II. Netbus Pro System

Software Manuals

Ver.1.3

II. Netbus Pro System - Software Manuals

II. Netbus Pro System - Software Manuals

1. Netbus Pro System Overview

Netbus Pro System is a “Cloud Software Platform” which contains several services as “data acquisition, collection, management, storage, presentation” which supports “modern data exchange protocols” as well as “conventional automation protocols” to be able to provide “modern, flexible, scalable and effective” solutions.

Netbus Pro System Block Diagram

The block diagram of Netbus Pro System is given below representing several services and integration capabilities of the system.

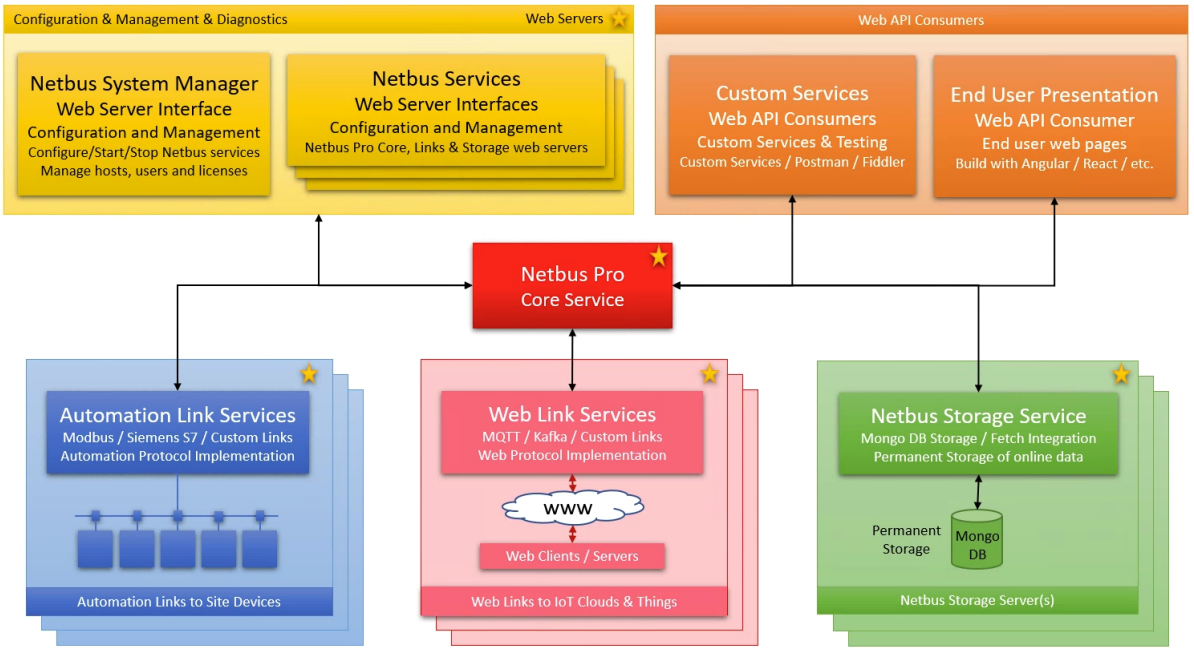


Figure 1: Block diagram of Netbus Pro System

Netbus Application = BackEnd service + Web site

Netbus Pro System consists set of services to achieve data management

Netbus Pro Core Service : Core application, data management & presentation
Netbus Storage Service : Storage application, data storage and fetch servers
Automation Link Services : Links to Automation Devices
Web Link Services : Links to Web Systems
Configuration & Management & Diagnostics : Web server interfaces
Web API Consumers : Custom Services and End User Presentation services

2. Netbus System Manager Web Server

2.1. Introduction

Netbus System Manager service is a web server and the main user interface for management & configuration & diagnostics of a Netbus device. In the process of Netbus Pro System Software setup, this service is installed as a Linux service and added to the boot startup sequence to be available always for management of the system.

This web server is the main management utility of the Netbus System.

2.2. Index Page

Netbus System Manager index page is the main hub which contains short explanations and several links to internal functionality as well as other Netbus service and explorer.

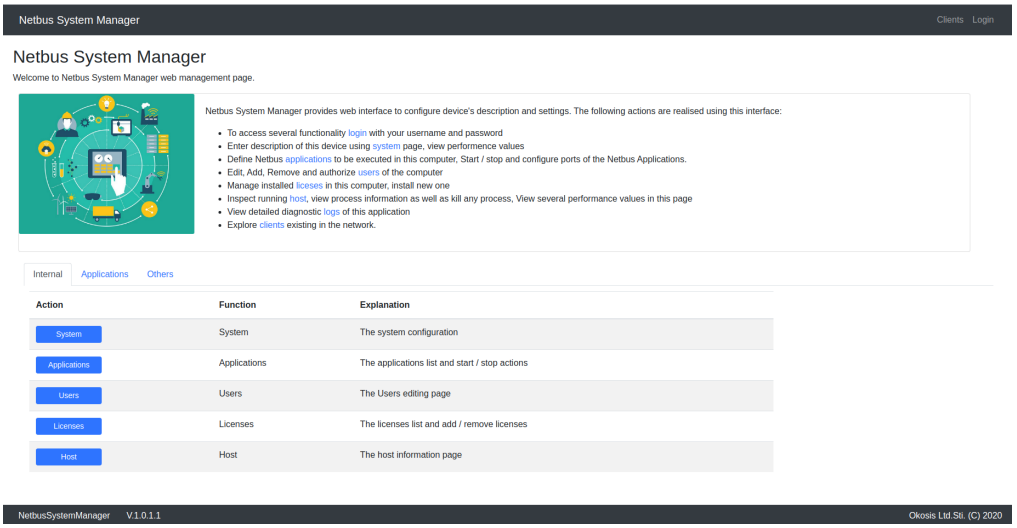


Figure 2: Netbus System Manager Web Server index page before login

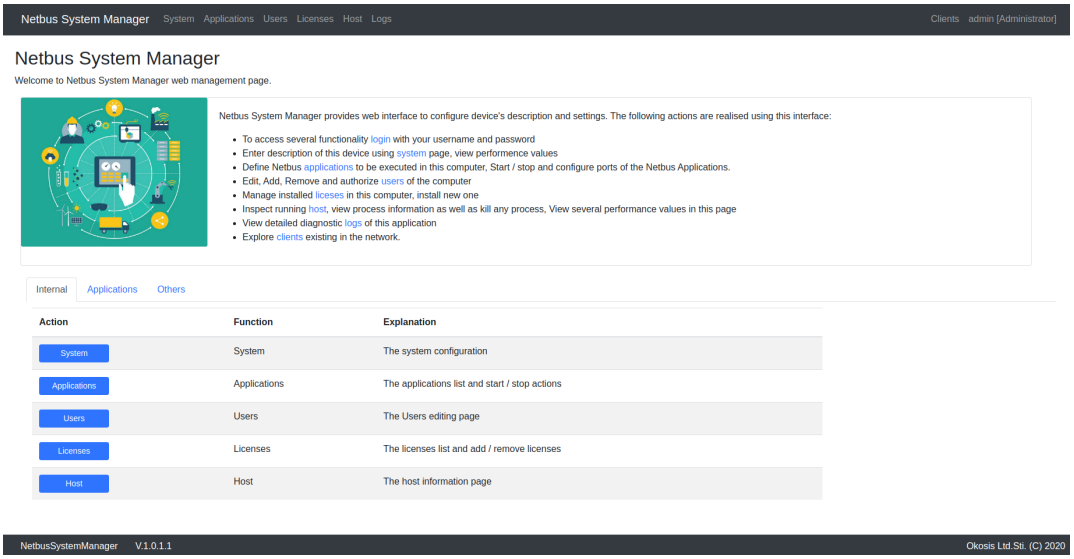


Figure 3: Netbus System Manager Web Server index page after successful login

2.2.1. Index Page Components

Index page contains **“menu bar”**, **“upper intro part”** and **“lower tab pages part”**.

2.2.1.1 Menu bar

The menu bar provides links to internal functionality, see section: **“2.2.2.Menu Bar”**.

2.2.1.2 Upper part

The upper part gives intro and provides links for internal functionality:

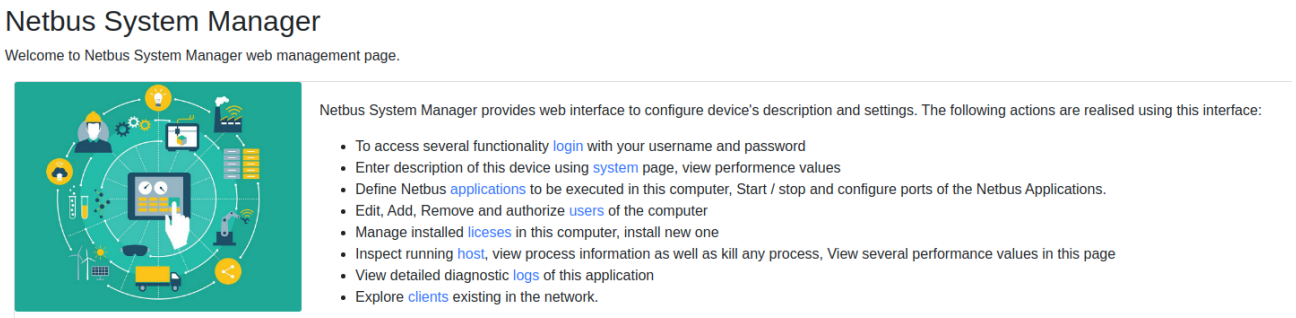


Figure 4: Index page’s upper intro part

2.2.1.3 Lower tab pages part

The lower tab page part contains organized links given as below:

Internal tab page provides links to its own functions which are also given on the menu bar. Application tab page shows the Netbus Pro Services and provides links to these websites. The Other tab page provides links to Logs, Clients explorer, Login interface.

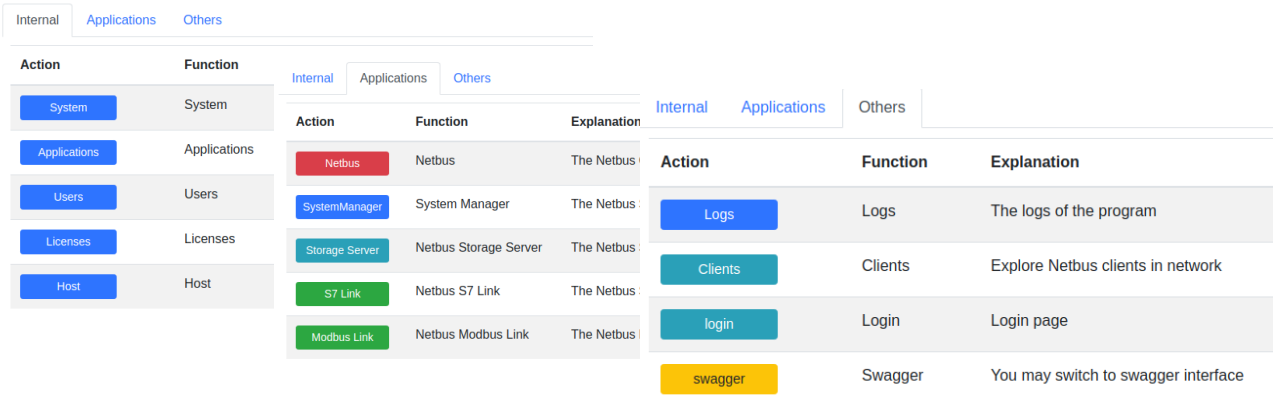


Figure 5: Index page’s lower tab pages part and it’s tab pages

2.2.1.4 Bottom Toolbar

The bottom toolbar contains following fields:
“Name of the service”, **“version”**, “company name” and “year of production”.

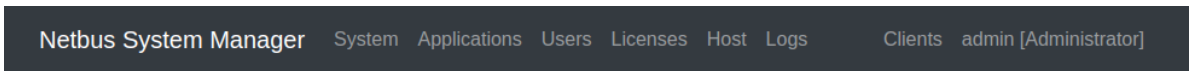
You may check the release by reviewing the “version” field which is visible always.

2.2.2. Menu Bar

On the top menu bar there are links for accessing functions according to login level.

If the user is logged out, only the “Client” and “Login” link are available for the user.

If the user logged in with Administration Level, the following functionality is available :



Web server presents following management functionality

System: Netbus system definition shared to Netbus systems exists on the network
The data entered on the System ID tab of this page is shown to all the explorers in the same network at their Client page when they perform exploring action.

Applications: Configuration & management of Netbus services run on this device
Netbus Pro System services which are planned to run on this host are entered here, it is possible to start / stop and configure the services through this page.

Users: User & Role definition of the Netbus System is done in this page.

Licenses : Installed licensed listed and managed on this page

Web server presents following diagnostic functionality

Host: The host system performance values and running processes are shown here and provides functionality to stop processes manually.

Logs: Netbus System Manager service Logs are shown on this page

Web server presents following utility page to explore clients

Clients: Netbus clients on the network are explored and shown in tabular view with a link providing access to this device’s **Netbus System Manager** web site.

2.3. Login Page

On the top right corner of the window there is a “Login” button that provides access to the login window. Please enter username and password and click the login button.

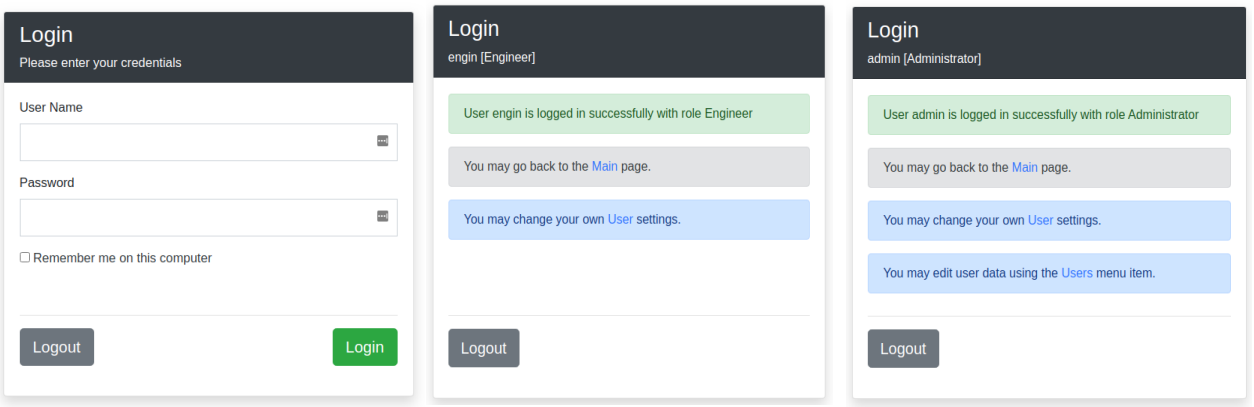


Figure 6: Netbus System Manager login window, after Engineer / Administrator level login

Role based login model used & some functions are accessible for specific roles only. For example, only administrators can see the **“Users”** page and define / change the user list.

After successful login the top toolbar shows the additional menu items to provide access to additional functionality which requires user authentication.

There is a unique **“Login Page”** which is valid for all the Netbus Pro System services. The login / logout action which is performed successfully in the Login page of the **Netbus System Manager** will be valid for all Netbus Pro System applications as well.

2.3.1. User Roles and Authority Matrix

Function	Actions	Administrator	Engineer	Operator
System	Change system properties	**OK**		
Applications	Edit / Add / Delete & Start / Stop	**OK**		
Users	Edit / Add / Delete & Assign Role	**OK**		
Licenses	Edit / Add / Delete	**OK**		
Host	Edit host definition	**OK**		
Logs	View Logs	**OK**	**OK**	
Login	Log in / Log out	**OK**	**OK**	**OK**
Clients	Explore clients	**OK**	**OK**	**OK**

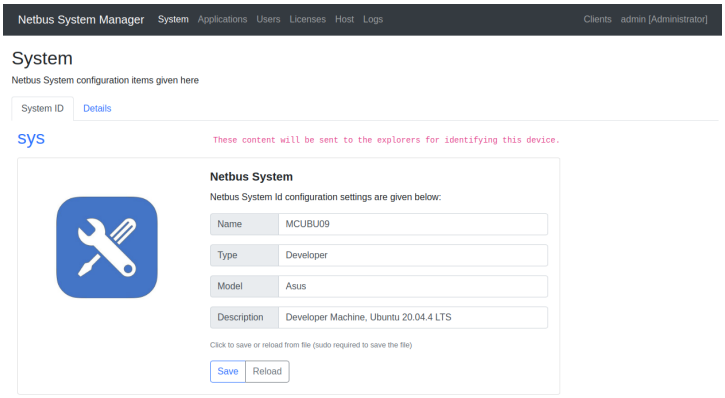
Table 1: User Roles and Authority matrix

Actual version V.1.0.1.1. does not have any authentication limit for the WebAPI access. It is allowed to get / post data from WebAPI without any authentication requirement.

2.4. System Page

System Page contains a short introduction and a tab page below which contains two tabs containing “**Netbus System”** properties given as **“System ID”** and a **“Details”.**

2.4.1.1 System ID Tab Page



“System ID” tab is the place where you enter “**Netbus System”** properties as:

Name: Name of the Netbus System. (Might be your host name or different)

Type : The category / type of the host, Please enter a descriptive text here.

Model: The model and the version of Netbus System is entered here.

Description: You can enter any arbitrary descriptive text into this field about the host.

Figure 7: System Page

The above properties are shown on “Clients” page’s tabular explored clients view :

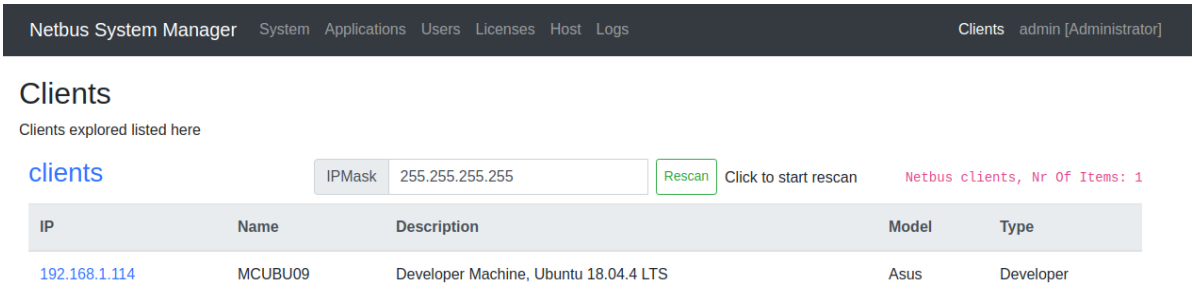


Figure 8: Netbus Clients Exploration procedure and System ID fields

For details of the client page, please review the section **2.10 Clients Page**

2.4.1.2 Details Tab Page

System page’s “Details” tab contains Host’s system values as System, CPU, Memory, Program information & performance & diagnostic values. you may enter “Filter” string for the Item column. If you don’t want to filter please enter “*” text into Filter area.

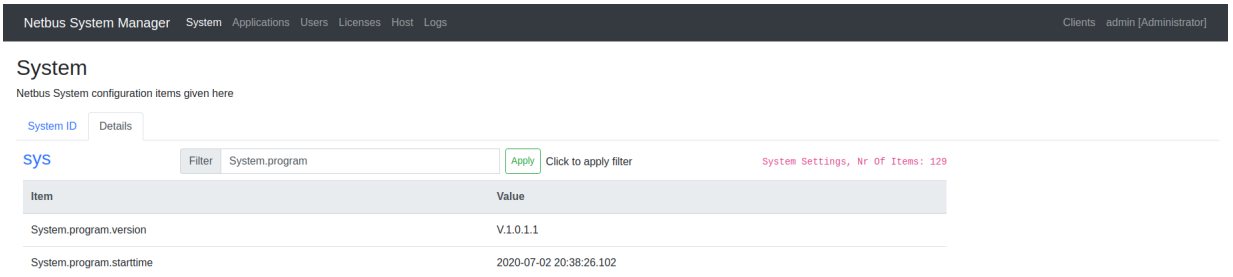


Figure 9: System - Details page

2.5. Applications Page

Application page contains a short introduction and a tabular list that contains **“Applications”** which is referred to as **“Netbus Service”** and its **“Web Server”** in this document context.

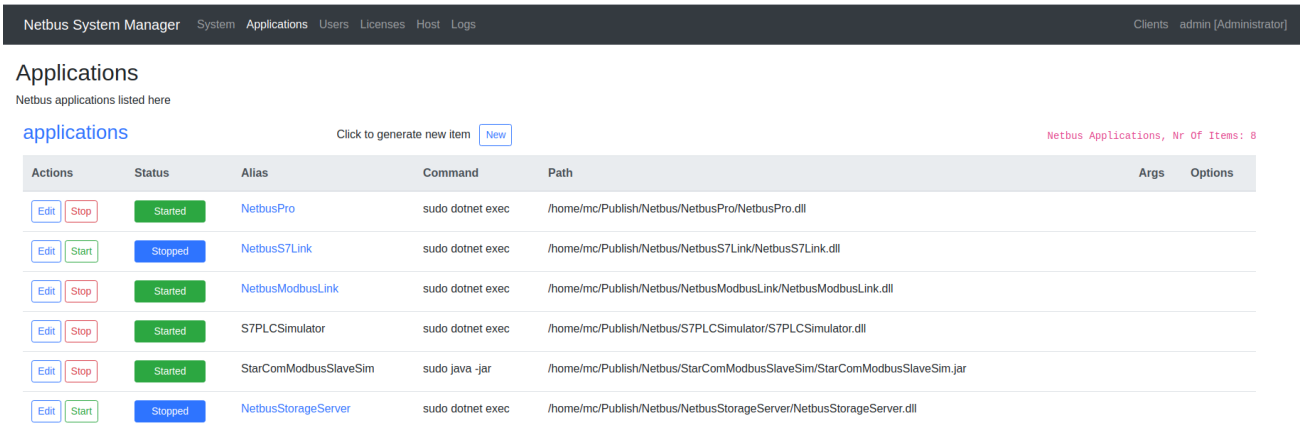


Figure 10: Applications list

In this list, the Applications which will be running on this very host are defined by the user. After the installation, a full set of services are listed here.

2.5.1. Action column

Action column contains several buttons Edit / Start / Stop to perform described actions.

2.5.1.1 Editing Applications

Figure 11: Applications editor

When you click the **“Edit”** button on the tabular view, the applications editor appears.

The editor forms contains following buttons:

Cancel: cancel operations, close editor.

