Impulse

Identity Management in PUbLic SErvices

D2.10 Implementation of basic system V2

Lead Author: Xavier Martínez (GRAD)
With contributions from: Jesús Prieto González (GRAD), and all public administrations (ARH, ERTZ, GIJON, MOP, RVK, UC/IC)
Reviewer: Kais Dai (TREE), Iñaki Gangoiti (ERTZ)

Deliverable nature:	Demonstrator (D)
Dissemination level: (Confidentiality)	Public (PU)
Delivery date:	31-05-2023
Version:	2.0
Total number of pages:	31
Keywords:	Configuration, deployment, instantiation, integration, interaction

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004459

Executive summary

This document summarizes the instantiation of the identity management service-oriented basic system of IMPULSE V2. This includes all the integrations that a public administration has to perform to connect one of their services to IMPULSE, the configuration of the deployable component of the solution, known as the Enterprise Service, and the deployment of the component itself. It also describes the interaction of a Public Servant with the dashboard of the Enterprise Service, and a brief summary of the different sessions that were held with the public administrations regarding the instantiation process.

The IMPULSE solution, as many other eID systems, has two main processes: the onboarding and the authentication. The onboarding process is performed by the natural persons who want to obtain and store an identity verifiable credential in their devices. The users can initiate this process by selecting their public administration from a list, reading a QR code, or clicking a deep link. The two last options of initiating the onboarding have to be integrated by the public administrations if they really want them to be available for the citizens, as they bring a better user experience. The authentication process is performed by the natural persons that want to authenticate themselves against a public administration to receive a service. In this case, it is mandatory that the public administration service has integrated one or both of the previously commented methods: reading a QR code or clicking a deep link. In this process, it also necessary that the public administration service is able to associate an attribute with the current user session of the web browser. For this service to know which user and which web browser to provide the service to after a successful authentication, it is necessary to expose an endpoint where it will receive an id token related to the user authenticated information.

To facilitate the whole instantiation process, a configuration tool that aims to automate the configuration and deployment of the Enterprise Service has been developed. This tool allows to configure general Enterprise Service information, trusted issuer information, Transport Layer Security, logging modes, the public servant dashboard, the face matching service, the document verification service, the MRZ reader service, the remote qSeal service, the Azure Key Vault, and the outbound proxy. As the Enterprise Service is shared as a docker image, the configuration tool is contained within the same image, and it is also able to generate a valid docker run command to ease the generation of the container. Finally, this tool is also able to refresh the EBSI Verifiable Authorization of the configured trusted issuer, which is the verifiable credential used to receive permission to write information into the EBSI ledger.

Document information

Grant agreement No.	**101004459**	**Acronym**	**IMPULSE**
Full title	**Identity Management in PUbLic SErvices**		
Call	DT-TRANSFORMATIONS-02-2020		
Project URL	https://www.impulse-h2020.eu/		
EU project officer	Giorgio CONSTANTINO		

Deliverable	**Number**	D2.10	**Title**	Implementation of basic system
Work package	**Number**	WP2	**Title**	Co-creative design and piloting
Task	**Number**	T2.4	**Title**	IMPULSE basic system instantiation

Date of delivery	**Contractual**	M28	**Actual**	M18
Status	version 2.0		☐Final version	
Nature	☐Report	☒Demonstrator	☐Other	☐ORDP (Open Research Data Pilot)
Dissemination level	☒Public	☐Confidential		

Authors (partners)	GRAD, ARH, ERTZ, GIJON, MOP, RVK, UC/IC			
Responsible author	**Name**	Xavier Martínez Luaña		
	Partner	GRAD	**E-mail**	xmartinez@gradiant.org

Summary (for dissemination)

This document summarizes the instantiation of the identity management service-oriented basic system of IMPULSE V2. This encompasses the necessary integrations that a public administration must undertake to connect one of their services to IMPULSE. Additionally, it involves configuring the deployable component, referred to as the *Enterprise Service*, and deploying the component itself. It also describes the interaction of a Public Servant with the dashboard of the Enterprise Service, and a summary of the different sessions that were held with the public administrations regarding the instantiation process.

The IMPULSE solution, as many other eID systems, has two main processes: the onboarding and the authentication. The onboarding process is performed by the natural persons who want to obtain and store an identity verifiable credential in their devices. The users can initiate this process by selecting their public administration from a list, reading a QR code, or clicking a deep link. The two last options of initiating the onboarding must be integrated by the public administrations if they really want them to be available for the citizens, as they bring a better user experience. The authentication process is performed by the natural persons that want to authenticate themselves against a public administration to receive a service. In this case, it is mandatory that the public administration service has integrated one or both previously commented methods: reading a QR code or clicking a deep link. In this process, it also necessary that the public administration service can associate an attribute with the current user session of the web browser. For this service to know which user and which web browser to provide the service to after a successful authentication, it is necessary to expose an endpoint where it will receive an id token related to the user authenticated information.

To facilitate the whole instantiation process, a configuration tool that aims to automate the configuration and deployment of the Enterprise Service has been developed. This tool allows to configure general Enterprise Service information, trusted issuer information, Transport Layer Security, logging

	modes, the public servant dashboard, the face matching service, the document verification service, the MRZ reader service, the remote qSeal service, the Azure Key Vault, and the outbound proxy. As the Enterprise Service is shared as a docker image, the configuration tool is contained within the same image, and it is also able to generate a valid docker run command to ease the generation of the container. Finally, this tool is also able to refresh the EBSI Verifiable Authorization of the configured trusted issuer, which is the verifiable credential used to receive permission to write information into the EBSI ledger.
Keywords	Configuration, deployment, instantiation, integration, interaction

Version Log			
Issue Date	**Rev. No.**	**Author**	**Change**
10-03-2022	0.1	Xavier Martínez Luaña	First Integration section
30-03-2022	0.2	Xavier Martínez Luaña, Pablo Dago Casas	Updates on Integration section and new Deployment section
25-04-2022	0.3	Xavier Martínez Luaña	New Instantiation section describing the configuration tool and new deployment method
12-05-2022	0.4	Xavier Martínez Luaña	New Interaction and Refreshing Trusted Issuer Sections
29-06-2022	0.5	Xavier Martínez Luaña, PAs	Updates on several sections after feedback from public administrations
05-07-2022	0.6	Xavier Martínez Luaña, PAs	Updates on several sections after more feedback from public administrations
12-07-2022	0.7	Xavier Martínez Luaña	New instantiation Activities section
20-07-2022	0.8	Xavier Martínez Luaña, Jaime Loureiro Acuña	Added missing sections (executive summary, list of figures, abbreviations, definitions, conclusions and references)
21-07-2022	0.9	Jaime Loureiro Acuña	First revision of the document prior to official internal review
28-07-2022	1.0	Xavier Martínez Luaña, Gianluca Markos, Jakob Asmussen	Official internal review
17-04-2023	1.1	Jesús Prieto González	New configuration options and new figures
01-05-2023	1.2	Xavier Martínez Luaña, Jesús Prieto González	Updated integration and instantiation steps
03-05-2023	1.3	Xavier Martínez Luaña	Updated dashboard interaction and created the general deployment architecture section
31-05-2023	1.4	Kais Dai, Iñaki Gangoiti, Alicia Jiménez	Internal review
31-05-2023	2.0	Xavier Martínez	Final adjustments

Table of contents

List of figures

Abbreviations and acronyms

CPU: Central Processing Unit

DID: Decentralized IDentifier

ES: Enterprise Service

GB: GigaByte

HTTP: HyperText Transfer Protocol

IT: Information Technology

JKS: Java KeyStore

JSON: JavaScript Object Notation

JWT: JSON Web Token

PA: Public Administration

PAS: Public Administration Service

QR Code: Quick Response Code

RAM: Random Access Memory

TLS: Transport Layer Security

URL: Uniform Resource Location

Definitions

Docker: It is an open-source project to automate the deployment of applications inside software containers, providing an additional layer of abstraction and automation of virtualization across multiple operating systems.

Dockerization: It is the process of packing, deploying and running applications using Docker containers.

1 Introduction

One of the main aspects to be considered when implementing IMPULSE is how the solution will be instantiated from the point of view of a public administration. These legal entities have a huge variety of environments, with different technologies, and with very diverse IT teams. This raised the need to create a simple installation process that emulates a plug & play system, so it could be adapted to almost every public administration in a relatively simple way. As it is reflected in section 6, the main idea behind the basic system instantiation was adapted to the needs that we continue to encounter from these institutions. The two most important actions that we carried out to speed up this process were the dockerization of the Enterprise Service and the creation of the configuration tool within the same Docker image. Some technical details of both are shown below:

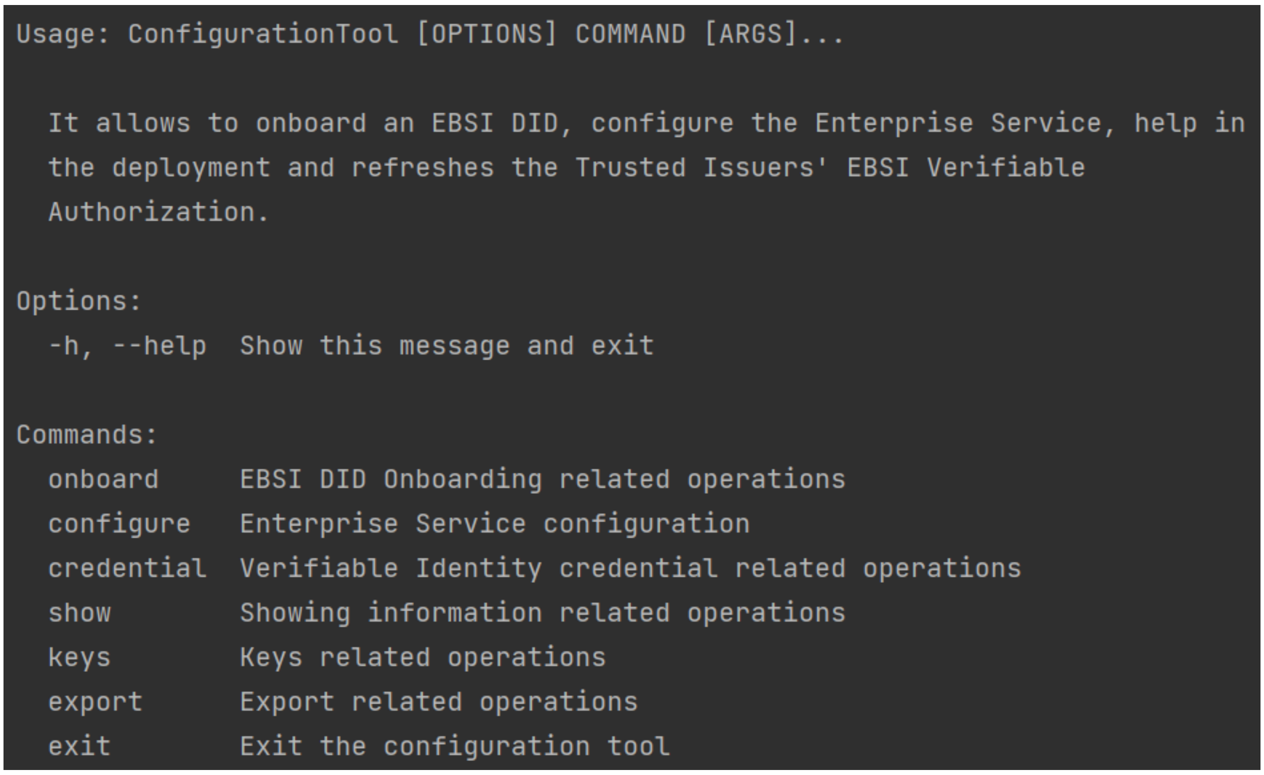
Dockerization of the Enterprise Service:

Docker image details:

- **Base image**: openjdk:16-slim-bullseye
- **Enterprise Service** added as a JAR
- **Common configuration files** added (service-matrix.properties, fsStore.conf, signatory.conf)
- **Entrypoint**: java –jar EnterpriseService.jar
- **Size**: 325MB
- **Version**: 2.0.1

Configuration Tool:

A Configuration Tool has been created to facilitate the public administrations the configuration and deployment of the Enterprise Service. It has been created as Command Line Interface (CLI) using the clikt library [1]. It is designed to be interactive and self-explanatory, i.e., everything that can be done with the tool is explained within the tool itself. This tool is included within the same docker image of the Enterprise Service to avoid any additional requirement.



```
Usage: ConfigurationTool [OPTIONS] COMMAND [ARGS]...

  It allows to onboard an EBSI DID, configure the Enterprise Service, help in
  the deployment and refreshes the Trusted Issuers' EBSI Verifiable
  Authorization.

Options:
  -h, --help  Show this message and exit

Commands:
  onboard     EBSI DID Onboarding related operations
  configure   Enterprise Service configuration
  credential  Verifiable Identity credential related operations
  show        Showing information related operations
  keys        Keys related operations
  export      Export related operations
  exit        Exit the configuration tool
```

Figure 1: Configuration Tool CLI

2 General Deployment Architecture

The next figure shows the general architecture of the IMPULSE solution from a deployment perspective. The Enterprise Service (ES), which is the component that orchestrates the whole system, is deployed as a Docker container exposing a series of endpoints intended to be used by the IMPULSE mobile application, the Public Administration Service (PAS) and a web browser in the PA domain. Each different component that interacts with the Enterprise Service can do it through different ports, facilitating the restriction of the different functionalities of this service to only the intended actors (e.g., internal port not exposed to internet, dashboard port only exposed to a specific machine).

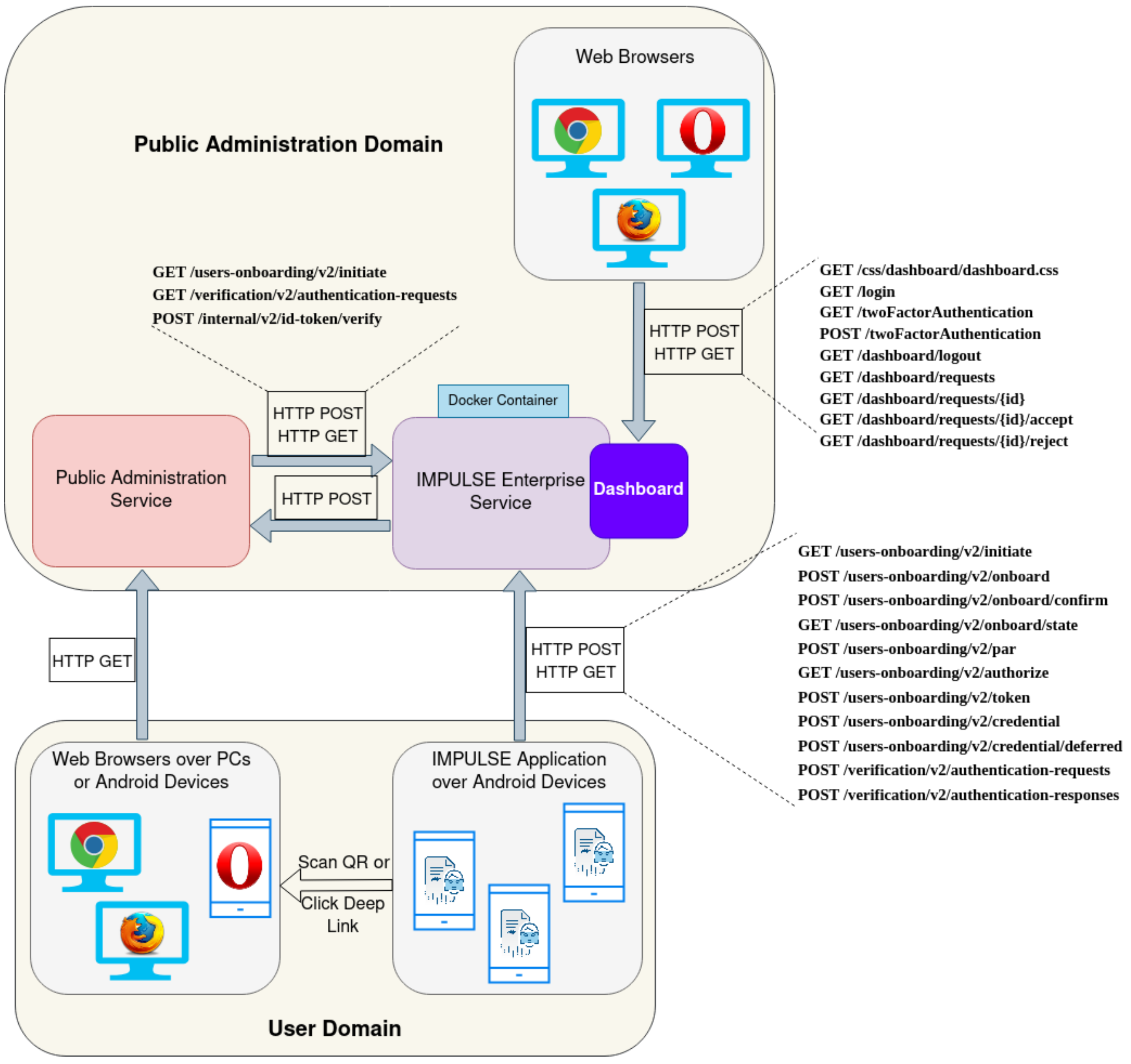

Public Administration Domain
Web Browsers
GET /users-onboarding/v2/initiate
GET /verification/v2/authentication-requests
POST /internal/v2/id-token/verify
HTTP POST
HTTP GET
HTTP POST
HTTP GET
GET /css/dashboard/dashboard.css
GET /login
GET /twoFactorAuthentication
POST /twoFactorAuthentication
GET /dashboard/logout
GET /dashboard/requests
GET /dashboard/requests/{id}
GET /dashboard/requests/{id}/accept
GET /dashboard/requests/{id}/reject
Docker Container
Public Administration Service
HTTP POST
IMPULSE Enterprise Service
Dashboard
HTTP GET
HTTP POST
HTTP GET
GET /users-onboarding/v2/initiate
POST /users-onboarding/v2/onboard
POST /users-onboarding/v2/onboard/confirm
GET /users-onboarding/v2/onboard/state
POST /users-onboarding/v2/par
GET /users-onboarding/v2/authorize
POST /users-onboarding/v2/token
POST /users-onboarding/v2/credential
POST /users-onboarding/v2/credential/deferred
POST /verification/v2/authentication-requests
POST /verification/v2/authentication-responses
Web Browsers over PCs or Android Devices
IMPULSE Application over Android Devices
Scan QR or Click Deep Link
User Domain

Figure 2: General Integration/Instantiation of the IMPULSE solution

It is also possible to configure a bidirectional authentication mechanism based on API Keys in the Enterprise Service:

- One API Key to ensure the PAS is the only component that can interact with the internal endpoint of the ES.
- One API Key to ensure that the ES is the actual component sending information to the exposed endpoint of the PAS.

