
# “Building the Next Generation Personal Data Platforms”

# G.A. n. 871370

**DELIVERABLE 2.1**

**Design of tools to improve user’s privacy**

**H2020-EU-2.1.1: PIMCity**

**Project No. 871370**

**Start date of project: 01-12-2019**

**Duration: 33 months**

Deliverable delivery: 07/12/2020

Deliverable due date: 30/11/2020

## Document Information

**Document Name: Design of tools to improve user's privacy**

**WP2 Title: Tools to improve data subjects' privacy**

**Task 2.1**

**Revision:** 02

**Revision Date: 25/10/2021**

**Author:** Martino Trevisan

## Dissemination Level

| Project co-funded by the EC within the H2020 Programme | | |
|---|---|---|
| PU | Public | X |
| PP | Restricted to other programme participants (including the Commission Services) | |
| RE | Restricted to a group specified by the consortium (including the Commission Services) | |
| CO | Confidential, only for members of the consortium (including the Commission Services) | |

(Tick the corresponding dissemination level of the deliverable according to Annex I).

## Approvals

| | Name | Entity | Date |
|---|---|---|---|
| **WP Leader** | Martino Trevisan | POLITO | 26/11/2020 |
| **Author** | Luca Vassio | POLITO | 26/11/2020 |
| **Author** | Nikhil Jha | POLITO | 26/11/2020 |
| **Author** | Federico Torta | POLITO | 26/11/2020 |
| **Author** | Stefano Traverso | ERMES | 26/11/2020 |
| **Author** | Davide Pozza | ERMES | 26/11/2020 |
| **Author** | Rodrigo Irarrazaval | GRANDATA | 26/11/2020 |
| **Author** | Daniel Fernandez | GRANDATA | 26/11/2020 |

| | | | |
|---|---|---|---|
| **Author** | Javier Calvo | GRANDATA | 26/11/2020 |
| **Author** | Roberto Gonzalez | NEC | 26/11/2020 |
| **Author** | Daniel Oñoro | NEC | 26/11/2020 |
| **Author** | Bhushan Kotnis | NEC | 26/11/2020 |
| **Reviewer** | Roberto Gonzalez | NEC | 26/11/2020 |
| **Coordinator** | Marco Mellia | POLITO | 26/11/2020 |

### Document history

| Revision | Date | Modification |
|---|---|---|
| **Version 1** | 10-15-2020 | V1 |
| **Version 2** | 22-10-2021 | Addressed reviewer suggestions. In details:<br>1. Checked the usage of PII for consistency, referring to "Personal Details" when needed.<br>2. We modified the Privacy Metrics section to specify that the description of data processing purpose must be meaningful and sufficient to base consent on |

## List of abbreviations and acronyms

| Abbreviation | Meaning |
|---|---|
| **PIMS** | Personal Information Management System |
| **PDK** | PIMS Development Kit |
| **TT** | Transparency Tags |
| **P-DS** | Personal-Data Safe |
| **P-PM** | Personal-Privacy Metrics |
| **P-CM** | Personal-Consent Manager |
| **P-PPA** | Personal-Privacy Preserving Analytics |

## Executive Summary

The Deliverable 2.1 describes the first complete design of tools devoted to improving the users' privacy. This document is part of the PIMCity Project, funded from the Horizon 2020 Programme (ICT-13-2018-2019) under Gran Agreement number 871370.

In this document, we illustrate the design of four tools which are developed in the context of the Work Package 2 of the PIMCity Project. These are the Personal-Consent Manager (P-CM), the Privacy-Preserving Metrics (P-PM), the Personal-Privacy Preserving Analytics (P-PPA) and the Personal-Data Safe (P-DS). The design choices and technical solutions have been discussed among the Work Package 2 members and during the PIMCity plenary meetings. Then, the partners directly involved in the development of the tools have taken care of the complete design, which is detailed in this document.

This document describes the following PDK modules:

1. Personal Consent Manager (P-CM)
2. Personal Privacy Metrics (P-PM):
3. Personal Privacy Preserving Analytics (P-PPA):
4. Personal Data Safe (P-DS):

An overview on the PIMCity PDK can be found in deliverable D1.1, while more details on the specific modules is available in deliverables D3.2 and 4.1. The final design and initial implementation of the modules here described will be released in deliverable D2.2.

## Index

# 1.- Introduction

Thanks to the introduction of regulatory frameworks focused on user's privacy such as EU's General Data Protection Regulation (GDPR) and California Consumer Privacy Act (CCPA), we are testifying the diffusion of new systems whose purpose is to help users in storing, understanding, and, possibly, monetizing their personal data in a transparent and easy way.

In this context, WP2 aims at building a set of software modules with the goal of enhancing the users' privacy. To this end, we design components to allow users store their data in a secure way. Second, users must be provided with tools to let them control and manage consent in a transparent way, i.e., decide who, how and when can access data. Third, we aim at creating tools for allowing queries on data that respect the users' consent choices and allow privacy-preserving data analyses. Finally, it is fundamental to present detailed information about the services willing to access data (i.e., data buyers) and the purposes of their business.

In the Work Package 2 of the PIMCity project, we target the goals described above and aim at designing and implementing modules that accomplish them. In this deliverable, we describe the high-level design, describe how the modules satisfy the requirements defined in Deliverable 1.1 and motivate the technical choices behind four independent modules, that, together, build PIMCity's sub-system for enhancing data transparency, query anonymization and users' awareness and control. As such, we defer the reader to Deliverable 1.1 for the full requirements for all modules. This deliverable *does not* cover module integration and the overall system design, that will be covered in the forthcoming deliverable. The aim of the WP is to investigate tools to improve end users' privacy, focusing on the following aspects:

- Design and develop a system able to empower the users to control their consent settings in multiple account and services. It should have an easy-to-use interface and provide auto configuration options to make it easier for the users to configure complex scenarios by using aggregated/crowdsourced data of the different users to build a set of common profiles.
- Design privacy metrics to unveil and communicate end users the data collected by online services, automatically identify and pinpoint possible privacy violations in data collection, communicate these findings to the end users in an easy and intuitive user interface.
- Develop a set of general-purpose building blocks to analyze users' data without affecting their privacy. It will offer some algorithms and methodologies able to provide a certain level of anonymity using concepts as or k-anonymity and differential privacy.

# 2.- Deliverable Objective

The objective of **Deliverable 2.1** is to provide the first complete design of the main components devoted to improving the users' privacy. These tools are part of the **PIMCity PIMS Development Kit (PDK),** basic and generic components that offer fundamental functionalities for PIMS. These modules aim at empowering the users to control how their data are stored, processed, shared, and for which purposes. In this deliverable, we describe the overall design of these modules, providing the overall functioning architecture, motivating the technical choices and describing how they are accessed by users or other system modules (i.e., the interfaces). Here, we do not describe in detail how these modules integrate together or the overall PIMCity ecosystem, being these topics out of the scope of the deliverable.

The Work Package 2 designs four PDK modules, which take care of implementing those functionalities centered on users in terms of data storage and sharing, consent handling and privacy risks presentation. The modules described in this deliverable are:

1. **Personal Consent Manager (P-CM):** The consent manager is the means to define all the user's privacy preferences. It defines which data a service is allowed to collect, process, or which can be shared with third parties by managing explicit consent. Users' settings are imposed on all participating systems. The P-CM s described in Section 3.
2. **Personal Privacy Metrics (P-PM):** They have the goal of increasing the user's awareness. They collect, compute and share easy to understand novel privacy metrics, indicating e.g., which information the system is collecting, how it stores and manages the data, if it shares it with third parties. This module is described in Section 4.
3. **Personal Privacy Preserving Analytics (P-PPA):** This module has the goal of allowing data analysts and stakeholders to retrieve useful information from the data, while preserving the privacy of the users whose data are in the studied datasets. It leverages concepts like Differential Privacy and K-Anonymity so that data can be exchanged among different systems while preserving the actual information as private. It is described in Section 5.
4. **Personal Data Safe (P-DS):** It is the means to store personal data in a controlled form. It implements a secure repository for the user's personal information like navigation history, contacts, preferences, personal details, etc. It is described in Section 6.

The objectives of this deliverable partially address the following objectives of WP2 described in the Grant Agreement:

- *Design and develop a system able to empower the users to control their consent settings in multiple account and services. It should have an easy-to-use interface and provide auto configuration options to make it easier for the users to configure complex scenarios by using aggregated/crowdsourced data of the different users to build a set of common profiles.*
- *Design the Privacy Metrics to i) unveil and communicate end users the data collected by online services, ii) automatically identify and pinpoint possible privacy violations in data collection, iii) communicate these findings to the end users in an easy and intuitive user interface.*

- *Develop a set of general-purpose building blocks to analyse users' data without affecting their privacy. It will offer some algorithms and methodologies able to provide a certain level of anonymity using concepts as zero-knowledge proof or k-anonymity.*

We use as reference the Deliverable D1.1, in which we precisely defined the requirements of the modules included in the Work Package 2 whose design is here described. As such, we defer the reader to the Deliverable 1.1 for a broader description of the PIMCity project. Moreover, details on the other PDK modules can be found in Deliverables 3.2 and 4.1.

# 3.- Personal Consent Manager (P-CM)

## 3.1.- Objective

In this section we introduce PIMCity's Personal Consent Manager, also known as P-CM.

Its primary objective is to give the users the transparency and control over their data in a GDPR compliant way. That is, give them the possibility to decide which data can be uploaded and stored in the platform, as well as how (raw, extracted or aggregated) data can be shared with Data Buyers in exchange for value when the opportunity arises.

The P-CM is presented as a web application and a REST API, not only providing users the possibility to use the component in a user-friendly way, but also enabling developers to integrate PIMCity Consent Management capabilities in their products.

