Smart5Grid

Demonstration of **5G** solutions for
SMART energy **GRID**s of the future

PLATFORM

OSR Platform User Guide

Abstract
This document provides a detailed overview of the functionalities and features implemented in the Smart5Grid platform with examples for NetworkApp creation and testing

5G PPP
The 5G Infrastructure Public Private Partnership

This project has received funding from the European
Union's *Horizon 2020 research and innovation programme*
under grant agreement n° 101016912

Disclaimer This document reflects the Smart5Grid consortium view and the European Commission (or the 5G-Public Private Partnership) is not responsible for any use that may be made of the information it contains

Contents

1. Introduction

This document aims to provide a comprehensive guide to the functionalities implemented in the Smart5Grid Platform. The primary objective of the platform is to create a centralized space for application developers and consumers in the energy grid applications market. To achieve this, Smart5Grid will develop a platform that includes a repository of thoroughly tested NetworkApps, which will be made available for consumers to utilize. The user interface (UI) of the platform relies on the underlying components of the Open Service Repository (OSR) and the Verification and Validation (V&V) platforms.

The OSR serves as an open Network App repository where Network Apps and their subcomponents such as Network Services (NSs), Virtual Network Functions (VNFs), Virtual Deployment Units (VDUs), charts, and images are stored. Additionally, the OSR implements an Authentication and Authorization (A&A) mechanism, ensuring that users can only access resources based on their defined roles.

The V&V Framework within the Smart5Grid platform facilitates the testing of NetworkApps. This framework is supported by a telecommunications and virtualization infrastructure, which forms the second layer of the Smart5Grid Platform. This infrastructure hosts the deployment of NetworkApp instances for validation purposes. Both the infrastructure and the deployment of NetworkApps are managed by a Management and Orchestration (M&O) framework. The M&O framework encompasses the necessary systems to interpret, deploy, and monitor the functions described by the NetworkApp descriptor throughout its lifecycle.

2. Technologies

The Smart5Grid platform is built using the React framework, providing a robust foundation for creating dynamic and interactive user interfaces. To ensure scalability and flexibility, the platform is deployed within a Kubernetes cluster, which offers powerful orchestration capabilities. This same Kubernetes cluster also hosts the OSR Harbor and GitLab, seamlessly integrating container registry and version control functionalities into the platform's ecosystem.

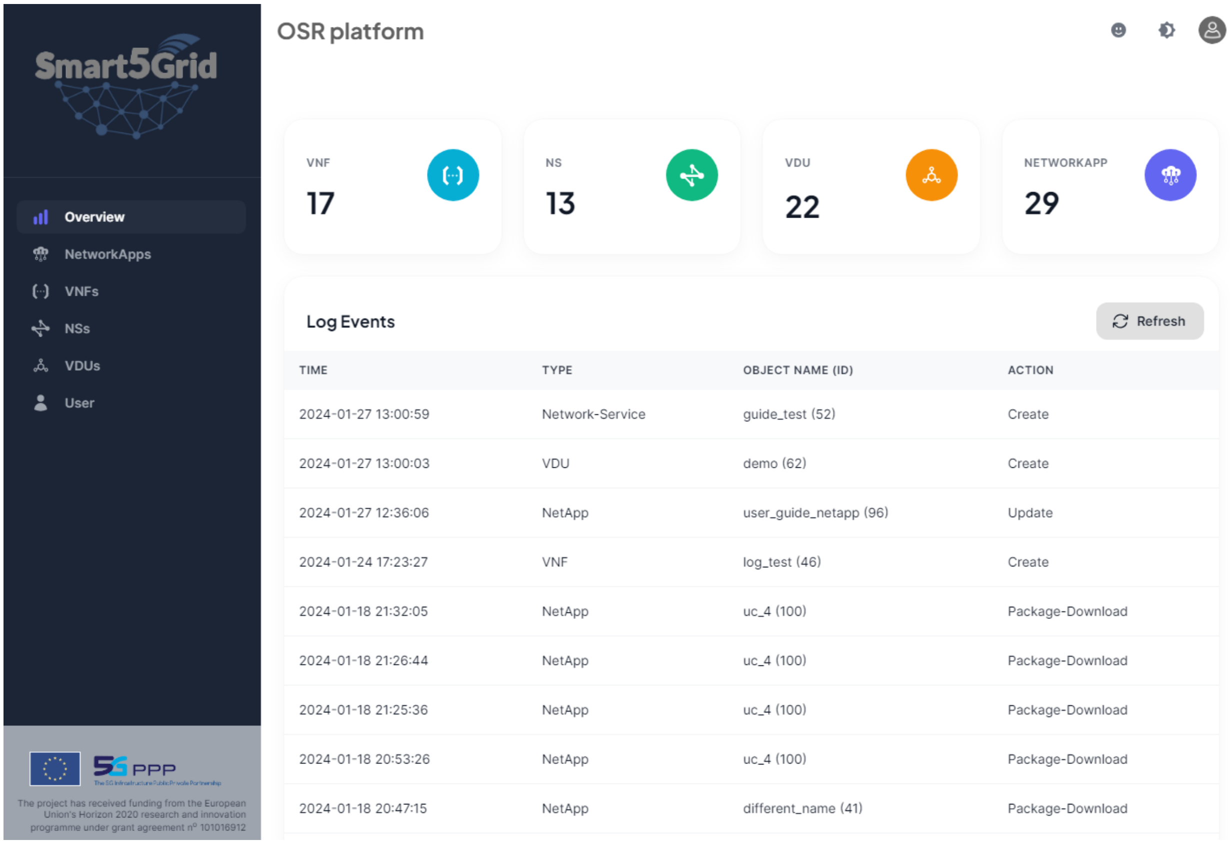
By leveraging the Kubernetes cluster, the Smart5Grid platform can effortlessly scale its resources in response to user demands. Whether it's an increase in user traffic or a need for additional capacity, the platform can easily adapt to meet these requirements. With Kubernetes' horizontal scaling capabilities, you can dynamically add or remove instances of the platform, ensuring optimal performance and efficient resource utilization.

Furthermore, the integration of OSR Harbor and GitLab within the same cluster enhances the development and deployment workflows. The OSR Harbor acts as a centralized repository for storing and managing container images used by the platform, while GitLab provides robust version control and collaboration features for the source code.

In summary, the Smart5Grid platform, written in React and deployed in a Kubernetes cluster alongside OSR Harbor and GitLab, offers a scalable and adaptable solution. Its seamless integration with Kubernetes allows for easy capacity adjustments based on user needs, while the inclusion of OSR Harbor and GitLab streamlines the development and deployment processes.

3. Structure of the Platform UI

Our OSR platform is crafted using React and adheres to the principles of Material Design for a resilient and high-speed interface. The page layout is meticulously designed to be highly responsive across various devices. Moreover, we've integrated a dark theme option to enhance user experience and accommodate different preferences.



The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement nº 101016912

VNF 17

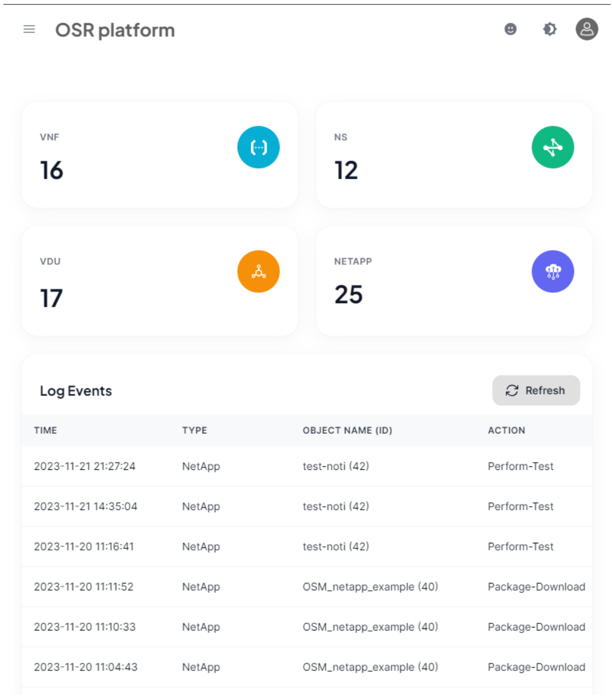
NS 13

VDU 22

NETWORKAPP 29

Log Events

TIME	TYPE	OBJECT NAME (ID)	ACTION
2024-01-27 13:00:59	Network-Service	guide_test (52)	Create
2024-01-27 13:00:03	VDU	demo (62)	Create
2024-01-27 12:36:06	NetApp	user_guide_netapp (96)	Update
2024-01-24 17:23:27	VNF	log_test (46)	Create
2024-01-18 21:32:05	NetApp	uc_4 (100)	Package-Download
2024-01-18 21:26:44	NetApp	uc_4 (100)	Package-Download
2024-01-18 21:25:36	NetApp	uc_4 (100)	Package-Download
2024-01-18 20:53:26	NetApp	uc_4 (100)	Package-Download
2024-01-18 20:47:15	NetApp	different_name (41)	Package-Download



VNF 16

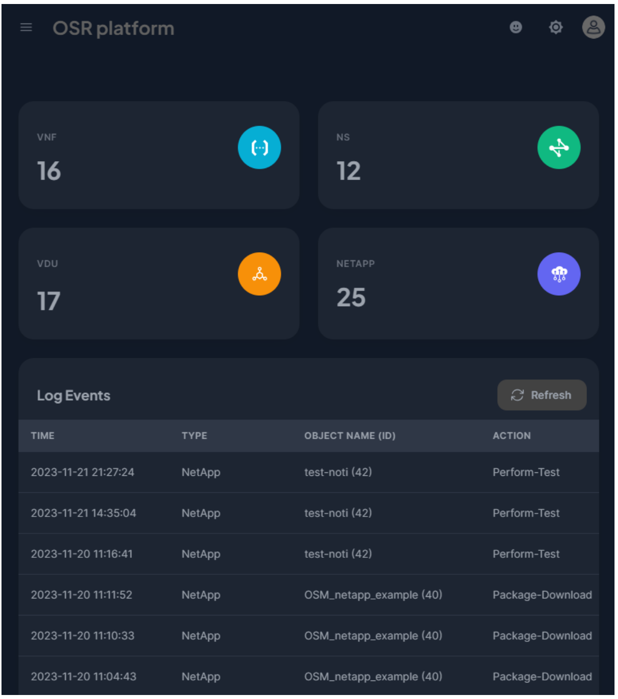
NS 12

VDU 17

NETAPP 25

Log Events

TIME	TYPE	OBJECT NAME (ID)	ACTION
2023-11-21 21:27:24	NetApp	test-noti (42)	Perform-Test
2023-11-21 14:35:04	NetApp	test-noti (42)	Perform-Test
2023-11-20 11:16:41	NetApp	test-noti (42)	Perform-Test
2023-11-20 11:11:52	NetApp	OSM_netapp_example (40)	Package-Download
2023-11-20 11:10:33	NetApp	OSM_netapp_example (40)	Package-Download
2023-11-20 11:04:43	NetApp	OSM_netapp_example (40)	Package-Download



VNF 16

NS 12

VDU 17

NETAPP 25

Log Events

TIME	TYPE	OBJECT NAME (ID)	ACTION
2023-11-21 21:27:24	NetApp	test-noti (42)	Perform-Test
2023-11-21 14:35:04	NetApp	test-noti (42)	Perform-Test
2023-11-20 11:16:41	NetApp	test-noti (42)	Perform-Test
2023-11-20 11:11:52	NetApp	OSM_netapp_example (40)	Package-Download
2023-11-20 11:10:33	NetApp	OSM_netapp_example (40)	Package-Download
2023-11-20 11:04:43	NetApp	OSM_netapp_example (40)	Package-Download

The functionalities of the Smart5Grid Platform UI can be categorized into the following components. By organizing the functionalities into these distinct components, the Smart5Grid Platform offers a user-friendly interface that allows users to efficiently navigate and interact with different aspects of the platform.

3.1. User Authentication and Registration

To access the platform, users are required to enter their credentials. New users have the option to create a new account through the registration process.

3.2. Main Page – Overview

The main page provides users with a comprehensive overview of all the resources available in the Open Service Repository (OSR). Additionally, it displays the most recent log events for quick reference.

3.3. User Details

On this page, users can view their personal information, along with general details about the repository and registry.

3.4. Virtual Deployment Units (VDUs)

This page presents detailed information about Virtual Deployment Units (VDUs) and allows users to perform various actions related to them.

3.5. Virtual Network Functions (VNFs)

Users can access this page to view comprehensive information about Virtual Network Functions (VNFs) and perform relevant actions.

3.6. Network Services (NSs)

The Network Services (NSs) page provides users with detailed information about NSs and enables them to perform specific actions related to NSs.

3.7. Network Applications (NetworkApps)

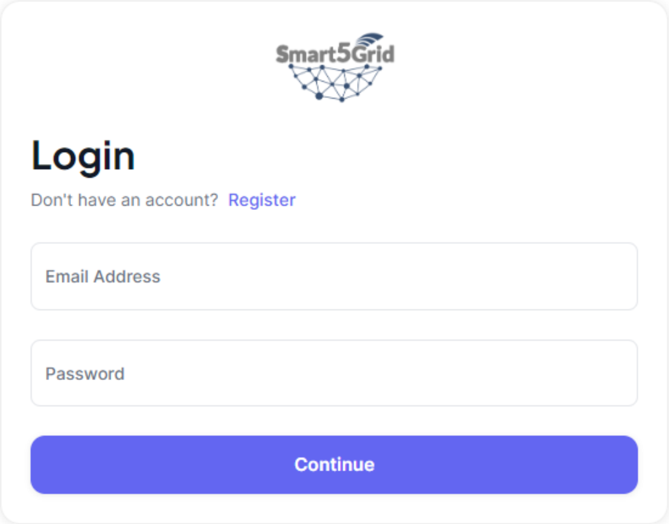
On this page, users can explore comprehensive information about Network Applications (NetworkApps) and perform various actions associated with them.

4. User Authentication and Registration

The Smart5Grid platform is publicly available through the following URI: https://platform.s5g.axonlogic.eu/. To utilize the platform's functionalities, users are required to authenticate themselves. Authentication and administration mechanisms are provided by the OSR platform. This ensures secure access to the Smart5Grid platform, allowing authorized users to make full use of its features and capabilities. By leveraging the authentication mechanism offered by the OSR platform, the Smart5Grid platform maintains a robust and secure environment for users to interact with energy grid applications effectively.

4.1. Login

This page provides a user-friendly interface for entering credentials and gaining authenticated access to the platform's functionalities.