2.1 Endpoints

IMPULSE Enterprise Service endpoints to be invoked by the Public Administration Service:

- GET /users-onboarding/v2/initiate
- GET /verification/v2/authentication-requests
- POST /internal/v2/id-token/verify

IMPULSE Enterprise Service endpoints to be invoked by the Dashboard Operator Web Browser:

- GET /css/dashboard/dashboard.css
- GET /login
- GET /twoFactorAuthentication
- POST /twoFactorAuthentication
- GET /dashboard/logout
- GET /dashboard/requests
- GET /dashboard/requests/{id}
- GET /dashboard/requests/{id}/accept
- GET /dashboard/requests/{id}/reject

IMPULSE Enterprise Service endpoints to be invoked by the IMPULSE User Application:

- GET /users-onboarding/v2/initiate
- POST /users-onboarding/v2/onboard
- POST /users-onboarding/v2/onboard/confirm
- GET /users-onboarding/v2/onboard/state
- POST /users-onboarding/v2/par
- GET /users-onboarding/v2/authorize
- POST /users-onboarding/v2/token
- POST /users-onboarding/v2/credential
- POST /users-onboarding/v2/credential/deferred
- POST /verification/v2/authentication-requests
- POST /verification/v2/authentication-responses

3 Integration with a Public Administration Service

From the public administration perspective, the Enterprise Service of the IMPULSE solution is the key component to which they must be integrated in order to use IMPULSE as an eID system. The next figure represents most of the configurations that can be done over the Enterprise Service.

The following is a description of all the integrations required to establish a connection between a Public Administration Service and IMPULSE Enterprise Service. The IMPULSE Solution has two main flows: **onboarding** (user obtains an Identity Verifiable Credential), **authentication** (user authenticates using the Identity Verifiable Credential). All the integrations needed in both operations are explained below.

3.1 Onboarding

In the onboarding process it is not mandatory to do any integration, but a few can be addressed to give a better user experience to the citizens. When a user wants to make the onboarding, there are three possible options:

- The user selects their public administration from a list of trusted issuers within the IMPULSE application.
- The user uses the IMPULSE application to scan a QR Code in the public administration website.
- The user clicks on a Deep Link in the public administration website, using a web browser installed on their mobile device, which redirects them to the IMPULSE application.

Since the first option doesn't require any integration, it's always available, and the two others are completely optional. For the public administrations to address these optional ways of initiating the onboarding process, they have to make the Public Administration Service to connect to the Enterprise Service, requesting the content of the QR Code / Deep Link. The request is in fact an HTTP GET Operation to the endpoint **/users-onboarding/v2/initiate**. The response of this endpoint is a string like this:

```
openid://initiate_issuance?issuer=did%3Aebsi%3AzkqSHiqQSH1jk6U6846ebmu&cr
edential_type=https%3A%2F%2Fapi-pilot.ebsi.eu%2Ftrusted-schemas-
registry%2Fv2%2Fschemas%2Fz8Y6JJnebU2UuQQNc2R8GYqkEiAMj3Hd861rQhsoNWxsM
```

This URI must be directly converted into a QR Code or a Deep Link.

3.1.1 Show a QR Code

To convert this **request URI** field into a QR Code, the Public Administration Service will have to make use of any library supported by their programming language that serves for the conversion of text strings into QR codes.

The following is a compilation of some of the libraries for this task from among the most popular web development programming languages:

- Javascript/Typescript: **node-qrcode** [2]
- Python: **python-qrcode** [3]
- Java: **zxing** [4]
- C#: **QRCoder** [5]

Once the conversion is complete, the QR entered on the web page would look like the figure 3:

Figure 3: QR shown in the PA Web Application

3.1.2 Show a Deep Link

To convert this **request URI** into a Deep Link, the Public Administration Service won't need to do any conversion, just introduce the URI in the web page as a link.

3.2 Authentication

In the authentication process it is mandatory to do some integrations. Since the Enterprise Service is in charge of the authentication mechanism, but the user accesses the service through a web browser, it's extremely necessary for the Public Administration to display a QR, a Deep Link or both to the user. This also implies that the Public Administration Service will have to open (and protect) an endpoint to receive notifications of which user has successfully logged in.

For the public administrations to address one (or both) of these ways of initiating the Authentication process, they have to make the Public Administration Service to connect to the Enterprise Service, requesting the content of the QR Code / Deep Link. The request is in fact an HTTP GET Operation to the endpoint **/verification/v2/authentication-requests**. The response of these endpoint is a string like this:

```
openid://?response_mode=post&scope=openid&response_type=id_token&redirect
_uri=http%3A%2F%2F192.168.1.47%3A8080%2Fverification%2Fv2%2Fauthenticatio
n-
responses&client_id=http%3A%2F%2F192.168.1.47%3A8080%2Fverification%2Fv2%
2Fauthentication-
responses&claims=%7B%22id_token%22+%3A+%7B%7D%2C+%22vp_token%22+%3A+%7B%2
2presentation_definition%22+%3A+%7B%22format%22+%3A+null%2C+%22id%22+%3A+
%22impulse_vp_token%22%2C+%22input_descriptors%22+%3A+%5B%7B%22constraint
s%22+%3A+%7B%22fields%22+%3A+%5B%7B%22filter%22+%3A+%7B%22allOf%22%3A+%7B
%22type%22%3A+%22array%22%2C+%22contains%22%3A+%7B%22type%22%3A+%22object
%22%2C+%22properties%22%3A+%7B%22id%22%3A+%7B%22type%22%3A+%22string%22%2
C+%22pattern%22%3A+%22openid%22%7D%7D%2C+%22required%22%3A+%5B%22id%22%5D
%7D%7D%7D%2C+%22id%22+%3A+null%2C+%22path%22+%3A+%5B%22%24.vc.credentialS
```

```
chema%22%5D%2C+%22purpose%22+%3A+null%7D%5D%7D%2C+%22format%22+%3A+%7B%22
jwt%22+%3A+null%2C+%22jwt_vc%22+%3A+null%2C+%22jwt_vp%22+%3A+%7B%22alg%22
+%3A+%5B%22ES256K%22%5D%2C+%22proof_type%22+%3A+null%7D%2C+%22ldp%22+%3A+
null%2C+%22ldp_vc%22+%3A+null%2C+%22ldp_vp%22+%3A+null%7D%2C+%22group%22+
%3A+null%2C+%22id%22+%3A+%22id+document+credential%22%2C+%22name%22+%3A+n
ull%2C+%22purpose%22+%3A+null%2C+%22schema%22+%3A+null%7D%5D%2C+%22name%2
2+%3A+null%2C+%22purpose%22+%3A+null%2C+%22submission_requirements%22+%3A
+null%7D%7D%7D&nonce=21t4SG5h3n7g7sxu4navq6nS5nmh3g58Ldhs1AAyq3rv
```

This **request URI** is the text string that has to be converted into a QR Code or a Deep Link.

3.2.1 Show a QR Code

It is exact same procedure as in section 3.1.1.

3.2.2 Show a Deep Link

It is exact same procedure as in section 3.1.2.

3.2.3 Handle the user session

Since the Authentication process occurs between the IMPULSE Application and the Enterprise Service, but the service is provided through other means (web service, locker...), it's mandatory to handle the user session using a **nonce** field received in the endpoint **/verification/v2/authentication-requests**. And this **nonce** has to be linked in some way with user session inside the device where the service is being provided. As a result, the Public Administration Service will be able to identify which device needs to receive the provided service.

The common scenario for the public administrations is that their web application handles the user sessions using some of the next mechanisms: cookies, JWTs, HTTP sessions, etc. The only required integration here is to associate this **nonce** field to the user session, using the mechanism already in place. It is up to the public administration to decide how long this **nonce** field is valid and how often it should be refreshed, but these three factors should be considered:

- If the PA web page can be automatically refreshed with a new **nonce** without user interaction, there is no need for **nonces** to have a long duration. If it is not the case, constant manual refreshing might impact the user experience.
- The **nonce** must be valid from the start to the end of the authentication process. This means that, while the nonce is valid, the user will scan the QR Code (or click a deep link), select a credential from a list, and take a selfie to unlock it. This is something that can be done very quick if the person has done it before (15~30 seconds), but to ensure that everyone can use it properly, a common safe value of nonce duration is 15 minutes.
- Long duration **nonces** are less secure if replay attacks are not handled properly.

3.2.4 Exposing the endpoint to receive the notification

After a citizen has successfully authenticated to the Enterprise Service, this component has to communicate to the Public Administration Service that a user has logged in. This communication will be in fact an HTTP POST Operation to one endpoint exposed from the Public Administration Service for this purpose. This endpoint must receive a JSON Object like this:

```
{
      "id_token":"eyJraWQiOiJkaWQ6ZWJzaTp6a3FTSGlxUVNIMWprNlU2ODQ2ZWJtdSN
      lNWQ4N2MzN2UzMTc0ZjNhOWIzMGE0N2MwMjljZDBjYSIsInR5cCI6IkpXVCIsImFsZy
      I6IkVTMjU2SyJ9.eyJhdWQiOiJodHRwOlwvXC8xOTIuMTY4LjEuNDY6ODA4MFwvaW50
      ZXJuYWxcL3YyXC9pZC10b2tlblwvdmVyaWZ5IiwiY3JlZGVudGlhbFN1YmplY3QiOns
      iaWQiOiJkaWQ6ZWJzaTp0ZXN0IiwiZmlyc3ROYW1lIjoiVGVzdGVyIiwiZmFtaWx5Tm
```

```
        FtZSI6IlRlc3RpbmciLCJkYXRlT2ZCaXJ0aCI6IjAxXC8wMVwvMTkwMCIsInBlcnNvb
        mFsSWRlbnRpZmllciI6IjEyMzQ1Njc4QSJ9LCJpc3MiOiJkaWQ6ZWJzaTp6a3FTSGlx
        UVNIMWprNlU2ODQ2ZWJtdSIsImV4cCI6MTY4NTM4MTQxMiwiaWF0IjoxNjg1MzgxMTE
        yLCJub25jZSI6Im5vbmNlIn0.CWio3X9Elu9Ogic1SMKmKRsz1n3AwWhtIq36gFAa_M
        F56WntVUzUa9yhhcjhLEn1MP83tuVfqLY0kDmY6vl6VA"
}
```

