verständlichkeit gelegt. Auch hier ist es wieder wichtig zu schauen, ob die Dokumentation auch ohne Vorkenntnisse gut zu verstehen ist. Da in diesen Kurztest nicht alle Funktionen durchgetestet werden können, ist es leider auch nicht möglich zu schauen, ob alle Funktionen korrekt und ausführlich dokumentiert sind. Sollte allerdings schon von den Grundfunktionen eine schlechte oder fehlende Dokumentation auffallen wird dies natürlich erwähnt.

3.1.5 Datenschutz

Hier geht es darum zu schauen, ob das Tool nach Hause telefoniert. Dann soll noch überprüft werden, ob die Logdateien anonymisiert und nach gewisser Zeit gelöscht werden können.

3.1.6 Absetzen einer Anfrage und Integration in PHP

Im letzten Schritt werden einige Query's abgesetzt. Dies wird zum einen über die Oberfläche geschehen, falls vorhanden, und was wichtiger ist über die Schnittstelle für PHP. Dafür wird ein PHP-Script geschrieben und die Laufzeit des Scripts gemessen.

Der erste Query ist einer der bisher am langsamsten laufenden Query's. Er ermittelt alle Lemmata vom Buchstaben S und baut alle Daten, die zur Anzeige benötigt werden zusammen 3.4. Die Tabellen die für diese Ansicht gebraucht werden, sind in diesen Diagramm 3.1 zu finden. Eigentlich muss man sagen, dass es sich hierbei nicht um einen Query handelt, sondern um zwei. Der erste Sammelt alle ID's aus der Datenbank, welche unter dem Buchstaben zu finden sind:

```
1       SELECT
2       lemma.id
3   FROM lemma
```

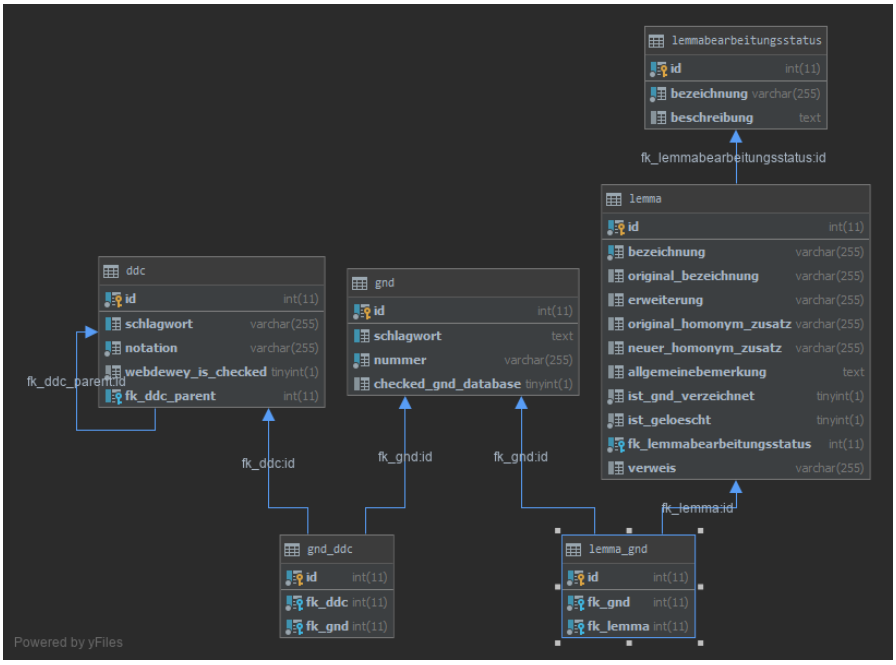

Abb. 3.1. Tabellenaufbau der Lemma-Administration Übersicht.