Login

Don't have an account? Register

Email Address

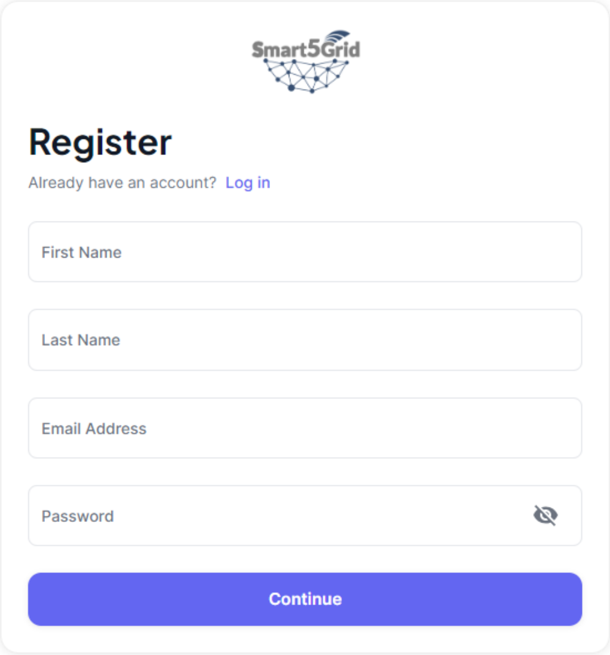
Password

Continue

You must enter your email and your password and press the **continue** to start using the Smart5Grid platform. If you don't have an account, you must press the **Register** button in order to create a new one.

4.2. Register

Register page can be reached either by pressing the register button from the login page, or via the URI: https://platform.s5g.axonlogic.eu/auth/register.



Register

Already have an account? Log in

First Name

Last Name

Email Address

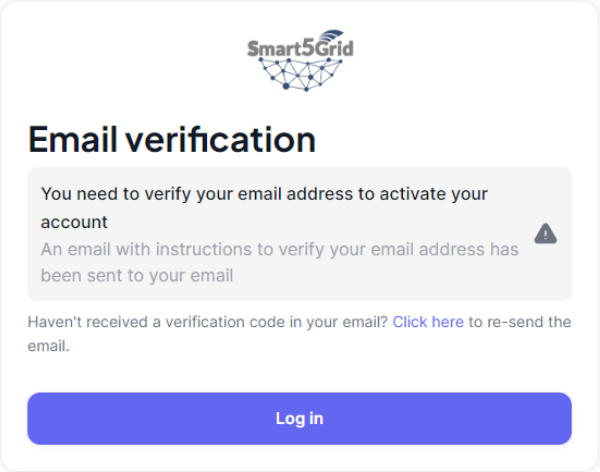
Password

Continue

To create a new account on the Smart5Grid platform, you will need to provide the following information:

- First name
- Last name
- Email address
- Password

Once you have entered these details, click on the **Continue** button. This will redirect you to the next page in the account creation process.



Email verification

You need to verify your email address to activate your account

An email with instructions to verify your email address has been sent to your email

Haven't received a verification code in your email? Click here to re-send the email.

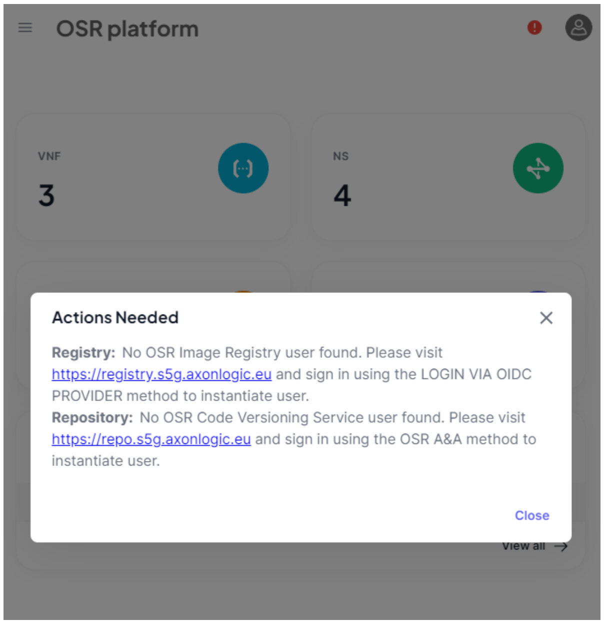
To complete the user creation process, it is necessary to verify your email address. A verification email will be sent to the provided email account. Please follow these steps:

1. Check your inbox for the verification email. If you do not see it in your inbox, please check your junk or spam folder as it may have been filtered.
2. Open the verification email and click on the provided link. This will confirm your email address and successfully register your account.

If you have not received the verification email, you can click on the "**Click here**" button to resend the email. Please ensure that you have entered the correct email address during the account creation process. By completing the email verification step, you may press the login button to be redirected to the login page.

4.3. Instantiation

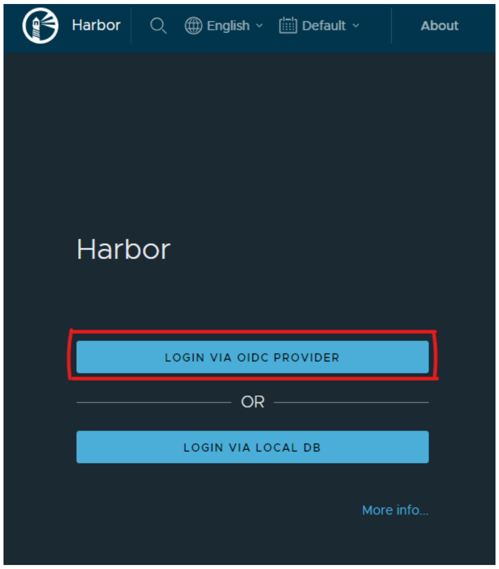
When you first start using the platform, you will notice a red question mark icon located at the top right corner of the interface. This icon serves as an alert indicator. By clicking on it, you will be presented with the following information:

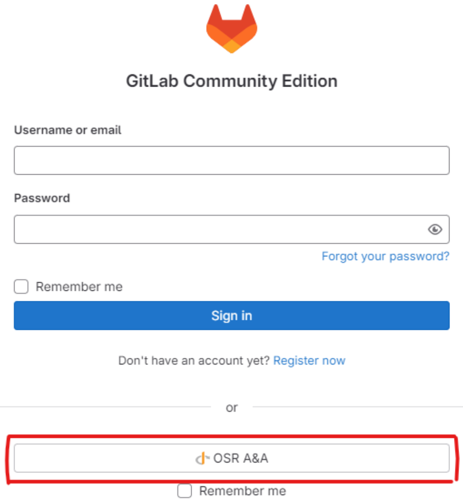


Actions Needed

Registry: No OSR Image Registry user found. Please visit https://registry.s5g.axonlogic.eu and sign in using the LOGIN VIA OIDC PROVIDER method to instantiate user.
Repository: No OSR Code Versioning Service user found. Please visit https://repo.s5g.axonlogic.eu and sign in using the OSR A&A method to instantiate user.

OSR is using as an image registry the Harbor and as code repository the Gitlab. After the user creation you have to login with your login credentials that you created at previous step at the harbor: https://registry.s5g.axonlogic.eu/ and at the gitlab: https://repo.s5g.axonlogic.eu/. Please select the **Login via OIDC Provider** and **OSR A&A** respectively.

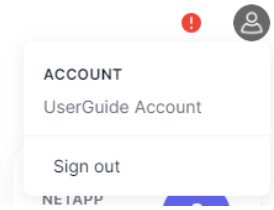




This step will instantiate the local databases of the gitlab and harbor respectively. After having completed these steps you are ready to start using the Smart5Grid platform. Please keep in mind that the red alert will not disappear until your next login.

4.4. Logout

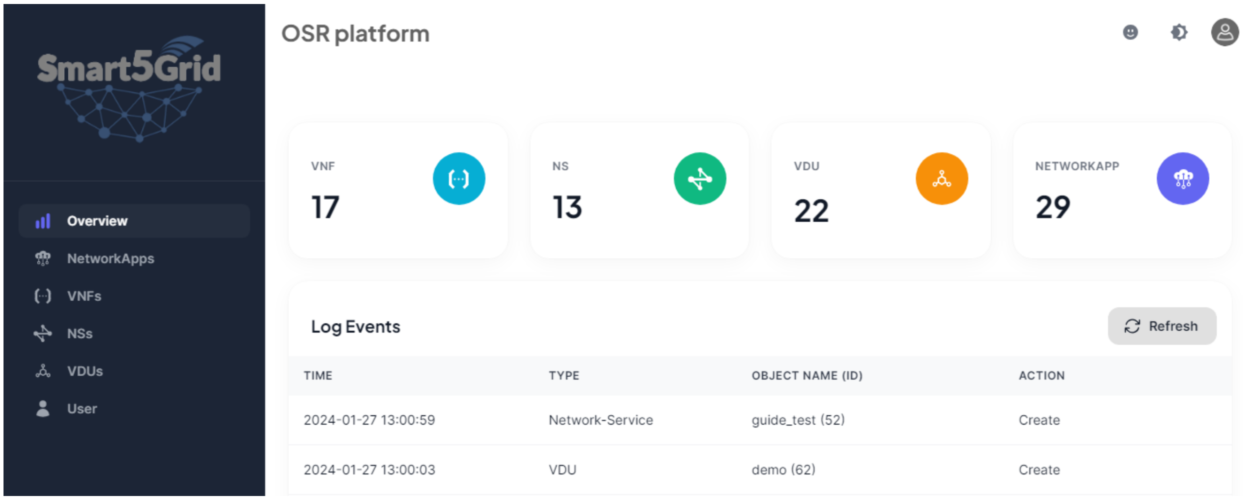
You can logout via pressing the top right icon:



Press the **Sign out** button and you will be redirected to the login page.

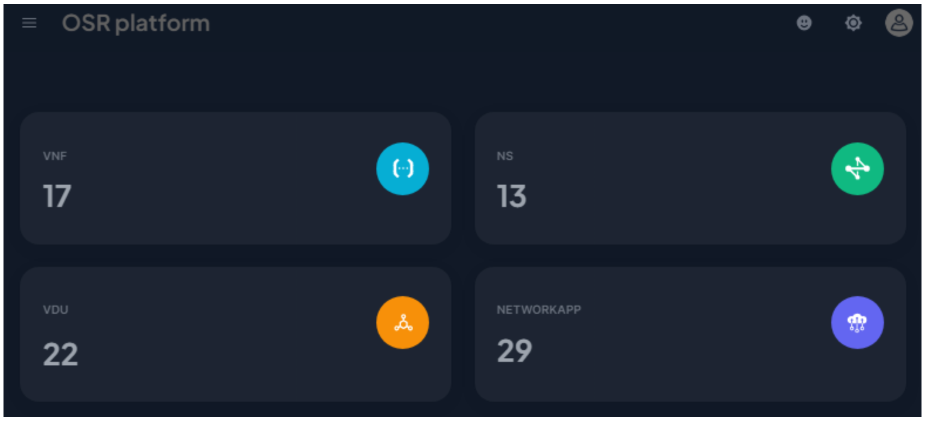
5. Main Page – Overview

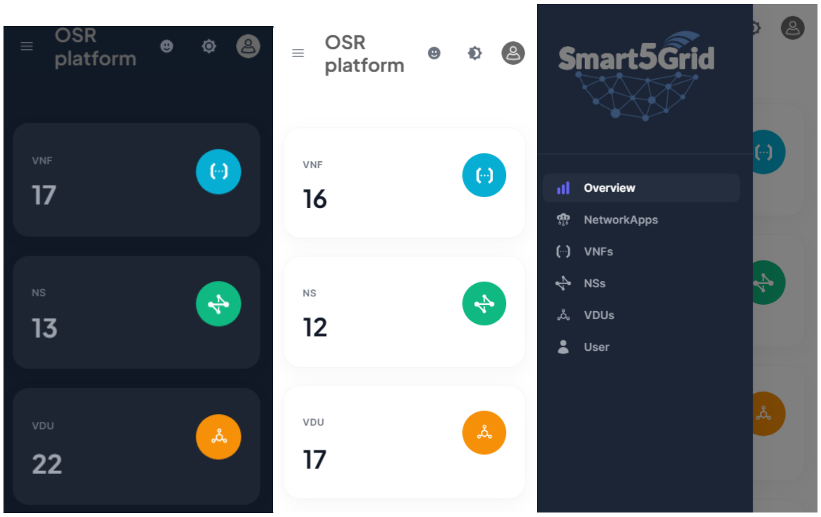
After successful login you are redirected to the main page. From this page you can see some general info regarding the platform, such us the number of the resources (VNF, NS, NETWORKAPP, VDU) and the latest log events.



TIME	TYPE	OBJECT NAME (ID)	ACTION
2024-01-27 13:00:59	Network-Service	guide_test (52)	Create
2024-01-27 13:00:03	VDU	demo (62)	Create

You have the option to enable auto-refresh for logs at predefined intervals of 5 or 30 seconds by simply clicking the Refresh button. To navigate through individual elements, you can access their respective pages in two ways: either by selecting the resource counters displayed in their respective boxes or by utilizing the list of available pages located on the right-hand side of the screen. Note that the list may be hidden depending on your screen size. In such cases, simply click the burger button to reveal the pages.

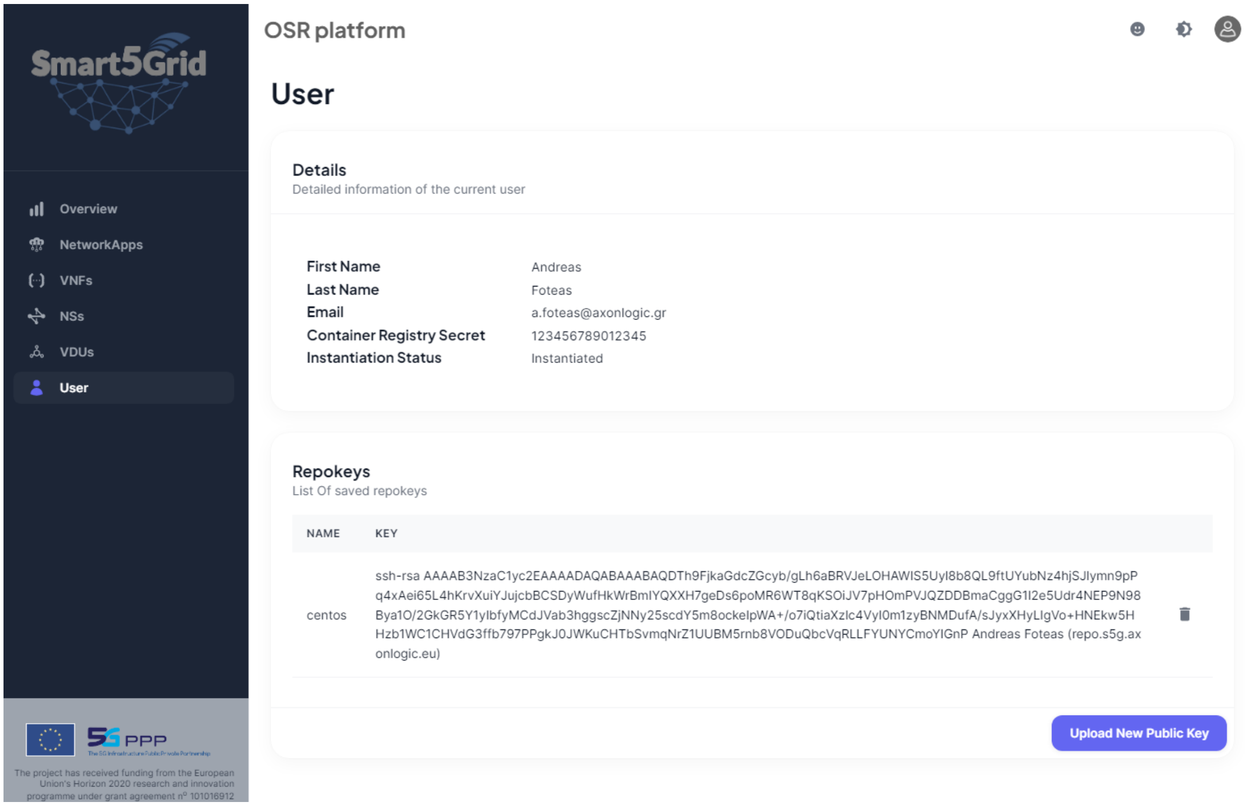




Finally, you can set the dark of light theme by pressing on the sun/moon button.