The **id_token** field is a JWT that contains user information related to its authentication. This JWT is signed by the digital wallet contained in the Enterprise Service (enterprise wallet) and provides the **credentialSubject** of the verifiable credential presented by the user during authentication, and the **nonce** that identifies the user session.

```
{
    "aud": "http://192.168.1.46:8080/internal/v2/id-token/verify",
    "credentialSubject": {
        "id": "did:ebsi:test",
        "firstName": "Tester",
        "familyName": "Testing",
        "dateOfBirth": "01/01/1900",
        "personalIdentifier": "12345678A"
    },
    "iss": "did:ebsi:zkqSHiqQSH1jk6U6846ebmu",
    "exp": 1685381412,
    "iat": 1685381112,
    "nonce": "nonce"
}
```

If the Public Administration Service has the capability of resolving DIDs in the EBSI ledger to get a public key and parsing and verifying JWTs, the PAS can do it itself. However, in the very plausible case that this was not possible, the Enterprise Service has an internal endpoint exposed (**/internal/v2/id-token/verify**) that performs all the necessary operations to parse and verify the **id_token**, that can be invoked via **HTTP POST** operation introducing the **id_token** in the JSON body of the request in the same format that was received. The response of this endpoint will be a JSON Object like this:

```
{
    "verified": true,
    "nonce": "nonce"
    "credentialSubject": {
        "id": "did:ebsi:test",
        "firstName": "Tester",
        "familyName": "Testing",
        "dateOfBirth": "01/01/1900",
        "personalIdentifier": "12345678A"
    }
}
```

It indicates if the **id_token** has been verified, and it provides the extracted **nonce** and **credentialSubject**.

4 Instantiation of the Enterprise Service

The instantiation process is thought to be very straightforward for any public administration that wants to deploy IMPULSE and connect to it. The Enterprise Service, which is the core element that will need to be instantiated by the public administrations, has been dockerized to facilitate the sharing and deployment of the component. To ease the configuration of this docker image prior to its deployment, a Configuration Tool has been created along with a manual that explains every step needed to perform an integration and instantiation of the IMPULSE solution.

4.1 Resources

1. Docker Image of the Enterprise Service.
   - It is shared through a robot account (only reading permissions) of a private docker repository (Harbor) owned by GRAD.
2. The configuration file (**enterprise-service.env**) that contains all the configurations.
   - To facilitate the instantiation, most of the configurations have been pre-emptively done by GRAD using the older configuration files, meaning that the PA IT team will need to update very few fields.
3. Data folder.
   - To facilitate the instantiation for the first pilot we have already onboarded one different DID for every public administration, and contacted EBSI's Service Desk to register these DIDs as Trusted Issuers. This means that each public administration receives a pre-configured **data/** folder with everything they need to setup IMPULSE.
4. Manual "How to IMPULSE: Integration, Instantiation and Interaction".

4.2 Requirements

- Docker 20+
- Virtual CPUs: 2+
- RAM: 8GB+

4.3 Configurations

The next figure represents, at a high level, the configurations that can be done with Enterprise Service configuration tool, reflecting their role in the system. Red squares are the configurable attributes that can be adjusted to deploy the ES in different PA infrastructures. It is important to note that not all the configurations are mandatory (e.g., a specific PA might not have an outbound proxy)

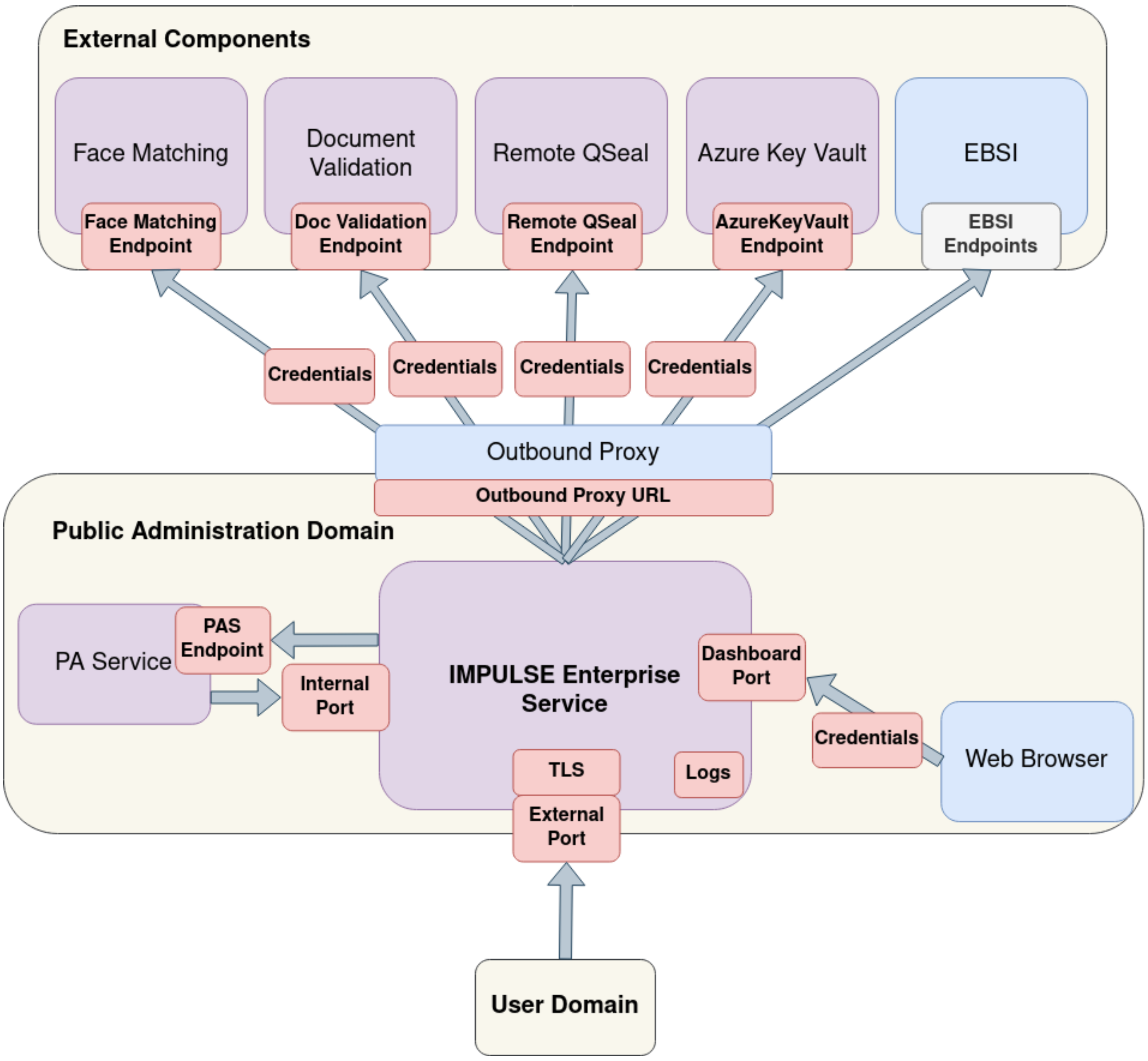


Figure 4: Overview of the role of the Enterprise Service configurable attributes.

In the following sections, all of the possible configurations of the Enterprise Service will be addressed, regardless of whether they are mandatory or optional.

4.3.1 Enterprise Service General Configuration

The next attributes are all the possible configurations related to the general information of the Enterprise Service.

- **Public URL**: The complete public URL where the Enterprise Service is exposed.
  - Example: https://www.gradiant.org/enterprise-service
  - Example: https://www.gradiant.org:8443/enterprise-service
  - Example: http://1.1.1.1
  - Example: http://1.1.1.1:8080

** This attribute also serves to configure the context path of the Enterprise Service. For instance, if you configure an URL like the one in the first example, the context path will be “/enterprise-service”.

- **External Port**: The external port that exposes the endpoints that the IMPULSE user application has to invoke.
    - Example: 80
    - Example: 443
    - Example: 8080
    - Example: 8443
- **Internal Port**: The internal port that exposes an endpoint that the PAS has to invoke.
    - Example: 80
    - Example: 443
    - Example: 8080
    - Example: 8443
- **Internal API Key**: The API Key that the PAS has to provide when accessing the ES internal endpoint as an authentication mechanism.
    - Example: jsdkfb92hfi2dswi0kfbf5gnyh98u234r
    - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u
- **Public Administration Service Notification Endpoint**: The complete public/private URL where the Public Administration Service has the notification endpoint exposed.
    - Example: https://www.gradiant.org/callback
    - Example: http://1.1.1.1:8080/notification
- **Public Administration Service Notification API Key**: The API Key that the ES has to provide when accessing the notification endpoint as an authentication mechanism.
    - Example: jsdkfb92hfi2dswi0kfbf5gnyh98u234r
    - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u
- **Manual Check Enabled**: Whether the manual check of the onboarding requests is enabled or not. Note that this option is only intended for testing purposes.
    - Example: yes
    - Example: NO

4.3.2 Face Matching Configuration

The next attributes are all the possible configurations related to the Face Matching Service information.

