Student and Citizen Identities Linked

D3.3 Technical documentation on web and mobile user interfaces

Document Identification			
Status	Final	**Due Date**	29/05/2020
Version	1.0	**Submission Date**	03/06/2020

Related Activity	Activity 2	**Document Reference**	D3.3
Related Deliverable(s)	D2.3, D3.2	**Dissemination Level**	PU
Lead Organization	UMA	**Lead Author**	Victoriano Giralt
Contributors	Laura Domínguez Manuel Baleriola Alberto Basurte Antonio Campos Raúl Ocaña	**Reviewers**	GRNET UJI

Keywords:
eIDAS, eduGAIN, eMRTD ePassport, Interface, Authentication, Identity Provider

This document is issued within the frame and for the purpose of the SEAL project. This project has received funding from the European Union's Innovation and Networks Executive Agency – Connecting Europe Facility (CEF) under Grant AGREEMENT No INEA/CEF/ICT/A2018/1633170; Action No 2018-EU-IA-0024. The opinions expressed, and arguments employed herein do not necessarily reflect the official views of the European Commission.

Document Information

List of Contributors
Partner
ATOS
UJI

Document History			
Version	Date	Change editors	Changes
0.1	27/04/2020	UMA	
0.2	29/05/2020	UMA	Deprecated authentication methods Description of the available operations GitHub URL references added
1.0	03/06/2020	ATOS	Quality review and submission to EC

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	UMA	29/05/2020
Technical manager	UJI	25/05/2020
Quality manager	ATOS	03/06/2020
Project Manager	ATOS	03/06/2020

Table of Contents

List of Figures

List of Acronyms

Abbreviation / acronym	Description
API	Application Programming Interface
BAC	Basic Access Control
CEF	Connecting Europa Facility
DID	Decentralised Identifier
DUMA	Directory of University of Málaga
eduGAIN	Education Global Authentication Infrastructure
eIDAS	Electronic Identification Authentication and trust Services
eMRTDs	Machine-Readable Travel Documents
EU	European Union
IdP	Identity Provider
LGPL	Lesser General Public License
NFC	Near Field Communication
OCR	Optical Character Recognition
PDS	Personal Data Store
SEAL	Student and Citizen Identity Linked
SSI	Self-Sovereign Identity
uPort	Secure, privacy preserving data-sharing infrastructure (https://www.uport.me/)
VC	Verifiable Credential

1. Introduction

The SEAL project has developed a dashboard application that can be accessed both from a web browser and from a specialised mobile application. The dashboard has the following functionalities:

- Login to SEAL service
- Selection of identity source
- Selection of storage backend
- Identity management, presentation and, derivation
- Identity linking

1.1. Purpose of the document

This deliverable provides a technical description of the dashboard implemented for user management of the different identities handled in SEAL.

1.2. Relation to another project work

This work is carried out as part of the SEAL Activity 3 to produce a SEAL Dashboard for managing linked identities.

It is the first deliverable of SEAL that makes use of the Services through the API gateway.

1.3. Structure of the document

This document is structured in 2 major chapters

Chapter 2 presents the SEAL dashboard design and functionalities.

Chapter 3 presents the Atos eMRTD Reader Android App software release note.

1.4. Scope

This document has a dual purpose, to serve as technical documentation and as the user guideline. This is a first version which focuses on the Web Dashboard including the web GUI operations and their screenshots. The second version will describe the SSI and it will explain in detail the mobile Dashboard app.

2. SEAL identity management dashboard

The SEAL Dashboard uses the SEAL API Gateway client for accessing the various SEAL services and allowing users to interact with their identities.

The dashboard screens presented below show the design integrated as a part of the University of Málaga "DUMA" identity management panel, because it accelerates the design and implementation, although the dashboard itself can be deployed as a separate Django application.

The implementation team has followed a Test-Driven Development pattern in order to be independent of other developments in the project.

Using the SEAL dashboard as a part of a bigger Identity Management System, like DUMA, has the advantage of creating a simple way of linking identities to an already validated academic identity. It is a valid approach to most use cases except the enrolment one, in which there is no pre-existing academic identity in the receiving institution For this use case the user needs to access an instance of the dashboard that allows for presenting a valid identity to the receiving institution enrolment system.

2.1. Relevant APIs for the dashboard

The Figure 1 below shows the APIs and dataflows used by the dashboard.

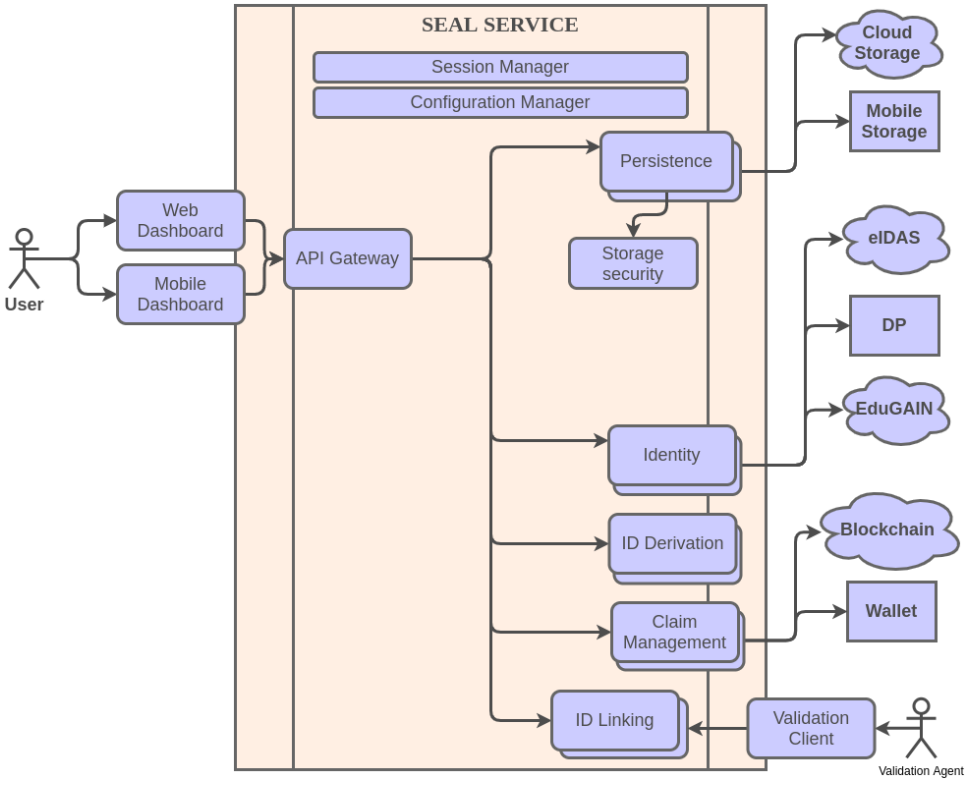

Figure 1 API Dataflow

2.2. Dashboard login use cases

The Figure 2 and Figure 3 below present the flows for the different authentication use cases for the user in order to get access into the dashboard. This is the first step for the management of the retrieve identities.

