



REFERENCE IMPLEMENTATION USER MANUAL

INDEX

1	INTRODUCCIÓN	4
1.1	Access to the administration interface	4
2	PLATFORM USER MANAGEMENT	5
2.1	Users	5
2.2	Groups	7
3	MANAGEMENT OF VERIFIABLE CREDENTIALS	9
3.1	OpenID	9
3.1.1	Defining an Issuance Flow	9
3.1.2	Defining the Issuance Information	10
3.1.3	Defining a Verify Flow	11
3.1.4	Defining the Presentation Definition	12

List of Figures

Figure 1:Django Admin Login	4
Figure 2: Users list.....	5
Figure 3: Users information.....	5
Figure 4:User Add form	6
Figure 5: User Permissions	6
Figure 6: User Permissions - 2	7
Figure 7: Group list	8
Figure 8: Group Add Form	8
Figure 9: OpenID configurations	9
Figure 10: Issuance flow.....	10
Figure 11: Issuance information	10
Figure 12: Issuance Information Entry	11
Figure 13: Verify flow	12
Figure 14: Presentation Definition	13

1 INTRODUCTION

The purpose of this document is to describe the functionalities that a USER of the reference implementation of the Enterprise Wallet associated with deliverable 3.6 can perform from the administration interface.

This manual includes, on the one hand, a description of how to use the functionalities related to the management of verifiable credentials currently provided by the platform and the management of users.

1.1 Access to the administration interface

The backend of the reference implementation exposes an administration interface that is accessible from any browser. To access it, once an instance of the Wallet has been launched, simply go to the URL where it is hosted. If everything works correctly, a screen similar to the one below should appear, prompting you to enter an username and password. The first time you access the system, you can use the username and password: "Identfy". This will allow you to log in as a predefined user in the system with superuser permissions.

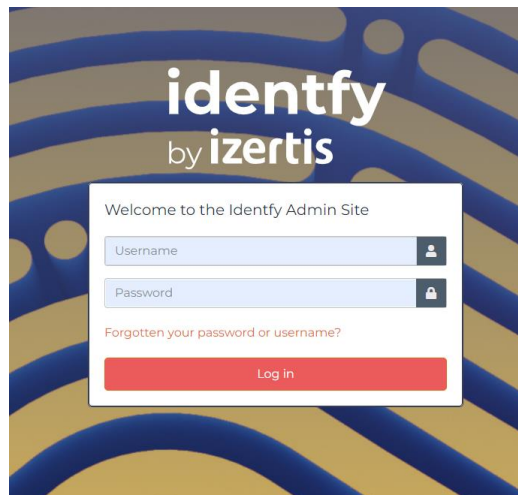


Figure 1:Django Admin Login

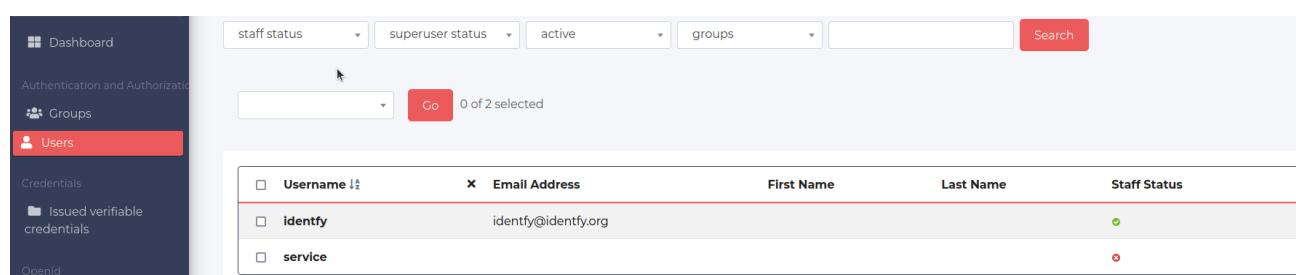
2 PLATFORM USER MANAGEMENT

The following sections explain how to view user information and manage the different models related to platform management.