- **Face Matching Enabled**: Whether the face matching service is enabled during the onboarding. Note that the disabling option is only intended for testing purposes.
    - Example: yes
    - Example: NO
- **URL**: The complete URL where the Face Matching Service API is available.
    - Example: https://apis.alicebiometrics.com/face
- **Client Id**: The client id that identifies the entity that is accessing the Face Matching Service API.
    - Example: impulse-gradiant-testing
- **API Key**: The API Key that the ES has to provide when accessing the Face Matching Service API as an authentication mechanism.
    - Example: jsdkfb92hfi2dswi0kfbf5gnyh98u234r
    - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u

4.3.3 Document Validation Configuration

The next attributes are all the possible configurations related to the Document Validation Service information.

- **Document Validation Enabled**: Whether the document validation service is enabled during the onboarding. Note that the disabling option is only intended for testing purposes.
- **URL**: The complete URL where the Document Validation Service API is available.
  - Example: https://impulse.tree-api.com
- **Username**: The username that identifies the entity that is accessing the Document Validation Service API.
  - Example: impulse-gradiant
- **Password**: The password that the ES has to provide when accessing the Document Validation Service API as an authentication mechanism.
  - Example: jsdkfb92hfi2dswi0kfbf5gnyh98u234r
  - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u

4.3.4 MRZ Reader Configuration

The next attributes are all the possible configurations related to the MRZ Reader Service information.

- **MRZ Reader Enabled**: Whether the MRZ Reader is enabled during the onboarding. Note that the disabling option is only intended for testing purposes.
- **URL**: The complete URL where the MRZ Reader Service API is available.
  - Example: https://impulse.tree-api.com
- **Username**: The username that identifies the entity that is accessing the MRZ Reader Service API.
  - Example: impulse-gradiant
- **Password**: The password that the ES has to provide when accessing the MRZ Reader Service API as an authentication mechanism.
  - Example: jsdkfb92hfi2dswi0kfbf5gnyh98u234r
  - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u

4.3.5 Trusted Issuer Configuration

The next attributes are all the possible configurations related to the Trusted Issuer information.

- **DID**: The decentralized identifier (DID) of the trusted issuer.
  - Example: did:ebsi:zkqSHiqQSH1jk6U6846ebmu
- **Tokens Key Id**: The key id used by the enterprise wallet to sign JWTs used during the onboarding. Note that the key referenced by this key id is not stored locally, but in a secure HSM Cloud in Azure Key Vault.
  - Example: e5d87c37e3174f3a9b30a47c029cd0ca
- **Tokens Key JWS Algorithm**: The JWS Algorithm of the tokens key id.
  - Example: ES256K
- **Credentials Key Id**: The key id used by the enterprise wallet to sign verifiable credentials issued to the user. Note that the key referenced by this key id is not stored locally, but in a secure HSM Cloud in Azure Key Vault, and note as well that this key is only used in the case that the Remote QSeal is disabled (only for testing purposes).
  - Example: e5d87c37e3174f3a9b30a47c029cd0ca

- **Credentials Key JWS Algorithm**: The JWS Algorithm of the credentials key id.
    - Example: ES256K
- **EBSI Key Id**: The key id used by the enterprise wallet as an authentication mechanism in EBSI APIs.
    - Example: e5d87c37e3174f3a9b30a47c029cd0ca
- **EBSI Key JWS Algorithm**: The JWS Algorithm of the EBSI key id.
    - Example: ES256K

4.3.6 Dashboard Configuration

The next attributes are all the possible configurations related to the Public Servant Dashboard information.

- **Port**: The port that will be expose the Public Servant Dashboard endpoints.
    - Example: 80
    - Example: 8443

Note that the next options refer to one user of the dashboard, but multiple can be configured, each with their respective credentials.

- **Username**: The username that the Public Servant will use to login in the dashboard.
    - Example: admin
    - Example: john_doe
- **Password**: The password that the Public Servant will use to login in the dashboard.
    - Example: changeit
- **Two Factor Authentication enabled**: Whether the two-factor authentication to access the dashboard is enabled.
    - Example: yes
    - Example: NO
- **Email**: The email account that will receive a TOTP (Time-based One-Time Password) to perform a two-factor authentication by the Public Servant.
    - Example: johndoe@impulse.com
- **Authentications Dashboard enabled**: Whether the authentications dashboard is enabled. This dashboard is specifically created for testing purposes and for the ARH use case, where you can see a list of the authenticated users in the system.
    - Example: yes
    - Example: NO

4.3.7 Remote QSeal Configuration

The next attributes are all the possible configurations related to the Remote QSeal Service information.

- **Remote QSeal Enabled**: Whether the remote QSeal service is enabled to issue verifiable credentials. Note that the disabling option is only intended for testing purposes.
    - Example: true
    - Example: false
- **Authentication URL**: The complete URL where the Remote QSeal Authentication API is available.
    - Example: https://impulse-keycloack.infocert-labs.eu
- **Username**: The username used to authenticate to the Remote QSeal Authentication API.
    - Example: GRADIANT

- **Password**: The password used to authenticate to the Remote QSeal Authentication API.
    - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u
- **Client Id**: The client id used to authenticate to the Remote QSeal Authentication API.
    - Example: impulse-login
- **Client Secret**: The client secret used to authenticate to the Remote QSeal Authentication API.
    - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u
- **Service URL**: The complete URL where the Remote QSeal Service API is available.
    - Example: https://impulse-credential-services.infocert-labs.eu
- **Packaging**: The format of the signature that is going to be issued by the Remote QSeal Service.
    - Example: JWS
- **Pin**: The pin to access the HSM that will use the Remote QSeal Service to sign the verifiable credential.
    - Example: 12345678
- **SAT**: Long duration JWT that the Enteprise Service has to be provide when accessing the Remote QSeal Service API as an authentication mechanism.
    - Example: eyJ4NXUiOiJodHRwczovL3RyaWFsLmNsaW50Ym...

4.3.8 TLS Configuration

The next attributes are all the possible configurations related to enabling TLS in the Enterprise Service.

- **TLS Enabled in External Port**: Whether TLS is enabled for the external port of the Enterprise Service.
    - Example: yes
    - Example: NO
- **TLS Enabled in Internal Port**: Whether TLS is enabled for the internal port of the Enterprise Service.
    - Example: yes
    - Example: NO
- **TLS Enabled in Dashboard Port**: Whether TLS is enabled for the dashboard port of the Enterprise Service.
    - Example: yes
    - Example: NO
- **Path to JKS PKCS12**: The relative or absolute path to the Java Keystore PKCS12 that contains the certificate and key to be used as part of the TLS configuration.
    - Example: ./impulse.jks
    - Example: /home/impulse/keystores/impulse.jks
- **JKS Password**: Password that unlocks the Java Keystore.
    - Example: changeit
- **Key Alias**: The alias of the key corresponding to the certificate inside the JKS.
    - Example: impulse
- **Key Password**: Password that unlocks the Key inside the Java Keystore, identified by the key alias.
    - Example: changeit

4.3.9 Azure Key Vault Configuration

The next attributes are all the possible configurations related to the Azure Key Vault information.

- **Base URL**: The base URL (domain) where the Azure Key Vault of the PA is located.
  - Example: impulse-grad-vault.vault.azure.net
- **Client Id**: The client id used to authenticate to the Azure Key Vault of the PA.
  - Example: 1457a01e-1018-4731-be6d-a81371f27ddb
- **Client Secret**: The client secret used to authenticate to the Azure Key Vault of the PA.
  - Example: fnhjur23489b8rhu24b34rf5n7yh9m8u

4.3.10 Logging configuration

The next attributes are all the possible configurations related to the logging of the Enterprise Service.

- **Level**: DEBUG, INFO levels available. The DEBUG level is meant to be used for finding problems in the integration and instantiation, while the INFO level is intended to be used in the piloting of the project (since it gives a high-level overview of user interactions with the Enterprise Service).
  - Example: DEBUG
  - Example: INFO

4.3.11 Proxy configuration

The next attributes are all the possible configurations related to the logging of the Enterprise Service.

- **Proxy Enabled**: Whether outbound traffic of the Enterprise Service has to go through a proxy located in the PA infrastructure.
  - Example: true
  - Example: false
- **URL**: The complete URL where the outbound proxy is located.
  - Example: http://localhost:3128
- **Non-Proxy Hosts**: The specific hosts that if the outbound traffic is directed to them, the proxy must not be used.
  - Example: localhost|*.gradiant.org

4.4 Deployment

Before deploying the Enterprise Service, it is necessary to obtain its docker image to run the container. For doing this, we first need to login into the GRAD private repository where the image is stored with a robot account provided to the public administrations, and then get the Enterprise Service image with a “docker pull” command. After acquiring the image, the deployment process is simple as the configuration tool can prepare a docker run command with every file needed for the instantiation.

Example of prepared run command:

```
docker run -d -p 8080:8080 -p 443:443 --name enterprise-service -v
$PWD/data:/data --env-file $PWD/enterprise-service.env
harbor.gradiant.org/syp-impulse-pr-01349/enterprise-service:2.0.1
```

5 Interaction with the Public Servant Dashboard

The Public Servant Dashboard is a component of the Enterprise Service that allows an operator to manually accept onboarding requests from the citizens. Every request is sent to this dashboard after it has been handled by the face matching, document validation, and MRZ reader services, including information on it about the results obtained.

5.1 Access

This dashboard will be available as a web application on route **/tasks**. After the first access in a web browser, basic user/password authentication will be requested. These values must have been configured according to section 3.3.3.

