EAsyAnon

\ Software Requirements Specification (SRS)

Recommender System

Version 1.0

Jakob Folz,
Robert Aufschläger,
Manjitha Dahanayaka Vidanalage,
Elena März,
Johann Guggumos,
Md Moin Uddin,
Sebastian Wilhelm

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung
Finanziert von der Europäischen Union NextGenerationEU

This document was created as part of the research project EAsyAnon – Empfehlungs- und Auditsystem zur Anonymisierung, funded by the German Federal Ministry of Education and Research (BMBF) and the European Union - NextGeneration EU.

The document is provided by:

Technische Hochschule Deggendorf
Dieter-Görlitz-Platz 1
94469 Deggendorf, Germany

Authors:

- Jakob Folz (jakob.folz@th-deg.de)
- Robert Aufschläger (robert.aufschlaeger@th-deg.de)
- Manjitha Dahanayaka Vidanalage (manjitha.dahanayaka-vidanalage@th-deg.de)
- Elena März (elena.maerz@jura.uni-augsburg.de)
- Johann Guggumos (johann.guggumos@jura.uni-augsburg.de)
- Md Moin Uddin (md.uddin@th-deg.de)
- Sebastian Wilhelm (sebastian.wilhelm@th-deg.de)

Document Version: This document may be updated from time to time. The most recent version of this document can always be found at the following URL: `https://www.doi.org/10.5281/zenodo.13318624`. Please ensure that you are using the latest version.

Disclaimer: The information contained in this document is intended for general informational purposes as part of the conceptual work within the EAsyAnon project. It is important to note that this document represents conceptual frameworks, which may differ from the final implemented demonstrators and solutions. While the authors strive to ensure the accuracy and reliability of the information provided, they make no representations or warranties of any kind, express or implied, regarding the completeness, accuracy, reliability, suitability, or availability of the content for any purpose. Any reliance you place on such information is strictly at your own risk.

In no event will the authors or the Technische Hochschule Deggendorf be liable for any loss or damage, including without limitation indirect or consequential loss or damage, or any loss or damage whatsoever arising from loss of data or profits arising out of, or in connection with, the use of this document.

Bibliographic Information:
Title: Software Requirements Specification: EAsyAnon Recommender System
Authors: Jakob Folz, Robert Aufschläger, Manjitha Dahanayaka Vidanalage, Elena März, Johann Guggumos, Md Moin Uddin, Sebastian Wilhelm
Publication Date: September 9, 2024
DOI: `10.5281/zenodo.13318624`

Contact Information:
Website: `https://www.easyanon.de`
Email: info-easyanon@th-deg.de

Contents

List of Figures

List of Tables

List of Abbreviations

EAsyAnon	Empfehlungs- und Auditsystem zur Anonymisierung
LGPL	Lesser General Public License (LGPL)
SRS	Software Requirements Specification
GDPR	General Data Protection Regulation
QA	Quality Assurance
ID	Direct Identifier
QI	Quasi Identifier
SA	Sensitive Attribute
HIPAA	Health Insurance Portability and Accountability Act
EU	European Union
MIET	Meta Information Extraction Tool
JSON	JavaScript Object Notation
CI/CD	Continuous Integration/Continuous Deployment
HTTPS	Hypertext Transfer Protocol Secure
GUI	Graphical User Interface
TLS	Transport Layer Security
DPIA	Data Protection Impact Assessment

Chapter 1

Introduction

Companies, healthcare institutions, research institutions and public authorities find themselves in a conflict between a too-low or too-high degree of anonymisation when it comes to anonymising data, which is supposed to be published as Open Data. It is always necessary to consider which degree of anonymisation is legally and ethically needed and practically reasonable. This balancing act is often a complex challenge. For this reason, an intelligent recommendation system will be developed to support those responsible for creating and implementing an anonymisation concept suitable for the respective datasets. The intelligent recommendation system is an essential component of the EAsyAnon project, which provides concrete recommendations for anonymisation based on a standardised meta-description of the specific dataset, in which both the legal and ethical aspects are considered. In this way, a suitable anonymisation procedure is found for each dataset, with appropriate target strengths in each case, to achieve the necessary balance between information preservation and the degree of anonymisation. In EAsyAnon, those responsible for the data are supported in the technical implementation of anonymisation. Therefore, sample implementations of the recommended anonymisation procedures and third-party tools are provided to help anonymise data to minimise the technical hurdle. The anonymisation of the dataset itself is carried out by the data owner directly locally on his site. Third parties, such as the operators of the recommendation system, do not have access to personal data at any time. In this way, the User is accompanied through the entire process of anonymisation and continuous compliance with the provisions of the General Data Protection Regulation (GDPR) is ensured.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the Recommender System from the EAsyAnon project in natural language and on a technical level for all project participants to ensure targeted development of the defined requirements and functionalities of the system over the entire project duration. For this purpose, this document describes the different requirements and functionalities of the Recommender System, how the various User interfaces must be designed, and which prerequisites must be met for the system to fulfil the desired functionality. The target audience of this document is, on the one hand, all project participants from EAsyAnon, who ensure that all requirements and functionalities of the Recommender System have been defined to realise the overall system of EAsyAnon. On the other hand, it is for the developers and experts responsible for the work package and are in charge of evaluating and realising the technical feasibility of the functionalities.

This document focuses on the Recommender System and not the overall system of EAsyAnon.

Separate SRS documents are available for the two other modules, the Audit System and the Trust Service.

1.2 Intended Audience and Reading Suggestions

This section introduces the different types of reader that the document is intended for, such as and gives an overview how the rest of this SRS contains as well as how it is organized. In addition, after every stakeholder there are brackets indicating which chapter is relevant for them. The intended audience for this SRS includes, but is not limited to, the following stakeholders:

- **Development Team:** The software developers and engineers who will be responsible for implementing the requirements outlined in this document. (3, 4, 5, 6, 7, 8, 9)
- **Administrators:** Administrators are key Users responsible for managing and maintaining the software system after the Development. They are entrusted with overseeing various aspects of system configuration, User management, security, databases and overall system performance. (1, 2, 7, 8, 9, 10)
- **Quality Assurance Team:** The testing and Quality Assurance (QA) professionals who will use this document as a reference for designing test cases and validating the software's functionality. (3, 4, 5, 6, 7, 8, 9)
- **Project Managers:** The project management team responsible for overseeing the software development process and ensuring that the Recommender Systems aligns with the specified requirements. (3, 4, 5, 6, 7, 8, 9)
- **System Architects:** The individuals responsible for the high-level design and architecture of the Recommender System. (3, 4, 5)
- **Security Specialists:** Security experts who will assess and validate the security requirements outlined in this document to ensure robust protection against potential threats. (3, 4, 5, 7)

The SRS document is structured into ten chapters, each offering a concise overview and further segmented into sections for structured coherence. Below is a brief outline of each chapter to acquaint readers with the document's content.

Chapter 1 - Introduction: Outlines the EAsyAnon project and its Recommender System, the focus of this SRS. It includes a list of abbreviations and significant terms used throughout the document.

Chapter 2 - Overall Description: Provides an overview of EAsyAnon, illustrating the Recommender System's role within the project and the interaction between its components. It briefly describes the system's main functions, User types, supported input files, forthcoming documentation, and pertinent design and implementation constraints.

Chapter 3 - External Interface Requirements: Initiates with the Recommender System's workflow, illustrated through a sequence diagram from User interaction to recommendation receipt. It details the User interface, hardware requirements for Users and the server, User-component interactions, and data flow.

Chapter 4 - System Features: A critical chapter detailing the Recommender System's main features. Each feature is elaborated with a description, objective, initiating stimulus, and requisite conditions for functionality.

Chapter 5 - Functional Requirements: Mirrors Chapter 4 in structure, focusing on non-functional requirements vital for the system's proper operation.

Chapter 6 - Performance Requirements: Specifies performance criteria concerning response times, scalability, throughput, and algorithmic efficiency for the Recommender System.

Chapter 7 - Security Requirements: Addresses security needs due to online accessibility and potential sensitivity of processed information. Lists requirements for information protection, unauthorized access prevention, and system integrity.

Chapter 8 - Privacy Requirements: Delves into privacy aspects, identifying risks like singling out, linkability, and inference. Discusses QIs, privacy model metrics, and outlines a data protection impact assessment with residual risks.

Chapter 9 - Software Quality Attributes: Focuses on the quality attributes of the Recommender System, listing characteristics that define its quality and performance. Each attribute is discussed with associated requirements and metrics for evaluation.

Chapter 10 - Business Rules: Details the Business Rules, defining operational policies, constraints, and decision-making criteria within the EAsyAnon project, impacting functionality indirectly.

1.3 Product Scope

To better clarify the role of the Recommender System in the project, this section describes a high-level perspective of how the process from recommendation to anonymisation is envisioned in the EAsyAnon project. In total, the EAsyAnon project aims to realise six key innovations through three main components: the Recommender System, the Audit System and the Trust Service. The Recommender System is an interface that allows Users to import personal data and receive a targeted recommendation on how to anonymise it. The recommended anonymisation can be done locally using existing sample implementations or third-party tools without the personal data having to leave the institution. If the anonymised dataset does not meet the desired quality of the User, the recommendation can be further tweaked with the help of the User, and the complete process can be repeated until the User is satisfied with the result. Afterwards, the anonymised dataset can be passed on to the Audit System, where it is further assessed for the quality of its anonymisation, first in an automated process and then through a crowd-sourced peer review. Before finally being made available to the public as Open Data. The Trust Service plays no role in the previously described loop between recommendation and audit. It is a stand-alone module with no dependencies regarding the Recommender and Audit System. The Trust Service uses alternative approaches to analyse and evaluate encrypted data without participants gaining access to sensitive information. Strictly speaking, six key innovations will be realised in this project, described below.

Key Innovation 1 - Intelligent recommendation system for the anonymisation of data sets based on a data description (focus of this SRS document): Within the framework of EAsyAnon, an intelligent recommendation system is being researched, which, based on a standardised meta-description of the specific dataset, provides concrete recommendations for anonymisation in which both the legal and ethical aspects are considered. In this way, the most suitable anonymisation procedure is found for each data set, with appropriate target strengths in each case, to achieve the necessary balance between information content and the degree of anonymisation.

Key Innovation 2 - Support with the technical implementation of the recommended anonymisation procedure: Data controllers are supported in EAsyAnon in the actual technical implementation of anonymisation. For this purpose, sample implementations are provided for each recommended anonymisation procedure, which minimises the technical hurdle. The anonymisation of the data set itself takes place directly at the data controller. Third parties, such as the operators of the recommendation system, do not have access to the personal data at any time. In this way, the User is accompanied through the entire process of anonymisation. Process of anonymisation and continuous conformity with the provisions of the GDPR is ensured.

Key Innovation 3 - Automated Audit System to determine the appropriateness of the anonymisation carried out: In EAsyAnon, an automated Audit System is designed which enables the User to have the implementation of the recommended anonymisation procedure checked automatically in the first stage. On the one hand, statistical methods are used, and on the other hand, it is checked whether de-anonymisation is possible using automated methods (e.g., brute force or outlier detection). The automated test procedure is based on the standardised meta-description of the data set and the anonymisation concept recommended by the system.

Key Innovation 4 - Crowd-sourced peer review process to determine the appropriateness of the anonymisation carried out: In the EAsyAnon project, a crowd-sourced peer review process is being set up in addition to the automated Audit System, in which experts from various scientific disciplines prepare a public expert opinion on the de-anonymisability of the data set before publication. Based on the expert opinions and the result of the prior automated review, the User receives an assessment of the de-anonymisability of the dataset substantiated by several independent parties. Possible errors in anonymisation implementation can thus be identified and corrected before general publication. The User is therefore supported in their responsibility.

Key Innovation 5 - EAsyAnon overall system: The overall system developed in EAsyAnon accompanies data stewards through the entire anonymisation process for data publication. This starts with creating a meta-description of the data set, followed by developing an anonymisation concept, and ends with the actual anonymisation. In addition, the system enables the anonymised data set to be checked in an audit procedure. The data owners can obtain confirmation via the system that a specific version of the dataset has successfully passed the audit procedure, which enables them to prove that anonymisation has been implemented in a high-quality manner. The expert reports from the crowd-sourced peer review process are generally made publicly accessible.

Key Innovation 6 - Trust Service as middleware for authorised access to pseudonymised data: As an additional service, the Trust Service will provide an interface between data Users and controllers, offering authorised access if information cannot be published anonymously for data pro-

tection reasons or other concerns. In the technical-legal discourse, the extent to which a legally compliant possibility can be created to use data protection-compliant techniques for collaborative data processing and data publication (open data aspect) and the possibilities of de-identification through pseudonymisation are being examined. Partial functionality of the service realises the partial pseudonymisation of data for publication, e.g. through measures of ”attribute-based encryption”, which can be rereleased as soon as the legal requirements (e.g. expiry of deadlines) are met. Furthermore, the Trust Service offers possibilities to carry out evaluations with ”homomorphic encryption” procedures on the pseudonymised (encrypted) information without unauthorised persons gaining access to the unencrypted data set or critical material. This procedure makes it possible, on the one hand, to provide data Users with specific parts of a data set that are necessary for a particular purpose or, on the other hand, to form aggregations across several data sets, e.g. for cohort formation.

1.4 Definitions, Acronyms and Abbreviations

- **Attributes:** In this document, attributes are considered in the context of the input data, which are available as a table in the form of a `.csv` file. Therefore, an attribute consists of an attribute name which corresponds to a column heading and an attribute content which would be the following rows of the corresponding column.
- **Direct Identifier (ID):** IDs are attributes that allow immediate re-identification of an identifiable natural person (Data Subject) [1]. Therefore, due to the GDPR definition of anonymisation, removing the direct identifier is compulsory and usually the first step in any anonymisation of tabular personal data. As IDs usually do not contain evaluable information, no disadvantages have to be accepted by removing them. A more detailed description can be found in the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule [2] by the U.S. Department of Health and Human Services.
- **General Data Protection Regulation (GDPR):** The GDPR is a basic data protection regulation of the European Union (EU). The regulation applies to all organisations that process personal data of EU citizens, regardless of being located in the EU or not. The GDPR applies to the processing of personal data by a controller or processor in the EU, regardless of the location of the processing. It also applies to the processing of personal data of individuals in the EU by a controller or processor not established in the EU if the processing is related to offering goods or services to those individuals in the EU or monitoring their behavior within the EU. The GDPR aims to strengthen the rights of individuals in relation to their personal data and to improve the protection of that data. The regulation sets strict rules for the processing of personal data and obliges organisations to comply with specific standards.
- **Generalization:** In generalization, the level of detail is coarsened. As a result, given attributes of individuals, re-identification in the dataset should be impossible. Further, generalization limits the possibility of finding correlations between different attribute columns and datasets. This also makes it difficult to combine and assigns records to an individual. There are several types of generalization, such as subtree generalization, full domain generalization, unrestricted subtree generalization, cell generalization, and multidimensional generalization. Often, generalization is achieved by generalizing attribute values by replacing parts of the value with a special character, for example * [3].