6. User Details

The User Details page provides fundamental information about the user. Below is an example screenshot showcasing its layout:


Smart5Grid
Overview
NetworkApps
VNFs
NSs
VDUs
User
5G PPP
The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement nº 101016912
OSR platform

User

Details

Detailed information of the current user

First Name	Andreas
Last Name	Foteas
Email	a.foteas@axonlogic.gr
Container Registry Secret	123456789012345
Instantiation Status	Instantiated

Repokeys

List Of saved repokeys

NAME	KEY
centos	ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDTh9FjkaGdcZGcyb/gLh6aBRVJeLOHAWIS5Uyl8b8QL9ftUYubNz4hjSJlymn9pPq4xAei65L4hKrvXuiYJujcbBCSDyWufHkWrBmIYQXXH7geDs6poMR6WT8qKSOiJV7pHOmPVJQZDDBmaCggG1I2e5Udr4NEP9N98Bya1O/2GkGR5Y1yIbfyMCdJVab3hggscZjNNy25scdY5m8ockeIpWA+/o7iQtiaXzIc4VyI0m1zyBNMDufA/sJyxXHyLIgVo+HNEkw5HHzb1WC1CHVdG3ffb797PPgkJ0JWKuCHTbSvmqNrZ1UUBM5rnb8VODuQbcVqRLLFYUNYCmoYIGnP Andreas Foteas (repo.s5g.axonlogic.eu)

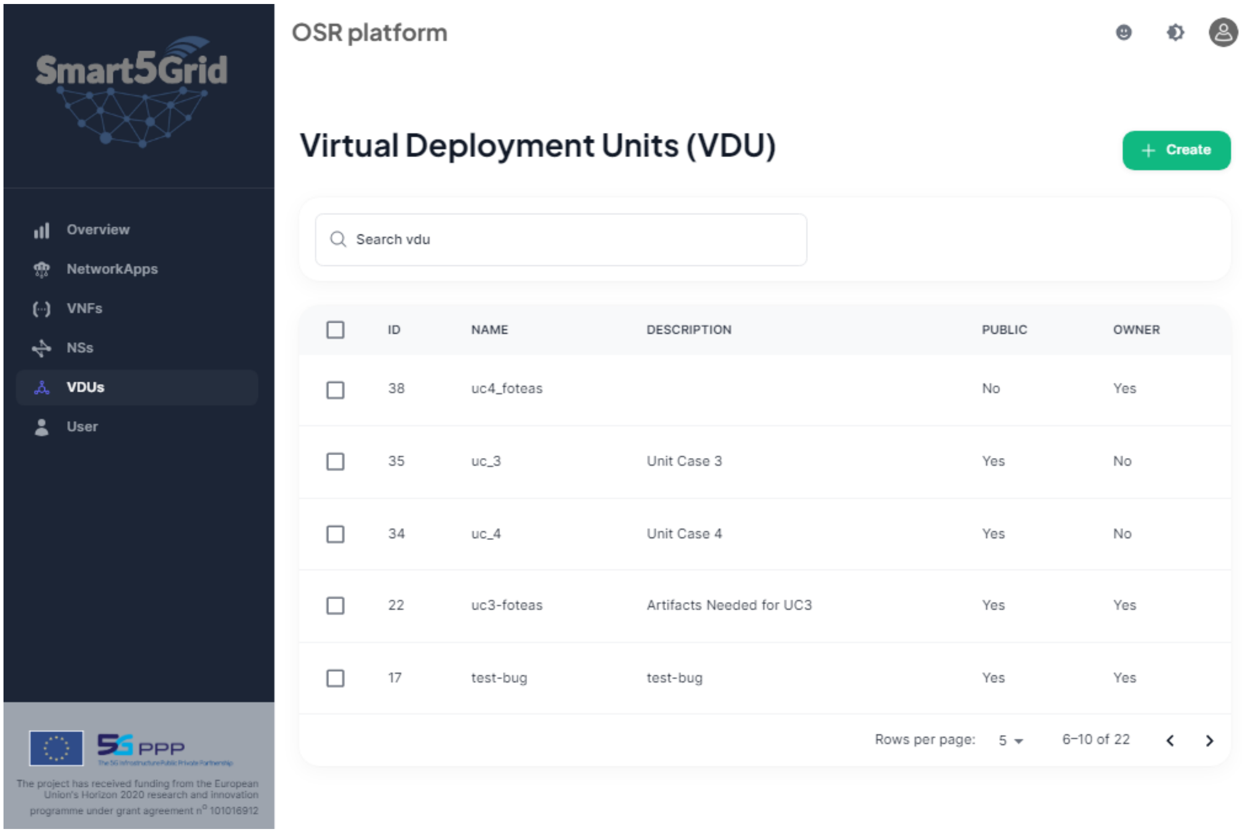
Apart from the First Name Last Name and Email that are self-explanatory, there are the following fields:

- **Container Registry Secret:** The password needed to login to the docker container registry if connection via cli is needed.
- **Instantiation Status:** Information regarding further actions needed to instantiate the Harbor and the Gitlab.

Finally, the repo-keys container allows users to manage their private keys, enabling SSH access for GitLab. Users have the capability to both add and delete private keys. Upon uploading a new key, it is automatically saved under the respective user's profile on GitLab, ensuring seamless access.

7. Virtual Deployment Units (VDUs)

At the Virtual Deployment Units (VDUs) page there is the list of all the created VDUs at the OSR. Additionally, there's a convenient Search Field feature available for locating a specific VDU swiftly. This search functionality operates across all columns within the array, enabling users to input a keyword that will be applied comprehensively across the dataset, facilitating quick and efficient VDU identification. A Virtual Deployment Unit (VDU) is essentially a package comprising a GitLab project and a Harbor project that share identical names and access permissions. Its primary purpose is to store essential artifacts required for each Network Application, serving as a repository to house these elements. Additionally, the VDU facilitates version control, allowing for the preservation and tracking of modifications made on descriptors over time. During the creation of each Network Application, it's necessary to assign a Virtual Deployment Unit (VDU) to it. This linkage ensures that every Network Application is associated with a specific VDU. Moreover, it's important to note that a single VDU can serve multiple Network Applications, allowing for shared utilization across different applications within the system.



Virtual Deployment Units (VDU)

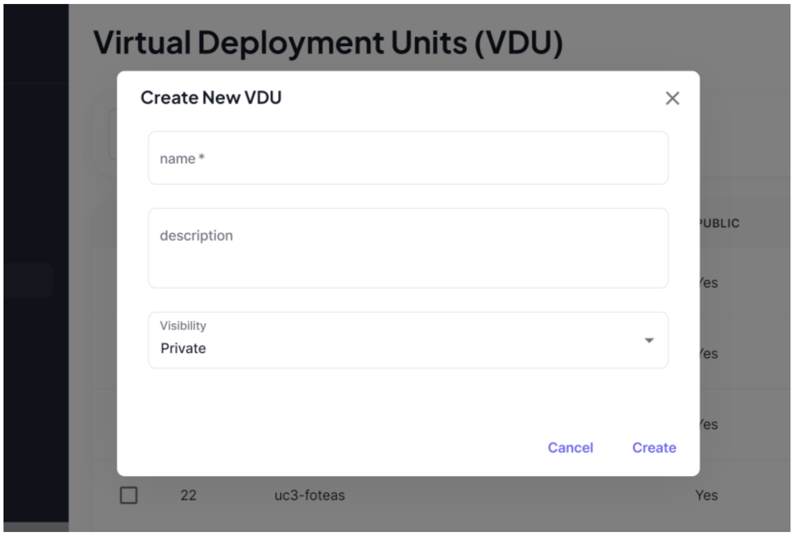
☐	ID	NAME	DESCRIPTION	PUBLIC	OWNER
☐	38	uc4_foteas		No	Yes
☐	35	uc_3	Unit Case 3	Yes	No
☐	34	uc_4	Unit Case 4	Yes	No
☐	22	uc3-foteas	Artifacts Needed for UC3	Yes	Yes
☐	17	test-bug	test-bug	Yes	Yes

Rows per page: 5 ▾ 6–10 of 22 ‹ ›

The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 101016912

7.1. Create

To create a new VDU, simply click the green "Create" button. Upon clicking, a dialog box will be displayed, guiding you through the creation process.



Virtual Deployment Units (VDU)

Create New VDU

name *

description

Visibility: Private

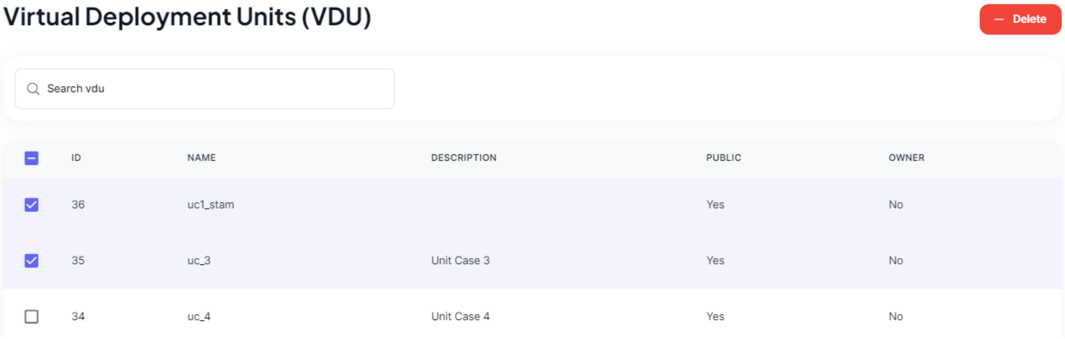
The fields available for a new VDU are:

- Name (mandatory)
- Description (optional)
- Visibility (mandatory)

Upon the creation of a new Virtual Deployment Unit (VDU), a corresponding entry will be generated within the OSR database, establishing a dedicated record for this VDU. Simultaneously, the creation process will trigger the generation of a new Harbor project alongside the formation of a new group and GitLab project. These newly created projects and groups will have visibility and permissions linked directly to the recently instantiated VDU object, ensuring alignment and access control across these interconnected elements within the system.

7.2. Delete

To delete a VDU, users are required to select one or more VDUs by checking the corresponding checkboxes in the list. Subsequently, pressing the "Delete" button initiates the deletion process.



Virtual Deployment Units (VDU)

▣	ID	NAME	DESCRIPTION	PUBLIC	OWNER
☑	36	uc1_stam		Yes	No
☑	35	uc_3	Unit Case 3	Yes	No
□	34	uc_4	Unit Case 4	Yes	No

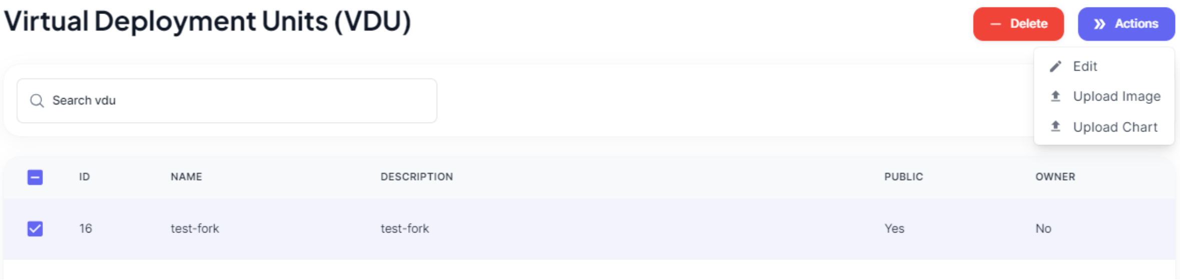
When a Virtual Deployment Unit (VDU) is deleted, the associated projects within Harbor and GitLab are also removed as part of the deletion process. However, there are certain conditions to consider:

- VDU Ownership: A VDU cannot be deleted if it is public and not owned by the platform user. Only the owner of a public VDU can initiate its deletion.
- VDU Assignment to NetworkApp: A VDU that is assigned to a NetworkApp cannot be deleted due to its association with this application. This restriction prevents the deletion of a VDU crucial for the functionality of a NetworkAapp.

7.3. Actions

When having selected only one VDU from the checkbox of the list the **Actions** button is appeared. The available options are:

- Edit
- Upload Image
- Upload Chart



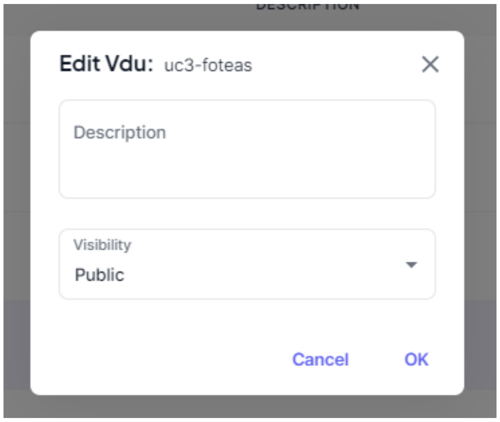
Virtual Deployment Units (VDU)

☐	ID	NAME	DESCRIPTION	PUBLIC	OWNER
☑	16	test-fork	test-fork	Yes	No

7.3.1. Edit

When pressed the edit action o popup window is show enabling to edit the fields:

- Description
- Visibility



Edit Vdu: uc3-foteas

Description

Visibility: Public

Only VDU owned by the platform user can be edited.