2.1 Users

Users is a native Django model that provides the data for logging into the Administration Platform. In this model, user permissions are configured, as well as personal information such as name, surname, email, and user type.

To view user information, go to the "Users" section in the side menu.



<input type="checkbox"/>	Username	Email Address	First Name	Last Name	Staff Status
<input type="checkbox"/>	Identify	identify@identify.org			
<input type="checkbox"/>	service				

Figure 2: Users list

As you can see, in addition to the user used for logging in, there is also a "service" user. This user has a set of special permissions that only allow it to read certain specific data models related to credential issuance and verification. It is a user designed to be used by the "Entity Service" component. If desired, you can change both the name and password of the default users, or even delete them if you already have others.

If you interact with any of the user entries, you will see a menu like the one below with all the configurable categories.



Users

Home > Authentication and Authorization > Users > identify

General

Personal info

Permissions

Important dates

Save

Delete

History

Figure 3: Users information

If, on the other hand, you want to create a new user, simply click the "add" button on the right-hand side. A form like the one below will appear on the screen:

First, enter a username and password. Then, you'll be able to edit more user options.

Username *

Required: 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email *

Required: 100 characters or fewer. Email format only.

Password *

Your password can't be too similar to your other personal information.

Your password must contain at least 8 characters.

Your password can't be a commonly used password.

Your password can't be entirely numeric.

Password confirmation *

Enter the same password as before, for verification.

Save

Save and add another

Figure 4:User Add form

All fields must be filled in to create a user. If the Wallet is properly configured with an email service, the user in question should receive an email to change their password. Once a user is created, you must access their entry to set their permissions.

Permissions

Active☒

Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

Staff status☒

Designates whether the user can log into this admin site.

Superuser status☒

Designates that this user has all permissions without explicitly assigning them.

Groups

+

Available groups

Filter

OWNER
SERVICE
VIEWER

Choose all

Remove all

Chosen groups

The groups this user belongs to. A user will get all permissions granted to each of

Figure 5: User Permissions

In the permissions section, you can indicate whether the user account is active or not, as well as their superuser status. The staff status is required to access the administration interface; otherwise, the user

will only be able to interact with the APIs that require authentication. The "groups" section allows you to specify which group the user belongs to. Doing so will grant the user all the permissions associated with that group. By default, there are three permission groups:

- **OWNER:** This permission group allows the user to perform both read and write operations on the various data models of the Wallet.
- **SERVICE:** Only allows reading specific data models. This corresponds to the group intended for users employed by the "Entity Service" component.
- **VIEWER:** This permission group grants the user read permissions, but no write permissions.

The specific permissions of each group can be checked in the "Groups" section of the left-hand side menu. It is not necessary to assign a group to users if not desired.

After assigning the groups, we can indicate additional permissions for the user. These permissions are additive to the previous ones.

