User Manual

Ed.1 Rev.0 – 15/04/2015

Newtohm Srl – Via Scossabrillo, 7 – Dozza (BO) – P.IVA 02180230399
Tel. 05461880010 – Fax 05461880013 – Email info@newtohm.it

1 FOREWORD

WebVisor is a complete SCADA system able to withstand even the most demanding application thanks to its built in functions.

WebVisor allows the user to quickly set up a data logger and reporter, a suprevision system or a complete solution tailored to every need.

Since all data is recorded on a permanent database it allows to analyze all the data acquired and to perform scripts based on the data behaviour.

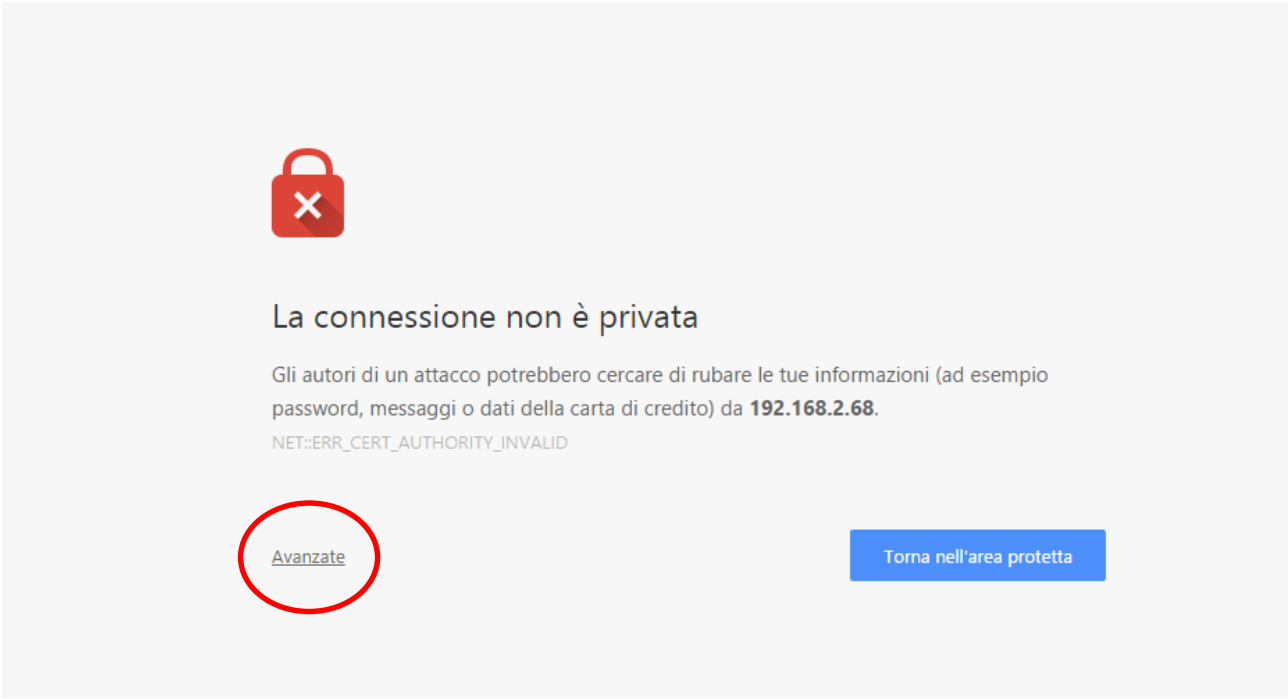
Last but not least since WebVisor is a web based SCADA it is possible to configure, run and control your system from every device.

2 CONFIGURATION

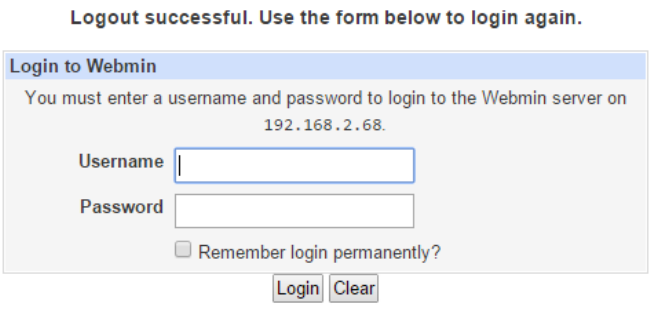
WebVisor has a builtin web server that allows to easily setup the network options.

The default IP address of the board is **192.168.2.68** so to connect to the board and to setup its networking follow the next steps:

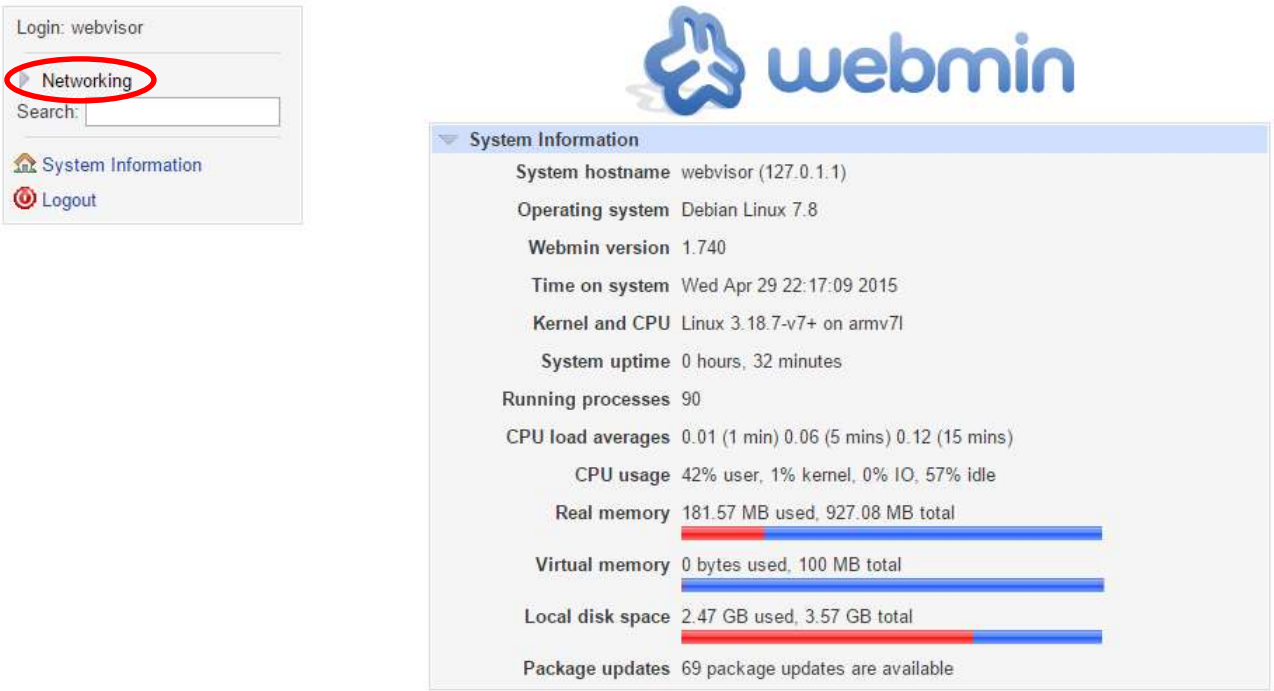
- Setup your PC with an IP address within the WebVisor's net range (i.e. 192.168.2.70)
- Open your browser and type the following address: **https://192.168.2.68:10000**
- If your browser gives a warning about the security extension of WebVisor please confirm that the site is trusted and continue to the login



