



A Holistic, Innovative Framework for the Design,
Development and Orchestration of 5G-ready
Applications and Network Services over Sliced
Programmable Infrastructure

MATILDA PLATFORM USER GUIDE

Co-funded by
the Horizon 2020
Framework Programme
of the European Union



Call:

H2020-ICT-2016-2

Type of Action:

IA

Project Acronym:

MATILDA

Project ID:

761898

Duration:

38 months

Start Date:

01/06/2017

Project Coordinator:

Name:

Franco Davoli

Phone:

+39 010 353 2732

Fax:

+39 010 353 2154

e-mail:

franco.davoli@cnit.it

Technical Coordinator:

Name:

Panagiotis Gouvas

Phone:

+30 216 5000 503

Fax:

+30 216 5000 599

e-mail:

pgouvas@ubitech.eu

Disclaimer

The information, documentation and figures available in this deliverable are written by the MATILDA Consortium partners under EC co-financing (project H2020-ICT-761898) and do not necessarily reflect the view of the European Commission.

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The reader uses the information at his/her sole risk and liability.

Copyright

Copyright © 2020 the MATILDA Consortium. All rights reserved.

The MATILDA Consortium consists of:

CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI (CNIT)

ATOS SPAIN SA (ATOS)

ERICSSON TELECOMUNICAZIONI (ERICSSON)

INTRASOFT INTERNATIONAL SA (INTRA)

COSMOTE KINITES TILEPIKOINONIES AE (COSM)

ORANGE ROMANIA SA (ORO)

EXXPERTSYSTEMS GMBH (EXXPERT)

*GIOUMPI TEK MELETI SCHEDIASMO S YLOPOIISI KAI POLISI ERGON PLIROFORIKIS
ETAI REIA PERIORISMENIS EFTHYNIS (UBITECH)*

INTERNET INSTITUTE, COMMUNICATIONS SOLUTIONS AND CONSULTING LTD (ININ)

INCELLIGENT IDIOTIKI KEFALAIOUCHIKI ETAIREIA (INC)

NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS” (NCSR)

UNIVERSITY OF BRISTOL (UNIVBRIS)

AALTO-KORKEAKOULUS AATIO (AALTO)

UNIVERSITY OF PIRAEUS RESEARCH CENTER (UPRC)

ITALTEL SPA (ITL)

BIBA - BREMER INSTITUT FÜR PRODUKTION UND LOGISTIK GMBH (BIBA)

SUITE5 DATA INTELLIGENCE SOLUTIONS LIMITED (S5)

This document may not be copied, reproduced or modified in whole or in part for any purpose without written permission from the MATILDA Consortium. In addition to such written permission to copy, reproduce or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

Table of Acronyms

Acronym	Definition
AWS	Amazon Web Services
BPF	Berkeley Packet Filter
CRUD	Create, Read, Update, and Delete
DDoS	Distributed Denial of Service
GUI	Graphical User Interface
OSS	Operations Support Systems
QI	Quality Indicator
TLP	Telecom Layer Platform
VAO	Vertical Application Orchestrator
VIM	Virtual Infrastructure Manager
XDP	eXpress Data Path
ZSM	Zero Touch Service Management

User Guide

This document provides detailed guidance in the MATILDA platform, covering the lifecycle management of 5G-ready applications. Detailed steps are provided for the onboarding of 5G-ready applications in the MATILDA platform, their deployment and management over 5G infrastructure.

The MATILDA platform (see Figure 1) can be accessed at the following link:

<https://matilda.euprojects.net>

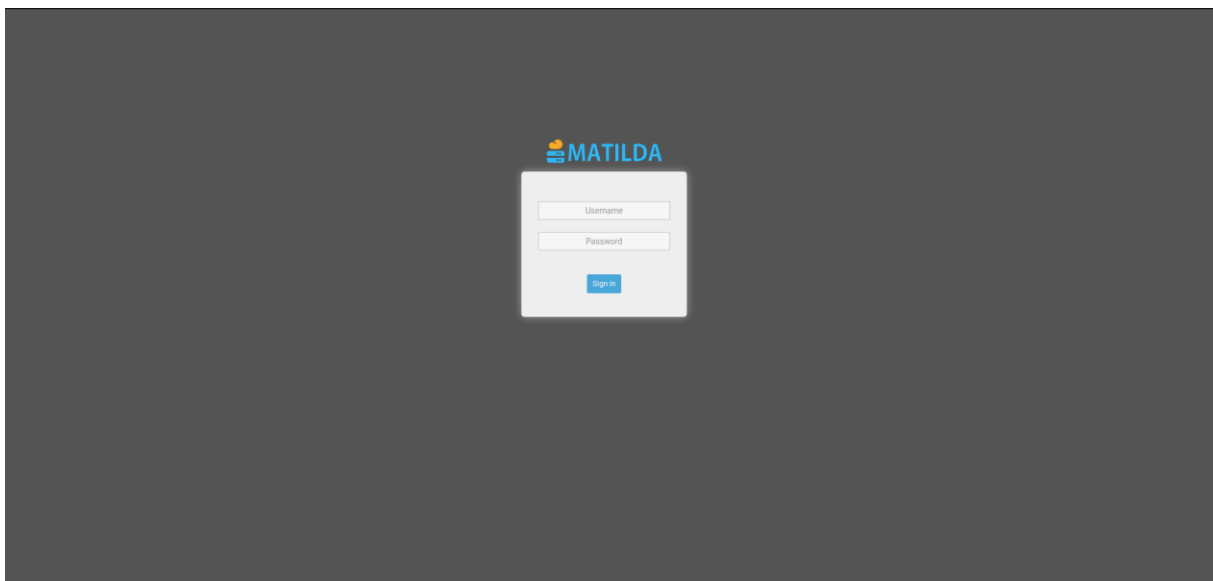


Figure 1: Login Screen in the MATILDA platform

The user is able to log in by using the provided credentials. Following, MATILDA welcomes the user with a sophisticated dashboard that provides a centralized overview of the current status of the platform (Figure 2). The dashboard is comprised of six panels, namely:

1. the top panel provides information about the currently deployed instances, the registered applications and components, as well as the physical resources being utilized.
2. a panel that outlines the resources being used in each provider against the total available resources.
3. a unified log panel consolidating the various log outputs of the instances.
4. two panels showing the active elasticity and security policies being used.
5. a panel showing charts for vCPU and vRAM utilization.

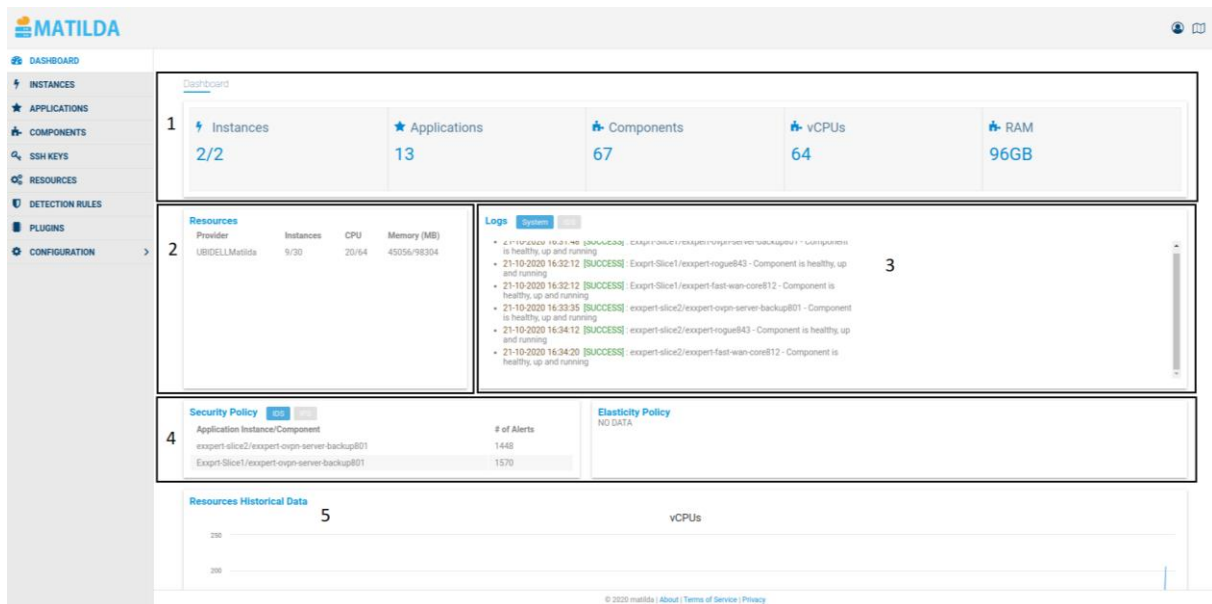


Figure 2: MATILDA Dashboard main screen

Additionally, on the left-hand side, a menu is available allowing the user to navigate to the various sections of the MATILDA platform. This allows the user to create and modify the various resources that are used by the MATILDA platform for deployments.

