Remarks

22: state of the art dla aplikacji sieciowych

23: Dlaczego aplikacja webowa?

Silesian
University
of Technology

SILESIAN UNIVERSITY OF TECHNOLOGY

FACULTY OF AUTOMATIC CONTROL, ELECTRONICS AND COMPUTER SCIENCE

PROGRAMME: INFORMATICS

Final Project

Data anonymisation web platform

author: Szymon Pluta

supervisor: Krzysztof Simiński, PhD DSc

Gliwice, January 2022

Contents

Abstract

The text of the abstract should be copied into a respective field in the APD system. The Abstract with keywords should not exceed one page.

Keywords: 2-5 keywords, separated by commas

Chapter 1

Introduction

Technological advancement being observed in the past years fundamentally changed the relevance of data in today's digitalized world. Information became an innovation stimulus in the area of research and development. The quantity of data that organizations produce, process, store and share is at a continuous growth. An enormous amount of 1.8 zettabytes ($1.8 \cdot 10^{21}$ bytes) of new data was produced only in the 2011, and every two consecutive years this number is doubling [52]. After decades of observed technological advancement and innovation, the global internet traffic finally entered the zettabyte-era, as it had reached a magnitude of one zettabyte in 2016, and in the calender month being as early as September [11].

The vast quantities of processed information allowed for brand new research fields such as data science or big data analytics to form, which are used by organizations to derive new insights in a way that was previously impossible. Organizations collect and process the data to enhance the services they provide to the customers through statistical analysis or newly developed computer science processes including data mining and machine learning. The utility of delivered services is increased at a lower cost and improved efficiency through the insights extracted from the collected information about how the services are consumed [5].

Następne do zrobienia, w uproszczeniu i w podanej kolejności:

- Jak widać, dane są używane wszędzie. Przeciętna osoba sobie nawet nie zdaje sprawy.
- Istota prywatności danych, ochrona danych osobowych.

- Prawo wolno reaguje na uregulowanie ochrony danych osobowych: GDPR dopiero w 2018. Prawo różnie działa w różnych krajach.
- W czasach narastających danych, dane muszą być zanominizowane, bo...
- ...ale:

Wniosek: generyczny sposób anonimizacji - nie ma takiego.

- Mogą istnieć dobre algorytmy anonimizacji, ale są optymalale jedynie w określonych kontekstach
- **Cel pracy**: stworzenie generycznego rozwiązania do anonimizacji baz danych w dowolnej formie danych (nieważne co dane reprezentują).
- Kontroler danych zna swoje dane i sam dostosowuje optymalny sposób anonimizacji.
- Rozwiązanie może być dostarczone jako free-access (research), B2C lub nawet B2B.
- Zakres pracy, opis rozdziałów - na samym końcu, po zrobieniu innych rozdziałów.

- introduction into the problem domain
- settling of the problem in the domain
- objective of the thesis
- scope of the thesis
- short description of chapters
- clear description of contribution of the thesis's author – in case of more authors table with enumeration of contribution of authors

Chapter 2

Problem analysis

2.1 Data explosion

2.1.1 Technology advancement

The continuous and rapid exponential growth of data being collected globally is further excited by improvements to the overall population's accessibility to the digital technology [7]. Cisco Systems estimates within its annual report [10] that 66 percent of the world population will have an access to the web by the 2023, compared to 51 percent in 2018, whereas the number of devices that are connected to the web will reach a staggering value of three times as many as the entire population size – demonstrating a total of 60 percent expansion when compared to 2018. Even the area of mobile connection, which was established long ago, is still sustaining a growth – by 2023, a mobile connectivity will be a privilege for 70 percent of the world's population, compared to 66 percent in 2018. The global average mobile network speeds will be tripled through a rapid increase from 13.2 Mbps in 2018 to 43.9 Mbps in 2023.

2.1.2 Data analytics

Raw representation form of the data is not interpretable until it is put under a context and processed into practical information. Acquiring relevant insights and conclusions from the information can be achieved through a wide term of analytics,

which encompasses the actions needed to be performed to produce new information, including: analysis, filtering, processing, grouping and contextualizing the information. Newly discovered knowledge is inferred from the produced information. Apart from the processes, analytics also includes the technologies, methodologies and algorithms to use and could be divided into descriptive analytics, diagnostic analytics, prescriptive analytics and predictive analytics [7].

2.1.3 Big data

Big data analytics deals with the difficulties of managing the observed exponentially increasing collected volumes of data. Its purpose is not only to handle the processing and analysis of the data through specialized software tools and frameworks, but also to handle the means on how this enormous amount of data is collected and stored in the first place. It is in its nature that big data is all about massive volumes of information that require specialized hardware infrastructure to store it [7].

Services of enterprise organizations are running on all the collected data which can take various forms such as database entries, metrics, logs or outgoing messages. New data streaming technologies working at a large scale needed to be engineered to handle the continuous flow of data between systems and databases. An example of such technology includes Apache Kafka which generates even more than a trillion of messages per day for individual large enterprise organizations taking advantage of it [28, 42].

This only proves that big data deals not only with massive volumes – it has also to deal with the high velocities of data generation, which is yet another characteristic of data [7]. According to DOMO report published back in 2020, 90% of world's data was generated just in the preceding two years, and on average every person in the world created 1.7 megabytes of data per second – which yields 2.5 quintillion ($2.5 \cdot 10^{18}$) bytes of new data each day [21].

The momentum of the immense big data interest growth among organizations is not fading away yet, as more and more new businesses and researches are drawn to this subject. The benefits of big data especially concern scientific organizations and large enterprises of which financial domain and IT industry are the common

consumers [23]. The organizations find the interest in information analytics for remarkably diversified reasons. It is recognized as a field that will entirely alter all parts of civilization such as businesses or society as a whole [35].

Applications

Various types of organizations collect data to take advantage from the insights derived out of the data. The big data analytics applications' impact can be observed already today in a broad spectrum of domains.

Leading technology companies, such as Google and Facebook, to name a few, sell anonymised collected user data access to their partner advertisers [35]. This is legally possible as the information that was anonymised, i.e., de-identified in a way that it is no longer bound to an individual, may flow from a system to a system.

Large Hadron Collider (LHC) located in CERN, being the largest physical experiment, annually produces approximately 30 petabytes of data. LHC takes an advantage of light sensors that monitor the collisions of hundreds of millions of particles accelerated nearly to the speed of light. The collisions create enormous amount of data to be processed by computer algorithms in the hope of discovering new particles, e.g., a Nobel prize awarding discovery of Higgs boson had taken place in 2012 [1].

Enterprise stores such as Amazon or SAP Commerce Cloud collect the information regarding the way of how the visitors browse and interact with these stores. Collected information may involve the behavioural data related to customer engagement, such as the pages we visit, event clicks, or the way we scroll the page. The insights derived from the collected information enable making future improvements of these services – for example by improving the digital marketing or performance improvements based on the metrics [33]. The customer experience is also improved as based on the collected data the advertisements or item recommendations can be tailored to the specific user's preferences. The recommendation engine may also attempt to match your profile data to people of similar profile to provide better recommendations. Services attempt to analyse the behavioural patterns such as time of day we browse the store or what circumstances caused our last visit to finish. Even the details such as the exact neighbourhood location we

live in, combined with its estimated wealth, organizations may attempt to guess our potential income level [35]. These data analytics are performed to improve the possibility of customer buying yet another item.

2.2 Data privacy

The privacy endangerment is inherent to big data and it is its major drawback. Our personal data we continuously give away to third parties is the big data fuel.

As the technology evolves, concerns relating to the privacy of our personal information should also grow – and for a relevant reason. We tend not to give a second thought to whom the data is shared, how it may be used and in what kind of circumstances. We don't wonder how our own data may be exploited to alter our thinking, decisions, or even ideas – whether in an ethical manner or not.

Although the use of our personal information existed ever since the very first census was created, data privacy is a relatively new concept, as it did not exist prior to the global adoption of the internet. Granted that our data was used by the researchers even before the digitalized era of the internet, the motives for that usage were not commercial [47]. Nowadays, the data has most certainly become an asset – a resource like any others, and a rather precious one.

It is argued that the data privacy should be centered only around the data usages that have the potential to be a privacy breach. On the other hand, it is argued that merely a collection of the data is already a privacy harm. The information that was collected is endangered by many threats, including data misuse, data breach, data leak or even authorities accesses without legal obligations. Anonymisation is the best method to mitigate conflicts raised by big data with respect to data privacy and data protection [51].

Luckily the law had finally caught up to the circumstances of the increasing data usages and the associated risks. New European regulations were adopted in 2018 in the form of General Data Protection Regulation (GDPR) to protect our data privacy in a refined fashion. The data subjects, i.e., the individuals represented by the data [5], now have a better control over their personal information – we are now entitled to know what information concerning us is being processed and for what purpose. We are also entitled to withdraw at any time the consent for

the processing of our data. In case of violations we have the right to complain to authorities and seek justice against both the data controllers, i.e., the entities that determine the intention and means of processing the personal information, and the data processors, i.e., the entities that process the personal information on behalf of the data controllers [53].