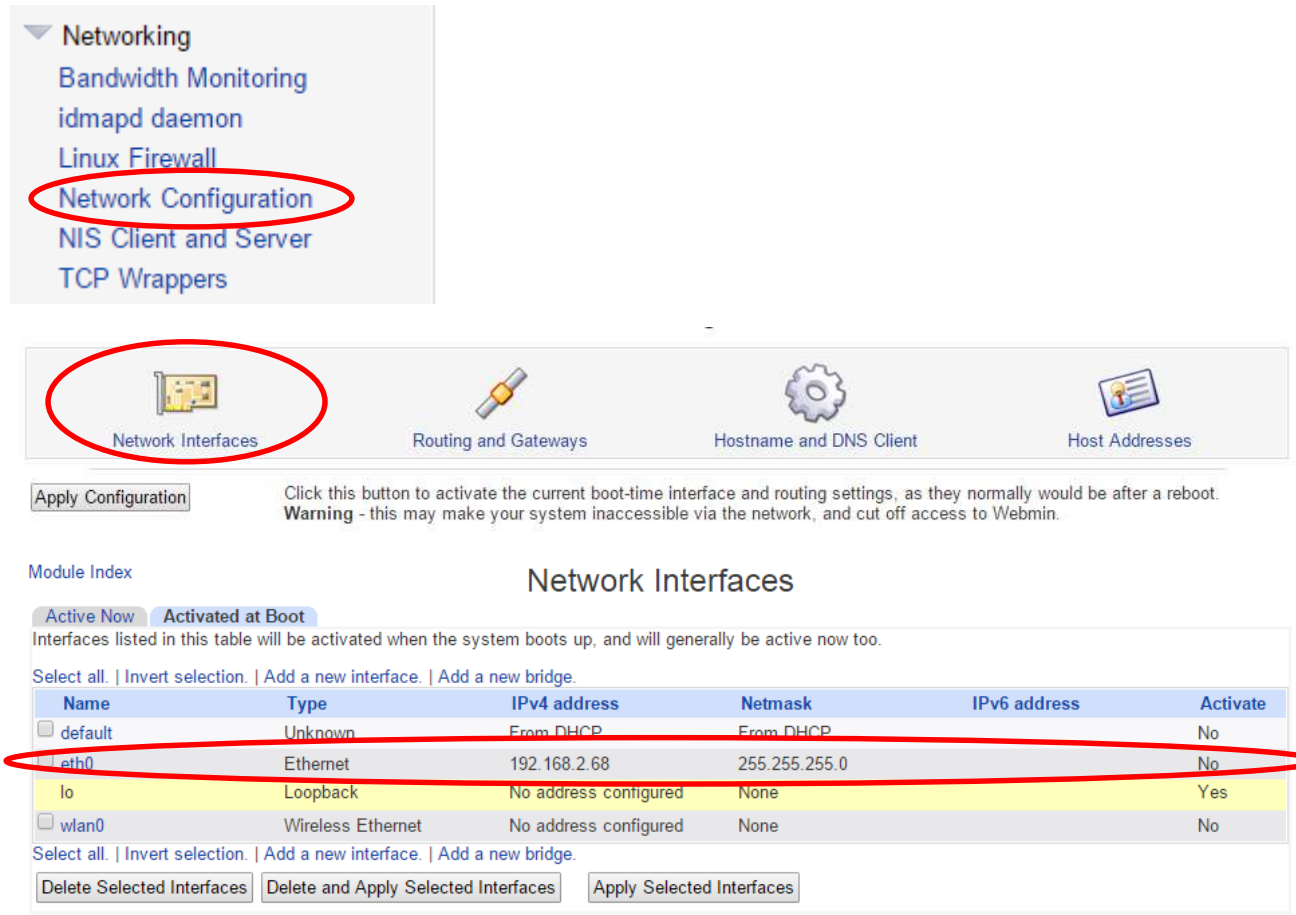
- In the login page use the following data:
  - Username: webvisor
  - Password: w3bV1s0r

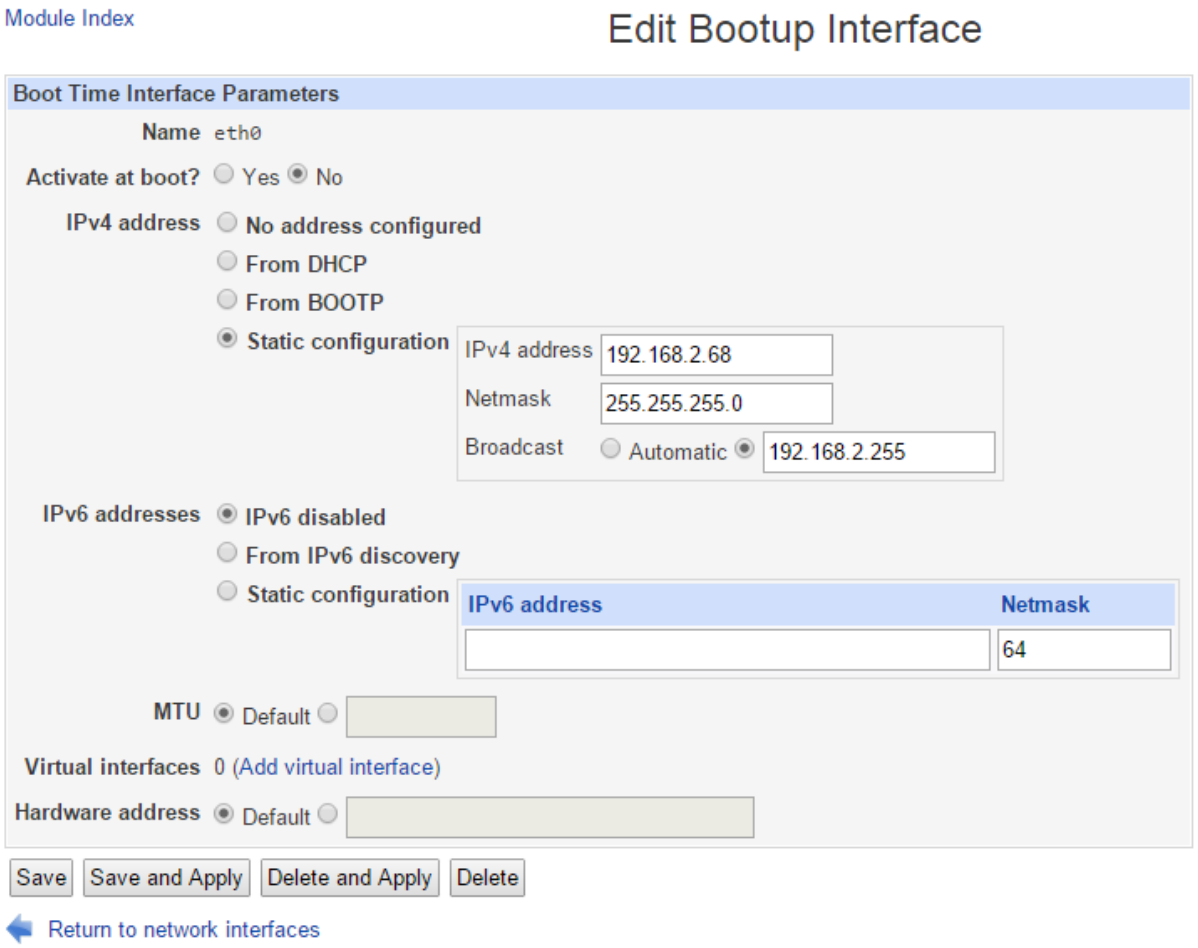


If the login is correct you will be presented with the following screen.



