Ref. Ares(2024)5631017 - 04/08/2024

Scaling Up secure Processing, Anonymization and generation of Health Data for EU cross border collaborative research and Innovation

D4.2 — Architecture Specification Analysis and Design

Grant Agreement Nr. 10109571

Project Information

Project Title	Scaling Up Secure Processing, Anonymization and Generation of Health Data for EU Cross Border Collaborative Research and Innovation		
Project Acronym	SECURED	**Project No.**	10109571
Start Date	01 January 2023	**Project Duration**	36 months
Project Website	https://secured-project.eu/		

Project Partners

Num.	Partner Name	Short Name	Country
1 (C)	Universiteit van Amsterdam	UvA	NL
2	Erasmus Universitair Medisch Centrum Rotterdam	EMC	NL
3	Budapesti Muszaki Es Gazdasagtudomanyi Egyetem	BME	HU
4	ATOS Spain SA	ATOS	ES
5	NXP Semiconductors Belgium NV	NXP	BE
6	THALES SIX GTS France SAS	THALES	FR
7	Barcelona Supercomputing Center Centro Nacional De Supercomputacion	BSC CNS	ES
8	Fundacion Para La Investigacion Biomedica Hospital Infantil Universitario Nino Jesus	HNJ	ES
9	Katholieke Universiteit Leuven	KUL	BE
10	Erevnitiko Panepistimiako Institouto Systimaton Epikoinonion Kai Ypolgiston-emp	ICCS	EL
11	Athina-Erevnitiko Kentro Kainotomias Stis Technologies Tis Pliroforias, Ton Epikoinonion Kai Tis Gnosis	ISI	EL
12	University College Cork - National University of Ireland, Cork	UCC	IE
13	Università Degli Studi di Sassari	UNISS	IT
14	Semmelweis Egyetem	SEM	HU
15	Fundacio Institut De Recerca Contra La Leucemia Josep Carreras	JCLRI	ES
16	Catalink Limited	CTL	CY
17	Circular Economy Foundation	CEF	BE

Project Coordinator: Francesco Regazzoni - University of Amsterdam - Amsterdam, The Netherlands

Deliverable Information

Workpackage	WP4
Workpakace Leader	(CTL)
Deliverable No.	D4.2
Deliverable Title	Architecture Specification Analysis and Design
Lead Beneficiary	ISI
Type of Deliverable	Report
Dissemination Level	Public
Due Date	30/06/2024

Document Information

Delivery Date	30/06/2024
No. pages	99
Version | Status	1.2 | Final
Deliverable Leader	Apostolos Fournaris (ISI, Evangelos Haleplidis (ISI)
Internal Reviewer #1	Juan Carlos Pérez Baún (Atos)
Internal Reviewer #2	Alberto Gutierrez-Torre (BSC)

Quality Control

Approved by Internal Reviewer #1	30/07/2024
Approved by Internal Reviewer #2	30/07/2024
Approved by Workpackage Leader	31/07/2024
Approved by Quality Manager	31/07/2024
Approved by Project Coordinator	1/08/2024

List of Authors

Name(s)	Partner
Apostolos Fournaris, Evangelos Haleplidis	ISI
Francesco Regazzoni, Marco Brohet, Kyrian Maat, Georgios Tasopoulos	UvA
Alberto Gutierrez-Torre	BSC
Juan Carlos Perez Baun, Miryam Villegas Jimenez, Dario Ruiz Lopez	ATOS
Paolo Palmieri	UCC
Gareth T. Davies	NXP
Gergely Acs, Balazs Pejo	BME
Peter Pollner	SEM
Ioannis N. Tzortzis, Charalampos Zafeiropoulos, Nikolaos Bakalos, Dimitrios Kalogeras	ICCS
Christos Avgerinos, Christina Michailidou, Eleni Palousi	CTL
Dario Guidotti, Laura Pandolfo	UNSS
Alice Heliou, Vincent Thouvenot	TSG

The list of authors reflects the major contributors to the activity described in the document. The list of authors does not imply any claim of ownership on the Intellectual Properties described in this document. The authors and the publishers make no expressed or implied warranty of any kind and assume no responsibilities for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information contained in this document.

Revision History

Date	Ver.	Author(s)	Summary of main changes
01.05.2024	0.1	Apostolos Fournaris (ISI)	Created the document and the initial version of its content
10.05.2024	0.2	Apostolos Fournaris (ISI)	Updated document structure and included assignment
15.05.2024	0.3	Apostolos Fournaris, Evangelos Haleplidis (ISI)	Reference Architecture figure and description provided
20.05.2024	0.4	All technical partners & ISI team	Agreed and structured final component naming and finalised section 3 ToC
30.05.2024	0.5	All Technical partners	First draft section 3 input provided
10.06.2024	0.6	Apostolos Fournaris & Evangelos Haleplidis (ISI) (ISI)	user types and draft User Journey provided in Section 4
15.06.2024	0.7	All Technical Partners	first draft on Section 5 inputs (Tables) provided
25.06.2023	0.8	Apostolos Fournaris (ISI)	Revised Reference architecture provided
06.07.2024	0.81	All Technical partners	3rd draft version of document finalized based on Revised reference architecrure
24.07.2024	0.9	Apostolos Fournaris (ISI), Christis Avgerinos (CTL), Vassilis Paliouras (ISI), Konstantina Karagianni (ISI) & Francesco Regazzoni (UvA)	Reviewed and updated Section 5 Tables and deliverable overall context
25.07.2024	1.0	All Technical Partners	Final deliverable provided. REady for internal review.

Date	Ver.	Author(s)	Summary of main changes
28.07.2024	1.1	Juan Carlos Pérez Baún (ATOS) & Alberto Gutierrez-Torre (BSC)	Internal Review completed
30.07.2024	1.11	All Technical Partners	Internal Comments addressed
1.08.2024	1.2	Apostolos Fournaris (ISI), Christis Avgerinos (CTL) & Francesco Regazzoni (UvA)	Reviewed and Finalized Deliverable
2.08.2024	1.2	Francesco Regazzoni (UvA)	Deliverable submitted to EU

Table of Contents

Acronyms

ADT Anonymized Data Transformation. 46, 49, 51, 54, 61, 62, 68

AES Advanced Encryption Standard. 23

AI Artificial Intelligence. 33, 39–43, 57, 86

API Application Programming Interface. 20, 22, 23, 29–31, 36, 37, 44–46, 58, 64, 73, 77, 80–84, 86–88, 96

CI/CD Continuous Integration and Continuous Deployment solution. 24, 25

CJ Customer Journey. 55

CRUD Create, Read, Update, Delete. 23

CTG Cardiotocography. 36

DB Data Base. 22, 23

DICOM Digital Imaging and Communications in Medicine. 24, 81, 82

DL Deep Learning. 63–67, 81

DoA Description of Action. 11, 12, 93

EDPB European Data Protection Board. 35

EHR Electronic Health Record. 44, 46, 81

EU European Union. 35, 63, 86

FL Federated Learning. 33, 52, 56, 57, 67, 78, 79, 87

GA Grant Agreement. 11, 27, 56

GDPR General Data Protection Regulation. 14, 33, 35, 60, 62, 63, 68

GPU Graphics Processing Unit. 73, 74, 79

HE Homomorphic Encryption. 52, 54, 56, 57, 67, 79, 97

HTTPS Hypertext Transfer Protocol Secure. 23, 74, 77

IAM Identity and Access Management. 20, 21

JSON JavaScript Object Notation. 21, 28–30, 34, 80

JWT JSON Web Token. 21, 23

ML Machine Learning. 14, 33, 57, 63–67, 81, 88

MRI Magnetic Resonance Imaging. 36, 37

OWL Web Ontology Language. 72

PET Privacy-Enhancing Technology. 15, 56, 57

RBAC Role-Based Access Control. 15, 20, 23

ReST REpresentational State Transfer. 20, 45, 46, 77, 81, 83–86, 96

SDG Synthetic Data Generator. 36–38, 51, 58, 59, 61, 64, 73–75, 78, 86

SMPC Secure Multi-Party Computation. 52–54, 56, 57, 67, 79, 97

SQL Structured Query Language. 72

SSL Secure Sockets Layer. 70–76

SWRL Semantic Web Rule Language. 25

TLS Transport Layer Security. 78, 79

UI User Interface. 15, 36

UJ User Journey. 5, 7, 11, 13, 55–58, 62–69

UJM User Journey Mapping. 12, 13, 20, 55, 56

UX User Experience. 15

WP Work Package. 12

1 Executive Summary

This document fulfills the goals of task T4.1 and outlines a detailed SECURED architecture, the interfaces between its components and its communications network architecture and the main characteristics of components' interfaces to be used as input in WP2 and WP3 for the application library development as well as WP4 where the SECURED infrastructure is built and the integration takes place.

The SECURED architecture consists of all the necessary components concerning scaling-up secure multi-party computation, privacy-preserving and robust federated learning, unbiased AI, private synthetic-data generation, as well as data anonymisation and anonymisation assessment (de-anonymisation or re-identification), amongst others, all of them will be interconnected through standard open interfaces.

The document delivers the following:

1. It details the SECURED architecture with all their components.
2. It identifies SECURED user types and maps their User Journey on the SECURED system
3. It documents all technical specifications per SECURED Architecture Component
4. It documents all Inputs, Outputs, Interfaces and Interactions per Component
5. It provides updates on SECURED Technical Requirements (in relation to the preliminary technical requirements of D4.1[1]) and documents new ones

1.1 Related Documents

- Grant Agreement (GA) Project 101095717 - SECURED; Description of Action (DoA) Annex 1
- Deliverable 4.1 "State of the Art and initial technical requirements"

2 Introduction

2.1 Purpose and Scope of the Document

This document is extending the preliminary work on the SECURED Reference Architecture and technical requirements that has been done in Deliverable 4.1. Thus, the Deliverable provides the final SECURED Reference Architecture that can accommodate the whole SECURED functionality as this has been described in the project Description of Action (DoA). Within the document we aim to offer details on each one of the Reference Architecture components and in some cases details on the internal structure of each component. We also provide a graphical view of the Reference Architecture that showcases all the component interactions in an abstract manner. By specifying the Reference Architecture and its building blocks, the SECURED consortium aims to fully and comprehensively map the various SECURED functionalities as those where already abstractly presented in the preliminary version of the deliverable (i.e., D4.1[1]). Thus, this Deliverable, D4.2, acts as a reference point of the current and upcoming project activities related to the implementation of the SECURED system including the SECURED Innohub and its capabilities.

In order to accomplish the aforementioned goals and also properly map all interactions of the SECURED Reference Architecture with its users, in this deliverable, we use the User Journey Mapping (UJM) approach in order to a) identify the various user types and b) specify the way each user type interacts with the SECURED system and what internal actions within this system are triggered due to these interactions. This information will dictate the overall SECURED solution implementation and the integration process of the various components into the realistic implementation SECURED solution that is going to be realized in the T4.4 of WP4 as well as the SECURED prototype implementation that will be utilized for the SECURED Open Call.

Apart from the above, it is within the scope of this document to provide the SECURED integrator and technical partners with as detailed as possible technical specifications for each one of the SECURED components, and to technical determine the component-to-component interconnection specifications. In an effort to provide such details, in this deliverable, we use the collected technical requirements from D4.1[1] and also introduce additional ones and assign them as requirements of each one of the SECURED components. This per-component list of technical specifications is complemented by another important SECURED Reference Architecture characteristic, the interaction from one component to the other. These types of specification provide a thorough insight on the interfaces that each component have and the types of data such interfaces use to communicate with other SECURED components. The above collection of described specifications provide detailed guidelines for the next design iteration of the SECURED solution that will be focused on the implementation and prototype development to be performed in the other tasks of WP4. It should be noted that the provided specifications and interactions provide a clear picture of the current status of the SECURED solution development but as the actual SECURED platform, Innohub, tools and services get integrated together they may be further refined and updated.

2.2 Contribution to WP4 and Relation to Work Packages, Deliverables, and Activities

The Deliverable constitutes the final outcome of T4.1 - State-of-the-Art, Technical Specifications & Architecture Design of WP4 and provide the backbone of requirements and specifications of the SECURED architecture for all WP activities of the project till M18. It also provides insight for the development and integration activities that are taking place or will take place in WPs 2, 3 and 4. The Deliverable extends the work presented in D4.1[1]. It realizes the WP4 objectives of a) Providing State of the Art in scale-up SMPC, Anonymisation, De-anonynimization and Synthetic data Generation and b) Providing SECURED architecture and technical requirements as those exist in the DoA. The Deliverable also strategically contributes to the design and implementation of the overall SECURED Innohub objective in WP4. Apart from WP4, the deliverable also relates to the WP2 and WP3 activities where the components described in it are been developed as well as the WP5 activities where the user requirements are specified, in Deliverable D5.1.

2.3 Structure of the Document

The document consists of 5 sections and 2 appendices. Apart from the introduction Section (Section 2 there are four technical section that document all aspects of the deliverable. In Section 3 the final SECURED Reference Architecture is presented and its six different domains are described shortly. Furthermore, in this section we provide an thorough description of each one of the components that appear in the six domains including figures, when needed, detailing the components' individual functionality and execution flow. In Section 4 we describe the User Journey Mapping methodology that we have used in order to identify SECURED user types and describe their interactions with the SECURED platform and the Innohub services and tools as well as the development libraries that are developed in WP2 and WP3. Having identified the final SECURED components and through the various UJ of Section 4 we manage to reevaluate the technical requirements of the deliverable D4.1[1] where the preliminary SECURED Reference Architecture is presented and we are also able to provide detailed technical specifications for each component and specifications on each component interactions and interfaces. These specifications are described per component in Section 5. In Section 6 the conclusions of the Deliverable are presented. Finally, the updated, revised technical requirements (originally presented in D4.1[1]) for the SECURED architecture and its components are briefly documented in Appendix A of the deliverable while in Appendix B we present new technical requirements that are identified by M18 of the project

3 SECURED Reference Architecture

3.1 Overall Architecture

Figure 1 provides a schematic of the SECURED Reference Architecture. As shown, the architecture is split into six domains, the Front End, the Back End, the Knowledge Base, the Innohub Services, the Innohub tools and the Innohub Software/Hardware Development libraries.

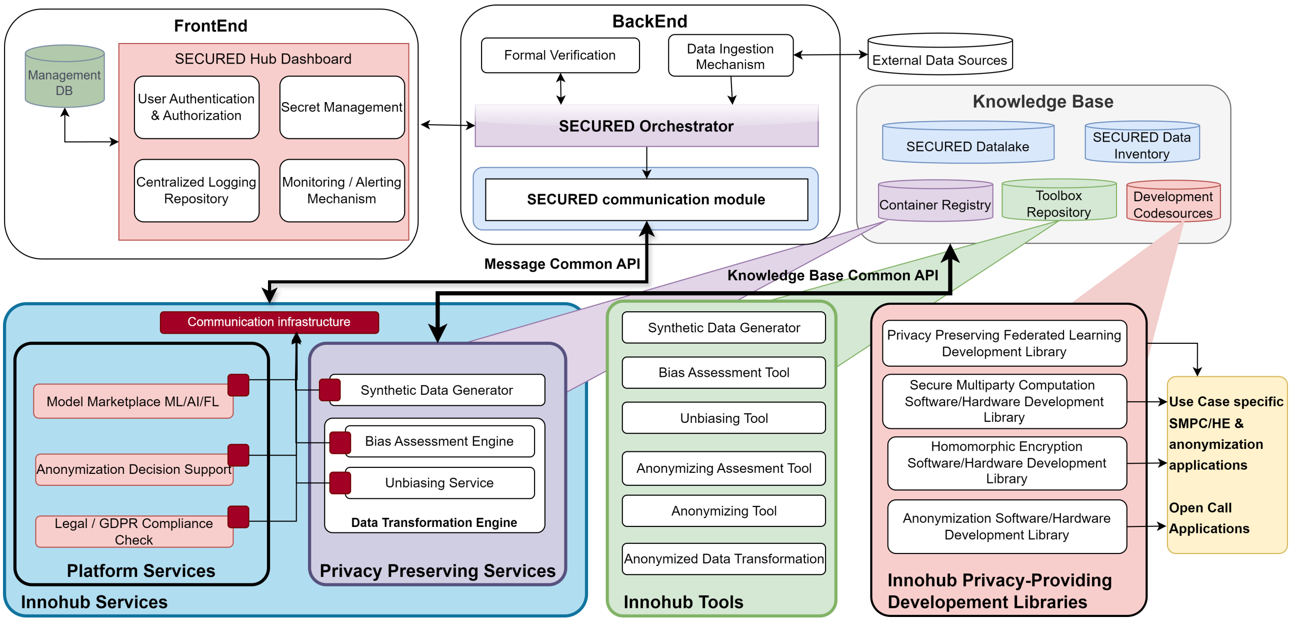


Figure 1 – SECURED Architecture

The **Front End** domain provides all the necessary SECURED components and facilitates the necessary interfacing between the users and the rest of the SECURED system, containing features such as user authentication, secret management and logging.

The **Back End** domain provides the core infrastructure of the whole solution, supporting various critical functionalities such as connectivity of the architectural components, orchestration of the SECURED operations and the overall SECURED functionality, data ingestion mechanisms from external data sources and a formal verification mechanism for the whole SECURED solution.

The **Knowledge Base** domain acts as the central repository of knowledge for the SECURED framework. The Knowledge Base contains the necessary data structures of the overall SECURED solution such as a container registry, toolbox repository, codesource repository and Data Inventory as well as the SECURED Data Lake which contains Synthetic Data, SECURED generated trained AI models, series of legal documents and in general any data that may be used, generated or handled by SECURED.

The **Innohub Services** domain provides all the mandatory functionalities as services to support the project objectives and is split into two subdomains. The first subdomain is the **Platform Services** which include all the project-wise services of the SECURED Innohub, providing an ML model marketplace, an anonynimisation decision support mechanism and a Legal/GDPR compliance check service. The second subdomain contains all the **Privacy Preserving Services** which include the synthetic data generation service, and the data transformation service of the SECURED Innohub that offers bias assessment and unbiasing through the bias assessment service and the unbiasing engine service. The above services can also be offered as downloadable tools to accommodate users that request offline usage of the SECURED Innohub capabilities.

The **Innohub tools** domain contains all the downloadable tools that are offered by the SECURED Innohub to be run on the user's premises. The provided tools include variations of the provided SECURED Innohub Services like the bias assessment tool, unbiasing tool and the synthetic data generator tool but also the Innohub toolset for anonymisation, more specifically, the anonymisation tool, the annonymisation assessment tool and the anonymised data transformation toolset. The later constitute an advanced toolset that allows the collaboration of several of the provided tools, as described in Section 3.5.2.6, in order to provide to the users a "properly" anonymised and as a standalone tool. By "properly" anonymised we refer to a dataset that has low or no bias, that has been anonymised with techniques and methodologies that provide an anonymisation result that is not susceptible to currently known de-anonymisation attacks.

Finally, the SECURED architecture features the **Innohub Privacy-Providing Development Libraries** domain that includes a large number of development libraries that can be provided to the Innohub users to develop their own Privacy-Enhancing Technologies (PETs) developed applications. The included development libraries can be characterized as privacy preserving federated learning development libraries, secure multiparty computation libraries, Homomorphic Encryption libraries and Anonymisation Software/Hardware Development libraries. The development libraries integrate several scalability enhancement mechanisms that rely on algorithmic and/or implementation enhancement using software or hardware means.

The rest of this section provides a detailed description of each of the components residing in each domain of the SECURED Architecture.

3.2 SECURED Front End

The front-end of the SECURED Innohub is a user-friendly digital platform, which is designed to empower users with the developed data privacy and security tools. This platform serves as a centralized gateway where users can explore, download, and experiment with various tools that address contemporary challenges in data protection. Key features of the SECURED Innohub front-end include:

1. **Role-Based Access Control (RBAC)**: To ensure secure access, the platform will utilize RBAC, which authorizes users based on their assigned role. This approach restricts access to tools, functionalities and/or datasets according to predefined roles (i.e., Administrator, User, Guest, etc.), thereby maintaining a controlled environment.
2. **Logging**: Comprehensive logging mechanisms are integrated into the platform to record all user activities and system events.
3. **Monitoring and Alerting System**: The platform includes a robust monitoring and alerting system that continuously tracks system performance and security events.

Through these features, the SECURED Innohub platform not only facilitates user engagement with innovative tools, but also ensures a secure, efficient, and reliable user experience. The following sections provide a detailed description of the aforementioned features and a set of mockups which visualize how we envision the platform.

3.2.1 SECURED Hub Dashboard

The main goal while developing the front-end of the SECURED Innohub is to follow all UI/UX principles in order to create an intuitive and user-friendly interface. By focusing both on the functionality and the design of an easy to navigate dashboard, we want to ensure that users can effortlessly engage with the tools and services without sacrificing the user experience. Following, a set of mockup pages is provided which depict the overall look and feel of the dashboard.

Login/ Registration Page

The Login page (Figure 2) is the entry point for users to access the SECURED Innohub. It features fields for entering an email and password, along with options for users who have forgotten their password or need to register for an account. The page emphasizes ease of access with a clear call to action to sign in. On the other hand, the Registration page (Figure 3) allows new users to create an account. It collects basic information such as first name, last name, email, company/organization, and password. Users must agree to the terms and conditions before registering.

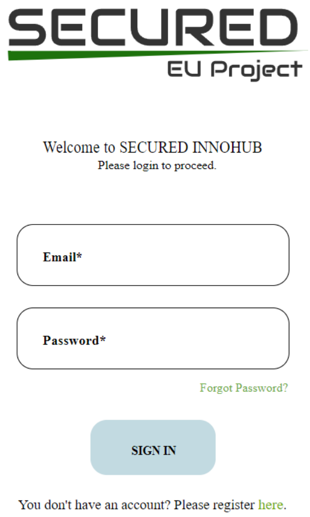


Figure 2 – The SECURED Login Page.

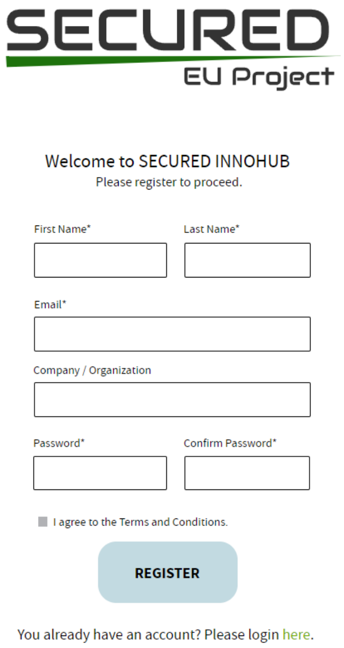


Figure 3 – The SECURED Registration Page.

Home Page

The home page (Figures 4 and 5) provides a comprehensive overview of the SECURED Innohub and its offerings. It includes navigation links to different sections such as Tools, Services, Knowledge Base, and the SECURED Project. The page highlights key areas of research, available tools, and services with brief descriptions and "Read more" links for additional information. Moreover users can find also useful tutorial videos on how to use the platform.

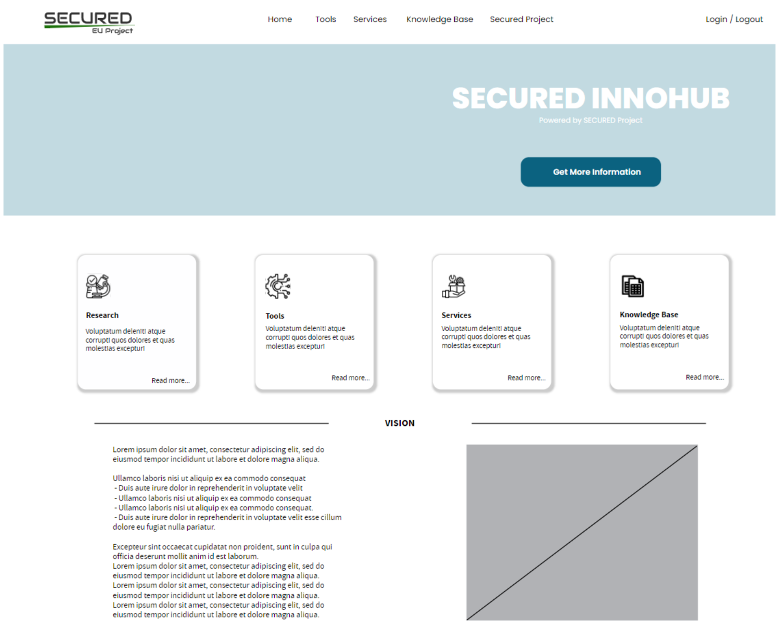


Figure 4 – The SECURED Home Page (1/2).

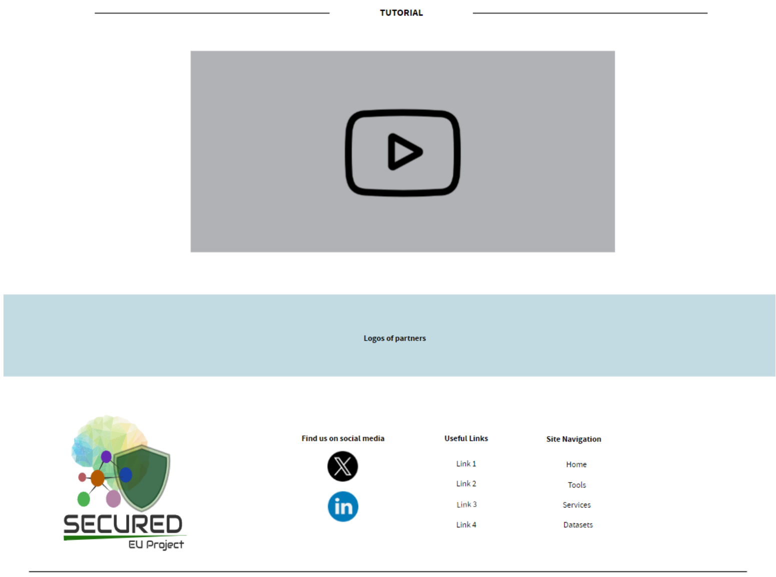


Figure 5 – The SECURED Home Page (2/2).

Tools/Services Pages

The Tools page (Figure 6) lists the various tools available within the SECURED Innohub. A similar page (Figure 7) is also provided for the services which are available within the Innohub. Each tool/service is briefly described (Figure 8), and users can read more about it or proceed to use them.

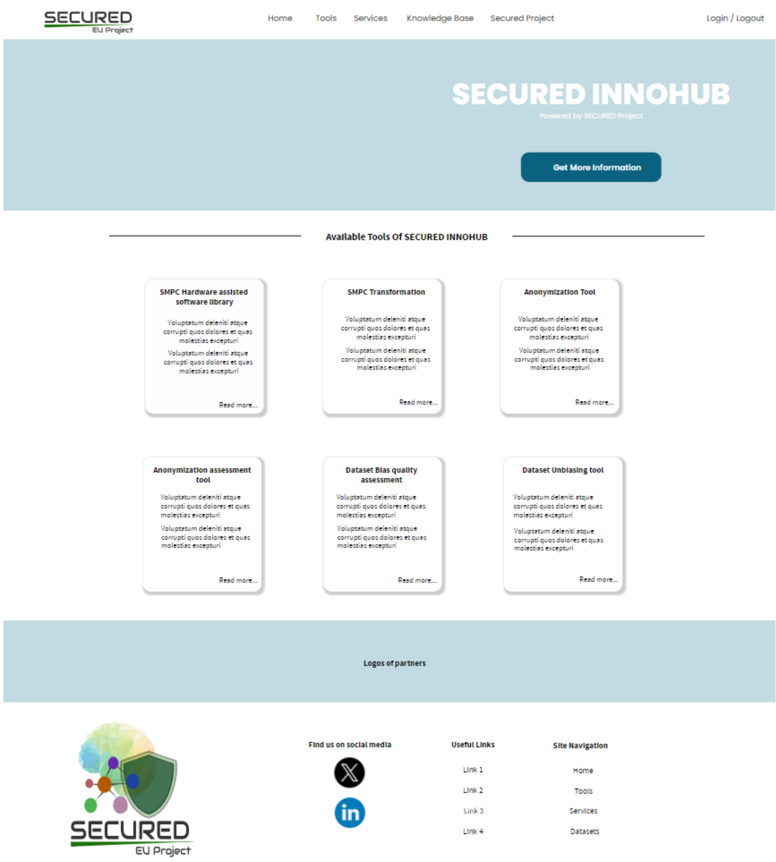


Figure 6 – The SECURED Tools Page.

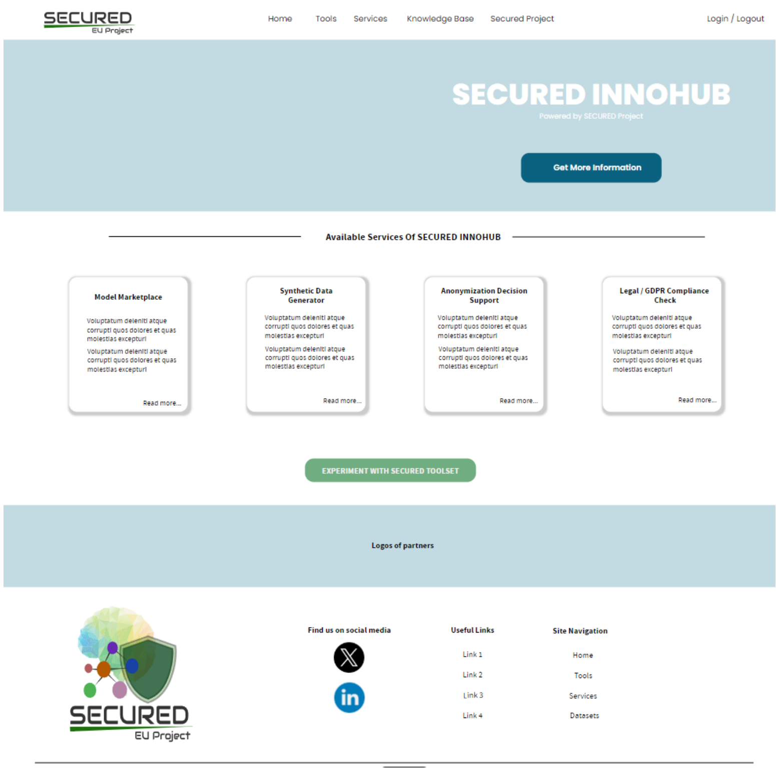
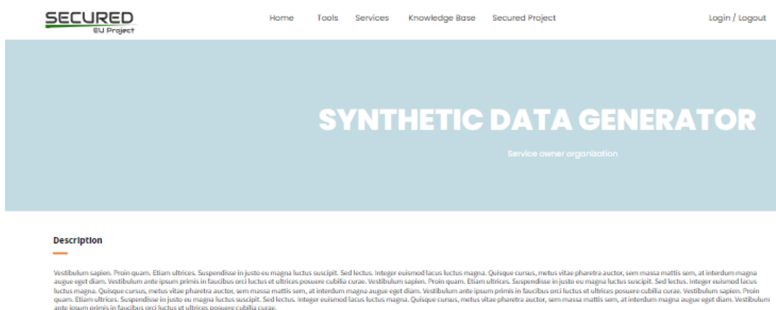


Figure 7 – The SECURED Services Page.



Pipeline Triggering

The pipeline page (Figure 9) allows users to experiment with the entire suite of SECURED tools. Users can either upload their datasets or choose from existing ones to trigger the tool pipeline. This page provides a seamless interface for conducting comprehensive data privacy and security experiments.

