D5.8 Final Release of DE4A Self-Sovereign Identity Supporting Framework

Document Identification			
Status	Final	**Due Date**	30/04/2022
Version	1.0	**Submission Date**	25/05/2022

Related WP	WP5	**Document Reference**	D5.8
Related Deliverable(s)	D4.1, D4.2, D5.7	**Dissemination Level (*)**	PU
Lead Participant	ATOS	**Lead Author**	Angel Palomares (ATOS)
Contributors	Martina Šesta, Muhamed Turkanović, Damijan Novak, Vid Keršič (UM), Javier Presa, Alberto Crespo (ATOS)	**Reviewers**	Miguel Correia (INESC) Harold Metselaar (ICTU)

Keywords :
Blockchain, Self-Sovereign identity, DID, Verifiable Credential, EBSI, ESSIF, Hyperledger GO.

Disclaimer for Deliverables with dissemination level PUBLIC

This document is issued within the frame and for the purpose of the DE4A project. This project has received funding from the European Union's Horizon2020 Framework Programme under Grant Agreement No. 870635 The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

[The dissemination of this document reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains. This deliverable is subject to final acceptance by the European Commission.

This document and its content are the property of the DE4A Consortium. The content of all or parts of this document can be used and distributed provided that the DE4A project and the document are properly referenced.

Each DE4A Partner may use this document in conformity with the DE4A Consortium Grant Agreement provisions.

(*) Dissemination level: PU: Public, fully open, e.g. web; CO: Confidential, restricted under conditions set out in Model Grant Agreement; CI: Classified, Int = Internal Working Document, information as referred to in Commission Decision 2001/844/EC.

Document Information

List of Contributors	
Name	Partner
Martina Šestak	UM
Muhamed Turkanović	UM
Damijan Novak	UM
Vid Keršič	UM
Javier Presa	Atos
Alberto Crespo	Atos
Angel Palomares	Atos

Document History			
Version	Date	Change editors	Changes
0.1	01/04/2022	Angel Palomares (ATOS)	Initial version of document
0.2	10/04/2022	Angel Palomares (ATOS)	Consolidated draft version of document
0.3	12/04/2022	Angel Palomares (ATOS)	Contribution section 2
0.4	17/04/2022	Muhamed Turkanović (UM)	Contribution section 2.3, section 3.3
0.5	18/04/2022	Angel Palomares (ATOS)	Contribution sections 3 and initial draft section 4
0.6	20/04/2022	Muhamed Turkanović (UM)	Contribution section 3 and section 4
0.7	25/04/2022	Angel Palomares (ATOS)	Contribution to annex I, section 5
0.8	5/05/2022	Angel Palomares (ATOS)	Review process
0.9	17/05/2022	Angel Palomares (ATOS)	Comments from reviewers included
0.91	18/05/2022	Julia Wells (ATOS)	Final format check and update for submission
1.0	24/05/2022	Ana Piñuela (ATOS)	Final for submission

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Angel Palomares (ATOS)	17/05/2022
Quality manager	Julia Wells (ATOS)	18/05/2022
Project Coordinator	Ana Piñuela Marcos (ATOS)	24/05/2022

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
AA	Authority Agent
API	Application programming interface
BEP	Backend of the Evidence Portal
DC	Data consumer
DLT	Distributed Ledger Technologies
DP	Data provider
Dx.y	Deliverable number y, belonging to WP number x
DID	Decentralized identifier
EBSI	European Blockchain Services Infrastructure
EDCI	European Digital Credential Infrastructure
eIDAS	Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC
ESSIF	European Self-Sovereign Identity Framework
FEP	Frontend of the Evidence Portal
JWT	JSON (JavaScript Object Notation) web Token
MDS	Minimum Data Set
MS	Member State
MVP	Minimum Viable Product
OOP	Once-only Principle
OOP TS	Once-only technical system for evidence exchange in the DE4A project
PKI	Public Key Infrastructure
QR (code)	Quick Response code
SA	Studying Abroad Pilot
SDGR	Single digital gateway regulation
SSI	Self-sovereign identity
TIR	Trusted Issuer Registry
TSR	Trusted schema registry
VC	Verifiable credential
VDR	Verifiable data registries
VP	Verifiable presentation

Executive Summary

This deliverable describes the final version of the DE4A Self-Sovereign Identity Supporting solution that implements the Verifiable Credentials pattern, which is piloted in the Diploma Recognition use case of the DE4A Studying Abroad pilot. This use case validates the scenario where students request their diploma in the form of Verifiable Credentials issued from national Higher Education portals and, after storing these credentials in their mobile digital wallets, present them to the service providers to finish the procedure. The Diplomas use case is piloted in DE4A by students of three Member States (Portugal, Slovenia and Spain).

The DE4A solution has two main components which are the Mobile Application or Mobile Edge Agent (to be used by the students) and the Authority Agent (to be installed/integrated in the server side). Students use the mobile app to store their diplomas (in the form of a Verifiable Credentials) issued from a competent authority (Data Provider or Issuer) in the country where they completed corresponding Higher Education studies and provide such evidence for recognition in another country to the Data Consumer authority (Verifier). The Authority Agent supports the interaction between the student (diploma Holder) and the diploma Issuer or Verifier and overall communication flow in the Verifiable Credentials pattern and the Mobile Edge Agent. Both Data Provider and Data Consumer leverage the Authority Agent.

The provided solution is a mature prototype that enables piloting with real students and demonstrates the real use in a cross-border context of a user-centric information exchange pattern realised over self-sovereign standards integrated with EBSI infrastructure. While differences in specific requirements and timing between the project and EBSI/ESSIF had an influence on functionalities that could be included into the Self-Sovereign Identity solution implemented by DE4A, these were not of a critical nature and a very high and satisfactory level of alignment was effectively achieved through participation of the project in EBSI’s Early Adopters programme. In addition to major technical achievements and valuable technical results like the generation of EBSI-compliant Decentralised Identifiers that are also interoperable with those generated and used in major DLT frameworks such as Hyperledger Aries to ensure compatibility with industry initiatives, the relevance of achieved integration with EBSI also acquires a strategic significance in future roadmaps of public sector stakeholders which are working with the European Commission for the introduction of a new European Digital Identity Framework under the proposal for new eIDAS regulation [37][38].

1 Introduction

1.1 Purpose of the document

This document is the second and final version of the *D5.7 First Release of DE4A Blockchain Supporting Framework [1].* As the reader may already noticed, the name of the deliverable has changed to *D5.8 Final Release of DE4A Self-Sovereign Identity Supporting Framework*. This change has been made to avoid misunderstandings, as DE4A does not develop and deploy a blockchain infrastructure, but a Self-Sovereign Identity solution integrated with EBSI/ESSIF and piloted in the Diplomas recognition use case of the Studying Abroad pilot. The consortium initially considered the possibility of implementing a dedicated ledger based on blockchain for storing the support material of the regular SSI operations for the DE4A use case in scope in case EBSI ledger was not available for integration within the project timeframe. However, as DE4A confirmed with EBSI the feasibility of timely integration within the Early Adopters program the dedicated ledger option was discarded.

The main aim of this deliverable is to describe from a technical perspective the DE4A Self-Sovereign Identity solution that will be used for the exchange of verifiable credentials in the Diploma recognition use case between the respective Competent Authorities and the students. In addition, it is worth highlighting that the dissemination level of this document is *Public*, with the main audience being future providers and adopters of decentralised approaches (technology providers, public sector authorities) and, in general, all those interested in the way that innovative technologies can be applied for the modernisation of electronic services. Achievements can be classified in three main groups:

- Development and deployment of all the infrastructure necessary for supporting the adoption of a Self-Sovereign Identity Solution.
- Development and integration of the components on both sides of the communication in the use cases, Data Providers ('Issuers' of Verifiable Attestations) and Data Consumers ('Verifiers' of Verifiable Presentations), with the support of the mobile wallet ('Edge Client') on the user side ('Holder').
- Development and integration of middleware layers between the EBSI [18]/ESSIF[31].

1.2 Structure of the document

The document is divided into the following chapters:

- Chapter 2 that provides the definition of concepts widely used across de document (e.g. DID, Verifiable Credential, etc) and makes a comparative of the available SSI solutions that could be used as underlaying framework for DE4A.
- Chapter 3 that describes EBSI and ESSIF initiatives and the main services provided by them and used by DE4A implementations.
- Chapter 4 that describes in detail the whole life-cycle design of the Self-Sovereign supporting framework:
  - Pilot and use case where the Self-Sovereign Identity solution will be piloted
  - Components of the solution
  - Deployment of the components
  - Message/Workflows with the data flows among the different components and stakeholders
  - Legal and Security considerations associated to the implementation of this solution.
- Chapter 5 with the conclusions.

- Appendix I - Legal discussion note on using Verifiable Credentials in the DE4A Studying Abroad pilot: covering, from the context of defined ambition in DE4A, an overview of different legal challenges and working assumptions which were presented and discussed together with EBSI-ESSIF experts.

2 Self-Sovereign Identity Background

2.1 SSI concepts

Some of the different concepts defined to detail a Self-Sovereign Identity approach are relativity new and sometimes can be confusing for an outside reader. Hence this section provides a definition of the terms and concepts that later will be widely used on the rest on this report. For the definition of these concepts and terms, this report follows the W3C Verifiable Credential Recommendation [32] and the EBSI/ESSIF terminology [33] as key references on which the solution is developed and integrated respectively.

Decentralised Identifier (DID)	
VC W3C Definition	
A portable URL-based identifier, also known as a DID, associated with an entity. These identifiers are most often used in a verifiable credential and are associated with subjects such that a verifiable credential itself can be easily ported from one repository to another without the need to reissue the credential. An example of a DID is did:example:123456abcdef.	
EBSI Definition	
A decentralised identifier can uniquely identify a Party (Issuer, Owner/Holder, Relying Party) is fully under this Party's control and used for referring to it. It can be anywise, pairwise or n-wise	

Verifiable Credential (VC)	
VC W3C Definition	
A verifiable credential is a set of one or more claims made by the same entity. Credentials might also include an identifier and metadata to describe properties of the credential, such as the issuer, the expiry date and time, a representative image, a public key to use for verification purposes, the revocation mechanism, and so on. The metadata might be signed by the issuer. A verifiable credential is a set of tamper-evident claims and metadata that cryptographically prove who issued it.	
EBSI Definition	
A verifiable credential is a tamper-evident credential that has authorship that can be cryptographically verified. Verifiable credentials can be used to build verifiable presentations, which can also be cryptographically verified. The claims in a credential can be about different subjects. Verifiable means that the integrity (no alteration) of a Verifiable Credential, as well as the authorship of a Verifiable Credential, can easily be checked using a cryptographic-based standard procedure	

Verifiable Presentation (VP)	
VC W3C Definition	
Data derived from one or more verifiable credentials, issued by one or more issuers, that is shared with a specific verifier. A verifiable presentation is a tamper-evident presentation encoded in such a way that authorship of the data can be trusted after a process of cryptographic verification. Certain types of verifiable presentations might contain data that is synthesized from, but do not contain, the original verifiable credentials (for example, zero-knowledge proofs).	

EBSI Definition	
A verifiable presentation represents the data passed from an entity to a relying party (often also the verifier).	

Holder	
VC W3C Definition	
A role an entity might perform by possessing one or more verifiable credentials and generating presentations from them. A holder is usually, but not always, a subject of the verifiable credentials they are holding. Holders store their credentials in credential repositories.	
EBSI Definition	
A holder will be defined as the entity that is the receiver of a verifiable credential (not necessarily owned by it) and that can use it.	

Issuer	
VC W3C Definition	
A role an entity can perform by asserting claims about one or more subjects, creating a verifiable credential from these claims, and transmitting the verifiable credential to a holder.	
EBSI Definition	
A role that an entity, a person, or a thing might perform by creating a verifiable credential, associating it with a specific subject, and transmitting it to a holder. Example issuers include corporations, non-profit organizations, trade associations, governments, and individuals. The trustworthiness of ESSIF will stand (or fall) with the trustworthiness of the verifiable credentials, mandates/consents, and/or claims. This trustworthiness will be determined by the trustworthiness of the respective issuers and their issued VCs (which can be low, substantial, or high).	

Verifier	
VC W3C Definition	
A role an entity performs by receiving one or more verifiable credentials, optionally inside a verifiable presentation for processing. Other specifications might refer to this concept as a relying party.	
EBSI Definition	
ESSIF doesn't define explicitly the Verifier role but assimilates it to that of Relying Party (parties which through their actors/agents rely on any verifiable credential they will receive)[33].	

2.2 Analysis of existing SSI solution platforms

The previous version of this deliverable, D5.7, includes a complete analysis on the available solutions to be used as underlaying framework by the DE4A project. As D5.7 is not a public document, the consortium has considered appropriate to summarize the previous analysis in the current deliverable for its wider audience.

2.2.1 Evaluated tools

The consortium assessed several different possible software options and the most promising ones were the following:

2.2.1.1 uPort

uPort [6] was an American company created in 2016, with the main aim of providing a Self-Sovereign Identity solution to the end user. The solution provided by uPort was one of the pioneers in the community and one of the first to provide a complete set of tools for applying a Self-Sovereign Identity approach. This company has been acquired by Consensys Mesh and since then its policy and strategy has changed, focusing more on providing services for companies and leaving aside the development focused on the end users. uPort had been one of the main actors in the SSI environment, as it provided a mature working open-source solution under an Apache 2 License. It is also worth mentioning that the uPort solution is strongly tied to an Ethereum ledger. uPort was discontinued in early 2021 and became serto and Veramo.

2.2.1.2 Decentralized Identity Foundation (DIF)

The Decentralized Identity Foundation[7] is a non-profit organization with more than 50 different partners. Among other companies and organizations, Microsoft, Sovrin, Evernym and uPort are members of the Identity Foundation. The Identity Foundation contributes to the SSI community in three different ways: contributing to the development of standards, dissemination of the main results of a Self-Sovereign Identity community and, finally, providing software tools. Therefore, this analysis is focused on the software tools provided by the Identity Foundation and the most significant groups providing different tools are as follows:

- Identifiers and Discovery: enable creation, resolution, and discovery of decentralized identifiers and names across decentralized systems, like blockchains and distributed ledgers.
- Authentication: Designing and implementing DID-based authentication specs, standards, and libraries used in authenticating DIDs across a wide variety of exchanges and use cases.
- Claims and Credentials: The ability to verify the claims and assertions of identities is key in establishing trust among entities on a decentralized system that lacks a centralized hierarchy.
- DID Communication: Produces one or more high-quality specs that embody a method (“DIDComm”) for secure, private and (where applicable) authenticated message-based communication.

It is worth mentioning that each of the groups mentioned above provides different sets of tools under an Apache 2 Licensing from different contributors and with different optional approaches/technological solutions for the same issue.

2.2.1.3 Hyperledger

Hyperledger is a project promoted by the Linux Foundation for developing a set of blockchain collaborative open-source solutions created in 2015 and with the support of companies such as IBM, Intel and SAP Ariba. Hyperledger provides different frameworks for specific issues proposing specific blockchain approaches and solutions for them. It is worth mentioning that some of the Hyperledger frameworks have become the most extended and applied in the industry. Nevertheless, this section will only be focused on those frameworks provided by the Hyperledger Project to be applied in Self-Sovereign Identity solutions: Hyperledger Indy and Hyperledger Aries.

Hyperledger Indy

Hyperledger Indy[17] is the first Hyperledger’s attempt to provide a Self-Sovereign Identity solution. Initially the solution provided by Hyperledger Indy was based on an existing solution implemented by the Sovrin Foundation. Thereby the solution proposed by Hyperledger Indy is strongly coupled with its own blockchain definition and totally aligned with the services provided by Sovrin. Although this project can be considered as a consolidated solution and with mature software, it is clear that this solution is implemented in Python, which makes the integration within mobiles developments very difficult **Hyperledger Aries**

Hyperledger Aries [22] is a project devoted to providing a ledger-agnostic framework for developing Self-Sovereign Identity solutions. Therefore, Aries provides a solution based on a blockchain-rooted, peer-to-peer functionality for any ledger infrastructure. Thereby the main objectives of Aries are to provide a framework that allows peer-to-peer interactions based on DID Comm standard, the management of secrets, verifiable credentials, and a secure messaging mechanism.

Hyperledger Aries provides several different implementations, among which the most prominent are the following repositories:

- aries-cloudagent-python: this repository provides a Python implementation of the framework. However, the code provided is an adaptation of previous development provided by Hyperledger Indy. Hence some of the functionality implemented by this software uses Indy project approach so it is strongly tied to the Indy's specific blockchain solution. Just as an example, the mobile solution proposed for this implementation embeds the Indy-SDK and therefore does not follow the ledger-agnostic Hyperledger Aries's policy.
- aries-framework-go: this repository has been developed from scratch focusing on the new ledger-agnostic approach implemented using Go language. Thereby, this repository is under development at the moment of writing of this report, so not all the aspects related with a Self-Sovereign Identity solution are covered. Nevertheless, the repository is very active and is evolving to a mature framework very quickly.

2.2.2 Conclusion

In order to evaluate the most suitable solution for the DE4A developments, the consortium has assessed, among others, the following requirements:

- **License**: DE4A has prioritised open-source solutions.
- **Multi-ledger support**: The SSI solution developed by DE4A must be aligned with EBSI and ESSIF and therefore it cannot be strongly tied to a specific ledger implementation.
- **Mobile integration:** The solution should provide mobile integration for allowing the students to easily use the DE4A implementation with their own mobile devices.
- **Standards-supported:** The solution should comply with the main common standards by the SSI community and EBSI/ESSIF.
- **Implementation Languages**: The final technology used by the underlying framework will have a direct impact on the DE4A implementations.

The following table summarizes the assessment of the above-mentioned solutions

Table 1: Available solutions

	License	Multi-ledger support	Mobile integration	Standard supported	Implementation Language
uPort	Apache 2	⛔	✓	DID, VC (partially)	JavaScript
DIF	Apache 2	✓	✓	DID, VC, DID Comm	JavaScript, Java
Hyperledger Indy	Apache 2	✓	✓	DID, VC	Python
Hyperledger Aries	Apache 2	✓	✓	DID, VC, DID Comm	GO

As the above table shows, the Hyperledger Aries option is the only one that complies with the different requirements necessary for the DE4A project's developments, especially the multi-ledger support (in order to facilitate the integration with EBSI/ESSIF) and the mobile integration (to allow the students to use their own devices to obtain their diplomas). Hence the project has chosen the already mentioned, Hyperledger Aries framework, as underlying software framework.

2.3 Interaction with Blockchain ledger

SSI solutions often utilize blockchain ledgers as Verifiable Data Registries (VDR), which enable SSI to play the role of traditional public-key infrastructure (PKI). The reason for this is that traditional (centralized) PKI-based certificates leverage the PKI for authenticity and identity verification reasons, which in the contrary is not simply possible in SSI, where the DIDs are decentralized identifiers generated by entities themselves. However, for the verification purposes of DID's subjects, SSI also needs some publicly available infrastructure for authenticity and identity information, which the blockchain ledgers play the main role due to their decentralized nature. Furthermore, SSI envisions also various business use cases where robust and publicly available data storage is needed. An example of such use cases is revocation lists of VCs. Hence, blockchain ledgers are again the solution for this. One of the most popular SSI blockchain ledgers is Hyperledger Indy, which is designed specifically for SSI use cases and DID/VC operations. In DE4A, the EBSI blockchain, based on the open-source Hyperledger Besu and Hyperledger Fabric and developed by the European Blockchain Partnership, is used as its blockchain supporting framework. Blockchain/ledgers services fulfil the following roles:

- PKI for public keys of decentralized identifiers (DIDs),
- Data registry for schemas of Verifiable Credentials,
- Data registry for trusted issuers (in case of EBSI) and
- Data registry for revocation lists.

Registering a DID and its associated public key on the blockchain is called DID anchoring. A public key is generated locally on the device, and the DID is derived from the public key. Based on that, a DID Document is constructed from the DID and the public key, which is then anchored on the ledger with blockchain transactions. After anchoring, anyone can retrieve DIDs using DID resolvers. Retrieving DIDs mostly happens when verifying signatures of VCs and VPs, which are signed by private keys; therefore, verifiers must obtain the public keys of issuers for the verification process. In DE4A, the EBSI blockchain serves as VDR with did:ebsi used as did method.

Each VC and VP must follow a specific structure and data model to facilitate interoperability between different parties and stakeholders. JSON Schemas force VC to follow data models, which must be publicly resolvable. Schemas are defined by representatives of the use cases and must be registered to VDR.

Everyone can issue and digitally sign VCs; thus, signatures only do not make them trustworthy. Issuers must be publicly identified and accredited by the government or institutions for a particular type of credentials and certificates. This data is also published on VDR to be retrieved and verified by holders and verifiers.

The last usage of the ledger is for revocation lists of VCs. Specific VCs can be revoked or cancelled by issuers, e.g., driver's license, when breaking the law. This change is written to the ledger, and the verifier must check the status of VC when performing the validation procedure (this is not yet supported by ESSIF v2.0).

3 DE4A Alignment with EBSI/ESSIF

This chapter describes the close relationship between DE4A project and EBSI/ESSIF. In order to pilot the diplomas' use case of the studying abroad pilot, DE4A has as an objective to develop an SSI solution that on one hand, is aligned with the Studying Abroad pilot requirements and technical capacities of official Member State Authorities and that on the other hand, is cross-border and can be extended to more Member States through its alignment with major existing initiatives (EBSI/ESSIF, Europass). To this end, DE4A project develops a solution aligned with the EBSI/ESSIF specifications in the context of its Diplomas use case in the Early Adopters programme.

3.1 EBSI

Since 2018, aiming to realise the potential of blockchain-based services for the benefit of citizens and society as well as to enable economic growth, the EU Member States, Norway, Lichtenstein and the European Commission are collaborating in the European Blockchain Partnership (EBP) to build the European Blockchain Services Infrastructure – EBSI.

EBSI is the first EU-wide blockchain infrastructure, driven by the public sector, in full respect of European values and legislation. It has deployed from 2020 an operational network of distributed blockchain nodes across Europe and is supporting applications focused on selected use-cases to provide authorities and citizens with highly innovative cross-border services and applications. To see the already deployed nodes at a glance, the following figure shows the number of deployed nodes in 29 different countries, where the green nodes represent the ready to use nodes and the yellow acts for the nodes still in testing.

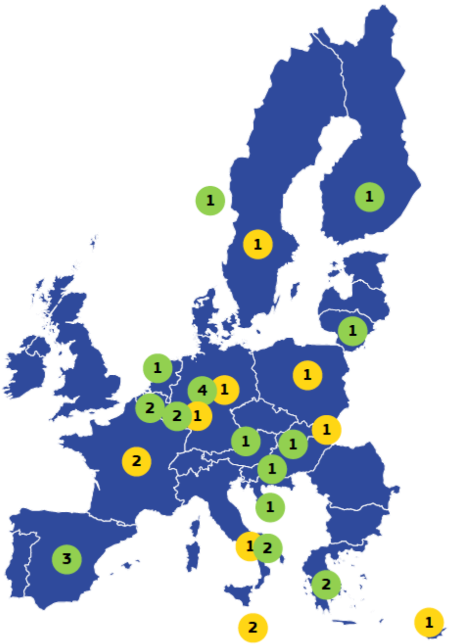


Figure 1: EBSI nodes distribution by country [45]

This was the initial set of EBSI use cases that were defined by the EBP in 2020 in order to test the infrastructure deployed by EBSI[44]:

- **European Digital Identity (ESSIF)**: Implementing a Self-Sovereign Identity model in Europe, allowing users to create and control their own identity across borders. For instance, citizens who want to set-up a digital wallet and manage their identity credentials across borders [41].
- **Diplomas**: Citizens gain digital control of their educational credentials, significantly reducing verification costs and improving trust in documents' authenticity. For example, students/young professionals who want to apply for a degree/job and manage their educational credentials [42].
- **Notarisation/Document Traceability**: Leveraging the power of blockchain to create trusted digital audit trails, automate compliance checks in time-sensitive processes and prove data integrity[45].
- **Trusted Data Sharing**: Leveraging blockchain technology to securely share data amongst authorities in the EU, starting with import one-stop-shops (IOSS) and the IOSS VAT identification numbers amongst customs and tax authorities.

Additional use cases were defined in 2021 including on social security and ESSPASS project (Social Security competent institution in a Member State can issue the PDA-1 document as a verifiable attestation and an inspector in another Member State can verify it), financing small and medium-sized enterprise (SME) through blockchain and facilitating the management of cross-border and cross-authority asylum demand processes.

As readers may already have noticed, among all the possible use cases for testing the blockchain infrastructure, those selected by EBSI on Digital Identity and Diplomas are strongly related and aligned with some of the DE4A objectives.