Delete: delete selected item.

Update: update selected item with new data

Also, you may close the window by clicking the X symbol on the right hand top corner of the editor window. That is equal to Cancel.

If you click the Delete or Apply button, a confirmation popup appears and asks for confirmation of the operation to proceed.

At the lower part of the editor window there is a message area which shows the status of actions.

2.5.1.2 Application Editor Fields

The “Edit Application” window contains following fields:

alias: Service name for example: **“NetbusPro”,”NetbusStorageServer”, etc.**
You may enter any descriptive text here, please don’t change if it is not necessary.

port: Enter any available port number here.
Please be careful, changing this port number might cause service interruptions.
This port number might be defined in other systems which have access to this service.

Command: The execution command for the host. Please do not change this text.

Path: The path of the service. Services unzipped in the /home/Publish/Netbus folder.

args: Extra parameters for the service are given here.
This text is forwarded to the related service as run-time arguments.
Please keep empty if otherwise not specified.

options: Extra options is given on this field, the possible options for this field are:
“autostart”: Start service on boot up.

2.5.1.3 Starting & Stopping Applications

Action column contains **“Start” & “Stop”** buttons for manual starting and stopping of applications. For starting & stopping applications, click the relevant button. If the start / stop operation succeeds, the Status will be changed and if you click the “Started” (or “Stopped”) status button after the operation, you may see the detailed status of action.

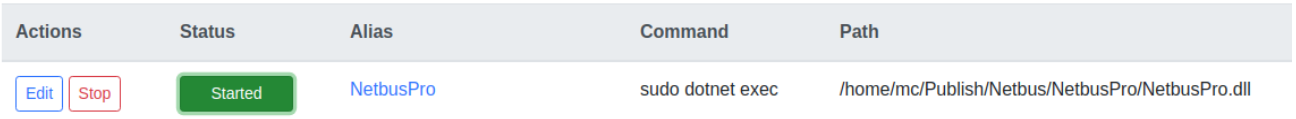


Figure 12: Started Service

2.5.1.4 Adding and Deleting Applications

You may add Applications to the list as well as delete any service.

2.5.1.5 Optional Actions

You may define new services by clicking the “New” button and entering data.

You may select the services you don’t want to use in this host by clicking the edit button and clicking “Delete” to remove it from the list.

2.5.2. Status column

The Status column shows the services actual status as : Started / Stopped / Error.

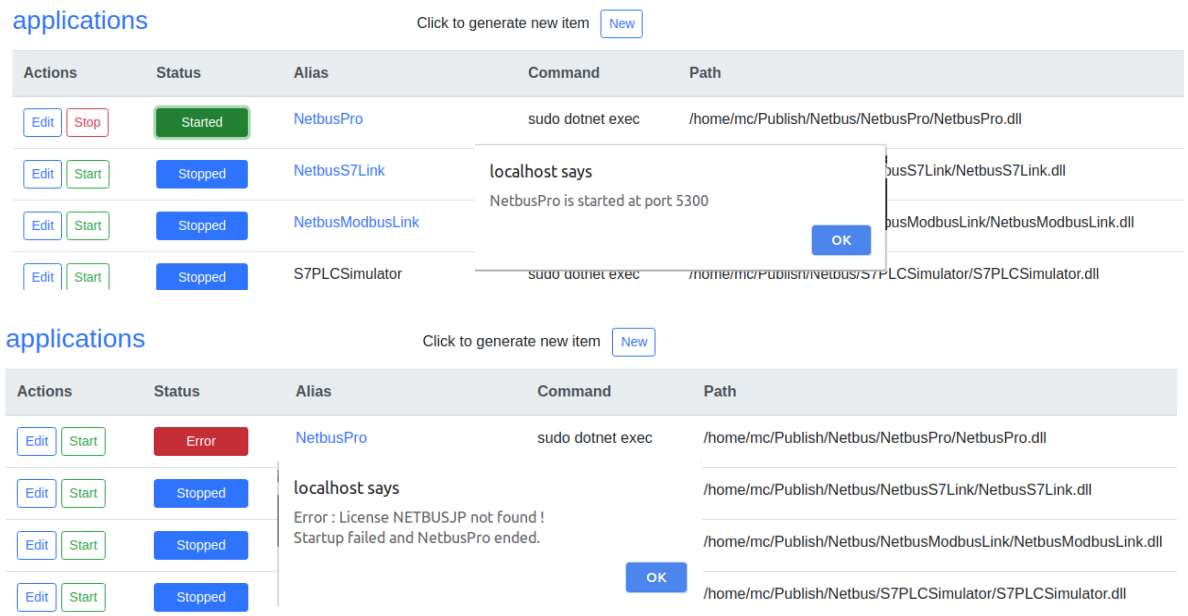


Figure-13. Status view of action – Start operation succeed and failed examples

2.5.3. Alias column

The Alias field contains the alias of service and a web Link for this service.
You may click that link to go to the web server of this specific service.

2.5.4. Command column

The execution command for host. Please do not change this text.

2.5.5. Path column

The path of the service. Services unzipped in the /home/Publish/Netbus folder.

2.6. Users Page

Users page provides a short description and a tabular view containing users list which displays user name / role, first / last name and an action field with **“Edit”** button.

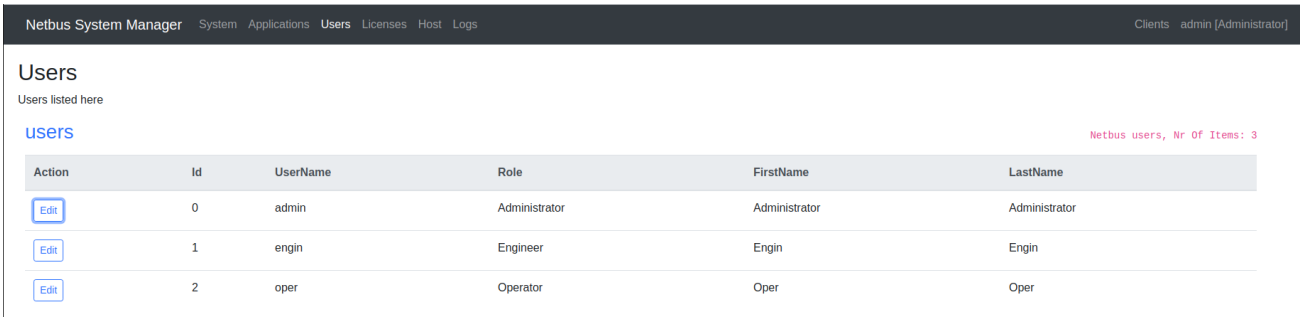
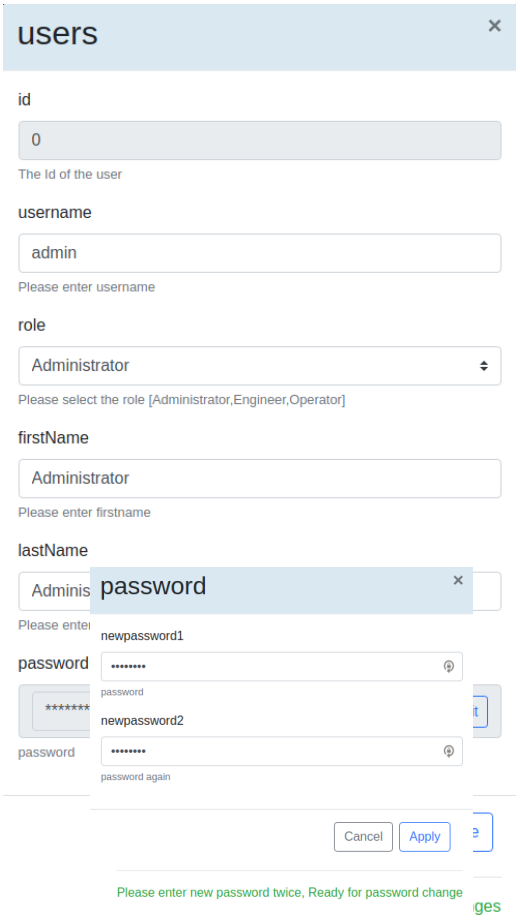


Figure-14. Users Page and User List

2.6.1.1 User and Password Editor

Clicking Edit button will open up following popup which provides functionality as add/delete user & change and apply user data as given below:



id: User index in user table, for information only.

username: Alias which will be shown in the system.

role: The role defines user authentication level which describes or limits the users’ functionality, please review the section: **“2.3.1 User Roles and Authority Matrix”.**

firstname: User’s first name, this is for information, has nothing to do with authority.

last name: User’s last name, this is for information, has nothing to do with authority.

“Apply” button save changes.

“Add” button provides functionality to add a new user to the system with the data available in the editor.

“Delete”button removes the user from the system.

Password: User’s password, you may edit / change using the **“Edit”** button given in the password field.

Clicking the **“Edit”** button given in the password area will open the password editor popup which contains new password fields. You may apply / undo changes after editing.

Don’t forget to click the apply button on the “users” popup after you change the password in the password editor.

Figure-15. User password editor popup

2.7. Licenses Page

Licenses page provides an interface to manage licenses installed in the host computer. The licenses page contains a short description and tabular view of licenses in the host.

Netbus applications require licenses to be able to run. You may obtain licenses from Okosis company. Please visit this link for details: http://www.okosis.com/license.html

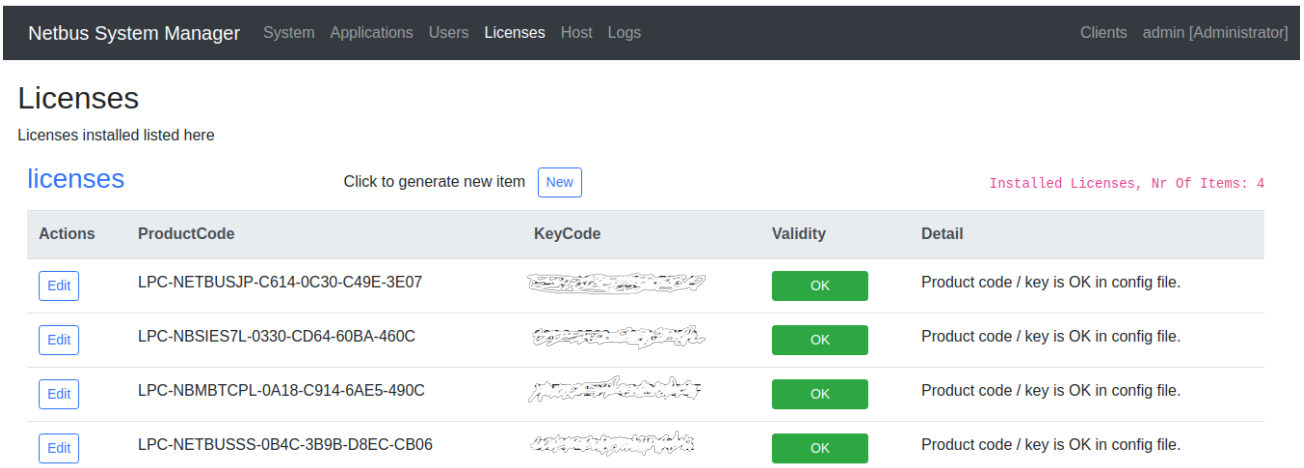


Figure-16. Licenses page

Licenses tabular view contains columns given as below:

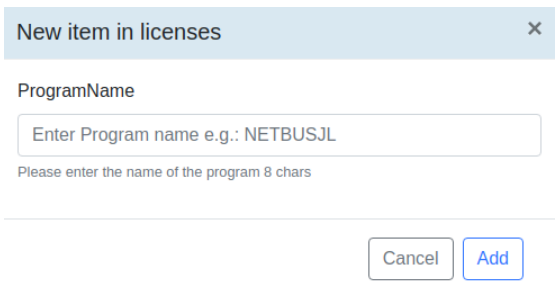
ProductCode: Text code containing Netbus Application's program name and your host specific codes. This text should be submitted to Okosis for acquiring KeyCode for your host.

KeyCode: This text is the specific license generated for your host using the Product code which was generated in your host system and transferred to Okosis company.

Validity: This column shows the validity status of your license. May contain either OK / Invalid. This field is a button, click the button to see the details of the validity status in popup.

Detail: The validity detail of your license is given in this column.

2.7.1.1 Adding a new license



To add a new license to your host please click the **"New"** button on top of the Licenses tabular view. The editor on the left site will popup containing following fields:

ProgramName: Please enter the "Netbus Application" name. It is an eight character text code for the specific application.

Figure-17. New License entry

Sample Netbus Program Names to be entered as new license

List of the sample program names are given below for your information.
For full list of program names please visit link: **http://www.okosis.com/license.html**

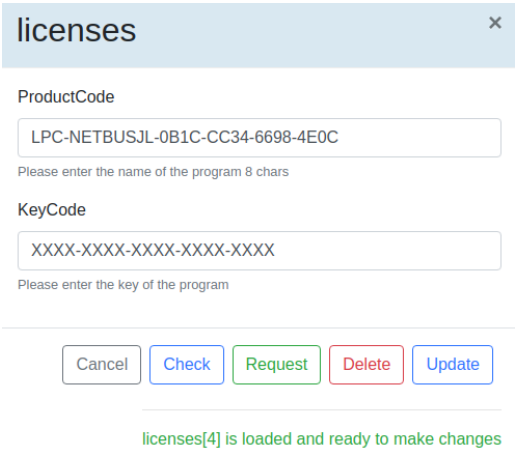
Application	ProgramName
Netbus Pro Core - Lite	NETBUSJL (future)
Netbus Pro Core - Standard	NETBUSJS (future)
Netbus Pro Core - Professional	NETBUSJP
Netbus Storage Server	NETBUSSS
Netbus Siemens S7 Link	NBSIES7L
Netbus Modbus TCP Link	NBMBTCPL

Table 2: Netbus Program Names

The actual version supports the NETBUSJP / NETBUSSS / NBMBTCPL / NBSIES7L application group. The code NETBUSJL and NETBUSJS are registered for future usages.

2.7.1.2 Editing a License

After you enter the new **ProgramName** and click enter, the new license will be shown in the license tabular view with a void license as XXXX-XXXX-XXXX-XXXX-XXXX.



Please click the **"Edit"** button of the license line, **"licenses"** window will popup. Please enter the license code you retrieved from the Okosis in the **KeyCode** text box. Click the **"Update"** button to proceed with the application.

If you don't want to make any changes please click the Cancel button which closes the page.

Figure 18: Licenses editor

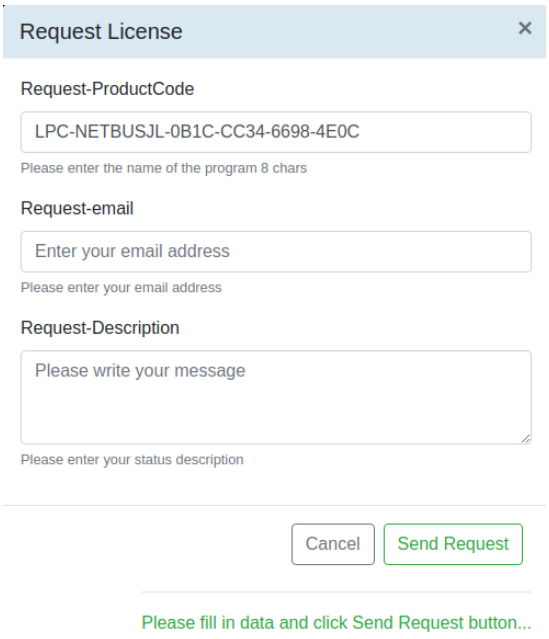
Delete button deletes an invalid license entry. Please be careful on deleting licenses, obviously a deleted license may cause some services to be stopped and interrupted !

If the license is valid, the delete button disappears to prevent deletion of the license.

There are also additional buttons given as "Check" and "Request" in the licenses editor. The explanation of "Requesting Licenses" will be given in the following section.

2.7.1.3 Requesting Licenses

If you don't have a valid license, you may request a license using the licenses popup. To request a license please click the **"Request"** button on the licenses editor. The Request License window will popup. Please enter your e-mail and request description and click the **"Send Request"** button to submit your request to Okosis system.



Your request will be processed soon and a valid license will be generated for you.

Please come back later to the licenses page & click the edit button to edit the license. The licenses popup will be shown as explained in the previous section which contains the "Check" button to accomplish license retrieval from the Okosis system.

Figure 19: Request license popup

2.7.1.4 Checking Request Response

To check the status of license request, please come back to the licenses page and click the relevant line to open up the licenses editor and click the "Check" button to ask the system if a license is generated for your request.

If your request is still processing, a popup showing this status is shown.

If your license is generated by Okosis, a popup showing the generated license appears and if you click the "OK" button, the generated license will be saved to the "KeyCode" area, click "Update" button to apply the received license code. After the application of the new code, the license should turn from red colored Invalid status to green OK status. The Validity of the license and license classes will be explained in the next section.

2.7.1.5 Check License Validity

If the “**KeyCode”** which you entered is valid, then the green **“OK”** button appears in the **“Validity”** column and relevant status is shown in the **“Detail”** column. If the KeyCode you entered in the Item editor is not valid, the **“Validity”** column shows a red colored **“Invalid”** button and an **“Detail”** column shows the relevant info about the situation. You may click the OK / Invalid status button to get detailed status of license registration.

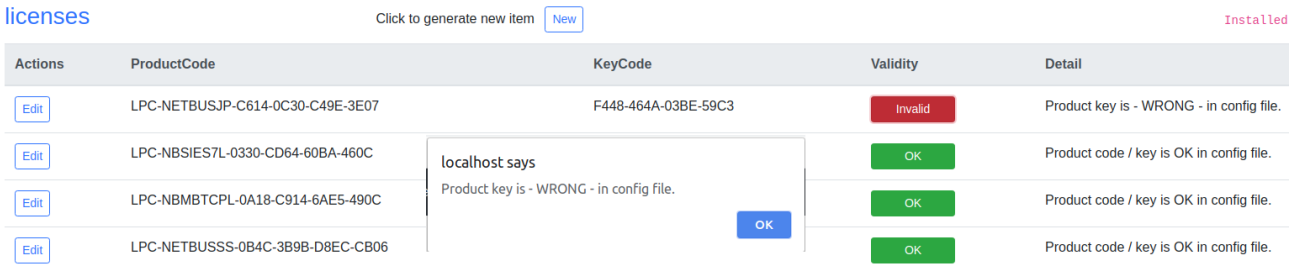


Figure 20: Wrong License and detail popup

2.7.1.6 License Classes

Netbus Pro System has four type of software licenses, details are given below:

License Class	Explanation	Validity
Class A	Premium Class	Unlimited, Upgrades
Class B	Standard Class	Unlimited
Class C	Long Temporary Class	Limited, 30 days valid only
Class D	Short Temporary Class	Limited, 10 days valid only

Table 3: License classes

In the licenses window, the license type, validity duration and status is shown.

A sample license list is given here for your reference

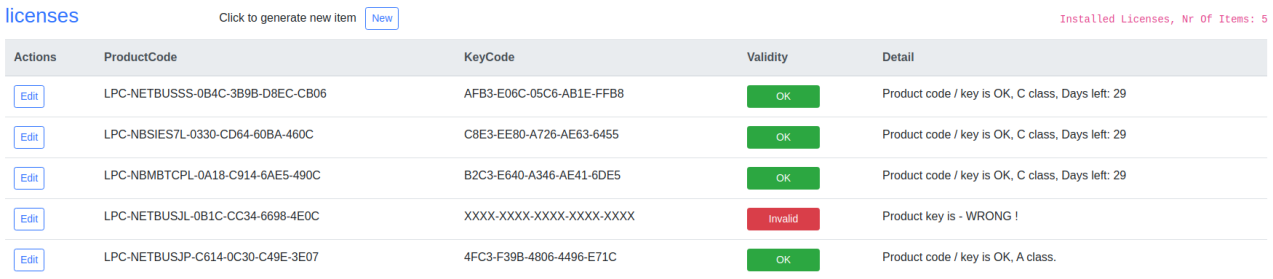


Figure 21: Sample license list

2.8. Host Page

The Host page contains host diagnostics views after giving a short introduction text presents two tab pages named as **“Performance”** and **“Processes”**.

2.8.1.1 Performance Tab

This contains progress bars of CPU and memory load values.
You can check **“Auto update values”** to update the progress bars periodically.

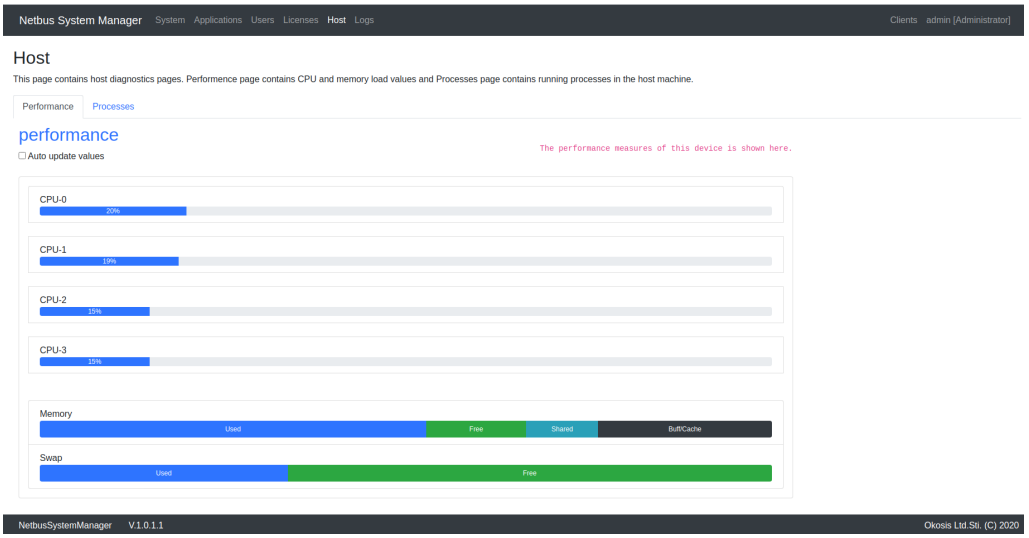


Figure 22: Host page performance view

2.8.1.2 Processes Tab

This tab contains running processes in the host machine. You can kill a process by clicking **“Kill”** button aside as well as enter a **“Filter”** to filter the presented processes.