7.3.2. Upload Image

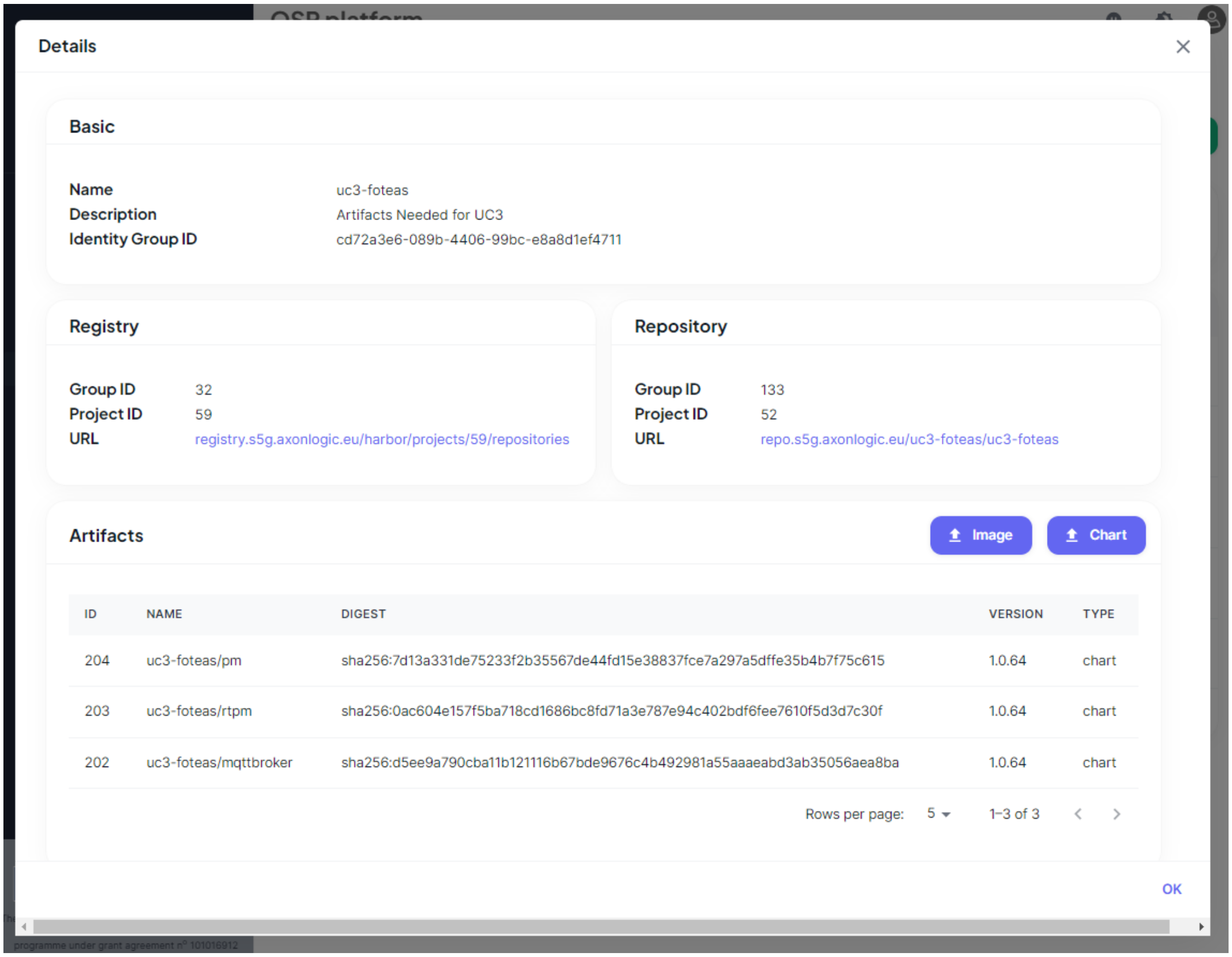
Upon selecting the "Upload Image" option, a popup window will appear, enabling users to choose a container image file stored within their computer storage. The chosen file must be of type .tar.gz, indicating a container image file format. Following selection, the uploaded image will be stored within the associated Harbor Project, aligning with the respective Virtual Deployment Unit (VDU).

7.3.3. Upload Chart

Upon choosing the "Upload Chart" option, a popup window will appear, allowing users to select a Helm chart file stored within their local storage. It's important to note that the chosen file should be of type .tgz, denoting a Helm chart file format. Once selected, the uploaded chart will be stored within the associated Harbor Project, ensuring alignment with the respective Virtual Deployment Unit (VDU).

7.4. Details

Clicking on any of the elements within the Virtual Deployment Unit (VDU) will prompt a dialog to appear, presenting comprehensive details regarding the selected VDU element.



Details

Basic

Name uc3-foteas
Description Artifacts Needed for UC3
Identity Group ID cd72a3e6-089b-4406-99bc-e8a8d1ef4711

Registry

Group ID 32
Project ID 59
URL registry.s5g.axonlogic.eu/harbor/projects/59/repositories

Repository

Group ID 133
Project ID 52
URL repo.s5g.axonlogic.eu/uc3-foteas/uc3-foteas

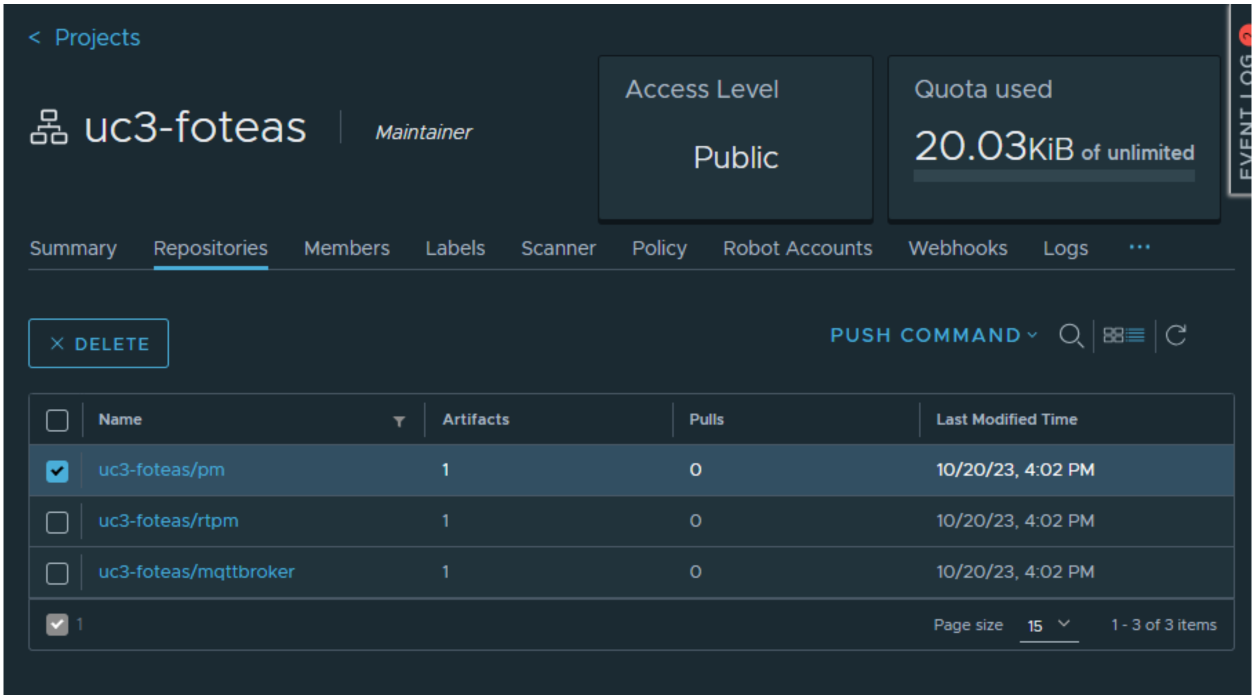
Artifacts

ID	NAME	DIGEST	VERSION	TYPE
204	uc3-foteas/pm	sha256:7d13a331de75233f2b35567de44fd15e38837fce7a297a5dffe35b4b7f75c615	1.0.64	chart
203	uc3-foteas/rtpm	sha256:0ac604e157f5ba718cd1686bc8fd71a3e787e94c402bdf6fee7610f5d3d7c30f	1.0.64	chart
202	uc3-foteas/mqttbroker	sha256:d5ee9a790cba11b121116b67bde9676c4b492981a55aaaeabd3ab35056aea8ba	1.0.64	chart

Rows per page: 5 1–3 of 3

In the Basic Card, essential details about the VDU are displayed, including its name, description, and the Identity Group ID. Within the Registry Card, information related to the GitLab project is showcased, along with a direct link to access the project. The Repository Card presents details about the Harbor project, along with a link to navigate to the respective Harbor project.

Furthermore, in the Artifacts Card, a comprehensive list of all uploaded artifacts currently residing in Harbor is provided. Additionally, there are two buttons available—one for uploading a chart and another for uploading an image. It's important to note that to delete an artifact, users are redirected to the Harbor UI, where they can perform the deletion action within the Harbor interface.



uc3-foteas | *Maintainer*

Access Level: Public

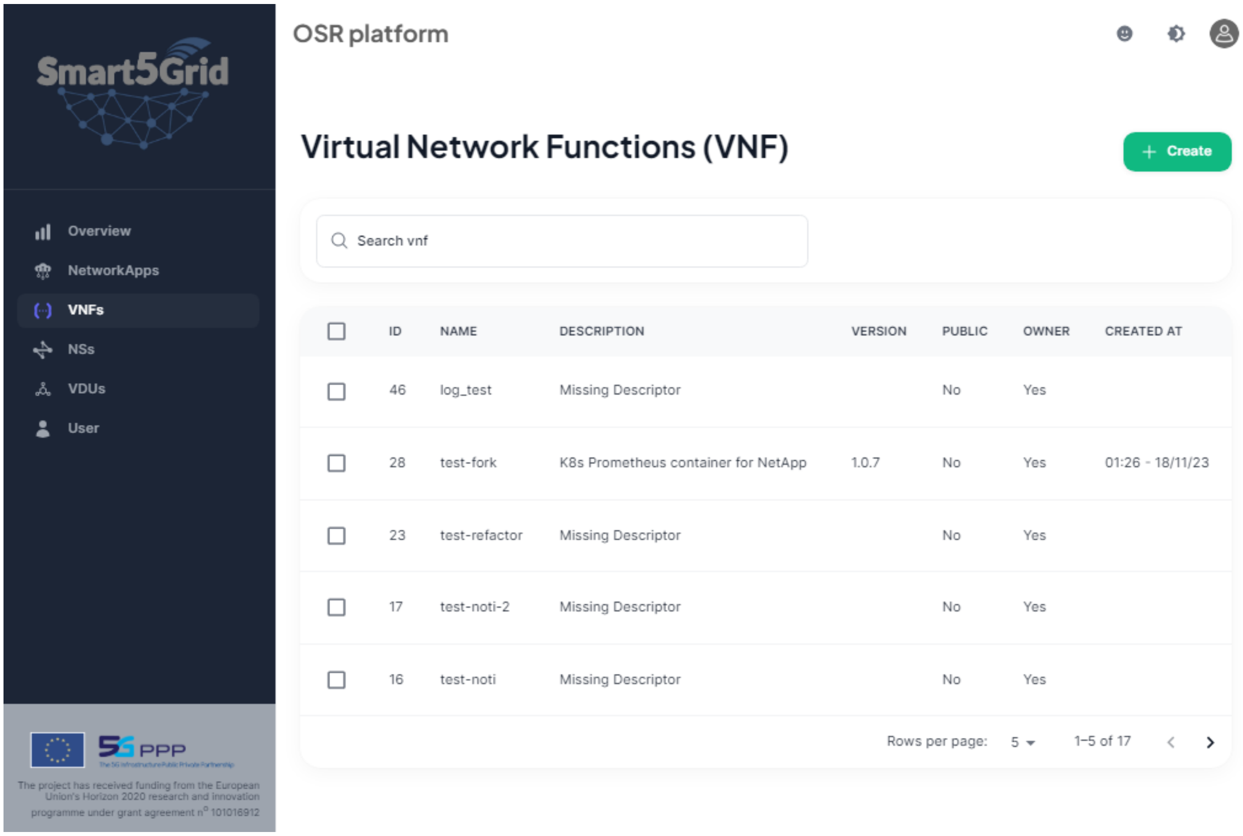
Quota used: 20.03KiB of unlimited

☐	Name	Artifacts	Pulls	Last Modified Time
☑	uc3-foteas/pm	1	0	10/20/23, 4:02 PM
☐	uc3-foteas/rtpm	1	0	10/20/23, 4:02 PM
☐	uc3-foteas/mqttbroker	1	0	10/20/23, 4:02 PM

☑ 1 Page size 15 1 - 3 of 3 items

8. Virtual Network Functions (VNFs)

At the Virtual Network Functions (VNFs) page in the platform, users can access a comprehensive list of all created VNFs. Moreover, there's a Search Field feature designed for easy and rapid retrieval of specific VNFs. This search functionality is capable of scanning across all columns within the dataset, allowing users to input keywords for a thorough search, thus enabling efficient and swift identification of the desired VNFs. VNFs refers to a software-based implementation of network functions that traditionally were performed by dedicated hardware devices. VNFs operate within a virtualized environment and perform specific network tasks or functions such as routing, firewalling, load balancing, and other network services. These functions are virtualized, allowing for flexibility, scalability, and easier management within modern software-defined networking (SDN) and network function virtualization (NFV) architectures. VNFs are essential components in Smart5Grid, enabling the creation and deployment of network services using software-based solutions rather than relying solely on physical hardware. OSR has the capability to store and manage versioning of descriptors specifically tailored for Virtual Network Functions (VNFs). These descriptors serve as templates or specifications outlining the configuration, behavior, and requirements of individual VNFs. Storing and versioning these descriptors in the OSR system is crucial as they act as blueprints or reference points for the creation and deployment of Network Services (NS) and Network Applications (NetworkApps).



The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 101016912

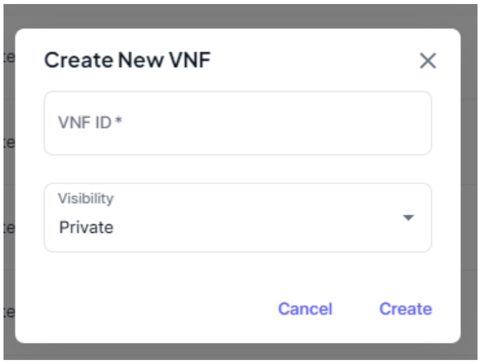
Virtual Network Functions (VNF)

□	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	CREATED AT
□	46	log_test	Missing Descriptor		No	Yes	
□	28	test-fork	K8s Prometheus container for NetApp	1.0.7	No	Yes	01:26 - 18/11/23
□	23	test-refactor	Missing Descriptor		No	Yes	
□	17	test-noti-2	Missing Descriptor		No	Yes	
□	16	test-noti	Missing Descriptor		No	Yes	

Rows per page: 5 1–5 of 17

8.1. Create

To create a new VNF, simply click the green "Create" button. Upon clicking, a dialog box will be displayed, guiding you through the creation process.