2.2.1. Dashboard PDS access

This authentication method uses a PDS file, which could be found either locally or in the cloud. That file represents the encrypted user identity. The different methods for loading this file in the SEAL service vary based on where the file is saved. The following diagrams of use cases represent different process flows but reaching the same outcome, in which the user gets her/his identity proved and loaded in the SEAL service.

2.2.1.1. Use case diagrams

2.2.1.1.1. Local PDS access diagram

The application must require the user's password for the local PDS file access. The encrypted password will be sent together with the PDS file to the persistence microservice, which will verify that the password allows to decrypt correctly the PDS file, store it and associate it to the current user session.

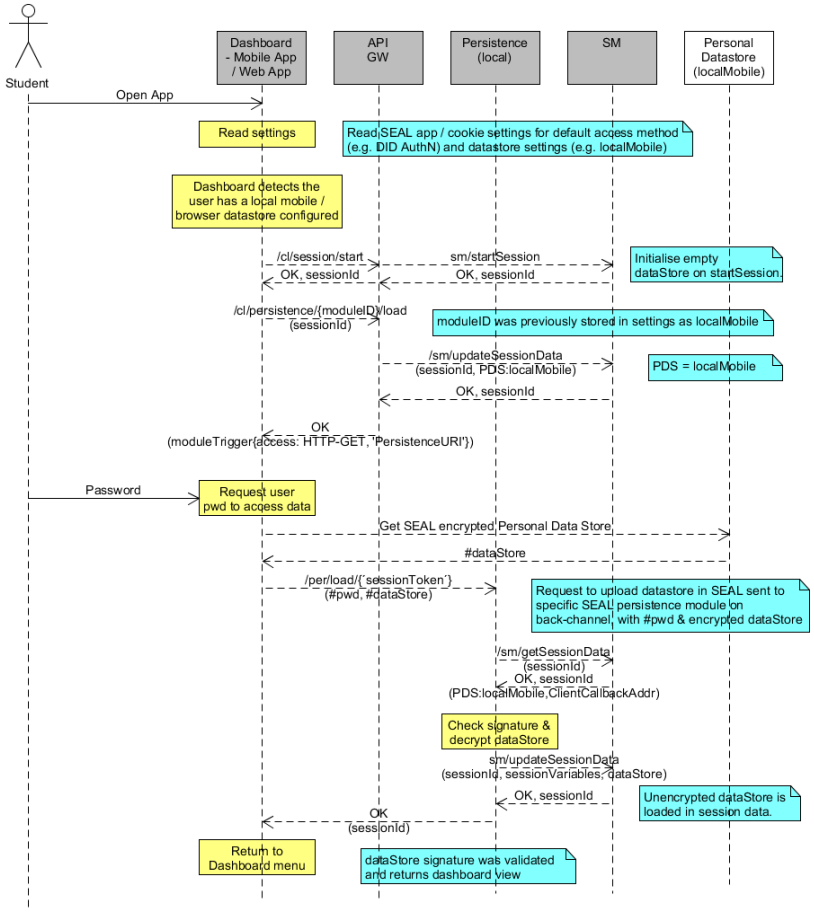

Figure 2 Use case 1.02 (local PDS access)

2.2.1.1.2. Cloud PDS access diagram

The cloud method allows the user to get the PDS from a cloud storage (either Google Drive or One Drive), in order to be loaded in SEAL. This process needs to redirect the user to the cloud and authenticate her/him into the platform. Finally, the persistence microservice automatically selects the PDS file from this specific cloud storage and loads it referred into the main SEAL service.