3.2 ESSIF

The European Self-Sovereign Identity Framework for decentralised identity management allows citizens to create, control, and use their own digital identity without having to rely on a single, centralised authority and it is also aligned with eIDAS revision. The approach taken by ESSIF is to encapsulate the access to the blockchain provided by EBSI through services accessed using a REST API.

The following figure shows the main components and services provided by ESSIF. It also shows a layered distribution of its components. The top part of the figure it can be seen the REST APIs of the services (labelled with Registry) for the end users either natural person or legal entities. Those services are the services used by DE4A project as well. The layer underneath, Smart Contract, contains the different components that will invoke the interaction with the blockchain executing smart contracts. And finally the bottom layer of the diagram, defined as State, corresponds to the components where the DLT technology stores the data. Following this layered approach, ESSIF provides a set of services that encapsulates in a black box the final use of the DLT infrastructure provided by EBSI, thus hiding the infrastructure implementation complexity (no need to use low-level primitives) from applications like those implemented in DE4A and other Early Adopters programme projects (this approach is also advantageous thinking of future adopters as it is not necessary to have a high-level expertise to use the Blockchain infrastructure).

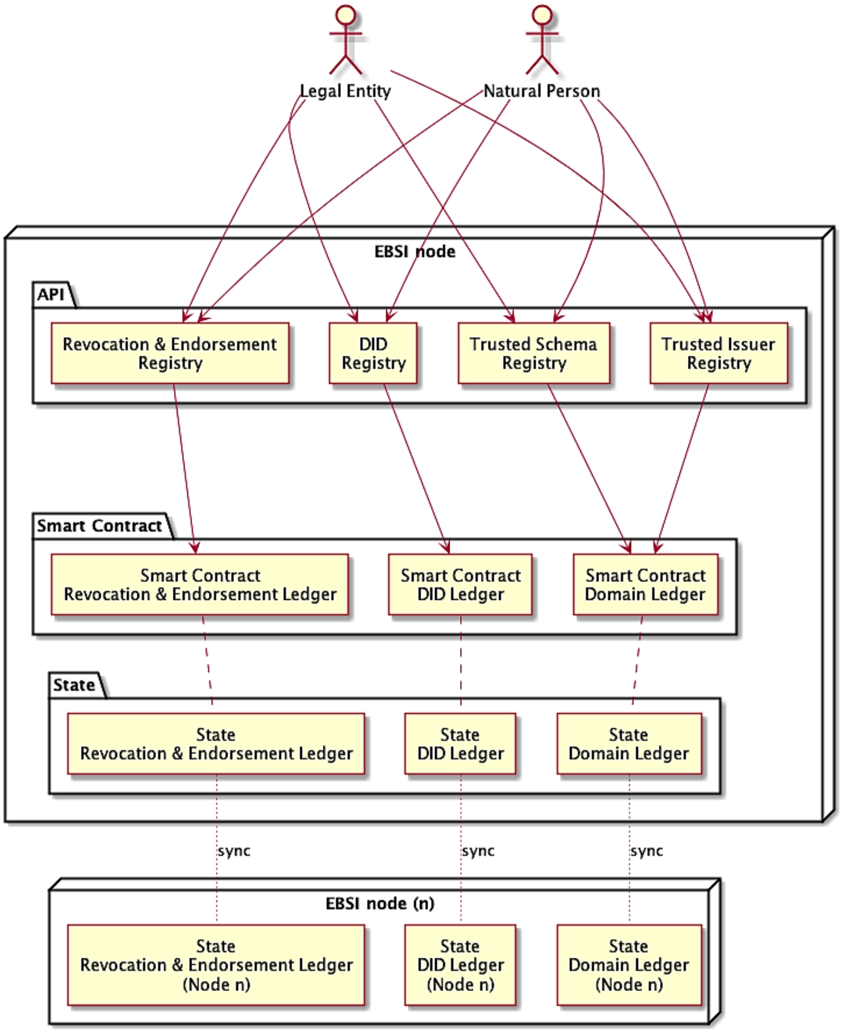

Legal Entity
Natural Person
EBSI node
API
Revocation & Endorsement Registry
DID Registry
Trusted Schema Registry
Trusted Issuer Registry
Smart Contract
Smart Contract Revocation & Endorsement Ledger
Smart Contract DID Ledger
Smart Contract Domain Ledger
State
State Revocation & Endorsement Ledger
State DID Ledger
State Domain Ledger
sync
sync
sync
EBSI node (n)
State Revocation & Endorsement Ledger (Node n)
State DID Ledger (Node n)
State Domain Ledger (Node n)

Figure 2: ESSIF main components[1]

Therefore, the main components used by DE4A are the following:

DID Registry [40]: This registry is the component which manages all the DIDs in the platform. It is responsible for writing, reading and updating all the DID and DID public keys for all the entities/users.

Trusted Issuer Registry (TIR)[39]: The TIR contains information about Issuers and which types of VCs they are accredited to issue. As such, its main purpose is to facilitate the evaluation of a VC's trustworthiness by a Verifier. For instance, through this component a legal entity could become an issuer, and then it could issue new credentials to the platform.

Trusted Schema Registry (TSR) [41]: The TSR contains data schemes (i.e., templates) of data objects, particularly of VCs. As such, its main purpose is to provide public information about (recognised) data models and contents of different VC types that can be re-used by Issuers to control the quality and semantic interoperability of VCs across systems.

[1] https://ec.europa.eu/digital-building-blocks/wikis/pages/viewpage.action?pageId=380862997

The Revocation & Endorsement Registry: Stores evidences about the verifiable credentials issued in the platform. This registry is unique per domain and it will be used for checking the validation of a credential.

Regarding ESSIF, it is important to mention that, in early 2021, ESSIF opened a call for the Early Adopters programme. As a result of this call, it selected and started to work directly with 22 projects led by Member States to implement the first batch of EBSI-chosen use cases (see section 3.1) to build an ecosystem with highly representative and outstanding real-life examples that will lead to the full rollout of EBSI. DE4A was chosen as one of the participating projects.

The Programme helps the first EBSI users and their partners to build and to launch their pilot project(s). EBSI team gave each project's private and public sector partners early access to the EBSI internal technical documentation, pre-production environment and invited them to develop their own pilot project(s) addressing a specific business or government use case involving the exchange of verifiable credentials. EBSI gave support during this process through periodic follow-up meetings with all projects and specific technical meetings to address integration issues and alignment and DE4A has been an active contributor in this process.

As mentioned before, DE4A participates from the ‘first wave’ in EBSI's Early Adopters' Programme with a pilot Use Case on Diplomas Verification which perfectly fits into the EBSI Use Case of Diplomas as it uses EBSI as a single point of truth to support in the education sector public services, the cross-border verification of such educational credentials based on Self-Sovereign Identity (SSI) principles. This represents a great benefit for DE4A project and the authorities involved as the developments provided by the project have a higher trustability than any other project providing a Self-Sovereign Identity solution but using their own DLT.

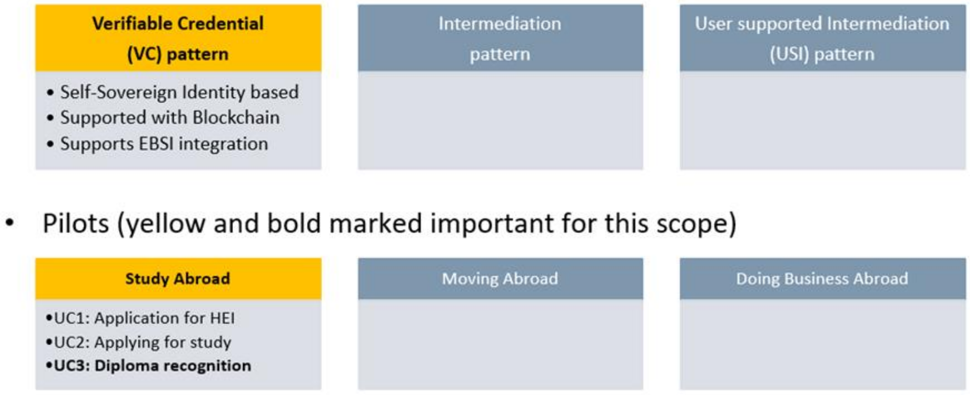
3.3 Adaptations for alignment with EBSI/ESSIF

One of DE4A project’s objectives was to explore innovative technologies (e.g., distributed ledgers, blockchain, SSI) in the context of the diplomas recognition use case of the Studying abroad pilot to analyse the benefits of such technologies related to improvement of process digitalisation in the public sector.

This use case uses the Verifiable Credentials (VC) pattern which is based on SSI principles and has its own vocabulary as it is shown in the table below. A complete description of this pattern and the rest of the patterns defined by the project is provided in D2.4 [2].

Table 2: Vocabulary differences between the DE4A VC pattern and SSI

VC pattern DE4A	SSI
Student	Holder
Evidence (Diploma)	Verifiable attestation (Verifiable Credential, Verifiable Presentation)
Data Producer (e.g., Ministry)	Issuer
Data Consumer (e.g., University)	Verifier
Identifier	DID, subject, …



Verifiable Credential (VC) pattern

- Self-Sovereign Identity based
- Supported with Blockchain
- Supports EBSI integration

Intermediation pattern

User supported Intermediation (USI) pattern

- Pilots (yellow and bold marked important for this scope)

Study Abroad

- UC1: Application for HEI
- UC2: Applying for study
- **UC3: Diploma recognition**

Moving Abroad

Doing Business Abroad

Figure 3: The scope and position of the VC pattern within the DE4A project

The usage of innovative technologies within the diploma's recognition use case in the Studying Abroad pilot was more complex than had been anticipated. The main challenges initially identified were:

- DE4A is a large-scale pilot EU MS project, which requires full legal and technical assurance for its piloting,
- Blockchain technologies are not yet mature and are still not well understood by stakeholders,
- Introduction of blockchain technology may require technical approaches which are not compatible with legacy systems of project partners,
- EBSI introduced its own identity management approach (Verifiable Identities), which did not seem suitable for DE4A due to the project requirement of using legally acceptable eID (eIDAS schemes and components),
- There were no standardized technical components, products and/or services, which could be used with ease while designing the DE4A self-sovereign solution,
- Storing personal data on blockchain ledgers is not acceptable according to GDPR,
- Usage of public blockchain ledgers is not possible due to additional costs for transactions and its lack of compatibility with legacy systems.

From the **Project Start Architecture** perspective, the VC pattern has been envisioning a blockchain-based Trust Architecture as it can be seen in the following figures.

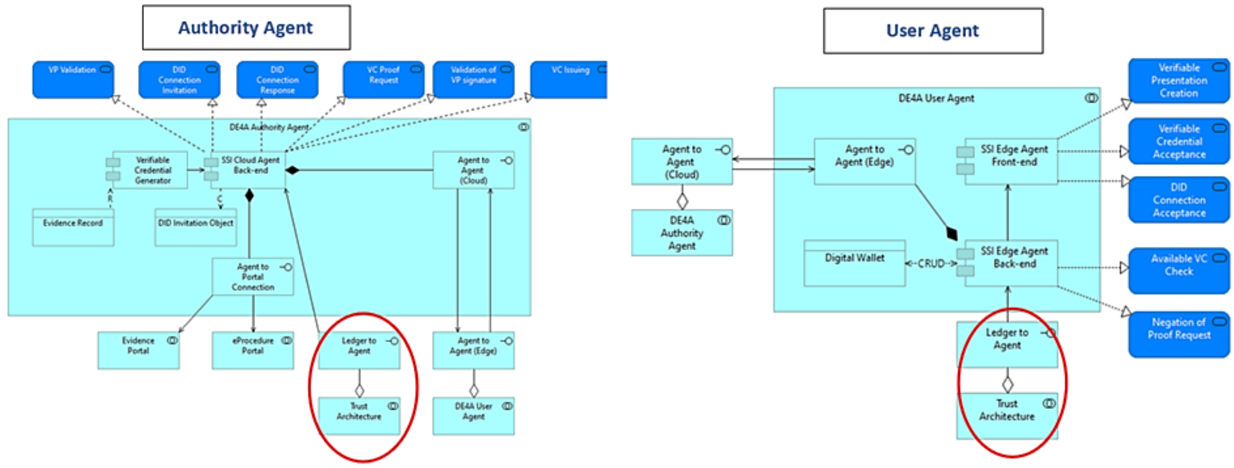


Figure 4: The VC pattern's SSI agents from the Project Start Architecture perspective

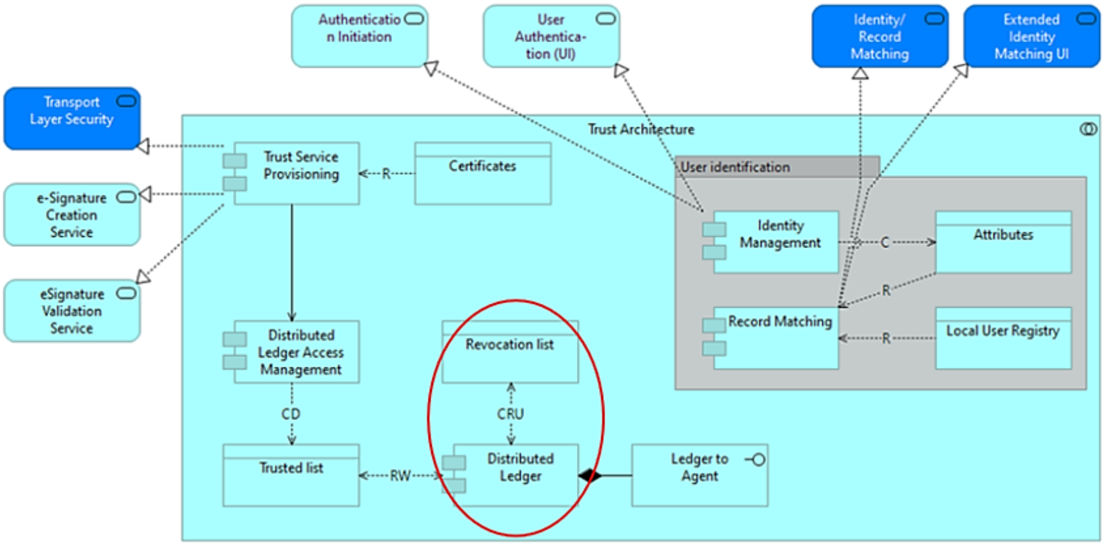


Figure 5: The connection to the Distributed ledger within the Trust Architecture

At the time of defining the requirements of the diplomas use case in the Studying abroad pilot and designing the VC pattern, the European Blockchain Service Infrastructure (EBSI) was in the process of building their own use cases, whereby two are worth mentioning due to the connection with DE4A i.e., Diploma use case and the European Self-Sovereign Identity Framework (ESSIF). Since DE4A use case is the Diploma recognition, it was clear that it was possible to align DE4A approach and objectives for this use case with EBSI use cases. Nevertheless, while designing the VC pattern, the EBSI use cases were not yet fully defined, motivating continuation of DE4A own research, design and developments. Following the early started collaboration between DE4A and EBSI, the DE4A project was selected as one of the projects in **EBSI's 1st wave Early Adopters programme**.

The DE4A-defined VC pattern had some requirements due to the nature of the project, whereby the most important one was the need to use existing legal and technical frameworks e.g., eID and eIDAS for authentication, which is also the reference framework for users' authentication in the Single Digital Gateway which gives overall context to the project. As such, the VC pattern (1) uses eID and eIDAS to facilitate the authentication process of students, as well as (2) uses the SSI approach to facilitate the user-centric, evidence management and exchange. Details of the VC pattern are elaborated in D2.3 [46].

After designing the VC pattern, the Minimum Viable Product for the first pilot iteration was defined, which considered all above-mentioned constraints. As such, the MVP v1 was built around the following main core facts/settings:

- Institutions (DC/DP) and students have to create their own decentralized identifiers (DIDs)
- Institutions (DP/DC) have to anchor their public DIDs on a ledger
- MVP has to support full DID exchange between students and DP/DC
- MVP has to use eIDAS for students' authentication and verification
- MVP has to support generation and digital signing of Diplomas in the form of VC by the DP using their public DID
- MVP has to support the exchange of VC/VPs
- Students have to use SSI edge agents (wallets)

- DP/DC have to use SSI authority/enterprise agents
- Students would communicate with the DP/DC only with one DID

Considering the design of the VC pattern and the MVP v1, a screening for potential technical solutions was carried out, which included various wallets (e.g., UPort) and technical frameworks (e.g., Hyperledger Aries). Considering that no solution was perfectly fitting, it was assessed that using framework Hyperledger Aries would provide the most adequate fit, since it supported generation of DIDs, DID exchange or DID-comm, generation of VC, signing of VCs using DIDs, exchange of VC/VPs etc. Furthermore, this framework was fully compatible with the W3C and DIF specifications for DIDs and VCs.

Nevertheless, some MVP v1 requirements were not satisfied only with the choice of Hyperledger Aries framework. The most important missing points were: (1) no SSI edge agent existed at that time, (2) no SSI enterprise agent existed at that time, (3) the usage of eIDAS for authentication was not possible, (4) for a use case involving public administrations, there was no standardized DID method, and (5) there was no appropriate blockchain ledger. All these challenges were conveniently addressed within the R&D activities of DE4A.

During the implementation of the VC pattern in DE4A, EBSI also continued to build on their ESSIF and Diploma use cases. A bidirectional communication channel was established with the EBSI core, ESSIF and EBSI Diploma teams, where information and design planning were exchanged in several internal formal meetings. At that time, DE4A tried to adapt the VC pattern and MVP v1 planning and align its design with the EBSI as much as possible. However, this was not possible due to different timelines, requirements and objectives of each of the projects. In order to continue with the pilot planning, DE4A had to define the MVP for the VC pattern in July 2020, even though EBSI/ESSIF compliance was not defined due to the unavailability of stable technical specifications. After designing the VC pattern, its architecture outline and choosing the core components for the SSI agents, the following differences were identified between DE4A and EBSI/ESSIF:

EBSI/ESSIF's objective was the introduction of Verifiable IDs for natural persons, which was not possible for DE4A, due to the requirement that piloting students identify to the DP/DCs using eID and eIDAS. Therefore, DE4A had to introduce the eIDAS Minimum Dataset (MDS) within the Diploma VC to guarantee the requirement. Furthermore, EBSI/ESSIF primarily focused on the OpenID Connect for authentication, while DE4A requirement was the usage of DID exchange and DID-comm as the more innovative and novel approach, which had to be tested and evaluated. EBSI/ESSIF's plan was also to introduce the e-SEALing of VCs, which however was not technically possible (at that time) due to the JSON formats of VCs, thus DE4A chose to sign Diploma VCs with DC/DP's public DIDs. The differences are depicted in Figure 6.

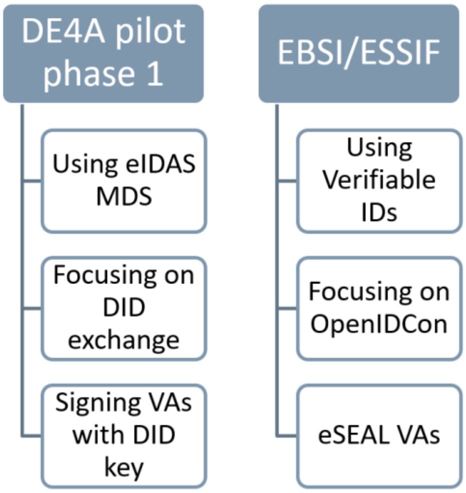


Figure 6: DE4A and EBSI/ESSIF phases comparative

Despite such differences, the DE4A's design and planning was adapted to EBSI/ESSIF as much as possible. The following adaptations were undertaken in this regard:

- DE4A Institutions (DC/DP) generate their own DIDs, which have to be based on the **DID:ebsi** method
- DE4A Institutions (DP/DC) have to anchor their public DID:ebsi on the **EBSI blockchain ledger**, precisely on the **DID registry**
- DE4A Data Provider (DP) Institutions have to be added to the **EBSI's Trusted Issuer Registry** (TIR)
- DE4A Data Consumers (DC) Institutions have to **verify** the Diploma VPs and their issuers using the EBSI DID registry and TIR
- The Diploma VCs (verifiable attestations – VAs) that the DE4A enterprise agents generate must be compliant with the EBSI VA schema (which in turn is compliant with the EDCI – Europass learning model [47])
- The Diploma VAs have to be signed using the DE4A DP's public DID:ebsi

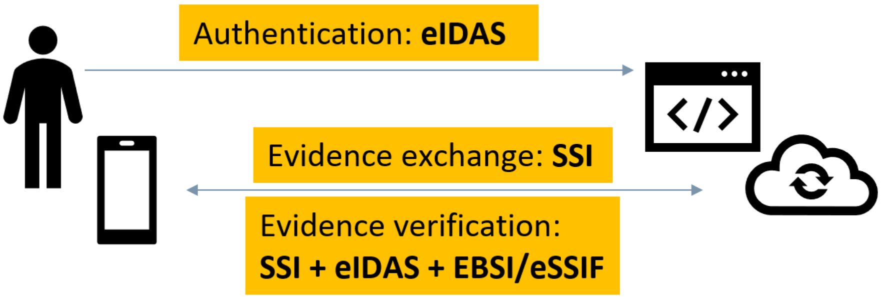

Authentication: **eIDAS**
Evidence exchange: **SSI**
Evidence verification:
SSI + eIDAS + EBSI/eSSIF

Figure 7: Use case high level description

Based on the adaptations of the DE4A VC pattern MVP v1, some technical changes had to be ensured.

- Once the use of EBSI's blockchain ledger (e.g., DID registry, TIR) was confirmed as technically feasible and within the timeframe of the project, there was no need to build a separate private/consortium blockchain network for DE4A or even use Hyperledger Indy, hence these options were discarded, and the developed SSI edge and enterprise agents used the EBSI ledger. Even though DE4A specific adaptation enables alignment with EBSI, it still renders the DE4A as

Blockchain ledger agnostic, meaning that DE4A is using currently EBSI as a trust anchor and ledger but could easily replace it with HL Indy or any other ledger.

- The EBSI VA example and schema was used when generating Diploma VAs. The EBSI VA example and schema are consequently based on the EDCI Europass model. However, given the requirement of the DE4A project to use eID and eIDAS for authentication purposes, it became necessary to adapt the EBSI VA slightly, adding eIDAS Minimum Data Set attributes to the subject parameter of the VA, discarding using the EBSI DID for students.

All the identified differences and adaptations of DE4A were regularly communicated with the EBSI/ESSIF team and as such also confirmed and recognised. The meetings are continuing internally even at the time of this document's writing, where the planning for the DE4A second iteration piloting is taking place. Following conclusions are being taken in this round:

- DE4A DP Institutions will revoke issued Diploma by using the **EBSI revocation list** (when accessible and enabled)
- DE4A will use the EBSI **Trusted Schema Registry** (TSR) and its EBSI VA schema and context (when accessible and enabled)

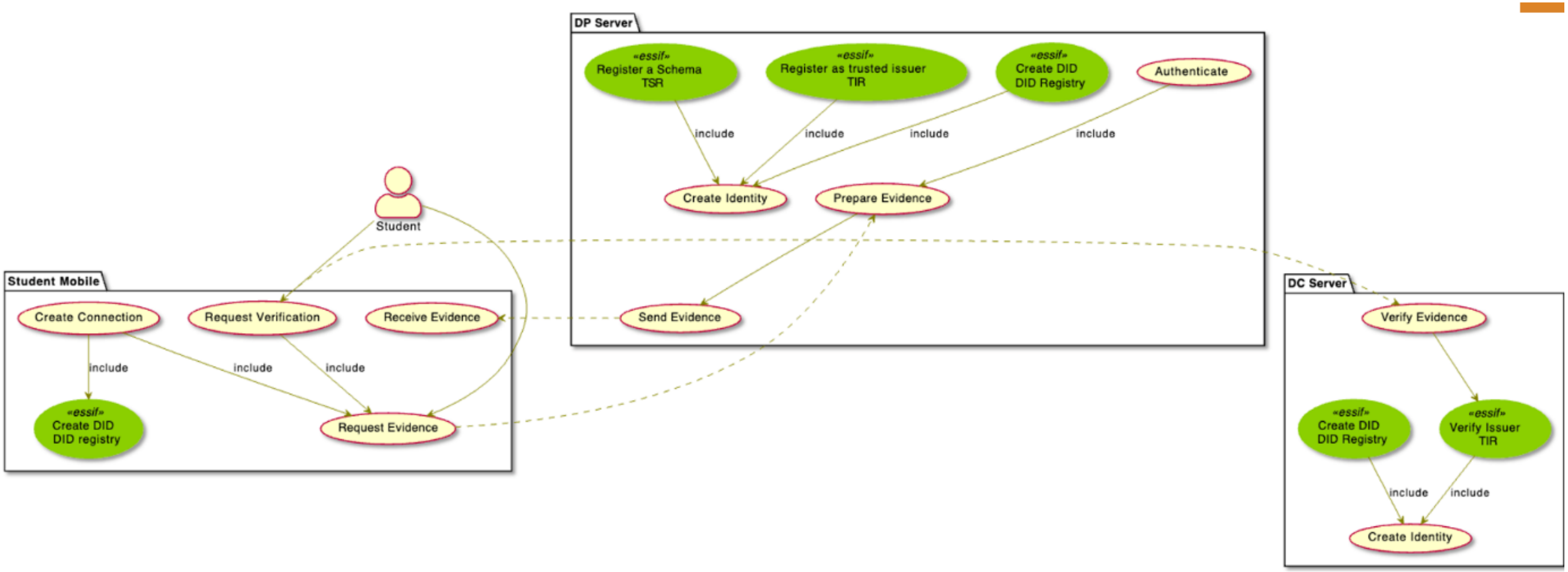


Figure 8: DE4A's use cases

4 DE4A Self-Sovereign Identity Solution

The environment where the users (university students) will validate in real-life scenarios the implemented components and integrated services needed for the DE4A Self- Sovereign Identity approach is, according to the DE4A contract with the European Commission, the "**Diplomas Recognition" use case of the Studying Abroad Pilot**.