Consent Management over the Internet is still brand new as regulations define new rights and responsibilities. Therefore, there is no standard way of gathering and providing consents to store and use data. With PIMCity's P-CM, we aim at:

- Defining a standard interface to store, update, delete, audit and activate consents received for multiple purposes, such as storing data and sharing it with third parties.
- Providing a web application and a REST API as an implementation of such standard interface.
- Implementing state-of-the-art techniques, using standard community-approved software libraries.
- Putting in place an integration flow that eases the deploy and allows for quick upgrades to the tool in the production environment.
- Making the component efficient, fast and scalable to accommodate the consents of a very large set of users.

## 3.2.- Background and state of the art

The General Data Protection Regulation (GDPR) [1] was made compulsory in May 2018. At that time, and still now, a huge number of companies are not fully prepared to comply with it. Even, some of them are widely apart from being compliant in the short term[1]. The regulation challenges all organizations on different areas such as the administrative, legal and technical ones. More specifically, they are challenged in the ability to keep data secure and updated at high scales, or even automate data protection processes. Not complying to

[1] https://www.technologymagazine.com/data-and-data-analytics/capgemini-85-companies-are-not-ready-gdpr

the rules is not an option, since fines can reach 20 million euros or 4% of the company's total annual global revenues.

Moreover, the value of the data market has grown significantly in the last seven years, e.g. it grew by 9% just in 2017 and it is expected to get over 60 billion Euros after 2020. Nevertheless, the difficulties faced by the companies to comply with GDPR makes it utterly hard to keep monetizing personal data, or even data in general, since there is no clear way for other businesses to verify the conditions under which the data was collected in the first place, exposing themselves to huge fines for using or acquiring data without its proper consent. This makes businesses increase the lack of trust for organizations that sell (or re-sell) data, preventing the data market from growing.

To tackle these problems and seize the opportunity of establishing a standard way of collecting consents along with data, several state-of-the-art solutions came to existence.

**Consent Manager by *PrivacyCloud***

The PrivacyCloud Consent Manager Chrome extension[2] removes most cookie banners and declines consent by omission. It will decline all cookies on your behalf by automatically removing most consent-gathering notifications on the fly. Since no positive action will have taken place (unless you "pause" Consent Manager), you will have said "no to all" while enjoying a smooth browsing experience.

They also notify the user by flagging any company or website that is not complying with the user's revoked consent or even the ones serving cookies by default as the user is browsing. The extension overlays the number of cookies being served illegally on top of the Consent Manager icon.

The product has more than three thousand active users.

**Matomo Analytics**

Matomo Analytics provides an analytics platform[3] as an alternative to Google Analytics, but focused on respecting users' privacy. That is, they enable consent gathering and maintenance out-of-the-box and they keep all the data management in compliance with GDPR (or even stricter cases like the French CNIL regulation). They also claim that they don't use the data their customers collect, as opposed to Google that does it to build profiles of internet users for their own re-marketing purposes.

Matomo Analytics are open source and can be used for free on the customer's premises and under its own maintenance. However, they also offer a paid version that can be run on their infrastructure.

**GDPR Transparency and Consent Framework by *IAB Tech Lab***

IAB Europe established the Transparency and Consent Framework (TCF)[4] standard to support compliance with the GDPR in the context of digital advertising. This framework is built on four components: a Global Vendor List (GVL), a Transparency and Consent String (TC String), an API for Consent Management Providers (CMPs) to create and process the TC String, and the Policies that govern how the TCF is used.

[2] https://chrome.google.com/webstore/detail/consent-manager/gpkoajillfmlpnglbagpplnphadbfalh
[3] https://matomo.org/feature-overview/
[4] https://iabeurope.eu/press-releases/iab-europe-iab-tech-lab-release-updated-transparency-consent-framework/

Prescribed use of the TCF may support compliance with the GDPR, but the real benefit to the digital advertising ecosystem is a safer Internet for consumers, and more reliable data for brands and publishers. As adoption of the TCF increases, the ecosystem becomes more standard, leaving room for automation and therefore compliance becomes more scalable and data becomes more meaningful.

To participate in the use of the TCF, vendors must make a public attestation of compliance with the Policies for using it. To have transparency and consent established and signalled status for their online services stored in a global database, they apply to be added to the GVL. To play a role in creating a TC String for signalling status on transparency and user consent, they sign up with IAB Europe to become a CMP. CMPs must follow technical standards provided in this document for creating TC Strings in compliance with TCF Policies. They must also follow technical standards guidance for using the CMP API specified in this document to receive and process information provided in a TC String.

**Quantcast Choice**

Quantcast Choice is a consent management provider (CMP)[5] solution built on and registered with the IAB Europe Transparency and Consent framework. The product has been tested by international publishers and advertisers on both sides of the Atlantic.

Consumers visiting sites with Quantcast Choice will see one among a number of available user-interface options (all of which can be customized by the website operator) asking the consumer to grant consent for their data to be used in line with GDPR.

Quantcast Choice offers multiple design templates to enable website operators to customize the look of the consent interface. Regardless of the template chosen, consumers will be able to access granular options, such as vendor or purpose-level consent preferences. The consent options the consumer selects will then be applied when the consumer engages with the rest of that site, and potentially other sites and vendors using the IAB Europe framework, depending on selected consent preferences.

**Wibson**

Wibson was launched initially as a blockchain based decentralized data marketplace where users could share their personal data to companies and obtain value for it [2].

Nowadays, Wibson also included new features to their app, now users also can access, manage and control their personal data.

Users install the app and get a complete list of companies that are holding their data, which types of data these companies have about them and have the option to request their data deletion and exercise their data rights[6].

Wibson data management app works as an easy PIMS and its features are listed below:

- Discover:
  - Show a full list of companies that are holding user's data and which data these companies have from them;
  - Map of the world showing where user's data is being used;
  - If a company leaked user's data.

[5] https://www.quantcast.com/blog/quantcast-choice-your-solution-for-gdpr-consent/#:~:text=What%20is%20Quantcast%20Choice%3F,both%20sides%20of%20the%20Atlantic.
[6] https://wibson.io/

- Control:
  - Possibility to request to delete account and data from a specific company;
  - Stop spams from a company.

## 3.3.- Technical Design

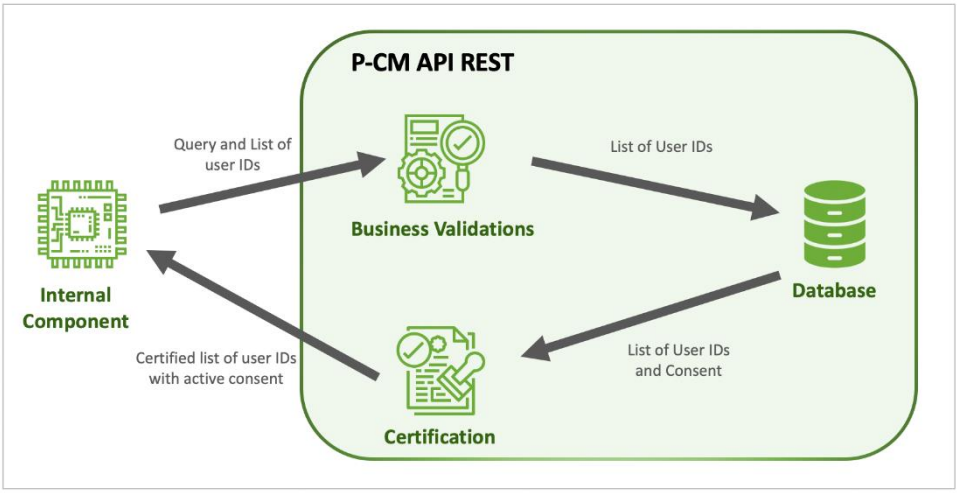
### 3.3.1.-Internal Operation

**Basic Functionality**

The P-CM maintains its own database to store all the up-to-date users' consents. That is, it receives all requests through an API REST, applies all the validations and modifies the state in the database if needed.

Afterwards, whenever an internal component from the PIMCity platform wants to query for end users with active consents, a list of user IDs is received in the API and the Personal Consent Manager looks into its database to filter out the ones that revoked such access to the data. The resulting list is signed with the P-CM private key, responding the request with the certificate.

In the following figure, we show how an internal component from the PIMCity platform can get a certified list of users with active consents for a specific query:




**Roles and Access**

The API is open with three different access levels:

- To the public
- To End Users
- To Internal Components

Therefore, the API defines two different roles:

- End User
- Internal Component

All endpoints related to reading, writing and removing any consent to upload data to the platform or the ability to share it with third parties are restricted to the End User role. Apart from that, an Authorization header is required to allow only the owner of those consents to perform such operations.

On the other hand, all endpoints related to querying for consents and signed certificates are restricted to the Internal Component role, i.e., only an internal component from the PIMCity platform can obtain a certified list of users with active consents.

Moreover, endpoints related to the sign in and out of users are restricted to End User and Internal Component roles for proper authentication.

Last but not least, endpoints related to the general status of the service, like exposing whether it is online or which version is deployed are open to the public.

**Architecture**

The P-CM is divided in four main sub-components:

- Web Interface: Front-end application used by End Users.
- API: REST back-end service that uses JSON documents for payloads. Used by End Users, Internal Components and the public.
- Database: Key-Value store to persist the P-CM state. Accessed only by the P-CM API.
- Documentation Website: Interactive online website showing the documentation for the API REST.

**Data Model**

The P-CM uses three different Key-Value stores to persist the business state:

- Users Store: It stores the name, email and account creation date for every user.
- Data Consents Store: It stores each consent given by the user to store a piece of data in the PIMCity platform.
- Data Sharing Consents Store: It stores each consent given by the user to share a piece of data with a specific company for a specific purpose.