- **k-Anonymity:** The so-called k-anonymity, first introduced in [4], $k \in N^+,\ k \leq n$ is a dataset property for anonymisation that considers a QI. If the attributes of the QI for each record in the dataset are identical to at least $k-1$ other records in the dataset, the dataset is called k-anonymous. When having k-anonymity, groups consist of at least k records. Technically, k-anonymity is defined by

$$k := \min_{group \in groups} |group|.$$

- **l-Diversity:** First introduced in [5], a common model for anonymisation, considers SAs and gives additional privacy protection to k-anonymity. Again, it considers groups of records with the same QI. When having distinct l-diversity, $l \in N^+,\ l \leq n$, each group has at least l different attribute values for every SAs. Therefore, it is not possible to assign a single attribute value to all records of an equivalence class and group membership does not imply assigning a unique SA to a person. Utilizing l-diversity for scoring anonymisation can be challenging, as it depends on the variety of values a SA can have. Technically, l-diversity is defined by

$$l := \min_{group \in groups} |\{R(j) \mid R \in group\}|,$$

  where $j \in \{1, \ldots, r+t\}$ denotes the column index of the SA.

- **User:** When this document refers to a Zser, the assumption is that the User is a Data Custodian. The Data Custodian is the key User involved in the process of data anonymization within the context of this SRS document. The primary responsibility of the Data Custodian is to upload datasets to the system and receive recommendations on how to effectively anonymize the data to protect individual privacy while maintaining data utility and integrity. The Data Custodian plays a critical role in ensuring compliance with data protection regulations and ethical standards governing the use of sensitive information.

- **Meta Description:** These are information obtained as a result of the system feature meta extractor. They are statistical information calculated for each individual attribute. This information does not contain any personal information but is then used to score the individual attributes and generate the recommendation The meta description is extracted locally from the User's dataset and later sent to the Recommender System.

- **Permutation:** With permutation, the order of individual QIs attribute values within a column is swapped. This reassigns information between columns, potentially breaking important relationships between attributes. This can result in a subsequent deterioration of analyses where the relationships are relevant.

- **Personal data:** Information that relates to an identified or identifiable living individual. In the GDPR personal data "means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person". Different pieces of information, which are collected together can lead to the identification of a particular person and also constitute personal data. Personal data that has been de-identified, encrypted or pseudonymised but can be used to re-identify a person remains personal data and falls within the scope of the GDPR. Personal data that has been rendered anonymous in such a way that the individual is not or no longer identifiable is no longer considered personal data.

For data to be truly anonymised, the anonymisation must be irreversible. The GDPR protects personal data regardless of the technology used for processing that data – it's technology neutral and applies to both automated and manual processing, provided the data is organised in accordance with pre-defined criteria (for example alphabetical order). It also doesn't matter how the data is stored – in an IT system, through video surveillance, or on paper; in all cases, personal data is subject to the protection requirements set out in the GDPR.

- **Pertubation:** In perturbation, additive or multiplicative noise is applied to original data. However, without a careful choice of noise, there is the possibility that utility will be hampered. On the contrary, especially the case of outliers, applying noise might not be enough to ensure privacy after anonymisation by perturbation. Perturbation mostly is applied to SAs. In [6], the perturbation approaches provide modified values for SAs. The authors consider two methods for modifying SAs while not using information about QIs. Besides value-class membership or discretization, the authors use value distortion as a method for privacy preservation in data mining. Probability distribution-based methods might be referred to as perturbation, too. However, because these methods replace the original data as a whole, it can also be assigned to synthetic data.

- **Quasi Identifier (QI):** A QI refers to a set of attributes, where the attributes are not identifiers by themselves, but together as a whole might enable unique identification in a dataset. QIs denotes the characteristics on which linking can be enforced [7]. The QI contains the attributes that are likely to appear in other known datasets, and in the context of privacy models, there is the assumption that a data holder can identify attributes in his private data that may also appear in external information and therefore, can accurately identify QI [4].

- **Sensitive Attribute (SA):** Besides direct identifiers and QI, there are so-called SAs that importantly should not be assignable to individuals after applying anonymisation. Sensitive data is defined as the following: (1) personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs of an individual. (2) trade union membership; genetic data, biometric data processed solely for the purpose of uniquely identifying a natural person; health data. (3) data concerning a person's sex life or sexual orientation [8].

- **t-Closeness:** t-closeness [9] takes into account SA values. Whereas l-diversity considers the variety of SA values in single groups, t-closeness checks the granularity of sensitive attribute values in a single group in comparison to the overall value distribution in the dataset. In the original paper, Li et. al define t-closeness as follows: A group is said to have t-closeness if the (earth mover) distance between the relative frequency distribution of a SA in this group and the relative frequency distribution of the attribute in the whole dataset is no more than a threshold $t > 0$. A dataset is said to have t-closeness if all equivalence classes have t-closeness. Originally, the authors considered the earth mover distance (EMD) for this purpose. The distance is calculated differently for integer, numerical and categorical attributes.

1.5 References

[1] "Art. 4 GDPR – definitions." (), [Online]. Available: `https://gdpr-info.eu/art-4-gdpr/` (visited on 08/24/2023).

[2] Rights (OCR), Office for Civil, Methods for de-identification of phi. hhs.gov. https://www.hhs.gov/hipaa/for-professionals/privacy/special-topics/de-identification/index.html, Accessed on 21.07.2023, Sep. 2012.

[3] A. Majeed and S. Lee, "Anonymization Techniques for Privacy Preserving Data Publishing: A Comprehensive Survey," en, IEEE Access, vol. 9, pp. 8512–8545, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3045700. [Online]. Available: https://ieeexplore.ieee.org/document/9298747/ (visited on 02/07/2023).

[4] L. Sweeney, "K-anonymity: A model for protecting privacy," Int. J. Uncertain. Fuzziness Knowl.-Based Syst., vol. 10, no. 5, pp. 557–570, Oct. 2002, ISSN: 0218-4885. DOI: 10.1142/S0218488502001648. [Online]. Available: https://doi.org/10.1142/S0218488502001648.

[5] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," in 22nd International Conference on Data Engineering (ICDE'06), 2006, pp. 24–24. DOI: 10.1109/ICDE.2006.1.

[6] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD '00, Dallas, Texas, USA: Association for Computing Machinery, 2000, pp. 439–450, ISBN: 1581132174. DOI: 10.1145/342009.335438. [Online]. Available: https://doi.org/10.1145/342009.335438.

[7] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: K-anonymity and its enforcement through generalization and suppression," 1998.

[8] "Welche personenbezogenen Daten gelten als sensibel?" (), [Online]. Available: https://commission.europa.eu/law/law-topic/data-protection/reform/rules-business-and-organisations/legal-grounds-processing-data/sensitive-data/what-personal-data-considered-sensitive_de (visited on 08/24/2023).

[9] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: Privacy beyond k-anonymity and l-diversity," in 2007 IEEE 23rd International Conference on Data Engineering, 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856.

Chapter 2

Overall Description

This chapter briefly overviews the whole EAsyAnon platform and how the Recommender System is integrated into the entire system. It describes how the Recommender System works and interacts with the other systems, introducing the basic functionalities. It also defines what types of stakeholders will use the system and what functionalities are available for the corresponding stakeholders. In the end, the constraints and assumptions of the system are presented. More detailed descriptions of the project requirements and functions are covered in later chapters.

2.1 Product Perspective

The overall system will consist of three components, the Recommendation System, the Audit System and the Trust Service. The Recommendation System provides recommendations on how to anonymise datasets based on meta description provided by a User. After receiving the recommendation, the User would complete the anonymisation process by using appropriate tools and technologies. On the other hand, the Audit System receives an anonymised dataset and evaluates it in a two-step process. The first step is an automated process that must first be passed as a hurdle before being evaluated in a manual audit by a crowd-sourced peer review. The Recommendation System and Audit System are thus interconnected; if an anonymised record fails at the Audit System, it can repeat the cycle based on a new recommendation until it meets the desired requirements. The trust service itself is detached from this system and follows secure computation approaches instead of relying on anonymisation. Here, a dataset is encrypted using methods such as Homomorphic Encryption and analyses are performed on the encrypted dataset. The interaction between the Recommender and Audit system is illustrated in figure 2.1 with the recommendation components at the top and the audit components at the bottom, the User referred to as data owner in the centre as well as the trust service at the right.

2.2 Product Functions

The Recommender System consists of three main components: the Meta Information Extraction Tool (MIET), the Attribute Scoring, and the Recommendation Engine. Each component is briefly introduced here, with a detailed explanation of its functionality.

Meta Information Extraction Tool (MIET): The MIET is the initial step in the Recommender System, where the user provides structured data as a single `.csv` file. This file organizes new entries in rows,

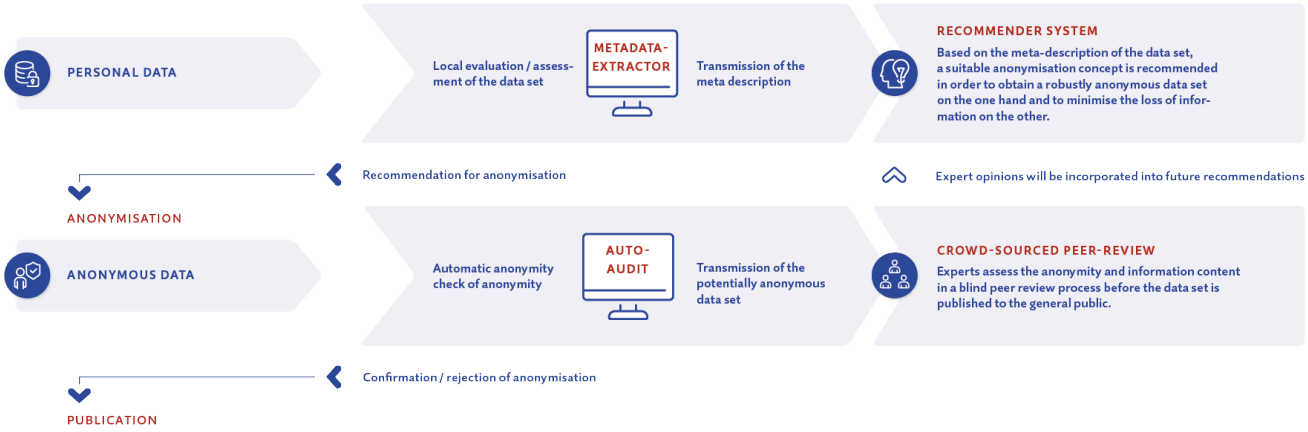

Figure 2.1: General overview of the entire EAsyAnon system (without Trust Service).

with attribute categories as column headers. The MIET processes the `.csv` file offline, extracting the present attributes, their data types, and value ranges and conducting fundamental statistical analyses (see Section 4.3). This meta information is then prepared for transmission to the Recommender System, but only after the user has explicitly agreed to send this data. All dataset analyses performed by the MIET occur locally on the user's machine.

Attribute Scoring: In the next phase, the Recommender System evaluates the attributes of the dataset. This occurs early in the anonymization process and determines the necessary types and levels of anonymization. Unlike the MIET, this phase requires an external connection to a central server. The server receives the standardized metadata extracted locally by the user and generates an initial assessment of the dataset's attributes. The user is then presented with these attributes, categorized by their importance for data utility and sensitivity regarding privacy. The user can make adjustments based on their knowledge and submit the revised scores back to the Recommender System. At no point is personal information transmitted to the server.

Recommendation Engine: The Recommendation Engine is the final component of the Recommender System. It utilizes the Meta Description and attribute scores to generate personalized recommendations on how to anonymize the dataset. The user receives a specific recommendation that can be applied using their chosen anonymization tools.

2.3 User Classes and Characteristics

Different Users will interact with the system, each playing a distinct role. This section outlines the characteristics and responsibilities of the main user group: Data Custodians.

Data Custodians are authorized for managing personal data and overseeing the anonymization process. While a deep understanding of anonymization techniques is not required, they should possess a comprehensive knowledge of the data within their domain, an awareness of the existing data landscape, and the ability to assess the utility of anonymized datasets for Open Data purposes. This requires a nuanced understanding of data attribute sensitivity and the relevance of anonymized datasets, informed by familiarity with previously published and circulating data. The key attributes of a Data Custodian include:

- **Data Management Expertise:** Skilled in data handling, storage, and processing, with familiarity across various data formats and structures, enabling effective management of diverse datasets.
- **Understanding of Data Privacy:** Well-versed in data privacy regulations, such as the GDPR, and ethical considerations, with a focus on protecting sensitive information and privacy rights.
- **Technical Proficiency:** Capable of effectively using the Recommender System, from uploading data files (e.g., `.csv`) to navigating the recommendation process with ease in user interfaces and workflows.
- **Collaboration Skills:** Works collaboratively with stakeholders, including data owners, analysts, and legal teams, utilizing strong communication skills to facilitate discussions, gather inputs, and make informed decisions.
- **Ethical Decision-Making:** Balances the need for data privacy with the utility of data for authorized analysis and research, demonstrating a commitment to ethical considerations in decision-making processes.
- **Analytical Skills:** Assesses the effectiveness of anonymization techniques, evaluates re-identification risks and fine-tunes anonymization strategies to maintain both privacy and data utility.
- **Adaptability:** Manages a variety of datasets and remains adaptable to updates from the Recommender System, staying current with new anonymization practices and techniques.

2.4 File Formats

This section delineates the file format specifications for interaction with the EAsyAnon system by users, specifically Data Custodians. It outlines the primary data storage format and anticipates the incorporation of additional formats to accommodate various data domains, with an initial emphasis on healthcare. The aim is to extend the Recommender System's compatibility with diverse data formats, detailed herein.

`.csv` Data (Comma-Separated Values): In this use case, users are either provided with a `.csv` file or must convert their dataset into this format. The `.csv` format organizes data in a tabular structure, where the first row lists attribute names (columns), and each subsequent row contains data entries. Attributes are arranged into distinct columns to represent the smallest units of information. For example, a full name in one column should be split into separate columns for first and last names. Proper organization of this format is crucial to ensure the atomicity of each attribute and to accommodate variations in the number of attributes and the size of the dataset. An illustrative example of `.csv` data organization is shown in Figure 2.2.
`.csv` files can exist in two distinct formats based on their content:

- **Option 1: User-level Data:** Each line corresponds to a unique data subject.
- **Option 2: Event-level Data:** Each line corresponds to a unique event related to a data subject, but a data subject may appear in multiple events. In this case, an attribute must serve as a unique identifier for the data subjects within the dataset.

Name	Education	Marital-Status
Hans Zimmer	Bachelors	Divorced
Lisa Simpson	Masters	Married
Bert Mustermann	Bachelors	Not-Married

First Name	Name	Education	Marital-Status
Hans	Zimmer	Bachelors	Divorced
Lisa	Simpson	Masters	Married
Bert	Mustermann	Bachelors	Not-Married

Figure 2.2: Example of a sanitized dataset

JavaScript Object Notation (JSON): JavaScript Object Notation (JSON) is a widely used format for hierarchical data, characterized by its ability to represent multiple levels and branches. It is the chosen format for delivering recommendations to data processors (Data Custodians) and for facilitating all internal communications within the EAsyAnon systems.