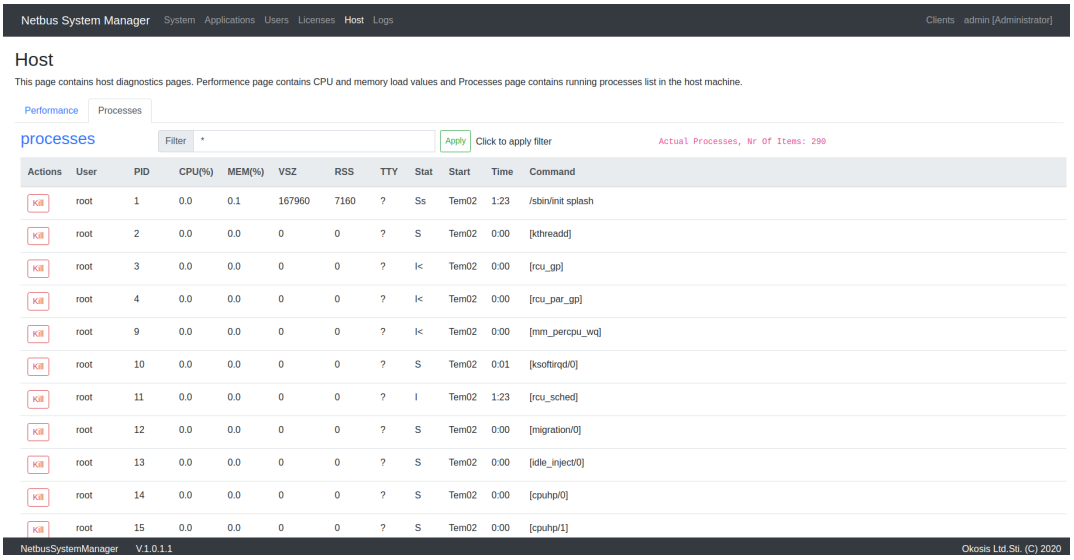


Figure 23: Host page processes view

2.9. Logs Page

The Logs page contains a short introduction text and a tabular view with a dark background showing diagnostic logs of the service.

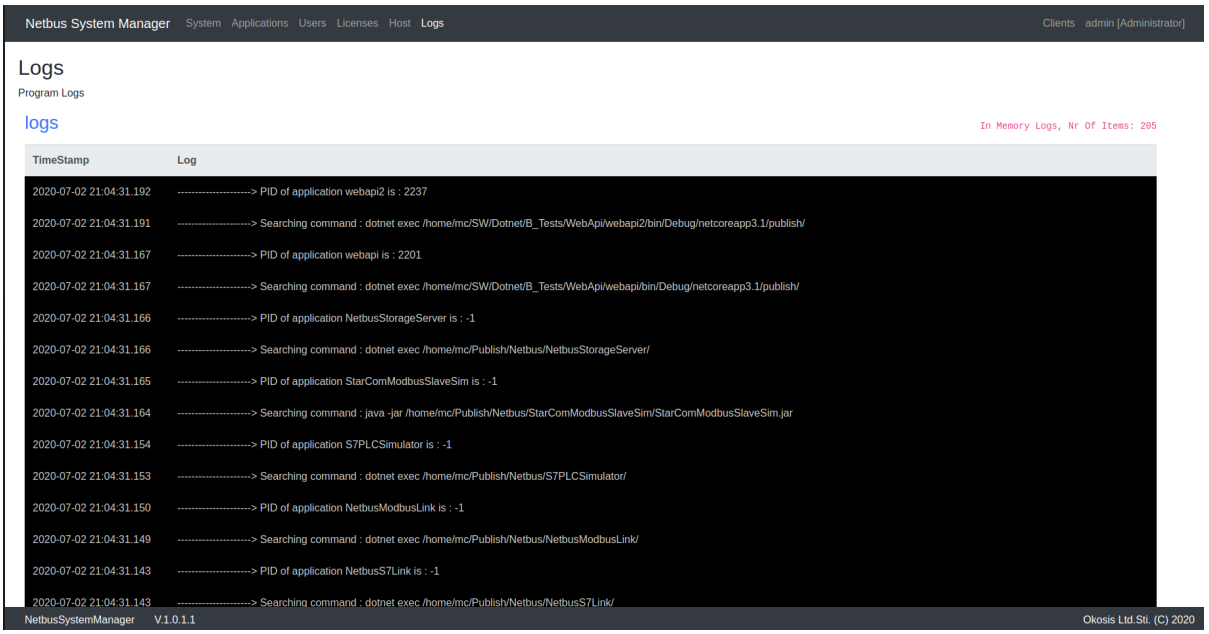


Figure 24: Logs page and tabular logs window

2.10. Clients Page

The Clients page provides an interface to explore “Netbus Clients” on the network. The page contains a short introduction text, an **“IPMask”** field and a tabular view which shows an explored clients list which exists on the same network.

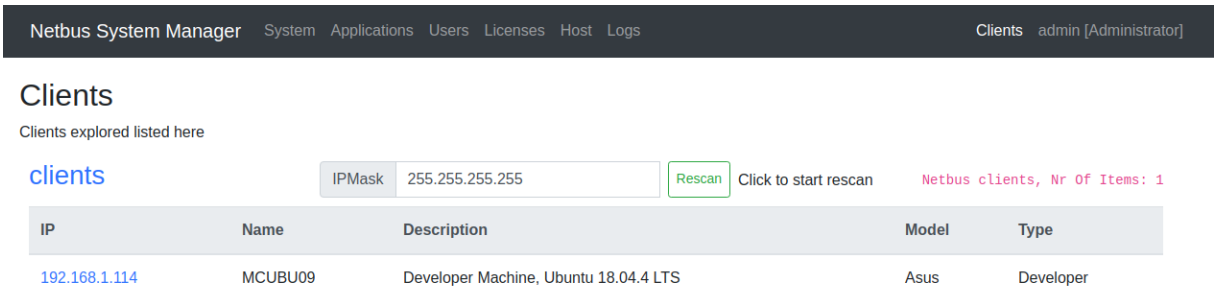


Figure 25: Clients Page

IPMask: This field is the mask applied to the scanning of the network

Rescan: Click this button to restart the scanning operation and explore the clients.

IP: This field contains explored Netbus Client’s IPAddress and a web link to this client. Click this link to access the Netbus System Manager page of this very client device.

Name, Description, Model, Type: These fields are coming from the “SystemID” page of this very client. Please have a look at section: ”**2.4 System Page”**

3. Netbus Pro Core Service

3.1. Netbus Pro Core Service Introduction

Netbus Pro Core Service is the main service in the heart of **Netbus Pro System** which provides **"modern & standardized effective data exchange interfaces"** for data management and presentation functionality.

Netbus Pro Core Services accept data from **"Link Services"** and present it to WebAPI Consumers **"End User Presentation Services"** and **"Custom Services".**
For details of provided WebAPI interfaces and data schema, please review the documentation: "**III. Netbus Pro System - WebAPI Interfaces".**

3.2. Netbus Pro Core Service Functions

Netbus Pro Core service processes several types of values which are acquired from the site and manages them for later usage for Netbus Applications and third party data exchange consumers. The Netbus Pro Core system has different types of strategies for these data types.

"Tag" and **"Parameter"** storage is managed by the **Netbus Pro Core** service itself. The data is stored into the persistent database to a collection named **"Netbus".** Data saved under the collection named as "Tags" and "Parameters" respectively. The collection **"Netbus"** contains the configuration data of the **Netbus System**.

Besides, Netbus Pro Core system contains list of **"Record"s** and **"Event"s** tables which are expected to be transferred by the **Netbus Storage Service** into the DB. **Netbus Storage Server** application saves this data to **"NetbusStorage"** database. Data saved under the collections named as **"Records"** and **"Events"** respectively. The collection **"NetbusStorage"** contains the **"RunTime"** data of the Netbus System.

For details of storage server please see the section **"4 Netbus Storage Service ".**

Netbus Pro Core functions as WebAPI interfaces and supported data types and data management strategies will be given in this section.

3.2.1. Netbus Core Service Redundancy

Netbus Pro System perfectly supports core redundancy. Netbus Pro Core System might be executed separately on the same computer (with different TCP ports) as well as on different computers perfectly. The Netbus data provider services (for example Link services) supports multiple data transmission capabilities for more than one core service simultaneously. In this case more than one core will be available at a time and providers feed the core services simultaneously. The consumer applications check availability of the core servers and choose to connect the preferred live server to get data. In a fail case the consumer could switch to the standby core server and operate without interruption.

3.2.2. Netbus Pro Core Service Data Types

3.2.3. Records

Record is a chunk of data which contains a set of recorded Tags and values in it.

3.2.3.1 Periodically recording of data-source (online – NSS stores to db)

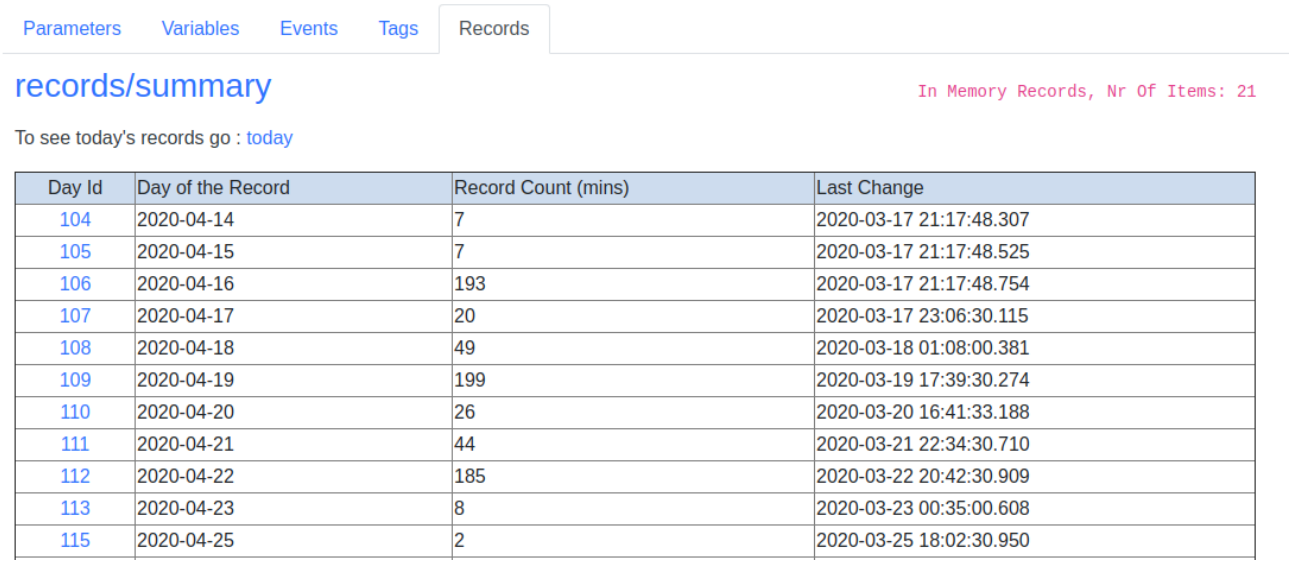
The “data-source” **which will be stored in the system** is described as a **“Record”**.

The **“Record”** represents a day and contains all the data as minutes & values in it. The list of records represents **“the site value data set”** in a definite period of time.

The data is managed online and is stored by the **Netbus Storage Server** to the persistent database named **“NetbusStorage”** as the collection of the “**Records”**.

3.2.3.2 Records Tab Page

Records tab page contains a summary table for all the records which are existing int the **Netbus Pro Core** and a link as **“today”** which redirects to a JSON data presentation export that contains the online data of today.



Day Id	Day of the Record	Record Count (mins)	Last Change
104	2020-04-14	7	2020-03-17 21:17:48.307
105	2020-04-15	7	2020-03-17 21:17:48.525
106	2020-04-16	193	2020-03-17 21:17:48.754
107	2020-04-17	20	2020-03-17 23:06:30.115
108	2020-04-18	49	2020-03-18 01:08:00.381
109	2020-04-19	199	2020-03-19 17:39:30.274
110	2020-04-20	26	2020-03-20 16:41:33.188
111	2020-04-21	44	2020-03-21 22:34:30.710
112	2020-04-22	185	2020-03-22 20:42:30.909
113	2020-04-23	8	2020-03-23 00:35:00.608
115	2020-04-25	2	2020-03-25 18:02:30.950

Figure 26: Records tab page

Records / Summary Table Fields

Day Id : This is the ID of the day, actually it is the day index, counting up starting from 01.01.2020, The Day Id is a link which redirects to a JSON export that contains the online data of that day.

Day of the Record: Text presentation of the day formatted as “yyyy-MM-dd”

Record Count (mins): The number of minutes’ records exists in the system

Last Change: The last change of this very record

3.2.3.3 Record Schema

Netbus Pro Core Application, accepts record values which are acquired from the site and transferred by the link applications through WebAPI data exchange interface.

Netbus Pro Core accepts posted records from the data providers, as the values in a form of JSON data collection as a daily record which is given as a schema of “Record”.

The **Record** data schema and example JSON data view of a day is given below.

Figure 27: Record data schema

3.2.3.4 Record schema fields

Every **“Record”** represents a day and contains data of this day as a list of **“minutes”** in the form of **“RecordMinute”** and a timestamp for changes **“lastChange”.** The day is identified as an integer named **“iDay”** which is the index of the day counts up from the date 01.01.2020.

Every **“RecordMinute”** represents a minute and contains a list of **“tags”** in form of **“RecordTag”** in it and an integer index **“iMinute”** which is the index of the minute from the time 00:00:00 as well as a timestamp for changes **“lastChange”.**

Every **“RecordTag”** contains a list of “**values”** in it in the form of **“RecordValue”** and identified by an integer **“no”** which represents **“tag number”**. The tag number is acquired from Netbus Pro Core at the tag registration phase once. The registered tags and its tag numbers(no) are listed in the **“Tags”** tab.

Every **“RecordValue”** contains **“value”** and **“timeStamp”** fields as string values. The meaningful data is this very structure which is organized in related fields inside the relevant tag, minute and day.

3.2.3.5 Record management mechanism

Netbus Pro Core manages these **“Record”**s collection. The Records collection “**updated and merged”** with the fresh data supplied by the WebAPI data providers.

The new **“RecordMinute”** which is received from providers for any particular day is inserted to the relevant day’s “**minutes”** collection.

The new data which received as list of collection of **“RecordTag”s** from providers for any particular minute is inserted to the relevant minute’s “**tags”** collection. The “**RecordTag”** data is organized as collection of “**RecordValue”**s which contains a string **“value”** and a **“timeStamp”** for it.

If the minute already exists some data, the new **“RecordValue”** data is inserted into the collection of **“values”** of relevant **“RecordTag”** in the relevant minute existing in the relevant day which is given as a **“Record”**.

3.2.3.6 Inserting Record to Netbus Pro Core

To insert a Record to Netbus Pro Core system it is necessary to provide data in following sample JSON format and POST it to : **localhost:5300/api/records**


```
{
    "iDay": 46,
    "minutes": [
        {
            "iMinute": 1265,
            "tags": [
                {
                    "no": 2,
                    "Values": [
                        {
                            "value": "22",
                            "timeStamp": "2/16/20 9:05:00 PM"
                        },
                        {
                            "value": "24",
                            "timeStamp": "2/16/20 9:05:30 PM"
                        }
                    ]
                },
                {
                    "no": 3,
                    "Values": [
                        {
                            "value": "33",
                            "timeStamp": "2/16/20 9:05:46 PM"
                        }
                    ]
                }
            ]
        }
    ]
}
```


“iDay”: The index of the day,
Counted from the date of **01.01.2020**
For example: Record **16.02.2020** is **46th** day.

“iMinute”: The index of minute,
Counted from the minute of **00:00:00**
For example: **21:05** is **1265th** minute.

“no”: This is the tag number, this number is taken at the registration phase once from NetbusProCore.

“Values”: This is the core data which is posted to the system. Please pay attention that it is an array and may contain several values inside.

For effective usage and keep the system load minimum, Netbus Pro system data provider applications (as Netbus Link Services) collects and arranges data, then generates JSON collections for the minute and sends this collection to the system.

Please organize your data and perform effective actions to minimize the system load.

3.2.3.7 Diagnosing and Inserting “Record”s using auxiliary utility tools

While Netbus Pro Core provides standardized WebAPI interfaces, it is perfectly possible to post / get JSON data to Netbus Pro system using auxiliary utility tools.

Here is an example of Postman GET for today: **localhost:5300/api/records/today**

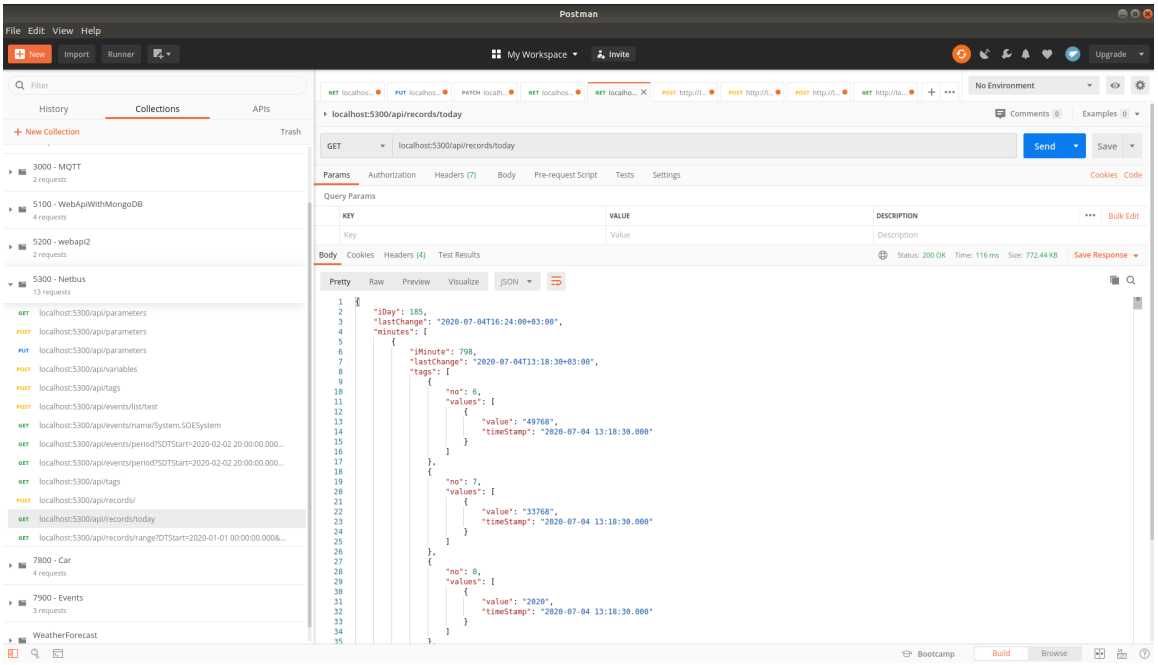


Figure 28: Acquiring data from Netbus Pro Core using Postman utility

Here is an example of Postman POST for today: localhost:5300/api/records

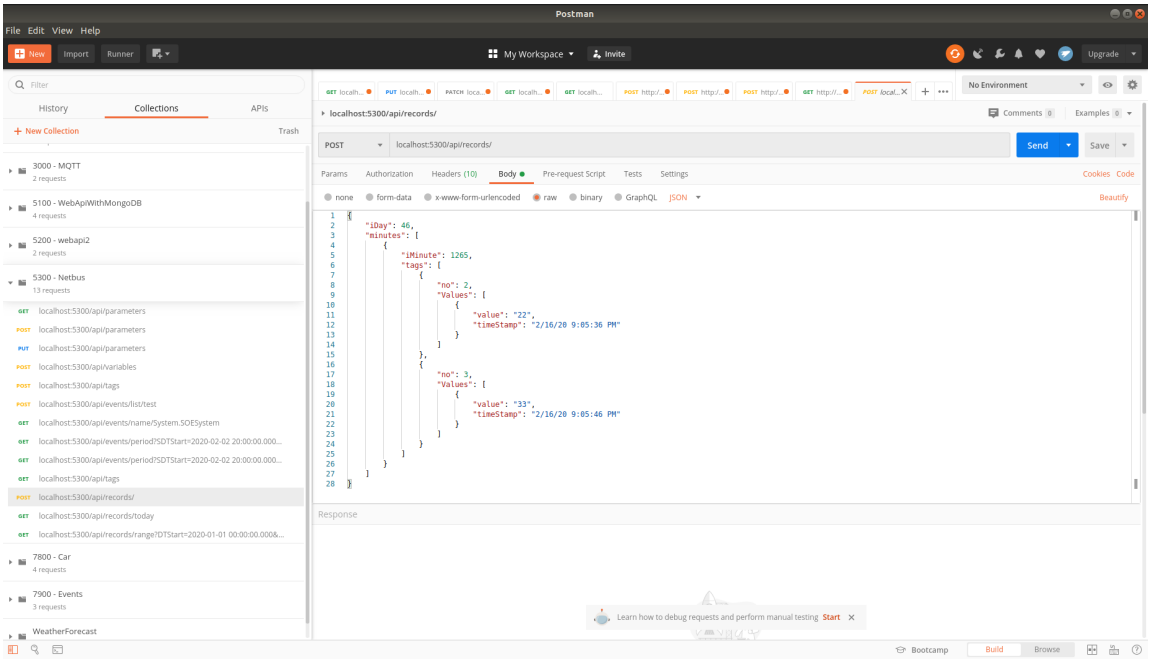


Figure 29: Posting data to Netbus Pro Core using Postman utility

3.2.4. Events

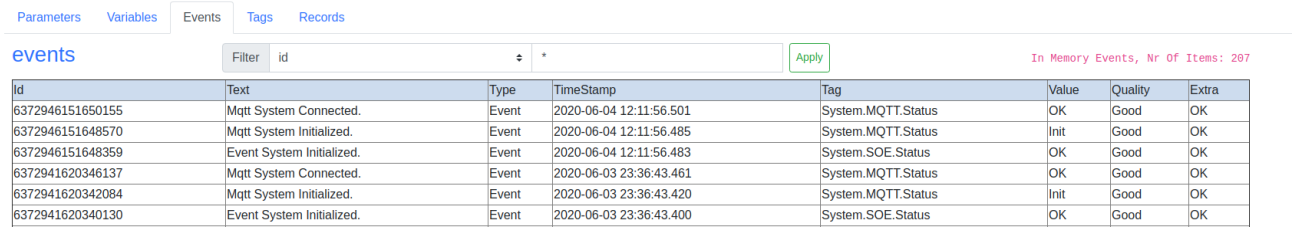
Event is the chronological record of the value change of the data-source.