### 3.3.2.-Interfaces

**Web Application**

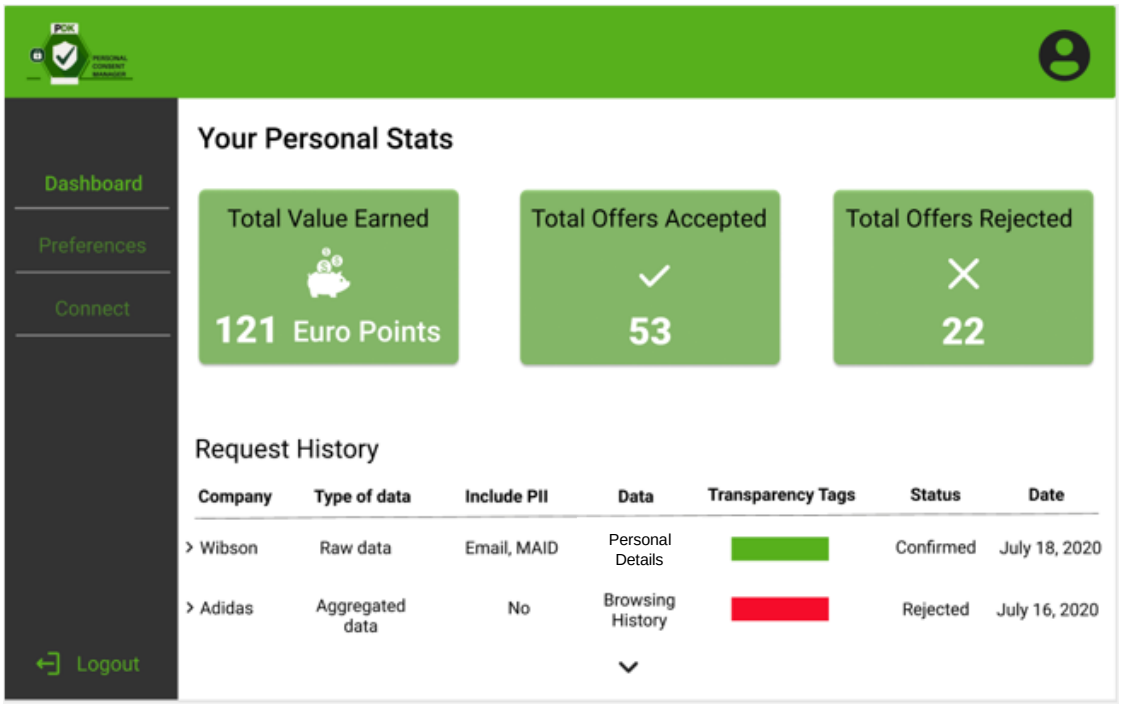
For the end user we propose a friendly web interface for the end user. This platform must have a simple but practical user interface. Users must gain the transparency and control they deserve without being a data or technical expert. This platform aims to be used by any "data owner" (user with personal data).

The web application will have different sections to administrate and select which data to share, to accept data offers, to change personal data preferences, to see a balance of the

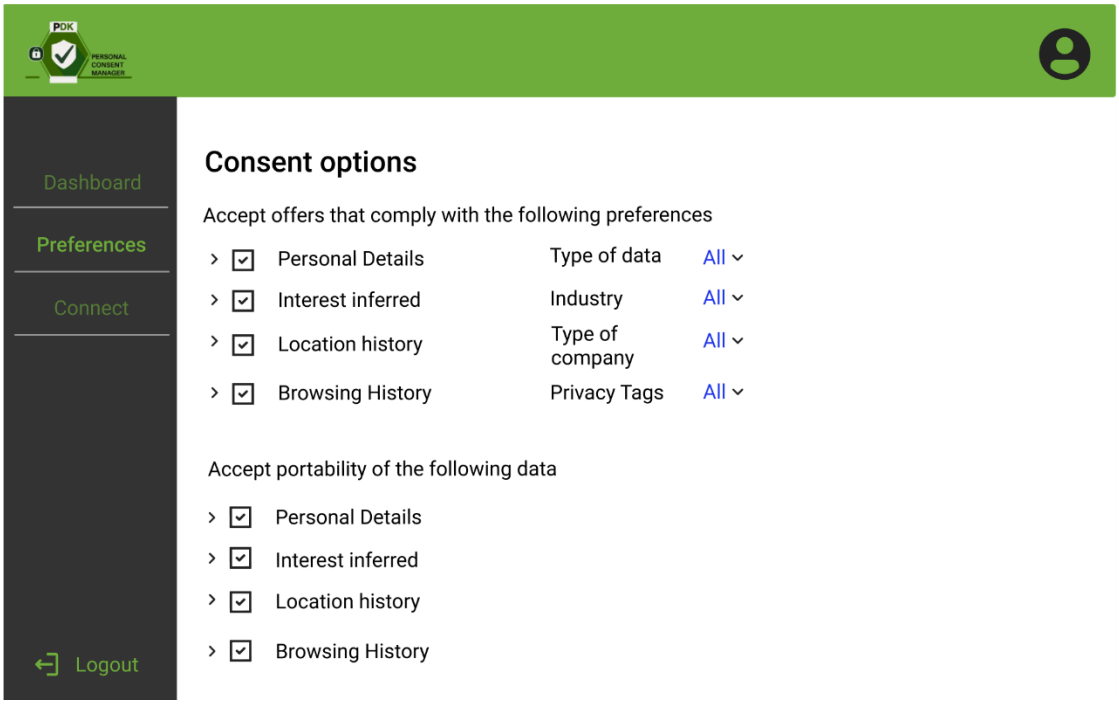
value generated by sharing their personal information and a transaction history to have always a register of who has their data.

Below there are two mock-ups of how this web application could be with their main features.

1. Dashboard is the main section of the web application with an overview of the main metrics and a transaction history:
   - friendly dashboard with the actual status of
     - how they have been sharing their personal data
     - what value have they obtained until that moment
   - List of history transaction to validate and have a copy of
     - who has their personal data
     - which type of data
     - since when a third party is using their data
     - other details regard the data transactions and data itself.




2. Preferences tab is the section where users can change and edit their data sharing preferences:
   - Edit/ change consent automated preferences to potentially
     - Accept data offers in an automated way
   - Select which type of data they are willing to share
     - Whitelist companies or industries




### API REST

The API REST is open for the following roles:

- End User: An individual giving and revoking consents to upload data to the platform and share it with Data Buyers;
- Internal Component: a PIMCity service or component using the P-CM API, such as the Trading Engine;
- Public: No restriction applied.

The API REST service then offers the following functionalities:

- Account Management: Sign in, sign out, sign up and profile details.
- Consent Management for storing data with the platform: List, add and revoke consents.
- Consent Management for sharing data with third-parties: List, add and revoke consents.
- Consents Certificates: Provide a signed certificate with the active consents for data storage and data sharing.
- Service Status: Anyone can check if the service is up or not.

### API REST Online Documentation Website

Since the API can be used directly by other Internal Components or End Users, it ends up being really powerful and handy to have an online interactive documentation website for the service. By visiting it, the reader can quickly understand and try out the API functionalities.


swagger
/api-docs.json
Explore
Personal Consent Manager API
1.0.0
Authorize
POST /accounts/sign-up
POST /accounts/sign-in
POST /accounts/sign-out
GET /accounts/me
POST /consents/data
GET /consents/data
POST /consents/data-sharing
GET /consents/data-sharing
POST /certificates/data-sharing

# 4.- Personal Privacy Metrics (P-PM)

## 4.1.- Objective

In this section we introduce PIMCity P-PMs, the means to increase users' awareness about data they share with services. For each service, this module collects and generates privacy and transparency metrics, based on the data that the module is collecting, and the information provided by the service itself. For instance, P-PMs report information about which personal data is collected and shared by data buyers. All these pieces of information are key for a user to take informed decisions and authorize interactions with services. The P-PM presents this information using a standard interface, thus offering an open knowledge information system which can be queried using standard interfaces.

With the P-PM module, we aim to satisfy the following needs:

- The P-PM should generate and provide easy-to-understand, complete, and useful information to allow users understanding the nature of a service (e.g., a website or a data buyer): reputation, trustworthiness, aim and business.
- The P-PM should contain high-level information easy enough to be understood by inexpert users, as well as more detailed, technical information to satisfy the tech-savvy ones.
- The P-PM should summarize and integrate both information actively provided by data buyers and information collected using automatic methodologies, possibly highlighting contradictions.
- The P-PM should be made available in the format of Transparency Tags and presented to the user using the Dashboard.

## 4.2.- Background and state of the art

In the last decade many initiatives have focused on the design of tools to increase the user's awareness regarding privacy. In the following, we resume them and briefly discuss their design and goals.

**Privacy Notices**

To the best of our knowledge, the first idea of building pieces of information to inform users about privacy practices has been developed in 2009 within a project held by Carniege Mellon's CUPS (Cylab Usable Privacy and Security Laboratory) team[7]. The project aimed to develop Privacy Nutrition Labels to allow users understand and compare privacy policies made available by web services. Despite the US' Federal Trade Commission officially supported the project and recommended the usage of such approach the project did not take hold in the industry, thus vanishing its efforts.

In 2015 Metwalley *et al.* proposed in [3] the design of a distributed system called CrowdSurf, to empower the means for users to get information about web services, supervise the data shared with them, and ultimately take informed decisions regarding their privacy. The paper presents the overall architecture of the system, describes how its elements should interact

[7] https://cups.cs.cmu.edu/privacyLabel/

with each other, but does not describe which information should be provided to users in details and how.