Select Networking, “Network Configuration” and “Network Interfaces” to configure the LAN card (eth0).





Setup the card as desired.

BE CAREFUL: A WRONG SETUP MAY RENDER THE BOARD UNUSABLE, DOUBLE CHECK EVERY STEP BEFORE APPLYING THE CHANGES.

3 OPERATING WEBVISOR

3.1 Introduction

3.1.1. Main Menu

Functionality in this application is primarily accessed using the controls in the header. Depending on the permissions granted to your user account, you will see various icons underneath the application logo. When you hover over these icons with your mouse, you will see an overlay message with a short description of what the icon is for.

Beside the header controls at the right-hand side of the header is the username of the user who is currently logged in.

When your application has an active alarm, you will see a flashing flag icon() and an associated description near the center of the header area. The color of the icon will indicate the severity of the alarm:

-  Information.
-  Urgent.
-  Critical.
-  Life safety.

You can click on the icon or the description to go to the list of active alarms.

3.1.2. Data Types

There are five supported data types:

- **Binary** (aka boolean) values have only two states, referred to in the system as the zero value and the one value. You can use a text renderer to convert the display of binary values to whatever alternate labels you need, such as "on/off", "high/low", "started/stopped", etc.
- **Multistate** values have multiple distinct states. (Strictly speaking, the binary type is a special case of a multistate.) Values are primitively represented as integers (e.g. 0, 1, 2, 7, ...), but, like binary values, you can use a text renderer to convert these values to appropriate text representations such as "on/off/disabled", "cool/heat/off", or anything else.
- **Numeric** (aka analog) values are decimal values represented in the system with a floating point variable. Temperature, humidity, price, and altitude are examples of numeric values. Text renderers can be used to determine display features like how many decimals places to display, whether to separate thousands (with commas or dots), and whether a suffix should be displayed (e.g. °F, kW/h, moles, etc). Range renderers can be used to convert ranges of numeric values into text labels.
- **Alphanumeric** values are simply strings of characters, such as the O/S description of an SNMP source.

- **Image** values are binary representations of image data. They are stored in files on the host file system (i.e. not in the database) and are cached in memory as necessary for performance purposes. Renderers can be used to create scaled images - such as thumbnails - for presentation.

3.1.3. Data Sources

Data sources are fundamental to the operation of this application. A **data source** is a *"place"* from which data is received. Virtually anything can be a data source, insofar as the communications protocol is supported by the application. Here are some examples:

- If you have a **Modbus** network accessible via RS232, RS485, TCP/IP, or UDP/IP, you can create a Modbus data source that will poll the network for data on an interval you can define.
- If you have equipment or an application that can send **data over HTTP** you can start an HTTP receiver data source that will listen for incoming connections and route the data to the appropriate points.
- For hardware that supports **SNMP**, start an SNMP data source. Values can be polled on an interval schedule, or traps can be received for report-on-exception.

Data can be read and updated in a **SQL data base** external to the system.

Data can be generated either randomly or predictively using a **Virtual data source**.

Data values that are received or collected by a data source are stored within data points.

3.1.4. Data Points

A data point is a collection of associated historical values. For example, a particular point might be a temperature reading from a room, while another point could be the humidity reading from the same room. Points can also be control values, such as a switch to turn a piece of equipment on or off.

There are many attributes that are used to control the behavior of points. Primarily, there is the concept of a **point locator**. Locators are used by data source to determine how to "find" the data for the particular point. For instance, a *SQL data source* has attributes where to find the specific database instance, *point locators* for the data source indicate the table and field names where to find specific values. The logical separation of data source and data point attributes depends on the communication protocol in question.

Data points attributes also determine many other aspects of the point, such as its name, how it should be logged (all data, changes in value only, or not at all), how long to keep logged data, how to format values for display, and how to chart values.

You can also configure data points with **event detectors**, which are used to detect conditions of interest in the point's values (eg. value has been too high for too long, is too low, changes too often, doesn't change at all, etc).

Points can be arranged into a hierarchy, or tree, to simplify management and display by using the Point Hierarchy functionality.

3.1.5. Monitoring

Monitoring of points within the system can be done in two ways. You can use the **watch list** to dynamically create tabular lists of points including their values, last update times, and charts of historical information (if the point is configured to support this). Values and charts are updated in real time without having to refresh your browser window. Charts of multiple points can also be displayed on demand.

You can also create **graphical views** of points using drag and drop functionality to position graphical representations of points on an arbitrary background image. Animated images can be used to create highly dynamic visualization of system behavior, and, like the watch lists, values are updated in real time without a browser refresh. These views can subsequently be marked as "public", so that they can be reused on public web sites.

3.1.6. Control

Control of external systems can also be achieved for points that can be set (aka writable or output). A **settable** point can be set to a user-defined value, such as a thermostat setting or equipment control switch. Both watch lists and graphical views provide simple means by which to input a value to set.
The point locator for the settable point determines how the data source sets the value within the external equipment.

3.1.7. Events

An **event** is the occurrence of a defined condition within the system. Both system-defined and user-defined events are available. System-defined events include data source operation errors, user log-ins, and system start-up/shutdown. User-defined events include point event detectors (see "event detectors" above), scheduled events, and compound events which detect conditions over multiple points using logical statements. There are also "audit events", which are raised when users make changes (additions, modifications, deletions) to run-time-affecting objects including data sources, data points, point event detectors, scheduled events, compound event detectors, and event handlers.

Once an event has been detected, it is acted upon by handlers. An **event handler** is a user-defined behavior that is to be performed upon the raising of a particular event, such as sending an email or setting the value of a settable point.

3.1.8. Application icons

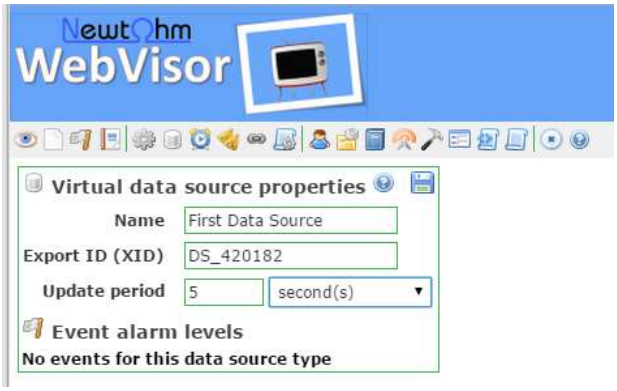
-  Data source
-  Data point
-  Chart

- Set point
- Watch list
- Graphical view
- Refresh
- Event detector
- Compound detector
- Scheduled event
- Event handler
- Alarm
- User
- User comment
- Report
- Mailing list
- Publisher
- Warning
- Log-out