The available sub-menus are the following:

- DASHBOARD
- INSTANCES
- APPLICATIONS
- COMPONENTS
- SSH KEYS
- RESOURCES
- DETECTION RULES
- PLUGINS
- CONFIGURATION

The steps that are described below are highlighting the concept that an application developer or an operator needs to register and deploy 5G-ready applications through MATILDA.

First the user needs to declare the basic configuration to be used by MATILDA, specifically:

- QIs: the quality indicators that are related to 3GPP specified Key Point Indicators (3GPP TS 32.450,32.451) (Figure 3).
- Radio Service Types: values that are used for specifying the type of connections needed by the user and how to be handled by the Slice Manager (Figure 4).
- User Management options for managing different users (user CRUD actions) (Figure 5).
- Organization Management options for managing different organizations (organizations CRUD actions) (Figure 6).

The specific interfaces for each of these configurations and their various properties can be found in the following figures.

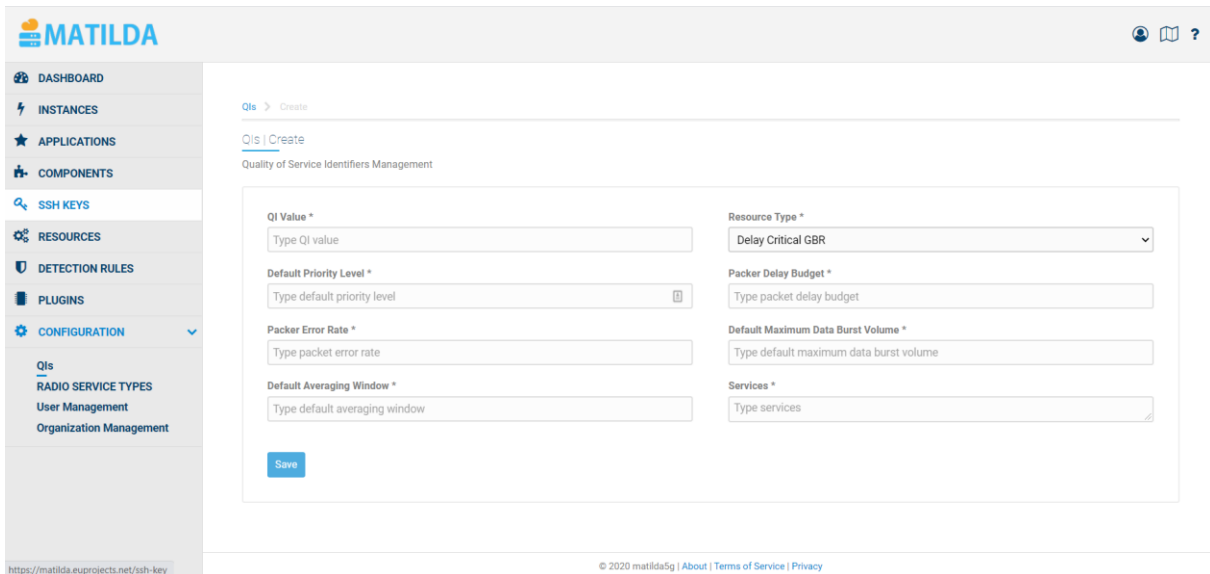


Figure 3: QIs specification

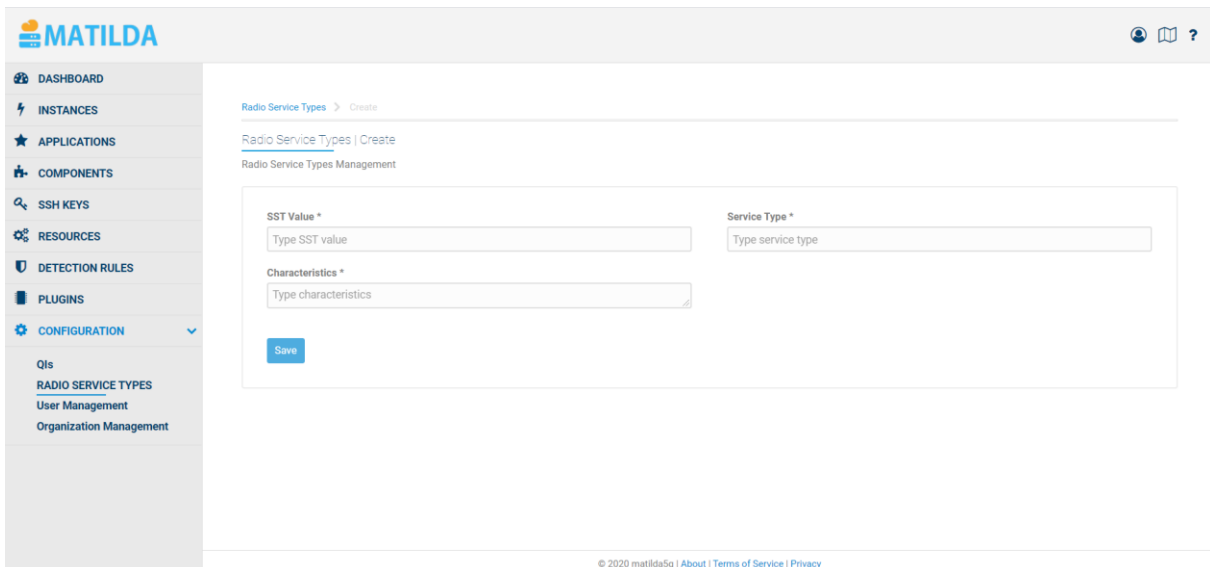
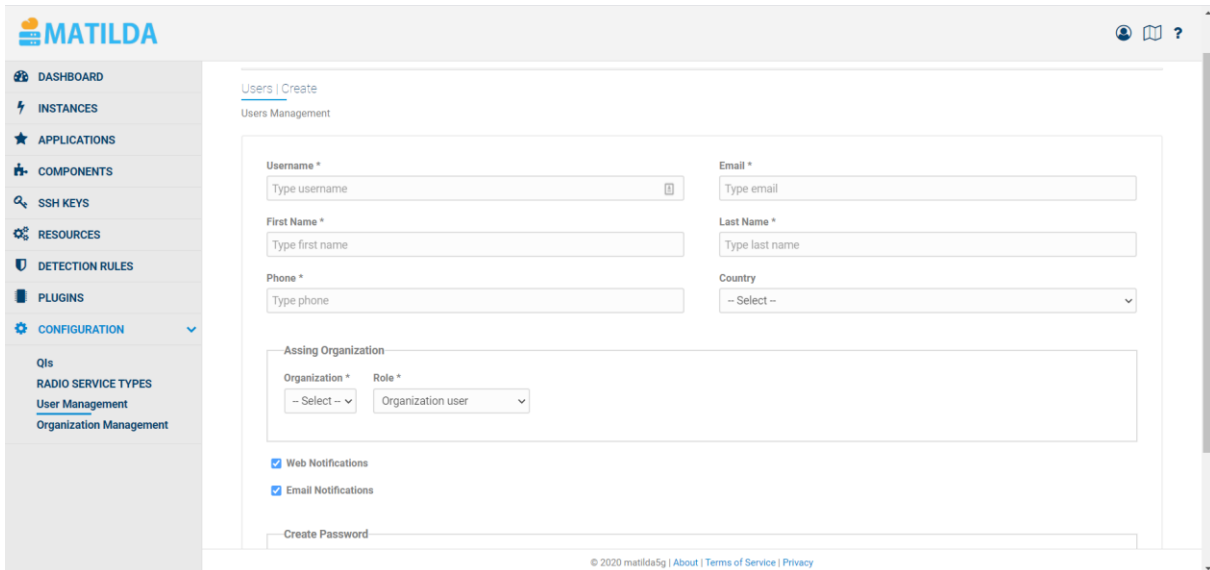