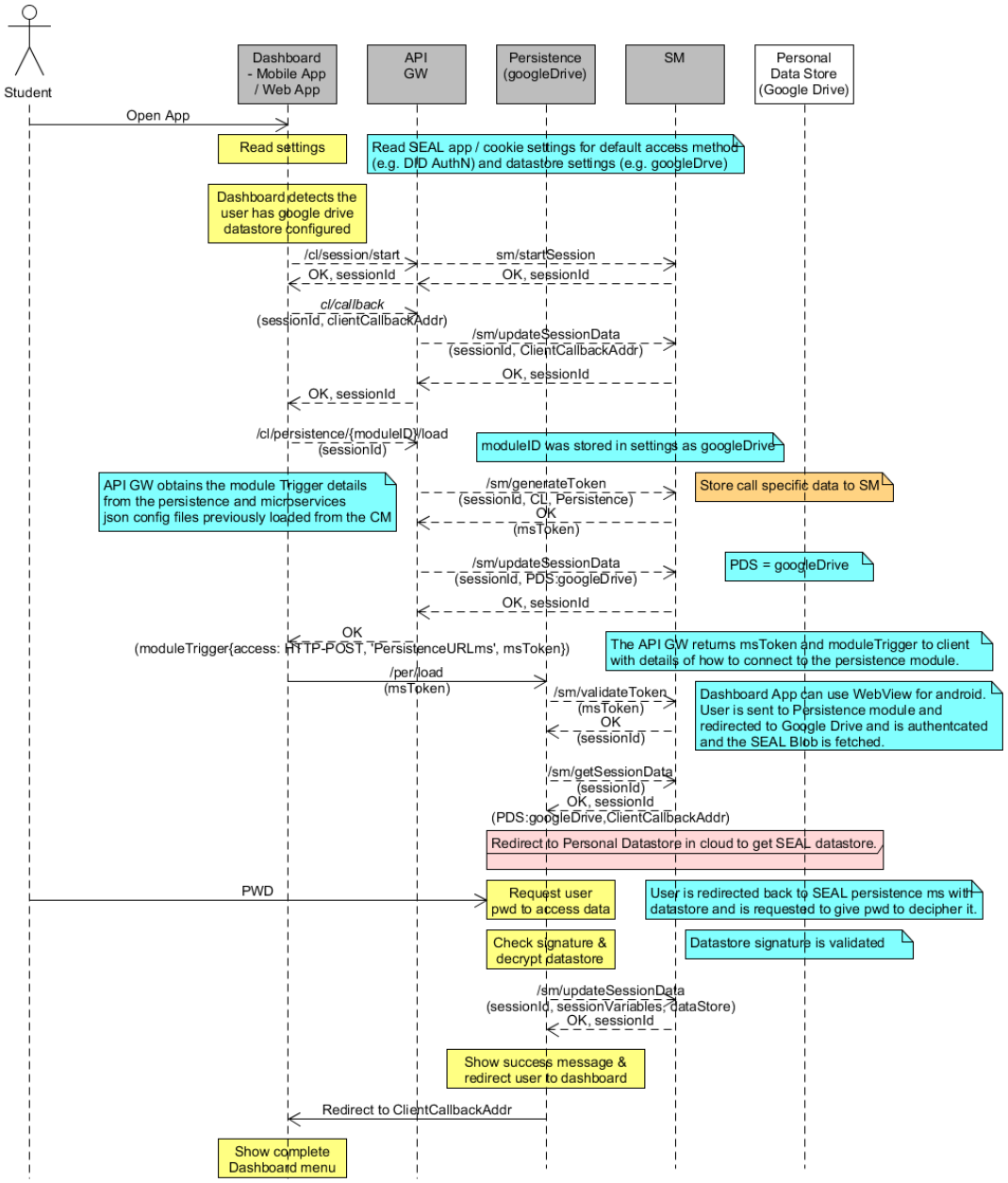

Figure 3 Use case 1.03 (cloud PDS access)

2.2.1.2. Integration details

Most of the needed requests for the correct local PDS file loading are API Gateway Client calls (cl), which is the main SEAL services. Nevertheless, there are also needed other API call methods for the persistence specific microservice (per). The loading and storing will be transparent and fulfilled, as well as independent from the different authentication methods.

Internally, the persistence microservice must communicate with the main service to update the session variables for the current user.

2.2.1.2.1. Integration of local load

2.2.1.2.1.1. Internal

- cl/session/start
- cl/persistence/{moduleID}/load
- per/load/{sessionToken}

2.2.1.2.1.2. External

- Ask for user password

2.2.1.2.2. Integration of cloud load

2.2.1.2.2.1. Internal

- cl/session/start
- cl/callback
- cl/persistence/{moduleID}/load
- per/load

2.2.1.2.2.2. External

- Cloud authentication (googleDrive or oneDrive)

2.2.1.3. Dashboard representation

The following steps represent the procedure for using the application in order to get the previously described behavior. Access options of users are the SSI and PDS methods, in other words, any user must select one of these two options to get her/his identity accredited and her/his access granted. When pushing the button for the PDS access method either a Local PDS or Cloud PDS load options will be displayed to the user, which are described in detail in the following paragraphs, as it is shown in Figure 4 and Figure 5.

Figure 4 Main screen of the access methods

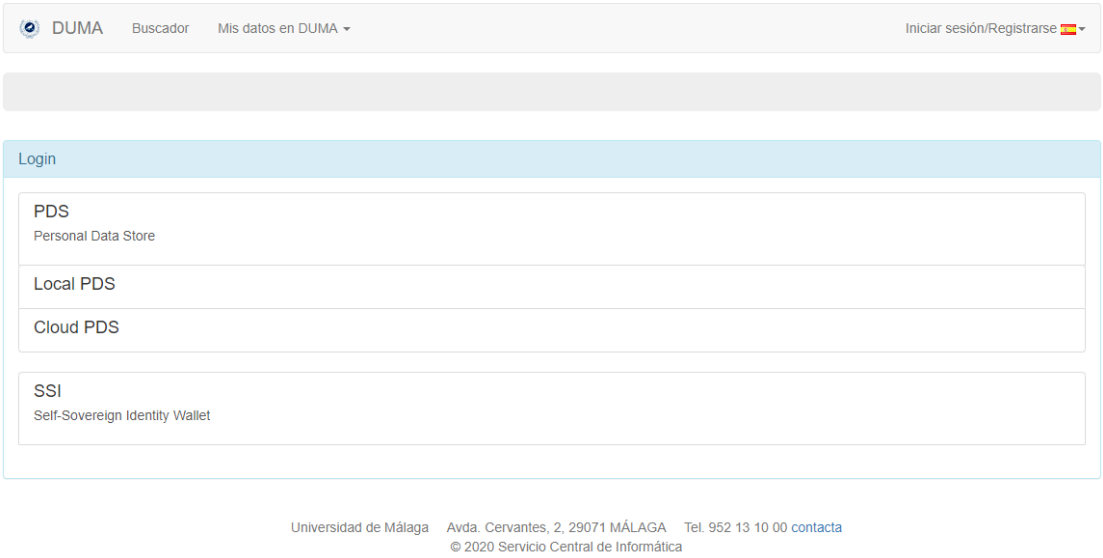

Figure 5 Dashboard screen of the PDS access load options (Local and cloud)

2.2.1.3.1. Local PDS representation

When the user selects this option, the application opens a new context window where the default path for the PDS file is displayed, as in Figure 6. Then, whether the file is in the default path or not, the user is able to select where the file is. Once the file has been selected, it is automatically loaded into the application, so the context window disappears, and it displays a message indicating that the process has been successful. In case the file selected is corrupted or the application is unable to read and loaded it, the context window will also be closed but an error code will be displayed instead.

