

Automated Verification of Scientific Literature References

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering und Internet Computing

eingereicht von

Thomas Ruf, BSc

Matrikelnummer 11777753

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Peter Purgathofer

Wien, 22. August 2024

Thomas Ruf

Peter Purgathofer

Automated Verification of Scientific Literature References

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering and Internet Computing

by

Thomas Ruf, BSc

Registration Number 11777753

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Peter Purgathofer

Vienna, August 22, 2024

Thomas Ruf

Peter Purgathofer

Erklärung zur Verfassung der Arbeit

Thomas Ruf, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, habe ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 22. August 2024

Thomas Ruf

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Entstehung dieser Diplomarbeit und meines Studiums unterstützt haben.

Mein besonderer Dank gilt meinen Eltern, die mich während meines Studiums nicht nur finanziell unterstützt haben, sondern mir auch stets den Rücken gestärkt haben. Ohne eure Hilfe wäre es mir nicht möglich gewesen, dieses Studium erfolgreich zu absolvieren.

Ebenso möchte ich mich bei meiner Freundin bedanken, die immer an meiner Seite war, mich stets motiviert und mir in schwierigen Momenten den nötigen Rückhalt gegeben hat. Deine Unterstützung und Geduld haben wesentlich zum Gelingen dieser Arbeit beigetragen.

Nicht zuletzt möchte ich mich bei meinem Betreuer, Herrn Professor Peter Purgathofer, bedanken, dessen wertvolle Betreuung, konstruktive Kritik und vor allem die gewidmete Zeit, maßgeblich dazu beigetragen haben, diese Arbeit in ihrer jetzigen Form zu realisieren. Vielen Dank für Ihr Engagement und Ihre Unterstützung.

Acknowledgements

I would like to take this opportunity to thank all those who have supported me during the development of this thesis and my studies.

My special thanks go to my parents, who not only supported me financially during my studies, but also always had my back. Without your help, it would not have been possible for me to successfully complete this degree.

I would also like to thank my girlfriend, who was always by my side, always motivated me and gave me the support I needed in difficult moments. Your support and patience have contributed significantly to the success of this thesis.

Last but not least, I would like to thank my supervisor, Professor Peter Purgathofer, whose valuable supervision, constructive criticism and, above all, the time he dedicated to me, have contributed significantly to the realization of this thesis in its current form. Thank you very much for your commitment and support.

Kurzfassung

Neueste Studien belegen eine Zunahme an wissenschaftlichen Werken, die unter Zuhilfenahme generativer KI-Tools verfasst wurden. Die Anwendung dieser Tools beschränkt sich nicht auf die Generierung von Textinhalten, sondern umfasst auch die Generierung von wissenschaftlichen Referenzen zu diesen Inhalten. Wie Studien zeigen, handelt es sich bei einem Großteil dieser generierten Referenzen um Halluzinationen, also ausgedachte Referenzen, die in der Realität nicht existieren. Daher sind technologische Strategien erforderlich, um den Prozess der Verifizierung wissenschaftlicher Referenzen zu vereinfachen.

Diese Arbeit befasste sich mit der Frage, in welchem Umfang ein technischer Prozess zur Authentifizierung wissenschaftlicher Referenzen eingesetzt werden kann. Zudem wurde erörtert, wie ein derartiges System für die Öffentlichkeit bereitgestellt werden sollte, um eine produktive Nutzung zu ermöglichen. Die theoretische Analyse umfasste die Erforschung des aktuellen Stand der Technik, von Zitierpraktiken, Möglichkeiten der Datensammlung sowie der Bereitstellung der Anwendung. Zusätzlich wurde eine Evaluierung unterschiedlicher wissenschaftlicher Suchmaschinen und Datenbanken durchgeführt.

Als Teil der praktischen Analyse wurde ein technisches Verfahren in Form einer Webapplikation entwickelt, welches eine effiziente Überprüfung von wissenschaftlichen Referenzen ermöglicht. Hierfür verfügt das System über die Funktionen, mehrere Referenzen entgegenzunehmen, gleichzeitig zu verarbeiten und die daraus resultierenden Ergebnisse übersichtlich darzustellen.

Die resultierende Anwendung diente als Grundlage für die anschließenden qualitativen Experteninterviews, die durchgeführt wurden, um empirische Daten über die Benutzerfreundlichkeit und Wahrnehmung sowie die Funktionalität der Anwendung im Hinblick auf ihren Verwendungszweck zu sammeln. Im Rahmen der Experteninterviews wurden sowohl die Möglichkeiten als auch die Grenzen der Anwendung hervorgehoben. Zudem konnte festgestellt werden, dass die Anwendung eine effektive Methode zur Erkennung erfundener wissenschaftlicher Referenzen darstellen könnte.

Keywords: *Verifizierung von Referenzen, Generative KI, Webapplikation, wissenschaftliche Referenzen, KI-generierter Inhalt, Halluzination, automatisierte Überprüfung, ChatGPT, technisches Verfahren*

Abstract

Latest studies have shown that the number of scientific papers that include content written by generative AI tools is increasing rapidly. These tools are not only used to generate text but also to provide scientific references to this content. As studies have shown, the majority of these generated references are fabricated, also known as hallucinations. Therefore, technical strategies are required to facilitate the process of verifying scientific references efficiently.

This thesis explored the extent to which a technical procedure can be used to determine whether a scientific reference is genuine or not and how such a system should be made available to people for productive use. The theoretical analysis included an investigation of current state of the art, citation practices, data collection methodologies, implementation strategies and an evaluation of scientific sources like search engines and databases.

As part of the practical analysis, this thesis examined the development of a technical procedure in the form of a web application that enables the automated checking of scientific references. Accordingly, the system is equipped with the capability to accept and process multiple scientific references concurrently, and to present the resulting data in a comprehensible format.

The resulting application served as the foundation for subsequent qualitative expert interviews, which were conducted to gather empirical data concerning user perceptions, usability, and functionality of the application in relation to its intended use. These qualitative interviews highlighted the applications' capabilities and limitations, while also indicating that it could be an effective solution to facilitate the process of identifying fabricated scientific references.

Keywords: *Reference verification, Generative AI, web application, scientific references, AI-generated content, hallucinations, automated checking, ChatGPT, technical procedure*

Contents

Kurzfassung	xi
Abstract	xiii
Contents	xv
1 Introduction	1
1.1 Motivation	1
1.2 Research Objectives	2
1.3 Methodology	2
1.4 Structure	3
2 State of the Art	5
2.1 Plagiarism Checkers	5
2.2 Recite	5
2.3 Scribbr Citation Checker	6
2.4 Checking References manually using Academic Search Engines	7
2.5 Scite.ai	7
2.6 Trinka.ai	8
2.7 Comparison	8
3 Concept	11
3.1 Features	11
3.2 Functionality	14
3.3 Expected Benefits	16
4 Development	19
4.1 Architecture	19
4.2 Technologies	19
4.3 Limitations	25
5 Design and Usability	27
5.1 Development Phase	27
5.2 Structure	31
	xv

5.3 Design Decisions	39
6 Evaluation	43
6.1 Process	43
6.2 Findings	45
6.3 Refinements	49
7 Discussion	51
8 Conclusion and Future Perspectives	55
Overview of Generative AI Tools Used	57
List of Figures	59
List of Tables	61
List of References for Testing	63
Bibliography	67



Introduction

The objective of this thesis is to analyze, design, develop, and test a prototype application that provides the functionality of verifying scientific literature references in an automated manner. Therefore, the following sections describe the motivation, research objectives, the methodology as well as the structure of the thesis.

1.1 Motivation

Generative AI tools, especially ChatGPT, have gained significant attention in recent months. According to OpenAI chief Sam Altman, ChatGPT has 100 million weekly users in less than a year after its launch [1]. Beside its numerous features like answering questions, solving math equations, translating between languages, debugging code or writing stories and poems [2], it offers valuable functionalities to academic research: literature review assistance, automated summarization, data analysis, but also text generation like creating draft versions of research papers or proposals [3]. A recently released study has estimated that at least 60,000 papers published in 2023 were written with the assistance of large language models (LLMs) [4]. Particularly in this context, ChatGPT can be used to provide automated reference and information services [3] and when prompted also include accurate-looking scientific references [5].

However, ChatGPT tends to generate fabricated references, which can include inexact authors, titles and years, but also DOI codes without correspondence [6] which has been preliminary reported [7][8]. Prior studies have shown that this problem does not only relate to ChatGPT but also Bard 2.0, where generated citations showed 6% accuracy or no citations at all for ChatGPT 3.5 and 0% accuracy or 66% accuracy after repeated queries for Bard 2.0 [9]. In academic research, this represents an enormous risk of bias that needs to be avoided promptly since references serve a crucial purpose in providing a basis for evaluating evidence, supporting claims, identifying knowledge gaps and establishing context [6].

1.2 Research Objectives

Since there are currently no products that offer the functionality to verify scientific references efficiently and conveniently, this thesis plans to fill this gap by developing an easily accessible web application. The technical procedure embedded in this application aims to check multiple references automatically and provide the user with informative statistics in order to support the assessment of used scientific references and in further consequence the quality of scientific literature.

The primary aim is to explore technical opportunities to empower users with a tool that simplifies the process of validating such references, offering a more efficient and user-friendly experience. Hence, the application is envisioned to generate comprehensive reports, presenting information and statistics pertaining to the obtained results. Throughout the course of this thesis, the study aims to address the following research questions:

1. To what extent can a technical procedure be used to find out whether a reference is genuine or not?
2. How should such a system be made available to people so that they can use it productively?

An application that automatically detects whether listed references are fake or real could help expose fraudulent scientific papers or articles that are possibly written by generative AI tools or which are simply referencing non-existing sources. While the automated reference-checking functionality promises a substantial reduction in time and effort compared to the traditional manual approach for everyone, it particularly benefits people who have to check multiple scientific papers or articles regularly. By making such a technical procedure available to the broad mass through an easily accessible web application, this could eventually combat the increasing number of scientific works with incorrect references, possibly written by generative AI.

1.3 Methodology

In order to answer the previously mentioned research questions, this thesis performs a theoretical, practical and empirical analysis. Starting with the theoretical analysis, a fundamental building block is laid, providing an understanding and overview of the different types of references and their characteristics, relevant sources for scientific works in different subject areas and where to find which type of reference. Additional research has to be done regarding how this information can then be efficiently collected and how such an application should be provided in order to be accessible and usable for most people.

The development phase then applies the gained insights from the theoretical analysis by coming up with a concept design, the actual implementation of a prototype and a final refinement based on the feedback from the empirical analysis.

In the course of the empirical analysis, post development qualitative interviews [10] are held to gather empirical data concerning user perceptions, usability and functionality of the application. The aim is to determine to what extent the prototype retrieved from the development phase can fulfill requirements regarding research questions 1 and 2. Since the regular engagement with scientific work is a requirement for significant interviews, purposive sampling [11] will be utilized to select interview participants.

1.4 Structure

This thesis consists of 8 consecutive chapters. Chapter 1 provides an overview of this thesis. Therefore, it outlines the motivation and significance of the topic. It introduces the research objectives and the methodology and how they are going to be achieved.

Subsequently, Chapter 2 does present the current state of the art, its methodologies and technologies. The concluding section provides a comparison and highlights gaps that this thesis addresses.

Chapter 3 provides an introduction to the planned concept of the application. It summarizes general information about the features and explains the assessment procedure used to evaluate supported search engines. Additionally, it describes the functionality of individual steps in the technical procedure for checking scientific references in an automated manner. Finally, expected benefits of a functional implementation are discussed.

Chapter 4 presents the development process of the application. While introducing the architecture and used technologies in detail, it also discusses associated limitations.

Details concerning the design and usability of the application are then presented in Chapter 5. It discusses the structure of the user interface, while explaining certain decisions concerning design and usability of individual components.

Chapter 6 provides an overview of the evaluation of the thesis. It describes the process of how the qualitative interviews have been conducted, the resulting findings, and the subsequent refinements to the application. These findings are then discussed in Chapter 7 with regard to the research questions.

Finally, this thesis concludes with Chapter 8. The primary contributions and findings are summarized, while also acknowledging the respective limitations. Furthermore, recommendations for future research are provided.

CHAPTER 2

State of the Art

Currently, there are various software solutions and services that support users when checking references in scientific literature. Although they do refer to the same subject area, the problems they solve can be fundamentally different. Therefore, the following chapters provide an overview of different applications and methods.

2.1 Plagiarism Checkers

Generally speaking, a plagiarism checker is a software which is utilized to cross-check text for duplicated content like quoted or paraphrased material and similarities in wording. This facilitates the verification if writing is original and correctly cited [12]. To do so, plagiarism checkers compare the text against an external collection and return all documents that contain passages similar above a certain threshold [13].

2.1.1 Turnitin Similarity

As an example, Turnitin Similarity is a widely used plagiarism checker which facilitates educators and students in the process of identifying plagiarism. Therefore, it compares text against a database of 47 billion internet pages, which includes both current and archived ones, 1.9 billion student papers and 190 million articles. As a result, Turnitin Similarity provides a similarity report (see Figure 2.1) which includes a similarity score and color coded results. Additionally, it provides users with a side-by-side comparison of sources and the functionality to exclude unnecessary matches [14].

2.2 Recite

Recite [15] is a web application which supports users in the process of cross-checking in text citations and the reference list. Therefore, it compares authors and dates, checks

2. STATE OF THE ART

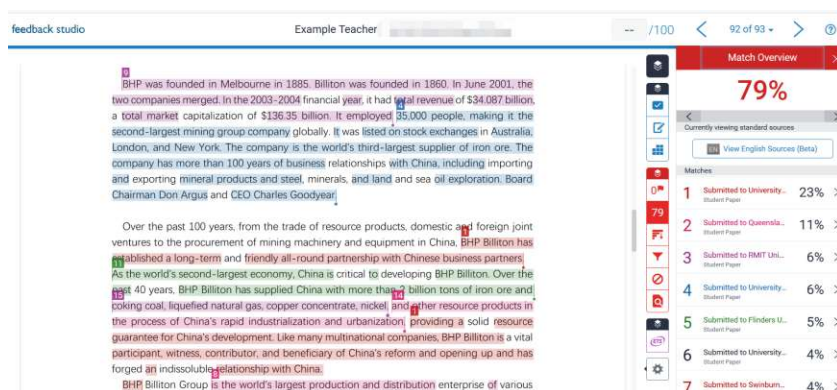


Figure 2.1: Example report of Turnitin Similarity

Source: <https://knowledgebase.xjtlu.edu.cn/article/how-to-check-the-similarity-report-in-turnitin-329.html>

missing citations and also reports stylistic errors related to referencing. As a result, it provides the user with a report that shows a reference by reference breakdown of all the in text references and a full list of all the references found in the paper (see Figure 2.2). Recite is designed for people writing academic articles, essays or books that require referencing. It's particularly suited for those adhering to APA or Harvard citation styles, as Recite is optimized for these formats.

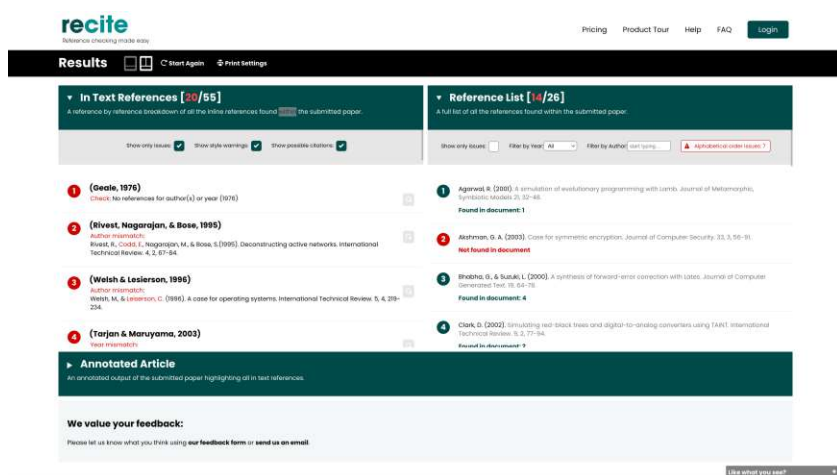


Figure 2.2: Resulting report of a reference analysis using Recite

2.3 Scribbr Citation Checker

The web application Scribbr Citation Checker [16] is an AI-based citation check for the APA citation style. It provides users with an interactive report which facilitates

2.4. Checking References manually using Academic Search Engines

checking and fixing found errors (see Figure 2.3). Therefore, it recognizes punctuation and capitalization errors, incorrect usage of et al., missing information and inconsistencies between in text citations and the reference list.