An overall awareness of the data privacy significance had improved in the society, and the means to achieve the data privacy through data protection had also improved as organizations needed to adopt to the new situation by applying enhanced measures to their protection of data – anonymisation and pseudonymisation being the notable examples of such measures.

2.2.1 Authorization to share data

Majority of data privacy regulations are based on a consent of an individual, i.e., it is lawful to process and use the information for secondary purposes only if an individual explicitly acknowledge their consent for that [55]. This may appear easier said than done due to the unobvious difficulties data controllers face when trying to obtain such a broad authorization consent that will take into account all possible secondary purpose usages.

Consider a patient entering medical facility for an ordinary appointment. The patient would likely find it unusual, disturbing or even shocking if upon his entrance to the facility he was to receive an overwhelming form that included dozens of independent consent authorization requests. The consents could give the impression of being seemingly unrelated to his visit in the first place, e.g., a constent to share the data with researchers of an university located on another continent. In the end that could destroy the data subject's trust – in this case patient's trust.

This theoretical scenario may not easily be implemented in the real world counterpart, as it could be even impossible to know or predict all possible secondary purpose usages in the first place, and consent based authorization is all about knowing the usages.

Consider a newly discovered purpose to process personal information of an already existing database. Getting a consent after the data had already been collected, i.e., backwards in time, would be impossible to accomplish as the data

controller would need to contact potentially hundreds of thousands of people for their explicit consent. New purpose can be discovered years after the data collection.

Having that in mind, no consent is required when processing the data that is already anonymised. A data that was stripped from personal identifying or identifiable information data can be used in any way and can be shared with third parties without previously agreed consent. Data controllers now face a realistic to solve problem of information anonymisation rather than an unrealistic problem of consents collection [22].

It is worth mentioning that there exists cases which are defined under Article 6 of General Data Protection Regulation (GDPR) [17] when consent is not required to process the data, e.g., if the processing is required to defend the data subject's interests or in order not to break the compliance with legal obligations as a data controller. GDPR is a new European law that replaces the preceeding Data Protection Directive regulation adopted by European Community in 1995 [55].

2.2.2 General Data Protection Regulation

One of the primary objectives of GDPR is personal data privacy protection which is a fundamental right and freedom of people as defined under the Recital 1 of GDPR [13] and the Charter of Fundamental Rights of the European Union [12]. Newly discovered challenges for the protection of personal data arise from the ongoing globalization and quick development of digital technologies. This in turn vastly had increased both the scope of the gathering of the data, and the sharing of thereof. General Data Protection Regulation (GDPR) is a data protection law that came into force on May 25, 2018 to addresses these data privacy related issues in a strict manner[15].

Compliance with GDPR law is critical for organizations in the view of significant administrative fines they face. Violations of the data processor and data controller obligations defined in GPDR are subject to costly penalties that are imposed by European authorities. Non compliance with technical rules imply a penalty of 2 percent of the total annual turnover of the previous year, or €10 milion, whichever one is higher, whereas non compliance with basic data pro-

tection principles imply an even higher penalty of 4 percent of the total annual turnover of the previous year, or €20 million, whichever one is higher [16][38].

Implementation of this law had immediately increased the significance of data anonymisation as an information sanitization process in today's world [43]. Anonymisation being a specific form of data masking suddenly became more relevant in today's world for the reason that the strict regulations, and therefore administrative fines, defined in GDPR do not apply to the anonymised information. Data protection principles covered throughout GDPR concern only the processing of information that is already identified to a natural person, or that is identifiable to a natural person, i.e., an individual is yet to be identified. Given the fact that anonymised information is by definition not relating to a person, hence it can be completely exempted from falling under the GDPR requirements, which apply only to personal data, as stated under Recital 26 [14]:

> The principles of data protection should apply to any information concerning an identified or identifiable natural person. [...] The principles of data protection should therefore not apply to anonymous information, namely information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable. This Regulation does not therefore concern the processing of such anonymous information, including for statistical or research purposes.

GDPR distinguishes personal data, anonymised data and pseudonymised data as distinct variations of data. The information that had gone merely through pseudonymisation process would still fall under the regulations of GDPR, due to the existing relevant possibility of future re-identification of the data subject, whereas in case of the anonymised data, such re-identification is by definition either impossible or extremely impractical, and the anonymisation is irreversible by definition. Anonymised data is completely exempted from being governed by GDPR. Nevertheless, pseudonymisation is still one of many possibilities for the data controllers and data processors to be GDPR compliant [55]. The point of anonymisation in the context of GDPR is to be completely exempted from being governed by this regulation.

2.3 Data anonymisation

The data released by organizations exclude identity related information such as names, addresses, telephone numbers or credit card numbers. Personal data is stripped from disseminated data sets through anonymisation to protect the anonymity of the individuals, i.e., data subjects [48].

2.3.1 Background

Consider a collection of medical data concerning patients' clinical information. Processing and sharing of medical data volumes is crucial for the evolution of world's healthcare services. Medical researchers and doctors take an advantage of the collected data sets to improve their comprehension of diseases and explore new possibilities to treat these diseases, and hence both the overall capability to treat the diseases and the general efficiency of health services are improved. At last it is the patients who benefit from the research conducted on their data since the services they are offered with continuously improve. Nevertheless, it is known that medical data is exceptionally sensitive by its nature due to the details that include e.g., patient data, laboratory tests results, diagnosis details, prescribed medications and history of diseases [30].

Having understood how sensitive by its essence is the patient information and the vital needs to share this data, this is where data anonymisation plays an indeed crucial role. It would be impossible to disseminate patient information without prior anonymization of thereof.

2.3.2 Definition

Anonymisation is a statistical disclosure control method of a particular importance. The ultimate anonymisation goal is to de-identify the data by pruning the personal information in such a way that the relation between data subject and corresponding records is blurred.

The data anonymisation is considered to be an effective one only if both of the following criterias hold true [32]:

- performed anonymisation operations are irreversible

- data subject re-identification is impossible or impractical

2.3.3 Data classification

The prerequisite for anonymisation is to understand the context of what does the data represent and by whom – in addition to how – will it be processed [22].

The commonly adopted approach when attempting to anonymise the data is to first classify the data attributes as direct identifiers (i.e., personal identifying confidential attributes unique to an individual, such as address or tax identification number), indirect identifiers (i.e., so called quasi-identifiers that when combined can discover the individual's identity – such as sex or date of birth) and non identifiers [20, 22].

It is context dependent to determine which attributes are quasi-identifiers – and finally which techniques need to be combined to achieve anonymisation. Virtually, even the most unexpected attributes could be quasi-identifiers. Ignoring such fields when preparing for anonymisation may eventually lead to future re-identification of the data subjects, as successfully demonstrated, e.g., when breaking the anonymity of the Netflix Prize dataset back in 2006 [4]. Since then significant advancements in the field of de-identification were yielded – and in various domains [6].

On the other hand, too much anonymisation of quasi-identifiers strips the data out of utility.

2.3.4 Utility trade-off

The organizations need to compromise between the utility and privacy when releasing the data. As anonymisation level increases when more and more restrictive techniques are applied, the utility of the same data decreases.

Consider the two theoretical boundary cases of either releasing data in its original form which maximizes the utility at the cost of discarding the protection of the data subjects' privacy altogether, and the second case of not releasing any data at all which completely preserves the privacy [58].

Clearly the data must not be released in its original state – strict GDPR that imposes large administrative fines [16] exists to mitigate such violations. On the other hand, data that completely suppresses all the information has no value.

The data controllers therefore need to strike a balance between the two cases. An appropriate anonymisation strategy maximizing the utility without endangering the privacy must be adopted.

2.3.5 Pseudonymisation

What is pseudonymization? How does it differ?

2.3.6 De-identification

Nawet jeśli dane personall identifiable information PII są usunięte, to można połączyć osobe z pomocą istniejących nieusuniętych danych np. płeć oraz external informacjach np. social media. Komputery są coraz szybsze, źródeł danych coraz więcej - napastnikom coraz łatwiej.

Przykłady takich ataków (Netflix Prize, IMDB). https://www.cs.princeton.edu/ arvindn/publ anonymization-retrospective.pdf https://arxiv.org/abs/1512.08546

// todo Conclusion: Anonymisation = false sense of security! All the known examples of this type of identification are from the research world — no commercial or malicious uses have yet come to light — but they prove that anonymization is not an absolute protection.

Why you can't really anonymize the data:

https://web.archive.org/web/20140109052803/http://strata.oreilly.com/2011/05/anonymize data-limits.html

2.4 Data masking techniques

It is not possible to predetermine – in a general fashion and without an extra context – the data masking techniques that need to be combined together in order to achieve an actual anonymisation of the previously identifiable information. Instead, a context consisting of:

- exact representation form of the data being processed *(What is the data?)*
- data processors who use the data *(Who will use the data?)*

- processing purpose, e.g., research objective *(How will the data be used?)*

is always required when considering an effective way to anonymise data under that context [22].