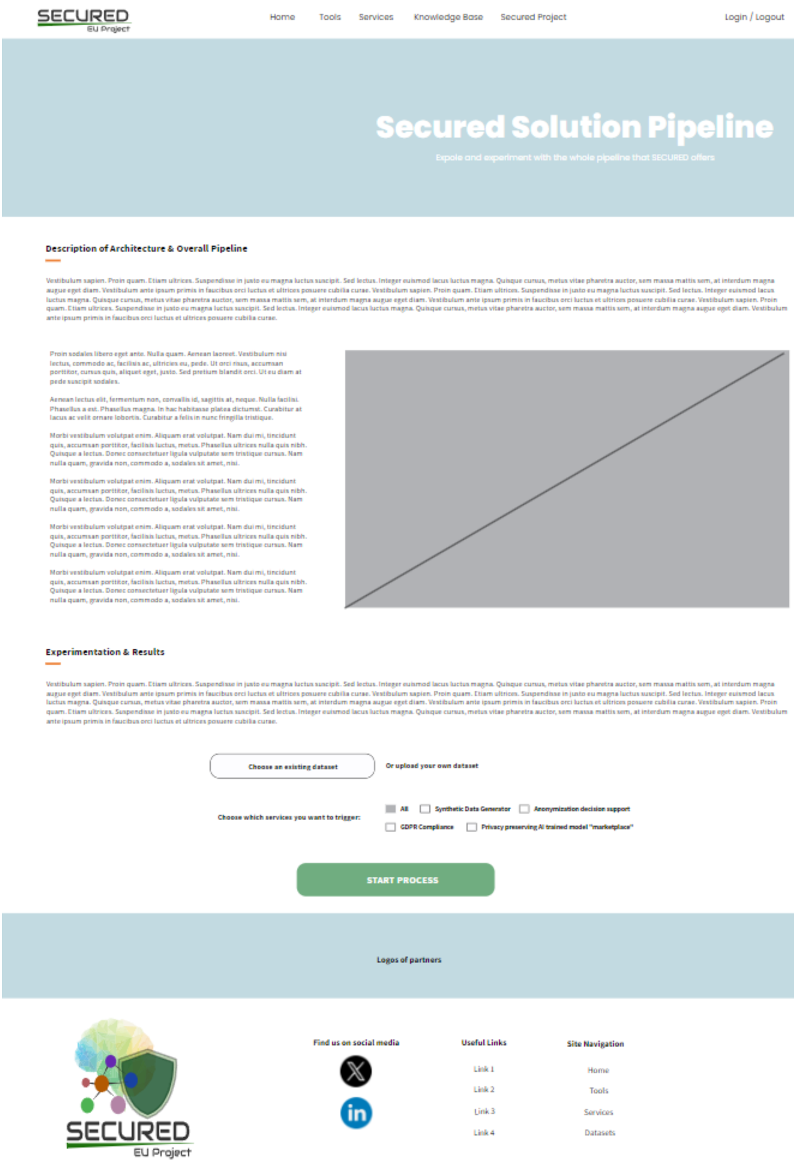


Figure 9 – The Pipeline Triggering Page.

Each page is designed to guide users through the process of exploring, understanding, and utilizing the tools developed under the SECURED project, ensuring a seamless and efficient user experience.

3.2.2 User Authentication & Authorization

The Identity and Access Management (IAM) is the component responsible for the user account management lifecycle of the SECURED platform and for controlling the access to the different resources of the platform. IAM undertakes the operations related to the registration, verification and authentication of all the users of the platform. The main functionality of IAM is to provide the security engine that implements the authentication, authorisation and role management functionalities of the SECURED platform. IAM follows the main principles defined by the RBAC paradigm. An RBAC model defines an access control mechanism in which access rights are granted to users based on their role within an ecosystem. In RBAC, permissions are assigned to specific roles rather than to individual users. Users are following assigned roles, and through these roles, they acquire the permissions to perform the relevant tasks. RBAC enables a structured and efficient way to handle access control, especially in environments with many users and a large number of permissions. Roles can be designed to reflect the organizational hierarchy and business functions, such as Administrator, Manager, User, and Guest. By assigning users to roles and managing roles independently of the users, RBAC provides a scalable and manageable approach to access control. To this end, the distinct roles which are currently available in the SECURED platform are as follows:

- Admin
- Tool owner
- Service owner
- User
- Guest

It must be noted that these roles are associated to access rights on the SECURED Innohub and are considered from a user access point of view instead of the User Types defined in Section 4.2 that depicts functional usage of the platform by its users/UJM personas.

Basic Principles

The IAM is capable of controlling the access to the various data/tools/services of the SECURED platform based on the role of requestor (user), as seen in Figure 10. The data are exposed to the end users in the form of ReST APIs. Each endpoint exposed by the platform should be added in a list of endpoints, describing thus the whole data platform functionalities. The access control mechanism is applied by assigning the roles that are granted access per endpoint. For this purpose, the list of roles that can have access to each endpoint, should be defined. It should be noted that multiple roles can be assigned to a single endpoint (N to 1 relationship). Hence, IAM is maintaining the access control information in the following form:

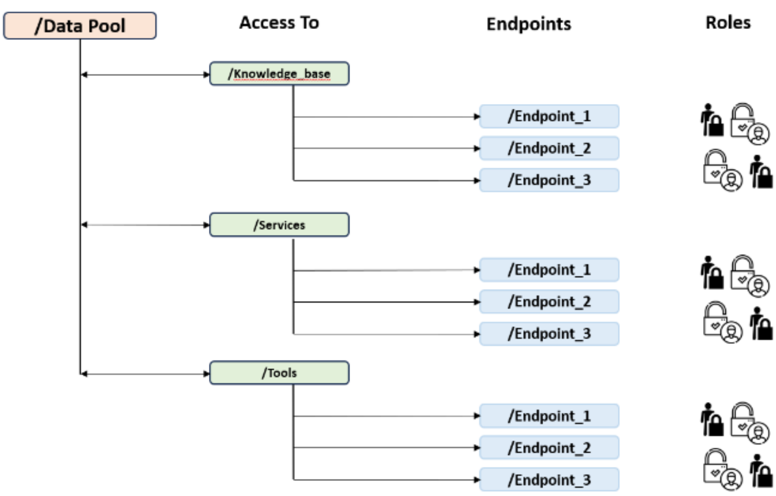


Figure 10 – Access to endpoints based on given role.

The core element of the authorisation, authentication and access approval, is a token-based mechanism, which is based on JSON Web Token (JWT). JWT is an open standard (RFC 7519) ([2]) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. JWT is considered a dominant solution for authentication, authorisation and secure exchange of information. Upon successful login, a valid token is generated, in which the access level is defined by incorporating the username and the role that is assigned to this user within the token. Any subsequent request should contain this generated token. The endpoint receiving a request should consult IAM with the requestor's token for validation and access control decision.

Basic Workflows

In order to access any resource of the platform (i.e., tools, services, datasets, etc.), users should have successfully logged in first (Figure 11), by providing their credentials to IAM which as mentioned previously undertakes the task of the authentication. The first basic workflow which is provided by the SECURED platform follows a streamlined, single sign-in process for authenticating users. This process includes the following steps.

1. **Initiating the Login Request**: Users submit their login credentials to the IAM system via the designated login endpoint.

2. **Credential Verification and Token Issuance**: Upon successful verification of the provided credentials, IAM issues an access token in JWT format and a refresh token. These tokens are included in the response header. The IAM system will respond with a Status Code 200 OK for a successful login or 401 Unauthorized if the authentication fails.

3. **Access Token Usage**: The access token must be included in the header of every subsequent request to access the platform's resources.

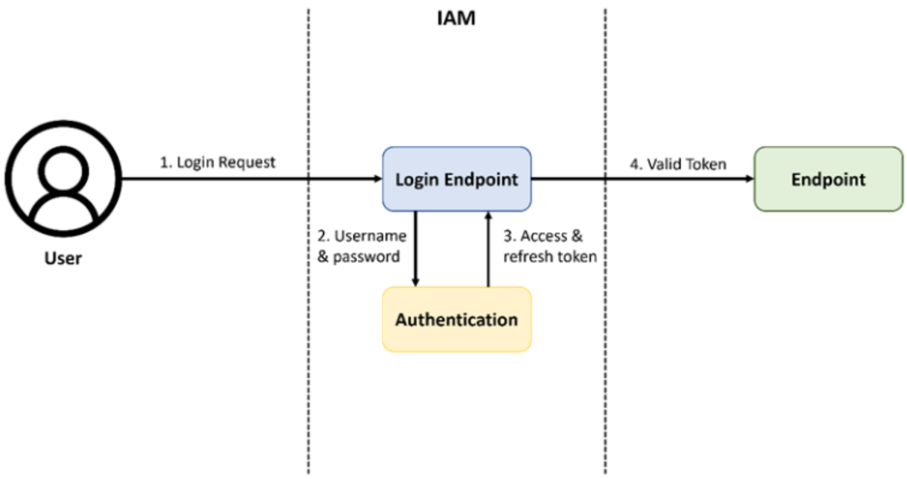


Figure 11 – The Login Process.

The second workflow (Figure 12) is related to any subsequent request, after the successful login, to access any available resource of the platform. This workflow consists of the following steps:

1. When the user initiates a request to access an endpoint, the endpoint should consult IAM, in order to validate the request and make an access control decision.

2. The endpoint should initiate a call to the access verification endpoint of IAM, providing the token in the Header of the request.

3. IAM will respond with either a Status Code 200 OK or a Status Code 403 Forbidden, if access is granted or denied, respectively.

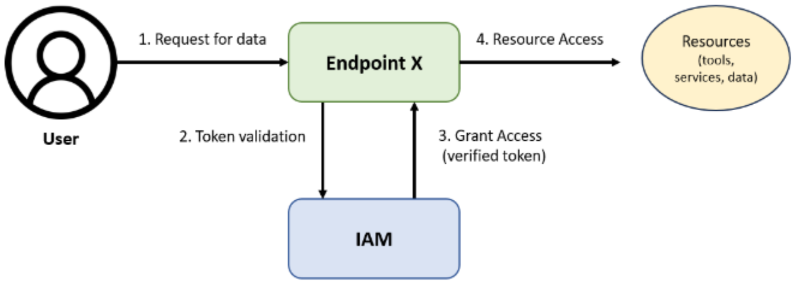


Figure 12 – Resource Access.

This robust authentication mechanism ensures secure and efficient access to the platform, safeguarding data integrity while providing a seamless user experience.

3.2.3 Logging and Monitoring/Alerting Mechanisms

The aim of implementing a centralized logging repository (Figure 13 is to collect, store, and manage in an efficient way the logs coming from all the interactions that a user will have with the SECURED Innohub. More specifically, this logging repository will include logs coming from the three main actors of the Innohub, which are the Management DB, the API which will facilitate the communication among the Management DB and the Front-end, and the Front-end itself. For the database, logs should include queries executed, changes to data, authentication attempts, and access patterns to identify potential misuse or anomalies. The API should log incoming requests, response times, error rates, and authentication attempts to track the performance and detect unauthorized access or abuse. The dashboard should log user activities, changes in configurations, and access attempts to monitor for suspicious behavior and ensure that only authorized users are making changes. The following schema, depicted in Figure 13, illustrates the SECURED Innohub centralized logging system.

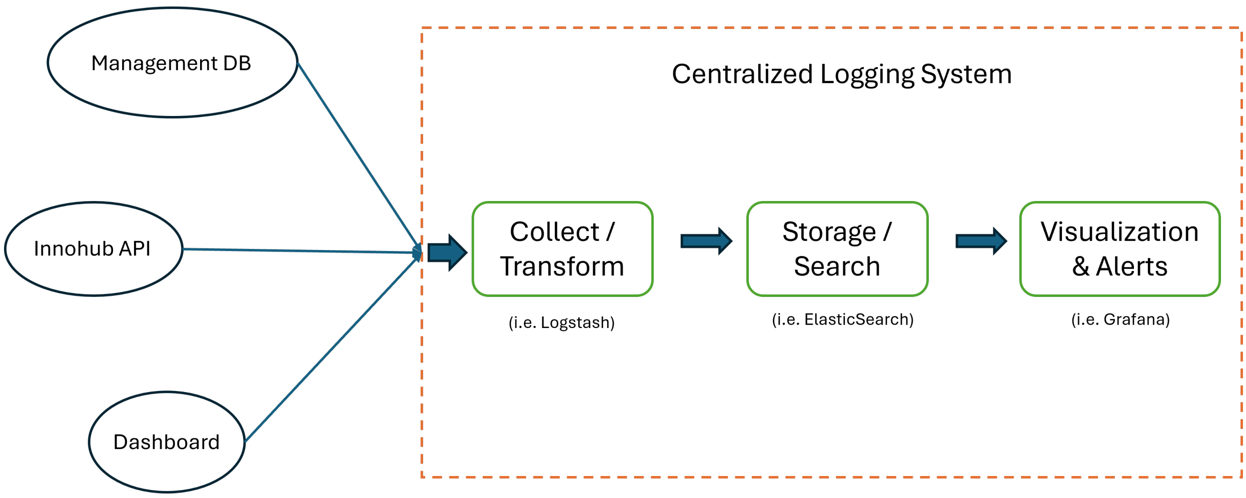


Figure 13 – The SECURED Centralized Logging System.

In essence, the several components that comprise the SECURED Innohub ecosystem will streamline their logs to a centralized infrastructure where a three-stage process will take place. The first stage will be responsible for aggregating logs from various sources and transforming them into a consistent format for easier processing and analysis. The second step involves storing the transformed log data in a scalable and searchable repository while the third and final step focuses on presenting the log data in a visual format that is easy to interpret and analyze. Visualization tools offer dashboards and charts that help users monitor system performance and identify issues in real-time. Moreover, alerts can be configured to notify administrators of critical events such as failed login attempts, unusual spikes in API traffic, significant changes in database queries, and unauthorized access attempts.

3.2.4 Secret Management

Secret management is the practice of handling sensitive information—referred to as "secrets"—in a secure manner throughout their lifecycle. Secrets can include passwords, API keys, tokens, encryption keys, and certificates that are essential for authenticating and securing communications between systems and applications. Effective secret management involves storing secrets in encrypted formats, controlling and monitoring access to these secrets, and ensuring they are regularly updated to mitigate the risk of unauthorized access. Encryption ensures that even if secrets are intercepted, they cannot be read without the corresponding decryption keys. Implementing RBAC ensures that only authorized users and systems can access specific secrets, thereby minimizing potential attack vectors. Regular audits and monitoring further enhance the security posture by tracking access patterns and identifying anomalies. Within the SECURED Innohub our approach involves the following security practices:

1. **Password Storage**: User passwords will be hashed using a hashing algorithm such as bcrypt. This ensures that even if the database is accessed, the passwords are not stored in a reversible format and are protected.
2. **Management DB Encryption**: To further protect content, an encryption mechanism (i.e., Advanced Encryption Standard (AES)) will be placed on top of password hashing, and possibly other sensitive data, stored in the Management DB. This provides an additional layer of security, ensuring that all data, including hashed passwords, remain encrypted at rest.
3. **Regular Token Updates**: As described in Section 3.2.2, the access to the SECURED Innohub is facilitated through the usage of JWT tokens. Those tokens will have a short period of life (i.e., 24 hours), forcing thus the users to re-login to the Innohub.
4. **Secure Communication**: All data transmissions between the Innohub and the server are secured using HTTPS. This ensures that the data, including sensitive information like passwords and tokens, is encrypted in transit and protected against eavesdropping and man-in-the-middle attacks.

3.2.5 Management DB

The Management DB serves as the backbone for the dashboard in our system, providing essential storage and retrieval functionalities for various critical components. Primarily, it is designed to manage user information, securely store secrets, and track user sessions. The database schema includes tables for user profiles, authentication details, encrypted secrets, and session records. Each user profile contains necessary personal and RBAC information, ensuring that permissions are appropriately managed and enforced. Encrypted secrets, such as passwords, API keys, and other sensitive data, are stored using strong encryption algorithms to maintain their confidentiality and integrity. Session tracking includes storing tokens and session metadata to monitor and manage active sessions, enhancing both security and user experience by allowing administrators to detect and handle anomalies like multiple simultaneous logins or expired sessions. To facilitate seamless communication between the Management DB and the dashboard, we employ FastAPI[1] as the intermediary API layer. FastAPI serves as the conduit through which the dashboard interacts with the database, providing a robust, high-performance framework that supports asynchronous operations and rapid response times. Through well-defined API endpoints, FastAPI handles user authentication, authorization, and session management processes. For example, when a user logs in, FastAPI validates their credentials against the stored hashes in the database, generates a session token upon successful authentication, and records the session details. Additionally, the API layer manages Create, Read, Update, Delete (CRUD) operations for user profiles and secrets, ensuring that all interactions with the database are secure and efficient. This setup not only guarantees the secure handling of sensitive data, but also promotes a scalable and maintainable architecture, capable of accommodating future expansions and increased user demands.

[1] https://fastapi.tiangolo.com/

3.3 SECURED Back End

The SECURED Back End architectural block encompasses all underlying methods, technologies and well-known third-party software solutions, assembled as an efficient project integration core. The SECURED Back End is responsible for handling data ingestion, acting as the middleman between raw data and the SECURED Knowledge Base, bringing input information to a certain level of serialization and data health. The formal verification of the SECURED components, a prime operation that ensures the continuous and unrestrained compliance of the developed components to the specific SECURED guidelines, also happens in the project's Back End. Furthermore, the overall orchestration of sequential or parallel execution of on-demand services, as well as the communication between them, originates from this block.

3.3.1 Data Ingestion Mechanism

The Data Ingestion Mechanism is a critical component within the SECURED Federation Infrastructure, ensuring accurate and up-to-date data ingestion from an abundance of diverse external data sources to the Data Warehouse and the Knowledge Base of the project. Additionally, the Data Ingestion module includes a variety of data transformation functions so that imported data are standardized, matching the data scopes of the SECURED Knowledge Base, and also, brought to a certain level of quality and homogenization. In order to guarantee the smooth and efficient flow of data into the SECURED ecosystem, the Data Ingestion mechanism is equipped with a plethora of data handling means. The module deals with non-structured data, such as DICOM images, .jpeg files, and documents, i.e data types that do not follow a specific structure or schema, making their ingestion particularly challenging. To tackle this, advanced tools such as custom data loaders and crawlers will be developed and integrated into the system. The module also manages semi-structured data, such as .csv, .json and .xml files. Although these loosely typed data formats contain some organizational properties, they do not conform to strict schemas. The data ingestion module must employ robust validators to accurately interpret these semi-structured data, ensuring consistency and reliability during the ingestion process. Finally and mostly expected, the module is capable of handling fully structured data such as large medical datasets, which come with predefined schemas, strong structures and even custom data loaders. These datasets are typically well-organized, allowing for straightforward mapping and integration into the data repositories. The module must ensure that the integrity and relationships within these structured datasets are preserved during the transfer. Instead of infusing raw data into the SECURED ecosystem, the Data Ingestion mechanism will apply transformation functions including but not limited to, cleaning, enriching and indexing of the input. Cleaning of data ensures that irrelevant or duplicate data is removed, missing values are handled and some normalization, such as transformation of all text to lowercase. Enrichment of data adds context to the imported data such as timestamp, geo-tagging, source or anonymisation level. Finally, software solutions like Elasticsearch may be considered for indexing text-heavy data and logging.

3.3.2 SECURED Orchestrator

The SECURED Orchestrator is the module responsible for developing and managing a robust Continuous Integration and Continuous Deployment solution (CI/CD), thus enhancing the project's overall development lifecycle. The inherit multi-party nature of the SECURED project dictates for a robust, plug-and-play integration of components which should maximize the capabilities of each tool or service in a highly efficient way. Shipping code in Docker containers has been the industry standard for portable and lightweight software packaging during the last decade. The dockerization of components is crucial because it ensures consistency across different development, testing, and production environments. By packaging applications and their dependencies into lightweight, portable docker containers, the SECURED integration scheme eliminates the "it works on my machine" problem, enabling all developed tools and services to work in a stable, portable and lightweight environment regardless of the responsible partner's local setup. This software encapsulation eventually leads to more reliable and reproducible builds and error tracing, simplifies the CI/CD pipeline, and accelerates the deployment process. Additionally, the dockerization of components enables efficient and scalable integration, allowing all developed components of SECURED to adapt quickly to potentially alternating demands during all stages of the overall integration. Finally, through the Docker Compose functionality, the integration overseer is able to organize the overall communication, co-operation and most importantly, selective isolation of the different containers, networks, and consistent data volumes.

Month 18 of the SECURED timeline marks the initiation of the overall integration, and up to this point, some preliminary, lightweight containers and communication configurations have been set up. Additionally, the core components of the CI/CD procedure have been established, whereas the automation rules and regulations are being investigated. Another very important aspect of the project integration is a centralized registry for keeping the Innohub services' Docker images, and for that matter, the SECURED Container Registry has been set up, available to host all partners' containerized software. Lastly, the potential incorporation of additional orchestration technologies such as Kubernetes for the overview and synchronization of microservices is currently being explored.

3.3.3 SECURED communication module

The SECURED Communication Module ensures secure, reliable and efficient exchange of messages and information among different parts of the project infrastructure. The central component of the module, Kafka, offers a distributed messaging system. The fundamental unit of organization, Kafka topics, divide data streams to enable cascading data pipelines. Producers write data to specific threads, while consumers read from it, enabling decoupled and scalable communication. The module ensures data integrity through end-to-end encryption, strong authentication and authorization mechanisms. Kafka's replication protocol and configurable retention policies guarantee fault tolerance and historical data analysis.

Integrated with the SECURED Orchestrator, the Communication Module plays a critical role in enabling inter-service communication and guaranteeing efficient information transfer between system elements. Through real-time monitoring tools, the module ensures the health and performance of communication processes, allowing for proactive issue detection and resolution. This integration greatly improves the overall project infrastructure by improving resilience, scalability and security, fostering smooth and reliable communication across all components.

3.3.4 Formal Verification

In the activity of formal verification, UNSS is developing two modules: an ontology-based module designed to ensure the consistency and integrity of the SECURED framework, and a module designed to evaluate the accuracy of synthetically generated data compared to real data. The first solution employs an ontology-based approach to model the key components of the architecture, identify potential threats, and verify system consistency. The ontology is divided into three main parts: the system sub-ontology, the data flows sub-ontology and the threat sub-ontology. The ontological model is populated with detailed data representing all components, libraries, relationships, and potential threats within the SECURED framework. This includes data flow modeling to capture how data moves through the system, enabling a comprehensive verification of data access, privacy compliance, and authorization. The module leverages the Semantic Web Rule Language (SWRL) to define logical relationships and constraints that ensure system consistency and identify threats. An example rule includes the following:

$$User(?p) \wedge HealthData(?d) \wedge hasRole(?p, ?r) \wedge AuthorizedRole(?r, ?d) \implies hasAccess(?p, ?d)$$

which ensures that a user can access specific datasets only if they have an authorized role for that data. A reasoner, such as Pellet[3] or HermiT[4], is used to apply the SWRL rules to the populated ontology. The reasoner performs automated reasoning to verify system consistency and identify threats based on the defined rules. Throughout the development process, the tool undergoes rigorous testing and validation using real-world scenarios and data sets. This ensures the accuracy and effectiveness of the SWRL rules and the reasoning process. The tool generates reports based on the reasoner's output, highlighting areas where the system's consistency is maintained and identifying any inconsistencies or threats. These reports are essential for identifying issues and improving the system configuration.

The second solution is the synthetic data validation module (SynthVal)[2], which is designed to evaluate the accuracy of synthetically generated data compared to real data. This module plays a critical role in ensuring the reliability of synthetic data, particularly in safety-critical domains such as medical education and training. SynthVal requires two main inputs: a set of synthetically generated data formatted consistently with the real dataset for meaningful comparison, and a corresponding set of real data, which serves as the benchmark for evaluating synthetic data quality. The module produces a series of measurements that provide insights into the fidelity of synthetically generated data. It also aims to identify specific data or features that contribute to lower fidelity whenever possible. Currently, the primary focus is on evaluating medical images contained within open datasets. One of the main challenges in evaluating these datasets lies in their high dimensionality. Therefore, current efforts are directed towards optimising methods to extract vectorial features from these images, which will facilitate more effective analysis.

[2]Since SynthVal has a somewhat different functionality than the formal verification ontology based module, we provide separate tables for each one of the formal verification modules with their technical specifications and interactions in section 5

3.4 Knowledge Base

In Figure 14, the architecture of the SECURED Knowledge Base schema is presented, as derived from the Task 4.3 description and the SECURED GA, along with the WP4 iterations that introduced corrections and essential refinements to specific components and connections. In general, the Knowledge Base includes the following components:

- the container registry for the SECURED applications/docker containers/docker container images,
- the legal documents repository for retaining the private legal documents referring to datasets,
- the toolbox repository for storing ready-to-use software tools,
- the SECURED Data Inventory aiming to store metadata for existing datasets,
- the Privacy Preserving AI component for storing and indexing the developed AI-based solutions of the project,
- the Synthetic Data Cache for facilitating the storage and indexing of generated synthetic data,
- the SECURED Data Lake for storing synthetic datasets, the trained AI models and the stand-alone software tools,
- the knowledge graph for assisting the idea of organizing the semantic information of the existing datasets,
- the API core and the corresponding endpoints that provide acces to the data and the related information.

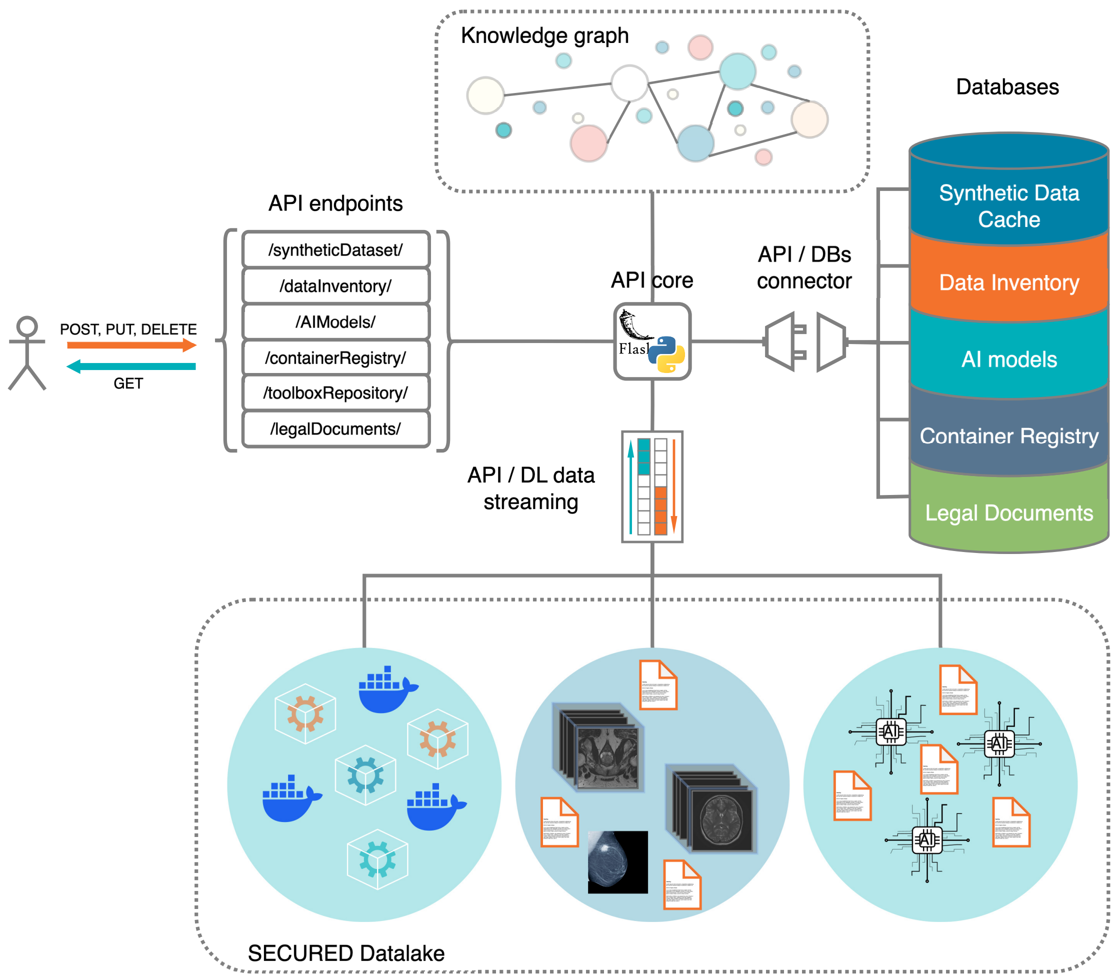


Figure 14 – The SECURED Knowledge Base architecture.

3.4.1 SECURE Data Lake

For addressing the demanding requirements of the project regarding the data storage, a Data Lake scheme is proposed aiming to provide flexibility, scalability and robustness. In this way, heterogeneous data, retrieved from different sources can coexist in such a centralized architecture, without the need of having a specific structure. The proposed scheme consists of three major components from the perspective of storage:

- The Containers/Applications Data Lake - including docker containers and images of SECURED applications along with software tools constituting the SECURED toolbox repository.
- The Synthetic–Data Data Lake - containing the synthetic data generated by the AI-based algorithms of T2.3.
- The AI Models Data Lake - storing the privacy preserving AI models and the corresponding weights of their trained instances.

From this point of the document and further, the Containers/Applications Data Lake will be referred as **KB_DL_CA** (Knowledge Base - Data Lake - Containers/Applications), the Synthetic–Data Data Lake as **KB_DL_SD** (Knowledge Base - Data Lake - Synthetic Data) and the AI Models Data Lake as **KB_DL_AI** (Knowledge Base - Data Lake - AI models). The functionality and utilization of these components is explained in the following subsections, along with the interaction with the rest of the architecture.

In an attempt to enhance the functionality of the Data Lake, a set of databases have been adopted for preserving essential metadata and building a strong indexing mechanism. Four distinct databases can be identified:

- the Synthetic Data Cache, abbreviated as **KB_DB_SD**
- the Data Inventory, referred to as **KB_DB_DI**
- the AI Models database, designated as **KB_DB_AI**
- the Container Registry database, abbreviated as **KB_DB_CR**

also accompanied by a Legal Documents Repository, referred to as **KB_DB_LD**.

3.4.2 Container Registry

Various services will be developed and integrated according to the requirements extracted from the corresponding tasks of the Grant Agreement. To provide ease of access, the Container Registry is introduced in order to collect metadata and store the service/application files by utilizing **KB_DB_CR** and **KB_DL_CA**. The related API endpoint is described below.

Container Registry API endpoint:

- **POST /containerRegistry/upload** This call uploads the zipped folder of the application or the docker container image to the **KB_DL_CA**. At the same time, it creates a new entry in the **KB_DB_CR** adding information like the name and the link of the application that should be provided as parameters to the call. The successful response includes the id of the new entry.
- **GET /containerRegistry/appInfo/appId** Given an existing application identifier, this call returns a JSON object containing all the requested information.
- **PUT /containerRegistry/appUpdate/appId** Given a valid application identifier, this call alters the corresponding entry and returns the proper success code.

3.4.3 Toolbox Repository

Specific applications, within the SECURED project, will be provided to the user as standalone tools that can be downloaded. To achieve this, the Toolbox Repository is introduced providing a proper API endpoint.

Toolbox Repository API endpoint:

- **POST /toolboxRepository/uploadTool** A zipped folder, containing the software files, along with additional metadata should be given in order for the call to be successful. In such a case, the zipped folder is uploaded, stored in the **KB_DL_CA** and the related metadata entry is retained in the corresponding database **KB_DL_CA**. If this process is completed, the tool identifier is returned.
- **GET /toolboxRepository/getTool/toolId** Given an existing service identifier, this call returns a JSON object containing all the requested information.
- **PUT /toolboxRepository/toolUpdate/toolId** Given a valid service identifier, this call alters the corresponding entry and returns the proper success code.

3.4.4 SECURED Data Inventory

The Data Inventory refers to private datasets, provided by the partners of the SECURED project, and open datasets proved to be essential for the purposes of the project. Since these datasets exist either on the premises of the corresponding partner or somewhere on the web, the Data Inventory aims to create a proper mechanism to index them by employing the **KB_DB_DI**.

Data Inventory API endpoint:

- **POST /dataInventory/create** Creates a new entry to the **KB_DB_DI** database by copying the parameters to the matching columns of the database table. The successful response returns the id of the new entry. Simultaneously, the new entry information feeds the Knowledge Graph component, resulting in the update of the nodes and the related links.
- **GET /dataInventory/getDatasetInfo/datasetId** Providing a valid dataset identifier, this call will return a JSON object with the information related to the specific dataset.
- **PUT /dataInventory/datasetUpdate/datasetId** Given an valid dataset identifier, this call alters the corresponding entry and returns the proper success code. In this situation, the Knowledge Graph is update as well.

3.4.5 Legal Documents Repository

The Legal Documents repository refers to private legal documents, provided by the partners of the SECURED project, and open legal documents proved to be essential for the purposes of the project. Since these documents exist either on the premises of the corresponding partner or somewhere on the web, the Legal Documents repository aims to create a proper mechanism to index them by employing the **KB_DB_LD**.

Legal Documents API endpoint:

- **POST /legalDocuments/create** Adds a new entry to the **KB_DB_LD** database by mapping the provided parameters to the appropriate columns of the database table. Upon success, it returns the ID of the new entry. This entry information is also used to update the nodes and related links in the Knowledge Graph component.
- **GET /legalDocuments/getDocumentInfo/documentId** Returns a JSON object containing details about a specific document when given a valid document identifier.