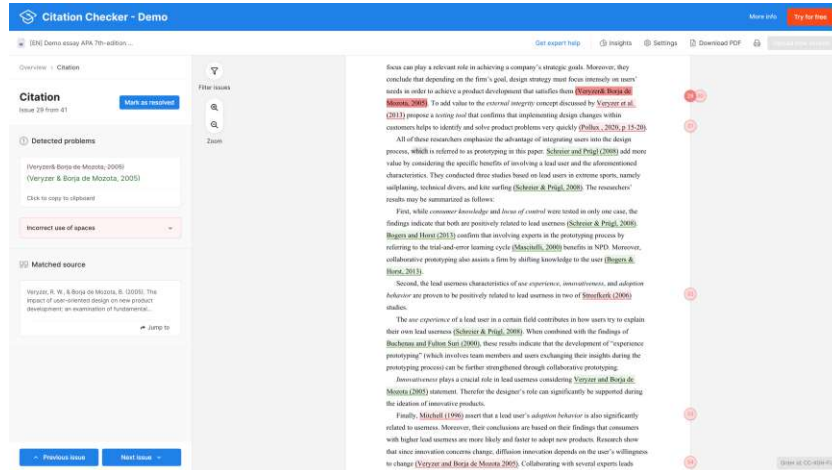


Figure 2.3: Resulting report of a reference analysis using Scribbr Citation Checker

2.4 Checking References manually using Academic Search Engines

A common method to check the existence of cited references is to do it manually by checking them on websites of cited journals or using Google Scholar [17]. By inserting the title in quotation marks, simple typographic errors in the name, volume, issue or page number can be prevented. Sources like web references can be checked by entering them in common search engines like Google or Bing, but also using internet archives like Wayback Machine [18][19]. The latter has 28+ years of web history accessible, providing users with 835 billion web pages, books, audio recordings etc. [20]. When checking references manually, users have to compare components of the searched reference with the components of the found references. Based on title, authors, year and other in the reference provided information it has to be determined whether the reference searched for and the reference found match.

2.5 Scite.ai

Scite.ai [21] is a web application that provides users with a reference analysis. It takes a PDF document as input and generates a report that facilitates the evaluation of the quality of used references 2.4. Therefore, it presents how the manuscript cites its own references, whether there are references with editorial notices and the amount of supports or contrasts for each reference. In order to achieve this, it utilizes machine learning for both detecting references and matching citation statements. Since this ability depends

2. STATE OF THE ART

upon the format of the reference, certain types that typically do not have DOI's will not be detected as references.

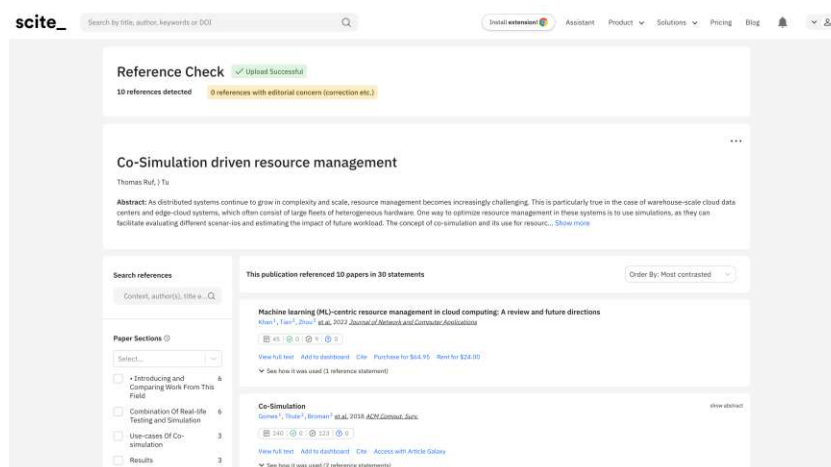


Figure 2.4: Resulting report of a reference analysis using the reference check of scite.ai

2.6 Trinkai.ai

Trinka.ai [22] is a web application that helps users classify their references by providing a reference analysis. It identifies different quality risks such as studies that have been recalled after publication, references that have not been peer-reviewed or citations that have not been frequently cited in other publications. Additionally, it facilitates the identification of outdated papers to ensure high relevance and can detect unintentional bias toward a journal. The references are validated against crossrefs publication repository and results are presented by a citation analysis score as well as detailed statistics (see Figure 2.5).

2.7 Comparison

The existing state-of-the-art approaches presented in previous chapters exhibit limitations in various aspects:

Plagiarism checkers like Turnitin Similarity do cross-check the text for duplicated content but do not assist in the checking whether the cited references do actually exist. Therefore, it could detect that text passages are from another source than referenced, but it would not be able to check if the literature reference is fabricated.

Recite and the Scribbr Citation Checker are pretty similar in their functionality. While they both provide the functionality to cross-check in text citations and the reference list, they do not give any information whether the references themselves do exist. Thus, fabricated references would not be recognized as incorrect.

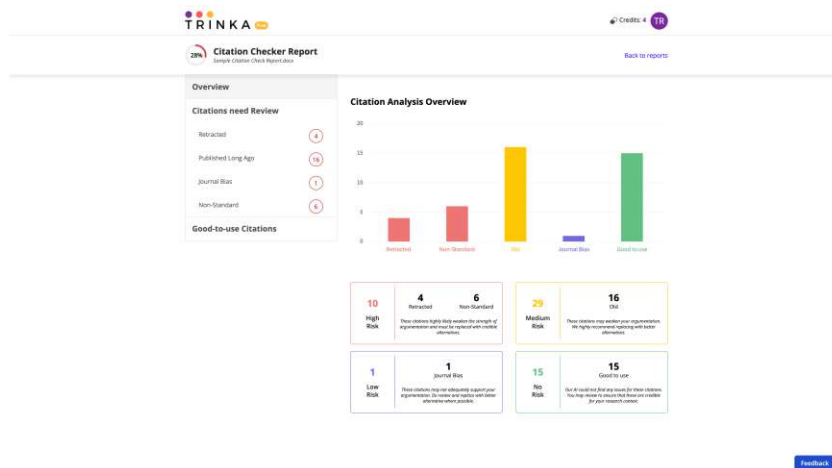


Figure 2.5: Resulting report of a reference analysis using the Trinka.ai citation checker.

Checking references manually by using academic search engines is in fact a working method. Since this has to be done sequentially reference by reference it is a time intensive task. Particularly as different components of the reference like title, authors and year have to be compared to found results by the user in order to determine whether the result found is the reference the user is looking for.

Scite.ai as well as Trinka.ai are powerful tools to analyze literature references of scientific work. While they both provide a detailed breakdown of how references have been used and whether they are good to use or not, they rely on the existence of the references. This leads to the fact that fabricated references, which therefore cannot be found in the databases of these applications are not analyzed and therefore not mentioned in the report at all. Subsequently, this distorts the results of the analysis in the case of fabricated references.

In conclusion there is no application that concentrates on the question whether a listed reference is fabricated or not.

CHAPTER 3

Concept

The concept is to develop an easily accessible web application that supports users in the process of verifying literature references. Therefore, it should provide an automated process to check multiple references simultaneously. It should then present the gained information by using informative statistics and a clear result structure. The process itself should mimic the strategy of checking references manually (see Chapter 2.4).

3.1 Features

The web application provides the user with a user interface that lets him insert one or multiple references in text format. Additionally, there is the option to configure the used academic search engines for reference checking, where users can multiselect between seven different options (see Section 3.1.1). After the analyzation process (see Section 3.2), where each reference gets a result for each search engine search and is classified as "match", "check match" or "no match" (see Section 3.2.3), the results are shown. This full reference analysis result consists on the one hand of general statistics relating to the full analysis and on the other hand of a list of elements, where each element represents one searched reference.

The statistics concerning the full reference analysis consists of three components. Firstly, the status of the full reference analysis presents the amount of references that could not be found, the number of references that need to be checked or that all reference have been found. Secondly, a pie chart shows the relation between the reference search results according to their status. Thirdly, a polar chart visualizes the amount of references that have been found for each search engine.

Each reference element supports the functionality to view detailed analysis information. Therefore, users can check each search engine search result, consisting of the closest found reference and its components like title, authors, DOI, year and the link to the source.

Additionally, each component has its own comparison status which displays the similarity to the searched reference.

3.1.1 Featured Academic Search Engines

Since certain academic search engines like IEEE Xplore or PubMed are rather area-specific when it comes to scientific literature, the developed prototype does support six different search engines:

1. Google Scholar
2. Semantic Scholar
3. Base
4. RefSeek
5. PubMed
6. IEEE Xplore

While this multi-search-engine strategy does cover a broader range of scientific fields, it also diminishes the impact of one of the search engine requests failing for any reason. Additionally, it gives users the possibility to adjust the used search engines for their needs.

The selection process was based on a comparison of results gained from a literature reference search test. To evaluate the coverage and therefore the power of each academic search engine, they were tested with 20 scientific literature references in total, randomly picked from ten scientific papers. These scientific papers were in the areas of physics, chemistry, biology, medicine, psychology, history, politics, mathematics, computer science and logic and should hence depict a holistic image of all scientific branches (see Table 3.1).

There were 18 different academic search engines [23][24][25] assessed. Each one underwent the checking process whether the search engine does find the scientific literature references by using their search functionality and the title. In order to determine if the obtained result matches the searched reference, components were compared manually (see Chapter 2.4). This led to a percentage value, which indicates the number of found references for each search engine (see Table 3.2).

Subsequently, the search engines with a hit rate above 50 percent were chosen: RefSeek, Google Scholar, Semantic Scholar, ResearchGate and BASE. Within the development phase, all of them were integrated in the reference checking process except for ResearchGate since it showed limiting factors (see Chapter 4.3).

Scientific Branch	Detailed Branch	Randomly picked Scientific Paper	Randomly picked Scientific Reference from this Paper
Natural Sciences	Physics	Simulation of Operating IV-VI Semiconductor...	Quantum cascade lasers operating from 1.2 to 1.6 THz
			Gas-phase databases for quantitative infrared spectroscopy
	Chemistry	Direct control of recombinant protein product...	Principles of bioprocess control
			Development of a high yielding e. coli periplasmic expression...
	Biology	Integrating cell-free fetal dna from amniotic...	Comparative Genomic Hybridization for Molecular Cytogenetic...
			Recommendations for reporting of secondary findings in clinical...
	Medicine	A 3D structured breast phantom with lesion...	Integration of 3D digital mammography with tomosynthesis for...
			Spectral Dependence of Glandular Tissue Dose in Screen-Film...
Social Sciences	Psychology	Die Rolle traditioneller Genderidentitäten...	Habitus und sozialer Raum: Zur Nutzung der Konzepte Pierre...
			Implicit ageism
	History	Die Teilung der Tschechoslowakei 1992 und...	Die unerträgliche Leichtigkeit der Trennung
			Juden in den 30er Jahren des 20. Jahrhunderts
	Politics	Die Systemkorruption in Rumänien	Perspectives on the Perception of Corruption
			Gradients of Corruption in Perceptions of American Public Life
Formal Sciences	Mathematics	A fixed-point theorem for Horn formula equations	Decidability of affine solution problems
	ComputerScience	A framework for execution-based model profiling	Untersuchungen über das Eliminationsproblem der mathematischen...
			Causally-connected and requirements-aware runtime models using...
	Logic	Automated induction by reflection	Process-aware Information Systems: Bridging People and...
			On the use of the constructive omega-rule within automated...
			Superposition with Structural Induction

Table 3.1: Collection of scientific references that were used to assess and compare academic search engines. (For complete reference list see appendix "List of References for Testing")

Academic Search Engine	Found References
RefSeek	95%
Google Scholar	90%
Semantic Scholar	80%
ResearchGate	65%
BASE	55%
Baidu Scholar	50%
Fatcat	50%
WorldWideScience	45%
Sciencegate	40%
Dandelon	15%
CORE	10%
OAIster	5%
CiteseerX	5%
ScienceResearch	0%
DataONE	0%
science.gov	0%
ERIC	0%
iSEEK Education	0%

Table 3.2: Result of found references in percentage for each academic search engine. Sorted in descending order from best to worst.

During the development of the prototype, the two search engines PubMed and IEEE Explore were also tested. Although they do not cover such a broad range of scientific fields compared to mentioned search engines, additional search engines can only improve results and therefore the functionality of the prototype, which was the deciding factor to keep them in the featured list of search engines.

3.1.2 Supported Citation Styles

There are multiple different citation styles in scientific literature, each having its own way on how the information is ordered and presented [26]. Supporting each citation style would lead to a great increase in workload, but does not necessarily impact the significance of insights gained from the development of the prototype, since tests can be done, limited to the supported citation styles. To still cover as many scientific literature references as possible, three of the most popular citation styles are supported: APA, IEEE and MLA [26][27][28].

3.2 Functionality

The process itself starts with the user input of one or multiple references as text via the web application. When being submitted, each reference is transferred to the backend application. There, the first step is to classify the reference according to its citation style. This step is essential in order to obtain accurate results when splitting the reference in its components like title, authors, year and DOI in the next step. Subsequently, the following steps are executed for each selected search engine. First, the reference is being searched on the search engine. If the search does not produce a result, a corresponding “not found” result is returned. Otherwise, the most promising result is extracted and split into its components. Next, the components of the searched reference and the components of the search engine result are compared. According to the outcome of the comparison, the result is marked as “found”, “needs to be checked” or “not found”.

To display the procedure more clearly, the diagram (see Figure 3.1) shows the process for a single reference. In the case of multiple references, the same process is executed for each reference.

Since the classification of the citation style, the splitting of the reference and the comparison of the different components of the reference do play a key role in the process, these subprocesses are explained in detail in the following subsections.

3.2.1 Citation Style Classification

The citation style classification of references is a significant part of the reference checking process. It ensures that the input reference is in a supported style, namely APA, IEEE or MLA. This process happens in the backend to guarantee that required components of the reference do exist for further processing.

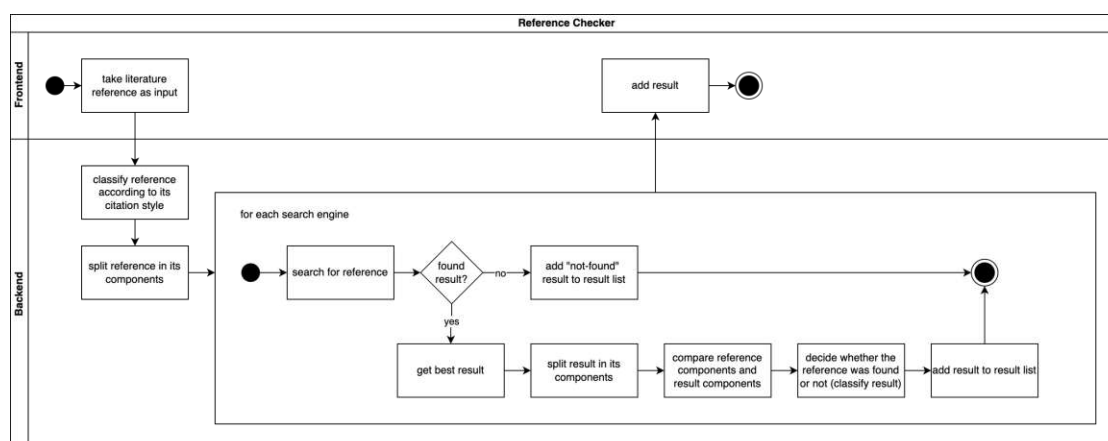


Figure 3.1: Process of checking a single reference from input to result.

In case of an input reference that does not match any supported citation style, it is marked as "unsupported". This means that it has to undergo a different checking procedure, whereas the probability of incorrect results is higher (see Chapter 3.2.3).

3.2.2 Component Splitting

In order to be able to check whether the searched reference was found or not, a strategy to compare it with the search results is needed. This evaluation process, described in the next chapter, relies on the splitting of the searched reference in its components, since the parts of a reference are compared individually. Therefore, each reference is split into authors, title, year and if existing, the DOI. These components are utilized, since they are part of most scientific literature references. Additionally, the majority of the supported search engines (see Chapter 3.1.1) do present these components on search results, which is a basic requirement for significant comparison results.

References that were marked as "unsupported" in the citation style classification process, are not split into their components since their structure is not clear.

3.2.3 Result Classification

The result classification happens on three levels (see Figure 3.2). Going bottom up, on the first level, each component of the reference is compared to its equivalent from the search engine result. In this comparison process, a status is assigned to the authors component, the title component, the year component and if existent, the DOI component. This status is either "MATCH", "CHECK_MATCH" or "NO_MATCH" and is determined based on a calculated similarity score (for technical details see Chapter 4.2.5).

Subsequently, on the second level, based on the four component statuses from level one, a holistic comparison status is determined for the whole search engine result. Equivalent

to the first level, "MATCH", "CHECK_MATCH" or "NO_MATCH" is assigned (for technical details see Chapter 4.2.6).