Figure 4: Radio service types' specification

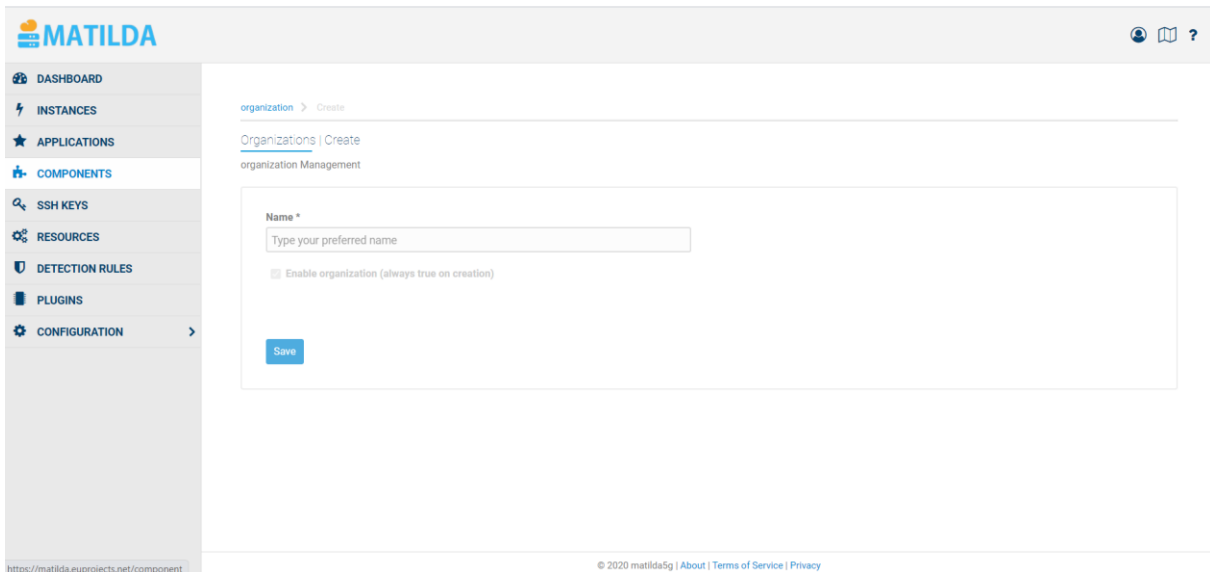


The screenshot shows the 'Users | Create' page in the MATILDA platform. The left sidebar contains a navigation menu with options: DASHBOARD, INSTANCES, APPLICATIONS, COMPONENTS, SSH KEYS, RESOURCES, DETECTION RULES, PLUGINS, and CONFIGURATION. The main content area is titled 'Users Management' and includes a 'Create' link. The form contains the following fields:

- Username ***: Text input with placeholder 'Type username'.
- Email ***: Text input with placeholder 'Type email'.
- First Name ***: Text input with placeholder 'Type first name'.
- Last Name ***: Text input with placeholder 'Type last name'.
- Phone ***: Text input with placeholder 'Type phone'.
- Country**: Dropdown menu with placeholder '-- Select --'.
- Assigning Organization**: A section with two dropdowns: **Organization *** (placeholder '-- Select --') and **Role *** (placeholder 'Organization user').
- Web Notifications**: Checked checkbox.
- Email Notifications**: Checked checkbox.
- Create Password**: A section with a text input field.

At the bottom of the page, there is a copyright notice: '© 2020 matilda5g | About | Terms of Service | Privacy'.

Figure 5: User management options



The screenshot shows the 'Organizations | Create' page in the MATILDA platform. The left sidebar is identical to the previous figure. The main content area is titled 'organization Management' and includes a 'Create' link. The form contains the following fields:

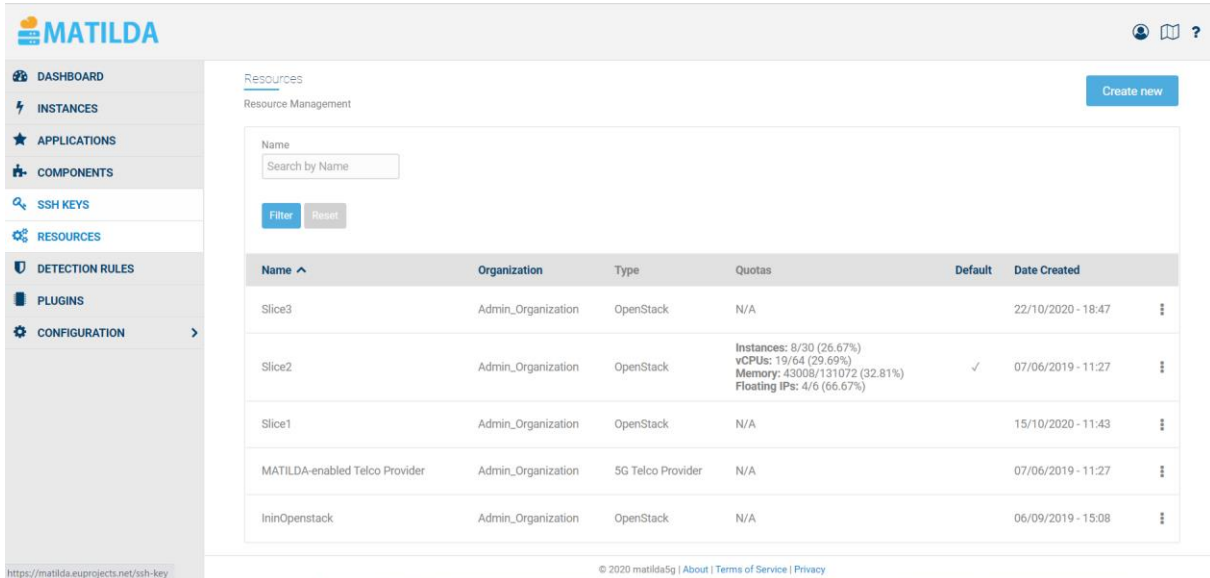
- Name ***: Text input with placeholder 'Type your preferred name'.
- Enable organization (always true on creation)**: A checkbox that is currently unchecked.
- Save**: A blue button to submit the form.

At the bottom of the page, there is a copyright notice: '© 2020 matilda5g | About | Terms of Service | Privacy'.

Figure 6: Organization management options

The next step regards the registration of resources. Since MATILDA follows the separation of concern between cloud and telco world, this is where the user needs to declare his/her available resources (dc, edges). Consequently, this is how the VAO becomes aware of the available resources. Multiple types of resources are supported, like AWS, Google Cloud, OpenStack and more.

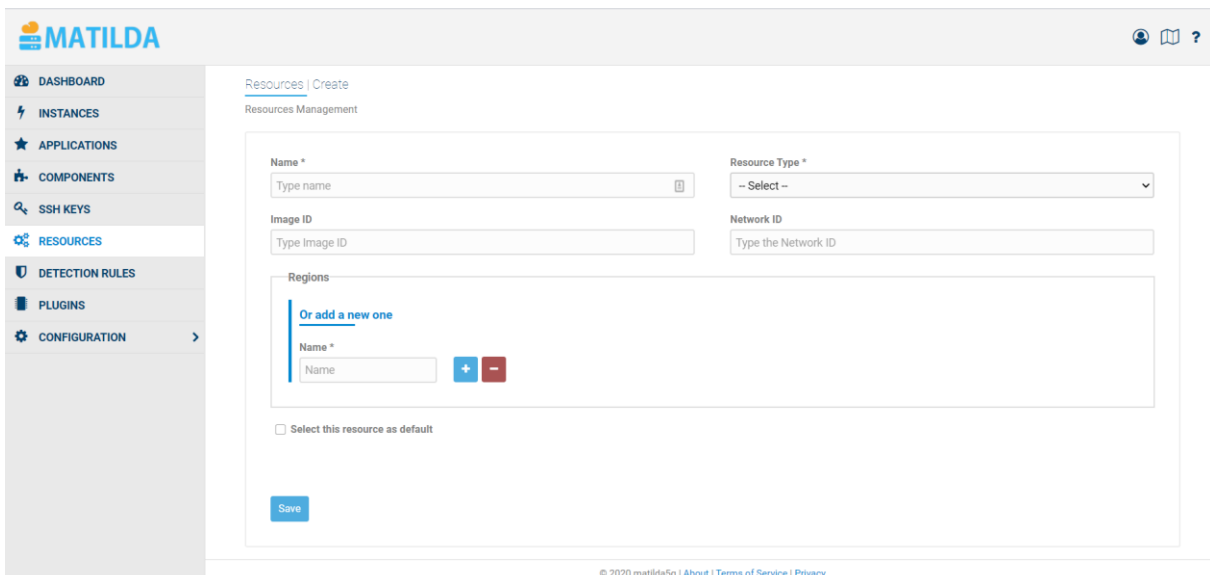
In Figure 7, a sample is provided of the currently used resources and their relevant information. Such information is made available through the MATILDA OSS.