In 2018, Karaj *et al.* proposed in [4] the design and results of Whotracks.me, a system to collect and present information about web tracking ecosystem to integrate in Cliqz browser. The system collects and presents a wide number of information about websites, tracking and advertisement services, with many detailed statistics and scores to make the user aware of which personal data such services collect and share. Whotracks.me represents a significant source of inspiration for the design of P-PM.

**Collection and analysis of information about web services**

There exist a wide number of works, projects and products which design and implement tools to automatically infer information about web services. These systems usually build on active measurements (web scrapers) to automatically visit websites, download data and later process it to perform classification for privacy analysis purposes. In this family it is worth citing Englehardt and Narayanan [5], Rizzo *et al.* [6], Matic *et al.* [7]. Usually, these systems focus on the complex problem of automatically collecting web information. They provide the tools rather than synthetizing the results and conceiving conclusions in an easy-to-understand format. Nevertheless, the web data collection is a fundamental part for the generation of P-PM.

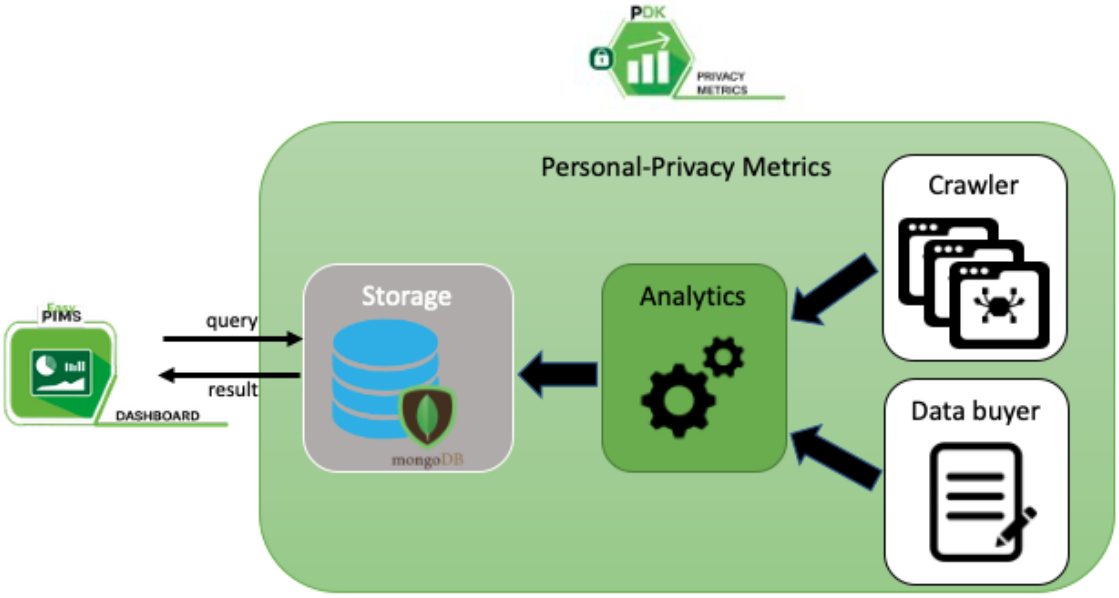
Considering the industry, there are several commercial products which provide web service classification functionalities. Among them, we mention zVelo[8], Similarweb[9], and Symantec[10]. However, little is known on which data is used for the classification and how this is actually performed[11].

## 4.3.- Technical Design

### 4.3.1.-Internal Operation

P-PM will be generated based on information actively provided by services participating PIMCity (i.e., data buyers) integrated with data gathered from measurement campaigns run using automatic web crawlers. Hence, the module is designed as a database populated with information from data buyers and automatic crawlers. Such information will be made available in read-only mode to the Dashboard which will present P-PMs as Transparency Tags. The picture below summarizes the generic design of P-PM components. Depending on P-PM provider's requirements, the P-PM may contain the database element only, or all the elements needed to build the whole data analysis, collection and storage workflow.

[8] zVelo https://www.zvelo.com/
[9] Similarweb https://www.similarweb.com/
[10] Symantec BlueCoat https://sitereview.bluecoat.com/#/
[11] https://eprints.networks.imdea.org/2183/1/paper.pdf




The fundamental block of the above architecture is the database whose content is firstly defined by schemas it will support. We envision a generic schema to use and adapt depending on the use case scenario. We describe it in the following table.

| P-PM template | | |
|---|---|---|
| Information from data buyer | Declared company name | *Company name as declared by the company itself* |
| | Website | *Website provided by the company* |
| | Country | *Establishment country* |
| | Category | *Self-declared categorisation information based on kind of business and mission* |
| | Collected data | *Describes which kind of data is collected* |
| | Purpose of data collection | *Describes how collected data will be used by the company. The description of data processing purpose must be meaningful and sufficient to base consent on.* |
| | Data Owner | *Describes who owns the data subject to processing* |
| | Data Processor | *Describes who is going to process data* |

| | | |
|---|---|---|
| | Data Subprocessors | *Describes possible sub-processors involved in the data processing* |
| | Data retention period | *Describes how long data is going to be stored by the data owner* |
| | Data location | *Describes where data is going to be physically stored (in or outside EU)* |
| | Contacts | *Mail address to contact to get information* |
| Web data | Company name | *Company name gathered by crawler* |
| | Operates under | *Website(s) gathered by crawler* |
| | Category | *Categorisation information obtained by crawler* |
| | Rank in category | *Popularity based on users' traffic* |
| | Connected third-party services | *List of third-party services connected to main website* |
| | Connected websites | *List of websites where the company appears as a third party* |
| | Connected categories | *List of website categories associated to this website* |
| | Tracking devices | *List of user tracking technologies* |
| | Privacy policy | *URL to privacy policy* |
| | Presence in security lists | *Website/company has been reported as risky* |
| | Profiling purposes | *Describes which information is extracted from user's data* |
| Scores | Privacy score | *[0 to 5] based on privacy-related information* |
| | Transparency score | *[0 to 5] based on matching between provided and gathered data* |

| | | |
|---|---|---|
| | Security score | *[0 to 5] based on security-related information* |

- **Information from data buyers**: this is information actively provided by data buyers participating a PIMCity deployment. Data buyers will be required to provide this information themselves for transparency purposes. In general, we recommend to avoid the usage of freetext data fields as these are not sufficient to get meaningful descriptions. For instance, for the case of "Purposes of data collection" field, we must limit the choice to a set of purposes that are legally sufficient to base consent on. For this, it could be helpful to provide a set of meaningful purposes in a drop-down list, and give additional explanation and examples when filling the form (e.g., tooltips popping up with mouse-over events). Examples of data processing purposes have been provided in D1.1 and D1.2.
- **Information from web data:** This will present information obtained by processing web data gathered from web crawlers. In particular, web crawlers will automatically visit the website (or the list of websites) provided by the data buyer and collect information useful to perform classification.
- **Scores:** these will be computed based on both information provided by data buyers and data collected from web crawlers. To enhance competitiveness among different P-PM providers, we decide not to provide specific guidelines for the computation of scores. Hence, P-PM providers are free to choose the data, methods and algorithms for computing the scores, and possibly describe the generation method behind. For instance, the P-PM provider may decide to modulate the Transparency Score based on a comparison among information provided by the data buyer and public web data.

We remark that the schema presented above represents a design proposal that may evolve during next design cycles because of new user requirements or implementation constraints. Furthermore, schemas may change depending on the P-PM provider: for instance, further information might be provided, or removed depending on availability, technical issues or others. Next, we provide some practical examples of P-PMs.

1. **P-PM for data buyer participating PIMCity**

| **P-PM for ACME Data Buyer** | | |
|---|---|---|
| Information from data buyer | Declared company name | *ACME Ltd* |
| | Website | *https://www.acme.com* |
| | Country | *US* |
| | Category | *Analytics* |
| | Purpose of data collection | *1) Basic Interaction and functionalities*<br>*2) Measurement* |
| | Data owner | *User* |
| | Data processor | *John Doe - ACME Ltd* |
| | Data sub-processors | *NA* |

| | | |
|---|---|---|
| | Data retention period | *5 years* |
| | Data location | *Ireland* |
| | Contacts | *Info@acme.com* |
| Web data | Company name | *ACME Group Ltd* |
| | Operates under | *https://www.acme.com* |
| | Category | *Marketing, analytics* |
| | Rank in category | *40/2344* |
| | Connected third-party services | *https://www.analytics.com*<br>*https://www.cdn.com*<br>*https://www.social.com* |
| | Connected websites | *https://www.acme-retail.com* |
| | Connected categories | *Shopping, social network* |
| | Tracking devices | *Cookies* |
| | Privacy policy | *https://www.acme.com/privacy* |
| | Presence in security lists | *No* |
| | Profiling purposes | *Age, job position, net income, family role* |
| Scores | Privacy score | *3/5* |
| | Transparency score | *2.5/5* |
| | Security score | *4.5/5* |