2.4.1 Suppression

Suppression is the strongest anonymisation technique which completely removes all the values associated with the given attribute, hence rendering the complete protection of the data and therefore enforcing the best data subjects privacy. It is guaranteed that no further implication attacks can be performed against this data [48]. Nevertheless, suppression being the strongest privacy preserving method implies the highest (i.e., worst) data utility loss as a inseparable consequence – the ultimate cost of complete anonymisation that is to be paid.

Consider the raw data of the three attributes presented in the table 2.1 to be put under the process of suppression.

Table 2.1: Data prior to suppression.

Sex	PIN codes	Phone number
F	3248	1212 – 345345
F	8090	4000 – 303030
M	1337	5191 – 915100

Exemplary results of performed suppression are depicted in the table 2.2. All of the three attributes values were suppressed to the value of the suppression token supplied by the data controller.

Table 2.2: Data after suppression.

Sex	PIN codes	Phone number
F/M	####	3000 – 123123
F/M	####	3000 – 123123
F/M	####	3000 – 123123

Phone number suppression deserves a brief discussion. Consider a context in which an attacker successfully obtained the information of exactly one individual

data subject of his interest. Consider that the attacker decided to call the anonymised phone number. This number could be suppressed in a way to specifically protect against this attack and the attacker could call someone intentionally most definitely did not, i.e., data protection officer or authorities. In this scenario it could be trivial to trace the call and find the potential attacker. As demonstrated, suppressed could be used in unobvious ways.

Suppression is typically applied on the column level of an attribute, however it is also applicable to the individual records. When compared to plain generalisation, suppression yields a higher information loss and can be though of as a particular case of generalisation [30].

The designed software offers column level suppression with a user specified value (i.e., suppression token). The values are transformed to this user specified value using a simple algorithm shown in Fig. 2.1.

```
1 procedure suppression(values, token)
2   for (value in values)
3     value := token
4   return values
```

Figure 2.1: Suppression pseudocode.

2.4.2 Generalisation

Generalisation is the second most common non-perturbative anonymisation technique [20]. This technique processes the raw data by aggregating and substituting the initial values of a given attribute to the values that are more general [3]. The process of generalisation constitutes a conversion of any value to a more general scope.

The proposed software solution includes two generalisation strategies, namely:

- based on the size of distributions
- based on the number of distributions

Consider the raw data of the three attributes presented in the table 2.3 to be put under the process of generalisation.

Table 2.3: Data prior to generalisation.

Age	Salary	Location
27	36 000	Poland
52	54 000	Canada
30	180 000	Poland
68	128 000	Switzerland

Results shown in the table 2.4 include exemplary possible values after being processed by the generalisation method. As depicted in the table, all three attributes have been grouped into broader value ranges, hence undergoing the generalisation, yet every column was generalised with a different generalisation strategy.

Table 2.4: Generalised data.

Age	Salary	Location
26 – 30	1 – 60000	Europe
51 – 55	1 – 60000	North America
26 – 30	120001 – 180000	Europe
66 – 70	120001 – 180000	Europe

Age column was generalised with a strategy based on the size of distribution. In this case the size of distribution was defined as 5, as every interval aggregates five distinct increasing integer values.

On the other hand, salary data was generalised with a strategy based on the number of distributions – the values were aggregated to three even distributions.

Finally, the generalisation does not necessarily concern only numerical values as observed in the generalisation of location attribute. To better understand this type of generalisation, consider that the age attribute could be generalised into the values of e.g., *young adult, adult, senior*. This type of generalisation is not supported by the developed software, however an illustration of existence of such method is necessary. The fundamental assumption of the developed software is to be generic, i.e., data context-agnostic, and this generalisation method is too specific in its essence to be implemented in a generic fashion.

Both supported generalisation strategies were further enhanced with customizations. The data processor is allowed to predefine minimum and maximum bound-

ary values for the generalised intervals. The computed starting value for the distribution concerning lowest interval is either minimum raw value or user predefined minimum value, whichever is lower. The maximal value for the highest interval is computed in an analogical way.

```
1  procedure generalise(values, configuration)
2    computedMin := min(values.min(), configuration.userMin())
3    computedMax := max(values.max(), configuration.userMax())
4    distributionSize := configuration.distributionSize()
5
6    // Phase 1: Generate empty intervals.
7    intervals := <Interval, Values>
8    i := computedMin
9    while (i < computedMax)
10     interval := asEmptyInterval(i, i + distributionSize)
11     insert interval into intervals
12     i += distributionSize
13
14   // Phase 2: Populate intervals.
15   for (value in values)
16     for (interval in intervals)
17       if (value is within interval)
18         interval := get interval from intervals
19         add value to interval
20   return intervals
```

Figure 2.2: Pseudocode for generalisation based on distribution size.

The generalisation strategies can be summarized with the pseudocodes shown in Fig. 2.2 and Fig. 2.3.

Generalisation may be applied on a global level or local level [48]. In terms of database related vocabulary, as this paper primarily concerns an anonymisation of databases, we say generalisation on a column or record level, respectively.

2.4.3 Perturbation

Perturbation is a data masking technique which encompasses the process of swapping the original values with artificial ones in a way that the statistical factors of the data are similar to the initial data. Perturbation is typically achieved by adding noise to the information so that the values are slightly different [27].

```
procedure generalise(values, configuration)
  computedMin := min(values.min(), configuration.userMin())
  computedMax := max(values.max(), configuration.userMax())
  numberOfDistributions = configuration.numberOfDistributions()

  // Phase 1: Generate empty intervals.
  intervals := <Interval, Values>
  i := computedMin

  distributionSize := (computedMax - computedMin) / numberOfDistributions
  j := 0
  while (j < numberOfDistributions)
    interval := asEmptyInterval(i, i + distributionSize)
    insert interval into intervals
    i += distributionSize
    j += 1

  // Phase 2: Populate intervals.
  for (value in values)
    for (interval in intervals)
      if (value is within interval)
        interval := get interval from intervals
        add value to interval
  return intervals
```

Figure 2.3: Pseudocode for generalisation based on number of distributions.

Consider weight and height data shown in the table 2.5.

Table 2.5: Data prior to perturbation.

Height	Weight
166	58
170	66
194	91

The table 2.6 shows the data after it was perturbed using the two different perturbation methods that are supported by the engineered software.

The supported methods include adding the noise based on the strategies of:

- fixed value
- percentage value

Table 2.6: Perturbed data.

Height	Weight
169	57
168	67
193	91

Height attribute values could have been perturbed using the former (i.e., fixed noise of ±3), whereas weight attribute values could have been perturbed using the latter (i.e., percentage value of ±5%). In either case, the linkage between individual records is blurred with this technique.

```
1 procedure perturbation(values, noise)
2   for (value in values)
3     random := pick random from [value - noise, value + noise] interval
4     value := random
5     value := fit value within boundaries
6   return values
```

Figure 2.4: Pseudocode for perturbation based on fixed noise.

Similarly to generalisation, both perturbation techniques allow the user to explicitly specify the minimum and maximum accepted boundary values.

```
1 procedure perturbation(values, noise)
2   for (value in values)
3     coeff := pick random from [1 - noise, 1 + noise] interval
4     value := value * coeff
5     value := fit value within boundaries
6   return values
```

Figure 2.5: Pseudocode for perturbation based on percentage noise.

Perturbation technique works the best with continuous values - it is an effective method for de-identification of quasi-identifiers such as dates and numbers [49]. Scientifically advanced perturbation approaches are further divided into linear and non-linear perturbation models [46].

2.4.4 Pattern masking

2.4.5 Randomisation

Column shuffle

Row shuffle

2.4.6 Artificial data

Random data

2.4.7 Shortening

2.4.8 Hashing

2.4.9 Encryption

2.4.10 Tokenisation

2.5 Existing solutions

List of existing solutions.

2.6 Platform

Why web platform, state of the art for web.

- definicja anonimizacji
- rodzaje animonimizacji (od razu odnośniki do literatury (bibliografia))
- metody, narzędzia
- problem analysis
- state of the art, problem statement
- literature research (all sources in the thesis have to be referenced [**bib:article**, **bib:book**, **bib:conference**, **bib:internet**])

- description of existing solutions (also scientific ones, if the problem is scientifically researched), algorithms, location of the thesis in the scientific domain

[state of the art dla aplikacji sieciowych]

Chapter 3

Design specification

[Dlaczego aplikacja webowa?]

3.1 Requirements engineering

3.1.1 Functional requirements

Account creation

To create a new account, an anonymous user needs to provide valid information including e-mail address, password and confirmation, name, surname, and optionally the purpose for anonymisation. Creating an account with an e-mail that is already in use must be prohibited. A new account must be verified to access the broad functionalities.

Account verification

Upon successful account creation, the user must receive an e-mail with a link to verify the account. User role should change from *unverified* to *verified* due to confirmation.

Admin verification

Admin has the means to verify the specified user on his behalf from the users panel.

Verification expiration

The verification link that is sent to the user after creating a new account should contain a token that expires after a set amount of time. The system administrator has to have a possibility of runtime configuration of the expiration time.

Mark account for removal

User must be able to initiate the process involving the removal of his account along with all the associated data, hence ensuring GDPR compliance.