2.5 Design and Implementation Constraints

The Recommender System comprises numerous components that are interconnected and communicate with one another. Some components must also interface with elements of the Audit System. To facilitate this, it is necessary to define clear interfaces for communication between components. The intended architecture for this communication is shown in Section 3.4. Each component should focus on core functionality and specific tasks, allowing for scalability as required. This project employs the Docker virtualization environment and Kubernetes for orchestration to ensure a suitable and portable environment. Version management is handled using git, along with a Continuous Integration/Continuous Deployment (CI/CD) pipeline. Since personal data is being processed, it is imperative that no unanonymized data leaves the local environment or is sent to the EAsyAnon server.

The user interfaces will be in English, but the system is designed to handle input data in both German and English.

2.6 User Documentation

This section provides an overview of the documentation available to users of the developed Recommender System.

- **User Manual or User Guide:** A comprehensive guide that offers step-by-step instructions on how to operate the software, covering its features and functionalities. It includes screenshots, diagrams, and detailed explanations to enhance User understanding.
- **Troubleshooting Guide:** This document helps users identify and resolve common issues encountered with the Recommender System, providing solutions and troubleshooting tips.
- **Frequently Asked Questions (FAQs):** A collection of common questions and concerns from Users, offering concise answers to help with quick problem resolution.
- **Release Notes:** This document provides updates on software changes, enhancements, and the resolution of known issues, keeping Users informed about the latest developments.
- **Support Contact Information:** Includes details on how to access support or technical assistance, ensuring users can obtain help when needed.

- **Open Educational Resources:** The EAsyAnon project is committed to creating and sharing a variety of resources (e.g., documents, videos, images, and implementations) to help users understand and apply anonymization techniques.

This section outlines the documentation provided to Users of the developed Recommender System.

- **User Manual or User Guide:** A detailed guide offering step-by-step instructions on the software's operation, including its features and functionalities. It is designed to facilitate User understanding through screenshots, diagrams, and comprehensive explanations.
- **Troubleshooting Guide:** This document assists Users in identifying and resolving common issues encountered with the Recommender System, offering solutions and troubleshooting tips.
- **Frequently Asked Questions (FAQs):** A compilation of common queries and concerns from Users, providing succinct answers to facilitate quick problem resolution.
- **Release Notes:** Updates on software changes, enhancements, and resolutions of known issues are documented here, keeping Users informed about the latest developments.
- **Support Contact Information:** Documentation includes details for accessing support or technical assistance, ensuring Users can seek help when necessary.
- **Open Educational Resources:** The EAsyAnon project commits to creating and sharing a variety of resources (e.g., documents, videos, images, and implementations) to aid Users in understanding and applying anonymization techniques.

Chapter 3

External Interface Requirements

This chapter gives a detailed overview of the user interfaces in terms of structure and functionality, the hardware interface with its technical requirements and the software interface with its interactions between the individual Recommender components.

3.1 Workflow

For an overview, the workflow is depicted as a sequence diagram of the Recommender System. Figure 3.1 illustrates the process from starting the browser to receiving the recommendation, thus mapping the entire workflow of the Recommender System. Each action leading from the browser to the user has its own interface and is detailed in Section 3.2. The browser (GUI), `.csv` Import, MIET, Attribute Scoring, Scoring Fine-tuning, and the Recommendation are all system-relevant features and are described comprehensively in Chapter 4. Activities between the browser and the server indicate when there is an online connection to the server, while activities between the user and the browser always occur locally without transmitting any information to the server. The final steps, enclosed within the green frame, can be repeated if necessary. This allows the user to adjust the scoring results and receive a new recommendation if they are not satisfied with the initial one.

3.2 User Interface

This section provides a detailed description of the user interface, which is implemented in a browser environment, including all necessary inputs and outputs. The goal is to create an experience that guides the user step by step through the process, helping them obtain a recommendation for a given dataset. If it is necessary to navigate to other sections, whether forward or backwards, users can move between sections via tabs at the top of the screen.

To start the web application, open the URL `recommendation.easyanon.de`. Upon opening the application, the user is greeted with a welcome screen, as shown in Figure 3.2. On the left side, the name of the application is displayed, while on the right side, a brief description of the functionality that the user can expect from the EAsyAnon Recommender System is provided.

After the initial welcome screen, the user is prompted to select the dataset they wish to anonymize. This can be done by clicking on the upload symbol and selecting the appropriate dataset. Once the dataset is selected, its path is retrieved, and the user is provided with the option to extract its information immediately. However, the user can only proceed if they confirm having read and understood the data protection impact assessment by placing a checkmark on the corresponding

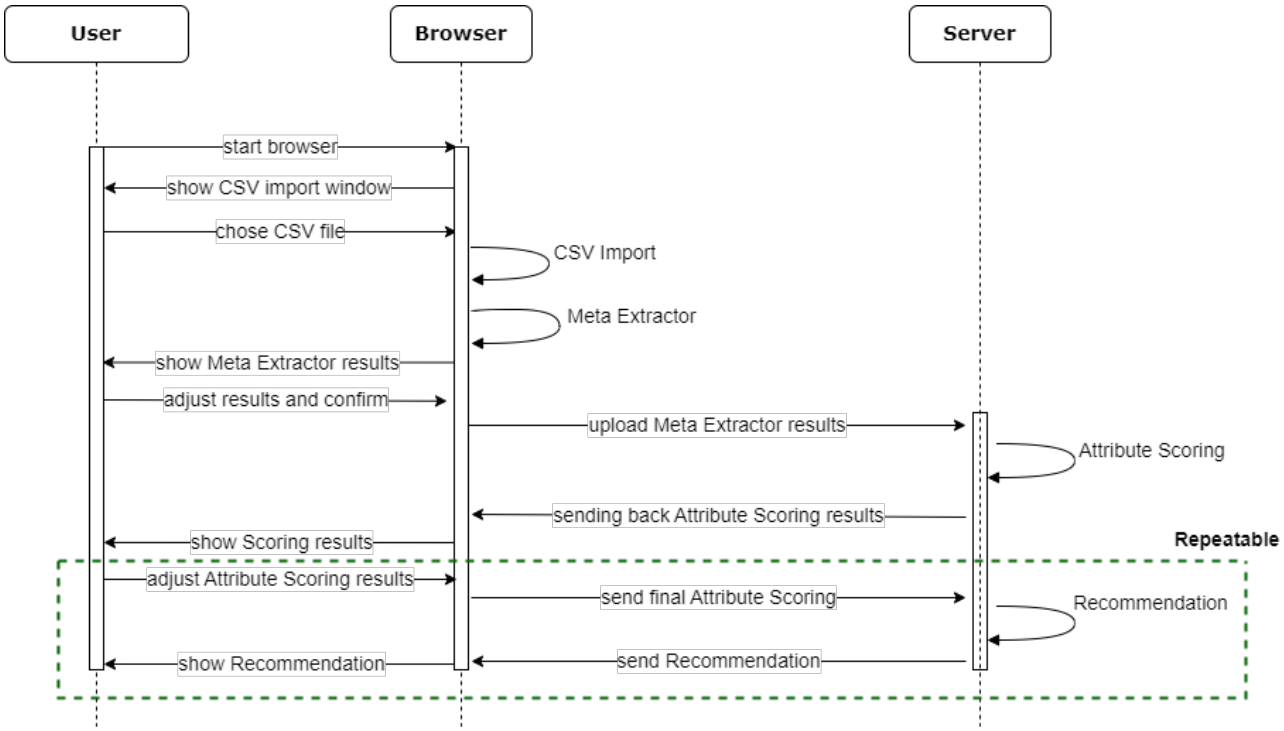

Figure 3.1: Sequence diagram of the Recommender System.

option. This step informs the user about evaluated privacy risks and implemented measures, as detailed in Chapter 7. This process is illustrated in Figure 3.3.

After successfully extracting the dataset's information, the extraction results are presented to the user. This extracted information is crucial for the Recommender System to generate a qualified recommendation for the specific dataset. At this stage, the user has the opportunity to review the extraction results before sending them to the EAsyAnon server. The user must explicitly confirm their intention to send the extracted information to the EAsyAnon server. The user interface for this step is shown in Figure 3.4.

Once the extracted information is submitted to the EAsyAnon server, an initial assessment of the given attributes is provided, as shown in Figure 3.5. The user can specify for each attribute whether it is an ID, QI, SA, or Target. The target attribute describes which attributes are of particular importance for the later evaluation, ensuring their special consideration during anonymization to avoid impairing their utility. Additionally, the user can rate each attribute based on its sensitivity—how much the attribute might contribute to the risk of re-identification—and its importance. Where applicable, the user receives attribute-specific suggestions based on previous ratings (see red-highlighted areas in Figure 3.5). However, using checkboxes and sliders, the user can override the suggestions if they disagree. After clicking the „Continue“ button, the system calculates the recommendation based on the extracted information and the user's assessment.

After the recommendation is calculated, it is presented to the user textually, as shown in Figure 3.6. The user has the option to download a configuration file of the recommendation or to save the entire project, which can later be passed to an anonymization framework and/or the Audit System.

By selecting „Imprint“ in the lower-left area of the screen, the user can access additional general information about the EAsyAnon project. This information includes contact details, legal information, terms and conditions, and more. The structure of this page can be seen in Figure 3.7.

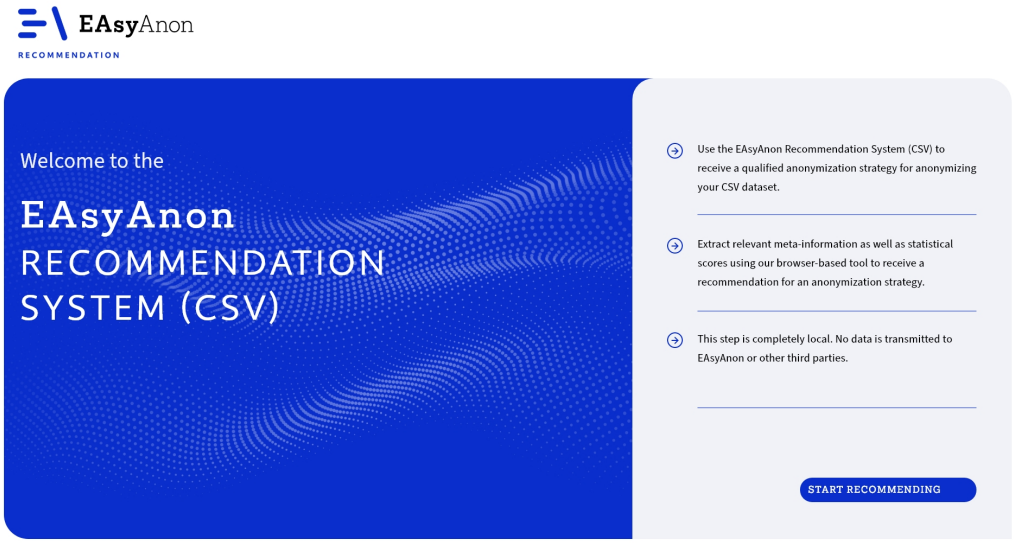

Figure 3.2: Welcome Page

3.3 Hardware Interface

This section lists the system requirements for the user devices and the server.

Client: Since it is not necessary to install software on the user's machine for the Recommender System, the hardware requirements align with those of current operating systems. For software requirements, it is necessary to use a Chromium-based browser and to keep both the operating system and the browser up to date. The minimum requirements are listed in Table 3.1.

Name	Requirement
OS	macOS 13.4.1, Windows 10/11, Ubuntu LTS 22.04 or higher
Browser	Chromium-based, Version: 114.0.5735.110
RAM	8 GB
CPU	Quad Core 3.40GHz

Table 3.1: System requirements for client

Server: Since users can obtain recommendations online without accessibility barriers, the server must meet more stringent requirements. In addition to general performance hardware, certain functionalities are required to ensure smooth and continuous server operation. The minimum requirements are listed in Table 3.2.

3.4 Software Interface

This subsection describes the software interfaces necessary for the operation of the proposed applications. In summary, the web application will utilize a combination of client-side and server-side technologies to provide a feature-rich User interface capable of recommending anonymisation

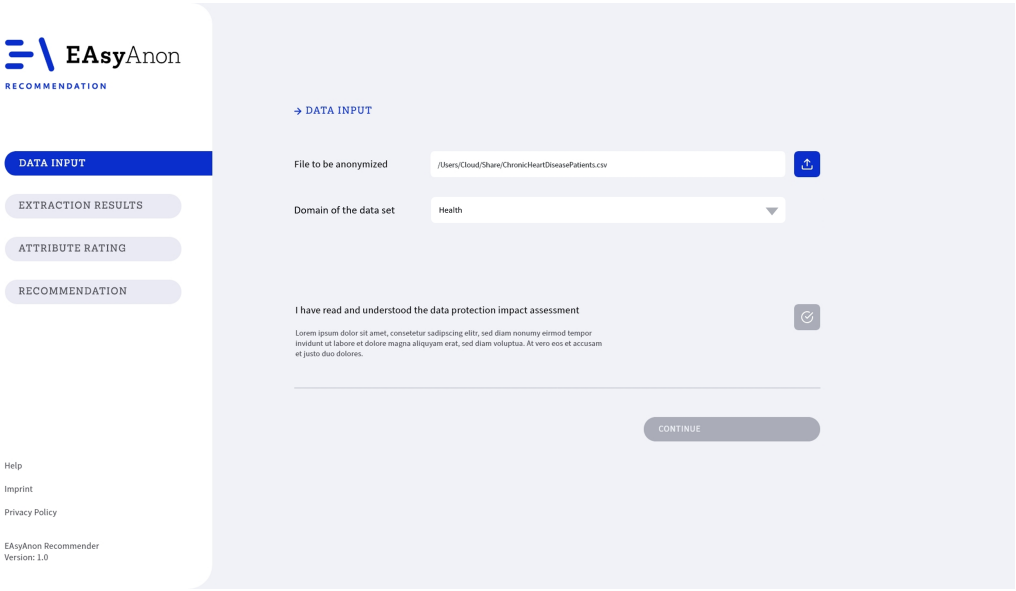

Figure 3.3: Step 1 - Selection of the file to be anonymized.

Name	Requirement
OS	Linux-based OS
RAM	128 GB
CPU	16 x 3.9GHz
GPU	NVIDIA A100 PCIe
Storage	2x 4TB SSD
Remote Management	IPMI
Backup	Cloud backup
Security Features	TPM
Network Interface Card	10 Gigabit Ethernet
Redundant Array of Independent Disks	RAID 1 (mirroring)
Formfactor	Rack-mountable
Database	PostgreSQL

Table 3.2: System requirements for server

methods for (tabular) data. The use of Docker containers for the server-side components and a CI/CD pipeline for automating the build and deployment processes will ensure a robust, scalable, and maintainable system. An overview of how the system is accessed and which parties are involved can be seen in Figure 3.8. The individual components are described in their sections.