3.2.4.1 Chronological changes log of data-sources (online – NSS stores to db)

The change log of a data-source **which is planned to be stored in the system** is described as an **"Event".**

The **"Event"** is a change log which is generated by the provider system and transferred to the Netbus Pro Core via WebAPI interface.

The Event table is stored to the **"NetbusStorage"** database by the Netbus Storage Server application and reloaded on the restart of the Netbus Pro Core application as the collection of the "**Events"**.


Parameters Variables Events Tags Records

events Filter id * Apply In Memory Events, Nr Of Items: 207

Id	Text	Type	TimeStamp	Tag	Value	Quality	Extra
6372946151650155	Mqtt System Connected.	Event	2020-06-04 12:11:56.501	System.MQTT.Status	OK	Good	OK
6372946151648570	Mqtt System Initialized.	Event	2020-06-04 12:11:56.485	System.MQTT.Status	Init	Good	OK
6372946151648359	Event System Initialized.	Event	2020-06-04 12:11:56.483	System.SOE.Status	OK	Good	OK
6372941620346137	Mqtt System Connected.	Event	2020-06-03 23:36:43.461	System.MQTT.Status	OK	Good	OK
6372941620342084	Mqtt System Initialized.	Event	2020-06-03 23:36:43.420	System.MQTT.Status	Init	Good	OK
6372941620340130	Event System Initialized.	Event	2020-06-03 23:36:43.400	System.SOE.Status	OK	Good	OK

Figure 30: Events table

3.2.4.2 Event Schema and Field's Explanations


Event ⌄ {
id integer($int64)
text string
nullable: true
timeStamp string($date-time)
eType EventType integer($int32)
Enum:
› Array [3]
tag string
nullable: true
value string
nullable: true
quality string
nullable: true
extra string
nullable: true
}

Figure 31: Event schema

id: Auto generated by system

no: This long integer is generated by the Netbus Pro Core system at the configuration phase and used as a reference for recording

name: The name of the tag

description: The text description of the tag

location: The tag related event change

client: The last value of the tag

extra: Extra information for change, string

3.2.5. Tags

“Tag” is a registered data-source to Netbus Pro System by any service to establish and manage the data integration and the data link between the service and the system.

3.2.5.1 Registered data-source for recording purpose(persistent in Netbus db)

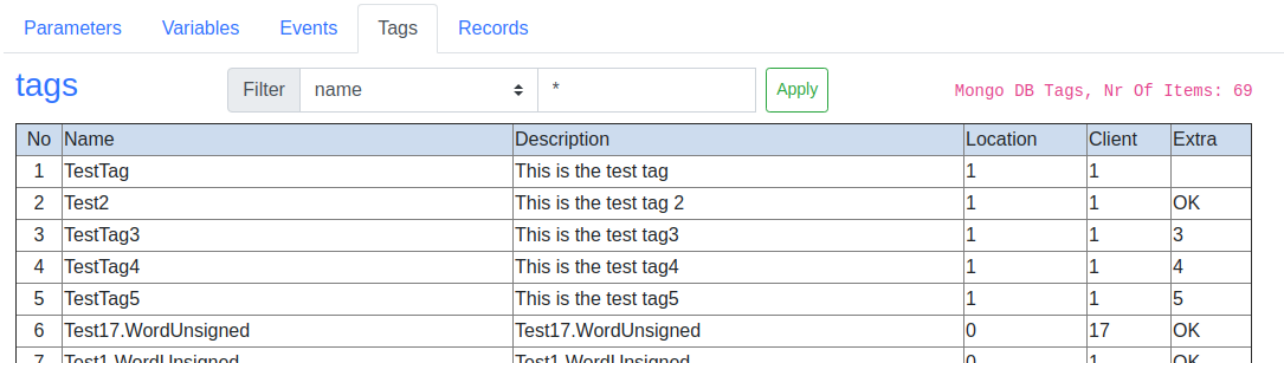
The “data-source” **which is planned to be recorded** is registered to the Netbus Pro Core application in configuration phase and described as a **“Tag”** and stored in database persistently on this very configuration phase.

The registration process is done once and the system generates a unique integer tag number named as “**no”** which is unique and assigned as an ID to this particular tag. This ID is used as a reference to this particular tag for all the **“recording”** works.

Registration of Tags

Services have to register their **“Tags”** to Netbus Pro System before using them as a variable and transmitting them to the Netbus Pro Recording system.

Tag registration is done on the startup phase by every service. The service introduces the tag name to the **Netbus Pro Service** and gets back the tag number **“no”** for every tag. If a tag is already registered to the system, the already assigned **“no”** is sent back, otherwise a new **“no”** is generated and sent back to the requesting service.



No	Name	Description	Location	Client	Extra
1	TestTag	This is the test tag	1	1	
2	Test2	This is the test tag 2	1	1	OK
3	TestTag3	This is the test tag3	1	1	3
4	TestTag4	This is the test tag4	1	1	4
5	TestTag5	This is the test tag5	1	1	5
6	Test17.WordUnsigned	Test17.WordUnsigned	0	17	OK
7	Test1.WordUnsigned	Test1.WordUnsigned	0	1	OK

Figure 32: Tags Table

3.2.5.2 Tag Schema and Field’s Explanations

```
Tag ⌄ {
    id                  string
    no                  integer($int64)
    name                string
                        nullable: true
    description         string
                        nullable: true
    location            integer($int32)
    client              integer($int32)
    extra               string
                        nullable: true
}
```

id: Auto generated by system

no: This integer is generated by the Netbus Pro Core system on the configuration phase and used as an identification reference for recording

name: The name of the tag

description: The textual description of the tag

location: The tag related event change

client: The last value of the tag

extra: Extra information for change, string

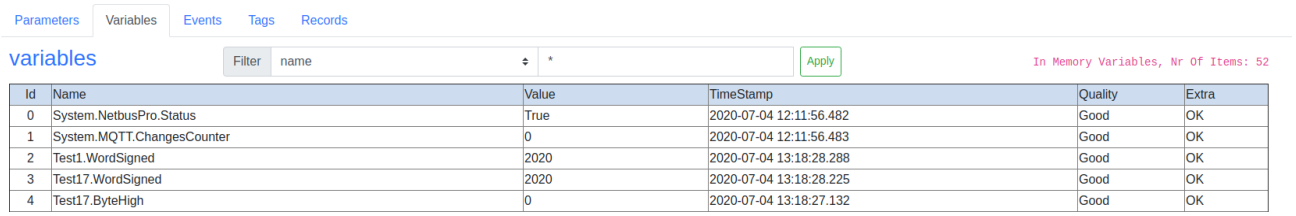
Figure 33: Tag schema

3.2.6. Variables

Variable is an online temporary value used as a data-exchange point between services.

3.2.6.1 Actual value of data-source (online - non stored)

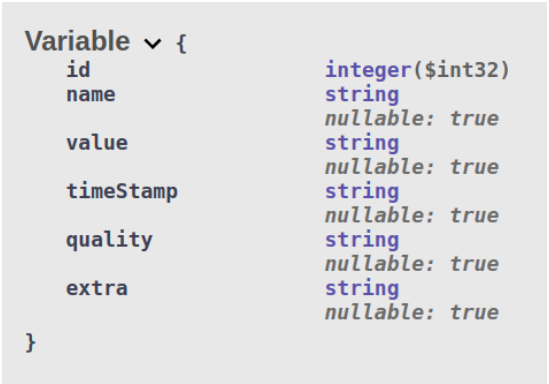
The actual value of a data-source is described as a **“Variable”.**
The **“Variable”** kept as an online actual value and presented via WebAPI interface.
At the start of the Netbus Pro Core application, the variables table is an empty table.
Netbus Pro Core Variable table contains the variable values posted from providers. Variable table is a place where all actual values are kept online as **“a single last value”**.



Id	Name	Value	TimeStamp	Quality	Extra
0	System.NetbusPro.Status	True	2020-07-04 12:11:56.482	Good	OK
1	System.MQTT.ChangesCounter	0	2020-07-04 12:11:56.483	Good	OK
2	Test1.WordSigned	2020	2020-07-04 13:18:28.288	Good	OK
3	Test17.WordSigned	2020	2020-07-04 13:18:28.225	Good	OK
4	Test17.ByteHigh	0	2020-07-04 13:18:27.132	Good	OK

Figure 34: Variables table

3.2.6.2 Variable Schema and Field’s Explanations



id: The variable identification, auto generated by system

name: The name of the variable

value: The value of the variable

timeStamp: The timestamp of variable change

quality: The last quality of the tag (OPC quality)

extra: Extra information for change, string

Figure 35: Variable schema

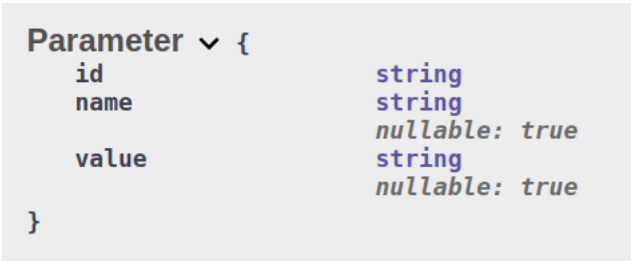
3.2.7. Parameters

3.2.7.1 Configuration parameters (persistent in Netbus db)

The configuration parameters of the system are described as a **“Parameter”.**
The **“Parameter”** is saved in database persistence and presented via WebAPI interface.
Netbus Pro Core Parameter table contains the configuration parameters which can be inspected and changed using WebAPI interface.

Figure 36: Parameters table

3.2.7.2 Parameter Schema and Field’s Explanations



id: Identification, auto generated by the system

name: The name of the parameter

value: The value of the parameter

Figure 37: Parameter schema

3.3. Netbus Pro Core Web Interface

3.3.1. Index Page

Index page contains short introduction text, informative link area and tab page containing links for Internal functionality, Netbus Applications etc.

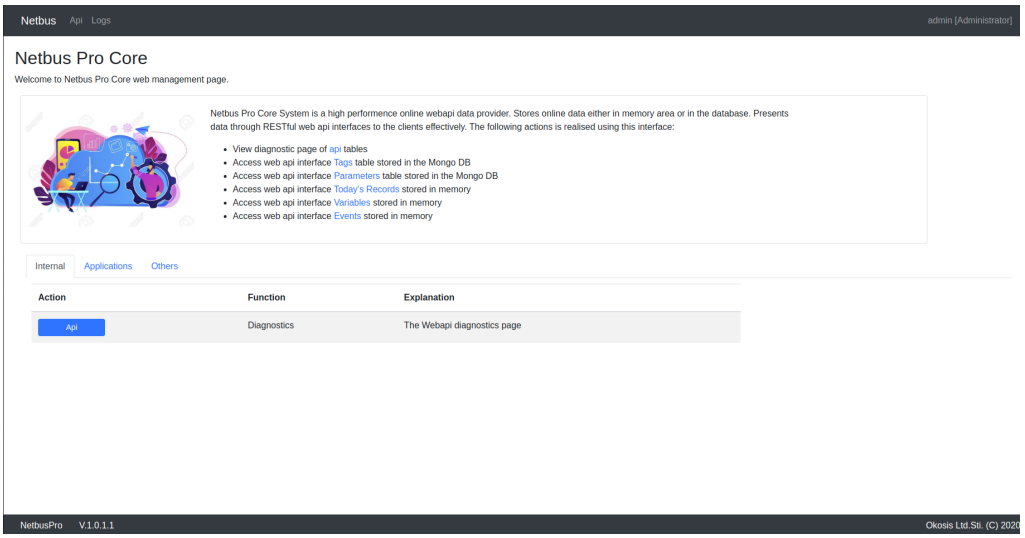


Figure 38: Netbus Pro Core Service Index Page

3.3.1.1 The upper link area

The link area contains short explanations and links for internal functionality, API page and its internal tabs as Tags, Parameters etc.

3.3.1.2 Tab group

Tab group contains **“Internal”**, **“Applications”** and **“Others”** tabs.

Internal tab contains links as follows:
API: Link to the API page, details given in section **“3.3.2. API Page”.**

Application tab contains links for Netbus Pro System applications as:
Netbus System Manager, Netbus Pro Core, Netbus Storage & Link servers.

The “Others” tab contains **Logs, Login and Swagger** links.
These pages explained at

3.3.1.3 Swagger

Swagger is a utility for exploring and testing WebAPI interfaces supported by services. For details of interfaces and swagger please review the documentation of WebAPI: “**III. Netbus Pro System - WebAPI Interfaces”.**

You may get the usage information from the web site of swagger :

https://swagger.io/tools/swagger-ui/

3.3.2. API Page

Netbus Pro Core's API page contains tab pages and tabular views to visualize data.

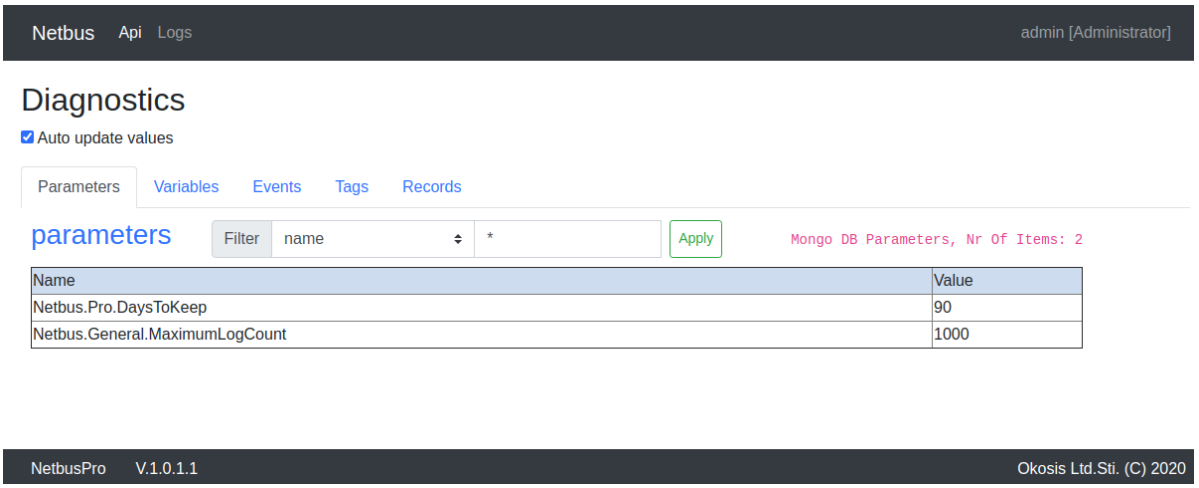


Figure 39: Netbus Pro API Page - WebAPI diagnostics view

3.3.2.1 Auto Update

There is a check box on top of the tab pages to enable updating values periodically. Update period is **3s** and the data fetched from Netbus Pro Service's WebAPI interface and all the tab pages are updated with this data.

3.3.2.2 Filter

There is a filter group on top of tabs to provide filtering functionality on displayed data. The filter contains a selection box for the selection of the field to be applied and a text box for the search string. The "*" string shows all the data and enters any filter string searched in the selected field and items containing this very text string are filtered and displayed in the tabular view after you click the green Apply button.

Filter box exists on Parameters, Variables, Events & Tags tab pages.Logs Page

Logs page contains Netbus Pro Service application logs represented in tabular view. Logs of the application are shown as a snapshot and don't update periodically.

Netbus Applications has a definite limit which is given in the "Parameters" table as ***Netbus.General.MaximumLogCount*** and removes the logs older than this limit.

3.3.3. Logs Page

The **"Logs page"** contains application logs of the service represented in tabular view.

While the netbus pages are prepared with the same principles, all the functionality will not be explained again. Please review the section **"2.9.Logs Page"** for details.

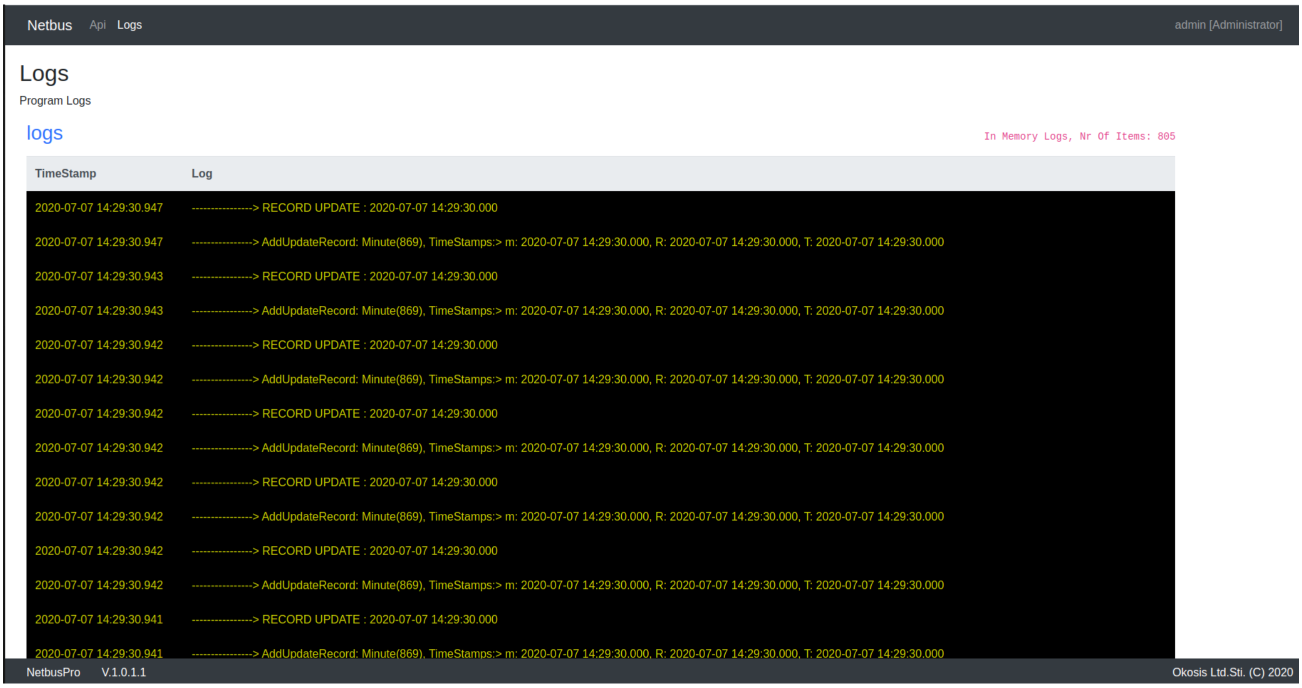


Figure 40: Netbus Pro Service Logs Page

3.3.4. Login Page

Login link exists on the upper right corner of the page which redirects to the unique login page of the service.

There is a unique **"Login Page"** which is valid for all the Netbus Pro System services. The login / logout action which is performed successfully in the Login page of the service will be valid for all Netbus Pro System applications as well.

For more details on Logging in please see section **"2.3.Login Page"**

4. Netbus Storage Service

4.1. Netbus Storage Service Introduction

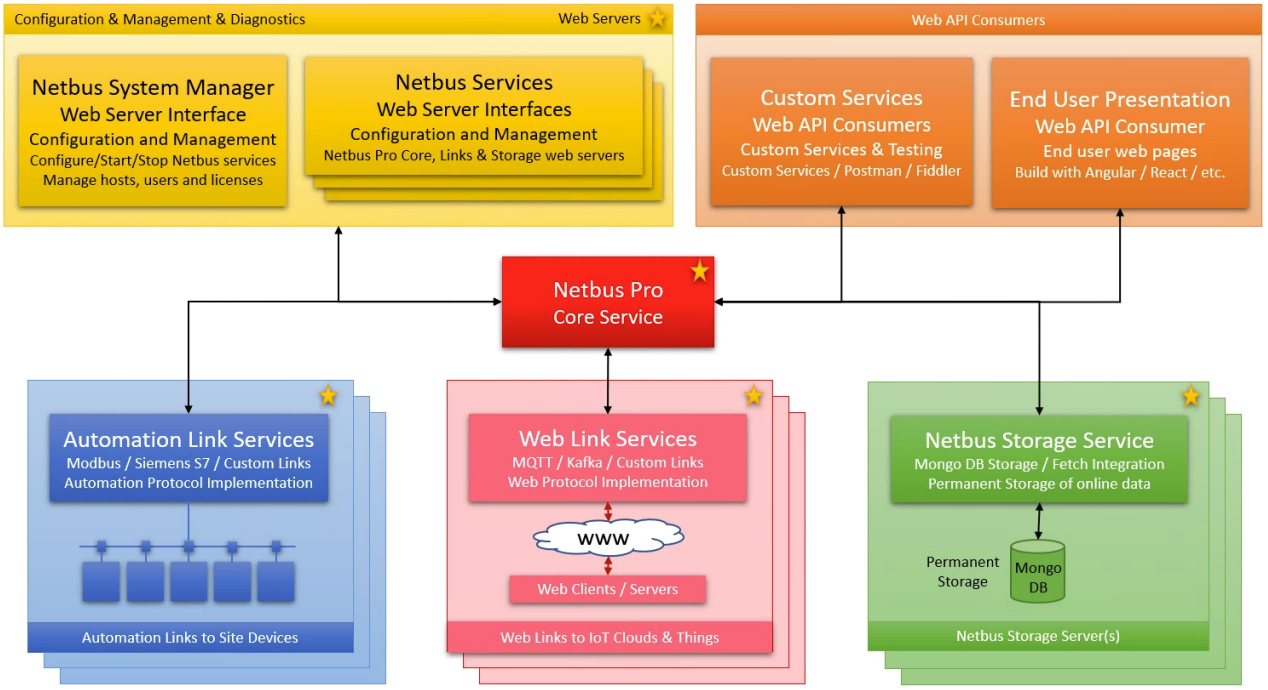


Figure 41: Netbus Pro Core Service

Netbus Storage Server application acquires data from **Netbus Core Application** to store its online data into the persistent storage media and reload it back on startup.

Netbus Storage Server establishes WebAPI data-exchange connection with the **Netbus Pro Core application** to be able to perform the storage server functionality.

4.2. Netbus Storage Server Functions

4.2.1.1 Retrieve Netbus Core application's online data to store it permanently

Netbus Storage Server establishes WebAPI link with **Netbus Pro Core service,** performs periodic queries for the new data entries since the **"lastChange"** which registered at the latest storage operation**.** When it detects new data inserted into the Netbus Pro Core service, it acquires this data and saves it to the persistent database.

