Ref. Ares(2024)8206500 - 19/11/2024

NGI TRUST CHAIN

TECHNOLOGICAL FRAMEWORK REQUIREMENTS AND SPECIFICATION REPORT

29/02/2024

Funded by the European Union

TrustChain Project. Funded by the European Union under GA No 101093274. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Grant Agreement No.: 957338
Call: H2020-ICT-2020-1
Topic: ICT-54-2020
Type of action: RIA

D3.5 TECHNOLOGICAL FRAMEWORK REQUIREMENTS AND SPECIFICATION REPORT

Work package	WP 3
Task	Task 3.2, Task 3.3
Due date	31/01/2024
Submission date	29/02/2024
Deliverable lead	UL
Version	1.0
Authors	Petar Kochovski (UL), Vlado Stankovski (UL), Vasilios A. Siris (AUEB), Thanasis G. Papaioannou (NKUA), Pablo Vela (ALA), Akanksha Dixit (ICS), Veniamin Boiarkin (ICS)
Reviewers	Michal Krol (ICS), Alexander Herranz (ALA)
Abstract	This deliverable provides initial specification information on the TRUSTCHAIN framework requirements, its specification and also description of its components (including those developed by third parties during Open Call 1).
Keywords	Decentralisation, blockchain, trustworthy content, data traceability, trustworthy knowledge exchange, privacy protection, digital wallet, digital identity, human-centric,

Funded by the European Union

service interoperability

Document Revision History

Version	Date	Description of change	List of contributor(s)
0.1	04/09/2023	Initial deliverable structure	Petar Kochovski (UL)
0.2	18/09/2023	Structure rediscussed	Petar Kochovski (UL)
0.3	13/12/2023	Contributions to all sections	Petar Kochovski (UL) Vlado Stankovski (UL) Akanksha Dixit (ICS) Vasilios Siris (AUEB) Thanasis Papaioannou (NKUA)
0.4	20/12/2023	List of abbreviations, and document template update	Petar Kochovski (UL) Vlado Stankovski (UL)
0.5	17/01/2024	Architecture design and description updated	Petar Kochovski (UL) Vlado Stankovski (UL) Vasilios Siris (AUEB) Thanasis Papaioannou (NKUA) Veniamin Boiarkin (ICS)
0.6	31/01/2024	Added component descriptions	Petar Kochovski (UL)
0.7	16/02/2024	Framework setup description added	Pablo Vela (ALA)
0.8	16/12/2024	First draft review and preliminary final version	Petar Kochovski (UL)
0.9	21/01/2024	Internal review	Michal Krol (ICS) Alexander Herranz (ALA)
0.9.1	22/01/2024	Addressing internal review comments	Petar Kochovski (ALA) Vlado Stankovski (UL)
1.0	29/02/2024	Final review before submission	Caroline Barelle (ED)

Funded by the European Union

DISCLAIMER

The information, documentation and figures available in this document are written by the TRUSTCHAIN project’s consortium under EC grant agreement 101093274 and do not necessarily reflect the views of the European Commission. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein. The information in this document is provided “as is” and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. Moreover, it is clearly stated that the TRUSTCHAIN Consortium reserves the right to update, amend or modify any part, section or detail of the document at any point in time without prior information.

Funded by the European Union

The TRUSTCHAIN project is funded by the European Union's Horizon Europe Research and Innovation programme under grant agreement no. 101093274.

COPYRIGHT NOTICE

© 2023 TRUSTCHAIN

This document may contain material that is copyrighted of certain TRUSTCHAIN beneficiaries and may not be reused or adapted without prior permission. All TRUSTCHAIN Consortium partners have agreed to the full publication of this document. The commercial use of any information contained in this document may require a license from the proprietor of that information. Reproduction for non-commercial use is authorised provided the source is acknowledged.

The TrustChain Consortium is the following:

Participant number	Participant organisation name	Short name	Country
1	EUROPEAN DYNAMICS LUXEMBOURG SA	ED	LU
2	F6S NETWORK IRELAND LIMITED	F6S	IE
3	UNIVERZA V LJUBLJANI	UL	SI
4	ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS - RESEARCH CENTER	AUEB	EL
5	FUNDACION CIBERVOLUNTARIOS	CIB	ES
6	CONSORCIO RED ALASTRIA	ALA	ES
7	TIME.LEX	TLX	BE
8	CITY UNIVERSITY OF LONDON	ICS	UK
9	NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS	NKUA	EL

Funded by the European Union

EXECUTIVE SUMMARY

This document is the "D3.5 Technological Framework requirements and specification report" deliverable of the TRUSTCHAIN project, which is funded by the Horizon Europe Research & Innovation program "TRUSTCHAIN– Fostering a Human-Centered, Trustworthy and Sustainable Internet".

The framework specification is the outcome of Task 3.2 and Task 3.3, which followed the completion of TRUSTCHAIN Open Call #1. It summarizes the current achievements of the OC-1 projects that started in July 2023 and incorporates them into a unified infrastructure design. This document will also serve as an input for the upcoming TRUSTCHAIN Open Calls projects. It will be updated regularly to reflect the progress made by the third parties funded under the Open Calls and will be used as a reference for developing and deploying the full TRUSTCHAIN software ecosystem.

The framework specification describes the software components that will make up the TRUSTCHAIN ecosystem, the APIs that will enable these components to interact, and the user-facing interfaces that the ecosystem will offer.

Funded by the European Union

TABLE OF CONTENTS

Funded by the European Union

Funded by the European Union

LIST OF FIGURES

Funded by the European Union

ABBREVIATIONS

AI	Artificial Intelligence	QEAA	Qualified Electronic Attestations of Attributes
API	Applicaton Programming Interface	RDF	Resource Description Framework
CCPA	California Consumer Privacy Act	SaaS	Software as a Service
DAA	Direct Anonymous Attestation	SLA	Service Level Agreement
DAO	Decentralized Autonomous Organization	SSI	Self-sovereign identities
DID	Decentralized identifiers	TAO	Trusted Accreditation Organisation
DIF	Decentralized Identity Foundation	TLS	Transport Layer Security
DLT	Decentralised Ledger Technology	TPM	Trusted Platform Module
DMA	Digital Market Act	TRL	Technology Readiness Level
DSA	Digital Services Act	UN	United Nations
EBSI	European Blockchain Services Infrastructure	URI	Uniform Resource Identifier
EUDI	EU Digital Identity	VC	Verifiable Credential
EV	Electric Vehicle	W3C	World Wide Web Consortium
GDPR	General Data Protection Regulation	ZKP	Zero-Knowledge-Proof
IBFT	Istanbul Byzantine Fault Tolerant	QEAA	Qualified Electronic Attestations of Attributes
JSON-LD	Javascript Object Notation for Linking Data	RDF	Resource Description Framework
KMIP	Key Management Interoperability Protocol	SaaS	Software as a Service
ML	Machine Learning	SLA	Service Level Agreement
MoU	Memorandum of Understanding	SSI	Self-sovereign identities
NFT	Non-fungible token	TAO	Trusted Accreditation Organisation
OC	Open Call	TLS	Transport Layer Security
OWL	Web Ontology Language	TPM	Trusted Platform Module
PCR	Platform Configuration Register	TRL	Technology Readiness Level
PID	Personal Identification Data	UN	United Nations
PoA	Proof of Authority	URI	Uniform Resource Identifier
POS	Proof of Stake	VC	Verifiable Credential
POW	Proof of Work	W3C	World Wide Web Consortium

Funded by the European Union

1 INTRODUCTION

This first version of the TRUSTCHAIN framework requirements and specification is the combined result of the initial architecture designed in the duration of the project's first year, and of revisions made during the execution of the third-party projects funded under OC1. This framework specification is a result of the discussions between the consortium partners, the funded third parties, and the Advisory Board. Its main goal is to define the requirements, provide common guidelines and recommendations to the applicants of the forthcoming open calls regarding important design choices that go beyond their own projects.

These activities will continue during the TRUSTCHAIN project and deliver their results in version 2 of the Framework Specification that is due at M35 in D3.6, including feedback and lessons learned from the remaining open calls.

The rest of this deliverable is organized as follows:

- Section 2 addresses the TRUSTCHAIN requirements from visionary, compliance, and human-centric perspective;
- Section 3 describes the initial version of the TRUSTCHAIN architecture; the architecture defines the high-level services and components that will implement the TRUSTCHAIN vision and describes the services that were implemented by third parties during OC1;
- Section 4 defines the application programming interfaces for the TRUSTCHAIN framework, i.e. the APIs that will be presented to application developers and users of the TRUSTCHAIN framework;
- Section 5 describes the planned setup of the TRUSTCHAIN framework which will be built on top of the ALASTRIA network;
- Section 6 delivers the conclusions of this deliverable.

Funded by the European Union

2 TRUSTCHAIN REQUIREMENTS

The TRUSTCHAIN project starts from a set of requirements that are broad and can support the development of many future applications. An enabling technology and standards to realize the concept of decentralized digital identities, has been for example, the W3C DID methods and standards. While specific requirements for the future digital identity can be effectively implemented (e.g. micro-credentials) by using such relaxed standards, other aspects may be difficult or impossible to realise without the use of decentralized infrastructures, such as blockchains (e.g. voting, the flexible use of tokens within applications). Hence, instead of leaning on one side or another, our TRUSTCHAIN project aims at coming to common terms:

- The limits of specific standards and technologies related to advanced user scenarios in specific aspects of human lives, such as (1) the use of the digital identity, (2) achieving privacy and security, (3) exerting voting rights and deploying economic mechanisms that may be instrumental for achieving sustainability, (4) our vision of various blockchain networks that would require to address interoperability aspects, and (5) the core substance of our project, an answer to the questions, can we use these technologies in order to address the humanity problems of the day: sustainability, human-rights, energy-efficiency, our care for the environment and so on.
- The necessary aspects of technology that should be influenced by society, such as legal and ethical compliance. This is important to our project as we intend to influence how technology is being used according to societal norms, and not vice-versa. This requires to re-assess the Collingridge Dilemma [1]. The Collingridge dilemma is a methodological quandary in which efforts to influence or control the further development of technology face a double-bind problem: (1) An information problem: impacts cannot be easily predicted until the technology is extensively developed and widely used. (2) A power problem: control or change is difficult when the technology has become entrenched. Since blockchains arguably allow for decentralization, they have the inherent ability to form basis for the evolution of sustainable ecosystems of the future. Here, our exploration intends to see how society can gradually use a technology to influence its sustainable, secure, and ethical evolution.
- Finally, at practical level, every technology is impaired by its usability aspects. Sometimes, the benefits of using a specific technology are so high that the users adapt themselves to be able to use it. However, in the TRUSTCHAIN project we do focus on users. Though usability stands as the paramount challenge in technology adoption, we recognize that ensuring usability is vital for the success of any emerging technology. While users can adapt to new innovations, widespread adoption hinges on the technology's usability. From the viewpoint of existing blockchain applications, it is currently difficult to foresee how a

Funded by the European Union

plethora of digital wallets can be used in order to achieve smooth execution of applications used by everyday technology users. From this viewpoint, it is our sole goal to fully understand the human-to-technology interactions and to see how any obstacles to usability and trustworthiness can be overcome.

The TRUSTCHAIN project requirements gathering, and analysis is a continuous process. The requirements for the TRUSTCHAIN open calls are designed in a way to highlight the above three categories of requirements, even though, sometimes implicit in the call texts. By exploring different use cases, protocols, and APIs, DID methods and eIDAS2 compliant solutions, we do address our requirements for the human-centred internet of the future. Following is a summary of some of our core OC requirements.

2.1 FRAMEWORK REQUIREMENTS

2.1.1 TECHNICAL REQUIREMENTS

Within the TRUSTCHAIN ecosystem, technical requirements serve as pillars to promote responsible and ethical behavior in data exchange, privacy protection, and intellectual property rights management. For instance, the implementation of end-to-end encryption, zero-knowledge proofs and decentralized storage mechanisms ensures the confidentiality and integrity of data shared among participants, upholding privacy rights and preventing unauthorized access. Moreover, the utilization of blockchain technology with smart contracts enables transparent and immutable records of transactions, facilitating accountability and trustworthiness in data exchanges. These technical measures not only safeguard sensitive information but also empower individuals to maintain control over their digital identities and personal data, fostering a culture of transparency and respect for privacy within the ecosystem. Additionally, adherence to interoperability standards and open-source development practices ensures compatibility, reliability, and community-driven innovation, further strengthening the foundation of responsible and ethical conduct in the TRUSTCHAIN ecosystem.

The TRUSTCHAIN ecosystem prioritizes security, transparency, and trustworthiness to promote ethical behavior in data exchange and intellectual property rights management. Collaboration with end-users, including vulnerable populations, is key to co-creating user-friendly solutions that address privacy, scalability, and energy efficiency concerns. Solutions must be designed, piloted, and validated with specific end-users in mind, demonstrating a TRL7 in real settings by the end of their funding period. Compatibility with existing frameworks and standards, along with cross-border data sharing considerations, are essential. Additionally, proposals should outline how the solution can be utilized by other TRUSTCHAIN developers, aligning with other Open Calls and maximizing impact through joint activities facilitated by the TRUSTCHAIN consortium.

Funded by the European Union

2.1.2 SUSTAINABILITY REQUIREMENTS

Various emerging technologies currently pose a significant environmental impact that needs to be evaluated against any potential benefits they may offer. For instance, the adoption of blockchain technology in various industries offers benefits such as increased transparency and security. However, the energy consumption associated with blockchain operations, particularly in proof-of-work consensus mechanisms, raises concerns about its environmental sustainability [2]. Participants in the TRUSTCHAIN framework are requested to provide a concise assessment of the trade-offs, considering the benefits of using the technology from one viewpoint and the potential energy-inefficiency from another and are encouraged to minimise their solution's energy consumption.

2.1.3 HUMAN CENTRIC REQUIREMENTS

The TRUSTCHAIN ecosystem requires its project to be user-centric, that is, that the end-users of such solutions are at the centre of the design process. The aim of this is to ensure adoption, involve the community in the development process and to produce applications that generate social value.

To ensure the implementation of this user-centric approach, the requirements briefly exposed below here encompass the checklist of what must encompass a user-centric approach while developing a winning TRUSTCHAIN project.

- The winning project must assess the needs of the target user's community. This could be done through different ways, such as rigorous (and pertinent to the case study) literature review, or more ideally, preliminary qualitative research (for example, focus groups, in-depth interviews, online ethnography) with the objective to explore the needs and expectancies towards your solution.
- The methodologies of choice must be defined for the subsequent testing and validation rounds. A specification of the methodology of choice and the reasons behind it. This must be accompanied by a justified the expected number and profiles of onboarded users (the sample selection)
- A scheduled recruitment plan for those users: with channels and dates for the onboarding in the pilot sessions
- The technical capacity of the team (in terms of user research) to carry out these activities.