Name ^	Organization	Type	Quotas	Default	Date Created	
Slice3	Admin_Organization	OpenStack	N/A		22/10/2020 - 18:47	
Slice2	Admin_Organization	OpenStack	Instances: 8/30 (26.67%) vCPUs: 19/64 (29.69%) Memory: 43008/131072 (32.81%) Floating IPs: 4/6 (66.67%)	✓	07/06/2019 - 11:27	
Slice1	Admin_Organization	OpenStack	N/A		15/10/2020 - 11:43	
MATILDA-enabled Telco Provider	Admin_Organization	5G Telco Provider	N/A		07/06/2019 - 11:27	
IninOpenstack	Admin_Organization	OpenStack	N/A		06/09/2019 - 15:08	

Figure 7: Registered resources

Additionally, in Figure 8 the options available for configuring a new resource can be seen.



Resources | Create

Resources Management

Name *
Type name

Resource Type *
-- Select --

Image ID
Type Image ID

Network ID
Type the Network ID

Regions

Or add a new one

Name *
Name

☐ Select this resource as default

Save

Figure 8: Resources configuration

Following, the user has to provide a ssh key for enabling the connectivity to the Virtual Machines that wrap every application component (Figure 9). This is a mandatory safety step for securing every application from the cloud perspective. If no SSH keys are configured, the Virtual Machines will not be directly accessible via ssh.

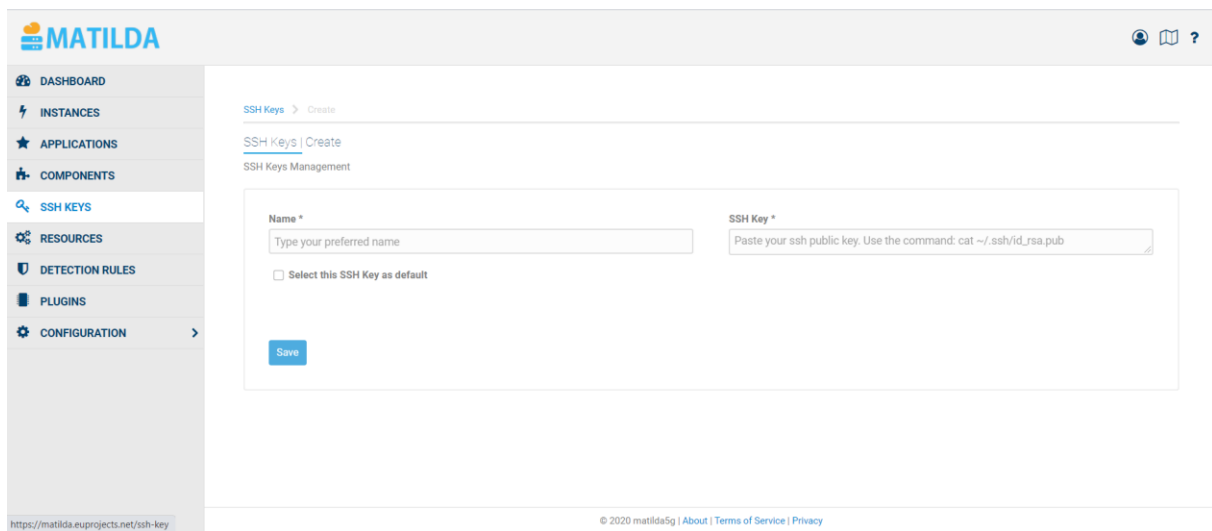


Figure 9: SSH key configuration

Following, the Components Registration is taking place. In this sub-menu (Figure 10) the user registers all the components one by one, creating a component registry which can be later used to compose the application's graph. The component registration process is not so trivial, since MATILDA expects specific parameters to be set.

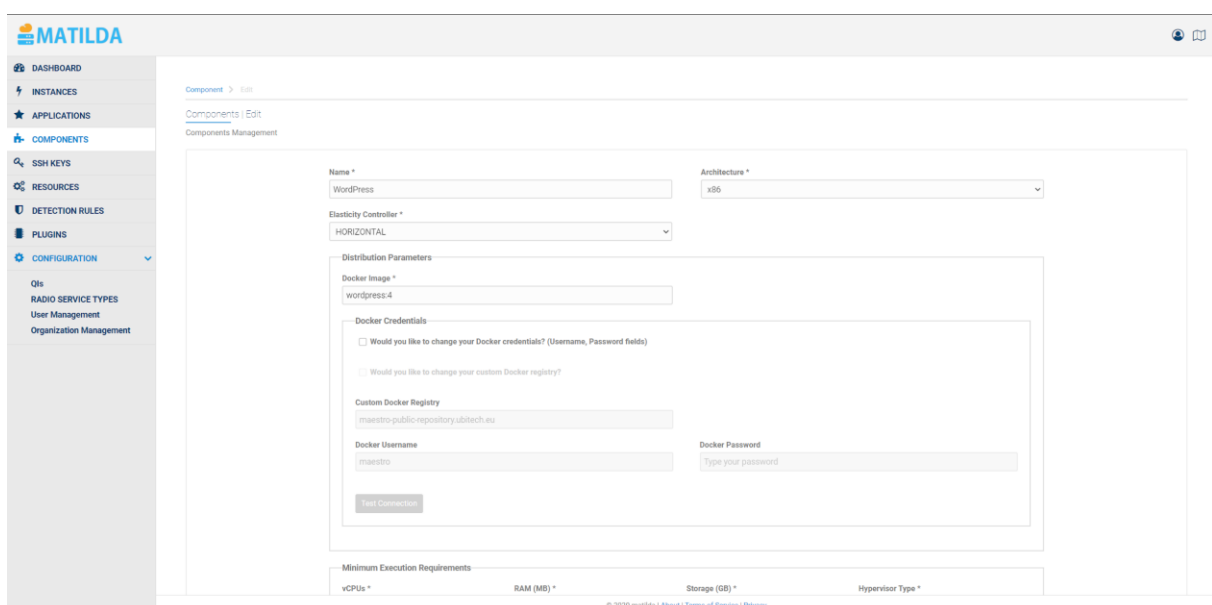


Figure 10: Components Registration

These parameters are the following:

Generic Parameters

- Name of the component
- Architecture needed to be deployed
- Elasticity Controller - declare if the component scales or not.

Distribution Parameters

- Docker Image - declare the specific docker image.
- Docker Credentials - if the docker image repository is private/the image is not publicly accessible.

Minimum Execution Requirements - This section sets the minimum system requirements for the component to be able to operate.

- vCPUs - number of required CPUs
- RAM (MB) - amount of RAM required
- Storage (GB) - total storage needed (in GB)
- Hypervisor Type - the type of hypervisor to be used
- GPU-Enabled - check if the component needs GPU resources for tasks.

Health Check - used for validating that the application component is up and running. The orchestrator checks that at every given time interval.

- HTTP - the url on which the orchestrator will perform the request
- Command - the specific command to be performed
- Time Interval (in seconds) - the time between each consecutive requests.

Container Execution - Optional - Command - Specific commands for executing Dockerfile are written here.

Environmental variables - Optional - Sets of Key Value pairs can be created here to be inserted as environmental variables in the component runtime.

Exposed Interfaces - Optional - The interfaces this component will expose.

- Name - name for the interface
- Port - the tcp/udp port on which the interface will be exposed on
- Interface Type - core or access (access is used to declare a hooking point for a User Equipment on OSS)
- Transmission Protocol - tcp, udp or both.

Required Interfaces - Optional - This field is for declaring dependencies among other components using pre-configured Interfaces.

Plugins - Optional - Extra software artifacts that the user would like to install inside the Virtual Machine.