- **"Tag"** and **"Parameter"** storage is managed by the **Netbus Pro Core** service itself.
- The data is stored into the persistent database to a collection named **"Netbus".**
- Data saved under the collection named as "Tags" and "Parameters" respectively.
- The collection **"Netbus"** contains the configuration data of the **Netbus System**.
- For detailed information about Netbus Pro Core elements please review the section: "**3. Netbus Pro Core Service".**

Besides, the Netbus Pro Core system contains a list of **"Record"s** and **"Event"s** tables which are expected to be transferred by the **Netbus Storage Service** into the DB. **The Netbus Storage Server** application saves this data to the **"NetbusStorage"** database. Data saved under the collections named as **"Records"** and **"Events"** respectively. The collection **"NetbusStorage"** contains the **"RunTime"** data of the Netbus System.

4.2.1.2 Reload previously stored data (on startup phase) to the core service.

Netbus Storage Server establishes WebAPI link with **Netbus Pro Core service,** on the startup phase, it performs an initial query and checks if Netbus Pro Core has **"the required depth"** of data in its data set. If there is some missing data, then the **Netbus Storage Server** reloads the missing data back to the **Netbus Pro Core**.

The required depth of data is determined by the parameter ***Netbus.Pro.DaysToKeep.***

4.2.1.3 Netbus Storage Server Redundancy

Netbus Pro System may contain more than one **Netbus Storage Server(s)** configured in the system specification. **Netbus Pro System** will work perfectly in this system configuration as well. In this case, all the servers will connect to the same **Netbus Pro Core** service to acquire / transmit its persistent data.

In storing direction, each Netbus Storage Server copies the online data of **Netbus Pro Core** system to itself perfectly thanks to WebAPI infrastructure specification.

In reloading direction, thanks to the effective Record management mechanism of the **Netbus Pro Core** service, all the **Netbus Storage Server(s)** may load their persistent data. if the relevant data is missing there, it will be loaded "**once"** while only the missing values are transferred to the online data set of **Netbus Pro Core** service.

4.3. Netbus Storage Web Server

4.3.1. Index Page

Netbus Storage Server Index page presents links and explanation for the pages status, configuration and logs. The following pictures show the page before/after login.

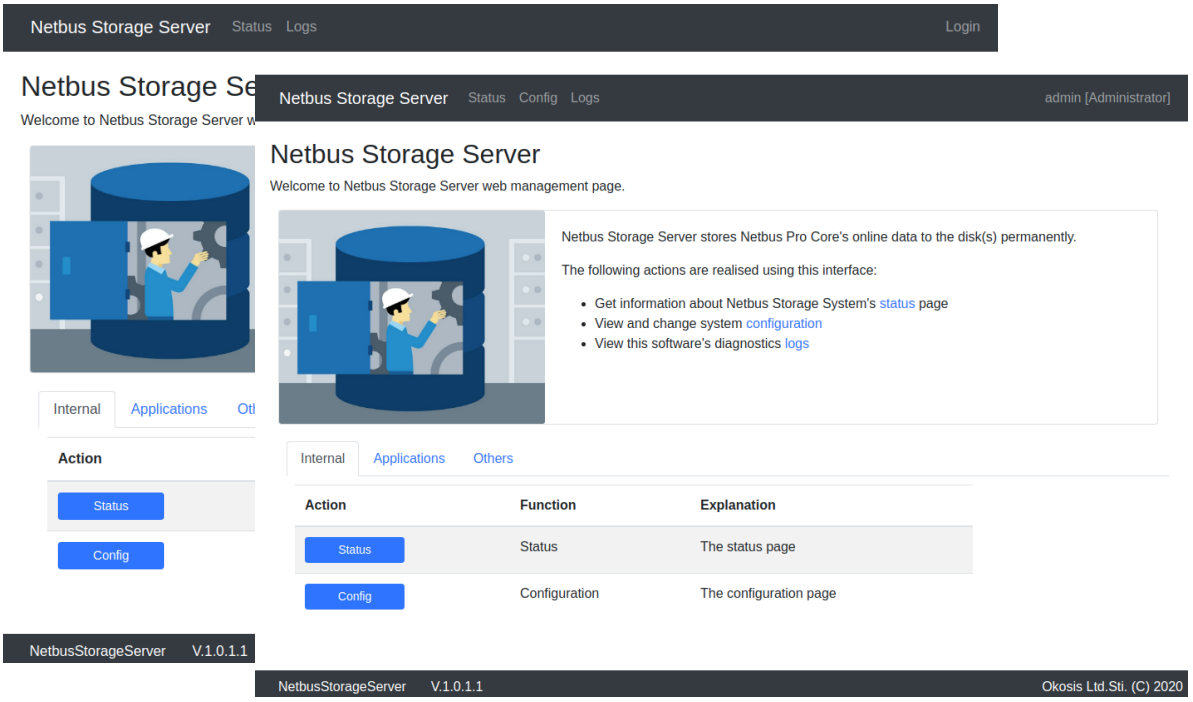


Figure 42: Netbus Storage Server Index page before and after login

The **"Config"** menu item represents the configuration page of the Netbus Storage Server and it is available only for the users which login with **"Administrator"** authorization.

The above menu bar is the presentation of a non-logged-in situation and below represents the menu bar after the successful login operation.

While the netbus pages are prepared with the same principles, functionality of the index page will not be explained again. Please review **"2.2.Index Page"** for details.

4.3.2. Status Page

Status page presents the actual status of the service with the following tab pages:

4.3.2.1 Chart View

The Chart View page represents the number of records which exist in the Netbus services. The diagram represents Netbus Storage Server's data beside Netbus Pro Core's data.

On the Y axis of the chart, record count, on the X axis date-time (days) are given. Please move the mouse cursor on any record to view detailed information about it.

Figure 43: Chart view of Netbus Pro Core data beside Netbus Storage data

4.3.2.2 Tabular View

Tabular View page represents the number of records which exist in the Netbus services. The table contains Netbus Storage Server's data beside Netbus Pro Core's data.

On top of the page there are links for JSON data exports from Netbus Pro Core as:

records / summary: All records

today: today's records

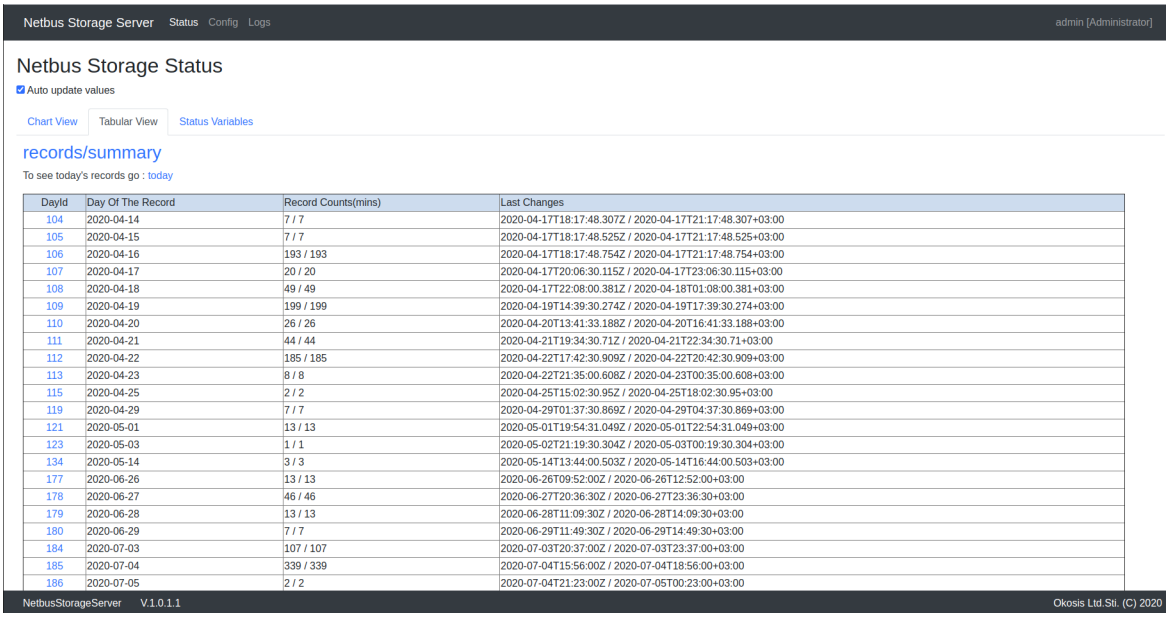


Figure 43: Chart view of Netbus Pro Core data beside Netbus Storage data

The fields of the on the table and explanations are given below:

DayId: This is the index of the day starting from 01.01.2020
This field contains a link for the relevant day data export as JSON.

Day Of The Record: The presentation of the date as "yyyy-MM-dd"

Record Counts(mins): The record count of minutes : storage / core

Last Changes: Last change timestamps: storage / core

4.3.2.3 Status Variables

In the status page of **Netbus Storage Server** there is **“Status Variables”** tab page which represents actual status of the service as given below:

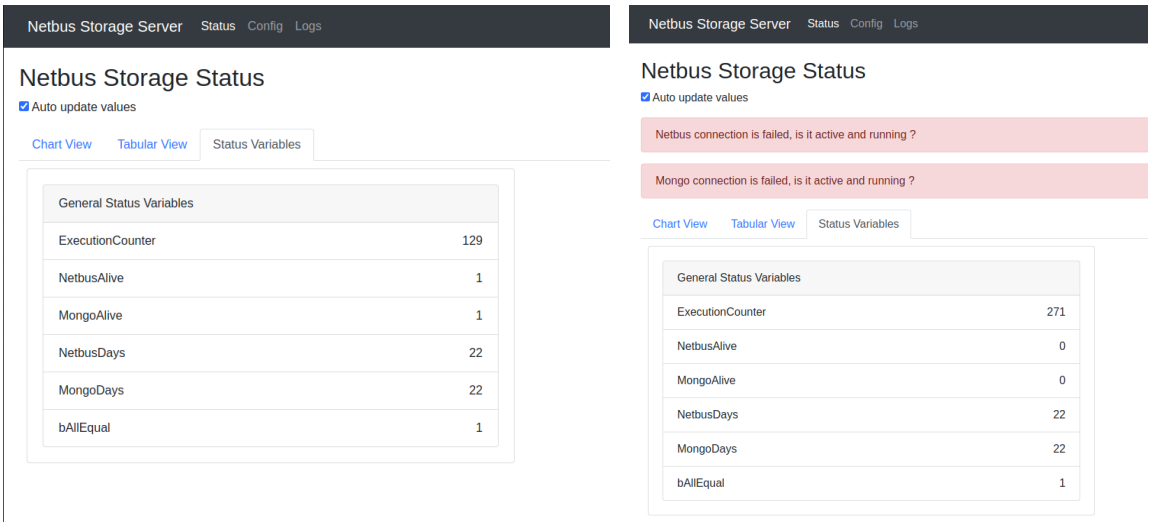


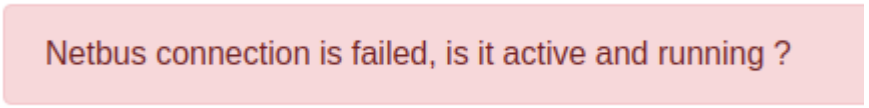
Figure 44: Status Variables page and fail status banners

The fields of the Status Variables tab page contains following lists:

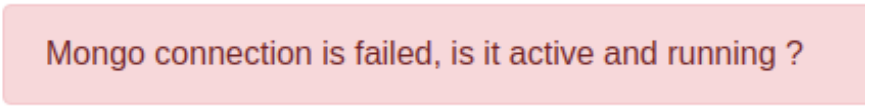
4.3.2.4 General Status Variables

ExecutionCounter: This is a counter value that represents executions of the service. It counts up every execution of Netbus Storage Server.

NetbusAlive: This variable indicates if Netbus Pro Core system is alive (1) or not (0). If Netbus is not alive, the following banner pops up to indicate the failure.



MongoAlive: This variable indicates if mongodb is available (1) or not (0). If mongodb is not alive, the following banner pops up to indicate the failure.



NetbusDays: This variable indicates the days exists in the Netbus Pro Core service

MongoDays: This variable indicates the days exists in the Netbus Storage service

bAllEqual: This variable indicates if core data and storage data is equal (1) or not (0)

4.3.3. Configuration Page

The **“Configuration page”** contains configuration parameters of the service.

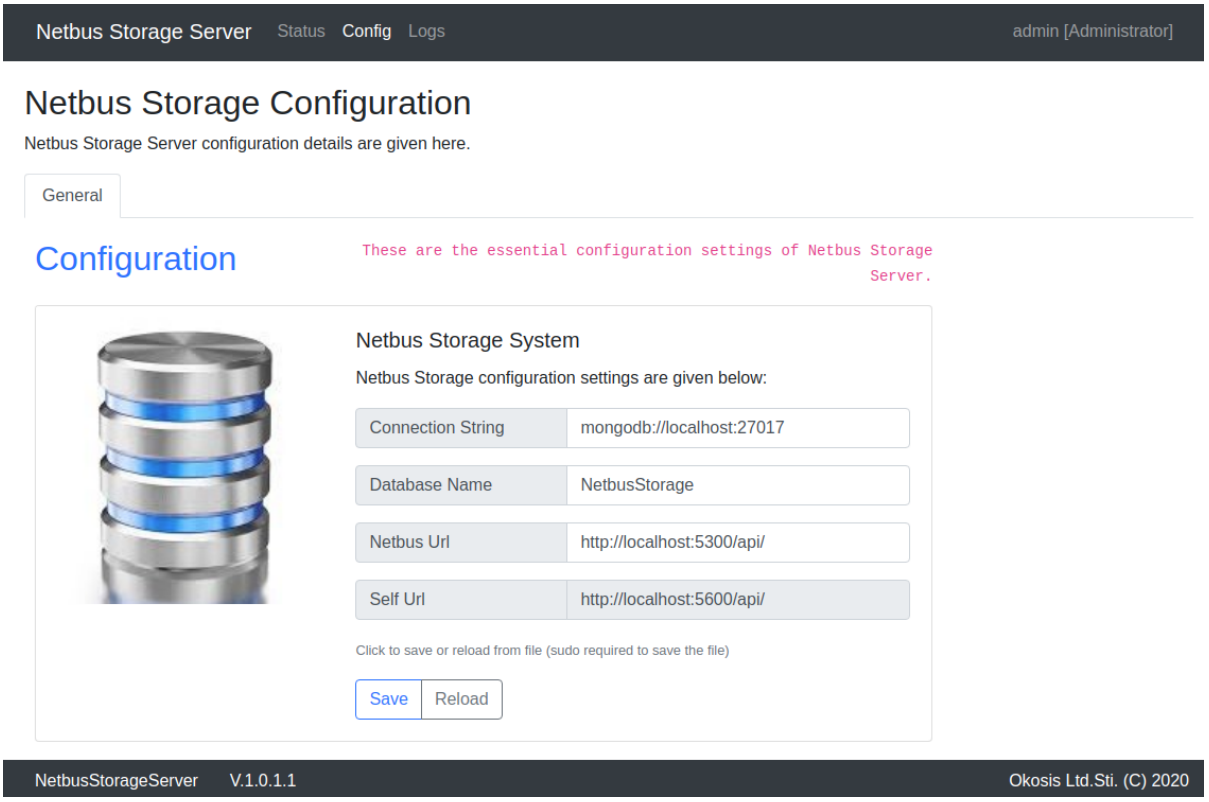


Figure 45: Status Variables page and fail status banners

4.3.3.1 The “Configuration parameters” given as follows:

Connection String : **mongodb://localhost:27017,** please do not change it !

Database Name : **”NetbusStorage”**, please do not change it !

Netbus Url : This is the Url of the **Netbus Core Application** is running at.

Required Action:

Please enter the Netbus Pro Core service url which you want to as the core service of your Netbus Pro System. The format of the text is important, please put the braces respectively, for example : http://localhost/5300/api/

Self Url : This text is read only, there is nothing to do with it.

Self url contains “localhost:” and port number, for example : **http://localhost:5600/api/**
The port number can be edited in the **Netbus System Manager** application web site, Please review at the section **“2.5 Applications Page”** for the details.

4.3.4. Logs Page

The **“Logs page”** contains application logs of the service represented in tabular view.

While the netbus pages are prepared with the same principles, all the functionality will not be explained again. Please review the section **“2.9.Logs Page”** for details.

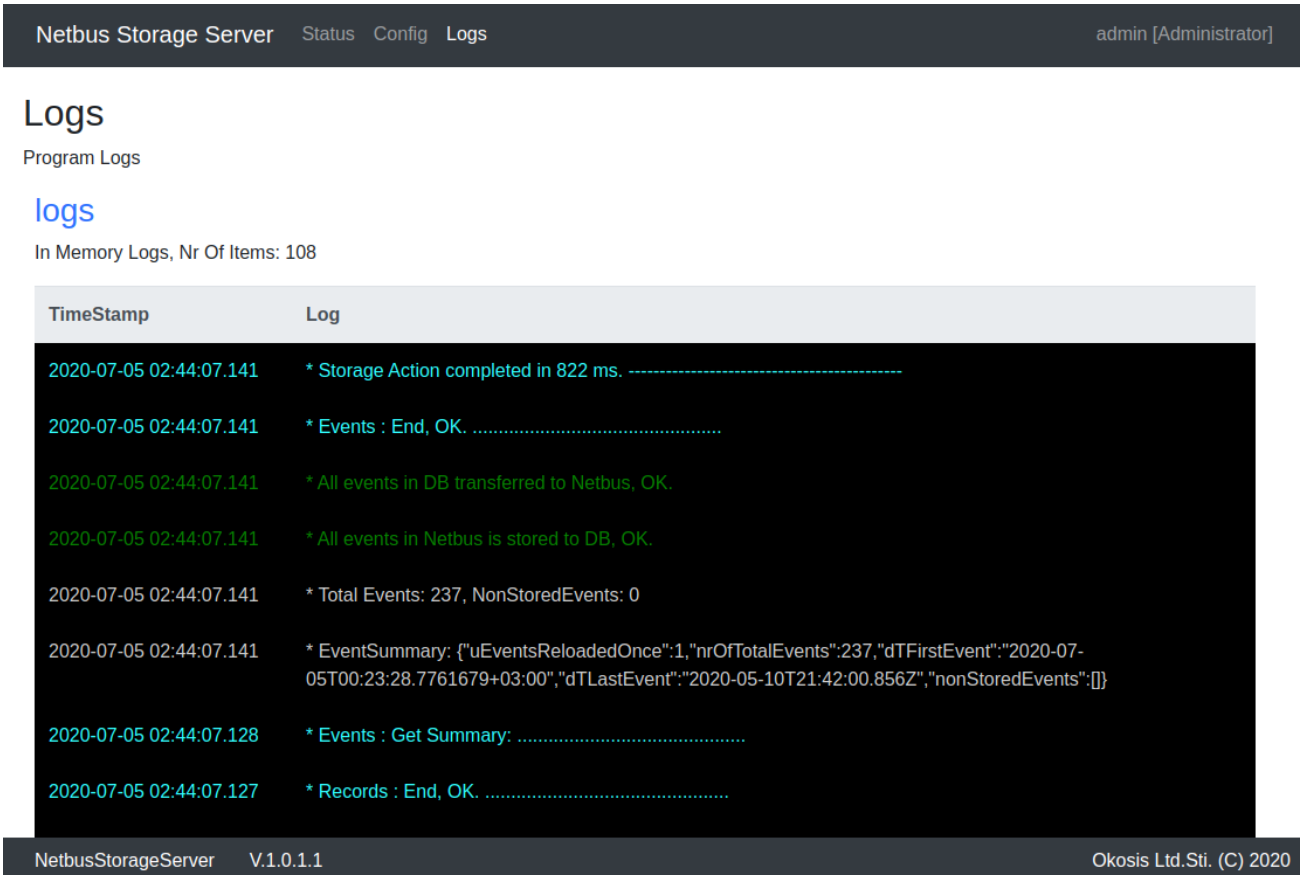


Figure 45: Netbus Storage Server Logs page

4.3.5. Login Page

Login link exists on the upper right corner of the page which redirects to the unique login page of the service.

There is a unique **“Login Page”** which is valid for all the Netbus Pro System services. The login / logout action which is performed successfully in the Login page of the service will be valid for all Netbus Pro System applications as well.

For more details on Logging in please see section **“2.3.Login Page”**

5. Netbus Pro Links Services

5.1. Introduction to Link Services

Netbus Pro System Link Services performs data exchange functionality with the slave devices which have standard protocol communication capabilities to establish data **"link"** and integration with Netbus Pro System.

5.2. Link Services' Functions

Implements communication protocol interface of the slave device set as a background service and perform communication actions as queries and commands to acquire slave data and send commands to it.

Transmit the acquired and collected data to the **Netbus Pro System** using data integration interfaces of the system as WebAPI, MQTT, Kafka, etc.

Besides, **The link services are full fledged web servers** and achieve several functionality through this web interface. The management of the service is realized through the web pages which presents Graphical User Interfaces (GUI) for configuration, diagnostics and log inspection of the service

"The service with a GUI" accepted as an "Application" of Netbus Pro System :

Netbus Pro Service + Web Server = Netbus Pro Application

Netbus Pro contains set of services given as :

Netbus Pro Core Service : Core application, data management & presentation

Netbus Storage Service : Storage application, data storage and fetch servers

Automation Link Services : Links to Automation Devices

Web Link Services : Links to Web Systems

Configuration & Management & Diagnostics : Web server interfaces

Web API Consumers : Custom Services and End User Presentation services

"Automation Link Services" and "Web Links Services" given in the above context are the link services of Netbus Pro System.

"Automation" devices referred as Programmable Logic Controllers (PLC), Remote Terminal Units (RTU) and Data Acquisition devices which capable of communication with standardized conventional protocols as **Modbus, Profibus, EtherCAT etc. besides data integration protocols as OPC**

On the other hand, "Web Link Services" referred here for integration of devices "especially Internet Of Things (IoT) devices and Industry 4.0 systems which are capable of communication with new age IoT protocols **WebAPI, AMQP, DDS, MQTT, Kafka etc.**

5.3. Up Links Introduction

The “Up links” concept defines the interface of a Netbus Link Service with the upper systems like Netbus Pro Core Application and similar systems.