On the third and last level, the final status of the searched reference is determined. If at least one of all search engine results from level two has the status "MATCH", this status is assigned to the searched reference and is therefore labelled as "Found" in the results view. The same applies for the status "CHECK_MATCH" resulting in "Needs to be checked" in the results view. It is important to note, that "MATCH" has a higher priority than "CHECK_MATCH" which means that a searched reference with two search engine results with the statuses "MATCH" and "CHECK_MATCH" would lead to the status "MATCH" for the searched reference. In the case of all individual search engine results from level two having the status "NO_MATCH", this status is also assigned to the searched reference and labelled as "Not found" in the results view.

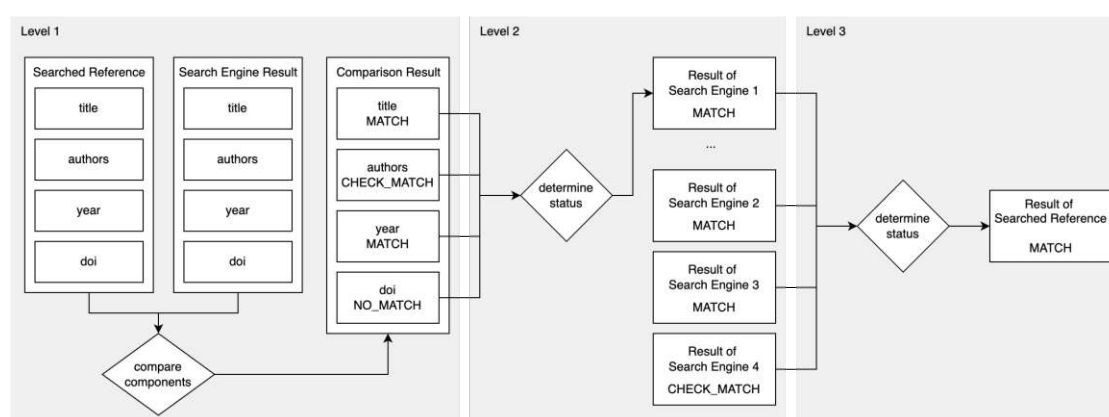


Figure 3.2: Process of classifying a searched reference.

References that were marked as "unsupported" in the citation style classification process and therefore could not be split into their components, have to undergo a different procedure to classify results. Whereas level two and three of the result classification process remain unchanged, on the first level, it is checked for each component of the search engine result whether the full reference does contain it or not. Since the probability of imprecise results is higher due to this processes one-sidedness (a reference could for example contain multiple additional authors that are not contained in the authorlist of the search engine result but still result in 100 percent similarity since it still contains all authors of the search engine result), references with style "unsupported" are marked as such in the results view together with a note to inform users.

3.3 Expected Benefits

Relying on the described automation process in previous chapters, the functionality of the prototype could lead to a substantial reduction in time and effort when checking literature references of scientific papers. This is achieved by enabling users to check

multiple references at once, but also by providing them with clear results, which offers statistics, comparison results and direct links to the found scientific papers.

Additionally, the system does not rely on a single academic search engine, which could be a single point of failure, but on six different ones, thus covering a broader range of data. Likewise, it offers users the option to configure their individual search engine settings.

As mentioned in the introduction, the integration of the functionality of this prototype could eventually combat the increasing number of scientific works with incorrect, more specifically non-existing references, possibly written by generative AI. By making it available to the broad mass through a web application, the reference checking process could be easily accessible and more efficient, and therefore more likely to be implemented.

CHAPTER 4

Development

The following sections provide an insight in the architecture and used technologies of the developed prototype. Furthermore, technical details of the implementation concerning the style classification, the component splitting, the similarity comparison of references and the status classification of results are presented. Lastly, limitations that come with the chosen technologies are explained.

4.1 Architecture

Since it was planned to make the application as easily accessible as possible, the frontend of the application consists of a web application. This enables users to access the application from every device that is equipped with a browser and has an active internet connectivity. The client-side web application communicates with the backend via a restful API. This facilitates a potential connection of other systems to the service. The backend itself consists of a Web Service, that holds the necessary logic for processing incoming requests and sending responses. To do so, it utilizes either academic search engine websites or APIs to gather data concerning scientific literature references (see Figure 4.1).

For the current use of the application, there is no need for a database on the backend side. This is the case, since dedicated user accounts are not supported. This means in further consequence that no user-related data is being saved server-side.

4.2 Technologies

The following chapters provide a detailed insight in the used frameworks and libraries for the development of the Frontend and Backend of the application. Subsequently, technologies and strategic approaches for web scraping, style classification and component splitting, similarity comparison and status classification are presented.

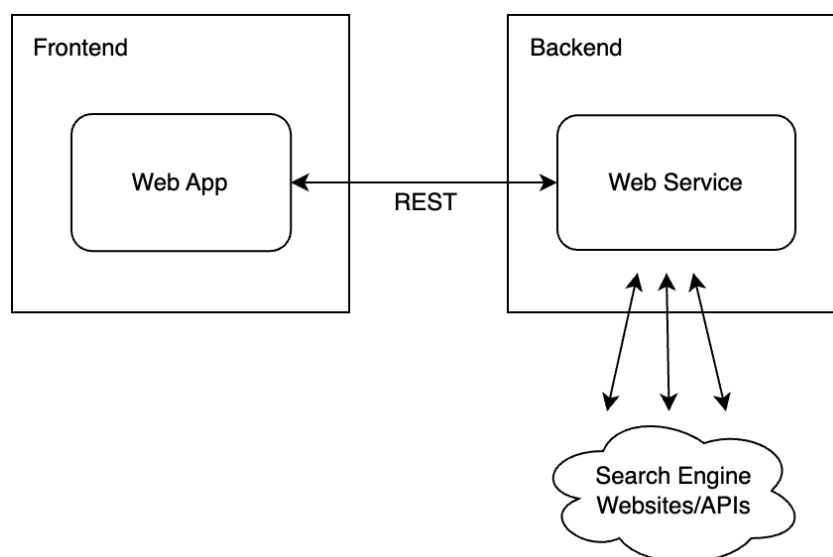


Figure 4.1: The technical architecture of the application.

4.2.1 Frontend

The used frontend framework for the development of the web application is Angular. It supports both small and large applications, thus ensures compatibility if the application may grow in the future [29]. Additionally, it offers two-way data binding, dependency injection and routing [30]. This facilitates the development of the web application tremendously.

Additionally, Angular supports component libraries. In this case, the used library for the development of the web application is PrimeNG. It provides a wide range of UI components, ranging from forms, buttons, overlays and menus to charts [31]. This ensures that the application has a consistent look and feel for users, as well as promotes efficiency and scalability.

4.2.2 Backend

The backend's functionality is built upon the interaction of various frameworks and libraries, as detailed in the upcoming chapters.

Spring

When developing the application's backend, Spring is leveraged, the world's most popular Java framework known for its emphasis on speed, simplicity and productivity. Spring streamlines Java programming, making it faster, easier and safer [32]. Concerning the developed application, it serves as the base framework, integrating the following technologies.

Selenium

The testing tool Selenium plays a key role when it comes to automating browser operations. To do so, Selenium WebDriver provides a collection of language specific bindings to drive a browser [33]. Selenium is needed in order to firstly run a headless Chromium browser server-side, secondly call the search engine websites and thirdly scrape the resulting search data.

Ricecode

Ricecode is a java library that provides the functionality to calculate similarity scores between strings. Therefore, it supports four different algorithms: the Jaro distance algorithm, the Jaro-Winkler distance algorithm, the Levenshtein distance algorithm and the Sørensen–Dice coefficient algorithm [34]. This functionality plays a key role when it comes to the similarity comparison of components (see Chapter 4.2.5).

Jackson

Jackson is a Java library which enables parsing JSON objects [35]. It facilitates parsing the JSON responses from the search engine APIs, which is needed to further process the result data.

Lombok

The Java library Lombok simplifies the development by replacing boilerplate code with easy-to-use annotations [36]. It can automatically generate getter and setter methods, constructors and enable the use of builders [37].

4.2.3 Web Scraping vs. API

As depicted in previous chapters, the functionality of checking or verifying scientific references relies on the use of different academic search engines. Therefore, a technology is needed which facilitates the exchange of data between the search engine and the applications' backend. Usually this would be done using an application programming interface (API), which enables applications to communicate with each other, but also share functions and market data to business partners or third parties [38]. Since only one of the six search engines, namely Semantic Scholar, provide an API, a different procedure called Web Scraping is being utilized.

API

The academic search engine Semantic Scholar provides an API called Academic Graph API (1.0) [39] that makes it comparatively easy to gather data concerning papers and authors. It provides an endpoint that returns a list of matching papers and configurable detailed information when requested with a search title. Compared to Web Scraping, this

not only results in faster processing times, but also in well-structured data that increases the stability and quality of results.

Web Scraping

Web scraping is the use of software to automatically extract data from websites. This process is divided into three stages. Firstly the fetching stage, where the required website is accessed, secondly the extraction stage where the needed data is extracted from the HTML code and thirdly the transformation stage where the gathered data is converted in a structured format [40].

When it comes to performing a literature search using academic search engines the manual approach includes the following steps:

1. Open the search engine website using a browser.
2. Insert the title of the reference to search for in the search input field.
3. Submit the search request by clicking on the submit button.
4. The search engine processes the request, routes to the result page and shows the findings.

In comparison, the same process using web scraping follows an equal strategy but in an optimized form. In the developed applications' backend, the web scraping process starts by taking the URL that is called when a search request is performed. Subsequently, the title of the reference to search for is URL-encoded and added to the URL as a URL-param. Thus, the resulting URL can be opened using the headless browser, which directly processes the request and shows the result page. Therefore, the first three steps of the manual process are skipped, which reduces loading times and unnecessary operations. Finally, the needed data is extracted from the HTML of the result page using predefined classnames and the structure of elements. This data is then brought in a structured format for further processing.

4.2.4 Style Classification and Component Splitting of References

As already mentioned in a previous chapter (see Chapter 3.2.1), the citation style classification of references ensures the compatibility of the input references and is the first step of the reference component splitting process. In order to determine whether a scientific reference matches APA, IEEE or MLA citation style, regular expression matching is used.

Regular expressions are patterns of characters that describe a set of strings. Thus, they facilitate finding, displaying or modifying occurrences of patterns in an input sequence [41]. Since it allows testing whether a string matches a specific syntactic form, it meets

the requirements to be used for style classification. Additionally, the functionality to define named capturing groups facilitates extracting certain parts of the reference, which is used for component splitting [42].

The fact that citation styles have different syntactical forms depending on the source makes it difficult to create regular expressions that uniquely match a particular citation style. IEEE for instance, has three different ways to display authors depending on the number of authors [43]. This leads to the need for multiple regular expressions for one citation style. Each regular expression has to be restrictive enough to match the citation style, but not too restrictive to also match certain edge cases. A meaningful example would be international author names, which can include multiple first names or multipart surnames. An exact match is particularly important to ensure the correct splitting of the reference into its components. Since the citation style of scientific references is classified iteratively by testing them against one pattern after the other, this could result in an incorrect classification and in further consequence incorrect split components. To prevent this, the application implements a pre-classification strategy. Firstly, the citation style is determined by using one simple regular expression for each citation style, which tests for a unique syntactic characteristic of this style. Secondly, it is tested against all regular expressions of this citation style, to find its exact syntactic match. If successful, the reference can be split into its components, namely authors, title, year and DOI using named capturing groups.

4.2.5 Similarity Comparison of Components

The component similarity comparison is the basis for determining whether a reference has been found and can be verified. Therefore, a score between zero and 100 is assigned, where zero indicates no similarity and 100 perfect similarity. The strategy for allocating this score varies from component to component. Starting with the authors, the two list of author names are compared. If for example 4 out of 5 authors do match, this results in a similarity score of 80. For comparing titles, the Levenshtein algorithm is utilized. Based on the Levenshtein distance, which is the number of single character edits like insertions, deletions or substitutions that are needed to transform one string into the other, a value between zero and one is computed which multiplied with 100 results again in a value between 0 and 100 [44]. While this algorithm is used for the similarity comparison of titles of all search engine results, the search engine RefSeek is an exception. Since RefSeek most of the time truncates the titles in results, the used algorithm has to adapt to this. In this case, the Jaro Winkler Algorithm is used, which prioritizes the start of the string when determining the similarity score [45]. Regarding the DOI and the year, they are simply tested for equality. Thus, they either get the similarity score of 100 if they are a perfect match or zero if they are not.

In case of an input reference that did not match any supported citation style and therefore is marked as "unsupported" the similarity comparison works differently. Starting with the title, the title of the search engine result is split into individual words. Subsequently, for each word it is tested, whether the full input reference (not only the title, since the

input reference could not be split up into its components) does contain this word or not. If for example, nine out of ten words of the search engine result title are contained, this results in a similarity score of 90. The same procedure is performed for authors, the DOI and the year, whereas there is no need to split up the DOI and the year.

4.2.6 Status Classification

While the previous chapter showed how components of a scientific reference are compared and as a result get a value between zero and 100, the status classification then assigns one of the statuses "MATCH", "CHECK_MATCH" or "NO_MATCH" to each component based on its similarity value. Starting with the title component, similarity scores greater than 95 get the status "MATCH". The status "MATCH" isn't limited to a score of 100 since titles of scientific papers are sometimes displayed by search engines with missing punctuation or differing special characters. Similarity scores between 95 and 80 get the status "CHECK_MATCH". Title components with a score less than or equal to 80 get the status "NO_MATCH".

The authors component gets the status "MATCH" if it has a similarity score of 100, "CHECK_MATCH" if it is between 100 and 80, and "NO_MATCH" if it has a value less than or equal to 80. Additionally, there is the case that the authors of the searched reference does contain et al. but the list of authors from the search engine result doesn't. If in this case, each author from the list of authors with et al. is contained in the list of authors without et al., it gets the status "MATCH". If some but not all are contained, it gets the status "CHECK_MATCH" and if none is contained "NO_MATCH".

Since DOI's are digital identifiers, that are unique and resolvable using the DOI-Foundations proxy server, there is no room for error [46]. Therefore, the DOI component gets the status "MATCH" if it has a similarity score of 100 and "NO_MATCH" in every other case. The same applies to the year component.

After each component has its status, the status for the entire search engine result is determined based on them. In this process, each component has a different weighting. A matching title, for instance, has a higher priority than a matching year. This is because years can differ due to different dates when, for example, an article was first published online and then in a print issue [47]. There are only two cases where the status "MATCH" is assigned: Either if the title component has the status "MATCH" and not all other components have the status "NO_MATCH" or if the title has the status "CHECK_MATCH" and both the authors as well as the year have the status "MATCH". "CHECK_MATCH" is assigned either if the title has the status "MATCH" but all other components have the status "NO_MATCH", if the title has the status "CHECK_MATCH" and not both the authors and the year have the status "MATCH", or if the title has the status "NO_MATCH" but both the authors and the year have the status "MATCH".

4.3 Limitations

There are different circumstances that eventually can or already do limit the functionality of the prototype. Starting with the limiting factors of web scraping. The first one being the vulnerability to structural frontend changes. Since the functionality of web scrapers do rely on a specific HTML structure, even small changes can lead to malfunctioning. Another problem lies in implemented anti-scraping techniques like captchas or IP blocking, which can both limit the automated access to search engine websites. While scraping public data is generally permissible, it is needed to comply with the website's terms of service, copyrights and privacy policies [48].

The API from Semantic Scholar does provide endpoints that are available to the public without authentication. However, an unauthenticated IP is limited to one request per second and access can be restricted in the case of traffic overload. Although the prototype uses an authenticated IP, which provides access to higher rate limits, this could lead to failed requests or longer response times [49].

Additionally, data quality does limit the functionality of the prototype. The scraped data which is the foundation of the comparison can only be as good as it is provided by the search engines. Particularly the search engines BASE, but even more RefSeek do lack in terms of consistency in formatting and structure of the provided data.

Although the prototype does support three different citation styles, namely APA, IEEE and MLA, during the development and testing it has been noticed that these styles are not always syntactical equal upon usage. Therefore, there can be citation style versions of for example IEEE, which follow the fundamental syntactical structure of IEEE, but with little deviations based on the scientific field or university. This can result in faulty style classification and component splitting.

References that are not in a supported citation style are still checked and classified, but the probability of imprecise results is higher due to the processes one-sidedness (see Chapter 3.2.3 and 4.2.5). Additionally, it is harder to get good results from search engines for these references since the search input is not the title of the reference but the whole reference.



Design and Usability

This chapter starts with the description of the initial development phase of design and usability features. Subsequently, the following sections provide an overview of the final structural design and usability of the application, showcasing its user interface, usability features and associated design decisions.

5.1 Development Phase

Based on the findings of the theoretical analysis and the concept of the application, the user interfaces' development phase consisted of wireframing followed by final mockups. The following sections provide an insight to these processes.