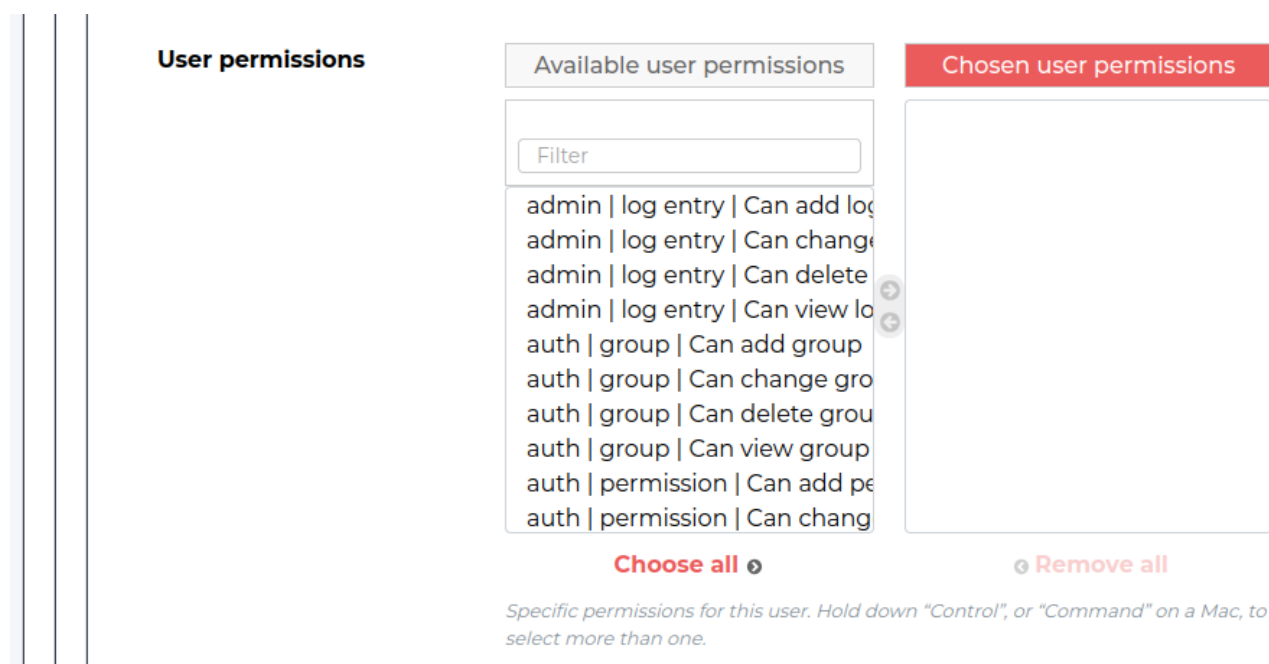


Figure 6: User Permissions - 2

2.2 Groups

The Wallet allows defining additional permission groups or modifying existing ones. To do this, you need to access the "Groups" option from the side menu.

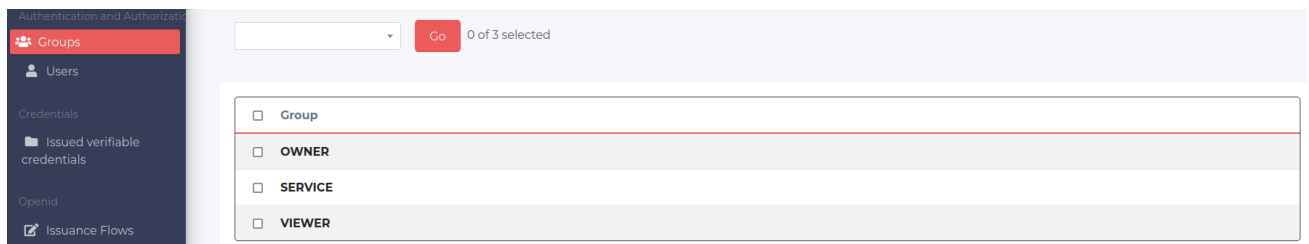


Figure 7: Group list

To add a new group, just like with users, click the “Add” button on the right-hand side.