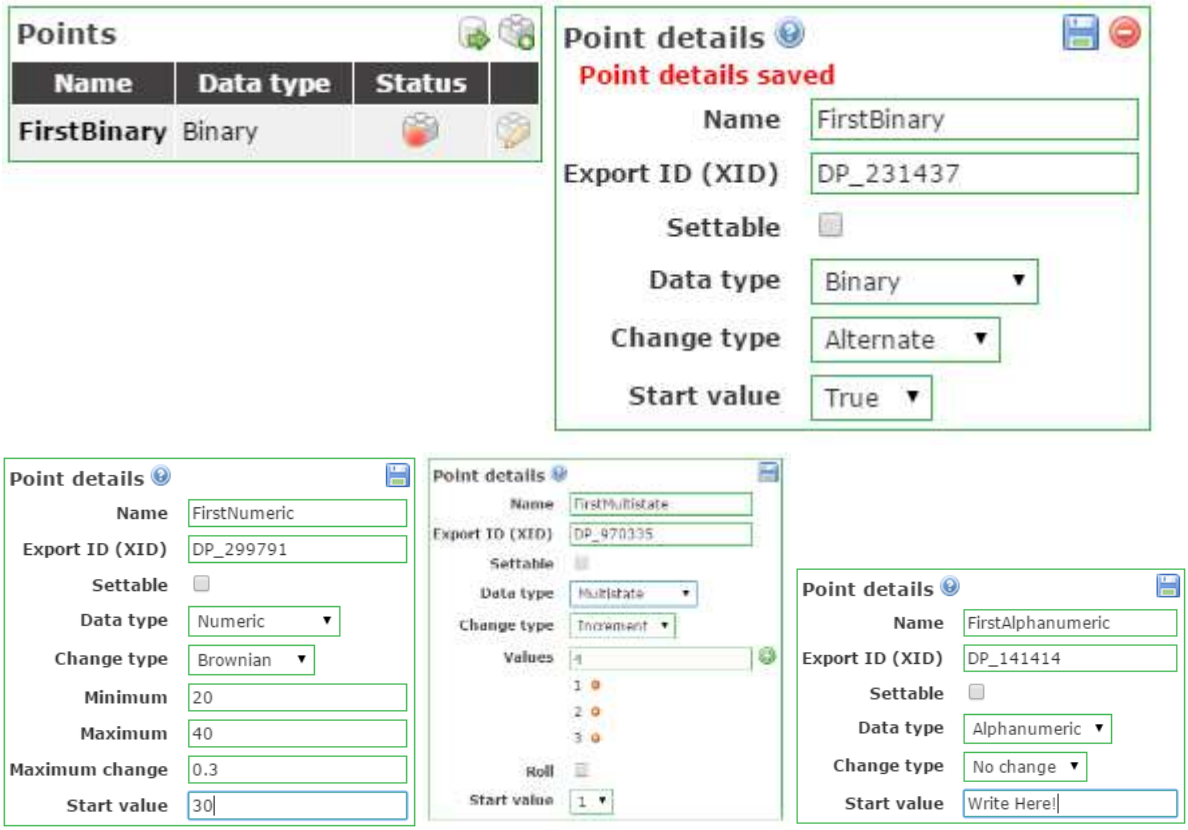
3.2 Adding Data Sources and Data Points

In this topic we show an example of how to create Data Sources and Data Points. In this example we will created a Data Source type *Virtual Data Source*, which is a simulated data source.

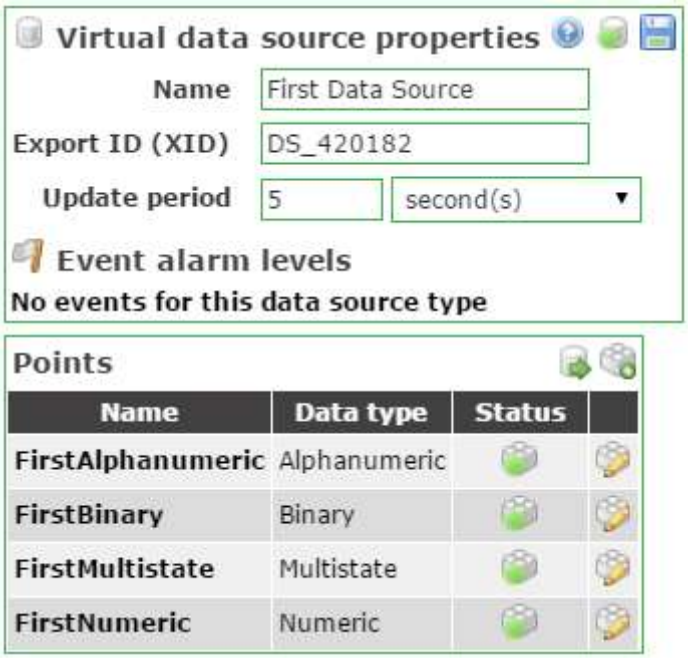
- From the main menu, choose *Data sources* .
- Select from the list *Virtual Data Source* and click the *Add* icon .
- Fill in as below and save your settings.



- After saving the Data source, the options for inclusion of Data points are enabled.
- Click *Add* and fill in as below and save your settings.



- Add a few more different type Data points.
- Than enable all the points by clicking on the icons and also enable the Data source .



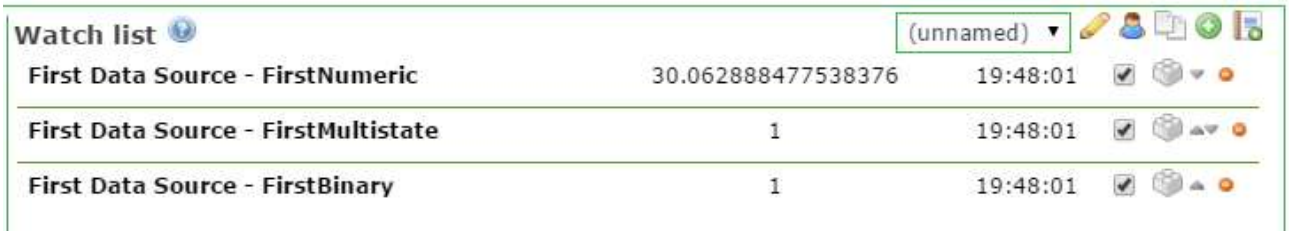
3.3 Viewing data: Watch Lists and Charts

Now that you have configured and registered your system's data sources and data points, you can view them:

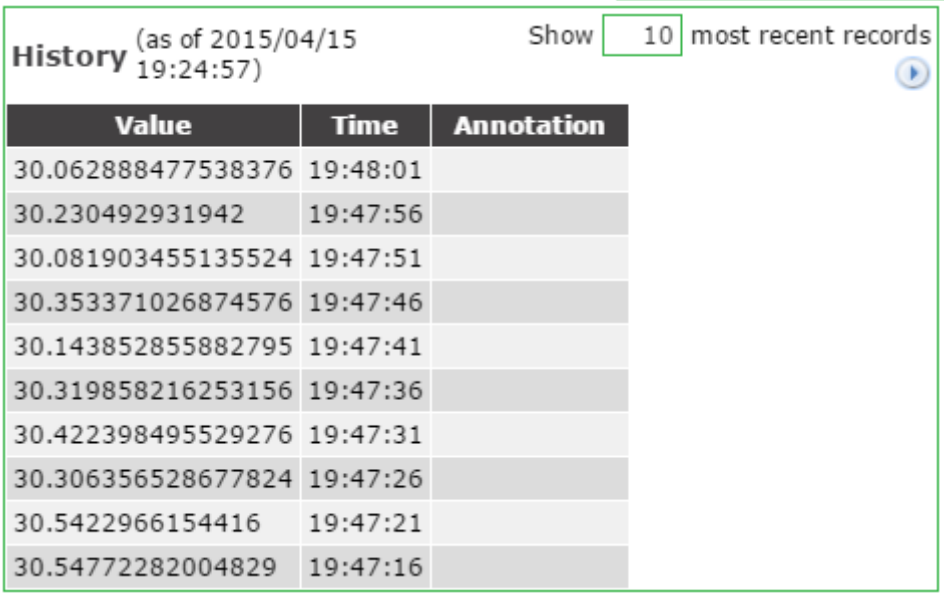
From the main menu, choose *Watch List* .