5.1.1 Wireframing

The first phase of the user interface and user experience design started with wireframing. Therefore low fidelity wireframes were drawn for each planned view of the application using the software Procreate.

Starting with the search view, which represents the initial view of the application, the first wireframe design featured three main components. First a header consisting of a logo and the applications title, second a column on the left side showing user information as well as past searches, and third a centered card which should provide the input functionality to either drag and drop a file or select a file and start the checking process (see Figure 5.1).

This initial design was discarded due to two reasons. Firstly, the functionality of having dedicated user accounts for storing data like the search history and past results is a reasonable feature, but has no real benefit for the prototypes' evaluation of the automated process itself. Secondly, providing the input functionality as a file upload would be too restrictive compared to text input.

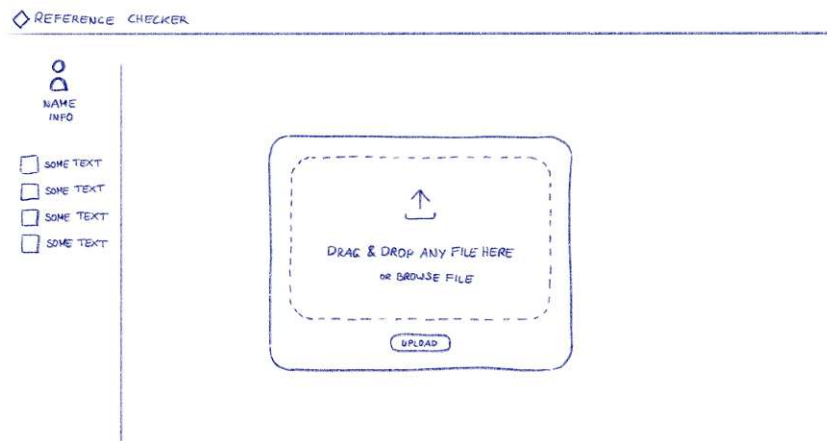


Figure 5.1: The initial wireframe design of the search view

Consequently, the second iteration of the wireframe for the search view saw the removal of user information and an update to the input card. The latter now featured text fields for the pasting of references, which can be added or removed dynamically (see Figure 5.2).

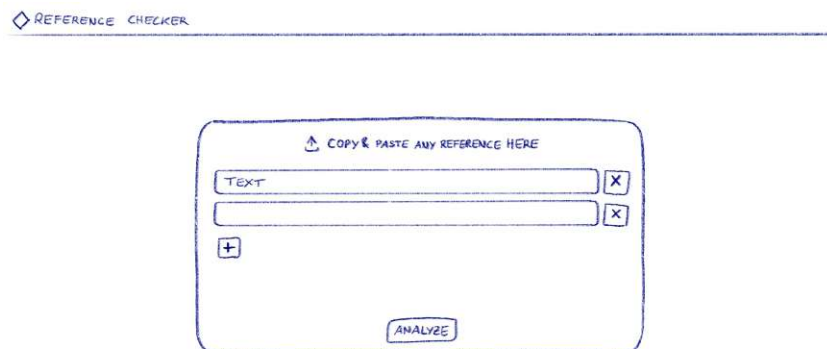


Figure 5.2: The final wireframe design of the search view

The initial and final wireframe designs of the results view were both divided into two columns. The left-hand column provided diagrams to facilitate the analysis of the results, while the right-hand column displayed the individual results for each searched reference. The initial wireframe design featured the latter as a table, which proved inadequate for displaying the information of a reference search result due to the length of individual components (see Figure 5.3). Consequently, this was updated to cards in the final wireframe (see Figure 5.4). Furthermore, a search bar for filtering and a status overview

were added.

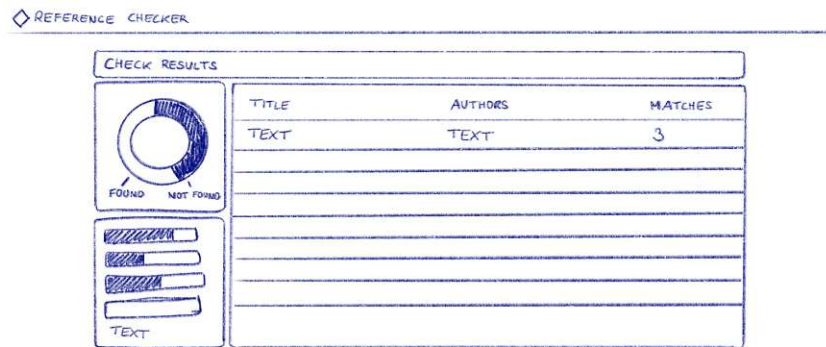


Figure 5.3: The initial wireframe design of the results view

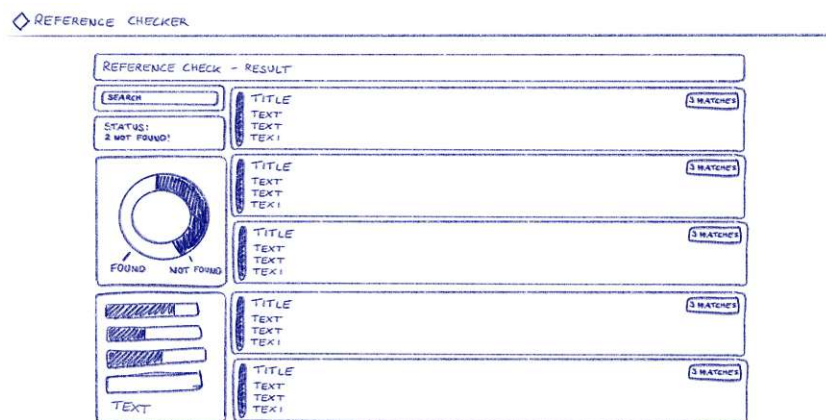


Figure 5.4: The final wireframe design of the results view

In order to provide additional information to each search reference result, the initial wireframe design of the details view showed the results of each search engine. Consequently, each displayed the title of the corresponding search engine and the degree of similarity of the components compared to the searched references' components (see Figure 5.5). While the titles' similarity was presented in percent, the others were limited to symbols. As the initial presentation happened to be too unclear due to the simultaneous display of all search engine results, the final wireframe displayed search engines as clickable tabs, thereby enhancing both readability and usability (see Figure 5.6). Furthermore, symbols next to the search engine names indicated the result status. Finally, the component similarity scores were modified to percentages, and a link to the identified result was added.

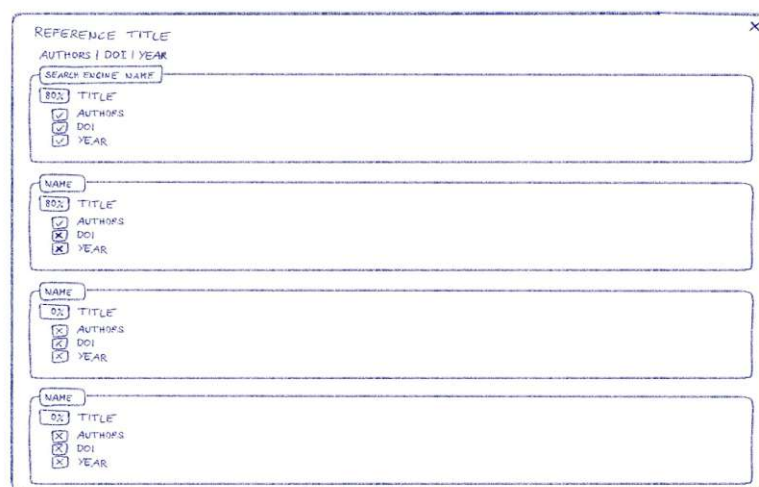


Figure 5.5: The initial wireframe design of the details view.



Figure 5.6: The final wireframe design of the details view.

5.1.2 Mockups

Based on the wireframes of the first phase of the user interface and user experience design, high fidelity mockups were developed for the search view (see Figure 5.7), the results view (see Figure 5.8), and the details view (see Figure 5.9). This was done using the online version of the software Figma.

While the wireframes focused on the design of the various components and their arrangement within each view, these mockups additionally presented fonts, colors, and design patterns that closely resemble the final structure and implementation of the application and are further discussed in the following chapter.

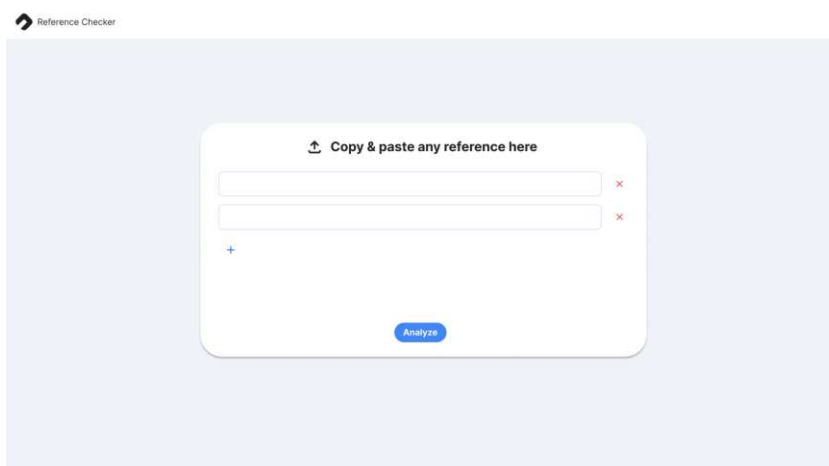


Figure 5.7: The final mockup of the search view

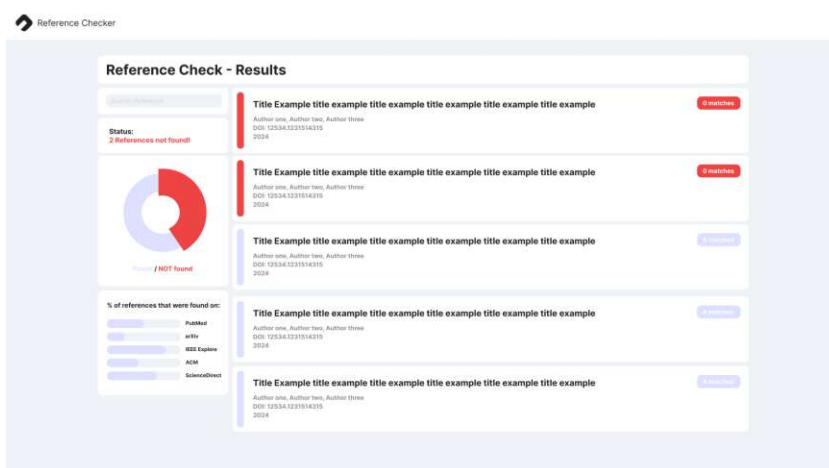


Figure 5.8: The final mockup of the results view

5.2 Structure

The application consists of two views, the search view and the results view. The search view serves as the starting point for the application, allowing users to insert references that need to be checked and providing configuration settings for the checking process itself. The results view then presents the results obtained, with additional statistics, filtering and search options.

5.2.1 Search View

Each reference check starts at the search view page (see Figure 5.10). Therefore, it provides users with the functionality to insert references. Whereas the view starts with a

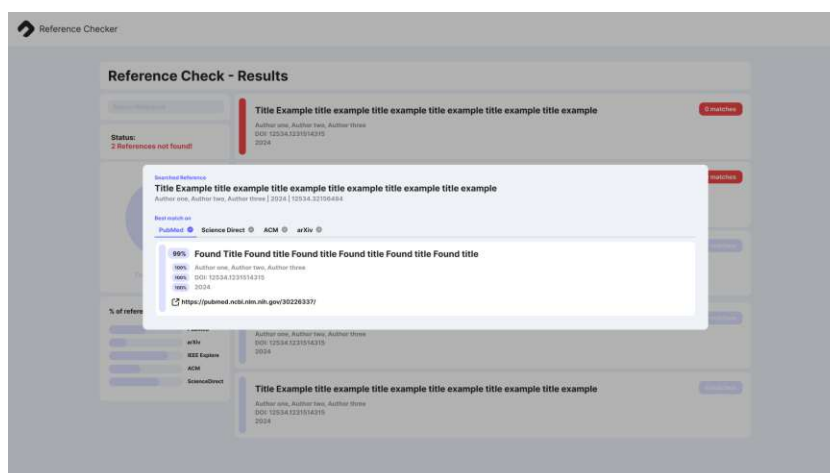


Figure 5.9: The final mockup of the details view

single input field, more can be added by pressing the dedicated button. Input fields can also be removed individually or all at once.

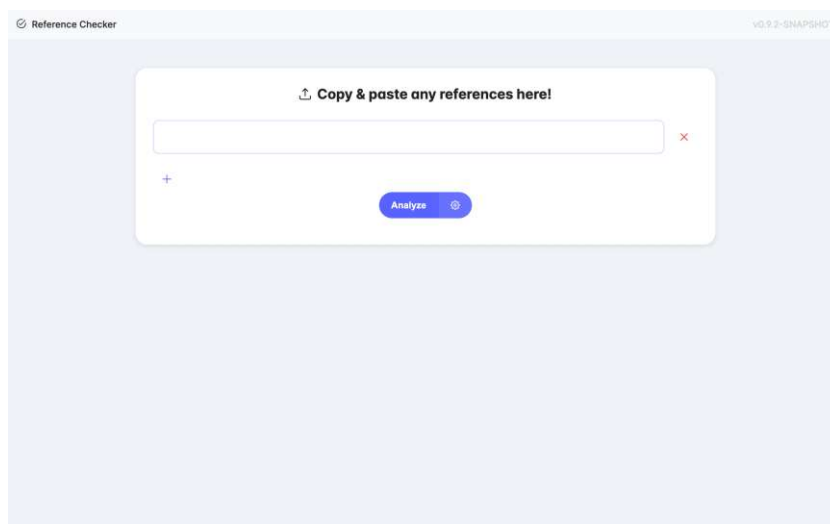


Figure 5.10: Search View Page

Reference Input

References can either be pasted one at a time, or multiple references simultaneously. Line breaks are removed automatically. If multiple references are copied from a paper, for example, and pasted into a single input field, the application detects that multiple references are present and hence provides the user with the option to split them up (see Figure 5.11). This splits the single input field with multiple references into individual input fields, each holding exactly one reference (see Figure 5.11). In the unlikely case

that a single reference in a certain format is confused for multiple references, the user has the option to reformat it automatically. In many cases, scientific literature does number its references in the reference list. In this case, numbering is removed automatically, which improves both usability and speed. In the case of input fields that contain an incorrectly formatted reference or multiple references that are not split up, the "Analyze" button which starts the reference check is disabled. This prevents incorrect results.

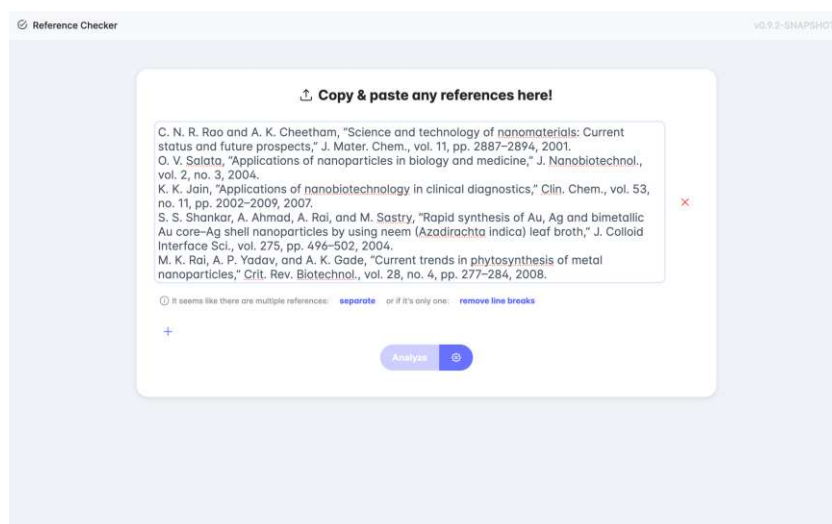


Figure 5.11: Multiple references pasted in a single input field

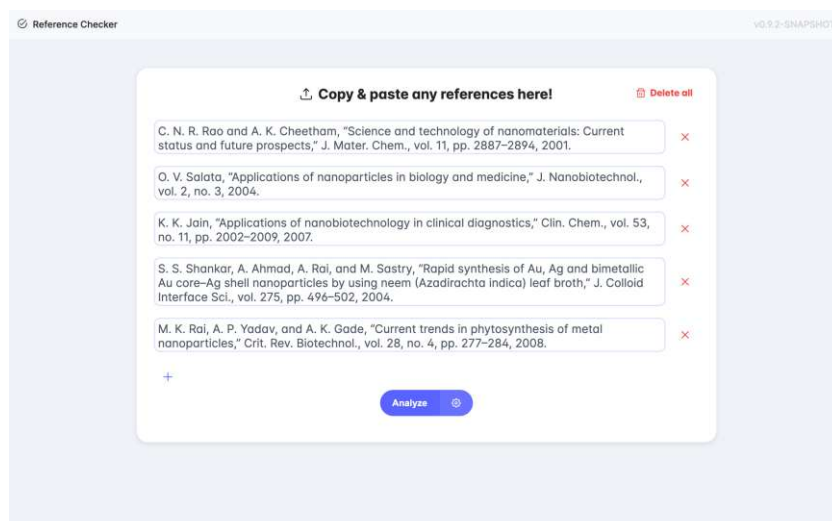


Figure 5.12: Multiple references split into multiple input fields

Search Engine Settings

By clicking the "Settings" button next to the "Analyze" button, users have the option to adjust their search engine settings (see Figure 5.13). Hence, each of the six search engines can either be activated or deactivated. The chosen settings are saved on the client side in the local storage of the user's browser. This way, the state of selected search engines is kept when the user closes and reopens the application.