Volumes - Optional - For declaring if volume support is needed for the application component.

Labels - Optional - Labels for component categorization.

Advanced Options

- Hostname - hostname capabilities for Docker.

Linux Capabilities - different capabilities can be set here like SYS_ADMIN, NET_ADMIN etc.

ulimit memlock - provides control over the resources available to the shell and to processes started . The soft limit is the value that the kernel enforces for the corresponding resource.

Docker Execution User

- Network Mode Host

Privileged Mode

Upon the registration of the applications components, the next step is to connect the components and form an acyclic directed graph. To do so, the user will need to use the Application sub-menu and create a new application.

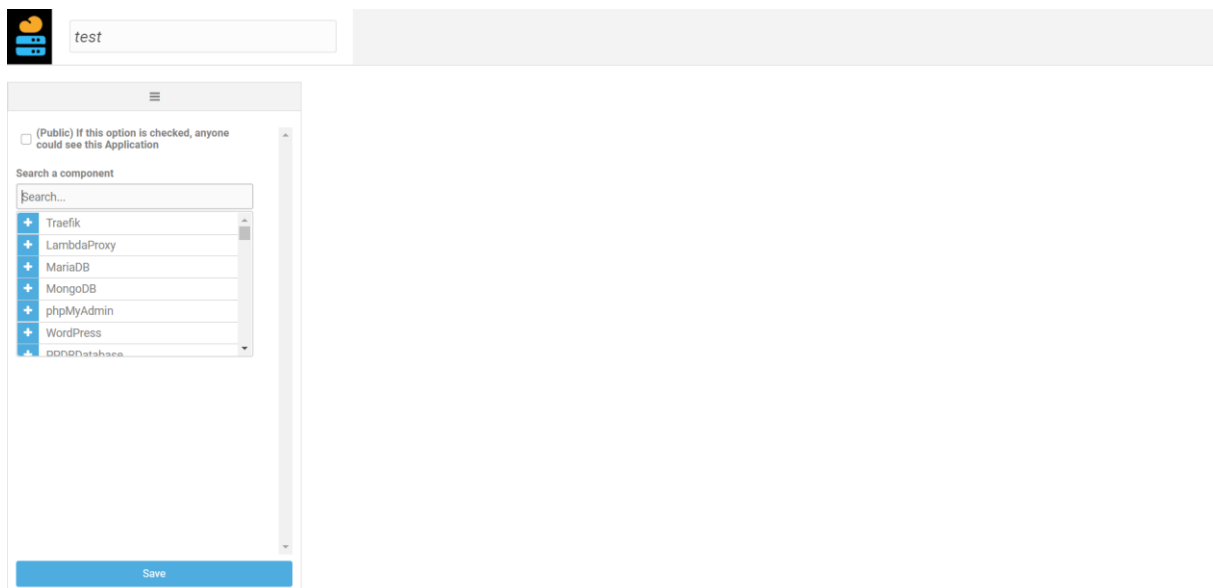


Figure 11: Application Composition

Providing a name for the application is mandatory using the textbox in the top left corner. The user will then need to select the components that comprise the application via the drop-down menu. Upon selection of the components, the user can click and drag one component towards another to create a “link”. A connection will be created with a popup window requesting user confirmation for the newly created interface (Figure 12).

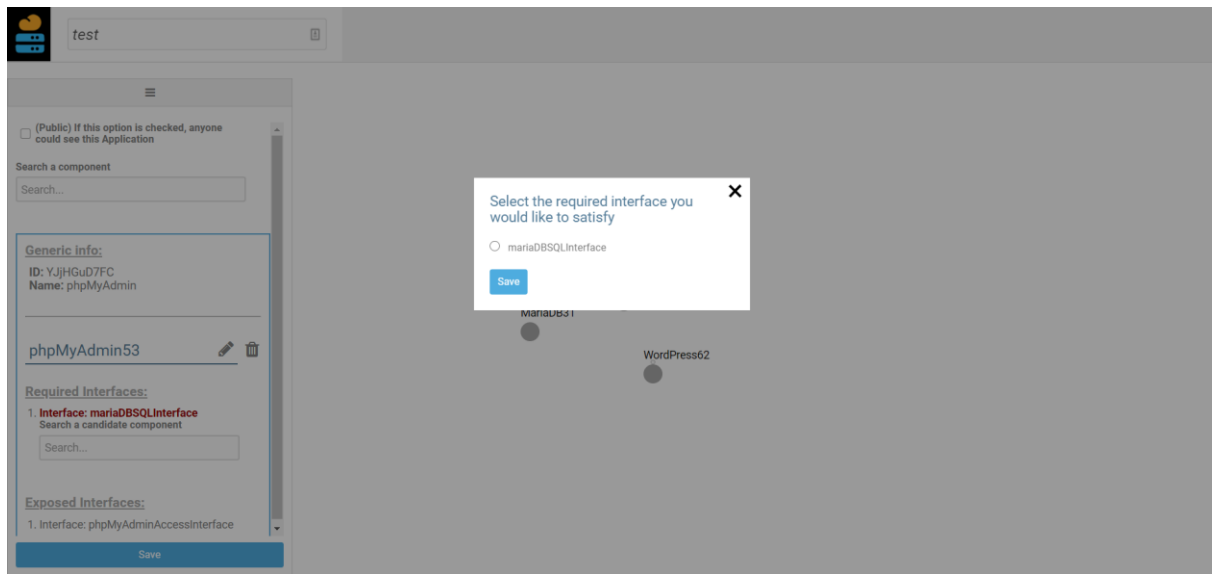


Figure 11: Interlinking of Components

The application graph is finalized after all the required interfaces of the components are satisfied. Following the application finalization, the user can instantiate it, creating in this way an actual running instance of his/her application graph. This can be achieved through the applications' list (in the applications menu), by going on the specific application and clicking "Create Instance", as shown in Figure 12.

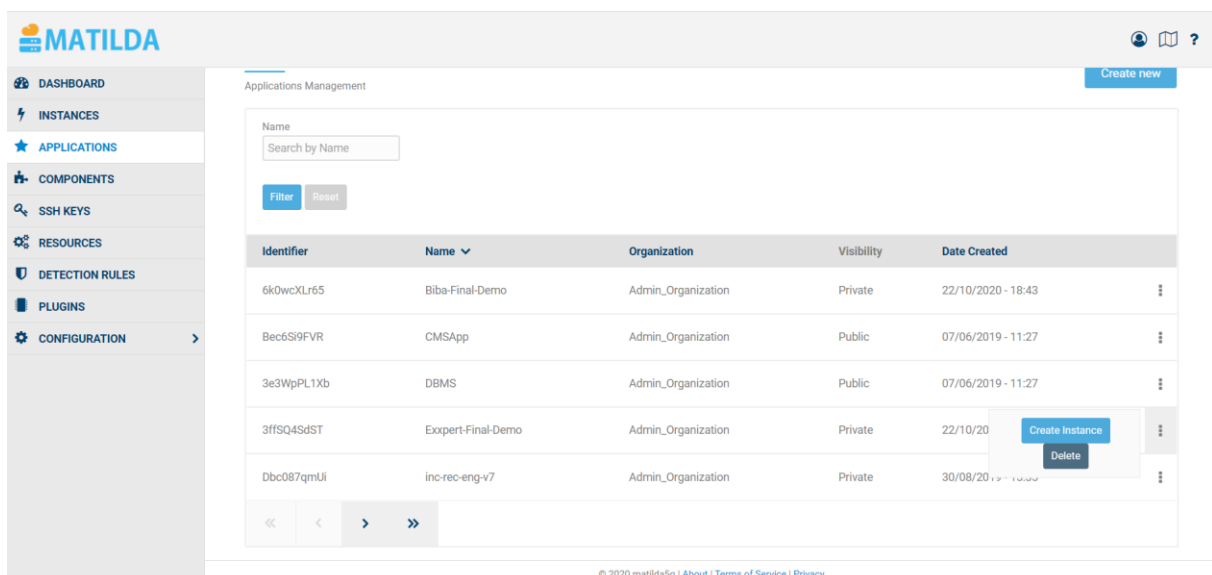


Figure 12: Application Instantiation

This will lead to the Graph Editor (Figure 13), where after naming the instance, additional instance specific configurations can be done.