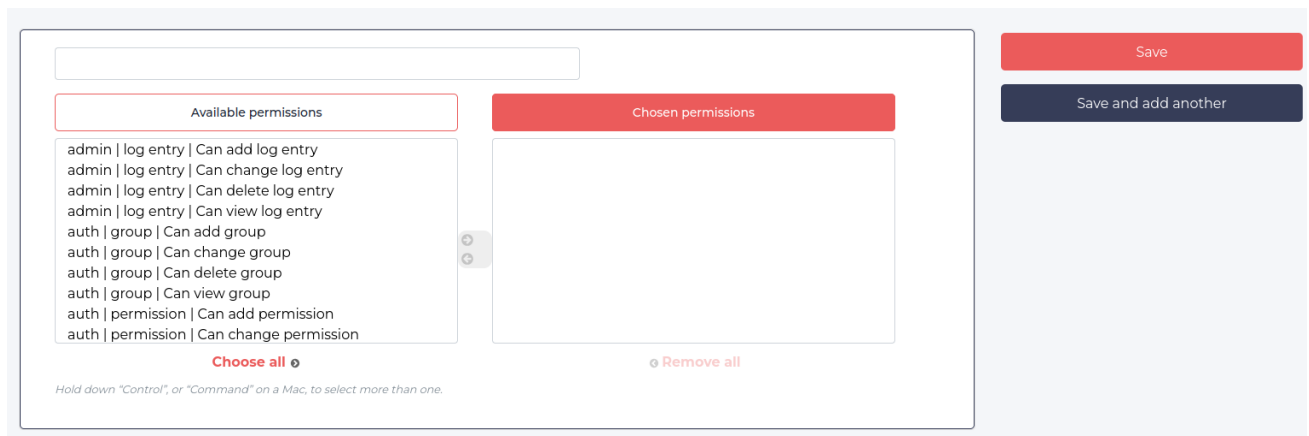


Figure 8: Group Add Form

You will need to provide the group's name in the top section and assign the associated permissions.

3 MANAGEMENT OF VERIFIABLE CREDENTIALS

When we access the administration interface with a user, the available menu options will depend on the associated permissions. For simplicity, this manual assumes that the user has permissions to perform all the operations that will be described from now on.

3.1 OpenID

All menu options related to the issuance or verification of verifiable credentials have been grouped under the same category named "OpenID".