The selected third-party projects as part of the TRUSTCHAIN open calls then must undergo design, implementation, pilot testing, and validation using a specific predefined and justified set of end users in an identified use case. The co-creation and validation approach should be clearly outlined in the applicant's proposal. Initially, target user groups need to be defined and involved in the co-creation process. Then,

Funded by the European Union

accessibility standards should be integrated during onboarding according to the vulnerable collectives' approach. Subsequently, a roadmap detailing the appropriate methodologies, objectives, testing phases, and sample size should be established. The sample must be both representative and randomized within the relevant characteristics of the target population. Users should contribute improvement proposals and insights during the co-creation process, with possible input from other stakeholders, such as developers or business partners, which must be justified in the deliverable documentation.

2.1.4 INTEROPERABILITY REQUIREMENTS

Interoperability is a crucial aspect of the TRUSTCHAIN project framework. Therefore, to ensure the success and effectiveness of the TRUSTCHAIN framework, it is essential to establish robust interoperability requirements across its various components. The interoperability of the framework can be achieved through different approaches, such as the standardized protocols, cross-platform compatibility, APIs and middleware, semantic interoperability, and cross-chain communication.

- Standardized Protocols

  The TRUSTCHAIN project will adopt standardized protocols for communication and data exchange between its different modules and applications. This includes protocols for transmitting data securely (e.g. TLS), verifying identities (e.g. DID, SSI), smart contracts best practices and recommendations (e.g. EVM-compatibility [3], OpenZeppelin [4]), and facilitating consensus mechanisms.

- Cross-Platform Compatibility

  Interoperability requirements will ensure that the TRUSTCHAIN framework can seamlessly integrate with different platforms, operating systems, and devices. For that purpose, the framework will provide well-defined APIs and middleware layers that abstract the complexity of underlying technologies, enabling easy integration with external systems and applications. These APIs will adhere to industry standards and best practices to facilitate interoperability. The interfaces enabled by the framework are described in detail in Section 3.

- Semantic Interoperability

  To enable meaningful exchange of data and information, the TRUSTCHAIN framework will support semantic interoperability standards. This involves use of semantic web standards (e.g. RDF, OWL, SPARQL), adopting ontologies to model data (e.g. defining ontologies for concepts such as transactions, identities, and permissions enables systems to interpret and process data consistently across different contexts), and adhering to linked data principles (e.g. URI, RDF triples).

- Cross-chain communication

Funded by the European Union

Given the multichain support within the TRUSTCHAIN framework, interoperability requirements should facilitate seamless communication and data transfer between different blockchain networks. This may involve implementing cross-chain protocols, interoperability standards, and atomic swap mechanisms. As a result, Open Call 5 will be solely addressing this topic.

2.1.5 SECURITY, TRANSPARENCY AND TRUSTWORTHINESS REQUIREMENTS

In the TRUSTCHAIN ecosystem, stringent requirements for security, transparency, and trustworthiness are foundational principles that promote responsible and ethical behavior in data exchange, privacy protection, and intellectual property rights management. Robust security measures, including encryption, multi-factor authentication, and decentralized storage, ensure the confidentiality and integrity of data shared within the ecosystem, safeguarding it from unauthorized access or tampering. Transparent data handling practices, such as publicly verifiable audit trails and clear consent mechanisms, empower users with visibility and control over their personal information, fostering trust and accountability among participants. Moreover, adherence to intellectual property rights laws and transparent licensing agreements ensures that creators receive fair recognition and compensation for their contributions, promoting innovation and creativity within the ecosystem. By prioritizing security, transparency, and trustworthiness, the TRUSTCHAIN ecosystem cultivates a culture of ethical behavior and mutual respect among its members, establishing a solid foundation for sustainable growth and collaboration.

2.1.6 LEGAL AND REGULATORY REQUIREMENTS

To participate in the TRUSTCHAIN framework, applicants are asked to outline existing or emerging privacy-enhancing data sharing platforms and infrastructure standards they plan to adhere to or contribute to. In addition, they should explain how their project aligns with the Digital Services Act (DSA) and the Digital Market Act (DMA). Applicants are also requested to present identity platforms and infrastructure standards they intend to comply with or contribute to. Lastly, they should propose new economic and business models for the ecosystem economy, focusing on user-centric data management, privacy considerations, and legal and regulatory compliance, including GDPR-compliance and verification of data processing activities. The legal and regulatory requirements ensure compliance with regulatory frameworks and governance policies governing data exchange, privacy protection, and intellectual property rights within the TRUSTCHAIN ecosystem. This involves implementing mechanisms for auditability, transparency, and accountability across interconnected systems and applications. Such requirements are essential within the TRUSTCHAIN ecosystem to ensure adherence to regulatory frameworks, governance policies, and ethical standards.

Funded by the European Union

- Data Privacy and Protection Policies

  Implementing robust data privacy and protection policies within the TRUSTCHAIN framework ensures compliance with regulations such as GDPR (General Data Protection Regulation) or CCPA (California Consumer Privacy Act). This involves incorporating mechanisms for data anonymization, encryption, access control, and consent management to safeguard sensitive information exchanged within the ecosystem.

- Audit Trails and Immutable Records

  The projects that are being part of the TRUSTCHAIN framework are requested to present in a clear and concise manner any existing and/or emerging privacy-enhancing data sharing platform (i.e., GAIA-X[1] & IDS[2], DECODE[3]) / infrastructure standards with which they intend to comply with, or they wish to contribute to within the course of the proposed project. It is also recommended to identify how the project aligns with the Digital Services Act (DSA) and the Digital Market Act (DMA). Moreover, implementing audit trails and maintaining immutable records of transactions within the TRUSTCHAIN framework enables traceability and accountability. By recording all interactions and changes to data in a tamper-proof manner, stakeholders can track the history of transactions and verify compliance with regulatory requirements, governance policies, and contractual agreements.

- Ethical Guidelines and Governance Frameworks

  Establishing ethical guidelines and governance frameworks for data exchange, privacy protection, and intellectual property rights promotes responsible and ethical behaviour within the TRUSTCHAIN ecosystem. This involves defining principles, values, and best practices to guide decision-making and ensure that ethical considerations are considered in all interactions and transactions.

  These requirements are mainly fostered in the context of WP2 with a direct interaction between the TRUSTCHAIN funded projects and the TRUSTCHAIN core team, which for the legal and regulatory activities is led by TLX.

  In the following are the ethical and governance guidelines that are considered:

  - Principle of Data Minimization: Encourage stakeholders within the TRUSTCHAIN ecosystem to collect and process only the minimum amount of data necessary to achieve specific objectives. This principle promotes privacy by design and minimizes the risk of unauthorized access or misuse of personal information.

[1] https://gaia-x.eu/
[2] https://internationaldataspaces.org/
[3] https://decodeproject.eu/index.html

Funded by the European Union

- Informed Consent: Require that individuals provide informed consent before their data is collected, processed, or shared within the TRUSTCHAIN ecosystem. This ensures that individuals are aware of how their data will be used and have the opportunity to make informed decisions about its sharing and disclosure.
- Transparency and Accountability: Establish mechanisms for transparent data practices and accountability within the TRUSTCHAIN ecosystem. This includes providing clear information about data handling practices, establishing accountability for data breaches or misuse, and enabling individuals to access and review their own data.
- Fair and Non-Discriminatory Data Use: Ensure that data collected and processed within the TRUSTCHAIN ecosystem is used fairly and without discrimination. This involves preventing the use of data for discriminatory purposes, such as profiling individuals based on sensitive attributes like race, ethnicity, or gender.
- Respect for Intellectual Property Rights: Establish guidelines for respecting intellectual property rights within the TRUSTCHAIN ecosystem, including copyrights, patents, and trademarks. This involves obtaining appropriate permissions and licenses before using or sharing intellectual property and respecting the rights of content creators and innovators.
- Ethical Use of Emerging Technologies: Consider the ethical implications of emerging technologies, such as artificial intelligence (AI) and blockchain, within the TRUSTCHAIN ecosystem. This includes addressing concerns related to algorithmic bias, data privacy, and transparency in decision-making processes, as well as ensuring the responsible use of blockchain technology to protect user privacy and prevent misuse of data.
- Community Engagement and Feedback: Foster a culture of community engagement and feedback within the TRUSTCHAIN ecosystem (e.g. coaching meetings, plenary meetings with the third party funded projects), allowing stakeholders to voice concerns, provide input on ethical issues, and contribute to the development of ethical guidelines and governance frameworks.

The challenges tackled under the open calls are explained in the following.

2.2 OPEN CHALLENGES

In the fast-growing landscape of digital technologies, the promises of enhanced security, privacy, and sustainability offered by decentralized solutions are gaining attention. However, mainstream adoption of those systems still encounters multiple challenges that should be addressed by the TRUSTCHAIN applications.

- OC1 : Decentralised Digital Identity Systems Challenges

The current ecosystem of Decentralised Digital Identity systems experienced a rapid growth in the last couple of years. Today's identity systems are faced with a multitude of challenges due to the centralised nature of the internet. The internet was initially developed without the human in the loop. However, with the exponential growth of the online usage, evolution of decentralised systems and the power of cloud and edge computing has made the centralised model obsolete for many future online applications. In order to develop a usable and interoperable decentralised future internet, some of the identity challenges that exist today need to be addressed. These include:

1. Usability, Privacy, and Compliance Issues: Existing identity systems lack usability, privacy, transparency, and compliance with GDPR. They often fall short in inclusivity.
2. Complexity of Technologies: The incorporation of advanced technologies like zero-knowledge-proof (ZKP) poses challenges [5], especially for non-tech-savvy users, impeding seamless integration.
3. Trust Deficit in Identity Credentials: There is a lack of trust in the way identity credentials are shared and used across multiple online services.
4. Data Minimization Violation: Authentication systems often request more identity data than necessary, violating the data minimization principle of GDPR.
5. Human-Centric Design Issues: The lack of human involvement from the initial design stages leads to a gap in understanding new technologies, hindering wider adoption.

- OC2: Privacy-Preserving Data Management Challenges

  In the current Internet, all user data is owned and managed by a few handful organizations, which dictate the terms of data exchange with third parties. In most cases, user consent is either not explicitly specified or is masked in elaborate notices. Purpose limitation and data minimization is a key data management practice that the current Internet is missing. Challenges in contemporary data management include:

1. Flawed Data Sharing Model: The existing data sharing model encourages duplication, long-term retention, and intensive collection, lacking traceability and accountability.
2. Privacy Concerns in Machine Learning: Privacy is compromised when user data is used for training machine learning models, necessitating exploration of privacy-preserving techniques like local differential privacy.
3. Illegal Data Copying: Addressing challenges related to illegal data copying is crucial for user privacy and data governance models.

Funded by the European Union

4. Lack of Trust Layer: A missing trust layer hampers the authenticity of data. Implementing mechanisms based on SSI technologies is vital for trustworthy data access and integrity. Users have little to no control over access to their personal data shared online. Therefore, automated user consent/smart user consent for data sharing needs to be implemented.

- OC3 Economics and Democracy

  Navigating the complex data exchange/trading arena involves challenges related to trust, privacy, consent, and regulatory complexities among multiple parties. Existing market mechanisms face hurdles due to dynamic data nature, market competition, fair settlement practices, and ownership rights. Addressing these requires collaborative efforts and innovative solutions to establish a secure, transparent, and ethical data exchange environment. Challenges in the data exchange include:

  1. Privacy and Security Concerns: Data trading and exchange raise significant concerns about privacy and security, risking mishandling of sensitive data and privacy breaches.
  2. Lack of Enforceable Contracts: Establishing enforceable data marketplace contracts is a challenge, requiring clear and fair service level agreements for trustworthy transactions.
  3. Unfair Data Reward Practices: The current form of data-sharing practises does not fairly reward the data owners/content producers. Data platform owners make decisions around the terms and conditions of data sharing.
  4. Undefined Pricing Models: Establishing enforceable data marketplace contracts for data exchange is missing; Clear and fair service level agreements for both sellers and buyers needs to be in place for a trustworthy marketplace.
  5. Data Provenance Verification: Verifying the source of data becomes challenging, necessitating solutions like open reputation management with careful design considerations.
  6. Complex Data Governance: Establishing clear data governance practices, including data access controls and usage policies, is essential for responsible data exchange. Data governance frameworks can be complex to implement and enforce.

- OC4. Multi-chains Support for NGI Protocols

  In the pursuit of decentralized Internet services and protocols, the need for multi-chains support within the Next Generation Internet (NGI) protocols arises. The challenges associated with achieving seamless interoperability and collaboration among diverse blockchain networks are substantial:

1. Interoperability Challenges: The physical decentralization of the Internet introduces vulnerabilities and privacy concerns. Existing DLT and blockchain networks lack interoperability, requiring bridges, gateways, and standardization to facilitate communication and collaboration between them.
2. Security Concerns: Ensuring the security of multi-chain communication becomes paramount, as vulnerabilities in one blockchain may have cascading effects on interconnected networks. Developing robust security measures to protect decentralized protocols is essential.

- OC5. Green Scalable and Sustainable DLTs

  The environmental impact of digital technologies, particularly decentralized technologies like blockchains, presents a significant challenge. Achieving scalability and sustainability while minimizing carbon emissions poses complex issues:

  1. Energy Consumption and Carbon Emissions: The rapid growth of Internet use contributes to a substantial share of global energy supplies. Consensus mechanisms in decentralized environments, such as PoW (Proof of Work) and PoS (Proof of Stake), heavily rely on energy-intensive processes, impacting the sustainability of blockchains and increasing carbon emissions.
  2. Sustainability Goals: The global community, represented by entities like the UN and EU, has set ambitious sustainability goals, aiming for affordable and green energy access and substantial emission reductions. Aligning decentralized technologies with these goals requires innovative approaches to reduce carbon footprints.
  3. Consensus Mechanism Impact: The choice of consensus mechanisms significantly influences the performance and sustainability of blockchains. Balancing the need for security with energy efficiency becomes a critical consideration in developing and adopting consensus protocols.
  4. Green Energy Alternatives: As the demand for green energy rises, exploring alternative approaches like local microgrids and markets becomes crucial. Integrating blockchain to establish trust, manage energy policies, and facilitate secure transactions requires careful consideration for minimizing environmental impact.

The challenges presented, from decentralized identity to data governance, economic considerations, interoperability, and environmental impact, underscore the complexity of this transformative journey towards a more secure, private, and sustainable future. The potential benefits of decentralized technologies are immense, but their realization requires overcoming these challenges.

Funded by the European Union

3 TRUSTCHAIN ARCHITECTURE

The TRUSTCHAIN project framework encompasses an innovative protocol suite and applications that address diverse requirements and topics. These include decentralized identity management, cryptography, secure data management, privacy-aware computation, anonymized proofs and consensus, intellectual property, data value sharing, trustworthy economics and democracy, multichain support, and solutions for more environmentally friendly, scalable, and sustainable decentralized ledger technologies. In the subsequent part of this section, we will delve into further details regarding the architectural design and provide a more comprehensive description of the framework's components.

3.1 ARCHITECTURE DESIGN

In the fast-growing landscape of digital technologies, the promises of enhanced security, privacy, and sustainability offered by decentralized solutions are gaining attention. However, mainstream adoption of those systems still encounters multiple challenges that should be addressed by the TRUSTCHAIN applications.