- **PUT /legalDocuments/documentUpdate/documentId** Modifies an existing entry identified by a valid document identifier and returns a success code. The Knowledge Graph is updated accordingly to reflect the changes.

3.4.6 Privacy Preserving AI-trained models

The Privacy Preserving AI points to the generative AI models of T4.3, trained to generate synthetic data. Their architecture and corresponding node-weights files are stored in the **KB_DL_AI**, while the related metadata are saved in the **KB_DB_AI**.

Privacy Preserving AI API endpoint:

- **POST /AIModels/upload** Uploads the aforementioned files and creates a new entry to the **KB_DB_AI** database by copying the parameters. The successful response returns the id of the new entry.
- **GET /AIModels/getModelInfo/modelId** Providing a valid model identifier, this call will return a JSON object with the information related to the specific dataset.
- **PUT /AIModels/modelUpdate/modelId** Given an valid model identifier, this call alters the corresponding entry and returns the proper success code.

3.4.7 Synthetic Data Cache

Synthetic Data Cache refers to the synthetic data generated by AI models of T2.3. The generated datasets should be stored in the **KB_DL_SD** and the related metadata in the **KB_DB_SD**.

Synthetic Data Cache API endpoint:

- **POST /syntheticDataset/upload** Uploads the selected, synthetic dataset and creates a new entry to the **KB_DB_SD** database by using the inserted parameters. The id of this new entry is returned in case of a successful response.
- **GET /syntheticDataset/getDatasetInfo/datasetId** Providing a valid dataset identifier, this call will return a JSON object with the information related to the specific dataset.
- **PUT /syntheticDataset/datasetUpdate/datasetId** Given an valid model identifier, this call alters the corresponding entry and returns the proper success code.

3.4.8 UML - Sequence diagrams

In Figures 15, 16, and 17, the sequential diagrams of the POST, GET and PUT methods are presented respectively, illustrating their general concepts according to the aformentioned explanations of the API endpoints. About the POST method including file uploading process, the following steps occur:

1. The actor (user or service) selects the file to upload and inserts the matching parameters.
2. The call to the proper API endpoint is initiated.
3. The compliance of the file is assessed by the server and, then, it is sent to the Data Lake.
4. The validity of the metadata is examined by the server and, then, it is saved to the database.
5. The final POST response is returned to the actor.

In case of a POST method, depicted in Figure 15 without the file uploading included, the same procedure is followed without taking into account the actions highlighted in orange.

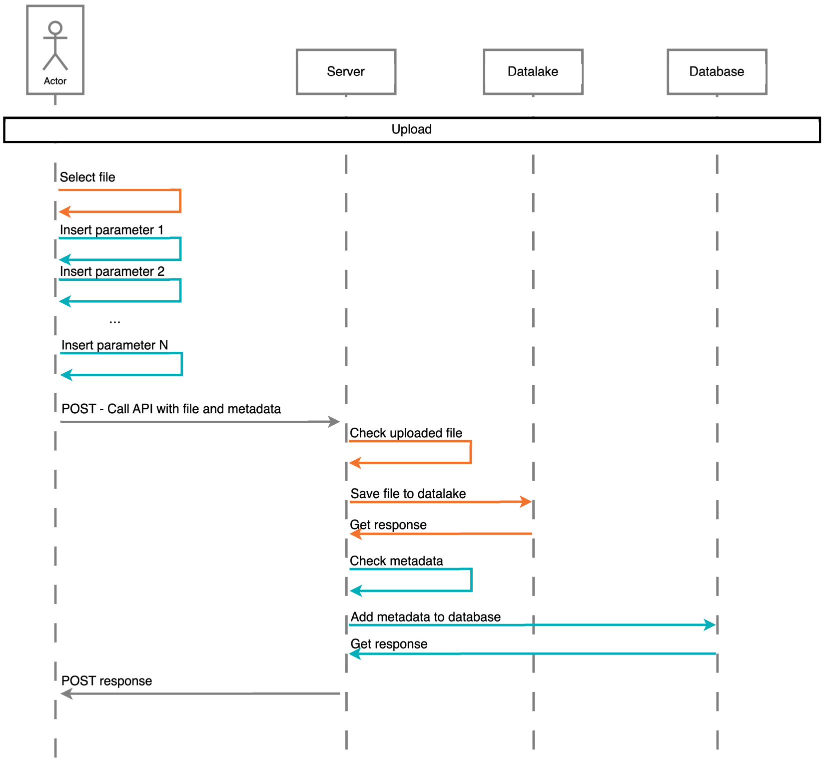


Figure 15 – The SECURED Knowledge Base Common API - POST.

Regarding the GET method, depicted in Figure 16, including file uploading process, the following steps occur:

1. The actor (user or service) inserts the id of the file to download

2. The call to the proper API endpoint is initiated.

3. The validity of the given identifier is reviewed by the server.

4. The server searches the corresponding database to locate the identifier.

5. If the identifier exists, the related file is returned to the actor along with the metadata of the entry encapsulated in the GET response body.

In case of a GET method without the file downloading included, the same procedure is followed without considering the actions highlighted in orange arrows.

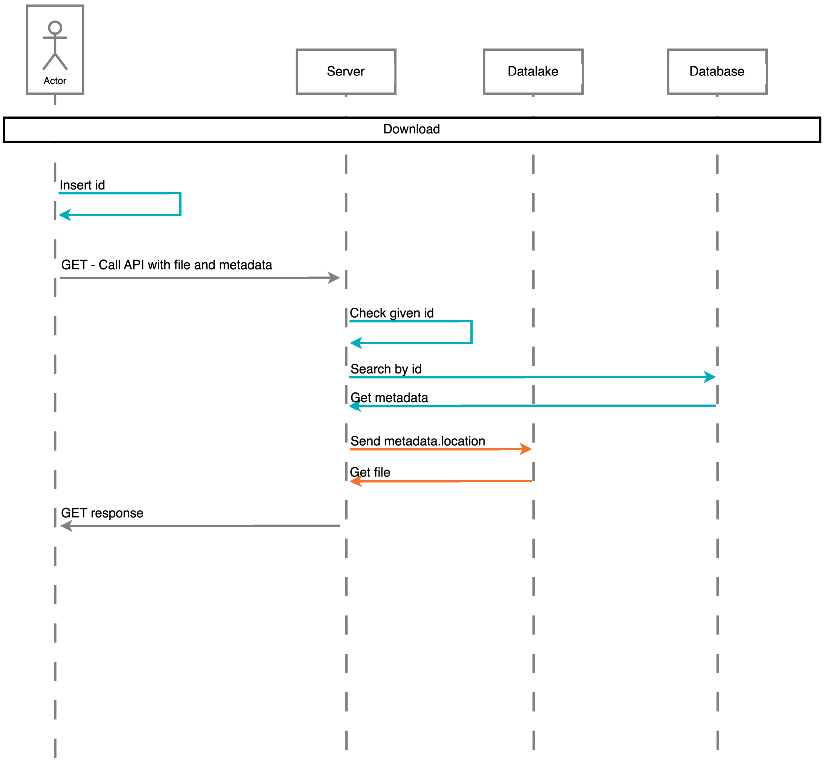


Figure 16 – The SECURED Knowledge Base Common API - GET.

About the PUT method, as depicted in Figure 17, the following steps occur:

1. The actor (user or service) inserts the id of the entry to be altered.

2. The actor (user or service) inserts the new parameters.

3. The call to the proper API endpoint is initiated.

4. The validity of the given identifier is assessed by sending a proper request to the database.

5. If the identifier is validated, the new parameters are reviewed by the server.

6. The server sends the metadata to the database

7. The PUT response is return to the user including the identifier, in case of a successful transaction

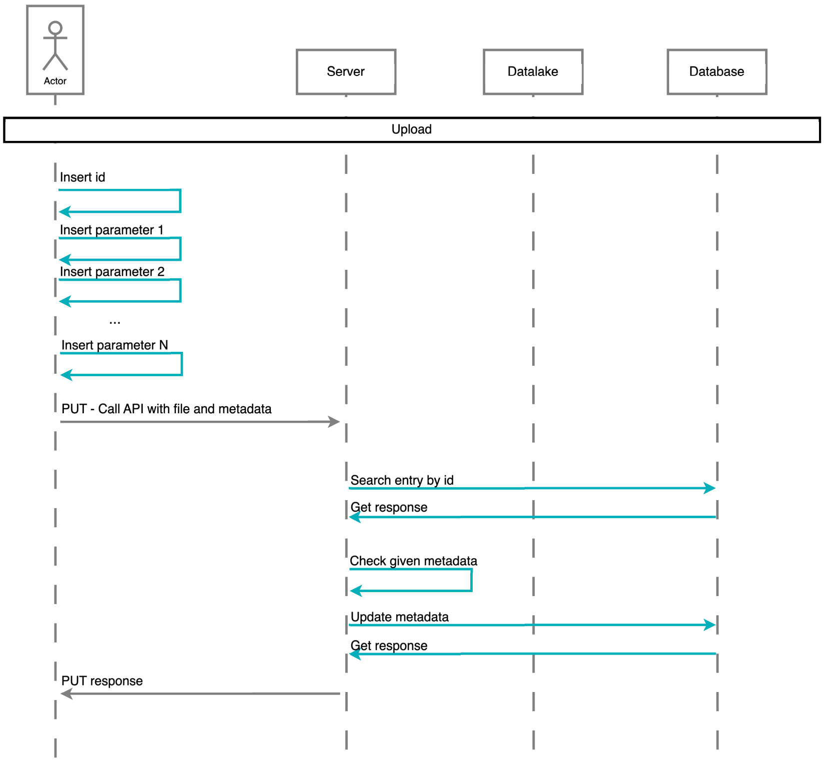


Figure 17 – The SECURED Knowledge Base Common API - PUT.

3.5 Innohub Services

The Innohub Services architectural block consists of two main service categories, namely the Platform and the Privacy Preserving Services. Each block is responsible for offering specific functionalities to the user, thus elevating the Innohub to a secure and trusted workshop offering different capabilities per use-case. The decentralized nature of the SECURED federated infrastructure is represented through the distinctive yet collaborative design and categorization of the available services, offered through the Innohub.

3.5.1 Platform Services

The platform services is comprised of three high-level SECURED services that enable the user to oversee and manage the data they own in a top-tier manner. Additionally, users are presented with options regarding the best governance of their data according to the already developed machine and deep learning techniques of the SECURED project. The Anonymisation Decision Support service provides dedicated support for selecting the most suitable anonymisation technique, tailored to the specific characteristics of a health dataset. Cross-border data processing legal/GDPR compliance is a dataset-specific legal framework to ensure the dataset's compliance with EU or country-specific legislation per case. Finally, the ML, AI, FL model Marketplace is a repository of anonymised, unbiased, trained models that have been produced by the SECURED ecosystem, available to the user for exploitation per use-case.

3.5.1.1 Model Markeplace ML/AI/FL

The Model Marketplace component serves as a comprehensive repository of anonymised, unbiased, and trained models developed by the SECURED ecosystem. The purpose of this repository is to provide users with access to a collection of models that can be employed per use case, ensuring that the proposed Machine/Deep/Federated Learning solutions are tailored to their specific needs. For accessing this repository, a front-end interface is used in order for the user to be able to search and download the desired model. The main component of the interface will be a search bar, serving the searching mechanism, followed by several filter options, including fields for entering the model name, selecting the use case from a drop-down menu, choosing the category, entering the developer's name, and specifying the date range for the model's creation date.

To begin the search, the user enters relevant criteria into the provided fields. For instance, the user might type part of the model name into the text input field, select "Mammogram" as the use case from the drop-down menu, choose "breast" from the category options, enter the developer's name if known, and specify a date range. Once all desired filters are set, the user clicks the "Search" button to initiate the search. The application processes the search query and displays the results in a list format. Each search result is presented with key information including the model name, a short description of the model's functionality, its primary use case, category, developer, and the creation date. This allows the user to get a quick overview of the options available. After reviewing the model information, the user can download the model by clicking the corresponding "Download" button, which is prominently displayed on the model's row, within the list. A download dialog appears, asking the user to confirm the process and select to download or not the related JSON file as well. Once the user confirms, the download begins, and a progress bar or notification indicates the download status.

3.5.1.2 Anonymisation Decision Support

The Anonymisation Decision Support component is meant to provide anonymisation guidelines for a given dataset and given constraints. Careful consideration is required to determine which measures are sufficiently adequate, especially when dealing with data in the health care domain. The central idea is that the user will submit the characteristics of the dataset, the requirements in terms of security, and the other constraints, such as the performance ones and based on the input provided by the user, the Anonymisation Decision Support will direct the user toward the most suitable solution for the given problem. There is a wide choice of anonymisation schemes available, and they all depend on a wide range of parameters. Because of this, the Anonymisation Decision Support can be coupled with design space exploration to ensure that the selected algorithms, parameters ranges and combinations are the best fit for the given characteristics of the dataset and the associated requirements / constraints.

The envisioned workflow of the Anonymisation Decision Support is as follow:

- The data owner provides, via a web interface, the characteristics of the data, the requirements in terms of performance, and the requirements in terms of privacy (these are provided by a means of guided dialog)
- The web platform reports the most suitable combination of parameters / algorithms that fulfill the privacy and the other provided requirements

3.5.1.3 Legal/GDPR Compliance Check

This service aims to assist the SECURED Innohub user in identifying the relevant legislation and guidelines that are associated with the functions he/she want to utilized within the Innohub. The services will operate in a online "wizard" fashion guiding the user to check if the approach on anonymizing or synthesizing data is compliant with the EU regulations and the offered privacy guidelines. For that, an ontology graph of the existing Privacy guidelines and associated regulations will be created by technical partners linked with the relevant documents stored in the SECURED Knowledge base. The service guided by the interactive user input will communicate with the Knowledge base through a dedicated API in order to fetch compliant privacy guidelines and also regulatory frameworks on the datasets registered through the Innohub. The source of the visualised information will be collected from the pertinent legal and ethical framework identified. In particular, legal norms established under the General Data Protection Regulation (GDPR), including, for instance, relevant GDPR definitions, principles and the data protection impact assessment have been reported in D1.2 [5]. These have been complemented by pertinent rules (hard law and soft law) in relation to data, data governance laws, artificial intelligence, and cybersecurity, as described in D2.5 [6], creating a regulatory roadmap and framework that contributes to the implementation of the SECURED Innohub and technologies. Eventually, the service will take into account the opinions and recommendations issued by independent EU bodies, such as the European Data Protection Board (EDPB) and its predecessor Article 29 Working Party (WP29), intending to ensure a consistent application of data protection provisions throughout the EU (e.g., WP29 Opinion 05/2014 on Anonymisation Techniques[3]; EDPB Guidelines 04/2019 on Article 25 Data Protection by Design and by Default[4]), or relevant guidelines issued by the European Union Agency for Cybersecurity dedicated to achieving a trusted cyberspace and a high common level of cybersecurity (e.g., ENISA Data Pseudonymisation: Advanced Techniques and Use Cases[5]).

3.5.2 Privacy Preserving Services and Innohub Tools

At the heart of the SECURED architecture lies the Privacy Preserving Services component, a comprehensive suite of essential resources for medical data bias assessment and amendment of possible biasing violations, in a plethora of ways. Additionally, users have access to the Synthetic Data Generation engine, enabling the creation or enrichment of existing datasets with realistic yet privacy-preserving entities for medical research and analysis purposes.

Apart from utilizing the interface for assessing, traversing, and transforming privacy-preserved services, Innohub users have the option of downloading these functionalities as tools, as standalone software versions that can be deployed on the user's premises. Architecturally, the Innohub Tools lie on the same level as the Services (see Figure 1). The tool versions of the components are offered with a handbook of instructions on how to execute, maintain, and experiment with the developed technologies, in addition to examples for efficient usage and maximum exploitation. A number of functionalities, specifically the tools related to anonymisation are not available as services.

[3] Article 29 Data Protection Working Party "Opinion 05/2014 on Anonymisation Techniques". Adopted on 10 April 2014 WP216. {https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf}

[4] European Data Protection Board "Guidelines 4/2019 on Article 25 Data Protection by Design and by Default". Version 2.0. Adopted on 20 October 2020. https://www.edpb.europa.eu/sites/default/files/files/file1/edpb_guidelines_201904_dataprotection_by_design_and_by_default_v2.0_en.pdf;)

[5] European Union Agency for Cybersecurity "Data Pseudonymisation: Advanced Techniques and Use Cases", 28 January 2021. https://www.enisa.europa.eu/publications/data-pseudonymisation-advanced-techniques-and-use-cases

3.5.2.1 Synthetic Data Generator Service and Tool

The Synthetic Data Generator (SDG) is the component that is in charge of generating the different data types that are provided by T2.3. The current main data modalities to generate, are the following:

- Images:
  - Mammograms
  - Pathological image
  - Chest X-ray
  - Missing MRI slices
- Time series:
  - Fetal Heartbeat Rate (CTG)

The service/tool is modularized as shown in Figure 18. The SECURED library is the main entry point to the application for both the Application Programming Interface (API) and the Tool User Interface (UI). This is done to avoid code duplication and provide different interfaces for the same tools. The SECURED library contains the SDG core, which is the main engine for all the available generators that were listed before.

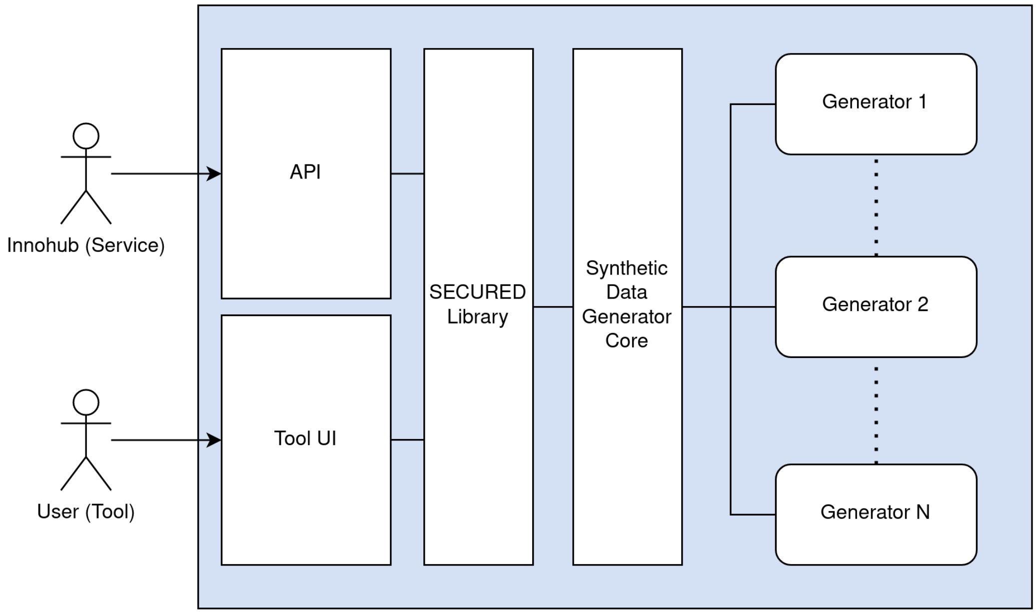


Figure 18 – Synthetic Data Generator (SDG) module structure. The SDG core integrates all the different generator models, while the library provides a common interface for both the Service and the Tool.

Service flow specification

The features of both the tool and the service can be divided into three distinct functionalities:

1. Generate data without customization: Request the generation of data of a particular type and modality without any other kind of input.
2. Generate data requesting specific characteristics (conditioning): Request the generation of data of a particular type and modality and specifying characteristics of the generated image, e.g., the density of the tumor in a mammogram.
3. Interpolate data or transfer data from one domain to another: Given a set of data, create data points that are in the middle, i.e., from an Magnetic Resonance Imaging (MRI) scan, create a new image that falls between two scans in the 3D representation.

The first and second functionalities are described in Figure 19; they are the basic ones, as they refer to invoking the generator without or with parameters, e.g., conditioning parameters to apply to the model to generate a given class of data. If no parameters are given, the first functionality is provided. If parameteres are given, the second functionality is provided. In the sequence diagram of Figure 19, Innohub interacts with the API performing a request to generate the data and, then the API internally handles the petition, e.g., queuing, and when possible it interacts with the library/SDG Core to get the appropriate generator and use it. The results are stored in the file system and subsequently sent to the Knowledge Base.

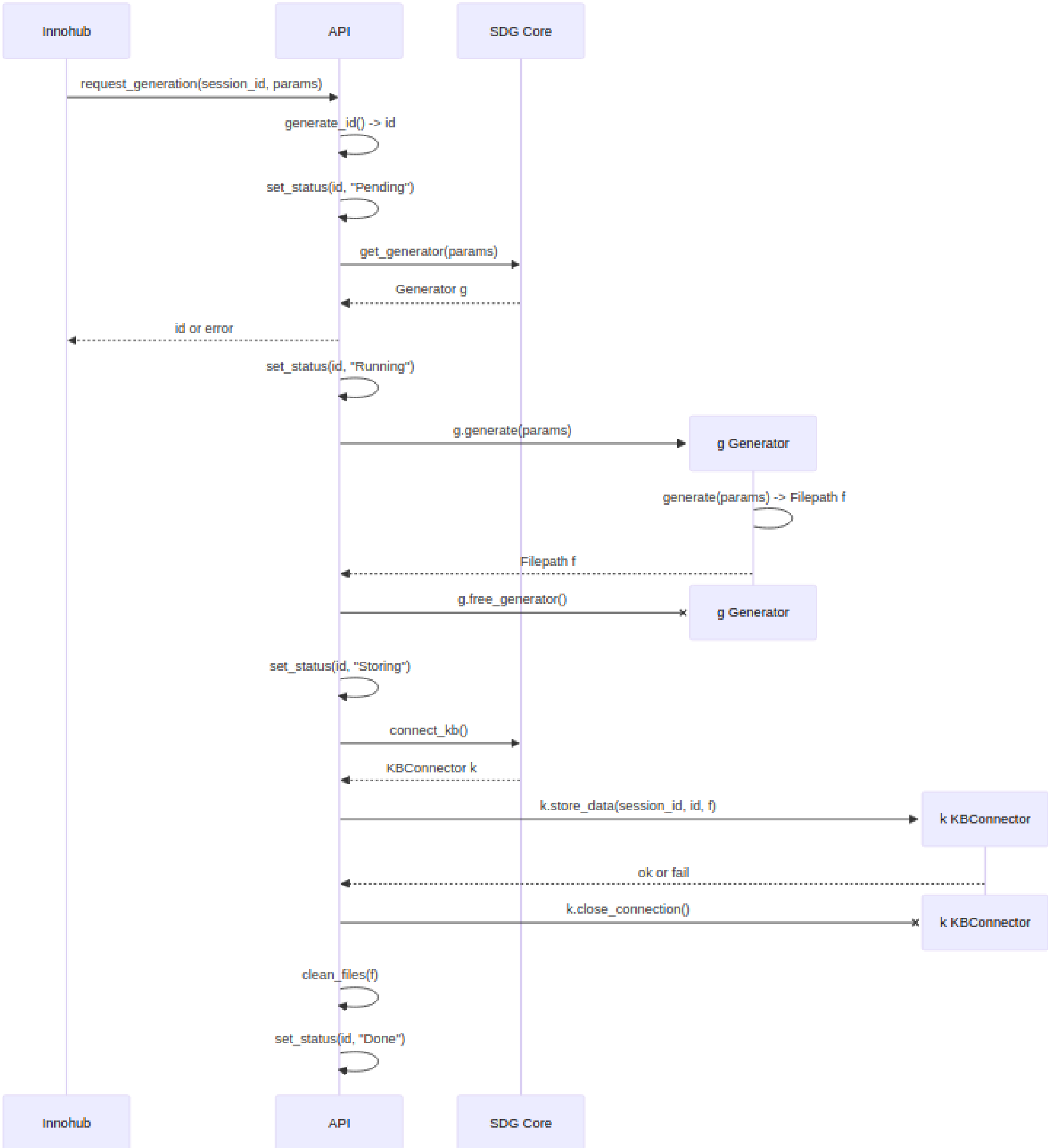


Figure 19 – Sequence diagram of *generate data* (with or without paremeters). Covers functionalities 1 and 2.

In a similar fashion, Figure 20 shows the generation of data using input data as a basis. These data might come from the Knowledge Base; however, this is not defined yet. This particular case covers the third functionality, e.g., interpolate between two existing MRI scans to create one that is in the middle.

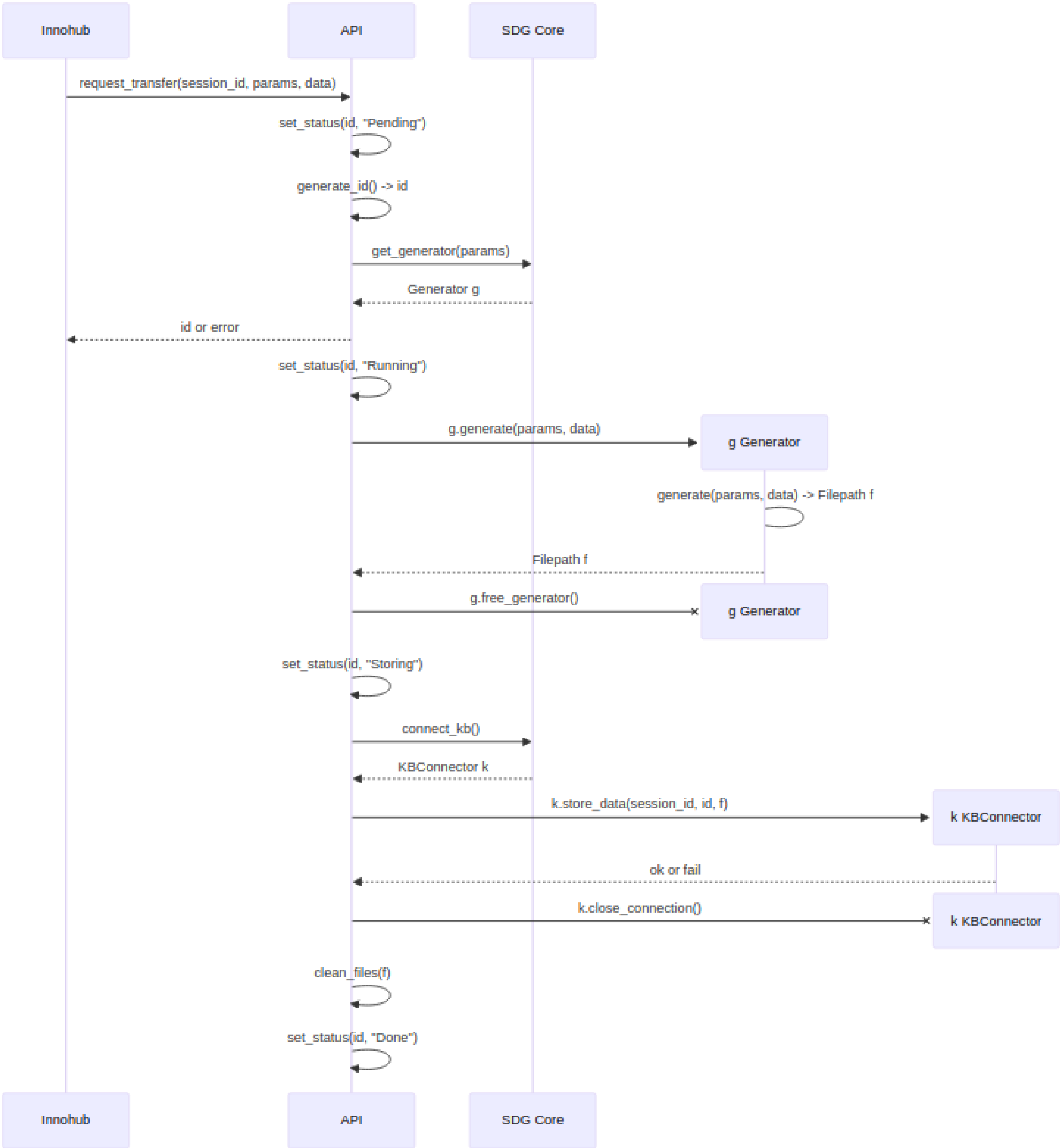


Figure 20 – Sequence diagram for *generate data using pre-existing data*. Covers functionality 3.

Finally, as auxiliary diagrams, Figure 21 shows the flow on how to retrieve the generators to be used and Figure 22 requests the status of the executions from the Synthetic Data Generator (SDG).

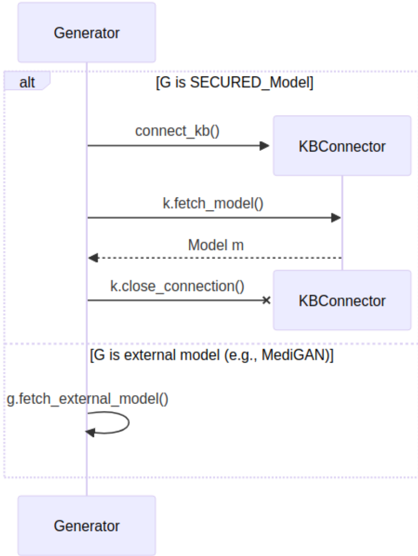


Figure 21 – Get generator

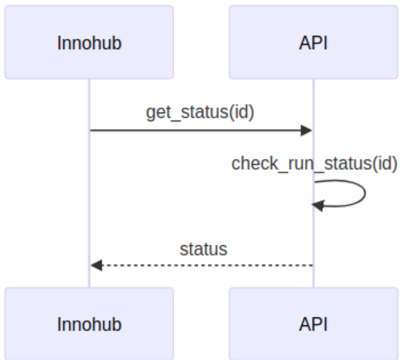


Figure 22 – Get run status

3.5.2.2 Unbiasing Service and Tool

The Unbiasing Service/Tool is the component responsible for mitigating existing bias in AI service. The service is not yet implemented. With potential modification during its implementation, the service will be modularized as shown in Figure 23. Fairness mitigation would be divided in three submodules:

- The sub-component dedicated to pre-processing approaches, that mitigates the bias directly on the dataset potentially used during AI model training;
- The sub-component dedicated to in-processing approaches, that mitigates the bias during AI model training;
- The sub-component dedicated to post-processing approaches, that mitigates the bias directly on the dataset potentially used during AI model training.

Details about pre-processing, in-processing, and post-processing methods for bias mitigation are given in Deliverables D3.1[7] and D4.1[1].

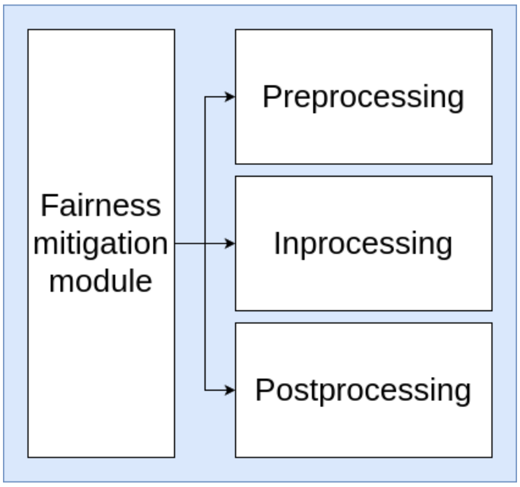


Figure 23 – Unbiasing service module structure.

In the sequential diagrams provided in Figures 24, 25, and 26:

- The user is the actor who want to train AI model;
- The server is a device with computational and storage ability;
- fairness_mitigation is the component dedicated to mitigation of bias.

The sequential diagram for the first subcomponent is shown in Figure 24. The user prepares a configuration file with the name of the method she wants to use, the path to the dataset, the sensitive attribute, etc. The fairness_mitigation tool applies the method chosen and provide to the user the new version of the dataset. The user then uses the new dataset to train her AI model.