Create New VNF

VNF ID *

Visibility
Private

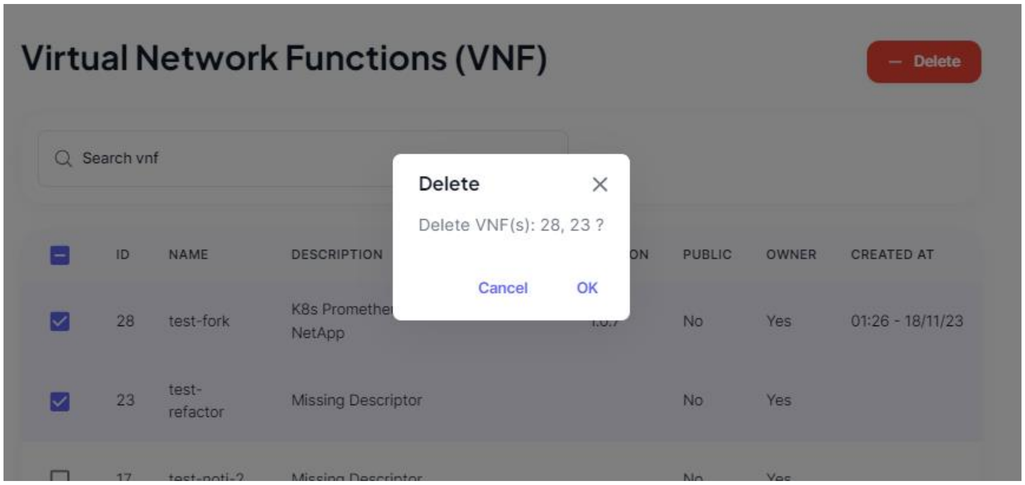
The fields available for a new VNF are:

- VNF ID (mandatory)
- Visibility (mandatory)

Upon the initiation of a new Virtual Network Function (VNF) creation process, the OSR system generates a dedicated entry within its database, specifically earmarked for this VNF. This record serves as the housing unit for the descriptor associated with the VNF, which will be uploaded in subsequent steps. It's important to note that the VNF's unique identifier, the "vnf-id," must align precisely with the available descriptor. This synchronization ensures that the descriptor being uploaded corresponds accurately with the designated VNF record, maintaining consistency and accuracy within the OSR database. The “vnf-id” field must be unique in the created VNFs of each user.

8.2. Delete

To delete a VNF, users are required to select one or more VNFs by checking the corresponding checkboxes in the list. Subsequently, pressing the "Delete" button initiates the deletion process.



Virtual Network Functions (VNF)

Delete VNF(s): 28, 23 ?

☐	ID	NAME	DESCRIPTION	ON	PUBLIC	OWNER	CREATED AT
☑	28	test-fork	K8s Promethe NetApp	1.0.7	No	Yes	01:26 - 18/11/23
☑	23	test-refactor	Missing Descriptor		No	Yes	
☐	17	test-noti-2	Missing Descriptor		No	Yes	

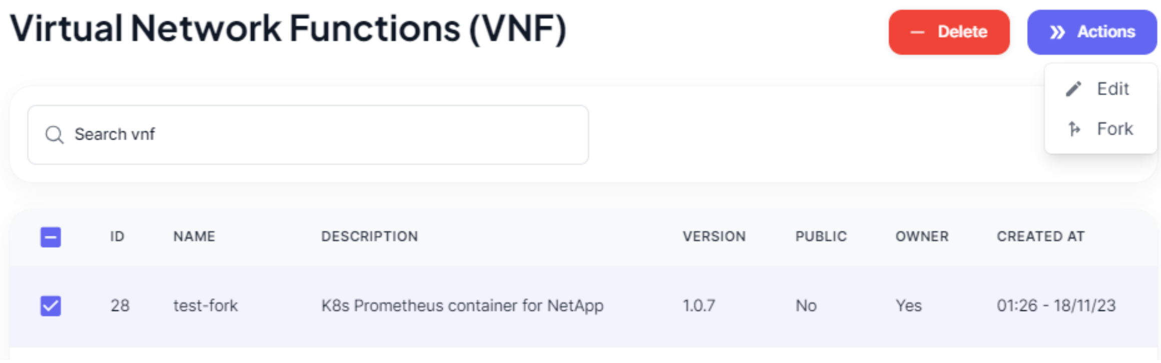
When a Virtual Network Function (VNF) is deleted, the descriptors associated with that VNF are removed from the OSR database. This deletion action can only be initiated by a user who is the owner of the specific VNF.

Furthermore, a crucial restriction is in place: a VNF that is referenced by a Network Service cannot be deleted. This precaution ensures that essential components within the network architecture remain intact and functional. Preventing the deletion of a VNF referenced by a Network Service maintains the integrity and continuity of the network, preventing disruptions that could occur if a vital component were removed while still being actively utilized by a Network Service.

8.3. Actions

When a single VNF is chosen from the checkbox list, an "Actions" button becomes visible, offering a set of available options:

- Edit
- Fork

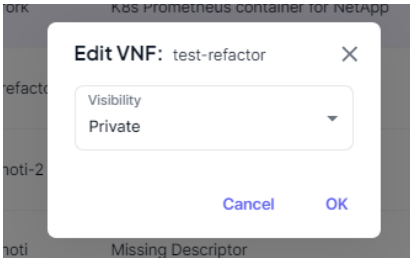


Virtual Network Functions (VNF)

☐	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	CREATED AT
☑	28	test-fork	K8s Prometheus container for NetApp	1.0.7	No	Yes	01:26 - 18/11/23

8.3.1. Edit

Upon selecting the "Edit" action, a popup window will appear, allowing users to modify the visibility fields associated with the chosen VNF.



Edit VNF: test-refactor

Visibility: Private

Only VNFs that are owned by the platform user can be edited.

8.3.2. Fork

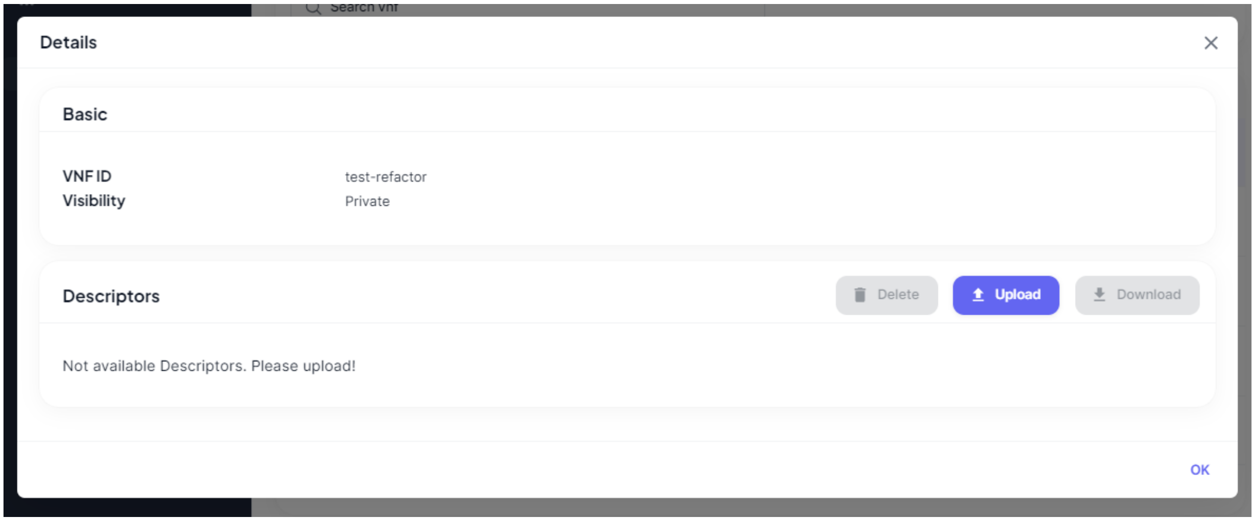
Platform users have the capability to fork other VNFs within the system, enabling them to utilize and subsequently make individual modifications. However, there are specific conditions for forking a VNF:

- The VNF intended for forking must be a public VNF that is not owned by the platform user.
- Once forked, the newly created VNF will inherit the same unique identifier ("vnf-id") as its parent VNF.

This functionality empowers users to create their versions of existing VNFs for personalized adjustments, ensuring that the forked VNF maintains a connection to its original version through the shared identifier.

8.4. Details

Clicking on any of the elements within the Virtual Network Functions (VNF) will prompt a dialog to appear, presenting comprehensive details regarding the selected VNF element.



Basic

VNF ID test-refactor
Visibility Private

Descriptors

Not available Descriptors. Please upload!

In the Basic Card, critical information regarding the VNF is presented, encompassing its ID, visibility status, and the most recent version of the descriptor, if available.

Within the Descriptors Card, the descriptors associated with the VNF are showcased. If descriptors exist, they are displayed directly; otherwise, a prompt encourages users to upload them. Additionally, if multiple versions of the descriptor exist, a dropdown menu allows users to select and view a specific version. Moreover, there are three action buttons provided:

Upload: Enables users to upload a descriptor for the VNF.
Download: Allows users to download the existing descriptor.
Delete: Provides the option to delete the current descriptor associated with the VNF.
There are specific restrictions and considerations regarding the processes for VNF descriptors:

Upload Restrictions:

- Ownership of the descriptor is necessary to upload a VNF descriptor.
- The "vnf-id" of the descriptor must align with the corresponding VNF object's "vnf-id".

Deletion Restrictions:

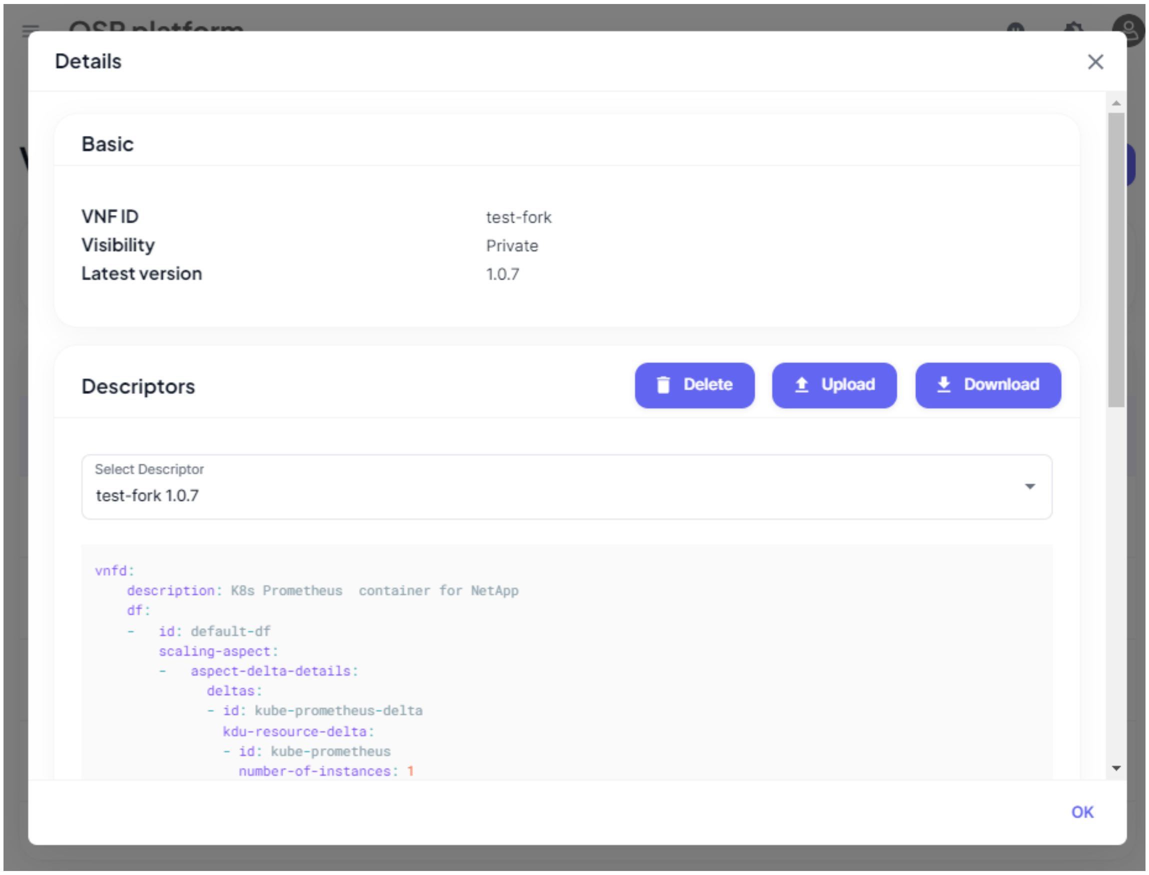
- Deleting a VNF requires ownership of the VNF by the user initiating the action.
- A descriptor cannot be deleted from a VNF that has only one descriptor associated with it.

Propagation of Changes:

- When uploading a new VNF descriptor to a VNF utilized by a NetworkApp, all necessary updates are performed.
- These updates ensure that changes propagate effectively to the parent elements for synchronization.

Download Functionality:

- No specific restrictions apply to downloading a descriptor if it exists for the VNF.



Basic

VNF ID test-fork
Visibility Private
Latest version 1.0.7

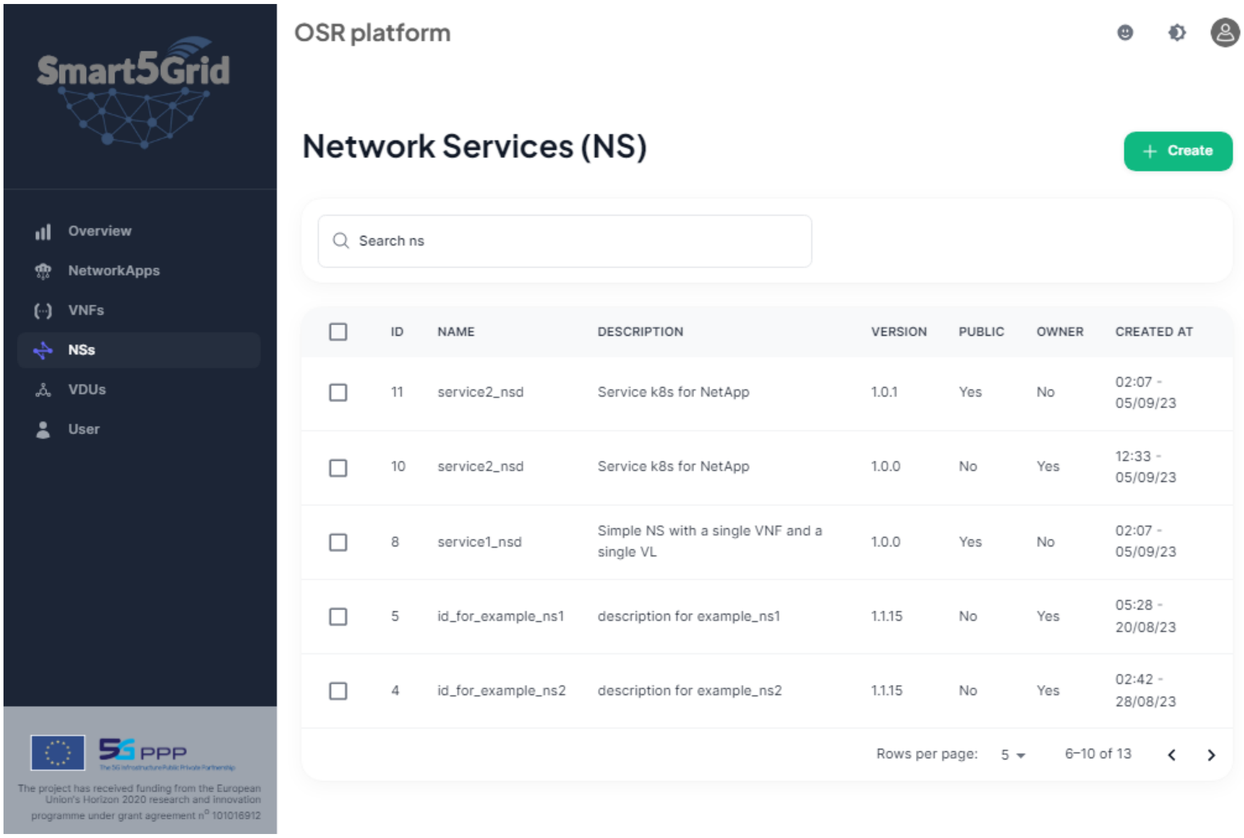
Descriptors

Select Descriptor
test-fork 1.0.7

```
vnfd:
    description: K8s Prometheus  container for NetApp
    df:
    -   id: default-df
        scaling-aspect:
        -   aspect-delta-details:
              deltas:
              - id: kube-prometheus-delta
                kdu-resource-delta:
                - id: kube-prometheus
                  number-of-instances: 1
```

9. Network Services (NSs)

In the Smart5Grid platform, the Network Services (NSs) page offers users access to a complete catalog of all established NSs. Additionally, a Search Field feature facilitates quick and precise retrieval of specific NSs. This search functionality scans across all columns in the dataset, enabling users to input keywords for a thorough and efficient search, ensuring swift identification of the desired NSs. A Network Service (NS) refers to a virtualized and software-based set of functionalities that combine various network resources to deliver specific services or applications. It orchestrates these resources to create a comprehensive service that can span across different network elements, offering functionalities such as routing, security, load balancing, and other network services. NSs play a vital role in modern networking by enabling the creation, management, and deployment of dynamic and customizable network services through software-defined networking (SDN) and network function virtualization (NFV) technologies.