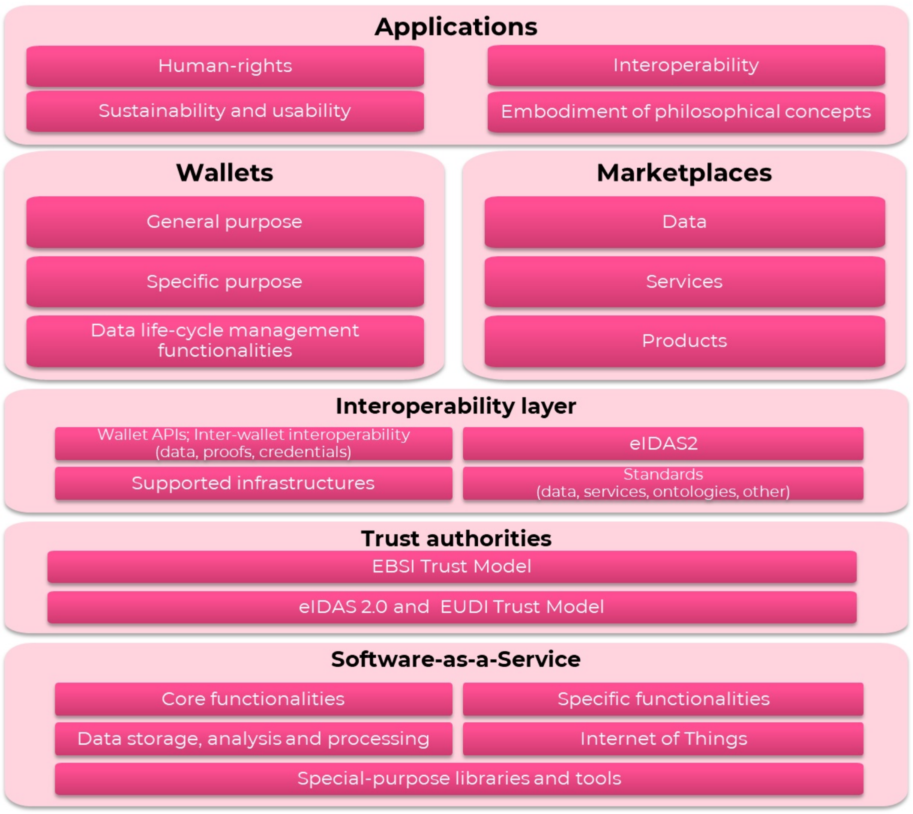
Applications
Human-rights
Interoperability
Sustainability and usability
Embodiment of philosophical concepts
Wallets
General purpose
Specific purpose
Data life-cycle management functionalities
Marketplaces
Data
Services
Products
Interoperability layer
Wallet APIs; Inter-wallet interoperability (data, proofs, credentials)
eIDAS2
Supported infrastructures
Standards (data, services, ontologies, other)
Trust authorities
EBSI Trust Model
eIDAS 2.0 and EUDI Trust Model
Software-as-a-Service
Core functionalities
Specific functionalities
Data storage, analysis and processing
Internet of Things
Special-purpose libraries and tools

FIGURE 1 TRUSTCHAIN LAYERED-ARCHITECTURE

Funded by the European Union

On one side, we categorize specific software solutions from the viewpoint of the use cases they address, the technology deployment options (e.g. running as an API, Smart Contract, used as a new library), from the viewpoint of the usability and user-centric aspects and compliance. Under other deliverables we focus more on the economic aspects of the delivered software tools and services.

3.1.1 SOFTWARE-AS-A-SERVICE

The Software as a Service (SaaS) layer is the layer at the bottom of the TRUSTCHAIN framework's architecture, providing essential functionalities and infrastructure to support its operations. This layer encompasses a set of core functionalities that form the backbone of the framework. These core functionalities include identity management, transaction processing, consensus mechanisms, and security protocols. They ensure the integrity, reliability, and security of the TRUSTCHAIN ecosystem's operations.

In addition to core functionalities, the SaaS layer also includes specific functionalities tailored to meet the diverse needs of users and applications within the ecosystem. These specific functionalities address domain-specific requirements such as supply chain management, healthcare data exchange, financial transactions, or intellectual property rights management. By offering specialized solutions, the TRUSTCHAIN framework can accommodate a wide range of use cases and industries.

The SaaS layer also incorporates robust data storage, analysis, and processing components to manage the vast amount of data generated and exchanged within the ecosystem. This includes distributed ledger technology for decentralized data storage, as well as tools and algorithms for data analytics, real-time processing. These components enable users to derive valuable insights from the data stored on the TRUSTCHAIN framework.

Furthermore, the SaaS layer includes the Internet of Things (IoT) to facilitate seamless communication and interaction between physical devices and the TRUSTCHAIN framework. IoT devices can securely exchange data with the whole TRUSTCHAIN ecosystem, enabling applications such as supply management and asset tracking, environmental monitoring, and smart city infrastructure management.

Finally, the SaaS layer includes special-purpose libraries and tools to streamline development, deployment, and integration efforts within the TRUSTCHAIN ecosystem. These libraries provide pre-built components, protocols, and templates for common tasks such as cryptographic operations, protocol implementation, and smart contract development. By offering a comprehensive set of tools and functionalities, the SaaS layer empowers developers and users to leverage the full capabilities of the TRUSTCHAIN framework while promoting interoperability, scalability, and ease of adoption.

Funded by the European Union

3.1.2 TRUST AUTHORITIES

EBSI (European Blockchain Services Infrastructure)[4] and eIDAS 2.0[5] - EU Digital Identity (EUDI) Wallet offer robust trust and governance models that are essential for ensuring the reliability and security of digital identities and transactions within the European Union. This section presents an overview of the EBSI and EUDI Wallet trust and governance models.

- EBSI (European Blockchain Services Infrastructure) Trust Model [6]

  EBSI employs a distributed trust model leveraging DIDs and blockchain technology. The model involves two roles:

  - Trusted Accreditation Organisation (TAO), which verifies, accredits and manages the entities, i.e. Trusted Issuers, which issue electronic documents.
  - Trusted Issuer, which is responsible for issuing Verifiable Credentials (VCs) and for managing DID documents.

  Trusted Issuers operate in a specific sector/domain and a specific geographic area, and they are allowed to issue certain types of Verifiable Credentials. For example, in the education domain, the Ministry of Education of a country, acting as the TAO, is responsible for accrediting the Universities of that country. Universities, acting as Trusted Issuers, are legal entities authorised to issue certain types of credentials, namely diplomas according to a specific Trusted Schema. TAOs also register the Trusted Schemas associated with the Verifiable Credentials.

  EBSI's distributed ledger acts as a public registry of Trusted Issuers, containing the list of trusted legal entities that are accredited by TAOs to issue certain types of credentials. Moreover, the TAO trust model involves a multilevel hierarchy, which has at the top a root TAO, which is the Ministry of Education in the case of University credentials. The root TAOs accredits a sub-TAO, which can be an Accreditation Body for Higher Education Institutions in the case of University credentials, to issue accreditations to Trusted Issuers, which are Universities in the case of University credentials. Trust Lists (or Trust Registries) stored on the EBSI ledger provide an implementation of EBSI's trust model offering high availability, accountability, resilience (avoidance of single points of failure and resistance to cyber attacks).

  Root accreditation promotes a legal entity into the first-root Trusted Accreditation Organisation (TAO). The registration of a Verifiable Authorisation for the root TAO starts a new trust chain. The next accreditation step promotes a legal entity into a Trusted Accreditation Organisation (sub-TAO) who can accredit other legal entities, according to the use-case policies. Finally, the third accreditation step promotes a

[4] https://ec.europa.eu/digital-building-blocks/sites/display/EBSI
[5] https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation

Funded by the European Union

legal entity into a Trusted Issuer, which can issue Verifiable Credentials according to the use-case policies.

Once accredited, EBSI' framework enables Trusted Issuers to manage their own keys, according to EBSI's DID method, independently of the TAO.

For TAOs, the EBSI ledger records the DID and DID Document of TAO, TAO's Verifiable Authorisation, TAO's Verifiable Accreditation, and the Accreditation Credential schema. The latter describes the data model for Organisations they are allowed to accredit, the Applicable jurisdiction, and the type of Verifiable Credential.

For Trusted Issuers, the EBSI ledger records the DID and DID Document of Trusted Issuer, the Issuer's Verifiable Accreditation, and the Verifiable Credential schema, which describes the data model for Verifiable Credentials they are allowed to issue.

Holders configure wallets that hold Verifiable Credentials and interact with Verifiers/Relying Parties, which request Verifiable Credentials.

- eIDAS 2.0 and EUDI Wallet Architecture and Reference Framework Trust Model

  The eIDAS 2.0 and EU Digital Identity (EUDI) Wallet Architecture and Reference trust model includes the following roles:

  - EUDI Wallet Provider: Member States or organisations either mandated or recognized by Member States that provides a certified EUDI wallet to the citizens of the Member State.
  - PID provider: Authority responsible for issuing the Personal Identification Data (PID) of natural and legal entities in a Member State. PID is a set of data enabling the identity of a natural or legal person, or a natural person representing a legal person, to be established. PID Providers are responsible for supporting functions to verify the identity of the EUDI Wallet User in compliance with LoA High requirements, to issue PID to the EUDI Wallet, and to make available information for Relying Parties to verify the validity of the PID. The latter function is performed without the PID Provider receiving any information about the PID's use.
  - (Qualified) Trust Service Provider: Certified entity providing (Qualified) Trust Services. A Qualified Trust Service Provider is granted the qualified status by the supervisory body. A QTSP issues Qualified Electronic Attestations of Attributes (QEAAs), which satisfy the eIDAS 2.0 regulation requirements.QTSPs are also referred to as QEAA Providers. QTSPs provide information or the location of the services that can be used to request the validity status of the QEAAs, without being able to receive any information about the use of the attestations.
  - Conformity Assessment Body (CAB): Entity responsible for assessing the conformity of Wallet Providers and QTSPs. CABs are accredited by a national accreditation body and are responsible for carrying out assessments on which

Funded by the European Union

Member States will have to rely before issuing a EUDI Wallet or providing the qualified status to a Trust Service Provider.

- National Accreditation Bodies: Authorities in Member States that perform accreditation with authority derived from the Member State. National Accreditation Bodies monitor the CABs to which they have issued accreditation certificates to ensure that all the relevant regulatory requirements are fulfilled
- Authentic Source: A repository or system, which is under the responsibility of a public sector body or private entity, that contains attributes about a natural or legal person and is considered to be the primary source of that information or recognised as authentic in national law. The data from the Authentic Source is used for certifying and issuing Qualified Electronic Attestation of Attributes (QEAA). Authentic Sources are required to provide interfaces to QEAA Providers to verify the authenticity of the above attributes, either directly or via designated intermediaries recognized at a national level.
- Trust List Provider: Authority providing Trust Lists containing the authoritative entities, including the QTSPs and certified EUDI Wallet Providers. This Authority can be the National Accreditation Bodies and the EC. A Trust List contains the current status and history information about authoritative entities in a particular legal or contractual context.
- Relying Parties: Natural or legal entities that require an electronic identification or a Trust Service. In the context of EUDI Wallets, they request the necessary attributes contained within the PID dataset, QEAA/EAA from the EUDI Wallet, subject to the acceptance by the Wallet owner and within the limits of applicable legislation and rules.

Other roles related to specific Trusted services provisioning include Qualified and Non-Qualified Certificates for Electronic Signature/Seal Providers and Qualified Electronic Attestation of Attributes Schema Providers.

The aforementioned roles are defined in the EUDI Architecture and Reference Framework [7] and are a superset of the roles defined in the eIDAS regulation and the eIDAS 2.0 amendment [8].

Trust Services include the following:

- the creation, verification, and validation of electronic signatures, electronic seals or electronic time stamps, electronic registered delivery services, electronic attestation of attributes and certificates related to those services;
- the creation, verification and validation of certificates for website authentication;
- the preservation of electronic signatures, seals or certificates related to those services;
- the electronic archiving of electronic documents;

- the management of remote electronic signature and seal creation devices;
- the recording of electronic data into an electronic ledger.

QTSPs provide an interface for requesting and providing QEAAs, including a mutual authentication interface with EUDI Wallets and potentially an interface towards Authentic Sources to verify attributes.

The eIDAS 2.0 / EUDI Wallet Infrastructure trust model includes a collection of rules that ensure the legitimacy of the components and the entities involved, covering the following:

- User authentication
- Issuer identification
- Issuer registration
- Recognised data models and schemas
- Relying Parties' registration and authentication
- Mechanisms to establish the trust in a cross-domain scenario

3.1.3 INTEROPERABILITY LAYER

Our interoperability layer is meticulously crafted to facilitate seamless collaboration among diverse users and stakeholders, fostering the development of a robust ecosystem. Within the scope of the project, we explore various methodologies to achieve this goal.

The TRUSTCHAIN project adheres to interoperability principles, emphasizing the integration of loosely coupled services and systems capable of cooperative interaction, thus enabling the formation of diverse ecosystems.

We advocate for the interoperability-by-design paradigm, systematically gathering pertinent information essential for interoperability across project outcomes while maintaining a unified vision of the requisite interoperability elements.

Our approach encompasses four layers of interoperability: legal, organizational, semantic, and technical [9]. Serving as a foundational component spanning these layers, our integrated TRUSTCHAIN services governance ensures coherence and efficiency.

- Interoperability governance

  Interoperability governance refers to decisions on interoperability frameworks, institutional arrangements, organizational structures, roles and responsibilities, policies, agreements, and other aspects of ensuring and monitoring interoperability within an ecosystem. Interoperability governance is the key to a holistic approach on interoperability, as it brings together all the instruments needed to apply it.

Funded by the European Union

Within the TRUSTCHAIN framework, interoperability governance plays a crucial role in ensuring seamless integration and collaboration across diverse systems, platforms, and stakeholders. This framework encompasses the establishment of interoperability standards, protocols, and specifications to facilitate data exchange and communication between different components of the ecosystem.

- Integrated TRUSTCHAIN services governance

  In the context of the TRUSTCHAIN framework, effective interoperability governance is vital when multiple organizations are involved. This necessitates coordinated efforts and oversight by authorities mandated to plan, implement, and operate services. Governance ensures that services are managed to facilitate integration, seamless execution, reuse of services and data, and the development of new services and 'building blocks'. This governance approach spans across all layers: legal, organizational, semantic, and technical. It involves defining organizational structures, roles, responsibilities, and decision-making processes for stakeholders. Additionally, it entails imposing requirements for various aspects of interoperability, such as quality, scalability, and availability of reusable building blocks, including information sources like registries and open data portals. Clear service level agreements are established for external information/services, ensuring interoperability standards are met. A comprehensive change management plan outlines procedures to manage and control changes effectively, while a business continuity plan ensures the uninterrupted operation of TRUSTCHAIN services.