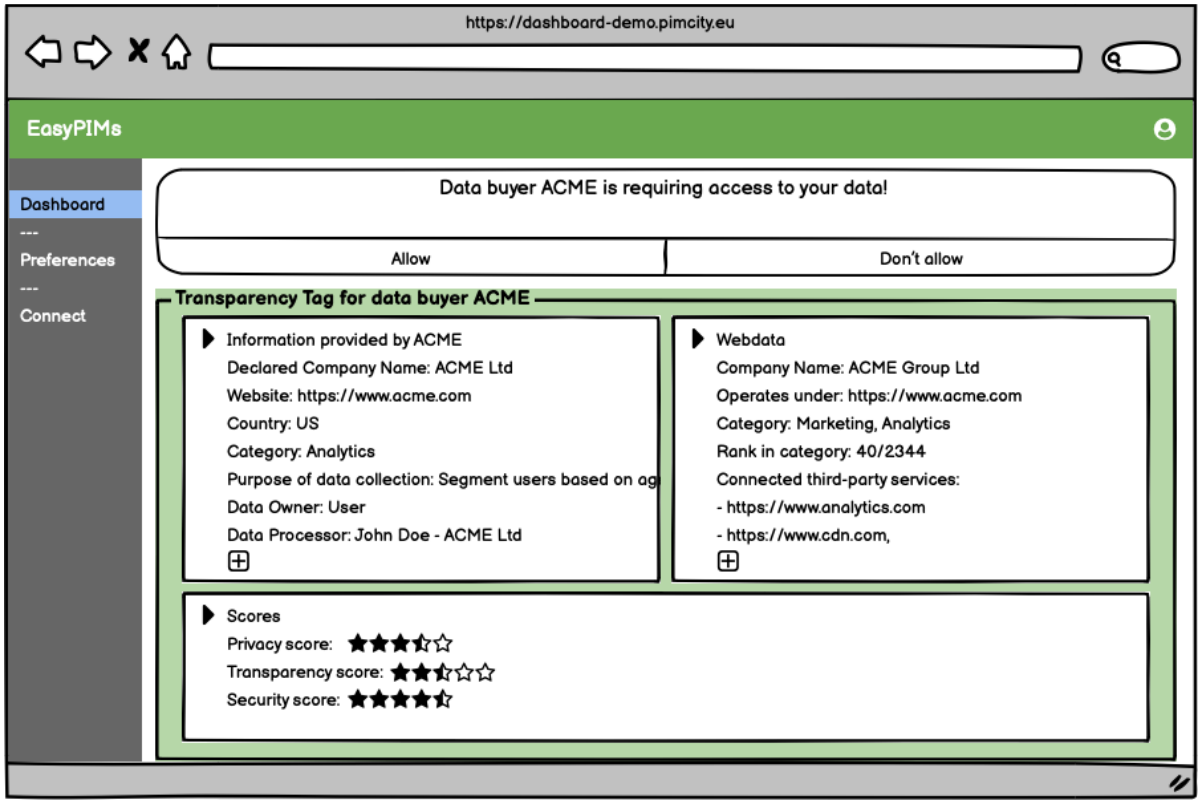
## 2. P-PM for service external to PIMCity

| **P-PM for EMCA service** | | |
|---|---|---|
| Information from data buyer | Declared company name | *NA* |
| | Website | *NA* |
| | Country | *NA* |
| | Category | *NA* |
| | Purpose of data collection | *NA* |
| | Data owner | *NA* |
| | Data processor | *NA* |
| | Data sub-processor | *NA* |
| | Data retention period | *NA* |

| | | |
|---|---|---|
| | Data location | *NA* |
| | Contacts | *NA* |
| Web data | Company name | *EMCA Ltd* |
| | Operates under | *https://track.emca.com*<br>*https://cdn.emca.com* |
| | Category | *Tracking* |
| | Rank in category | *2/4050* |
| | Connected third-party services | *https://www.emca-social.com* |
| | Connected websites | *https://www.news.com*<br>*https://www.sport.com*<br>*https://www.entertainment.com*<br>*https://www.payments.com*<br>*https://www.bank.com*<br>*https://www.social.com* |
| | Connected categories | *Shopping, social network, news, banking, electronic payments* |
| | Tracking devices | *Cookies, fingerprinting* |
| | Privacy policy | *NA* |
| | Presence in security lists | *Yes* |
| | Profiling | *Sexual orientation, consumer profiling,* |
| Scores | Privacy score | *1.5/5* |
| | Transparency score | *1.5/5* |
| | Security score | *2/5* |

### 4.3.2.-Interfaces

The P-PM module interacts with the Dashboard. The Dashboard web app queries the DB contained in P-PM in read-only mode to fetch privacy metrics in JSON-like format and present them as per-data-buyer/service Transparency Tags. This can be performed to provide users information to take decisions during the configuration of the consent manager, or simply to provide a browsable collection of P-PMs using their interface, the Transparency Tags. In the following we present a simple mockup draft for consultation of P-PMs using the interface provided by Transparency Tags:




We emphasize that the development of the interface (i.e., a web page or an app) to present the content of P-PMs will be discussed when we will design the Transparency Tags in WP5. Hence, for the sake of flexibility, choices strictly regarding the presentation of P-PMs will be discussed in the design phase of the Transparency Tags. For instance, we will let the designer decide whether to develop proper UIs to allow the user to compare information provided by data buyers and public web data or aggregate information depending on user's technical skills.

The Dashboard component will receive P-PMs in JSON-like format via the API. Along with the query generated by the Dashboard, some metadata about the request is broadcasted as well, in order to authenticate the inquiring party and to check other requirements.

| **Personal Privacy Metrics APIs** | |
|---|---|
| GET | /privacymetrics/*serviceX* |
| GET | /privacymetrics/services |

# 5.- Personal Privacy Preserving Analytics (P-PPA)

## 5.1.- Objective

In this section we introduce PIMCity's P-PPA, a tool to allow analysts and stakeholders to retrieve useful information from personal data provided by the users, while preserving their privacy. When talking about analysis of personal data, we must balance the aim of companies to draw knowledge from personal data -also by the means of Machine Learning (ML) algorithms -and users' requirement of maintaining privacy intact by not exposing their private information. However, usually the user wants to achieve this goal – preserving its own privacy – without having to give up the online service she is providing her data to.

With the P-PPA module, our goal is to achieve the following objectives:

- Personal data exporting to third parties is kept under control with the privacy-preserving goal in mind.
- The P-PPA should be able to provide different types of output, depending of the need of the analysis: this includes aggregating data, minimizing loss of information and more.
- The P-PPA should act both in a static way (for analysts who require already-cumulated data) and in a dynamic way (for those who are instead interested in analysing a stream of incoming data, with the minimum possible delay).
- We want to design P-PPA component in a modular and extensible way, in order to include possible future mechanism at a low cost

Finally, notice that the Data Aggregation PDK module provides data anonymization techniques, but covers a different use case (see Deliverable 4.1). It provides anonymization functions to allow bulk insert of data into EasyPIMS, while the P-PPA regulate how these data are exported to the stakeholders.

## 5.2.- Background and state of the art

In the past decades, several data anonymization techniques have been proposed to build privacy-preserving analytics. Since the analytics are strictly tailored to the needs of the inquirer, there exists no standard approach, while many alternatives aim at solving the problem in different scenarios. In the following, we present the most popular data anonymization approaches.

**K-anonymity**

Many researches in the past years showed that removing the identifiers from a dataset is not a sufficient way to protect privacy: in 1997, Sweeney was able to identify the Massachusetts governor in a publicly listed set of medical records, using its birth date, its zip code and its gender [8]. In 2006, Netflix published a dataset while launching a contest to improve its recommendation algorithm: the dataset consisted of rating provided by users, whose identities were not revealed. However, linking the information provided by Netflix with the one publicly available on sites like IMDB, Narayanan and Shmatikov were able to identify some of the Netflix users [9].

K-anonymity [10] aims at reducing the probability for a user in a dataset to be identified down to 1/k. The goal of k-anonymity is to group the users according to their quasi-identifiers

(those attributes who are apparently harmless, but whose combination may lead to re-identification – such as zip code, birth date and so on) and enforce that each one of these groups (called *Equivalent Classes* – ECs) contains at least k rows of the dataset – i.e., k users: it is common to assume that each user only appears in a row. This property guarantees that if a user is linked with a sensitive information (such as a medical record, the income, and so on), it can hide himself among other k-1 users. K-anonymity is typically achieved through two methods: generalization and suppression. In the latter method, some rows may be removed from the published dataset in order to form ECs with appropriate size. With generalization, instead, the single attributes may be modified in order to find k indistinguishable users. For numeric attributes, the actual number is replaced by a range that includes it. For categorical attributes, a common parent value in a hierarchy tree may replace the child ones (e.g., two values such as "French" and "Spanish" may be generalized as "European").

Being k-anonymity a data property and not an algorithm, several mechanisms to obtain it have been proposed, such as the Incognito [11] and Mondrian [12] algorithms.

K-anonymity suffers from a number of known flaws that affect its privacy preservation promises. First, it is susceptible of the *homogeneity attack*: if all the users in an equivalence class have the same sensitive attribute, there is no protection for a user in it (we do not know *which her identity*, but we know her sensitive attribute). K-anonymity also suffers *background knowledge attack*. For instance, if we are aware that Japanese people are less affected by heart attack than the average, we could use this information to weaken the k-anonymity protection in a dataset reporting heart attack records of a multi-ethnic population.

Hence, alternatives to k-anonymity are l-diversity and t-closeness. The former adds the constraint «that the values of the sensitive attributes are "well represented" in each group» [13] (and thus faces the homogeneity attack), while the latter requires «that the distribution of a sensitive attribute in any equivalence class is close to the distribution of the attribute in the overall table» [14] (tackling the background knowledge attack). Regardless on the introduction of new privacy properties, k-anonymity, for its simplicity, is still regarded as the golden standard for anonymization.

**Differential privacy**

Differential privacy aims at providing a formal definition of privacy. This definition claims that a user's privacy is protected if the knowledge an attacker has on a queried dataset is not too affected by whether the user is in the dataset or not. The "too affected" notion is parametrized by ε [15].

A common mechanism to obtain differential privacy is to apply the Laplace mechanism, which consists in adding a Laplace-distributed noise to the query result. The noise is extracted from a Laplace distribution which depends both on the ε value and on the *sensitivity* of the query – i.e., on how much the presence or absence of a user can affect the result. It is proved that the Laplace mechanism guarantees differential privacy.

**Existing libraries**

Many libraries implement algorithms for achieving k-anonymity. Some are based on the Mondrian algorithm[12]. Programs exist that apply k-anonymity principles to Microsoft Excel, CSV files or SQL databases tables, privacy-protecting them[13].