3.4.1 Client interfaces

The users can access the Recommender System through a chromium-based browser of their choice. It is ensured that the non-anonymised data from the client is only processed locally in the browser, while metadata and statistics will be sent to the server to obtain a recommendation (`.json`). The appearance of the interfaces can be derived from the mockups in Section 3.2.

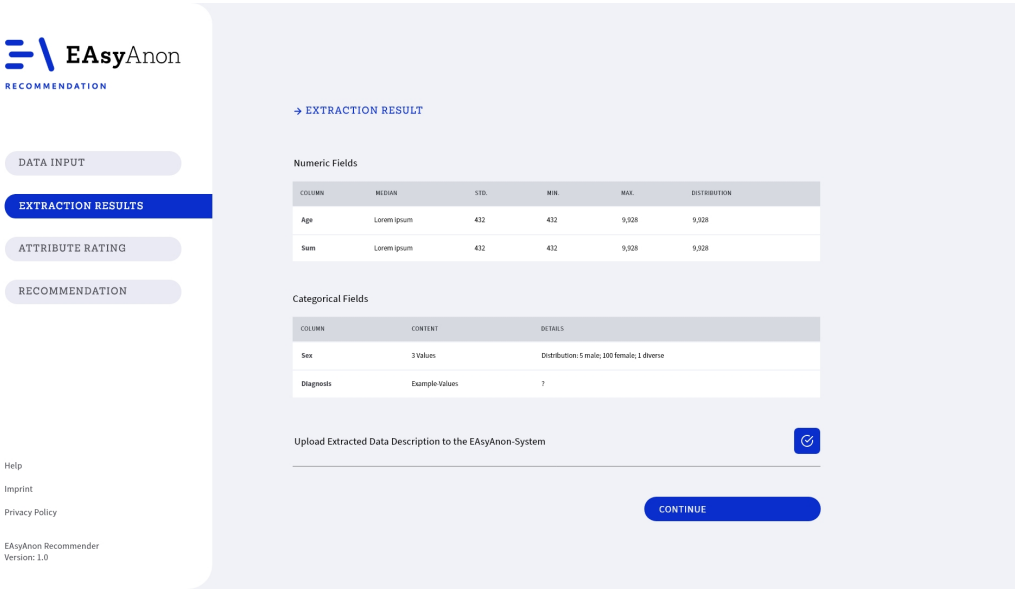

Figure 3.4: Step 2 - Information screen displaying the extracted meta description.

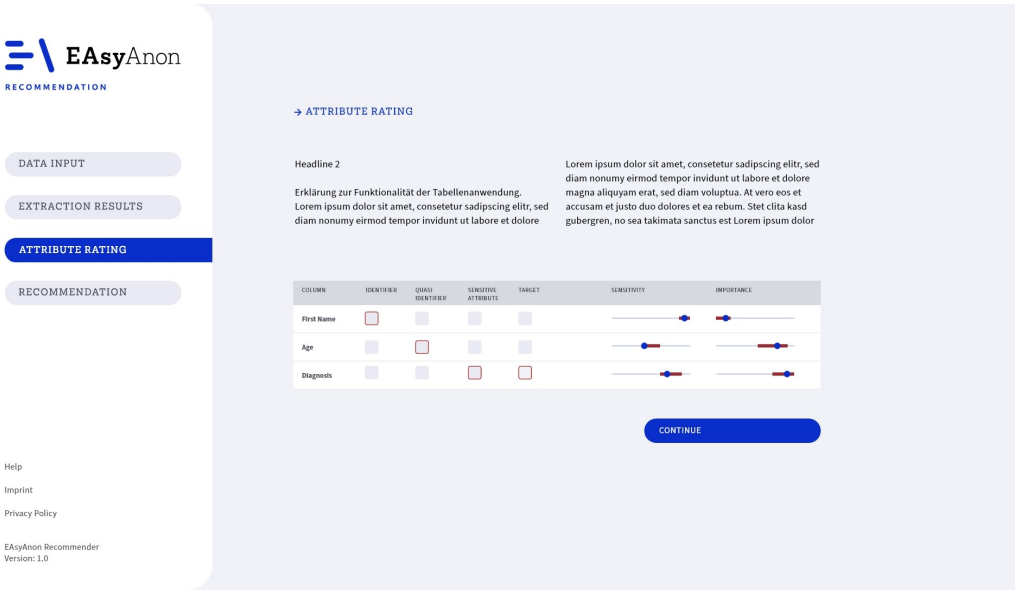

Figure 3.5: Step 3 - Initial evaluation of dataset attributes with options for adjustment.

3.4.2 Server interfaces

Web Framework: The web framework handles incoming Hypertext Transfer Protocol Secure (HTTPS) requests from the clients, processes them, and returns HTTP responses. The PostgreSQL relational database management system stores and retrieves data as required by the application. The server components will be encapsulated within Docker containers to ensure isolation, ease of deployment, and compatibility across different server environments like AWS.

CI/CD Pipeline: The server and client software will employ a CI/CD pipeline. This will automate the process of building, testing and deploying code changes, that are pushed to the source code repository. This ensures that the latest version of the application is always available for testing and

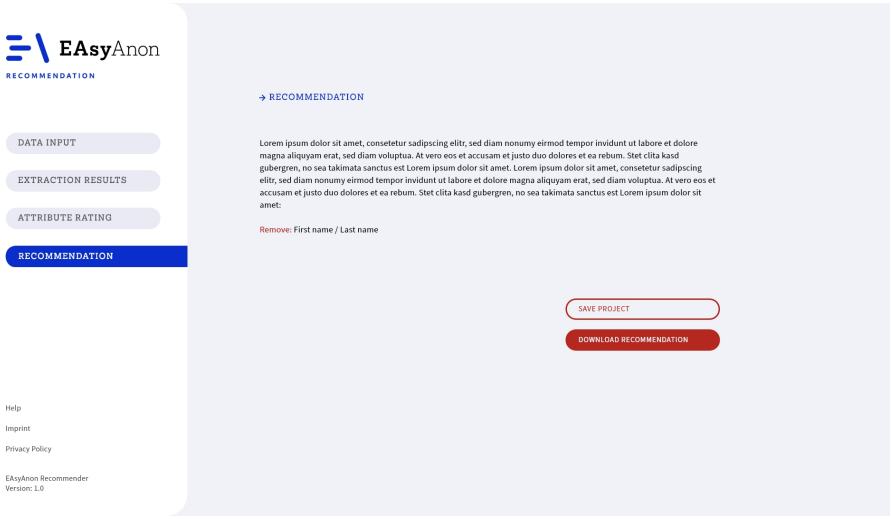

Figure 3.6: Step 4 - Generated recommendation based on the evaluation of attributes with the option to proceed with anonymization.

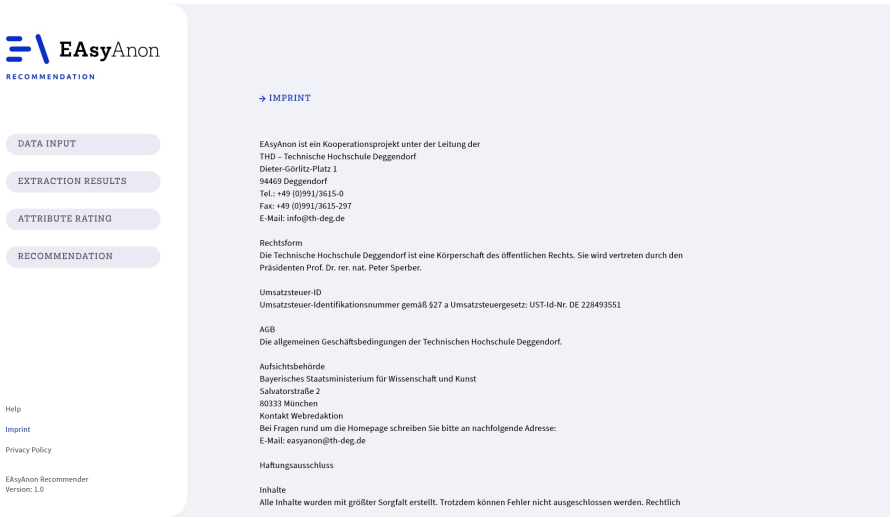

Figure 3.7: EAsyAnon Recommender Imprint

deployment, and it reduces the risk of integration issues.

Docker Interfaces: The Docker containerization platform will be used to deploy different modules described in chapter 4. The server-side software will be packaged into several Docker images, ensuring that the applications, along with all their dependencies, are running uniformly regardless of the underlying system. The Docker images will be built and updated using the Dockerfiles in the source code repository, which outlines the application's environment specifics and dependencies.

Database Interfaces: Databases are only available on the online server. Users have the option to save their recommendation or project locally as a file if they choose to do so. In total, several databases are created, each with a different focus. These are shown in Figure 3.9.
The Recom - DB is the database in which all information is stored that results in a user being given an anonymisation recommendation based on the dataset the User has on their hands. Data stored

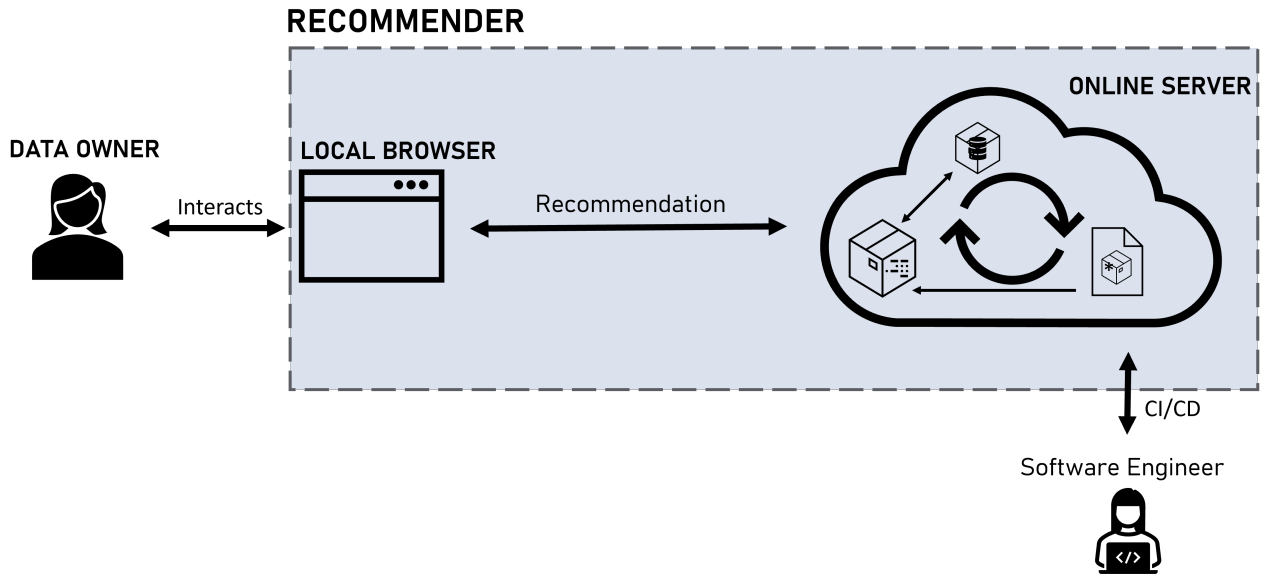

Figure 3.8: Overview of the Recommender System with all involved parties.

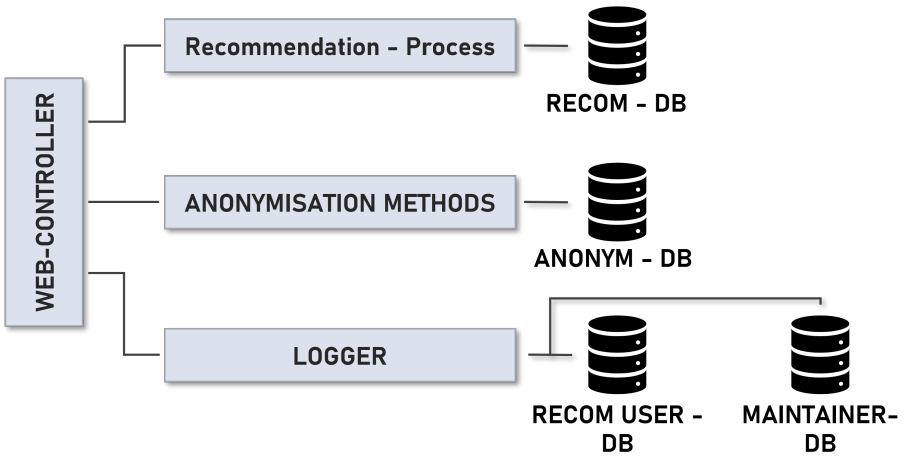

Figure 3.9: Database structure of the Recommender System.

in this database are:

- Meta Extractor Information
- Attribute Scoring from Online Server
- Fine-Tuned Attribute Scoring from User
- The recommendation given for the anonymisation of the dataset

This information will be used to improve the recommendation of anonymization technique, as well as recommendation algorithm.

The Anonym - DB contains the anonymisation methods that are supported by the Recommender System at a given time. New methods can be added or removed during the life cycle of the Recommender System. In addition, two databases are created: Recom User - DB and Maintainer - DB.

These databases hold interactions and error messages generated by Users while interacting with the Recommender System (Recom User - DB), which is used for scientific evaluation of the system and by the maintainer while interacting with the databases on the server (Maintainer - DB).

3.5 Data Flow

The flow of data from the recommendation system can be divided into a recommendation flow and a developer flow. Both flows are described below and can be seen in the figure 3.10.

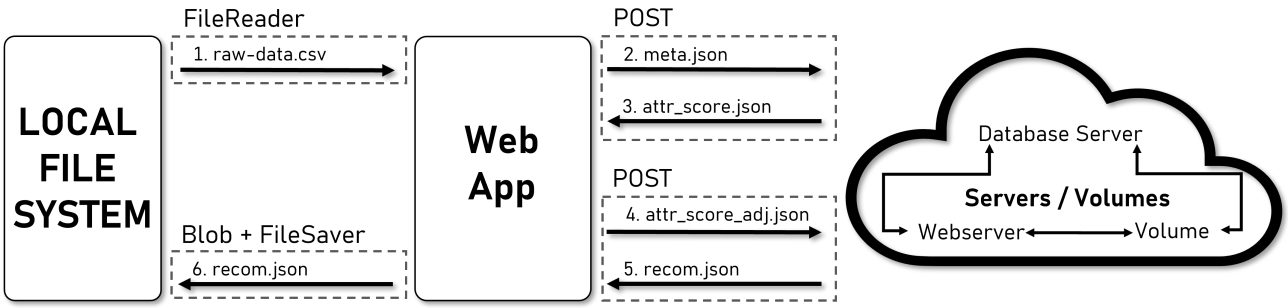

Figure 3.10: Overview of the data flow in the Recommender System.