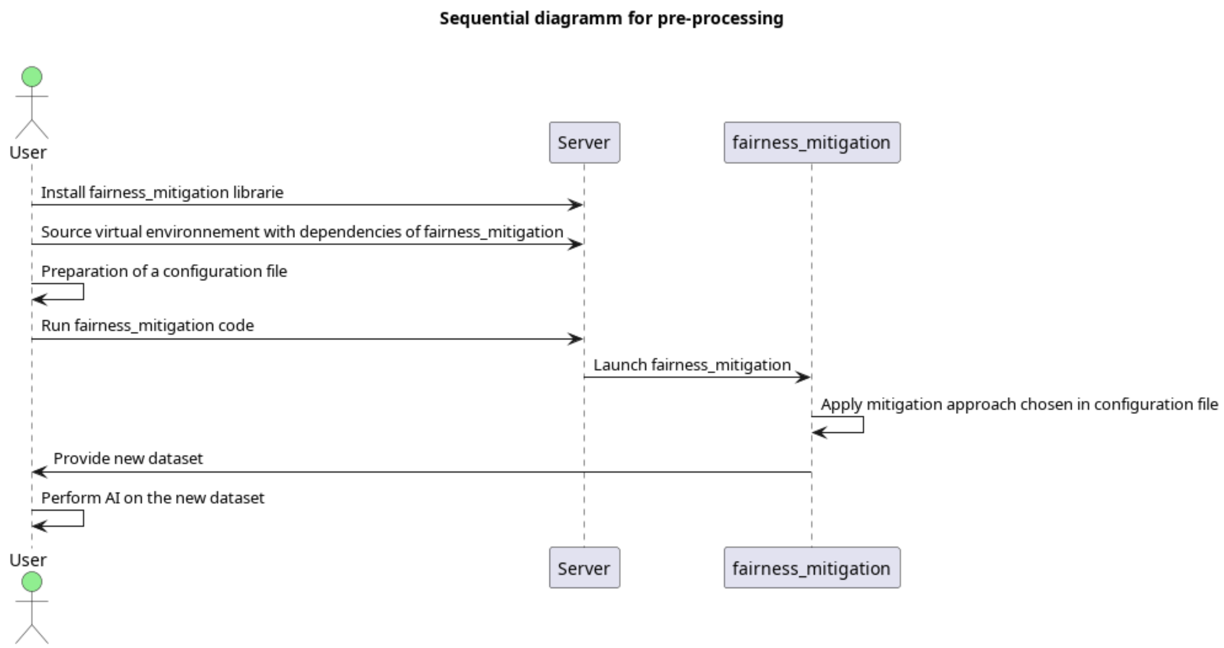


Figure 24 – Sequential diagram for preprocessing approach.

The sequential diagram for the second subcomponent is shown in Figure 25. The user prepares a configuration file with the name of the method she wants to use, the path to the dataset, the sensitive attribute, etc. Moreover, she implements the AI Model that will be trained. The fairness_mitigation component proposes tools and examples with inprocessing approaches to mitigate the bias and help the user who uses them to train a Fair AI model.

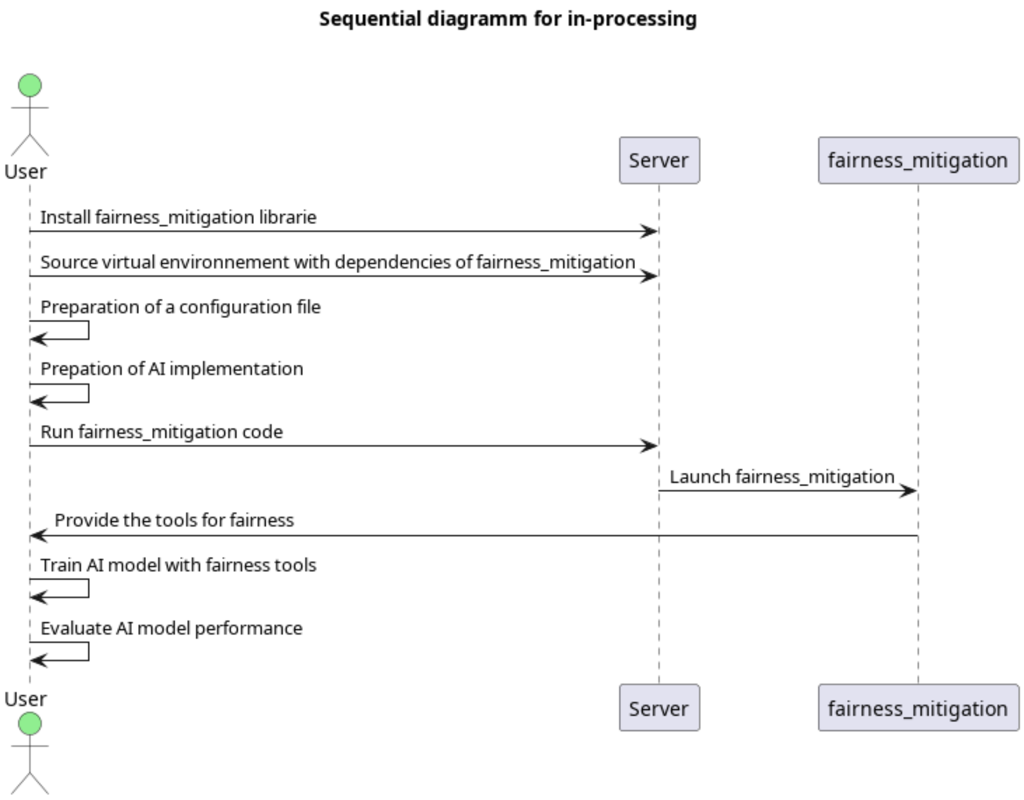


Figure 25 – Sequential diagram for inprocessing approach.

The sequential diagram for the last sub-component is given by Figure 26. The user prepares a configuration file with the name of the method she wants to use, the path to the dataset, the sensitive attribute, etc. Moreover, she implements and trains her own AI model. The fairness_mitigation component applies post-processing methods to this AI model and provides to the user fairer version of her AI model.

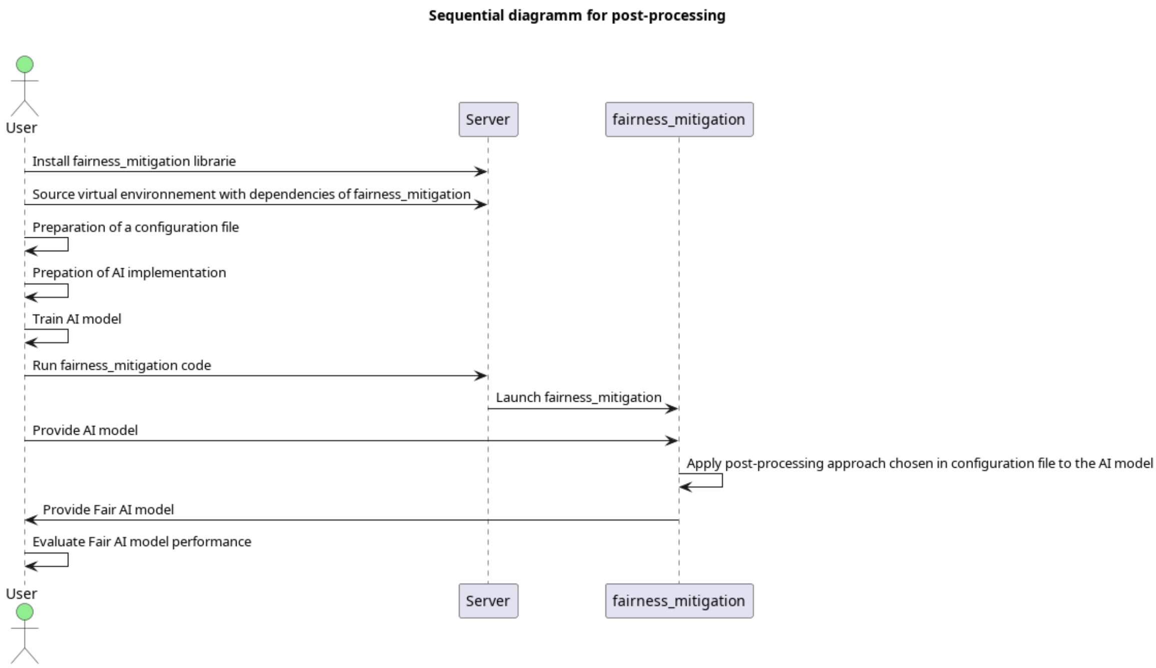


Figure 26 – Sequential diagram for postprocessing approach.

3.5.2.3 Bias Assessment Service and Tool

The Bias Assessment Service/Tool is the component responsible for measuring existing bias in an AI service. With potential modification during its implementation, the service will be modularized as shown in Figure 27. The Bias Assessment Service would be divided in two submodules:

- The sub-component dedicated to bias assessment for AI model used for data generation;
- The sub-component dedicated to bias assessment for AI model used for classification.

Details about the metrics used are given in Deliverables D3.1[7] and D4.1[1].

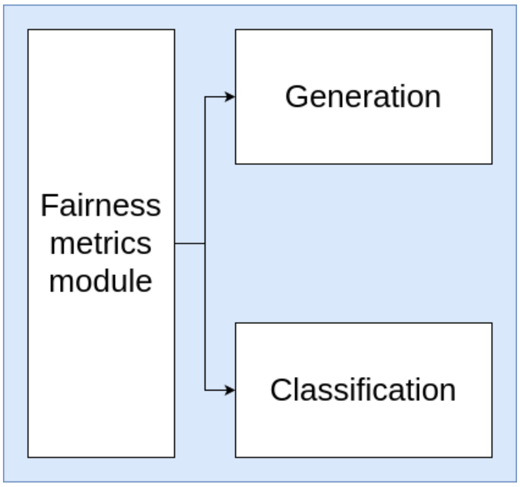


Figure 27 – Bias assessment service module structure.

In the sequential diagrams provided in Figures 28 and 29 :

- The user is the actor who inspects an AI model;

- The server is a device with computational and storage ability;

- fairness_metric is the component dedicated to bias assessment.

The sequential diagram of the first subcomponent is shown in Figure 28. The user prepares a configuration file with information such as the path to a subset of real images (called reference), a path to a subset of generated images (called generated) with the generator that is evaluated, information about the metrics used, etc. Then the fairness_metric component extracts from the paths provided by the user, deeper information about the sensitive attribute values and loads the reference and generated images. Then it computes the metrics and provides a summary and boxplots about the metrics. The results are made available to the user.

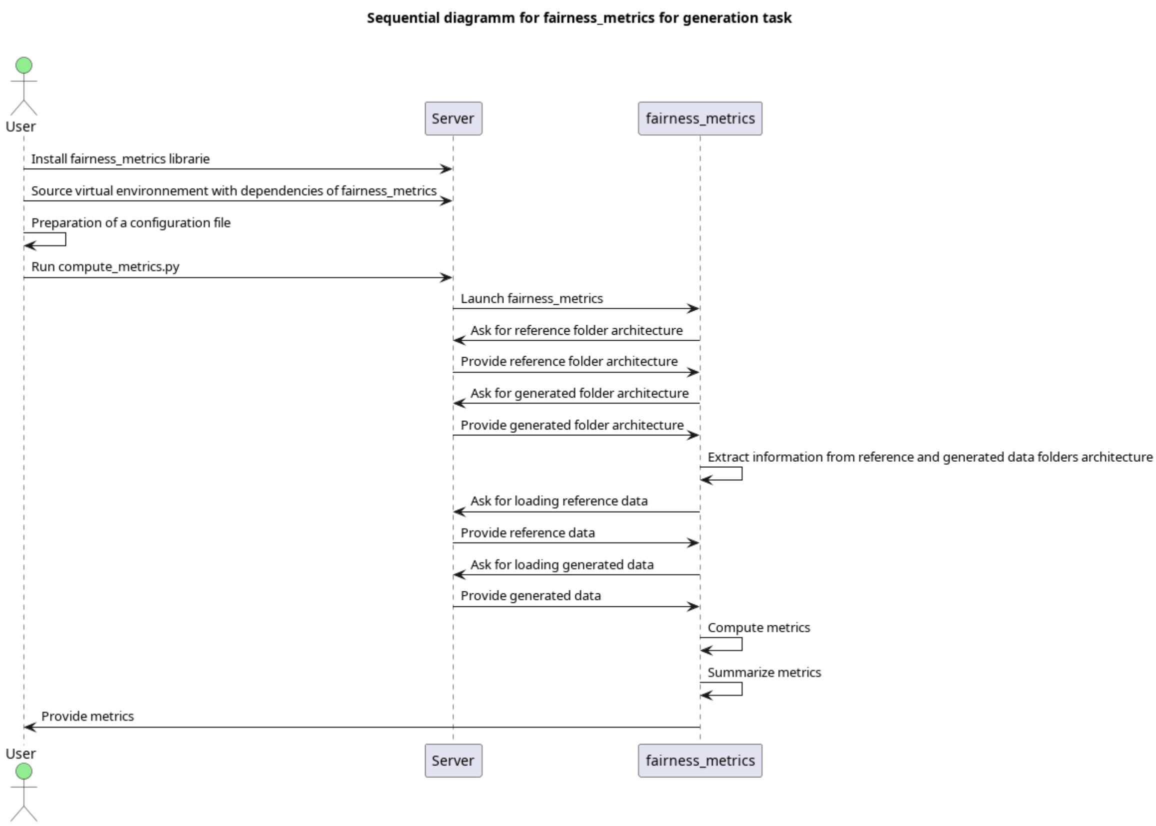


Figure 28 – Fairness_metric for generation task.

The sequential diagram of the first sub-component is given Figure 29. The user prepares a configuration file with information such as the path to the dataset to inspect, the prediction function or values from the AI model to inspect, the sensitive attributes, the name of the metrics to compute, etc. fairness_metric computes the metrics and provides the result to the user.

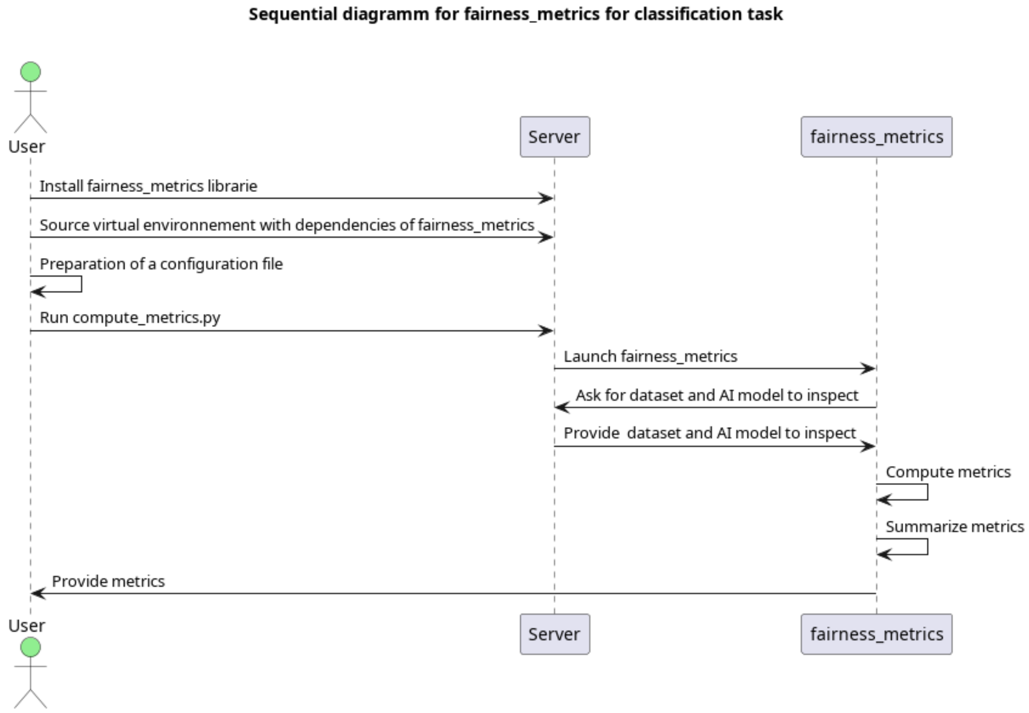


Figure 29 – Fairness_metric for classification task.

3.5.2.4 Anonymizing Tool

The Anonymisation tool, also named DANS 2.0, is developed by Atos and is devoted to preserving data privacy, mitigating data leakage, avoiding reidentification, and keeping data utility. DANS 2.0 is based on open-source libraries (ARX[6] and Amnesia[7]) providing data anonymisation techniques (e.g., generalisation, suppression, microaggregation) and supporting different privacy models (e.g., k-anonymity, l-diversity, t-closeness, km-anonymity, differential privacy) [1]. These techniques and models will be applied to different types of data such as Electronic Health Records (EHRs) or time-series data. In the context of the SECURED project, this anonymisation tool will be offered as a tool and as a service, to be deployed on the data provider infrastructure. An OpenAPI is provided for facilitating their integration on wider frameworks such as the SECURED Innohub Platform.

Component description

The Anonymisation tool is based on the architecture design provided in Section 4 of D2.1 [8] comprising the following five layers, as depicted in Figure 30:

- Anonymisation services layer
- Anonymisation Manager
- Public API
- Visualisation layer
- Storage layer

Based on this architecture, the anonymisation tool is built on different microservices as depicted in Figure 31.

- Legacy DANS microservice provides several privacy models such as k-anonymity, l-diversity, t-closeness or Differential Privacy to be applied on big datasets.

[6] https://arx.deidentifier.org/
[7] https://amnesia.openaire.eu/

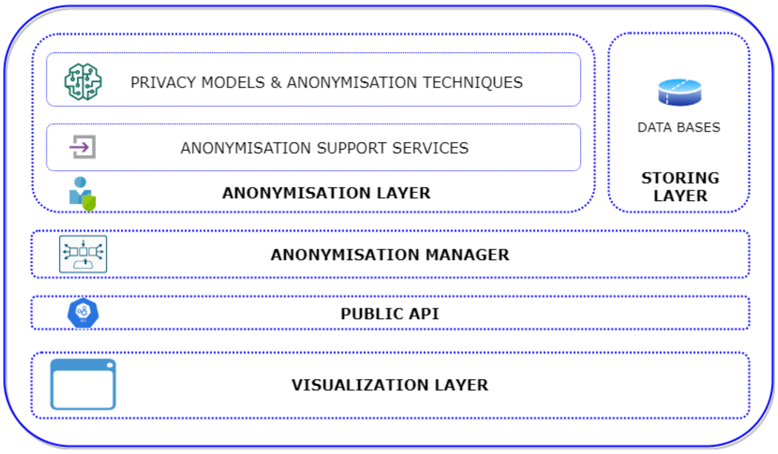


Figure 30 – High-level view of DANS 2.0 architecture [8].

- Amnesia microservice provides as k-anonymity and as km-anonymity addressed to small datasets.
- Anonym Manager microservice is in charge of orchestrating the anonymisation process and offers a ReST API for accessing the functionalities offered.
- The different microservices can store the datasets, hierarchies, and associated metadata on linked data bases.

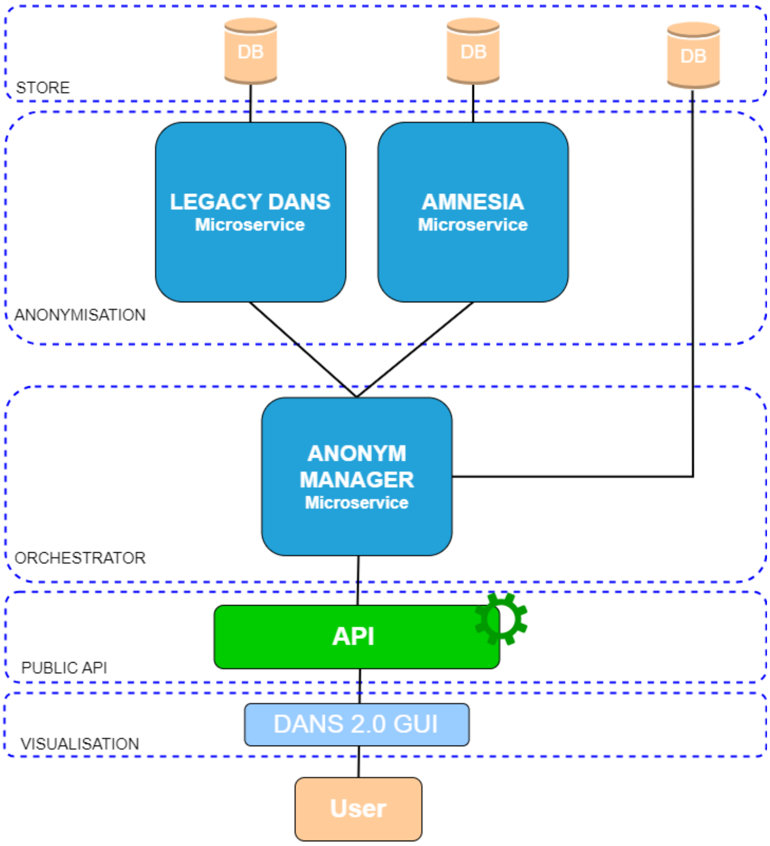


Figure 31 – Anonymisation tool (DANS 2.0) Microservices overview.

Relationship with the use cases

The Anonymisation tool is able to anonymise datasets generated in different environments. As introduced in D2.1[8], this tool will be involved in the protection of the use cases indicated in Table 4.

Table 4 – Uses cases related to Anonymisation Tool

UC1 Real-Time Tumor Classification	UC2 Telemonitoring for Children	UC3 Synthetic Data Generation for Education	UC4 Access to Genomic Data
NA	X	X	X

The data to be anonymised will be basically EHRs in a first stage and time-series data later on.

Technical specifications

The Anonymisation tool will be integrated into the Anonymized Data Transformation (ADT) as shown in Section 3.5.2.6, which is a composition of all the services related to anonymisation, re-identification, bias and synthetic data generation. For integration with the rest of the SECURED components and specifically with the ADT components, a ReST API is provided to facilitate access to the anonymization functions offered. Figure 32 displays the endpoints offered by the API.

This OpenAPI contains three groups of endpoints for accessing the anonymisation functionalities as follow:

- **Dataset group** for Upload/Download/Storage files and metadata.
  - **POST /dataset/loadDataset** Upload datasets for the anonymisation process. Provides the file and additional data description of the dataset, file extension, type of data, etc. Returns a file identifier and suggested tools to anonymise the file.
  - **GET /dataset/getFile/fileId** Download a file dataset providing the file identifier. Returns the dataset.
  - **GET /dataset/getFileMetadata/fileId** Request the file metadata associated to a dataset file. Provides the file identifier. Returns a list of attributes definition and metadata associated to the dataset.
  - **POST /dataset/updateFileAttributes/fileId** Update the attributes definition of a file dataset. Returns the file identifier.
- **Anonymisation group** for accessing anonymisation functionalities.
  - **POST /anonymisation/anonymise** Anonymises a dataset based on specified parameters. Provides the dataset identifier, the privacy models and the tool to apply. Returns the anonymised dataset, the anonymised identifier, the anonymisation status, metrics of the process and additional information.
  - **GET /anonymisation/getAnomymisedFile/fileId** Retrieve anonymised file based on the unique fileId. Provides the anonymised file.
  - **GET /anonymisation/getStatisticsFile/anonymisedFileId** Generates a report including suppression percentage, number of records, risk profile, etc. Provides the anonymised file identifier. Returns the statistics report of the anonymisation process.
  - **GET /anonymisation/analyse** Generates information related with the privacy risk and utility related to the dataset to be anonymised. Provides the file identifier, the tool to apply and the k parameter.
- **Hierarchy group** for Upload/Download/Create hierarchies.
  - **POST /hierarchy/loadHierarchy** Load an already created hierarchy file. Provides the hierarchy file and the associated metadata. Returns the unique hierarchy's name and the tool associated.
  - **GET /hierarchy/getHierarchyFile/hierarchyName** Retrieve hierarchy file based on the unique hierarchy's name. Provides the hierarchy's name. Returns the hierarchy file.
  - **POST /hierarchy/createHierarchy** Creates a hierarchy file based on the dataset and the target tool. Provides the associated dataset identifier, the associated attribute, the type of attribute, the tool to be used, the hierarchy's name, the hierarchy type and the additional parameters if needed. Returns the hierarchy file and the associated tool.

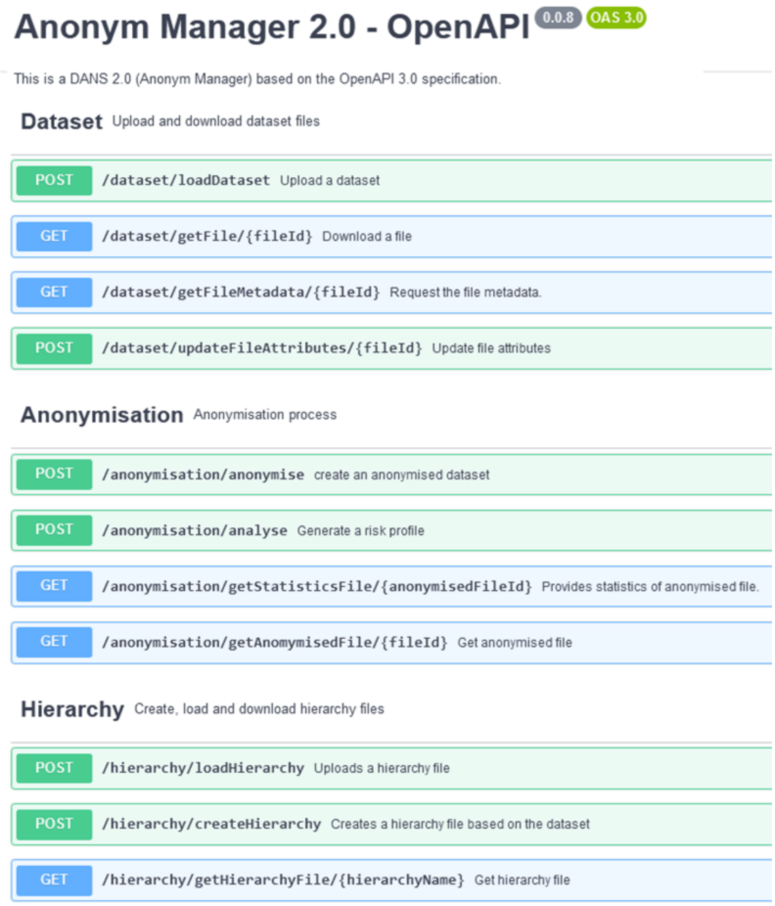


Figure 32 – Swagger OpenAPI provided by the Anonym Manager.

The anonymisation process requires to follow several steps for setting up some parameters such as the description of the dataset, the definition of the attributes, the privacy models and the anonymisation techniques to apply, etc. In this context, Figures 33, 34 and 35 provides the flow for the anonymisation process, which can be split in a group of actions as follows:

- **Load Dataset**: for uploading, storing and describing datasets. Also, updating attributes' definition (Figure 33).
- **Download Files**: for retrieving file datasets (Figure 33).
- **Metadata**: for retrieving metadata associated to the datasets (Figure 33).
- **Hierarchies**: for uploading already created hierarchy files and create hierarachies. Also, retrieve hierarchy files already created. Also, retrieve hierarchy files already created (Figure 34).
- **Anonymise dataset**: for setting privacy models and associated parameters (Figure 35).
- **Statistics**: for obtaining a report of the anonymisation process (Figure 35).
- **Analyse dataset**: for getting a previous anonymisation analysis (Figure 35).

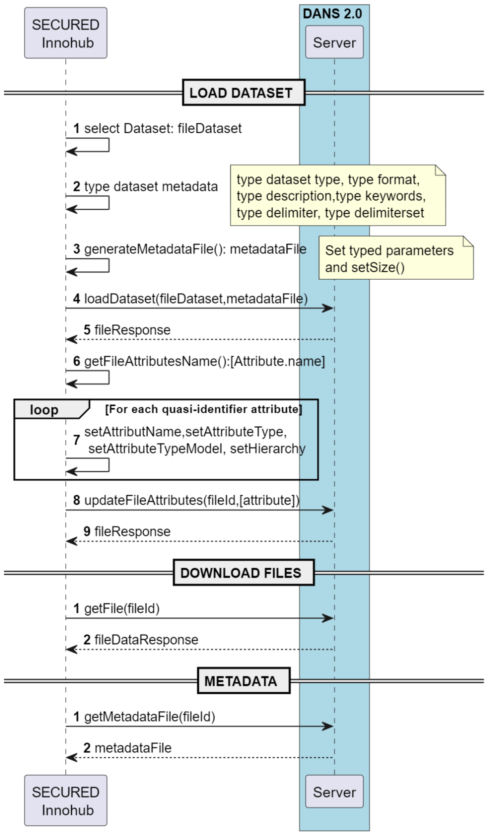


Figure 33 – Upload and download datasets, and get metadata flow.

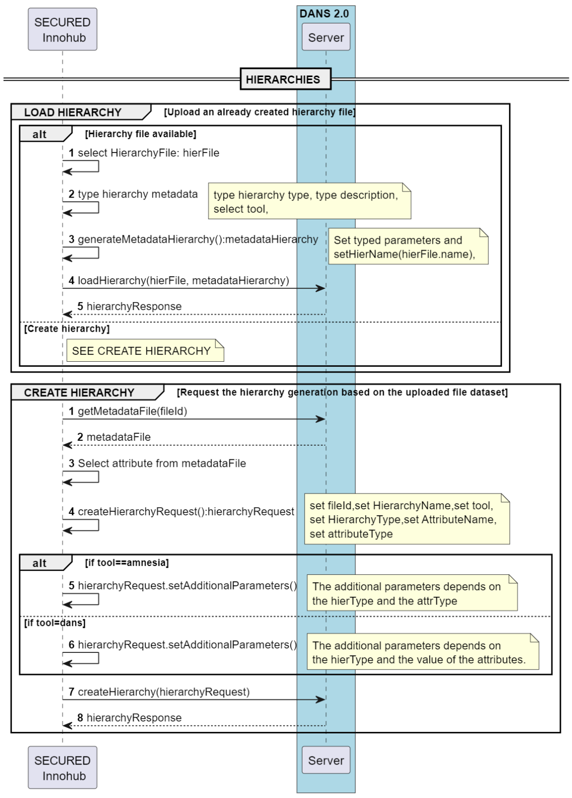


Figure 34 – Load and create hierarchies flow.

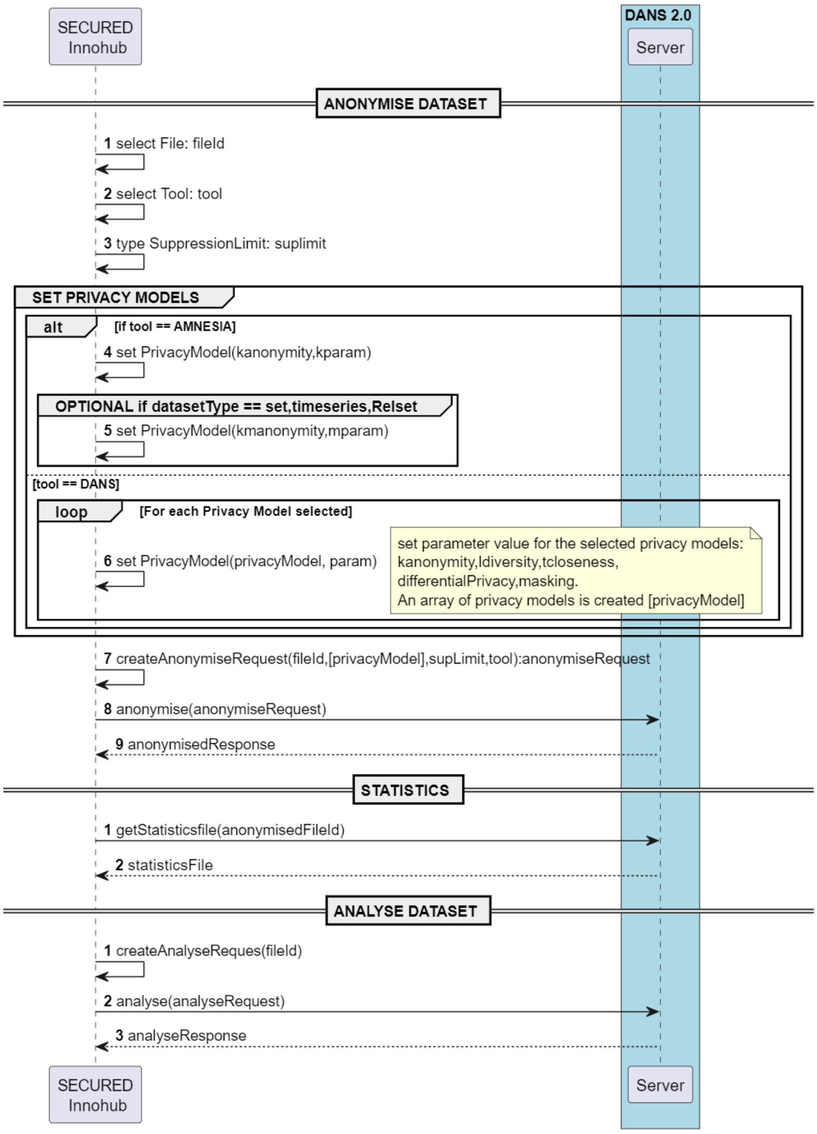


Figure 35 – Anonymisation process flow.