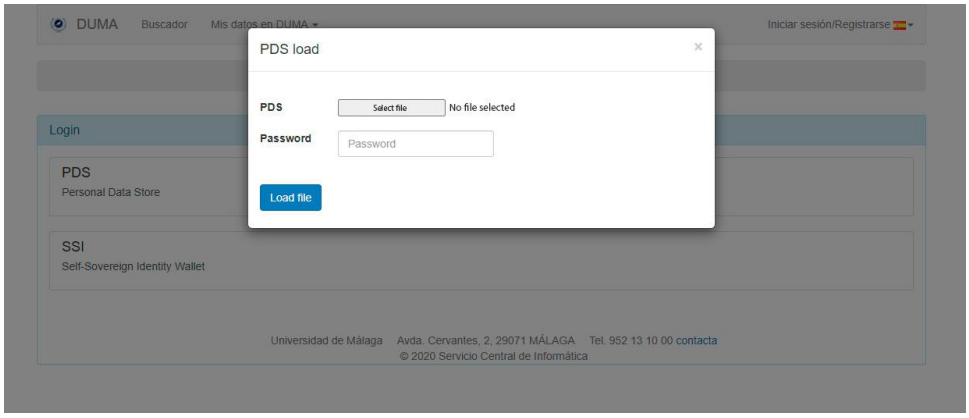

Figure 6 Local PDS contextual window

2.2.1.3.2. Cloud PDS representation

The user must choose one of the cloud platforms available: Google Drive or One Drive, after the selection the application will redirect her/him to the authentication login process in its platform (which may occur in a new context window, like the one in Figure 7[1]). When the authentication in the storage cloud happens successfully, then, the window referred will be closed and the service will transfer automatically the PDS file to the dashboard in order to be loaded in SEAL. Besides, the context window can be closed manually at any time, so we cover the cases in which the user does not authenticate successfully, or she/he eventually does not want to carry on with the cloud PDS method; after the ending the process the previous dashboard screen is displayed.

Figure 7 Cloud PDS contextual window

2.2.1.4. GitHub references

[1] The contextual window is currently under development, the final one may differ.

APIs specification could be found here: https://github.com/EC-SEAL/interface-specs

2.2.2. Dashboard app SSI access

To-do in version 2 of this document: It will be described for the use cases referred.

2.3. Dashboard Identity Manager

The initial dashboard screen is shown in Figure 8. The Identity Manager presents two functionalities: managing the identity data and configuring the data store. The "Save in Cloud Storage" link presented on the screen in Figure 8 below, changes according the storage selected via "Configure Data Store".

2.3.1. Integration details

The API functions will be specified in a new menu for each button when it is accessed, even though some calls could be done to the API if needed directly when buttons like 'Identify Reconciliation', 'Identify Reconciliation Status', or 'Derive Identifier' are clicked:

- cl/ident/linking/{moduleID}/request
- cl/ident/linking/{moduleID}/{requestId}/status
- cl/ident/derivation/{moduleID}/generate
- cl/persistence/{moduleID}/store

2.3.2. Dashboard representation

The list with the seven functionalities available in the Identity Manager dashboard is shown in Figure 9.

The first one is the Manage Identity Data option which shows the identity data loaded from the available storage into session and it also allows to add a new identity. The Configure Data Store option allows the user to select an available persistence module and use it to store identity data, required for saving in either the cloud or local storage. The Retrieve Identity Data option allows the user to obtain data from the available identity source and store them in the session storage. After the Identity Reconciliation functionality is presented, it displays a selector of available identity reconciliation procedures options, where the user can choose among them.

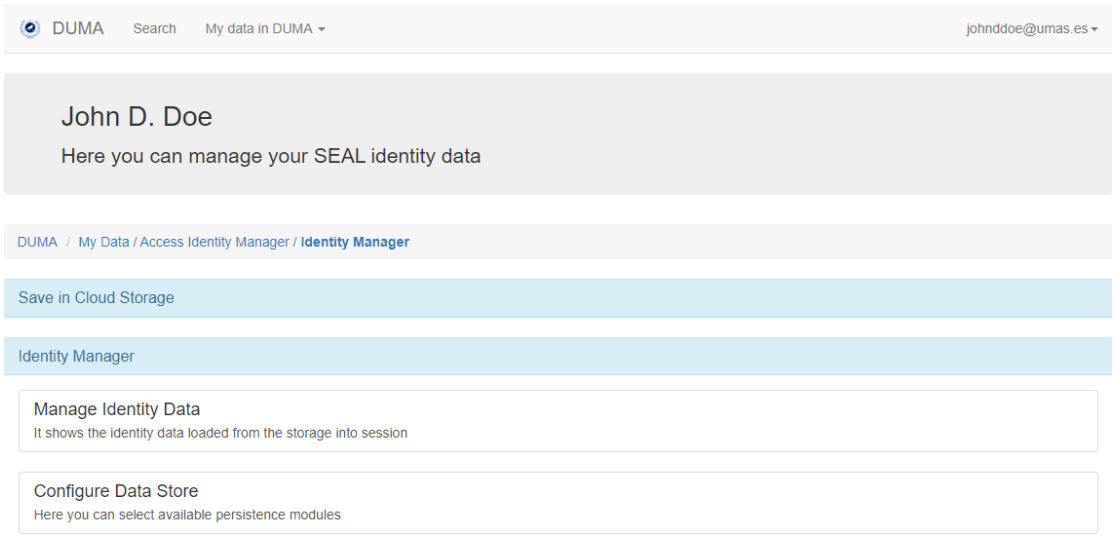

Figure 8 Access to the dashboard (integrated in UMA identity management web portal)

The Identity Reconciliation Status option allows the user to verify the result of a requested reconciliation procedure. Once this operation succeeds, the user can either save or cancel the generated data set.

The Manage Verifiable Claims functionality displays the available verifiable claim modules where the user can select a VC creation option, that will trigger a procedure to retrieve the data from the sources and/or the store, to build a VC that will be written on the user wallet. This option will only be available on the mobile interface and when there is a valid user wallet (which will have triggered a proper DID authentication process as soon as the user selects it as her/his authentication method).

The last functionality listed, the Derive Identifier, shows a selector where the user can select a method that will create a new identifier for the user and link it automatically to the authenticated identity (if the user has not authenticated in any source by now, the authentication procedure will be triggered). At the end of the procedure, a data set for the new identity and a data set for the link will be added to the store.