3.5.1 Recommendation Flow:

The file and command communication with the User is based on the HTTPS. The user locally reads in an un-anonymised `.csv` file via the web app, that is written in JavaScript (1). The app, which runs in the browser, communicates with the web server, by sending a first HTTPS POST request `meta.json` (2) from the User and receives an attribute scoring `.json` file `attr_score_adj.json` (3). In a further POST request, the User sends his updated attribute scoring `.json` file `attr_score_adj.json` (4) file before the server sends his anonymisation recommendation file `recom.json` (5) back. In the last step, the web app enables saving the obtained result into the Local File System (6).

3.5.2 Development Flow:

In a server environment, the application maintains three containers: web server, Database-Server and Volume. The server's source code is version-controlled by Git. The containerized Servers are to be maintained by GitLab.

Chapter 4

System Features

This chapter presents the system features of the Recommender System that need to be implemented. For each feature, a general description is provided (Desc) that introduces the feature, the expected output from the feature (Goal), how the feature is activated (Stimulus/Response Sequences), and what requirements must be satisfied for it to function properly (Functional Requirements). An overview of the features can be seen in Figure 4.1. Features that require a connection to the online service are outlined in red, while those executed locally on the User's machine are outlined in grey.

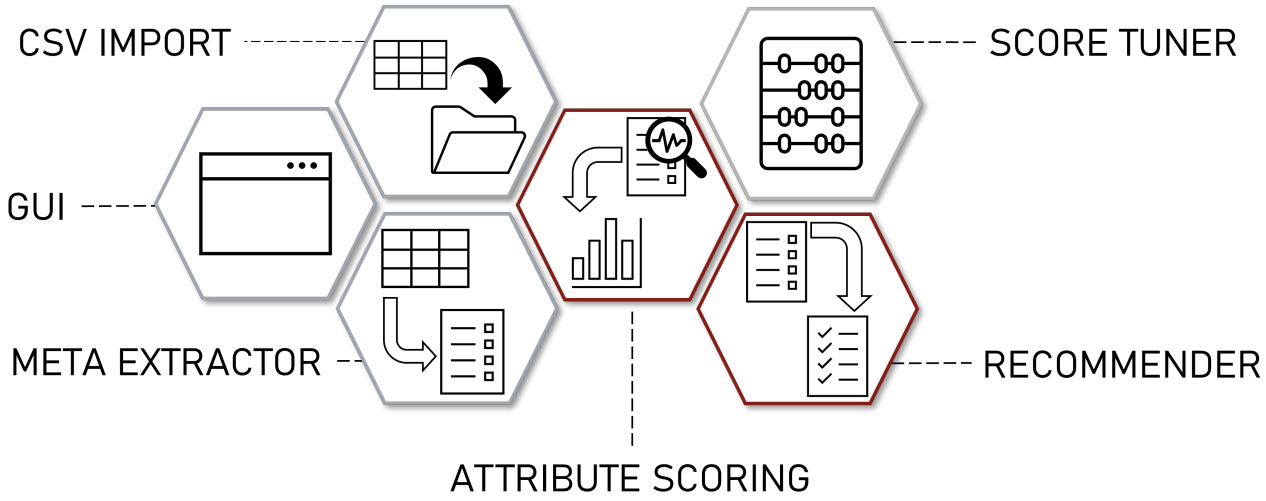

Figure 4.1: Overview of the Recommender System features

4.1 Graphical User Interface (GUI)

Desc: The institution needs a simplified process to get an anonymisation recommendation for its dataset. For this purpose, it should be possible for institutions to access the official website with a browser of their choice to get a recommendation without the need to install any application or the need for authentication. For this purpose, a GUI is developed within a browser that guides the User through the anonymisation process in a straightforward manner. The structure of the individual pages should correspond to the illustrations from Section 3.2.

Goal: Users can get a recommendation for their anonymised record in a straightforward process without having to download an application or authenticate themselves. The screens wich are

required at least are listed below.

- `.csv` Data Input Page, as shown in Figure 3.3
- Extraction Result Page, as shown in Figure 3.4
- Attribute Rating Page, as shown in Figure 3.5
- Recommendation Page, as shown in Figure 3.6

Stimulus/Response Sequences The User visits the official EAsyAnon site, navigates to the recommendation section where he can start the recommendation process.

Functional Requirements

- User must be online
- Functional browser available on the system and usable

4.2 Dataset Import Function

Desc: The User is shown a dialogue screen in which he can browse through his system and select a dataset to be anonymised. After selecting the file and confirming, the `.csv` file is saved as a FileReader object in order to be used later for the Metainformation Extraction Function.

Goal: It must be possible to select a `.csv` file from the User's system in the User's browser as input and as output to save it locally as an object. The entire process is to be realised in HTML and JavaScript. The `.csv` file is loaded into a FileReader object, which can later be used by the Metainformation Extractor. The `.csv` file is not allowed to be sent to the server and may only be stored locally on the User's system. The User is shown a notification informing him about the current upload and processing status.

Stimulus/Response Sequences: As soon as the User visits the website `recommendation.easyanon.de` and clicks on the Button for file selection, a request to select a `.csv` file is displayed. By pressing the upload button, which is located to the right of the file path as shown in figure 3.3, the process starts.

Functional Requirements:

- **File is not a `.csv` file:** Upload is aborted file is not loaded into the FileReader object.
- **`.csv` is not cleaned:** Upload is aborted File is not loaded into the FileReader object.
- **No file selected:** Upload is aborted File is not loaded into the FileReader object.
- **Cleaned `.csv` file selected:** Will be extracted and loaded into the FileReader object.

4.3 Meta Information Extraction Tool

MIET is an independent software tool that supports preparing a report to explain a given health dataset. This tool is a sub-system under the recommender system module. The main function of MIET is to perform a scan on a given health dataset, providing detailed information on column data type, descriptive statistics for each columns, correlation analysis between pairs of column and collecting metadata about the datasets. Finally, the output will be provided in JSON format. Below different functions of MIET are discussed.

4.3.1 CSV File Input

Desc: This feature allows Users to upload CSV files to the system, serving as the initial step for data profiling. MIET supports CSV formats, accommodating different delimiter characters. Figure 3.3 shows the User interface for the CSV file input.
Goal: To enable seamless input of data into the system for further analysis.

- The system must accept CSV files of varying sizes up to 1 GB.
- The system should automatically detect and handle different types of delimiters (e.g., commas, semicolons).
- The system must validate the format of the CSV file upon upload, providing error messages for format discrepancies.

4.3.2 Statistical Information Generation

Desc: Generates essential descriptive statistics for each column in the dataset, including mean, median, mode, minimum, maximum, standard deviation, and null value count. Table 4.1 shows a set of statistical measures which are provided by MIET for numerical and categorical data.
Goal: To provide Users with foundational insights into the distribution and basic characteristics of their data.
Functional Requirements:

- For each numerical column, compute mean, median, mode, minimum, maximum, and standard deviation
- For each column, calculate the count of null or missing values.
- Display the count of unique values for categorical data.
- Detect data type on each column

4.3.3 Correlation Analysis

Desc: Calculates and presents correlation metrics between pairs of columns, helping to identify relationships within the data.
Goal: To help Users understand the interdependencies between different data variables, which can be crucial for further data analysis or feature selection.

- The tool must compute the Pearson correlation coefficient for pairs of numerical columns.
- Include options for Users to select specific columns for correlation analysis.
- The system should handle and report errors or non-applicability in correlation calculations (e.g., for non-numeric data).

Statistical measures	Data type	Insights
Mean	Numerical	Mean is sensitive to extreme (small or large) value
		Shape of data distribution (e.g., skewness)
Trimmed mean	Numerical	Resistant to extreme value
Median	Numerical	Resistant to extreme value
		Shape of data distribution (e.g., skewness)
Mode	Numerical & categorical	Most frequent category or value
Standard deviation	Numerical	Spread of data around the mean
Skewness	Numerical	Indication about long-tail
Kurtosis	Numerical	Peakedness
Minimum, maximum	Numerical	Identify smallest and largest value
Percentiles / Quartiles (Q) / Interquartile range (IQR)	Numerical	Identification of potential outliers
Frequency (count)	Numerical (through binning), categorical	Use as a diagnostic tool (e.g., through visualization detect potential outliers, skewness)
Correlation analysis (e.g., Pearson, Spearman)	Numerical & categorical	Connectedness between variables

Table 4.1: Statistical measurements for different data types

4.3.4 Metadata

Desc: Extracts and provides metadata about the dataset, such as the number of rows and columns, column names, and data types of each column.
Goal: To give Users an overview of the dataset structure, enhancing data understanding and readiness for more detailed analysis.

- Automatically determine and list the data type of each column upon data upload.
- Provide the total count of rows and columns in the dataset.
- Extract and display column names as part of the metadata summary.

4.3.5 JSON Output Generation

Desc: Formats and outputs the results of the data profiling (including statistical information, correlation results, and metadata) in JSON format. Figure 4.2 shows the structure of an JSON file generated by MIET.
Goal: To provide a structured, machine-readable output that can be easily integrated with other systems or used for further analysis.

- Generate a JSON file that encapsulates all derived statistical information, correlation data, and metadata.
- Ensure the JSON output is properly formatted according to JSON standards.
- Provide Users with the ability to download the JSON output directly from the User interface.

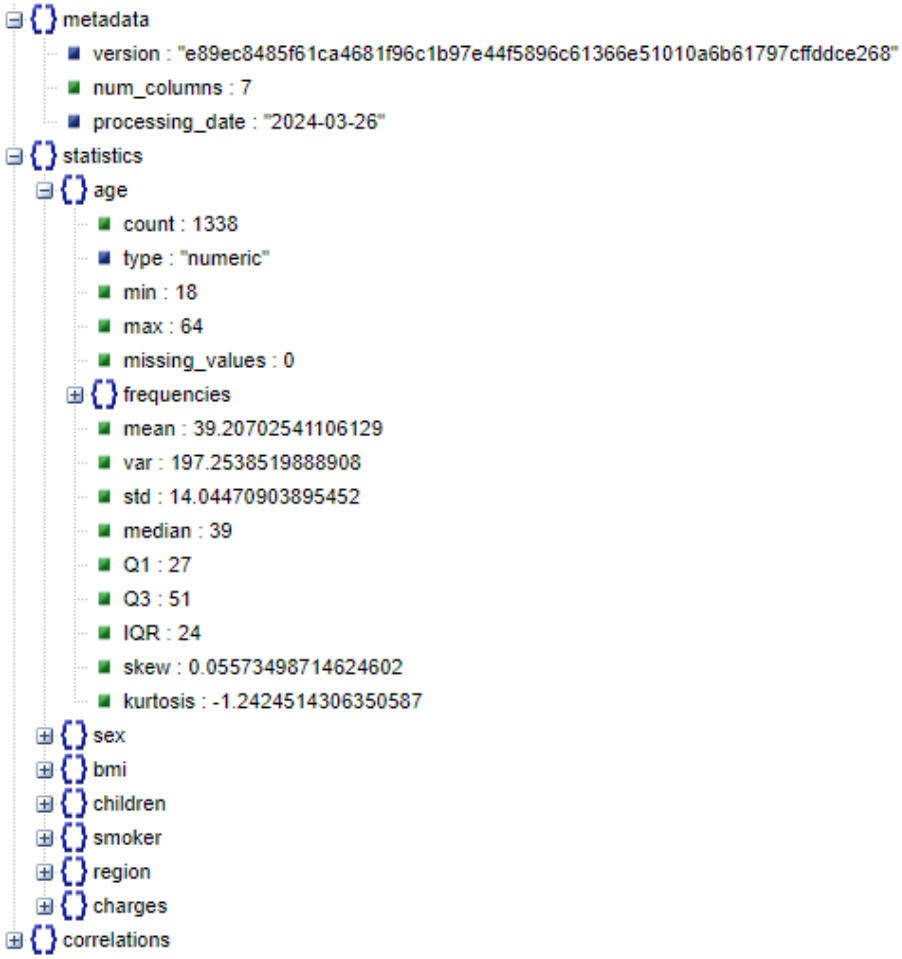

Figure 4.2: Meta JSON file (current version)

4.4 Attribute Scoring

Desc: Attribute Scoring creates an overview of the attributes and an initial rating for the User based on the metadata. Each extracted attribute is scored according to six categories Identifier, QI, SA, Target, Sensitivity and Importance as can be seen in Figure 3.5. The first four identifiers have a value of $\in (0, 1)$ and thus indicate whether an attribute has been classified as an identifier or not. This attribute should be completely removed from the dataset. QIs also range from $\in (0, 1)$ and, in combination with other QIs, can pose a threat to re-identify the dataset. SAs show whether it would be critical if this attribute could be directly assigned to a person or not. While a target emphasises that this attribute is of particular importance for later analyses. Anonymisation procedures are applied appropriately to the individual attributes based on the scoring. Sensitivity has a value range

of $\in [0, 1]$ and describes from the point of view of the recommender how sensitive the information would be if it were to become public and could be reassigned to a person. Importance, like Sensitivity, ranges from $\in [0, 1]$. The focus of this parameter is to show how relevant an attribute is for later analysis.

Goal: A component for an online automatic evaluation is developed that takes the metadata JSON created by Metadata Extractor as input. The online component runs on the EAsyAnon Recommender System server and waits for requests sent by Users via the web interface. The component uses the received meta information to properly categorize each attribute based on the categories presented above (Direct Identifier, Quasi Identifier, etc.) and evaluate their sensitivity and importance. The results are then sent back to the Users as a JSON file as shown in 4.4. Which approaches are most suitable for this purpose will first be evaluated and then subsequently implemented as a dedicated component. The individual categories are listed again below.

- Identifier $\in (0, 1)$; 0 is not an identifier, 1 is an identifier
- QI $\in (0, 1)$; 0 is not a QI, 1 is a QI
- SA $\in (0, 1)$; 0 is not a SA, 1 is a SA
- Target $\in (0, 1)$; ; 0 is not an target, 1 is an target
- Sensitivity $\in [0, 1]$; 0 is not sensitive, 1 is high sensitive
- Importance $\in [0, 1]$; 0 is not important, 1 is very important
- Subject Identifier $\in (0, 1)$; 0 is not an subject identifier, 1 is an subject identifier

Stimulus/Response Sequences: The User triggers the process by pressing the „Continue“ button as shown in Figure 3.4 and by enabling the checkbox „Upload Extracted Data Description to the EAsyAnon system“. This will send an online HTTPS request to EAsyAnon's Recommender System, where the initail attribute scoring will take place.