The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 101016912

OSR platform

Network Services (NS)

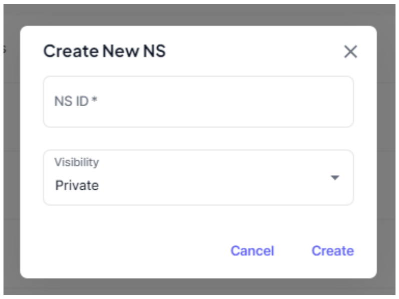
☐	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	CREATED AT
☐	11	service2_nsd	Service k8s for NetApp	1.0.1	Yes	No	02:07 - 05/09/23
☐	10	service2_nsd	Service k8s for NetApp	1.0.0	No	Yes	12:33 - 05/09/23
☐	8	service1_nsd	Simple NS with a single VNF and a single VL	1.0.0	Yes	No	02:07 - 05/09/23
☐	5	id_for_example_ns1	description for example_ns1	1.1.15	No	Yes	05:28 - 20/08/23
☐	4	id_for_example_ns2	description for example_ns2	1.1.15	No	Yes	02:42 - 28/08/23

Rows per page: 5 6–10 of 13

The OSR is equipped to store and handle versioning of descriptors customized for Network Services (NSs). These descriptors act as detailed templates, outlining the configuration, behavior, and specific requirements of each NS. The storage and version control of these descriptors within the OSR are pivotal as they serve as guiding blueprints or reference markers for developing and deploying Network Applications (NetworkApps).

9.1. Create

To initiate the creation of a new NS, click on the green "Create" button. This action will prompt the display of a dialog box that will guide you through the entire creation process.



Create New NS

NS ID *

Visibility: Private

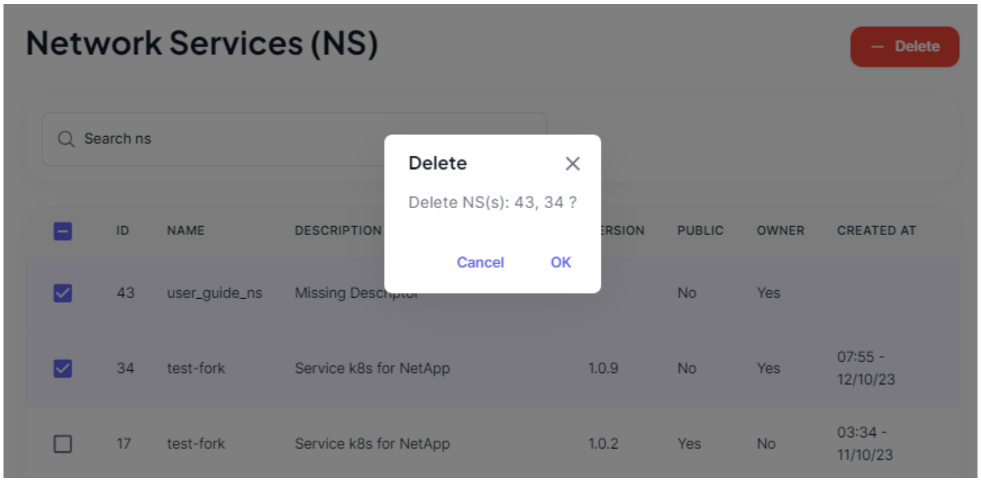
The fields available for a new NS are:

- NS ID (mandatory)
- Visibility (mandatory)

When a new Network Service (NS) creation process is initiated, the OSR system creates a designated entry within its database exclusively for this NS. This record acts as the storage unit for the associated descriptor, to be uploaded in later stages. It's essential to emphasize that the NS's unique identifier, known as the "ns-id," must perfectly match the available descriptor. This alignment guarantees that the uploaded descriptor precisely corresponds to the intended NS record, ensuring consistency and accuracy in the OSR database. Additionally, the "ns-id" field needs to be unique among the NSs created by each user.

9.2. Delete

To delete a Network Service (NS), users are required to select one or more NSs by checking the corresponding checkboxes in the list. Subsequently, pressing the "Delete" button initiates the deletion process.



Network Services (NS)

Delete

Delete NS(s): 43, 34 ?

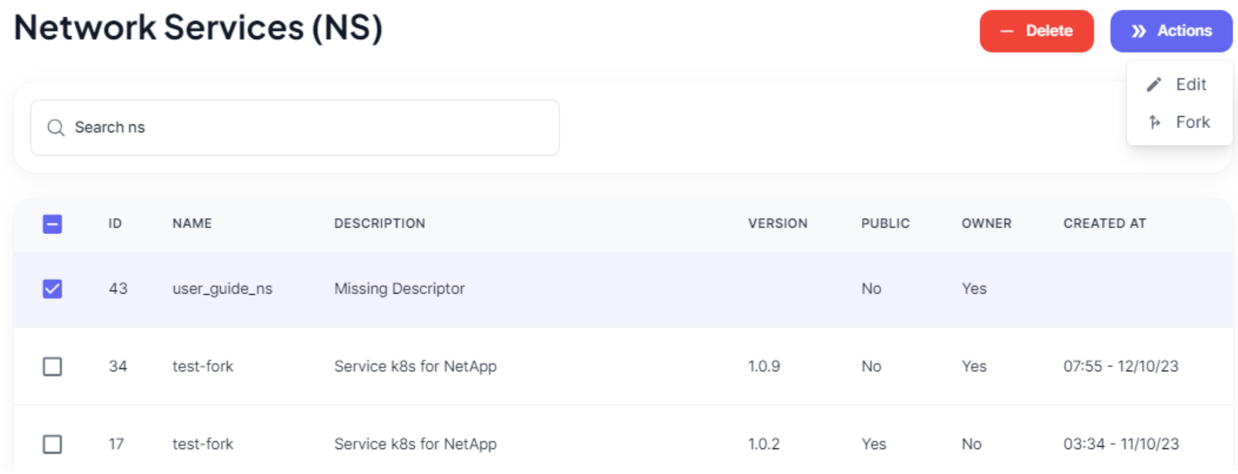
⊟	ID	NAME	DESCRIPTION	ERSION	PUBLIC	OWNER	CREATED AT
☑	43	user_guide_ns	Missing Descr		No	Yes	
☑	34	test-fork	Service k8s for NetApp	1.0.9	No	Yes	07:55 - 12/10/23
□	17	test-fork	Service k8s for NetApp	1.0.2	Yes	No	03:34 - 11/10/23

Upon deletion of a Network Service (NS), the descriptors linked to that NS are removed from the OSR database. This deletion process can only be triggered by the NS's owner. Additionally, an important restriction applies: an NS referenced by a NetworkApp cannot be deleted. This measure safeguards critical network components, ensuring their stability and functionality. Preventing the deletion of an NS referenced by a NetworkApp preserves the network's integrity and continuous operation, averting disruptions that might arise from the removal of a vital component actively utilized by a NetworkApp.

9.3. Actions

When a single NS is chosen from the checkbox list, an "Actions" button becomes visible, offering a set of available options:

- Edit
- Fork

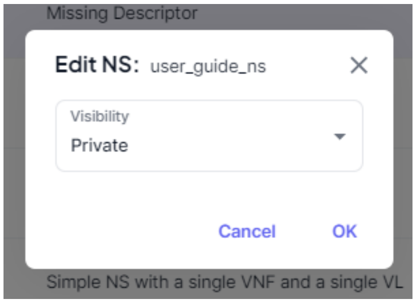


Network Services (NS)

⊟	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	CREATED AT
☑	43	user_guide_ns	Missing Descriptor		No	Yes	
☐	34	test-fork	Service k8s for NetApp	1.0.9	No	Yes	07:55 - 12/10/23
☐	17	test-fork	Service k8s for NetApp	1.0.2	Yes	No	03:34 - 11/10/23

9.3.1. Edit

Upon selecting the "Edit" action, a popup window will appear, allowing users to modify the visibility fields associated with the chosen NS.



Missing Descriptor

Edit NS: user_guide_ns

Visibility: Private

Simple NS with a single VNF and a single VL

Only Network services that owned by the platform user can be edited.

9.3.2. Fork

Platform users have the capability to fork other Network Services (NSs) within the system, granting them the ability to use and subsequently make individual modifications. However, specific conditions apply for forking an NS:

- The NS intended for forking must be a public NS that is not owned by the platform user.
- Once forked, the newly created NS will inherit the same unique identifier ("ns-id") as its parent NS.
- If an NS has dependencies to VNFs, these VNFs will be also forked

This functionality empowers users to create personalized versions of existing NSs, ensuring that the forked NS maintains a connection to its original version through the shared identifier.

9.4. Details

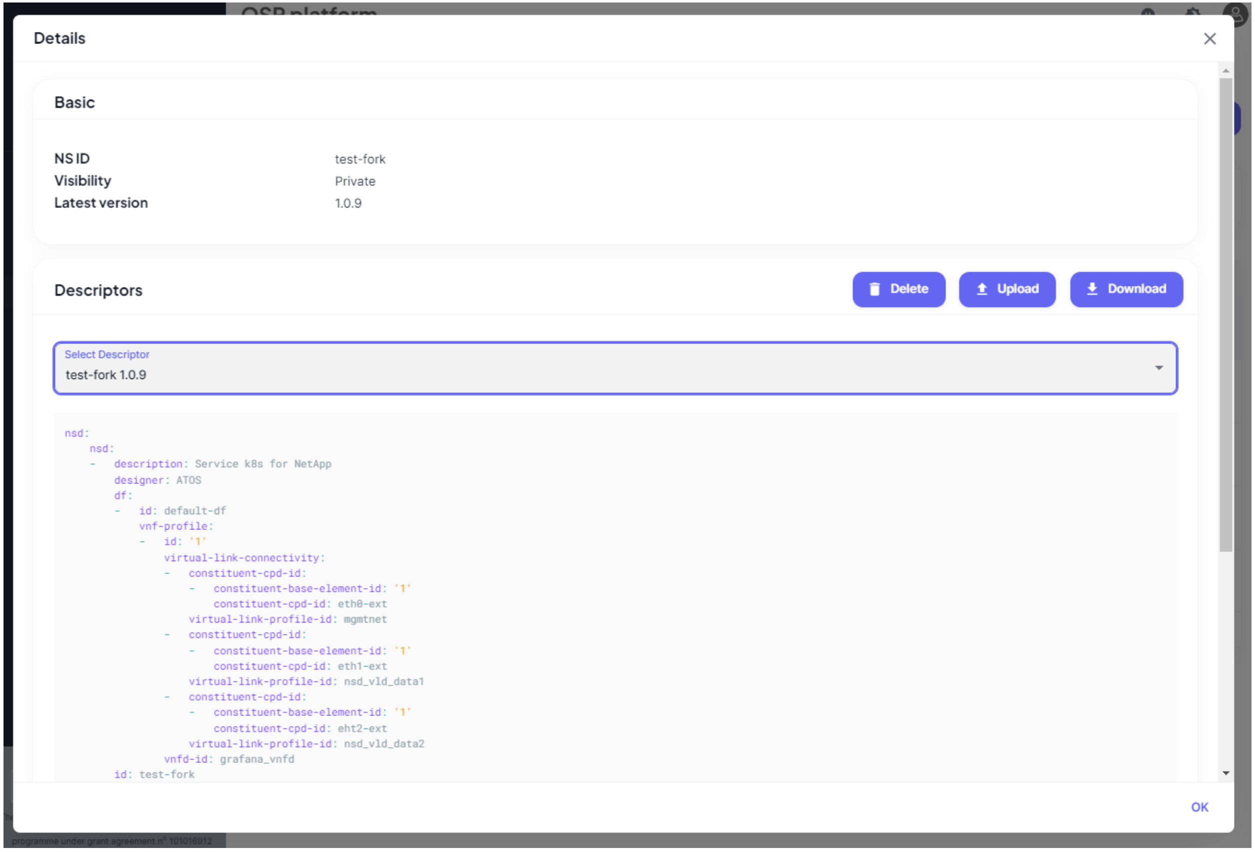
Clicking on any of the elements within a Network Service (NS) array will prompt a dialog to appear, presenting comprehensive details regarding the selected NS element.

In the Basic Card, critical information regarding the NS is presented, encompassing its ID, visibility status, and the most recent version of the descriptor, if available.

Within the Descriptors Card, the descriptors associated with the NS are showcased. If descriptors exist, they are displayed directly; otherwise, a prompt encourages users to upload them. Additionally, if multiple versions of the descriptor exist, a dropdown menu allows users to select and view a specific version.

Moreover, there are three action buttons provided:

Upload: Enables users to upload a descriptor for the NS.
Download: Allows users to download the existing descriptor.
Delete: Provides the option to delete the current descriptor associated with the NS.