Component installation methods

The Anonymisation tool will be deployed as a docker container following the diagram depicted in Figure 36. Based on the components described in the Technical Specifications Section 5, this docker container comprises the following microservices:

- The Anonym Manager service
- The Legacy-DANS service
- The Amnesia service
- Other anonymisation services to be included in the future
- A Data Base for storing datasets, hierarchies, and associated metadata.

The upper part of Figure 36 shows the interactions with the ADT components.

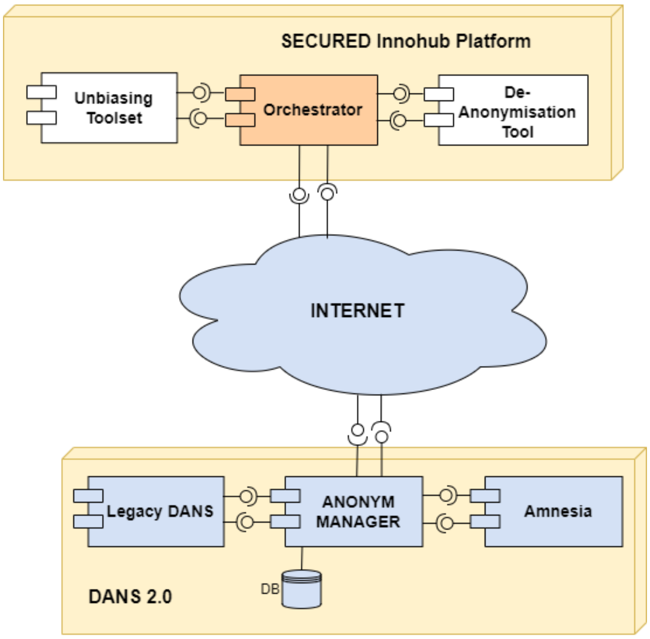


Figure 36 – Anonymisation tool (DANS 2.0) deployment.

3.5.2.5 Anonymisation Assessment Tool

The service will provide a platform enabling data owners to input details of their datasets (datatype and structure, parameters, size and population etc.). The service will the return a ranked list of existing de-anonymisation and re-identification attacks (including those designed as part of the project) that are relevant to their specific data and parameters, and where available, provide the tools implementing those attacks and guidance on their use and interpretation of the results. This will enable the data owner to assess their anonymisation strategy against known threats.

The platform will be web-based, and its development will follow a modular design and implementation concept, enabling future expandability.

The Anonymisation Assessment Tool will operate as follows.

- **User:** data owner with access to locally-stored anonymised data. In order to check the assessment results (last step of the workflow below), access to the original non-anonymised data may be beneficial, but is not a requirement.
- **User interface:** web platform, with links to the SECURED library tools.
- **Workflow:** [*Figure 37*] the user interacts with the platform following the series of steps below:
  - Data owner enters in the web-based anonymisation assessment user interface the parameters of the anonymised health data (such as electronic health records (EHR) and time-series data), and (optionally) uploads a small data sample (of non-sensitive test data). The actual data itself should not be shared to ensure privacy.
  - The web platform returns a user-readable known threat (i.e. re-identification attack) list, with user guidance, including the generality of applicability of the attack, and conditions required for the attack to be performed (e.g. access to external complementary data sources etc.). In addition, where the tools implementing the attacks are publicly available (i.e. released publicly with permissive licences, as well as those created in SECURED) a link to the implementation in the SECURED library is provided.

– The user can manually run the specific tools integrated in the library (see point above) locally, as the dataset is not uploaded to the platform (if the dataset is vulnerable to re-identification, this would constitute a data breach). Individual tools output de-anonymisation results against the user dataset. The web platform provides guidance on the interpretation of the results.

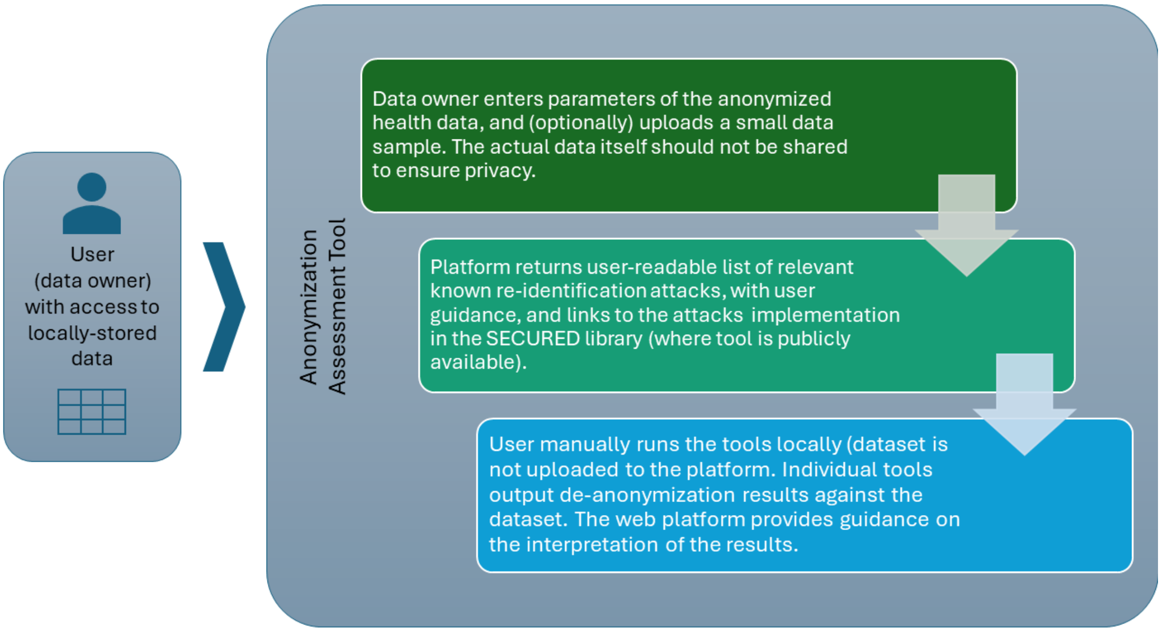


Figure 37 – Data providers and owners, such as doctors or researchers, can use this component for assessing the security of their anonymised health data against known threats, before sharing.

3.5.2.6 Anonymised Data Transformation toolset

The Anonymized Data Transformation (ADT) consists of four different tools working together: Anonymisation, Anonymisation Assessment (reidentification), bias assessment, and synthetic data generation, as shown in Figure 38. The main data flow starts with the application of the Bias Assessment toolkit to evaluate data bias. If bias is detected then the Unbiasing tool will be applied based on the previous assessment. Additionally, if bias is detected and can be corrected, the synthetic data generated by the Synthetic Data Generator tool can be used to increase the misrepresented population. This process can be performed by requesting new data or using what is already present in the data cache.

The unbiased dataset will then be anonymised by the Anonymisation tool applying the user's parameters. Once the dataset is anonymised, the Anonymisation Assessment tool will be applied by the data owner to identify relevant re-identification attacks, and, through manual local testing on the dataset, that is not uploaded to the platform, whether the resultant dataset is vulnerable to such attacks. In case the user considers that the vulnerabilities reported are feasible, or the anonymised dataset does not fulfill the requirements in terms of utility or privacy leakage, the data will be anonymised again based on new anonymisation parameters, and a new anonymisation assessment is performed.

Once the resultant dataset fulfills the user requirements, the data will be available for sharing. It must be noted though, that the anonymisation process will try to mitigate the privacy leakage but also to maintain data utility. Also, the depicted flow is performed mostly manually by the user as anonymisation and re-identification tools require interaction with the user. Finally it must also be stated that, as the tools are still work in progress, the final design might undergo small alterations, in order to fit better with the desired outcomes, but the overall data flow will remain the same.

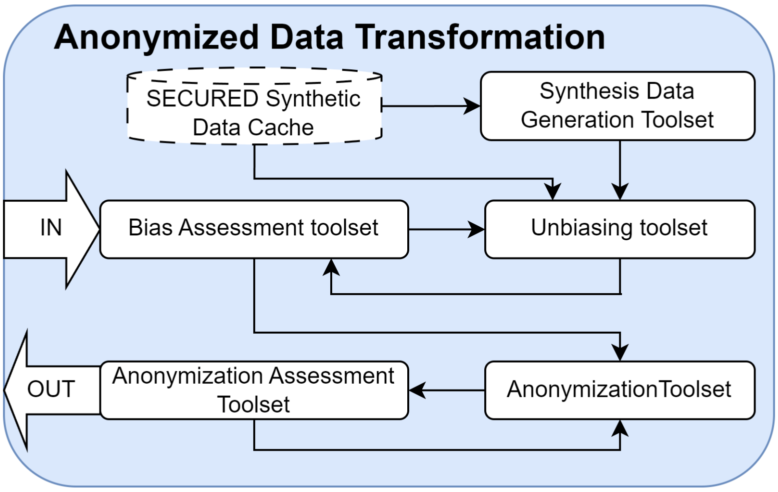


Figure 38 – Flow for the dataset anonymisation followed in the Anonymised Data Transformation Toolset.

3.5.3 Innohub Development Libraries

This Innohub subsection also hosts use-case-specific Secure Multi-Party Computation (SMPC) and Homomorphic Encryption (HE) applications, for further showcasing the inherent power of the developed system. Finally, the Innohub Tools section contains the SECURED libraries for the development and extension of the Federated Learning, SMPC, HE, and Anonymisation mechanisms, generated from the source code of each component, as developed in WP2 and WP3.

3.5.3.1 Privacy Preserving Federated Learning Development Library

The foundation of our privacy-preserving federated learning library will be built upon `Flower` [9], a widely used framework in the research community. We are currently considering an enhanced version known as `Pybiscus` [8] as the primary candidate. However, it is important to note that the final design may undergo minor adjustments as development progresses. This framework utilizes established solutions like `Secure Aggregation` to meet the unique demands of medical applications. By integrating Secure Multi-Party Computation (SMPC) and Homomorphic Encryption (HE) with Federated Learning (FL), we aim to offer additional customizable privacy features beyond those inherent to the federated learning protocol.

Furthermore, a private set intersection protocol is utilized in advance to remove any redundancy within the datasets amongst the clients. Lastly, we will implement privacy-preserving contribution evaluation techniques such as Leave-One-Out and Include-One-In, to assess the value of each participating client's contribution. These allow us to adjust the weight of client updates, hence optimizing the overall performance.

The exact data flow is depicted in Figure 39 as a sequence diagram. During the initialization phase, clients establish a shared secret using a key-exchange protocol, such as Diffie-Hellman, which is essential for the Secure Aggregation protocol. This step is followed by a Private Set Intersection protocol to identify and remove cross-client duplicate records. Subsequently, the client designated as the server initializes the model. It is important to note that these protocols are distributed and all communication is encrypted to ensure that no information is disclosed beyond the final results to the participants.

During training rounds, clients train the received models, which are then securely aggregated by the client acting as the server. The updated model is broadcast to the clients, who subsequently self-evaluate their contributions before the next round begins. Note that all communication is encrypted via TLS to protect against potential eavesdroppers, as the communication channel is not inherently secure.

[8] https://github.com/ThalesGroup/pybiscus/blob/main/README.md

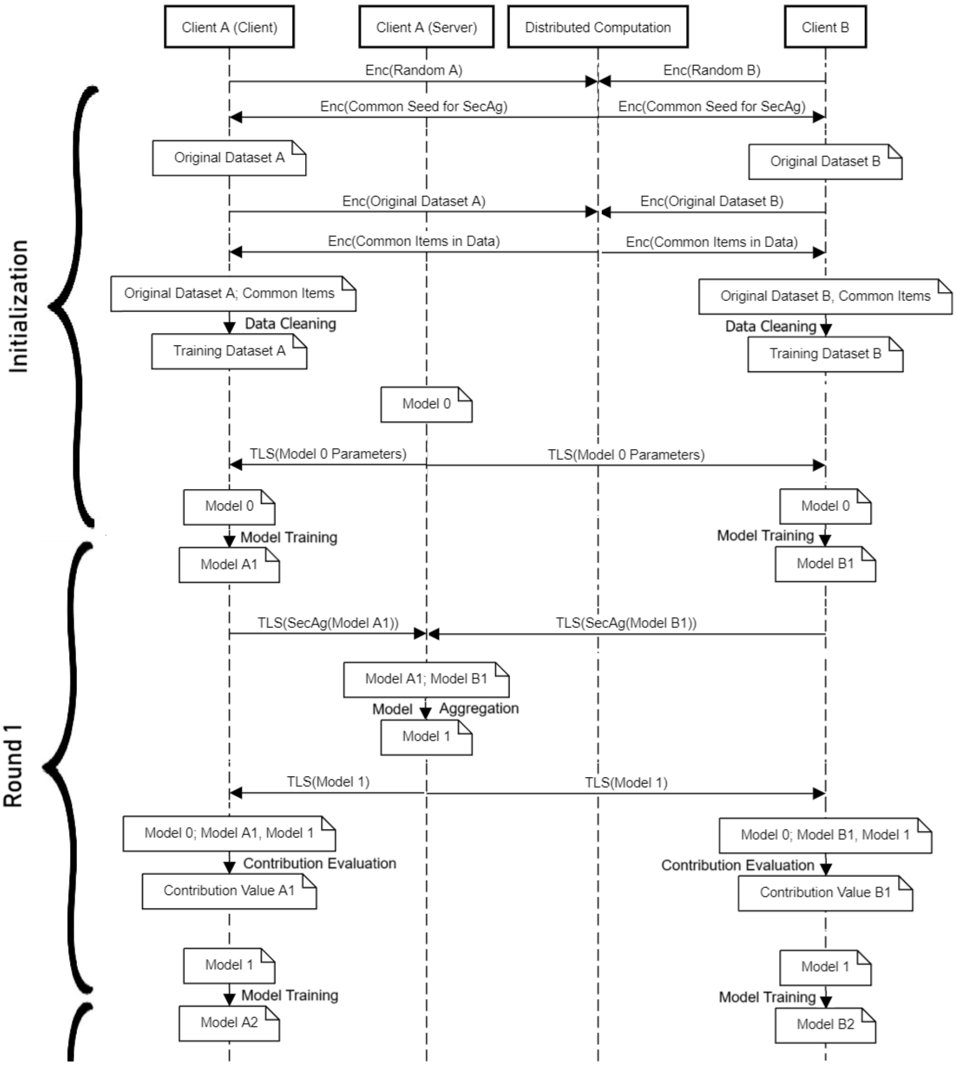


Figure 39 – Flow for the data within the FL framework.

3.5.3.2 Secure Multiparty Computation Software/Hardware Development Library

The Secure Multi-Party Computation (SMPC) library will leverage and extend and use existing solutions such as `MP-SPDZ` to suit the medical application landscape. One of the main contribution of the SECURED library of SMPC is the integration with the Federated Learning. As proven by previous works in the domain, combining multiparty computation with federated learning enables additional configurable privacy properties on top of what is already provided by federated learning alone. Within this landscape, users can specify their particular requirements whilst benefiting from the vast number of configuration options that come from a general purpose tool like SMPC.

The integration between SMPC and Federated Learning will be provided in the SECURED library as a set of functions that will instantiate the needed Federated Learning functionalities with the SMPC ones. Internally, this extension will use the same data structure of the libraries used as starting point. Since the SMPC library will be based on existing libraries, it is fundamental to ensure portability and forward compatibility of the added functionalies. The user that wants to use the SMPC and Federated Learning integration just needs to use the dedicated functions developed within SECURED, since the SMPC underlying functions and the Federated Learning support will be transparently instantiated by the dedicated SECURED functions.

3.5.3.3 Homomorphic Encryption Software/Hardware Development Library

The approach that is followed for the Homomorphic Encryption (HE) library is similar to the one carried out for the SMPC one, namely to start from existing libraries and extend them with hardware and software components. The starting libraries that have been selected as starting point are `OpenFHE` and `Concrete(-ML)`, since they provide state-of-the-art implementations of the most important HE schemes. These libraries have however limitations in performance and in the size of the input that can be processed. These limitation will be addressed in the SECURED library at different levels, including algorithmic level and architectural level.

At algorithmic level, it is worth highlighting the use of an advanced, data driven pre-processing. In a nutshell, the pre-processing step developed in SECURED includes and combine packing and decomposition, and it is applied to the data to be processed homomorphically before they are passed to the cryptographic functions. The pre-processing step transforms the data into a more HE-friendly representations that allow to improve the performance and allows to process data of larger sizes. The pre-processing functionalities, will be provided as set of functions and, together with the related documentation, will be integrated into the whole SECURED library. Users that want to exploit these functionalities, would have to apply the needed functions before applying the cryptographic functions. To ensure a wide adoption and an effort-less integration, the pre-processing functionalities will be using the same data structures of the cryptographic ones.

At architectural level, critical functions will be accelerated by means of specific hardware. Particular attention is devoted to the design of interfaces between the accelerated functions and the rest of the library. This is important because on the one side, it is crucial to ensure that the gain obtained by the acceleration is not overshadowed by the communication cost, on the other, the interface should ensure large flexibility to guarantee easy integration with libraries developed also beyond the SECURED project.

3.5.3.4 Anonymisation Software/Hardware Development Library [UvA/BSC]

The anonymisation library will group all the anonymisation functions developed within SECURED. In addition to the methods for anonymisation, one of the main contribution of the library is the definition and use of a common interface for all the anonymisation services. The interface will allow to easily swap between different methods as the interaction with data will be standardized using common formats.

The goals of the common interface are mostly two: The first is to provide a single and simple interface to instantiate all the anonymisation services developed in the SECURED project. The second is to allow an easy use of the library for further development, thus ensuring a large adoption of the SECURED results beyond the end of the project.

This library will also be adopted internally in SECURED. For example, this library will be used as part of the ADT to provide the interaction required between the Anonymisation, Re-identification, Unbiasing and Synthetic Data Generation tools.

4 SECURED Processes and User Interactions

Section 3 showcased the overall SECURED architecture detailing all tools, services, knowledge base and libraries. This section views the architecture from the scope of different types of users. In Section 4.1, we present the overall methodology used into identifying users and their interaction points. In Section 4.2 we identify the four user types outlining their responsibilities and requirements of the architecture and then map their activities within the SECURED architecture.

4.1 User, Roles and Interactions Methodology

As analytically described in Deliverable 4.1 "State of the Art and initial technical requirements"[1], in SECURED we follow the user journey and process mapping approach in order to identify requirements and to specify user interactions within the SECURED system in an effort to address the whole problem of specifying the characteristics and behavior of the system in a user-centric manner. The notion of User Journey (UJ) or Customer Journey (CJ) as the majority of the academia tends to call it, focuses on the entire user experience and is considered by an increasing majority of scholars [10] as the optimal method of putting together intuitive, easy to use platforms with the user at their center. The UJ approach is well-fitting for platform designers and framework architects that need to create platforms which interact with a broader set of active users. The approach is very much in-line with the conceptual objectives of SECURED Innohub that aims to create a privacy-related ecosystem.

As described in D4.1[1], the UJ approach includes several stages that include touchpoints, and personas [11, 12]. Eventually, as latest research indicates, the phases introduce strict contact points between users and services that allow a platform architect to structure an architecture and the interactions between components properly yet allowing a certain degree of freedom to adjust the details of those interactions during implementation. For the sake of completeness, the basic concepts that are considered behind the scenes of the UJ mapping are presented briefly below. The full description is provided in D4.1[1].

- **Stages:** They can be considered snapshots of a specific UJ that have conceptually a specific meaning for the user and they may contain the contact points, called also "touchpoints", between the user and the service as well as the generated responses in each contact point. A UJ may include one or more stages that may or may not be encapsulated and also it may include one or more user types, also called "personas" in them.
- **Touchpoints:** A touchpoint is a direct or indirect contact [13, 14] of a user with a specific service/component/tool delivered to him/her via online platforms or other methods of personal interactions [15, 16]. Users form an experience at each touchpoint [17], which is then aggregated into the UJ.
- **Personas:** Personas are descriptive models of archetypal users derived from user research. A Persona, also called in SECURED simply as user type, is a user category that characterizes users that all share similar goals, motivations and behaviors [18].

The goal of adopting the UJ approach in SECURED is, through depicting user behavior in a systematic manner, to introduce a User Journey Mapping (UJM) that will allow the project consortium to better understand the high level functionality of the SECURED Reference architecture and map the interactions of the various personas to the SECURED architectural components. To align with the SECURED scope, UJM elements defined earlier can be simplified and the UJ be focused on single, simple, Personas (from this point on described as User Roles) and touchpoints (considered and visualized as interaction points between the distinct architectural components of the SECURED). We consider that each user type (a persona) is primarily associated with a single stage, however during the analysis we have identified some common stages that may be used regardless of the involved user type. Those common stages (also called common processes in this deliverable) are encapsulated to the user type specific stages.

Given the above clarifications, in this deliverable, in contrast to the abstraction existing in D4.1[1], we provide a significantly more analytic view of the UJ approach outcomes that include a sequence diagram type of visual representation of the overall user type experience showing how each type interacts directly and indirectly with the SECURED architecture components. As described in D4.1[1], the UJM that is followed consists of several steps:

- **Step 1: Set clear objectives for the UJM** Initially, a clear series of objectives on what needs to be achieved through a UJM needs to be provided. This will allow the identification of problems the system designer needs to solve, as well as ease up the result extraction process.
- **Step 2: Identify users and define their actual goals.** In this step we specify the user types, describe their basic characteristics that constitute their identification points. Also, at this step we identify the main goals of each user type and how they aim to achieve them through SECURED.
- **Step 3: Identify all possible user touchpoints.** Having identifying User types allow the UJM process to specify the touchpoints of each user type since those may differ for each user type and his/her interaction with the components of the SECURED architecture.
- **Step 4: Identify user actions for every stage of the journey.** This step revolves around user type actions and specifically what is a user type doing in each step of a predefined path inside the overall user journey.
- **Step 5: Identify potential changes which may compromise the overall flow, technical obstacles, or pain points.** In this step we aim to identify whether users may run into problems during a certain stage/process. Such problems may hinder the user experience and lead to a problematic UJ.
- **Step 6: Identify opportunities for improvement.** Identify possible specific issues, bugs, experience pain points that can be improved in both the short and long run.

Regarding step 1, in SECURED the goal of the UJM is specific and is provided by the corresponding task (T4.1) description as well as the GA document in general. We aim to identify the key user types of the SECURED Innohub and map their touchpoints to the SECURED architecture in a clear manner that will facilitate the implementation and integration of the SECURED solution into a realistic prototype. In the following subsections we provide the outcomes of the above described six-step UJM process for SECURED.

4.2 User Types

By adapting the User Journey Method to the practical needs of the SECURED Project, it was possible to identify a number of different users and their intended use of the SECURED solution. This document identifies four specific types of users (UJ personas) as seen in Table 5 and discussed in the following subsections.

Table 5 – Use cases related to Anonymisation Tool

#	Type	Short Description and Main Goals
1	End user	Use of tools/services for non-development purposes.
2	Model Developers	Develop new model from scratch or download and enhance existing anonymised model.
3	Privacy Preserving application Developers	Develop new/existing solution enhanced with PET SECURED libraries, use the SMPC/HE library to setup some PET computation including the establishment of a server/client model or use the Anonymisation/Unbias and/or FL libraries to handle PET application data
4	Data Developer	Develop new unbiased, anonymised dataset from scratch or synthetically generated data and register to SECURED Innohub or enhance existing dataset with anonymisation/unbiased characteristics and register to SECURED Innohub

4.2.1 End User

The first user type is the End User. The primary objective for the End User is to utilize the SECURED tools and services to develop products for non-development purposes. The End User is essentially a SECURED consumer and requires the use of one or more tools and services, as is, of the SECURED Innohub. The End User downloads and installs tools on their premise and does not register anything to SECURED Innohub. Examples of such users are the SECURED use-case pilots.

4.2.2 Model Developers

The next user type is the Model Developer. The primary objectives for the Model Developer is either to develop a new AI/ML model from scratch or download and enhance an existing anonymised model that has been developed using SECURED tools and services. The Model Developer may require the Anonymised SECURED Innohub AI models for use in Federated Learning or Deep Learning applications, outside of SECURED control or engagement. The final product, the model, must be registered to the SECURED Innohub.

4.2.3 Privacy Preserving Application Developers

The third type of user is the Privacy-Preserving Application Developers. The primary objectives for this user type is to either develop new or existing solution enhanced with PET SECURED libraries, use the SMPC/HE library to setup some PET computation including the establishment of a server/client model or uses the Anonymisation/Unbias and/or FL libraries to handle PET application data. The Privacy Preserving Application Developer requires the SECURED libraries and/or SECURED core services, such as synthetic data generation, for developing privacy preserving application. The final product, the application, must be registered to the SECURED Innohub.

4.2.4 Data Developer

The final user type is the Data Developer. The primary objective of the Data Developer is to either develop a new unbiased, anonymised dataset, from scratch or synthetically generated data, and register to SECURED Innohub, or enhance an existing dataset with anonymisation/unbiased characteristics and register to SECURED Innohub. The Data Developer wants to register a new dataset to be used by SECURED ecosystem first by downloading or running the Anonymisation Data transformation service, which generates an anonymised version of his/her dataset and registers this dataset to the SECURED hub.

4.3 User Journeys

The User Journeys (UJs) defined in this section provide a high level view of a User's interaction with the SECURED architecture throughout the Innohub. We have involved in such UJ the most critical Reference Architecture components and we have grouped, for the sake of simplicity, as one component all the Innohub tools and as one component all the Innohub services. For more details on each individual component and its characteristics as well as more detailed interactions of the User within each component the reader is referred to Section 3 where such information is provided.

4.3.1 Common Processes-Stages

The SECURED architecture may provide two different approaches for providing its functionalities to the users, mainly functionalities as tools and functionalities as services. Tools are functionalities that can be downloaded and run offline at the user's premises whilst services can be used online and are hosted by the SECURED Innohub. In both cases, the interaction of user types with some tools or services may be the same hence we can consider such group of interactions as common processes (common UJ stages) that we can present only once and then encapsulate as black boxes in other processes/stages per user type. The following subsections outlines the two different viewpoints from the perspective of the user, one for tools and one for services for the same functionality where it is applicable, that are irrespective of the user types hence common for all user types.

4.3.1.1 Synthetic Data Generation process

The Synthetic Data Generator (SDG) of SECURED can be instantiated both as a tool and as a service. It has its own internal tools and a dashboard as well as an API allowing it to have both tool and service type of interaction with all possible user types. In the following paragraphs and Figures 40 and 41 we provide sequence diagrams as a result of the SDG UJs.

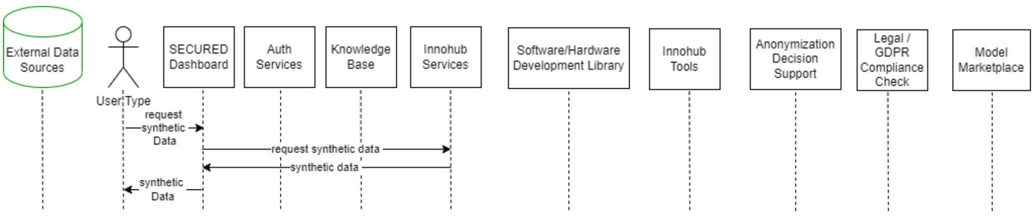


Figure 40 – Common Process for Synthetic Data Generation as Service

As shown in Figure 40, when SDG is utilized as a service it has a simple interaction with the user. The user requests some synthetically generated data from the SECURED Dashboard by providing a series of health data characteristics/configuration. The dashboard, through the back-end orchestrator, calls the SDG service, uses the relevant API to start the service and feeds the configuration input. The SDG service results are then provided to the dashboard from which the user can download them. Depending on the final implementation of the service there may be several intermediate interactions with the user in order to properly setup the service.

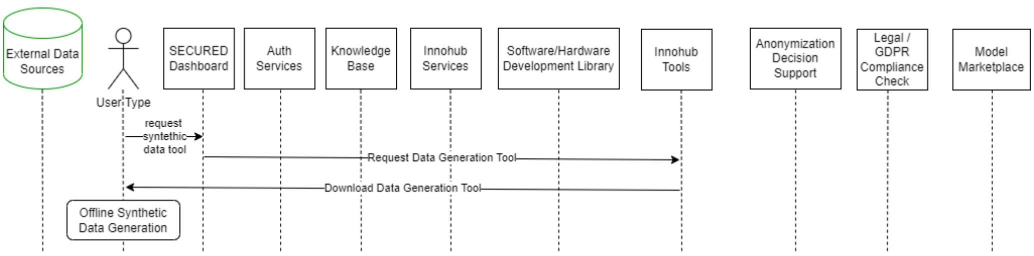


Figure 41 – Common Process for Synthetic Data Generation as Tool

As shown in Figure 41, when SDG is utilized as a tool, the user interacts in the same way as all SECURED tools. The user requests the tool to be downloaded from the SECURED Dashboard, then the request is forwarded through the orchestrator backend to the Innohub Tools that reside in the Knowledge base toolbox repository, which is used in order to allow the user to download the tool and perform synthetic data generation on the user premises offline.

4.3.1.2 Bias Assessment/Unbias process

The Bias Assessment and Unbiasing of SECURED can be instantiated as both a tool and a service. The corresponding common process thus can have two viewpoints (Service or Tool) and consist of a utilization of both services (or tools). The user provides a dataset that wants to assess its bias level, and if bias is discovered then proceeds to unbias the dataset. The Unbiasing process may require the use of synthetic data generation, thus it may utilize the SDG service or tool. After the Unbiasing is finished the dataset is rechecked for bias, using the bias assessment service or tool. The process finishes when bias is no longer found or is trivial in the dataset. The common process is presented in Figure 42 for Bias Assessment/Unbiasing services and in Figure 43 for Bias Assessment/Unbiasing tools. Note that in Figure 43 we provide an optional interaction with the SDG service but alternatively the process may include the SDG tool in the sequence (not visible in the figure).