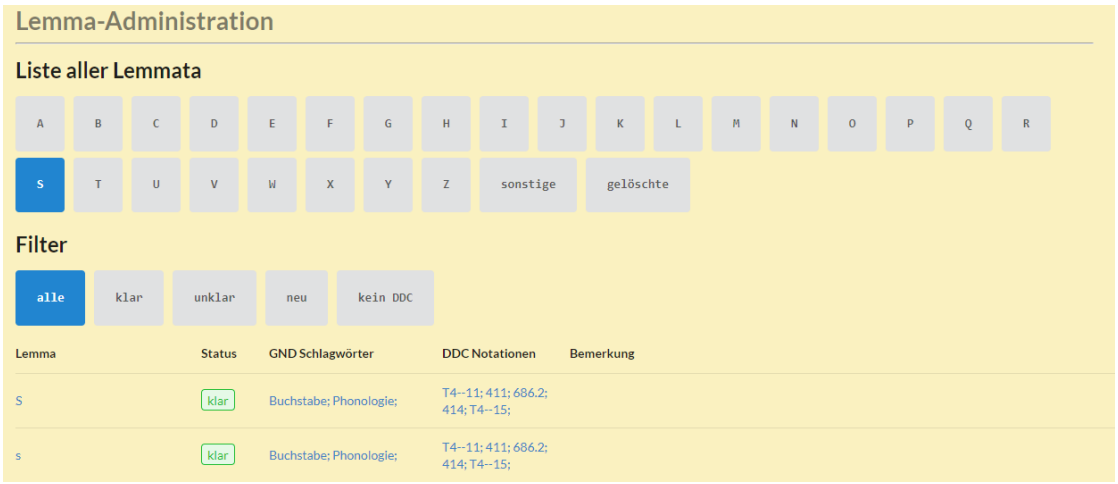

Abb. 3.2. Frontend Ansicht der Lemma-Administration mit geladenen Buchstaben S (Ausschnitt).

```
4  WHERE
5      lemma.bezeichnung LIKE 'S%'
6      AND lemma.ist_geloescht = 0
7  ORDER BY
8      lemma.bezeichnung ASC,
9      lemma.id ASC;
```

Im zweiten Schritt werden dann die gerade geholten ID's mit vielen JOIN's für die Darstellung vorbereitet.

```
1  SELECT  lemma.id,
2          lemma.bezeichnung,
3          lemma.original_bezeichnung,
```

```
4           lemma.erweiterung,
5           lemma.original_homonym_zusatz,
6           lemma.neuer_homonym_zusatz,
7           lemma.allgemeinebemerkung,
8           lemma.ist_gnd_verzeichnet,
9           lemma.ist_geloescht,
10          lemma.verweis,
11          lemma.fk_lemmabearbeitungsstatus,
12          lemmaBStatus.id,
13          lemmaBStatus.bezeichnung,
14          lemmaBStatus.beschreibung,
15          lemma_gnd_map.id,
16          gnd.id,
17          gnd.nummer,
18          gnd.schlagwort,
19          gnd_ddc_map.id,
20          ddc.id,
21          ddc.notation,
22          ddc.schlagwort,
23          ddc.webdewey_is_checked,
24          lemma_gnd_map.fk_lemma,
25          lemma_gnd_map.fk_gnd,
26          gnd_ddc_map.fk_gnd,
27          gnd_ddc_map.fk_ddc,
28          ddc.fk_ddc_parent
29  FROM lemma lemma
30
31  INNER JOIN lemmabearbeitungsstatus lemmaBStatus
32  ON lemma.fk_lemmabearbeitungsstatus = lemmaBStatus.id
33
34  LEFT JOIN lemma_gnd lemma_gnd_map ON lemma.id = lemma_gnd_map.fk_lemma
35  LEFT JOIN gnd gnd ON lemma_gnd_map.fk_gnd = gnd.id
36  LEFT JOIN gnd_ddc gnd_ddc_map ON gnd.id = gnd_ddc_map.fk_gnd
37  LEFT JOIN ddc ddc ON gnd_ddc_map.fk_ddc = ddc.id
38  WHERE lemma.id IN ([Array of Lemma IDs])
39  ORDER BY lemma.bezeichnung ASC, lemma.id ASC;
```