  - Interoperability agreements: organisations involved the provisioning of TRUSTCHAIN services may need to interoperability agreements. We explore this at our joint technical meetings.
  - Legal interoperability: legal interoperability is about ensuring that organisations operating under different legal frameworks, policies and strategies can work together. Additionally, legislation undergoes a ‘digital check’: to ensure that it suits not only the physical but also the digital world (e.g. the internet); to identify any barriers to digital exchange; and to identify and assess its ICT impact on stakeholders. All organizations and physical persons taking part in TRUSTCHAIN, work within its national legal frameworks.
  - Organisational interoperability: this refers to the way in which the TRUSTCHAIN partners align their business processes, responsibilities and expectations to achieve commonly agreed and mutually beneficial goals. Organisational interoperability also aims to meet the requirements of the user community by making services available, easily identifiable, accessible and user focused.
  - Business process alignment: for various partners to collaborate efficiently and seamlessly, they may need to harmonize their current business processes or establish new ones altogether.
  - Organisational relationships: service orientation, foundational to the TRUSTCHAIN service model, requires a clearly defined relationship between

Funded by the European Union

providers and consumers. This entails formalizing mutual assistance, joint action, and interconnected business processes through instruments like MoUs and SLAs among participating organizations.

- Semantic interoperability: semantic interoperability ensures that the precise format and meaning of exchanged data and information is preserved and understood throughout exchanges between parties, in other words 'what is sent is what is understood'. This is implemented through TRUSTCHAIN registries of specific metadata formats that are being used in the project.
- Technical interoperability: This covers the applications and infrastructures linking systems and services. Requirements for technical interoperability include interface specifications, interconnection services, data integration services, data presentation and exchange, and secure communication protocols, which were defined in Section 2.1.2.

3.1.4 MARKETPLACES

In the marketplace layer, we envision the trustworthy exchange of data, services and products. To this end, the marketplace layer entails the following functionality, among others:

- Mechanisms for trustworthiness assessment of transacting entities by means of social reputation systems or social recommendation systems, verifiable credentials chains, etc.
- Mechanisms for the trustworthy exchange of data/services/products, e.g., escrow smart contracts, access token mechanisms for authorized use, etc.
- Mechanisms for privacy-aware transactions employing ZKPs.
- Supply-chain management mechanisms for transaction logging and service/asset provenance.
- NFT support for digitization of physical assets, tokenization and support for multi-ownership. Co-investment schemes and fair revenue sharing models are also considered.
- Price-determination mechanisms, such as support for sealed-bid auctions (e.g., English, VCG, etc.), liquidity pools and more.
- Crypto-coin minting and management mechanisms.
- Ontologies for service description, and mechanisms for service discoverability and matchmaking between service requests and offers.
- Incentive mechanisms for service/data federation schemes including tokenization approaches.
- Decentralized governance of co-owned services and assets.

Funded by the European Union

- Supporting mechanisms for AI/ML model marketplaces.
- Mechanisms for community-driven service provision, i.e., crowd-sourcing schemes, participatory sensing, etc.

3.1.5 WALLETS

A digital wallet is an application, which allows the end-user to store, manage, view, and share their personal identity and credential information online or offline [7] based on the use-case. A digital wallet is a mobile application where the end-user can store his/her digital documents and credentials such as passport, driver's license, certifications, and so on.

Generic purpose functionalities of a digital wallet:

- Storing the end-user's identity and credentials information

  A digital wallet allows the end-user to store his personal identity information including, but not limited to name, date of birth, address, contact details, passport, national ID cards.

- Authentication and Authorization

  A digital wallet provides a secure way to the end-user to prove their identity online. Authentication mechanism, such as biometrics, fingerprint, or PIN code is used to access and use the digital wallet. Authorization features enable the end-user to control which parties may access specific information stored in the wallet.

- Verification and Validation

  A digital wallet provides the functionality to verify the authenticity of stored information. A wallet can communicate with third-party entities, such as government agencies, to confirm the validity/accuracy of the provided information.

- Privacy and Security

  Privacy-preserving techniques, such as encryption, are used to protect the information stored in a wallet from unauthorized access. Taking into account the user-centric approach, as well as regulations, such as GDPR, the end-user is provided with control over the release of specific identity details that ensures that privacy and consent are maintained. Multi-Factor Authentication can be used to enhance the authentication process.

- Interoperability

  By integrating with standardized identity protocols (e.g., DID, VC, DIDComm), a digital wallet enables seamless interactions with different platforms and systems that accept (issue/verify) digital identity and credential information.

Funded by the European Union

- Revocation and Updates of digital credentials

  A digital wallet provides a functionality to update or revoke identity information in case the information has been changed or lost. Updates can be done on a regular basis to ensure that the stored information is valid and up to date.

- Backup and Recovery

  Backup mechanisms are used to prevent data loss or device failure. Recovery mechanism enables the end-user to regain access to the digital wallet in case of forgotten credentials or device loss.

  Specific-purpose functionalities may vary depending on the use case. For example, a digital wallet can be used for managing cryptocurrencies, including the ability to store, send, receive, and trade digital assets. By integrating with mobile payment services, a digital wallet can be used for secure and convenient mobile payments. In addition, by integrating with access control systems, a digital wallet can be used to access buildings, offices, or events by storing purpose-specific credentials.

  Data life-cycle management functionalities include:

  - Identity Creation: Generation of a unique decentralized identifier (DID) for the end-users.
  - Credential Issuance: A particular entity, such as government, issues a Verifiable Credential (VC) that contains claims about the individual's identity, attributes, or qualifications.
  - Credential Presentation: The end-user may prove his identity or other information by presenting a VC to a verifier. Selective Disclosure allows the end-users to disclose only specific VC or attributes. Moreover, Zero-Knowledge Proof (ZKP) concept is used to prove the authenticity of a presented VC without disclosing the details of VC.
  - Credential Revocation and Deletion: A verifiable credential can be deleted from a digital wallet when no longer needed. In case of any changes or updates that make a VC invalid or expired, it should be revoked/deleted.

3.1.6 APPLICATIONS

The top layer of the TRUSTCHAIN framework architecture comprises diverse applications that leverage the underlying components of the framework across its layers.

- Interoperability of applications

  Achieving interoperability among different applications is crucial for widespread adoption and seamless user experiences. Some key considerations for ensuring interoperability include:

Funded by the European Union

- Standardized Protocols: Embrace and adhere to established specifications such as Decentralized Identifiers (DID), Verifiable Credentials (VC), and DIDComm, as well as interoperable protocols such as W3C standards for Verifiable Credentials, JSON-LD, and Linked Data Signatures (LD Signature).
- Decentralized Identifiers (DIDs): Implement DIDs as a core component of an application, allowing individuals to create and own their identifiers. In addition, ensure that DIDs conform to the W3C DID specification for compatibility.
- Verifiable Credentials: Support the issuance, presentation, and verification of verifiable credentials (VC) based on W3C standards. In addition, the use of common data formats (e.g., JSON-LD) for encoding verifiable credentials enhances interoperability.
- Decentralized Key Management: Decentralized key management systems can be used to enable secure and user-controlled access to cryptographic keys. Interoperability may be enhanced by ensuring compatibility with emerging standards for key management, such as the Key Management Interoperability Protocol (KMIP).
- DIDComm Messaging: Supporting and implementing DIDComm (Decentralized Identity Communication) for secure and private communication between parties contributes to an application's compatibility with other services.
- APIs and Microservices: The development of open APIs and microservices allows different applications and services to interact seamlessly with components of your application. Ensure that APIs are well-documented and conform to the relevant standards, which may attract interest to the product.
- Blockchain and Distributed Ledger Technology: Leverage Distributed Ledger Technologies (DLTs) for the decentralized storage of data, including but not limited to European Blockchain Services Infrastructure (EBSI) and Alastria blockchain. Ensuring compatibility with common DLT platforms enhances interoperability.
- Collaboration and Ecosystem Building: Active participation in collaborations and initiatives contributes to the development of a robust ecosystem. Engagement with other stakeholders is one of the key activities to promote interoperability.

- Human Rights

  Embedding respect for human rights when designing and developing applications and services is crucial to ensure ethical, equitable, and inclusive identity and information management. Some of the key aspects for upholding human rights include but not limited to:

  - Informed Consent: Prioritize informed and explicit consent in collecting, using, and sharing personal identity information. Ensure that users understand the implications of sharing their data and provide them with meaningful choices.
  - Privacy by Design: Integrating privacy by design principles into the development process is key to minimizing the impact on individuals' privacy. Some of the techniques that can be used include but are not limited to user-centric control over personal data, data minimization, data perturbation, and encryption.
  - User Empowerment: Empower individuals to have control over their personal information, allowing them to manage and share data as per their preferences. Users should be provided with the ability to revoke consent/credentials and manage access to their information.
  - Accessibility: Design applications and services with accessibility in mind to ensure that all individuals, including those with disabilities, can equally use developed solutions.
  - Human-Centric Design: Employ human-centric design methodologies to create user interfaces and experiences that are intuitive, user-friendly, and respectful of individual needs and preferences. Solicit user feedback throughout the development lifecycle.
- Sustainability

  Sustainability of applications involves ensuring the long-term viability, scalability, and environmental responsibility of the applications and components involved. Some of the key aspects for ensuring sustainability include:

  - Scalability and Performance: Design applications and services with scalability in mind to accommodate a growing number of users and transactions. Performance optimization is a key aspect of allowing the processing of a vast number of transactions (requests) without compromising efficiency.
  - Lifecycle Management: Implement proper lifecycle management of information to prevent unnecessary data accumulation.
  - Upgradability and Flexibility: Designing applications and services to be easily upgraded would allow to accommodate evolving technological standards and security requirements.

  By incorporating the aforementioned considerations into the design and development of applications and services, it is possible to build sustainable and environmentally responsible systems that align with long-term goals and values.

- Usability

  Usability is critical in the success and adoption of developed solutions and applications. Some important aspects that might be considered when designing and developing applications include:

Funded by the European Union

- User-Centric Design: Prioritize user-centric design principles to create intuitive and user-friendly interfaces considering the needs and preferences of diverse user groups.
- Intuitive User Interfaces: Design intuitive and easy-to-understand interfaces, minimizing the need for extensive user training. Streamline the onboarding process to make it easy for individuals to sign up and manage their digital identities and information.
- Clear Consent Mechanisms: Implement transparent and easy-to-understand consent mechanisms when users are required to share their information. Clearly explain the implications of granting or revoking consent.

Taking into account the abovementioned considerations allow the development of applications that are not only secure and privacy-preserving but also user-friendly, encouraging widespread adoption and engagement.

3.2 COMPONENTS DESCRIPTION

To realise the system architecture from the previous section, we will integrate software components from the TRUSTCHAIN open calls. These components (See Figure 2) are developed by third parties and selected under the TRUSTCHAIN OC1. They will be part of the final TRUSTCHAIN framework and used by the future open call projects. This section gives a short overview of these components and their interfaces for integration with the rest of the TRUSTCHAIN components.

Funded by the European Union

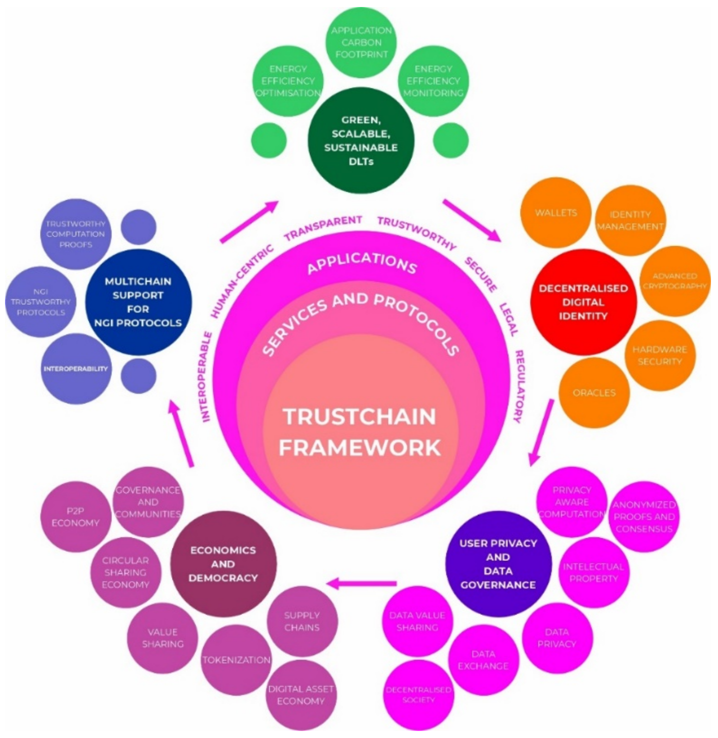

FIGURE 2 TRUSTCHAIN FRAMEWORK

Figure 3 illustrates the 13 TRUSTCHAIN OC-1 funded projects and their corelation with the baseline TRUSTCHAIN technologies.

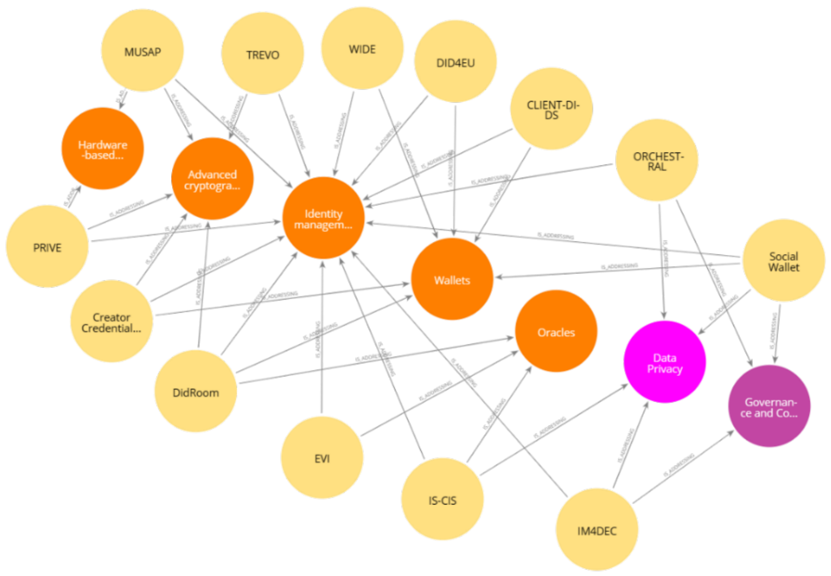

FIGURE 3 MAPPING BETWEEN THE OC-1 PROJECTS AND THE TRUSTCHAIN BASELINE TECHNOLOGIES

3.2.1 DIDroom

Description

DIDroom6 is a fully open-source multiplatform, highly modular and multifunctional Identity identity/SSI wallet, compliant with the W3C-DID and W3C-VC standards and with the current "The European Digital Identity Wallet Architecture and Reference Framework" (EUDI – ARF, version 1.0.0 from January 2023) which is the technical core of the eIDAS 2.0 regulation, as well as the OpenID for verifiable presentation standardisation. DIDroom also has advanced cryptographic and blockchain functions, including signatures, multi-signatures and blockchain interoperability (for Ethereum, Hyperledger Fabric and Sawtooth, and Planetmint). DIDroom is based on a stack of open-source components and middleware components implemented by the Dyne.org Foundation during several H2020 and EC-funded research projects.