Basic

NS ID test-fork
Visibility Private
Latest version 1.0.9

Descriptors

Select Descriptor
test-fork 1.0.9

```
nsd:
   nsd:
   -   description: Service k8s for NetApp
       designer: ATOS
       df:
       -   id: default-df
           vnf-profile:
           -   id: '1'
               virtual-link-connectivity:
               -   constituent-cpd-id:
                   -   constituent-base-element-id: '1'
                       constituent-cpd-id: eth0-ext
                   virtual-link-profile-id: mgmtnet
               -   constituent-cpd-id:
                   -   constituent-base-element-id: '1'
                       constituent-cpd-id: eth1-ext
                   virtual-link-profile-id: nsd_vld_data1
               -   constituent-cpd-id:
                   -   constituent-base-element-id: '1'
                       constituent-cpd-id: eht2-ext
                   virtual-link-profile-id: nsd_vld_data2
               vnfd-id: grafana_vnfd
       id: test-fork
```

There are specific restrictions and considerations regarding the processes for NS descriptors:

Upload Restrictions:

- Ownership of the descriptor is necessary to upload a NS descriptor.
- The "ns-id" of the descriptor must align with the corresponding NS object's "nsf-id".

Deletion Restrictions:

- Deleting a NS requires ownership of the NS by the user initiating the action.
- A descriptor cannot be deleted from a NS that has only one descriptor associated with it.

Propagation of Changes:

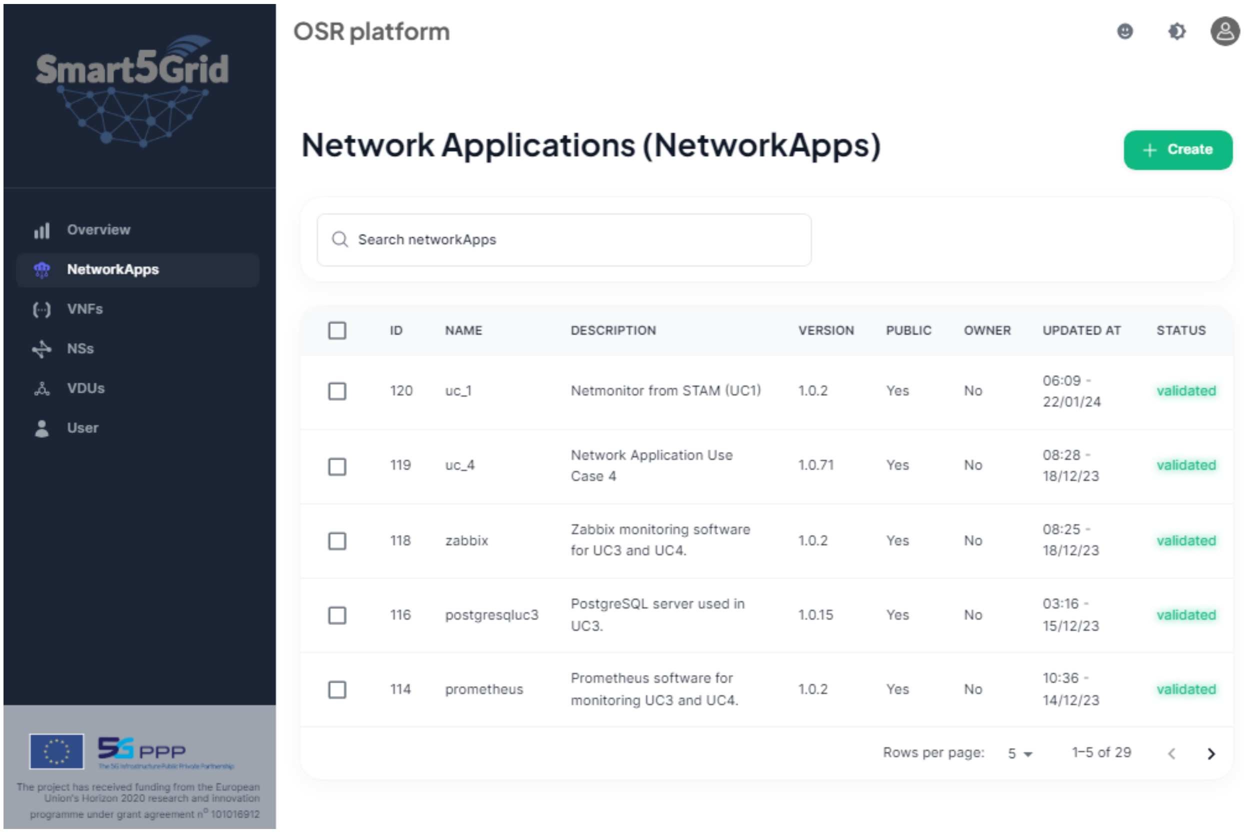
- When uploading a new NS descriptor to a NS utilized by a NetworkApp, all necessary updates are performed.
- These updates ensure that changes propagate effectively to the parent elements for synchronization.

Download Functionality:

- No specific restrictions apply to downloading a descriptor if it exists for the NS.

10. Network Applications (NetworkApps)

In the Smart5Grid platform, the NetworkApps page provides users with access to a comprehensive catalog of all established NetworkApps. Furthermore, a Search Field feature streamlines the quick and accurate retrieval of specific NetworkApps. This search functionality conducts scans across all dataset columns, allowing users to input keywords for a comprehensive and efficient search, ensuring rapid identification of the desired NetworkApps. A NetworkApp, short for Network Application, refers to a software-based application or service that utilizes various network resources to deliver specific functionalities or services within a network infrastructure. NetworkApps are designed to provide tailored solutions for networking needs, combining diverse network elements, functionalities, and services to create comprehensive and customized applications. They play a crucial role in modern networking environments by enabling the creation, deployment, and management of dynamic, adaptable, and scalable network services, contributing to the efficient functioning and performance of network architectures.



The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 101016912

Network Applications (NetworkApps)

□	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	UPDATED AT	STATUS
□	120	uc_1	Netmonitor from STAM (UC1)	1.0.2	Yes	No	06:09 - 22/01/24	validated
□	119	uc_4	Network Application Use Case 4	1.0.71	Yes	No	08:28 - 18/12/23	validated
□	118	zabbix	Zabbix monitoring software for UC3 and UC4.	1.0.2	Yes	No	08:25 - 18/12/23	validated
□	116	postgresqluc3	PostgreSQL server used in UC3.	1.0.15	Yes	No	03:16 - 15/12/23	validated
□	114	prometheus	Prometheus software for monitoring UC3 and UC4.	1.0.2	Yes	No	10:36 - 14/12/23	validated

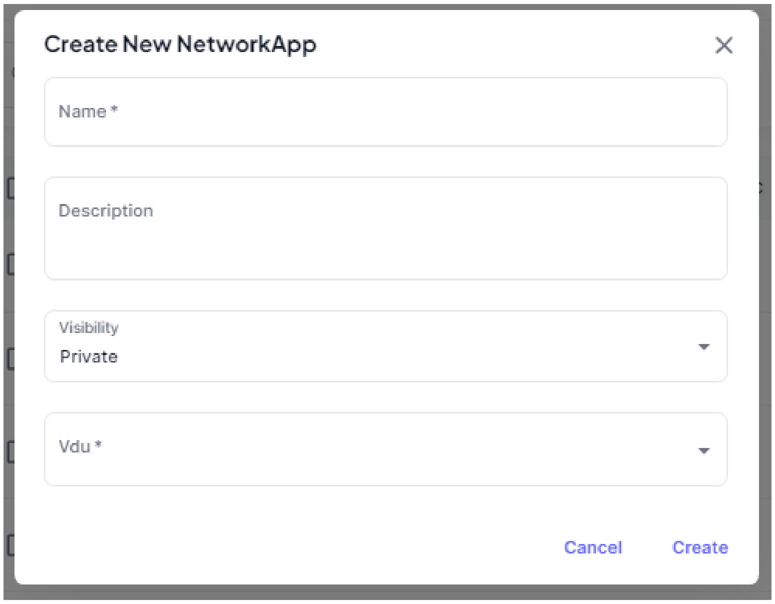
The OSR is designed to store and manage versioning of descriptors tailored for NetworkApps. These descriptors serve as comprehensive templates, detailing the configuration, behavior, and specific requirements of each individual NetworkApp.

10.1. Create

To create a new NetworkApp, simply click the green "Create" button. Upon clicking, a dialog box will be displayed, guiding you through the creation process.

The fields available for a new NetworkApp are:

- Name (mandatory)
- Description (optional)
- Visibility (mandatory)
- VDU (mandatory)



Create New NetworkApp ✕

Name *

Description

Visibility
Private

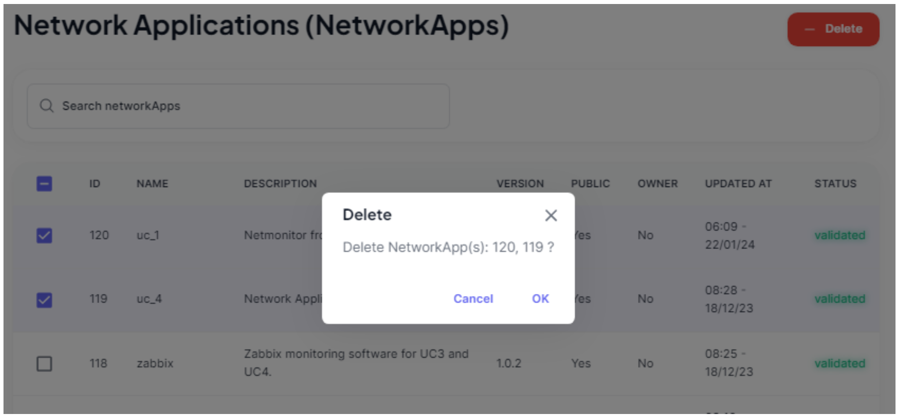
Vdu *

Cancel Create

When a new Network Application (NetworkApp) creation process is initiated, the OSR system creates a designated entry within its database exclusively for this NetworkApp. This record acts as the storage unit for the associated descriptor, to be uploaded in later stages. To create a NetworkApp, it is necessary to select a Virtual Deployment Unit (VDU) owned by the user initiating the NetworkApp creation process. This implies that, before creating a NetworkApp, a new VDU owned by the user needs to be created.

10.2. Delete

To delete a Network Application, users are required to select one or more NetworkApp by checking the corresponding checkboxes in the list. Subsequently, pressing the "Delete" button initiates the deletion process.



Network Applications (NetworkApps) — Delete

Search networkApps

▣	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	UPDATED AT	STATUS
☑	120	uc_1	Netmonitor fr		Yes	No	06:09 - 22/01/24	validated
☑	119	uc_4	Network Appl		Yes	No	08:28 - 18/12/23	validated
□	118	zabbix	Zabbix monitoring software for UC3 and UC4.	1.0.2	Yes	No	08:25 - 18/12/23	validated

Delete ✕

Delete NetworkApp(s): 120, 119 ?

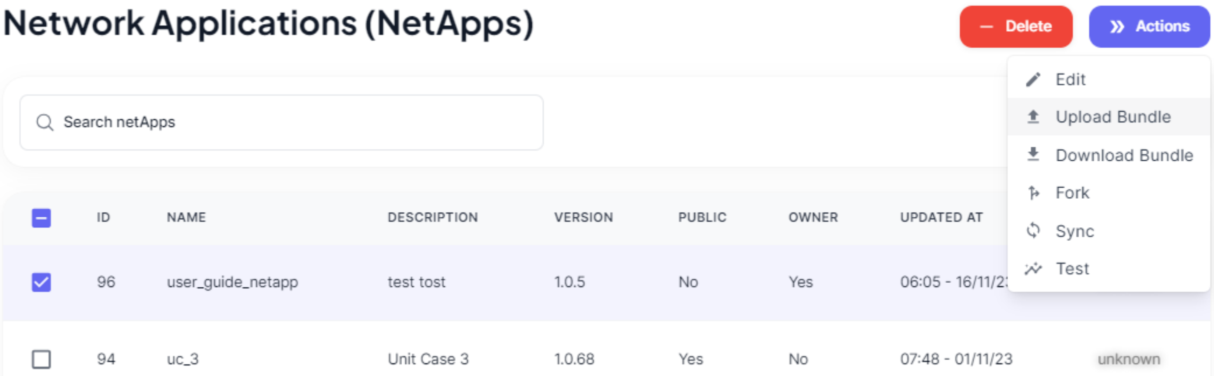
Cancel OK

Upon deletion of a NetworkApp, the descriptors linked to that NetworkApp are removed from the OSR database. This deletion process can only be triggered by the NetworkApp's owner. Also, all the information stored at the Gitlab repo of the VDU regarding this NetworkApp is removed.

10.3. Actions

When a single NetworkApp is chosen from the checkbox list, an "Actions" button becomes visible, offering a set of available options:

- Edit
- Upload Bundle
- Download Bundle
- Fork
- Sync
- Test

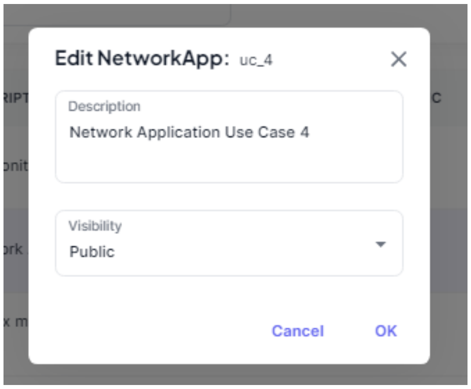


Network Applications (NetApps)

☐	ID	NAME	DESCRIPTION	VERSION	PUBLIC	OWNER	UPDATED AT	
☑	96	user_guide_netapp	test tost	1.0.5	No	Yes	06:05 - 16/11/2	
☐	94	uc_3	Unit Case 3	1.0.68	Yes	No	07:48 - 01/11/23	unknown

10.3.1. Edit

Upon selecting the "Edit" action, a popup window will appear, allowing users to modify the visibility and description fields associated with the chosen Networkapp.



Edit NetworkApp: uc_4

Description: Network Application Use Case 4

Visibility: Public

10.3.2. Upload Bundle

The OSR, and consequently, the Smart5Grid platform, provides users with the capability to upload a NetworkApp as a bundled package. This package is in tar.gz format and encompasses all the necessary descriptors and artifacts, arranged within a predefined file structure. The platform supports two types of NetworkApps: OSM and HELM formats. The upload bundle functionality streamlines the entire process of

VNF, NS, and NetworkApp creation automatically. Additionally, it handles descriptor binding by utilizing the naming conventions of the respective descriptors, enabling a seamless integration of the NetworkApp into the system. All images and charts included in the bundle are automatically stored in the Harbor project associated with the Virtual Deployment Unit (VDU). Furthermore, a commit containing all the descriptors is executed within the GitLab repository corresponding to the respective VDU.