Der zweite Query ist von dem Fehlermodul 3.4. Dieser Query verbindet die drei Tabellen 3.3 zu einer Darstellung. Es gibt für jeden Fehler einen Eintrag, dieser wird in tf_fehler gespeichert. Da es für jeden Eintrag mehrere Fehler geben kann und diese Fehler für jeden Eintrag verfügbar sind, gibt es hier eine n zu m Beziehung. Um diese Abzubilden wurde die Tabelle tfl_fehler_rptmap eingefügt. Diese verbindet jeweils die ID's mit den dazugehörigen Fehlercodes.

```
1   SELECT  fehler.eid,
2           fehler.originalText,
3           fehler.band,
4           fehler.bd_seite,
5           fehler.bd_spalte,
6           fehler.bd_zeile,
7           fehler.lemma,
8           fehler.charErrors,
9           fehlercodes.description
10  FROM tfl_fehler fehler
11
12  LEFT JOIN tfl_fehler_rptmap fehler_fehlercode_map
13  ON fehler.eid = fehler_fehlercode_map.fk_fehler
14
15  LEFT JOIN tfl_rptmap fehlercodes
16  ON fehlercodes.id = fehler_fehlercode_map.fk_rptmap
17
18  WHERE fehler.band = ('1')
19  AND fehler.severity > 3
20  ORDER BY fehler.eid;
```

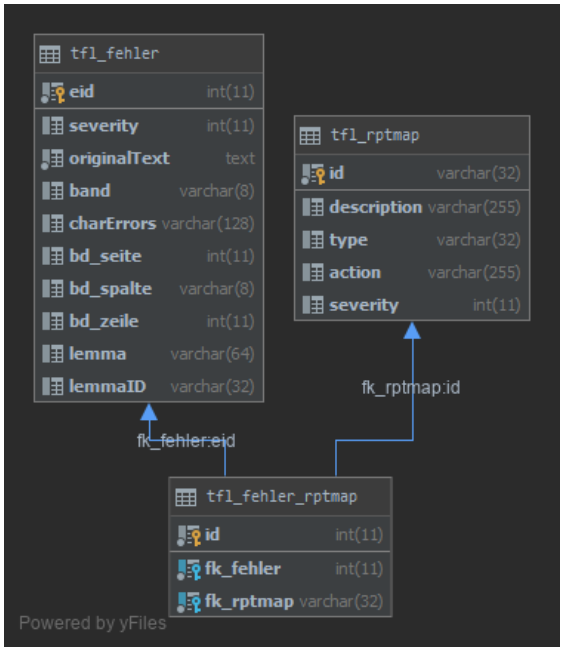

Abb. 3.3. Tabellenaufbau der Fehlerliste.

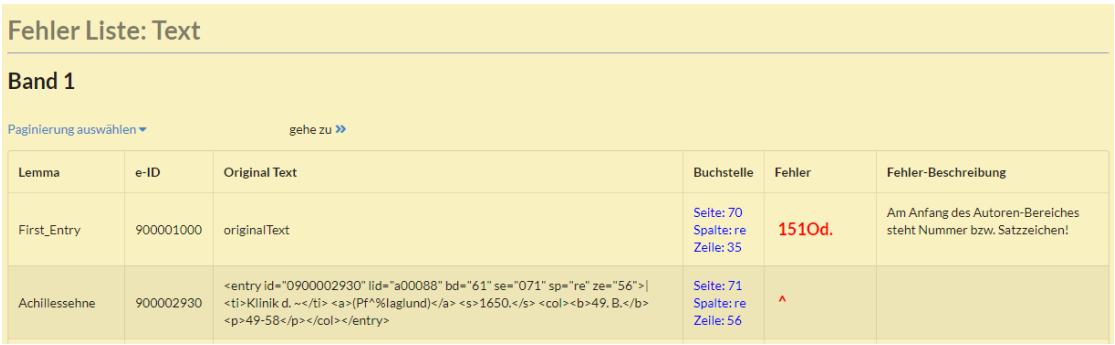

Abb. 3.4. Ansicht der Frontend Tabelle.