The general objective of this pilot is to facilitate the virtual or physical mobility needs of Higher Education students in the European Higher Education Area, based on electronic procedures that enable the application of the 'Once-Only' principle and Digital-by-Default. The pilot aims to enhance cross-border automated exchange of evidence among national digital infrastructures of Higher Educational institutions.

The service provided by DE4A allows students to be identified using their national eIDs (using eIDAS) to access foreign higher education digital portals and request existing evidences related to them and required by the procedure in the portal, such as their higher education diplomas, and receive them electronically from a trusted source in their home country.

Under the scope of this pilot running activities, enabling novel technologies, such as blockchain and mobile digital wallets, and concepts, such as verifiable credentials, self-sovereign identities and distributed ledgers will be validated.

Each of the three use cases of the pilot focuses on different procedures corresponding to Annex II of the SDGR procedures, being "Application to Public Higher Education" (UC#1), "Applying for Study Grant" (UC#2), and "Diploma/Certs/Studies/Professional Recognition" (UC#3). Those three use cases are being validated by students of the participating Member States (Slovenia, Portugal and Spain). It is under the scope of this third case, "Diploma recognition", where DE4A includes, implements and validates the capabilities of cutting edge blockchain technologies, as it focuses on the cross-border procedure of presentation of academic studies certifications to simplify the further use of such information by competent education authorities. As the other two use cases of the Studying Abroad Pilot are related to an application procedure, DE4A considered the "Diploma recognition" use case as the most appropriate environment to validate all the benefits of the principles of identity self-sovereignty, distributed ledger technology and related building blocks and infrastructures (e.g., the EBSI infrastructure). This use case will leverage the existence of three roles in the VC pattern (Issuer, Holder and Verifier), in order to provide the student/user with the holder role and to enable him/her to request only once the Verifiable Credential but to present it many times when required.

4.1 Components

Figure 9 depicts in a single view all the different components involved in the Self-Sovereign Identity solution implemented in DE4A. As it can be seen, the main components developed by the Consortium (in green) are the Mobile Application or Mobile Edge Agent (to be used by the students) and the Authority Agent (to be installed/integrated in the server side). Students use the mobile app to store their diplomas (in the form of a Verifiable Credentials) issued from a competent authority (Data Provider or Issuer) in the country where they completed corresponding Higher Education studies and provide such evidence for recognition in another country to the Data Consumer authority (Verifier). The Authority Agent supports the interaction between the student (diploma Holder) and the diploma Issuer or Verifier and overall communication flow in the Verifiable Credentials pattern and the Mobile Edge Agent. Both Data Provider and Data Consumer leverage the Authority Agent.

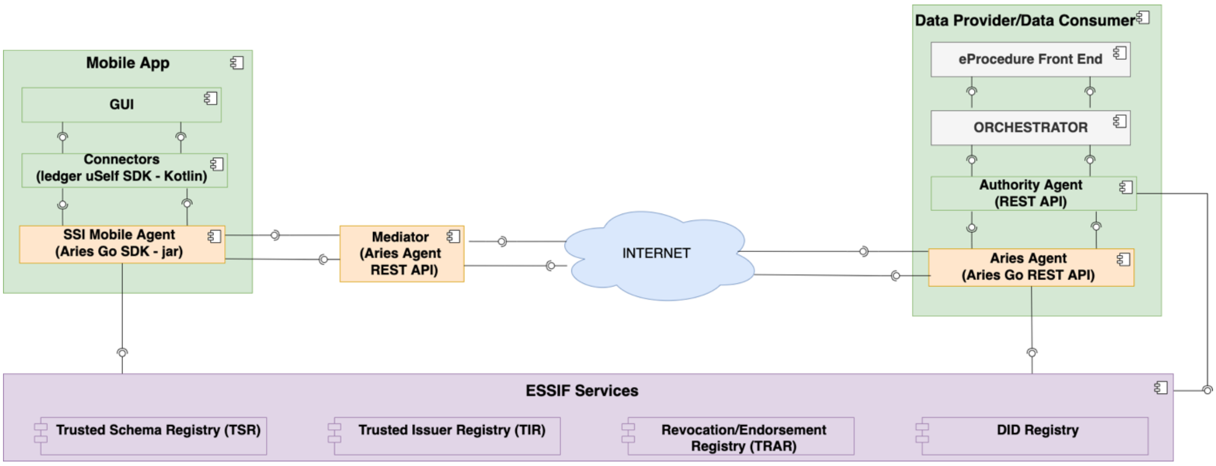


Figure 9: DE4A SSI components diagram

The following sections describe the different details of both key components for the SSI solution developed by this task.

4.1.1 Authority agent

The Authority Agent is an enterprise-level solution developed to support the interaction between the student (diploma Holder) and the diploma Issuer or Verifier and overall communication flow in the VC pattern. It enables students to follow the VC pattern flow and obtain or submit their diplomas as Verifiable Credentials (VCs) through the interaction with HL Aries agent deployed on the Issuer/Verifier side and the Edge Agent running in the student's mobile wallet. The Authority Agent is implemented as a high-level API, which provides a set of methods, which are to be called by the underlying Evidence (Data Provider) and eProcedure (Data Consumer) Portals. Specifically, the Authority Agent provides the following functionalities:

1. Generating an EBSI-compliant DID for the diploma Issuer;
2. Establishing a DID connection between the student (i.e., Edge Agent) and the Issuer/Verifier;
3. Issuing a digitally signed student's diploma in the form of VC by the Issuer;
4. Receiving and validating the submitted student's diploma in the form of VP by the Verifier.

Throughout the entire VC pattern flow, the Authority Agent facilitates the necessary communication with relevant EBSI ledgers for storing the information about the trusted diploma issuers (EBSI TIR) and their EBSI-compliant DIDs used for digitally signing the issued VCs and validating the issuers of submitted VPs (EBSI DID and TIR registries, respectively).

Generating an EBSI-compliant DID for the diploma Issuer

To digitally sign verifiable credentials for the users, which can later be validated by the Verifier, the DID used for signing must be trustworthy and publicly available. To achieve this, the information about organizations listed as trusted diploma issuers and their DIDs is anchored to EBSI ledgers, where it can be accessed by calling the DID and Trusted Issuers Registry REST APIs. This information is produced on the Authority Agent start-up (initialisation) by the underlying EBSI Connector component, which makes sure that the necessary keys are generated and imported to the cloud HL Aries agent, so that they can be used to sign the verifiable credentials. During the VP validation, the Authority Agent is then able to retrieve and resolve the DID information from the EBSI ledgers to validate the diploma issuer.

Establishing a DID connection between agents

The first step necessary in the diploma issuance/submission flow is to establish a secure connection between the Evidence Portal/eProcedure Portal and the user's Edge Agent (i.e. digital wallet). This is done by generating a QR code to be displayed on the portal side, which includes information about the DID invitation generated by the Authority Agent. The users can use their mobile application to scan the QR code and accept the DID invitation. Once this is done, a DID connection is established between the two agents (specifically, between the two HL Aries agents in the background). This step is a pre-condition needed to uniquely identify the two agents that will exchange messages in the later flow.

Issuing a Verifiable Credential

Once deployed on the Issuer side, the Authority Agent supports the process of issuing a diploma in the form of a Verifiable Credential (VC) digitally signed with an EBSI-compliant DID of the Issuer. The VC information is retrieved from the received diploma evidence data in the canonical XML format. Sending the Verifiable Credential produced by the Authority Agent includes sending the VC offer for the user to preview the included data, followed by actually sending the Verifiable Credential once the offer is accepted.

Receiving and validating a Verifiable Presentation

When deployed on the Verifier side, the Authority Agent enables organizations to request a diploma in the form of Verifiable Presentations from the students. In that case, a student can directly submit his/her diploma from the mobile wallet to the eProcedure Portal. Once received, the Portal can validate several aspects of the diploma validity: schema, digital signature, issuer and subject.

Within the Authority Agent, the interaction with EBSI ledgers is handled by the integrated EBSI Connector built by using the Walt.ID library. The EBSI Connector component is launched automatically during the Authority Agent start-up (initialisation), and it supports the process of registering the Issuer/Verifier in the EBSI DID and TIR registries, which includes generating and anchoring an EBSI-compliant DID for the Issuer/Verifier (only during the first start-up). The Walt.ID library is also used later during the validation of submitted VPs, as it allows to check if the diploma issuer listed in the submitted VC is a trusted diploma issuer (i.e., if the Issuer is registered in the EBSI TIR Registry).

4.1.2 Mobile Edge Agent

On the other side of the communication, the SSI approach requires an agent on the edge, namely SSI edge agent. This works as standalone mobile device application that integrates and runs different components to implement SSI approach. During this final release, the mobile edge agent has been improved notably on:

- providing new functionalities, such as signing the presentation proofs,
- refining and enhancing the underlying services
- improving the GUI, to gain a satisfactory end user's experience, by simplifying the flows/iterations with the users but also optimising the general application look and feel.

The main goal of this mobile edge agent is to enable users (students) to manage their digital credentials (diplomas on the pilot scope) and to interact with the Issuer's/Verifier's portals.

To achieve these objectives, there are three main actions that can be performed in the mobile edge agent by the user:

DID exchange: This protocol is used to create a connection between the service provider and the user, based on DID (see 2.1). There are two roles, *requester* and *responder,* where the *requester* is who initiates the exchange generating an invitation message sent to the *responder*.

In DE4A, several interfaces are implemented, based on Hyperledger Aries, to deploy this protocol: create-invitation, receive-invitation, accept-invitation.

Obtain Diploma: This service allows the user to request a credential (diploma) from a High Education portal, which acting as issuer, will provide the student with a Verifiable Credential that can be stored in the mobile device. Once a connection is available, the user will request (*send-request*) for a credential and the portal will offer (*send-offer*) all diplomas available for that user. The user will accept (*accept-offer)* or decline (*decline-offer*) the choices.

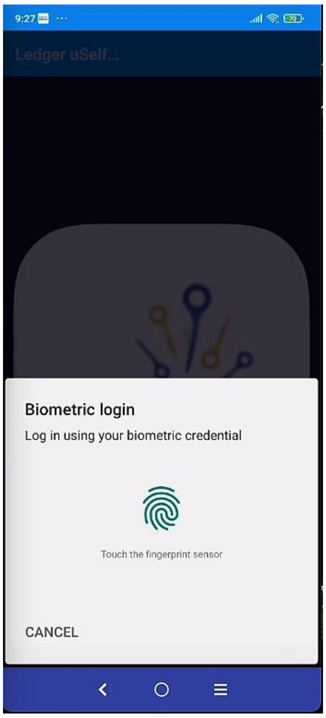
Send Diploma: With this service the students are able to present their diploma/certificates/studies to any European Higher Education portal where the procedure requires it. It is mandatory, obviously, to have first a Verifiable Credential available in the mobile agent to be presented.

When the *verifier* (data consumer) sends a request for presentation (*request-presentation*), the mobile agent will prompt the user to select any stored diploma and accept the presentation (*accept-request-presentation).*

The following sections describe the subsequent steps to be performed in order to obtain a Diploma from a data provider (e.g. university portal) and present it in a data consumer, under the scope of the use case “Diploma/Certs/Studies/Professional Recognition" (UC#3).

4.1.2.1 Accessing the main menu on the edge agent

The user should perform the first step by clicking on the icon of DE4A application that is installed in the mobile device. This action opens the user interface and prompts the student to login using her/his fingerprint on the device sensor to access to the main menu, as shown in Figure 10.



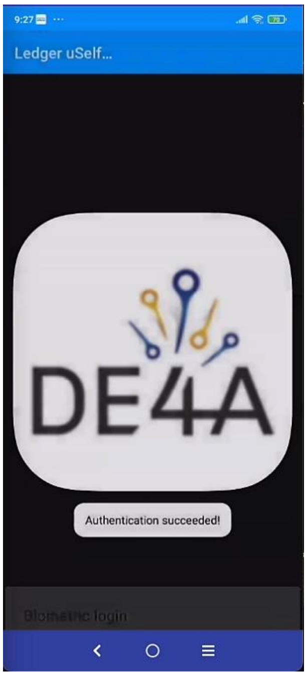
Biometric login

Log in using your biometric credential

Touch the fingerprint sensor

Figure 10: Fingerprint request screen

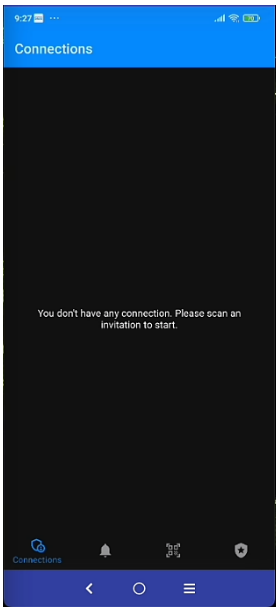
If this unlock process is successful, the user interface will welcome the user with the following splash screen, notifying the authentication was successful, as shown in Figure 10.



Authentication succeeded!

Figure 11: Welcome screen

The user interface flow continues to the main screen of the mobile application, where the connections available are displayed, as shown in Figure 12. The application is ready to be used by the student.



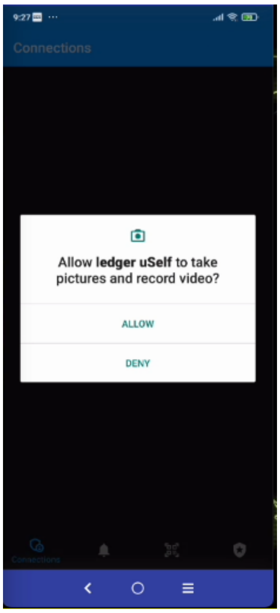
You don't have any connection. Please scan an invitation to start.

Figure 12: Connections available

4.1.2.2 Establishing a connection

In case there is no connections established, a message invites the user to scan an invitation to start.

Before continuing with the scanning process, the mobile OS will prompt a question to grant the Application access to the camera, to take pictures and record video. The user must give permission to the application for scanning the QR, as shown in Figure 13.



Connections

Allow **ledger uSelf** to take pictures and record video?

Figure 13: Allow application to access camera

Once the permission is granted, the user can click on the Scanner option on bottom menu, as shown in Figure 14.

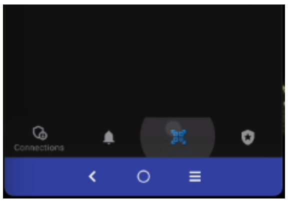


Figure 14: Scanner option button

After clicking on the "QR Scan" button, the mobile application will open the camera to scan the QR Code containing the Data Provider Invitation. This QR Code is displayed in the evidence portal web page once the proper identification is successfully performed by the student. An example for this QR is shown in Figure 15.

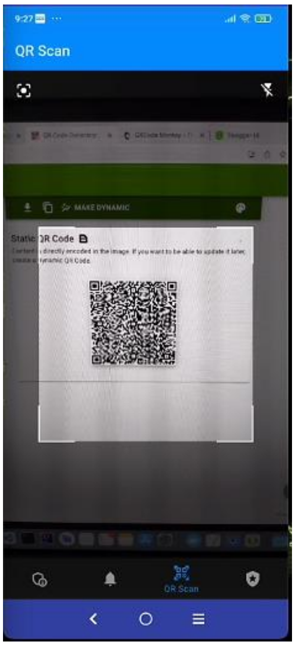
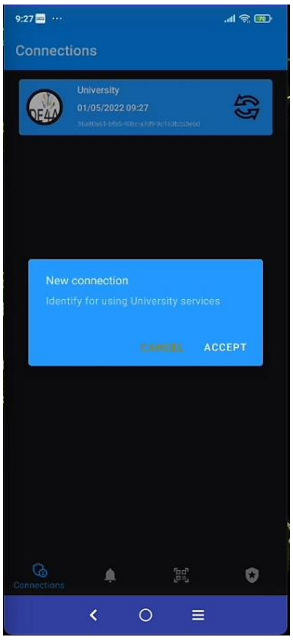


Figure 15: Scanning the QR Code on the Data Provider

Once the QR code is correctly read, the connection invitation will appear in the mobile interface, asking the user to accept the new connection with a pop-up message, as shown in following Figure 16:

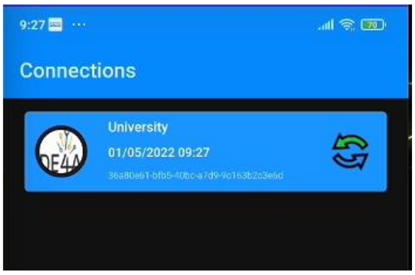


New connection

Identify for using University services

Figure 16: Connection invitation

If the user accepts the connection invitation, the interface will change the upper arrow to green, showing the connection is accepted on the mobile device. See Figure 17.



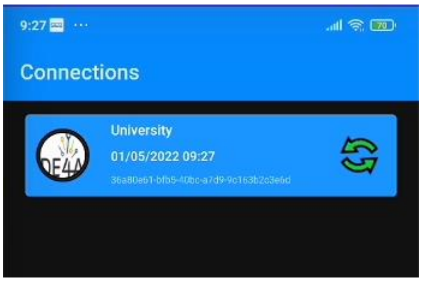
9:27

Connections

University
01/05/2022 09:27
36a80e61-bfb5-40bc-a7d9-9c163b2c3e6d

Figure 17: Connection accepted by user

Once the connection is accepted by the user, the Data Provider receives this acceptance and establishes the connection (generating a unique connection ID). This event is reflected on the mobile interface with the lower arrow becoming green, as shown in the Figure 18.



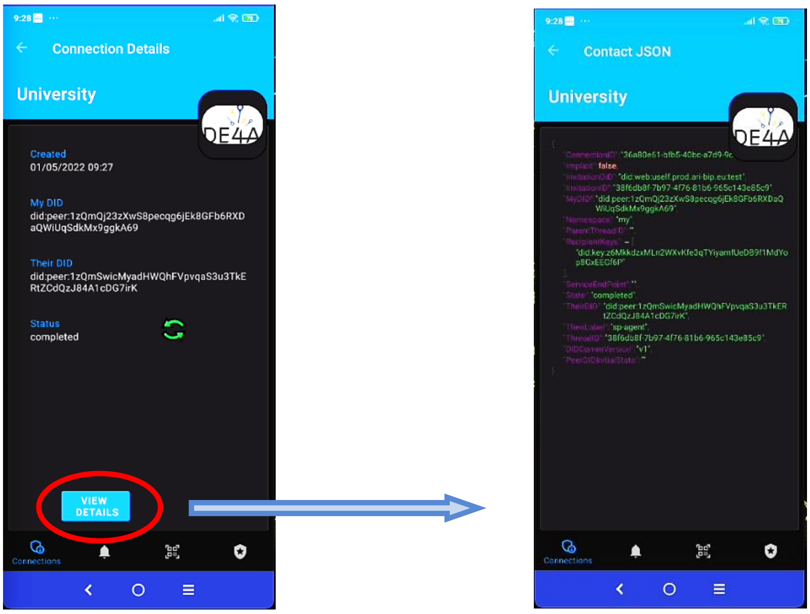
9:27

Connections

University
01/05/2022 09:27
36a80e61-bfb5-40bc-a7d9-9c163b2c3e6d

Figure 18: Connection established

If the user taps on the connection already established, the details now appear on the next screen. If the user needs to know additional details, she/he could press the button “View details” on the bottom of the screen.



9:28

Connection Details

University

Created
01/05/2022 09:27

My DID
did:peer:1zQmQj23zXwS8pecqg6jEk8GFb6RXDaQWiUqSdkMx9ggkA69

Their DID
did:peer:1zQmSwicMyadHWQhFVpvqaS3u3TkERtZCdQzJ84A1cDG7irK

Status
completed

9:28

Contact JSON

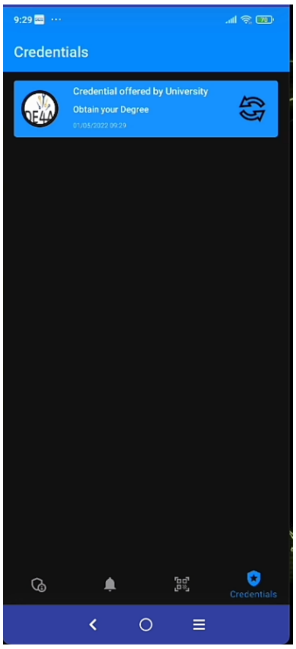
University

```
{
    "ConnectionID": "36a80e61-bfb5-40bc-a7d9-9c
    "Implicit": false,
    "InvitationDID": "did:web:uself.prod.ari-bip.eu:test",
    "InvitationID": "38f6db8f-7b97-4f76-81b6-965c143e85c9",
    "MyDID": "did:peer:1zQmQj23zXwS8pecqg6jEk8GFb6RXDaQ
              WiUqSdkMx9ggkA69",
    "Namespace": "my",
    "ParentThreadID": "",
    "RecipientKeys": – [
        "did:key:z6MkkdzxMLn2WXvKfe3qTYiyamfUeDB9f1MdYo
        p8GxEECf6P"
    ],
    "ServiceEndPoint": "",
    "State": "completed",
    "TheirDID": "did:peer:1zQmSwicMyadHWQhFVpvqaS3u3TkER
                 tZCdQzJ84A1cDG7irK",
    "TheirLabel": "sp-agent",
    "ThreadID": "38f6db8f-7b97-4f76-81b6-965c143e85c9",
    "DIDCommVersion": "v1",
    "PeerDIDInitialState": ""
}
```

Figure 19: Connection details and raw values

4.1.2.3 Obtaining a Diploma

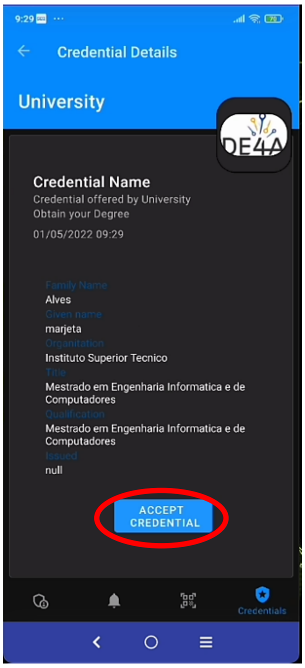
Once the invitation of the connection is accepted and established in the mobile app side, the evidence portal (Data provider) will send a Verifiable Credential offer to the mobile device.
It will generate a notification on the mobile, as it is shown in the following Figure 20.



Credential offered by University
Obtain your Degree
01/05/2022 09:29

Figure 20: Credential to be accepted

If the user taps on this credential information banner to accept the offer, a new screen with the information of the credential that represents her/his diploma will appear and the button "Accept Credential" is enabled, as shown in Figure 21. The user accepts the credential offered by pressing this button after checking the information.