The process is started by an explicit user's requests to do so. The system should warn the user with a detailed message and a prompt to type in both the confirmation and the password. If successful, the account is marked for future removal - the account is removed by a dedicated cron-based data pruning service.

Data pruning

A service responsible for deleting accounts that were marked for removal should be executed in a periodic manner. Such service should prune the user data. The frequency of how often is the service executed is described as a cron expression. It should be a system administrator runtime configurable property.

Forcing account removal

Admin must have the possibility to force the account to be removed in case of urgent needs e.g. due to reasons concerning system security or performance.

Revert marking account for removal

A scenario in which user changes his mind to delete his account needs to be supported. The user can revoke his request to remove the account by accessing the link that was earlier sent to his e-mail address.

The revert action should be possible only if all three conditions are met:

- the link has not yet expired
- the admin did not explicitly force the account removal

- the account has not already been removed by the pruning service

Revert account removal expiration

The undo removal link sent to the user after requesting his account removal contains a token that expires after a set amount of time. The time of exactly how long is the link valid should be determined by the runtime configurable by the system administrator.

User login

User should be able to login to the system and access his account's resources.

Forgot password

There should be a way to reset the password that was forgotten. The link should be sent to the user's e-mail address and should be valid only for a configured duration.

Access token

A successful authentication should generate an access token which encompasses all the information necessary for the user authentication and authorization. The token should be valid for a brief duration, e.g., 5 minutes – which should be a matter of runtime configuration conducted by the system administrator. The token should be implemented using JSON Web Token (JWT) standard and should be stored on the client's side.

Refresh Token

The refresh token which is generated together with access token during authentication should be used to periodically generate a new pair of the access and refresh tokens. There should exist a REST endpoint that will accept current valid refresh token and will output a new access and refresh token - effectively prolonging user's session. This token should be valid for a very long duration, e.g., 6 months – which should be a matter of runtime configuration.

User logout

Discarding the access and refresh tokens permits user logout.

Remember me

The user should not have to login to his account when starting a new visit unless

- the refresh token has expired after e.g., 6 months of inactivity
- the user has cleared browser cache
- the user is logging from a new device

Restarting the anonymisation platform should not logout the user.

Seamless tokens re-generation

Axios HTTP client should be configured in such a manner that it will detect a state of expired access token and will seamlessly generate new tokens. When the access token expires, the client should invoke refresh token service to generate a new tokens pairings in such a manner that the user does not even notice this implementation related behaviour. It is required that no unnecessary page refreshes, errors, logouts or prompts to login show up to the user.

Password encryption

User password should be stored in the database in an ecrypted form using e.g., bcrypt hashing.

Request validation – client

The system should disallow sending HTTP requests that fail to meet the validation criterias set with Yup library.

In-depth validation – client

Validation rules set by Yup library for the client forms should be enforced by React Hook Forms.

The client should have the possibility to specify required and optional fields and use custom validation rules such as e-mail or file presence validation.

Display validation errors

Client displays validation errors below the concerning HTML inputs.

Error notifications

Client uses toast pop-ups to display client and server HTTP errors [26] – for example 400 Bad Request. The notifications along with the associated message are displayed in the upper-right corner. The pop-ups are red to underline its purpose and the purpose should be further emphasized - by adding more red color – when displaying the errors of extreme importance e.g. database restore failure.

Success notifications

Client sent HTTP requests that finished with a successful response [26] – for example 201 Accepted - should be displayed in a green coloured pop-up along with the associated message.

Block user

Admin should have the means to block the user from the users panel, effectively disabling the possibility of logging in.

Unblock user

Admin will have the means to unblock the blocked user from the users panel, restoring the login capabilities.

Data masking

There should exist data transforming services that will offer the functionalities of the following techniques:

- suppression
- generalisation
- perturbation
- randomisation
- tokenisation
- random number replacement
- hashing
- column shuffle
- row shuffle
- shortening
- pattern masking

There should a validation standard established that will determine if particular anonymisation techniques are applicable together or applicable to the given data type. The techniques should be configurable.

Request validation – server

The system must validate the HTTP requests on the server to protect the system from passing malformed requests sent by the malicious external HTTP clients. The validations should be declaratively used on the Data Transfer Object (DTO) classes.

Exemplary validation constraints could include checking for values that meet any of the following criterias:

- not null
- not blank
- not empty
- for integers – of a value in an allowed range
- for strings – of size in an allowed range
- non-standard such as e-mail

Maximum file size

System administrator should be in control of configuration of the maximum accepted sizes of the files sent to the system (i.e. database dumps). The file sizes that are too large must not be accepted.

In-depth validation – server

Server should perform numerous other advanced validations specific to individual scenarios.

Exemplary cases related to infrastructure should check if:

- the system is running in cloud environment
- an entity non compliant with the database schema is trying to be persisted
- database connection is established

Exemplary cases related to authorization should check if:

- user is trying to access another user's resource
- user is unauthorized to access the resource due to lack of privileges
- long lasting refresh token has expired

Exemplary cases related to domain should check if:

- anonymisation operation is applicable the particular column

- column is a primary or foreign key
- a given task is manually executable

Client tables settings

The tables presented used extensively by the client application should offer the functionalities of:

- pagination to a set page size
- filtering by attribute
- sorting by attribute
- hiding and showing of columns

Navigation menu

The client application should have a navigation menu. The panel should be located on the left side of the screen. Each navigation item should consist of a large icon and a text label. The entire panel should be hideable – the entire page content should seamlessly adapt to this. Optionally, smooth transition animation should be configurable during TypeScript compilation time.

The items should be easily defined, ordered and configured by adopting a standard. Visibility of a particular item must be role-based, e.g. only admin can see *users* item.

User profile

The user should be able to view and edit his profile details. The user must not be able to change his e-mail address.

Timestamp data collection

The system should collect timestamp data about various resources such as users or templates. Some of the data should be client renderable, and some of the data should be merely collected.

Exemplary dates concern:

- account registration, block, request for removal, removal
- template generation

Requests data collection

All HTTP requests to the anonymisation server should be logged in the system. The logged information should include HTTP method, HTTP response status and the request URI.

File selector

There should be a custom file selector input that will allow uploading files i.e., database dumps. The component should show the upload progress bar and success or error message upon completion.

Uniform design

The user must not be surprised with a particular page looking in an unexpected way. There should be established a uniform design for the client pages. The design should uniformise, e.g., colors, borders, paddings, margins, shadows or components.

Mail notifications

The system has the capabilities of sending e-mails notifying about various actions such as expiring account. This functionality should be runtime configurable by the system administrator.

Role-based authorization

All non-whitelisted REST resources should be protected with declarative role-based authorization.

Resources whitelist

System administrator should be able to configure whitelisted (i.e., unprotected) REST resources. Unprotected endpoints do not require to be authenticated (i.e., logged in) to access them.

Database preloader

There should be a functionality to preloaded the system with prepared data. A new system instance must include at least one administrator account.

Dynamic database connection

There should exist a factory for data sources that will establish a connection to the dynamically created databases restored from dump files uploaded by the users.

Prune connection pool

There should be a functionality to drop all existing database connections, effectively allowing to create the database mirrors dynamically.

Environment based configuration

Different environments, i.e., production, develop and local should use different configurations of the environment and anonymisation server.

Running processes from Java

There should be the possibility of running external processes such as *psql* directly from the Java code.

Asynchronous events

There should be a configuration for handling asynchronous events. *Uploading* and *processing* server modules should be started with an asynchronous non-blocking event.

Scheduling

There should be a functionality for defining schedulable task services. The services should rely on the cron expression based triggers. Individual tasks should be runtime configurable in terms of their:

- cron expression
- whether the task is enabled for automated scheduling
- whether the task is enabled for manual execution
- displayed description in the panel

Tasks panel

There should be a tasks panel for the admin to retrieve and monitor them. The panel should allow a non-blocking on demand execution of the selected individual tasks.

Users panel

There should be a users panel for the admin to present an overview and manage the users' accounts.

Generate template

The system should have the functionality to

Secrets

There should be a functionality prepared to load the secrets (i.e., password for the database, JWT secret key, mail service password) from an external file non-published to the public git repository. Continuous Integration related secrets should be configured directly in GitHub.

SonarQube

Both the client and the server should be scanned by the SonarQube static analysis tool. This software should be integrated into Quality CI workflow.

CodeQL

Application should be scanned by CodeQL for security issues as a part of Security CI workflow.

Containerization

Software components of the system should be containerized. All services should be running as containers when running the system in production environment.

Multiple environments

The designed system has to be runnable in multiple environments, i.e.:

- production
- development
- development with the *server* profile enabled
- locally on the host with a database container
- fully locally on the host

Distinct configuration files must exist for the distinct environments.

Swagger

The REST API of the anonymisation server is to be documented with Swagger UI.

Starting template generation

The user should be able to start the generation of a new template based on the database dump file he provides. The acceptable dump files should include compressed archive or plain script file. The user must successfully upload the dump before starting the template generation.

Template generation progress

The template generation should involve the sequence of three steps: persisting the dump file, restoring a new database from the dump, extracting metadata about the restored database. The user should be able to view the template generation progress.