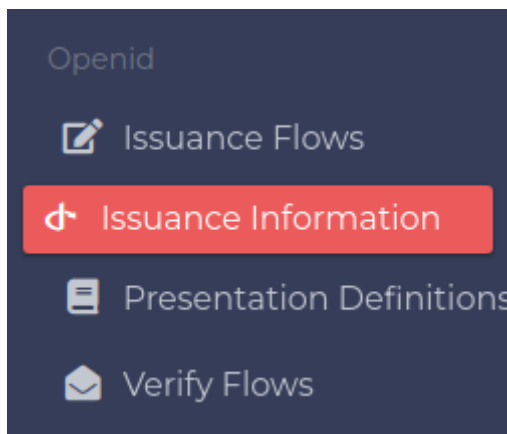


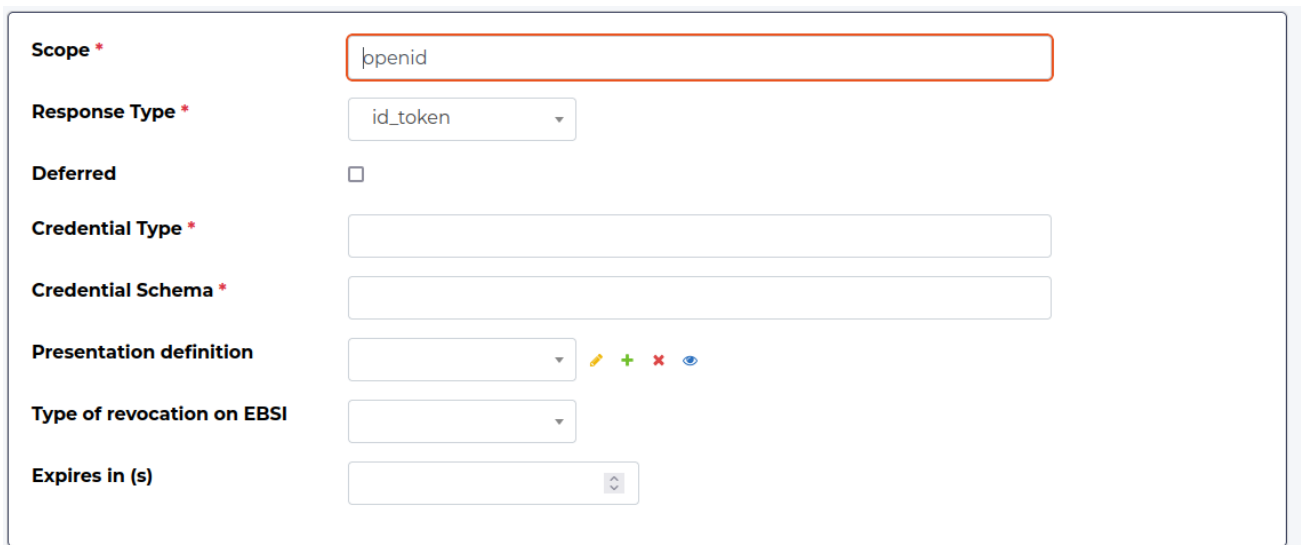
Figure 9: OpenID configurations

3.1.1 Defining an Issuance Flow

To define an issuance flow, we must access the option with the same name from the menu. Once inside, we add a new flow with the "add" button. The user will be asked to fill out a form with the following attributes:

- **Scope:** Defines the identifier of the operation, which the Holder Wallet must use to indicate its interest in this process. By default, "openid".
- **Response Type:** Specifies what type of token the user must provide to authenticate.
- **Deferred:** A flag that determines whether the credential will use the deferred flow.
- **Credential Type:** The name of the credential type, e.g., "MyAcademicId".
- **Credential Schema:** The URL to the schema specification.
- **Presentation Definition:** If a VP Token is requested, this section allows the specification of the presentation definition to be used.
- **Type of Revocation on EBSI:** Allow to select the type of revocation to be used. Currently, only StatusList is available.

- **Expires In:** This parameter allows us to specify when we want the issued credentials to expire.



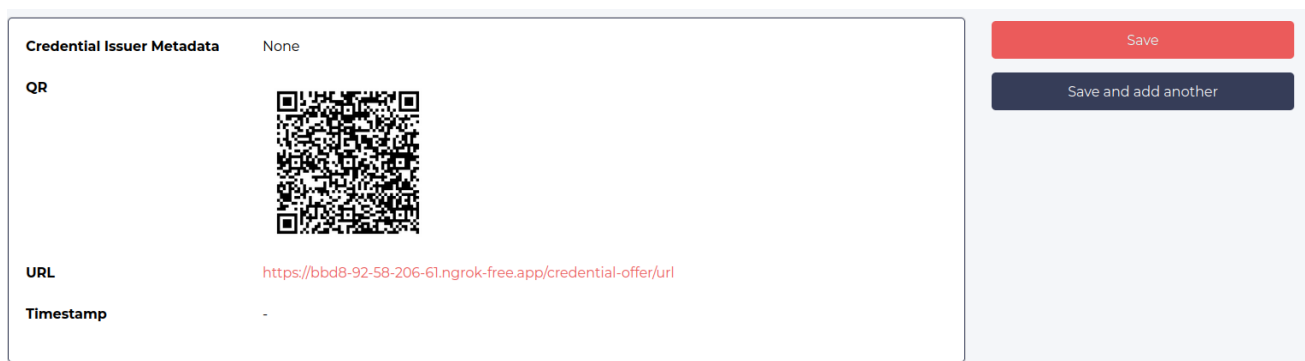
The screenshot shows a configuration form for an issuance flow. It includes the following fields and controls:

- Scope ***: A text input field containing the value "bpenid".
- Response Type ***: A dropdown menu with "id_token" selected.
- Deferred**: A checkbox that is currently unchecked.
- Credential Type ***: An empty text input field.
- Credential Schema ***: An empty text input field.
- Presentation definition**: A dropdown menu with a pencil icon, a plus icon, a minus icon, and an eye icon to its right.
- Type of revocation on EBSI**: A dropdown menu.
- Expires in (s)**: A text input field with a small up/down arrow icon on the right.