Functional Requirements:

- **No meta description available:** The process is aborted
- **No internet connection to the EAsyAnon platform available:** Attribute Scoring cannot be performed without EAsyAnon Platform. A message is displayed that no connection can be established.
- **Checkmark „Upload Extracted Data Description to the EAsyAnon System“ not set:** The User is informed by a message that this process is only possible by uploading the information.
- **`meta.json` file is not valid:** The `meta.json` file contains errors and cannot be uploaded due e.g. tampering by a User.
- **All four of the above conditions are met:** Attribute's Scoring function is executed.

```
{
    "scores_recommender": [
        {
            "attribute_name": "age",
            "identifier": "0",
            "quasi_identifier": "1",
            "sensitive_attribute": "0",
            "target": "0",
            "sensitivity": "0",
            "importance": "0.9"
            }
        },
        .....
    ],
    "general_information";{
        "score_recom_version": "",
        "hash_value": ""
    }
}
```


Figure 4.3: Example: A Attribute Score file `score.json` with one Attribute age, which was classified as not an identifier (0), but as a QI (1), not as a SA (0) and not as a target (0). Age is not considered a sensitive (0) here, but has a high importance (0.9) for later evaluations.

4.5 Scoring Finetuning

Desc: After the attribute scoring has been performed, the results are sent back to the User via HTTPS. The results are displayed as shown in Figure 3.5. The estimates from the Recommender System are marked in orange. They move within the value ranges already described for attribute scoring. The difference is that here the User has the possibility to adjust each attribute with a blue slider in a GUI. It is also possible to select or deselect other identifiers and QIs.

Goal: The User is given the opportunity to make adjustments to the attribute scoring results. For each attribute, there are two checkboxes and two sliders with which the User can interact. With the checkboxes, the User is given the option of unchecking the IDs, QIs, SAs or Target assessed by the Recommender System at his own judgement (0) or selecting other attributes (1). In the case of the sliders, a more finely granulated adjustment by the User is possible $\in [0, 1]$. The User is able to reduce the sensitivity and importance of the individual attributes, so that the value goes more towards 0, or increase it further towards 1. Afterwards, the updated values can be sent to the Recommender.

- ID, QI, SA, Target can be selected and deselected for each attribute $\in (0, 1)$
- importance, sensitivity can be moved towards 0 or 1 for each attribute $\in [0, 1]$

Stimulus/Response Sequences: Several interaction options are included. For the identifiers and QIs, it is possible to set a checkmark. If none is available, a click sets one; if one is already available, a click removes it. The sliders for Sensitivity and Importance can be moved left and right by the User. By clicking and holding down the left mouse button, the slider can be moved; by releasing it,

```
{
    "scores_User": [
        {
            "attribute_name": "age",
            "identifier": "0",
            "quasi_identifier": "1",
            "sensitive_attribute": "0",
            "target": "1",
            "sensitivity": "0",
            "importance": "0.9"
            }
        },
        .....
    ],
    "general_information":{
        "score_User_version": "",
        "hash_value": ""
    }
}
```


Figure 4.4: Example: A fine-tuned Attribute Score file `score.json`. The previous values from the attribute scoring were mostly accepted by the User, except that age was determined as a target value and therefore considered by the User as an important attribute for later analysis.

it remains in place. By clicking on the „Continue“ button, the information will be sent back to the Recommender System.

Functional Requirements:

- The initial attribute evaluation must have been carried out by the Recommender so that the User can adjust the values if necessary.

4.6 Recommender Function

Desc: The Recommender Function is the core of the Recommender System. Together with the output from Meta Extraction Function and the fine-tuned attribute score, the User receives a personalized recommendation for the dataset. Noted that, to receive recommendation, meta description of dataset shall be forwarded to Recommender System instead of the original dataset. The recommendation procedure takes place online. The output from the User's local machine will be used as input into the Recommender System which is located in the cloud. The recommendation will be forwarded to the User again in a `.json` file format.

Goal: The key objective of the recommender function is to generate personalized recommendations based on the dataset a User has and forward the recommendation to the User in `.json` file format. Based on this recommendation the User will complete a data de-identification task with his/her responsibility. To complete this task, the User can choose a suitable tool. Regarding

generating recommendations, a knowledge-based recommendation technique can be considered. Below there are some potential privacy models and anonymization techniques mentioned. The generated `.json` should have the following format as seen in Figure 4.5.

- Privacy Models:
  - **K-Anonymity:** $k > 10$
  - **T-Closeness:** $t = 0.5$
  - **L-Diversity:** $l \in N^+$, $l \leq n$, each group has at least l different attribute values for every SA.
- Anonymisation Methods:
  - Removal of direct identifiers
  - Generalization
  - Permutation
  - Pertubation

Stimulus/Response Sequences: After pressing the „Continue“ button as seen in the Fine Tuning of the attributes in Figure 3.5, the metadata and the results of the attribute scoring are sent to the online recommender server. The recommendation process is then started automatically. Once the recommendation process is complete, the results are fed back to the User, allowing them to save the results, as shown in Figure 3.6.

Functional Requirements:

- Online connection must be available
- Metadata must be available
- Attribute scoring results must be available and valid

```
{
    "recommendations": [
        {
            "quasi-identifier": [
                "attr_1",
                "attr_2"
            ],
            "generalization": [
                "attr_1": [["sub_attr_1_1", "sub_attr_1_2", "sub_attr_1_3"],["sub_attr_1_4", "
                    sub_attr_1_5"]],
                "attr_2": [["sub_attr_2_2", "sub_attr_2_3"],["sub_attr_2_4", "sub_attr_2_5"]]
            ],
            "suppression": [
                "attr_1": ["sub_attr_1_7", "sub_attr_1_8", "sub_attr_1_9"],
            ],
            "eliminating_direct_identifiers": [],
            "permutation": False,
            "perturbation": [],
        },
        {
            "quasi-identifier": [
                "attr_3",
                "attr_4",
                "attr_5"
            ],
            "generalization": [
                "attr_3": [["sub_attr_3_4", "sub_attr_3_5"]],
                "attr_5": [["sub_attr_5_2", "sub_attr_5_3"],["sub_attr_5_6", "sub_attr_5_11"]]
            ],
            "suppression": [],
            "eliminating_direct_identifiers": [],
            "permutation": False,
            "perturbation": [],
        },
        {
            "quasi-identifier": [
                "attr_6"
            ],
            "generalization": [],
            "suppression": [],
            "eliminating_direct_identifiers": [],
            "permutation": False,
            "perturbation": [],
        }
    ],
    "general_information":{
    "recommender_version": "",
    "hash_value": ""
    }
}
```


Figure 4.5: Example: Recommendation file, referred to as `recom.json`, generated to guide the anonymisation of a given tabular dataset with ≥ 6 attributes. Each object in the recommendations array of the file represents a set of attributes, that form a QI. For each QI, the recommended anonymisation methods are listed. For instance, the QI formed by the attributes `attr_1` and `attr_2` could be anonymised using the "generalization" or "suppression" methods.

Chapter 5

Functional Requirements

This section outlines the functional requirements for the Recommender System, focusing on scalability, updating mechanisms, and the logging system. The requirements ensure that the system is flexible, maintainable, and capable of handling large datasets and user interactions efficiently.

5.1 Scalability of the Recommender Server

Description: The Recommender System is designed to suggest anonymization procedures for datasets that vary in content and data type. Although the system is initially developed for specific use cases, it must be extensible to accommodate new datasets with different data types and content in the future. Additionally, as recommendations may evolve over time, the system must allow for the incorporation of new procedures.

Goal: The Recommender System should be scalable to include new privacy models, anonymization procedures, and dataset types as they become necessary.

Stimulus/Response Sequences:

- **Stimulus:** HTTP request for attribute scoring.
- **Response:** The system processes the request and returns the appropriate attribute scores.
- **Stimulus:** HTTP request for recommendations.
- **Response:** The system processes the request and returns the recommended anonymization procedures.

Functional Requirements:

- The server must be maintained in a distributed version control system to facilitate scalability and versioning.

5.2 Update Mechanism for the Recommender

Description: After the initial release of the Recommender System, ongoing support is necessary to adapt to changing requirements for recommendations and anonymization procedures. Continuous improvement of the recommendations is expected over time. Therefore, the system must include mechanisms to ensure it remains up-to-date, providing users with the best possible privacy protection.

Goal: The Recommender System should be optimized continuously based on user input, particularly for the following functions:

- Attribute Scoring Function.
- Recommender Function.

Stimulus/Response Sequences:

- **Stimulus:** An update process is triggered, either manually or automatically.
- **Response:** The system applies updates to ensure that the latest functionalities and recommendations are available.

Functional Requirements:

- The system's functions must be designed to allow rapid updates with minimal disruption to users.

5.3 Logger System

Description: The logger system is responsible for recording and storing general user actions and errors. Additionally, it records results from various system features, such as attribute scores, adjustments to these scores, and anonymization recommendations. The logger must handle large volumes of data and store them efficiently in a robust database such as PostgreSQL.

Goal: The logger system aims to create a comprehensive and reliable log of user actions, errors, and results from attribute scoring and recommendations. These logs will be crucial for system improvements, analysis, and audits.

Functional Requirements:

- Capture and store general user actions performed within the system.
- Capture and store error details whenever an error or exception occurs.
- Capture and store adjustments made to attribute scores within the system.
- Capture and store anonymization recommendations generated by the system.

- Integrate with a scalable database system capable of efficiently handling large volumes of log data.
- Ensure proper data organization, indexing, and storage to facilitate efficient retrieval and analysis of logged information.

Stimulus/Response Sequences:

- **Stimulus:** A user action is performed within the system.
- **Response:** The logger records the action, along with relevant details such as timestamp and context.
- **Stimulus:** An error or exception occurs during system operation.
- **Response:** The logger records the error, including relevant details such as error type, message, and stack trace.
- **Stimulus:** An attribute scoring value is modified by the user.
- **Response:** The logger records the adjustment, including the original and new values.
- **Stimulus:** A recommendation for anonymization is generated based on attribute scores.
- **Response:** The logger records the recommendation, including details such as the recommended anonymization technique and privacy model parameters.

Functional Requirements:

- **User Actions:** Log entries must include details such as timestamp, action description, and relevant system context.
- **Error Occurrences:** Log entries must include information such as error type, timestamp, error message, stack trace, and relevant system context.
- **Attribute Scoring Adjustments:** Log entries must include details such as the attribute being adjusted, the original value, and the new adjusted value.
- **Anonymization Recommendations:** Log entries must include details such as the recommended anonymization technique for each attribute and the parameters of the privacy model used.

Chapter 6

Performance Requirements

The speed and efficiency of the recommendation process and its updates are crucial, as the system operates in real-time. In simpler terms, the system must be fast enough to process data promptly. To achieve real-time functionality, the system needs to respond immediately after receiving the meta description from the user's local machine. Additionally, our web service must be capable of handling multiple users simultaneously.

This chapter outlines various evaluation metrics that will ensure the performance and effectiveness of the system-of-interest. First, the performance requirements for the entire system will be explained, followed by different evaluation metrics for the recommendation algorithm.

6.1 Performance Requirements (whole system-of-interest)

- **Response Time:** The Recommender System shall provide recommendations to the user (Data Custodian) promptly after receiving a request. The system should process bulk recommendation requests for a batch of users within 1 second per 100 users.
- **Throughput:** The system shall support a minimum of 100 recommendation requests per minute. The Recommender System should be able to handle concurrent user requests efficiently without causing queuing delays.
- **Redundancy and Failover:** The Recommender System shall incorporate redundancy mechanisms and failover procedures to ensure high availability and fault tolerance.

6.2 Performance Requirements (recommender algorithm)

- **Evaluation Methods:** The Recommender System component shall be evaluated through both offline and online methods, as well as experiments with selected user groups. This includes the verification of the recommendation algorithm and preparing a representative dataset.
- **Offline Evaluation:** Offline evaluation can be performed without any interaction with real users. Different evaluation metrics can be chosen based on the recommendation algorithm.
- **Online Evaluation:** After offline evaluation, an online evaluation should be conducted to ensure the quality of the recommendations.

Chapter 7

Security Requirements

Security requirements are an essential aspect of any software development project and, therefore, significantly impact the Recommender System. This chapter describes the requirements that protect sensitive information, prevent unauthorized access, and ensure overall integrity and trustworthiness. Each section is structured to provide a brief description of the subject matter, followed by a list of the various requirements necessary to ensure the security of the Recommender System.

7.1 Authentication and Authorization

The authentication and authorisation requirements are only valid for access to the Git repository in which the versions of the Recommender System are managed and the PostgreSQL database of the Recommender System. Users who need an anonymisation recommendation for their dataset do not need to log in and therefore do not need to register.

Authentication Requirements:

- The Recommender System itself does not require User registration. For the registration of new Users with Git and PostgreSQL, the provided functionalities are used and do not have to be developed.
- Passwords for authentication with the database are only stored in the PostgreSQL system database according to the intended PostgreSQL guidelines. Users' private passwords are not allowed to be stored in any locations in the Recommender System's file system.
- For the creation of passwords, password managers have to be used to generate secure passwords according to industry standards. Passwords should have a minimum length of 16 characters, containing upper and lower case, numbers and special characters.
- A password history policy will be enforced to prevent Users from reusing their previous passwords.
- After a defined number of 5 consecutive failed login attempts, accounts will be locked out until unlocked by an administrator again.
- A secure password recovery mechanism, with email verification, will be implemented if a User forgets his password.

- As a Git repository, the GitLab Community Edition hosted by the Deggendorf Institute of Technology `https://mygit.th-deg.de/` will be used.

Authorization Requirements:

- In terms of Git roles, the provided roles Guest, Reporter, Developer, Maintainer, Owner and Minimal Access from GitLab will be used.
- With PostgreSQL, the required access rights for database Users are assigned individually via targeted privileges.
- The principle of least privilege is applied to both the Git roles and the PostgreSQL privileges, and Users are only given exactly the rights they need to be able to fulfil their tasks.

7.2 Data Security

This section describes the requirements for transferring the Recommender System's information, especially when data is sent online to the server. It also describes how the data should be encrypted locally at the User's site.

Key Requirements:

- Data sent to the Recommenders System is sent exclusively via HTTPS with the associated Transport Layer Security (TLS) protocol.
- The Recommender System will validate server certificates during the TLS handshake to prevent man-in-the-middle attacks and ensure the authenticity of the server.
- Files that the User can store locally via the Recommender System do not have any special encryption, as they do not contain any personal information or general critical information. The institution can then manage the files themselves according to its existing policies.

7.3 Access Control

In this section, the requirements for access control are defined. A distinction is made between access for a user to obtain a recommendation for their dataset and access to the database of the Recommender System server by developers.

Key Requirements:

- Users who interact with the system only have the option of obtaining a recommendation for anonymisation based on their dataset. Thus, they only have access to the functions necessary for this purpose.
- Users do not have access to the database of the Recommender System at any time.
- Access to stored data will be strictly controlled based on user roles and privileges defined.
- Developer authentication is required to access any sensitive information.

7.4 Auditing and Logging

As there are many different requirements for auditing and logging, the requirements in this section have been divided into different categories to provide a better overview. The requirements include recording the actions of authorized database users, errors that occurred during operation, and all necessary information from uploading the meta description to the actual recommendation. Furthermore, it is also specified how this data is to be stored and for how long.