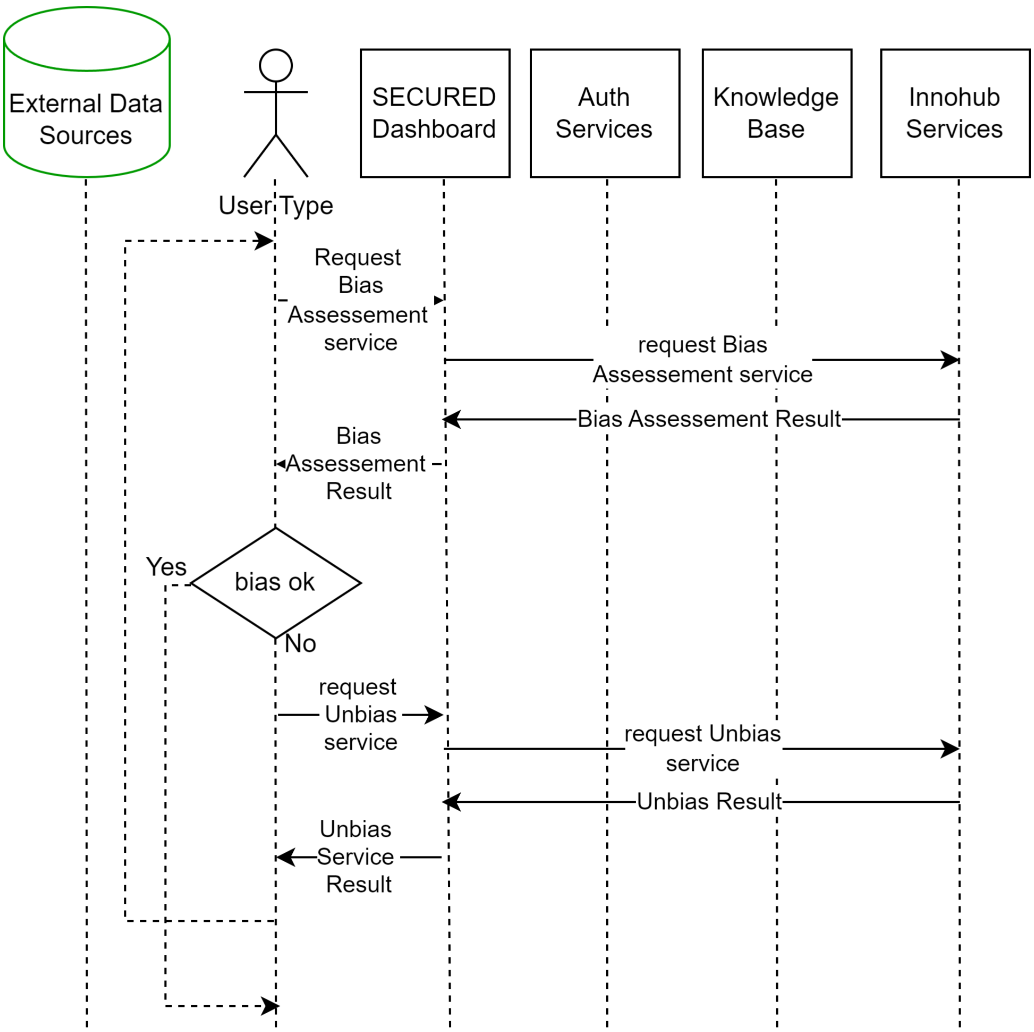


Figure 42 – Common Process for Bias Assessment and Unbiasing as Service

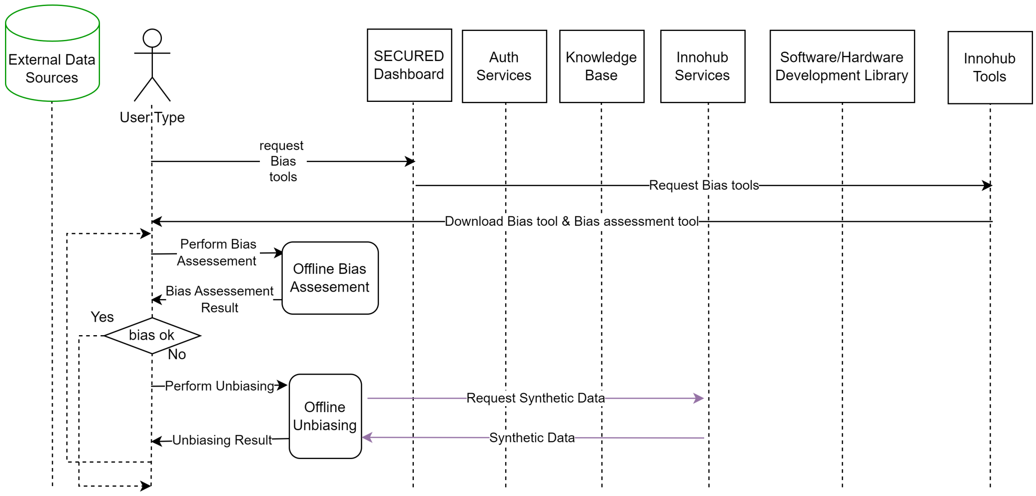


Figure 43 – Common Process for Bias Assessment and Unbiasing as Tool

4.3.1.3 Anonymisation process

The Anonymisation process involves the Anonymisation and Anonymisation Assessment tools since such tools do not have corresponding services. Based on the SECURED use-case specifications and relevant user requirements as well as GDPR compliance reasons there is no need to upload a private dataset to SECURED in order to Anonymise or Assess the anonymity using an online service. The common process for anonymisation is presented in Figure 44 and includes the optional usage of the Platform services and the mandatory usage of the Anonymisation and Anonymisation Assessment tools. More specifically, initially the user requests the anonymisation tool and the anonymisation assessment tool from the SECURED Innohub by placing a request to the Dashboard which provides a download link of the Innohub tools stored in the Knowledge Base toolbox repository. The Anonymisation Decision Support service will provide tailored anonymisation guidelines that the user can then use to configure the anonymisation tool. If the user wants to check the anonymisation of an existing anonymised dataset then it follows the exact flow of Figure 44. This includes initially the use of the Anonymisation assessement tool, an evaluation of the tool result that can trigger a re-anonymisation in order to check the anonymisation strength and possibly increase it. The user in order to properly anonymise/reanonymise the dataset may request anonymisation support through the dashboard by providing some dataset characteristics.

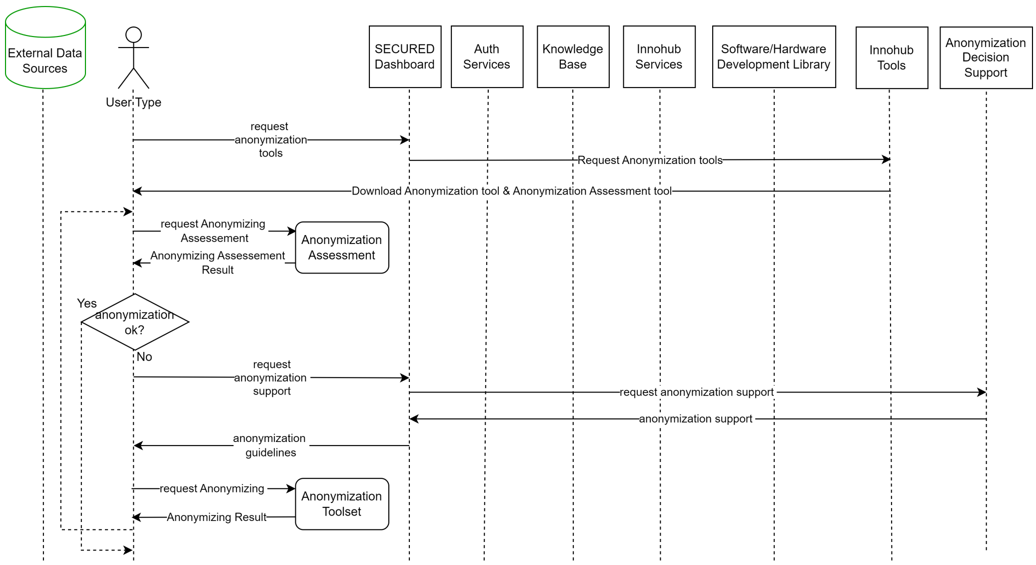


Figure 44 – Common Process for Anonymisation & Anonymisation Assessment Toolset

In the case where the user has a dataset that he/she wants to anonymise and then check the anonymisation strength by executing the anonymisation assessment, the process that is followed can be seen in Figure 45. More specifically, the user, after requesting the anonymisation and anonymisation assessment tools and downloading them, requests anonymisation guidelines to properly configure the anonymisation tool. The results of the tool are forwarded to the anonymisation tool, which produces an anonymised dataset, and then feeds the anonymised data set to the anonymisation assessment tool. If the result of the assessment shows that the dataset has strong anonymisation, the process is finished; otherwise, new guidelines are requested and provided, and the anonymisation tool is executed again, thus repeating the whole process.

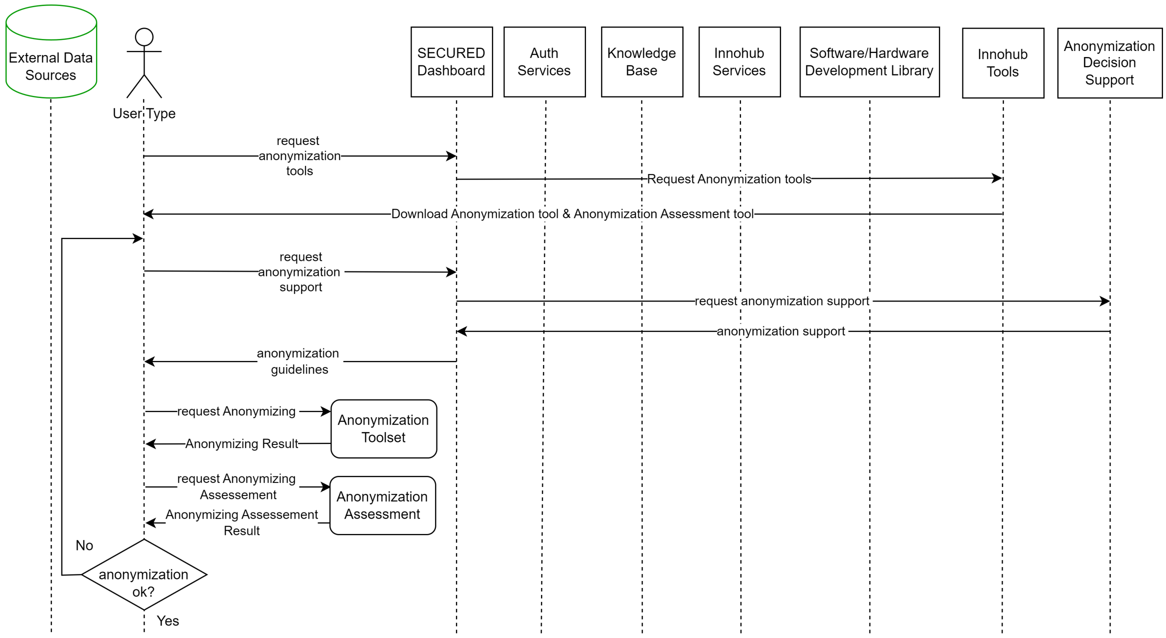


Figure 45 – Alternative Common Process for Anonymisation & Anonymisation Assessment Toolset

4.3.1.4 Anonymized Data Transformation (ADT) toolset process

The ADT toolset constitutes an advanced tool offered by the SECURED innohub that can get as input some dataset and provide as output an anonymised version of that dataset with guarantees that it has been checked for strong anonymity and for not having bias, with the bias removed if found. To achieve these goals the ADT process orchestrates all the anonymisation related tools of SECURED as is described in more detail in sub-subsection/paragraph 3.5.2.6. The usage of the tool includes a series of steps that are followed regardless of the user type hence it can be considered a common process.

As shown in Figure 46, a registered user initially requests through the Dashboard some anonymisation guidelines by utilizing the Anonymisation Decision Support service, by providing a series of inputs to the SECURED Dashboard. Then the Dashboard through the Back-end orchestrator transfers the input to the Anonymisation Decision Support service that provides user-specific (personalized) anonymisation guidelines as results. Such guidelines can be used by the user in order to optimally configure the ADT toolset. After acquiring anonymisation guidelines by the SECURED Innohub, the user requests to download the ADT Toolset. This action is performed through the Dashboard that forwards the request to the Innohub tool repository in the Knowledge base and allows the download to take place. When the user installs the ADT on premise, outside the SECURED Innohub platform, he/she provides proper configuration, based on the Innohub provided guidelines and initiates the anonymisation process of a given dataset, given as input to the toolset.

The ADT Toolset in action utilizes the Anonymisation and Anonymisation Assessment tools, the Bias Assessment and Unbiasing Tools and potentially the SDG service or tool. Initially, the toolset performs a Bias Assessment on the Dataset to be anonymised and if bias is discovered the Unbiasing tool is executed. If bias is considered trivial the anonymisation operation can begin directly without Unbiasing. The output dataset of this operation is rechecked for bias. In case bias is still not trivial after unbiasing the Unbiasing tool is executed again with different and possibly more strict parameters. This procedure is repeated until bias is considered trivial. In such case, the unbiased dataset is anonymised using the Anonymisation tool. The anonymised dataset is then assessed for anonymisation strength, i.e., deanonymisation techniques are applied on the dataset and success rate (e.g., accuracy of results) is provided. Based on the anonymisation assessment result, the user decides if the anonymised dataset fulfill her/his privacy-preserving requirements. In such a case the user considers the process finished, otherwise the process is rerun with different configuration and the result is checked repeatedly till the user deems the anonymisation process finished.

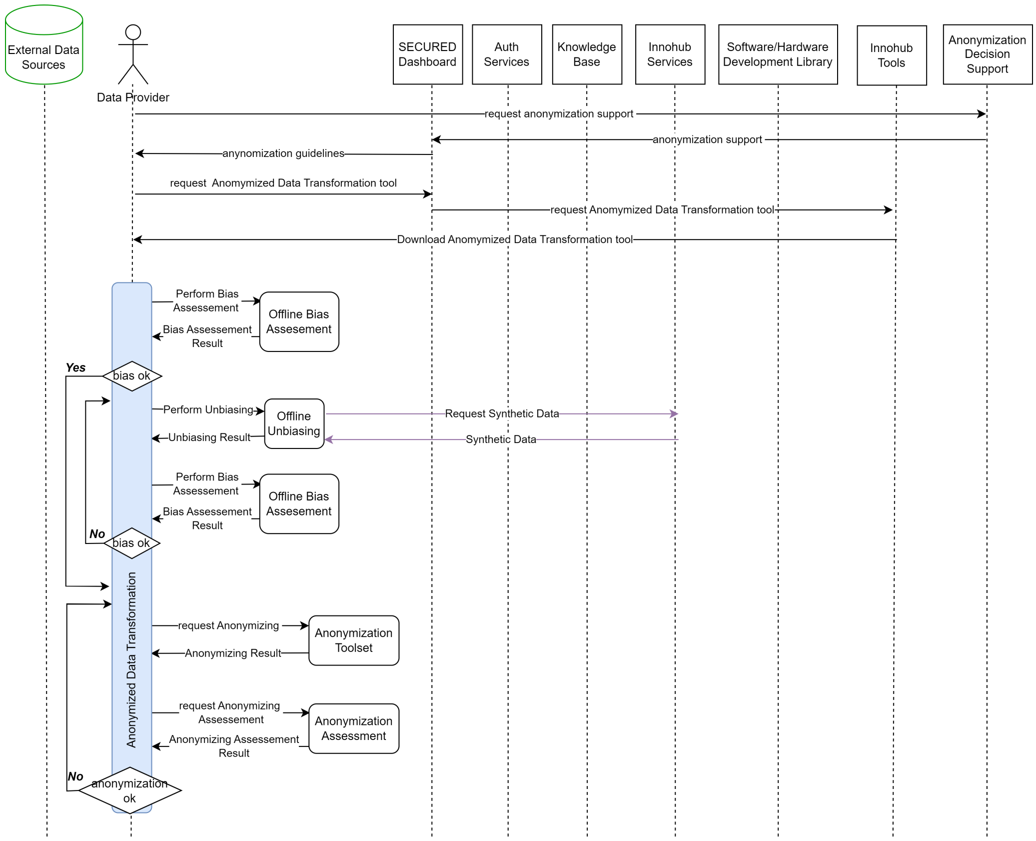


Figure 46 – Common Process for the Anonymized Data Transformation Toolset

4.3.2 End User UJ

As discussed in Section 4.2.1, the main purpose of the End User is to utilize the Innohub tools and services to create non development products. As such, initially, the end user signs up to SECURED and creates a new user account in the Innohub, if the user has no record, he/she must first register by providing some initial personal information and log-in credentials through the dashboard. This request is forwarded to the Auth services which generate the required secure credentials which are provided to the Dashboard each time the user logs in to perform authentication. The sign-up procedure is performed only once and from this point on the end user can log-in using the credential that he/she originally provided during sign-up to the system, e.g., a username and a password. Thus, an end user becomes registered on the SECURED platform and can access all tools and services as discussed in Section 3.2.2 and depicted in Figure 11. After registration / log-in, the end user can do one of the following (also presented in Figure 47):

- Use the Anonymisation Decision Support service through the SECURED Dashboard by providing a series of inputs. Then the Dashboard through the Back-end orchestrator transfers the input to the Anonymisation Decision Support service that provides user-specific (personalized) anonymisation suggestions as results. Those results reach the end user through the SECURED Dashboard.
- Use the Legal/GDPR Compliance check service through the SECURED Dashboard by providing a series of inputs. Then the Dashboard through the Back-end orchestrator transfers the input to the Legal/GDPR

Compliance Check service that provides to the end user relevant to the user input legal framework as results. Those results reach the end user through the SECURED Dashboard.

- Request a Privacy-Preserving service from the SECURED Innohub through the SECURED Dashboard. From the list of offered Privacy Preserving services, the end user can choose his/her preference, access the service, and see the results through the SECURED Dashboard.
- Download a SECURED Innohub tool from the tool list that is visualized in the dashboard.

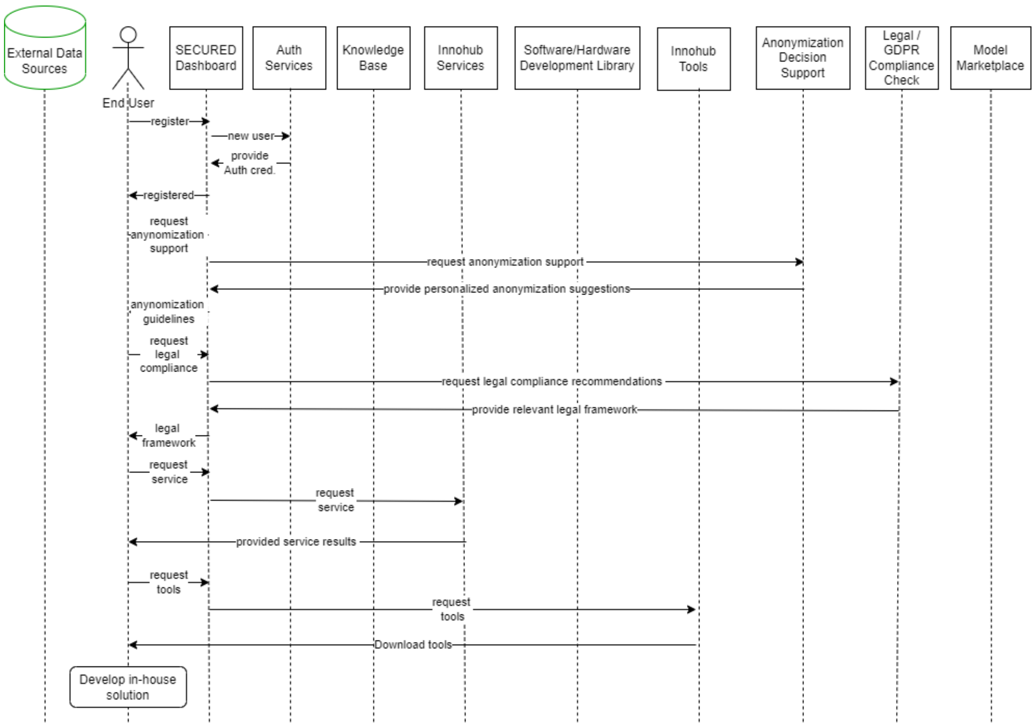


Figure 47 – User Journey for the End User.

4.3.3 Model Developer UJ

The model developer user type connects to the SECURED Innohub, as shown in Figure 48. In order to use an existing Innohub ML/DL model for some health-related application developed in-house, which is considered out-of-scope for SECURED, that may or may not involve ML/DL model retraining. Alternatively, the model developer is creating/developing/training a new ML/DL model using datasets generated or anonymised through the SECURED Innohub and wishes to make such models available to the SECURED community by registering them in the Innohub and uploading them through the Innohub model market place service.

As with all user types, the process starts by signing-up to the SECURED system. This process is described in sub-subsection 4.3.2 and is the same for all user types. When a model developer is registered and has logged in to the SECURED system, i.e., the Dashboard, given that he/she wants to use existing Innohub ML/DL models, requests a list of the existing models in the Innohub Model Marketplace. The service will contact the Knowledge base to get the list of models and their metadata and forward the available list to the Dashboard where they are visualized for the model developer. The model developer then requests that legal requirements/regulation status of some model and also requests a legal compliance check on the model he/she wishes to use in order to determine if legally (based on the national or EU regulations) is allowed to use it. The request goes to the Legal/GDPR Compliance Check service through the Back-end orchestrator component and the service provides a reply that is visualized through the Dashboard.

Assuming that the model developer complies with the provided legal regulations, this type of user makes a request to download and use a certain Innohub model. This request is sent from the Dashboard through the

Back-end orchestrator to the Model Marketplace service. The service retrieves the model from the Knowledge base and forwards it to the Dashboard and eventually to the user that downloads it. The UJ for the model developer of existing Innohub models may end at this point. However, if the model developer wants to extend the model, e.g., by using the Innohub tools or services to extend it, the UJ has several more steps as shown in Figure 49. To accommodate the model retraining capability using the Innohub, the model developer can initiate one or more of the common processes described in subsection 4.3.1. More specifically, the model developer user type can use the SDG process to generate new synthetic data for retraining, can anonymise synthetic data or private-sensitive health datasets (that are collected without the SECURED involvement) using the Anonymisation common process and/or check for bias and perform unbiasing on the datasets using the Bias Assessment/Unbias common process. Apart from that, the model user may request access to the data registry, in order to retrieve private datasets that are registered through the Innohub and get information on the registered datasets through the Dashboard. Then the model developer can contact the dataset owner and through a bilateral agreement get such datasets for retraining (out of scope for SECURED).

It should be noted though that such datasets can still be anonymised using the SECURED Innohub tools if they are not already anonymised by their owner. After all or some (depending on the model developer intentions and capabilities) processes have been performed, the model developer can retrain an Innohub ML/DL model and use it as he/she sees fit. Finally, the model developer can register the new retrained model to the Innohub Model Marketplace. To do that, the user registers the model and uploads it to the Dashboard and through the Back-end orchestrator this action is forwarded to the Model Marketplace service. The service then uses the Knowledge base API to register and upload the model there and when this action is completed provides to the Dashboard a receipt that the model has been properly registered to the system. All the above activities are visualized in detail in Figure 49

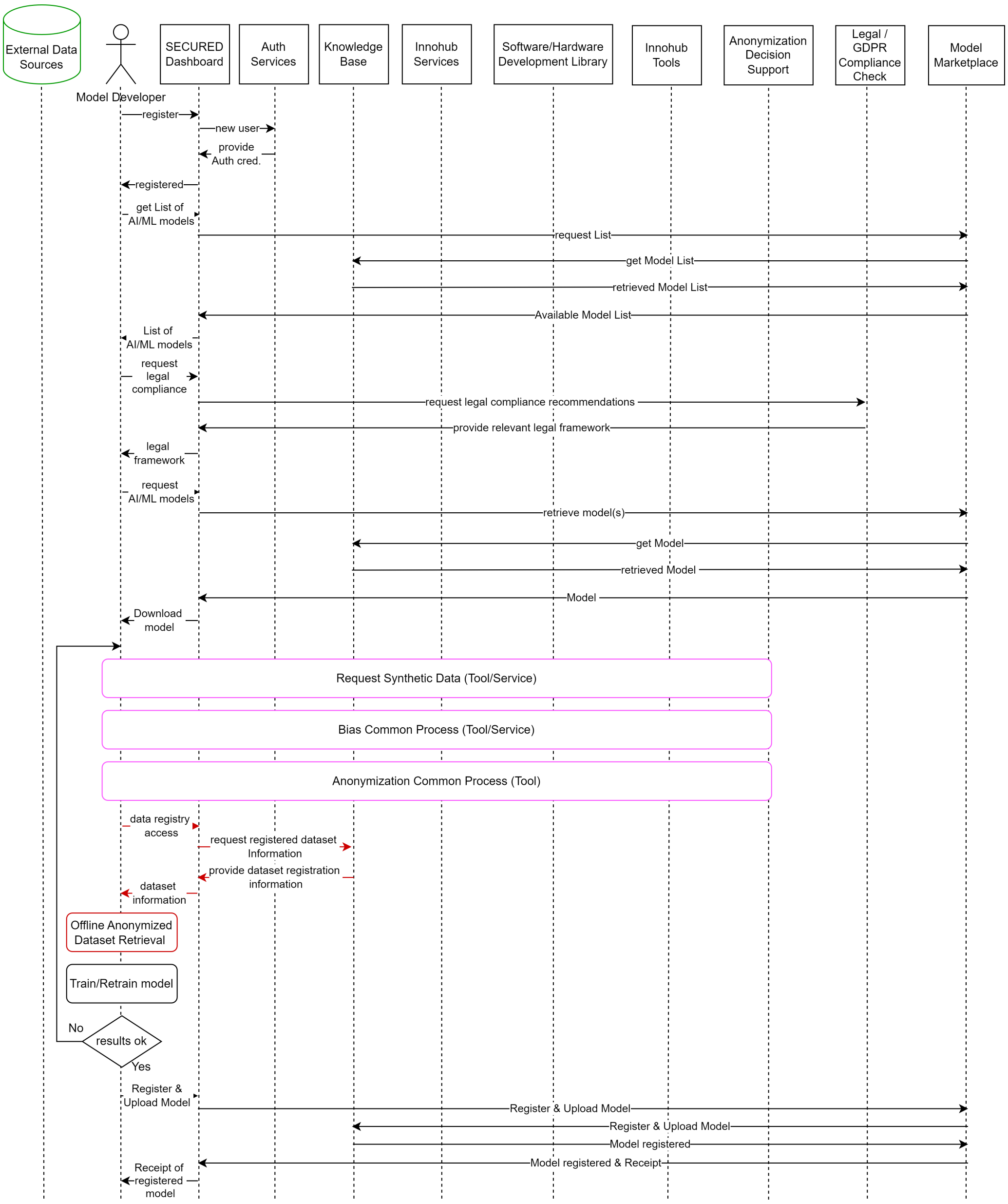


Figure 48 – User Journey for the Model User or Developer for the use of an existing SECURED model.

In a variation of the above UJ, the model developer may just want to use Innohub datasets or tools/services for generating them in order to train a new ML/DL model. In such case, shown in Figure 48, the initial steps of the model developer user journey related to retrieving available models and downloading them are omitted and the new UJ starts when the user asks if he/she has legal compliance to create a model using the Innohub datasets/services/tools. As seen in the Figure 48, after this legal compliance check the UJ actions for the model developer are the same as those in Figure 49.

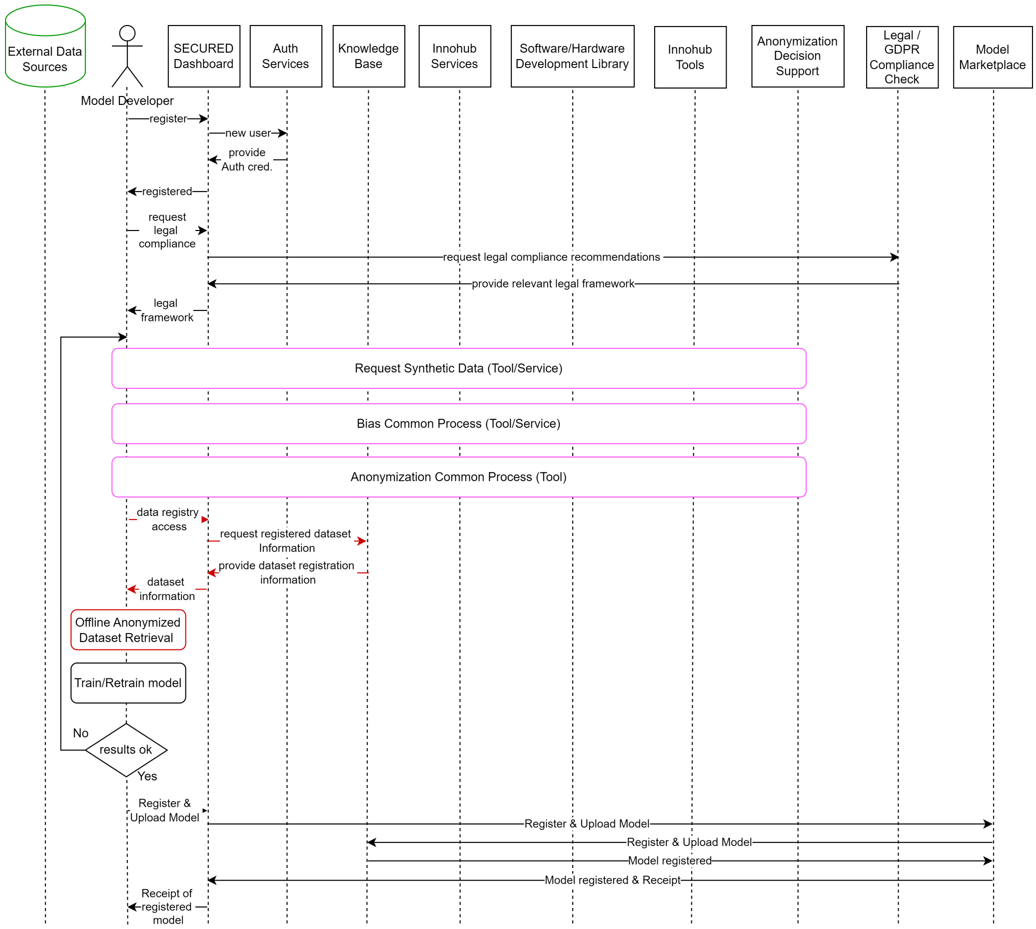


Figure 49 – User Journey for the Model User or Developer for the development of a new SECURED model.

4.3.4 Privacy Preserving Application Developers

The Privacy Preserving Application Developer user type is able to utilize a significant part of the SECURED Innohub capabilities that includes the services, the tools and also the development software/hardware libraries of the project. Given the various possibilities that such a developer may explore using SECURED to develop some health data related application, the Privacy Preserving Application developer UJ may have various different optional paths that are related to the developers need to create, use or enhance a SECURED Innohub ML/DL model, to anonymise and utilize datasets or to synthetically generate data and to assess bias and anonymity. The main mandatory action of the Privacy Preserving Application Developer UJ is the request to download and use one or more Software/Hardware Development libraries and eventually use them to produce new code for some application as seen in the initial steps of Figure 50. The non-mandatory steps shown in the Figure50, are practically the same as those of the model developer UJ (see Figure 49).

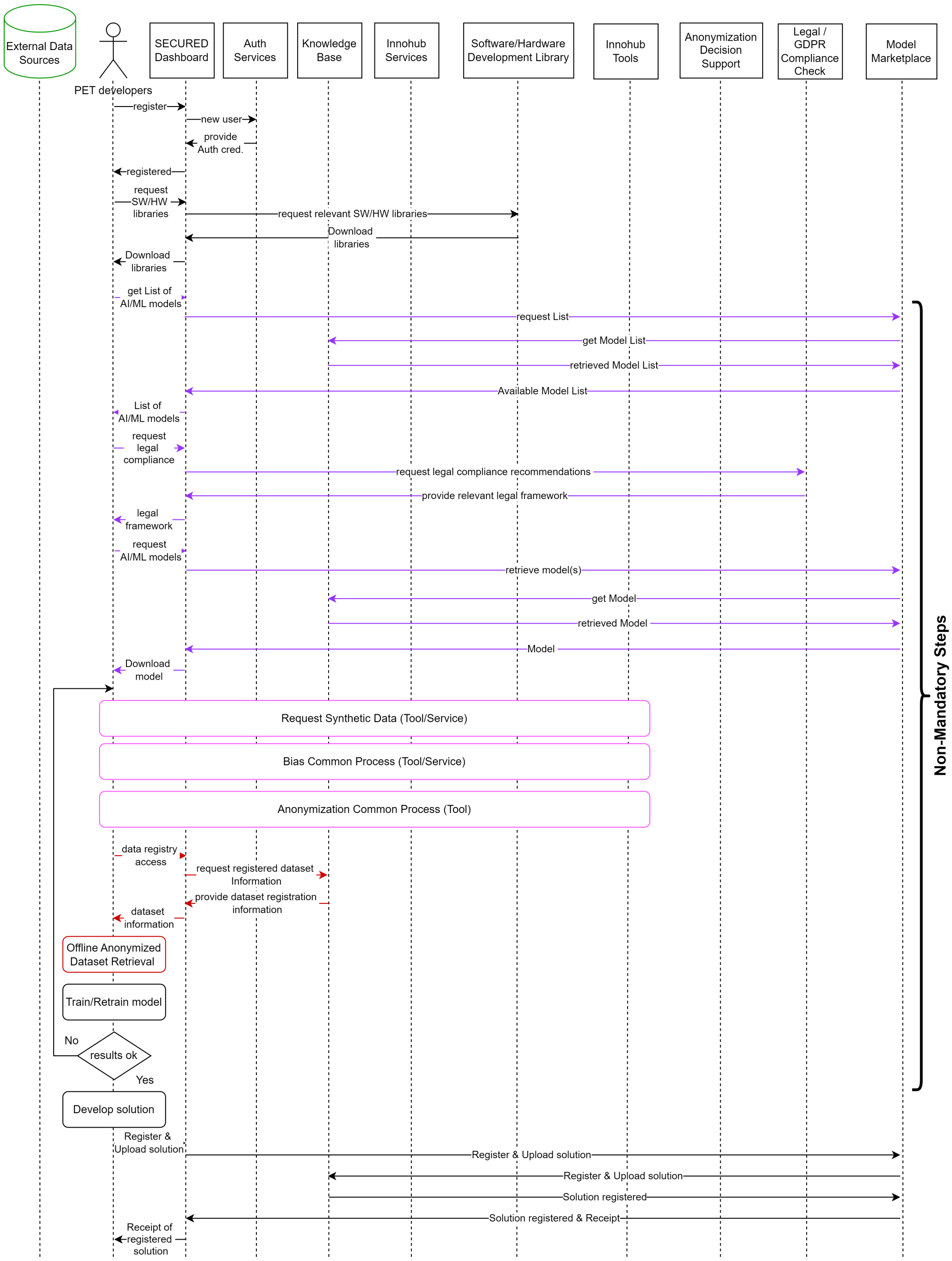


Figure 50 – User Journey for the Privacy Preserving Application Developer.