Interfaces

Endpoint	HTTP method	Description
/credential	POST	Issues a single verifiable credential or transaction ID, preparing the corresponding credential response.
/credential_offer	GET	Provides information about an offer for a verifiable credential or transaction ID, including terms and associated details.
/authorize	GET	Initiates the authorization process for obtaining a verifiable credential.
/batch_credential	POST	Issues multiple verifiable credentials or transaction IDs in a batch, preparing the corresponding credential responses.

6 https://forkbomb.solutions/solution/didroom/)

Funded by the European Union

/authz_server/token	POST	Allows clients to obtain an access token using the Authorization Code grant type with Proof Key for Code Exchange (PKCE) for enhanced security.
/authz_server/check_token	POST	Validates the provided access token and retrieves details associated with the access token
/api/collections/orgAuthorizations/records	POST	Allows a user to request to join an organization.
/api/collections/services/records?filters=(organization='{orgId}')	GET	Lists services of a specified organization.
/api/collections/webauthnCredentials/records?filter=(user='{userId}')	GET	Lists WebAuthn sessions/devices for a user.
/api/collections/sessionDataWebauthn/records?filter=(user='{userId}')	POST	Initiates a new WebAuthn device registration.
/api/collections/users/records/:{userId}	GET	Retrieves the profile information of a user.
/api/webauthn/register/begin/{username}	GET	Initiates the registration of a new WebAuthn method (device).
/api/webauthn/register/finish/{username}	GET	Completes the registration of a new WebAuthn method (device).
/api/collections/users/records/:{userId}	POST	Updates the profile information of a user.
/dids/{DID}	GET	Read DID

Funded by the European Union

/api/v1/{domain}/pubkeys-update.chain	POST	Update DID
/api/v1/{domain}/pubkeys-deactivate.chain	POST	Deactivate DID
/api/v1/{domain}/pubkeys-broadcast-ganache.chain	POST	Broadcast to Ganache Blockchain
/api/v1/{domain}/pubkeys-broadcast-polygon.chain	POST	Broadcast to Polygon Blockchain

3.2.2 Creator Credentials

Description

Creator Credentials is a decentralised user-centric digital identity management framework specifically designed for the cultural and creative industries.

The project will develop a software application and a legal framework that can be used by media organisations to provide services to issue verifiable creator credentials. The app will be based on new and upcoming W3C and ISO standards for decentralised content identification (ISCC), decentralised identifiers (DIDs), verifiable credentials (VCs), and other established online reputation systems. It will be aligned with emerging European regulations on digital identity, such as eIDAS, as well as the directives on copyright (DSM), the Digital Services Act (DSA) and Digital Markets Act (DMA).

Interfaces

The provided specifications outline APIs for a system that manages credentials and user information, using OpenAPI 3.0.0 format for defining RESTful APIs.

These specifications define a comprehensive system for managing digital credentials and user information, with a focus on interoperability, security, and user-friendliness. The APIs cover a broad spectrum of functionalities, from basic user and credential management to more complex interactions involving credential verification and issuer-creator relationships.

- Credentials collection API

The Credentials Collection API focuses on managing credentials, specifically email credentials, for users.

Endpoint	HTTP method	Description

/credentials/email	GET	Allows creating an email credential for a user. It requires email and DID (Decentralized Identifier) in the request body
/credentials/{id}	GET	Retrieves a specific credential by its ID, requiring the credential ID and clerk ID in the query.
/credentials/email/{userId}	GET	Fetches email credentials for a user based on their user ID.
/credentials/all/{userId}	GET	Retrieves all types of credentials for a user, identified by their user ID.
/credentials/email/remove/{userId}	DELETE	Deletes an email credential for a user, specified by their user ID.

User Service API

This API is designed for managing user data.

Endpoint	HTTP method	Description
/users	POST	Creates a new user with a clerk ID and clerk role specified in the request body.
/users/{clerkId}	GET	Retrieves user details based on their clerk ID

- Collections API

This API is designed for managing user data.

Endpoint	HTTP method	Description

/test	GET	Returns a mock string for testing purposes.
/test/issuer/creators	GET	Get Issuer Creators
/test/issuer/creators/{creatorId}	GET	Get Credentials Request Details
/test/issuer/creators/accept	POST	Accept Creator Connection
/test/issuer/creators/reject	POST	Reject Creator Connection
/test/creator/issuers	GET	Get Creator Issuers
/test/creator/issuers/{issuerId}	GET	Get Issuer Details With Credentials
/test/creator/issuers/{issuerId}/confirm-request	POST	Confirm Creator to Issuer Connection
/test/issuer/profile	GET	Get Issuer Profile
/test/users/credentials	GET	Get Creator Credentials
/test/users/nonce	POST	Generate MetaMask Nonce
/test/users/{walletAddress}	POST	Generate MetaMask Nonce by Address
/test/verification/domain/txt-record	POST	Create Txt Record for Domain
/test/verification/domain/confirm	POST	Confirm Domain Txt Record
/test/verification/did-web/create-file	POST	Create DID Web JSON File

Funded by the European Union

/test/verification/did-web/confirm-upload	POST	Confirm DID Web JSON File Upload
/test/creator/credentials/issuers	GET	Get Issuers by Selected Credentials
/test/creator/credentials	GET	Get Requestable Credentials
/test/issuer/credentials	GET	Get Issuer Credentials
/test/creator/credentials/request	GET	Send Credentials Request

3.2.3 MUSAP

Description

The primary objective of the MUSAP project is three fold: firstly, to develop an open-source API library that streamlines the integration of various Secure Signature Creation Devices (SSCDs) into smartphone applications, thereby facilitating the creation of robust authentication solutions. Secondly, MUSAP aims to seamlessly integrate with both centralized and decentralized identity management systems, allowing SSCD keys to function effectively in both environments. This approach empowers end-users to access services without being constrained by the specific identity management model in use. Lastly, allow support for multiple certificates/credentials in one device. This approach allows end-users to have identities with various level of assurances in use.

Interfaces

- General interfaces

General API has functions that interact directly with MUSAP, and not with any external modules.

Function	Return value	Description
init(arg1)	void	Initialize MUSAP library for use. End-User Application must call this upon startup.
EnableSscd(arg1)	void	Enable a SSCD for End-User Application. Enabled means MUSAP can do either key

		binding or key generation on the SSCD.
setDebugLog(arg1)	void	Enable or disable debug logging of MUSAP.

- Key discovery interfaces

Key Discovery API has functions that relate to SSCD and Key Discovery.

Function	Return value	Description
listKeys	List<MusapKey>	List all generated or bound keys on this device.
listKeys(arg1)	List<MusapKey>	List all generated or bound keys that match request criteria on this device.
listActiveSSCDs	List<MusapSscd>	Lists all active SSCDs on this device. Active means MUSAP has done either key binding or key generation on the SSCD.
istActiveSscds(arg1)	List<MusapSscd>	Lists all active SSCDs that match the request criteria on this device. Active means MUSAP has done either key binding or key generation on the SSCD.
listEnabledSscds	List<MusapSscd>	Lists all enabled SSCDs on this device. Enabled means MUSAP can do either key binding or key generation on the SSCD.
istEnabledSscds(arg1)	List<MusapSscd>	Lists all enabled SSCDs that match the request criteria. on this device. Enabled means MUSAP can do either key

Funded by the European Union

		binding or key generation on the SSCD.
getKeyByUri(arg1)	MusapKey	Get a single generated or bound key by its KeyURI

- Key search request

Key search request is defined in the table below.

KeySearchReq		
Parameter	Description	Example Value
SSCD Type	Type of the SSCD	Local Keystore
SSCD Provider	Who provides the SSCD. For example "AKS" for Android Keystore.	AKS
SSCD Country	ISO 3166-1 country code (alpha-2 format) of the SSCD	FI
Key Algorithm	Desired key algorithm.	KeyAlgorithm.ECC_P256_K1
Signature Algorithm	Signature algorithm that the key should support.	SignatureAlgorithm.SHA256_WITH_ECDSA
LoA	Desired LoA of the key.	MusapLoA.EIDAS_SUBSTANTIAL
Alias	Exact key alias	TestKey

Funded by the European Union

DID	Which DID the key has been generated for.	did:method:value
KeyURI	Partial or full KeyURI	mss:alias=TestKey
SSCD Attribute	Arbitrary attribute of the SSCD. This is SSCD specific.	managementkey, FFFFFFFF
Key Attribute	Arbitrary attribute of the Key. This is key specific.	keyslot, 1
Role	Role the keys have been generated for	personal

- SSCD search request

The SSCD search request can be used to search for an activated SSCD.

KeySearchReq		
Parameter	Description	Example Value
SSCD Type	Type of the SSCD	Local Keystore
SSCD Provider	Who provides the SSCD. For example "AKS" for Android Keystore.	AKS
SSCD Country	ISO 3166-1 country code (alpha-2 format) of the SSCD	FI

Funded by the European Union

Key Algorithm	Desired key algorithm.	KeyAlgorithm.ECC_P256_K1
Signature Algorithm	Signature algorithm that the key should support.	SignatureAlgorithm.SHA256_WITH_ECDSA
LoA	Desired LoA of the key.	MusapLoA.EIDAS_SUBSTANTIAL

- Signature API

Signature API contains the function to produce signatures with any MUSAP SSCD.

Function	Return value	Description
sign(arg1, arg2)	void	Requests MUSAP to produce a signature with a certain SSCD. This function is asynchronous. MUSAP will notify the callback parameter when complete.

- Signature request

The signature request contains mainly the key, data and signature algorithm to use.

KeySearchReq		
Parameter	Description	Example Value
Key	MUSAP key to sign with	N/A
Data	Data to sign as a byte array. May contain a list of data to sign. Each	FFFF

	list element will be signed.	
Signature Algorithm	Signature algorithm to use. Make sure that this matches the supported algorithms of the key.	SignatureAlgorithm.SHA256_WITH_ECDSA
Signature Format	Format of the signature. If not specified, default is RAW.	SignatureFormat.CMS

- Key Lifecycle API

Key Lifecycle API has functions that relate to key generation and key removal.

Function	Return value	Description
generateKey(arg1, arg2, arg3)	void	Request MUSAP to generate a key on a certain SSCD. This function is asynchronous. MUSAP will notify the callback parameter when complete.
removeKey(arg1)	boolean	Remove a previously generated key from MUSAP. Return value indicates a successful removal.
updateKey(arg1)	boolean	Update a bound or generated key metadata. Return value indicates a successful update.
removeSscd(arg1)	boolean	Remove a previously enabled SSCD. Return value indicates a successful removal.

- Key generation request

The key generation request is used when requesting key generation

KeyGenerationReq		
Parameter	Description	Example Value
Key Algorithm	Desired key algorithm.	KeyAlgorithm.ECC_P256_K1
Alias	Exact key alias.	TestKey
DID	Specify the DID to generate the key for.	did:method:value
Key Attribute	Arbitrary attribute of the Key. This is key specific.	keyslot, 1
Role	Role to generate the key for	Personal
Step-up policy	Step-up policy that states when/if step-up authentication is used when accessing the key.	{ "frequency": "ALWAYS", "validity": "10 min", "type": "biometric" }

- Key update request

The key generation request is used when requesting key generation

KeyUpdateReq		
Parameter	Description	Example Value
Alias	Exact key alias	TestKey

Key	Which key to update.	
DID	Specify the DID to generate the key for.	did:method:value
Key Attribute	Arbitrary attribute of the Key. This is key specific.	keyslot, 1
Role	Role to generate the key for	Personal
State	Update key state to one of: ACTIVE, INACTIVE (key is permanently inactive), SUSPENDED (key cannot be used until restated), REVOKED (key certificate is revoked).	INACTIVE

- Key Binding API

Key Lifecycle API has functions that relate to key generation and key removal.

Function	Return value	Description
bindKey(arg1, arg2, arg3)	void	Request MUSAP to bind a key on a certain SSCD. This function is asynchronous. MUSAP will notify the callback parameter when complete.

- Key Binding request

The key binding request is used when binding MUSAP to an existing key on an SSCD.

KeyUpdateReq		
Parameter	Description	Example Value
Alias	Exact key alias	TestKey
DID	Specify the DID to generate the key for.	did:method:value
Key Attribute	Arbitrary attribute of the Key. This is key specific.	keyslot, 1
Role	Role to generate the key for	Personal
Step-up policy	Step-up policy that states when/if step-up authentication is used when accessing the key.	{ "frequency": "ALWAYS", "validity": "10 min", "type": "biometric" }

- Import/Export API

Import/export API allows the End-User Application to transfer metadata to another End-User Application.

Function	Return value	Description
importData(arg1)	void	Import data from another MUSAP app to this End-User Application. Example use case is user changing their device.

Funded by the European Union

exportData	String	Export key and SSCD metadata from this End-User Application for use in another app. Example use case is user changing their device.

- SSCD interface API

The SSCD interface allows End-User Application to add any SSCD for use with MUSAP. End-User Application should not call these methods directly, but instead use other MUSAP APIs which can call the new SSCD.

Function	Return value	Description
bindKey(arg1)	MusapKey	Bind a key on SSCD with MUSAP.
generateKey(arg1)	MusapKey	Generate a new key on the SSCD.
sign(arg1)	MusapSignature	Produce a new signature on the SSCD.
getSscdInfo	MusapSscd	Get info of this SSCD such as supported signing algorithms, SSCD type and name, etc.
isKeyGenSupported	boolean	Returns true if the SSCD can generate new keys and false otherwise.
getSettings	SscdSettings	Return a map of settings this SSCD can use.

- Coupling API

This call API is provided by the MUSAP library to the Android/iOS app.

Function	Return value	Description
enableLink(arg1)	MusapLink object that	Enable the MUSAP link connection. Sets the URL to

	contains link properties	MUSAP link and an ID to use during linking.
getLink()	MusapLink object or null	Get the details of the enabled MUSAP Link.
getLinkId()	MUSAP Link ID	Get a Link ID used to bind a web application
disableLink()	void	Disable the MUSAP link
isLinkEnabled()	boolean	Check if MUSAP link is enabled
pollLink()	Signature Request or null	Poll the MUSAP link for a signature request. This should be called periodically after receiving a notification.
coupleWithRelyingParty(arg1, arg2)	void	Couple this instance of a MUSAP app with a relying party.
updateFcmToken(arg1, arg2)	void	Update the MUSAP Link database with this apps latest identifier used for push notifications.
listRelyingParties()	List<RelyingParty >	List all Relying Parties this MUSAP app has been coupled with.
removeRelyingParty(arg1)	boolean	Remove a previously coupled Relying Party. Return value indicates success.
sendSignatureCallback(arg1, arg2)	void	Send a successful signature to MUSAP Link so it knows that the signature is successful.
sendKeygenCallback(arg1, arg2)	void	Send a successful key generation response to MUSAP Link so it knows that

Funded by the European Union

		the key generation is successful.

- REST API

API can be called by an application to request binding to an end-user application that uses the MUSAP library. This API allows the application to request signatures from the user.