For what concerns Differential Privacy, an interesting tool has been provided by IBM[14]. The library offers the possibility to apply differential private algorithms to perform machine learning tasks such as classification and clustering, or to directly interact with the primitives to create differentially private queries to the database.

## 5.3.- Technical Design

### 5.3.1.-Internal Operation

The P-PPA module aims at offering different analytics solutions, depending on the available data and on the requirements of the querying party. Each analytics solution is implemented into a separate function. In our view, the functions should be the following, but a subset or a superset of these may be implemented by the P-PPA provider:

- k-anonymity,
- differential privacy,
- z-anonymity (specifically designed for PIMCity a described below).

The P-PPA module can work both with *historic* data – i.e., values that are single for each user and rarely or not-editable, such as date of birth, address and so on – and with *recurrent data*, which may instead be exposed by the user several times: they include browsing history (one record per site visited by a user), location history (a periodic update on user's location) and similar.

**k-anonymity**

The k-anonymity function aims at implementing k-anonymity on incoming datasets, especially those composed of historic, static datasets. We assume that the data will arrive in a structured form (e.g., a Python Pandas Dataframe object). The function will take the data and k-anonymize it; to perform k-anonymization, we will use existing open-source implementations. They will be tested to be ensure they work in a correct and efficient way. This holds for l-diversity and t-closeness.

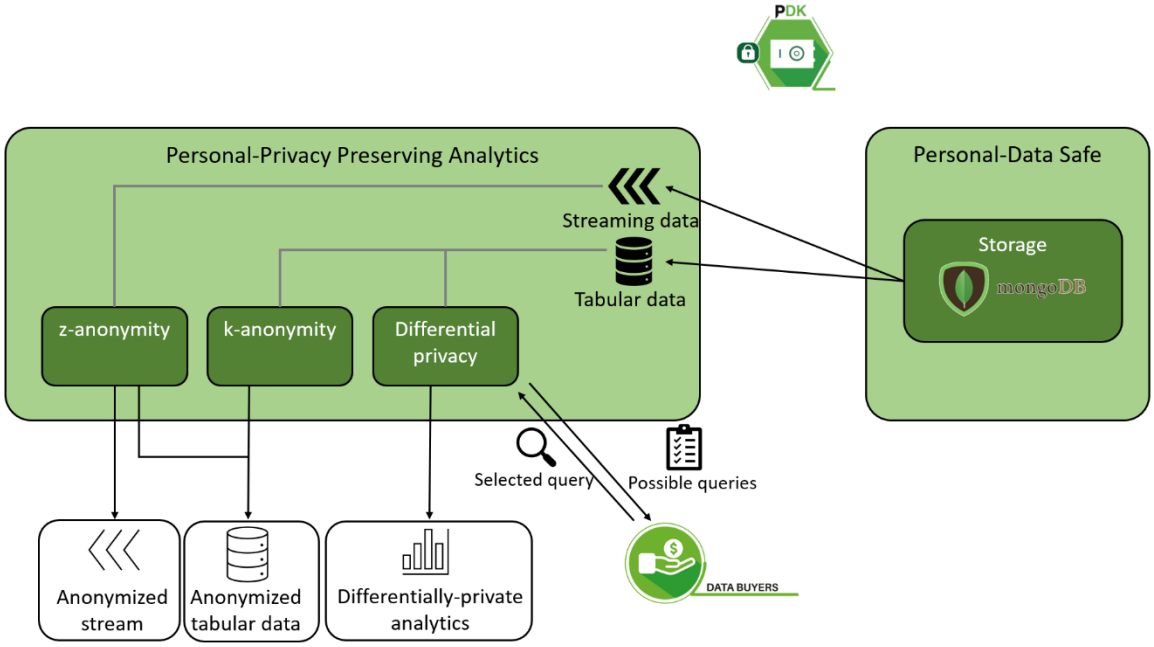
At the end of the anonymization process, the data can be provided to the third party in the same structural form it has been received by the P-DS.

Notice that k-anonymity is adopted in the Data Aggregation (DA) PDK module as well, with the goal of achieveing anonymized bulk insert into EasyPIMs. However, it has a very different goal, as it seeks at anonymizing data at insertion-time, while the P-PPA aim at providing the data buyers with anonymized data. More details on the DA module can be found in Deliverable 4.1.

[12] https://github.com/qiyuangong/Mondrian
[13] https://github.com/arx-deidentifier/arx
[14] https://github.com/IBM/differential-privacy-library

### Differential privacy

For the differential privacy function, the P-PPA module exposes a set of queries that the querying party can use on the dataset. The queries could be stand-alone or parametrized, with the possibility for the inquirer to model the query following its own preferences, while maintaining a query structure that guarantees differential privacy.




Following the principles of differential privacy, every querying party is given a *privacy budget* to spend, meaning that it cannot perform an unlimited number of queries. Indeed, each query sent to the P-PPA has a privacy cost that decrements the budget of the querying party. To avoid behind-the-scenes data exchange between querying parties to avoid the budget constraint, a global budget could also be set. The privacy budget could be checked on a moving time window.

A differentially-private algorithm returns the query results as a JSON file, composed by key-value pairs: a single pair is needed if the output of a query is a value, several pairs are used if the output consists of a histogram of values, whose bin values are used as key of the JSON file, and whose value are the respective values.

When inquiring a static dataset, the differential privacy function may work especially well for historic data.

### z-anonymity

Another module inside P-PPA implements the *z-anonymity* [16]. z-anonymity is a privacy property that aims at protecting *z-private* attributes. An attribute is said to be z-private if it has not been exposed by at least z users in the past time period Δt.

z-anonymity is suited especially for streaming datasets, that need to be analyzed while protecting users' privacy – e.g., web traffic monitoring. It guarantees a zero-delay (hence the name) evaluation of the attribute privacy and its potential output. If the attribute is not z-private, z-anonymity assumes that the information is not potentially harmful for the user and thus can be published (the user's identifying attributes, such as the name, are replaced by

random, identifying codes that are assigned to each user). Otherwise, the incoming information is blurred not to risk user's private information leakage.

By design, the z-anonymity function suits the usage of recurrent data.

### 5.3.2.-Interfaces

The P-PPA module interacts with two separate entities: on one side, it obtains data from the P-DS module. On the other one, it interacts with the parties requiring the data for analysis.

**Interface with P-DS**

To retrieve the data to treat in order to deliver P-PPA, the module has to receive the users' data via the P-DS module (see section 6). In particular, the data will be retrieved from the database that lays in the P-DS – whose choice is detailed in Section 6.3.1.

The module receives the data in a JSON-like format. The file(s) to retrieve from the database is evaluated by parsing the queries from the data-querying parties (see the next section for details).

The P-PPA module can offer the results under several shapes, related to which of the three functions have been applied to the incoming data: k-anonymization returns another table with the same format of the input, but with generalized (and/or suppressed, depending on the used algorithm) attributes to ensure users' anonymity. For what concerns differentially private queries, the results may be broadcasted in the form of a single value or a histogram, depending on the query that have been required to the module. Finally, the output of the z-anonymity module can be both broadcasted as a table – which cumulates the real-time output of the system – or tuple-by-tuple. All the query results will be offered to the inquiring third parties as an answer to the query, through an API.

**Interface with data-inquiring parties**

The data-inquiring parties explicit their interest in the data by querying the P-PPA module over one of the functions (z-anonymity, k-anonymity, differential privacy) – or, in alternative, it defines the type of output it desires from a given source, from which the system can infer the optimal function to apply for the task. The set of possible queries is exposed by the P-PPA (e.g., as a JSON-like file). Among them, the inquirer chooses that one that best suits its needs and sends it again in a JSON-like format via the API: it is composed by the required function, the skeleton of the query, plus some extra parameters to possibly customize the query. The queries may have different aspects depending on the module's function the inquiring party is interested into. Along with the query itself, some metadata about the inquiring party is broadcasted as well, in order for the system to authenticate the inquiring party and to check other requirements, e.g., having enough data budget to perform a differentially private query.

# 6.- Personal Data Safe (P-DS)

This section describes the design and implementation of the PIMCity component Personal Data Safe (P-DS). We first introduce the overall goal of the module and then describe the state of the art on the existing solution for the storage of personal data. We also carefully describe the architecture and design of the PIMCity P-DS in terms of internal functioning and interfaces for users and other PIMCity modules.

## 6.1.- Objective

The objective of this section is to describe the PIMCity P-DS, the PDK component to store securely and transparently the users' data, as well as providing a set of innovative functionalities. While regulations such as (GDPR) defined guidelines and rules on personal data management, there is no precise and clear regulation for governing the storage of personal data. Nowadays, users' data are distributed over different systems and platforms: Facebook, for example, knows users' likes and interests, Google holds the list of the web search queries and Amazon the list of purchased items. The European law mandates that users have the right of retrieving their data at any time, but this fragmentation leads to a loss of control because the user does not have a centralized tool for handling these data. A possible solution to this problem is represented by particular storage systems, called Personal Information Management Systems (PIMS), which store data in dedicated Personal Data Safes (P-DS). The objective of this section is to describe the PIMCity Personal-Data Safe.

PIMSs and P-DSs are still at an early stage of development, and there is no standard definition. With the PIMCity P-DS, we aim at fulfilling the following requirements:

- The P-DS is the means to store personal data in a controlled form.
- The data structure shall be flexible in order to accommodate diverse types of user data, such as generic personal details but also, e.g., browsing or location history.
- It shall allow automatic and manual data import for users as well as interfaces for data buyers that shall retrieve data upon the users' consent.
- The P-DS gives users the possibility to store new data manually or import them from existing platforms. As such, it must offer a user interface as well as APIs for automatic data management.
- The P-DS shall also be easy to deploy and use state-of-the-art techniques and software libraries.
- The P-DS shall be efficient, fast and scalable to accommodate the data of a very large set of users.