The registered data points are to the left.

To monitor your values, click on each of them to add them to the Watch list.



You can access and edit more information about each Data point by clicking on Point details . and watch your point's numeric and graphic history and its values.

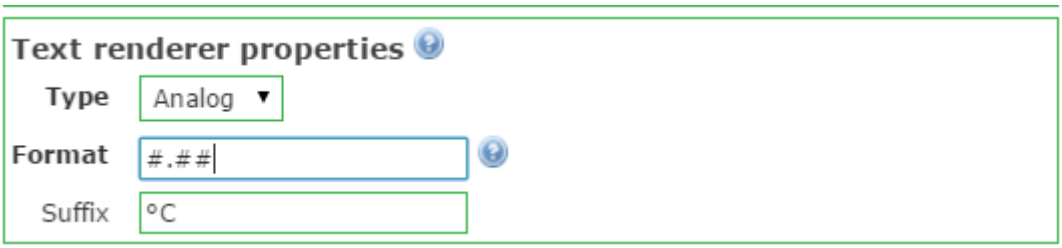


Value	Time	Annotation
30.062888477538376	19:48:01	
30.230492931942	19:47:56	
30.081903455135524	19:47:51	
30.353371026874576	19:47:46	
30.143852855882795	19:47:41	
30.319858216253156	19:47:36	
30.422398495529276	19:47:31	
30.306356528677824	19:47:26	
30.5422966154416	19:47:21	
30.54772282004829	19:47:16	

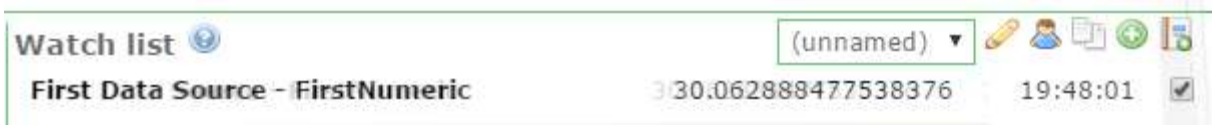
You can edit the properties of your Data point.

In your watch list, click the *Point details* then, on the next screen, click *Edit data point* .

Here you find different options. For example you can change the Text renderer properties, with the settings shown below:



This will change the rendering of the Data point from:

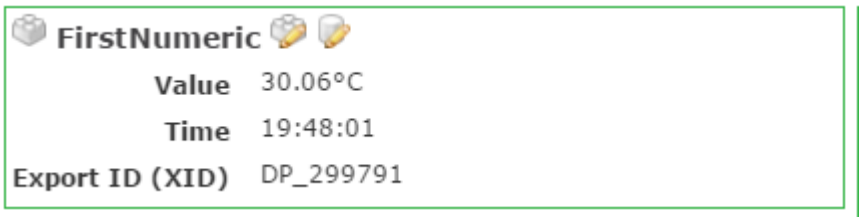


To:

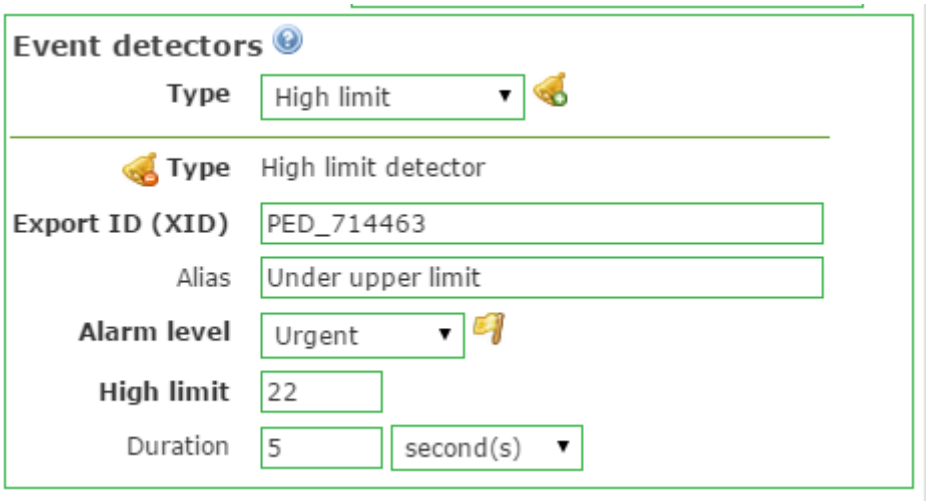
3.4 Defining Events

Now we will define an event.

In your watch list, click the *Point details* then, on the next screen, click *Edit data point* .

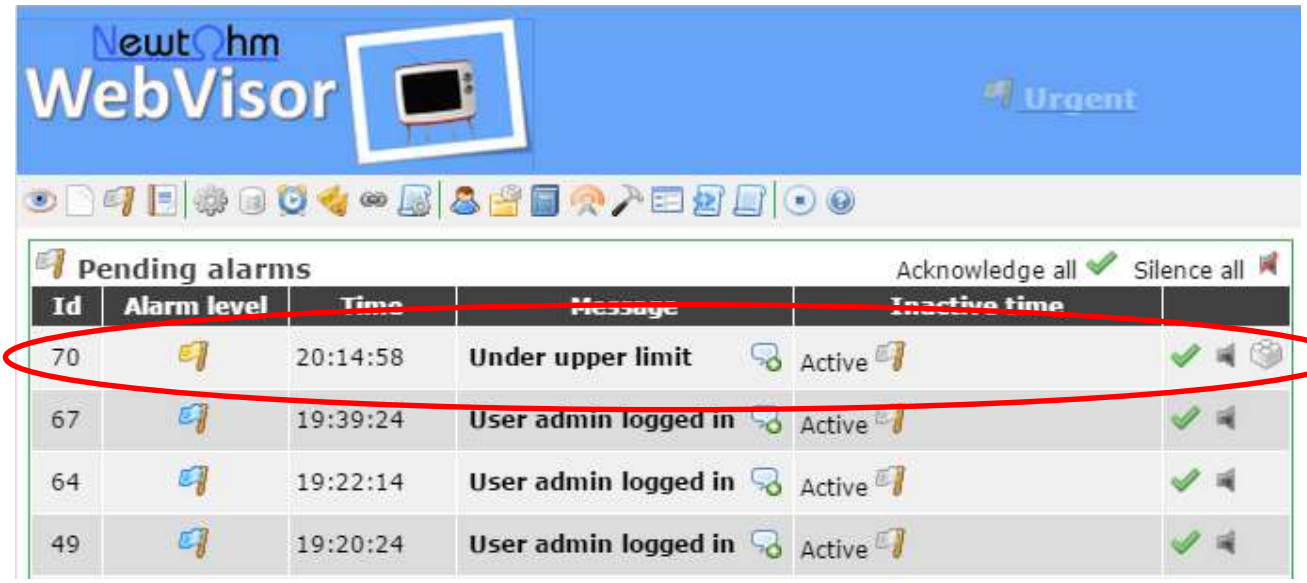


In the list of ***Event detectors*** on the right of the page, select "Low Limit" and click *Add* to add a new detector.



Fill in the fields as below and click the ***Save*** button at the bottom left of the screen to save your settings.