Endpoint	HTTP method	Description
/link	POST	Start the linking process between a web application and an end-user application using MUSAP
/sign	POST	Request a signature from MUSAP. The operation has to define which key to use and which MUSAP app instance to communicate with.
/generatekey	POST	Request MUSAP to generate a new key. The operation can define required key features like algorithm.
/docsign		Request a signature from MUSAP. The operation can define multiple DTBS. MUSAP will sign the DTBS related to the key chosen by the user.
/updatekey	POST	Request updating key details within MUSAP Link. Currently only used to update the key name.

- MUSAP coupling API

MUSAP Coupling API is an HTTP API between the MUSAP library and the MUSAP link server. Coupling API is used when a server-side wallet tries to access a mobile SSCD on user's smartphone. This API is optional and only used when MUSAP link is enabled.

Endpoint	HTTP method	Description
/musap	POST	Send a request to the MUSAP Link

The MUSAP Coupling API has one end-point that supports the following message types:

Type	Request Payload	Response Payload	Description
"getdata"	N/A	(JSON) type, payload,musapid	Check for pending requests in MUSAP Link.
"signcallback"	(JSON)	N/A	Response for the signature
"signcallback"	(JSON) linkid, publickey, signature	N/A	Response for the key generation request
"enrolldata"	(JSON) fcmtoken, apnstoken, transid, tokendata	(JSON) status	Enroll MUSAP library to MUSAP Link
"updatedata"	(JSON) fcmtoken, apnstoken	(JSON) status	Update FCM/APNs token in MUSAP Link
"linkaccount"	(JSON) couplingcode, musapid	(JSON) status, linkid	Complete linking MUSAP to a RP service
"externalsignature"	(JSON) clientid, data, display, keyid,	(JSON) status, signature,	Request a signature from an SSCD integrated with MUSAP Link

Funded by the European Union

	timeout, transid, attributes	publickey, attributes

The MUSAP coupling API supports the following messages as a response to the GetData operation:

Type	Request Payload	Description

"sign"	(JSON)
	data, display, mode, format, linkid, attributes,
	key {
	keyid, keyname, keyusage, algorithm, publickeyhash
	}
		Request a signature or key generation from MUSAP

3.2.4 TREVO

Description

Voting systems have evolved during the last hundreds of years to become more sophisticated and complex, starting from paper-based ballots up to electronic voting machines and internet voting which have been introduced as new voting technologies. However, electronic-based methods have raised concerns about security and the potential for tampering results, manipulation or hacking. The TREVO project aims to revolutionize electronic voting systems by employing decentralized identities rooted on blockchain and an SSI approach that puts the user at the center of the process from the early phases of the design phase.

The main objective of TREVO is to tackle main challenges in electronic voting that are still open, such as voter anonymity, ballot privacy, trusted tally/audit as well as verifiability. It employs blockchain technology and more specifically Decentralised Identities, Verifiable Credentials and state-of-the-art communication protocols and architectures, following the latest EU guidelines and regulations in terms of digital identities and data protection. The framework incorporates a mobile wallet that enables EU-wide interoperability for citizen authentication and authorization based on well-established technologies entailing trust from anchors of the public sector.

Interfaces

Endpoint	HTTP method	Description
/eleader/create	POST	Create Election Leader
/eleader/update	PUT	Update Election Leader
/eleader/delete	DELETE	Delete Election Leader
/eleader/get/{username}	GET	Get Election Leader by username
/eleader/get/all	GET	Get Election Leaders
/voter/create	POST	Create voter
/voter/update	PUT	Update voter
/voter/add/poll	PUT	Add polls to voter
/voter/delete	DELETE	Delete voter
/voter/get	GET	Get voter
/voter/get/all	GET	Get voter
/group/create	POST	Create voters group
/group/update	PUT	Update voters group
/group/delete	DELETE	Delete voters group
/group/get	GET	Get voters group
/group/get/all	GET	Get voters group
/poll/template	GET	Get Poll Template
/poll/template	PUT	Update Poll Template
/poll/template	POST	Create Poll Template

/poll/template	DELETE	Delete Poll Template
/poll/template/all	GET	Get Poll Templates
/poll/template/draft	GET	Get Draft Templates
/keycloak/jwt	GET	Get JWT
/keycloak/logout	GET	Logout
/oauth/callback	POST	Auth Callback
/query	POST	Submit a query to the analytics engine.
/data-overview	GET	Retrieve an overview of the available data for analysis
/query-history	GET	Retrieve a history of submitted queries
/query/{query-id}	GET	Retrieve the results of a specific query by its ID

3.2.5 ORCHESTRAL

Description

The project aims to co-develop an identity management system for marginalised and internet activist communities built by mature communities that work with Pangea's digital service and circular device management services. The system will allow users to manage their online identities and access community-centred internet services trusted high quality data according to their identity profile. The system development uses and will be open-source software. The system will be evaluated and disseminated to other communities. The system will be designed to be trustworthy and to preserve personal privacy. It will be aligned with decentralised identity models, including considering EIDAS and build on existing and emerging digital identity technology solutions, but adapted to the target and other similar communities of practice.

Interfaces

The project does not provide any SDK or APIs for third parties. This GitHub repository has a software snapshot of the actual developed code:

https://github.com/NGI-TRUSTCHAIN/Orchestral

Funded by the European Union

We describe the software in terms of the project foreground: Idhub, including the IdP and verifier portal-related modules, the VC backend sskit_trustchain for verifiable credentials, credential schemas developed for the social and solidarity economy pilots, the dockerised software packaging for deployment and demos, and the third-party libraries used.

3.2.6 THE SOCIAL WALLET

Description

The project focuses on crafting digital solutions to enhance the efficient and inclusive delivery of benefits to marginalized groups. The existing platform in Weert, based on older blockchain technology, is set for an upgrade with the new "Social Wallet", utilizing Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs). Key features include a Management Platform for digital identities, APIs for secure transactions, mobile applications for consumers/providers, and inclusive functions like paper-based wallet scanning.

Emphasis is on robust privacy standards, environmental sustainability, and a dual open-source and SaaS offering. Challenges arise from transitioning technologies, ensuring user privacy, and scalability. Addressing these will be pivotal for the project's widespread success and global adaptability.

Interfaces

- Agent SD-JWT

This API is responsible for issuing and verifying SD-JWT Verifiable Credentials. It is being used in the Agent module.

Function	Return value	Description
createVerifiableCredential	(string) a JWT conforming to SD-JWT	This is an external API to create verifiable Credentials in multiple formats. In this project SDJWT is being added as a proof format, resulting in a JWT string conforming to the SD-JWT format.
verifyCredential	(IVerifyResult) A object containing the verified Boolean	Returns whether the VC was verified successfully or not

Funded by the European Union

	indicating success and an optional error object, denoting any errors	

Statuslist2021 API

This API is responsible for checking the status of a Verifiable Credential containing a StatusList 2021 property against the status list maintained by the issuer. This API is being used in the Agent module.

Function	Return value	Description
createVerifiableCredential	(string) a JWT conforming to SD-JWT	This is an external API to create verifiable Credentials in multiple formats. In this project SDJWT is being added as a proof format, resulting in a JWT string conforming to the SD-JWT format.

- Authentication, authorization endpoint configuration API

This API allows for uniform configuration of both authentication and authorization on endpoints of REST APIs exposed by the Agent. These are so called middleware functions to be used in Express (JavaScript API/endpoint solution). This allows the endpoints to be configured either at a global scope per API or even per endpoint, allowing to integrate OAuth2, OpenID Connect, static bearer tokens for authentication, but also for instance role checking (RBAC/ABAC) and Casbin integration for authorization.

Function	Return value	Description
checkAuthentication	Returns the response object or invokes the next function in case of both success and errors	This is middleware function to check authentication. It is to be used on an individual REST endpoint and allows the user to configure what authentication should be applied

Funded by the European Union

checkAuthorization	Returns the response object or invokes the next function in case of both success and errors	This is middleware function to check authorizations. It is to be used on an individual REST endpoint and allows the user to configure which and how authorization should be applied

3.2.7 DID4EU

Description

DID4EU will offer developers and organizations a holistic open source decentralized identity infrastructure that makes it easy to build applications using all major identity technologies (e.g. SSI based on W3C and OIDF standards, mdocs/mDL based on ISO/IEC 18013-5:2021), NFTs, SBTs) in a way that is ecosystem- and blockchain-agnostic and compliant with EU's existing and emerging regulation on digital identity like eIDAS2 or GDPR.

Interfaces

- Issuer API

Endpoint	HTTP method	Description
/raw/jwt/sign	POST	Signs W3C credentials as JWT. No exchange mechanism used.
/openid4vc/jwt/issue	POST	Signes W3C credential as JWT and prepares an OID4VC offer URL for holders to receive the credential in their wallet.
/openid4vc/jwt/issueBatch	POST	Signes W3C credentials as JWTs and prepares an OID4VC offer URL for holders to receive the credentials in their wallet.
openid4vc/sdjwt/issue	POST	Signes W3C credentials as SD-JWT and prepares an OID4VC offer URL or holders to receive the credential in their wallet.
openid4vc/mdoc/issue	POST	This endpoint issues a mdoc and returns an issuance URL

- Verifier API

Endpoint	HTTP method	Description
/openid4vc/verify	POST	Initializes an OID presentation session, with the given presentation definition and parameters. The URL returned can be rendered as QR code for the holder wallet to scan, or called directly on the holder if the wallet base URL is given.
openid4vc/session/{id}	GET	Get info about OIDC presentation session that was previously initialised
openid4vc/policy-list	GET	List registered policies
/openid4vc/verify/{state}	POST	Verify vp_token response for a verification request identified by the state
/openid4vc/pd/{id}	GET	Get presentation definition object by ID

3.2.8 IM4DEC

Description

UN convention Article 9 requires countries to take measures for the full and equal participation of persons with disabilities, including access to communication and information services. Despite this, there are still about 1 million deaf and hard of hearing persons in Europe who currently rely on outdated technology (e.g. fax) and help from others to make an emergency call.

IM4DEC is addressing the following:

- Presenting a verified identity when delivering an emergency chat: replace current SMS verification with an eIDAS 2.0 compliant identity based on DIDs,
- Deliver relevant (pre-recorded) information in case of an emergency: extend current free-text stored on the smartphone with structured information stored securely online,

- Operators struggle with chat from deaf persons: introduce an AI-based chatbot to train users and share this information with emergency organisations as basis for new training material,
- Such data (identity, emergency information, training chats) are considered special category data under the GDPR and we will perform a formal DPIA (Data Protection Impact Assessment) for the end-to-end dataflow.

This is not only for the benefit of deaf people but also individuals oppressed by domestic violence can make use of this technology through the use of a silent emergency notification; already in operation since 2022 in Austria we will provide an SDK to include this functionality in an EU Digital Identity Wallet to get such functionality on every phone.

Interfaces

- oydid-js

Function	Return value	Description
create	(str) did	create a new DID and store underlying DID Document and log data
read	(json) did document	resolve DID to DID Document
update	(str) did - new DID	update DID Document for a given DID
deactivate	(str) did	deactivate DID through publishing revocation information
encrypt	(str) cipher (str) nonce	encrypt payload using public key from a DID provided in options
decrypt	(str) decrypted payload	decrypt cipher message using private key from a DID provided in options

Funded by the European Union

sign	(str) signature of payload	sign a payload for a given DID using the private key provided in options
verify	(bool) signature verification	verify signature for a given DID using the public key provided in options

- ng112-js

Function	Return value	Description
agent = new Agent	(object) connection to the SIP proxy	establish a connection to the originating ESRP
agent.initialize	initialisedagent	initialises the agent (sends initial SIP REGISTER)
agent.updateVCard	void	set the agent's vcard can be called anytime and is reflected in the conversation
agent.updateLocation	void	set the agent's location can be called anytime and is reflected in the conversation
conversation = agent.createConversation	conversation object	start a new emergency conversation
conversation.addMessageListener	void	register a callback for both incoming and outgoing messages
conversation.start	message object (with Promise to ensure successful forwarding)	initiate the emergency conversation and send the initial START message to the ESRP

Funded by the European Union

conversation.sendMessage	message object (with Promise to ensure successful forwarding)	send subsequent messages
conversation.stop	message object (with Promise to ensure successful forwarding)	stop the emergency conversation and send STOP message to the ESRP
agent.dispose	Promise	dispose the agent

- Registration API

Endpoint	HTTP method	Description
/api/v3/register	POST	create an initial request to start the onboarding of a new user
api/v3/register	PUT	request in the onboarding process of a new user
api/v3/register/:reg_id	GET	returns the current status for a given registration id
oydid/init	POST	initiates the did_auth sequence for a given DID
oydid/token	POST	generates a OAuth2 bearer token upon successful verification of the challenge
/version	GET	provide current version of the component

- Chatbot API

Endpoint	HTTP method	Description
api/v1/chatbot/welcome	POST	return the pre-configured welcome message

Funded by the European Union

api/v1/chatbot/reply	POST	return answer from ChatGPT based on previous messages with <call_id>
api/v1/chat/list?page=X&items=X	GET	return a paged list of all conversations
api/v1/chat/<call_id>	GET	return available attributes for given conversation

- DID Lint Service API

Endpoint	HTTP method	Description
resolve/<did>	GET	resolve DID using internal resolver functions or fall back to uniresolver
validate/<did>	GET	resolves a given DID and validates it against the SOyA DID structure; list any errors and show suggestions in "infos
/validate	POST	validate input against the SOyA DID structure; list any errors and show suggestions in "infos"

3.2.9 WIDE

Description

The project proposal focuses on developing a Decentralized Identity (DID) bridge prototype for managing user identities, and connecting the European Commission's eIDAS 2.0 initiative with decentralized autonomous organizations (DAOs) on public-permissionless distributed ledger technologies (DLT). This use-case agnostic solution aims to enhance credential access for Web3-native organizations and protect individuals' data privacy rights.

The solution, WIDE, aims to combine existing technologies from traditional finance and the cryptocurrency sector with innovative DID concepts. It features a novel architecture that preserves privacy and user control, while freeing users from the responsibility of managing their data directly. Our DID bridging client relies on existing wallet solutions to empower DAOs to access user data without the need for custom integrations with individual identity solutions.

Interfaces

- WIDE Backend APIs

Funded by the European Union

Endpoint	HTTP method	Description
siwe/generate_signin	GET	Generates the sign in SIWE message for the user so that they may log into the WIDE client. If the user has a valid session already they are immediately logged in.
siwe/generate_signup	GET	Generates the Terms of Service SIWE message, and stores this to persistent storage to indicate that the user accepted the Terms of Service
siwe/verify_signin	POST	Recovers the address from the SIWE message, and verifies that it matches that of the requesting user. If so, the session is stored in an HTTPS Cookie which gives the user access to the storage endpoints.
siwe/tos	DELETE	This is a temporary endpoint used for the development phase and will be deprecated. It is used to invalidate a user's Terms of Service signature.
storage/user/{address}/issuedcredentials	GET	Sends the list of stored credentials (without details) to the user. This will contain a key that allows the user to retrieve the encrypted individual credentials
storage/user/{address}/credentials/{key}	GET	Sends a list of encrypted credentials for a particular user/issuer pair

Funded by the European Union

storage/user/{address}/credentials	POST	String confirming success or failure.