An additional setting called "parallel requests" improves the speed of the reference check dramatically if activated. Due to request limitations (see Chapter 4.3) this restricts the supported search engines to Semantic Scholar and therefore deactivates all other search engines (see Figure 5.13). This "parallel requests" setting is a feature that acts for demonstration purposes for the expert interviews (see Chapter 6) to show the application's possibilities regarding speed if all search engines would provide dedicated APIs.

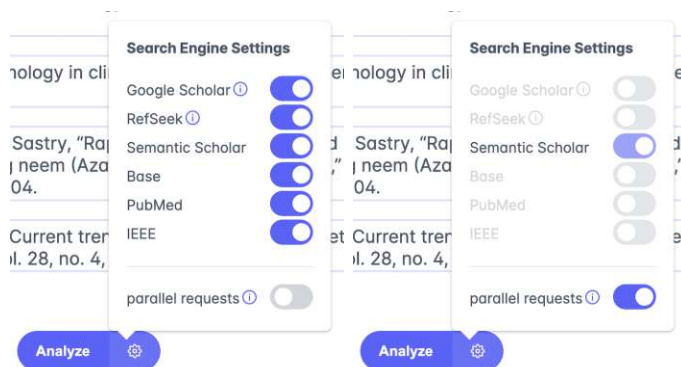


Figure 5.13: Overview of the Search Engine Settings with parallel requests deactivated and activated

Loading Screen

After pressing the "Analyze" button, a loading screen is shown and the reference check procedure in the backend is started. To inform users about the current status of the check, a progress bar is displayed which shows the amount of finished reference checks (see Figure 5.14).

5.2.2 Results View

Subsequently, the results view displays the results obtained from the reference check (see Figure 5.15). It is divided into two columns. The left column holds options to filter the result list of references either by a search bar or result filters, the resulting status of the reference check and statistics. The right column displays a list of all searched references depicted as cards with additional information concerning their status. This list of cards is sorted according to their importance, therefore searched references that have the status "not found" are shown on top, followed by references with the resulting status "needs to

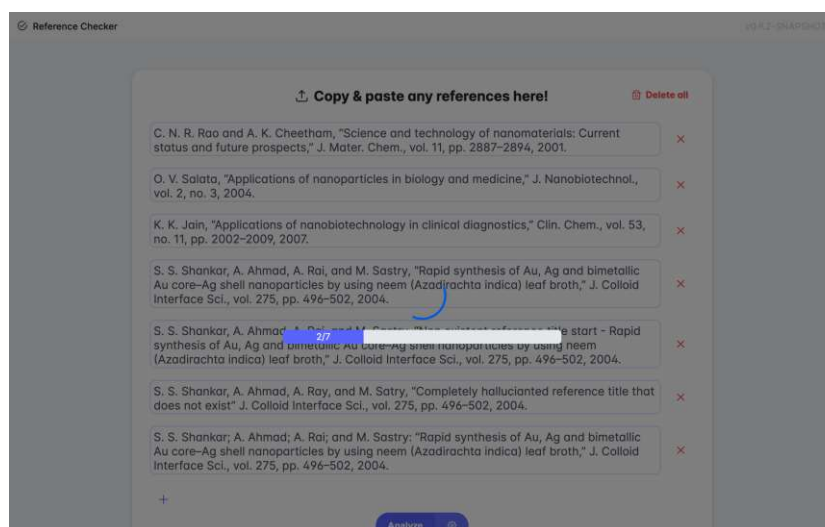


Figure 5.14: Loading screen showing the progress of a reference check with 7 references

be checked” and “found”. If there are references, which are in an unsupported format, an information alert is shown on top which informs users about supported citation styles. The “new check” button in the top right corner routes back to the search view, which enables users to start with a new reference check.

Search and Filtering Controls

The search and filter controls facilitate the analysis of the reference checks results. By using the search field, users can search the titles of the references. Consequently, only those results are displayed that include the searched string in their title. This facilitates the search for a specific reference check result if multiple references had been checked.

The filter options allow users to filter the reference check results to their needs (see Figure 5.16). Depending on the results statuses, it can be selected to either show or hide reference check results with the status “Not found”, “Needs to be checked” and “Found” individually. Additionally, there’s the option to only show reference check results where the searched reference is in an unsupported format by selecting “Only Unsupported”.

Status and Statistics

The application offers a status overview and diagrams to facilitate the preliminary evaluation of the reference check result. The status panel either informs about the amount of references that were “not found” or “need to be checked” or that all references have been found (see Figure 5.17). The donut chart also illustrates the amount of references with their dedicated status, but its visual representation facilitates the recognition of ratios (see Figure 5.17). By hovering over the diagrams sectors, exact numbers are shown.

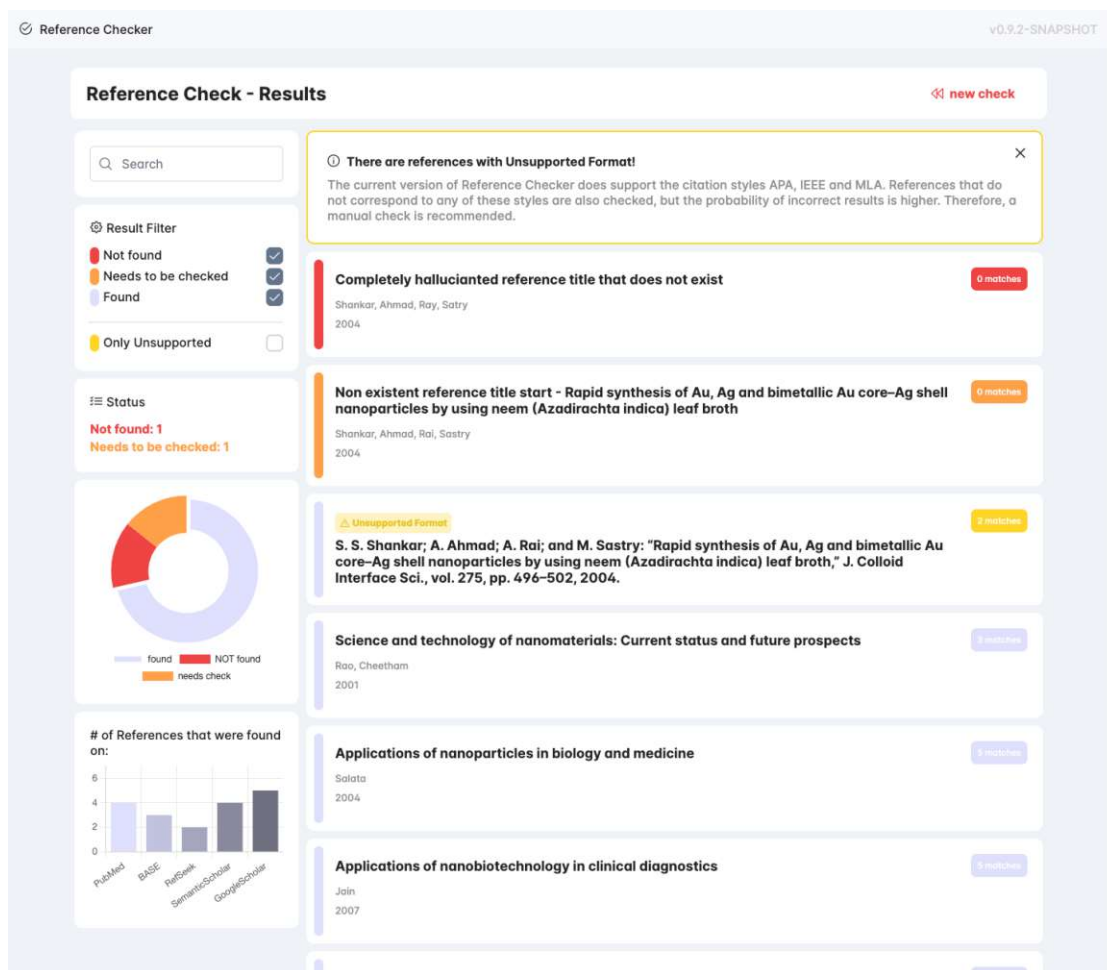


Figure 5.15: The results view of the application showing the outcome of a reference check with seven references

The bar chart provides an overview of the number of references that were found on each search engine (see Figure 5.18). A key observation from the chart is that the results are either concentrated in a single search engine or evenly spread over all.

Presentation of References

All searched references are displayed as cards. On the left side of the card, the colored vertical bar displays the resulting status of the check for this reference. Light blue represents "found" (see Figure 5.19), orange "needs to be checked" (see Figure 5.20) and red "not found" (see Figure 5.21). Since red and orange do stand out from the rest of the interface colors, it does help users to easily recognize references that could not be found or need to be checked. Likewise, the tag in the top right corner of the card has the same color as the bar but does also present, by how many search engines the reference has

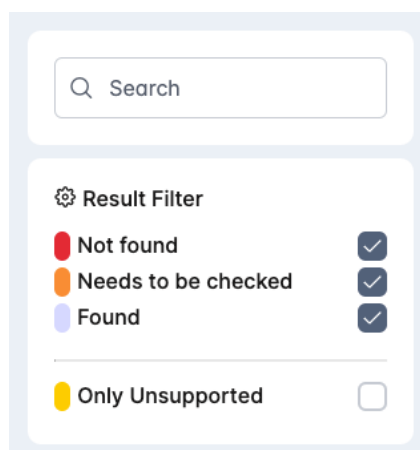


Figure 5.16: The search and filter controls in the results view

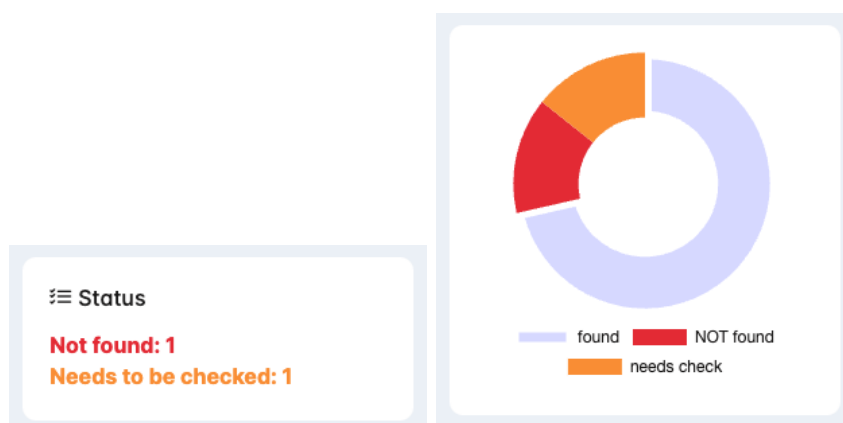


Figure 5.17: The status overview and donut chart showing the resulting status of the reference check

been found. Additionally, the card does show the components of the searched reference. Starting with the title in bold type, the authors, the DOI (if existent) and the year.

Searched references that are in an unsupported format are displayed similarly. Since the reference could not be split up in the reference check process, it is shown as a whole. In order to indicate to the user that this reference is in an unsupported format, the card displays a yellow tag in two locations. The tag in the top left corner displays the text "Unsupported Format," while the tag in the top right corner, which indicates the number of search engines in which the reference has been found, has a yellow background (see Figure 5.22).

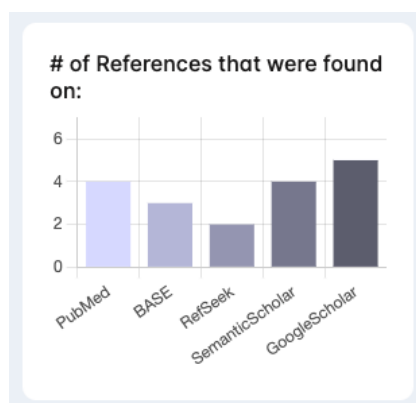


Figure 5.18: Bar chart showing the number of references that were found on each search engine

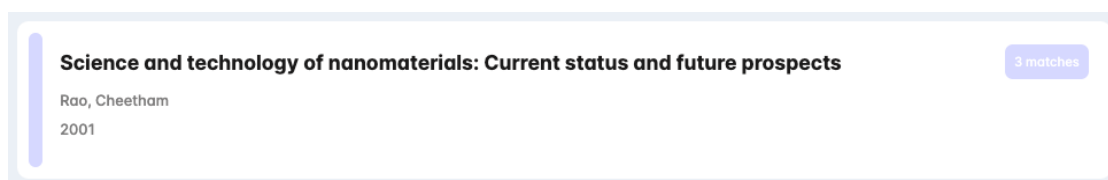


Figure 5.19: Result card of a reference with the resulting status "Found"

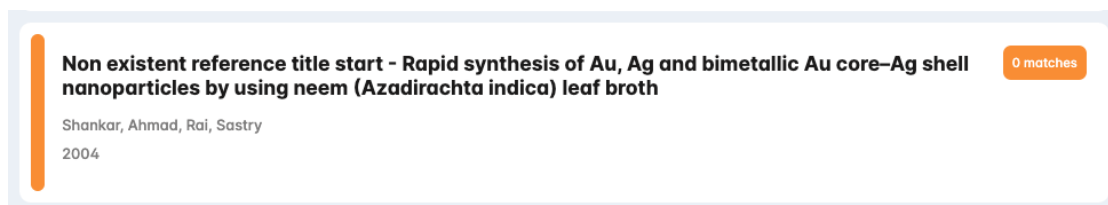


Figure 5.20: Result card of a reference with the resulting status "Needs to be checked"

Detail View

When clicking on one of the result cards in the right column of the results view, the dedicated detail view is shown in a modal window (see Figure 5.23). On the top half, the searched reference is displayed, split up into its components. The title of the reference is shown first, followed by the author names, the DOI (if existent) and the year.

On the bottom half of the detail view, the results of the selected search engines are shown. Sorted by their result status from “found”, “needs to be checked” to “not found”, the results for each tab are accessible via a simple click. Additionally, the icon near the search engine name represents the result status. The check mark for “found”, the exclamation mark for “needs to be checked” and the cross for “not found”. Therefore, users can recognize the resulting status of each search engine result without having to click on it.

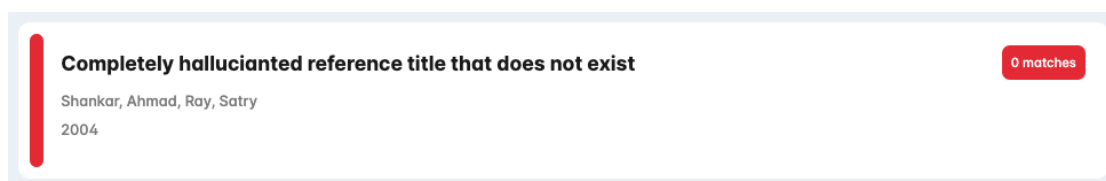


Figure 5.21: Result card of a reference with the resulting status "Not found"

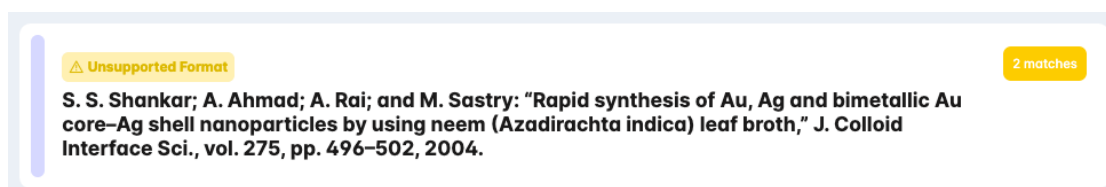


Figure 5.22: Result card of a reference in an unsupported format with the resulting status "Found"

The results of the search engines follow the same structure as the searched reference. The title of the reference is shown first, followed by the author names, the DOI and the year. On the left side, similar to the reference card in the overview, the colored vertical bar displays the resulting status of the check (whereas in the detail view the displayed status relates to the check result of the respective search engine, the displayed status in the reference card overview is the united final status from all search engines). Additionally, each of the components of the search engine result has a colored tag at the start, holding a percentage. The percentage shows the similarity of the respective component to the same component in the searched reference. Additionally, the color of the tag shows the resulting status of the component similarity check. In this way, users can easily compare their searched reference to the search engine result and check which components do differ and why it is resulting in the status “found”, “needs to be checked” (see Figure 5.24) and “not found” (see Figure 5.25).