Credential Details

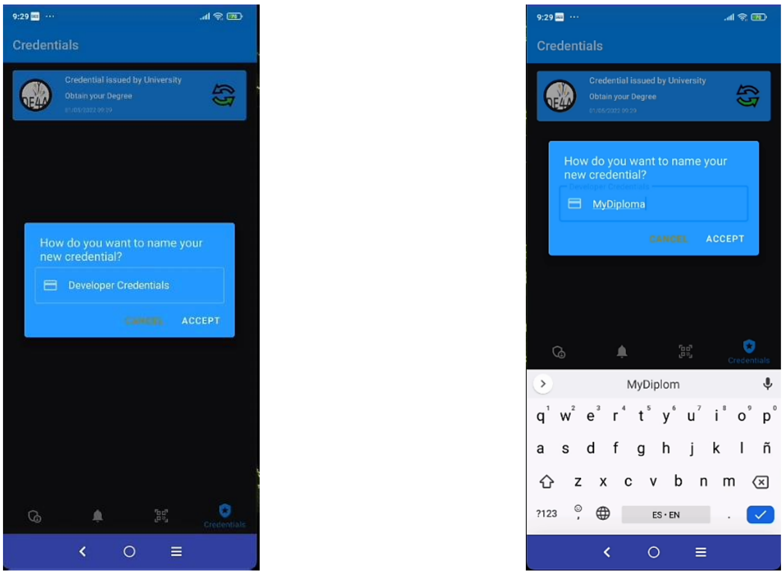
University

Credential Name
Credential offered by University
Obtain your Degree
01/05/2022 09:29

Family Name
Alves
Given name
marjeta
Organization
Instituto Superior Tecnico
Title
Mestrado em Engenharia Informatica e de Computadores
Qualification
Mestrado em Engenharia Informatica e de Computadores
Issued
null

Figure 21: Credential details

Once it is accepted, automatically, the user interface prompts the student to enter a name for this credential, as shown in Figure 22.



Credentials

Credential issued by University
Obtain your Degree
01/05/2022 09:29

How do you want to name your new credential?

Developer Credentials

Credentials

Credential issued by University
Obtain your Degree
01/05/2022 09:29

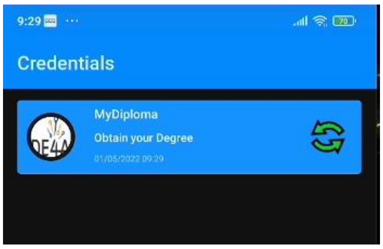
How do you want to name your new credential?

Developer Credentials

MyDiploma

Figure 22: Enter a name for the Verifiable Credential

Once the name is given, the user interface reflects the acceptance of the credential, displaying the two green arrows and the new name given by the user. The following Figure 23 displays the current situation:



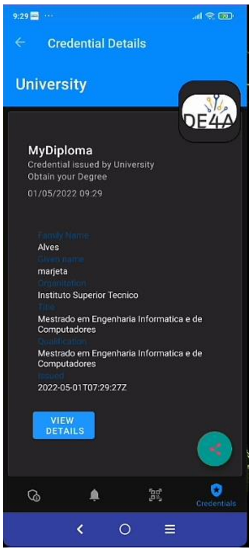
9:29

Credentials

MyDiploma
Obtain your Degree
01/05/2022 09:29

Figure 23: Credentials stored in the mobile device

It is in this Credentials menu where all the credentials from the user can be found and can be checked at any moment by tapping in each credential. The details screen will be like the following Figure 24:



9:29

Credential Details

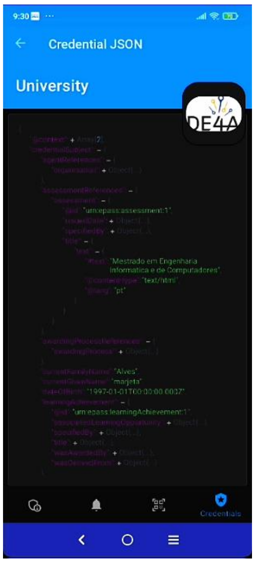
University

MyDiploma
Credential issued by University
Obtain your Degree
01/05/2022 09:29

Family Name
Alves
Given name
marjeta
Organisation
Instituto Superior Tecnico
Title
Mestrado em Engenharia Informatica e de Computadores
Qualification
Mestrado em Engenharia Informatica e de Computadores
Issued
2022-05-01T07:29:27Z

Figure 24: Credential´s information

The user could check on more details by clicking on the "View Details" link on the bottom right corner. An example of the details is shown in the following Figure 25:



9:30

Credential JSON

University

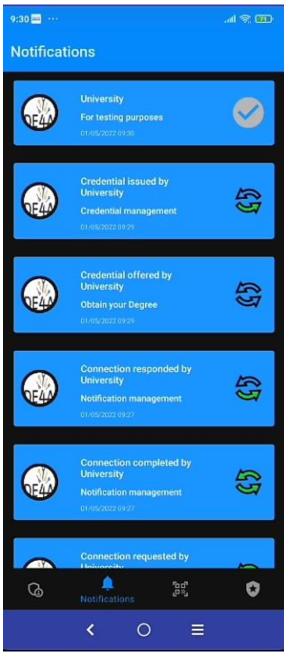
```
{
  "@context": + Array[2],
  "credentialSubject": - {
    "agentReferences": - {
      "organisation": + Object{...}
    },
    "assessmentReferences": - {
      "assessment": - {
        "@id": "urn:epass:assessment:1",
        "issuedDate": + Object{...},
        "specifiedBy": + Object{...},
        "title": - {
          "text": - {
            "#text": "Mestrado em Engenharia
                     Informatica e de Computadores",
            "@content-type": "text/html",
            "@lang": "pt"
          }
        }
      }
    },
    "awardingProcessReferences": - {
      "awardingProcess": + Object{...}
    },
    "currentFamilyName": "Alves",
    "currentGivenName": "marjeta",
    "dateOfBirth": "1997-01-01T00:00:00.000Z",
    "learningAchievement": - {
      "@id": "urn:epass:learningAchievement:1",
      "associatedLearningOpportunity": + Object{...},
      "specifiedBy": + Object{...},
      "title": + Object{...},
      "wasAwardedBy": + Object{...},
      "wasDerivedFrom": + Object{...}
    },
```

Figure 25: Additional information of the credential

4.1.2.4 Present a Diploma

In this section, it is explained how to present a credential when required in a data consumer portal using the DE4A mobile application. For this purpose, first a connection must be established with the data portal where the credential has to be presented. For this procedure, the steps are the same as explained in section 4.1.2.2 .

Once the portal of the data consumer requests a credential to be provided, the mobile device will receive a notification that can be checked in the **notification's** menu, as shown in Figure 26.



University
For testing purposes
01/05/2022 09:30

Credential issued by University
Credential management
01/05/2022 09:29

Credential offered by University
Obtain your Degree
01/05/2022 09:29

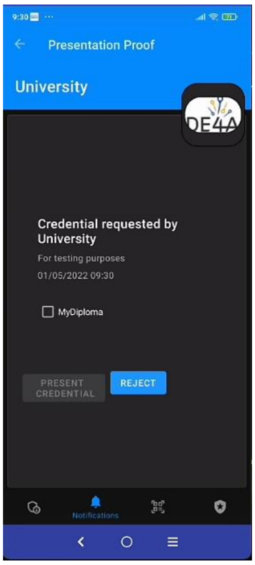
Connection responded by University
Notification management
01/05/2022 09:27

Connection completed by University
Notification management
01/05/2022 09:27

Connection requested by University

Figure 26: Notification of credential request

The user must tap on the credential request notification to check the request details and the requester information, as the Figure 27 shows:

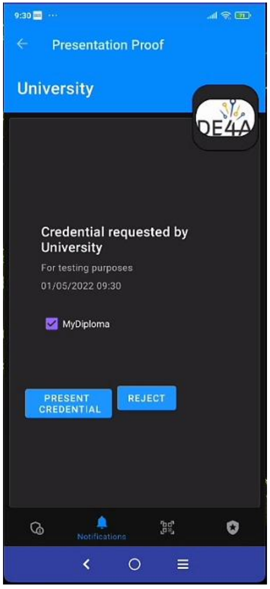


Credential requested by University
For testing purposes
01/05/2022 09:30

□ MyDiploma

Figure 27: Notification of credential request

To present any credential, the user has to select the credential to be presented from the ones available ones listed below the request date/time information. Once the credential is selected the "Present Credential" button will be enabled. In case the user prefers to cancel the request, she/he needs to tap on "Reject" button to abort the operation. See Figure 28:

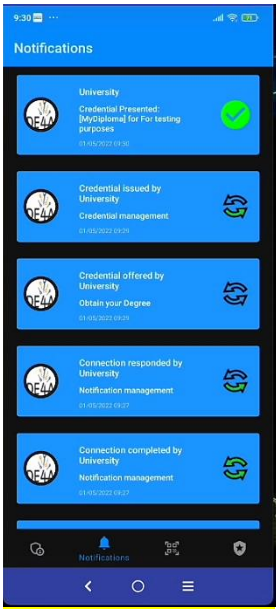


Credential requested by University
For testing purposes
01/05/2022 09:30

☑ MyDiploma

Figure 28: Credential ready to be presented

After presenting the credential, the user can verify the credential was received by Data Provider by checking the Notification screen, where a new notification with green icon is displayed, as shown in Figure 29



University
Credential Presented: [MyDiploma] for For testing purposes
01/05/2022 09:30

Credential issued by University
Credential management
01/05/2022 09:29

Credential offered by University
Obtain your Degree
01/05/2022 09:29

Connection responded by University
Notification management
01/05/2022 09:27

Connection completed by University
Notification management
01/05/2022 09:27

Figure 29: Credential Presentation sent

4.1.3 Portals (Evidence, Procedure)

As the final element of the DE4A Self-Sovereign Identity Supporting Framework, the Evidence and the eProcedure Portals facilitate the direct interaction between the student and the organization acting respectively as an Issuer of Verifier. Both portals follow the VC pattern flows and provide the previously described functionalities to end users.

The Evidence Portal is a software component deployed on the Issuer's side, which interacts with the Issuer's Authority Agent to:

1. Establish a DID connection between the Evidence Portal and student (i.e., the two HL Aries agents in the background).
2. Send an offer with the generated Verifiable Credential to the student.
3. In case of offer acceptance, the Evidence Portal sends the actual VC to the student (automatically or by explicitly requesting to send the VC).

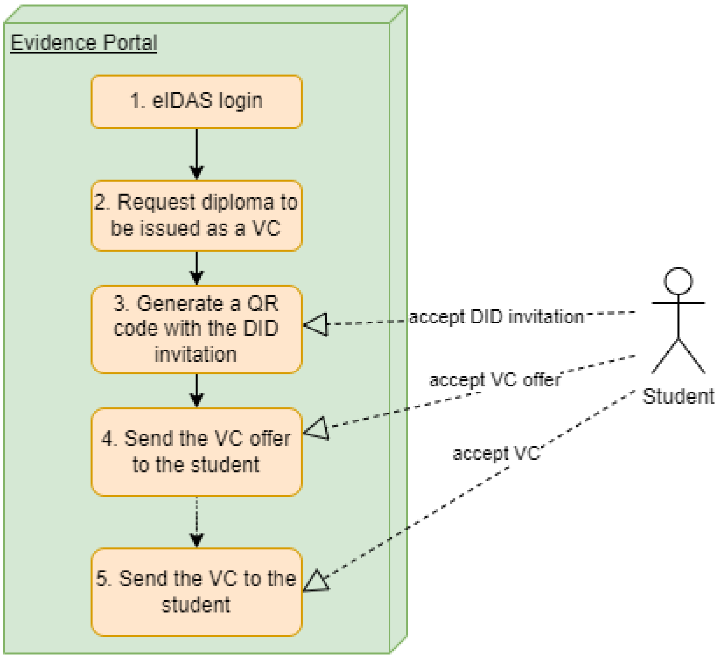

Evidence Portal
1. eIDAS login
2. Request diploma to be issued as a VC
3. Generate a QR code with the DID invitation
4. Send the VC offer to the student
5. Send the VC to the student
accept DID invitation
accept VC offer
accept VC
Student

Figure 30: Interaction flow between the student and the Evidence Portal

Figure 30 presents an overview of the interaction flow between the student and the Evidence Portal during the VC issuance process. Upon successful eIDAS login with his/her credentials, the student explicitly initiates the diploma-as-a-VC issuance process. The Evidence Portal calls the Authority Agent API in the background to generate a DID connection invitation based on which the two HL Aries agents (the government and the edge agent) will establish a secure DID connection. The result of this action is a generated invitation displayed as a QR code on the Evidence Portal, which the student scans with his/her mobile phone and accepts.

Once the DID connection is established, the student requests the portal to send him/her the VC offer with included VC generated and signed by the Issuer's Authority Agent. In this step, the Evidence Portal retrieves the diploma in the canonical format from the Issuer's internal data source (e.g. database) and generates the VC based on the retrieved data. If the student accepts the received VC offer, in Step 5, the Evidence Portal sends him/her the actual VC to be stored in the student's mobile wallet.

It should be noted that this step can be done automatically or by having the student explicitly requesting the Portal to send him/her the VC via a button. In the first case, the Authority Agent enables to automatically detect that the student accepted the offer by analysing the HL Aries logs. Through the web socket communication between the Authority Agent and the Evidence Portal, the Authority Agent notifies the Evidence Portal about the change in the VC offer status and the Portal performs the necessary GUI changes to reflect the recent changes. In that case, Step 5 becomes optional, optimising the user experience.

On the other side, the eProcedure Portal is a software component developed and deployed on the Verifier's side, which interacts with the Verifier's Authority Agent to:

1. Establish a DID connection between the eProcedure Portal and student (i.e., the two HL Aries agents in the background).
2. Send a request to the student to submit his/her diploma as a Verifiable Credential packaged into a Verifiable Presentation;
3. Validate the submitted diploma in terms of its schema, digital signature, issuer and subject (holder).

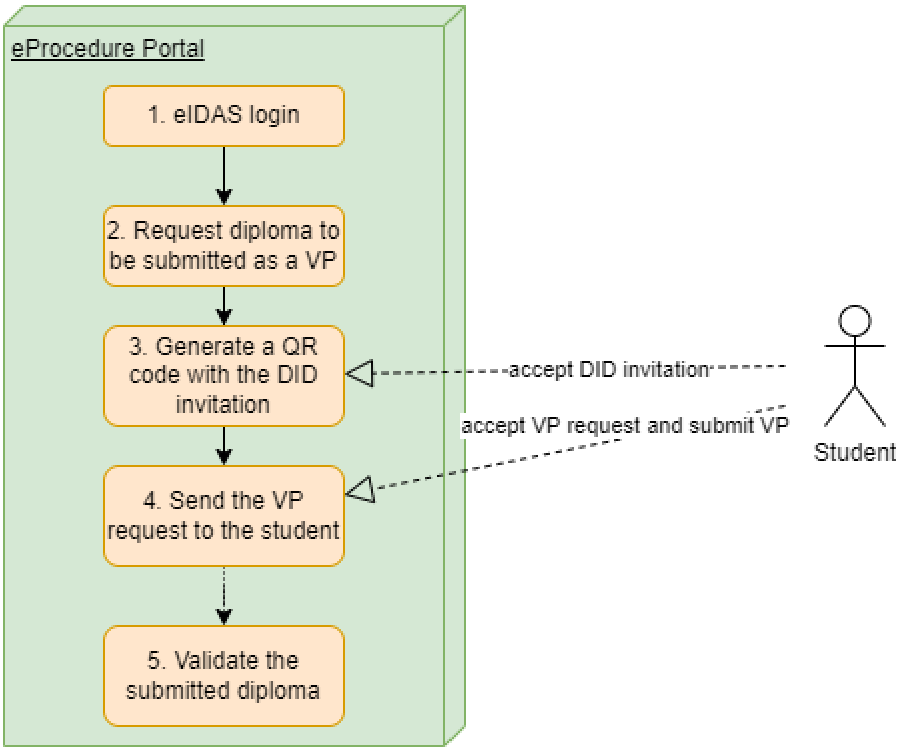

eProcedure Portal
1. eIDAS login
2. Request diploma to be submitted as a VP
3. Generate a QR code with the DID invitation
4. Send the VP request to the student
5. Validate the submitted diploma
accept DID invitation
accept VP request and submit VP
Student

Figure 31: Interaction flow between the student and the eProcedure Portal

The interaction flow between the student and the eProcedure Portal shown in Figure 31 resembles the flow with the Evidence Portal in its initial three steps. The student needs to login by using his/her eIDAS credentials, explicitly request to submit the diploma as a VP and scan and accept the generated QR code to establish the DID connection. In Step 4, the student requests the Portal to send him/her the VP request, which is generated and sent by interacting with the underlying Authority Agent. To accept the received VP request, the student selects the diploma to be submitted to the eProcedure Portal. Once the VP has been received and stored on the Verifier's side, the eProcedure Portal enables to validate the submitted diploma in terms of the four aspects mentioned above and display the validation results to the student.

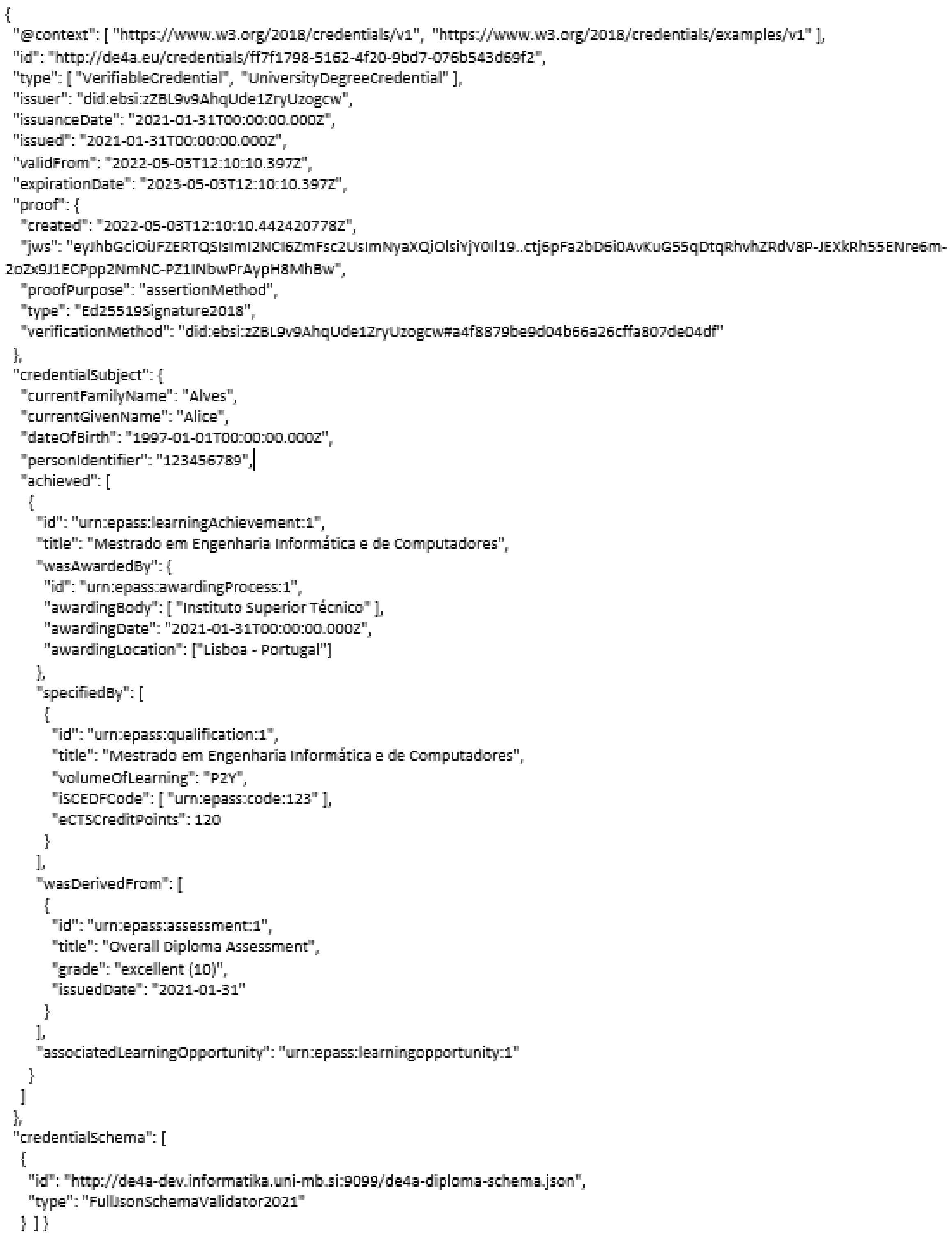
4.1.4 Verifiable Credentials (VC), Verifiable Presentations (VP)

This section describes the usage of Verifiable Credentials and Verifiable Presentations in the DE4A SSI solution. VC present the basic data model for DE4A Diploma, which also follows the guidelines from EBSI/ESSIF by conforming as much as possible to their Verifiable Attestation and Verifiable Diploma (based on Europass [47]) data models. But still, there are some differences:

- Property *credentialSchema* is an array, not an object; this is due to the HyperLedger Aries Go framework changing the property format during the signing procedure;
- Property *id* in *credentialSubject*, which is set to did:ebsi of the holder, is replaced with eIDAS Minimum Data Set, thus removing the need for Verifiable IDs defined by EBSI/ESSIF;
- Properties that are required by DE4A flow are set as required in JSON schemas.

Modified Verifiable Attestation can be found here: http://de4a-dev.informatika.uni-mb.si:9099/de4a-verifiable-attestation.json.

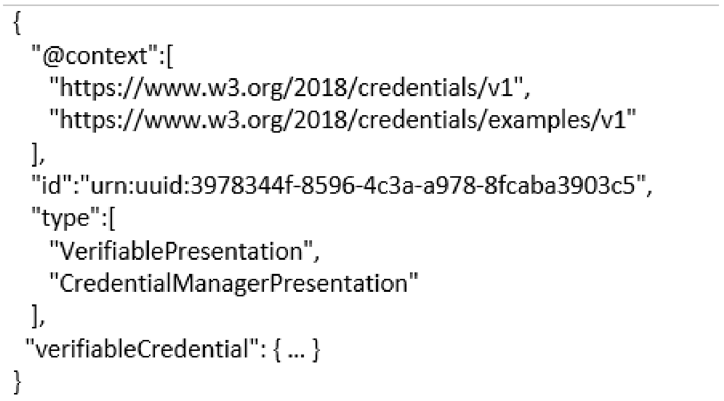
Modified Verifiable Diploma (Europass) can be found here: http://de4a-dev.informatika.uni-mb.si:9099/de4a-diploma-schema.json. An example DE4A diploma generated in the form of a VC by following the pre-defined DE4A diploma schema is presented in Figure 32.



```
{
 "@context": [ "https://www.w3.org/2018/credentials/v1",  "https://www.w3.org/2018/credentials/examples/v1" ],
 "id": "http://de4a.eu/credentials/ff7f1798-5162-4f20-9bd7-076b543d69f2",
 "type": [ "VerifiableCredential",  "UniversityDegreeCredential" ],
 "issuer": "did:ebsi:zZBL9v9AhqUde1ZryUzogcw",
 "issuanceDate": "2021-01-31T00:00:00.000Z",
 "issued": "2021-01-31T00:00:00.000Z",
 "validFrom": "2022-05-03T12:10:10.397Z",
 "expirationDate": "2023-05-03T12:10:10.397Z",
 "proof": {
  "created": "2022-05-03T12:10:10.442420778Z",
  "jws": "eyJhbGciOiJFZERTQSIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..ctj6pFa2bD6i0AvKuG55qDtqRhvhZRdV8P-JEXkRh55ENre6m-
2oZx9J1ECPpp2NmNC-PZ1INbwPrAypH8MhBw",
  "proofPurpose": "assertionMethod",
  "type": "Ed25519Signature2018",
  "verificationMethod": "did:ebsi:zZBL9v9AhqUde1ZryUzogcw#a4f8879be9d04b66a26cffa807de04df"
 },
 "credentialSubject": {
  "currentFamilyName": "Alves",
  "currentGivenName": "Alice",
  "dateOfBirth": "1997-01-01T00:00:00.000Z",
  "personIdentifier": "123456789",
  "achieved": [
   {
    "id": "urn:epass:learningAchievement:1",
    "title": "Mestrado em Engenharia Informática e de Computadores",
    "wasAwardedBy": {
     "id": "urn:epass:awardingProcess:1",
     "awardingBody": [ "Instituto Superior Técnico" ],
     "awardingDate": "2021-01-31T00:00:00.000Z",
     "awardingLocation": ["Lisboa - Portugal"]
    },
    "specifiedBy": [
     {
      "id": "urn:epass:qualification:1",
      "title": "Mestrado em Engenharia Informática e de Computadores",
      "volumeOfLearning": "P2Y",
      "iSCEDFCode": [ "urn:epass:code:123" ],
      "eCTSCreditPoints": 120
     }
    ],
    "wasDerivedFrom": [
     {
      "id": "urn:epass:assessment:1",
      "title": "Overall Diploma Assessment",
      "grade": "excellent (10)",
      "issuedDate": "2021-01-31"
     }
    ],
    "associatedLearningOpportunity": "urn:epass:learningopportunity:1"
   }
  ]
 },
 "credentialSchema": [
  {
   "id": "http://de4a-dev.informatika.uni-mb.si:9099/de4a-diploma-schema.json",
   "type": "FullJsonSchemaValidator2021"
  } ] }
```

Figure 32: An example of the DE4A VC diploma

Verifiable Presentations are used when transferring the diploma between holders and verifiers. VPs define the format of presentation of VC. VPs also enable selective disclosure, showing just a subset of properties in VC, but this feature requires a special kind of digital signature that EBSI/ESSIF does not yet define for their data models. An example of the DE4A Verifiable Presentation to be submitted by the student is shown in Figure 33. Besides the embedded VC with the contents like the VC presented in Figure 32, the VP includes the information about the context, generated VP identifier, and the type.



```
{
  "@context":[
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id":"urn:uuid:3978344f-8596-4c3a-a978-8fcaba3903c5",
  "type":[
    "VerifiablePresentation",
    "CredentialManagerPresentation"
  ],
 "verifiableCredential": { … }
}
```

Figure 33: An example of a DE4A Verifiable Presentation

VC data model also specifies context property, which is used for defining the JSON-LD applied to the VC. Contexts help with interoperability between different data models of VCs, providing meaning and reference to the formal definition of each VC's property. Because Europass [47] did not provide context for the diploma yet, DE4A's credential references only context for the core VC data model, which W3C released.