Audit Trail Requirements:

- The system will maintain an audit trail that logs all database events of the Recommender System, including but not limited to login attempts, user access, data modifications, and administrative actions.
- Each database log entry will include a timestamp, user identifier, and a description of the event.

Log Access and Storage Requirements:

- Access to audit logs is restricted to authorized developers only, with appropriate access controls and user permissions.
- Audit logs shall be stored securely and protected in the PostgreSQL Database from unauthorized access, tampering, or deletion.

Retention Period Requirements:

- The system will retain audit logs for a minimum of 12 months to facilitate security investigations and for further improvement possibilities regarding the Recommender System.
- After the retention period elapses, old logs will be archived and securely stored.

Log Integrity and Tamper Detection Requirements:

- Mechanisms will be implemented to ensure the integrity of audit logs and alert administrators in case of tampering attempts.
- Any detected tampering incidents will be logged and reported to administrators immediately.

General Log Requirements:

- Critical system events, errors, and exceptions will be logged, indicating the event type, timestamp, and relevant context for troubleshooting and analysis.
- User activities, such as extracted meta descriptions, attribute scores, user adjustments, and generated recommendations will be recorded and used for later improvements.
- The system will provide configurable logging levels (e.g., INFO, DEBUG, ERROR) to allow administrators to adjust the level of detail captured in the logs based on system needs.
- Logging mechanisms should be designed to have minimal impact on system performance to ensure efficient operation while capturing essential security information.

7.5 Error Handling and Reporting

Error handling is often underestimated in many applications, although it is a central component of a secure application. This section presents the requirements for the Recommender System with regard to error handling. The requirements are also divided into individual groups and describe the requirements that are important for the User, the internal exception handling, and the reporting of such exceptions.

User-Friendly Error Messages Requirements:

- The system shall provide clear and concise error messages that are user-friendly and easy to understand.
- Error messages should avoid technical jargon and provide actionable information to assist users in resolving the issues.

Exception Handling Requirements:

- Proper exception handling will be implemented to catch and handle errors systematically without crashing or exposing sensitive information.
- A workflow will be defined for error resolution, including steps to reproduce the error, possible mitigation actions, and resolution timelines.
- The system will include recovery mechanisms to attempt automatic resolution of specific errors when possible or provide guidance to users for manual recovery steps.

Reporting Requirements:

- In the event of an error that impacts user experience or functionality, users will be provided with appropriate notifications to inform them of the issue and any potential workarounds.
- Critical errors and exceptions will trigger real-time alerts and notifications to system administrators for immediate attention and resolution.

7.6 Secure Storage

Data is stored on the Recommender Server, which should be used later to improve the system sustainably. For this reason, requirements are set out in this section that enable the secure storage of data.

Key Requirements:

- Input validation will be implemented and parameterized queries used to prevent SQL injection attacks on the database. (Protection Against Injection Attacks)
- User input will be sanitized to prevent malicious scripts from being stored in the database and displayed to other users. (Protection Against Cross-Site Scripting)

- Files containing sensitive or confidential information will be stored securely and restricted to authorized user access.
- Configuration files containing sensitive information (e.g., database credentials, API keys) will be stored securely with restricted access.
- Data integrity checks will be implemented to detect any unauthorized alterations or corruption of stored data.

7.7 Software Updates and Patching

The Recommender System is intended to be operated beyond the project duration, therefore it is necessary to perform regular updates to meet security standards and provide a smooth user experience. The requirements in this section focus on software updates and patching. Included are requirements to keep used libraries up to date, rollback capabilities, and a defined patch process with associated documentation and testing.

Key Requirements:

- The storage solutions used by the application (e.g., databases, file servers) will be kept up to date with the latest security patches to address known vulnerabilities.
- Rollback will be supported to restore previous versions in case the update causes critical issues or compatibility problems.
- A defined process for managing patches will be developed to address known security vulnerabilities and bugs.
- Patches for critical security issues will be prioritized and deployed promptly.
- Software updates will be accompanied by detailed patch notes describing the changes, bug fixes, and security enhancements included in the update.
- Before releasing updates, the software development team will conduct code reviews and testing to ensure the updates do not introduce new security vulnerabilities or bugs.
- The development team will maintain a testing environment to validate patches before deployment to the production system.

7.8 Security Testing

In addition to functional tests of the Recommender System, it is also necessary to carry out security tests at regular intervals and after updates. The requirements include penetration tests, vulnerability analyses, code reviews, and the development of suitable test scenarios.

Requirements:

- Regular vulnerability assessments will be conducted for known security issues and weaknesses.
- The code will be reviewed to identify and rectify security weaknesses and potential vulnerabilities.
- A comprehensive set of security tests will be defined and executed during the testing phase to validate the effectiveness of implemented security controls.
- After updates, regression testing shall be performed to verify that the fixes did not introduce new issues.
- Testing will be conducted to validate the secure handling of file uploads to prevent potential security risks associated with malicious files.
- The application's secure communication protocols will be tested to verify the proper implementation of encryption and validation of certificates.

7.9 Compliance and Regulations

During the initial phase, the main focus of the Recommender System will be on the healthcare sector in the European region. Therefore, requirements are listed to ensure that the developed platform will meet the regulations in these areas.

Key Requirements:

- Relevant data protection regulations and standards must be adhered to, such as the General Data Protection Regulation (GDPR) and the Federal Data Protection Act (BDSG), ensuring the lawful processing of user data.
- The Recommender System must not infringe on any intellectual property rights or use third-party intellectual property without proper authorization.

7.10 Disaster Recovery and Business Continuity

This section describes the requirements for the Recommender System to ensure that the system can be restored to operation after an emergency situation.

Requirements

- Daily backups of the Recommender System database, as well as any additional critical data, will be performed and stored securely at an offsite location.
- A disaster recovery plan will be established to ensure data availability and continuity of operations in the event of a system failure or breach.

Chapter 8

Privacy Requirements

This chapter focuses on the privacy requirements of the Recommender System, addressing three primary risk factors. It provides a detailed explanation of the stages at which attributes must be considered QIs and the metrics that must be adhered to in privacy models. Finally, the chapter presents the DPIA, outlining the individual risks that remain when using the Recommender System.

8.1 Risk-assessment on the basis of three factors

When using the Recommender System, it is essential to consider potential risks of re-identification. In the context of data protection, the Art. 29 Data Protection Working Party criteria offer a useful framework for evaluating these risks. The three primary risk factors—singling out, linkability, and inference—are critical in any anonymization process.

Singling out refers to the risk of individuals being identified from anonymized data records that have been generalized. The goal of legally secure anonymization is to create groups large enough that individual assignment of attributes to a single person is no longer possible. This requires sufficiently large datasets with similar attributes while preserving the dataset's information content to avoid rendering other attributes unrecognizable or falsified.

Linkability involves the risk of individuals being identified by combining two individually anonymous datasets. To prevent this, it is crucial to ensure that datasets preserve anonymity in their entirety. With each new publication, already anonymized and published data should be included in the evaluation.

Inference is the risk of drawing conclusions from the published dataset that could infer information about an individual. To mitigate this risk, anonymization must be carried out in a way that prevents any such inferences.

Overall, it is crucial to design and use the Recommender System with careful consideration of these potential risks. By adhering to relevant policies and regulations and ensuring that the anonymization process addresses the primary risk factors and passes the motivated intruder test, the system's stability can be guaranteed, and the privacy of individuals involved can be protected.

To ensure legally compliant anonymisation and to make a recommendation, it must be ensured that the dataset is first analysed. Since security against linkage attacks can only be guaranteed if only a few variables with a high re-identification risk are present in a dataset. These must be evaluated beforehand using a rating system. Three evaluation points are crucial here: Replication, Availability, Distinguishability.

8.2 Evaluation of the attributes

All variables are listed and evaluated within the framework of three case groups. The assessment ranges from low (1), medium (2) to high (3). The first category for the individual variables is „replication“, in which the information is assessed according to how consistently it appears in connection with a person. A low score is given to measured blood pressure, while a high score is given to a person's date of birth. The assessment is based on how statically and consistently a corresponding value is linked to a person. The second group is concerned with the ”availability” of the information. The decisive factor here is how available this information or variable is for third parties to re-identify. In this context, it must be considered and anticipated how far-reaching additional knowledge is available to the data User. Therefore, laboratory values of a person are difficult to obtain, whereas, the address or age are quite easily acquired. The last category concerns „distinguishability“, according to which it is possible to assess how people can be distinguished from each other by means of individual values. For example, a ZIP code with a complete reproduction is to be classified as higher than a shortened reproduction of it. Once this assessment has been made for the individual variables of the dataset, all attributes with an overall score of > 5 are to be classified as high-risk identifiers. After this classification, the anonymisation methods are to be applied.

8.3 Privacy Models

To meet the legal requirements for the robustness of re-identification, the privacy model k-anonymity is used as anonymisation technique. The known weaknesses of this privacy model in inference attacks can be closed by adding t-closeness.

k-anonymity is able to protect individuals from singling out. If the attributes of the QI for each record in the dataset are identical to at least $k - 1$ other records in the dataset, the dataset is called k-anonymous. When having k-anonymity, groups consist of at least k records. According to the recommendations of the Article 29 Data Protection Working Party, $k > 10$ is chosen in their Opinion 05/2014 on Anonymisation Techniques as the k-value on the high-risk identifiers. This chosen k-value can also protect against linkability since the probability is $1/k$ that correlations about affected individuals are made between records in a k-group.

To prevent information disclosure from inference attacks on SAs, the Article 29 Data Protection Working Party proposes the usage of t-closeness. Unlike their recommendation on k-anonymity, there is no specification for the value of t. In order to provide a working basis for subsequent re-identification risk quantification, t is set equal to 0.5 on SAs, as in the Lean European Open Survey on SARS-CoV-2 infected patients (LEOSS).

8.4 Data Protection Impact Assessment (DPIA)

According to Article 35 (1) of the GDPR, a DPIA is required for processing activities that pose a high risk to the rights and freedoms of data subjects. The scope of the DPIA extends to anonymisation as it involves a processing activity. Its applicability is indicated since it includes the processing of health data, which qualifies as a special category of personal data under Article 9 (1) of the GDPR, in conjunction with Article 35 (3) (b) of the GDPR. This DPIA aims to assess the privacy risks associated with the Recommender System.

1. **Recommended anonymisation methods:**
   The recommended anonymisation methods align with the guidelines specified in Section 8.3.

Please note that these recommendations are not legally approved by a court or an officially binding institution. They are merely guidelines based on successful examples of data anonymisation. As such, there remains a possibility of incomplete anonymisation from a legal standpoint. Additionally, the effectiveness of these methods may vary depending on the unique characteristics of each dataset, necessitating a case-by-case evaluation. This is due to the ongoing unclear situation. Currently, anonymization cannot be carried out in a legally secure and automated manner. The system serves only as guidance and support. It is essential to acknowledge that the ultimate responsibility for data anonymisation lies with the User of the system. Therefore, we strongly recommend consulting relevant authorities before publishing any data following these recommendations. Such consultations can ensure alignment with the most current legal and regulatory requirements and help mitigate potential risks associated with data anonymisation.

2. **Assessment of benefits and sensitivity:**
   The evaluation of the benefits and sensitivity of the data is delegated to the data holder. As the data holder possesses the most comprehensive knowledge of the data and its value, they are best suited to determine the potential benefits derived from the processing activities and assess the sensitivity of the data. This approach ensures that the assessment incorporates the data holder's expertise and perspective.

3. **Data ownership and minimal data sharing:**
   Personal data remains with the data holder, and only metadata is shared. This significantly reduces the risk of data breaches or unauthorized access. The focus on metadata minimizes the exposure of personal information, thus enhancing privacy protection.

4. **Protection against attacks:**
   The Recommender System itself is safeguarded against attacks through technical and organizational measures. (see section 7) These measures aim to ensure the system's resilience and protect against potential security threats, maintaining the confidentiality and integrity of the processed data.

5. **Documentation of anonymisation and attribute assessments:**
   The recommended anonymisation techniques and attribute assessments will be documented and stored within the system. This documentation serves as a record of the applied anonymization methods, enabling transparency and accountability in the data processing activities.

6. **Non-storage of personal data:**
   It is important to note that personal data is not stored within the system. Only anonymised and metadata are processed and used for generating recommendations. This approach further reduces privacy risks and safeguards individuals' personal information.

Through this DPIA, the aim is to identify and address potential privacy risks associated with data processing activities. By adhering to recommended anonymisation methods, involving the data holder in assessing benefits and sensitivity, and implementing security measures, the privacy of individuals is safeguarded. The documentation and non-storage of personal data contribute to compliance with data protection regulations and the promotion of privacy-conscious practices.

The DPIA for the Recommender System can be found on the system's website for data owners to access and review. Transparency and disclosure of privacy practices are of utmost importance. Therefore, the DPIA is conveniently located on the website, providing data owners with detailed information regarding the conducted data protection impact assessment. Data owners are encouraged

to periodically review the DPIA to stay informed about the evaluated privacy risks and implemented measures.

Chapter 9

Software Quality Attributes

Software quality attributes are specific characteristics and properties of a software system that define its overall quality and performance. By defining these attributes with measurable criteria, developers can create software that meets the expectations and needs of end users while adhering to the project's technical requirements and constraints. This systematic approach enhances communication, guides development efforts, and facilitates the evaluation and verification of the software's conformity to the desired attributes.

9.1 Scalability

Description: The Recommender platform, as an online service used by users, must be scalable to handle increased user traffic when the platform goes live. The developed system must be capable of dynamically adapting its resources to ensure consistent performance, response time, and reliability.

Key Requirements

- **Horizontal Scaling:** The system must support the addition of servers or instances to distribute the increasing load, ensuring scalability without significant loss of performance.
- **Performance Monitoring:** Continuous performance monitoring tools should be implemented to measure system performance, response times, and resource utilization. This allows for the identification of bottlenecks and triggers scaling actions when necessary.

Quality Metrics

- **Response Time under Load:** Users should receive their recommendations promptly after uploading their dataset.
- **Average Waiting Time for Recommendations:** The system should track and calculate the average time users wait to receive recommendations for their datasets.
- **Throughput:** The system's ability to process a specific number of requests within a given time frame should be monitored. The throughput can be calculated using the formula $Throughput = \frac{Number\ of\ Transactions}{Time\ Taken}$.
- **Resource Utilization:** Monitoring of system resources such as CPU, memory, and disk space should be conducted to ensure they perform efficiently under the workload.

9.2 Availability

Description: The availability of the Recommender platform refers to the degree to which the system, service, or application is operational, accessible, and ready to perform its intended functions when users require it. High availability ensures that users can consistently access and use the system, minimizing disruptions and enhancing the user experience.

Key Requirements

- **High Uptime:** The system must maintain a high uptime percentage, ensuring it is available and operational most of the time.
- **Fault Tolerance:** The system should be designed to handle failures gracefully, continuing to provide service even if certain components or subsystems fail.
- **Redundancy:** Redundant components or backup systems must be in place to ensure continuous operation in case of hardware or software failures.
- **Disaster Recovery:** A robust disaster recovery plan should be established to recover from major incidents or catastrophic failures and restore services as quickly as possible.
- **Monitoring and Alerting:** Comprehensive monitoring and alerting mechanisms should be implemented to proactively detect potential issues and notify relevant personnel for quick resolution.