The detail view of a searched reference that is not in a supported format differs only marginally. Since the searched reference could not be split up into its components, the full reference is shown on top of the detail view (see Figure 5.26).

5.3 Design Decisions

The general design strategy of the application was to keep it minimalist and clear. Therefore, as mentioned previously, it consists of the two main components: A view to insert references and adjust search engine settings and a view to present and analyze the results. Additionally, unobtrusive colors like white and light grayish blue are used for backgrounds and blue as a contrast color for elements like buttons or specific texts which have to stand out. More details are discussed in the following chapters.

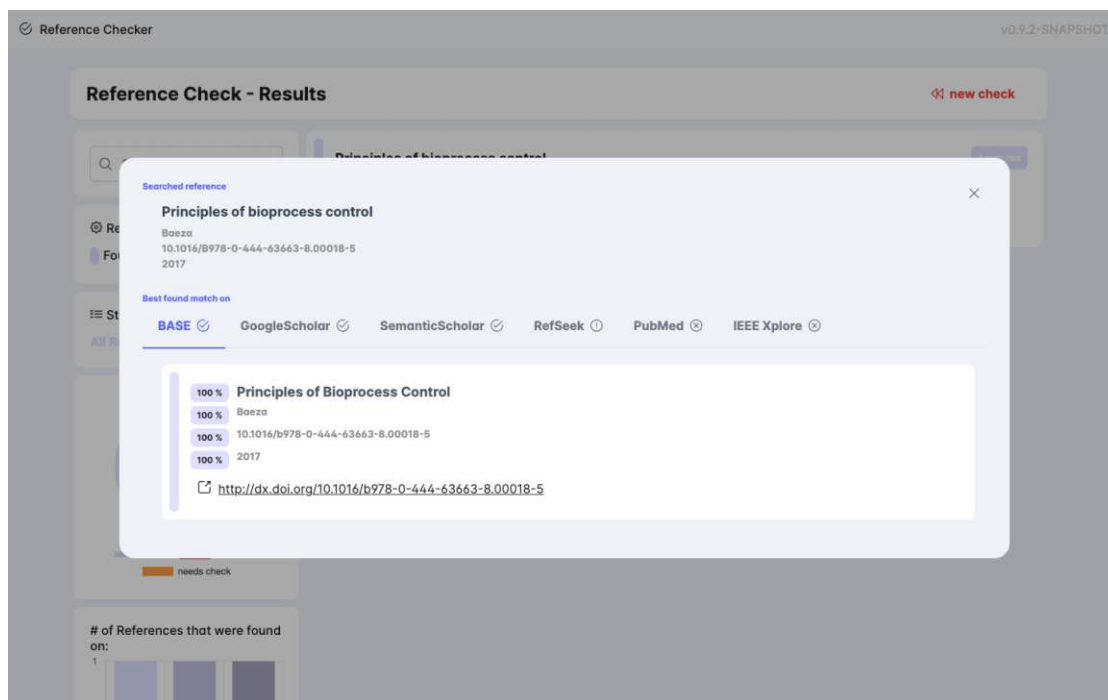


Figure 5.23: The detail view of a reference check which resulted in the status "Found"

5.3.1 Design Decisions - Search View

The idea was to set the page where users would insert their references for the checking process directly as the landing page of the application. Therefore, users can directly start with the process when visiting the web application.

Since there is no frontend-side validation concerning the citation style of the references, users are not informed that their input references are in an unsupported style. This is the case, since these references are still checked, regardless whether they are supported or not.

Although each input form holds exactly one reference, users can copy and paste a list of references directly in one form and subsequently press a button to split them up in multiple forms. This facilitates the check of multiple references from one paper, making it easy not only to add, but also to edit and delete references since they are split up.

The “Analyze” button and the “search engine settings” button are implemented as a split button in order to express that the search engine settings do have a substantial effect on the reference check itself.

5.3.2 Design Decisions - Results View

The results view is divided into two columns of unequal width. The narrower column, situated on the left, contains the result filters, statistical data and charts. The wider

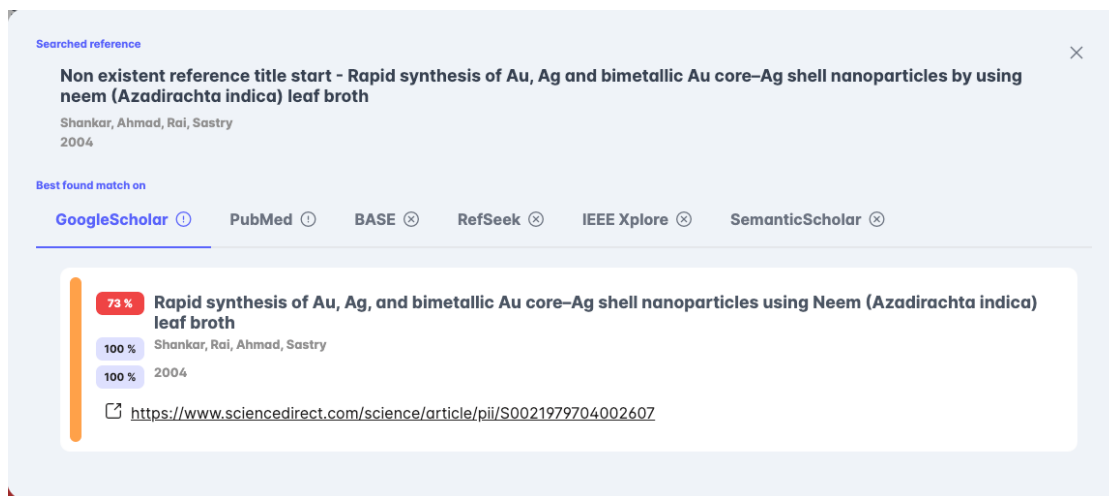


Figure 5.24: The detail view of a reference check which resulted in the status "Needs to be checked"

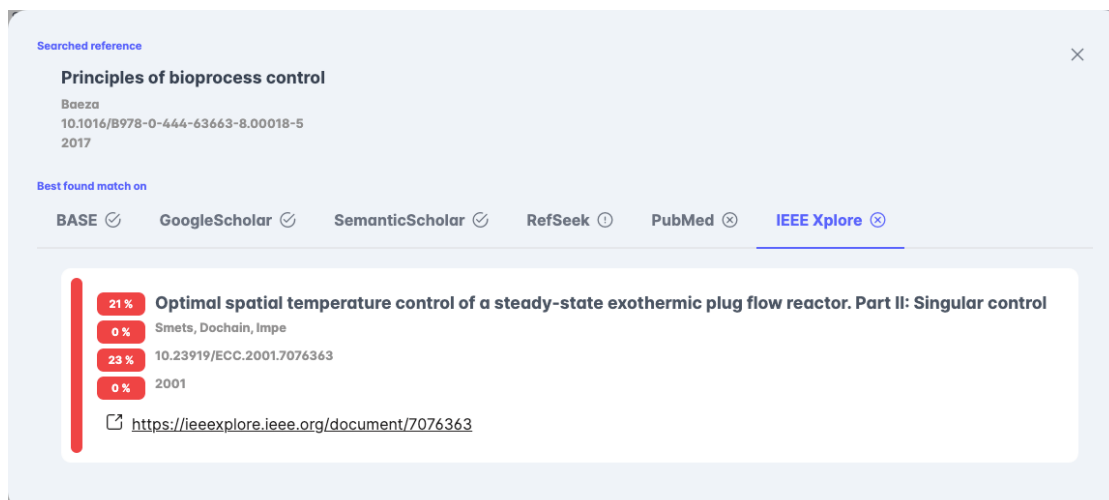


Figure 5.25: The detail view of a reference check with a selected search engine whose check resulted in the status "Not found"

column, located on the right, presents the result list of checked references. Given the disproportionate size of the two columns, the primary focus is on the result list of references, with the additional features of the left-hand column serving to provide context and support.

The three result states are expressed with different colors. All elements that represent the state “not found” are in a vibrant red, “needs to be checked” in orange, and “found” in a light grayish blue (see Figure 5.27). Whereas the color for the state “found” does not particularly stand out to the rest of the user interface, the orange of “needs to be

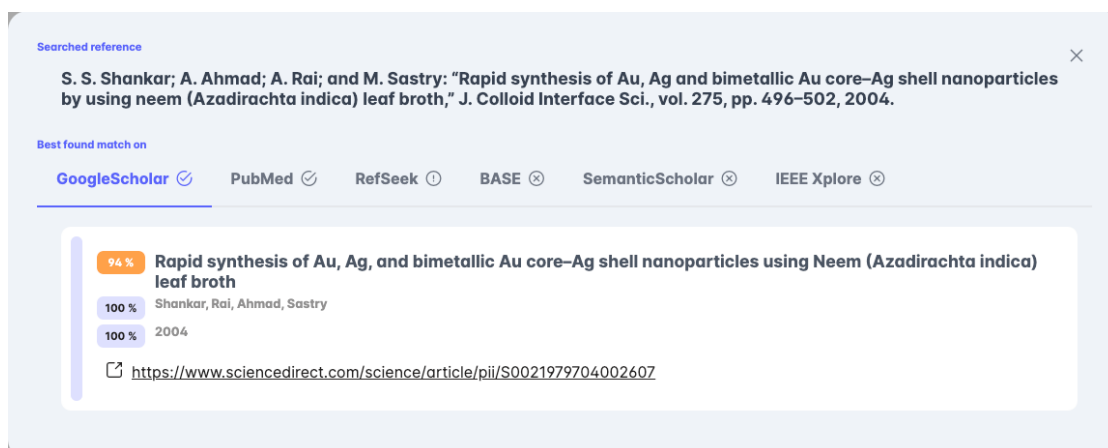


Figure 5.26: The detail view of a reference check with a searched reference in an unsupported format

checked” but even more the red of “not found” does attract attention. This is done to indicate users the most important results, namely the references that could not be found or could not be found with certainty. References that are in a not supported citation style are additionally marked with yellow highlights. This signals to the user that this is a special instance which needs to be observed.



Figure 5.27: The used status colors from left to right: "Not found", "Needs to be checked" and "Found"

Since the comparison of the components of the searched reference and the found reference on the search engines is a crucial part of the application and provides useful information for users, it is shown in the detail view of a reference result. This is implemented using a modal to prevent cluttering the result overview.

CHAPTER 6

Evaluation

Qualitative interviews were conducted for the purpose of evaluating the developed prototype and answering the research questions. Since work with scientific papers and therefore scientific references is a rather specific subject that requires a certain degree of expertise in this particular field, qualitative interviews with experts were chosen over other research methods. On the one hand the semi-structured nature of qualitative interviews enables an in-depth understanding of the participant's experiences and thoughts on the topic and on the other offers the flexibility to adapt to the participant's responses [50].

6.1 Process

The following chapter describes the full process of the qualitative expert interviews from the selection of participants, the preceding testing phase to the final execution and analysis of the interviews.

6.1.1 Participants

The focus regarding the selection of the qualitative interview participants was their expertise in the field of academic work, in particular the writing, assessment and general handling with scientific papers and thus scientific literature and references. Consequently, interviews were conducted with four experts, each of whom holds a position at the Technical University of Vienna as Professor, Assistant Professor or PreDoc Researcher.

6.1.2 Testing Phase

Preliminary to the interviews, a testing phase was carried out in order to introduce the prototype of the application to the participants. Since the interviews themselves should focus on the participants' experience with the topic and assessment of the application's features, functionality and usability, this was a crucial step for participants to familiarize

and form an opinion that would serve as the basis for the interviews. The testing phase itself spanned a period of between three and five days. In addition to the access to the application, a short description of the motivation, the application and its features and some important side notes as well as a list of example references were provided to participants.

6.1.3 Application Deployment

In order to grant access to the application, both frontend and backend were deployed on Amazon Web Services. The frontend was hosted on the Amazon Simple Storage Service (S3). The backend was hosted using Amazon Elastic Compute Cloud (EC2). Thus, participants were able to test the application by simply accessing the link via a browser.

6.1.4 Limitations of the Testing Phase

As the testing phase was conducted simultaneously for all participants, there was the risk that the search engines GoogleScholar and RefSeek would ban the server's IP if there were too many requests in a short period of time (see Chapter 4.3). This would have led to failed requests to these Search Engines. Although it is possible to circumvent this limitation (for instance by using a proxy pool), the state of the testing phases deployment did not support this feature. Therefore, participants were informed that these Search Engine Settings should be tested, but not activated for each test run.

Furthermore, due to the aforementioned reason, if multiple references were checked without the “parallel requests” setting enabled, the references were processed in a sequential manner with a waiting period between them. Such an approach would inevitably result in significantly prolonged processing times. Whereas a reference check of ten references with all search engines enabled would take more than 100 seconds, the same check with “parallel requests” setting enabled would take about 10 seconds or less.

6.1.5 Interviews

The interviews were held remotely via Microsoft Teams. Therefore, both audio and video could be recorded directly over the application, which was used for the subsequent analysis. Each interview took between 30 and 40 minutes.

In order to create the basis to answer the research questions, the main objective of the interviews was to discuss the experts' opinion on the prototype's functionality, features and usability. Also, the aim was to speak about experiences concerning the topic, and how participants think they could integrate the application in their area.

As a start, the motivation for the topic, the research questions and resulting challenges were presented. The subsequent discussion was flexible, based on the participant's responses, but still guided by the following questions:

- To what extent do you believe that it is possible to find out whether a scientific source or reference exists using a technical procedure (see prototype)?
- In which situations do you think such an application brings added value (evaluation of scientific papers, literature research, other areas?)
- How do you rate the strategy of comparing individual segments of a reference (title, authors, year and DOI) in order to draw conclusions about its correctness?
- How do you rate the selection of supported search engines or research databases, and do you think it makes sense to query the results of several at the same time?
- How do you rate the interaction with the application, especially with regard to the reference input and presentation of results?
- What do you see as advantages or disadvantages of using this type of application to check references?
- Do you see the use of this type of technical process as more of a support, or do you think that a completely autonomous use would be possible and sensible?
- What features do you think would additionally improve the application?

6.1.6 Data Analysis

Based on the interview recordings, the participants' statements were written down. Subsequently, these statements were structured according to the respective topic and certain themes and patterns were identified. Finally, the resulting findings of the individual interviews were compared with each other in order to identify similarities and differences.

6.2 Findings

The following sections present the findings of the qualitative interviews structured by individual topics. The initial five sections concentrate on the applications' user interface, its usability, features, and implementation. In contrast, the final two sections present the findings concerning the technical procedure, functionality, and use.

6.2.1 Input of References

Participant A had problems with the format of input. Although the application provided the information which citation styles it does support, the lack of understanding which format the respective citation styles follow, initially prevented him from entering references in a supported style. When pasting references in the input fields, he had some artifacts which had to be removed subsequently. Participant A mentioned that it would be ideal if the result of the reference check would not directly depend on an input of a 100 percent

correctly cited reference. Since he gets a lot of papers where references are in any kind of citation style, the functionality of the application should not be limited to specific styles.

Participant B considered the copy and paste feature of multiple references, which are then automatically recognized and separated, as beneficial. While inserted references did not work at first, since they were not in a supported citation style, after using a tool to convert these references in a supported format, namely APA, it did work like he expected.

Participant C noted that it should be supported to upload a PDF as input or provide a URL for a PDF, since copying and pasting references is too much effort. Concerning the feature of copying multiple references in one input field, he stated that it cannot be expected from users to handle line breaks themselves. Just like Participant D, he considered the initial title of the search view “Copy and paste any reference here” as misleading, since reference is singular and does not suggest that pasting multiple references is supported.

Participant D pointed out that at first it was not clear how the input of references works and hence there should be more information or restriction. In his opinion, a PDF upload is not required since he doesn’t download papers. Thus, he noted that a URL input of a paper’s online version where all references from this paper would be automatically extracted would be a great feature.

6.2.2 Supported Search Engines

Participant A noted that the majority of the tested references were found by the search engine SemanticScholar, but even more by GoogleScholar. Less were found by IEEE Xplore and PubMed. Both Participant A and B were satisfied with the supported search engines.

Participant B also stated that he achieved the greatest number of hits with SemanticScholar and that the default search engine settings (SemanticScholar, IEEE Xplore and PubMed) have always provided good results.