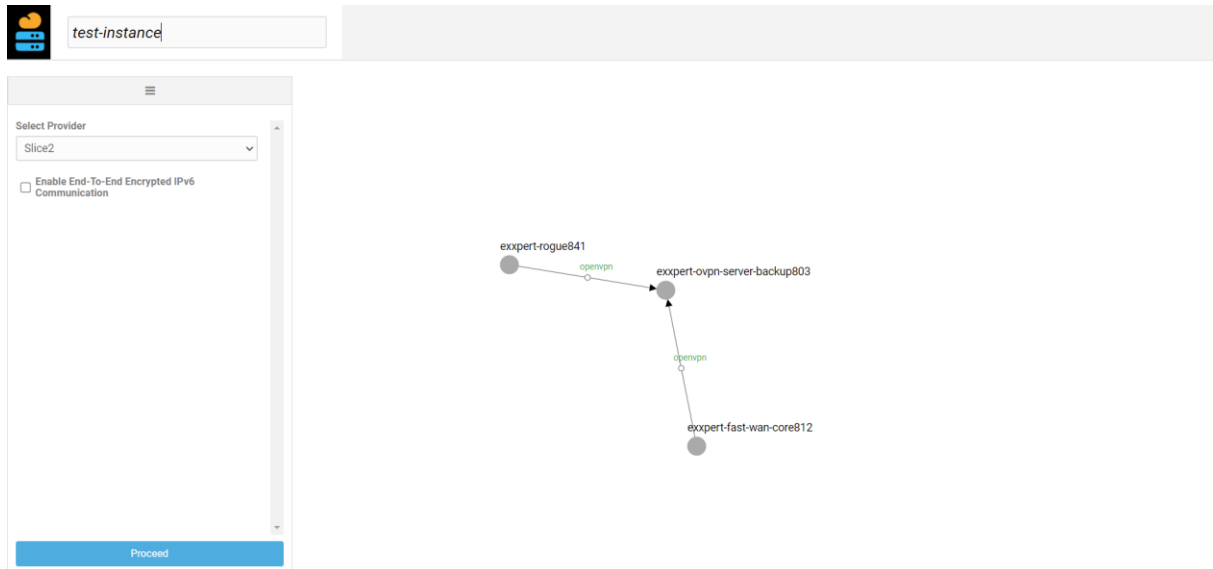


Figure 13: Graph Editor

By clicking on a component, the user can make component-specific configurations for this instance. Some of the parameters that can be tuned in this last-mile configuration will override the initial values, while other additional parameters are only instance-specific and can be configured here (Figure 14).

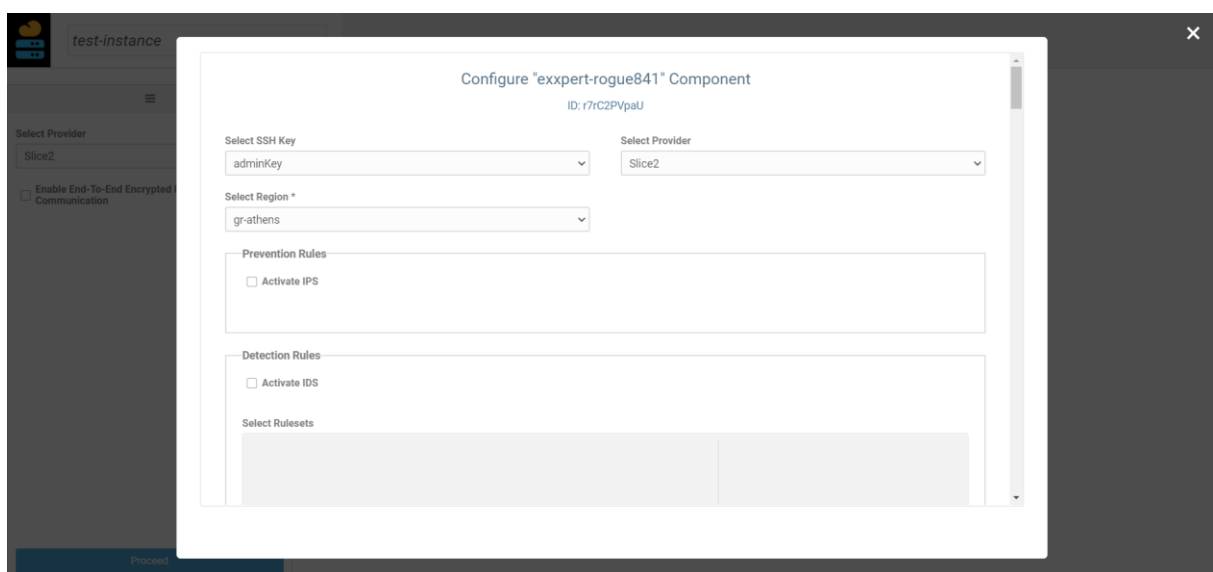


Figure 14: Configuration of deployed component

Additionally, the user can also define the QoS for the links connecting the components (Figure 15). The list of choices is:

- Maximum Delay (ms)
- Maximum Jitter (ms)
- Maximum Packet Loss (%)
- Throughput (kbps)

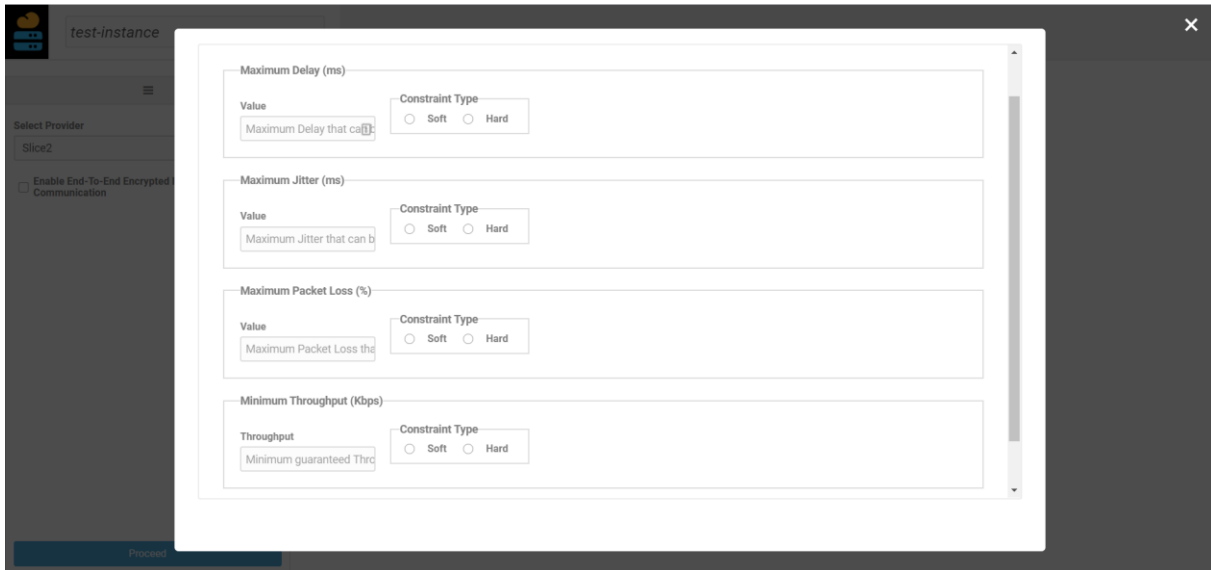


Figure 15: QoS requirements specification

After finalizing the configuration for each link (if necessary), the user can then deploy the graph instance by clicking on the *Proceed* button. This is the last step before the deployment starts. After that, in the INSTANCES menu the user can see his/her application deployment and the status of it through detailed log entries (Figure 16).