If you entered the fields as above the system will now inform you with on screen "Alarms" in the main menu, whenever the value of your data point falls below the minimum limit (22) for more than 5 seconds.



3.5 Graphical Views

3.5.1. Overview

To create more elaborate views of the data, we can build "Graphical Views". All attributes of a graphical view and its associated points are editable in the graphical view edit page. Views contain a collection of view components, which can be static content or dynamic content based upon one or more points.

3.5.2. Graphical view attributes

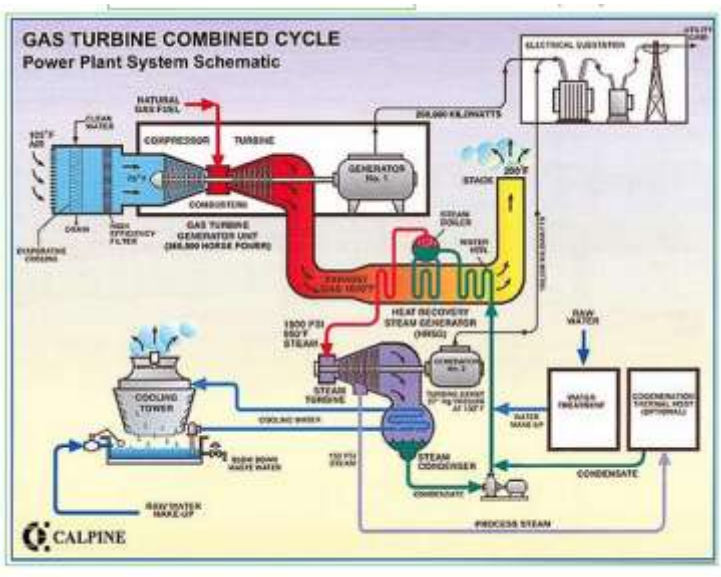
All graphical views have a ***Name*** by which they are referred within lists of graphical views. This name should, but is not required to, be unique.
A view may optionally have a ***Background image*** upon which its components are superimposed. Users can use any type of image displayable within a browser for the background image, including GIF, JPEG, and PNG.
The ***Anonymous access*** attribute determines whether users who have not logged in may access the view.

- *Edit* means that anonymous users can set the values of settable points in the view.
- *Read* means that anonymous users can access the view, but not set point values.
- *None* means that anonymous users may not access the view.

In this example we will use a file called "powerplant.jpg" (just go to google/images website and search for powerplant schematic).

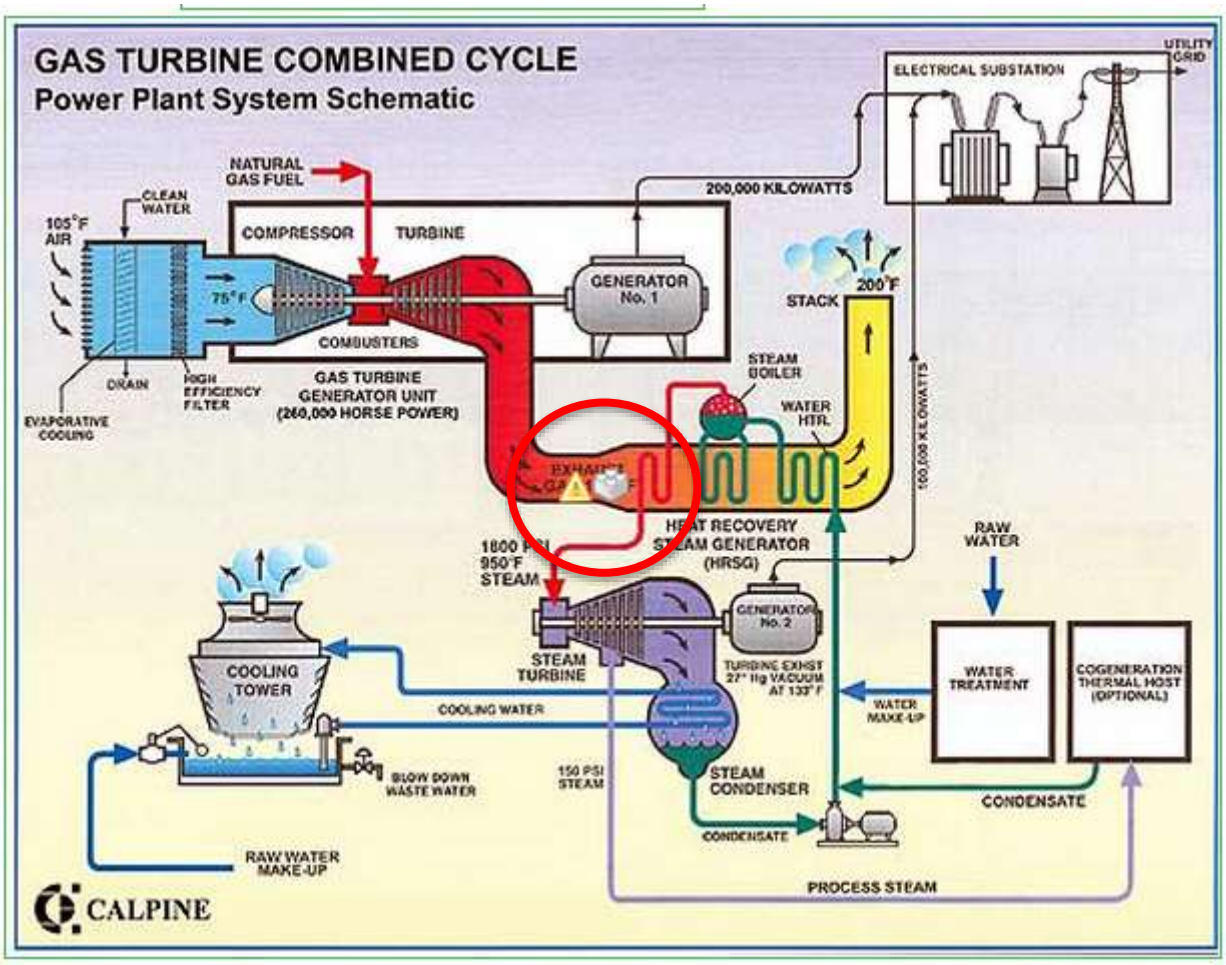
From the main menu, choose *Graphical views* .

- Next click *New view*() .
- In ***View properties*** enter a Name.
- Next click on Choose File, select your image (powerplant.jpg), confirm and click ***Upload Image***.



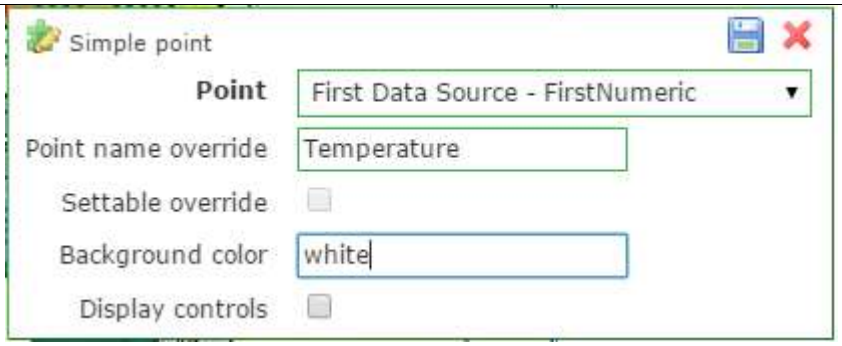
Save Delete Cancel

- In the ***Components*** list, select "simple point" and click *Add component to view* to enable it.
- You will see now a new icon() added in the middle of the "Graphical view".