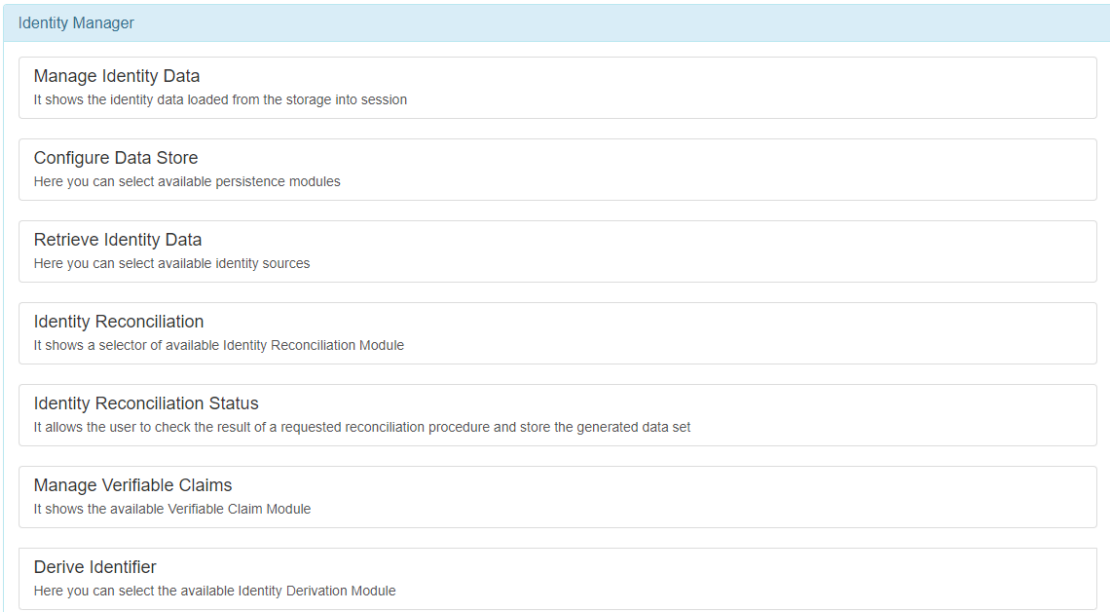

Figure 9 Access Identity Manager functionalities

2.3.3. GitHub references

Dashboard test and code could be found here: https://github.com/EC-SEAL/interface-specs

APIs specification could be found here: https://github.com/EC-SEAL/interface-specs

2.4. Identity Cards

An Identity Card represents the user's information that identifies a person in a specific digital environment. This identity card is unique and untransferable, and although a user can have a variety of different IDs, every time the source which provides an ID creates or updates a new one for the user, the previous ID comes 'out-dated' so the user can only have one unique ID per source.

2.4.1. Integration details

- cl/ident/source/{moduleID}/retrieve
- cl/ident/mgr/list
- cl/ident/mgr/{datasetID}/refresh
- cl/ident/mgr/{datasetID}/delete

2.4.2. Dashboard representation

Figure 10, Figure 11 and Figure 12 show three examples of Identity Cards that can be retrieved and managed by the user in the Identity Manager. Identities are presented in card style sorted by tabs. Tabs, referring to the main organisation or mechanism that provided the ID, are created when a new ID is retrieved. If an identity provided by one of the mechanisms already exists, then the ID and the date of retrieval will be loaded in the appropriate tab. Similarly, an ID can be updated by refreshing (blue button) or deleted (red button). When an ID is deleted, it will be removed together with the corresponding tab containing it. The *"New ID"* button is a handy shortcut for getting to the identity sources screen, for selecting a new Identity Source.

Differences in content among ID Cards are the result of having different sources to identify the user, as it will be provided by each organisation. Thus, the current benefit of this service is to reconcile these IDs, so the user can authenticate in other services with an identity and receive trusted info relating it to another one existing in, for example, an imported data set. If a trusted entity establishes a link between two identities (with a certain Link Level of Assurance), any consumer can have certainty (to the awarded level) that both identities belong to the same individual.