## 6.2.- Background and state of the art

The advent of the GDPR fostered the creation of personal information management systems. Companies and organizations proposed their own solutions over the last (few) years, without following any standard or common guidelines. Indeed, the world of P-DS is very heterogeneous, and each solution is peculiar, and differs from the others. Here, we aim at providing a brief description of the popular products available nowadays in the market and as open-source solutions.

## OpenPDS

OpenPDS[15] is an open-source project, implemented at the MIT. It consists of a personal metadata management framework that allows individuals to collect, store, and give fine-grained access to their metadata to third parties. It also introduces the SafeAnswer (SA) algorithm, a practical way of protecting the privacy of metadata at an individual level: The SA module processes raw metadata to compute relevant metrics within the safe environment of the PDS. In this way only a processed result is returned to the third parties, that can thus access only aggregated or processed information.

OpenPDS is composed of three main blocks:

- Database: metadata are stored in a CouchDB database[16], a NoSQL store that provides built-in functionalities to reduce the dimensionality of the metadata.
- PDS front-end: composed of several SA modules. All the SA access to the database must be authorized, and each SA module executes inside a sandbox. Only processed data (the so-called safe answers) leave the SA module, so third parties get the minimal amount of information they need and not additional metadata that could harm users' privacy.
- Data requester: any application or website that want to access user personal information.

**Issues**: Although OpenPDS may be a potential standard solution for personal metadata management, it still faces a number of challenges. The development of SafeAnswer privacy-preserving techniques at an individual level hardly scales for high-dimensional and ever-evolving data. Moreover, the development or adaptation of privacy, preserving data-mining algorithms to an ecosystem consisting of distributed PDSs is still an open issue.

## Mydex

The Mydex platform[17] is a commercial tool that provides its users with the Mydex Trusted Framework and Platform, which enables to define single Personal Data Stores. Each PDS allows the collection, the management and the distribution of personal data, that are generally stored in the cloud, where the majority of the operations are performed as well. The Personal Data Store is an independent collection of files that are encrypted using a private encryption key known only by the individual. The data can be accessed uniquely by the owner user and the third party with a permission.

Mydex offers its PDS and additional tools free of charge. The user has full control over his data and can decide what kind of information to insert in the PDS. The user can decide which people or organizations can access and define connections, that enable to send or receive data to or from others.

**Issues**: Mydex is not an open-source platform, and, as such, it is not possible to run new installations of the PDS, neither to modify nor extend it to implement customization or additional functionalities.

[15] https://openpds.media.mit.edu/
[16] https://couchdb.apache.org/
[17] https://mydex.org/

### Hub of All Things

Promoted by HAT Community Foundation, the Hub of All Things (HAT) microserver is a scalable technology to enhance the digital rights of users through the ownership of a personal data server. The HAT microserver is hosted in the cloud and individuals can install plugins to bring their data in from the Internet, exchange data with applications and install tools in their microservers to have private analytics for insights into their data.

A Hat microserver consists of four main components:

- HAT Web Server: each HAT microserver offers web-based APIs, which are customizable by the users.
- HAT Database: users own a HAT Database and they have complete control over the contents of the database. Each HAT Database contains a data schema, allowing to store individual's data from any source without losing the structure specific to the source. The Database is characterized by namespaces, that identify, which group data just like folders in normal operating systems.
- File Storage System: files are held in the Amazon S3 storage system offered by AWS and managed by Dataswift.
- HAT Computation: this component enables the creation of private analytics. HAT Computation provides an environment where third-party code written in different languages can operate on HAT data.

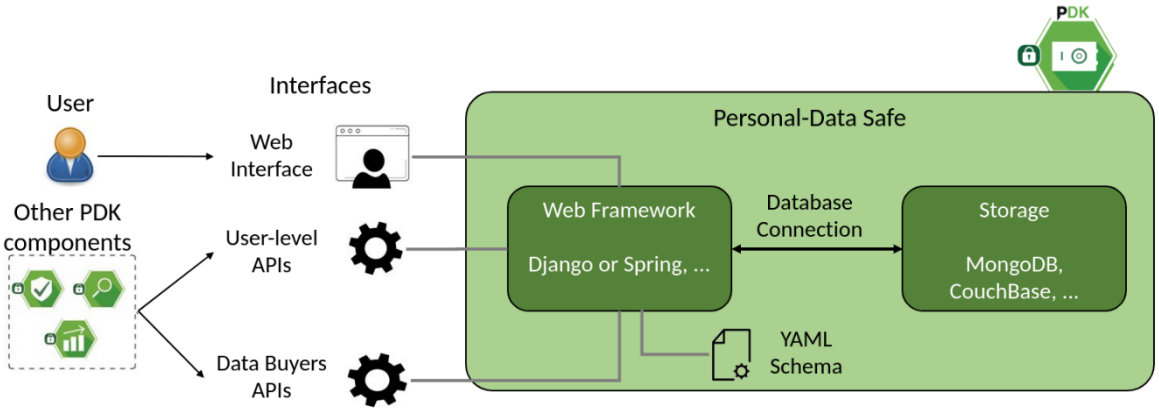
**Issues**: The Hub of All Things, despite being an open-source and flexible solution, also has a few drawbacks. In particular, the extreme customization offered to the users makes it difficult to achieve a minimum data standardization. This is an issue in particular to achieve a transparent data market, in which data buyers shall buy structured user data, which can be used for automatic analyses, data mining and business intelligence. Indeed, we aim at creating a solution that can help users in exchanging their data transparently in a consistent and interoperable fashion. Moreover, it lacks all the surrounding tools fundamental to build a fully-fledged PIMS, e.g., a consent manager or data market. Filling this gap is the goal PIMCity through the PDK modules.

## 6.3.- Technical Design

In this section, we describe the main design choices that have been evaluated for developing the PIMCity P-DS. Indeed, different technical solutions can be exploited for the various aspects of the P-DS, such as the web interface, data storage, etc. Moreover, in this section, we also describe the external interfaces of the P-DS, in terms of user interface and APIs for automatic operations. The PIMCity P-DS is a system which decouples data presentation from the operation and storage. The backend of the system implements the data model and takes care of handling the data, authenticating the users. The frontend provides a simple user interface to allow the user to perform necessary management actions on the stored personal information. Plus, the system provides APIs to allow other components to automatically access data. Indeed, we want the P-DS to be flexible and interoperable, and, like most of the modern IT systems, will be reachable through the network, with a Web interface and network APIs. To improve the modularity and flexibility of the tool, users shall be authenticated *either* locally on the P-DS *or* by external providers

using authorization token based on the standard OAuth[18] or JWT[19] mechanisms. Both ways will be considered.

The overall architecture of the P-DS is depicted in the figure below. The operation of the building blocks is explained in detail in the next sections.




### 6.3.1.-Internal Operation

Here, we describe the P-DS backend, while the interfaces for the users and external APIs are described in the next section. The backend is composed of a web component which handles the access to the P-DS data and a database which stores the data in a reliable and scalable way.

**Web framework**

As mentioned previously, we want the P-DS to be reachable through the network to allow universal access from any user, location and device. To this end, we opt for using a web framework to ease the development of all network-based P-DS components. Today, web frameworks are very common and popular, and they are also used for big and complex applications with thousands of users. They offer useful built-in functionalities (security, authentication, logging, auditing, etc.) and can speed up the implementation as they generally come as middlewares that connect the high-level language used by the developer with the underlying storage layer. This allows the developer to avoid writing boilerplate code to access data.

We will evaluate all the state-of-the art web frameworks and motivate our choice in the next phases of the PIMCity project, which will be covered by the next deliverables. For each candidate web framework, we will evaluate its ease of use, the implemented features, its scalability and security as well as how it integrates with the other project modules. Potential candidates are Django[20], Flask[21] and Spring[22].

**Data storage**

The main goal of the P-DS is to be a secure repository for user personal information, so it is fundamental to choose the nature of the data storage component carefully. We opt to use

[18] https://oauth.net/2/
[19] https://jwt.io/
[20] https://www.djangoproject.com/
[21] https://flask.palletsprojects.com/en/1.1.x/
[22] https://spring.io/

the NoSQL approach, in which the Database Management System abandons the use of the SQL language and prefer simpler APIs. This enables the insertion and the processing of data that do not follow a fixed and rigid schema, very useful in the case of applications that need to evolve rapidly or that deal with mutable data.

To make the final choices, we will evaluate all the state-of-the art NoSQL databases, comparing them in terms of flexibility, ease of use, scalability. Moreover, we will consider how they integrate with the selected Web framework, giving priority to couples of web framework and database for which has been proved a solid and reliable compatibility. We will evaluate as potential candidates MongoDB[23], HBase[24], Couchbase[25] and Redis[26].

**Data Model**

The P-DS configuration is based on a schema, that is stored as a YAML file and contains the possible type of information that can be stored, listing all the possible fields for each type group. The schema has the primary goal of controlling the data that can be inserted on the Personal Data Safe, in order to prevent the user from inserting any data, that may be too complex to handle and process, or just erroneous. On the other side, the schema can be easily changed or expanded, providing the flexibility required by the module goals. Examples of fields that can be inserted in the schema are personal details such as name or phone number, as well as specific data like browsing and location history. Fields are clustered in macro groups, which represent semantic sets of fields organized in a hierarchical structure. We report an example schema in the figure below, in which a field group called "personal-information" defines two string fields called first-name and last-name and date field called "birth-data".

```
name: 'PIMCity default PDS schema'
version: 0.1
author: 'John Doe'
content:
    - group-name: personal-details
      types:
          - name: first-name
            type: string
          - name: last-name
            type: string
          - name: birth-date
            type: date
```

The P-DS will come with a **default schema** covering most of the PIMCity and EasyPIMs use cases. The default schema will contain three macro groups, namely **Personal Details, Location History and Browsing History**. Further customization will however be allowed.