This should be presented as a progress panel containing a visualization of three steps. Completion of a step either marks it as a success or error. Currently processing step should be shown. Errors informing about e.g. invalid databases must be reported to the user. The generation progress should internally update status of the template and should be performed as a sequence of events.

Restore database

The system should be able to restore a new database from a dump file of either compressed archive or script file.

Mirror database

The system should be able to create a mirror of an existing user database. The mirror should be used for writing the anonymisation script into and then dumping into the final outcome result.

Extract metadata

The system should extract the metadata from the uploaded database when creating a new template. The metadata must include information concerning the structure of the database, e.g., the tables, columns and primary keys the database

contains. Extracted metadata should be persisted in the server database as JSON type.

Inspecting metadata

The user should be able to inspect the extracted metadata.

Download metadata

The user should be able to download the extract metadata.

Viewing templates

The user should be able to view his templates.

Producing worksheet

The user should be able to produce a new worksheet from the templates he created earlier on. Worksheet should be used for building up the anonymisation setup. The worksheet must not be accessible to other users.

Viewing worksheets

There should be a table giving an overview of the created worksheets. Only worksheets belonging to the user should be shown. The table should embed buttons to go to viewing the summary or outcome generation.

Viewing summary

There should be a summary view which is an overview for the given worksheet. It should contain four distinct sections of information of the worksheet: template section, tables section, operations section and outcomes section.

Viewing tables

The tables included in the template should be enlisted in a worksheet summary.

Adding column operations

User must have the possibility of adding new anonymisation operations to the anonymisation concerned columns.

Viewing column operations

The so far accumulated column operations should be enlisted for each individual database table. Different operations should use different colors to provide a better visualization. Apart from accumulated operations, the view should also show information such as column name, type, nullability or whether the column is a primary or foreign key.

PK and FK detection

Primary and foreign keys should be disallowed to undergo anonymisation. An appropriate information to the user should exist to indicate that a given column is primary or foreign key.

Starting outcome generation

The user should be able to start the generation of an outcome based on the worksheet he prepared. It should be possible to select compressed archive or script file as a target output for the outcome. Starting the outcome generation must happen in an asynchronous non-blocking way.

On a high level, the outcome generation should include the complex sequential steps of:

- creating a writeable mirror database of the read-only template database
- creating a new empty anonymisation script
- populating the anonymisation script based on the user defined anonymisation setup
- executing the anonymisation script against the mirror database
- dumping the anonymised database to the compressed archive or script file

There should be an outcome database entity created with status field. Each consecutive step should update this status. The total time it took to process an outcome should be measured.

Browsing outcomes

The user has to have the possibility to browse the outcomes he created in a form of dedicated table. Each outcome should include the status, anonymisation script download button, outcome dump download button. processing time, template name, etc.

Anonymisation script download

There should be a way to download an anonymisation script.

Anonymised dump download

There should be a way to download an anonymised dump result.

File storage

There should be a way to store the files, i.e., user input dumps, anonymisation scripts, anonymised output dumps (archives, scripts). The files should be stored in grouped directories whose path are runtime configurable. The file names should be randomly picked e.g. as a universally unique identifier (uuid).

Prune system

There should be a script pruning the Docker data from the host system.

Theme

The client application should allow an easy way to change the theme colors of the entire application by changing it in one configuration.

Redirection

System should redirect the user to login page when accessing routes that are non existing or unauthorized to them.

Skeletons and spinners

The client should use skeletons and spinners to provide a seamless user experience.

User fetch

Upon successful login the information about principal user should be fetched and stored on the client's side in the user context.

3.1.2 Non-functional requirements

3.2 System engineering

3.2.1 Use cases

3.2.2 REST API

3.3 Software tools and technologies

The following listings are presented to give an overview of the designed system in terms of tools and technologies involved to engineer it without diving into the implementation details. This is a brief summary that merely presents the technological landscape of the designed system – should a particular technology receive a greater detail of explanation, e.g., exemplary usage, it will be described in another dedicated unit.

3.3.1 Tools

This section encompasses a summary of tools needed to develop, build, and maintain the system.

Figure 3.1: Use cases concerning management

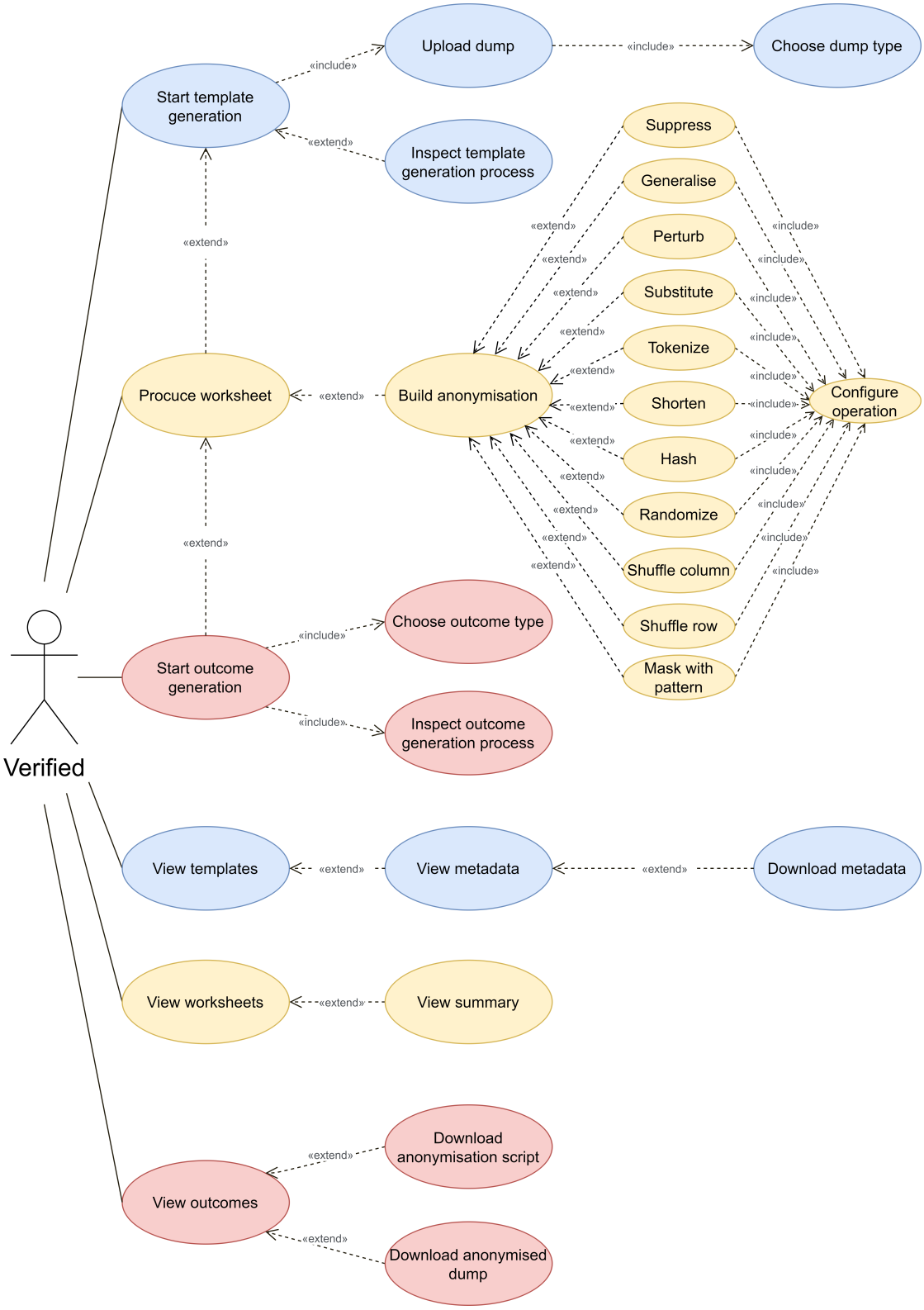

Figure 3.2: Use cases concerning anonymisation

Docker

Docker is a virtualization software used to develop, run and manage applications in the form of containers taking advantage of Linux kernel features namespaces and cgroups. It consists of three components: the runtime, the daemon engine and the orchaestrator.

To run the production environment of anonymisation platform, only prior installation of Docker on the host system is required, because containerization technologies like Docker enable a separation of the software from the infrastructure the software is running on [19][44].

Apache Maven

Apache Maven is an open source that automates the build process of foremostly Java based projects. Primary Maven objectives include ensuring an easy build process and an uniformly established build system which is achieved through Maven's build lifecycle (i.e., predefined build steps, e.g., *validate*, *compile*, *test* or *install*) [29].

Anonymisation platform uses this tool to build the backend server. Maven is put into service in one of the two ways – depending on the selected environment – namely:

- internally from the Docker container where Maven can be considered as an abstraction that the system administrator does not have be aware of
- on the host system itself

Maven is also used to run the tests. Gradle could be a great alternative to Maven, the choice being preference based in usual scenarios.