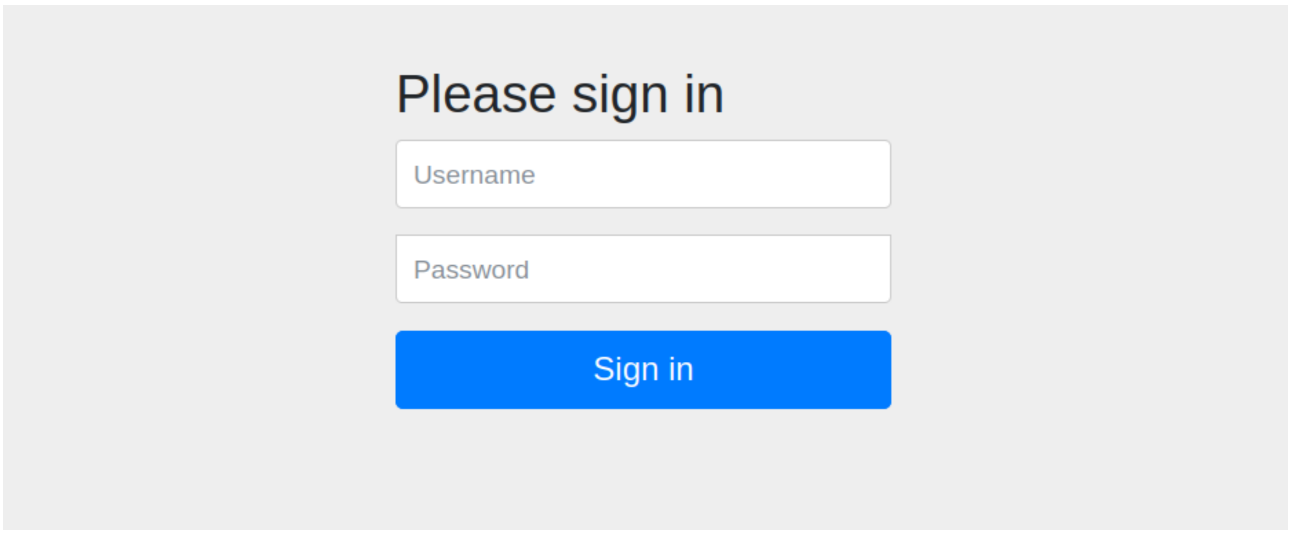


Figure 5: Basic Login in the Public Servant Dashboard

If the dashboard has been configured to have a two-factor authentication, the next screen will appear.

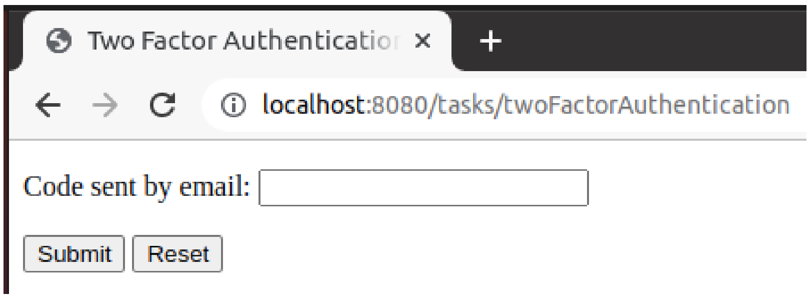
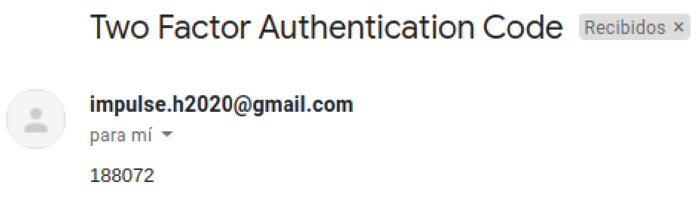


Figure 6: Two-factor authentication method in the Public Servant Dashboard

One valid code that lasts 15 minutes is sent to the previously configured email (section 3.3.3). You can resend a valid code refreshing the page.



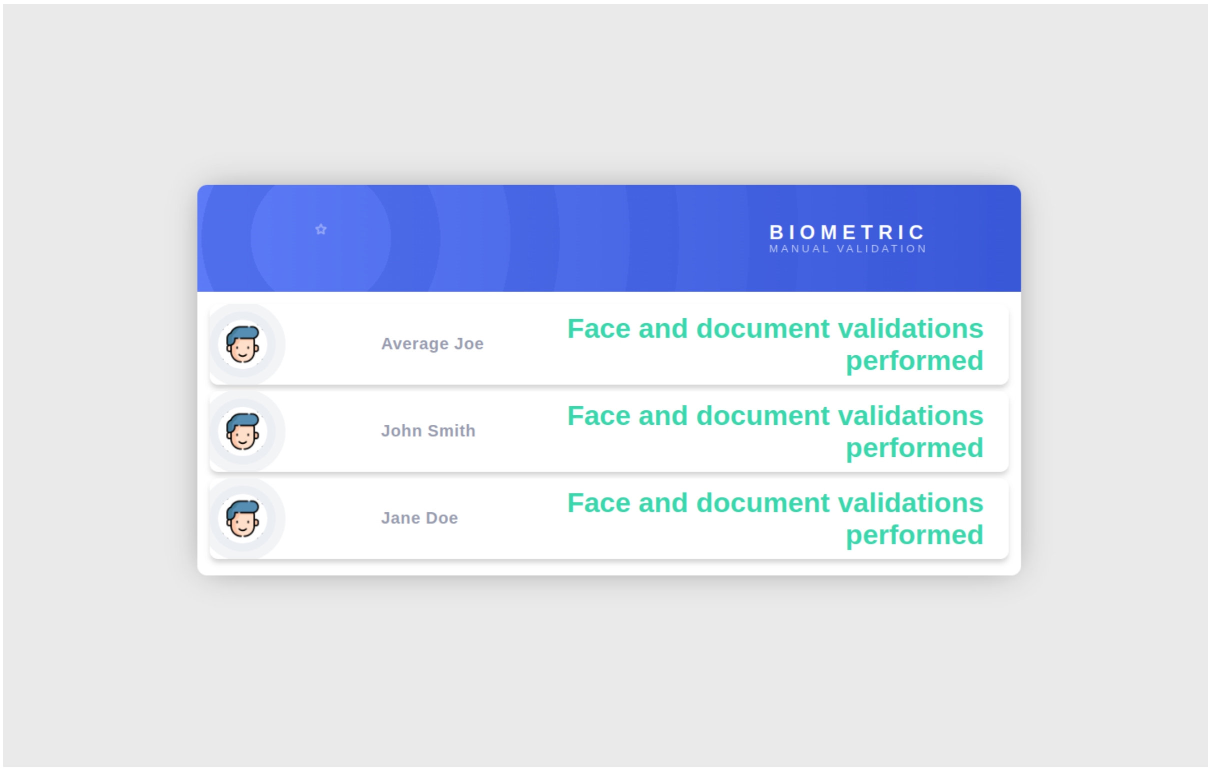
Two Factor Authentication Code Recibidos ×

impulse.h2020@gmail.com
para mí
188072

Figure 7: Two-factor authentication code in the public servant's email.

5.2 Usage

After login completely, the dashboard will appear, and it consists of a list of requests with the names of the citizens and the validations performed.



Average Joe	Face and document validations performed
John Smith	Face and document validations performed
Jane Doe	Face and document validations performed

Figure 8: Batched onboarding requests in the Public Servant Dashboard

After clicking on any request, the specific page with all the information it contains will be shown to the Public Servant. First, we will see the identifier of the request, the user information, the description of the face matching validation service result, and the description of the document validation service result. Then, the images sent by the user will appear, being the selfie [6] on the left, the actual images of the ID Document in the middle, and the proof images created by the document validation service on the right. Below the images, there are three buttons that correspond to the actions that the dashboard operator can perform. The request can be accepted or rejected, being possible to go back to the onboarding requests list and leave it for later.