### 6.3.2.-Interfaces

[23] https://www.mongodb.com/it
[24] https://hbase.apache.org/
[25] https://www.couchbase.com/
[26] https://redis.io/

The PIMCity P-DS includes two types of interfaces for external access. First, a web interface allows users to interact with the P-DS directly. Second, web APIs allow automatic access by other systems, helping in obtaining a fully distributed system. The web APIs also provide functionalities to interface with the Data Portability module for portable data transfers between systems. See Deliverable 4.1 for details on the Data Portability.

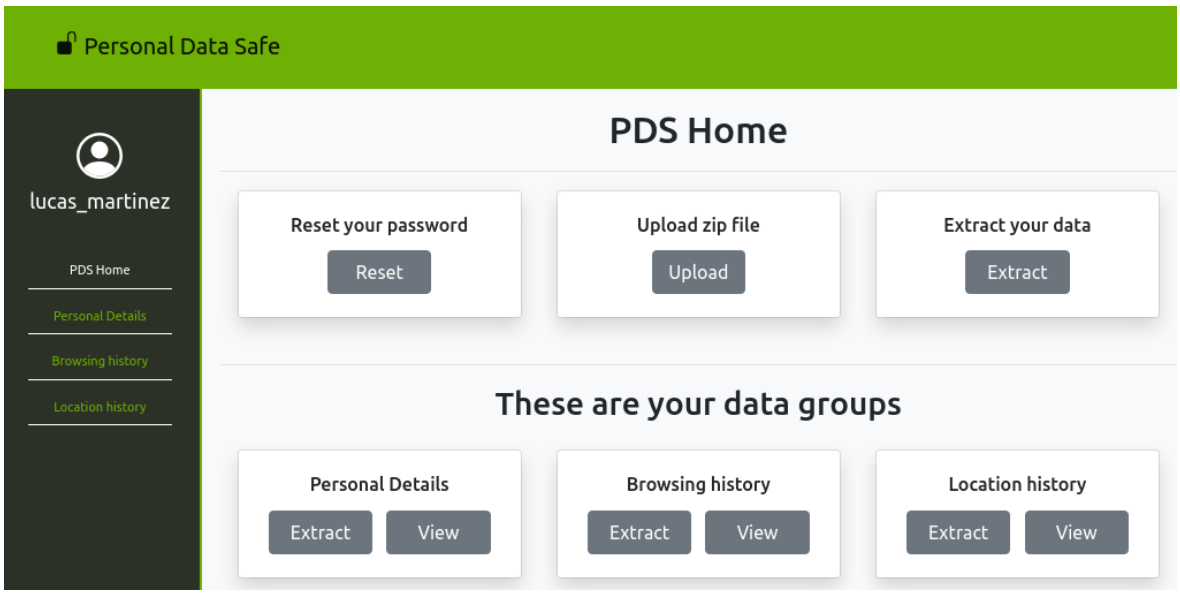
### Web interface

The P-DS is the means the individual can use to store her personal information and, as such, offers a web interface, that enables to view, organize and possibly update or insert data. The user interface is composed by an HTML-based website, that leverage Django templates, the Bootstrap library[27] for the graphic aspect and JavaScript code for all the client-side dynamic parts. . All the elements are designed to be as dynamic as possible, in order to have a structure that can evolve based on the P-DS the schema configuration.

The main actions a user can perform are:

- *Add (or update) personal information:* the user can insert new data entries or modify the existing ones. In order to avoid insertions not compliant with the schema, controls on type consistency are applied on the server-side.
- *Delete personal information*: the user can select a subset of information and delete them.
- *Data Extraction*: the user has the right to get a copy of her data, the user can download a zip file, which contains the stored data entries in JSON format.
- *Visualization*: the user can view the stored data entries, by applying flexible filters (for example by date in case of entries with time information)

Below, we provide a simple mock-up sketching how the user interface may look like. Once logged-in, the user may visualize, insert, modify, add or delete the personal information stored on the P-DS.




### Web APIs

[27] https://getbootstrap.com/

The P-DS functionalities are exposed to external components through REST APIs. REST is the acronym of REpresentational State Transfer and represents a programming approach that enables the building of distributed systems, achieving properties such as scalability, possibility of evolving, efficiency, and resilience. The REST paradigm is based on the client-server model. It has a stateless approach, in which any request from a client to the server must contain all of the information necessary to be understood. The server is composed by one or more services, each of which manages a set of information, that have a globally unique name (URI), can have multiple equivalent representations (generally JSON) and support CRUD operations (Create Read Update Delete). The URLs' names follow a uniform convention, and it fits well with the HTTP protocol, whose verbs support a CRUD environment natively, as in the following list:

- Create → POST.
- Read → GET.
- Update → PUT.
- Delete → DELETE.

We design **two sets of APIs**, one to be used on behalf of the users to manipulate the personal information stored in the P-DS, and the other to be used by data buyers or by any system authorized to retrieve users' data. Indeed, using the REST APIs, any PDK component or application on behalf of the **user** may store new data, which must comply with the P-DS schema. With these APIs, it is also possible to modify or delete existing data. The second set of APIs is reserved for **data buyers**, which have the possibility of retrieving users' data upon receiving their consent from a P-CM. The P-DS will verify the consent presented at request-time by the data buyer (or any system on behalf of it) and will provide the requested data in the response. Suitable security mechanisms will be designed at the project-level to enforce the desired sharing policies. We will use certificate and signature-based security mechanisms to allow data sharing to data buyers.

# 7.- Conclusions

This document describes the high-level design of the PIMCity modules in charge of protecting the users' privacy, enhancing awareness and control, which are part of the WP2. This deliverable defines the tools and the set of techniques to be used to safely store and process users' data. The modules whose first design has been presented in this document are:

- **Personal Consent Manager (P-CM)**, from Task 2.1, the means to define once and for all the user's privacy preferences for consent management.
- **Personal Privacy Metrics (P-PM)**, from Task 2.2, easy to understand novel privacy metrics.
- **Personal Privacy Preserving Analytics (P-PPA),** from Task 2.3, controlling which data users are exposing
- **Personal Data Safe (P-DS),** from Task 2.4, the means to store personal data in a controlled form.

The overall architecture of each module has been discussed with all the project partners to allow easy integration in the PIMCity PDK. Then, the partner responsible for each module has finalized the design in cooperation with the other partners of the WP2. Finally, this document has been reviewed by all the WP2 partners. We refer the reader to the Deliverable 1.1 to have a broader description of the PIMCity project, the other PDK modules and the EasyPIMS architecture. The four modules included in the WP2 will be then implemented in the next months, and the final design as well as the preliminary implementations, will be described in Deliverable 2.2.

# 8.- References

[1] *Regulation (EU) 2016/679 of the European Parlament and of the Council,* 2016.

[2] D. Fernandez, A. Futoransky, G. Ajzenman, M. Travizano and C. Sarraute, *Wibson Protocol for Secure Data Exchange and Batch Payments},,* 2020.

[3] H. Metwalley, S. Traverso, M. Mellia, S. Miskovic and M. Baldi, "Crowdsurf: Empowering transparency in the web," *ACM SIGCOMM Computer Communication Review,* vol. 45, no. 5, pp. 5--12, 2015.

[4] A. Karaj, S. Macbeth, R. Berson and J. M. Pujol, "WhoTracks. Me: Shedding light on the opaque world of online tracking," *arXiv preprint arXiv:1804.08959,* 2018.

[5] S. Englehardt and A. Narayanan, "Online Tracking: A 1-Million-Site Measurement and Analysis," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Vienna, Austria, Association for Computing Machinery, 2016, p. 1388–1401.

[6] V. Rizzo, S. Traverso and M. Mellia, "Unveiling Web Fingerprinting in the Wild Via Code Mining and Machine Learning," 2020. [Online]. Available: https://www.pimcity-h2020.eu/publication/unveiling-web-fingerprinting-in-the-wild-via-code-mining-and-machine-learning/.

[7] S. Matic, C. Iordanou, G. Smaragdakis and N. Laoutaris, "Identifying Sensitive URLs at Web-Scale," 2020. [Online]. Available: https://laoutaris.info/wp-content/uploads/2020/09/imc20.pdf.

[8] P. Ohm, "Broken promises of privacy: Responding to the surprising failure of anonymization," *UCLA Law Review,* p. 1701, 2010.

[9] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *2008 IEEE Symposium on Security and Privacy*, 2008, pp. 111-125.

[10] P. Samarati, "Protecting respondents identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering,* vol. 13, no. 6, pp. 1010-1027, 2001.

[11] K. LeFevre, D. J. DeWitt and R. Ramakrishnan, "Incognito: Efficient Full-Domain K-Anonymity," in *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, Baltimore, Maryland, Association for Computing Machinery, 2005, p. 49–60.

[12] K. LeFevre, D. J. DeWitt and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *22nd International conference on data engineering (ICDE'06)*, IEEE, 2006.

[13] A. Machanavajjhala, D. Kifer, J. Gehrke and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data,* vol. 1, no. 1, pp. 3-es, 2007.

[14] 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.

[15] C. Dwork, F. McSherry, K. Nissim and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, Springer, 2006, pp. 265--284.

[16] N. Jha, T. Favale, L. Vassio, M. Trevisan and M. Mellia, "z-anonymity: Zero-Delay Anonymization for Data Streams," to appear in *2020 IEEE International Conference on Big Data (Big Data)*, 2020.