3.1.7 Vorbereitung

Auf den Server werden folgenden Programme zur Vorbereitung installiert.

Für die Konfiguration des Servers wurde, von dem Hauptadministrator folgende Programme aufgespielt:

LISTE MARTIN

Ich habe dann zusätlich noch den MariaDB-Server installiert, um eine Replikation der aktuellen Datenbank aufzuspielen.

3.2 Solr

Der erste Kandidat ist Solr. Der Download gestaltet sich hierbei denkbar einfach mit einem großen Download-Knopf direkt auf der Homepage [8]. Da Solr komplett Open Source ist, kann sich neben den Binary's auch der Source-Code heruntergeladen werden.

3.2.1 Installation

Bevor die Installation starten kann, muss zuerst eine Java Version ¿ 8 installiert werden. Ich habe mir daraufhin für Java 11 entschieden, da eine neuere Version von OpenJDK nicht auf dieser Ubuntu-Version verfügbar ist.

Die Development-Installation ist denkbar einfach. Zuerst wird Solr aus dem Archiv entpackt und dann mit `bin/solrstart` gestartet. Hierbei wurde ich allerdings direkt von 2 Warnungen begrüßt.

```
1       *** [WARN] *** Your open file limit is currently 1024.
2       It should be set to 65000 to avoid operational disruption.
3
4       *** [WARN] ***  Your Max Processes Limit is currently 63918.
5       It should be set to 65000 to avoid operational disruption.
```

Dieses Problem ließ sich aber schnell mit einer Anpassung in der `/etc/security/limits.conf` 3.2.1 lösen. Danach muss sich nur neu eingeloggt werden und die Warnungen waren verschwunden.

```
1
2       #<domain>               <type>  <item>    <value>
3         reitz                 soft    nofile    65000
4         reitz                 hard    nproc     65000
5         reitz                 soft    nproc     65000
```

Nun Startete der Server komplett ohne Warnungen. Nun musste nur noch der Port in der hier sehr restriktiven Firewall freigeben werden, es wird sämtliche ankommende Verbindungen geblockt, und schon lief die Weboberfläche ohne Probleme. Dazu dann mehr in den Oberflächen-Teil.

Nun noch die richtige Installation aus Service. In dem entpackten Solr-Ordner findet man ein Installation-Skript. Diese benötigt das gepackte Solr als Argument. Solar legt sich dabei einen eigenen Nutzer an, also müssen die Limits auch für diesen User wieder geändert werden. Solr selbst startet sich unter dem Nutzer solr, der auch alle Rechte für die Verzeichnisse besitzt. Will man also etwas an den Konfigurationsdateien ändern, muss auf diesen Nutzer geloggt werden.

3.2.2 Indexierung

Um mit der Indexierung starten zu können, muss zuerst in sogenannter „Core“ erstellt werden. Dieser ist ein Index mit dazugehörigen Transaktionslog und Konfigurationsdateien. Nur mit diesen ist es möglich Dateien zu indexieren und auf ihnen zu suchen. Die Erstellung eines Core's läuft mit dem Befehl: `bin/solrcreate-c<name>` ab. Dieser erscheint daraufhin auch direkt in der Oberfläche. Nun ist das Solr-System bereit für die Indexierung.

Als ich nun über die Oberfläche, unter den Punkt Dataimport meine Datenbank einlesen wollte, ließ mich Solr wissen, dass die solrconfig.xml noch keinen DataImportHandler besitzt.