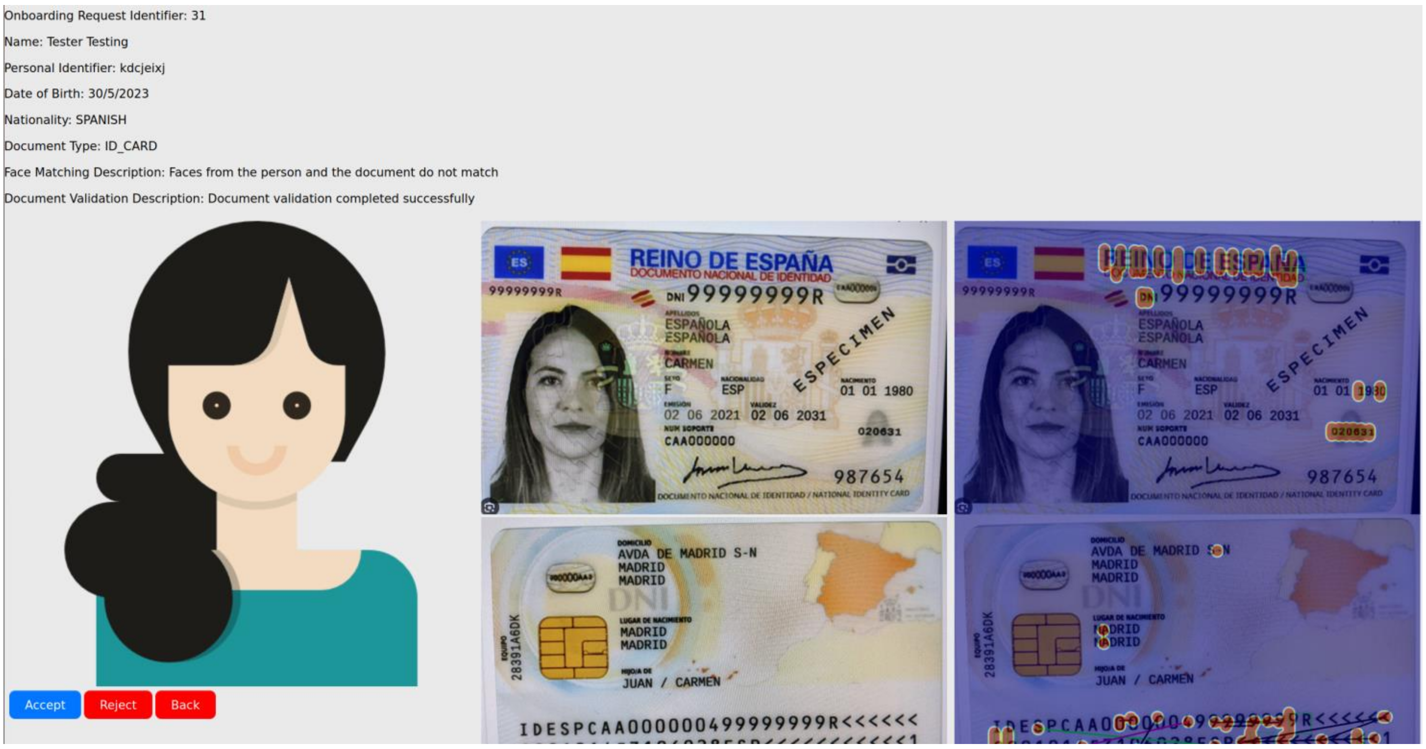
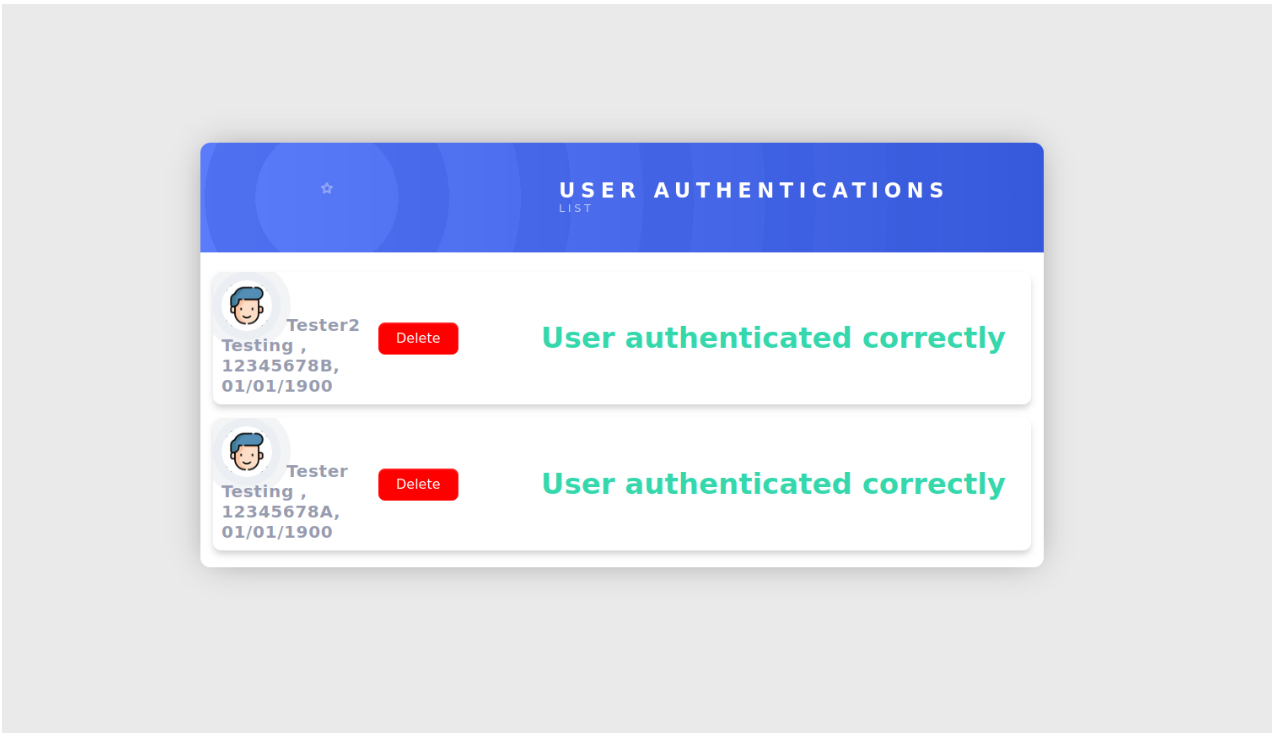

Onboarding Request Identifier: 31
Name: Tester Testing
Personal Identifier: kdcjeixj
Date of Birth: 30/5/2023
Nationality: SPANISH
Document Type: ID_CARD
Face Matching Description: Faces from the person and the document do not match
Document Validation Description: Document validation completed successfully

Figure 9: Detailed onboarding request in the Public Servant Dashboard

In the case that the authentications dashboard is enabled, a list of correctly authenticated users will be available for the Public Servant. Each element of the list contains the personal information of the citizen, and a delete button to be removed from the list after the corresponding checks. This dashboard was explicitly made for testing purposes and for the ARH use case.


USER AUTHENTICATIONS
LIST

Tester2 Testing , 12345678B, 01/01/1900 — Delete — User authenticated correctly

Tester Testing , 12345678A, 01/01/1900 — Delete — User authenticated correctly

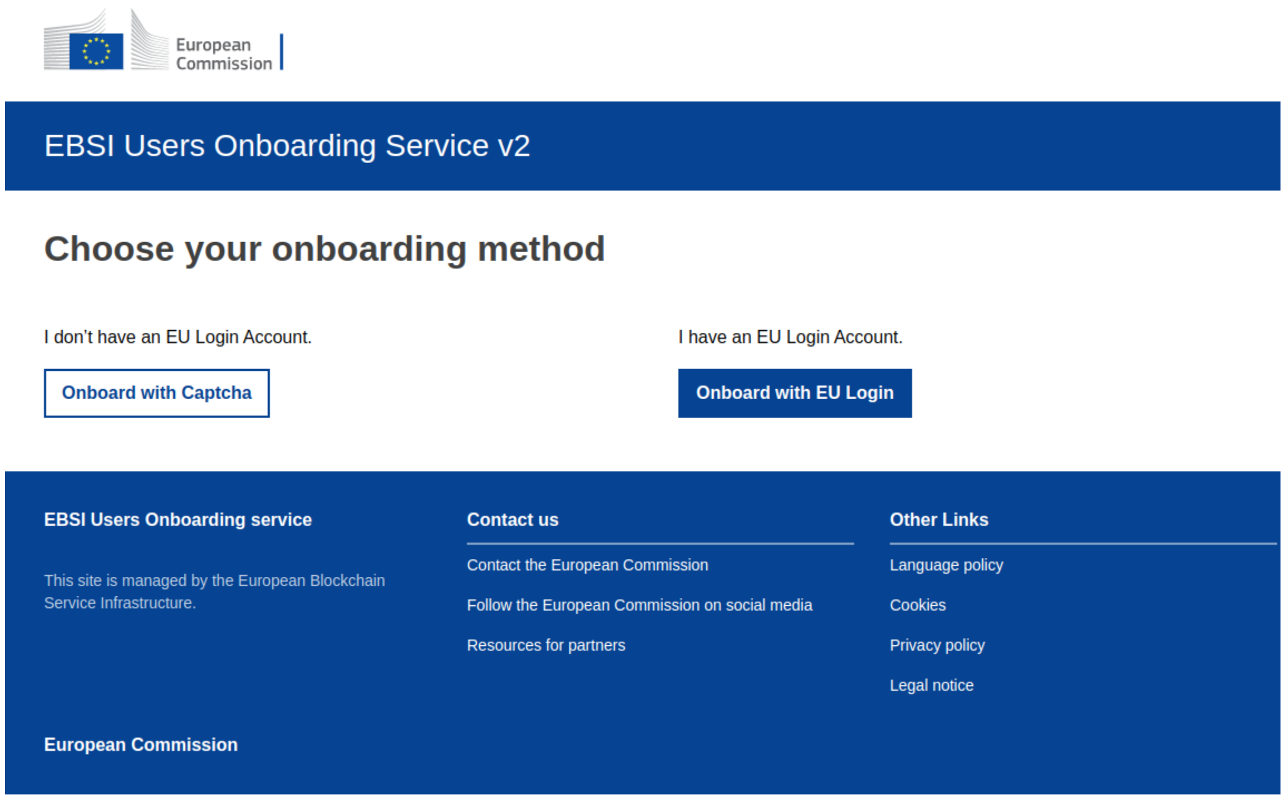
Figure 10: Batched authentications in the Public Servant Dashboard

6 Refreshing the Trusted Issuer

The verifiable authorizations, issued to each public administration Trusted Issuer, have a validity of 6 months. This means that at least every 6 months, one IT department member of the public administration should stop the deployed docker container, update this verifiable authorization and redeploy it again. If this verifiable authorization expires, the public administration won't be able to change the URL of the Enterprise Service, so this is something that is recommended to be done once every 3 months.

The method of updating this EBSI Verifiable Authorization is pretty simple, as the configuration tool has the functionality to handle this process. The only requirement is to get a session token from the EBSI Users Onboarding Page.

To get this, the operator that is refreshing the Trusted Issuer has to enter the page: https://app-pilot.ebsi.eu/users-onboarding/v2.


European Commission

EBSI Users Onboarding Service v2

Choose your onboarding method

I don't have an EU Login Account.

Onboard with Captcha

I have an EU Login Account.

Onboard with EU Login

EBSI Users Onboarding service

This site is managed by the European Blockchain Service Infrastructure.