Participant C mentioned some search engines or research databases that would be relevant and could be added: Firstly ACM which he declared as often used in his area, secondly DBLP and thirdly ORCID. Concerning ORCID he added that it could eventually be added in the future since there are too little entries at the moment.

Participant D pointed out that the supported search engine GoogleScholar is the most important platform for checking scientific literature references. In his opinion, SemanticScholar makes sense too, but ACM and other journals and journal publishers could be added.

6.2.3 Process Speed

Participant A considered the current processing speed not as a problem. In his opinion it is acceptable, especially if the progress is shown. Nevertheless, he mentioned that the increased processing speed of the “parallel requests” setting is beneficial.

Likewise, Participant B noted that if the “parallel requests” setting is enabled, the process is considerably faster. Additionally, he mentioned that the time taken for one reference is ok, but sees a problem if it scales linearly with the number of references. Though he stated that if he would use it for correcting a paper, he would start the checking process only once and therefore considered the speed to be of secondary importance. While waiting for the results of the reference check, he would correct the paper in the meantime.

Participant C indicated that a processing time of 20 seconds for the references of a full PDF would be ok, but would be too much if each reference would take that long.

Participant D pointed out that a processing time below 10 seconds is ok. He added that when these 10 seconds are exceeded, it is the time when a human switches tasks. Therefore, he would let the process continue in the background and like Participant B come back when the reference check is finished.

6.2.4 Presentation of Results

Participant A did suggest a different title for the status “needs to be checked” since he perceived it as a bit misleading and unintuitive. In his opinion, an entry that was found, although not matching enough for having the status “found”, should still be seen as a match. Additionally, he missed an option to return to the search view while keeping the state of the input references in order to be able to edit them.

Participant B noted that the results view is very clear. He appreciated the chart’s results summary as well as the approach that important information is on top of the list. Summarized, he was convinced that the application is ready for use. As an improvement, Participant B suggested moving the link to the source of a found reference from the detail view to the overview, which would save one click.

Participant C titled the results view as nice and modern. Similar to Participant A, he missed the function to use the browser’s “back-button” to revise the input references. Furthermore, the functionality of the search bar “Search Reference” was not clear to him, since he initially thought that it had the same functionality as the reference check. Regarding the presentation of the results list, he would additionally display other information of the references like the journal title since it is relevant information.

Participant D perceived the results view as logical, serious and tidy. Similarly, as Participant A noted, in his opinion the status “needs to be checked” should be clarified as a continuum. While proposing “probably found” he explained that he assumed that a reference can result in the status “not found” and “needs to be checked” at the same time. Concerning the charts, Participant D would replace the polar chart by a bar chart, since bar charts are clearer and provide the same information.

6.2.5 Detail View

Participant A mentioned that the percent matching strategy in the detail view is a good idea since it provides precise feedback concerning the similarity of the respective components. Due to this feature, he was able to find a typo in the references of one of his own papers. Furthermore, he valued the provided links, which facilitated the inspection of the found sources.

Participant B noted that it is of secondary importance for him on which search engine or scientific database the searched reference was found on. He added that if the check results in the status “found” for a reference, he trusts the search engines and the tool.

Participant C stated that if the searched reference does not contain a DOI and the search engine result does neither, it is confusing if the component comparison in the detail view shows a similarity of zero percent. Therefore, he would not display the DOI component in this case.

Participant D mentioned that the detail view is clearly structured and does provide useful insights to the individual results of the search engines.

6.2.6 Checking References of Type “Unsupported”

All participants shared the opinion that checking references which are not in a supported citation style does provide useful information, despite the fact that the check itself is less accurate.

Participant A stated that he would like to be provided with additional information, why a pasted reference resulted in an “unsupported reference”. Participant B noted that if the links to the sources of found results weren’t provided, it would probably be a bit confusing. Since these links are accessible in the detail view, he can verify it manually.

6.2.7 Functionality and Use

In his opinion, Participant A believes, that the application in its current form is capable of achieving its intended purpose. Accordingly, he would principally use the application to assess scientific documentation, whether originating from students or the Internet, as it streamlines the process of deciding whether real looking references do indeed exist. Additionally, he mentioned to see a benefit in on the one hand being able to determine in which database the checked references were found and in which not and on the other hand to use the application to double-check whether references have been entered correctly when writing a scientific paper.

Participant B mentioned that currently at university the sources in scientific papers are not really checked and that there could well have been non-existent references in the past. Therefore, he believes that the application could be mainly used as a tool for teaching, correcting and checking student work but also for research in order to check sources from ChatGPT directly. Furthermore, he added that if the links to the sources

of found references were displayed directly in the overview, the application could be used to manage sources in the assessment process of for example bachelor theses. According to him, the use case would then be less focused on checking the existence of the source, but on facilitating checking statements in the paper.

Participant C noted that in his own scientific field, it is rather unlikely that people invent references by using ChatGPT. Though, similar to Participant B, he mentioned to see the applications' utilization in checking bachelor and diploma thesis and student work in general. Accordingly, he could imagine using it in the assessment pipeline like plagiarism checks where the system provides a score, or in the case of the reference checker suspicious references that could not be found, which then have to be reviewed manually.

Participant D explained that the current, more severe problem with scientific papers is that the text itself is often generated by LLMs and therefore of poor quality. This is shown by seminar reports plagiarism checks against AI-generated material. Therefore, he pointed out that the application's functionality is an important step in the bigger picture. Furthermore, Participant D added that particularly in the context of the academic publishing industry, where a vast quantity of material is produced with minimal oversight, it makes perfect sense that tools such as this are employed to verify references. In addition, he indicated that the application could be used as an add-on for browsers to facilitate the verification of references directly on websites, thereby enabling rapid assessment of the authenticity of the listed references.

6.3 Refinements

Based on the findings, certain refinements were made to the application. With regard to the input feature of multiple references, the functionality was enhanced in such a way that it is now independent of the source document. Furthermore, the application now automatically removes line breaks in instances where no multiple references are detected.

Additionally, certain component titles of the application were renamed in order to be clearer to users. This applies to the title of the main card in the search view, where “copy and paste any reference here” was updated to “copy and paste any references here”, the search bar in the results view which was updated from “search references” to “search” and the status in the status overview where “needs further checking” was replaced by “needs to be checked” to be more consistent.

Since one of the experts indicated the superiority of bar charts in comparison to polar charts, the diagram in the results view which displays the amount of references for each selected search engine was updated accordingly.

At last, the detail view was modified so that it only displays the DOI similarity in the event that it is present in the searched reference. As experts have pointed out, displaying it consistently would provide no additional information and could potentially be confusing.

CHAPTER 7

Discussion

This chapter discusses the most significant findings resulting from the research questions addressed in this thesis.

Research Question 1: To what extent can a technical procedure be used to find out whether a reference is genuine or not?

As a result of the qualitative interviews, it can be stated that all participants concur that the application developed as part of this thesis can be used to verify scientific literature references in student papers. Additionally, the interviewed experts pointed out that it could be utilized to check the references of scientific work from the internet, sources directly from ChatGPT or references that are listed on public websites.

Subsequently, based on the confirmed functionality of the application by experts, the following can be determined:

The implemented process of checking a reference automatically does enable users to decide whether a reference does exist on the supported search engines or not and to what degree respective components like title, authors, year and DOI do match. As shown by the application, the underlying process and strategy of citation style classification, component splitting, web scraping or dedicated APIs for accessing the search engines and the described strategy for result classification can be evaluated as a functional technical procedure (see Chapter 3.2).

By providing the functionality of multiple search engines and scientific databases in the procedure, results of the automated checking process are improved. Parallel to the manual process, where checking a reference on multiple different search engines results in a higher certainty that a reference is genuine or not, this also applies to the automated process and can be achieved by adding further search engines or scientific databases to the application. As experts have indicated, there are specific scientific databases for respective scientific fields which could be added.

As the testing phase of the qualitative interviews and the following interviews have shown, supporting three of the most popular citation styles concerning the input format of scientific references is not sufficient. As mentioned by an interview participant, the checking process should not heavily depend on the citation style and citation correctness of the reference. In further consequence, this questions the utilization of regex for component splitting, since the used technology should be both consistent and adaptive.

Although checking references without splitting the reference in its components, which is part of the applications' process when checking references in an unsupported citation style (see Chapter 3.2.3), does provide useful information, the quality of results can be worse and is more dependent on the functionality of used search engines. This could be prevented by breaking the processes' dependency from the citation style, as described previously.

Using web scraping in order to automatically access the functionality of search engines by searching for a reference and extracting the data of the results can be evaluated as functional. However, it is important to note that this strategy comes with certain downsides when being compared to the utilization of dedicated APIs. Firstly, the development and testing phase has shown that certain search engines do restrict the access to their service. Too many requests in a brief period of time can lead to captchas or IP bans, which in further consequence lets the service reject every further automated request. In regard to the supported search engines of the application, this could be observed with both GoogleScholar and RefSeek. Furthermore, as previously mentioned in the limitations section of the Development chapter (see Chapter 4.3), web scraping is extremely vulnerable to structural frontend changes, whereas small changes can already lead to malfunctioning of the application.

As a result of the development phase and the qualitative expert interviews, it can be observed that the utilization of APIs like the one provided by SemanticScholar would not only improve consistency and stability of the procedure, but also increase the process speed dramatically. This would be achieved by parallel API requests and faster response times.

As indicated by one of the experts in the qualitative interviews, the technical procedure implemented in the application can be used to verify a reference, but it is important to note that there is no guarantee of a direct correlation between the authenticity of the references and the quality of the scientific literature content itself. It is possible that a paper with authentic references may still contain generated text with poor quality. Consequently, it is necessary to view the checking of references as a crucial step that may indicate questionable content. However, this should be complemented by a thorough assessment of the content itself.

Research Question 2: How should such a system be made available to people so that they can use it productively?

The expert interviews have shown that the strategy to provide the application as a web application is sound. While this makes it comparatively easy to access and use, there are some challenges that come with this, as the testing phase has shown. Having a single server unit that hosts the backend and therefore calls the search engine services and scrapes the search result data makes it a central point of communication. This is a problem since the number of requests in a certain time period is limited by some search engines, as described previously. As the number of active users utilizing the application simultaneously increases, the aforementioned issues intensify. Getting direct access to the search engine services via an API, as exemplified by SemanticScholar, would solve this issue and hence facilitate the delivery of a stable web-based version of the application. As one of the experts indicated in the interviews, providing the application as an add-on for browsers, could be more convenient for users when directly checking scientific references on websites.

It is sufficient to structure the application in two views. One to provide the functionality to insert the references that need to be checked and edit the search settings, and one to display and analyze the results obtained from the reference check. As pointed out by experts, there should be the functionality to return from the results view to the search view without discarding the input state. This would allow users to modify any erroneous pasted references.

Whereas the strategy of providing input fields for copy and pasting references is sufficient, the expert interviews have shown that input options should be extended. A direct PDF upload could streamline the process of checking all references of a paper, for instance. Likewise, the functionality to input a URL of a scientific literature's online version could facilitate checking the references even without having to download the document. However, the mentioned options would require extracting the references from the document, which could be prone to errors and may decrease processing speed.

Resulting from the interviews, it can be determined that a process time between 10 and 20 seconds is acceptable. Yet, it should be prevented to scale linearly with the number of searched references. While this is achievable with the current version of the application when selecting the "parallel requests" setting, which limits the selected search engines to SemanticScholar, it is not feasible regarding search engines like GoogleScholar and RefSeek to the aforementioned restrictions.

With regard to the process speed, experts agreed that the progress bar is essential to provide feedback when checking multiple references.

As two of the qualitative interview participants have indicated, the title of the resulting status classification is decisive in order that users can interpret the results correctly. Instead of "found", "needs to be checked" and "not found", it would be more efficient to utilize a representation of a continuum, such as "found", "probably found" and "not found."

Additionally, the following design decisions and features can be evaluated as functional for a productive utilization of the application: Using expressive colors for the resulting status is beneficial when it comes to the analysis of the reference check results. It facilitates a fast categorization and generates a clear picture. Similarly, organizing the results as a list, with items sorted from "not found" to "found," has an equivalent impact. This approach guides the user's attention to the information of the greatest relevance.

The featured charts on the one hand allow users a preliminary evaluation of the reference check results, and on the other enable the assessment of ratios regarding both result statuses and success rates of selected search engines.

Although seen as a not necessary addition by one of the experts, the others proposed that the application's detail view is a sensible solution to provide additional information concerning the individual search engine results and the corresponding similarity scores. These percent scores for each component provide users with a detailed understanding of the similarity in general and in further consequence the rationale behind the classification of a reference with a specific status. Additionally, it presents the underlying source URL of the search engine result which is according to the experts a helpful feature to both recheck the source manually or for research purposes.

Conclusion and Future Perspectives

This thesis explored the development of an application that provides the functionality of verifying scientific literature references in an automated manner. The primary aim was to empower users with a tool that simplifies the process of validating such references, offering a more efficient and user-friendly experience by generating comprehensive reports and presenting information and statistics pertaining to the obtained results. The resulting findings of this thesis indicate, that such an application can be a useful tool, to streamline the checking and analyzation process of scientific literature references.

Throughout the course of this thesis, the following research questions

- To what extent can a technical procedure be used to find out whether a reference is genuine or not?
- How should such a system be made available to people so that they can use it productively?

have been answered by performing a theoretical, practical and empirical analysis. The theoretical analysis provided a foundation in the current state of the art and approaches, references, and their citation styles, as well as relevant sources for scientific works such as search engines and scientific databases. Further research was required to ascertain the most effective methods for the collection of this data, as well as the optimal approach of making the application available to users and determining which features should be included.

Consequently, the development phase employed the insights derived from the theoretical analysis to come up with a concept design, construct a prototype, and conduct a final

refinement based on the feedback obtained from the empirical analysis.

Subsequent to a testing phase lasting between three and five days, post development qualitative expert interviews were conducted. This resulted in empirical data concerning user perceptions, usability, and functionality of the application, which were crucial to determining the extent to which the prototype retrieved from the development phase could fulfill the requirements regarding research questions 1 and 2.

The principal conclusions of this thesis are as follows: The developed prototype application presents a functional strategy for automating the verification of scientific literature references. This process entails a comparison of the similarity of the key components of a reference, including the title, authors, year of publication, and DOI. In order to obtain the data from search engines utilized for this similarity comparison, web scraping is a viable approach. However, it does have some adverse consequences: the application is hardly scalable in terms of concurrent user requests, processing times are increased, the quality of responses is potentially inferior, and it shows a high degree of vulnerability to structural frontend changes of the search engines. If the search engine services provide dedicated APIs, these disadvantages could be eliminated. The incorporation of multiple search engines into the application's process has been demonstrated to enhance the reliability of the resulting data.

The provision of the application as a web application ensures its accessibility. In regard to the input options, the application should be capable of processing text, document files, and URLs to online versions of documents. The methodology for extracting components of a reference should be independent of the citation style. The provision of both statistical charts and a sorted list of results according to their status facilitates the comprehension of the results view. Furthermore, it is beneficial to display additional information regarding each search engine result in a detail view, including similarity percentages for each component.

It is important to consider the limitations of this thesis concerning the testing phase and the qualitative expert interviews. Given that the participants were highly experienced experts with demanding schedules, the testing phase itself was adjusted accordingly. An additional constraint is the limited sample size of four participants in the qualitative interviews. While qualitative research prioritizes depth of insight, this could still limit the diversity. Also, all the experts were either Professor, Assistant Professor or PreDoc Researcher at the Technical University of Vienna in the field of computer science. Future research could address this limitation by including a larger and more diverse participant pool, thereby enhancing the robustness and generalizability of the findings.

As recommended, future studies could enhance the process of verifying scientific literature references by decoupling the component extraction procedure from the citation style. Furthermore, as one of the experts noted, while verification of references is an essential step in evaluating the quality of scientific literature, it is not a sufficient measure. It would be beneficial to investigate strategies for evaluating both the referenced sources and the assertion's presence within those sources.