Node.js

Node.js is a Javascript runtime engine encapsulating V8 engine developed by Google which implements ECMAScript [34]. While node.js may function as a dedicated backend server, in the case of anonymisation platform it is used merely to build the client application with yarn.

yarn

Yet another resource negotiator (yarn for short) is a package manager developed by Facebook, Google and others [41], whose primary purpose is – similarly to *npm* – installation and management of dependencies used by Node.js runtime environment based projects. Yarn also focuses on ensuring high level of security, performance and reliability [25].

While the purpose of Maven is to build and run the backend server, yarn's purpose is to build and run the frontend client application. Consequently, yarn is used in the same way as Maven – either internally in Docker or on the host system – depending on the chosen environment.

nginx

Nginx is an open-source that works as a reverse proxy, web server and HTTP load balancer with the first one being the primary function for anonymisation platform. Reverse proxies improve resiliency, security, scalability and performance of the software systems. This is particularly relevant when designing systems that scale horizontally [18].

git

It's been a while since most developers had already started using git for development, which is the leading software for version control, however not everyone knows it was originally invented by Linus Torvalds specifically for Linux kernel development needs [54]. It is clearly necessary to use git during software development due to the essential needs to publish, stash or revert the code changes [9].

GitHub Actions

GitHub Actions is a continuous integration (CI) and continuous delivery (CD) tool that enable creating workflows which automate development processes such as building, testing and deployment of the system.

Developed software uses three distinct workflows to ensure an automation of testing, quality assurance and security resilience.

SonarQube

SonarQube is a large scale open-source tool to conduct the code inspection through static analysis methods. This tool platform can reliably detect vulnerabilities, security hotspots, code smells and bugs.

In the context of designed anonymisation platform, this software is integrated together with Github Actions to ensure quality and security of developed software. Two distinct instances of SonarQube are configured, independently for the client and the server.

ESLint

ESLint is yet another tool for static code analysis for JavaScript applications.

It had been necessary to enhance this tool with TypeScript extension in order to successfully integrate it with client application and Github Actions workflow concerning quality.

Prettier

For an unified code formatting standard to be established for the client's application, prettier tool which is a modern code formatter was configured within JetBrains Webstorm IDE.

3.3.2 Server

This section encompasses a summary of technologies used for the implementation of modular and REST driven server.

Java

Anonymisation server uses the newest long-term support JDK 17 version released in September 2021, although the server is backwards compatible to JDK 11.

Java is still globally the most widely adopted development language [8], even after the 25 years that had passed after its first release – it is a truly mature language, accompanied by all types of frameworks and libraries that one can imagine.

Spring Framework

Spring projects can be viewed together as a modular toolkit that addresses the concerns of modern software development [56].

Spring driven software system may include the following Spring projects [24]:

- Spring Framework - the core supporting e.g., dependency injection and web applications
- Spring Boot – to achieve an easier setup of the Spring driven system
- Spring Data – to retrieve information from databases in a consistent, database-agnostic way
- Spring Security – to secure the application with authentication and authorization

Java based systems typically employ Spring technologies, and so does the anonymisation server, which extensively uses all of the listed above projects.

Hibernate

Hibernate – a JPA implementation – which is a well tested and high performant [37] object-relational mapper was a natural choice for the engineered system given the fact that Spring has a built-in support for this library. It is used to map the database entities to the corresponding database tables [45].

Hibernate functionality was additionally enhanced with Hibernate Types library [36] to provide the support for JSON column type – which is not supported by the plain Hibernate but is greatly supported by the underlying PostgreSQL database [31].

PostgreSQL – Server

The choice of appropriate database was of particular importance for due to domain of the designed system.

PostgreSQL was fourth most popular database management system worldwide back in 2017 as shown in the annual Stack Overflow Developer Survey report

[39]. Each consequtive report demonstrated further growth of PostgreSQL. As PostgreSQL is at continuous growth, MySQL (i.e., the most popular database) is at continuous decline. Latest report published in 2021 [40] placed PostgreSQL on the second place and it is believed that it will at last beat MySQL this year.

For this reason PostgreSQL was chosen as the database for the server system. This database was also chosen as the supported database type for the data dumps uploaded by data controllers for the purpose of anonymisation.

PostgreSQL – Client

PostgreSQL client is installed in the containerized anonymisation server to execute *psql* processes throughout the application's runtime, hence providing the communication capabilities with the potentially remote PostgreSQL server.

The client is used by the ZT Processor Executor library – directly from the Java code.

ZT Processor Executor

ZT Process Executor provides capabilities to run external processes directly from the Java code.

This library found many domain-critical applications related to PostgreSQL resources management. The library executes PostgreSQL client processes including:

- creation of a new database
- replication of an existing databases
- database dump restoration
- anonymisation script execution
- database dump to script or to archive

JUnit 5

JUnit 5 is the new and refined version of the worldwide recognized Java testing framework. It is used to drive the unit and integration tests.

Testcontainers

Testcontainers is a cutting edge Java library to support instantiation of lightweight Docker containers for the tests needs. Effectively – the tests run in a containerized environment just like the real application hence blurring the technical differences between real and test environments [2].

This state-of-the-art library is used by the integration tests to bootstrap the PostgreSQL database for the tests.

REST Assured

REST Assured is an integration tests library and a HTTP client which allows sending true HTTP requests to the test server (bootstrapped by Testcontainers library). This library also offers the means to validate received HTTP responses.

The library is used by the integration tests.

Swagger UI

One of many Swagger modules is Swagger UI which deals with the design, description and documentation of REST APIs. The library allows visualization and interaction with the API [50].

This library is relevant due to the complexity of the designed system, as it provides the API documentation for the client.

JJWT

JSON Web Tokens are required to transport the implemented access and refresh tokens necessary for authentication and authorization purposes. This popular library deals with the creation and validation of the tokens.

Bouncy Castle

One of many cryptographic APIs offered by Bouncy Castle library is SHA 3 which internally implements Keccak-f[1600] algorithm.

This library is used by the hashing data masking technique.

Apache Commons Lang

The sole purpose of this library is a generation of random alphabetic and alphanumeric characters for pattern masking anonymisation technique.

3.3.3 Client

This section encompasses a summary of technologies used for the implementation of React based client.

TypeScript

TypeScript is a transcompiled superset of JavaScript that the developers want to work with the most unless already doing so [40]. The language was designed by Microsoft to extend JavaScript with types, interfaces and generics.

The language is powering the frontend client application and React.

React.js

React.js was the most popular web framework as of 2021 [40] as it has finally surpassed the obsolete jQuery library. This is an open source library developed by Facebook to build user interfaces for the web. As a high level overview – React centers around the concept of virtual DOM [57].

React ecosystem

Independent authors create many external libraries to drive the React with extended functionalities. React ecosystem is rich in various libraries which typically include *React* word in their names.

Anonymisation client code extensively uses the following popular libraries:

- React Router – to handle the URL related concepts including routing, navigation and query parameters
- React Query – as an abstraction and improvement layer for fetching the data
- React Hook Form – to provide user input validation in the forms based on Yup defined rules

Yup

Yup is a validation schema builder. In the case of engineered software, the schemas created by Yup are consumed by React Hook Form library to validate the forms.

Material UI

Material UI is yet another React library and a large one. This library is responsible for the appearence of the application.

Axios

Axios is an asynchronous HTTP client used for sending the requests to the anonymisation server.

3.4 Development methodology

- functional and nonfunctional requirements
- use cases (UML diagrams)
- description of tools
- methodology of design and implementation

Chapter 4

External specification

- hardware and software requirements
- installation procedure
- activation procedure
- types of users
- user manual
- system administration
- security issues
- example of usage
- working scenarios (with screenshots or output files)

4.1 Environments

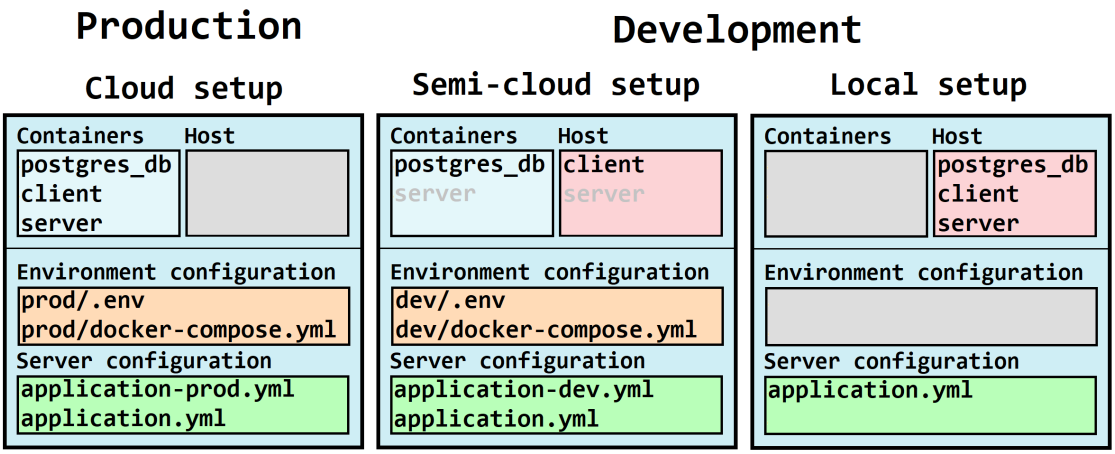

Figure 4.1: Supported environments

Figure 4.2: Figure caption (below the figure).