European Commission

Contact us

Contact the European Commission

Follow the European Commission on social media

Resources for partners

Other Links

Language policy

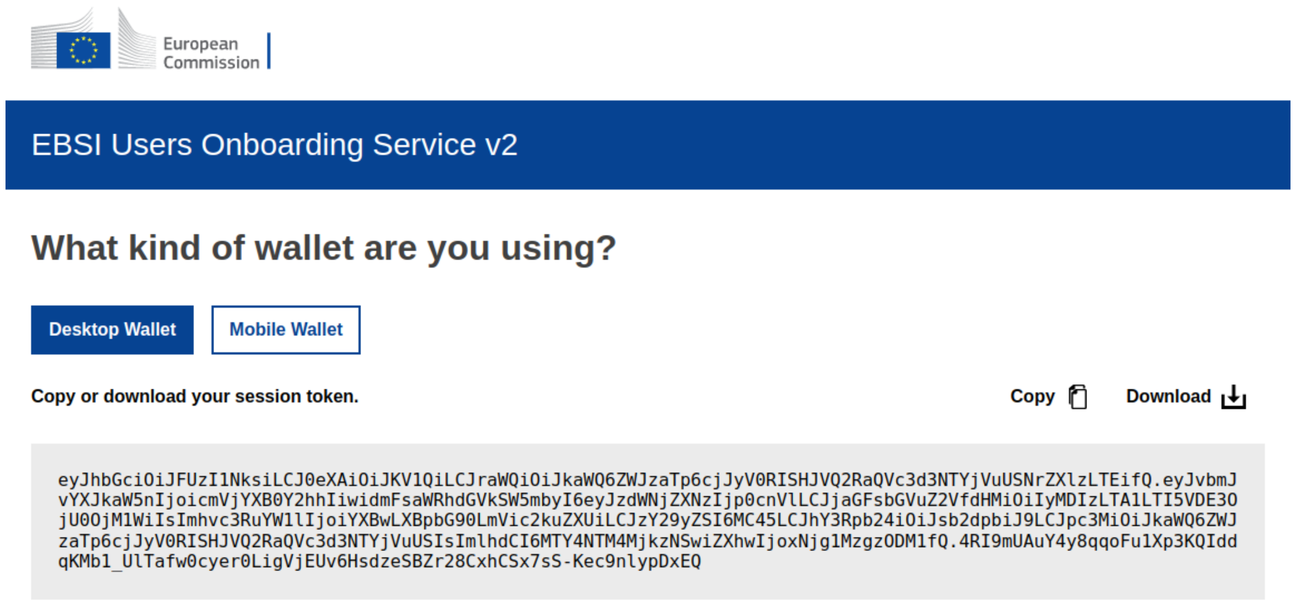
Cookies

Privacy policy

Legal notice

Figure 11: EBSI Onboarding Service v2 page for the pilot environment

Once inside, they will have to click on "Onboard with Captcha", and then select "Desktop Wallet", which will make the session token appear.



What kind of wallet are you using?

Copy or download your session token.

```
eyJhbGciOiJFUzI1NksiLCJ0eXAiOiJKV1QiLCJraWQiOiJkaWQ6ZWJzaTp6cjJyV0RISHJVQ2RaQVc3d3NTYjVuUSNrZXlzLTEifQ.eyJvbmJ
vYXJkaW5nIjoicmVjYXB0Y2hhIiwidmFsaWRhdGVkSW5mbyI6eyJzdWNjZXNzIjp0cnVlLCJjaGFsbGVuZ2VfdHMiOiIyMDIzLTA1LTI5VDE3O
jU0OjM1WiIsImhvc3RuYW1lIjoiYXBwLXBpbG90LmVic2kuZXUiLCJzY29yZSI6MC45LCJhY3Rpb24iOiJsb2dpbiJ9LCJpc3MiOiJkaWQ6ZWJ
zaTp6cjJyV0RISHJVQ2RaQVc3d3NTYjVuUSIsImlhdCI6MTY4NTM4MjkzNSwiZXhwIjoxNjg1MzgzODM1fQ.4RI9mUAuY4y8qqoFu1Xp3KQIdd
qKMb1_UlTafw0cyer0LigVjEUv6HsdzeSBZr28CxhCSx7sS-Kec9nlypDxEQ
```

Figure 12: Onboarding session token obtained from the EBSI Onboarding Service v2

After copying it to the clipboard, it can be introduced in the Configuration Tool to update the EBSI Verifiable Authorization.

7 Instantiation/Integration Activities

7.1 First Prototype Session

In February 2022, GRAD organized an online meeting with the public administrations of the Consortium to show the first prototype of the IMPULSE solution. It included a presentation of the message flow of the solution, an explanation of the points of the flow in which the public administrations were involved, a compilation of the requirements needed for the complete integration with the Enterprise Service, and a live demo of the solution. In this demonstration, the entire onboarding and authentication processes were shown, and then, GRAD answered all the questions that arose from the participants.

From this session, GRAD received some feedback which led to a series of changes in the solution itself, and its configuration/deployment:

- Dockerization of the Enterprise Service was addressed after receiving some concerns about the deployment process.
- The notification system via email was changed to a PUSH Notification system due to some concerns about the need to collect the emails of the users.
- An internal evaluation on the use of Legal Entity Verifiable Credentials for the Verifiers was addressed to avoid the need of the Verifiers to be Trusted Issuers.
- An internal evaluation on the possibility of cross-border scenarios among the public administrations that use IMPULSE as an eID system.
- The first version of the Configuration Tool was made to facilitate the onboarding of the public administrations' DIDs as Trusted Issuers.

7.2 F2F IMPULSE Instantiation

In March 2022, a face-to-face meeting was held in Vigo gathering Consortium's partners. Among the different presentations, activities, or workshops carried out, GRAD made a live instantiation process of the Enterprise Service with a Dummy Public Administration Service that was developed and integrated by ICERT. Prior to the demo, the integrations that are necessary to connect a PAS with IMPULSE were explained. In this live demonstration of a real instantiation, an Azure Machine with Ubuntu 18.04 was created, an SSH connection was established with the machine, the configuration tool was used to onboard a DID in EBSI, the DID was requested to be registered as Trusted Issuer through the EBSI Service Desk, and both Dummy PAS Service and Enterprise Service were built as Docker images and deployed. Then, a demo of the authentication process was made to show that the instantiation and integration was successful. Finally, GRAD answered all the questions that came up among the participants.

From this session, GRAD received some feedback which led to a series of changes in the solution itself, and its configuration/deployment:

- The Deep Link method to initiate an onboarding or authentication was developed due to some concerns about the possibility of performing the two IMPULSE main processes only with the mobile device. This was made for the specific citizens that want to receive the PAS service in a web browser on the same device where the IMPULSE application is installed, without the need of using a second one (PC, laptop, tablet, etc.).
- The Docker images build step was removed from the instantiation process to simplify it. Instead, a pre-built image is stored in a private Docker repository owned by GRAD, and shared through a robot account with only reading permissions.
- The DID Onboarding and Trusted Issuers registration steps were removed, as we could just simply make all the registrations ourselves (GRAD), and then share the pre-configured wallet to each public administration for this first round of IMPULSE.
- The functionalities of the Configuration Tool were expanded to cover every possible configurable area (organization name, URLs, TLS, logging, dashboard, face matching service, document verification service).

- The manual "How to IMPULSE: Integration, Instantiation and Interaction" was made to cover everything that the public administrations needed to know to correctly perform the Instantiation phase.

7.3 Meeting instantiations for the first piloting

From May 2022 to September 2022, several meetings were held with the IT teams of the PAs to help them with the integration and instantiation of the IMPULSE solution.

From these sessions, GRAD received some feedback which led to a series of changes in the solution itself, and its configuration/deployment for the second version:

- The configuration tool is provided within the same docker image as the Enterprise Service. In this way, the PAs could avoid installing a Java JDK and other issues that could appear executing the configuration tool.
- The Enterprise Service can expose different functionalities in different ports: external endpoints, internal endpoint, and dashboard endpoints. This facilitates the PA team the work to control the access to these endpoints, as they can limit it by port, and not having to create specific path rules for each group of endpoints.
- The Enterprise Service can redirect its outbound traffic through an HTTP proxy. In one use case, all of the outgoing traffic had to go through an outbound proxy that checked that all of the requests were allowed.

7.4 Meeting instantiations for the second piloting

During May 2023, several meetings were held with the IT teams of the PAs to help them with the integration and instantiation of the IMPULSE solution. As it was the second time the solution was instantiated, most of these meetings were very straight forward and successful, and all of the issues that appeared were rapidly tackled.

8 Conclusions

With the creation of the Docker image of the Enterprise Service, the implementation of the configuration tool, and the preparation of the manual "How to IMPULSE: Integration, Instantiation and Interaction", we have successfully fulfilled the requirements of the public administrations to deploy the IMPULSE solution.

GRAD has conducted several meetings with the PAs to effectively address the specific issues faced by each entity. The feedback obtained by the PAs in several meetings allowed to improve the instantiation automation mechanisms as well the documentation to facilitate the deployment of the solution. As a result, GRAD has built up a robust process for the IMPULSE system instantiation to be used by the PAs for the second piloting activities.

References

[1] **[AJ Alt, 2022]** https://ajalt.github.io/clikt/
[2] **[Ryan Day, 2022]** https://github.com/soldair/node-qrcode
[3] **[Lincoln Loop, 2021]** https://github.com/lincolnloop/python-qrcode
[4] **[Zxing, 2022]** https://github.com/zxing/zxing
[5] **[Rafael Herrmann, 2021]** https://github.com/codebude/QRCoder
[6] **[Brian J Brennan, 2020] [dDara, 2022]** https://www.flaticon.com/free-icon/woman_949666