- Dapp WIDE Library

Function	Return value	Description
Verify	bool) indicates whether the authentication checks in the smart contract are valid or not	Calls the smart contract to verify that the credential was recorde by the user on the blockchain at a previous date

3.2.10 CLIENT-DIDS

Description

CLIENT-DIDS improves the Universal Registrar tool, which is a well-known open-source project at the Decentralized Identity Foundation (DIF). Parallel to the Universal Resolver (which allows resolution of DIDs), the Universal Registrar allows creation of DIDs across different DID methods and networks. It offers an abstraction layer with a universal interface, which means that clients of this tool can create DIDs without having to know or implement details of the underlying DID method (which may involve blockchains, web servers, or any other technology). This tool can be self-hosted, it should not be operated by a single centralized authority.

Interfaces

- WIDE Backend APIs

Endpoint	HTTP method	Description
/siwe/generate_signin	GET	This function creates a new DID and associated DID document, according to a known DID method, using various options and optionally an initial DID document.
/update	POST	This function updates the DID document associated with the DID,

Funded by the European Union

		either by completely replacing it, or by performing an incremental update.
/methods	POST	Returns a list of supported DID methods.
/properties	GET	Returns a map of properties of the registrar.

- Universal registrar drivers

Endpoint	HTTP method	Description
/create	POST	-
/update	POST	-
/deactivate	POST	-

3.2.11 EVI

Description

Drivers of electric vehicles (EVs) face significant data privacy risks when charging their vehicles in Public Charging Stations. Each charge point operator (CPO) uses different software to manage its stations and collect charging fees. Drivers are forced to sign up with multiple applications to start a charging session in Public Charging Stations. This further complicates drivers' experience as each application requires personal and financial data before it enables the driver to initiate a charging session. An underappreciated risk with the dispersion of information across multiple platforms is that vehicle and user data can be used to pinpoint users' locations and everyday activities. Drivers do not retain control on how 3rd parties exploit their personal data. For example, CPOs can use data related to users' daily location, vehicle type and frequency of charging sessions for targeted advertising or provide these data to 3rd party advertisers that seek to target specific user groups. Most drivers do not fully understand the potential uses of their private data whenever they sign up for an EV charging application.

Interfaces

- Module 1

Funded by the European Union

Endpoint	HTTP method	Description
/create	POST	This POST function installs a new contract certificate to the vehicle that is connected to a charging station compatible to EVI. The endpoint is accessible to third party EV charging applications registered with EVI

- Module 2

Endpoint	HTTP method	Description
https://evi.evloader.com/v1/verify	POST	This post function runs when a vehicle with enabled plug and charge function is connected to a charging station compatible with EVI and plug and charge. 3rd party that is managing the charging stations relays information related to the vehicle's contract certificate via REST API to EVI. The function receives the contract certificate from the vehicle and then verifies it via the appropriate issuer. The verification will fail when no contract with this public key is found. If contract is found but no available balance or working payment method is associated to the contract "insufficient balance" is returned.

3.2.12 IS-CIS

Description

We propose a generic framework that mimics human nature in disclosure of identity and has a myriad of different social and business applications. It can allow the disclosure of sensitive medical data for the purposes of recruiting a cohort of a medical trial or guide the disclosure of personal data in a social setting. It could become a de

Funded by the European Union

facto standard for identity disclosure from human to IT and enable complex multi-person chains of disclosure.

It reserves control and repeal rights in the hands of the individual. It allows discoverability. It places an onus on the asker to justify and convince the askee. It retains a permanent record of who requested, and who granted, what and when. Our proposed framework does not replace validation– it does not verify the data in the system with external sources of truth – as such it is synergic with all other solutions that do provide that validation. Its purpose is to hand a safe, verifiable control to the owner of the data.

Interfaces

- DID Wallet SVC (DID_SVC)

Function	Return value	Description
constructor(mnemonic)	void	Initialises Account Manager class
createAccount(identifier)	(string) JSON object with account details.	Creates a new account based on the mnemonic.
addAccount(identifier, account)	(string) JSON object with account details	
getAccount(identifier)	(string) JSON object with accounts identifiers.	Extracts the keys of #accounts and returns them in JSON format.
getAccountDetails(identifier)	(string) JSON object with the account details.	Extracts the account that matches identifier on accounts and returns it in JSON

- Consent token smart contract SDK

Function	Return value	Description
startProcess(DH_DID,DO_DID,data_tuple)	int) process identifier	Initialize the request data management

Funded by the European Union

approveRequest(process_identifier, DO_DID)	(bool) The result informs about the accept/reject of the request	Method to allow DO accept/reject the request data management.
deployNewCTInstnace(process_identifier)	(int) process_identifier Identifier of specific request data management	Deploy a specific Consent Token Smart Contract for a specific and approved request.
createConsentToken(process_identifier, data_tuple)	(str) Id Token / DID Id Token	Create (mint) a new Token Consent for a specific request
claimConsentToken(idToken, idActor)	(bool) The result informs about the success or failure. Id transaction	Method to transfer the Consent Token to another actor
burnConsentToken(idToken)	(bool) The result informs about the success or failure. Id transaction	Method to destroy (burn) the Consent Token when the process is completed

- Request data management

Endpoint	HTTP method	Description
Data_management/<data_tuple>/<DH_DID>/<DO_DID>/<conditions>	POST	It initialises the transaction that services as starting point for data management
Data_management/<data_tuple>/<DH_DID>/<DO_DID>/<conditions>	PATCH	It finalises the transaction that services as starting point for data management with the confirmation or rejection of the DO

Data_management/<data_tuple>/<DH_DID>/<DO_DID>/<conditions>	GET	It offers the status of the request at the moment of the request (opened, confirmed, rejected, expired?

- Request data set based on parameters

Endpoint	HTTP method	Description
Data_set/<DH_DID>/<DS_DID>/<conditions>	POST	This returns a comprehensive list of the data sets that match Data Seeker search parameters

- Process access to specific data tuple

Endpoint	HTTP method	Description
Data_access/<DO_DID>/<DS_DID>/<conditions>	POST	This resource initiates the minting of the Consent Management Token grant that enables DS to use DO details held by DH.

- Data tuple consumption process

Endpoint	HTTP method	Description
Data_use/<DH_DID>/<DS_DID>/<conditions>	POST	This resource initiates the burning of the

Funded by the European Union

		Consent Management Token grant that enables DS to use DO details held by DH

3.2.13 PRIVÉ

Description

PRIVÈ extends the decentralized user-centric identity management framework by building an open-source library that can be added as an extension to any SSI wallet on the Holder side to enable the use of hardware-based keys. This offers the possibility to bind Verifiable Credentials (VCs) to the wallet of the holder and transfer the root of trust of the SSI ecosystem purely to the digital wallet by considering an underlying Trusted Component as part of the wallet, without making any assumptions on the trustworthiness of the other layers. This enables digital identity wallets to align with emerging regulations and standards like eIDAS that require higher level of assurances for services. At the same time, we make sure that privacy-enhancing properties like selective-disclosure are fully supported, to make the wallet compliant with privacy regulations like GDPR. To this end, PRIVÈ utilizes a privacy-preserving cryptographic protocol, namely Direct Anonymous Attestation (DAA) to provide verifiable evidence and assurances about the presented VC's origin and integrity. We can now enforce that a VC can only be issued by an attested Issuer and that this VC is bound to the Holder's device (wallet), overcoming the current limitations of bare proof-of-possession of a sw-based key.

Interfaces

- Wallet & DAA Bridge Communication API

Function	Return value	Description
pairing	Success/Fail	Variant 1: The pairing function is a part of the DAA_Bridge library and is invoked by the Wallet to initiate the pairing process. Pairing is a crucial security step that establishes trust between the Wallet and the Software-Based TPM (Trusted Platform Module).

		Variant 2: The Pairing Process in Variant 2 is obsolete, so this function does not exist at all.
register	Success/Fail	Variant 1: Register function is responsible for generating proof of the successful creation of a DAA Key, a critical component of the registration process. Variant 2: This call is obsolete because the Registration Process is being triggered by default in the app launch.
createVP	signed_VP	Variant 1 & 2: In the Verification Process, the Wallet creates a verifiable presentation with all the attributes that need to be disclosed, after it "gathers" them from its VCs. The final signed VP will come from DAA Core after it passes through DAA Bridge to get all the attribute keys for every attribute inside the VC.
getSignature	DAASignatur e	Variant 1: In the Issuance Process, the Wallet calls the GetSignature function in DAA Bridge in order to receive Core's DAA Signature. It passes a concatenated hash of the DAA_Issuer_Nonce value and the public part of the Wallet's key. Variant 2: In the Issuance Process, the Wallet calls the GetSignature function in DAA Bridge in order to receive Core's DAA Signature. It passes a DAA Nonce received from the DAA Issuer through the VC Issuer and user's fingerprint hash.
verifyVC	Success/Fail	Variant 1 & 2: VerifyVC is a function exposed by the DAA Bridge to be invoked by the Wallet each time it receives freshly issued verifiable

		credentials from a PRIVÉ enabled VC Issuer. The verification process is feasible through cryptology, leveraging the cryptographic capabilities of pairings.
requestTPMNonce	TPMNonce	Variant 1: RequestTPMNonce is the function that gets called from the Wallet to the DAA Bridge in order to eventually get a TPM Nonce from the DAA Core. Variant 2: In this variant the function is obsolete and does not exist at all
initializeTPM	void	Variant 1: InitializeTPM is the function that creates a fresh session with the underlying TPM, preparing the non-volatile memory and the Platform Configuration Registers of the TPM for a new set of security parameters. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete
InitiateTPM	publicBuffer	Variant 1: InitiateTPM is the function that gets called right after InitializeTPM after launching the wallet app.In this invocation the user creates his attestation key, binded with the wallet's key and loads to the PCRs the user's fingerprint. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
CorePairing	Success/Fail	Variant 2: This function is a part of the DAA Core and is invoked by DAA_Bridge Library for the pairing process.CorePairing is dedicated for

Funded by the European Union

		storing the hashed fingerprint to the Platform Configuration Registers (PCRs) of the underlying TPM. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete
CoreRequestSignVP	tpm_nonce	Variant 1: In the Verification Process, the Wallet has to initially prove to the DAA Core that it has not been compromised in order to request a DAA Key signature for the created VP. For that purpose, DAA Core's CoreRequestSignVP is called from DAA Bridge continuing RequestSignVP, in order to contact the DAA Core. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
CoreSignVP	attributes_block	Variant 1: CoreSignVP is vital for the Verification Process since it passes the attributes tobe-disclosed as Bytecode for the DAA Core to sign them. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
HandleChllenge	CertBuffer, DAAsig_commit	Variant 1: HandleChallenge is a function exposed by the DAA Core lib to be invoked by the DAA Bridge. The DAA bridge invokes HandleChallenge on the first response of the DAA Issuer, in order for the device to complete the first phase of secure enrollment. More specifically, HandleChallenge is responsible for decrypting the certificate, created by the DAA Issuer,

		wrapped with the Endorsement and the DAA key of this particular TPM and creating a DAA signature for the DAA Issuer to verify. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
CoreTPMNonce	TPMNonce	Variant 1: CoreTPMNonce is a function of the DAA Core and is invoked by the DAA Bridge in order to get a TPM Nonce. This TPM nonce will be signed by the wallet's key to prevent replay attacks and contribute to the key restriction usage policy enforcement rule. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
GetDAA_Credential	DAAcre	Variant 1: GetDAACredential is a function exposed by the DAA Core to be invoked by the DAA Bridge, so that the device can complete the secure enrollment. More specifically, the DAA Core takes as input the a symmetric encryption key, wrapped with the TPM's Endorsement key and DAA key, and the encrypted DAA credential.The DAA Core unwraps the symmetric encryption key, and decrypts the issued DAA credential in order to be stored in the DAA Bridge registry for the creation of DAA signatures. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.

Funded by the European Union

CreateDAA_Signature	DAAsignature	Variant 1: CreateDAAsignature is a function exposed by the DAA Core to be invoked by the DAA Bridge in order to provide attestation evidence to the service provider. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
VerifyVCs	VerificationStatus	Variant 1 & 2: VerifyVC is a function exposed by the DAA Core to be invoked by the DAA Bridge each time the wallet receives freshly issued verifiable credentials from a PRIVÉ enabled VC Issuer. The verification process is feasible through cryptology, leveraging the cryptographic capabilities of pairings.
RequestDAA_VP	VP	Variant 1: RequestDAA_VP is a function exposed by the DAA Core to be invoked by the DAA Bridge in order for the DAA Core to create a verifiable presentation for a specific set of attributes that the wallet has already acquired and verified their corresponding verifiable credential. Variant 2: In the second variant this process is an internal process of the DAA Core therefore the function is obsolete.
vc_randomize_credential	Randomised VerifiableCredential	RandomiseVC is a function exposed by the DAA Core to be invoked by the DAA Core each time the device requests the creation of a verifiable presentation (VP).More specifically, RandomiseVC takes as input the issued verifiable credential and randomises them both in the trusted and the untrusted world

Funded by the European Union

		in order to create anonymous signatures.
DAAcredential	DAAcredential	Randomise Credential is a function exposed by the DAA Core to be invoked by the DAA Core each time the device requests the creation of a DAA signature. More specifically, RandomiseCredential takes as input the issued DAA credential and randomises them both in the trusted and the untrusted world in order to create anonymous signatures.
SatisfySignPolicy	Session	SatisfySignPolicy is a function of the DAA Core and is invoked by the DAA Core each time the DAA key is needed to sign a message.As aforementioned, the DAA key is binded with a key restriction usage policy, for that reason the DAA Core has to satisfy the key restriction usage policy through a specific set of a policy verification chain.