Chapter 5

Internal specification

- concept of the system
- system architecture
- description of data structures (and data bases)
- components, modules, libraries, resume of important classes (if used)
- resume of important algorithms (if used)
- details of implementation of selected parts
- applied design patterns
- UML diagrams

Use special environment for inline code, eg **descriptor** or **descriptor_gaussian**. Longer parts of code put in the figure environment, eg. code in Fig. 5.1. Very long listings–move to an appendix.

```
1  class descriptor_gaussian : virtual public descriptor
2  {
3      protected:
4          /** core of the gaussian fuzzy set */
5          double _mean;
6          /** fuzzyfication of the gaussian fuzzy set */
7          double _stddev;
8
9      public:
10         /** @param mean core of the set
11             @param stddev standard deviation */
12         descriptor_gaussian (double mean, double stddev);
13         descriptor_gaussian (const descriptor_gaussian & w
               );
14         virtual ~descriptor_gaussian();
15         virtual descriptor * clone () const;
16
17         /** The method elaborates membership to the
               gaussian fuzzy set. */
18         virtual double getMembership (double x) const;
19
20 };
```

Figure 5.1: The **descriptor_gaussian** class.

Chapter 6

Verification and validation

- testing paradigm (eg V model)
- test cases, testing scope (full / partial)
- detected and fixed bugs
- results of experiments (optional)

Chapter 7

Conclusions

```
1  public class ColumnShuffleFacade extends AnonymisationFacade {
2
3    AnonymisationService<ColumnShuffle> anonymisationService = new
     ↪  ColumnShuffleService();
4
5    @Override
6    List<Pair<String, String>> getAnonymisedRows(ColumnOperations oper,
     ↪  List<Pair<String, String>> rows) {
7      ColumnShuffle columnShuffle = oper.getColumnShuffle();
8      if (columnShuffle == null) {
9        return null;
10     }
11     return anonymisationService.anonymise(rows, columnShuffle);
12   }
13
14   @Override
15   protected AnonymisationService<ColumnShuffle> getAnonymisationService() {
16     return anonymisationService;
17   }
18 }
```

Figure 7.1: Java code gets easily verbose.

- achieved results with regard to objectives of the thesis and requirements
- path of further development (eg functional extension . . .)
- encountered difficulties and problems

Table 7.1: A caption of a table is **above** it.

	method						
			alg. 3			alg. 4, $\gamma = 2$	
ζ	alg. 1	alg. 2	$\alpha = 1.5$	$\alpha = 2$	$\alpha = 3$	$\beta = 0.1$	$\beta = -0.1$
0	8.3250	1.45305	7.5791	14.8517	20.0028	1.16396	1.1365
5	0.6111	2.27126	6.9952	13.8560	18.6064	1.18659	1.1630
10	11.6126	2.69218	6.2520	12.5202	16.8278	1.23180	1.2045
15	0.5665	2.95046	5.7753	11.4588	15.4837	1.25131	1.2614
20	15.8728	3.07225	5.3071	10.3935	13.8738	1.25307	1.2217
25	0.9791	3.19034	5.4575	9.9533	13.0721	1.27104	1.2640
30	2.0228	3.27474	5.7461	9.7164	12.2637	1.33404	1.3209
35	13.4210	3.36086	6.6735	10.0442	12.0270	1.35385	1.3059
40	13.2226	3.36420	7.7248	10.4495	12.0379	1.34919	1.2768
45	12.8445	3.47436	8.5539	10.8552	12.2773	1.42303	1.4362
50	12.9245	3.58228	9.2702	11.2183	12.3990	1.40922	1.3724

Bibliography

[1] H. Abramowicz et al. 'Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC'. In: *Physics Letter B* 716.1 (2012), pp. 1–29.

[2] Richard North et al. *Testcontainers*. URL: `https://www.testcontainers.org` (visited on 21/01/2022).

[3] Jens Albrecht, Marc Fiedler and Tim Kiefer. 'A Rule-Based Approach to Local Anonymization for Exclusivity Handling in Statistical Databases'. In: *Privacy in Statistical Databases: UNESCO Chair in Data Privacy International Conference*. Ed. by Josep Domingo-Ferrer and Mirjana Pejić-Bach. Springer Publishing, 2016, pp. 81–93. ISBN: 978-3319453804.

[4] Jens Albrecht, Marc Fiedler and Tim Kiefer. 'Robust De-anonymization of Large Sparse Datasets'. In: *Privacy in Statistical Databases: UNESCO Chair in Data Privacy International Conference*. Ed. by Josep Domingo-Ferrer and Mirjana Pejić-Bach. Springer Publishing, 2016, pp. 81–93. ISBN: 978-3319453804.

[5] Luk Arbuckle and Khaled El Emam. *Building an Anonymization Pipeline: Creating Safe Data*. O'Reilly Media, Inc, 2020. ISBN: 978-1492053439.

[6] Narayanan Arvind and Shmatikov Vitaly. *Robust de-anonymization of large sparse datasets: a decade later*. URL: `https://www.cs.princeton.edu/~arvindn/publications/de-anonymization-retrospective.pdf` (visited on 21/01/2022).

[7] Arshdeep Bahga and Vijay Madisetti. *Big Data Science and Analytics: A Hands-On Approach*. Vpt, 2016. ISBN: 978-0996025539.

[8] Damien Carey. *Celebrating 25 Years of Java*. URL: https://blogs.oracle.com/oracleuniversity/post/celebrating-25-years-of-java (visited on 21/01/2022).

[9] Scott Chacon and Ben Straub. *Pro Git*. Apress, 2014. ISBN: 978-1484200773.

[10] Inc. Cisco Systems. *Cisco Annual Internet Report (2018–2023)*. Tech. rep. URL: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf (visited on 21/01/2022).

[11] Inc. Cisco Systems. *The Zettabyte Era Officially Begins*. URL: https://blogs.cisco.com/sp/the-zettabyte-era-officially-begins-how-much-is-that (visited on 21/01/2022).

[12] European Comission. *Charter of Fundamental Rights of the European Union*. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:12012P/TXT&from=EN (visited on 21/01/2022).

[13] European Comission. *General Data Protection Regulation - Recital 1*. URL: https://gdpr-info.eu/recitals/no-1 (visited on 21/01/2022).

[14] European Comission. *General Data Protection Regulation - Recital 26*. URL: https://www.privacy-regulation.eu/en/recital-26-GDPR.htm (visited on 21/01/2022).

[15] European Comission. *General Data Protection Regulation - Recital 6*. URL: https://gdpr-info.eu/recitals/no-6 (visited on 21/01/2022).

[16] European Comission. *General Data Protection Regulation – Art. 83 – General conditions for imposing administrative fines*. URL: https://gdpr-info.eu/art-83-gdpr/ (visited on 21/01/2022).

[17] European Comission. *General Data Protection Regulation – Art. 83 – Lawfulness of processing*. URL: https://gdpr-info.eu/art-6-gdpr/ (visited on 21/01/2022).

[18] Derek DeJonghe. *NGINX Cookbook: Advanced Recipes for High-Performance Load Balancing*. O'Reilly Media, Inc., 2020. ISBN: 978-1492078487.

[19] Inc. Docker. *Docker overview*. URL: `https://docs.docker.com/get-started/overview` (visited on 21/01/2022).

[20] Josep Domingo-Ferrer and Jordi Soria-Comas. 'Anonymization in the Time of Big Data'. In: *Privacy in Statistical Databases: UNESCO Chair in Data Privacy International Conference*. Ed. by Josep Domingo-Ferrer and Mirjana Pejić-Bach. Springer Publishing, 2016, pp. 57–68. ISBN: 978-3319453804.

[21] Inc. Domo. *Data Never Sleeps 6.0*. Tech. rep. URL: `https://www.domo.com/assets/downloads/18_domo_data-never-sleeps-6+verticals.pdf` (visited on 21/01/2022).

[22] Khaled El Emam and Luk Arbuckle. *Anonymizing Health Data: Case Studies and Methods to Get You Started*. O'Reilly Media, Inc, 2020. ISBN: 978-1449363079.

[23] Can Eyupoglu et al. 'An Efficient Big Data Anonymization Algorithm Based on Chaos and Perturbation Techniques'. In: *Entropy* 20.5 (2018), p. 373.

[24] Facebook. *Spring – Projects*. URL: `https://spring.io/projects` (visited on 21/01/2022).

[25] Facebook. *Yarn – Introduction*. URL: `https://yarnpkg.com/getting-started` (visited on 21/01/2022).

[26] Roy T. Fielding and Julian F. Reschke. *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. URL: `https://datatracker.ietf.org/doc/html/rfc7231` (visited on 21/01/2022).

[27] Carl Folkesson. 'Anonymization of directory-structured sensitive data'. MA thesis. Linköping University, 2019.

[28] Apache Software Foundation. *Apache Kafka – uses*. URL: `https://kafka.apache.org/uses` (visited on 21/01/2022).