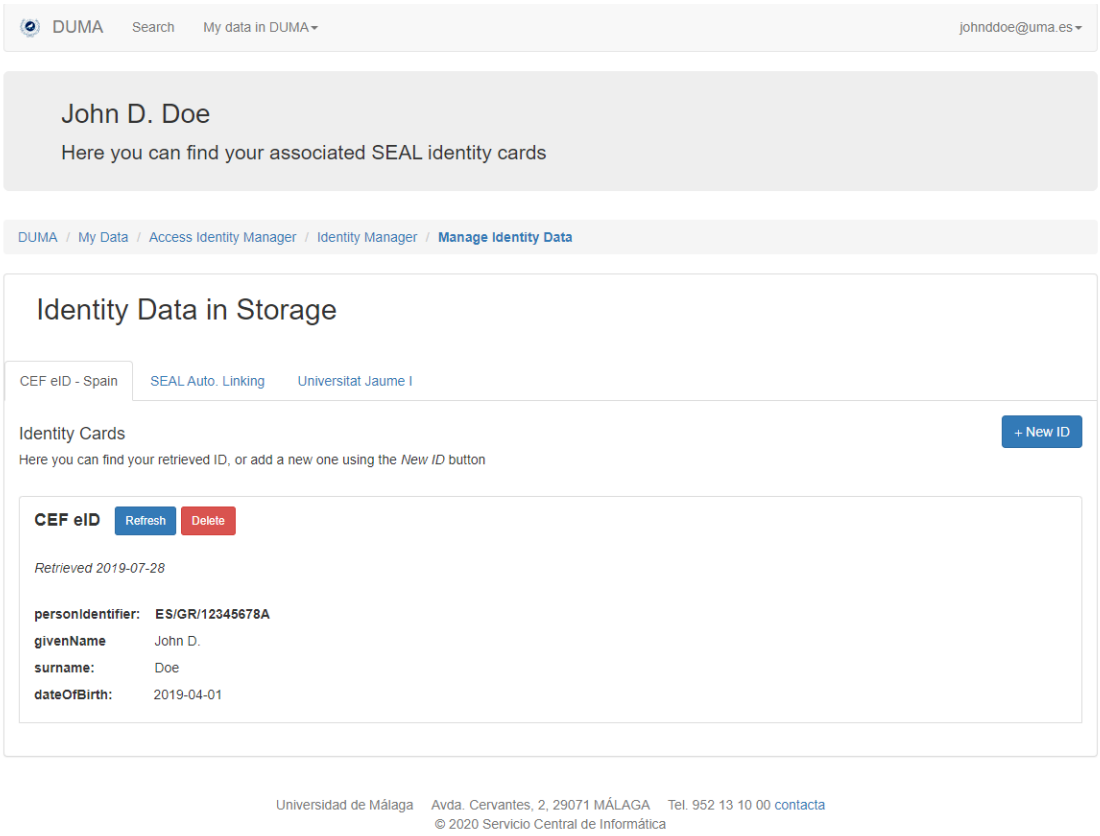

Figure 10 Identity Data Manager – CEF eID

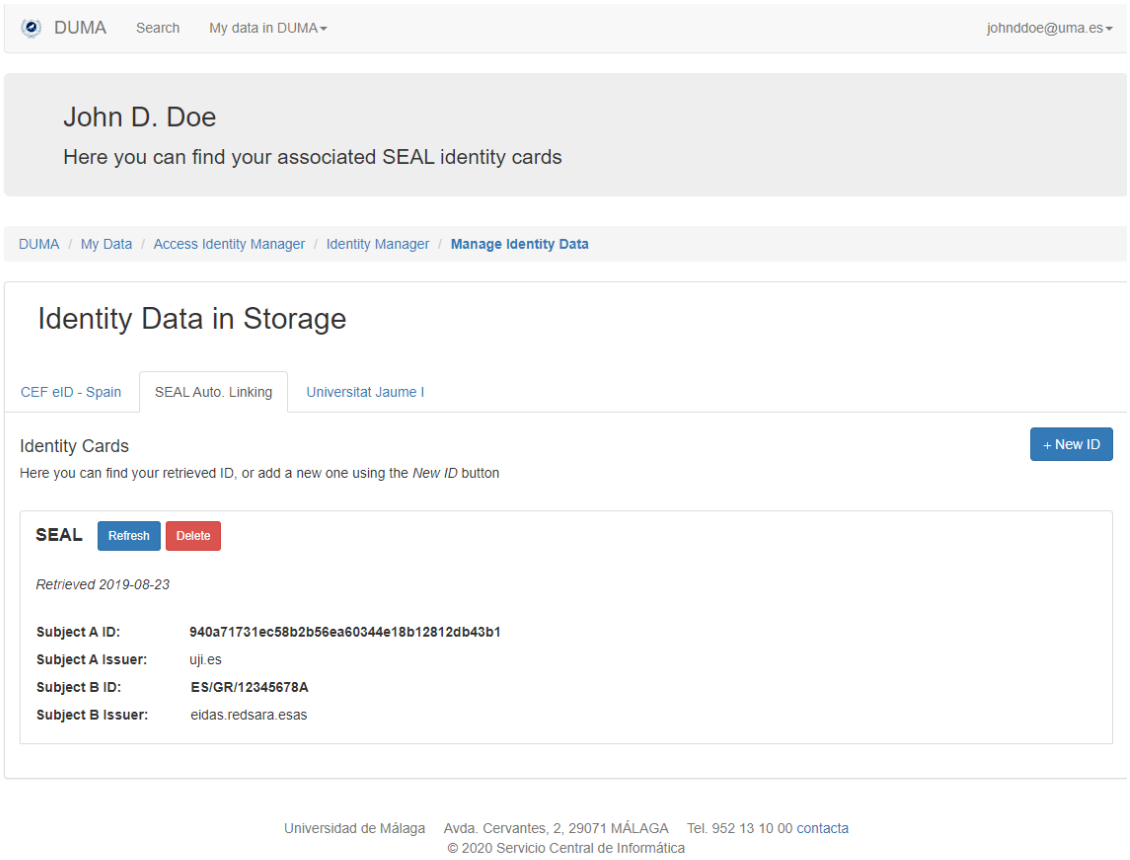

Figure 11 Identity Data Manager – SEAL

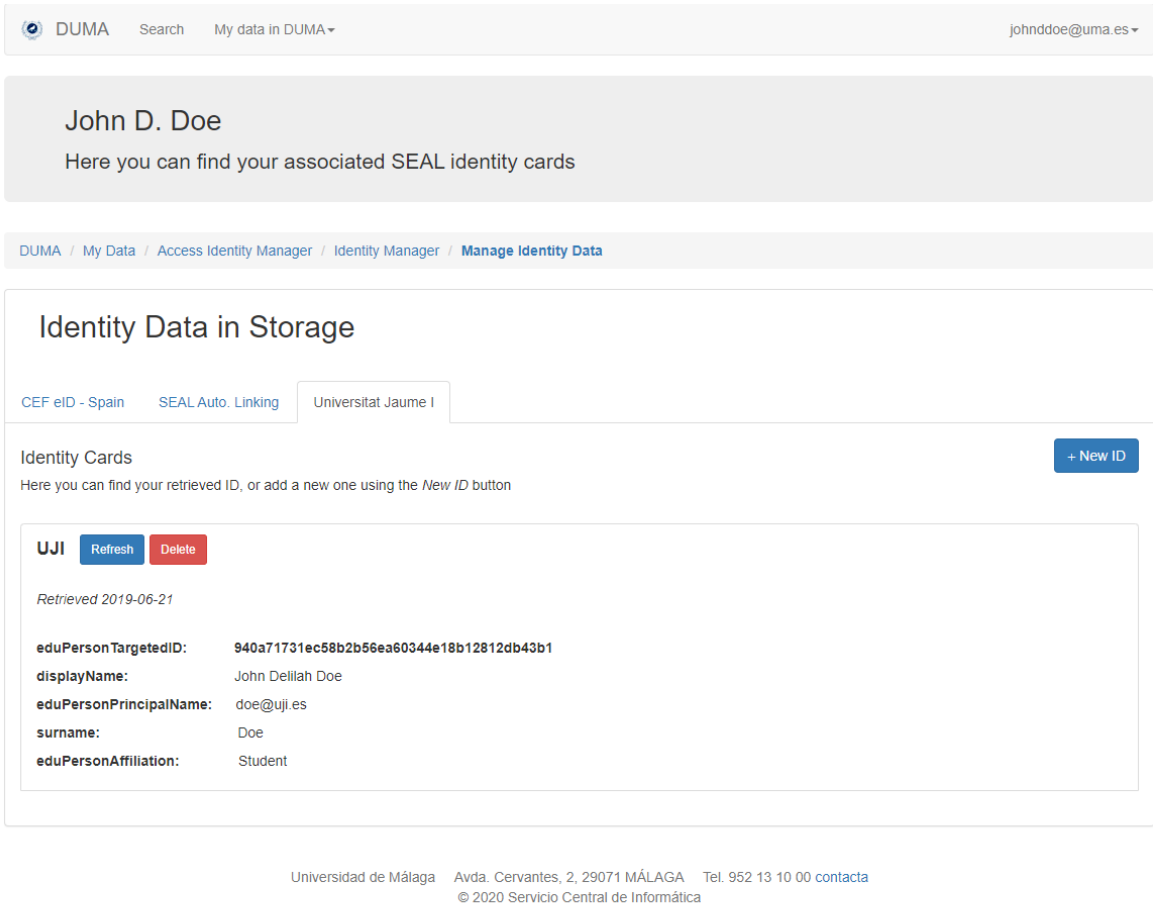

Figure 12 Identity Data Manager – UJI

2.4.3. GitHub references

APIs specification could be found here: https://github.com/EC-SEAL/interface-specs

2.5. Mobile Dashboard

The mobile dashboard is an adaptation of the web dashboard app for the different sorts of mobile devices. So, with that in mind, the same functionalities are available for the user with the possibility of making even more value when using with a linked valid mobile wallet. This can trigger the DID authentication process automatically when the user selects this option in the mobile dashboard.

2.5.1. Integration details

The integration functions used in this mobile dashboard are the same of those in the web dashboard. This is possible as the internal calls from the API to the SEAL service are identical, so the only change is the programming layout, but the core is still immutable. Also, the Atos eMRTD reader is integrated in the mobile dashboard, it is described in detail in chapter 3.

2.5.2. Dashboard representation

While the figures in the previous sections show the web implementation of the identity manager application, Figure 13 and Figure 14 show the first and the second screens found in the pursuit of accessing the Identity Manager by the UMA Mobile App, hereinafter called UMA App. Figure 13 shows the displayed menu after pushing "*My Info*" button, located in the main screen of UMA App. The second screen is displayed in figure 13. It draws up a list of the different authentication methods available to access into Identity Manager.

Again, embedding the dashboard into an App that is already linked to a valid academic identity, facilitates the heuristics for validating other identities. The modularity of the UMA App allows for the easy creation of an independent self-contained SEAL dashboard mobile App.