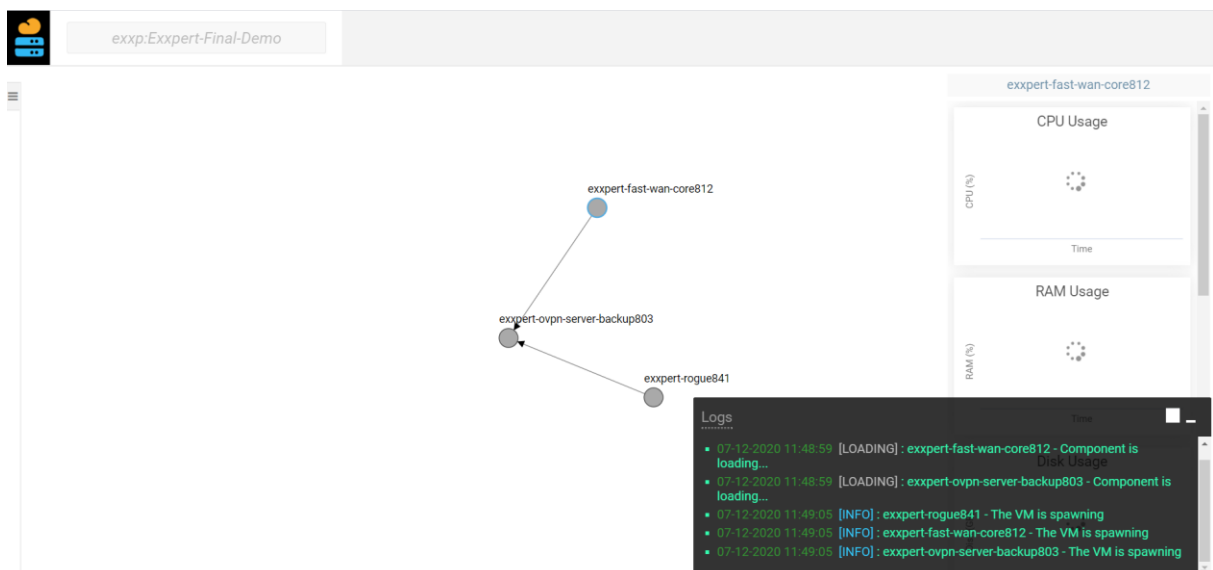


Figure 16: Deployment instance information

One of the features is the Real-Time Monitoring engine. By making use of Prometheus, various system and networking metrics can be accessed and displayed for each of the deployed components (Figure 17 and Figure 18).

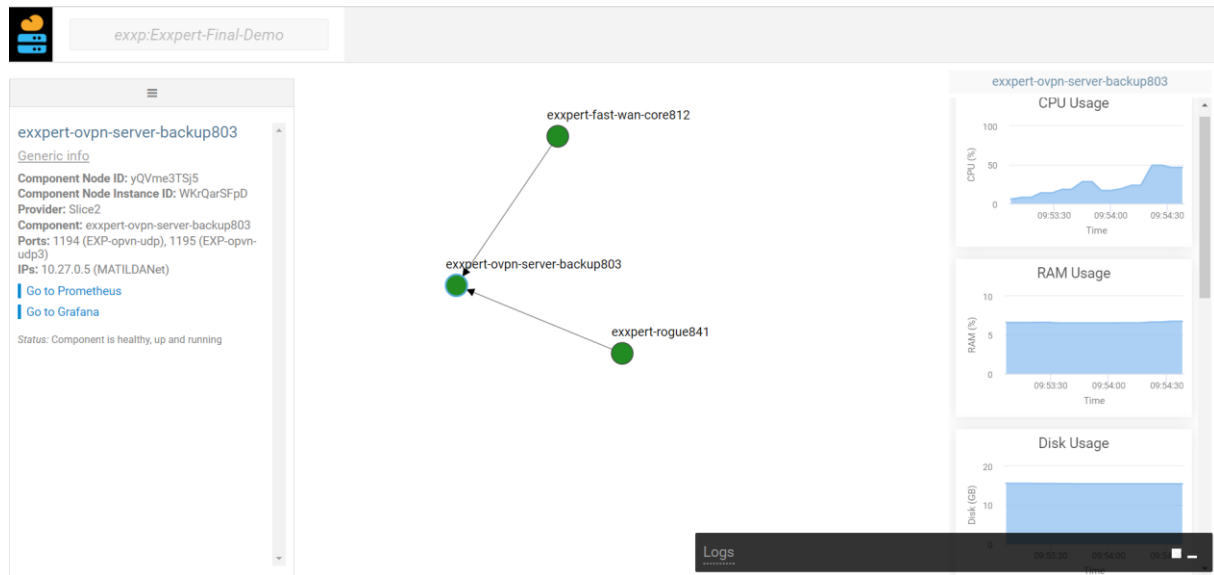


Figure 17: Deployment instance monitoring information



Figure 18: Deployment instance monitoring information

The Elasticity Framework is a post deployment feature of MATILDA. Through the GUI (Figure 19), the user can compose his/her elasticity policies and apply them to the already-deployed application. These elasticity policies allow for fine-tuned automated scaling depending on an entirely custom set of parameters. By making use of the various metrics of

each component, the elasticity policy can combine these metrics as defined by the user to decide when a component should be scaled in or scaled out, or just notify the user of the conditions being fulfilled.

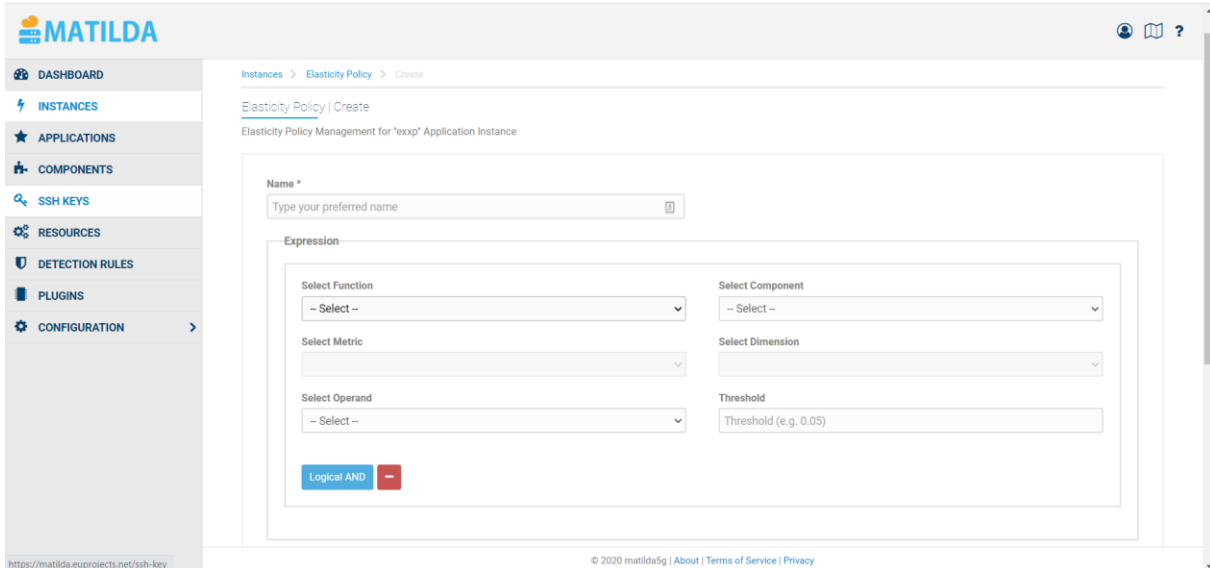
The screenshot shows the 'Elasticity Policy | Create' form in the MATILDA web interface. The left sidebar contains navigation links: DASHBOARD, INSTANCES, APPLICATIONS, COMPONENTS, SSH KEYS, RESOURCES, DETECTION RULES, PLUGINS, and CONFIGURATION. The main content area is titled 'Elasticity Policy | Create' and 'Elasticity Policy Management for 'excp' Application Instance'. It features a 'Name' field with a placeholder 'Type your preferred name'. Below this is an 'Expression' section with two columns of dropdown menus: 'Select Function' (with '- Select -'), 'Select Metric', 'Select Operand' (with '- Select -'), 'Select Component' (with '- Select -'), and 'Select Dimension'. A 'Threshold' field with a placeholder 'Threshold (e.g. 0.05)' is also present. At the bottom of the expression section are 'Logical AND' and '-' buttons. The footer of the interface shows the URL 'https://matilda.euprjects.net/ssh-key' and copyright information '© 2020 matilda5g | About | Terms of Service | Privacy'.

Figure 19: Elasticity Policies specification

Specifically, the parameters available for the elasticity policy are as follows:

- Name - the name of the policy
- Expression - the logical expressions which will be evaluated. Multiple expressions can be added (logical AND)
 - Function - the function to be applied on the metric.
 - Component - the component which the elasticity policy will monitor.
 - Metric - the metric/metrics which the elasticity policy will monitor.
 - Dimension - the dimension of the selected metric.
 - Operand - the operand to be used to evaluate the result of the metric function against the threshold (equals, greater than, etc.)
 - Threshold - the threshold value which the function result will be evaluated against.
- Period - interval on which the expression is evaluated
- Inertia Time - interval between a successful trigger of the policy and the next expression evaluation.
- Actions - Actions to be performed on policy trigger
 - Type - the type of action, scale in, scale out, info, push on topic.
 - Select component - the component which is the target of the action.
 - Message - when the type of action is info, the message to be transmitted is written here.