The Up Links schema contains following fields:

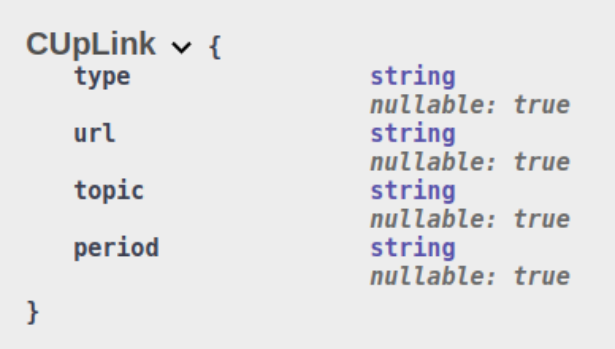


Figure 46: Up link schema

The explanation of the fields are given as follows:

type	Url	topic
mqtt	tcp://localhost:1883	IOChange
webapi	http://localhost:5300/api/variables	Variable
webapi	http://localhost:5300/api/records	record.P
webapi	http://localhost:5300/api/tags	Tags

Table 4: Up Links Table

The explanation of the fields are given as follows:

type: Type of interface which given as one of the :

“mqtt” : MQTT interface

“webapi” : WebAPI interface

url: The url of the upper system for example:

topic: the topic detail for the upper system, Netbus Pro System has following topics:

for type “mqtt”:

“IOChange” : This is the mqtt interface

for type “webapi”:

“Variable” : Variable interface topic for **“Variable”**s

“Record.P”: Record interface topic for **“Record”**s

“Tag”: Tag definition interface topic for **“Tag”**s

period: Execution period of action

5.4. Netbus Modbus TCP Link Service

Modbus TCP Link Services performs data exchange functionality with the slave devices which have Modbus TCP communication capabilities to establish data **“link”** and integration with Netbus Pro System.

5.4.1. Netbus Modbus TCP Link Web Server

5.4.1.1 Index Page

Index page contains short introduction text, informative link area and tab page containing links for Internal functionality, Netbus Applications etc.

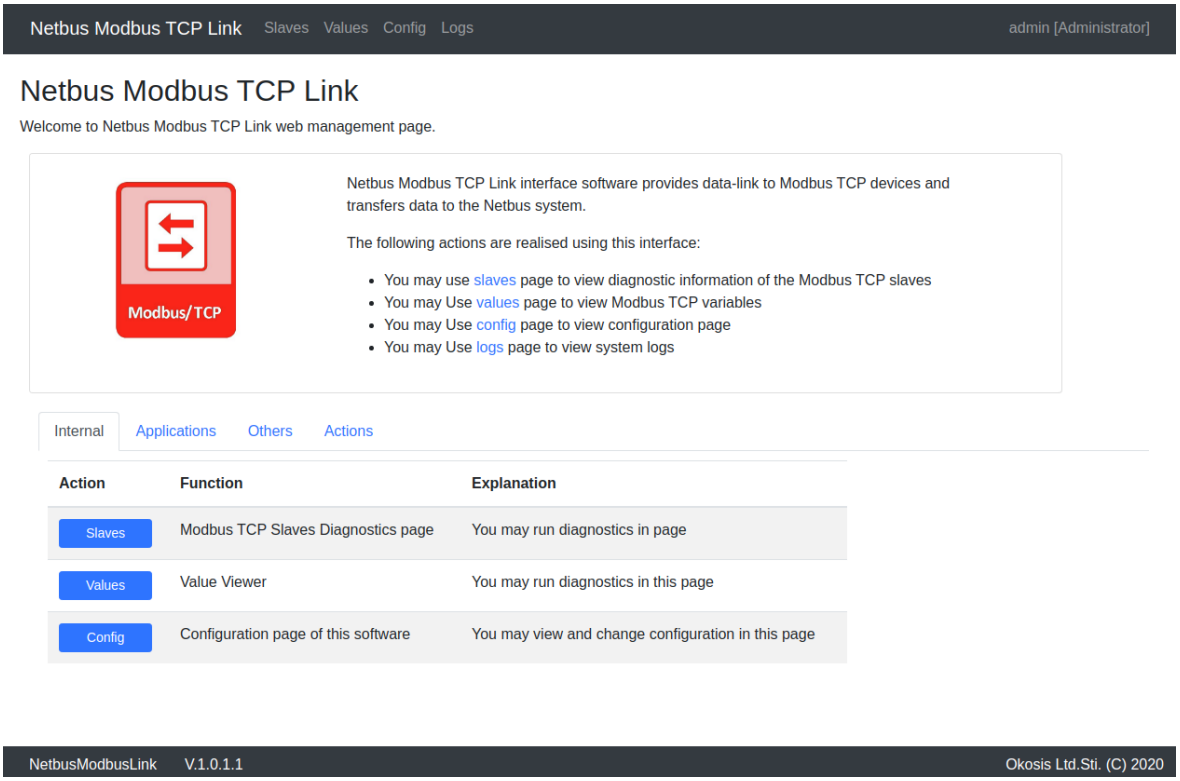


Figure 47: Netbus Modbus Link Service web server index page

The **“Config”** menu item represents the configuration page of the Modbus Link Service and it is available only for the users which login with **“Administrator”** authorization.

While the netbus pages are prepared with the same principles, all the functionality of the index page will not be explained again.

Please review the section **“2.2.Index Page”** for details.

5.4.1.2 Slaves Page

Slaves page contains short introduction text and a tab page for each Modbus slave device which contains “Communication Summary” and “Query” table as given below.

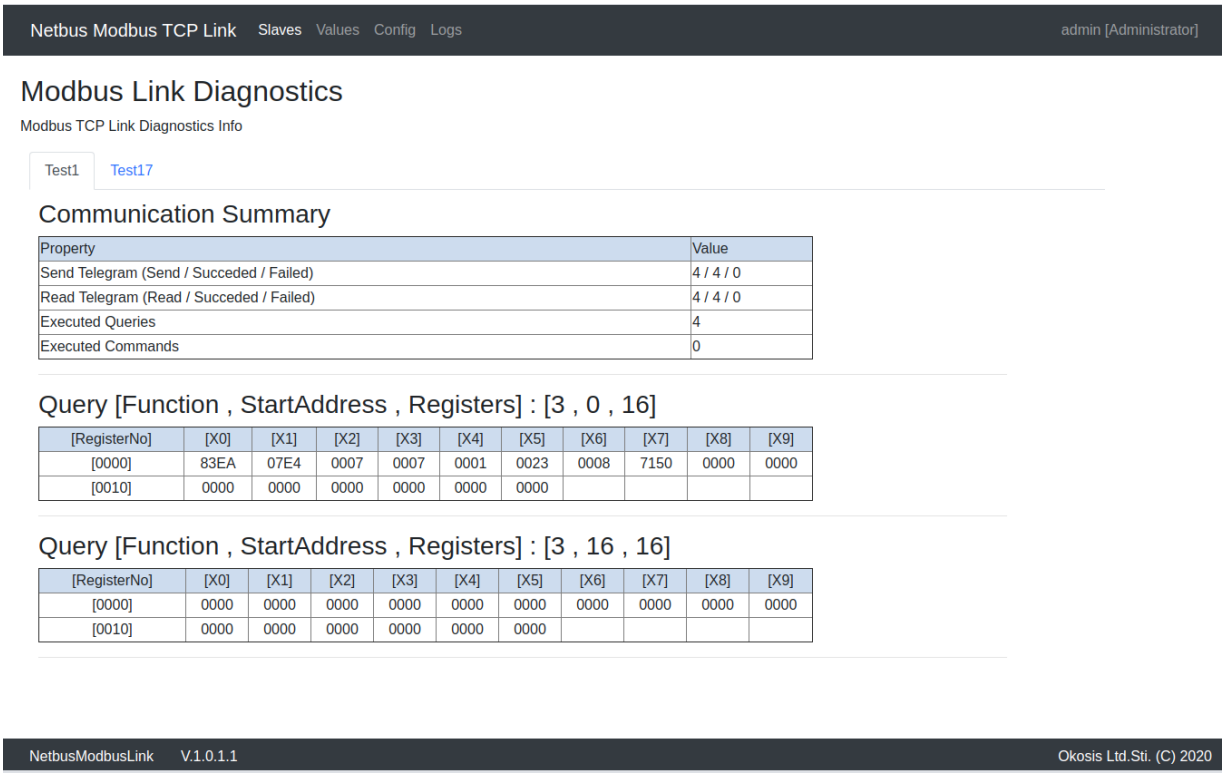


Figure 48: Modbus TCP Link Service Slaves page

Communication Summary

This table contains diagnostic information of the slave device as:

Send Telegram (Send / Succeeded / Failed)
Read Telegram (Send / Succeeded / Failed)
Executed Queries
Executed Commands

Query Table(s)

Every query which defined in **“5.4.1.4 Config Page ”** is represented with a table that contains short information about the query definition and below a raw register value presentation of the read values from the slave device. The newly added queries will be displayed here automatically.

The conversion from the raw register bits to a meaningful value is performed by the **Modbus TCP Link** service according to the configuration data described in the **“5.4.1.4.Config Page ”** and diagnostic values displayed in **“5.4.1.3.Values Page”.**

Note: Please be aware that page is just a snapshot and not updated cyclically.

5.4.1.3 Values Page

Values page contains short introduction, an **“autoupdate”** field, a filter group and the **“modbustags”** tabular view to show the actual values for diagnostic purposes.

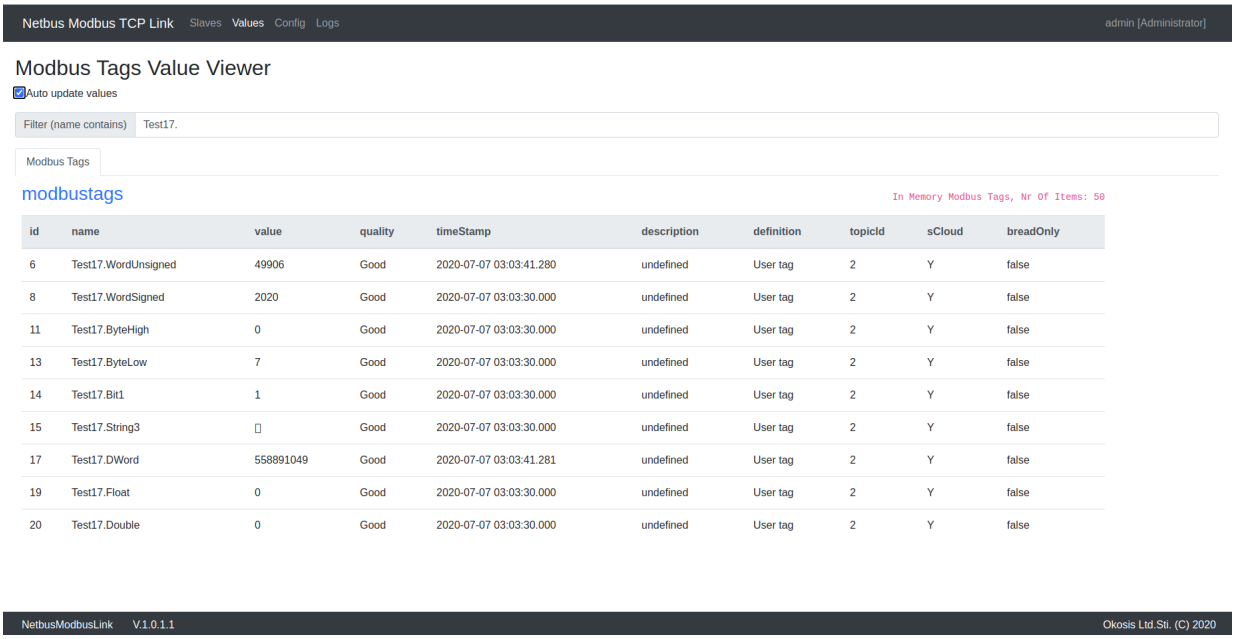


Figure 49: Modbus TCP Link Service Values Page

autoupdate:

Clicking this checkbox enables a periodical updating mechanism to refresh all values.

Filter

The text entered in this field is accepted as a **“check if contains this text”** filter for the **“name”** field which represents the name of the value.

modbustags tabular view

The conversion from the raw register bits to a meaningful value is performed by the **Modbus TCP Link** service according to the configuration data described in the **“5.4.1.4 Config Page ”** and extracted diagnostic values are displayed here.

The fields of the tabular view and explanation for them is as follows:

id: This is the auto generated id value, nothing to do.
Name: The name of the variable, The slave name inserted as a prefix to all values
value: The actual online value
quality: This is OPC quality (0x80: Good)
timestamp: Date time stamp of the value
description: Textual description of the value, nothing to do with software
definition: User tag / System tag (The tags with register values can be auto-generated)
topicsId: The integer id of the topic, Start system, nothing to do here, backward compatibility.
sCloud: The cloud string for Start system, nothing to do here, backward compatibility.
bReadOnly: The tag is read only which is 1 or not 0.

5.4.1.4 Config Page

Config page contains short introduction, a button and status group to Save/Load configuration files and "**slaves / tags / uplinks"** tabular view to summarize / edit the Modbus TCP slaves and their configuration settings.

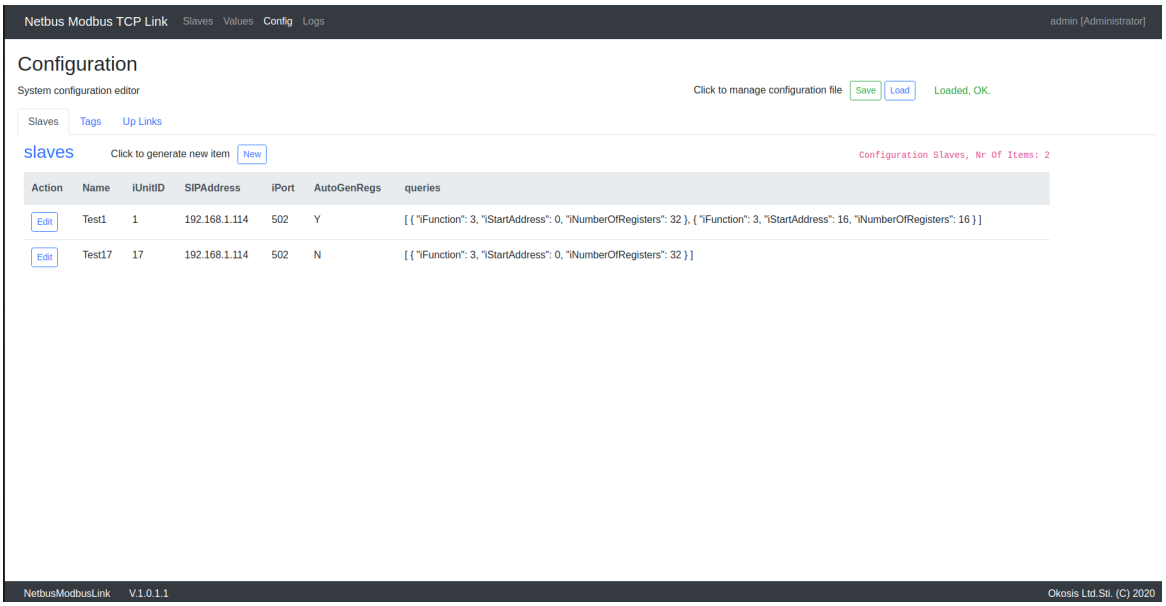


Figure 50: Modbus TCP Links Config Page

- **Save & Load configuration files**

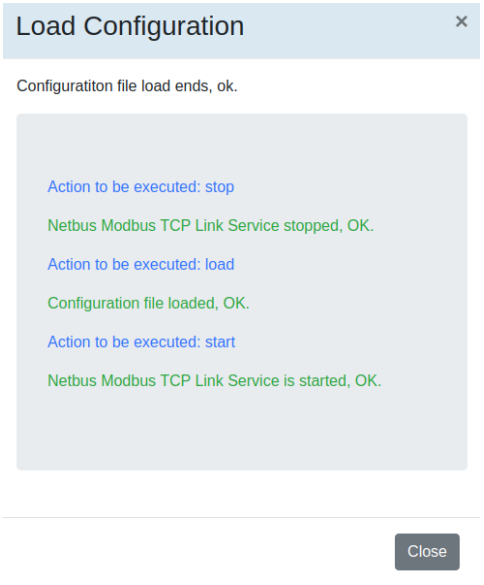


Figure 51: Configuration loading popup

By clicking the Save button, you save the online configuration data to the disk as the persistent configuration file. The configuration file exists in the **"config"** folder of Netbus base folder.

The exact root and file name is as given below :
/home/root/Publish/Netbus/config/ModbusTCP.json

By clicking the Load button, you stop the service, load the configuration data that exists in the disk to the actual system and restart the service with these newly loaded configuration data promptly.

The reloading actions are performed by the system in the correct order and timing and the system becomes off and available in possible minimum time.

Please be aware that Load operation causes a short service interruption in the system.

- **Slaves / tags / uplinks tab views**

Slaves tabular view contains links for JSON formatted data export page of slaves summary, a “new” button to serve generation of a slave and a tabular view containing configuration data. In this area there are three tabs as given below:

5.4.1.5 Slaves tab page

Here listed the Modbus TCP slave devices and their configuration parameters.

Both editors display some buttons and action result status displayer fields at the end of popup window which have following functionality:

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

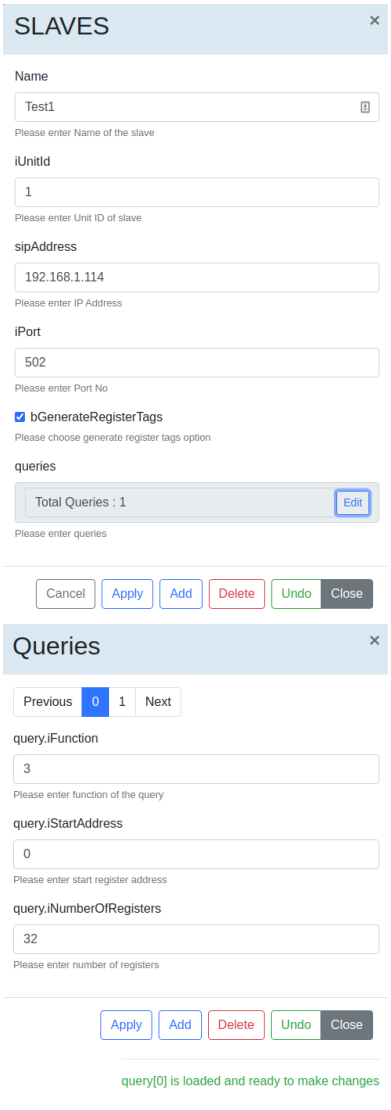


Figure 52: Slaves and Queries editors

Slaves editor

Action: Editor actions listed here, If you click the “Edit” button, “SLAVES” window popup and shows the parameters of slave device.

Name : The Modbus TCP slave device name. This parameter is very important and will be a prefix for tag names defined in Netbus System. The tag name will in format of : “**Name**.AnyTag”

iUnitID: Modbus Unit ID of the device, integer

SIPAddress : IP Address of the slave device

iPort: The Port number of slave device.(usually 502 for Modbus TCP)

bGenerteRegisterTags: Automatic generation of the query registers. The Modbus registers of the slave device which received as responses of the queries are generated and transmitted to the Netbus System, tags as: R0017:Word:16bSigned : Register 17 as unsigned integer.
Please review the section **“5.4.1.6Tags Description”.**

queries: this field is for the Modbus TCP queries which will be performed for data exchange. The total queries shown in the disabled edit box and an Edit button exist which opens queries popup as given on the left side.

Queries editor

On the queries window there is a query item index selection editor and parameters given as:

queryFunction: Modbus Function, integer (FC3 and FC4 is supported)

QueryStartAddress: Query Start address, integer

queryNumberOfRegisters: Number of register, integer

Details are given in the following sections:

- **5.4.1.7.Tags tab page**
- **5.4.1.8.Up Links tab page**

5.4.1.6 Tags Description

For detailed information about Tags please see the section **“3.2.5.Tags ”**.

The Modbus TCP tags can be generated in two ways as given below:

Auto generated Tags

In he configuration phase, in the slave editor of **“5.4.1.4.Config Page ”**, you may check the option “**bGenerteRegisterTags**” for automatic generation of the query registers as Tags. The Modbus registers of the slave device which received as responses of the queries are generated and transmitted to the Netbus System in these tags.
The tag description format of the tag will be formatted as given below:
R0011:Word:16bUnsigned : Register 11 as unsigned integer.

These types of variables will be shown as “System Tag” in the definition column of Values table please see section **“5.4.1.3.Values Page”**.

User Tags

You may define your own set of tags which represents queried register(s) values extracted and converted in custom data formats :

Register definition format : R[XXXX].DataType.Extra
XXXX: Register number possible values: (0 .. 9999)
DataType: Bit, Byte, Word, Float, Double
Extra : this part contains extra information in context with DataType as:
R[XXXX].Bit:b : The extra information **b** is the bit number possible values: **(0..7)**
R[XXXX].Byte:[Place] : Place contains byte’s place as : “**Low**” and “**High**”
R[XXXX].Word:[Extra] : Extra field contains description as given below:
16bSigned: Signed value, 16 bit length = 2 bytes
16bUnsigned: Signed value, 16 bit length = 2 bytes
32bSigned: Signed value, 32 bit length = 4 bytes
32bUnsigned: Signed value, 32 bit length = 4 bytes
64bSigned: Signed value, 64 bit length = 8 bytes
64bUnsigned: Unsigned value, 64 bit length = 8 bytes

Examples:
Here is some examples of tag Description

R0001:Word:16bSigned : This is “auto” register format, Register 1 as 16 bits, signed
R0003:Byte:Low: Modbus register 3’s low byte, as a byte = 1 byte
R0005:String:3: This is a string starting from Modbus register 5, 3 chars = 3 bytes
R0008:Float : Float based on Modbus registers 8 and 9 = 4 bytes
R0012:Double: Double integer based on Modbus registers 12-13-14-15 = 8 bytes

5.4.1.7 Tags tab page

Modbus TCP Tags tab page contains a “New” button and tabular list of “Tag”s.