In short, the Privacy Preserving Application Developer after downloading the SECURED Innohub development libraries he/she uses the steps that exist in the model developer UJ to get an existing Innohub model from the Model Marketplace that can be retrained enhanced with additional datasets that can be generated or anonymised using the SECURED common processes. Similar actions can be followed if the Privacy Preserving Application Developer trains his own ML/DL model using SECURED anonymised or/and synthetically generated datasets (see the model developer UJ for detailed description of the steps). After having the needed trained ML/DL models a Privacy Preserving Application Developer can use them in a FL setup or individually to perform prediction or classification. In case, of course, the developed application does not require ML/DL models as in some pilot scenarios that require SMPC or HE coding for statistics analysis. Eventually, the Privacy Preserving Application Developer UJ is concluding by registering the developed solution to the Innohub. When this is concluded a receipt is provided to the user through the dashboard.

4.3.5 Data Developer

The final user type, the Data Developer, is mostly focused on generating new Datasets for training or educational purposes, such as synthetically generated health datasets for medical students to train on, or new datasets to be used by SECURED Model Developers.

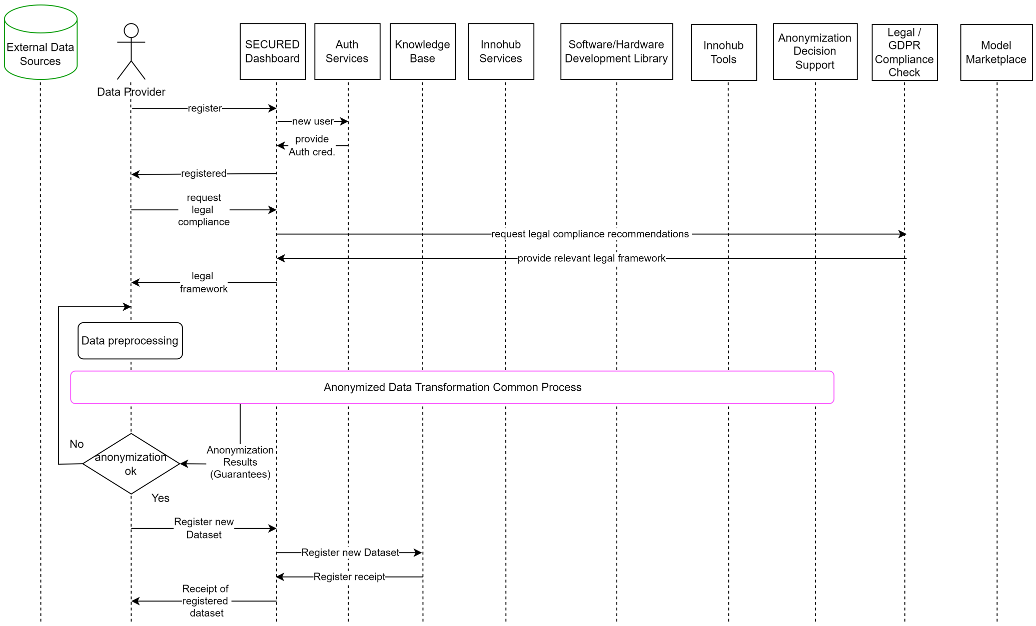


Figure 51 – User Journey for the Data Developer.

As in all user types, the Data Developer signs up, or logs in to the SECURED Innohub. Since the Data Developer aims to generate a new Dataset, it is imperative to be aware of the relevant legal process and regulations for such a dataset, hence the Data Developer initially requests a legal regulation compliance check from the Legal/GDPR Compliance Check service of the Innohub. The Legal/GDPR Compliance Check service provides a result that is visualized in the Dashboard, based on user input on it.

When having the relevant legal framework available, the data developer can start collecting and preprocessing data on premise. It is assumed that private health related data are involved which cannot be shared with any third party and therefore any process on such data prior to anonymisation must occur on the user's premises. After this action is completed, the Anonymised Data Transformation Common Process can be used, in the way described in sub-section 4.3.1.4. Eventually, through the ADT tooset the data developer's datasets are anonymised and unbiased without being able to be deanonymised. Apart from the anonymised dataset, the ADT also provides anonymisation guarantees that the overall anonymisation and unbiasing procedure has been performed properly. If the data developer user deems that the guarantees are not sufficient, then the data preprocessing and ADT common process are repeated. When the provided guarantees are considered sufficient, the process is finished and a new anonymised dataset developed by this user type can be registered through the Dashboard to the SECURED Knowledge base. An overview of the data developer UJ is shown in Figure 51.

5 Technical Specifications and Interconnections of SECURED Architecture Components

This section outlines the technical specifications of each component of the Reference Architecture as described in detail in Section 3. We also take into account the UJ that have been described in Section 4.

The collection of technical specifications as well as the interfaces and interconnections was a group process that involved all partners of the consortium. Tables 6 and 7 were provided to all partners and required to be filled for every component (tool or service or engine etc.) described in Section 3 for which each partner is responsible for. In Tables 6 and 7, we also provide instructions on what each field of the Table means and how it should be filled by each partner. The functional and non-functional requirements of each tool, component, and engine that have been described in D4.1[1] were used to complete the tables reported in this Section; the requirements also appear in this document for completeness in Appendix A. The completed Tables for each component/sub-component of the Reference Architecture described in Section 3, are presented in the following subsections. More specifically, subsection 5.1 contains tables that report the Technical Specifications of the components, while subsection 5.2 lists the interfaces and interconnections of the components.

Table 6 – SECURED Component Technical Specifications Template

Main functional Requirements	For the specific component/engine please provide the main functional requirements as defined in D4.1[1]
Non functional Requirements	For the specific component/engine please provide the non functional requirements as defined in D4.1[1]
Development Environment	Please specify the development environment and the programming language to be used
Software Requirements	Please specify any SW requirements or dependencies
Hardware Requirements	Please specify the minimum HW required for the best functionality of the component
Communications	Please indicate specific communication requirements between inputs, outputs or between submodules
Integration Requirements	Please indicate any specific integration requirements.
Deployment Requirements	Please indicate any specific deployment requirements
Security Requirements	Please indicate any specific security requirements
Privacy Requirements	Please indicate any specific privacy requirements
Critical Factors	Please describe any critical factors that might affect the development or functionality of the component
Containerisation	Please indicate whether this tool/component requires/will be deployed in any/and which containerised framework

Table 7 – SECURED Components Interfaces/Interconnections Template

Main Inputs	Please specify the main inputs
Input Data from Partner	Please specify from which partner/component data will be used as input
Nature of Expected Input	Please indicate the input format that your component would expect (e.g. JSON, image files, ect)
Related Scenarios	Please indicate the use case scenarios requiring this data
Interfaces	Please indicate the connection interfaces - APIs
Triggered by	Please indicate the events or conditions that trigger the component's functionality
Main Outputs	Please specify the main outputs
Output Data to Partner	Please specify to which partner/component data will be sent as output of this tool/component
Nature of Expected Output	Please indicate the output format that your component would be expected to produce (e.g. JSON, image files, ect)
Related Scenarios	Please indicate the use case scenarios requiring this data
Interfaces	Please indicate the connection interfaces - APIs

5.1 Technical Specifications

5.1.1 SECURED Front End

Main functional Requirements	REQ-PLAT-PORT-M-01, REQ-PLAT-PRIV-M-09, REQ-PLAT-SEC-D-11, REQ-PLAT-SEC-M-12, REQ-PLAT-SEC-M-13, REQ-PLAT-SEC-D-14, REQ-PLAT-SEC-M-15, REQ-PLAT-SEC-M-16, REQ-PLAT-SEC-P-17, REQ-PLAT-COMP-M-21, REQ-PLAT-MAINT-M-27,REQ-PLAT-MAINT-D-28, REQ-PLAT-MAINT-D-29 REQ-PLAT-MAINT-M-30, REQ-PLAT-MAINT-M-31, REQ-PLAT-MAINT-M-32
Non functional Requirements	REQ-PLAT-SEC-M-07, REQ-PLAT-PRIV-M-15
Development Environment	React/Next.js, FastAPI, Javascript, Typescript
Software Requirements	HTTP communication/certificates, Web Server (e.g., Apache, Nginx, etc.)
Hardware Requirements	2 CPU cores - 16GB RAM - Storage to be determined after initial installations of Management DB
Communications	Secure HTTP, Hosts ports availability, Data adapters alignment for each data source/provider
Integration Requirements	Docker
Deployment Requirements	Web server, Docker setup, Container Registry, Internet access, Browser
Security Requirements	SSL, Kafka certificates, Keycloak integration, Elasticsearch
Privacy Requirements	N/A
Critical Factors	SECURED Dashboard environment will be a web-application, so we need to ensure that the volume of data derived is among a reasonable range; Individual tool technological readiness
Containerisation	Yes (Docker images)

5.1.2 SECURED Back End

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-PLAT-USE-M-03, REQ-PLAT-REL-M-04_05, REQ-PLAT-PRIV-M-08, REQ-PLAT-SEC-M-10, REQ-PLAT-SEC-F-17, REQ-PLAT-SEC-M-18_19_20, REQ-PLAT-MAINT-M-27, REQ-PLAT-MAINT-D-29, REQ-PLAT-MAINT-M-26, REQ-DEV-REL-D-58, REQ-DEV-COMP-M-61, REQ-DEV-COMP-M-63, REQ-DEV-COMP-M-64, REQ-DEV-COMP-M-66, REQ-DEV-COMP-D-67, REQ-DEV-COMP-M-68
Non functional Requirements	REQ-PLAT-AVL-D-06, REQ-PLAT-SEC-M-07, REQ-DEV-COMP-D-59, REQ-DEV-USE-D-60, REQ-DEV-COMP-M-67
Development Environment	Python, .yml files for configuring docker and gitlab-ci, bash scripts
Software Requirements	Docker engine, gitlab runner
Hardware Requirements	Too early- To be determined after first demonstration
Communications	Secure HTTP, Hosts ports availability, Data adapters alignment for each data source/provider, kafka topics, Knowledge Base API
Integration Requirements	Docker engine, Gitlab runner
Deployment Requirements	Web server, Docker engine, Container Registry
Security Requirements	SSL, Kafka certificates
Privacy Requirements	N/A
Critical Factors	Technological Readiness Level of individual components
Containerisation	Yes (Docker images)

5.1.3 Knowledge Base

Main functional Requirements	REQ-PLAT-COMP-M-21,REQ-PLAT-MAINT-M-22, REQ-PLAT-MAINT-D-23, REQ-PLAT-MAINT-D-24, REQ-PLAT-MAINT-D-25, REQ-PLAT-MAINT-M-26, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-PLAT-DATA-M-44, REQ-PLAT-DATA-D-48, REQ-DEV-USE-M-51, REQ-DEV-USE-D-54, REQ-DEV-USE-D-55, REQ-PLAT-USE-M-57, REQ-DEV-COMP-M-61, REQ-DEV-COMP-M-68, REQ-PLAT-PRIV-M-08, REQ-PLAT-SEC-M-10, REQ-PLAT-SEC-M-18_19_20, REQ-PLAT-USE-D-02, REQ-DEV-REL-D-58
Non functional Requirements	REQ-PLAT-SEC-D-11, REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-PLAT-DATA-M-45, REQ-PLAT-DATA-M-46, REQ-PLAT-DATA-M-47, REQ-PLAT-DATA-M-49, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60, REQ-DEV-COMP-M-67
Development Environment	Python environment (essential packages like Flask etc), Web Ontology Language (OWL), SQL
Software Requirements	Docker Engine
Hardware Requirements	Server Machine with at least 4 CPU cores and at least 256 GB RAM
Communications	Secure HTTP, Hosts ports availability, Data adapters alignment for each data source/provider, Knowledge Base API
Integration Requirements	Docker Engine
Deployment Requirements	Docker Engine
Security Requirements	Secure Sockets Layer (SSL) technology
Privacy Requirements	All included Datasets and models anonymised
Critical Factors	Not Identified
Containerisation	No

5.1.4 Innohub Services/Tools

5.1.4.1 Anonymizing Tool

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-PLAT-USE-M-57, REQ-DATA-PRIV-M-69
Non functional Requirements	REQ-DATA-PRIV-M-39, REQ-DATA-PRIV-M-40, REQ-DATA-PRIV-M-41, REQ-DATA-PRIV-M-42, REQ-DATA-PRIV-M-43, REQ-DATA-PRIV-D-70, REQ-DATA-PRIV-D-71, REQ-DATA-PRIV-M-72, REQ-DATA-USE-D-82, REQ-DEV-USE-D-50
Development Environment	Java, Kotlin, Javascript
Software Requirements	Kotlin, Java libraries
Hardware Requirements	8 Gb RAM
Communications	I/O for datasets and configuration parameters
Integration Requirements	Docker containers
Deployment Requirements	Docker for deploying docker images
Security Requirements	SSL
Privacy Requirements	To be deployed on the data provider premises
Critical Factors	Not Identified
Containerisation	Yes (Docker images)

5.1.4.2 Anonymisation Assessment Tool

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-DATA-PRIV-D-71, REQ-DATA-SEC-M-73, REQ-DATA-SEC-M-74, REQ-DATA-SEC-M-75, REQ-DATA-SEC-O-76, REQ-DATA-SEC-M-77
Non functional Requirements	REQ-DEV-USE-D-50, REQ-DATA-USE-D-82
Development Environment	Python, Javascript, Flask, HTML.
Software Requirements	Libraries defined in requirements.txt
Hardware Requirements	8GB RAM
Communications	I/O for parameters and configurations.
Integration Requirements	Docker containers
Deployment Requirements	Docker
Security Requirements	SSL
Privacy Requirements	Deployed at user premises and no uploading of identifiable data
Critical Factors	To Be Determined
Containerisation	Docker

5.1.4.3 Synthetic Data Generator Tool and Service

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-PLAT-USE-M-03, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-DATA-DATA-M-79, REQ-DATA-DATA-D-80, REQ-PLAT-PRIV-M-08, REQ-DEV-REL-D-58
Non functional Requirements	REQ-SHW-PERF-M-78, REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60, REQ-DEV-USE-D-50
Development Environment	Mainly developed in Python with pytorch (there might be exceptions).
Software Requirements	Python libraries defined by requirements.txt
Hardware Requirements	GPU with at least 32 GB of RAM
Communications	I/O for parameters and configurations. I/O of data that might be big (from MB to GB).
Integration Requirements	Platform capable of running dockers that deploy an API.
Deployment Requirements	Platform capable of running dockers that deploy an API. A DB to interchange data.
Security Requirements	SSL
Privacy Requirements	N/A
Critical Factors	Insufficient resources for SDG execution as a tool or service
Containerisation	Yes (Docker images)

5.1.4.4 Anonymised Data Transformation Toolset

Main functional Requirements	This tool inherits all the requirements of the Bias Assessment, Unbiasing, Anonymisation, SDG and Anonymisation Assessment tools, REQ-PLAT-USE-D-02
Non functional Requirements	This tool inherits all the requirements of the Bias Assessment, Unbiasing, Anonymisation, SDG and Anonymisation Assessment tools
Development Environment	Python, Java, Javascript
Software Requirements	Python libraries, Javascript libraries
Hardware Requirements	high end Personal Computer with mid-range GPU
Communications	HTTPS
Integration Requirements	Knowledge Base integration
Deployment Requirements	Toolbox repository, on premise deployment
Security Requirements	SSL, HTTPS
Privacy Requirements	N/A
Critical Factors	Not identified
Containerisation	Yes (Docker container)

5.1.4.5 Bias Assessment Service and Tool

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-PLAT-USE-M-03, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-DATA-REL-M-88; REQ-DATA-REL-M-89; REQ-DATA-REL-M-90; REQ-DATA-REL-M-91; REQ-DATA-REL-M-92; REQ-DATA-PRIV-M-93, REQ-PLAT-PRIV-M-08, REQ-PLAT-PRIV-M-08, REQ-DEV-REL-D-58
Non functional Requirements	REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60
Development Environment	Python >=3.10 (containerized in a Docker)
Software Requirements	Python libraries
Hardware Requirements	PC or Server with Few CPU and RAM proportional to the datasets that are processed
Communications	HTTPs, SFTP for I/O for datasets, models and configuration parameters
Integration Requirements	Integration to Container Registry and to Toolbox repository, integration with SDG service/tool
Deployment Requirements	As microservice in cloud server or as a docker container
Security Requirements	SSL, Secure dataset transmission
Privacy Requirements	N/A
Critical Factors	Not identified
Containerisation	Yes (Docker container)

5.1.4.6 Unbiasing Service and Tool

Main functional Requirements	REQ-PLAT-USE-D-02, REQ-PLAT-USE-M-03, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-DATA-PRIV-M-94; REQ-DATA-PRIV-M-95; REQ-DATA-PRIV-M-96; REQ-DATA-PRIV-M-97; REQ-DATA-PRIV-M-98, REQ-DEV-REL-D-58
Non functional Requirements	REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60, REQ-DEV-USE-D-50
Development Environment	Python >=3.10 (containerized in a Docker)
Software Requirements	Python libraries
Hardware Requirements	PC or Server with Few CPU and RAM proportional to the datasets that are processed
Communications	HTTPs, SFTP for I/O for datasets, models and configuration parameters
Integration Requirements	Integration to Container Registry and to Toolbox repository, integration with SDG service/tool
Deployment Requirements	As microservice in cloud server or as a docker container
Security Requirements	SSL, Secure dataset transmission
Privacy Requirements	N/A
Critical Factors	Not identified
Containerisation	Yes (Docker container)

5.1.4.7 Formal Verification

Main functional Requirements	REQ-PLAT-SEC-M-102
Non functional Requirements	REQ-DEV-USE-D-50, REQ-DEV-USE-D-52, REQ-DEV-USE-D-53, REQ-DEV-AVL-M-56, REQ-PLAT-REL-M-103
Development Environment	Python
Software Requirements	Python environment and various packages (specific packages not still defined)
Hardware Requirements	No particular HW requirements
Communications	NA
Integration Requirements	NA
Deployment Requirements	NA
Security Requirements	NA
Privacy Requirements	NA
Critical Factors	NA
Containerisation	NA

5.1.4.8 Synthetic Data Validation Tool (SynthVal)

Main functional Requirements	REQ-DATA-REL-M-104
Non functional Requirements	REQ-DEV-USE-D-50, REQ-DATA-REL-M-105
Development Environment	Python environment and various packages still not defined
Software Requirements	Python environment and various packages still not defined
Hardware Requirements	No particular HW requirements
Communications	NA
Integration Requirements	NA
Deployment Requirements	NA
Security Requirements	NA
Privacy Requirements	NA
Critical Factors	NA
Containerisation	NA

5.1.4.9 Model Marketplace ML/AI/FL

Main functional Requirements	REQ-PLAT-USE-M-03, REQ-PLAT-PRIV-M-08, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35,REQ-DEV-REL-D-58
Non functional Requirements	REQ-PLAT-MAINT-D-28, REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60, REQ-DEV-USE-D-50
Development Environment	Python, .yml files for configuring docker and gitlab-ci, bash scripts, sql database, SFTP
Software Requirements	SQL server, Python interpreter
Hardware Requirements	Storage space
Communications	API endpoints
Integration Requirements	Docker containers
Deployment Requirements	Docker engine
Security Requirements	SSL
Privacy Requirements	N/A
Critical Factors	N/A
Containerisation	Docker containers

5.1.4.10 Legal/GDPR Compliance Check

Main functional Requirements	REQ-PLAT-USE-M-03, REQ-PLAT-PRIV-M-08, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-DEV-REL-D-58, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60
Non functional Requirements	REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-USE-D-50
Development Environment	Node.JS or Next.JS, Javascript, Python
Software Requirements	N/A
Hardware Requirements	N/A
Communications	ReST API communication
Integration Requirements	Remote Server acting as Web application host
Deployment Requirements	Web Server front-end and Back-end
Security Requirements	HTTPS, Knowledge Base API, User Authentication through SECURED Dashboard
Privacy Requirements	N/A
Critical Factors	Not Identified
Containerisation	Possible

5.1.4.11 Anonymisation Decision Support

Main functional Requirements	REQ-PLAT-USE-M-03, REQ-PLAT-SEC-M-10, REQ-PLAT-PERF-M-33, REQ-PLAT-DATA-O-34, REQ-PLAT-MAINT-D-35, REQ-DATA-PRIV-M-81, REQ-PLAT-PRIV-M-08, REQ-DEV-REL-D-58
Non functional Requirements	REQ-DATA-PRIV-D-82, REQ-PLAT-SEC-M-07, REQ-PLAT-DATA-D-36, REQ-PLAT-DATA-M-37, REQ-PLAT-PERF-M-38, REQ-DEV-COMP-D-59, REQ-DEV-COMP-D-60, REQ-DEV-USE-D-50
Development Environment	Node.JS or Next.JS, Javascript, Python
Software Requirements	N/A
Hardware Requirements	N/A
Communications	ReST API communication
Integration Requirements	Remote Server acting as Web application host
Deployment Requirements	Web Server front-end and Back-end
Security Requirements	HTTPS, Knowledge Base API, User Authentication through SECURED Dashboard
Privacy Requirements	N/A
Critical Factors	Not Identified
Containerisation	Possible

5.1.5 Innohub Development Libraries

5.1.5.1 Anonymisation Software/Hardware Development Library

Main functional Requirements	The library complies with all the (Functional and non Functional) Technical Requirements of the Anonymisation Tool, the Anonymisation assessment tool, the SDG tool
Non functional Requirements	REQ-DEV-USE-D-50
Development Environment	C and Python
Software Requirements	N/A
Hardware Requirements	N/A
Communications	TCP/TLS communication channels
Integration Requirements	Provides common interface for all the anonymization services
Deployment Requirements	N/A
Security Requirements	N/A
Privacy Requirements	N/A
Critical Factors	Not Identified
Containerisation	Possible

5.1.5.2 Secure Multiparty Computation Software/Hardware Development Library

Main functional Requirements	REQ-DPROC-SEC-M-87
Non functional Requirements	REQ-PLAT-DATA-M-47, REQ-DPROC-SEC-M-83, REQ-DPROC-SEC-O-84, REQ-DPROC-PERF-D-86
Development Environment	Python for both MPC components
Software Requirements	MP-SPDZ requires glibc 2.1 and Python 3, so Linux 2014 or later or MacOS High Sierra
Hardware Requirements	Typical PC configuration
Communications	TCP/TLS communication channels
Integration Requirements	Needs to interface with FL component for FL-MPC. Secrets Management might also be necessary, for temporary storage of secret protocol values or persistent party identifiers.
Deployment Requirements	Underlying libraries require addresses and port numbers of communicating protocol participants
Security Requirements	Library and related documentation need to be transmitted to users so point-to-point TLS channels would be sufficient. Entity impersonation is a major threat here, e.g. one party participating in a protocol claiming to be someone they are not.
Privacy Requirements	Tooling need to provide encrypted channels and restrict access to plaintext data for participating entities, so no additional privacy measures are foreseen.
Critical Factors	Availability of MP-SPDZ library crucial: no competitor MPC library available. SW requirements (dependencies) might be different from FL layer, meaning that some participants might have to be included from FL process: might cause issues with model accuracy etc.
Containerisation	To be decided at a later development stage

5.1.5.3 Homomorphic Encryption Software/Hardware Development Library

Main functional Requirements	REQ-DPROC-SEC-M-87
Non functional Requirements	REQ-PLAT-DATA-M-47, REQ-DPROC-SEC-M-83, REQ-DPROC-SEC-O-84, REQ-DPROC-PERF-D-85, REQ-DPROC-PERF-D-86
Development Environment	C++ and Python
Software Requirements	OpenFHE requires g++ v9 or later/clang++ v10 or later. CONCRETE available via Docker for most OSs.
Hardware Requirements	For encryption/decryption: Typical PC hardware configuration. For HE evaluation: high end server with at least 20 CPU cores, dedicated high end GPU with at least 32 GB RAM is desirable .
Communications	TCP/TLS Communication channel
Integration Requirements	Secret Management for temporary storage of secret protocol values or persistent party identifiers.
Deployment Requirements	Underlying libraries require addresses and port numbers of communicating protocol participants.
Security Requirements	Entity authentication is necessary: participating parties must be certain of the identity of their counterparties
Privacy Requirements	N/A, provided by tools
Critical Factors	Availability of libraries, speed of execution very dependent on hardware
Containerisation	To be decided at a later development stage

5.1.5.4 Privacy Preserving Federated Learning Development Library

Main functional Requirements	REQ-DATA-PRIV-M-81, REQ-DATA-PRIV-M-99
Non functional Requirements	REQ-DATA-PRIV-M-43, REQ-DEV-USE-D-50, REQ-DATA-PRIV-M-100, REQ-DATA-PRIV-M-101, REQ-DEV-USE-D-50
Development Environment	Pytorch 2.3 and Python 3.10
Software Requirements	Python libraries defined by requirements.txt (Python 3.10, Pytorch 2.3.1, Flower 2.0.1, Numpy 1.26, Scipy 1.13, Scikit-learn 1.5, matplotlib 3.9, xgboost 2.1, seaborn 0.13, shap 0.46, pandas 2.2)
Hardware Requirements	GPU with 32 GB RAM
Communications	TCP/TLS communication channels
Integration Requirements	Needs to interface with SMPC Library and service to use secure aggregation. Secrets Management is also necessary to store pairwise secrets established for secure aggregation.
Deployment Requirements	Pairwise communication of FL parties including the server are required to establish pairwise secrets used in secure aggregation. Network configuration is necessary (network addresses of all parties must be shared and configured). All point-to-point communication must be protected with TLS.
Security Requirements	TLS, SMPC (secure aggregation)
Privacy Requirements	TLS, SMPC (secure aggregation)
Critical Factors	Not Identified
Containerisation	To be decided at a later development stage

5.2 Interconnections and Interfaces

5.2.1 SECURED Front End

Main Inputs	Request Secure Modules
Input Data from Partner	User (Developer)
Nature of Expected Input	User input; Menu selection, free text, decision trees
Interfaces	KB-API, Management DB, Innohub API, Dashboard
Triggered by	User request
Main Outputs	List of secure modules
Output Data to Partner	List of secure relative modules/source code
Nature of Expected Output	Module source code, JSON, CSV, images
Related Scenarios	All related scenarios
Interfaces	KB-API, Management DB, Innohub API, Dashboard
Triggered by	Once internal checks are concluded, output is provided to all stakeholders

5.2.2 SECURED Back End

Main Inputs	Synthetic/open data for the Data Ingestion Mechanism. Kafka messages for the communication module.
Input Data from Partner	Developers and End-Users
Nature of Expected Input	Serialized (JSON, XML, CSV) data in the form of Apache Kafka messages.
Interfaces	All developed tools and Knowledge Base APIs
Triggered by	User request, Automatic/rule-based pipelines
Main Outputs	Kafka messages/ Logs for the communication module. Synthetic Data pointers for the Knowledge Base
Output Data to Partner	No actual data to be shared through Innohub
Nature of Expected Output	Serialized Messages (JSON, CSV)
Related Scenarios	All related scenarios
Interfaces	All developed tools and Knowledge Base APIs
Triggered by	User request, Automatic/rule-based pipelines

5.2.3 Knowledge Base

Main Inputs	Synthetic datasets, metadata for synthetic datasets, metadata for public datasets, ready-to-use software tools, AI trained models, metadata for Secured services, metadata for trained models
Input Data from Partner	All partners
Nature of Expected Input	JSON files, image files, docker files, CSV files, DICOM files, h5 files, pth files
Interfaces	ReST API
Triggered by	SECURED Tools and Services, SECURED Back End
Main Outputs	outputs data, ML/DL Models, tools, containers and data in general from the SECURED datalake, SECURED data inventory, Synthetic data cache, container registry, toolbox repository, knowledge graph subcomponents
Output Data to Partner	SECURED Tools and Services
Nature of Expected Output	Relational databases, file system storage
Related Scenarios	All related scenarios
Interfaces	endpoints: /syntheticDataset/, /dataInventory/, /AIModels/, /containerRegistry/, /toolboxRepository/
Triggered by	SECURED Back End

5.2.4 Innohub Services/Tools

5.2.4.1 Anonymizing Tool

Main Inputs	Raw data to anonymise in csv, xlsx or txt format files
Input Data from Partner	Unbiased data from Unbiasing tool is provided for anonymising
Nature of Expected Input	Data such as Electronic Health Records (EHRs) and time-series data.
Interfaces	An API is provided for accessing the services. A Graphical User Interface can be provided for user interaction
Triggered by	User through a user interface
Main Outputs	Anonymised dataset and anonymisation process report in csv format and pdf format, respectively
Output Data to Partner	Anonymised data to Anonymisation Assesment tool
Nature of Expected Output	Anonymised health data (csv) and report (pdf) of the anonymisation process to be evaluated by the user.
Related Scenarios	Expected participation in Use Case 2, 3 and 4
Interfaces	API
Triggered by	Orchestrator application (SECURED Back End)

5.2.4.2 Anonymisation Assessment Tool

Main Inputs	Anonymised data types and details, with data samples in csv, xlsx, image or txt format files
Input Data from Partner	For testing only: anonymised data from anonymisation tool is provided for anonymisation assessment
Nature of Expected Input	Parameters of anonymised health data (such as electronic health records (EHR) and time-series data), with the possibility of providing small data samples. The actual data itself should not be shared to ensure privacy.
Interfaces	Web-based user interface. A limited number of tools (those developed as part of the project, and potentially attacks for which the implementation is openly available) will be integrated in the WP library.
Triggered by	User through a user interface
Main Outputs	The main output is a user-readable threat (attack) list, with user guidance. In addition, specific tools (those integrated in the library) will output de-anonymisation results when used against a user-provided dataset locally (no data to be uploaded on the web tool).
Output Data to Partner	No direct output. However, the platform may provide useful insight into the improvement of the anonymisation service
Nature of Expected Output	Output displayed in web interface. Potentially, a PDF report may be extracted from the page (feasibility to be assessed).
Related Scenarios	We will focus on use cases 2, 3, and 4
Interfaces	Web-based user interface.
Triggered by	User uploads a dataset via the web interface

5.2.4.3 Synthetic Data Generator Service and Tool

Main Inputs	Configuration parameters for the generator. If the model works with input data, we require data in .dcm (images), pathological image native format and .csv (tabular, time series).
Input Data from Partner	We expect interaction with Bias assessment
Nature of Expected Input	DICOM, Histopathological image native format and CSV (and related)
Interfaces	API and library
Triggered by	A coordination application for the advanced services or user interface
Main Outputs	Generated images in png, DICOM or pathological image format. Tabular/time series data is expected to be in CSV.
Output Data to Partner	Data/Requests to be obtained from Bias tools
Nature of Expected Output	DICOM, Histopathological image native format and CSV (and related)
Related Scenarios	Use Case 2 and 3
Interfaces	API, library
Triggered by	A coordination application for the advanced services or user interface

5.2.4.4 Anonymised Data Transformation Toolset

Main Inputs	Raw data to anonymise in csv, xlsx or txt format files
Input Data from Partner	Use Case Partners and Open Call participants
Nature of Expected Input	Anonymised and Unbiased datasets
Interfaces	ReST API
Triggered by	Users
Main Outputs	Anonymised and Unbiased csv, xlsx or txt format datasets
Output Data to Partner	User
Nature of Expected Output	dataset in various popular data formats
Related Scenarios	Related to Use Cases 2,3 and 4
Interfaces	ReST API
Triggered by	User

5.2.4.5 Bias Assessment Service and Tool

Main Inputs	Dataset and sensitive attribute (one or several) of interest.
Input Data from Partner	Sensitive attribute of interest from the use case provider. The dataset from UC provider in case of raw data, BSC (T2.3) for the synthetic data, T2.1 (ATOS) for anonymised dataset
Nature of Expected Input	dataset dependent.
Interfaces	ReST API
Triggered by	A request for bias assessment by SECURED Back End (for service) or by the user (for tool)
Main Outputs	List of metrics
Output Data to Partner	Entity that made the request (data provider)
Nature of Expected Output	Raw text, CSV of JSON (to be further refined in later development stage)
Related Scenarios	Use Case 3 and 4
Interfaces	ReST API
Triggered by	SECURED Back End (for service) or by the user (for tool)