Overview of Generative AI Tools Used

In certain cases the following tool has been used to achieve more professional and natural formulations:

- DeepL Write (Online Version: <https://www.deepl.com/de/write>)

List of Figures

2.1	Example report of Turnitin Similarity	6
2.2	Resulting report of a reference analysis using Recite	6
2.3	Resulting report of a reference analysis using Scribbr Citation Checker . .	7
2.4	Resulting report of a reference analysis using the reference check of scite.ai	8
2.5	Resulting report of a reference analysis using the Trinkai.ai citation checker.	9
3.1	Process of checking a single reference from input to result.	15
3.2	Process of classifying a searched reference.	16
4.1	The technical architecture of the application.	20
5.1	The initial wireframe design of the search view	28
5.2	The final wireframe design of the search view	28
5.3	The initial wireframe design of the results view	29
5.4	The final wireframe design of the results view	29
5.5	The initial wireframe design of the details view.	30
5.6	The final wireframe design of the details view.	30
5.7	The final mockup of the search view	31
5.8	The final mockup of the results view	31
5.9	The final mockup of the details view	32
5.10	Search View Page	32
5.11	Multiple references pasted in a single input field	33
5.12	Multiple references split into multiple input fields	33
5.13	Overview of the Search Engine Settings with parallel requests deactivated and activated	34
5.14	Loading screen showing the progress of a reference check with 7 references	35
5.15	The results view of the application showing the outcome of a reference check with seven references	36
5.16	The search and filter controls in the results view	37
5.17	The status overview and donut chart showing the resulting status of the reference check	37
5.18	Bar chart showing the number of references that were found on each search engine	38
5.19	Result card of a reference with the resulting status "Found"	38
		59

5.20	Result card of a reference with the resulting status "Needs to be checked"	38
5.21	Result card of a reference with the resulting status "Not found"	39
5.22	Result card of a reference in an unsupported format with the resulting status "Found"	39
5.23	The detail view of a reference check which resulted in the status "Found"	40
5.24	The detail view of a reference check which resulted in the status "Needs to be checked"	41
5.25	The detail view of a reference check with a selected search engine whose check resulted in the status "Not found"	41
5.26	The detail view of a reference check with a searched reference in an unsupported format	42
5.27	The used status colors from left to right: "Not found", "Needs to be checked" and "Found"	42

List of Tables

3.1	Collection of scientific references that were used to assess and compare academic search engines. (For complete reference list see appendix "List of References for Testing")	13
3.2	Result of found references in percentage for each academic search engine. Sorted in descending order from best to worst.	13

List of References for Testing

1. P. Gerges, "Simulation of Operating IV-VI Semiconductor Laser Chips for the Development of Continuous Wave VECSELs," Diploma Thesis, Technische Universität Wien, 2020.
 - a) C. Walther, M. Fischer, G. Scalari, R. Terazzi, N. Hoyler, and J. Faist, "Quantum cascade lasers operating from 1.2 to 1.6 THz," *Applied Physics Letters*, vol. 91, no. 13, p. 131122, 2007.
 - b) S. W. Sharpe, T. J. Johnson, R. L. Sams, P. M. Chu, G. C. Rhoderick, and P. A. Johnson, "Gas-phase databases for quantitative infrared spectroscopy," *Applied Spectroscopy*, vol. 58, no. 12, pp. 1452-1461, 2004.
2. J. Kager, J. Bartlechner, C. Herwig, and S. Jakubek, "Direct control of recombinant protein production rates in E. coli fed-batch processes by nonlinear feedback linearization," *Chemical Engineering Research and Design*, vol. 182, pp. 290–304, 2022. [Online]. Available: <https://doi.org/10.1016/j.cherd.2022.03.043>
 - a) J. A. Baeza, "Principles of bioprocess control," in *Current Developments in Biotechnology and Bioengineering*, Elsevier, pp. 527–561, 2017. [Online]. Available: <https://doi.org/10.1016/B978-0-444-63663-8.00018-5>
 - b) M. Ellis, P. Patel, M. Edon, W. Ramage, R. Dickinson, and D. P. Humphreys, "Development of a high yielding e. coli periplasmic expression system for the production of humanized fab fragments," *Biotechnol. Progress*, vol. 33, pp. 212-220, 2017.
3. J. Ackermann, "Integrating cell-free fetal DNA from amniotic fluid in prenatal diagnostics," Master's Thesis, Universität Wien, 2023. [Online]. Available: [doi: 10.25365/THESIS.75017](https://doi.org/10.25365/THESIS.75017).
 - a) A. Kallioniemi, O. P. Kallioniemi, D. Sudar, et al., "Comparative Genomic Hybridization for Molecular Cytogenetic Analysis of Solid Tumors," *Science*, vol. 258, no. 5083, pp. 818-821, Oct. 1992. [Online]. Available: <https://www.science.org/doi/10.1126/science.1359641>
 - b) D. T. Miller, K. Lee, A. S. Gordon, et al., "Recommendations for reporting of secondary findings in clinical exome and genome sequencing, 2021 update: a

policy statement of the American College of Medical Genetics and Genomics (ACMG)," *Genetics in Medicine*, vol. 23, no. 8, pp. 1391-1398, Aug. 2021. [Online]. Available: <https://doi.org/10.1038/S41436-021-01171-4>

4. E. Salomon, "A 3D structured breast phantom with lesion models for quality control of digital breast tomosynthesis," Doctoral Thesis, Medizinische Universität Wien, 2021.
 - a) S. Ciatto et al., "Integration of 3D digital mammography with tomosynthesis for population breast-cancer screening (STORM): a prospective comparison study," *Lancet*, vol. 14, no. 7, pp. 583-589, 2013. [Online]. Available: [https://doi.org/10.1016/S1470-2045\(13\)70134-7](https://doi.org/10.1016/S1470-2045(13)70134-7)
 - b) X. Wu, G. T. Barnes, and D. M. Tucker, "Spectral Dependence of Glandular Tissue Dose in Screen-Film Mammography," *Radiology*, vol. 179, pp. 143-148, 1991.
5. L. Nitzschke, "Die Rolle traditioneller Genderidentitäten bei den Auswirkungen negativer Altersstereotype in sozialen Bereichen," Master's Thesis, Universität Wien, 2023. [Online]. Available: doi: 10.25365/thesis.73787.
 - a) S. Engler, "Habitus und sozialer Raum: Zur Nutzung der Konzepte Pierre Bordieus in der Frauen- und Geschlechterforschung," in *Handbuch Frauen- und Geschlechterforschung*, R. Becker & B. Kortendieck, Eds. VS Verlag für Sozialwissenschaften, 2008, pp. 250-262. doi: 10.1007/978-3-531-91972-0_29.
 - b) B. R. Levy and M. R. Banaji, "Implicit ageism," in *Ageism: Stereotyping and prejudice against older persons*, T. Nelson, Ed. MIT Press, 2002, pp. 49-75. doi: 10.7551/mitpress/1157.001.0001.
6. A. Bauer, "Die Teilung der Tschechoslowakei 1992 und eine Analyse ihrer historischen Ursachen," Diploma Thesis, Universität Wien, 2009. [Online]. Available: doi: 10.25365/thesis.3614.
 - a) M. Bútorá and Z. Bútorová, "Die unerträgliche Leichtigkeit der Trennung," in *Abschied von der Tschechoslowakei. Ursachen und Folgen der tschechisch-slowakischen Trennung*, R. Kipke & K. Vodička, Eds. Köln, 1993, pp. 108-139.
 - b) H. Krejčová, "Juden in den 30er Jahren des 20. Jahrhunderts," in *Juden zwischen Deutschen und Tschechen. Sprachliche und kulturelle Identitäten in Böhmen 1800 – 1945*, W. Koschmal & M. Nekula, Eds. München, 2006, pp. 85-103.
7. G. C. Scholz, "Die Systemkorruption in Rumänien," Diploma Thesis, Universität Wien, 2008. [Online]. Available: doi: 10.25365/thesis.2484.
 - a) A. Heidenheimer, "Perspectives on the Perception of Corruption," in *Political Corruption: concepts and contexts*, A. J. Heidenheimer & M. Johnston, Eds. New Brunswick/ London, 3rd ed. 2005, pp. 141-154.

- b) J. G. Peters and S. Welch, "Gradients of Corruption in Perceptions of American Public Life," in *Political Corruption: concepts and contexts*, A. J. Heidenheimer & M. Johnston, Eds. New Brunswick/ London, 3rd ed. 2005, pp. 155-172.
- 8. J. Kloibhofer, "A fixed-point theorem for Horn formula equations," Diploma Thesis, Technische Universität Wien, 2021. [Online]. Available: [repositUM](https://repositum.tu-wien.ac.at/handle/10.34726/hss.2021.85542). doi: 10.34726/hss.2021.85542.
 - a) S. Hetzl and S. Zivota, "Decidability of affine solution problems," *Journal of Logic and Computation*, vol. 30, no. 3, pp. 697-714, 2019.
 - b) W. Ackermann, "Untersuchungen über das Eliminationsproblem der mathematischen Logik," *Mathematische Annalen*, vol. 110, no. 1, pp. 390-413, Dec. 1935.
- 9. P. Patsuk-Bösch, "A framework for execution-based model profiling," Diploma Thesis, Technische Universität Wien, 2020. [Online]. Available: [repositUM](https://repositum.tu-wien.ac.at/handle/10.34726/hss.2020.40537). doi: 10.34726/hss.2020.40537.
 - a) N. Bencomo and L. H. Garcia Paucar, "Ram: Causally-connected and requirements-aware runtime models using Bayesian learning," in *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*, Sep. 2019, pp. 216-226.
 - b) M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede, *Process-aware Information Systems: Bridging People and Software Through Process Technology*. John Wiley & Sons, Inc., New York, NY, USA, 2005.
- 10. J. Schoisswohl, "Automated induction by reflection," Diploma Thesis, Technische Universität Wien, 2020. [Online]. Available: [repositUM](https://repositum.tu-wien.ac.at/handle/10.34726/hss.2020.75342). doi: 10.34726/hss.2020.75342.
 - a) S. Baker, A. Ireland, and A. Smail, "On the use of the constructive omega-rule within automated deduction," in *Logic Programming and Automated Reasoning, International Conference LPAR'92, St. Petersburg, Russia, July 15-20, 1992, Proc.*, vol. 624 of *Lecture Notes in Computer Science*, A. Voronkov, Ed. Springer, 1992, pp. 214-225.
 - b) S. Cruanes, "Superposition with Structural Induction," in *Proc. of FROCoS*, 2017, pp. 172-188.

Bibliography

- [1] “OpenAI DevDay: Opening Keynote.” <http://web.archive.org>. Accessed: 22.12.2023.
- [2] “Chat GPT: What is it?.” <https://uca.edu/cetal/chat-gpt/#wrap>. Accessed: 25.12.2023.
- [3] B. Lund and T. Wang, “Chatting about ChatGPT: How may AI and GPT impact academia and libraries?,” *Library Hi Tech News*, vol. 40, 02 2023.
- [4] A. Gray, “Chatgpt “contamination”: estimating the prevalence of llms in the scholarly literature,” 2024.
- [5] “Chatgpt and Generative Artificial Intelligence (AI): Incorrect bibliographic references.” https://subjectguides.uwaterloo.ca/chatgpt_generative_ai/incorrectbibreferences. Accessed: 25.12.2023.
- [6] A. Frosolini, P. Gennaro, F. Cascino, and G. Gabriele, “In reference to “Role of Chat GPT in Public Health”, to highlight the AI’s incorrect reference generation,” *Annals of Biomedical Engineering*, vol. 51, no. 10, pp. 2120–2122, 2023.
- [7] E. Hill-Yardin, M. Hutchinson, R. Laycock, and S. Spencer, “A chat(gpt) about the future of scientific publishing,” *Brain, Behavior, and Immunity*, vol. 110, 03 2023.
- [8] N. Manohar and S. Prasad, “Use of chatgpt in academic publishing: A rare case of seronegative systemic lupus erythematosus in a patient with hiv infection,” *Cureus*, vol. 15, 02 2023.
- [9] A. McGowan, Y. Gui, M. Dobbs, S. Shuster, M. Cotter, A. Selloni, M. Goodman, A. Srivastava, G. A. Cecchi, and C. M. Corcoran, “ChatGPT and Bard exhibit spontaneous citation fabrication during psychiatry literature search,” *Psychiatry Research*, vol. 326, 2023.
- [10] S. Hove and B. Anda, “Experiences from conducting semi-structured interviews in empirical software engineering research,” in *11th IEEE International Software Metrics Symposium (METRICS’05)*, pp. 10 pp.–23, 2005.

- [11] https://link.springer.com/referenceworkentry/10.1007/978-94-007-0753-5_2337. Accessed: 27.12.2023.
- [12] "What is a plagiarism checker and where can i find one?." <https://libanswers.snhu.edu/faq/75352>. Accessed: 12.04.2024.
- [13] B. Gipp, N. Meuschke, and C. Breitingner, "Citation-based plagiarism detection: Practicability on a large-scale scientific corpus," *Journal of the Association for Information Science and Technology*, vol. 65, no. 8, pp. 1527–1540, 2014.
- [14] <https://www.turnitin.com/products/similarity/>. Accessed: 27.12.2023.
- [15] <https://reciteworks.com/>. Accessed: 29.12.2023.
- [16] <https://www.scribbr.com/citation/checker/>. Accessed: 30.12.2023.
- [17] <https://scholar.google.com/>. Accessed: 29.12.2023.
- [18] <https://web.archive.org/>. Accessed: 29.12.2023.
- [19] T. Day, "A preliminary investigation of fake peer-reviewed citations and references generated by ChatGPT," *The Professional Geographer*, vol. 75, no. 6, pp. 1024–1027, 2023.
- [20] "About the internet archive." <https://archive.org/about/>. Accessed: 12.04.2024.
- [21] <https://scite.ai/home#reference-check>. Accessed: 30.12.2023.
- [22] <https://www.trinka.ai/features/citation-checker>. Accessed: 02.01.2024.
- [23] "Suchmaschinen." <https://www.hochschule-trier.de/hauptcampus/bibliothek/recherche-werkzeuge/suchmaschinen>. Accessed: 02.03.2024.
- [24] "The top list of academic search engines." <https://paperpile.com/g/academic-search-engines/>. Accessed: 08.03.2024.
- [25] "28 best academic search engines that make your research easier." <https://www.scijournal.org/articles/academic-search-engines>. Accessed: 08.03.2024.
- [26] "Citation styles: Apa, mla, chicago, turabian, ieee." <https://pitt.libguides.com/citationhelp/overview>. Accessed: 01.05.2024.
- [27] "Manuskript und proofreading." <https://www.tuwien.at/bibliothek/publizieren/manuskript-und-proofreading>. Accessed: 01.05.2024.

- [28] "Citation." <https://libguides.brown.edu/citations/styles>. Accessed: 01.05.2024.
- [29] "Angular features." <https://angular.io/features>. Accessed: 09.05.2024.
- [30] "8 most popular web programming frameworks (2024)." <https://www.appacademy.io/blog/8-most-popular-web-programming-frameworks>. Accessed: 09.05.2024.
- [31] <https://primeng.org/>. Accessed: 09.05.2024.
- [32] <https://spring.io/why-spring>. Accessed: 14.05.2024.
- [33] <https://www.selenium.dev/>. Accessed: 14.05.2024.
- [34] <https://projectlombok.org/>. Accessed: 14.05.2024.
- [35] <https://github.com/rrice/java-string-similarity?tab=readme-ov-file>. Accessed: 14.05.2024.
- [36] <https://github.com/FasterXML/jackson>. Accessed: 14.05.2024.
- [37] <https://www.oracle.com/corporate/features/project-lombok.html>. Accessed: 14.05.2024.
- [38] "What is an api?." <https://www.ibm.com/topics/api>. Accessed: 16.05.2024.
- [39] "Academic graph api (1.0)." <https://api.semanticscholar.org/api-docs/>. Accessed: 16.05.2024.
- [40] M. A. Khder, "Web scraping or web crawling: State of art, techniques, approaches and application.," *International Journal of Advances in Soft Computing & Its Applications*, vol. 13, no. 3, 2021.
- [41] D. Nourie and M. McCloskey, "Regular expressions and the java programming language." <https://www.oracle.com/technical-resources/articles/java/regex.html>, 2001. Accessed: 17.05.2024.
- [42] "Named capturing group." https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Regular_expressions/Named_capturing_group. Accessed: 017.05.2024.
- [43] "Citation styles: Apa, mla, chicago, turabian, ieee." <https://pitt.libguides.com/citationhelp/ieee>. Accessed: 28.07.2024.
- [44] K. M. M. Aung, *Comparison of levenshtein distance algorithm and needleman-wunsch distance algorithm for string matching*. PhD thesis, MERAL Portal, 2019.
- [45] "String similarity metrics." <https://www.baeldung.com/cs/string-similarity-edit-distance>. Accessed: 28.07.2024.

- [46] “What is a doi?” <https://www.doi.org/the-identifier/what-is-a-doi/>. Accessed: 18.05.2024.
- [47] “It’s hard getting a date (of publication).” <https://unlockingresearch-blog.lib.cam.ac.uk/?p=1716>. Accessed: 20.05.2024.
- [48] “Pros and cons of web scraping.” <https://hasdata.com/blog/pros-and-cons-of-web-scraping>. Accessed: 31.05.2024.
- [49] <https://www.semanticscholar.org/product/api>. Accessed: 31.05.2024.
- [50] W. Adams, *Conducting Semi-Structured Interviews*. 08 2015.