Um diesen zu erstellen, muss auf den Server zuerst in das Verzeichnis des Core's gewechselt werden, dies ist normalerweise unter `SOLRORDNER/server/solr/CORENAME` zu finden. In dem Verzeichnis gibt es die solrconfig.xml. In dieser muss jetzt der DataImportHandler angegeben werden. [19]

Dafür habe ich zuerst einen passenden Treiber heruntergeladen und diesen nach `SOLRORDNER/contrib/dataimporthandler/lib` entpackt. Als Nächstes muss eine XML-Datei definiert werden, die angibt, welche Spalten und Tabellen indiziert werden sollten. GRAFIK EINFÜGEN. Nachdem dies geschehen ist, muss man Solr noch sagen, dass es dieses DataImportHandler auch verwenden soll. Dafür muss ein Eintrag in der solrconfig.xml gemacht werden. Ist dies geschehen, kann nun in der Weboberfläche der Crawler über die Daten gelaufen lassen. [20]

Als ich den Crawler allerdings startete, meldet mir Solr, das wenige Einträge indexiert wurden. Bei genauerer Betrachtung fand ich heraus, dass nur eine Tabelle von Solr betrachtet wurde. Als ich mir nun die Daten dieser Tabelle herausgegeben habe, bemerkte ich, dass nur das Feld „id“ indexiert wurde. Dies liegt daran, dass nur Felder in den Solr-Index geschrieben werden, die vorher im Schema festgelegt wurden. Dieses Schema kann man entweder über die Weboberfläche oder die API geändert werden kann. Es gibt auch eine Möglichkeit, das Schema direkt zu ändern, allerdings ist diese Methode nicht mehr erwünscht, da die API Fehler erkennt und Einträge so direkt ablehnt.

RUNTIME LOW RAM Requests: 435,026 4,439/s, Fetched: 490,052 5,001/s, Skipped: 0 , Processed: 138,366 1,412/s

3.2.3 Oberfläche

3.2.4 Dokumentation

3.2.5 Absetzen einer Anfrage und Integration in PHP

4

Nutzung des Open Archives Initiative Protokolls für Metadaten

Das Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) ist ein Protokoll zum Austausch von Metadaten. Dabei werden Anfragen per GET oder POST-Request angefragt. Als Antwort erhält man im Folgenden ein XML-Dokument. So können die Metadaten mit bestimmten Facetten abgefragt werden (zum Beispiel Autor). Dabei geht es allerdings darum primär darum Änderungen weiterzugeben. So können durch dieses Protokoll neue Einträge oder Änderungen in der Datenbank weitergeben werden. [21]

4.1 OAI Harverester

Ein OAI Harverester ist ein Programm, welches durchgehend einen Abgleich der Daten vollführt. Dabei lässt es sich die Änderungen mit einem List-Befehl von dem Server geben und gleicht diese danach mit der eigenen Struktur ab. Sollten dabei Unterschiede festgestellt werden, werden daraufhin die Änderungen auch beim Harverester eingefügt. So steht der Harverester immer mit dem Server auf einen Stand. [21]

4.2 Support der Enterprise Search Engines

Bei den vorhin genannten Enterprise Search Engines gibt es keine mit nativen OAI Harverester Support. Es gibt die Möglichkeit für manche der Suchmaschinen ein solches Verhalten mithilfe von Plugins zu implementieren. Allerdings sind die meisten dieser Add-ons auch schon veraltet.

4.3 Auswertung

Durch eine fehlende Basisimplementierung des Protokolls in den einzelnen Suchmaschinen und der Möglichkeit eines direkten Zugriffs auf die Datenbank, sehe ich keinen Grund dieses Protokoll zu verwenden. Es müsste ein Server vor die Datenbank installiert werden und ein Harverester vor der ESE. Dies ist ein großer Mehraufwand, welcher bei diesen Anwendungsfall nicht notwendig ist. Sollte allerdings

diese Suchmaschine ein übergreifendes System werden, kann darüber nachgedacht werden, die anderen Datenbanken per OAI-Harverester anzusprechen.