5.2.4.6 Unbiasing Service and Tool

Main Inputs	dataset, the model (architecture, weights, pre-processing etc), sensitive attribute description
Input Data from Partner	the training dataset, the whole training procedure (pre-processing, augmentation, training, model architecture, weights, etc)
Nature of Expected Input	Files in a folder architecture. To be defined for the training procedure (code)
Interfaces	ReST API
Triggered by	model developer user (for tool) or SECURED Back End (for service)
Main Outputs	Depending on the scenario, either a training procedure including bias mitigation or the weights of the model trained with a bias reduction approach.
Output Data to Partner	
Nature of Expected Output	To be defined in later development stage
Related Scenarios	Use Case 3 and 4
Interfaces	ReST API
Triggered by	SECURED Back End (for service) or the model developer user (for tool)

5.2.4.7 Formal Verification

Main Inputs	Updates about components of the framework; Updates about new threats to consider.
Input Data from Partner	Data/information related to components and dataflows
Nature of Expected Input	CSV files
Interfaces	NA
Triggered by	NA
Main Outputs	Feedback about the consistency of the framework with respect to cybersecurity aspects.
Output Data to Partner	NA
Nature of Expected Output	Textual output
Related Scenarios	NA
Interfaces	NA
Triggered by	NA

5.2.4.8 Synthetic Data Validation Tool (SynthVal)

Main Inputs	Real data and synthetic data produced within the Synthetic Data Generator architectural block
Input Data from Partner	NA
Nature of Expected Input	Data such as images and time series
Interfaces	NA
Triggered by	NA
Main Outputs	Information regarding the similarity between real and synthetic data
Output Data to Partner	NA
Nature of Expected Output	Textual Output
Related Scenarios	NA
Interfaces	NA
Triggered by	NA

5.2.4.9 Model Marketplace ML/AI/FL

Main Inputs	Trained AI models
Input Data from Partner	Model weights files
Nature of Expected Input	Raw files, JSON
Interfaces	ReST API
Triggered by	User input
Main Outputs	Requested model, JSON formatted information
Output Data to Partner	ReST API
Nature of Expected Output	.pth or .h5 files, JSON file
Related Scenarios	All relevant scenarios
Interfaces	ReST API
Triggered by	User request, Automatic/rule-based pipelines (SECURED Back End)

5.2.4.10 Legal/GDPR Compliance Check

Main Inputs	User input on datasets and AI models specification as well as possible usage (country of origin, country of usage etc) provided through a Dashboard related front-end
Input Data from Partner	User
Nature of Expected Input	JSON format
Interfaces	ReST API (to be finalized in a later development stage)
Triggered by	SECURED Dashboard and user
Main Outputs	Series of Legal Guidelines on how to comply with EU and national regulations on privacy/anonymity
Output Data to Partner	SECURED Dashboard and User
Nature of Expected Output	JSON format
Related Scenarios	All relevant scenarios
Interfaces	ReST API
Triggered by	SECURED Dashboard and User

5.2.4.11 Anonymisation Decision Support

Main Inputs	User configuration input provided through a Dashboard related front-end
Input Data from Partner	User
Nature of Expected Input	JSON format
Interfaces	ReST API (to be finalized in a later development stage)
Triggered by	SECURED Dashboard and user
Main Outputs	Series of Technical Guidelines on how to configure the Anonymisation toolset, Unbiasing and Synthetic Data Generator tools or services
Output Data to Partner	SECURED Dashboard and User
Nature of Expected Output	JSON format
Related Scenarios	All relevant scenarios
Interfaces	ReST API
Triggered by	SECURED Dashboard and User

5.2.5 Innohub Development Libraries

5.2.5.1 Anonymisation Software/Hardware Development Library

Main Inputs	Library is offering common API to the Anonymisation functions and used dataset to be anonymised as inputs
Input Data from Partner	User of the library
Nature of Expected Input	Specific API calls, data to be anonymised in csv, txt, png, dicom or raw format
Interfaces	Dedicated API
Triggered by	User
Main Outputs	anonymised data and relevant metadata
Output Data to Partner	User
Nature of Expected Output	Same as the input
Related Scenarios	All relevant scenario
Interfaces	Dedicated API
Triggered by	User

5.2.5.2 Secure Multiparty Computation Software/Hardware Development Library

Main Inputs	For FL component, this component acts as a layer on top of FL component to provide secure aggregation functionality. For encrypted training component, the input is the training data.
Input Data from Partner	FL-MPC component requires stable interface with FL component; encrypted training component requires knowledge of the possible ranges of (encoding of) all data fields in training data plus model output field.
Nature of Expected Input	Varies, depending on usage: the users will use our documentation to configure the library and provide input as instructed.
Interfaces	API
Triggered by	FL-MPC triggered as an option in FL component, (but unclear how this will work: not all FL approaches can support SecAgg and user needs to choose which aggregation type to be used, depending on number of parties and threat model). Encrypted training is a standalone component.
Main Outputs	FL-MPC gives an aggregated model back to FL component; encrypted training gives a (tree-based) ML model. Format TBC for both.
Output Data to Partner	Varies, depending on usage: FL-MPC will feed back into FL component so would be ML model weights at each round, encrypted training will provide a ML model.
Nature of Expected Output	Varies depending on usage, see above and below. Data formats for input and output must be agreed by protocol participants in encrypted training processes.
Related Scenarios	The output of this component is a ML model or model weights so it will either go into model marketplace or would be stored on an entity's system for inference access. Second option also has potential for adding encryption layer, linking with HE library tooling (TC3.4a)
Interfaces	FL-MPC connected to FL component
Triggered by	User

5.2.5.3 Homomorphic Encryption Software/Hardware Development Library

Main Inputs	Data to be encrypted/decrypted, function to be evaluated
Input Data from Partner	For ML inference, data input is the inference query (by the querying party) and the model (by the model owner). For scenarios like UC1, input would be image data for regridding process. For UC2 and UC4, the input to encrypted inference usage is the ML model
Nature of Expected Input	Entirely depends on the usage, and would be different for two parties in the same workflow, e.g. in encrypted ML inference one party provides the inference input and another provides the ML model
Interfaces	API
Triggered by	Request from the model owner/model trainer
Main Outputs	For ML inference, output would be the decrypted query answer
Output Data to Partner	
Nature of Expected Output	Entirely depends on the usage, and as above outputs may be different for two participants In the same protocol.
Related Scenarios	Anything that requires computation to be made on an encrypted value.
Interfaces	API
Triggered by	USer

5.2.5.4 Privacy Preserving Federated Learning Development Library

Main Inputs	(1) Architecture and type of the machine learning model to be trained. (2) Local training data that is not shared but used to compute gradients (model update) to be shared. (3) Hyper-parameters of the training. In particular, (1) Model architecture saved by torch.save as a serialized python object. (2) Pre-processed training and validation data saved as numpy arrays by numpy.save(z). The features and ground-truth labels are saved in separate numpy arrays. The validation and training data must have the same format. (3) The hyper-parameters of the training (batch-size, optimizer, loss function, learning rate, etc.)
Input Data from Partner	Expected input from TC3.1 (FL-MPC); secure aggregation and potentially encrypted inference, potentially from TC3.3 (unbiased tools) in case such tools need to be integrated into FL framework (alternatively the user can manually build a loss function the incorporates bias mitigation), TC2.3 (synthetic data generation) if the model is provided by an innohub service or from knowledgebase, TC2.1 (Anonymised training data)
Nature of Expected Input	Numpy arrays, serialized python objects
Interfaces	API
Triggered by	Request from the model owner/ model trainer
Main Outputs	Trained machine learning model (model parameters), evaluation the trained model and the training process
Output Data to Partner	UC2, UC4
Nature of Expected Output	Trained machine learning model (model parameters) serialized python objects by torch.save, statistics about training procedures (loss curves, evaluation metric values)
Related Scenarios	UC2 (telemetry monitoring) and UC4 (genomics)
Interfaces	API
Triggered by	A coordination application for the advanced services or user interface

6 Conclusions

This document concluded the work on Task 4.1 by producing this Deliverable on the Architecture Specifications, Analysis and Design. This Deliverable showcases the overall SECURED architecture with all its domains, individual services, components, tools, libraries and knowledge bases, as well as their interactions initially in a high-level overview by updating and elaborating on the architecture defined in D4.1. The overall architecture was followed with an in-depth description and analysis of every the domains, individual services, components, tools, libraries and knowledge bases, its function, interaction with every connected component of the architecture and the respective output.

Having the SECURED architecture defined in such a fine-grained detail, allowed us to provide a user-centric view of the SECURED solution by adopting the User Journey paradigm. We set clear objectives, identified the key users of the solution and their actual goals, and showcased how each one of them would use the SECURED platform and what interactions these users have between them and the SECURED solution. Using this approach we identified four distinct user types each one with a different end goal of using the SECURED architecture and showcased how each one will use the various architectural components in order to achieve their goals.

Finally, we presented the technical requirements, containing all the necessary information, such as functional and nonfunctional requirements, software, hardware, integration, deployment security and privacy requirements. The technical requirements were followed by the interface and interconnections of all domains, services and tools defining information, such as inputs and outputs, integration, deployment security and privacy requirements.

In conclusion this Deliverable showcased the final SECURED Reference Architecture providing the necessary blueprint, along with technical requirements and details for the current and upcoming integration activities and paving the road towards the expected SECURED Innohub solution for the rest of the WPs to follow towards the final results.

A Appendix: Overview of Identified Technical Requirements

In D4.1 [1] the consortium partners have provided an extensive list of technical requirements that optimally fit the preliminary SECURED architecture and the components. Given that in M18 the activities of T4.1 have been completed fully and have served their full purpose and that the activities of the tasks WP2 and WP3 have progressed significantly, some of the technical requirements described in D4.1 can be amended to better fit the final SECURED Reference Architecture. In this subsection, we provide the D4.1 technical requirements that are amended or merged into more comprehensive and meaningful requirements. We also provide Technical Requirements that are out-of-scope for the SECURED Reference Architecture since they refer to other aspects of SECURED, e.g., the Open Call. For each such merge/amendment, we provide a relevant justification.

A.1 Altered or Merged Technical Requirements

A.1.1 Merging A

A.1.1.1 Original D4.1 Technical Requirements

REQ-PLAT-REL-M-04	**Short Name:**	Application/Service Private deployment support			
Description	Downloaded Application/Service binaries/artifacts, should be able to be easily instantiated in Private Cloud Environments (K8s clusters) using standardized (K8s) software/tools				
Priority	Mandatory	**Type**	Functional	**CUR**	13 - 15, 30, 31

REQ-PLAT-REL-M-05	**Short Name:**	Application/Service Public deployment support			
Description	Downloaded Application/Service binaries/artifacts, should be able to be easily instantiated in Public Cloud Environments (K8s clusters) using standardized (K8s) software/tools				
Priority	Mandatory	**Type**	Functional	**CUR**	13 - 15, 30, 31

A.1.1.2 Merged Technical requirement

REQ-PLAT-REL-M-04_05	**Short Name:**	Application/Service Deployment Support			
Description	Downloaded Application/Service binaries/artifacts, should be able to be easily instantiated in Public or Private Cloud Environments				
Priority	Mandatory	**Type**	Functional	**CUR**	13 - 15, 30, 31
Merging Justification	Both REQ-PLAT-REL-M-04 and REQ-PLAT-REL-M-05 refer to the same concept of cloud service/Application deployment so the could be addressed by one requirement. The original requirements restrict the developer on specific type of tools (e.g K8s) that may be mandatory for extremely complex systems but not for systems of like SECURED. To allow the flexibility to use a broader range of tools for instantiating Application/Service binaries/artifacts is is preferable that the type of tools used for matching the requirements be decided by the SECURED project developers/integrators.				

A.1.2 Merging B

A.1.2.1 Original D4.1 Technical Requirements

REQ-PLAT-SEC-M-18	**Short Name:**	Secrets Management			
Description	The Vault must be able to Securely store and tightly control access to tokens, passwords, certificates, API keys, and other secrets. In addition, the Vault must also support K8s-related secret management (i.e., KV Secrets Engine, Database Credentials, Kubernetes Secrets)				
Priority	Mandatory	**Type**	Functional	**CUR**	10, 12, 13 - 18, 30, 32

REQ-PLAT-SEC-M-19	**Short Name:**	Key Management			
Description	The Vault must support the creation and control of the encryption keys used for all types of data encryption (data at rest/in transit).				
Priority	Mandatory	**Type**	Functional	**CUR**	10, 30, 32

REQ-PLAT-SEC-M-20	**Short Name:**	Certificate Management			
Description	The Vault must support the provisioning, management, and deployment of public and private Transport Layer Security/Secure Sockets Layer (TLS/SSL) certificates which could be used to secure external/internal connected resources.				
Priority	Mandatory	**Type**	Functional	**CUR**	10, 30, 32

A.1.2.2 Merged Technical requirement

REQ-PLAT-SEC-M-18_19_20	**Short Name:**	Security Management			
Description	SECURED mus be able to provide structures that will handle the overall secret management of the system. This may include the generation and management of cryptography keys. the generation and management of digital certificates for use in Transport Layer Security/Secure Sockets Layer (TLS/SSL) sessions as well as any access control tokens that are needed in the overall management of the SECURED Innohub				
Priority	Mandatory	**Type**	Functional	**CUR**	10, 12, 13 - 18, 30, 32
Merging Justification	REQ-PLAT-SEC-M-18, REQ-PLAT-SEC-M-19 and REQ-PLAT-SEC-M-20 refer different angles of the same overall concept for managing security related secrets within the SECURED platform. The requirements seem to point to specific commercial Vault type of tools that are necessary in order to handle secrets for highly complex systems. Given the decentralized nature of loosely interconnected services and downloadable tools in the SECURED architecture as well as given the small number of tools and services offered by the Innohub, such complex secret management systems may not be necessary. In order to allow the developers/integrators to explore a broad range of options on security management including open source solutions, the 3 requirements are merged into one that describe the same requirement goals but without profiling specific tools to achieve them.				

A.1.3 Merging C

A.1.3.1 Original D4.1 Technical Requirements

REQ-DEV-COMP-M-61	**Short Name:**	Extension - SECURED Infrastructure interaction			
Description	New services shall be able to interact with the SECURED Infrastructure though the opensource SECURED REST API that will be specified in the project.				
Priority	Mandatory	**Type**	Functional	**CUR**	30 - 40

REQ-DEV-COMP-M-63	**Short Name:**	User authorization to experimentation/development data			
Description	SECURED Infrastructure should ensure that third party developers/experimenters are authorized to perform tests isolated, fully independent, without the ability to access other experimenter's data(sets).				
Priority	Mandatory	**Type**	Functional	**CUR**	30 - 40

A.1.3.2 Merged Technical requirement

REQ-DEV-COMP-M-61_63	**Short Name:**	Extension - SECURED Infrastructure Interactions and data experimentation			
Description	New services shall be able to interact with the SECURED Infrastructure though the various open source SECURED REST APIs. SECURED Infrastructure will be able to offer isolated fully independent instances to third party developers/experimenters for SECURED service experimentation				
Priority	Mandatory	**Type**	Functional	**CUR**	30 - 40
Merging Justification	The REQ-DEV-COMP-M-61 and REQ-DEV-COMP-M-63 requirements refer to open call external participants interacting with the SECURED platform. As expected such users and their developed services through the open-call must be able to access SECURED in a secure manner (isolated from other services) using the platform's various REST APIs. The same authentication mechanism will be applied to users that are interacting with the platform and the principles for service isolation is applied also to the users. Thus, we consider that the two requirements are in the actual system handled by the same conceptual structures and can be merged into one. Note that SECURED does not store anonymized datasets but rather registers their presence of the data owner's premises. The only datasets that are handled through SECURED are anonymized synthetically generated data and therefore the experimentation is restricted to those.				

A.1.4 Updated/Revised Requirement

REQ-DEV-COMP-D-67	Short Name:	Extension – Behavior Monitoring			
Description	New services should be authorized to access exposed APIs based on continuous monitoring (behavior, traffic patterns, queries, etc) provided by the SECURED Infrastructure. Monitored access could be utilized to ensure appropriate behavior or to detect potentially malicious actions (i.e. DDoS-type attacks taking advantage of exposed APIs)				
Priority	Desirable	**Type**	Non-Functional	**CUR**	30, 31, 36
Update Justification	The requirement refers to open call external participants interacting with the SECURED platform and performing malicious activities. While such a requirement is desirable it is not mandatory for the TRL level of the SECURED project/ To introduce malicious behavior monitoring mechanisms on top of the SECURED platform may need the introduction of new tools that are not foreseen in the DoA. Therefore, the Priority of this requirement has been changed from Mandatory to Desirable.				

A.1.5 Out of Scope Requirements for the SECURED Architecture

The SECURED Reference Architecture include a series of components for which individually in Section 5 we provide an association with the D4.1 identified Technical Requirements (with the updates provided in the previous subsections of this appendix). However, in D4.1 we provide also the needed requirements of the external tools and services developed during the SECURED open call in order to comply with the SECURED Innohub. Those requirements while important for the open call process cannot directly be linked to the SECURED Reference Architecture components (since they focus on the external open-call components). Nevertheless, regardless of being out-of-scope for inclusion in Section 5, we report them for the sake of completeness in this subsection since they must be considered by the open-call participants. Note that all these requirements are documenting needs for the open call services and tools that are supported (based on other Technical requirements) by the SECURED platform.

REQ-DEV-COMP-M-62	Short Name:	Extension - Remote Operational Control			
Description	New services should provide means for remote operational control from the corresponding SECURED Infrastructure entity (Controller / CLI / Admin UI)				
Priority	Mandatory	**Type**	Functional	**CUR**	30 - 40

REQ-DEV-COMP-M-64	Short Name:	Extension – SECURED Infrastructure secure communication			
Description	The connectivity link/communication channel between all entities/services shall be secure, potentially with end-to-end encryption.				
Priority	Mandatory	**Type**	Functional	**CUR**	10 - 12, 30 - 40

REQ-DEV-COMP-M-65	Short Name:	Extension - User Authentication			
Description	New services should support various levels of authorization (i.e. remote user / centralized user / administrator) and identification				
Priority	Mandatory	**Type**	Functional	**CUR**	16, 30 - 40

REQ-DEV-COMP-M-66	Short Name:	QoS Alerting Mechanism

Description	New services should generate alerts if expected/predefined QoS cannot be reached, in order to trigger adaptation/improvement mechanisms on the SECURED Infrastructure side. The QoS should be the output of monitoring values/performance metrics such as latency, throughput, uptime as well as application-specific KPIs				
Priority	Mandatory	**Type**	Non-Functional	**CUR**	25, 33, 39, 40

A.2 Final D4.1/D4.2 Technical Requirements

Requirement	Short description	Priority	Type
REQ-PLAT-PORT-M-01	Integration of experimenter's complimentary components	Mandatory	Functional
REQ-PLAT-USE-D-02	Application/Service deployment at the Edge	Desirable	Functional
REQ-PLAT-USE-M-03	Application/Service deployment infrastructure support	Mandatory	Functional
REQ-PLAT-REL-M-04_05	Application/Service deployment support	Mandatory	Functional
REQ-PLAT-AVL-D-06	Cloud native elasticity (scale out/scale in) support	Desirable	Non-Functional
REQ-PLAT-SEC-M-07	Security, inAccess support	Mandatory	Functional
REQ-PLAT-SEC-M-10	Strong User Authentication support	Mandatory	Functional
REQ-PLAT-SEC-D-11	Single-Sign On / Out	Desirable	Functional
REQ-PLAT-SEC-M-12	User Federation Support	Mandatory	Functional
REQ-PLAT-SEC-M-13	Standard Protocol and Identity Brokering Support	Mandatory	Functional
REQ-PLAT-SEC-D-14	Administration Console	Desirable	Functional
REQ-PLAT-SEC-M-15	RBAC Authorization Service and Customized Policy Support	Mandatory	Non-Functional
REQ-PLAT-SEC-P-16	Account Management Console / Environment (OPTIONAL)	Possible	Non-Functional
REQ-PLAT-SEC-P-17	2(M)Factor Authentication Support (OPTIONAL)	Possible	Functional
REQ-PLAT-SEC-M-18_19_20	Security Management	Mandatory	Functional
REQ-PLAT-COMP-M-21	Source-agnostic data ingestion	Mandatory	Functional
REQ-PLAT-MAINT-M-22	Metrics Handling support	Mandatory	Functional
REQ-PLAT-MAINT-D-23	Log Ingestion support	Desirable	Functional
REQ-PLAT-MAINT-D-24	Custom Query Support	Desirable	Functional
REQ-PLAT-MAINT-D-25	Traces Handling support	Desirable	Functional
REQ-PLAT-MAINT-M-26	Centralized Logging Repository query environment / interface	Mandatory	Functional

REQ-PLAT-MAINT-M-27	Metric Alerts support	Mandatory	Functional
REQ-PLAT-MAINT-D-28	Activity Log Alert support	Desirable	Functional
REQ-PLAT-MAINT-D-29	Customized Alert Rules support	Desirable	Functional
REQ-PLAT-MAINT-M-30	Dashboards and Tables	Mandatory	Functional
REQ-PLAT-MAINT-M-31	Log Reports	Mandatory	Functional
REQ-PLAT-MAINT-M-32	Analytics Graphs	Mandatory	Functional
REQ-PLAT-PERF-M-33	Accelerated Data Retrieval Mechanism	Mandatory	Functional
REQ-PLAT-DATA-O-34	Common Data Integration Pattern Functionality (ETL/ELT)	Optional	Functional
REQ-PLAT-MAINT-D-35	Data Pipeline Definition and Execution	Desirable	Functional
REQ-PLAT-DATA-D-36	Dataset Identification and Handling	Desirable	Non-Functional
REQ-PLAT-DATA-M-37	Data Flow Mapping	Mandatory	Non-Functional
REQ-PLAT-PERF-M-38	Integration Runtime	Mandatory	Non-Functional
REQ-DATA-PRIV-M-39	Data Masking Support	Mandatory	Non-Functional
REQ-DATA-PRIV-M-40	Pseudoanonymization Support	Mandatory	Non-Functional
REQ-DATA-PRIV-M-41	Generalization Support	Mandatory	Non-Functional
REQ-DATA-PRIV-M-42	Data Swapping Support	Mandatory	Non-Functional
REQ-DATA-PRIV-M-43	Data Pertubation Support	Mandatory	Non-Functional
REQ-PLAT-DATA-M-44	Compatibility with the Hadoop Distributed File System (HDFS)	Mandatory	Functional
REQ-PLAT-DATA-M-45	Compatibility with standardized Analytics Engines via a dedicated Query Layer	Mandatory	Non-Functional
REQ-PLAT-DATA-M-46	Data Lake solution must be Data Source Agnostic	Mandatory	Non-Functional
REQ-PLAT-DATA-M-47	Native Data Type Support	Mandatory	Non-Functional
REQ-PLAT-DATA-D-48	Data Lake REST API	Desirable	Functional
REQ-PLAT-DATA-M-49	Layered and Isolated Architecture	Mandatory	Non-Functional
REQ-DEV-USE-D-50	Software Documentation	Desirable	Other
REQ-DEV-USE-M-51	Main Codebase Repository Requirements	Mandatory	Other

REQ-DEV-USE-D-52	Verification tools virtualization	Desirable	Non-Functional
REQ-DEV-USE-D-53	Exclusive verification tests	Desirable	Non-Functional
REQ-DEV-USE-D-54	Open-sourced validation tools	Desirable	Non-Functional
REQ-DEV-USE-D-55	Validation framework containerization	Desirable	Non-Functional
REQ-DEV-AVL-M-56	Cloud-native compatibility	Mandatory	Non-Functional
REQ-PLAT-USE-M-57	SECURED Toolbox / Software Repository	Mandatory	Functional
REQ-DEV-REL-D-58	SECURED ReST API for facilitating component interconnection	Desirable	Functional
REQ-DEV-COMP-D-59	SECURED API Security Practices	Desirable	Non-Functional
REQ-DEV-USE-D-60	SECURED API Documentation	Desirable	Other
REQ-DEV-COMP-M-61_63	Extension - SECURED Infrastructure Interactions and data experimentation	Mandatory	Functional
REQ-DEV-COMP-D-67	Extension – Behavior Monitoring	Desirable	Non-Functional
REQ-DEV-COMP-M-68	Testbed-Experimenter collaboration	Mandatory	Functional
REQ-DATA-PRIV-M-69	Anonymization service and tool for different, heterogeneous Health data types	Mandatory	Functional
REQ-DATA-PRIV-D-70	Anonymization of high data volumes	Desired	Non Functional
REQ-DATA-PRIV-D-71	Offered Anonymization to withstand de-anonymization attacks	Desired	Functional
REQ-DATA-MAINT-M-72	Provide report/guarantee of anonymization process	Mandatory	Functional
REQ-DATA-SEC-M-73	Assess anonymized dataset for Timeseries Health Data	Mandatory	Functional
REQ-DATA-SEC-M-74	Assess anonymized dataset for Image Health Data	Mandatory	Functional
REQ-DATA-SEC-M-75	Assess anonymized dataset for Electronic Health Record Data	Mandatory	Functional
REQ-DATA-SEC-O-76	Provide broad-scope de-anonymization techniques	Optional	Functional
REQ-DATA-SEC-M-77	Provide report of Anonymization assessment	Mandatory	Functional
REQ-SHW-PERF-M-78	Access to HPC hardware for efficient synthesis (several CPUs, GPUs,...)	Mandatory	Non-Functional

REQ-DATA-DATA-M-79	Generate data for different data types and modalities	Mandatory	Functional
REQ-DATA-DATA-D-80	Data novelty evaluation	Desirable	Functional
REQ-DATA-PRIV-M-81	Privacy risk and data utility trade-off mechanisms for different health data types.	Mandatory	Functional
REQ-DATA-PRIV-D-82	User friendly anonymisation decision support and anonymization tools	Desirable	Non-Functional
REQ-DPROC-SEC-M-83	Seamless integration of SotA open-source SMPC/HE libraries.)	Mandatory	Non-Functional
REQ-DPROC-SEC-O-84	Cost Estimator for MPC/HE protocols.	Optional	Non-Functional
REQ-DPROC-PERF-D-85	Circuit optimizer for hardware acceleration of HE.	Desirable	Non-Functional
REQ-DPROC-PERF-D-86	SMPC or HE solutions very fast response time	Desirable	Non-Functional
REQ-DPROC-SEC-M-87	Customized, adaptable SMPC Transformation process	Mandatory	Functional
REQ-DATA-REL-M-88	Provide accurate bias score for a given dataset.	Mandatory	Functional
REQ-DATA-REL-M-89	Detection of Bias in Timeseries Health Data.	Mandatory	Functional
REQ-DATA-REL-M-90	Detection of Bias in Image Health Data	Mandatory	Functional
REQ-DATA-REL-M-91	Detection of Bias in Electronic Health Record Data.	Mandatory	Functional
REQ-DATA-REL-M-92	Detection of Bias in Anonymized Datasets.	Mandatory	Functional
REQ-DATA-PRIV-M-93	Provide analytic bias assessment reports.	Mandatory	Functional
REQ-DATA-PRIV-M-94	Unbiasing of Timeseries Health Data.	Mandatory	Functional
REQ-DATA-PRIV-M-95	Unbiasing of Image Health Data.	Mandatory	Functional
REQ-DATA-PRIV-M-96	Unbiasing of Electronic Health Record Data	Mandatory	Functional
REQ-DATA-PRIV-M-97	Unbiasing of Anonymized Datasets	Mandatory	Functional
REQ-DATA-PRIV-M-98	Provide report/guarantee of the Unbiasing process	Mandatory	Functional

B Appendix: New Technical Requirements

REQ-DATA-PRIV-M-99	**Short Name:**	Privacy-preserving Distributed Learning Framework for Health Data	
Description	Federated Learning of Machine Learning Models for Times-series and EHR & Genomic Health Data		
Priority	Mandatory	**Type**	Functional

REQ-DATA-PRIV-M-100	**Short Name:**	Privacy Assessment of Federated Learning	
Description	Measuring the unintended information leakage about the training data through Federated Learning including model updates and the trained model.		
Priority	Mandatory	**Type**	Functional

REQ-DATA-PRIV-M-101	**Short Name:**	Contribution Scoring in Privacy-Preserving Federated Learning	
Description	Measuring the contribution of the model updates of the participating clients in Federated Learning (e.g., to boost accuracy).		
Priority	Mandatory	**Type**	Non-Functional

REQ-PLAT-SEC-M-102	**Short Name:**	Provide automated reasoning capabilities for verification and threat identification.	
Description	In the context of SECURED the platform tools and services should be verifiable in a measurable manner and relevant reasoning on the achieved metrics must be provided with out user involvement in the overall process		
Priority	Mandatory	**Type**	Functional

REQ-PLAT-REL-M-103	**Short Name:**	Maintain high reliability and accuracy in consistency checking and threat detection.	
Description	The verification metrics must be reasonably reliable and accurate therefore reflecting the real status of the verified system/component		
Priority	Mandatory	**Type**	Non-Functional

REQ-DATA-REL-M-104	**Short Name:**	Provide validation of synthetic data generated	
Description	Synthetically Generated Data must verified that they reflect real health related conditions and that closely match real data		
Priority	Mandatory	**Type**	Functional

REQ-DATA-REL-M-105	**Short Name:**	Maintain reliability and accuracy in validating the data	
Description	The similarity between real and synthetic data must be reasonably reliable and accurate therefore reflecting the real status of the synthetically generated datasets		
Priority	Mandatory	**Type**	Non-Functional

References

[1] SECURED project: D4.1-State of the Art and initial technical requirements. Fournaris, Apostolos Editor. 2023.

[2] M. B. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," RFC 7519, May 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7519

[3] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Web Semant.*, vol. 5, no. 2, p. 51–53, jun 2007. [Online]. Available: https://doi.org/10.1016/j.websem.2007.03.004

[4] R. D. Shearer, B. Motik, and I. Horrocks, "Hermit: A highly-efficient owl reasoner." in *Owled*, vol. 432, 2008, p. 91.

[5] SECURED project: D1.2-GDPR and Ethics Project Guidelines, Spajić, Daniela Editor. 2023.

[6] SECURED project: D2.5-Legal and ethical framework and analysis, Spajić, Daniela Editor. 2024.

[7] SECURED project: D3.1-Interim report on Scalable Secure Multiparty Computation, Federated Learning and Unbiased AI techniques and tools. Aszalos, Albert Editor. 2024.

[8] SECURED project: D2.1-Interim report on Data anonymization, de-anonymization and Synthetic data generation techniques, tools and services, Palmieri, Paolo Editor. 2024.

[9] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[10] A. Følstad and K. Kvale, "Customer journeys: a systematic literature review," *Journal of Service Theory and Practice*, 2018.

[11] J. Rudkowski, C. Heney, H. Yu, S. Sedlezky, and F. Gunn, "Here today, gone tomorrow? mapping and modeling the pop-up retail customer journey," *Journal of Retailing and Consumer Services*, vol. 54, p. 101698, 2020.

[12] A.-M. Kranzbühler, M. H. Kleijnen, and P. W. Verlegh, "Outsourcing the pain, keeping the pleasure: effects of outsourced touchpoints in the customer journey," *Journal of the Academy of Marketing Science*, vol. 47, pp. 308–327, 2019.

[13] B. Bosio, K. Rainer, and M. Stickdorn, "Customer experience research with mobile ethnography: A case study of the alpine destination serfaus-fiss-ladis," in *Qualitative consumer research*. Emerald Publishing Limited, 2017, vol. 14, pp. 111–137.

[14] C. Meyer, A. Schwager *et al.*, "Understanding customer experience," *Harvard business review*, vol. 85, no. 2, p. 116, 2007.

[15] M. Ieva and C. Ziliani, "The role of customer experience touchpoints in driving loyalty intentions in services," *The TQM Journal*, 2018.

[16] C. M. Voorhees, P. W. Fombelle, Y. Gregoire, S. Bone, A. Gustafsson, R. Sousa, and T. Walkowiak, "Service encounters, experiences and the customer journey: Defining the field and a call to expand our lens," *Journal of Business Research*, vol. 79, pp. 269–280, 2017.

[17] F. Ponsignon, F. Durrieu, and T. Bouzdine-Chameeva, "Customer experience design: a case study in the cultural sector," *Journal of Service Management*, vol. 28, no. 4, pp. 763–787, 2017.

[18] A. Pantouvakis and A. Gerou, "The theoretical and practical evolution of customer journey and its significance in services sustainability," *Sustainability*, vol. 14, no. 15, p. 9610, 2022.