JSON-LD contexts for DE4A diploma can be found here: http://de4a-dev.informatika.uni-mb.si:9099/agent-startup-contexts.json.

4.2 Deployment

4.2.1 Authority agent

This section is focuses on presenting the steps and the guidelines for deploying the Authority Agent infrastructure on the Issuer's/Verifier's premises. The deployment of the Authority Agent includes the deployment of the following software components, where each is running in a separate Docker container:

- HL Aries Go REST server representing the government agent, which includes a set of API endpoints for interacting with another HL Aries agent (e.g., Edge Mobile Agent).
- CouchDB database server used by the REST API for storing information about DID connections, valid EBSI-compliant DID of the organization, as well as the status of VC and VP interactions with Edge Agents.
- Authority Agent REST API service, which includes a set of endpoints that implement the VC pattern functionality of the Authority Agent.

The complete installation guidelines are available at https://github.com/de4a-wp5/ssi-authority-agent. Besides the general system and database configurations, the following properties need to be specified to successfully run the Authority Agent REST API:

- Property *alias* of the organization's Authority Agent, which will be displayed to the user when sending invitations for connecting agents (e.g., MIZŠ Slovenia).
- Property *aries.enterprise.ip.address* is the URL to the HL Aries government agent to which the REST API will send HTTP requests.
- Property *signature.type* refers to the type of the digital signature used to sign verifiable credentials issued to the student. By default, the verifiable credentials will be signed by the Ed25519Signature2018 type.

- Property *bearer.token* refers to the token obtained from the EBSI onboarding service, which is used by the EBSI Connector in case of onboarding and registering the Authority Agent's DID into the EBSI DID Registry during its first start-up.
- Property *organization.img.url* is a URL to the public repository with the logo image of the Issuer/Verifier, which is displayed on the mobile phone during establishing the DID connection.

Once all configuration parameters are configured, it is possible to start the Docker containers by running the docker-compose.yml file. The CouchDB database GUI will be available at http://<IP ADDRESS:5984>/_utils.

Once the Authority Agent is started, the EBSI Connector first checks if the EBSI-compliant DID for that AA instance has already been generated and anchored to the EBSI DID Registry. If this is not the case, the EBSI Connector performs this process. After that, the AA REST API is ready to handle any incoming calls from the Evidence/eProcedure portal.

Specifically, the flow of API requests on the Issuer side is the following:

1. */generate-invitation* – method used to generate a DID connection invitation and send it to the student's Edge Agent. The method returns the information about the DIDComm invitation, which is displayed as a QR code on the Evidence Portal;
2. */did-conn-status/{userId}* – (optional) method used to check the current status of the DID connection for a particular student identified by the *userId* parameter;
3. */send-vc-offer* – method used to generate a diploma in the VC form based on source diploma data, digitally sign it with the Issuer's EBSI-compliant DID and send it for preview to the student's mobile wallet;
4. */check-offer-vc-response/{userId}* – (optional) method for checking the current status of the VC (offer) for a given student identified by the *userId* parameter;
5. */send-vc* – method used to send the approved VC with actual diploma data to the student.

The flow of API requests for the Verifier's side is the following:

1. */generate-invitation* – method used to generate a DID connection invitation and send it to the student's Edge Agent. The method returns the information about the DIDComm invitation, which is displayed as a QR code on the eProcedure Portal;
2. */did-conn-status/{userId}* – (optional) method used to check the current status of the DID connection for a particular student identified by the *userId* parameter;
3. */send-vp-request* – method used to generate and send a request for submitting a VP to the student;
4. */check-request-vp-response/{userId}* – (optional) method for checking the current status of the VP (request) for a given student identified by the *userId* parameter;
5. */validate-vp/{userId}* – method used to validate the VP submitted by the student (*userId)* based on its VC schema, digital signature, issuer and eIDAS minimum dataset information.

4.2.2 Mobile Edge Agent

This section is dedicated to the definition of the steps necessary for the use of the mobile application for students in order they can obtain and use their Diplomas. In order to do so, this section details on one hand the deployment of the communication necessary infrastructure and, on the other hand, how the student can install the application in their own mobile device.

4.2.2.1 Communication Infrastructure Deployment: Mediator

This infrastructure is required for providing a gateway between the different mobile apps (based on web-sockets communications) and the other Aries Agent (using https communication). It is worth to highlight that all the messages that pass through the Mediator are encrypted at the origin and only the extreme nodes of the communication, in DE4A case, the mobile app and the Data Producer/Data Consumer will be able to access the decrypted message data. The following diagram shows the interfaces exposed by the mediator and the different components necessary for its normal execution.

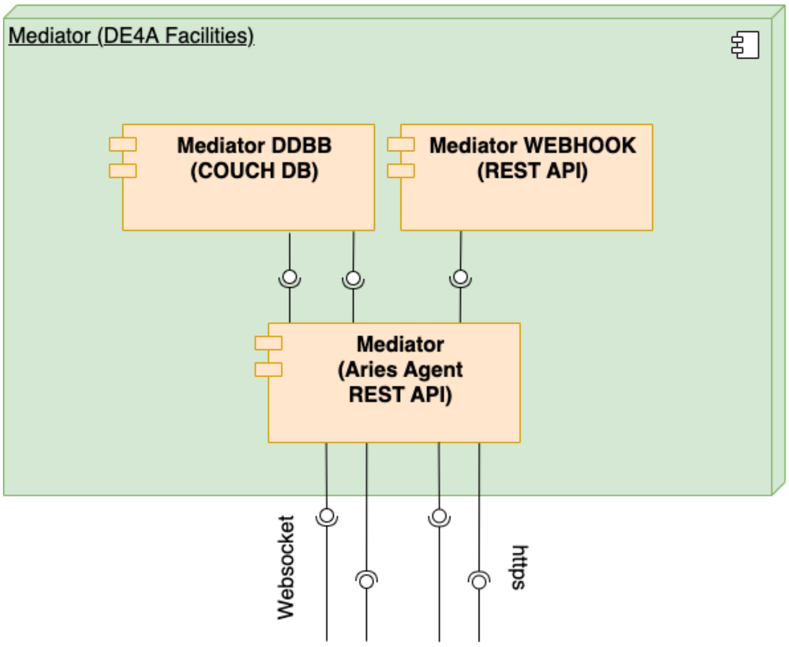

Mediator (DE4A Facilities)
Mediator DDBB (COUCH DB)
Mediator WEBHOOK (REST API)
Mediator (Aries Agent REST API)
Websocket
https

Figure 34: Mediator Deployment details

Regarding the different components necessary for the use of the mediator, they are the following:

- **Mediator Agent**: this a special instance of Aries Agent, configured ad-hoc to work as a mediator, exposing the corresponding ports for the web-socket communication and a different port for the https communication.
- **Mediator DDBB:** this component facilitates the storage of the different data necessary for the normal execution of the Mediator Agent.
- **Mediator Webhook:** enables logging of the operations executed by the Mediator Agent. As mentioned above all the messages that pass through the mediator agent are encrypted at origin and therefore the mediator doesn't have access to the cryptographic material necessary for de-encrypting the content of the data.

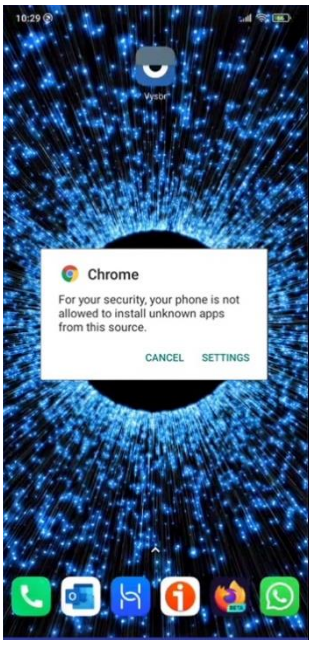
On the other hand, the mediator exposes two different endpoints:

- **Web-socket interface:** d facilitates the communication with the mobile devices.
- **Https interface:** allowing to other agents to communicate with the mobiles using a https protocol.

4.2.2.2 Installation of the Edge Agent

In this chapter, the steps needed to install the DE4A mobile application in an Android mobile device are explained. The mobile edge application can be installed using the following link: https://github.com/de4a-wp5/de4a-mobile-app.

Once the application is downloaded into the mobile device, the user needs to tap on the file to open it. The Android device will prompt the user for allowing apps from an unknow source to be installed.

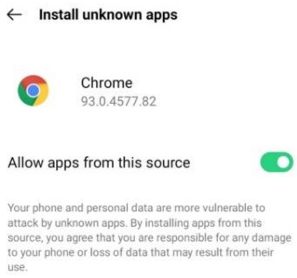


Chrome

For your security, your phone is not allowed to install unknown apps from this source.

Figure 35: Warning for unknown sources apps

The user click on “Settings” button and the settings menu will appear with the following option for allowing the apps installation from this source:



Install unknown apps

Chrome
93.0.4577.82

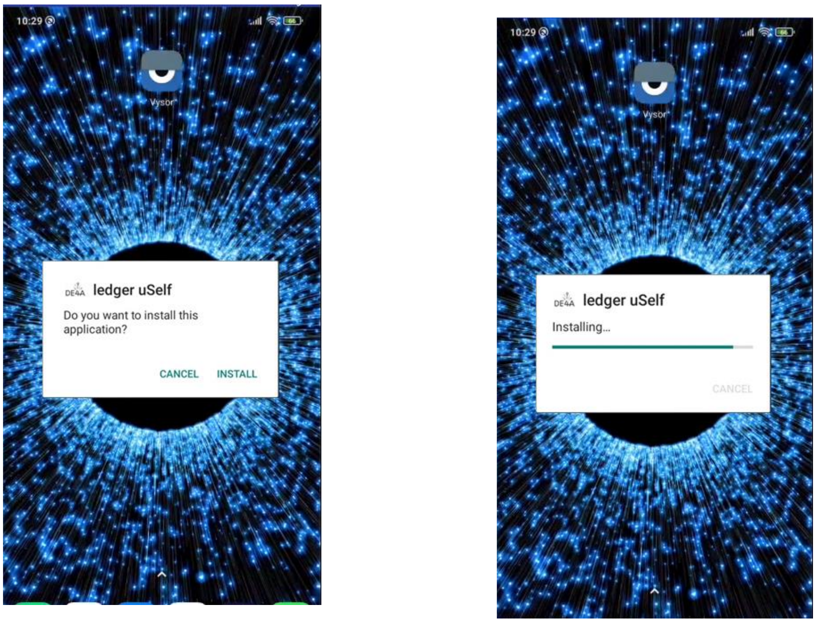
Allow apps from this source

Your phone and personal data are more vulnerable to attack by unknown apps. By installing apps from this source, you agree that you are responsible for any damage to your phone or loss of data that may result from their use.

Figure 36: Allow apps from this source option

For security reasons, it is highly recommended to turn off this setting after the installation

Once this option is enabled, the user is able to install the application properly.



ledger uSelf

Do you want to install this application?

ledger uSelf

Installing...

Figure 37: Application allowed to be installed

4.3 Message/Workflows

Before delving deeper into the workflows of communication between the agents/parties (in DE4A case DC/DP and the student), familiarity with the Hyperledger Aries Framework Go agent (Aries API) messaging system is helpful. The basic message (currently of version 2.0), which can be sent through the Aries API REST services, is defined by didcomm.org. The basic message consists of the message's id, the type of the message [34], an indication of the language used in the message, the time the message was created, and the content.

An example of such a basic message can be observed here:

```
{
   "id": "876543210",
   "type": "https://didcomm.org/basicmessage/2.0/message",
   "lang": "en",
   "created_time": 1547577731,
   "body": {
      "content": "This is the D5.8 Final Release of DE4A Self-Sovereign Identity Supporting Framework."
   }
}
```

The messaging system is not limited to using just basic messages. It can also make use of other types of messages, such as problem report messages [35] or acknowledgment (ACK) messages [36]. There are also multiple ways to transport and use messages in Aries API. One way is through the usage of the Aries API REST services (e.g., HTTP POST /issue credential/send-offer), where the mechanisms for sending (e.g., through the usage of my_did/their_did) and handling of the messages (e.g., built in storage arrays such as connections) are already in place. Note that the "handling of the message" means that the message has a REST API-specific body/content. Another way of using the message is by sending a custom message through the built-in REST API messaging system. In that case, one party first needs to register the service before a message can be delivered to the party.

Example of registering a service:

```
POST /message/register-service
{
  "name": "testingissuingcredential",
  "purpose": [
    "issuecredential"
  ],
  "type": "https://didcomm.org/basicmessage/2.0/"
}
```

The second party can now send a message to the registered service party.

For example:

```
POST /message/services
{
 "connection_ID": "Here_comes_the_connection_ID",
 "message_body": {
   "@type":"https://didcomm.org/basicmessage/2.0/",
    "~purpose":["issuecredential"],
    "message":"Testing if sending is successful.",
   "sender":"Student"
  }
}
```

When the parties are using the specific REST API's (e.g., /connections, /issue_credential, /present_proof etc.) or sending the messages directly (i.e., /messages) between each other, the messaging system is in the background also utilizing the webhooks [49]. Webhooks are user-defined callbacks made through HTTP POST requests (on the unique URL). When the message comes to the webhook, it is up to the receiver what it will do with it (e.g., trigger an event). Webhooks are usually used because they are faster than polling and require little work to handle.

The basics of a messaging system were explained for easier understanding of the DC/DP and the student communication with each other while executing UC#3 work/message flows. There are three work/message flows in UC#3:

- establishment of a secure connection between actors using DID Exchange Protocol 1.0 [28],
- issuance of a diploma as a Verifiable Credential using Issue Credential Protocol 2.0 [29] and
- verification of a diploma ownership as a Verifiable Presentation using Present Proof Protocol 2.0 [30].

During the further explanation of workflows, it will be observed that parties act in roles depending on the workflow requirements. Meaning, parties can act as a sender and/or receiver. When one of the parties begins the communication, it becomes the sender, and the other assumes the role of the receiver. Specifically, the first message flows occur between all actors for communication purposes between their agents. The second one happens between students and data providers, and the last one between students and data consumers.

The sub-sections of "Establishing a secure SSI-based connection between DC/DP and student", "Issuance of a VC-based Diploma" and "Diploma verification" details the different workflows applied in the development of the Self-Sovereign Identity solution.

Note: The template for the examples provided during the explanation of each of the next subsections comes in the following form:

Input: The input for the call that the party is making (if needed)

Execute: Aries API service call

Response: Aries API message response

4.3.1 Establishing a secure SSI-based connection between DC/DP and student

The connection between SSI-agents is established with the process shown in

Figure 38. In all three message flows, the eProcedure Front-End portal is connected to the DE4A authority server agent over the portal's backend. On the user side, students interact with the eProcedure portal with their browser while simultaneously using their mobile DE4A user agent for agent-to-agent communication with authority agents. The first message flow begins with authentication using eIDAS (step 0) and the user selecting on the eProcedure portal the desired service (step 1). Next, an invitation for DID communication is generated in QR form (by the SSI-authority agent) and shown to the user/student on the eProcedure website (steps 2, 3, 4). The user scans the QR code with his/her SSI-edge/mobile agent and accepts the invitation (steps 5, 6). Invitation response is sent agent-to-agent directly from the mobile user agent to the SSI-authority server agent (step 7). After establishing a DID connection, the eProcedure website is updated with connection status (steps 8, 9).

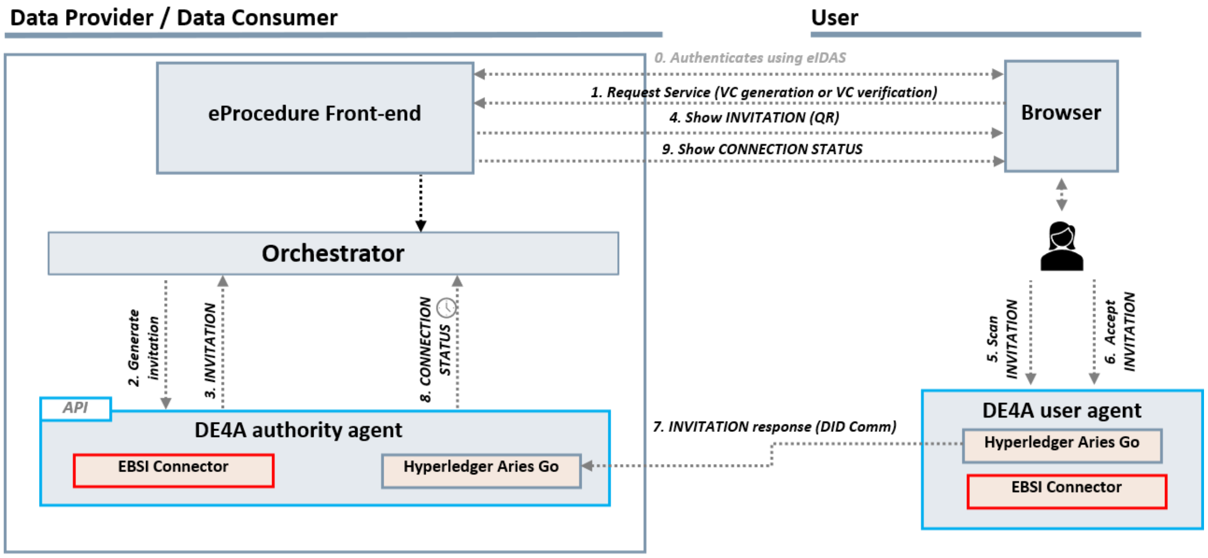


Figure 38: Process of DID connection establishment between Data Provider/Data Consumer and User

On the lower levels, the establishment of a secure connection between actors using DID Exchange Protocol is provided by the Aries API's, which is a straightforward procedure. It consists of the following five steps, where the DC/DP is the sender (IP_PARTY1) and the student acts as the receiver (IP_PARTY2):

1. Creating an invitation through the usage of HTTP POST /connections/create-invitation

Input: none
Execute: curl -X POST "http://IP_PARTY1:PORT/connections/create-invitation" -H "accept: application/json" -d ""
Response:
{"invitation":{"serviceEndpoint":"http://WEB-DOMAIN:PORT","recipientKeys":["did:key:DID-KEY-IS-HERE"],"@id":"ID-IS-HERE","label":"LABEL-NAME","@type":"https://didcomm.org/didexchange/1.0/invitation"},"alias":"","invitation_url":""}
Extracted response:
{"serviceEndpoint":" http://WEB-DOMAIN:PORT","recipientKeys":["did:key:DID-KEY-IS-HERE"],"@id":"ID-IS-HERE","label":"LABEL-NAME","@type":"https://didcomm.org/didexchange/1.0/invitation"}

2. Receiving an invitation by the usage of HTTP POST /connections/receive-invitation

Input: (extracted) JSON invitation from the previous step

Execute:

```
curl -X POST "http://IP_PARTY2:PORT/connections/receive-invitation" -H  "accept:
application/json" -H  "Content-Type: application/json" -d "{\"serviceEndpoint\":\"
http://WEB-DOMAIN:PORT\",\"recipientKeys\":[\"did:key: DID-KEY-IS-HERE\"],\"@id\":\"ID-
IS-HERE\",\"label\":\"LABEL-
NAME\",\"@type\":\"https://didcomm.org/didexchange/1.0/invitation\"}"
```

Response:

```
{"state":"","created_at":"0001-01-01T00:00:00Z","updated_at":"0001-01-
01T00:00:00Z","connection_id":"CONNECTION-ID-IS-HERE","request_id":"","my_did":""}
```

3. Accepting an invitation by the usage of HTTP POST /connections/{id}/accept-invitation

Input: id (connection ID)

Execute:

```
curl -X POST "http://IP_PARTY2:PORT/connections/CONNECTION-ID-COMES-HERE/accept-
invitation" -H  "accept: application/json" -d ""
```

Result:

```
{"created_at":"0001-01-01T00:00:00Z","updated_at":"0001-01-
01T00:00:00Z","connection_id":"CONNECTION-ID-IS-HERE"}
```

4. Checking the connections list through the HTTP GET /connections

Input: none

Execute:

```
curl -X GET "http://IP_PARTY1:PORT/connections" -H  "accept: application/json"
```

Response:

```
{"results":[{"ConnectionID":"CONNECTION-ID-IS-
HERE","State":"requested","ThreadID":"af62fe2f-d554-46dc-8700-
1d92813abab4","ParentThreadID":"","TheirLabel":"student","TheirDID":"did:peer:THEIR-DID-
IS-
HERE","MyDID":"","ServiceEndPoint":"","RecipientKeys":null,"RoutingKeys":null,"InvitationID
":"INVITATION-ID-IS-HERE","InvitationDID":"","Implicit":false,"Namespace":"their"}]}
```

5. Accepting the request by the usage of HTTP POST /connections/{id}/accept-request

Input: id (connection ID)

Execute:

```
curl -X POST "http://IP_PARTY1:PORT/connections/CONNECTION-ID-COMES-HERE/accept-
request" -H  "accept: application/json" -d ""
```

Response:

```
{"their_did":"","request_id":"","connection_id":"CONNECTION-ID-IS-
HERE","updated_at":"0001-01-01T00:00:00Z","created_at":"0001-01-
01T00:00:00Z","state":""}
```

One more Aries API step for finding the relevant data of established connection is important and necessary for more advanced actions (e.g., such as issuing credentials where the MyDID and TheirDID information is needed):

Input: None (but the connection ID is needed for the targeted search through the list of connections)
Execute:

```
curl -X GET "http://IP_PARTY1:PORT/connections" -H  "accept: application/json"
```

Response:

```
{"results":[{"ConnectionID":"CONNECTION-ID-IS-HERE","State":"completed","ThreadID":"af62fe2f-d554-46dc-8700-1d92813abab4","ParentThreadID":"","TheirLabel":"student","TheirDID":"did:peer:THEIR-DID-IS-HERE","MyDID":"did:peer:MY-DID-IS-HERE","ServiceEndPoint":"","RecipientKeys":null,"RoutingKeys":null,"InvitationID":"INVITATION-ID-IS-HERE","InvitationDID":"","Implicit":false,"Namespace":"their"}]}
```

By the usage of the Aries API DID Exchange Protocol 1.0 the following workflows were designed for the DE4A SSI-authority agent (AA) regarding the connections` handling:

1. **Start DID connection initialization** (Figure 39): The student begins the DID connection initialization by clicking the applicable button at the Frontend of the Evidence Portal (FEP) (eVŠ:ePortalFrontend). The FEP performs *initConnection()* on the Evidence Portal (BEP) (eVŠ:ePortalBackend). The BEP calls the *generateInvitation(userID)* on the AA (userID is needed to associate the given user/student with the connection that it is being initialized). The AA calls the Aries API by */connections/create-invitation*, which, in return, provides the invitation as a JSON file. At this point, the AA saves the data of the successful connection (userID, invitationID, invitationJSON, and the status of the invitation generated) to the Database, and returns the JSON file to the BEP. The BEP generates the QR code from the JSON file, and the FEP displays it to the student.