Figure 10: Issuance flow

3.1.2 Defining the Issuance Information

Once at least one issuance flow has been defined, the next step is to generate an "Issuance Information" entry. To do this, we must access the option with the same name in the menu and click "Add".



The screenshot shows the "Issuance information" form. It contains the following fields and controls:

- Credential Issuer Metadata**: A text input field with the value "None".
- QR**: A QR code.
- URL**: A text input field with the value "https://bbd8-92-58-206-61.ngrok-free.app/credential-offer/url".
- Timestamp**: A text input field with the value "-".
- Buttons**: Two buttons on the right side: "Save" (red) and "Save and add another" (dark blue).

Figure 11: Issuance information

As you can see, the form does not allow for modification. Simply click "save," and the system will automatically generate the relevant information.

If you interact again with the newly created entry:

Credential Issuer Metadata	<pre> { "authorization_server": "https://bbd8-92-58-206-61.ngrok-free.app", "credential_endpoint": "https://bbd8-92-58-206-61.ngrok-free.app/credentials/", "credential_issuer": "https://bbd8-92-58-206-61.ngrok-free.app", "credentials_supported": [{ "format": "jwt_vc", "types": ["VerifiableAttestation", "VerifiableCredential", "TestVC"] }], "deferred_credential_endpoint": "https://bbd8-92-58-206-61.ngrok-free.app/credential_ </pre>
QR	
URL	https://bbd8-92-58-206-61.ngrok-free.app/credential-offer/url

Figure 12: Issuance Information Entry

You will be able to view the information automatically generated by the Wallet. You do not need to create more instances of this model. Each time you update an issuance flow or add a new one, the system will automatically update the "Issuance Information" instance.

3.1.3 Defining a Verify Flow

To define a verification flow, go to the "Verify Flow" section of the menu. In this case, the form to be filled out is as follows:

- **Scope**: Defines the scope of the operation, which also acts as the operation's identifier.
- **Response Type**: Specifies what type of token the user must provide to authenticate. Generally, VP Token will be used, but the option of ID Token is allowed to complete EBSI conformance tests.
- **Presentation Definition**: This section allows for the specification of the presentation definition to be used. Only necessary if a VP Token is requested.
- **ID**: This is the unique identifier of the verification process, which is required to obtain the VP offer. It is automatically assigned.

Figure 13: Verify flow

3.1.4 Defining the Presentation Definition

To use VP Tokens, whether in an issuance or verification process, it is necessary to define at least one Presentation Definition. To do this, you must access the menu option with the same name. Below is a description of what each of the fields shown when adding a new entry to the database corresponds to:

- **Identifier:** Defines the unique identifier of the definition. The user can specify it, but if not defined, the Wallet will generate a default UUID.
- **Format:** This is a JSON object where the format in which the credential must be delivered for validation is defined. For example:

```
{
  "jwt_vc": {
    "alg": [
      "ES256"
    ]
  },
  "jwt_vp": {
    "alg": [
      "ES256"
    ]
  }
}
```

- **Input Descriptors:** This is a JSON object where the credential data to be presented is defined, as well as any restrictive data, meaning data that must be of a certain type. For example:

```
[
  {
    "id": "<any id, random or static>",
    "format": {
      "jwt_vc": {
        "alg": [
          "ES256"
        ]
      }
    },
    "constraints": {
      "fields": [
        {
          "path": [
```

```
    "$.vc.type"  
  ],  
  "filter": {  
    "type": "array",  
    "contains": {  
      "const": "VerifiableAttestation"  
    }  
  }  
}  
]  
}  
]  
]  
]
```

Identifier

Format *

Ln: 1 Col: 1

1 hull

Input Descriptors *

Ln: 1 Col: 1

1 hull

Figure 14: Presentation Definition