[29] Apache Software Foundation. *Maven – Introduction*. URL: `https://maven.apache.org/what-is-maven.html` (visited on 21/01/2022).

[30] Aris Gkoulalas-Divanis and Grigorios Loukides. *Anonymization of Electronic Medical Records to Support Clinical Analysis*. Springer Publishing, 2012. ISBN: 978-1461456698.

[31] PostgreSQL Global Development Group. *PostgreSQL Documentation – JSON types*. URL: https://www.postgresql.org/docs/14/datatype-json.html (visited on 21/01/2022).

[32] Richie Koch. *Data anonymization and GDPR compliance: the case of Taxa 4×35*. URL: https://gdpr.eu/data-anonymization-taxa-4x35 (visited on 21/01/2022).

[33] Chris Lee. *Integrating Web Analytics with your SAP Commerce Cloud Storefront - An Overview*. URL: https://www.sap.com/cxworks/article/2589633935/integrating_web_analytics_with_your_sap_commerce_cloud_storefront_an_overview (visited on 21/01/2022).

[34] Google LLC. *V8 JavaScript Engine*. URL: https://v8.dev/docs (visited on 21/01/2022).

[35] Bernard Marr. *Big Data in Practice: How 45 Successful Companies Used Big Data Analytics to Deliver Extraordinary Results*. John Wiley and Sons Ltd, 2016. ISBN: 978-1119231387.

[36] Vlad Mihalcea. *Hibernate Types – GitHub repository*. URL: https://github.com/vladmihalcea/hibernate-types (visited on 21/01/2022).

[37] Vlad Mihalcea. *High-Performance Java Persistence*. Independently published, 2016. ISBN: 978-9730228236.

[38] Andrew Moore. *The GDPR and Managing Data Risk*. John Wiley and Sons, Ltd., 2018. ISBN: 978-1119487746.

[39] Prosus N.V. *Stack Overflow Developer Survey 2017*. Tech. rep. URL: https://insights.stackoverflow.com/survey/2017 (visited on 21/01/2022).

[40] Prosus N.V. *Stack Overflow Developer Survey 2021*. Tech. rep. URL: https://insights.stackoverflow.com/survey/2021 (visited on 21/01/2022).

[41] Christoph Nakazawa, Sebastian McKenzie and Jamie Kyle. *Yarn: A new package manager for JavaScript*. URL: https://engineering.fb.com/2016/10/11/web/yarn-a-new-package-manager-for-javascript (visited on 21/01/2022).

[42] Neha Narkhede, Gwen Shapira and Todd Palino. *Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale*. O'Reilly Media, Inc., 2017. ISBN: 978-1491936160.

[43] Petter Ødegård. 'Data Anonymization for Research'. MA thesis. Norwegian University of Science and Technology, 2019.

[44] Nigel Poulton. *Docker Deep Dive: Zero to Docker in a single book*. Independently published, 2016. ISBN: 978-1521822807.

[45] Inc. Red Hat. *Hibernate Documentation – ORM*. URL: `https://hibernate.org/orm` (visited on 21/01/2022).

[46] Rathindra Sarathy and Krish Muralidhar. 'Perturbation Methods for Protecting Numerical Data: Evolution and Evaluation'. In: ed. by Ranajit Chakraborty, C.R. Rao and Pranab Sen. Vol. 28. Handbook of Statistics. Elsevier, 2012, pp. 513–531. DOI: `https://doi.org/10.1016/B978-0-44-451875-0.00019-1`. URL: `https://www.sciencedirect.com/science/article/pii/B9780444518750000191`.

[47] Sanjay Sharma. *Data Privacy and GDPR Handbook*. John Wiley and Sons Ltd, 2019. ISBN: 978-1119594246.

[48] M. S. Simi, Nayaki K. Sankara and Elayidom M. Sudheep. 'An Extensive Study on Data Anonymization Algorithms Based on K-Anonymity'. In: *IOP Conference Series: Materials Science and Engineering* 225.1 (2017).

[49] Personal Data Protection Commission of Singapore. *Guide to Basic Data Anonymisation Techniques*. 2018. URL: `https://www.pdpc.gov.sg/-/media/Files/PDPC/PDF-Files/Other-Guides/Guide-to-Anonymisation_v1-(250118).pd` (visited on 21/01/2022).

[50] SmartBear Software. *REST API Documentation Tool – Swagger UI*. URL: `https://swagger.io/tools/swagger-ui` (visited on 21/01/2022).

[51] Jordi Soria-Comas and Josep Domingo-Ferrer. 'Big Data Privacy: Challenges to Privacy Principles and Model'. In: *Data Science and Engineering* 1.1 (2015), pp. 21–28.

[52] Colin Tankard. *Big data security*. URL: https://www.sciencedirect.com/science/article/abs/pii/S1353485812700636 (visited on 21/01/2022).

[53] IT Governance Privacy Team. *EU General Data Protection Regulation (GDPR): An Implementation and Compliance Guide*. IT Governance Publishing Ltd, 2020. ISBN: 978-1787782495.

[54] Linus Torvalds. *Tech Talk: Linus Torvalds on git*. 2007. URL: https://www.youtube.com/watch?v=4XpnKHJAok8 (visited on 21/01/2022).

[55] Paul Voigt and Axel von dem Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing, 2017. ISBN: 978-3319579580.

[56] Craig Walls. *Spring in Action*. Manning, 2018. ISBN: 978-1617294945.

[57] Frank Zametti. *Modern Full-Stack Development: Using TypeScript, React, Node.js, Webpack, and Docker*. Apress, 2020. ISBN: 978-1484257371.

[58] Sherali Zeadally and Mohamad Badra. *Privacy in a Digital, Networked World: Technologies, Implications and Solutions*. Springer Publishing, 2015. ISBN: 978-3319084718.

Appendices

Index of abbreviations and symbols

DNA	deoxyribonucleic acid
MVC	model–view–controller
N	cardinality of data set
μ	membership function of a fuzzy set
$\mathbb{E}$	set of edges of a graph
$\mathcal{L}$	Laplace transformation

Listings

(Put long listings in the appendix.)

```
1  partition fcm_possibilistic::doPartition
2                                  (const dataset & ds)
3  {
4     try
5     {
6        if (_nClusters < 1)
7           throw std::string ("unknown number of clusters"
               );
8        if (_nIterations < 1 and _epsilon < 0)
9           throw std::string ("You should set a maximal
               number of iteration or minimal difference --
               epsilon.");
10       if (_nIterations > 0 and _epsilon > 0)
11          throw std::string ("Both number of iterations
               and minimal epsilon set -- you should set
               either number of iterations or minimal
               epsilon.");
12
13       auto mX = ds.getMatrix();
14       std::size_t nAttr = ds.getNumberOfAttributes();
15       std::size_t nX    = ds.getNumberOfData();
16       std::vector<std::vector<double>> mV;
17       mU = std::vector<std::vector<double>> (_nClusters)
```

```
        ;
18      for (auto & u : mU)
19          u = std::vector<double> (nX);
20      randomise(mU);
21      normaliseByColumns(mU);
22      calculateEtas(_nClusters, nX, ds);
23      if (_nIterations > 0)
24      {
25          for (int iter = 0; iter < _nIterations; iter++)
26          {
27              mV = calculateClusterCentres(mU, mX);
28              mU = modifyPartitionMatrix (mV, mX);
29          }
30      }
31      else if (_epsilon > 0)
32      {
33          double frob;
34          do
35          {
36              mV = calculateClusterCentres(mU, mX);
37              auto mUnew = modifyPartitionMatrix (mV, mX);
38
39              frob = Frobenius_norm_of_difference (mU,
                    mUnew);
40              mU = mUnew;
41          } while (frob > _epsilon);
42      }
43      mV = calculateClusterCentres(mU, mX);
44      std::vector<std::vector<double>> mS =
            calculateClusterFuzzification(mU, mV, mX);
45
46      partition part;
47      for (int c = 0; c < _nClusters; c++)
```

```
48        {
49            cluster cl;
50            for (std::size_t a = 0; a < nAttr; a++)
51            {
52                descriptor_gaussian d (mV[c][a], mS[c][a]);
53                cl.addDescriptor(d);
54            }
55            part.addCluster(cl);
56        }
57        return part;
58    }
59    catch (my_exception & ex)
60    {
61        throw my_exception (__FILE__, __FUNCTION__,
              __LINE__, ex.what());
62    }
63    catch (std::exception & ex)
64    {
65        throw my_exceptionn (__FILE__, __FUNCTION__,
              __LINE__, ex.what());
66    }
67    catch (std::string & ex)
68    {
69        throw my_exception (__FILE__, __FUNCTION__,
              __LINE__, ex);
70    }
71    catch (...)
72    {
73        throw my_exception (__FILE__, __FUNCTION__,
              __LINE__, "unknown␣expection");
74    }
75 }
```

List of additional files in electronic submission (if applicable)

Additional files uploaded to the system include:

- source code of the application,
- test data,
- a video file showing how software or hardware developed for thesis is used,
- etc.

List of Figures

List of Tables