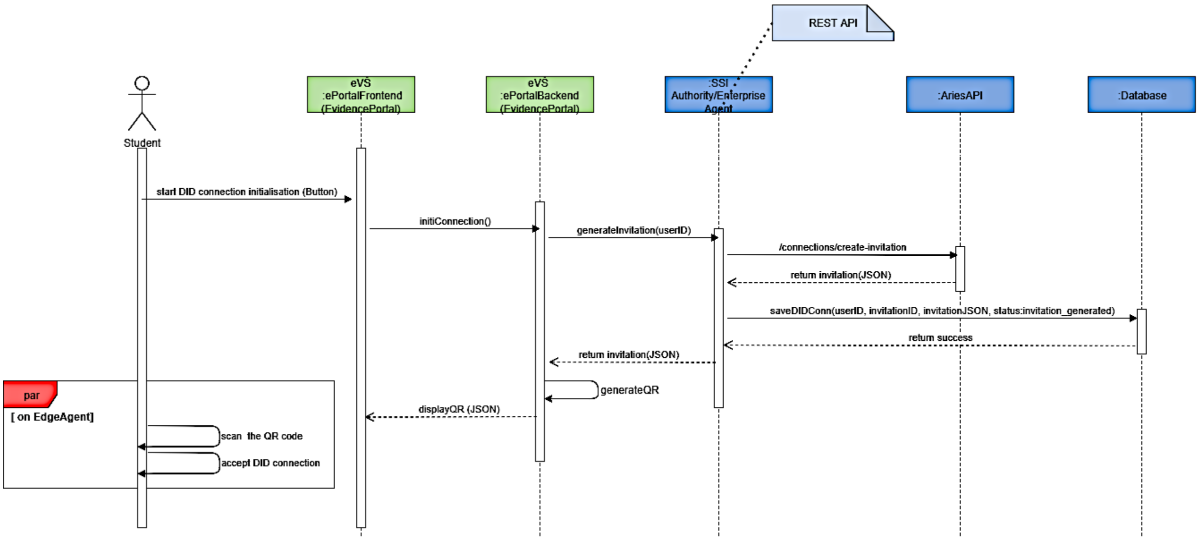

REST API
Student
eVŠ :ePortalFrontend (EvidencePortal)
eVŠ :ePortalBackend (EvidencePortal)
:SSI Authority/Enterprise Agent
:AriesAPI
:Database
start DID connection initialisation (Button)
initiConnection()
generateInvitation(userID)
/connections/create-invitation
return invitation(JSON)
saveDIDConn(userID, invitationID, invitationJSON, status:invitation_generated)
return success
return invitation(JSON)
generateQR
displayQR (JSON)
par
[on EdgeAgent]
scan the QR code
accept DID connection

Figure 39: Workflow of the DE4A SSI-authority agent for starting a DID connection initialisation

2. **Show connection status** (Figure 40): The student can check the connection status by clicking the respective button at the FEP. The FEP calls the *showConnectionStatus()* on the BEP. The BEP calls the REST service of *DIDConnStatus(userID)* on the AA, with the userID as a necessary

input argument to the service. The AA makes an internal call to the Database with the *getDIDConn(userID)*. The DIDConn value is returned, based on which multiple possible alternatives can be executed, which will result in different values being returned from AA to BEP:

- If no DID connection was saved in the Database (there is no DIDConn), the value -1 will be returned to the BEP. Value -1 signals that the invitation was never generated, and that the student should first create a DID connection.
- If there is a DIDConn and its status enum value is set to connection_established, the value 1 will be returned to the BEP. Value 1 signals that the connection has already been established, and that it can proceed to the VC pattern options.
- If there is a DIDConn and its status enum value is set to 'invitation_generated', a check must be made to see if the student has already accepted the invitation or not. Therefore, the AA first acquires a list of all the connections from the AriesAPI by calling the */connections* on AriesAPI, based on which there are two possible alternatives:
    - If no match is found (searched by the InvitationID) in the connections list, the connectionID is null, and the value of 0 is returned. 0 signals that the invitation was generated, but the student never responded (i.e., the Edge Agent must ask the student to respond).
    - If there is a match found in the connections list, the connectionID exists (the student responded to the invitation). The call is, therefore, made from the AA to the AriesAPI with the */connections/(connectionID)/accept-request.* The AA also updates/saves the connectionID of the DID connection in the Database. After both operations are complete, the AA returns the value of 1 to the BEP. Value 1 signals that the connection has already been established and that the student can proceed to the VC pattern options.

The BEP returns the connection status to the FEP.

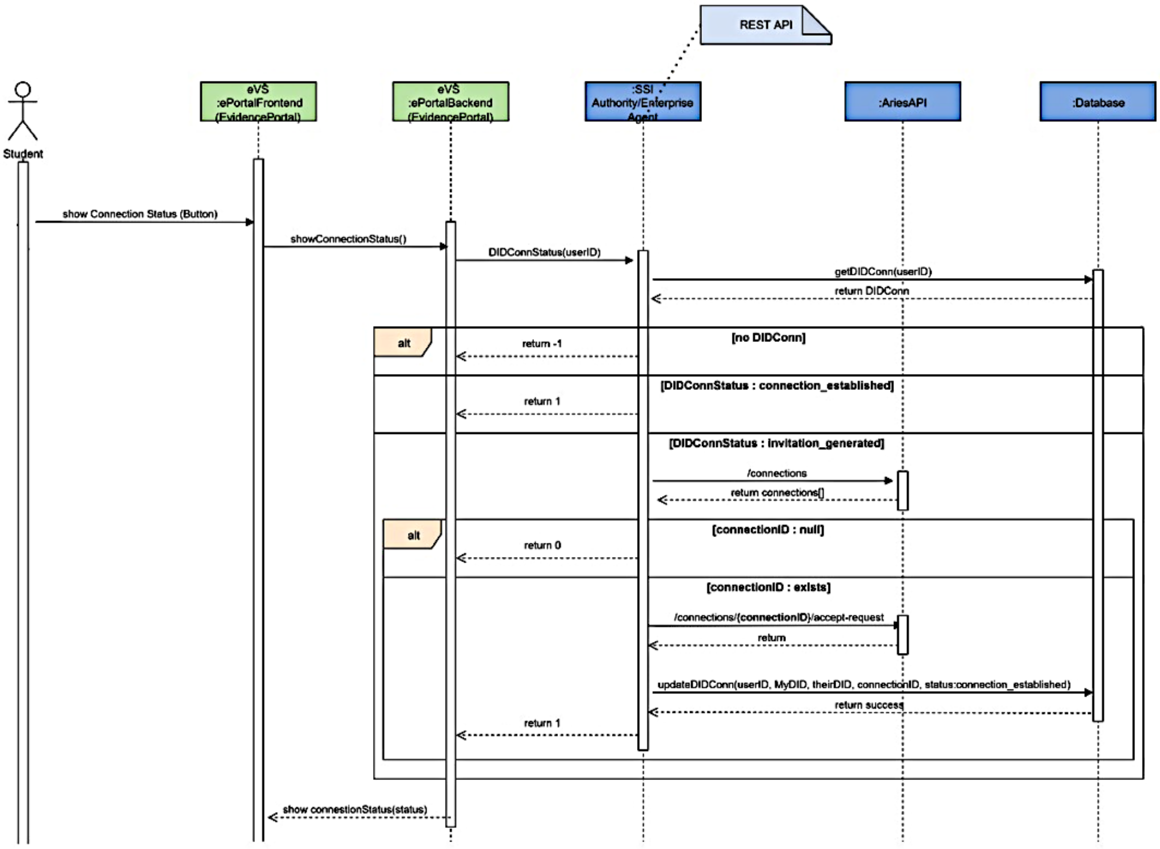

REST API
Student
eVS :ePortalFrontend (EvidencePortal)
eVS :ePortalBackend (EvidencePortal)
:SSI Authority/Enterprise Agent
:AriesAPI
:Database
show Connection Status (Button)
showConnectionStatus()
DIDConnStatus(userID)
getDIDConn(userID)
return DIDConn
alt
[no DIDConn]
return -1
[DIDConnStatus : connection_established]
return 1
[DIDConnStatus : invitation_generated]
/connections
return connections[]
alt
[connectionID : null]
return 0
[connectionID : exists]
/connections/{connectionID}/accept-request
return
updateDIDConn(userID, MyDID, theirDID, connectionID, status:connection_established)
return success
return 1
show connestionStatus(status)

Figure 40: Workflow of the DE4A SSI-authority agent for showing a Connection Status

4.3.2 Issuance of a VC-based Diploma

Figure 40 shows the process of the issuance of a Diploma as a Verifiable Credential. After authentication with eIDAS and establishment of secure DID connection (step 0), the Data Provider obtains evidence about diploma for the eIDAS authenticated user from the data owner (steps 1, 2). From the obtained evidence, a Verifiable Credential of student's diploma is generated based on the Europass EDCI Data Model (step 3). This verifiable credential is signed by DP’s DID (anchored on TIR) and later on sent to students' SSI-edge/mobile agent over the previously established DID connection (step 4). Users can preview and accept/save the received verifiable credential to their SSI-edge/mobile agent (steps 5, 6). After credential acceptance, the Data Provider backend updates and refreshes the eProcedure website and shows VC issuance status (steps 7, 8).

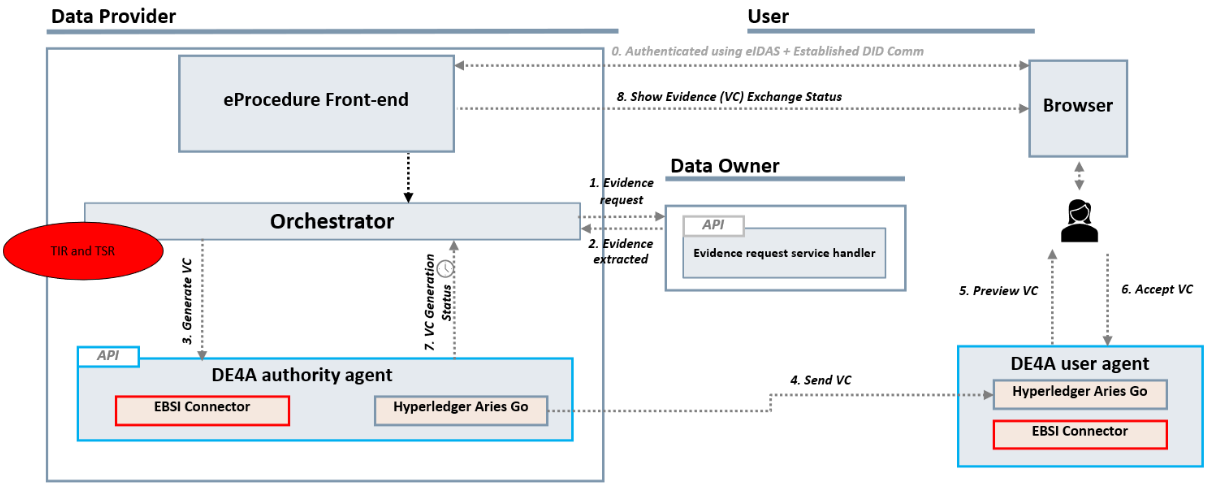


Figure 41: Process of Verifiable Credential issuance by Data Provider to User

To enable the issuing of the VC-based Diploma the following four steps of the process from Aries Go Issue Credential Protocol 2.0 are available:

1. Sending a VC offer by the usage of HTTP POST /issuecredential/send-offer

Input: MyDID, TheirDID

Execute:

```
curl -k -X POST "http://IP_PARTY1:PORT/issuecredential/send-offer" \
-H  "accept: application/json" \
-H  "Content-Type: application/json" \
-d '{
   "my_did": "did:peer:MyDID-COMES-HERE",
   "their_did": "did:peer:TheirDID-COMES-HERE",
   "offer_credential": {}
}'
```

Response:

```
{"piid":"PIID-IS-HERE"}
```

2. Accepting a VC offer through the usage of HTTP POST /issuecredential/{piid}/accept-offer

Input: piid

Execute:

```
curl -k -X POST "http://IP_PARTY2:PORT/issuecredential/PIID-COMES-HERE/accept-offer" -H
"accept: application/json"
```

Response:

```
{} -> this is OK response
```

3. Accepting a request by the usage of HTTP POST /issuecredential/{piid}/accept-request

Input: piid, VC (Diploma)
Execute:

```
curl -k -X POST "http://IP_PARTY1:PORT/issuecredential/PIID-COMES-HERE/accept-request" \
     -H  "accept: application/json" \
     -H  "Content-Type: application/json" \
     -d '{
       "issue_credential":{
         "credentials~attach":[
           VC-DIPLOMA-COMES-HERE
           ]
         }
     }'
```

Response:
{} -> this is OK response

4. Accepting the credential through the usage of HTTP POST /issuecredential/{piid}/accept-credential

Input: piid
Execute:

```
curl -k -X POST "http://IP_PARTY2:PORT/issuecredential/PIID-COMES-HERE/accept-credential"
\
-H  "accept: application/json" \
-H  "Content-Type: application/json" \
-d '{
  "names":[
    "VC-DIPLOMA-CREDENTIAL-LABEL-COMES-HERE"
  ]
}'
```

Response:
{} -> this is OK response

Aries API’s also allow for signing of the VCs. Only one step is needed to do so and is as follows:

1. Signing of the credential through the usage of POST /verifiable/signcredential

Input: VC, did, signature type (e.g., Ed25519Signature2018)

Example:

```
{
 "created": "2021-05-05T10:22:25.999Z",
 "credential": {
                    VC-COMES-HERE
               },
 "did": "did:peer:DID-COMES-HERE",
 "signatureType": "Ed25519Signature2018"
}
```

Execute:

```
curl -X POST "IP_PARTY2:PORT/verifiable/signcredential" -H  "accept: application/json" -H
"Content-Type: application/json" -d "{  \"created\": \"2021-05-05T10:22:25.999Z\",
\"credential\":{ VC-COMES-HERE }, \"did\": \"did:peer: DID-COMES-HERE \",
\"signatureType\": \"Ed25519Signature2018\"      }"
```

Example of the result of the signed VC:

```
{"verifiableCredential":{"@context":["https://www.w3.org/2018/credentials/v1","https://ww
w.w3.org/2018/credentials/examples/v1"],"credentialSubject":"sample-credential-subject-
id","id":"http://example.edu/credentials/1872","issuanceDate":"2010-01-
01T19:23:24Z","issuer":{"id":"did:example:09s12ec712ebc6f1c671ebfeb1f","name":"Example
University"},"proof":{"created":"2021-05-
05T10:22:25.999Z","jws":"eyJhbGciOiJFZERTQSIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..K
WqtauoLBkFQ8FopZ11V7SGho-sgS3fa_KvT9CrU9mDrvVAvR-
SDZ27NnRoMYUvkyJLXbwrL4FpuNVVX0w6MBg","proofPurpose":"assertionMethod","type":"
Ed25519Signature2018","verificationMethod":"did:peer:1zQmcpPretUFERRqA55FzUTDxBKXw
EjA6gQBqkUpyHE79Bh2#k_dVx2lYD4zVt2QhYs4SVk2nFmyGWsNVU6BFbOWYUGo"},"referenc
eNumber":83294847,"type":["VerifiableCredential","UniversityDegreeCredential"]}}
```

Through the use of the Aries API Issue Credential Protocol 2.0 the following workflows were designed for the DE4A SSI-authority agent (AA) regarding the VC-based Diploma handling on the DP side:

1. **Send me the VC offer** (Figure 42): The student executes the action of receiving the VC offer by clicking the related button at the Frontend of the Evidence Portal (FEP). The FEP calls the *sendVCOffer()* on the Backend of the Evidence Portal (BEP), which first prepares the evidence for the userID by executing the *prepareEvidence(userID)*. After the evidence is prepared, the BEP sends it to the AA through the *sendVCOffer(userID, evidence)*. The AA acquires the DIDConn values from the Database by calling the *getDIDConnStatus(userID)*, and then it generates the VC by executing the *generateVC(evidence, publicDID, myDID, theirDID)*. At this point, the VC needs to be signed. The AA sends the VC for signing to the AriesAPI through the */verifiable/signcredential (VC)*. After it is signed, the AA can send the signed offer to the AriesAPI through */issuecredential/send-offer (myDID, theirDID, VC)*. The AriesAPI responds with the return of the PIID (the presentation ID). The VC Status entity in the Database is updated for the PIID, and the enum status gets set to 'offer_sent' through calling *saveVCStatus(userID, PIID, VC, Status : offer_sent)*. The AA returns a successful sending of the VC offer to the BEP, which then shows the sending of VC information at the FEP.

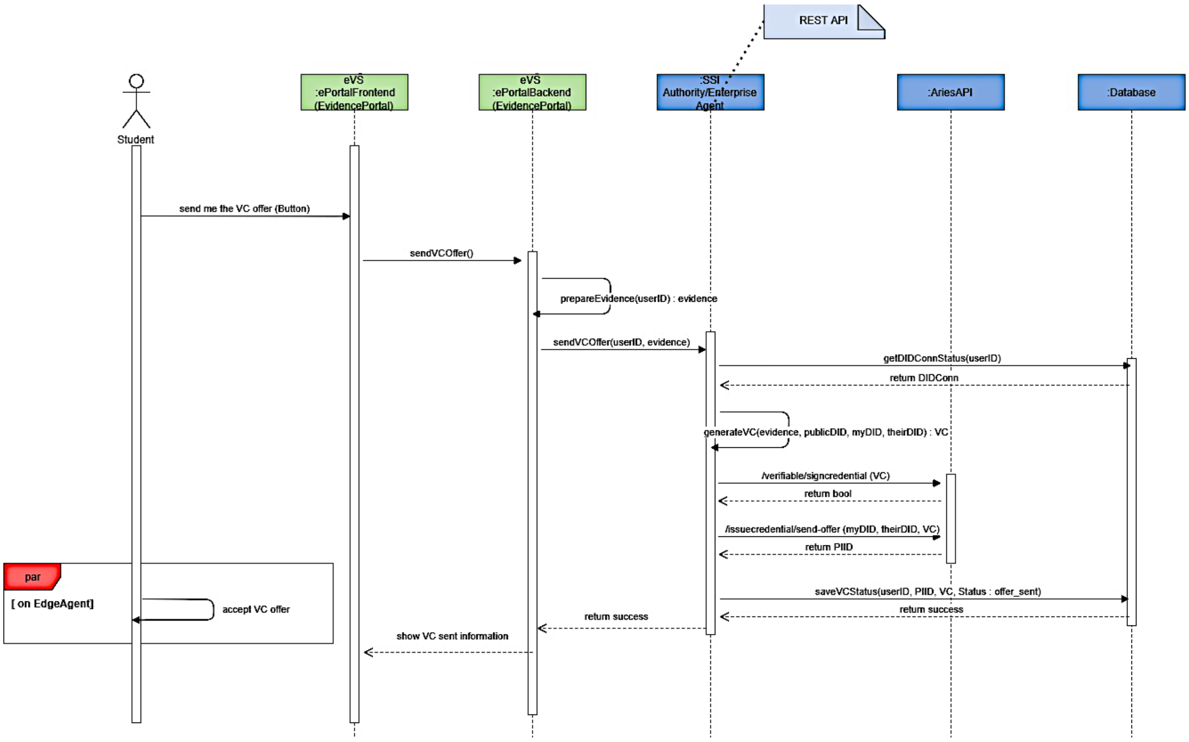


Figure 42: Workflow of the DE4A SSI-authority agent for sending the Verifiable Credential offer

2. **Check VCStatus** (Figure 43): The student executes the action of checking the VC status by clicking the related button at the FEP. The FEP calls the *checkOfferVCResponse()* on the BEP, which calls the AA's REST service VCStatus(userID). The AA executes the *getVCStatus(userID)* on the Database, and, in return, receives the VCStatus (values). Based on the VCStatus (values), there are the following possible alternatives:
    a. if there is no VCStatus found, the AA will return value -1 to the BEP. Value -1 signals that the offer was never sent beforehand, and the FEP shows the send me VC offer.
    b. If the enum status of the VCStatus is set to the ‘offer_accepted’, the AA will return value 1 to the BEP, and if it is set to the ‘vc_accepted’, the AA will return value 5. Value 1 signals that the offer was accepted, and the FEP shows the send me VC. Value 5 signals that the VC was accepted, and the FEP shows the send me offer.
    c. If the enum status of the VCStatus is set to the ‘offer_rejected’, the AA will return value -2 to the BEP, and if it is set to the ‘vc_rejected’, the AA will return value -4. Value -2 signals that the offer was rejected, and the FEP shows send me an offer. Value -4 signals that the VC was rejected, and the FEP shows send me an offer.
    d. If the enum status of the VCStatus is set to the ‘offer_sent’ or ‘vc_sent’ the AA will first acquire a list of pending actions from the AriesAPI by calling */issuecredential/actions*, and based on the information from the list, there are three alternatives possible:
        i. If the PIID doesn't exist in the actions list and the offer was sent, the AA will return a value of 0 to the BEP. Value 0 signals that the FEP informs the student to respond on the Edge Agent.
        ii. If the PIID doesn't exist in the actions list and the VC was sent, the AA will return value 2. Value 2 signals that the FEP informs the student to respond on the Edge Agent.

iii. In case the PIID exists in the actions list, and the enum status is set to rejected, the AA first updates the Database through the call *updateVCStatus(userID, Status: offer_rejected || vc_rejected)* (note: The student can either reject the sent offer from the start or initially accept the offer, but then, later on, decide to reject the VC), and secondly, returns the value of -2 or -4 to the BEP. Value -2 signals that the offer was rejected, and value -4 signals that the VC was rejected. In both cases, the FEP shows send an offer.

iv. Suppose the PIID exists in the actions list, and the enum status is set to accepted. In that case, the AA first updates the Database through the call *updateVCStatus(userID, Status: offer_accepted || vc_accepted)*, and secondly returns the value of 1 or 5 to the BEP. Value 1 signals that the offer was accepted, and the FEP shows send VC. Value 5 signals that the VC was accepted, and the FEP shows send an offer.

The BEP returns the VC offer status to the FEP.

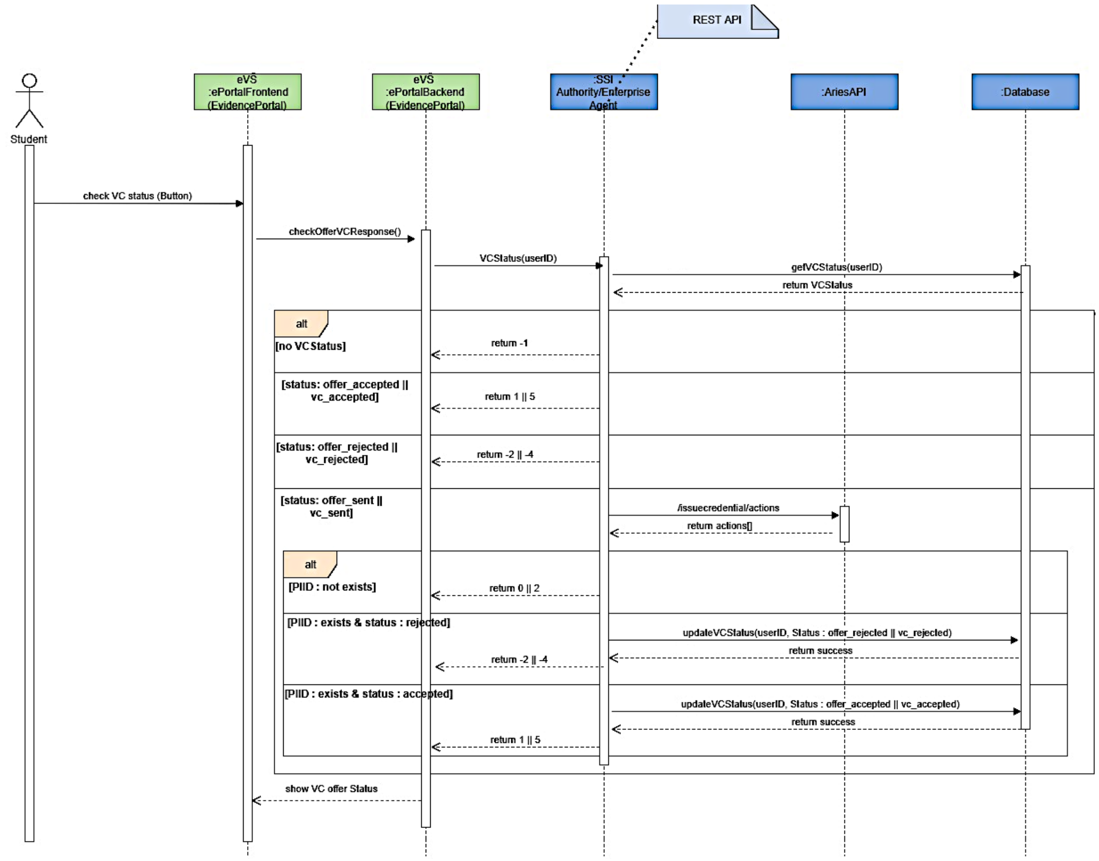

REST API
Student
eVS :ePortalFrontend (EvidencePortal)
eVS :ePortalBackend (EvidencePortal)
:SSI Authority/Enterprise Agent
:AriesAPI
:Database
check VC status (Button)
checkOfferVCResponse()
VCStatus(userID)
getVCStatus(userID)
return VCStatus
alt
[no VCStatus]
return -1
[status: offer_accepted || vc_accepted]
return 1 || 5
[status: offer_rejected || vc_rejected]
return -2 || -4
[status: offer_sent || vc_sent]
/issuecredential/actions
return actions[]
alt
[PIID : not exists]
return 0 || 2
[PIID : exists & status : rejected]
updateVCStatus(userID, Status : offer_rejected || vc_rejected)
return success
return -2 || -4
[PIID : exists & status : accepted]
updateVCStatus(userID, Status : offer_accepted || vc_accepted)
return success
return 1 || 5
show VC offer Status

Figure 43: Workflow of the DE4A SSI-authority agent for checking the Verifiable Credential status

3. **Send me the VC** (Figure 44): The student executes the action of sending the VC by clicking the corresponding button at the FEP. The FEP calls the *sendVC()* on the BEP. The BEP calls the REST service of *sendVC(userID)*, with the userID as a necessary input argument to the service, on the AA. The AA makes an internal call to the Database with the getVCStatus(userID) and receives the VCStatus, including the VC. The AA then executes */issuecredential/{PIID}/accept-request(VC)*. The AA makes an update to the Database through the call *updateVCStatus(userID,*

status: vc_sent). The AA then returns the value stating success to the BEP, and the FEP shows the VC sent information.