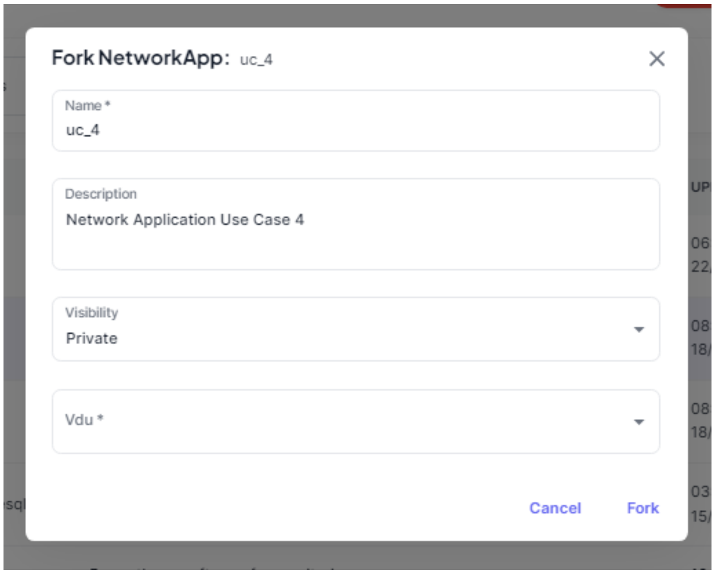
10.3.3. Download Bundle

The OSR platform enables users to download a NetworkApp along with its associated resources, encompassing Network Services (NS), VNs (Virtual Networks), and descriptors via the Download Bundle option. During this download process, a live packaging procedure is initiated to gather all artifacts from the defined Harbor associated with the NetworkApp's Virtual Deployment Unit (VDU). These artifacts are then saved in a tar.gz archive, comprising all descriptors bound to the NetworkApp.

The Download Bundle functionality retrieves the latest version of each resource, encompassing descriptors and artifacts. Specifically for artifacts, all saved items within the Harbor project are bundled together, disregarding the content of the descriptors. Developers should be careful to upload all the needed images and charts to the VDU before any package download.

10.3.4. Fork

Similar to the process for VNFs and NSs, platform users can also initiate a NetworkApp fork. However, the mechanism for forking a NetworkApp is a bit more sophisticated than that of the others. When the option is selected, a popup window is displayed:



Fork NetworkApp: uc_4

Name *
uc_4

Description
Network Application Use Case 4

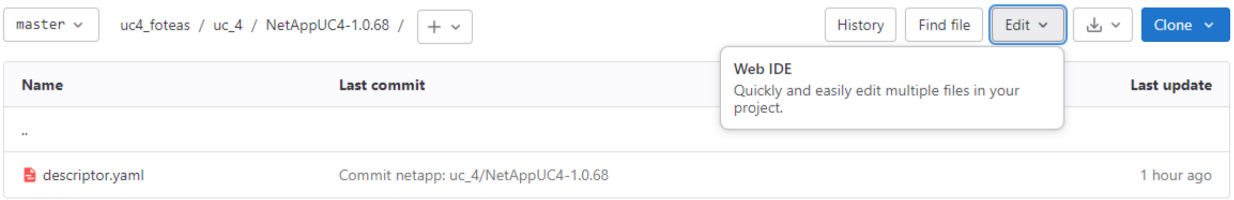
Visibility
Private

Vdu *

The user is required to specify the name, description, visibility, and the Virtual Deployment Unit (VDU) to be utilized by the newly forked NetworkApp. For better identification, it's advisable to use the same name as the parent NetworkApp. This ensures coherence and facilitates clearer identification between the parent and the forked NetworkApp. When a NetworkApp is forked all the resources needed are also forked for the new NetworkApp. This includes all the Network Services (NSs) the Virtual Network Functions (VNFs) and all the artifacts are copied to the new selected VDU instance. In cases that there are multiple versions in any of the resources the latest version is selected for the fork action.

10.3.5. Sync

Sync functionality is a beta functionality that is supported by OSR and platform. Users can update or add descriptors to the respective Gitlab repository that is created with the NetworkApp, and after performing the commit to the Gitlab, all these changes will be also applied to the OSR data model.

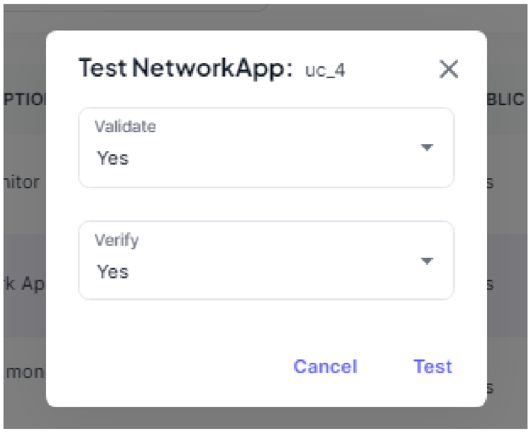

master
uc4_foteas / uc_4 / NetAppUC4-1.0.68 /
+
History
Find file
Edit
Clone
Web IDE
Quickly and easily edit multiple files in your project.

Name	Last commit	Last update
..		
descriptor.yaml	Commit netapp: uc_4/NetAppUC4-1.0.68	1 hour ago

Update can be also achieved directly from the Gitlab UI without the need to clone the code locally to your computer.

10.3.6. Test

The last but very important action that is available for NetworkApps is the test. Via the test action V&V is triggered to perform a series of tests to the NetworkApp.


Test NetworkApp: uc_4

Validate: Yes

Verify: Yes

The available options are Validation and Verification. NetworkApp validation and verification are two distinct processes in the lifecycle management of Network Applications (NetworkApps):

NetworkApp Verification:

- Explanation: NetworkApp verification involves the process of confirming whether the implemented NetworkApp aligns with the original specifications and design requirements.
- Objective: The key objective of NetworkApp verification is to validate that the implemented NetworkApp corresponds accurately with the initial design, ensuring that all features and functionalities are correctly implemented.
- Methods: Verification includes a series of checks, reviews, and assessments to validate that the NetworkApp's features and behaviors match the intended design and requirements documented during the planning phase.

NetworkApp Validation:

- Explanation: NetworkApp validation refers to the process of ensuring that the designed or implemented NetworkApp complies with the specified requirements, standards, and expectations.

- **Objective:** The primary goal of NetworkApp validation is to confirm that the NetworkApp functions as intended, meeting the defined criteria and delivering the expected performance.
- **Methods:** Validation involves various testing procedures, simulations, and checks to confirm that the NetworkApp meets the specified standards and fulfills its intended purpose without errors or deviations.

Regarding the testing a NetworkApp can have the following statuses.

- Uknown: No tests have been triggered.
- Pending: Tests triggered and waiting results from the V&V engine
- Verified: Verification step passed but validation step failed
- Validated: Both validation and verification steps are completed successfully
- Error: Neither verification nor validation were successful

More information regarding the erroneous test runs are displayed at the details dialog and are described at next chapter.

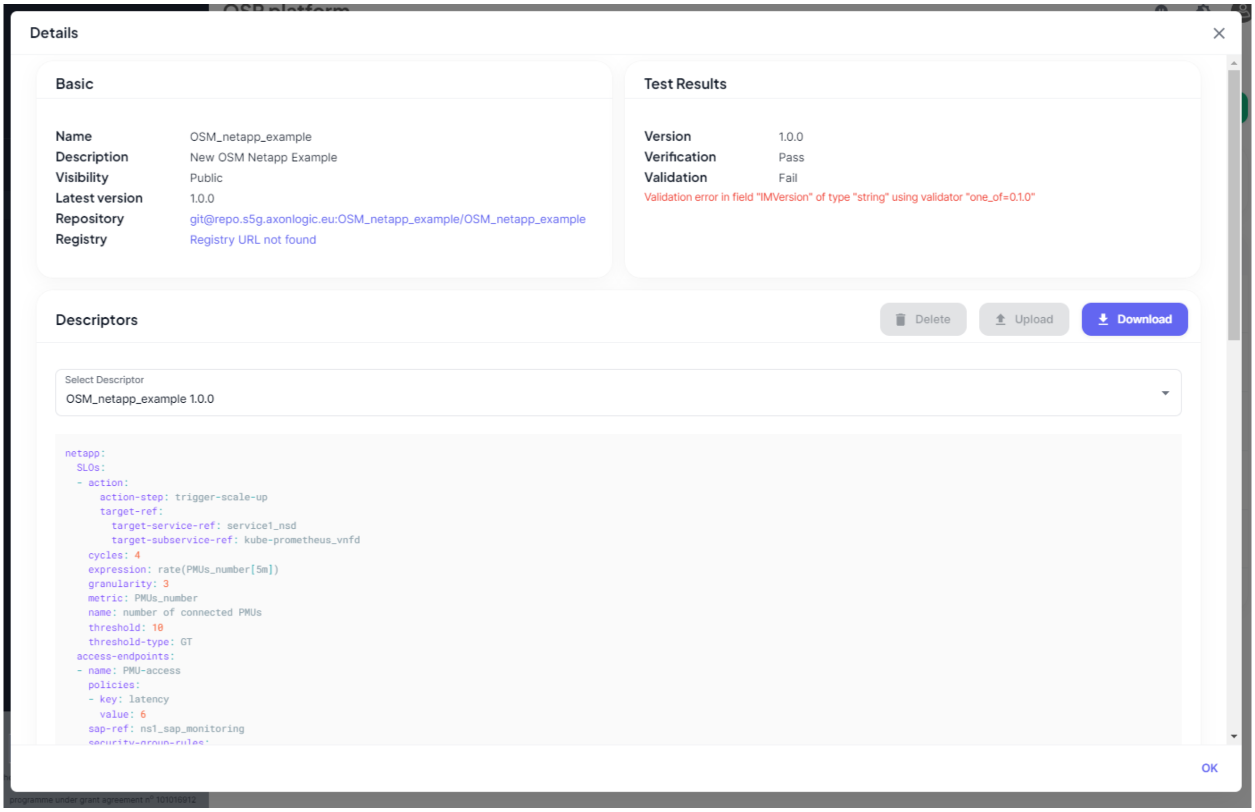
10.4. Details

Clicking on any item within a Network Application (NetworkApp) array triggers a detailed dialog showcasing comprehensive information about the selected NetworkApp element.

Within the Basic Card, crucial details about the NetworkApp are highlighted, including its Name, Description, visibility status, and the latest available version of the descriptor. Additionally, direct links to the repository and registry, specifically associated with the linked VDU element, are provided.

In the Test Results Card, the version of the tested NetworkApp is displayed alongside the results of the Verification and Validation processes. In cases where no tests have been conducted, both Verification and Validation will display as "none". Should a test fail, an explanation for the failure is presented in red letters below the corresponding test.

The Descriptors section offers a dropdown menu for selecting among uploaded descriptors. Changing the descriptor triggers an update to align the test results with the newly selected descriptor.



Basic

Name	OSM_netapp_example
Description	New OSM Netapp Example
Visibility	Public
Latest version	1.0.0
Repository	git@repo.s5g.axonlogic.eu:OSM_netapp_example/OSM_netapp_example
Registry	Registry URL not found

Test Results

Version	1.0.0
Verification	Pass
Validation	Fail

Validation error in field "IMVersion" of type "string" using validator "one_of=0.1.0"

Descriptors

Select Descriptor
OSM_netapp_example 1.0.0

```
netapp:
  SLOs:
  - action:
      action-step: trigger-scale-up
      target-ref:
        target-service-ref: service1_nsd
        target-subservice-ref: kube-prometheus_vnfd
    cycles: 4
    expression: rate(PMUs_number[5m])
    granularity: 3
    metric: PMUs_number
    name: number of connected PMUs
    threshold: 10
    threshold-type: GT
  access-endpoints:
  - name: PMU-access
    policies:
    - key: latency
      value: 6
    sap-ref: ns1_sap_monitoring
    security-group-rules:
```

Moreover, there are three action buttons provided:

Upload: Enables users to upload a descriptor for the NetworkApp.
Download: Allows users to download the existing descriptor.
Delete: Provides the option to delete the current descriptor associated with the NetworkApp.

There are specific restrictions and considerations regarding the processes for NetworkApp descriptors:

Upload Restrictions:

- Ownership of the descriptor is necessary to upload a NetworkApp descriptor.

Deletion Restrictions:

- Deleting a NetworkApp requires ownership of the NetworkApp by the user initiating the action.
- A descriptor cannot be deleted from a NetworkApp that has only one descriptor associated with it.

Updating Gitlab:

- When uploading a new NetworkApp, the updated version of the NetworkApp is also committed to the gitlab

Download Functionality:

- No specific restrictions apply to downloading a descriptor if it exists for the NetworkApp.

11. Examples

In this chapter, we'll provide a concise overview of the essential steps required to execute basic actions with a NetworkApp. The first example pertains to the upload of a NetworkApp Bundle, while the second involves the individual upload of each element. Both examples have a prerequisite step the platform registration that is described at chapter 4.2.

11.1. Upload and test a NetworkApp Bundle

The steps required to upload and test a NetworkApp Bundle are as follows:

- Create a VDU. Details are described at chapter 7.1. This VDU will be associated with the NetworkApp to store all artifacts and provide the GitLab project for version control purposes.
- Create a NetworkApp. Details are described at chapter 10.1. At the creation step select the VDU that you created at the previous step.
- Select the checkbox of the newly created NetworkApp and press the Actions Button that is appeared. Then select the Upload Bundle option. Details at chapter 10.3.2
- Depending on the size of the package that is uploaded you may need to wait up to some minutes.
- Verify that the NetworkApp descriptor is Successfully uploaded.
- If the NetworkApp is of type OSM check that also the respective VNF and NS have been created.
- Check at the VDU details that all the artifacts are successfully uploaded at the VDU.
- Select the checkbox of the newly created NetworkApp and press the Actions Button that is appeared. Then select the Test option. Details at chapter 10.3.6
- Now the NetworkApp must be at pending state. Wait a couple of minutes for the V&V to run the tests and inform the OSR for the test results.
- You may need to refresh the page to see the test results. More details regarding the test results can be found at the details section of the NetworkApp. See chapter 10.4

11.2. Upload and test a NetworkApp Individually

The steps required to upload and test a NetworkApp by uploading each element individually are as follows:

- Create a VDU. Details are described at chapter 7.1. This VDU will be associated with the NetworkApp to store all artifacts and provide the GitLab project for version control purposes.
- If the NetworkApp is of type OSM create a VNF object for each VNF descriptor. See details at chapter 8.1. Upload the descriptor for each of the VNF that you just created. See details at chapter 8.4.
- If the NetworkApp is of type OSM create a NS object for each NS descriptor. See details at chapter 9.1. Upload the descriptor for each of the VNF that you just created. See details at chapter 9.4.
- Create a NetworkApp. Details are described at chapter 10.1. At the creation step select the VDU that you created at the first step.
- Upload the descriptor of the NetworkApp, by selection on the details of new NetworkApp. See details chapter 10.4
- For each of the artifact that needed for the just uploaded descriptor, follow the instructions of chapter 7.3.2 to upload the images and the instructions of chapter 7.3.3 to upload the charts.
- Select the checkbox of the newly created NetworkApp and press the Actions Button that appears. Then select the Test option. Details at chapter 10.3.6
- Now the NetworkApp must be at pending state. Wait a couple of minutes for the V&V to run the tests and inform the OSR for the test results.

- You may need to refresh the page to see the test results. More details regarding the test results can be found at the details section of the NetworkApp. See chapter 10.4