- Wallet & DAA Issuer and VC Issuer Communication API

Endpoint	HTTP method	Description
/authenticate	POST	This feature offers an eIDAS authentication REST API capable of authenticating both the PRIVÉ User and wallet owner through a genuine eIDAS node. Utilising this API, we obtain authorization to access verified credentials after successful authentication.
/registerWallet	POST	This function lives inside the VC Issuer and is called by the Wallet after the user presses the "Register" button in order to initiate the registration process. It will eventually store the user's fingerprint by

Funded by the European Union

		calling DAA Issuer's StoreFingerprint and it will return to the Wallet a proofOfDAAKey String demanding a proof of DAA Key creation from the Wallet in order for the Registration Process to be complete.
/issueVC	POST	IssueVC is an endpoint exposed by the Verifiable Credential (VC) issuer dedicated for the creation of verifiable credential for a specific set of attributes. More specifically the VC issuer uses the DAASignature as an argument and it eventually receives two Verified Credentials: SSI_VC and PRIVÉ_VC.
/storeDaaPubKey	POST	The purpose of this API, owned by the VC issuer, is to handle the storage and comparison of the fingerprint data from the PRIVÉ wallet user with the corresponding public key generated by the DAA issuer.
/join	POST	This endpoint is exposed by the DAA Issuer and is dedicated for handling the initial challenge of a wallet that wants to get onboarded to the PRIVÉ infrastructure. More specifically, the DAA Bridge sends to the DAA Issuer, the structures that hold the public part of the DAA and Endorsement keys and the key restriction usage policy in which each key is binded .The DAA Issuer checks whether or not he is communicating with a valid TPM, via the provided Endorsement key and recalculates the approved key restriction usage policy under which the DAA key should be binded, in order to attest against the key restriction usage policy provided by the wallet.If both checks are completed successfully the DAA Issuer creates a first set of

		credentials encrypted under the both the Endorsement and the DAA key of this particular TPMand an authorization ticket that dictates the wallet correctness.
/sign	POST	The Sign endpoint is exposed by the DAA Issuer and is responsible for completing the onboarding of a wallet into the PRIVÉ infrastructure.More specifically, after the wallet has handled the DAA Issuer's response over the initial join challenge and created a DAA signature over the secret that was wrapped with the Both the Endorsement and the DAA key of the wallet's TPM, the DAA Issuer verifies that the signed message is indeed the secret that he issued for this wallet and then verifies the provided DAA signature. If both the DAA signature and the signed message are verified successfully, the DAA Issuer creates the full DAA credentials for this specific wallet, signs them, encrypts them and finally wraps the secret encryption key under the TPM's Endorsement and DAA key.The encrypted credential, secret encryption key and the signature of the credential are sent back the wallet.
/verify	POST	/Verify, is residing within the DAA Issuer. Its main purpose is to verify a DAA signature and notify DAA Bridge with a Success/Fail response.
/make_daa_credential	POST	MakeCredential is a function of the DAA Issuer embedded in both /InitJoin and /Sign. More specifically it takes as input the public part of the endorsement key of the challenging TPM and the public part of the DAA key ,in order to create a

		secret symmetric encryption key and wrap it under the aforementioned keys.
/consumeServices	POST	This API currently serves as a simulated tool within the PRIVÉ ecosystem, demonstrating how the wallet interacts with external service providers such as banks and institutions. In a real-world scenario, this functionality could be seamlessly integrated into the user interface (UI) rather than being a separate API. In this process, the wallet showcases the ID of the service to the institution and, in return, obtains a policy with the list of attributes needed by the wallet holder and also a Nonce to secure the communication from replay attacks.
/enableService	POST	This API currently serves as a simulated tool within the PRIVÉ ecosystem, demonstrating how the wallet interacts with external service providers such as banks and institutions. In a real-world scenario, this functionality could be seamlessly integrated into the user interface (UI) rather than being a separate API. In this process, the wallet showcases the Verified Presentation (VP) to the institution and, in return, it gets a message that the access in the service has been granted.
/InitJoin	POST	This API currently resides in the remote wallet of PRIVE. It is responsible for handling the creation of the DAA key and handling the issuance of the DAA credential. More specifically, through this API call the PRIVE wallet offloads the heavy cryptographic operations to an external server, the Remote Core. The Remote Core that has a TPM, creates the DAA key and establishes a secure and

		authenticated channel with the DAA Issuer in order to acquire its DAA credential.
/DAAsign	POST	This API is offered by the Remote Core to be invoked by the PRIVÉ wallet. With this API the PRIVÉ wallet offloads the creation of a DAA signature to the Remote Core. More specifically the Remote Core uses the specialised issued DAA credential in order to to randomize them and then calculates the DAA signature. For this functionality to be completed successfully the Remote Core has to prove that it is in a correct configuration/state, by passing all the key restriction policy checks that the DAA Issuer has defined.
/enableService	POST	This API resides in the Remote Core and is responsible for verifying a Verifiable Credential issued from the VC Issuer. More specifically the wallet forwards the verifiable credential issued for a new set of attributes to the Remote Core. The Remote Core uses the public key of the VC Issuer to create a bilinear map and verify using pairings the validity of the freshly acquired Verifiable Credential.
/createVP	POST	This API resides in the Remote Core and is responsible for creating a Verifiable Presentation over a specific Verifiable Credential, disclosing a specific subset of the corresponding attributes. More specifically, the Remote Core randomizes the requested Verifiable Credential, discloses only the requested attributes and finally uses its DAA key to finalise the calculation of the DAA Verifiable Presentation. It goes without saying that the Remote Core has to successfully pass all the policy checks

that are defined by the DAA Issuer in order to use its DAA key, as it is considered unTrusted and needs to prove that it is in a correct state.

4 TRUSTCHAIN FRAMEWORK SETUP

One of the components of the TRUSTCHAIN consortium is Alastria, which is a non-profit association that promotes the development and adoption of blockchain technology. It aims to create a collaborative environment for businesses, institutions, and developers to leverage blockchain for various uses.

Alastria's role as a network provider in TRUSTCHAIN is integral to the framework's success. Their contributions in providing a robust blockchain platform, technical support, fostering innovation, ensuring interoperability, and community engagement are key to realizing the vision of TRUSTCHAIN.

Alastria provides the underlying blockchain network essential for the TRUSTCHAIN framework. This network serves as the backbone for all decentralized applications, ensuring secure and transparent transactions. The Alastria network is designed to be scalable, reliable, and capable of supporting a diverse range of use cases.

4.1 INFRASTRUCTURE FOR DEPLOYMENT

Alastria facilitates various blockchain networks, supporting different technologies to meet the diverse needs of its members. These networks focus on general-purpose applications and are aligned with regulations.

In blockchain technology, there are three main types of networks:

- Public Blockchains: These are open and decentralized networks where anyone can join and participate without permission. They offer transparency and immutability but can be less efficient due to their size and openness.
- Private Blockchains: Operated by a single organization or entity, these blockchains are closed and access is restricted. They offer greater control and efficiency but at the cost of decentralization.
- Permissioned Public Blockchains: These are a hybrid model, like the network Alastria offers. They are public but require permission to join. This model provides a balance between the openness of public blockchains and the control of private ones. Alastria's network supports this model, allowing for regulated participation while maintaining some benefits of decentralization.

Funded by the European Union

In networks such as Alastria, all participants are required to be identified since they are part of a closed group. Governance is decentralized through consensus mechanisms, specifically Proof of Authority (PoA), executed by the nodes of various participants.

In Alastria, it is mandatory for each partner to operate their own node. There are three types of nodes:

- Regular Nodes: They play a role by replicating the blockchain, accepting blocks generated by the validators, and executing transactions contained within these blocks. Additionally, they are authorized to inject transactions into the network from external sources. Each regular node maintains a real-time, updated copy of the network, thereby ensuring a secure and decentralized DLT blockchain network.
- Validator Nodes: Validator nodes are tasked with validating transactions and relaying them to regular nodes. These nodes then accept and distribute the new blocks throughout the network. Validator nodes are crucial for maintaining network consensus and block generation, achieved by running the consensus algorithm (IBFT).
- Boot Nodes: Authorization or enabling of the Permitting Nodes to the Regular Nodes so that, using the gas limit granted, they propose (initiate), carry out (write), or consult (read) transactions. They act like proxy between Regular and Validator nodes.

For the TRUSTCHAIN project, an API has been developed that allows teams to access the Alastia blockchain network without the need to be a member of the association or contribute a node to the network infrastructure. This ensures that all participants have access to a distributed and secure network, where, most importantly, all project use cases are guaranteed a predictable gas cost: zero (0).

At this point, there are two networks running separately in the Alastria's ecosystem.

- T Network: This is the network based on Quorum technology. It is a robust, stable, and highly resilient network. The digital identity model Alastria_ID is deployed on this network.
- B Network: This network is based on Hyperledger Besu technology. It is a modern, powerful network with the same technology used by the European EBSI network and the Latin American LACChain network. The digital identity model Alastria_ID is deployed on this network.

4.2 DIGITAL IDENTITY

Alastria_ID is a digital identity model proposed by the consortium of Alastria for use in digital services and is inspired by the concept of self-sovereign identity (SSI). The aim is to provide a digital identity that anyone can use for all kinds of online transactions. Itis deployed as one of the core applications of the blockchain infrastructure and is based on the ten key SSI principles, which are grouped into three pillars: security, controllability, and portability.

It is an open-source model built collaboratively by the members of the consortium, undergoing continuous revision, and testing to bolster security and privacy, and complying with Spanish and European regulations.

The MVP 2.0 version of Alastria_ID is currently available to all members, as is a set of how-to documents in Alastria's Git repository. Version 2.0 has introduced some new optional fields that are needed in the items to be able to adapt real use cases. It also comes with improved how-to documents to explain how to deploy the upgradeable smart contracts, and the libraries and examples have been tested after the development of these new smart contracts to ensure that the model works correctly.

4.3 SUSTAINABILITY AND GROWTH OF THE PILOT DEPLOYMENT

Progress has also been made on the backup and recovery procedures for the identity wallet, and the GDPR guidelines have been taken into consideration to offer users a greater degree of discretion when registering credentials and submissions, bringing us closer to a model in which only the information that users want and find most useful is added to the blockchain ledger.

Alastria_ID's roadmap for the coming months sets out a series of improvements that will lead to versions 2.1 and 2.2, which seek to reduce the number of logs made on the blockchain and increase the optional features for the different changes for greater alignment with the GDPR. Version V2.1 will allow fully optional anchoring of credentials and submissions, adding some new features while preserving the previous ones. V2.2 will then remove some unneeded features, entailing a more disruptive change. Both versions will be developed and phased in progressively to allow members ample time to adapt.

Looking to the longer term, work has started on EPIC, which will be a major enhancement to the ID_Alastria model. Using single-use derived keys, it will further increase security and privacy while maintaining full capabilities in terms of key rotation and revoking credentials and submissions.

In addition to the model's technological upgrade, the working groups also presented developments of a legal nature. In this regard, progress has been made on the findings of the preliminary impact assessment (PIA), which seeks to gauge the impact and risks of ID_Alastria's processing activities and to ensure the protection of personal data. These findings are being reviewed in line with the technical developments and modifications being made to ID_Alastria's configuration. As a final point, work has continued the definition of the ID_Alastria model's trust framework regarding the policy guidelines for issuing credentials.

4.3.1 Development

Long-term Vision:

- Establish TRUSTCHAIN as a leading framework in user privacy and data governance, contributing significantly to the evolution of data economics and democracy.
- Aim to position TRUSTCHAIN as a key integrator in multi-chain environments, enabling seamless data and asset exchange across various blockchain networks.
- Foster TRUSTCHAIN as a sustainable ecosystem with long-term societal impact, integrating cutting-edge solutions in user privacy and data governance.
- Establish Alastria as a key player in advancing blockchain technology for various sectors, emphasizing its role in fostering a sustainable and ethical digital economy.
- Focus on enhancing data privacy and security while promoting transparent and efficient data governance models.
- Strive to become a benchmark for blockchain innovation and collaboration globally.

Scalability Plans:

- Design the TRUSTCHAIN infrastructure to support a growing number of users and diverse applications, ensuring it can handle increased data traffic and transactions.
- Implement scalable solutions that can be easily adapted or expanded to meet future demands, focusing on maintaining performance and security at scale.
- Encourage the development and integration of technologies that promote scalability, such as advanced data management tools and efficient consensus mechanisms.
- Design the network architecture to handle an increasing volume of transactions and users seamlessly.
- Implement adaptable and modular technologies that allow for easy expansion and upgrades.
- Foster partnerships and collaborations to integrate emerging technologies that enhance network scalability.

Environmental Considerations:

Funded by the European Union

- Incorporate energy-efficient technologies and sustainable practices in the development and operation of TRUSTCHAIN.
- Strive for a minimal environmental footprint by optimizing resource usage and reducing energy consumption.
- Regularly assess and update environmental strategies to align with global sustainability goals.
- Incorporate energy-efficient technologies and practices to minimize the environmental impact.
- Explore and implement sustainable blockchain solutions like Proof of Stake or other less energy-intensive consensus mechanisms.
- Commit to regular reviews and updates of practices to align with the best environmental standards.

Community and Ecosystem Growth:

- Develop a vibrant community of innovators, investors, industry players, and public authorities, creating networking opportunities and collaboration platforms.
- Offer support services like communication toolkits, co-branding materials, and promotional opportunities to increase visibility and engagement within the TRUSTCHAIN ecosystem.
- Invite participation in key events and media channels, fostering a culture of innovation and collaboration among all stakeholders.
- Encourage a diverse and inclusive community of developers, users, and stakeholders.
- Launch initiatives like hackathons, workshops, and training programs to engage and expand the community.

Promote open-source development and collaborative projects to foster innovation within the network.

Funded by the European Union

5 CONCLUSION

This document compiles the results achieved from projects funded between August 2023 and April 2024. It serves as a guiding resource for prospective applicants for upcoming open calls.

Within this report, we meticulously outline the prerequisites and the hierarchical structure of the TRUSTCHAIN framework. Specifically, the development efforts of the 13 funded projects under Open Call #1 have introduced novel functionalities to the TRUSTCHAIN framework and facilitated the creation of various components within it. Furthermore, the comprehensive documentation of these components and their interfaces in this deliverable will aid future participants in extending these functionalities and incorporating new features into the TRUSTCHAIN framework.

Additionally, we provide specifications for setting up the TRUSTCHAIN framework, offering a deeper understanding of the outcomes of this endeavour.

Funded by the European Union

REFERENCES

[1] D. Collingridge, The Social Control of Technology, New York: St. Martin's Press, 1980.

[2] W. Moritz, D. My Hanh and S. Remmer, “The environmental impact of cryptocurrencies using proof of work and proof of stake consensus algorithms: A systematic review,” *Journal of Environmental Management,* pp. 116-530, 2023.

[3] “Ethreum Virtual Machine (EVM),” 7 November 2023. [Online]. Available: https://ethereum.org/developers/docs/evm. [Accessed 19 January 2024].

[4] OpenZeppelin, “OpenZeppelin | docs,” 2023. [Online]. Available: https://docs.openzeppelin.com/contracts/5.x/. [Accessed 10 January 2024].

[5] D. Abebe, Z. Lu, S. Akanksha, K. Shahriar and H. Pham Cong, “Leveraging zero knowledge proofs for blockchain-based identity sharing: A survey of advancements, challenges and opportunities,” *Journal of Information Security and Applications,* vol. 80, pp. 103-678, 2024.

[6] E. Comission, “EBSI explained, Chapter 5 - EBSI Trust Models explained,” June 2022. [Online]. Available: https://ec.europa.eu/digital-building-blocks/sites/display/EBSI/Explained+Series. [Accessed 17 January 2024].

[7] E. Comission, “European Digital Identity Architecture and Reference Framework,” January 2023. [Online]. Available: https://ec.europa.eu/newsroom/dae/redirection/document/93678. [Accessed 17 January 2024].

[8] “eIDAS Regulation,” 17 May 2023. [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation. [Accessed 17 January 2024].

[9] E. Comission, “European Interoperability Framework - Implementation Strategy,” Brussels, 2017.

Funded by the European Union