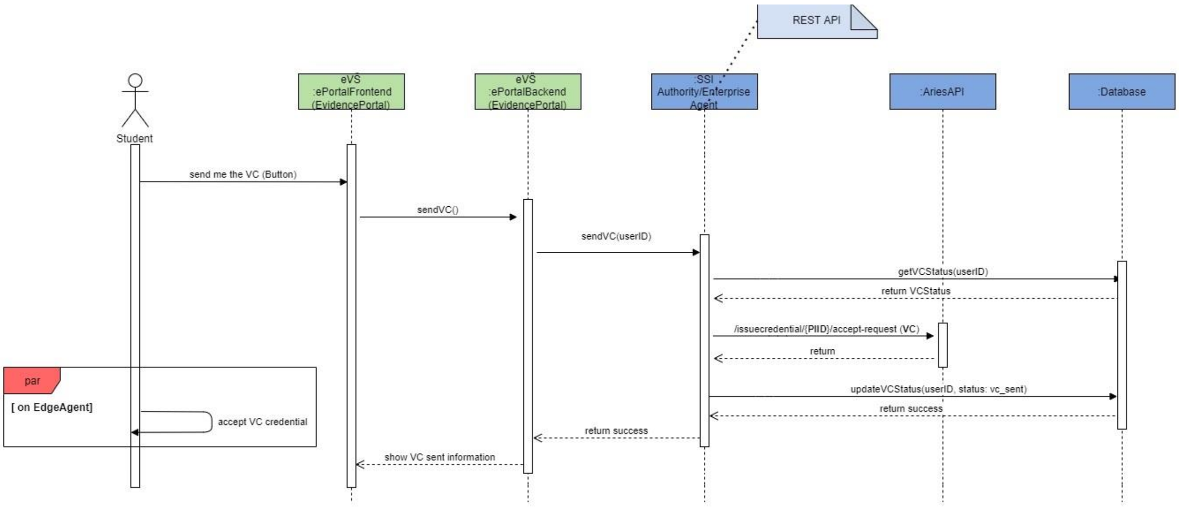


Figure 44: Workflow of the DE4A SSI-authority agent for sending the Verifiable Credential

4.3.3 Diploma verification

The last message flow is shown in Figure 45. Data consumer starts the process with a proof request that contains data definition of Verifiable Credential of the diploma (step 1). The proof is sent agent-to-agent to the user (step 2). Students can generate proof/evidence (Verifiable Presentation) based on their Verifiable Credential saved in the SSI-edge/mobile agent (steps 3, 4). Verifiable Presentation is then sent agent-to-agent to the SSI-authority agent (step 5). After receiving a Verifiable Presentation, Data Consumers can verify it using their Authority Agent (AA):

- Verify the VC integrity by checking the VC/VP issuer's digital signature (DID),
- verify the authenticity by checking VC/VP is in the EBSI/ESSIF TIR,
- verify the correctness by checking VC/VP schema is correct and in the EBSI/ESSIF TSR,
- verify the authenticity of the student/holder's identity by comparing the VC subject parameter data with eIDAS minimal dataset based on the eProcedure Portal eIDAS login (step 6).

After evaluating the received Verifiable Presentation, the eProcedure website is updated and refreshed with exchange status and validation of evidence (step 7).

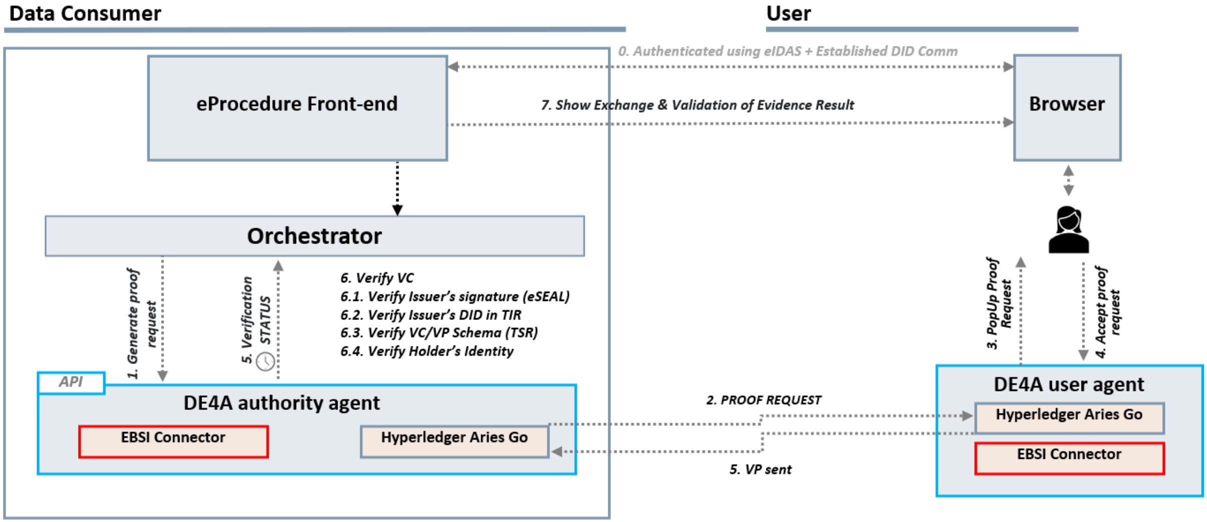


Figure 45: Process of providing Verifiable Presentation by User for Data Consumer

To enable an exchange of the Verifiable Presentation (VP) by the usage of Aries Go Present Proof Protocol 2.0, the following process consisting of three steps is used:

1. Sending a request for VP by the usage of POST /presentproof/send-request-presentation

   Input: MyDID, TheirDID
   Execute:
   ```
   curl -k -X POST "http://IP_PARTY1:PORT/presentproof/send-request-presentation" -H
   "accept: application/json" -H  "Content-Type: application/json" -d '{
   "my_did":"did:peer:MyDID-COMES-HERE",
   "their_did":"did:peer:TheirDID-COMES-HERE ",
   "request_presentation":{}
   }'
   ```
   Result:
   ```
   {"piid":"PIID-IS-HERE"}
   ```

2. Accepting the VP by the usage of HTTP POST /presentproof/{piid}/accept-request-presentation
   (Note: includes step for accepting the request presentation with base64 encoded VP payload.)

   Input: piid
   Execute:
   ```
   curl -k -X POST "http://IP_PARTY2:PORT/presentproof/PIID-COMES-HERE/accept-request-presentation" \
   -H  "accept: application/json" \
   -H  "Content-Type: application/json" \
   -d '{
     "presentation":{
       "presentations~attach":[
         {
           "lastmod_time":"0001-01-01T00:00:00Z",
           "data":{
             "base64":"ENCODED-VP-COMES-HERE"
             // other data formats are possible as well (e.g. JSON)
           }
         }
       ]
     }
   }'
   ```
   Result:
   ```
   {} -> this is OK response
   ```

3. Accepting the VP by the usage of HTTP POST /presentproof/{piid}/accept-presentation

Input: piid
Execute:

```
curl -k -X POST "http://IP_PARTY1:PORT/presentproof/PIID-COMES-HERE" \
-H  "accept: application/json" \
-H  "Content-Type: application/json" \
-d '{
  "names":[
    "VP-LABEL-COMES-HERE"
  ]
   }'
```

By the usage of the Aries Go Present Proof Protocol 2.0 process, the following workflows were designed for the DE4A SSI-authority agent regarding the VP Diploma handling on the DC side:

1. **Send me the VP request** (Figure 46)**:** The student executes the action of receiving the VP request by clicking the corresponding button at the FEP. The FEP calls the *sendVPRequest()* on the BEP, which first obtains the presentation. The optional step at this point is to execute the checking of the schema on the EBSI (TSR). The FEP then calls the *sendVPRequest(userID, presentation)* on the AA. The AA requests the DIDConn from the Database by calling the *getDIDConnStatus(userID)*, and generates the VPRequest through the *generateVPRequest(presentation, myDID, theirDID)* (note: myDID and theirDID are part of the DIDConn). The AA then executes the */presentproof/send-request-presentation(VPRequest)* on the AriesAPI, which returns the PIID, and saves the VPStatus to the Database through *saveVPStatus(userID, PIID, Status: request_sent)*. The value signaling success is then transmitted from the AA to the BEP, and verifiable credential sent information is shown on the FEP.

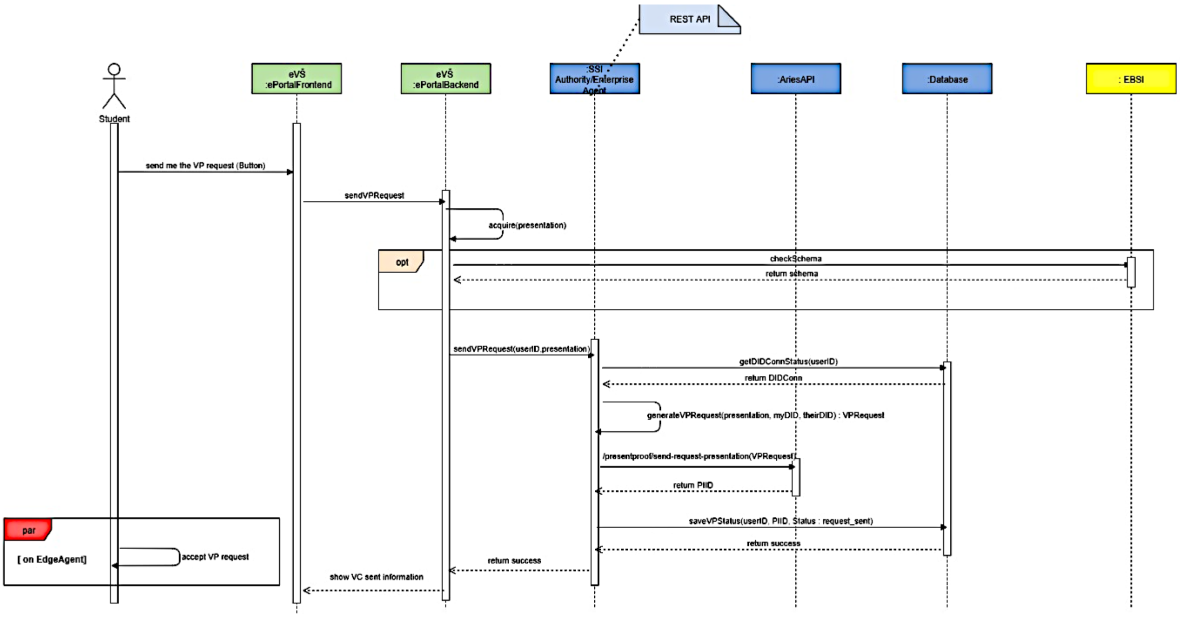


Figure 46: Workflow of the DE4A SSI-authority agent for sending the Verifiable Presentation request

2. **Check VP status** (Figure 47): The student executes the action of checking the VP status by clicking the related button at the FEP. The FEP calls the *checkRequestVPResponse()* on the BEP, which calls the REST service *VPStatus(userID)* of the AA. The AA executes the *getVPStatus(userID)* on the Database, and, in return, receives the VPStatus (values). Based on the VPStatus (values), there are the following possible alternatives:
    - If there is no VPStatus found, the AA will return value -1 to the BEP. Value -1 signals that the request was never sent beforehand, and the FEP shows the send me VP request.
    - If the enum status of the VPStatus is set to 'vp_received', the AA will return value 1 to the BEP. Value 1 signals that the VP was received, and the FEP shows the info.
    - If the enum status of the VPStatus is set to 'vp_rejected', the AA will return value -2 to the BEP. Value -2 signals that the request was rejected, and the FEP shows the send request.
    - In case the enum status of the VPStatus is set to the value 'request_sent', the AA will first obtain the list of pending actions from the AriesAPI by calling */presentproof/actions* and, based on the information from the list, there are three alternatives possible:
        - If the PIID doesn't exist in the actions list, the AA will return a value 0 to the BEP. Value 0 signals that the request was sent, and the FEP informs the student to respond on the Edge Agent.
        - If the PIID exists in the actions list and the VC was sent, the AA will return value 2. Value 2 signals that the FEP informs the student to respond on the Edge Agent.
        - Suppose the PIID exists in the actions list, and the enum status is set to 'rejected'. In that case, the AA first updates the Database through the call *updateVPStatus(userID, Status: vp_rejected)*, and then returns the value of -2 to the BEP. Value -2 signals that the request was rejected, and the FEP shows send a request.
        - If the PIID exists in the actions list and the enum status is set to 'accepted', then the AA firstly generates the name of the presentation based on userID and PIID. Secondly, the call is made to the AriesAPI for accepting the presentation (i.e., saving the presentation in the Aries Agent) by executing */presentproof/{PIID}/accept-presentation(name)*. Thirdly, the AA updates the Database through the call *updateVPStatus(userID, Status: vp_received).* Then the value 1 is returned to the BEP. Value 1 signals that the VP was received, and the FEP shows the info.

    The BEP returns the VP request status back to the FEP.

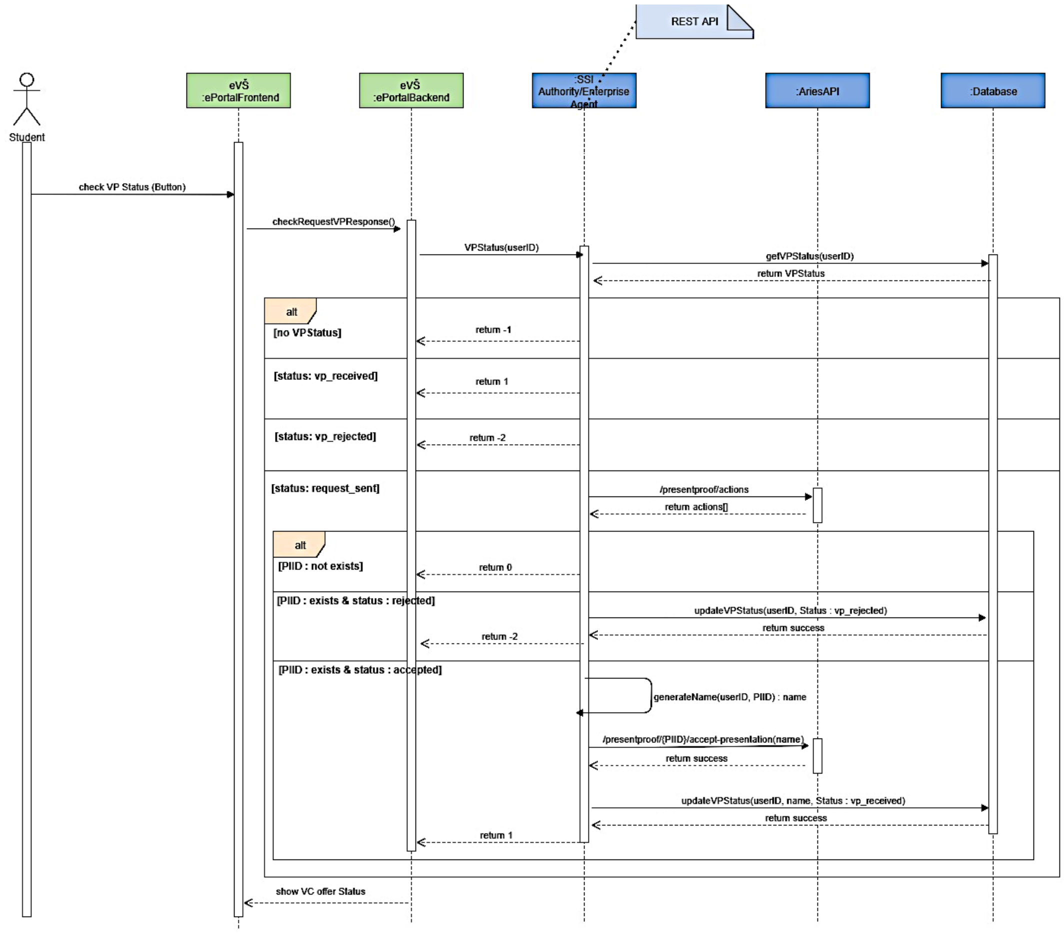


Figure 47: Workflow of the DE4A SSI-authority agent for checking the Verifiable Presentation status

3. **Validate VP** (optional) (Figure 48): The student executes the optional operation of validating the VP by clicking the applicable button at the FEP. The FEP calls the *validateVP()* on the BEP, and the BEP calls the *ValidateVP(userid, eidasMDS)* on the AA. Then, the AA first retrieves the VPStatus (values) from the Database by executing *getVPStatus(userID)*. Secondly, the AA receives the list of presentations from the AriesAPI by executing */verifiable/presentations*. Thirdly, the AA uses the name of the presentation (the name follows the format of vp-{userId}-{piid}) to find the id of the VP from the list of presentations. Fourthly, the id is encoded to the base64 format (the format required by the AriesAPI when searching for a specific presentation). Fifthly, by using the encoded en_id, the call is made to AriesAPI through */verifiable/presentation/(en_id)* and the VP is returned. Sixthly, the AA checks the VPSubject and eidasMDS. There are two possible alternatives based on the check:
   - If the VP subject does not match the eIDAS login the value -1 is returned to the BEP, and the FEP shows the info.
   - If the VP subject matches the eIDAS login the issuer is checked on the EBSI through the method checkISSUER(). Two alternatives that follow the check are:
     - If the issuer is not a valid TIR the value -2 is returned to the BEP, and the info about VP status will be shown on the FEP.

- If the issuer is a valid TIR the value 1 is returned to the BEP, and the info about VP status will be shown on the FEP.

Lastly (step 7), the AA updates the VPStatus in the Database by calling the *updateVPStatus(userID, Status : vp_validated, VP)*. The AA returns the status and the name of the presentation to the BEP. The FEP shows the VP validation status to the student.

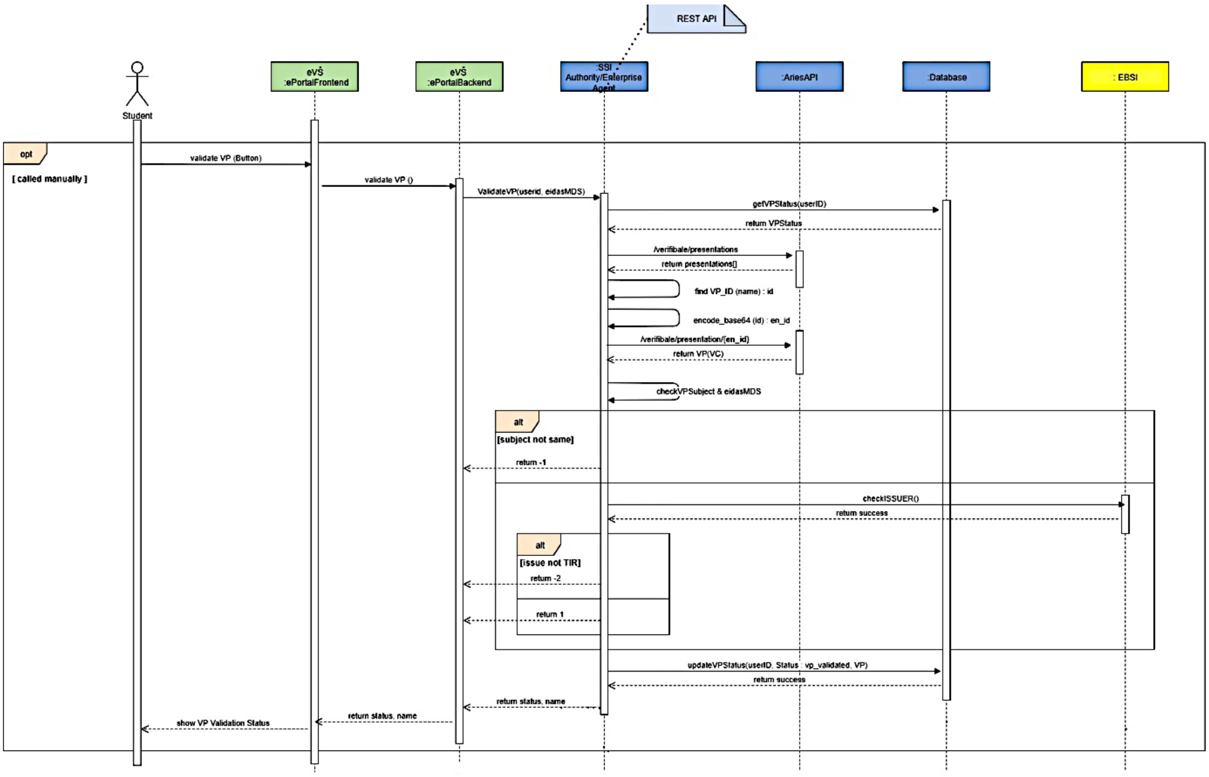


Figure 48: Workflow of the DE4A SSI-authority agent for validating a Verifiable Presentation

4. **getVP** (optional)(Figure 49): The student executes the optional operation of getting the VP by clicking the applicable button at the FEP. The FEP calls the getVP() on the BEP, and the BEP calls the getVP(name) on the AA (the name follows the format of vp-{userId}-{piid}). The AA then first checks the VP in the Database by calling the method checkVP(name). Secondly, the list of all presentations is acquired from the AriesAPI. Thirdly, the id of the VP is searched in the list by calling the method findVP_ID(name) on the AA. Fourthly, the id is encoded to the base64 format. Fifthly, by using the encoded en_id, the call is made to AriesAPI through /verifiable/presentation/(en_id), and the VP is returned. Lastly, the AA returns the VP to the BEP, and the FEP displays it.