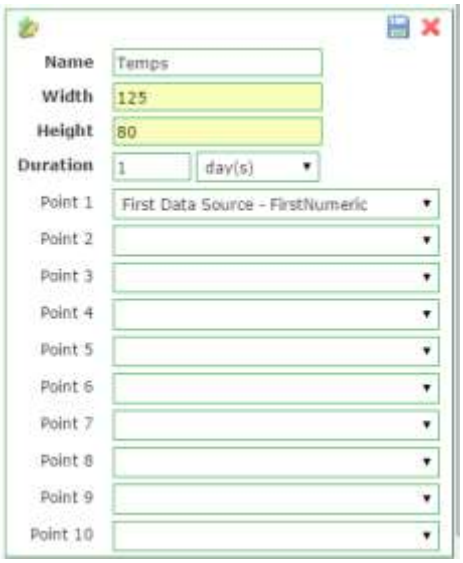
- Navigate with your mouse to the new icon, and click on the second option *Edit point component settings*.

In ***Point*** choose one of your Data points from the list and click on *Save*() to save your settings.

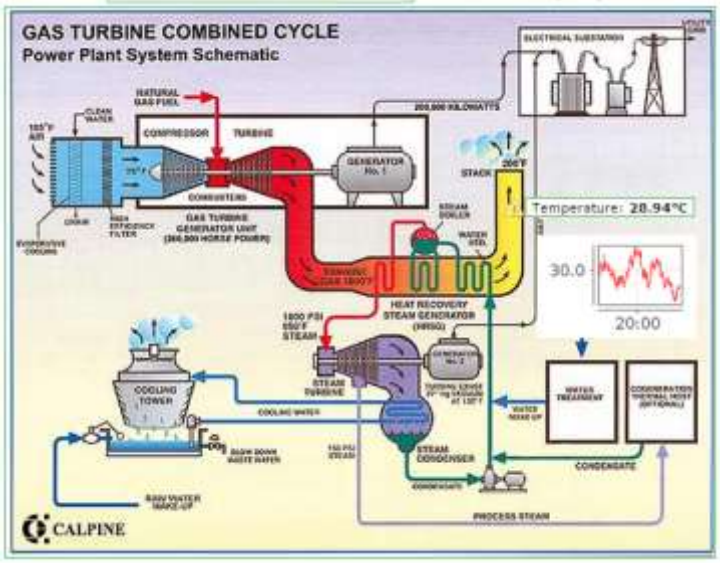


Position your newly added "Component" on the screen by dragging it with your mouse.

Now add an *"Image Chart"* to the graphical view. In the edit window set the values as in the following image.



Finally click the ***Save*** button at the bottom of the screen to save your settings.



4 DATA SOURCES

4.1 Modbus

4.1.1. Overview

Modbus is a data source from a Modbus device. Two types of Modbus devices are supported **Modbus IP** and **Modbus serial**.

- The **Modbus IP** data source is used to gather data from Modbus equipment accessible over an I/P network. Equipment can be in a local network or intranet, or could also be anywhere in the internet. This is a polling data source.
- The **Modbus serial** data source is used to gather data from a local Modbus network, accessible via RS232 or RS485 (requires adapter) communication. This is a polling data source.

The access control mechanism is *client-server* for **Modbus IP** or *master-slave* for **Modbus Serial**.

4.1.2. Basic Configuration

To add a Modbus Data source see chapter *3.2. Adding Data Sources and Data Points* and select from the list *Modbus IP* or*Modbus Serial*.

- Every data source requires a ***Name***. A Name can be any description.
- The ***Update period*** determines how often the Modbus equipment is polled for data.
- Checking ***Quantize*** will cause the data source start-up to be delayed so that pollings occur at a "rounded" point in time (e.g. if polling is every 10 seconds, polls will occur on the 10th second, the 20th second, etc). Otherwise, polling will begin immediately.
- The ***Timeout*** and ***Retries*** fields determine the system behavior in the case of a polling failure. The data source waits the given number of timeout milliseconds for a response from the network. If it is not received the request will be retried the given number of times.
- The ***Contiguous batches only*** check-box can be used to specify that the Modbus implementation should not attempt to optimize disparate value requests into a single request. Checking this field will cause the implementation to only make requests for multiple values when those value form contiguous register space.
- The ***Create slave monitor points*** check-box indicates whether slave monitor points should automatically be created by the data source. A slave monitor point is a binary point that indicates the current status of a slave. If a polling request to a slave fails due to timeout or error, the slave is considered off line. These points can then be used for control just like other points.

- The ***Max read bit count***, ***Max read register count***, and ***Max write register count*** count fields control how large batch requests can be. The Modbus specification provides defaults for these values, but some hardware may impose different limits.

4.1.3. Modbus IP properties

Three ***Transport types*** are supported. See the documentation for you equipment to determine which setting can be used.

- The *TCP* setting uses a new TCP socket for each poll. (The given retry value applies when connection exceptions occur while opening the socket. The timeout used for connection depends upon the TCP stack being used). This setting is useful if the equipment is polled infrequently.
- The *TCP with keep-alive* setting creates a TCP connection upon the first poll, which is left open for reuse. If the connection is closed for any reason, a new one is created when needed. Connection exception behavior for this option is the same as that for TCP. This is the recommended setting for most users since it provides efficient and robust communication while avoiding many configuration issues typical to UDP.
- The *UDP* setting uses UDP packets for communication. This setting provides the greatest network efficiency, but typically requires more network configuration since both WebVisor and the equipment need to be visible on the network (as opposed to TCP where WebVisor can be behind a firewall).

The ***Host*** and ***Port*** settings determine how to find the Modbus equipment on the network. The host can be a domain name or an IP address.

4.1.4. Modbus Serial properties

- Serial communication is controlled with the ***Baud rate***, ***Flow control in***, ***Flow control out***, ***Data bits***, ***Stop bits***, and ***Parity values***.
- The ***Echo*** setting can be used with RS485 networks as appropriate.
- The ***Encoding*** value determines how Modbus requests are formatted. Most production hardware uses RTU formatted messages. See the Modbus documentation for your equipment to determine how to set this field.
- The ***Concurrency*** value determines how Modbus requests are synchronized.
- A value of ***Transport*** means that any request must complete before the next can begin.
- ***Slave means*** that any request to a slave must complete before another for the same slave can begin, but requests to separate slaves can occur concurrently.
- ***Function*** means that any request of a particular slave/function must complete before another with the same slave/function can begin. The Function value allows the highest level of concurrency possible in Modbus serial, but may cause data corruption in some networks depending on duplexing.

4.1.5. Modbus point locator properties

Both serial and IP networks use the same point attributes to locate values. The ***Slave id*** is the id with which the Modbus node was configured; it is a number between 1 and 240.

The ***Register range*** determines in which of the four ranges the value is to be found. Consult the documentation for your equipment to determine what should be used.

- *Coil status* represents the hexadecimal range 0x00000 to 0x0FFFF. Each register contains a single, writable bit. Values in this range are always Binary.
- *Input status* represents the hexadecimal range 0x10000 to 0x1FFFF. Each register contains a single, read-only bit. Values in this range are always Binary.
- *Holding register* represents the hexadecimal range 0x40000 to 0x4FFFF. Each register is a 2 bytes (or a "word"), and is writable. Values in this range can be Binary or Numeric depending upon further settings.
- *Input register* represents the hexadecimal range 0x30000 to 0x3FFFF. Each register is a 2 bytes (or a "word"), and is read-only. Values in this range can be Binary or Numeric depending upon further settings.