5

Zusammenfassung und Ausblick

In diesem Kapitel soll die Arbeit noch einmal kurz zusammengefasst werden. Insbesondere sollen die wesentlichen Ergebnisse Ihrer Arbeit herausgehoben werden. Erfahrungen, die zum Beispiel Benutzer mit der Mensch-Maschine-Schnittstelle gemacht haben oder Ergebnisse von Leistungsmessungen sollen an dieser Stelle präsentiert werden. Sie können in diesem Kapitel auch die Ergebnisse oder das Arbeitsumfeld Ihrer Arbeit kritisch bewerten. Wünschenswerte Erweiterungen sollen als Hinweise auf weiterführende Arbeiten erwähnt werden.

Literaturverzeichnis

1. “Apache lucene - wikipedia,” 27.10.2019. [Online]. Available: https://en.wikipedia.org/w/index.php?oldid=915250662
2. “Apache lucene - apache lucene core,” 26.07.2019. [Online]. Available: https://lucene.apache.org/core/
3. R. McCreadie and Craig Macdonald and Jie Peng, “Terrier ir platform - homepage,” 25.01.2019. [Online]. Available: http://terrier.org/
4. “Sphinx — open source search server.” [Online]. Available: http://sphinxsearch.com/docs/manual-2.3.2.html#intro
5. “Sphinx search server.” [Online]. Available: https://github.com/sphinxsearch/sphinx
6. “Sphinx — open source search engine.” [Online]. Available: http://sphinxsearch.com/
7. “Apache solr - wikipedia,” 14.10.2019. [Online]. Available: https://en.wikipedia.org/w/index.php?oldid=915250761
8. “Apache solr,” 26.07.2019. [Online]. Available: https://lucene.apache.org/solr/
9. “Elasticsearch: Verteilte restful-suchmaschine und -analytics engine — elastic.” [Online]. Available: https://www.elastic.co/de/products/elasticsearch
10. “Professional support — n2sm, inc.” [Online]. Available: https://www.n2sm.net/en/support/fess_support.html
11. “Fess installation guide,” 31.10.2019. [Online]. Available: https://fess.codelibs.org/13.4/install/index.html
12. “Fast, reliable and modern search and discovery.” [Online]. Available: https://www.algolia.com/
13. “Manticore search – open source text search engine for big data and stream filtering.” [Online]. Available: https://manticoresearch.com/
14. “The xapian project,” 14.10.2019. [Online]. Available: https://xapian.org/
15. F. Labs, “About << france labs: Open source enterprise search,” 2018. [Online]. Available: https://www.francelabs.com/en/about.html
16. Michael Brandenburg, “Suchtrupp: Eine eigene suchmaschine bauen (teil 1),” *LINUX-Magazin: Die Zeitschrift für LINUX-Professionals*, no. 11, pp. 62–68, 2019. [Online]. Available: https://www.linux-magazin.de/ausgaben/2019/11/datafari/

17. F. Labs, “Datafari enterprise search.” [Online]. Available: https://www.datafari.com/en/index.html
18. “Xapian users,” 14.10.2019. [Online]. Available: https://xapian.org/users
19. “Apache solr reference guide — apache solr reference guide 8.3,” 05.11.2019. [Online]. Available: https://lucene.apache.org/solr/guide/8_3/index.html
20. Iqubal Mustafa Kaki, “Solr indexing - mariadb table data into apache solr,” 2016. [Online]. Available: https://erimkaki.blogspot.com/2016/01/solr-indexing-mariadb-table-data.html
21. “Oai-schnittstelle,” 31.05.2019. [Online]. Available: https://www.dnb.de/DE/Professionell/Metadatendienste/Datenbezug/OAI/oai_node.html

A

Glossar

ESE	Enterprise Search Engine
Facetten	Filter in Bibliothekarssprache
OAI	Open Archives Initiative

B

Erklärung der Kandidatin / des Kandidaten

☐ Die Arbeit habe ich selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Datum	Unterschrift der Kandidatin / des Kandidaten