The mobile App is integrated with an SSI wallet (uPort) in order to be able to generate Verifiable Claims for it from the data in the app. When accessing the app, it will try to perform a DID authentication to securely bind the session in the app with the wallet where the user can choose to write VCs.

Functionalities for reading NFC enable to include government issued identification objects such as Spanish DNIe or ePassport. The app is not trusted, so when the data is sent to the server side, its original signature will be validated.

Besides, the App can be used as a backend storage in a mobile device, but for security reasons, data is managed on the server side and is only written on the device after it has been signed to preserve its integrity.

Data transport will happen over the relevant API calls.

The following screens in Figure 13 and Figure 14 use the same APIs and at the same order than its fellows at Web App, so it is not necessary to recall them to the explanatory purpose of the images.

Figure 13 UMA App menu to access Identity Manager (access methods screenshot to follow

(image to follow in version 2 of this document)

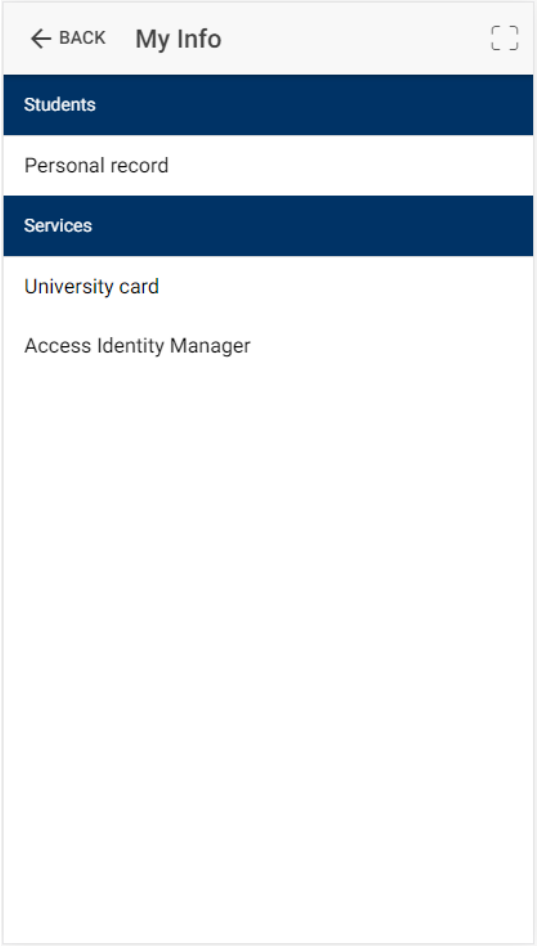

Figure 14 UMA App menu to access Identity Manager

Figure 15 shows the Identity Manager in detail. The left side of the figure displays the image of the Identity Manager accessed by the Access Identity Manager screen. As it can be seen it includes all the available options for the UMA mobile app. These options are presented in two columns and comprise all the functionalities described for the web Identity Manager dashboard. The image in the right side of Figure 15 presents the Manage Identity Data functionality. It displays the list of the user's identity cards.