Quality Metrics

- **Availability Percentage:** The target availability percentage should be at least 90%. This means the system should be available and operational for at least 90% of the time over a specific period, such as a month or a year. This level of availability ensures that users can access and use the system with minimal disruptions, with the remaining 10% downtime allowing for scheduled maintenance.
- **Failure Rate:** The failure rate should be less than 0.1 failures per 1,000 hours of system operation, indicating a highly reliable system with infrequent failures.
- **Load Handling:** The system's load-handling capacity should be tested at different levels of user traffic and demand. The target load-handling capacity should be at least 100 concurrent users with consistent response times, ensuring the system can handle varying levels of load without performance degradation.
- **Response Time:** The system should process bulk recommendation requests for a batch of users within 1 second per 100 users.

9.3 Correctness

Description: Correctness refers to the degree to which the Recommender System accurately fulfills its intended purpose and requirements. A correct system behaves as expected, producing

precise and reliable recommendations, and adhering to specified rules and constraints. Ensuring correctness is essential for building trust in the system and its effectiveness in providing accurate anonymization recommendations to users.

Key Requirements

- **Accurate Evaluation:** The Recommender System must accurately extract metadata from the provided datasets and recommend the most appropriate anonymization methods based on the data's characteristics and the intended use case.
- **Use Cases and User Stories:** Develop comprehensive use cases and user stories that detail the interactions between users and the system. These scenarios should be thoroughly validated to ensure that the system behaves correctly in all expected situations.
- **Specifications and Design Documentation:** Maintain precise and up-to-date documentation that outlines the architecture, design, and specifications of the system. This documentation serves as a reference for developers and testers to ensure the system behaves as intended and meets its design goals.

Quality Metrics

- **Accuracy:** The system's accuracy should be measured by the percentage of correct and appropriate anonymization recommendations provided relative to the total number of datasets processed. The target accuracy rate should be at least 90%, ensuring that the majority of recommendations are precise and relevant.
- **Completeness:** The target completeness rate should be 100%, ensuring that all identified functionalities and scenarios are covered by the system's capabilities.
- **Defect Density:** Measure the number of defects found during testing or post-release relative to the total lines of code or modules. The goal is to reduce defect density over time, aiming for a reduction of at least 10% per release.
- **Error Rate:** The target error rate should be less than 2%, indicating a high-quality system with minimal errors in its operation.
- **Validation and Verification Coverage:** Aim for a validation and verification coverage of at least 95% to ensure comprehensive testing of the system's functionality and correctness.
- **Test Coverage:** Strive for a test coverage rate of at least 80%, ensuring that the majority of the codebase has been thoroughly tested to identify and correct potential issues.
- **Compliance with Standards:** Conduct regular audits to ensure the system adheres to industry standards, best practices, and relevant regulations. The goal is to achieve 100% compliance, ensuring the system is robust, reliable, and compliant.

9.4 Flexibility

Description: Flexibility refers to the system's ability to accommodate modifications, scale to varying workloads, and integrate seamlessly with other systems. It allows users to configure settings and parameters without requiring extensive code changes and employs adaptive algorithms to make real-time adjustments. Ultimately, flexibility enables agility and resilience, allowing organizations to remain relevant and competitive in dynamic and evolving environments.

Key Requirements

- **Modularity:** The system should be designed with modular components that can be easily added, removed, or replaced without disrupting overall functionality. This design supports quick adjustments and updates.
- **Configurability:** The ability to configure various aspects of the system without extensive code changes is crucial. This includes parameters, settings, or rules that can be easily modified by users.
- **Adaptive Algorithms:** Algorithms and decision-making processes should be adaptable to changing conditions or inputs, enabling the system to respond effectively to dynamic scenarios.
- **Standardization and Openness:** Adherence to open standards and interfaces promotes flexibility by ensuring compatibility with diverse technologies and reducing the risk of vendor lock-in.

Quality Metrics

- **Configurability Index:** The configurability index measures the percentage of parameters or settings that can be configured without code changes. A target metric for a good configurability index is to have at least 90% of parameters configurable without code modifications.
- **Module Independence:** Module independence can be assessed on a scale of 1 to 10, with 10 representing the highest level of independence. The target score for module independence should be at least 8.
- **Integration Capability:** Integration capability can be measured on a scale of 1 to 5, with 5 representing the highest integration capability. The goal should be to achieve a score of at least 4.
- **Failure Recovery Time:** Failure recovery time measures the time required to recover from a failure, typically in minutes. The system should incorporate redundancy mechanisms and failover procedures to ensure high availability and fault tolerance, minimizing downtime.

9.5 Interoperability

Description: Interoperability refers to the ability of different systems, software, or devices to seamlessly communicate and work together, exchanging data and information efficiently and coherently. It involves establishing common standards, data formats, and protocols to ensure smooth interactions

between diverse technologies. The ultimate goal of interoperability is to enable interconnected systems to collaborate effectively, promoting data sharing, compatibility, and ease of integration across various platforms and environments.

Key Requirements

- **Standardization:** Adopting and adhering to widely accepted industry standards is crucial for ensuring interoperability. Standards help ensure that data formats, protocols, and interfaces remain consistent and compatible across different systems.
- **Data Formats and Protocols:** Systems must utilize compatible data formats and communication protocols to facilitate seamless data exchange. Common examples include data formats like XML, JSON, and `.csv`, and communication protocols such as HTTP, REST, and SOAP.
- **Interface Definitions:** Clear and well-documented interface definitions are essential for enabling different systems to understand each other's capabilities and communicate effectively. This often involves using APIs (Application Programming Interfaces) or other contract-based communication methods.
- **Compatibility Testing:** Rigorous testing is necessary to verify that systems can successfully interact with each other. Compatibility testing should cover various scenarios, data types, and error conditions to ensure reliability.
- **Upgradability and Versioning:** As systems evolve, it is essential to maintain backward compatibility and support upgrades without disrupting interoperability. This process should be managed using version control systems like GitLab.

Quality Metrics

- **Conformance to Standards:** Conformance to standards should be measured on a scale from 1 to 100, with 100 indicating full adherence to all relevant industry standards, data formats, and communication protocols. The goal should be to achieve a score of at least 90.
- **Usability and User Experience:** Usability and user experience can be assessed using a standardized usability scale, such as the System Usability Scale (SUS). The target average SUS score should be at least 70.
- **Success Rate:** The success rate measures the percentage of successful interactions between different systems or components. The target success rate should be at least 95%.
- **Data Accuracy:** Data accuracy should be measured to ensure at least 98% accuracy in data exchanges, ensuring the reliability of the information being shared.

9.6 Maintainability

Description: Maintainability refers to the ease and efficiency with which a software system can be modified, updated, repaired, or enhanced throughout its lifespan. A maintainable software design and codebase are well-organized, thoroughly documented, and adhere to coding standards, making it easier for developers to understand, implement changes, and fix issues. By prioritizing

maintainability, software teams can reduce long-term costs, increase productivity, and ensure the system remains adaptable and responsive to evolving requirements and technological advancements.

Key Requirements

- **Modularity:** The software should be organized into distinct, cohesive modules, allowing changes to be isolated and minimizing the risk of unintended side effects.
- **Readability:** Code should be written clearly and concisely, using meaningful variable names, comments, and well-structured code blocks.
- **Documentation:** Comprehensive and up-to-date documentation should be provided to explain the software's architecture, design decisions, and interfaces.
- **Consistency:** The codebase should follow consistent coding standards and conventions to improve readability and maintainability.
- **Loose Coupling:** Dependencies between components should be minimized, allowing changes to one module without affecting others.
- **High Cohesion:** Modules should have a single, well-defined responsibility, improving code maintainability and reusability.

Quality Metrics

- **Cyclomatic Complexity:** Cyclomatic complexity measures the number of independent paths through the code. A target score of less than 10 for each function or method is ideal.
- **Code Duplication:** This metric indicates the amount of duplicate code. Duplication should be minimized to prevent inconsistencies and reduce maintenance efforts, with the goal of less than 5% of the codebase being duplicated.
- **Code Churn:** Code churn refers to the frequency of code changes. A high churn rate may indicate unstable code that requires constant maintenance. The goal is to maintain a churn rate of less than 10% per week.
- **Test Coverage:** Test coverage measures the percentage of code covered by automated tests, providing insights into the stability and maintainability of the software. The aim should be at least 80% code coverage.
- **Code Review Effectiveness:** This metric assesses the quality of code reviews, ensuring potential issues are caught before they become maintenance problems. A good benchmark is to have at least 90% of critical issues identified and addressed during code reviews.

9.7 Portability

Description: The portability of the software refers to its ability to run on different platforms and environments without requiring major modifications or adjustments. It should be accessible to users on various operating systems and browsers.

Key Requirements

- **Platform Independence:** The software must be designed and developed to be platform-independent, ensuring it can run on major operating systems such as Windows, macOS, and Linux.
- **Browser Compatibility:** The software should be compatible with popular web browsers, particularly those based on Chromium, such as Chrome, to provide a consistent user experience across different environments.

Quality Metrics

- **Platform Coverage:** The software should be tested and verified to run smoothly on all major operating systems, including Windows, macOS, and Linux.
- **Browser Compatibility:** The software should achieve compatibility with at least the major Chromium-based browsers to ensure a consistent user experience.

Quality Metrics

- **Platform Coverage** The platform coverage should include all major operating systems like Windows, MacOS and Linux.
- **Browser Compatibility** The coverage should be aiming for at least 95% compatibility with widely used browsers, such as Chrome, Firefox, Safari and Edge.

9.8 Reliability and Robustness

Description: The software is designed to provide reliable and consistent performance, even when handling large datasets. It incorporates robust error-handling mechanisms to minimize the impact of errors and is built to be resistant to various types of attacks.

Key Requirements

- **Handling Large Datasets:** The system must efficiently manage large datasets while maintaining optimal performance during recommendation processes. This ensures that, regardless of dataset size, the system operates smoothly without compromising reliability.
- **Error Handling:** The software includes robust error detection and handling mechanisms to gracefully manage unexpected errors or exceptions during data processing. Errors should be appropriately logged, and the system should continue functioning without causing significant disruptions.
- **Graceful Degradation:** In the event of failures or resource constraints, the software should adopt a graceful degradation approach, allowing it to continue functioning with reduced capacity and performance rather than completely failing.

- **Resistance Against Attacks:** The system must demonstrate robustness against various types of attacks, including DDoS (Distributed Denial of Service), SQL injection, and cross-site scripting. It should implement strong security measures such as input validation, encryption, and intrusion detection to mitigate potential threats and safeguard against unauthorized access or manipulation of sensitive data. The system should maintain its functionality and protect user information even under sustained attack attempts.

Quality Metrics

- **Mean Time Between Failures (MTBF):** The target MTBF should be set to at least 10,000 hours, indicating higher reliability and longer intervals between failures.
- **Failure Rate:** The failure rate should be measured as the number of failures per 1,000 hours of system operation. The target failure rate should be less than 0.1 failures per 1,000 hours, indicating a highly reliable system.
- **Fault Tolerance:** The system should be designed with a fault tolerance level of at least 99.99%. This means that even in the presence of faults or errors, the system should maintain correct functionality in nearly all cases.
- **Security Testing:** The system should undergo rigorous security testing to ensure its resilience against attacks. The product should achieve a security rating of at least 95%, indicating its robustness in protecting sensitive data and maintaining confidentiality and integrity.

Chapter 10

Business Rules

The following business rules are specific statements that define the operational policies, constraints, and decision-making criteria within the organization. Although they are distinct from functional requirements, these rules significantly influence the software's functionality and may necessitate additional requirements to enforce them effectively. They also serve as guidelines for how different users or roles can interact with the software system. It is important to note that the developed software is intended as a non-profit web application designed to ensure consistency, conformance, and user-centered functionality.

10.1 User Roles and Permissions

- The product shall support multiple user roles, including administrators, database administrators, and Data Custodians.
- Administrators shall have the authority to manage user roles, permissions, and system configurations.
- Data Custodians shall have access to perform data anonymization tasks, including receiving recommendations on anonymization methods based on input data.

10.2 Data Ownership and Access

The Recommender System accesses only the metadata provided by the Data Custodian to generate personalized recommendations. The MIET is employed to extract this information. Since the datasets remain under the Data Custodian's control, they retain full authority over their data, allowing them to enforce their data management policies and structures effectively. Our software is designed to be compatible with the Data Custodian's existing policies and structures, ensuring the data's integrity and confidentiality. We prioritize respecting these access rights within our software, ensuring that only the provided metadata is accessed for generating recommendations.

10.3 Anonymisation Recommendations

- The product shall recommend anonymisation methods based on the nature of the input data.

- The recommended anonymisation methods shall comply with relevant data privacy regulations, standards, and best practices.
- The product shall take into account the sensitivity and uniqueness of data attributes when suggesting anonymisation techniques.
- The recommendations shall consider the trade-off between data utility and privacy preservation to ensure an appropriate level of anonymisation.

10.4 Compliance with Privacy Regulations

- The product shall adhere to applicable data privacy regulations, such as the GDPR.
- The product shall enforce privacy rules and restrictions specified by the data owner or administrator during the process.

10.5 Audit Trail and Logging

- The product shall maintain an audit trail of all activities, including User actions, data modifications, and system events.
- The audit trail shall capture relevant metadata, such as timestamps, User identities, and the nature of the performed actions.
- The product shall provide logging capabilities to record any errors, warnings, or exceptions encountered during the anonymisation process.
- Additionally, the product shall support extended logging functionalities for scientific purposes and evaluation.

10.6 Documentation and Training

- The product shall provide comprehensive documentation on the business rules, including User roles, permissions, and access control policies.
- The documentation shall include guidelines on applying anonymisation techniques, interpreting anonymisation recommendations, and ensuring compliance with privacy regulations.
- The product shall offer training materials and resources to educate users on the proper usage of the software and the enforcement of business rules.

10.7 Open Source System

The proposed system will follow an open-source development model, encouraging collaboration, transparency, and community engagement. The GNU Lesser General Public License (LGPL) (LGPL) stands as a crucial licensing option for developers seeking to distribute their software under an open-source framework while maintaining flexibility in usage and distribution. One of the key features

of the LGPL is its compatibility with proprietary software, making it a preferred choice for libraries and components intended for integration into both open-source and proprietary projects. This compatibility arises from the LGPL's permissive nature, allowing developers to link their software with libraries licensed under the LGPL without mandating the release of their entire codebase under the same terms. This aspect encourages collaboration and innovation by enabling the seamless incorporation of LGPL-licensed components into a wide range of projects. This open approach aligns with our commitment to shared knowledge and innovation. Users will have the freedom distribute, and enhance the software while adhering to the terms of the LGPL.