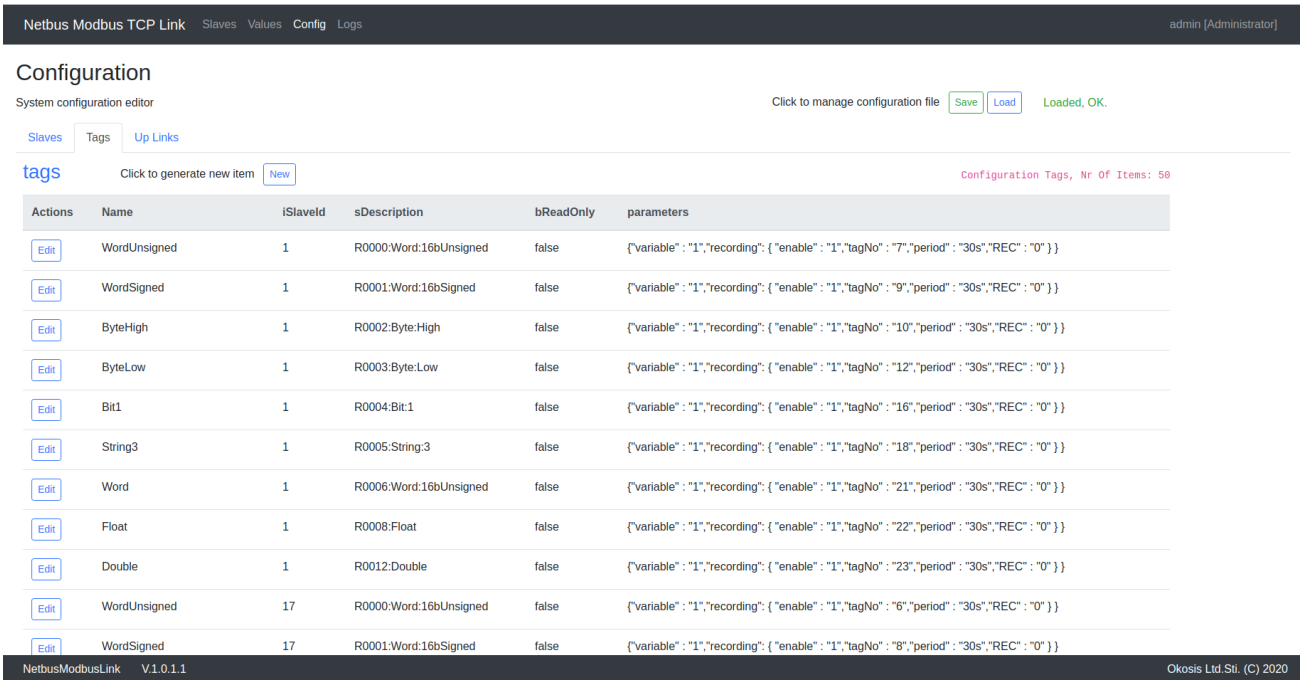


Figure 53: Modbus TCP Tag List

The fields of the tags tabular view is shown below:

Actions: Editor actions listed here, If you click the “Edit” button, “TAGS” window popup and shows parameters of the tag.

Name: Name of the tag, please enter a unique name

iSlaveId: This is the ID of the slave which the tag will be assigned to.

sDescription: Textual description

bReadOnly: Check this option if you want the tag to be read only, Read only tags are write protected.

parameters: Recording parameters of the tag is given. The text box is disabled, Click the edit button to see the “Parameters” popup.

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

Clicking the **“New”** button, opens up the Tags popup window and shows default values.

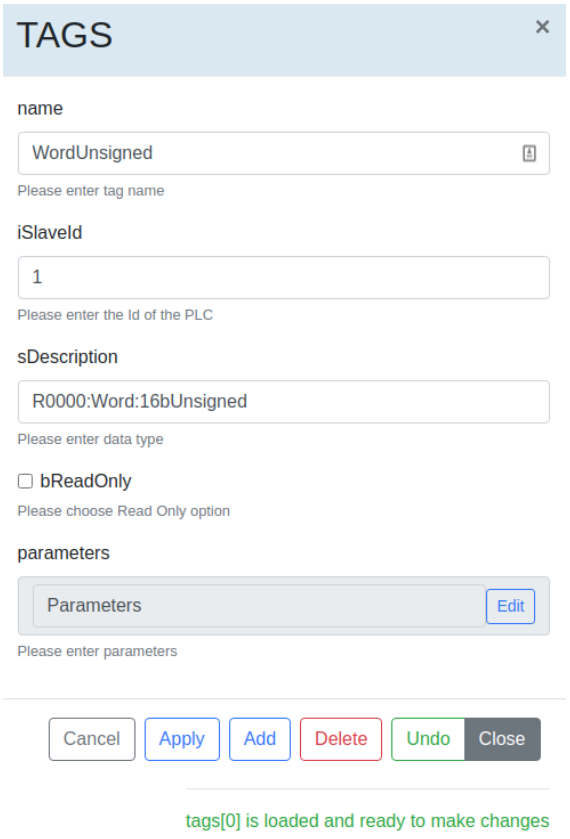


Figure 54: Modbus TCP Link Tags editor

Tags Popup

The tags popup shows following fields:

Actions: Editor actions listed here, If you click the “Edit” button, “TAGS” window popup and shows parameters of the tag.

Name: Name of the tag, please enter a unique name

iSlaveId: This is the ID of the slave which the tag will be assigned to.

sDescription: Textual description

bReadOnly: Check this option if you want the tag to be read only, Read only tags are write protected.

parameters: Recording parameters of the tag is given. The text box is disabled, Click the edit button to see the “Parameters” popup.

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

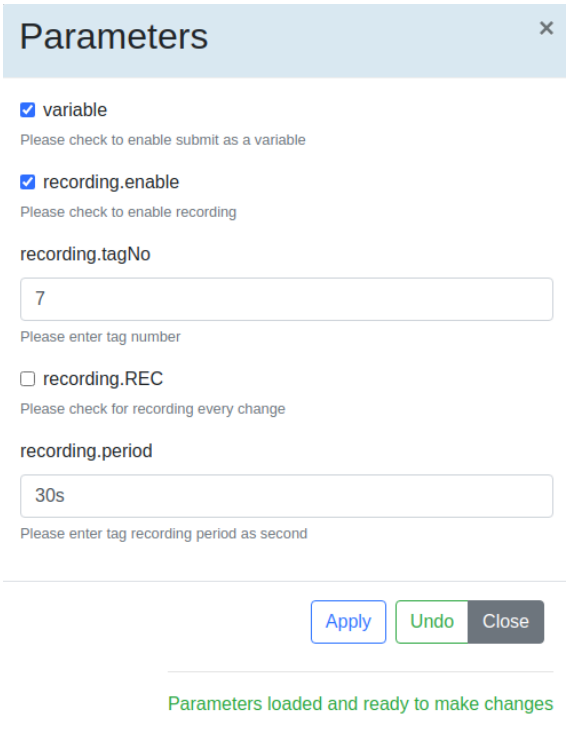


Figure 55: Parameters popup

Parameters Popup

The parameters popup displays tag’s parameters:

variable: Check this option if you want this Tag to be a Variable which will be updated and presented online.

Recording parameters:

recording.enable: Check this option if you want to enable Recording actions for this Tag

recording.tagNo: While the Tag registration procedure initiated at the startup of the link service, the unique tagNo is generated by the Netbus Pro Core system and send back to the link service.

recording.REC: Check to enable “Record Every Change”

recording.period: The recording period in seconds.

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

5.4.1.8 Up Links tab page

Modbus TCP Up Links tab page contains a “New” button and tabular list of “uplink”s.

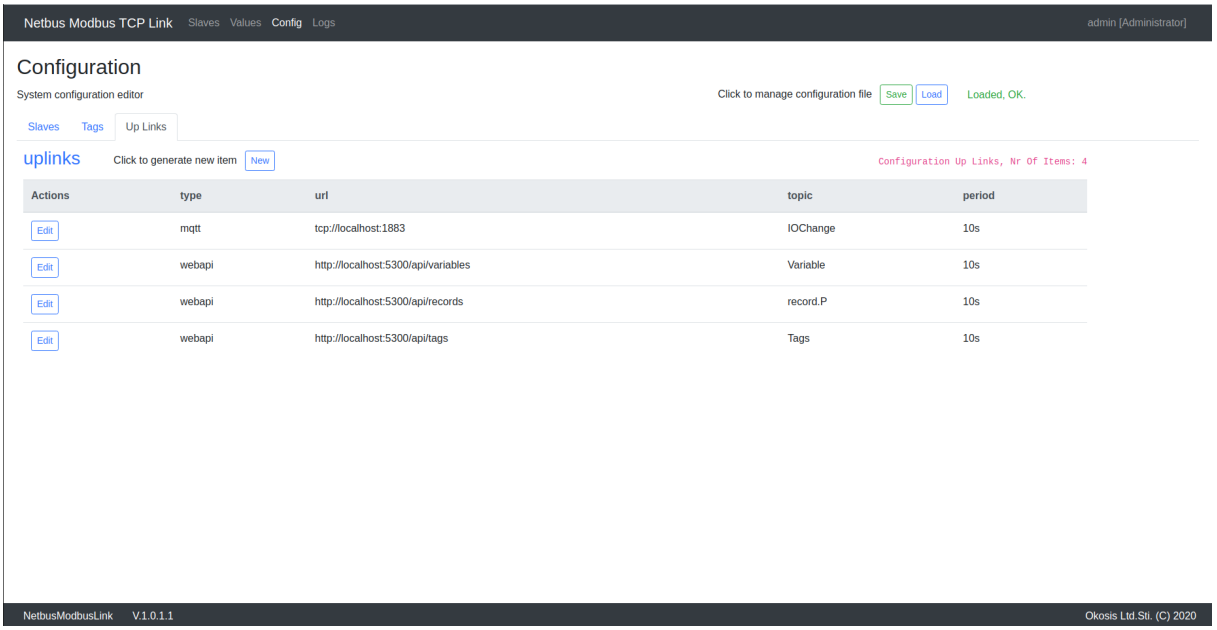


Figure 56: Up Links Tab view

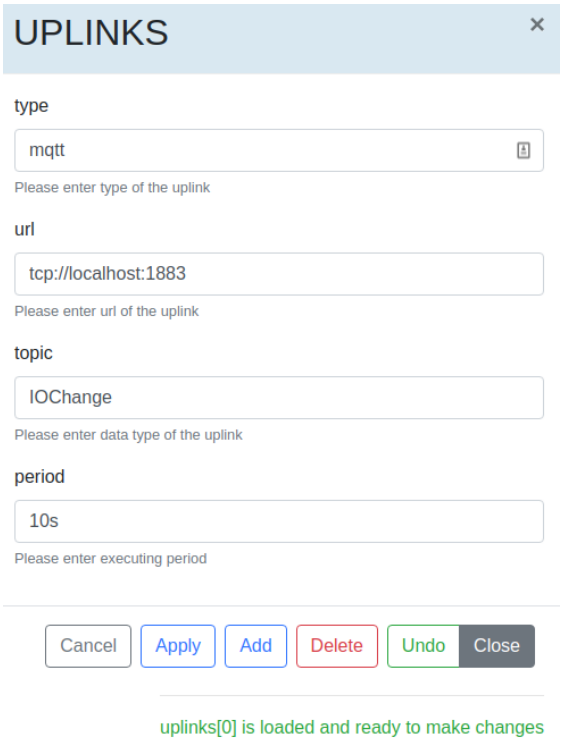


Figure 57: Up Links editor

The Up Links schema contains following fields:

Actions: This field contains Edit button shows

type: Type of interface which given as one of the :

“mqtt” : MQTT interface

“webapi” : WebAPI interface

url: the url of the upper system

topic: the topic detail for the upper system

Netbus Pro System has following topics:

for type “mqtt”:

“IOChange” : This is the mqtt interface

for type “webapi”:

“Variable” : Variable interface topic for **“Variable”**s

“Record.P”: Record interface topic for **“Record”**s

“Tag”: Tag definition interface topic for **“Tag”**s

period: Execution period of action

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

5.4.1.9 Logs Page

The **“Logs page”** contains application logs of the service represented in tabular view.

While the netbus pages are prepared with the same principles, all the functionality will not be explained again. Please review the section **“2.9.Logs Page”** for details.

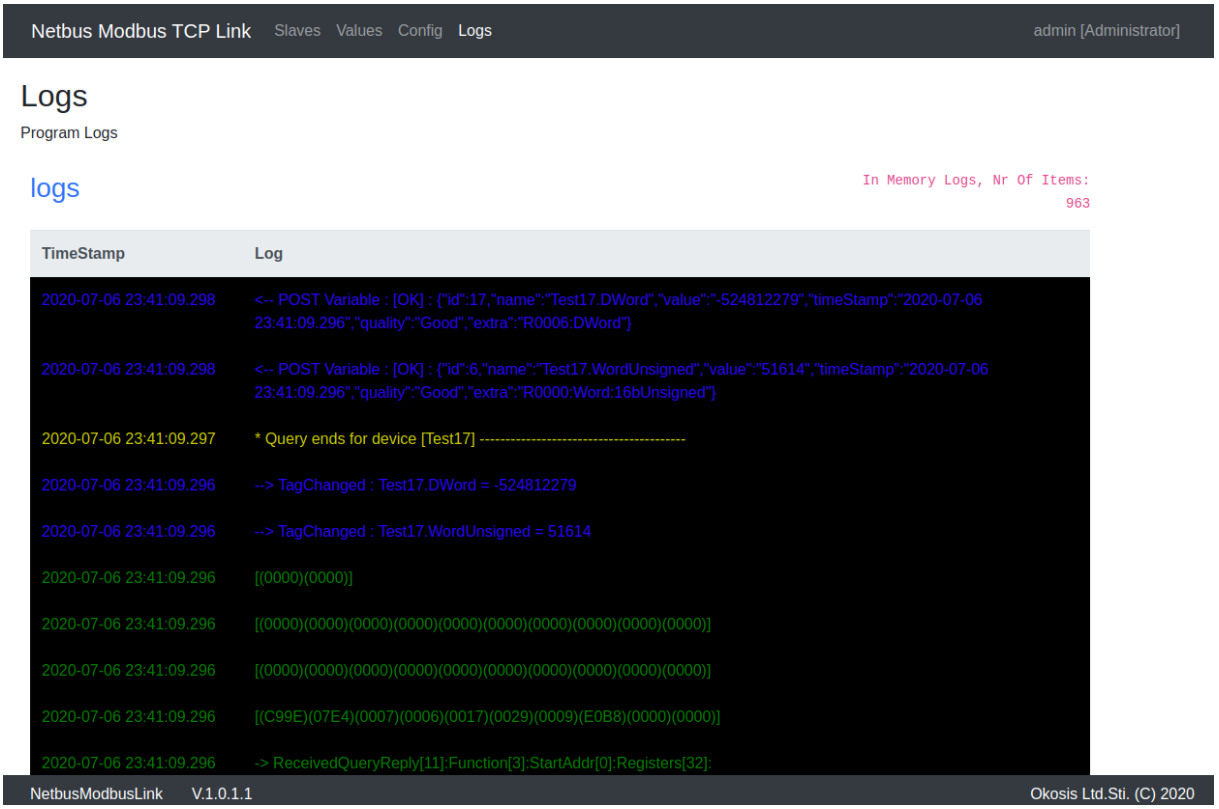


Figure 58: Netbus Modbus Link Logs Page

5.4.1.10 Login Page

Login link exists on the upper right corner of the page which redirects to the unique login page of the service.

There is a unique **“Login Page”** which is valid for all the Netbus Pro System services. The login / logout action which is performed successfully in the Login page of the service will be valid for all Netbus Pro System applications as well.

For more details on Logging in please see section **“2.3.Login Page”**

5.5. Netbus S7 Link Service

Netbus S7 Link Services performs data exchange functionality with the slave devices which have Siemens S7 communication capabilities to establish data **“link”** and integration with Netbus Pro System.

5.5.1. Netbus S7 Link Web Server

5.5.1.1 Index Page

Index page contains short introduction text, informative link area and tab page containing links for Internal functionality, Netbus Applications etc.

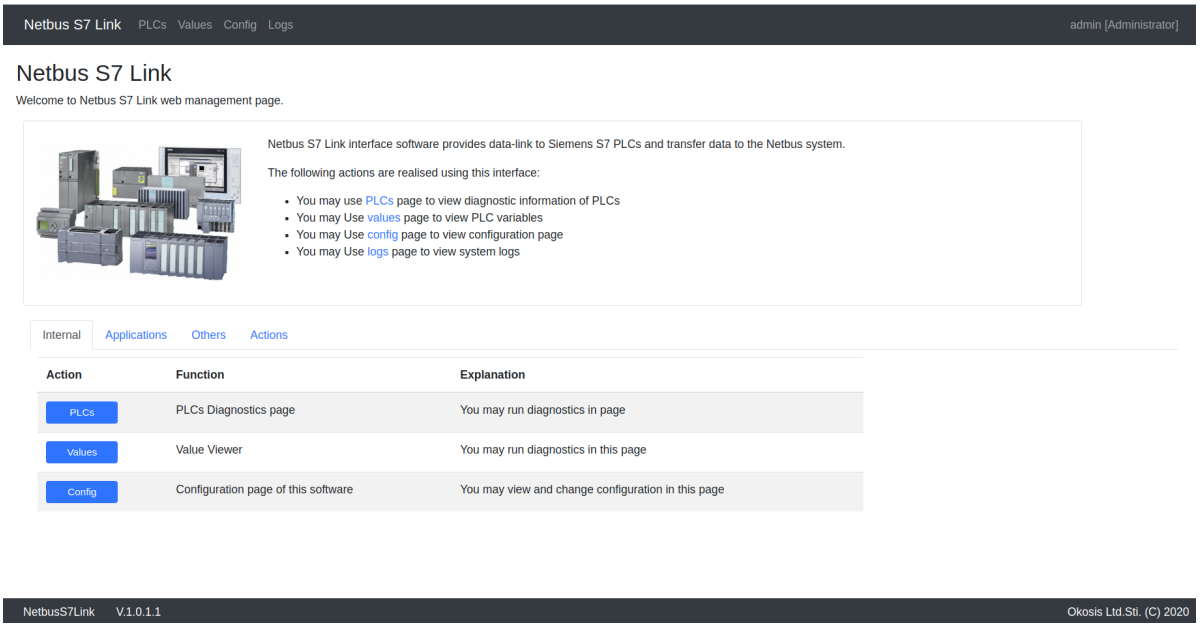


Figure 59: Index Page

The **“Config”** menu item represents the configuration page of the Netbus S7 Link Service and it is available only for the users which login with **“Administrator”** authorization.

While the netbus pages are prepared with the same principles, all the functionality of the index page will not be explained again.

Please review the section **“2.2.Index Page”** for details.

5.5.1.2 PLCs Page

PLCs page contains short introduction text and a tab page container which contains a tab page for each S7 device that contains “Communication Summary”, “PLC Device Information”, and “list of queries of the device” given as below.

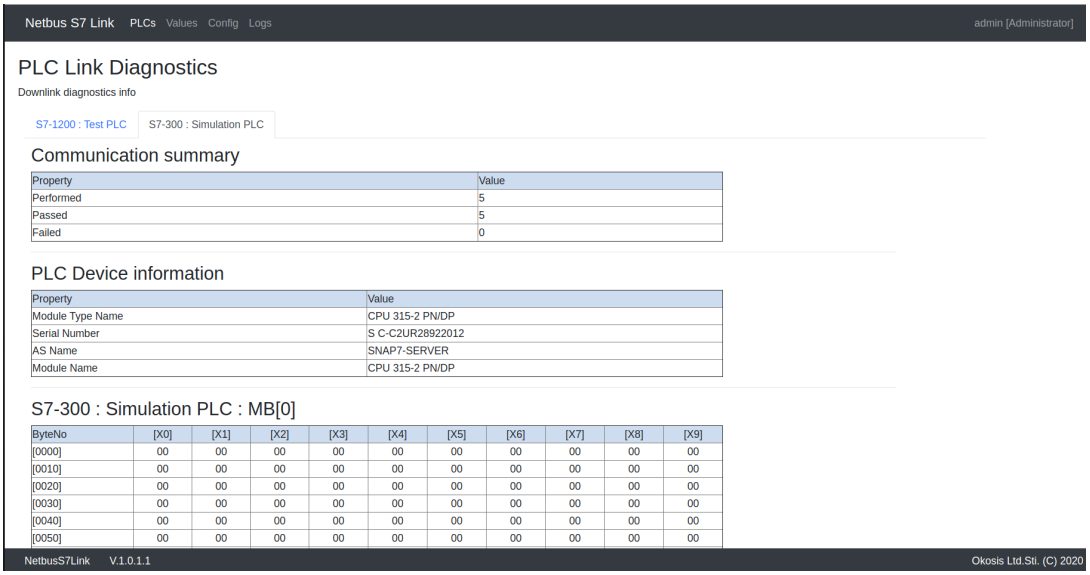


Figure 60: PLCs Page

Communication Summary

This table contains diagnostic information of the PLC queries as:

Performed : Performed number of queries
Passed : Passed number of queries
Failed : Failed number of queries

PLC Device Information

This table contains identification information of the PLC as:

Module Type Name : The type of CPU, for example : CPU 315-2 PN/DP
Serial Number : Serial number of system, for example : S C-C2UR28922012
AS Name : AS Name, for example SNAP7-SERVER
Module Name : Module Type, for example : CPU 315-2 PN/DP

Query Table(s)

Every query which defined in **“5.5.1.4.Config Page ”** is represented with a table that contains short information about the query definition and below a raw register value presentation of the read values from the slave device. The newly added queries will be displayed here automatically. The conversion from the raw register bits to a meaningful value is performed by the **Netbus S7 Link** service according to the configuration data described in the **“5.5.1.4.Config Page ”** and diagnostic values displayed in **“5.5.1.3.Values Page ”.**

Note: Please be aware that page is just a snapshot and not updated cyclically.

5.5.1.3 Values Page

Values page contains short introduction, an **“autoupdate”** field, a filter group and the **“plctags”** tabular view to show the actual values for diagnostic purposes.