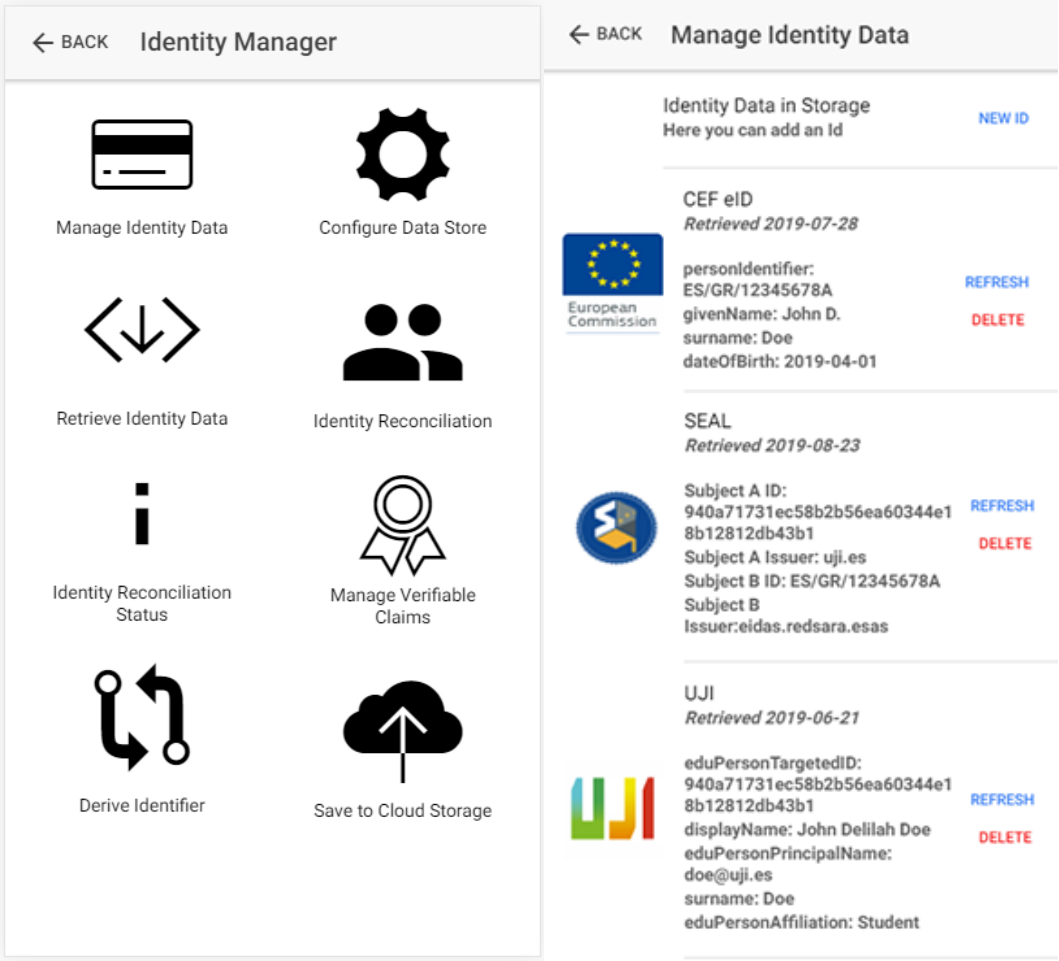

Figure 15 Identity Manager menu and Identity Cards

2.5.3. GitHub references

Dashboard test and code could be found here: https://github.com/EC-SEAL/dashboard

3. Atos eMRTD Reader Android App software release note

3.1. Introduction

This software provides the electronic reading of contactless electronic Machine-Readable Travel Documents (eMRTDs) as specified by ICAO 9303[2] standard for the reading of ePassports and national IDs. The Android app reads the eMRTD chip to access the biographic Data Groups to retrieve the user´s personal data (Name, DoB, Nationality etc.), and also the passport image.

The access to the data is protected by BAC authentication where it is necessary to first scan the printed Machine Readable Zone (MRZ), before accessing the contactless chip to retrieve the user´s biographic data and biometric face image

3.2. Software Release Notes

The released AtosReader App is an Android application with two main utilities:

- **An OCR Reader**: The app scans the MRZ of an ePassport or nationl eID using the camera on an android phone. We use the ML KIT(https://firebase.google.com/docs/ml-kit/recognize-text) for MRZ text recognition. In the case of ePassports it only reads the second line, whereas in the case of the national eIDs the two first lines of the MRZ are read.
- **NFC Reader**: the application connects via NFC with a passport or a national eID, and reads from the electronic document the biographic and face image data groups from the electronic document. For developing this part, the base code used the JMRTD library (https://jmrtd.org/) with LGPL license(https://jmrtd.org/license.shtml)

[2] https://www.icao.int/publications/pages/publication.aspx?docnum=9303

3.3. Supported attributes

The list of eMRTD attributes for the reading of ePassports and national eIDs supported by SEAL application are listed below, and are referenced on the SEAL GitHub repository[3]


```
{
        AttributesList:
    [
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#DocumentCode",
         "friendlyName":"DocumentCode",
         "encoding": "UTF-8",
         "isMandatory": "true",
         "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#IssuingState",
         "friendlyName":"IssuingState",
         "encoding": "UTF-8",
         "isMandatory": "true",
         "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#DocumentNumber",
         "friendlyName":"DocumentNumber",
         "encoding": "UTF-8",
         "isMandatory": "true",
         "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#DateOfExpiry",
         "friendlyName":"DateOfExpiry",
         "encoding": "UTF-8",
         "isMandatory": "true",
         "values": "string"
```


[3] https://github.com/EC-SEAL/conf-manager/blob/development/src/test/resources/attributeLists/eMRTD.json

```
    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#GivenName",
        "friendlyName":"GivenName",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": ["string"]

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#Surname",
        "friendlyName":"Surname",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": ["string"]

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#Nationality",
        "friendlyName":"Nationality",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#DateOfBirth",
        "friendlyName":"DateOfBirth",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#Sex",
        "friendlyName":"Sex",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": "string"

    },
        {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#PlaceOfBirth",
```

```
        "friendlyName":"PlaceOfBirth",
        "encoding": "UTF-8",
        "isMandatory": "true",
        "values": "string"

    },
                {
                "name":
"https://www.icao.int/publications/Documents/9303_p10_cons_en.pdf#FaceImage",
        "friendlyName":"FaceImage",
        "encoding": "Binary",
        "isMandatory": "true",
        "values": "string"

    }
       ]

}
```

4. Conclusions

The Test Driven Development methodology allowed us to have a very fast development cycle, decoupled from the development of other parts of the SEAL architecture, thanks to the clear definition of the APIs.

We have found that it is of paramount importance to have a persistence layer API that decouples SEAL applications and services from the diverse storage services available.

We expect that further testing of the dashboard will influence the refining of the APIs and allow for a clear definition of the storage API needs.

5. References

[1] SEAL D3.2 Technical documentation on modular interfaces for different types of identities

[2] SEAL D2.3 Web and mobile user interfaces design