Coil status and *Input status* values are always Binary. However, Modbus vendors are often very creative in the ways that *Holding* and *Input registers* are used. The ***Modbus data type field*** reflects the many ways in which data can be encoded. Consult the documentation for your Modbus equipment to determine the proper setting.

Specific values are located with the ***Offset*** setting. This is a 0-indexed value, meaning that counting starts from 0. Some Modbus vendors provide documentation that is 1-indexed, where counting starts from one. As such, it is sometimes necessary to subtract one from documented index to determine the 0-indexed offset. When registers include their range, e.g. written as 0x30001, 1-indexing is typically implied.

The ***Bit*** field is used when binary values are encoded into individual register bits.

The ***Settable*** field can be used to make a point that would normally be settable (according to its range) not settable.

The ***Multiplier*** and ***Additive*** fields can be used when trivial value conversions are required. Numeric values read from the network are calculated as follows: (raw value) * multiplier + additive. The reverse is applied when a numeric value is written to the network.

4.1.6. Modbus Node Scanning

Networks can be scanned for slave nodes using the scan utility. This feature iterates from slave id 1 to 240, sending each a 'ReadExceptionStatus' (function code 7). If a response is received (using the given timeout and retries settings), the node is considered available.

Note: Not all equipment supports this function code, so false negatives are possible.

5 REPORTS

5.1 Overview

Reports in WebVisor have two representations:

- **templates** which define the content of a report, and
- **instances** which are the result of running a report with a given template.

Simply: the report engine creates instances from templates. A template has attributes like a name, a list of points that it includes, and the date range of values it extracts. An instance will inherit its name from the template, but have its own attributes such as the time it ran, the run duration, the number of records it includes, and, of course, the data.

An instance can be considered a snapshot of the data at the time it was run. Instance attributes cannot be changed. Also, the data that the instance reports exists as long as the instance exists. (I.e. it survives the purging of the source data.) For this reason, it is important to purge report instances—especially large instances—when they are no longer needed since they can consume large amounts of storage.

Report templates—and the instances they create—belong to the user that created them. They cannot be explicitly shared within the system, but may be shared implicitly. See the *"Report templates"* documentation for more information.

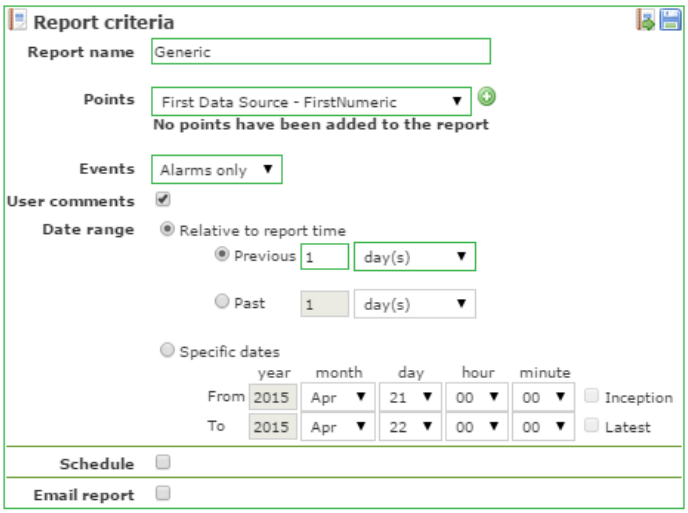
In this section we will discuss the ways howto generate reports from WebVisor.

5.2 Generating Reports in WebVisor

5.2.1. Setting up a new Report template

Report templates provides the definition of how to create report instances. For a description of templates vs. instances see the "Report instances" documentation.

From the main menu, choose Reports . To add a new report template, click the icon. To edit an existing template, click the template's name in the *"Report templates"* list.



- The ***Report name*** is used to visually reference the template. It is recommended that a unique name be used for each detector, but it is not required.
- Use the ***Points*** list to select the points to include in the report. To add a point, select it from the list and click the  icon. To delete an existing point, click the  icon associated with the point.
- The ***Date range*** is used to determine what values to select for the report. The range can be relative or absolute. In most cases a relative range is appropriate. Relative date ranges can be either based upon "previous" data or "past" data. A past range includes the given time period ending now. A previous range also includes the given time period, but its end time is quantized to correspond to the period type.

For example, if the period type is "Hour(s)" and number of periods is 1, and the report runs at 18:05, the time span that will be used is from 17:00 (inclusive) to 18:00 (exclusive). If the number of periods were, say, 3, the time span would be from 15:00 to 18:00. Similarly, "Month(s)" starts the time span at midnight on the first day of the previous month and ends it on the last day of the previous month (when the number of periods is 1). Other period types work the same. A week starts on Monday at midnight in accordance with ISO standards.

5.2.2. Report scheduling

Reports can be **scheduled** to run automatically. Use the **Run every...** selection to determine a simple time event upon which it should run, or define a cron pattern for more specific control. (See the "Cron patterns" documentation for more information.) Time events occur at the beginning of the given period. Weeks start on Monday in accordance with ISO standards. A **Run delay** can be applied if data expected on the report tends to be collected later than the absolute run time.

Scheduled reports for disabled users do not run.

5.2.3. Report emailing

Although reports cannot be explicitly shared with the system, the can be implicitly shared by creating a mailing list to which to send the generated report instance. The content of the email is the same as the "report chart" window opened from within the Report queue panel. To include the CSV export file with the email, check the ***Include tabular*** data box.

Select the **Email recipients** to which to send the report email. Recipients can be mailing lists, system users, or free-form email addresses. Click the  icon to send a test message to the selected recipients.

Important: Report instances that are emailed are automatically deleted immediately after being sent.

5.2.4. Template management

To save a report template click the  icon at the top right of the Report criteria panel. To delete an existing template, click the  icon. To run a report template immediately, click the  icon.

- Note that some report templates may include a great deal of information, and so may take a long time to run.
- As such, all reports are run asynchronously to user interface. When a the () icon is clicked, the report instance will be added to the report queue where the user can monitor its progress.
- To make a copy of a report template, click the () icon.

5.2.5. Report Queue

The Report queue lists all of the report instances for the current user. This list does not automatically refresh itself. A manual list refresh can be triggered by clicking the  icon. Certain events, however, will also trigger a list refresh, such as running a report.

Report instances in any state will be listed, whether they be completed or still running. The columns of the list will reflect the current state of the instance.

The columns of the list have the following definitions:

- *Report name:* the name inherited from the report template. Once the instance is created the report name does not change even if the name of the template is changed.
- *Run time start:* the time that the report engine began creating the report instance
- *Run duration:* the amount of time it took to create the instance
- *From:* the time from which report records were selected (inclusive)
- *To:* the time until which report records were selected (exclusive)
- *Records:* the total number of records selected

The **Do not purge** checkbox allows users to prevent important report instances from being deleted by the scheduled purge process. (See the "Other settings" documentation for more information on report purging.) This feature should only be used when necessary since report instances can potentially consume large amounts of storage.

The final column in the table provides controls for managing instances. The () icon initiates the download of a CSV file of the report's data, suitable for loading into spreadsheet software. The () icon opens a new browser window displaying report instance information, data statistics, and an image chart of the report data. Clicking the () icon deletes the report instance.
Note that large report instances may take some time to delete.