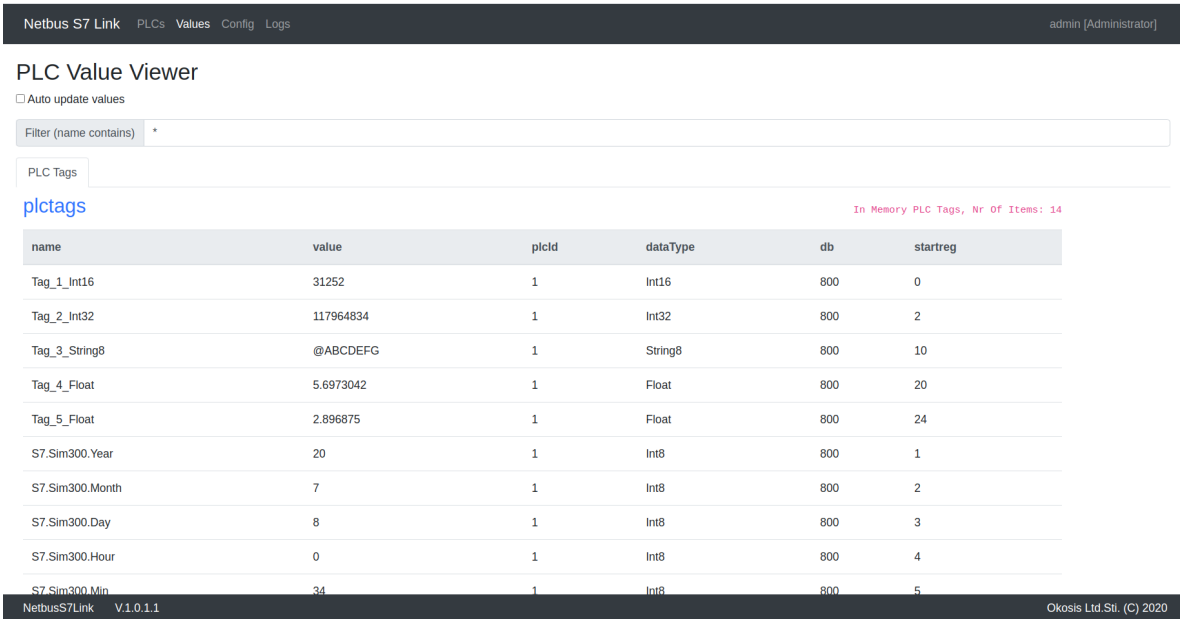


Figure 61: Values Page

autoupdate:

Clicking this checkbox enables a periodical updating mechanism to refresh all values.

Filter

The text entered in this field is accepted as a **“check if contains this text”** filter for the **“name”** field which represents the name of the value.

plctags tabular view

The conversion from the raw register bits to a meaningful value is performed by the **Netbus S7 Link** service according to the configuration data described in the **“5.5.1.4.Config Page ”** and extracted diagnostic values are displayed here.

The fields of the tabular view and explanation for them is as follows:

name: The name of the variable, The slave name inserted as a prefix to all values
value: The actual online value
plcid: The Id of the PLC
datatype: The type of data,
Int8 : Integer, 1 byte
Int16 : Integer, 2 bytes
Int32 : Integer, 4 bytes
StringX : String, X bytes
db: The DB number of the query
startReg: The start register no of the query

5.5.1.4 Config Page

Config page contains short introduction, a button and status group to Save/Load configuration files and **“PLCs / tags / uplinks”** tabular view to summarize / edit the Netbus S7 slaves and their configuration settings.

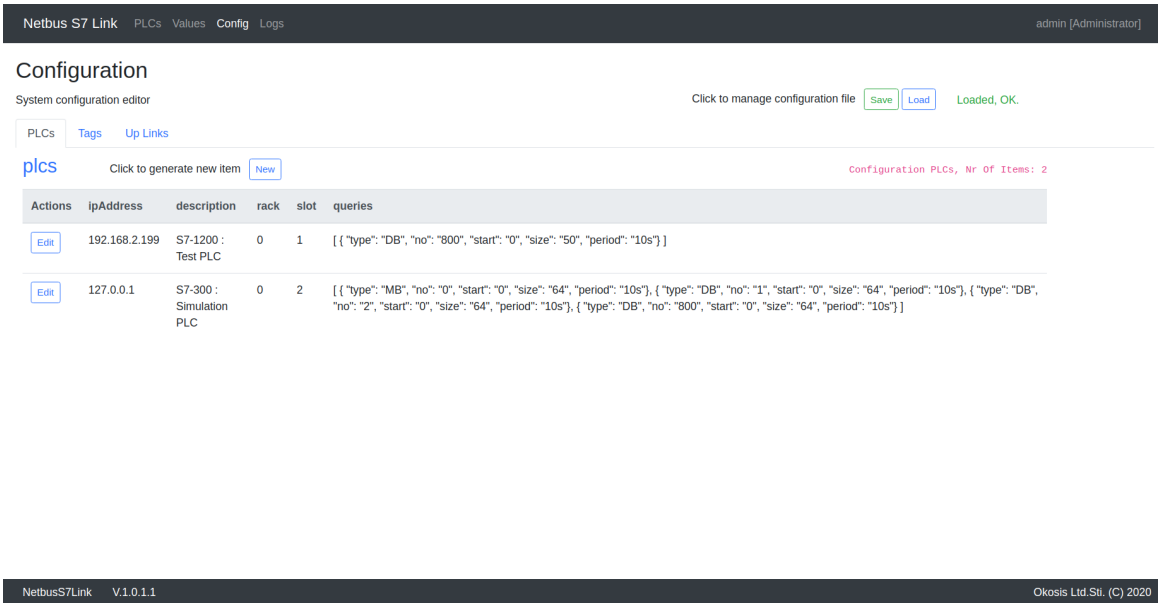


Figure 62: Config Page

Save & Load configuration files

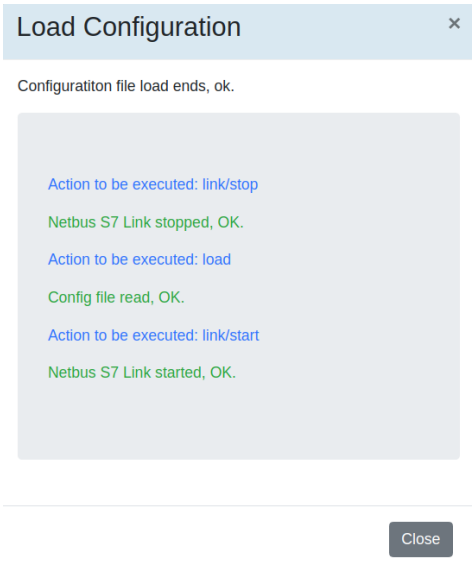


Figure 63: Configuration loading popup

By clicking the Save button, you save the online configuration data to the disk as the persistent configuration file. The configuration file exists in the **“config”** folder of Netbus base folder.

The exact root and file name is as given below :
/home/root/Publish/Netbus/config/S7Link.json

By clicking the Load button, you stop the service, load the configuration data that exists in the disk to the actual system and restart the service with these newly loaded configuration data promptly.

The reloading actions are performed by the system in the correct order and timing and the system becomes off and available in possible minimum time.

Please be aware that Load operation causes a short service interruption in the system.

PLCs / tags / uplinks tab views

Slaves tabular view contains links for JSON formatted data export page of slaves summary, a “new” button to serve generation of a slave and a tabular view containing configuration data. In this area there are three tabs as given below:

5.5.1.5 PLCs Tab Page

Here listed the Siemens S7 PLC devices and their configuration parameters.

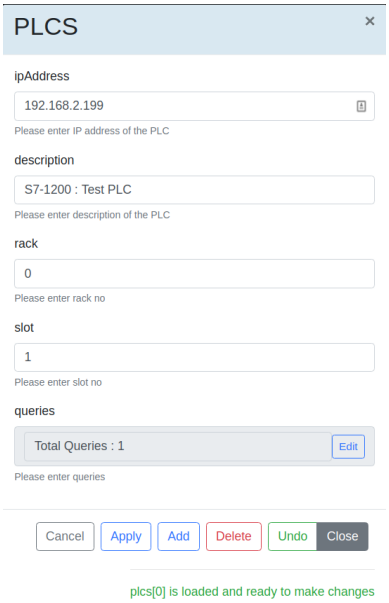


Figure 64: PLCs editor

PLCs editor

Action: Editor actions listed here, If you click the “Edit” button, “PLCS” window popup and shows the parameters of slave device.

ipAddress : IP Address of the slave device

description: Textual description of the device

rack: Rack number, for example: 0

slot: Slot number, for example : 1

queries: this field is for the Modbus TCP queries which will be performed for data exchange. The total queries shown int the disabled edit box and an Edit button exist which opens queries popup as given on the left side.

Figure 65: Queries editor

Queries editor

On the queries window there is a query item index selection editor and parameters given as:

query.type: Type of query: enter DB

query.no: Data block to query

query.start: Start address, for example : 0

query.size: Size of query, number of bytes to be read, for example: 50

query.period: Period of query, for example: 10s

Both editors display some buttons and action result status displayer fields at the end of popup window which have following functionality:

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

Tags tab page : given in section **“5.5.1.6.Tags tab page”.**
Up Links tab page: given in section **“5.5.1.7.Up Links tab page”.**

5.5.1.6 Tags tab page

The Siemens S7 tags can be generated manually as your own set of tags which represents queried register(s) values extracted and converted in custom data formats.

For detailed information about Tags please see the section **"3.2.5.Tags "**.

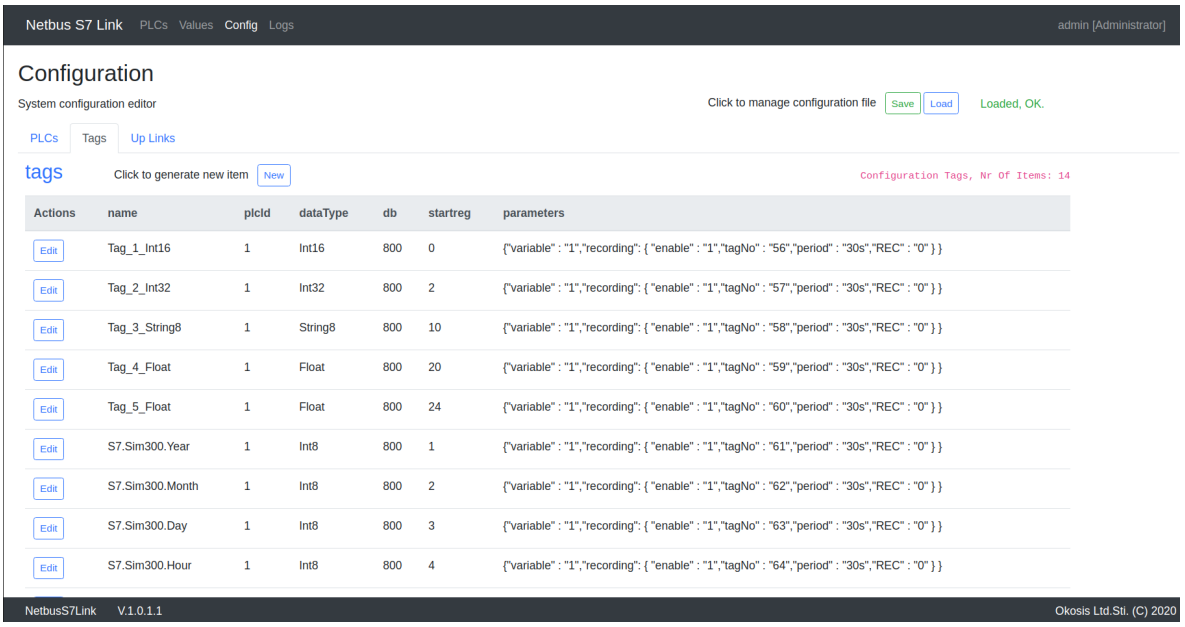


Figure 66: Netbus S7 Link tags tab page

The fields of the tags tabular view is shown below:

Actions: Editor actions listed here, If you click the "Edit" button, "TAGS" window popup and shows parameters of the tag.

Name: Name of the tag, please enter a unique name

plcId: This is the ID of the PLC which the tag will be assigned to.

dataType: Type of the data

db: Data Block number of the tag

startReg: Starting register of the tag

Parameters: Recording parameters of the tag is given. The text box is disabled, Click the edit button to see the "Parameters" popup.

Examples:

Here is some examples of Siemens S7 tags :

Tag_1_Int16	1	Int16	800	0
Tag_2_Int32	1	Int32	800	2
Tag_3_String8	1	String8	800	10
Tag_4_Float	1	Float	800	20

Clicking the **"New"** button, opens up the Tags popup window and shows default values.

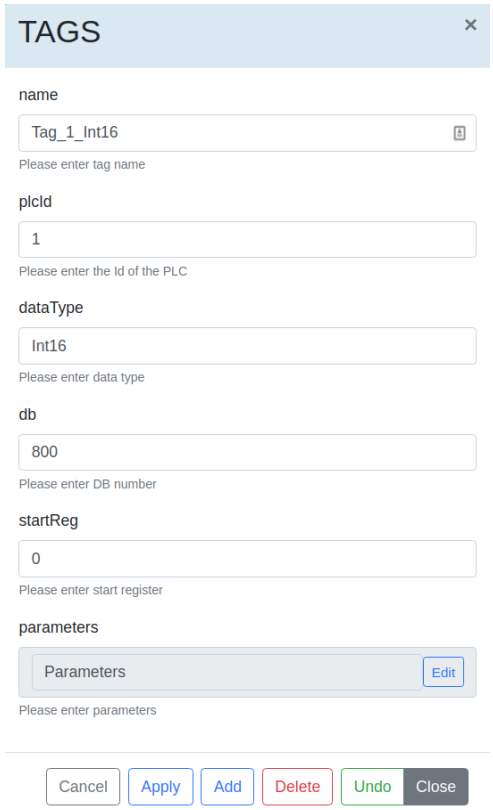


Figure 67: Netbus S7 Tags editor

Tags Popup

The tags popup shows following fields:

Actions: Editor actions listed here, If you click the “Edit” button, “TAGS” window popup and shows parameters of the tag.

Name: Name of the tag, please enter a unique name

plcId: This is the ID of the PLC which the tag will be assigned to.

dataType: Type of the data for example :

Int16,Int32, Float, Double, String8

db: Data block number of the tag

startReg: Starting register of the tag

parameters: Recording parameters of the tag is given. The text box is disabled, Click the edit button to see the “Parameters” popup.

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

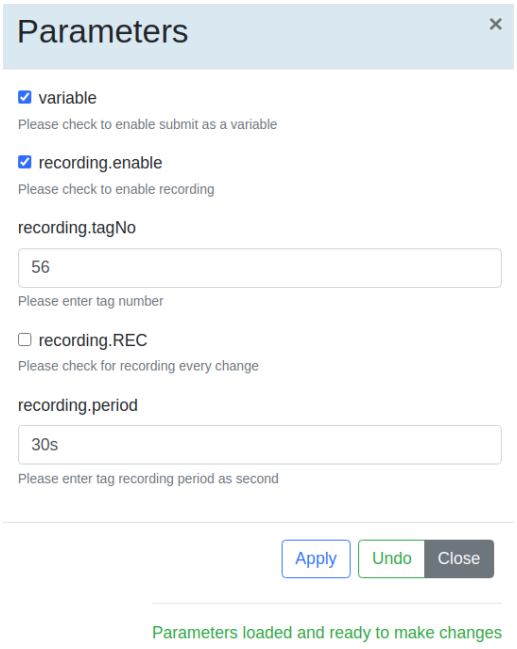


Figure 68: Netbus S7 tags parameters editor

Parameters Popup

The parameters popup shows following fields:

variable: Check this option if you want this Tag to be a Variable which will be updated and presented online.

Recording parameters:

recording.enable: Check this option if you want to enable Recording actions for this Tag

recording.tagNo: While the Tag registration procedure initiated at the startup of the link service, the unique tagNo is generated by the NetbusProCore system and sent back to the link service.

recording.REC: Check this option if you want to enable “Record Every Change”

recording.period: The recording period in seconds.

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

5.5.1.7 Up Links tab page

Modbus TCP Up Links tab page contains a “New” button and tabular list of “uplink”s.

For introduction of Up Links please review section :

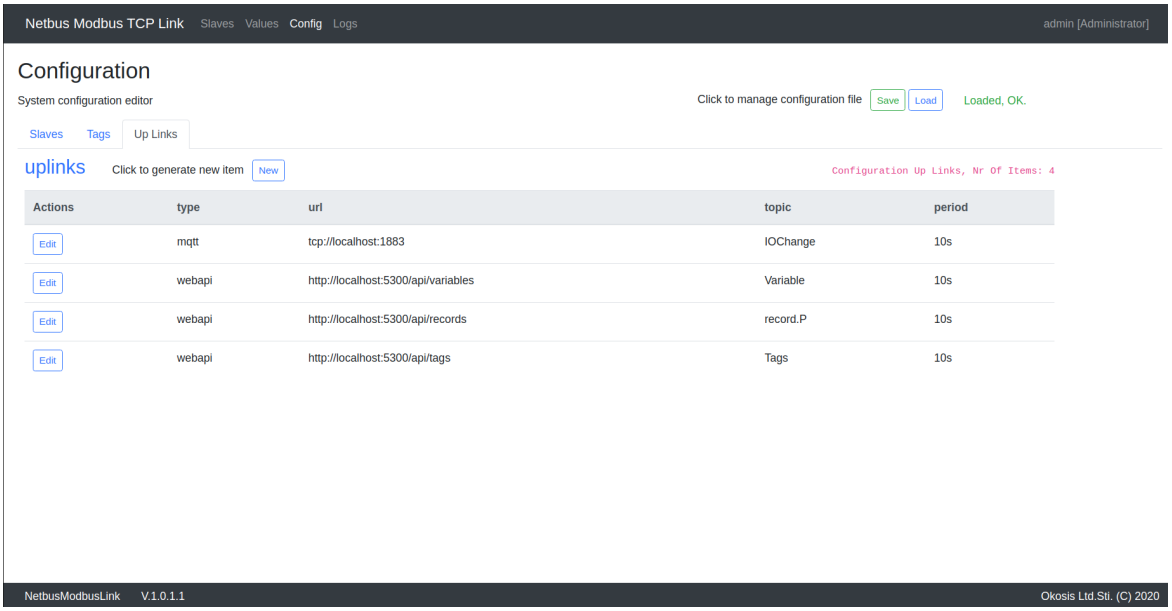


Figure 69: Siemens S7 Link Up Links

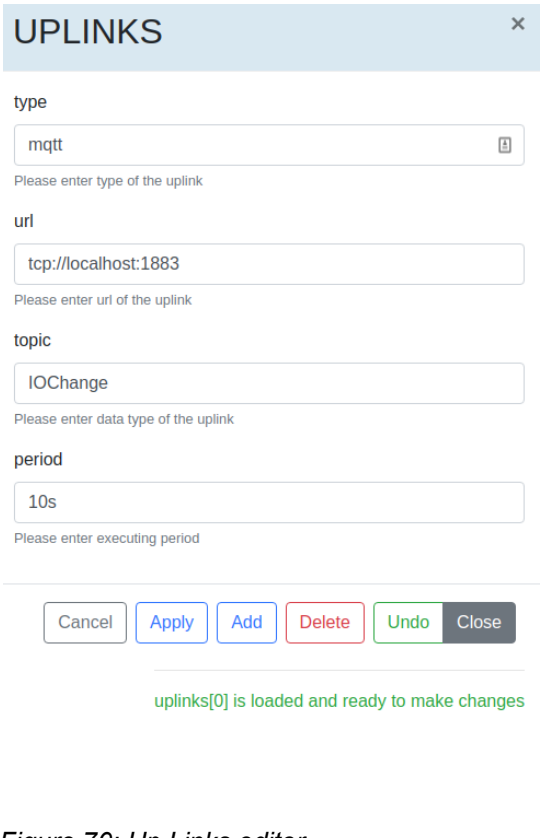


Figure 70: Up Links editor

The Up Links schema contains following fields:

Actions: This field contains Edit button shows

type: Type of interface which given as one of the :

“mqtt” : MQTT interface

“webapi” : WebAPI interface

url: the url of the upper system

topic: the topic detail for the upper system

Netbus Pro System has following topics:

for type “mqtt”:

“IOChange” : This is the mqtt interface

for type “webapi”:

“Variable” : Variable interface topic for **“Variable”**s

“Record.P”: Record interface topic for **“Record”**s

“Tag”: Tag definition interface topic for **“Tag”**s

period: Execution period of action

buttons area: Both popup contains cancel, apply, add, delete, undo and close buttons which operate respectively.

action result status displayer: Below the buttons area there is a text field to present the result of the actions provided in the buttons area and performed by the user.

5.5.1.8 Logs Page

The **“Logs page”** contains application logs of the service represented in tabular view.

While the netbus pages are prepared with the same principles, all the functionality will not be explained again. Please review the section **“2.9.Logs Page”** for details.

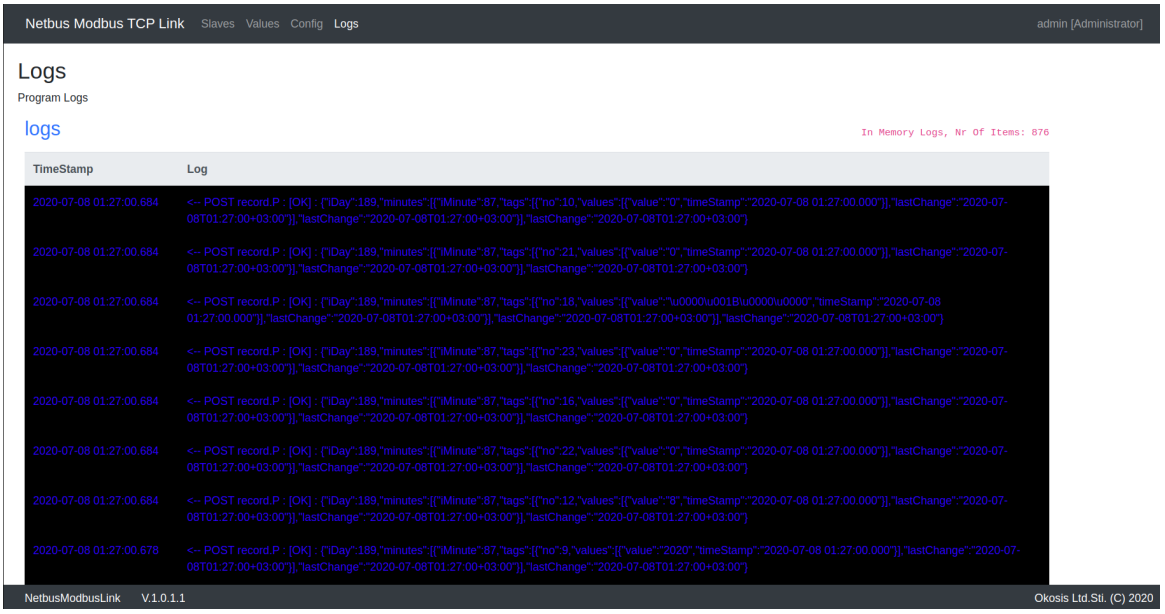


Figure 71: Siemens S7 Links Logs Page

5.5.1.9 Login Page

Login link exists on the upper right corner of the page which redirects to the unique login page of the service.

There is a unique **“Login Page”** which is valid for all the Netbus Pro System services. The login / logout action which is performed successfully in the Login page of the service will be valid for all Netbus Pro System applications as well.

For more details on Logging in please see section **“2.3.Login Page”**