As is the case with functions, multiple actions can be triggered by a single elasticity policy targeting a variety of different components.

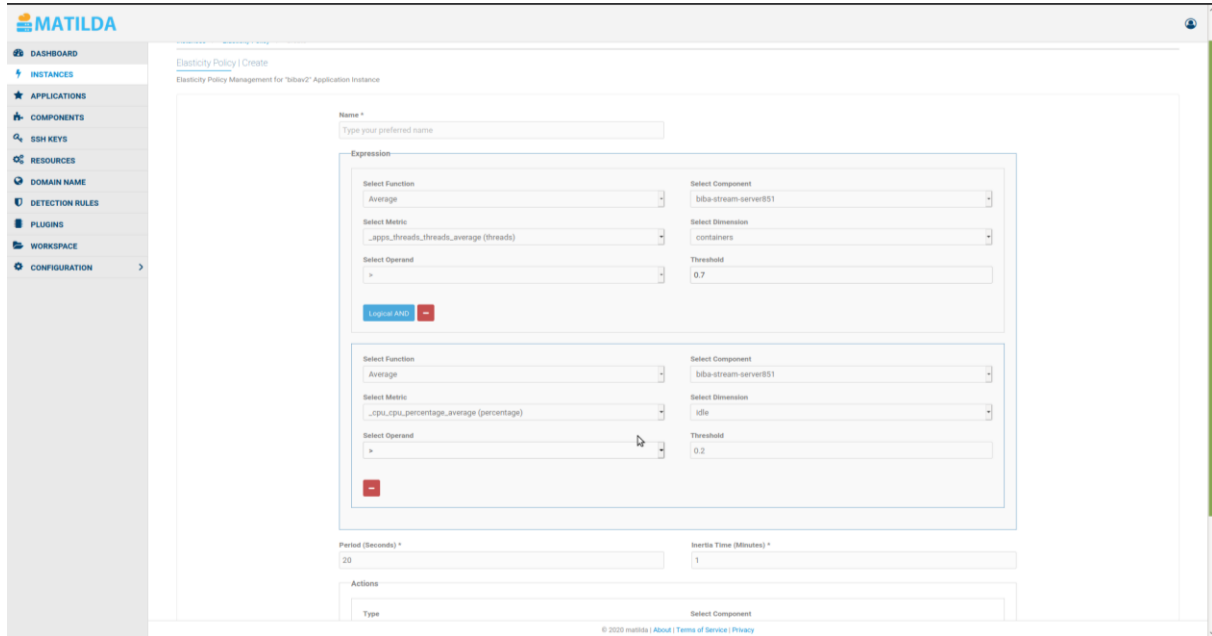


Figure 20: Elasticity Policies actions definition

The Security Framework is a post-deployment feature, as well. Leveraging different technologies (XDP, eBPF, Snort) MATILDA can provide some security features like:

- Intrusion Detection
- Pre-stack processing like filtering to support DDoS mitigation
- Forwarding

Users have to define specific fields like Action, Rules, IP address of the component, in order to compose their security policy (Figure 21).

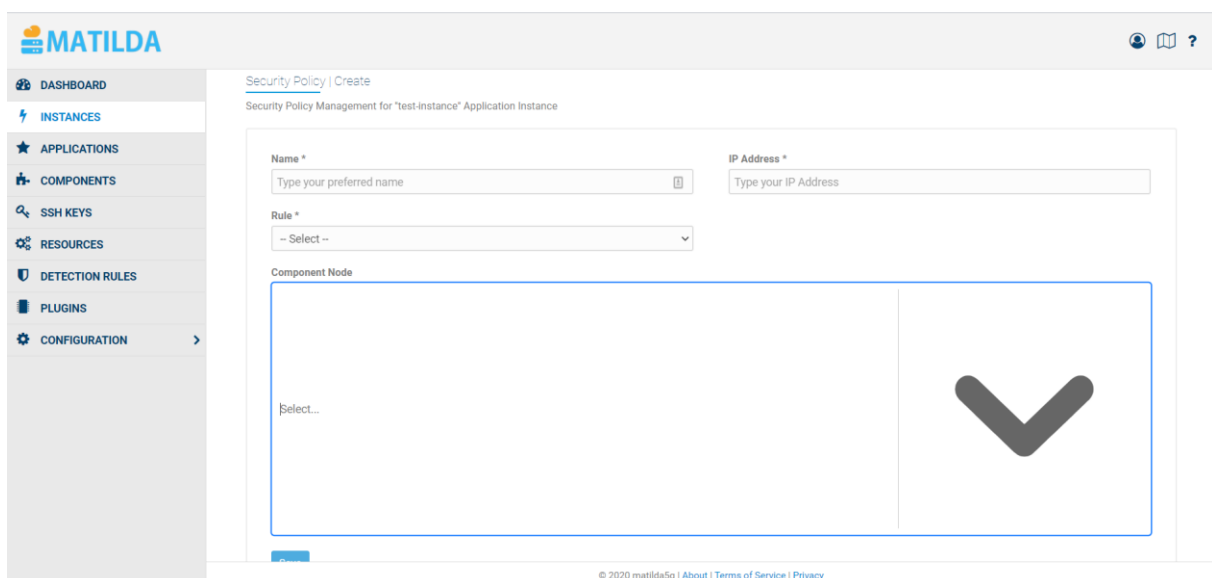
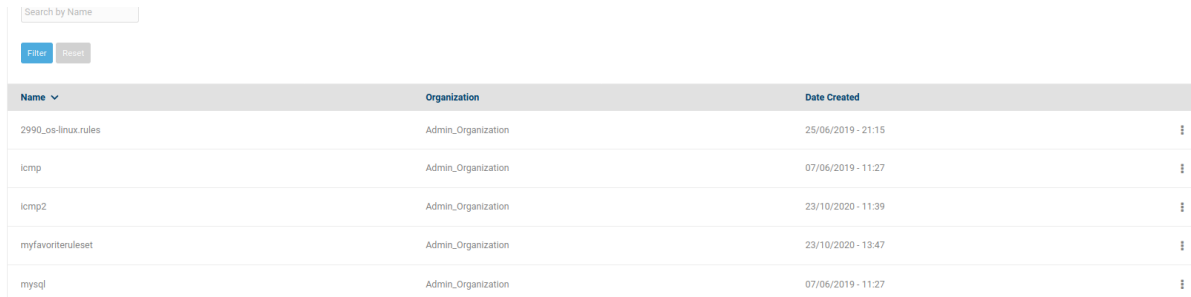


Figure 21: Security Policies definition

The users can also provide their custom detection rules that will trigger different kinds of alerts (Figure 22).



The screenshot shows a web interface for managing security policies. At the top, there is a search bar labeled 'Search by Name' and two buttons, 'Filter' and 'Reset'. Below this is a table with three columns: 'Name', 'Organization', and 'Date Created'. The table contains five rows of data, each representing a different security policy rule.

Name	Organization	Date Created
2990_os-linux.rules	Admin_Organization	25/06/2019 - 21:15
icmp	Admin_Organization	07/06/2019 - 11:27
icmp2	Admin_Organization	23/10/2020 - 11:39
myfavoriterulest	Admin_Organization	23/10/2020 - 13:47
mysql	Admin_Organization	07/06/2019 - 11:27

Figure 22: Security Policies alerts

These operations are completed with the generation of a message (the so-called *slice intent*) from the VAO to the MATILDA Telecom Layer Platform (TLP) that triggers the creation of a network slice in a fully automated way and transparent to the application developer. Details on the operations performed at the TLP level can be found in the Deliverable 4.2, “Network and Computing Slice”, publicly available at <https://www.matilda-5g.eu/index.php/outcomes>.

It is worth noting that this automated operation (in a Zero Touch Service Management – ZSM – fashion) is one among the main achievements of MATILDA, which allows the application developer to operate by simply instantiating Virtual Infrastructure Managers (VIMs) in the network, rather than in the cloud.