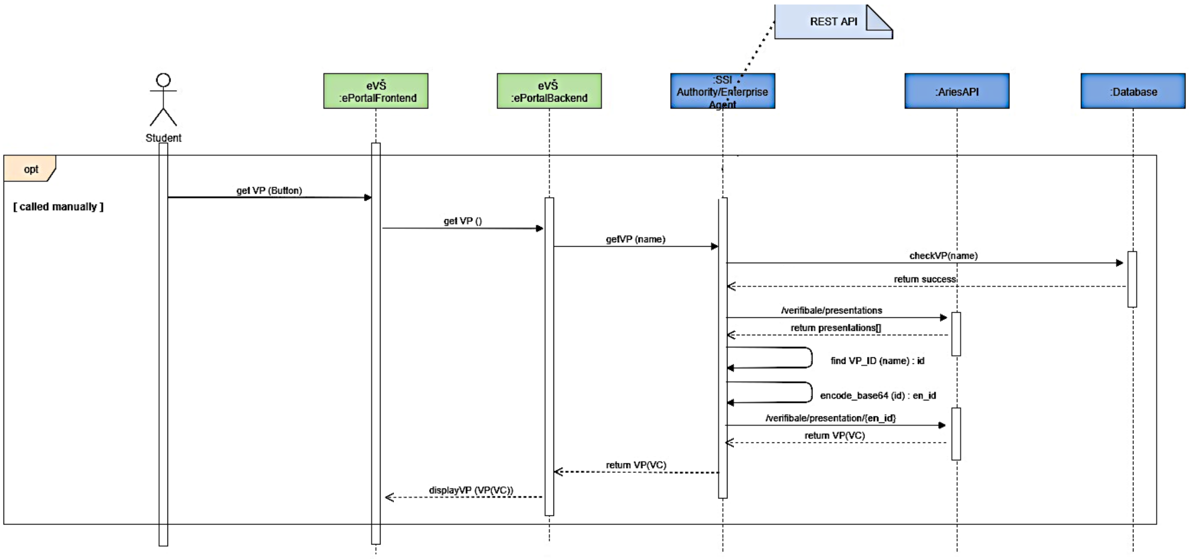


Figure 49: Workflow of the DE4A SSI-authority agent for getting a Verifiable Presentation

4.4 Legal and Security considerations

Although the groundwork on the DE4A SSI supporting framework (the design of the DE4A VC pattern and the development of the required DE4A SSI components), is finalising and preparing for the second pilot iteration, there are still some open questions and challenges from the legal and security perspective, which must be outlined and are of conceptual nature:

- While issuing Diploma VC (VA), these are being generated and signed by Data Providers (i.e., ministries), who sign the VA with their public DID:ebsi. Although this achieves cryptographically sound solution in terms of security, such an e-document is not legally binding due to the missing eSEAL (a cross-border recognised type of Trust Service under eIDAS regulation). In the case of DE4A, the Data Consumers in case of the Diploma VC, validate the aforementioned document by verifying the DID-based signature and by searching for the issuer's legal information within the EBSI Trust Issuer Registry. The implementation of eSEALs was not possible due to technical restrictions e.g., using W3C JSON-LD based VC, which cannot be eSEALed. Nevertheless, there is already work in progress by DE4A and other projects to achieve this in the near future.
- Within the DE4A VC pattern planning it was decided that innovative approaches would be used, thus DID-comm was chosen as the way to transfer evidence (i.e., Diploma in the form of VC) between students and DP/DC. However, in order to achieve DID-comm, a DID exchange has to be negotiated between the aforementioned stakeholders, which in the case of DE4A was designed with a pure SSI approach i.e., by generating DID invitation QR codes, which are then scanned by the students and their SSI edge agents (i.e., wallets). The QR codes are however showcased on the display of the formal ePortals (evidence portal, eProcedure portal), which furthermore require from the student a successful eID or eIDAS authentication. The issue with the DID invitation QR code is the fact that it is prone to so-called shoulder surfing attacks, which could produce a malicious attacker who scans the QR code intended for a legitimate user, thus resulting in a DID-comm with the portals, who could in the future send it a Diploma VC from the legitimate user. DE4A addresses this issue in the sense that even if the malicious attacker would execute the attack, he/she would not be able to use received VC from legitimate users, since the VCs (VPs) are cross checked with the VC subject parameter values – a reason why eIDAS MDS are used within DE4A Diploma VCs. Nevertheless, there is already R&D work in progress by DE4A to mitigate the issue, whereby encryption techniques are being envisaged in forefront.

More details on legal and security considerations can be found in Annex I: Legal discussion note on using Verifiable Credentials in the DE4A Studying Abroad pilot.

5 Conclusions

This document describes the final version of the Self-Sovereign Identity solution implemented by the DE4A project and integrated with the EBSI/ESSIF infrastructure.

The challenges faced by the project during the implementation can be summarised as follows:

- the standards which the implementation relies on are still not complete and some aspects are still not fully defined,
- delays in having APIs with corresponding final technical documentation available for integration by Early Adopters,
- the underlying software used by the consortium is still under development by the open-source community,
- given the DE4A piloting timeline, DE4A development had to start before EBSI/ESSIF started providing services and even before they published any documentation/guidelines.

Despite these challenges that the consortium successfully addressed in collaboration with EBSI, the following items have been delivered:

- Generation of EBSI compliant DIDs
- Adoption/Integration of the cryptographic techniques supported by ESSIF services
- Anchoring public DIDs into the EBSI blockchain infrastructure following ESSIF guidelines
- Registering the Data Providers' DID into EBSI blockchain infrastructure by using the EBSI Trusted Issuer Registry (TIR)
- Signing Verifiable Credentials with the EBSI compliant DID
- Generation of DIDs exchange between Data Providers/Data Consumers and students based on Hyperledger Aries protocols
- Exchanging Verifiable Credentials as representations of evidence signed with EBSI compliant DID through the DID Comm standard
- Generating Verifiable Credentials Schema definitions fully complying with ESSIF Trusted Schema Registry (TSR) and based on EDCI-Europass learning model.
- Going beyond the ESSIF services, providing JSON-LD support for the above-mentioned schema definitions.
- Generating DID-signed proofs from the students as evidence that they are in possession of the Verifiable Credentials
- Providing an asynchronous GUI at both ends (Authority Agent and Mobile app), improving the end user experience considerably
- Enhancing the GUI at both ends drastically (Authority Agent and Mobile app) by simplifying the iterations with the end user, consequently improving the usability of the final solution.

While differences in specific requirements and timing between the project and external entities, such as eIDAS or EBSI/ESSIF, had an influence on functionalities that could be included into the Self-Sovereign Identity solution implemented by DE4A, these were not of a critical nature and a very high and satisfactory level of alignment was effectively achieved through participation of the project in EBSI's Early Adopters programme. In some cases, fundamental technical implementation choices underlying the solution have been maintained, when necessary, after thorough discussion with EBSI team. For example, ESSIF detailed in their guidelines the use of the Self-Issued OpenID Provider (SIOP) [50] standard for the Self-Sovereign Identity message interchange quite late for development process of DE4A, considering the DE4A project had already been working for months in the development of the solution based on a different standard (DID Comm [51]). Having discussed this point with ESSIF's team during different specific teleconferences, they communicated to the consortium the intent to support providing services using different standards (including the solution based on DID Comm). Hence, the Consortium considered it better to continue refining and improving the solution based on DID Comm. Unfortunately, at the moment of writing this report ESSIF has not yet extended their services supporting other standards. Therefore, for a further integration with ESSIF following currently supported standards in the framework, it can be envisaged to include implementation of the SIOP standard in Mobile Edge Agent and Authority Agent components which will in addition provide added value in terms of facilitating interoperability with stakeholders using federated identity protocols such as OIDC.

In June 2021 the European Commission has issued a proposal for revision of the eIDAS regulation, which heavily relies on the concept of Digital Identity Wallets for the implementation of a new European Digital Identity Framework. It is in this context that the DE4A Self-Sovereign Identity Solution acquires special significance, even if the technical specifications of the eIDAS wallets will not be available until October 2022 (common Union Toolbox [52]), as it already embodies much of the needed functionality required both for governmental agencies as issuers and verifiers of electronic attestations and for EU citizens as final holders of wallets to be provided by their Member States to them. DE4A and its Member States closely follow discussions and proposals from the Commission to address synergies between the revised eIDAS ecosystem of digital wallets and the SDG Once-Only Technical System for C2G and C2B interactions which are clear when considering how personal identity data in wallets will be used for authentication to online SDG procedures and where standardised electronic attestations of attributes will also become usable to be presented from wallets by citizens and company representatives to such procedures, similar to the Diplomas use case in which DE4A SSI solution is being validated.

References

[1] DE4A D5.7 First Release of DE4A Blockchain Supporting Framework
https://www.de4a.eu/project-deliverables, retrieved 2021-05-30

[2] DE4A D2.4 Project Start Architecture – First Iteration
https://www.de4a.eu/project-deliverables, retrieved 2021-03-22

[3] uPort home page
https://www.serto.id, retrieved 2021-03-17

[4] ConsenSys Mesh home page,
https://mesh.xyz, retrieved 2021-03-17

[5] Ethereum home page,
https://ethereum.org/en/, retrieved 2021-03-17

[6] uPort mobile app in Github,
https://github.com/uport-project/uport-mobile, retrieved 2021-04-22

[7] DIF home page,
https://identity.foundation retrieved 2021-03-25

[8] Infura Home page,
https://infura.io , retrieved 2021-03-22

[9] Sovrin Home page,
https://sovrin.org , retrieved 2021-03-22

[10]Evernym Home page,
https://www.evernym.com, retrieved 2021-03-24

[11]W3C Credentials Community Group,
https://w3c-ccg.github.io , retrieved 2021-03-26

[12]DIF Identifiers & Discovery,
https://identity.foundation/working-groups/identifiers-discovery.html, retrieved 2021-04-01

[13]DIF authentication,
https://identity.foundation/working-groups/authentication.html, retrieved 2021-04-01

[14]DIF Claims & Credentials,
https://identity.foundation/working-groups/claims-credentials.html, retrieved 2021-04-01

[15]DID communication,
https://identity.foundation/working-groups/did-comm.html, retrieved 2021-04-01

[16]Indy node repository,
https://github.com/hyperledger/indy-node, retrieved 2021-04-06

[17]Indy SDK repository,
https://github.com/hyperledger/indy-sdk, retrieved 2021-04-06

[18]EBSI home page,
https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/EBSI, retrieved 2021-03-22

[19]Hyperledger Besu
https://www.hyperledger.org/use/besu , retrieved 2021-03-29

[20]Hyperledger Fabric, https://www.hyperledger.org/use/fabric, retrieved 2021-03-29

[21]Aries Cloud Agent, https://github.com/hyperledger/aries-cloudagent-python, retrieved 2021-04-01

[22]Aries GO, https://github.com/hyperledger/aries-framework-go, retrieved 2021-04-01

[23]DE4A D4.1 Studying Abroad -Use cases definition and requirements, https://www.de4a.eu/project-deliverables, retrieved 2021-03-22

[24]De4A D2.2 Initial DE4A Trust Management Models and Blockchain Support Framework, https://www.de4a.eu/project-deliverables, retrieved 2021-03-22

[25]Verifiable Credentials Data Model, https://www.w3.org/TR/vc-data-model/, retrieved 2021-04-1

[26]Europass Learning Model, https://github.com/anthonycamilleri/ELM_XSD2JSON, retrieved 2021-04-18

[27]Data model EDCI, retrieved 2021-03-29 https://europa.eu/europass/en/europass-digital-credentials-interoperability

[28]Aries RFC 0023: DID Exchange Protocol 1.0, retrieved 2021-03-29 https://github.com/hyperledger/aries-rfcs/tree/master/features/0023-did-exchange

[29]Aries RFC 0453: Issue Credential Protocol 2.0, retrieved 2021-03-29 https://github.com/hyperledger/aries-rfcs/tree/master/features/0453-issue-credential-v2

[30]Aries RFC 0454: Present Proof Protocol 2.0, retrieved 2021-03-29 https://github.com/hyperledger/aries-rfcs/tree/master/features/0454-present-proof-v2

[31]ESSIF home page, retrieved 2021-04-01 https://ec.europa.eu/cefdigital/wiki/pages/viewpage.action?pageId=262505734

[32]W3C Terminology, retrieved 2022-05-04 https://www.w3.org/TR/vc-data-model/#terminology

[33]EBSI Terminology, retrieved 2022-04-23 https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/EBSI+Terminology

[34]DIDComm basic message, retrieved 2022-04-23 https://didcomm.org/basicmessage/2.0/

[35]DIDComm Report problem, retrieved 2022-04-23 https://didcomm.org/report-problem/2.0/

[36]Aries ACK, retrieved 2022-04-23 https://github.com/hyperledger/aries-rfcs/blob/main/features/0015-acks/README.md#tutorial

[37]A trusted and secure European e-ID – Regulation, Proposal for a Regulation of the European Parliament and of the Council amending Regulation (EU) No 910/2014 as regards establishing a framework for a European Digital Identity (SEC(2021) 228 final) - (SWD(2021) 124 final) - (SWD(2021) 125 final), https://digital-strategy.ec.europa.eu/en/library/trusted-and-secure-european-e-id-regulation

[38]Commission Recommendation (EU) 2021/946 of 3 June 2021 on a common Union Toolbox for a coordinated approach towards a European Digital Identity Framework, retrieved 2022-05-03 https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32021H0946

[39]Trusted Issuers Registry API, retrieved 2022-05-03, https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/Trusted+Issuers+Registry+API

[40]DID Registry API, retrieved 2022-05-03, https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/DID+Registry+API

[41]Trusted Schemas Registry API, retrieved 2022-05-03, https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/Trusted+Schemas+Registry+API

[42]ESSIF Functional Scope, retrieved 2022-05-06, https://ec.europa.eu/digital-building-blocks/wikis/pages/viewpage.action?pageId=379913698

[43]Diplomas Functional Documentation, retrieved 2022-05-06, https://ec.europa.eu/digital-building-blocks/wikis/display/EBSIDOC/%5Barchived%5DDiplomas+Functional+Documentation

[44]European Blockchain Services Infrastructure-EBSI Use Cases and Roadmap, retrieved 2022-05-06, https://digital-strategy.ec.europa.eu/en/policies/european-blockchain-services-infrastructure

[45] ESBI Architecture explained, retrieved 2021-06-10

https://ec.europa.eu/digital-building-blocks/wikis/download/attachments/385876155/%28210610%29%28EBSI_Architecture_Explained%29%28v1.02%29.pdf?version=1&modificationDate=1623329489960&api=v2

[46]DE4A D2.3 Final DE4A Trust Management Models and Blockchain Support Framework Design, https://www.de4a.eu/project-deliverables, retrieved 2021-05-03

[47]Europass Learning Model, https://github.com/european-commission-empl/European-Learning-Model, retrieved 2022-05-04

[48]What are digital credentials?, https://europa.eu/europass/en/what-are-digital-credentials, retrieved 2022-05-04

[49] Aries Cloud Agent Python (ACA-py) Webhooks https://ldej.nl/post/aries-cloudagent-python-webhooks/ retrieved 2022-05-04

[50]Self-Issued OpenID Provider v2, https://openid.net/specs/openid-connect-self-issued-v2-1_0.html retrieved 2022-05-04

[51] DIDComm Messaging, https://identity.foundation/didcomm-messaging/spec/, retrieved 2022-05-04

[52]Commission Recommendation (EU) 2021/946 of 3 June 2021 on a common Union Toolbox for a coordinated approach towards a European Digital Identity Framework, retrieved 2022-05-04, EUR-Lex - 32021H0946 - EN - EUR-Lex (europa.eu)

Annex I: Legal discussion note on using Verifiable Credentials in the DE4A Studying Abroad pilot

A. Contextual background on current ambitions in DE4A

Within the DE4A project, the Studying Abroad pilot aims to explore the use of an information exchange pattern based on Verifiable Credentials (VC), specifically for the use case of Diploma Recognition. The VC exchange pattern is a form of a User-managed access pattern, in the sense that the user intermediates in the data exchange (rather than the data being exchanged directly between public administrations). In this way, the pattern can conceptually more easily support multi-evidence exchanges and interrupted/deferred procedures where data may not be instantaneously available from all relevant sources (e.g. a public administration first needs to create specific digital evidence that is not immediately available for access or download when the user requests it, and the creation process takes minutes, hours or even days). Such interrupted/deferred procedures should be feasible from a technological perspective but are currently not expected to be piloted in DE4A Studying Abroad pilot.

As explained in the deliverable D2.4 Project Start Architecture (PSA) – First iteration [2], "*Data stored in the form of Verifiable Credentials (VC) are data representations in the form of a set of claims about some subject (i.e. User) issued by the issuer (i.e. Data Provider). Verifiable Credentials can be cryptographically verified by any third party (i.e. Data Consumer (DC) to whom Verifiable Credentials is presented (usually in the form of a Verifiable Presentation).*

The Verifiable Credentials pattern (VC pattern) utilizes blockchain technology features in several ways.

First, storing decentralized identifiers (DIDs) and its correlating DID documents, which includes all relevant entity pieces of information about the issuer, including associated cryptographic keys, endpoints, etc. that can be used to authenticate the issuer (i.e. Data Provider (DP), and cryptographically validate VC that was issued by its DID.

Second, storing and maintaining a list of verified/trusted issuers, i.e. DPs.

Third, keep the list of revoked VCs [ed. This functionality is not expected to be piloted in DE4A as it will be supported in future versions of ESSIF]. *Furthermore, all other entities (i.e. DC, and Users) also have DIDs, and related DID documents, that are different than the DC information stored directly on their devices, i.e. Agents (edge or cloud). These DIDs are used for setting direct, i.e. DID communication between entities.*

The VCs are issued to a User in a cryptographically secure manner collected in a user-maintained digital wallet that is part of the edge agent (i.e. mobile phone) under his possession. Edge agent serves as an instrument with which all secure interchanges are managed (i.e. Initiate DID connection, Accept DID connection, Accept Verifiable Credential, Present Verifiable Credential). Moreover, the managing of DID connections, VC issuing and verifying operated by DPs and DCs is handled through a dedicated cloud agent".

The ambition is that the DIDs of Data Providers – meaning in practice ministries of education and/or educational institutions within the DE4A project – will be stored in the EBSI ledger through the Early Adopters programme, as well as the schema describing such VCs. The EBSI ledger framework would be used for verification when the VC (containing a diploma) is presented to a relying party (e.g., a university).

Contrary to the standard EBSI framework approach, DE4A will not be using eSeals to sign the VCs, but instead use a DID signing method.

Further to this, DE4A will not use ESSIF Verifiable IDs since DE4A will be authenticating the user with eIDAS notified eIDs towards the Issuer of the Diploma information (Data Provider), and also towards the Verifier of the Diploma to whom the student will present this information (Data Consumer) from a Wallet they control.

The eIDAS MDS information will be included in the Verifiable Credential itself by the Issuer, so that it will be possible to verify that the user authenticating at the Verifier is the same referred to in the Diploma VC.

B. Legal challenges and working assumptions

The constraints introduced above were partially driven by the following legal considerations:

- DE4A needs to consider the legal framework that exists today. The legal framework of the SDGR does not enter into force until the end of 2023, and as a result the eIDAS Regulation is the principal form of eID that is legally acceptable towards both Data Providers and Data Consumers. Support for these will at any rate be implemented, which is why **user authentication is based on direct identification towards these entities with an eIDAS notified eID**[2]; introduction of ESSIF Verifiable IDs is unlikely to be a facilitator on that point.
- For completeness, it is also worth noting that the Commission has published a proposal for amendment of the eIDAS Regulation[3], and that this proposal would create a specific legal framework for the creation and maintenance of electronic ledgers, for electronic attestations of attributes, and for a European Digital Identity Wallet. In many ways, the VC patterns are a preview of the potential impact of this proposal since it pilots many of the new legislative concepts. The proposal does not have direct binding impacts on DE4A however, since it is currently not yet finalised, and will not enter into application until some time after the completion of the DE4A project.
- Furthermore, the DE4A consortium would want to **avoid introducing any personal data on any ledger** under the current state of play. Participants in DE4A include public sector bodies, who see no legal basis for processing personal data via blockchain technology – consent is not viable in a public service where consent cannot be freely given, and legitimate interest is unavailable for public sector bodies. In the absence of a specific legal basis, blockchains would be usable for the DIDs of Data Providers and Data Consumers – where the working assumption to be validated is that these can be used without any personal data of any kind – and for schema information where needed, but not for any personal data. In that way, no GDPR compliance issues would be triggered on this point.
- The **VC contains both the MDS and any other personal data comprised in the diploma**, and therefore manifestly contains personal data; but this should not be problematic to the extent that it moves directly from Data Provider to the Wallet, and then to the Data Consumer. The DE4A consortium recognises that it is somewhat open to discussion whether this pattern can be construed as a once-only exchange in the interpretation given by the SDGR, given the intervention of the user as a data holder between Provider and Consumer; but given the clear ancillary benefits of this pattern and the fact that the DE4A consortium is not restricted to implementing the SDGR, this is not considered a blocking point.

[2] Except in the case of Slovenia, which has not yet notified an eID and therefore cannot pilot on the basis of notified eIDs. In the Slovenian case, non-notified eIDs will be used, purely for piloting purposes, using eIDAS pre-production nodes.

[3] Proposal for a Regulation of the European Parliament and of the Council amending Regulation (EU) No 910/2014 as regards establishing a framework for a European Digital Identity; see https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=COM%3A2021%3A281%3AFIN

- With respect to the **use of DID signing rather than eIDAS recognised eSeals,** the main driver for this plan is that Data Providers at any rate have no structural approach in place for sealing VCs (qualified or nonqualified). As a proxy for qualified eSeals, the use of DIDs as a sealing mechanism for VCs is driven by the consideration that a list of verified/trusted issuers is planned to be kept on the ledger, so that the ledger can act as a *de facto* trust list for competent issuers. While that trust list has no legal value as such, it would be sufficient and adequate to pilot the concept. Moreover, the use of qualified eSeals would not solve the problem comprehensively either: while it would allow easy recognition of the eSeal as such, it wouldn't address the challenge of verifying whether the qualified eSeal indeed belongs to a competent issuer. The latter approach requires a separate trust list, for which the DIDs on the ledger would be used. Future iterations may combine the approach (i.e. using qualified eSeals, with the competence of the certificate holders being verifiable by crosschecking it with the DIDs on the ledger).

There are of course still some legal challenges. At the highest level, the **legal validity of the VC (and the question whether it qualifies as a 'diploma' for the purposes of real-life administrative proceedings) depends on national law. Based on current practices, a VCs's legal validity as evidence under the SDGR seems questionable.** The eIDAS Regulation introduced a horizontal non-discrimination rule for eDocuments, but this principally implies that legal validity of an electronic document cannot be denied merely *because* it is issued in an electronic form. That will however not be the legal problem; in DE4A, the main challenge is that likely no Data Provider is issuing VCs as legally valid evidence at this stage. The VCs can therefore be considered as *representations of evidences*, but not as *evidences* in the sense of the SDGR in their own right. As a result, the pilot is largely a proof of concept showing that this interaction pattern *can* work, rather than demonstrating that it is *already* compliant under current law. This issue is independent of the signing/sealing method used.

For the avoidance of doubt: even if the VC cannot legally be considered to be a diploma (and therefore 'evidence' in the sense of the SDGR) as such, that does not imply that piloting cannot take place, or that real persons cannot be involved in any piloting. It merely implies that the exchange of VCs is not a part of Data Provider's regulated task of providing evidences as targeted by the SDGR, and that it requires a separate justification. Piloting could still occur, even with real persons (including real students), on the basis of the consent of these persons in accordance with the requirements of the GDPR. This implies notably that the research participants are adequately informed of the fact that they are participating in a pilot, and that appropriate measures are taken to avoid undesired consequences. Mainly, if the exchange is indeed principally a proof of concept without real-life operational implications (i.e. no actual registration in a university should happen), then it should be clear to all participants that no legal consequences should occur as a result of participating in the pilot.

Beyond that, there are still more basic security and compliance challenges. Notably, a **link needs to be established between the VC in the Wallet, and the eIDAS identity of the claimed diploma holder**.

It is currently envisaged that the DID links between on the one hand the edge agent (the phone), and on the other hand respectively the Data Provider and Data Consumer, are established by scanning a QR code which is shown during an eIDAS authenticated session with the Data Provider, respectively the Data Consumer. Thereafter, the VC is transferred to the Wallet, and can be made available to the Data Consumer.

This approach presents a security risk, since other persons than the eIDAS holder could scan the QR code of the Data Provider while it is shown on the screen; and furthermore, the link to the eIDAS identity and the VC is lost after the scanning process – any party who has access to the mobile device and the app could potentially access and use the VC by showing it to a Data Consumer. That opens the risk of credential fraud.

This risk can be mitigated by:

- user education and proper interface design, i.e., warning users to be cautious about over-the-shoulder attacks before showing the QR code;
- the integration of the MDS in the VC. In that way, a third party who unlawfully obtained access to the VC via an unlawfully established DID would not be able to use it towards a Data Consumer in an eIDAS authenticated session (which is the baseline set by DE4A at this stage): while the "VC thief" could successfully authenticate with their own eIDAS eID towards the Data Consumer, the verification by the Data Consumer would trigger an error, since the MDS of the eIDAS eID would (with extremely high likelihood[4]) not match the MDS in the VC.
- Potentially encrypting the VC with an encryption key based on the eIDAS credential itself when the VC is issued by the Data Provider, so that the VC would not be usable towards a Data Consumer until it is successfully decrypted using the same eIDAS credential that was used towards the Data Provider. This measure is still under examination but could recreate a link between the eIDAS eID and the VC.
- Limiting the validity period of the VCs in the Wallets to mitigate the chance of fraudulent use. While this somewhat limits the utility of the VCs as well, the objective of DE4A as a pilot project is to facilitate eGovernment transactions, and not necessarily changing data sovereignty paradigms in a way that permanently transfers control over official credentials from public administrations to citizens. Given that these transactions generally should happen in relatively short timespans, shorter validity periods are appropriate functionally and as a risk mitigation measure.

These measures would not resolve all legal challenges – notably and principally, they are unable to address the problem that the SDGR requires the exchange of 'evidences', and that VCs cannot be considered as 'evidences' unless they are lawfully issued as such by Data Providers under national law. Nonetheless, the model seems suitable to demonstrate the viability of the model in cases where national laws and practices have been adapted to enable VC issuance as legally valid evidence.

[4] That is to say: with the exception of the highly unlikely scenario that both the legitimate VC holder and the unlawful user would have the exact same MDS. Beyond being extremely unlikely, one might also raise the objection that the same scenario would actually be *easier* to perform with a paper diploma, which often contains less information than the MDS, so even with the acceptance of this risk the process would still be significantly safer than analogue procedures.